4

Problem Analysis : Concepts and Techniques

Problem Analysis

- Definition: the process of understanding the real-world problems and users needs and proposing abstract solutions to those problems.
- Goal: gain a better understanding, before development begins, of the problem to be solved.
- Avoid to jump to conclusions by identifying the root cause of the problem.
- Identify the sources of information for system analysis.

Five steps of problem analysis

- Step1 : Gain agreement on the problem definition
  - □ Write a simple and clear definition of the problem description
  - □ Establish an order of importance for all features of the system
  - □ Come to an agreement with all stakeholders
  - □ Resolve conflicts by negotiation

Five steps of problem analysis

- Step 2 : Identify the root causes of the problem
  - □ Make sure that the problem identified is the real problem
  - □ Sometimes, a problem hides other more important problems
  - □ Addressing the wrong problem leads to failure
  - □ A problem can have several causes:
    - Some might be eliminated by non-software solutions
    - Some might need contradictory solutions
    - More than one solution might be needed
  - □ This part of the analysis requires input from extremely knowledgeable, insightful and experienced persons.

Five steps of problem analysis

- Step 3 : Identify stakeholders and users
  - □ Stakeholder: anyone who could be affected by the new system or has input to provide in the implementation of the new system
  - □ Complex problems always involve the input of different stakeholders that have different viewpoints on the problem.
    - Users: will use the system
    - Managers: will pay for the system, or will manage the users
    - IT people: will install, manage and maintain the system
    - External regulators: will impose constraints on the system operation
    - System developers: will implement a solution to the problem
  - □ Forgetting one of these might lead to major rework later on, or even to project failure.

Five steps of problem analysis

- Step 4 : Define the system boundary
  - □ Any software system has to interact with its environment
  - □ System boundary describes an envelope in which the solution is contained.
  - □ System is divided as:
    - The system itself and its functionalities
    - The things (outside the system) that interacts with the system
  - □ Actors:
    - Supplies, uses, or modifies the information in the system
    - Someone or something, outside the system, that interacts with the system
  - □ Later on, this early information will direct how the system interfaces will be defined.

Five steps of problem analysis

- Step 5 : Identify the constraints on the system
  - Constraint : a restriction on the degree of freedom we have in providing a solution
  - They are as important as requirements : they direct what the system should not do, or what the system should not be.
  - Table 4-4, p45.

Summary

- After that, we have :
  - A good, general understanding of the problem and its causes
  - Identified the stakeholders whose collective input and judgment will determine the nature of the system
  - A notion of the boundary of the system and its interface with the exterior
  - An understanding of the constraints imposed on the system

Problem Analysis : Concepts and Techniques

Business Modeling

Business Modeling

- Definition: a problem analysis technique especially suitable for the IS/IT environment.
- Goal: help define systems and their application domains.
- Purpose: twofold
  - To understand the structure and dynamics of the organization
  - To ensure that customers, end users, and developers have a common understanding of the organization

Business Modeling

- Apply Software Engineering Techniques to Business Modeling
  - □ Using the same techniques or very similar techniques for both business domain and software domain
  - □ Using UML concepts

Business Modeling Using UML

- Business Use-Case Model
- Business Object Model

Business Models

- Business Use-Case Model
  - A model of the intended functions of the business
  - Consists of actors and the use cases
  - Describes who (or what other system) is involved in this business activity and how this activity takes place
  - Could represent a preliminary version of the use case diagram as developed in the Use-Case driven approach
  - Its primary goal is to show how things are working now, not what the system should be

Business Models

- Business Object Model
  - Describes the entities and how they interact to deliver the functionality to realize the business use cases
  - Entities:
    - Business workers: users, other systems
    - Business entities: anything that business workers produce or use in their business activities
  - Could represent a preliminary version of the object diagrams (sequence and class diagrams) as developed in the Use-Case driven approach

Business Models

- Taken together, the business use-case model and object model:
  - □ Provide a overview of how the business works;
  - □ Allow the developers to focus on the areas in which systems can be provided;
  - □ Help the developers to understand what changes in the business process will have to take place.

Business Modeling

- Business modeling clearly fits in the use-case driven approach
- It provides a first overview of the problem domain.
- It forces a first draft using simple terms that belong to the problem domain:
  - □ Forces early stakeholder implication
  - □ Forces problem domain understanding by the software developers
- Question: How can these models be integrated in the use case driven approach, or to an object-oriented design methodology?
- Translations from business models to the system model
  - □ business workers -> actors
  - □ behaviors of the workers -> system use cases, functionality, scenario
  - □ business entities -> entity classes

Business Modeling

- When to use business modeling?
  - □ The application environment:
    - Complex (requires problem domain analysis)
    - Multidimensional (several sub-problems are concerned)
    - Many people are directly involved in using the system (user centered application)
  - □ Not for every software engineering effort

Summary

- By discussing the business modeling, we defined:
  - □ Why you might need to model the business
  - □ How to use UML for business modeling
  - □ business modeling, the business use-case model, and the business object model
  - □ How you can define software applications and derive software requirements from models of the business.

Problem Analysis : Concepts and Techniques

System Engineering

Systems Engineering

- Systems engineering provides eight principles (INCOSE 1993)
  - □ Know the problem, know the customer, and know the consumer.
  - □ Use effectiveness criteria based on needs to make the system decisions.
  - □ Establish and manage requirements.
  - □ Identify and assess alternatives so as to converge on a solution.
  - □ Verify and validate requirements and solution performance.
  - □ Maintain the integrity of the system.
  - □ Use an articulated and documented process.
  - □ Manage against a plan.

Systems Engineering

- Requirements Flowdown
  - □ Assigning a system-level requirement to a subsystem.
  - □ A matter of ensuring that all system requirements are filled by a subsystem somewhere or by a set of subsystems collaborating together.

Systems Engineering

- The initial requirements of the system (system level requirements) are normally very high level and abstract.
- System decomposition will also decompose these high level requirements into subsystem level requirements.
- Derived requirements:
- Two subclasses of Derived Requirements
  - □ Subsystem requirements
    - must be imposed on the subsystems
    - do not necessarily provide a direct benefit to the end user
  - □ Interface requirements
    - may arise when the subsystems need to communicate with one another to accomplish an overall result.
- The propagation of requirements and levels of requirements derivation increases the complexity of requirements management

Systems Engineering

- Systems complexity has moved from hardware to software components. Why?
  - □ Cheaper, easier to change, lighter, etc.
- Nowadays, software, not hardware
  - □ will determine the functionality of the system
  - □ will determine the success of the system
  - □ will consume the majority of the costs of research and system development
  - □ will absorb most of the changes that occur during development
  - □ will be evolved over the next few years to meet the changing needs of the system
- The great majority of systems requirements are now software requirements, even though these are still hardware systems

Systems Engineering

- Tips for doing a good job
  - □ Develop, understand, and maintain the system-level requirements and use cases.
  - □ Do the best possible job of partitioning and isolating functionality within subsystems (minimize requirements relationships).
  - □ Develop software for the system as a whole if possible.
  - □ Use common code on both sides of the interface when coding the interfaces: promote software reuse.
  - □ Define interface specifications that can do more than would be necessary to simply meet the known conditions.

Problem Analysis Summary

- Various techniques can be used in problem analysis
  - Use-case driven approach
    - A general technique based on OO technology
  - Problem analysis
    - A general method used to gain a global understanding of the problem
  - Business modeling
    - To build a model of business infrastructures
  - Systems engineering
    - To analyze embedded systems (software controlling/using hardware)

Use case driven approach

- Process :
  - Inception Phase:
    - Project description agreement
    - Project risks
    - Context of the project
    - Scope of the project
  - Elaboration Phase:
    - Detailed definition of all use cases
    - UML diagrams modeling scenarios
    - Use case diagram(s)

Problem analysis

- Process :
  - Gain agreement on the problem definition
  - Understand the root causes of the problem
  - Identify the stakeholders and users
  - Determine the boundaries of the solution
  - Understand the constraints

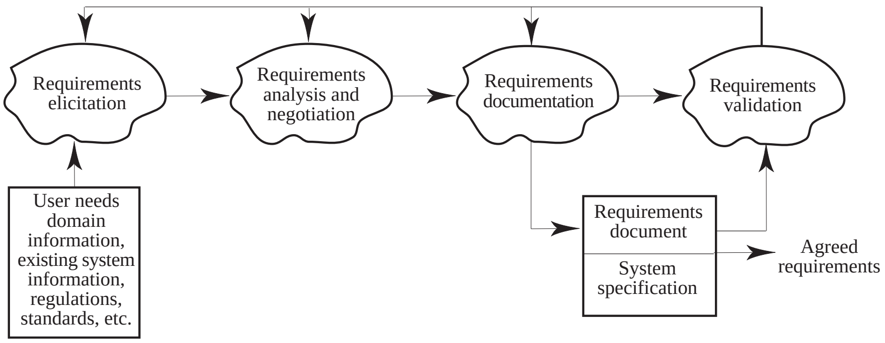
Business modeling

- Business Use-Case Model
  - Describes who (or what other system) is involved in this business activity and how this activity takes place
  - Could represent a preliminary version of the use case diagram as developed in the Use-Case driven approach
- Business Object Model
  - Describes the entities and how they interact to deliver the functionality to realize the business use cases
  - Could represent a preliminary version of the object diagrams (sequence and class diagrams) as developed in the Use-Case driven approach

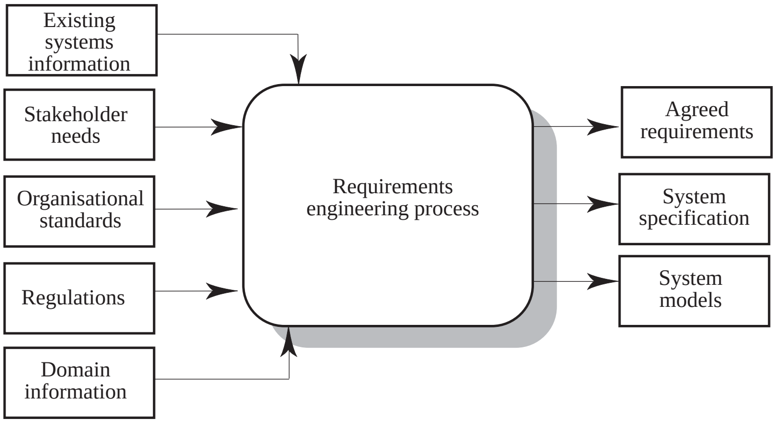
Systems engineering

- Composition and decomposition process is very important
- Requirements also have to be composed and decomposed as the system is specified
- Subsystem interfaces have to be clear and flexible
- Development process has to take into account the physical constraints of the apparatus in which it is embedded

They all fit in our general process


Requirements elicitation
Requirements analysis and negotiation
Requirements documentation
Requirements validation
User needs domain information, existing system information, regulations, standards, etc.
Requirements document
System specification
Agreed requirements

They all fit in our general process



Problem Analysis : Concepts and Techniques

The UML
Use-Case-Driven Approach to Requirements Engineering

UML vs. Requirements Modeling

- UML: Unified Modeling Language
- A software analysis and design methodology mainly based on diagrams
- Requirements Modeling in UML:
  The Use-Case-Driven Approach
- Use cases are used to describe the externally visible requirements of a system
- They can be used later on in system design
- Developed by Booch, Jacobson and Rumbaugh of Rational Software (www.rational.com)

The Process

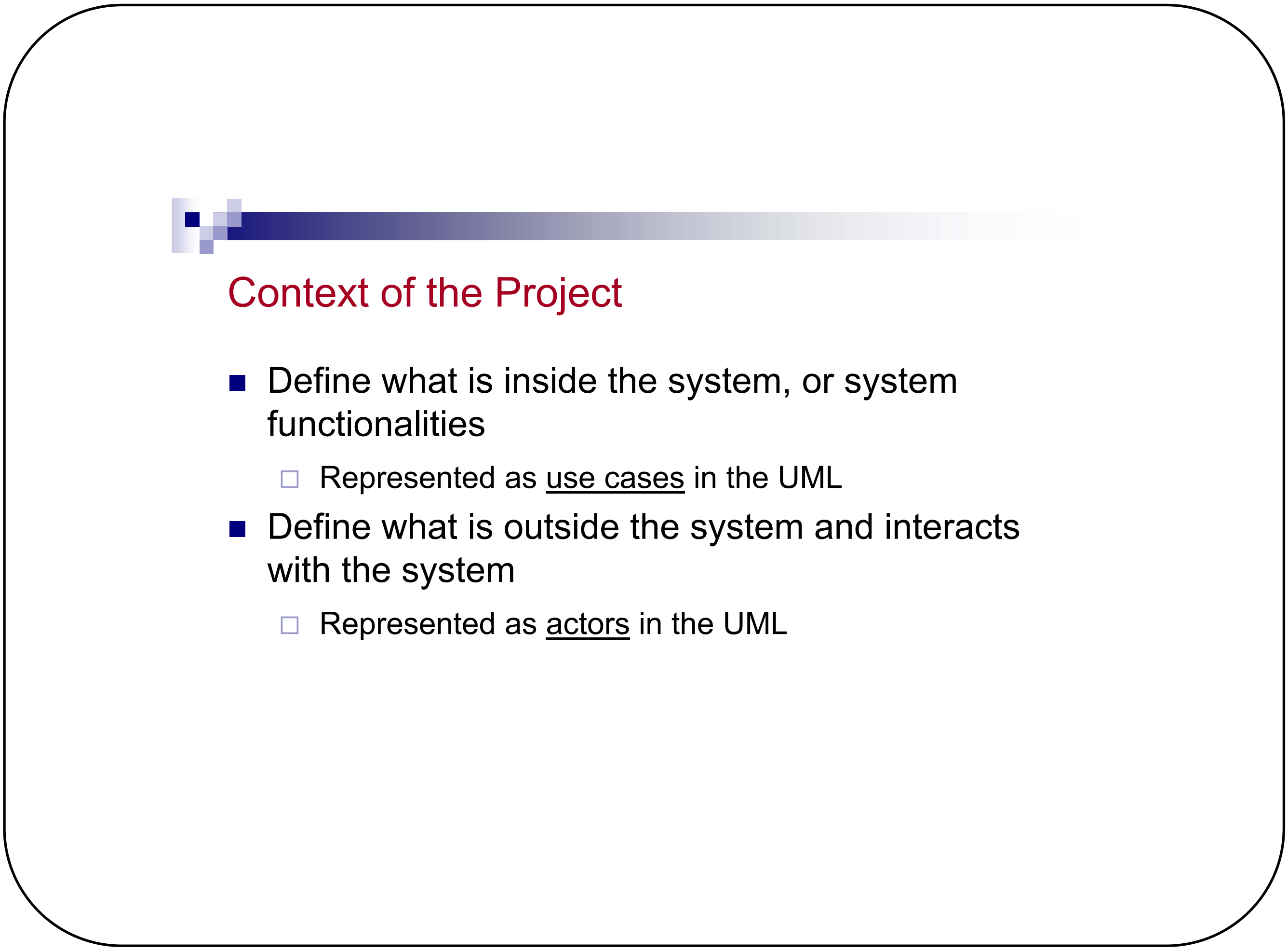
- Inception Phase:
  - □ Project description agreement
  - □ Project risks
  - □ Context of the project
  - □ Scope of the project
- Elaboration Phase:
  - □ Detailed definition of all use cases
  - □ UML diagrams modeling scenarios
  - □ Use case diagram(s)

Problem Analysis : Concepts and Techniques

Inception Phase

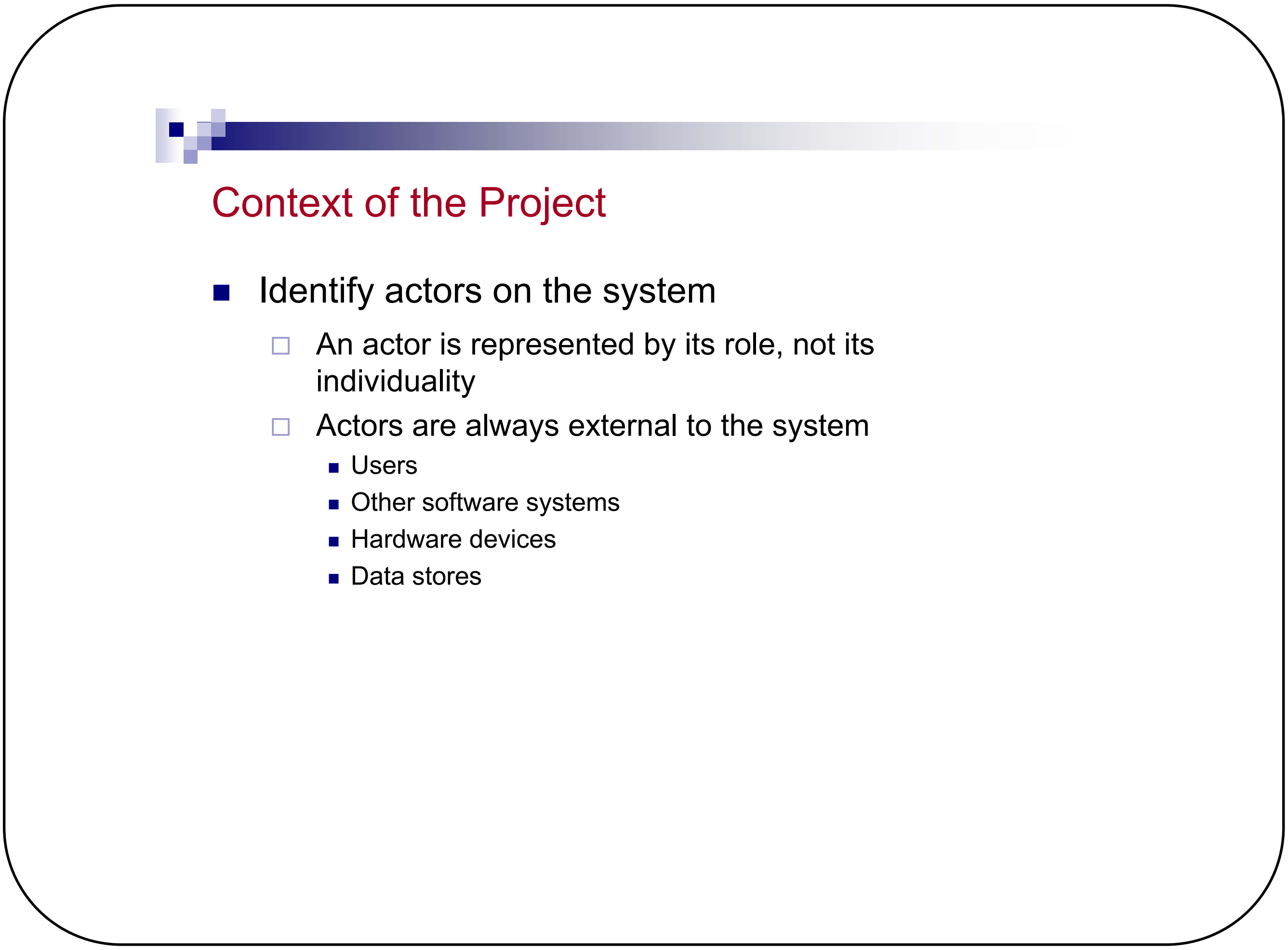
Project Description

- Project description agreement
  - □ Identify the problem and its root causes
  - □ Write a short textual description of the problem to be solved, and the key features of the system
  - □ Should not describe solutions
  - □ From a paragraph to a couple of pages for a complex project
  - □ Every stakeholder has to agree on the project description
- Project risks
  - □ Look at the system from many viewpoints
    - Other systems, marketing, technology, users, managers
  - □ Identify things that can go wrong
    - User resistance, inexperienced developers, system dependencies



Context of the Project

- Define what is inside the system, or system functionalities
  - Represented as use cases in the UML
- Define what is outside the system and interacts with the system
  - Represented as actors in the UML

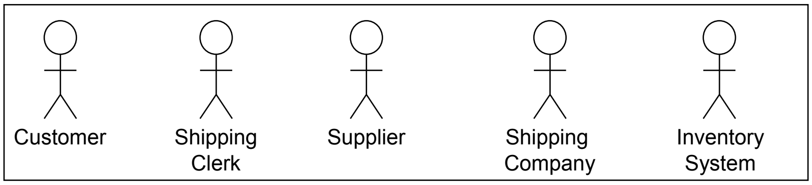


Context of the Project

- Identify actors on the system
  - An actor is represented by its role, not its individuality
  - Actors are always external to the system
    - Users
    - Other software systems
    - Hardware devices
    - Data stores

Context of the Project

- Describe actors
  - □ Customer: a person who orders products through the system.
  - □ Shipping company: UPS, FedEx, DHL.
  - □ Shipping clerk: user of the system who packages, labels and ships orders.
  - □ Inventory system: software that tracks the company inventory.

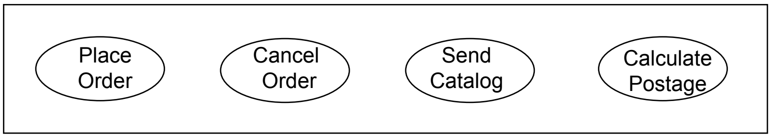


Context of the Project

- Identify use cases
  - □ What are the services used by the actors?
  - □ Who stores, accesses or deletes information in the database?
  - □ Startup, shutdown, diagnostics, installation
  - □ Maintenance
- Go through all the actors and identify how they can use the system

Context of the Project

- Order-processing use cases
  - □ Customer: place order, send catalog, get status on order, return product, cancel order, register complaint
  - □ Shipping clerk: print mailing labels, calculate postage
  - □ Inventory system: give product information, update product quantities



Place Order

Cancel Order

Send Catalog

Calculate Postage

Scope of the Project

- Estimate what could realistically be implemented considering factors such as:
  - □ Time frame available
  - □ Budgetary envelope
  - □ Physical resources available
- The system description, risk analysis and assumptions must be met
- End of the inception phase
- Next step: adding details and structure

Problem Analysis : Concepts and Techniques

Elaboration Phase

Define Use Cases

- Use case: A coherent unit of externally visible functionality provided by a system unit.
- Used to define a behavior without revealing the internal details.
- A use case describes what the system does, not how it does it.
- Scenario: flow of events describing how a use case is realized.
- Each use case has a primary scenario.
- Eventually also has a set of alternate scenarios.
- Pre-conditions and post-conditions are stated.

Define Use Cases

Place Order
Pre-conditions:
A valid user has logged into the system
Primary Flow of Events:
1. (start) The customer selects Place Order
2. The customer enters its address
3. The customer enters the product codes it wants to order
4. The system provides the items description and prices, and a running total
5. The customer enters its credit card number
6. The customer clicks on submit
7. The system validates the information, saves the order and forwards the transaction request to the accounting system
8. (end) When the payment is confirmed, the order is marked as paid

Alternate Flow of Events 1:
In step 7, the system prompts the user to correct any incorrect information
Alternate Flow of Events 2:
In step 8, if the transaction is refused by the bank, the order is marked as pending
Post-conditions:
The order has been saved in the database

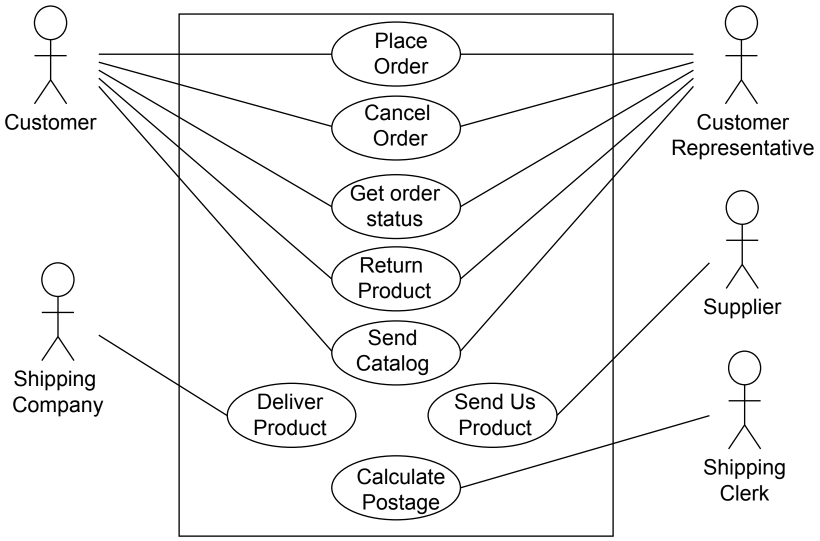
Scenarios: Diagrams

- Complex scenarios are better expressed using diagrams.
- The UML provides two kinds of diagrams:
  - Activity diagrams for a high-level description.
  - Sequence diagrams for more in-depth analysis.

Use Case Diagrams

- Roles
  - □ Model the context of the system. Define what are the actors that are external to the system
  - □ Model the requirements of the system. Define what the system should do from an external point of view

Order-Processing Use Case Diagram


Customer
Place Order
Cancel Order
Get order status
Return Product
Send Catalog
Deliver Product
Send Us Product
Calculate Postage
Customer Representative
Supplier
Shipping Company
Shipping Clerk