Outline

1. What is use case analysis?
2. How to do use case analysis?
   • What is use-case diagram?
   • How to identify use cases?
   • How to write use cases?
3. Examples
4. Summary
What is use case analysis?
Where are we?

- **Plan**
- **Analysis**
  - Use Case Analysis
  - Data Flow Diagram
  - Logical Process Model
  - Logical Data Model
- **Design**
  - Architecture Design
  - Physical Process Model
- **Implementation**
  - Developing
  - Testing
  - Documentation
Overview of Use Case Analysis

A use case analysis is the primary form for gathering usage requirements for a new software program or task to be completed.

Primary Goals:
• designing a system from the user’s perspective
• communicating system behavior in the user’s terms
• specifying all externally visible behaviors
What is a Use Case?

A use case represents how a system interacts with its environment by illustrating the activities that are performed by the users and the system’s responses.

Use cases are a means of expressing user requirements.

Use cases are used extensively in the analysis phase.
Use Cases and the Functional Requirements

Use cases are very useful tools to us to understand user requirements. However, use cases only *convey the user’s point of view.*

Transforming the user’s view into the developer’s view by *creating functional requirements* is one of the important contributions of system analyst.

The derived functional requirements give more information to the *developer* about what the system must do.
Use case analysis

Use case analysis is important and useful technique not only in the analysis phase, but also in the design phase or even in the implementation phase!

Why?
The use-case model serves as a unifying thread throughout system development. It is used as the primary specification of the functional requirements for the system, as the basis for analysis and design, as an input to iteration planning, as the basis of defining test cases and as the basis for user documentation.
How to do use case analysis?
Two kinds of use case techniques

1. Visual Modeling
   Use case diagram: typically used in conjunction with the textual use case.

2. Textual Document
   Templates

Note:
While a use case itself might drill into a lot of detail about every possibility, a use-case diagram can help provide a higher-level view of the system.

It has been said before that "Use case diagrams are the blueprints for your system". They provide the simplified and graphical representation of what the system must actually do.
Use Case Diagram

A use-case diagram is used to graphically depict a subset of the use-case model to simplify communications.

Much of the use-case model is in fact textual, with the text captured in the use-case specifications that are associated with each use-case model element. These specifications describe the flow of events of the use case.
Use Case Diagram

1. A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case.

2. A use case diagram can **portray** the different types of users of a system and the various ways that they interact with the system.

3. A use case diagram **summarizes** all of the use cases (for the part of the system being modeled) together in one picture.

4. Typically, the use case diagram is **drawn early** on in the SDLC.
## Elements of a Use Case Diagram

<table>
<thead>
<tr>
<th>Term and Definition</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>An actor</strong></td>
<td></td>
</tr>
<tr>
<td>- Is a person or system that derives benefit from and is external to the system.</td>
<td></td>
</tr>
<tr>
<td>- Is labeled with its role.</td>
<td></td>
</tr>
<tr>
<td>- Can be associated with other actors by a specialization/superclass association,</td>
<td></td>
</tr>
<tr>
<td>denoted by an arrow with a hollow arrowhead.</td>
<td></td>
</tr>
<tr>
<td>- Is placed outside the system boundary.</td>
<td></td>
</tr>
<tr>
<td><strong>A use case</strong></td>
<td></td>
</tr>
<tr>
<td>- Represents a major piece of system functionality.</td>
<td></td>
</tr>
<tr>
<td>- Can extend another use case.</td>
<td></td>
</tr>
<tr>
<td>- Can use another use case.</td>
<td></td>
</tr>
<tr>
<td>- Is placed inside the system boundary.</td>
<td></td>
</tr>
<tr>
<td>- Is labeled with a descriptive verb–noun phrase.</td>
<td></td>
</tr>
<tr>
<td><strong>A system boundary</strong></td>
<td></td>
</tr>
<tr>
<td>- Includes the name of the system inside or on top.</td>
<td></td>
</tr>
<tr>
<td>- Represents the scope of the system.</td>
<td></td>
</tr>
<tr>
<td><strong>An association relationship</strong></td>
<td></td>
</tr>
<tr>
<td>- Links an actor with the use case(s) with which it interacts.</td>
<td></td>
</tr>
</tbody>
</table>
More Words on Actor

An actor might be a person, a company or organization, a computer program, or a computer system — hardware, software, or both.

They are not part of the system and are situated outside of the system boundary.

Actors may be both at input and output ends of a use case.
Relationships in Use Case Diagram

You can now model the interactions between the users and the system by creating the relationships between the actors and the use cases.

1. Association
2. Include (Use)
3. Extend
4. Generalization
Association Relationship

A Use Case diagram illustrates a set of use cases for a system, i.e. the actors and the relationships between the actors and use cases.

student

borrow books
Include Relationship

The include relationship adds \textit{additional functionality} not specified in the base use case. 

\texttt{<<include>>} is used to \textbf{include common behavior} from an included use case into a base use case in order to support re-use of common behavior.

\textbf{Note:}
An include relationship connects a base use case (i.e. borrow books) to an inclusion use case (i.e. check Fine). An include relationship specifies how behaviour in the inclusion use case is used by the base use case.
Extend Relationship

The extend relationships are important because they show *optional functionality* or system behavior. 

`<<extend>>` is used to include *optional behavior* from an extending use case in an extended use case.

Notes:

Notice the extend relationship between Request a book and Search. The extend relationship is significant because it shows optional functionality. If the student desires, he/she can search the book through the system. However, the student may only Request a book through the system without searching the book if the student knows the call number.
Generalization Relationship

A generalisation relationship means that a child use case inherits the behaviour and meaning of the parent use case. The child may add or override the behaviour of the parent.
Examples
An Example

Simple ATM Machine System

- System Maintainence
- System Reporting
- System Shutdown
- Bad PIN
- Login
- Print Receipt
- Deposit
- Check Balance
- Withdraw

Administrator (Third-party)
Bank Customer
Bank
Another Example
More Examples

Bad!
More Examples

Also Bad!
More Examples

Good one!
Steps in creating a use case diagram

1. Identify use cases.
2. Draw the system boundary.
3. Place the use cases on the diagram.
4. Identify the actors.
5. Add relationships.
How to identify use cases?

There are two ways to identify Use Cases:

• Using the actors
  o identify the actors related to a system or organisation
  o for each actor, identify the processes it initiates or participates in

• Using events
  o identify the external events that a system must respond to
  o relate the events to actors and use cases
Tips for use case diagram

• Always structure and organize the use case diagram from the perspective of the actor.
• Use cases should start off simple and at the highest view possible. Only then can they be refined and detailed further.
• Use case diagrams are based upon functionality and thus should focus on the "what" and not the "how".
How to write use cases?
Major Parts:
1. Preconditions
2. Normal Course
3. Alternative Courses
4. Postconditions
5. Exceptions
6. Summary
Alternative Use Case Formats

1. A full-dressed use case is very thorough, detailed, and highly structured.

2. The project team may decide that a more casual use case format is acceptable.
### Alternative template

#### Major parts:

1. **Preconditions**
2. **Normal course**
3. **Postconditions**
4. **Exceptions**

#### Preconditions:
- The LCA identity is authenticated.
- The LCA has necessary training and credentials on file.
- The Chemical Supply datastore is up-to-date and online.

#### Normal Course:
1. Request a lawn chemical from the chemical supply warehouse.
2. The LCA specifies a chemical needed and the quantity needed.
3. The system lists chemical and quantity on hand from Chemical Supply datastore.
   - If the quantity on hand is less than the quantity needed, the LCA specifies the quantity he will take.
   - Purchasing is notified of chemical shortage.
4. The system gives the LCA a Chemical Pick-up Authorization for the quantity requested.
5. The system notifies the Chemical Supply Warehouse of the chemical pick-up.
6. The system stores the Lawn Chemical Request in the Chemical Request datastore.

#### Postconditions:
1. The Lawn Chemical Request is stored in the Chemical Management System.
2. The Chemical Pick-up Authorization is produced for the LCA.
3. The Chemical Supply Warehouse is notified of the chemical pick-up.
4. Purchasing is notified of chemical outage.

#### Exceptions:
1. Chemical no longer approved for use (occurs at step 1).
   1. The system displays message: "That chemical is no longer approved for use."
   2. The system asks the LCA if he wants to request another chemical or to exit.
   3a. The LCA asks to request another chemical.
   4a. The system restarts Normal Course again.
   3b. The LCA asks to exit.
   4b. The system terminates the use case.
Steps in writing use cases

1. Identify the major use cases
2. Identify the major steps for each use case
3. Identify elements within steps
4. Confirm the use case
Tips for Writing Use Cases

1. Based on a goal.
   A use case describes how an actor uses the system to achieve a goal.

2. Complete or not complete.
   When an actor has performed the steps in a use case, the goal should be either 100% complete or 0% complete.

3. One person, one place, one time, one event.
   Try to write use cases that describe how one actor responds to one event in one place at one time.

4. Six to ten steps.
   Try to keep the main success scenario (aka primary flow) of a use case between six and ten steps. Use cases should make requirements easier to comprehend.
Summary

Focus on use case analysis with two types of techniques:

1. Use case diagram
   a. Components: actors, relationships, use case, system boundary
   b. Examples

2. Textual use case:
   a. Full dressed template
   b. Alternative template
Take-away messages

A use case contains all the information needed to build one part of a process model, expressed in an informal, simple way.

When writing a use case,
- identify the triggering event,
- develop a list of the major steps,
- identify the input(s) and output(s) for every step,
- have the users role-play the use case to verify.
Thank you!

Questions?