Lecture 8: Use Cases

Kenneth M. Anderson
Object-Oriented Analysis and Design
CSCI 6448 - Spring Semester, 2001

Goals for this Lecture

- Define Use Cases
- Review UML Notation for Use Cases
- Look at a variety of Use Case examples
  - from the book

    Writing Effective Use Cases
    by Alistair Cockburn
    ISBN: 0-201-70225-8

Use Case Terminology

- Use Case Model
  - consists of actors and use cases
- Actors
  - entities which interact with a system
  - Actors are different from users
    - An actor represents a role that a user can play
    - Actors are classes; Users are instances
    - Actors are unlike other objects in that their behavior is non-deterministic

Use Case Terminology

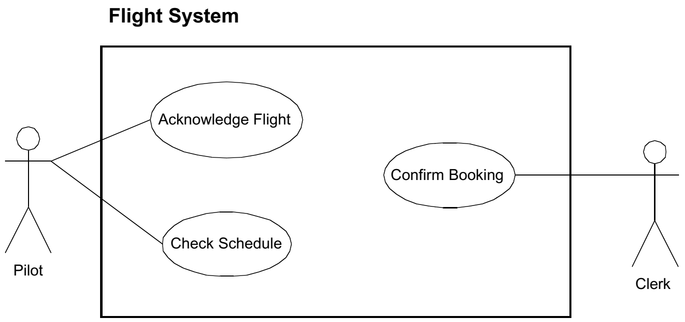
- Use Cases
  - An actor can carry out many different operations on the system
    - When a user performs an operation, he or she will perform "a behaviorally related sequence of transactions in a dialogue with the system"
      - This sequence is a use case
  - Use case descriptions are classes; an instance of a use case is created when an an actor initiates an operation

Use Cases as Requirements

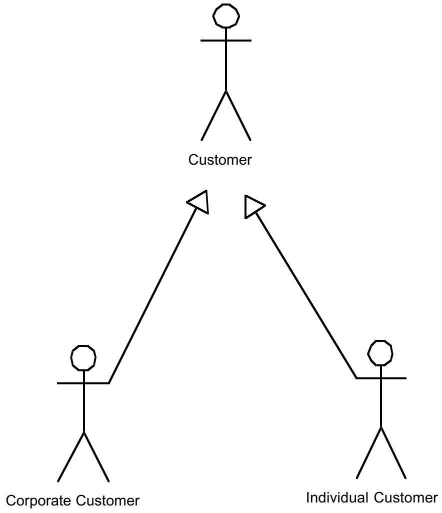
- A use case class is a description that specifies the transactions of the use case
- The set of all use case descriptions specifies the complete functionality of a system

- Are use cases sufficient to serve as the requirements document of a system?

Use Case Example



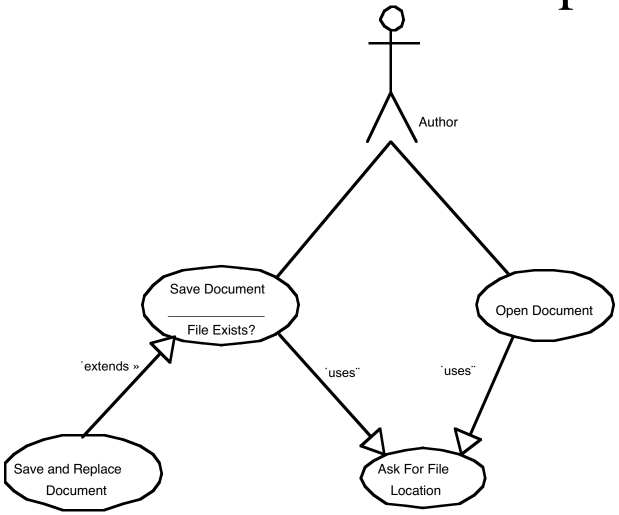
Actors as Classes



Use Case Relationships

- A use case can be related to other use cases in various ways
  - «extend»
    - One use case extends another
      - This typically means that the extension describes an alternate scenario from the original use case
  - «include» or «uses»
    - A use case includes another use case within it
      - This means that one use case includes the sequence of events from another use case as part of its sequence

Extends and Uses Example



More on Use Cases

- A use case captures a contract between the stakeholders of a system about its behavior
  - The use case is initiated by the primary actor; secondary actors may come into play while the use case is executing
  - Note: actors are not restricted to human beings, other computer systems may serve as secondary actors
- The primary actor is trying to achieve a goal
  - Many things may happen; the goal can be achieved (in more than one way) or the use case may fail (also, in more than one way)
  - A use case captures all of these possible scenarios

More on Use Cases

- Despite the notation given previously:
  - Use Cases are primarily a *textual* object
  - The notation is useful for specifying relationships between use cases
    - but it is completely inappropriate for specifying the details of the use case
- Writing good use cases is thus a question of style; some writing styles are more effective than others

Parts of a Use Case

- A use case can be as simple as
  - a paragraph of informal text
- to
  - template-based forms that remind developers what information to include
  - and supported by more formal notations
- What to use depends on the ceremony level of the project
  - high ceremony projects will tend towards formal templates
  - mid ceremony projects will use forms with some or all of the recommended fields
  - low ceremony projects will get by with paragraphs of text

Parts of a Use Case

- As recommended by Alistair Cockburn

Primary Actor	Goal in Context
Scope	Level
Stakeholders and Interests	Precondition
Minimal Guarantees	Success Guarantees
Trigger	Main Success Scenario
Extensions	Technology and Data Variations List
Priority	Releases
Response Time	Frequency of Use
Channel to Primary Actor	Secondary Actors
Channels to Secondary Actor	Open Issues

Highlights from Parts List

- Primary Actor
  - Actor that initiated use case
- Goal Level
  - Can be one of "very high summary", "summary", "user goal", "subfunction", and "too low"
  - Rule of thumb
    - a user goal is one that can be completed in one sitting at a computer
    - a summary goal is one that cannot be completed in one sitting, and may require multiple people, organizations, and systems interacting to achieve the goal

Highlights from Parts List

- Main Success Scenario
  - How is the goal accomplished successfully
- Extensions
  - How might the main success scenario be altered and
    - 1) still succeed
  - or
    - 2) fail

Lets look at some examples…

- From Alistair Cockburn's book
  - pages 4-6 and 9-11
  - page 18
  - See lecture for actual examples
    - If you are an in-class student, you can review these examples without buying Cockburn's book by viewing the tape in the Math Library
    - (across the Engineering plaza to the Northwest)

Two Models of Use Cases

- Cockburn has developed two models for understanding use cases
  - – Actors and Goals
  - – Stakeholders and Interests
- These models can help clarify how to think about and write use cases

Stakeholders With Interests

- A use case can be viewed as a contract between stakeholders with interests
  - – This model identifies what to include in a use case and what to exclude
- Not all stakeholders are present during the operation of the system; when a primary actor interacts with a system, the system must uphold the interests of the "off-stage" actors

Stakeholders/Interests Continued

- Ways to uphold stakeholder interests
  - – Gather Information
    - What information do off-stage actors require to understand the actions of the primary actor
  - – Running Validation Checks
    - Is the primary actor entering valid information
  - – Updating Logs
    - When did the primary actor perform his actions
- Modeling stakeholder interests gives us a rule of thumb: a use cases contains all and only the behaviors related to satisfying stakeholder interests

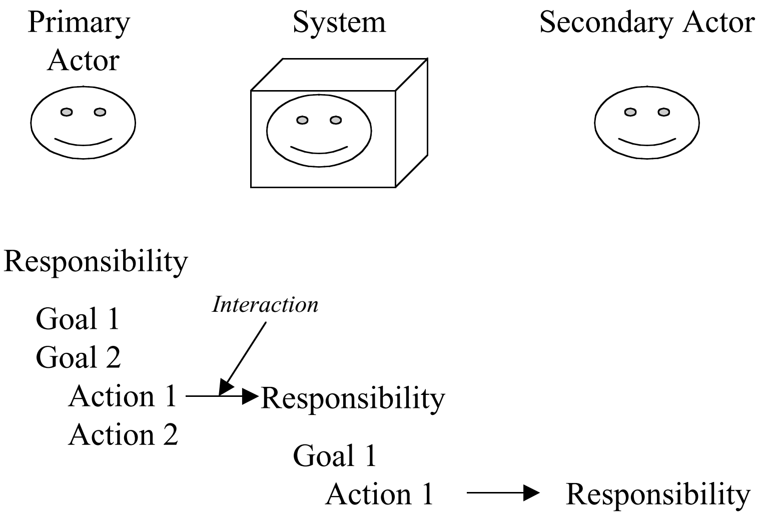
Using the model

- In writing use cases, this model recommends
  - – List all Stakeholders
  - – Name their interests with respect to the use case
  - – State what it means to each stakeholder that the use case completes successfully
  - – List what guarantees each stakeholder wants from the system
- Now, we can write actions steps
  - – This brings us to the Actors and Goals model

Actors and Goals

- An actor has goals
  - To achieve a goal an actor has to take *actions*
  - Achieving a goal may require accomplishing *sub-goals*
  - Achieving sub-goals may require the support and collaboration of *secondary actors*
  - An action may call upon the *responsibilities* of a secondary actor; this sets up an *interaction* where the calling actor must wait for the secondary actor to achieve the goals associated with that responsibility

Actors and Goals Illustrated



Primary Actor

System

Secondary Actor

Responsibility

Goal 1

Goal 2

Action 1 → Responsibility

Interaction

Action 2

Goal 1

Action 1 → Responsibility

Discussion

- Goals have sub-goals
  - avoid having too many sub-goals however
- Goals can fail
  - We must specify how to respond to failure conditions using extensions
- Actions capture Interactions
  - Writing Action Steps is critical to writing good use cases

Writing Action Steps

- Action Steps are written in one grammatical form
  - a simple action in which one actor either
    - accomplishes a task
    - or passes information to another actor
- Examples
  - User enters name and address
  - At any time, user can request the money back
  - The system verifies that the name and account are current

Guidelines for Writing Action Steps

- Use Simple Grammar
  - Subject…verb…direct object…prepositional phrase
  - The system…deducts…the amount…from the account
- Show Clearly “Who Has the Ball”
  - For each step, who is performing it
  - At the end of the step, who has the ball?
- Write From a Bird’s Eye View
  - Not “Get ATM Card and PIN” but “The customer puts in the ATM card and PIN”

Guidelines for Writing Action Steps

- Show the Process Moving Forward
  - Not “User hits tab key” but “User enters Name”
- Show the Actor’s Intent, Not the Movements
  - Before
    - System asks for name; User enters name
    - System prompts for address; User enters address
    - User clicks “OK”
    - System presents user’s profile
  - After
    - User enters name and address
    - System presents user’s profile

Guidelines for Writing Action Steps

- Include a “Reasonable” Set of Actions
  - Ivar Jacobson’s notion of a transaction
    - Actor sends request and data to system
    - System validates the request and data
    - System alters its internal state
    - System responds to actor with result
  - An action step can contain all four; or start with some in one step and end with the others in the subsequent step

Guidelines for Writing Action Steps

- “Validate” Do not “Check Whether”
  - Before
    - The system checks whether the password is correct
    - If it is, the system presents the available actions for the user
  - After
    - The system validates the password is correct
    - The system presents the available actions for the user

The Writing Process

- Cockburn recommends the following process for writing use cases
  - Name the system scope and boundaries
  - Brainstorm and list the primary actors
  - Brainstorm and exhaustively list user goals for the system
  - Capture the outermost summary use cases to see who really cares
  - Reconsider and revise the summary use cases. Add, subtract, or merge goals

The Writing Process, continued

- Select one use case to expand
- Capture stakeholders and interests, preconditions, and guarantees
- Write the main success scenario (MSS)
- Brainstorm and exhaustively list the extension conditions
- Write the extension-handling steps
- Extract complex flows to sub use cases; merge trivial sub use cases
- Readjust the set: add, subtract, merge, as needed