CS 619 Introduction to OO Design and Development

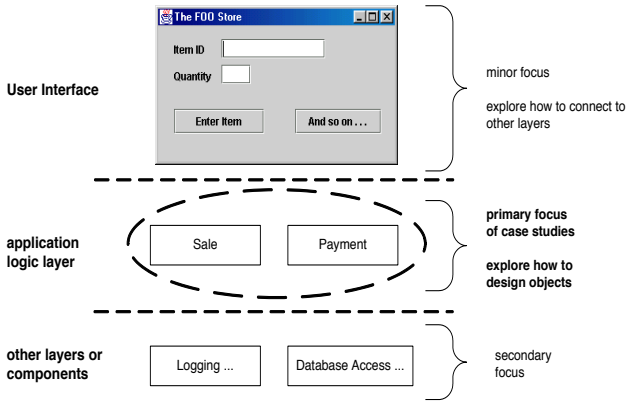
Use Cases

Fall 2014

Define the Problem

The most critical question:

"Is this the right system to make?"



Requirements

Requirements are capabilities and conditions to which the system must conform.

Functional Requirements:

- Specify what the system do.
- System functions requested by the user

Non-functional Requirements:

- Specify how the system work; are global constraints (quality attributes) of the system.
- E.g. Development cost, usability, reliability, performance and supportability, robustness, responsiveness… (more – ilities and -ness)

Requirement Models

In the UP, requirements are categorized according to the **FURPS+ model**

Functional - features, capabilities, security.

Usability **-** human factors, help, documentation.

Reliability - frequency of failure, recoverability, predictability.

Performance - response times, throughput, accuracy, availability, resource usage.

Supportability - adaptability, maintainability, internationalization, configurability.

The "**+**" means

Implementation - resource limitations, languages and tools, hardware..

Interface - constraints imposed by interfacing with external systems.

Operations - system management in its operational setting.

Packaging - for example, a physical box.

Legal - licensing and so forth.

Use Cases

A **use case** is a collection of related success and failure scenarios that describe an actor using a system to support a goal.

- An **actor** is something with behavior and have responsibilities.

  To carry out responsibilities, an actor sets goals
  • **Primary actor** (= stakeholder) has unsatisfied goal and needs system assistance
  • **Secondary actor** provides assistance to satisfy the goal

- A **scenario** is a specific sequence of actions and interactions between actors and the system.

  It is one particular story of using a system, or one path through the use case.

Use case in brief format

Rent Videos

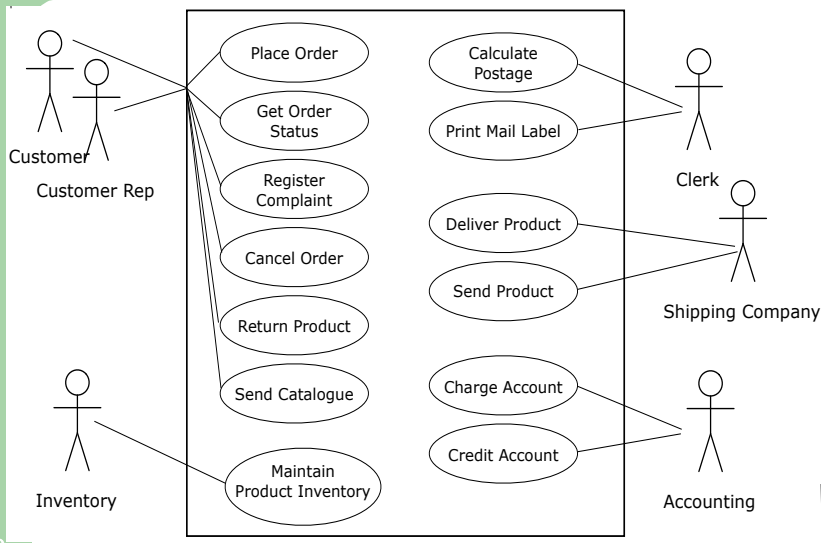
A Customer arrives with videos to rent. The Clerk enters their ID, and each video ID. The System outputs information on each. The Clerk requests the rental report. The System outputs it, which is given to the Customer with their videos.

Three kinds of actors

- Primary actor
  - has user goals fulfilled through using services of the System Under Discussion (SuD). E.g., the cashier.
    Why identify? To find user goals, which drive the use cases.
- Supporting actor
  - provides a service to the SuD. E.g. automated payment authorization service. Often a computer system, but could be an organization or person.
    Why identify? To clarify external interfaces and protocols.
- Offstage actor
  - has an interest in the behavior of the use case, but is not primary or supporting. E.g. a government tax agency.
    Why identify? To ensure that all necessary interests are identified and satisfied. Offstage actor interests are sometimes subtle or easy to miss unless these actors are explicitly named.

Primary and *supporting* actors will appear in the action steps of the use case text.

Example: Use Cases Diagram



Fully Dressed Format Explained

- **Preconditions and Success Guarantees (Postconditions)**
  - **Preconditions** state what must always be true before a scenario is begun in the use case.
    - Preconditions communicate noteworthy assumptions that the writer thinks readers should be alerted to.
  - Success guarantees (or **postconditions**) state what must be true on successful completion of the use case - either the main success scenario or some alternate path.
    - The guarantee should meet the needs of all stakeholders.

Fully Dressed Format Explained

- **Main Success Scenario and Steps (or Basic Flow)**
  - "**happy path**" scenario, or the more prosaic "Basic Flow" or "Typical Flow."
  - It describes a typical success path that satisfies the interests of the stakeholders.
- Guideline
  - defer all conditional handling to the Extensions section.

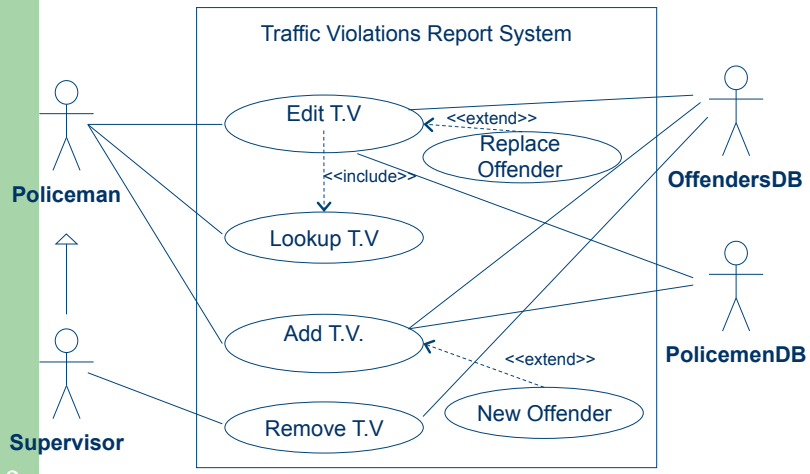
Fully Dressed Format Explained

- **Extensions (or Alternate Flows)**
  - Normally comprise the majority of the text.
  - They indicate all the other scenarios or branches, both success and failure.
  - Extension scenarios are branches from the main success scenario, and so can be notated with respect to its steps 1…N.
  - An extension has two parts: the condition and the handling.

Fully Dressed Format Explained

- Performing Another Use Case Scenario
  - 3a. Invalid item ID:
    1. System signals error and rejects entry.
    2. Cashier responds to the error:
       - 2a. …
       - 2c. Cashier performs ***Find Product Help*** to obtain true item ID and price.

Example: TVRS Use Case Diagram



Example:

Which of these requirements should be represented directly in a use case?

- The precision of calculations shall be at least $1/10^3$.
- The system defect rate shall be less than 1 failure per 1000 hours of operation.
- The application shall ensure that the name of the employee in the official human resource and payroll databases exactly matches the name printed on the employee's social security card.
- The system response time is less than 2 seconds.
- Number of simultaneous users will be 500 max.
- Uptime requirement is 99.9%.

90% 36.5 days/year
99% 3.65 days/year
99.9% 8.76 hours/year
99.99% 52 minutes/year
99.999% 5 minutes/year
99.9999% 31 seconds/year

Identifying Use Case Tests

Boss Test: Especially for the "architecturally significant" use cases, your boss should think this is essential to the business.

EBP Test: An Elementary Business Process is defined as

> A task performed by one person in one place at one time, in response to a business event, that adds measurable value to the business and leaves business data in a consistent state.

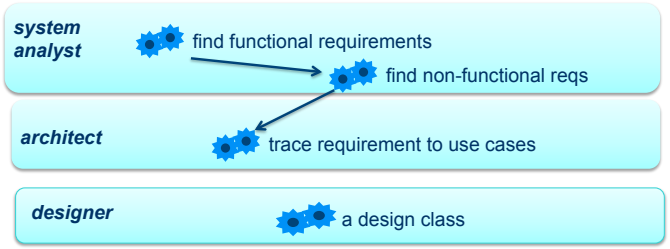
Size Test: Use cases that you can express in less than a page are often not significant. Fully-dressed use cases take 3-10 pages to explain.

Exercises: which of the following is a possible use cases?

- negotiate a supplier contract
- handle returns
- log in
- move pieces on a game board

Benefits of Use Cases

- **Captures functional requirements** from user's perspective. Gives a clear and consistent description of what the system should do
- A basis for performing system tests
- Provides the ability to trace functional requirements into actual classes and operations in the system
- The smallest unit of delivery
  - Each increment that is planned and delivered is described in terms of the Use Cases that will be delivered in that increment



User Story

- A user story is one primary development artifacts for Scrum and Extreme Programming teams.
- A user story is a very high-level definition of a requirement, containing just enough information so developers can produce a reasonable estimate of the effort to implement it.

Template:

As a [type of user] I want to [perform some task] so that I can [reach some goal]

As a registered user I want to log in so I can access subscriber-only content

User Story

- Describes features that are of value to the user, written in the user's language.
- Should be **Independent**, **Negotiable**, **Valuable**, **Estimatable**, **Small** and **Testable**.
- Detail just enough information and no more.

Details are deferred and captured through collaboration just in time for development.

Use Case v.s. User Story

Fill in here:

Resources:

http://www.stellman-greene.com/2009/05/03/requirements-101-user-stories-vs-use-cases/

http://scrumtrainingseries.com/BacklogRefinementMeeting/BacklogRefinementMeeting.htm