Software Re-Engineering

Lecture: 05

Dr. Syed Muazzam Ali Shah
Department of Software Engineering
National University of Computer & Emerging Sciences
muazzam.ali@nu.edu.pk

Sequence [**Todays Agenda**]

Content of Lecture

- Requirements Analysis: Use Case Modeling

NextGen POS

- A POS system is a computerized application used (in part) to record sales and handle payments; it is typically used in a retail store.
- It includes hardware components such as a computer and bar code scanner, and software to run the system.
- It interfaces to various service applications, such as a third-party tax calculator and inventory control.

Introduction

- Use cases are **text stories**, widely used mechanism to discover and record requirements (especially functional)
  - The idea was first introduced by Ivar Jacobson in 1986
- They influence many aspects of a project including OOA/D
- Writing use cases — **stories of using a system** — is an excellent technique to understand and describe requirements
- The UP defines the **Use-Case Model** within the Requirements discipline
- Use-Case model is the set of all use cases; it is a model of the system's functionality and environment

Goals and Stories

- Customers and end users have **goals** (*also known as needs in the UP*) and want computer systems to help meet them
- Use cases are text stories of some actor using a system to meet goals, i.e. ***cases of use***
- Example: **Process Sale**
  1. A customer arrives at a checkout with items to purchase.
  2. The cashier uses the POS system to record each purchased item.
  3. The system presents a running total and line-item details.
  4. The customer enters payment information, which the system validates and records.
  5. The system updates inventory.
  6. The customer receives a receipt from the system and then leaves with the items.

Use Cases by Ivar Jacobson

“A use case is <u>a specific way of using the system</u> by performing some part of the functionality. Each use case constitutes a complete course of events initiated by an actor, and it <u>specifies the interaction that takes place between an actor and the system</u>……

The collected use cases specify all the existing ways of using the system.”

Use Cases

- A use case is a typical sequence of actions that a user performs in order to complete a given task
  - Use cases are functional requirements that indicate what the system will do
- Use cases describe what a system does from the standpoint of an external observer
  - The emphasis is on **what** a system does rather than **how**
  - In terms of the FURPS+ requirements types, they emphasize the "F" (functional or behavioral)
- A use case describes a unit of behavior
  - A use case can satisfy one or more functional requirements
  - A functional requirement may be satisfied by one or more use cases
- The set of all use cases together describes the complete behavior of the system

Use Cases (contd.)

- It is one of the key activities in requirements analysis
- The objective of use case analysis is to model the system from the point of view of
  - … how users interact with this system, when trying to achieve their objectives
- Use cases are text documents, not diagrams
  - Use-case modeling is primarily an act of writing text, not drawing
  - Text is important, diagram is optional
- However, the UML defines a **use case diagram** to illustrate the names of use cases and actors, and their relationships

- Focusing on secondary-value UML use case diagrams rather than the important use case text is a common mistake for use case novices

Use Cases (contd.)

- The focus is:
  - what functionality is involved and who will ask for the provided services
  - How the services are carried out by the system is not addressed

Use Cases are	Use Cases are NOT
What the system will do	How it will be done
Analysis Products	Design Products

Basic Definitions

- **Actor:** Something with behavior, such as a person (identified by role), computer system, or organization OR
  - A user or outside system that interacts with the system being designed in order to obtain some value from that interaction
  - For example, a cashier
- **Scenario:** Specific sequence of actions and interactions between actors and the system
  - Also called a use-case instance
  - One particular story of using a system
  - For example, the scenario of successfully purchasing items with cash, or the scenario of failing to purchase items because of a credit payment denial
  - A **use case** is a collection of related success and failure scenarios that describe an actor using a system to support a goal

Example

- Use case: **Register Course**
- **Main Success Scenario:**
  - The system presents all possible courses for this student. The student select course and commits the entry by pressing the Enter button.
- **Alternate Scenario:**
  - The student can cancel a registration by pressing the Cancel button.

Use Cases and the Use-Case Model

- The UP defines the Use-Case Model within the Requirements discipline
- This is the set of all written use cases
  - It is a model of the system's functionality and environment
- The Use-Case Model may optionally include a UML use case diagram to show the names of use cases and actors, and their relationships
  - This gives a nice context diagram of a system and its environment
  - It also provides a quick way to list the use cases by name

Why Use Cases ???

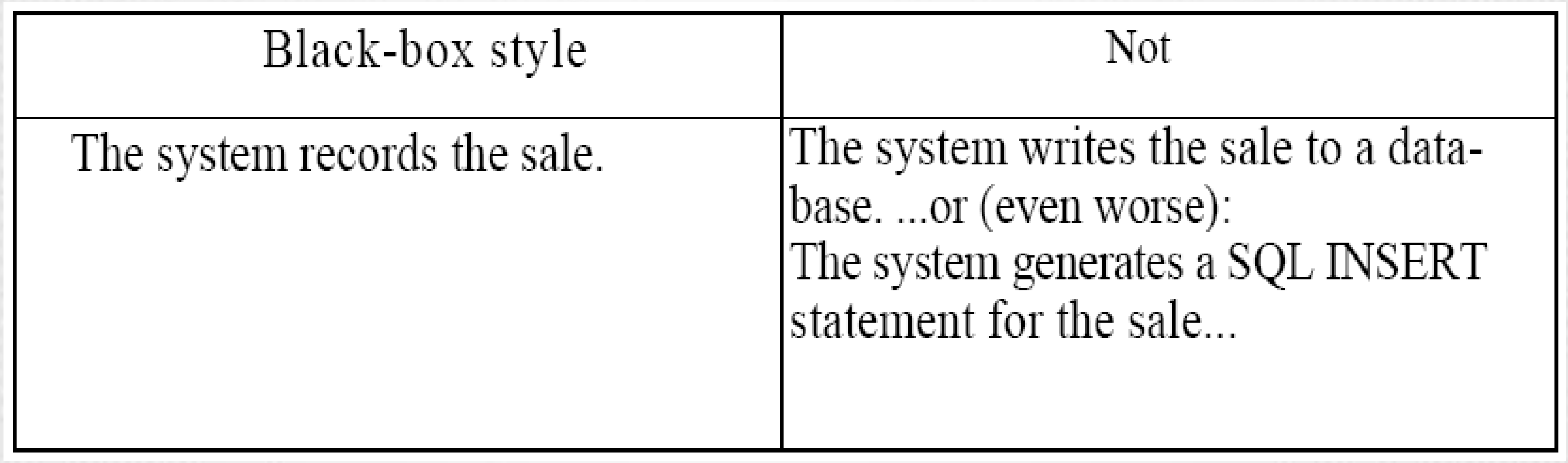
- Lack of user involvement in software projects is near the top of the list of reasons for project failure, so anything that can help keep them involved is truly desirable

  - Another value of use cases is that they emphasize the user goals and perspective

  - We ask the question "Who is using the system, what are their typical scenarios of use, and what are their goals?" This is a more user-centric emphasis compared to simply asking for a list of system features

Black-Box Use Cases

- Most common and recommended kind
- They do **not** describe the internal workings of the system, its components, or design
- During requirements analysis avoid making "how" decisions, and specify the external behavior for the system, as a black box



Black-box style	Not
The system records the sale.	The system writes the sale to a data-base. ...or (even worse): The system generates a SQL INSERT statement for the sale...

Use Case Formats

- **Brief:** Terse one paragraph summary, usually of main success scenario
  - *When?* During early requirements analysis

- **Casual:** Informal paragraph format. Multiple paragraphs that cover various scenarios
  - *When?* During early requirements analysis

- **Fully-Dressed:** The most elaborate. All steps and variations are written in detail, and there are supporting sections, such as preconditions and success guarantees
  - Useful in order to obtain a deep understanding of the goals, tasks, and requirements
  - Various templates are available
  - *When?* After many use cases have been identified and written in a brief format, then during the first requirements workshop a few (such as 10%) of the architecturally significant and high-value use cases are written in detail.

Example: Brief Use Case

- Use Case: **Process Sale**

  A customer arrives at a checkout with items to purchase. The cashier uses the POS system to record each purchased item. The system presents a running total and line-item details. The customer enters payment information, which the system validates and records. The system updates inventory. The customer receives a receipt from the system and then leaves with the items.

Example: Brief Use Case (contd.)

- Use Case: **Buy a Product Online**

  The customer browses the catalog and adds desired items to the Shopping basket. When the customer wishes to pay, the customer describes the shipping and credit card information and confirms the sale. The System Checks the authorization an the credit card and confirms the sale both immediately and with a follow-up e-mail.

Example: Casual Use Case (contd.)

- Use case: **Buy a Product Online**
- **Main Success Scenario:**

1. Customer browses catalog and selects items to buy
2. Customer fills in shipping information (address ; next-day or 3-day delivery)
3. System presents full pricing information, including shipping
4. Customer fills in credit card information
5. System authorizes purchase
6. System confirms sale immediately
7. System sends confirming e-mail to customer

- **Alternate Scenarios:**

6a: System fails to authorize credit purchase

1: Customer may reenter credit card information or may cancel

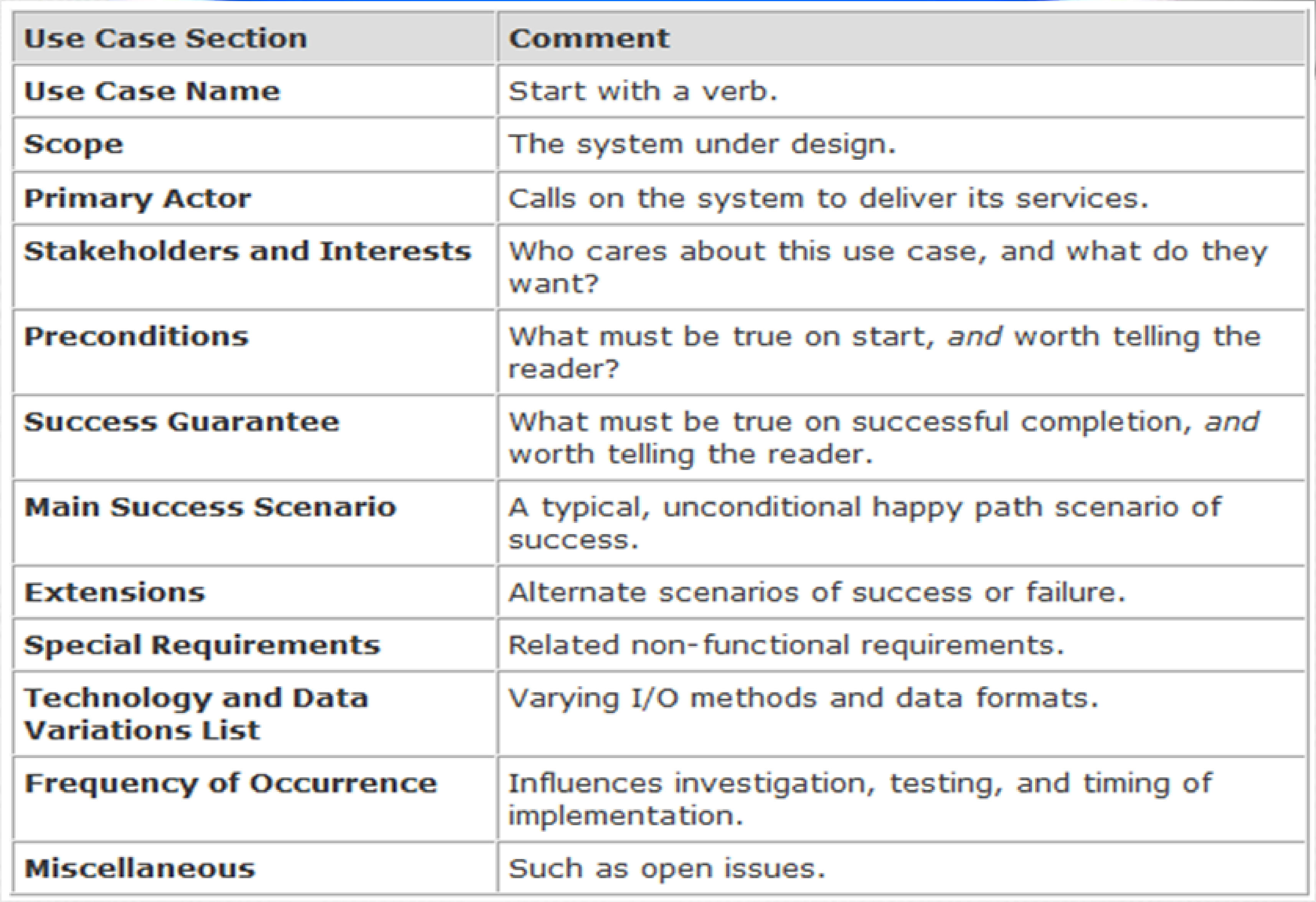
Example: Fully-dressed Use Case

- In iterative and evolutionary UP requirements analysis, 10% of the critical use cases would be written this way during the first requirements workshop

- Then design and programming starts on the most architecturally significant use cases or scenarios from that 10% set

- **Applying UML and Patterns**
  - Section 6.8: Example: Process Sale, Fully Dressed Style

Template: Fully-dressed Use Case



Use Case Section	Comment
Use Case Name	Start with a verb.
Scope	The system under design.
Primary Actor	Calls on the system to deliver its services.
Stakeholders and Interests	Who cares about this use case, and what do they want?
Preconditions	What must be true on start, *and* worth telling the reader?
Success Guarantee	What must be true on successful completion, *and* worth telling the reader.
Main Success Scenario	A typical, unconditional happy path scenario of success.
Extensions	Alternate scenarios of success or failure.
Special Requirements	Related non-functional requirements.
Technology and Data Variations List	Varying I/O methods and data formats.
Frequency of Occurrence	Influences investigation, testing, and timing of implementation.
Miscellaneous	Such as open issues.

Example: Fully-dressed Use Case

Use Case UC1: Process Sale

- **Scope:** NextGen POS application
- **Primary Actor:** Cashier
- **Stakeholders and Interests:**
  - **Cashier:** Wants accurate, fast entry, and no payment errors, as cash drawer shortages are deducted from his/her salary.
  - **Salesperson:** Wants sales commissions updated.
  - **Customer:** Wants purchase and fast service with minimal effort. Wants easily visible display of entered items and prices. Wants proof of purchase to support returns.
  - **Company:** Wants to accurately record transactions and satisfy customer interests. Wants to ensure that Payment Authorization Service payment receivables are recorded. Wants some fault tolerance to allow sales capture even if server components (e.g., remote credit validation) are unavailable. Wants automatic and fast update of accounting and inventory.

Use Case UC1: Process Sale

- **Manager:** Wants to be able to quickly perform override operations, and easily debug Cashier problems.
- **Government Tax Agencies:** Want to collect tax from every sale. May be multiple agencies, such as national, state, and county.
- **Payment Authorization Service:** Wants to receive digital authorization requests in the correct format and protocol. Wants to accurately account for their payables to the store.

- **Preconditions:** Cashier is identified and authenticated.
- **Success Guarantee (or Postconditions):** Sale is saved. Tax is correctly calculated. Accounting and Inventory are updated. Commissions recorded. Receipt is generated. Payment authorization approvals are recorded.

Use Case UC1: Process Sale (contd.)

Main Success Scenario (or Basic Flow):

1. Customer arrives at POS checkout with goods and/or services to purchase.
2. Cashier starts a new sale.
3. Cashier enters item identifier.
4. System records sale line item and presents item description, price, and running total. Price calculated from a set of price rules.
5. Cashier repeats steps 3-4 until indicates done.
6. System presents total with taxes calculated.
7. Cashier tells Customer the total, and asks for payment.
8. Customer pays and System handles payment.
9. System logs completed sale and sends sale and payment information to the external Accounting system (for accounting and commissions) and Inventory system (to update inventory).
10. System presents receipt.
11. Customer leaves with receipt and goods (if any).

Use Case UC1: Process Sale (contd.)

⌗ **Extensions (or Alternate Flows):**

***a.** At any time, System fails:

To support recovery and correct accounting, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

- **1.** Cashier restarts System, logs in, and requests recovery of prior state.
- **2.** System reconstructs prior state.
  - **2a.** System detects anomalies preventing recovery:
    - **1.** System signals error to the Cashier, records the error, and enters a clean state.
    - **2.** Cashier starts a new sale.

3a. Invalid item ID (not found in system):

- **1.** System signals error and rejects entry.
- **2.** Cashier responds to the error.

3b. There are multiple of same item category and tracking unique item identity not important (e.g., 5 packages of veggie-burgers):

- **1.** Cashier can enter item category identifier and the quantity.

3-6a: Customer asks Cashier to remove an item from the purchase:

- **1.** Cashier enters item identifier for removal from sale.
- **2.** System displays updated running total.

3-6b. Customer tells Cashier to cancel sale:

- **1.** Cashier cancels sale on System.

Naming Use Cases

- Must be a complete process from the viewpoint of the end user
- Usually in verb-object form, like *Buy Pizza*
- Use enough detail to make it specific
- Use active voice, not passive
- From viewpoint of the actor, not the system

Examples: Use Case Names

- **Excellent** - Purchase Concert Ticket
- **Very Good** - Purchase Concert Tickets
- **Good** - Purchase Ticket (*insufficient detail*)
- **Fair** - Ticket Purchase (*passive*)
- **Poor** - Ticket Order (*system view, not user*)
- **Unacceptable** - Pay for Ticket (*procedure, not process*)

Class Activity

Question no.1: Web Customer uses some web site to make purchases online. Top level use cases are View Items, Make Purchase and Client Register. View Items use case could be used by customer as top-level use case if customer only wants to find and see some products. This use case could also be used as a part of Make Purchase use case. Client Register use case allows customer to register on the web site, for example to get some coupons or be invited to private sales. Note that Checkout use case is included use case not available by itself - Checkout is part of making purchase and checkout is possible after the payment by Credit card payment service i.e. PayPal.

Draw the Use case diagram of the given scenario of Online Shopping.

References

- **Applying UML and Patterns**
  - Chapter 6: Use Cases
- **UML Distilled**
  - Chapter 9: Use Cases

Thank You!