

Use Cases And Use Case Level Test Cases

Lesson 00

IGATE is now a part of Capgemini

People matter, results count.



©2016 Capgemini. All rights reserved.

The information contained in this document is proprietary and confidential. For Capgemini only.

Document History

Date	Course Version No.	Software Version No.	Developer / SME	Reviewer(s)	Approver	Change Record Remarks
	0.1D	NA				Content Creation
	0.1	NA	Sudhir Karhadkar			Review
May-2009	2.0	NA	Priya Rane			Material Revamp
May 2011	2.1	NA	Vaishali Kunchur			Material Revamp

Course Goals and Non Goals

- Course Goals

- At the end of this program, participants gain an understanding of how to create effective Use cases to get a good coverage of a software application


- Course Non Goals

- This course does not cover tool training for Use cases



Pre-requisites

- None

 Capgemini
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 4


Intended Audience

- Test Engineers and Senior Test Engineers



Day Wise Schedule

- Day 1
 - Lesson 1: Introduction to Use cases

 Capgemini
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 6

Table of Contents

- Lesson 1: Introduction to Use cases
 - 1.1. Use case modeling
 - 1.2. Advantage of use cases
 - 1.3. Actor
 - 1.4 Goals and Requirements
 - 1.5 Goals and scenarios
 - 1.6 Naming Conventions
 - 1.7 Alternate Path
 - 1.8 Exceptions
 - 1.9 Errors
 - 1.10 Precondition & Post condition
 - 1.11 Steps for Use case modeling
 - 1.12 Good practices
 - 1.13 Failure scenarios

References

- Book:

- UML User's Guide; by Grady Booch, Ivar Jacobson, and James Rumbaugh

- Web-site:

- <http://www.uml.org/>




Next Step Courses (if applicable)

- Task Based Approach



Other Parallel Technology Areas

- UML

 Capgemini
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 10

Use Cases And Use Case Level Test Cases

Lesson 1: Introduction to Use
Cases

Lesson Objectives

- To understand the following topics:
 - Use case modeling
 - Advantage of use cases
 - Actor
 - Goals and Requirements
 - Goals and scenarios
 - Naming Conventions
 - Alternate Path
 - Exceptions



Lesson Objectives


- To understand the following topics:
 - Errors
 - Precondition & Postcondition
 - Steps for Use case modeling
 - Good practices
 - Failure scenarios



1.1: Use case modeling

Use Case Usage

- Use cases help to:
 - Describe a business process
 - Capture functional requirements of a system
 - Document design details of a system

Copyright © Capgemini 2015. All Rights Reserved 4

Consider the example of ATM machine. If we have to design an application to create an ATM system then we will have to first understand stepwise working of ATM system.

1. Swipe your card
2. System will ask you for the pin no
3. Provide the pin no
4. System will authenticate your pin and provide the options
5. Select the option eg. Withdraw
6. System will prompt you to enter the amount
7. Provide the amount
8. System will check available balance and dispense money
9. Collect money and collect receipt

This describes the steps followed to perform withdrawal transaction in ATM machine.

Now we have to find out what function do we expect from the ATM system? We would want the ATM to provide us with functionalities like withdrawal, deposit, funds transfer, mini statement etc. These are called as the functional requirements. These functional requirements should be captured and detailed steps (like the one given above) should be known. This can be done with the help of USE CASES. USE CASE captures the functional requirements of a system. Thus the Use case document will provide the design details of the system.

1.1: Use case modeling

Use Case Diagrams

- Use Case Diagrams model the functionality of system by using Actors and Use Cases:
 - Actor is a user of the system
 - Use cases are services or functions provided by the system to its users

The diagram illustrates the symbols used in Use Case Diagrams. On the left is an orange stick figure representing an Actor, with the word 'Actor' written below it. On the right is an orange oval representing a Use Case, with the words 'Use Case' written below it.

Capgemini
CONSTRUCTIVE TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 5

Use Case Diagrams:

Use Case is a description of a system's behavior from a user's point of view. It is a set of scenarios that describe an interaction between a user and a system. It also displays the relationship among Actors and Use Cases.

Two main components of Use Case diagram are Use Cases and Actors.

Use case diagrams, which render the User View of the system, describe the functionality (Use Cases) provided by the system to its users (Actors).


An Actor represents a user or another system that will interact with the system you are modeling.

An Use Case is an external view of a system that represents some action that the user might perform in order to complete a task.

1.1: Use case modeling

Use Cases

- Use Case:
 - An Use Case can be defined as a set of activities performed within a system by a User
 - Each Use Case:
 - describes one logical interaction between the Actor and the system
 - defines what has changed by the interaction

Capgemini
CONNECTING TECHNOLOGY. OUTSOURCING.

Copyright © Capgemini 2015. All Rights Reserved 6

Use Cases:

The Use Cases define “units of functionality” provided by system. They model “work units” that the system provides to its outside world.

A Use Case is one usage of the system. It is a generic description of a use of the system. It allows interactions in a specific sequence.

At the lowest level, they are nothing but methods which need to be implemented by various classes in the system.

Use Cases determines everything that the Actor wants to do with the system.

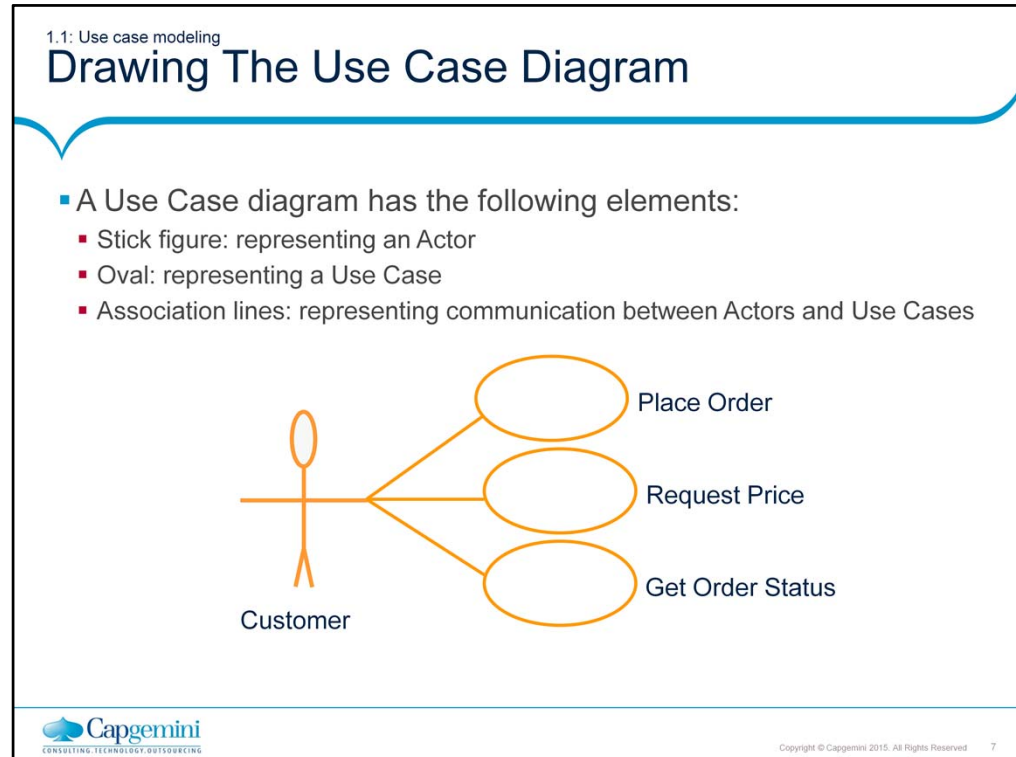
A Use Case performs the following functions:

- Defines main tasks of the system

- Reads, writes, and changes system information

- Informs the system of real world changes

A Use Case needs to be updated / informed about system changes.



Drawing the Use Case Diagram:

The Use Case Diagram has the following elements:

A stick figure, which represents Actors (sometimes stereotyped classes, as explained later, are also used to represent Actors). They differ from tool to tool.

Ovals or ellipses, which represent Use Cases

Association lines, which indicate interactions between Actors and Use Cases.


Use Cases will have description of what the Use Case is supposed to do when it is used.

An example of use case description is given.

1.1: Use case modeling

Use Case Modeling

- Very effective elicitation technique
- Can be used throughout SDLC (should be refined)
- Identifies Who (Actors) and the What (Behavior)
- Useful for gathering behavioral aspects and documenting functionality
- Does NOT capture how system will perform a function
- The rule is "Model it if it is useful"
- Corollary is also true..."don't model if it is not useful"



Capgemini
CONSTRUCTIVE TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 8

To Understand detailed system requirements Use Cases are created and they can be used during entire software development life cycle.

For E.g. Use case – Place Order

User enters item number in the system

System displays item details.

User enters quantity.

System displays price and total value for the order.

Steps 1 to 4 can be repeated.

User enters payment details in the system.


This Use case shows steps for normal behavior. What if, item does not exist? Or if quantity is more than quantity on hand? Use case will capture these as different scenarios.

Use Case is a collection of possible scenarios between the system and external actors, characterized by the goal the primary actor has toward the system's declared responsibilities, showing how the primary actor's goal might be delivered or might fail.

1.2: Advantage of use cases

Advantages

- Easy to write
- Expressed in simple language that user understands
- Can be visually represented
- Can be used for validation (acceptance testing)
- Tools support
- Ideal for OO development
- Function oriented system (different types of users, complex behavioral patterns)
- When Not to use case?
 - System has few users
 - System dominated by non-functional requirements

Capgemini
CONSTRUCTIVE TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 9

Advantages of Use Cases:

Use cases are written in simple users language and are easy to write. Its visual representation has uniform interpretation. Use cases will capture validation requirements and hence will be useful for acceptance testing.

Different tools like Rational Rose, Rational Software Architect, Star UML etc. can be used to create Use case diagrams. They are ideal for object oriented development as we have a clear representation of objects in the form of Actors, external systems etc. Use cases are ideal for systems that have many functional requirements.

We cannot create Use cases for systems which have few users or systems which have more non-functional requirements.

1.3: Actor

Definition

■ Actor:

■ An Actor can be described as follows:

- Actor is any entity that is external to the system and directly interacts with the system, thus deriving some benefit from the interaction
- Actor can be a human being, a machine, or a software
- Actor is a role that a particular user plays while interacting with the system
- Examples of Actors are End-user (roles), External systems, and External passive objects (entities)



Copyright © Capgemini 2015. All Rights Reserved 10

Actor:

Actors are people, organizations, systems, or devices which use or interact with our system. The system exists to support that interaction. Therefore, the important part of the project is to identify the Actors and find out what they want from the system. Actors are characterized by their external view rather than their internal structures. It is a role that the user plays to get something from the system.

Role and organization Actors only require logical interactions with the system. Ask who wants what from our system, rather than who operates the system.


For example: ABC and XYZ are users who wish to buy from an online store. For the online stores system, they play the role of a customer, and hence customer is the Actor for the system. The database for this system may already be existing, and hence this may be another Actor (note that user in this case is not a human).

The Actors will finally be used to describe classes, which will interact with other classes of the system.

1.3: Actor

Use Case Modeling: Actors

- Define the Actors
 - An Actor is someone or something that interacts with the system
 - Type of actors: People (Users), Devices, Other Systems, Organizations
 - Actors live outside the boundary of the system being defined
 - Actors initiate an action
 - Note: The ACTOR implies an individual or system in action and denotes a ROLE and not necessarily the functional responsibility of the individual

 Copyright © Capgemini 2015. All Rights Reserved 11

An actor is always outside the system. An actor characterizes role of a system. The actor starts the use case because it wants something from the system.


A use case answers a question of the form, "How do I get money out of that banking machine?" You, the primary actor, have a goal, to get money out the system, which is that banking machine. There are a number of situations you could find yourself in. The use case collects them into one place. It breaks down your goal into sub goals, eventually individual message actions, plus the interactions between various actors as they try to reach that goal, or eventually fail to reach it.

1.3: Actor

Use Case Modeling: Actors

Type of Actors:

- Primary Actor
 - Initiates the Use Case
 - Calls on the system to deliver a service
 - Has a goal with respect to the system
- Supporting Actor
 - provides a service to the system under design

 Copyright © Capgemini 2015. All Rights Reserved 12

PRIMARY ACTOR

A PRIMARY ACTOR is one who calls on the system to deliver a service

A PRIMARY ACTOR is typically who triggers the use case

Examples: A caller initiating a telephone call, a customer withdrawing money from the ATM

Note: Use cases help identify Primary actors early in the lifecycle

For e.g. The customer requests to open a bank account, the bank clerk collects the customer's details and those of the requested account. This information is validated. If validation is ok the account is created otherwise the request is rejected. In this scenario, Customer is a primary actor and bank clerk is supporting actor.

SUPPORTING ACTOR

A SUPPORTING ACTOR is an external actor that provides a service to the system under design.

Examples: A high-speed printer, a web service


Note: We can use supporting actors to identify external interfaces and the protocols the system will use. An actor can be primary in one use case and supporting in another

For e.g. In ATM system, if user wants a to print a statement of his transaction, printer in this scenario acts as a supporting actor, which is outside the system for supporting print utility.

1.3: Actor

Use Case Modeling: Actors and goals

- A GOAL is:
 - What an ACTOR wants to get with the help of the system
- An ACTION is
 - what the ACTOR must perform to reach the GOAL
- An INTERACTION is
 - a sequence of steps that must be followed to complete the ACTION

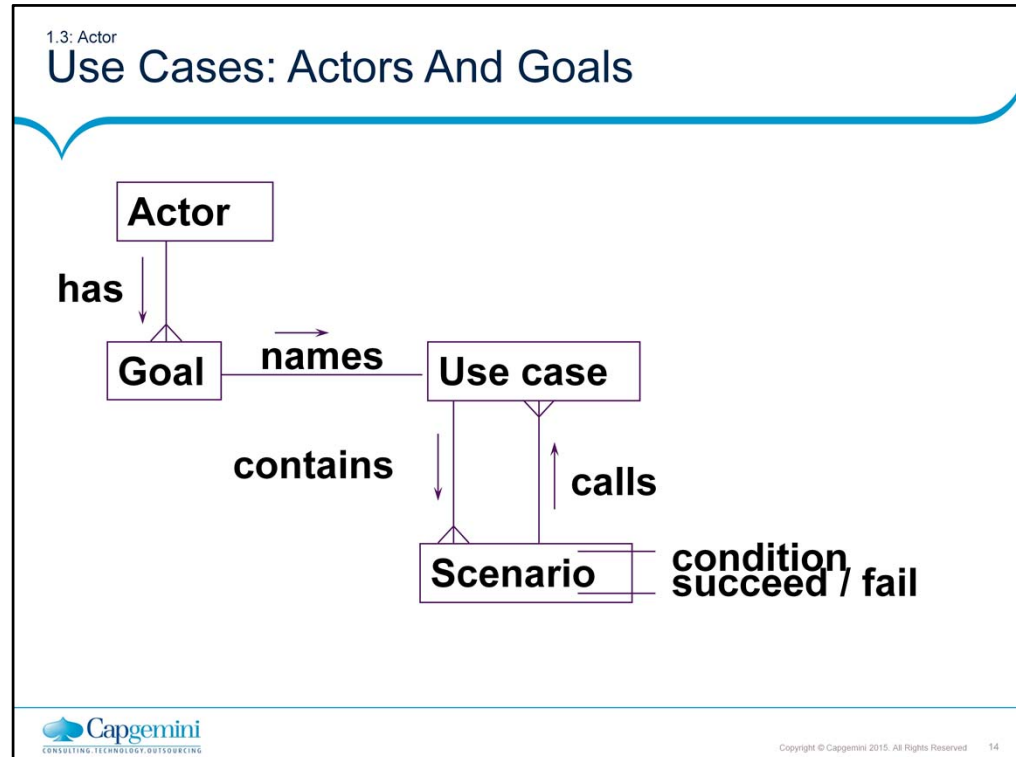
 Copyright © Capgemini 2015. All Rights Reserved 13

Each actor has a set of “responsibilities”. To carry out this responsibility, it sets some goals. To reach a goal, it performs some actions. An “action” is the trigger of an interaction with another actor, calling upon one of the responsibilities of the other actor.

An interaction also could be a “sequence of interactions”. This is a recursive definition. At the bottom level, it consists of messages. We sometime want to bundle a sequence of messages into a single interaction item.

A collection of scenarios is a “use case”.

A use case answers a question of the form, “How do I get money out of that banking machine?” You, the primary actor, have a goal, to get money out the system, which is that banking machine. There are a number of situations you could find yourself in. The use case collects them into one place. It breaks down your goal into sub goals, eventually individual message actions, plus the interactions between various actors as they try to reach that goal, or eventually fail to reach it.



Actor has some goal with respect to the system. This goal is identified as Use case. Use cases contains different scenarios. For different scenarios, separate use case is prepared. The scenarios are either successful or failure scenarios. Test engineer has to consider successful as well as failure scenarios.

For E.g. The insurance company

Primary Actor: the claimant

Goal: Get paid for car accident

Conditions: Everything is in order

Outcome: Insurance company pays claim

1. Claimant submits claim with substantiating data.
2. Insurance company verifies claimant owns a valid policy
3. Insurance company assigns agent to examine case
4. Agent verifies all details are within policy guidelines
5. Insurance company pays claimant

1.4: Goals and Requirements

Use Cases: Goals And Requirements

- Examining the Goals the system supports makes good functional requirements.
 - "Place an order."
 - "Get money from my bank account."
 - "Get a quote."
 - "Find the most attractive alternative."
 - "Set up an advertising program."
- Goals summarize system function in understandable, verifiable terms of use.



Copyright © Capgemini 2015. All Rights Reserved 15


Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform.


1.4: Goals and Requirements

Use Cases: Goals And Requirements

- Structured narrative keeps the context visible
- Just paragraphs:
 - “ATM system has in interface with Bank database. It verifies the customer details and provides options. ...”
- With structured narrative:
 - “Customer swipes the ATM card and enters the pin number. ATM system has an interface with Bank database to verify the customer pin. The system displays options to the customer”

When? How?



 Capgemini
ENLIGHTENING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 16

Many times requirements are captured in the form of a paragraph. These paragraphs are written in a structured or unstructured format.

There are two paragraphs mentioned in the above slide, one is just a normal unstructured paragraph which does not give us clear information about when does the customer verification happen? Or how does it happen?

Another one is structured paragraph:

Eg: “The order entry system has an interface to system EBMS and to a terminal. It computes and displays the sum of the order items’ cost.

Unstructured will not give us details about when is the “sum of ordered items” displayed? Or how is it calculated. It simply gives us some information but not complete information.


Eg: The person who enters order identifies the name of the customer & the items on the order. The system displays the cost of the total order. If the items are in stock and the client has sufficient credit”

Structured paragraph on the other hand will give us unambiguous information about the same function.

1.5: Goals and scenarios

Goals And Scenarios

- A use case pulls goals and scenarios together
- In Use Case “Withdraw money from ATM”
- Scenario 1: Everything works well ...
- Scenario 2: Customer enters invalid pin ...
- Scenario 3: Insufficient balance ...
- Use case is goal statement plus the scenarios.
- Note the grammar for Use Case: active verb first

Capgemini
CONSTRUCTIVE TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 17

Example: “Order product from catalog”

Scenario 1: Everything works out well ...

Scenario 2: Insufficient credit ...

Scenario 3: Product out of stock ..

Use case is goal statement plus the scenarios.

A sequence has no branching or alternatives. It therefore is used to describe the past or a definite future, with conditions stated. Such a sequence is known as a “scenario”. So “Sequence of Interactions” is the same as “scenario”.

Scenario is a sequence of interactions happening under certain conditions, to achieve the primary actor's goal, and having a particular result with respect to that goal. The interactions start from the triggering action and continue until the goal is delivered or abandoned, and the system completes whatever responsibilities it has with respect to the interaction.

A scenario is an instance of a use case, and represents a single path through the use case. Thus, one may construct a scenario for the main flow through the use case, and other scenarios for each possible variation of flow through the use case (e.g., triggered by options, error conditions, security breaches, etc.). Scenarios may be depicted using sequence diagrams.

1.6: Naming Conventions

Use Cases: Naming Use Cases

- Name a use case with a verb-noun phrase that states the actor's goal
- Use concrete, "strong" verbs instead of generalized, weaker ones.
- Weak verbs may indicate uncertainty
 - Strong Verbs: create, merge, calculate, migrate, receive, register
 - Weaker Verbs: make, report, use, copy, organize, record, find etc
- Be explicit. Use specific terms. They are stronger
 - Strong Nouns: property, payment, transcript, account
 - Weaker Nouns: data, paper, report, system, form



Copyright © Capgemini 2015. All Rights Reserved 18

How to name a Use Case?


Use case names should describe the functionality of the system. Some of the tips are given in above slide will help you give appropriate names to the Use Cases
Eg: ATM system Use cases can be:

1. Withdraw money – (use of strong verb and strong noun)
2. View Mini statement

1.6: Naming Conventions

Use Cases: Finding Use Cases

- What are the primary tasks the actor wants the system to perform?
“I want to ...”
 - transfer money between accounts”
 - get money from my account
 - make payments”
- Does actor create, read, update, and delete information
- Does actors need any notification of changes to the internal state of the system?

 **Capgemini**
CONSTRUCTIVE TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 19

Identify the type of functions are expected by each actor from the system under design:

In the ATM example :

We expect ATM to perform following functions for us (Customer)

1. Withdraw Money
2. Deposit Money
3. Maintain Customer Account (“Maintain” will take care of create, modify, remove)
4. Funds Transfer
5. View Mini statement

The system exists only for its use, and therefore it should be based on users needs.

Ask the questions on the slide and answers to these questions represent the flow of events that identify candidate use cases.

1.6: Naming Conventions

Use Cases: Naming Actors

- Group individuals according to their common use of the system
- Identify the roles they take on when they use the system
- Each role is a potential actor
- Name each role and define its distinguishing characteristics
- Do not equate job title with role name. Roles cut across job titles
- Use the common name for an existing system; don't invent a new name to match its role



Copyright © Capgemini 2015. All Rights Reserved 20

OMG gives the description of Actor as follows:


Actors may represent roles played by human users, external hardware, or other subjects. Note that an actor does not necessarily represent a specific physical entity but merely a particular facet (i.e., "role") of some entity that is relevant to the specification of its associated use cases. Thus, a single physical instance may play the role of several different actors and, conversely, a given actor may be played by multiple different instances.

1.7: Alternate Path

Use Cases: Alternative Paths

For each significant action:

- Is there another significant way to accomplish it that could be taken at this point? (Variation)
- Is there something that could go wrong? (Exception)
- Is there something that could go really, really wrong (Error)

Capgemini
CONSTRUCTIVE TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 21

When you have a particular task that can be accomplished in two ways. One normal way is the basic path and the other way is alternate path.

While following the basic path, there is a possibility of occurrence of some problems which will be captured in the exception flow.

And if some error occur, which cannot be handled, then it is error flow.

Example

In a human resources system, for the “Hire Employee” use case, the following scenarios may apply:

Typical success scenario: Hire a person from outside of the company, for example, from another company

Alternative success scenario: Hire a person from within the same company, for instance, from another division

Exceptional failure scenario: No qualified person could be hired

1.8: Exceptions

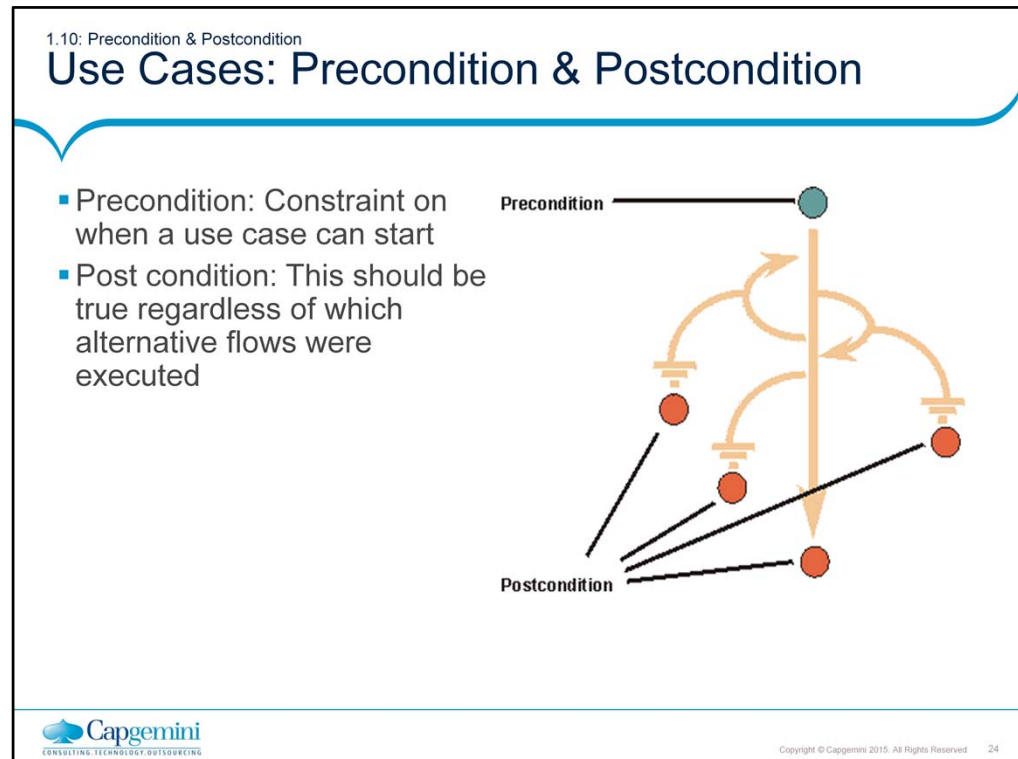
Use Cases: Exceptions

- Exceptions are deviations from the typical case that happen during the normal course of events
 - They should be handled, not ignored
 - How to resolve them can be open to debate
- What if a user mistypes her password?
- What if an order can't be fulfilled?
- What if a connection to a web browser is dropped?

1.9: Errors

Use Cases: Errors

- Errors are when things unexpectedly go wrong. They can result from malformed data, bad programs or logic errors, or broken hardware
 - Little can be done easily to “fix things up and proceed”
 - Recovery requires drastic measures
- What if the disk is full?
- What if equipment cannot be provisioned?
- What if the OS crashes?



The states described by pre- or postconditions should be states that the user can observe. "The user has logged on to the system" or "The user has opened the document" are examples of observable states.

A precondition is a constraint on when a use case can start. It is not the event that starts the use case.

A precondition for a use case is not a precondition for only one subflow, although you can define preconditions and postconditions at the subflow level.

A postcondition for a use case should be true regardless of which alternative flows were executed; it should not be true only for the main flow. If something could fail, you would cover that in the postcondition by saying "The action is completed, or if something failed, the action is not performed", rather than just "The action is completed".

When you use postconditions together with extend-relationships, you should take care that the extending use case does not introduce a subflow that violates the postcondition in the base use case.

postconditions can be a powerful tool for describing use cases. You first define what the use case is supposed to achieve, the postcondition. You can then describe how to reach this condition (the flow of events needed).

1.10: Precondition & Postcondition

Use Cases: Precondition & Postcondition

- Precondition

- is constraint not the event that starts the use case
- state what must always be true before beginning a scenario in use case are not tested within the use case; rather, they are conditions that are assumed to be true
- communicate noteworthy assumptions that the use case writer thinks readers should be alerted to condition to start an execution of test case

For e.g. To open a new bank account, some pre conditions or constraints are applied as follows.

Applicant must be 18 years of age or older and reside in the European Union.

E.g “Cashier is identified and authenticated”

1.10: Precondition & Postcondition

Use Cases: Precondition & Postcondition


- Postconditions

- state what must be true on successful completion of the use case—either the main success scenario or some alternate path
- should be true regardless of which alternative flows were executed
- can be a powerful tool for describing use cases
- First define what the use case is supposed to achieve, the postcondition

For e.g. After the business process for opening an account is done, Customer is informed of creation of new account or Customer is informed of rejected application. This is the post condition for the scenario.

Demo

- Use case_template.doc
- Sample Use Case.doc
- Scenario Matrix.doc
- Use Case to Test Case.doc



Capgemini
CONSULTING TECHNOLOGY OUTSOURCING


Copyright © Capgemini 2015. All Rights Reserved 27

Add the notes here.

1.11: Steps for Use case modeling

Steps In Use Case Modeling

- Identify the actors and their goals
 - What computers, subsystems and people will drive our system?
 - What does each actor need our system to do?
- Result: a list of use cases, a sketch of the system

 Capgemini
CONSTRUCTIVE TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 28

Tips: Identifying actors

Who uses the system?

Who gets information from this system?

Who provides information to the system?

Who installs, maintains the system?

Example:

Order management System:

Customer wants to place Order

Sales representative wants to view Orders

Customer wants to view Order status

Sales representative enters/modifies/deletes Order details

Administrator manages master data

Managers want to generate Sales Reports

So The Use cases are

Place Order

View Orders

Manage Orders

Generate Sales Reports

Actors:

Customer, Sales Rep., Administrator

1.11: Steps for Use case modeling

Steps In Use Case Modeling (Cont...)

- Write the simple case: Goal delivers
 - The main success scenario, the happy day case – make it easy to understand
 - Capture each actor's intent and responsibility, from trigger to delivery
 - Say what information passes between them
- Result: readable description of system's function



Copyright © Capgemini 2015. All Rights Reserved 29

Tips:

Show actor intent, not precise movements

Example: "User enters ID and Password" instead of "User enters ID, User enters Password"

Write actions that move the process forward

Example: "Validate" as opposed to "Check whether"

1.11: Steps for Use case modeling

Steps In Use Case Modeling (Cont...)

- Write failure conditions / alternatives as extensions
 - Usually, each step can fail
 - Note the failure condition separately, after the main success scenario
- Result: List of alternate scenarios. An extension can also be another use case

1.11: Steps for Use case modeling

Steps In Use Case Modeling (Cont...)

- Follow the failure till it ends or rejoins
 - Recoverable extensions rejoin main course. Non-recoverable extensions fail directly
 - Each scenario goes from trigger to completion
 - Can write each scenario from beginning to end
 - Result: Complete use case

1.11: Steps for Use case modeling

Steps In Use Case Modeling (Cont...)

- Note the data variations
 - Some extensions are too low-level to cover “here”. e.g. “Reimburse customer”
Reimburse by cash, check, EFT, or purchase credit?
- Deferred variations note cases that must be handled eventually, by lower-level use cases
- Useful for tracking requirements at high level
- Result: Feed-forward information, rolled up into an easy-to-track format

1.11: Steps for Use case modeling

Write A New Use Case When...

- An alternative appears complex
- When you want to emphasize an alternative
- Document alternatives
- For better clarity
- Give specific names to use cases based on conditions

1.12: Good practices

Use Cases: Good Practices

- Make the use cases clear, short and easy to read
- Use active voice, present tense, make sure actor and actor's intent are visible in each step
- Every Use case has two possible endings: Success and Failure / alternate courses. Gather both
- Create a list of primary actors and their goals (actor-goal list)
- Restrict use cases to capture system behavior....use cases are not suitable for other type of requirements

1.13: Failure scenarios

Failure Scenarios

- Don't overlook failure scenarios
 - "What if their credit is too low?"
 - "What if they run over the extended credit limit?"
 - "What if we cannot deliver the quantity?"
 - "What if data is corrupt in the database?"
 - (These are commonly overlooked situations)

Summary

- In this lesson, you have learnt:

- The use case model provides a complete, black-box, outside-in view of system functionality.
- A use case is a procedure by which an actor may use the system.
- A good use case is largely a sequence of interactions across the system boundary between the actor(s) and the system written in easy to understand natural language.



Summary

- In this lesson, you have learnt:
 - An actor is always outside the system being defined
 - The actor that starts the use case is the primary actor for that use case
 - An actor is a role defined by the set of use cases with which it is associated



Review Question

- Question 1: _____ is very effective elicitation technique.
- Question 2: Actors live outside the boundary of the system.
 - Option: True / False
- Question 3: Postconditions are basically conditions to start an execution of test case.
 - Option: True / False



Review Question: Match the Following

1. Precondition

2. Actor

3. Errors

4. Post condition

5. Exception

A. Non-recoverable extensions

B. Expectedly go wrong

C. These are not tested within the use case

D. Unexpectedly go wrong

E. Must be true on successful completion of the use case

F. Live outside the boundary of the system



Use Cases and Use Case Level Test Cases

Lab Book

Document Revision History

Date	Revision No.	Author	Summary of Changes
12/8/09	1	Priya Rane	Material Revamp
30/6/2011	1.1	Vaishali Kunchur	Material Revamp
12/7/2015	2.0	Shilpa Bhosle	Material Revamp

Table of Contents

<i>Document Revision History.....</i>	<i>2</i>
<i>Table of Contents</i>	<i>3</i>
<i>Lab 1. Use Cases Basics</i>	<i>4</i>
<i>Lab 2. Creating Use Cases</i>	<i>9</i>
<i>Lab 3. Test Cases for Use case</i>	<i>13</i>

Lab 1. Use Cases Basics

Goals	<ul style="list-style-type: none">• Understand the process of creating Use Cases.• Learn to apply basic techniques for writing Use Cases.• Understand the application and write creative Use Cases
Time	90 minutes

Note - To perform this lab you need to refer to the Banking Exam Portal Case Study lab from the Requirements Validation and Functional Decomposition lab book. To write the effective Use Cases you need to capture all those details from the Requirements Validation and Functional Decomposition lab book as well as the details given in the lab book.

Precondition	<ul style="list-style-type: none">• Administrator is an employee of a Banking Exam Portal company• Administrator has an access to the Banking Exam Portal Website• Administrator has a valid user id and a password
Description	<ul style="list-style-type: none">• This use case describes the process by which administrator can log into the Banking Exam Portal system
Steps to Login to the system	<ul style="list-style-type: none">• Administrator visits the Banking Exam Portal URL www.bankersjob.com/exam using any standard browser.• Administrator enters his/her user-id and password and clicks on the login button in the section of a home page given for administrator login.• The system validates the user id and password combination with the database.• If the authentication is unsuccessful, a login failure message is displayed.• On successful login, the admin home page is displayed.

	<ul style="list-style-type: none"> On Successful login, the user-id and role fields are stored in the HTTP session for future use during that session.
--	---

1.1 Write a use case for the following with the help of given details.

1.1.1 Login – Administrator

Rules for login	<ul style="list-style-type: none"> All fields on the Login web page are mandatory The User Name field can accept only characters The User Name field should accept maximum 10 characters Passwords must have at least six characters. Passwords must use at least three of the four available character types: lowercase letters, uppercase letters, numbers, and symbols. <p>NOTE – Detailed information on the rules can be read from RVFD lab book</p>
------------------------	---

UI Prototype:

Banking Exam Portal

Welcome to the Banking Exam Portal!!

<p><u>List of Exams on the Portal</u></p>	<p>Admin Login</p> <p>User Name: <input style="width: 100%;" type="text"/></p> <p>Password : <input style="width: 100%;" type="password"/></p> <p style="text-align: center;"><input type="button" value="Submit"/></p>	<p>New Candidate? Click here to register</p> <p>Existing Candidate Login</p> <p>Registration No: <input style="width: 100%;" type="text"/></p> <p>Password: <input style="width: 100%;" type="password"/></p> <p style="text-align: center;"><input type="button" value="Submit"/></p>
---	--	--

1.1.2 Add Exam Details

Precondition	<ul style="list-style-type: none"> Administrator is an employee of a Banking Exam Portal company Administrator has an access to the Banking Exam Portal Website Administrator has a valid user id and a password Administrator has already logged in to the banking exam portal
Description	<ul style="list-style-type: none"> This Use Case describes the process by which

	administrator can add a new exam to the existing set of exams.
Steps to login to the system	<ul style="list-style-type: none"> Administrator clicks on the Add Exam Details link available on the Admin Home Page Administrator will be navigated to the Add an Exam web page Administrator fills all the required details to add a new exam to the banking exams database Administrator clicks on the Submit button to complete the process of adding a new exam
Rules for adding a new exam	<ul style="list-style-type: none"> All fields are mandatory The exam name should be unique The exam code will be auto-generated once the exam name is entered The commencement of online registration date should be a future date The closure date of an online application should be 30 days from the commencement date The date of the exam should be at least 60 days ahead of the commencement date The exam description and fees needs to be mentioned along with the exam details <p>NOTE – Detailed information on the rules can be read from RVFD lab book</p>

UI Prototype:

Banking Exam Portal

Back
Add an Exam
LogOut

Exam Name

Exam Code

Bank Name

Commencement of Online Registration

Closure of Online Application

Exam Date

Description about the exam

Examination Fees

1.1.3 Register (New User/Applicant)

UI Prototype:

Precondition	<ul style="list-style-type: none"> • New user/Applicant is not a member of the website • New user/Applicant has an access to the Banking Exam Portal Website
Description	<ul style="list-style-type: none"> • This Use Case describes the process by which a new user/applicant can register on the Banking Exam Portal website
Steps to login to the system	<ul style="list-style-type: none"> • New user/Applicant navigates to the home page and clicks on the "Click here" link to start the registration process • New user/Applicant will be navigated to the Register web page • New user/Applicant fills all the required details to register on the website • Applicant clicks on the Submit button to complete the process of registration
Rules for adding a new exam	<ul style="list-style-type: none"> • All fields are mandatory • Exam to be applied for will be selected from the dropdown box containing the list of all available exams on the banking exam website • The exam code will be displayed automatically as per the exam selection • First Name, Middle Name and Last Name fields should accept character data • Other details like DOB should be a valid date and Email ID should follow valid email id format • The educational qualification details like degree, subject, year of passing, percentage of marks should be entered • Photo and signature should be uploaded by the applicant as image files <p>NOTE – Detailed information on the rules can be read from RVFD lab book</p>

Banking Exam Portal			
Back		New Registration	
Logout			
Exam Applied For <input type="text" value="Select Exam Name"/>			
Exam Code : <input type="text"/>			
Personal Details			
First Name	<input type="text"/>	Middle Name	<input type="text"/>
Last Name	<input type="text"/>		
Mobile Number	<input type="text"/>	Confirm Mobile No	<input type="text"/>
Email Address	<input type="text"/>		
Date of Birth	<input type="text" value="Select Date"/>		
Gender	<input checked="" type="radio"/> Male <input type="radio"/> Female		
Marital Status	<input type="radio"/> Married <input checked="" type="radio"/> UnMarried		
Educational Qualification Details			
Graduation/Equivalent Passes <input type="text" value="Select Graduation"/>			
Degree/Stream Subject	<input type="text"/>		Year of Passing <input type="text" value="select Year"/>
%of Marks	<input type="text"/>		
Photograph and Signature			
Insert Signature	<input type="text" value="Browse..."/>	<input type="text"/>	
(Signature should be 10Kb to 20Kb)			
Insert Photograph	<input type="text" value="Browse..."/>	<input type="text"/>	
(Applicant's photo should be upto 50Kb)			
Save & Submit		Reset	Cancel

Lab 2. Creating Use Cases

Goals	<ul style="list-style-type: none"> Understand the application and develop creative Use cases.
Time	90 minutes

2.1: Supporting document for Leave application

Please note that all fields in the leave application form are Mandatory fields.

A. Pre Condition:

- a. User is employee of company and having access to Intranet
- b. User has leave balance for applying a leave

B. Description: User wants to register for Leave (1/2 day or more)

C. Steps for accessing Leave application:

1. User will access Intranet
2. Click on Employee Corner
3. Click on Attendance System
4. Select Leave Application – Leave application form will be displayed (Diagram 1)

D. Procedure for applying for leave:

1. ½ day leave
 - a. Specify start date of the leave (Using Calendar Option – Diagram 2) Dates are non-enterable and hence need to be selected from calendar.
 - b. If the half day is in the start date then choose check box for half day in start date
 - c. Specify in which session the half day is. (Default value is first session)
 - d. Specify end date of the leave. (Using Calendar Option – Diagram 2) Dates are non-enterable and hence need to be selected from calendar.
 - e. Since you are applying for half day leave start date should be same as end date
 - f. If the half day is in the end date then choose check box for half day in end date (Make sure that half day in start date check box is not checked in this case)
 - g. Specify in which session the half day is. (Default value is first session)
 - h. Specify reason of leave from the given list of leaves
 - i. Specify Reason of leave
 - j. Submit the leave application
2. 1 or more than 1 day leave
 - a. Specify start date of the leave
 - b. Specify end date of the leave
 - c. Specify type of leave from the list provided
 - d. Specify reason of leave

- e. Submit the leave application
- f. 1 day leave can be applied by choosing Half day in start date second session and half day in end date first session

E. Rules for leave application:

1. CL cannot be applied for more than 3 continues days
2. CL cannot be combined with any other type of leave
3. SL cannot suffix PL
4. PL cannot prefix SL
5. ML cannot be prefixed with any other type of leave and can be continued by SL / PL
6. FF can be combined with any leave either prefix or suffix but cannot be sandwiched between any leave (Weekends / Holidays)
7. FF cannot be applied in break-ups
8. Once the leave is approved it cannot be cancelled unless cancellation of leave is submitted and is sanctioned.
9. Pending leave (Leave before approval) can be cancelled.
10. When leave type is FF – Comp off dates should be selected by clicking on the link provided and using form displayed .(Diagram 3)
11. Leave cannot be applied for the future dates (Dates in the next year)
12. Leave cannot be applied for the dates where attendance is already marked.
13. Leave cannot be applied for Week offs and holidays.

Leave Application - Microsoft Internet Explorer

Address: <https://intranet.patni.com/sso/INARMS/application1.ASP>

ARMS - Leave Application [Back](#) [Home](#)

NAME : Madhura Gowaikar
 REPORTING TO : KULDEEP CHAWLA
 Note: All dates are in mm/dd/yyyy format
 Status 'S**' indicates the application for cancellation of sanctioned leave is pending.
 Compensatory Off (Comp Off) is applicable to employees upto the grade of Senior Consultants in : Software, Hardware & Support functions.
 Designation above Manager in software or its equivalent is not eligible for Compensatory Off.

[Help](#)
[Legends](#)
[Compooff Details](#)
[Current Year Absenteeism](#)

Compensatory Off Leave Balance			
Leave Code	Accumulated	Availed	Current Balance
FF	1	0	1

Leave Balance for 2005					
Leave Code	Opening Balance	Leaves Used	Current Balance	No. Of Times Used	Advance Used
-	-	-	-	-	-

Leave Balance for 2006					
Leave Code	Opening Balance	Leaves Used	Current Balance	No. Of Times Used	Advance Used
CL	4.5	0	4.5	0	0
SL	7.5	0	7.5	0	0

Leave Balance Adjustment				
Date	Leave	No.Of Days Adjusted	Added/Deducted	Reason

From Date:
 Is HalfDay In Start date: ☐
 session: ☒ 1st ☐ 2nd
 To Date:
 Is HalfDay In End date: ☐
 session: ☒ 1st ☐ 2nd
 Type Of Request:
 Reason For Leave:
 Submit

Figure 1

Leave Application - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss

Address <https://intranet.patni.com/sso/INARMS/application1.ASP> Go Links

Note: All dates are in mm/dd/yyyy format
 Status 'S*' indicates the application for cancellation of sanctioned leave is pending.
 Compensatory Off (Comp Off) is applicable to employees upto the grade of Senior Consultants in : Software, Hardware & Support functions.
 Designation above Manager in software or its equivalent is not eligible for Compensatory Off.

[Compoft Details](#)
[Current Year Absenteeism](#)

From Date: 06/07/2006
 Is HalfDay In Start date: ☐
 session: ☒ 1st ☐ 2nd
 To Date: 06/07/2006
 Is HalfDay In End date: ☐
 session: ☒ 1st ☐ 2nd
 Type Of Request: CL-CASUAL LEAVE
 Reason For Leave:
 Submit

Calendar - Microsoft Internet ...

May 2006

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3

Off Leave Balance

Availed	Current Balance
0	1

Balance for 2005

Current Balance	No. Of Times Used	Advance Used
-	-	-

Balance for 2006

Current Balance	No. Of Times Used	Advance Used
4.5	0	0
7.5	0	0

Balance Adjustment

Added/Deducted	Reason
-	-

Leave History (Last Five)

Leave Code	Apply Dt	From Dt	To Dt	Actual No. Of Leave Days	Status	Reason For Leave	Reason For Rejection
-	-	-	-	-	-	-	-

Done

Start | Inbox - Microsoft ... | Exploring - Sampl... | Leave Applicatio... | Leave applicatio... | Calendar - Mi... | Local intranet | 4:41 PM

Figure 2

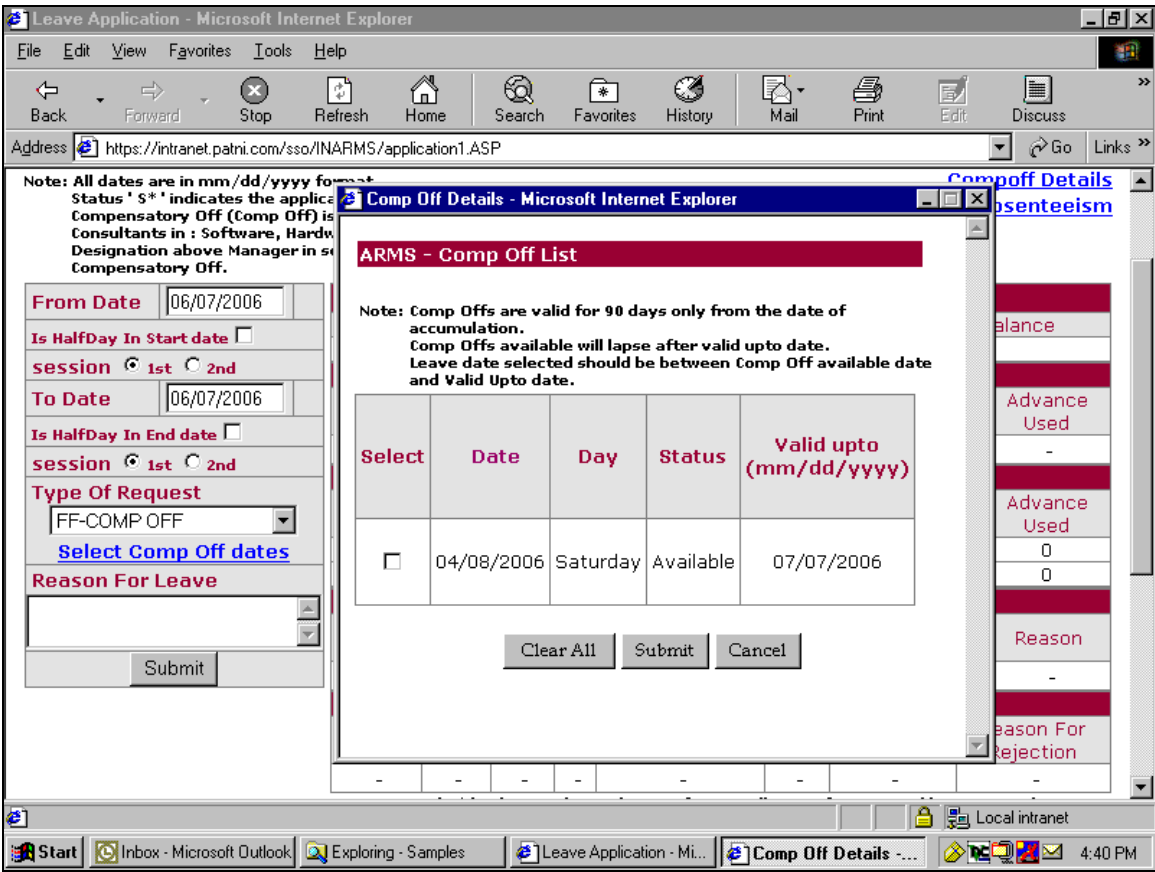


Figure 3

Lab 3. Test Cases for Use case

Goals	• Learn to write test cases for given Use cases description
Time	60 minutes

Transfer Funds Use Case Specification

Brief Description:

This use case allows account holder to transfer funds between two different accounts of this bank. This also includes the transfer between Current and Savings account of this bank.

Actor(s)

Actor	Description
Account Holder	Any user who has at least one account with the bank
Cur_ICENTURIAN	The external bank system that stores the current accounts
Sav_ICENTURIAN	The external bank system that stores the savings accounts

Preconditions

The user has successfully logged on to the banking system.

Flow of Events

Basic Flow

1. User initiates Transfer Funds use case
2. System retrieves the source accounts (i.e. the set of accounts owned by the account holder) and the destination accounts (i.e. accounts in this bank which have been activated to receive funds. By default all accounts owned by the account holder are activated to receive funds) from the external Bank Systems (Sav_ICCENTURIAN and Cur_ICENTURIAN)
3. System requests user to provide information like First Delivery Date, Frequency (Weekly, Every 2 Weeks, Monthly, Quarterly, half Yearly, Yearly), Number of Times and Transfer Amount
4. User provides necessary details to transfer funds
5. System validates the information entered by user (for validation criteria refer Business Rules Section)
6. On successful validation, System calculates the applicable transaction fee (if any)
7. System performs Insufficient Balance Check (refer Business Rule section) if the selected source account is current account
8. System displays the transfer details along with transaction fee details for user's verification.
9. User provides the verification and confirms the transfer
10. System completes the transfer and accordingly credits the destination account and debits the source account

Alternative Flows

Minimum Balance Check

This flow is executed at step 7 of the basic flow when the selected source account is savings account.

1. System performs Minimum Balance Check (refer Business Rule section).
2. On Minimum Balance Check failure, System displays appropriate message and ends the Use Case.
3. On successful Minimum Balance Check, System continues from the basic flow at step 8.

Exceptional Flows

Invalid Input

This flow is executed when there is a validation failure at step 5 of basic flow.

1. System displays the appropriate error message and asks user to enter the correct information.
2. System continues the use case from the basic flow at step 3.

Insufficient Balance

This flow is executed at step 7 of the basic flow when Insufficient Balance Check business rule is violated.

1. System displays the appropriate error message and ends the Use Case.

Post Conditions

Transfer Amount is successfully credited to destination account and debited from source account

Business Rules

1. First Delivery Date, Transfer Amount are mandatory
2. First Delivery Date > Today's Date
3. Following rules are applicable to Number of Times
 - a. It will be applicable only if Frequency is selected
 - b. It should be a positive integer
4. Transfer Amount should be a positive number
5. Insufficient Balance Check for Current Account
$$AB - (TA + TF) < 0$$

6. Minimum Balance Check for Savings Account
$$AB - (TA + TF) \geq 1000$$

Where, AB = Account Balance

TA = Transfer Amount

TF = Transaction Fee