NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

Assa

SE-220 – Software Design & Architecture Lab

LAB Instructor: Zarnain Maryam Awan

Lab 02

How to download and install Eclipse Papyrus:

Go to <https://eclipse.dev/papyrus/download.html> and download the 64 bit version of Papyrus for your Windows, Linux or OS platform.

Once the download is completed, extract the compressed folder. You will get a Folder ‘papyrus-win-64’. This folder contains a folder ‘Papyrus’ which contains a runnable ‘papyrus’, which you can simply open.

Select a workspace directory and click ‘Launch’.

**Use case Modeling:**

Use case modeling is a technique used in software engineering to document how different types of users interact with the system to solve a problem. A use case model describes the goals of the users, the interactions between the users and the system, and the required behavior of the system in satisfying these goals.

Here's a step-by-step guide to creating use case models:

1. **Identify Actors**: Actors are the entities (e.g., users, external systems) that interact with the system. Identify all the actors involved in the system. Actors can be human users or other systems. There are two types of actors:

Primary and Secondary Actors:

A primary actor initiates an interaction with the system.

The system initiates interactions with secondary actors.

For example:

A user clicks the search button on an application's user interface. The application sends the SQL query to a database system. The database system responds with a result set. The application formats and displays the result set to the user.

In this scenario the user is a primary actor because he initiates the interaction with the system (application). The database system is a secondary actor because the application initiates the interaction by sending an SQL query.

1. **Identify Use Cases**: Use cases represent the functionality provided by the system to its actors. They describe a sequence of interactions between the system and its actors to achieve a specific goal. Identify all the use cases by considering the goals and tasks that actors need to perform using the system.
2. **Define Use Case Descriptions**: For each identified use case, create a detailed description. This should include:
   * **Use Case Name**: A descriptive name that summarizes the purpose of the use case.
   * **Actors**: The actors involved in the use case.
   * **Description**: A brief description of what the use case accomplishes.
   * **Preconditions**: Any conditions that must be true before the use case can be executed.
   * **Basic Flow**: The main flow of events that occur when the use case is executed successfully.
   * **Alternative Flows**: Any alternate paths or exceptions that can occur during the execution of the use case.
   * **Postconditions**: The state of the system after the use case has been successfully executed.
3. **Identify Relationships**: Determine how actors and use cases are related to each other. Actors may initiate multiple use cases, and use cases may involve multiple actors. There are four types of relationships:

1.Association: It is represented by a line. Defines a relationship between an actor and a use case.

2.Include: It is represented by a dashed arrow pointing from the including use case to the included use case. It is a relationship between two use cases to indicate that one use case depends on the other. The Include Relationship indicates that a use case includes the functionality of another use case. This relationship promotes modular and reusable design.

3.Extend: It is represented by a dashed arrow with the keyword extend pointing from the extended use case to the extending use case. The Extend Relationship illustrates that a use case can be extended by another use case under specific conditions. This relationship is useful for handling optional or exceptional behavior.

4.Generalization: It is represented by an arrow pointing from the specialized use case to the general use case. The Generalization Relationship establishes an “is-a” connection between two use cases, indicating that one use case is a specialized version of another

OR

A relationship between actors to support re-use of common properties.

1. **Create Use Case Diagrams**: Use case diagrams provide a visual representation of the actors, use cases, and their relationships. Draw a diagram illustrating the actors, use cases, and their associations using standard UML notation.
2. **Refine and Validate**: Review the use case model to ensure completeness, consistency, and correctness. Validate the model with stakeholders to ensure that it accurately captures the system requirements.
3. **Iterate**: Use case modeling is an iterative process. As you gather more information and gain a better understanding of the system, refine and update the use case model accordingly.
4. **Maintain**: Keep the use case model up-to-date as the project progresses and requirements evolve. Regularly review and update the model to reflect any changes in the system.

By following these steps, you can create a comprehensive use case model that effectively captures the functional requirements of your system.

**Example Scenario**

The Automated Teller Machine is a remote unit connected to the bank computer systems; the bank computer system is responsible for verifying bank customers, authorizing transaction and recording completed transactions. The purpose of the system is to bring regular bank services closer to the customer and increase the working hours to around the clock. It is also important to decrease the amount of bank cashiers. An ATM withdrawal is less expensive for the Bank than a withdrawal from a teller.  
  
The ATM system requires that each bank customer has an ATM card and remembers his PIN code. The whole security of this system builds on the PIN code. The cashier responsibility is to count the money in the security box to verify all deposits. The maintenance person is responsible for maintaining the Automated Teller Machine, refilling paper, and replenishing cash.

**Usecase Modeling of the Scenario**

**1.Identify Actors**

**Bank Customer**

This actor represents a person with a valid Bank Card. The Bank Card is theirs and they know the PIN Code

**Cashier**

From the ATM system point of view, the Cashier's only responsibility is to count the money in the security box to verify all deposits.

**Bank**

This actor represents the financial institution that provides services to the ATM. Responsible for verifying Bank Customers, authorizing transactions and recording completed transactions.

**Maintenance Person**

This actor represents the person responsible for maintaining the Automated Teller Machine, refilling paper, and replenishing cash.

**2.Identify Use cases**

**Validate User**

This use case describes general behavior for the ATM to validate the Bank Customer. It includes all steps that are the same no matter what kind of transaction the Bank Customer does.

**Withdraw Cash**

This use case describes how the Bank Customer uses the ATM to withdraw money from his/her bank account.

**Transfer Funds**

This use case describes how the Bank Customer uses the ATM to transfer money between different bank accounts.

**Deposit Funds**

This use case describes how the Bank Customer deposits money to an account.

**Refill Machine**

This use case describes how the Maintenance Person refills money, receipt paper and envelopes.

**3.Usecase Description**

Example Of Withdraw Cash Use case:

Use case Name: Withdraw cash

Actor: Bank Customer

Use case description: This use case describes how the Bank Customer uses the ATM to withdraw money from his/her bank account.

Preconditions: The validate user use case must have been successfully completed (in other words The customer must have a valid ATM card and PIN number). The ATM must be functioning properly and connected to the banking network.

Basic Flow:

1. Insert card
2. Enter pin
3. Select withdraw
4. Select type of account
5. Select amount

7. Send transaction

8. Receive ok

9. Dispense money

10. Print receipt

11. Eject card

List of Alternative flows

A1 wrong PIN

A2 no money

A3 Attempt to withdraw more than daily amount

A5 link goes down ­ hmm, if the link goes down before the transaction reaches the actual account? ­ No big fuss ­ but if the transaction reaches the account and then the link is down? The money is withdrawn but never dispensed!

A6 stolen card

A7 out of money

Postconditions: The cash is dispatched and the amount has been withdrawn from the

selected account.

**Use case Diagram:**

