UML modelling & Use Case Diagrams

Overview

Unified modelling Language (UML) is a widely used modelling approach that is used in systems analysis and design. Some of its 14 diagrams provide an overview of certain aspects of a system while others provide very detail information.

Why do analysts and designers develop models?

- Information Technology systems are often very complex and difficult to understand
- Models are abstractions of systems: in software modelling abstraction is the process removing unnecessary detail from a system model while maintaining the set of essential characteristics.
- Models can make certain aspects more clearly visible than in the real system
- They provide focus on certain aspects of the system.
- Often they are and easier to understand than lengthy word based descriptions.

The UML diagrams are categorised as Structural Diagrams or Behavioural Diagrams. (See over)

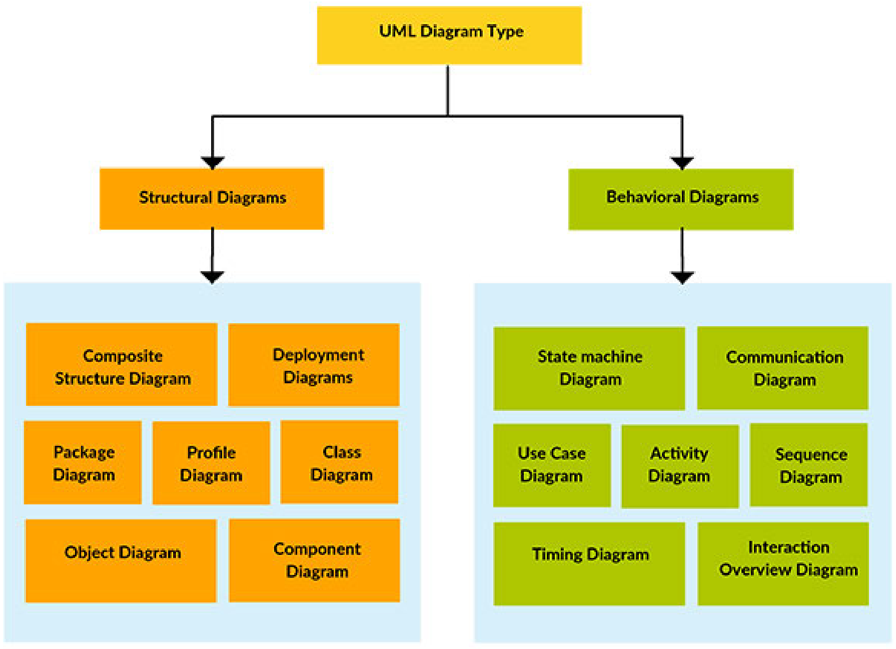
Structural Models (diagrams)

Structural models depict a view of a system that focuses on the structure of the system in terms of the components and their relationships. For Instance a Class Diagram captures the logical structure of the system, the Classes and objects that make up the model, its attributes and behaviour.

Behavioural Diagrams

Behavioural diagrams depict the dynamic aspects or behaviour of the system. As an example a Use Case diagram captures details of the system tasks, actors (usually people), and the relationships between the Actors and the system.

UML modelling & Use Case Diagrams



UML Diagram types

Source : Creately http://creately.com/blog/diagrams/uml-diagram-types-examples/ accessed 6/3/2017

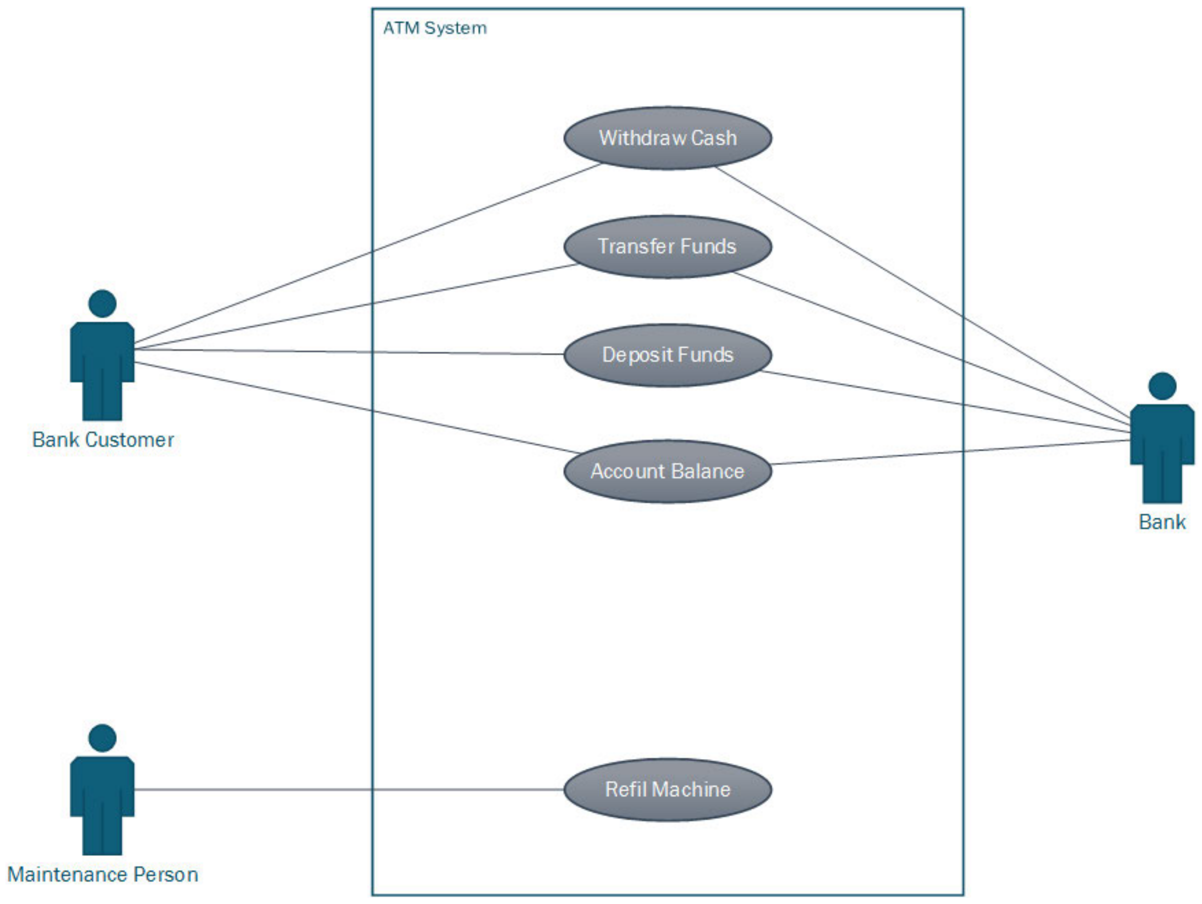
Use Case Diagrams

A use case is a behaviour that describes the interactions between one of more Actors and the system in order to record the intended behaviour of the system. The functionality of a system is defined by different use cases, each of which represents a specific goal for a particular actor. **Use Case diagrams document functional requirements or tasks that must be performed by the system.**

In an automated teller machine shown in Figure 1, the Bank Customer can withdraw cash from an account, transfer funds between accounts, or deposit funds to an account. These correspond to specific goals that the actor has in using the system.

UML modelling & Use Case Diagrams

An example of a small Use Case Diagram



Source http://epf.eclipse.org/wikis/openup/ Accessed 10/5/2016

The elements of a use case diagram are intended to be intuitive, even for a person who is unfamiliar with the notation. The diagram will be used by analysts and stakeholders to come to a clear understanding of the main functions of a particular business.

The names of the Use Cases should be kept short. Two or three words that describe the task is what we want.

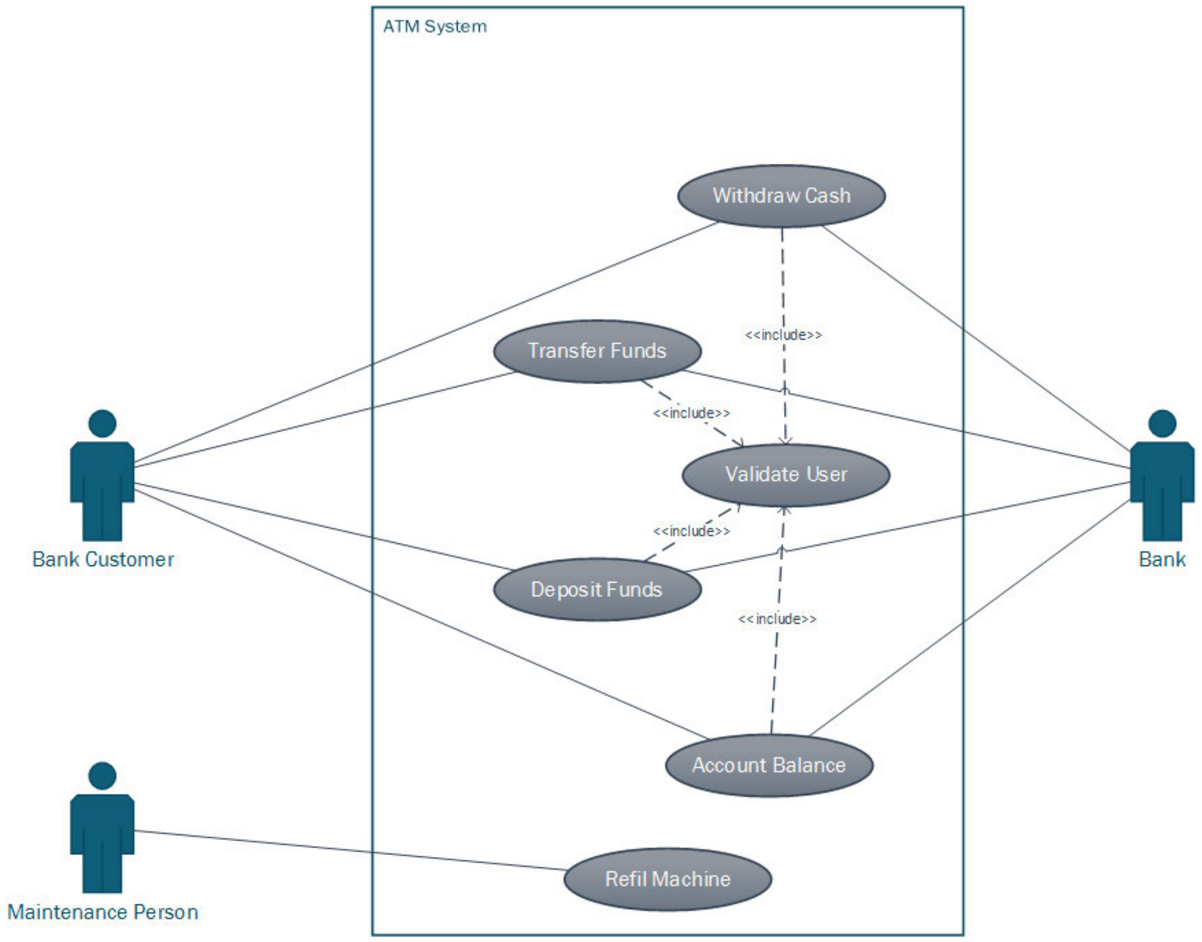
Each use case is associated with at least of one of the actors. The collection of use cases constitutes all the possible tasks included in the system. You should be able to determine the goal of a use case simply by observing its name.

When you have finished the Use Case Diagrams for a system they should document all of the system’s functional requirements except the System Wide Functional requirements that are documented in the *System Wide Requirements.*

UML modelling & Use Case Diagrams

Documenting Re-usability

<<**include**>> is used to include behaviour from an included use case into a base use case in order to support reuse of common behaviour. Only use this if the behaviour or task is required by other use cases as well.



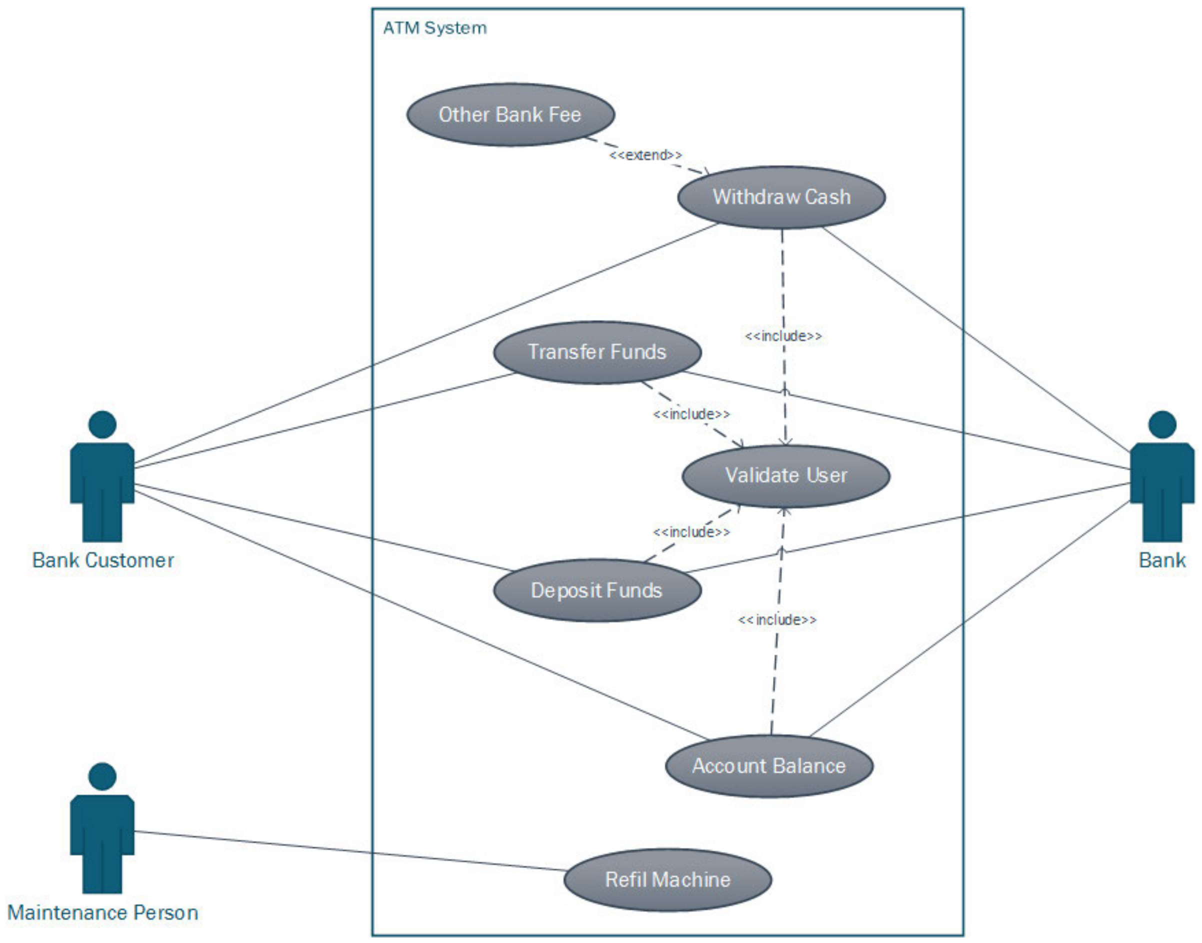
The diagram above shows the use case Validate User. The task of validating the user needs to be performed when a customer withdraws Cash, Transfers Funds, Deposits funds & gets an account balance and that is why it is defined as a separate use case that is included in the other four use cases.

UML modelling & Use Case Diagrams

Documenting Optional behaviour.

<<**Extends>>** is used when a use case adds steps to another use case. It is an optional item that may occur.

Consider the diagram below: Sometimes the withdrawal is for another Banks customer. If it is then an additional fee is charged, if not the optional steps are not used. Only use this if the task is optional.



UML modelling & Use Case Diagrams

Sets of Use Case Diagrams

In many systems the use cases (tasks) that are part of the system cannot all be represented on a single diagram because there are too many of them. If that occurs the best approach is to create several diagrams representing different parts of the system (sub-systems)

As an example consider the Mavis Car rental System. We might create Use Cases Diagrams for the following Sub-systems

Vehicle Rental Activities including (doesn't include all tasks).

- Adding a new customer
- Changing Customer details
- Making a reservation
- View vehicle details
- Renting a vehicle
- Returning a vehicle
- Making a payment
- Recording vehicle damage
- Updating customer Details

Vehicle Management including (doesn't include all tasks).

- buying new cars
- display car details
- editing car details
- selling the cars when they need to be replaced
- setting (adding) the fees for the vehicle type
- viewing the fees for a vehicle type
- organizing vehicle services

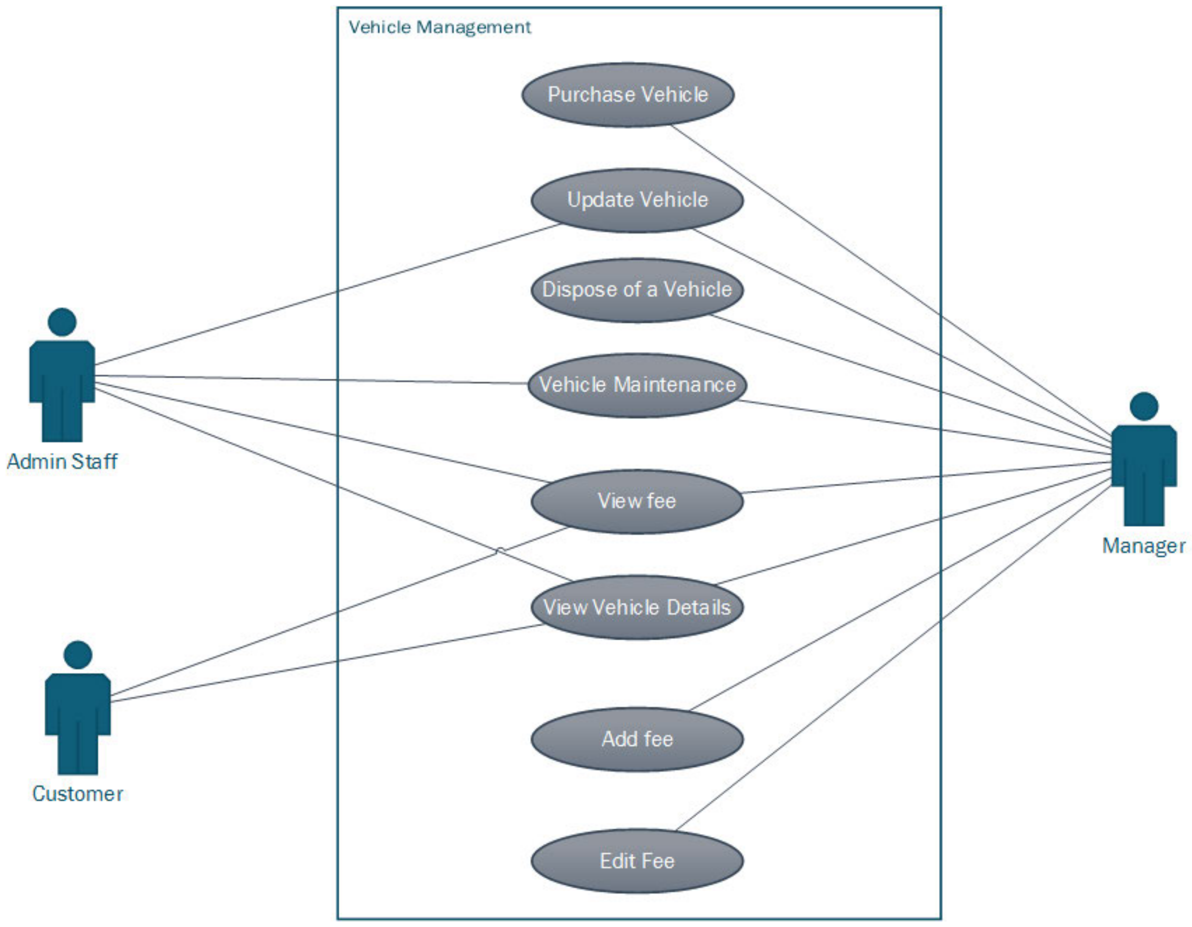
Management Activities including the following tasks (doesn't include all tasks).

- Daily Rental Report
- Setting up new Branches
- Updating existing Branch details
- Hiring new staff (add staff)
- Adding new Vehicle types
- Updating Vehicle types

Continued on the next page…

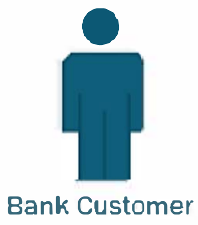
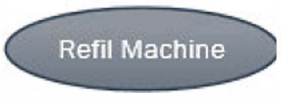
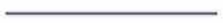
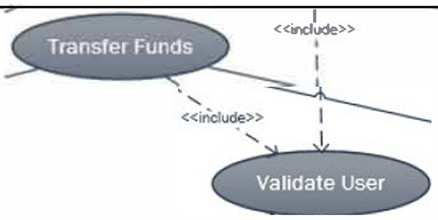
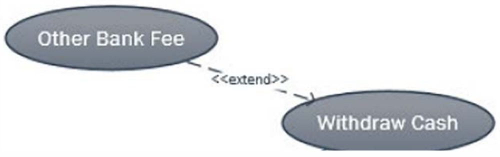
UML modelling & Use Case Diagrams

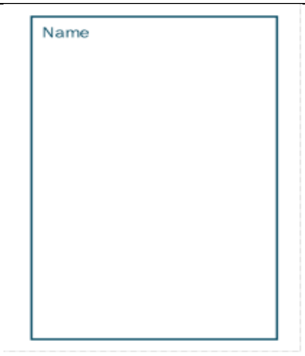
The Use Case Diagram for the Vehicle Management sub-system is shown below.



UML modelling & Use Case Diagrams

Use Case Diagrams have a number of components. The core components are described below.

Diagram Component	Shape	Description
Actor		An actor is anything outside a software system that interacts with it. For example, in a system that allows people to buy goods over the Internet, the human users will be significant actors, but so too will be the credit card system that enables users to pay for their purchases. Components of your system such as your database or web site are not Actors.
Use Case		A use case represents a typical task that would be carried out by the interaction of the actor and this system. In the ATM system shown above refilling the ATM with cash would be one of the tasks the maintenance person undertakes. The detailed description of each use case is held elsewhere and we will look in coming classes.
Association		Associations between actors and use cases are indicated in use case diagrams by solid lines. An association exists whenever an actor is involved with an interaction described by a use case. Associations are modelled as lines connecting use cases and actors.
Include Relationship	 Transfer Funds <<include>> <<include>> Validate User	<<include>> is used to include behaviour from an included use case into a base use case in order to support reuse of common behaviour.
Extend Relationship		<<Extends>> is used when a use case adds steps to another use case. It is an optional task that may occur. E.g. Sometimes the withdrawal is for another bank's customer. If it is then an additional fee is charged.

System Boundary		The system boundary, depicted by a rectangle surrounding the use cases is an important conceptual line that separates the system we are interested in from the rest of the world. By drawing the boundary around the system represented in a use case diagram, you are establishing the scope of your solution.

Additional Information can be located at

http://epf.eclipse.org/wikis/openup/

OpenUP, a product of the Eclipse organisation

http://www.agilemodeling.com/artifacts/useCaseDiagram.htm and
http://www.agilemodeling.com/style/useCaseDiagram.htm

(Scott W. Amlyer: Agile Modelling)