

GUJARAT TECHNOLOGICAL UNIVERSITY

Chandkheda, Ahmedabad

Affiliated



Birla Vishwakarma Mahavidyalaya
Engg. College, V.V. Nagar

A
Project Report
On

E - PAPER

Under subject of
DESIGN ENGINEERING – II-A
B.E. III, Semester – V
IT Branch

Submitted by:
Group:

Sr.	Name of Student	Enrollment No.
1	Bhansali Dhwani	130080116001
2	Bhavsar Bhavik	130080116002
3	Gabani Jaydeep	130080116013
4	Kalariya Hit	130080116016

Prof. Chintan Mahant
Faculty Guide

Prof. Zankhana H. Shah
Head of Department

Academic Year
(2015-16)

CERTIFICATE

This is to certify that project entitled with “**E-PAPER**” has been carried out by **DHWANI BHANSALI, BHAVIK BHAVSAR, JAYDEEP GABANI, HIT KALARIYA** under my guidance in fulfillment of 5TH semester subject Design Engineering of Bachelor of Engineering in Information Technology Department during the academic year 2015-16.

Date: 16/10/2015

Place: Birla Vishvakarma Mahavidyalaya Engineering College.

Project Guide: Prof. Mr. Chintan Mahant

TABLE OF CONTENTS

TITLE PAGE	I
TABLE OF CONTENTS	III
LIST OF TABLES	IV
LIST OF FIGURES	V
Chapter 1 Introduction to PD canvas	1
Chapter 2 AEIOU Framework for Observation	3
2.1 Activity Record Sheet	3
2.2 Environment Record Sheet	4
2.3 Interaction Record Sheet	5
2.4 Object Record Sheet	6
2.5 User Record Sheet	7
Chapter 3 Review Report of Prior Art Search	8
3.1 Review of saving paper in school	8
3.2 Review of online bus ticketing	9
3.3 Review of e-paper	10
Chapter 4 Learning Need Matrix	11
Chapter 5 Snapshot of Fast Prototype	12
Chapter 6 Data dictionary	15
Chapter 7 Diagrams	16
7.1 Use case diagram	17
7.2 Sequence diagram	22
7.3 Activity diagram	30
7.4 Entity-Relationship diagram	43
7.5 Data Flow Diagram	46

Chapter 1: Product Development

Product Purpose:

- Save paper
- Save trees and time
- User satisfaction

Product Experience:

- Easy to use and very efficient
- Awesome
- No worries for losing tickets

Product Functions:

- Calculation: - calculate total price and travelling charge.
- Check : - can see the bill and tickets or receipt any time.
- Compatibilities: - useful in shopping mall, shops, toll booths, bus/railway stations and air ports, etc.
- Filters: - message or pdf.

Product Features:

- Multiple-language support
- Easy to keep record.
- No worries of losing ticket/receipt/bills.
- No need of paper/printer.

Product Components:

- Computers : - to make bills
- Mobile : - to receive bills/tickets/receipts along with code in form of message
- Messaging software : - to send bill/tickets/receipts
- Code generator : - to generate code
- Code matcher : - to match the generated code
- Server and database : - to keep records
- Internet : - to run messaging software online
- Mobile network : - to get texts

Customer revalidation:

- By talking to other people about this product , we came to know that most of the people are satisfied with the E – PAPER as it saves a lot of paper.

Function reject, redesign and retain:

- They suggested us to remove barcode as it was firstly added by us.
- After some suggestions, barcode is replaced by simple code.

Chapter 2: AEIOU Framework

2.1 Activities:

General impressions/observations

- People standing in queue
- People buying stuff
- Employees making bills
- Person misplaced his ticket
- Civilians paying toll – tax
- Checking of paid bills

Elements, features and special notes

- Messaging software that reduces waste of paper and makes easy to save message than paper
- Code matcher that reduces human efforts and saves time of user and keeps record.

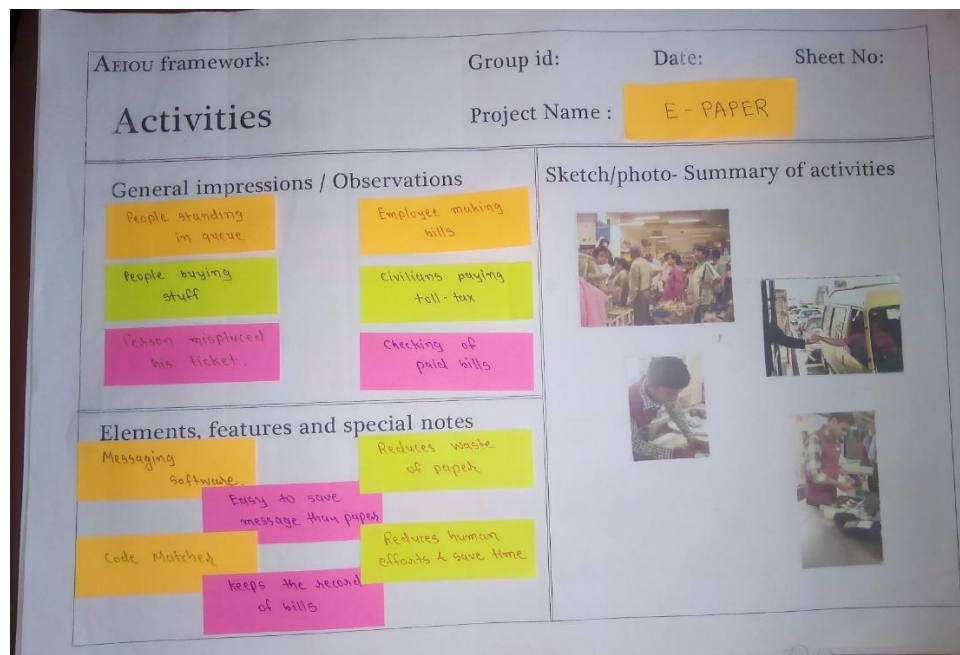


Figure 1 (Activity canvas)

2.2 Environment:

Impressions/Observations:

- Rainy
- Crowd
- Less customers
- Sale/vacation/festival
- Huge to more customers
- Requires more time for billing

Elements, features and special notes

- Dustbins
- Ticket checker
- Printer
- Security guards
- Parking


A EIOU framework:		Group id:	Date:	Sheet No:						
Environment		Project Name :	E-PAPER							
General impressions / Observations (Style, materials & atmosphere)		Floor plan								
<table border="1"><tr><td>Rainy season</td><td>Less customers</td></tr><tr><td>Crowd</td><td>Requires more time for billing</td></tr><tr><td>Sale / vacation / festivals</td><td>Huge waste of paper due to more customers</td></tr></table>					Rainy season	Less customers	Crowd	Requires more time for billing	Sale / vacation / festivals	Huge waste of paper due to more customers
Rainy season	Less customers									
Crowd	Requires more time for billing									
Sale / vacation / festivals	Huge waste of paper due to more customers									
Elements, features and special notes		Scene								
<table border="1"><tr><td>Dustbins</td><td>Printer</td></tr><tr><td>Conductor</td><td>Parking</td></tr><tr><td>Security guards</td><td></td></tr></table>		Dustbins	Printer	Conductor	Parking	Security guards				
Dustbins	Printer									
Conductor	Parking									
Security guards										

Figure 2 (Environment canvas)

2.3 Interactions:

Impressions/observation

- Customer to customer/helper/cashier
- Travellers to travellers/guide/driver
- Purpose of interaction is asking about products/rates of tickets/bargaining/normal chat

Elements, features and special notes

- Shopping malls
- Bus stations
- Rail – way stations
- Toll booths




AEIOU framework:		Group id:	Date:	Sheet No:
Interactions		Project Name : E-PAPER		
General impressions / Observations (Who is interacting with whom, what?)		Scene of interection (How it is being done)		
<div>Customers</div> <div>Helpers / cashier</div> <div>Citizens</div> <div>Travellers / conductor</div> <div>Travellers</div> <div>Guide / Driver</div> <div>About product / rates of tickets / Bargaining / Normal chat</div>		  		
Elements, features and special notes				
<div>Shopping malls</div> <div>Bus - station</div> <div>Railway - station</div> <div>Toll - booths</div>				

Figure 3 (Ideation canvas)

2.4 Objects:

Impressions/Observations

- Mobile
- Computer
- Internet connection
- Bags/luggage/purse
- Listening music
- People using mobiles
- People waiting for bus/train
- Throwing rubbish in dustbin
- Purchasing grocery

Elements, features and special notes

- Puzzle/crossword
- Books/newspapers
- Mobile phones
- Food

AEIOU framework: Group id: Date: Sheet No:

Objects Project Name : **E - PAPER**






General impressions / Observations (What components are involved?)	Inventory of key objects
<p>Bag/Luggage/Purse</p> <p>Listening music via Ear phones</p> <p>People using mobile phones</p> <p>People waiting for bus/train</p> <p>Throwing rubbish in dustbin</p> <p>Purchasing grocery</p>	    
Elements, features and special notes (How objects are relating to the activities?)	
<p>Puzzle / crossword</p> <p>Books / Newspapers</p> <p>Mobile phones</p> <p>Food</p>	

Figure 4 (Object canvas)

2.5 Users

Imperations/Observation

- Civilians waiting for bus/train
- Customers buying stuff
- Shopkeeper making bills
- Customers taking cart to counter
- Driver paying toll - tax

Elements, features and special notes

- Students
- Politicians
- Businessman
- Employees
- Senior citizens
- House wives
- Travellers
- Farmer
- Car/bus drivers
- Shopkeepers

AEIOU framework: Group id: Date: Sheet No:

Project Name : E- PAPER

Users


General impressions / Observations (Who is present roles & responsibilities?)	Scene of users in context
<ul style="list-style-type: none">Civilians waiting for Bus/TrainCustomers buying stuffShop-keeper making billsCustomers taking cart to the counterDriver paying toll-tax	
<h3>Elements, features and</h3> <p>(List)</p> <ul style="list-style-type: none">StudentsEmployeesPoliticiansSenior citizensHousewivesTravellersFarmerCar/Bus DriversShop-keepers	

Figure 5 (User canvas)

Chapter 3 : Review Report

3.1 Fact sheet: Saving paper in schools

Why It Matters

Producing and disposing of paper pollutes land, air and water, threatens wildlife and affects human health.

A single sheet of paper may seem like an insignificant thing. But the use of thousands of sheets each day can have a huge impact on the environment. That's especially true in schools, where students and teachers and other staff all use lots of paper.

It all starts when trees are cut down, which hurts both forests and the animals that live in them. Cutting down forests even affects the earth's climate, since trees absorb carbon dioxide, one of the greenhouse gases that cause global warming. (The paper industry also creates lots of greenhouse gases -- it's the third largest source of global warming pollution in the world!)

Project Ideas

- **Paper proposals:** Take some time to observe how paper is used in your school. You'll probably discover examples of waste. Make a list of five to 10 ways your school can use paper more efficiently, then present it to your principal.
- **Scrap pile:** Choose a spot in your class for collecting paper that can be reused. Some of it might make good scribble paper. Or you could collect some bigger sheets and staple them together, perhaps on a cardboard backing, to make assignment pads. Colored paper is good for art projects -- try cutting it up to use for "mosaics."
- **Make your own recycled paper:** Making your own paper is a fun project. And you'll be doing your own recycling! There are lots of books and websites to help you. Try this [guide](#) from the Exploratorium.
- **Paper review:** Find out what kind of paper your school buys for its offices and classrooms.

3.2 Development of an Online Bus Ticket Reservation System for a Transportation Service in Nigeria

E-ticketing could be extended to major entertainment and touristic sites and thus facilitate access to major points of interest within cities, making e-ticketing also interesting for travelers. Urban tourism is the fastest growing tourism sector in the world (Paskaleva, 2014).

Public transport operators have been trying to replace paper-based tickets with electronic media, and many countries have implemented or are about to introduce e-ticketing systems. The main characteristic of e-ticketing is that tickets are sold and stored in electronic devices.

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features:

- It will ensure data accuracy.
- Records will be efficiently maintained by DBMS.
- Availability of seats can be enquired easily.
- Passengers can also cancel their tickets easily.
- Minimum time needed for the various processing.
- It will provide better Service.

3.3 E - Paper

Basic Features:

- User can get a copy of bill/ticket/receipt through mobile phone.
- It's easy to keep record of customers.
- No worries of storing bill/ticket/receipt in paper format.

Key features:

- Saves paper that is saving nature itself.
- Saves time.
- Reduces the efforts of human beings.

Compared to similar project:

- Instead of recycling of paper we are just removing the paper by providing E-PAPER.
- The second project is as similar as ours but we are not providing online transactions.

Chapter 4 : Learning need matrix

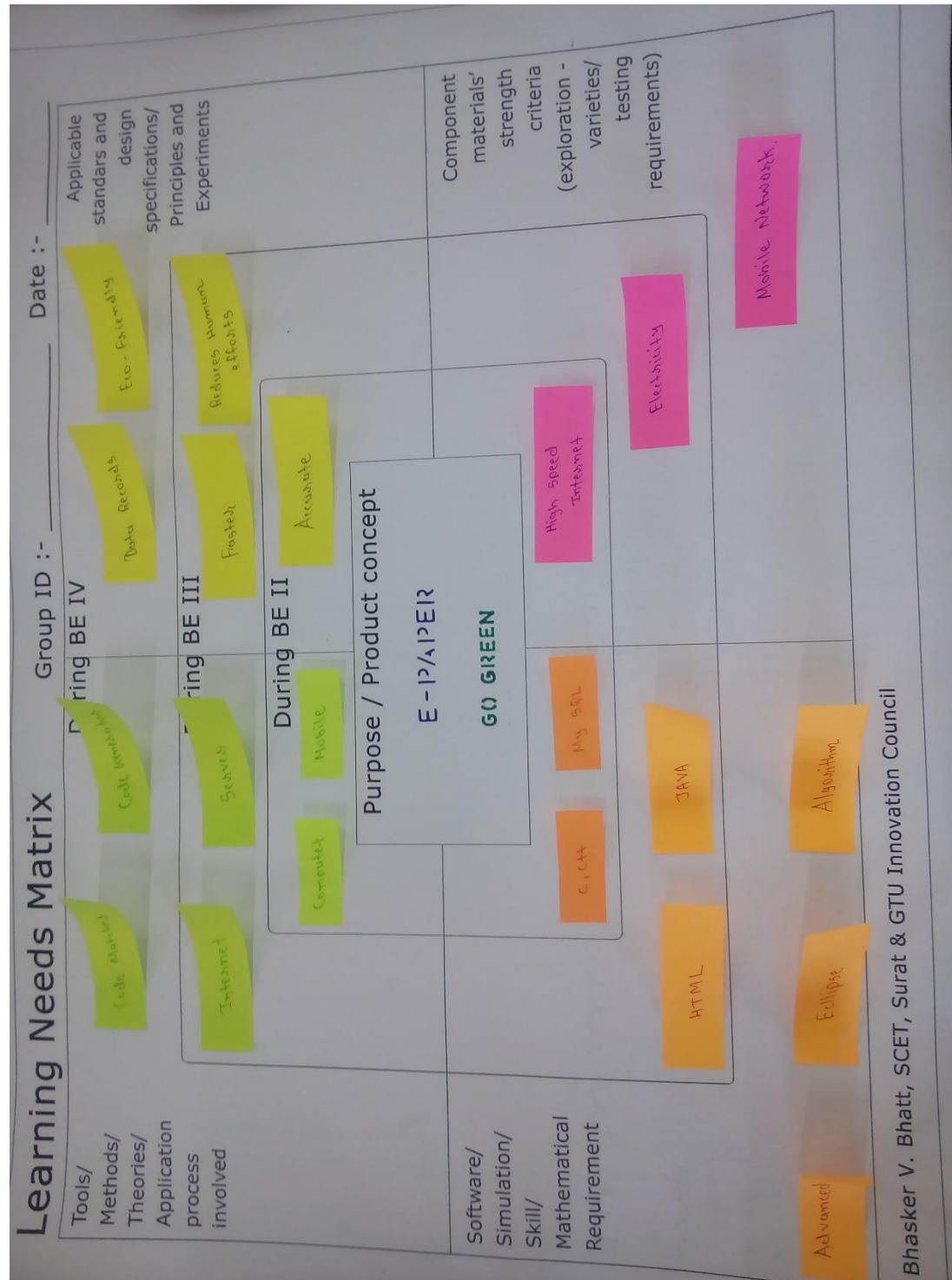


Figure 6 (Learning need matrix)

Chapter 5: Fast prototype

For shopping malls/stores:

<input type="text" value="Name of store/mall."/>			<input type="text" value="Bill number:"/>
<input type="text" value="Item"/>	<input type="text" value="Qty"/>	<input type="text" value="Amt"/>	<input type="text" value="Mobile number"/>
<input type="text" value="Total"/>			<input type="text" value="Code"/>
			<input type="button" value="SEND"/>

For toll – booth receipt

Toll - Booth	
<input type="text" value="Vehical No."/>	<input type="text" value="Mobile No."/>
<input type="text" value="One/two way"/>	<input type="text" value="Vehical type"/>
<input type="text" value="Amt"/>	<input type="text" value="Send"/>

For ticketing

Ticket	
Name	Mobile No.
From	To
Date of journey	No. of seat(s)
Adult/child/sc	Amt
	Send

Chapter 6: Data Dictionary

Customer		
Entity	Data type	Size
Mobile no	number	10
Amount	number	10

Table 1 (customer)

Item		
Entity	Data type	Size
Item no	Varchar2	10
Price	number	10

Table 2 (item)

Toll Tax		
Entity	Data type	Size
Mobile no	number	10
Receipt no	Varchar 2	10
Vehical no	Varchar 2	10
Vehical Type	Varchar 2	10
One/Two way	Varchar 2	3

Table 3 (toll-tax)

Bill		
Entity	Data type	Size
Mobile no	number	10
Bill no	Varchar 2	10
Item no	Varchar 2	10
Qty	number	6
Price	number	10
Code	Varchar 2	10
Total	number	10

Table 4 (bill)

Ticket		
Entity	Data type	Size
Mobile no	number	10
Ticket no	Varchar 2	10
Source	Varchar 2	20
Destination	Varchar 2	20
No of Adults	number	2
No of Child	number	2
No of Senior cit.	number	2
No of seats	number	2
Distance	number	4
Fair	number	5

Table 5 (ticket)

Chapter 7: Diagrams

7.1 Use case diagram:



Use case diagrams represent the functions of a system from the user's point of view. A use case diagram establishes the capability of the system as a whole


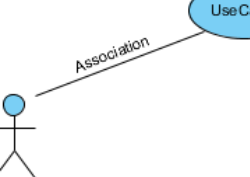

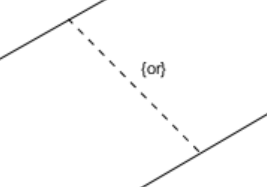
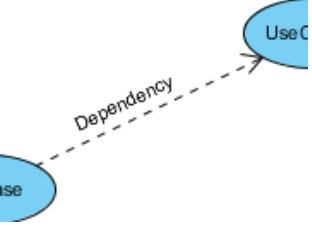
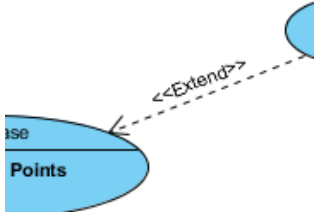
- It is a snapshot of one aspect of system.
- They model a dialog between actor and system.
- A use case typically represents a major piece of functionality that is complete from beginning to end.
- Most of the use cases are generated in initial phase, but you find some more as you proceed.
- A use case may be small or large. It captures a broad view of a primary functionality of the system in a manner that can be easily grasped by non-technical user.
- A use case must deliver something of value to an actor.
- The use cases may be decomposed into other use cases.

Use:

- Use cases are a valuable tool to help understand the functional requirements of a system.
- It is important to remember that use cases represent an external view of the system.
- As such, don't expect any correlations between use cases and the classes inside the system.

Symbol:

Symbol	Description
	System: If a subject (or system boundary) is displayed, the use case ellipse is visually located inside the system boundary rectangle.
	Actor: An Actor models a type of role played by an entity that interacts with the subject (e.g., by exchanging signals and data), but which is external to the subject (i.e., in the sense that an instance of an actor is not a part of the instance of its corresponding subject). 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.

	<p>Use Case: A use case is the specification of a set of actions performed by a system, which yields an observable result that is, typically, of value for one or more actors or other stakeholders of the system.</p>
	<p>Association: An association specifies a semantic relationship that can occur between typed instances. It has at least two ends represented by properties, each of which is connected to the type of the end. More than one end of the association may have the same type.</p> <p>An end property of an association that is owned by an end class or that is a navigable owned end of the association indicates that the association is navigable from the opposite ends; otherwise, the association is not navigable from the opposite ends.</p>
	<p>Collaboration: A collaboration specifies a view (or projection) of a set of cooperating classifiers. It describes the required links between instances that play the roles of the collaboration, as well as the features required of the classifiers that specify the participating instances. Several collaborations may describe different projections of the same set of classifiers.</p>
	<p>Constraint: A condition or restriction expressed in natural language text or in a machine readable language for the purpose of declaring some of the semantics of an element.</p>
	<p>Dependency: A dependency is a relationship that signifies that a single or a set of model elements requires other model elements for their specification or implementation. This means that the complete semantics of the depending elements is either semantically or structurally dependent on the definition of the supplier element(s).</p>
	<p>Extend: This relationship specifies that the behaviour of a use case may be extended by the behaviour of another (usually supplementary) use case. The extension takes place at one or more specific extension points defined in the extended use case. The same extending use case can extend more than one</p>

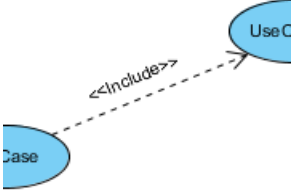
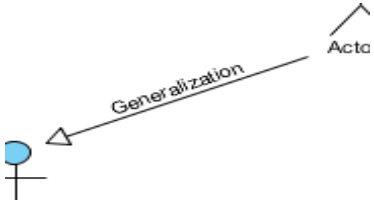
	use case. Furthermore, an extending use case may itself be extended.
	<p>Include:</p> <p>Include is a Directed Relationship between two use cases, implying that the behaviour of the included use case is inserted into the behaviour of the including use case. The including use case may only depend on the result (value) of the included use case. This value is obtained as a result of the execution of the included use case.</p>
	<p>Generalization:</p> <p>A generalization is a taxonomic relationship between a more general classifier and a more specific classifier. Each instance of the specific classifier is also an indirect instance of the general classifier. Thus, the specific classifier inherits the features of the more general classifier.</p>

Table 6 (Use case Symbol Table)

Use case Diagram:

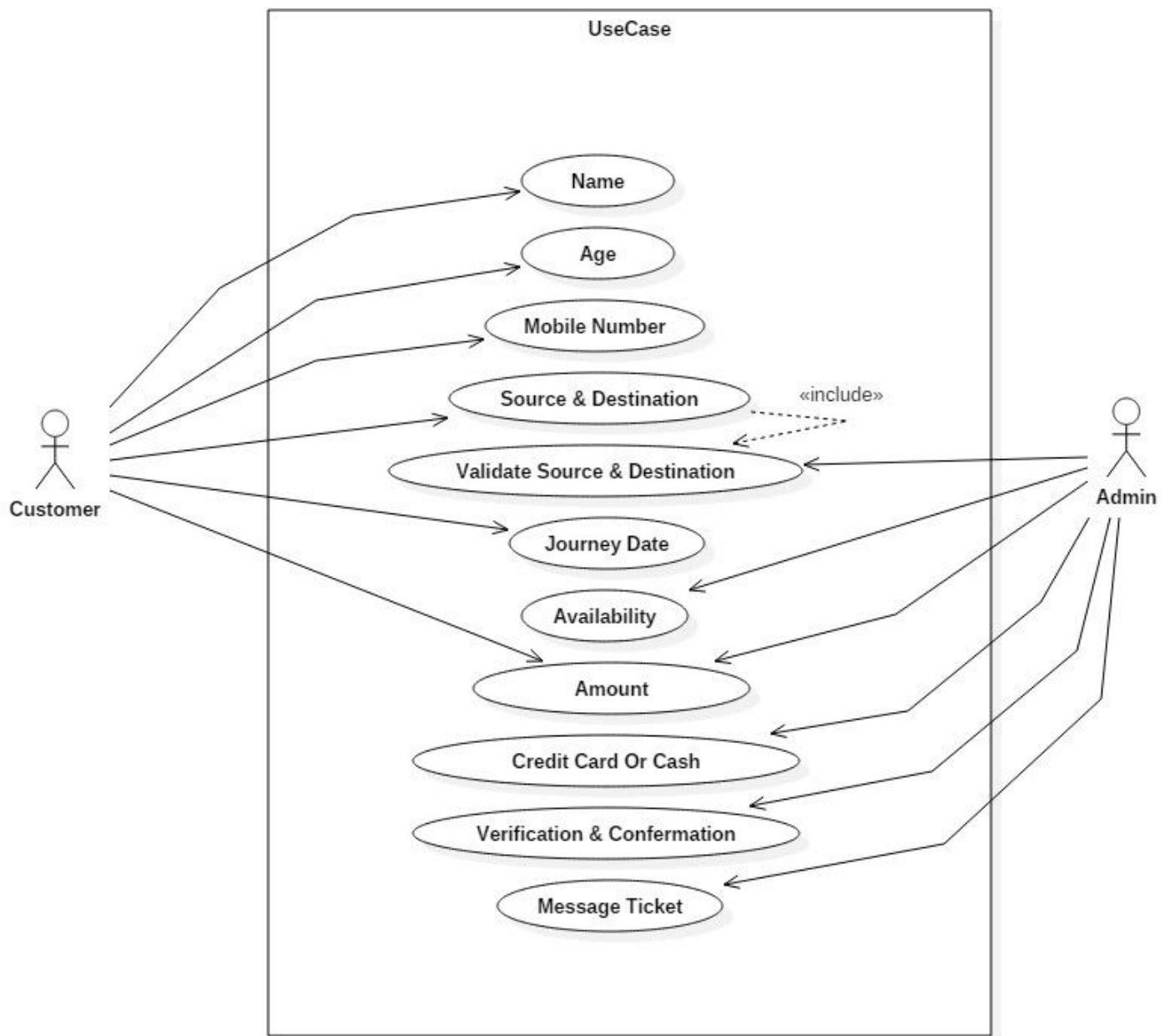


Figure 7 (Usecase diagram)

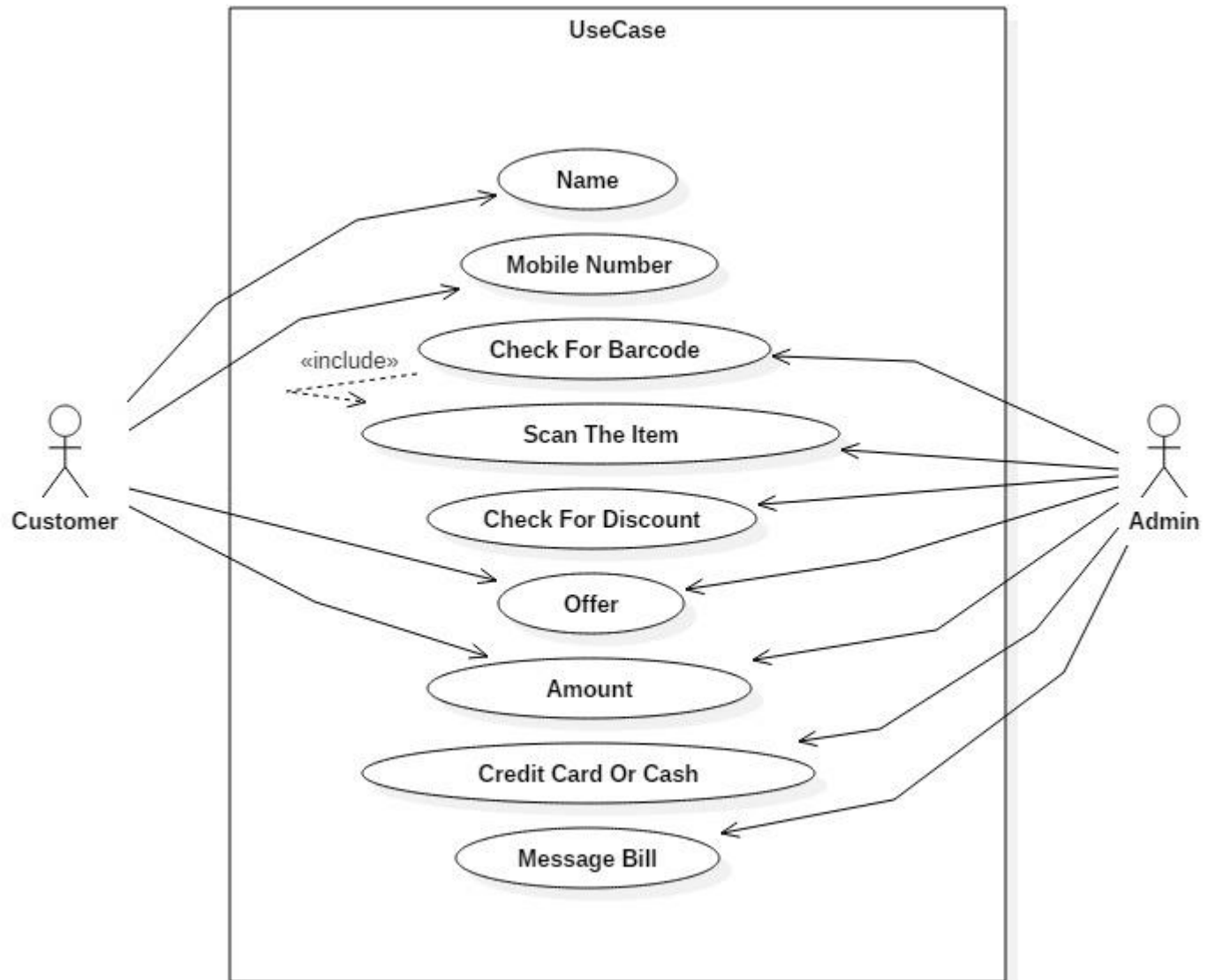


Figure 8 (Usecase diagram)

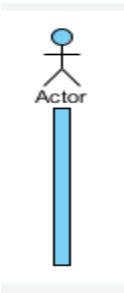
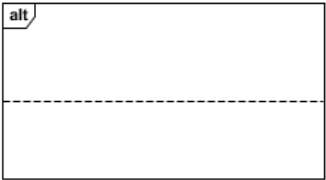
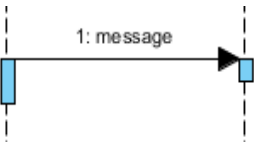
7.2 – Sequence diagram:


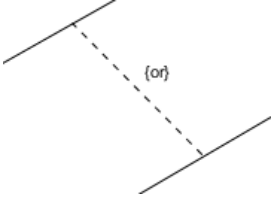
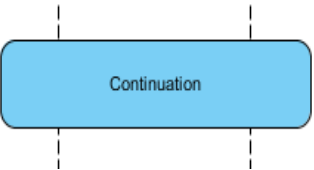
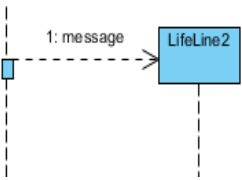
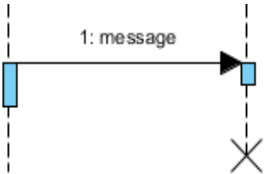
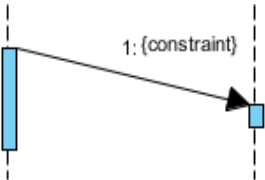

- The first thing to note about sequence diagram is how clearly the sequence diagram indicates the differences in how the participants interact.
- This is the great strength of interaction diagrams.
- They aren't good at showing details of algorithms, such as loops and conditional behavior, but they make the calls between participants crystal clear and give a really good picture about which participants are doing which processing.

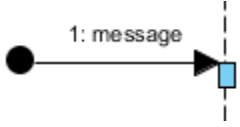
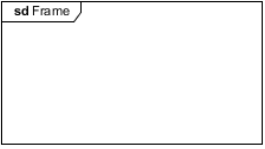


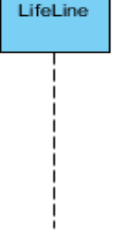
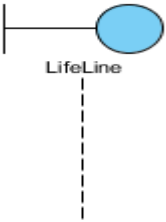
Use:


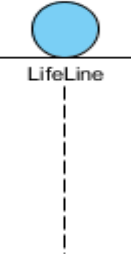


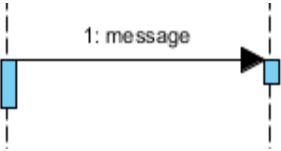
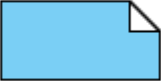
- You should use sequence diagrams when you want to look at the behavior of several objects within a single use case.
- Sequence diagrams are good at showing collaborations among the objects; Sequence diagrams are not so good at precise definition of the behavior.

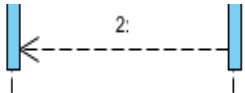
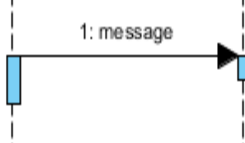



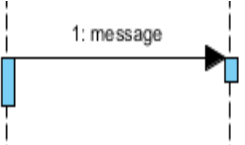
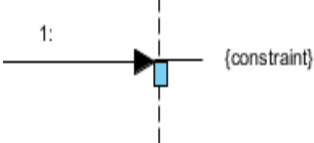
Symbol:

Symbol	Description
	Actor: An Actor models a type of role played by an entity that interacts with the subject (e.g., by exchanging signals and data), but which is external to the subject (i.e., in the sense that an instance of an actor is not a part of the instance of its corresponding subject). 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.
	Alternative Combined Fragment: A combined fragment defines an expression of interaction fragments. A combined fragment is defined by an interaction operator and corresponding interaction operands. Through the use of Combined Fragments the user will be able to describe a number of traces in a compact and concise manner.
	Call Message: A message defines a particular communication between Lifelines of an Interaction. Call message is a kind of message that represents an invocation of operation of target lifeline.

	<p>Concurrent: A concurrent represents a session of concurrent method invocation along an activation. It is placed on top of an activation.</p>
	<p>Constraint: A condition or restriction expressed in natural language text or in a machine readable language for the purpose of declaring some of the semantics of an element.</p>
	<p>Continuation: A Continuation is a syntactic way to define continuations of different branches of an Alternative Combined Fragment. Continuation is intuitively similar to labels representing intermediate points in a flow of control.</p>
	<p>Create Message: A message defines a particular communication between Lifelines of an Interaction. Create message is a kind of message that represents the instantiation of (target) lifeline.</p>
	<p>Destroy Message: A message defines a particular communication between Lifelines of an Interaction. Destroy message is a kind of message that represents the request of destroying the lifecycle of target lifeline.</p>
	<p>Duration Constraint: A Duration Constraint defines a Constraint that refers to a Duration Interval. A duration used to determine whether the constraint is satisfied.</p>
	<p>Duration Message: A message defines a particular communication between Lifelines of an Interaction.</p> <p>Duration message shows the distance between two time instants for a message invocation.</p>

	<p>Found Message: A found message is a message where the receiving event occurrence is known, but there is no (known) sending event occurrence. We interpret this to be because the origin of the message is outside the scope of the description. This may for example be noise or other activity that we do not want to describe in detail.</p>
	<p>Frame: A frame represents an interaction, which is a unit of behaviour that focuses on the observable exchange of information between Connectable Elements.</p>
	<p>Gate: A Gate is a connection point for relating a Message outside an Interaction Fragment with a Message inside the Interaction Fragment.</p>
	<p>Interaction Use: An Interaction Use refers to an Interaction. The Interaction Use is a shorthand for copying the contents of the referred Interaction where the Interaction Use is. To be accurate the copying must take into account substituting parameters with arguments and connect the formal gates with the actual ones.</p>
	<p>LifeLine: A lifeline represents an individual participant in the Interaction.</p>
	<p>LifeLine <<Boundary>>: A lifeline represents an individual participant in the Interaction.</p>

	<p>LifeLine <<Control>>: A lifeline represents an individual participant in the Interaction.</p>
	<p>LifeLine <<Entity>>: A lifeline represents an individual participant in the Interaction.</p>
	<p>Loop Combined Fragment: A combined fragment defines an expression of interaction fragments. A combined fragment is defined by an interaction operator and corresponding interaction operands. Through the use of Combined Fragments the user will be able to describe a number of traces in a compact and concise manner.</p>
	<p>Lost Message: A lost message is a message where the sending event occurrence is known, but there is no receiving event occurrence. We interpret this to be because the message never reached its destination.</p>
	<p>Message: A message defines a particular communication between Lifelines of an Interaction.</p>
	<p>Note: A note (comment) gives the ability to attach various remarks to elements. A comment carries no semantic force, but may contain information that is useful to a modeller.</p>

	<p>Return Message: A message defines a particular communication between Lifelines of an Interaction. Return message is a kind of message that represents the pass of information back to the caller of a corresponded former message.</p>
	<p>Send Message: A message defines a particular communication between Lifelines of an Interaction. Send message is a kind of message that represents the start of execution.</p>
	<p>Recursive Mess A message defines a particular communication between Lifelines of an Interaction. Recursive message is a kind of message that represents the invocation of message of the same lifeline. Its target points to an activation on top of the activation where the message was invoked from.</p>
	<p>Re-entrant Message: A message defines a particular communication between Lifelines of an Interaction. A re-entrant message points to an activation on top of another activation.</p>
	<p>Self-Message: A message defines a particular communication between Lifelines of an Interaction. Self-message is a kind of message that represents the invocation of message of the same lifeline</p>
	<p>Terminate Message: A message defines a particular communication between Lifelines of an Interaction. Terminate message is a kind of message that represents the termination of execution.</p>
	<p>Time Constraint: A Time Constraint defines a Constraint that refers to a Time Interval.</p>

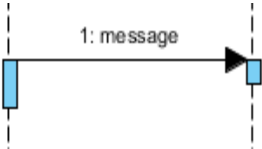
 <p>The diagram shows two vertical dashed lines representing lifelines. On the left lifeline, there is a light blue rectangular activation bar. A horizontal arrow points from this bar to the right lifeline, where it ends in a small light blue square. The text "1: message" is written above the arrow.</p>	<p>Uninterrupted Message: A message defines a particular communication between Lifelines of an Interaction. Uninterrupted message is a kind of message that represents an uninterrupted call.</p>
---	--

Table 7 (Sequence Diagram Symbol)

Sequence Diagram:

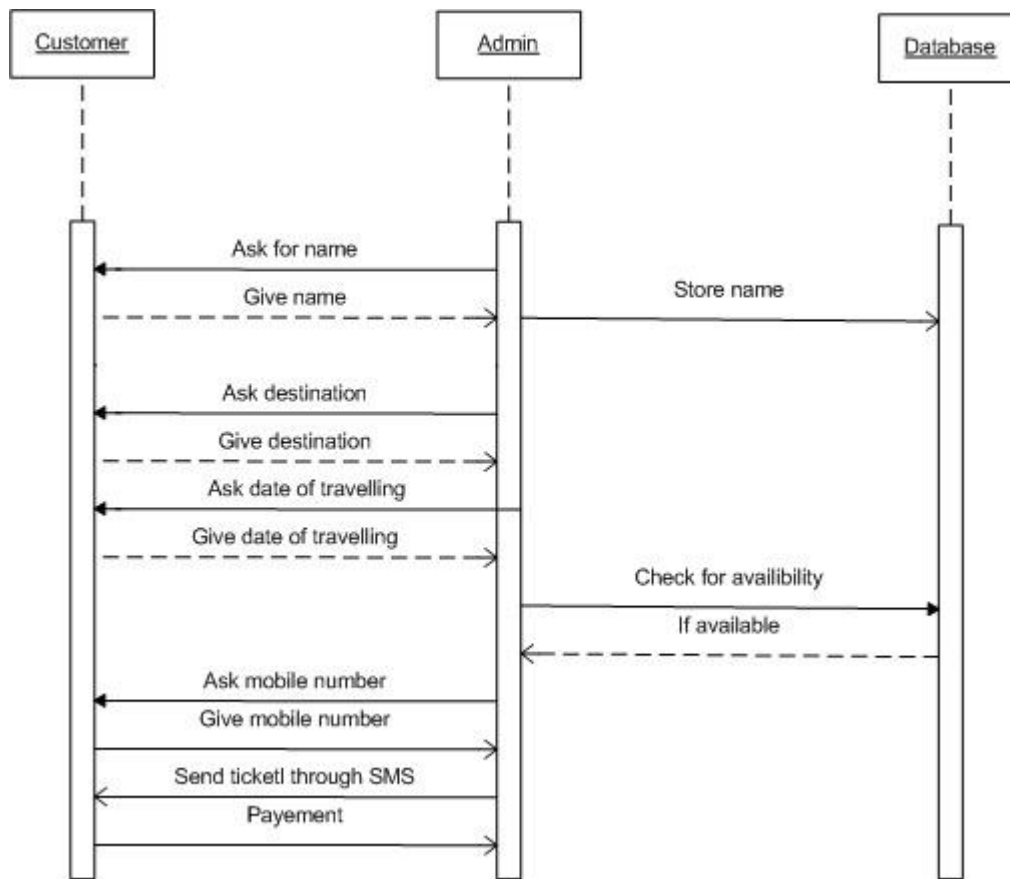


Figure 9 (Sequence diagram)

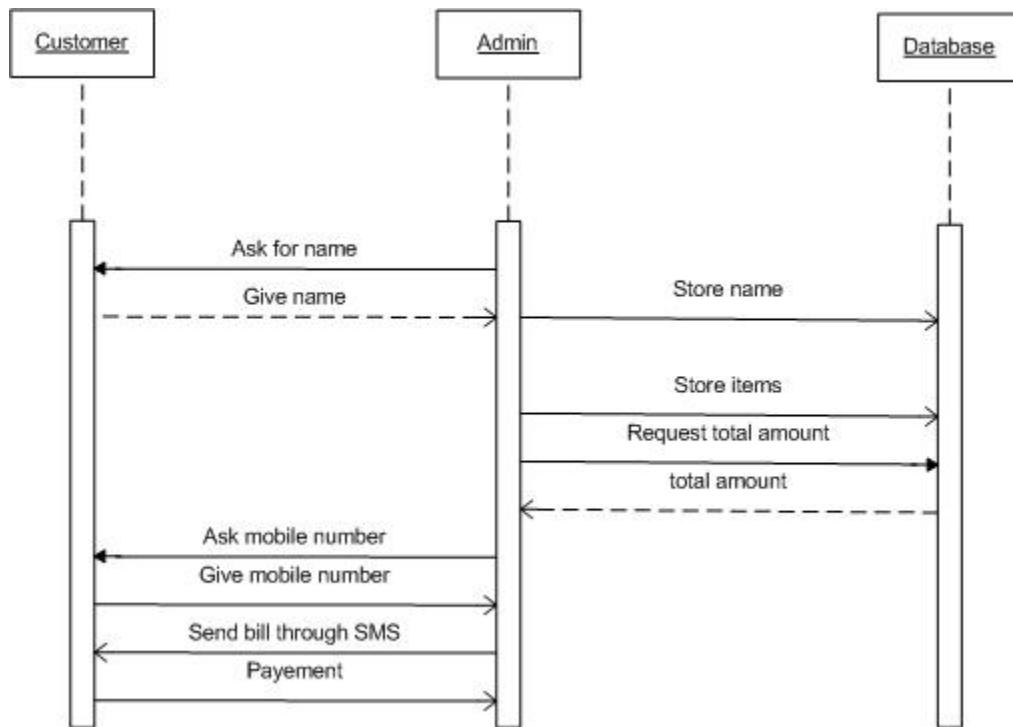


Figure 10 (Sequence diagram)

7.3 – Activity diagram


- Activity diagrams are the object-oriented equivalent of flow charts and data-flow diagrams from structured development
- Activity diagrams describe the workflow behavior of a system
- The process flows in the system are captured in the activity diagram
- Activity diagram illustrates the dynamic nature of a system by modeling the flow of control from activity to activity.
- Activity diagrams are a technique to describe procedural logic, business process, and work flow.
- In many ways, they play a role similar to flowcharts, but the principal difference between them and flowchart notation is that they support parallel behavior.

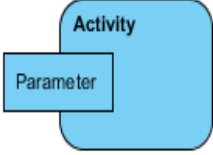




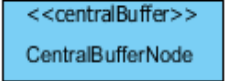
Use:

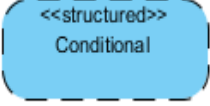
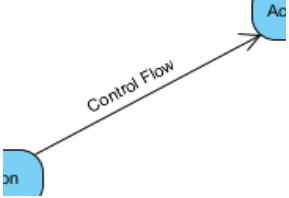
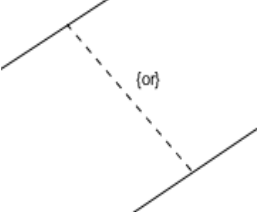


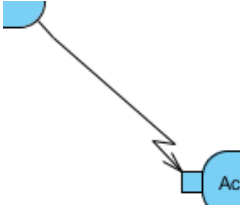

- Analyzing use case
- Dealing with multithreaded application
- Understanding workflow across many use cases.
- To explore the logic of Use Case








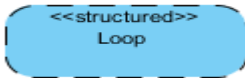
Useful for showing workflow and parallel processing



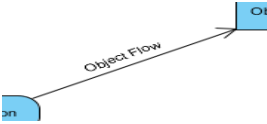
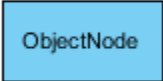

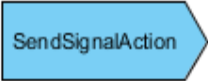
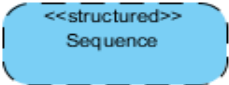

Symbol:

Symbol	Description
	<p>Activity:</p> <p>Activities may describe procedural computation. In this context, they are the methods corresponding to operations on classes. Activities may be applied to organizational modelling for business process engineering and workflow. In this context, events often originate from inside the system, such as the finishing of a task, but also from outside the system, such as a customer call. Activities can also be used for information system modelling to specify system level processes.</p>

	<p>Activity Parameter Node: Activity parameter nodes are object nodes at the beginning and end of flows that provide a means to accept inputs to an activity and provide outputs from the activity, through the activity parameters. Activity parameters inherit support for streaming and exceptions from Parameter.</p>
	<p>Action: An action represents a single step within an activity, that is, one that is not further decomposed within the activity. An activity represents a behaviour that is composed of individual elements that are actions. An action may have sets of incoming and outgoing activity edges that specify control flow and data flow from and to other nodes. An action will not begin execution until all of its input conditions are satisfied. The completion of the execution of an action may enable the execution of a set of successor nodes and actions that take their inputs from the outputs of the action.</p>
	<p>Accept Event Action: Accept Event Action is an action that waits for the occurrence of an event meeting specified condition.</p>
	<p>Accept Time Event Action: If the occurrence is a time event occurrence, the result value contains the time at which the occurrence transpired. Such an action is informally called a wait time action.</p>
	<p>Activity Final Node: An activity may have more than one activity final node. The first one reached stops all flows in the activity.</p>
	<p>Central Buffer Node: A central buffer node accepts tokens from upstream object nodes and passes them along to downstream object nodes. They act as a buffer for multiple in flows and out flows from other object nodes. They do not connect directly to actions.</p>

	<p>Conditional Node Specification: A conditional node is a structured activity node that represents an exclusive choice among some number of alternatives.</p>
	<p>Control Flow: A control flow is an edge that starts an activity node after the previous one is finished.</p>
	<p>Constraint: A condition or restriction expressed in natural language text or in a machine readable language for the purpose of declaring some of the semantics of an element.</p>
	<p>Data Store Node:: Determines where the data store node appears within different Namespaces within the overall model, and its accessibility.</p>
	<p>Decision Node: A decision node accepts tokens on an incoming edge and presents them to multiple outgoing edges. Which of the edges is actually traversed depends on the evaluation of the guards on the outgoing edges..</p>
	<p>Exception Handler: An exception handler is an element that specifies a body to execute in case the specified exception occurs during the execution of the protected node.</p>
	<p>Expansion Node: An expansion node is an object node used to indicate a flow across the boundary of an expansion region. A flow into a region contains a collection that is broken into its individual elements inside the region, which is executed once per element. A flow out of a region combines individual elements into a collection for use outside the region.</p>

	<p>Expansion Region: An expansion region is a strictly nested region of an activity with explicit input and outputs (modelled as Expansion Nodes). Each input is a collection of values. If there are multiple inputs, each of them must hold the same kind of collection, although the types of the elements in the different collections may vary. The expansion region is executed once for each element (or position) in the input collection.</p>
	<p>Flow Final Node: A flow final destroys all tokens that arrive at it. It has no effect on other flows in the activity.</p>
	<p>Fork Node: A fork node is a control node that splits a flow into multiple concurrent flows. A fork node has one incoming edge and multiple outgoing edges.</p>
	<p>Initial Node: An initial node is a control node at which flow starts when the activity is invoked. An activity may have more than one initial node.</p>
	<p>Input Pin: Input pins are object nodes that receive values from other actions through object flows. See Pin, Action, and ObjectNode for more details.</p>
	<p>Join Node: An interruptible activity region is an activity group that supports termination of tokens flowing in the portions of an activity. An interruptible region contains activity nodes. When a token leaves an interruptible region via edges designated by the region as interrupting edges, all tokens and behaviour in the region are terminated.</p>
	<p>Join Node: A join node is a control node that synchronizes multiple flows. A join node has multiple incoming edges and one outgoing edge.</p>
	<p>Loop Node: A loop node is a structured activity node that represents a loop with setup, test, and body sections.</p>

	<p>Merge Node: A merge node is a control node that brings together multiple alternate flows. It is not used to synchronize concurrent flows but to accept one among several alternate flows. A merge node has multiple incoming edges and a single outgoing edge.</p>
	<p>Note: A note (comment) gives the ability to attach various remarks to elements. A comment carries no semantic force, but may contain information that is useful to a modeller.</p>
	<p>Object Flow: An object flow is an activity edge that can have objects or data passing along it.</p>
	<p>Object Node: An object node is an activity node that indicates an instance of a particular classifier, possibly in a particular state, may be available at a particular point in the activity. Object nodes can be used in a variety of ways, depending on where objects are flowing from and to, as described in the semantics sub clause.</p>
	<p>Output Pin: Output pins are object nodes that deliver values to other actions through object flows.</p>
	<p>Send Signal Action: Send Signal Action is an action that creates a signal instance from its inputs, and transmits it to the target object, where it may cause the firing of a state machine transition or the execution of an activity.</p>
	<p>Sequence Node: A sequence node is a structured activity node that executes its actions in order.</p>
	<p>Structured Activity Node: A sequence node is a structured activity node that executes its actions in order.</p>

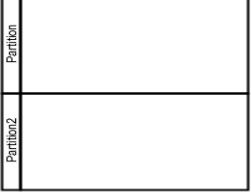

	<p>Swim lane: Swim lane is used for partitioning the children in an activity diagram.</p>
	<p>Value Pin: A value pin is an input pin that provides a value to an action that does not come from an incoming object flow edge.</p>

Table 8 (Activity Diagram)

Activity Diagram:

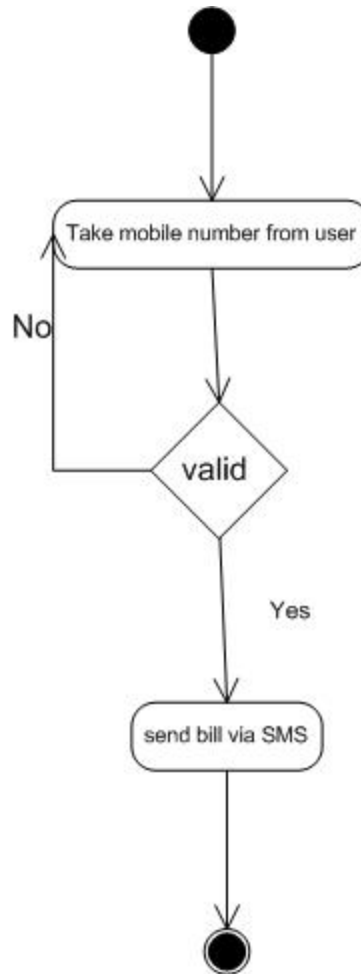


Figure 11 (Activity diagram 1)

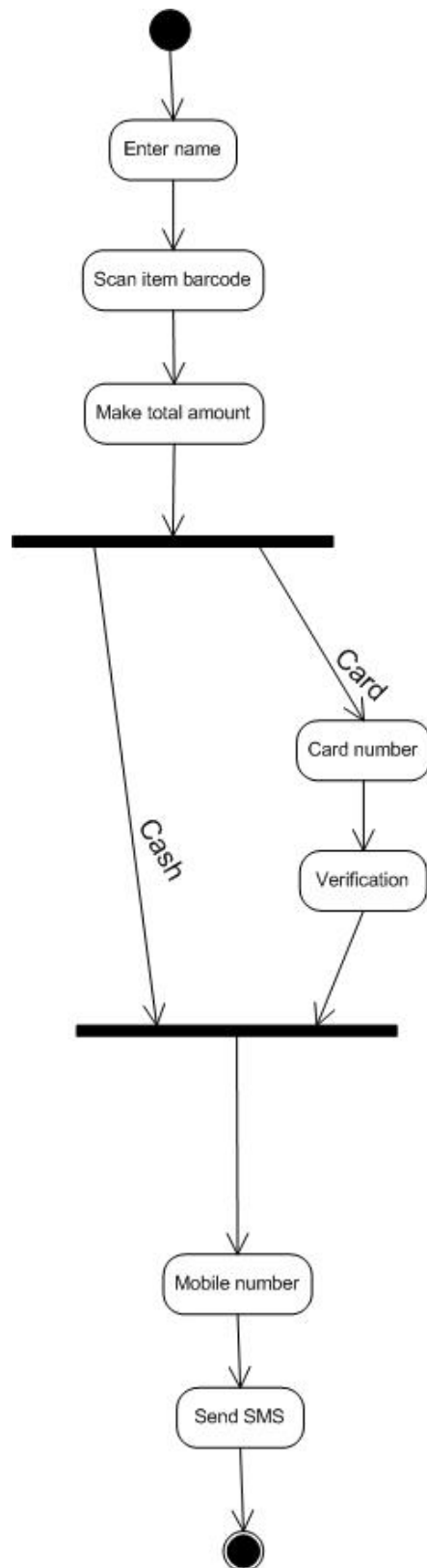


Figure 12 (Activity diagram 2)

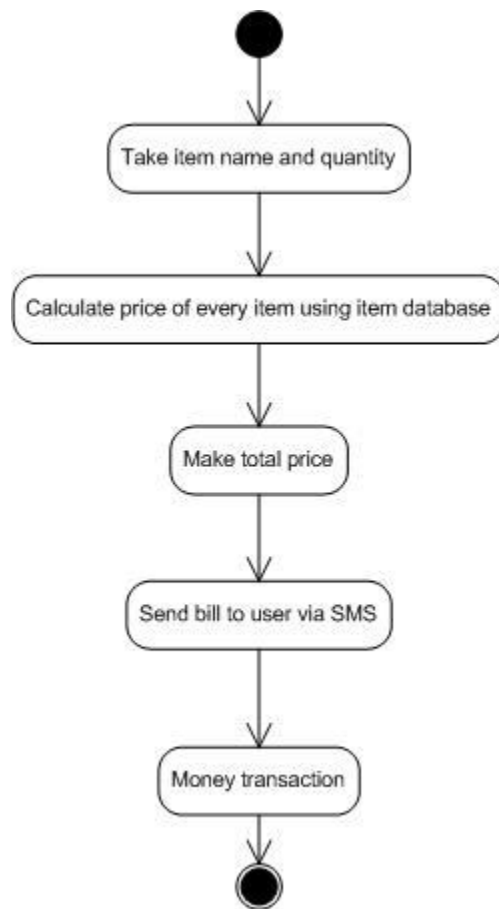


Figure 13 (Activity diagram 3)

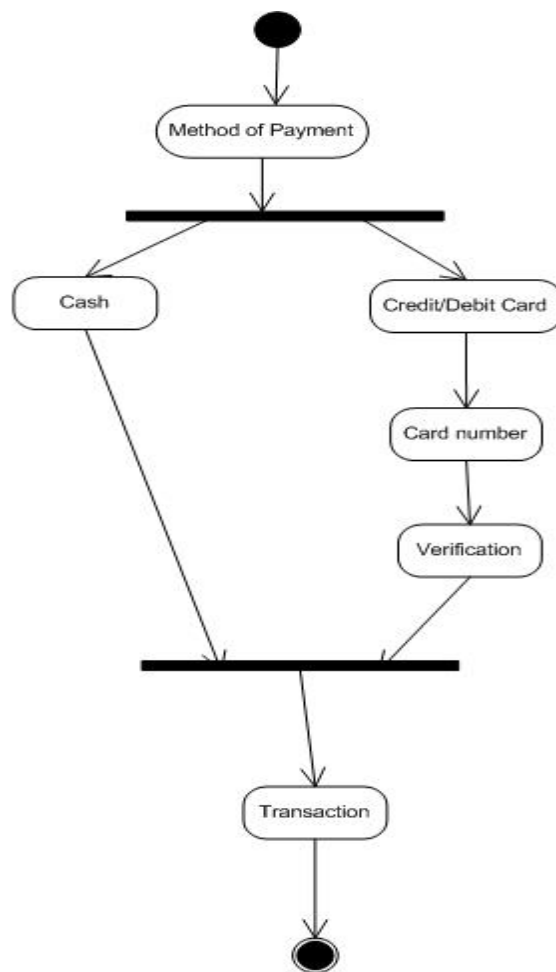


Figure 14 (Activity diagram 4)

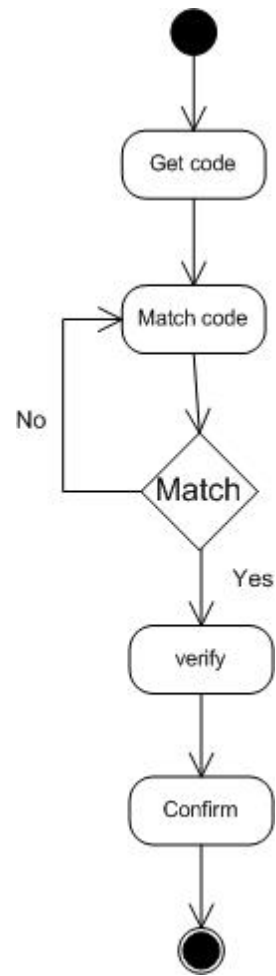


Figure 15 (Activity diagram 5)

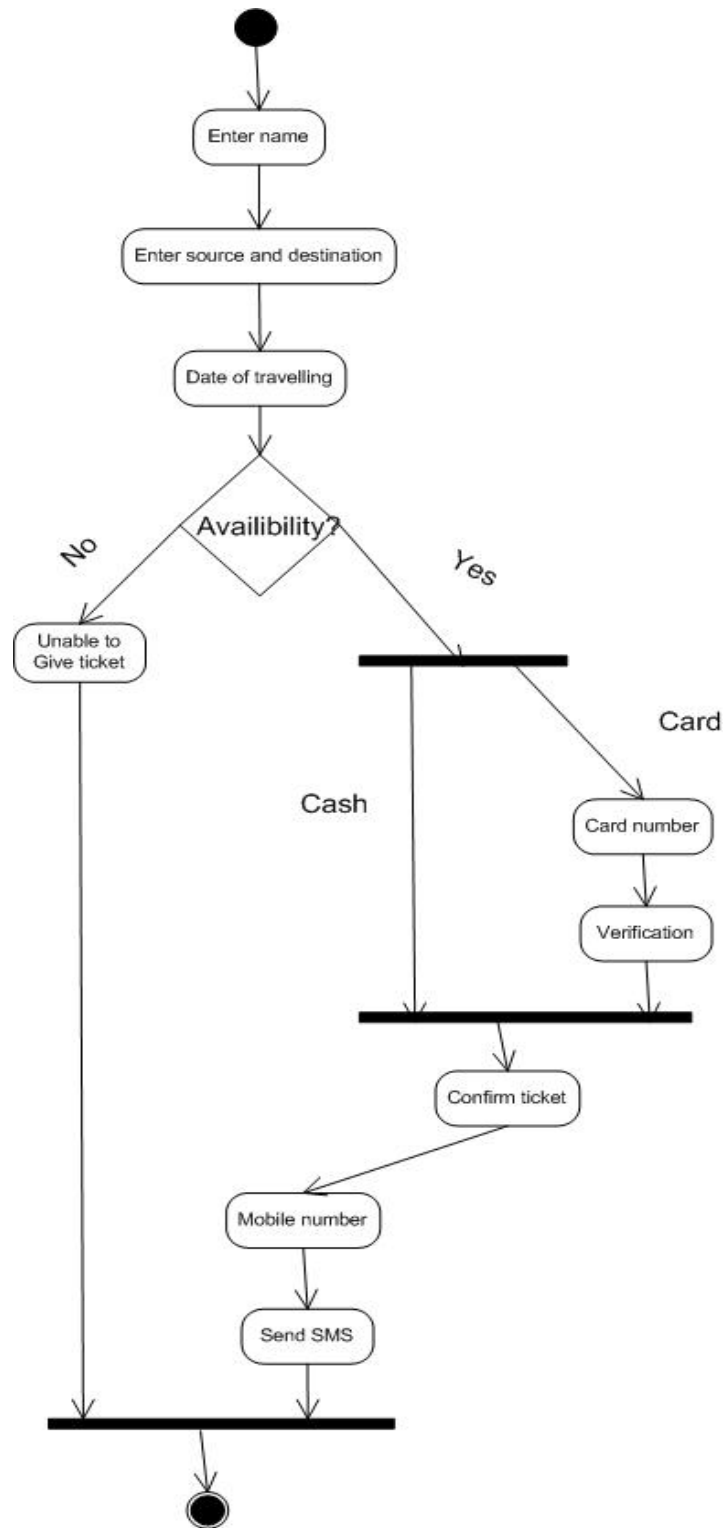


Figure 16 (Activity diagram 6)

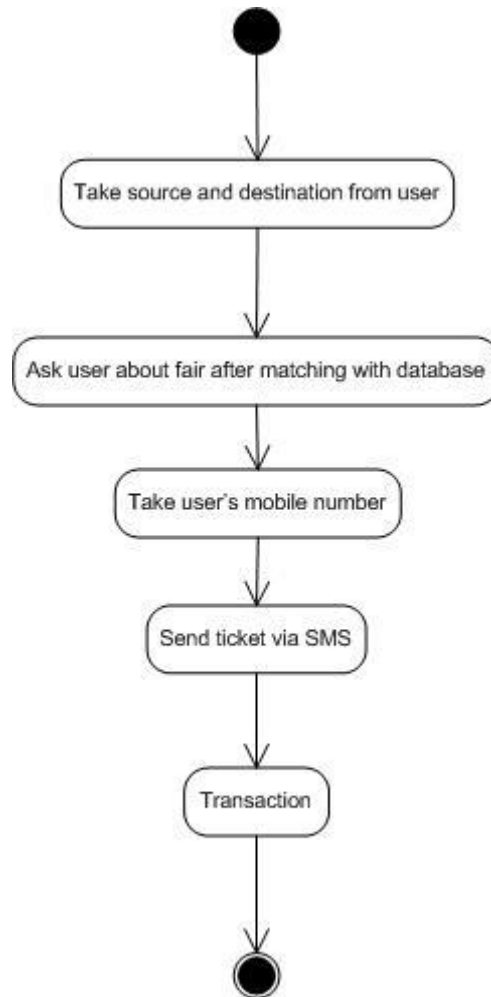


Figure 17 (Activity diagram 7)

7.4 – Entity-Relationship diagram



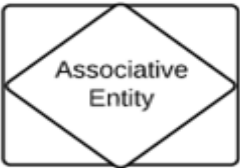
An entity relationship model is the result of using a systematic process to describe and define a subject, area of business data. It doesn't define business process only visualize business data.

- The data is represented as a components that are linked with each other by relationship that express the dependencies and requirements between them.

Use:

- The entity relationship model adopts the more natural view that the real world consists of entities and relationships.
- It incorporates some of the important semantic information about the information about the real world.

Symbol:

Symbol	Description
	Entity: Entities are objects or concepts that represent important data. They are typically nouns, e.g. <i>Customer</i> , <i>supervisor</i> , <i>location</i> , or <i>promotion</i> . Strong entities exist independently from other entity types. They always possess one or more attributes that uniquely distinguish each occurrence of the entity.
	Weak entity: Weak entities depend on some other entity type. They don't possess unique attributes (also known as a primary key) and have no meaning in the diagram without depending on another entity. This other entity is known as the owner.
	Associative entity: Associative entities are entities that associate the instances of one or more entity types. They also contain attributes that are unique to the relationship between those entity instances.






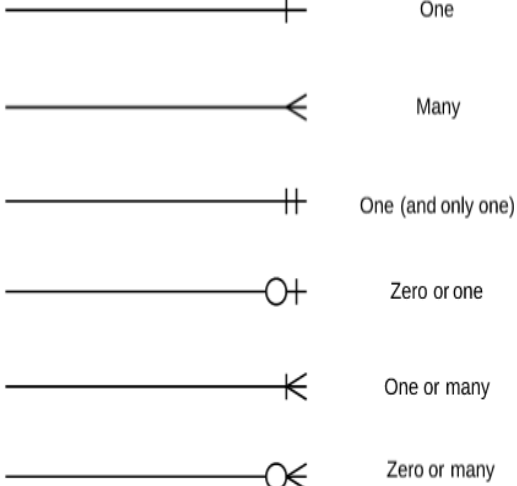
	<p>Relationships: Relationships are meaningful associations between or among entities. They are usually verbs, e.g. <i>Assign</i>, <i>associate</i>, or <i>track</i>. A relationship provides useful information that could not be discerned with just the entity types.</p>
	<p>Weak relationship: Weak relationships, or identifying relationships, are connections that exist between a weak entity type and its owner.</p>
	<p>Attributes: Attributes are characteristics of either an entity, a many-to-many relationship, or a one-to-one relationship.</p>
	<p>Multivalued attribute: Multivalued attributes are those that are capable of taking on more than one value.</p>
	<p>Derived attributes : Derived attributes are attributes whose value can be calculated from related attribute values.</p>
	<p>Cardinality: Cardinality and ordinality, respectively, refer to the maximum number of times an instance in one entity can be associated with instances in the related entity, and the minimum number of times an instance in one entity can be associated with an instance in the related entity. Cardinality and ordinality are represented by the styling of a line and its endpoint, as denoted by the chosen notation style.</p>

Table 9 (Entity Relationship)

Entity Relationship Diagram:

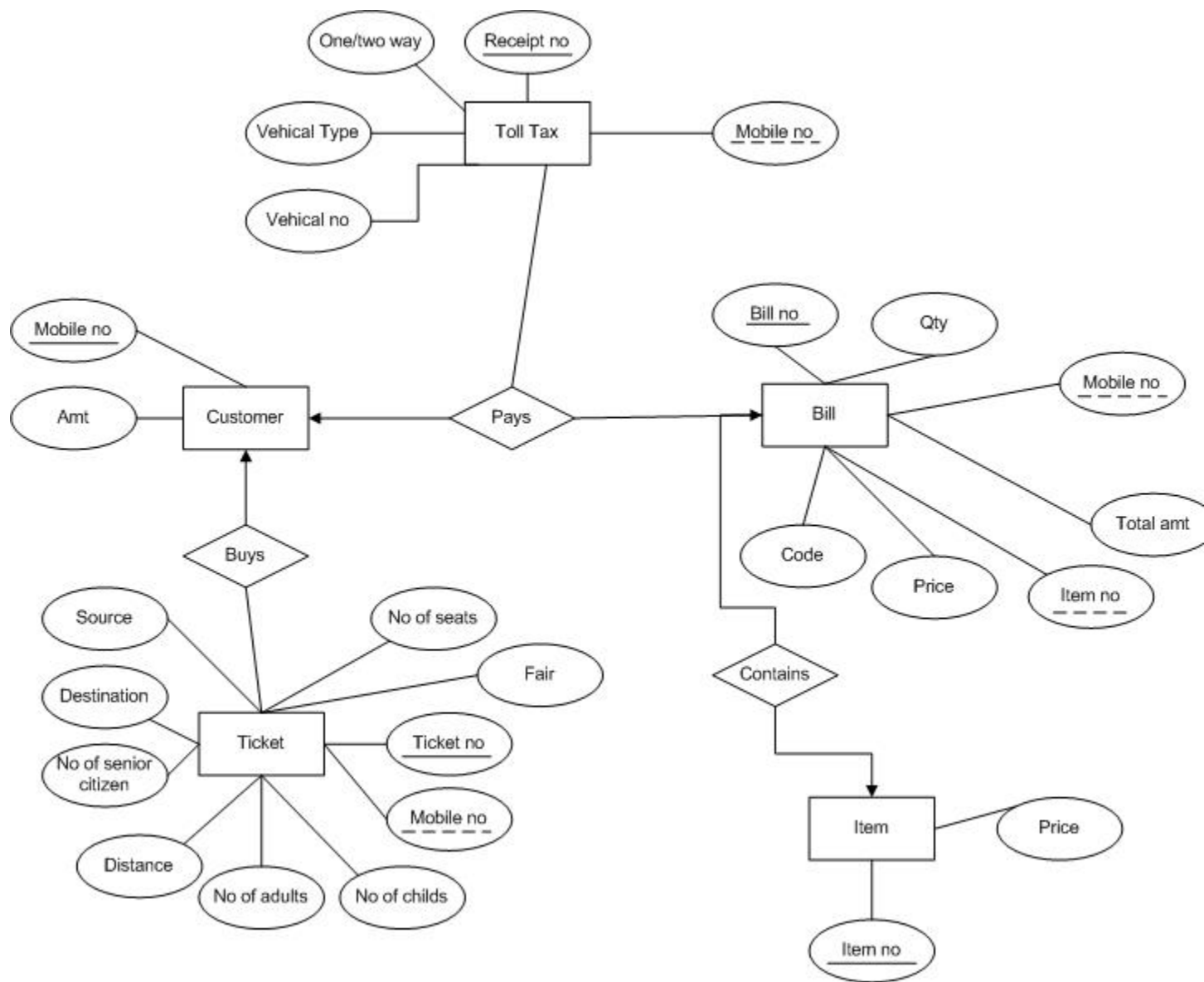


Figure 18 (E-R diagram)

7.5 – Data flow diagram

A data flow diagram (DFD) is a graphical representation of the “flow” of data through an Information system, modelling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used For the visualization of data processing.

A DFD show what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored.

Symbol:

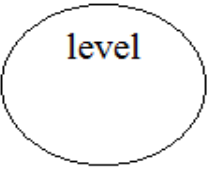
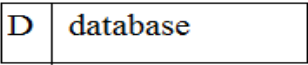

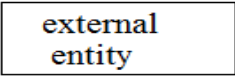
Symbol	Description
	Circle: A process transforms incoming Data flow into outgoing data flow.
	Datastore/database: Datastore/database are repositories of data in the system. They are sometimes also referred to us files.
	Dataflow: Dataflows are pipelines through which packets of information flow. Label the arrows with the name of the data that moves through it.
	External Entity: External entities are objects outside the system, with which the system communicates .External entities are sources and destinations of the system’s inputs and outputs.

Table 10 (Data Flow)

7.5.1 Context level 0 diagram

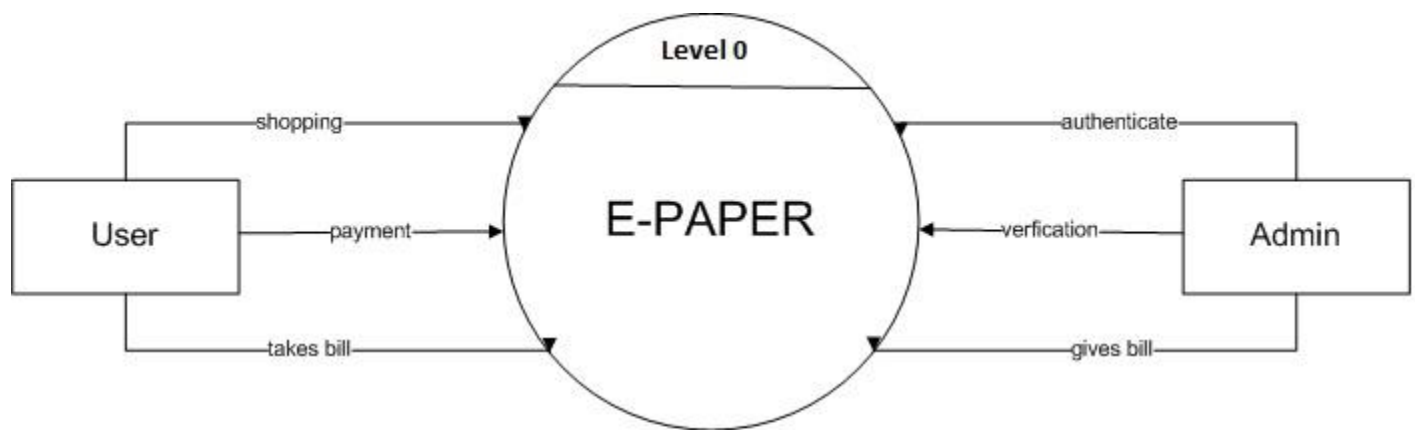


Figure 19 (Context level 0 diagram)

7.5.1 Context level 1 diagram

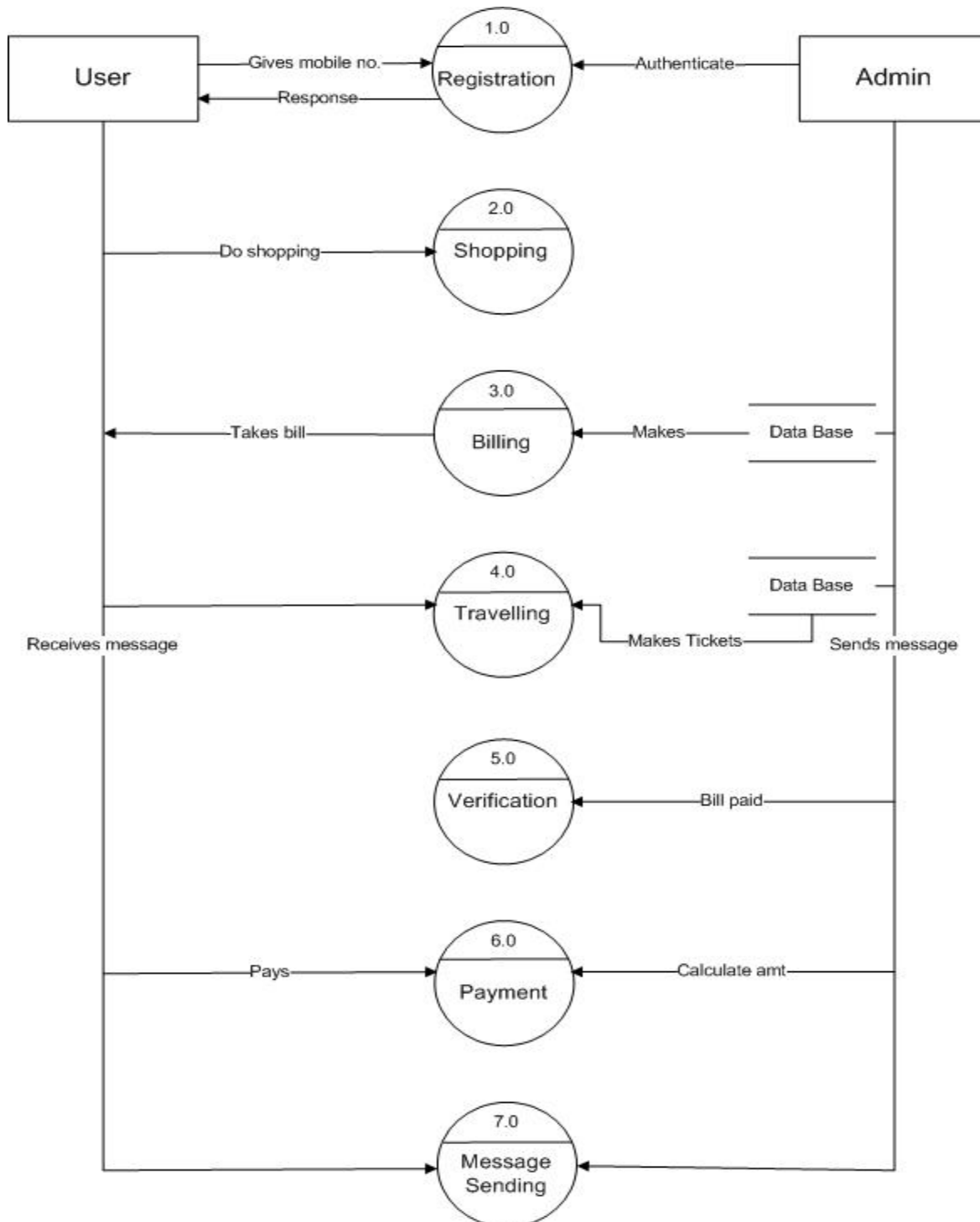


Figure 20 (Context level 0 diagram)