1. UML OVERVIEW

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.

1.1 KEY POINTS:

- UML stands for **U**nified **M**odeling **L**anguage.
- UML is different from the other common programming languages like C++, Java, and COBOL etc.
- UML is a pictorial language used to make software blue prints.

So UML can be described as a general purpose visual modeling language to visualize, specify, construct and document software system. Although UML is generally used to model software systems but it is not limited within this boundary. It is also used to model non software systems as well like process flow in a manufacturing unit etc.

UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object oriented analysis and design.

1.2 Use of UML

UML offers a way to visualize a system's architectural blueprints in a diagram including elements such as:

- any activities (jobs);
- individual components of the system;
- and how they can interact with other software components;
- how the system will run;
- how entities interact with others (components and interfaces);
- external user interface.

class

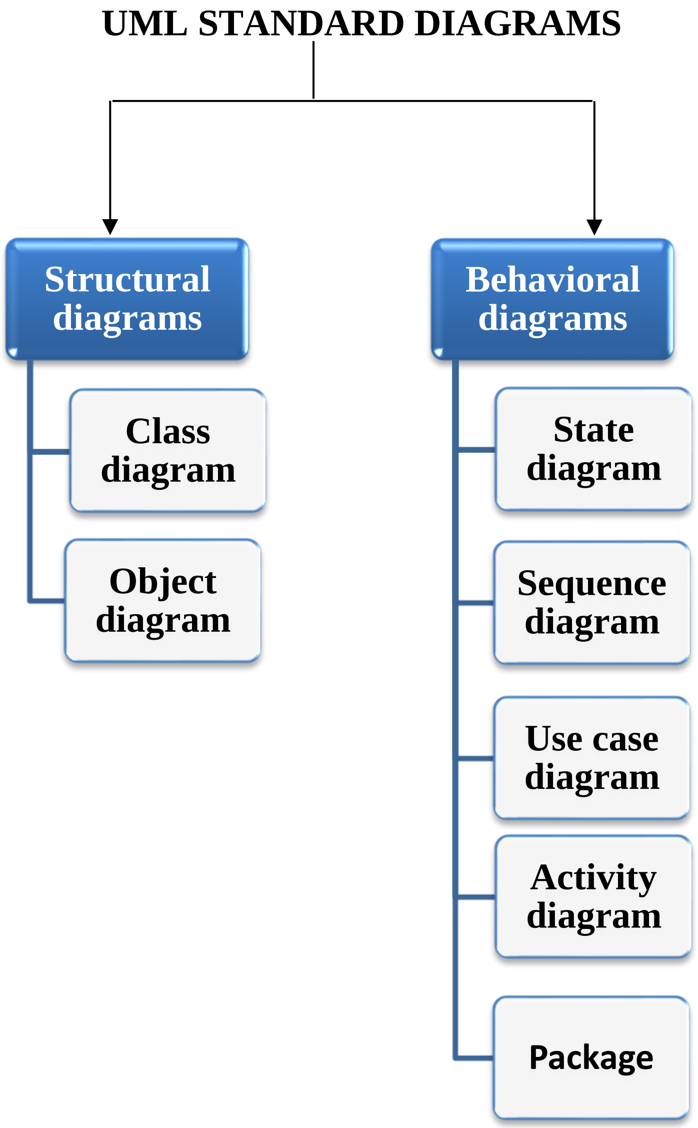


Figure 1. UML Diagrams.

1.3 CLASS MODEL:

Class diagrams provide a graphic notation for modeling classes and their relationships, there by describing possible objects. Class diagrams are useful for both abstract modeling and designing actual programs. They are concise, easy to understand and work well in practice.

class

1.3.1 UML notation for class: The UML symbol for a class is a box. The class name should be in bold face, center the name in the box, and capitalize the first letter.

ClassName
Attribute1: type1=initialvalue1 Attribute2:type2=initalvalue2 ….
Operation 1(argument list1):return type1 Operation2(argument list2):returntype2 ….

1.3.2. CLASS: class describes a group of objects with the same properties (attributes), behavior (operations), kind of relationships and semantics. Classes often appear as common nouns and noun phrases in the problem description and discussion with users.

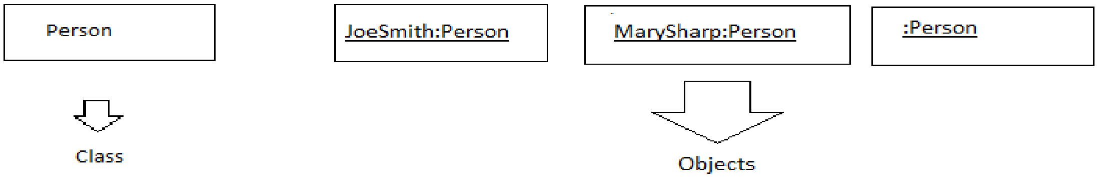


Figure 2. Object and Class Diagram.

1.3.3. OBJECT: An object is a concept, or thing with identity that has meaning for an application. Object often appear as proper nouns or specific references in problem descriptions and discussions with users. An object is a instance-or occurrence of a class.

1.3.4. VALUES and ATTRIBUTES: A value is a piece of data. An attribute is a named property of class that describes a value held by each object of the class. The UML notation lists the attributes in the second compartment of the class box. Optional details, such as type and default value, may follow each attribute. A colon precedes the type. The equal sign precedes the default value.

class

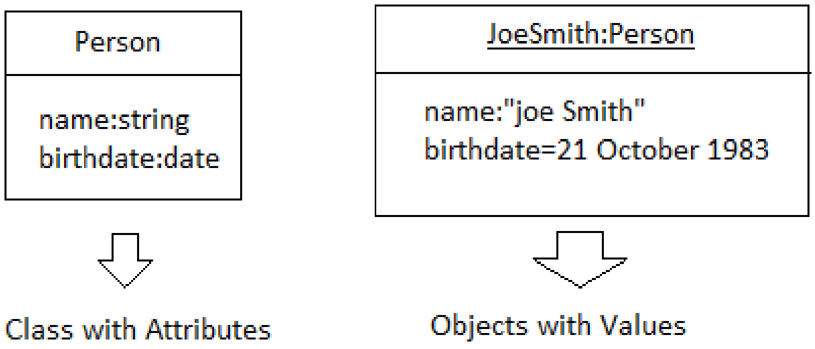


Figure 3. Values and Attributes.

1.3.5. OPERATION and METHODS: An operation is a function or procedure that may be applied to or by objects in the class. All objects in a class share the same operations. An operation may have arguments in addition to its target object.

A method is the implementation of an operation for a class. When an operation has methods on several classes, it is important that the methods all have the same signature-the number and types of arguments and the type of result value. The UML notation is to list the operations in the third compartment of the class box.

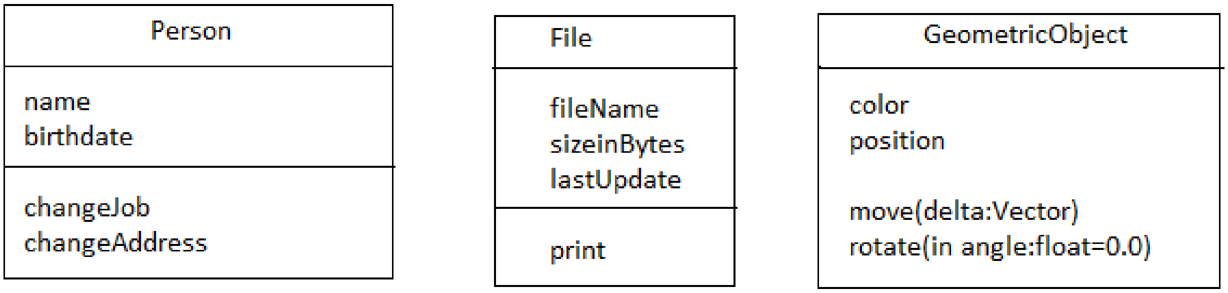


Figure 4. Operations and Methods.

1.3.6. LINKS and ASSOCIATION: A link is a physical or conceptual connection among objects. Most inks relate two objects, but some links relate three or more objects. A link is an instance of an association.

An association is a description of a group of links with common structure and common semantics. An association describes a set of potential links in the same way that a class describes set of potential objects.

class

Notation:

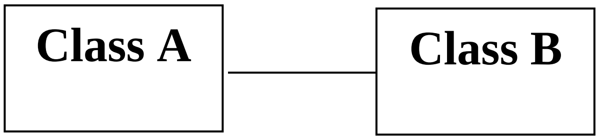


Figure 5. Association.

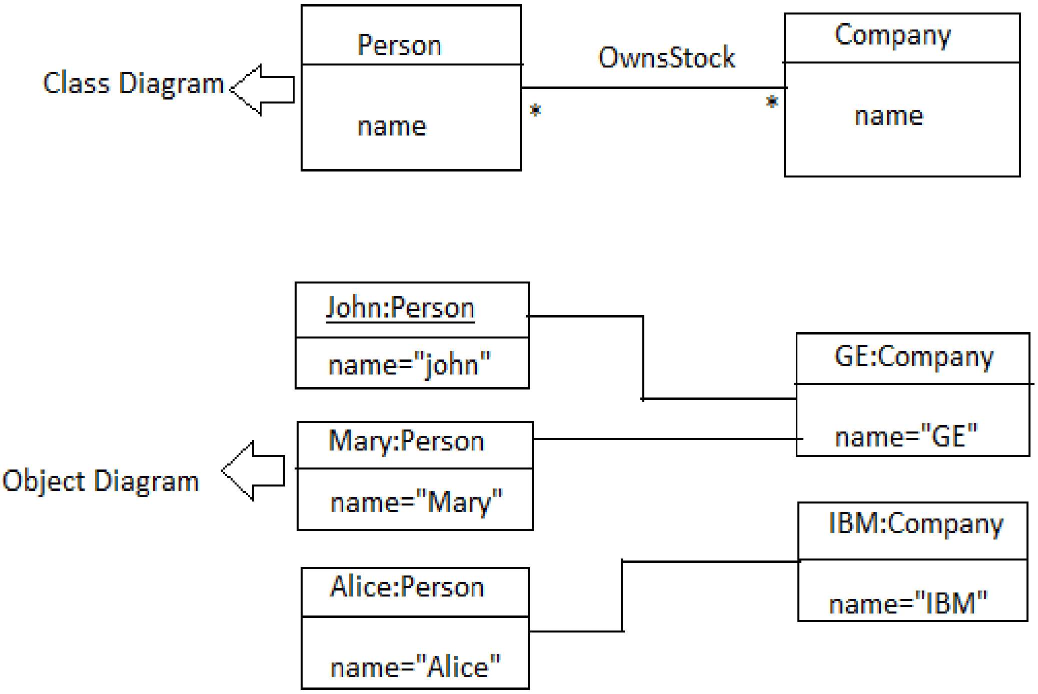


Figure 6. Association and Links.

1.3.7 MULTIPLICITY: Multiplicity specifies the number of instances of one class that may relate to a single instance of an associated class. Multiplicity constrains the number of related objects. UML diagrams explicitly list multiplicity at the ends of association lines the UML specifies multiplicity with an interval such as "1","1...*".The symbol "*" is the short hand notation that denotes "many".

1: no more than one

0..1: zero or one

* : many

0..*: zero or many

1..* :one or many

class

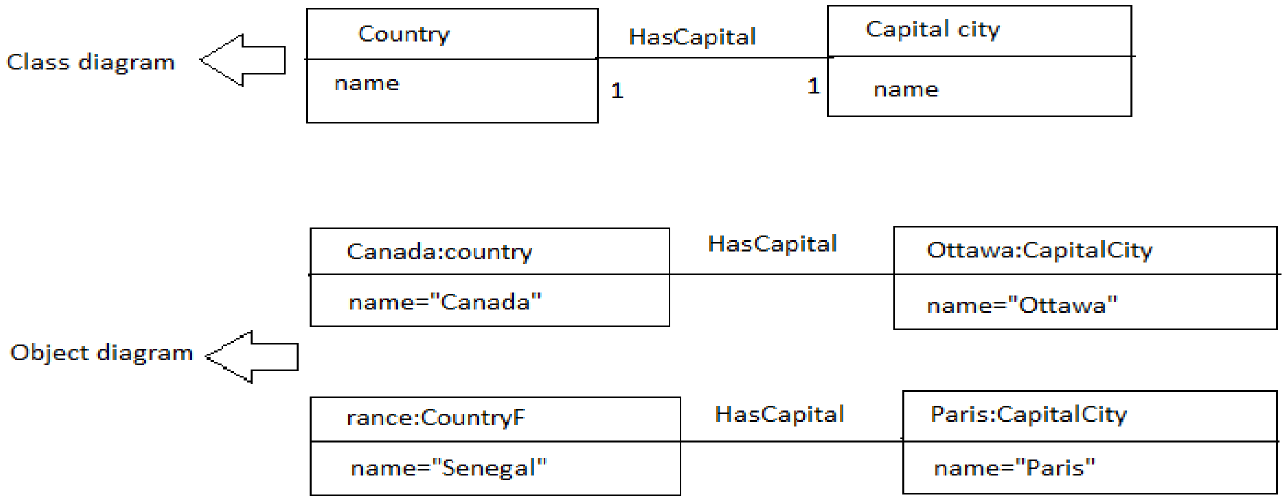


Figure 7. Multiplicity.

1.3.8. ASSOCIATION END NAMES: Association ends can not only be assigned multiplicity they can also be named. In a problem description they are generally identified by nouns.

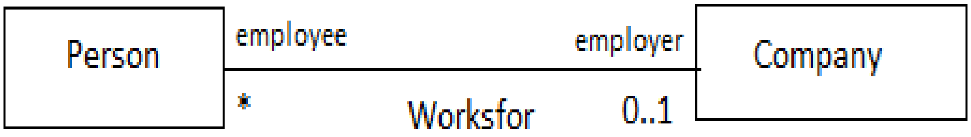
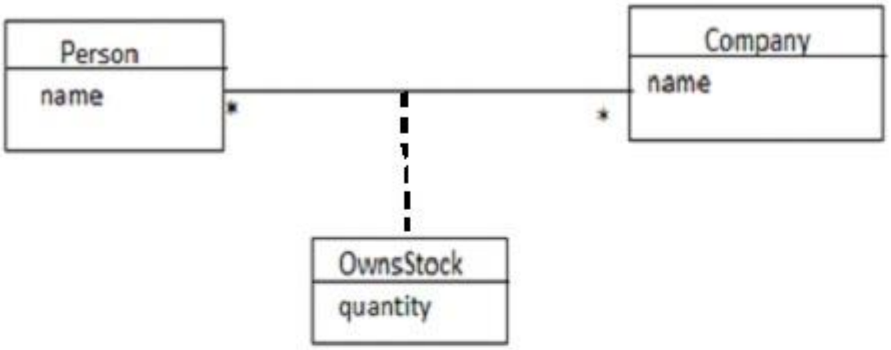


Figure 8. Association End Names.

1.3.9. ASSOCIATION CLASS: An association class is an association that is also a class. Like the links of an association the instances of an association class derive identity from instances of the constituent classes. Association lets us to specify identity and navigation paths precisely.

Figure 9.



Association Classes.

class

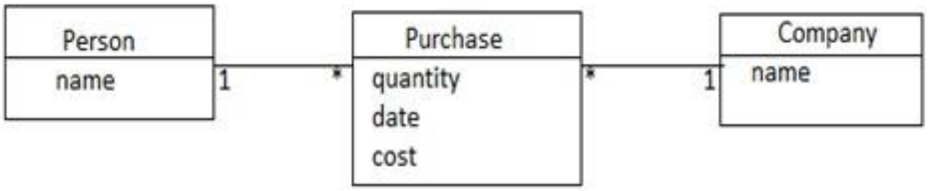


Figure 10. Ordinary Class.

1.3.10. QUALIFIED ASSOCIATION: A qualified association is a association in which an attribute called the qualifier disambiguates the objects for a "many" association end. It is possible to define qualifiers from one-to-many and many-to-many associations. Qualification increases the precision of the model.

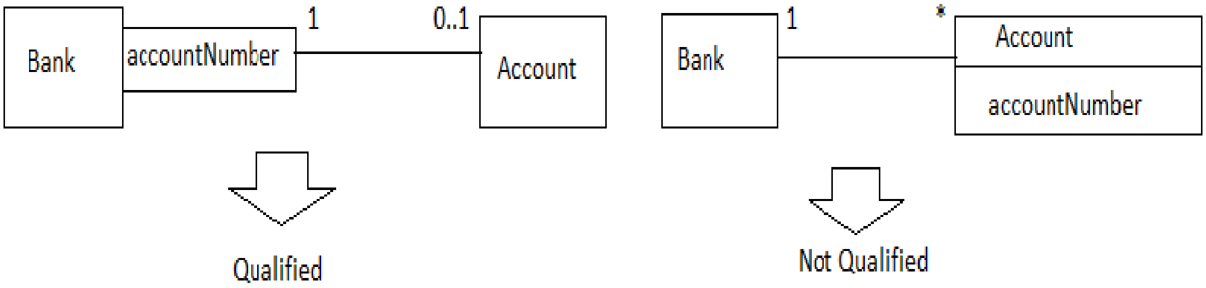


Figure 11. Qualified Association.

1.3.11. ENUMERATION: An enumeration is a data type that has finite set of values. A data is description of values. Data types include numbers, strings and enumeration. An enumeration is list of values. We declare enumeration by listing the keyword 'enumeration' in guillemetes (<< >>) above the enumeration name in the top section of box.

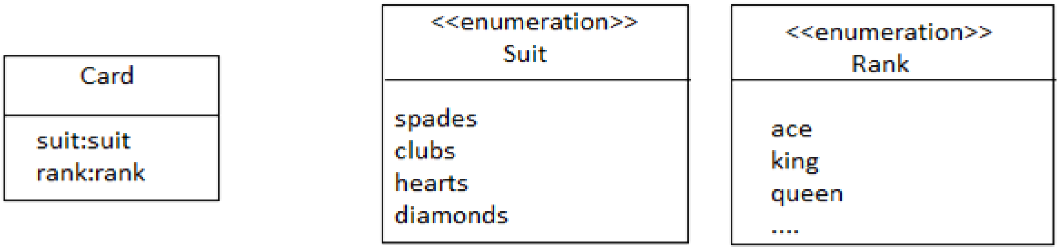


Figure 12. Enumeration.

class

1.3.12. COMPOSITION and AGGREGATION: Composition is a special type of aggregation that denotes a strong ownership between Class A, the whole and Class B ,its parts. Illustrates composition with the filled diamond. Use a hollow diamond to represent simple aggregation relationship, in which the “whole” class plays a more important role than the “part” class, but the two classes are not dependent on each other. The diamond end in both the composition and aggregation relationship points towards the “whole” class or the aggregate.

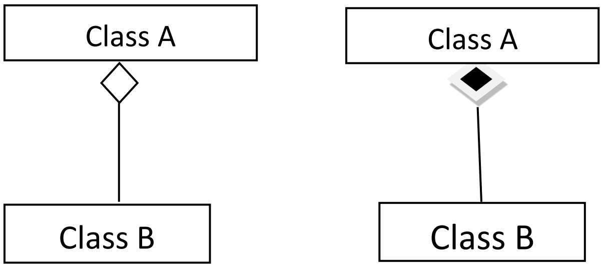


Figure 13(a) Aggregation and 13(b) Composition.

1.3.13. COMMENTS: Comments in a class diagram gives us the extra information and thus increases the conceptual view of the class model.UML notation for comments is:

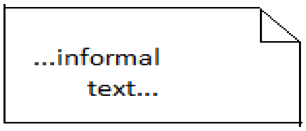


Figure 14. Comment Diagram.

1.3.14. GENERALIZATION: Generalization is a relationship between a class (super class) and one or more variations of the class(subclasses). Generalization organizes the classes by their similarities and differences, structuring the description of objects. Simple generalization organizes classes into a hierarchy; each subclass has a single immediate super class . There can be multiple levels of generalization.

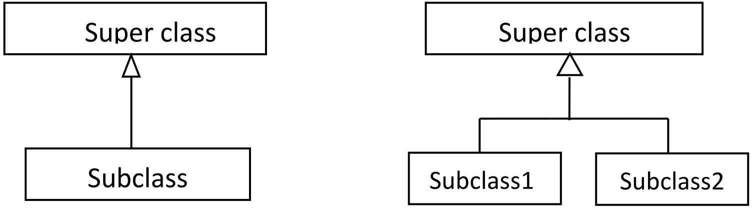


Figure 15. Generalization.

class

1.4. STATE DIAGRAM

The name of the diagram itself clarifies the purpose of the diagram and other details. It describes different states of a component in a system.

1.4.1. Purpose:

Following are the main purposes of using State diagrams:

- To model dynamic aspect of a system.
- To model life time of a reactive system.
- To describe different states of an object during its life time.
- Define a state machine to model states of an object.

1.4.2. How to draw State Diagram?

State diagram is used to describe the states of different objects in its life cycle. So the emphasis is given on the state changes upon some internal or external events. These states of objects are important to analyze and implement them accurately. State diagrams are very important for describing the states. States can be identified as the condition of objects when a particular event occurs. Before drawing a State diagram we must have clarified the following points:

- Identify important objects to be analyzed.
- Identify the states.
- Identify the events.

1.4.3. Where to use State Diagrams?

- To model object states of a system.
- To model reactive system. Reactive system consists of reactive objects.
- To identify events responsible for state changes.
- Forward and reverse engineering.

class

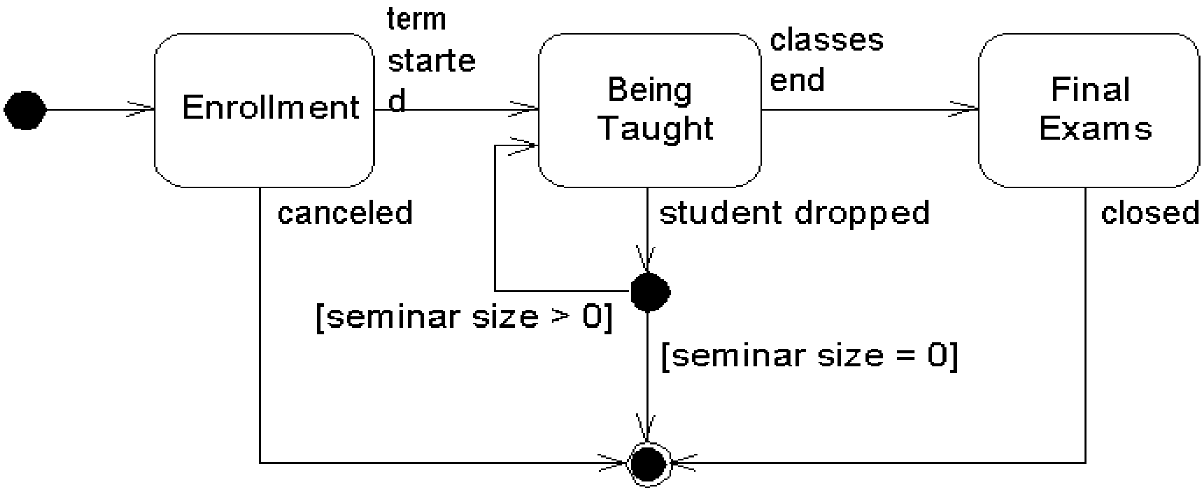


Figure 16. State Diagram.

1.5. USE CASE DIAGRAM

To model a system the most important aspect is to capture the dynamic behavior. Dynamic behavior means the behavior of the system when it is running operating. In UML there are five diagrams available to model dynamic nature and use case diagram is one of them. As use case diagram is dynamic in nature there should be some internal or external factors for interacting with system. These internal and external agents are known as actors

1.5.1. Purpose

The purposes of use case diagrams can be as follows:

- Used to gather requirements of a system.
- Used to get an outside view of a system.
- Identify external and internal factors influencing the system.
- Show the interacting among the requirements are actors.

1.5.2. Use Case Diagram

Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analyzed the functionalities are captured in use cases. So we can say that use cases are nothing but the system functionalities written in an organized manner. Now the second things which are relevant to the use cases are the actors. Actors can be defined as something that interacts with the system. The actors can be human user, some internal applications or may

class

be some external applications. So in a brief when we are planning to draw an use case diagram we should have the following items identified.

- Functionalities to be represented as an use case
- Actors
- Relationships among the use cases and actors.

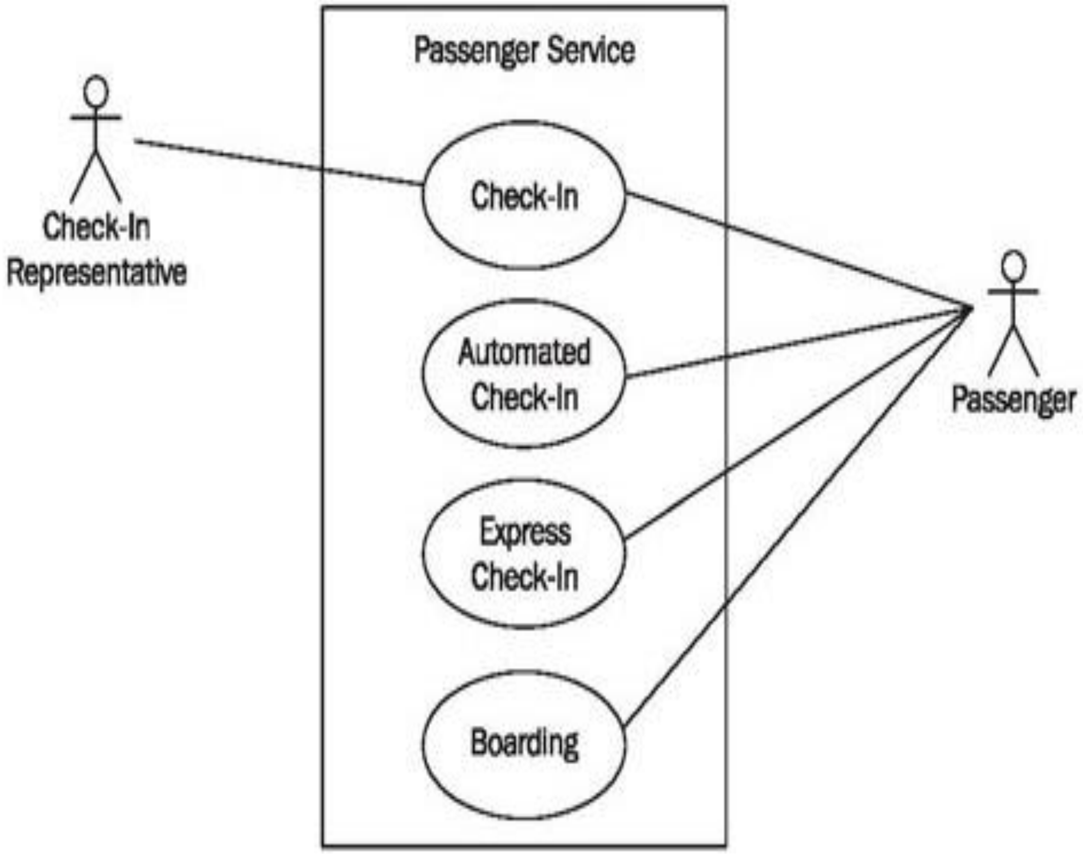


Figure 17. Use Case Diagram.

1.5.3 .Where to Use Case Diagrams?

- To model object states of a system.
- To model reactive system. Reactive system consists of reactive objects.
- To identify events responsible for state changes.
- Forward and reverse engineering.

class

1.5.4. Guidelines for use case model

- First determine the system boundary.
- Ensure that actors are focused.
- Each use case must provide value to users.
- Relate use case and actors.
- Remember that use cases are informal.
- Use cases can be structured.

1.6. INTERACTION MODEL: From the name Interaction it is clear that the diagram is used to describe some type of interactions among the different elements in the model. So this interaction is a part of dynamic behavior of the system. This interactive behavior is represented in UML by Sequence diagram. Sequence diagram emphasizes on time sequence of messages.

1.6.1. Purpose:

The purposes of interaction diagram are:

- To capture dynamic behavior of a system.
- To describe the message flow in the system.
- To describe structural organization of the objects.
- To describe interaction among objects.

1.6.2. How to draw Interaction Diagram?

As we have already discussed that the purpose of interaction diagrams are to capture the dynamic aspect of a system. So to capture the dynamic aspect we need to understand what a dynamic aspect is and how it is visualized. Dynamic aspect can be defined as the snap shot of the running system at a particular moment. The following things are to identified clearly before drawing the interaction diagram:

- Objects taking part in the interaction.
- Message flows among the objects.
- The sequence in which the messages are flowing.
- Object organization.

class

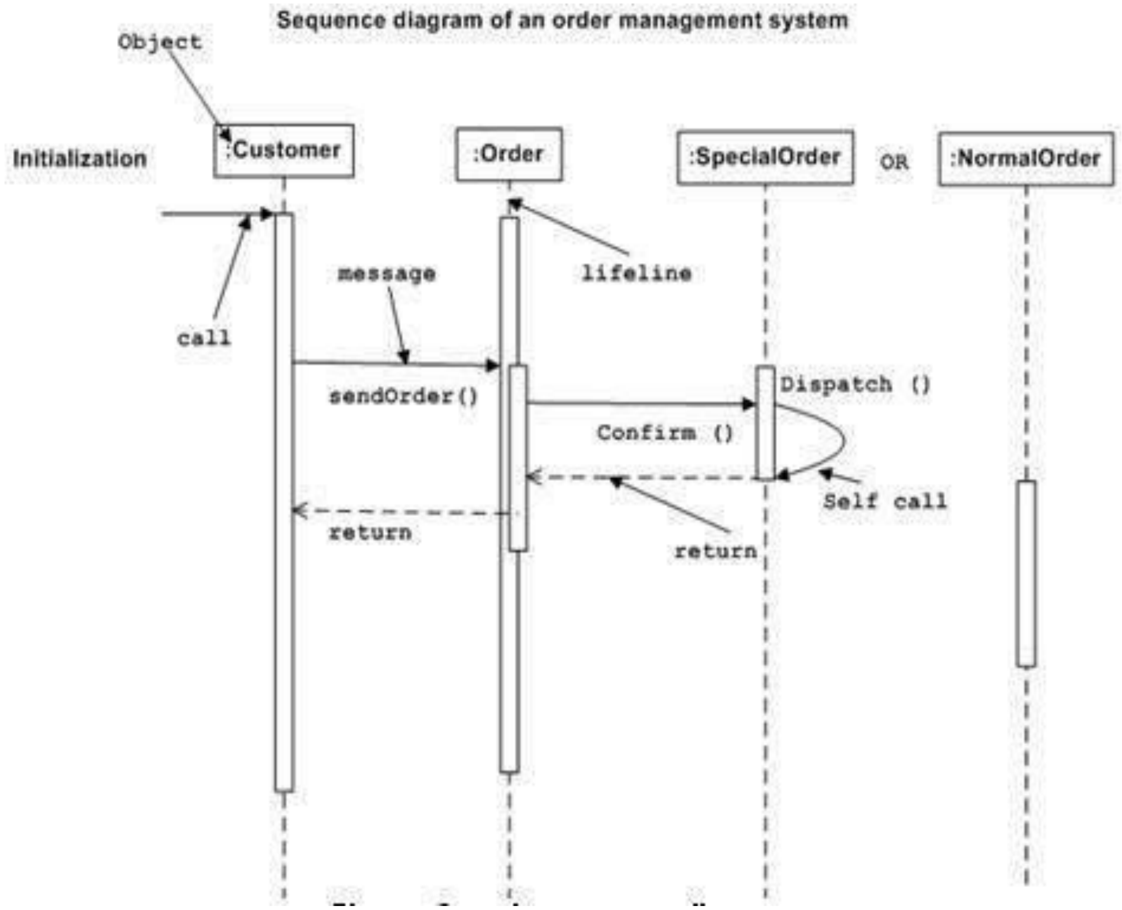


Figure 18. Sequence Diagram.

1.6.3. Use case relationships

1.6.3.1. Include relationship:

Include relationship incorporates one use case within the behavior sequence of another use case.

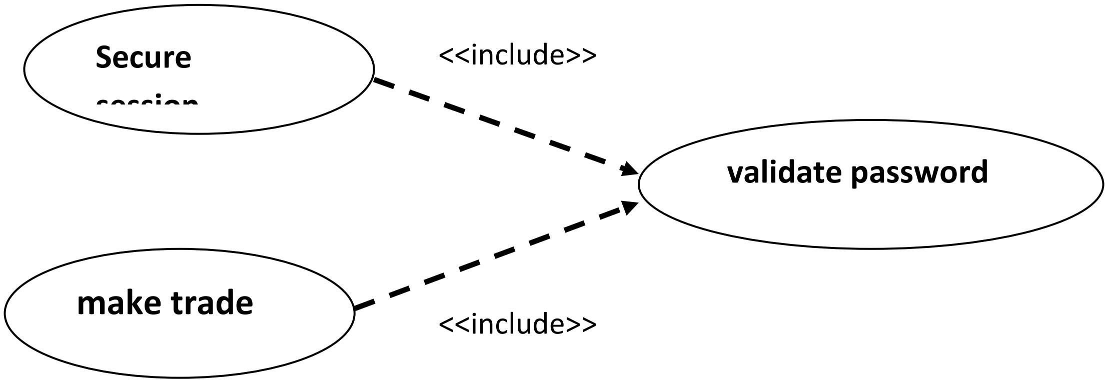


Figure 19. Use Case Inclusion.

class

1.6.3.2. Extend relationship:

The extend relationship adds incremental behavior to a use case. It is like an include relationship looked at from the opposite direction, in which the extension adds itself to the base, rather than the base explicitly incorporating the extension.

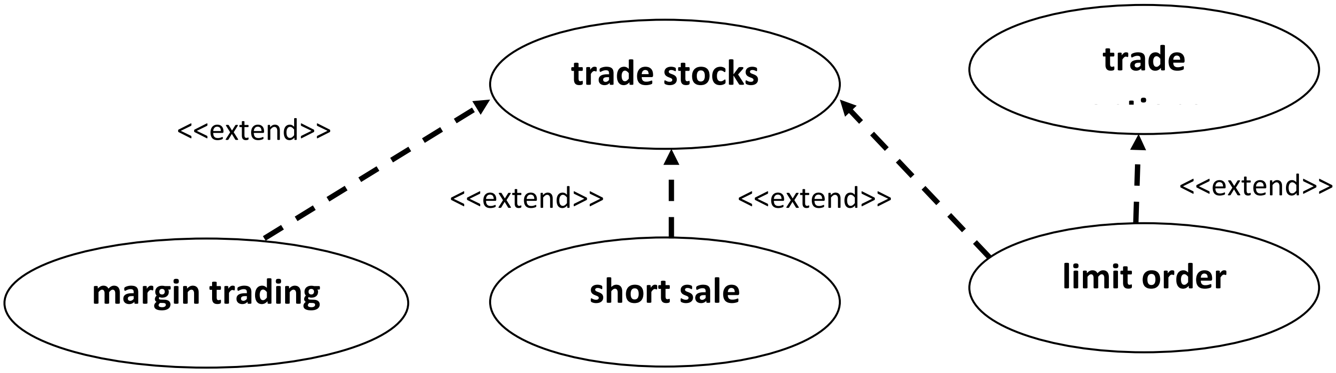


Figure 20. Use Case Extension.

1.6.4. Where to use Interaction Diagrams?

The following are the usages of interaction diagrams:

- To model flow of control by time sequence.
- To model flow of control by structural organizations.
- For forward engineering.
- For reverse engineering.
-

1.6.5. Guidelines for Sequence models

- Prepare at least one scenario per use case.
- Abstract the scenarios into sequence diagrams.
- Divide complex interactions.
- Prepare a sequence diagram for each error condition.

1.7 ACTIVITY DIAGRAM:

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn

class

from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deals with all type of flow control by using different elements like fork, join etc.

1.7.1. Purpose

The purposes can be described as:

- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system.

1.7.2. How to draw Activity Diagram?

Activity diagrams are mainly used as a flow chart consists of activities performed by the system. But activity diagram are not exactly a flow chart as they have some additional capabilities. These additional capabilities include branching, parallel flow, swimlane etc. Before drawing an activity diagram we must have a clear understanding about the elements used in activity diagram. The main element of an activity diagram is the activity itself. An activity is a function performed by the system. After identifying the activities we need to understand how they are associated with constraints and conditions. So before drawing an activity diagram we should identify the following elements:

- Activities
- Association
- Conditions
- Constraints

Once the above mentioned parameters are identified we need to make a mental layout of the entire flow. This mental layout is then transformed into an activity diagram. The following is an example of an activity diagram for order management system. In the diagram four activities are identified which are associated with conditions. One important point should be clearly understood that an activity diagram cannot be exactly matched with the code. The activity diagram is made to understand the flow of activities and mainly used by the business users.

The following diagram is drawn with the four main activities:

- Send order by the customer
- Receipt of the order
- Confirm order
- Dispatch order

class

After receiving the order request condition checks are performed to check if it is normal or special order. After the type of order is identified dispatch activity is performed and that is marked as the termination of the process.

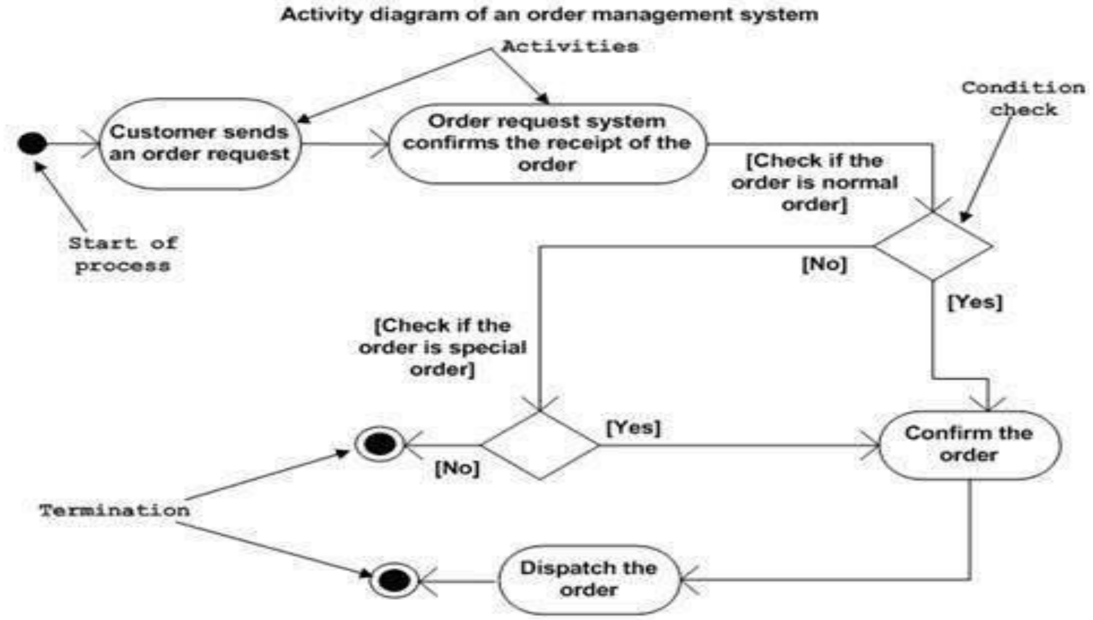

Activity diagram of an order management system
Activities
Condition check
Customer sends an order request
Order request system confirms the receipt of the order
[Check if the order is normal order]
Start of process
[No]
[Yes]
[Check if the order is special order]
[Yes]
[No]
Confirm the order
Termination
Dispatch the order

Figure 21. Activity Diagram.

1.7.3. Use of Activity Diagrams

The main usages of activity diagram:

- Modeling work flow by using activities.
- Modeling business requirements.
- High level understanding of the system's functionalities.
- Investigate business requirements at a later stage.

1.7.4. Guidelines for Activity models

- Don't misuse activity diagrams.
- Be careful with branches and conditions.
- Be careful with concurrent activities.
- Consider executable activity diagrams

1.8. PACKAGE DIAGRAM:

class

Package diagrams are used to reflect the organization of packages and their elements. When used to represent class elements, package diagrams provide a visualization of the namespaces. The most common use for package diagrams is to organize use case diagrams and class diagrams, although the use of package diagrams is not limited to these UML elements.

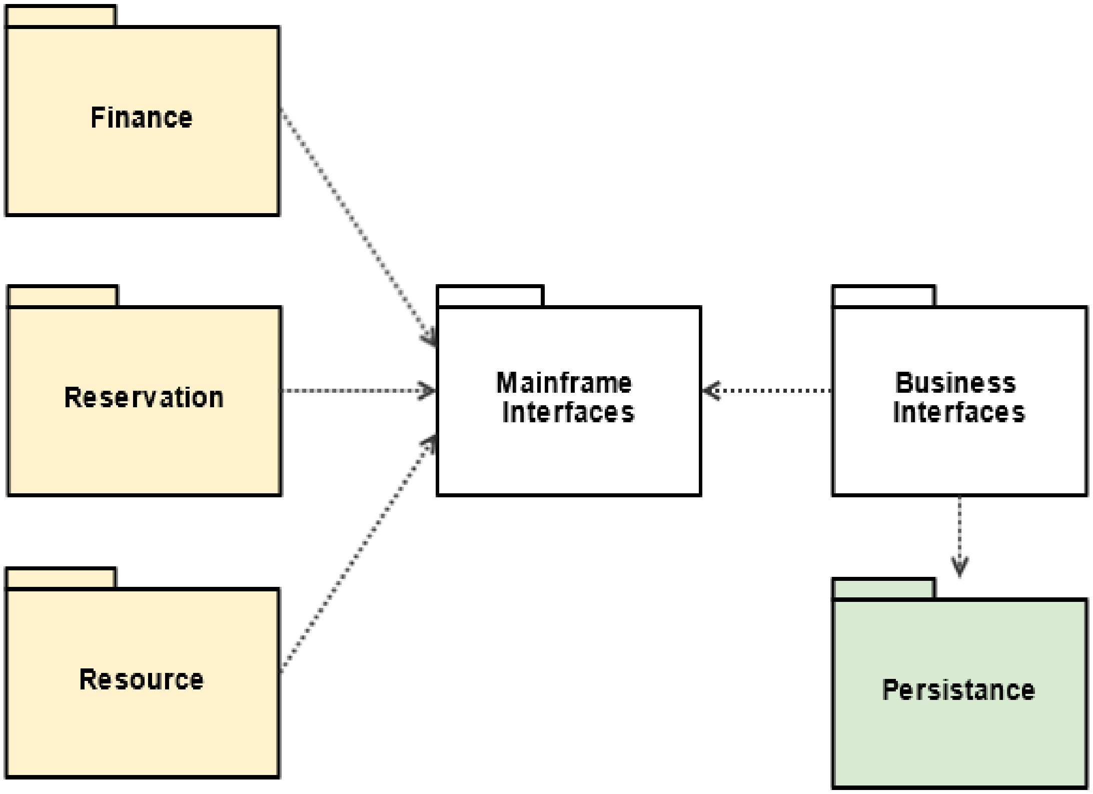
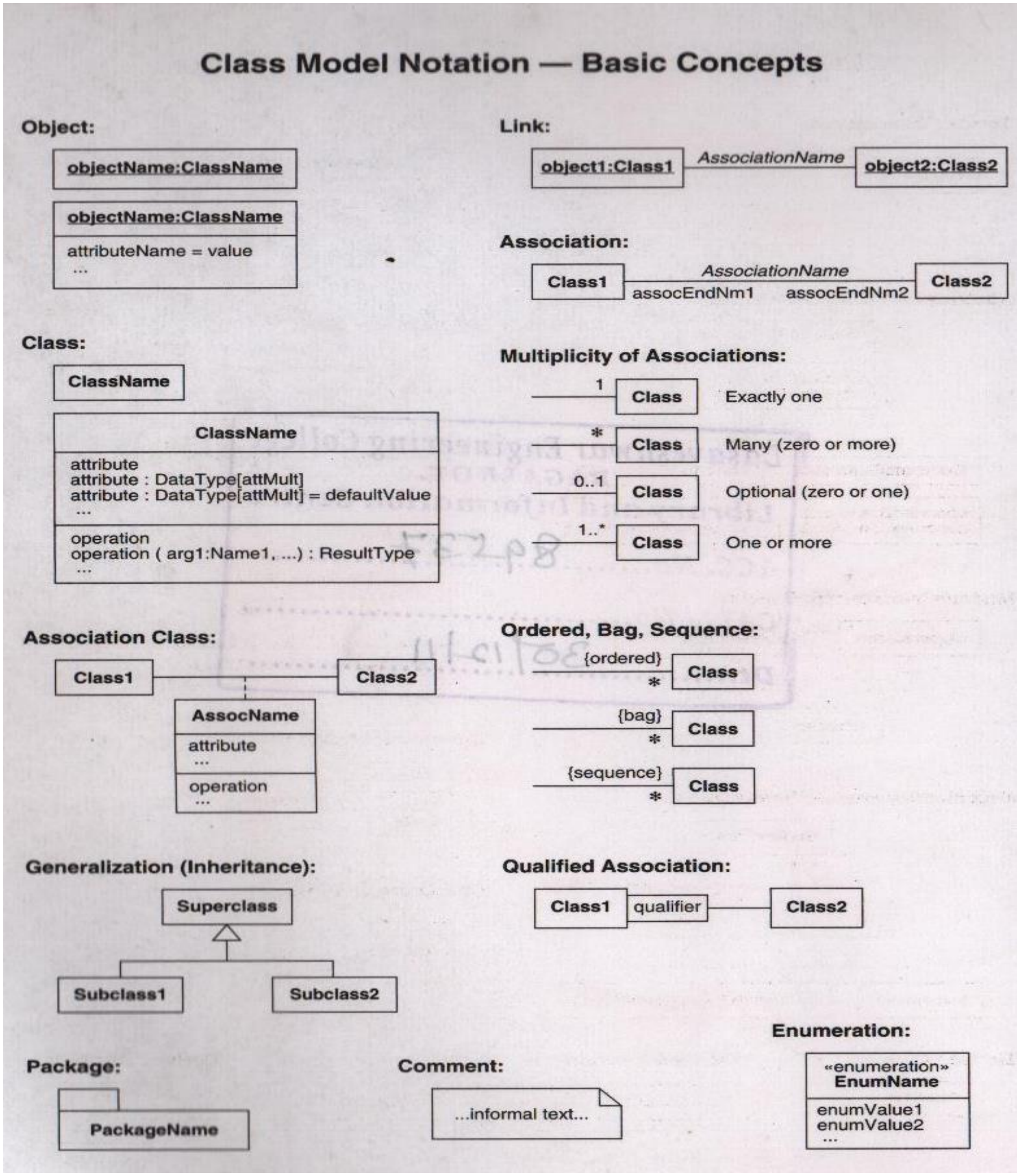


Figure 22. Package Diagram.

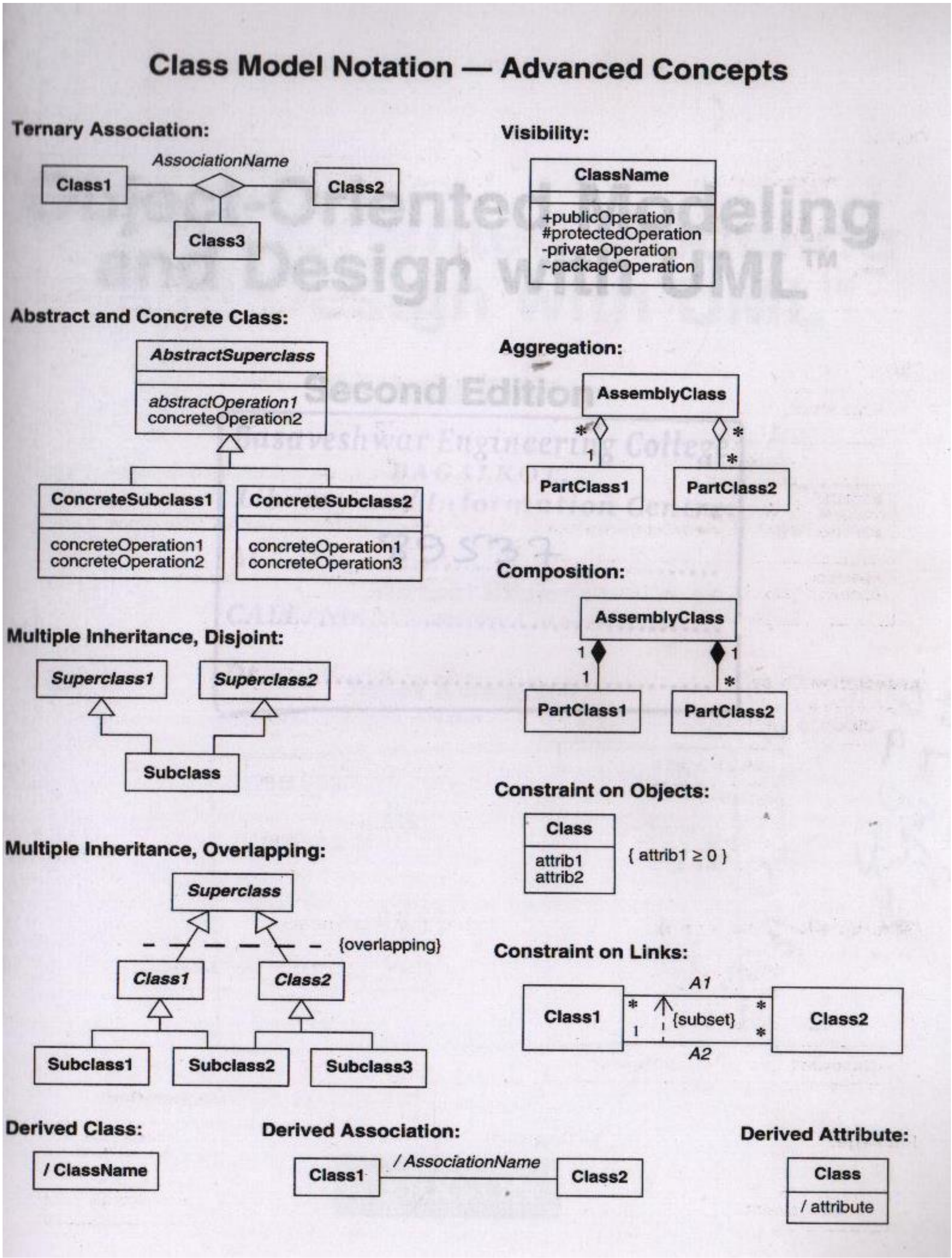
class

UML NOTATIONS

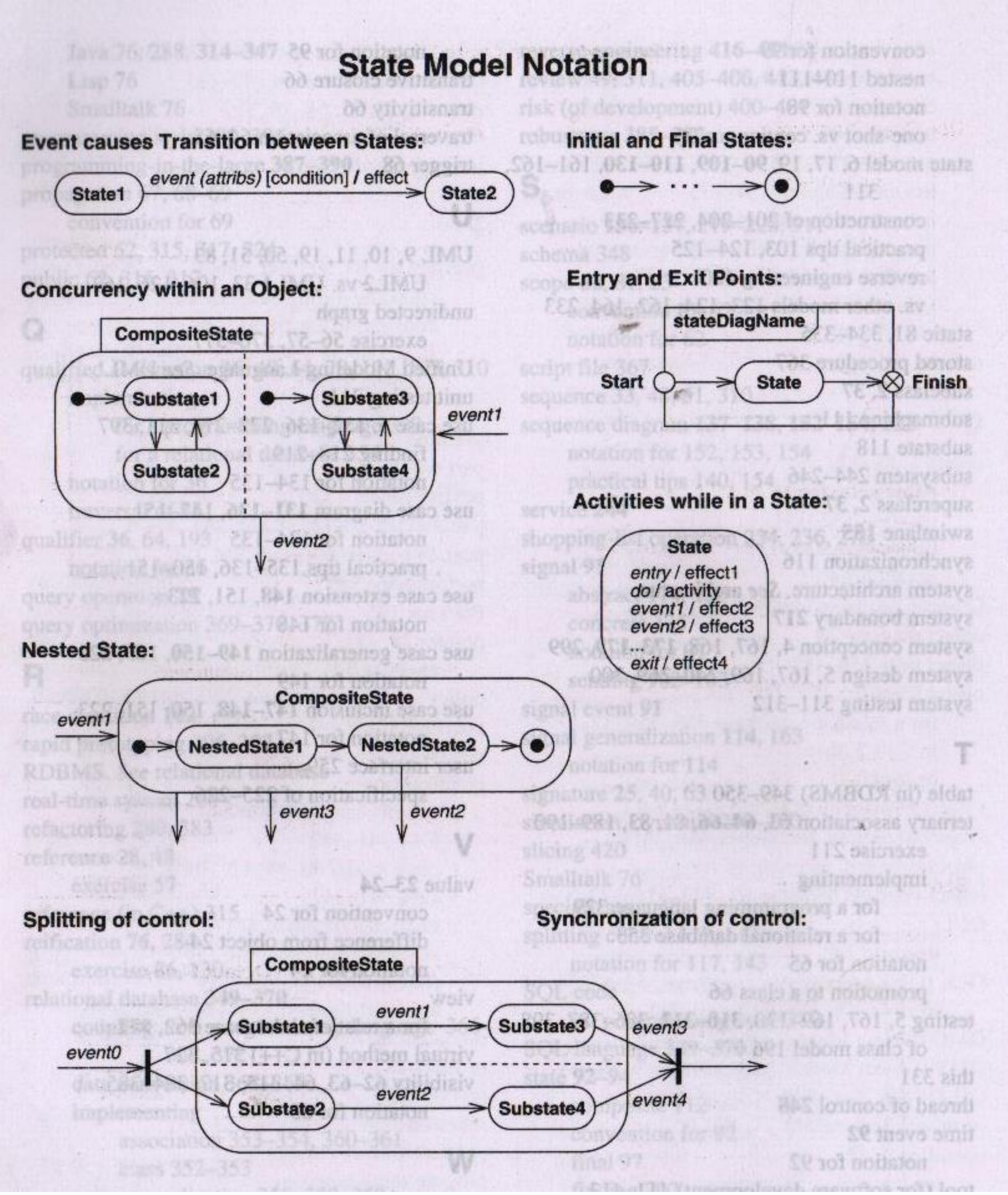
class


Class Model Notation — Basic Concepts
Object:
objectName:ClassName
objectName:ClassName
attributeName = value
...
Link:
object1:Class1
AssociationName
object2:Class2
Association:
Class1
AssociationName
assocEndNm1
assocEndNm2
Class2
Class:
ClassName
ClassName
attribute
attribute : DataType[attMult]
attribute : DataType[attMult] = defaultValue
...
operation
operation (arg1:Name1, ...) : ResultType
...
Multiplicity of Associations:
1
Class
Exactly one
*
Class
Many (zero or more)
0..1
Class
Optional (zero or one)
1..*
Class
One or more
Association Class:
Class1
Class2
AssocName
attribute
...
operation
...
Ordered, Bag, Sequence:
{ordered}
*
Class
{bag}
*
Class
{sequence}
*
Class
Generalization (Inheritance):
Superclass
Subclass1
Subclass2
Qualified Association:
Class1
qualifier
Class2
Package:
PackageName
Comment:
...informal text...
Enumeration:
«enumeration»
EnumName
enumValue1
enumValue2
...

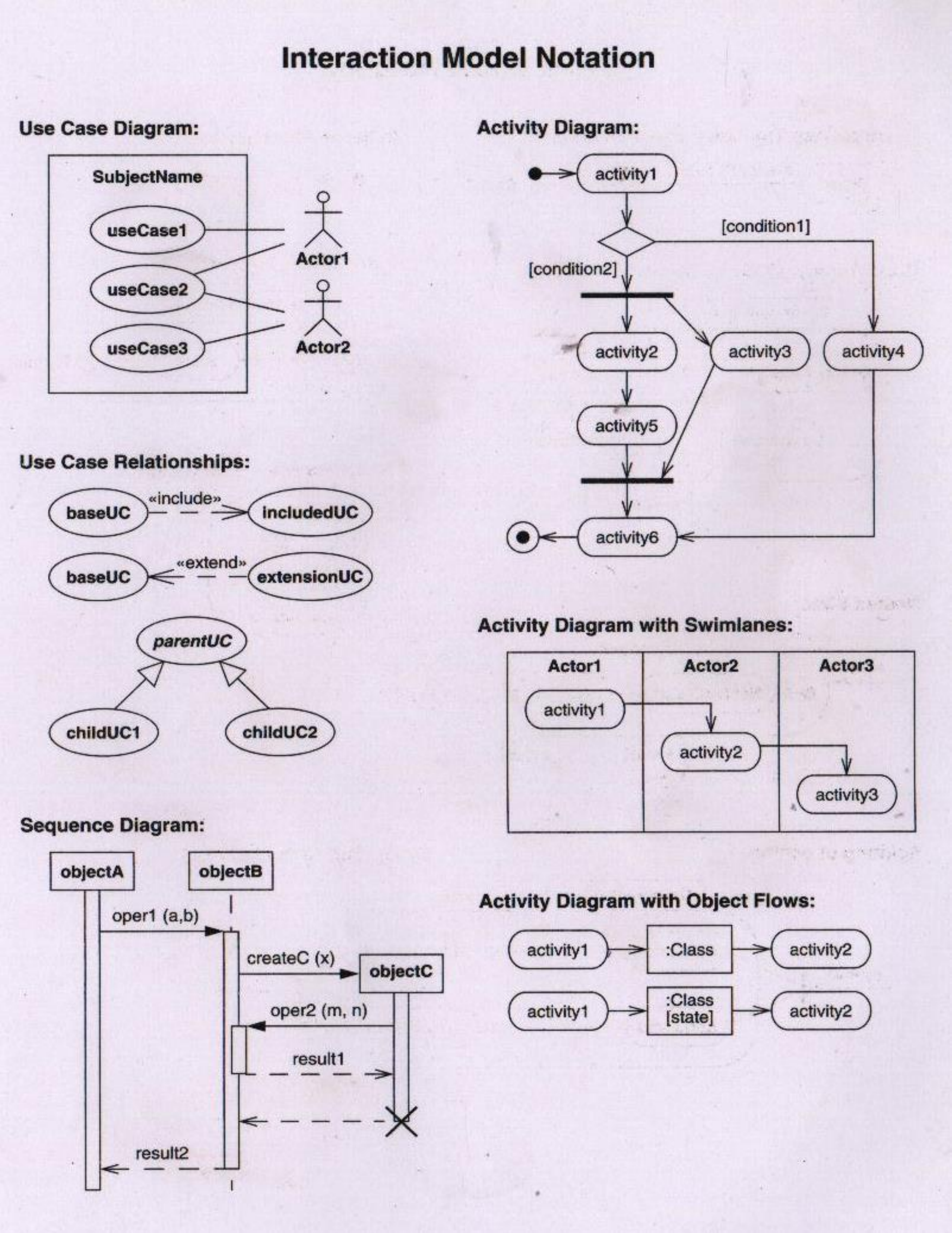
class


Class Model Notation — Advanced Concepts
Ternary Association:
AssociationName
Class1
Class2
Class3
Visibility:
ClassName
+publicOperation
#protectedOperation
-privateOperation
~packageOperation
Abstract and Concrete Class:
AbstractSuperclass
abstractOperation1
concreteOperation2
ConcreteSubclass1
concreteOperation1
concreteOperation2
ConcreteSubclass2
concreteOperation1
concreteOperation3
Aggregation:
AssemblyClass
*
1
*
*
PartClass1
PartClass2
Composition:
AssemblyClass
1
1
1
*
PartClass1
PartClass2
Multiple Inheritance, Disjoint:
Superclass1
Superclass2
Subclass
Constraint on Objects:
Class
attrib1
attrib2
{ attrib1 ≥ 0 }
Multiple Inheritance, Overlapping:
Superclass
{overlapping}
Class1
Class2
Subclass1
Subclass2
Subclass3
Constraint on Links:
A1
Class1
*
1
{subset}
*
*
Class2
A2
Derived Class:
/ ClassName
Derived Association:
Class1
/ AssociationName
Class2
Derived Attribute:
Class
/ attribute

class


State Model Notation
Event causes Transition between States:
State1
event (attribs) [condition] / effect
State2
Initial and Final States:
Concurrency within an Object:
CompositeState
Substate1
Substate2
Substate3
Substate4
event1
event2
Entry and Exit Points:
stateDiagName
Start
State
Finish
Activities while in a State:
State
entry / effect1
do / activity
event1 / effect2
event2 / effect3
...
exit / effect4
Nested State:
CompositeState
event1
NestedState1
NestedState2
event3
event2
Splitting of control:
CompositeState
event0
Substate1
Substate2
event1
event2
Substate3
Substate4
Synchronization of control:
event3
event4

class



Interaction Model Notation

Use Case Diagram:

Activity Diagram:

Use Case Relationships:

Activity Diagram with Swimlanes:

Sequence Diagram:

Activity Diagram with Object Flows:

class

ASSIGNMENT 1

Design the Library system: Identify the use cases of the system (Suggestive use cases: borrow book, return books, read news papers, reference and digital library). Develop the use case diagram, packages and documentation for the same. Preferable use of uses and extends relationships expected.

Class diagram:

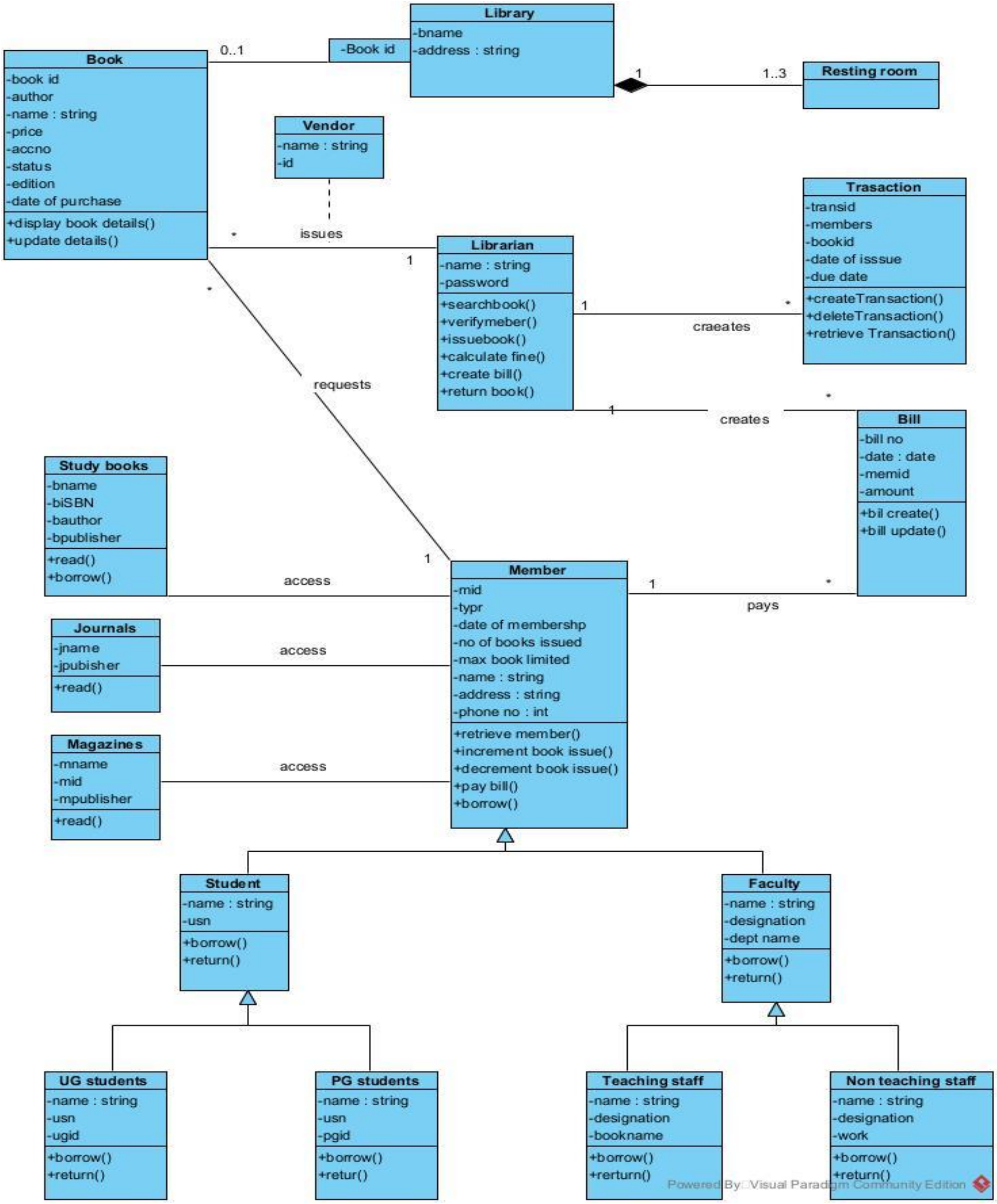


Fig 1.1 Class diagram of library system.

class

Use case diagram:

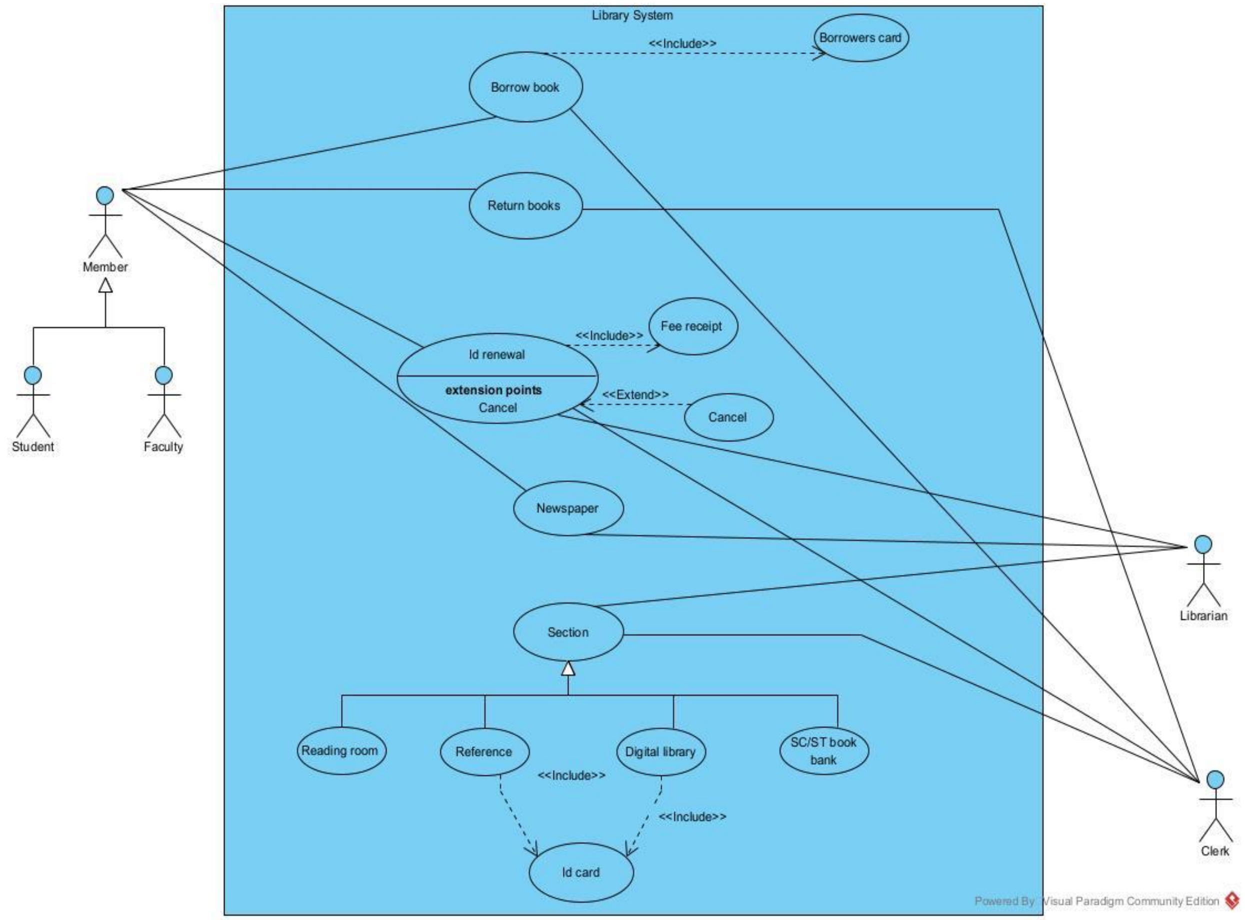


Fig 1.2 Use case diagram of library system

- **BORROW BOOK** : The student or staff takes book from library to read
- **BORROWER'S CARD** : The library staff will provide books by issuing the borrowers card
- **RETURN** : The student brings borrowed books back to the library
- **ID RENEWAL** : A student submit his/her admission receipt and old ID card for the renewal
- **READ NEWS PAPER** : The student or staff can read the news papers available in the library
- **DIGITAL LIBRARY** : The digital library enables student or staff to get the information on web or internet
- **SC/ST BOOK BANK** : The library provides free books for sc / st students.

class

PACKAGE DIAGRAM:

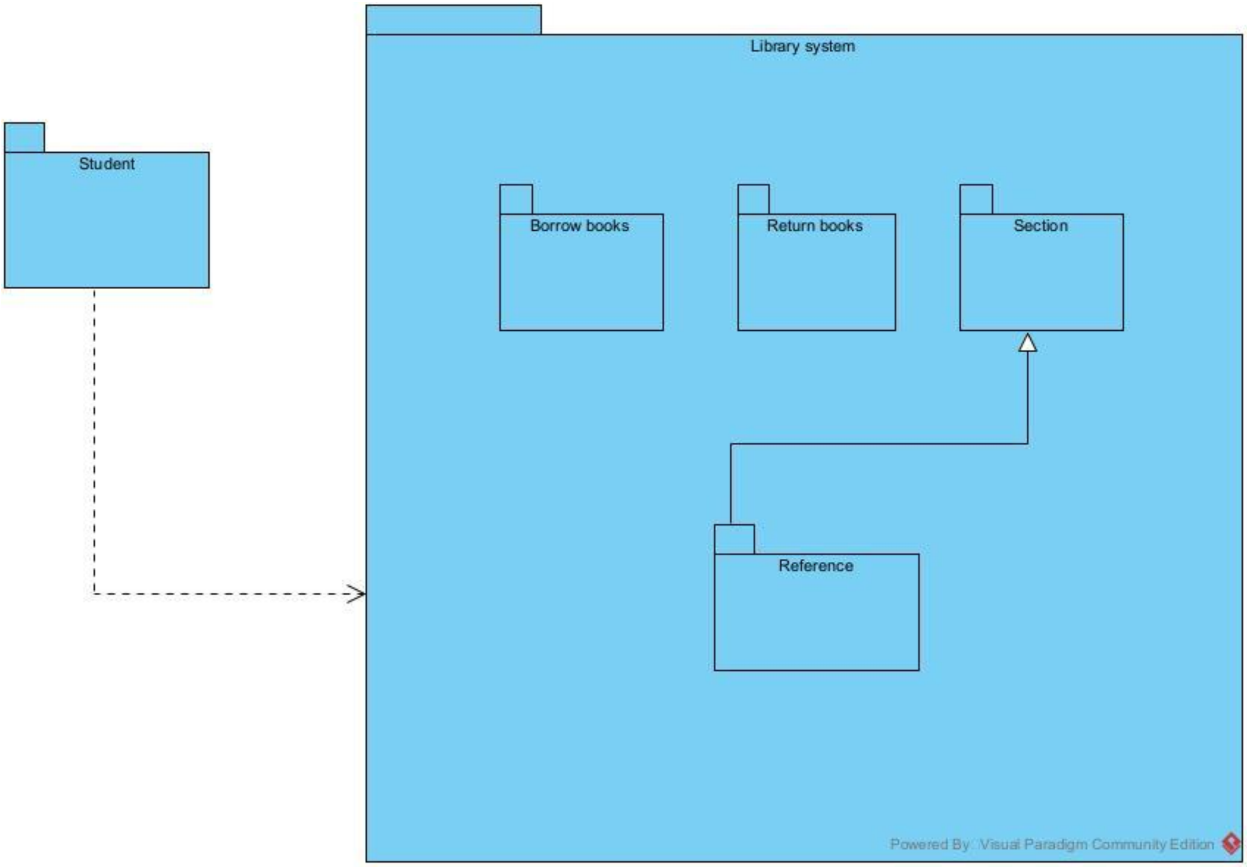


Fig 1.3 Package diagram of library system

Package is a collection of elements such as classes, associations, generalizations and lesser packages. The above package diagram considered that student package depends on library system which includes borrow books, return books and do research.

class

ASSIGNMENT 2

Design the Examination system: Identify the use cases. (Suggestive use cases:– Form-filling, Get Hall Ticket, Write exam, get result Verify Hall Ticket) Develop the use-case diagram, Packages and documentation for the same. Preferable use of uses & extends relationships expected.

class

USE CASE DIAGRAM:

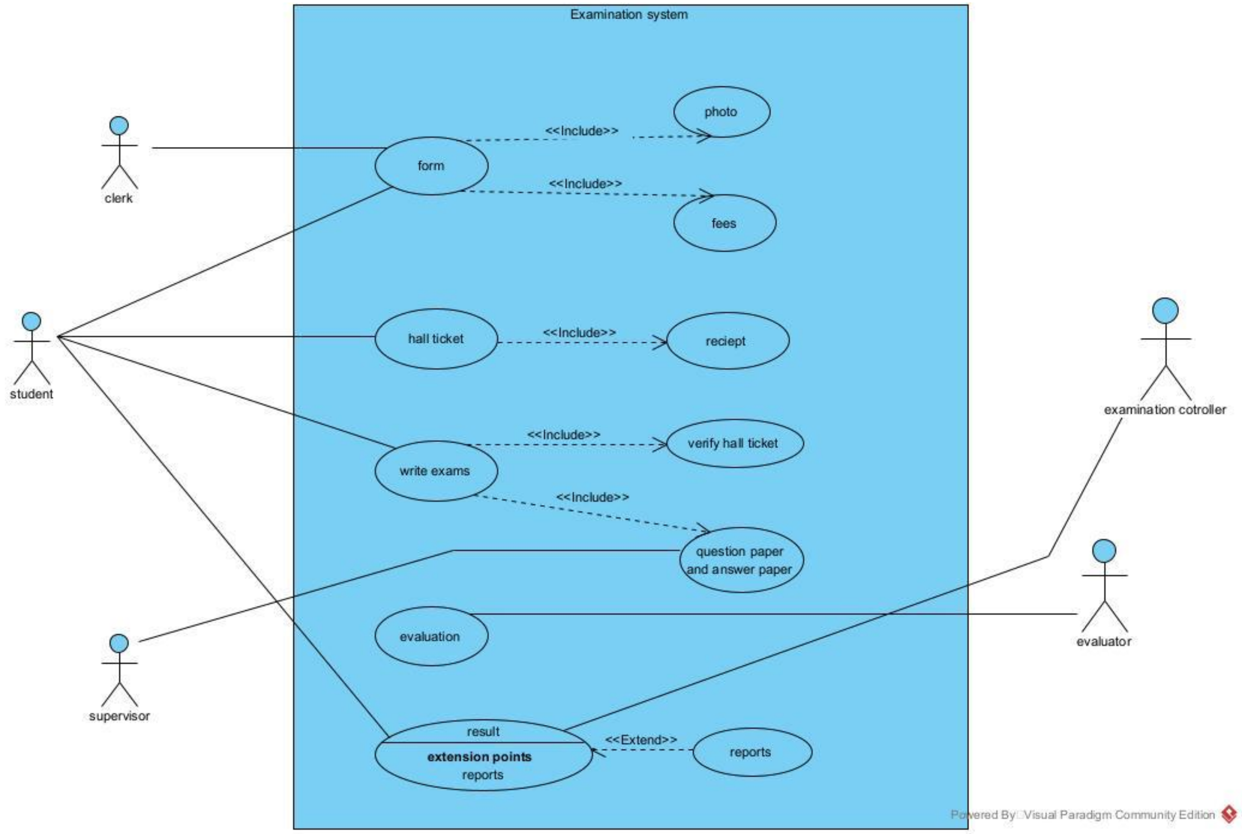


Fig 2.1 Use case diagram of examination system

CASES OF EXAMINATION SYSTEM

Form: It is the first process of examination system, here the student fills up the examination form which includes name, sem, usn, subjects taken, along with photograph, finally student submits the form for the verification also pays examination fees and takes receipts.

Get Hall ticket: Involves receiving of hall ticket from examination clerk by showing ID card.

Write Exam: The student writes the exam, the supervisor provides the question paper.

Evaluator: Here evaluator evaluates the answer papers.

Get Results: Controller of the examination system announce the result where results are recorded, each students can get their results by USN.

class

PACKAGE DIAGRAM:

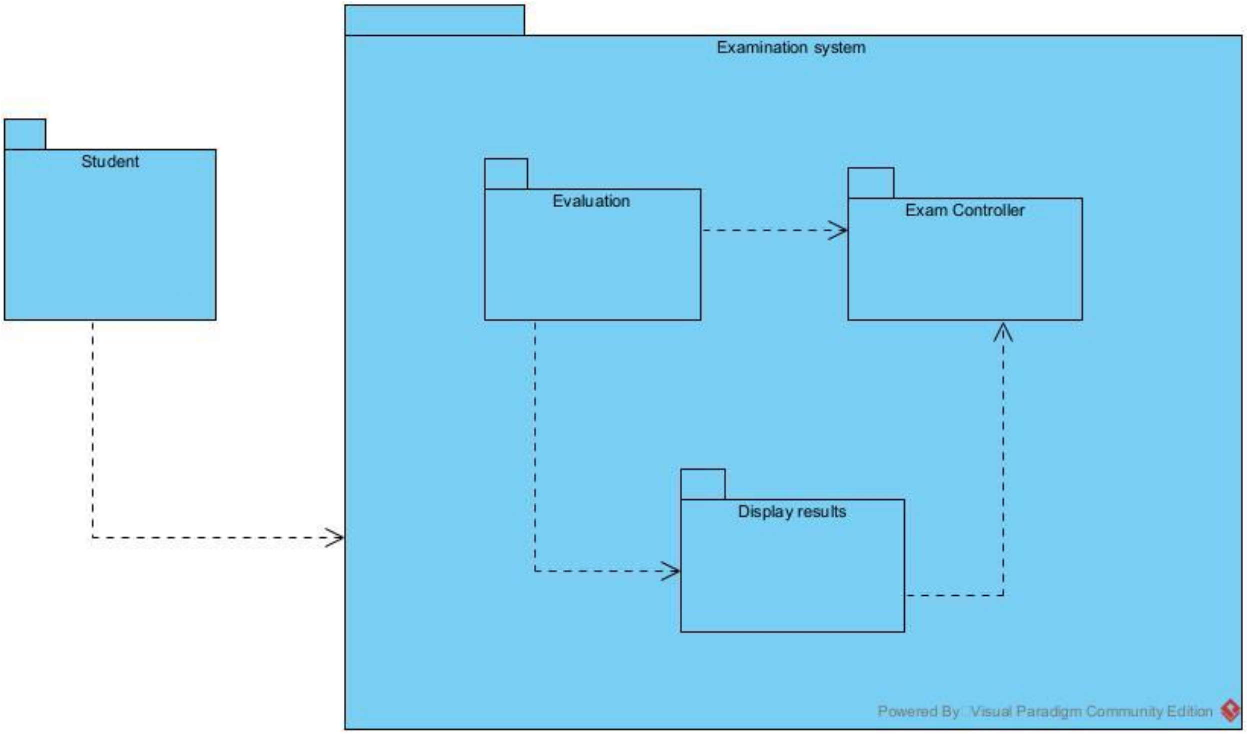


Fig 2.2 Package diagram of examination system

Package is a collection of elements such as classes, associations, generalizations and lesser packages. The above package diagram considered that display results package is depends upon evaluation package as well as controller of examination packages. Hence student is completely dependent on whole depend examination system.

class

ASSIGMENT 3

Analyze and design the system for ATM Transaction: Identify the use cases.(Suggestive use cases: Transaction, Approval process, Invalid PIN, Deposit Amount, Deposit savings, Deposit checking, withdraw Amount, withdraw checking, saving, withdraw saving denied, checking Transaction History, saving Transaction History). Package, documentation Develop the use case diagram, Packages and documentation for the same. Draw the essential class diagrams.

class

USE CASE DIAGRAM:

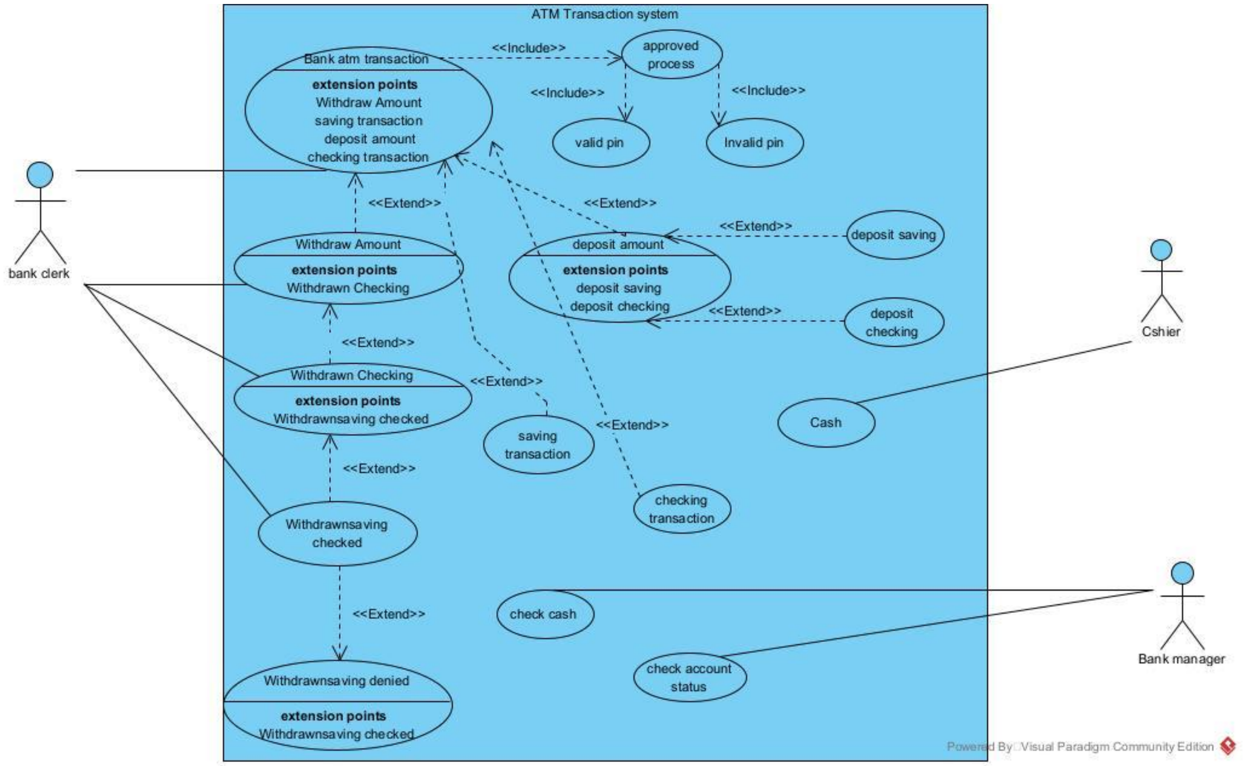


Fig 3.1 Use case diagram of Atm transaction system

USE CASES OF ATM TRANSACTION

• **BANK ATM TRANSACTION**: The bank clients interact with the bank system by going through the approval process. After the approval process, The bank B client can perform the transaction. The transaction involves steps like insert ATM card, Perform the approval process, Ask the type of transaction, Enter the ype of transaction, Perform the transaction, Eject card, Request take card and Take card.

• **APPROVAL PROCESS**: The client process consists a PIN code of four digits. If the PIN code is valid, the client's accounts become available.

• **INVALID PIN**: If PIN code is not valid ,an appropriate message is displayed to the client.

• **DEPOSIT AMOUNT**: The bank clients interact with the bank system after the approval process by requesting to deposit money to an account. The client selects the account for which a deposit is going to be maid and enters the amount to be deposited.

class

- **DEPOSITE SAVINGS**: The client selects the savings account for which deposit is going to be made.
- **DEPOSITE CHECKING**: The client selects the checking account for which a deposit is going to be made. This use case extends the deposit amount use case.
- **WITHDRAW AMOUNT**: The bank clients interact with the bank system by requesting to withdraw money from an account. The client tries to withdraw an amount from a checking account. After verifying that the funds are sufficient, the transaction is performed.
- **WITHDRAW CHECKING**: The client tries to withdrawn an amount from his or her checking account. The amount is less than or equal to the checking account's balance, and the transaction is performed.
- **WITHDRAW SAVINGS**: The client tries to withdraw an amount from a saving account. The amount is less than or equal to the balance and the transaction is performed on the saving account.
- **DENIED WITHDRAW SAVINGS**: The client withdraw amount from a savings account. If the amount is more than the balance, the transaction is halted and a message is displayed.
- **CHECKING TRANSACTION HISTORY**: The bank client requests a history of transactions for checking account.
- **SAVING TRANSACTION HISTORY**: The bank client requests a history of a transaction for a savings account.

class

CLASS DIAGRAM:

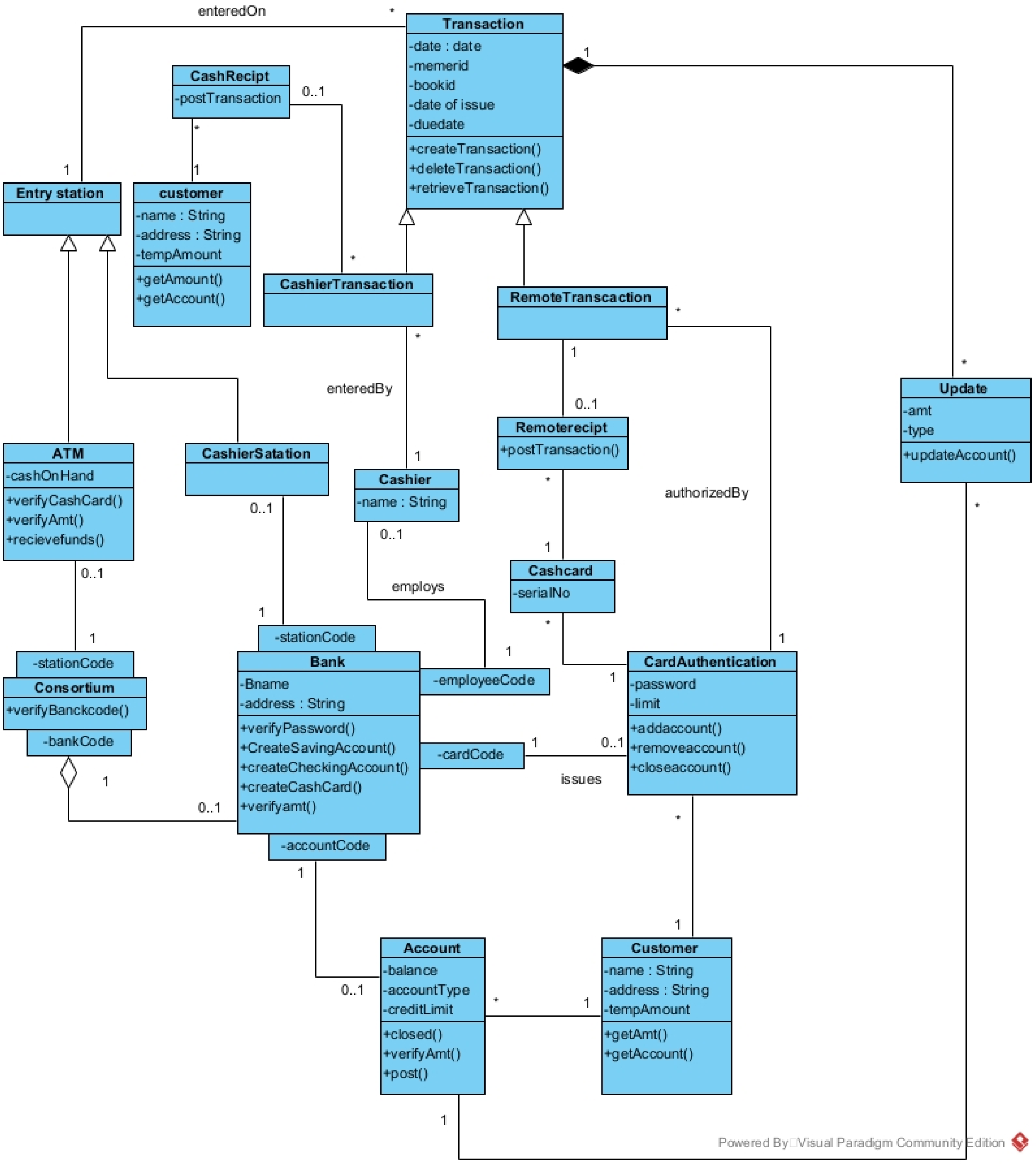


Fig 3.2 Class diagram of Atm transaction system

class

Transaction: Handles all transactions. It is related to the classes entry station(* to 1) and update(1 to *).

Cashier receipt: provides balance receipt for the customer after the transaction has been completed. It is related to the classes customer(* to 1) and cashier transaction(0..1 to *).

Entry station: Transaction can be done on both Cashier Station and ATM. Entry Station **generalizes** cashier station and ATM

Customer: Maintains database of customers which includes details of ach customer. It is related to the classes card authorization(1 to *) and account(1 to *).

Cashier transaction: Maintains information about all transactions. It is related to the classes cash receipt(* to ..1) and cashier(* to 1).

Remote transaction: Handles remote transactions. It is related to the classes transaction card authorization(* to 0..1).

Update : Updates all the transactions. It is related to the classes transaction(* to 1) and account(* to 1).

ATM: switches on system to begin the transaction. Shut down the system. Begins new transaction for the customer. It is related to the classes card authorization (* to *) bank (0..1 to 1) customer(* to 1) and update(1 to *) .

Cashier station: Maintains information about transactions held in bank. It is related to the classes entry station and bank (0..1 to 1).

Cashier : Maintains information about cahier. It is related to the classes cashier transaction(1 to *) and bank(0..1 to 1).

Remote accept: Accepts remote transactions. It is related to the classes remote transaction(0..1 to *) and cash card(* to 1)

Card authorization: verifies customer’s cash card to begin transaction. It is related to the classes remote transaction (1 to *) cash card(1 to *) bank(0..1 to 1) account(* to *) and customer(* to 1).

Cash card: Maintains information about customer’s cash card. It is related to the classes remote receipt(1 to *) and card authorization(* to 1).

Bank: Maintains information about the entire day to day activities in the bank and its details. It is related to the classes cashier station(1 to 0..1) cashier(1 to 0..1) consortium(0..1 to 1) account(1 to 0..1) and card authorization(1 to 0..1).

Consortium: Maintains information about all banks. It is related to the classes atm(1 to 0..1) and bank(1 to 0..1).

class

Account: Maintains information about customers account. It is related to the classes bank(0..1 to 1) update(1 to *) card authorization(* to *) and customer(* to 1).

Package diagram:

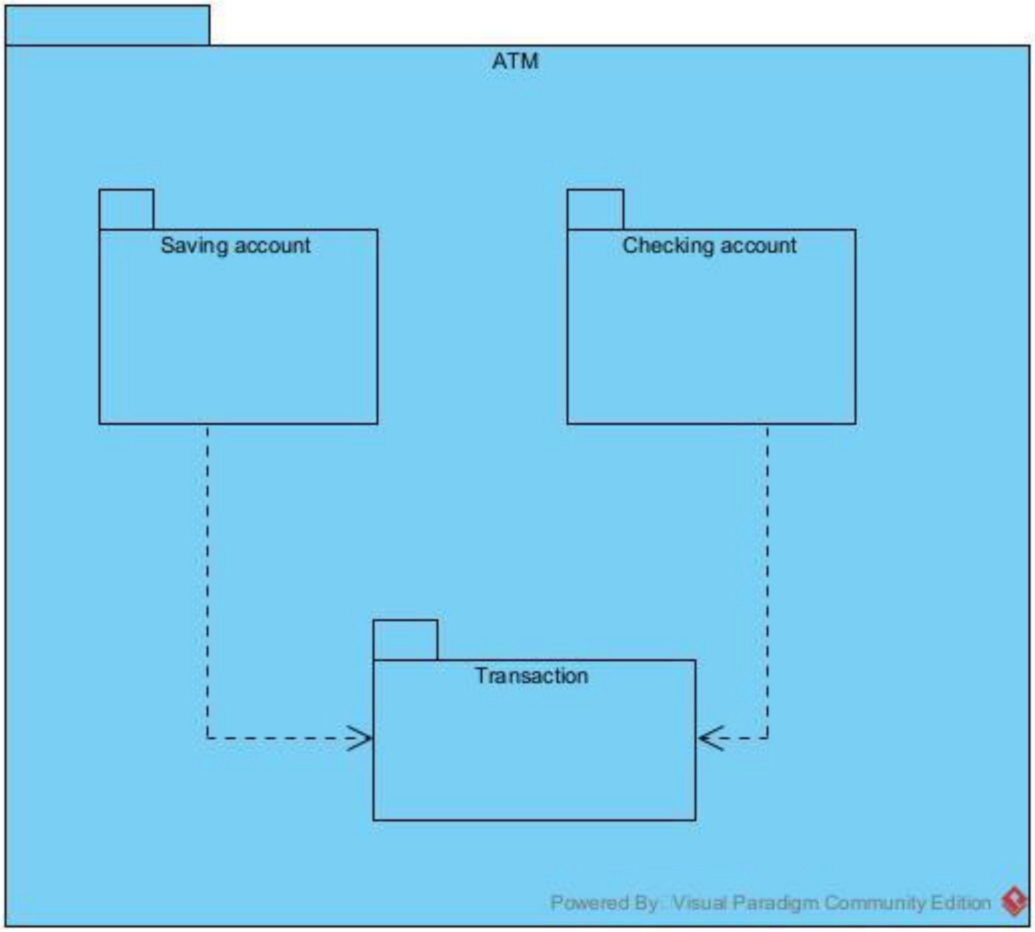


Fig 3.3 Package diagram of Atm transaction system

The package diagram shows how various classes are grouped into packages. Each of these classes in turn depends upon the package ATM which contains the class atm which represents the system as a whole. The sub packages saving account and checking account depends upon the sub package transaction.

class

ASSIGNMENT 4

Analyze and design the system for Electronics voting system (The actors are presiding officer, 1st polling officer, 2nd polling officer, voters list, Election officer, voter candidate, EVM ID; Processes: Vote counting, and announcement of results). Develop the use case diagram, Packages and documentation for the same. Draw the essential sequence diagrams and state chart diagrams

class

ELECTRONIC VOTING SYSTEM (EVM):

An election is a formal decision-making process by which a population chooses an individual to hold public office. In olden days election used to take place manually, but recently due to Electronic Voting Machine, the entire system is automated.

Electronic voting (also known as e-voting) is a term encompassing several different types of voting, embracing both electronic means of casting a vote and electronic means of counting votes. Electronic Voting Machine (EVM) retains all the characteristics of voting by ballot papers while making polling a lot more expedient. Being fast and absolutely reliable, the EVM saves considerable time, money and man power. And of course, helps maintain total voting secrecy without the use of ballot papers. The EVM is 100% tamper proof. And at the end of polling, just press a button and there we have the results.

An EVM unit comprise of two units that can be interlinked, a ballot unit which a voter uses to exercise his vote and the other a control unit used by the polling officials.

class

USE CASE DIAGRAM:

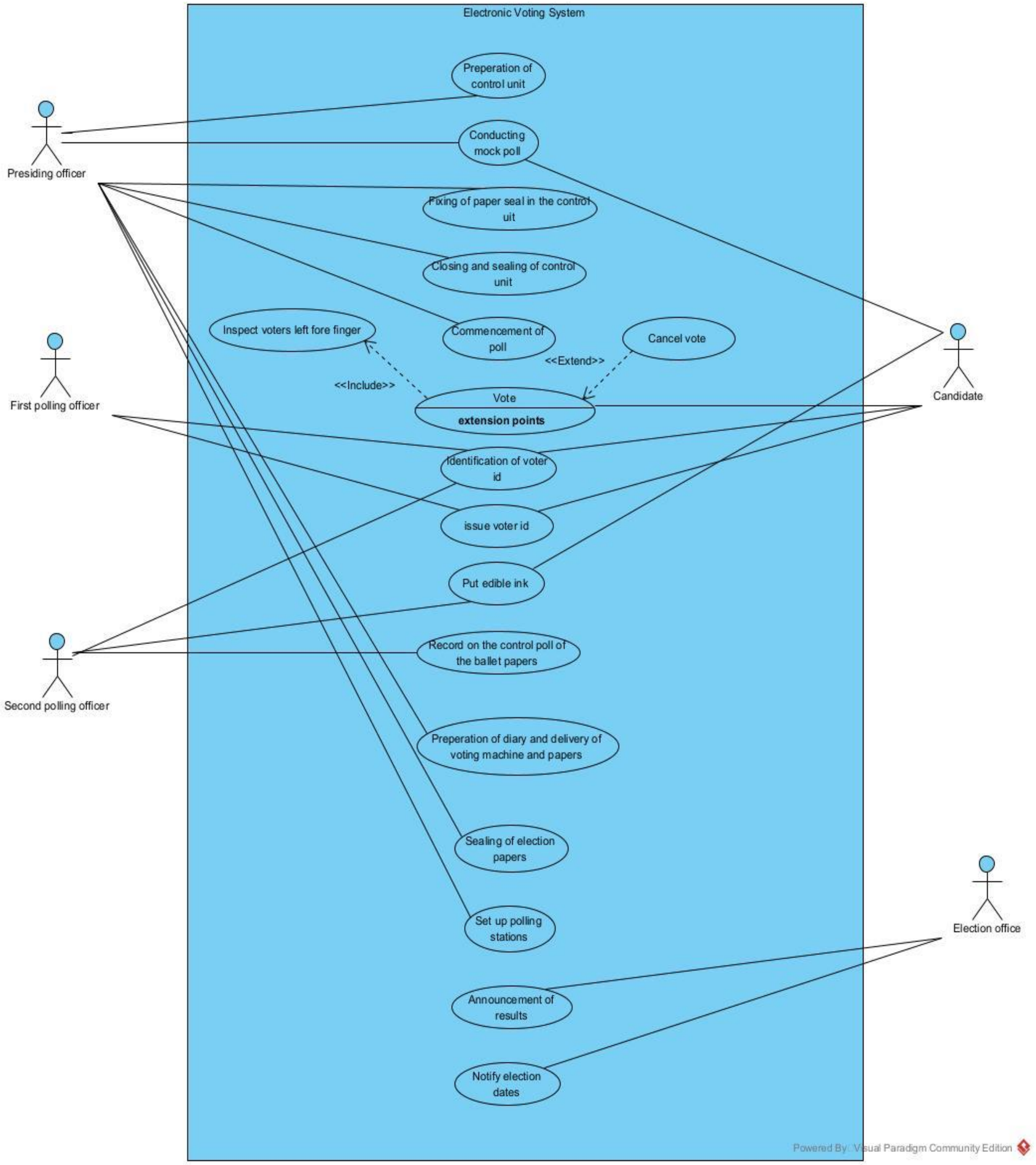

Electronic Voting System
Preperation of control unit
Conducting mock poll
Fixing of paper seal in the control uit
Closing and sealing of control unit
Inspect voters left fore finger
Commencement of poll
Cancel vote
<<Extend>>
<<Include>>
Vote
extension points
Identification of voter id
issue voter id
Put edible ink
Record on the control poll of the ballet papers
Preperation of diary and delivery of voting machine and papers
Sealing of election papers
Set up polling stations
Announcement of results
Notify election dates
Presiding officer
First polling officer
Second polling officer
Candidate
Election office
Powered By: Visual Paradigm Community Edition

Fig 4.1 Use case diagram of electronic voting system

class

USE CASE DIAGRAM DESCRIPTION:

- **Preparation** of **control** unit: Setting up the control unit for the voting.

  Conducting mock poll: Demonstration of voting by EVM machine to the user /presiding officer.
- **Fixing of paper seal** in **the control unit**: fixing of poll Preparation of control unit Conducting mock poll Fixing of green paper seal in the control unit Closing and sealing of control unit Commencement of the poll...
- **Closing and sealing of control unit**:Closes the control unit after the voting.
- **Commencement of poll**: Announcement of the polling.
- **Vote:** The candidate put the vote to the right person.
- **Identification of voter id**:The polling officer verify/identify the voter id before the voting.
- **Issue voter id**:candidate 18 and above are issued with the voter id by government.
- **Put indelible link**:Before voting the presiding officer put the indelible link to the fore finger.
- **Order copy of the electrol roll**:It is paper-based voting system where votes are casted and counted manually,using ballets.
- **Inspector voters left fore finger**: Presiding officer checks the fore finger to reduce the duplication of the vote..
- **Preparation of diary and delivery of voting**:The presiding officer prepares the voting foe delivery.
- **Sealing of election paper**:After completion of voting seal the election paper.
- **Announcement of results**: announces the commisioner results.
- **Notify election date**:Election officer are going to announce the date.

class

SEQUENCE DIAGRAM:

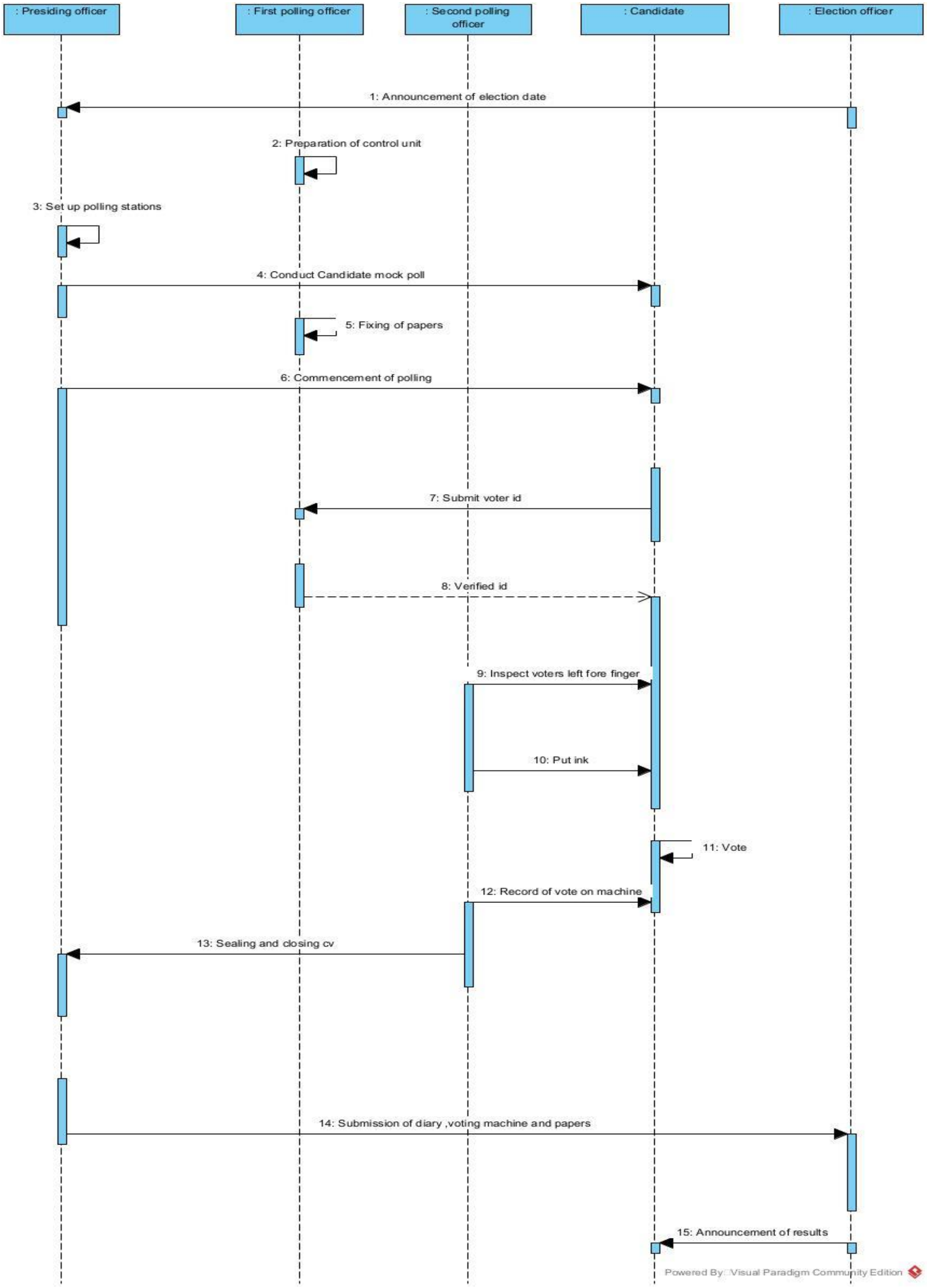

: Presiding officer
: First polling officer
: Second polling officer
: Candidate
: Election officer
1: Announcement of election date
2: Preparation of control unit
3: Set up polling stations
4: Conduct Candidate mock poll
5: Fixing of papers
6: Commencement of polling
7: Submit voter id
8: Verified id
9: Inspect voters left fore finger
10: Put ink
11: Vote
12: Record of vote on machine
13: Sealing and closing cv
14: Submission of diary ,voting machine and papers
15: Announcement of results
Powered By: Visual Paradigm Community Edition

Fig 4.2 Sequence diagram of electronic voting system

class

SEQUENCE DIAGRAM DESCRIPTION:

The EVM machine of sequence diagram represents the presiding officer,1st polling officer,2nd polling officer, Candidate ,Election officer. The Election officer is associated with presiding officer by election of announcement of election date then the 1st polling officer prepares the control unit. The presiding officer set up the polling station. The pressing officer associated with candidate by conduct mock poll ,after conducting the mock poll the 1st polling officer seal and close the election paper. The presiding officer announce the candidate result. The candidate issue the voter id from the 1st polling officer. The polling officer verify the voter id. The second polling officer put the ink to candidate then votes are casted and counted manually, finally the election officer associated with 2nd polling officer, the second polling officer announce the candidate results.

State diagram:

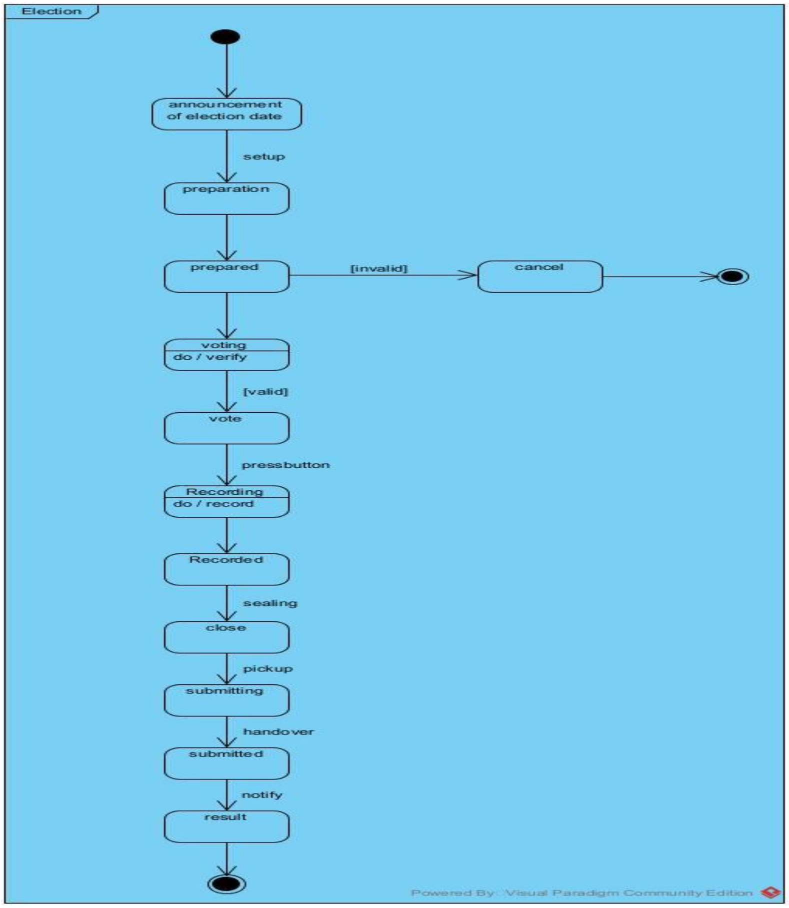


Fig 4.3 State diagram of electronic voting system

class

ASSIGNMENT 5

Analyze and design the system for Employee reference. (The Process HR Manager contacts Employees of his company and HR manager of other company to publicize about the vacancy. The person, who has referred the right candidate, will be given bonus. Interview, Short-listing, selection list announcement, Bonus for referred employees are all parts of the process.). Develop the use case diagram, sequence diagrams and state chart diagrams.

class

USE CASE DIAGRAM:

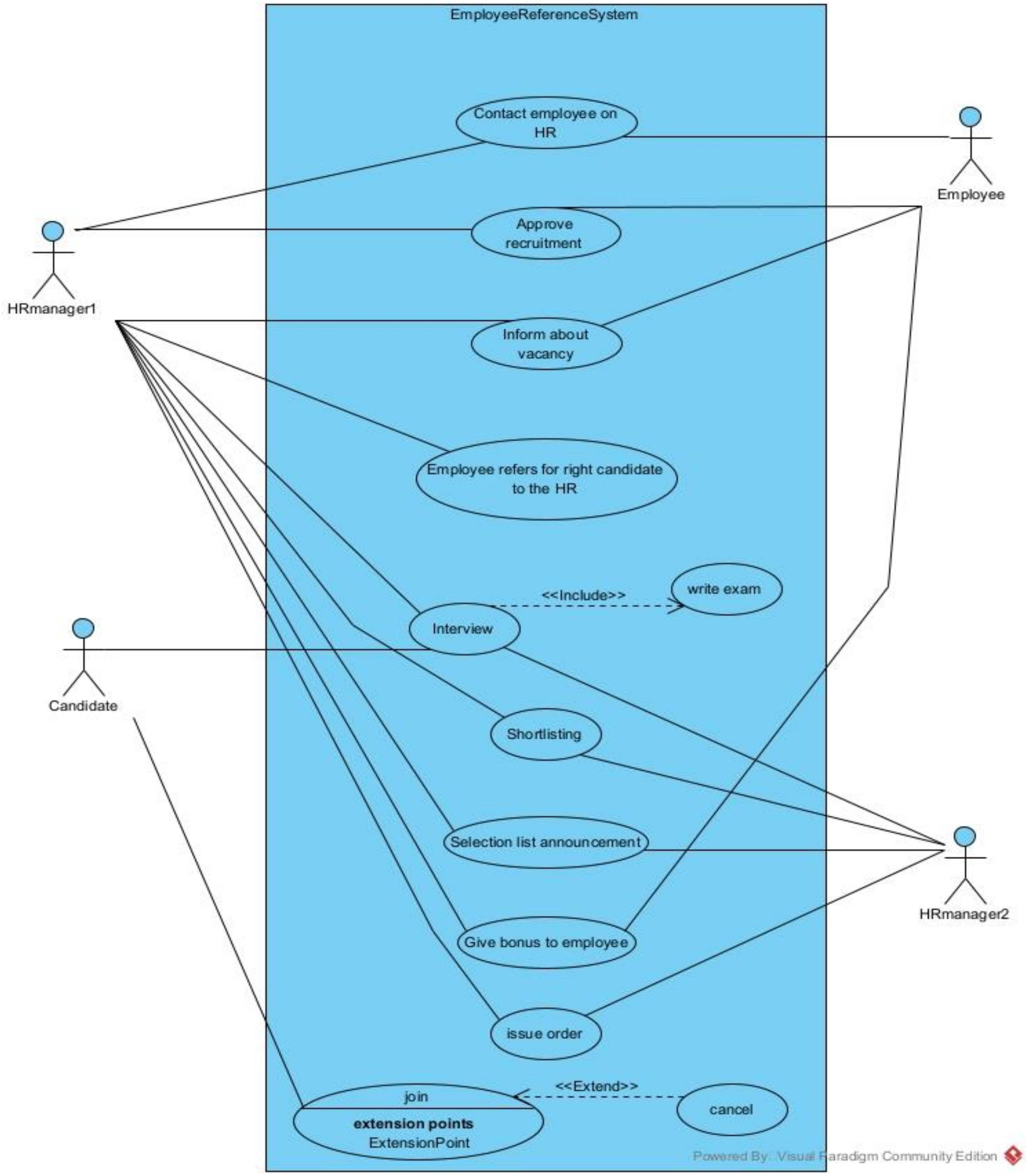


Fig 5.1 Use case diagram of employee reference system

-

class

- **Contacts employee or HR**: The HR contacts the employee or HR manager of the other company for vacancies.
- **Approval for Recruitment**: The employee gives an approval for the recruitment of the candidates to the HR.
- **Inform about vacancy:** The HR will inform employee regarding the vacancies.
- Employee refers right candidate to HR: Employee refers the right candidate to the HR.
- **Interview:** During interview, candidate writes the exam.
- **Short listing:** Selected candidates are short listed.
- **Selection list announcement:** Selected candidates list will be announced.
- **Give bonus to employees:** The right candidate referred by the employee will be given bonus.
- **Issue order:** HR will issue order copy.
- **Join:** The candidate will join the company.

class

SEQUENCE DIAGRAM:

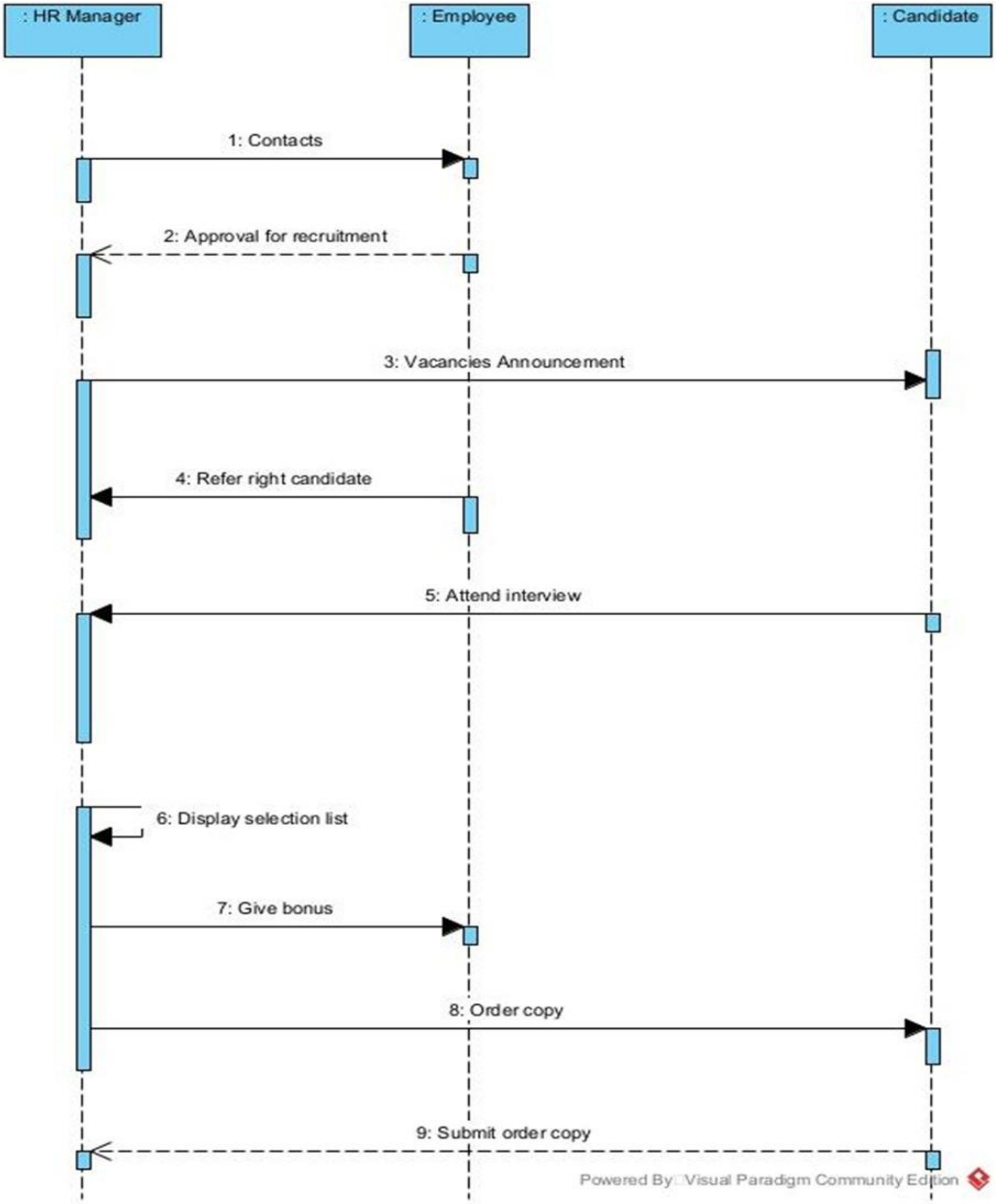


Fig 5.2 Sequence diagram of employee reference system

class

The entire sequence diagram depicts the actors in the employee reference system and how they interact with the system. The HR manager will contact the employee for the vacancy. If there is a vacancy then, the employee will inform to the HR manager to recruit the candidates. HR will announce for the vacancy.

Employees refers the right candidate. If the right candidate gets selected then, the employee who has given reference to the selected candidate will be rewarded by bonus.

State DIAGRAM:

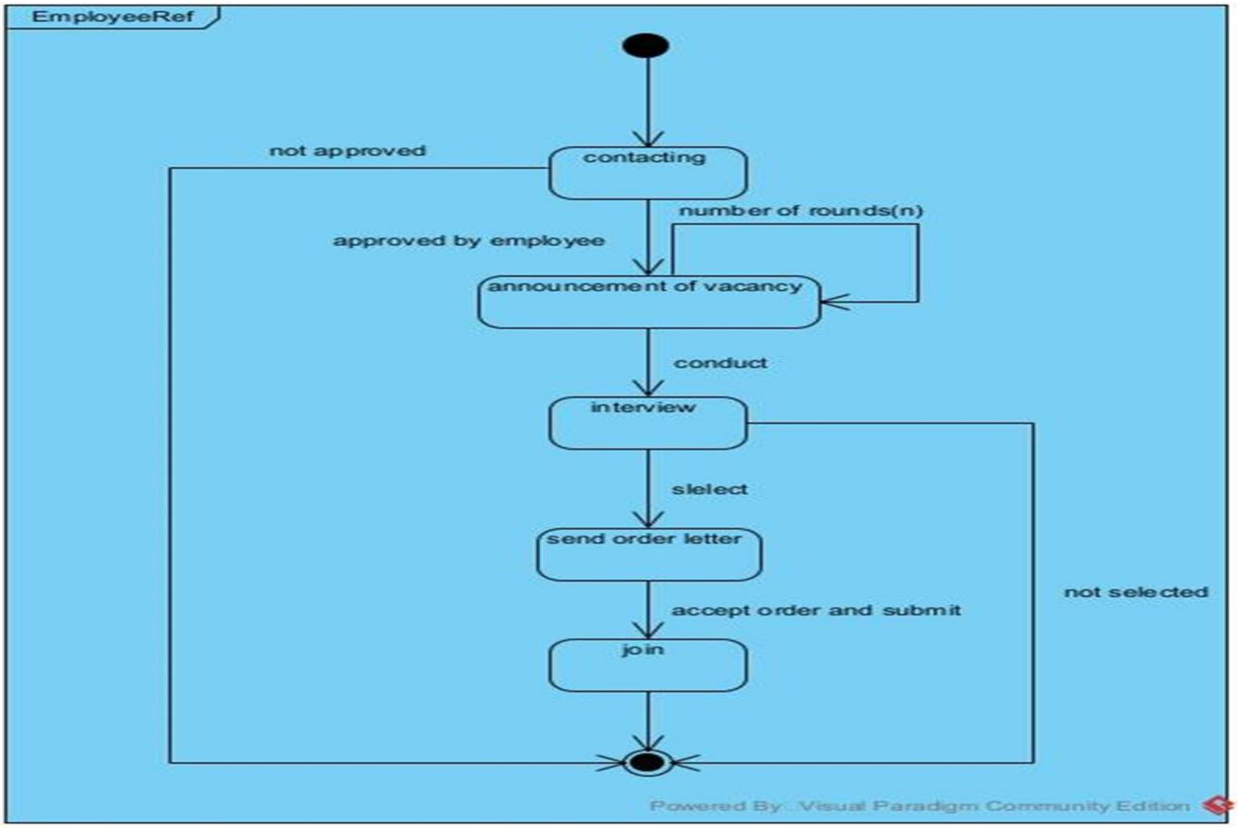


Fig 5.3 State diagram of employee reference system

This diagram depicts the flow of control of the employee reference system. It begins with the HR contacting an employee. An employee of the firm referencing a candidate for the job. If the employee gives reference then the referred goes through further steps of this process or else exits the whole recruitment process. The selected candidates have give interviews. If a candidate is selected then he is issued with joining order. The candidates are expected to join on the date given on joining order.

class

ASSIGNMENT 6

Analyze and design the system for vehicle purchase, registration and licensing systems. Develop the use case diagram, sequence diagram, activity diagram, packages.

class

Use Case diagram:

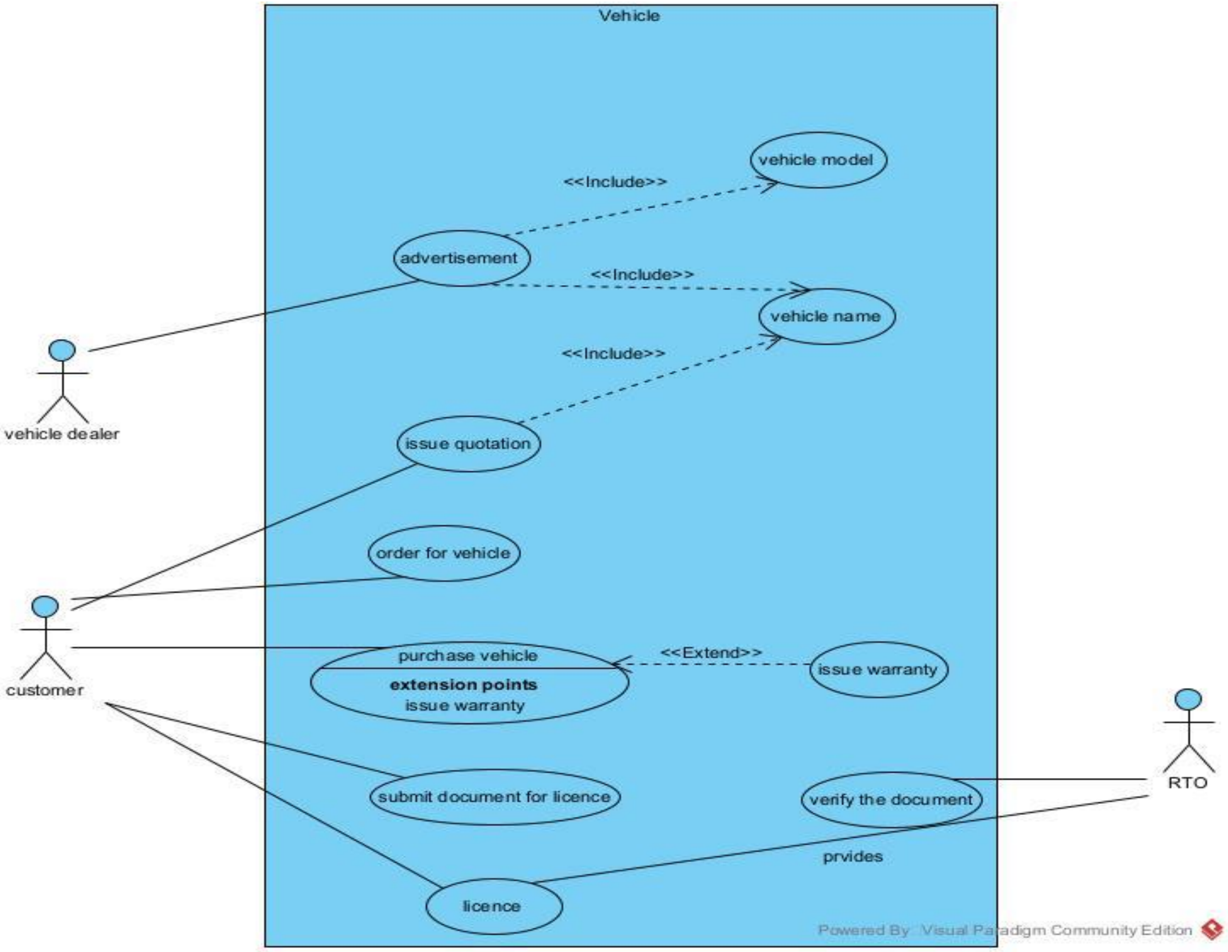


Fig 6.1 Use case diagram

Use cases:

- **Advertisement**: The dealer advertises the vehicle name and vehicle model.
- **Issue quotation:** The dealer provides the quotation to the customer.
- **Order for vehicle**: Customer orders for the particular vehicle.
- **Purchase vehicle :** Customer purchases the vehicle which he has ordered.
- **Submit documents for license :** Owner of the vehicle submits the required documents to the RTO officer in request for license.
- **License :** By verifying the submitted documents RTO officer issues license to the owner.

class

Sequence diagram:

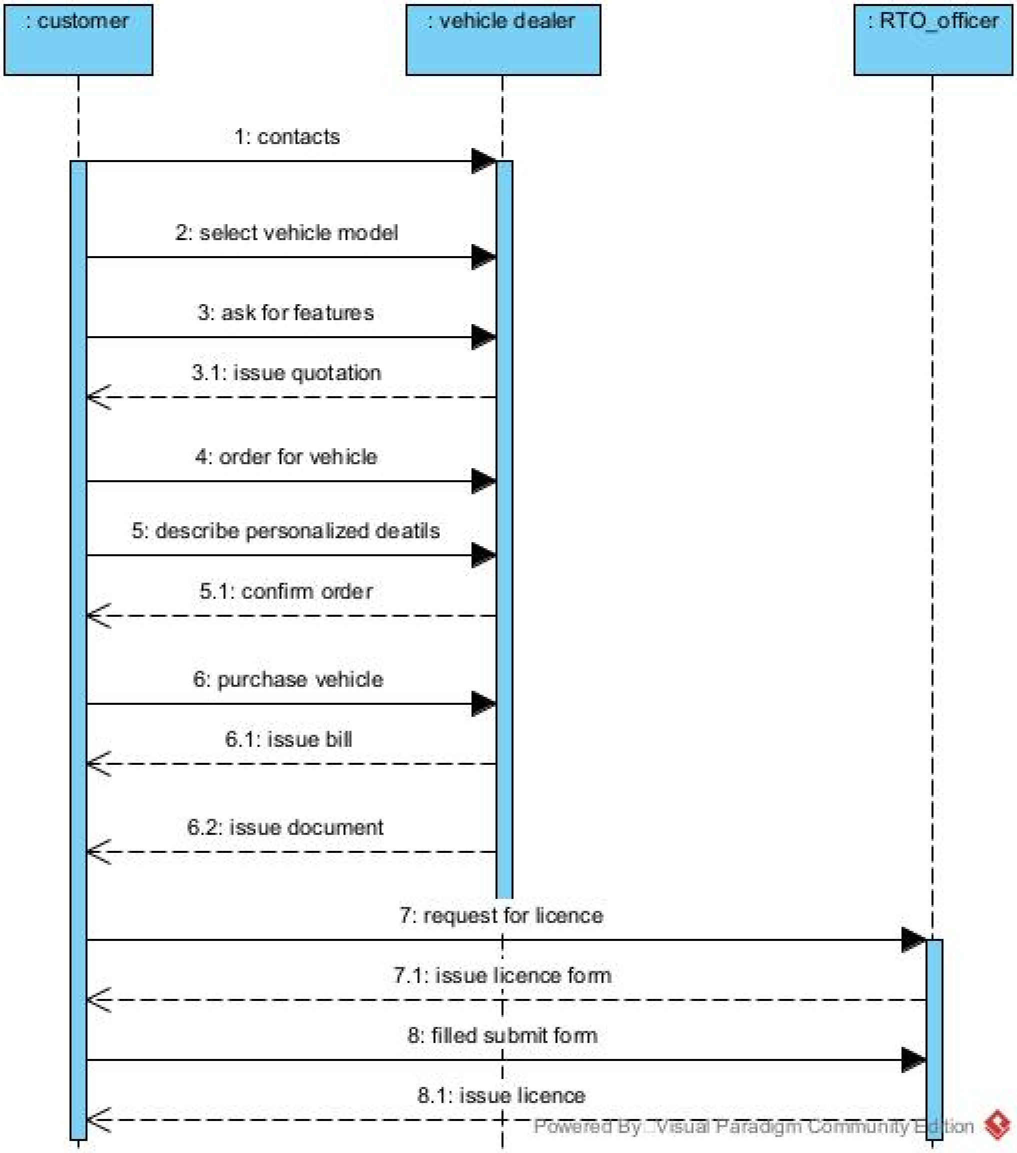


Fig 6.2 Sequence diagram

Sequence diagram shows an interaction between the objects. The above sequence diagram shows the interaction between the customer, dealer and RTO officer. The customer contacts the dealer, selects the vehicle and asks for its details. The dealer provides the quotation to the customer and customer orders the vehicle, describes his personal details. The dealer confirms the order and customer purchases the vehicle. The customer issues the bills and documents of the vehicle. The customer requests the RTO officer for the license and issues the license form. After filling the form, customer submits the form and will issue the license by the RTO officer.

class

Activity diagram:

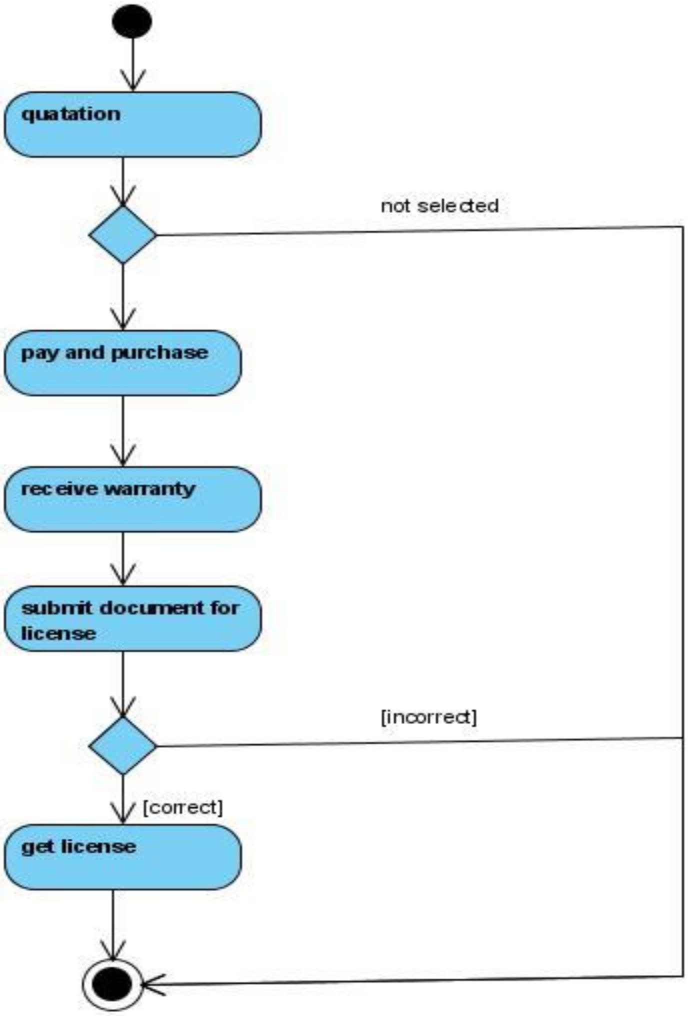


Fig 6.3 Activity diagram

Activity diagram illustrates the nature of the system. The above activity diagram describes the workflow of purchasing the vehicle where the quotation state describes the details of the models. The customer decides whether to select the vehicle or not. If he selects, he'll pay and purchase the vehicle and will be received the warranty receipts. Customer applies for the license submits the required documents. If the documents are correct then he'll get the license. Otherwise, he will be subjected to submit the correct documents.

Package diagram:

class

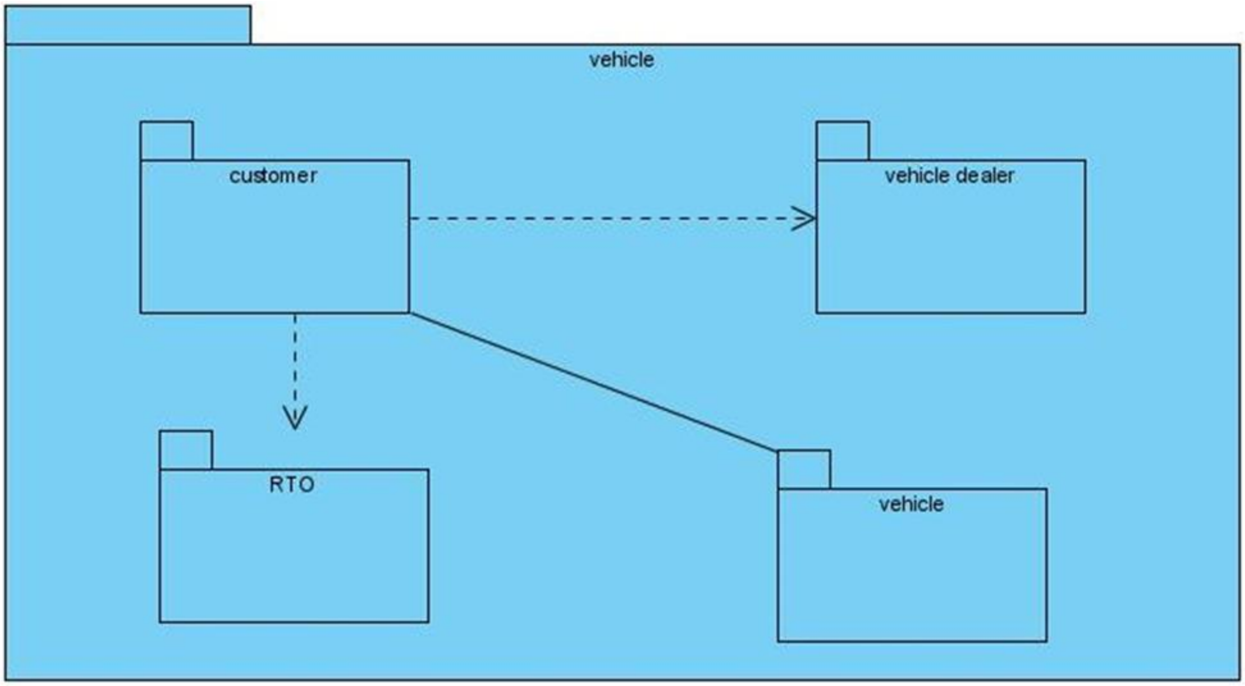


Fig 6.4 Package diagram

Package diagrams are the subset of class diagrams. The above package diagram describes the subset of classes in the vehicle purchase system. It consists of four packages: customer, vehicle dealer, RTO officer and vehicle. The customer is dependent on the vehicle dealer for getting the correct details of the vehicle. The customer issues the license from the RTO officer.

class

ASSIGNMENT 7

Develop state transition diagrams for

a) Telephone line system

b) Nested State diagram for vehicle transmission states.

State Diagram For Telephone line system:

class

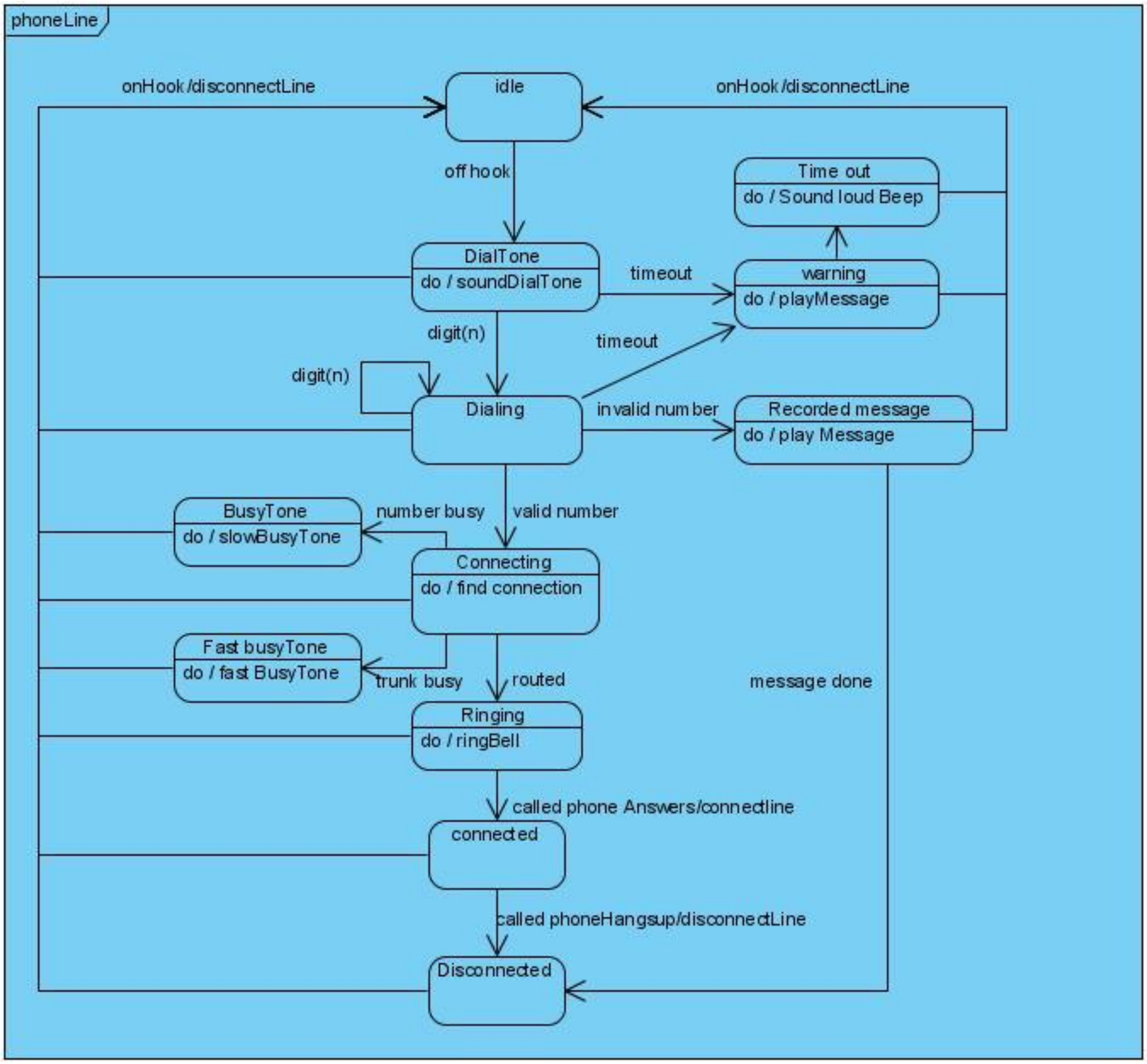


Fig 7.a Telephone line state diagram

Nested State Diagram For Vehicle Transmission States:

class

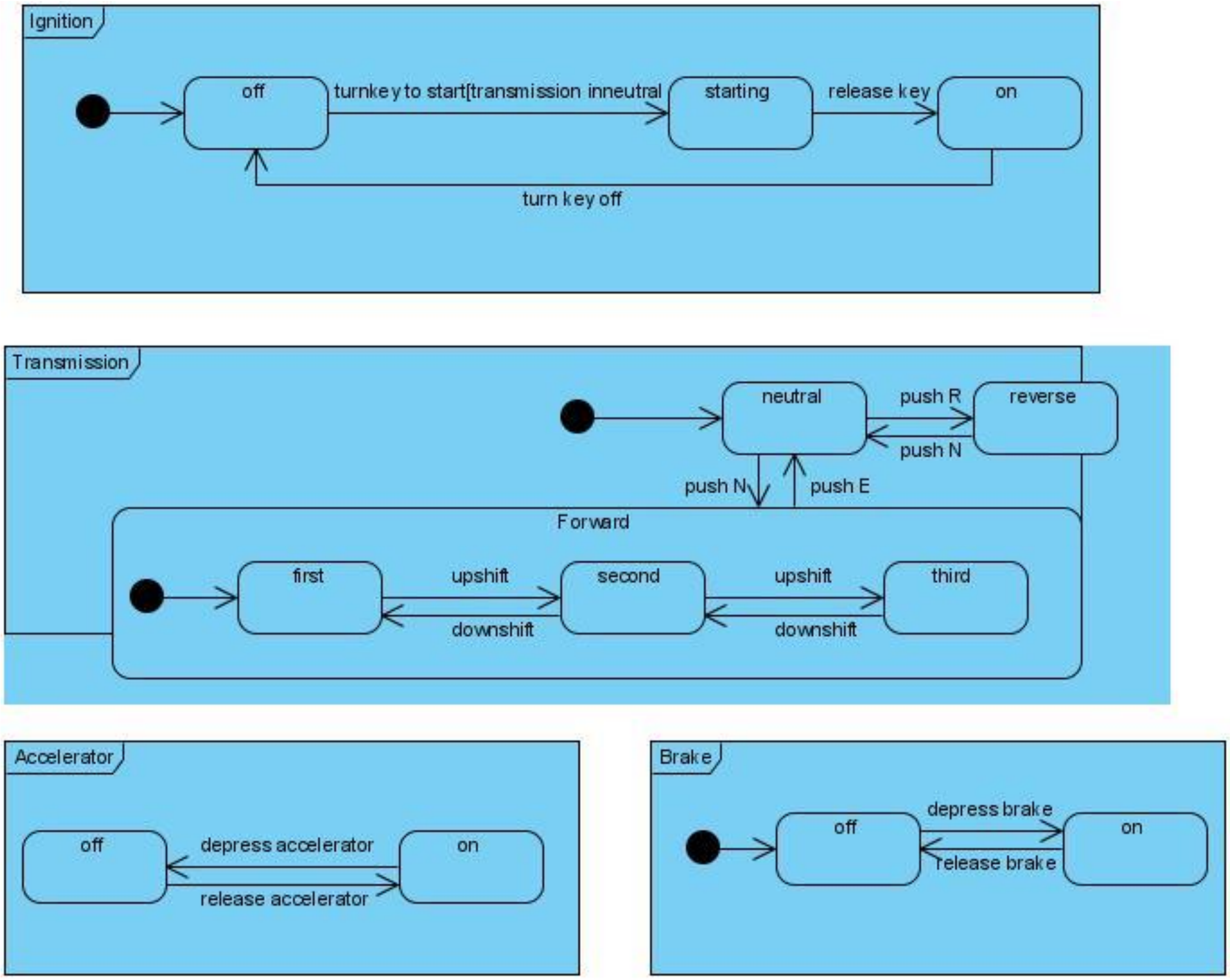


Fig 7.b Nested State Diagram For Vehicle Transmission States

class

The state of a Car is an aggregation of part states: Ignition, Transmission, Accelerator, and Brake. The state of the car includes one state from each part. Each part undergoes transitions in parallel with all the others. The state diagrams of the parts are almost, but not quite, independent-the car will not start unless the transmission is in neutral. This is shown by the guard expression Transmission in Neutral on the transition from Ignition-Off to Ignition-Starting.

DEFINITION:

A state diagram is a graph whose nodes are states and whose directed arcs are transition between the states.

EXPLAINATION:

The diagram concerns a phone line and not the caller nor callee. The diagram contains sequences associated with normal calls as well as some abnormal sequences, such as timing out while dialing or getting busy lines. The UML notations for a state diagram is a rectangle with its name in a small pentagonal tag in the upper left corner. The constituent states and transitions lie within the rectangle.

At the start of a call, the telephone line is idle. When the phone is removed from the hook, it emits a dial tone and can accept the dialing of digits. Upon entry of a valid number, the phone system tries to connect the call and route it to the proper destination. The connection can fail if the number or trunk are busy. If the connection is successful, the called phone begins ringing. If the called party answers the phone, a conversation can occur. When the called party hangs up, the phone disconnects and reverts to idle when put on hook again. The receipt of the signal onHook causes a transition from any state to Idle.

class