UNIT-II

OBJECT ORIENTED METHODOLOGIES

Contents

PREFACE:

This unit introduces the core aspects of object oriented methodologies. It deals with Patterns, Framework. This unit gives the idea about Unified Modeling Language. It deals with Object modeling technique like Rumbaugh et al' method, Jacobson et al method and Booch method.

INTRODUCTION:

Many methodologies are available to choose from for the system development. Each methodology is based on modeling the business problem and implementation in an object oriented fashion. The Rumbaugh et al method has a strong method for producing object models. Jacobson et al have a strong method for producing user-driven requirement and object oriented analysis model. Booch has a strong method for producing detailed object oriented design models.

Rumbaugh's Object Modeling Technique:

• Describes the dynamic behavior of objects in a system using the OMT dynamic model.

• Four phases.

Analysis – results are objects, dynamic and functional models.

System design – gives a structure of the basic architecture.

Object design – produces a design document.

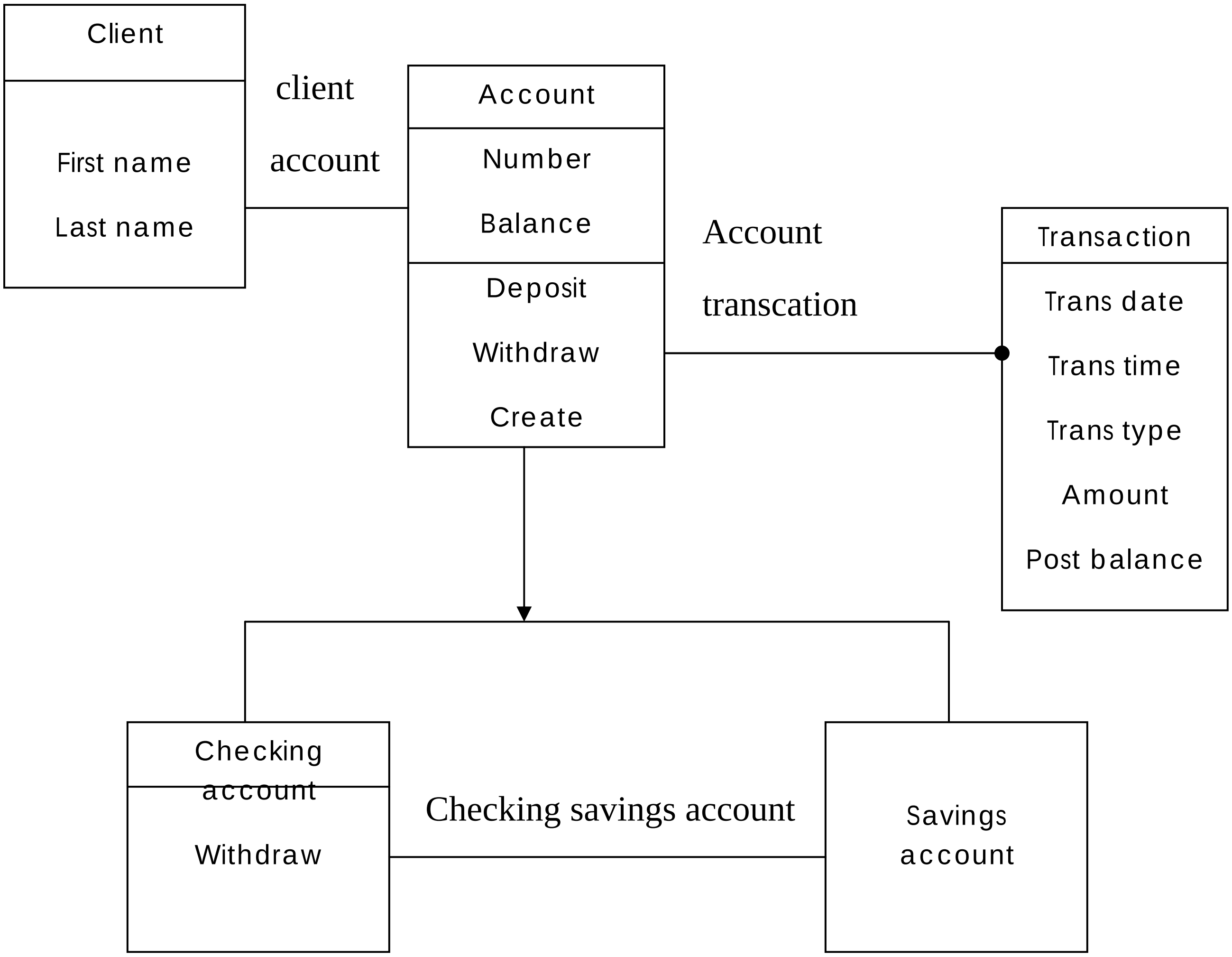
Implementation – produces reusable code.

• OMT separates modeling in to three different parts

- **Object Model –** presented by object model and the data dictionary.
- **Dynamic model -** presented by the state diagrams and event Flow diagrams.
- **Functional Model –** presented by data flow and constraints.

➢ OBJECT MODEL:-

  - Object model describes the structure of objects in a system, their identity and relationships to other objects, attributes, and operations.
  - The object model is represented graphically with an object diagram.



OMT object model of a bank system.

Boxes represent classes

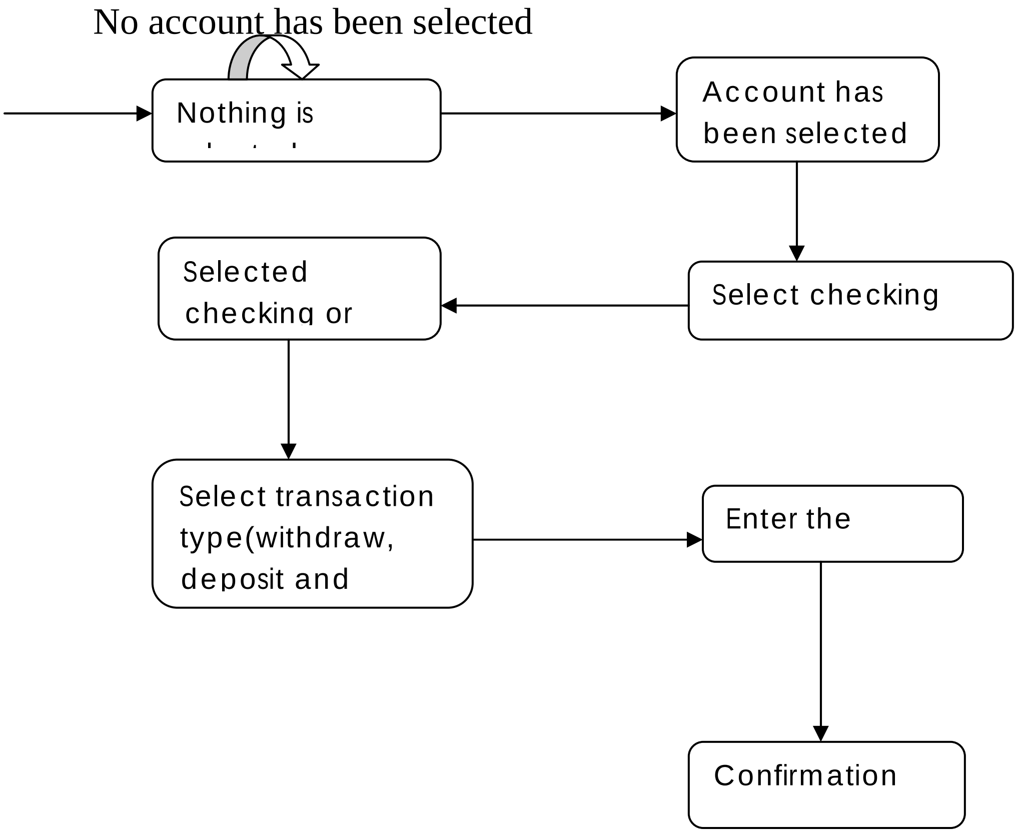
Filled represents specialization

- THE OMT DYNAMIC MODEL:

OMT provides a detailed and comprehensive dynamic model

- The OMT state transition diagram is a network of states and events.

- Each state receives one or more events, at which it makes the transition to the next state.
- The next state depends on the current state as well as the events.


No account has been selected
Nothing is
Account has been selected
Selected checking or
Select checking
Select transaction type(withdraw, deposit and
Enter the
Confirmation

State transition diagrams for the bank application user interface.

- Round boxes represents states
- Arrows represents transitions

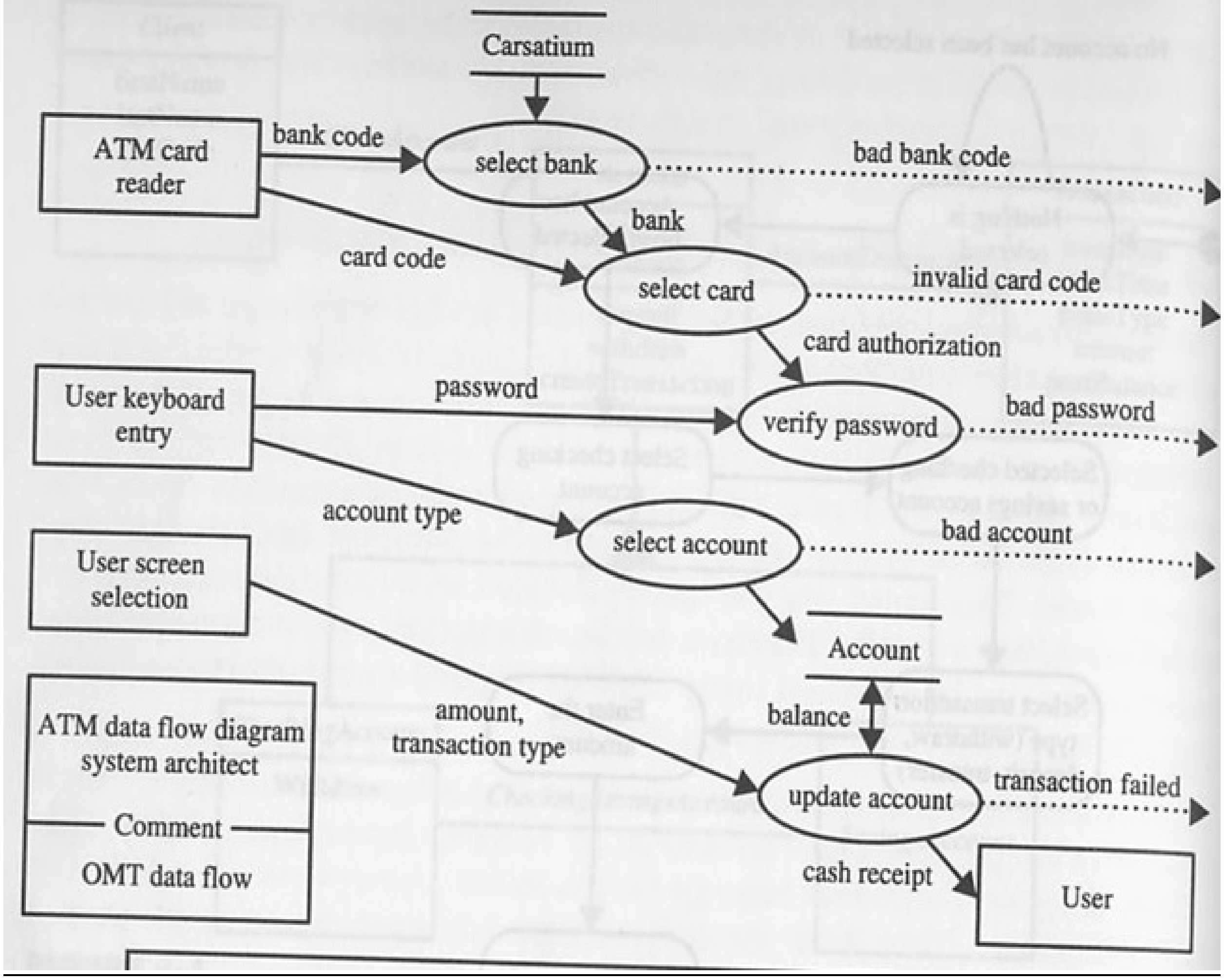
The OMT FUNCTIONAL MODEL

- The OMT data flow diagram (DFD) shows the flow of data between different processing in a business.
- Data Flow Diagrams use four primary symbols:
  - The *process* is any function being performed
  - The *data flow* shows the direction of data element movement

- The *data store* is a location where data are stored.
- An *external entity* is a source or destination of a data element.

Example:

OMT DFD model of the ATM system



Rumbaugh OMT methodology provides one of the strong tolls set for the analysis and design of object-oriented system.

THE BOOCH METHODOLOGY :

•Booch methodology is a widely used object-oriented method that helps to design your system using the object paradigm.

• Is criticized for his large set of symbols.

•It consists of the following diagrams:

_ Class diagrams.

_ Object diagrams.

_ State transition diagrams.

_ Module diagrams.

_ Process diagrams.

_ Interaction diagrams.

• Two processes:

_ Macro development process

_ Micro development process.

• Macro development process.

Primary concern – technical management of the system.

Steps involved:

❖ Conceptualization.

Establish the core requirements and develop a prototype.

❖ Analysis and development of the model

Use the class diagram to describe the roles and responsibilities of objects. Use the object diagram to describe the desired behavior of the system.

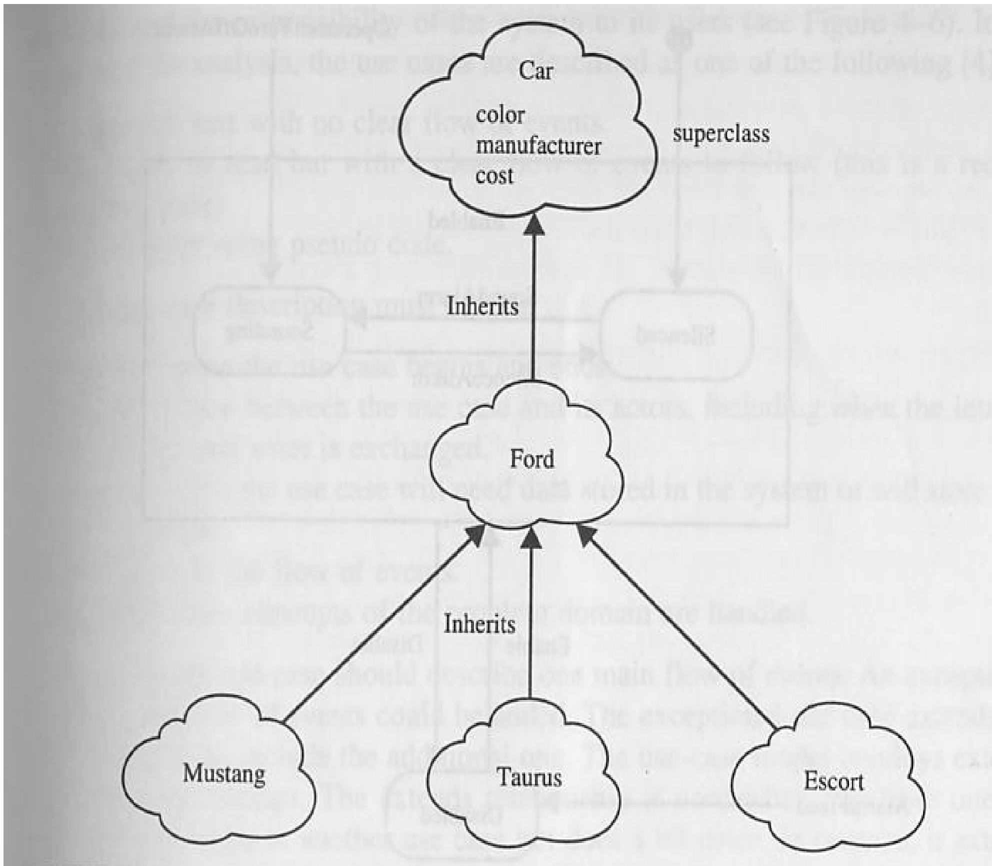


FIGURE 4–4
Object modeling using Booch notation. The arrows represent specialization; for example, the class Taurus is subclass of the class Ford.

- Design or create the system architecture.

  Use the class diagram to decide what classes exist and how they relate to each other, the object diagram to decide what mechanisms are used, the module diagram to map out where each class and object should be declared, and the process diagram to determine to which processor to allocate a process.

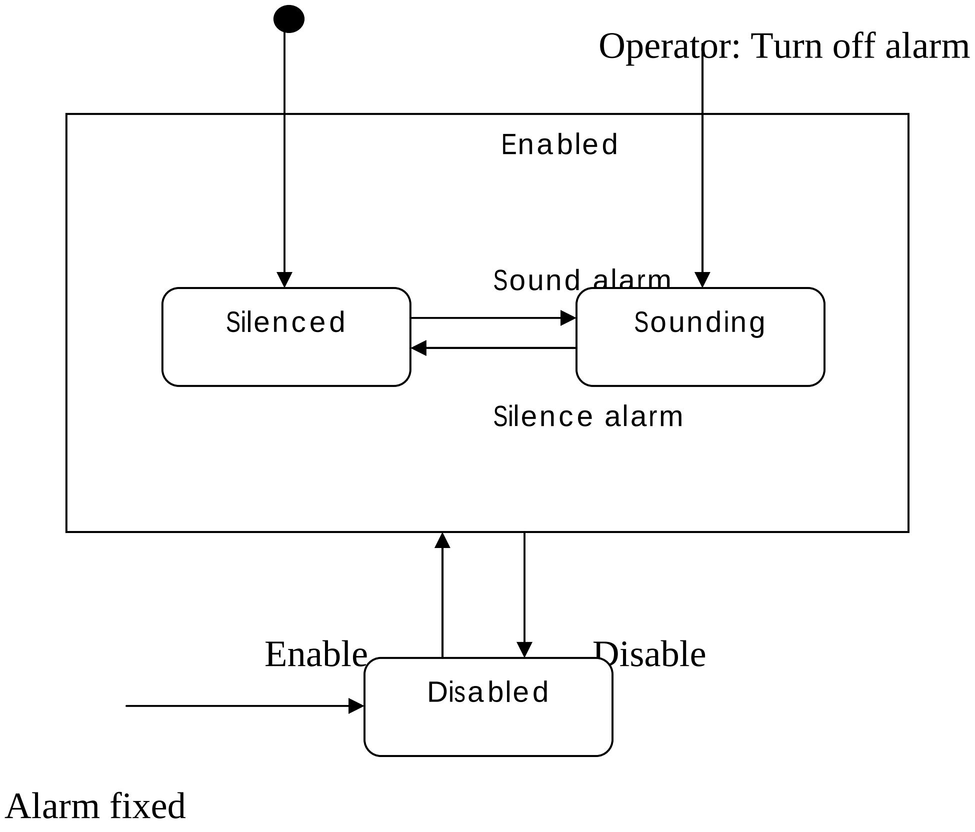
- Evolution or implementation.

  Refine the system through much iteration.

- Maintenance.

Make localized changes to the system to add new requirements and eliminate bugs.

- **Micro development process.**
  - The micro development process is a description of the day-to-day activities.
  - Steps involved:
    - Identify classes and objects.
    - Identify classes and object semantics.
    - Identify classes and object relationships.
    - Identify classes and object interfaces and implementation.

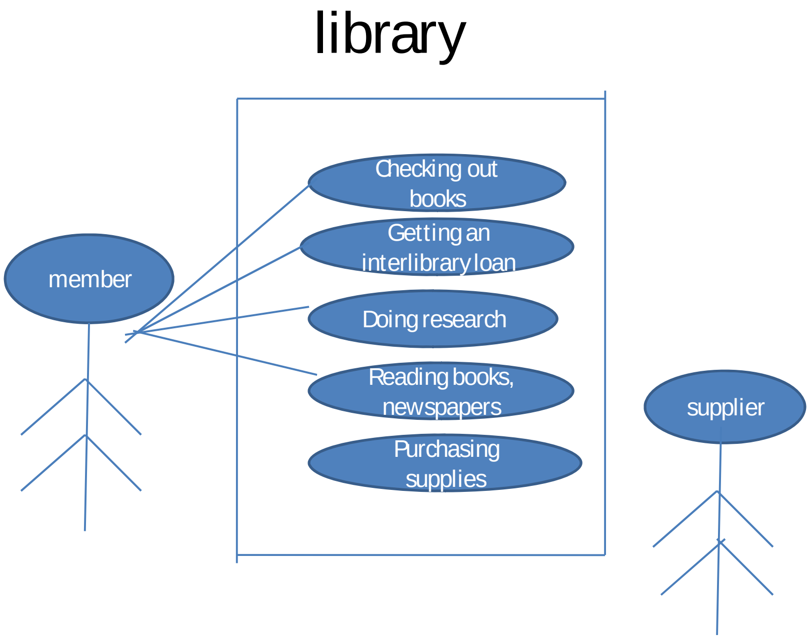


An alarm state transition diagram with Booch notation

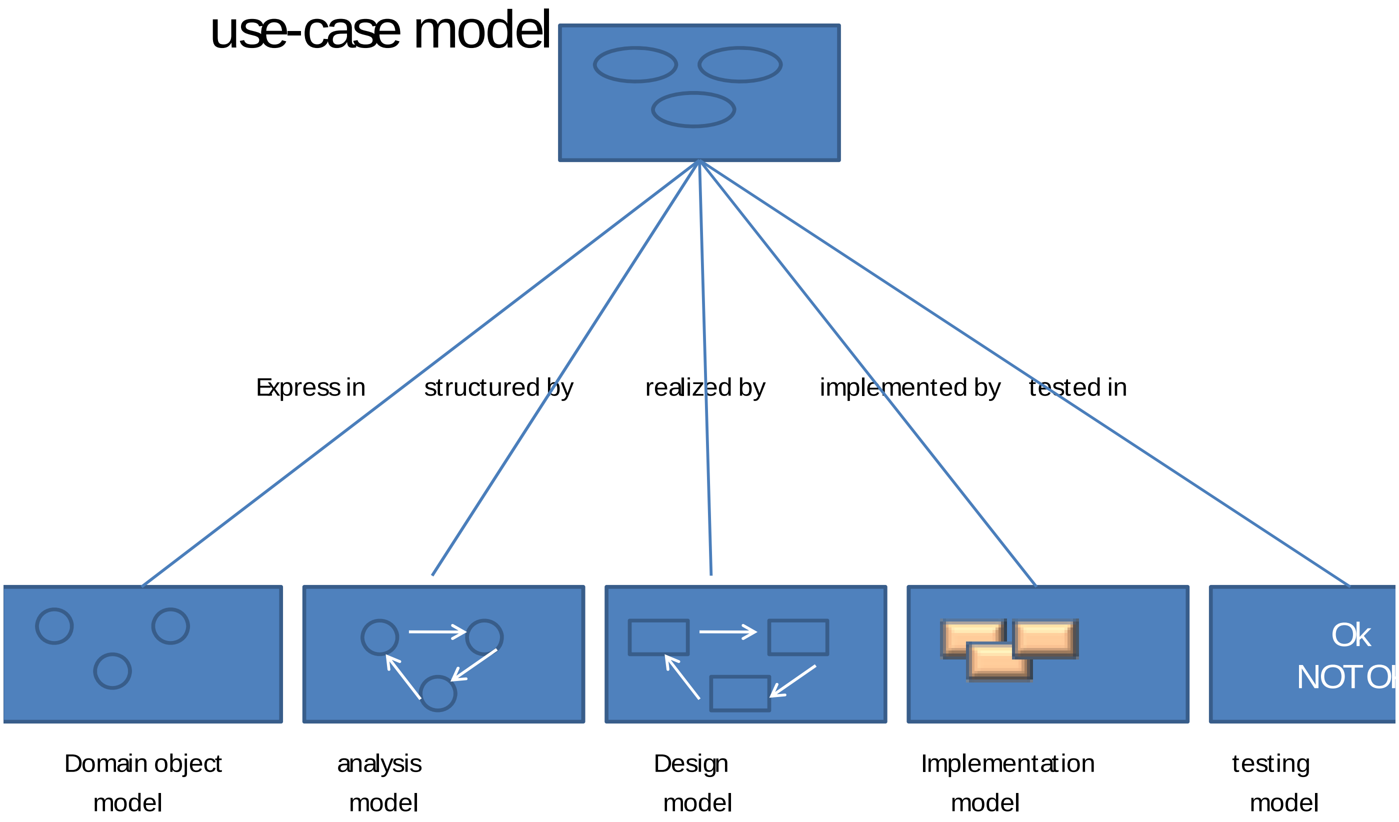
JACOBSON METHODOLOGY:

- Object-oriented business engineering (OOBE)
- Object-oriented software engineering (OOSE)
  - It covers the entire life cycle
  - Stress traceability(enables reuse of analysis and design work) both forward and backward

- Use cases
  - ✓ Scenarios for understanding system requirements.
  - ✓ Non formal text with no clear flow of events.
  - ✓ Text easy to read.
  - ✓ Formal style using pseudo code.
  - ✓ Can be viewed as concrete or abstract (not initiated by actors).
    - Understanding system requirements
    - Interaction between user and system
    - It captures the goal of the user and responsibility of the system to its users



- Object oriented software Engineering: Objectory
  - OOSE is also called Objectory
  - Development process is also called as use case driven development
  - The system development method based on OOSE is a process for the industrialized development of the s/w



- Use case model:-

The use-case model defines the outside and inside of the system behaviour

- Domain Object Model:-

The objects of the real worls are mapped in to the domain object model

- Analysis Object Model:-

The analysis object model presents how the source code should be carried out written

Implementation model:-

The implementation model represents the implementation of the system

- Test model:-

  The test model constitutes the test plans, specification and reports.

• Object oriented business Engineering

- Analysis phase:-
  - The analysis phase defines the system to be built in terms of the
    - problem-domain object model
    - the requirements model
    - analysis model
- Design and Implementation phase:-
  - The implementation environment must be identified for the design model.
- Testing phase:-
  - This level includes unit testing, integration testing and system testing.

PATTERNS AND THE VARIOUS PATTERN TEMPLATES:

➢ Pattern –

- Identifies a common structure that makes it useful for design.
- provide common vocabulary
- provide "shorthand" for effectively communicating complex principles
- help document software architecture
- capture essential parts of a design in compact form
- show more than one solution

    - describe software abstractions

- Patterns do not...

- provide an exact solution
- solve all design problems
- only apply for object-oriented design

- Involves a general description of a solution to a recurring problem.
- Properties of a good pattern:
    - It solves a problem:-

      Patterns capture solutions not just abstract principles or strategies
    - It is a proven concept:-

      Patterns capture solutions with a track record, not theory or speculation
    - Solution is not obvious:-

      The best patterns generate a solution to a problem indirectly
    - Describes a relationship:-

      Patterns do not just describes modules but describes deeper system structures and mechanisms
    - Has significant human component:-

      All software serves human comfort or quality of life
- Generative patterns – Tells us how to generate something.
    - observed in a system
    - descriptive and passive
- Non generative patterns –
    - generate systems or pats of systems
    - perspective and active

- Patterns templates.
    - Name:-
        - A meaningful name allows us to use a single word or short phrase to refer to the pattern and the knowledge and structure it describes.
        - Some pattern forms also provide a classification of the pattern in addition to its name.
    - Problem:-
        - A statement of the problem that describes its intent: the goals and objectives it wants to reach within the given context and forces.
        - The forces oppose these objectives as well as each other.
    - Context:-
        - The preconditions under which the problem and its solution seem to recur and for which the solution is desirable.
        - It can be thought of as the initial configuration of the system before the pattern is applied to it.
    - Forces:-
        - A description of the relevant forces and constraints and how they interact or with one another and with the goals that the user wish to achieve.
        - A concrete scenario that serves as the motivation for the pattern frequently is employed.
    - Solution:-
        - Static relationships and dynamic rules describing how to realize the desired outcome.

    - It describes how to construct the necessary products.
    - It encompasses the pictures, diagrams and prose that identify the pattern structure, and their participants and collaborations to show how the problem is solved.
- Examples:-
    - One or more sample applications of the pattern that illustrate a specific initial context; how the pattern is applied to and transforms that context.
- Resulting context:-
    - The state or configuration of the system after the pattern has been applied, including the consequences of applying the pattern and other problems and patterns that may arise from the new context.
    - Rationale:-
    - Steps or rules in the pattern and also of the pattern as a whole in terms of how and why it resolves it forces in a particular way to be in alignment with desired goals.
- Related patterns:-
    - The static and dynamic relationships between this pattern and others within the same pattern language or system.
    - Related pattern often share common forces.
    - They also frequently have an initial or resulting context that is compatible with the resulting or initial context of another pattern.
- Known uses:-
    - The known occurrences of the pattern and its application within existing systems.

- This helps to validate a pattern by verifying that it indeed is a proven solution to the recurring problem.

FRAMEWORKS:

Frameworks are a way of delivering application development patterns to support best practice sharing during application development .

A frame work is a way of presenting a generic solution to a problem that can be applied to all levels in a development. Several design patterns in fact a framework can be viewed as the implementation of a system of design patterns.

The major differences between design patterns and frameworks as follows

- Design patterns are more abstract than frameworks.
- Design patterns are smaller architectural elements than frameworks.
- Design patterns are less specialized than frameworks.

THE UNIFIED APPROACH:

- Establishes a unifying and unitary framework by utilizing UML.
- The processes are:
    - Use case driven development.
    - Object oriented analysis.
    - Object oriented design.
    - Incremental development and prototyping
    - Continuous testing.

- Methods and technologies employed include:
    - UML:- Unified Modeling approach is used for modeling
    - Layered approach:-
    - Repository:- Repository for object-oriented system development patterns and frameworks
    - CBD:- Component based development
        - The Unified Approach allows iterative development by allowing to go back and forth between the design and the modeling or analysis phase.
        - It makes backtracking very easy and departs from the linear waterfall process, which allows no form of backtracking.
- Object oriented Analysis.
    - Identify actors.
    - Develop a simple process model.
    - Develop the use case.

  - Develop interaction diagrams.
  - Identify classes.
- Object oriented design
  - Design classes, attributes, methods etc.
  - Design the access layer.
  - Design the prototype user interface.
  - User satisfaction and usability tests.
  - Iterate and refine the design.
- Continuous testing
- UML – modeling language
- Repository
  - Allows maximum reuse of previous experience.
  - Should be accessible by many people.
- Layered approach
  - The business layer.
    - Displaying results.
    - Data access details
- The user interface or view layer.
    - Responding to user interaction:-
      - It must be designed to translate actions by the user, such as clicking on a button or selecting from a menu.
    - Displaying business objects:-
      - This layer must paint a best possible picture of the business objects for the user.

- The access layer.
    - Translate request:-
        - The access layer must be able to translate any data-related requests from the business layer in to the appropriate protocol for data access.
    - Translate results:-
        - The access layer also must be able to translate the data retrieved back into the appropriate business objects and pass those objects back up into the business layer.

UML:

UML stands for Unified Modeling Language. This object-oriented system of notation has evolved from the work of Grady Booch, James Rumbaugh, Ivar Jacobson, and the Rational Software Corporation. These renowned computer scientists fused their respective technologies into a single, standardized model. Today, UML is accepted by the Object Management Group (OMG) as the standard for modeling object oriented programs.

UML survival introduces the Unified Modeling Language and leads you through an orderly progress towards mastery of the language. You'll begin by learning how UML is used to model the structure of a system. Many key UML concepts, especially that of the general (classes) versus the specific (objects), are illustrated in the chapter on class and object diagrams.

Model:

A Model is an abstract representation of a system Constructed to understand the system prior to building or modifying it.

Modeling:

- Building a model for a software system prior to its construction is as essential as having a blue print for building a large building.
- Good models are essential for communication among project teams.
- It has

1. Model elements → fundamental modeling concepts semantics
2. Notation→ Visual rendering of model elements.
3. Guidelines→Expression of usage with in the trade.

- This visual notation provides benefits related to clarity , familiarity and maintenance.

Clarity: Much better at picking out errors.

Familiarity: Similar to the way in which information is actually stored.

Maintenance: Visual notation can improve the maintainability of a system.

Advantages:

1. Easier to express complex ideas
2. Reduction of complexity.
3. Cost is lower.
4. Manipulation is easier.

Static model:

It Can be viewed as a snapshot of a system's parameters at rest or at a specific point in time Ex: Class diagram

Dynamic model

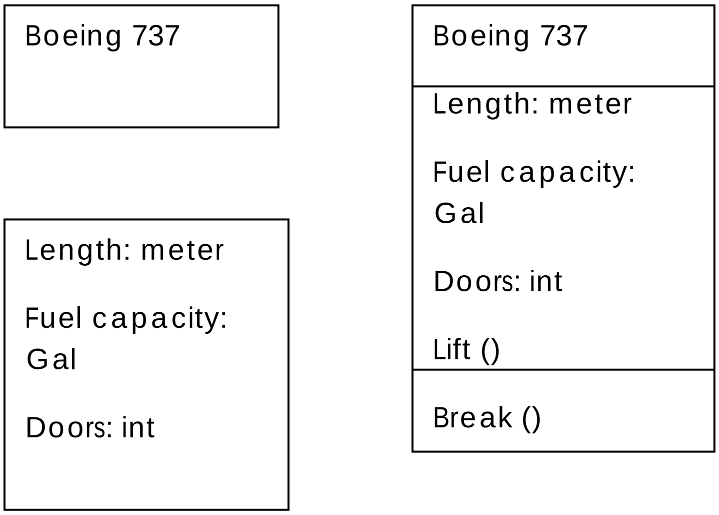
It can be viewed as a collection of procedures or behaviors that taken together reflect the behavior of a system over time ex: Interaction diagram

UML Class diagram:

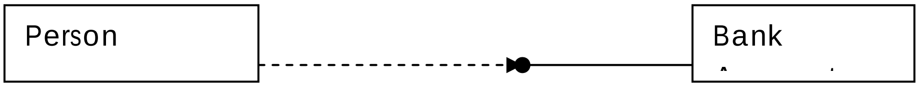
- UML Class Diagram is also referred to as Object modeling.
- Collection of static modeling elements.
- Objects will form the system , because in the object-oriented viewpoint, objects are the primary abstraction.

- Each object in the problem domain describe the structure and the relationship among objects.

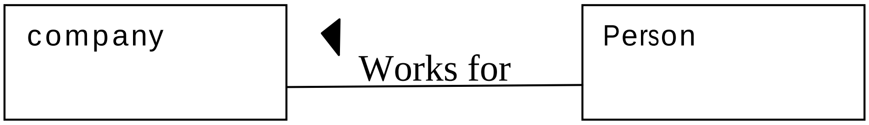
- **Class notation** – rectangle with three compartments.

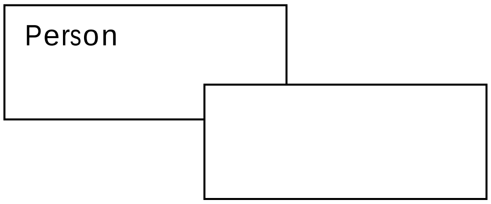


- **Object diagram** – instance of a class diagram.
- **Class interface notation** – small circle with the interface name connected to the class.



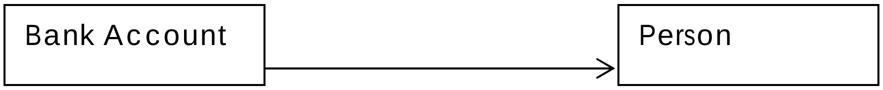
- **Binary association** – solid path connecting two classes.

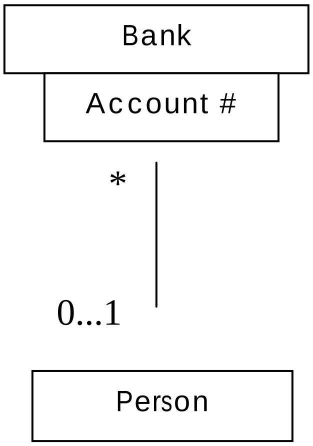




◀ Married to

- **Association role** – end of an association.



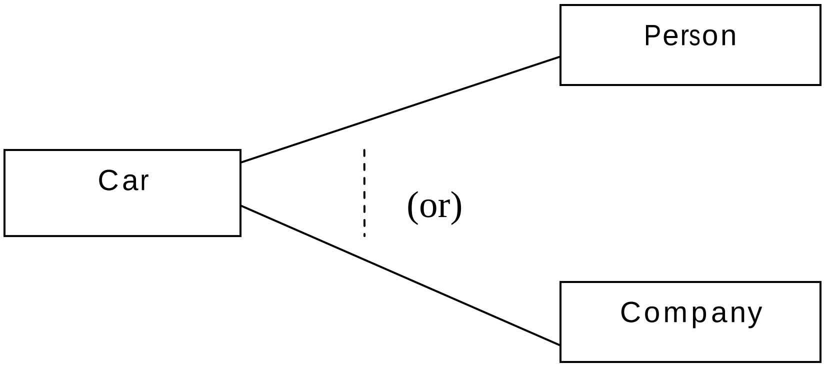
- 

- **Qualifier –** small rectangle attached to the end of an association.
- **Multiplicity –** specifies the range of allowable associated classes.

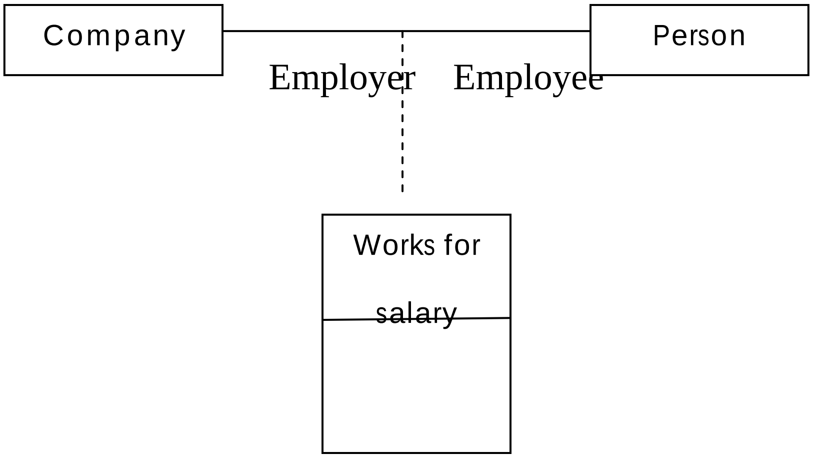
  It is given for roles within associations. Sequence of integer intervals, where an interval represents a range of integers in this format. 0…1(lower bound …..upper bound)

  0…*

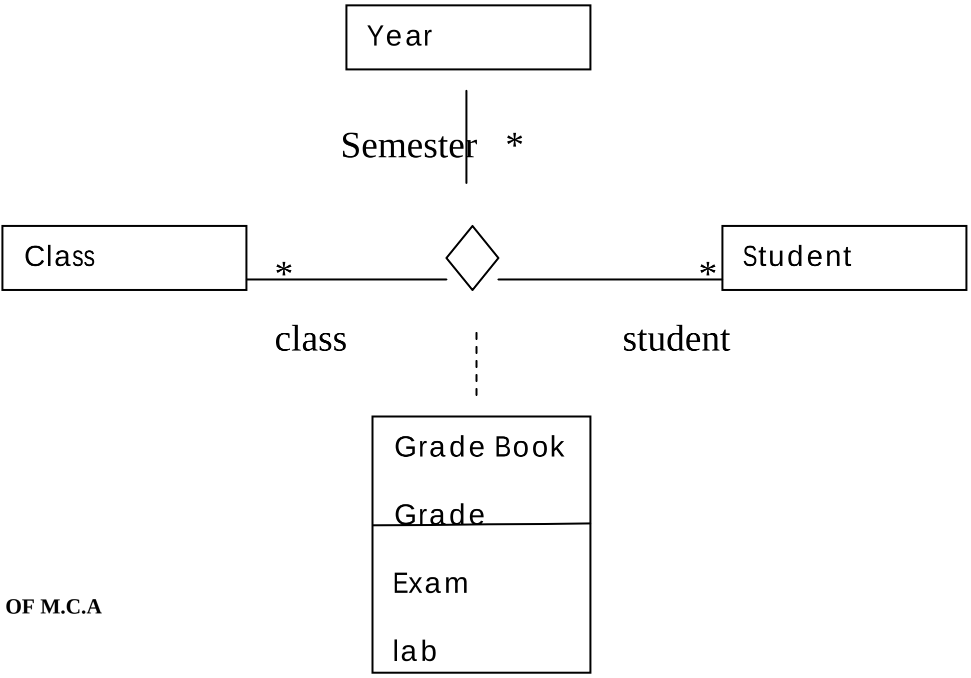
  1..3, 7…10, 15, 19…*
- **OR association –** dashed line connecting two or more associations.



- **Association class –** association that has class properties.



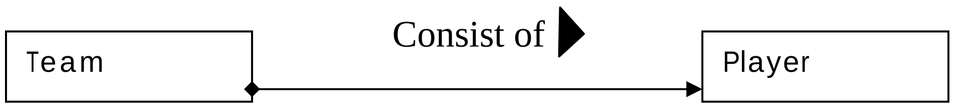
- **N-ary association –** association among more than two classes.

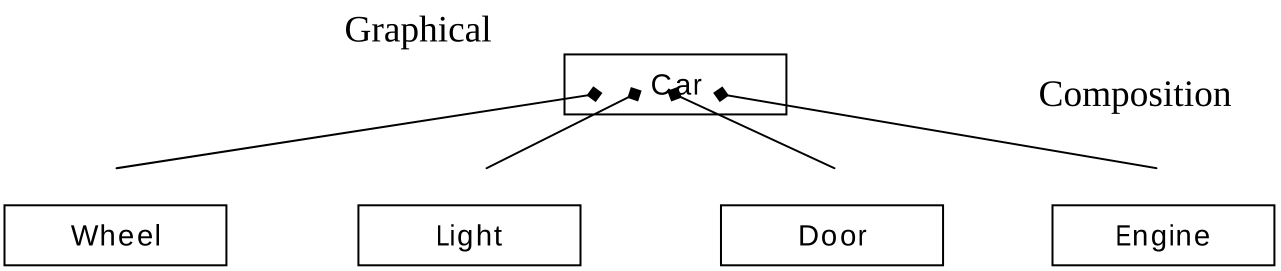


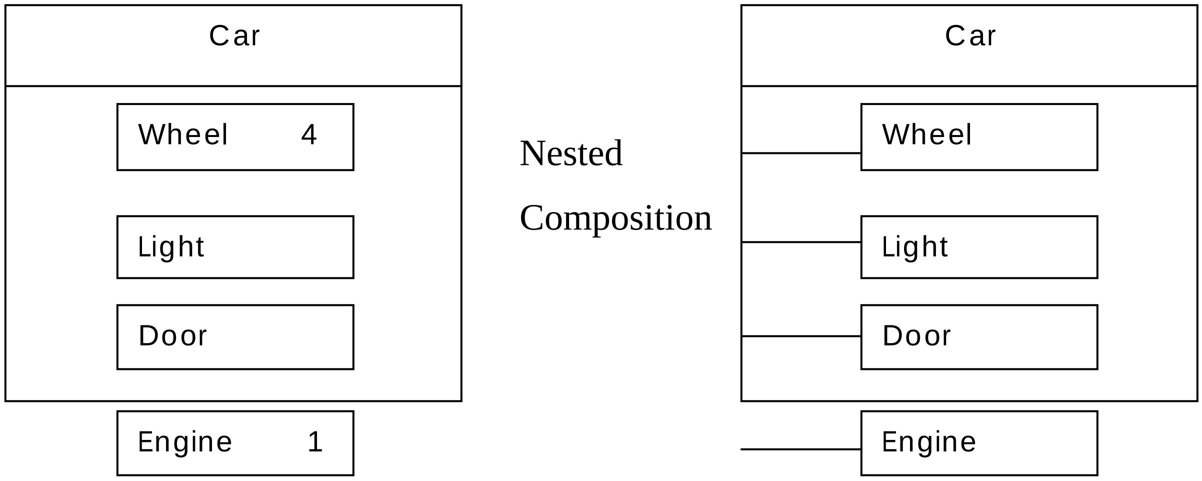
- **Aggregation** and composition– hollow diamond attached to the end of the path.

  Aggregation is a form of association.

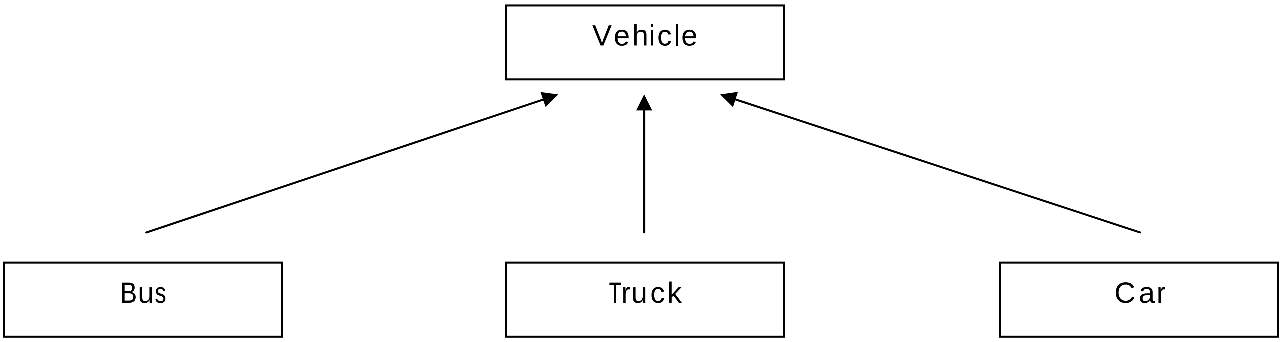
  Composition also known as the *a-part-of* is a form of aggregation with strong ownership to represent the component of a complex object.

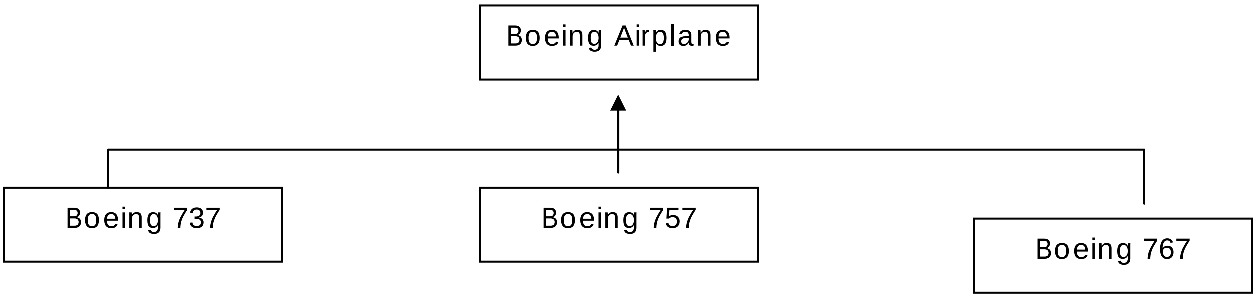






- **Generalization –** relationship between a general class and a more specific class.





UML Dynamic Modeling:

- UML INTERACTION DIAGRAMS

Interactions diagram describes how group of objects collaborate to get the job done.

It captures the behavior of the single use case showing the pattern of interaction among objects.

o UML Sequence Diagrams

- Sequence diagrams illustrate how objects interact with each other.

• They focus on message sequences, that is, how messages are sent and received between a number of objects.

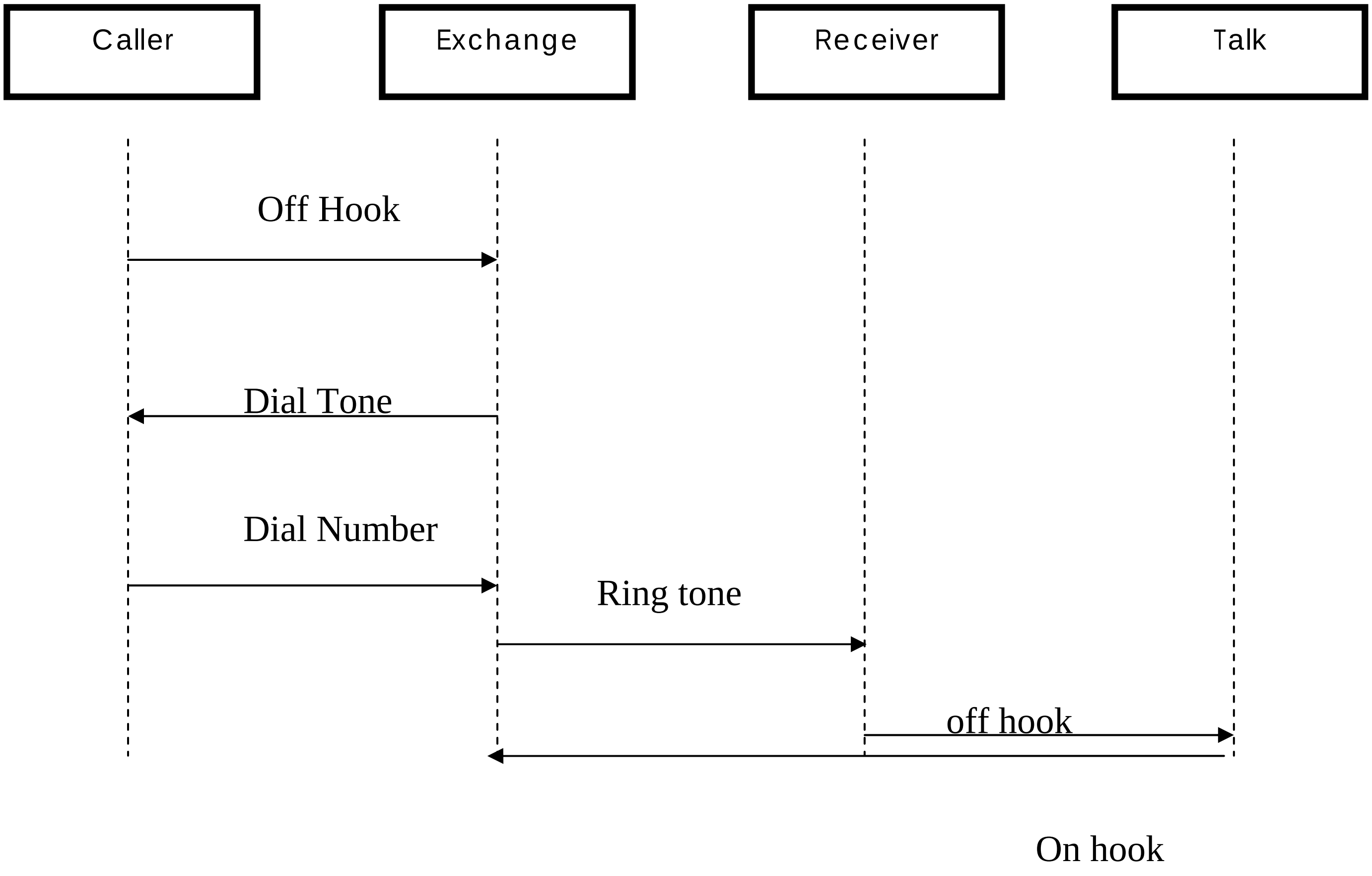
• Sequence diagrams have two axes: the vertical axis shows time and the horizontal axis shows a set of objects.

• The Instance form describes a specific scenario in detail

• The Generic form describes all possible alternatives in a scenario, therefore b ranches, conditions, and loops.

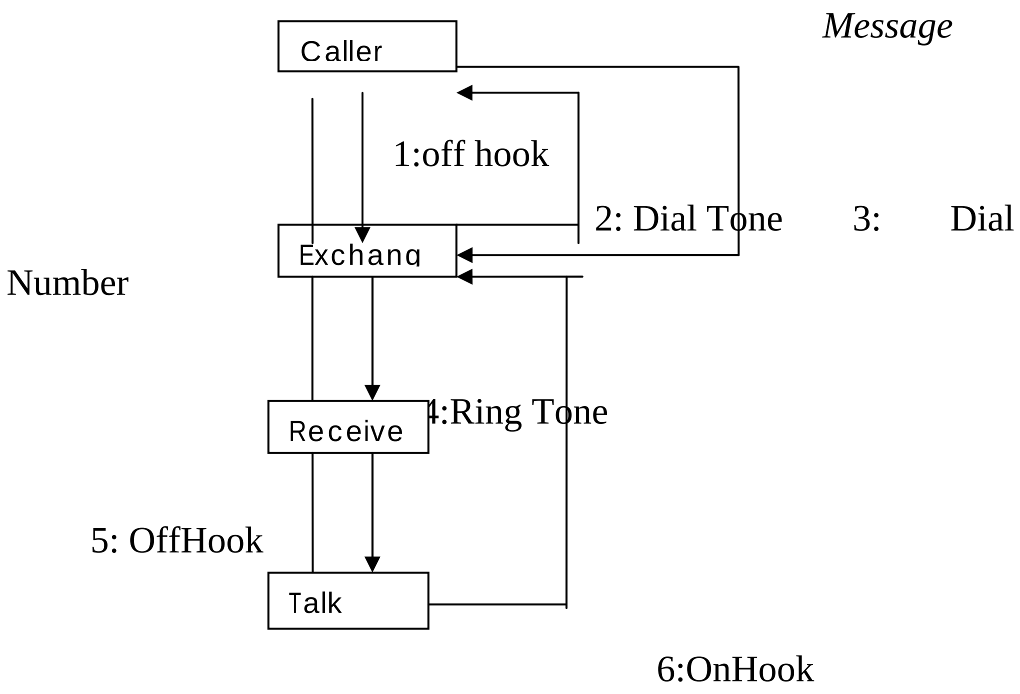
An example of a sequence diagram

Telephone call:



UML COLLABORATION DIAGRAM

Collaboration diagrams focus on the interaction and the line between a set of collaborating objects. The sequence diagram focuses on time but the collaboration diagram focuses on space.



UML STATE CHART DIAGRAM

A State chart diagram shows the sequence of states that an object goes through during its life in response to outside and messages.

The state is a set of values that describes an object at a specific point in time and is represented by state symbols and the transitions are represented by rows connecting the state symbols.

A state chart diagram may contain sub diagrams.

A state diagram represents the state of the method execution and activities in the diagram represent the activities of the object that performs the method.

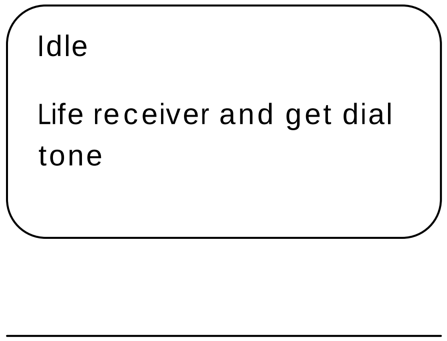
The purpose of the state diagram is to understand the algorithm involved in performing the method.

To complete the OOD, the activities within the diagram must be assigned to objects and the control flows assigned to links in the object diagram.

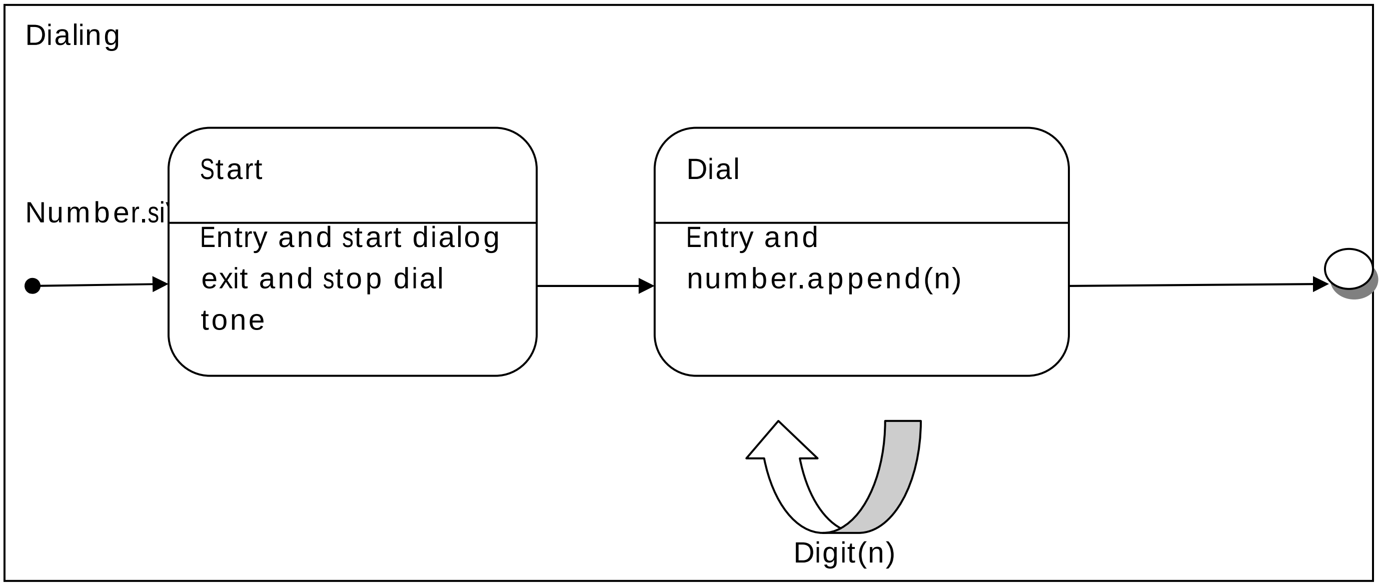
Example:

A simple state idle and a nested state.

The dialing state contains sub states, which consist of start and dial states.



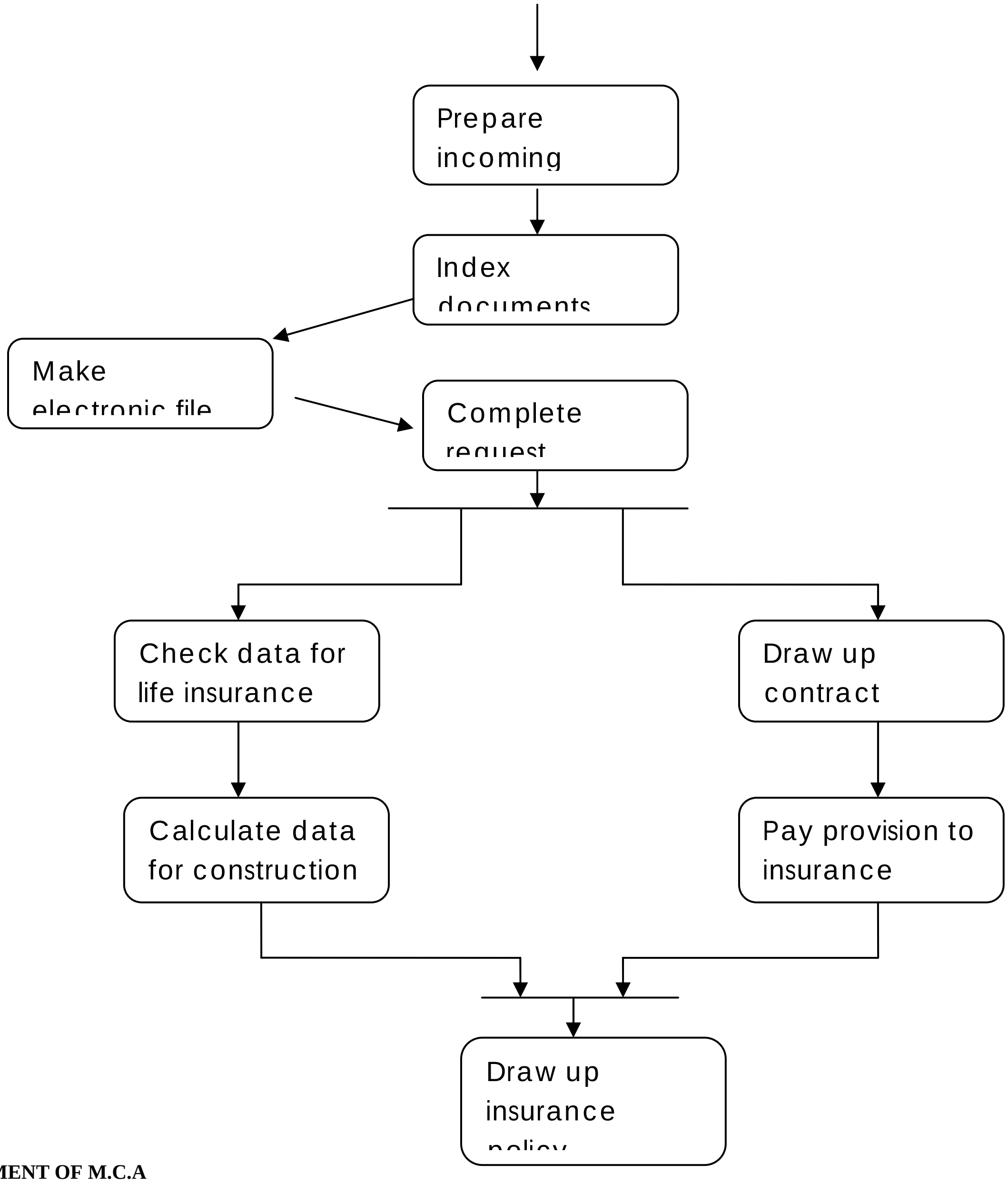
State


Dialing
Number.si
Start
Entry and start dialog exit and stop dial tone
Dial
Entry and number.append(n)
Digit(n)

UML ACTIVITY DIAGRAM:

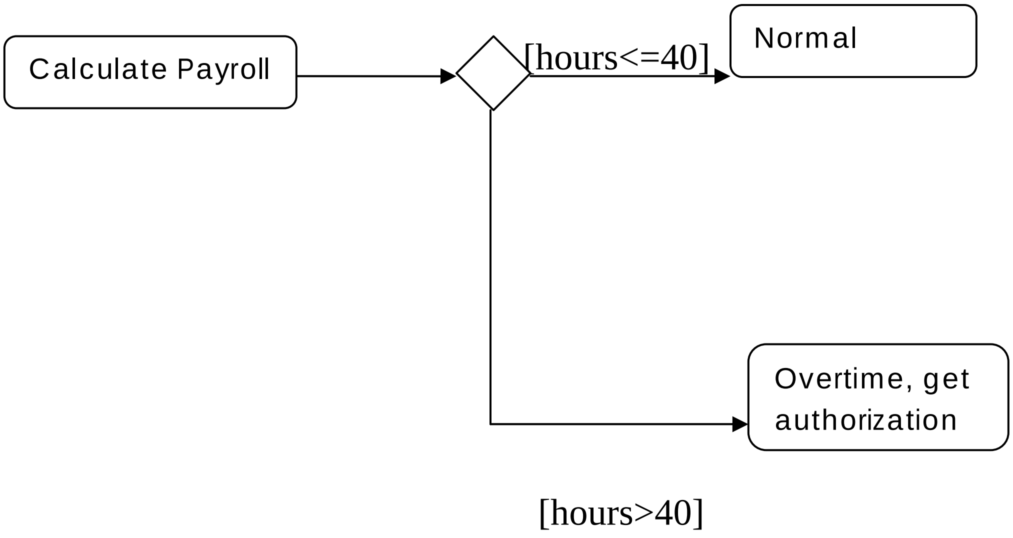
An activity diagram is a variation or special case of a state machine, in which the states are activities representing the performance of operations and the transitions are triggered by the completion of the operations.

An activity diagram for processing mortgage request



An activity diagram is used mostly to show the internal state of an object, but external events may appear in them.

An external event appears when the object is in a "wait state" a state during which there is no internal activity by the object and the object is waiting of some external event to occur as a result of an activity of an another object.



Activity and state diagrams express a decision when conditions are used to indicate different possible transitions that depend on Boolean conditions of container object.

Key Terms:

Object design

Analysis object model

class diagram

Object oriented software engineering

Booch Methodology

Object model

Object Oriented Business Engineering

State transition

Use Case Model

Questions:

1. __________ phase produces a design document, consisting of detailed objects static, dynamic, and functional models.

2. ___________ model describes the structure of objects in a system.

3. The OMT ________________ diagram is a network of states and events.

4. Object-oriented method that helps you design your system using the object paradigm is ____________________.

5. OOBE is_________________.

6. OOSE is__________________.

7. The ________________defines the outside (actors) and inside (use case) of the System's behavior.

8. ________________presents how the source code (implementation) should be carried out and written.

9. A ______________ describes the types of objects in the system and the various kinds of static relationships that exist among them

Test your Understanding

Objective Questions

1. Which is a method of object oriented development with the specific aim to fit the development of large, real time systems?

   a. Rambaugh b. Booch C. Objectory d. Jacobson

2. __________ is instructive information that captures the essential structure and insight of a successful family of proven solutions to a recurring problem that arises within a certain context and system of forces.

   a. Frame work b. Patterns c. Anti pattern d. Objectory

3. Which patterns that, not only describe a recurring problem, they can tell us how to generate something and can be observed in the resulting system architectures they helped shape.

   a. Anti pattern b. Framework c. Generative Pattern

   d. non generative pattern

4. Every pattern must be expressed in the form of a rule called

   a. Pattern b. Framework c. Pattern Template d. none

5. A pattern represents a best practice is

   a. Anti pattern b. Pattern c. Pattern template d. Generative pattern

6. The way of delivering application development patterns to support best practice sharing during application development is

    a. Pattern b. Framework c. Anti pattern d. Template

7. OMG stands for

    a. Object Modeling Group b. Object Management group c. none

8. It can be viewed as a collection of procedures or behaviors that taken together reflect the behavior of a system over time.

    a. Static Model b. Dynamic model c. Functional model d. none

9. In UML , Solid line that represents that one entity uses another entity as part of its behavior. This relationship is called

    a. Dependency b. Generalization c. none d. Association

10. Sequence diagram describes the behavior of a system viewing the interaction between the system, Vertical line represents

    a. Object b. Class c. Time d. none

11. Method of object-oriented development with the specific aim to fit the development of large, real-time systems is

    a. Objectory b. OOSE c. Both d. None

12. _________ is a familiar technique to describe the behavior of a system

    a. Object Diagram b. Class Diagram c. State diagram d. none

Part – A (2 MARKS)

1. What is OMT?

2. What are the phases of OMT?

3. Compare functional and dynamic model.

4. Differentiate between Pattern and Frameworks.

5. Why is unified approach needed? List its components.

6. Why do we want to model the problem?

7. What do you mean by object diagram?

8. What are the primary symbols used in Data Flow Diagrams?

9. Define Use Cases.

10. What are the diagrams used in Booch methodology?

11.What are the steps involved in Macro development process in Booch methodology.

12. Write short note on Objectory.

13.What are the various models of objectory?

14. What is an association class?

15. Define proto-patterns.

16. Define pattern template? What are the components in pattern template?

17. Define anti-patterns.

18. Define pattern mining. Give the steps involved in capturing pattern.

19. Define frame work.

20. Write short note on UA proposed Repository.

21. Define model. Explain about the types of model. .

22. What are the advantages of Modeling?

23. Define UML.

24. Mention the primary goals in the design of the UML.

25. Give the nine UML graphical diagrams.

26.How interfaces are represented in UML?

PART-B (16 MARKS)

1. Explain Rumbaugh’s Object Modeling Technique?

2. Explain the basic concepts of Unified Modeling Language.

3. Compare and Contrast the Booch and Jacobson methodology.

4. Describe patterns and the various pattern templates?

5. a) Explain in detail about the Unified approach? (8)

   b) Describe the UML Class diagram? (8)

6. Explain the Class diagram and Use case diagram for Railway Reservation System

7. Explain the Interaction diagrams with an example.

2 MARKS (Questions and Answers)

1. Write about the four phases in OMT?

OMT consists of four phases. They are

• Analysis-The results are objects and dynamic & functional models.

• System design-The results are a structure of the basic architecture of the system along with high-level strategy decisions.

• Object Design-Produces a design document, consisting of detailed objects static, dynamic and functional models

• Implementation-This activity produces reusable, extendible, robust code.

2. What do you mean by object diagram?

- The object model of OMT is represented graphically with an object diagram.
- The object diagram contains classes interconnected by association lines. Each class represents a set of individual objects.
- The association lines establish relationships among the classes. Each association line represents a set of links from the objects o f one class to the objects of another class

3. What are the primary symbols used in Data Flow Diagrams?

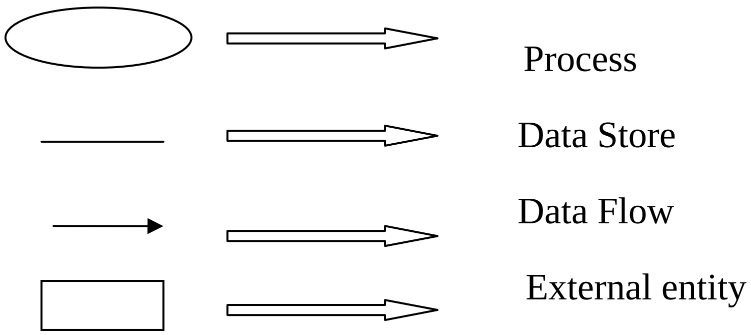
Data flow diagrams use four primary symbols:

The **process** is any function being performed.(verify password in the ATM)

The **data flow** shows the direction of data element movement.(PIN code)

The **data store** is a location where data are stored.(Account in ATM)

An **external entity** is a source or destination of a data element .(the ATM card reader)



4. What are the diagrams used in Booch methodology?

The Booch methodology consists of the following diagrams:

- Class diagrams.
- Object diagrams.
- State transition diagrams.
- Module diagrams.
- Process diagrams.
- Interaction diagrams.

5. Give the steps involved in Macro development process in Booch methodology.

The macro development process consists of the following steps:

o Conceptualization

Establish the core requirements and develop a prototype.

o Analysis and development of the model

Use the class diagram to describe the roles and responsibilities of objects. Use the object diagram to describe the desired behavior of the system.

o Design or create the system architecture.

Use the class diagram to decide what classes exist and how they relate to each other, the object diagram to decide what mechanisms are used, the module diagram to map out where each class and object should be declared, and the process diagram to determine to

which processor to allocate a process.

o Evolution or implementation- Refine the system through much iteration.

o Maintenance-

Make localized changes to the system to add new requirements and eliminate bugs.

6. Give the steps involved in Micro development process in Booch methodology.

The micro process is a description of the day-to-day activities by a single or small group of software developers. It consists of the following steps.

- Identify classes and objects.
- Identify class and object semantics.
- Identify class and object relationships.
- Identify class and object interface and implementation.

7. Write briefly about Use Cases.

Use cases are scenarios for understanding system requirements. A use case is an interaction between users and a system. The use-case model captures the goal of the user and the responsibility of the system to its users. The use-case model employs extends and uses relationships. The use cases are described as one of the following:

- Non-formal text with no clear flow of events
- Text with a clear flow of events
- Formal style using pseudo code.

8. Write short note on Objectory.

Object-oriented software engineering (OOSE), also called Objectory, is a method or object-oriented development with the specific aim to fit the development of large, real-time systems. Objectory, is a disciplined process for the industrialized development of software, based on a use-case driven design. Objectory is built around several different models:

- Use case-model.
- Domain object model.
- Analysis object model.
- Implementation model.
- Test model.

9. Define patterns.

- Design pattern identifies the key aspects of a common design structure that makes it useful for creating a reusable object-oriented design.
- Furthermore, it identifies the participating classes and instances, their roles and collaborations, and the distribution of responsibilities.
- It describes when it applies, whether it can be applied in view of other design constraints and the consequences and trade-offs of its use.
- A pattern is an instructive information that captures the essential structure and insight of a successful family of proven solutions to a recurring problem that arises within a certain context and system of forces.

10. Define proto-patterns.

A proto-pattern is the “pattern in waiting” which is not yet known to recur. It is the pattern waiting to undergo some degree of peer scrutiny or review.

11. Define patterns template. Give some examples for components in pattern.

Every pattern must be expressed in the form of a rule which is called as a template. It should establish a relationship between a context, a system of forces which arises in the context, and a configuration. Some of the essential components are:

- Name
- Problem
- Context
- Forces
- Solution
- Examples

12. Define anti-patterns.

An anti-pattern represents a worst practice while a pattern represents a best practice. Anti-patterns come in two varieties:

• Those describing a bad solution to a problem that resulted in a bad situation.

• Those describing how to get out of a bad situation

13. Define pattern mining. Give the steps involved in capturing pattern.

The process of looking for patterns to document is called pattern mining sometimes called reverse architecturing. The steps involved are:

Focus on practicability.

o Patterns should describe proven solutions to problems rather than the latest scientific results.

Aggressive disregard of originality.

o Pattern writers do not need to be the original inventor of the solution.

Non anonymous review.

o Pattern submissions are shepherded rather than reviewed.

Writers' workshops instead of presentations.

o Rather than being presented by the authors, the patterns are discussed in the writers' workshops.

Careful editing.

o The pattern authors should have the opportunity to incorporate all the comments and insights during the shepherding.

14. Define frame work.

A frame work is a way of presenting a generic solution to a problem that can be applied to all levels in a development.

Frameworks are a way of delivering application development patterns.

A framework provides architectural guidance, captures the design decisions that are common to its application domain and thus emphasize design reuse over code reuse.

15. Give the differences between design patterns and frameworks.

The major differences between design patterns and frameworks are as follows:

- A framework is executable software whereas design patterns represent knowledge and experience about the software.
- Design patterns are more abstract than frameworks.
- Design patterns are smaller architectural elements than frameworks.
- Design patterns are less specialized than frameworks.

16. Why do we go for unified approach?

The main motivation for going to unified approach is to combine the best practices, processes, methodologies, and guidelines along with UML (Unified Modeling Language) notations and diagrams for better understanding objectoriented concepts and system development.

17. Write short note on UA proposed Repository.

The repository allows the maximum reuse of previous experience and previously defined objects, patterns, frameworks, and user interfaces in an easily accessible manner with a completely available and easily utilized format.

Everything from the user request to maintenance of the project should be kept in the repository. This will reduce the cost and development time. It should be relatively easy to search the repository.

18. Define model.

A model is an abstract representation of a system, constructed to understand the system prior to building or modifying it. It is a model of a simplified representation of reality. Models can represent static or dynamic situations.

19. Explain about the types of model.

• Static model

→ It can be viewed as a snapshot of a system's parameters at rest or a specific point in time. They are needed to represent the structural or static aspect of a system. The UML class diagram is an example of static model.

• Dynamic model

→It can be viewed as a collection of procedures or behaviors that taken together reflect the behavior of a system over time. Dynamic modeling is the most useful during the design and implementation phases of the system development. The UML interaction diagrams and activity models are examples of dynamic models.

20. What are the advantages of Modeling?

Good models are essential for communication among project teams. As the complexity of systems increases, so does the importance of good modeling techniques. Some of the advantages are as follows:

- Models make it easier to express complex ideas.
- The main reason for modeling is to reduction of complexity.
- Models enhance and reinforce learning and training.
- The cost of modeling analysis is much lower than the cost of similar perimentation conducted in real system.

- Manipulation of the model is much easier.

21. Define UML.

The unified modeling language (UML) is a language for specifying, constructing, visualizing and documenting the software system and its components. The UML is a graphical language with sets of rules and semantics.

22. Mention the primary goals in the design of the UML.

The primary goals in the design of the UML were as follows:

- Provide users a ready-to-use, expressive visual modeling language so they can develop and exchange meaningful models
- Provide extensibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development processes.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of the OO tools market.
- Support higher-level development concepts.
- Integrate best practices and methodologies.

23. Give the nine UML graphical diagrams.

a. Class diagram(static)

b. Use-case diagram

c. Behavior diagram(dynamic)

    i. Interaction diagram

        1. Sequence diagram

        2. Collaboration diagram

    ii. State chart diagram

    iii. Activity diagram

d. Implementation diagram.

    i. Component diagram

ii. Deployment diagram.

24. What is a Package?

A package groups and manages the modeling elements, such as classes, their associations, and their structures. Packages themselves may be nested within other packages.

25. Explain implementation diagram

Implementation diagrams show the implementation phase of system development, such as the source code structure and the run-time implementation structure.

Reference Book(s):

1. Booch G.,"Object Oriented Analysis And Design",Addison- Wesley Publishing Company,1994.

2. Rambaugh J, Blaha.M. Premeriani, W., Eddy F and Loresen W., "Object Oriented Modeling and Design", PHI, 1997.

Reference Site : www.scribd.com