

Chapter-01

Information System- What are they?

1. What is System?

Ans. System is an abstraction of a complex interacting set of elements, for which it is possible to identify a boundary, and environment, inputs and outputs, a control mechanism and some process or transformation that the system achieves.

2. What are the characteristics of a system?

Ans. A System has the following characteristics

- a) A System exists in an **environment**.
- b) A system is separated from its environment by some kind of **boundary**.
- c) Systems have inputs and outputs. They receive inputs from their **environment** and send outputs into their **environment**.
- d) Systems have **interfaces**. An interface allows **communication between two systems**.
- e) A system may have **sub-systems**. A sub-system is also a system and may have further subsystems of its own.
- f) System control relies on **feedback** (and sometimes feed-forward). These comprise information about the System's operation as its environment that is passed to the control mechanism.
- g) A system has some properties that are **not directly dependent on the properties** as they only **emerge** at the level of the system as a whole.

3. What are the element of a system?

Ans. Elements of a system are-

- a) Boundary and environment
- b) Input, output and interface
- c) Sub-systems
- d) Control in systems
- e) Feedback
- f) Feed-forward
- g) Emergent properties

4. What is the difference between information and data?

Ans. Information is the facts that have been selected as relevant to a purpose and then organized or processed in such a way that they have meaning for that purpose. Information is conveyed by message and has meaning. On the other hand, data is raw facts that not yet identified as relevant to any particular purpose.

5. Difference between IS and IT.

Ans.

Information system	Information Technology
IS played an important part of human affairs. Used to capture, store, organize and display	IT Strategy is responsible for identifying the hardware component and configurations that

information.	will allow the software to operate effectively.
IS only be considered in the context of well thought-out business strategy.	IT only be considered in the context of specific information systems that are planned for development.
IS strategy is about what is feasible.	The role of IT strategy is to enable the successful defined in the information system strategy.

6. Difference between feedback and feed forward.

Ans.

Feedback	Feed forward
Feedback is sampling one or more outputs of a system for comparison to a control value.	Feed forward is sampling a system input, usually before it enters the system.
No one can develop a new skill, without receiving appropriate feedback.	Feed forward control information can help a system to be more responsive to environmental fluctuations.

7. What is the purpose of MIS?

Ans. The purpose of MIS is to

-Extract data from existing operational system.

-and analysis or combine it to give managing in information about the part of the organization for which they were responsible.

Chapter-02

Problems in Information System Development

1. Define quality.

Ans. Quality means fitness for purpose. Since it can be hard to identify the purpose.

2. What is Stakeholder?

Ans. A stakeholder has an interest in a project because they are(or will be) affected by its programs or by its results.

3. What are the main underline causes of problem in information system?

Ans. The main underline causes of problem in IS are discussed in the following perspective

a) And end users perspective-

1. A software product that is much talked about but never released to its intended users.
2. System may fail to meet the criterion of usability in a number of ways such as poor interface design, inappropriate or illogical sequence of data entry, incomprehensive error messages, unhelpful help, poor response times and unreliability in operation
3. System is very pretty but it does not do anything useful.

b) A clients perspective-

1. Some clients also have the power to stop a project once it is underway.

c) A developers perspective.

4. Difference between Quality problem and productivity problem.

Ans.

<u>Quality problem</u>	<u>Productivity problem</u>
It means fitness for purpose.	It relates to rate of progress of a project and the resources that it consumes along the way.
For quality assurance it is necessary to know -the purpose of system and -how to measure its fitness.	For productivity it is necessary to know -if the product delivered. -if it is delivered in time. -if it is affordable

Chapter-03

Avoiding the problems

1. Advantages & disadvantages of traditional waterfall life cycle.

Ans. Advantages:

- a) Teams with specialized skill can be assigned to tasks in particular phases.
- b) Progress can be evaluated at the end of each phase.
- c) Attendant risk can be controlled and managed.

Disadvantages:

- a) Real project rarely follow a simple sequential life cycle include.
- b) Iteration are almost inevitable.
- c) The elapsed time between inception and delivery is frequently too long.
- d) It is unresponsive to changes in the technology or requirements.

2. Phases of waterfall life cycle.

- a) System engineering
- b) Requirements analysis.
- c) Design.
- d) Construction
- e) Testing
- f) Installation
- g) Maintenance

3. Prototyping

In software development a prototype is a system or a partially complete system that is build quickly to explore some aspect of a system requirements and that is not intended as the final working system.

Main system require to prepare prototype

- 1) Perform an initial analysis.
- 2) Define prototype objectives.
- 3) Specify prototype.
- 4) Construct prototype.
- 5) Evaluate prototype and recommend change.

Advantages of prototype:

- 1) Early demonstrations of system functionality help identify any misunderstandings between developers and client.
- 2) Client requirements that have been missed are identified.
- 3) Difficulties in the interface can be indentifies
- 4) The feasibility and usefulness of the system can be tested even though by its very nature the prototype is incomplete.

Disadvantages:

- 1) The client may perceive the prototype as part of the final system, may not understand the effort that will be required to produce a working production system and may expect delivery soon.
- 2) The prototype may divert attention from functional to solely interface issue.
- 3) Prototype requires significant user involvement.
- 4) Managing the prototyping life cycle requires careful decision making.

4. USDP

The **Unified Software Development Process** (USDP) (Jacobson et al, 1999) reflects the current emphasis on iterative and incremental life cycles.

A development cycle for the USDP comprises four phases-

- a) Inception
- b) Elaboration
- c) Construction
- d) Transition

5. Incremental Development

Incremental development involves some initial analysis to scope the problem and identify the major requirements. The requirements are then reviewed and those that deliver most benefit to the client become the focus of the first increment of development and delivery. The installation of the first increment provides valuable feedback to the development team and informs the development of the second increment and so on.

Chapter-05

Modeling Concepts

1. What is the difference between model and diagram?

Ans. **Model:**

Like any map, models represent something also. Models are usually both abstract and visible. They are useful in several different ways, precisely because they differ from the things that they represent-

- a) A model is quicker and easier to build.
- b) A model can be used in simulations to learn more about the thing it represents.
- c) A model can evolve as we learn more about a task or problem.
- d) We can choose which details to represent in a model and which to ignore. It is an abstraction.
- e) A model can represent real or imaginary things from any domain.

Diagram: Diagrams are used to build models of system in the systems in the same way as architects use drawings and diagrams to model buildings. Diagrammatical models are used extensively by system analysts and designers in order to-

- a) Communicate ideas
- b) Generate new ideas and possibilities
- c) Test ideas and make predictions
- d) Understand structures and relationships

A model provides a complete view of a system at a particular stage and from a particular perspective.

2. What are the basic elements of UML model diagram?

Ans. UML diagrams are made up of four elements-

- a) Icons
- b) Two dimensional symbols
- c) Paths
- d) Strings

UML diagrams are graphs composed of various kinds of shapes, known as nodes, joined together by lines, known as paths.

- Different models present different views of the system. Booch et al (1999) suggest five views to be used with UML
 - 1. The use case view
 - 2. The design view
 - 3. The process view
 - 4. The implementation view
 - 5. The development view

3. What is the UML notation for each of the following package, sub-system and model?

Ans.

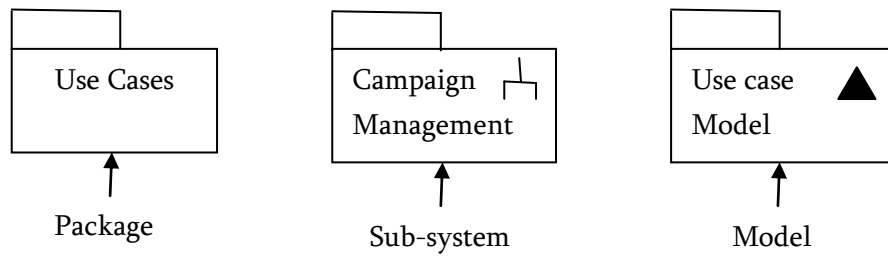


Fig: UML notation for packages, sub-systems and models.

4. Draw a simple Activity diagram.

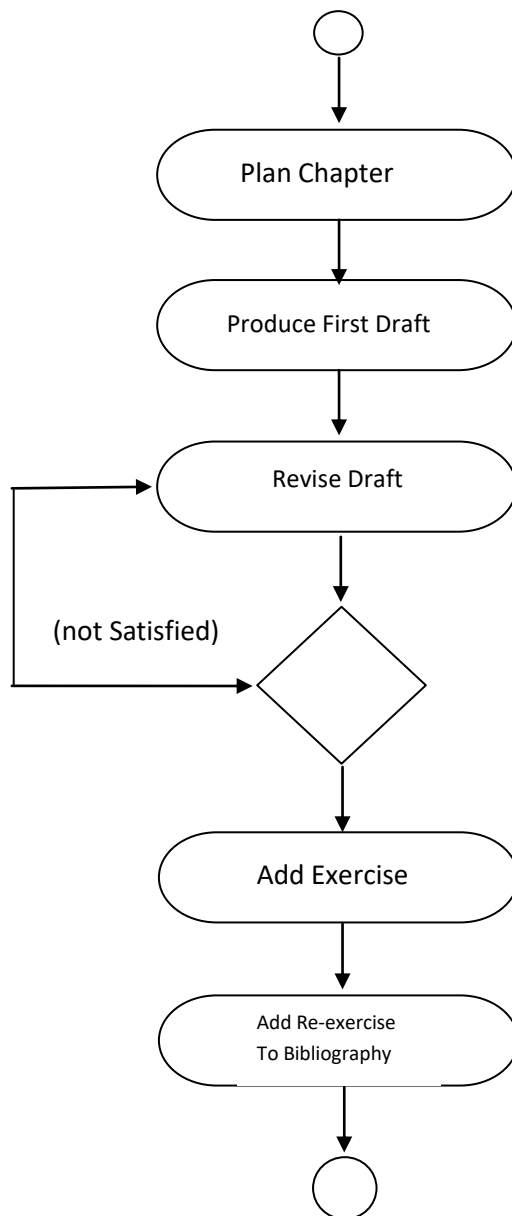


Fig: Activity diagram for the Activity write chapter.

Activity Diagram:

Activity diagram can be used to model different aspects of a system. At a high level, they can be used to model business activities in an existing or potential system.

Activity diagram can be used for the following purpose:

- 1) To model a task
- 2) To describe a system function that is represented by a use case.
- 3) In operation specifications, to describe the logic of an operation.
- 4) In UML to model the activities that make up the life cycle.

5. Guard Condition

Ans. Guard condition is a Boolean expression associated with a transition that is evaluated at the time the event fires. The transition only takes place if the condition is true. A guard condition is a function that may involve parameters of the triggering event and also attributes and links of the object that owns the state chart.

Chapter-06

Requirements Capture

1. List the name of the fact finding techniques.

Ans. There are 5 main fact finding techniques that are used by analyst to investigate requirements-

- a) Background reading
- b) Interviewing
- c) Observation
- d) Document Sampling
- e) Questionnaires

Advantage & disadvantage of fact finding technique:

- a) Background reading:
 - (+) Background reading helps the analyst to get on understanding of the organization before meeting the people who work there
 - (-) written document often do not match the reality, they may be out of date or they may reflect the official policy on matter that are dealt with differently in practice
- b) Interviewing:
 - (+) Personal contact allows the analyst to be responsive and adapt to what the user says. Because of this interviews produce high quality information.
 - (-) Interviews are time consuming and can be the most costly form of fact gathering.
- c) Observation:
 - (+) Observation of people at work provide firsthand experience of the way that the current system operates.
 - (-) Observation requires a trained and skilled observer for it to be most effective.
- d) Document Sampling:
 - (+) Can be used to gather quantitative data such as the average number of lines on an invoice.
 - (-) If the system is going to change dramatically. existing documents may not reflect how it will be in future.
- e) Questionnaires :
 - (+) An economical way of gathering data from a large number of people.
 - (-) Good questionnaires are difficult to construct.

2. What is use case? What is the purpose of use case?

Ans. Use cases are descriptions of the functionality of the system from the users' perspective. Use case diagrams are used to show the functionality that the system will provide and to show which users will communicate with the system in some way to use that functionality.

Purpose:

Use cases are supported by behavior specifications that specify the behavior of each use case either using UML diagrams, such as collaboration diagrams or sequence diagrams or in text form as use case descriptions.

Textual use case description provides a description of the interaction between the users of the system, termed actors and the high level functions within the system the use case.

3. What is Stereotypes? Describe include & extend.

Ans. Stereotype:

A stereotype is a special use of a model element that is constrained to behave in a particular way. Stereotypes can be shown by using a keyword. Such as 'extend' or 'include' in matched notations like <<extend>>. Stereotype can also be represented using special icon. The actor symbol in use case diagrams is a stereotyped icon- an actor is a stereotyped class and could also be shown as a class rectangle with the stereotype <<actor>> above the name of the actor.

<<extend>> is used when we wish to show that a use case provides additional functionality that may be required in another use case.

<<include>> applies when there is a sequence of behavior that is used frequently in a number of use cases and we want to avoid copying the same description of it into each use case in which it is used.

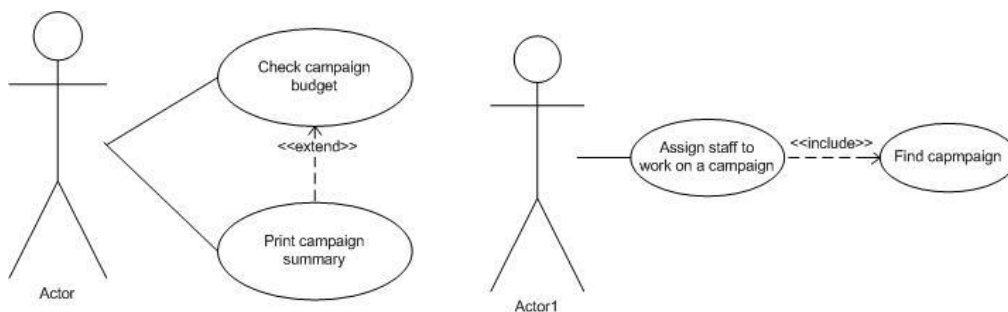


Fig: use case diagram showing <<extend>> and <<include>>

Chapter-07

Requirements Analysis

1. What is a collaboration diagram?

Ans. A collaboration diagram shows an interaction between objects and the context of the interaction in terms of the link between the objects.

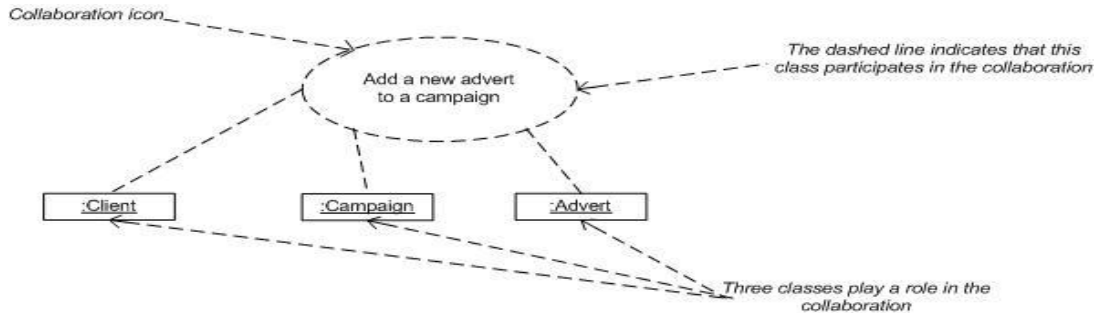


Fig: Collaboration for add a new advert to a campaign.

2. Define boundary class, entity class and control class?

Ans. Boundary class:

Boundary class is a stereotyped class that provides an interface to users or other system.

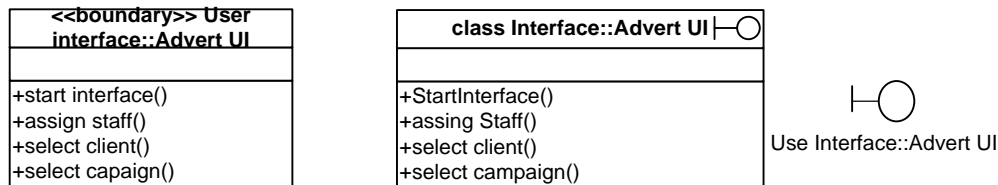


Fig: Alternative notations for boundary class stereotype.

Entity class:

Entity class is a stereotyped class that represents objects in the business domain model.

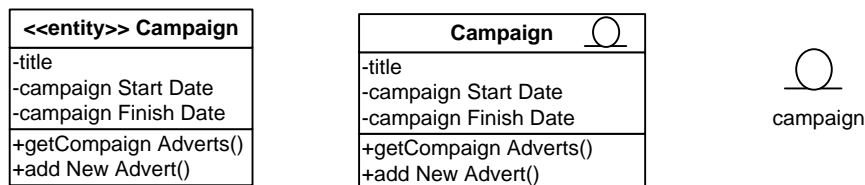


Fig: Alternative notation for an entity class.

Control class:

Control class is a stereotyped class that controls the interaction between boundary classes and entity classes.

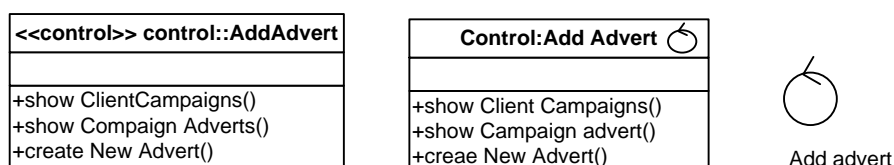


Fig: Alternative notation for a control class.

3. Distinguish between link and association.

Ans. Link: Link is a connection between objects and instance of an association.

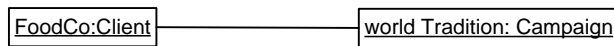


Fig: A link between instances.

Association: - Association is a logical connection usually between different classes although in some circumstances a class can have an association with itself. An association describes possible links between objects and may correspond either to logical relationships in the application domain or to message paths in software.

4. What is multiplicity and why can it be called a constraint?

Ans. Multiplicity: Multiplicity devotes the range of values of the members of objects that can be linked to a single object by a specific association.

It is a constraint because it limits the behavior of a system. If a client can have only one staff contact it should be impossible to link a second.

5. How does a collaboration diagram differ from class diagram?

Ans. A collaboration diagram shows only those classes that collaborate to provide the functionality of a particular use cases (or operation); the links that are shown are those that are required for that purpose.

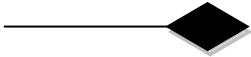
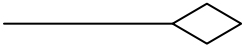
a class diagram typically shows all the classes in a particular package and all the associations between them.

Chapter-08

Refining the Requirements Model

1. Distinguish between composition from aggregation.

Ans.

Composition	Aggregation
<p>1. Composition is a type of abstraction that encapsulates groups of classes that collectively have the capacity to be a reusable sub-assembly. Represent the whole and the other part of the whole.</p>	<p>1. Aggregation represents a whole part association between two or more objects.</p>
<p>2. Symbol </p>	<p>2. Symbol </p>
<p>3. A part can belong only one composition.</p>	<p>3. A part can belong more than one aggregation.</p>

2. What are the advantages of components?

Ans. The use of component saves time and work.

3. Pattern:

A pattern is an abstract solution to a commonly occurring problem in a given context.

Chapter-09

Object Interaction

1. What is interaction sequence diagram?

Ans. Sequence diagram or interaction sequence diagram shows an interaction between objects arranged in a time sequence. Sequence diagram can be drawn at different levels of detail and also to meet different purpose and several stages in the development life cycle.

❖ Interaction diagrams are two types-

- a. Sequence diagram
- b. Collaboration diagram

2. Difference between sequence diagram and collaboration diagram.

Ans.

Sequence diagram	Collaboration diagram
<ol style="list-style-type: none">1. Sequence diagram shows an interaction between objects arranged in a time sequence.2. Sequence diagrams have a time dimension.3. It does not show the link between object.	<ol style="list-style-type: none">1. Collaboration diagram shows an interaction between object and the content of the interaction in terms of the links between the objects.2. Do not have time dimension.3. It shows the link between objects.

3. What is an object lifeline and focus of control?

Ans.

Object lifeline: An object lifeline represents the existence of an object during an interaction represented in a sequence diagram.

Focus of control: Focus of control indicates which operation is executing at a particular stage in an interaction represented in a sequence diagram.

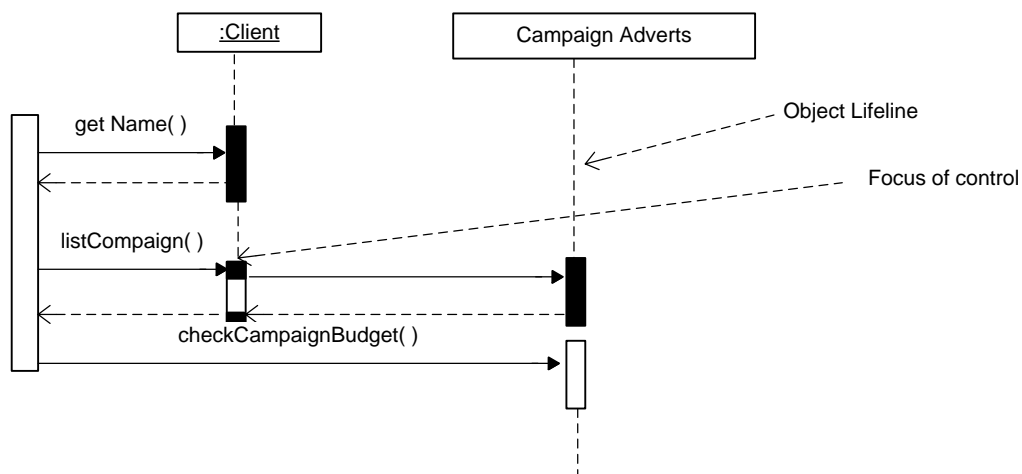




Fig: Sequence diagram showing object lifeline and Focus of control.

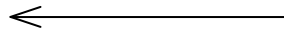
4. **Difference between Synchronous and Asynchronous message.**

Ans.

Synchronous message	Asynchronous message
<ol style="list-style-type: none"> 1. Synchronous message or procedural call is shown with a full arrowhead.  <ol style="list-style-type: none"> 2. It causes the invoking operation to suspended execution until the focus of control has been returned. 	<ol style="list-style-type: none"> 1. Asynchronous message is shown with an open arrowhead.  <ol style="list-style-type: none"> 2. It does not cause the invoking operation to Holt execution while it awaits a return.

5. Callback:

It may be necessary for an operation that has been invoked asynchronously to notify the object that invoked it when it has terminated. This is done by explicitly sending a message (known as callback) to the originating object.



Chapter-10

Specifying Operations

1. What are the two main purpose of operation specification?

Ans.

- a) They confirm the user's view of the logical behavior of a model.
- b) They also specify what the designers and programmers must product or meet the user's requirements.

2. Difference between algorithmic and non-algorithmic technique to operation specification.

Ans.

Algorithmic technique	Non-Algorithmic technique
<ul style="list-style-type: none">1. An algorithm defines the step-by-step behavior of an operation.2. An algorithm also specifies the sequence in which the steps are performed.3. Generally do not prefer in object-oriented development.4. Describe the internal logic. eg. Activity Diagram.	<ul style="list-style-type: none">1. A non-algorithmic approach defines only inputs and results.2. If does not specifies the sequence.3. Generally preferred in object-oriented because Non-algorithmic methods of operation specification emphasize encapsulation.4. Do not describe. eg. Decision table.

3. What are the main component of OCL operation?

Ans. The UML has also a formal language known as Object Constraint Language(OCL). which is intended mainly for specifying general constraints on a model.

Most OCL statement consists of the following three components-

- a) A context within which the expression is valid (for example, a specified class).
- b) A property within the context to which the expression applies(for example and attribute of the specified class).
- c) An operation that is applied to the property (for example a mathematical expression that tests the value of the attribute)

4. Define Activity Diagram.

Ans.

Activity Diagram: It is a version of state chart diagram that focuses on a flow of activity driven by internal processing within an object rather than by events that are external to it. In an activity diagram most (or all) states are action states (also called activities) each of which represent the execution of an operation. Activity diagrams can be used to specify the logic or procedurally complex operations.

Chapter-11

Specifying Control

1. What do you mean by state chart?

Ans. The state chart is a versatile technique and can be used within an Object-Oriented approach for other purpose than the modeling of object life cycles. A state chart must have at least one initial state.

2. Mention the important link between state-chart and iteration diagram.

Ans. There is an important link between state chart and iteration diagrams. A model of state behavior in a state chart captures all the possible responses of a single object to all the use cases in which it is involved. By contrast a sequence or a collaboration diagram captures the responses of the entire object that are involved in a single use case.

3. What are the main approaches for state-chart?

Ans. The steps involved in the life cycle to state modeling are as follows-

- a) Identify major system events.
- b) Identify each class that is likely to behave a state dependent response to these events.
- c) For each of these classes produce a first-cut state chart by considering the typical life cycle of an instance of the class.
- d) Examine the state chart and elaborate to encompass more detailed event behavior.
- e) Enhance the state chart to include alternative scenario us.
- f) Review the state chart to ensure that is consistent with the use cases. In particular check that the constraints that the state chart implies are appropriate.
- g) Iterate through steps d, e & f until the state chart captures the necessary level of detail.
- h) Ensure consistency with class diagram and interaction diagrams and other state charts.

Chapter-12

Moving in to Design

1. *Difference between Cohesion and Coupling.*

Ans.

- ✓ Cohesion: Cohesion is the degree to which the responsibilities of a single component form a meaningful unit.
- ✓ Coupling: Coupling describes the relationship between software components.
- ✓ Goal- Reduce coupling increase cohesion.

2. *What is the advantage to separating analysis from designing?*

3. *List 12 quality criteria for good design?*

Ans. Functional, efficient, economical, reliable, secure, flexible, general, buildable, manageable, maintainable, usable, reusable.

4. *What make good analysis?*

- a) Correct scope
- b) Completeness
- c) Correct content and
- d) Consistency.

Chapter-13

System Design

1. What are the major elements of system design?

Ans. Standard for code development and human computer interaction.

2. What is software architecture?

Ans. A software architecture is a description of the sub-systems and components of a software system and the relationship between them.

3. What is Layering and partitioning?

Ans. There are two general approaches to the division of a software system into subsystems. These are known as Layering and partitioning.

Layering- The different sub-systems usually represent different levels of abstraction.

Partitioning- Usually means that each subsystem focuses on different aspect to the functionality of the system as a whole.

Guidelines on the development of Layered architecture:

- i) Define the criteria by which the application will be grouped into layers.
- j) Determine the member of layers.
- k) Name the layers and assign functionality to them.
- l) Specify the services for each layer.
- m) Refine the layering by iterating through steps i to l.
- n) Specify interfaces for each layer.
- o) Specify the structure of each layer.
- p) Specify the communication between adjacent layer.
- q) Reduce the coupling between adjacent layer.

➤ What is MVC?

MVC means Model View Controller architecture where-

- ✓ Model provides the central functionality of the application and is aware of each of its dependent view and controllers components.
 - ✓ View corresponds to a particular style and format of presentation of information to the user.
 - ✓ Controller accept user input in the form of events that trigger the execution of operation within the model.
-
- Open layered architectures are more difficult to maintain because each layer communicate with all lower layers hence increasing the degree of coupling in the architecture. A change to one layer may ripple to many layers.
 - A closed layer architecture may require more processing as messages have to be passed through intervening layers.

Chapter-14

Object Design

- What levels of visibility may be assigned to an attribute or an operation?

Ans. Public, Private and Protected.

- Attributes should be designated private to enforce encapsulation.

Attribute:

Attribute is an element of the data structure that together with operation, defines a class.

Describes some property of instances of the class.

- An attributes data type is declared in UML using the following system:

name ':' type-expression '=' initial-value '{' property-string '}'

attribute name

Data type

balance : Money =0.00

- Operation: Operation in an aspect of the behavior that defines a class an element of the services that are provided by a class; a specification of an element of system functionality that will be implemented as a method of an object.

The syntax used for an operation is-

operation name '(' parameter-list ')' ':' return-type expression

- Object visibility: Visibility is an UML modeling element (eg, attributes or operations) may be designated with different levels of accessibility or visibility.

Visibility symbol	Visibility	Meaning
+	public	The feature is directly accessible by an instance of any class
-	private	The feature may only be used by an instance of the class that include it.
#	protected	The feature may be used them by instances if the class that includes it or of a subclass or descendant of that class.
~	package	The feature is directly accessible only by instance of a class in the same package.

- Interfaces: An interface in UML is a group of externally visible (i.e. public) operations. An interface is equivalent to associations and only abstract operation.

- Further design guidelines:

1. Design clarity: design should be mode as easy as possible.
2. Don't over design
3. Control Inheritance Hierarchies.
4. Keep message and operation simple
5. Design volatility.

6. Evaluate by scenario
7. Design by Delegation
8. Keep classes Separate

➤ Association:

Association is a logical connection, usually between different classes.

1. One-to-one association
2. One-to-many
3. Many-to-many

➤ Integrity constraint:

A constraint that has to be enforced to ensure that the information system holds data that is manually consistent and is manipulated correctly.

- ✓ Referential integrity ensures that an object identifier in one object actually refers to an object that exists.
- ✓ Dependencies constraints ensure that attribute dependencies, values are maintained consistently where the value of one attribute is calculated from other attributes are maintained consistently.
- ✓ Domain integrity ensures that attributes only hold permissible values.

➤ Normalization: Normalization is a technique that group's attributes based upon functional dependencies according to several rules to produce normalized data structures that are largely redundancy.

Chapter-15

Design Patterns

1. Difference between patterns and framework:

Pattern	framework
<ol style="list-style-type: none">1. A pattern is an abstract solution to a commonly occurring problem in a given context.2. Patterns are more abstract and general.3. Patterns are more primitive4. A pattern cannot be directly implemented in particular software.	<ol style="list-style-type: none">1. Framework is a reusable mini-architecture that provides structure and behavior common to all application.2. Frameworks are abstract and general.3. Frameworks are primitive.4. A framework can be directly implemented in a particular to software.

2. The three main categories of purpose that a pattern can have are-

- a. Creational —————> Singleton Pattern
- b. Structural —————> Composite Pattern
- c. Behavioral —————> State Pattern

3. Singleton Pattern:

Can be used to ensure that only one instance of a class is created.

4. Benefits and danger of using patterns:

Benefits:

- a. Pattern provides a mechanism for the reuse of generic solutions for object oriented and other approaches.
- b. Pattern offers a vocabulary for discussing the problem domain.

Danger:

- a. Some people believe that the use of patterns can limit creativity.
- b. The use of pattern in an uncontrolled manner may lead to over design.

5. Related patterns are grouped together in catalogues.

Chapter-16

Human Computer Interaction

➤ *There are two metaphors that are widely used to represent the user interface-*

1. The idea that the user in conduction a dialogue with the system.
2. The idea that the user is directly manipulating objects on screen.

➤ *Dialogue metaphor:*

The idea that the user is carrying or a dialogue with the system is a metaphor.

a metaphor is a term that is used figuratively to describe something but is not applied literally.

➤ *There are a number of import at general characteristics of good dialogue design*

1. Consistency
2. appropriate user support
3. adequate feedback from the system
4. Minimal user input.