iNurture Education Solutions
TOMORROW'S HERE

Subject: Object Oriented Analysis and Design

Module Number: 01

Module Name: Introduction to UML

SME NAME: ENTER NAME
SUBMISSION DATE: MENTION DATE
VERSION CODE: TO BE FILLED AT HO
RELEASED DATE: TO BE FILLED AT HO

Introduction to UML

AIM:

To furnish students with the understanding of Unified Modeling Language (UML) to envision the design of a system in software engineering.

Introduction to UML

Objectives:

The objectives of this module is to:

- Understating the conceptual model of the UML and its architecture.
- Understanding about the Software Development Life Cycle (SDLC).

Introduction to UML

Outcomes:

At the end of this module, the students are expected to:

- To explain about UML and its application in software design.
- To explain about SDLC and it’s importance in software development.

Introduction to UML

Contents:

Introduction to UML

According to Grady Booch:

"Modeling is a central part of all the activities that lead up to the deployment of good software. We build models to communicate the desired structure and behavior of our system. We build models to visualize and control the system's architecture. We build models to better understand the system we are building, often exposing opportunities for simplification and reuse. And we build models to manage risk."

- The Unified Modeling Language User Guide (2nd Edition), Grady Booch.

Introduction to UML

Importance of modeling

- Modeling is a well acknowledged and substantiated engineering technique.
- In UML, a model has to be systematic, underscoring the behavioral, structural, and dynamics of the system.
- Basically, a model is a prescription of reality, or blueprints of a system.

Introduction to UML

Unified Modeling Language (UML)

- UML (Unified Modeling Language) is a standardized modeling language, comprising with a set of integrated diagrams, which were developed to assist system and software builders for describing, constructing, visualizing, and documenting the artifacts in the premises of software engineering that can further be extended for business modeling as well.
- UML was developed by the "*Object Management Group (OMG)*" and UML 1.0 prescription draft was put forward to the OMG in January 1997.
- It is quite different from other programming languages such as C, C++, Java, etc.

Introduction to UML

- The UML utilizes mostly graphical formats or notations to elaborate the designing of the software projects.
- It can assist project teams to communicate, explore different potential designs, and justify the architectural design for the particular software.

Introduction to UML

UML, in particular:

- Enables us to visualize a system.
- Simplifies the complex systems to understand it parts wise.
- Facilitates us to specify the behavior or structure of a system.
- Helps in building templates that may guide us in establishing a system.
- Assists in documenting the decisions that we have finalized.

Introduction to UML

Acceptance of UML:

- The language is nonproprietary and open to all.
- Companies are free to use it with their own methods. Tool vendors are free to create tools for it, and authors are encouraged to write books about it.
- During 1996, a number of organizations including Digital Equipment Corporation, Hewlett-Packard, I-Logix, Intellicorp, IBM, ICON Computing, MCI Systemhouse, joined to form the UML Partners consortium.

Introduction to UML

- Microsoft, Oracle, Texas Instruments, Unisys, and Rational saw UML as strategic to their businesses and contributed to the definition of UML.
- The companies also supported the proposal to adopt UML as an Object Management Group (OMG) standard.
- This made UML independent of any one company's business plan, making it more likely to act as a standard throughout the industry.

Introduction to UML

The Object Management Group (OMG)

- The Object Management Group was founded in 1989 by 11 member companies, including 3Com Corporation; American Airlines; Canon, Inc.; Data General; Hewlett-Packard; Philips telecommunications N.V.; Sun Microsystems; and Unisys Corporation, and now has over 800 members.
- The OMG is a not-for-profit corporation that is dedicated to establishing a component-based software industry through vendor independent specifications.
- Its goal is to produce industry guidelines and specifications in order to provide a common framework for application development.

Introduction to UML

Unified Modelling Language Elements :

With OMG support, the UML is now very well documented with a formal specification of the semantics of the language. At a high level, the details of UML are as follows:

- All UML diagrams describe object-oriented information.
- Class and object diagrams illustrate a system’s static structure and the relationship between different objects.

Introduction to UML

- Interaction diagrams, state machines, and activity diagrams, show the dynamic behavior of objects, as well as communications between objects.
- Use-case and activity diagrams show system requirements and process workflows.
- The composite structure diagram shows the collaborating features as they combine at run time to implement a specific UML element.
- Deployment diagrams help with the deployment of the software in a specific environment.
- Because no notation can cover every possible type of information, UML also supports the use of comments and defines their notation.

Introduction to UML

Methods and Modelling Languages

- There are important differences between a *method* and a *modeling language*.
- A ***method***, also called a methodology, is an explicit way of structuring one’s thinking and actions.
- It consists of a process, a standard vocabulary, and a set of rules and guidelines.
- It tells the user what to do, how to do it, when to do it, and why it is done.
- The method defines a set of activities that will accomplish the goals of the project, including the purpose of each specific activity, and what resulted from that activity.
- Examples include the Unified Process, Shlaer-Mellor, CRC (Class, Responsibilities, and Collaborators), and Extreme Programming.

Introduction to UML

A mature and effective methodology for object-oriented development will encompass many different elements:

- A full life-cycle process
- A language defining concepts and models that are consistent, including notation for how ideas are presented
- Defined roles with prescribed responsibilities
- Rules and guidelines that define activities and how they are performed, including the work products and deliverables they produce
- Defined analysis, design, development, and test strategies

Introduction to UML

- The main difference between a method and a modeling language is that the modeling language lacks a process or the instructions for what to do, how to do it, when to do it, and why it is done.
- Though UML standardizes models and has incorporated elements gathered from many different methods, the development approaches that use UML are as diverse as the environments in which it is used.
- It can provide the underlying notation and language of the process.
- Still, different projects will emphasize different diagrams and extensions to UML.

Introduction to UML

A ***modeling language*** consists of notation—the symbols used in the models—and a set of rules directing how to use it. The rules are syntactic, semantic, and pragmatic.

- **Syntax** tells us how the symbols should look and how the symbols in the modeling language should be combined.
- **Semantic** rules explain what each symbol means and how it should be interpreted, either by itself or in the context of other symbols.
- The **pragmatic** rules define the intentions of the symbols through which the purpose of a model is achieved and becomes understandable for others.

Introduction to UML

Principles of Unified Modeling Language

1. The selection of model is crucial.
2. Each and every model may be represented with the different precision levels.
3. The best considered model should belong to reality.
4. No model may claim to be sufficient.

Introduction to UML

Object Oriented Modeling

- "Object oriented design methods" made an appearance in the 1980s, and "object oriented analysis methods" emanated during the 1990s.
- Grady Booch worked on *how to design for Ada*, and released a paper entitled, "*Object Oriented Design"* , in the years of 1980s.
- Later in 1991, Booch extended his ideas to a genuinely *object oriented design method* in his same book, revised in 1993.

Introduction to UML

- The "*Object Modeling Technique (OMT)*" extends aspects of "*object oriented analysis and design (OOAD)*".
- OOT provides a very practical and productive way of Software development.
- As OOT is independent of any computer language, there is no need for considering a final implementation language, during Object Oriented Modeling (OOM).
- OOT incorporate control, structural and functional perspectives of the system.

Introduction to UML

The principal concepts of object orientation are as follows.

- Object orientation is a technology for producing models that reflect a domain, such as a business domain or a machine domain.
- Object-oriented software development has five underlying concepts: objects, messages, classes, inheritance, and polymorphism. Software objects have a state, a behavior, and an identity. The behavior can depend upon the state, and the state may be modified by the behavior. Messages that provide the communication between the objects of a system, and between systems themselves, have five main categories: constructors, destructors, selectors, modifiers, and iterators.

Introduction to UML

- Every object is a real-world “instance” of a class, which is a type of template used to define the characteristics of an object. Each object has a name, attributes, and operations. Classes are said to have an association if the objects they instantiate are linked or related.
- Object-oriented models, when constructed correctly, are easy to communicate, change, expand, validate, and verify.
- When done correctly, systems built using object-oriented technology are flexible in response to change, have well-defined architectures, and provide the opportunity to create and implement reusable components. The requirements of the system are traceable to the code of the system.

Introduction to UML

- Object-oriented models are conveniently implemented in software using object-oriented programming languages. Using programming languages that are not object-oriented to implement object-oriented systems is not recommended. However, it’s important to realize that object-oriented software engineering is much more than just a couple of mechanisms in a programming language.
- Object orientation is not just a theory, but a well-proven technology used in a large number of projects and for building many different types of systems. The field still lacks standardization to show the way to an industrialization of object technology. The work provided by OMG strives to achieve such standardization.
- Object orientation requires a method that integrates a development process and a modeling language with suitable construction techniques and tools.

Introduction to UML

Software Development Life Cycle

- A SDLC model (also called as process model) is a diagrammatic or pictorial representation of the software life cycle.
- This model illustrates all the required methods to develop a software commodity throughout its life cycle stages i.e., from genesis to retirement. It also covers the framework where these methods are to be carried out.
- In the span of any life cycle stage, more than one course of actions may also be undertaken.

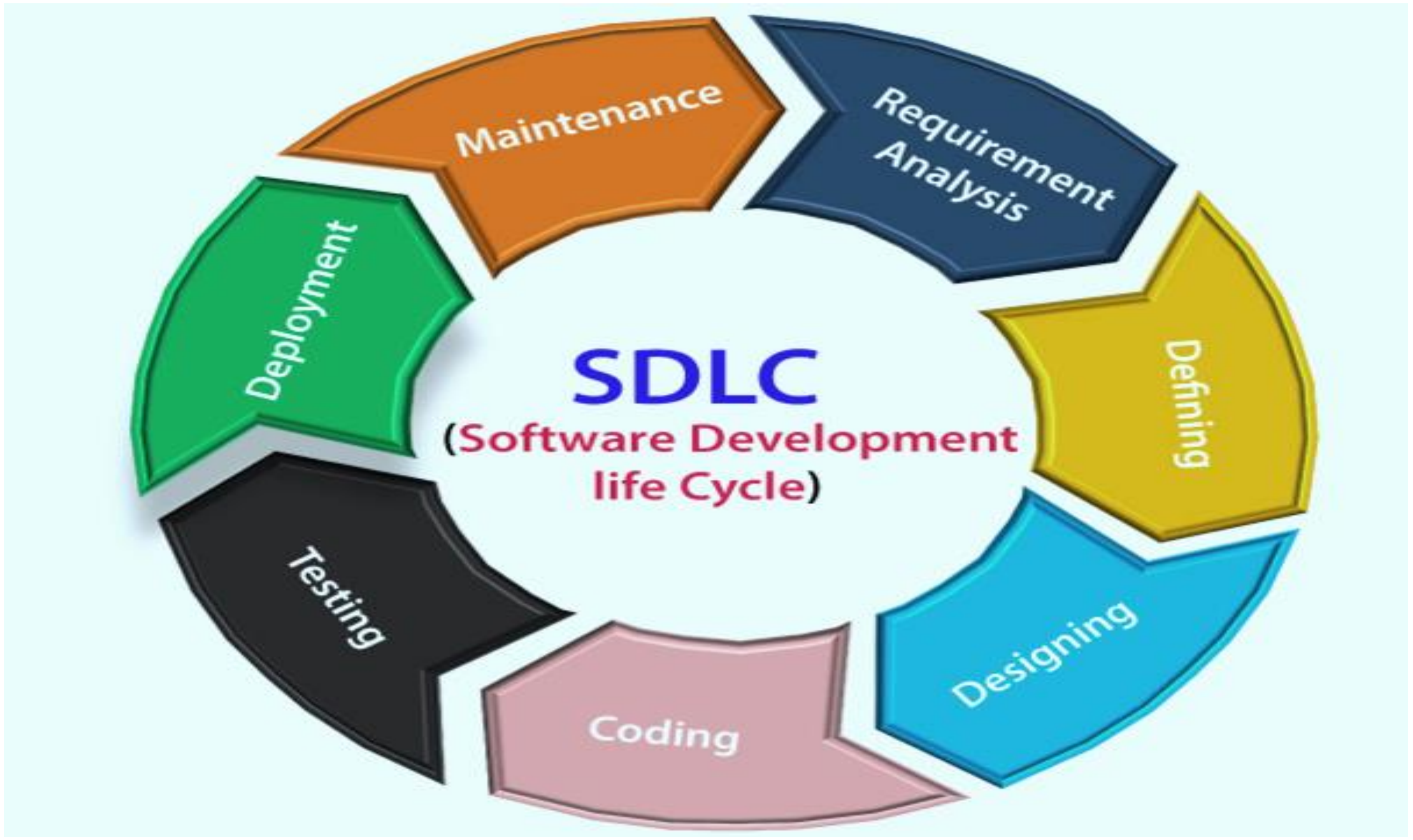
Introduction to UML

Requirement of SDLC

- A suitable life cycle model must be developed by a team to execute a particular plan effectively.
- An effective life cycle model can only develop a good software product with disciplined and systematic manner.
- While planning a life cycle model, every points are carried out effectively and systematically such as all the team members must be in synchronization, they must understand their roles and responsibilities properly, entry and exit criteria for each and every phase are well defined, etc.

Introduction to UML

SDLC is a framework for a set of well defined phases for developing a software. Those step are given as follows:



Introduction to UML

The phase of SDLC can be discussed as follows:

Phase1: Planning and requirement analysis

- Example, A client’s requirement is to application which performs money transactions. So, the requirements may be the number of operations to be done, their working process, currency, etc.
- After completing the required functions, an analysis is performed with auditing the feasibility concerns and of the future performance of the product. If any ambiguity occurs, a flag is raised for further discussion.
- After finalizing the requirements, the Software Requirement Specification (SRS) document is developed. First the developers go through the full document and then it is discussed with the customer for futher reference.

Introduction to UML

Phase2: Defining Requirements

- After performing requirement analysis, the next phage is to defining the set of requirements and it’s feasibilities and then it should be accepted by the project stakeholders.
- The output of this phase is a well defined "SRS document” which comprises all the requirements to be fulfilled during the project life cycle.

Introduction to UML

Phase 3: Designing the Software

- Collecting all the information of requirements, analyzing, and designing of the software project.
- This phase can be considered as the output of the last two phases, comprising inputs from the stakeholders and customer requirements.

Introduction to UML

Phase 4: Developing the project

- Here, the development of the project starts with programming or coding, considering the guidelines set by the team, using the tools like interpreters, compilers, debuggers, etc.

Introduction to UML

Phase 5: Testing

- In this phase testing is preformed against all the requirements to assure that it is cooperating with all the requirements set by the stakeholders.
- This phase consists system testing, unit testing, integration testing, and acceptance testing.

Introduction to UML

Phase 6: Deployment

- After coding is done with testing, and in the condition of no errors or bugs, the software is now deployed.
- Further, the software is released based on the assessments.

Introduction to UML

Phase 7: Maintenance

- Based on use of the software, real issues appear and further requirements are considered by the developer team.

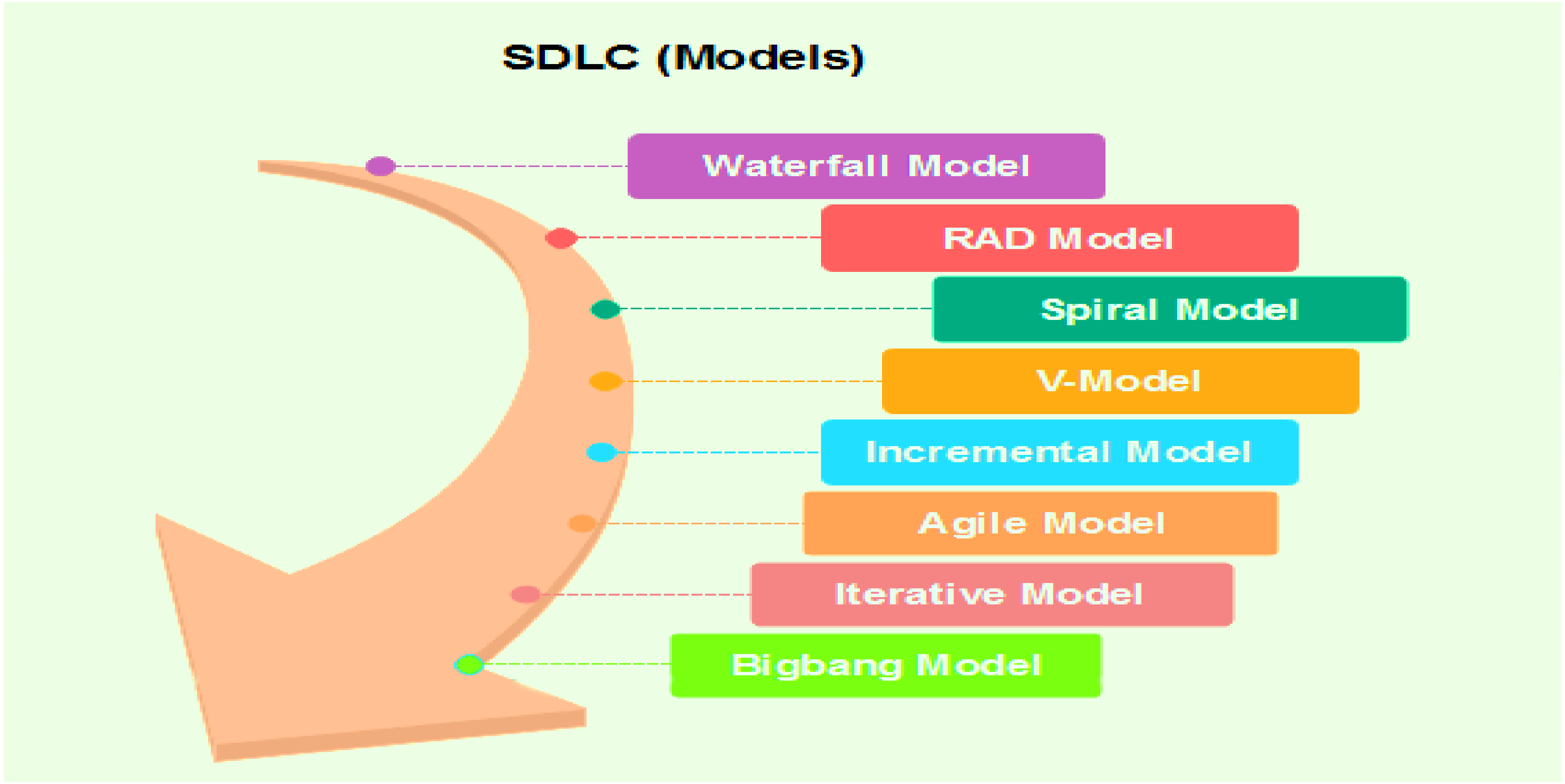
Introduction to UML

SDLC Models

- There are number of SDLC models, which are considered during the phases of software development .
- These are called "Software Development Process Models."
- Each and every process model comprises a series of unique phases which ensure success of different phases during the software development.

Introduction to UML

Here, are some widely accepted phases of SDLC model:



Introduction to UML

Self Assessment Question

1. An object oriented model is a __________ of reality.

a. Complication

b. Simplification

c. Realization

d. Generalization

Answer: b

Introduction to UML

Self Assessment Question

2. Object oriented models assist us to_________ a system as in a proper way.

a. Analyze

b. Design

c. Visualize

d. Measure

Answer: c

Introduction to UML

Self Assessment Question

3. object-oriented and Algorithmic are the two usual ways for modeling ________

a. Non-software Systems

b. Software Systems

c. Vocabulary of a System

d. Client/Server System

Answer: b

Introduction to UML

Self Assessment Question

4. _________ assists to convey the overall system architecture comprehensively.

a. Flow charts

b. Designing

c. SRS

d. Templates

Answer: b

Introduction to UML

Self Assessment Question

5. The leading type of models assists to choose ________

a. Degree of detail

b. Designing view

c. Single model

d. Choice of the model

Answer: a

Introduction to UML

Self Assessment Question

6. UML is beneficial to _________ a system in its primitive or in a well suited way.

a. Visualize

b. Specify

c. Document

d. All of the these

Answer: d

Introduction to UML

Self Assessment Question

7. The illustrative sections of the UML model are considered as ________

a. Behavioral objects

b. Grouping objects

c. Structural objects

d. Annotational objects

Answer: d

Introduction to UML

Self Assessment Question

8. Interfaces of UML are applied to ________

a. Explain an API for all the classes

b. Utilize in Java, but not in Smalltalk or C++

c. Describe executable logic to reapply across classes

d. Set down the required services for different types of objects

Answer: d

Introduction to UML

Self Assessment Question

9. How many phases are there in SDLC model ?

a. 4

b. 7

c. 6

d. 8

Answer: b

Introduction to UML

Self Assessment Question

10. Which of the following is not a phase of SDLC model ?

a. Requirement analysis

b. Discovering

c. Testing

d. Maintenance

Answer: b

Introduction to UML

Assignment

1. Explain about UML in your own wordings in at least 500 words.
2. Describe the different phases of SDLC model.
3. Write a note on at least 4 most preferred SDLC models.

Introduction to UML

Summary

1. Modeling can be considered as a collection of the predefined activities that assist us to the development and deployment of a good software.
2. Models are merely build to visualize and govern the system's architecture.
3. Basically, a model is a prescription of reality, or blueprints of a system.
4. The UML utilizes mostly graphical formats or notations to elaborate the designing of the software projects.
5. A SDLC model (also called as process model) is a diagrammatic or pictorial representation of the software life cycle.

Subject (Helvetica Bold- 24pt)

Document Link

Topic	URL	Notes
Importance of modeling, Principles of modeling, object oriented modeling	Web link: https://en.wikipedia.org/wiki/Object-oriented_analysis_and_design and https://en.wikipedia.org/wiki/Unified_Modeling_Language	To differentiate between object oriented design and object oriented modeling.

Subject (Helvetica Bold- 24pt)

Video Link

Topic	URL	Notes
	Video link https://www.linkedin.com/learning/programming-foundations-object-oriented-design-3/learn-object-oriented-design-principles?u=92695330	Basic concepts and principles of object oriented analysis and design
	Video link: Software Design: Modeling with UML https://www.linkedin.com/learning/software-design-modeling-with-uml/a-picture-is-worth-a-thousand-words?u=92695330	UML, Architecture

Subject (Helvetica Bold- 24pt)

E- Book Link

E-book name	URL
Object Oriented Modeling & Design Using UML	https://www.pdfdrive.com/the-unified-modeling-language-user-guide-second-edition-by-grady-booch-james-d191677284.html
UML 2 Toolkit	https://www.pdfdrive.com/uml-2-toolkit-d158470306.html