1. **Be exposed to the standard syntax used by the Unified Modelling Language.**

The Unified Modeling Language (UML) is a language for representing, outlining, building, and documenting the components of a software-intensive system. Still, it draws the line when you get close to writing actual code. For instance, a textual programming language is more suited for expressing complex branches and joins. People can benefit from the best of both worlds because of the tight mapping between the UML and a family of object languages (OMG UML, 2017).

UML offers a standard syntax roughly divided into three categories for different diagrams: structure, interaction, and behaviour. Diagrams of a system or an organisation's behaviour show how it functions. Diagrams that illustrate interactions show how different system components interact. The static organisation of a system's elements is described in structural diagrams (OMG Unified Modeling, 2003).

According to GeeksforGeeks (2017), the standard syntax of UML is based on the following principles:

- A **class** which describes an object's blueprint, structure, and operations.
- Using **objects**, we can break down complex systems and make them more modular. To create our system piece by piece, modularity allows us to break it down into manageable parts. A system's basic unit (building block) used to represent an entity is called an object.
- Child classes can inherit the properties of their parent classes through the **inheritance** mechanism.
- **Abstraction**: A method for hiding implementation specifics from the user.
- **Encapsulation** is the process of grouping and shielding data from the outside world.
- **Polymorphism** is a mechanism that allows for the existence of functions or things in several forms.

According to OMG UML (2017), here are some of the benefits of using the standard syntax of UML:

More effective communication: UML's uniform syntax offers a shared vocabulary for discussing software systems. This may enhance communication between management, customers, and software engineers.

Better documentation: UML models can generate documentation using the standard UML syntax. This can assist in making software system documentation easier to comprehend and maintain.

Increased reuse: UML models can be shared between many software development teams using the uniform syntax of UML. This could promote the reuse of programme components, increasing the effectiveness of software development.

Quality improvement: UML's standard syntax can impose ethical software design principles. This could aid in raising the calibre of software systems.

2. Explore the major UML models used throughout the object-oriented analysis, design and development process.

Finding out what users and consumers of a software project want the system to do is the goal of requirements analysis. According to Fowler (2004), the object-oriented analysis, design, and development process use the following UML models described as follow:

Class Diagram: describes the representation of objects and ideas like class, association, and multiplicity. It is a critical UML diagram used throughout the analysis, design, and development processes.

Object Diagram: A system's objects at a particular time are captured in an object diagram. An object diagram is frequently called an instance diagram because it displays instances rather than classes. Object diagrams are commonly employed to show how a system behaves.

Use Case Diagram: The use case diagram illustrates the actor, use case, and relationship connections. Use case diagrams are used to represent a system's requirements.

Activity Diagram: Workflow, business process, and procedural logic are all described using activity diagrams. Activity diagrams are frequently used to represent the business operations that a system supports.

Sequence Diagram: Using a sequence diagram, you can show how various items work together to handle a complicated task. To represent the interactions of system components, sequence diagrams are frequently described.

Communication Diagram: The data connections between the various players in the conversation are highlighted in communication diagrams. Communication diagrams frequently represent objects' interactions in a distributed system.

Deployment Diagram: Deployment diagrams display the physical arrangement of a system, indicating which software components are paired with which hardware components. Modelling the infrastructure that supports a system is frequently done via deployment diagrams.

The quality, efficacy, and efficiency of the object-oriented analysis, design, and development processes can all be enhanced with the help of UML models.

References:

An OMG Unified Modeling Language Publication OMG Unified Modeling Language (OMG UML), (2017). Available at: https://www.omg.org/spec/UML/2.5.1/PDF.

OMG Unified Modeling Language Specification. (2003). Available at: https://www.omg.org/spec/UML/1.5/PDF.

GeeksforGeeks (2017). *Unified Modeling Language (UML)* | *An Introduction - GeeksforGeeks*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/.

Fowler, M. (2004). *UML distilled: a brief guide to the standard object modeling language*. Addison-Wesley Professional.