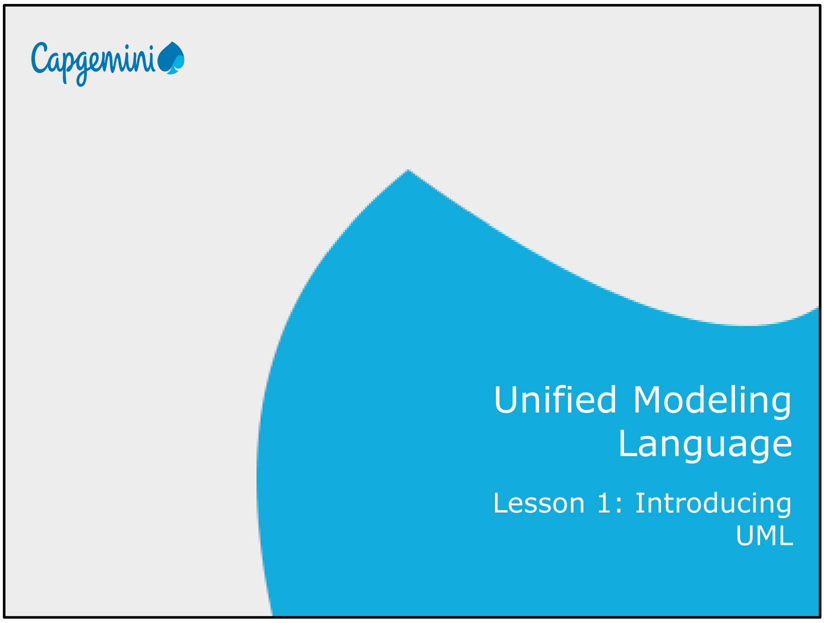
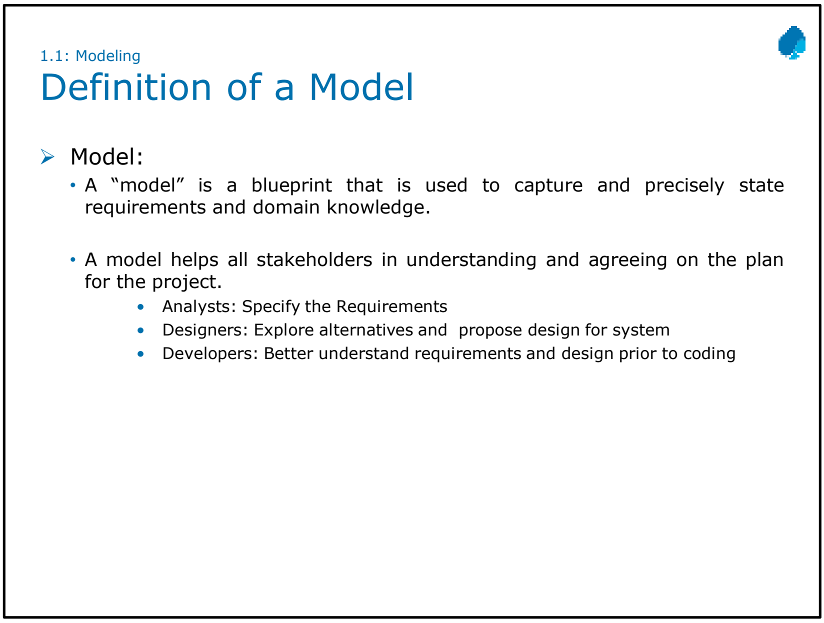


Unified Modeling Language

Lesson 1: Introducing UML

Lesson Objectives

- To understand the following topics:
  - Principles of Modeling
  - Basics of UML – What is UML? What UML is NOT?
  - UML building blocks
  - List of UML diagrams



1.1: Modeling

Definition of a Model

- Model:
  - A “model” is a blueprint that is used to capture and precisely state requirements and domain knowledge.
  - A model helps all stakeholders in understanding and agreeing on the plan for the project.
    - Analysts: Specify the Requirements
    - Designers: Explore alternatives and propose design for system
    - Developers: Better understand requirements and design prior to coding

Principles of Modeling: What is Modeling?

Modeling is an essential activity in many domains, including the fields of construction and engineering.

Actual building of a house is almost always preceded by a blueprint, a model, which describes the architectural layout and other details. Such a blueprint is essential for understanding the system and to convey the same to concerned parties.

The same concept applies to software systems as well. Models can be used to capture the knowledge about the system. A model helps to capture and precisely state the requirements and domain knowledge so that all stakeholders may understand and agree on the plan for the project.

Different models of a software system may capture the following:

requirements about its application domain,

the ways in which users will use the application,

its breakdown into application modules,

common patterns used in its construction, etc.

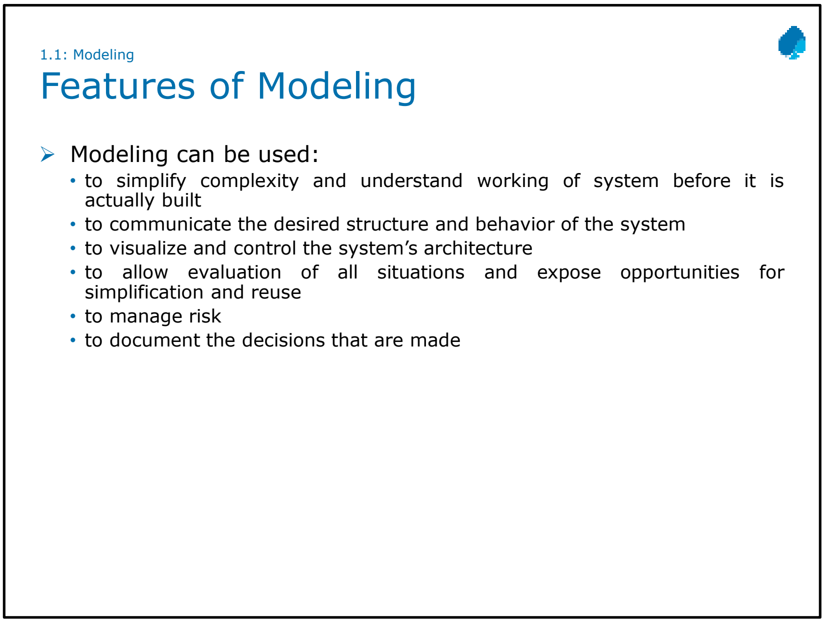
Thus modeling helps the developers easily explore several architectures and design solutions before writing code.

Models have two major aspects:

Semantic information (semantics)

Visual presentation (notation)

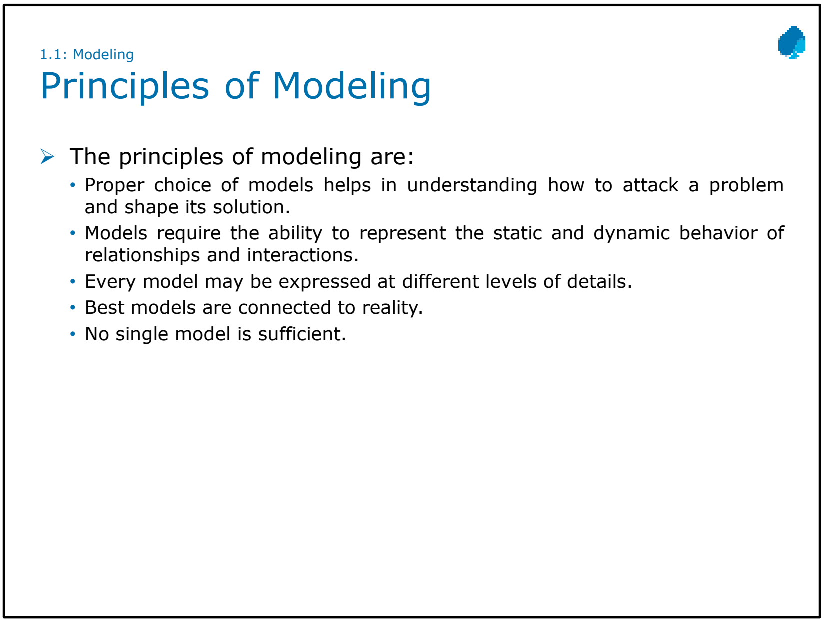
A model can tell what a function does (specification), and also how the function is accomplished (implementation).



1.1: Modeling

Features of Modeling

- Modeling can be used:
  - to simplify complexity and understand working of system before it is actually built
  - to communicate the desired structure and behavior of the system
  - to visualize and control the system’s architecture
  - to allow evaluation of all situations and expose opportunities for simplification and reuse
  - to manage risk
  - to document the decisions that are made



1.1: Modeling

Principles of Modeling

- The principles of modeling are:
  - Proper choice of models helps in understanding how to attack a problem and shape its solution.
  - Models require the ability to represent the static and dynamic behavior of relationships and interactions.
  - Every model may be expressed at different levels of details.
  - Best models are connected to reality.
  - No single model is sufficient.

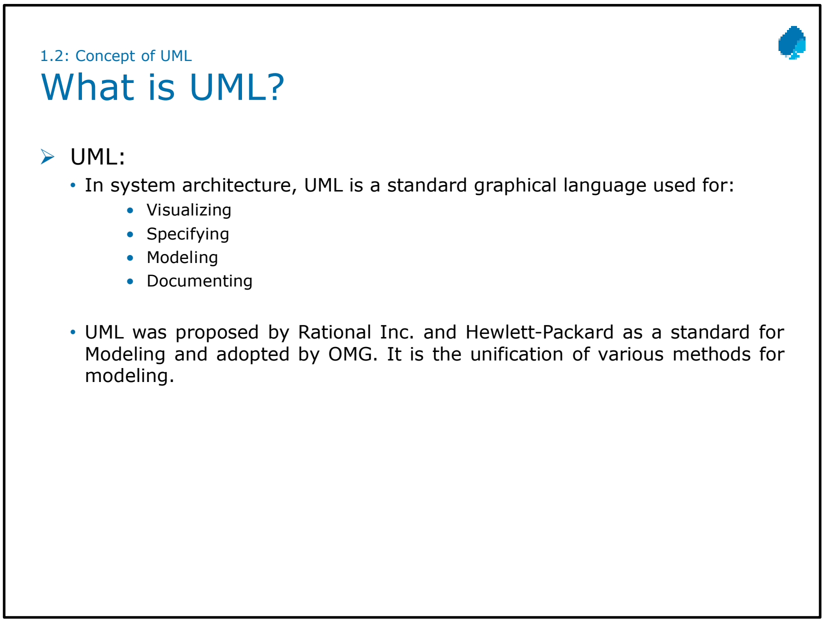
Principles of Modeling (contd.):

While modeling, the problem must always be kept in mind. Only the models that will add value to a “view” of the system must be considered. For example: If a system is supposed to be a stand alone system, having a model to depict the deployment details may not be required. However, such a model would be required for a system which is supposed to be deployed across a network.

Besides, every system has static as well as dynamic aspects. So the models must be capable of depicting the same.

Further, based on the view and the people it is intended for, models may be at different granular levels. Each user may require different degrees of details.

It is unlikely that one model gives the complete system description. This is where multiple models, each giving a different (but relevant) view of the system, becomes important. So proper choice of models is important, which will best help in understanding how to attack the problem and shape its solution.



1.2: Concept of UML

What is UML?

- UML:
  - In system architecture, UML is a standard graphical language used for:
    - Visualizing
    - Specifying
    - Modeling
    - Documenting
  - UML was proposed by Rational Inc. and Hewlett-Packard as a standard for Modeling and adopted by OMG. It is the unification of various methods for modeling.

What is UML?

OMG specification states: “The Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system”. The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions, as well as concrete things such as programming language statements, database schemas, and reusable software components.

UML is used for the following tasks:

Visualizing - Visual Model helps better communication and goes beyond what can otherwise be textually described.

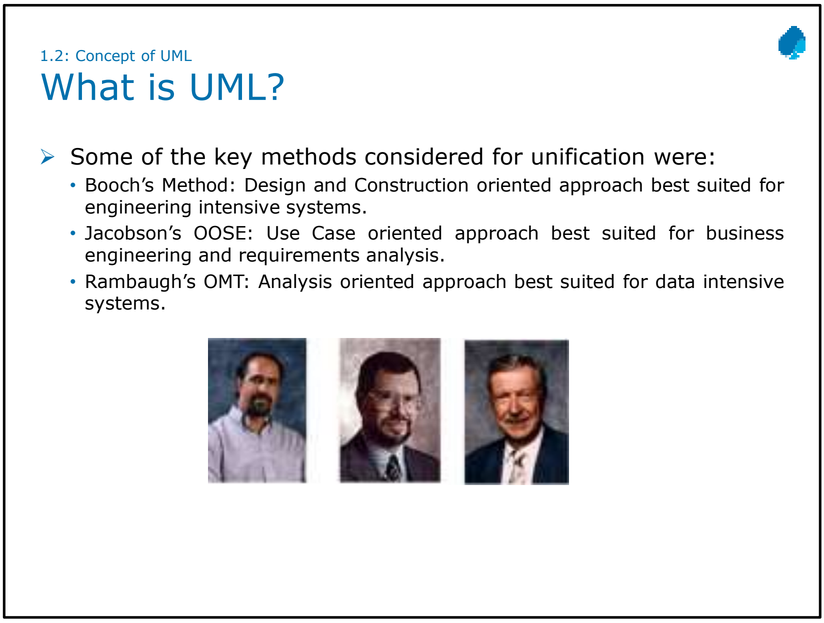
Specifying - UML can help in specifying all important analysis, design, and implementation decisions.

Modeling - Allows for construction of the system from the various models.

Documenting - Models can help in documenting all decisions taken during the entire system development lifecycle.

Prior to UML, there were many methods with similar modeling languages having minor differences in overall expressive power. However, there was no single “leading” modeling language. Lack of disagreement on a general-purpose modeling language discouraged new users from adopting the OO approach.

contd.



1.2: Concept of UML

What is UML?

- Some of the key methods considered for unification were:
  - Booch’s Method: Design and Construction oriented approach best suited for engineering intensive systems.
  - Jacobson’s OOSE: Use Case oriented approach best suited for business engineering and requirements analysis.
  - Rambaugh’s OMT: Analysis oriented approach best suited for data intensive systems.

What is UML? (contd.)

In the period around mid-90s, there were efforts made in the direction of unifying the prominent methods available at that time. After a couple of drafts, UML was adopted by OMG in 1997. Some of the key methods considered for unification were:

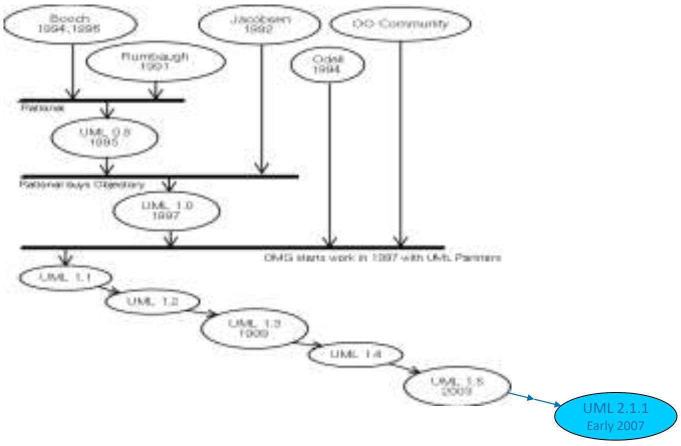
Booch’s Method: Design and Construction oriented approach best suited for engineering intensive systems.

Jacobson’s OOSE: Use Case oriented approach best suited for business engineering and requirements analysis.

Rambaugh’s OMT: Analysis oriented approach best suited for data intensive systems.

1.2: Concept of UML

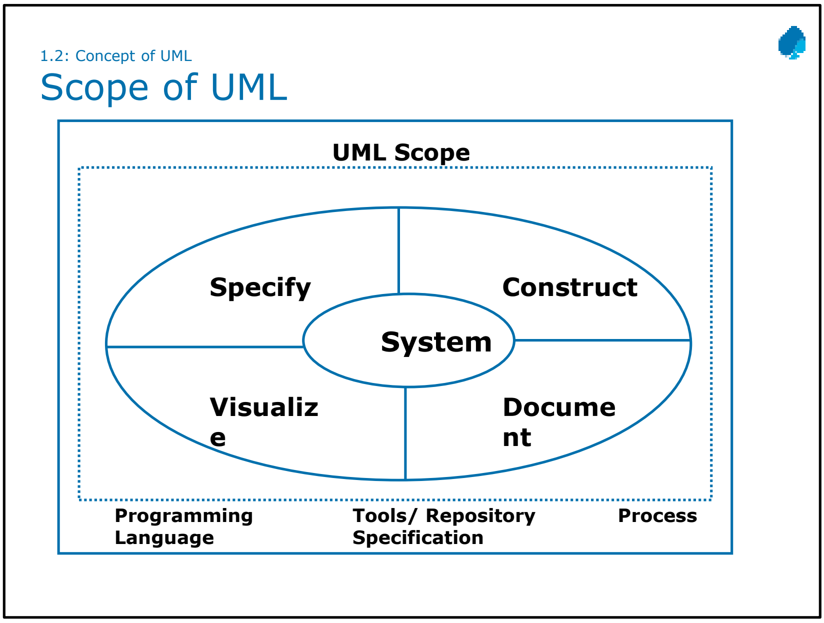
UML Evolution



Here is a figure illustrating how UML has evolved over a period of time. Ver 2.x

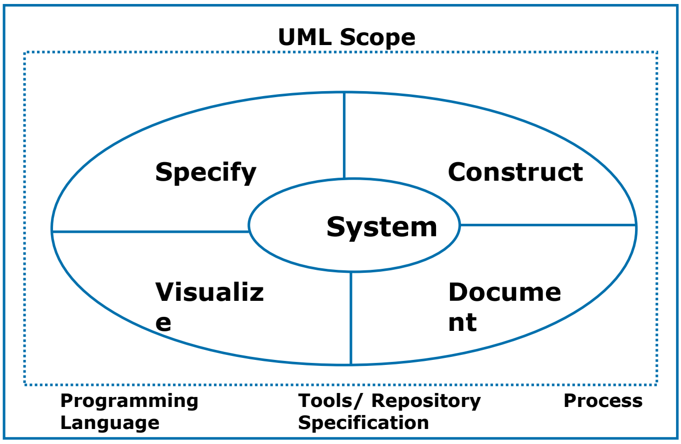
is the current industry standard. We are dealing with Ver 1.4 in our current

course.



1.2: Concept of UML

Scope of UML



Scope of UML:

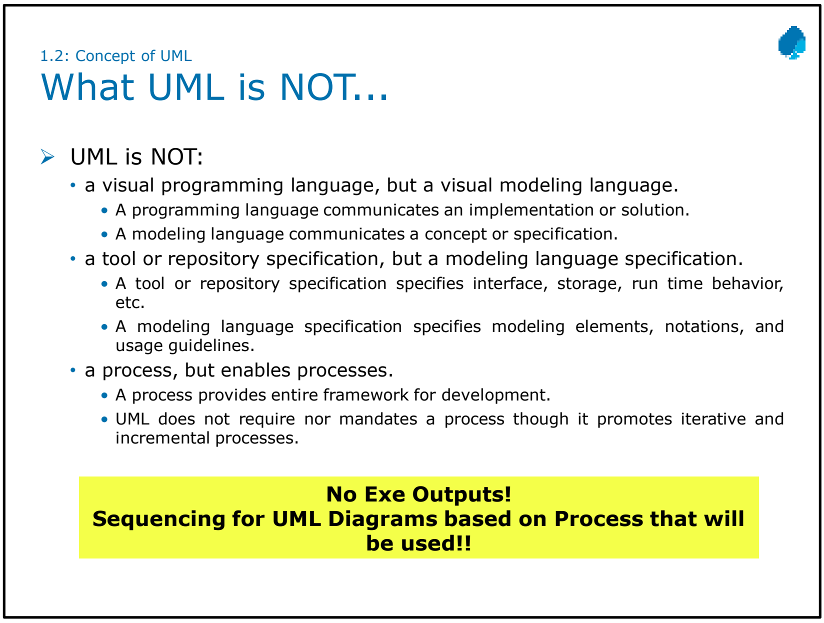
The figure shown in the slide illustrates what is within the scope of UML, and what is outside of the UML scope. While it provides adequate notation and semantics to address contemporary modeling issues, there are some items outside the scope of UML as explained in subsequent slides.

Some of the things that are outside the scope of UML are:

Programming Language: UML is a “visual modeling language”. It is not intended to be a visual programming language. It is a language for visualizing, specifying, constructing, and documenting the artifacts of a software intensive system. It does have a close mapping with OO languages.

Tools: Language standards form the foundation for tools and process. UML defines a semantic meta-model, and not a tool interface, storage, or run-time model.

Process: Software development process will use UML as a common language for its project artifacts, but will use the same type of UML diagram in the context of different processes. UML is process independent.



1.2: Concept of UML

What UML is NOT…

- UML is NOT:
  - a visual programming language, but a visual modeling language.
    - A programming language communicates an implementation or solution.
    - A modeling language communicates a concept or specification.
  - a tool or repository specification, but a modeling language specification.
    - A tool or repository specification specifies interface, storage, run time behavior, etc.
    - A modeling language specification specifies modeling elements, notations, and usage guidelines.
  - a process, but enables processes.
    - A process provides entire framework for development.
    - UML does not require nor mandates a process though it promotes iterative and incremental processes.

No Exe Outputs!
Sequencing for UML Diagrams based on Process that will be used!!

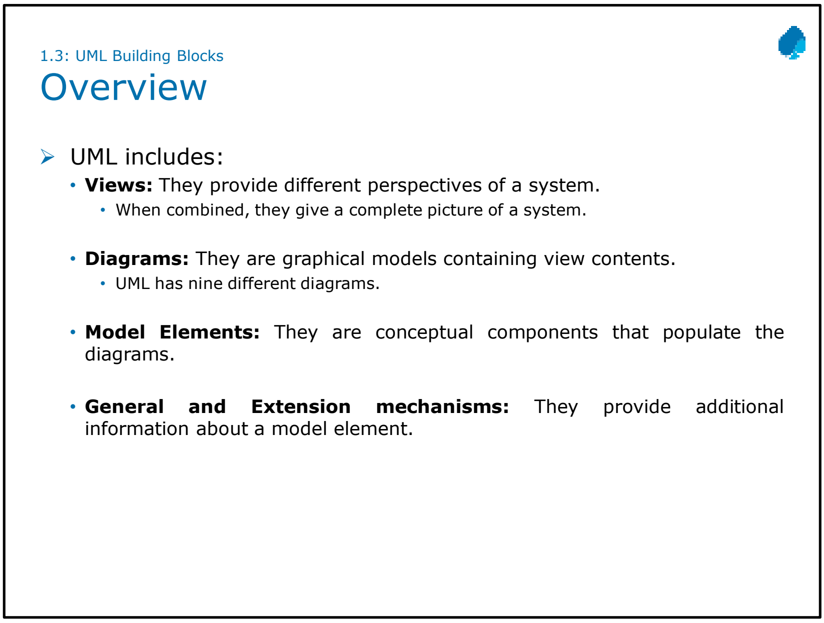
And What UML is NOT …

UML is not meant to be a programming language, rather it is a language meant for modeling. By using UML, one can convey a concept or a specification but not a solution (which a program does).

UML comprises model elements, each with its own associated notation and semantics. UML is not meant to specify a tool or repository in terms of interfaces, storage, or run time behavior.

Similarly, UML is not a process. A process will provide guidance regarding order of activities, and spell out the work products that have to be developed. They are usually domain specific.

UML does not require a process. However, it enables and promotes Object-Oriented and component-based processes.



1.3: UML Building Blocks

Overview

- UML includes:
  - **Views:** They provide different perspectives of a system.
    - When combined, they give a complete picture of a system.
  - **Diagrams:** They are graphical models containing view contents.
    - UML has nine different diagrams.
  - **Model Elements:** They are conceptual components that populate the diagrams.
  - **General and Extension mechanisms:** They provide additional information about a model element.

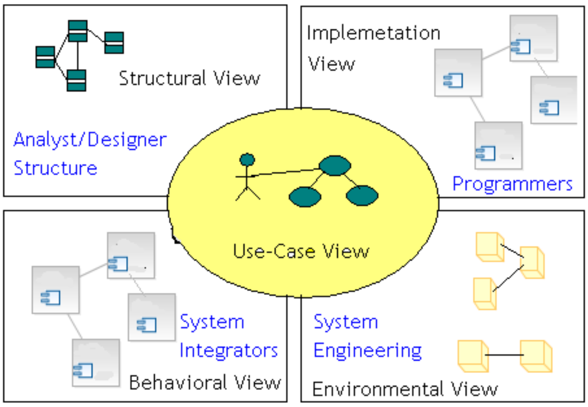
UML Building Blocks:

Let us look at the building blocks of UML. UML offers different views of the system, each view containing different diagrams. The diagrams are made up of specific modeling elements.

In addition to existing modeling elements, UML allows extending available notation and semantics by the use of extension mechanisms.

1.3: UML Building Blocks

Views and Diagrams



UML Building Blocks – Views and Diagrams:

For the end user, the User view is useful to understand the functionality that will be provided by the system. Various views are:

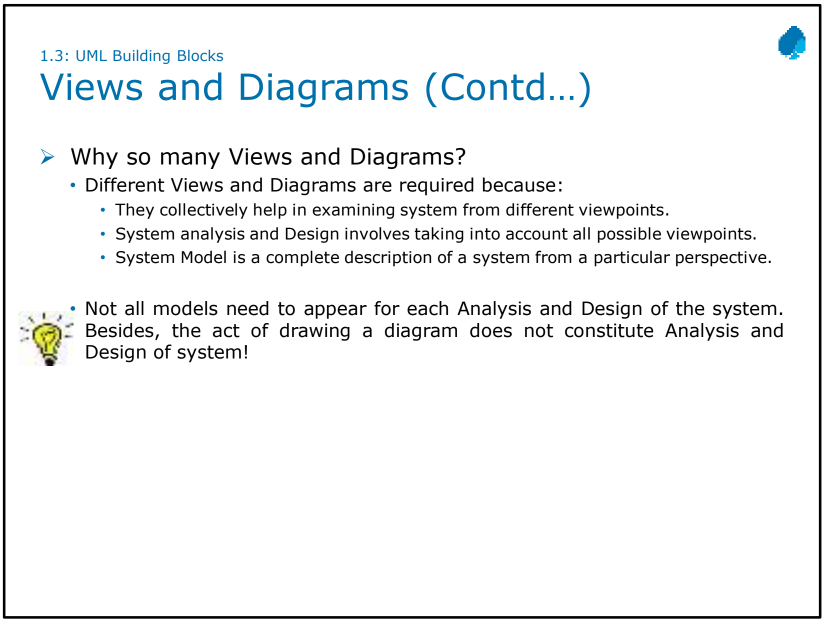
Structural view - Structure diagrams define the static architecture of a model. They are used to model the "things" that make up a model. They are used to model the relationships and dependencies. Analysts and Designers can get the view of the structural aspects of the system through the Structural view.

Behavioral view - It can give important inputs in terms of performance, scalability, and throughput, which can be used by the system integrator.

Implementation view - This view is helpful for programmers.

Environment view – This view can convey decisions relating to system topology, delivery mode, installation, and communication.

contd.



1.3: UML Building Blocks

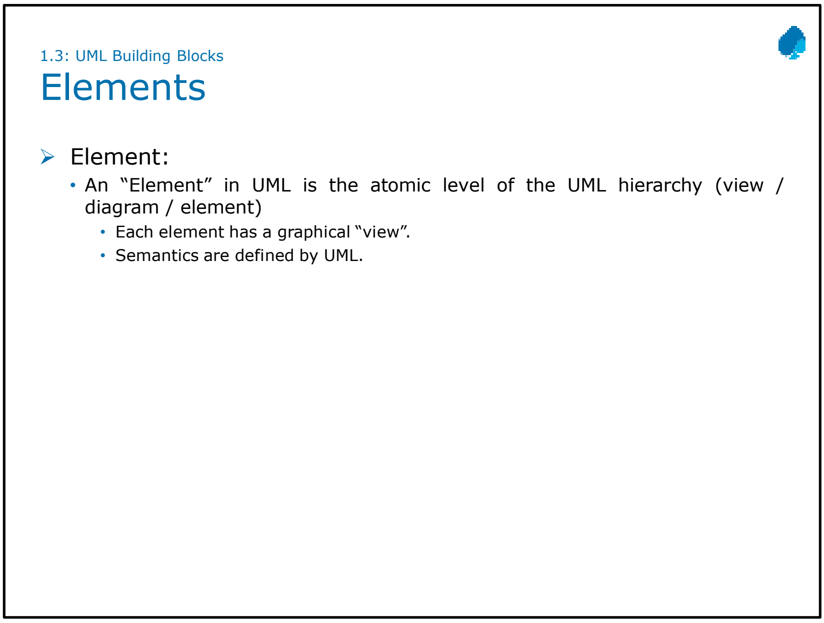
Views and Diagrams (Contd…)

- Why so many Views and Diagrams?
  - Different Views and Diagrams are required because:
    - They collectively help in examining system from different viewpoints.
    - System analysis and Design involves taking into account all possible viewpoints.
    - System Model is a complete description of a system from a particular perspective.
  - Not all models need to appear for each Analysis and Design of the system. Besides, the act of drawing a diagram does not constitute Analysis and Design of system!

UML Building Blocks – Views and Diagrams (contd.):

Often one has to decide which views / diagrams are required for the system under consideration. While deciding on this, consider the “reason for communication” of models. Depending on what aspects of the system need to be emphasized on, the views / diagrams can be chosen. (To that extent, each view / diagram is an independent entity in itself). Hence, it may not be required to have all models for each Analysis and Design.

It is important to note that the activity of drawing diagram by itself is not Analysis and Design. Rather, the diagrams are a means of representing and conveying the “Analysis and Design decisions”.



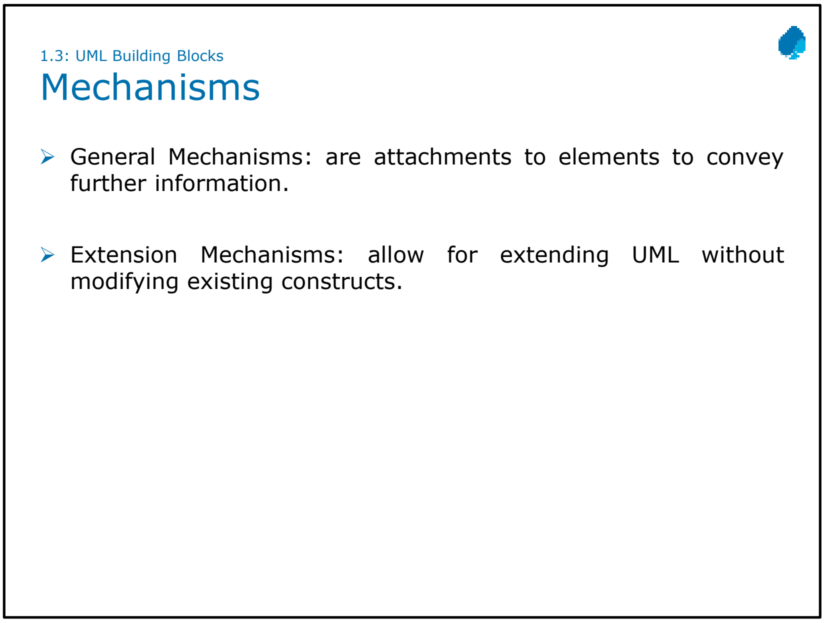
1.3: UML Building Blocks

Elements

- Element:
  - An “Element” in UML is the atomic level of the UML hierarchy (view / diagram / element)
    - Each element has a graphical “view”.
    - Semantics are defined by UML.

UML Building Blocks – Elements:

Elements form the atomic level of the UML hierarchy. Each element has a predefined “meaning” and a “graphical notation” associated with it.



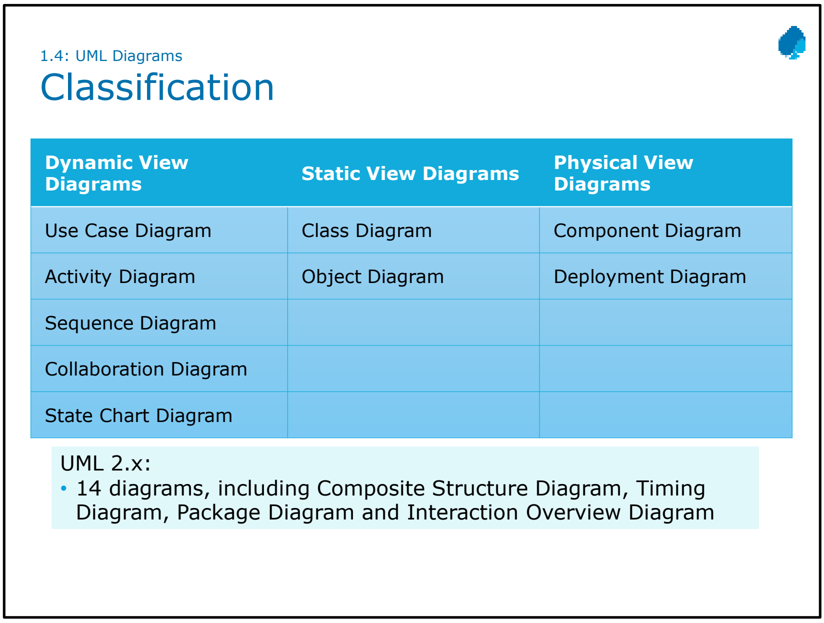
1.3: UML Building Blocks

Mechanisms

- General Mechanisms: are attachments to elements to convey further information.
- Extension Mechanisms: allow for extending UML without modifying existing constructs.

UML Building Blocks – Mechanisms:

There are some mechanisms available to add on to the expressive power of UML. They are broadly categorized as General mechanisms and Extension mechanisms.



1.4: UML Diagrams

Classification

Dynamic View Diagrams	Static View Diagrams	Physical View Diagrams
Use Case Diagram	Class Diagram	Component Diagram
Activity Diagram	Object Diagram	Deployment Diagram
Sequence Diagram		
Collaboration Diagram		
State Chart Diagram		

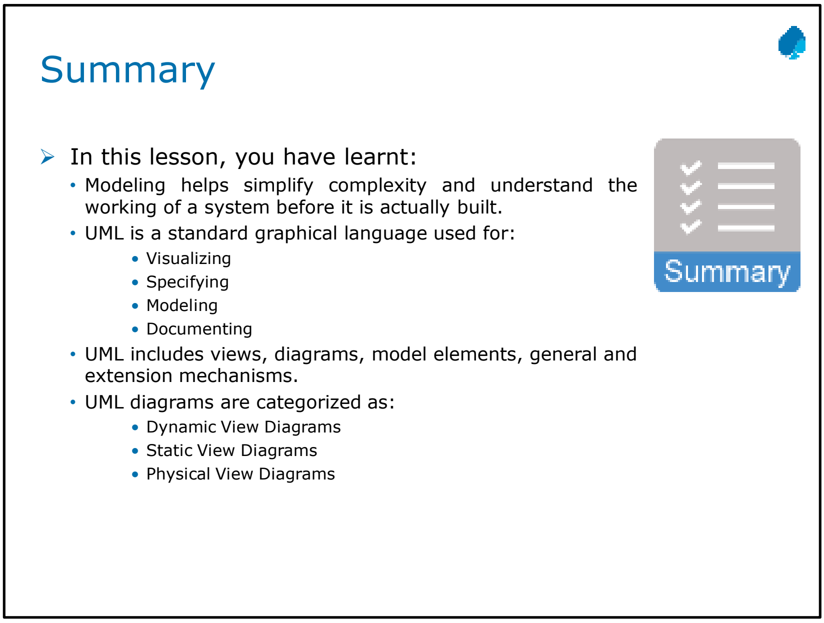
UML 2.x:

- 14 diagrams, including Composite Structure Diagram, Timing Diagram, Package Diagram and Interaction Overview Diagram

UML Diagrams:

The slides shows a list of the nine diagrams in UML 1.4. Also mentions the additional diagrams of UML 2.x.

In the subsequent sections we will be looking at each UML 1.4 diagram in detail. The sections provide the notations and associated semantics for the constituents of each diagram.



Summary

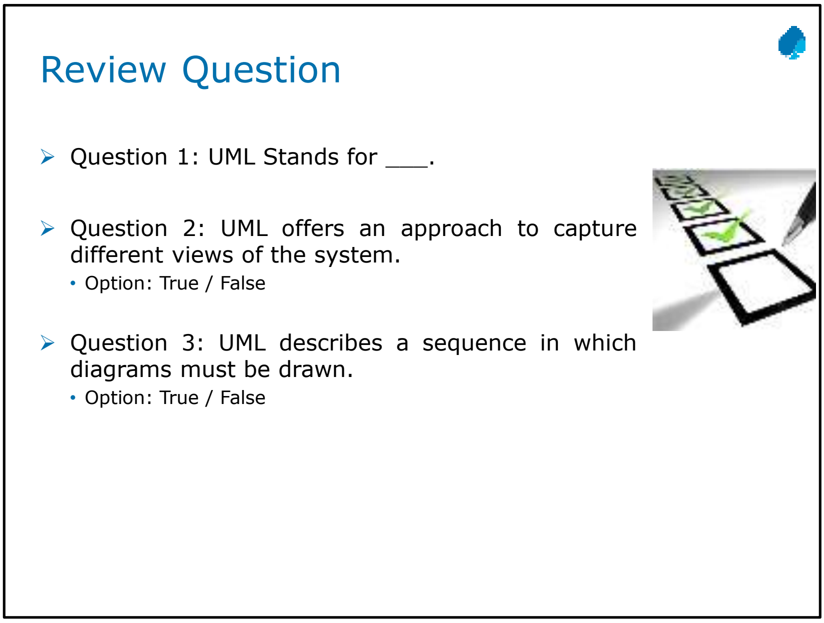
- In this lesson, you have learnt:
  - Modeling helps simplify complexity and understand the working of a system before it is actually built.
  - UML is a standard graphical language used for:
    - Visualizing
    - Specifying
    - Modeling
    - Documenting
  - UML includes views, diagrams, model elements, general and extension mechanisms.
  - UML diagrams are categorized as:
    - Dynamic View Diagrams
    - Static View Diagrams
    - Physical View Diagrams

Answers to review questions.

Question 1: Unified Modeling Language.

Question 2 : True

Question 3: False



Review Question

- Question 1: UML Stands for ___.
- Question 2: UML offers an approach to capture different views of the system.
  - Option: True / False
- Question 3: UML describes a sequence in which diagrams must be drawn.
  - Option: True / False

Answers to review questions.

Question 1:
1- A, D,E
2- C
3-B,F

Review Question: Match the Following

1. Dynamic View Diagrams	A. Use Case Diagram
2. Static View Diagrams	B. Deployment Diagram
3. Physical View Diagrams	C. Class Diagram
	D. Activity Diagram
	E. Sequence Diagram
	F. Component Diagram