Information System Design

IT60105

Lecture 7

Unified Modeling Language

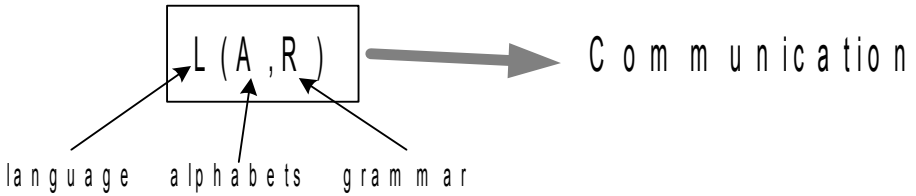
Lecture #07

- **Unified Modeling Language**
  - Introduction to UML
  - Applications of UML
  - UML Definition
  - Learning UML
  - Things in UML
    - Structural Things
    - Behavioral Things
    - Grouping Things
    - Annotational Things
  - Relationships in UML
  - Diagrams in UML

Introduction to UML

Introduction to UML

- UML is an abbreviation of Unified Modelling Language
- UML is a language



  - It has a set of vocabulary (like rectangles, lines, ellipses etc.) and the rules for combining words in that vocabulary for the purpose of communication

  - UML is a graphical language

Introduction to UML

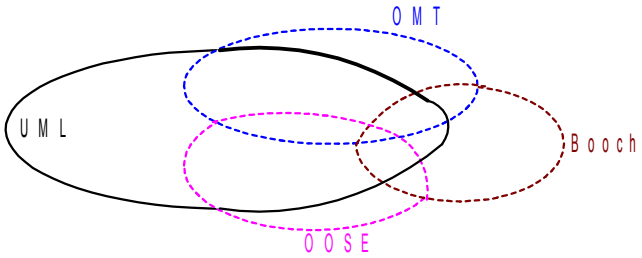
- UML is a modeling language
  - UML is a language to create models (software blue prints) of software intensive systems
  - UML focuses on conceptual and physical representation of a system

Introduction to UML

- UML is a unified modeling language
  - It provides a standard for modeling a system, the standard was derived from previously exercised methodologies such as
    - Booch's Methodology
      by Grady Booch (1991)
    - Object Modeling Technique (OMT)
      by James Rumbaugh (1991)
    - Object Oriented Software Engineering (OOSE)
      by Ivar Jacobson (1992)

Introduction to UML

- UML is a unified modelling language
  - UML is developed by Grady Booch, James Rumbaugh and Ivar Jacobson towards the direction of an unification effort
  - UML was adopted by Object Management Group (OMG) and became a *de facto* standard in 1997



  - Details on UML and its evolution can be seen in www.omg.org

Applications of UML

Applications of UML

- The UML is appropriate for modeling systems ranging from enterprise systems to distributed web based applications and even to hardcore real time embedded systems

- UML follows object oriented approach and hence the best deals with object oriented analysis and design (OOA&D) of systems

Definition of UML

Definition of UML

- **The UML is a language for**
  - Visualizing
  - Specifying
  - Constructing
  - Documenting

  the artifacts of a software-intensive system

Definition of UML : Visualizing

- **Visualizing**

  UML models a system to facilitate communication for all range of people

Definition of UML : Specifying

- **Specifying**

  UML builds models that are precise, unambiguous and complete. In particular UML addresses the specification of all the important analysis, design and implementation decision that must be made in developing and deploying a software system

Definition of UML : Constructing

- **Constructing**
  - UML models can be directly mapped to
    - An object oriented programming language such as C++, Java or Visual Basic
    - Tables in relational database
    - Persistent store of an object-oriented database
  - This permits forward engineering: the generation of a code from a UML model into a programming language

Definition of UML : Constructing

- It also permits reverse engineering: One can reconstruct a model from an implementation back into the UML
  - Reverse engineering requires tools support with human intervention
- Roundtrip engineering: combining the two paths of forward code generation and reverse engineering
  - Meaning the ability to work in either a graphical or technical view
  - UML keeps two views consistent

Definition of UML : Documenting

- **Documenting**
  - UML facilitates all sorts of system artifacts like requirements, design, project planning, coding, testing, prototyping, maintenance, release etc.
  - These artifacts not only the deliverables of a project, they also in controlling, measuring and communicating about a system during its development and after its deployment

Learning UML

Learning UML

- There are three major elements in UML

    1. Basic building blocks (vocabulary of the language)

    2. The rules how these building blocks can be put together

    3. Some common mechanism that can be applied throughout the UML

Basic Building Blocks in UML

- Vocabulary of the UML encompasses three kinds of building blocks
  - Things
    - Things are abstractions in a model
  - Relationships
    - Relationships tie the things together
  - Diagrams
    - Diagrams group relevant collection of things

Basic Building Blocks: Things

- There are four kinds of things in the UML

    1. Structural things
    2. Behavioral things
    3. Grouping things
    4. Annotational things

Structural Things in UML

Structural Things in UML

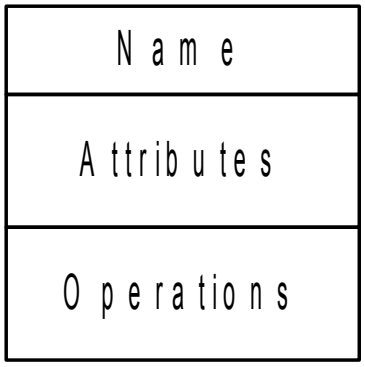
- Relevant things that are either conceptual or physical
  Examples: class, object, etc.

- Structural things are nouns in the UML model

- There are seven structural things in UML
  1. Class
  2. Active class
  3. Component
  4. Interface
  5. Use case
  6. Collaboration
  7. Node

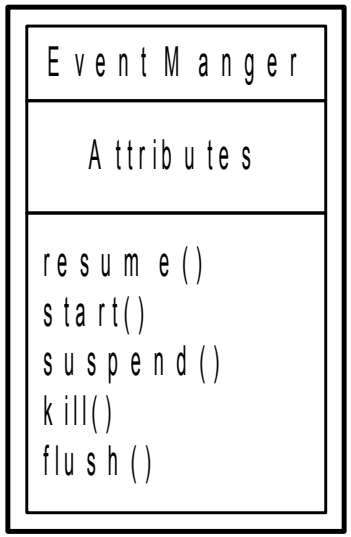
Structural Things: Class

- **Class**
  - A class is a description of a set of objects (or an abstraction of object)
  - Graphically a class is represented as a rectangle, including its name, attributes, and operations



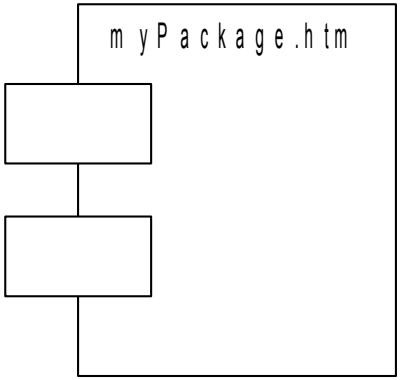
Structural Things: Active Class

- **Active Class**
  - An active class is just like a class expect that its object represent element whose behavior is concurrent with other elements
  - An object of an active class owns one or more processes or threads and therefore initiate control activity
  - Graphically, an active class is denoted just like a class, but with a heavy line, usually including its name, attributes, and operations



Structural Things: Component

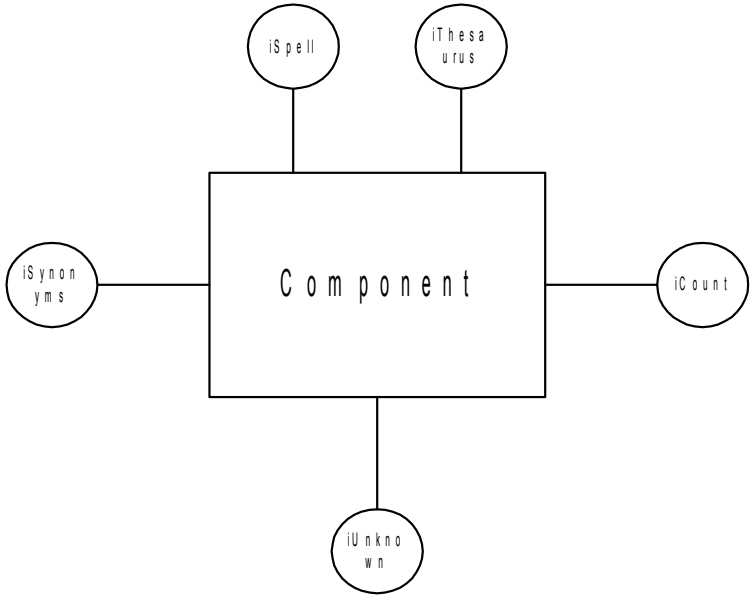
- **Component**
  - Component of a system is to conform with the realization of a set of interfaces
  - Example: An applet, com+ component, Java bean or ASP file. In fact, a component is typically represents a physical package of otherwise logical elements such as classes, interfaces and collaborations
  - Graphically a component denoted as a rectangle with tabs, usually including only its name



Structural Things: Interface

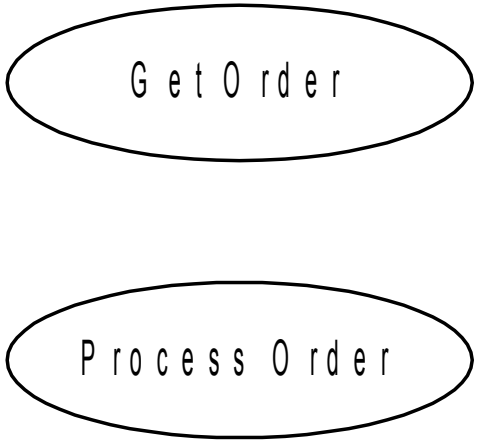
- **Interface**
  - An interface is a collection of operators that specify a service of a class or a component
  - In UML, Interfaces are used to model the seams (layers) in a system
  - Graphically, an interface rendered as a circle together with its name

Note: An interface rarely stands alone, rather it is attached to a class or component



Structural Things: Use case

- **Use case**
  - Use case represents a functional components in a model
  - Graphically a use case is represented by an ellipse including only its name



Structural Things: Collaboration

- **Collaboration**
  - A collaboration names a society of classes, interfaces, and other components that work together to provide cooperative behavior that is bigger than the sum of its individual parts

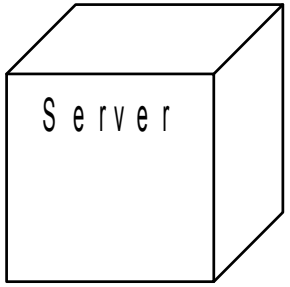
    Example:

    A big building (world trade center) is a collaboration of some structural construction, electrical wiring system, telephone connectivity, internet connectivity, water supply system, etc.

  - Graphically a collaboration is rendered as an ellipse with dashed line

Structural Things: Node

- **Node**
  - Node is a physical element and represent a computational resource, generally having same memory and processing compatibility
  - Graphically, a node is denoted by a cube, usually includes its name



Behavioral Things in UML

Behavioral Things in UML

- Unlike structural things, behavioral things are to represent dynamic parts of UML models. Structural things are the mostly static parts of the model and are the nouns of the UML model

- In the contrary, behavioral things are the verbs of UML models

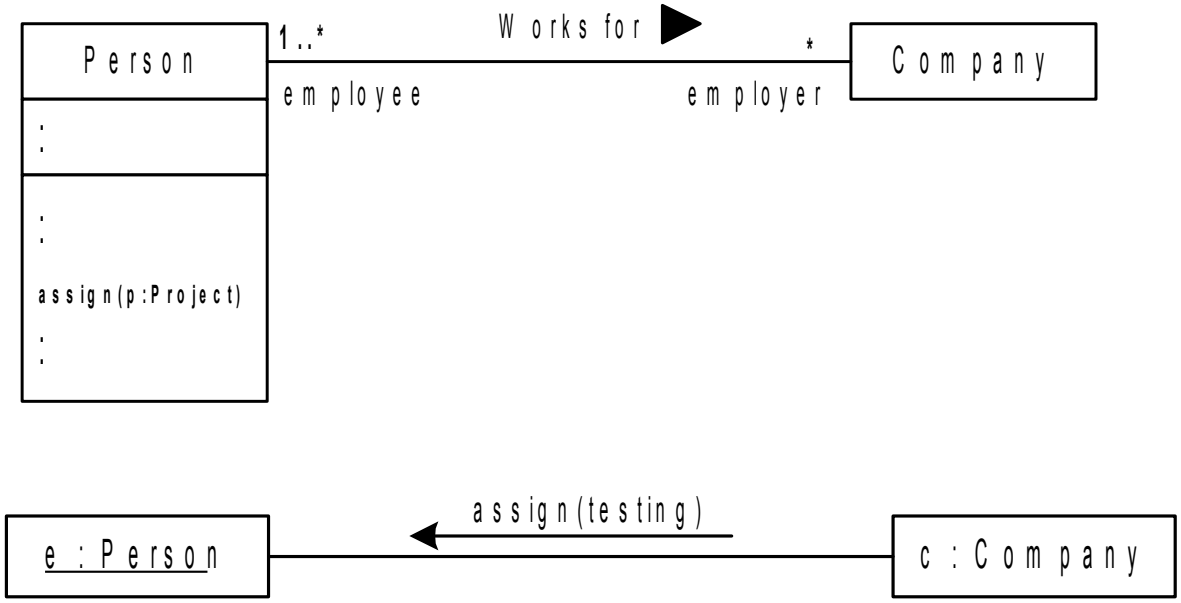
- There are two primary kind of behavioral things in the UML
  1. Interaction
  2. State machine

Behavioral Things: Interaction

- **Interaction**
  - An interaction is a behavior that comprises a set of messages exchanged among set of objects within a context to accomplish a purpose
  - Basically interaction used to link two instances of classes (objects), and through which one object can send a message to the other object

Behavioral Things: Interaction

- Graphically, an interaction is represented as a directed line, usually including the name of its operation

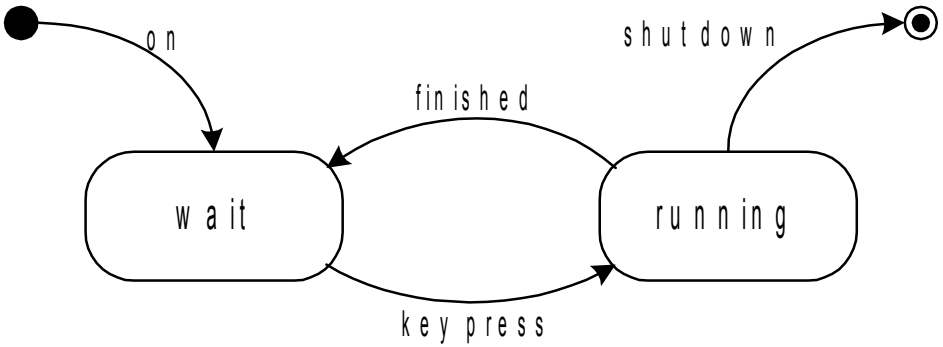


Behavioral Things: State Machine

- **State machine**
  - Using an interaction one can model the behavior of a society of objects that work together
  - On the other hand, using a state machine one can model the behavior of an individual object
  - A state machine is a behavior that specifies the sequence of states of an object goes through during its life time in response to a message

Behavioral Things: State Machine

- Graphically a state is denoted by a rounded rectangle, including its name and its states, if any



A state machine for an object
Button

- **Note:** These two behavioral things are usually connected though various structural elements like classes, collaborations, objects etc.

Grouping Things in UML

Grouping Things in UML

- Visualizing, specifying, constructing and documenting large systems involves manipulating potentially large members of classes, interfaces, components, nodes, diagrams, and other elements

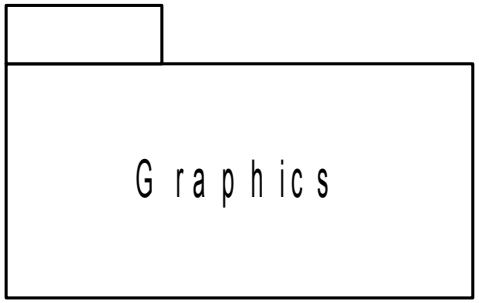
- It is therefore necessary to organize these things into larger chunks

- In UML, grouping things have been planned for this purpose

- In all, there is one primary kind of grouping things, namely, package

Grouping Things: Package

- **Package**
  - A package is a general purpose mechanism for organizing elements into groups
  - Graphically, package is rendered as a tabbed folder, usually including only the name, sometimes references to its contents



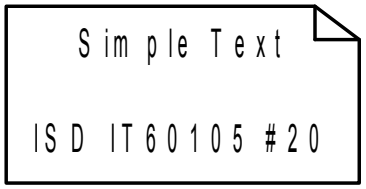
Annotational Things in UML

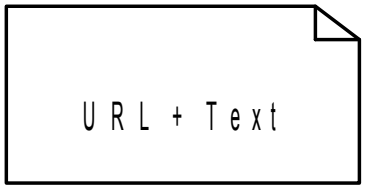
Annotational Things in UML

- Annotational things are the explanation parts of UML models
- Designers follows these to describe, illuminate, and remark about any elements in a model
- There is one primary kind of annotational things called note
- A note is simply a symbol for renders a comment has no semantic impact, means, its contents has no effect on models

Annotational Things: Note

- Graphically a note is rendered as a rectangle with a “dog earned ” corner, together with a textual or graphical comment





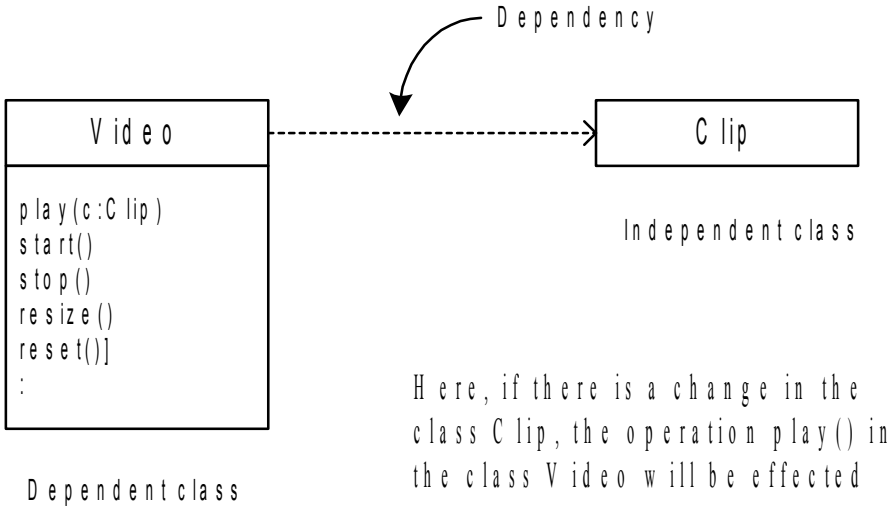
Relationships in UML

Relationships in UML

- A relationship is a concern among things
- There are four relationships in the UML
    1. Dependency
    2. Association
    3. Generalization
    4. Realization

Relationships: Dependency

- **Dependency**
  - A dependency is a semantic relation between two things in which a change to one thing (the independent thing) may affect another thing that use it , but not necessarily the reverse



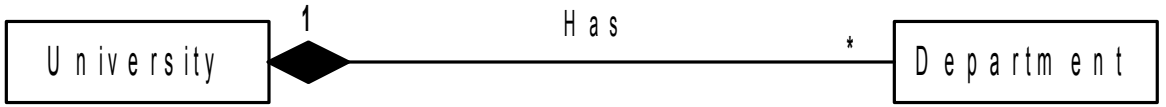
Relationships: Association

- **Association**
  - An association is a structural relationship that describe a set of links, a link being a connection among objects
  - Aggregation is a special kind of association, representing a structural relationship between a whole and its parts

Relationships: Association

- Graphically, association/aggregation is denoted with solid/diamond-edged line with label and multiplicities at both ends of line

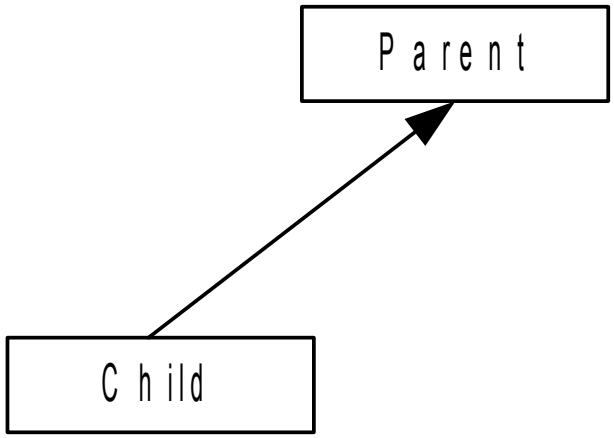
Association



Aggregation

Relationships: Generalization

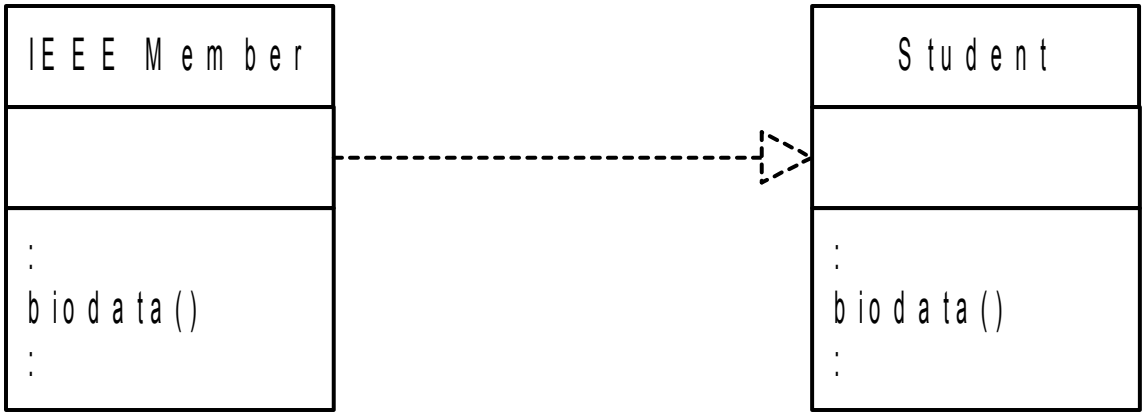
- **Generalization**
  - A generalization is a generalized/specialized relationship in which objects of the specialized elements (the child), are suitable for objects of the generalized elements (the parent)



Relationships: Realization

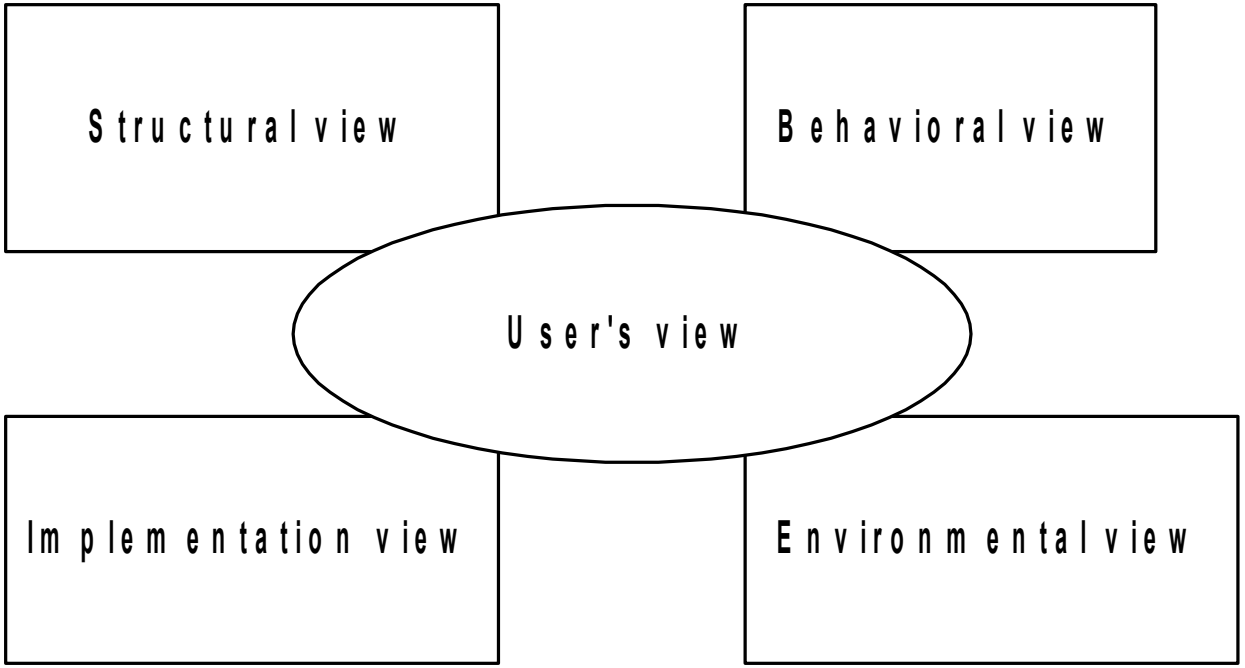
- Realization
  - A realization is a semantic connection
    - between two things: interface, class, component
    - between use-case and collaboration

    etc.



Diagrams in UML

- Any system can be viewed with five **interrelated** views



Diagrams in UML

- To represent each view, UML provides nine diagrams, popularly termed as UML diagrams or UML artifacts