Domain Model and Class Diagram

Classes and Objects

- What are Class Diagrams?
  - Class
  - Property (Attributes)
  - Operation (Methods)
  - Examples

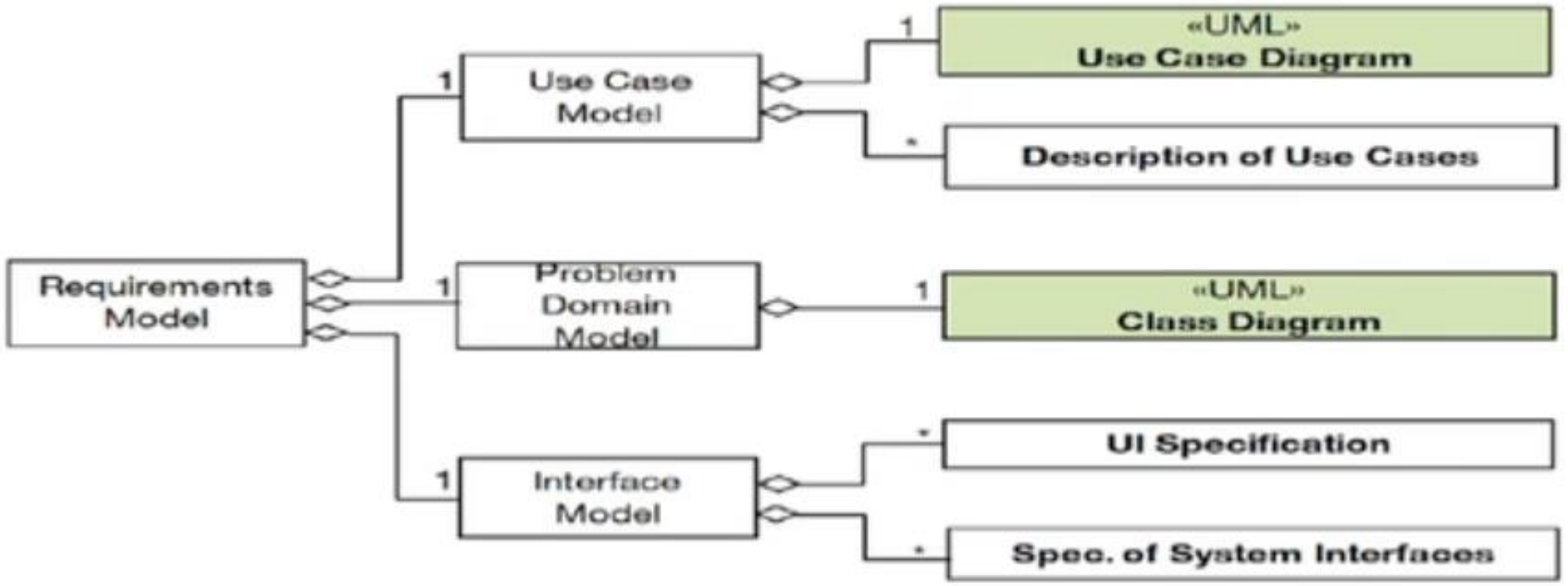
Whereas an object is a concrete entity that exists in time and space, a class represents only an abstraction, the "essence" of an object, as it were

- For a class Faculty, objects may be:
  - {Partha Pratim Das, Professor, CSE}
  - {Prabir Kumar Biswas, Professor, ECE}
  - {Shyamal Das Mondal, Assistant Professor, CET}
- Class Faculty abstracts – *Name*, *Designation*, and *Department*

A class is a set of objects share a common stru... common behavior, and common semantics

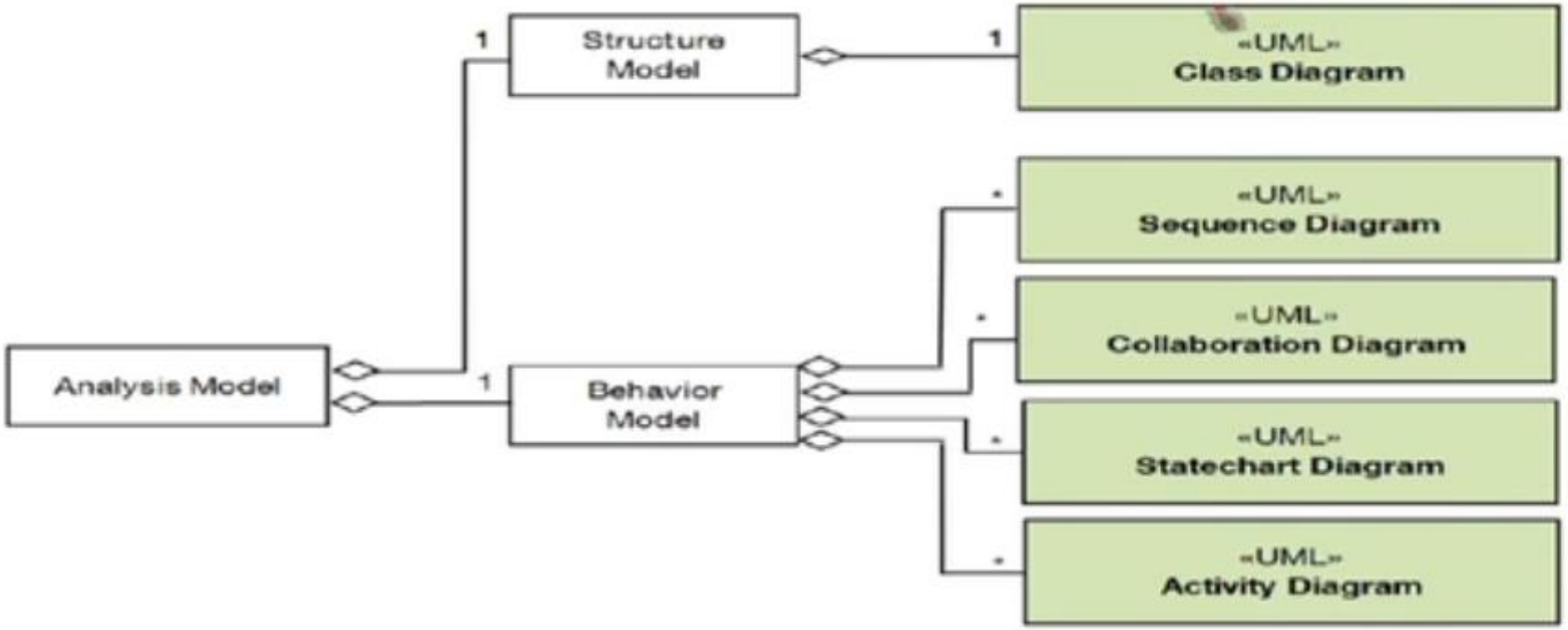
A single object is simply an instance of a class

Class Diagram



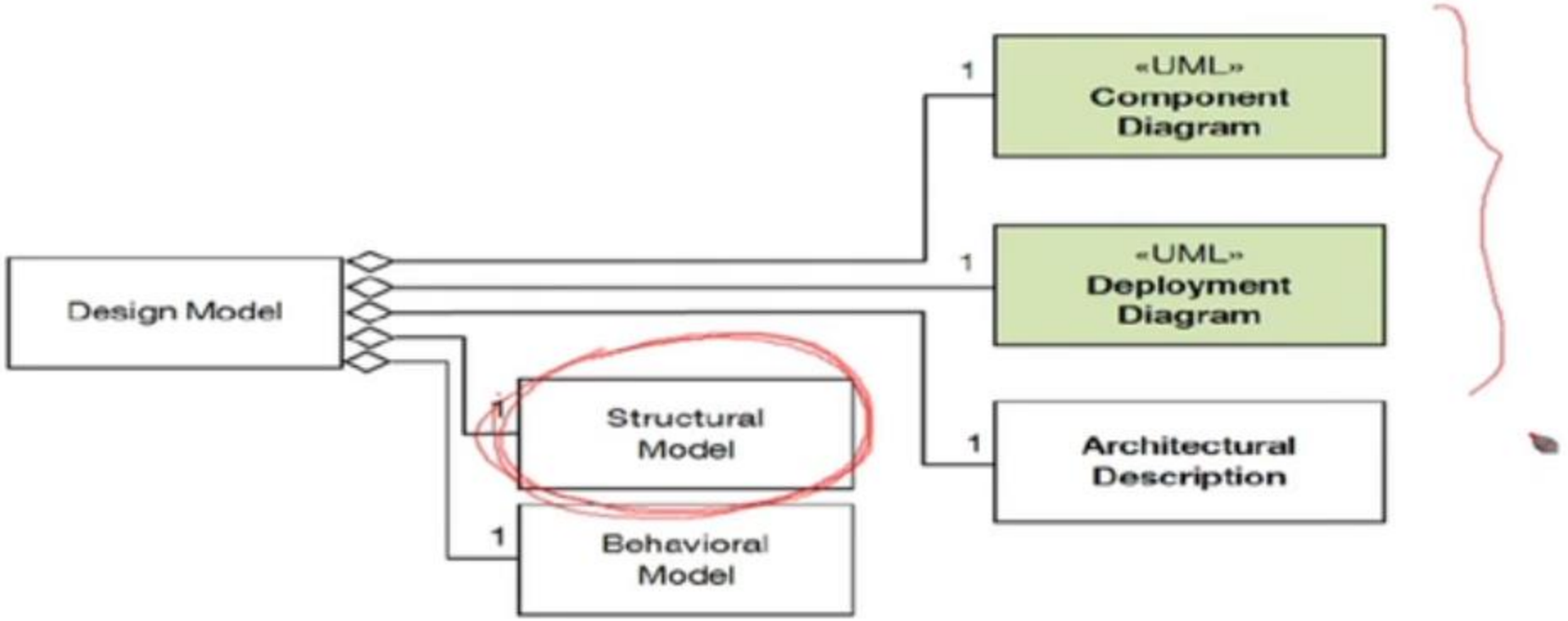
- In the Requirements Phase, the class diagram is used to identify the major abstractions
- At this stage the attributes and operation of each abstraction may not be known
- Classes are identified as domain models

Class Diagrams



- After analysis of each abstraction, attributes and operation of each abstraction is known
- Hence the class diagram in the Analysis Phase is more detailed
- Classes are refined as domain models

Class Diagrams



- Class diagram is included in the Structural Model
- In the Design Phase is further detailed
- As we engage in HLD to LLD, implementation classes are added

Class Diagrams and Features of a Class

- Class diagram is UML structure diagram which shows structure of the designed system at the level of classes and interfaces, shows their features, constraints and relationships – associations, generalizations, dependencies, etc.
- Some common types of class diagrams are:
  - Domain model diagram
  - Diagram of implementation classes

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (17-Aug-16)

- **Non Static Features:** characterizes individual instances of class
- **Static Features:** represents some characteristic of the class itself
- **Structural Features (attributes):** is a typed feature of a class that specifies the structure of instances of the class
- **Behavioral Features (Methods):** is a feature of a class that specifies an aspect of the behavior of its instances

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (17-Aug-16)

Notation of a Class

- Class name should be centered and in bold face inside a solid-outline rectangle, with the first letter of class name capitalized

Class Student - details suppressed

- Abstract Classes (which cannot be instantiated) have the keyword abstract mentioned within { }

Abstract Class Teacher - details suppressed

- A class has optional compartments separated by horizontal lines containing attributes and methods in order

Notation for Property (Attributes)

- **Property (Attributes) specification format:**

  Visibility PropertyName : Type [Multiplicity] = DefaultValue
  {Property string}

  - The visibility of the properties are denoted by +(public), #(protected) and −(private)
  - PropertyName is underlined if the Property is static
  - A property may be *Read Only*, *Static*, *Ordered*, *Unique* or *Optional* (to indicate allowable null value)
  - Property could have multiplicity. The multiplicity bounds constrain the size of the collection of property values. By default the maximum bound is 1
  - The default-value option is an expression for the default value or values of the property
  - A derived Property, designated by a preceding /, is one that can be computed from other properties, but doesn't actually exist

Notation for Operation (Methods)

- **Operation (Methods) specification format:**

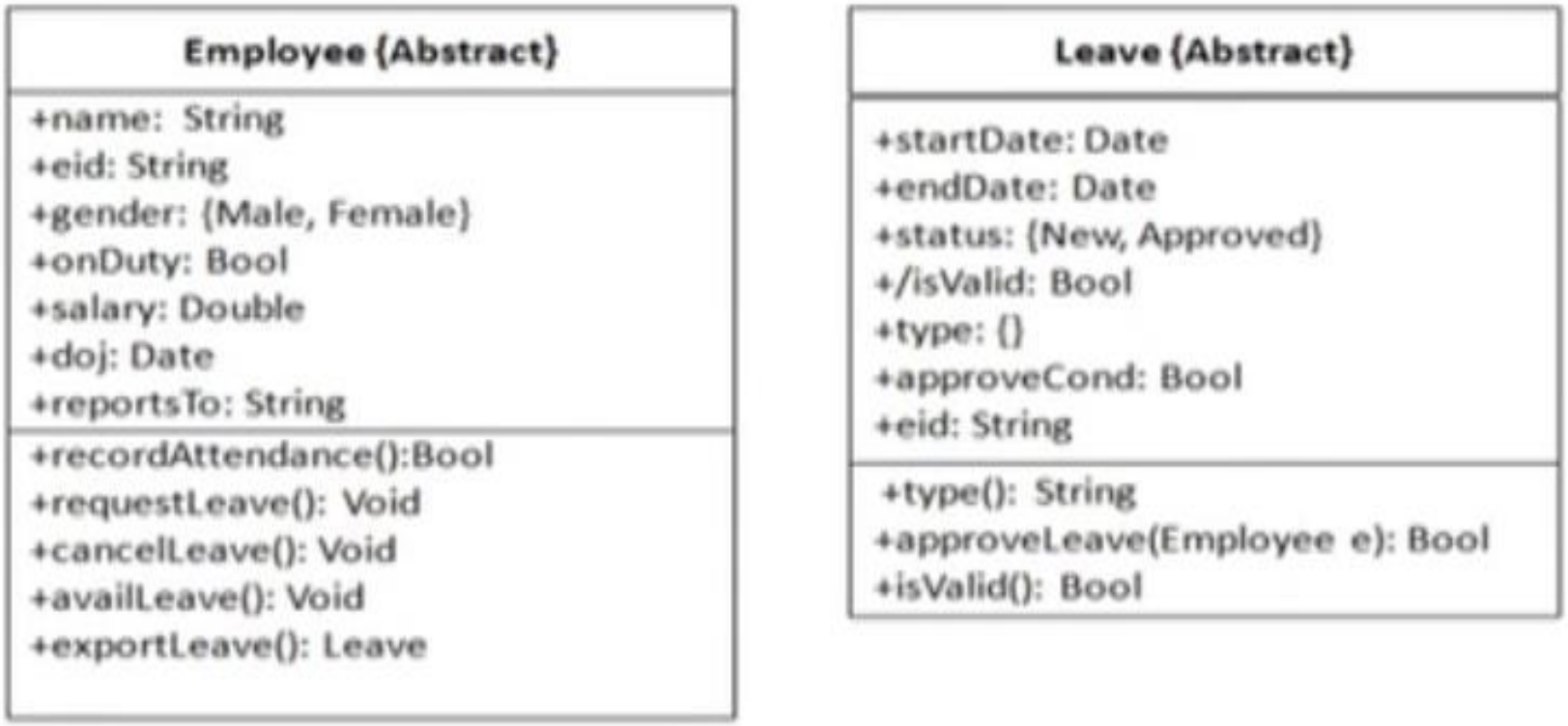
  Visibility OperationName (ParameterName : Type) : ReturnType {Property string}

  - The visibility of the operations are denoted by +(public), #(protected) and −(private)
  - OperationName is underlined if it is Static, and is italic if it is Abstract
  - Return type is optional
  - An operation may be *Read Only*, *Static*, *Ordered*, *Unique*, *Abstract*, *Sequential*, *Guarded* or *Concurrent*

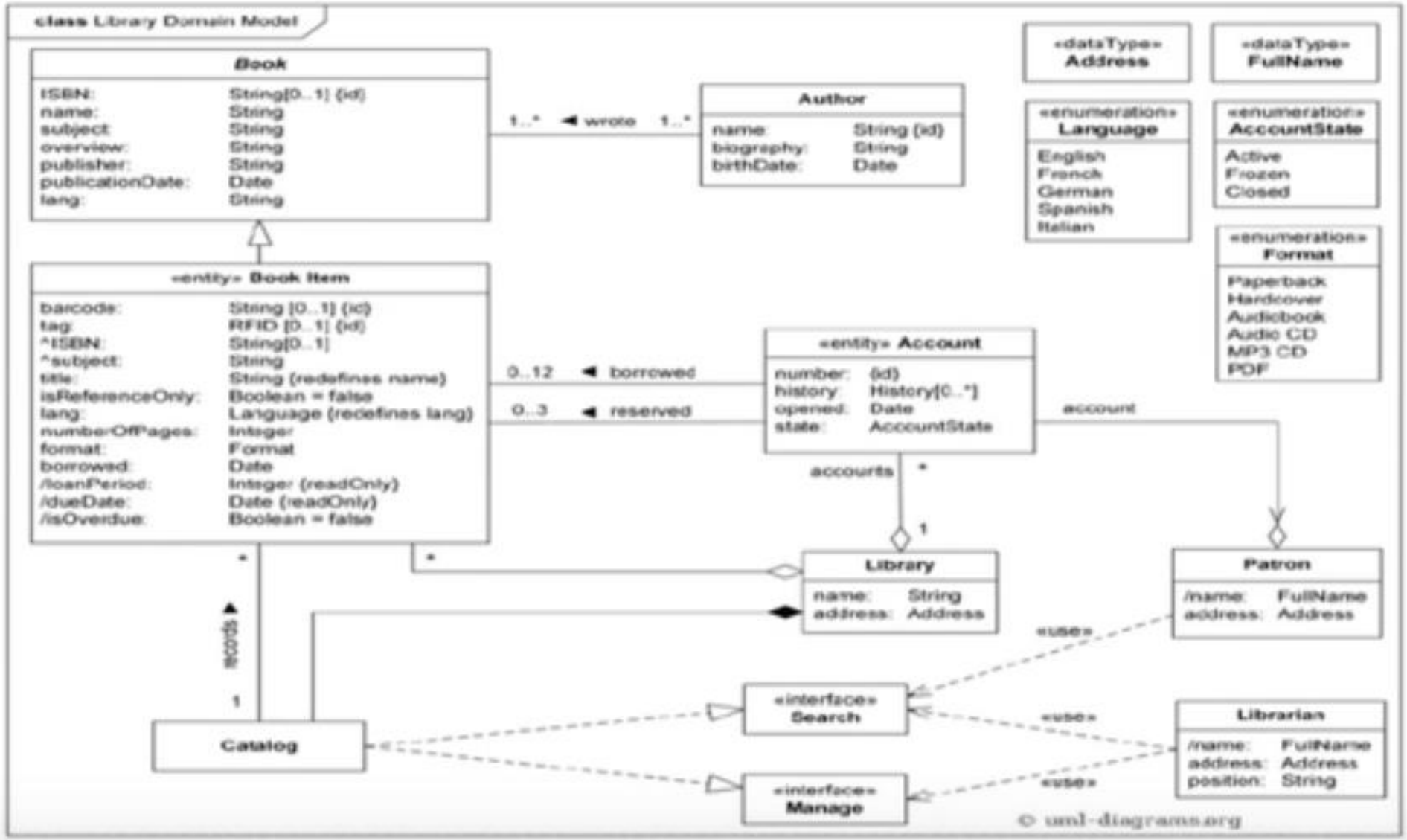
Student
+name: String +date_of_birth: Date +roll_no: String unique +/age: Integer +subject: Subject[1..*]
#recordAttendance(): bool +getCertificates(): Certificates[*] {unique, ordered} −changeSubject(Subject s): bool +calculateAge(): Integer +bookMusicClassSlots (): bool {concurrent}

Abstract Classes of Leave Management System

- We represent below the two abstract classes of LMS

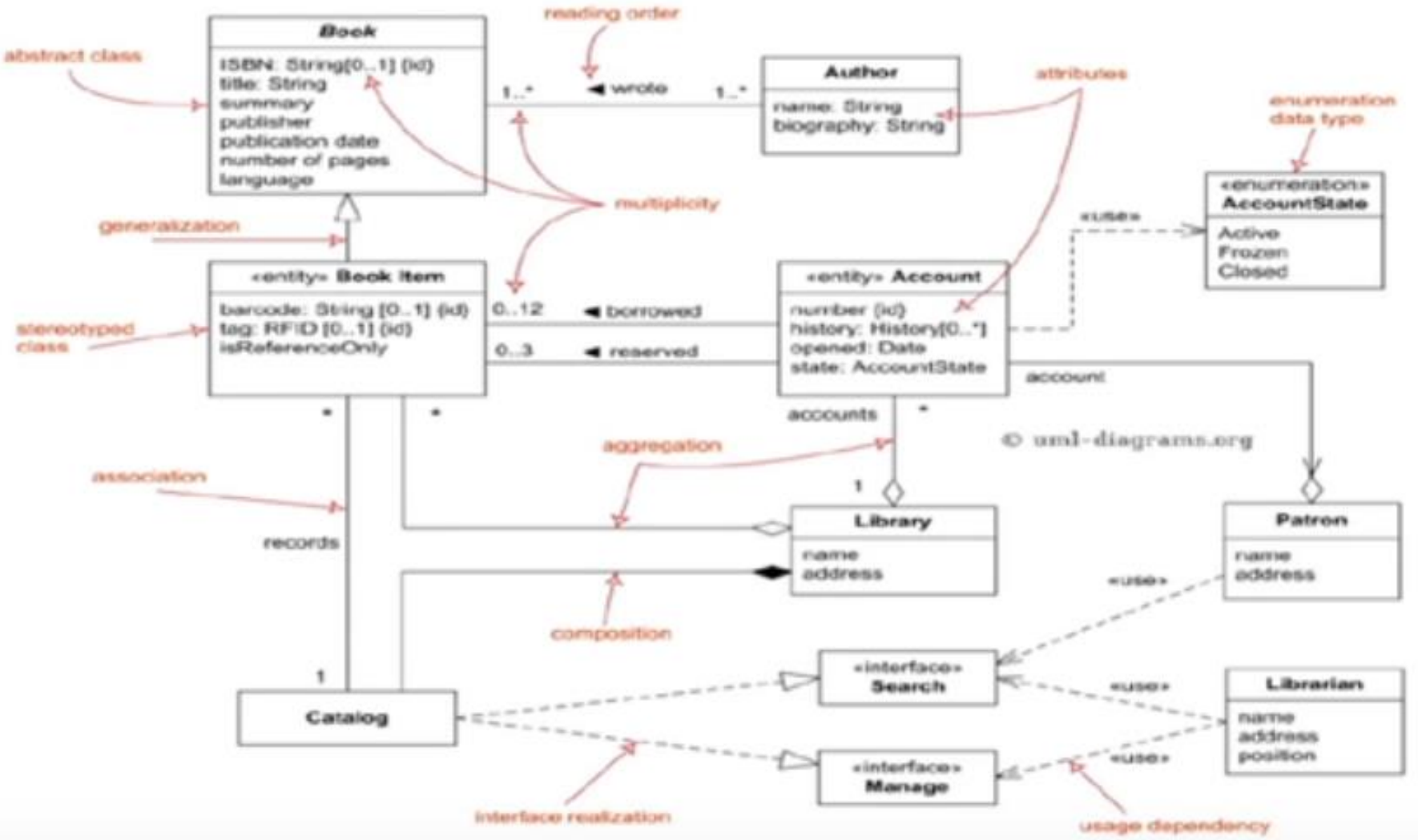


Domain Model for Library Management System



Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (17-Aug-16)

Domain Model for Library Management System



Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (17-Aug-16)

Summary

- Class diagrams are introduced
- Representations for properties and operations are discussed
- An example is used for detailed illustration

Reference

- Source: NPTEL **Object-Oriented Analysis and Design, by**

Prof. Partha Pratim Das Prof. Samiran Chattopadhyay Prof. Kausik Datta

IIT Kharagpur

- https://nptel.ac.in/courses/106105153