Module 26: Object Oriented Analysis & Design

Class Diagrams: Part 1 (Class, Property & Operation)

Partha Pratim Das

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

ppd@cse.iitkgp.ernet.in
Tanwi Mallick
Srijoni Majumdar
Himadri B G S Bhuyan

This presentation uses diagrams, examples and selected texts from *Object-Oriented Analysis and Design – With Applications* by Grady Booch et. al. (3rd Ed, 2007) with kind permission of the author

Module Objectives

- Understanding Class Diagrams

Module Outline

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

What is a Class?
RECAP (Module 13)

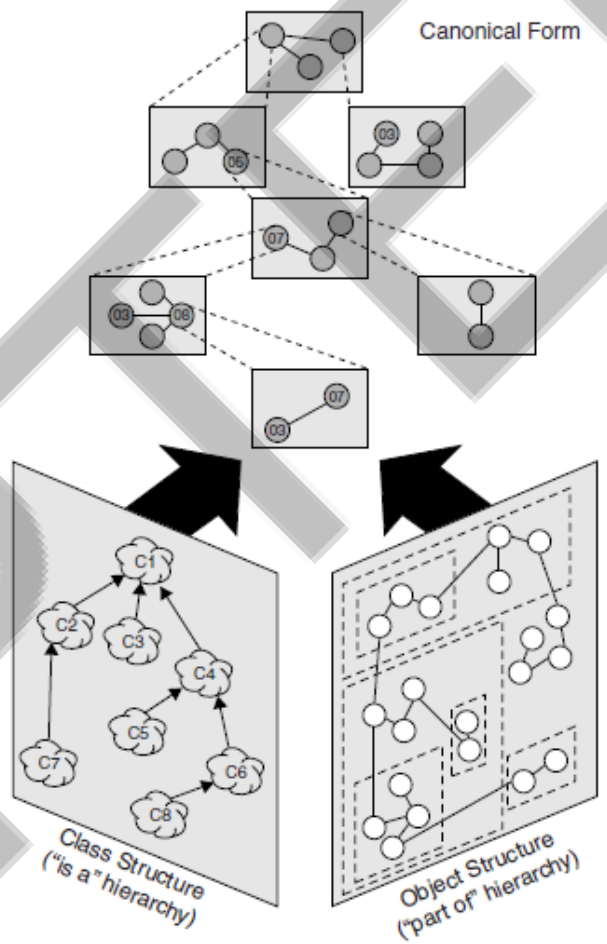
Whereas an object is a concrete entity that exists in time and space, a class represents only an abstraction, the "essence" of an object, as it were

- 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 structure, common behavior, and common semantics

A single object is simply an instance of a class

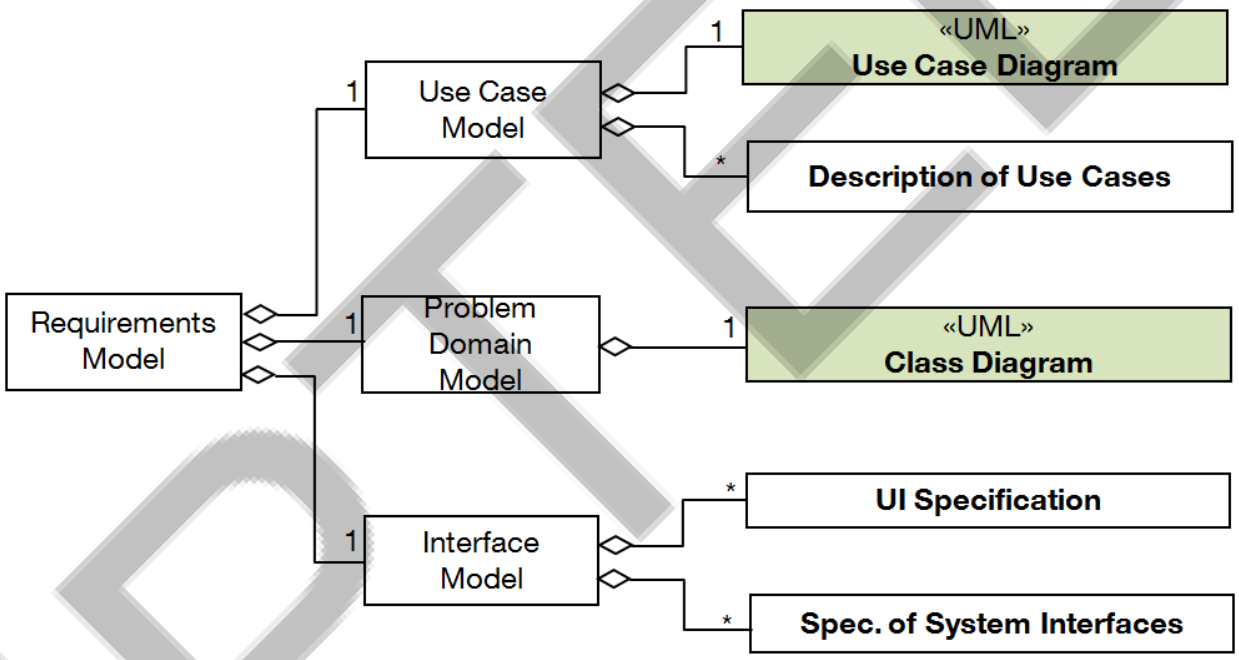
The Canonical Form of a Complex System: RECAP (Module 04)



The Canonical Form of a Complex System

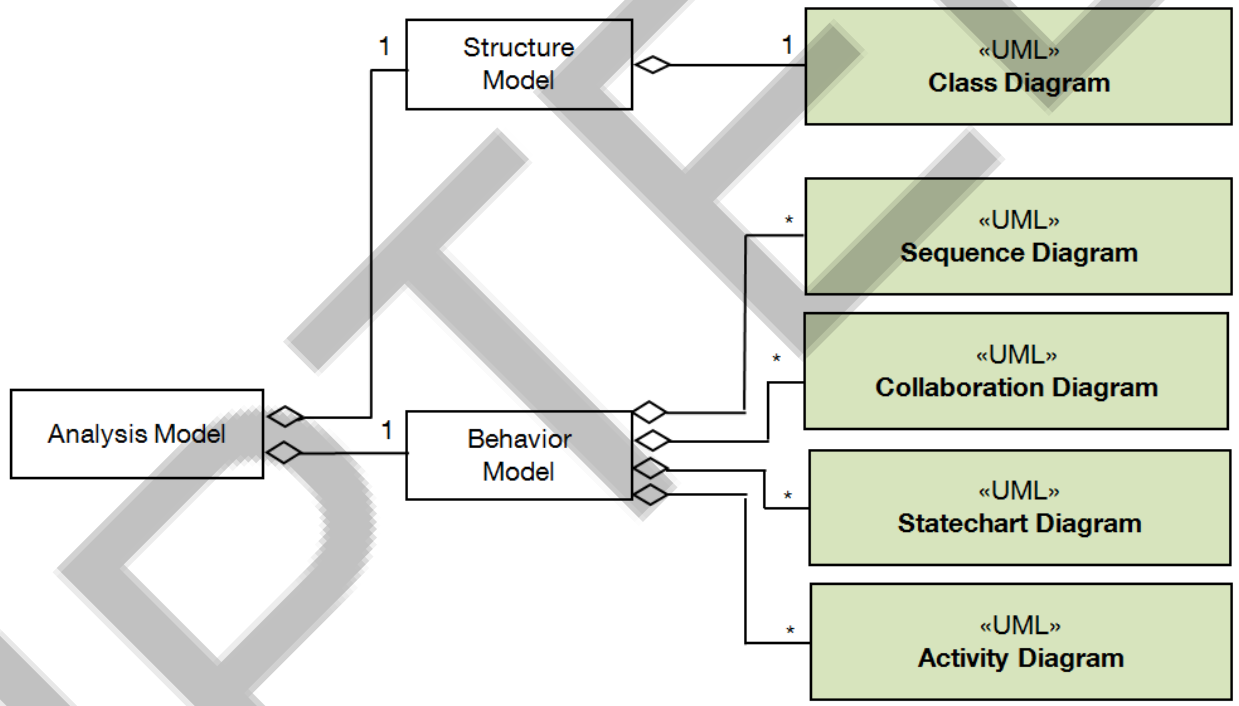
Source: *Object-Oriented Analysis and Design – With Applications* by Grady Booch et. al. (3rd Ed, 2007)

Class Diagrams in SDLC phases: RECAP (Module 22)



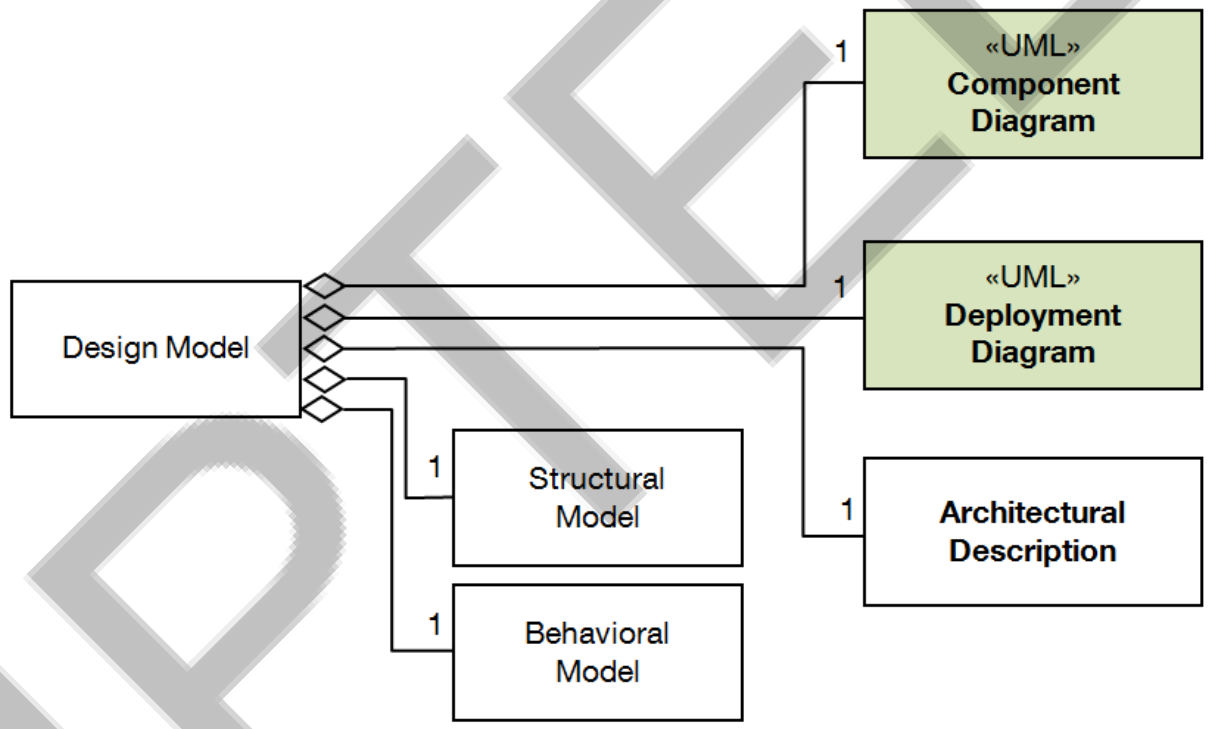
- In the Requirements Phase, the class diagram is used to identify the major abstractions
- At this stage the attributes and operation of each abstraction may not be known
- Classes are identified as domain models

Class Diagrams in SDLC phases: RECAP (Module 22)



- 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 in SDLC phases: RECAP (Module 22)



- 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 Diagram

- 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)

Features of a class

- **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

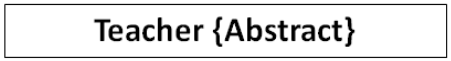
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (17-Aug-16)

Notation for 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

Student
+ name: String +date_of_birth: Date +roll_no: String {unique} +/age: Integer +subject: Subject[1..*]

Notation for Operations (Methods)

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

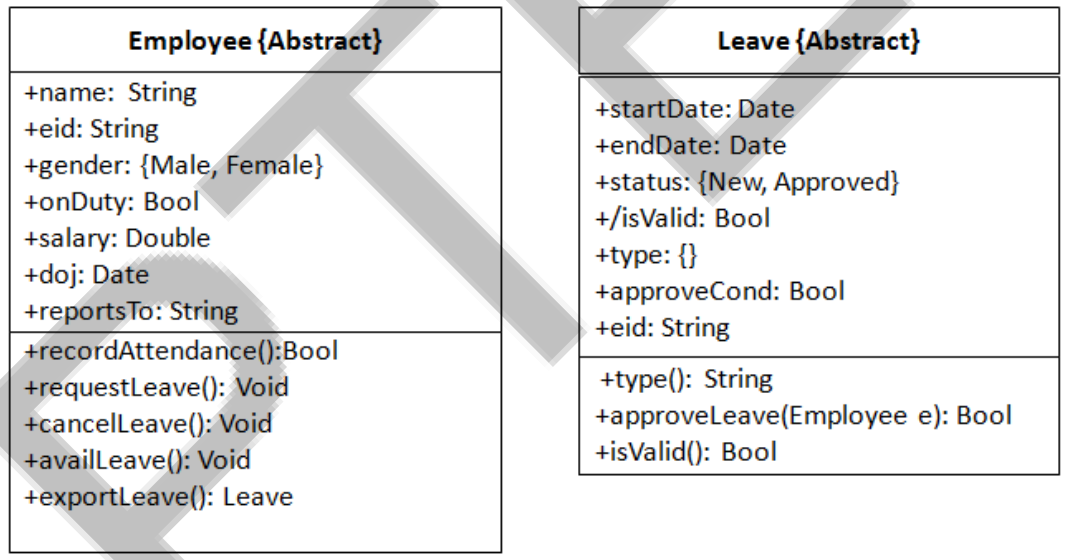
  Visibility OperationName (ParameterName : Type) : ReturnType {Property string}

  - 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*

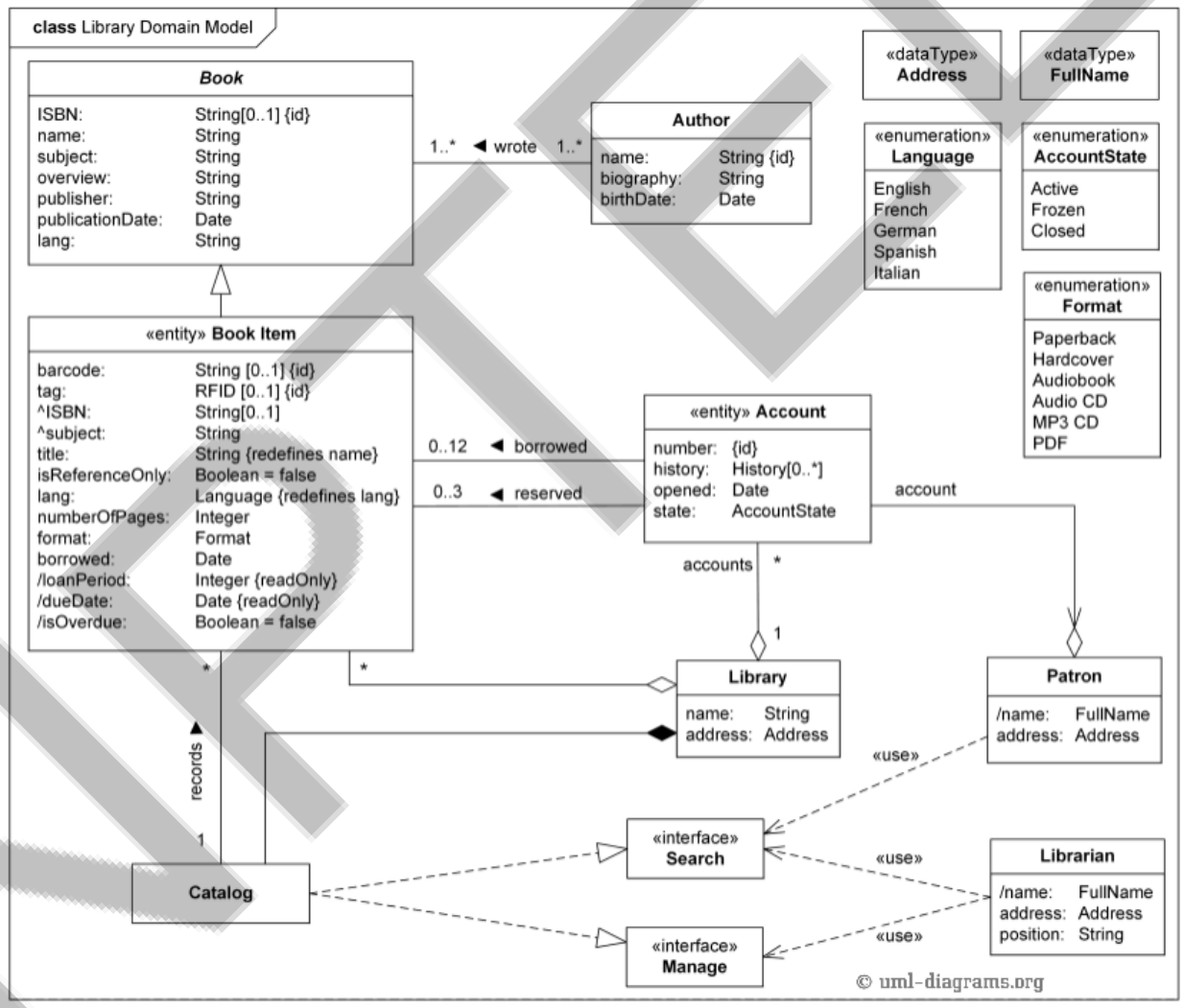
Student
+name: String +date_of_birth: Date +roll_no: String unique +/age: Integer +subject: Subject[1..*]
#recordAttendance(): bool +getCertificates(): Certificates[*] {unique, ordered} −changeSubject(Subject s): bool +calculateAge(): Integer +bookMusicClassSlots (): bool {concurrent}

Abstract Classes of LMS

- We represent below the two abstract classes of LMS

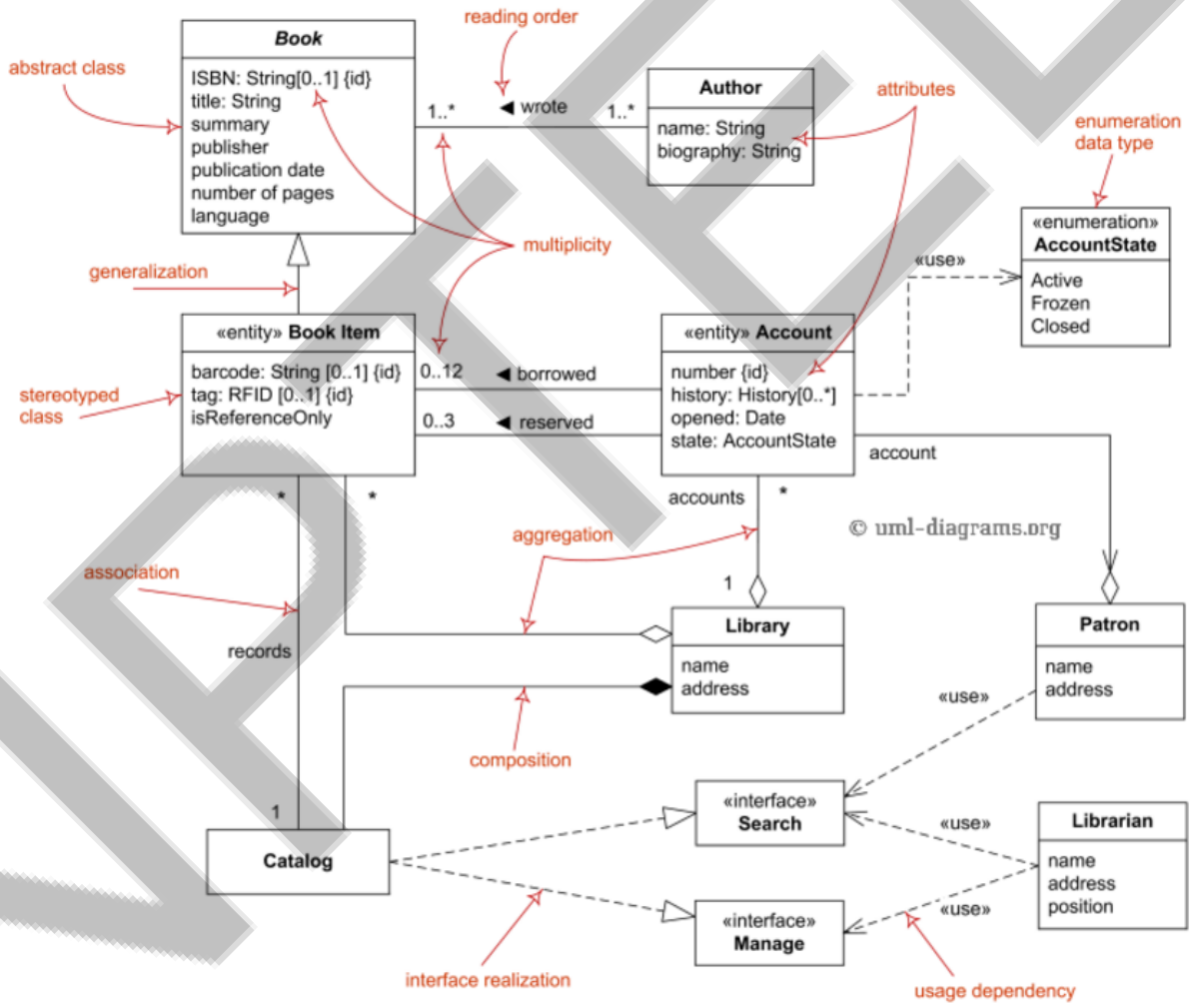


Library Domain Model



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

Library Domain Model: Annotated



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

Module Summary

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

Instructor and TAs

Name	Mail	Mobile
Partha Pratim Das, *Instructor*	ppd@cse.iitkgp.ernet.in	9830030880
Tanwi Mallick, *TA*	tanwimallick@gmail.com	9674277774
Srijoni Majumdar, *TA*	majumdarsrijoni@gmail.com	9674474267
Himadri B G S Bhuyan, *TA*	himadribhuyan@gmail.com	9438911655

Module 27: Object Oriented Analysis & Design

Class Diagrams: Part 2 (Association, Weak & Strong Aggregation)

Partha Pratim Das

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

ppd@cse.iitkgp.ernet.in
Tanwi Mallick
Srijoni Majumdar
Himadri B G S Bhuyan

This presentation uses diagrams, examples and selected texts from *Object-Oriented Analysis and Design – With Applications* by Grady Booch et. al. (3rd Ed, 2007) with kind permission of the author

Module Objectives

- Understanding relationships in Class Diagrams

Module Outline

- Relationships among classes
  - Association
  - Weak Aggregation
  - Strong Aggregation
  - Generalization
  - Dependency
  - Constraints

Relationships of Classes: RECAP (Module 14)

Class	Relationship
• A daisy is a kind of flower	Sharing connection – daisies and roses are both kinds of flowers – bright colored petals, fragrance, etc. Daisy IS_A Flower
• A rose is a (different) kind of flower	Sharing connection – daisies and roses are both kinds of flowers ... Rose IS_A Flower
• Red roses and yellow roses are both kinds of roses	Semantic connection – red roses and yellow roses are more alike than are daisies & roses Red Rose IS_A Rose, Yellow Rose IS_A Rose
• A petal is a part of both kinds of flowers	Semantic connection – daisies and roses are more closely related than are petals & flowers Flower HAS_A Petal
• Ladybugs eat certain pests such as aphids, which may be infesting certain kinds of flowers	Symbiotic connection – Ladybugs protect flowers from certain pests Semantic Dependency

Are Roses and Candles related? – Both decorate dinner tables

Source: *Object-Oriented Analysis and Design – With Applications* by Grady Booch et. al. (3rd Ed, 2007)

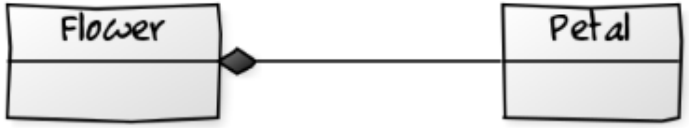
Association: RECAP (Module 14)

- Semantic Dependencies
  - Most general and most semantically weak
  - Bidirectional by default
  - Often refined over the analysis process

Early relationship

Refined to IS_A

Early relationship



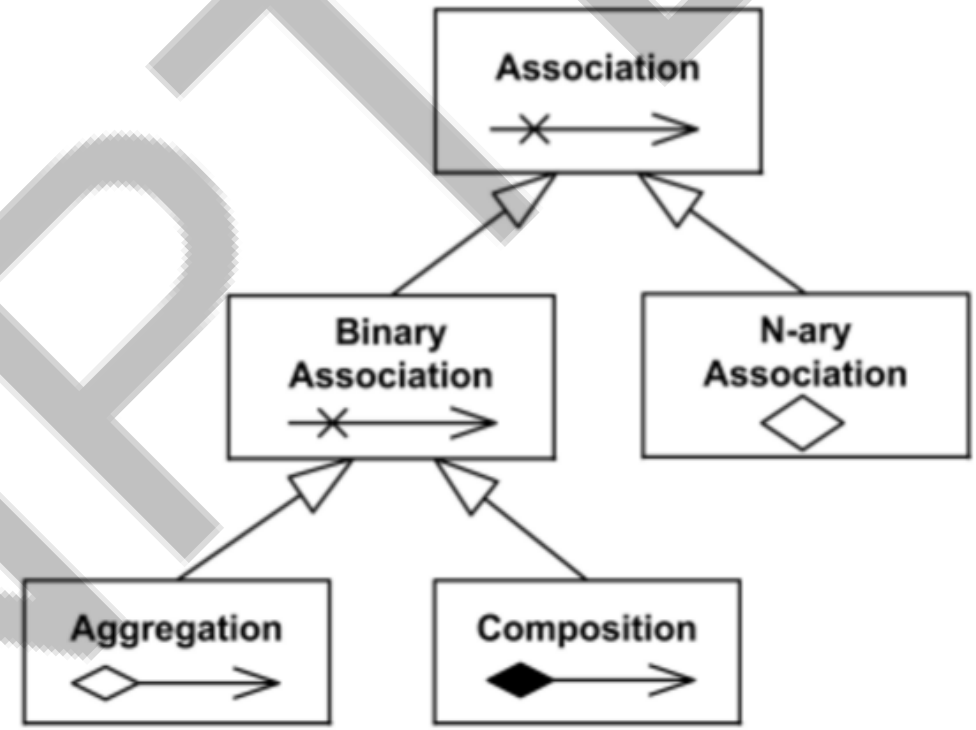
Refined to HAS_A

Early relationship

Refined to ?

Association: Notation

- An association icon (a line connector with label – association name) connects multiple classes and denotes a logical connection
- Associations can be binary of N-ary
- A class may have association to itself (Reflexive)



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

Association: Notation

We show an association below between a Professor and a Book

An association has three main concepts

- Association End
- Navigability
- Association Arity

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

Association End

- Association end is a connection between the line depicting an association and the icon depicting the connected classifier
- The association end name is commonly referred to as role name
- The role name is optional and suppressible

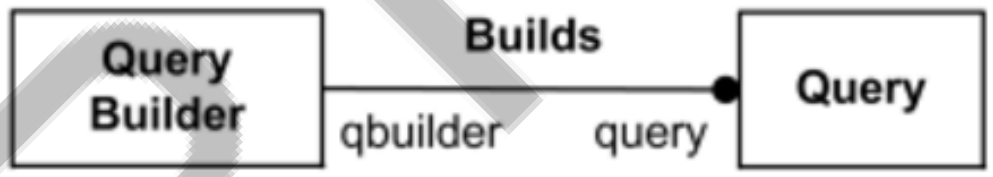
Professor "playing the role" of author is associated with textbook end typed as Book.

- Professor can have multiple roles, like author of some Books or an editor.

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

Association End

- Association end could be owned either by end class or association itself
- Ownership of association ends by an associated classifier may be indicated graphically by a small filled circle (aka dot)



Association end query is owned by classifier QueryBuilder and

association end qbuilder is owned by association Builds itself

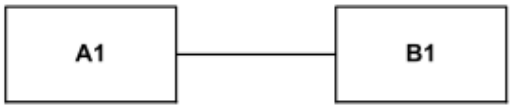
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (17-Aug-16)

Navigability

- End property of association is navigable from the opposite end(s) of association if instances of the classes at this end of the link can be accessed efficiently at run-time from instances at the other ends of the link
- Navigable end is indicated by an open arrowhead on the end of an association
- Not navigable end is indicated with a small x on the end of an association

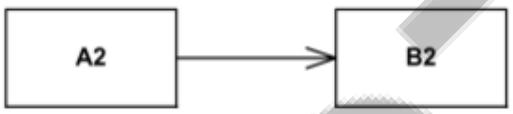
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (17-Aug-16)

Navigability

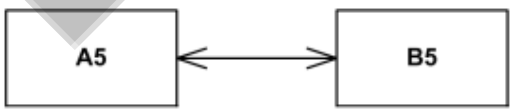


Both ends of association have ***unspecified navigability****.*

A4 is ***not navigable*** *from B4 while B4 is* ***navigable*** *from A4.*

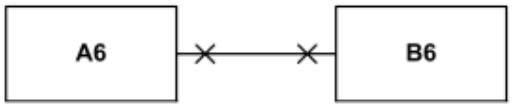


A2 has ***unspecified navigability*** *while B2 is* ***navigable*** *from A2.*



A5 is ***navigable*** *from B5 and B5 is* ***navigable*** *from A5.*

A3 is ***not navigable*** *from B3 while B3 has* ***unspecified navigability****.*



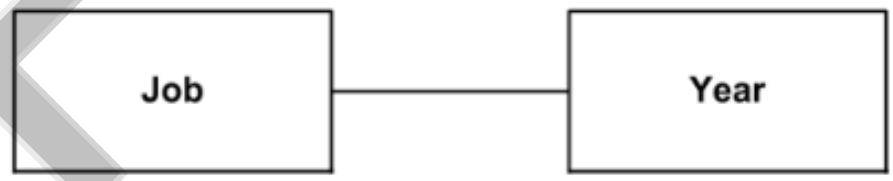
A6 is ***not navigable*** *from B6 and B6 is* ***not navigable*** *from A6.*

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

Arity – Binary Association

Each association has specific arity as it could relate two or more classes

- **Binary association** relates two typed instances
- It is normally rendered as a solid line connecting two classifiers, or a solid line connecting a single classifier to itself (the two ends are distinct)
- The line may consist of one or more connected segments

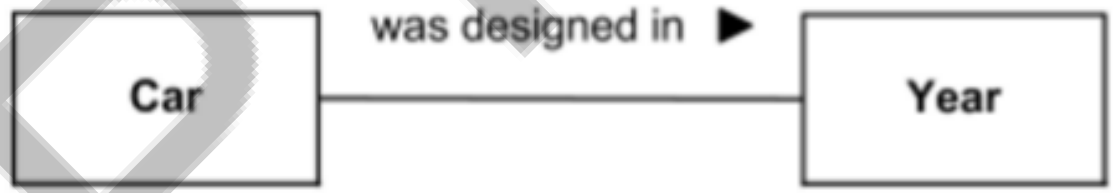


Job and Year classes are associated

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

Arity – Binary Association

- A small solid triangle could be placed next to or in place of the name of binary association (drawn as a solid line) to show the order of the ends of the association
- The arrow points along the line in the direction of the last end in the order of the association ends

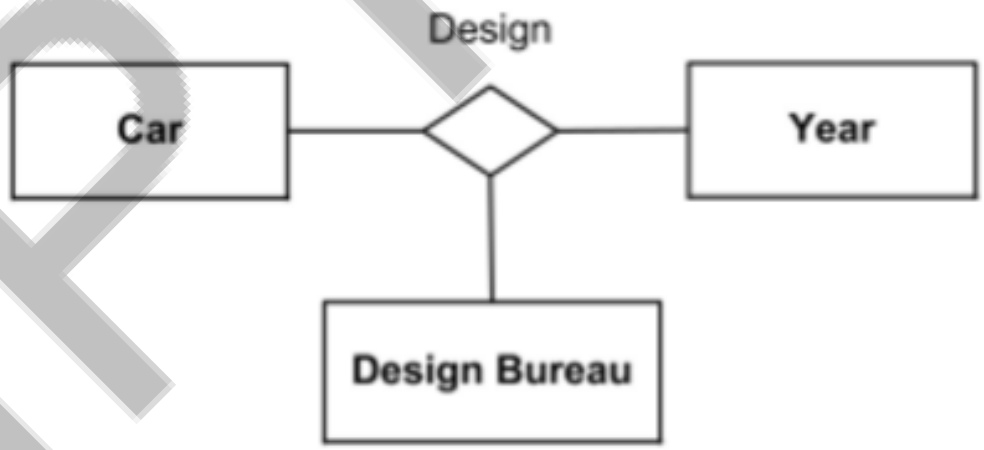


Order of the ends and reading: Car - was designed in - Year

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

Arity – N-ary Association

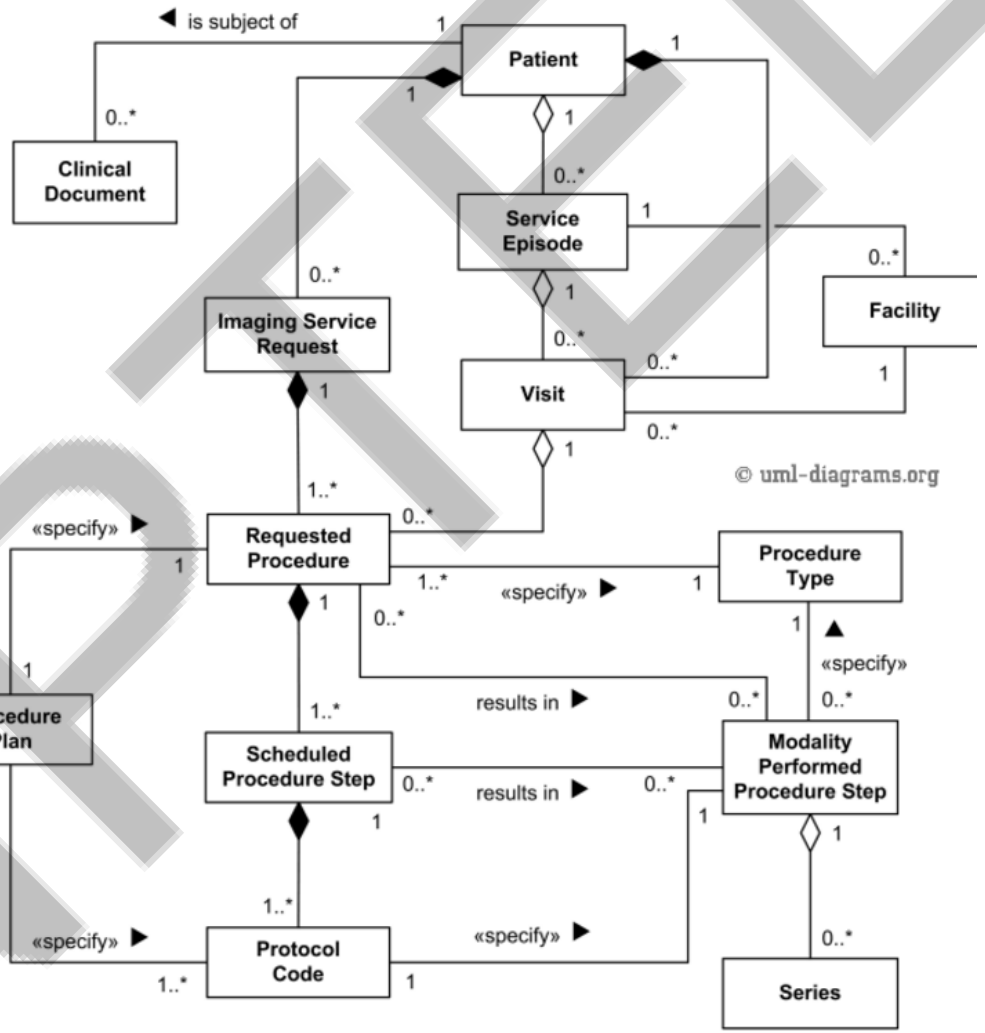
- **N-ary association** may be drawn as a diamond (larger than a terminator on a line) with a solid line for each association end connecting the diamond to the classifier that is the end's type
- N-ary association with more than two ends can only be drawn the following way



Ternary association Design relates three classes

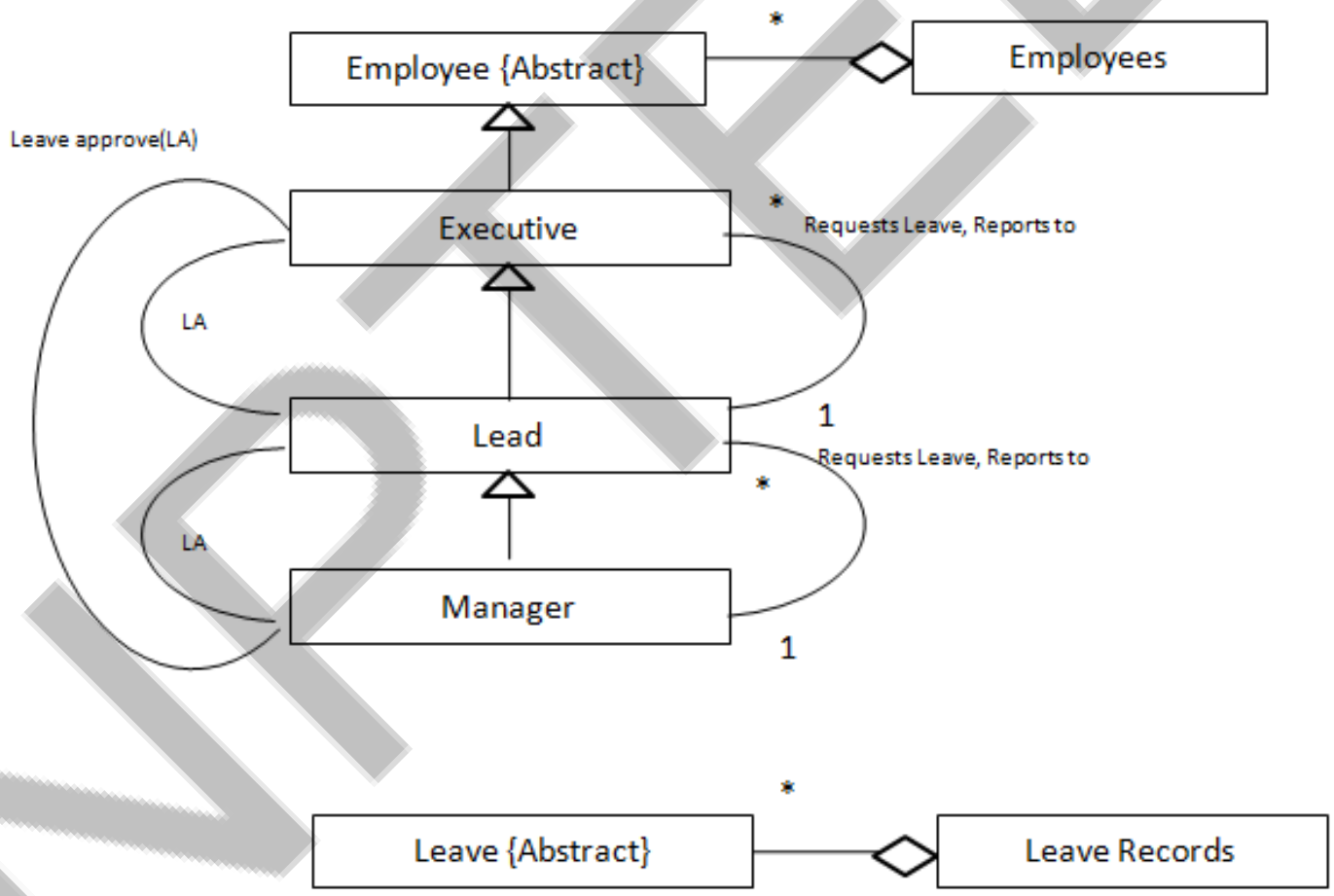
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (17-Aug-16)

Health-care Organization Model



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

Associations in LMS

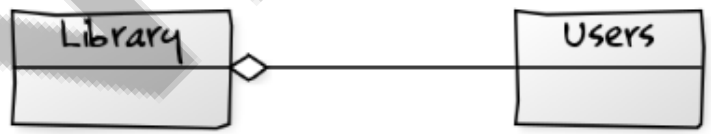


Associations in LMS

Aggregation (HAS_A): RECAP (Module 14)

- Whole / Part relationships
  - Say, we model Flower HAS_A Petal
  - Flower contains many Petals
  - Flower is the Whole, Petal is the Part
  - Depicted as:

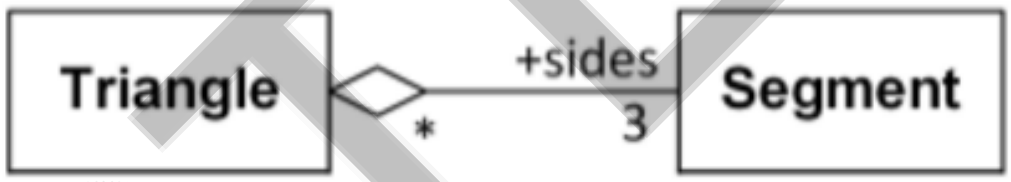
  - Physical Containment – Composition / Strong Aggregation
- Member relationship
  - Say, we model Library HAS Users
  - Library enrolls many Users
  - Library does not contain the Users
  - Depicted as:



  - Conceptual Containment – Weak Aggregation

Weak Aggregation

- Weak aggregation is depicted as an association decorated with a hollow diamond at the aggregate end of the association line



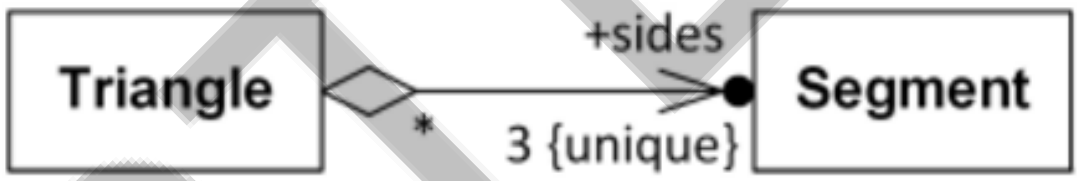
Triangle has 'sides' collection of three line Segments

Each line Segment could be part of none, one, or several triangles

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

Weak Aggregation

- Weak aggregation could be depicted together with navigability and association end ownership



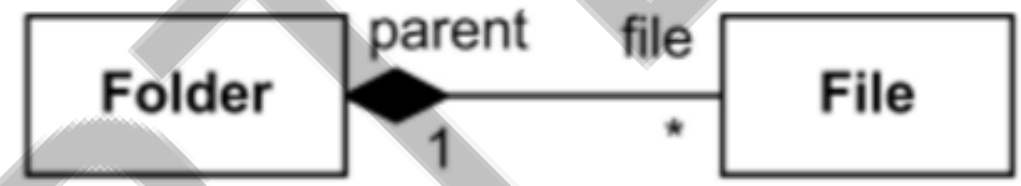
Triangle has 'sides' collection of three unique line Segments.
Line segments are navigable from Triangle.

Association end 'sides' is owned by Triangle, not by association itself

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

Strong Aggregation (Composition)

- Strong aggregation (Composition) is depicted as a binary association decorated with a filled black diamond at the aggregate (whole) end

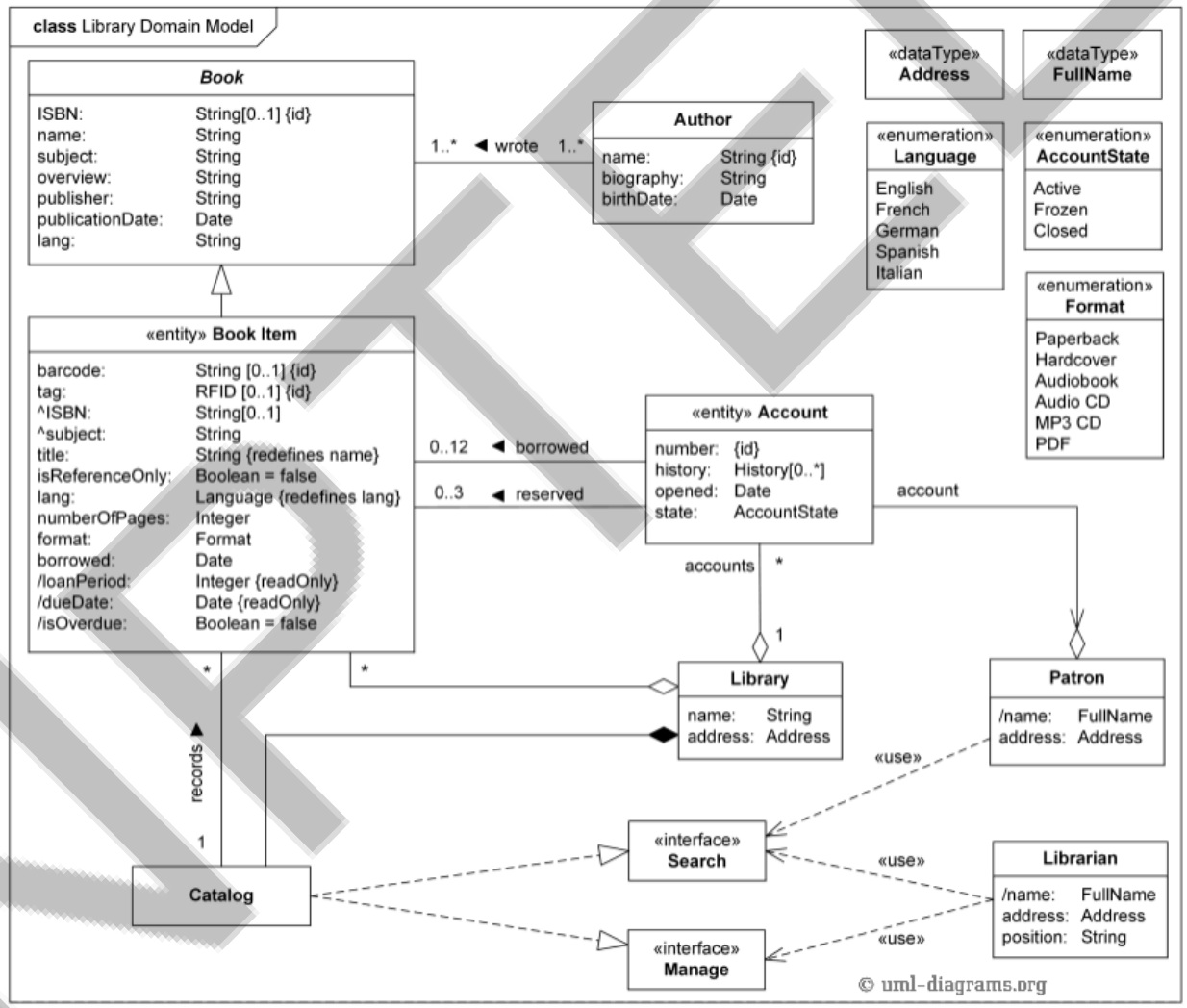


Folder could contain many files, while each File has exactly one Folder parent

If Folder is deleted, all contained Files are deleted as well

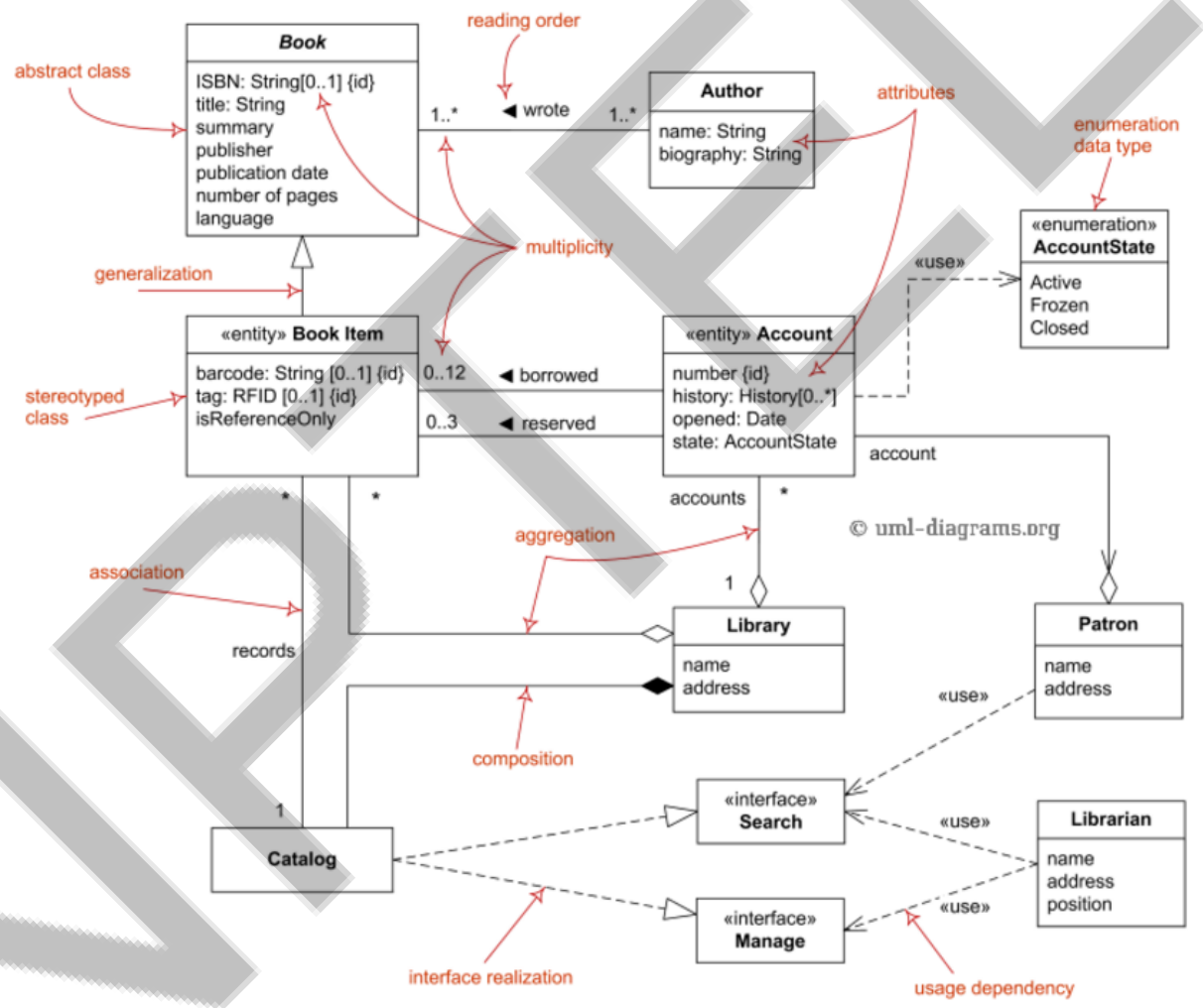
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (17-Aug-16)

Library Domain Model



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

Library Domain Model: Annotated



Domain diagram overview - classes, interfaces, associations, usage, realization, multiplicity.

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

Module Summary

- Association Relationships among classes are discussed
- Weak Aggregation and Strong Aggregation are important binary associations

Instructor and TAs

Name	Mail	Mobile
Partha Pratim Das, *Instructor*	ppd@cse.iitkgp.ernet.in	9830030880
Tanwi Mallick, *TA*	tanwimallick@gmail.com	9674277774
Srijoni Majumdar, *TA*	majumdarsrijoni@gmail.com	9674474267
Himadri B G S Bhuyan, *TA*	himadribhuyan@gmail.com	9438911655

Module 28: Object Oriented Analysis & Design

Class Diagrams: Part 3

(Generalization, Dependency & Constraints)

Partha Pratim Das

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

ppd@cse.iitkgp.ernet.in
Tanwi Mallick
Srijoni Majumdar
Himadri B G S Bhuyan

This presentation uses diagrams, examples and selected texts from *Object-Oriented Analysis and Design – With Applications* by Grady Booch et. al. (3rd Ed, 2007) with kind permission of the author

Module Objectives

- Understanding relationships in Class Diagrams

Module Outline

- Relationships among classes
  - Association
  - Weak Aggregation
  - Strong Aggregation
  - Generalization
  - Dependency
  - Constraints
- Class Diagram for LMS

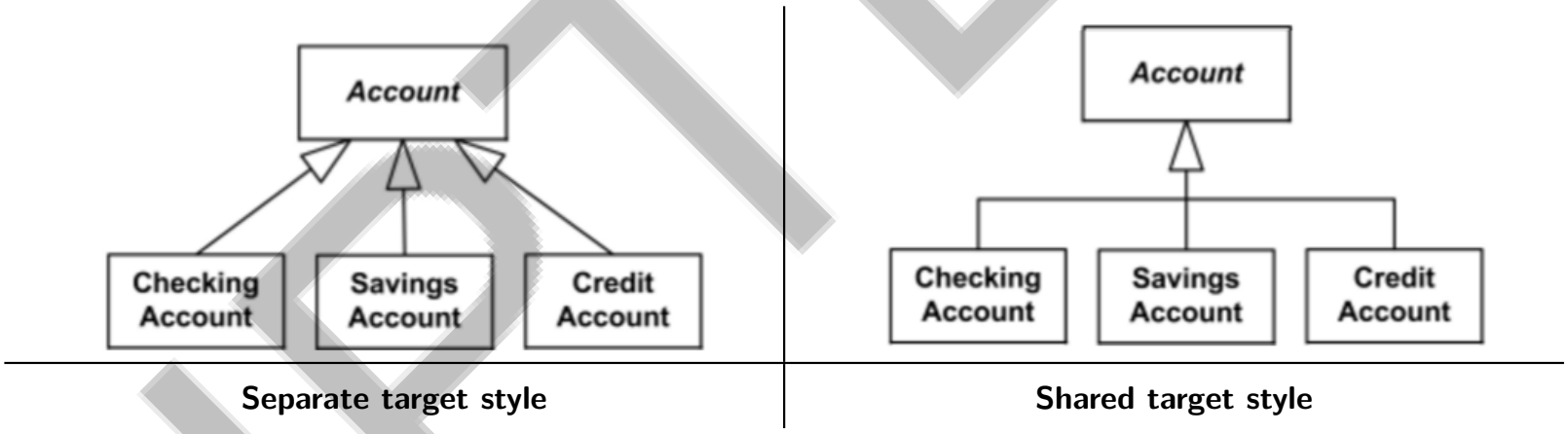
Inheritance (IS_A): RECAP (Module 14)

- Generalization / Specialization relationships
  - Say, we model Daisy IS_A Flower
  - Daisy will inherit the properties of Flower, and have some more of its own
  - Flower is the Generalization
  - Daisy is the Specialization
  - Depicted as:

- Semantically most interesting
- Can *delegate* behavior to related objects
- Comes in a number of flavors
  - Single / Multilevel / Hierarchical Inheritance
  - Multiple Inheritance
  - Hybrid Inheritance

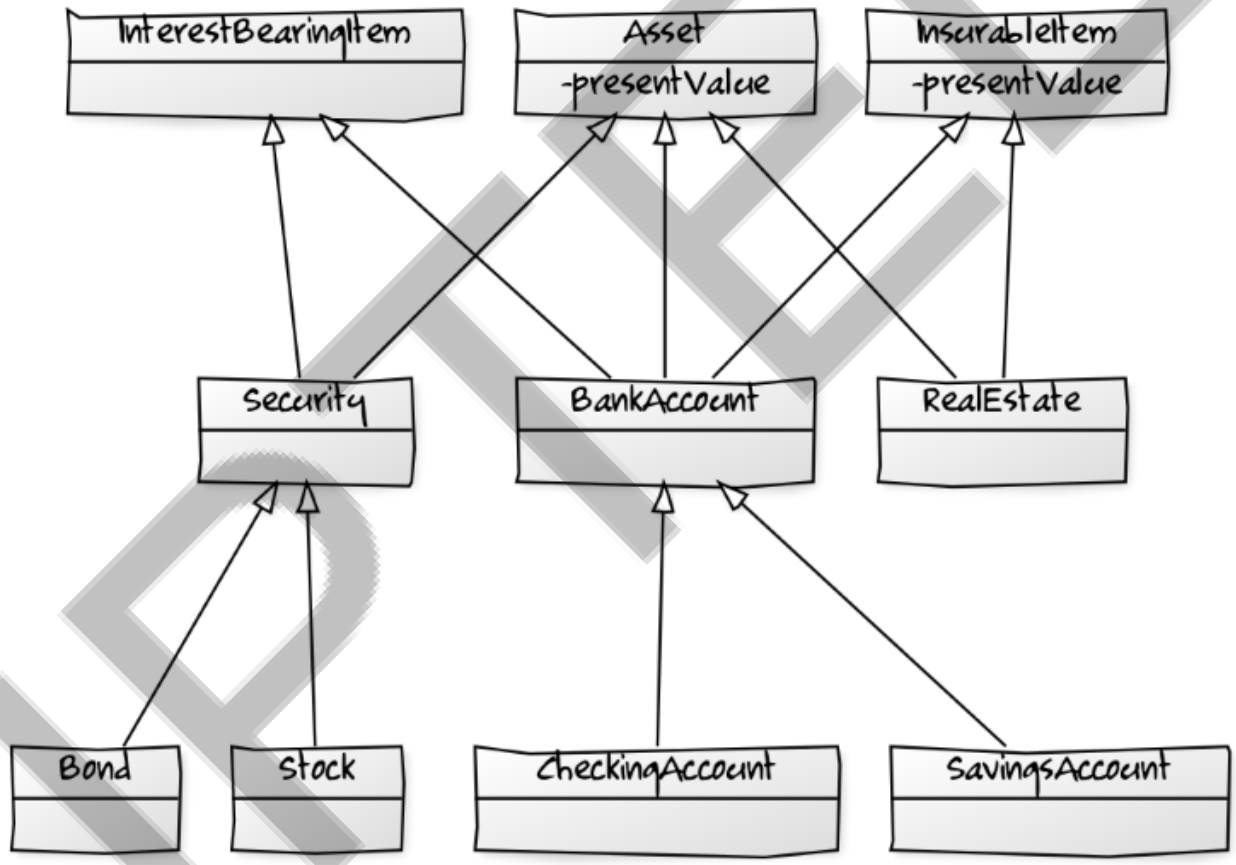
Generalization

- A generalization is shown as a line with a hollow triangle as an arrowhead



Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (10-Aug-16)

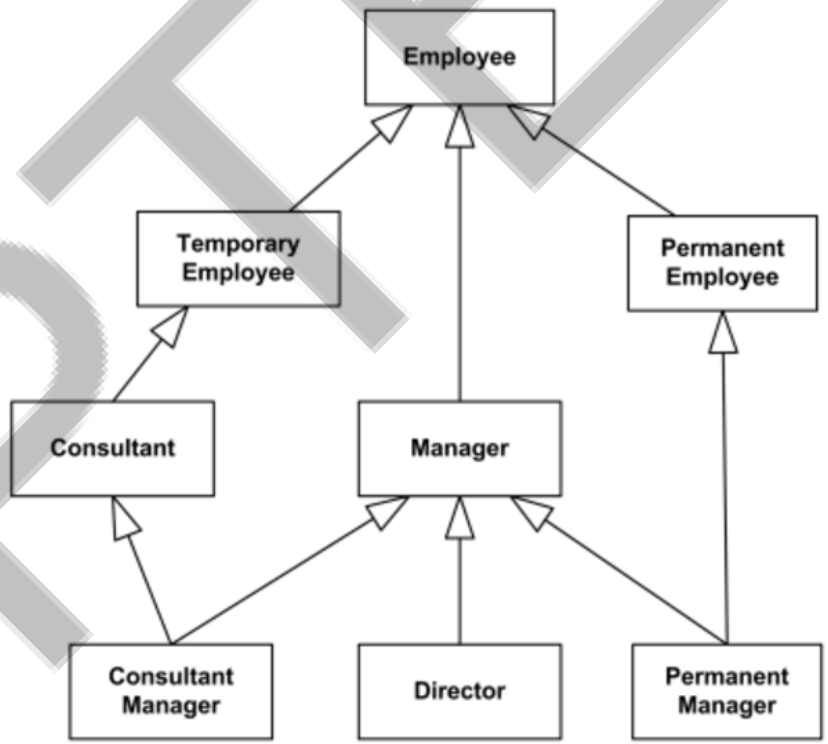
Multiple Inheritance: RECAP (Module 14)



- More than one superclass for a subclass
- RealEstate IS_A Asset, InsurableItem

Multiple Inheritance

- Multiple inheritance is implicitly allowed by UML standard, while the standard provides no definition of what it is

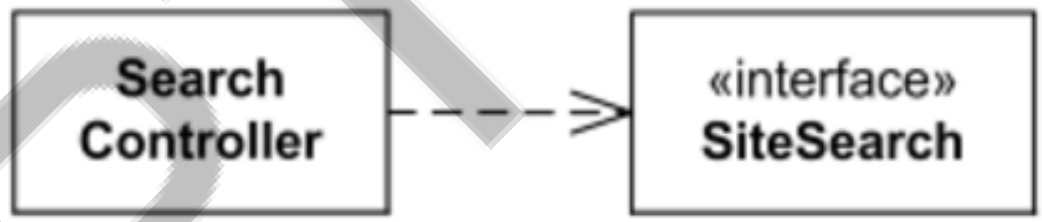


Multiple inheritance for Consultant Manager and Permanent Manager – both inherit from two classes

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (10-Aug-16)

Dependency

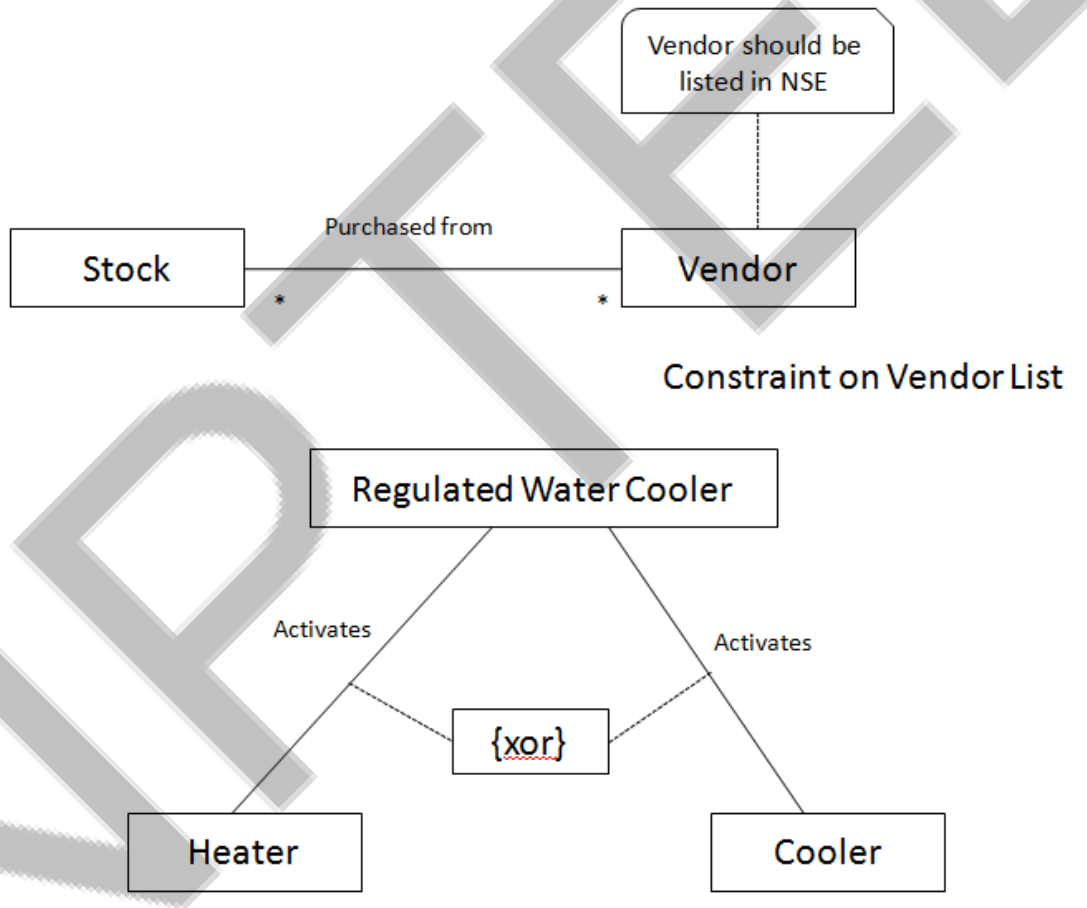
- Dependency is a directed relationship which is used to show that some UML element or a set of elements requires, needs or depends on other model elements for specification or implementation



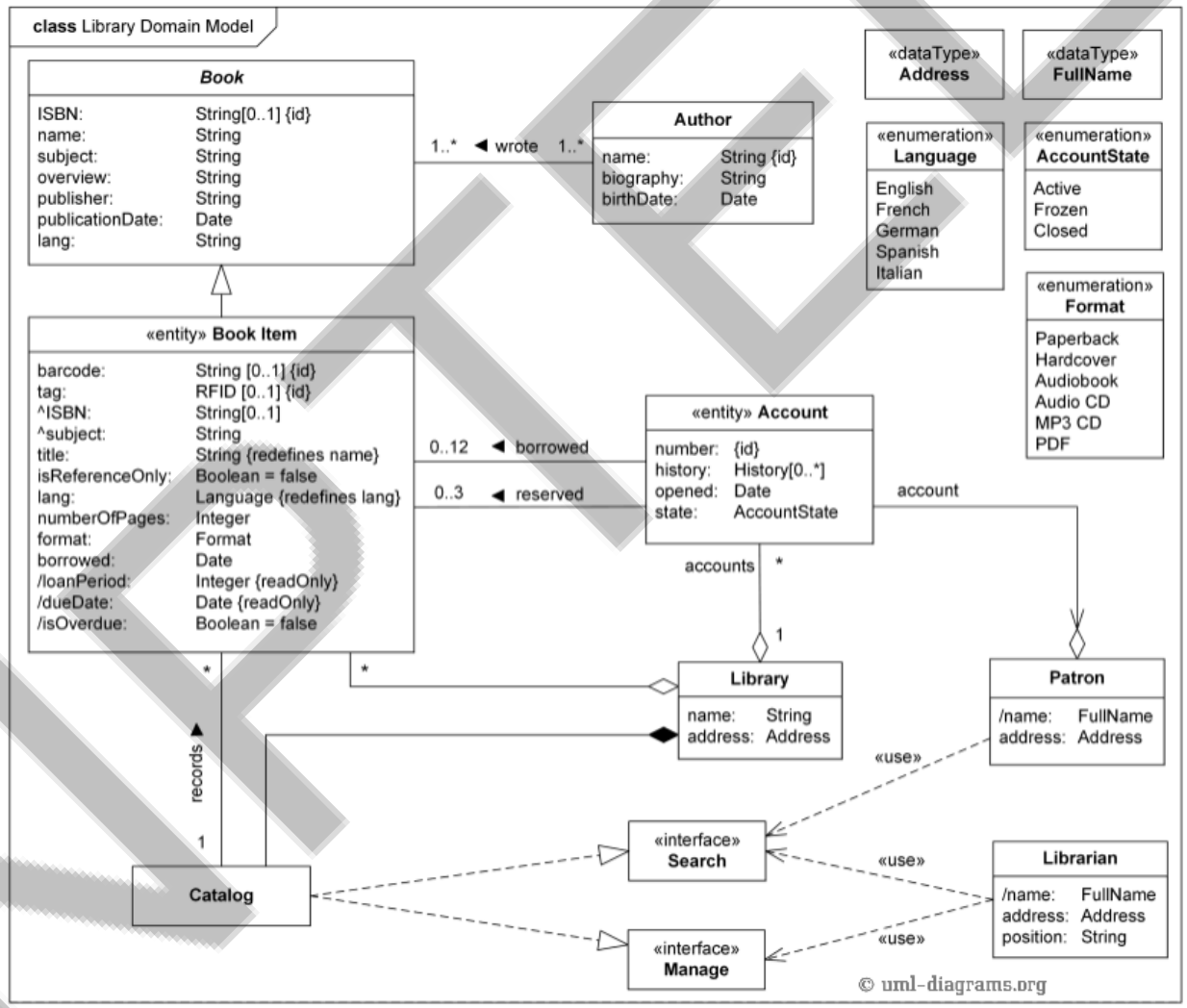
Class SearchController depends on (requires) SiteSearch interface

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

Constraints

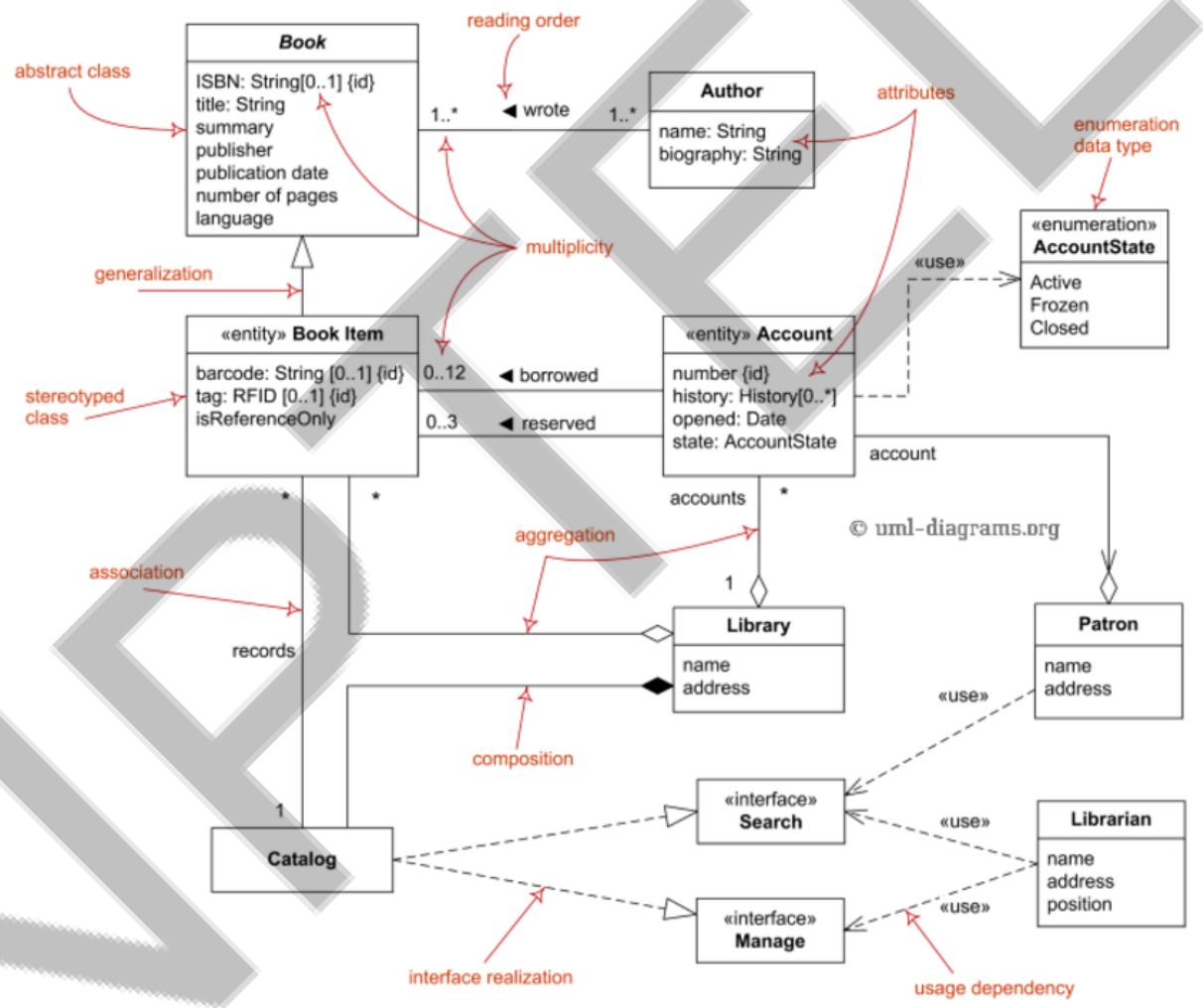


Library Domain Model



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

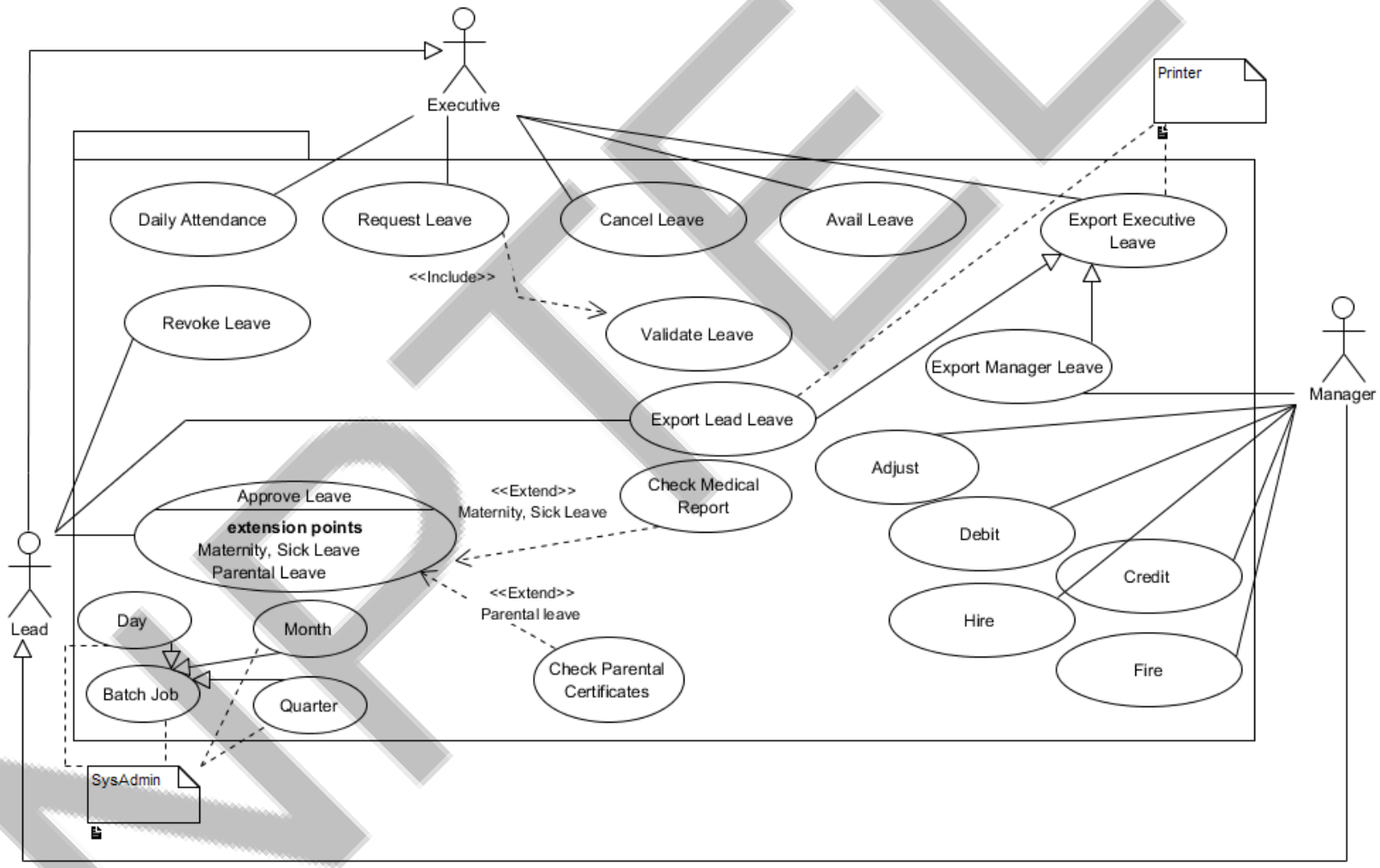
Library Domain Model: Annotated



Domain diagram overview - classes, interfaces, associations, usage, realization, multiplicity.

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

Use-Case Diagram for LMS RECAP (Module 25)



Not all use cases are shown in details

Class Diagram for LMS

We now derive the Class Diagram for LMS. The steps involved are:

- Identify Classes {Abstract Classes}
- Identify Properties and Operations
- Identify the Relationships among Classes
- Class Diagram

Identification of Classes {Abstract Classes}

- Reading through the specification of the Leave Management System, we identify the various instances, that is, objects
- We categorize them into two abstract classes: Employee and Leave

Employee {Abstract}	Leave {Abstract}

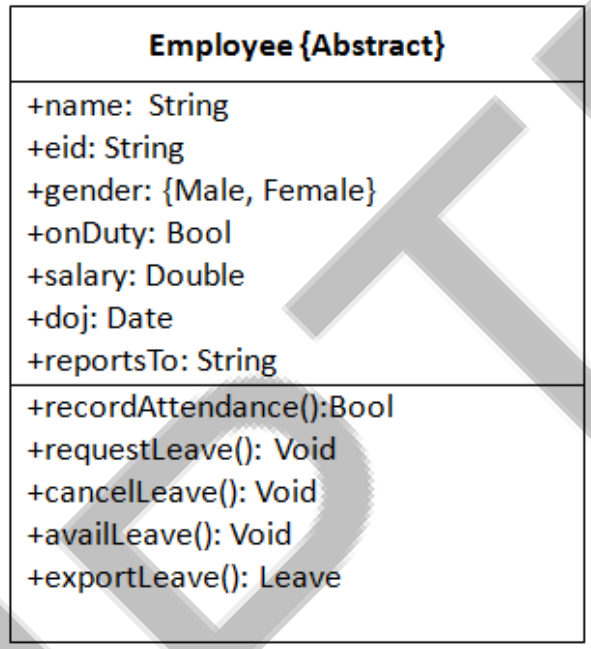
Identification of Properties

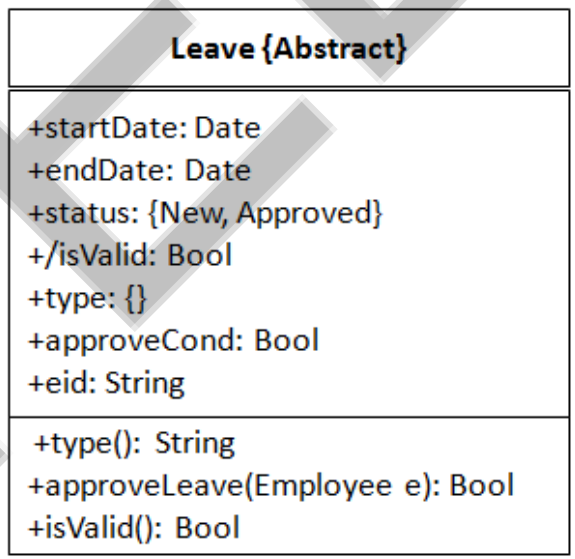
Properties of the two abstract class of LMS

Employee {Abstract}
+name: String +eid: String +gender: {Male, Female} +onDuty: Bool +salary: Double +doj: Date +reportsTo: String

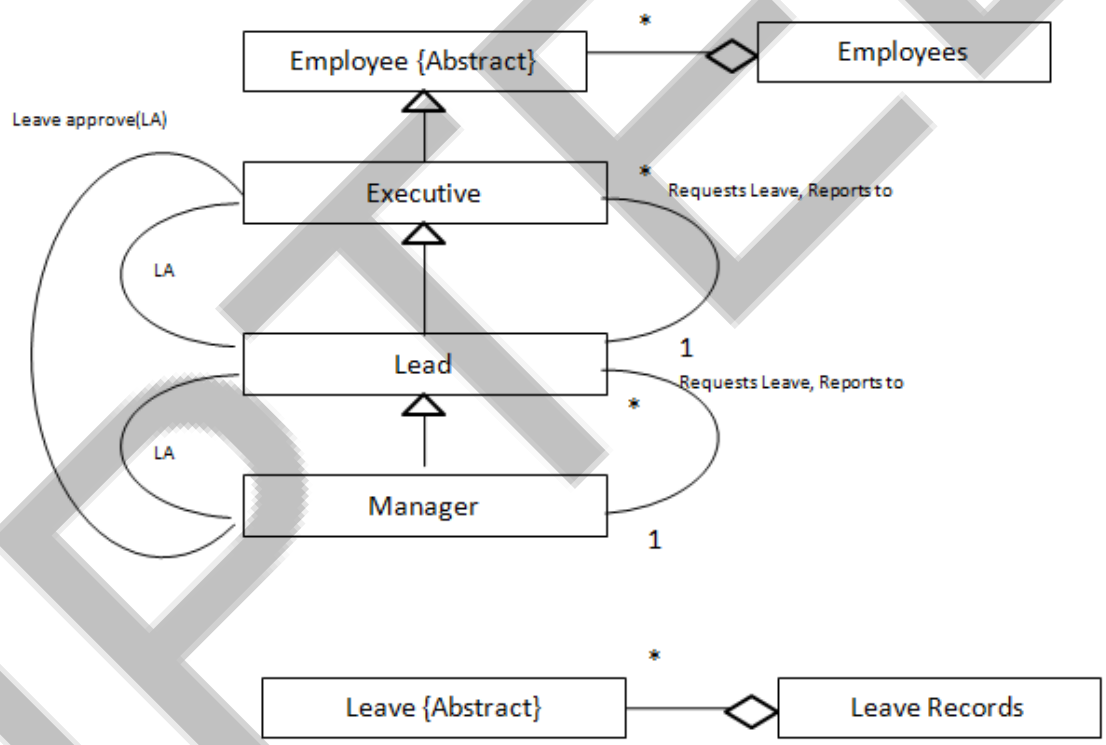
Leave {Abstract}
+startDate: Date +endDate: Date +status: {New, Approved} +/isValid: Bool +type: {} +approveCond: Bool +eid: String

Identification of Operations

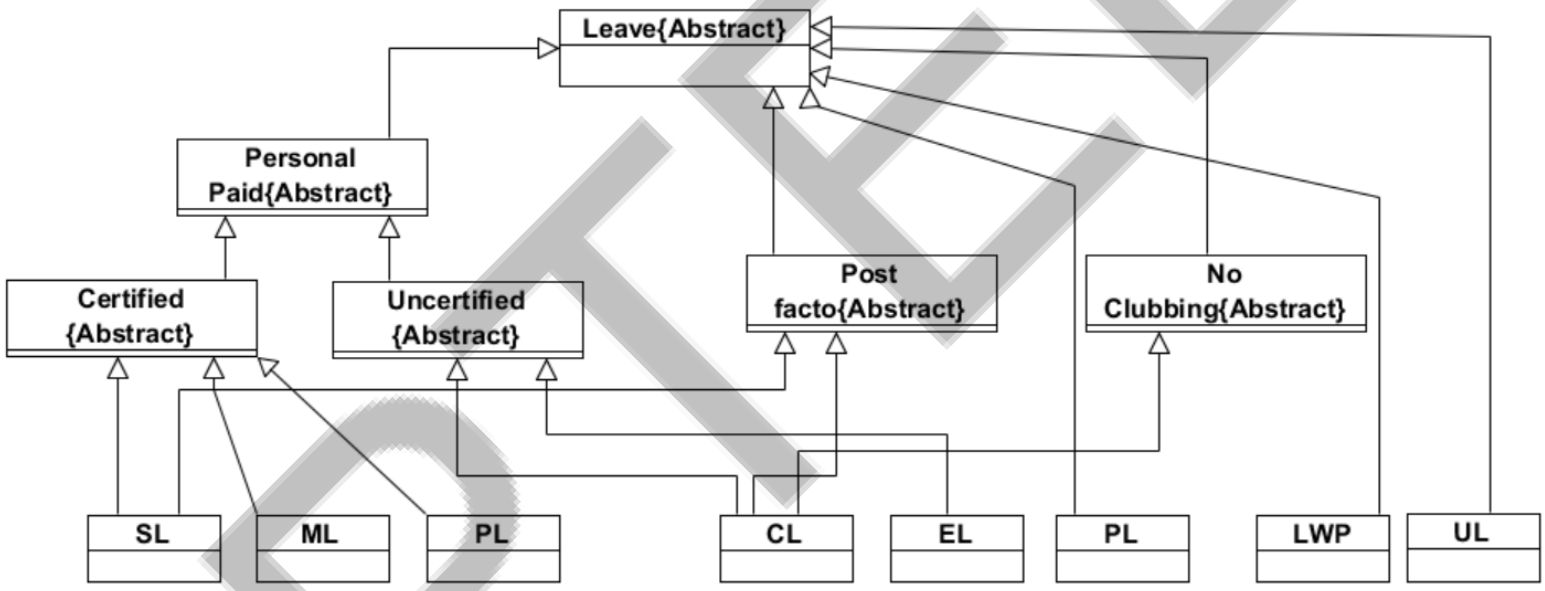




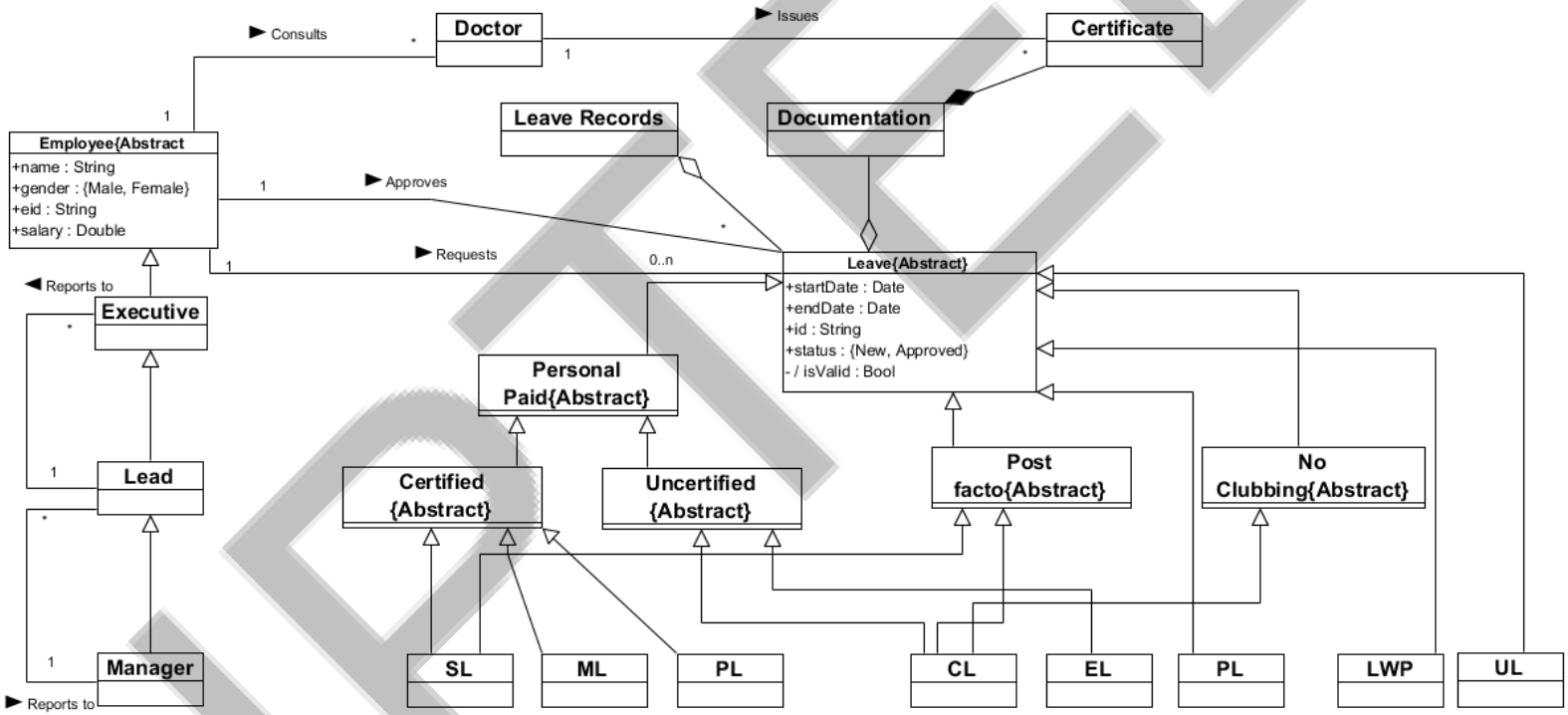
Identification of Associations



Identification of Generalizations



LMS Class Diagram (Partial)



Module Summary

- Discussed Generalization, Dependency and Constraint relationships
- A partial Class Diagram for the Leave Management System (LMS)

Instructor and TAs

Name	Mail	Mobile
Partha Pratim Das, *Instructor*	ppd@cse.iitkgp.ernet.in	9830030880
Tanwi Mallick, *TA*	tanwimallick@gmail.com	9674277774
Srijoni Majumdar, *TA*	majumdarsrijoni@gmail.com	9674474267
Himadri B G S Bhuyan, *TA*	himadribhuyan@gmail.com	9438911655

Module 29: Object Oriented Analysis & Design

Sequence Diagrams: Part 1

Partha Pratim Das

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

ppd@cse.iitkgp.ernet.in
Tanwi Mallick
Srijoni Majumdar
Himadri B G S Bhuyan

This presentation uses diagrams, examples and selected texts from *Object-Oriented Analysis and Design – With Applications* by Grady Booch et. al. (3rd Ed, 2007) with kind permission of the author

Module Objectives

- Understanding Sequence Diagrams

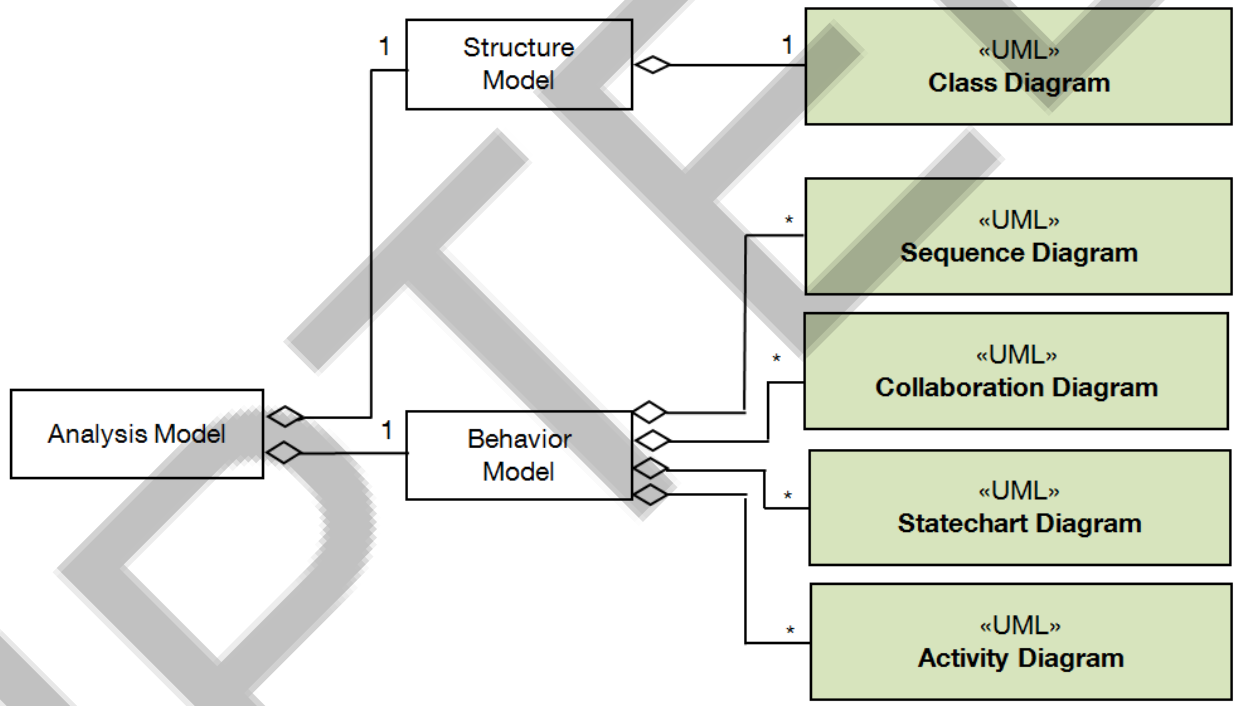
Module Outline

- What are Sequence Diagrams?
  - Lifeline
  - Messages
  - Interaction Fragments
  - Examples

Client-Server Computing Model: RECAP (Modules 05, 11)

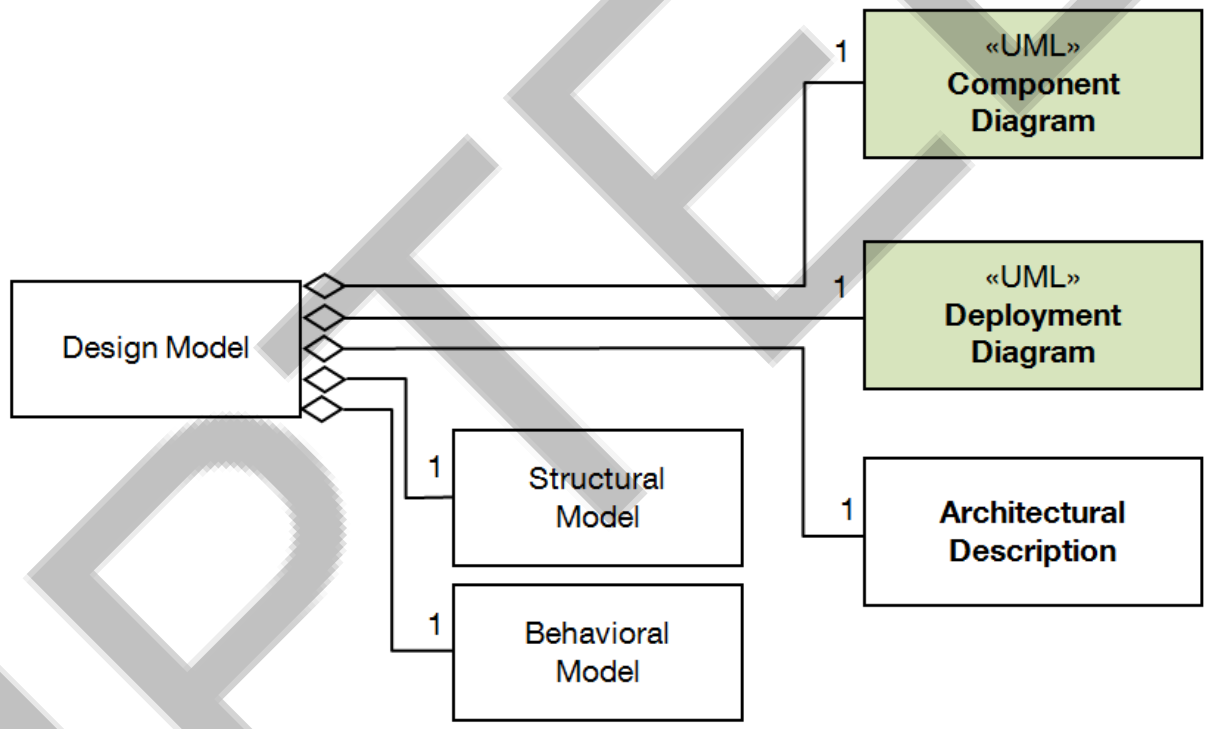
- No object exists in isolation
- Objects are acted on and themselves act on other objects
- Leads to the **Client-Server Model** of computing where
  - Behavior is
    - Services provided by an object
  - Services are requested by
    - Sending Messages, Invoking Operations
  - In Client-Server View
    - Clients request for Services
    - Servers provide Services
    - Contract between client and server ensures correctness

Sequence Diagrams in SDLC phases: RECAP (Module 22)



- In the Analysis Phase the problem domain is analyzed and refined from the Requirements Phase
- The behavior model of the system is hence understood in this phase
- Sequence diagram is a major result of the Analysis Phase

Sequence Diagrams in SDLC phases: RECAP (Module 22)



- Sequence diagram is included in the Behavioral Model
- It is further refined in the Design Phase

What are Sequence Diagrams?

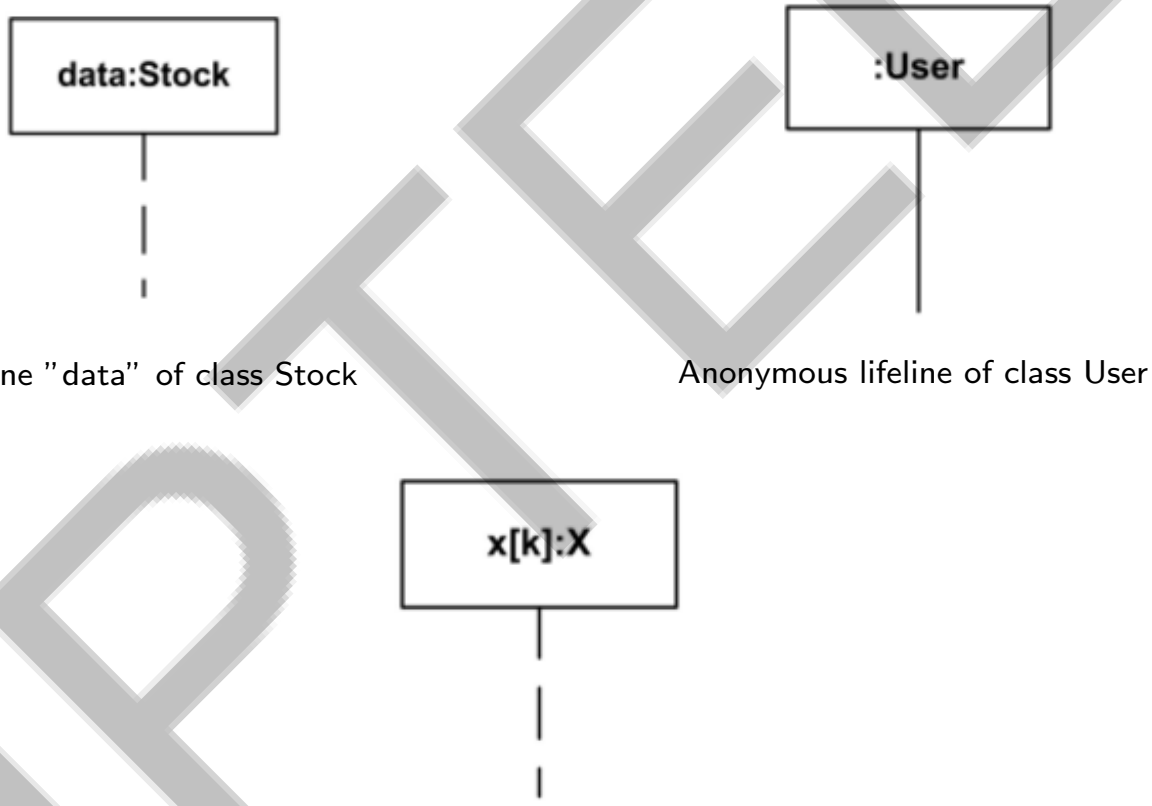
- Sequence diagram is the most common kind of Interaction diagram, which focuses on the message interchange between a number of lifelines
- Sequence diagram is a UML behavior diagram
- Sequence diagram depicts the inter-object behavior of a system, ordered by time
- The major components of a Sequence Diagram are:
  - Lifeline
  - Messages
  - Interaction Fragments

Lifeline

- **Lifeline** is an element which represents an individual participant in the interaction
- Lifelines represent only one interacting entity
- If the referenced connectable element is multi-valued (that is, has a multiplicity > 1), then the lifeline may have an expression (selector) that specifies which particular part is represented by this lifeline
- A lifeline is shown using a symbol that consists of a rectangle forming its "head" followed by a vertical line (which may be dashed) that represents the lifetime of the participant
- The information identifying a lifeline is depicted as ObjectName[selector]:ClassName

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Lifeline



Lifeline "data" of class Stock

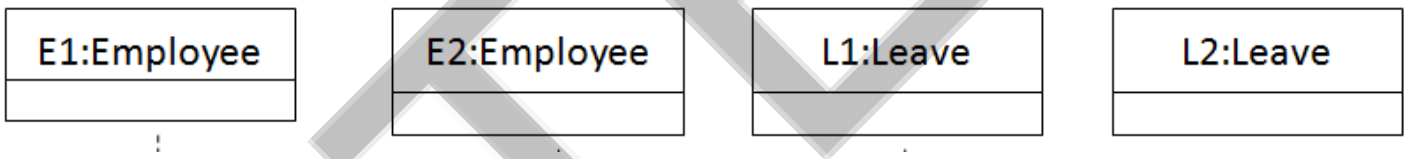
Anonymous lifeline of class User

Lifeline "x" of class X is selected with selector [k]

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Named Elements of LMS

- The major named elements of LMS are Employee and Leave. Few instances of them shown below.



- The major interaction activity of LMS is **Request Leave**, **Approve Leave** which requires interaction between the two major classes, Employee and Leave

Types of Messages

- Message is an element that defines one specific kind of communication between lifelines of an interaction
- There are 2 major types of message in Sequence Diagram
  - Messages by Action Type
  - Messages by Presence of Events

Message by Action Type: A message reflects either an operation call and start of execution or a sending and reception of a signal

Message by Presence of Events: A message depends on whether message send event and receive events are present

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Messages by Action Type

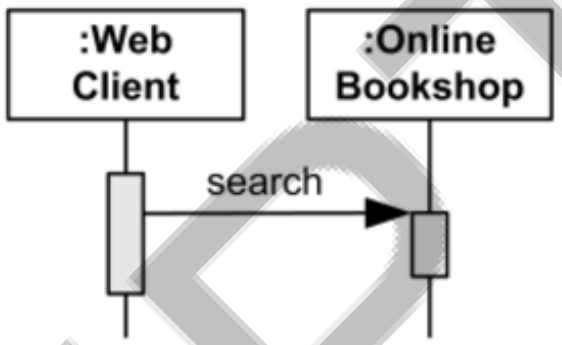
The various types of Messages by Action type are:

- synchronous call
- asynchronous call / signal
- create
- delete
- reply

Messages by Action Type

Synchronous call typically represents operation call - send message and suspend execution while waiting for response

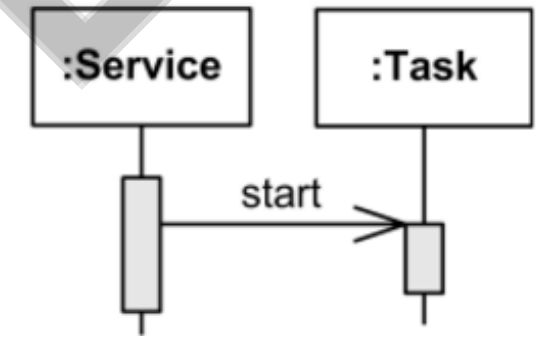
Notation: filled arrow head.



Web Client searches Online Bookshop and waits for results

Asynchronous call - send message and proceed immediately without waiting for return value

Notation: Open arrow head



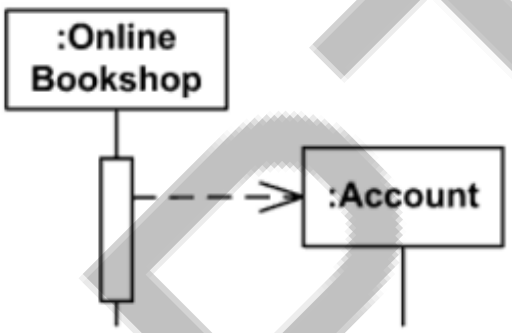
Service starts Task and proceeds in parallel without waiting

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Messages by Action Type

Create message is sent to a lifeline to create itself

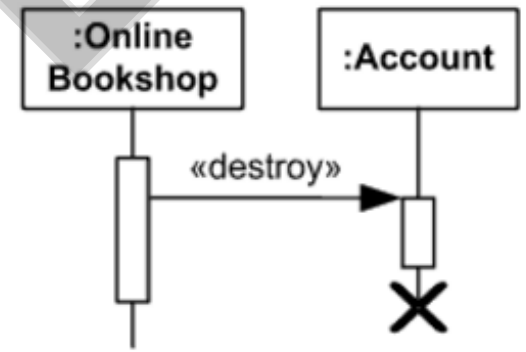
Notation: Dashed line with open arrowhead



Online Bookshop creates Account

Delete message is sent to terminate another lifeline

Notation: lifeline usually ends with a cross (X) at the bottom



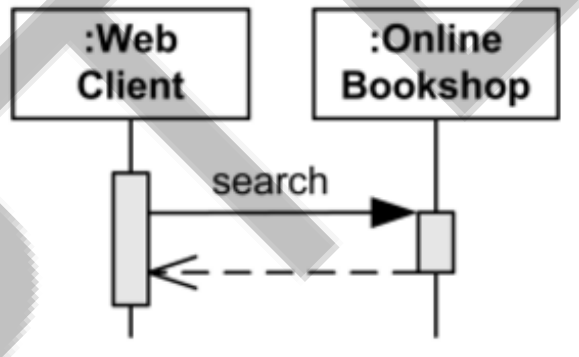
Online Bookshop terminates Account

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Messages by Action Type

Reply message to an operation call

Notation: Dashed line with open arrow head



Web Client searches Online Bookshop and waits for results to be returned

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

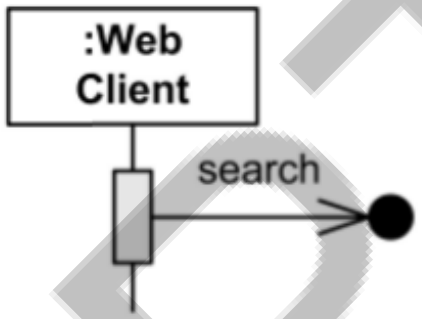
Messages by Presence of Events

The various types of Messages by Presence of Events are:

- complete message
  - The semantics of a complete message is the trace <sendEvent, receiveEvent>
  - Both sendEvent and receiveEvent are present
- lost message
- found message
- unknown message (default) – both sendEvent and receiveEvent are absent (should not appear)

Messages by Presence of Events

Lost Message is a message where the sending event is known, but there is no receiving event



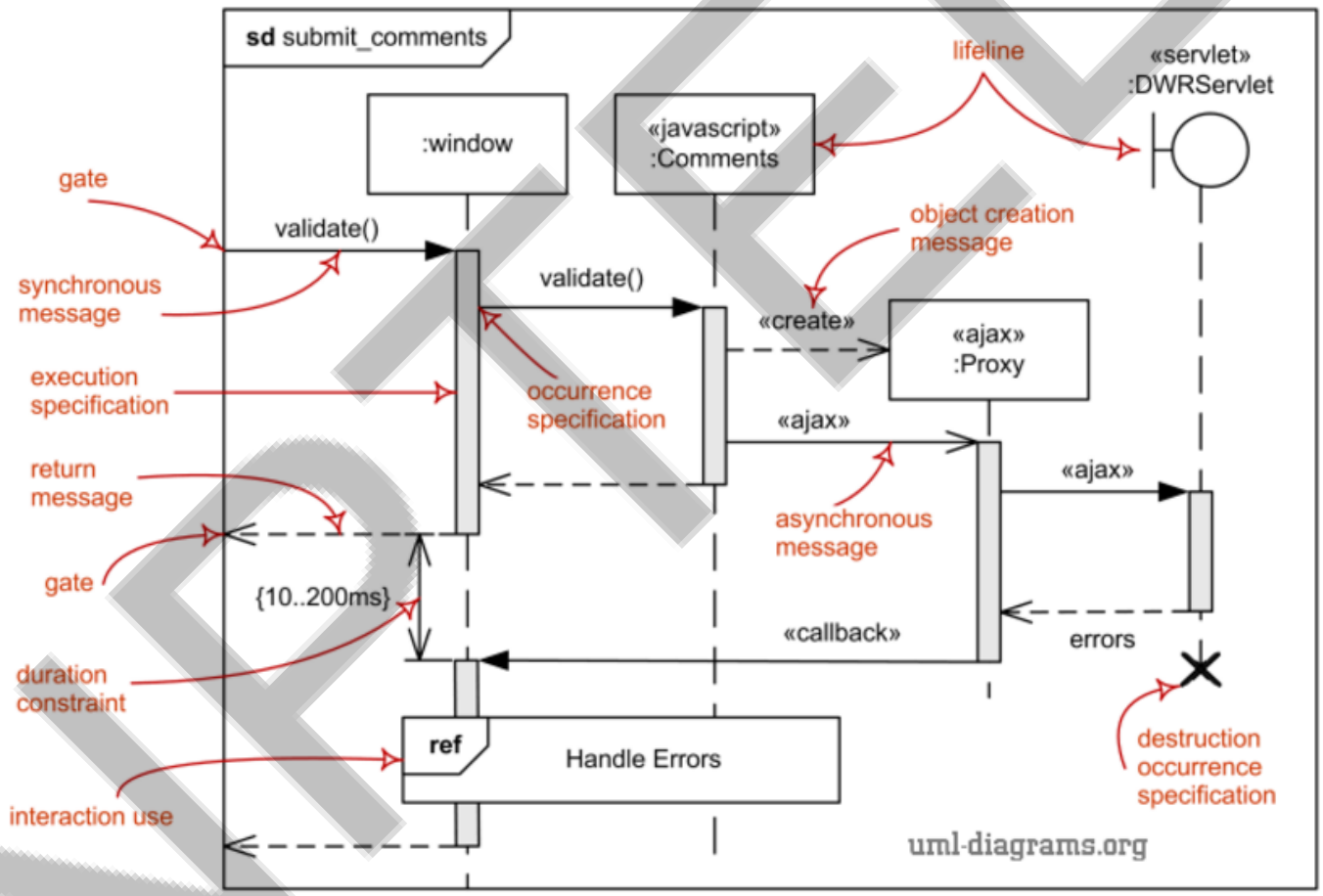
Web Client sent search message which was lost

Found Message is a message where the receiving event is known, but there is no (known) sending event

Online Bookshop gets search message of unknown origin

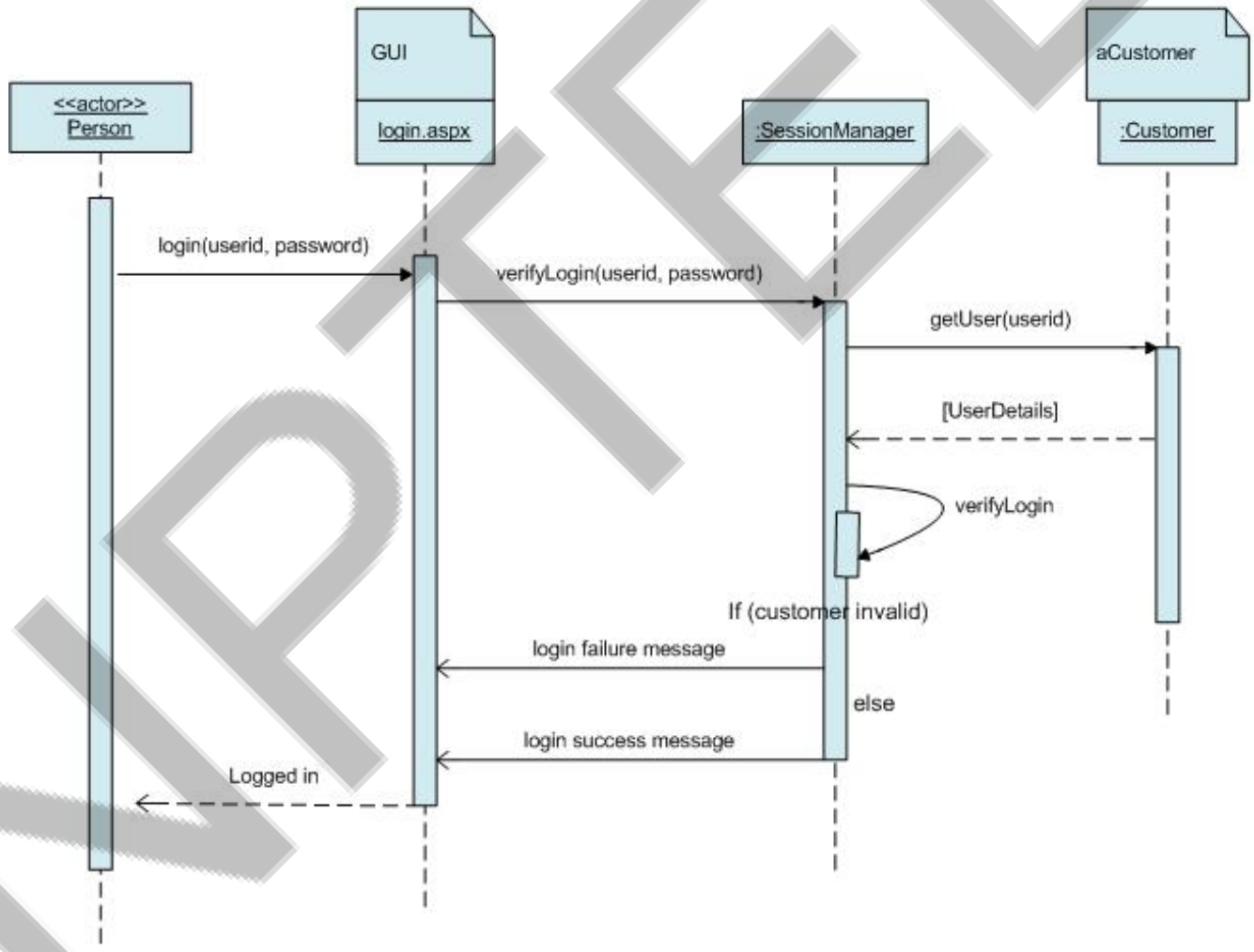
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

An Annotated Sequence Diagram



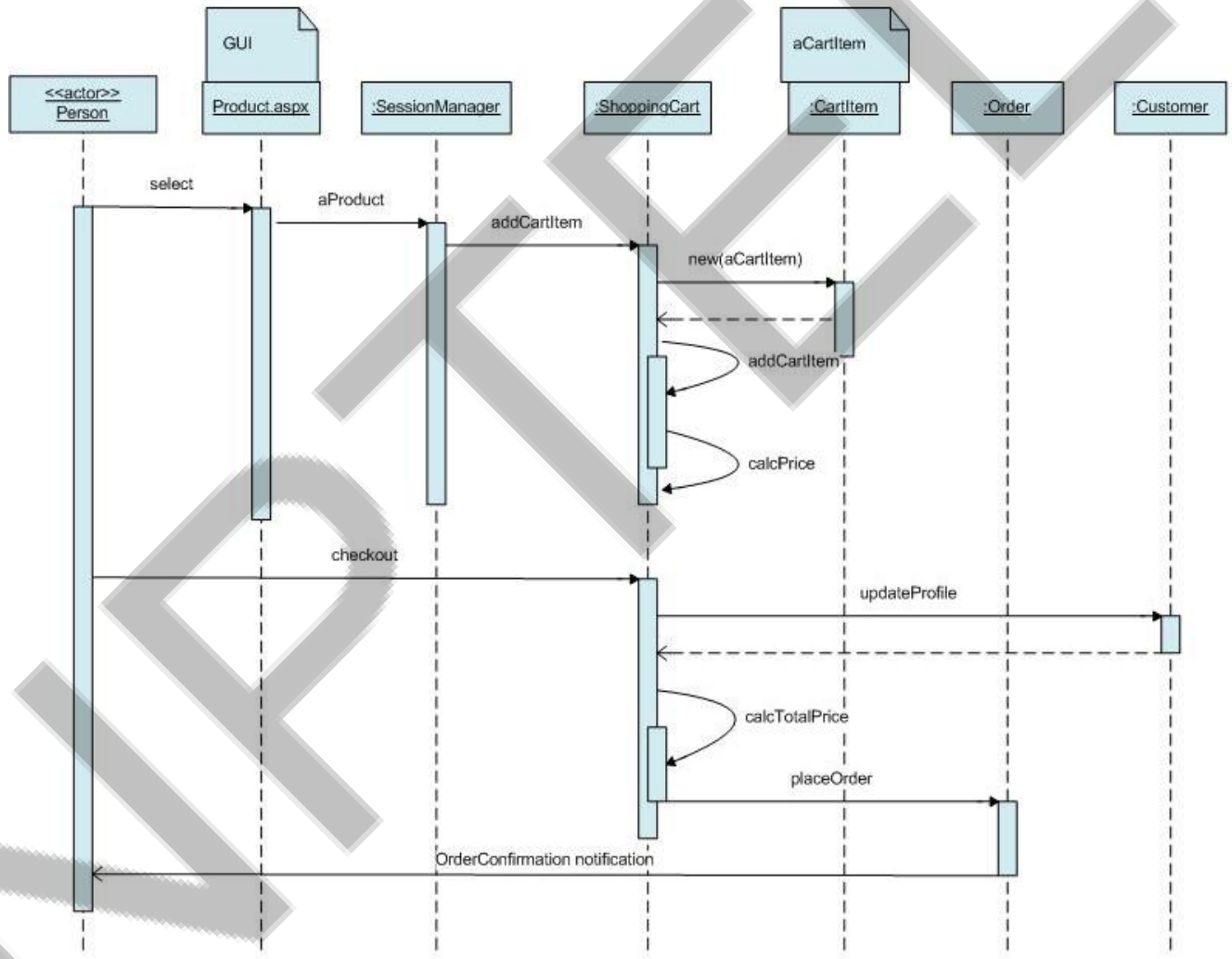
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Example: Login



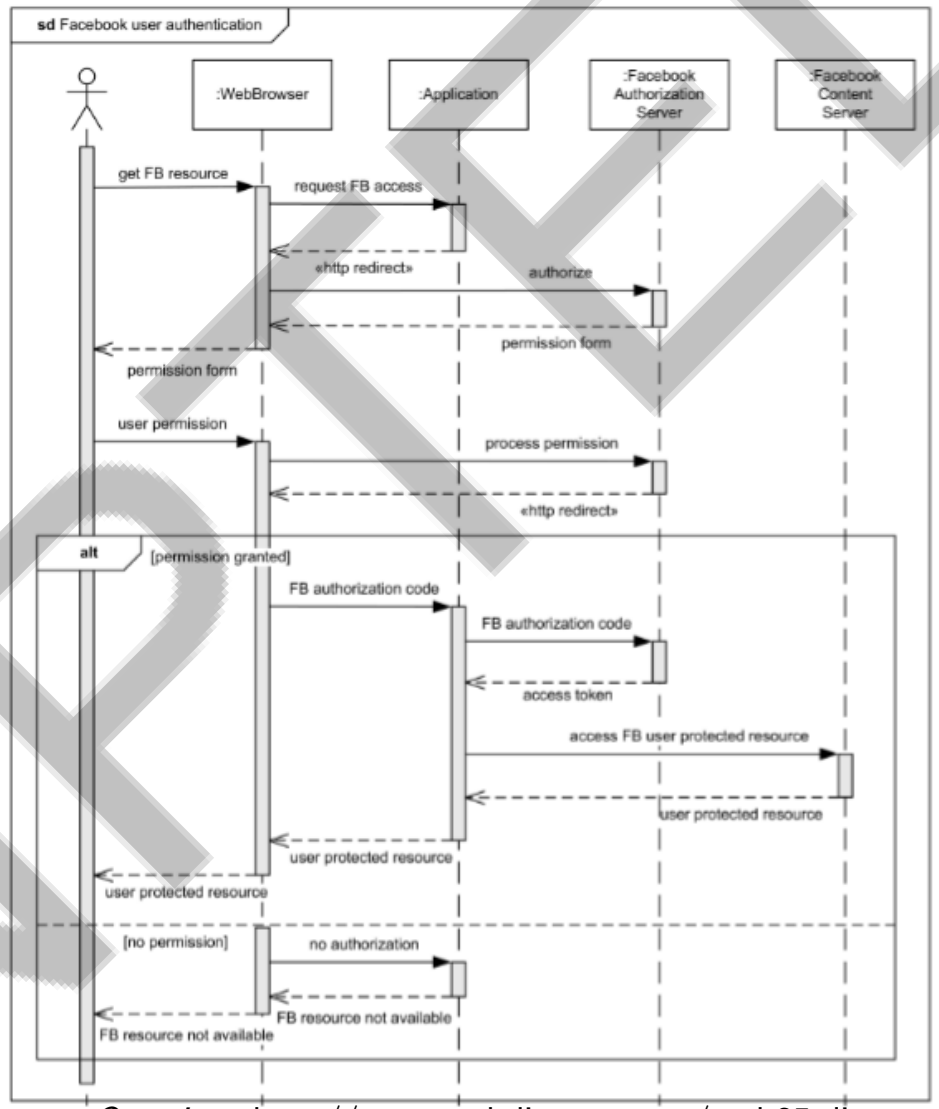
Source: http://people.cs.ksu.edu/ reshma/798_SequenceDiagram.htm (18-Aug-16)

Example: Place Order



Source: http://people.cs.ksu.edu/ reshma/798_SequenceDiagram.htm (18-Aug-16)

Example: Facebook Authentication



Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Messages of LMS

The messages for the major activities of LMS are given below:

- **Request Leave**
  - Request Leave() from Employee
  - new() Leave
  - isValid() Leave
  - return(ifvalid == true)
- **Approve Leave**
  - Approve Leave() from Employee
  - Approver()
  - Reportingto()
  - return(Reportingto)

Module Summary

- Introduced sequence diagram to capture the detailed execution flows of objects, their interactions and lifeline with a temporal ordering among events
- Discussed lifeline and messages in depth with examples

Instructor and TAs

Name	Mail	Mobile
Partha Pratim Das, *Instructor*	ppd@cse.iitkgp.ernet.in	9830030880
Tanwi Mallick, *TA*	tanwimallick@gmail.com	9674277774
Srijoni Majumdar, *TA*	majumdarsrijoni@gmail.com	9674474267
Himadri B G S Bhuyan, *TA*	himadribhuyan@gmail.com	9438911655

Module 30: Object Oriented Analysis & Design

Sequence Diagrams: Part 2

Partha Pratim Das

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

ppd@cse.iitkgp.ernet.in
Tanwi Mallick
Srijoni Majumdar
Himadri B G S Bhuyan

This presentation uses diagrams, examples and selected texts from *Object-Oriented Analysis and Design – With Applications* by Grady Booch et. al. (3rd Ed, 2007) with kind permission of the author

Module Objectives

- Understanding Sequence Diagrams
- Sequence diagram for Leave Management System (LMS)

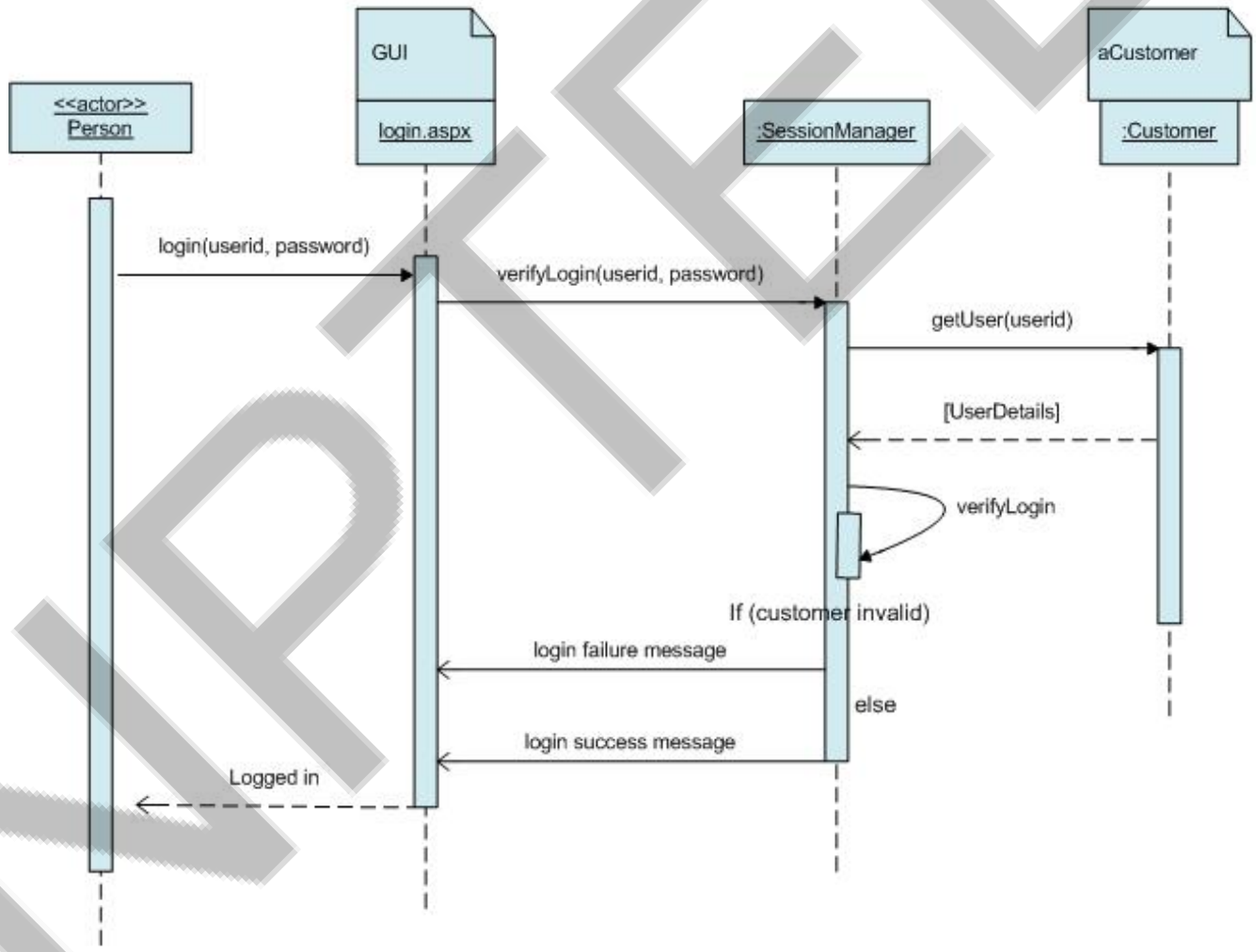
Module Outline

- What are Sequence Diagrams?
  - Lifeline
  - Messages
  - Interaction Fragments
  - Examples
- Sequence Diagram for LMS

Sequence Diagrams: RECAP (Module 29)

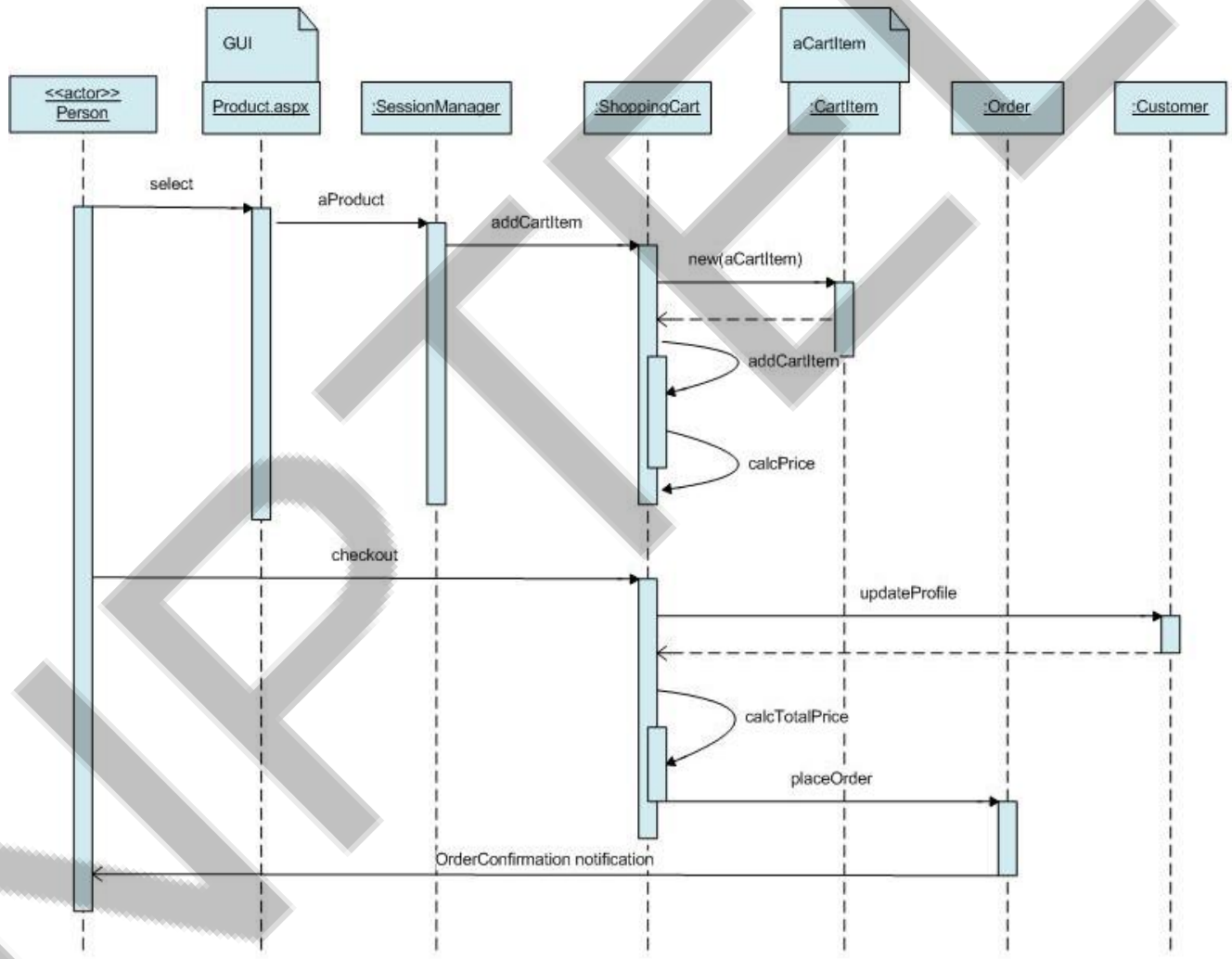
- The various objects of the system interact with each other, through exchange of messages to invoke the various operations of the object.
- Sequence diagram is a major diagram to depict the inter object behaviour of a system, ordered by time.
- Sequence diagram is the most common kind of interaction diagram, which focuses on the message interchange between a number of lifelines.
- The major components of a Sequence Diagram are
  - Lifeline
  - Messages
  - Interaction Fragments

Example: Login: RECAP (Module 29)



Source: http://people.cs.ksu.edu/ reshma/798_SequenceDiagram.htm (18-Aug-16)

Example: Place Order: RECAP (Module 29)



Source: http://people.cs.ksu.edu/ reshma/798_SequenceDiagram.htm (18-Aug-16)

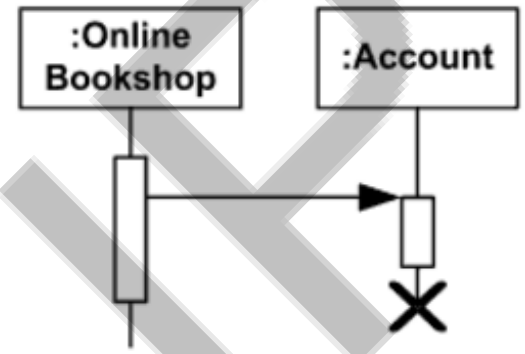
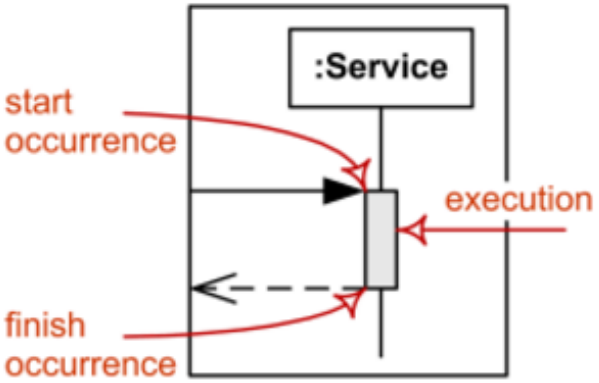
Interaction Fragments

- Interaction fragment is a named element representing the most general interaction unit
- Each interaction fragment is conceptually like an interaction by itself
- There is no general notation for an interaction fragment. Its sub-classes define their own notation
- Examples of Interaction Fragments include:
  - Occurrence
  - Execution
  - State invariant
  - Combined fragment
  - Interaction use

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Interaction Fragment: Occurrence

Occurrence is interaction fragment which represents a moment in time (event) at the beginning or end of a message or at the beginning or end of an execution

Message occurrence represents events as sending and receiving of signals **Destruction occurrence** destruction of the instance described by the life-line	**Execution occurrence** represents moments in time at which actions or behaviors start or finish.
 Account lifeline is terminated	 Duration of an execution is represented by two execution occurrences - start and finish

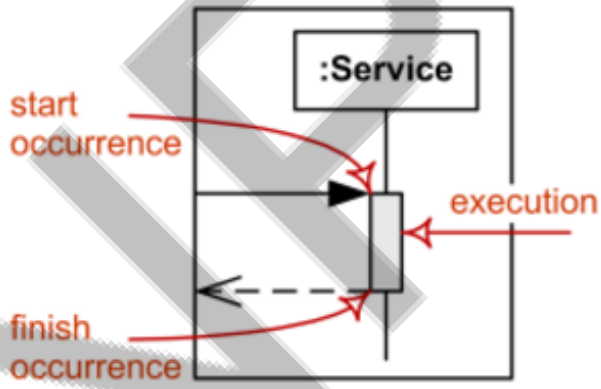
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Interaction Fragment: Execution

Execution (Activation) is an interaction fragment which represents a period in the participant's lifetime when it is

- executing a unit of behavior or action within the lifeline, or
- sending a signal to another participant, or
- waiting for a reply message from another participant

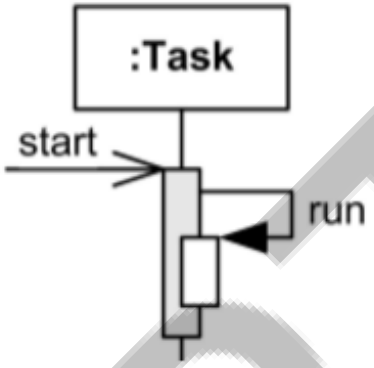
Execution is represented as a thin grey or white rectangle on the life-line	Execution can be represented by a wider labeled rectangle, where the label identifies the action



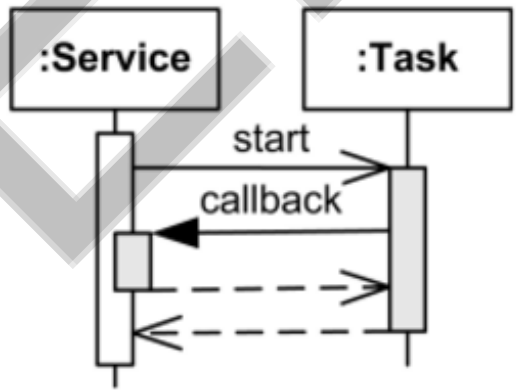
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Interaction Fragment: Execution

Overlapping execution specifications on the same lifeline are represented by overlapping rectangles



Overlapping execution specifications on the same lifeline - message to self

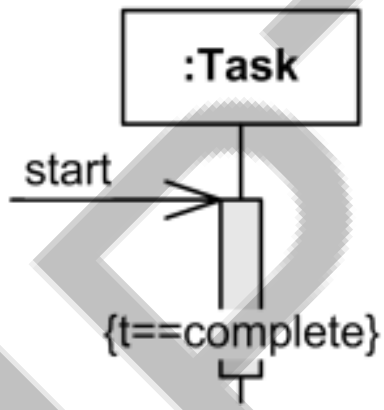
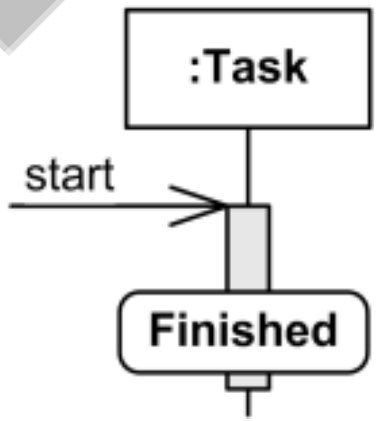


Overlapping execution specifications on the same lifeline - callback message.

Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Interaction Fragment: State Invariant

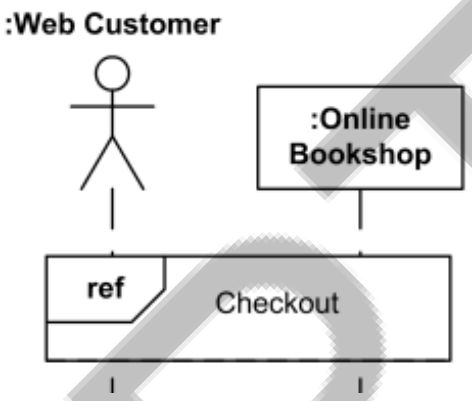
State Invariant is an interaction fragment which represents a run-time constraint on the participants of the interaction. It may be used to specify different kinds of constraints, such as values of attributes or variables, internal / external states, etc.

State invariant is usually shown as a constraint in curly braces on the lifeline	State invariant may be shown as a state symbol
	
Attribute t of Task should be equal to complete	Task should be in Finished state

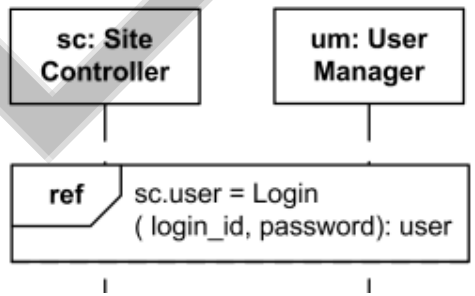
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Interaction Fragment: Interaction Use

Interaction Use is an an interaction fragment which allows to use (or call) another interaction. Large and complex sequence diagrams could be simplified with interaction uses. It is also common to reuse some interaction between several other interactions



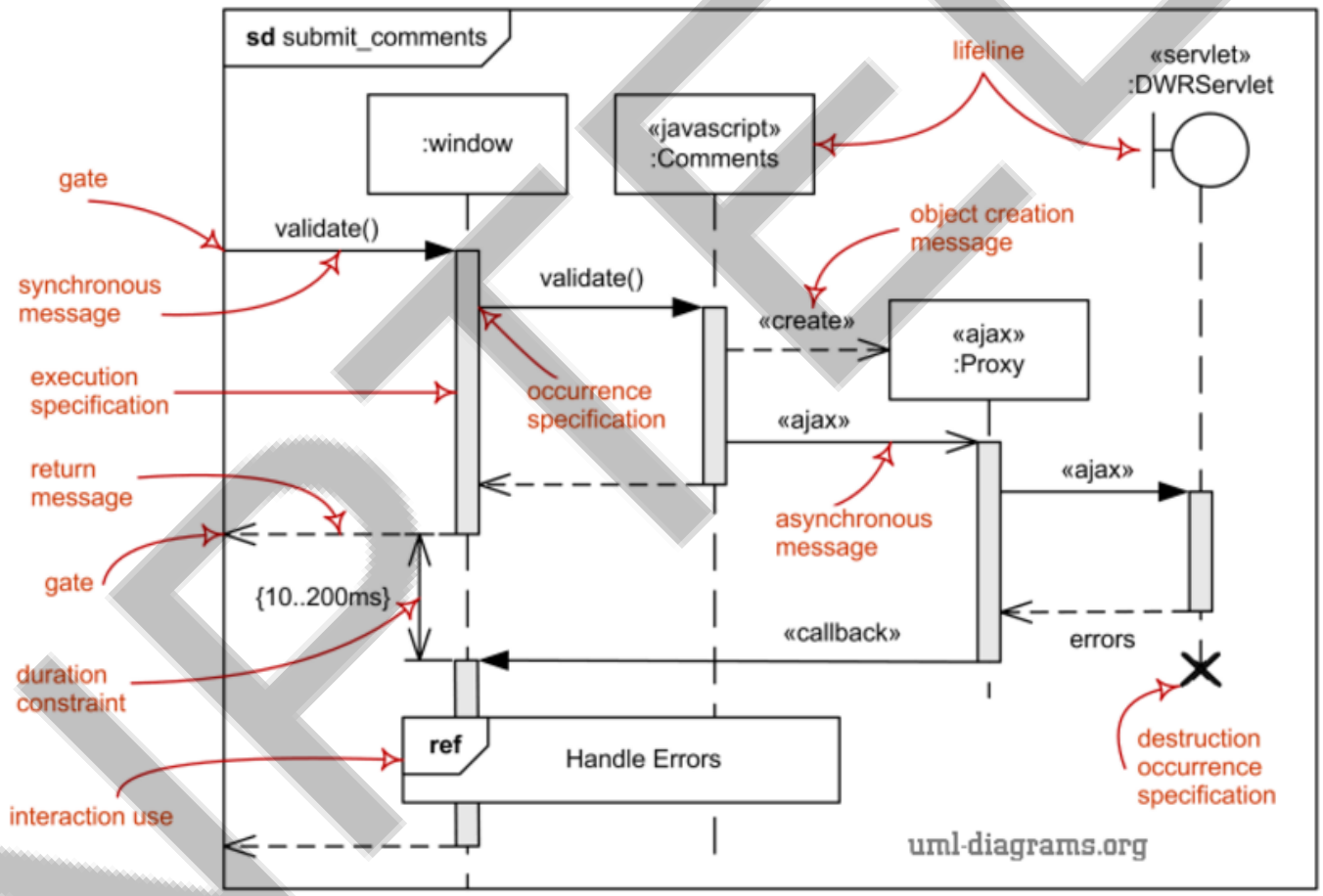
Web customer and Bookshop use (reference) interaction Checkout



Use Login interaction to authenticate user and assign result back to the user attribute of Site Controller

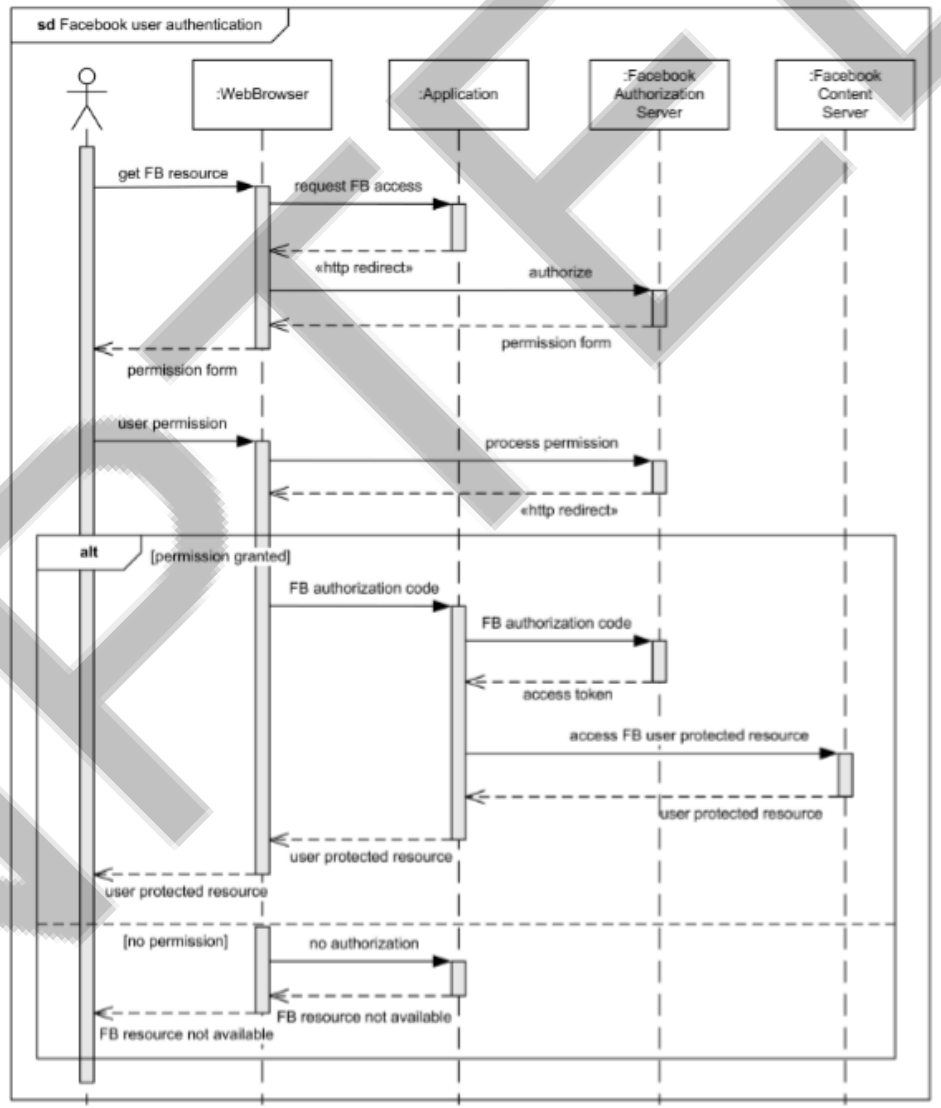
Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

An Annotated Sequence Diagram



Source: *UML 2.5 Diagrams Overview*: http://www.uml-diagrams.org/uml-25-diagrams.html (18-Aug-16)

Example: Facebook Authentication

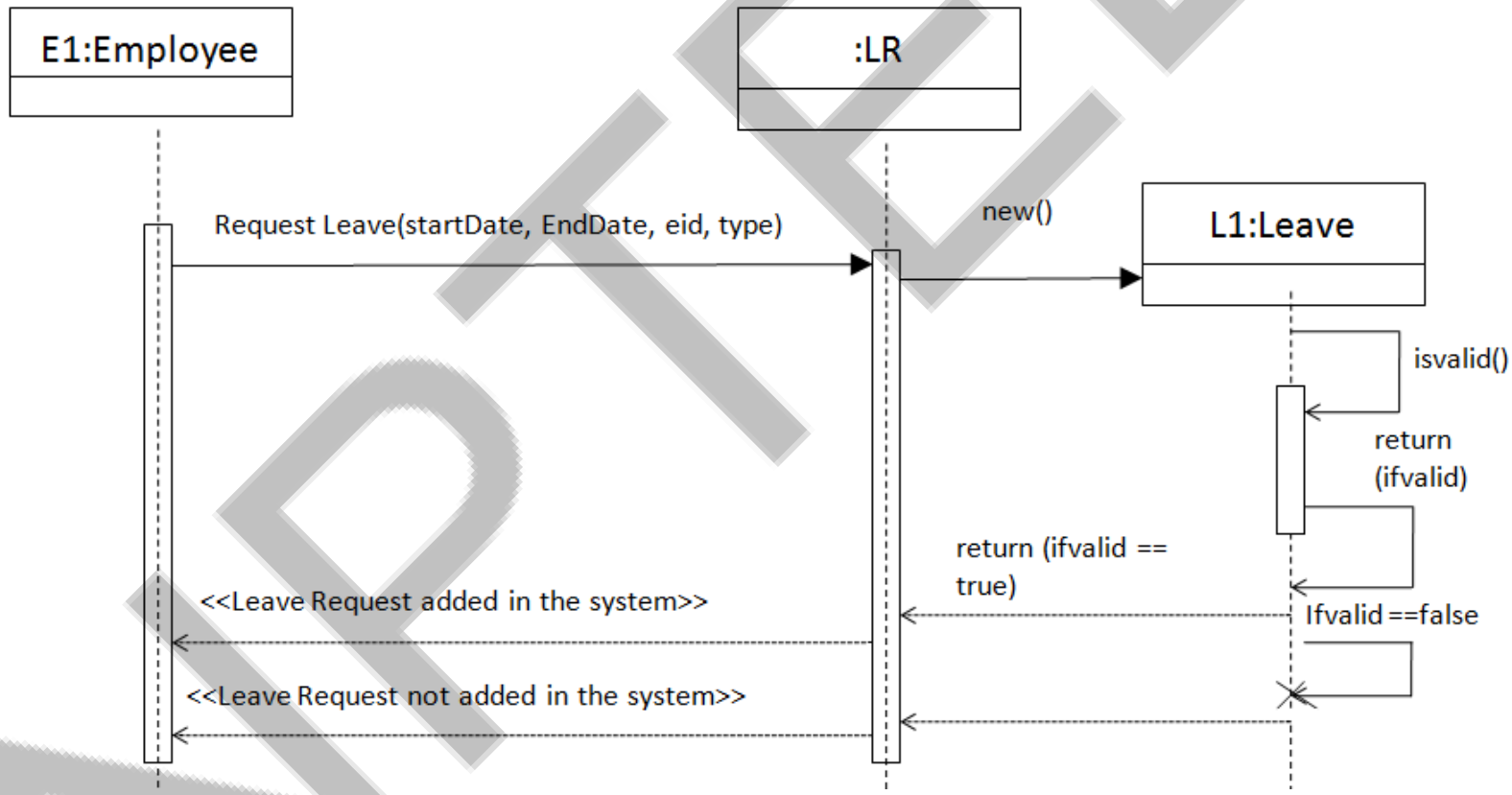


Identifying the Major Elements and their Lifelines of the Sequence Diagram

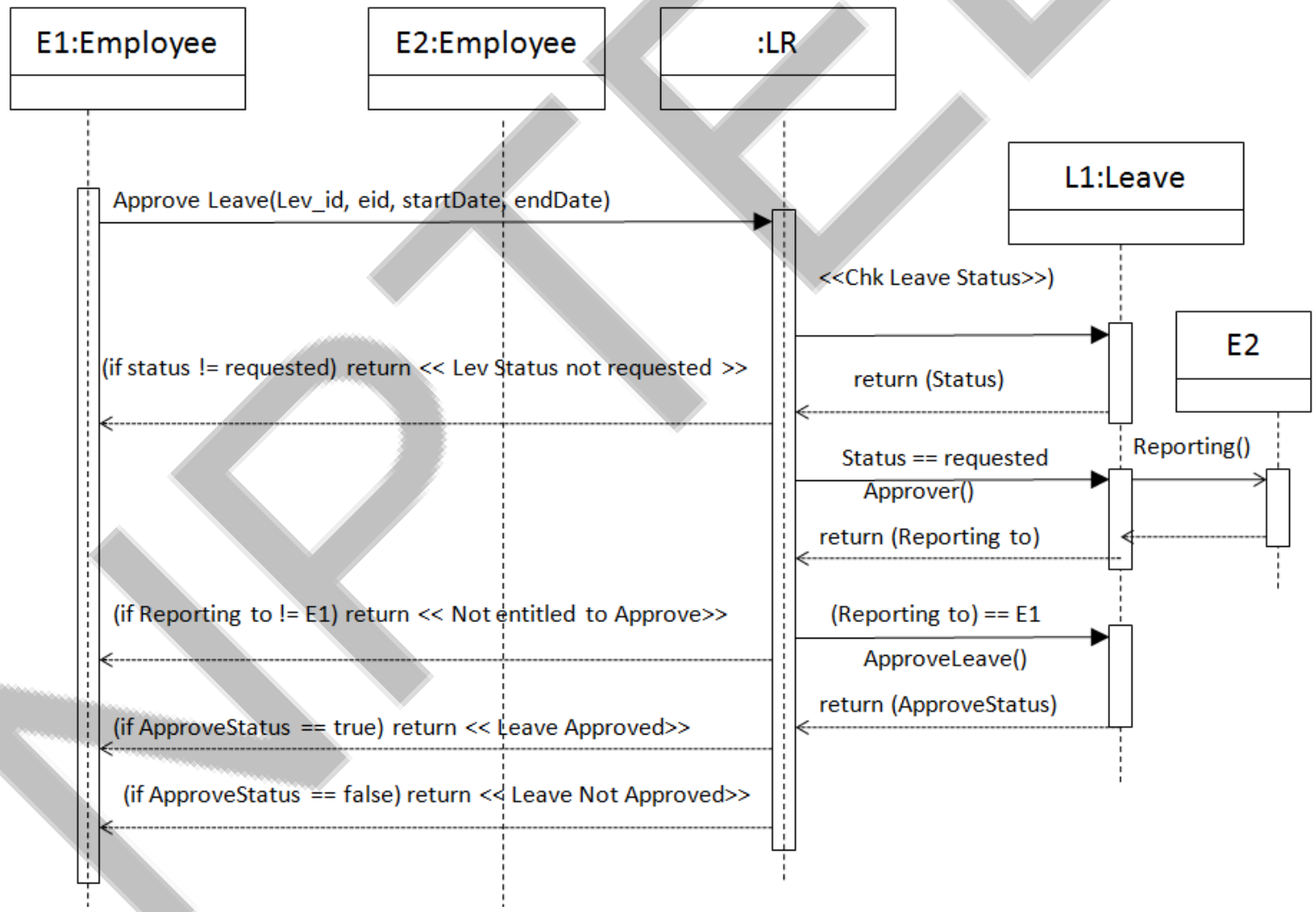
- Reading through the specification of the Leave Management System, we identify the major identifying elements for LMS : Employee and Leave.

- In addition, we have an LR class (Leave Record / Repository) to maintain all leave data

Sequence Diagram for LMS: Request Leave



Sequence Diagram for LMS: Approve Leave



Module Summary

- Discussed about interaction fragments in Sequence Diagrams
- Worked out the sequence diagram for two sample use-cases of LMS

Instructor and TAs

Name	Mail	Mobile
Partha Pratim Das, *Instructor*	ppd@cse.iitkgp.ernet.in	9830030880
Tanwi Mallick, *TA*	tanwimallick@gmail.com	9674277774
Srijoni Majumdar, *TA*	majumdarsrijoni@gmail.com	9674474267
Himadri B G S Bhuyan, *TA*	himadribhuyan@gmail.com	9438911655