**NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES**

**SL-2002 – Software Design & Architecture Lab**

**Lab Instructor: Zarnain Maryam Awan**

Lab 03

**Domain Model:**

∙ The domain model is created during object-oriented analysis to decompose the domain into concepts or objects in the real world.

∙ Domain Model illustrates meaningful conceptual classes in a problem

domain. It is a representation of real-world concepts, not software

components.

∙ It is NOT a set of diagrams describing software classes, or software objects and their responsibilities.

∙ It is the basis for the design of the software

∙ In UML, the Domain Model is illustrated with a set of class diagrams without methods

∙ Why: Domain modeling helps us to identify the relevant concepts and ideas of a domain

∙ When: Domain modeling is done during object-oriented analysis.

**Domain Model covers:**

o The conceptual or domain classes

o Attributes of the conceptual classes

o Associations between the conceptual classes

**STEPS TO CREATE A DOMAIN MODEL**

1. Identify candidate conceptual classes

2. Draw them in a UML domain model

3. Add associations necessary to record the relationships that must be retained

4. Add attributes necessary for information to be preserved

**Example (Library Scenario)**

▪ A library has one or many librarians

▪ A borrower borrows zero or many books from the library

▪ No or many books stock by zero or many librarians

▪ Zero or many borrowers visit no or more times a library

**Domain Model For The Library Scenario**

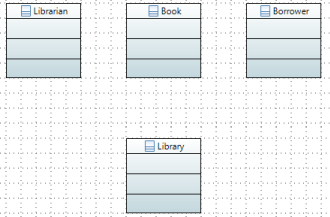
**1.Identifying Conceptual Classes**

A conceptual model captures the important concepts and relationships in some domain. Concepts are represented by classes, while relationships are represented by associations.

**Conceptual or Domain Class:**

▪ Each domain class denotes a type of object

▪ Consider a use case description

The conceptual classes of the above scenario may be the following

**2.Adding Associations**

**Association Relationship**

A link between two classes

For example: A person works for a company

An association can be defined as a relationship between classes. An association relation is established when two classes are connected to each other in any way. A class can be associated with itself too.

**Examples of Associations:**

**a. Association with 1 multiplicity:**

Assume some application needs to associate a home and business phone to each person in a database.

Example: A person has two phone numbers, one for office and one for home, both are private. This relationship can be represented by following association and code:



**Code:**

class Person {

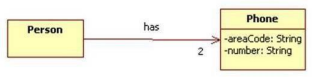
private Phone home;

private Phone office;

// etc.

}

**b. Association with more than 1 multiplicity:**

Taking the example described above, we can also model the scenario as follows: This can be represented in code as follows:

class Person {

private Phone[] phones = new

Phone[2];

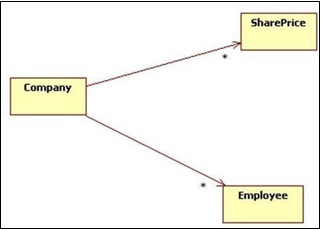
// etc.

}

**c. Association with many (\*) multiplicity:**

Take the example that a company can have many employees and many

share prices.



Code:

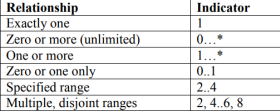
class Company {

private List<Employee> employees;

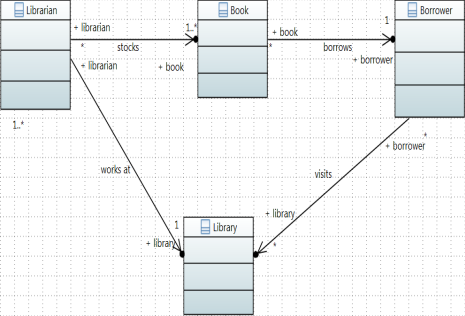
private List<SharePrice> sharePrices;

// etc.

The association relationship is further described by the multiplicity concept.



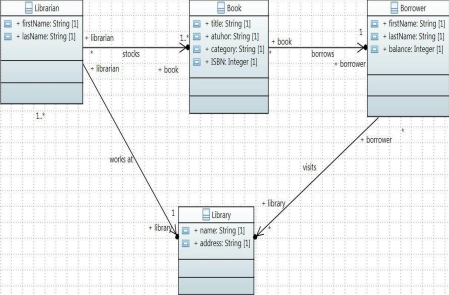
Following the above scenario, association with multiplicity is added



**3.Adding Attributes:**

**Attributes:**

Attributes refer to property that define the class.

****