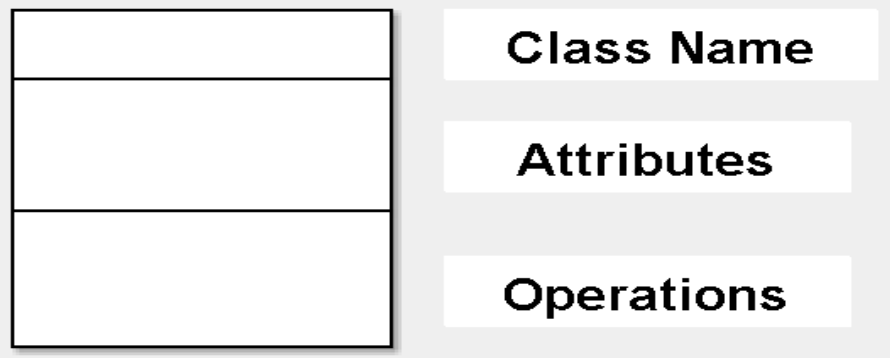
1. UML CLASS DIAGRAM:

- It shows the static structure of the model
- It is type of **UML-STATIC DIAGRAM**
- It is a collection of static modeling elements like classes and their relationships, connected as a graph tp each other and to their contents
- Class diagrams does not show temporal information, which is required in dynamic modeling
- A class diagram is drawn as a rectangle with three components, separated by horizontal lines.

SYMBOLS:

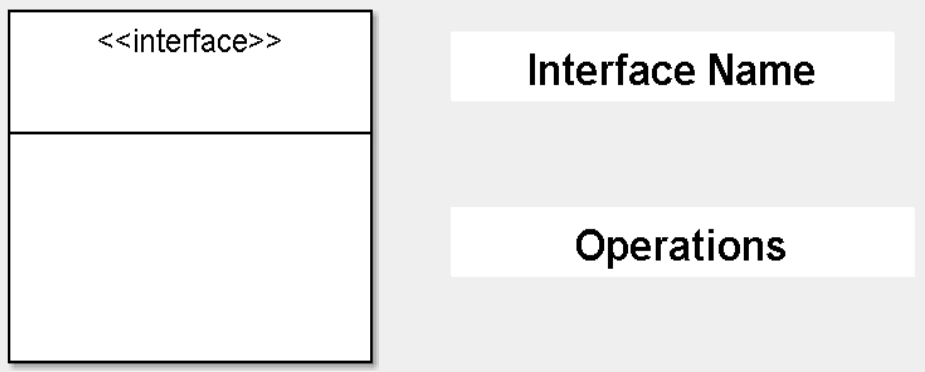
1. **Class**
   - Set of attributes and operations
   - It has three components

   Symbol



2. **Interface**
   - Set of operations only
   - Has two components

Symbol



3. **Association**
4. **Multiplicity**

0 ← exactly zero

1 ← exactly one

1..* ← one or more

0..* ← zero or more

n ← many

5. **Generalization**

- It creates an inheritance relationship
- It has is-a relationship

Symbol 

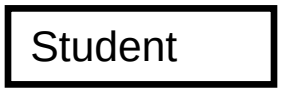
6. **Aggregation**
7. **Composition**

THREE VIEWS OF CLASS IN CLASS DIAGRAM

1. CONCEPT VIEW

- It identify the software class in application
- It is noun (Object Oriented Analysis =OOA)
- Has only class name no attributes and operations
- It is optional

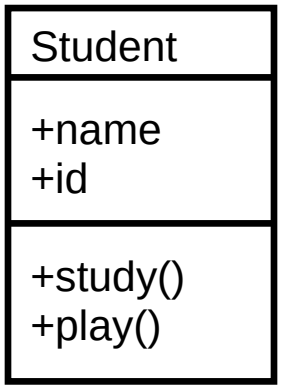
Ex:



2. SOFTWARE VIEW

- It is nothing but OOA with details (OOD)
- It is optional

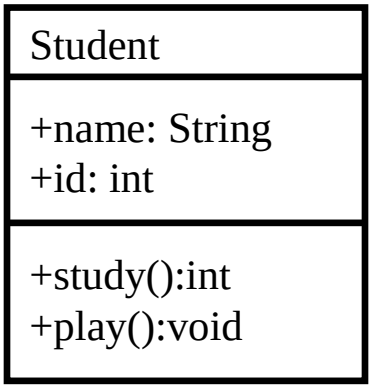
Ex:



3. IMPLEMENTATION VIEW

- Programming types with specification views
- It is optional

Ex:

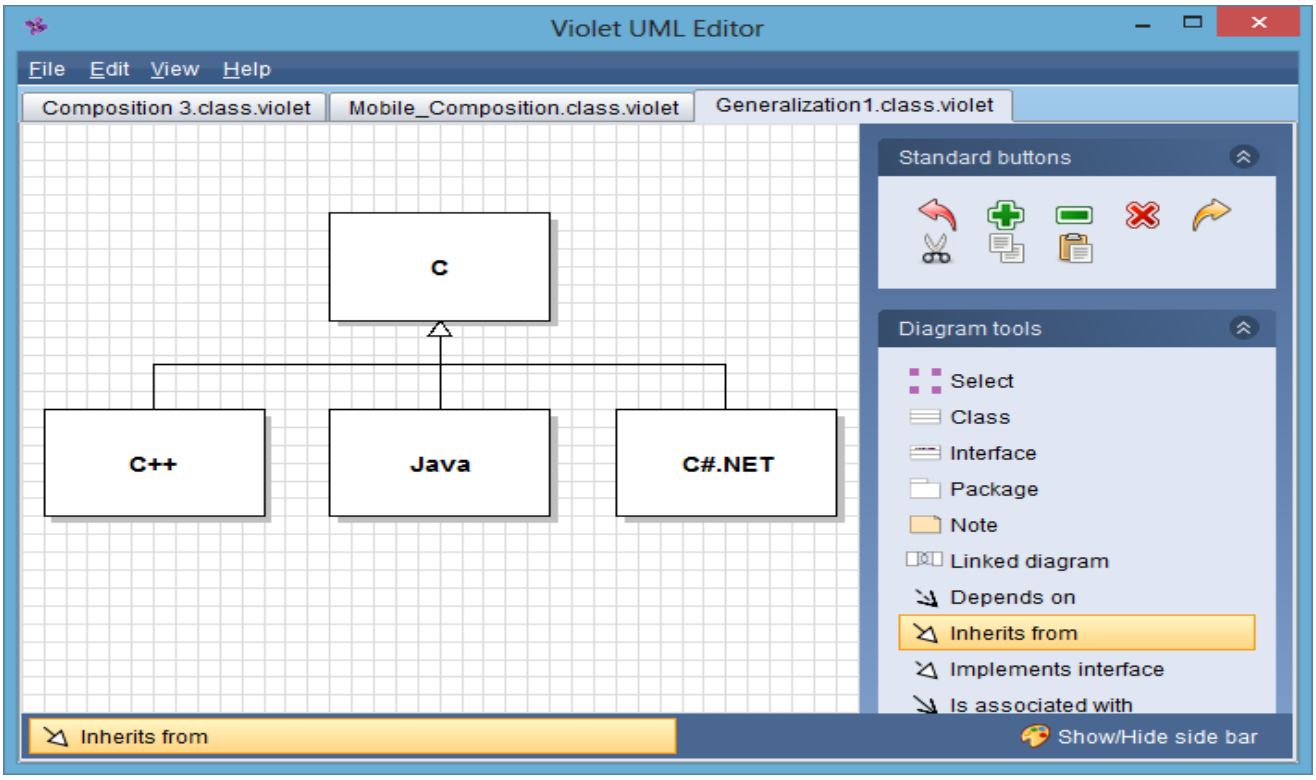


GENERALIZATION

- It creates an inheritance relationships
- It has **is-a** relationship
- Symbol: 

I. GENERALIZATION EXAMPLES

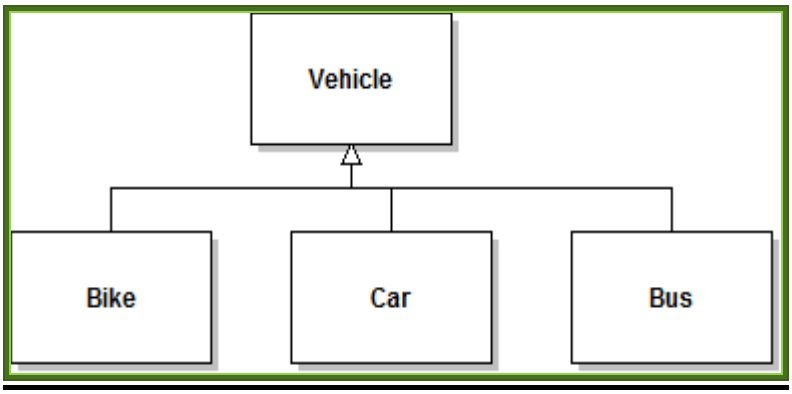
Ex: 1



NOTE:

- Here C is a super class and C++, Java and C#.NET is a sub classes of super class
- All sub classes (C++, Java and C#.NET) are derived from a single class called "C"

Ex: 2

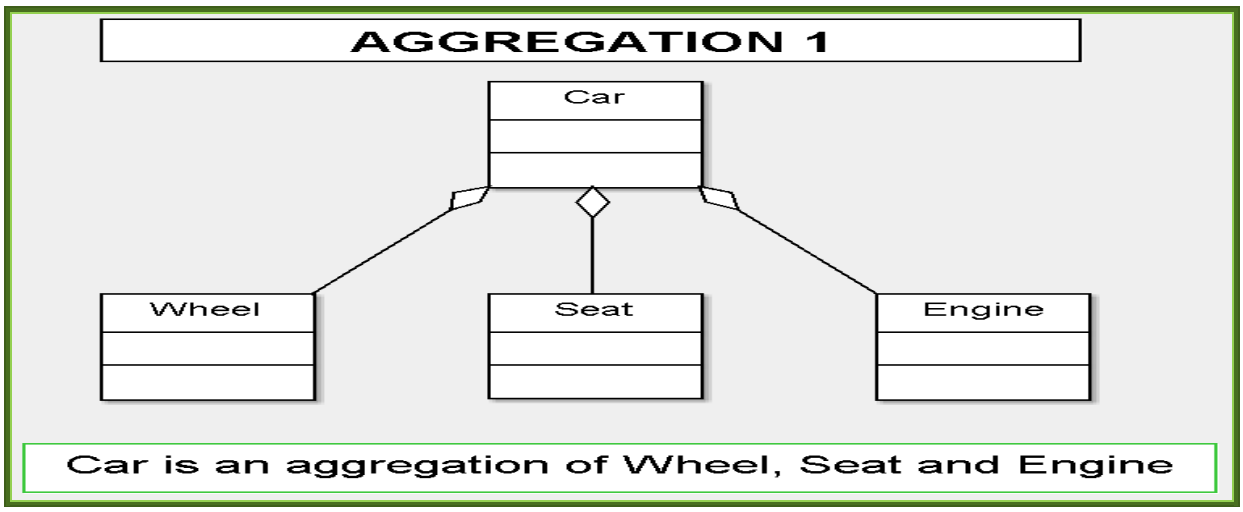


NOTE:

- Here C is a super class and Bike, Car and Bus is a sub classes of super class
- All sub classes (Bike, Car and Bus) are derived from a single class called "Vehicle"

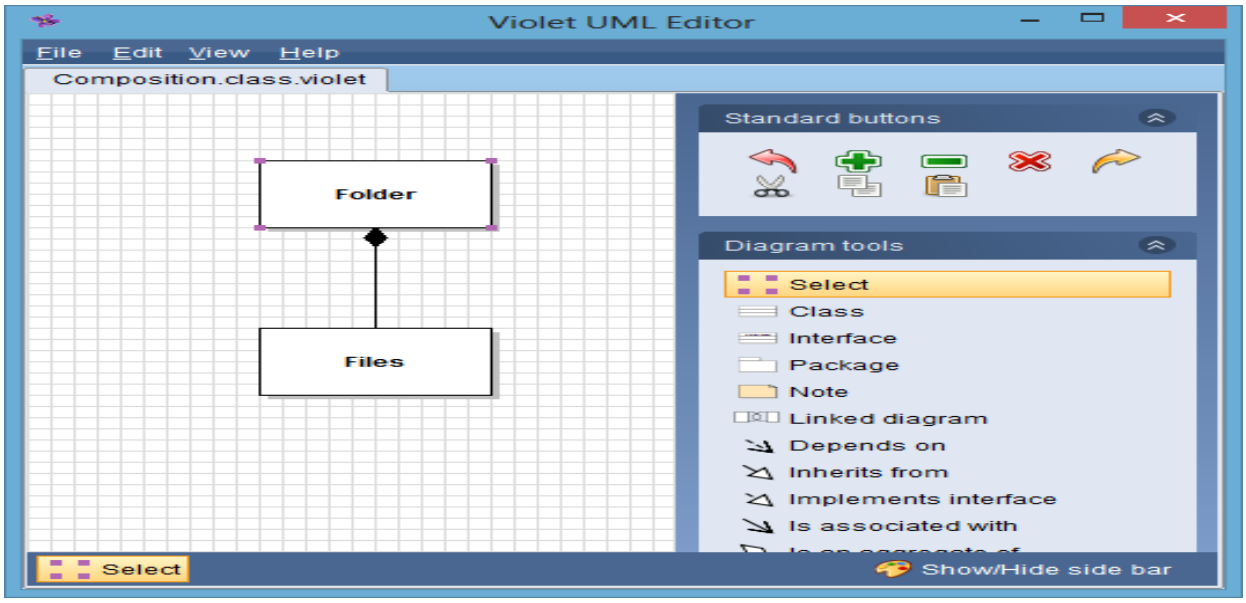
II. AGGREGATION EXAMPLES

Ex: 3



III. COMPOSITION EXAMPLES

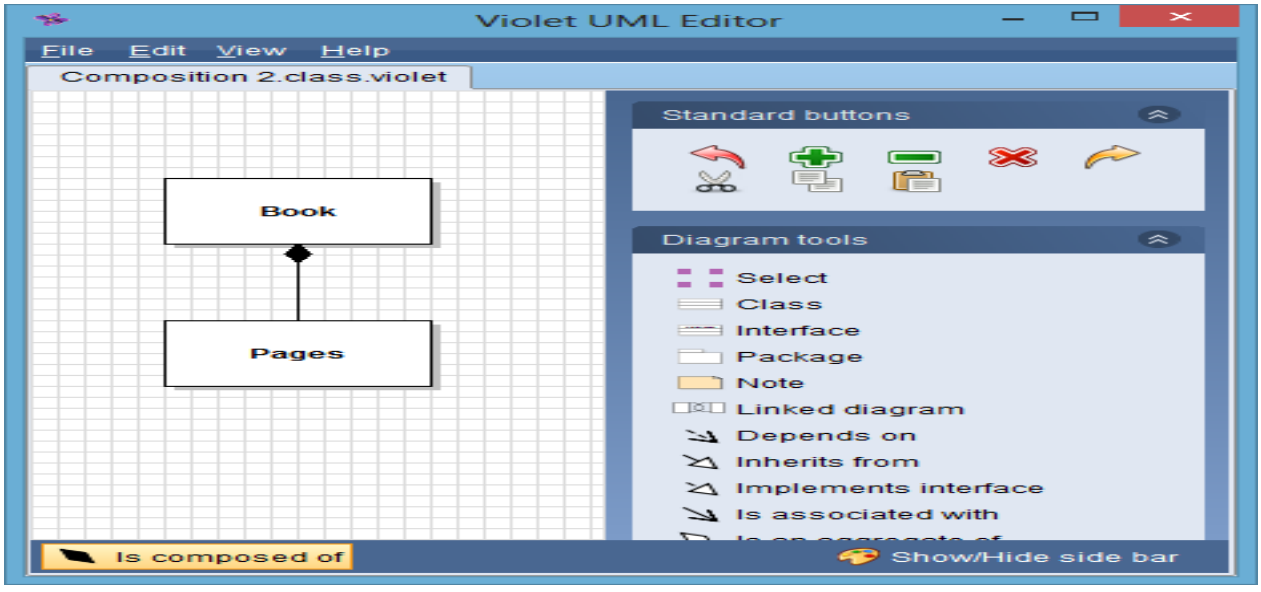
Ex:4 [COMPOSITION]



NOTE:

- If Folder(Container Object) is detroyed then Files (contained object) will be deleted
- Because composition has a **strong-life-dependancy.**

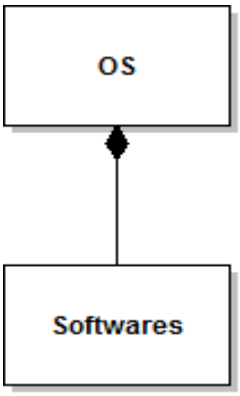
Ex:5 [COMPOSITION]



NOTE:

- If Book(Container Object) is detroyed then **Pages**(contained object) will be deleted
- Because composition has a **strong-life-dependancy.**

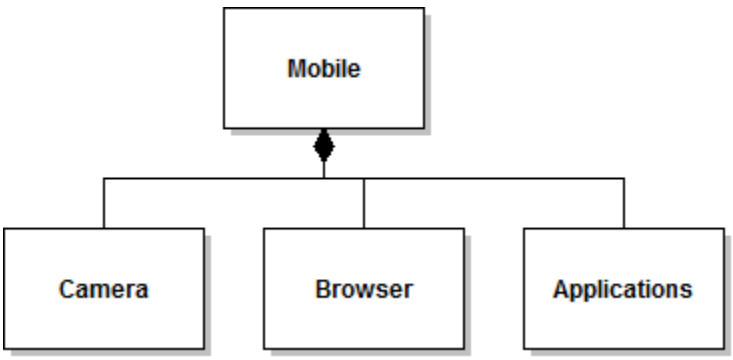
Ex:6 [COMPOSITION]



NOTE:

- If Os(Container Object) is detroyed then **Softwares**(contained object) will be deleted
- Because composition has a **strong-life-dependancy.**

Ex: 7 [COMPOSITION]

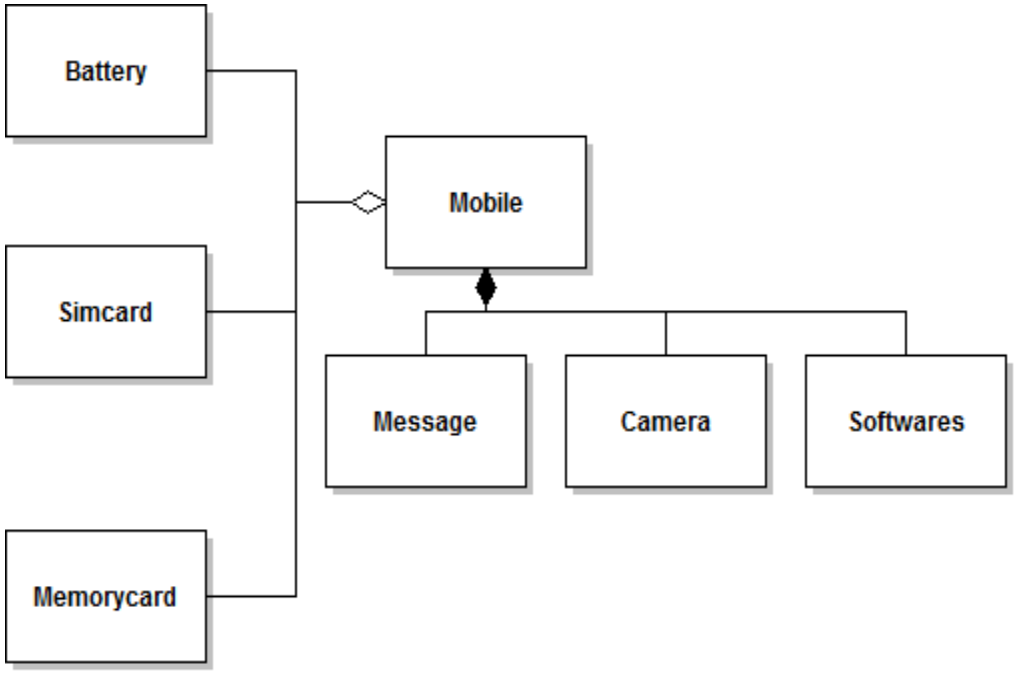


NOTE:

- If Mobile(Container Object) is detroyed then **Camera, Browser and Applications**(contained object) will be deleted
- Because composition has a **strong-life-dependancy.**

AGGREGATION vs COMPOSITION

Ex: 1 [AGGREGATION vs COMPOSITION]



NOTE:

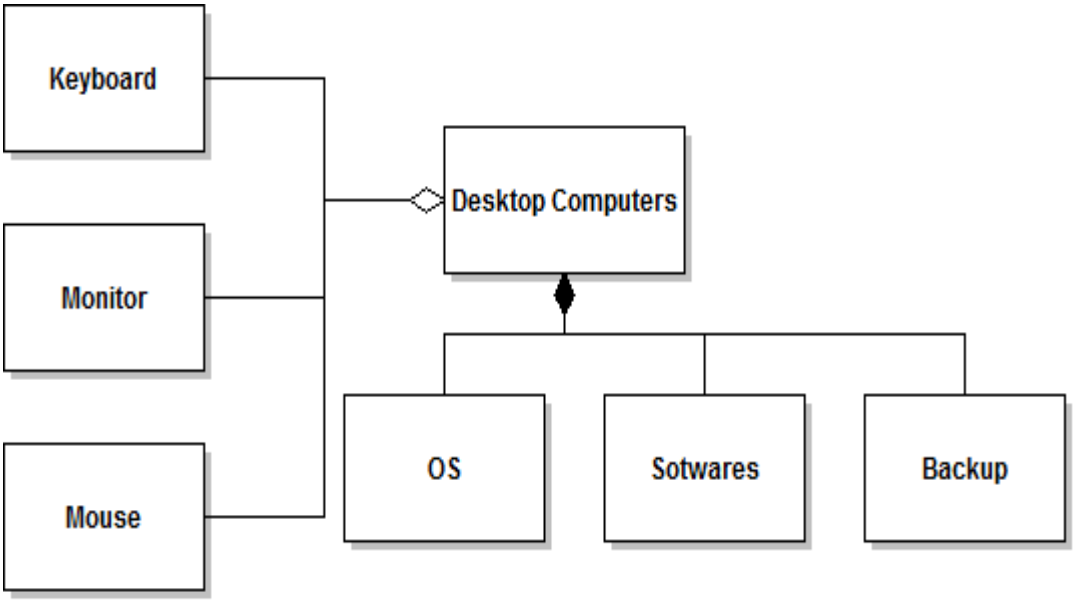
COMPOSITION

- If the Mobile(Container Object) is destroyed/deleted, then its contents(Contained Object: **Message, Camera, Softwares**) **will be destroyed** beacause of strong-life cycle dependancy

AGGREGATION

- If the Mobile(Container Object) is destroyed/deleted, then its contents(Contained Object: **Battery, Simcard, Memorycard**) **will not be destroyed**

Ex:2 [AGGREGATION vs COMPOSITION]



NOTE:

COMPOSITION

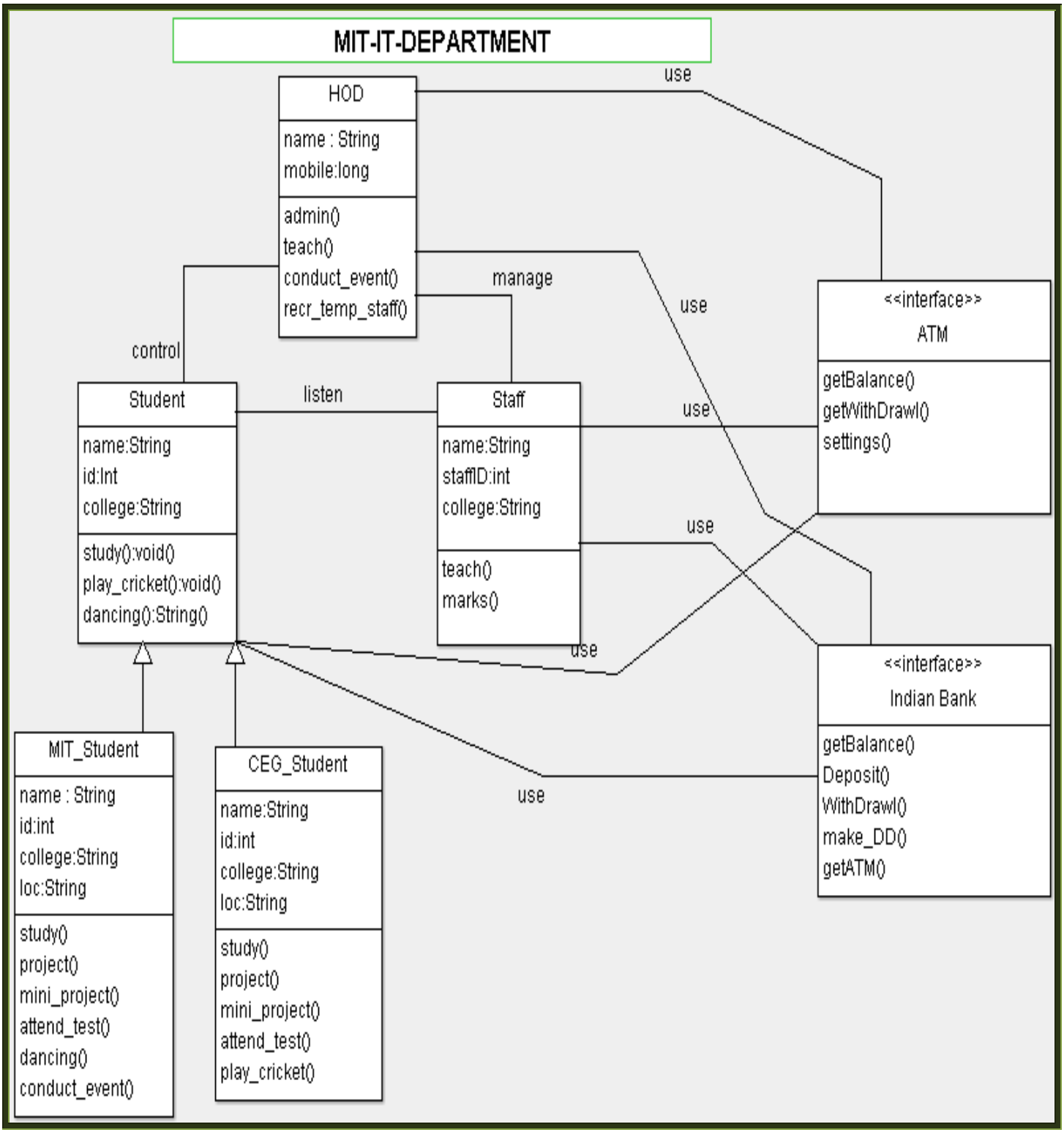
- If the **Desktop Computers** (Container Object) is destroyed/deleted, then its contents (Contained Object: **OS, Softwares, Backup**) **will be destroyed** beacause of strong-life cycle dependancy

AGGREGATION

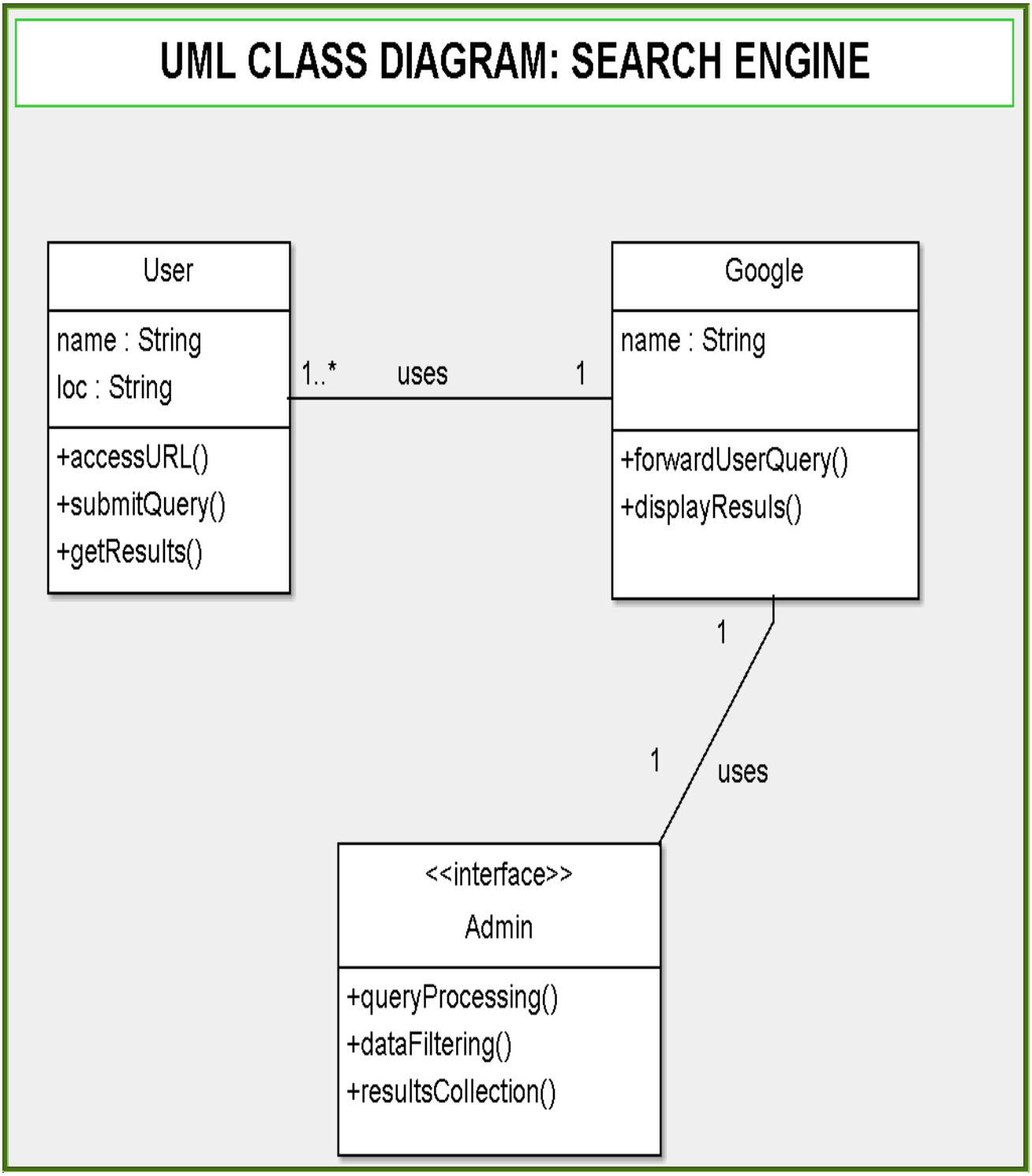
- If the **Desktop Computers** (Container Object) is destroyed/deleted, then its contents(Contained Object: **OS, Softwares, Backup**) **will not be destroyed**

I. UML CLASS DIAGRAM EXAMPLES

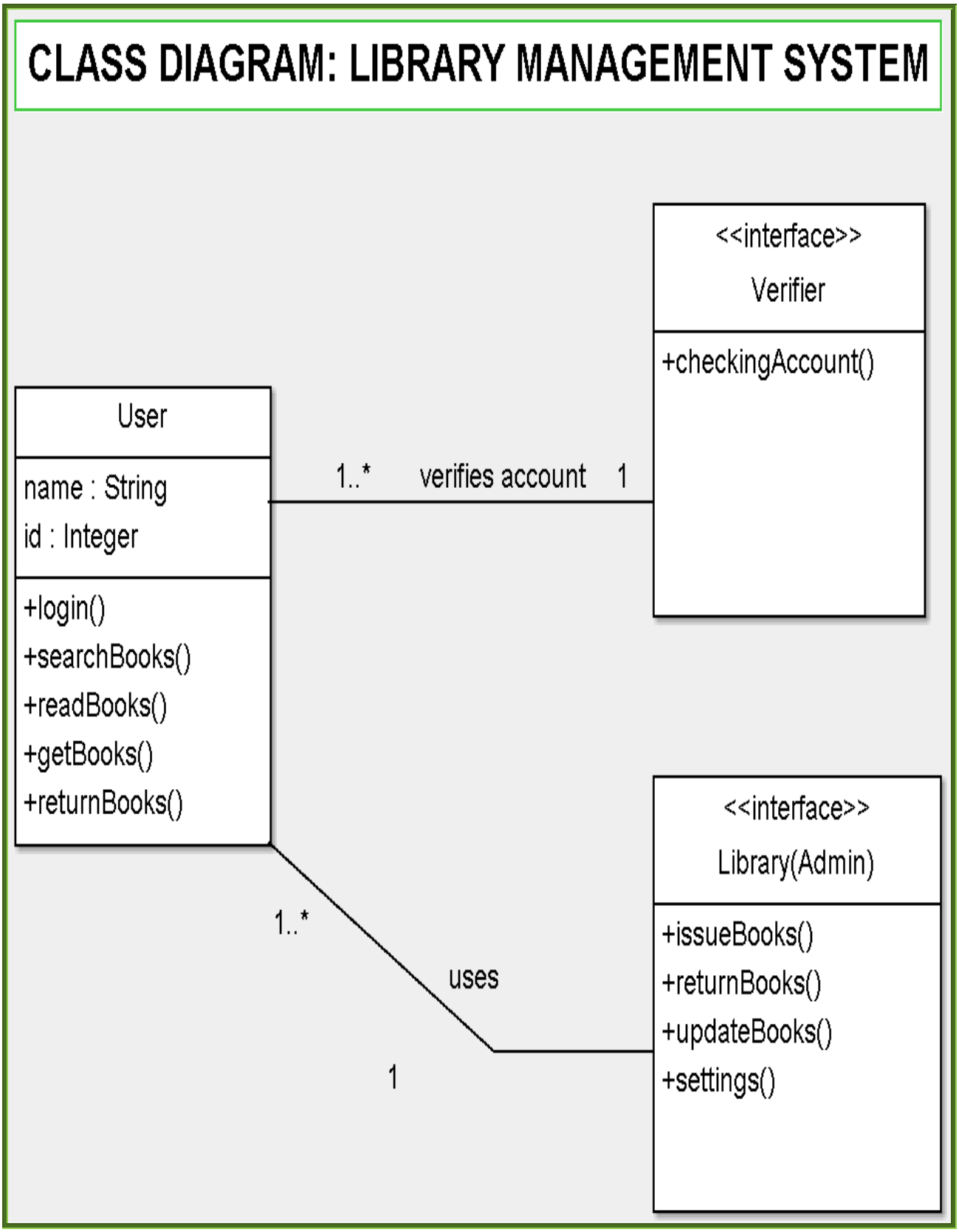
1. IT DEPARTMENT:



2. SEARCH ENGINE:



3. LIBRARY MANAGEMENT SYSTEM:



4. HOSPITAL MANAGEMENT SYSTEM: