UML Class Diagrams part 2

Idaho State University | Computer Science

Isaac Griffith

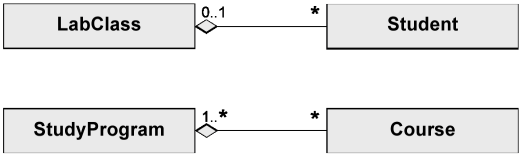
CS 3321
Department of Computer Science
Idaho State University

Outline

- UML Class Diagrams
- UML Object Diagrams

Shared Aggregation

- Expresses a weak belonging of the parts to a whole
  - Parts also exist independently of the whole
- Multiplicity at the aggregating end may be > 1
  - One element can be part of multiple other elements
- Spans a directed acyclic graph
- Syntax: Hollow diamond at the aggregating end
- Example:
  - `Student` is part of `LabClass`
  - `Course` is part of `StudyProgram`



Composition

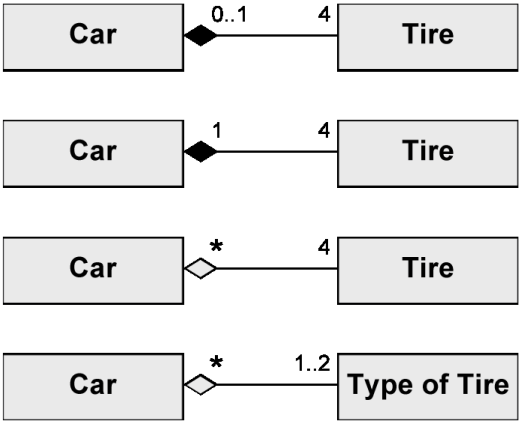
- Existence dependency between the composite object and its parts
- One part can only be contained in at most one composite object at one specific point in time
  - Multiplicity at the aggregating end max: 1
  - The composite objects form a tree
- If the composite object is deleted, its parts are also deleted
- Syntax: Solid diamond at the aggregating end
- Example: `Beamer` is part of `LectureHall` is part of `Building`

If the `Building` is deleted, the `LectureHall` is also deleted

The `Beamer` can exist without the `LectureHall`, but if it is contained in the `LectureHall` while it is deleted, the `Beamer` is also deleted

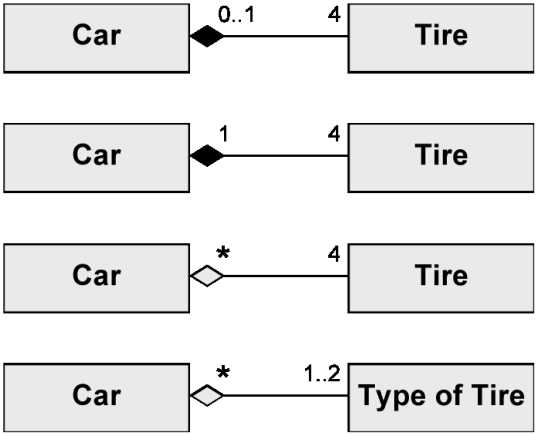
Shared Aggregation and Composition

- Which model applies?



Shared Aggregation and Composition

- Which model applies?



A `Tire` can exist without a `Car`. A `Tire` belongs to one `Car` at most. ---Yes---

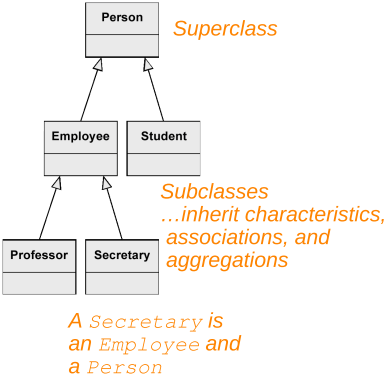
A `Tire` cannot exist without a `Car`. ---No---

A `Tire` can belong to multiple `Car`s ---No---

A `Car` has one or two types of `Tire`s. Several `Car`s may have the same Type of `Tire`s. ---Yes---

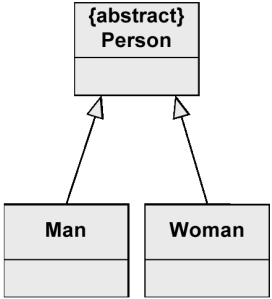
Generalization

- Characteristics (attributes and operations), associations, and aggregations that are specified for a general class (superclass) are passed on to its subclasses.
- Every instance of a subclass is simultaneously an indirect instance of the superclass.
- Subclass inherits all characteristics, associations, and aggregations of the superclass except private ones.
- Subclass may have further characteristics, associations, and aggregations.
- Generalizations are transitive.

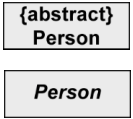


Abstract Class

- Used to highlight common characteristics of their subclasses.
- Used to ensure that there are no direct instances of the superclass.
- Only its non-abstract subclasses can be instantiated.
- Useful in the context of generalization relationships.
- Notation: keyword `{abstract}` or class name in italic font.



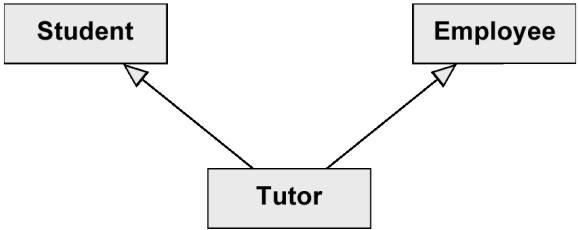
No `Person`-object possible



Two types of `Person`: `Man` and `Woman`

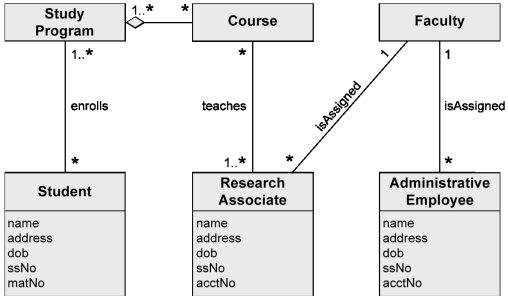
Multiple Inheritance

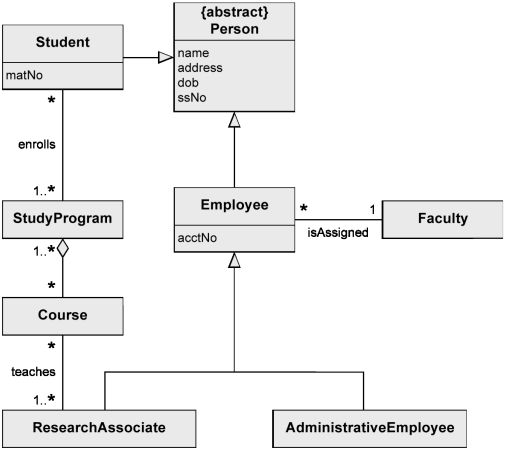
- UML allows multiple inheritance.
- A class may have multiple superclasses
- Example:



A `Tutor` is both an `Employee` and a `Student`

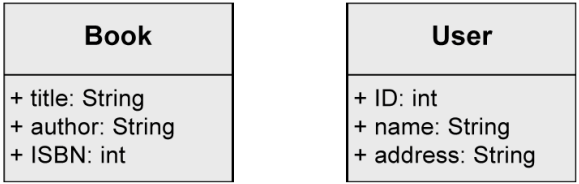
With/Without Generalization





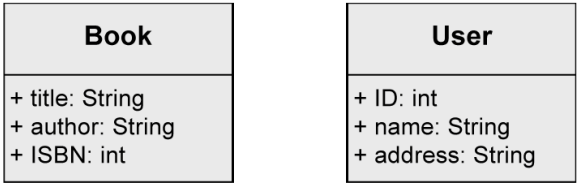
Creating a Class Diagram

- Not possible to completely extract classes, attributes and associations from a natural language text automatically.
- Guidelines
  - Nouns often indicate classes
  - Adjectives indicate attribute values
  - Verbs indicate operations
- Example: The library management system stores users with their unique ID, name and address as well as books with their title, author and ISBN number. Ann Foster wants to use the library.



Creating a Class Diagram

- Not possible to completely extract classes, attributes and associations from a natural language text automatically.
- Guidelines
  - Nouns often indicate classes
  - Adjectives indicate attribute values
  - Verbs indicate operations
- Example: The library management system stores users with their unique ID, name and address as well as books with their title, author and ISBN number. Ann Foster wants to use the library.



Question: What about Ann Foster?

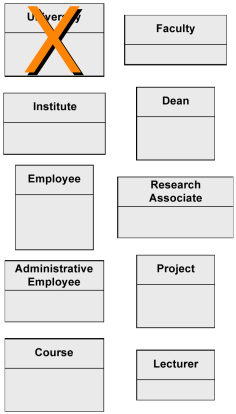
Example - University Info Sys

- A university consists of multiple faculties which are composed of various institutes. Each faculty and each institute has a name. An address is know for each institute.
- Each faculty is led by a dean, who is an employee of the university.
- The total number of employees is known. Employees have a social security number, a name, and an email address. There is a distinction between research and administrative personnel.
- Research associates are assigned to at least one institute. The field of study of each research associate is known. Furthermore, research associates can be involved in projects for a certain number of hours, and the name, starting date, and end date of the projects are known. Some research associates hold courses. Then they are called lecturers.
- Courses have a unique number (ID), a name, and a weekly duration in hours.

Step 1: Identify Classes

- A **university** consists of multiple **faculties** which are composed of various **institutes**. Each faculty and each institute has a name. An address is know for each institute.
- Each faculty is led by a **dean**, who is an **employee** of the university.
- The total number of employees is known. Employees have a social security number, a name, and an email address. There is a distinction between **research** and **administrative personnel**.
- Research associates are assigned to at least one institute. The field of study of each research associate is known. Furthermore, research associates can be involved in **projects** for a certain number of hours, and the name, starting date, and end date of the projects are known. Some research associates hold **courses**. Then they are called **lecturers**.
- Courses have a unique number (ID), a name, and a weekly duration in hours.

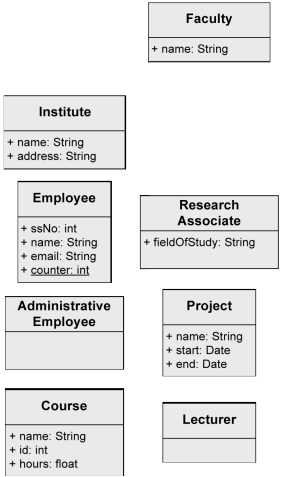
We model the system "University"



Dean has no further attributes than any other employee

Step 2: Identify Attributes

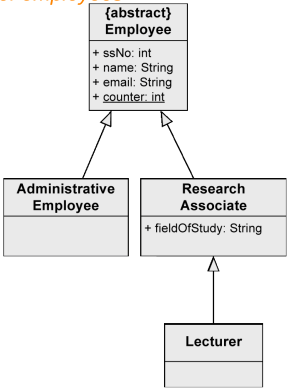
- A university consists of multiple faculties which are composed of various institutes. Each faculty and each institute has a **name**. An **address** is know for each institute.
- Each faculty is led by a dean, who is an employee of the university.
- The total **number of employees** is known. Employees have a **social security number**, a **name**, and an **email address**. There is a distinction between research and administrative personnel.
- Research associates are assigned to at least one institute. The field of study of each research associate is known. Furthermore, research associates can be involved in projects for a certain number of **hours**, and the **name**, **starting date**, and **end date** of the projects are known. Some research associates hold courses. Then they are called lecturers.
- Courses have a **unique number** (ID), a **name**, and a **weekly duration** in hours.



Step 2: Identify Relationships

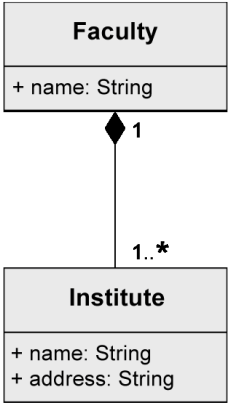
- Three kinds of relationships:
  - Association
  - Generalization
  - Aggregation
- Indication of generalization
- “There is a distinction between research and administrative personnel.”
- “Some research associates hold courses. Then they are called lecturers.”

Abstract, i.e., no other types of employees



Step 2: Identify Relationships

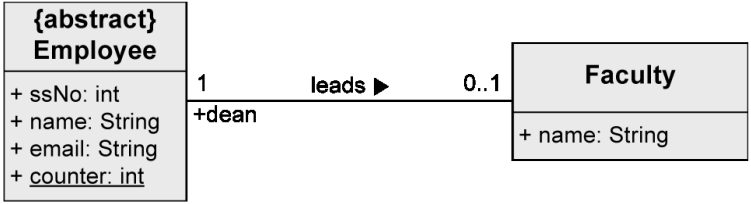
- "A university consists of multiple faculties which are composed of various institutes."



Composition to show existence dependency

Step 2: Identify Relationships

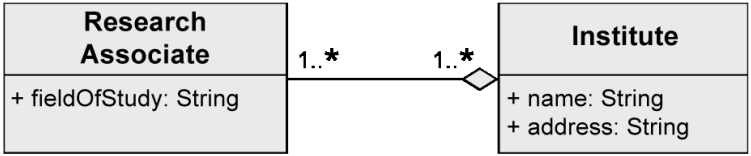
- "Each faculty is led by a dean, who is an employee of the university"



In the `leads`-relationship, the `Employee` takes the role of a `dean`.

Step 2: Identify Relationships

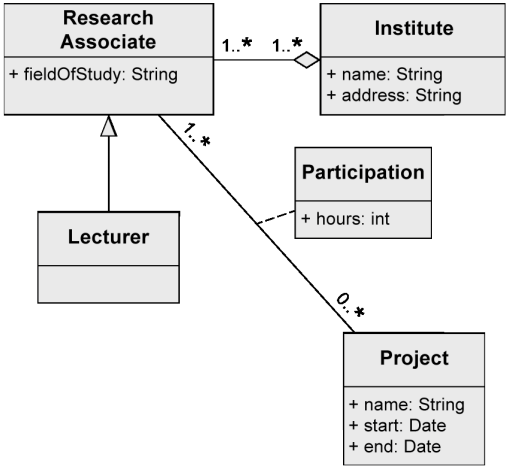
- "Research associates are assigned to at least one institute."



Shared aggregation to show that `ResearchAssociates` *are part of an* `Institute`*, but there is no existence dependency*

Step 2: Identify Relationships

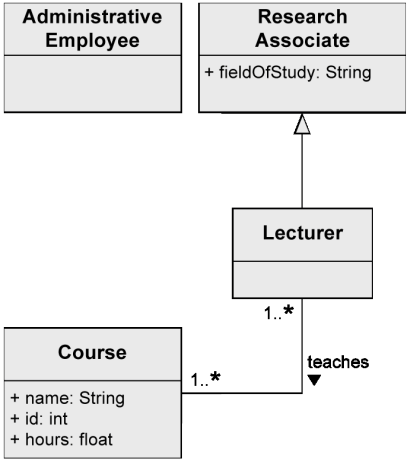
- "Furthermore, research associates can be involved in projects for a certain number of hours."



Association class enables to store the number of hours for every single `Project` of every single `ResearchAssociate`

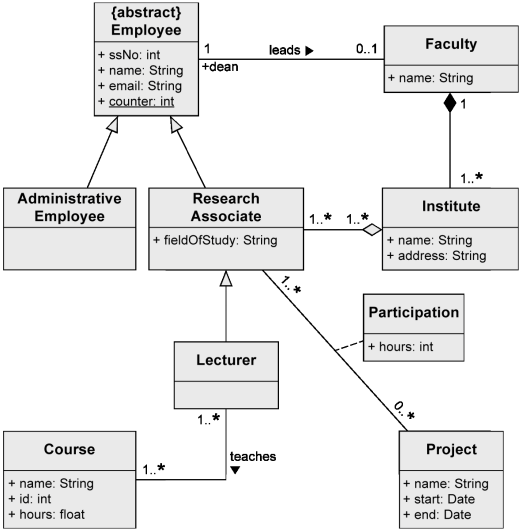
Step 2: Identify Relationships

- "Some research associates hold courses. Then they are called lecturers."



`Lecturer` inherits all characteristics, associations, and aggregations from `ResearchAssociate`.
In addtion, a `Lecturer` has an association `teaches` to `Course`.

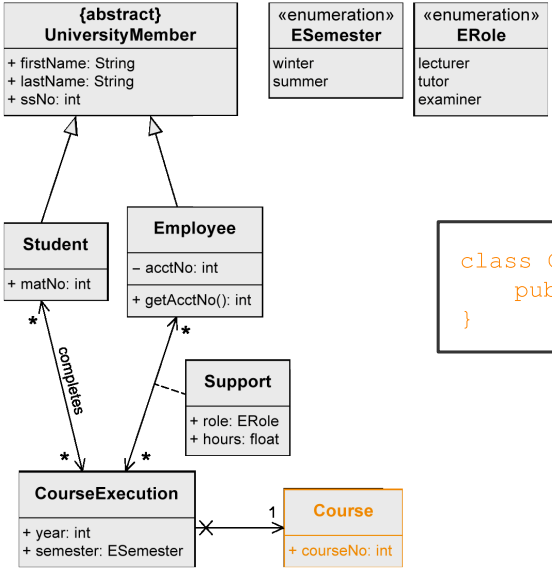
Complete Class Diagram



Code Generation

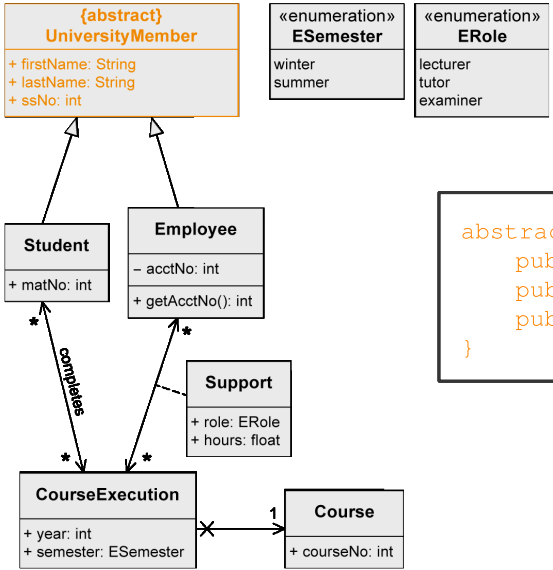
- Class diagrams are often created with the intention of implementing the modeled elements in an object-oriented programming language.
- Often, translation is semi-automatic and requires only minimal manual intervention.

Code Generation



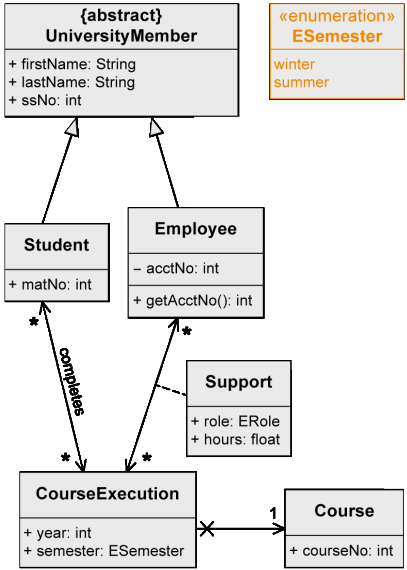

```
class Course {
    public int courseNo;
}
```

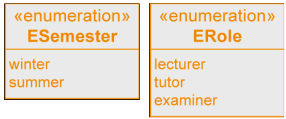
Code Generation




```
abstract class UniversityMember {
    public String firstName;
    public String lastName;
    public int ssNo;
}
```

Code Generation

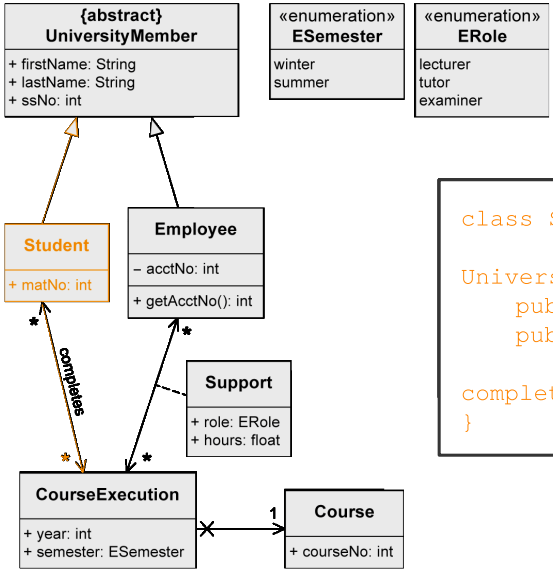





```
Enumeration ESemester {
    winter,
    summer
}
```

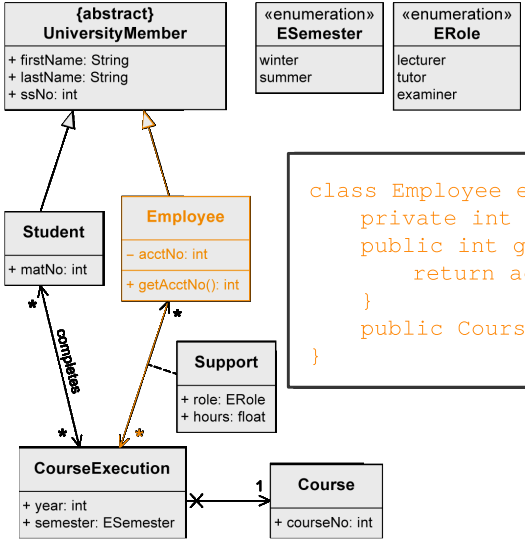
```
Enumeration ERole {
    lecturer,
    tutor,
    examiner
}
```

Code Generation



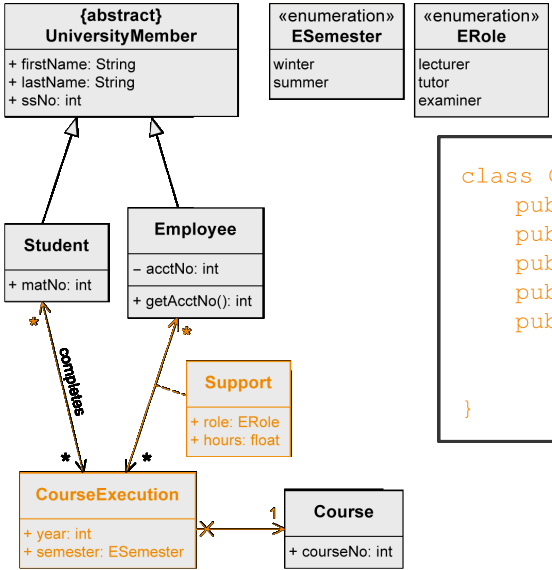

```
class Student extends

UniversityMember {
    public int matNo;
    public CourseExecution []

completedCourses;
}
```

Code Generation



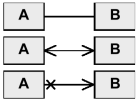

```
class Employee extends UniversityMember {
    private int acctNo;
    public int getAcctNo () {
        return acctNo;
    }
    public CourseExecution [] courseExecutions;
}
```

Code Generation

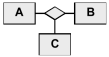
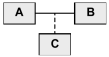
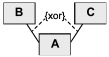



```
class CourseExecution {
    public int year;
    public ESemester semester;
    public Student [] student;
    public Course course;
    public Hashtable support;
        // Key: employee
        // Value: (role, hours)
}
```

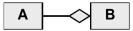
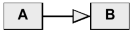
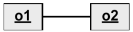
Notation Elements

Name	Notation	Description
Class		Description of the structure and behavior of a set of objects.
Abstract Class		Class that cannot be instantiated
Association		Relationship between classes

Notation Elements

Name	Notation	Description
n-ary Association		Relationship between n (here 3) classes
Association Class		More detailed description of an association
`xor` Relationship		An object of `C` is in a relationship with an object of `A` or with an object of `B` but not with both

Notation Elements

Name	Notation	Description
Shared Aggregation		Parts-whole relationship (`A` is part of `B`)
Composition		Existence-dependent parts-whole relationship (`A` is part of `B`)
Generalization		Inheritance relationship (`A` inherits from `B`)
Object		Instance of a class
Link		Relationship between objects

Trial 1: Home Heating System

Problem: Design a simple Home Heating System. This system includes at least a thermostat and a heater. The house is a combination of rooms and a thermostat controls the heater output for a room. A heater can have one thermostat. We know about a specific type of heater called an electric heater and a specific type of thermostat called the AubeTH101D.

Trial 2: Chess Game Backend

Problem: Describe, using a class diagram, the pieces, board, and game tree for a simple chess game. These components will be used to create a chess game used for play either standalone or in network mode.

Trial 3: Domain Model of Outside

Problem: Describe, using a class diagram, the relationships between the following concepts: Oak tree, Maple tree, Shrub, Branch, Lawn, Leaf, Grass.

Are there any questions?