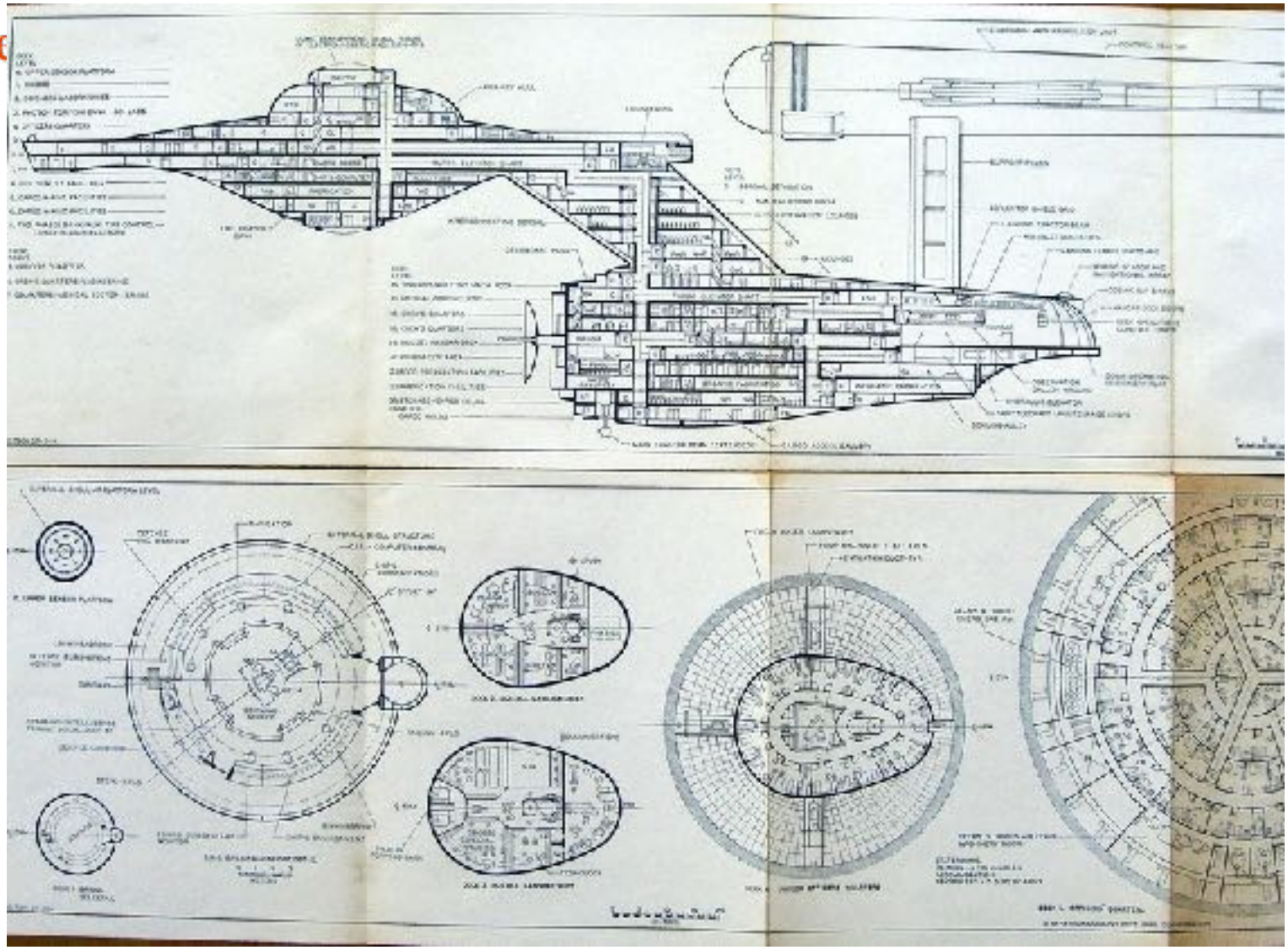
Diagram Notations - UML



Announcement

- Sprint 3 is done
- Sprint 2 grading by Thu
- Informal feedback on class (google form) today

Oregon State University

Some UML diagrams

- **Activity Diagram**
- **Class Diagram**
- Communication Diagram
- Component Diagram
- Composite Structure Diagram
- Deployment Diagram
- Interaction Overview Diagram
- Object Diagram
- Package Diagram
- **Sequence Diagram**
- **State Machine Diagram**
- Timing Diagram
- **Use Case Diagram**

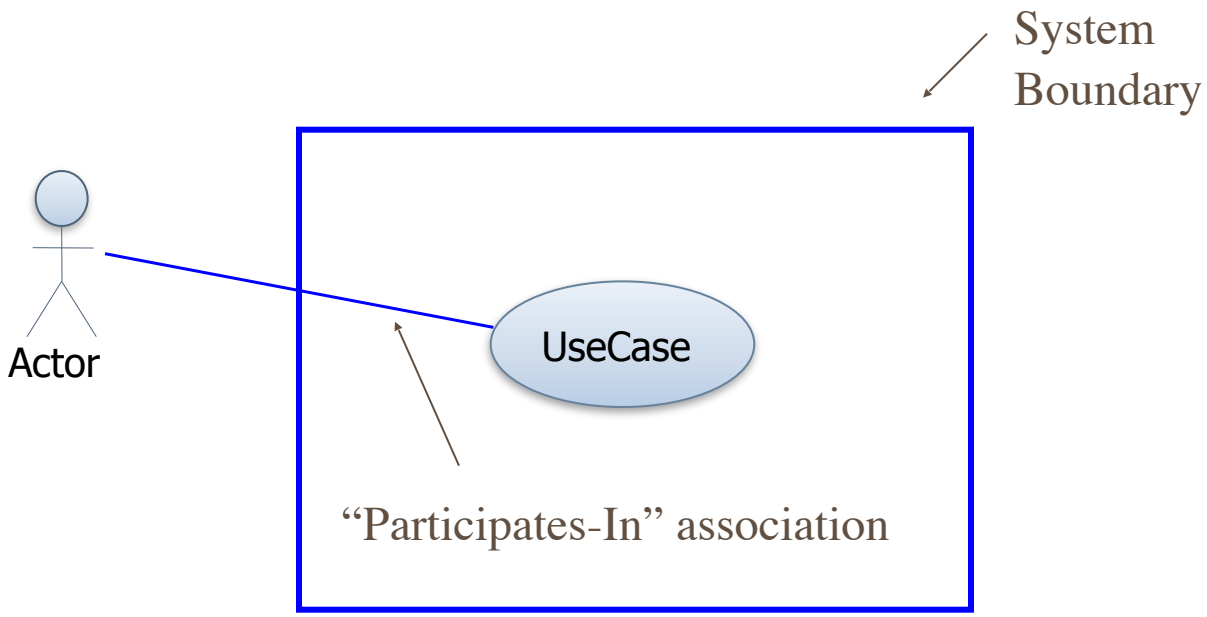
Oregon State University

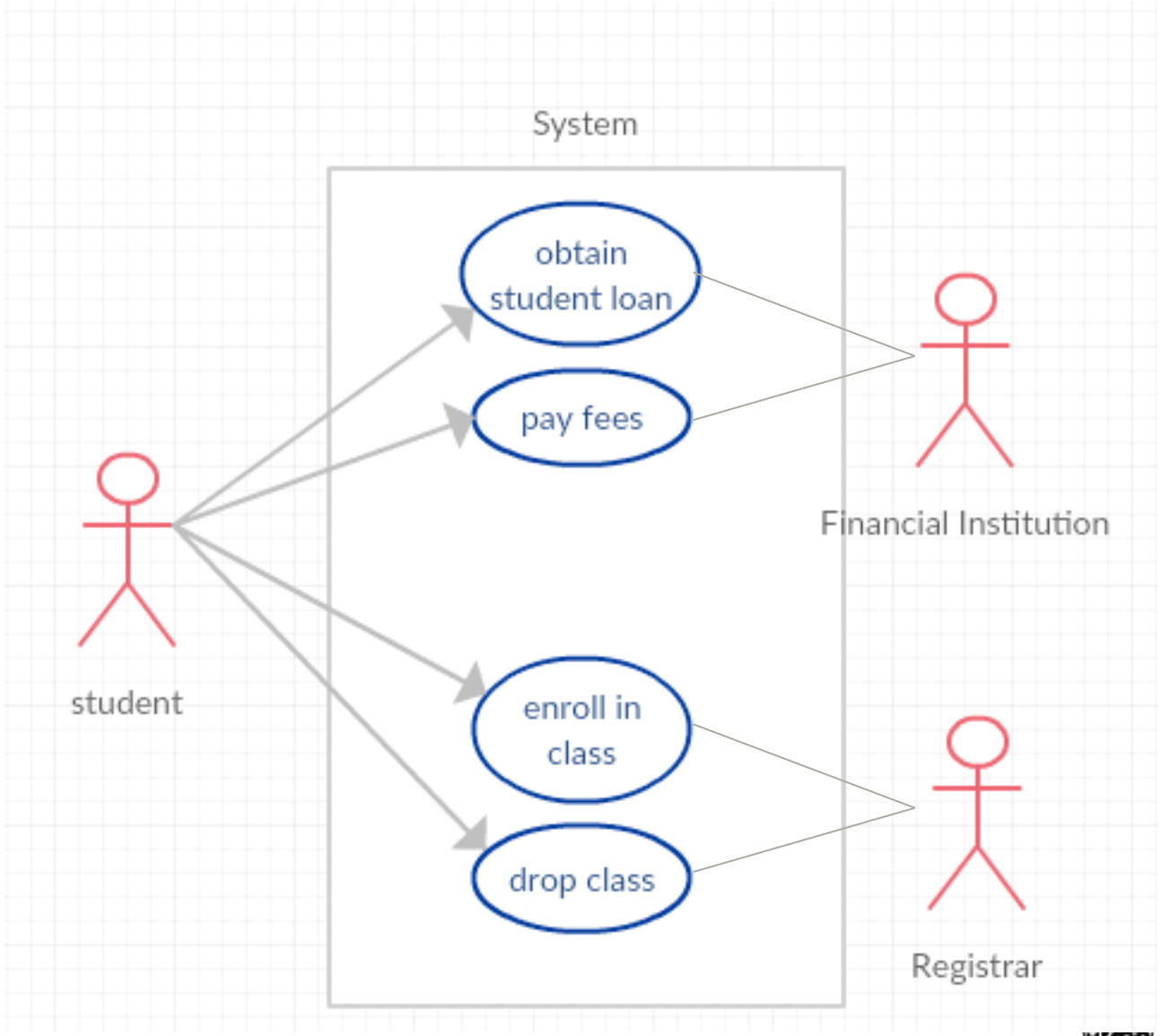
Use Case Diagrams

- Use Case Diagram at its simplest is a representation of a user's interaction with a system.
- Use Cases similar to User Stories, but more formal and more complex

Use Case Includes

- Summary of usage requirements
- From users point of view
- Basic Course of Events
- Alternative Paths
- Preconditions / Postconditions





Oregon State University

http://agilemodeling.com/artifacts/useCaseDiagram.htm

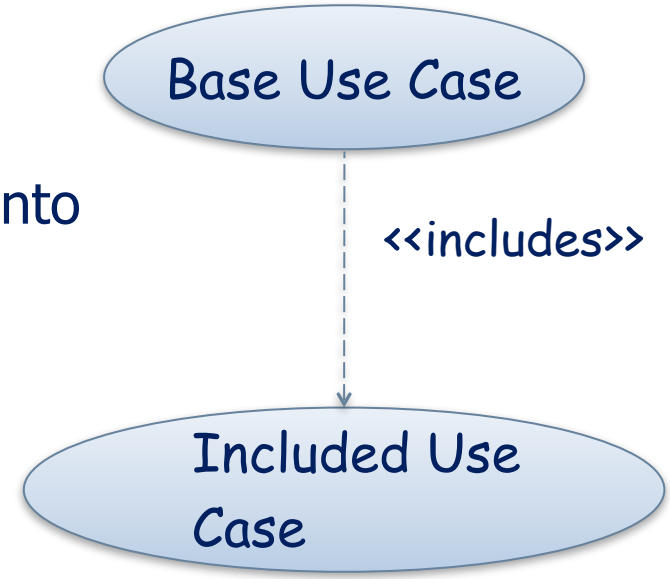
Includes vs. Generalization vs. Extends

- Use "includes" when you are repeating yourself in two or more separate use cases and you want to avoid repetition.

Oregon State University

Include

Factors use cases into additional ones



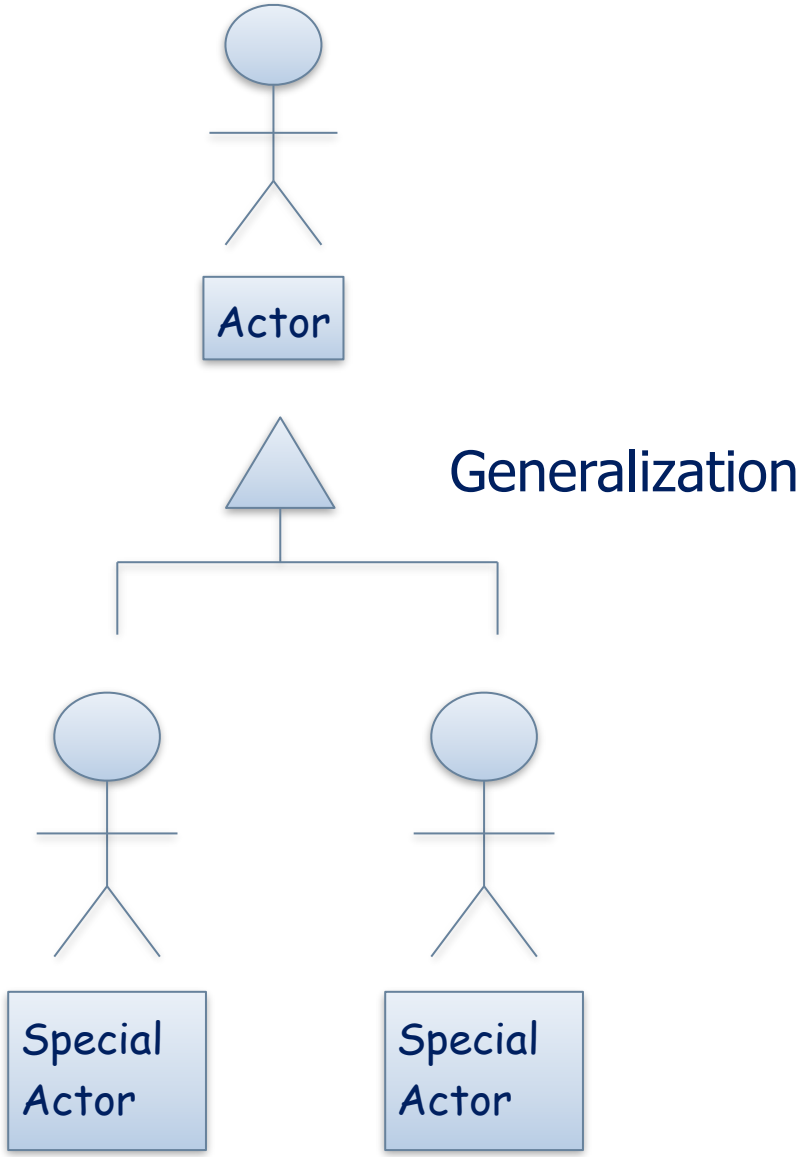
Includes vs. Generalization vs. Extends

- Use “includes” when you are repeating yourself in two or more separate use cases and you want to avoid repetition.

- **Use “generalization”, when you have a use case that is a special type of another use case**

Oregon State University

Generalization



Oregon State University

Includes vs. Generalization vs. Extends

- Use “includes” when you are repeating yourself in two or more separate use cases and you want to avoid repetition.

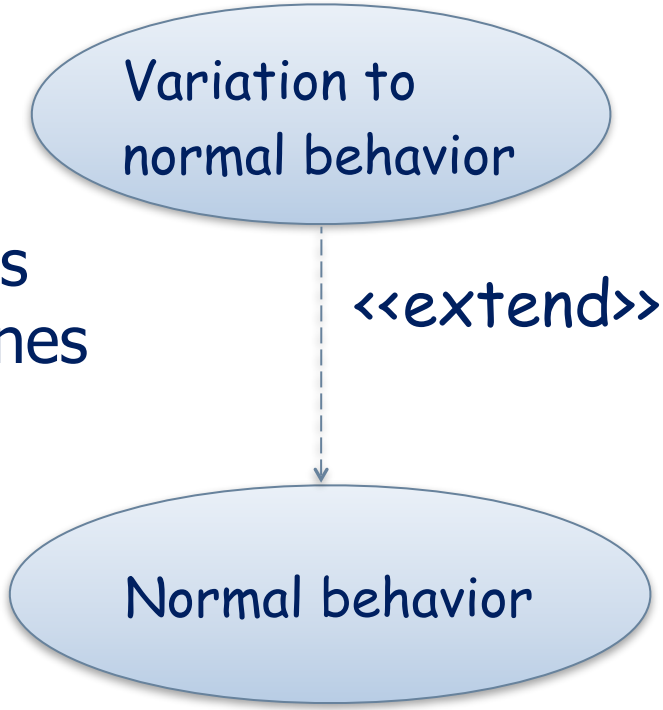
- Use “generalization”, when you have a use case that is a special type of another use case

- **Use “extends” when you are describing a variation on normal behavior - you can completely reuse another’s behavior, but is dependent either on runtime or system implementation decision**

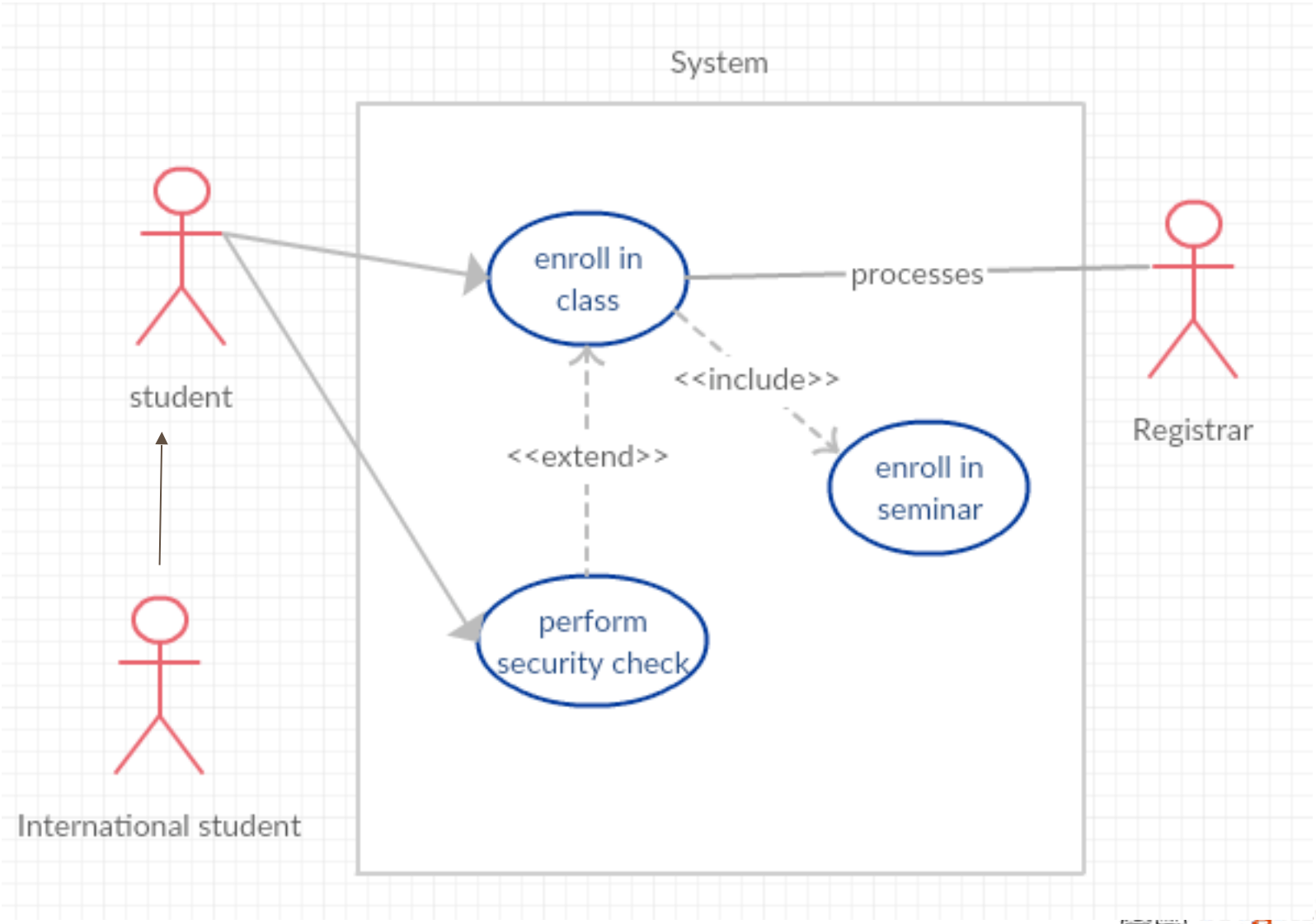
Oregon State University

Extends

Factors use cases into additional ones



Oregon State University



Exercise

- Get (buy/rent) the (SE) book from Amazon

Class Diagrams

But first, Object Terminology Review

- An object mirrors real world entity
- Examples:
  - Person, student, book, card, game, etc.

Object Terminology Review

- Objects Contain (class):
  - attributes (variables)
  - functionality (methods)
- Objects can have properties or be acted upon

Inheritance

- Allows one Class to automatically “assume” the attributes of another class
- Defines an “is a” relationship for classes

Building an Object Oriented Model

Our model should:

- represent entities
- show connections and interactions
- show enough detail to evaluate designs

Oregon State University

Classes

A class describes a group of objects with:

- similar properties (attributes)
- common behavior (operations)
- common relationships
- common meaning

Example class:

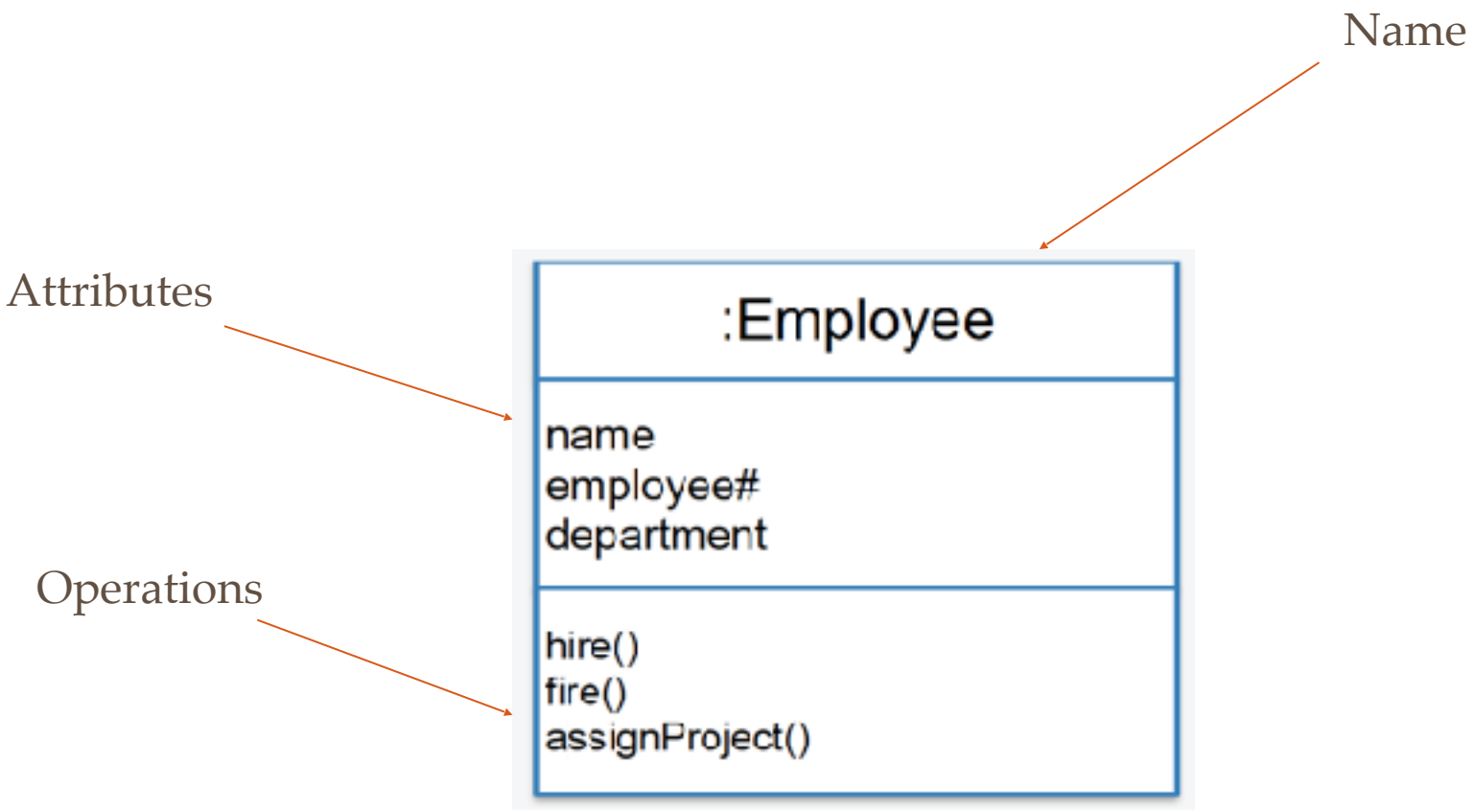
employee:

has a name,
employee#,
department

an employee is

hired,
fired;

an employee works in one or more projects



Oregon State University

UML Class Diagram parts

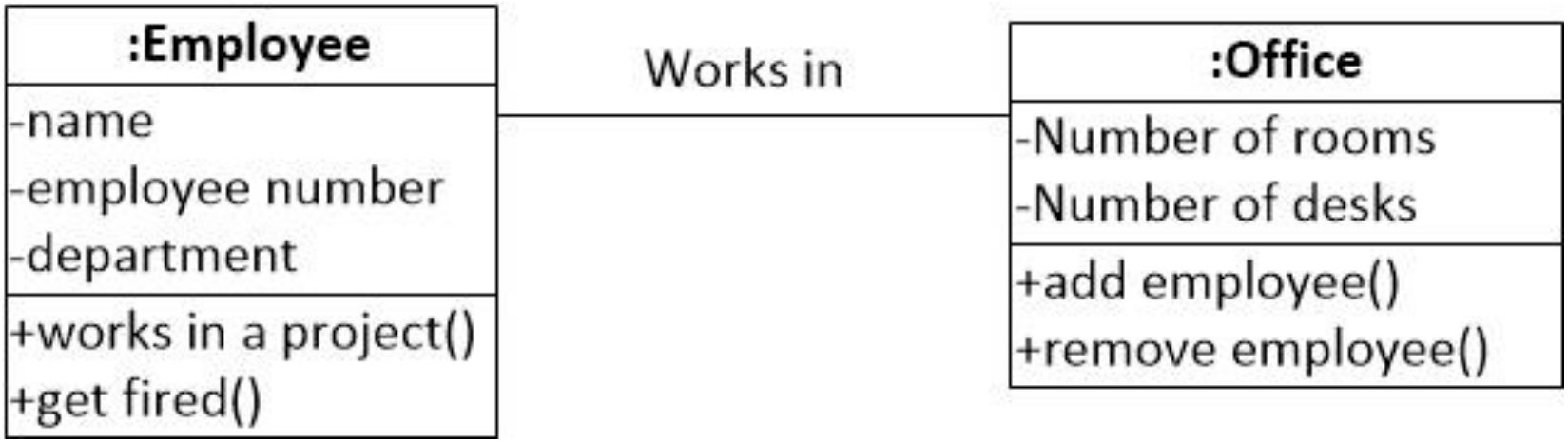
Objects do not exist in isolation

UML supports:

- Association
- Aggregation and Composition
- Generalization
- Dependency

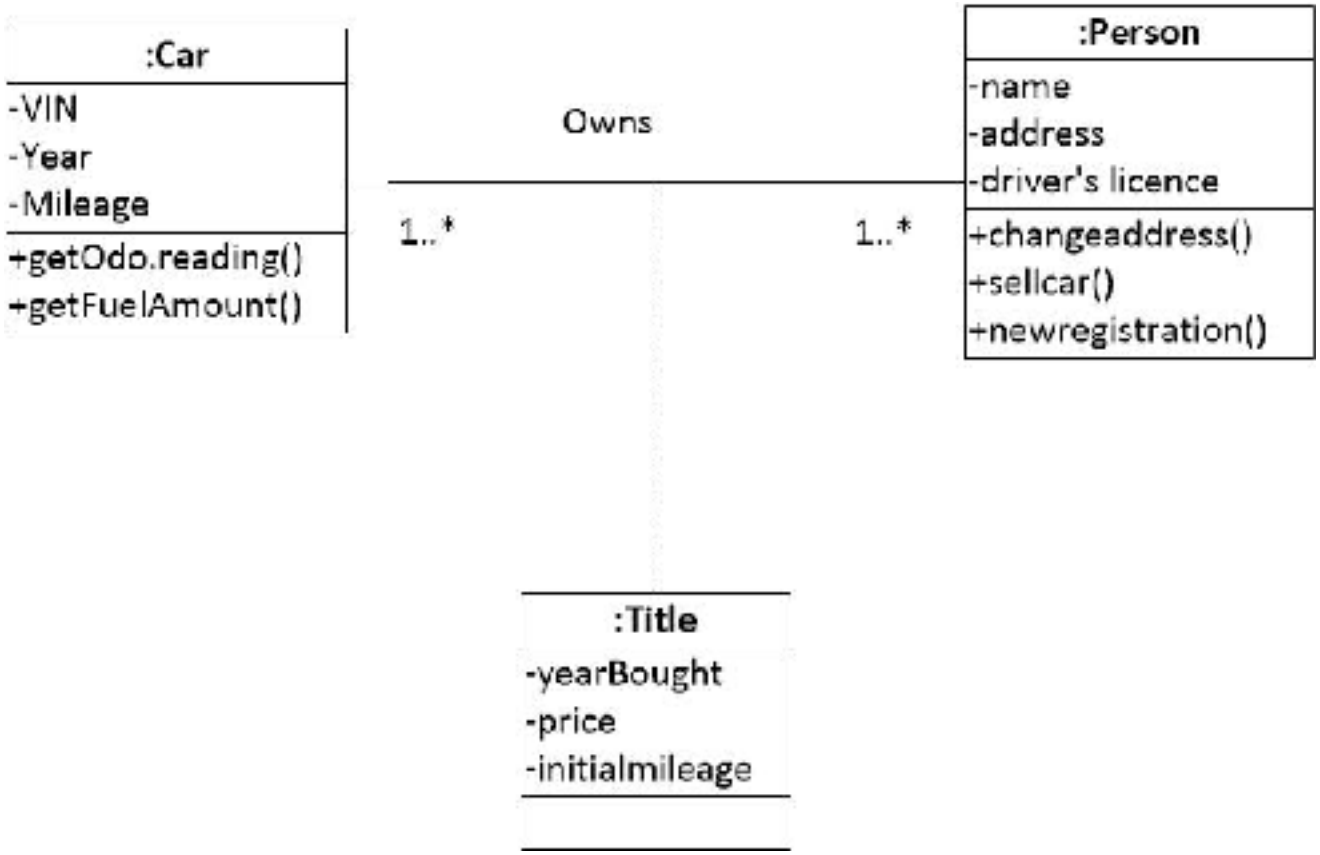
Class Associations

- Most generic kind of relationship
  - Instructor <teaches a> class
  - Kid <plays with a > friend
  - Employee <works in an> office



Oregon State University

Example of an association class



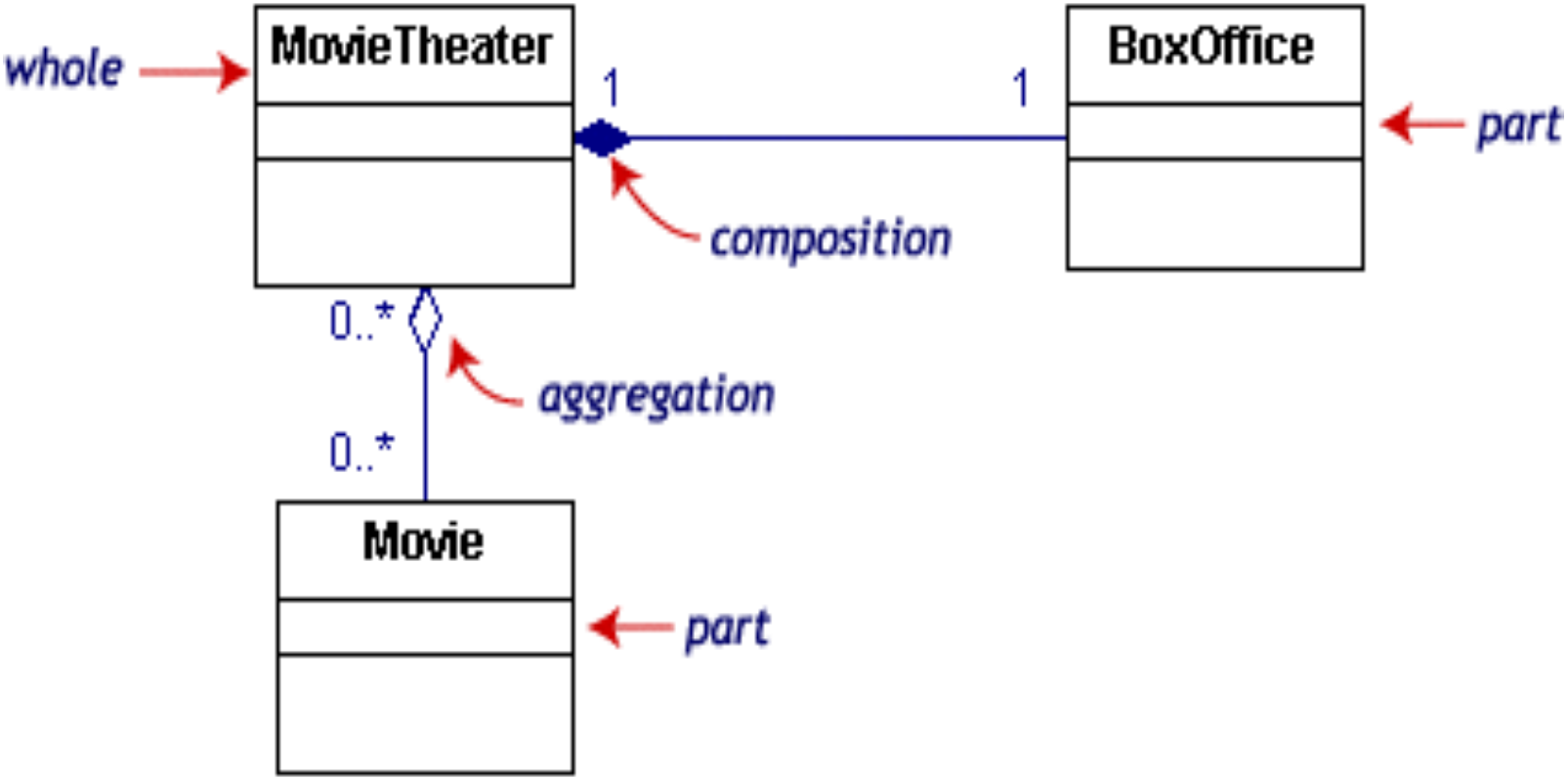
Oregon State University

Aggregation and Composition

- Aggregation - a more specific kind of relationship
  - “has-a-relationship”
  - “is a part of relationship”
  - part can exist independent of the whole
- bird <is-part-of-a> flock, airplane type <has-a> engine model

- Composition: more specific yet
  - consists-of relationship
  - contains relationship
  - part cannot exist independent of the whole
- house <consists-of a> room, university <contains a> department

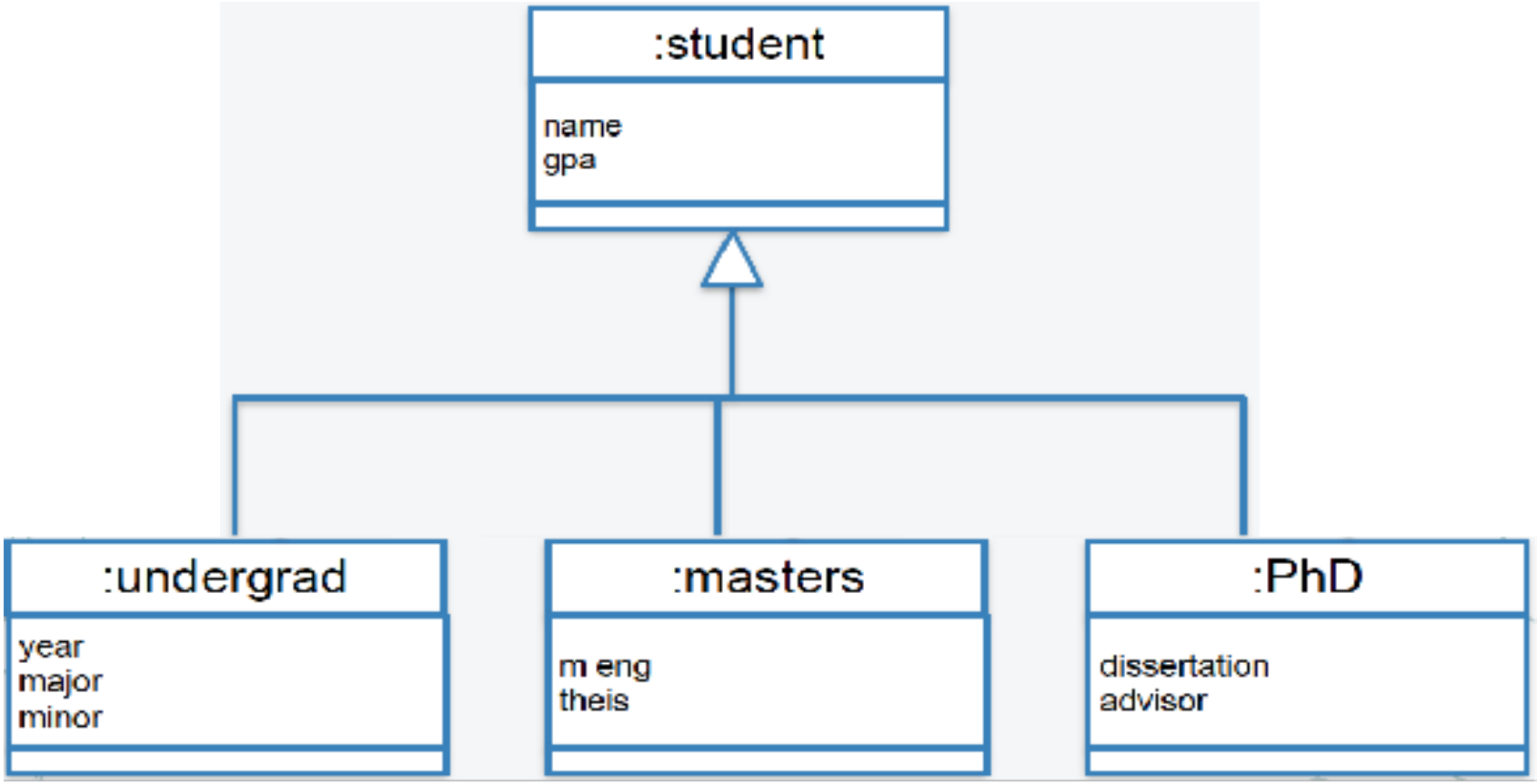
Aggregation and Composition example



Generalization

- <child> is more specific versions of the <parent>
- <child> inherits attributes, associations, & operations from the <parent>
- <child> can override an inherited aspect

Generalization example

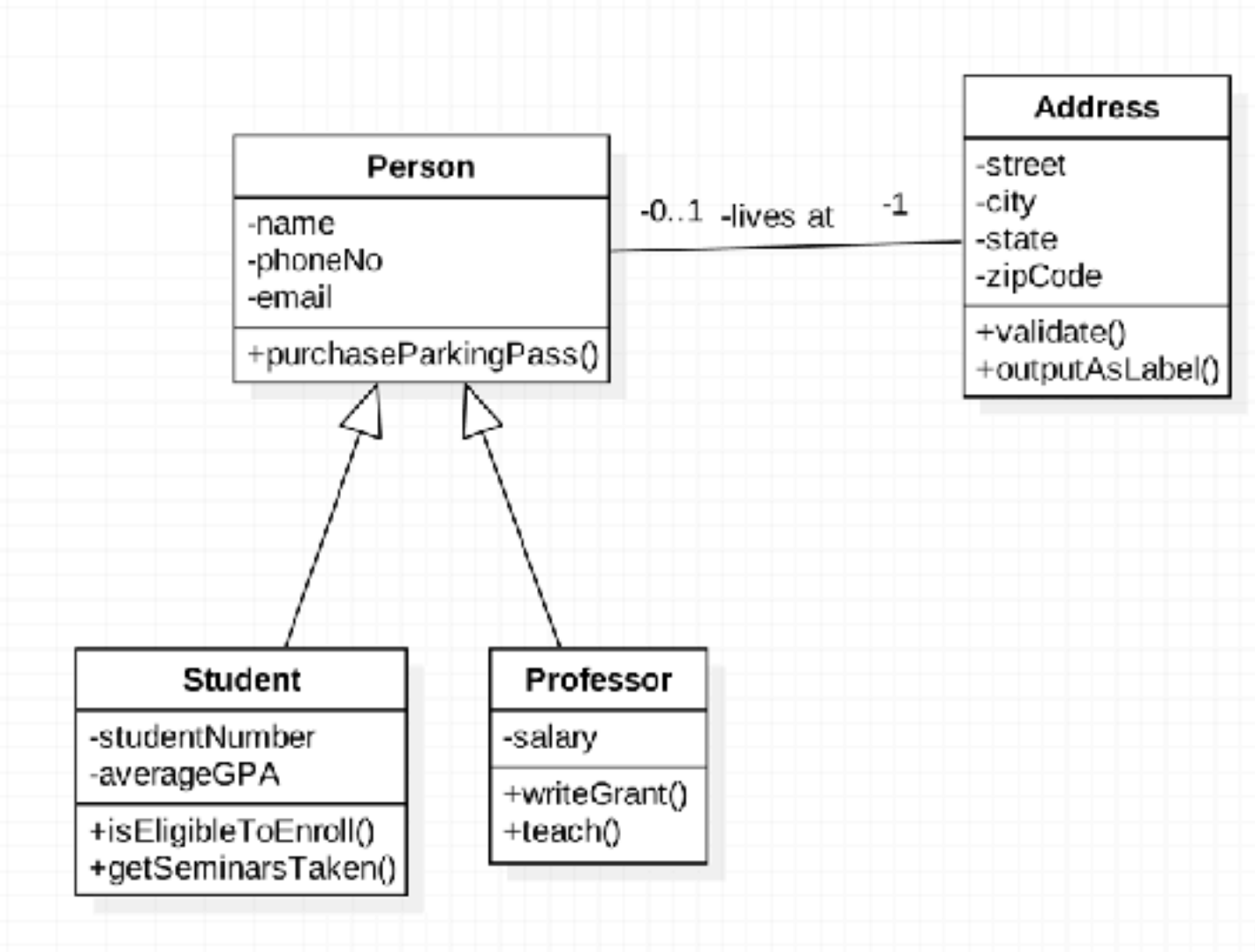


Example: University system

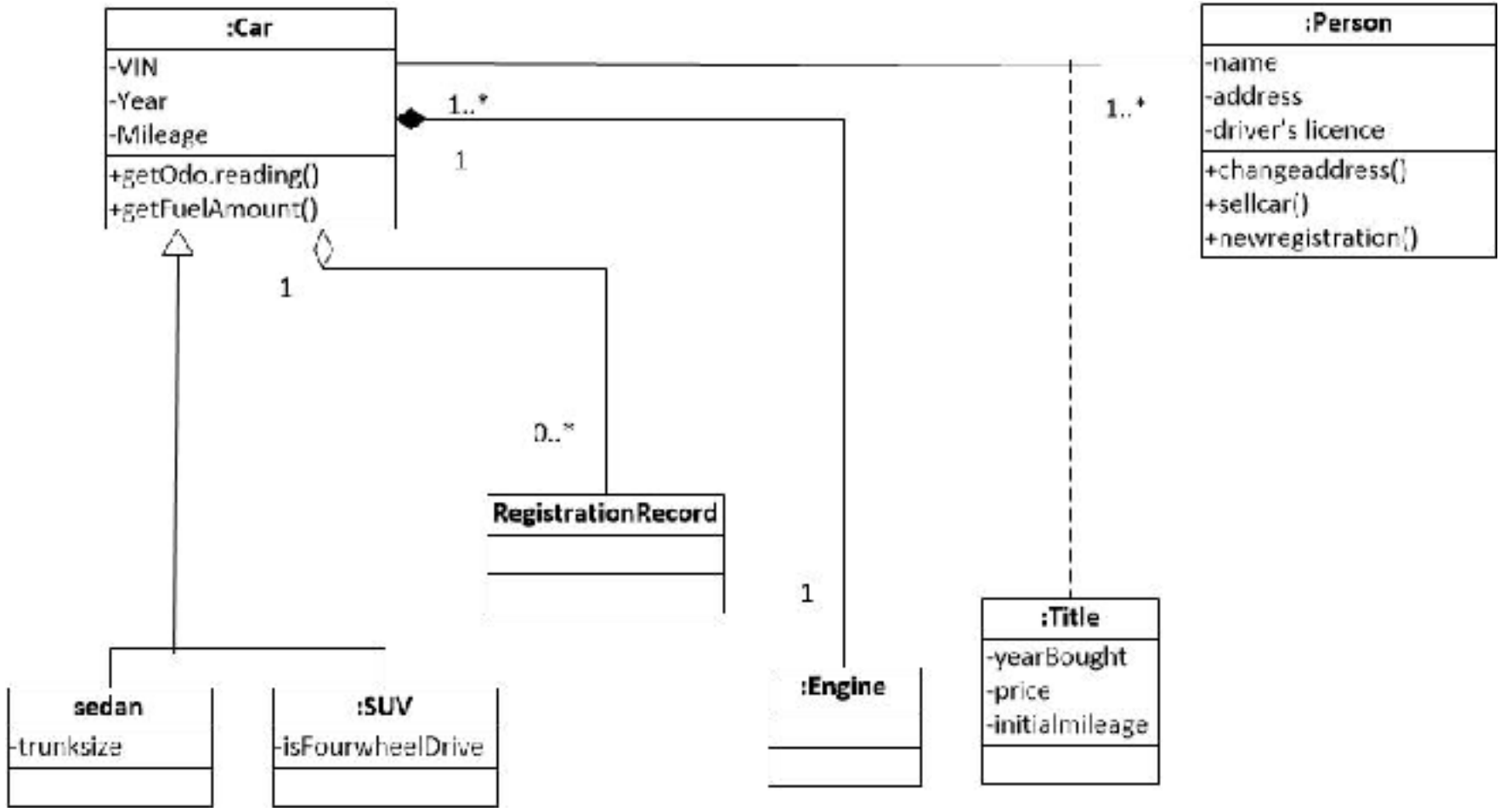
1. System records for each university employee
   - Details
   - (mailing) Address
2. Students
   - buy parking pass
   - enroll in classes
   - enroll in (health) seminars
- Faculty
   - buy parking pass
   - write grant funding
   - teach classes

Oregon State University

Example



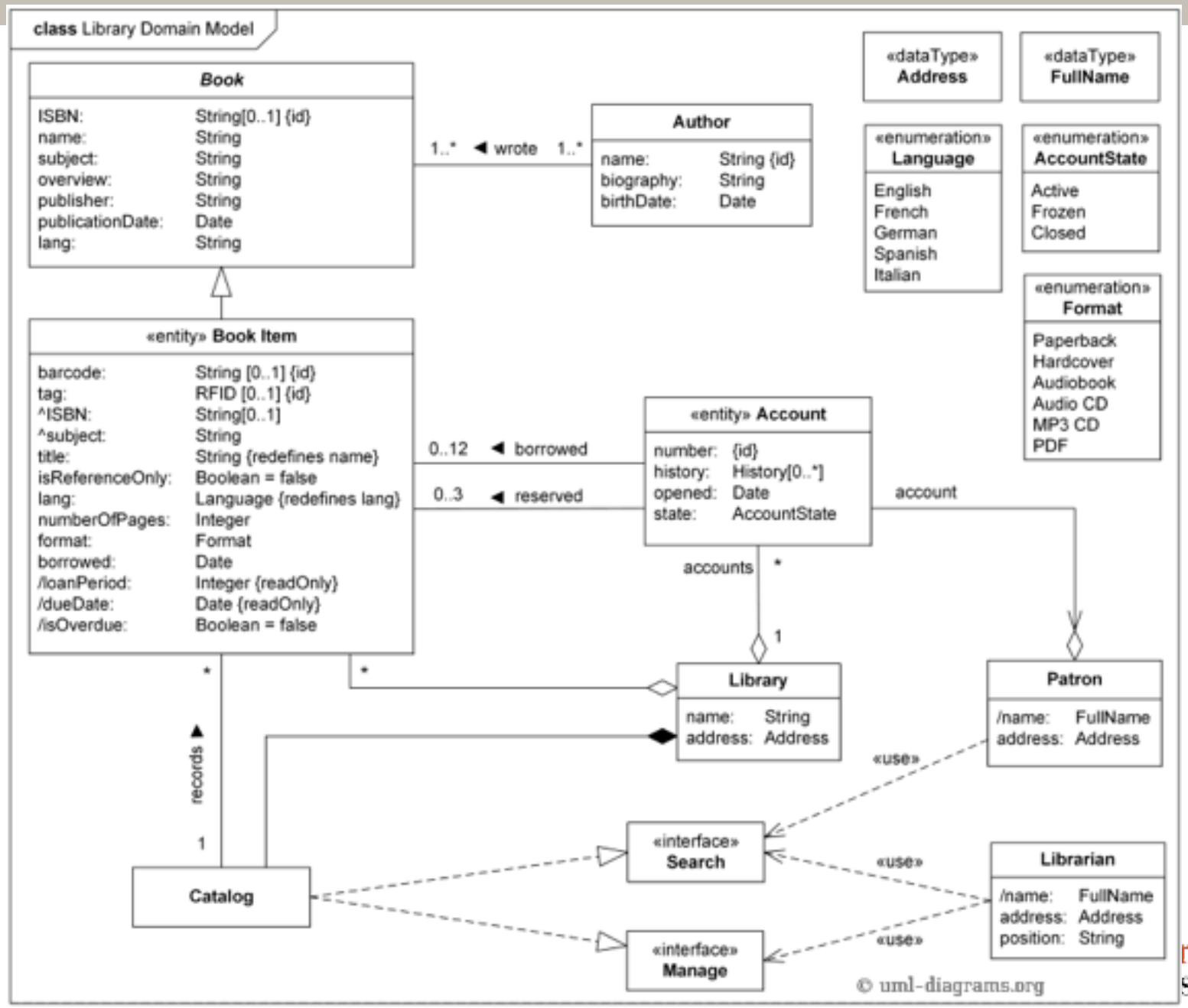
Oregon State University



Oregon State University

Exercise

- Book buying in Amazon system
  - Specific book versions may have differences (online, hard copy, soft copy)
  - Books are written by authors
  - Books have reviews
  - A user (or their account)



http://www.uml-diagrams.org/class-diagrams-overview.html


Oregon State
University

Informal Early Feedback

- http://tinyurl.com/cs361-ief

Oregon State University