Introduction to Design Principles

CS6.401 Software Engineering

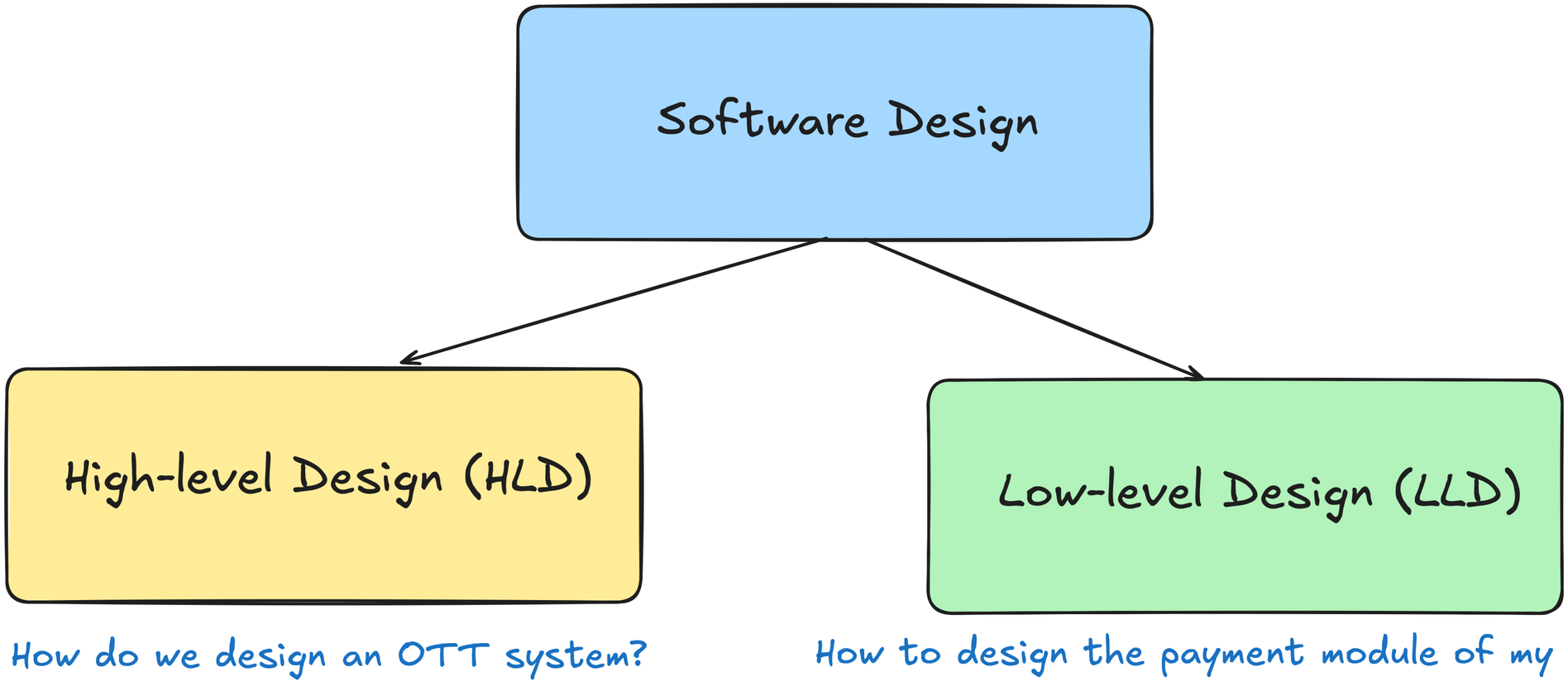
Karthik Vaidhyanathan
https://karthikvaidhyanathan.com

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

SERC
Software Engineering Research Centre

Software Design

The function of good software is to make the complex appear to be simple. - Grady Booch



Focus is on Architecture of Software

Focus is more at code level or much lower levels

Lets take a simple case into consideration

Course Management System Use case

- High level architecture is done now time for low-level
- Helps manage courses, students and teachers in a single environment
- **Entities:** Student, Course, Instructor, Enrolment
- **Functionalities:**
  - Students register for the course
  - Instructors create and manage course content
  - The system can generate a performance report

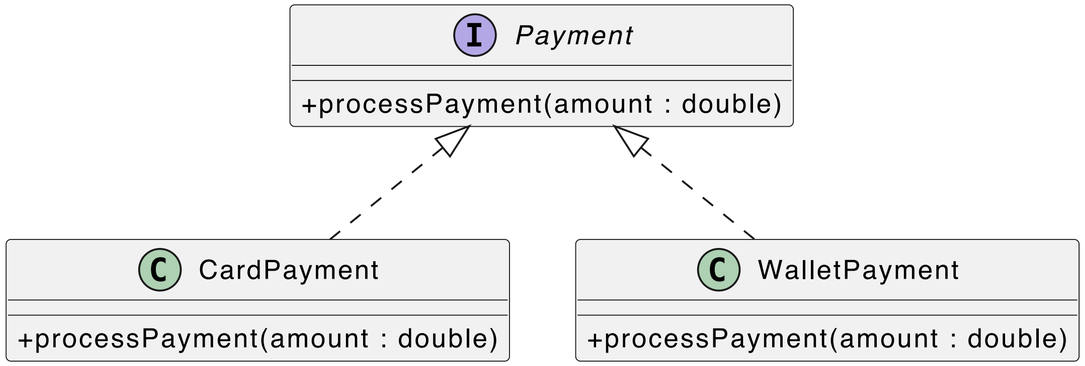
Key Design Principles

Lets revisit through some of the OOPs concepts

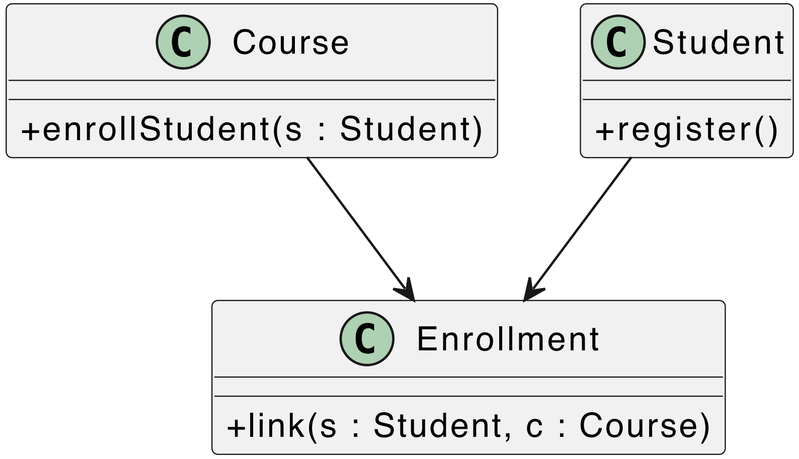
- **Abstraction:** Focus on what the object does and not how does it do!
- **Encapsulation:** Keep data and behaviour in one place, protect internal state
- **Modularization:** Divide system into independent modules (classes)
- **Hierarchy:** Structure into ordered layer of abstractions

What it means to apply the above design principles to our case study system?

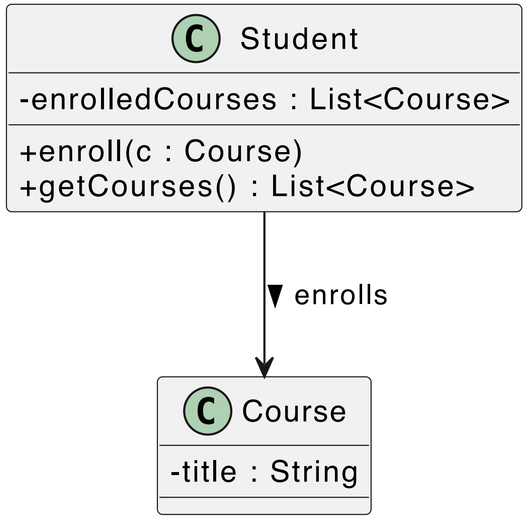
Design Principles in Action



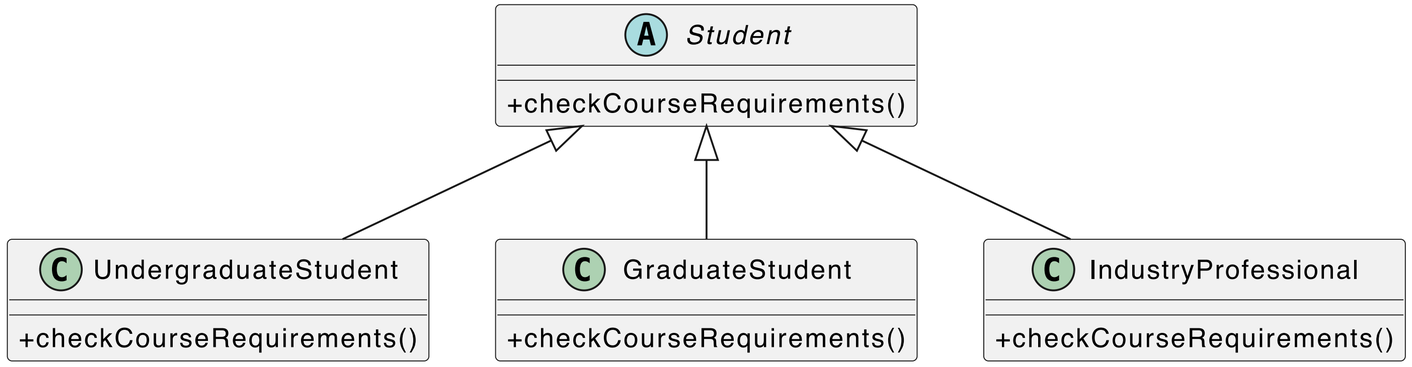
Abstraction



Modularization



Encapsulation



Hierarchy

Designing that too OO systems is not Straightforward

How to identify objects

How to group objects to classes

Interfaces have to defined

Hierarchies too!

Relationships among Objects!!

Some thoughts on the Process of Design

- Designs should be reusable, flexible and understandable
- Very difficult to get it right the first time – Not hard though!!
- Experience people also take multiple iterations
- Novice find it even more difficult to get their head around

Experts are able to make good designs… How?

Things improve with Practice

Experts tend to reuse solution that have worked in the past!

The way objects are identified, relationships are established becomes a recurring activity

When something has been tried and worked well, why not use it again!!

They start seeing recurring **patterns** over time

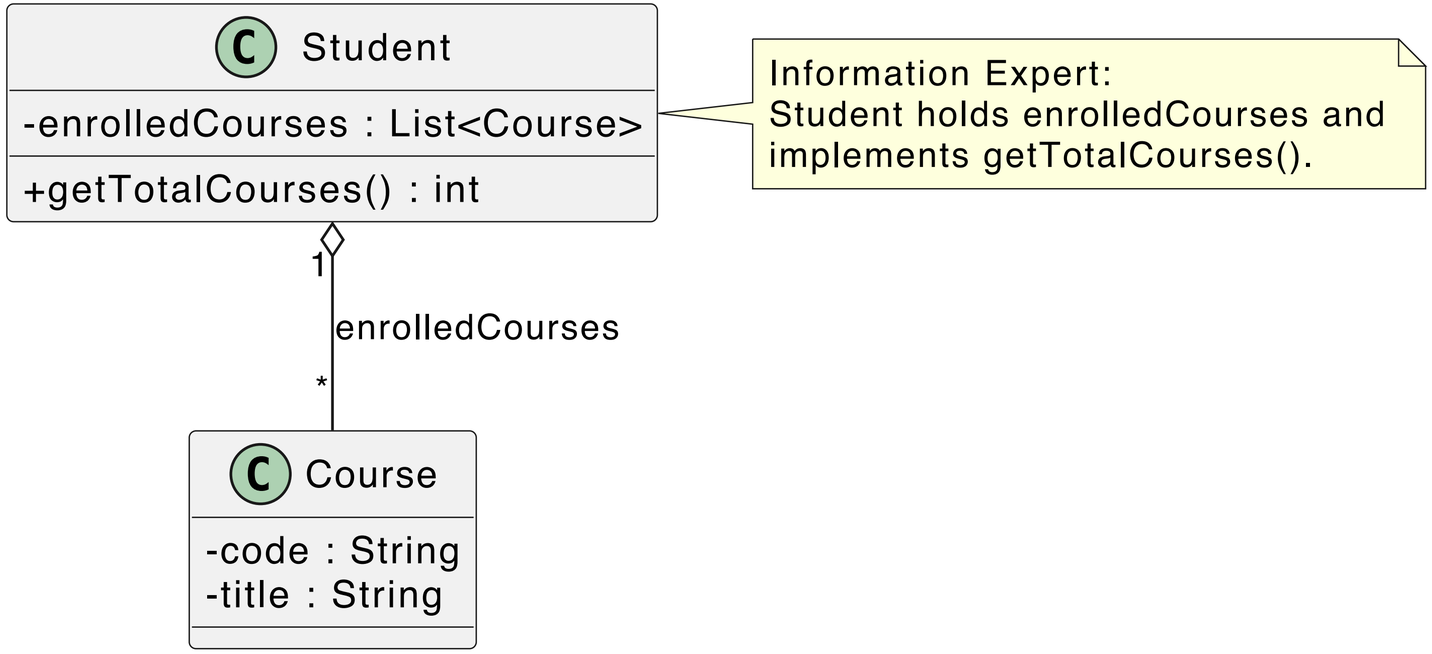
What if this experience could be recorded for reuse?

This is where principles like **GRASP or SOLID** can help

Information Expert Principle

Calculate how many courses a student is enrolled in

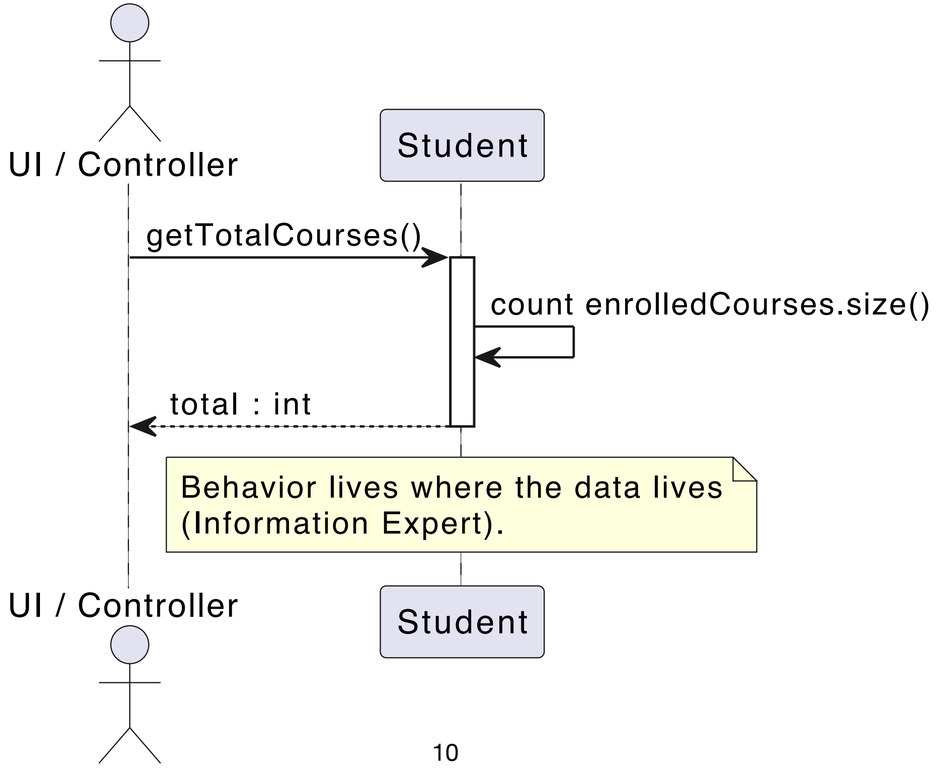
- The **Student** class holds the list of enrolled courses -> information expert
- No need to create a new dedicated class to do this action! - anti-pattern



Information Expert Principle

Calculate how many courses a student is enrolled in

- The **Student** class holds the list of enrolled courses -> information expert
- No need to create a new dedicated class to do this action! - anti-pattern



General Responsibility Assignment Software Patterns or Principles

Information Expert

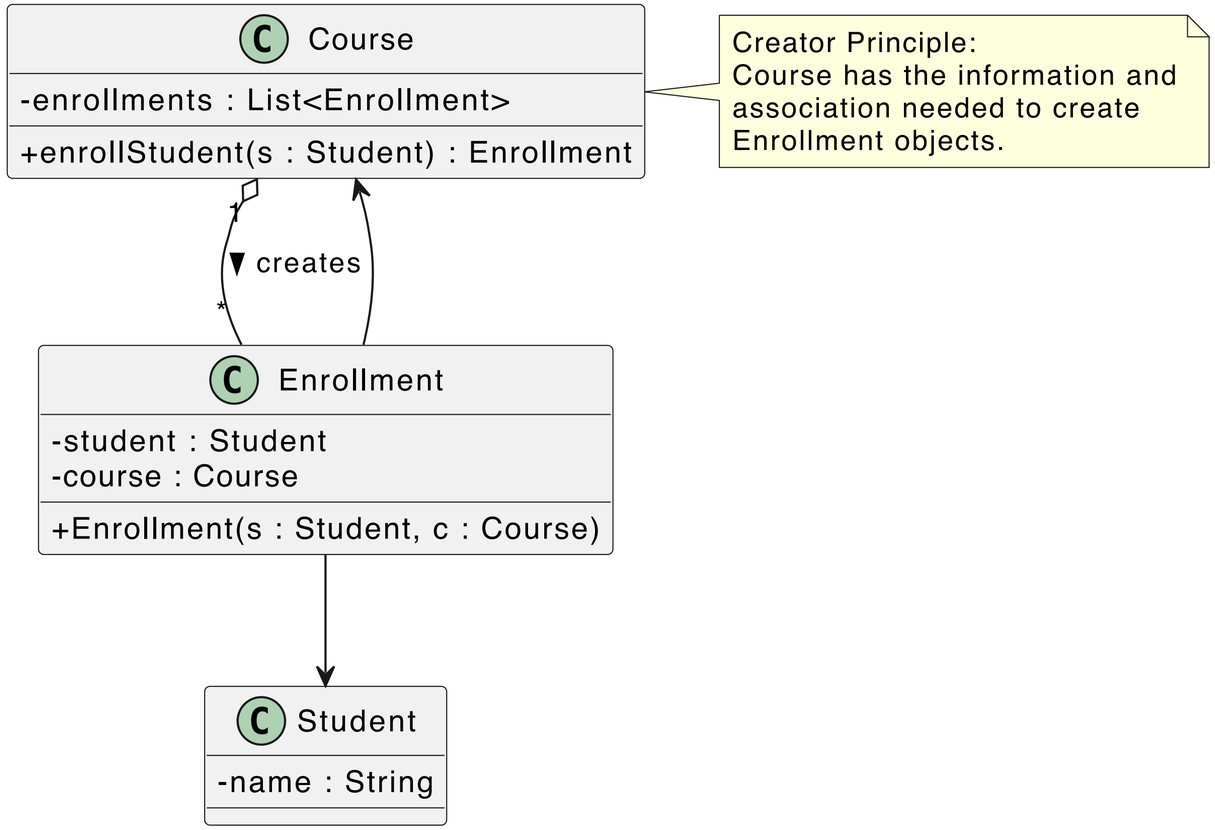
Assign a responsibility to the class that has the necessary information to fulfil it.

- The one who has data should also have the operations to perform on the data
- Check where the information naturally resides in the domain model
- The class that has the data required to perform a task should also perform the behaviour related to it
- Helps achieve good encapsulation - data and behaviour reside together
- Simpler and maintainable design

Creator Principle

Student needs to Enroll for a course

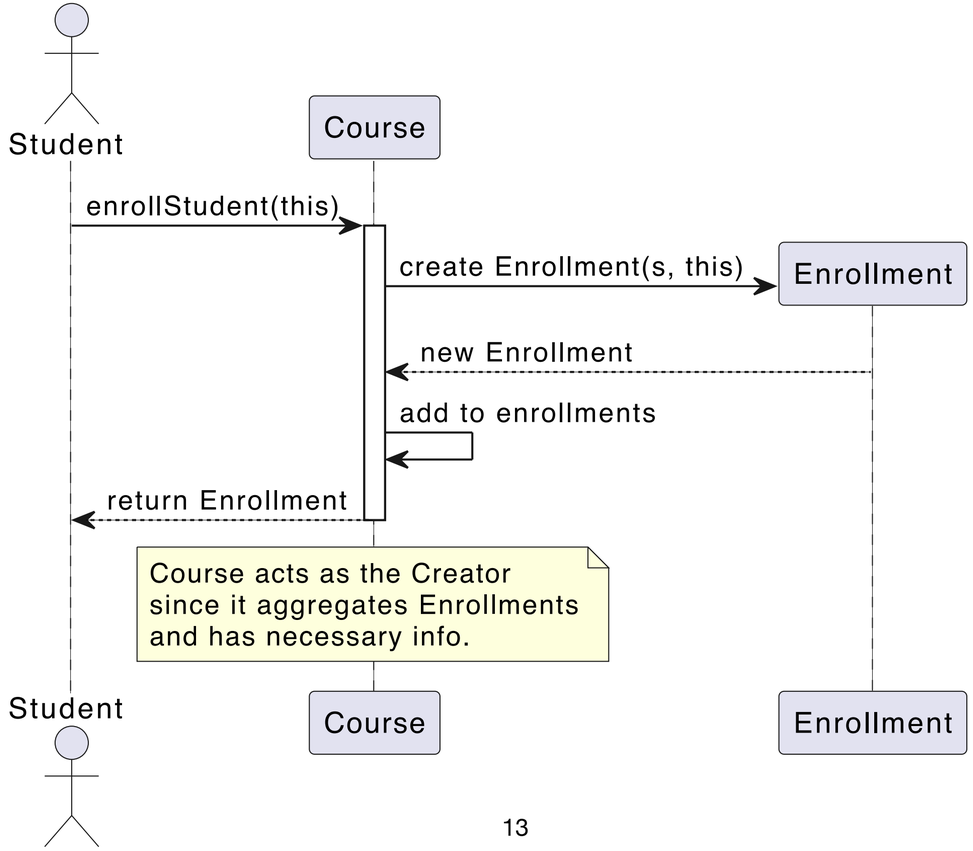
- The enrolment object needs to be created to link Student and Course
- The course has all the details - Let that class create the Enrollment Object



Creator Principle

Student needs to Enroll for a course

- The enrolment object needs to be created to link Student and Course
- The course has all the details - Let that class create the Enrollment Object



GRASP: Creator

A class should be responsible for creating instances of another class if it has the information needed to initialize the object or has a close relationship with it

Defines guidelines for which class should be in charge of creating objects of other type

Enhances Cohesion: Same responsibilities grouped together

E.g. Class B should be in charge of creating objects of A if:

- B contains or compositely aggregates A
- B closely uses A
- B has inputs to construct A
- B records A

Thank you

Email: karthik.vaidhyanathan@iiit.ac.in
X: @karthi_ishere