COEN 6312

Model Driven Software Engineering

Project Deliverable 3

Class diagram & Constraints

Course Instructor
Dr. Wahab Hamou-Lhadj

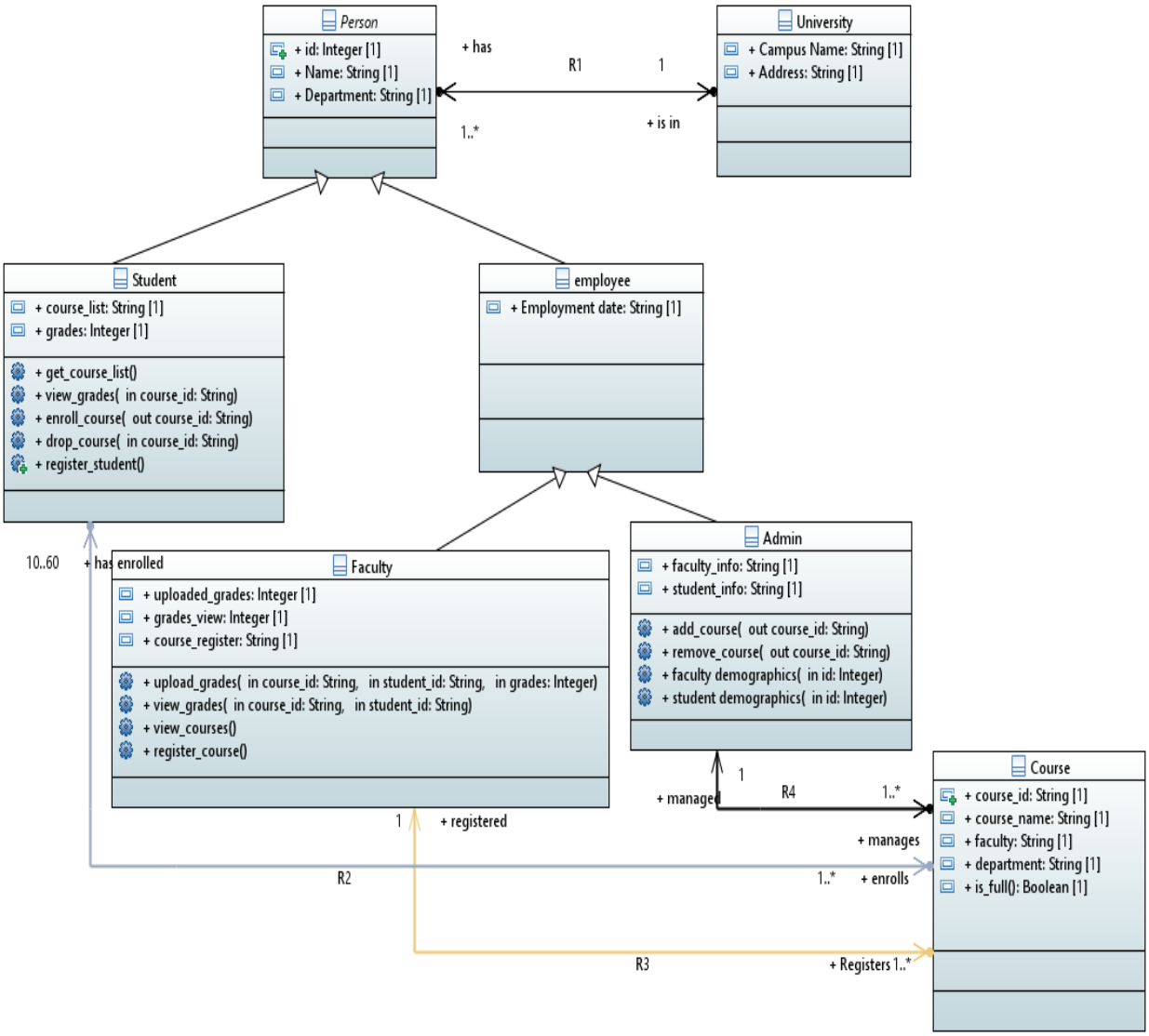
Team: Techno_reg
Project Title: Course Registration System

Submitted by

Farhan Saeed	**40039670**
Dinesh Pattapu	**40070809**
Nikitha Papani	**40070806**
Mohammed Hakeemuddin	**40059007**
Sai Abishek Thorikonda	**40071051**

Table of Contents

Class Diagram



Class diagram representation (Papyrus)

Class Diagram Description

There are seven key classes namely: Person class, Student class, Employee class, Faculty class, Admin class, Course class, University class.

1. University Class

University Class consists of Campus name and address of the university.

Attributes of University Class

Attribute Name	Attribute Description	Attribute Type
Campus Name	Name of campus	String
Address	Campus address	String

Association

University class and Person class has a bi-directional association, related as person is in 1 university and university has 1 or many persons.

2. Person Class

A person can be either a student or an employee consisting of faculty and admin.

Attributes of Person Class

Attribute Name	Attribute Description	Attribute Type
Id	Represents unique ID of a person	Int
Name	Represents student, faculty or admin name	String
Department	Represents department to which a person belongs	String

Associations

The Person Class and University Class has bi-directional association, this means that both classes are aware of each other and their relationship, Also Person class acts as a parent class for student and employee.

3. Student Class

Students will be able to view the Course List offered, can register to the course of their choice. Students can also drop the course if they intend to. Lastly students will be able to view the grades posted by the Faculty.

Attributes of Student Class

Attribute Name	Attribute Description	Attribute Type
course_list	Contains list of available courses	String
grades	Contain student grades	Integar

Operations of Student Class

Operation Type	Description
get_course_list()	Students will be able to view the courses offered by the university.
view_grades(course_id:c)	Students will be able to view past grades.
enroll_course(course_id:c)	Students will be able to enroll to the course of their choice.
drop_course(course_id:c)	Students will be able to drop the course if intended.
register_student()	In case of new student, they will be able to get registered to have their id and password.

Associations

Students class inherits few functionalities from the person class namely: Id, Name and the Department they belong to.

4. Employee Class

Employee class gives the information about date of joining.

Attributes in Employee Class

Attribute Name	Attribute Description	Attribute Type
Employment date	Employment date of employee	String

Associations

Employee class inherits functionalities from Person class namely: Id, Name, Department and acts as a parent class for faculty and admin.

5. Faculty Class

Faculty can select which courses they would like to teach in the semester of their choice and can also upload and view results of examinations they conducted.

Attributes of Faculty Class

Attribute Name	Attribute Description	Attribute Type
uploaded_grades	Contains uploaded grades	Integar
grades_view	To view past grades	Integar
course_register	To register for a course to teach	String

Operations in Faculty Class

Operation Type	Description
upload_grades()	Faculty will be able to upload the results of the past exams.
view_grades()	Faculty will be able to view the uploaded grades.
view_courses()	Faculty will be able to view course list.
register_course()	Faculty will be able to register to the course of his choice they would like to teach.

Associations

Faculty class inherits few functionalities from the Employee class namely: Employment date i.e. Date of joining.

6. Admin Class

Admin has the facility to add or remove course depending on the number of students. Admin will be able to view all the students and faculty demographics.

Attributes of Faculty Class

Attribute Name	Attribute Description	Attribute Type
faculty_info	Contains faculty information	String
student_info	Contains student information	String

Operations of Admin Class

Operation Type	Description
add_course(course_id:c)	Admin will be able to add the course depending on the availability of the faculty and number of students.
remove_course(course_id:c)	Admin will be able to drop the course if there are no sufficient students registered for the course.
faculty_demographics(id:c)	Admin will be able to view faculty demographics.
student_demographics(id:c)	Admin will be able to view students demographics.

Associations

Admin class inherits all the functionalities of Employee class. i.e. Employment date etc.

7. Course Class

The course class consist of list of courses with unique course code, course name, faculty and department. Successful login is required to access this class, if a number of students in a particular course is reached to 60 then no new students can be registered to the course.

Attributes of Course Class

Attribute Name	Attribute Description	Attribute Type
course_id	Course identification number	String
course_name	Course name	String
faculty	Faculty name	String
department	Department name	String
Is_full	Course has reached its full student capacity	Boolean

Associations

Students, Faculty, Admin are associated to Course class by R2, R3 and R4 respectively.

1) Student can enroll for a course if is_full is set to false.
2) Admin will be able to add or remove courses.
3) Faculty will be able to view course list and can register themselves to the course of their choice.

OCL Expressions

Target	Each person should have unique ID
Expression	Context Person Inv: allInstances() →forAll(p1,p2 |Person p1<>p2 implies p1.id<>p2.id)

Target	Each person should have a valid name
Expression	Context Person Inv: self.Name→length() > 0

Target	Student be able to view grades
Expression	Context Student::view_grades(course_id:c):void Pre: Self.R2 → exists(c) Post: return = self.grades

Target	Student should not be enrolled in same course twice
Expression	Context Student::enroll_course(course_id:c):void Pre: c.is_full() = False and Self.R2 → excludes(c) Post: c.is_full() = True and Self.R2 → includes(c)

Target	Student should be able to drop courses in which he is enrolled
Expression	Context Student::drop_course(course_id:c):void Pre: Self.R2 → Includes(c) Post: Self.R2 → Excludes (c)

Target	New students should be able to register into the system
Expression	Context Student::get_register():void Pre: Self.R2 → Includes(c) Post: Self.R2 → Excludes (c)

Target	Student should be able to get course list
Expression	Context Student::get_course_list():void Pre: Self.R2 → notEmpty() Post: return = includesall(self.course_list)

Target	Faculty should be able to upload grades
Expression	Context Faculty::upload_grades(course_id:c, student_id:s, grades:g):void Pre: Self.R3 → exists(c) AND Self.R3.R2 → exists(s) Post: uploaded_grades = grades

Target	Faculty should be able to view grades
Expression	Context Faculty::view_grades(course_id:c, student_id:s):void Pre: Self.R3 → exists(c) AND Self.R3.R2 → exists(s) Post: return = grades_view

Target	Faculty should be able to register course
Expression	Context Faculty::register_course(course_id:c):void Pre: Self.R3 → exists(c) Post: Self.course_register → includes(c)

Target	Admin should be able to add new courses
Expression	Context Admin::add_course(course_id:c):void Pre: Self.R4 → Excludes(c) Post: Self.R4 → Includes(c)

Target	If a course has 10 students or less admin should be able to remove it
Expression	Context Admin::remove course(course:c):void Pre: Self.R4 → Includes(c) AND self.R4.R2→size()<=10 Post: Self.R4 → Excludes (c)

Target	Admin should be able to see faculty demographics
Expression	Context Admin::faculty_demographics(id:c):void Pre: Self.R4.R3 → exists(c) Post: return = Self.faculty_info

Target	Course name must be unique
Expression	Context Course Inv: Self.R2 → forAll(c1,c2 | Course c1<>c2 implies c1.course_name <> c2.course_name)

Target	If a course has 60 students isfull must be true
Expression	Context Course::is_full():void Pre: Self.R2 → size()=60 Post: is_full='True'