SENG 300 Winter 2020

UCID: 30063099

Tutorial Number: 4

Assignment 2

By
Haohu Shen

February 24, 2020

Class Diagrams

Assumption 1: Each person who has the permission to login the system should have a unique ID, including the department head. Every user must use its ID to log in.

Assumption 2: Each person should have its name shown on its account.

Assumption 3: Though the web-base student information system is developed by the computer science department, it should serve all members of Emerging university. Since a campus is consist of several departments, we assume that each department should have its name and its department head.

Assumption 4: Each instance of ‘Course’ class should store its ‘Department’ instance as a member in order to avoid the user cannot differentiate the courses with the same name but come from different departments.

Assumption 5: Since a course plan is defined weekly, it should have a member indicates the number of week since the start of a semester.

Assumption 6: Since each weekly course plan may contain different kinds of materials, we use a list to store all ‘Material’ instances such that ‘Material’ is the base class of lecture notes, helping materials and assignments.

Assumption 7: Each section/session should have its unique ID and the course it contains.

Assumption 8: I assume that the assignment mark is an integer-type data.

Assumption 9: From my understanding, since a session is a concept of period, it should contain a start-date and an end-date, also all sections that happen during this period of time.

Assumption 10: When a student registers a course, he/she actually registers a section of a session that contains a course he selects. For example, for a UofC student, when he registers SENG300, he actually registers a section which contains the course SENG300 with its lecture number in a specific academic term(session). A student can only register a course in one section if there are many sections with the same course(with different lecture numbers). On the other hand, a student can register multiple sections as long as they do not contain the same course.

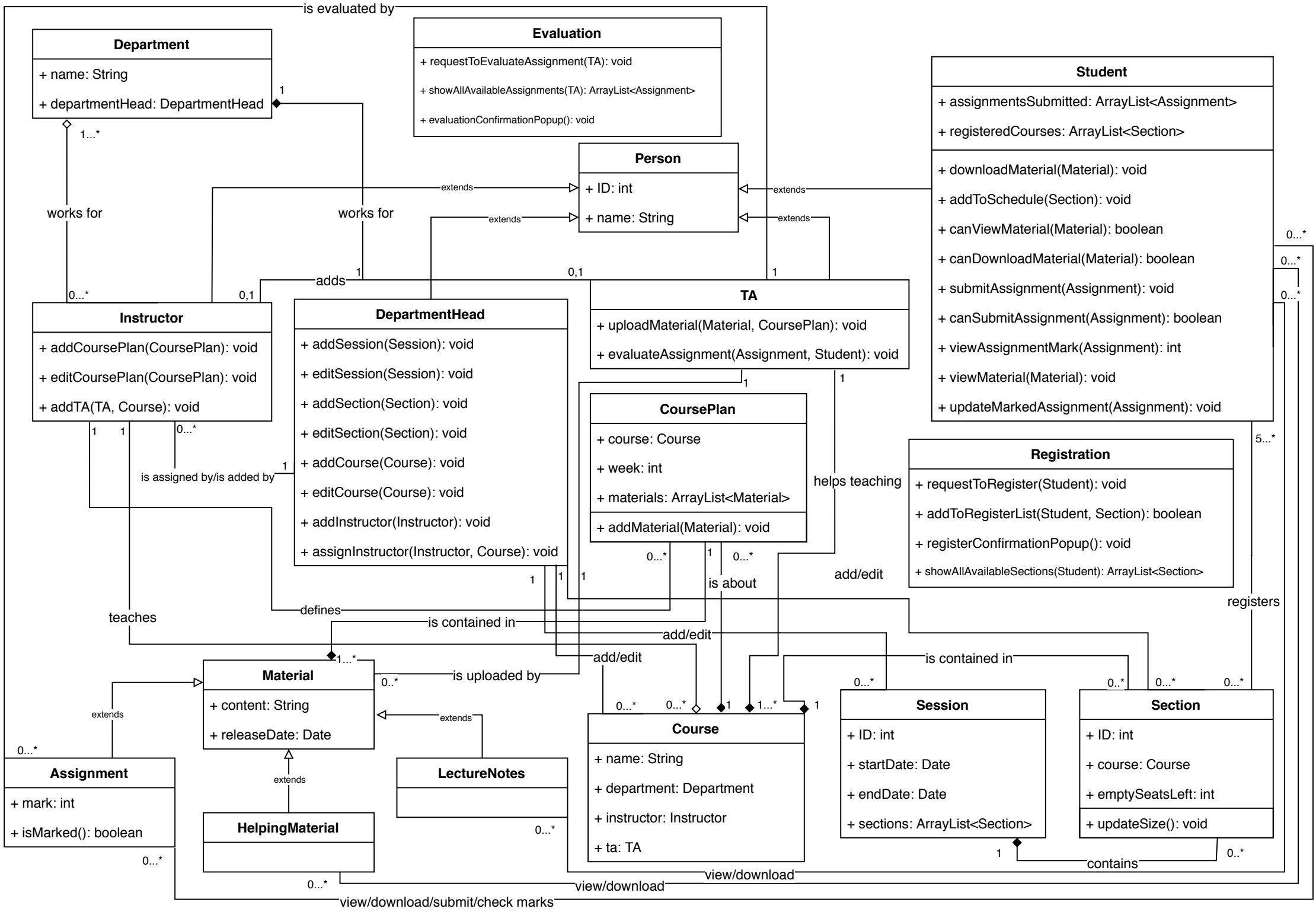
Assumption 11: We assume that a section opens only if there are at least 5 students register, and each section should have a data member indicates how many seats of the class left.

Assumption 12: We assume that an instructor can work for multiple departments. And a department can temporarily have zero instructors.

Assumption 13: The class ‘Registration’ contains a collection of static behaviors helping students to register a course.

Assumption 14: The class ‘Evaluation’ contains a collection of static behaviors helping a TA to evaluate an assignment.

The class diagram is shown on the next page


is evaluated by
Department
+ name: String
+ departmentHead: DepartmentHead
Evaluation
+ requestToEvaluateAssignment(TA): void
+ showAllAvailableAssignments(TA): ArrayList<Assignment>
+ evaluationConfirmationPopup(): void
Person
+ ID: int
+ name: String
Student
+ assignmentsSubmitted: ArrayList<Assignment>
+ registeredCourses: ArrayList<Section>
+ downloadMaterial(Material): void
+ addToSchedule(Section): void
+ canViewMaterial(Material): boolean
+ canDownloadMaterial(Material): boolean
+ submitAssignment(Assignment): void
+ canSubmitAssignment(Assignment): boolean
+ viewAssignmentMark(Assignment): int
+ viewMaterial(Material): void
+ updateMarkedAssignment(Assignment): void
works for
works for
extends
extends
extends
extends
adds
Instructor
+ addCoursePlan(CoursePlan): void
+ editCoursePlan(CoursePlan): void
+ addTA(TA, Course): void
DepartmentHead
+ addSession(Session): void
+ editSession(Session): void
+ addSection(Section): void
+ editSection(Section): void
+ addCourse(Course): void
+ editCourse(Course): void
+ addInstructor(Instructor): void
+ assignInstructor(Instructor, Course): void
TA
+ uploadMaterial(Material, CoursePlan): void
+ evaluateAssignment(Assignment, Student): void
CoursePlan
+ course: Course
+ week: int
+ materials: ArrayList<Material>
+ addMaterial(Material): void
Registration
+ requestToRegister(Student): void
+ addToRegisterList(Student, Section): boolean
+ registerConfirmationPopup(): void
+ showAllAvailableSections(Student): ArrayList<Section>
is assigned by/is added by
helps teaching
add/edit
is about
registers
teaches
defines
is contained in
add/edit
add/edit
is contained in
Material
+ content: String
+ releaseDate: Date
is uploaded by
extends
extends
extends
Assignment
+ mark: int
+ isMarked(): boolean
HelpingMaterial
LectureNotes
Course
+ name: String
+ department: Department
+ instructor: Instructor
+ ta: TA
Session
+ ID: int
+ startDate: Date
+ endDate: Date
+ sections: ArrayList<Section>
Section
+ ID: int
+ course: Course
+ emptySeatsLeft: int
+ updateSize(): void
contains
view/download
view/download
view/download/submit/check marks

Sequence diagrams base on the use cases selected

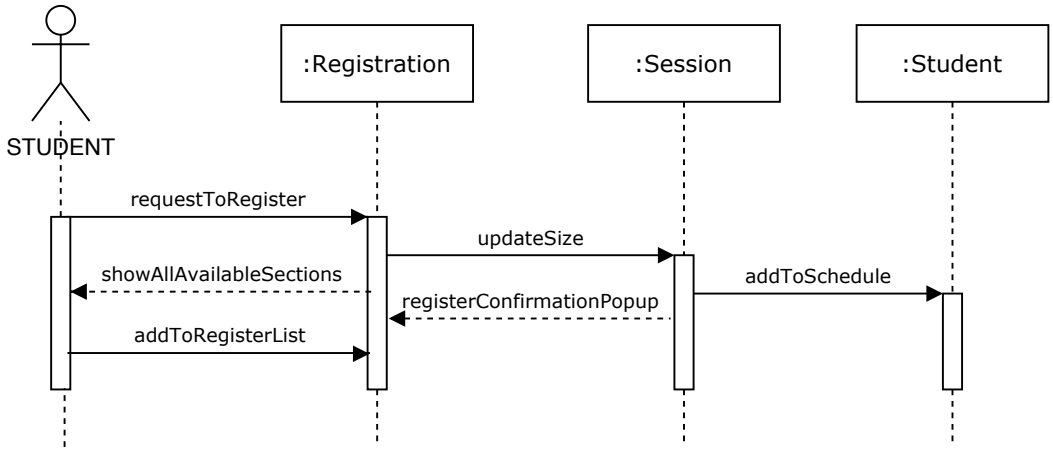
Use case 1: A student registers a course section

Use case 2: A TA evaluates an assignment

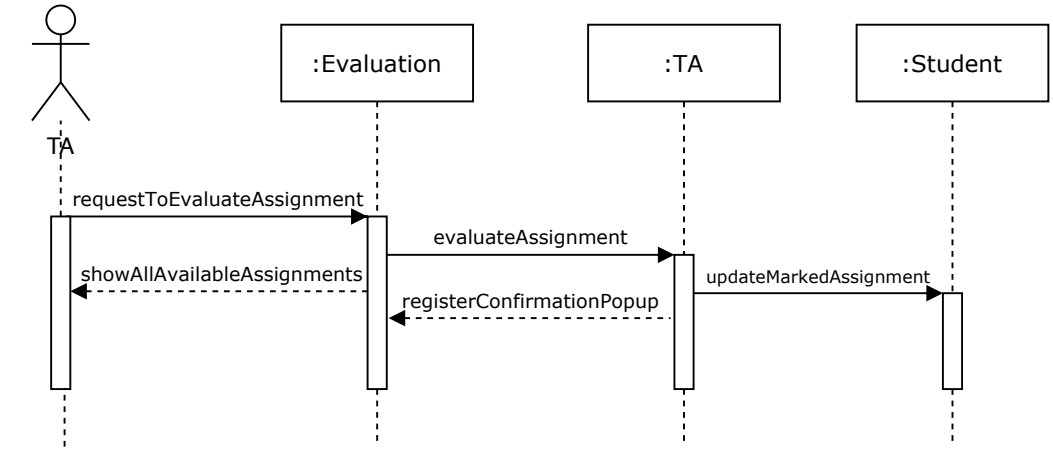
Use case 3: A TA uploads material to a course plan

Sequence diagrams are shown on the next page

The sequence diagram of Use Case #1



The sequence diagram of Use Case #2



The sequence diagram of Use Case #3