|  |  |  |  |
| --- | --- | --- | --- |
| **e of UML Notation within Design** | Very solid use of UML syntax and structure | Mostly correct, but a few minor areas where UML elements are used either incorrectly or inconsistently | One or more very major/significant misuses of UML Notation |
| **This overall design is** | Very Solid - clear, correct and consistent | Good Overall, with only a few Minor Issues | Two or More Major Problems/Issues |

Feedback

Observation #1: "Hired Instructor/Working Student" classes  
Pros: Could be used for storing attributes unique to an active instructor (with classes scheduled in the coming terms).  
Cons: Why not merge with Instructor Class? Could merge this and then add an Active attribute to signify if hired/working or not.  
  
Observation #2: Thorough Relationships section  
Pros: Clear and concise. All details appear to be there with no typos or room for improvement.  
Cons: None  
  
Observation #3: Course operations in Instructor Class  
Pros: Instructor would be the one to perform these tasks.  
Cons: Should exist in Course class and only executable by the assigned Instructor.

Feedback 51707

Observation #1: "Students" classes  
Pros:

* “MaxCoursePerTerm” and “ApprovedSequenceofCourse” attributes allow to distinguish whether a student is a working student or not;
* “ActiveEnrollmentStatus” attribute makes it easier to identify the student’s current degree active enrollment status.

Cons: The operation of “BecomeInstructor()” and relationship of ”BecomeInstructor/BecomeStudent” may not be clear. How to determine if one student (instructor) is qualified to “become” an instructor (student)?   
  
Observation #2: “Courses”/”MainCourse”/”NonMainCourse” classes  
Pros:

* Good observation of main course in the requirement document;
* Distinction between “MaxSeat” and “AvailableSeat” in “Course” attribute

Cons:

* “MeetPrerequisiteReirementCheck()” should not be an operation of “MainCourse” class, as it is a constraint for whether a student can take this class.
* The number of available seats for a course is determined by the instructors who have been assigned to teach the course. Thus, the “AvailableSeat” attribute is not determined by the Course itself. It could change term by term. It might be more appropriate to assign it to the attribute of “Session”, which is determined by the instructor and course together.

Observation #3: “Client” Class  
Pros: It might be useful to have this class to help manage the courses  
Cons: The HireInstructor() operation should not be decided by the client. Rather, the decision is usually made by the university or program. My understanding is that all the hidden operations, such as instructor hiring, course creation/update can be handled by the whole course management system we are developing.

Feedback 51640

Observation #1: "Students"/ “Instructor”/”Person” classes  
Pros: “Person” as a super class makes it convenient to achieve “A student (instructor) becomes an Instructor (Student)”

Cons:

* UUID should be an attribute of Person, as when student/instructor switch role, their ID don’t change
* “Occupation”/” Schedule” should be attributes of “Working student”, not every student has that.
* The “courses” attribute for student and instructor are not clear: Are they the courses that have been taken (instructed)?

Observation #2: “Assignment”/”Courses” classes  
Pros: It looks like this course corresponds to “Course Assignment” (like a session) and it make sense to have the attributes listed

Cons:

* It is not clear what “updates” mean between “Assignment” and “Courses”
* “Capacity” in the “Courses” class: the capacity should be attribute of “Assignment” because the capacity of a course is not determined by the course. Rather, it is determined by the available instructors.

Observation #3: “Request” Class  
Pros: It might be useful to have this class to manage all the requests  
Cons: However, the way it is designed might be difficult to distinguish different type of requests. Requesting enrollment/disenrollment and requesting record update are from different subjects. Furthermore, the attributes are a mix of Attributes from “Course”, “Assignment” and “Record”, making it difficult to identify which relationship they belong to.

Feedback 51677

Observation #1: "Teacher” classes  
Pros: Good choice of “Teacher” class and its attributes. “Person” as a super class of “Teacher” and “Student” makes it convenient to achieve “A student (teacher) becomes a teacher (student)”

Cons:

* Missing relationships between “Teacher” and “Course”, “Teacher” and “AcademicRecord”.
* Missing operations such as “Teach\_Course()”, “Manage\_Course()”, “Update\_Records()”, etc

Observation #2: “Student” classes and “AcademicRecord” and “DegreeProgram” classes  
Pros:

* Adding “academicRecord” and “degreeProgram” as attributes for a student
* Making “AcademicRecord” and “DegreeProgram” as objects

Cons:

* “AcademicRecord” and “DegreeProgram” should have student information, otherwise it is impossible to provide the attributes of the student
* “Create/Update Academic Record” should be operations of Teacher.

Observation #3: “CourseForTerm” and “Term” Class  
Pros: Good idea to have these classes to deliver available seats and hired instructor information   
Cons: “CourseForTerm” should be determined by instructors that are qualified to teach the courses and the students’ requests to take the courses. However, these relationships are missing.