College of Computer Science & Technology

2016 Software Engineering Practice Project

2016.5.30 - 2016.6.24

Course Registration Requirements

Problem Statement

As the head of information systems for Wylie College you are tasked with developing a new student registration system. The college would like a new client-server system to replace its much older system developed around mainframe technology. The new system will allow students to register for courses and view report cards from personal computers attached to the campus LAN. Professors will be able to access the system to sign up to teach courses as well as record grades.

Due to a decrease in federal funding, the college cannot afford to replace the entire system at once. The college will keep the existing course catalog database where all course information is maintained. This database is an Ingres relational database running on a DEC VAX. Fortunately the college has invested in an open SQL interface that allows access to this database from college's Unix servers. The legacy system performance is rather poor, so the new system must ensure that access to the data on the legacy system occurs in a timely manner. The new system will access course information from the legacy database but will not update it. The registrar's office will continue to maintain course information through another system.

At the beginning of each semester, students may request a course catalogue containing a list of course offerings for the semester. Information about each course, such as professor, department, and prerequisites, will be included to help students make informed decisions.

The new system will allow students to select four course offerings for the coming semester. In addition, each student will indicate two alternative choices in case the student cannot be assigned to a primary selection. Course offerings will have a maximum of ten students and a minimum of three students. A course offering with fewer than three students will be canceled. For each semester, there is a period of time that students can change their schedule. Students must be able to access the system during this time to add or drop courses. Once the registration process is completed for a student, the registration system sends information to the billing system so the student can be billed for the semester. If a course fills up during the actual registration process, the student must be notified of the change before submitting the schedule for processing.

At the end of the semester, the student will be able to access the system to view an electronic report card. Since student grades are sensitive information, the system must employ extra security measures to prevent unauthorized access.

Professors must be able to access the on-line system to indicate which courses they will be teaching. They will also need to see which students signed up for their course offerings. In addition, the professors will be able to record the grades for the students in each class.

Glossary

Introduction

This document is used to define terminology specific to the problem domain, explaining terms, which may be unfamiliar to the reader of the use-case descriptions or other project documents. Often, this document can be used as an informal *data dictionary*, capturing data definitions so that use-case descriptions and other project documents can focus on what the system must do with the information.

Definitions

The glossary contains the working definitions for the key concepts in the Course Registration System.

Course

A class offered by the university.

Course Offering

A specific delivery of the course for a specific semester – you could run the same course in parallel sessions in the semester. Includes the days of the week and times it is offered.

Course Catalog

The unabridged catalog of all courses offered by the university.

Faculty

All the professors teaching at the university.

Finance System

The system used for processing billing information.

Grade

The evaluation of a particular student for a particular course offering.

Professor

A person teaching classes at the university.

Report Card

All the grades for all courses taken by a student in a given semester.

Roster

All the students enrolled in a particular course offering.

Student

A person enrolled in classes at the university.

Schedule

The courses a student has selected for the current semester.

Transcript

The history of the grades for all courses, for a particular student sent to the finance system, which in turn bills the students.

Supplementary Specification

Objectives

The purpose of this document is to define requirements of the Course Registration System. This Supplementary Specification lists the requirements that are not readily captured in the use cases of the use-case model. The Supplementary Specifications and the use-case model together capture a complete set of requirements on the system.

Scope

This Supplementary Specification applies to the Course Registration System, which will be developed by the OOAD students.

This specification defines the non-functional requirements of the system; such as reliability, usability, performance, and supportability, as well as functional requirements that are common across a number of use cases. (The functional requirements are defined in the Use Case Specifications.)

References

None.

Functionality

Multiple users must be able to perform their work concurrently.

If a course offering becomes full while a student is building a schedule including that offering, the student must be notified.

Usability

The desktop user-interface shall be Windows 95/98 compliant.

Reliability

The system shall be available 24 hours a day 7 days a week, with no more than 10% down time.

Performance

The system shall support up to 2000 simultaneous users against the central database at any given time, and up to 500 simultaneous users against the local servers at any one time.

The system shall provide access to the legacy course catalog database with no more than a 10 second latency.
Note: Risk-based prototypes have found that the legacy course catalog database cannot meet our performance needs without some creative use of mid-tier processing power

The system must be able to complete 80% of all transactions within 2 minutes.

Supportability

None.

Security

The system must prevent students from changing any schedules other than their own, and professors from modifying assigned course offerings for other professors.

Only Professors can enter grades for students.

Only the Registrar is allowed to change any student information.

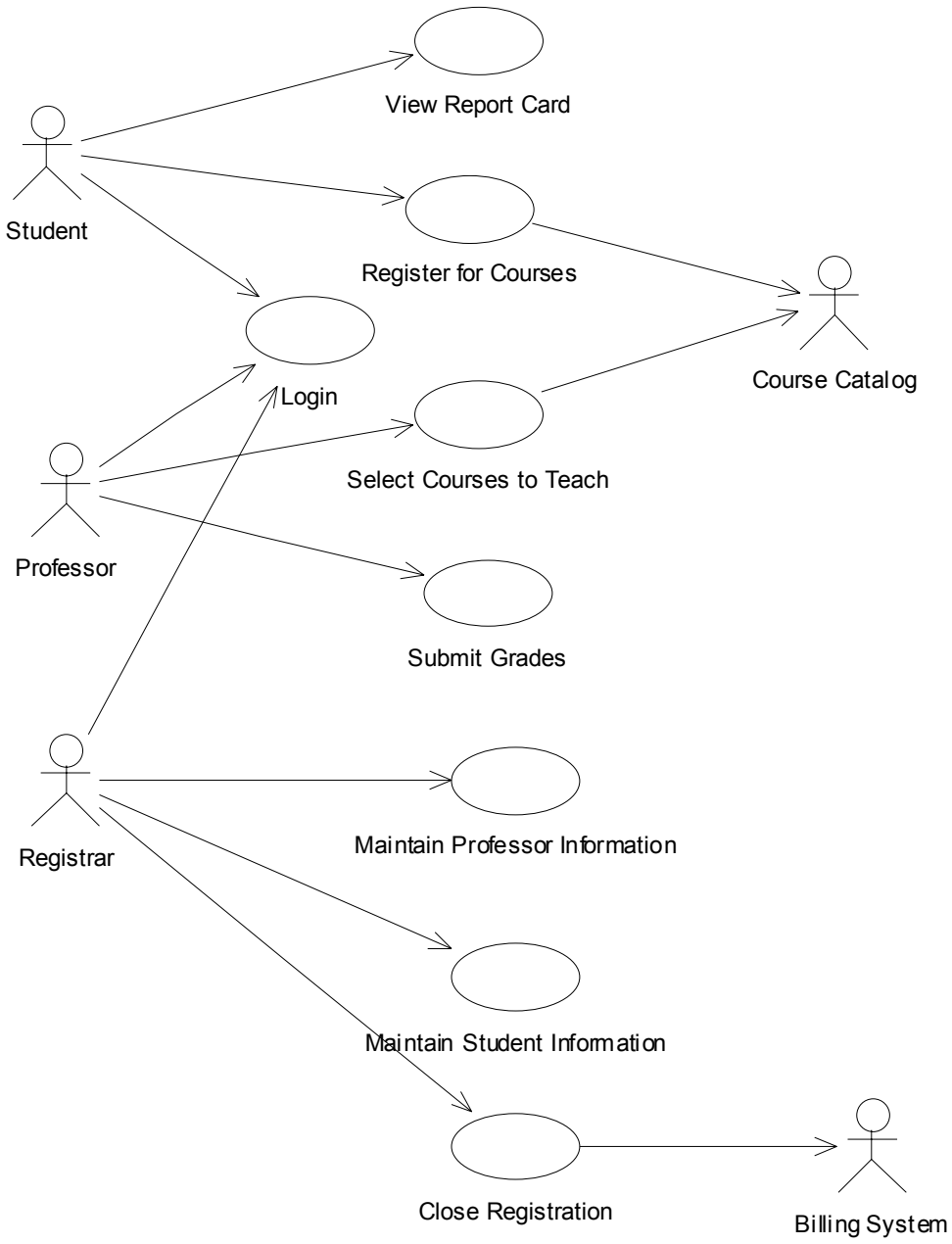
Design Constraints

The system shall integrate with an existing legacy system, the Course Catalog System, which is an RDBMS database.

The system shall provide a Windows-based desktop interface.

Use-Case Model

Course Registration System Use-Case Model Main Diagram



Close Registration

Brief Description

This use case allows a Registrar to close the registration process. Course offerings that do not have enough students are cancelled. Course offerings must have a minimum of three students in them. The billing system is notified for each student in each course offering that is not cancelled, so the student can be billed for the course offering.

Flow of Events

Basic Flow

This use case starts when the Registrar requests that the system close registration.

1. The system checks to see if registration is in progress. If it is, then a message is displayed to the Registrar, and the use case terminates. The Close Registration processing cannot be performed if registration is in progress.
2. For each course offering, the system checks if a professor has signed up to teach the course offering and at least three students have registered. If so, the system commits the course offering for each schedule that contains it.
3. For each schedule, the system "levels" the schedule: if the schedule does not have the maximum number of primary courses selected, the system attempts to select alternates from the schedule's list of alternates. The first available alternate course offerings will be selected. If no alternates are available, then no substitution will be made.
4. For each course offering, the system closes all course offerings. If the course offerings do not have at least three students at this point (some may have been added as a result of leveling), then the system cancels the course offering. The system cancels the course offering for each schedule that contains it.
5. The system calculates the tuition owed by each student for his current semester schedule and sends a transaction to the Billing System. The Billing System will send the bill to the students, which will include a copy of their final schedule.

Alternative Flows

No Professor for the Course Offering

If, in the **Basic Flow**, there is no professor signed up to teach the course offering, the system will cancel the course offering. The system cancels the course offering for each schedule that contains it.

Billing System Unavailable

If the system is unable to communicate with the Billing System, the system will attempt to re-send the request after a specified period. The system will continue to attempt to re-send until the Billing System becomes available.

Special Requirements

None.

Pre-Conditions

The Registrar must be logged onto the system in order for this use case to begin.

Post-Conditions

If the use case was successful, registration is now closed. If not, the system state remains unchanged.

Extension Points

None.

Login

Brief Description

This use case describes how a user logs into the Course Registration System.

Flow of Events

Basic Flow

This use case starts when the actor wishes to log into the Course Registration System.

1. The actor enters his/her name and password.
2. The system validates the entered name and password and logs the actor into the system.

Alternative Flows

Invalid Name/Password

If, in the **Basic Flow**, the actor enters an invalid name and/or password, the system displays an error message. The actor can choose to either return to the beginning of the **Basic Flow** or cancel the login, at which point the use case ends.

Special Requirements

None.

Pre-Conditions

The system is in the login state and has the login screen displayed.

Post-Conditions

If the use case was successful, the actor is now logged into the system. If not, the system state is unchanged.

Extension Points

None.

Maintain Professor Information

Brief Description

This use case allows the Registrar to maintain professor information in the registration system. This includes adding, modifying, and deleting professors from the system.

Flow of Events

Basic Flow

This use case starts when the Registrar wishes to add, change, and/or delete professor information in the system.

1. The system requests that the Registrar specify the function he/she would like to perform (either Add a Professor, Update a Professor, or Delete a Professor)
2. Once the Registrar provides the requested information, one of the sub flows is executed.
   If the Registrar selected "Add a Professor", the **Add a Professor** subflow is executed.
   If the Registrar selected "Update a Professor", the **Update a Professor** subflow is executed.
   If the Registrar selected "Delete a Professor", the **Delete a Professor** subflow is executed.

Add a Professor

The system requests that the Registrar enter the professor information. This includes:
- name
- date of birth
- social security number
- status
- department

1. Once the Registrar provides the requested information, the system generates and assigns a unique id number to the professor. The professor is added to the system.
2. The system provides the Registrar with the new professor id.

Update a Professor

1. The system requests that the Registrar enter the professor id.
2. The Registrar enters the professor id. The system retrieves and displays the professor information.
3. The Registrar makes the desired changes to the professor information. This includes any of the information specified in the **Add a Professor** sub-flow.
4. Once the Registrar updates the necessary information, the system updates the professor record.

Delete a Professor

1. The system requests that the Registrar enter the professor id
2. The Registrar enters the professor id. The system retrieves and displays the professor information.
3. The system prompts the Registrar to confirm the deletion of the professor.
4. The Registrar verifies the deletion.
5. The system deletes the professor from the system.

Alternative Flows

Professor Not Found

If, in the **Update a Professor** or **Delete a Professor** sub-flows, a professor with the specified id number does not exist, the system displays an error message. The Registrar can then enter a different id number or cancel the operation, at which point the use case ends.

Delete Cancelled

If, in the **Delete A Professor** sub-flow, the Registrar decides not to delete the professor, the delete is cancelled, and the **Basic Flow** is re-started at the beginning.

Special Requirements

None.

Pre-Conditions

The Registrar must be logged onto the system before this use case begins.

Post-Conditions

If the use case was successful, the professor information is added, updated, or deleted from the system. Otherwise, the system state is unchanged.

Extension Points

None.

Maintain Student Information

Brief Description

This use case allows the Registrar to maintain student information in the registration system. This includes adding, modifying, and deleting Students from the system.

Flow of Events

Basic Flow

This use case starts when the Registrar wishes to add, change, and/or delete student information in the system.

1. The system requests that the Registrar specify the function he/she would like to perform (either Add a Student, Update a Student, or Delete a Student)
2. Once the Registrar provides the requested information, one of the sub flows is executed.
   If the Registrar selected "Add a Student", the **Add a Student** subflow is executed.
   If the Registrar selected "Update a Student", the **Update a Student** subflow is executed.
   If the Registrar selected "Delete a Student", the **Delete a Student** subflow is executed.

Add a Student

1. The system requests that the Registrar enter the student information. This includes:
   - name
   - date of birth
   - social security number
   - status
   - graduation date
2. Once the Registrar provides the requested information, the system generates and assigns a unique id number to the student. The student is added to the system.
3. The system provides the Registrar with the new student id.

Update a Student

1. The system requests that the Registrar enter the student id.
2. The Registrar enters the student id. The system retrieves and displays the student information.
3. The Registrar makes the desired changes to the student information. This includes any of the information specified in the **Add a Student** sub-flow.
4. Once the Registrar updates the necessary information, the system updates the student information.

Delete a Student

1. The system requests that the Registrar enter the student id
2. The Registrar enters the student id. The system retrieves and displays the student information.
3. The system prompts the Registrar to confirm the deletion of the student.
4. The Registrar verifies the deletion.
5. The system deletes the student from the system.

Alternative Flows

Student Not Found

If, in the **Update a Student** or **Delete a Student** sub-flows, a student with the specified id number does not exist, the system displays an error message. The Registrar can then enter a different id number or cancel the operation, at which point the use case ends.

Delete Cancelled

If, in the **Delete A Student** sub-flow, the Registrar decides not to delete the student, the delete is cancelled and the **Basic Flow** is re-started at the beginning.

Special Requirements

None.

Pre-Conditions

The Registrar must be logged onto the system before this use case begins.

Post-Conditions

If the use case was successful, the student information is added, updated, or deleted from the system. Otherwise, the system state is unchanged.

Extension Points

None.

Register for Courses

Brief Description

This use case allows a Student to register for course offerings in the current semester. The Student can also update or delete course selections if changes are made within the add/drop period at the beginning of the semester. The Course Catalog System provides a list of all the course offerings for the current semester.

Flow of Events

Basic Flow

This use case starts when a Student wishes to register for course offerings, or to change his/her existing course schedule.

1. The Student provides the function to perform (one of the sub flows is executed):
   If the Student selected "Create a Schedule", the **Create a Schedule** subflow is executed.
   If the Student selected "Update a Schedule", the **Update a Schedule** subflow is executed.
   If the Student selected "Delete a Schedule", the **Delete a Schedule** subflow is executed.

Create a Schedule

1. The system retrieves a list of available course offerings from the Course Catalog System and displays the list to the Student.
2. The Select Offerings subflow is executed.
3. The Submit Schedule subflow is executed.

Update a Schedule

1. The system retrieves and displays the Student's current schedule (e.g., the schedule for the current semester).
2. The system retrieves a list of available course offerings from the Course Catalog System and displays the list to the Student.
3. The Student may update the course selections on the current selection by deleting and adding new course offerings. The Student selects the course offerings to add from the list of available course offerings. The Student also selects any course offerings to delete from the existing schedule.
4. Once the student has made his/her selections, the system updates the schedule for the Student using the selected course offerings.
5. The Submit Schedule subflow is executed.

Delete a Schedule

1. The system retrieves and displays the Student's current schedule (e.g., the schedule for the current semester).
2. The system prompts the Student to confirm the deletion of the schedule.
3. The Student verifies the deletion.
4. The system deletes the Schedule. If the schedule contains "enrolled in" course offerings, the Student must be removed from the course offering.

Select Offerings

The Student selects 4 primary course offerings and 2 alternate course offerings from the list of available offerings.

Once the student has made his/her selections, the system creates a schedule for the Student containing the selected course offerings.

Submit Schedule

For each selected course offering on the schedule not already marked as "enrolled in", the system verifies that the Student has the necessary prerequisites, that the course offering is open, and that there are no schedule conflicts.

The system then adds the Student to the selected course offering. The course offering is marked as "enrolled in" in the schedule.

The schedule is saved in the system.

Alternative Flows

Save a Schedule

At any point, the Student may choose to save a schedule rather than submitting it. If this occurs, the Submit Schedule step is replaced with the following:

The course offerings not marked as "enrolled in" are marked as "selected" in the schedule.

The schedule is saved in the system.

Unfulfilled Prerequisites, Course Full, or Schedule Conflicts

If, in the **Submit Schedule** sub-flow, the system determines that the Student has not satisfied the necessary prerequisites, or that the selected course offering is full, or that there are schedule conflicts, an error message is displayed. The Student can either select a different course offering and the use case continues, save the schedule, as is (see **Save a Schedule** subflow), or cancel the operation, at which point the **Basic Flow** is re-started at the beginning.

No Schedule Found

If, in the **Update a Schedule** or **Delete a Schedule** sub-flows, the system is unable to retrieve the Student's schedule, an error message is displayed. The Student acknowledges the error, and the **Basic Flow** is re-started at the beginning.

Course Catalog System Unavailable

If the system is unable to communicate with the Course Catalog System, the system will display an error message to the Student. The Student acknowledges the error message, and the use case terminates.

Course Registration Closed

When the use case starts, if it is determined that registration for the current semester has been closed, a message is displayed to the Student, and the use case terminates. Students cannot register for course offerings after registration for the current semester has been closed.

Delete Cancelled

If, in the **Delete A Schedule** sub-flow, the Student decides not to delete the schedule, the delete is cancelled, and the **Basic Flow** is re-started at the beginning.

Special Requirements

None.

Pre-Conditions

The Student must be logged onto the system before this use case begins.

Post-Conditions

If the use case was successful, the student schedule is created, updated, or deleted. Otherwise, the system state is unchanged.

Extension Points

None.

Select Courses to Teach

Brief Description

This use case allows a Professor to select the course offerings from the course catalog for the courses that he/she is eligible for and wishes to teach in the upcoming semester.

Flow of Events

Basic Flow

This use case starts when a Professor wishes to sign up to teach some course offerings for the upcoming semester.

1. The system retrieves and displays the list of course offerings the professor is eligible to teach for the current semester. The system also retrieves and displays the list of courses the professor has previously selected to teach.
2. The professor selects and/or de-selects the course offerings that he/she wishes to teach for the upcoming semester.
3. The system removes the professor from teaching the de-selected course offerings.
4. The system verifies that the selected offerings do not conflict (i.e., have the same dates and times) with each other or any course offerings that the professor has previously signed up to teach. If there is no conflict, the system updates the course offering information for each offering the professor selects (i.e., records the professor as the instructor for the course offering).

Alternative Flows

No Course Offerings Available

If, in the **Basic Flow**, the professor is not eligible to teach any course offerings in the upcoming semester, the system will display an error message. The professor acknowledges the message and the use case ends.

Schedule Conflict

If the systems find a schedule conflict when trying to establish the course offerings the Professor should take, the system will display an error message indicating that a schedule conflict has occurred. The system will also indicate which are the conflicting courses. The Professor can either resolve the schedule conflict (i.e., by canceling his selection to teach one of the course offerings), or cancel the operation, in which case, any selections will be lost, and the use case ends.

Course Catalog System Unavailable

If the system is unable to communicate with the Course Catalog System, the system will display an error message to the Student. The Student acknowledges the error message, and the use case terminates.

Course Registration Closed

When the use case starts, if it is determined that registration for the current semester has been closed, a message is displayed to the Professor, and the use case terminates. Professors cannot change the course offerings they teach after registration for the current semester has been closed. If a professor change is needed after registration has been closed, it is handled outside the scope of this system.

Special Requirements

None.

Pre-Conditions

The Professor must be logged onto the system before this use case begins.

Post-Conditions

If the use case was successful, the course offerings a Professor is scheduled to teach have been updated. Otherwise, the system state is unchanged.

Extension Points

None.

Submit Grades

Brief Description

This use case allows a Professor to submit student grades for one or more classes completed in the previous semester.

Flow of Events

Basic Flow

This use case starts when a Professor wishes to submit student grades for one or more classes completed in the previous semester.

1. The system displays a list of course offerings the Professor taught in the previous semester.
2. The Professor selects a course offering.
3. The system retrieves a list of all students who were registered for the course offering. The system displays each student and any grade that was previously assigned for the offering.
4. For each student on the list, the Professor enters a grade: A, B, C, D, F, or I. The system records the student's grade for the course offering. If the Professor wishes to skip a particular student, the grade information can be left blank and filled in at a later time. The Professor may also change the grade for a student by entering a new grade.

Alternative Flows

No Course Offerings Taught

If, in the **Basic Flow**, the Professor did not teach any course offerings in the previous semester, the system will display an error message. The Professor acknowledges the message, and the use case ends.

Special Requirements

None.

Pre-Conditions

The Professor must be logged onto the system before this use case begins.

Post-Conditions

If the use case was successful, student grades for a course offering are updated. Otherwise, the system state is unchanged.

Extension Points

None.

View Report Card

Brief Description

This use case allows a Student to view his/her report card for the previously completed semester.

Flow of Events

Basic Flow

This use case starts when a Student wishes to view his/her report card for the previously completed semester.

1. The system retrieves and displays the grade information for each of the course offerings the Student completed during the previous semester.
2. When the Student indicates that he/she is done viewing the grades, the use case terminates.

Alternative Flows

No Grade Information Available

If, in the **Basic Flow**, the system cannot find any grade information from the previous semester for the Student, a message is displayed. Once the Student acknowledges the message, the use case terminates.

Special Requirements

None.

Pre-Conditions

The Student must be logged onto the system before this use case begins.

Post-Conditions

The system state is unchanged by this use case.

Extension Points

None.