2019

Student name: MD TAUHIDUL ISLAM

Supervisor name: Professor 李旭伟

Project Report

Report of final Course project

Project title： Course registration for SCU foreign students

Student number: 2016521460553

Student name: MD TAUHIDUL ISLAM

Project type: Individual Course project

Project Supervisor: Professor 李旭伟

Course title: Environment & Tools for Software Development

Grading Results:

Comments: _____

_____

_____

Time for Submitting the Report: 2019/06/18

Project Name: Course registration for SCU foreign students

Student Name: MD TAUHIDUL ISLAM Supervisor: Professor 李旭伟

Table of contents:

1. Problem Statement

As we don't have a course registration option for foreign student in our university student registration system. I would like to suggest a new client-server system. As our university already have this feature for Chinese student. so, I will not design the entire system to replace the current system. The college can keep the existing course information database where all course information is maintained.

The new system will allow overseas students to register for courses and view their report cards from personal computers attached to the campus LAN or using his own computer. In the new system, Professors will be able to get access to the system to sign up to teach courses and he also will be able to record grades.

As we are not going to replace the current database, the new system will get access to course information from the legacy database but it will not update it. The registrar's office will continue to maintain course information through another system as it used to.

In my new system, students can check course information containing a list of course offerings for the particular semester at the beginning of each particular semester. The new system will offer clear course information and requirements such as professor, department, and prerequisites, etc. it will help the students to select and register their expected courses successfully.

The new system will allow students to select at best four courses offered for the coming semester. In addition, there will be an additional 3 alternative course choice for each student in case the student cannot be assigned to a primary choice. we define Course offerings condition as it can have a maximum of 30 students and a minimum of 5 students. A course offering with fewer than 5 students will be canceled. we also give a schedule change case that will help to make a change in their primary choice within a particular time. we enable Students to get access to the system during this time to add or drop courses. Once the registration process is completed for a student, the system sends information to the billing system so that the student can make the required payment for the semester. when a course fills up during the actual registration process, the system will let know the student before submitting the schedule for processing.

At the end of the semester, the student will be allowed to get access to the system and he/she can see his/her own electronic report card. Since student grades are sensitive information, the system provides an extra security system to prevent unauthorized access.

The system will allow the Professors to get access to the online system to select the courses which they will teach. They are also allowed to see the students who signed up for their course offerings. In addition, the system provides grade recording methods. therefore, the professors will be able to record the grades for the students in each course.

2. List of use-cases and actors:

ACTORS	Use-cases
1. OVERSEAS_PROFeSSOR	1.LOGIN
2. rEGISTER_sCU	2.REGISTER FOR COURSE
3. sCU_SRTUDENT	3.VIEW REPORT CARD
4. sCU_BILLING_SYSTEM	4.SELECT COURSE TO TEACH
5. SCU_INFORMATION SYSTEM	5.SUBMIT GRADE
	6.PLAN A SCHEDULE
	7.CLOSE REGISTRATION
	8. MAINTAIN PROFILE

3. System's Use Case Diagram:

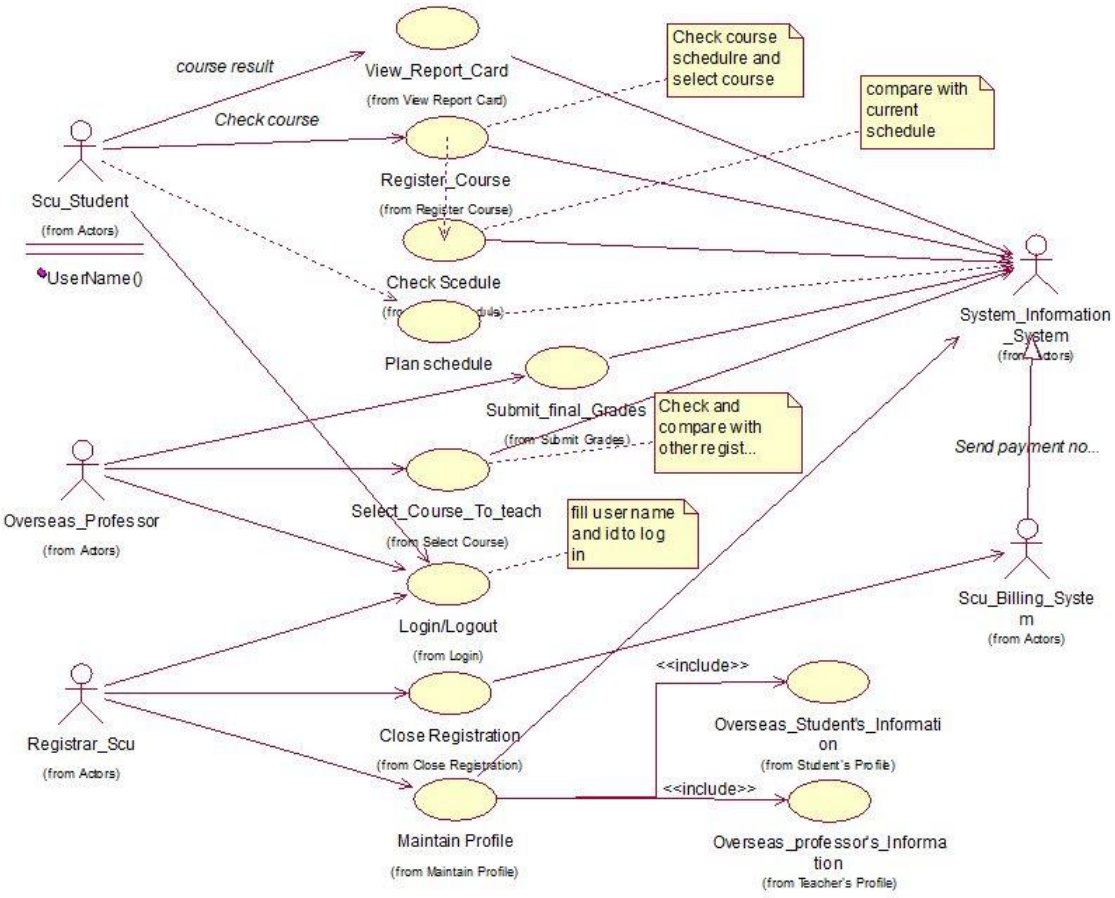


Figure: Use-Case Diagram

4. Use-Case Specification with template and it's activity diagram

4.1 login use-case specification and activity diagram:

4.1.1 Brief Description: A user of the System logs in to the System.

4.1.2 Flow of events

4.1.2.1 Basic Flow:

1. This use-case starts when a user tries (student, professor and register) to login to the system.
2. The System prompts the user for a username and password.
3. The user enters his/her username and password.
4. The system validates the entered username and password, making sure that the entered username is a valid username in the System, and that the required password is entered for the entered username
5. The user is signed in and returned to the home page as a Logged In User.
6. The use case ends.

4.1.2.2 Alternative Flow:

4.1.2.2.1 Invalid Name/Password:

1. The system presents the User with suggestions for changes necessary to allow the User to pass authentication.
2. Or the system allows the users to exit the system.

4.1.3 Special Requirements: None

4.1.4 Pre-Conditions : None

4.1.5 Post-Conditions: If the use case was successful, the actor is now logged into the system. If not, the system state is unchanged.

4.1.6 Extension Points: None

4.1.7 Activity Diagram:

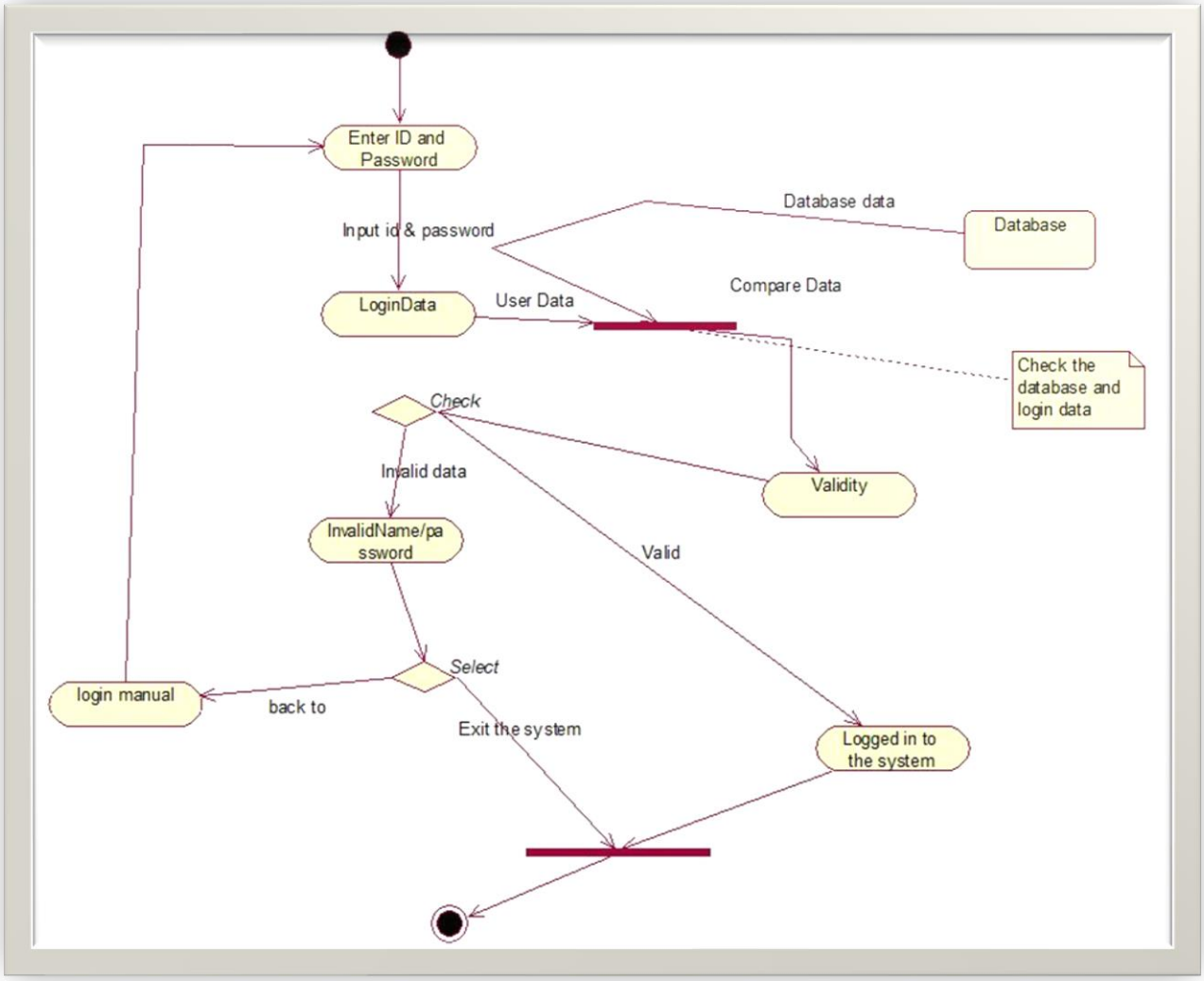


Figure: Activity Diagram (Login)

4.2 Register Courses

4.2.1 Brief Description

This use case allows a Student to register for course offerings in the current semester.

4.2.2 Flow of Events

4.2.2.1 Basic Flow

1. This use case starts when a Student wishes to register for course offerings, or to check course schedule or to select course.
2. The system requests that the Student specify the function he/she would like to perform. Here a student can check schedule (offering course & current course), check offering courses and select course.
3. Once the Student provides the requested information, one of the sub flows is executed. If the student selected “check a Schedule”, the Check a Schedule sub-flow is executed. If the Registrar selected “check current course offerings”, the check course offerings sub-flow is executed. If the student register selected “select course”, the select course sub flow is executed.

4.2.2.1.1 **Check a Schedule:** student can check course schedule in check schedule use case. Check schedule use-case provides current course offerings and enrolled courses schedule for students to avoid the time complicity.

4.2.2.1.2 **Check course offering:** the system will provide a list of current course offerings with the course details and professor information.

4.2.2.1.3 select course: The Student can select at best 4 primary course offerings and 3 alternate course offerings from the list of available offerings.

4.2.2.2 Alternative Flows

4**.2.2.2.1Unfulfilled Prerequisites, Course Full, or Schedule Conflicts:** If, in the selected course offering if student seat is full, or that there are schedule conflicts, or there is less student then pre-requirement an error message is displayed. The Student can either select a different course offering and the use case continues, save the schedule, as is or cancel the operation, at which point the Basic Flow is re-started at the beginning.

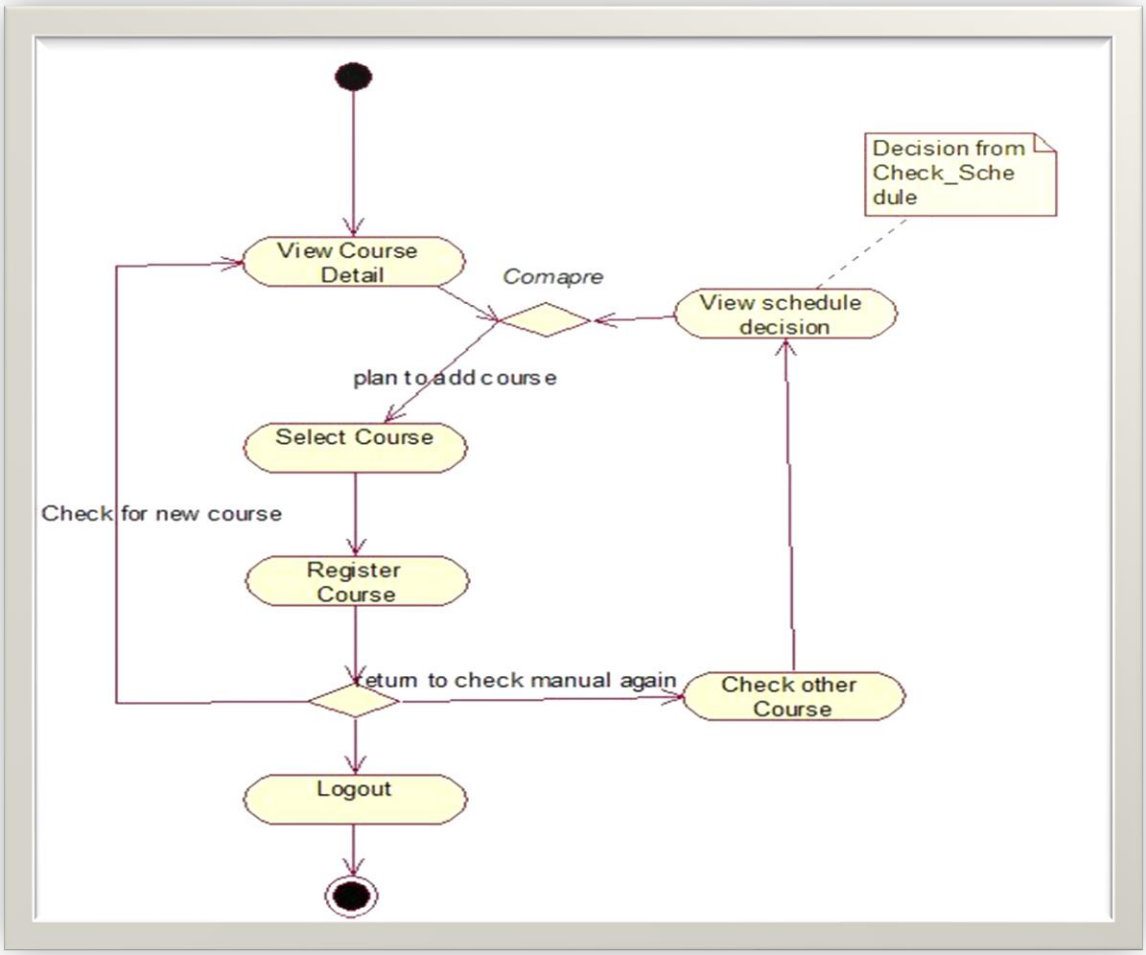
4.2.2.2.2Course Registration Closed: When the use case starts, if it is determined

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.

4.2.3 Special Requirements: None

4.2.4 Pre-Conditions: The Student must be logged onto the system before this use case begins.

4.2.5 Post-Conditions: If the use case was successful, the student registration is completed otherwise the system is unchanged.



4.2.6 Extension Points: None

Figure: Activity diagram (Register Course)

4.3 View Report Card

4.3.1 **Brief Description:** This use case allows a Student to view his/her report card for the previously completed semester.

4.3.2 Flow of Events

4.3.2.1 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.

4.3.2.2 Alternative Flows

4.3.2.2.1 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.

4.3.3 Special Requirements:

4.3.4 Pre-Conditions: The Student must be logged onto the system before this use case begins.

4.3.5 **Post-Conditions:** The system state is unchanged by this use case.

4.3.6 **Extension Points:** None

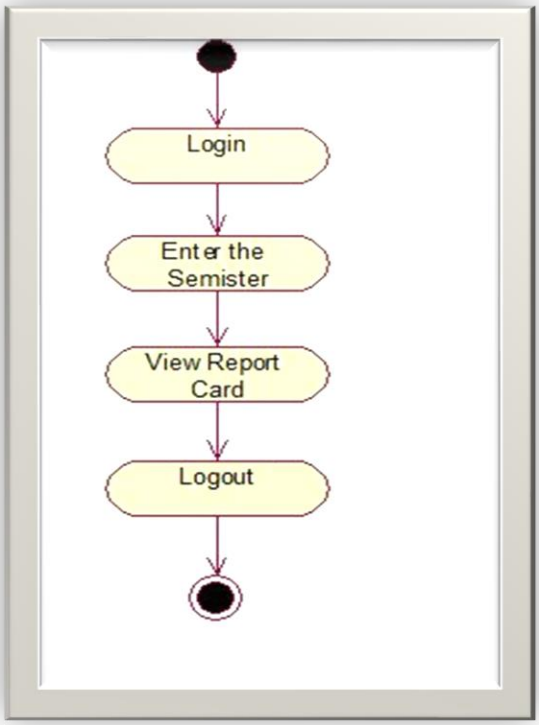


Figure: Activity diagram (view report card)

4.4 Select Courses to Teach:

4.4.1 Brief Description:

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

4.4.2 Flow of Events

4.4.3 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 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.

4.4.4 Alternative flow

4.4.4.1 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.

4.4.4.2 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 or cancel the operation, in which case, any selections will be lost, and the use case ends.

4.4.4.3 Course offerings is unavailable

If the system is unable to communicate with the Course offering's System, the system will display an error message to the Professor. The Professor acknowledges the error message, and the use case terminates

4.4.4.4 Registration closed:

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.

4.4.5 **Special Requirements:** None

4.4.6 Pre-Conditions:

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

4.4.7 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.

4.4.8 **Extension Points:** None

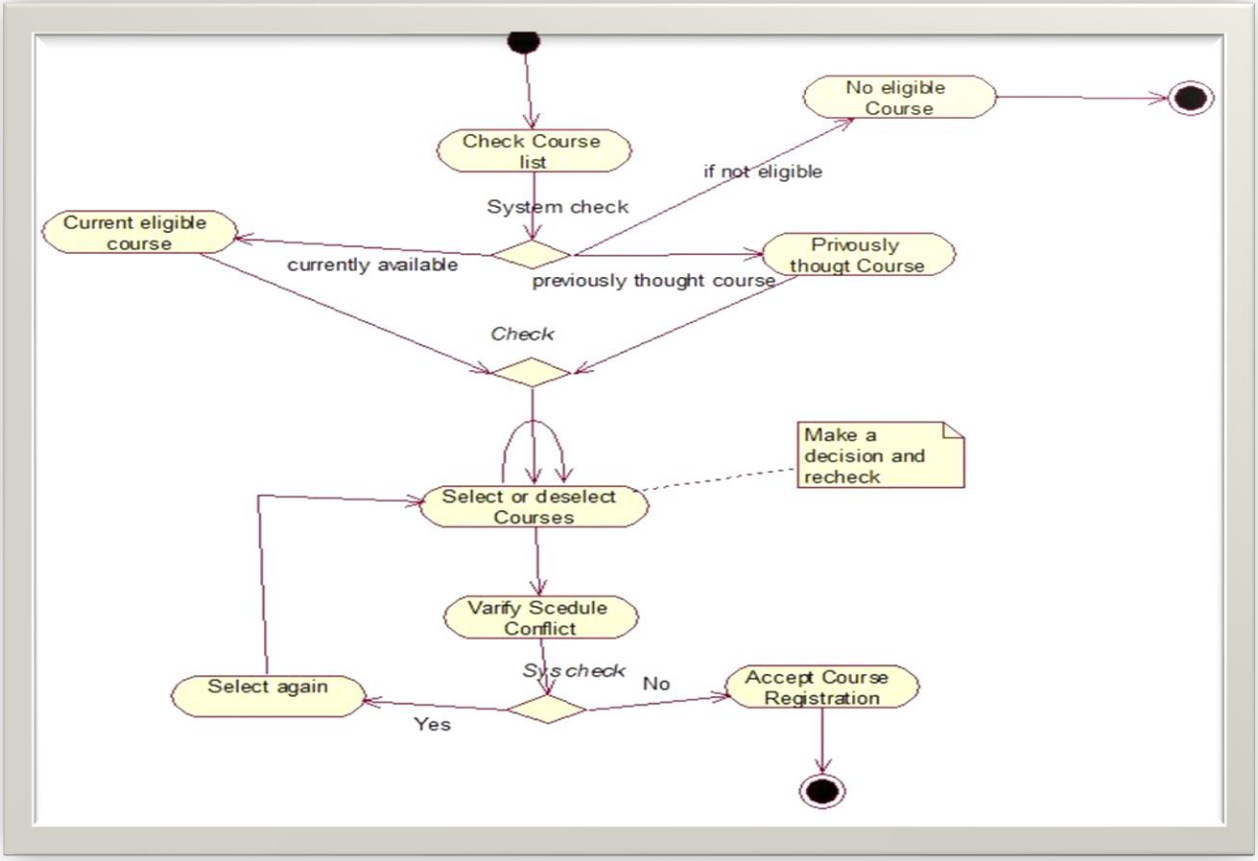


Figure: Activity Diagram (Select course to teach**)**

4.5 Submit Grades

4.5.1 Brief Description:

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

4.5.2 Flow of Events

4.5.3 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, or F. 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.

4.5.4 Alternative flow

4.5.4.1 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.

4.5.5 **Special Requirements:** None.

4.5.6 Pre-Conditions:

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

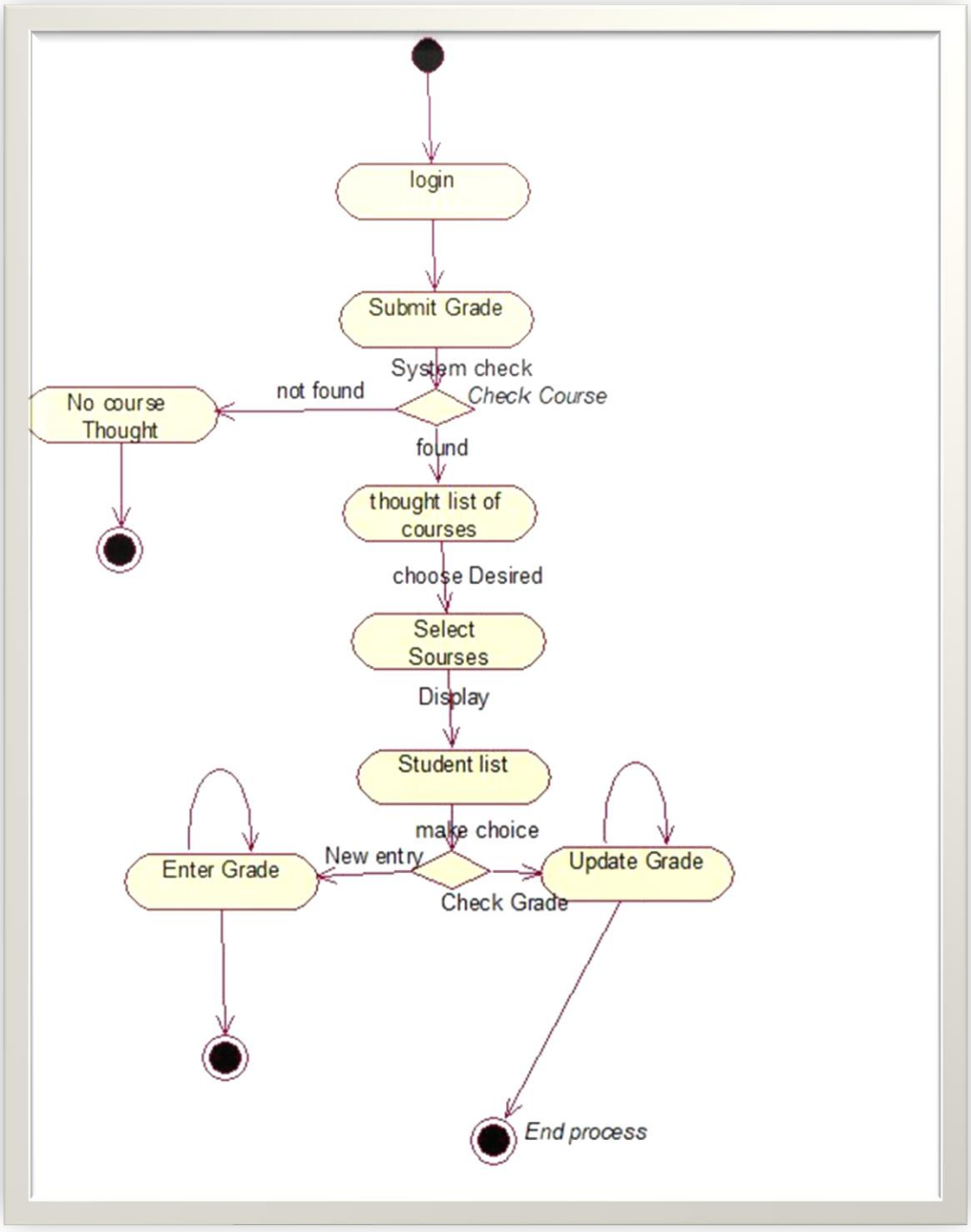
4.5.7 Post-Conditions:

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

4.5.8 **Extension Points:** None

4.5.9 **Activity Diagram**

Figure: Activity Diagram (Submit Grade)



4.6 Plan a schedule

4.6.1 Brief Description:

This use-case allows a user to check schedule to register a course or to change in course schedule.

4.6.2 Flow of Events

4.6.3 Basic flow:

This use-case start when a user tries to check schedule to register a course or to update a register.

1. The system requests that the Student specify the function he/she would like to perform. Here a student can check time (offering course & current course), check teacher, Change time and Change teacher.
2. Once the Student provides the requested information, one of the sub flows is executed. If the student selected “check time”, the Check a time sub-flow is executed. If the Registrar selected “check teacher”, the check teacher sub-flow is executed. If the student register selected “Change time”, the Change time sub flow is executed. If the student register selected “Change teacher”, the Change teacher sub flow is executed.

4.6.4 Alternative flow

4.6.4.1 Not registered:

If the student still didn’t register for any courses, the system will check message and execute not registered flow. The student will acknowledge the message and will check out the register use case.

4.6.5 **Special Requirements:** none

4.6.6 Pre-Conditions:

students must log in to the system.

4.6.7 Post-Conditions:

system will save the change the schedule if only the user can check the registered course otherwise system won’t save any change.

4.6.8 **Extension Points:** None

4.6.9 **Activity Diagram:**

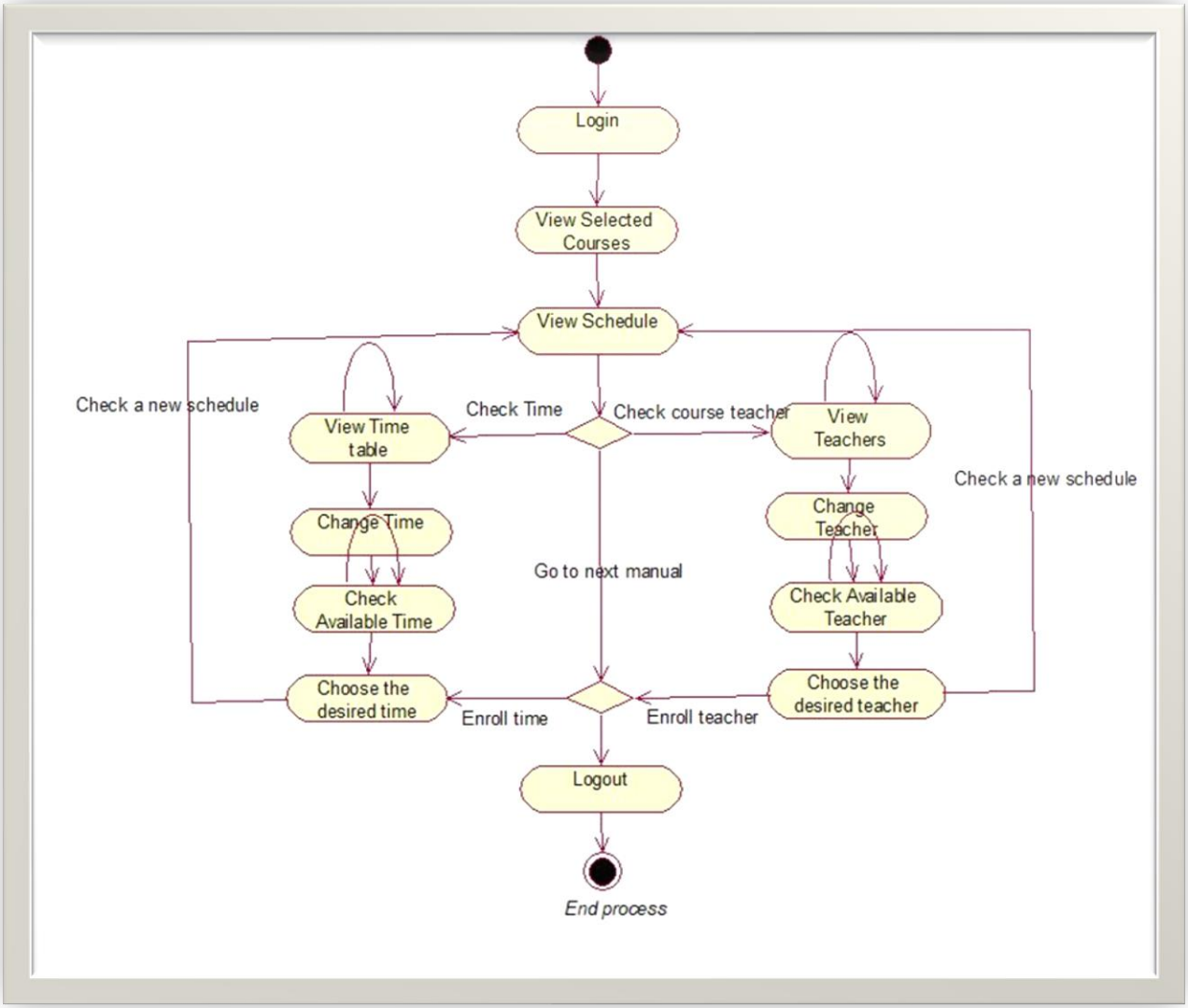


Figure: Activity Diagram (Plan_Schedule)

4.7 Close Registration

4.7.1 Brief Description:

This use case allows Registrar_Scu to close the registration process. Course offerings that do not have enough students are cancelled. Course offerings must have at least minimum of five students. The billing system is notified for each student in each course offering that is not cancelled, so the student can pay for their course.

4.7.2 Flow of Events

4.7.3 Basic flow:

This use case starts when the Registrar requests the system to 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 five 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 five students at this point, 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.

4.7.4 Alternative flow

4.7.4.1 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.

4.7.4.2 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.

4.7.5 Special Requirements: None

4.7.6 Pre-Conditions:

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

4.7.7 Post-Conditions:

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

4.7.8 **Extension Points:** None

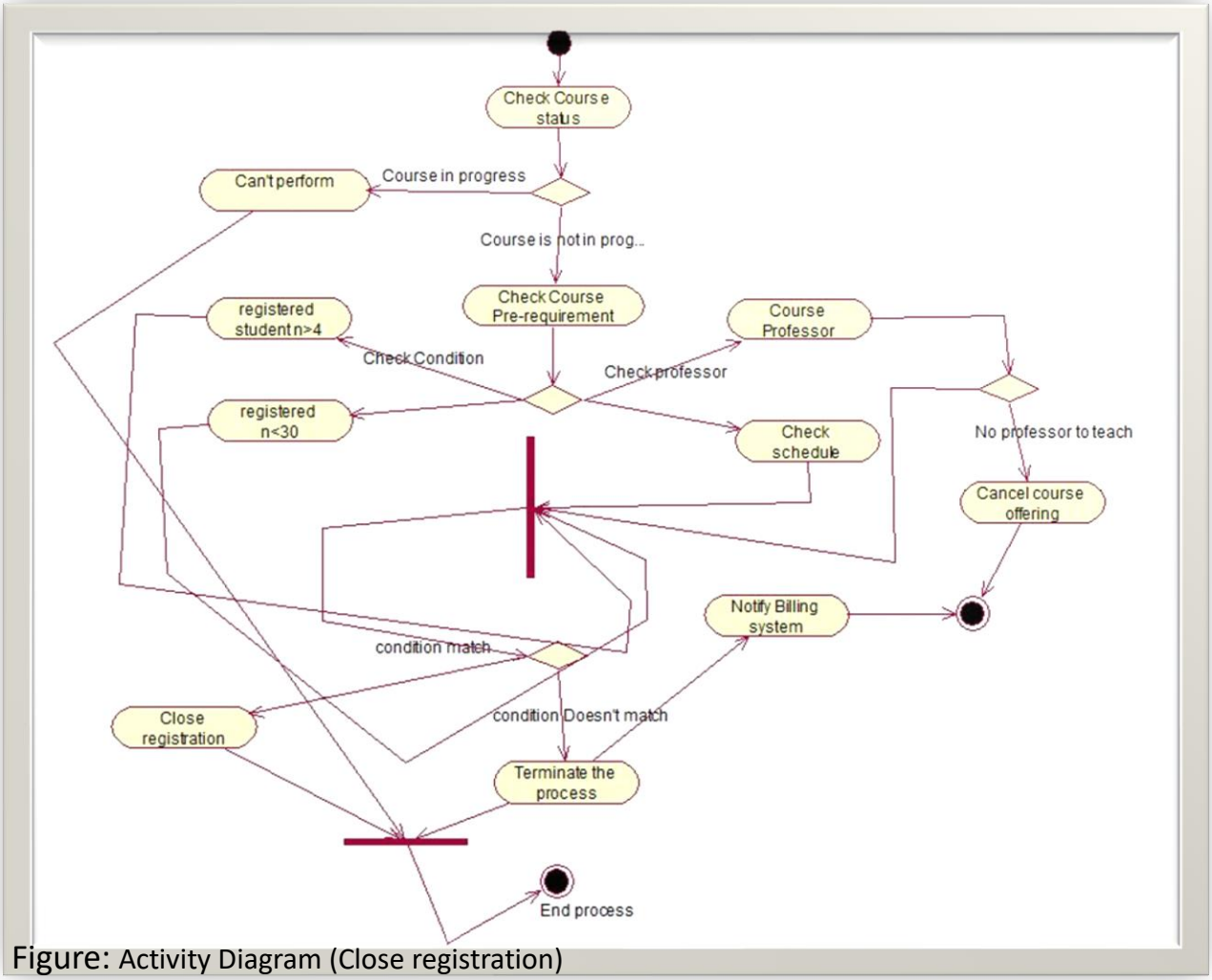


Figure: Activity Diagram (Close registration)

4.8 Maintain profile

(a)Maintain Professor Information

4.8.1 Brief Description:

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

4.8.2 Flow of Events

4.8.3 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_SCU to enter Professor Id to select the function specified in the system.
2. If the system can't find the Professor with entered Professor id, add a Professor function will be available for the Registrar_SCU to add a new Professor.
3. If the system can find the Professor with the entered id, it will pop up the update a Professor and delete a Professor for the Registrar_SCU.
4. 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 sub-flow is executed. If the Registrar selected "Update a Professor", the Update a Professor sub-flow is executed. If the Registrar selected "Delete a Professor", the Delete a Professor sub-flow is executed.

4.8.3.1 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.

4.8.3.2 Update a Professor:

1. The system requests that the Registrar_SCU enter the professor id.
2. The Registrar_SCU enters the professor id. The system retrieves and displays the professor information
3. The Registrar_SCU 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_SCU updates the necessary information, the system updates the professor record.

4.8.3.3 Delete a Professor:

1. The system requests that the Registrar_SCU enter the professor id.
2. The Registrar_SCU enters the professor id. The system retrieves and displays the professor information.

3. The system prompts the Registrar_SCU to confirm the deletion of the professor.
4. The Registrar_SCU verifies the deletion.
5. The system deletes the professor from the system.

4.8.4 Alternative Flows

4.8.4.1 Professor Not Found:

If, in the search flow, a professor with the specified id number does not exist, the system displays an error message. The Registrar_SCU can then add a new professor or enter a different id number or cancel the operation, at which point the use case ends.

4.8.4.2 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.

4.8.5 **Special Requirements:** None

4.8.6 Pre-Conditions:

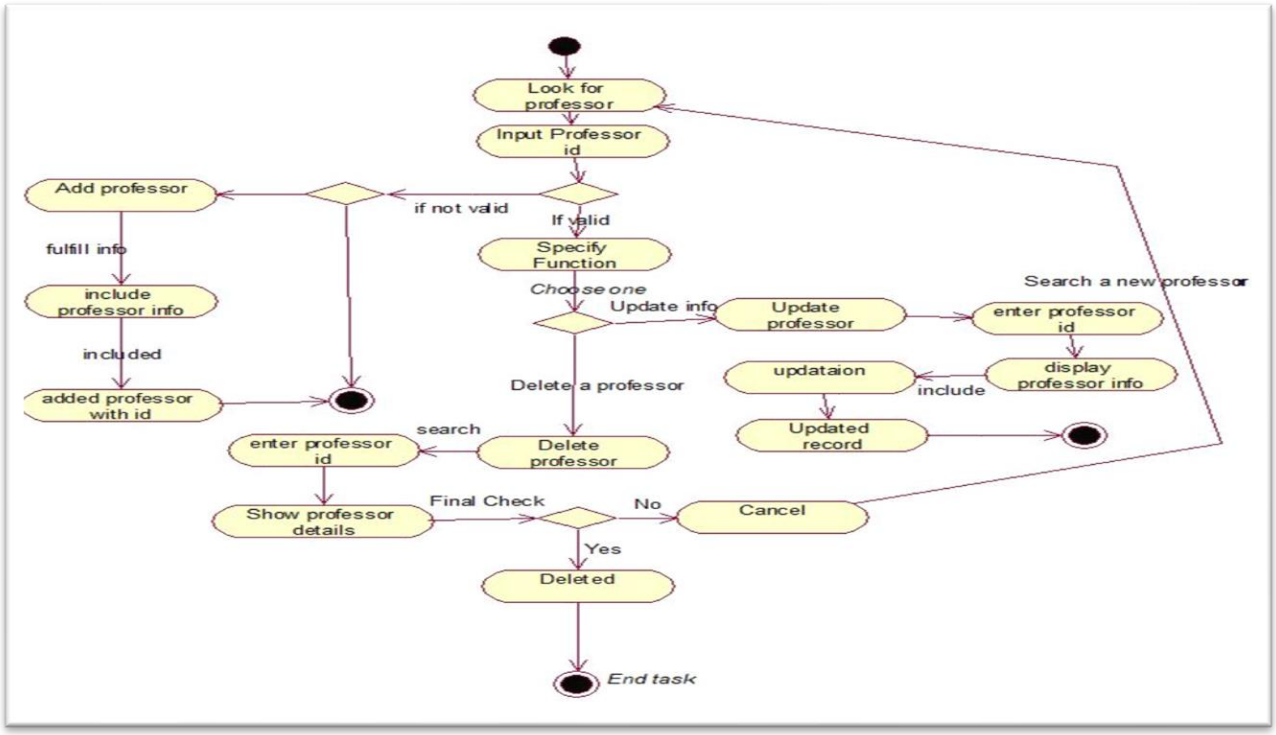
The Registrar must be logged onto the system before this use case begins.

4.8.7 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.

4.8.8 **Extension Point:** None

Figure: Activity Diagram (Maintain Professor)



(b) Maintain Student Information

4.8.1.1 Brief Description:

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

4.8.1.2 Flow of Events

4.8.1.3 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_SCU to enter student Id to select the function specified in the system.
2. If the system can't find the student with entered student id, add a student function will be available for the Registrar_SCU to add a new student.
3. If the system can find the student with the entered id, it will poop up the update a student and delete a student for the Registrar_SCU.
4. 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 sub-flow is executed. If the Registrar selected "Update a Student", the Update a Student sub-flow is executed. If the Registrar selected "Delete a Student", the Delete a Student sub-flow is executed.

4.8.1.3.1 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.

4.8.1.3.2 Update a student:

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

4.8.1.3.3 Delete a student:

1. The system retrieves and displays the student information.

2. The system prompts the Registrar to confirm the deletion of the student.
3. The Registrar verifies the deletion.
4. The system deletes the student from the system.

4.8.1.4 Alternative Flow

4.8.1.4.1 Student Not found:

If, in the Search Flow, 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.

4.8.1.4.2 Delete canceled:

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.

4.8.1.5 **Special Requirement:** None

4.8.1.6 Pre-Condition:

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

4.8.1.7 Post condition:

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

4.8.1.8 **Extension Point:** None

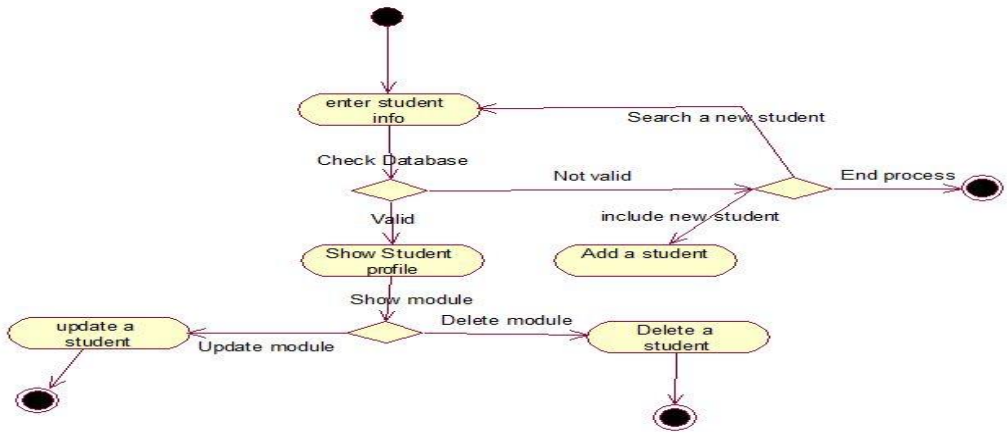
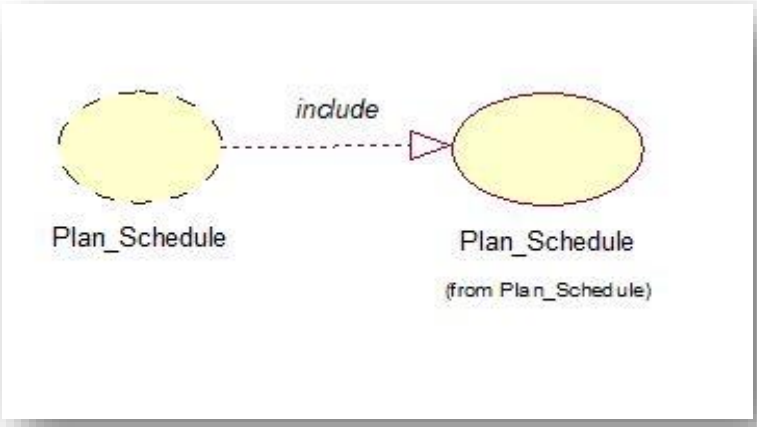


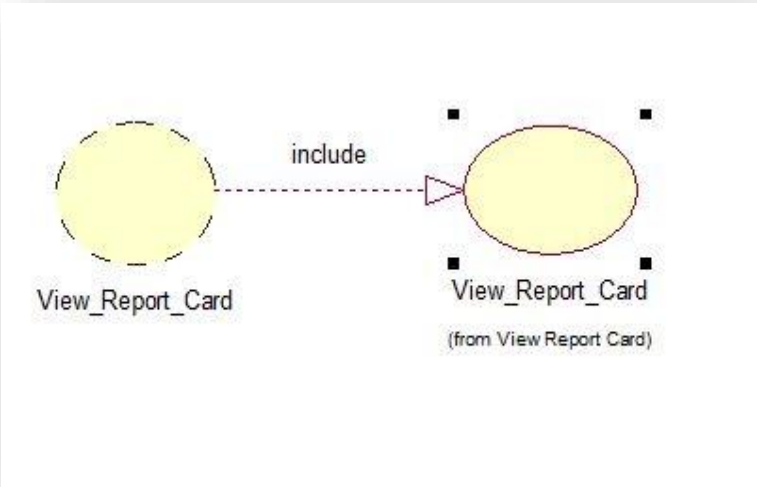
Figure: Activity Diagram (Maintain Student Profile)

5. Use-Case Realization (included dependency)

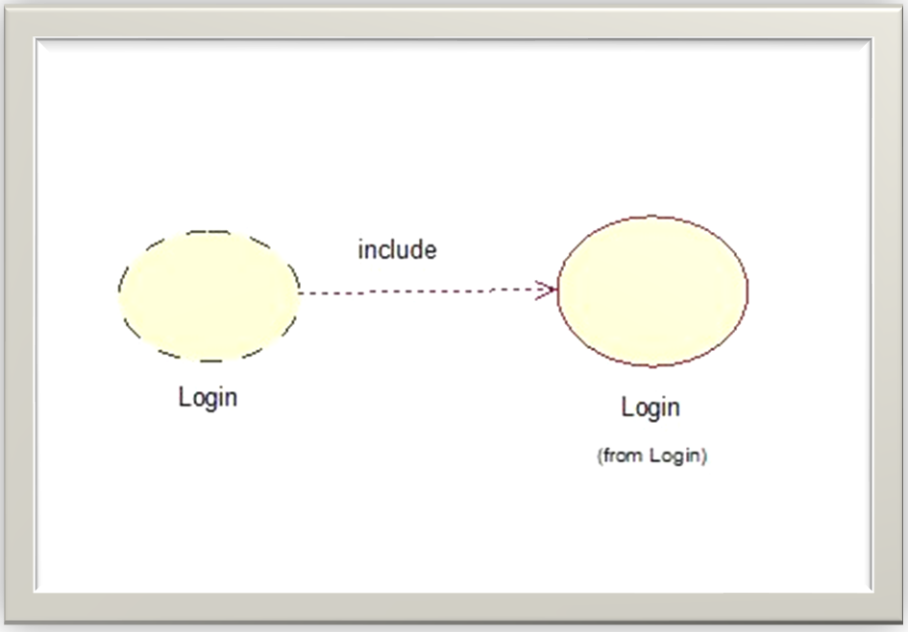
- **Use-Case: Plan_Schedule**



- **Use-Case: View_Report_Card**

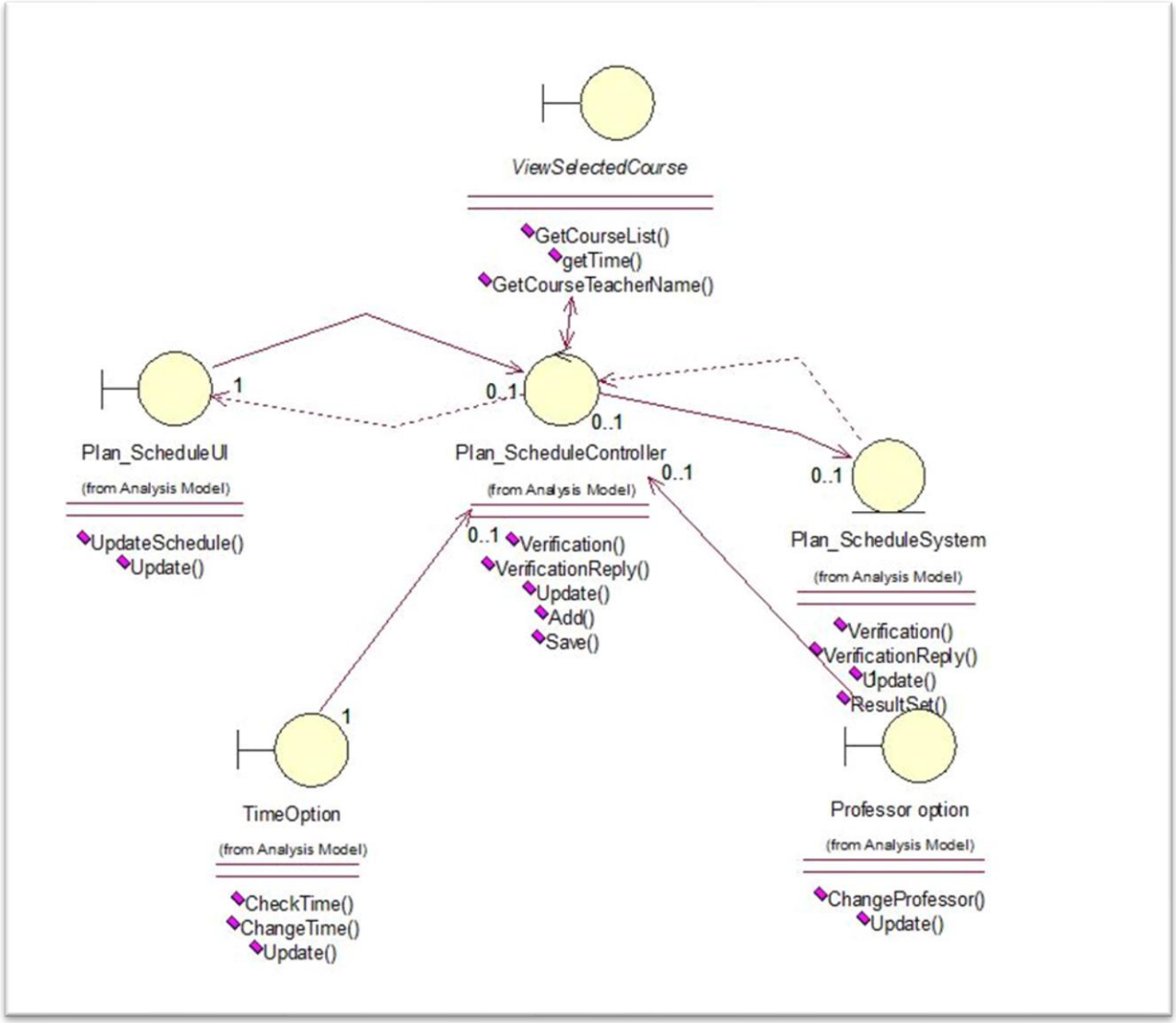


- Use-Case: Login



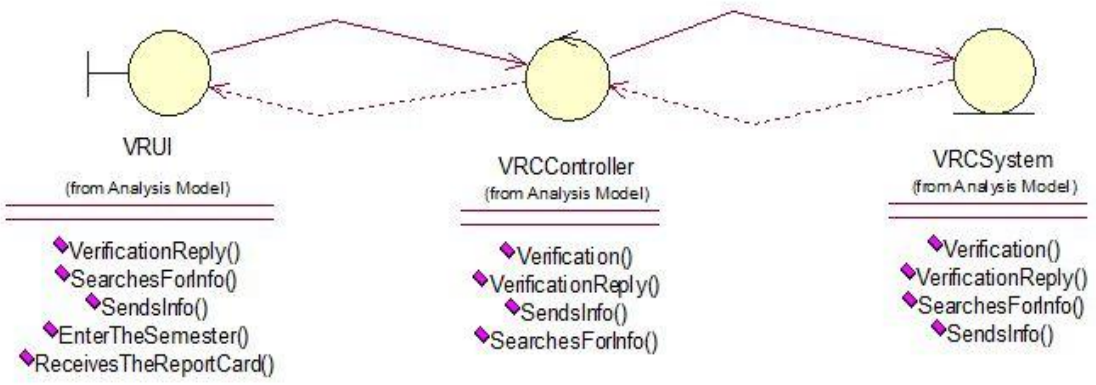
6. VOPC Class Diagram

- Plan_Schedule



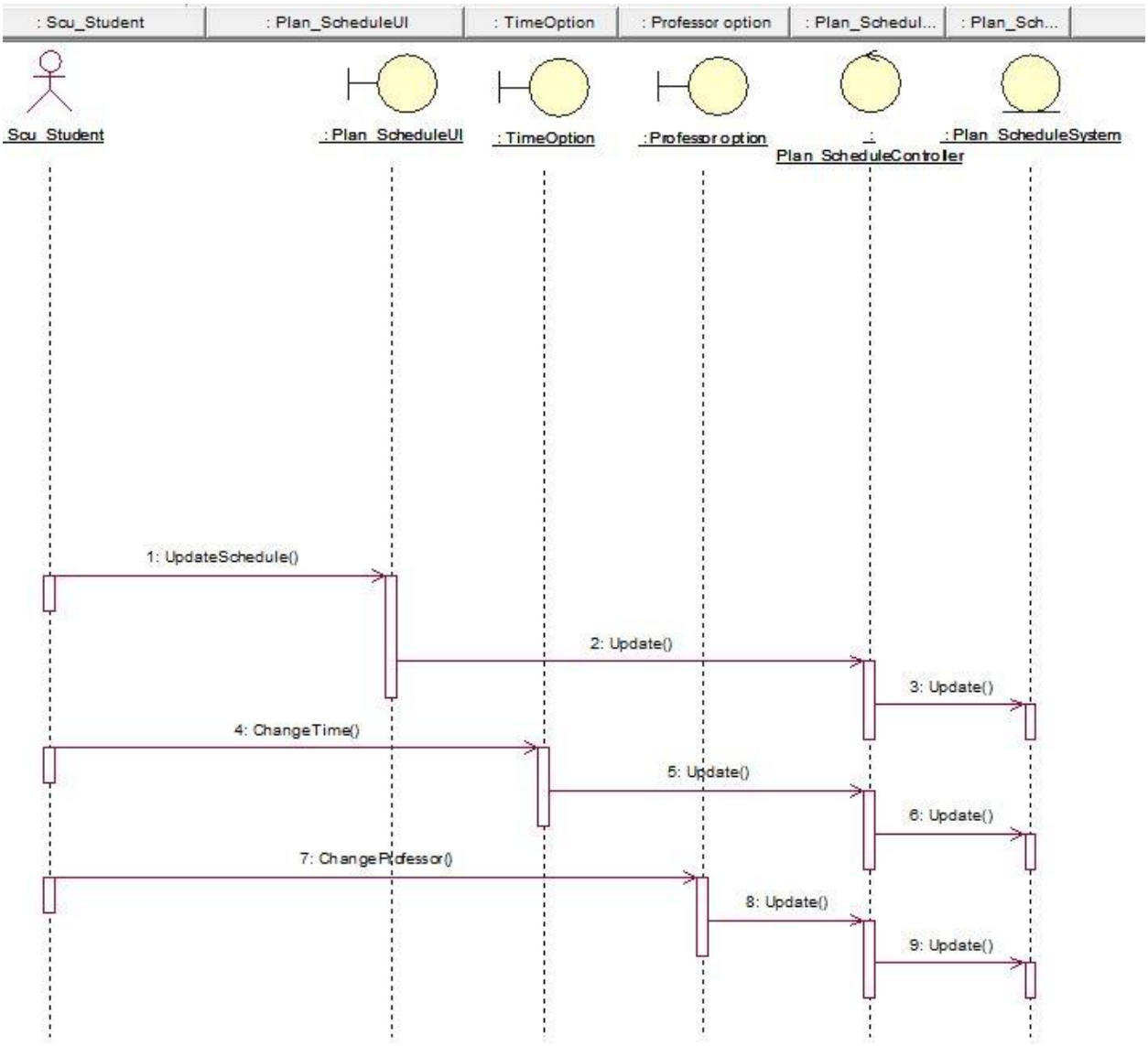
VOPC Class Diagram

- **View_Report_Card:**



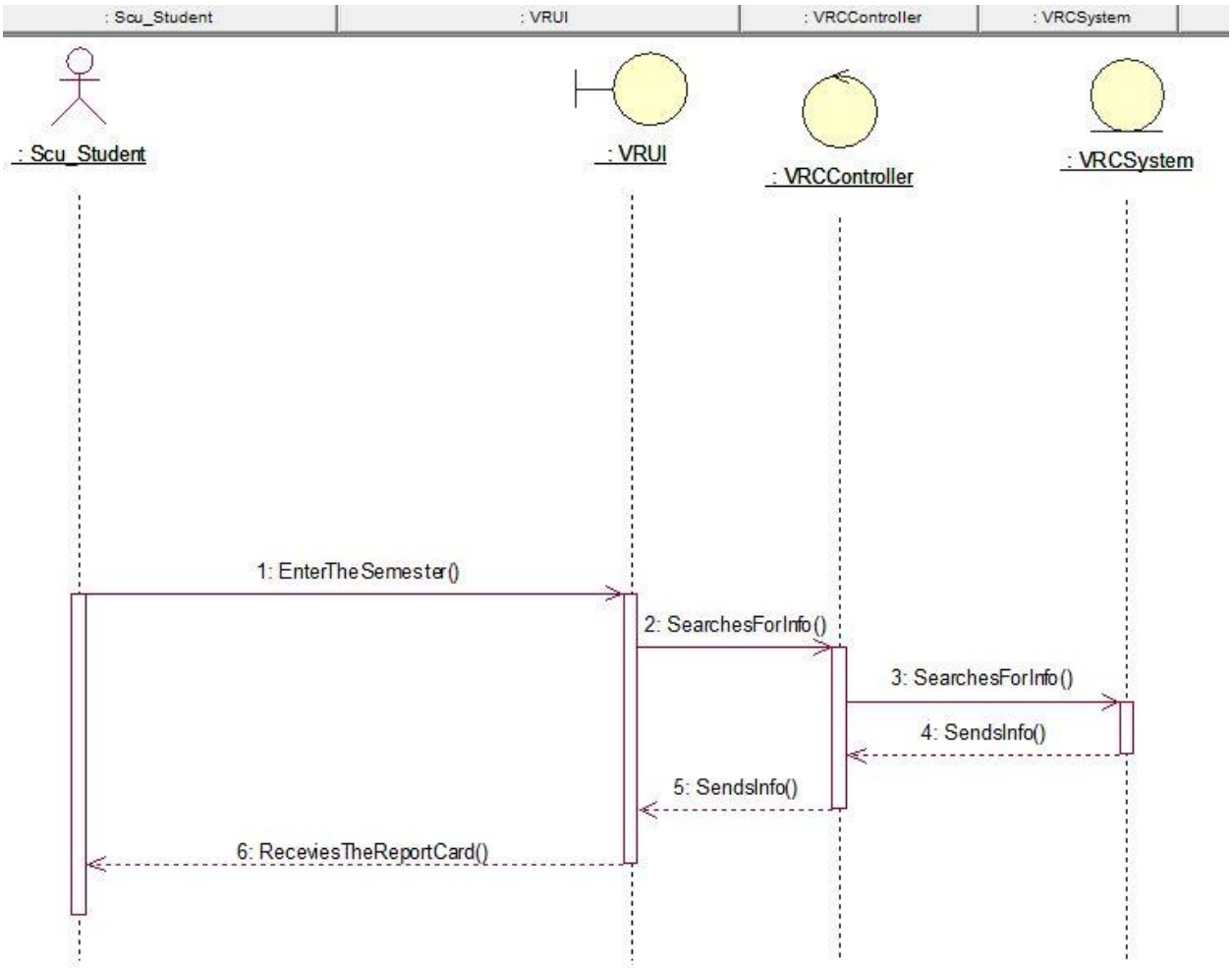
7. Sequence Diagram

1. Plan_Schedule:



7. Sequence Diagram

2. View_Report_Card:



8.Conclusion:

Student course registration indeed is an integral facet within the educational institution providing for adequate and efficient resource management aligned with available human resource. Although current available models have been designed with most of feature and developed technology. I hope that my proposed model will be useful in further development and updating.

In the end, I would like to thank my project supervisor Professor 李旭伟 for his valuable suggestion throughout the project.

9.References:

1. Alhir, S.S. (1998) *UML in a Nutshell*, O'Reilly and Associates , Sebastopol, CA.
2. Embarcadero Developer Network UML Tutorial
3. IBM UML Tutorial
4. SPARX Systems UML Tutorial
5. Visual Paradigm for UML User's Guide
6. Booch, G. , J. Rumbaugh and I. Jacobson (1999) The Unified Modeling Language User Guide, Addison Wesley , Reading, MA.
7. Course PPT.
8. Class room Exercise.