

**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
імені ТАРАСА ШЕВЧЕНКА**



ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра прикладних інформаційних систем

Звіт до лабораторної роботи №3

3 курсу

«Інженерія програмного забезпечення»

студента 2 курсу

групи ПП-22

спеціальності 122 «Комп'ютерні науки»

ОП «Прикладне програмування»

Шевлюк Вікторії Віталіївни

Перевірила:

Доц. Бойко Юлія Петрівна

Київ 2022

Тема: Розробка діаграм взаємодії

Мета роботи: вивчення діаграм взаємодії та їх застосування в процесі проектування.

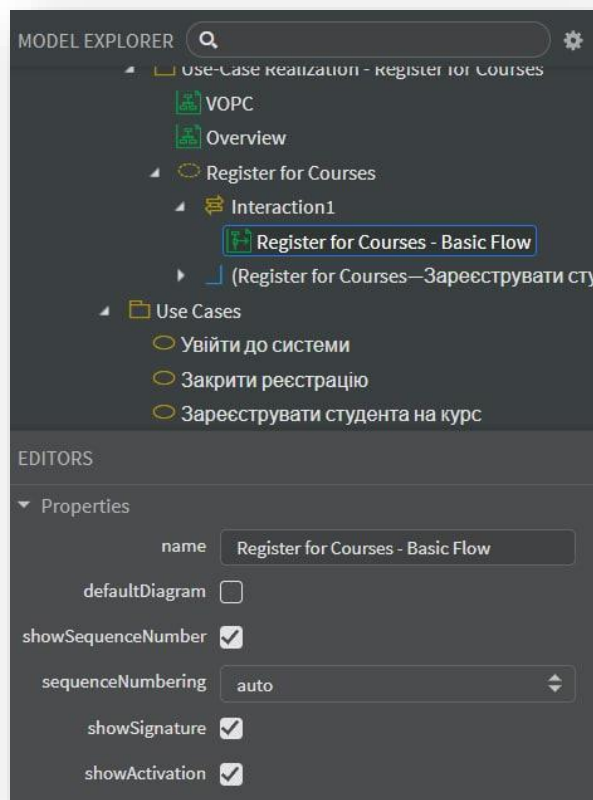
Хід роботи:

Вправа 3.1. Створення діаграми взаємодії основного потоку





Створимо діаграми послідовності і кооперативні діаграми для основного потоку подій варіанту використання Register for Courses.

Створення діаграми послідовності.

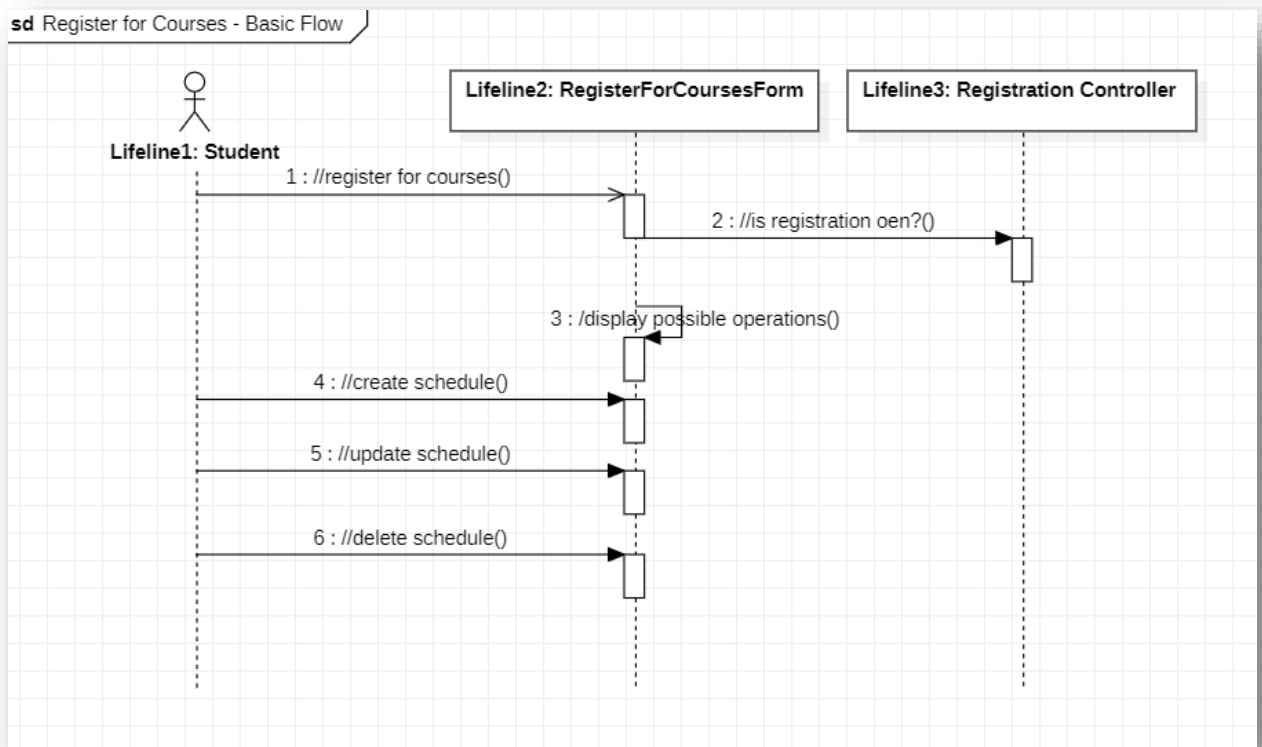
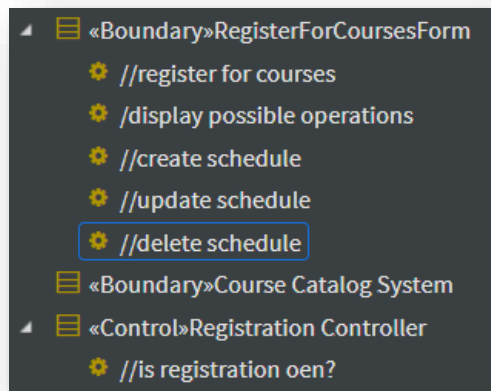
1. Натискаємо правою кнопкою миші на кооперації Register for Courses в пакеті Use-CaseRealization - Register for Courses.
2. В меню оберемо пункт *New/ Sequence Diagram* (Діаграма послідовності).
3. Назвемо нову діаграму Register for Courses - Basic Flow.
4. Двічі натискаємо на ній в браузері, щоб відкрити її вікно **Register for Courses - Basic Flow**.



Додавання на діаграму дійових осіб, об'єктів і повідомлень.

5. Перетягнемо дійову особу Студент з пакету  Use-CaseModel браузера на діаграму.
 6. Перенесемо класи RegisterForCoursesForm та RegistrationController з  Analysis Model на діаграму Register for Courses - Basic Flow. Кожна особа і кожен клас має на діаграмі пунктирну вертикаль, яка називається лінією життя.
 7. На панелі інструментів натискаємо кнопку  Object Message (Повідомлення об'єкта).
 8. Проведемо мишею від лінії життя дійової особи Студент до лінії життя об'єкта RegisterForCoursesForm.
 9. Виділивши повідомлення над стрілкою, введемо його ім'я: `//register for courses()`.
 10. Поміщаємо на діаграму інші п'ять повідомлень: `//is registration open?()`, `//display possible operations()`, `//create schedule()`, `//update schedule()`,
`//delete schedule`. Для рефлексивного повідомлення з використовується кнопка  Message to Self.
- ### **Співвідношення повідомлень з операціями і створення операцій**
11. Клацнемо правою кнопкою на тексті повідомленні 1: `//register for courses`.
 12. У меню, обираємо пункт `<new operation>`. З'явиться вікно Operation Specification специфікації операції.
 18. У полі імені залишаємо ім'я повідомлення - `// register for courses`. Це буде ім'ям операції.
 19. Натискаємо на кнопку ОК, щоб закрити вікно специфікації операції і повернутися на діаграму.

20. Повторюємо дії 16 - 19, поки не зіставите з операціями всі інші п'ять повідомлень: // is registration open? (), // display possible operations (), // create schedule (), // update schedule (), // delete schedule ().



Створення приміток.


Щоб помістити на діаграму примітка, виконайте наступне.

13. Натискаємо на панелі інструментів кнопку  Note (Зауваження).

14. Клацаємо мишею в тому місці діаграми, куди збираємося

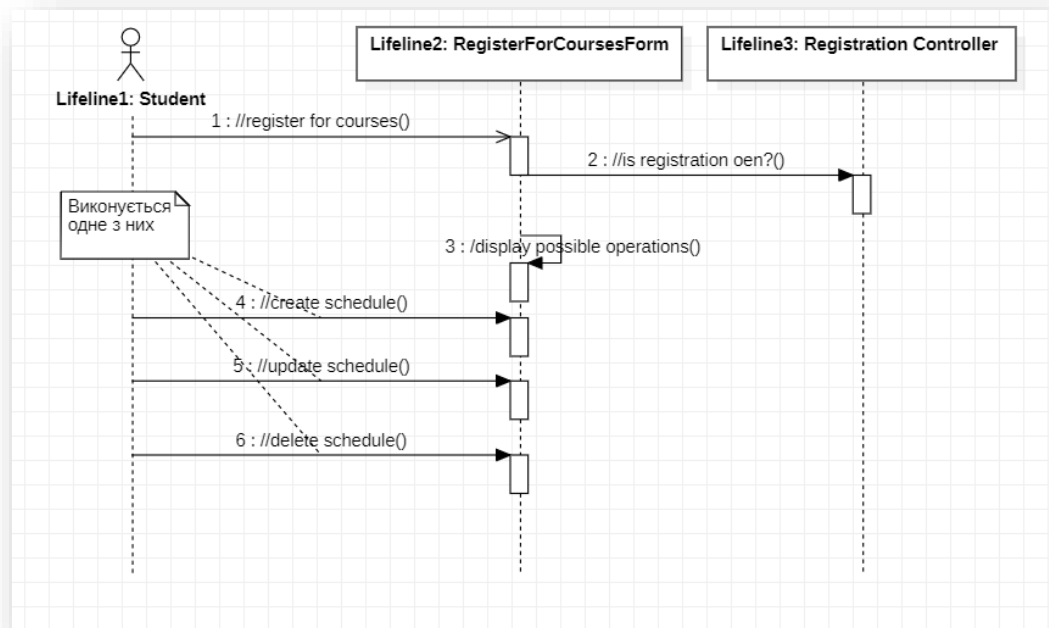
розмістити примітку.

15. Виділивши нову примітку вводимо туди текст.

16. Щоб прикріпити примітку до елементу діаграми, на панелі інструментів натискаємо кнопку  Anchor Notes To Item (Прикріпити примітку до елементу).

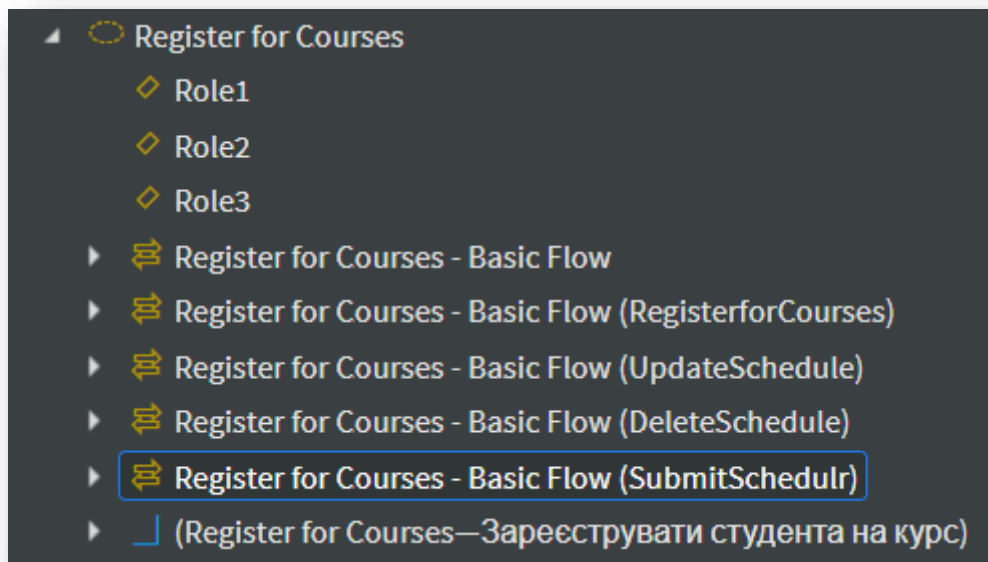
17. Натиснувши ліву кнопку миші, проведемо покажчик від примітки до елемента діаграми, з яким воно буде пов'язано. Між приміткою і елементом виникне штрихова лінія.

18. Прикріпимо примітку «Виконується одне з них» до трьох повідомлень від особи Студент.

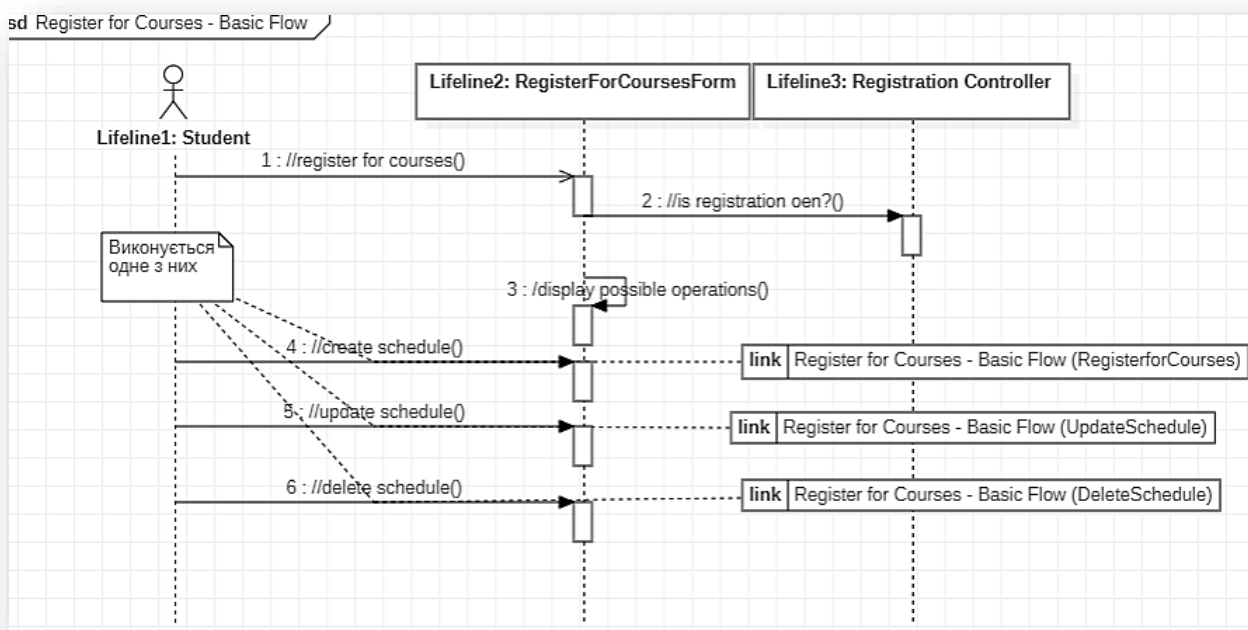


Створення посилань на діаграми

19. Відповідно до кроку «Створення діаграми послідовності» даної вправи, створимо в кооперації Register for Courses чотири пусті діаграми послідовності з іменами RegisterforCourses - BasicFlow (CreateSchedule), RegisterforCourses - BasicFlow (UpdateSchedule), RegisterforCourses - BasicFlow (DeleteSchedule), RegisterforCourses - BasicFlow (SubmitSchedule).





Отримана діаграма Register for Courses - Basic Flow:



20. Збережемо модель File / Save. Збережемо модель в своєму каталозі за допомогою пункту меню File / Save As під новим ім'ям Шевлюк3. Другий етап створення моделі залишиться в файлі Шевлюк2.

Вправа 3.2. Створення діаграм взаємодії для потоків додавання, зміни, видалення графіку


В кооперації Register for Courses пакетку  Use-Case Realization-Register for Courses виконаємо аналогічні дії: Додавання на діаграму дійових осіб, об'єктів і повідомлень, Співвідношення повідомлень з операціями, Створення приміток, Створення посилань на діаграми - для зміни діаграм послідовності Register for Courses - Basic Flow (Create Schedule), Register for Courses - Basic Flow (Update Schedule), Register for Courses - Basic Flow (Delete Schedule).


На діаграмі  Register for Courses - Basic Flow (Create Schedule) створимо повідомлення та операції з іменами: `//create schedule`, `//get course offering`, `//get course offering(forSemestr)`, `//display course offering`, `//display blank schedule`, `//select 4 primary and 2 alternate offerings`, `//create schedule with offerings`, `//create with offerings`, `//add schedule(Schedule)`.

Через праву кнопку миші можна пов'язати повідомлення з існуючою операцією. Можна створити нову операцію `<new operation>` і відразу зв'язати її з повідомленням. Після цього можна скорегувати повідомлення.


Крім приміток, на діаграму можна помістити також і текстову область. З її допомогою можна, наприклад, додати до діаграми заголовок.

Щоб помістити на діаграму текстову область, виконаємо наступне:

1. На панелі управління натискаємо кнопку  Text Box.
2. Натискаємо мишею всередині діаграми, щоб помістити туди текстову область.
3. Виділивши цю область, вводимо в неї текст, наприклад: «У цей момент виконується потік SubmitSchedule».

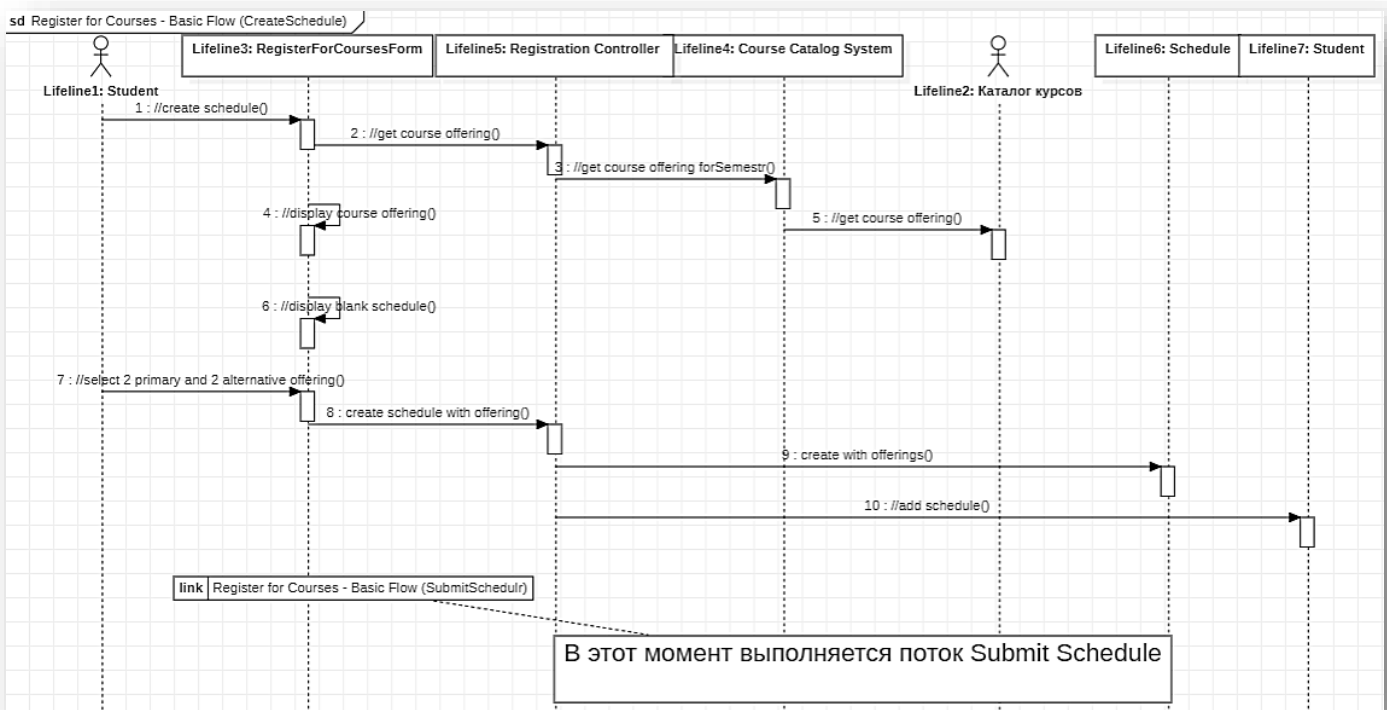
На діаграмі  RegisterforCourses - BasicFlow (Update Schedule) створюємо повідомлення та можливі операції з іменами: `//update schedule`, `//get current schedule(Student,forSemester)`, `//get schedule(forSemestr)`, `//display schedule(Schedule)`, `//get course offerings`, `//get course offering(forSemestr)`,

//display course offering, //update offering selections, //update schedule with new selections, //create with offerings.

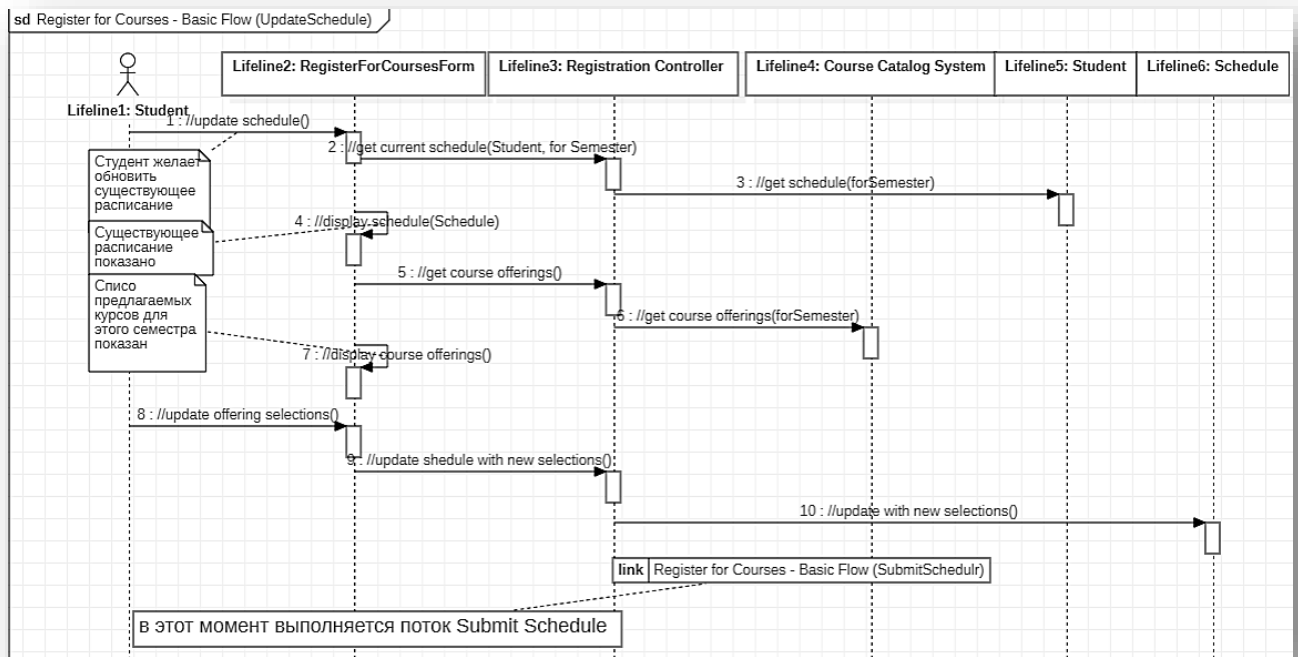
На діаграмі  RegisterforCourses - BasicFlow (DeleteSchedule) створюємо повідомлення та можливо операції з іменами: //delete schedule, //get current schedule, //get schedule(forSemestr), //display schedule(Schedule), //request schedule delete confirmation, //confirm schedule deletion, //delete current schedule, //delete schedule(forSemestr), //delete, //remove student(Schedule).

Для контролю та виправлення помилок подивимося список операцій для класу: права миша в браузері на класі, наприклад, RegisterForCoursesForm, меню *Open Specification*, вікно **Class Specification**, вкладка Operations.

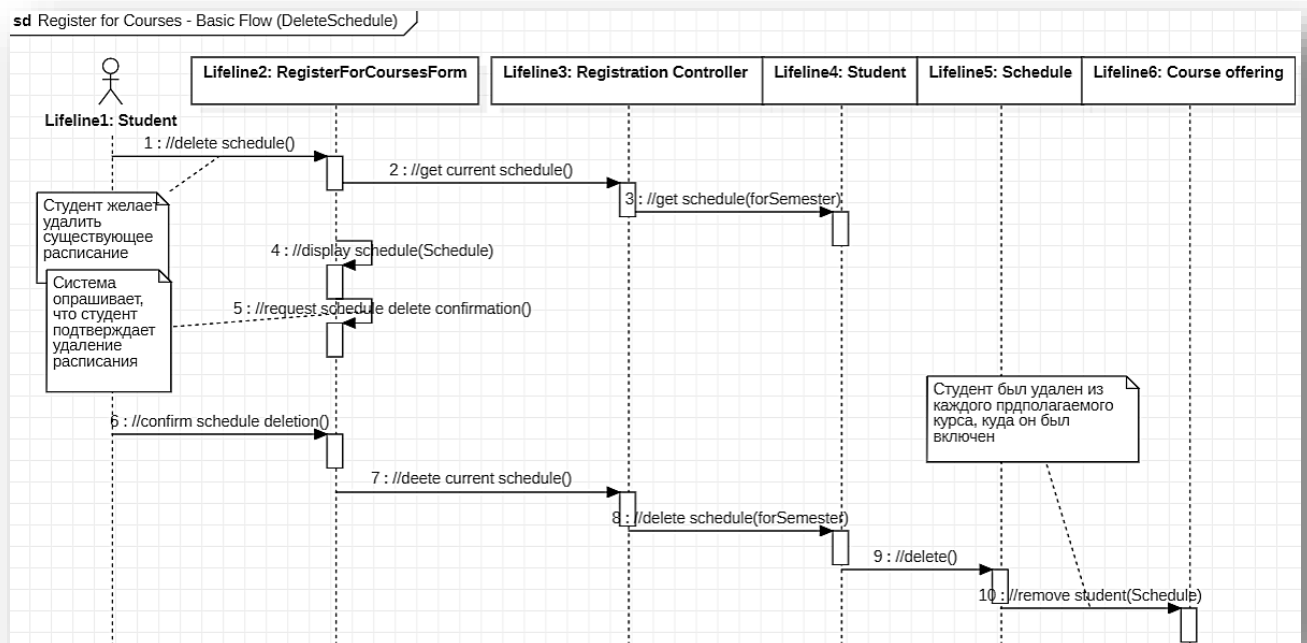
Помістимо текстові області на діаграмі. Зв'язуємо тексти з посиланнями на діаграмі.



1 Діаграма послідовності Register for Courses - Basic Flow (Create Schedule)



2 *Діаграма послідовності Register for Courses - Basic Flow (Update Schedule)*

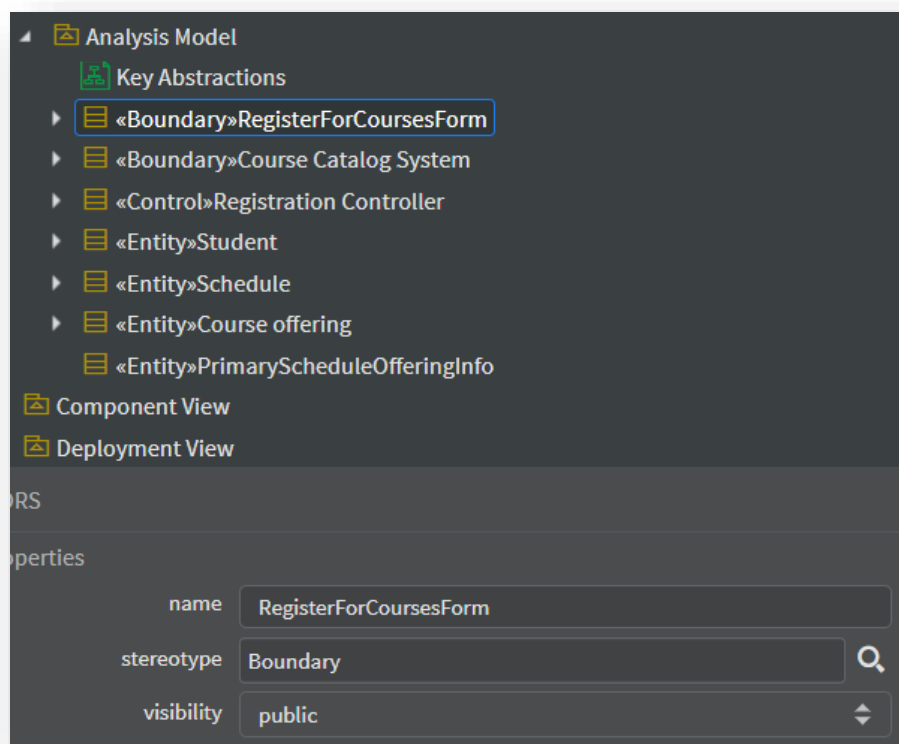


3 *Діаграма послідовності Register for Courses - Basic Flow (Delete Schedule)*

Зберігаємо модель File / Save. Збережемо модель в своєму каталозі за допомогою пункту меню File / Save As під новим ім'ям Шевлюк4. Третій етап створення моделі залишиться в файлі Шевлюк3.

Вправа 3.3 Створення діаграми взаємодії до потоку пропозиції графіка

1. В пакеті Analysis Model логічного представлення створюємо новий клас New Class.
2. Двічі натискаємо на нього мишею. Відкриється вікно **Class Specification**.
3. На вкладці General задаємо Name = PrimaryScheduleOfferingInfo, Stereotype= entity.
4. Натискаємо ОК.



5. В кооперації Register for Courses пакету Use-Case Realizations виконуємо дії (див. Вправа 1 цієї лабораторної роботи): Додавання на діаграму дійових осіб, об'єктів і повідомлень, Співвідношення повідомлень з операціями, Створення приміток - для зміни діаграми послідовності Register


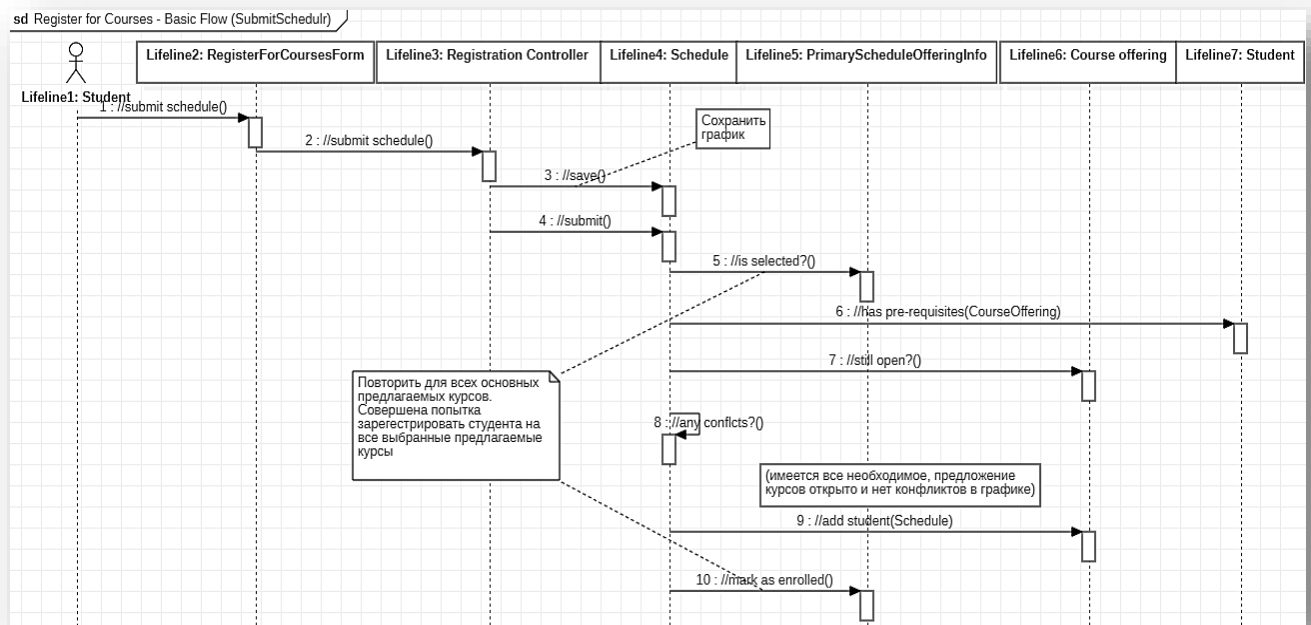
for Courses - Basic Flow (Submit Schedule). На діаграмі  RegisterforCourses - BasicFlow (SubmitSchedule) створимо повідомлення та можливо операції з іменами: //submit schedule, //save, //submit, //is selected?, //has pre-requisites(CourseOffering), //still open?, //any conflicts?, //add student (Schedule), //mark as enrolled in.

Рис. 3.8. Діаграма послідовності Register for Courses - Basic Flow (Submit Schedule)

6. Зберігаємо модель *File/ Save*.



Вправа 3.4. Створення діаграми кооперації

Для створення діаграми кооперації достатньо відкрити діаграму послідовності, наприклад, Register for Courses - Basic Flow і натиснути клавішу F5

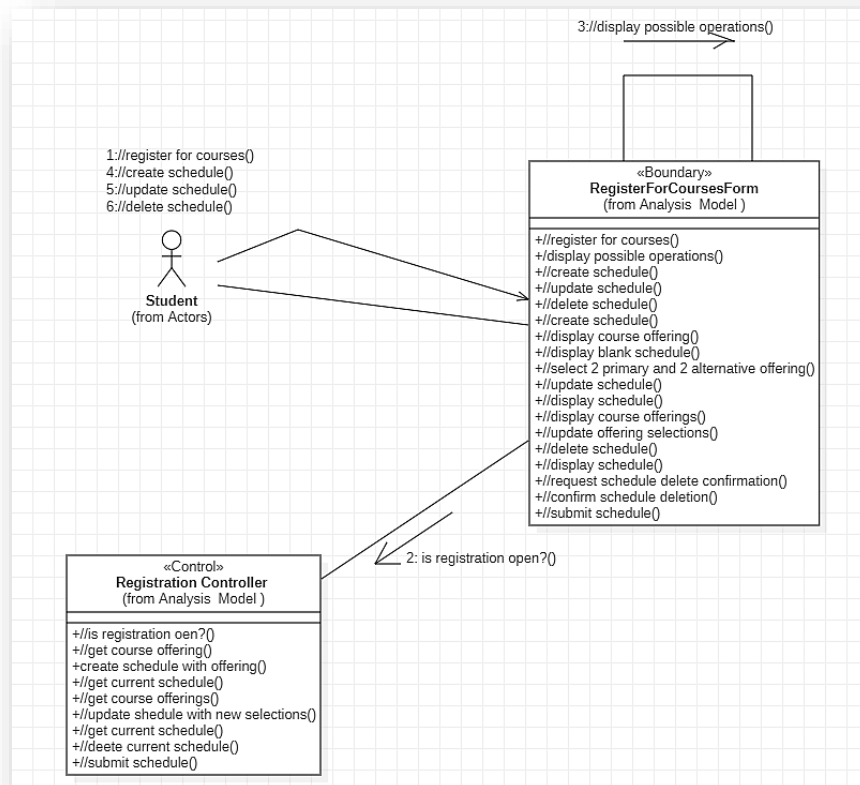
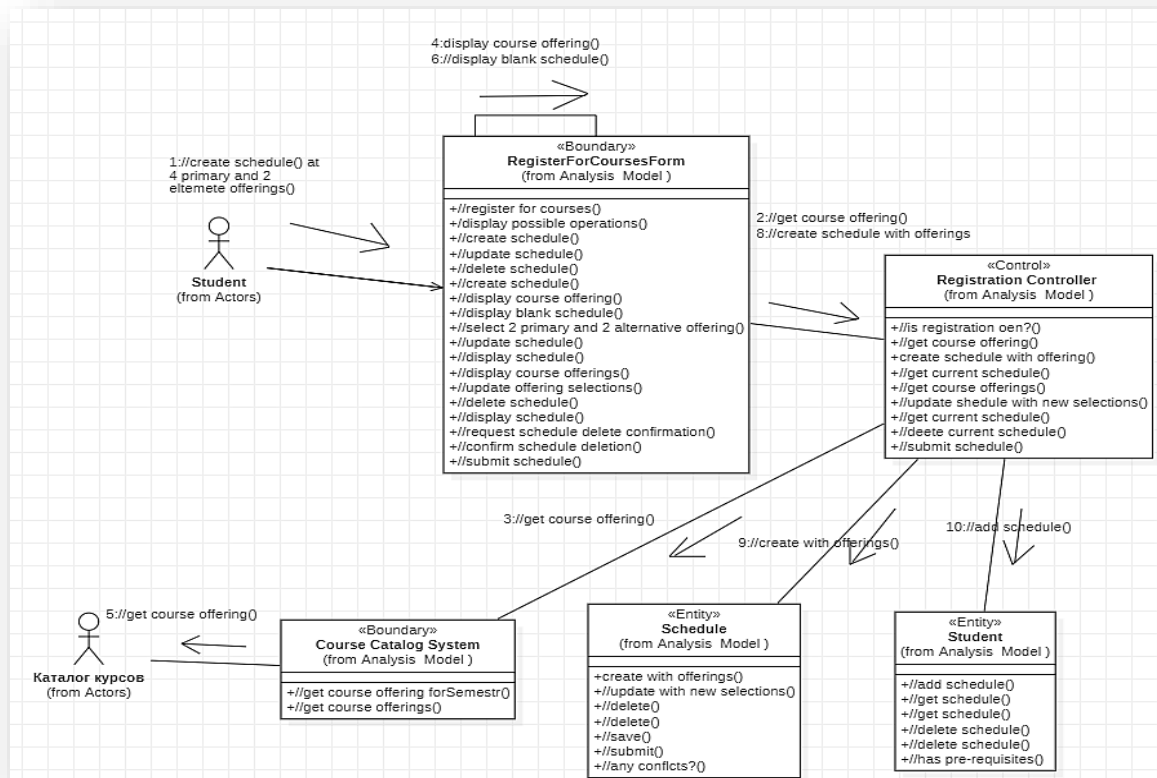
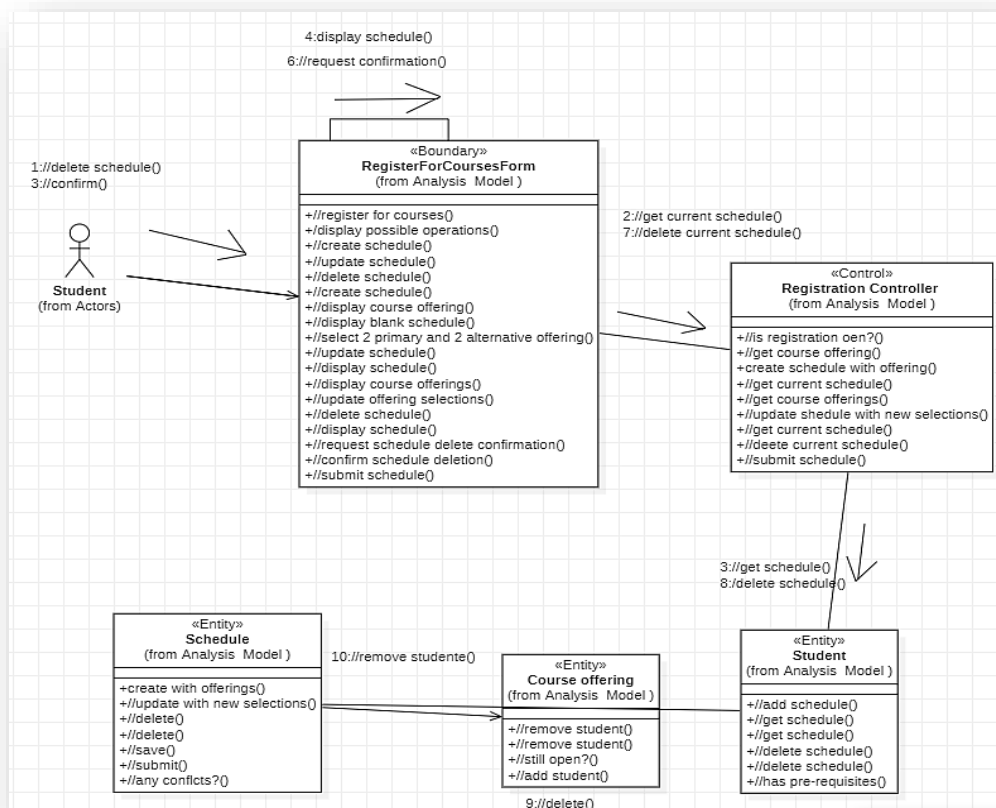


Рис. 3.9. Діаграма кооперації Register for Courses - Basic Flow

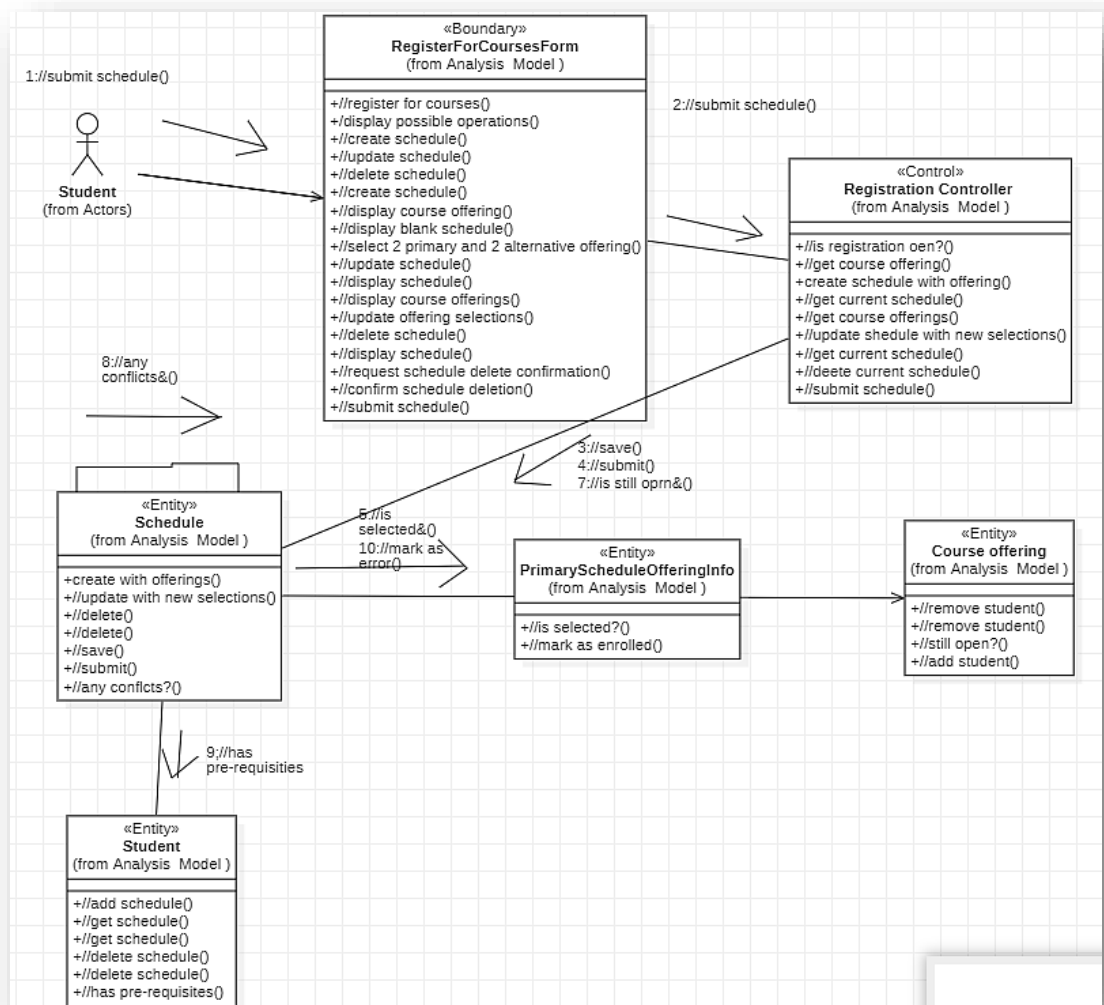
Створюємо ще чотири діаграми кооперації: Register for Courses - Basic Flow (Create Schedule), Register for Courses - Basic Flow (Update Schedule), Register for Courses - Basic Flow (Delete Schedule), Register for Courses - Basic Flow (Submit Schedule).



4 Register for Courses - Basic Flow (Update Schedule)



5 Register for Courses - Basic Flow (Submit Schedule).



6 Register for Courses - Basic Flow (Submit Schedule)

Висновок:

Під час виконання даної лабораторної роботи я навчилась створювати діаграми взаємодій та діаграми послідовностей. Діаграми взаємодії є моделями, що описують поведження взаємодіючих груп об'єктів. Як правило, діаграма взаємодії охоплює поведження тільки одного варіанта використання. На такій діаграмі відображається ряд об'єктів і ті повідомлення, якими вони обмінюються між собою в рамках даного варіанта використання. На діаграмі послідовності об'єкт зображується у вигляді прямокутника на вершині пунктирної вертикальної лінії. Ця вертикальна лінія називається лінією життя

об'єкта (lifeline). Вона являє собою фрагмент життєвого циклу об'єкта в процесі взаємодії. Кожне повідомлення представляється у вигляді стрілки між лініями життя двох об'єктів. Повідомлення з'являються в тому порядку, як вони показані на діаграмі (зверху вниз). Кожне повідомлення може бути позначено ім'ям, за бажанням можна вказати також аргументи й деяку керуючу інформацію. Також можна використовувати самоделегування - повідомлення, яке об'єкт посилає самому собі, при цьому стрілка повідомлення вказує на ту ж саму лінію життя.