Міністерство освіти і науки України
Тернопільський національний технічний університет
імені Івана ПУЛЮЯ

кафедра програмної інженерії

Лабораторний практикум

до дисципліни

«Моделювання та аналіз програмного забезпечення»

Тернопіль 2015

Лабораторний практикум до дисципліни «Моделювання та аналіз програмного забезпечення» // Уклад.: М.Р. Петрик, О.Ю.Петрик - Тернопіль: ТНТУ імені Івана Пулюя, 2015. 46с.

Призначений для допомоги студентам у виконанні лабораторних робіт з дисципліни “Моделювання та аналіз програмного забезпечення” і контролю знань та практичних навіків студентів щодо застосування методів моделювання програмного забезпечення та технологій побудови основних моделей аналізу прграмного забезпечення на різних стадіях проектування на основі універсальної мови моделювання UML. Розроблено з урахуванням модульної системи навчання, рекомендацій до самостійної роботи і індивідуальних завдань, широкого спектру тем предметних областей при виконанні лабораторних робіт.

Укладачі: М.Р. Петрик, О.Ю.Петрик
Відповідальний за випуск: М.Р.Петрик
Рецензент: С.А. Лупенко

Розглянуто на засіданні кафедри програмної інженерії, протокол №1 від 31.08.2015р.

Схвалено на засіданні методичної ради факультету комп’ютерно-інформаційних систем і програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя, протокол №1 від 31 серпня 2015 р

ЗМІСТ

Лабораторна робота №1. Вибір концептуальної моделі розробки ПЗ на основі аналізу вимог предметної області (згідно варіантів)

Теоретичні відомості

На сучасному етапі розвитку інформаційних систем і технологій особлива увага приділяється структурі та організації програмного забезпечення.

При розробці структури програмного забезпечення використовують об'єктно-орієнтований підхід. В основі об'єктно-орієнтованого підходу лежить об'єктна декомпозиція, тобто подання розроблюваного програмного забезпечення у вигляді сукупностей об'єктів, в процесі взаємодії яких, через передачу повідомлень, відбувається виконання необхідних функцій.

Специфікація розроблюваного програмного забезпечення об'єднує в собі наступні моделі:

1. Модель використання являє собою опис функціональності програмного забезпечення з точки зору користувача.
2. Концептуальна модель – модель, що описує основні абстракції предметної області, які забезпечують необхідну функціональність ПЗ та їх взаємодія;
3. Модель реалізації визначає реальну організацію програмних модулів і файлів.

Для побудови цих моделей був використовують уніфіковану мову моделювання (UML).

Завдання: відповідно до варіанту завдання, узгодженого з викладачем, знайти приклади 3-4 аналогічних програмних систем та проаналізувати принципи їх функціонування. На основі отриманих даних створити проект технічного завдання для розробки власної системи.

Звіт повинен складатися з двох частин.

Перша частина містить аналіз аналогічних систем. Для кожного аналогу ПЗ визначити наступні характеристики:

- назва;
- розробник (дистриб'ютор);
- архітектура (desktop application, client-server, 3tier web application);
- мова реалізації;
- перелік функцій, характеристик (не менше 5);
- аналіз переваг та недоліків даного ПЗ;
- джерело інформації (веб-сайт).

Друга частина визначає перелік характеристик системи, що проектується та розробляється студентом. Для програмної системи, що проектується визначити:

- призначення ПЗ;
- основні функції (не менше 5);
- користувачі системи (2-3 користувачі);
- опис структури (схема);
- сценарії роботи (не менше 3-х сценаріїв);
- засоби апаратної та програмної реалізації (платформа, ОС та мова програмування);
- output (вихідні дані): таблиці, звіти, графіки (не менше 3).

Приклад виконання. Предметна область «ДИСКОНТНІ СИСТЕМИ»

«Якщо клієнт зрозумів пропозицію, його не потрібно буде переконувати, він сам прийде і зробить покупку» Джон Паттерсон.

I.

Дисконтні системи – це системи, що створені для обслуговування закладів харчування, розважальних закладів чи навіть підприємств. Основною складовою таких систем є використання дисконтних карток.

Дисконтна картка (купон, ваучер, талон тощо) - це засіб, що гарантує отримання споживачем знижки в торгівельних точках продавця (-ів) або учасників дисконтного клубу, при дотриманні правил використання цих карток.

Дисконтні картки як правило випускають мережі магазинів, АЗС, автосалони, спортивні клуби, пральні, аптеки, клініки, авіакомпанії, салони краси, ресторани, клуби, інтернет-провайдери, автомайстерні, зоомагазини і т.п. для обслуговування власних програм лояльності знижок. Прикладом таких карток може бути звичайна кредитна картка («Приват Банк», «Аваль»), картка нарахування бонусів АЗС «WOG», «ОККО» або мережі супермаркетів «Сільпо». В залежності від мети створення дисконтної системи картки можуть використовуватись для оплати послуг чи товарів («Сільпо») або для накопичення бонусів («WOG»).

Заклад чи підприємство, що використовує дисконтну систему, вже по своїй суті вважається дисконтним клубом. Дисконтні клуби можуть бути відкритого і закритого типу.

Закритий дисконтний клуб відрізняється тим, що стати клієнтом або партнером дисконтного клубу можна тільки за рекомендацією та / або згоди інших клієнтів або партнерів клубу відповідно.

Відкритий дисконтний клуб пропонує партнерам та клієнтам клубу можливість участі в клубі без узгодження з іншими партнерами або клієнтами дисконтного клубу.

Розглянемо приклади закладів Тернополя, що використовують дисконтну систему:

1. Кожному з нас хоча би раз приходилось робити покупки в супермаркеті, для прикладу в «Сільпо». Ця мережа супермаркетів є потужною і масштабною і є в кожному куточку України. Для зручності оплати або для нарахування бонусів (у вигляді знижок для постійних клієнтів) ці заклади використовують дисконтні системи, а якщо точніше то двох типів одночасно.

Закритий тип системи проявляється в тому, що кожний працівник супермаркету має свою пластикову (дисконтну) картку, якою він засвідчує свою присутність на робочому місці, отримує на неї заробітну плату, розраховується нею за товари і отримує бонуси. Кожна картка реєструється в базі даних і є «прив'язаною» до робітника. Якщо робітник звільняється то з бази даних видаляється його персональні дані і картка стає не дійсною.

Клієнту який розплачується за велику суму куплених товарів можуть вручити дисконтну картку клієнта, яку він може використовувати при наступних покупках в маркеті і отримувати бонуси або застосувати при оплаті для отримання певного відсотка знижки, що накопичився на картці. Це вже є проявом відкритого дисконтного клубу. Картки чи купони, що видаються клієнтам суттєво відрізняються від карток робочих своїми функціональними можливостями, тобто клієнт не може засвідчити себе на робочому місці, тому що спеціальний апарат розрізняє картки, клієнт навіть не зможе добратись до спеціального апарата – йому будуть заважати службові двері, замок яких також розрізняє типи карток.

2. Розважальний заклад «Concert Hall HOLIDAY» є представником відкритого клубу, використовує дисконтні картки, що зовнішньо схожі на звичайні кредитні картки, але із зображення логотипу нічного клубу. Клієнти, які відвідують даний заклад повинні мати таку дисконтну картку для оплати послуги, наприклад за коктейль. Такий спосіб оплати є дуже зручним і швидким, тому що клієнт не витрачає час на рахування грошей і не відчуває дискомфорт при оплаті «дрібними» грішми. І в свою чергу бармен чи офіціант не витрачає багато часу на перераховування грошей і видачі здачі, а просто проводить через прилад, який зчитує дані і автоматично знімає потрібні кошти з картки. Єдиною незручністю може бути простоювання в черзі для отримання чи поповнення такої картки.

3. Закритим дискретним клубом є розважальний заклад є Arena Hall (колишній «Марцепан») в якому робочий персонал (подібно «Сільпо») засвідчує свою присутність, може розплачуватись нею, але особливістю є те, що офіціант чи бармен за допомогою цієї картки може відкрити касовий апарат для внесення коштів чи видачі решти. При цьому касовий апарат містить вмонтований сенсорний екран, на якому встановлене програмне забезпечення і за допомогою якого офіціант робить замовлення і автомат видає квитанцію.

4. Але найпоширенішими і найуживанішими дисконтами є кредитні картки банків. Сфера їх використання є практично необмеженою і функціонал їх є багатшим порівняно з вище вказаним. На малюнку зображено кредитку Приват Банку «універсальна».

Для її отримання потрібно дати свої персональні дані в відділені банку, після чого вона стає прив'язаною до вашого мобільного номеру, паспорту, і E-mail. Нею можна розраховуватись за покупки як і онлайн (наприклад в інтернет-магазинах чи поповнення рахунку онлайн) так і в спеціальних закладах (супермаркетах, магазинах). Особливістю «універсальної» кредитки є те, що в неї можливість нараховувати бонуси, що є відсотками цін оплачених карткою товарів. Ці бонуси складають окремий рахунок.

Система обслуговування кафе за допомогою дисконтних карт персоналу (подібно опису ArenaHall) є подібна програмному забезпеченню для встановлення на звичайний стаціонарний комп'ютер. Комп'ютер може бути один, а при збільшенні кількості можуть бути об'єднані в звичайну локальну мережу. Таку систему можна реалізувати на мові програмування, яка б підтримувала зв'язок з базою даних.

II.

Програмне забезпечення призначене для обслуговуванням персоналом, а точніше – для заміни замовлень «на папері» в зручну, швидку і надійну систему створення, зберігання історії замовлень, підсумовування прибутку.

Функціоналом такої програми є:

- *Вказати столик (який буде обслуговуватись).*
- *Додати замовлення.*
- *Редагувати замовлення.*
- *Сплатити замовлення.*
- *Роздрукувати чек замовлення.*
- *Змінити користувача (офіціант проводить своєю карткою і замовлення буде записано на його обслуговування).*
- *Вихід.*

Користувачами системи можуть бути власники дисконтних карток на яких записані дані персони-користувача. Оскільки кафе є замкнутого дисконтного типу, тому картки доступу можуть мати тільки люди представники персоналу і/або менеджер. В закладі можуть бути 2-3 офіціанти і один бармен – вони і будуть головними користувачами системи.

Оскільки програмне забезпечення є маленьким (функціонал малий) то структура його є теж простою – лінійною. Сценарієм роботи ПЗ:

1. *Офіціант підходить до комп'ютера, і проводить карткою по спеціальному апараті.*
2. *З картки зчитуються дані про користувача і відкривається головне вікно програми на якому зображено розміщення столиків закладу. Користувач вибирає столик на який буде зроблено замовлення.*
3. *Після обрання столика на екрані зображуються загальні критерії напоїв і страв з меню. Офіціант набирає замовлення після чого роздруковується чек*
4. *Офіціант повертається до клієнтів і дає їм чек. Клієнт передає офіціанту вказану у чеку суму.*
5. *Офіціант повертається до апарату, вибирає столик і вказує «сплатити замовлення», вносить кошти в касу*
6. *Після внесення коштів електронна версія чеку зберігається в пам'яті.*

Дане ПЗ можна реалізувати мовою програмування C++ або/і Java з підтримкою бази даних з СУБД. Підтримуватись буде на звичайних комп'ютерах з операційними системами Windows: XP, Vista, 7. Вихідними даними даної програми будуть чек (текстові файли), або електронні таблиці історії замовлень. Також може складатись статистика прибутків.

Лабораторна робота № 2. Аналіз сутностей предметної області (словника імен та дій) для побудови моделей архітектури та варіантів використання створюваного ПЗ

Теоретичні відомості

Usecase - це текстовий опис сукупності сценаріїв, що виконуються користувачем при роботі з системою для досягнення певної мети.

Сценарій - послідовність дій при взаємодії користувача із системою для виконання певної операції.

Наприклад, купівля товарів у супермаркеті з використанням кредитної картки і невдала спроба купівлі через перевищення кредитного ліміту - це сценарії, а купівля товарів у супермаркеті - usecase.

Приклад 1. Зняття готівки в банкоматі

Користувач підходить до банкомату та вставляє свою картку. Система перевіряє картку та просить ввести пін-код. Після перевірки пін-коду система виводить головне меню, де користувач обирає опцію видачі готівки. Він визначає суму грошей та підтверджує виконання операції. Банкомат видає готівку та чек. Система видає запит про здійснення наступної операції. Користувач обирає опцію «не виконувати», забирає картку та йде.

Приклад 2. Замовлення літератури в бібліотеці

Головний сценарій (успішний):

Користувач підходить до бібліотекаря і надає йому замовлення (перелік видань). Бібліотекар заносить номер читацького квитка до системи і перевіряє облікову картку користувача. Бібліотекар шукає кожне видання в базі і визначає кількість екземплярів вільних у даний момент часу. Система видає шифри (коди), за якими бібліотекар знаходить видання у сховищі, заносить номери книг до картки користувача і видає йому літературу. При цьому читацький квиток лишається у бібліотекаря.

Альтернативні сценарії:

1. Користувач бібліотеки є боржником, бібліотекар не може видати йому літературу.
2. Бібліотекар не може видати користувачу літературу, оскільки в даний момент часу немає вільних екземплярів у сховищі.
3. Читацький квиток користувача є недійсним. Бібліотекар вилучає його.
4. Технічний збій роботи системи. Видається повідомлення «немає зв'язку з сервером баз даних». Бібліотекар викликає адміністратора системи.

Написання usecases дозволяє чітко визначити хто є користувачем системи, які її сценарії роботи, що є метою використання системи. Usecases - це функціональні та поведінкові вимоги до системи, які показують, що саме вона має робити.

Існують три форми написання usecases:

1. Коротка - короткий опис в один абзац одного зі сценаріїв (зазвичай успішного) роботи системи (приклад 1). Виконуються під час початкового аналізу вимог до системи.
2. Поверхнева - поверхневий опис у вільній формі усіх сценаріїв (головного і альтернативних) одного з usecases (приклад 2). Виконуються під час початкового аналізу вимог до системи.
3. Повна - всі кроки і дії детально описані, включаючи перед- та постумови виконання usecase. Виконуються на стадії відбору з повного переліку usecases у короткій та поверхневій формах невеликої частини важливих (критичних для роботи системи) usecases.

Повна форма опису usecase має перелік розділів, коротко описаних у таблиці 1. Нижче наведено приклад складання usecase у повній формі для системи продажу товарів у супермаркеті.

Параметри опису usecase

Usecase section	Comment

Use Case Name	Назва usecase (починається з дієслова)
Scope	System or Business
Level	User-goal or Sub-function
Primary Actor	Головний актор
Stakeholders and interests	Перелік осіб, які зацікавлені в виконанні даного usecase та мета, яку вони при цьому переслідують
Preconditions	Список передумов, які повинні виконуватись для початку виконання usecase
Success guarantee	Список умов, при виконанні яких можна говорити про успішне закінчення виконання usecase
Main Success Scenario	Головний успішний сценарій usecase. Найчастіше є безумовним
Extensions	Альтернативні сценарії успішного чи неуспішного закінчення usecase
Special Requirements	Спеціальні (нефункціональні) вимоги
Technology and Data Variations List	Поради для реалізації певних кроків usecase
Frequency of Occurrence	Частота виконання usecase при користуванні системою (у відсотках)
Miscellaneous	Додаткові вимоги чи відкриті питання

Приклад 3. Купівля товарів у супермаркеті.

Scope: система продажу товарів в супермаркеті (System)

Level: User-goal

Primary Actor: касир

Stakeholders and interests:

1. Касир: зацікавлений у точному швидкому вводі інформації про товари та відсутності помилок вводу, які призведуть до штрафу (зменшення зарплатні касира)
2. Покупець: зацікавлений у швидкому придбанні товарів (отриманні послуг) та зручному відображенні розрахунків суми покупки.
3. Компанія (продавець): зацікавлена в точній обробці транзакцій при покупці товарів та задоволенні інтересів користувача.
4. Менеджер: зацікавлений у швидкому розв'язанні проблем під час повернення товарів та легкій перевірці операцій, що здійснюють касири.
5. Державна податкова адміністрація: зацікавлена в отриманні податків від кожного продажу товарів.

Preconditions: касир виконав вхід до системи (авторизація)

Main Success Scenario:

1. Покупець підходить до каси з сукупністю товарів (послуг), які він хоче придбати.
2. Касир стартує нову продаж.
3. Касир вводить послідовно всі товари.
4. Система опрацьовує код кожного товару та виводить його назву, кількість та суму покупки для кожного товару.
5. Система розраховує загальну суму покупки та суму нарахованих податків.
6. Касир озвучує суму покупцеві та просить розрахуватися.
7. Покупець розраховується і система опрацьовує оплату.
8. Система запам'ятовує суму покупки до бази даних.
9. Система друкує чек.
10. Користувач відходить від каси з чеком і товарами.

Extensions:

a) у будь-який час менеджер виконує специфічну операцію:

1) менеджер вводить свій код авторизації;
2) менеджер виконує специфічну операцію (наприклад, перевіряє баланс даної каси);
3) менеджер виходить з системи, і вона повертається до режиму роботи з касиром.

b) у будь-який час виникає фатальна помилка в системі:

1) касир перевантажує систему, входить до неї та вводить запит про повернення до попереднього стану;

2) система відновлює попередній стан.

2а) система не може відновити попередній стан:

- система видає помилку на екран та зберігає її у лог;
- касир починає новий продаж.

1а) покупець чи менеджер хоче повернутися до відкладеного продажу:

1) касир виконує операцію повернення та вводить код продажу;

2) система відображає відтворений продаж. 2а) продаж не було знайдено:

- система виводить помилку на екран;
- касир розпочинає новий продаж і вводить дані про всі товари з початку.

2-4а) покупець говорить касиру, що у нього є картка для отримання знижки:

1) касир вводить номер картки до системи;

2) система запам'ятовує код картки та вид знижки для розрахунку загальної суми продажу.

3а) код товару не знайдено в базі даних:

1) система виводить помилку на екран;

2) касир опрацьовує помилку:

2а) додатковий код товару знаходиться на ньому і касир може його прочитати:

- касир вводить код вручну;
- система опрацьовує код та виводить назву товару та ціну. 2b) касир виконує usecase «Знайти код невідомого товару»

для ідентифікації товару. 3b) у покупця є декілька однакових товарів:

1) касир може ввести код товару та його кількість вручну.

3-6а) користувач просить видалити один з товарів із поточного продажу:

1) касир вводить номер товару для видалення;

2) система видаляє строку товару з продажу і перераховує загальну суму.

3-6b) покупець просить касира відмінити продаж у цілому:

1) касир відміняє продаж.

3-6с) касир відкладає продаж:

1) система запам'ятовує продаж, яка може бути відновлена в будь-який час;

2) система видає спеціальний чек із переліком товарів та кодом продажу, який дозволить продовжити продаж.

4b) Покупець вважає, що товар є зіпсованим і пропонує купити його за нижчою ціною:

1) касир робить запит у менеджера;

2) менеджер дозволяє виконати продаж за нижчою ціною;

3) касир вводить вручну ціну, що є нижчою за попередню;

4) система відображає нову ціну для даного товару в поточному продажі.

5 а) користувач говорить, що він повинен отримати знижку:

1) касир робить запит щодо знижки;

2) касир вводить ідентифікаційний код покупця;

3) система розраховує сумарну знижку на продаж. 6а) покупець говорить касиру, що він хотів розплатитися готівкою,

але не має достатньо грошей:

1) касир просить обрати інший метод сплати за товари:

2) 1 а) покупець просить відмінити продаж. 7а) оплата готівкою:

1) касир вводить суму грошей, яку дав покупець;

2) система розраховує суму решти і відкриває касу;

3) касир видає решту покупцю;

4) система вносить випадок оплати до бази даних.

7b) оплата кредитною карткою:

1) покупець вводить інформацію про кредитну картку;

2) система показує суму оплати;

3) касир підтверджує суму оплати;

4) система відсилає авторизаційний запит до зовнішньої Системи Оплати Товарів та послуг та робить запит на підтвердження оплати;

5) система отримує підтвердження оплати, виводить інформацію для касира та відкриває касу;

6) система зберігає даний вид оплати за товари у базі даних;

7) касир просить покупця зробити підпис на чеку для підтвердження оплати. Покупець ставить підпис;

8) касир складає чек до каси та замикає її.

7c) касир робить відміну оплати:

1) система повертається до режиму вводу товарів. 9a) покупець просить видати подарунковий чек (без суми продажу):

1) касир робить запит про подарунковий чек. Система друкує подарунковий чек. 9b) в принтері закінчився папір:

1) система дає сигнал про закінчення паперу;

2) касир замінює папір;

3) касир робить запит про повторення друку чеку. *Special Requirements:*

1) великий плоский монітор, що має функції сенсорного екрану. Текст повинно бути гарно видно з відстані в 1 м;

2) система авторизації запитів про оплату кредитною карткою повинна видавати результат за 30 секунд у 90 відсотків випадків;

3) система перекладу тексту на декілька мов;

4) додаткові бізнес-правила можуть бути додані до пунктів 3-7.

Technology and Data Variations List:

1) менеджер виконує авторизацію під час скасування певної операції за допомогою зчитування номера картки менеджера кард-рідером чи введенням коду менеджера з клавіатури;

2) код товару вводиться сканером штрих кодів товарів чи з клавіатури;

3) система може використовувати декілька схем кодування товарів (UTC, EAN, JAN, SKU);

4) інформація про кредитну картку вводиться за допомогою кард-рідера чи з клавіатури;

5) підпис покупця для підтвердження кредитної операції робиться на товарному чеці.

Frequency of Occurrence: 95 %. *Miscellaneous (Open Issues):*

1) провести аналіз різних варіантів сплати податків;

2) вивчити можливість відновлення роботи системи після збою;

3) яка додаткова функціональність потрібна для різних прикладних галузей?

4) чи повинен касир забирати гроші з каси, коли він виходить із системи (закінчує зміну)?

5) чи може покупець використовувати сканер для ідентифікації товарів під час покупки, чи це повинен робити касир?

Даний приклад є досить детальним прикладом опису usecase в повній формі, але є цілком реальним (розроблений під час ОО аналізу вимог для системи продажу товарів NextGen POS, яка реалізована мовою Java).

Завдання. Відповідно до обраної теми (варіанта) на основі проведеного аналізу прикладної галузі (лабораторна робота № 1) розробити три різні usecases (по одному в короткій, поверхневій та повній формах відповідно) для своєї системи. Повна форма опису має містити всі пункти наведені в таблиці 1. Головний успішний сценарій повинен мати не менше 10 кроків. Передбачити не менше 5 альтернативних сценаріїв.

Приклад виконання

Вибрана предметна область: «Інформаційна система підтримки роботи веломайстерні». Дана предметна область має такі основні варіанти використання:

Таблиця 1. Варіанти використання

Основний актор	*Найменування*	*Формулювання*
Менеджер організації	*Реєстрація(прийом) замовлення*	*Дозволяє менеджеру організації приймати нові замовлення від клієнтів та передавати їх у виробництво.*
Менеджер організації	*Реєстрація термінового замовлення*	*Менеджер організації має можливість передавати у виробництво нові замовлення, які необхідно виконати в найкоротші строки.*
Менеджер	*Корегування*	*Менеджер організації може змінювати інформацію*

організації	*замовлення*	*про замовлення і вносити корективи.*
Менеджер організації	*Видалення замовлення.*	*При необхідності зняти замовлення з виробництва або при завершенні виконання менеджер організації має можливість функцію «Видалення замовлення».*
Менеджер організації	*Обробка запиту щодо виконання та змісту замовлення*	*Використовується менеджером організації для пошуку потрібної інформації про стан замовлення у виробництві, відповідного клієнта та іншої існуючої інформації. А також для надання довідкової інформації.*
Менеджер організації	*Передача замовлення на стадію виконання*	*Менеджер організації розміщує замовлення в плані в кінець черги виконуваних завдань*
Начальник виробничого відділу	*Перегляд інформації про нові та уже залучені, завершені замовлення*	*Можливість перегляду планів та замовлень. Перегляд інформації про замовлення, що уже виконуються чи уже завершені.*
Начальник виробничого відділу	*Створення та коректування черг замовлень*	*Можливість розміщувати замовлення у чергу обслуговування, яка базується на термінах виконання та надходження замовлень; черга замовлень зміщується.*
Начальник виробничого відділу	*Передача замовлення на заміну виконаному*	*Начальник виробничого відділу приймає виконані замовлення від майстрів з ремонту та обслуговування, та надає їм замовлення для виконання. Слідкує за завантаженістю виробництва та надходженням замовлень на лінію обслуговування .*
Начальник виробничого відділу	*Призначення виконавців*	*Начальник виробничого відділу призначає виконавців роботи (персонал майстерні) із поточної черги замовлень.*
Майстер з ремонту та обслуговування	*Перегляд інформації про нові та уже залучені, завершені замовлення*	*Можливість перегляду планів та замовлень. Перегляд інформації про замовлення, що уже виконуються чи уже завершені.*
Майстер з ремонту та обслуговування	*Фіксація результатів виконання*	*Майстер цеху фіксує результати виконання роботи(не виконання).*

Відповідно до вибраної предметної області, було вибрано 3 основні варіанти використання, та, у відповідності до завдання, складено схему сценаріїв UseCases.

I. Повна форма написання:

Опис сценарію варіанту використання «Реєстарція замовлення»

Scope: система підтримки роботи веломайстерні (System)

Level: User-goal

Primary Actor: менеджер-консультант.

Stakeholders and interests:

6. Менеджер-консультант: зацікавлений у точному швидкому вводі інформації про замовлення та відсутності помилок вводу, які призведуть до помилок та неточностей обслуговування, невдоволення клієнтів.

7. Замовник-клієнт: зацікавлений у швидкій реєстрації товарів(економія часу) та зручному відображенні результатів формування звіту замовлення.

8. Компанія: зацікавлена в точності надання послуг, веденні електронної звітності та зацікавлення клієнтів.

9. Начальник виробничого відділу: зацікавлений у швидкому розв'язанні проблем під час прийому-повернення замовлення з обслуговування та легкій перевірці операцій, що здійснюють виконавці замовлення.

Preconditions: Менеджер-консультант виконав вхід до системи (авторизація).

Клієнт знає свої можливості та мету візиту.

Main Success Scenario:

1) Клієнт-замовник звертається до Менеджера-консультанта з замовленням на обслуговування велосипеда або окремої деталі.

2) Менеджер звертається до Бази Даних ведення замовлень. Extensions1: відсутній доступ до бази даних.

3) Менеджер вводить дані про замовлення: назва деталі/велосипеда, серійні номера, мета обслуговування, дата надходження та приблизний час виконання замовлення, контактні дані. Extensions2: відсутні критичні дані для прийому замовлення.

4) Менеджер-консультант вибирає зі списку можливих послуг для надання обслуговування те завдання, яке необхідне для вирішення проблеми клієнта. Extensions3: відсутня послуга для надання в списку.

5) Система автоматично оцінює дані щодо замовлення та видає сукупну вартість обслуговування.

6) Менеджер-консультант озвучує вартість обслуговування клієнту і просить розрахуватися за передоплату.

7) Клієнт розраховується за передоплату. Extensions4: Клієнт не може розрахуватися.

8) Менеджер вносить оплату в систему та приймає замовлення(товар/велосипед/деталь).

9) Система приймає оплату за замовлення та формує звіт замовлення. Extensions5: Система не може сформувати звіт.

10) Менеджер-консультант просить підписатися на відривному бланку звіту клієнта.

11) Клієнт підписується на відривному бланку звіту.

12) Менеджер-консультант віддає відривний бланк звіту клієнту у якості документа прийому на обслуговування.

13) Клієнт завершує дію реєстрації замовлення і відходить від менеджера-консультанта.

14) Менеджер-консультант закриває Базу Даних замовлень. Extensions6: База даних видає помилку завершення з`єднання.

Extensions:

а) У будь-який момент виникає фатальна помилка системи:

3) Менеджер перевантажує систему, входить до неї та вводить запит про повернення до попереднього стану;

4) система відновлює попередній стан.

2а) система не може відновити попередній стан:

- *система видає помилку на екран та зберігає її у лог;*
- *Менеджер створює новий запис.*

Extensions1: відсутній доступ до бази даних.

1) Менеджер перезавантажує Базу Даних.

2) База даних відновлює доступ.

а) База даних не може відновити сенс.

- Менеджер перезавантажує систему.

- система відновлює початкові значення сеансу.

- Менеджер звертається до резервної Бази даних.

Extensions2: відсутні критичні дані для прийому замовлення.

1) Менеджер вводить усі можливі дані для ідентифікації замовлення.

2) Менеджер приймає замовлення з існуючими даними.

а) Менеджер не можу ідентифікувати дані для замовлення і звертається за допомогою до начальника виробничого відділу.

б) Менеджер не приймає замовлення через відсутність необхідних даних.

Extensions3: відсутня послуга для надання в списку.

1) Менеджер вводить консультується з начальником виробничого відділу та вводить нову послугу.

2) Менеджер відмовляє в обслуговуванні.

Extensions4: Клієнт не може розрахуватися.

1) Клієнт просить зберегти замовлення в базі до найближчого часу, коли буде змога розрахуватися.

- Менеджер приймає замовлення та зберігає в базі.

2) Клієнт не розраховується, а бажає розрахуватися після завершення обслуговування.

- Менеджер приймає замовлення без внесення оплати та видає відповідний бланк.

3) Менеджер відмовляє в ремонті без передоплати.

a. Клієнт бажає розрахуватися

b. Клієнт відмовляється від замовлення

c. Менеджер скасовує замовлення

Extensions5: Система не може сформувати звіт.

1) Менеджер повторно реєструє замовлення та формує звіт.

2) Менеджер вручу формує звіт та видає клієнтові.

3) Менеджер не формує звіт та система видає помилку про неможливість сформувати звіт. Повідомляє клієнта про відсутність звітного документа.

a. В принтері закінчився звіт. Менеджер використовує інший принтер/ формує звіт вручну.

4) Менеджер вносить поправки в дані про замовлення та клієнт видає звіт.

Extensions6: База даних видає помилку завершення з`єднання.

1) Менеджер зберігає зміни в БД та виконує аварійне закриття БД.

2) Менеджер не закриває БД до завершення сеансу.

3) Менеджер не можу зберегти дані та закрити сеанс БД:

a. Відбувається аварійне закриття без збереження даних. При цьому менеджер вносить дані заново в БД.

b. Виникає помилка. Менеджер корегує дані для коректного завершення роботи з БД.

c. Менеджер перезавантажує систему.

Special Requirements:

1) Система для роботи з електронними носіями та роботою з Базами Даних з відповідними характеристиками для швидкого обслуговування.

2) Монітор користувача для зручного інтерпретування даних.

3) Система формує звіт за 5-10 секунд.

4) Додатковий бізнес-документ.

Technology and Data Variations List:

- *Клієнт робить підпис на відривному бланку звіту.*
- *Клієнт та менеджер засвідчують поточний стан об`єкта замовлення.*

Frequency of Occurrence: 97 %.

Miscellaneous (Open Issues):

1) Провести технічну оцінку об`єкта замовлення та зовнішній вигляд.

2) Провести аналіз можливостей оплати замовлення.

3) Чи повинен консультуватися менеджер при прийомі замовлення з начальником виробничого відділу?

4) Чи потрібно клієнтові вносити передоплату?

5) Чи є можливість відмови від виконання обслуговування після оформлення замовлення?

6) Чи надаються певні гарантії якості та надійності роботи?

II. Поверхнева форма аналізу:

Відповідно до даної форми аналізу, відбувається поверхневий опис головного сценарію UseCases та основних виключень з сценарію (головного і альтернативних).

Для поверхневої форми аналізу було використано варіант використання «Корегування замовлення»

Опис сценарію варіанту використання «Корегування замовлення»:

1). Головний сценарій(успішний):

Клієнт, який звернувся за обслуговуванням, звертається до менеджера-консультанта з метою зміни в замовленні або ж корегування даних(наприклад вибір додаткових послуг або розширення списку послуг для даного обслуговування). Менеджер звертається до системи та перевіряє наявність даного замовлення та основні атрибути(дані про клієнта, поточний стан

замовлення, опис). Менеджер запитує в клієнта, яким чином скорегувати замовлення. Клієнт надає нові дані для замовлення. Менеджер змінює запис в базі даних та визначає нову оплату або ж доплату до існуючого замовлення(за необхідності). Клієнт розраховується за деталі/послуги. Менеджер роздруковує накладну на обслуговування і передає клієнту. Клієнт покидає майстерню.

2) Альтернативні сценарії:

1. Замовлення уже на стадії виконання і змінити в ньому на даний час нічого не можна. Менеджер звертається до клієнта з можливістю створення нового замовлення на інший термін.

2. Надання послуг неможливе технічно(майстерня не надає подібних послуг). Менеджер не може надати дозвіл на обслуговування.

3. Технічний збій роботи системи. Менеджер не може зв`язатися з Базою даних або записами замовлень. Менеджер викликає адміністратора системи.

ІІІ. Коротка форма аналізу:

Для демонстрації короткої форми аналізу сценаріїв UseCases було використано варіант використання «Створення та корегування черги замовлень» з основним користувачем у вигляді Начальника виробничого відділу.

Опис сценарію варіанту використання «Створення та корегування черги замовлень».

Начальник виробничого відділу, отримавши форму замовлення від менеджера-консультанта, реєструє його в своїй системі як нове. Начальник виробничого відділу використовує систему для створення та корегування черг замовлень, яку поступають на виконання до майстрів з ремонту та обслуговування. Тому Начальник враховує основні фактори розподілення даних замовлень між майстрами: терміни та час виконання, завантаженість обладнання, зайнятість майстрів, кваліфікацію та спеціалізацію майстрів з ремонту та обслуговування, Терміново/не терміновість замовлення та деякі інші. При цьому у відповідності до терміновості та часових даних, начальник завантажує замовлення або в «хвіст» черги, або ж в «голову». При цьому система слідкує за часовими обмеженнями. На завершення, система зберігає дані в черзі та проставляє відповідний таймер/лічильник для слідкування за замовленнями. Далі ці ж замовлення з черги використовуються в варіанті використання «Призначення виконавців».

Питання до контролю

1. Дайте визначення варіанта використання (usecase).
2. Чим відрізняються варіант використання та сценарій (scenario)?
3. Які форми опису варіантів використання ви знаєте?
4. Дайте визначення та наведіть приклад короткої форми опису usecase.
5. Дайте визначення та наведіть приклад поверхневої форми опису usecase.
6. Що таке головний успішний сценарій?
7. Дайте визначення альтернативних сценаріїв? Чи можуть вони бути успішними/неуспішними?
8. Дайте визначення повної форми опису usecase.
9. Хто може виступати в ролі актора для варіанта використання?
10. Хто може виступати в ролі зацікавлених осіб (stakeholders) для варіанта використання?
11. Яким чином визначаються альтернативні сценарії і як вони пов' язані з головним успішним?
12. Яким чином визначається параметр frequency of occurrence для повної форми опису варіанта використання?

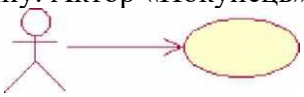
Лабораторна робота 3. МОДЕЛЮВАННЯ ДІАГРАМ ВАРІАНТІВ ВИКОРИСТАННЯ (USECASE DIAGRAMS)

Теоретичні відомості

Діаграми варіантів використання *(usecase diagrams)* використовуються для відображення сценаріїв використання системи *(usecases)* та користувачів системи *(actors)*, які використовують її функції.

Актори на діаграмі варіантів використання позначаються символом людини, а варіанти використання - еліпсом. Актори та варіанти використання поєднуються напрямленою асоціацією (unidirectional *association)* - стрілкою, що спрямована від актора до варіанта використання. Також актори можуть поєднуватися з використанням зв' язків

узагальнення. На рис. 1 показано фрагмент діаграми варіантів використання для інтернет-магазину. Актор «Покупець» при цьому може виконати сценарій «Замовити товар».



Customer Order the product

Рисунок 1 - Фрагмент діаграми варіантів використання в RSA

Варіанти використання можуть бути пов'язані між собою трьома видами зв'язків: узагальненням *(generalization),* розширенням *(extend* relationship) та включенням *(include* relationship).

Відношення узагальнення (generalization) показують відношення між загальним і частковим. Наприклад на рис. 2 варіанти використання «Search by category» та «Search by producer» є частковими випадками загального варіанта «Search product», тому вони поєднані даним відношенням. Також дане відношення може використовуватися для поєднання акторів. Актор «Administrator» може виконувати всі функції актора «Customer», тобто виступає частковим випадком покупця, але може виконувати і специфічні операції (варіант використання «Check db info»).

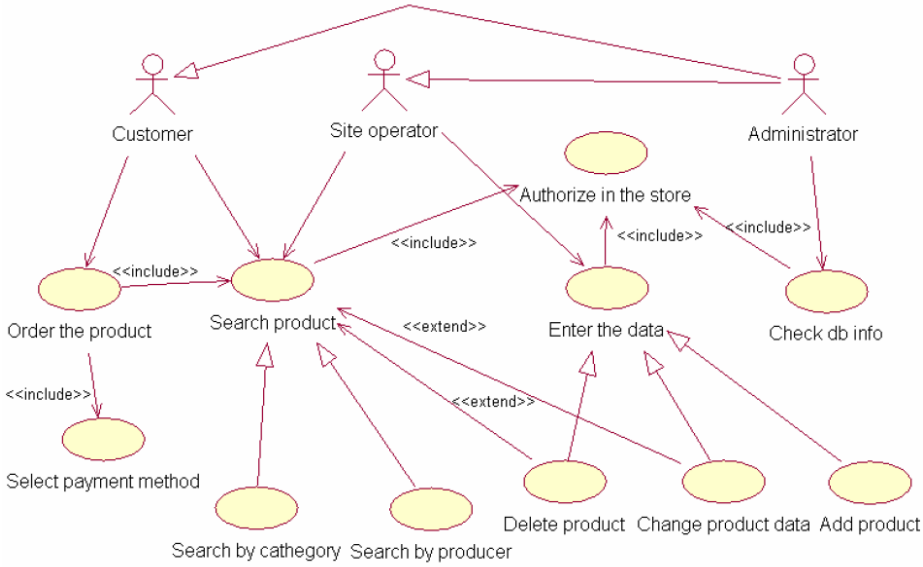


Рисунок 2 - Повна діаграма варіантів використання для інтернет-магазину

Відношення включення (include) відображає зв'язок «ціле - частина», тобто один варіант завжди в певний момент виконання повністю включає інший. Для прикладу частиною варіанта використання «Order the product» є сценарії «Search product» та «Select payment method», оскільки для того, щоб замовити товар покупець завжди має відшукати його в каталозі та обрати метод оплати.

Відношення розширення (extend) визначає такий тип відношення, коли один варіант за певних умов повністю використовує інший (розширює його). Так, наприклад, оператор Інтернет магазину може видалити товар («Delete product»), знаючи його ідентифікатор, або провівши попередньо пошук товару («Search product»).

Завдання. Створити діаграму варіантів використання для обраного варіанта комп'ютерної системи. Діаграма повинна містити усіх акторів (користувачів системи) та хоча б по три варіанти використання для кожного актора. Пов'язати варіанти використання та акторів, при цьому використати усі види зв'язків (unidirectional association, generalization, extend relationship, include relationship).

Приклад виконання

В результаті аналізу предметної галузі «Система продажу для Інтернет-

магазину» виявлено наступні актори системи.

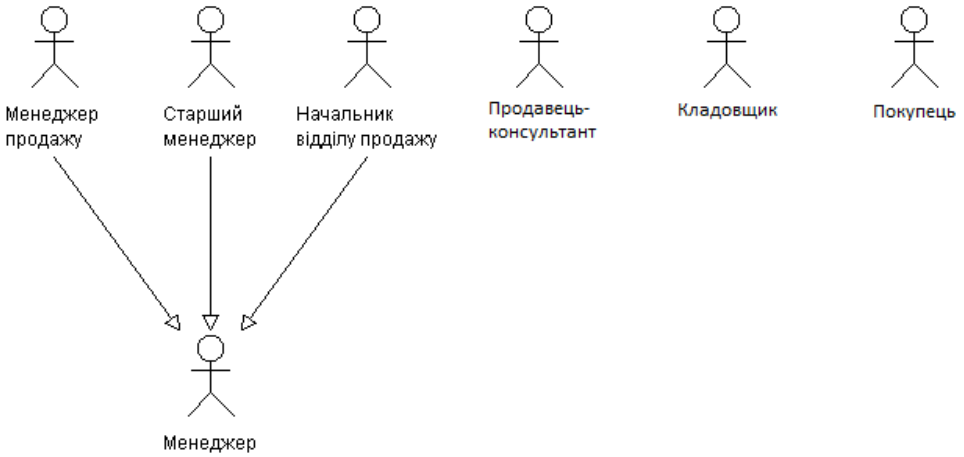


Рисунок 3 - Актори системи

Інтерв'ю, проведене з вказаними вище кандидатами показало, що менеджер продажів, старший менеджер і начальник відділу продажів припускають використовувати АСУ, що розробляється, однотипно. Це дозволило узагальнити ці 3 ролі в одну.

Короткий опис акторів

Актор	*Короткий опис*
Менеджер	*Вводить дані про товари. Створює/оформляє рекламу на сайті. Здійснює контроль замовлень.*
Диспетчер	*Консультування покупців. Оформлення замовлення.*
Кладовщик	*Виконання замовлень(Надсилання товару).*
Покупець	*Замовлення товару*

Виявлено наступні варіанти використання

Основний актор	*Найменування*	*Формулювання*
Менеджер	*Ввід даних про товар*	*Цей варіант використання дозволяє менеджеру вводити дані про товар.*
	Редагування/ видалення інформація	*Цей варіант використання дозволяє менеджеру редагувати/видалити дані про товар на сайті.*
Продавець-консультант	*Зміна замовлення*	*Продавець –консультант може змінювати замовлення при необхідності (якщо про це попросив покупець)*
Продавець-консультант	*Видалення замовлення.*	*Продавець –консультант може видаляти замовлення (якщо про це попросив покупець)*
	Підтвердження замовлення	*Продавець –консультант підтверджує замовлення після консультації.*
Покупець	*Здійснює замовлення*	*Покупець може здійснювати замовлення.*
	Відміняє замовлення	*Покупець може відмінити замовлення.*
	Оплата замовлення	*Покупець може здійснити оплату через сайт(або безпосередньо при отриманні товару)*
	Підтвердження оплати	*Покупець підтверджує оплату.*
Кладовщик	*Перегляд замовлень*	*Кладовщик може переглядати замовлення і виконує їх в порядку надходження*
	Зміна стану замовлення/товару (відправлений/ невідправлений)	*Відмічає надісланий товар, який покупець оплатив*
Менеджер/ Продавець-консультант/ Покупець/ Кладовщик	*Авторизація*	*Користувачі авторизуються на сайті*
Покупець	*Реєстрація*	*Покупець повинен зареєструватися на сайті, вказавши контактні дані та інші відомості.*

Аналіз варіантів використання виявив наступні взаємозв'язки

1. Варіант використання Зміна стану товару(замовлення) включає в себе 2 стани. Це можна відобразити наступним чином.

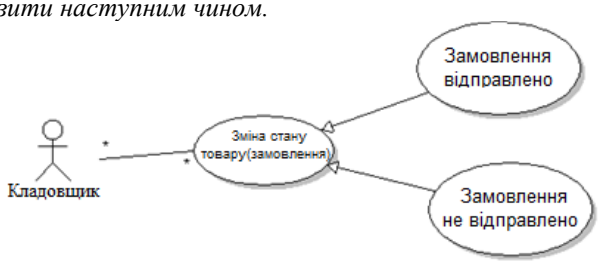


Рисунок 4 - Узагальнення варіантів зміни стану

Провівши аналогічний аналіз для кожного з акторів, отримаємо наступну діаграму варіантів використання системи в цілому:

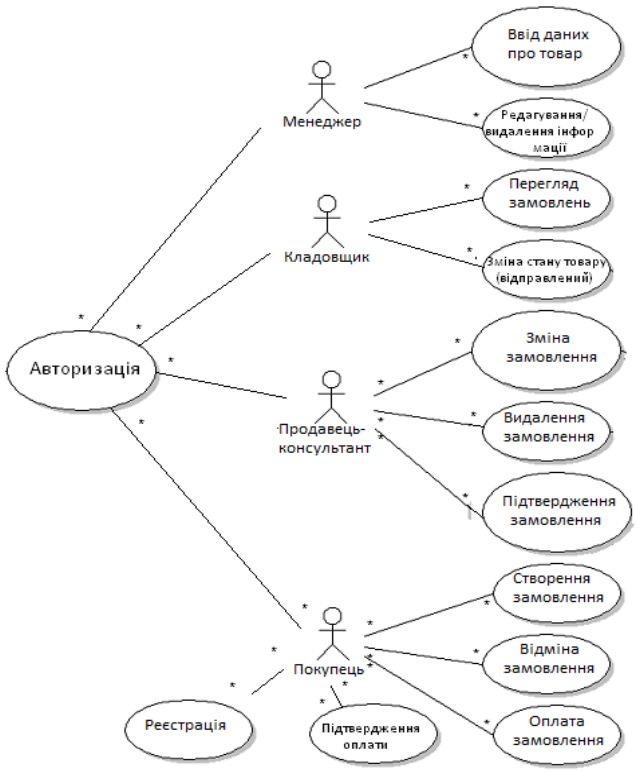


Рисунок 5 - Модифікована діаграма прецедентів системи

Конкретизація варіантів використання

М1. Ввід даних про товар

Основна дійова особа: Менеджер.

Інші учасники прецеденту: відсутні

Зв'язки з іншими варіантами використання: відсутні

Короткий опис. Даний варіант використання дозволяє Менеджерові вводити дані про товар, всі важливі характеристики(ціну, типу товару, марку, модель, бренд, потужність, та ін.) товару, ряд банерів(якщо потрібно).

М2. Редагування видалення інформації

Основна дійова особа: Менеджер.

Інші учасники прецеденту: відсутні.

Зв'язки з іншими варіантами використання: відсутні

Короткий опис. Даний варіант використання дозволяє менеджерові вносити зміни або видаляти інформацію в Інтернет-магазині. Це необхідно коли змінюється ціна або інші характеристики, а також видалення якщо товар зняли з виробництва або немає в наявності.

РК1. Зміна замовлення.

Основна дійова особа: Продавець-консультант.

Інші учасники прецеденту: Покупець.

Зв'язки з іншими варіантами використання: відсутні

Короткий опис. Даний варіант використання дозволяє Продавцю-консультанту змінити замовлення, якщо цього побажав покупець після консультації.

РК2. Видалення замовлення

Основна дійова особа: Продавець-консультант.

Інші учасники прецеденту: Покупець

Зв'язки з іншими варіантами використання: відсутні

Короткий опис. Даний варіант використання дозволяє Продавцю-консультанту видалити замовлення якщо того побажав покупець, або якщо покупець не оплатив замовлення, або якщо товару немає в наявності. Після видалення замовлення покупець який його робив буде проінформований.

РК3. Підтвердження замовлення

Основна дійова особа: Продавець-консультант.

Інші учасники прецеденту: Покупець

Зв'язки з іншими варіантами використання: відсутні.

Короткий опис. Даний варіант використання дозволяє продавцю-консультанту підтвердити замовлення після консультації покупця. Після цього замовлення буде направлено до кладовщика.

Р1. Створення замовлення

Основна дійова особа: Покупець.

Інші учасники прецеденту:

Зв'язки з іншими варіантами використання: відсутні.

Короткий опис. Цей варіант використання дозволяє покупцеві створити замовлення.

Р2. Відміна замовлення

Основна дійова особа: Покупець.

Інші учасники прецеденту: відсутні

Зв'язки з іншими варіантами використання: РК2 «Видалення замовлення».

Короткий опис. Цей варіант використання дозволяє покупцеві відмінити своє замовлення безпосередньо за допомогою сайту, про це буде повідомлено продавця-консультанта і кладовщика (якщо замовлення вже було підтверджено і оплачено)

Р3. Оплата замовлення

Основна дійова особа: Покупець.

Інші учасники прецеденту: відсутні

Зв'язки з іншими варіантами використання: відсутні

Короткий опис. Покупець може здійснювати оплату-онлайн через сайт за допомогою кредитних рахунків Visa, MasterCard та інші. Він повинен вказати номер рахунку і код безпеки.

Р4. Підтвердження оплати

Основна дійова особа: Покупець.

Інші учасники прецеденту: відсутні

Зв'язки з іншими варіантами використання: відсутні

Короткий опис. Система надсилає покупцеві повідомлення про підтвердження оплати, де можна натиснути «підтвердити» або «відхилити». Повідомлення приходить після того коли покупець ввід дані онлайн рахунку. Всі підтверджені замовлення надходять кладовщику.

Р5. Реєстрація

Основна дійова особа: Покупець.

Інші учасники прецеденту: відсутні

Зв'язки з іншими варіантами використання: Р1,Р2,Р3,Р4.

Для доступу до всіх вказаних вище варіантів використання покупець повинен пройти реєстрацію в системі(на сайті) і потім авторизуватися.

К1. Перегляд замовлень

Основна дійова особа: Кладовщик.

Інші учасники прецеденту: відсутні

Зв'язки з іншими варіантами використання: відсутні.

Система показує кладовщику всі наявні замовлення.

К2. Зміна стану замовлення

Основна дійова особа: Кладовщик.

Інші учасники прецеденту: відсутні

Зв'язки з іншими варіантами використання: відсутні.

Кладовщик може відмічати відправлений товар, коли товар не відправлений є додатковий атрибут «немає в наявності». В випадку коли товару немає в наявності система надсилає повідомлення покупцеві.

МРК. Авторизація

Основна дійова особа: Покупець.

Інші учасники прецеденту: відсутні

Зв'язки з іншими варіантами використання: відсутні

Короткий опис. Для розпізнавання акторів на сайті введена авторизація. Її повинні пройти усі актори, щоб отримати доступ до свої варіантів використання.

Питання до контролю

1. Для чого призначена діаграма варіантів використання (usecase diagram)?
2. Дайте визначення актора (actor) програмної системи. Хто може виступати в ролі акторів?
3. Дайте визначення варіанта використання (usecase). Яким чином варіант використання визначається на діаграмі?
4. Перелічіть відношення, що можуть використовуватися на діаграмі.
5. Дайте визначення відношення узагальнення (generalization). Наведіть приклад.
6. У чому відмінність між відношенням узагальнення між акторами та між варіантами використання?
7. Дайте визначення відношення асоціації (association). Наведіть приклад.
8. Дайте визначення відношення включення (include). Наведіть приклад.
9. Дайте визначення відношення розширення (extend). Наведіть приклад.
10. Які відношення можуть використовуватися для поєднання тільки акторів, акторів і варіантів використання?

Лабораторна робота № 4. ПОБУДОВА ДІАГРАМ ВЗАЄМОДІЇ (INTERACTION DIAGRAMS)

Теоретичні відомості

Діаграми взаємодії *(Interaction Diagrams)* відображають взаємодію логічних елементів системи між собою у процесі виконання актором певного варіанта використання. Розрізняють два типи діаграм взаємодії: діаграми послідовності *(Sequence Diagrams)* та кооперації *(Collaboration Diagrams),* які є різними представленнями одного і того ж процесу.

Головними елементами *діаграм послідовності* є *об'єкти,* які є логічними сутностями, що представляють окремі елементи системи та *повідомлення,* якими вони обмінюються. В якості об'єктів можуть виступати також актори. Повідомлення можуть бути не тільки абстрактними діями, що виконуються, але і методи класів, створених на діаграмі класів. Повідомлення на діаграмі послідовності пронумеровані, тобто мають чітку послідовність.

Для прикладу розглянемо систему електронного документообігу на підприємстві. Для побудови діаграми послідовності оберемо варіант використання «Створити документ», який виконує актор «Користувач» (рис. 3).

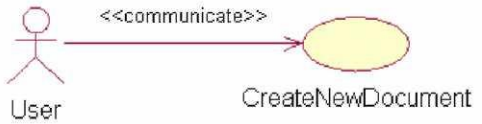


Рисунок 6 - Фрагмент діаграми варіантів використання для системи документообігу

Наведемо опис головного успішного сценарію даного варіанта використання у короткій формі: *Користувач має на екрані форму із запитом на створення документа, він обирає шаблон документа зі списку шаблонів, задає шлях для збереження та назву документа; система зберігає введені дані в якості параметрів документа, запускає прикладне програмне забезпечення, яке відповідає формату документа (наприклад MS Word, MS Excel і т. д.) та створює документ із відповідною назвою.*

На рис. 7 наведено діаграму послідовності, створену в середовищі RSA для даного прикладу. У верхній частині діаграми наведено перелік об'єктів (логічні сутності), які взаємодіють між собою у процесі виконання сценарію. Часова шкала на даній діаграмі направлена згори донизу, крім того повідомлення пронумеровані відповідно до черги їх пересилання між об'єктами. Нижче наведено короткий опис подій, що відбуваються при виконанні даного варіанта використання.

При ініціалізації діалогу для створення нового документа (1) завантажується (2) та відображається (3) список шаблонів документів. При виборі користувачем одного з шаблонів (4), в діалозі відображається його початковий вигляд (5). Після того, як користувач підтверджує свій вибір остаточно, наприклад, скориставшись методом DoubleClick (6), система ініціалізує діалог (7), де просить ввести шлях до директорії та ім'я файла, у якому документ буде зберігатися. При цьому відображаються default значення для шляху та імені файла (8). Користувач вводить шлях (9) та ім'я файла (10), система перевіряє введені дані (11). Система зчитує інформацію шаблона (12) і записує всі параметри до об'єкта, який представляє документ (13). Потім документ додається до списку активних документів (14) і відповідно до параметрів документа запускається прикладна програма, яка відповідає формату шаблона документа (15). При цьому в прикладній програмі створюється новий документ з ім' ям, яке ввів користувач (16) (рис. 7).

Додатковими елементами діаграми послідовності є лінії життя об’єктів (довгі вертикальні пунктирні лінії), які відображають час життя об'єкта від його створення до знищення; фокус керування (прямокутники на лініях життя об’єктів) відображає, який об’єкт виконує операції в певний момент часу; та символи знищення об’єктів (хрест на лінії життя об’єкта), що відображає процес знищення об’єкта (рис. 5). Наприклад, об'єкт «Authorization Record* на діаграмі створюється на 2-му кроці виконання сценарію, використовується протягом 4-6 кроків та знищується після шостого кроку.

Основні типи повідомлень на діаграмі послідовності:

- пряме - повідомлення, яке об'єкт-ініціатор надсилає об'єкту-приймачу (1-4);
- рефлексивне, яке об’єкт надсилає сам собі (5);
- зворотне, при якому управління повертається об’єкту ініціатору (6), часто повідомлення даного типу не мають назви.

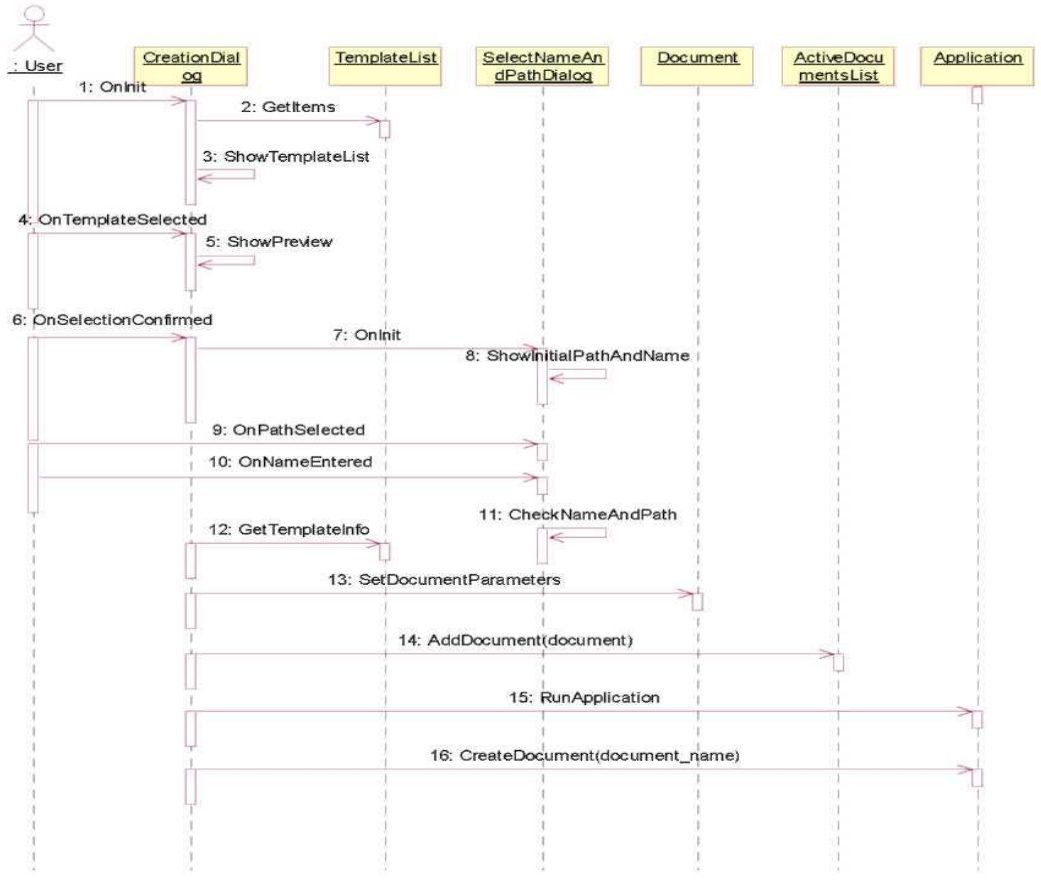


Рисунок 7 - Діаграма послідовності для варіанта використання «Створити новий документ»

Діаграма кооперації має те ж саме призначення, що і діаграма послідовності, але відображає процес виконання варіанта використання в іншій нотації (рис. 8). Її головними компонентами є об'єкти, назви повідомлень та їх напрям. Характеристика даних компонентів була надана у процесі опису діаграми послідовності.

Примітка: діаграму послідовності можна автоматично перетворити в діаграму кооперації (або навпаки), натиснувши клавішу F5.

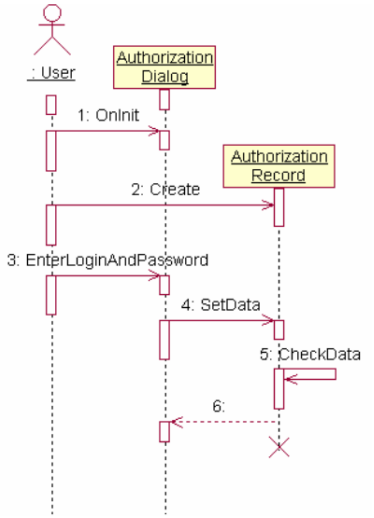


Рисунок 8 - Фрагмент діаграми послідовності для процесу авторизації користувача

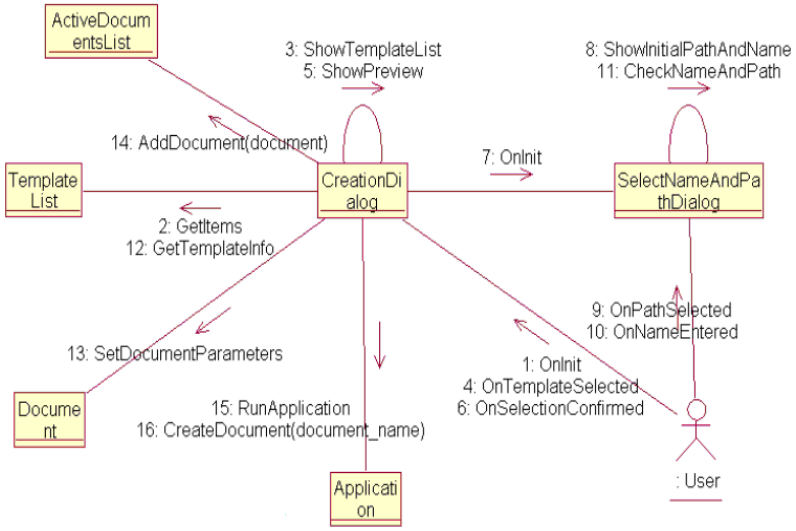


Рисунок 9 – Діаграма кооперації для варіанту використання «Створити новий документ»

Завдання. Для кожного варіанта використання на Usecase Diagram створити Sequence або Collaboration Diagram (тобто у проекті повинно бути не менше шести діаграм кооперації та послідовності). На кожній діаграмі взаємодії повинен бути головний актор (при наявності) та не менше 5 об'єктів. Кожна діаграма взаємодії повинна містити не менше 10 повідомлень, якими обмінюються об'єкти в процесі виконання сценарію. Загальна сума різних об'єктів у проекті повинна налічувати 12-15 об'єктів. Об'єкти та повідомлення на діаграмах повинні мати зрозумілі назви. При побудові діаграм використовувати прямі, рефлексивні та зворотні типи повідомлень, а також символи знищення об'єктів.

Приклад виконання

«Інформаційна система підтримки веломайстерні»

1) *Діаграма послідовностей для варіанту використання «Реєстрація замовлення».*

Реєстрація замовлення

Короткий опис. Менеджер організації отримує дані від клієнта та вносить замовлення з даними в систему.

Дійові особи цього прецеденту – Менеджер організації.

Потік подій

Прецедент починається, коли Менеджер організації вибирає функцію "Реєстрація замовлення " з «Головної форми» АРМ «Менеджер організації».

Базовий потік – Реєстрація нового замовлення:

1. *Менеджер організації обирає пункт меню системи - «Реєстрація нового замовлення».*
2. *Система відображає реєстраційну форму для внесення даних(Інформація про клієнта, серійний номер рами/велосипеда, термін прийому замовлення, марка велосипеда) про замовлення.*
3. *Менеджер організації вносить в форму дані (інформація про клієнта, серійний номер рами/велосипеда, термін прийому замовлення, марка велосипеда), отримані від клієнта.*
4. *Система зберігає дані та відображує список, який передбачає множинний вибір можливих робіт над замовленням, а також їхні вартості.*
5. *Менеджер організації вибирає множину робіт над замовленням з списку-чекбоксу, які необхідні для обслуговування даного клієнта.*
6. *Система виводить запит про формування звіту та підраховує загальну вартість замовлення і приблизний час завершення виконання замовлення.*
7. *Менеджер вибирає пункт сформувати звіт замовлення.*
8. *Система формує звіт замовлення та надсилає його на екран користувача (роздруковує його на папері).*
9. *За погодження клієнта, менеджер обирає пункт зберегти замовлення.*
10. *Система зберігає замовлення в БД замовлень та резервує відповідний час надходження замовлення, час виконання.*
11. *Система повертається до п.1(тобто до можливості вибору менеджером пункту «Реєстрація замовлення»).*

Рисунок10- Діаграма послідовностей для прецеденту «Реєстрація замовлення».

2) Діаграма послідовностей для варіанту використання «Формування черги замовлень»
Короткий опис

Начальник виробничого відділу отримує дані від Менеджера організації після виконання ним операції «передачі замовлення на стацію виконання». Начальник виробничого відділу розташовує дані в черзі замовлень, додаючи їх у відповідності з часовими даними.

Дійові особи цього прецеденту – Начальник виробничого відділу.

Потік подій

Прецедент починається, коли Начальник виробничого відділу вибирає функцію "Сформувати чергу замовлень " з «Головної форми» АРМ «Начальник виробничого відділу».

Базовий потік – Сформувати чергу замовлень.:

1. Начальник виробничого відділу обирає пункт меню «Сформувати чергу замовлень».

2. Система надає Начальнику виробничого відділу дані про поточний стан черги невиконаних
замовлень та про замовлення, які ще не поміщені в чергу.

3. Начальник виробничого відділу вибирає операцію «Додати нове замовлення в чергу».

4. Система відображає часову шкалу з чергами замовлень, а також вільні проміжки, не заняті замовленнями.

5. Начальник виробничого відділу методом «Drag & Drop» переносить записи з списку тих, що тільки найшли на виробництво, в часову шкалу черги замовлень(в хвіст черги).

6. Система перевіряє можливість виконання замовлення в дані терміни та сумісність з ресурсами. Система резервує даний час на виконання замовлення в черзі.

7. Менеджер підтверджує дані зміни в черзі.

8. Система видаляє замовлення з списку замовлень, що очікували на входження чергу.

9. Пункти сценарію 4-12 повторюються, поки не буде заповнена черга, використано всі замовлення, що надійшли або Начальник відділу на завершить роботу з системою.

10. Система робить відповідні відмітки в базі даних про змінену чергу замовлень.

Рисунок11 - Діаграма послідовностей для прецеденту «Формування черги замовлень».

3) Діаграма послідовностей для прецеденту «Фіксація результатів виконання замовлення»

Короткий опис

Після проведення обслуговування, майстрові з ремонту та обслуговування необхідно заповнити форму для фіксації виконаних робіт та результату роботи.

Дійові особи цього прецеденту – Майстер з ремонту та обслуговування.

Потік подій

Прецедент починається, коли Майстер з ремонту та обслуговування вибирає функцію "Зафіксувати результат обслуговування " з «Головної форми» АРМ «Майстер з ремонту та обслуговування».

Базовий потік – Фіксація результатів виконання замовлення.:

1. *Майстер з ремонту та обслуговування обирає пункт меню системи - «Результат обслуговування» та підтверджує завершення робіт.*
2. *Система реєструє час завершення та виводить форму для заповнення майстром даних.*
3. *Майстер заповнює дані про проведене обслуговування та про результат роботи.*
4. *Система зберігає дані, заповнені майстром з ремонту та обслуговування та формує звіт завершення робіт.*
5. *Майстер з ремонту та обслуговування підтверджує сформований звіт.*
6. *Система відправляє звіт Начальникові виробничого відділу для завершення виконання замовлення та вилучає дане замовлення з черги замовлень та відзначає даного майстра як такого, що очікує на нове замовлення. Система виводить відповідне повідомлення Майстру з ремонту та обслуговування.*
7. *Система повертається до пункту 1.*

Рисунок12 - Діаграма послідовностей для прецеденту «Фіксація результатів виконання замовлення».

4) Діаграма послідовностей для прецеденту «Обробка запиту щодо виконання та змісту замовлення»

Обробка запиту щодо виконання та змісту замовлення

М4	*Менеджер організації або Начальник виробничого відділу*	*Обробка запиту щодо виконання та змісту замовлення*	*Використовується менеджером організації або начальником виробничого відділу для пошуку потрібної інформації про стан замовлення у виробництві, довідкову інформацію.*

Основна дійова особа: Менеджер організації, або Начальник виробничого відділу

Короткий опис. Даний варіант використання дозволяє Менеджерові організації та Начальнику виробничого відділу дізнаватися про план виробництва замовлення, а також про результати виконання робіт. Виконує функцію контролю виконання замовлень.

Базовий потік подій:

1. *Користувач обирає пункт меню «Запит щодо обслуговування».*
2. *Система надає вибір критерій сортування та створення вибірки з БД.*
3. *Користувач обирає критерії відбору даних і підтверджує даний вибір.*
4. *Система робить вибірку з даних, що існують в БД і що попадають під критерії відбору. Система виводить дані користувачеві.*
5. *Користувач отримує дані для перегляду.*

Рисунок13 - Діаграма послідовностей для прецеденту «Запит щодо виконання та змітсу замовлення»

Діаграма кооперацій

1)Діаграма кооперацій для варіанту використання «Реєстрація нового замовлення»

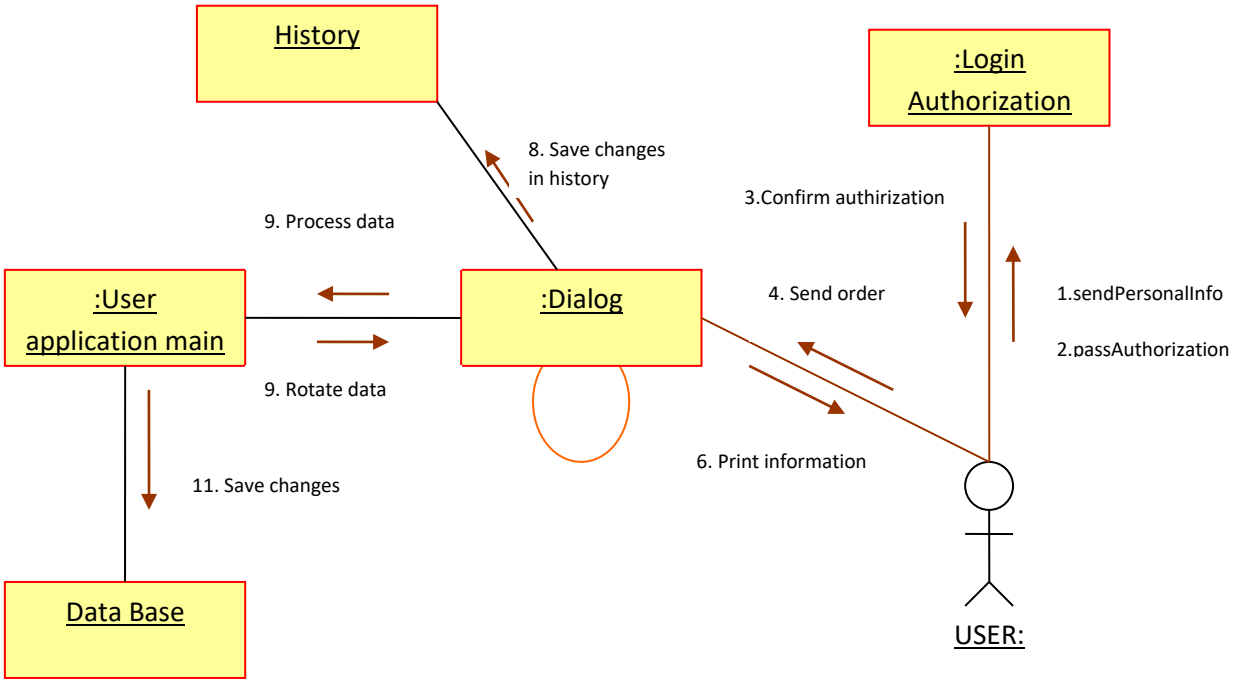


Рисунок14 - Діаграма кооперацій для варіанту використання «Реєстрація нового замовлення».

Опис діаграми кооперацій:

Для успішного виконання прецеденту «Реєстрація нового замовлення», та відповідно успішного завершення реєстрації нового замовлення необхідно: (1) та (2) – користувачеві авторизуватися в системі АІС.

При цьому йому доведеться ввести свої персональні дані та пройти авторизацію(одноразово при вході в систему) і система повинна підтвердити авторизацію(3). Після цього буде запропоновано ввести відповідні дані для оформлення замовлення(4) та відправити дані системі на обробку(5). Система у відповідь на ці запити обробляє ці дані(9) та вносить відповідні повідомлення, трансфер даних(10). Внесена інформація відповідно зберігається в БазіДаних (11) та ведуться записи в історії(лог) (8). Потім всі оброблені дані повідомляються користувачеві(6, 7).

2) Діаграма кооперацій для прецеденту «Обробка запиту щодо виконання та змісту замовлення»:

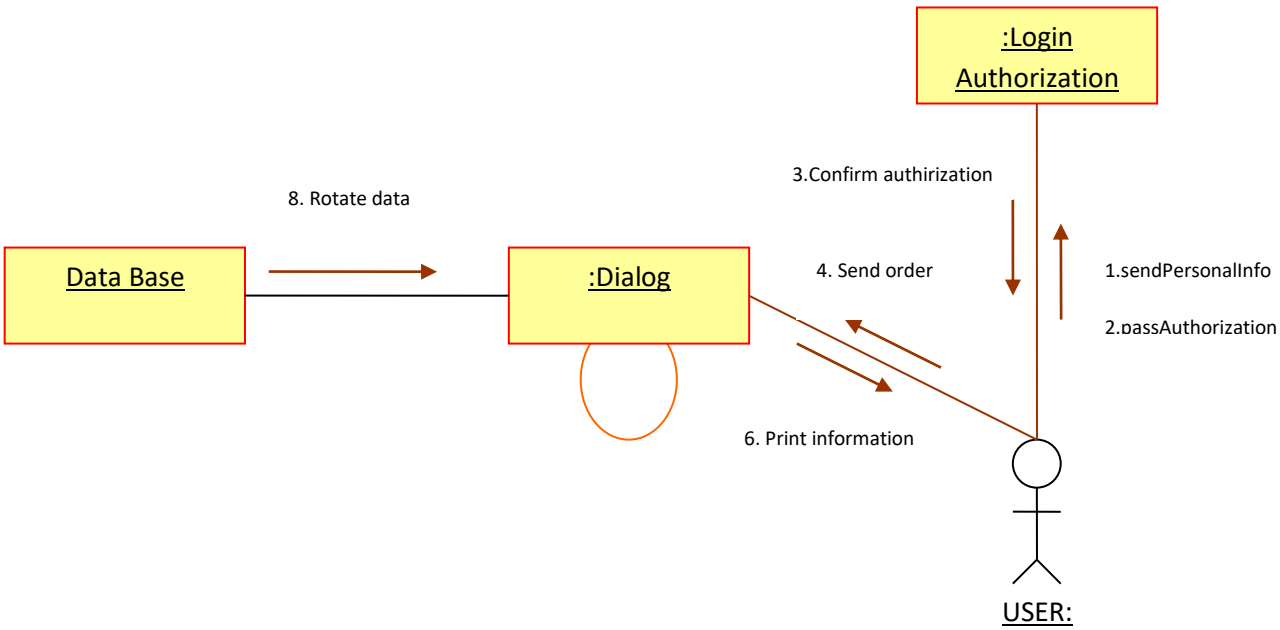


Рисунок15 - Діаграма кооперацій для прецеденту «Обробка запиту щодо виконання та змісту замовлення»

Опис сценарію виконання даного прецеденту:

Для виконання прецеденту «Обробка запиту щодо виконання та змісту замовлення» необхідно:

авторизуватися користувачеві у системі для введення своїх персональних даних та отримання доступу до інформації (1,2). Після цього він має можливість надіслати запит для отримання інформації системі(4, 5). Система обробляє інформацію користувача, формує запит в Базу Даних та надсилає запит за даними в БД(6,7).

БД отримає інформацію про запит, робить вибірку даних і надсилає їх системі, в якій працює користувач (8,9).

Після цього система видає дані на розгляд користувача.

Питання до контролю

1. Які види діаграм взаємодії (interaction diagram) ви знаєте?
2. Для чого призначена діаграма послідовності (sequence diagram)? Наведіть її основні елементи.
3. Дайте визначення об'єктів діаграми послідовності, наведіть їх основні характеристики.
4. Дайте визначення повідомлень діаграми послідовності, перелічіть основні типи повідомлень.
5. Для чого використовуються прямі, зворотні та рефлексивні повідомлення?
6. Яким чином на діаграмі послідовності використовуються актори?
7. Що таке лінія життя об'єкта? Чим визначається початковий і кінцевий момент періоду життя кожного об'єкта?
8. Для чого призначена діаграма кооперації (collaboration diagram)? Наведіть її основні елементи.
9. В яких випадках зручніше використовувати діаграму послідовності?
10. В яких випадках зручніше використовувати діаграму кооперації?
11. Як діаграму послідовності можна перетворити в діаграму кооперації?

Лабораторна робота № 5. ПОБУДОВА ДІАГРАМ КЛАСІВ

Теоретичні відомості

Діаграми класів *(Class diagrams)* - головний тип діаграм UML, які відображають логічну структуру програмної системи та суттєво впливають на процес генерації програмного коду. Основними елементами діаграми класів у RSA є безпосередньо класи *(classes)* та відношення між ними *(relations).*

Клас - сукупність логічних об'єктів, які мають схожі характеристики і відрізняються однаковою поведінкою. В ООП характеристики об'єктів певного класу представлені сукупністю атрибутів, а поведінка - сукупністю операцій.

Рисунок 16 - Графічне представлення класу на діаграмі класів у RSA

У Rational Software Architect (RSA) кожен клас графічно представлений прямокутником, що має три секції: ім'я класу, перелік атрибутів та перелік операцій (рис. 16). Для кожного атрибуту задається тип даних, для кожної операції тип даних для значення, що повертається, та перелік параметрів. Атрибути та операції можуть бути визначені для кожного об’єкта класу, чи для класу в цілому *(static* attributes and operations). Також для всіх атрибутів та операцій можна визначити тип видимості (public, protected, private).

Розрізняють декілька типів класів:

1. Конкретний клас - для даного класу можна створити об’єкт. Абстрактний клас - клас, для якого не можна створити об’єкт. Даний клас виступає чистою абстракцією, від нього можна наслідувати конкретні класи.

2. Параметризований клас - клас, для визначення якого використовується список формальних параметрів, які впливають на атрибути та операції (аналог шаблона в C++). Найчастіше такі класи використовуються для створення абстрактних типів даних (списки, вектори, стеки, черги і т. п.).

4. Інтерфейс - клас, що містить тільки визначення набору операцій (без реалізації). У мові C++ аналогом даного класу є клас, всі операції якого є чистими віртуальними функціями. В мові Java класи можуть реалізовувати декілька інтерфейсів (інтерфейси використовуються для реалізації концепції множинного наслідування).

Створення нових класів та операцій класів:

У RSA для кожного об’єкта на діаграмах взаємодії існує можливість призначити певний клас (чи створити новий). При цьому в прямокутнику об’єкта дані відображаються в наступному форматі <ім'я об'єкта>:<ім'я класу>. Також кожне повідомлення на діаграмі взаємодії представляє певну операцію класу-приймача. Для кожного повідомлення можна обрати існуючий метод класу-приймача чи створити новий. На рис. 17 наведено фрагмент модифікованої діаграми послідовності для варіанта використання «Переглянути список студентів».

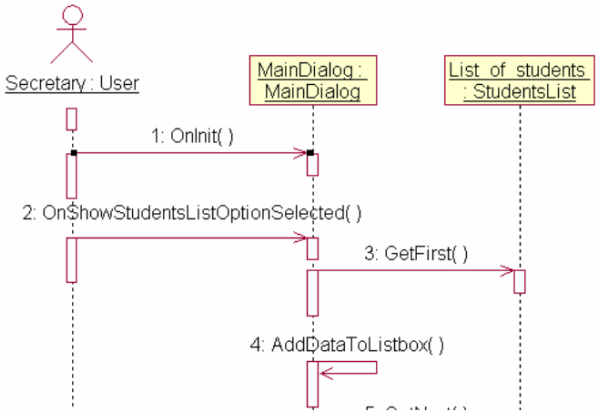


Рисунок 17 - Фрагмент діаграми послідовності для варіанта використання «Переглянути список студентів»

Після створення набору класів з операціями можна відобразити їх на діаграмі класів. При цьому кожен клас буде мати ім'я та перелік операцій, визначених

користувачем. На рис. 18 показано відображення класу MainDialog з операціями, створеними для повідомлень 2 та 4 попередньої діаграми (повідомлення 1 буде показано пізніше на загальному вигляді діаграми класів, оскільки воно визначене для класу Dialog, який є базовим для даного класу).

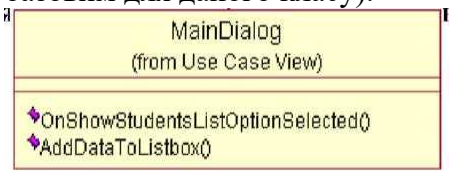


Рисунок 18 - Відображення класу MainDialog на діаграмі класів

Також нові класи та операції класів можна створювати безпосередньо з діаграми класів.

Додавання атрибутів класів:

Після створення переліку класів із набором операцій необхідно для кожного класу створити набір атрибутів - даних, якими оперує програма у процесі виконання операцій класу. Наприклад, виходячи з діаграми послідовності, зображеної на рис.19 можна визначити для класу MainDialog атрибути для кнопки (Button), яку натискає користувач для перегляду списку студентів та списку, до якого безпосередньо додається інформація про кожного студента.

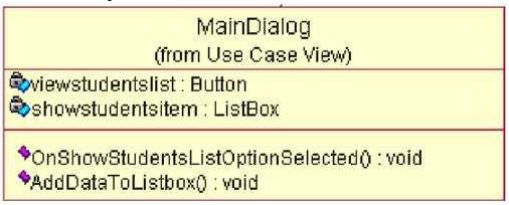


Рисунок 19 - Клас MainDialog з повним набором атрибутів та операцій

Повний формат відображення атрибутів на діаграмі класів:

<attribute_name>:<attribute_type>

Повний формат відображення операцій на діаграмі класів:

<operation_name>(<list_of_arguments>):<type_of_return_value> *Відношення між класами:*

RSA для нотації Unified можна задати 5 основних типів *відношень* між класами:

1. Відношення асоціації *(association)* - визначає абстрактний зв'язок між класами, може представляти аналог відношення «m до n». *Приклад:* відношення між класами Student та Course (кожен студент відвідує декілька курсів (дисциплін), для кожної дисципліни визначений набір студентів, які її відвідують). *Представлення в програмному коді:*

```
class Student
{  // …..
Course* theCourses;
// ..........
};
class Course
{//….
Student* theStudents;
//…..
};
```

При цьому Course* в класі Student може вказувати на масив дисциплін, а Student* у класі Course - на перелік студентів.

2. Відношення агрегації *(aggregation)* - відношення типу «частина - ціле», при якому час життя об'єкта класу частини не співпадає з часом життя об'єкта класу цілого.

Приклад: відношення між класами Student та Ticket (у час переоформлення студентського квитка студент не має його). *Представлення у програмному коді:*

```
class Student
{//      …
Ticket* ticket;
//…
};
```

Атрибут ticket при цьому може створюватися та видалятися (за допомогою операторів new та delete) в коді будь-яких методів класу Student.

3. Відношення композиції *(composition)* - відношення типу «частина - ціле», при якому час життя об'єкта класу частини співпадає з часом життя об'єкта класу цілого.

Приклад: відношення між класами Student та Date_of_Birth (кожен студент у будь-який момент часу характеризується датою народження).

Представлення у програмному коді:

```
class Student
{//…
Date_of_Birth theDate_of_Birth;
//…
 };
```

Атрибут theDate_of_Birth існує протягом усього часу життя об'єкта класу Student.

4. Відношення наслідування *(generalization)* - відношення типу «загальне - часткове».

Приклад: відношення між класами Student та Astudent (студент відмінник є частковим випадком студента).

Представлення в програмному коді:

```
class Student;
class Astudent : public Student{
float GetScholarBonus();
        };
```

Клас Astudent має додатковий метод GetScholarBonus(), який розраховує надбавку до стипендії для студента відмінника.

5. Відношення інстанціювання (instantiation) - визначає інстанціювання нового класу з параметризованого класу шляхом підставлення фактичних параметрів у формальні параметри шаблона. *Приклад:* відношення між класами List та StudentsList (з абстрактного списку інстанціюється список студентів). *Представлення в програмному коді:*

```
template <class T> class List {       } // List - абстрактний список
typedef List <Student> StudentsList; // List<Student> - список студентів
```

Для класу, що інстанціюється, генерація програмного коду відсутня, оскільки він генерується компілятором автоматично при зверненні до параметризованого класу (підставлення фактичних параметрів у формальні параметри шаблона).

На рис. 20 наведено приклад діаграми класів для системи обліку успішності студентів у деканаті, яка містить перелік класів, пов'язаних різними типами відношень.

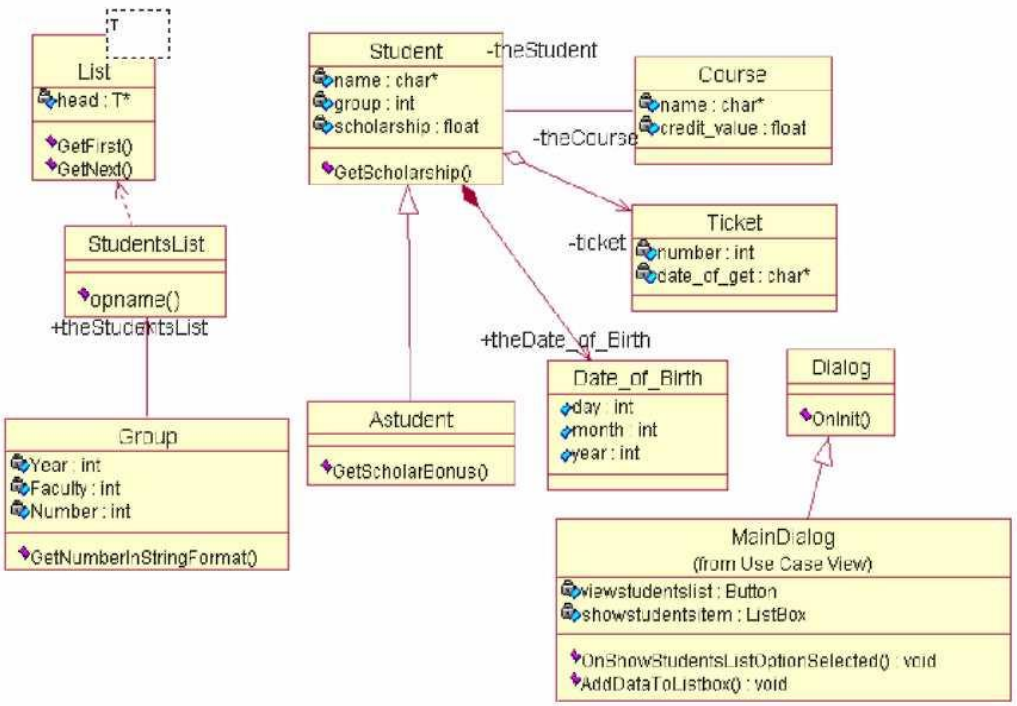


Рисунок 20 - Діаграма класів для системи обліку успішності навчання студентів у деканаті

Завдання. Для всіх об'єктів на діаграмах взаємодії призначити (створити) певний клас; для кожного повідомлення призначити (створити) відповідний метод (операцію) для класу об'єкта-приймача.

1. Розташувати створені класи з переліком операцій на діаграмі класів.
2. Для кожної операції визначити атрибути, які вона використовує та при необхідності додати їх до списку атрибутів класу.
3. Для кожного атрибуту задати логічний тип даних, для кожної операції логічний тип даних для return value та для переліку аргументів, якщо вони присутні.
4. Пов'язати класи на діаграмі класів, використовуючи різні типи відношень (асоціацію, агрегацію, композицію, наслідування, інстанціювання).

Вимоги

1. Діаграма класів повинна містити не менше 10 класів.
2. Для кожного класу визначити не менше 5 атрибутів та 5 операцій.
3. По можливості використати всі типи відношень між класами.

Приклад виконання

Дана діаграма класів побудована на основі прецеденту «Реєстрація нового замовлення» для Інформаційної система підтримки роботи веломайстерні.

На діаграмі позначено класи:

1) Клас-інтерфейс користувача для авторизації, виходу з системи та вибору меню реєстрації замовлення.

2) Клас для авторизації в системі користувача.

3) Клас-меню для вибору операції реєстрації нового замовлення.

4) Клас для представлення даних та для внесення основної інормації про замовлення.

5) Клас для роботи з списком можливих робіт над замовленням.

6) Список можливих робіт над замовленням та його часові дані та вартість виконання.

7) Додаткові системи – база даних та файл логу-історії.

Зв`язки:

1) Класи «Login» є класом-зв`язкою для класу «User». Тобто між ними існує зв`язок типу «залежгість»

2) Класи «User», «History» та «DataBase» пов`язані з класом для роботи з даними за допомогою зв`язку «асоціація».

3) Клас «List of work» також залежить від класу «New order» і між цими класами істує зв'язок типу «Асоціація».

4) Зв'язок «Агрегація» існує між класами «List of works» і «List of work» та між «User» і «Manager interfase». Це забезпечує існування відношення між ними типу «частина - ціле»

Рисунок 21-Діаграма класів для прецеденту «Реєстрація новго замовлення»

Питання до контролю

1. Для чого призначена діаграма класів (class diagram)? Назвіть її основні елементи.
2. Дайте визначення класу. Які типи класів ви знаєте?
3. Чим відрізняється абстрактний клас від конкретного?
4. Чим відрізняється абстрактний клас від інтерфейсу?
5. Що таке параметризований клас? Дайте визначення процесу інстанціювання. Наведіть приклад.
6. Назвіть основні характеристики класу.
7. Що таке атрибути класу? Які типи атрибутів ви знаєте?
8. Що таке операції класу? Які типи операцій ви знаєте?
9. Дайте визначення статичних атрибутів та операцій класу.
10. Дайте визначення наслідування. Назвіть основні типи наслідування.
11. Яким чином реалізується інкапсуляція для атрибутів та операцій класу? Назвіть типи модифікаторів доступу до елементів класу.
12. Яким чином реалізується поліморфізм для класів, що знаходяться в єдиній ієрархії? Наведіть приклад.
13. Що таке віртуальні функції, для чого вони призначені?

14. Яким чином можна створити класи та їх операції з діаграм взаємодії?
15. В якому форматі відображають атрибути та операції на діаграмі класів?
16. Перелічіть можливі відношення між класами на діаграмі класів.
17. Які характеристики можна визначити для відношення асоціації?
18. Як відношення асоціації можна перетворити в агрегацію/композицію?
19. У чому відмінність між відношеннями агрегації та композиції?
20. Який клас є базовим, а який похідним при визначенні відношення наслідування?
21. Які типи класів поєднує відношення інстанціювання?
22. Яким чином кожний тип відношень між класами впливає на генерацію програмного коду?

Лабораторна робота № 6. ДІАГРАМИ СТАНІВ ТА ПЕРЕХОДІВ

Теоретичні відомості

Діаграми станів та переходів (statechart diagrams) разом із діаграмами діяльності та взаємодії, відображають певний сценарій, що виконується у процесі функціонування системи в цілому, або певної її частини.

Діаграма станів відображає скінчений автомат у вигляді графу, вершинами якого є стани об'єкта, поведінка якого моделюється, а переходами - події, які переводять об'єкт, який розглядається, з одного стану в інший. При цьому вважається, що час перебування об'єкта в певному стані набагато більший за час, необхідний для переходу з одного стану в інший, тобто переходи між станами здійснюються миттєво.

Стан (state) - це логічна сутність, що використовується для моделювання певної ситуації, дії, процесу. Кожен стан має ім'я та список внутрішніх дій. В якості імені стану найчастіше використовуються іменники, наприклад: «Введення паролю», «Очікування», «Перевірка параметрів». Список внутрішніх дій містить перелік дій, які виконуються у процесі знаходження системи чи об’єкта в даному стані. Кожна дія відображається у форматі:

<період виконання>/<назва дії>,

де поле <період виконання> може набувати наступних значень:

OnEntry - дія виконується під час того, як система входить у даний стан;

OnExit - дія виконується при виході з даного стану;

Do - дія виконується під час знаходження в даному стані;

OnEvent - дія виконується при настанні певної (зовнішньої) події.

Графічне представлення стану «Введення паролю» з трьома внутрішніми діями наведено на рис. 22. Перехід у даний стан ініціюється при введенні користувачем символів логіну. Для кожного введеного символу система зчитує строку логіну, додає додатковий символ до неї і зберігає її.

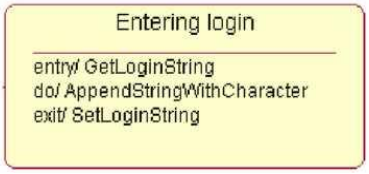


Рисунок 22 - Графічне представлення стану в середовищі RSA

У RSA існує можливість створити також один із специфічних станів:

1. Вхідний стан - стан, в якому знаходиться система (об'єкт) у початковий момент часу.
2. Вихідний стан - стан, в якому знаходиться система (об'єкт) в момент закінчення виконання певної послідовності дій.
3. Стан історії - стан, який запам'ятовує дані, що використовувалися при попередньому входженні системи в даний стан.

Рисунок 23 - Представлення початкового та кінцевого станів на statechart diagrams

Рис. 24 демонструє стан історії «Перевірка параметрів дзвінка». Після перевірки параметрів перед здійсненням виклику запам'ятовується номер абонента для можливості його повторного виклику в майбутньому.

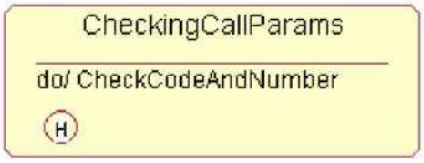


Рисунок 24 - Графічне представлення стану історії

Переходи (transitions) на statechart diagrams представлені стрілкою, що виходить з попереднього стану і входить у наступний. Кожен перехід має наступну специфікацію:

<тригер>(<параметри>) [<гранична умова>]/<дія>, де <тригер> - подія, що ініціює можливість переходу;

(<параметри>) - параметри події;

[<гранична умова>] - умова, необхідна для здійснення переходу;

<дія> - дія, що виконується у процесі переходу.

На statechart diagram можна задати два типи переходів:

1. Звичайний - перехід з одного стану в інший.
2. Рефлексивний - перехід із даного стану в цей же стан (зображається у вигляді петлі на графі).

На рис. 25 представлений фрагмент діаграми станів і переходів із застосуванням різних типів переходів для системи «Міні-АТС». Перехід із початкового стану у стан «Очікування» відбувається при ввімкненні системи, за умови її успішної ініціалізації. При цьому виконується дія Tone (подання тонового сигналу). Рефлексивний перехід для стану введення коду міста виникає при введенні нової цифри, параметр int d символізує код цифри.

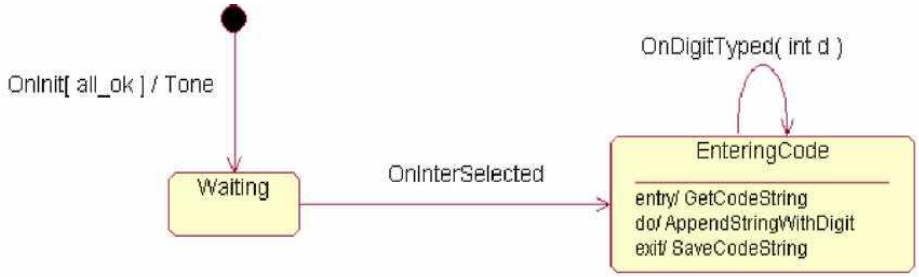


Рисунок 25 - Фрагмент діаграми станів та переходів для Міні-АТС

Приклади діаграм:

На рис. 26 зображено приклад діаграми станів та переходів для конкретного об’єкта (діалога авторизації певної системи). Рис. 6 представляє діаграму станів та переходів для Міні-АТС.

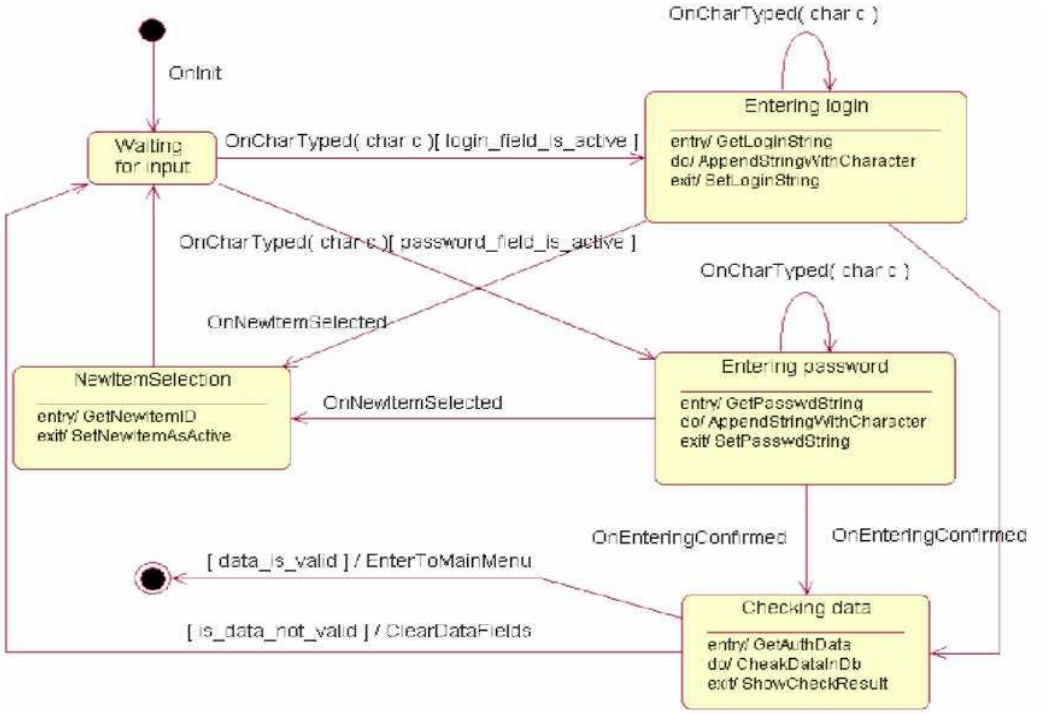


Рисунок 26 - Приклад діаграми станів та переходів для авторизації користувача в системі

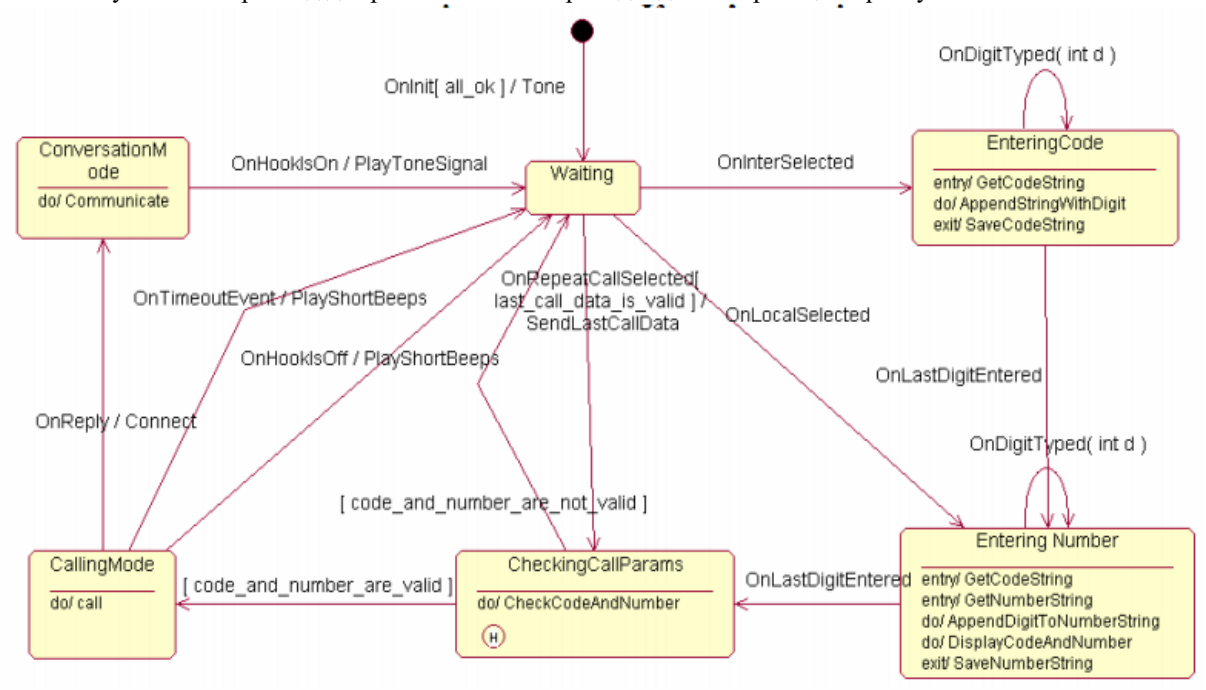


Рисунок 27 - Приклад діаграми станів та переходів для системи «Міні-АТС»

Завдання. Створити одну діаграму станів для опису процесу функціонування обраної системи в цілому і дві діаграми для конкретних елементів системи. Використовувати діаграму станів для авторизації користувачів забороняється.

Вимоги

1. Кожна діаграма повинна містити не менше 6 станів.
2. По можливості використати обидва типи переходів (звичайний і рефлексивний).
3. Для кожного переходу визначити хоча б одну з характеристик (тригер, гранична умова, дія).

Приклад виконання

I. *Діаграма станів для опису процесу функціонування системи в цілому.*

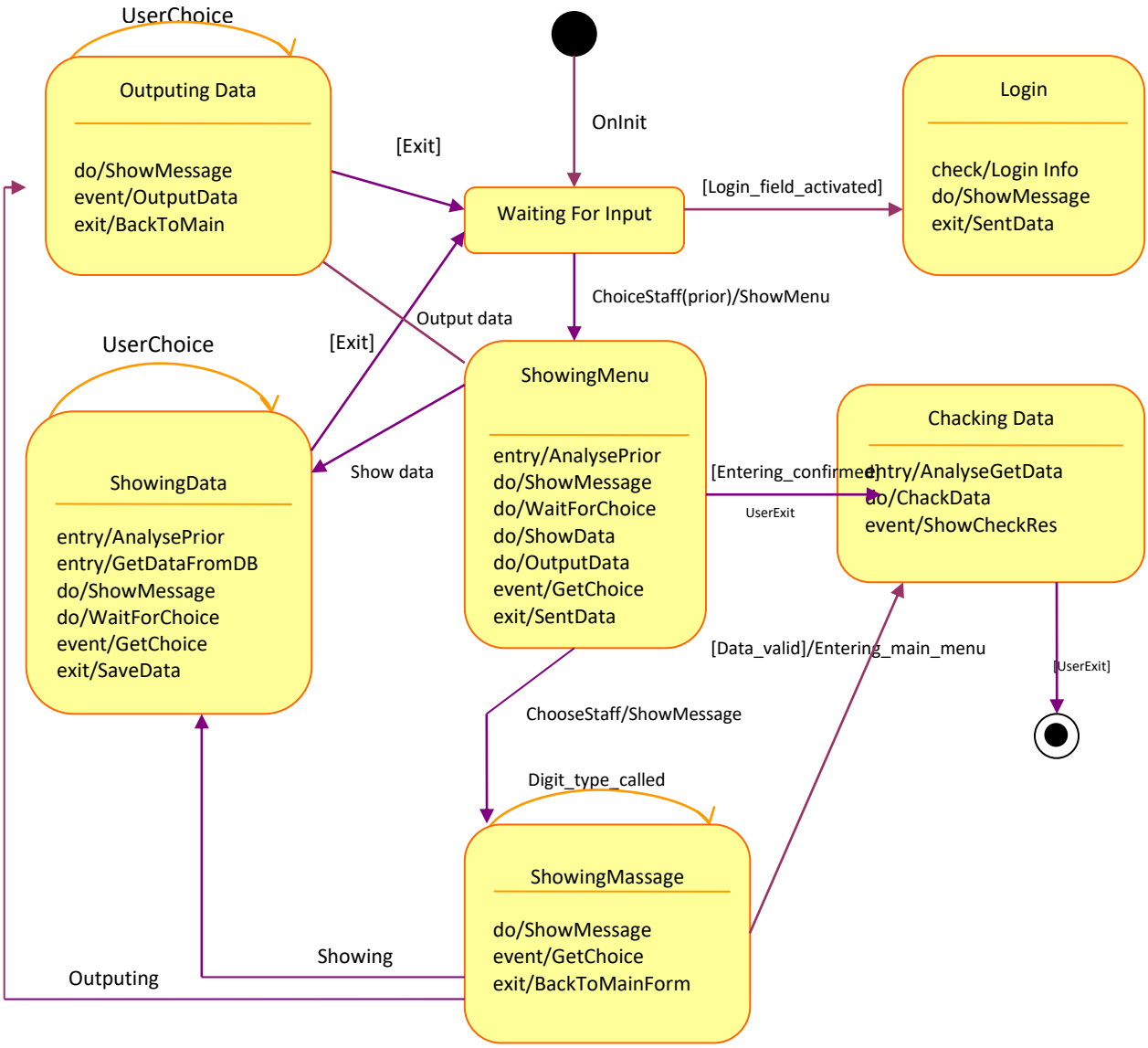


Рисунок28- Діаграма станів для опису процесу функціонування системи в цілому.

Опис подій в діаграмі 1:

Увійшовши в програму, ми отримуємо можливість авторизуватися в системі. Після вдалої авторизації, система переходить в стан роботи в «Меню». При цьому система щоразу при зверненні до неї очікую виконання певних дій, що супроводжується станом «Очікування». Якщо ж програма перебуває в стані існування меню системи, то перед нами виникає вибір: перейти в стан виводу повідомлення, в стан виведення даних назовні чи продемонструвати дані з БД. При цьому можна переходити у дані стани, а також повертатися з них у головне меню, тобто існує можливість переходу з даних станів в стан «Меню». З «Меню» можна перейти в стан передачі та «Перевіки» даних, їх збереження та виходу з системи(вихідний стан).

II. *Діаграма станів для опису прецеденту «Виконання запиту з пошуку даних про замовлення».*

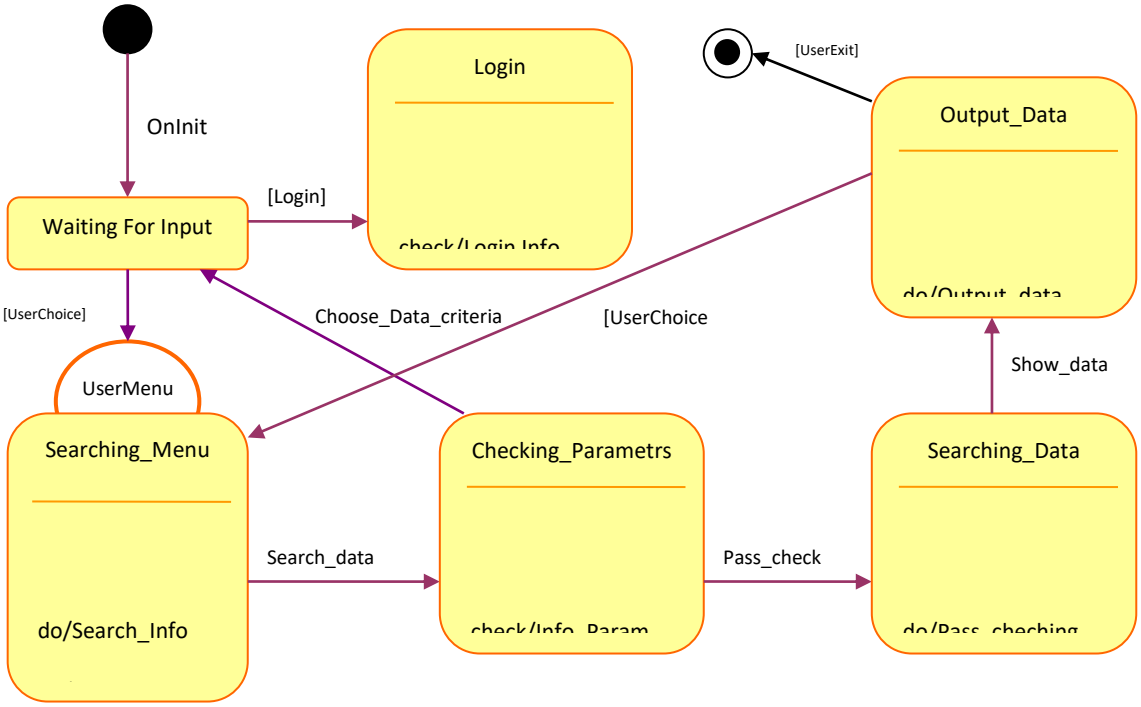
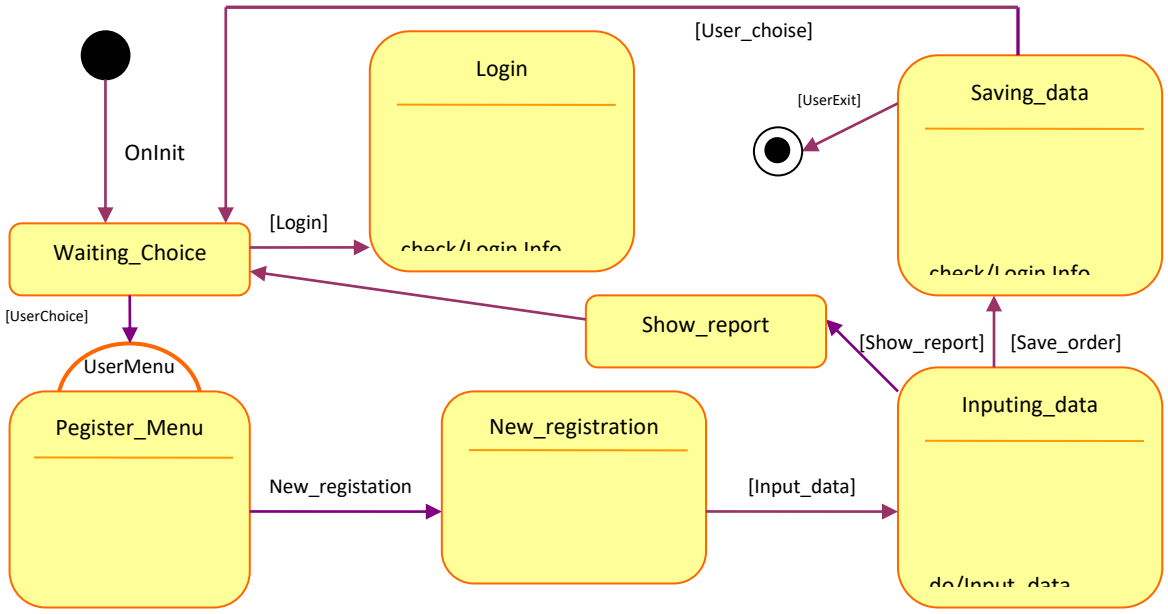


Рисунок29- Діаграма станів для прецеденту «Виконання запиту з пошуку даних про замовлення».

Опис подій в діаграмі 2:

На даній діаграмі продемонстровані стани та переходи між ними для виконання прецеденту «Виконання запиту з пошуку даних про замовлення». Початок діаграми передбачає авторизацію користувача і вибір виконання даного прецеденту. Цісля цього система переходить в стан очікування дії користувача. Якщо користувач продовжує виконання дій, то система з стану «очікування» переходить в стан роботи з виконанням запиту, а саме пошуку даних за критеріями. Для цього існує стан системи «Перевірка_параметрів» в якому відбувається валідація та перевірка даних та їх критеріїв пошуку. Після цього система переходить в стан пошуку даних. Уже після завершення пошуку даних, система переходить в стан виведення даних. Після виведення даних існує можливість продовження виконання пошуку даних (тобто повернення до стану системи «меню_пошуку»), або ж виходу з виконання даних дій системою.

III. Діаграма станів для прецеденту «Реєстрація нового замовлення».



Рисуно30 - Діаграма стану для варіанту використання «Реєстрація нового замовлення».
Опис діаграми 3:

На даній діаграмі продемонстровані стани системи при виконанні варіанту використання «Реєстрація нового замовлення». Система передбачає авторизацію та вхід в область роботи даного прецеденту. після чого виникає можливість проведення реєстрації нового замовлення. При виборі реєстрації система потрапляє в стан Реєстраційного меню. У «Новій реєстрації» проводиться перевірка та отримання реєстраційних форм від системи. Згодом можна перейти до стану «Введення даних для реєстрації», в якому перевіряються та вводяться дані для завершення реєстрації. При завершенні даного стану виникає можливість збереження даних або ж повернення до стану реєстраційної форми.

Питання до контролю

1. Для чого призначена діаграма станів та переходів (statechart diagram)?
2. Назвіть основні елементи statechart diagram.
3. Що таке стан? Перелічіть список тригерів для внутрішніх дій стану.
4. Перелічіть специфічні стани та наведіть їх призначення.
5. Для чого призначені переходи на statechart diagram?
6. Які характеристики властиві для кожного переходу? Чи всі вони обов'язкові?
7. Які типи переходів існують на діаграмі? Чим вони відрізняються?

Лабораторна робота № 7. ПОБУДОВА ДІАГРАМ ДІЯЛЬНОСТІ

Теоретичні відомості

Діаграми діяльності (activity diagrams) відображають послідовність дій, що виконується в процесі реалізації певного варіанта використання або функціонування системи в цілому. Діаграми діяльності є аналогом блок-схеми будь-якого алгоритму. Вони, як і діаграми станів та переходів, відображаються у вигляді орієнтованого графу, вершинами якого є дії, а ребрами - переходи між діями.

Діяльність (activity) є частковим випадком стану (state) без назви, який має одну вхідну подію (OnEntry action). Тому для кожної діяльності назва складається з дієслова та декількох пояснюючих слів, наприклад «Розрахувати заробітну платню» чи «Перевірити результати запиту». Графічне зображення діяльності «Перекласти слово» подано на рис. 31.

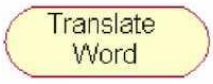


Рисунок 31- Графічне представлення елементу Activity на діаграмі діяльності

Події (events) на переходах діаграми діяльності не задаються, оскільки вважається, що перехід від однієї дії до іншої здійснюється безумовно. Гранична умова (guard condition) використовується лише для визначення дії, до якої переходить керування у випадку неоднозначності (рис. 32). Тобто, якщо з даної вершини на діаграмі діяльності можна перейти до декількох інших вершин для всіх переходів необхідно визначити граничну умову. Характеристика дії (action) для переходу також не має сенсу, оскільки всі дії на цій діаграмі представлені вершинами графу.

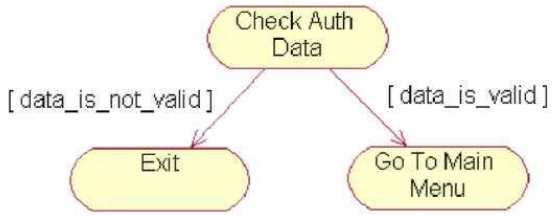


Рисунок 32 - Фрагмент діаграми діяльності для процесу авторизації

Для діаграми діяльності характерними є наступні спеціальні стани:

1. Початковий стан - аналогічний до діаграми станів та переходів.
2. Кінцевий стан - аналогічний до діаграми станів та переходів.
3. Стан прийняття рішення - стан, в якому здійснюється прийняття рішення про перенаправлення потоку управління до одного зі станів, пов'язаних із даним станом.
4. Стан синхронізації - стан, в якому здійснюється розділення загального потоку управління на декілька гілок (чи навпаки, декілька гілок поєднуються в єдиний потік).

Спеціальні стани прийняття рішення та синхронізації представлені на рис. 33а та 33б відповідно.

Рисунок 33 - Графічне представлення спеціальних станів діаграми діяльності

Розглянемо застосування спеціальних станів на конкретному прикладі (варіант використання «Перекласти слово» для електронного словника). На рис. 34 показано повний вигляд діаграми діяльності. На початковому етапі перекладу обирається словник, користувач вводить слово та система робить його переклад, далі з використанням стану прийняття рішення визначається чи обрані додаткові опції перекладу. У випадку, якщо опцій не обрано, система переходить до показу результату перекладу. В іншому випадку, потік керування розподіляється на дві гілки, кожна з яких виконує певну дію (отримати

транскрипцію слова та список синонімів відповідно). Після закінчення виконання обох операцій дві гілки поєднуються в єдиний потік і здійснюється показ результату. Потім визначається чи потрібен друк для отриманої інформації, і у випадку необхідності вона друкується.

Також діаграми діяльності використовуються для відображення послідовності дій при моделюванні бізнес-процесів. При цьому використовується додатковий елемент діаграми, який має назву swimlane. Swimlane в дослівному перекладі означає дорожка плавального басейну (за аналогією з графічним відображенням). В якості swimlanes на діаграмі можуть виступати фізичні особи, групи осіб, відділи підприємства, чи навіть окремі організації. Розглянемо діаграму діяльності зі swimlanes для спрощеного варіанта бізнес-процесу «Розробка програмного забезпечення» (рис. 35).

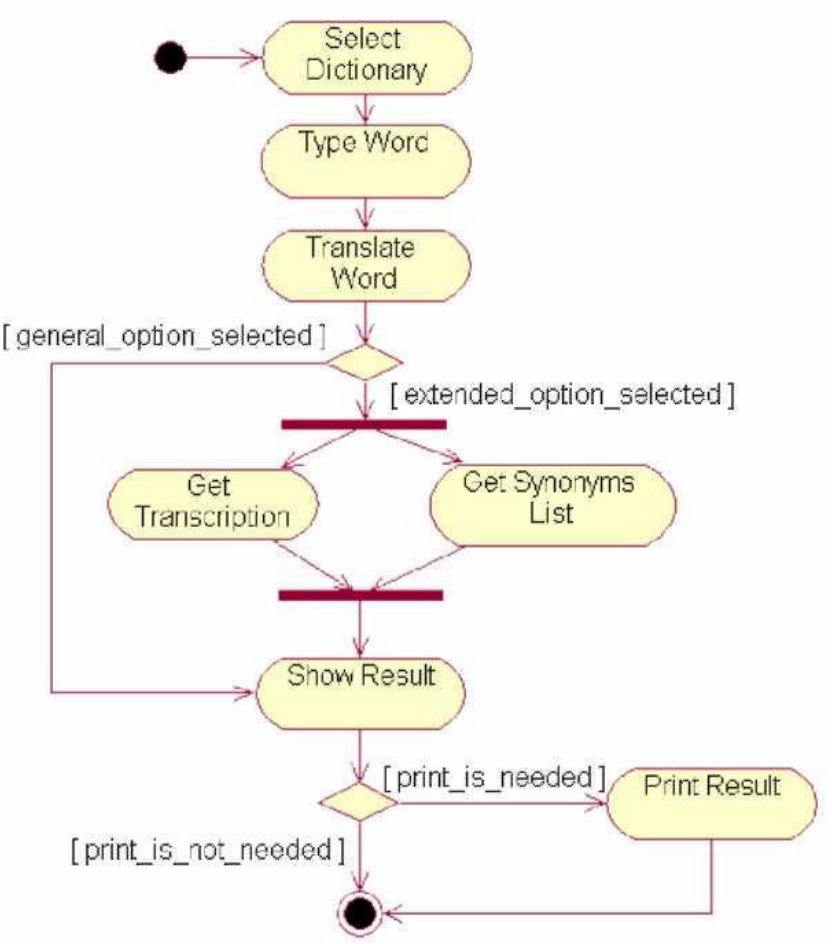


Рисунок 34 - Діаграма діяльності для перекладу слова електронним словником

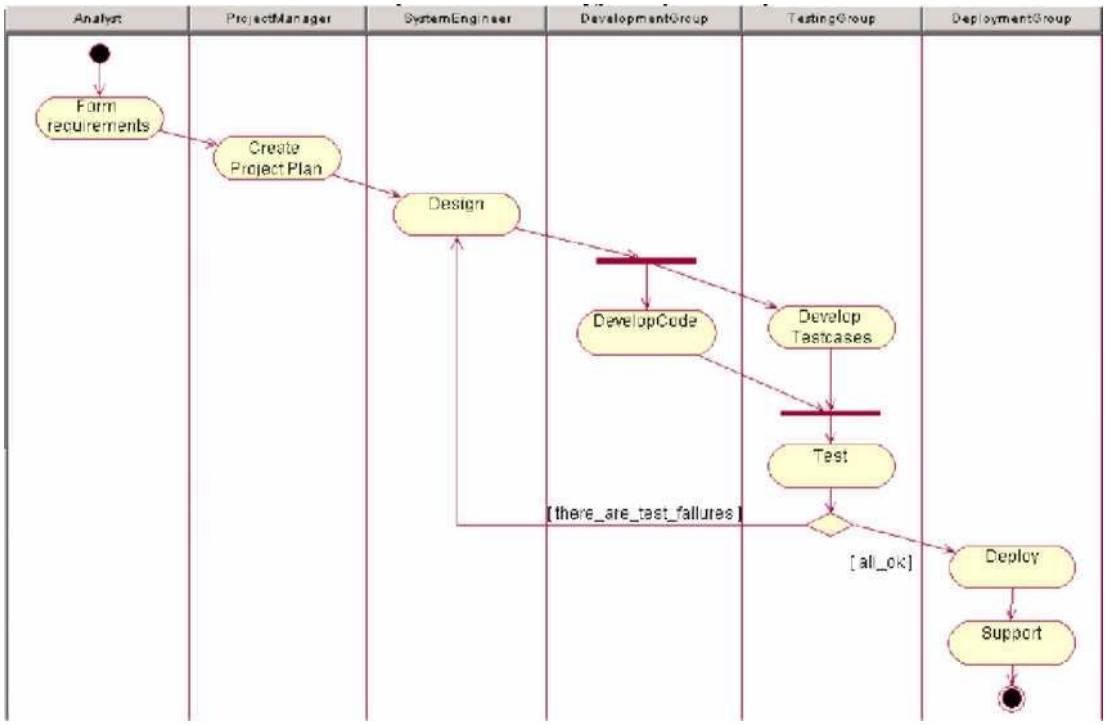


Рисунок 35 - Використання діаграми діяльності для відображення бізнес-процесів

В якості swimlanes в даній діаграмі виступають наступні особи (групи осіб):

1) аналітик, який розробляє вимоги до проекту;
2) керівник проекту, який складає план виконання робіт;
3) системний інженер, що проектує систему;
4) група розробників, які створюють програмний код;
5) група тестувальників, які формують варіанти тестування та тестують створену систему;
6) група впровадження, яка поставляє систему кінцевому користувачу та здійснює підтримку.

Завдання. Побудувати 3 діаграми діяльності для окремих варіантів використання системи.

Вимоги

1. Кожна діаграма повинна містити не менше 6 діяльностей.
2. При побудові кожної діаграми використовувати стани прийняття рішення та синхронізації.

Приклад виконання

I. Діаграма діяльності для варіанту використання «Реєстрація замовлення».

Дана діаграма діяльності ілюструє послідовність дій, що виконуються в процесі функціонування варіанта використання «Реєстрація замовлення». Перше, що повинне існувати в виконанні даного варіанту використання – існування замовлення(можливість отримати замовлення).

Отримавши замовлення, користувач системи переходить до внесення даних в систему для створення звіту та збереження даних в БД. Також дані повинні бути перевірені на відповідність і критичність в існуванні замовлення. Тут виникає проблема в можливості невірних вхідних даних або ж неможливості подальшого виконанні даного варіанту. Тому передбачено розгалуження умови. В разі, якщо ж з інформацією в замовленні все добре, користувач додає зі списку ті роботи, які необхідні і можливі у виконанні над цим замовленням. Після завершення всіх дій дані про замовлення зберігаються і формується звіт. І діяльність даного ВВ переходить на завершальний етап.

Рисунок36 - Діаграма діяльності для варіанту використання «Реєстрація замовлення».

II. Діаграма діяльності для варіанту використання «Внесення замовлень в чергу на обслуговування».

Дана діаграма ілюструє діяльність системи при виконанні варіанту використання «Внесення замовлень в чергу на обслуговування». Для виконання потрібно мати список з замовленнями в списку замовлень що поступили. Далі ці ж замовлення переміщуються в чергу замовлень та видаляються зі списку очікуючих на виконання цих дій. Якщо ж замовлення неможливо перемістити в чергу то ніяких дій з ним не відбувається до вияснення обставин, через які це не відбулося. На кінцевому етапі зберігаються зміни в системі та БД замовлень і закінчується виконання даного прецеденту.

Рисунок37 - Діаграма діяльності для варіанту використання «Внесення замовлень в чергу на обслуговування».

III. Діаграма діяльності для варіанту використання «Виконання запиту пошуку даних про замовлення».

Рисунок38 - Діаграма діяльності для варіанту використання «Виконання запиту з пошуку даних про замовлення»

Дана діаграма діяльності ілюструє покрокове виконання і переходи між діяльністю для варіанта використання «Виконання запиту з пошуку даних про замовлення». Запит розпочинається з звернення в

систему та формування запиту про пошук даних про замовлення в системі(в БД замовлень). Далі система обробляє даний запит та перевіряє його на валідність. Якщо ж знайдено записи за даним запитом то вони виводяться користувачеві. Якщо ж ні, то виводиться повідомлення про відсутність таких даних. Далі слідує завершення виконання діяльності.

Питання до контролю

1. Для чого призначена діаграма діяльності (activity diagram)? Назвіть її основні елементи.
2. У чому відмінність між діаграмою діяльності і діаграмою станів та переходів?
3. Що таке діяльність? Чим діяльність відрізняється від стану?
4. Які характеристики властиві для переходу на діаграмі діяльності.
5. Що таке стан прийняття рішення?
6. Що таке стан синхронізації?
7. Для чого використовується елемент swimlane?

ЛІТЕРАТУРА

1. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на С++. Пер. с англ. – М.: СПб.: “Издательство БИНОМ”-“Невский Диалект”, 2001.-560 с.
2. Джекобсон А., Буч Г., Рамбо Д. Унифицированный язык моделированя UML. Руководсто пользователя. - М.: Изд. дом «Вильямс», 2004.–460с
3. Джекобсон А., Буч Г., Рамбо Д. Рациональный унифицированный подход. - М.: Изд. дом «Вильямс», 2003.–600с
6. Советов Б., Яковлев С. Моделирование систем. Учебник . М.: Юрайт-Издат, 2012
7. Гвоздева Т.В. Проектирование информационных систем: учеб. пособие / Т.В. Гвоздева, Б.А. Баллод. – Ростов н/Д: Феникс, 2009. –508 с.
8. Гама Х. UML проектирование систем реального времени параллельных и распределенных приложений. М.: ДМК Пресс, 2011. - 704 с.

Лабораторний практикум

до дисципліни

«**Моделювання та аналіз програмного забезпечення**»

Упорядники:

М.Р. Петрик,
О.Ю. Петрик

Відповідальний за випуск М.Р. Петрик

Видавництво ТНТУ ім. Івана Пулюя

46000, Тернопіль, вул. Руська 56