ЛАБОРАТОРНА РОБОТА 1

ВИВЧЕННЯ АРХІТЕКТУРИ, ВІЗУАЛЬНИХ ІНТЕРФЕЙСІВ ТА ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ CASE-СИСТЕМИ VISUAL PARADIGM

Мета – вивчити такі запитання:

- призначення системи, склад та перелік її основних компонентів;
- вивчення основних понять: пакети, класи, відношення (зв'язки) компоненти, стереотипи, представлення програмної системи як сукупності UML-діаграм;
- інтерфейси та можливості пакета UML Diagrams.

1.1. Загальні відомості

Призначення системи, склад та перелік її основних компонентів

Visual Paradigm для UML – це інструмент для проектування програмного забезпечення (ПЗ) будь-якої складності, який підтримує UML 2, SysML і Business Process Modeling Notation (BPMN) від Object Management Group (OMG). Дистрибутив пакета Visual Paradigm можна вільно отримати на офіційному Web-сайті розробника [1].

На цій сторінці необхідно обрати продукт Visual Paradigm for UML (VP-UML), а саме безкоштовну версію (community edition), завантажити її та пройти процедуру реєстрації для отримання ліцензійного ключа. Після інсталяції необхідно імпортувати ліцензійний ключ до системи.

VP-UML надає розробнику ПЗ засоби для аналізу і моделювання програмної системи (ПС), що розробляється. Крім підтримки моделювання різних аспектів функціонування ПС, він забезпечує генерацію коду, а також побудову звітів та деякі інші можливості. Однією з корисних особливостей VP-UML є функція зворотного інжинірингу проекту (reverse code engineering), тобто генерування UML-діаграми з існуючого програмного коду, але ці можливості є доступними лише в повній, комерційній версії системи.

До основних компонентів інтерфейсу CASE-засобу VP-UML можна віднести такі (рис. 1.1): 1) головне меню; 2) панель UML – вона, окрім стандартних функцій (створення, відкриття, друку проекту тощо), надає можливість вибрати тип нової ді-

аграми; 3) вікно документації – це область для ведення документації проекту; 4) браузер проекту – містить дерево проекту: наявні діаграми та пакети; 5) робоча область – у ній безпосередньо будуються діаграми; 6) панель інструментів діаграми – це набір візуальних засобів для побудови окремих діаграм; 7) вікно виводу повідом- лень – область, де відображуються необхідні системні повідомлення.

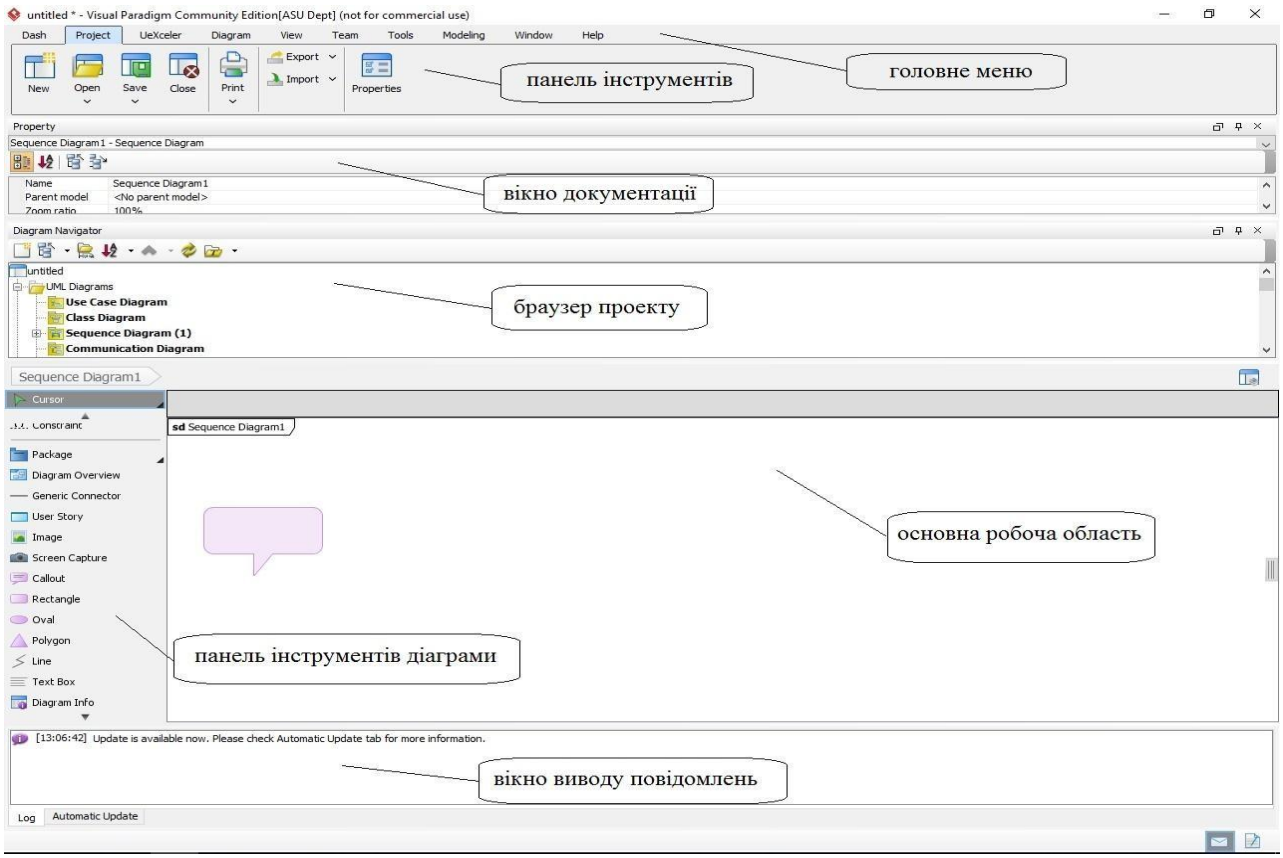

Рисунок 1.1 – Загальний вигляд інтерфейсу користувача CASE-засобу VP-UML

Вивчення основних понять uml-діаграм: пакети, класи, відношення (зв'язки), компоненти, стереотипи

У цьому циклі лабораторних робіт використовується нотація UML-діаграм версії UML 2.0 [2].

Конструктивні блоки UML

Сутності (entity) є основою всіх типів UML-моделей. Прив'язку сутностей одну до одної забезпечують *відношення* (relationship), а діаграми групують сутності у певні набори (конфігурації). У UML подані чотири типи сутностей: *структурні* (struc-

tural), *поведінкові* (behavioral), сутності, що угруповують інші сутності (grouping entity), та анотації (annotation). Графічне зображення окремих типів сутностей, прийняте в UML, наводиться нижче.

Єдиним представником сутності, що угруповує інші сутності, є *пакет* (package). Пакет – це механізм загального призначення для організації елементів у вигляді єдиної групи сутностей. Структурні, поведінкові та навіть інші що групують сутності, можуть бути поміщені всередині певного пакета.

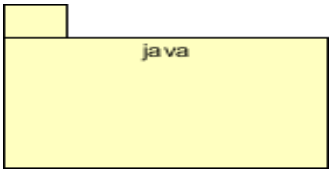

Рисунок 1.2 – Позначення сутності – пакета

Анотації є пояснювальною та коментованою частиною UML. Єдиним її типом є *примітка* (note). Примітка з'єднується пунктирною лінією із сутністю, до якої вона належить.

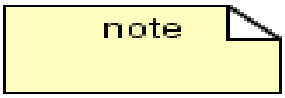

Рисунок 1.3 – Позначення примітки

Клас (class) у мові UML використовується для визначення множини об'єктів, які мають однакову структуру, поведінку та відношення з об'єктами з інших класів. Графічно клас зображується у вигляді прямокутника, який додатково може бути розділений горизонтальними лініями на розділи або секції (рис. 1.4). У цих розділах указуються ім'я класа, атрибути (змінні) та операції (методи).

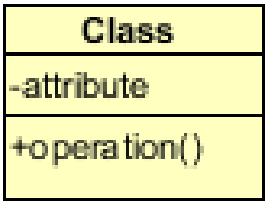

Рисунок 1.4 – Позначення класу

Атрибут (attribute) – у другій зверху секції прямокутника класу записуються його атрибути (attributes) або властивості. У мові UML прийнята визначена стандартизація запису атрибутів класу, яка підпорядковується деяким синтаксичним правилам. Кожному атрибуту класу відповідає окремий рядок тексту, який складається із квантора видимості атрибута, імені атрибута, його кратності, типу значень атрибута й можливо, його початкового значення:

<квантор видимості><ім'я атрибута>[кратність]:

<тип атрибута> = <початкове значення>{строка-властивість}.

Квантор видимості може набувати одне із трьох можливих значень та, відповідно, відображається за допомогою спеціальних символів:

1. Символ «+» означає атрибут з областю видимості типу загальнодоступний (public). Атрибут із цією областю видимості доступний або видний із будь-якого іншого класу пакета, в якому визначена діаграма.
2. Символ «#» означає атрибут з областю видимості типу захищений (protected). Атрибут із цією областю видимості недоступний або невидимий для всіх класів, за виключенням підкласів цього класу.
3. Знак «-» означає атрибут з областю видимості типу закритий (private). Атрибут із цією областю видимості недосяжний або невидимий для всіх класів без виключення.

Метод (method) – у третій зверху секції прямокутника класу записуються операції або методи класу. Метод являє собою деякий сервіс, що кожний екземпляр класу надає за умови виклику цього методу. Сукупність операцій характеризує функціональний аспект поведінки класу. Запис операцій класу у мові UML також стандартизований та підпорядкований певним синтаксичним правилам. При цьому кожній

операції класу відповідає окремий рядок, який складається із квантора видимості операції, імені операції, типу даних, значення, яких повертає операція і, можливо, строка-властивостей цієї операції, а саме:

<квантор видимості>< ім'я атрибута>(список параметрів):

<вираз типу повертального значення>{строка-властивість}.

Квантор видимості, як і у випадку атрибутів класу, може приймати одне із трьох можливих значень та, відповідно, відображається за допомогою спеціального символу. Символ «+» визначає операцію із областю видимості типу загальнодоступний (public). Символ «#» визначає операцію із областю видимості типу захищений (protected). І, нарешті, символ «-» використовується для визначення операції з областю видимості типу закритий (private).

Крім внутрішньої побудови або структури класів, на відповідній діаграмі вказуються різноманітні *відношення* або *зв'язки* (relationship) між класами. При цьому сукупність типів таких відношень в UML визначена семантикою типів цих відношень. Базовими відношеннями або зв'язками у мові UML є:

1) залежності (dependency relationship);
2) асоціації (association relationship);
3) узагальнення (generalization relationship);
4) агрегації (aggregation relationship);
5) композиції (composition relationship) як окремий випадок агрегації.

Відношення залежності у загальному випадку вказує на деяке семантичне відношення між двома елементами моделі або між двома множинами таких елементів, яке не є відношенням асоціації, узагальнення або агрегації. Воно стосується тільки самих елементів моделі і не вимагає множина окремих прикладів для пояснення свого сенсу. Відношення залежності використовується у такій ситуації, коли деяка зміна одного елемента моделі може потребувати зміни іншого залежного від нього елемента моделі.

Відношення залежності графічно зображується пунктирною лінією між відповідними елементами із стрілкою на одному з її кінців («->» або «<-»). На діаграмі класів це відношення зв'язує окремі класи між собою, при цьому стрілка направлена від класу-клієнта залежності до незалежного класу або класу-джерела (рис. 1.5). На

ньому зображено два класи: клас_А і клас_Б, при цьому клас_Б є джерелом деякої залежності, а клас_А – клієнтом цієї залежності.

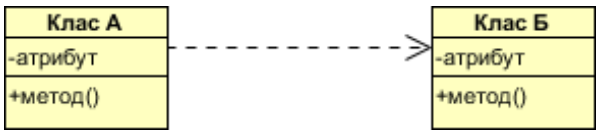

Рисунок 1.5 – Позначення відношення залежності

Відношення асоціації відповідає наявності деякої функціональної залежності між класами. Це відношення позначається суцільною лінією з додатковими спеціальними символами, які характеризують окремі властивості конкретної асоціації. Як додаткові спеціальні символи можуть використовуватися імена асоціації, а також імена і кратність класів-ролей асоціації. Ім'я асоціації є необов'язковим елементом її позначення. Якщо воно задане, то записується із заголовної (великої) букви поряд з лінією відповідної асоціації. Найбільш простий випадок цього відношення – це бінарна асоціація.

Нехай у деякій UML-моделі є клас «А» (співробітник) і клас «Б» (компанія). Тоді прикладом відношення асоціації між ними буде твердження «Розробник працює в Компанії» і відповідна діаграма класів показана на рис. 1.6.

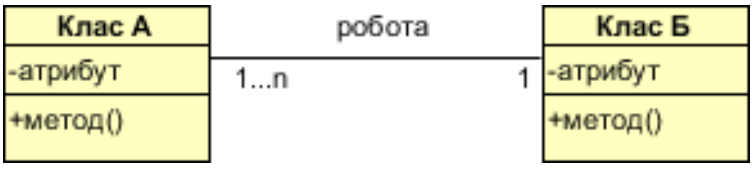

Рисунок 1.6 – Позначення відношення асоціації

Відношення агрегації має місце між декількома класами в тому випадку, якщо один з класів є деякою сутністю, що включає як складові частини певні інші сутнос- ті.

Це відношення має фундаментальне значення для опису структури складних систем, оскільки застосовується для представлення системних взаємозв'язків типу «частина-ціле» (part of). Розкриваючи внутрішню структуру системи, відношення агрегації показує, з яких компонентів складається система і як вони зв'язані між собою. З погляду моделі окремі частини системи можуть виступати як у вигляді елементів, так і у вигляді підсистем, які, у свою чергу, теж можуть утворювати складні компо- ненти або підсистеми. Це відношення за своєю суттю описує декомпозицію або роз- биття складної системи на простіші складові частини, які також можуть бути піддані декомпозиції, якщо в цьому виникне необхідність у подальшому. Наприклад: розподіл комп'ютера на складові частини – системний блок, монітор, клавіатура, мишка (рис. 1.7). В UML 2.0 визначено дві форми відношення «частина-ціле»: загальна (агрегація) та приватна (композиція).

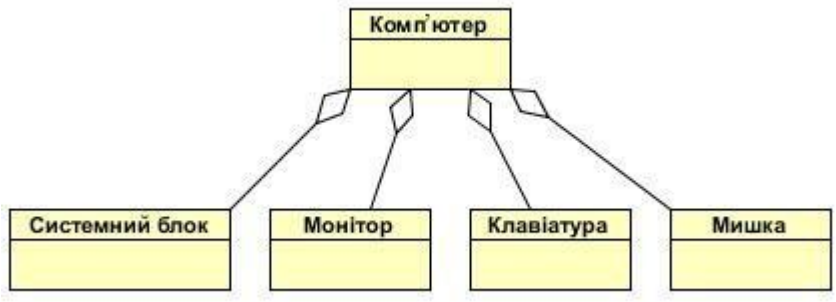

Рисунок 1.7 – Позначення відношення агрегації

Композиція (composition) – це окремий випадок агрегації, що характеризується двома додатковими обмеженнями. Складова частина може належати не більш ніж одному агрегату. Таким чином, у композиції малося на увазі, що частини належать цілому, тому видалення об'єкта-агрегата автоматично викликає видалення всіх його складових, якщо він утворює їхню композицію. Для позначення композиції використовується невеликий зафарбований ромбик, який ставиться поряд з класом-агрегатом (для агрегації, що не є композицією, використовується не зафарбований ромбик).

Наприклад: компанія складається з відділень, які, у свою чергу, складаються з відділів. Компанія побічно є композицією відділів (рис. 1.8).

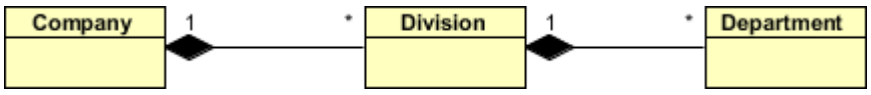

Рисунок 1.8 – Позначення відношення композиції

Відношення узагальнення є таксономічнім відношенням між більш загальним елементом (батьком або предком) і більш спеціальним елементом (дочірнім або нащадком). Це відношення може використовуватися для представлення взаємозв'язків між пакетами, класами, варіантами використання і іншими елементами мови UML. На діаграмі класів це відношення описує ієрархічну побудову деяких класів і наслідування їх властивостей (через відповідні атрибути) та поведінки (за рахунок методів). При цьому передбачається, що клас-нащадок володіє всіма властивостями і поведінкою класу-предка, а також має свої власні властивості і поведінку, які відсутні у класу-предка. На діаграмах відношення узагальнення позначається суцільною лінією з трикутною стрілкою на одному з кінців (рис. 1.9). Стрілка вказує на узагальнюючий клас (клас-предок або суперклас), а її відсутність – на спеціалізований клас (клас-нащадок або підклас).

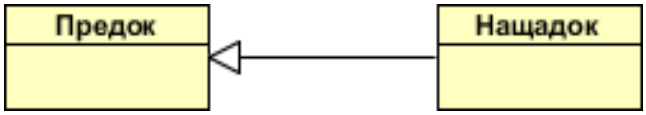

Рисунок 1.9 – Позначення відношення узагальнення (наслідування)

Компонент (component) – спеціальний термін, який застосовується в мові UML для представлення окремої сутності на фізичному рівні моделювання ПС. Компо- нент реалізує деякий набір інтерфейсів і використовується для загального позна- чення варіантів можливої програмної реалізації моделі, наприклад, це: бінарний код, що виконується (*.exe – файл, java-аплет і інш.), бібліотека, що динамічно підк-

лючається (*.ddl – файл), програмний скрипт, написаний деякою мовою опису сценаріїв (*.asp, *.php – файли) та ін. Для графічного представлення компонента у UML 2.0 використовується спеціальний символ – прямокутник, що містить у собі праворуч дрібніший прямокутник зі вставленими у нього двома ще дрібнішими прямокутниками (рис. 1.10). Всередині цього прямокутника записується ім'я компонента і, можливо, деяка додаткова інформація.

Рисунок 1.10 – Позначення компонента (фізичної сутності)

Стереотип (stereotype) – це механізм утворення нового типу (виду) класів в мета-моделі UML. На рис. 1.11 зображено існуючий список наявних стереотипів певного класу.

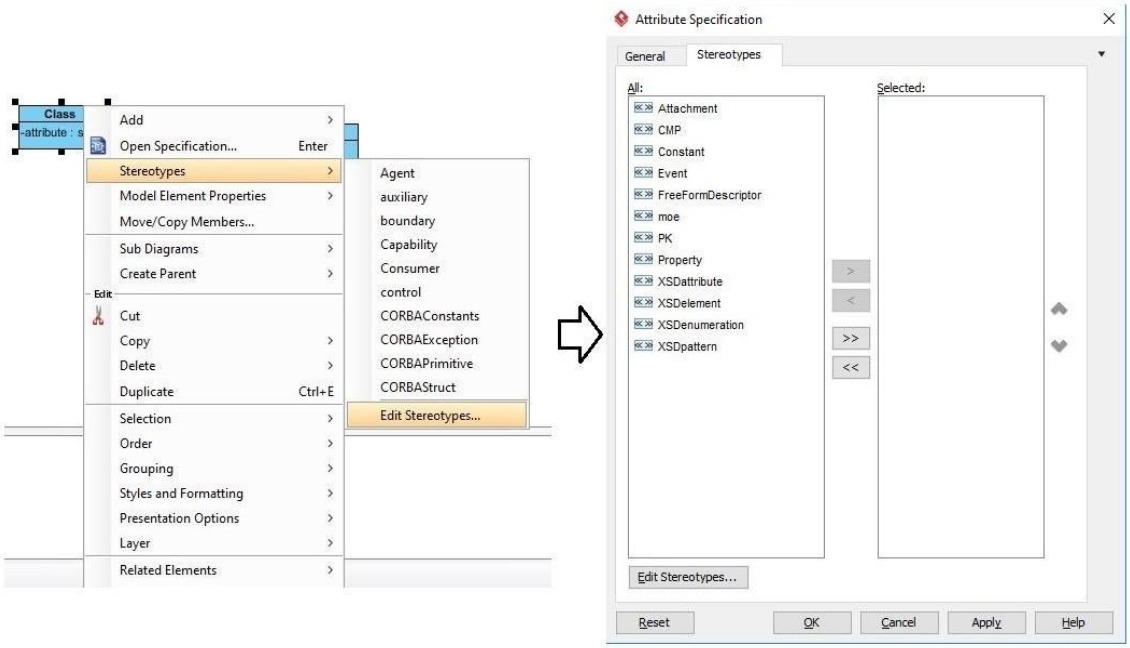

Рисунок 1.11 – Приклад списку наявних стереотипів

Для того щоб додати потрібний стереотип у список, необхідно натиснути правою кнопкою миші на клас. Далі на вкладці *Stereotypes* натиснути на кнопку *Edit Stereotypes* та у вікні, що з'явилося, натиснути на кнопку *Add.* Потім необхідно ввести назву нового стереотипу, також є можливість встановлення додаткових параметрів (таких, як тип шрифту, логотип і та ін.). Для додавання натискаємо *OK*. На рис. 1.12 наведений приклад компонента *Server Machine* з доданим стереотипом *Intel*.

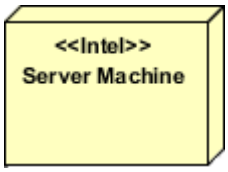

Рисунок 1.12 – Позначення компонента, що має стереотип

У середовищі VP / UML існує 13 типів діаграм у межах пакета UML Diagrams, а саме:

1. UseCase diagram (діаграма прецедентів).
2. Class diagram (діаграма класів).
3. Sequence diagram (діаграма послідовностей).
4. Communication diagram (діаграма комунікації).
5. State machine diagram (діаграма кінцевого автомату).
6. Activity diagram (діаграма активності).
7. Component diagram (діаграма компонентів).
8. Deployment diagram (діаграма розгортання).
9. Package diagram (діаграма пакетів).
10. Object diagram (діаграма об'єктів).
11. Composite structure diagram (діаграма композитної структури).
12. Timing diagram (часова діаграма).
13. Interaction overview diagram (діаграма огляду взаємодії).

Усі ці діаграми у VP / UML розбиті на 4 різних групи (пакети), що знаходяться на вкладці UML головного меню системи (рис. 1.13):

- моделювання прецедентів (Use Case Modeling);
- структурне моделювання (Structural Modeling);
- моделювання поведінки (Behavioral Modeling);
- моделювання архітектури (Architectural Modeling).

За допомогою цих діаграм можливо розробляти моделі ПЗ на концептуальному, логічному та фізичному рівнях проектування відповідної ПС.

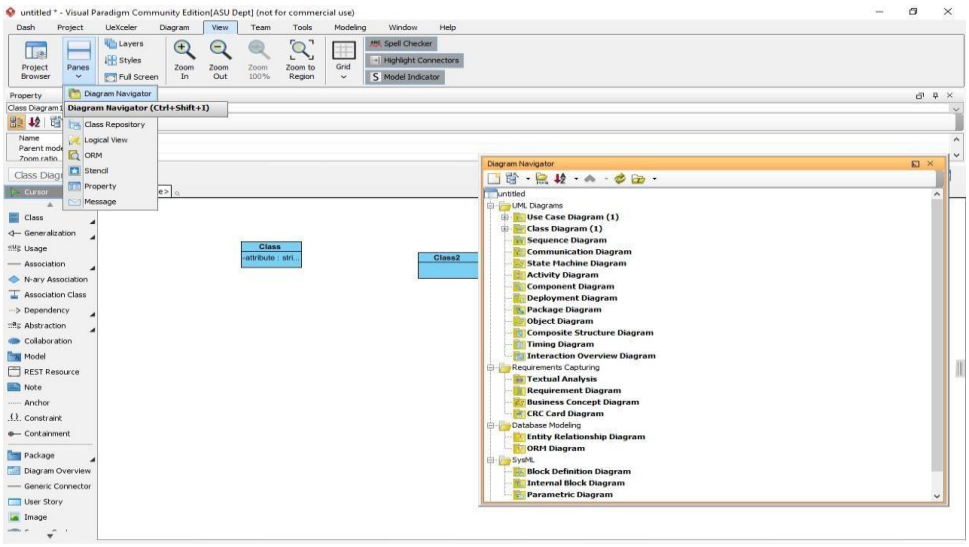

Рисунок 1.13 – Основні групи (пакети) UML-діаграм

Пакет Use Case Modeling

Цей пакет може містити одну або сукупність діаграм варіантів використання (або прецедентів – Use Case diagram), які використовуються для концептуального рівня проектування ПС. Приклад інтерфейсу цього пакета показано на рис. 1.14. На панелі інструментів цього пакета присутні специфічні для цього типу діаграм візуальні інструменти: *Актори, Варіанти використання, Асоціації* тощо.

Пакет Structural Modeling

Цей пакет може містити одну діаграму або сукупність діаграм логічного рівня проектування ПС, які забезпечують моделювання її структурних (тобто статичних) аспектів (рис. 1.15).

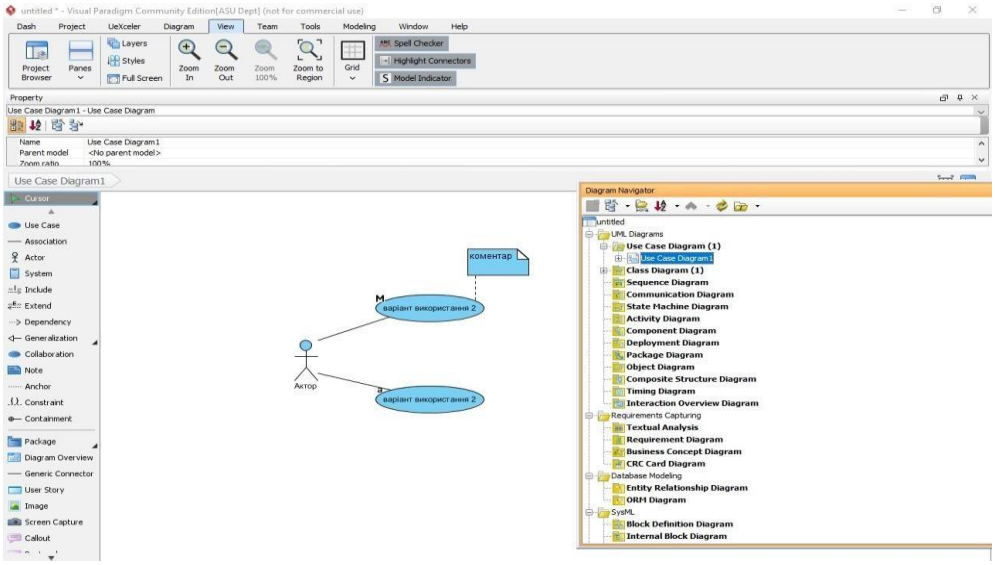

Рисунок 1.14 – Приклад інтерфейсу в пакеті «Use Case Diagram»

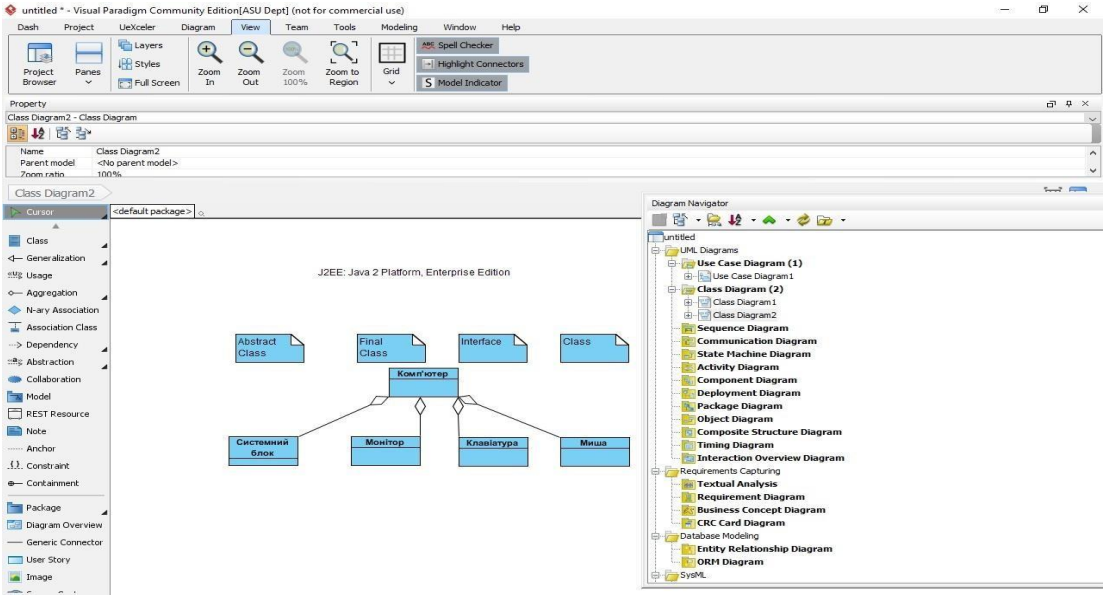

Рисунок 1.15 – Приклад інтерфейсу в пакеті ”Structural Modeling”

Це такі діаграми як

– діаграма класів (Class diagram);

– діаграма композитної структури (Composite structure diagram),

діаграма об’єктів (Object diagram). Щоб додати до логічного рівня нову діаграму, потрібно на відповідному пакеті натиснути праву кнопку мишки і у меню, яке

з’явиться, обрати опцію New Diagram та обрати потрібну діаграму. Ці діаграми будуть більш детально розглянуті у лабораторних роботах 2–5.

Пакет Behavioral Modeling

Діаграми цього пакета описують динамічні аспекти програмної системи, що моделюється: процес функціонування елементів системи, включаючи їхні методи та взаємодію між ними, а також процес зміни станів окремих елементів і системи у цілому. До нього входять такі діаграми: Sequence diagram (діаграма послідовності), Communication diagram (діаграма комунікації), Activity diagram (діаграма активності), State Machine diagram (діаграма кінцевого автомату), Timing diagram (діаграма часу), Interaction Overview diagram (діаграма обзору взаємодії) (рис. 1.16).

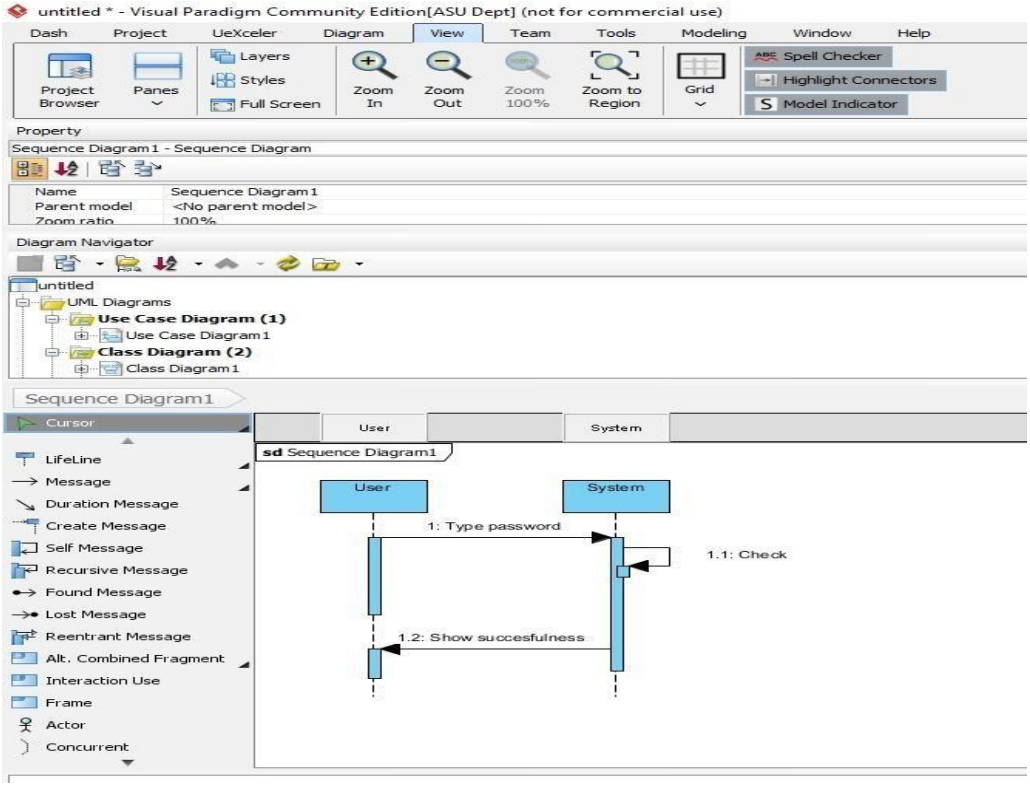

Рисунок 1.16 – Приклад інтерфейсу в пакеті «Behavioral Modeling»

Пакет Architectural Modeling

На цьому рівні містяться такі діаграми: Component diagram (діаграма компонентів), Deployment diagram (діаграма розгортання), Package diagram (діаграма пакетів) апаратного і програмного забезпечення системи на етапі її фізичного проектування.

Для того щоб створити специфікацію будь-якого елемента цих діаграм, треба натиснути правою кнопкою мишки на потрібному елементі діаграми та вибрати пункт Open Specification у списку, або ж нажати лівою кнопкою мишки на елемент та натиснути клавішу Enter (рис.1.17). Пакети Use Case Modeling, Structural Model- ing та Architectural Modeling описують виключно статичні аспекти побудови про- грамної системи, що моделюється, на відміну від пакета Behavioral Modeling, який описує динамічні аспекти цієї ПС.

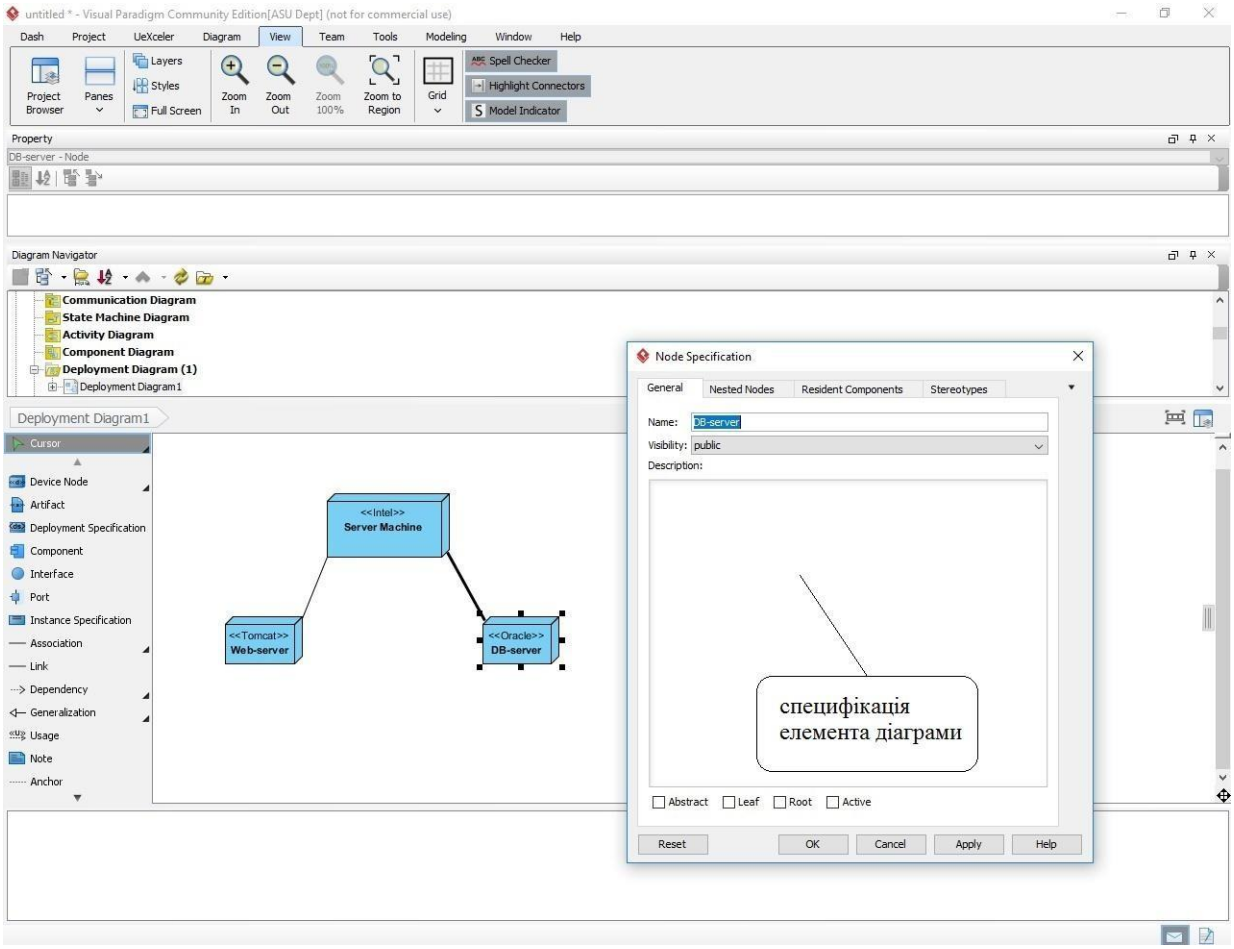

Рисунок 1.17 – Приклад інтерфейсу в пакеті «Architectural Modeling»

1.2. Виконання лабораторної роботи

1) У середовищі системи VP-UML розглянути всі його основні режими та функ- ціональні можливості;

2) створити власну стислу інструкцію для роботи з VP-UML;

3) обрати певну предметну область розробки ПС (за індивідуальним завданням викладача).

1.3. Результати виконання роботи та оформлення звіту

Зміст звіту:

1) титульний аркуш;
2) короткі теоретичні відомості;
3) результати практичного виконання, висновки та рекомендації.

Контрольні запитання

1. Назвіть призначення та основні компоненти архітектури CASE-засобу Visual Paradigm / UML.
2. Назвіть та поясніть, для чого застосовують основні конструктивні елементи (абстракції) UML.
3. Навести типи зв'язків в UML–діаграмі класів і надайте їх стислу характеристику.
4. В чому полягає різниця між відношеннями агрегації та узагальнення? Наведіть конкретні приклади цих типів відношень у деякій ПрО.
5. Що є спільного і у чому полягає різниця між відношеннями агрегації та композиції? Наведіть приклади цих типів відношень.
6. Які основні пакети існують у середовищі Visual Paradigm та для чого використовуються?
7. Які діаграми містяться в пакеті Use Case Modeling?
8. Які діаграми містяться в пакеті Structural Modeling?
9. Які діаграми містяться в пакеті Behavioral Modeling?
10) Які діаграми містяться в пакеті Architectural Modeling?
11) Назвіть існуючі рівні проектування ПЗ та види UML-діаграм, що використо- вуються для моделювання властивостей ПС на кожному з них.
12) Які пакети (діаграми) застосовуються для моделювання статичних аспектів побудови ПС, і які – для моделювання її динамічних характеристик?