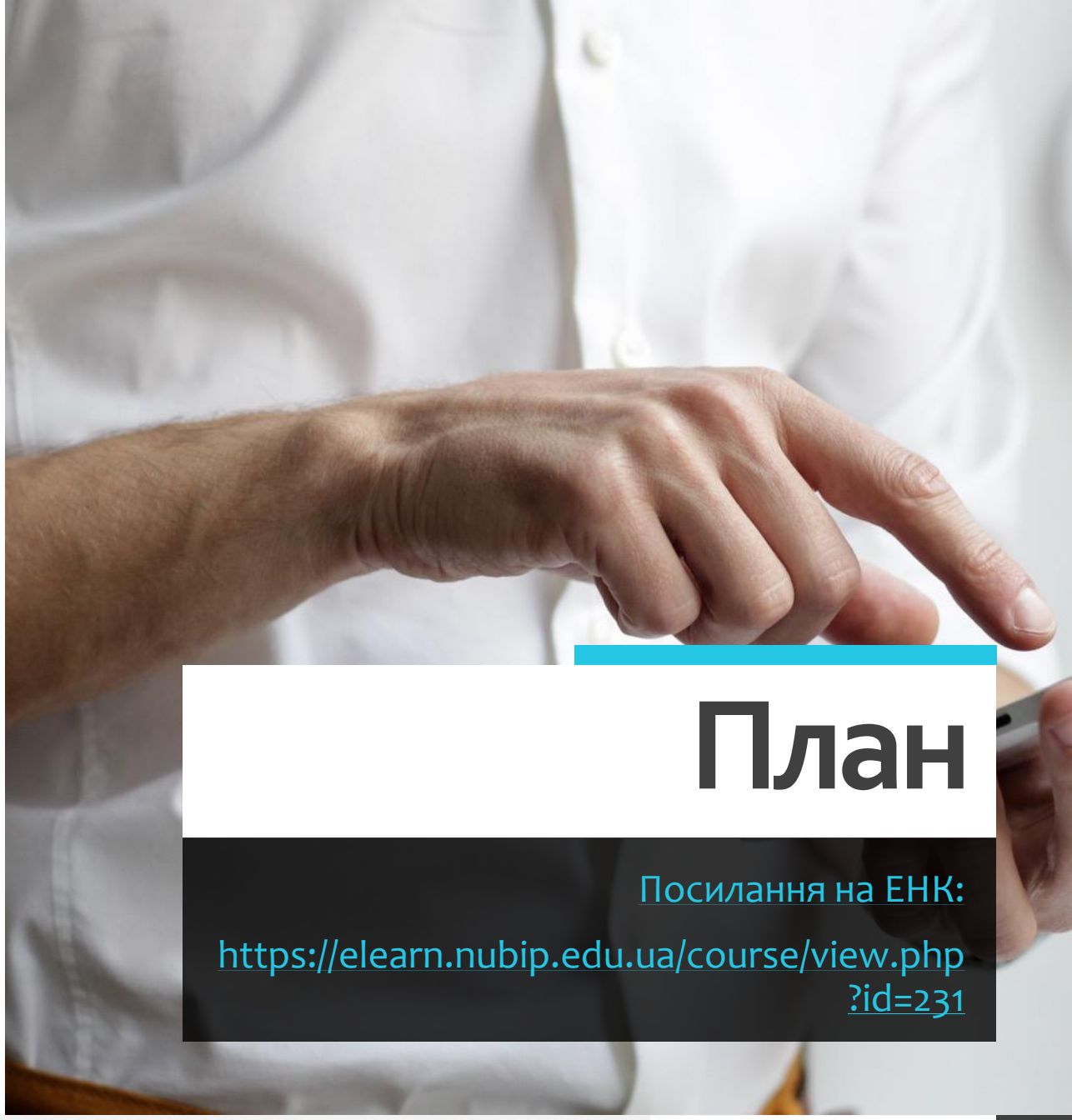
Моделі об'єктно-орієнтованого проектування

Проектування інформаційних систем

Лекція4

Глазунова О.Г.,
д.п.н., проф.

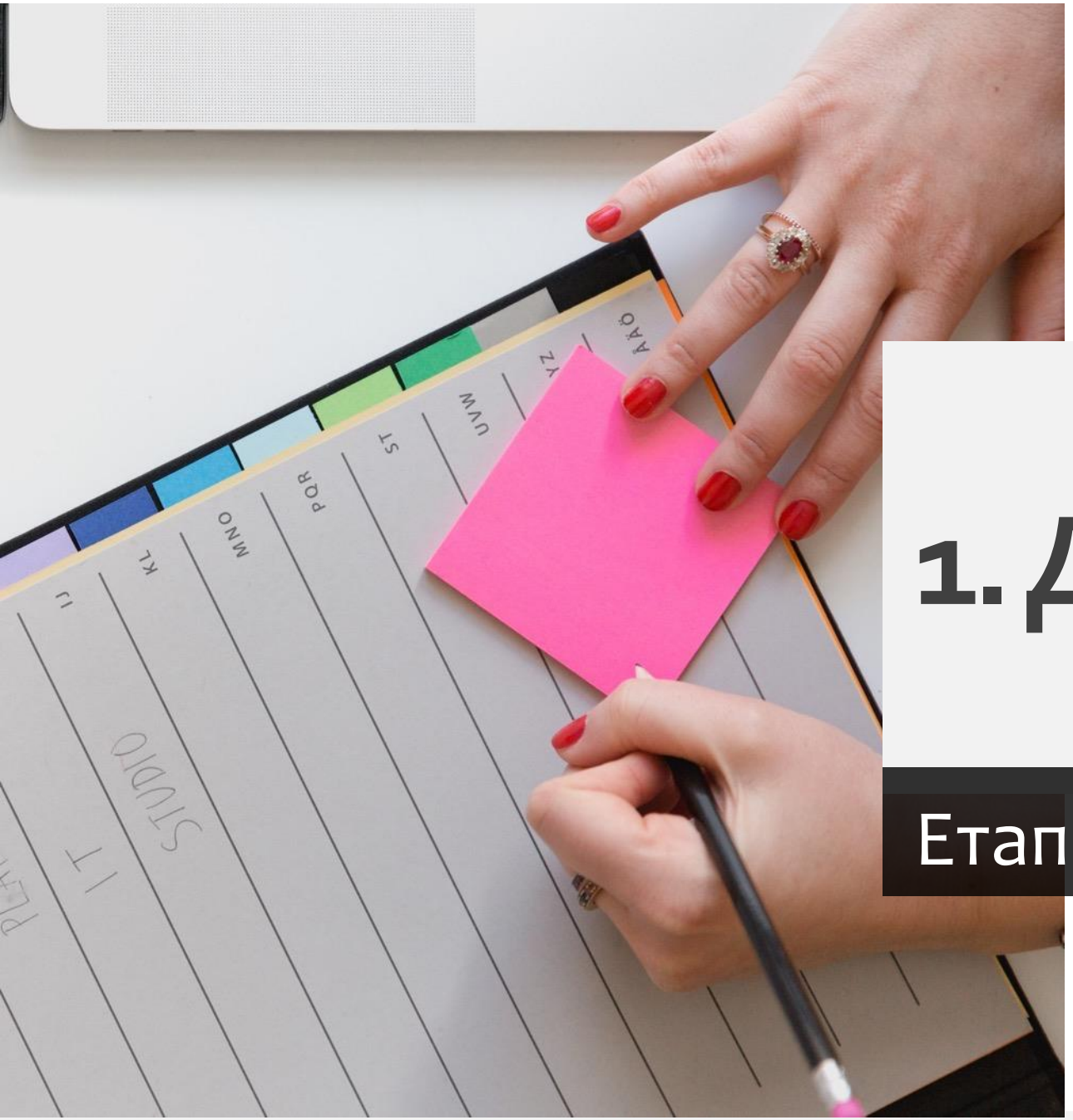
1. Діаграма класів: етап аналізу/етап проектування
2. Класи сутності та відношення між ними
3. Класи інтерфейсні та відношення реалізації
4. Класи керування та моделі послідовності



План

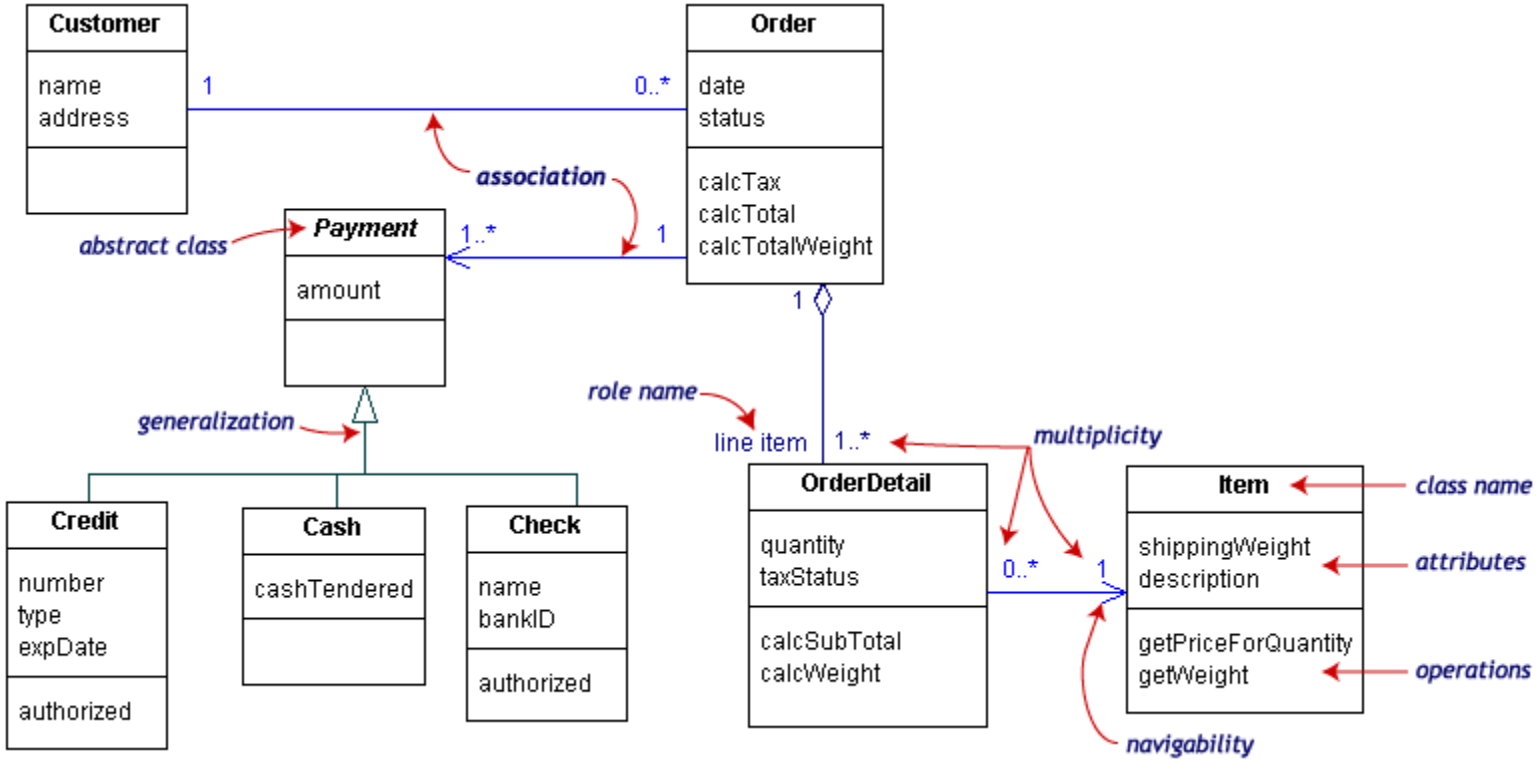
Посилання на ЕНК:

https://elearn.nubip.edu.ua/course/view.php?id=231



1. Діаграма класів

Етап проектування



Типи класів:

Клас керування (control class) відповідає за координацію поведінки об'єктів системи. У кожного прецеденту, зазвичай, є хоча б один клас керування, який контролює послідовність виконання дій цього прецеденту. Зазвичай, цей клас є активним та ініціює розсилання множини повідомлень іншим класам моделі. Клас керування може бути відсутнім у прецедентах, які здійснюють прості маніпуляції зі збереженими даними.

Клас-сутність (entity) містить інформацію, яка повинна зберігатися постійно і не знищуватися зі знищенням об'єктів певного класу чи припиненням роботи системи (вимиканням системи чи завершенням програми). Цей клас може відповідати окремій таблиці бази даних; у цьому випадку його атрибути є полями таблиці, а операції - збереженими процедурами. Зазвичай, цей клас є пасивним і лише приймає повідомлення від інших класів моделі. Класи-сутності є *ключовими абстракціями* (поняттями) ПО системи.

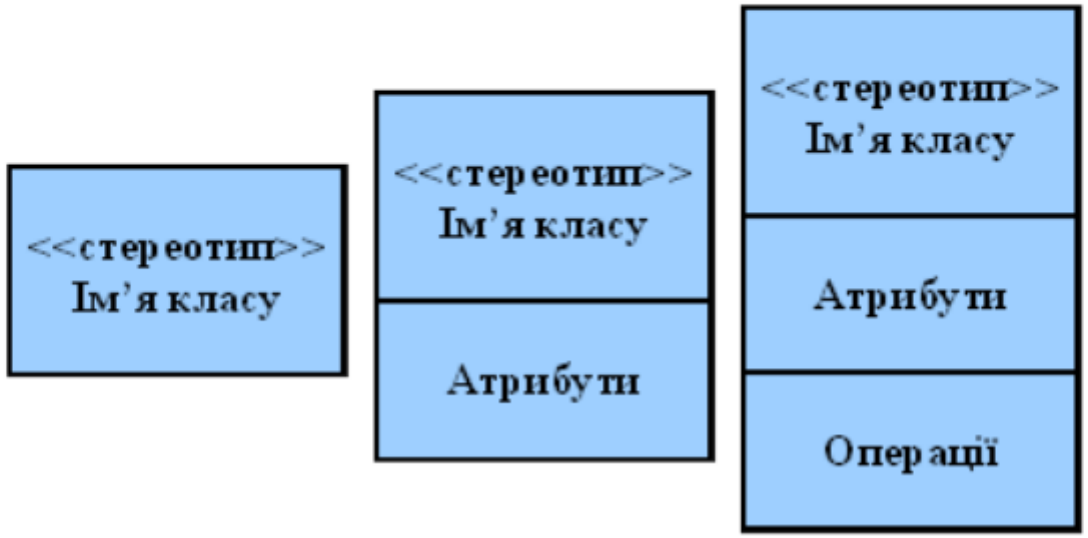
Межовий клас (boundary) розташовується на межі системи з зовнішнім середовищем, однак є складовою частиною системи.

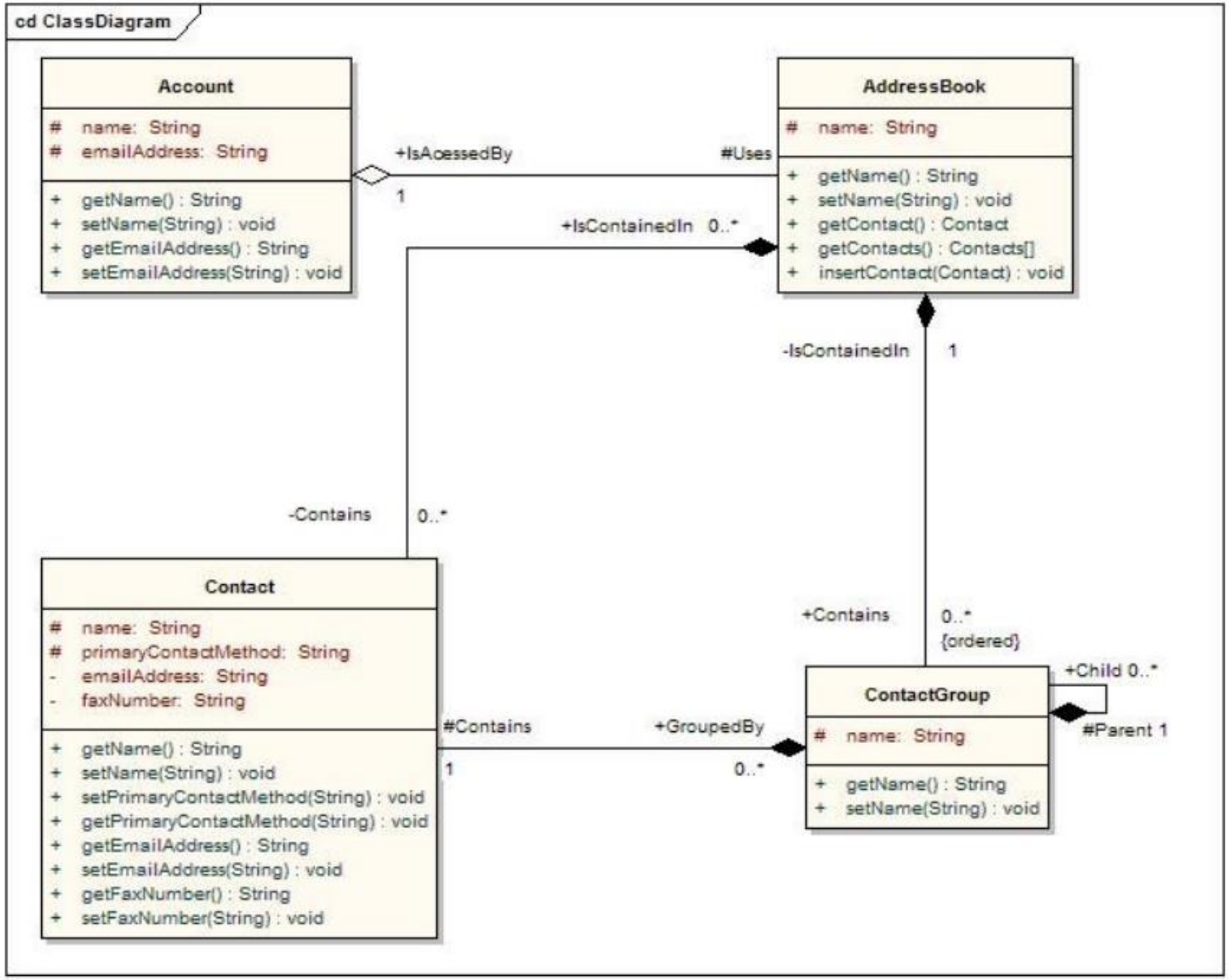
Цей клас слугує посередником під час взаємодії зовнішніх об'єктів із системою. Зазвичай, для кожної пари актор - прецедент визначається один межовий клас.

1. Класи-сутності

Етап аналізу

- **Клас (class)** у мові UML служить для позначення конкретної сутності, яка має однакову структуру та поведінку. Як видно з прикладу, графічно клас зображується у вигляді прямокутника, що додатково може бути розділений горизонтальними лініями на 3 розділи або секції: ім'я класу, атрибути (змінні-члени класу) та операції (методи класу)





Класи-сутності

Етап проектування

Відношення між класами

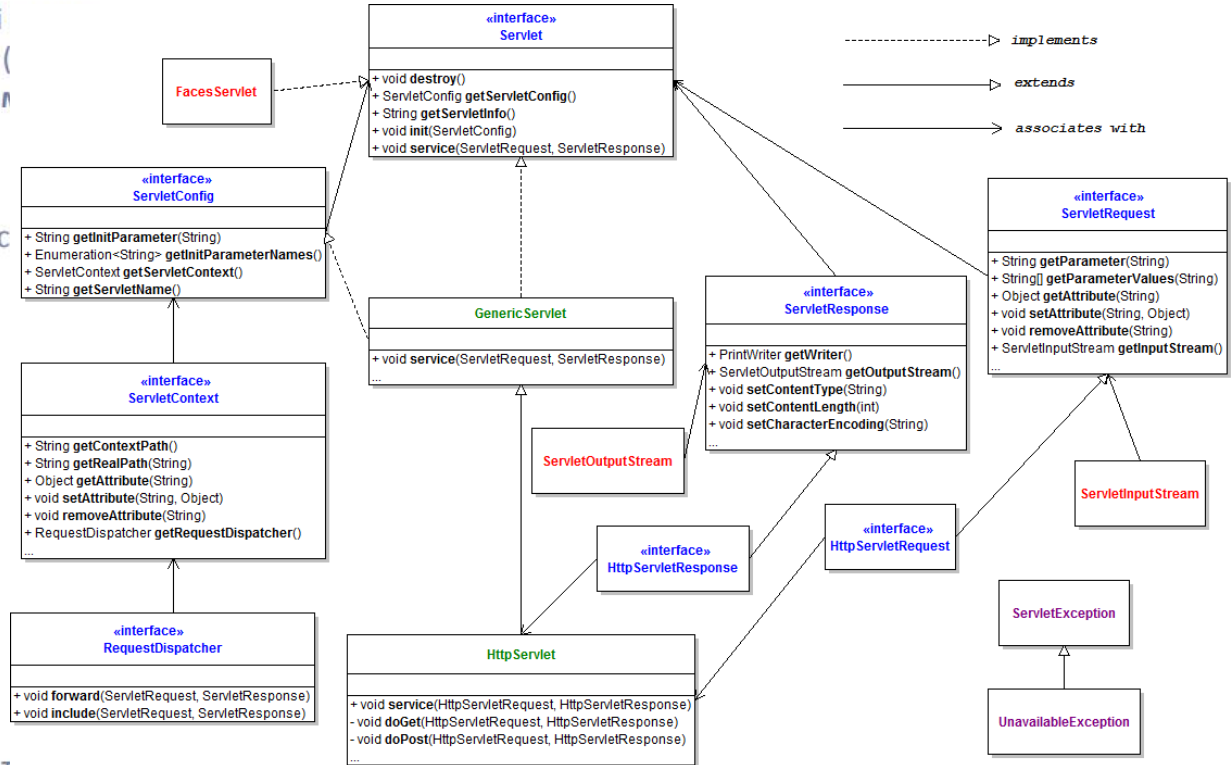
- **Відношення узагальнення (generalization relationship)** – зв’язують дочірні класи з батьківськими;
- **Відношення асоціації (association relationship)** – представляють структурні відношення між класами;
- **Відношення агрегації (aggregation relationship)** – відображає взаємовідносини типу «частина-ціле» між класом та його складовими частинами;
- **Відношення композиції (composition relationship)** – підвид попереднього.

2. Класи граничні

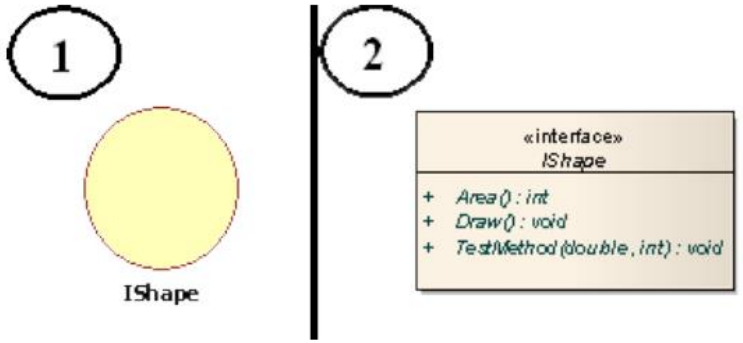
Етап проектування

2 - Інтерфейси

- Інтерфейс в контексті мови UML являється спеціальним випадком класу, в якого присутні операції, але відсутні атрибути. Тому для відображення інтерфейсів використається **круг (** основному використовується в діаграмах компонентів і розгортки (deployment)) або **прям класу**, який поділений тепер на **два розділи**:
  - розділ, в якому вказується назва інтерфейсу з стереотипом <<interface>>.
  - поле для методів інтерфейсу, які потребують подальшої реалізації в дочірних клас



- Наприклад, відобразимо стандартний інтерфейс фігури:



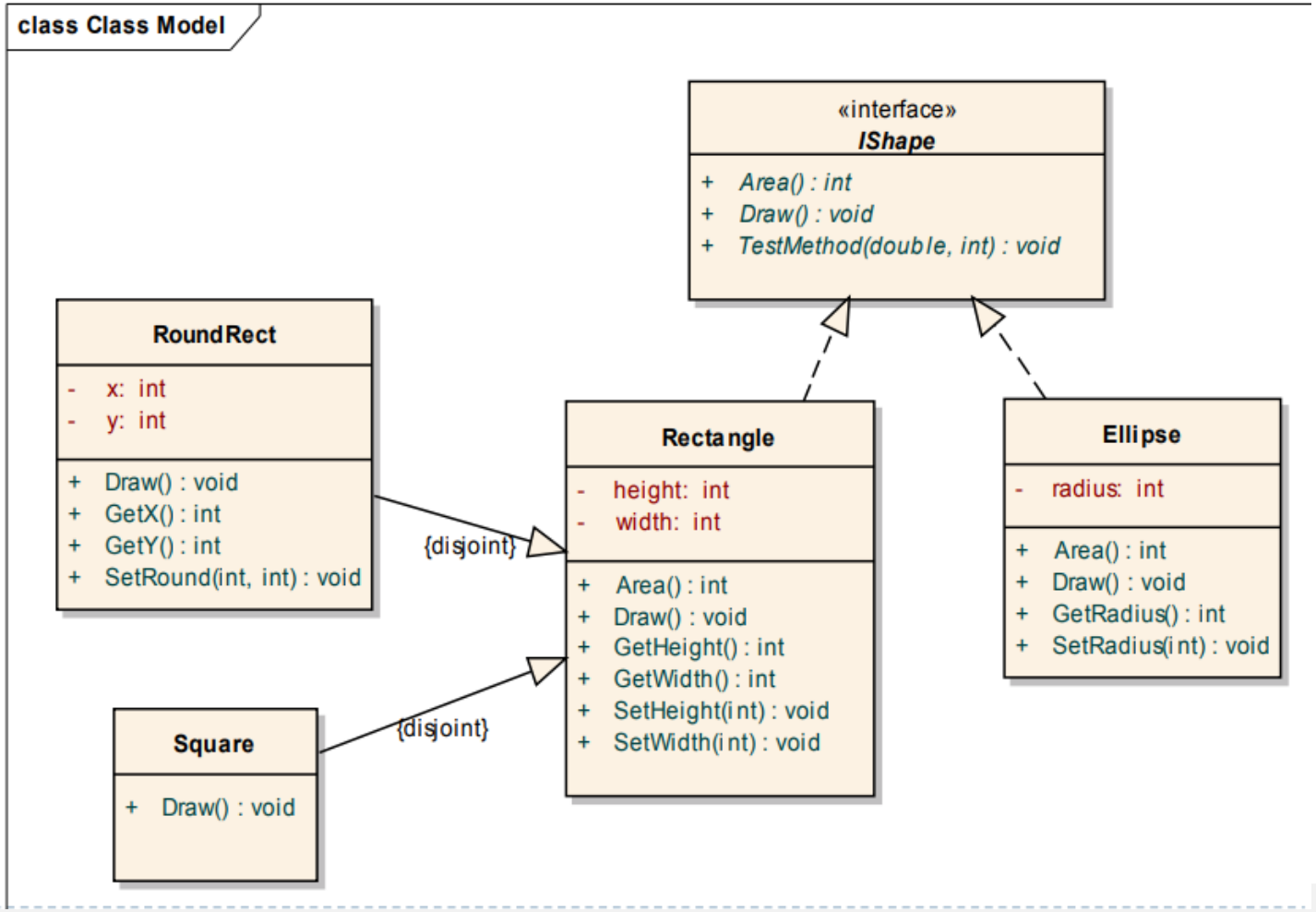
- Другий спосіб може бути більш уточнений за допомогою деяких програм розробки. При такому уточненні третій метод буде описаний наступним чином:

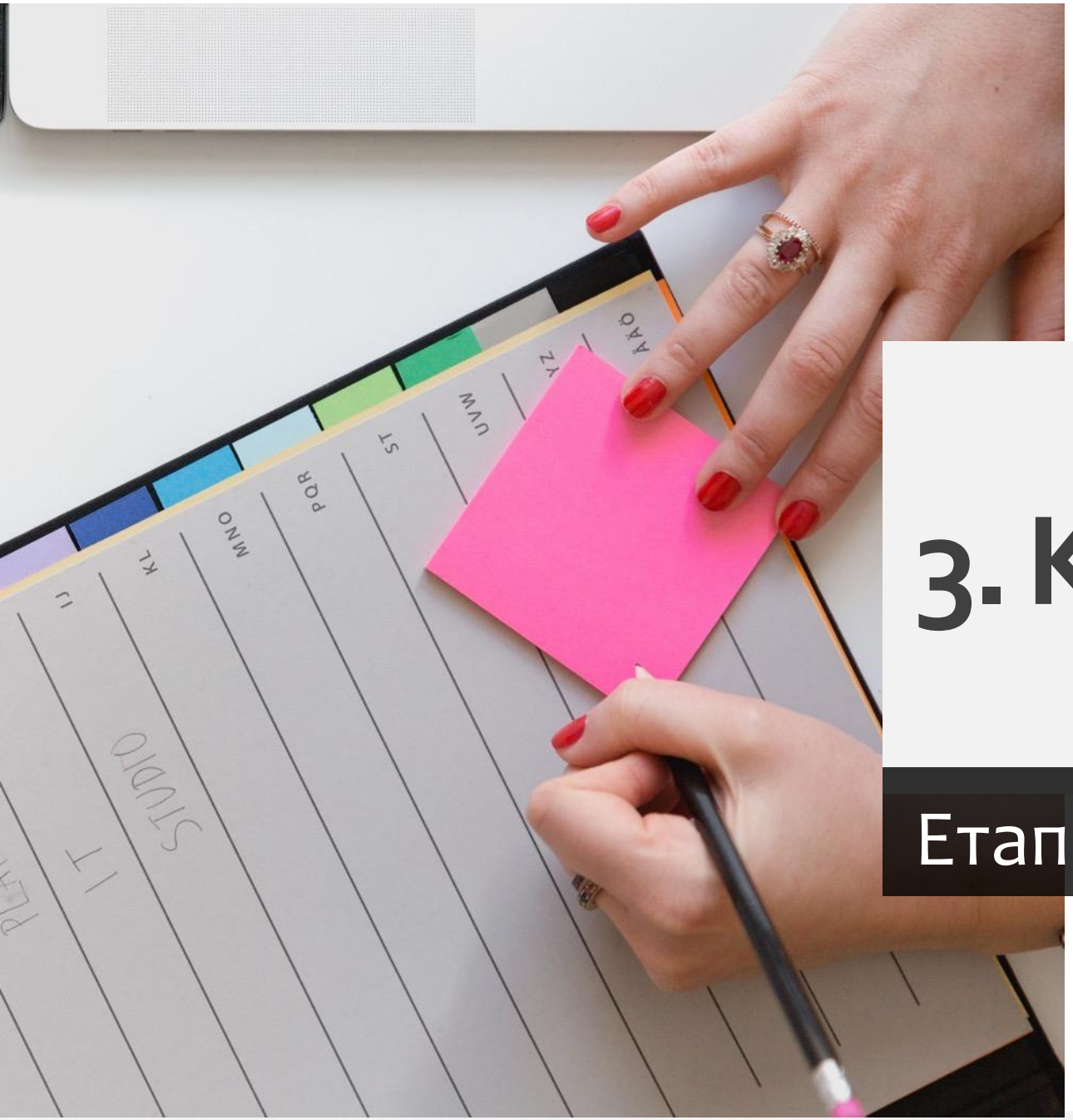
+ TestMethod(b: double, a: int) : void

Відношення реалізації

- В відношення узагальнення є ще один підвид – **відношення реалізації**, які вказують на взаємовідносини між інтерфейсом або абстрактним класом і дочірним класом. В такому випадку дочірний клас повністю реалізує перший і тому кажуть про реалізацію, а не успадкування.

- Для відображення відношень реалізації використовується пунктирна лінія з трикутною стрілкою, яка направлена від дочірного класу до батьківського.

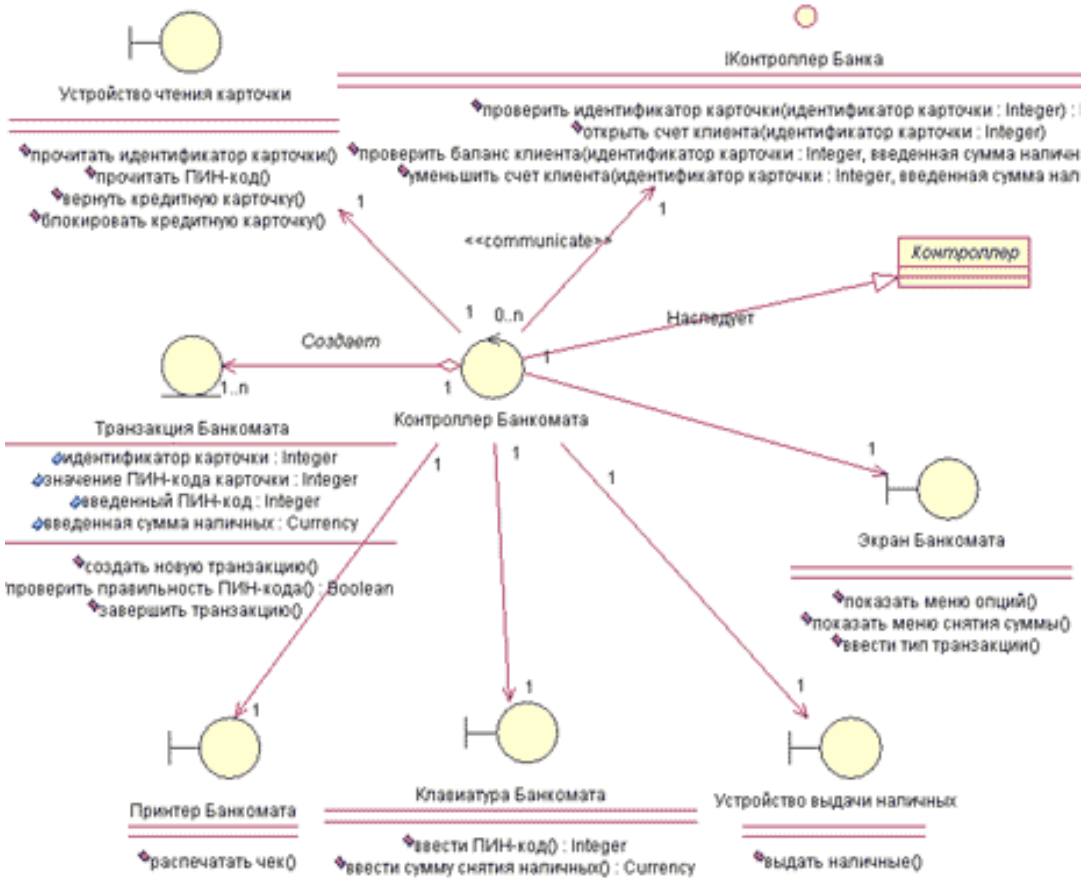


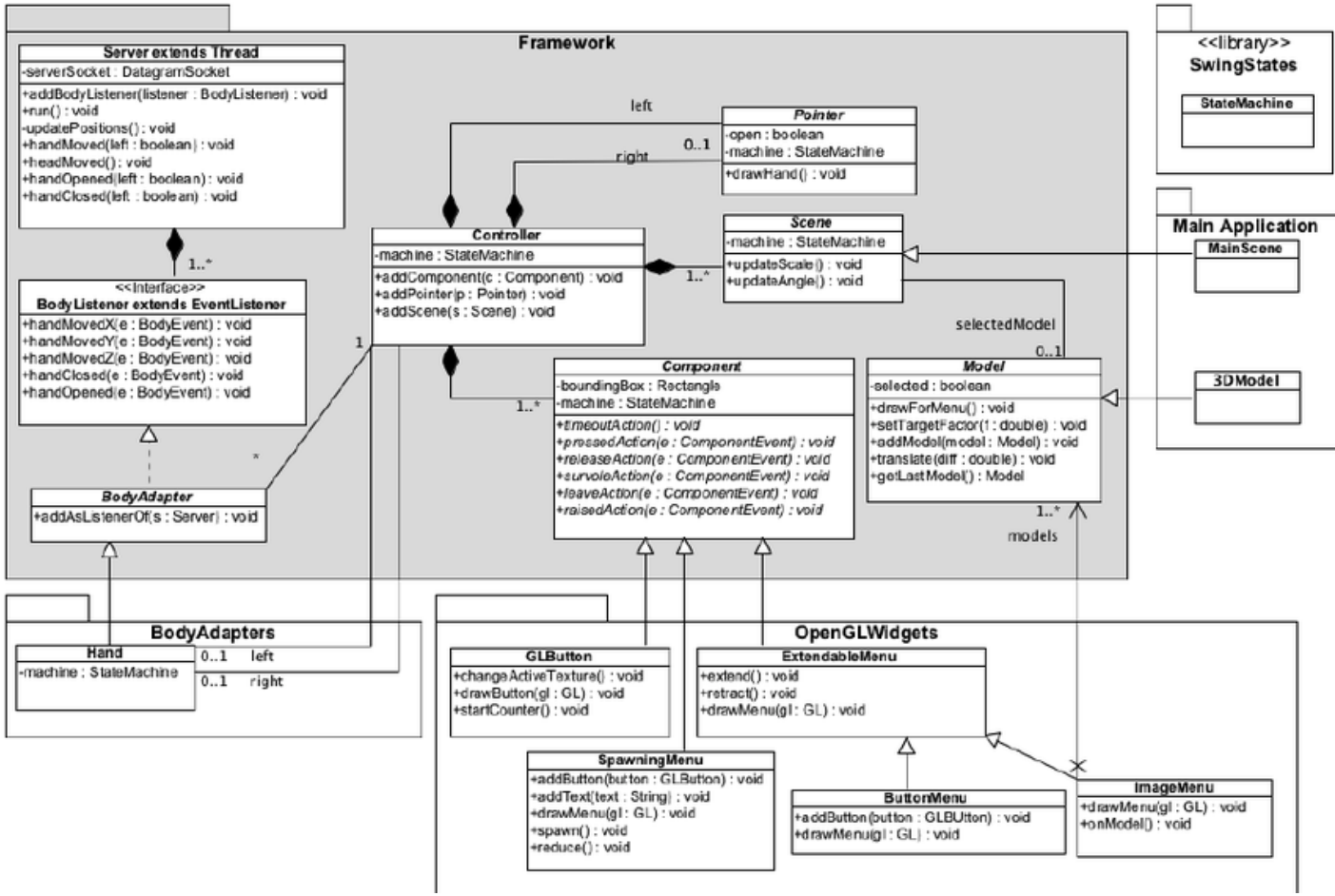


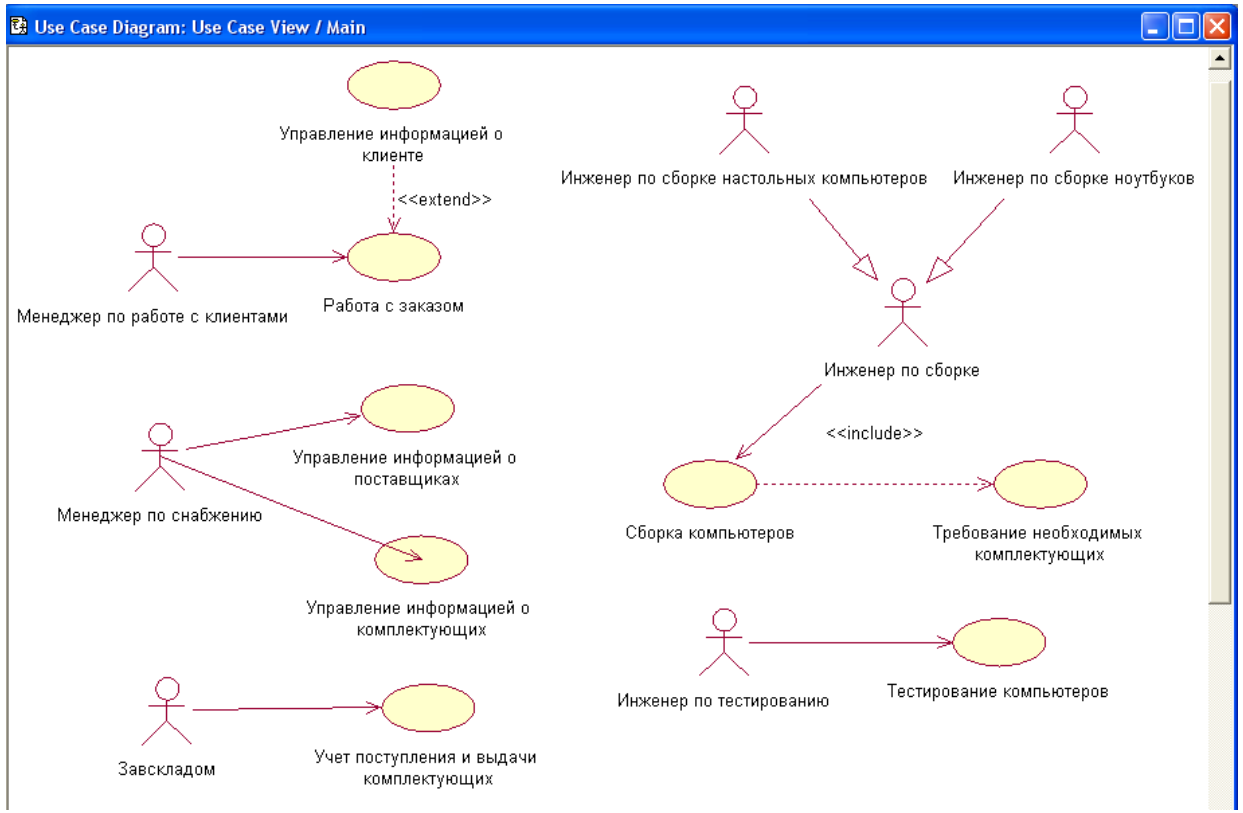
3. Класи керування

Етап проектування

3 - Класи керування



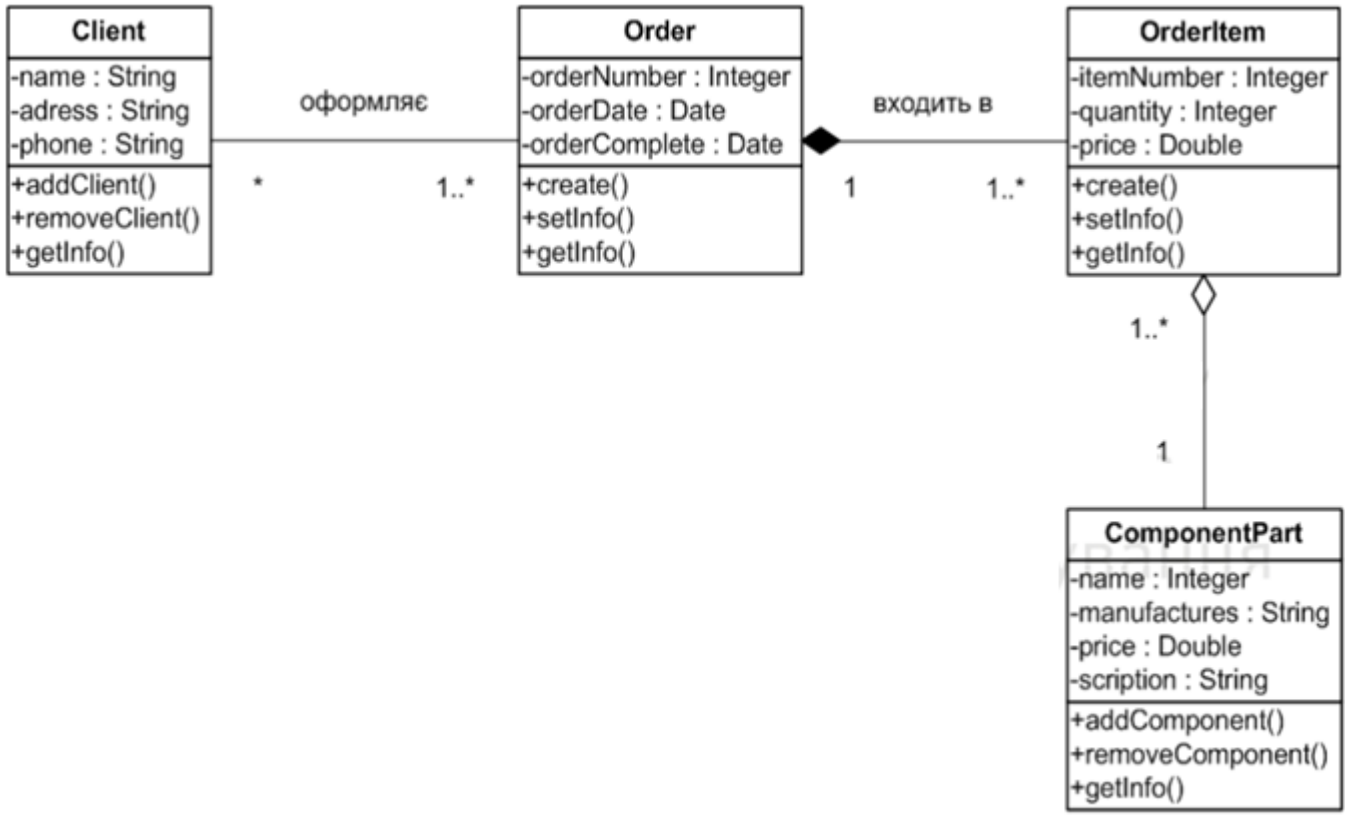




Методика моделювання

1. Обрати прецедент
2. Виявити об'єкти
3. Спроектувати класи сутності

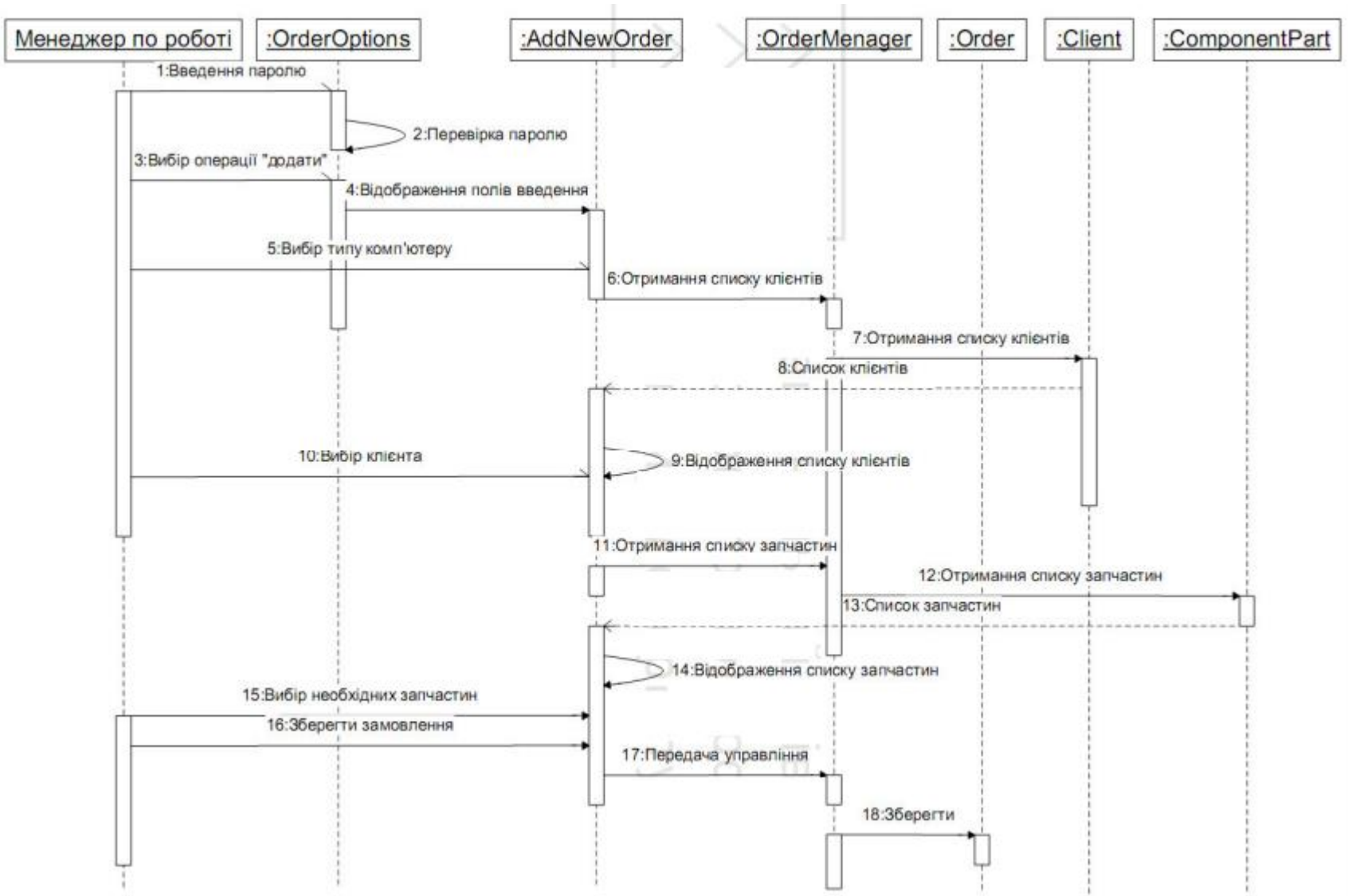
Client	Order	OrderItem	ComponentPart
-name : String -adress : String -phone : String	-orderNumber : Integer -orderDate : Date -orderComplete : Date	-itemNumber : Integer -quantity : Integer -price : Double	-name : Integer -manufactures : String -price : Double -scription : String
+addClient() +removeClient() +getInfo()	+create() +setInfo() +getInfo()	+create() +setInfo() +getInfo()	+addComponent() +removeComponent() +getInfo()



4. Встановити зв'язки між класами

5. Додати класи граничні та керування

6. Спроектувати стани об'єктів



7. Спроектувати поведінку керуючих класів