**ТЕМА 5. ПОБУДОВА ДІАГРАМИ КЛАСІВ В UML. МОДЕЛЬ ПРЕДМЕТНОЇ ОБЛАСТІ.**

**Мета:** набути практичних навичок застосування методології об’єктно орієнтірованного моделювання для виявлення об’єктів системи за допомогою моделі предметної області та діаграми класів UML.

* 1. **Короткі теорітичні відомості з призначення та елементів діаграми класів UML.**

Діаграма класів використовуються для моделювання статичного виду системи з точки зору проектування.

Зазвичай діаграми класів використовуються для таких цілей:

* для моделювання словника системи;
* для моделювання простих кооперацій, **кооперація** - це співтовариство класів, інтерфейсів і інших елементів, що працюють спільно для забезпечення деякого кооперативного поведінки, більш значущого, ніж сума складових його елементів;
* для моделювання логічної схеми бази даних.

***Класом (Class)*** називається опис сукупності об'єктів із загальними атрибутами, операціями, відносинами і семантикою. Можливі класи виявляютьсяпри розгляді трьох стереотипів: сутність (entity), межа (boundary) і управління (control). Клас-сутністьмістить інформацію, яка зберегається постійно. Використовується для моделювання даних і поведінки об'єктів з довгим життєвим циклом. Вони можуть представляти інформацію про предметну область, а можуть представляти елементи самої системи. У UML клас коротко познаяается двома способами рис.5.1

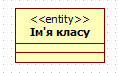


Рисунок 5.1 – Просте зображення класу в UML.

***Атрибут*** - це іменоване властивість класу, що включає опис безлічі значень, які можуть приймати екземпляри цієї властивості. Клас може мати будь-яке число атрибутів або не мати їх зовсім. Атрибути поміщаються на другий рівень елемента «клас» рис. 5.2. Для іменування атрибута використовують одне або кілька коротких іменників, відповідних деякому властивості класу. Кожне слово в імені атрибута, крім самого першого, зазвичай пишеться з великої літери.

Атрибут має наступні параметри: видимість, ім’я, тип даних, значення по замовченню, властивості. Окрім ім’я, інші параметри не обов’язкові.

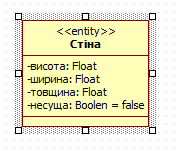


Рисунок 5.2 – Представлення атрибутів в елетенту класу UML.

***Операція класу (operation****)* являє собою деякий сервіс, що надається для кожного екземпляра класу за певним вимогу. Операції записуються на третьому рівні елемента класу. Кожній операції класу в UML відповідає окремий рядок виду:

< видимість> <ім'я операції> (список параметрів): <вираз типу значення, що повертається> {рядок-властивість}

Приклади запису операцій UML:

display - тільки ім'я;

+ Display - видимість і ім'я;

set (n: Name, s: String) - ім'я і параметри;

getID (): Integer - ім'я і повертається значення;

restart () {guarded} - ім'я та властивість.

Часто виникає необхідність створити клас, у якого:

* немає жодного екземпляру - тоді клас стає службовим (Utility), що містить лише атрибути і операції з областю дії класу;
* рівно один екземпляр - такий клас називають синглетним (Singleton);
* заданий число екземплярів;
* довільне число екземплярів - варіант за замовчуванням.

Кількість екземплярів класу називається його кратністю. У мові UML кратність класу задається виразом, написаним в правому верхньому кутку його піктограми.

**Об'єкт класу** (object) є окремим екземпляром класу, який створюється на етапі виконання програми. Він має своє власне ім'я і конкретні значення атрибутів. На рис 5.3. представлені варіанти імен об’єктів класів.

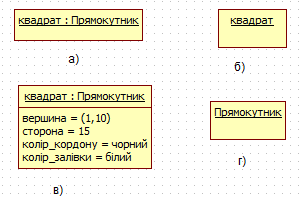


Рисунок 5.3 – Створення елементу об’єкт класу.

Між класами в мові UML можно створювати базовими відносинами або зв'язками:

* ***відносини асоціації*** (Association relationship)
  + Бінарна
  + N-арная
  + Агрегація
  + композиція
* ***ставлення узагальнення*** (Generalization relationship)
* ***ставлення реалізації*** (Realization relationship)
* ***ставлення залежності*** (Dependency relationship).

***Ставлення узагальнення - це спадкування***. Ставлення узагальнення є звичайним відношенням між більш загальним елементом (батьком або предком) і більш приватним або спеціальним елементом (дочірнім або нащадком). На діаграмі класів зображається прямою лінією з пустим трикутноком на кінці рис. 5.4.

 a)

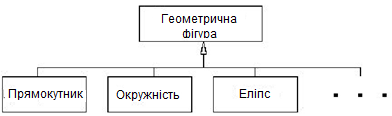
 b)

Рисунок 5.4 – зв’язок узагальнення.

***Асоціація показує відносини між реалізованими об'єктами класу***. Асоціації мають навігацію: двосторонню або односпрямовану, що вказує на напрям зв'язку. В якості додаткових спеціальних символів можуть використовуватися ім'я асоціації, а також імена і кратність класів-ролей асоціації рис. 5.5.

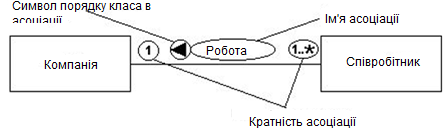


Рисунок 5.5 - Зв’язок асоціації.

Спеціальною формою або окремим випадком відносини асоціації є **відношення агрегації** (з'єднання частин), яке, в свою чергу, теж має спеціальну форму - **відношення композиції**. В UML ***агрегація*** відображає зв'язок класів, коли ***об'єкт одного класу є атрибутом іншого.*** Дане відношення має фундаментальне значення для опису структури складних систем, оскільки застосовується **для подання системних взаємозв'язків типу "частина-ціле**". Розкриваючи внутрішню структуру системи, відношення агрегації показує, з яких компонентів складається система і як вони пов'язані між собою рис 5.6.

 a)

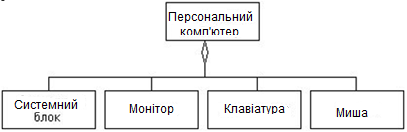
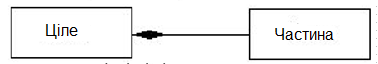
 b)

Рисунок 5.6 - Зв’язок агрегації.

***Композиція*** служить для виділення спеціальної форми відносини "частина-ціле", при якій складові частини в деякому сенсі перебувають всередині цілого рис.5.7. Специфіка взаємозв'язку між ними полягає в тому, що ***частини не можуть виступати у відриві від цілого***, тобто зі знищенням цілого знищуються і всі його складові частини. Різниця між композицією і агрегацією: компоненти зібрані агрегацией можна роз'єднати, а з композицією цього зробити не вийде.

 a)

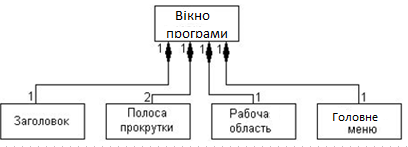
 b)

Рисунок 5.7- Зв’язок композиції.

***Реалізація* *показує відношення: клас - об'єкт***. На діаграмі реалізація показується пунктирною лінією і незафарбовані стрілочкою.

Ставлення ***залежності*** в загальному випадку вказує деяке семантичне відношення між двома елементами моделі або двома множинами таких елементів, яке **не є відношенням асоціації, узагальнення або реалізації**. Зображується пунктирною лінією зі стрілкою. Стрілка спрямована від класу-клієнта залежності до незалежного класу або класу-джерела рис 5.8.

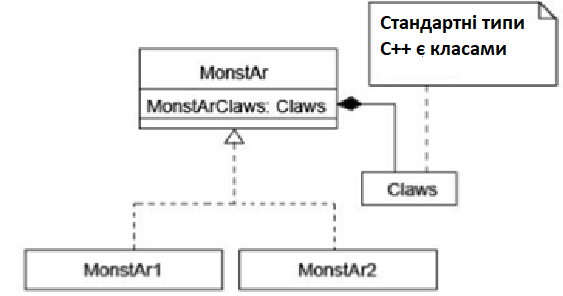
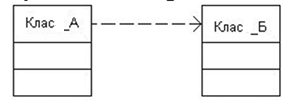


Рисунок 5.8 - Зв’язки реалізації та залежності.

**5.2 Модель предметної області в межах методології UP.**

***Модель предметної області*** - це найважливіша модель об'єктно-орієнтованого аналізу. Вона відображає основні, тобто моделює класи понять (концептуальні класи) предметної області. Кожній ітерації відповідає своя модель предметної області, яка відображає реалізовані на даному етапі сценарії прецедентів. Таким чином, модель предметної області еволюціонує в процесі розробки системи. Модель предметної області пов'язана з моделлю проектування, особливо програмними об'єктами, відносяться до рівня предметної області. Основні поняття цієї моделі відображаються в словнику термінів.

Ідентифікація набору концептуальних класів - основне завдання об'єктно-орієнтованого аналізу. На початкових ітераціях фази розвидку побудова моделі предметної області у досвідченого розробника може зайняти всього лише кілька годин, але на наступних етапах, коли вимоги до системи визначаються більш чітко, уточнення моделі предметної області потребує значно більше часу

Модель предметної області в рамках UP є конкретизацією більш загального поняття моделі бізнес-об’єктів (business object model- BOM), що забезпечує подання понять, що грають важливу роль в даній предметній області. Таким чином, подібна модель описує поняття однієї предметної області, наприклад, пов'язані з роздрібною торгівлею.

На мові UML Модель предметної області представляється у вигляді набору діаграм класів, на яких не визначені жодні операції. Модель предметної області може відображати наступне:

* Об'єкти предметної області або концептуальні класи.
* Асоціації між концептуальними класами.
* Атрибути концептуальних класів.

Модель предметної області можна розглядати як візуальний словник важливих абстракцій або словник предметної області. Таким чином, Модель предметної області - це результат візуалізації понять реального світу в термінах предметної області, а не програмних елементів, таких як класи Jаvа або С #. Отже, в моделі предметної області не використовуються наступні елементи.

* Артефакти програмування на зразок вікон або бази даних, якщо тільки розробляється система не є моделлю програмного засобу, наприклад моделлю графічного інтерфейсу користувача.
* Обов'язки або методи.

Модель предметної області не є моделлю даних (data model), до якої по визначенню відносяться дані, що зберігаються на постійному носії, оскільки невідомо, чи потрібно зберігати в базі даних інформацію про класи. Концептуальні класи можуть взагалі не містити атрибутів і грати в предметної області чисто поведінкову, а не інформаційну роль.

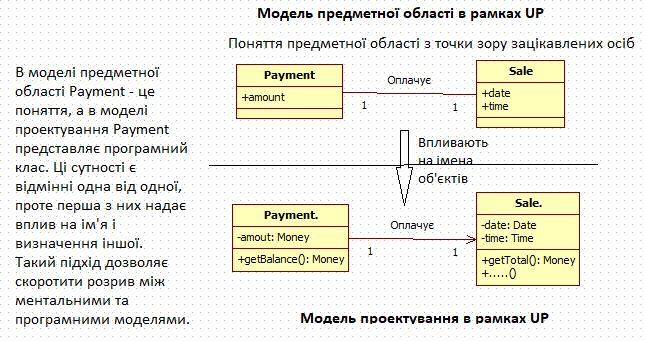


Рисунок 5.9 – Модель предметної області в межах UP.

Для створення Моделі предметної області на даній ітерації треба виконати наступні дії.

* Виділити концептуальні класи (дивись нижченаведені процедури з рекомендаціями).
* Додати їх в моделі предметної області у вигляді класів на діаграмі UML.
* Додати необхідні асоціації і атрибути.

Існує три способи виявлення концептуальних класів.

1. Повторне використання або модифікація існуючих моделей. Це самий перший, найкращий і звичайно, найпростіший підхід, з якого автор завжди намагається почати процес виділення концептуальних класів. В літературі описані моделі багатьох предметних областей і моделі даних, які можна трансформувати в моделі предметної області. Такі моделі існують для галузі фінансів, охорони здоров'я і т.п.

2. З використанням списку категорій концептуальних класів.

3. На основі виділення іменників.

***Використання списку категорій концептуальних класів***

Напочатку створення моделі предметної області доцільно скласти список кандидатів на роль концептуальних класів. У табл. 5.1 наведені стандартні категорії, які зазвичай мають важливе значення для предметної області системи торгівлі, резервування авіаквитків і гри "Монополія".

Таблиця 5.1 – Категорії концептуальних класів.

| Категория | Приклад |
| --- | --- |
| ***Транзакції*** - ці класи особливо критичні, оскільки часто описують фінансові операції, тому процес виділення концептуальних класів слід почати саме з них | Sale (Продаж), Payment (Платіж),  Reservation (Резервування) |
| ***Елементи транзакцій*** - транзакції часто складаються з елементів | SalesLineitem (Елемент продажу) |
| ***Товари або служби, пов'язані з транзакціями або їх елементами*** -  транзакції виконуються над деякими  елементами (товарами або службами) | Item (Елемент),  Flight (Рейс),  Seat (Місце) |
| ***Місця записи транзакцій*** - дуже важлива категорія | Register (Реєстр),  FlightManifest (Розклад польотів) |
| ***Ролі людей або організації, які пов'язані з транзакціями, виконавці прецедентів -***  необхідно знати, хто бере участь в транзакції | Cashier (Касир),  Customer (Покупець),  Store (Магазин),  MonopolyPlayer (Ігрок)  Pilot (Пілот)  Passenger (Пасажир) |
| ***Місця транзакцій*** | Store (Магазин),  Airport (Аеропорт),  Plane (Самоліт),  Seat (Місце) |
| ***Важливі події, для яких необхідно зберігати час і місце*** | Sale (Продаж), Payment (Платіж),  MonopolyGame (Монополія),  Flight (Політ) |
| ***Фізичні об'єкти*** - такі об'єкти зазвичай відповідають програмних системам, які призначені для управління або моделювання | Register (Реєстр),  Airplane (Самоліт),  Item (Товар),  Board (дошка),  Die (Игральна кість) |
| ***Опис об’єктів*** | ProductDescription (Специфікація товара),  FlightDescription (Опис польота) |
| ***Каталоги*** - опис часто наводиться  у каталозі | ProductCatalog (Каталог товарів),  FlightCatalog (Каталог рейсів) |
| ***Контейнери інших об’єктів*** (фізичних або інформаціоних) | Store (Магазин), Bin (Бункер),  Airplane (Самоліт), Board (Дошка) |
| ***Вміст контейнерів*** | Item (Елемент),  Square (Клітина на досці),  Passenger (Пасажир) |
| ***Інші системи, зовнішні по відношенню до даної системі*** | CreditAuthorizationSystem (Система авторізації  кредитних платежів),  AirTrafficControl (Система керування рухом) |
| ***Записи фінансової, трудової, юридичної***  ***та іншої діяльності*** | Receipt (Чек), Ledger (Гроссбух),  MaintenanceLog (Журнал обслуговування) |
| ***Финансові інструменти*** | LineOfCredit (Кредитна лінія),  Cash (Готівка), Check (Чек) |
| ***Керівництва, документи, статті, книги, на які посилаються в процесі роботи*** | DailyPriceChangeList (Бюлетень щоденної зміни цін),  RepairManual (Керівництво про відновлення) |

***Визначення концептуальних класів на основі виділення іменників.***

Ще один корисний (і дуже простий) прийом ідентифікації концептуальних класів реалізується на основі лінгвістичного аналізу текстових описів предметної області,той самий, який ми використовували при розробці моделі ER - виділення іменників та їх виборі в як кандидатів в концептуальні класи або атрібути.

Розгляне застосування цього методу на прикладі опису головного сценарію для прецедента «Оформити замовлення». Виделимо іменники та категорії.

1. Створити **замовлення** (**клієнт**, тип б/т, назва, номер, що не робить, дата)
2. Додати інформацію
3. Призначити **майстра з діагностики** за **правилами фірми**
4. Призначення **статуса**
5. Роздрукувати **квітакцію**

Можливо, найбільш типовою помилкою при створенні моделі предметної області є зарахування деякого об'єкта до атрибутів, в той час як він повинен відноситися до концептуальних класів. Щоб уникнути цієї помилки, слід дотримуватися простого правила, якщо деякий об'єкт *Х* в реальному світі не є числом або текстом, значить, це скоріше концептуальний клас, ніж атрибут. Наприклад, чи є Store (магазин) атрибутом об'єкта Sale (Продаж) або окремим концептуальним класом Store? У реальному світі магазин не є числом або текстом, він представляє реальну сутність, організацію, що займає деякий місце. Отже, Store потрібно розглядати як концептуального класу.

Можно видилити наступні категорії для виявлення концептуальних класів прецедента «Оформлення замовлення»: Транзакції (замовлення), Товари або служби, пов'язані з транзакціями або їх елементами (б/т) , Ролі людей або організації, які пов'язані з транзакціями, виконавці прецедентів (клієнт, Адміністратор, Майстер з діагностики), Місця транзакцій (Фірма), Важливі події, для яких необхідно зберігати час і місце (замовлення) , Фізичні об'єкти (б/т) , Опис об’єктів (Специфікація б/т), Записи фінансової, трудової, юридичної та іншої діяльності (квітанція), Керівництва, документи на які посилаються в процесі роботи (правила).

Як бачимо, кількість іменників та категорій значно різниться, тому що категорії вже враховує рекомендації для виявлення концептуальних класів для деяких предметних областей. Об’єкт Квітанція треба враховувати, якщо квітанція буде використовуватися при поверненні б/т в межах гарантії, інакше – це просто звіт про замовлення. Іменник «статус» є атрибутом, тому що це просто строка. Таким чином, можемо видилити наступні концептуальні класи для прецедента:

Order (замовлення)

Client (клієнт)

Diagnostic master (майстер)

Admin (адміністратор)

Appliance (б/т)

ApplianceDescription (Специфікація б/т) – потрібен, якщо існує необхідність опису елементів або служб незалежно від існування конкретних екземплярів цих об'єктів.

Firm (фірма)

Rules (правила)

***Виявлення асоціацій між концептуальними класами в прецеденті «Оформлення замовлення».***

Асоціація (association) - це відношення між класами (або точніше, екземплярами цих класів), що відображає деякий значущі і корисні зв'язки між ними. У мові UML, як це було описано више, асоціації описуються як "семантичні взаємозв'язки між двома або кількома класифікаторами і їх екземплярами ". Заслуговують на увагу асоціації, що зазвичай містять знання про взаємозв'язок між об'єктами, які повинні зберігатися протягом деякого періоду. Цей період може вимірюватися в мілісекундах або роках в залежності від конкретного контексту.

***Лінії асоціацій в моделі предметної області не описують потоки даних, зовнішні ключі баз даних, змінні примірників або зв'язку між об'єктами в програмної реалізації.*** Такі асоціації слід розглядати лише в концептуальному ракурсі - як зв'язок в реальній предметній області. Багато з цих зв'язків будуть реалізовані в програмному забезпеченні (як в моделі проектування, так і в моделі даних). Однак модель предметної області це не модель даних. Додані до неї ***асоціації дозволяють лише краще зрозуміти взаємозв'язок між об'єктами, але не відображають структури даних***.

Імена асоціацій повинні починатися з великої літери, оскільки асоціація зазвичай являє класифікатор зв'язків між екземплярами. У мові UML ім'я класифікатора починається з великої літери. Для імен асоціацій прийнято використовувати два формати: Records-Current (з дефісом) чи RecordsCurrent (без дефіса). Кожен кінець асоціації називається роллю (role). Роль додатково може мати такі характеристики: кратність, ім’я, напрямок зв’язку рис 5.10.

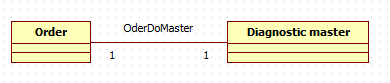


Рисунок 5.10 – Концептуальні класи та асоціація.

***Пошук асоціацій за допомогою списку стандартних асоціацій.***

Приступимо до додавання асоціацій з використанням списку, представленого в табл. 5.2. У ньому вказані стандартні категорії, якими зазвичай не слід нехтувати, особливо в комерційних інформаційних системах. Приклади асоціацій взяті з предметних областей резервування авіаквитків, роздрібної торгівлі та гри "Монополія".

Таблиця 5.2 – Стандартні асоціації.

| Опис | Назва асоціації |
| --- | --- |
| ***А*** є транзакцією, яка пов'язана  з іншого транзакцією ***В*** | CashPayment-Sale (Платіж готівкою -Продаж)  Reservation-Cancellation (Заказ білета-Скасування заказу) |
| ***А*** є елемент транзакції | SalesLineitem-Sale (Елемент продажі-Продаж) |
| ***А*** є товаром чи пуслугою для  транзакції ***В*** | Item-SalesLineitem (Елемент-Елемент продажу)  Flight·Reservation (Рейс-Резервування) |
| ***А*** є роль, яка пов’язана з транзакцією ***В*** | Customer·Payment (Покупець-Платіж)  Passenger-Ticket (Пасажир-Білет) |
| ***А*** є физичною або логичною  частиною ***В*** | Drawer-Register (Пристрій друку торгових чеків-Реєстр)  Seat-Airplane (Місце-Самоліт)  Square-Board (Клітина-Дошка) |
| ***А*** физично або логично міститься у ***В*** | Register-Store (Реєстр-Магазин)  Item-Shelf (Товар-Полка)  Square-Board (Клітина-Дошка)  Passenger-Airplane (Пасажир-Самоліт) |
| ***А*** є описом ***В*** | ProductDescription-Item (Опис товара-Товар)  FlightDescription-Flight (Опис польоту-Політ) |
| ***А*** відомий / зареєстрований / записаний / включений в ***B*** | Sale-Register (Продаж-Реєстр)  Piece-Square (Об’єкт-Клітина)  Reservation-FlightManifest (Замовлення білета-Декларація) |
| ***А*** є членом ***В*** | Cashier-Store (Касир-Магазин)  Player-MonopolyGame (Ігрок-Гра "Монополія")  Pilot-Airline (Пілот-Авіарейс) |
| ***А*** є организацийною одиницєю ***В*** | Department-Store (Відділ-Магазин)  Maintenance·Airline (Послуга підтримки-Авіарейс) |
| ***А*** використовує, керує або володіє ***В*** | Cashier-Register (Касир-Реєстр)  Player-Piece (Ігрок-Об’єкт)  Pilot-Airplane (Пілот-Самоліт) |
| ***А*** слідує за ***В*** | SalesLineitem-SalesLineitem (Назва товару - Назва товару)  city-City (Місто-Місто)  Square-Square (Клітина-Клітина) |

Визначемо асоціації для прецедента «Оформлення замовлення», застосовуя таблицю стандартних асоціацій.

***А*** є товаром чи пуслугою для транзакції ***В*** (Order записує прийом Appliance для ремонту)

***А*** є роль, яка пов’язана з транзакцією ***В*** (Client ініціїрує Order, Admin оформлює Order),

***А*** є членом ***В*** (Diagnostic master призначається для виконання Order),

***А*** є описом ***В*** (ApplianceDescription описує Appliance),

***А*** физично або логично міститься у ***В*** (Order належить Firm).

***Виявлення атрибутів концептуальних класів.***

В Моделі предметної області додаються атрибути, для яких визначені відповідні вимоги (наприклад, прецеденти) або для яких необхідно зберігати певну інформацію. Атрибути розміщуються у другому розділі умовного зображення класу. Додатково може бути вказано також тип атрибута або інша необов'язкова інформація. Рекомендується описувати вимоги до атрибутів у словниках термінів, який в майбутньому використовується як словник даних.

В Моделях предметної області атрибути повинні бути типами даних (data type). До стандартного типу атрибутів відноситься логічне значення, дата, номер, рядок (текст), час. Інші часто використовувані типи: Адреса (адреса), Колір (колір), геометрія (точка, прямокутник) (геометричні фігури: точка, прямокутник), Номер телефону (номер телефону), Номер соціального страхування (номер страхового поліса), Універсальний код товару (UPC) (універсальний код товару), SKU, ZIP або поштовий індекс (поштовий індекс), тип який перераховується. Стандартна помилка - це моделювання складного поняття предметної області у формі атрибута, наприклад, зовнішній ключ сутності, які потрібно відображати через асоціації.

Тип даних, особливо розглядається як число або рядок, може бути представлені у вигляді нового типу даних у моделях предметних областей у наступних випадках:

* Якщо він складений з окремих частин (Номер телефону, ім’я людини).
* Якщо з цим типом зазвичай асоціюються операції, такі як синтаксичний

аналіз і перевірка (Номер страхового поліса).

* Якщо він містить інші атрибути. Для пільгової ціни можуть встановлюватися термінові дії (початок і кінець).
* Якщо цей тип використовується для завдання кількості з одиницями розмірності (Вартість товару вимірюється в деяких одиницях).
* Це абстракція одного або кількох типів. Ідентифікатор товару - це кілька об’єднань типів UPC (Universal Product Code) і EAN (European Article Number).

Якщо клас представляє тип даних (унікальна тождественність не використовується в перевірці на равенство), його можна помістити в розділ атрибутів відповідного класу. Якщо же клас типу даних є новим типом і у нього є власні атрибути та асоціації, до більшого інтересу він буде представляти в якості окремого концептуального класу.

Більшість кількісних сутностей неможно представляти у вигляді кількох типов данных. Візьміть, до прикладу, вартість або швидкості. З цими кількісними сутностями пов'язані певні одиниці виміру, наприклад, якщо програмна система призначена для використання в різних країнах, тому вона повинна підтримувати різні типи валют. Рішення включає в себе окремий концептуальний клас Quantity (Кількість), з яким буде асоціюватися клас Unit (Єдиниця виміру). Наскільки кількість розглядається в якості даних типу, відповідний атрибут можна розмістити в розділі атрибутів. Зазвичай кількість вимірюється в деяких одиницях. Гроші - це кількість, одиниця вимірювання якої служить тип валюти. Вага - це кількість, яка вимірюється в килограмах або фунтах.

Розглянемо атрибути концептуальних класів для прецедента «Оформлення замовлення», які відповідають інформаційним вимогам першої ітерації сценарію. Опис сценарію предедента дозволяє виявити наступні атрибути, які пов’язані с необідностью друку квітанції – звіту про замовлення:

Order ( datetime)

Client (lastname)

Diagnostic master (id)

Admin (id)

Appliance (type, numbur)

ApplianceDescription (descreption)

Firm (name, address)

Створемо діаграму моделі предметної області прецедента «Оформлення замовлення» рис. 5.3

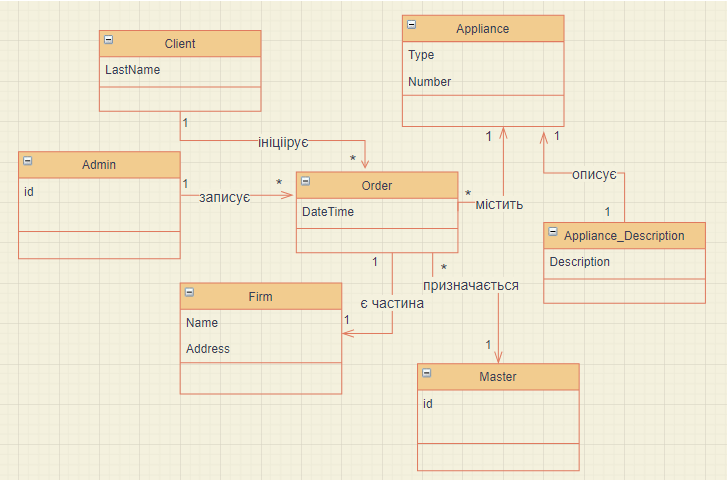


Рисунок 5.11 – Діаграма моделі предметної області для прецедента «Оформлення замовлення».

При ітеративній розробці модель предметної області поступово еволюціонує протягом кількох ітерацій. На кожній ітерації будується модель, яка відноситься тільки до розглянутого сценарію та не охоплює всі можливі концептуальні класи та взаємосвязи між ними. Наприклад, на першій ітерації системи «RepairТеch» розглядається лише непрофесійний сценарій прецедента «Оформлення замовлення», без друку квітанції замовлення, тому модель предметної області, яка створюється на цій ітерації, відображає лише поняття, необхідні для даной ітерації сценарія.

**Питання для самоконтролю**

1. Перерахувати статичні діаграми UML. З якою метою створюються діаграми классів?
2. Призначення, зображення та правила наіменування елементів діаграми класів: клас, атрибут, опреація.
3. Які типи зв’язків підтримуються на діаграмах класів? Описати зображення основних типів зв’язків.
4. Надати визначення моделі предметної області (МПО) в межах методології UP. На якій фазі ЖЦ UP будується МПО? Що таке концептуальний клас МПО? Як зображається МПО засобами UML?
5. Які методи використовуються для виявлення концептуальних класів?
6. Які методи використовуються для виявлення асоціацій концептуальних класів?
7. Як визначити атрибути концептуальних класів?
8. Перерахувати статичні діаграми UMl. З якою метою створюються діаграми классів?
9. Призначення, зображення та правила наіменування елементів діаграми класів: клас, атрибут, опреація.
10. Які типи зв’язків підтримуються на діаграмах класів? Описати зображення основних типів зв’язків.
11. Надати визначення моделі предметної області (МПО) в межах методології UP. На якій фазі ЖЦ UP будується МПО? Що таке концептуальний клас МПО? Як зображається МПО засобами UML?
12. Які методи використовуються для виявлення концептуальних класів?
13. Які методи використовуються для виявлення асоціацій концептуальних класів?
14. Як визначити атрибути концептуальних класів?

**Завдання до лабораторної роботи** **№4**

1. Відповісти на питання для самоконтролю.
2. Розробити модель предметної області для власного варіанту концептуальні класи, асоціації, атрибути).
3. Оформити протокол лабораторної роботи.