# Практична робота № 8 ПОБУДОВА ДІАГРАМ КОМПОНЕНТІВ

**Теоретичні відомості** ***Діаграми компонентів*** *(****component diagrams****)* відображають фізичне представлення програмної системи у вигляді сукупності елементів, які називають компонентами *(components).* Кожен компонент має ім’я, мову реалізації та перелік призначених класів. Фізично кожен компонент може бути представлений у вигляді окремого файлу, директорії чи програмного продукту. Графічний вигляд компоненту наведено на рис. 23.



**Рис. 23.** Графічне представлення компоненту в Rational Rose У залежності від обраного стереотипу компонент може належати до одного з наступних типів: Applet, Application, Database, DLL, EXE, MainProgram, SubProgram. Тип компонента є частиною його графічного представлення і не впливає на генерацію програмного коду.

Одним з елементів мови UML є також пакет (packаge). Головною відмінністю компонентів від пакетів є те, що пакет є логічним об’єднанням класів (аналог namespace в С++) за їх функціональністю, а компонент є окремим фізичним елементом системи, і може виступати в якості прикладної програми, підсистеми, бібліотеки тощо.

Компоненти поєднуються між собою за допомогою відношення залежності (dependency) відповідно до відношень між класами, що належать до цих компонентів.

Розглянемо побудову діаграми компонентів, виходячи з діаграми класів абстрактної системи. На рис. 24 відображено діаграму класів програмної системи, яка має так звану тришарову архітектуру.

Перший шар (позначений синім кольором у лівій частині діаграми) складається з сукупності класів, які формують інтерфейс користувача. До нього належить ієрархія діалогів системи та клас прав користувача, що використовується в діалозі авторизації.

Другий шар (сукупність класів у центральній частині діаграми, обмежена лінією зеленого кольору) відповідає за бізнес-логіку (перелік алгоритмів, що оперують даними для виконання функцій системи). Він сформований сукупністю класів алгоритмів та класом прав користувача на виконання тих чи інших операцій.

**Рис**

**. 24.**

Приклад

діаграми

класів

для

програмної

системи

з

тришаровою

архітектурою

**Практикум з об’єктно-орієнтованих методологій створення комп’ютерних систем**

Третій шар (сукупність класів у правій частині діаграми, обмежена лінією червоного кольору) забезпечує обмін даними з базою даних. Він складається зі специфічних структур даних, кожна з яких відповідає рядку відповідної таблиці бази даних, контейнерів структур даних, класу інтерфейсу бази даних та прав доступу до таблиць БД.

Програмний комплекс є поєднанням цих трьох шарів, кожен з яких може бути представленим окремою бібліотекою, які можуть бути реалізовані різними мовами програмування. На рис. 25 представлена діаграма компонентів для даного програмного комплексу.



**Рис. 25.** Діаграма компонентів програмного комплексу

***Розподіл класів за компонентами:***

1. MainComponent має стереотип Application та містить єдиний клас Application.
2. Presentation Logic містить класи Dialog, MainDialog, AuthDialog, UserPermission.
3. BusinessLogic містить класи Algorithm, AlgorithmType1, AlgorithmType2, AlgorithmType3, ProjectPermission.
4. DBAccessLogic має стереотип Database та містить класи

DBContainer, DBInterface, DatabasePermission, DataStructure1, DataStructure2, ContainerofDataStructures1, ContainerofDataStructures2.

Зверніть увагу, що класи UserPermission, ProjectPermission та DatabasePermissionможуть належати до одного пакету Security діаграми класів, але реалізуються в різних компонентах. Тобто, незважаючи на схожість пакетів та компонентів, вони можуть не бути ідентичними.

***Послідовність дій для призначення класів до компонентів:***

1. У контекстному меню одного з компонентів обрати

OpenSpecification.

1. Відкрити вкладку Realizes та призначити класи один за одним, натискаючи праву кнопку миші та обираючи пункт Assign.

Відношення залежності на діаграмі компонентів визначається відповідно до відношень класів компонентів на діаграмі класів. Клас Application пов’язаний відношенням направленої асоціації з класами компонентів PresentationLogic та BusinessLogic (залежить від них), тому між MainComponent та обома даними компонентами визначається відношення залежності (позначається штриховою стрілкою). Аналогічне твердження справедливе для пари компонентів BusinessLogic та

DBAccessLogic.

## Завдання

У середовищі Rational Rose створити діаграму компонентів для обраної програмної системи. Діаграма повинна містити не менше трьох компонентів. Розподілити всі класи між компонентами.

**Контрольні запитання** 1. Для чого призначена діаграма послідовності (sequence diagram)? Наведіть її основні елементи.

1. Дайте характеристику компонента. Що може виступати в ролі компонента? Наведіть приклад.
2. Назвіть головну відмінність між пакетами та компонентами.
3. Що таке 3 tier архітектура? Наведіть приклад системи, для якої можна використати дану архітектуру.
4. Які характеристики визначаються для кожного компонента на діаграмі?
5. Яким чином класи розподіляються за компонентами?
6. Яким чином визначаються відношення між компонентами?