70. Види UML-діаграм з короткою характеристикою кожної. Приклади.

Діаграма *UML* – це зображення у вигляді ***графа з вершинами (сутностями) і ребрами (відношеннями)***.

Основна ***мета*** діаграм – ***візуалізація архітектури*** розроблюваної системи з різних точок зору.

Діаграма – деякий зріз системи. Звичайно діаграми дають згорнуте представлення елементів, із яких складається розроблювана система. При цьому один і той самий елемент може бути присутнім у декількох (а іноді й в усіх) діаграмах.

При візуальному моделюванні з *UML* використовуються ***вісім видів діаграм***, кожна з яких може містити елементи певного типу. Типи можливих елементів і відношень між ними залежать від виду діаграми.

Діаграми прецедентів або діаграми використання (use case diagrams). Такі діаграми описують ***функціональність***, яка буде надаватись користувачам системи, котра проектується. Представляються шляхом використання ***прецедентів*** та ***акторів***, а також ***відношень між ними***. Набір усіх прецедентів діаграми фактично визначає ***функціональні вимоги***, за допомогою яких може бути сформульоване ***технічне завдання***.

Діаграми класів (class diagrams) описують статичну структуру класів. Дозволяють (на концептуальному рівні) формувати "словник предметної області" та (на рівні специфікацій і рівні реалізацій) визначати структуру класів у програмній реалізації системи. Можуть використовуватись для генерації каркасного програмного коду (в реальній мові програмування).

Для опису динаміки використовуються ***діаграми поведінки (behavior diagrams)***, що підрозділяються на

- ***діаграми станів (statechart diagrams);***
- ***діаграми діяльності (активності) (activity diagrams);***
- ***діаграми взаємодії (interaction diagrams),*** що у свою чергу підрозділяються на
  - ***діаграм послідовності (sequence diagrams);***
  - ***діаграм кооперації (співробітництва) (collaboration diagrams).***

І, нарешті, ***діаграми реалізації (implementation diagrams)*** поділяються на

- ***компонентні діаграми· (діаграми компонентів) (component diagrams);***
- ***діаграми розгортання· (deployment diagrams).***

71. Шаблон State. Призначення, структура, учасники. Порівняти реалізації з переключенням станів контекстом та конкретним стейтом.

Стан (англ. *State*) — шаблон проектування, відноситься до класу шаблонів поведінки.

Призначення

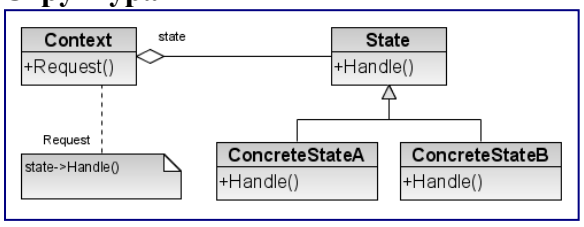
Дозволяє об'єктові варіювати свою поведінку у залежності від внутрішнього стану. Ззовні здається, що змінився клас об'єкта.

Застосовність

Слід використовувати шаблон Стан у випадках, коли:

- поведінка об'єкта залежить від його стану та повинно змінюватись під час виконання програми;
- у коді операцій зустрічаються умовні оператори, що складаються з багатьох частин, у котрих вибір гілки залежить від стану. Зазвичай у такому разі стан представлено константами, що перелічуються. Часто одна й та ж структура умовного оператору повторюється у декількох операціях. Шаблон Стан пропонує помістити кожну гілку у окремий клас. Це дозволить трактувати стан об'єкта як самостійний об'єкт, котрий можна змінитися незалежно від інших.

Структура



Context — контекст:

- визначає інтерфейс, що є корисним для клієнтів;
- зберігає екземпляр підкласу ConcreteState, котрим визначається поточний стан;

State — стан:

- визначає інтерфейс для інкапсуляції поведінки, асоційованої з конкретним станом контексту Context;

Підкласи ConcreteState — конкретні стани:

- кожний підклас реалізує поведінку, асоційовану з деяким станом контексту Context.

Відносини

- клас Context делегує залежні від стану запити до поточного об'єкта ConcreteState;
- контекст може передати себе у якості аргументу об'єкта State, котрий буде обробляти запит. Це надає можливість об'єкта-стану при необхідності отримати доступ до контексту;
- Context — це головний інтерфейс для клієнтів. Клієнти можуть конфігурувати контекст об'єктами стану State. Один раз зконфігурувавши контекст, клієнти вже не повинні напряму зв'язуватися з об'єктами стану;
- або Context, або підкласи ConcreteState можуть вирішити, за яких умов та у якій послідовності відбувається зміна станів.