**67. UML. Призначення. Історія створення.**

UML ( Unified Modeling Language — уніфікована мова моделювання) — мова графічного опису для об'єктного моделювання в області розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, називаною UML моделлю. UML був створений для визначення, візуалізації, проектування й документування здебільшого програмних систем.

Використання UML не обмежується моделюванням програмного забезпечення. Його також використовують для моделювання бізнес-процесів, системного проектування й відображення організаційних структур.

UML дозволяє розроблювачам ПЗ досягти угоди в графічних позначеннях для представлення загальних понять (таких як клас, компонент, узагальнення (generalization), об'єднання (aggregation) і поведінка) і більше сконцентруватися на проектуванні й архітектурі.

**Завдання UML.**

Мова UML призначена для рішення наступних завдань:

1.надати в розпорядження користувачів готову до використання виразну потужну мову візуального моделювання, що дозволяє розробляти осмислені моделі й обмінюватися ними;

2. передбачити внутрішні механізми розширюваності й спеціалізації базових концепцій мови;

3. забезпечити максимальну незалежність проекту створення програмного забезпечення від конкретних мов програмування й процесів розробки;

4. забезпечити формальну основу для однозначної інтерпретації мови;

5.стимулювати розширення ринку об’єктно-орієнтованих інструментальних засобів створення програмного забезпечення;

6. інтегрувати кращий практичний досвід використання мови й реалізації програмних засобів його підтримки.

**Історія**

У середині 90-х існувало більше ***50*** різних ***об'єктно-орієнтованих методів чи мов*** ***моделювання***. У цей же період часу оновлюються версії таких досить розповсюджених методів як: Booch'93, OMT-2 (Object Modelling Technique), Fusion, OOSE (Object-Oriented Software Engineering). І розроблювачів ПС, і замовників охоплювало занепокоєння при виборі метода проектування ПС, кожен із яких до того ж, як правило, спирався на власну нотацію. Отже, на часі визріла проблема в ***стандартизації й уніфікації*** підходів до моделювання.

Початком розробки *UML* вважається жовтень 1994 року, коли у *Rational Software Corporation* силами ***Греді Буча*** (***Grady Booch***) і ***Джима Рамбо*** (***Jim Rumbaugh***) була започаткована робота з уніфікації їх власних методів Booch'93 та OMT. Перша версія Уніфікованого Метода (Unified Method 0.8) була опублікована в жовтні 1995.

Трохи згодом, у тому ж 1995 році, до роботи приєднався ***Айвер Якобсон*** (***Ivar Jacobson***), залучаючи до процесу інтеграції й уніфікації ще один метод – власний метод *OOSE*.

Таким чином, на першому концептуальному етапі *UML* отримав трьох авторів: Буча, Рамбо і Якобсона, кожен із яких був ідеологом свого власного об'єктно-орієнтованого метода візуального моделювання

**68. Шаблон Memento. Призначення, структура, учасники. Порівняти з альтернативними рішеннями.**

Зні́мок (англ. Memento) — шаблон проектування, відноситься до класу шаблонів поведінки.

**Призначення**

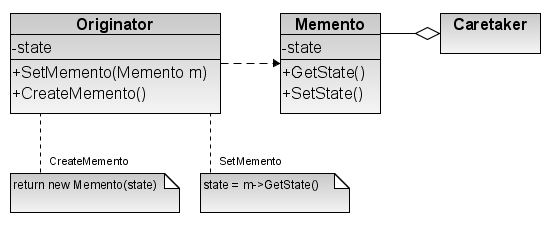
Не порушуючи інкапсуляції, фіксує та виносить за межі об'єкта його внутрішній стан так, щоб пізніше можна було відновити з нього об'єкт.

Слід використовувати шаблон Знімок у випадках, коли:

• необхідно зберегти миттєвий знімок стану об'єкта (або його частини), щоб згодом об'єкт можна було відтворити у тому ж самому стані;

• безпосереднє вилучення цього стану розкриває деталі реалізації та порушує інкапсуляцію об'єкта.

**Структура**

 • Memento — контекст:

• зберігає внутрішній стан об'єкта Originator. Обсяг інформації, що зберігається, може бути різним та визначається потребами хазяїна;

• забороняє доступ усім іншим об'єктам окрім хазяїна. По суті знімок має два інтерфейси. Опікун Caretaker користується лише вузьким інтерфейсом знімку — він може лише передавати знімок іншим об'єктам. Напроти, хазяїн користується широким інтерфейсом, котрий забезпечує доступ до всіх даних, необхідних для відтворення об'єкта (чи його частини) у попередньому стані. Ідеальний варіант — коли тільки хазяїну, що створив знімок, відкритий доступ до внутрішнього стану знімку;

Originator — хазяїн:

• створює знімок, що утримує поточний внутрішній стан;

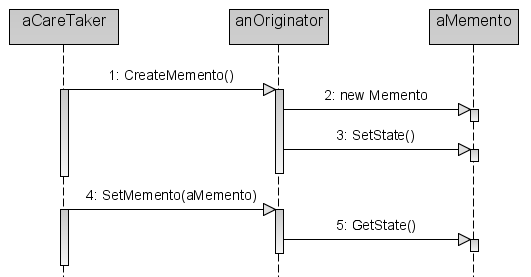
• використовує знімок для відтворення внутрішнього стану;

CareTaker — опікун:

• відповідає за зберігання знімку;

• не проводить жодних операцій над знімком та не має уяви про його внутрішній зміст.

**Відносини**

• опікун запитує знімок у хазяїна, деякий час тримає його у себе, опісля повертає хазяїну. Іноді цього не відбувається, бо хазяїн не має необхідності відтворювати свій попередній стан; 

• знімки пасивні. Тільки хазяїн, що створив знімок, має доступ до інформації про стан.