# Унифицированный язык моделирования

# UML — это язык моделирования

**модель UML** — это, прежде всего, описание объекта или явления, а также и кое-что другое, а именно все, что авторам UML удалось включить в язык, не нарушая принципа унификации.

# НАЗНАЧЕНИЕ UML

**Язык UML** — это графический язык моделирования общего назначения, предназначенный для спецификации, визуализации, проектирования и документирования всех артефактов, создаваемых при разработке программных систем.

# Спецификация

Одним из ключевых этапов разработки приложения является определение того, каким требованиям должно удовлетворять разрабатываемое приложение. В результате этого этапа появляется формальный или неформальный документ (артефакт), который называют по-разному, имея в виду примерно одно и то же: постановка задачи, требования, техническое задание, внешние спецификации и др.

**Спецификация** — это декларативное описание того, как нечто устроено или работает.

**Основное назначение UML** — предоставить, с одной стороны, достаточно формальное, с другой стороны, достаточно удобное, и, с третьей стороны, достаточно универсальное средство, позволяющее до некоторой степени снизить риск расхождений в толковании спецификаций.

- **Во-первых**, UML не является языком программирования.
- Во-вторых, UML не является спецификацией инструмента.
- В-третьих, UML не является моделью процесса разработки приложений.

# Способы использования UML

- Рисование картинок.
- Обмен информацией.
- Спецификация систем. *Это важнейший способ использования UML.*
- Повторное использование архитектурных решений.
- Генерация кода.
- Имитационное моделирование.
- Верификация моделей.

# МОДЕЛЬ И ЕЕ ЭЛЕМЕНТЫ

**Модель UML (UML model)** — это совокупность конечного множества конструкций языка, главные из которых — это сущности и отношения между ними.

# Сущности

- структурные;
- поведенческие;
- группирующие;
- аннотационные.

# К структурным сущностям относят следующие:

- Объект (object) — сущность, обладающая уникальностью и инкапсулирующая в себе состояние и поведение.
- Класс (class) — описание множества объектов с общими атрибутами, определяющими состояние, и операциями, определяющими поведение.
- Интерфейс (interface) — именованное множество операций, определяющее набор услуг, которые могут быть запрошены потребителем и предоставлены поставщиком услуг.

- Кооперация (collaboration) — совокупность объектов, которые взаимодействуют для достижения некоторой цели.
- Действующее лицо (actor) — сущность, находящаяся вне моделируемой системы и непосредственно взаимодействующая с ней.
- Компонент (component) — модульная часть системы с четко определенным набором требуемых и предоставляемых интерфейсов.

- Артефакт (artifact) — элемент информации, который используется или порождается в процессе разработки программного обеспечения. Другими словами, артефакт — это физическая единица реализации, получаемая из элемента модели (например, класса или компонента).

- Узел (node) — вычислительный ресурс, на котором размещаются и при необходимости выполняются артефакты.

# Поведенческие сущности предназначены для описания поведения.

- Состояние (state) — период в жизненном цикле объекта, находясь в котором объект удовлетворяет некоторому условию и осуществляет собственную деятельность или ожидает наступления некоторого события.
- Деятельность (activity) можно считать частным случаем состояния, который характеризуется продолжительными (по времени) не атомарными вычислениями.
- Действие (action) — примитивное атомарное вычисление.

# Сущность — вариант использования

- Вариант использования (use case) — множество сценариев, объединенных по некоторому критерию и описывающих последовательности производимых системой действий, доставляющих значимый для некоторого действующего лица результат.

# Группирующая сущность в UML одна — пакет — зато универсальная.

- Пакет (package) — группа элементов модели (в том числе пакетов).

**Аннотационная сущность тоже одна — примечание (comment) — зато в нее можно поместить все что угодно, так как содержание примечания UML не ограничивает.**

## Нотация основных сущностей

| Название | Графическая нотация |
|---|---|
| Артефакт | «artifact» Requirement Specification; «library» QT |
| Вариант использования | Make Order; Make Order |
| Действующее лицо | Customer; Customer |
| Деятельность и действие | Display main menu; a=15 |
| Интерфейс | IAudio; «interface» IAudio |
| Класс | Product; Order {abstract} |
| Компонент | DataBase; «component» DataBase |

*Окончание табл. 1.1*

| Название | Графическая нотация |
|---|---|
| Кооперация | Visitor |
| Объект | :Rectangle |
| Пакет | Analysis Model |
| Примечание | Здесь находится комментарий |
| Состояние | Logged<br>entry/OpenLog<br>exit/CloseLog |
| Узел | Server |

# Отношения

В UML используются четыре основных типа отношений:

- зависимость (dependency);
- ассоциация (association);
- обобщение (generalization);
- реализация (realization).

**Зависимость — это наиболее общий тип отношения между двумя сущностями.**

**Отношение зависимости указывает на то, что изменение независимой сущности каким-то образом влияет на зависимую сущность. Графически отношение зависимости изображается в виде пунктирной линии со стрелкой (1), направленной от зависимой сущности (2) к независимой (3) (рис. 1.3).**

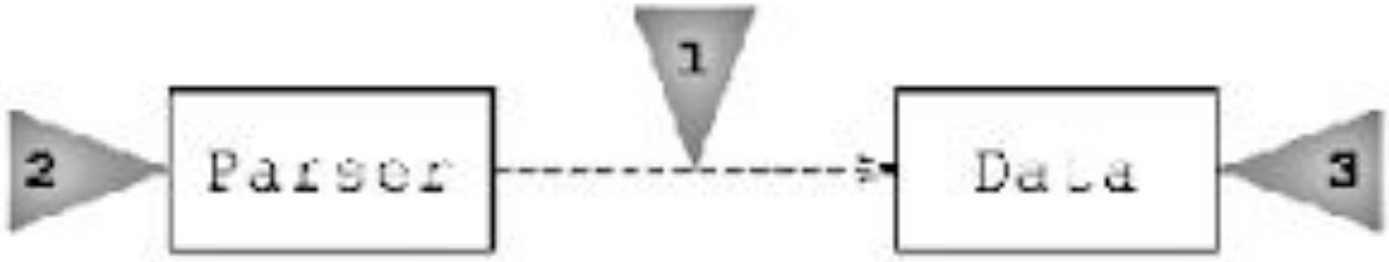


Рис. 1.3. Отношение зависимости

Ассоциация — это наиболее часто используемый тип отношения между сущностями.

Отношение ассоциации имеет место, если одна сущность непосредственно связана с другой (или с другими — ассоциация может быть не только бинарной).

Графически ассоциация изображается в виде сплошной линии (1) с различными дополнениями, соединяющей связанные сущности (рис. 1.4).

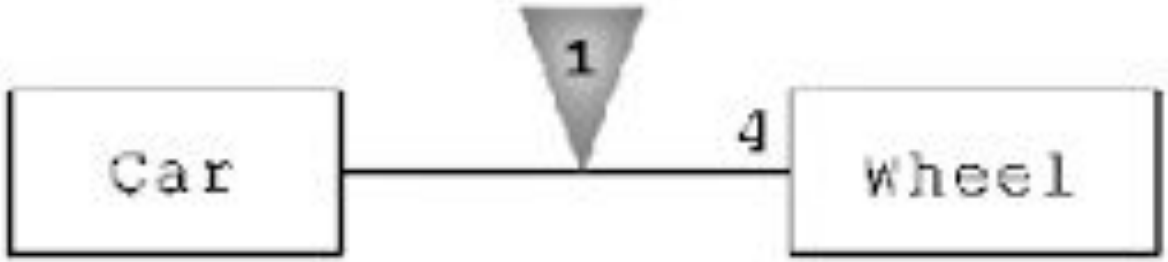


Рис. 1.4. Отношение ассоциации

**Обобщение — это отношение между двумя сущностями, одна их которых является частным (специализированным) случаем другой.**

**Графически обобщение изображается в виде линии с треугольной незакрашенной стрелкой на конце (1), направленной от частного (2) (подкласса) к общему (3) (суперклассу) (рис. 1.5).**

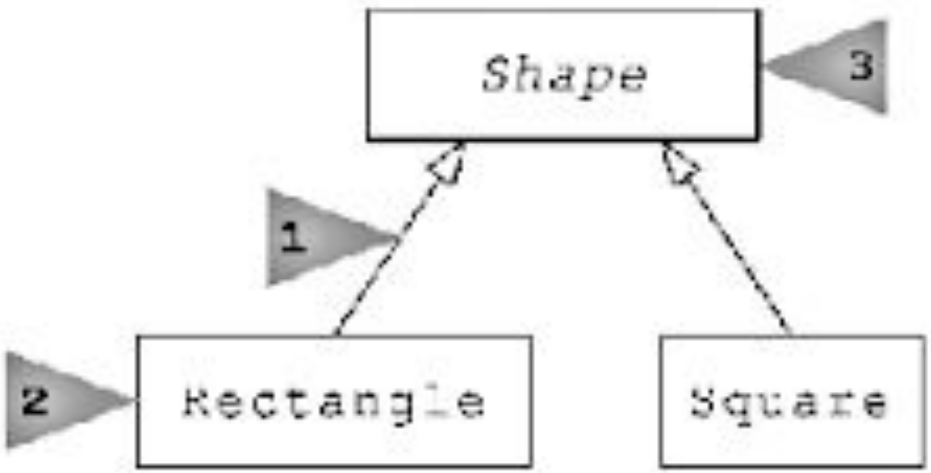


Рис. 1.5. Отношение обобщения

Отношение реализации используется несколько реже, чем предыдущие три типа отношений, поскольку часто подразумеваются по умолчанию.

Отношение реализации указывает, что одна сущность является реализацией другой. Например, класс является реализацией интерфейса.

Графически реализация изображается в виде пунктирной линии с треугольной незакрашенной стрелкой на конце (1), направленной от реализующей сущности (2) к реализуемой (3) (рис. 1.6).

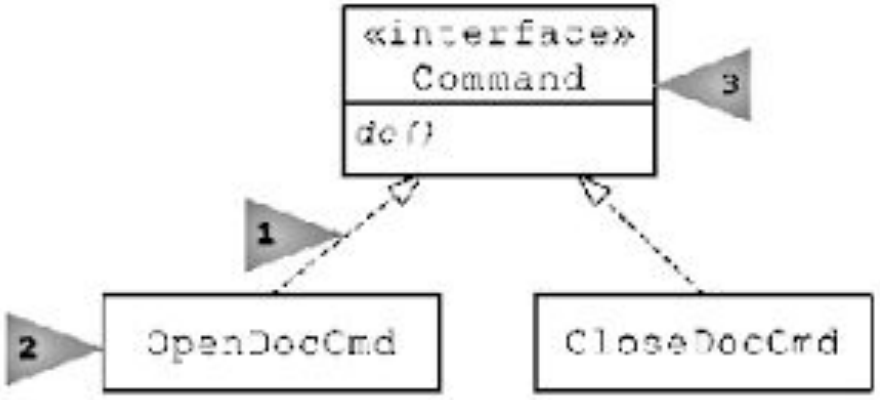


Рис. 1.6. Отношение реализации