Тема 1. Введение в UML

Унифицированный язык объектно-ориентированного моделирования Unified Modeling Language (UML).

UML — это язык.

Язык — это знаковая система для хранения и передачи информации.

Различаются языки ***формальные,*** *правила употребления которых строго и явно* определены и ***неформальные****, употребление которых основано на сложившейся* практике.
Различаются также языки ***естественные****, появляющиеся как бы сами* собой в результате неперсонифицированных усилий массы людей и языки ***искусственные****, являющиеся плодом видимых усилий определенных лиц.*

UML можно охарактеризовать как формальный искусственный язык. Признаком искусственности служит наличие трех общепризнанных авторов.

В то же время в формирование языка внесли вклад многие теоретики и разработчики, имя которым легион. Языкотворческая практика применительно к UML непрерывно продолжается, что дает основание считать UML до некоторой степени естественным языком.

UML - это язык документирования. UML предоставляет выразительные средства для создания визуальных моделей, которые:

- единообразно понимаются всеми разработчиками, вовлеченными в проект и являются средством коммуникации в рамках проекта.

Унифицированный Язык Моделирования (UML):

- не зависит от объектно-ориентированных (ОО) языков программирования,
- не зависит от используемой методологии разработки проекта,
- может поддерживать любой ОО язык программирования.

UML является *открытым* и обладает средствами расширения базового ядра. На UML можно содержательно описывать классы, объекты и компоненты в различных предметных областях, часто сильно отличающихся друг от друга.

Где используется UML

Язык UML предназначен прежде всего для разработки программных систем. Его использование особенно эффективно в следующих областях:

- информационные системы масштаба предприятия;
- банковские и финансовые услуги;
- телекоммуникации;
- транспорт;
- оборонная промышленность, авиация и космонавтика;
- розничная торговля;
- медицинская электроника;
- наука;
- распределенные Web-системы.

Сфера применения UML не ограничивается моделированием программного обеспечения. Его выразительность позволяет моделировать, скажем, документооборот в юридических системах, структуру и функционирование системы обслуживания пациентов в больницах, осуществлять проектирование аппаратных средств.

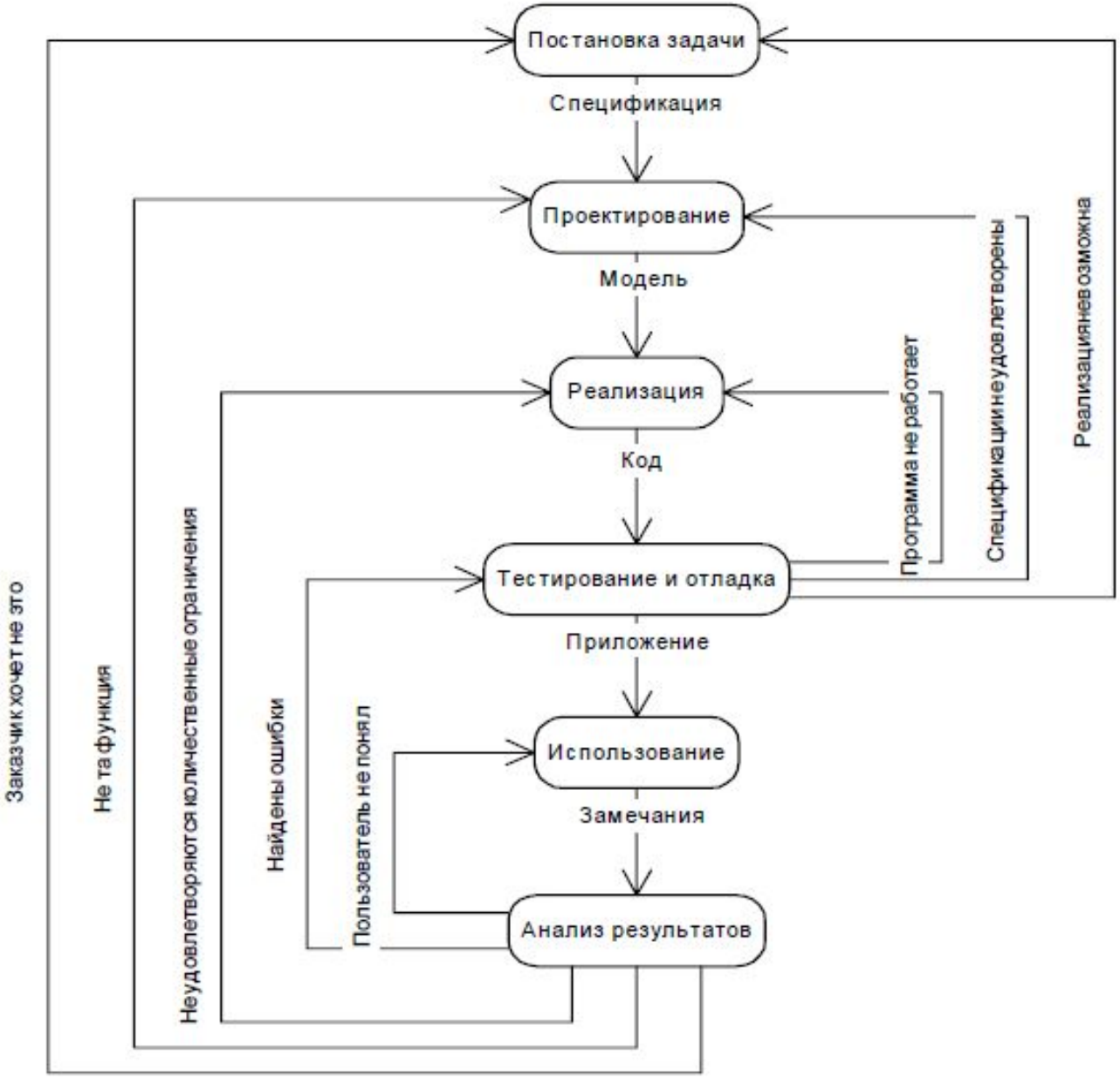
Особенности , как *точки вариации семантики и стандартные механизмы расширения, заметно отличают UML от языков, которые, по* общему мнению, являются образцами формализма.

- Синтаксис - определение правил конструирования выражений языка
- Семантика - определение правил приписывания смысла выражениям языка.
- Прагматика - определение правил использования выражений языка для достижения определенных целей.

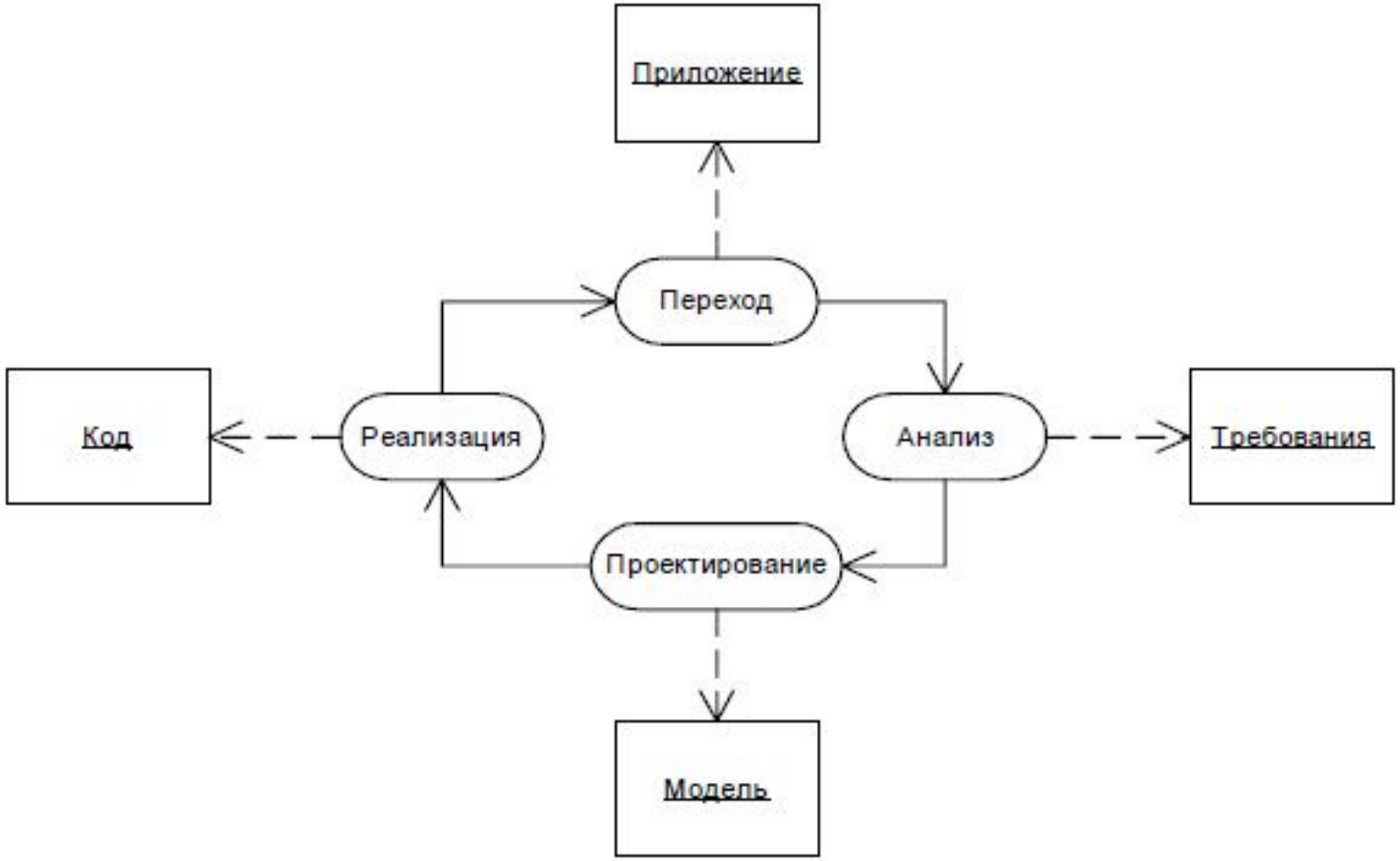
Слово "моделирование", входящее в название UML, имеет множество смысловых оттенков и сложившихся способов употребления.

Обычно совокупность и последовательность этих изменений называется *жизненный цикл.*

Жизненный цикл приложения



Большинство используемых в настоящее время моделей разработки носят циклический характер (так называемая *итеративная или инкрементальная* разработка).



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

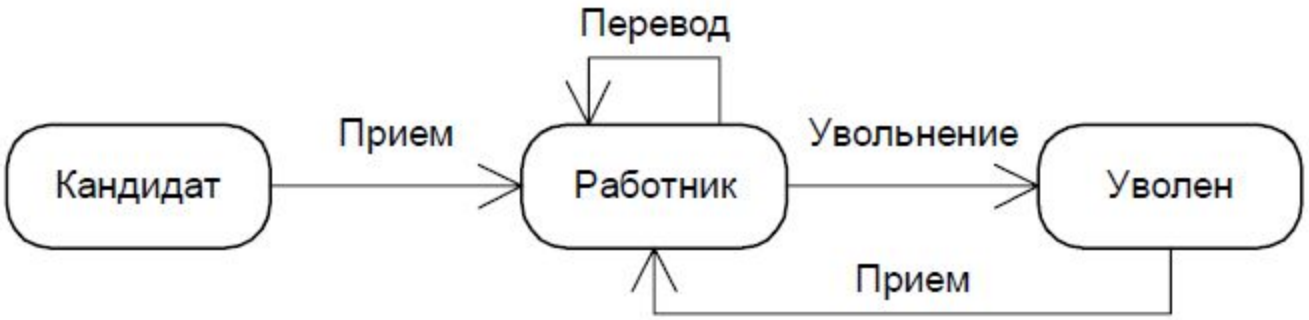
Необходимо принимать во внимание три толкования спецификаций.

- То, которое имеет в виду действующее лицо, являющееся источником спецификации (например, заказчик).
- То, которое имеет в виду действующее лицо, являющееся потребителем спецификации (например, разработчик).
- То, которое объективно обусловлено природой специфицируемого объекта.

Основное назначение UML — предоставить формальное, удобное и универсальное средство, позволяющее до некоторой степени снизить риск расхождений в толковании спецификаций.

Модели UML допускают представление в форме картинок, причем эти картинки наглядны, интуитивно понятны, практически однозначно интерпретируются и легко составляются.

Жизненный цикл работника на предприятии



Модели UML являются артефактами, которые можно хранить и использовать как в форме электронных документов, и в виде твердой копии.

Метод определения UML

В описании UML используются три языковых уровня.

- Мета-метамодель, то есть описание языка, на котором описана метамодель.
- Метамодель, то есть описание языка, на котором описываются модели.
- Модель, то есть описание самой моделируемой предметной области.

Терминология и нотация

Типы элементов нотации четыре:

- фигуры;
- линии;
- значки;
- тексты.

Модель и ее элементы

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

Сами сущности и отношения модели являются экземплярами метаклассов метамодели.

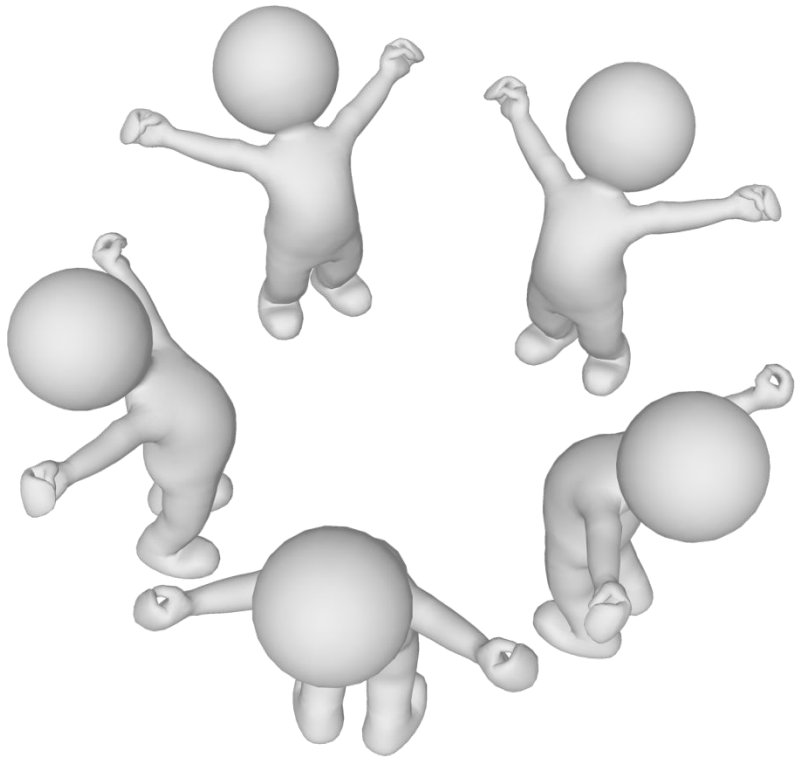
Словарь языка UML включает три вида строительных блоков:

- сущности;
- отношения;
-диаграммы.

Сущности - это абстракции, являющиеся основными элементами модели. Отношения связывают различные сущности; диаграммы группируют представляющие интерес совокупности сущностей.

Для удобства обзора сущности в UML можно подразделить на четыре группы:

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

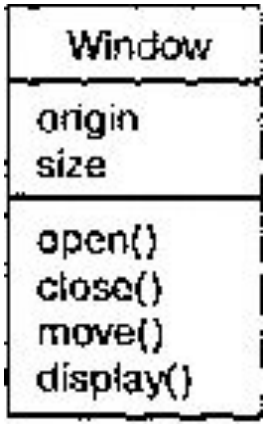


Структурные сущности - это имена существительные в моделях на языке UML. Они представляют собой статические части модели, соответствующие концептуальным или физическим элементам системы.

К структурным сущностям относят следующие.

Объект (object) 1 – сущность, обладающая уникальностью и инкапсулирующая в себе состояние и поведение.

Класс (class) 2 – описание множества объектов с общими атрибутами, определяющими состояние, и операциями, определяющими поведение.

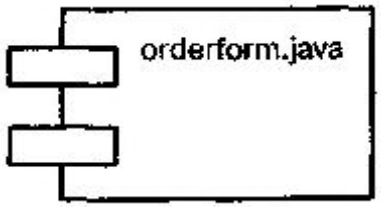


Интерфейс (interface) 3 – именованное множество операций, определяющее набор услуг, которые могут быть запрошены потребителем и предоставлены поставщиком услуг.

Кооперация (collaboration) 4 – совокупность объектов, которые взаимодействуют для достижения некоторой цели.

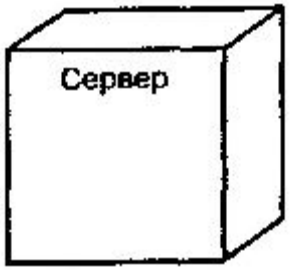
Действующее лицо (actor) 5 – сущность, находящаяся вне моделируемой системы и непосредственно взаимодействующая с ней.

Компонент∇ (component) 6 – модульная часть системы с четко определенным набором требуемых и предоставляемых интерфейсов.



Прецедент (Use case) - это описание последовательности выполняемых системой действий, которая производит наблюдаемый результат, значимый для какого-то определенного актера (Actor).

Узел (node) 8 – вычислительный ресурс, на котором размещаются и при необходимости выполня



Эти семь базовых элементов - классы, интерфейсы, кооперации, прецеденты, активные классы, компоненты и узлы - являются основными структурными сущностями, которые могут быть включены в модель UML.

Существуют также разновидности этих сущностей: актеры, сигналы, утилиты (виды классов), процессы и нити (виды активных классов), приложения, документы, файлы, библиотеки, страницы и таблицы (виды компонентов).

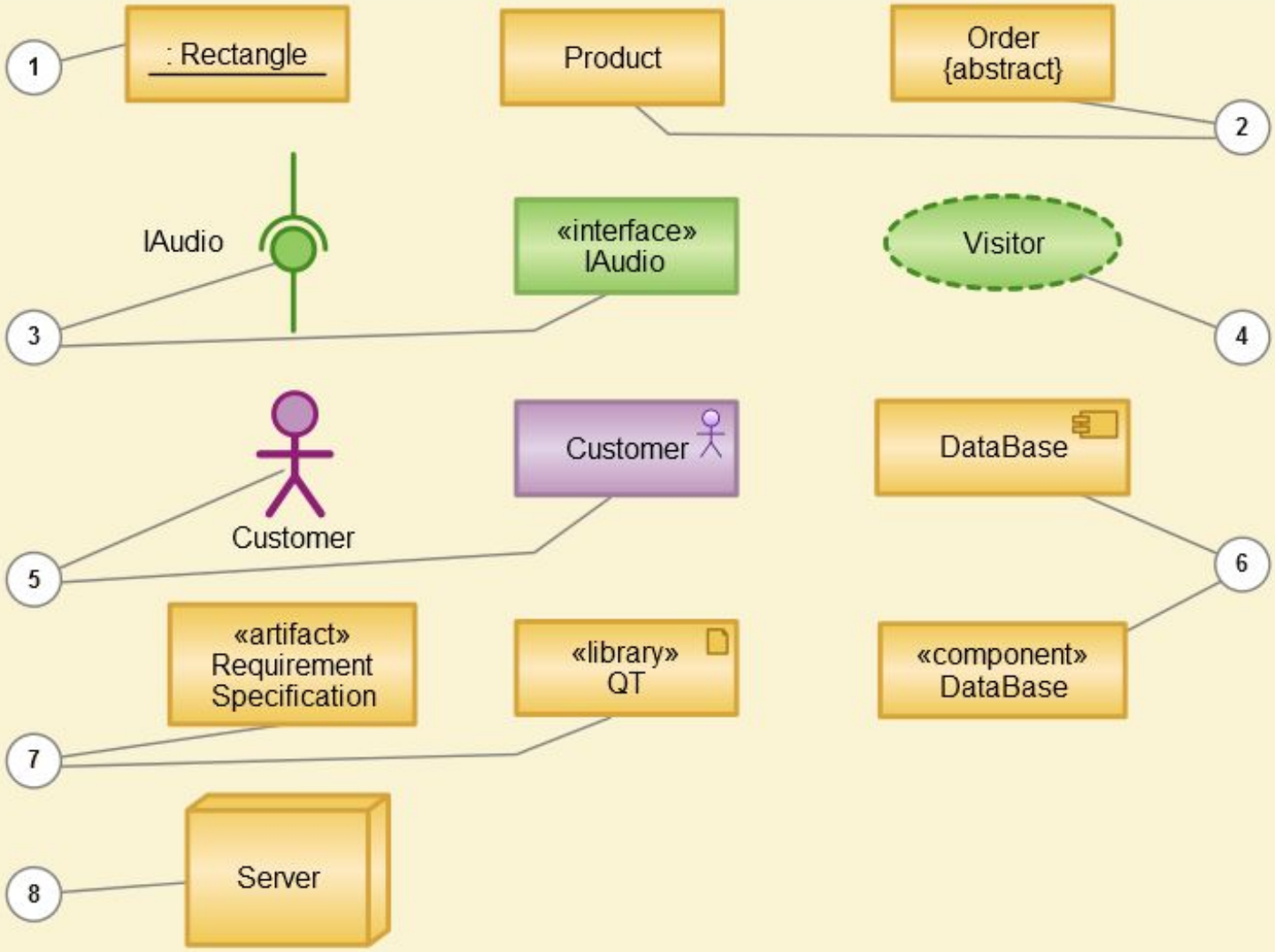


Рис. **Нотация структурных сущностей**

Поведенческие сущности предназначены для описания поведения. Основных поведенческих сущностей всего две: состояние и действие.

Состояние (state) 1 – период в жизненном цикле объекта, находясь в котором объект удовлетворяет некоторому условию и осуществляет собственную деятельность или ожидает наступления некоторого события.

Деятельность (activity) 2 можно считать частным случаем состояния, который характеризуется продолжительными (по времени) не атомарными вычислениями.

Действие (action) 3 – примитивное атомарное вычисление.

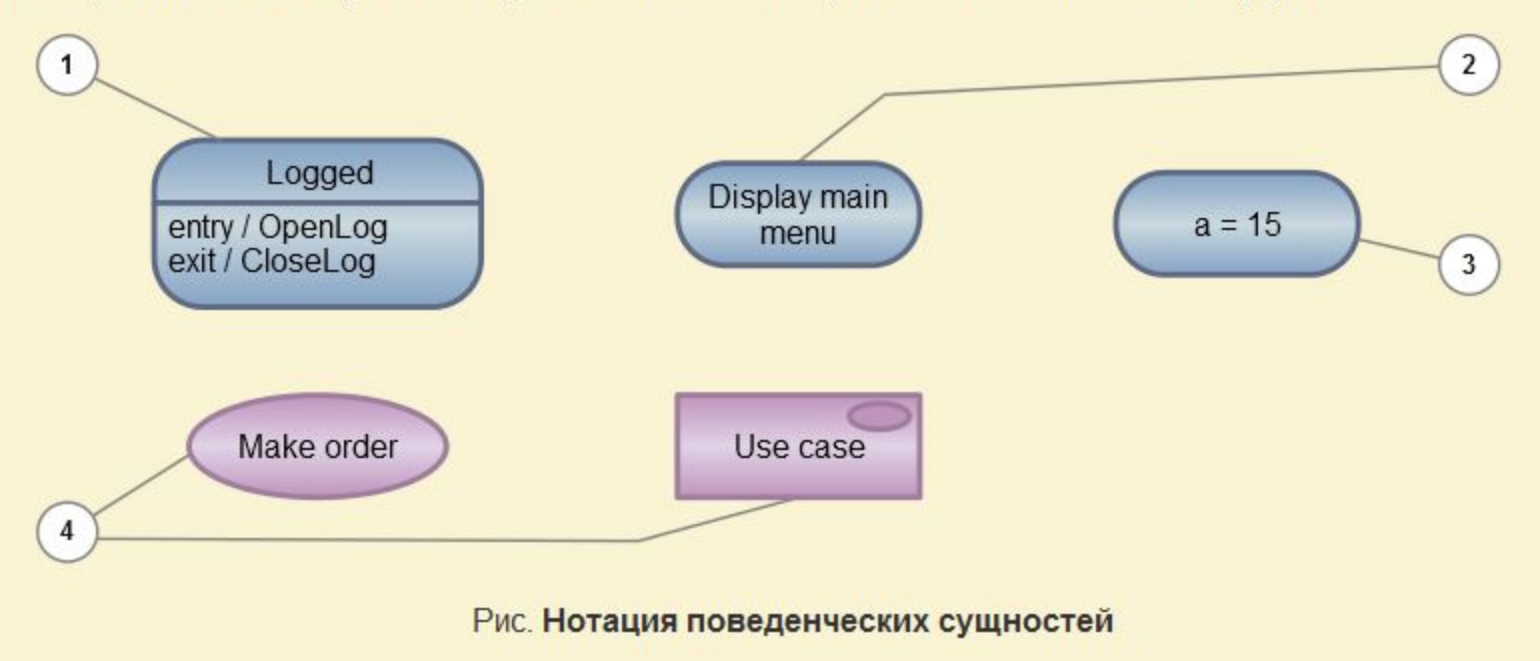


Рис. **Нотация поведенческих сущностей**

Вариант использования (use case) 4 – множество сценариев, объединенных по некоторому критерию и описывающих последовательности производимых системой действий, доставляющих значимый для некоторого действующего лица результат.

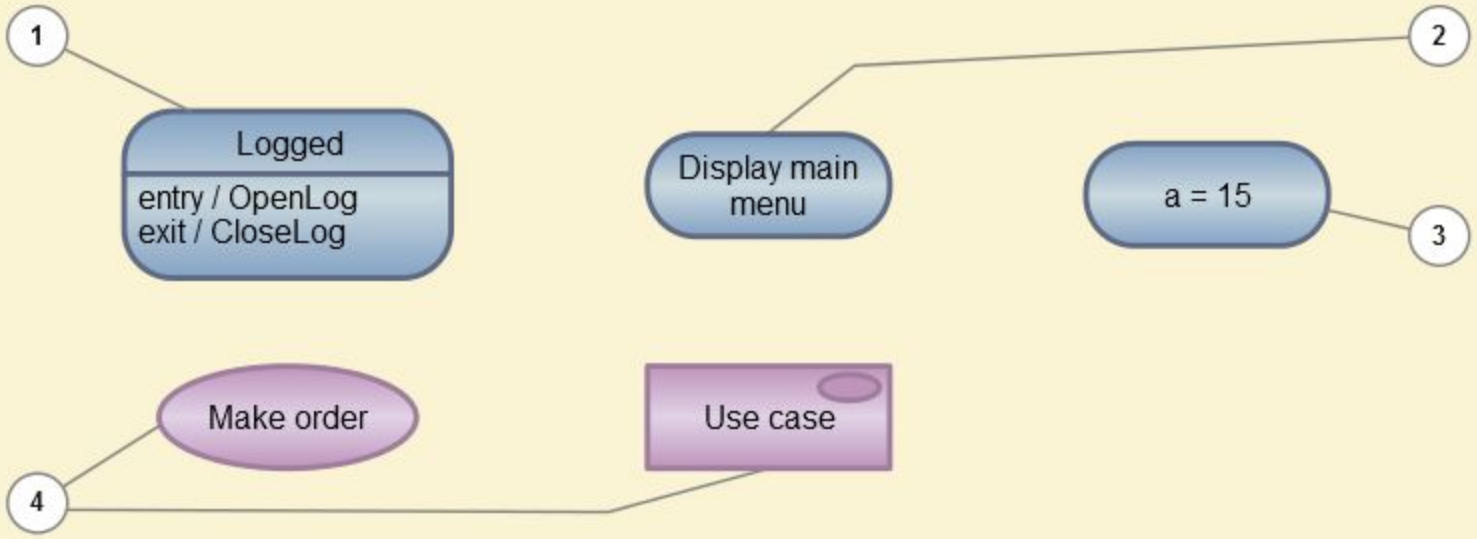


Рис. **Нотация поведенческих сущностей**

Группирующая сущность в UML одна – пакет – зато универсальная.

Пакет (package) 1 – группа элементов модели (в том числе пакетов).

Аннотационная сущность тоже одна – комментарий (примечание).

Комментарий (comment) 2 – произвольное по формату и содержанию описание одного или нескольких элементов модели.

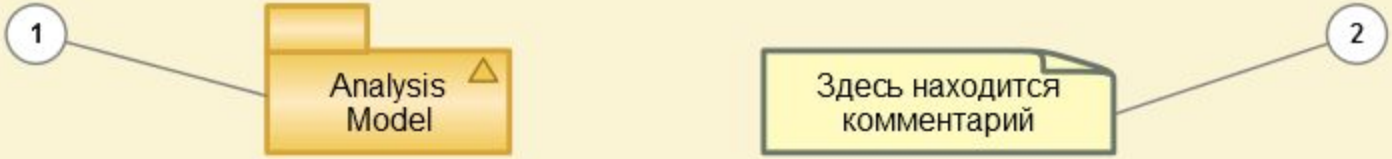
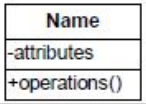
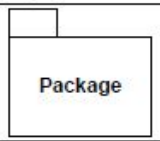
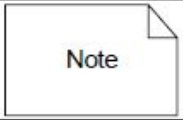
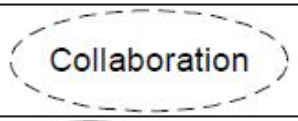
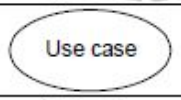
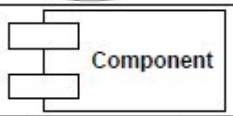
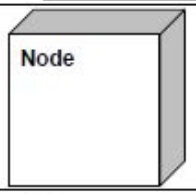
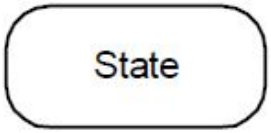


Рис. **Нотация группирующей и аннотационной сущностей**

Класс	
Действующее лицо	
Интерфейс	Interface ○

Пакет	
Примечание	

Кооперация	
Вариант использования	
Компонент	
Узел	
Состояние	
Деятельность	
Развилка / слияние	
Ветвление / соединение	

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.

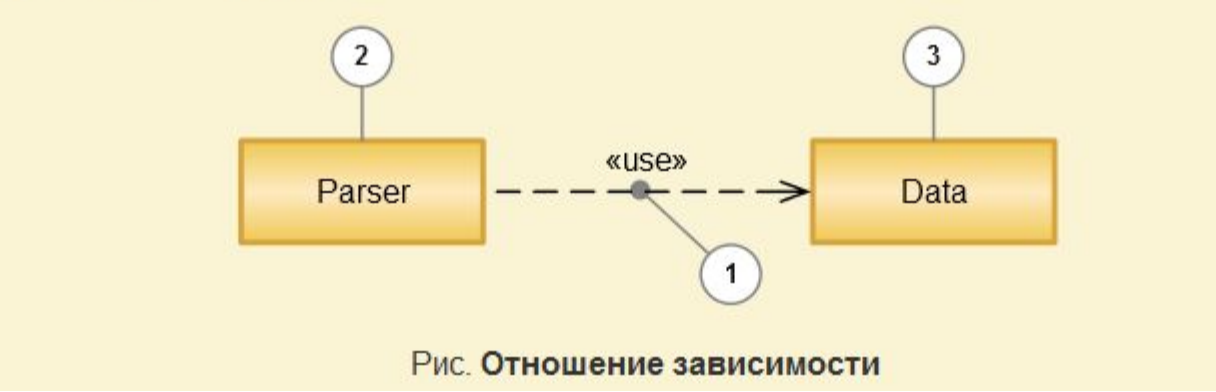


Рис. **Отношение зависимости**

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.

На программном уровне непосредственная связь может быть реализована различным образом, главное, что ассоциированные сущности знают друг о друге. Например, отношение часть-целое является частным случаем ассоциации и называется отношением агрегации.

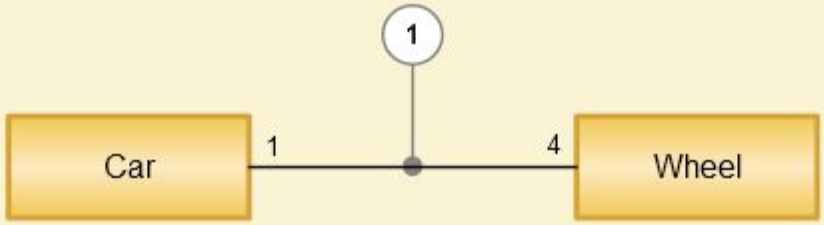


Рис. **Отношение ассоциации**

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.

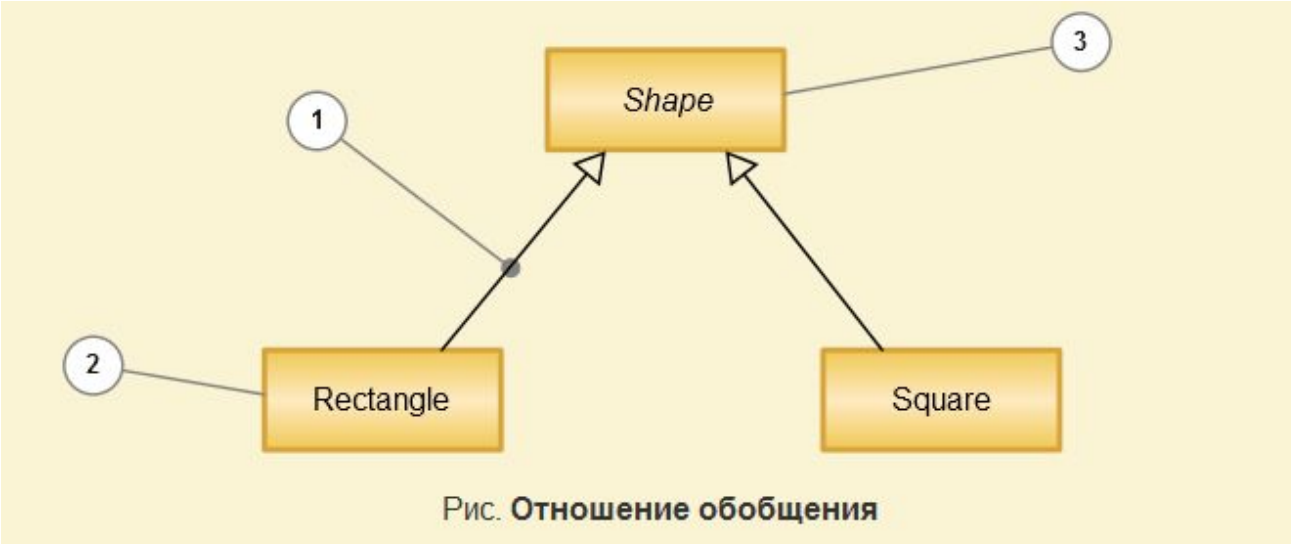


Рис. **Отношение обобщения**

Отношение реализациии 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.

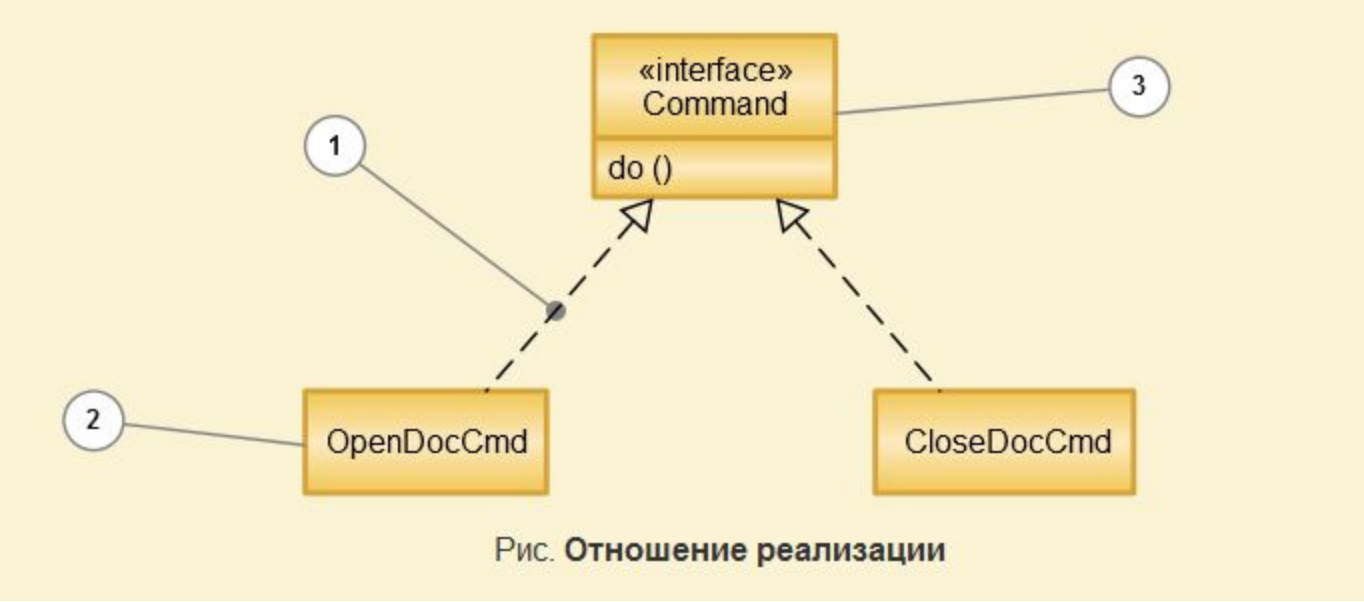


Рис. **Отношение реализации**

Как проектировать ИС в объектах/компонентах

Процесс проектирования с использованием той или иной визуальной нотации принято называть **методологией проектирования**, и все нотации, предшествующие UML, использовались в рамках соответствующей методологии.

Методологию трудно стандартизировать, и UML – это только *нотация*, которая может использоваться в рамках разных методологий. Одной из таких методологий является Rational Unified Process (RUP) - методология фирмы Rational Software.

RUP описывает успешно проверенные на практике подходы к созданию ИС и определяет организацию коллективной работы над проектом на основе следующих принципов:

- итерационная разработка проекта,
- управление требованиями,
- использование компонентной архитектуры,
- визуальное моделирование,
- тестирование качества ИС.

Диаграммы

Диаграммы UML - накладываемая на модель структура, которая облегчает создание и использование модели.
Диаграмма — это графическое представление некоторой части графа модели.

Классификация диаграмм

В UML 1.x всего определено 9 канонических типов диаграмм:
- Диаграмма использования
- Диаграмма классов
- Диаграмма объектов
- Диаграмма состояний
- Диаграмма деятельности
- Диаграмма последовательности
- Диаграмма кооперации
- Диаграмма компонентов
- Диаграмма размещения

Иерархия типов диаграмм

Диаграмма использования

Диаграмма использования — это наиболее общее представление функционального назначения системы.

Диаграмма использования призвана ответить на главный вопрос моделирования: что делает система во внешнем мире?

На диаграмме использования применяются два типа основных сущностей: *варианты использования* и *действующие лица*, между которыми устанавливаются следующие основные типы отношений:

- ассоциация между действующим лицом и вариантом использования;
- обобщение между действующими лицами;
- обобщение между вариантами использования;
- зависимости (различных типов) между вариантами использования.

Нотация диаграммы использования (UML 2.0)

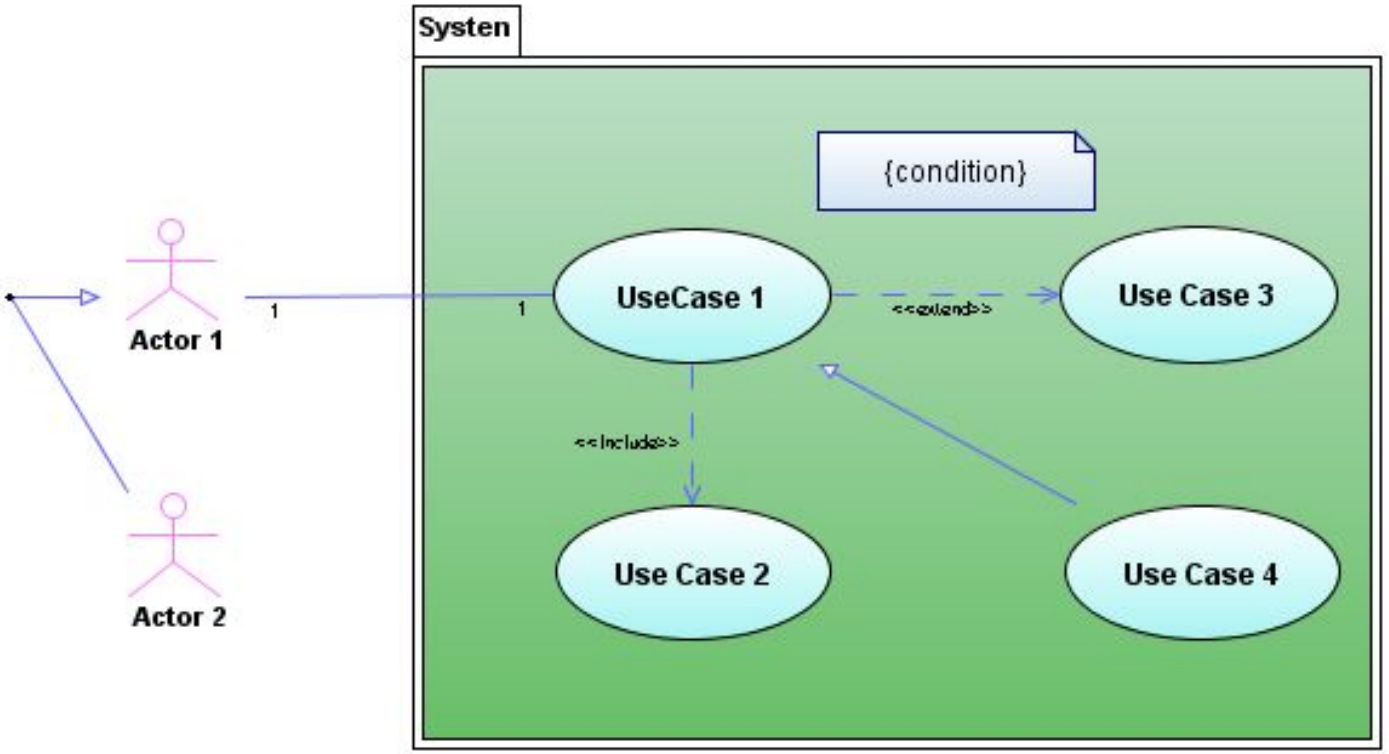


Диаграмма классов

Диаграмма классов — основной способ описания структуры системы.

На диаграмме классов применяются один основной тип сущностей: классы (включая многочисленные частные случаи классов: интерфейсы, типы, классы - ассоциации и многие другие), между которыми устанавливаются следующие основные типы отношений:

- ассоциация между классами (с множеством дополнительных подробностей);
- обобщение между классами;
- зависимости (различных типов) между классами и интерфейсами.

Нотация диаграммы классов

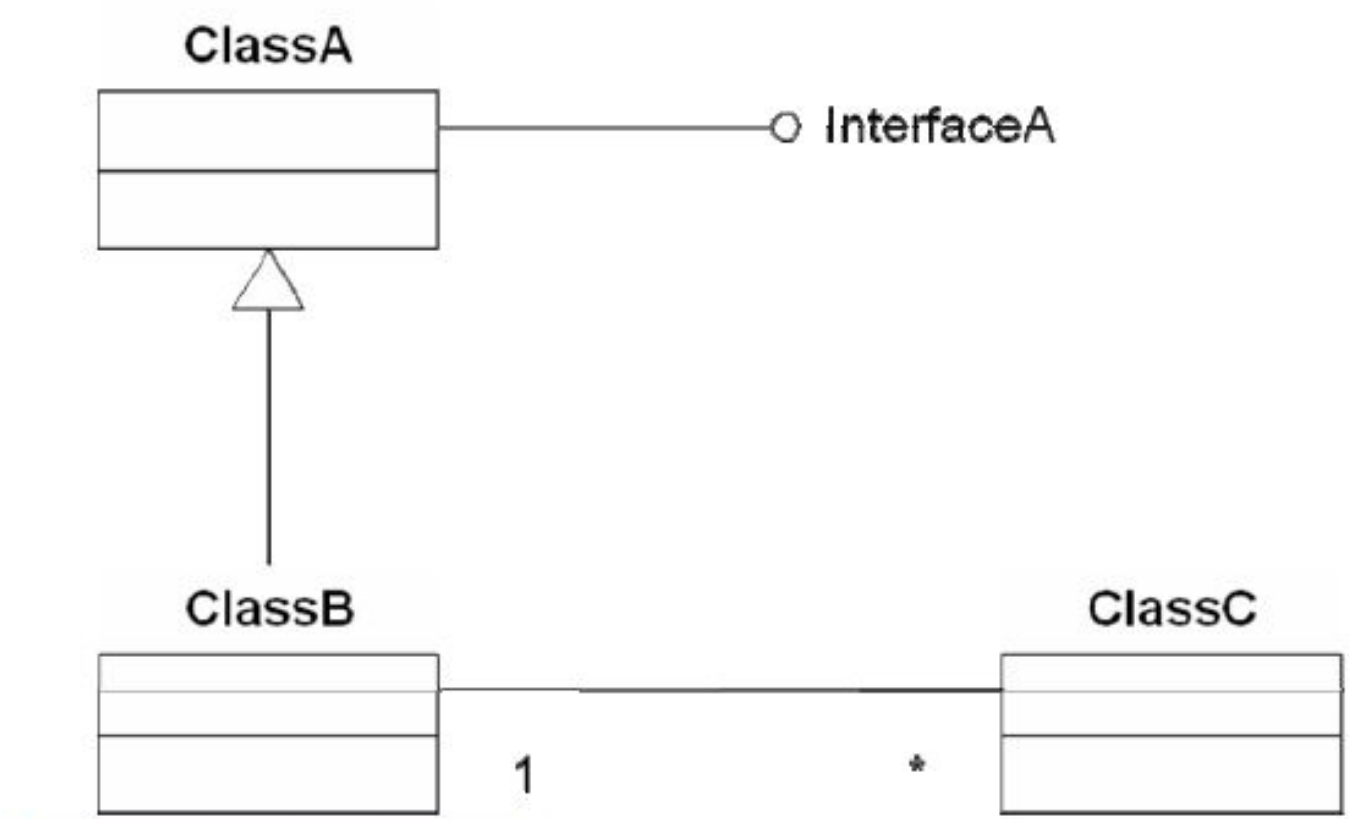


Диаграмма объектов

Диаграмма объектов — это частный случай диаграммы классов. Диаграммы объектов имеют вспомогательный характер — по сути это примеры, показывающие, какие имеются объекты и связи между ними в некоторый конкретный момент функционирования системы.

На диаграмме объектов применяют один основной тип сущностей: объекты (экземпляры классов), между которыми указываются конкретные связи (экземпляры ассоциаций).

Нотация диаграммы объектов

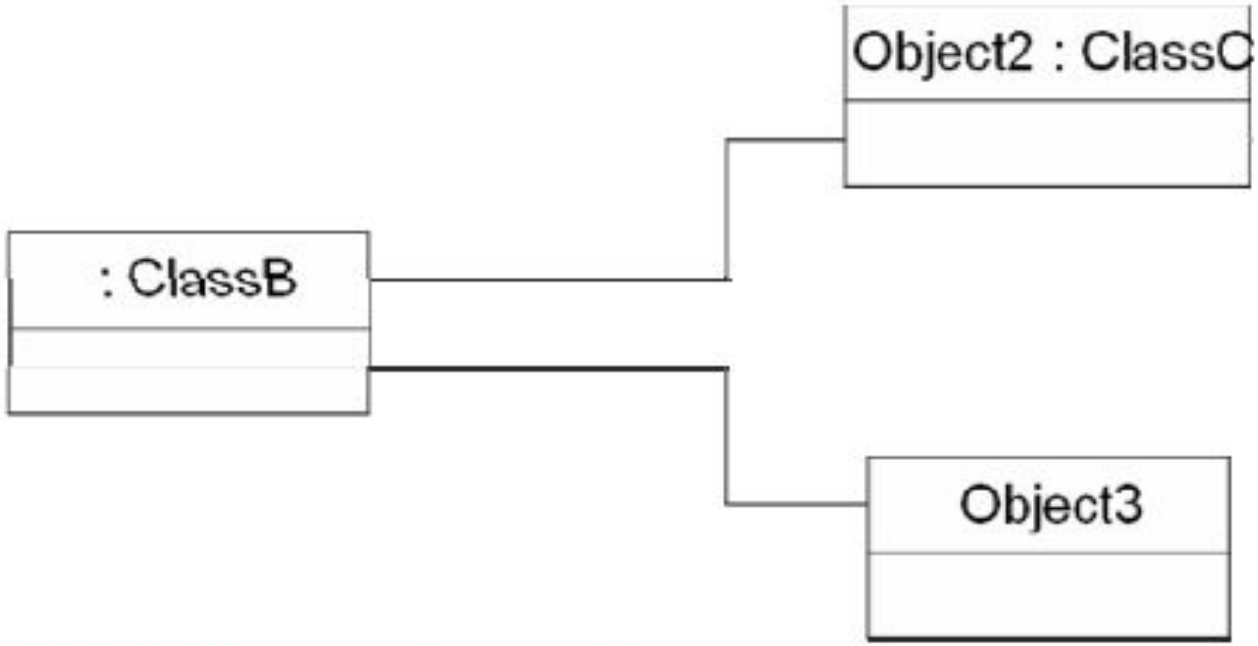


Диаграмма состояний

Диаграмма состояний — это основной способ детального описания поведения в UML. В сущности, диаграммы состояний представляют собой граф состояний и переходов конечного автомата, нагруженный множеством дополнительных деталей и подробностей.

На диаграмме состояний применяют один основной тип сущностей — состояния, и один тип отношений — переходы, но и для тех и для других определено множество разновидностей, специальных случаев и дополнительных обозначений.

Нотация диаграммы состояний (UML 1.x)

Диаграмма деятельности

Диаграмма деятельности — это, фактически, старая добрая блок-схема алгоритма, в которой модернизированы обозначения, а семантика согласована с современным объектно-ориентированным подходом, что позволило органично включить диаграммы деятельности в UML.

На диаграмме деятельности применяют один основной тип сущностей — деятельность, и один тип отношений — переходы (передачи управления), а также графические обозначения (развилки, слияния и ветвления), которые похожи на сущности, но таковыми на самом деле не являются, а представляют собой графический способ изображения некоторых частных случаев гипердуг в гиперграфе.

Помимо потока управления на диаграмме деятельности можно показать и поток данных, используя такую сущность, как объект (в определенном состоянии) и соответствующую зависимость. Кроме того, на диаграмме деятельности можно применить специальный графический комментарий — так называемые дорожки — подчеркивающие, что некоторые деятельности отличаются друг от друга, например, выполняются в разных местах.

Нотация диаграммы деятельности (UML 2.0)

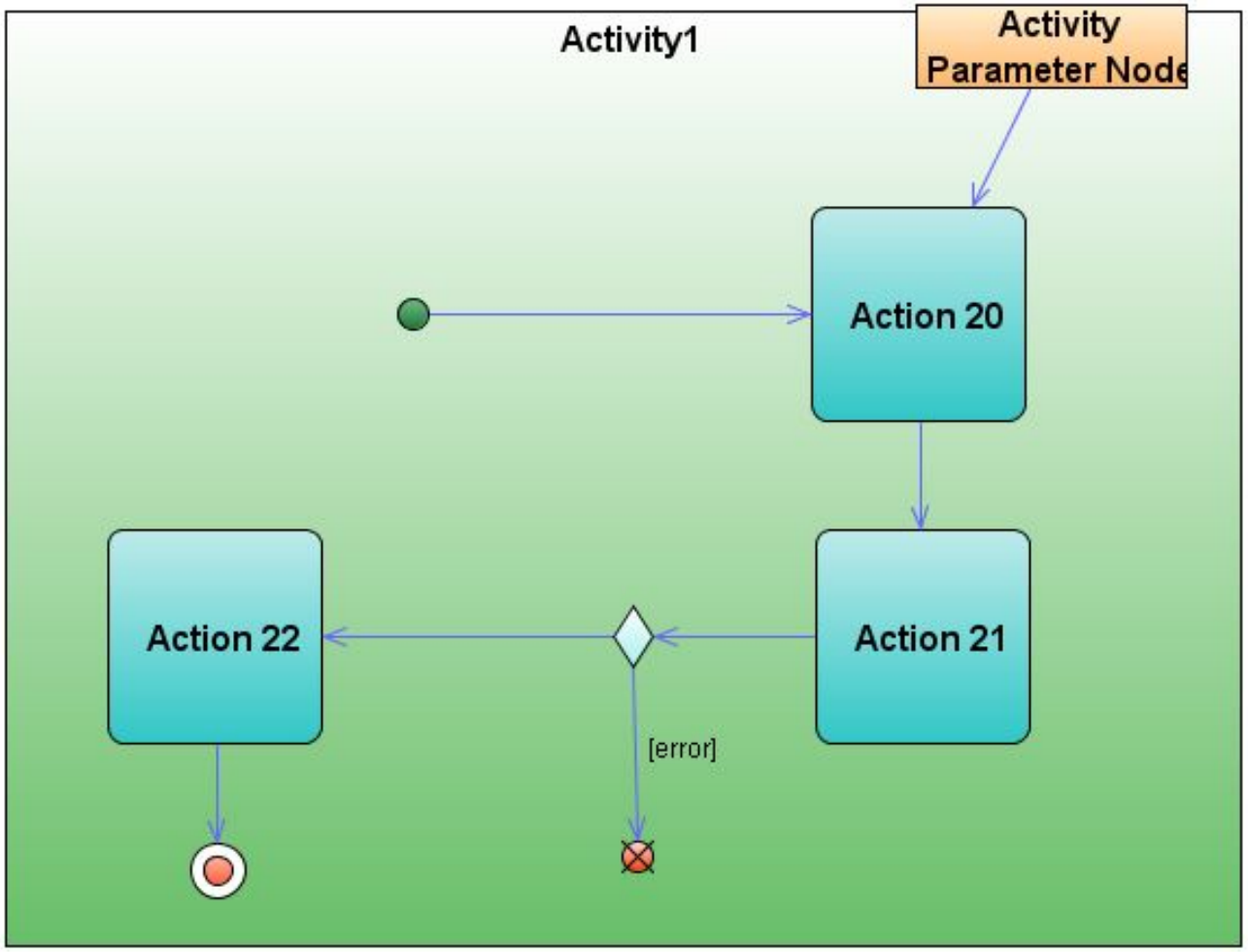


Диаграмма последовательности

Диаграмма последовательности — это способ описать поведение системы "на примерах". Фактически, диаграмма последовательности — это запись протокола конкретного сеанса работы системы (или фрагмента такого протокола). В объектно-ориентированном программировании самым существенным во время выполнения является посылка сообщений взаимодействующими объектами. Именно последовательность посылки сообщений отображается на данной диаграмме, отсюда и название.

На диаграмме последовательности применяют один основной тип сущностей — объекты (экземпляры взаимодействующих классов и действующих лиц), и один тип отношений — сообщения, которыми обмениваются взаимодействующие объекты. Предусмотрено несколько типов сообщений, которые в графической нотации различаются видом стрелки, соответствующей отношению. Важным аспектом диаграммы последовательности является явное отображение течения времени. В отличии от всех других типов диаграмм, на диаграмме последовательности имеет значение не только наличие графических связей между элементами, но и взаимное положение элементов на диаграмме.

Для обозначения самих взаимодействующих объектов применяется стандартная нотация — прямоугольник с подчеркнутым именем объекта. Пунктирная линия, выходящая из объекта, называется *линией жизни. Это* не обозначение отношение в модели, а графический комментарий, призванный направить взгляд читателя диаграммы в правильном направлении. Фигуры в виде узких полосок, наложенных на линию жизни, также не являются изображениями моделируемых сущностей. Это графический комментарий, показывающий отрезки времени, в течении которых объект имеет управление. Создание объекта в процессе взаимодействия отмечается тем, что значок объекта расположен ниже (т. е. объект появляется позже). Уничтожение объекта отмечает большим косым крестом и прекращением линии жизни.

Нотация диаграммы последовательности (UML 2.0)

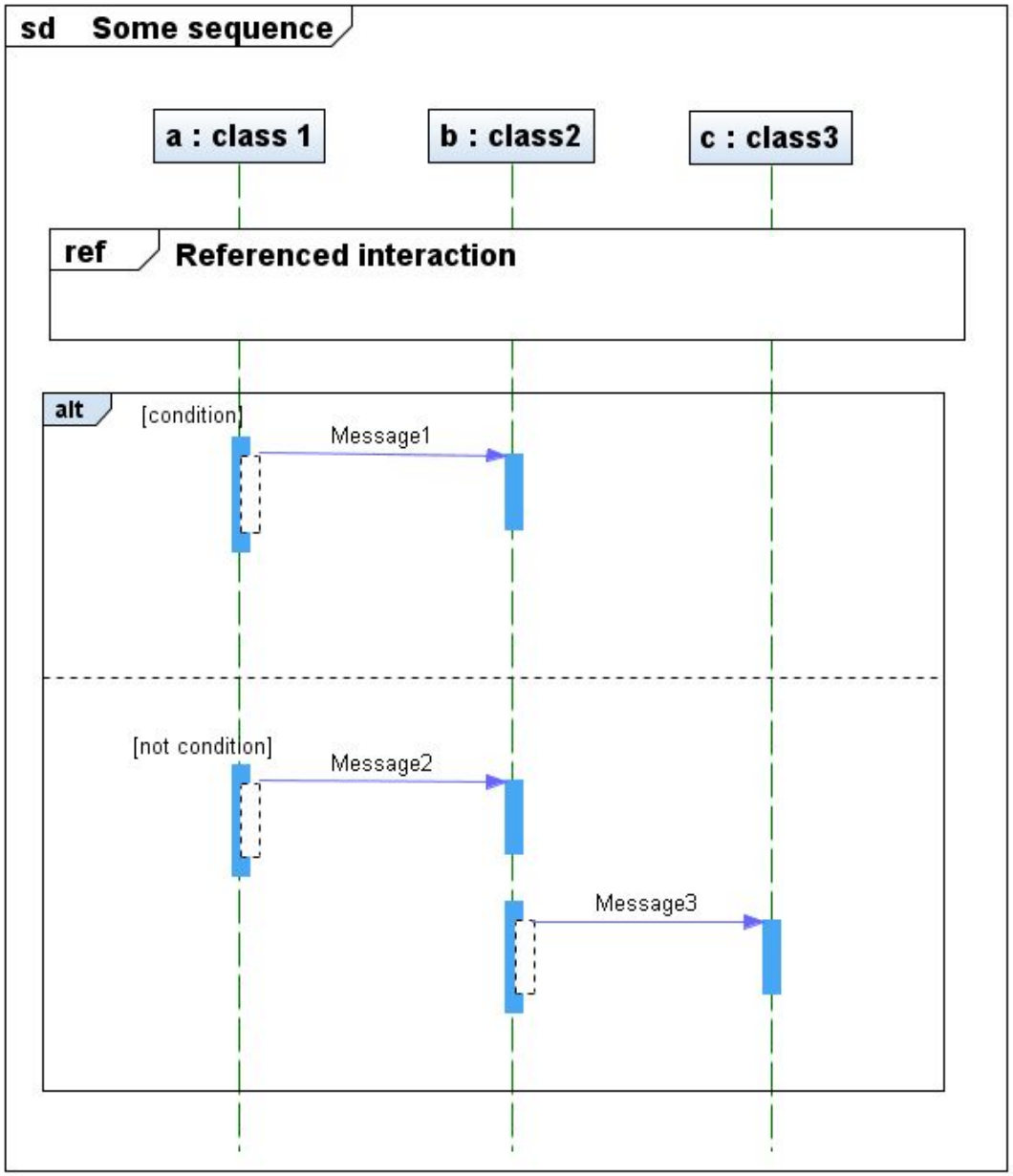


Диаграмма кооперации

Диаграмма кооперации, которая в UML 2 переименована в диаграмму коммуникации, семантически эквивалентна диаграмме последовательности.

На диаграмме кооперации также применяют один основной тип сущностей — объекты (экземпляры взаимодействующих классов и действующих лиц), и один тип отношений — сообщения, которыми обмениваются взаимодействующие объекты. Однако здесь акцент делается не на времени, а на связях между конкретными объектами.

Для обозначения самих взаимодействующих объектов применяется стандартная нотация — прямоугольник с подчеркнутым именем объекта. Взаимное положение объектов на диаграмме кооперации не имеет значения — важны только связи (экземпляры ассоциаций), вдоль которых передаются сообщения. Для отображения упорядоченности сообщений во времени применяется иерархическая десятичная нумерация.

Нотация диаграммы кооперации

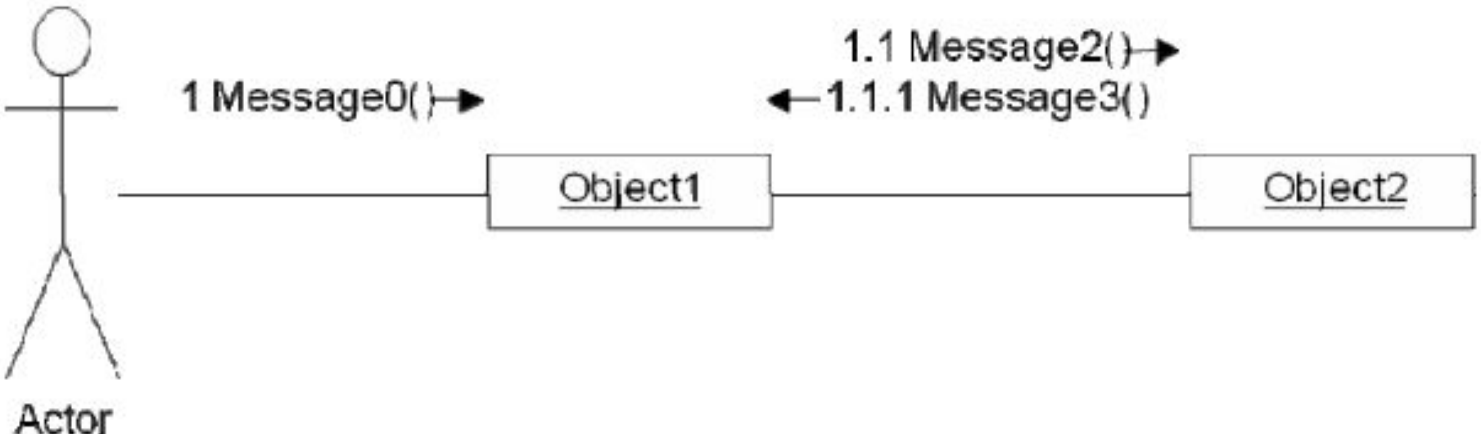


Диаграмма компонентов

Диаграмма компонентов — это, фактически, список артефактов, из которых состоит моделируемая система, с указанием некоторых отношений между артефактами. Наиболее существенным типом артефактов программных систем являются программы. Таким образом, на диаграмме компонентов основной тип сущностей — это компоненты (как исполнимые модули, так и другие артефакты), а также интерфейсы (чтобы указывать взаимосвязь между компонентами) и объекты (входящие в состав компонентов). На диаграмме компонентов применяются следующие отношения:

- реализации между компонентами и интерфейсами (компонент реализует интерфейс);
- зависимости между компонентами и интерфейсами (компонент использует интерфейс);
- зависимости между объектами и компонентами (объект входит в компонент).

Отношение зависимости, соответствующее включению (например, объекта в компонент), часто изображают, помещая фигуру одной сущности внутрь фигуры другой сущности.

Нотация диаграммы компонентов (UML 2.0)

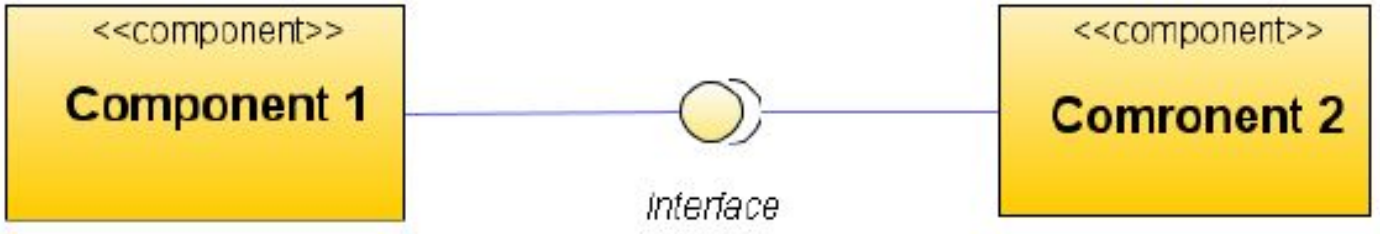
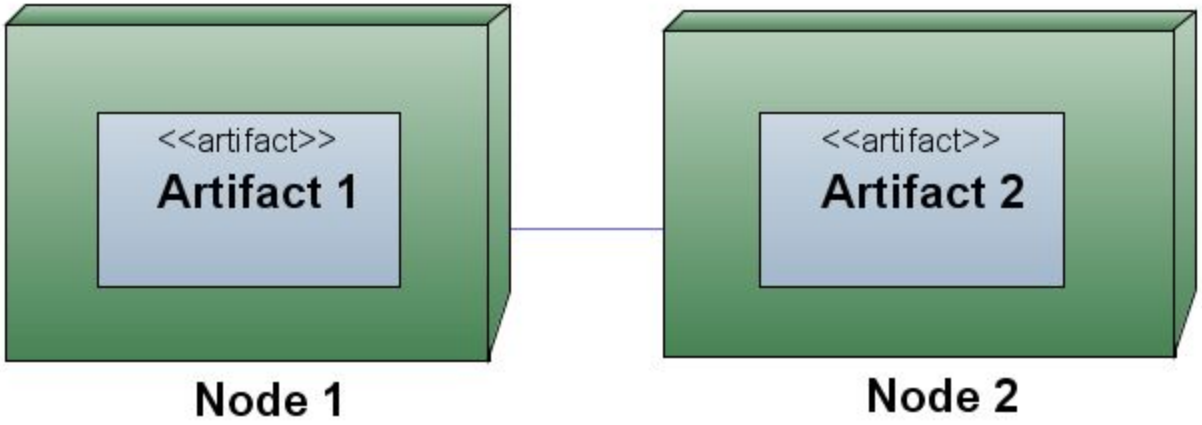


Диаграмма размещения

Диаграмма размещения немногим отличается от диаграммы компонентов. Фактически, показывается, как физически размещены компоненты на вычислительных ресурсах во время выполнения. Таким образом, на диаграмме размещения, по сравнению с диаграммой компонентов, добавляется один тип сущностей — узел (может быть как классификатор, описывающий тип узла, так и конкретный экземпляр), а также отношение ассоциации между узлами, показывающее, что узлы физически связаны во время выполнения.

Нотация диаграммы размещения



Спасибо за внимание!