Сегодня на лекции

- Цель
  - Ознакомиться с инструментарием языка UML
- План
  - Что такое UML?
  - Нотация UML
  - Модель и ее элементы
    - Типы сущностей
    - Типы отношений
    - Типы диаграмм
  - Инструментальные средства

Язык UML

- **Unified**
  - Унификация – объединение и обобщение методик моделирования
- **Modelling**
  - Моделирование — исследование объектов познания на их моделях; построение и изучение моделей объектов, процессов или явлений с целью получения объяснений этих явлений, а также для предсказания явлений, интересующих исследователя.
  - модель UML – это, прежде всего, описание объекта или явления
- **Language**
  - Язык – это знаковая система для хранения и передачи информации.
    - UML - формальный искусственный графический язык

http://book.uml3.ru

UML. Предпосылки возникновения

- **Потребность получить единый язык моделирования, который объединил бы в себе всю мощь объектно-ориентированного подхода и давал бы четкую модель системы, отражающую все ее значимые стороны.**
- Предшественники UML:
- Booch (Grady Booch) - подходил для стадий дизайна и разработки
- OMT-2 (Jim Rumbaugh) - на стадиях анализа и разработки информационных систем
- OOSE - Object-Oriented Software Engineering (Ivar Jacobson) - на стадии анализа проблемной области.
- **UML - стандарт де-факто языков моделирования**
- **UML поддерживают практически все известные разработчики средств объектного моделирования**

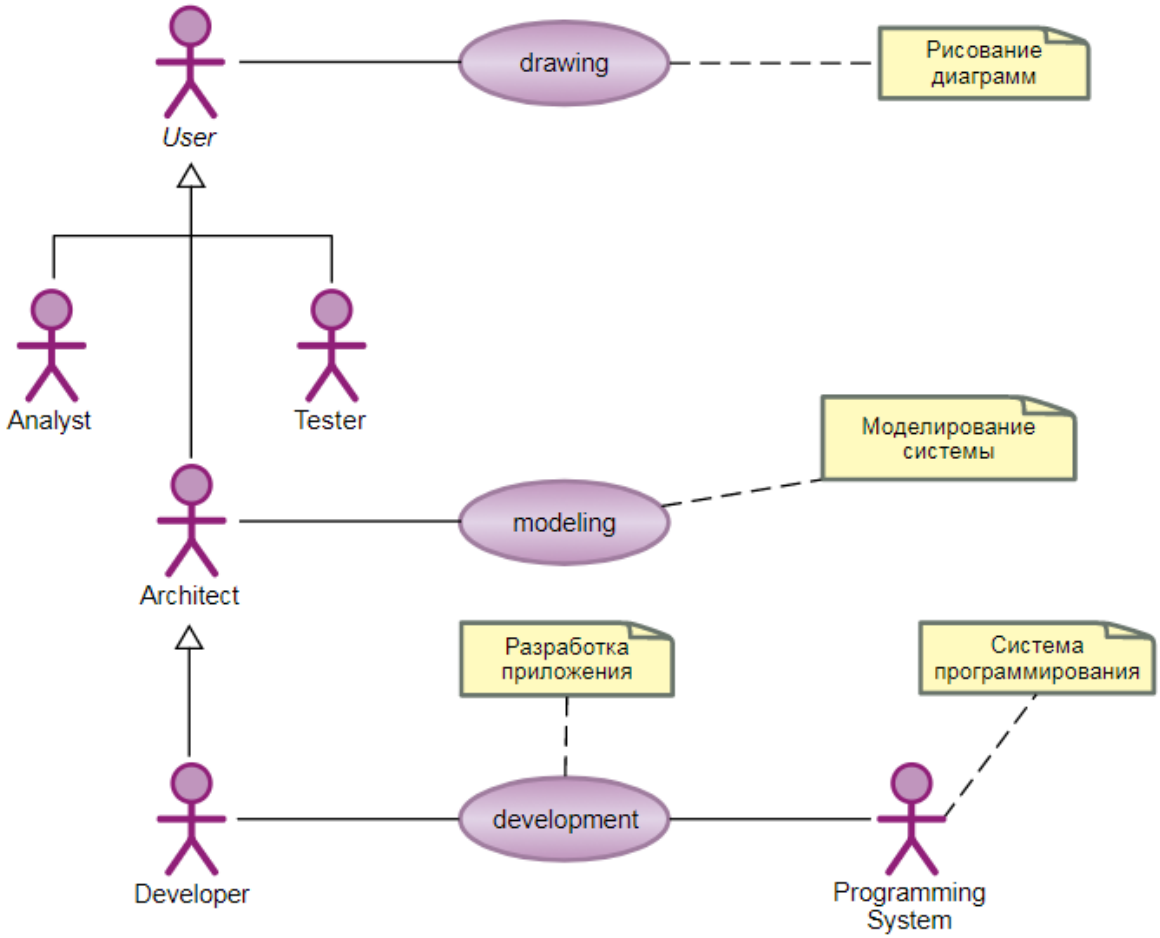
UML. Определение

- UML – это графический язык моделирования общего назначения, предназначенный для специфицирования, визуализации, конструирования и документирования программных систем, а так же бизнес моделей и прочих не программных систем.
  - Спецификация – это декларативное описание того, как нечто устроено или работает.
  - Визуализация (графическое представление != модели!)
  - Конструирование как проектирование + генерация кода по модели
  - Документирование – диаграммы как артефакты разработки
- Чем UML не является:
  - Языком программирования
  - Спецификацией инструмента
  - Моделью процесса разработки

Способы использования UML

- **Рисование картинок**.
  - Даже простая визуализация позволяет упорядочить мысли и зафиксировать для себя существенную информацию о моделируемом приложении или иной системе.
- **Обмен информацией**.
  - Общепринятая нотация позволяет находить общий язык
- **Спецификация систем**.
  - Важнейший способ использования
- **Повторное использование архитектурных решений**.
  - Документирование шаблонных решений
- **Генерация кода**.
  - Базовые возможности (обычно, лишь структурные модели)
- **Имитационное моделирование**.
  - Пока что слабо развито
- **Верификация моделей**.
  - Пока что слабо развита

Способы использования UML



Нотация UML

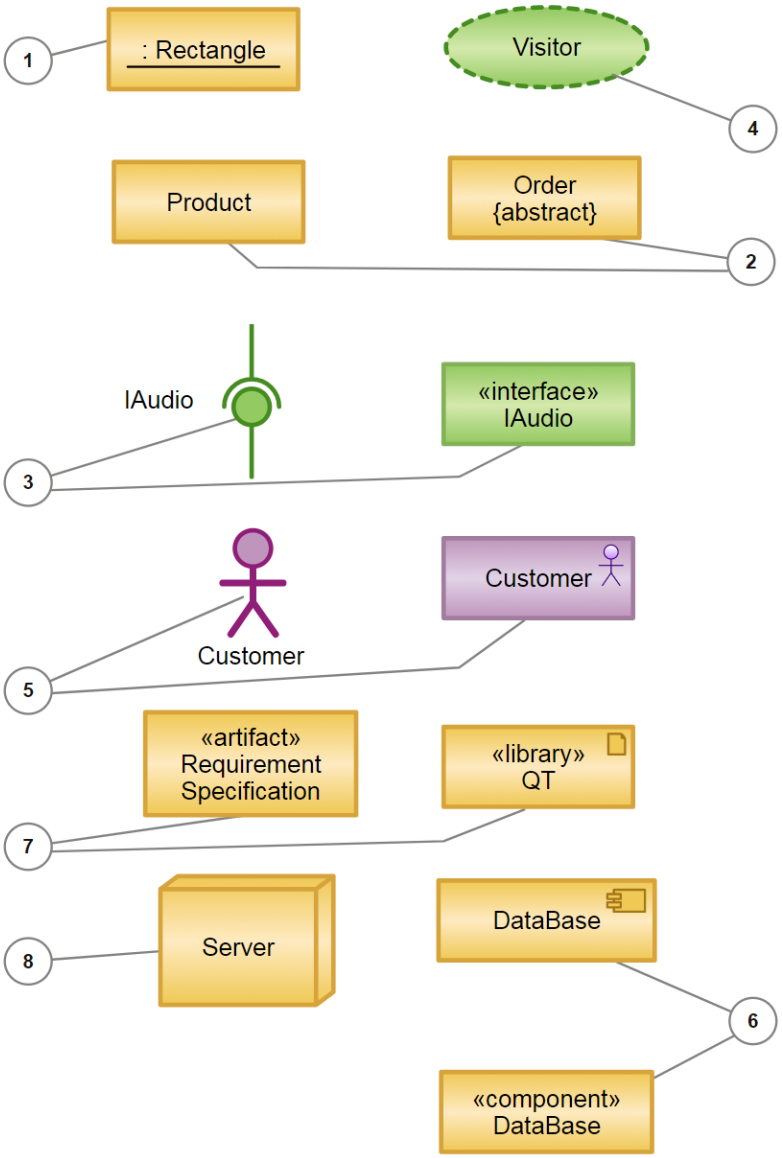
- Фигура
  - Двумерные замкнутые
  - Соотношения и размеры произвольны, сохраняя различимость фигур
  - Могут содержать внутри другие элементы
- Линия
  - Толщина и форма произвольны
  - Стили фиксированы (сплошная и пунктирная)
- Значок
  - Частный случай фигуры
  - Не может содержать внутренних элементов
- Текст
  - Гарнитура и размер произвольны
  - Стили фиксированы (обычный, *курсив* и <u>подчеркнутый</u>)
- Рамка
  - Частный случай фигуры
  - Не может НЕ содержать внутренних элементов
  - Содержит ярлычок с текстом

Модель UML и ее элементы.

- Модель UML – это совокупность конечного множества конструкций языка, главные из которых – это **сущности** и **отношения** между ними.
- Иными словами, модель UML, можно считать своеобразным графом, в котором вершины и ребра нагружены дополнительной информацией и могут иметь сложную внутреннюю структуру. **Вершины этого графа называются сущностями, а ребра – отношениями**.

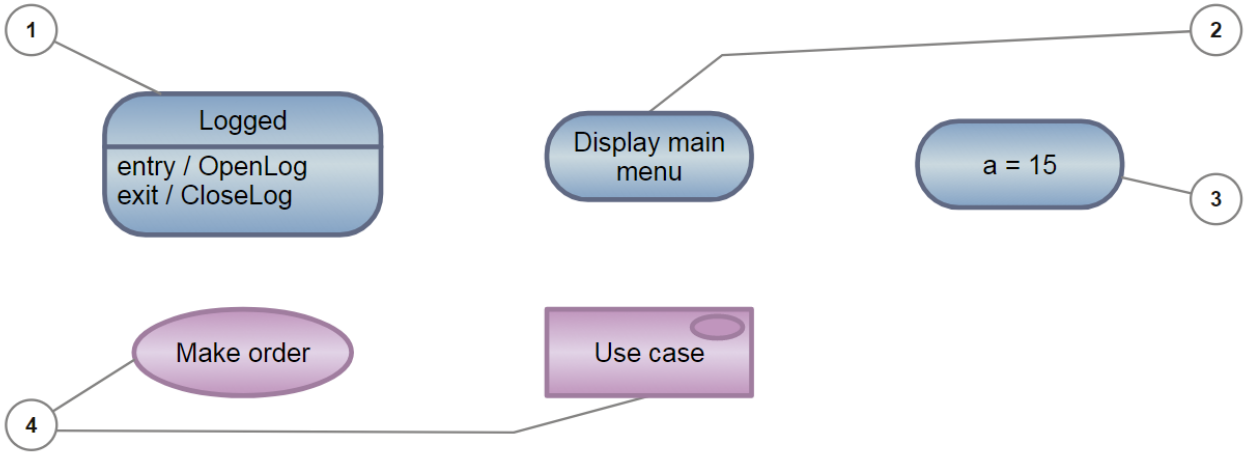
- Сущности UML:
  - структурные;
  - поведенческие;
  - группирующие;
  - аннотационные.
- Отношения UML:
  - зависимость;
  - ассоциация;
  - обобщение;
  - реализация.

Структурные сущности UML



- **Объект** (1) – сущность, обладающая уникальностью и инкапсулирующая в себе состояние и поведение.
- **Класс** (2) – описание множества объектов с общими атрибутами, определяющими состояние, и операциями, определяющими поведение.
- **Интерфейс** (3) – именованное множество операций, определяющее набор услуг, которые могут быть запрошены потребителем и предоставлены поставщиком услуг.
- **Кооперация** (4) – совокупность объектов, которые взаимодействуют для достижения некоторой цели.
- **Действующее лицо** (5) – сущность, находящаяся вне моделируемой системы и непосредственно взаимодействующая с ней.
- **Компонент** (6) – модульная часть системы с четко определенным набором требуемых и предоставляемых интерфейсов.
- **Артефакт** (7) – 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** (8) – вычислительный ресурс, на котором размещаются и при необходимости выполняются артефакты.

Поведенческие сущности UML

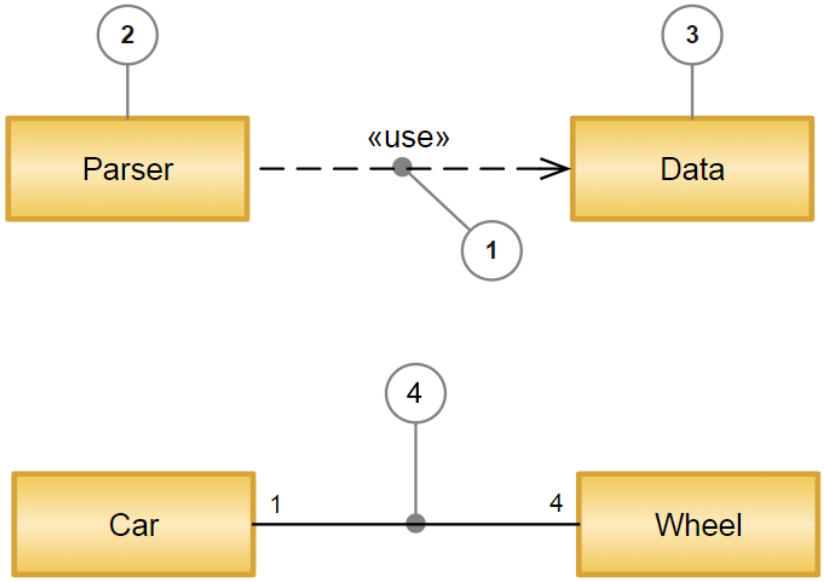


- **Состояние** (1) – период в жизненном цикле объекта, находясь в котором объект удовлетворяет некоторому условию и осуществляет собственную деятельность или ожидает наступления некоторого события.
- **Деятельность** (2) можно считать частным случаем состояния, который характеризуется продолжительными (по времени) не атомарными вычислениями.
- **Действие** (3) – примитивное атомарное вычисление.
- **Вариант использования** (4) – множество сценариев, объединенных по некоторому критерию и описывающих последовательности производимых системой действий, доставляющих значимый для некоторого действующего лица результат.

Прочие сущности UML

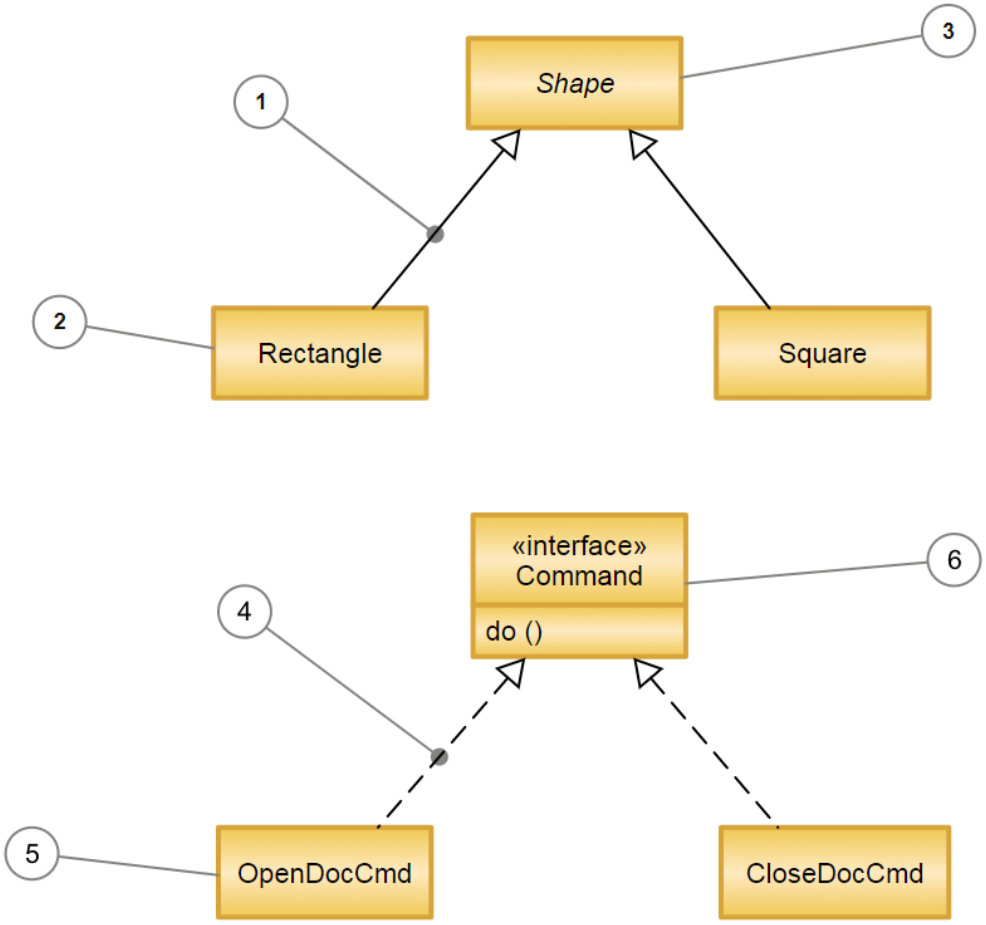
- Группирующая сущность: **Пакет** (1) – группа элементов модели (в том числе пакетов).
- Аннотационная сущность: **Комментарий** (2) – произвольное по формату и содержанию описание одного или нескольких элементов модели.

Отношения UML



- **Отношение зависимости** (1) указывает на то, что изменение независимой сущности (3) каким-то образом влияет на зависимую сущность (2).
- **Отношение ассоциации** (4) имеет место, если одна сущность непосредственно связана с другой (или с другими – ассоциация может быть не только бинарной).

Отношения UML



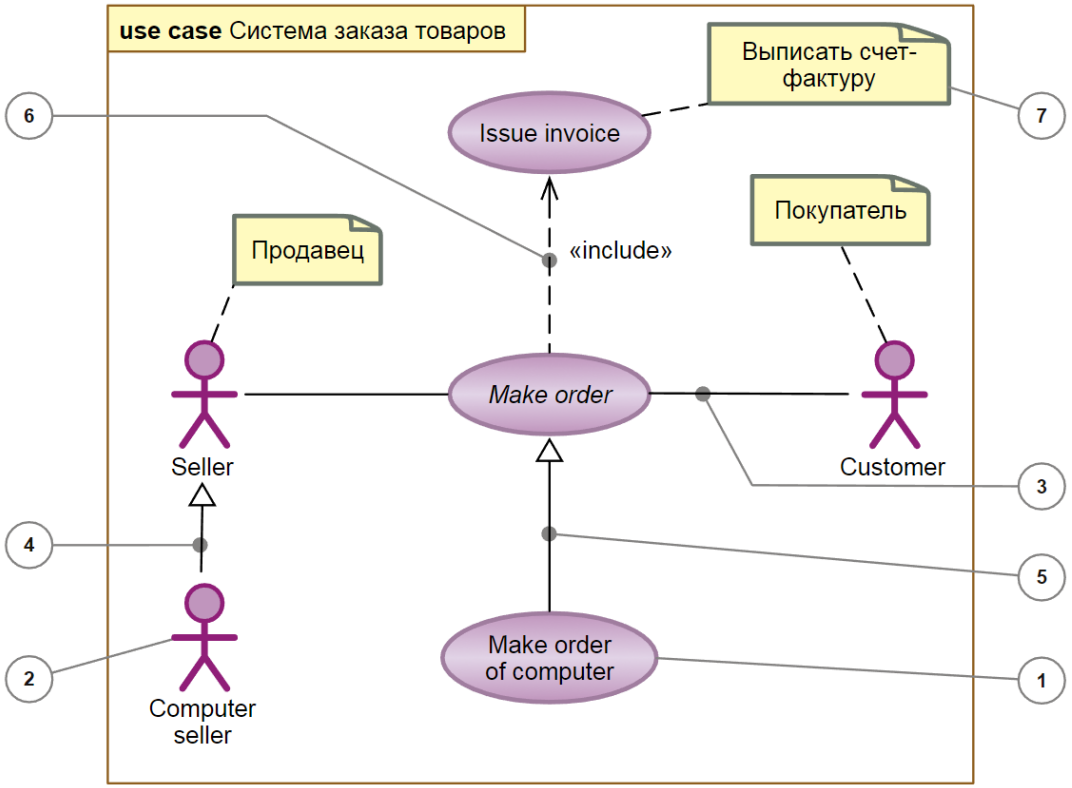
- **Обобщение** (1) – это отношение между двумя сущностями, одна их которых (2) является частным случаем («подкласс») другой (3) («суперкласс»).
- **Отношение реализации** (4) указывает, что одна сущность (5) является реализацией другой (6).
  - Используется, чтобы показать, что, класс является реализацией интерфейса.

Диаграммы UML

Диаграммы UML задают правила использования элементов нотации для построения модели определенного аспекта структуры или поведения системы:

- Диаграммы для изображения структуры системы:
  - Диаграмма компонентов (component diagram, тег component);
  - Диаграмма размещения (deployment diagram, тег deployment);
  - Диаграмма классов (class diagram, тег class);
  - Диаграмма объектов (object diagram, тег object);
  - Диаграмма внутренней структуры (composite structure diagram, тег class);
- Диаграммы для изображения поведения системы:
  - Диаграмма синхронизации (interaction diagram, тег timing);
  - Диаграмма деятельности (activity diagram, тег activity);
  - Диаграмма последовательности (sequence diagram, тег sd);
  - Диаграмма коммуникации (communication diagram, тег comm);
  - Диаграмма автомата (state machine diagram, тег state machine);
  - Обзорная диаграмма взаимодействия (interaction overview diagram, тег interaction);
- Диаграмма использования(use case diagram, тег use case);
- Диаграмма пакетов (package diagram, тег package);

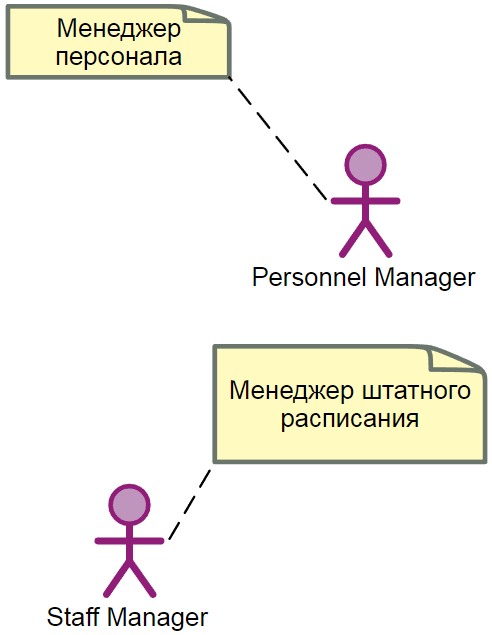
Диаграмма вариантов использования



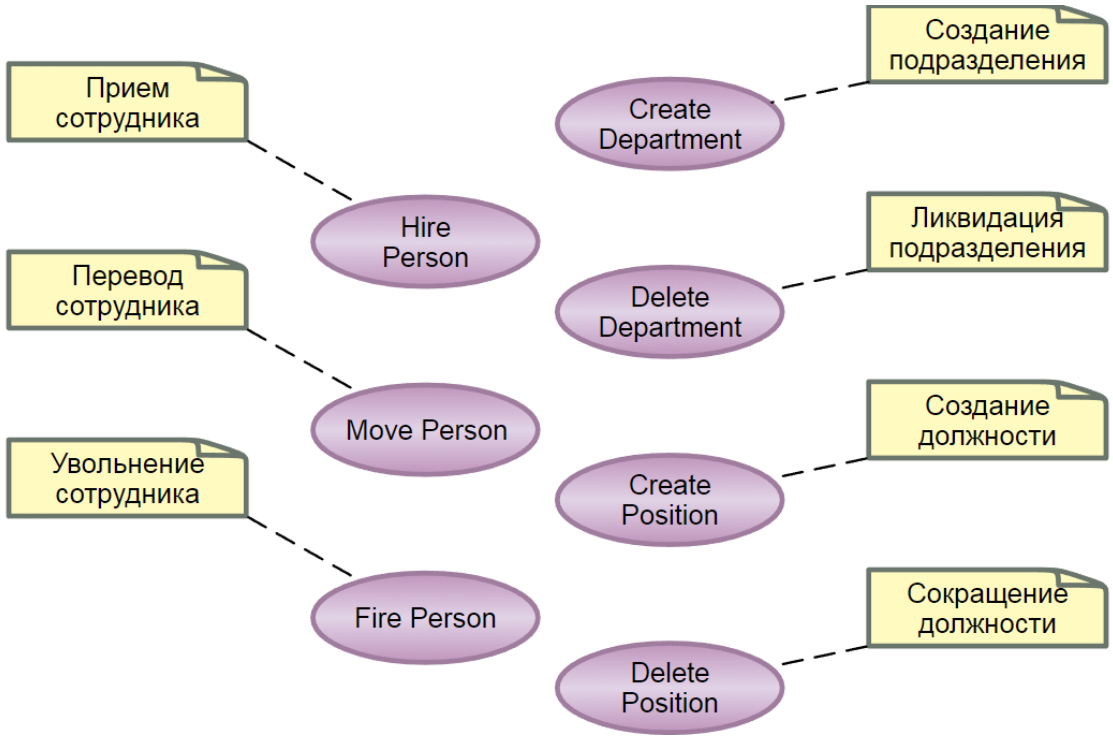
- **Диаграмма использования** (use case diagram) – это наиболее общее представление функционального назначения системы.
  - призвана ответить на главный вопрос моделирования: что делает система во внешнем мире?
- На диаграмме использования применяются два типа основных сущностей: варианты использования (1) и действующие лица (2), между которыми устанавливаются следующие основные типы отношений:
  - ассоциация между действующим лицом и вариантом использования (3);
  - обобщение между действующими лицами (4);
  - обобщение между вариантами использования (5);
  - зависимости (различных типов) между вариантами использования (6).
- На диаграмме использования, как и на любой другой, могут присутствовать комментарии (7).

Элементы диаграммы вариантов использования

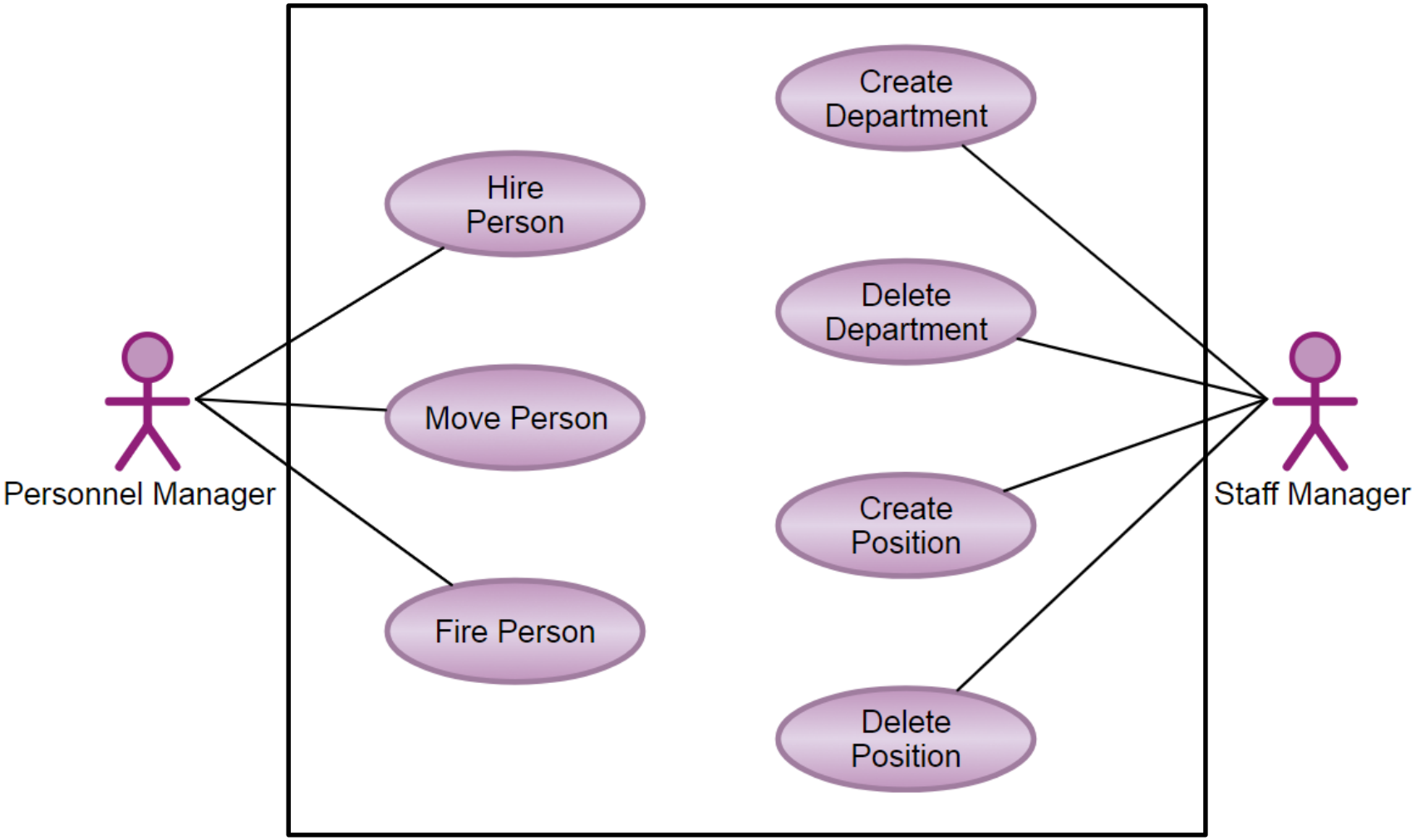
Действующие лица



Варианты использования



Ассоциации на диаграмме вариантов использования



Обобщения на диаграмме вариантов использования

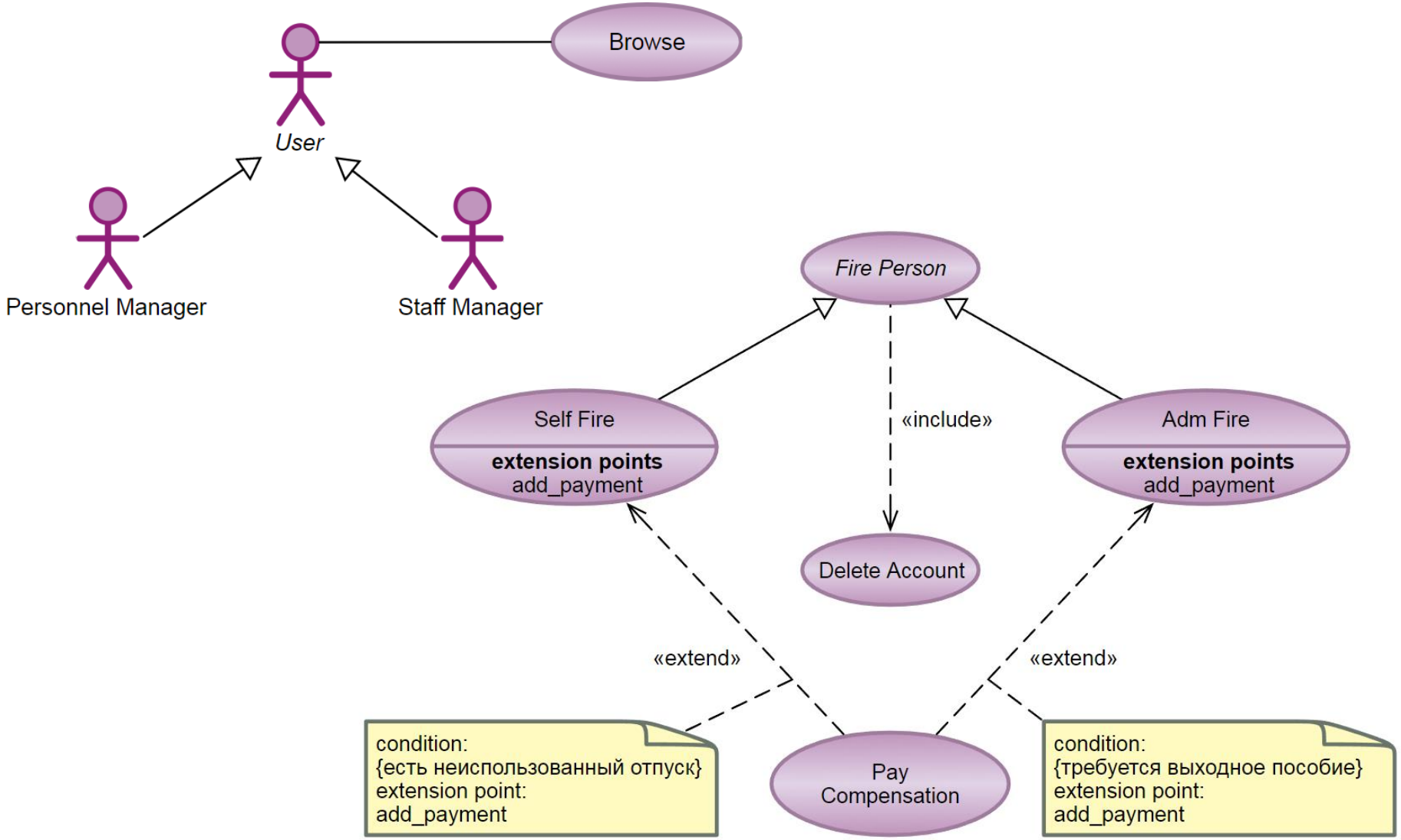
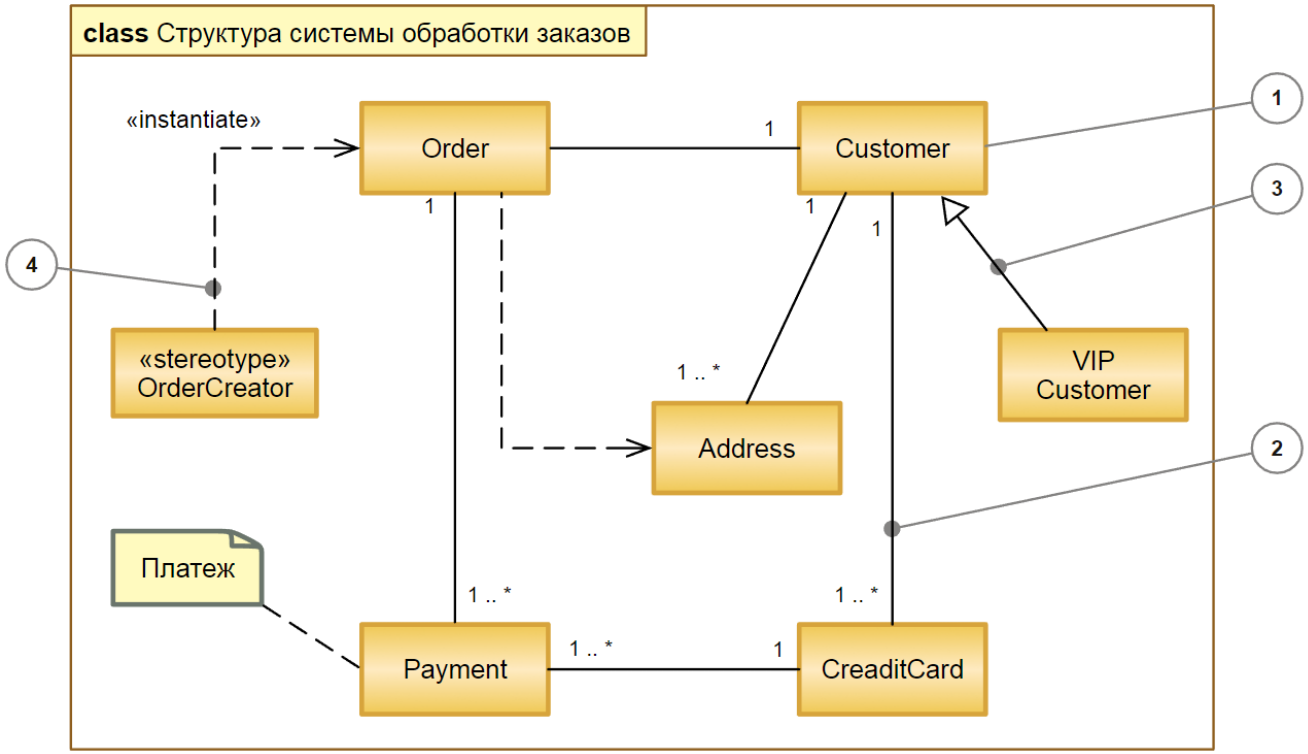
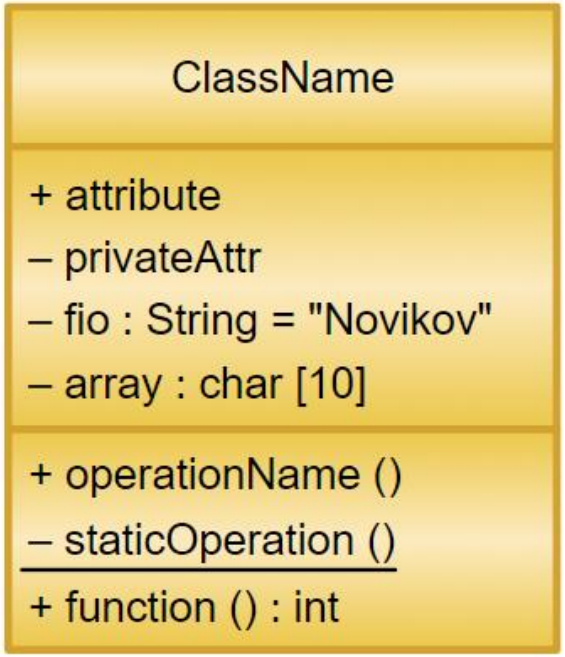


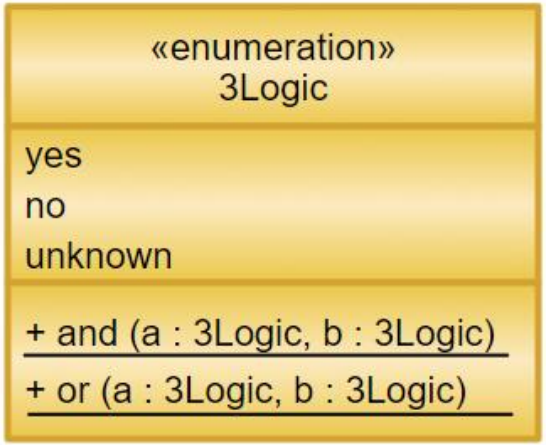
Диаграмма классов



- **Диаграмма классов** – основной способ описания структуры системы. Содержит:
  - классы и интерфейсы (1)
  - отношения ассоциации между классами (2) (с множеством дополнительных подробностей);
  - отношения обобщения между классами (3);
  - отношения зависимости (различных типов) между классами (4) и между классами и интерфейсами.

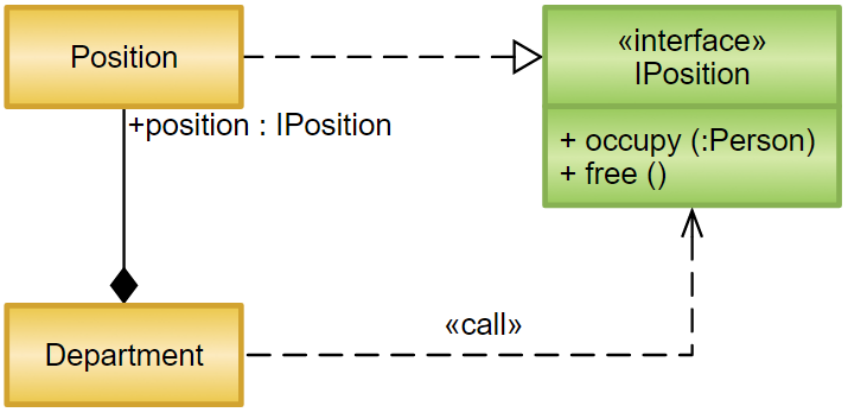
Нотация класса

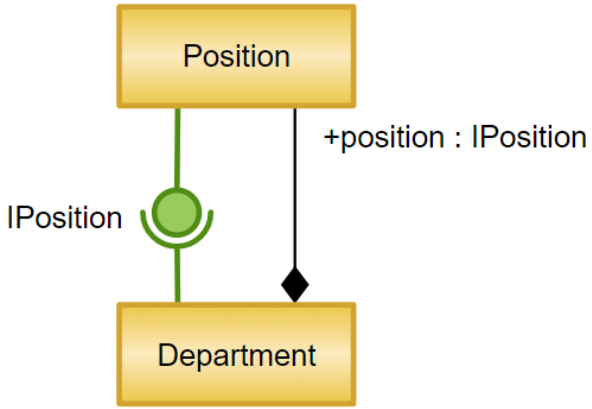




- Имя (обязательно)
  - «стереотип»
  - {ограничения/свойства}
  - *Абстрактность*
  - Объект
- Атрибуты
  - Видимость
    - + public
    - - private
    - # protected
    - ~ package
  - Имя
  - : тип
  - = начальное значение
  - {свойства}
- Методы
  - Видимость
  - Имя
  - (Параметры)
  - : тип результата

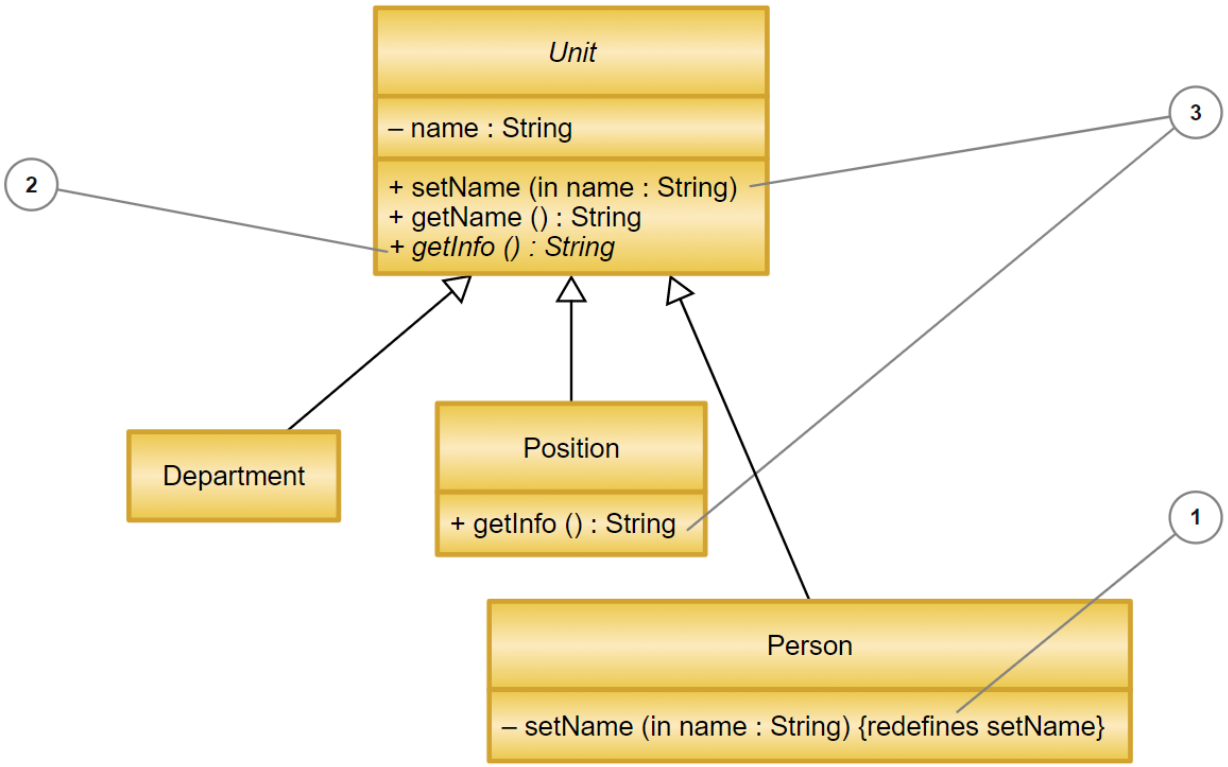
Отношения зависимости и реализации





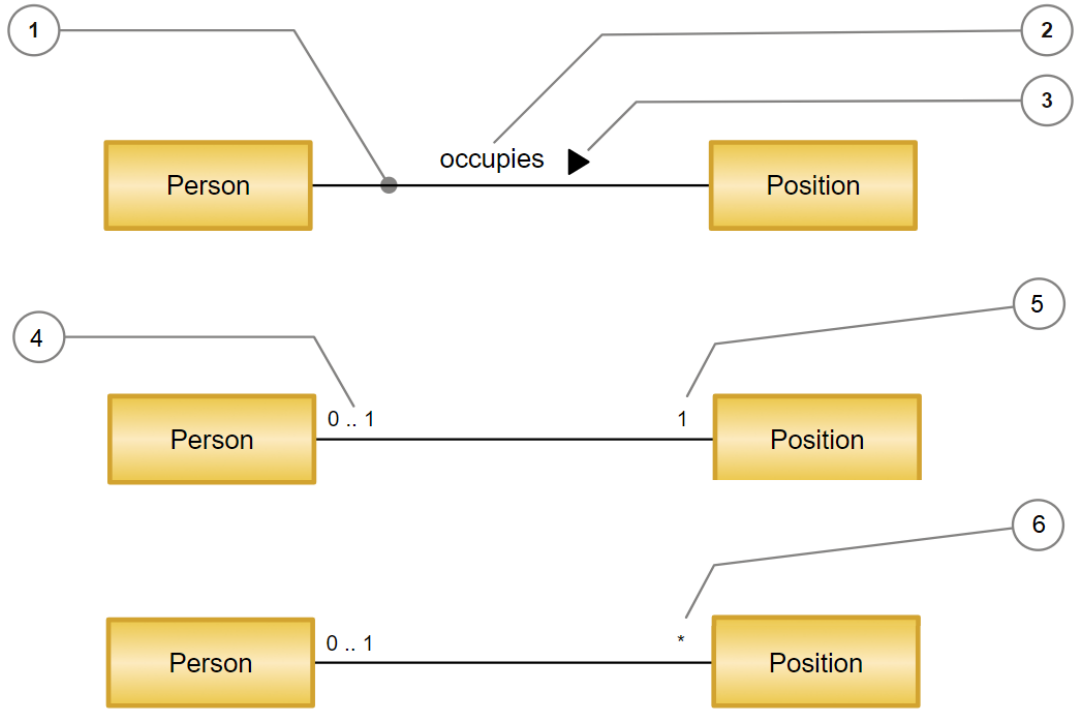
- Нотация UML 1
  - Отношение реализации показывает, что класс реализует интерфейс
  - Отношение зависимости показывает, что класс использует интерфейс
- Нотация UML 2
  - «леденец» на реализующей стороне
  - «рот» на использующей стороне

Отношение обобщения



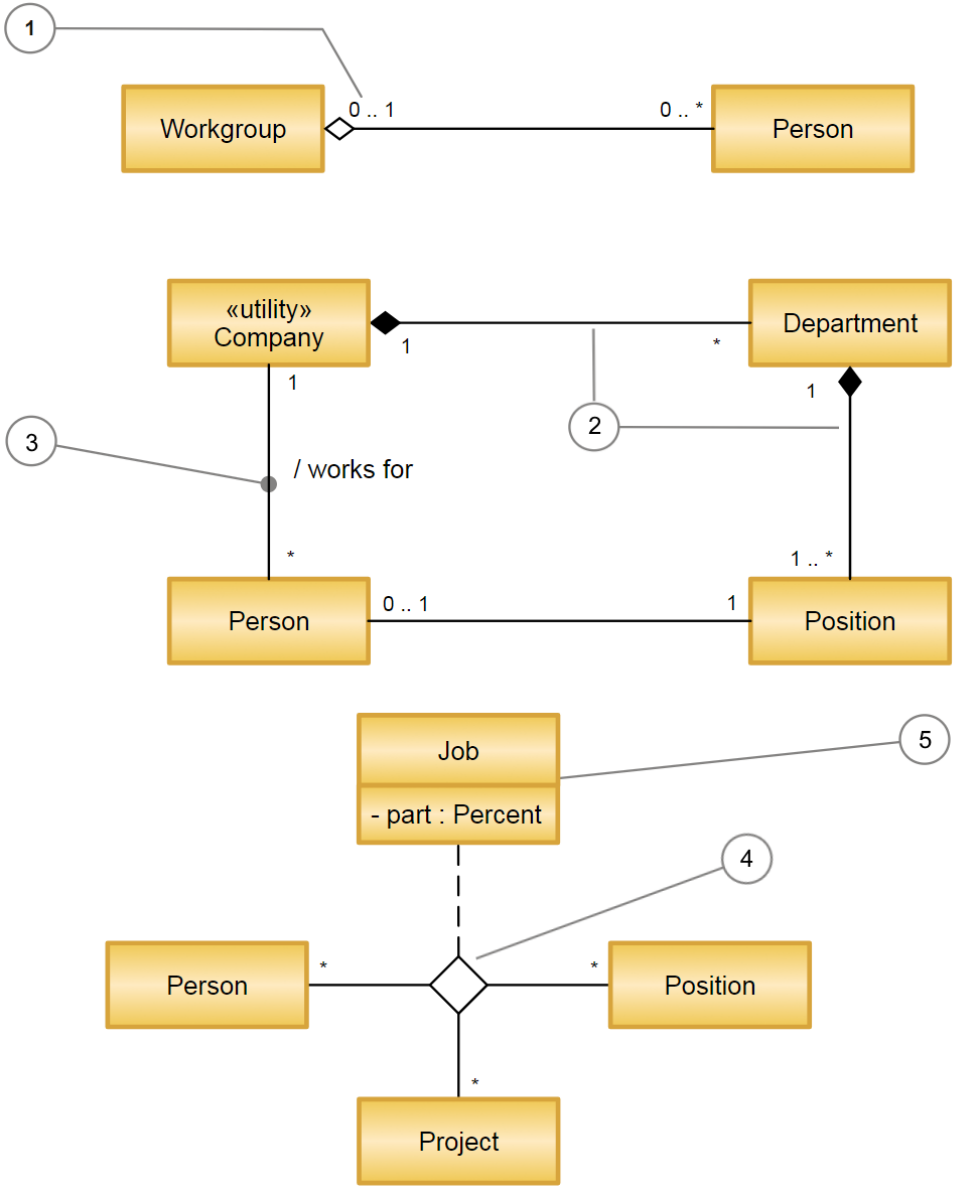
- Наследование
  - (1) Переопределение операции
  - (2) Абстрактная операция (чисто виртуальная)
  - (3) Операция с параметром

Отношение ассоциации



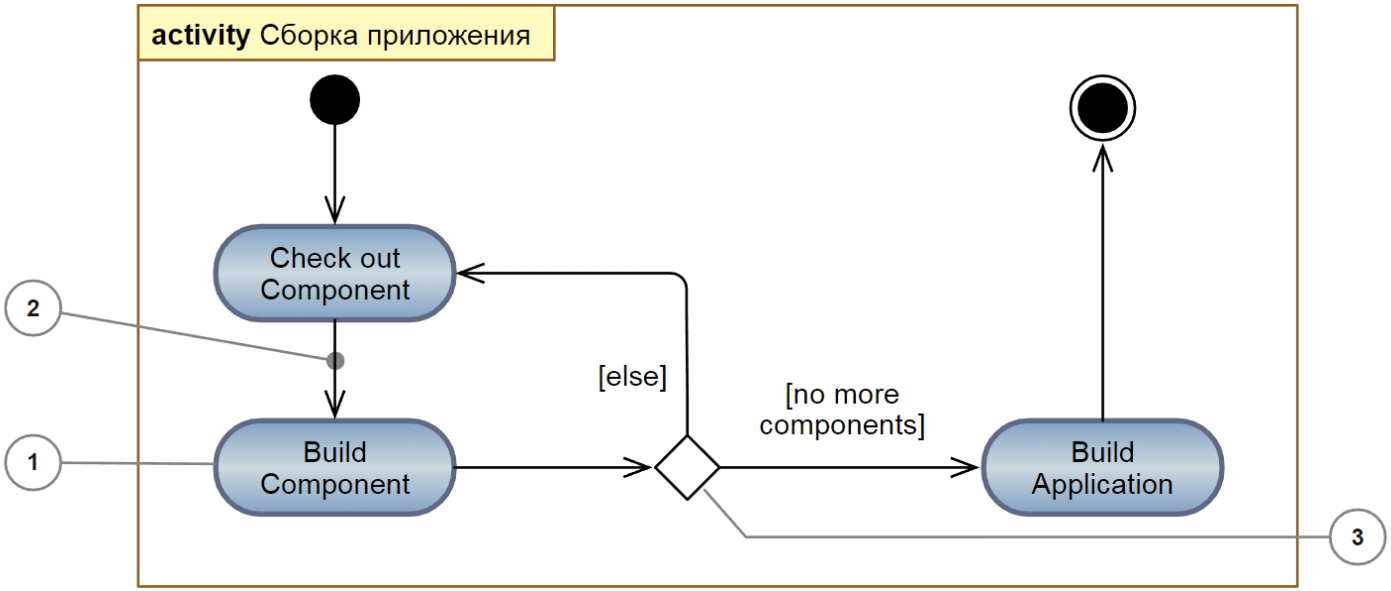
- Отношение ассоциации (1)
  - Имя связи (2)
  - Направление (3)
  - Кратность
    - Диапазон (4)
    - Точная (5)
    - Неопределенная (6)

Агрегация и композиция



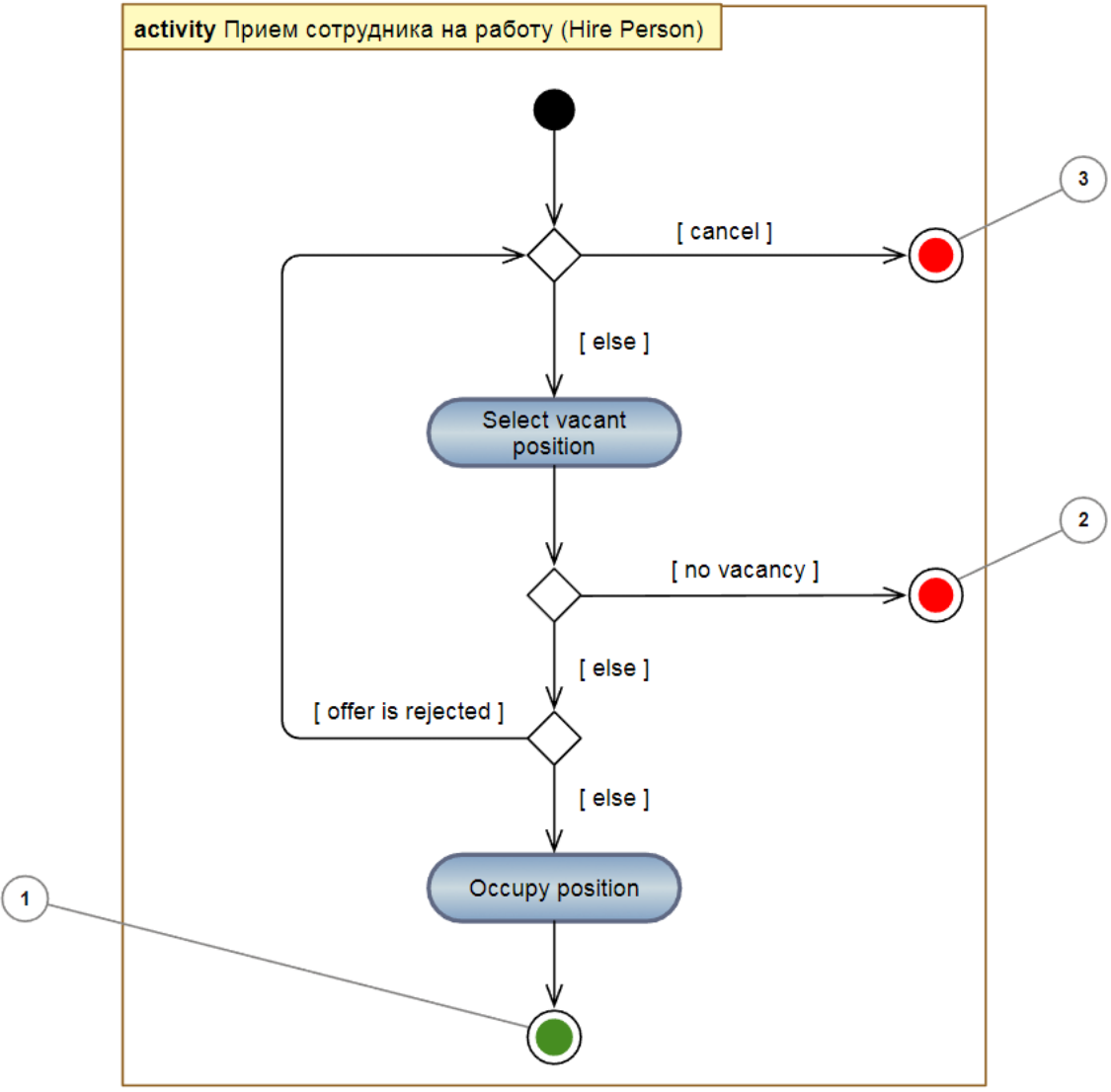
- **Агрегация** (1) – это ассоциация между классом A (часть) и классом B (целое), которая означает, что экземпляры (один или несколько) класса A входят в состав экземпляра класса B
- **Композиция** (2) – это ассоциация между классом A (часть) и классом B (целое), означающая, что часть A может входить только в одно целое B, часть существует, только пока существует целое и прекращает свое существование вместе с целым.
- **Производная ассоциация** (3) – ассоциация, которая может быть вычислена по другим элементам
- **Многополюсная ассоциация** (4) связывает более двух сущностей
- **Класс ассоциации** (5) – это сущность, которая является ассоциацией, но также имеет в своем составе составляющие класса.

Диаграмма деятельности



- **Диаграмма деятельности** (activity diagram) – способ описания поведения на основе указания потоков управления и потоков данных на основе сетей Петри. Элементы:
  - Сущность действие (1),
  - Отношение переход (2) – передача управления и данных.
  - Конструкции: развилки, слияния, соединения, ветвления (3)

Элементы диаграммы деятельности



- Старт
- Развилка
  - [условие]
- Действие
- Завершение
  - Нормальное (1)
  - Исключительная ситуация (2)
  - Завершение без видимого результата (3)

Диаграмма деятельности (activity diagram) – Усложненный пример

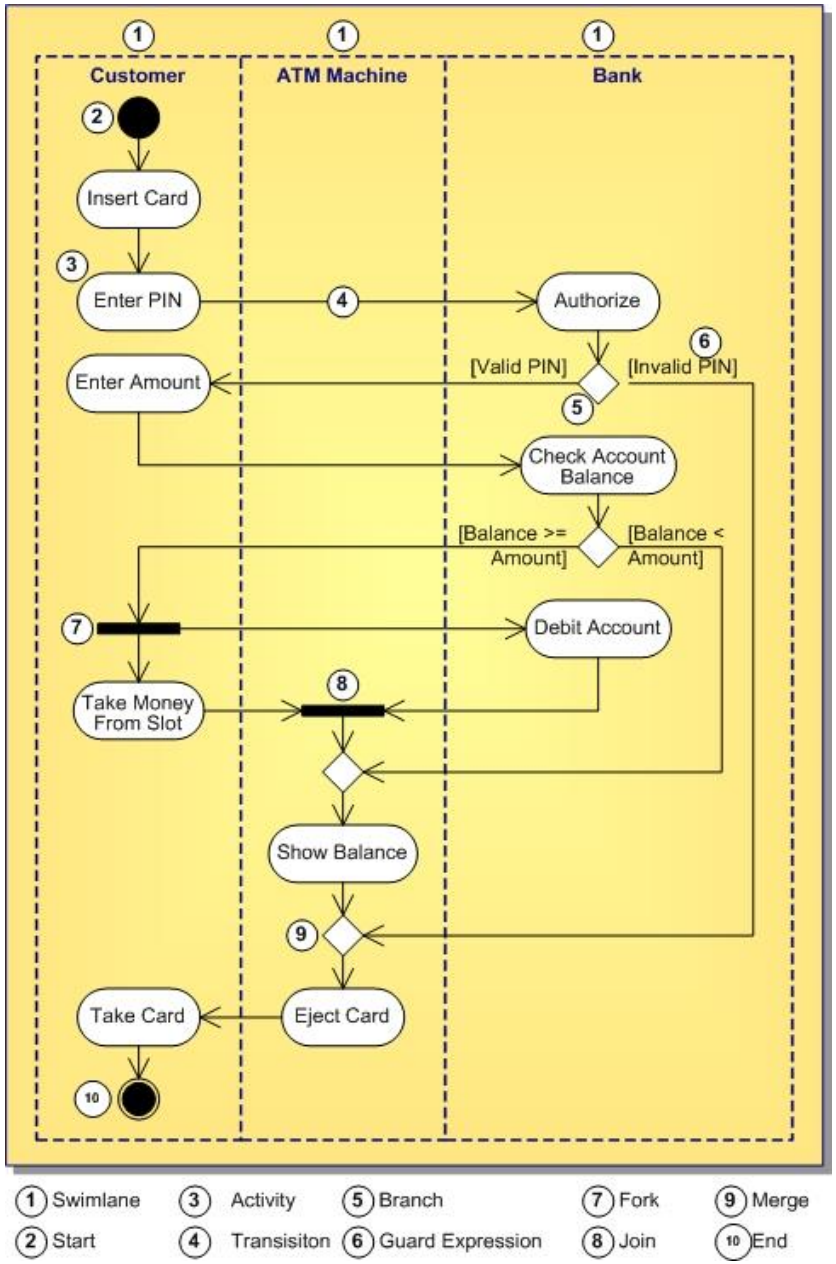


Диаграмма деятельности (activity diagram) – события и синхронизация

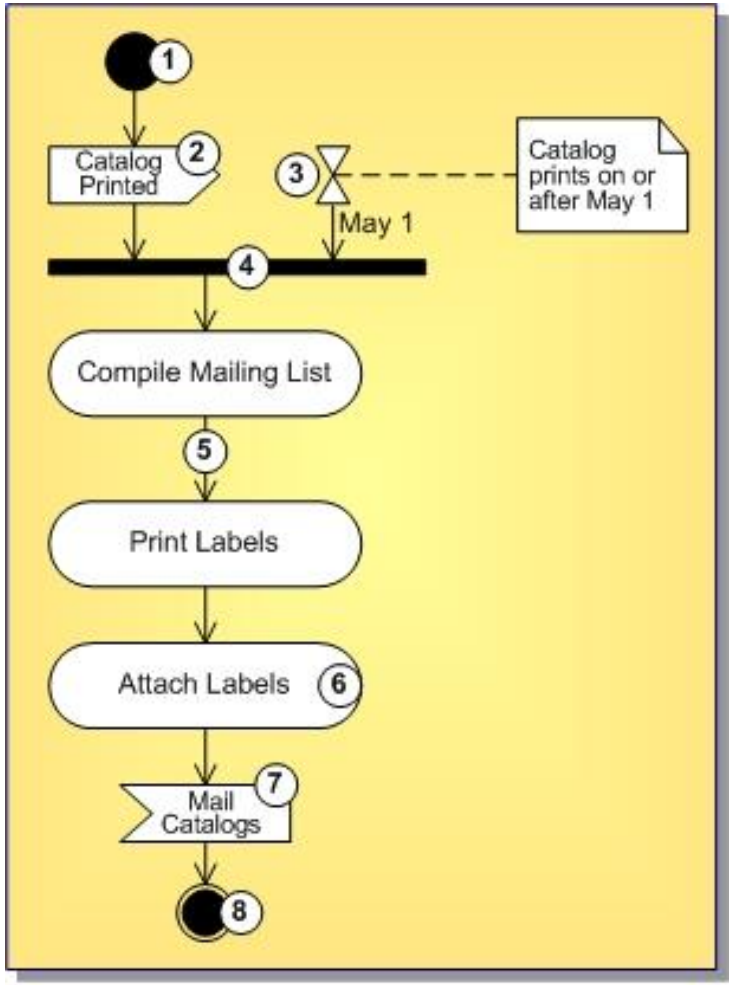


Диаграмма деятельности (activity diagram) - потоки данных

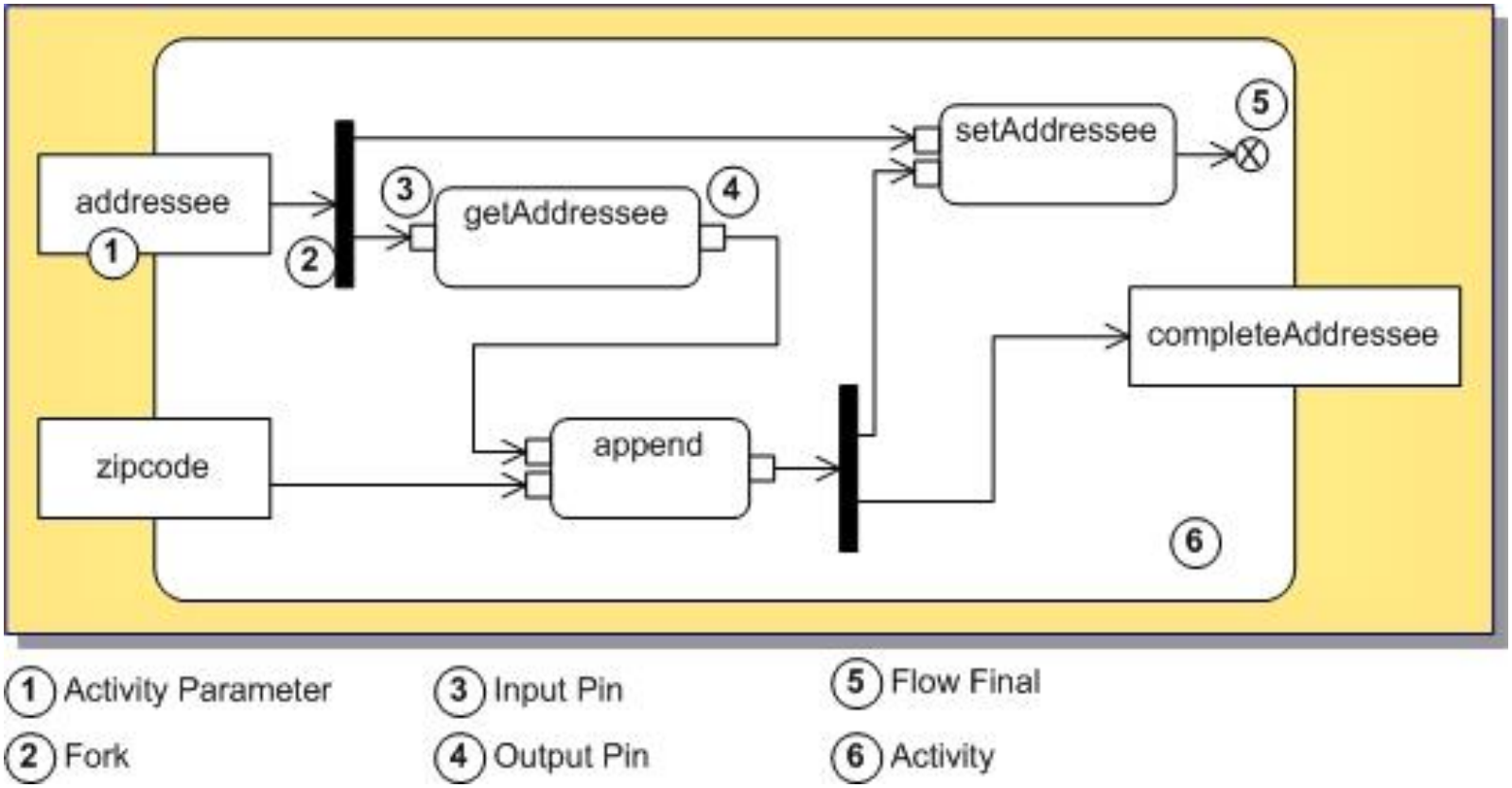
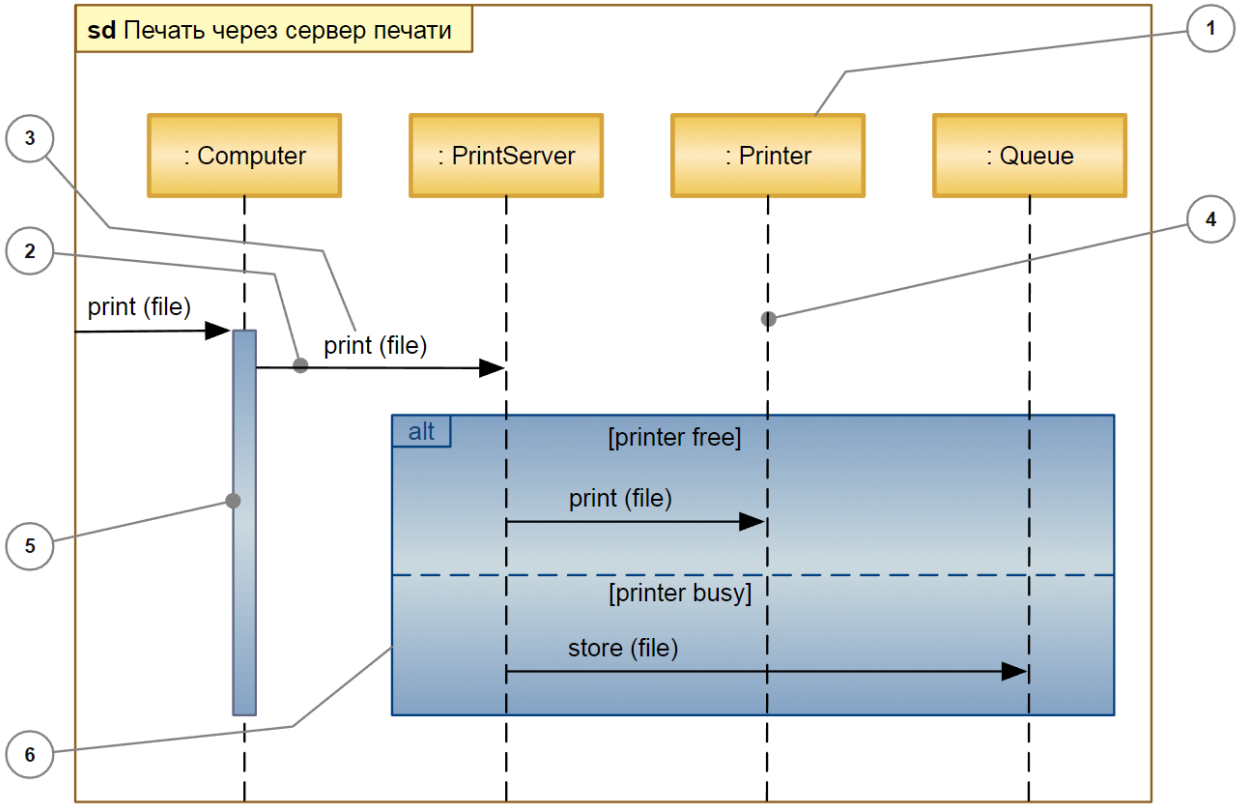
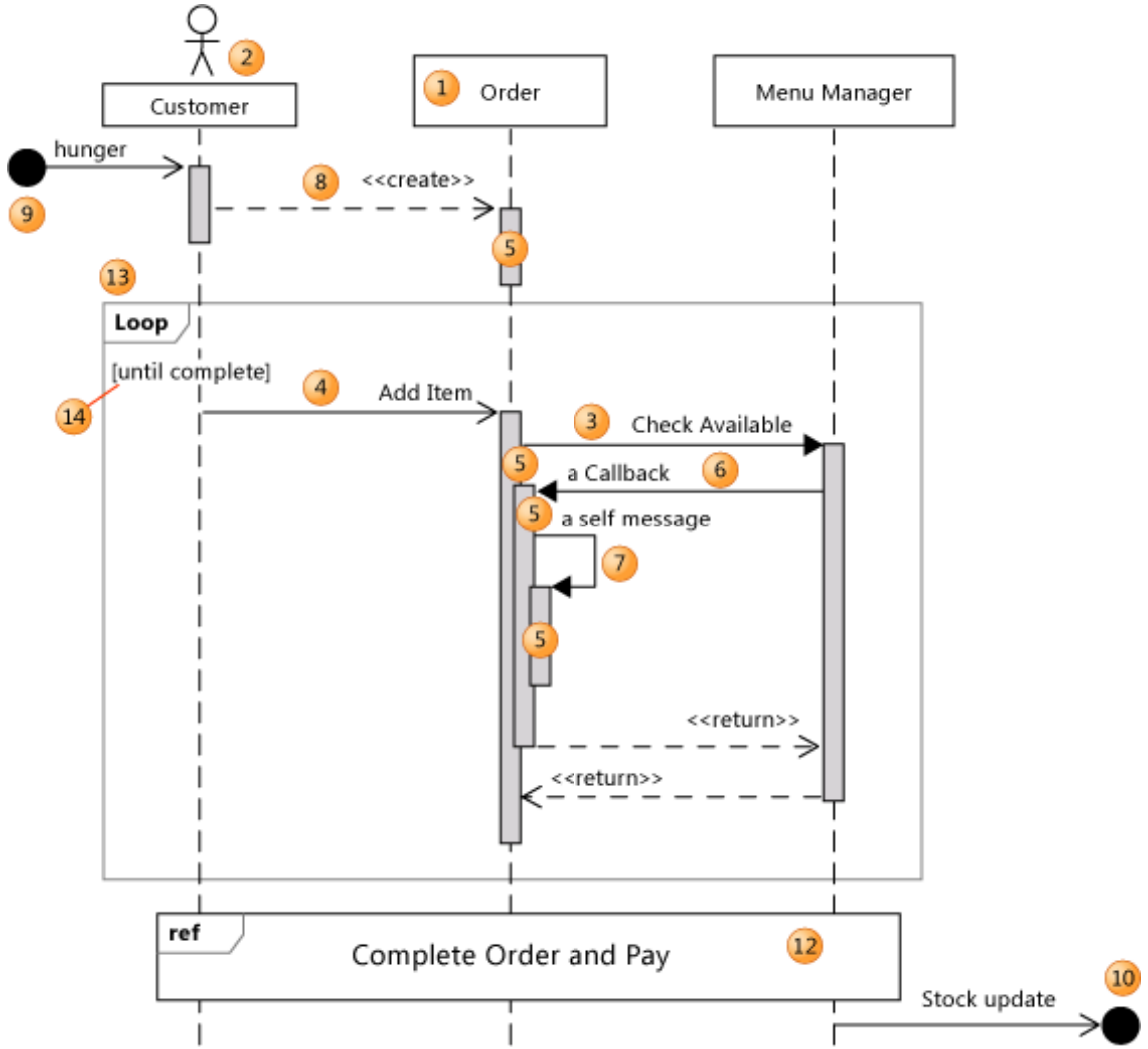


Диаграмма последовательности



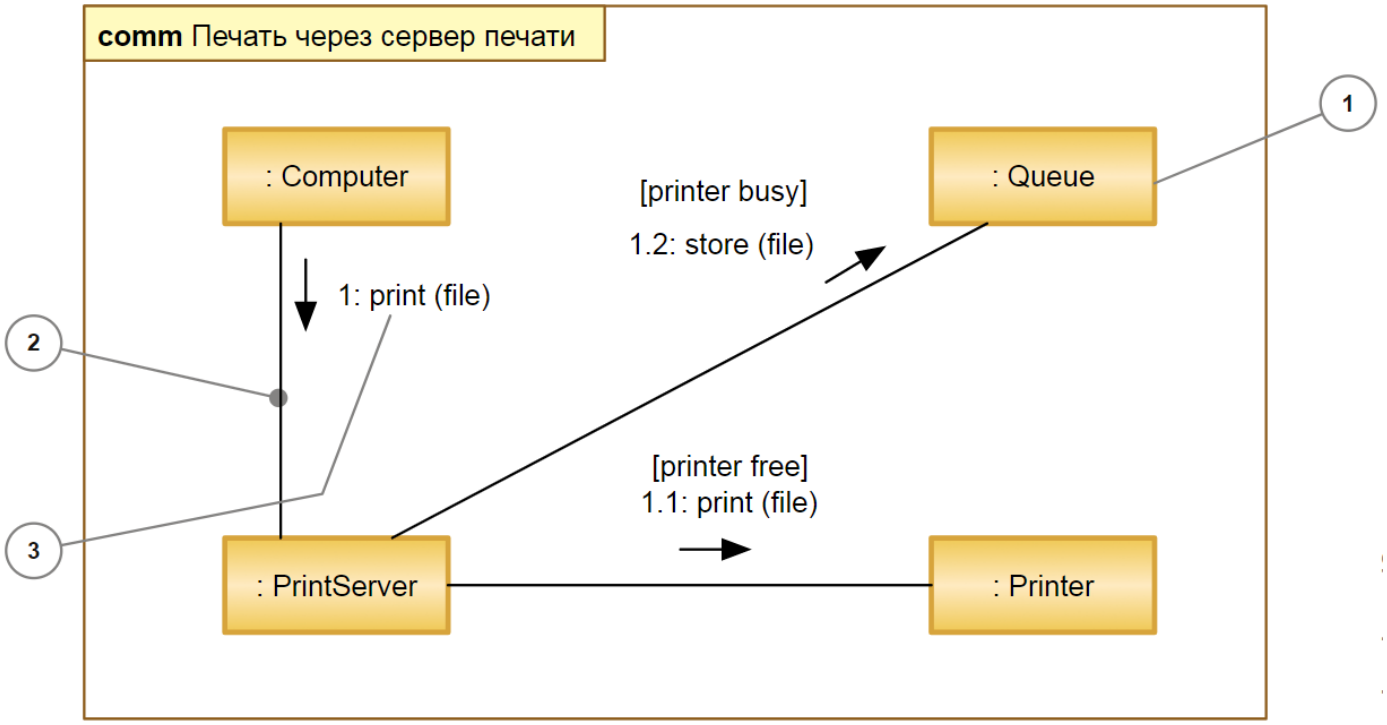
- **Диаграмма последовательности** (sequence diagram) – это способ описания поведения системы на основе указания последовательности передаваемых сообщений между участниками.
  - сущности – участники (1) (обычно объекты, классы, компоненты и действующие лица);
  - отношения – связи (2), по которым происходит обмен сообщениями (3);
  - ось времени, по умолчанию направлена сверху вниз, и то сообщение, которое отправлено позже, нарисовано ниже;
  - На линии жизни (4) участника отмечается его активация (5) или захват им управления;
  - Составные шаги (6) позволяют описывать сложные сценарии (ветвления, циклы)

Диаграмма последовательности (sequence diagram)



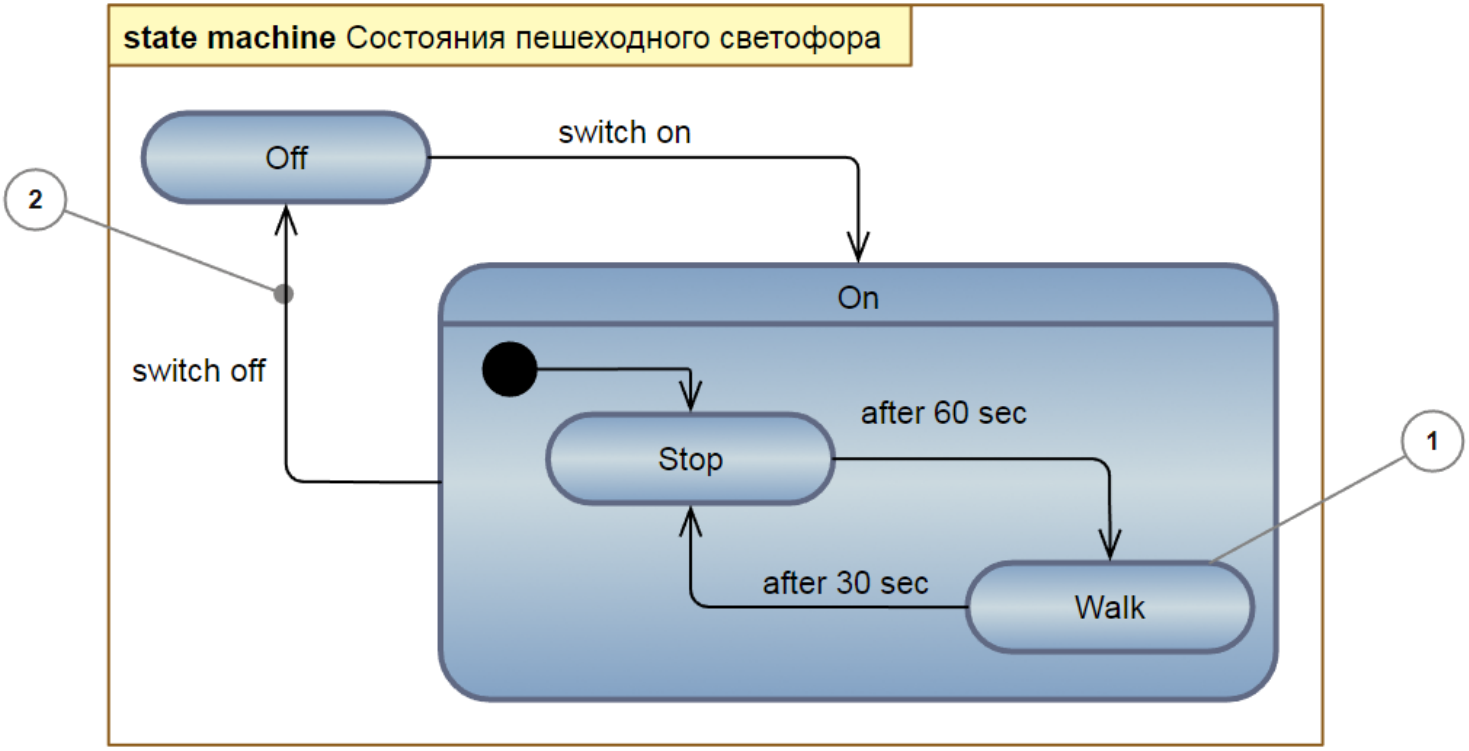
1 – Линия жизни, 2 – Действующее лицо, 3 – Синхронное сообщение, 4 - Асинхронное сообщение,
5 -Исполнение, 6 – Сообщение обратного вызова, 7 – Сообщение самому себе, 8 – Сообщение создания,
9, 10 – Внешние сообщения, 12 – Ссылка на другую диаграмму, 13 – Составной шаг, 14 – Сторожевое выражение,

Диаграмма коммуникации



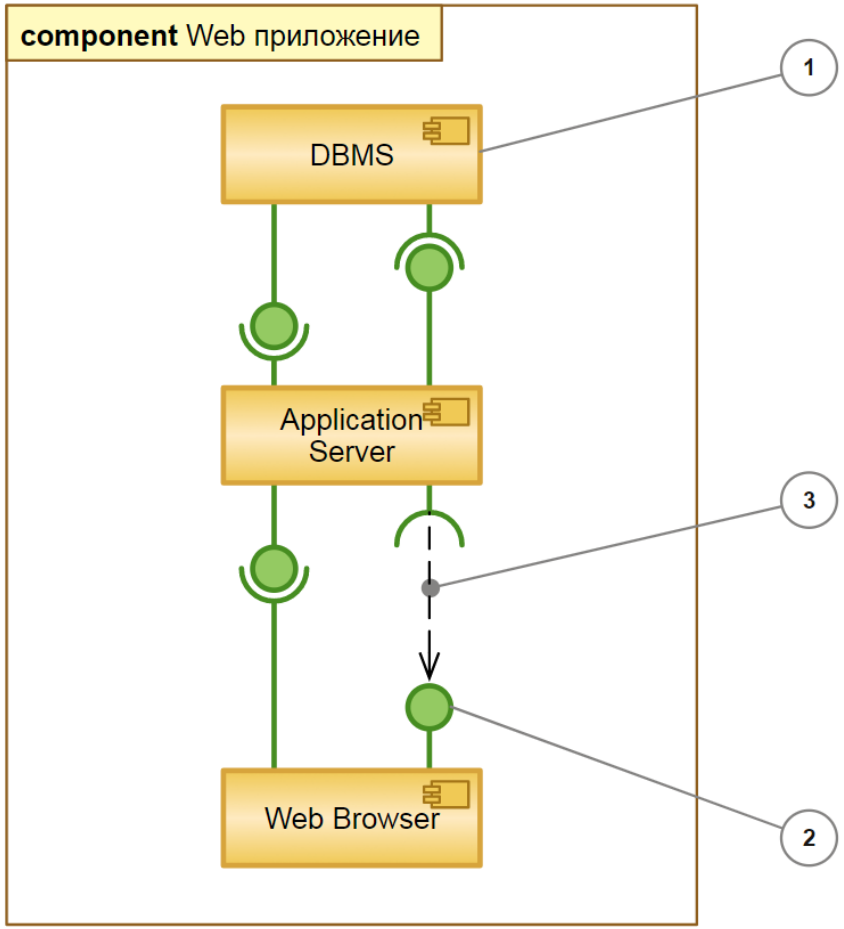
- **Диаграмма коммуникации** (communication diagram) – – способ описания поведения, семантически эквивалентный диаграмме последовательности, однако здесь акцент делается не на времени, а на структуре связей между конкретными экземплярами:
  - сущности – участники (1) (обычно объекты, классы, компоненты и действующие лица);
  - отношения – связи (2), по которым происходит обмен сообщениями (3), для отображения упорядоченности сообщений во времени применяется иерархическая десятичная нумерация.

Диаграмма автомата



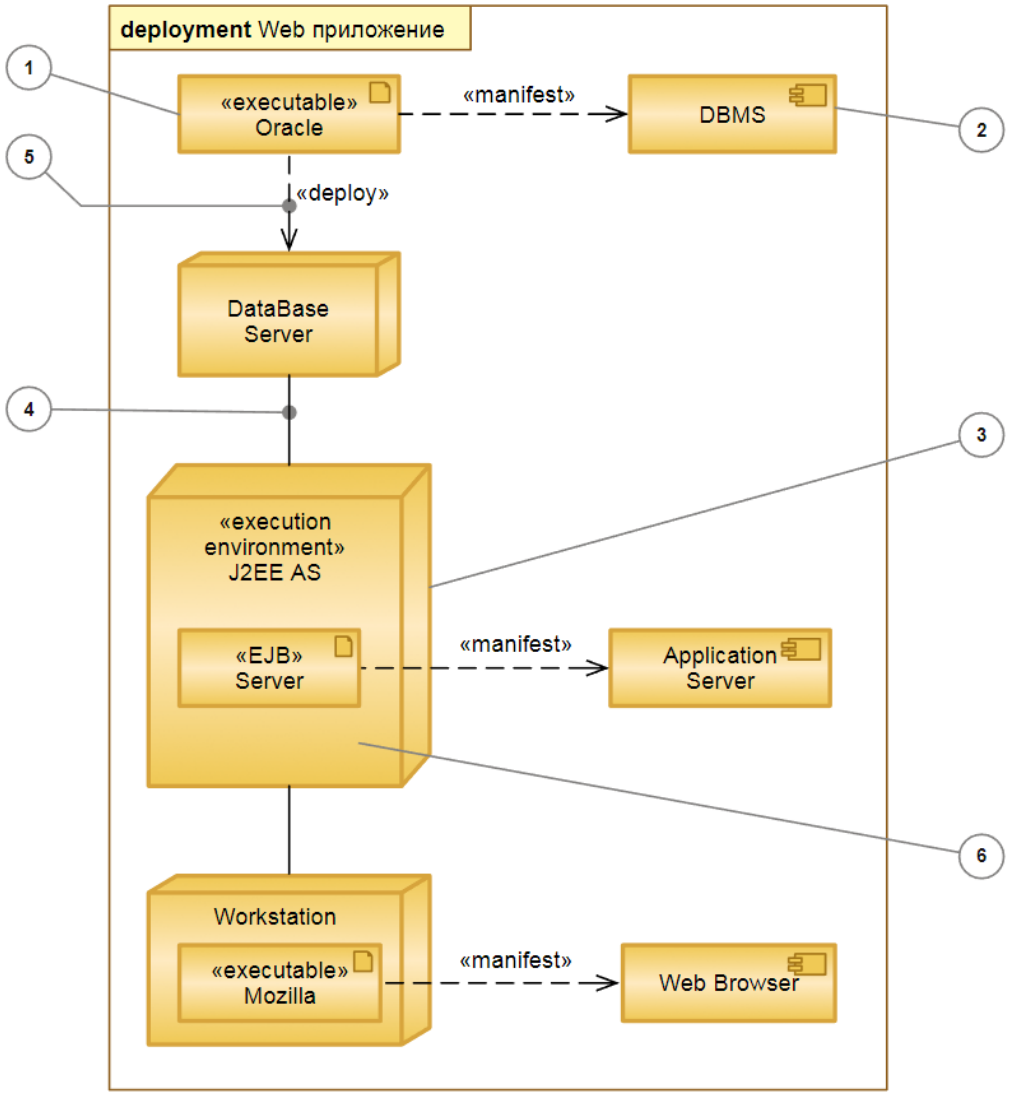
- **Диаграмма автомата** (state machine diagram) – это один из способов детального описания поведения в UML на основе явного выделения состояний и описания переходов между состояниями:
  - сущности – состояния (1)
  - отношения – переходы (2)

Диаграмма компонентов



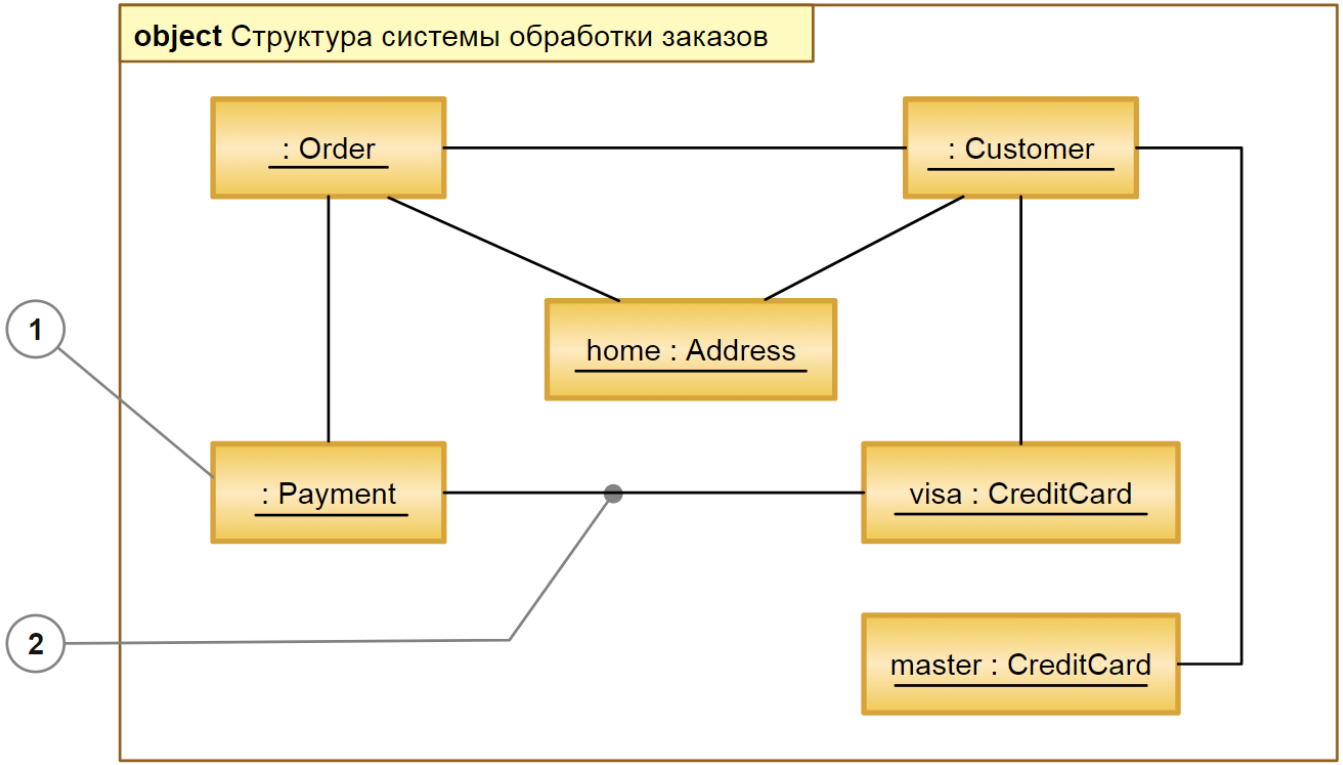
- **Диаграмма компонентов** (component diagram) – показывает взаимосвязи между модулями (логическими или физическими), из которых состоит моделируемая система.
  - сущности – компоненты (1), и интерфейсы (2), посредством которых указывается взаимосвязь между компонентами.
  - отношения – реализации между компонентами и интерфейсами (компонент реализует интерфейс) и зависимости между компонентами и интерфейсами (компонент использует интерфейс) (3).

Диаграмма размещения



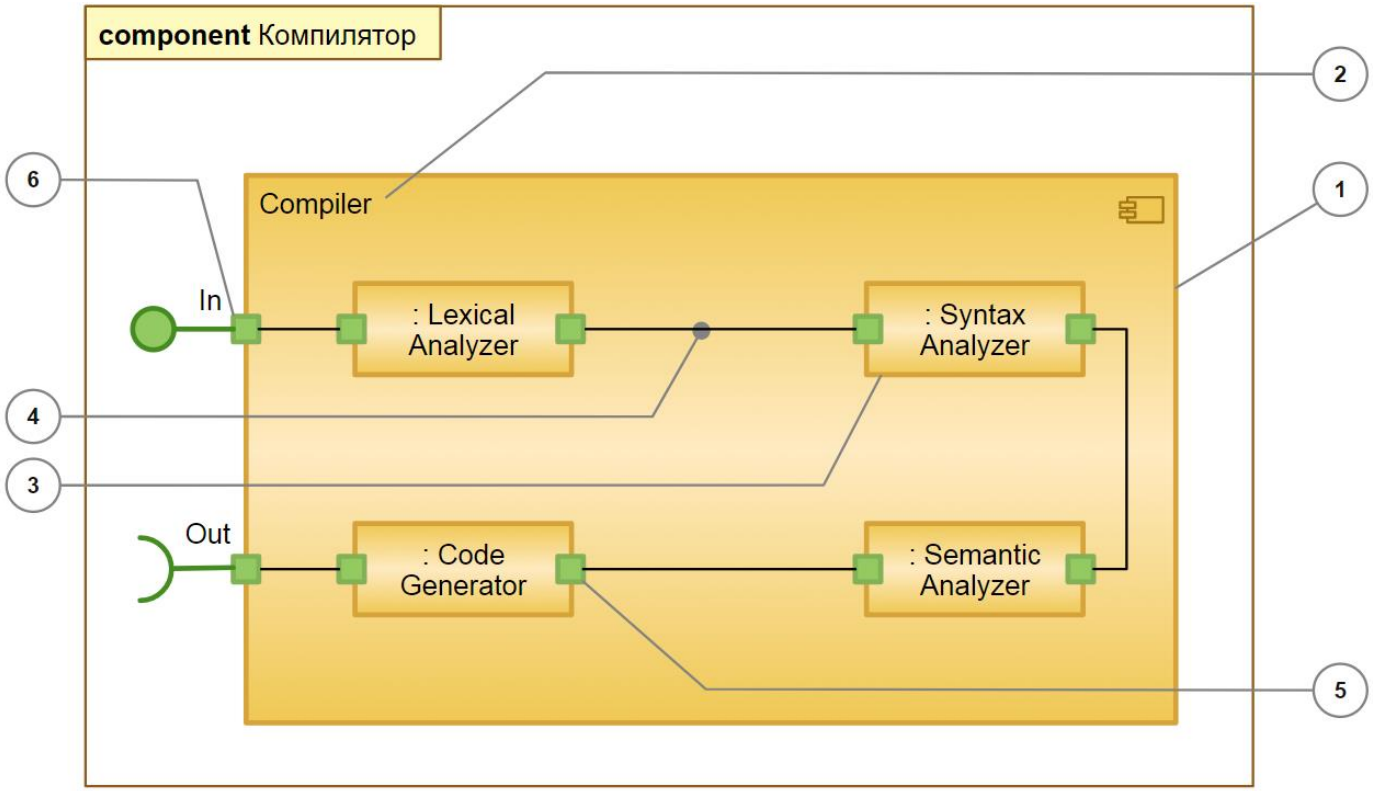
- **Диаграмма размещения** (deployment diagram) наряду с отображением состава и связей элементов системы показывает, как они физически размещены на вычислительных ресурсах во время выполнения.
- Сущности:
  - артефакт (1), который является реализацией компонента (2)
  - узел (3) на котором размещается артефакт
- Отношения
  - ассоциации между узлами (4), показывающее, что узлы физически связаны во время выполнения.
  - для того чтобы показать, что одна сущность является частью другой, применяется отношение зависимости «deploy» (5), либо фигура одной сущности помещается внутрь фигуры другой сущности (6)

Диаграмма объектов



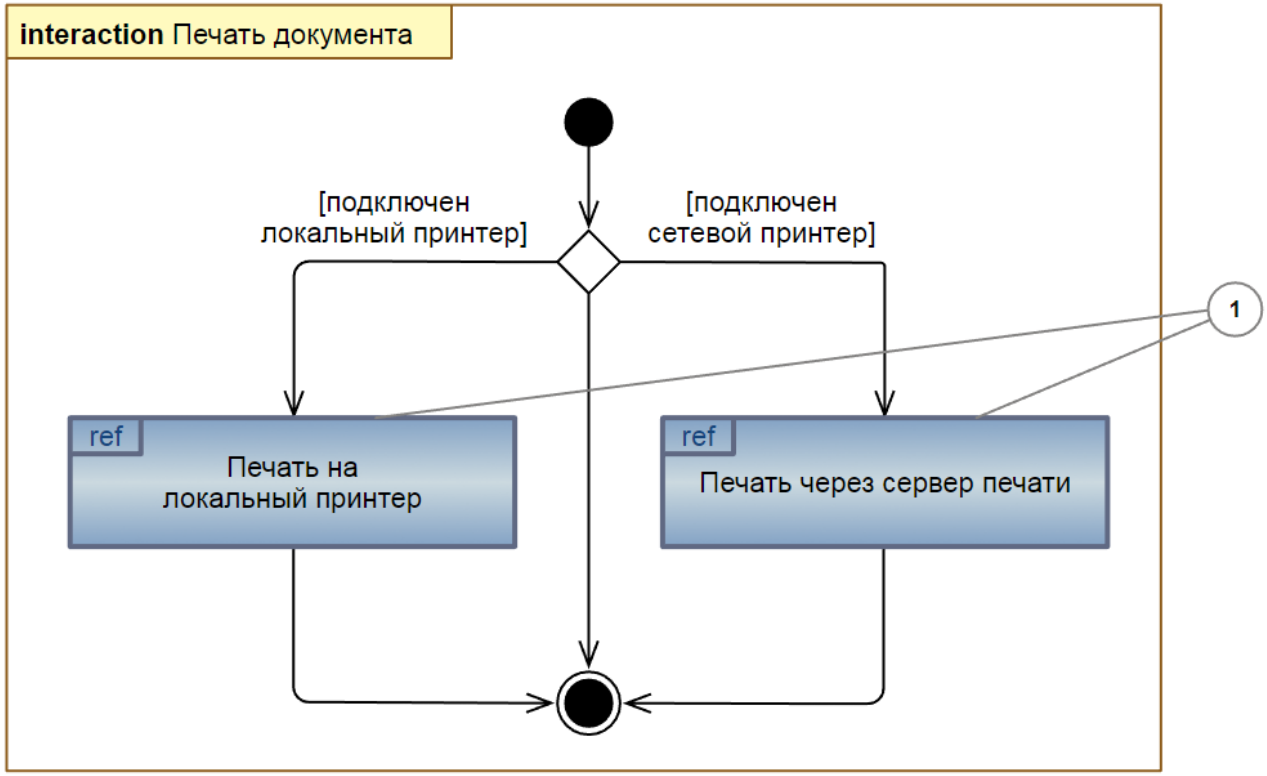
- **Диаграмма объектов** (object diagram) – является экземпляром диаграммы классов
  - сущности – объекты(1)
  - отношения – связи, чаще всего-ассоциации (2)

Диаграмма внутренней структуры



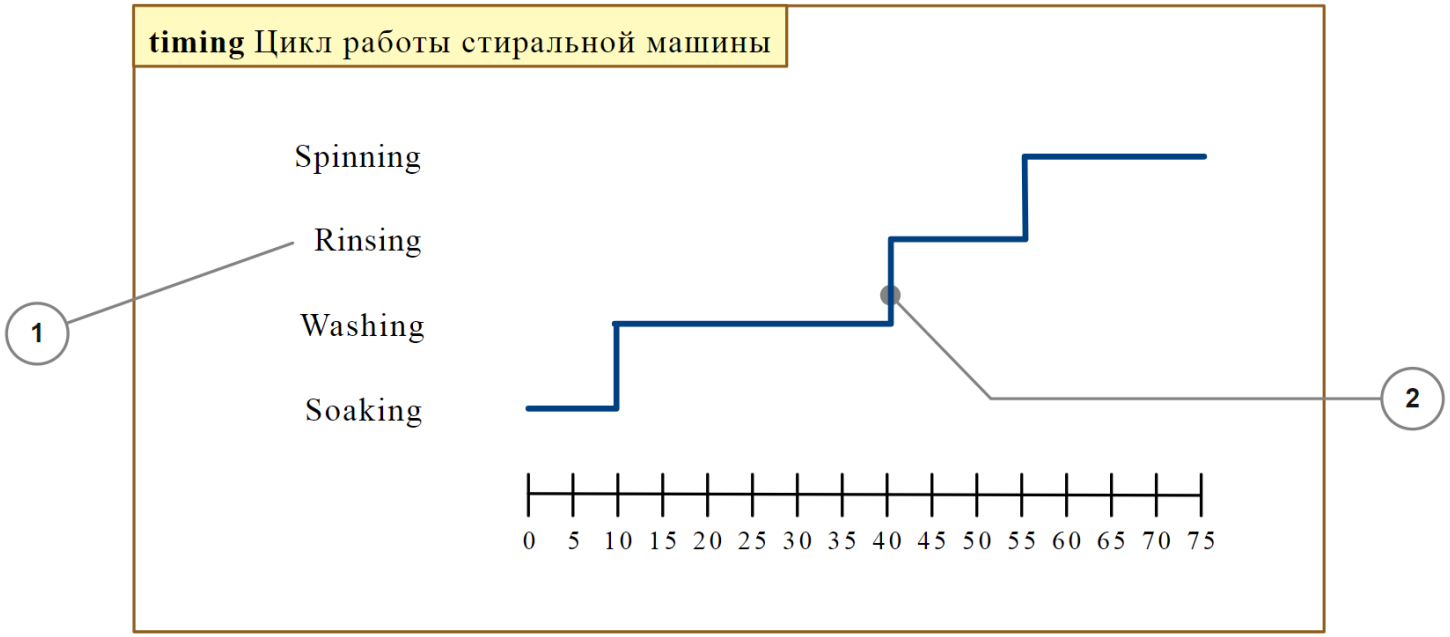
- **Диаграмма внутренней структуры** (composite structure diagram) используется для более подробного представления структурных классификаторов, прежде всего классов и компонентов, изображающихся в виде прямоугольника (1), в верхней части которого находится имя классификатора (2).
- Внутри находятся *части* (3). Каждая часть является экземпляром некоторого другого классификатора. Части могут взаимодействовать друг с другом. Это обозначается с помощью *соединителей* (4) различных видов.
- Место на внешней границе части, к которому присоединяется соединитель, называется *портом* (5). Порты располагаются также на внешней границе структурного классификатора (6), обеспечивая ему связь с внешним миром.

Обзорная диаграмма взаимодействия



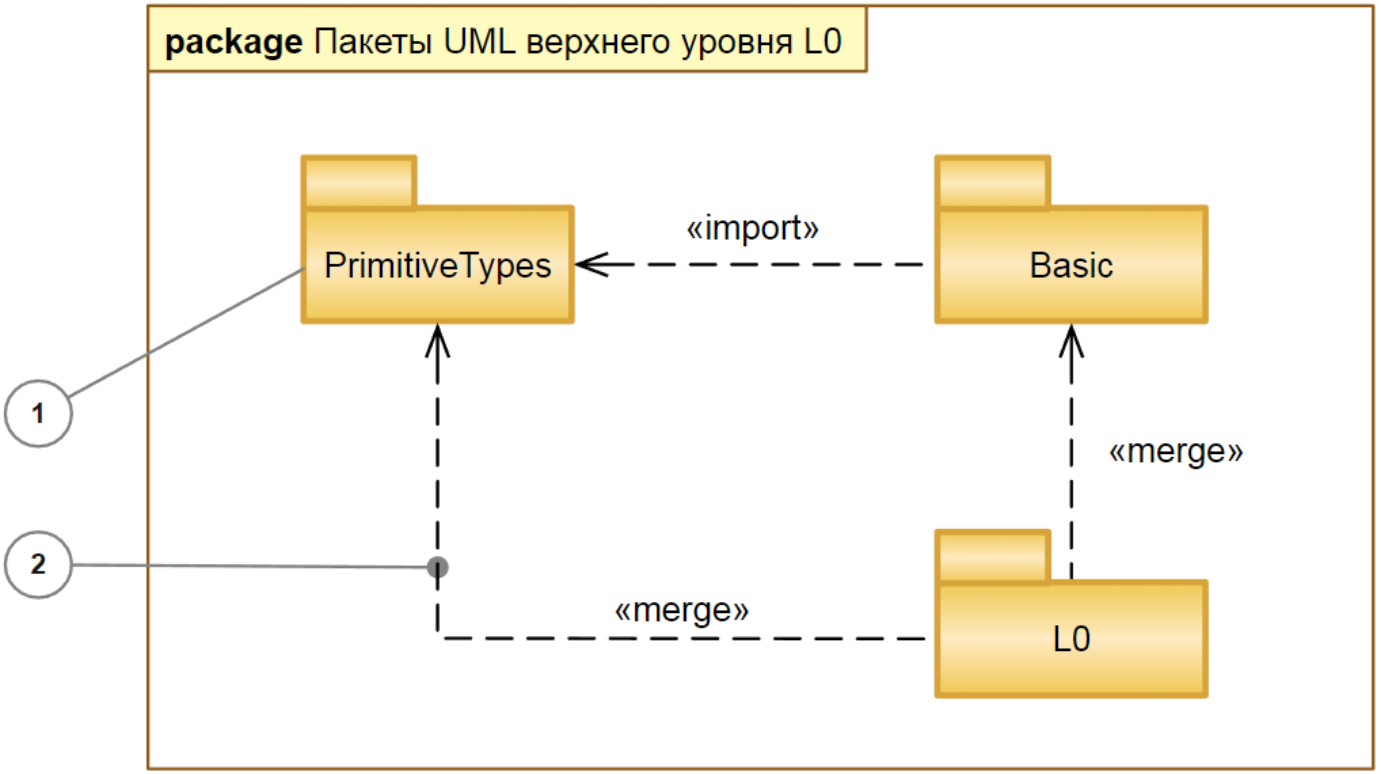
- **Обзорная диаграмма взаимодействия** (interaction overview diagram) является разновидностью диаграммы деятельности с расширенным синтаксисом: в качестве элементов обзорной диаграммы взаимодействия могут выступать ссылки на взаимодействия (1), определяемые диаграммами последовательности.

Диаграмма синхронизации



- **Диаграмма синхронизации** (timing diagram) представляет собой особую форму диаграммы последовательности, на которой особое внимание уделяется изменению состояний (1) различных экземпляров классификаторов и их временной синхронизации (2).

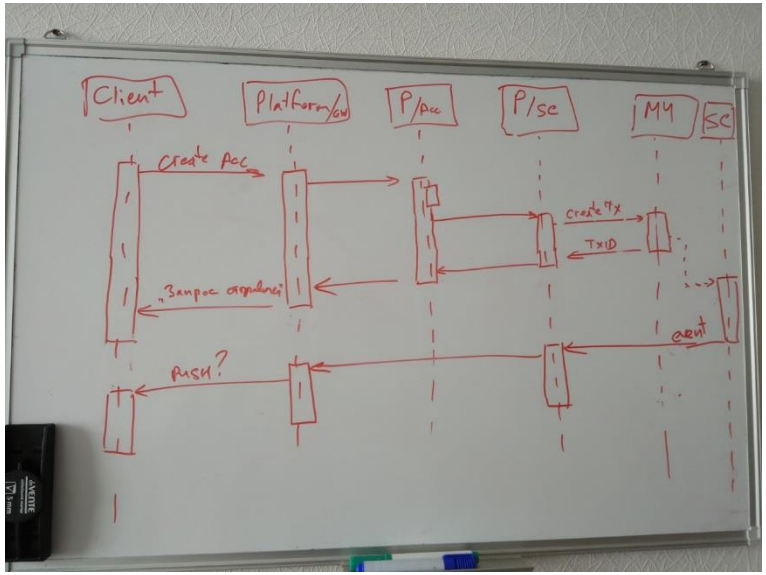
Диаграмма пакетов

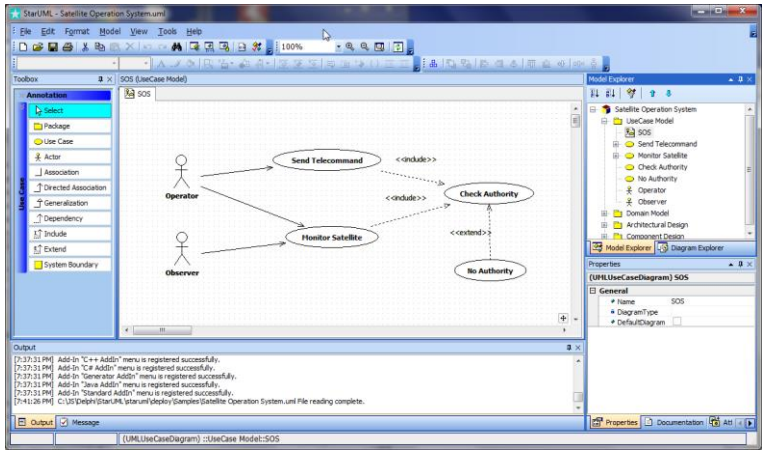


- **Диаграмма пакетов** (package diagram) – средство группирования элементов модели, единственное средство, позволяющее управлять сложностью самой модели.
- Основные элементы нотации – пакеты (1) и зависимости с различными стереотипами (2)

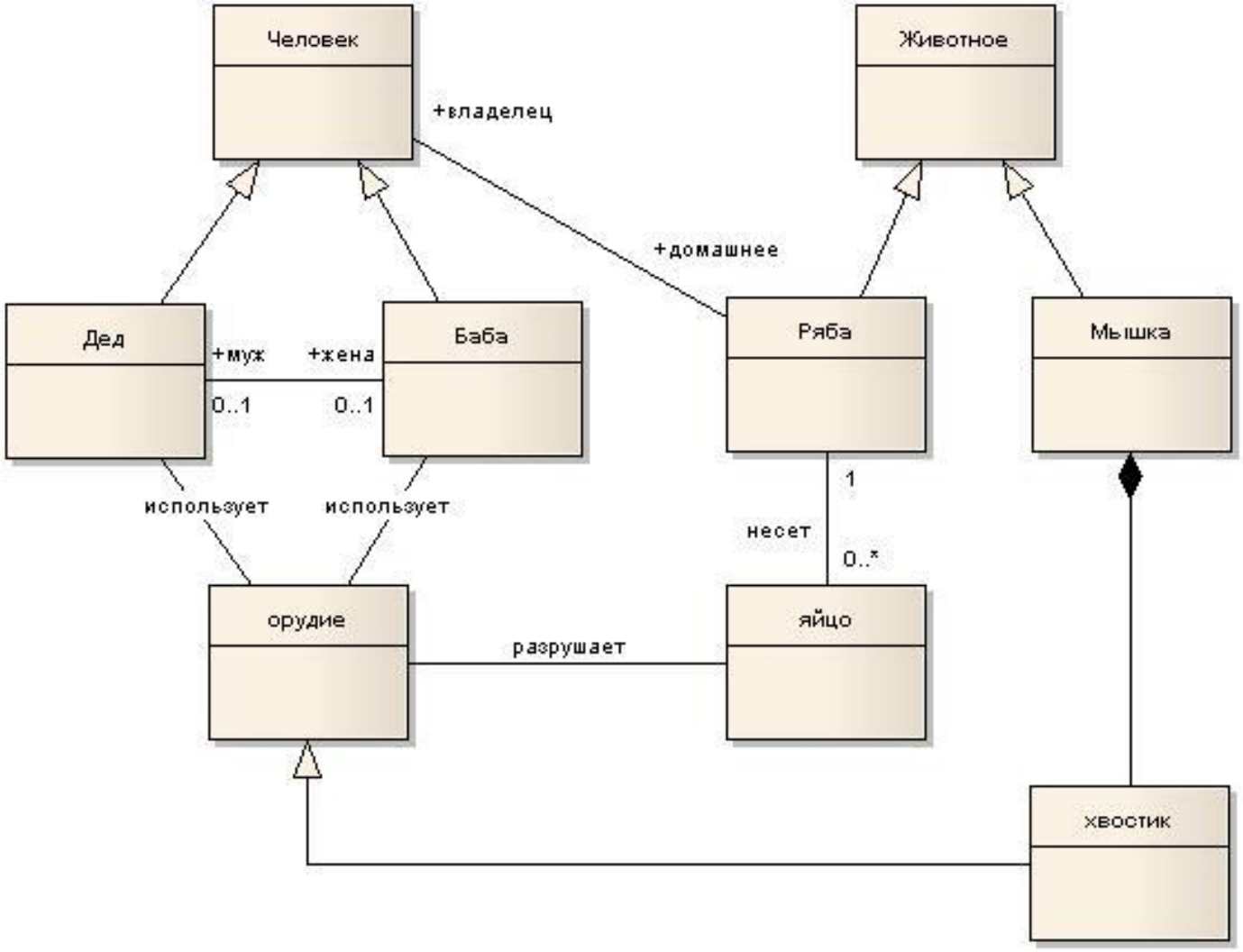
UML инструменты

- Любые изобразительные средства:
  - Бумага + пишущие принадлежности
  - Доска + маркеры
- Программные продукты:
  - Rational Software – подразделение IBM. Выпускает ряд продуктов для моделирования систем. Семейство продуктов Rational Rose.
  - Free UML Tool.
  - White Star UML.
  - Violet UML editor.





И напоследок: Угадайте, что это?



А теперь немного в другом аспекте