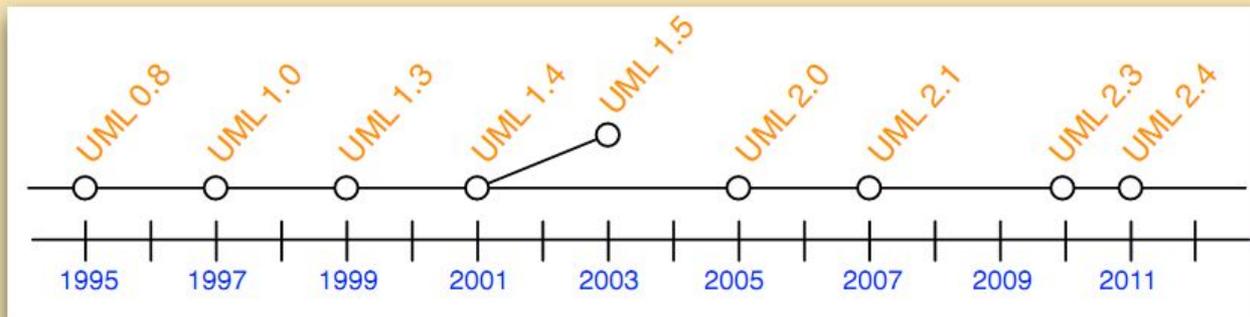


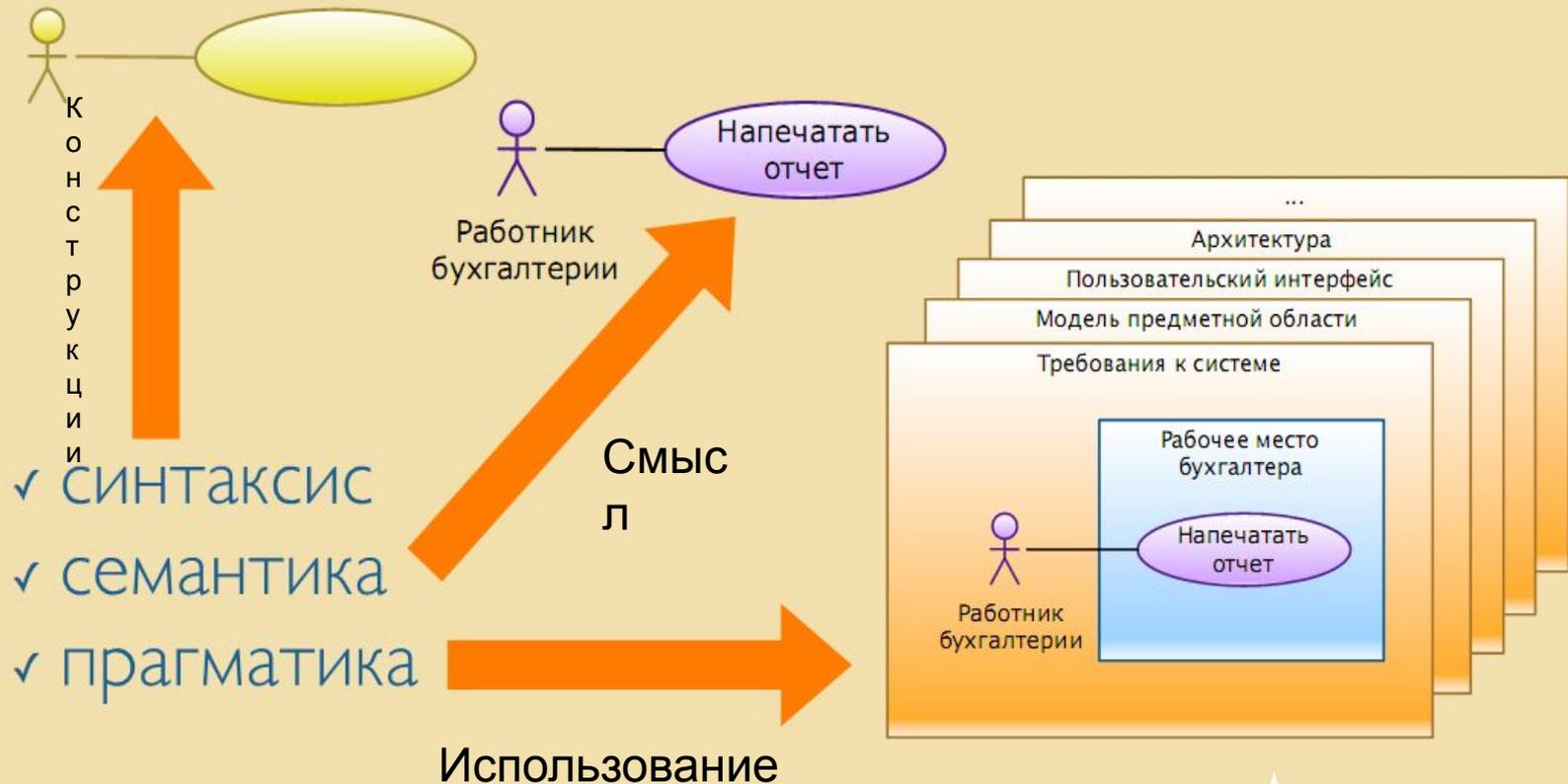
Применение UML при разработке программного обеспечения

Что такое UML (Unified Modeling Language)

- UML - это **язык**
 - знаковая система для хранения и передачи информации
- UML - это язык **моделирования**
 - Модель - это описание системы, предметной области, явления
- UML - это **унифицированный** язык моделирования



Сущность UML



UML – графический язык моделирования, предназначенный для **спецификации, визуализации, конструирования** и **документирования** систем

Назначение - спецификация, визуализация, конструирование, документирование

Спецификация = описание (системы)

Средство описания - Как устроена и работает

взгляд заказчика \neq взгляд разработчик

Визуализация

Средство коммуникации (наглядность)



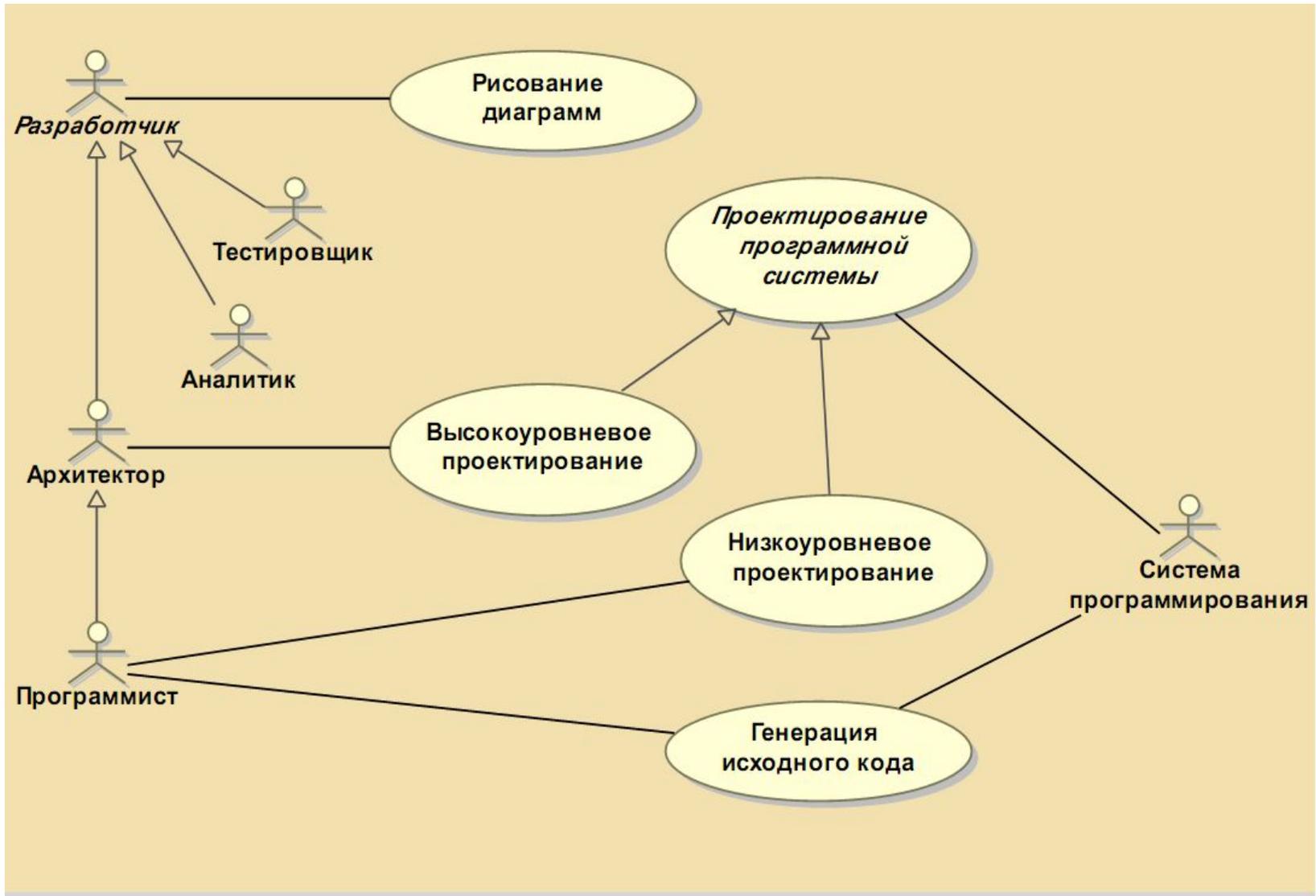
Конструирование = модель \rightarrow артефакт (документ)

\approx автоматический синтез программ

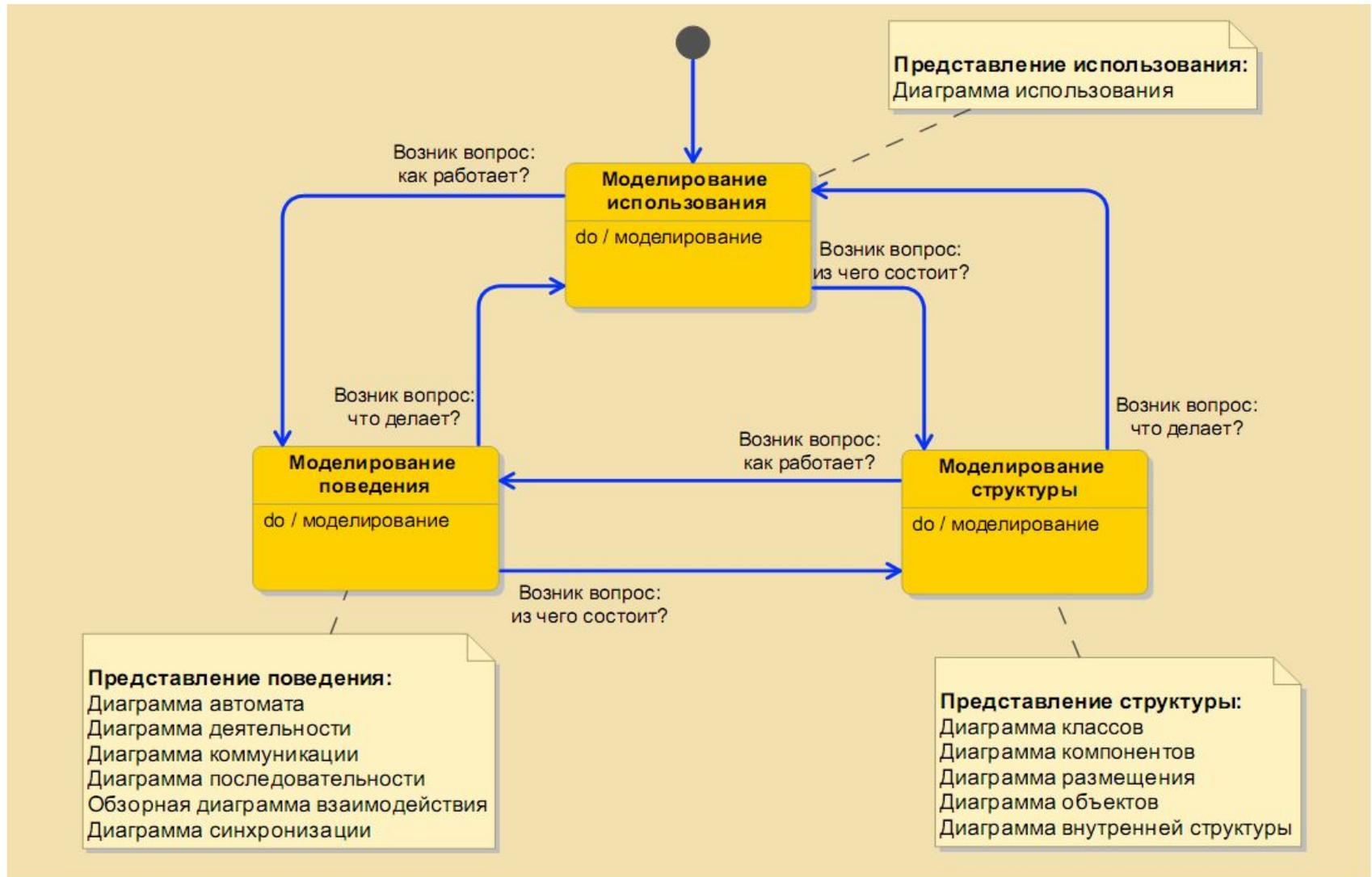
Документирование

все элементы модели могут быть описаны

Использование UML



Модель процесса моделирования



Сущности представлений

Представление использования

- Действующие лица
- Границы системы
- Функциональные требования
- Ответственность компонентов

Представление структуры

- Предметная область
- Архитектура системы
- Структура хранения
- Детали реализации

Представление поведения

- Бизнес-процессы
- Пользовательский интерфейс
- Алгоритмы обработки
- Жизненные циклы

Стандарт UML



UNIFIED MODELING LANGUAGE™



www.uml.org



Meta-Object Facility core specification, 07-08-2011, v 2.4.1



Infrastructure, 05-08-2011, v 2.4.1

230 стр.

Superstructure, 06-08-2011, v 2.4.1

748 стр.

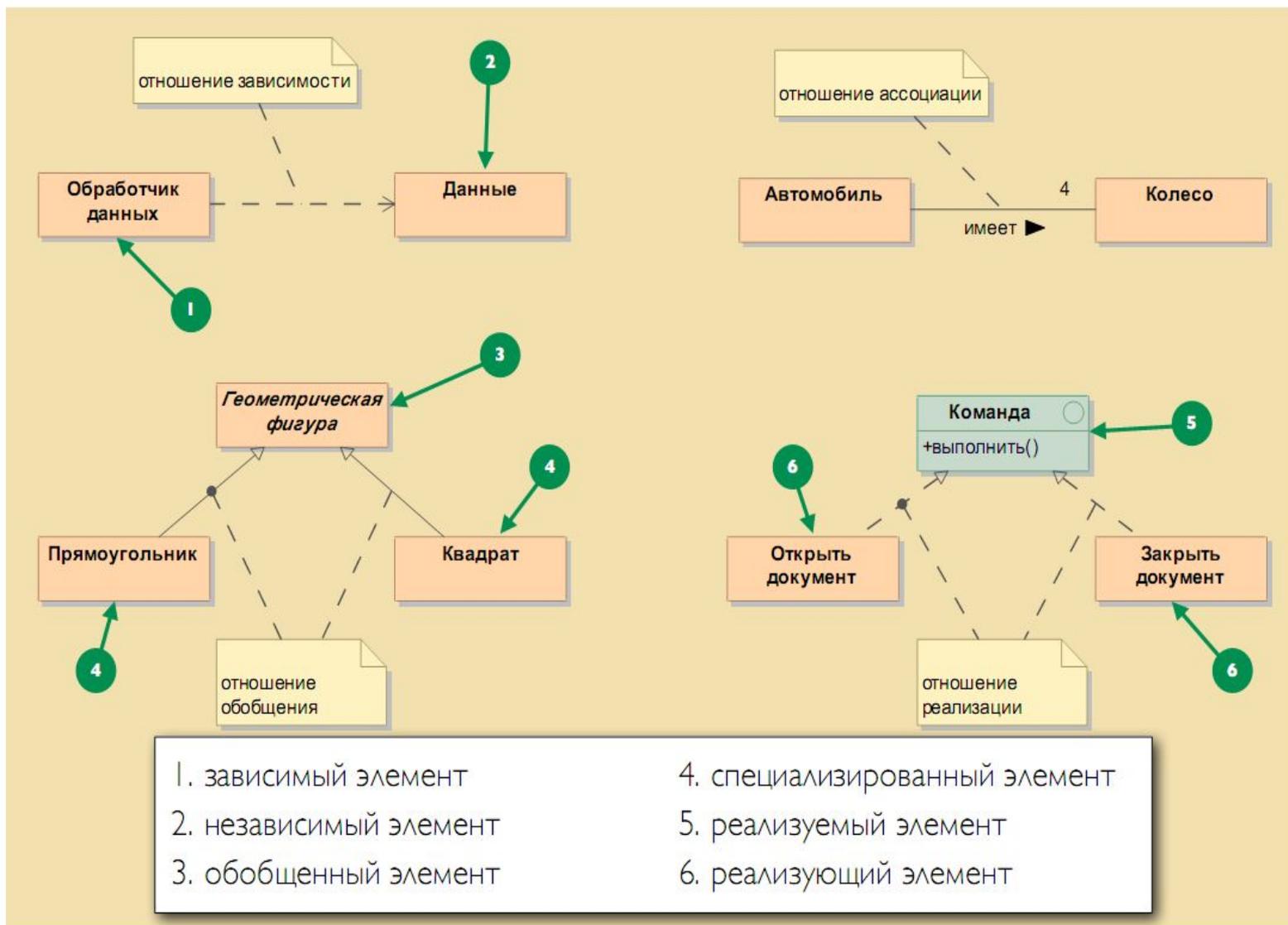
Object Constraint Language, 01-01-2012, v 2.3.1

UML Diagram Interchange specification, 04-2006, v 1.0

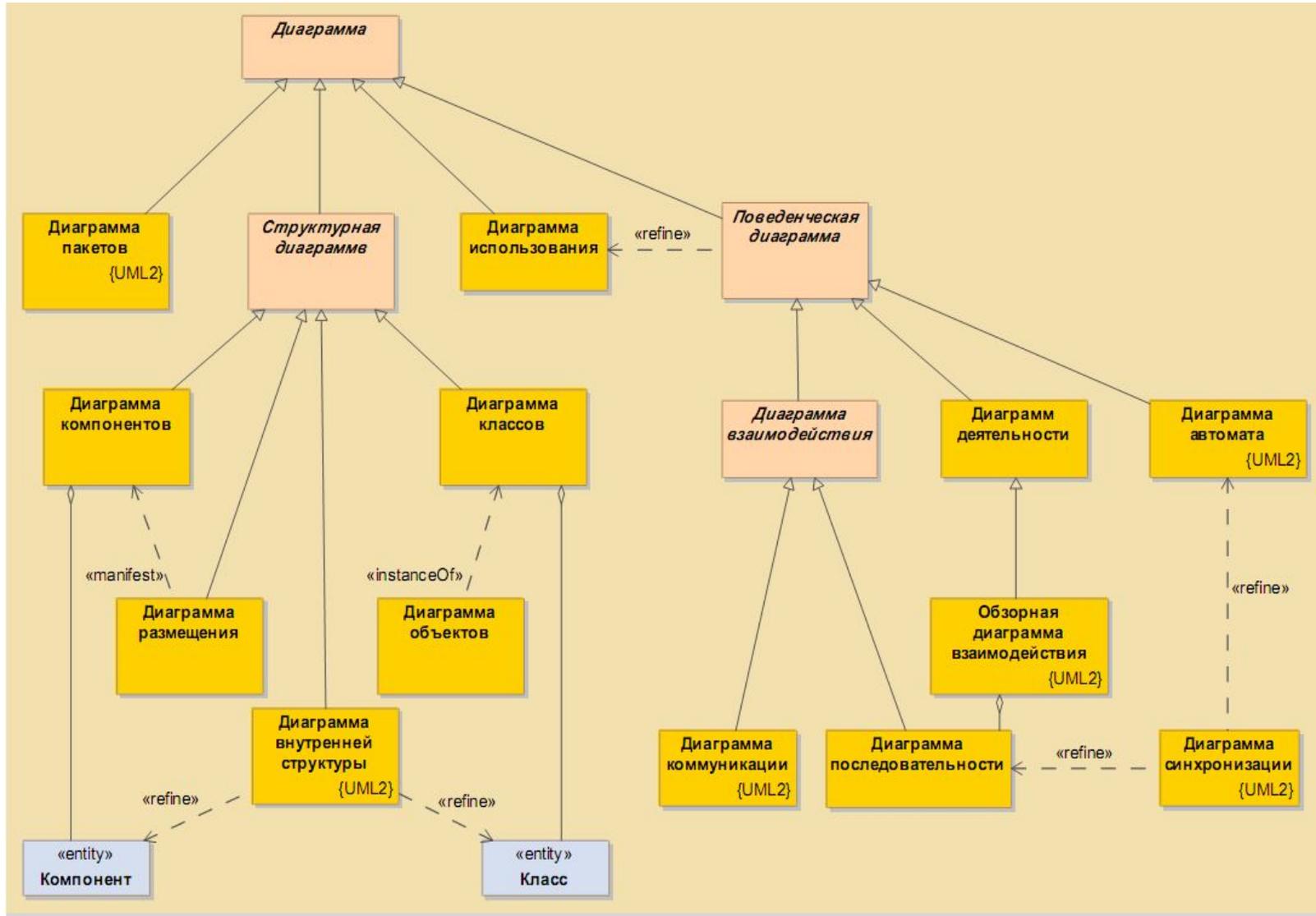
Сущности UML

Имя	Нотация	Имя	Нотация
Артефакт Artifact		Компонент Component	
Вариант использования Use case		Объект Object	
Действующее лицо Actor		Пакет Package	
Деятельность / действие Activity / action		Комментарий Comment	
Интерфейс Interface		Состояние State	
Класс Class		Вычислительный узел Node	
Активный класс Active class		Кооперация Collaboration	

Отношения UML



Диаграммы



Назначение диаграмм

Что делает система?

- Диаграмма использования / Use case diagram

Из чего состоит система?

- Диаграмма классов / Class diagram
- Диаграмма компонентов / Component diagram
- Диаграмма размещения / Deployment diagram
- Диаграмма объектов / Object diagram
- Диаграмма внутренней структуры / Composite structure diagram

Как работает система?

- Диаграмма автомата / State machine diagram
- Диаграмма деятельности / Activity diagram
- Диаграмма коммуникации / Communication diagram
- Диаграмма последовательности / Sequence diagram
- Обзорная диаграмма взаимодействия / Interaction overview diagram
- Диаграмма синхронизации / Timing diagram

Как уменьшить сложность модели?

- Диаграмма пакетов / Package diagram

Диаграммы использования (Use Case)

Диаграммы вариантов использования описывают функциональное назначение системы или то, что система должна делать. Разработка диаграммы преследует следующие цели:

- определить общие границы и контекст моделируемой предметной области;
- сформулировать общие требования к функциональному поведению проектируемой системы;
- разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
- подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

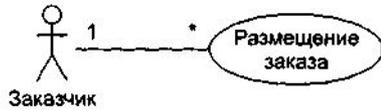
Отдельный **вариант использования** (прецедент) обозначается на диаграмме **эллипсом**, внутри которого содержится его краткое название.

Актер представляет собой любую внешнюю по отношению к моделируемой системе сущность, которая взаимодействует с системой и использует ее функциональные возможности для достижения определенных целей.

Стандартным графическим обозначением актера на диаграммах является **фигурка человечка**, под которой записывается имя актера.

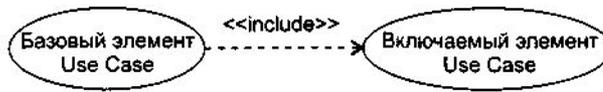
Отношения в диаграммах использования

Ассоциация



Включение

Включаемый элемент является составной частью базового элемента



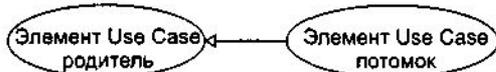
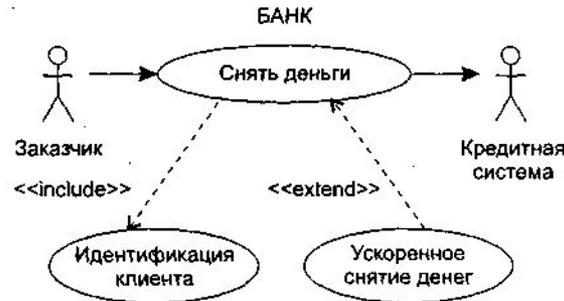
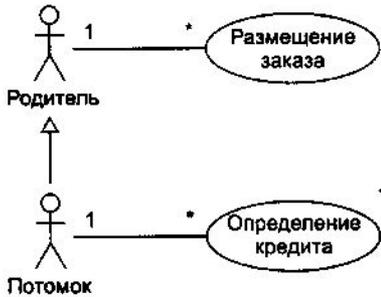
Расширение

Частный вариант использования



Обобщение

Потомок наследует поведение родителя



Пример диаграммы использования

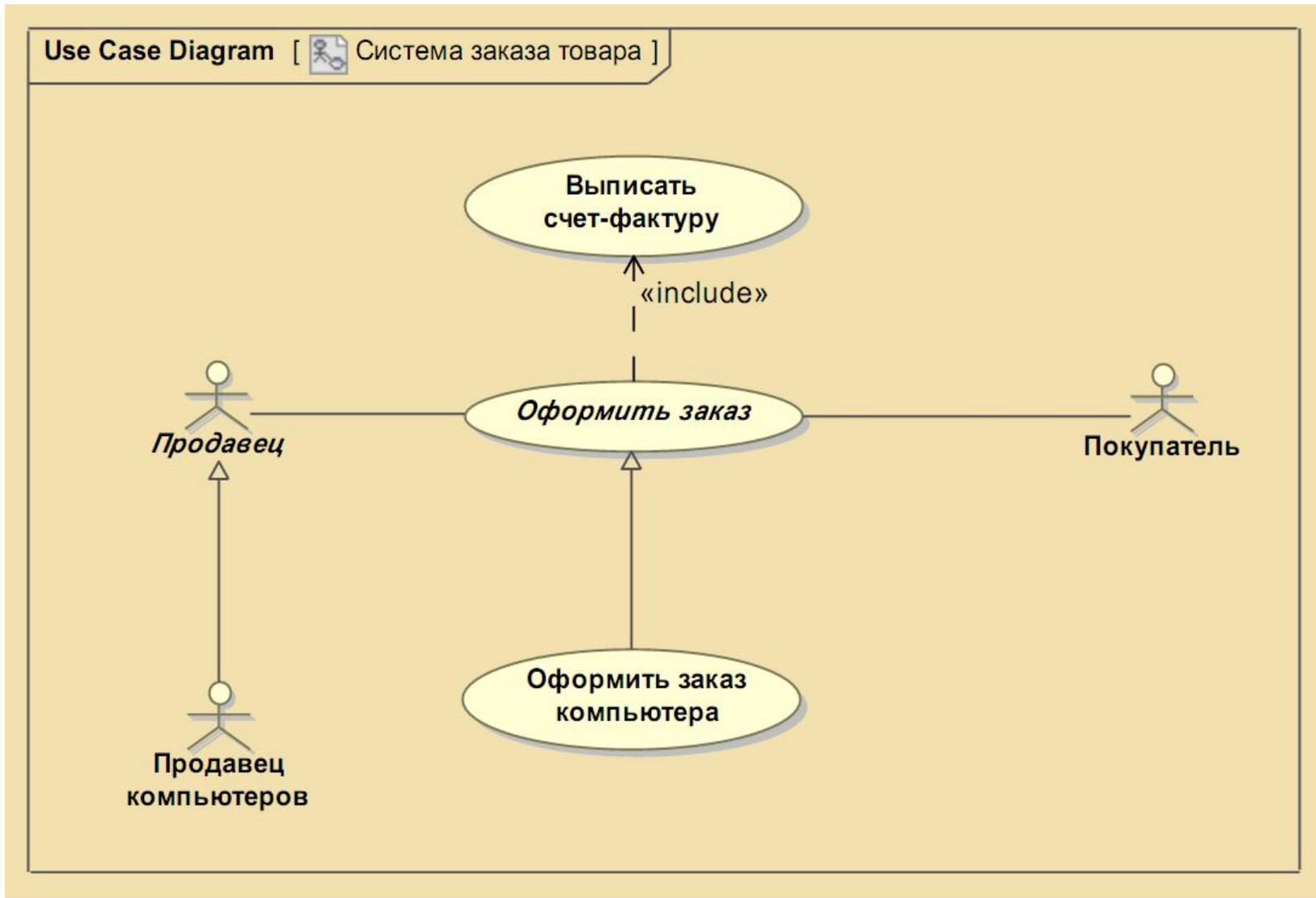
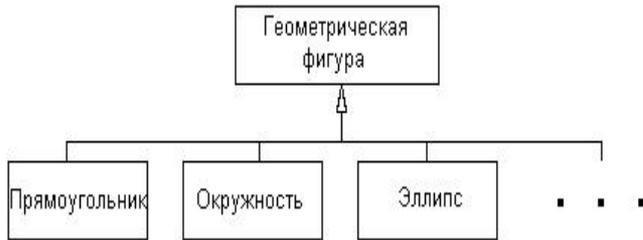


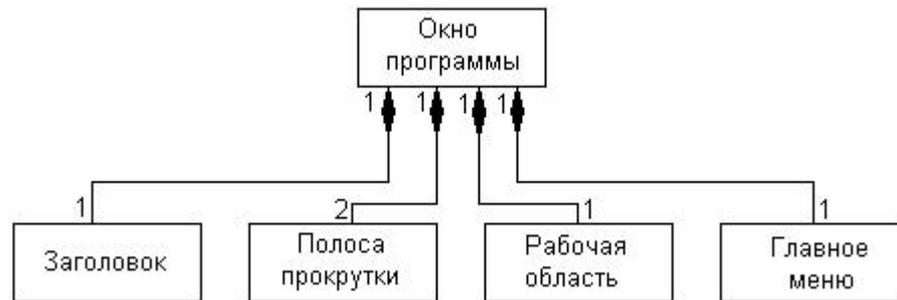
Диаграмма классов

Диаграмма классов представляет собой граф, вершинами которого являются элементы типа «классификатор», связанные различными типами структурных отношений.

Обобщение (наследовани



Композиции Сильная агрегация

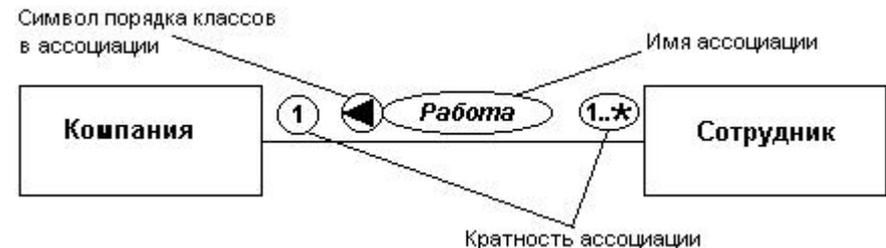


Агрегации



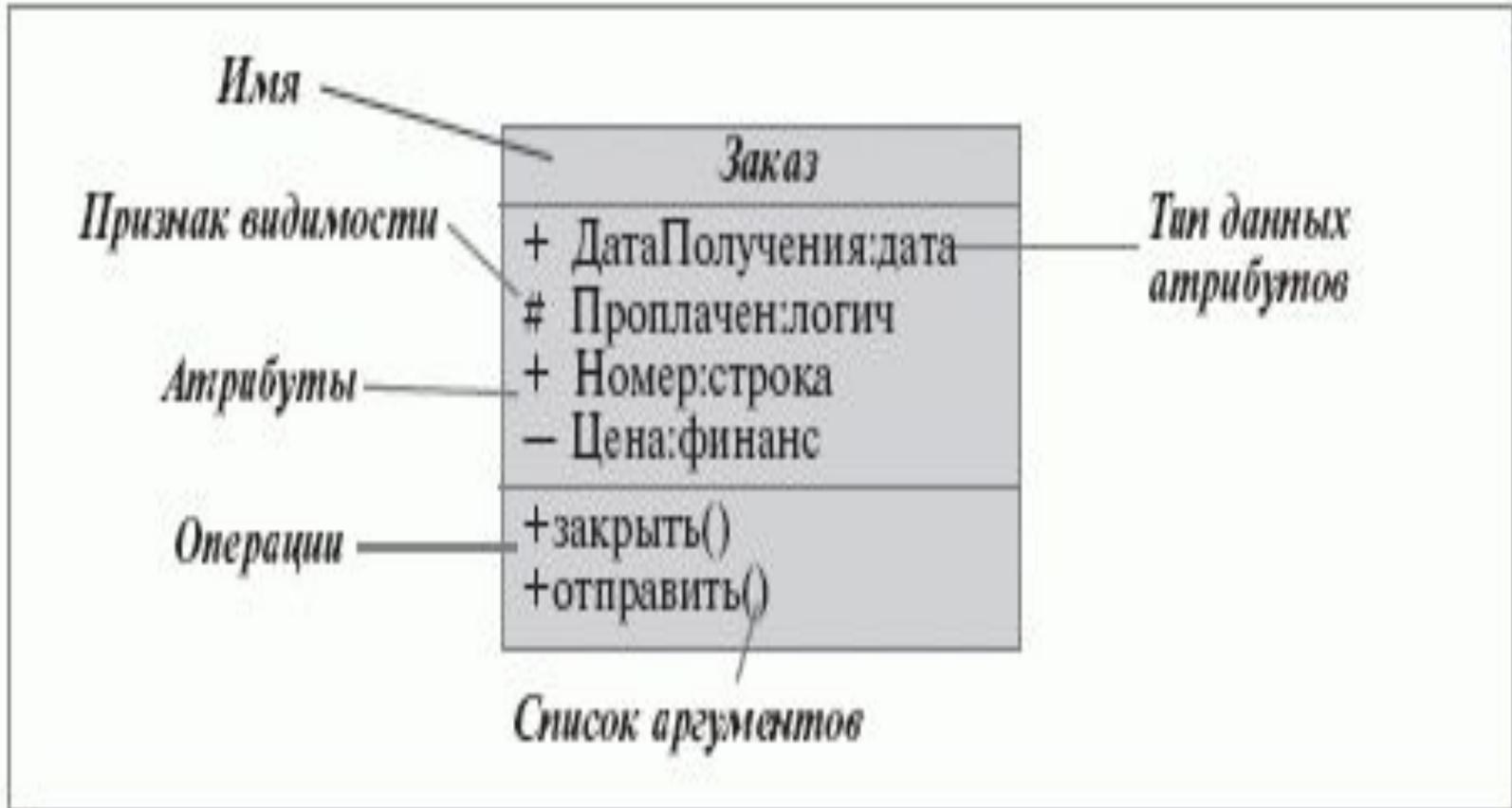
Ассоциации

Отношения между экземплярами класса

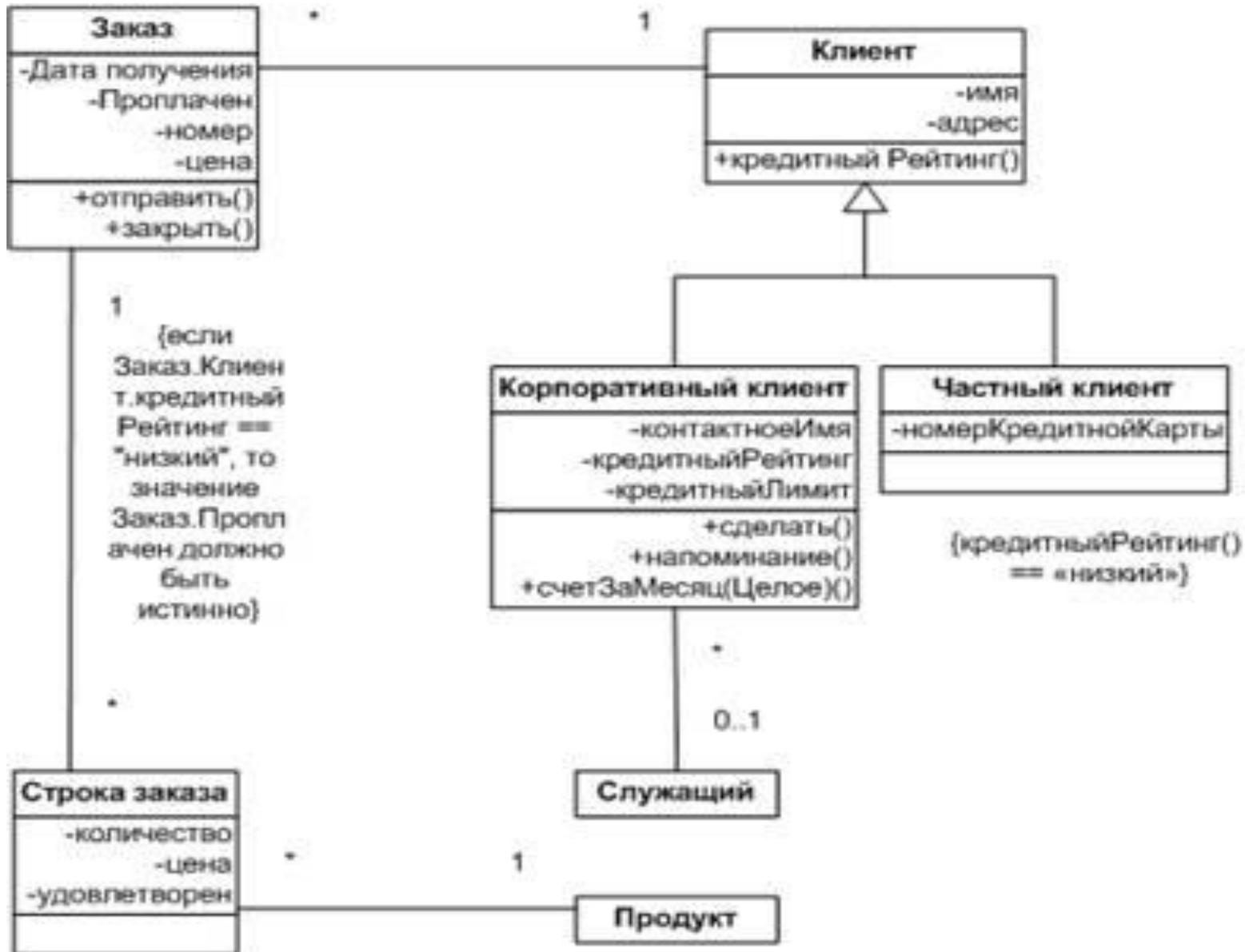


Классификатор

Классификатор – это элемент, описывающий структурные и поведенческие свойства.



Пример диаграммы классов



Пример диаграммы классов

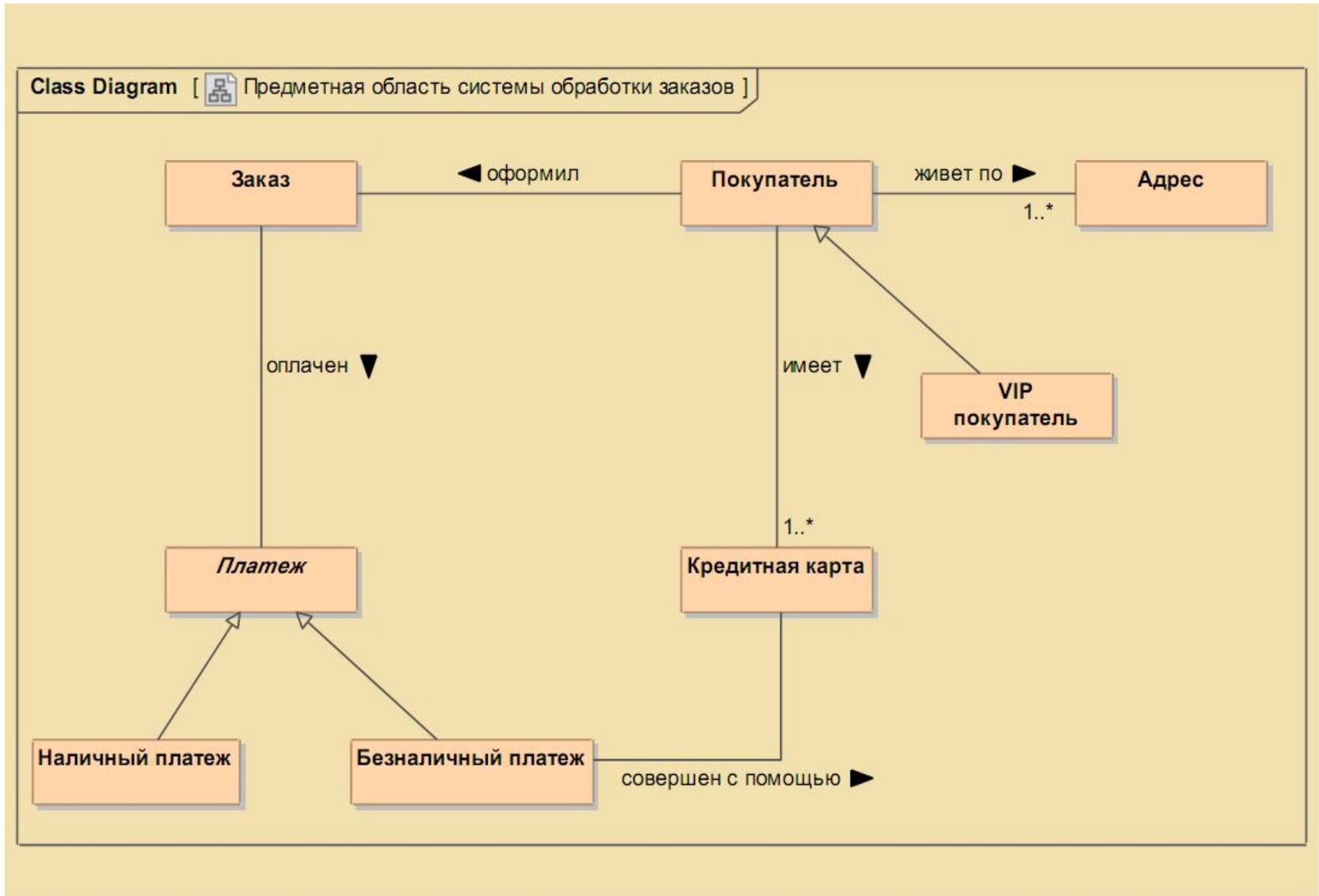


Диаграмма объектов

Диаграмма объектов представляет статический «моментальный снимок» с экземпляров предметов, которые находятся в диаграммах классов.

Диаграмма объектов характеризуется следующими свойствами:

- акцентирует внимание на одном аспекте статического вида системы с точки зрения проектирования или процессов;
- представляет лишь один из кадров динамического сценария, показанного на диаграмме взаимодействия;
- содержит только существенные для понимания данного аспекта элементы;
- уровень ее детализации соответствует уровню абстракции системы. (Показывайте только те значения атрибутов и дополнения, которые существенны для понимания);

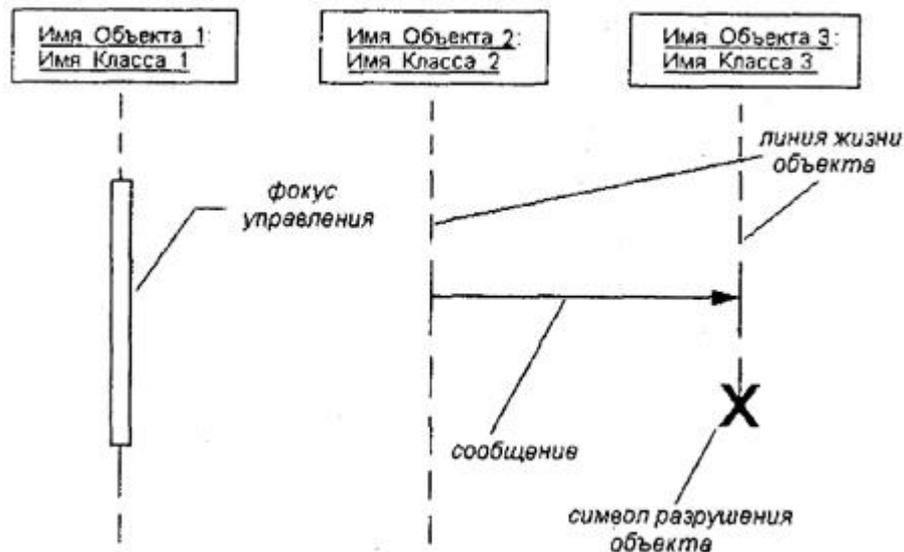
Пример диаграммы объектов



Диаграмма последовательности

Данный вид диаграмм отражает следующие аспекты проектируемой Системы:

- обмен сообщениями между объектами;
- ограничения, накладываемые на взаимодействие объектов;
- события, инициирующие взаимодействия объектов.



Элементы диаграммы последовательности

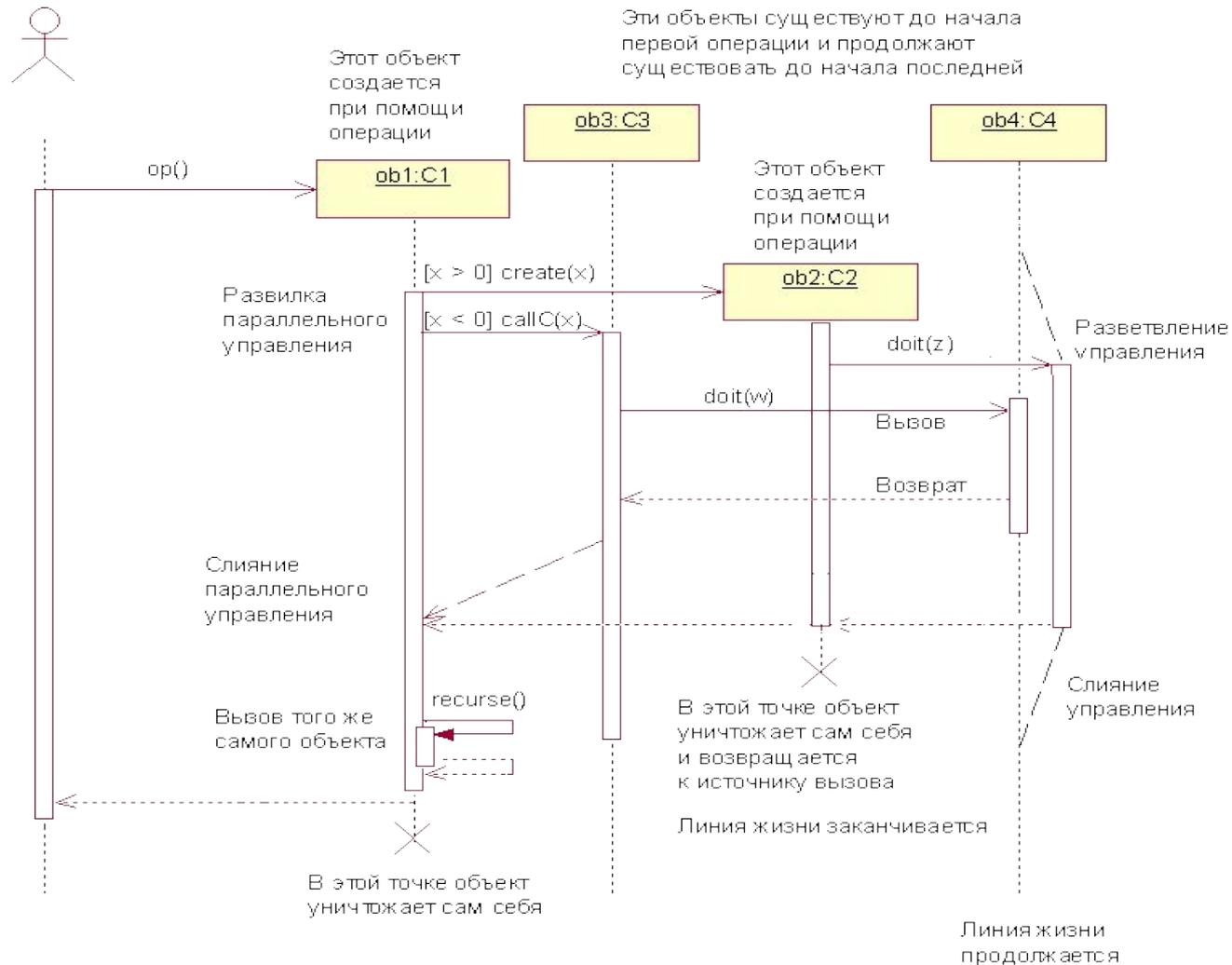


Рис. 162. Диаграмма последовательности с процедурным потоком управления

Пример диаграммы последовательности

Просмотр телевизора

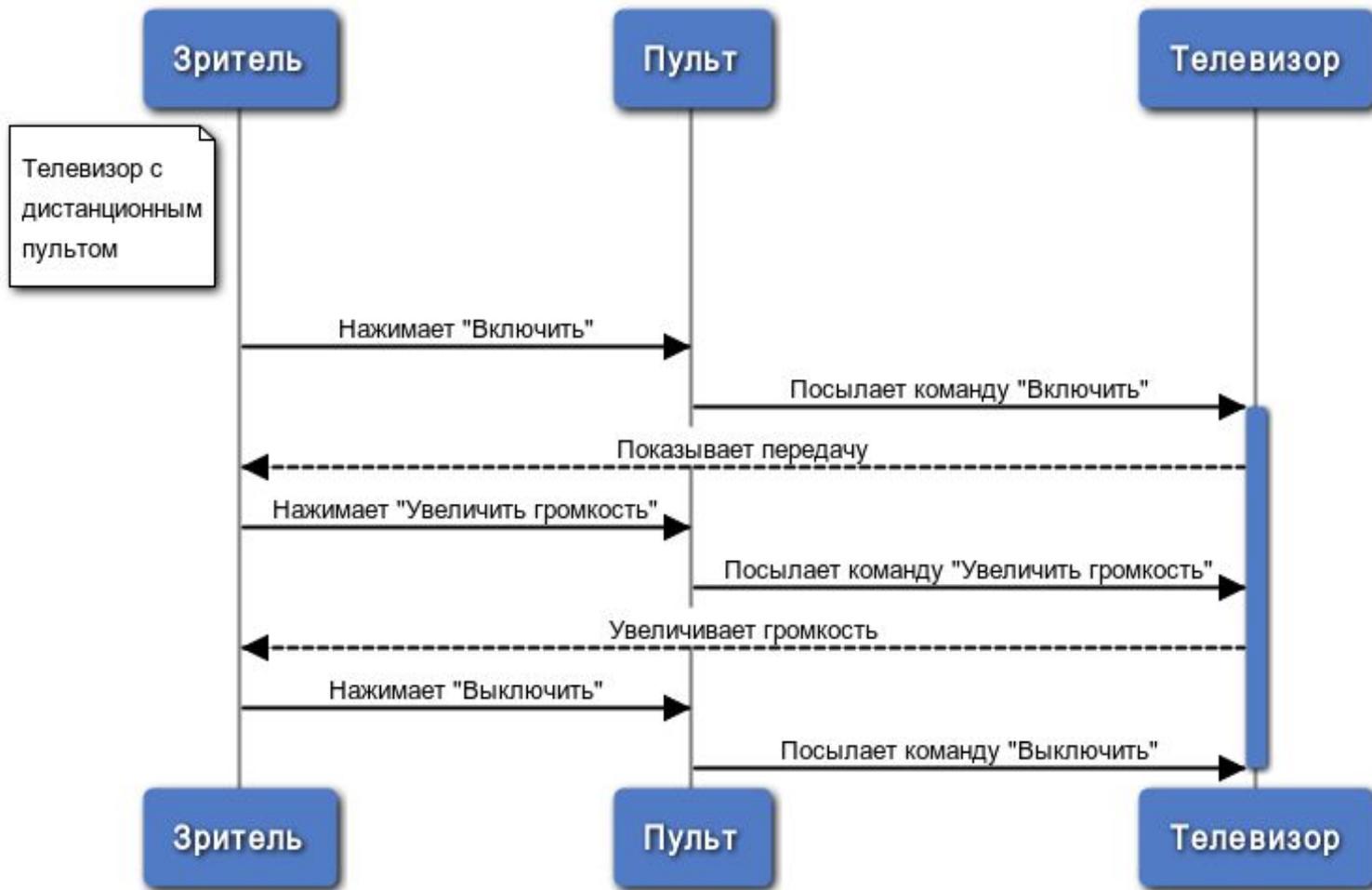


Диаграмма коммуникаций

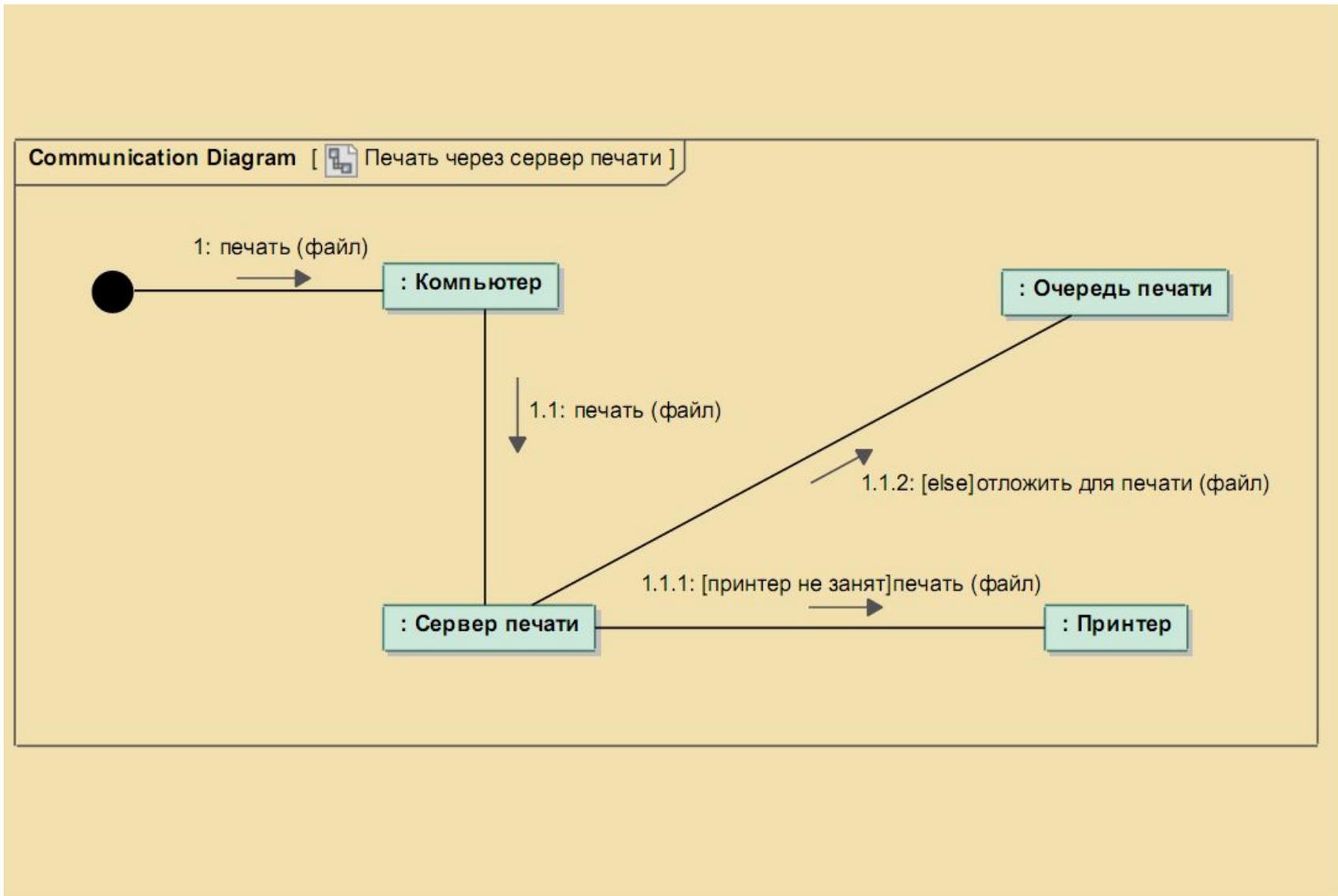


Диаграмма состояний

Диаграммы состояний показывают различные состояния объекта в течение его времени жизни

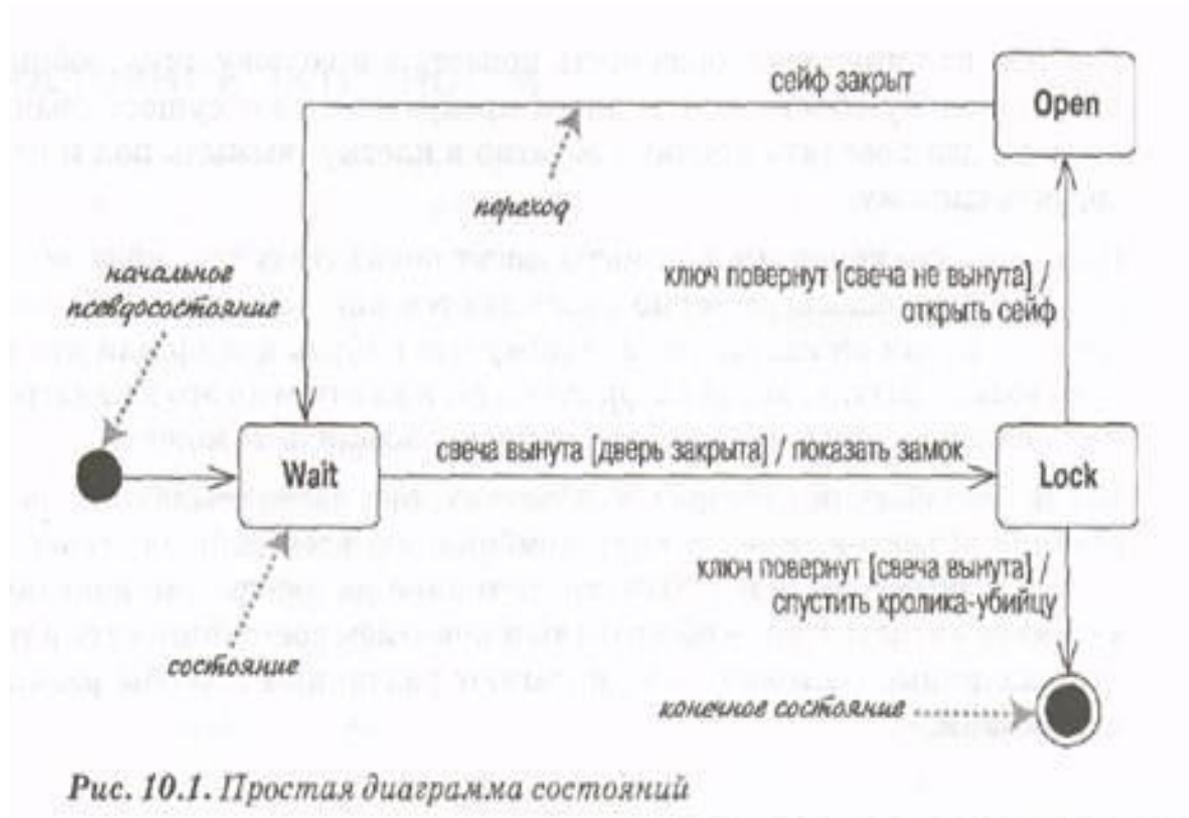


Рис. 10.1. Простая диаграмма состояний

Синхронизирующие состояния

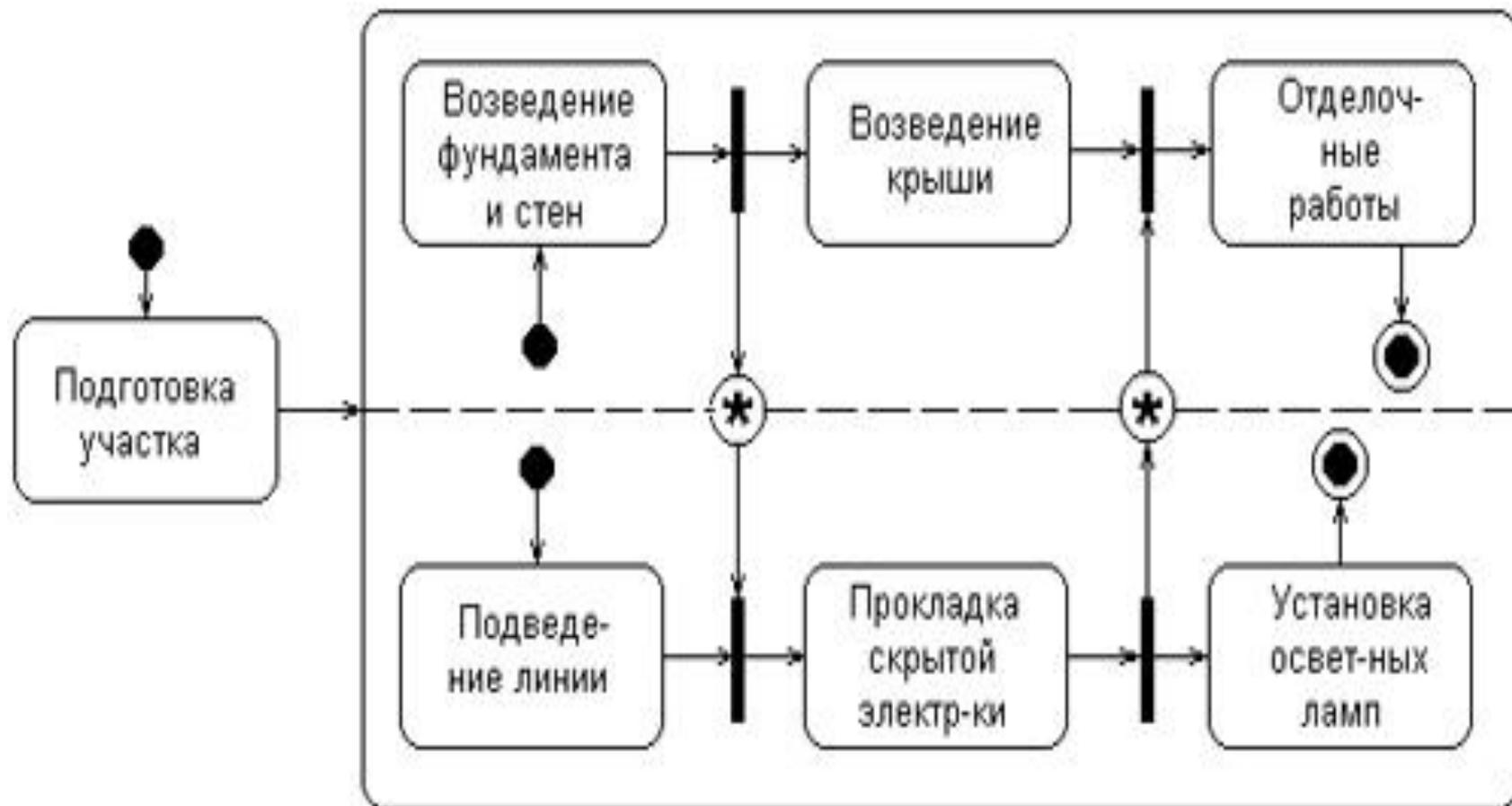
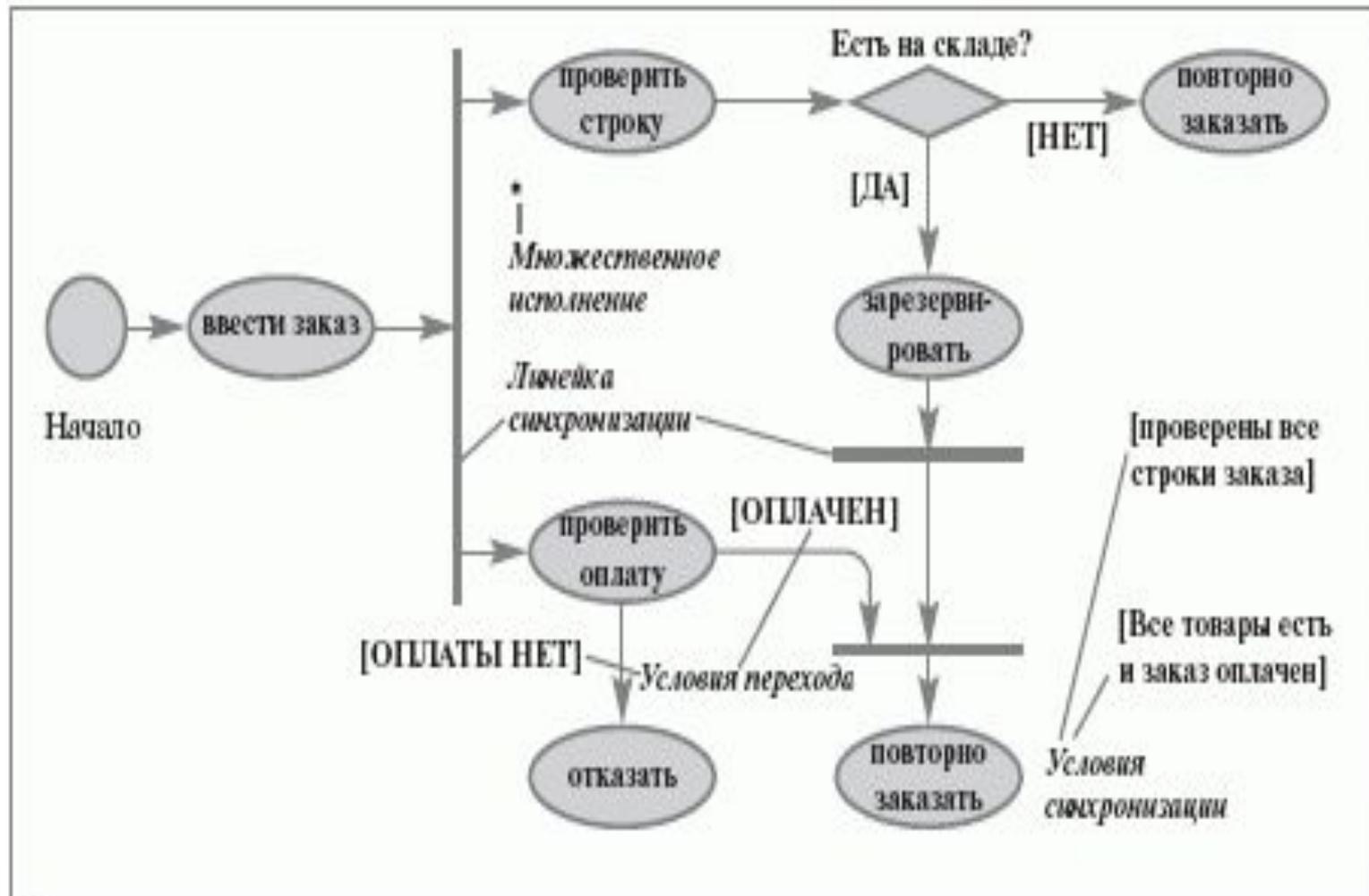


Диаграмма деятельности



Пример диаграммы деятельности

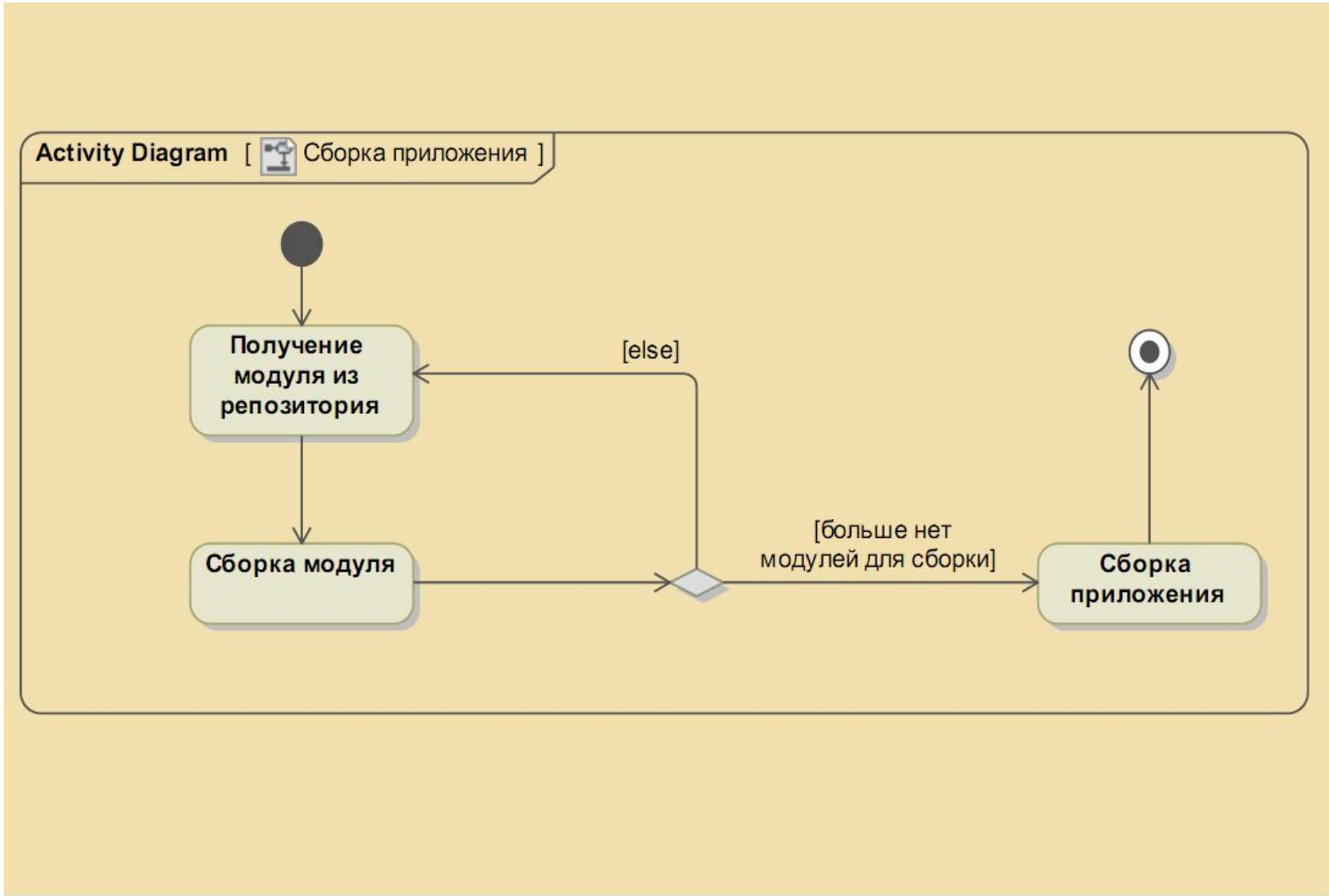


Диаграмма компонентов

Диаграмма компонентов описывает особенности физического представления системы.



Диаграмма внутренней структуры

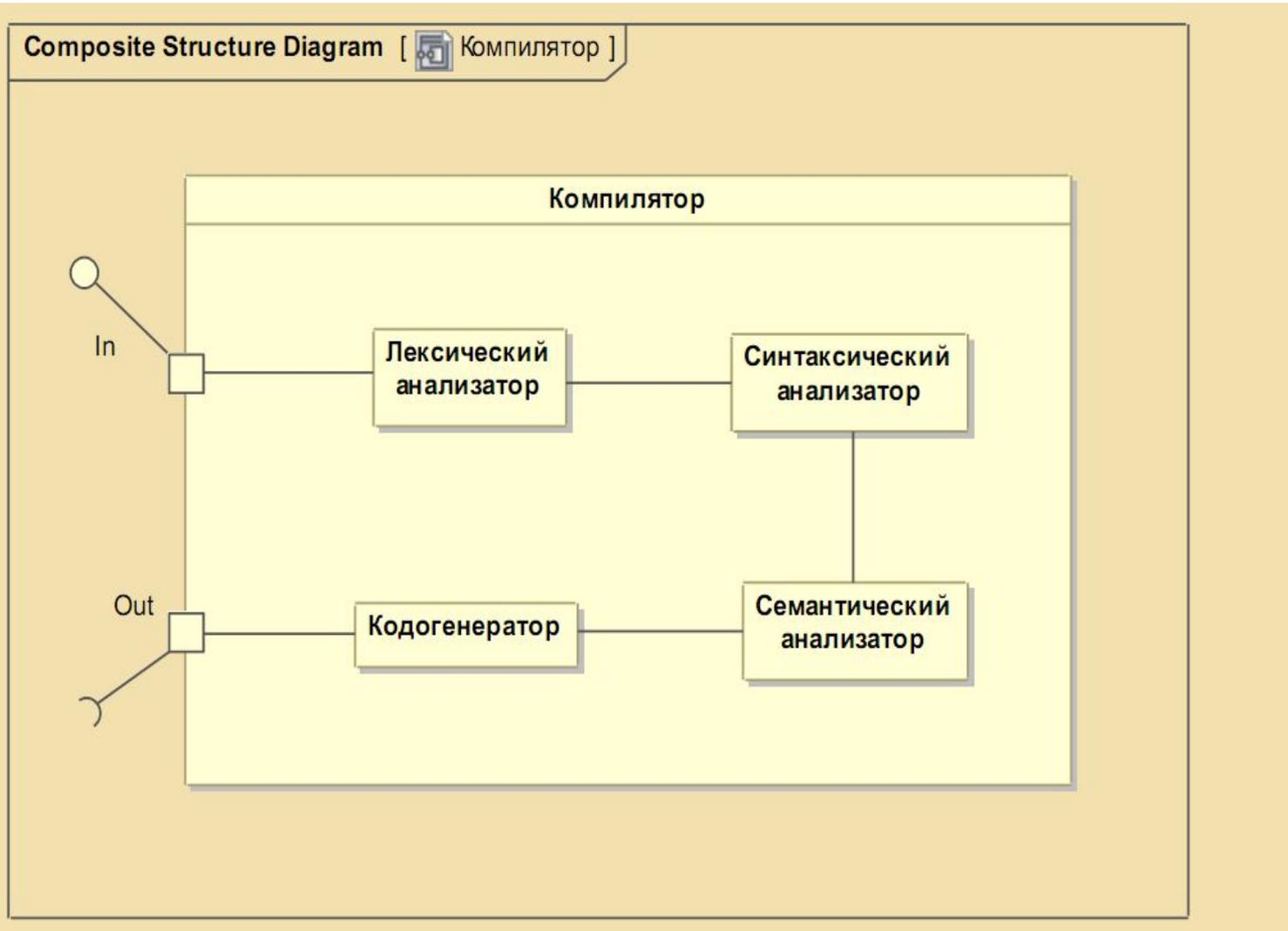


Диаграмма размещения

Диаграмма размещения наряду с отображением состава и связей элементов системы показывает, как они физически размещены на вычислительных ресурсах во время выполнения.

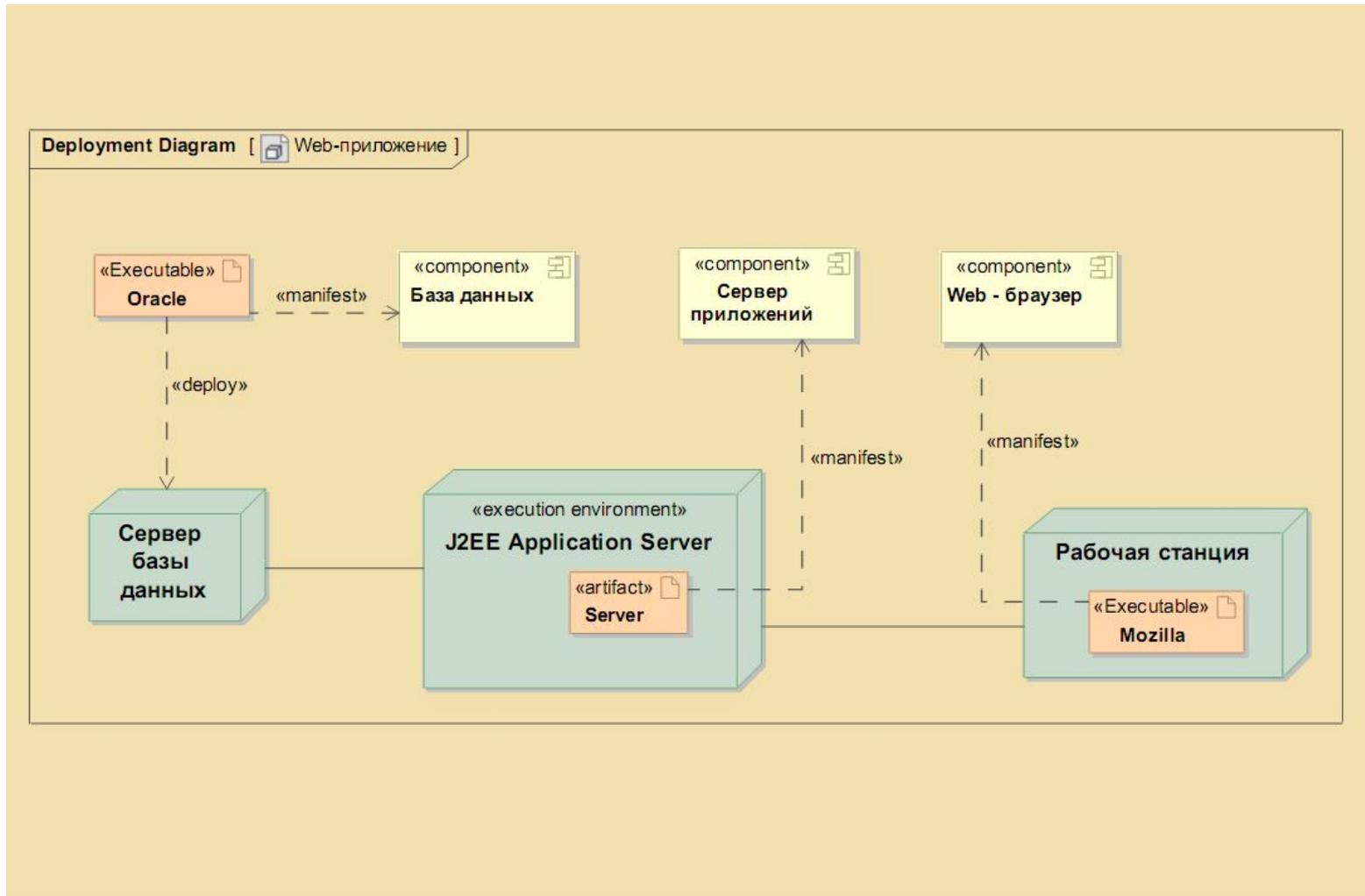
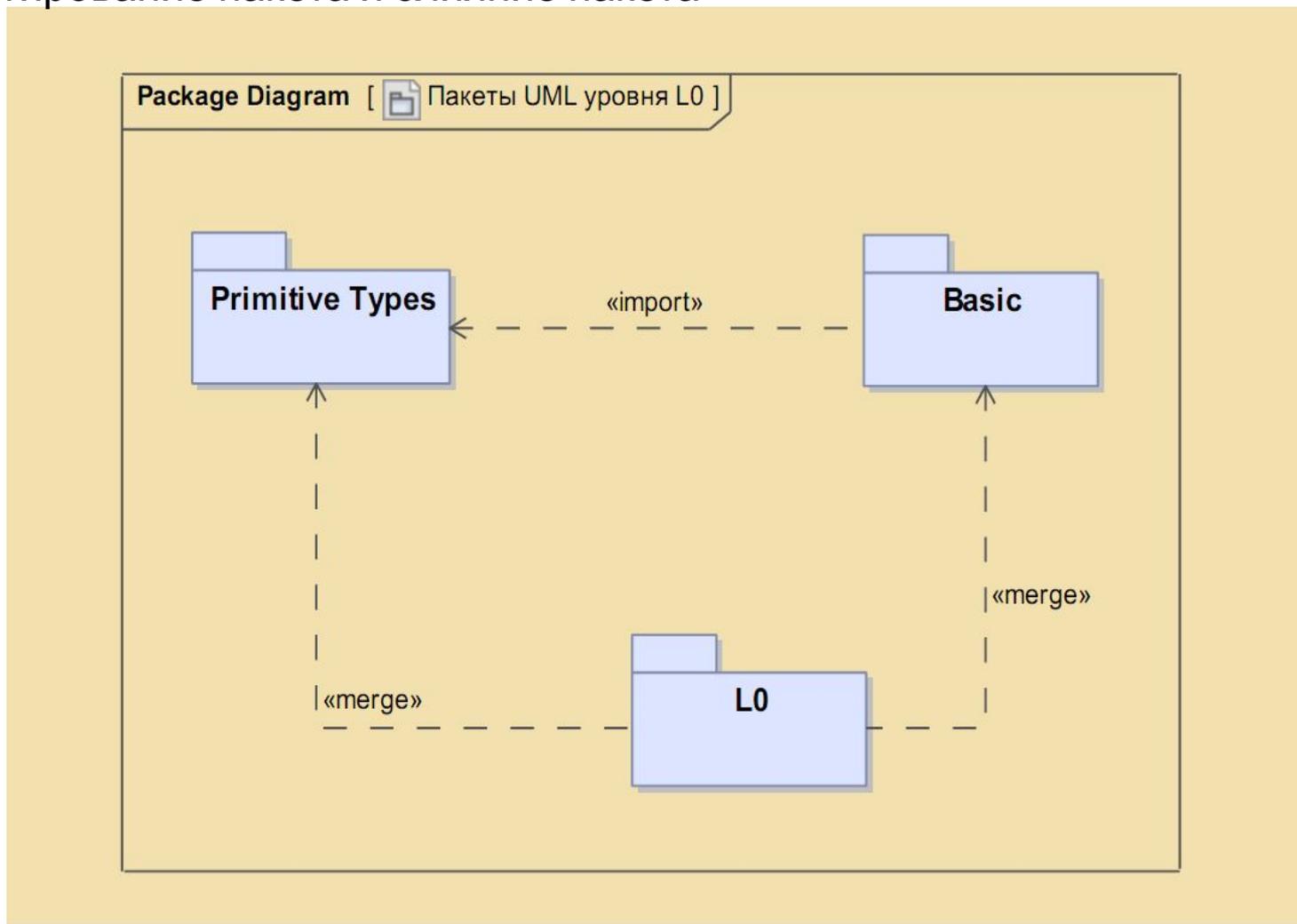


Диаграмма пакетов

Диаграммы пакетов отображают зависимости между пакетами: импорт пакета и слияние пакета



```
public class HelloWorld
{
    public static void main(String[] args)
    {
        System.out.println("Hello, world!");
    }
}
```

HelloWorld.java

Приложение Hello, World

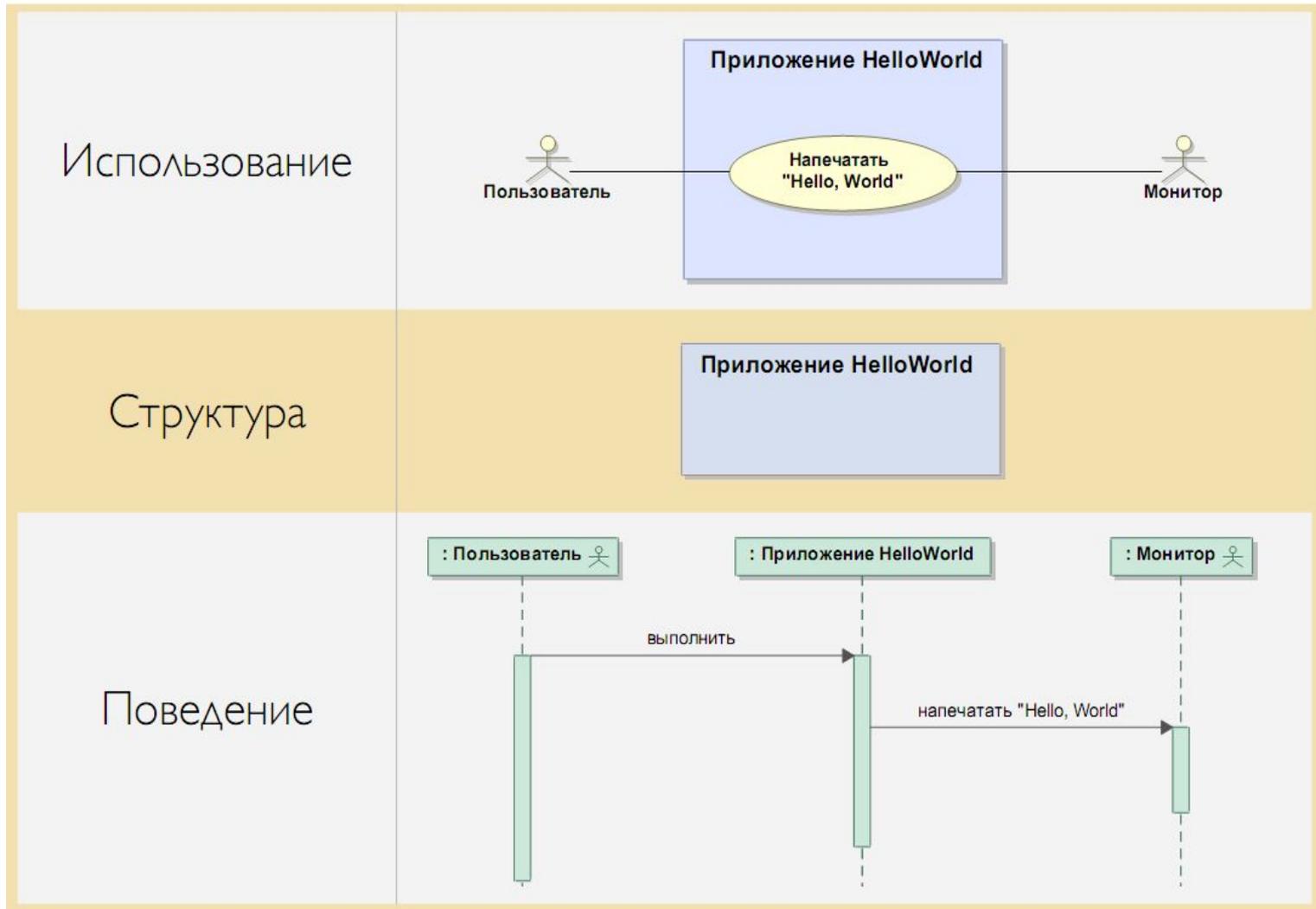
```
> javac ./HelloWorld.java
> java HelloWorld
```

HelloWorld.java

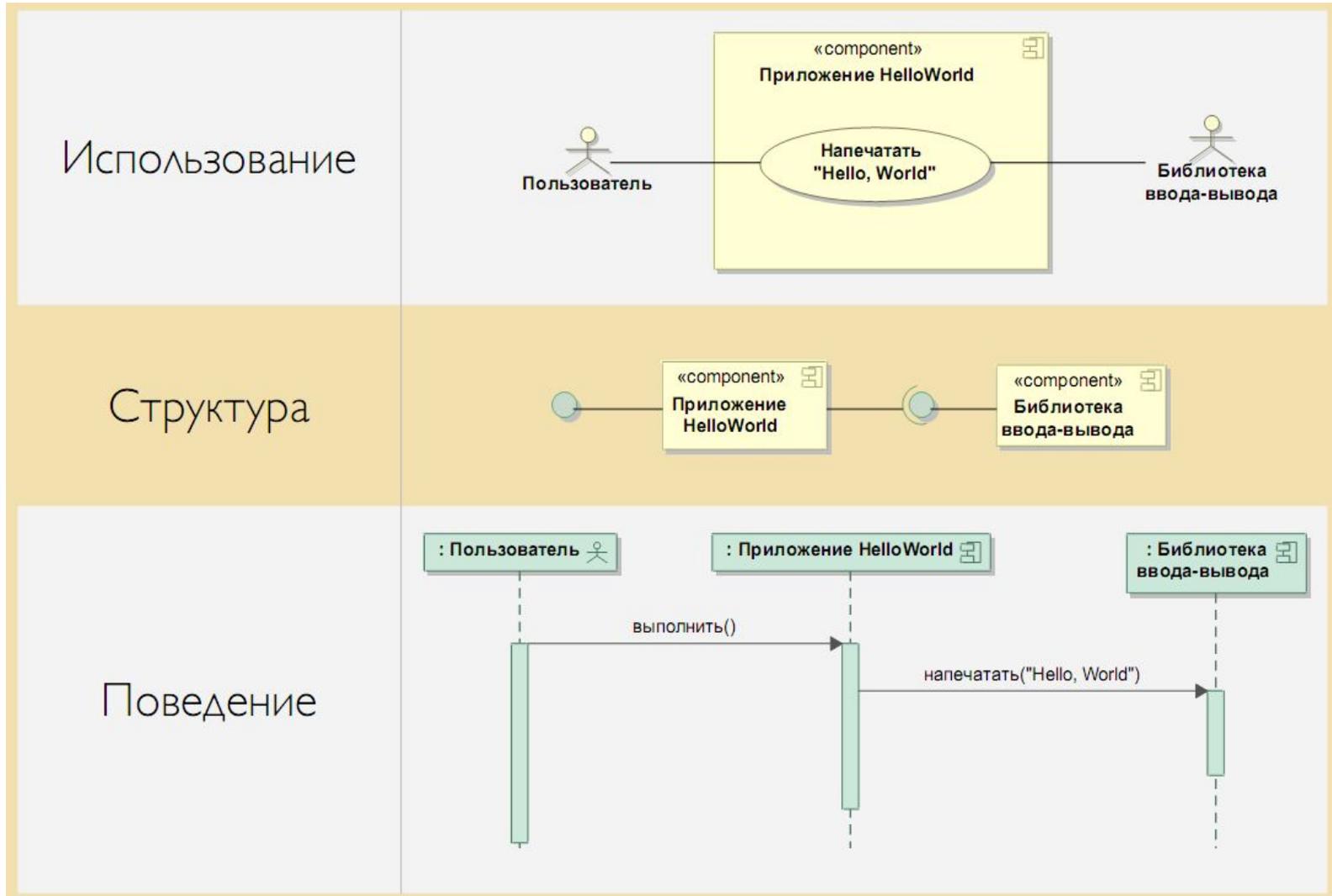


HelloWorld.class

Уровень системы

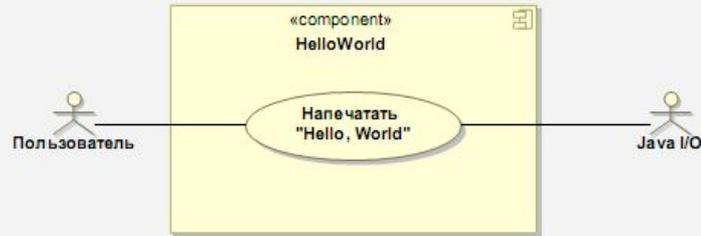


Уровень модуля

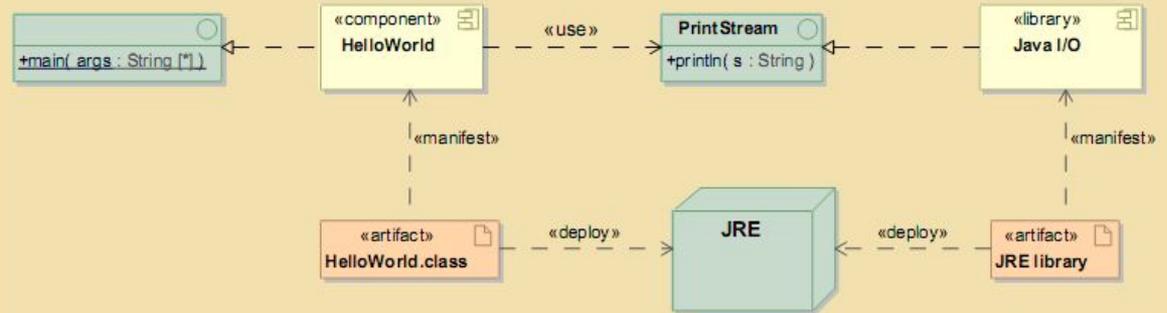


Уровень модулей Java

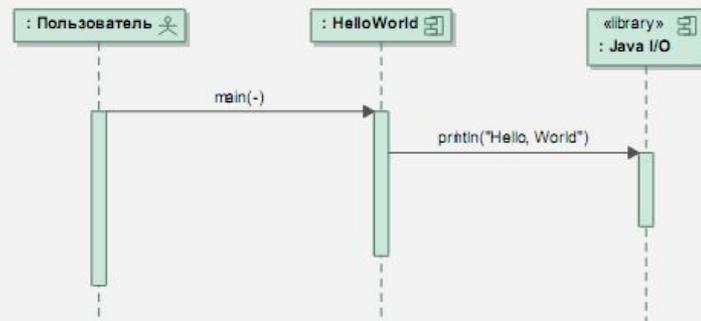
Использование



Структура

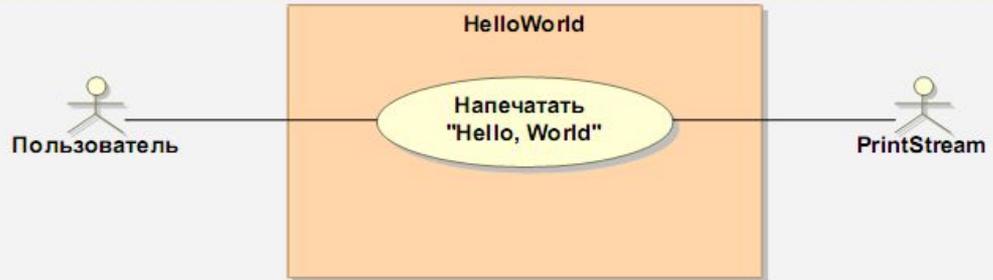


Поведение

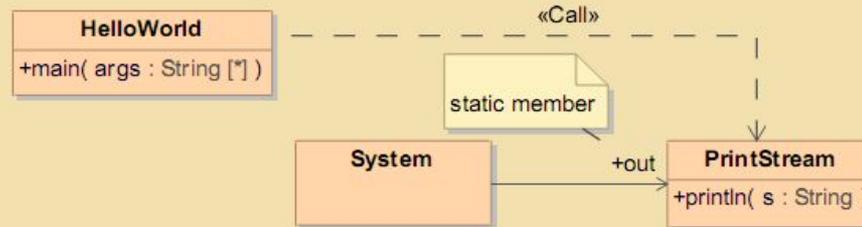


Уровень объектов Java

Использование



Структура



Поведение

