

# Моделирование на UML. Вторая ступень

---

## Введение в UML



Иванов Д.Ю., Новиков Ф.А.



# Структура курса

---

Об этом курсе

✓ **Часть 1. Введение в UML**

Часть 2. Моделирование использования

Часть 3. Моделирование структуры

Часть 4. Моделирование поведения

Часть 5. Дисциплина моделирования

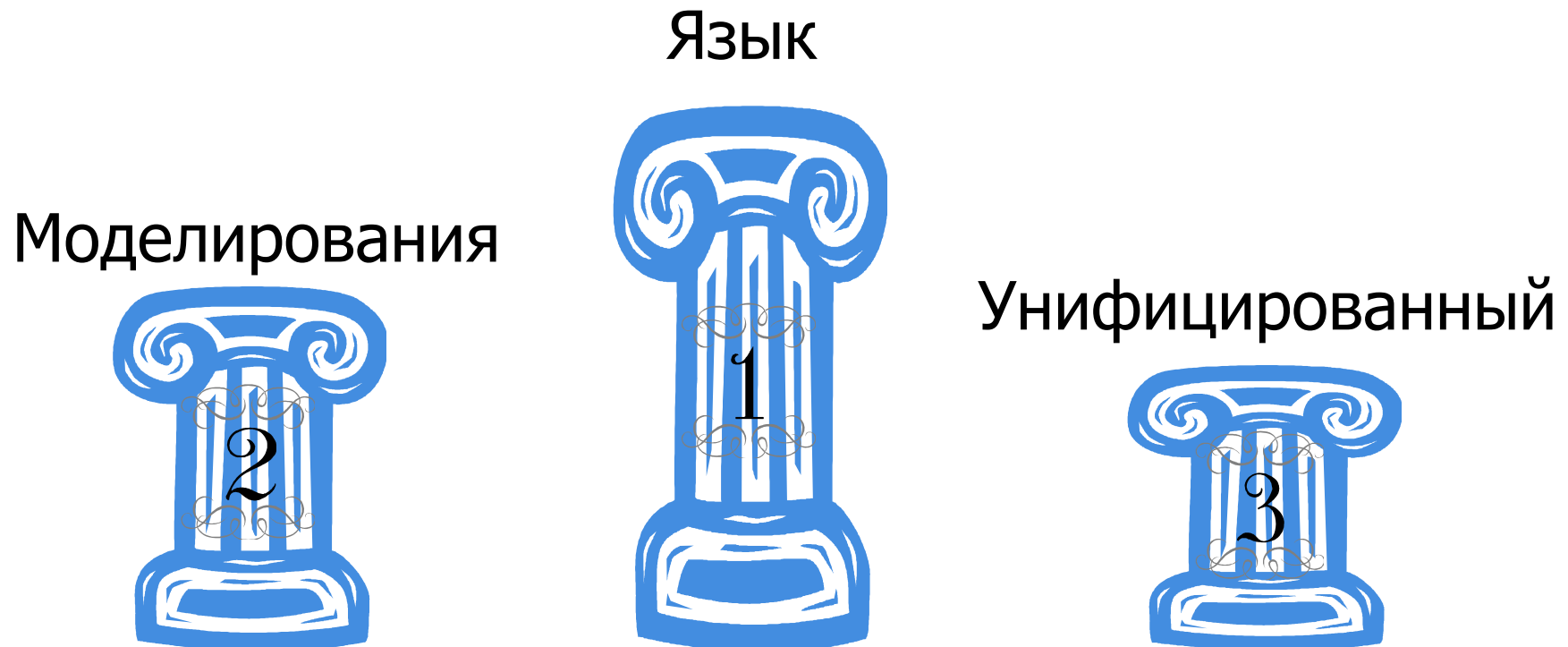
# Содержание

---

1. Что такое UML?
2. Назначение UML
3. Определение UML
4. Модель и ее элементы
5. Диаграммы
6. Модели и представления
7. Общие механизмы
8. Выводы

# 1. Что такое UML?

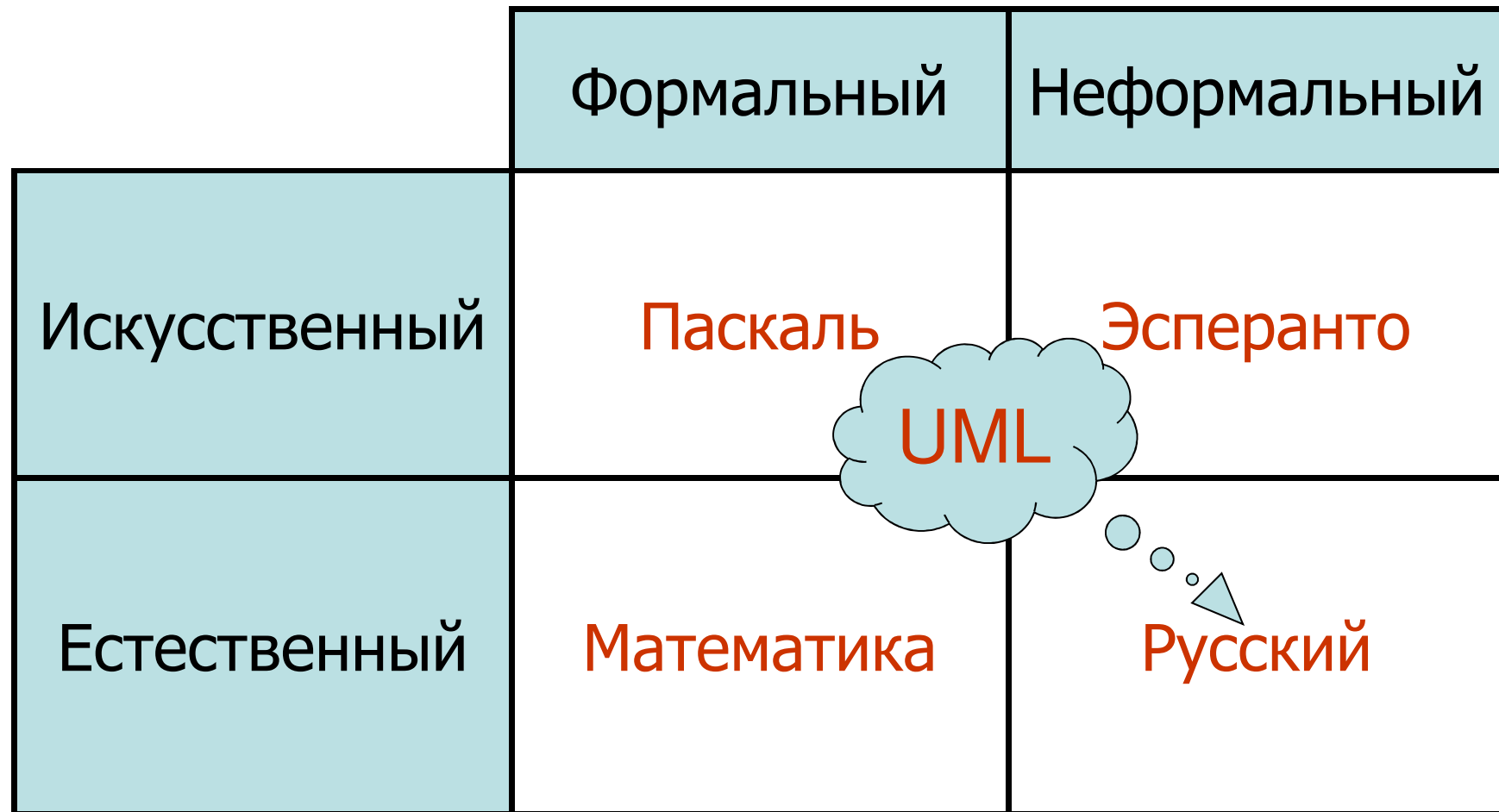
---



UML = Unified Modeling Language

# UML – ЭТО ЯЗЫК

---



# Авторы UML: Буч

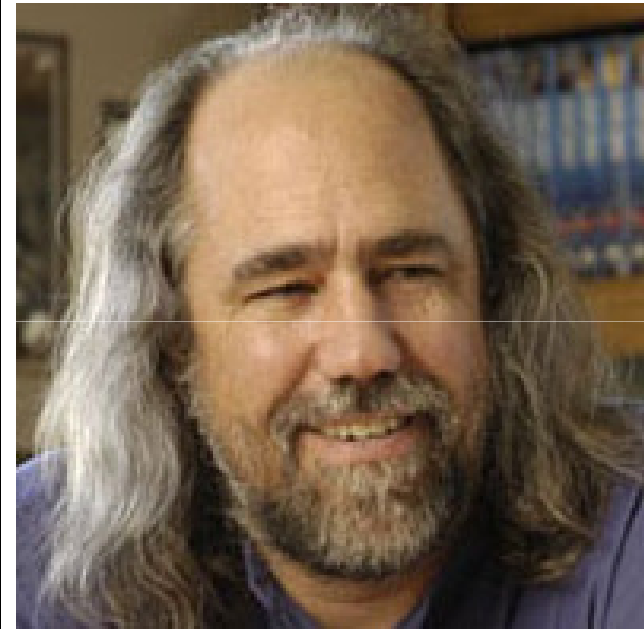
---

Гради Буч (Grady Booch), р. 1955 — американский ученый и инженер

- Степень бакалавра получил в 1977 в Военно-воздушной академии США, а степень магистра — в 1979 в Калифорнийском университете, Санта Барбара

- Работал главным научным сотрудником корпорации Rational Software, которая была куплена корпорацией IBM в 2003 году

- В настоящее время является главным научным сотрудником IBM Research



# Авторы UML: Якобсон

Ивар Якобсон (Ivar Hjalmar Jacobson),

р.1939 — шведский ученый и инженер

- Степень магистра получил в 1962 году, а степень доктора в 1985 году в Королевском технологическом институте в Стокгольме
- Идея сборочного программирования на основе программных компонентов (1967)
- Изобретение диаграмм последовательности (1985) и вариантов использования (1986)
- Соавторство при разработке языков SDL (1976) и UML (1997)



# Авторы UML: Рамбо

Джеймс Рамбо (James Rumbaugh) — американский инженер и ученый

- Бакалаврская степень по физике в Массачусетском технологическом институте, магистерская степень по астрономии в Калтехе, докторская степень по информатике в Массачусетском технологическом институте

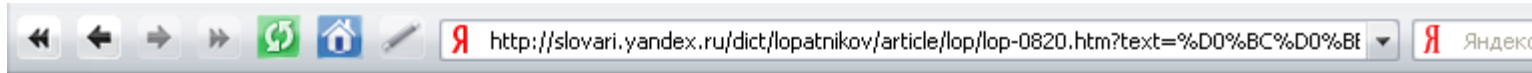
- Более 25 лет проработал в исследовательском центре корпорации General Electric

- В 1994 перешел в Rational Software, где вместе с Бучем и Якосоном разработал UML





# UML – ЭТО ЯЗЫК моделирования

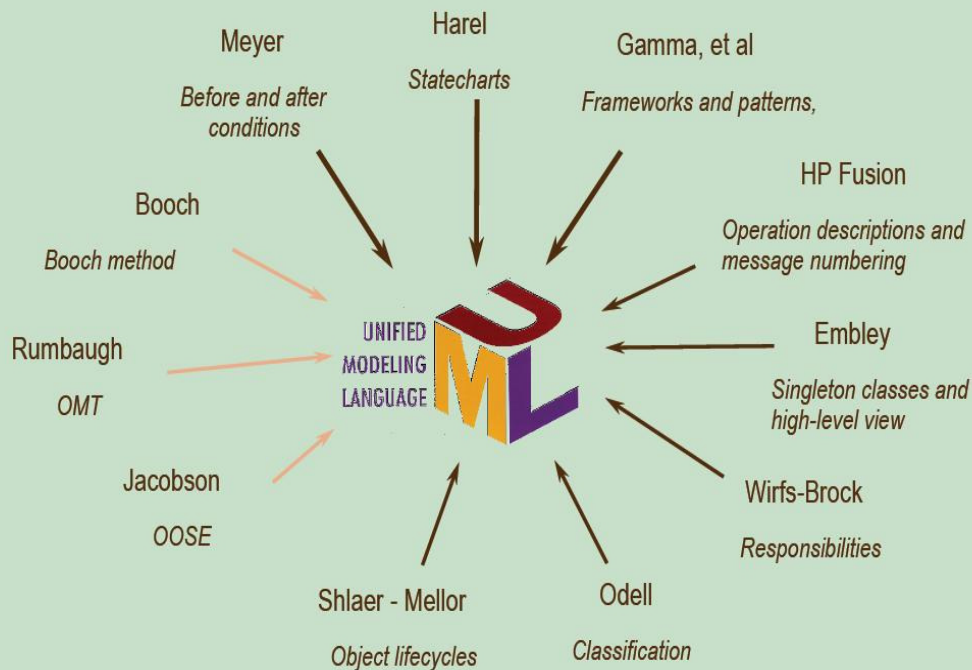


**МОДЕЛИРОВАНИЕ** [modelling, model-building] — 1. Исследование объектов познания на моделях. 2. Построение и изучение моделей реально существующих предметов и явлений, а также предполагаемых (конструируемых или проектируемых) объектов.

- Модель – основной артефакт фазы проектирования
- Деятельность – моделирование
  - Составление и использование моделей
- Должность – (системный) архитектор

# UML – это унифицированный язык моделирования

In 1994, more than 50 OO methods!

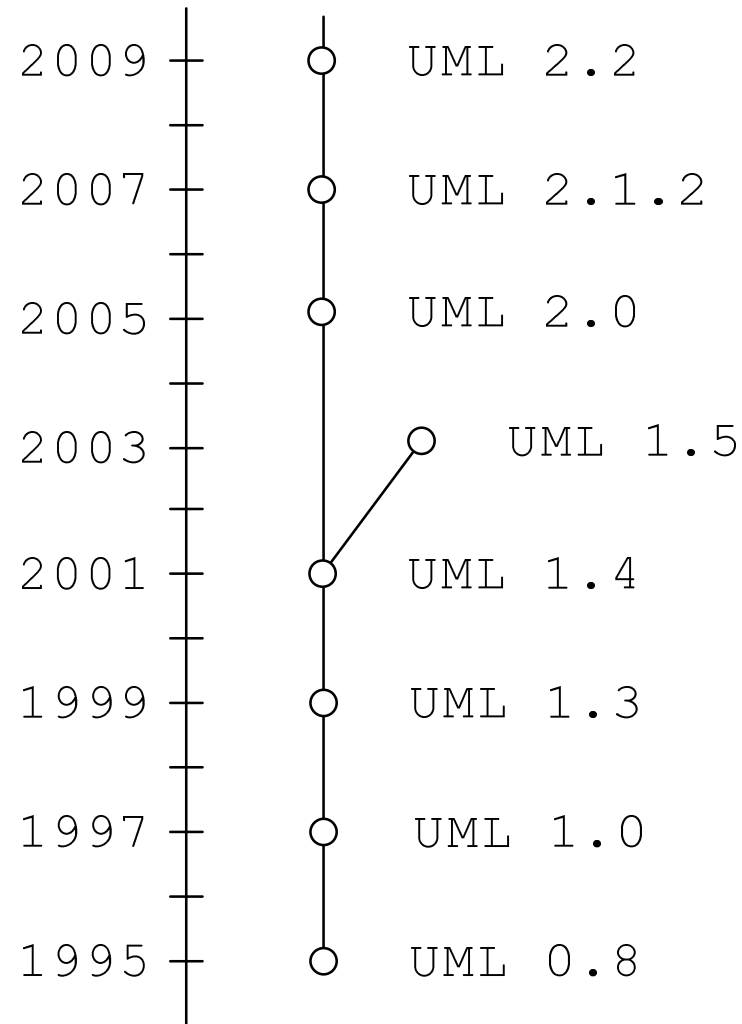


- ...
- Петроглифы
- Блок-схемы
- Р-технология
- Диаграмма потоков данных (DFD)
- Диаграмма "сущность-связь" (ERD)
- Методология структурного анализа и проектирования (SADT)
- ...

• ... • SDL • MSC • ...

# История развития UML

---



## 2. Назначение UML

---

“UML – графический язык **моделирования** общего назначения, предназначенный для **спецификации, визуализации, проектирования и документирования** всех артефактов, создаваемых при разработке программных систем”

Г. Буч

# Моделирование =

---

**Спецификация** (specification)

- формальная, но наглядная

+ **Визуализация** (visualization)

- для общения

+ **Проектирование** (construction)

- конструирование?

+ **Документирование** (documenting)

- всех артефактов

# Спецификация

---

- **Спецификация** = описание (программы)
  - в понимании заказчика  $\neq$
  - в понимании разработчика  $\neq$
  - на самом деле
- UML –
  - (полу) формальное
  - (иногда) удобное
  - (почти) универсальное

**средство** для уменьшения расхождений в  
толковании спецификаций

# Как это бывает



Так объяснил  
заказчик



Так понял менеджер  
проекта



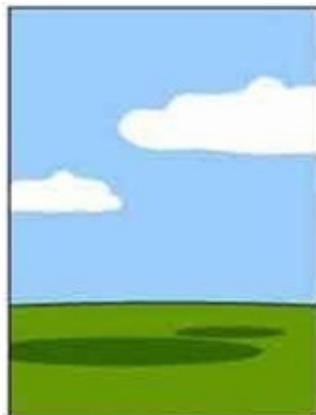
Так описал аналитик



Так реализовал  
программист



Так презентовал  
проект менеджер



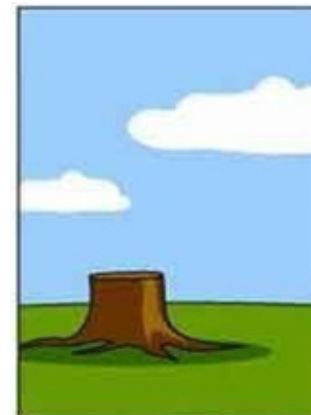
Такой оказалась  
документация



Таким оказался  
продукт



Такими оказались  
затраты



Такой оказалась  
работоспособность



Что реально хотел  
заказчик ...

# Формальная спецификация

- Полная формальная математическая спецификация
  - Невозможна ☹ ? Возможна ☺ ! Но ...
- Объем математической спецификации больше объема кода (в несколько раз)
- Математики дороже программистов
- Математические теории разрабатываются медленнее, чем программы
- Наилучшая спецификация программы – код ☺
- Формализация дороже автоматизации



# Пример формальной спецификации

---

- **Спецификация:** «программа вычисления двух вещественных корней квадратного уравнения»
- **Полная (?)** формальная математическая спецификация:
  - Предусловие:  $a \neq 0 \ \& \ b^2 - 4ac > 0$
  - Постусловие:  $ax^2 + bx + c = 0$
- **Пропущено** «вещественных»
- Тип **bool** = {F,T} + таблицы истинности
- Тип **int** = коммутативное кольцо с единицей вычетов по модулю  $2^{32}$
- Тип **double** = ? ? ?

# Визуализация

## Текст

Каждый день до нас доносится эгоголизм маркетинговых войн между брендами с экранов телевизора, с динамиков радио, со страниц журналов и газет, рекламных щитов... Парадоксально, но в сегодняшнем мире стоимость бренда (нематериального актива) может значительно превосходить стоимость всех материальных активов компании. Судите сами, по данным на июль 2003 года суммарная стоимость самых дорогих брендов мира составляет почти 190 миллиардов долларов (!):

1. Coca-Cola – \$70 млрд.
2. Microsoft – \$65 млрд.
3. IBM – \$51 млрд.

Эти данные приведены на основании рейтинга, ежегодно публикуемых "BusinessWeek" и компанией "Interbrand". Всего по данным этого рейтинга 100 самых дорогих брендов мира суммарно стоят почти 1 трлн. долларов (\$973,98 млрд.)! Впечатляющие цифры, не правда ли?

Ну и что, скажете вы. А что ценного для себя, я могу извлечь из этой информации? Где здесь моя выгода?

Не спешите перелистывать страницу. Очень скоро я вам сообщу очень важную информацию, имеющую непосредственное отношение к вам. Но сначала небольшое отступление.

Еще недавно, каких-то 10 лет назад, мы жили совершенно в другом мире. В мире, где одним из составляющих жизненного успеха было понятие "стабильности". Мы могли рассчитывать на получение хорошего образования, устройство на работу, где у нас всегда была возможность неторопливой карьеры и стабильного повышения заработной платы в течение всего трудового стажа в 40 лет. У нас были гарантии, льготы, привилегии, и... долгожданная пенсия. Так жили наши родители, так жили и верили в такую жизнь многие из читающих эти строки.

Но наступило новое тысячелетие. Тысячелетие невероятных, стремительных перемен. Внезапно мы обнаруживаем, что все то, что еще недавно нам казалось неизменным и стабильным, сегодня уже не существует. Образование уже не гарантирует получение высокооплачиваемой работы. Да и работа на одном предприятии в течение всей жизни – это уже, скорее, из области мифов, чем реальности: сегодня ты нужен, а завтра, в любой момент, тебя могут сократить.

Да, мир изменился и продолжает стремительно меняться. Вчерашние ценности сегодня уже не существуют. На их место приходит новое. И ключевым качеством современного человека является способность к изменениям. Способность быстро меняться и приспосабливаться к постоянно изменяющемуся миру. А можно ли сегодня рассчитывать на стабильность? Уверен, что нет. Во всяком случае, в "старом" понимании значения этого слова.

## Текст



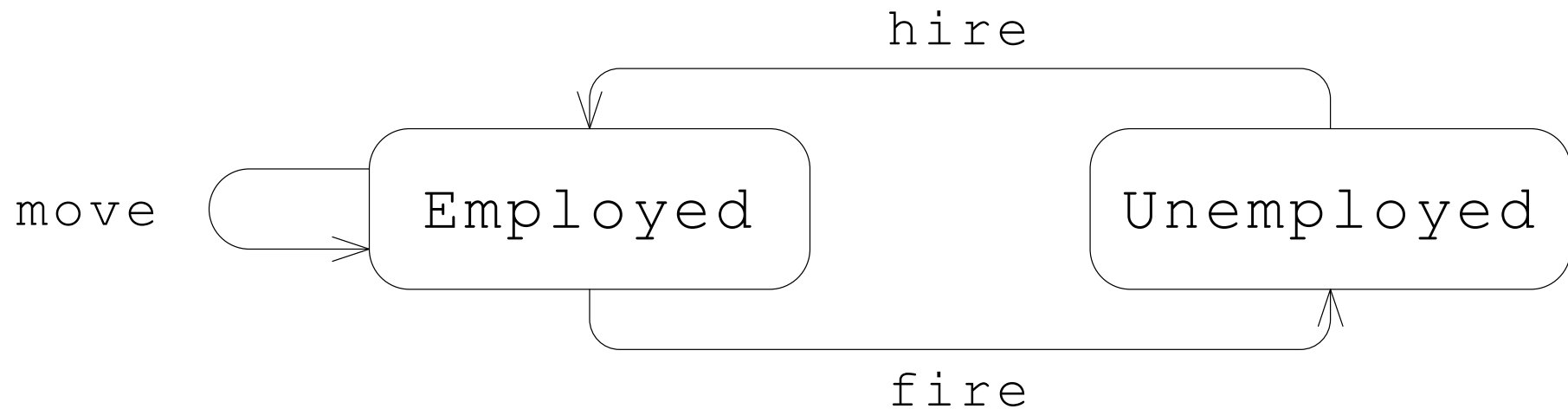
## КОМИКС



## с картинками

# Жизненный цикл работника на предприятии

---



# Проектирование

---

- Автоматический синтез программ
  - Алгоритмически неразрешимая массовая проблема
  - Известны разрешимые подклассы
- Автоматическая (частичная) генерация программного кода по модели
  - Генеративное и
  - ☺ дегенеративное программирование (непосредственная интерпретация моделей)
- Автоматическое построение моделей по коду готового приложения
  - Инженерный анализ программ (Reverse engineering)

# Документирование

---

- Все элементы модели могут содержать текстовое описание
- Почти все инструменты могут собирать из них осмысленные документы
- Почти никто из разработчиков этим не пользуется

# Чем *не* является UML?

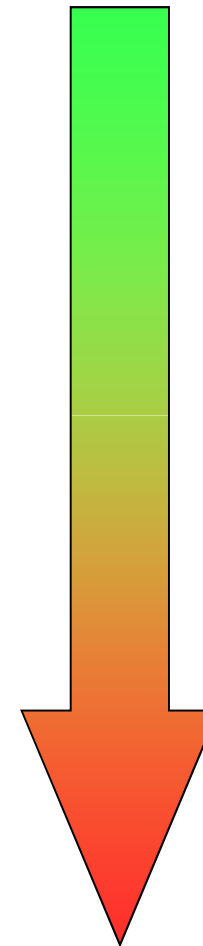
---

1. Языком программирования
  - Генерация кода возможна
2. Спецификацией инструмента
  - Инструменты подразумеваются и имеются
3. Моделью процесса разработки приложений
  - Модель необходима и имеется – Rational Unified Process (RUP)

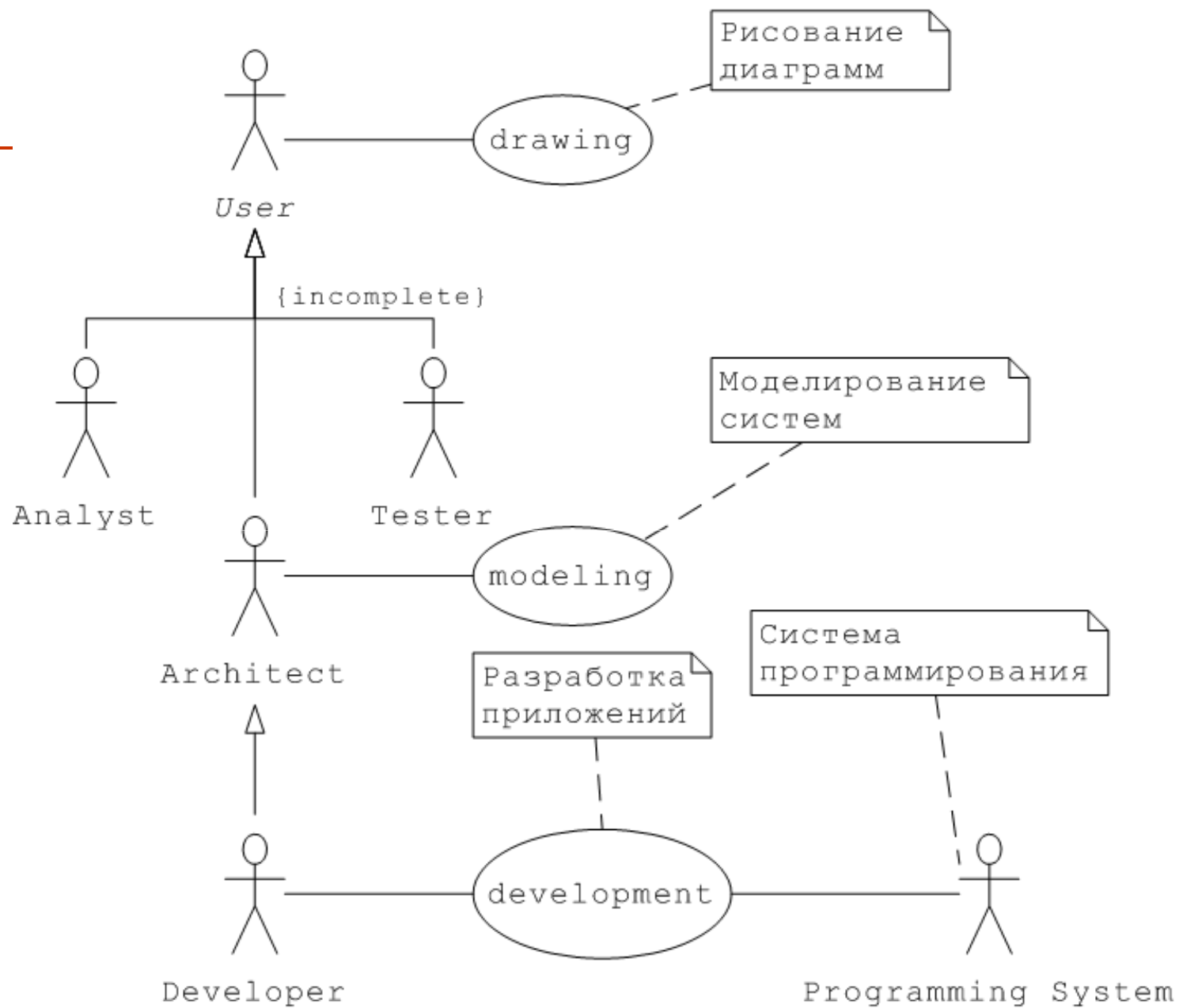
# Способы использования UML

---

- Рисование картинок
- Обмен информацией
- Спецификация систем
- Повторное использование архитектурных решений
- Генерация кода
- Имитационное моделирование
- Верификация моделей



# Варианты использования UML





# 3. Определение UML

---

- Точный  $\leftrightarrow$  понятный, краткий  $\leftrightarrow$  полный: компромисс
- Определение методом **раскрутки**
  - Семантика: диаграммы классов/пакетов + ограничения (**OCL**) + текст (**Plain** English)
  - Нотация: отображение семантики в картинку (а не наоборот!)

**OCL = Object Constraints Language**

# Языковые уровни

---

- Мета-метамодель – описание используемого формализма
  - Контекстно-свободная грамматика
  - MOF = Meta Object Facility
- Метамодель – описание языка
  - Infrastructure + Superstructure
- Модель – использование языка
  - Примеры

# Структура стандарта UML 2.2

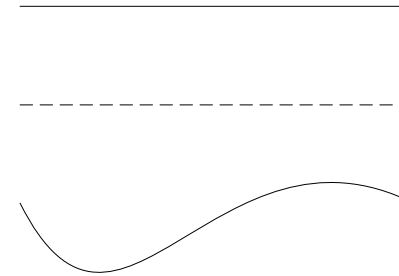
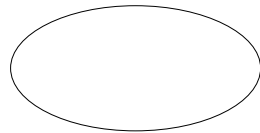
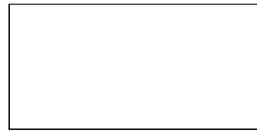
Документ	Содержание	Стр.
UML Infrastructure	Описание базовых механизмов	226
UML Superstructure	Описание самого языка	740
UML Diagram interchange	Описание формата обмена дополнительными данными	86
OCL Specifications	Описание объектного языка ограничений	232
		1284

# Терминология

---

1. UML – **независимый** от конкретных языков программирования  
→ новые термины для исключения совпадений
  2. UML – **унифицированный**  
→ разные терминологические традиции
  3. **По-русски**: устоявшейся терминологии нет  
→ основной критерий: как можно точнее передать смысл
- Терминология UML довольно замысловатая и не всегда последовательная

# Нотация

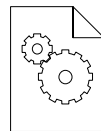
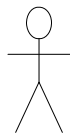


Фигуры

Линии

Значки

Тексты



Class

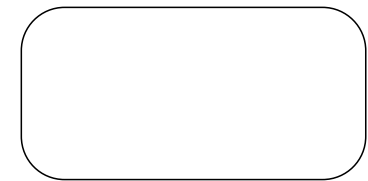
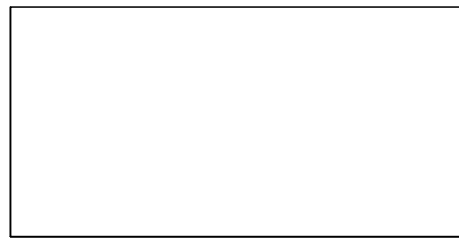
Object

*Interface*

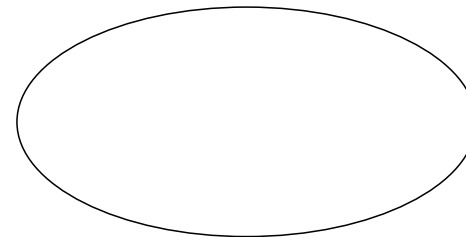
# Фигуры

---

- 2D
- Замкнутые
- Контейнер



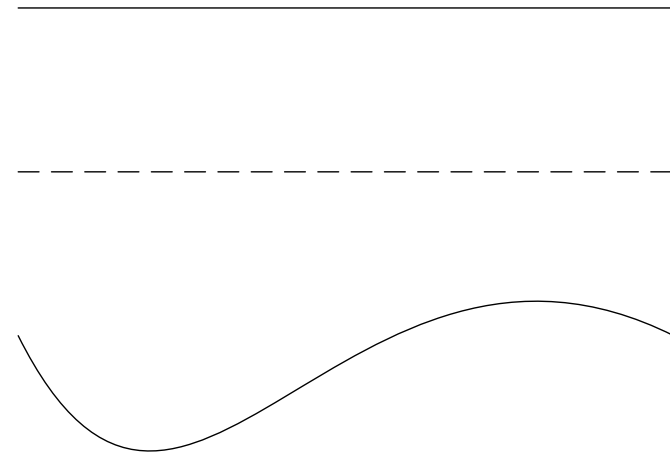
В UML 2 – **рамки** =  
фигура, которая всегда  
**контейнер** для других фигур и  
имеет **ярлычок**



# Линии

---

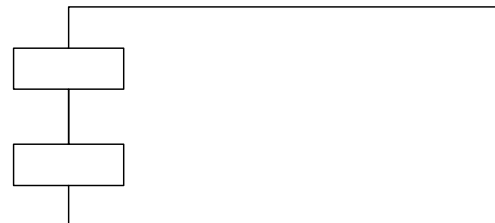
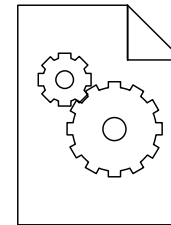
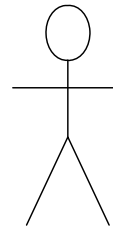
- Одномерные
- Точки присоединения
- Прямые, ломанные, плавные кривые
- Толщина
- Пересечение
- **Стиль**
  - сплошная
  - пунктирная
- **Дополнительные элементы**
  - текст
  - украшения (стрелки)



# Значки

---

- Не контейнер
- Дополнительный элемент
  - текст





# Тексты

---

- Смысл
- Гарнитура, размер, цвет
- Начертание
  - прямое
  - подчеркнутое
  - *курсив*

Class

Object

*Interface*

# Литература



Фаулер М.  
UML. Основы.  
3-е издание. Символ-Плюс, 2005



Буч Г., Рамбо Д., Якобсон А.  
Язык UML. Руководство пользователя.  
2-е издание. ДМК, 2006



Буч Г., Якобсон А., Рамбо Д.  
UML. 2-е издание.  
Классика CS. Питер, 2006

**Иванов Д., Новиков Ф.  
Моделирование на UML.  
НиТ, 2010**

## 4. Модель и ее элементы

---

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

Сущности и отношения модели являются  
экземплярами **метаклассов**  
**метамодели**

# Модель UML

---

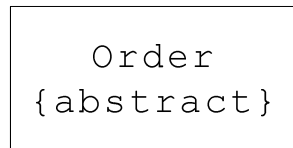
Нагруженный мульти-псевдо-гипер-орграф

- Вершины и ребра **нагружены** информацией и могут иметь сложную внутреннюю структуру
- Вершины - **сущности**,  
ребра - **отношения**

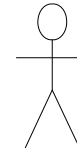
# Сущности (i)

Тип сущности	Название	Перевод
<b>Структурные</b>	артефакт	artifact
	вариант использования	use case
	действующее лицо	actor
	интерфейс	interface
	класс	class
	компонент	component
	кооперация	collaboration
	объект	object
	узел	node

# Нотация сущностей (i)



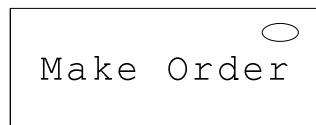
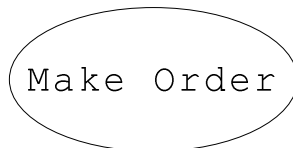
**класс**



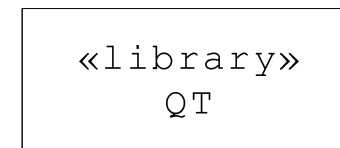
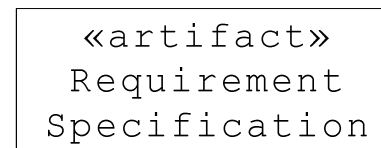
Customer



**действующее лицо**



**вариант использования**



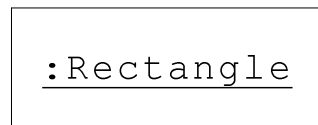
**артефакт**



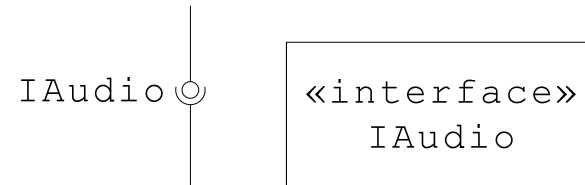
**КОМПОНЕНТ**

# Нотация сущностей (ii)

---



**объект**



**интерфейс**



**узел**



**кооперация**

# Сущности (ii)

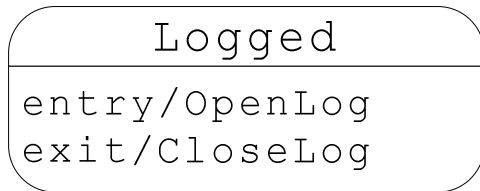
---

Тип сущности	Название	Перевод
<b>Поведенческие</b>	действие	action
	деятельность	activity
	состояние	state
<b>Группирующие</b>	пакет	package
<b>Аннотационные</b>	примечание	note

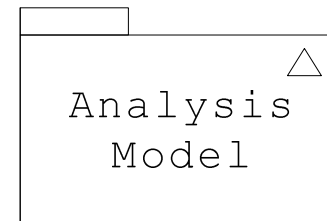


# Нотация сущностей (iii)

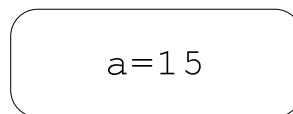
---



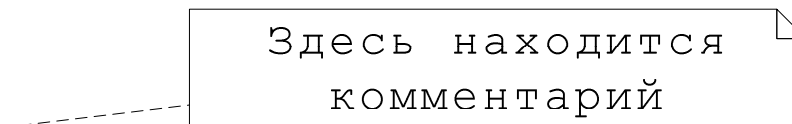
**состояние**



**пакет**



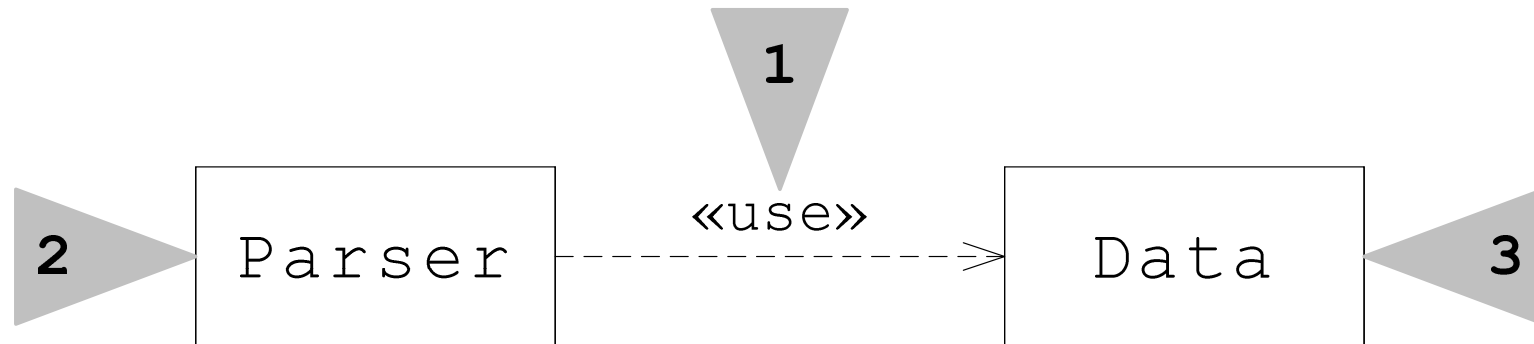
**деятельность и действие**



**примечание**

# Отношение зависимости

- **Зависимость**: независимая сущность *каким-то образом* влияет на зависимую
- Смысл часто уточняется стереотипом: « »

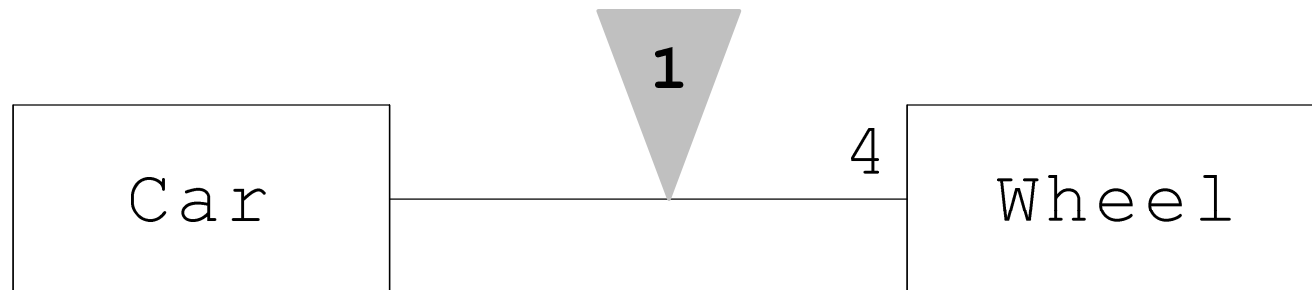


- 1) отношение зависимость
- 2) зависимая сущность
- 3) независимая сущность

# Отношение ассоциации

---

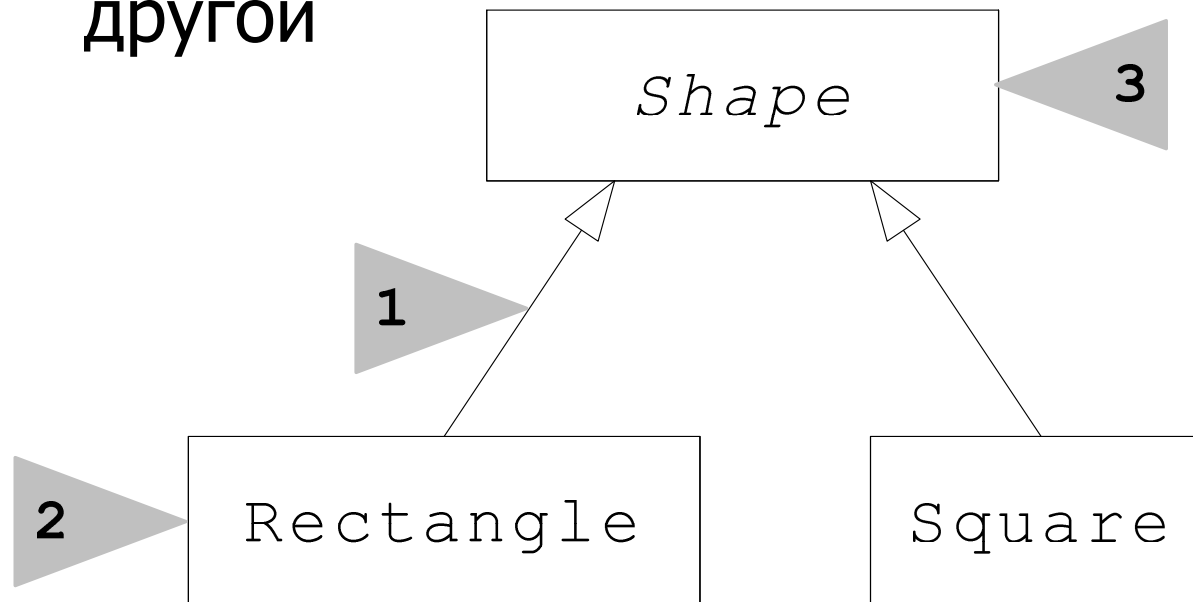
- **Ассоциации:** одна сущность непосредственно связана с другой



1) отношение ассоциация

# Отношение обобщения

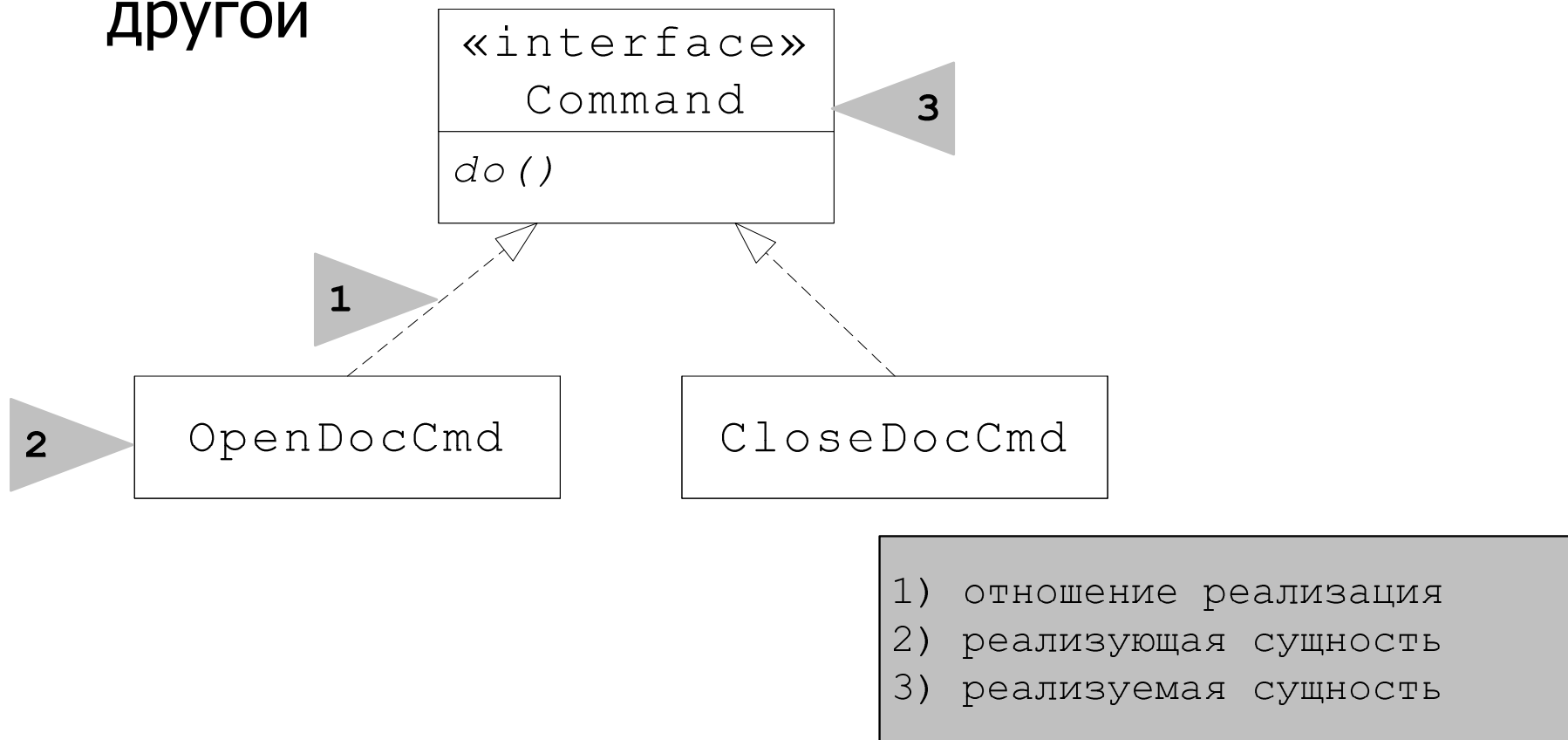
- **Обобщение:** одна сущность частный случай другой



- 1) отношение обобщения
- 2) специализация (подкласс)
- 3) обобщение (суперкласс)

# Отношение реализации

- **Реализация:** одна сущность *реализация* другой

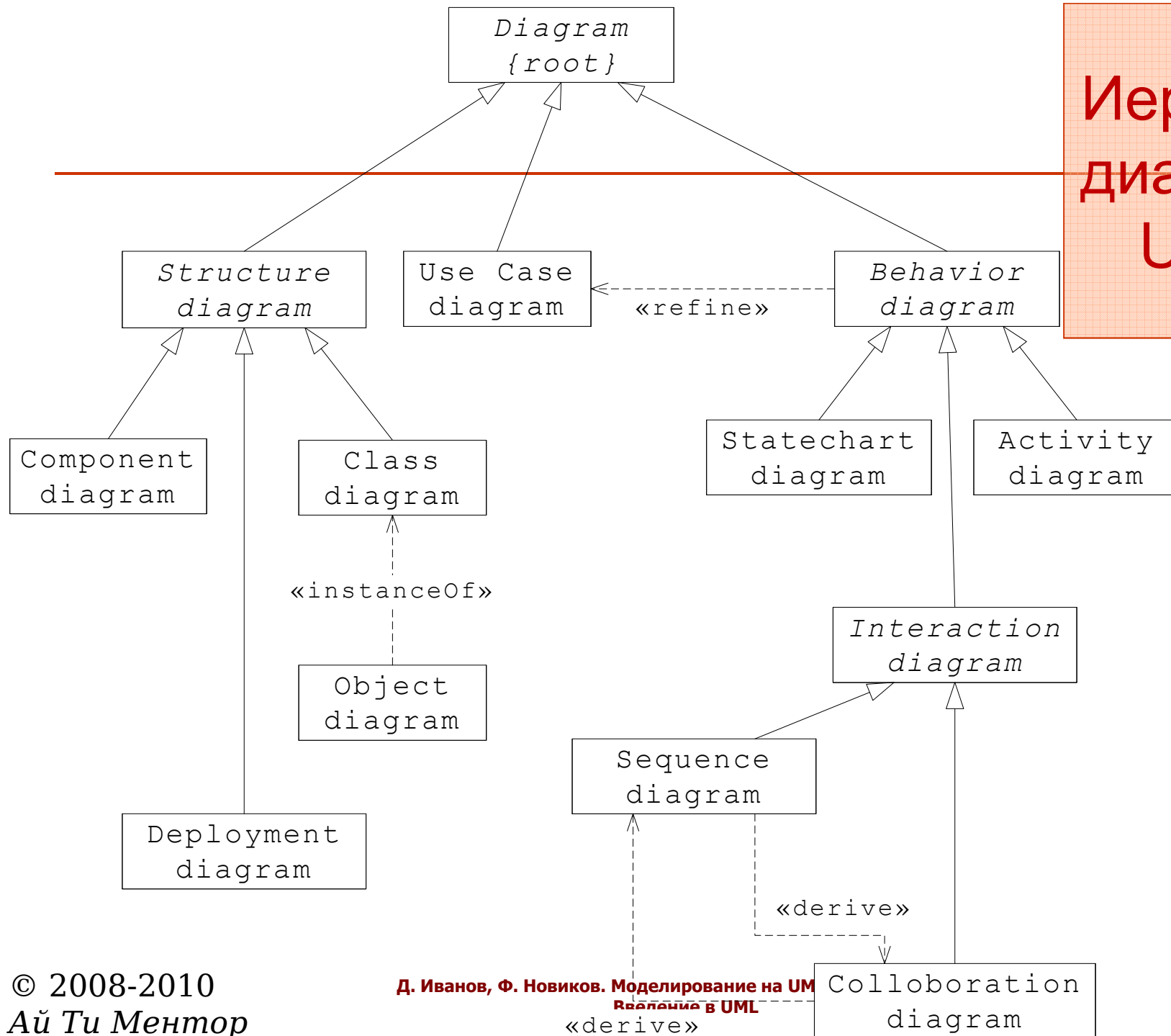


## 5. Диаграммы

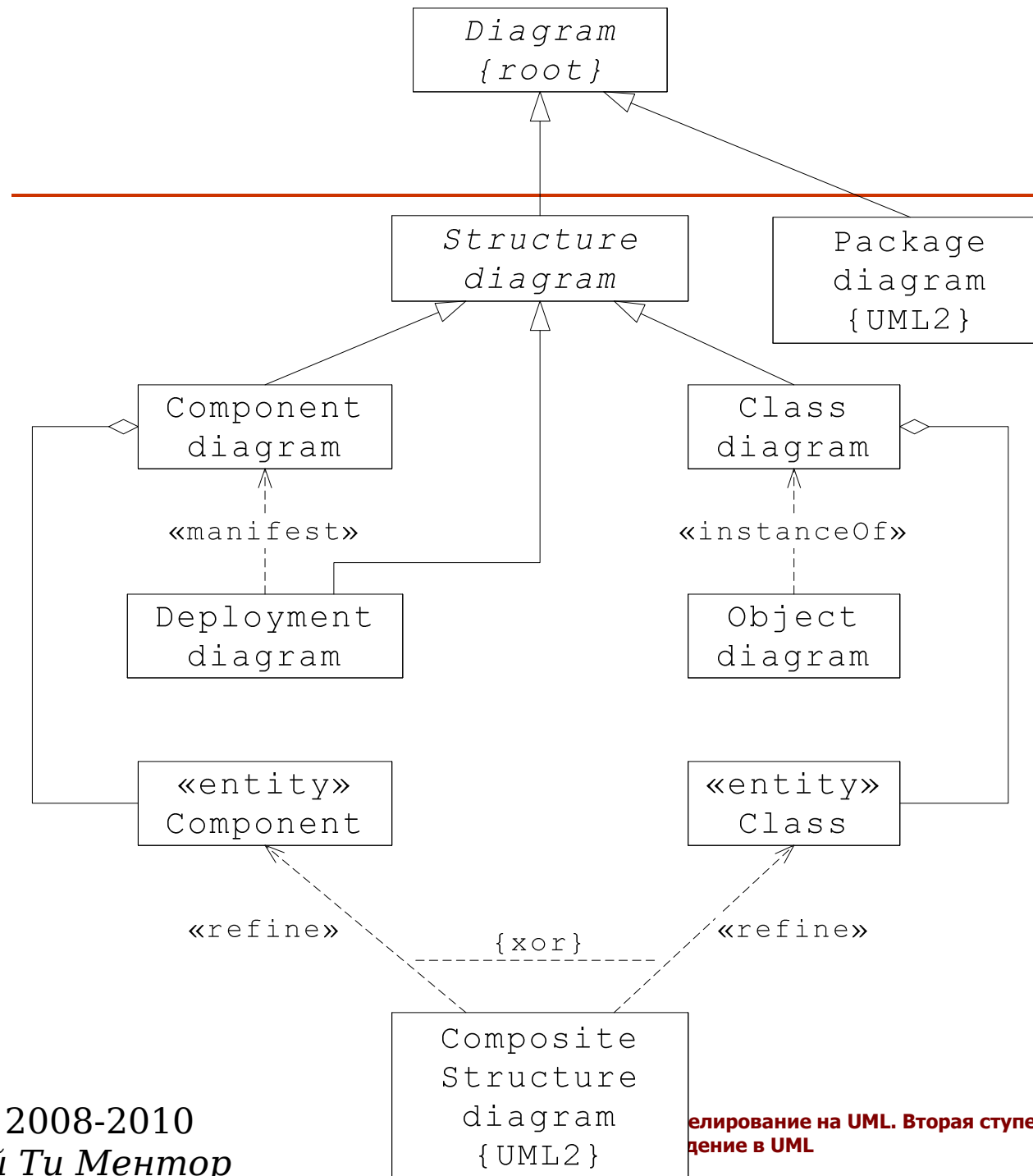
---

- **Диаграмма** — это графическое представление некоторой части графа  
— это накладываемая на модель структура,  
которая облегчает создание и  
использование модели
- **Модель** – объединение диаграмм

# Иерархия диаграмм UML 1

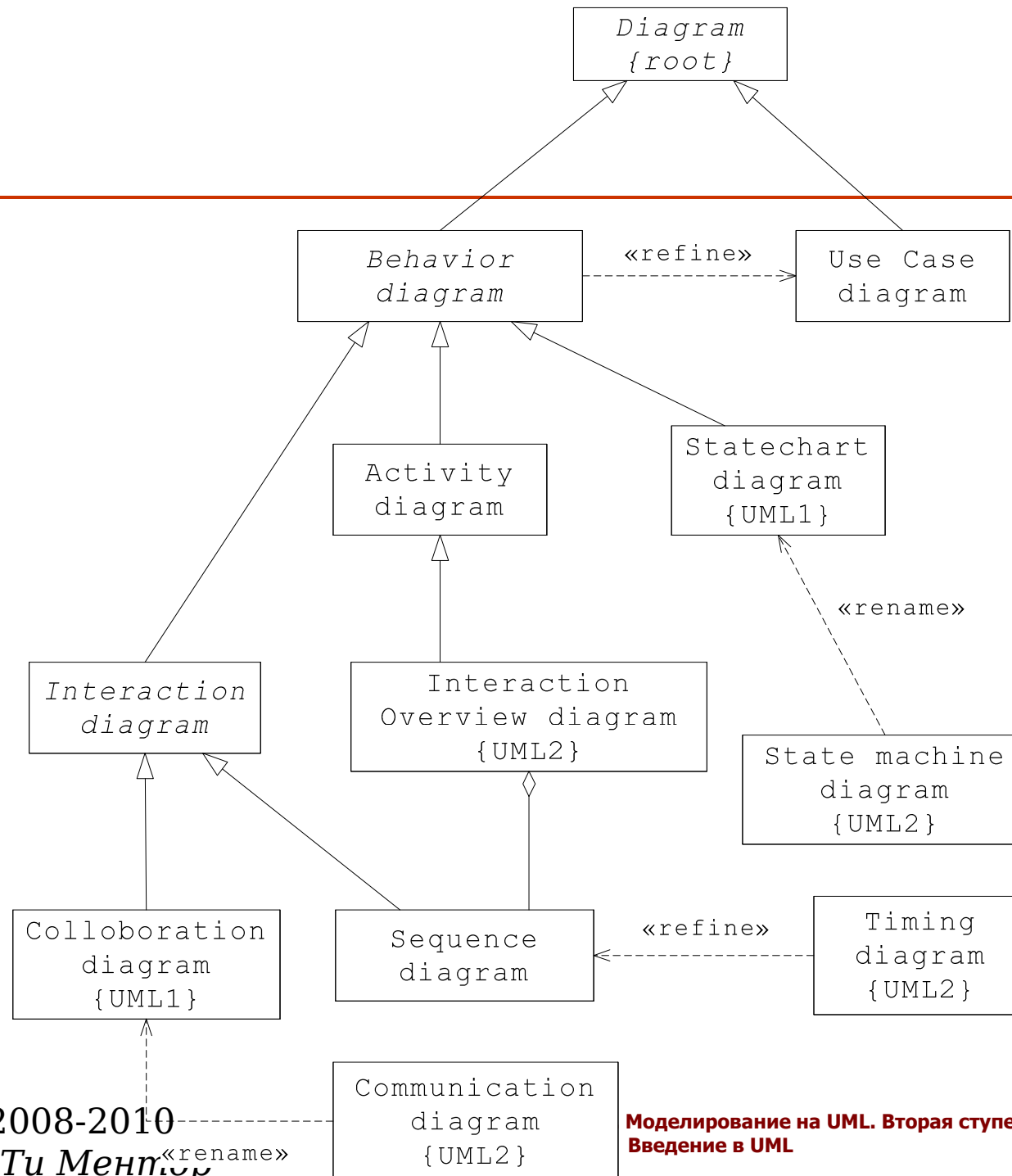


# Иерархия диаграмм UML 2 (i)

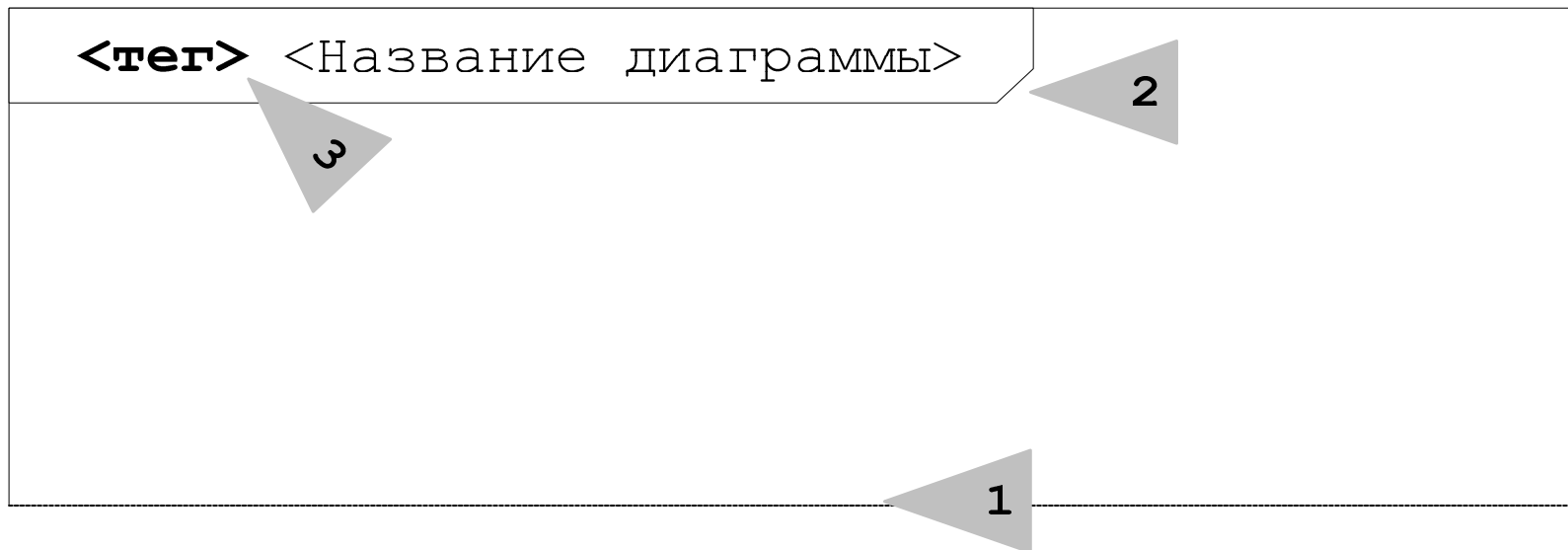




# Иерархия диаграмм UML 2 (ii)



# Нотация диаграммы



- 1) наружная рамка
- 2) ярлычок
- 3) тег и название

# Теги диаграмм (i)

Диаграмма	Тег (стандарт)	Тег (практика)
использования	use case или uc	use case
классов	class	class
автомата	state machine или stm	state machine
деятельности	activity или act	activity
последовательност и	interaction или sd	sd
коммуникации	interaction или sd	comm

# Теги диаграмм (ii)

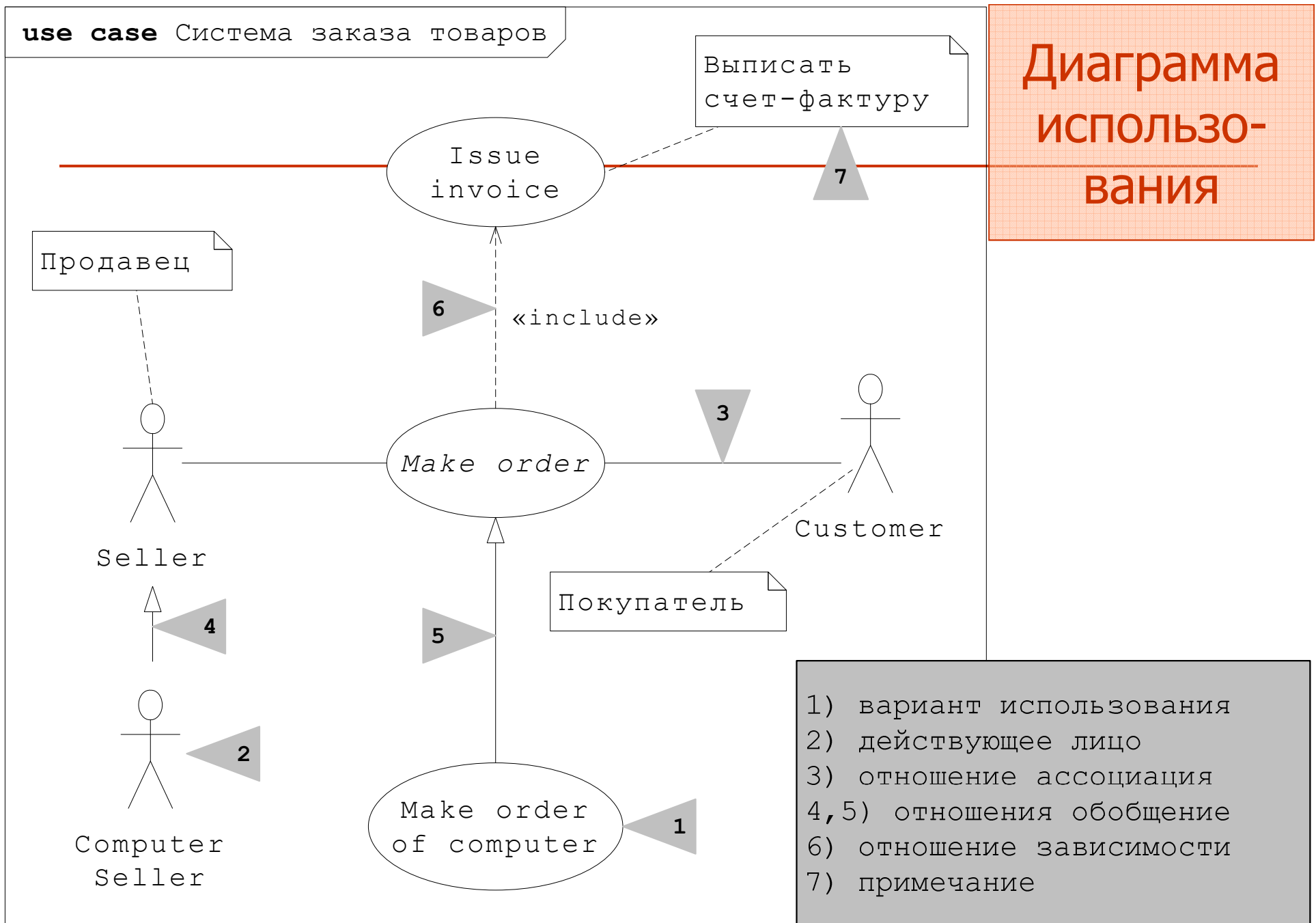
Диаграмма	Тег (стандарт)	Тег (практика)
компонентов	component или cpr	component
размещения	<i>не определен</i>	deployment
объектов	<i>не определен</i>	object
внутренней структуры	class	class или component
обзорная взаимодействия	interaction или sd	interaction
синхронизации	interaction или sd	timing
пакетов	package или pkg	package

# Общие диаграммы

---

- Практически не зависят от предмета моделирования
- Могут применяться в любом программном проекте

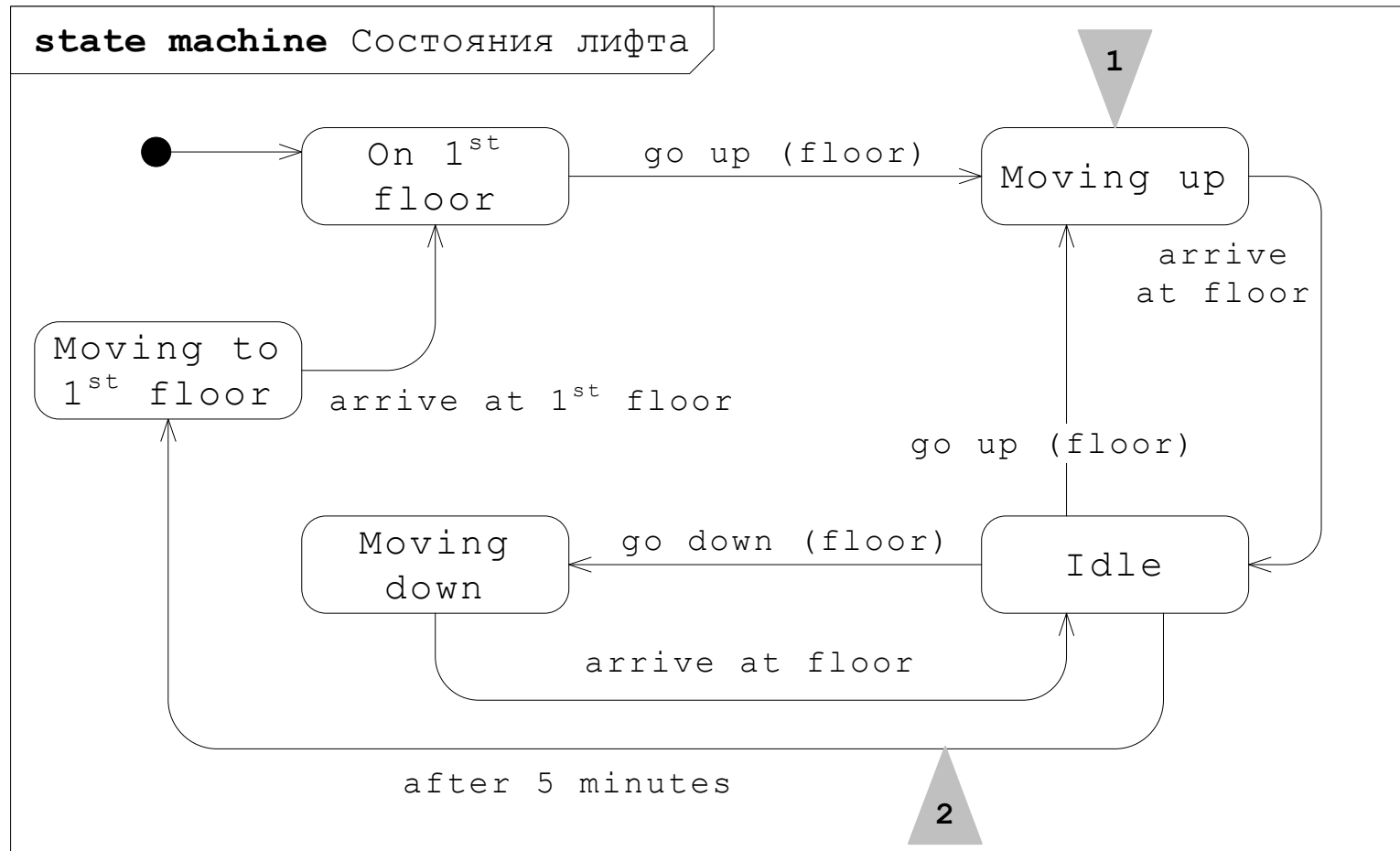
1. Диаграмма использования → часть 2
2. Диаграмма классов → часть 3
3. Диаграмма автомата → часть 4
4. Диаграмма деятельности → часть 4
5. Диаграмма последовательности → часть 4
6. Диаграмма коммуникации → часть 4
7. Диаграмма компонентов → часть 3
8. Диаграмма размещения → часть 3



# Диаграмма классов



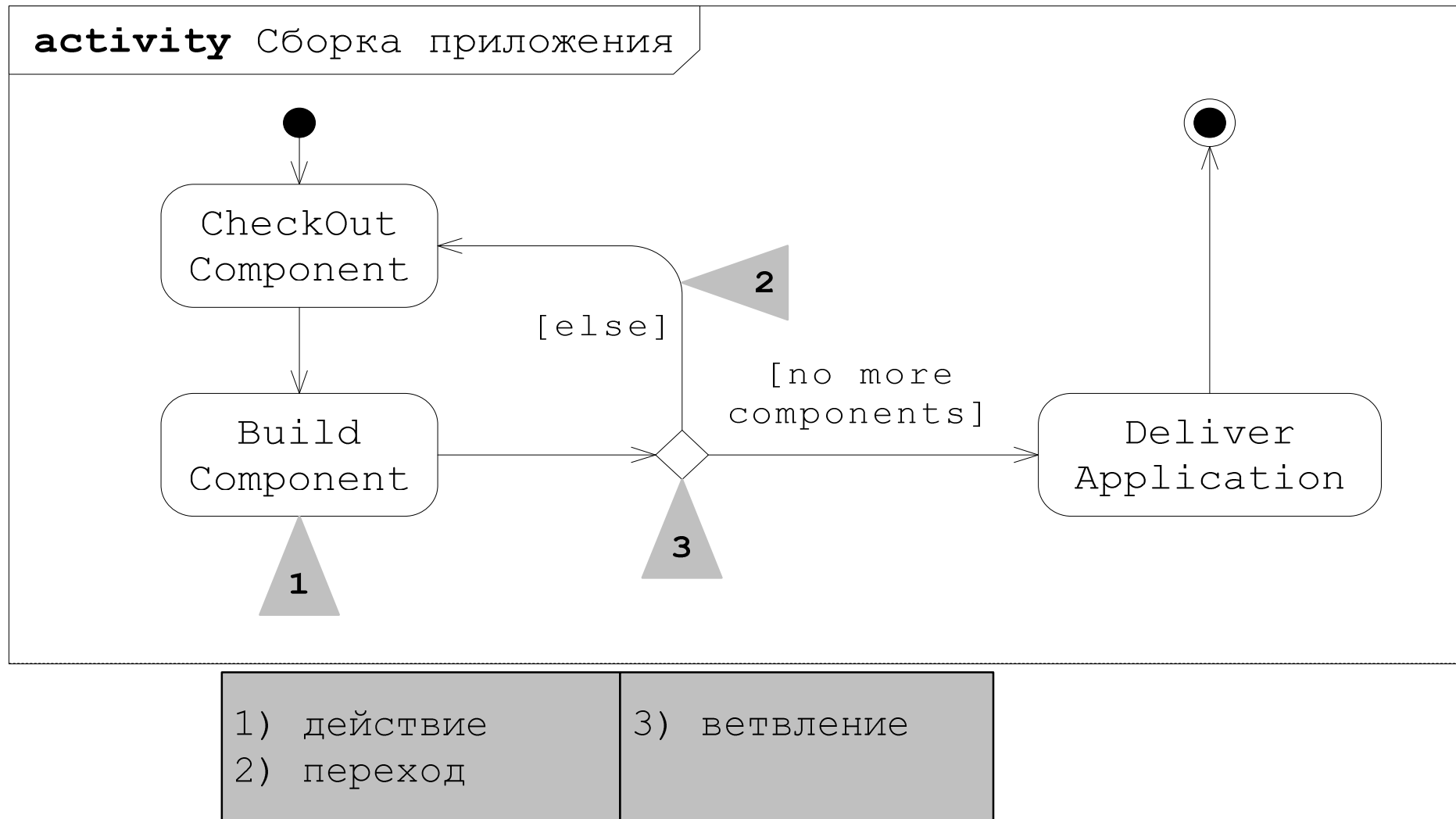
# Диаграмма автомата



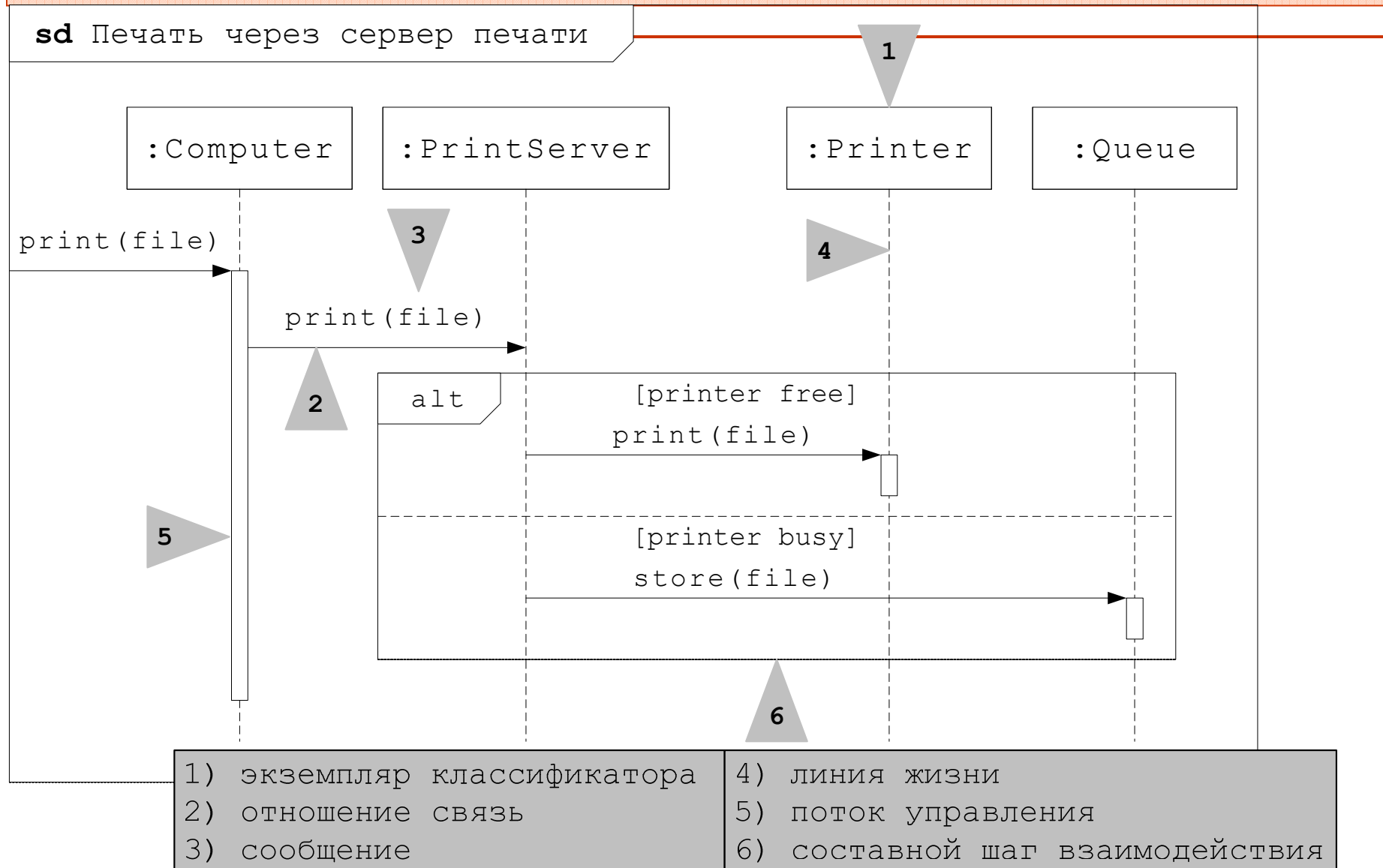
1) состояние      2) переход



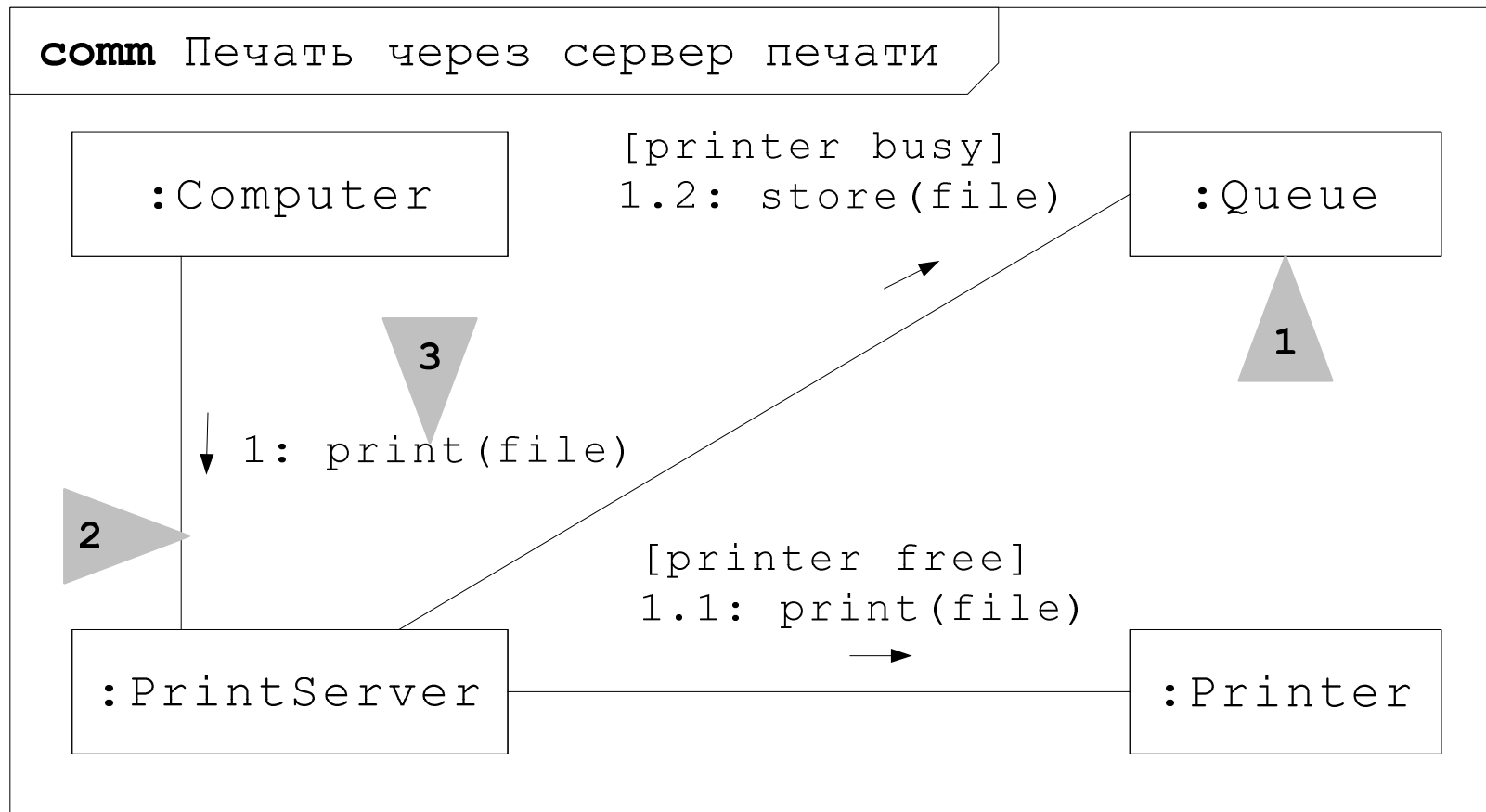
# Диаграмма деятельности



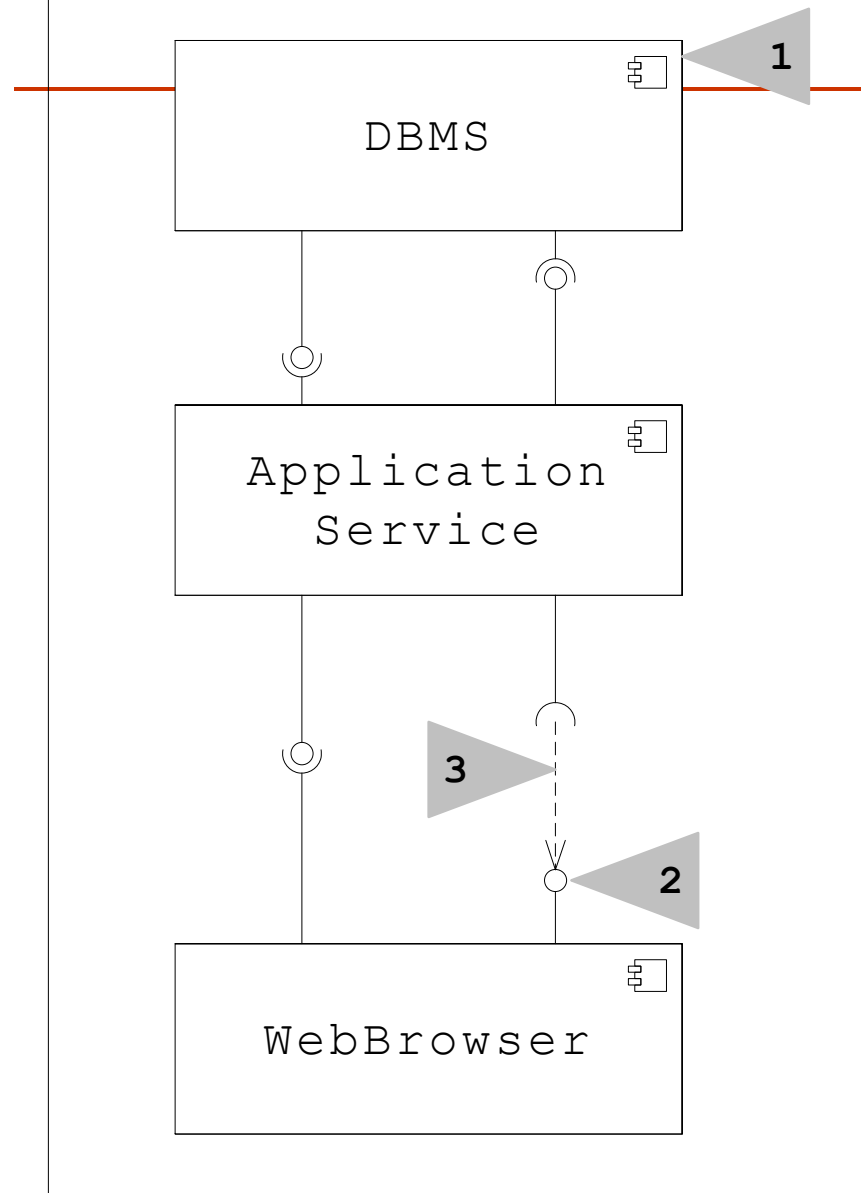
# Диаграмма последовательности



# Диаграмма коммуникации

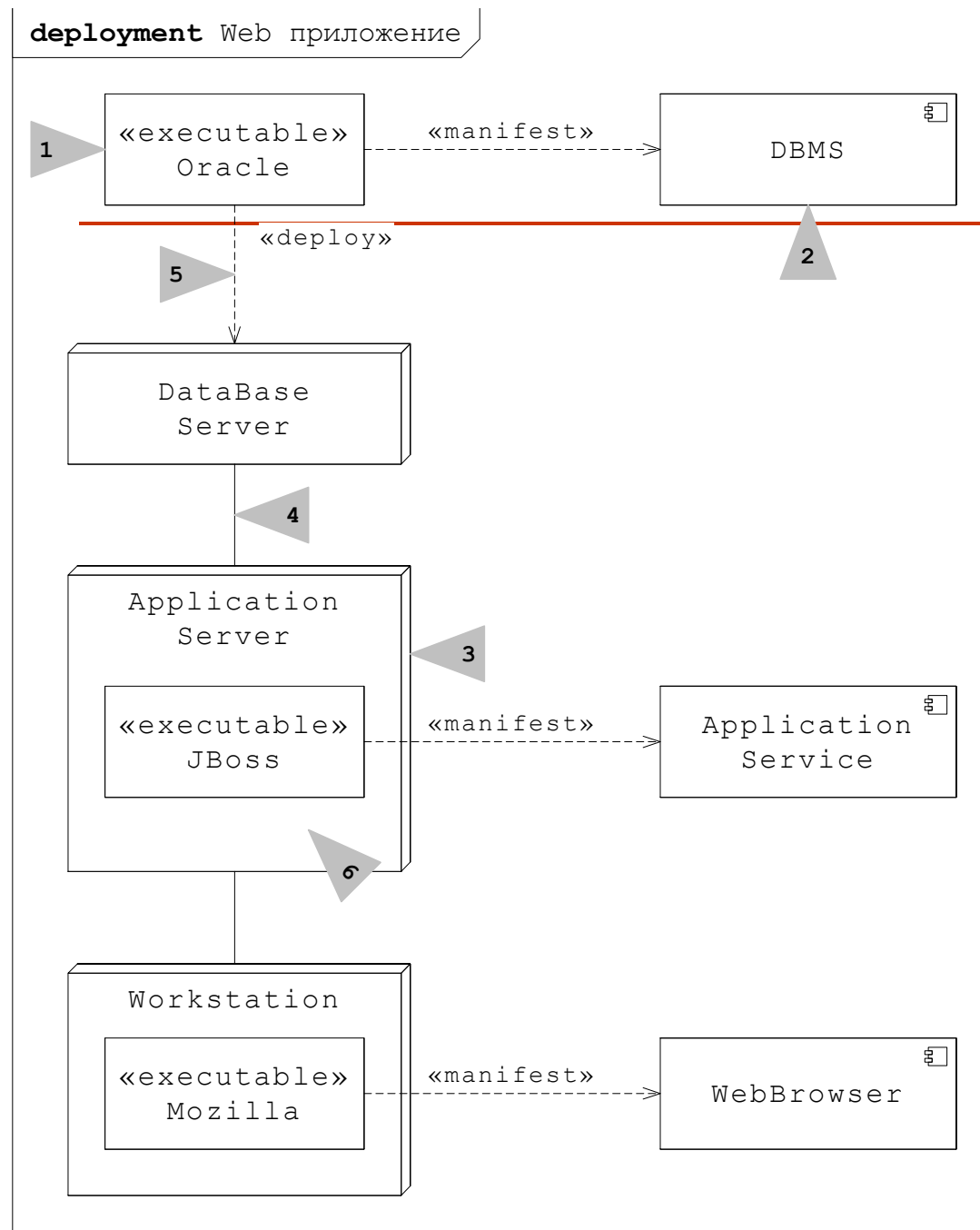


**component** Web приложение



# Диаграмма компонентов

- 1) компонент
- 2) интерфейс
- 3) отношение зависимости



# Диаграмма размещения

- 1) артефакт
- 2) компонент
- 3) узел
- 4) отношение ассоциации
- 5) отношение зависимости
- 6) сущность является частью другой сущности

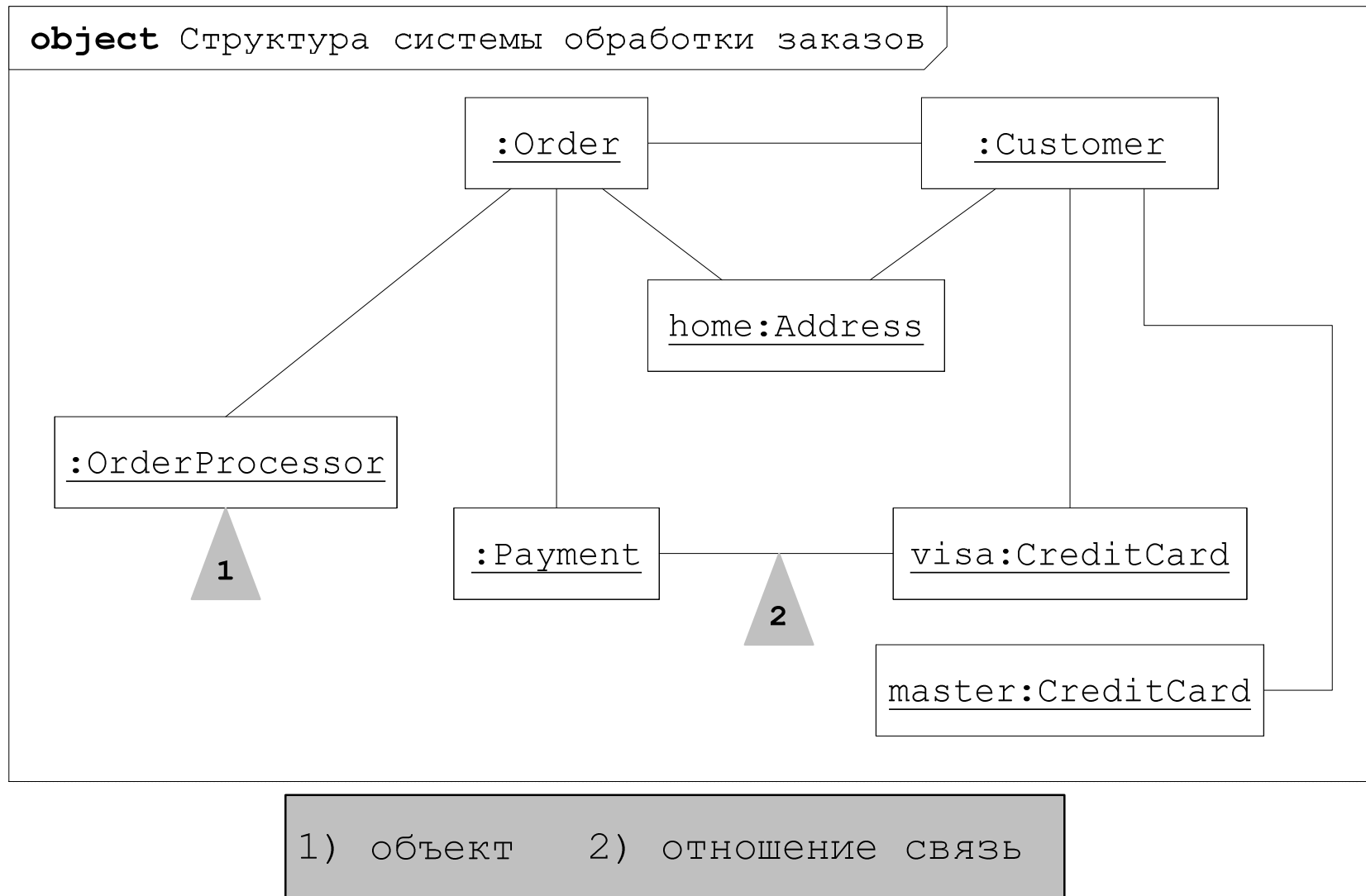
# Специальные диаграммы

---

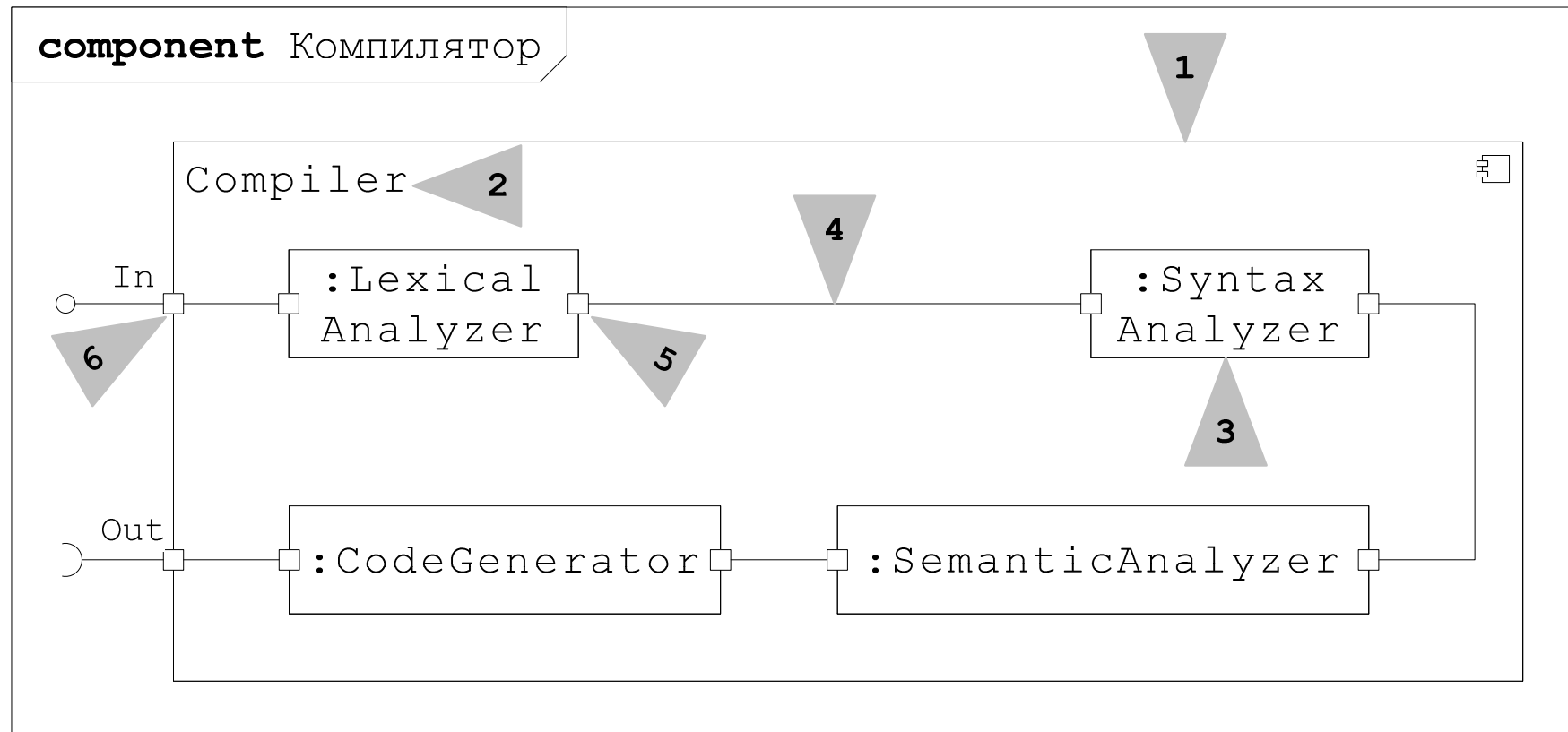
- служат для дополнения какой-либо общей диаграммы
- являются частным случаем
- уточняют детали

1. Диаграмма объектов → часть 3
2. Диаграмма внутренней структуры → часть 3
3. Обзорная диаграмма взаимодействия → часть 4
4. Диаграмма синхронизации → часть 4
5. Диаграмма пакетов → часть 5

# Диаграмма объектов



# Диаграмма внутренней структуры

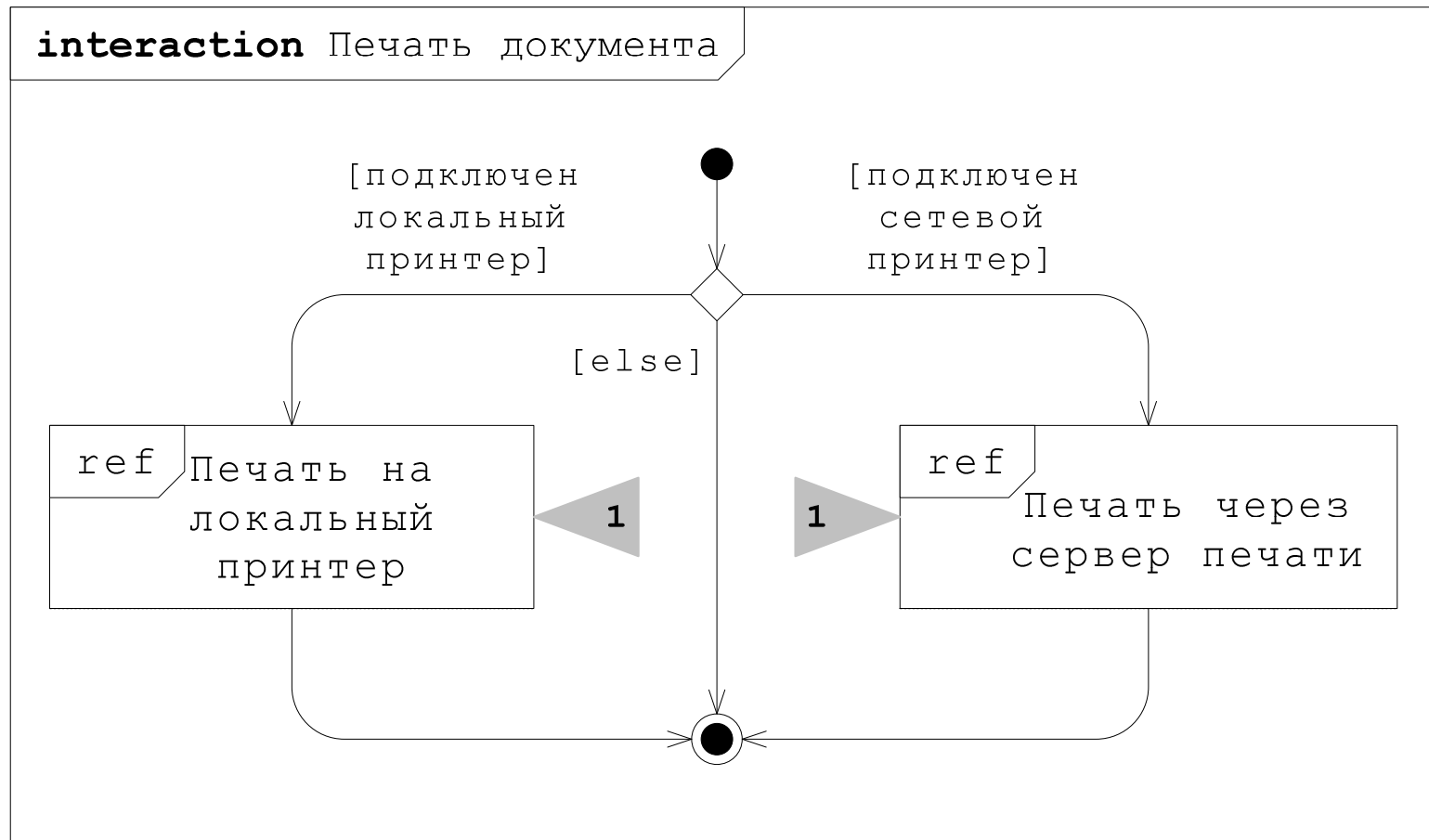


1) структурный классификатор  
2) имя классификатора  
3) часть

4) соединитель  
5, 6) порт

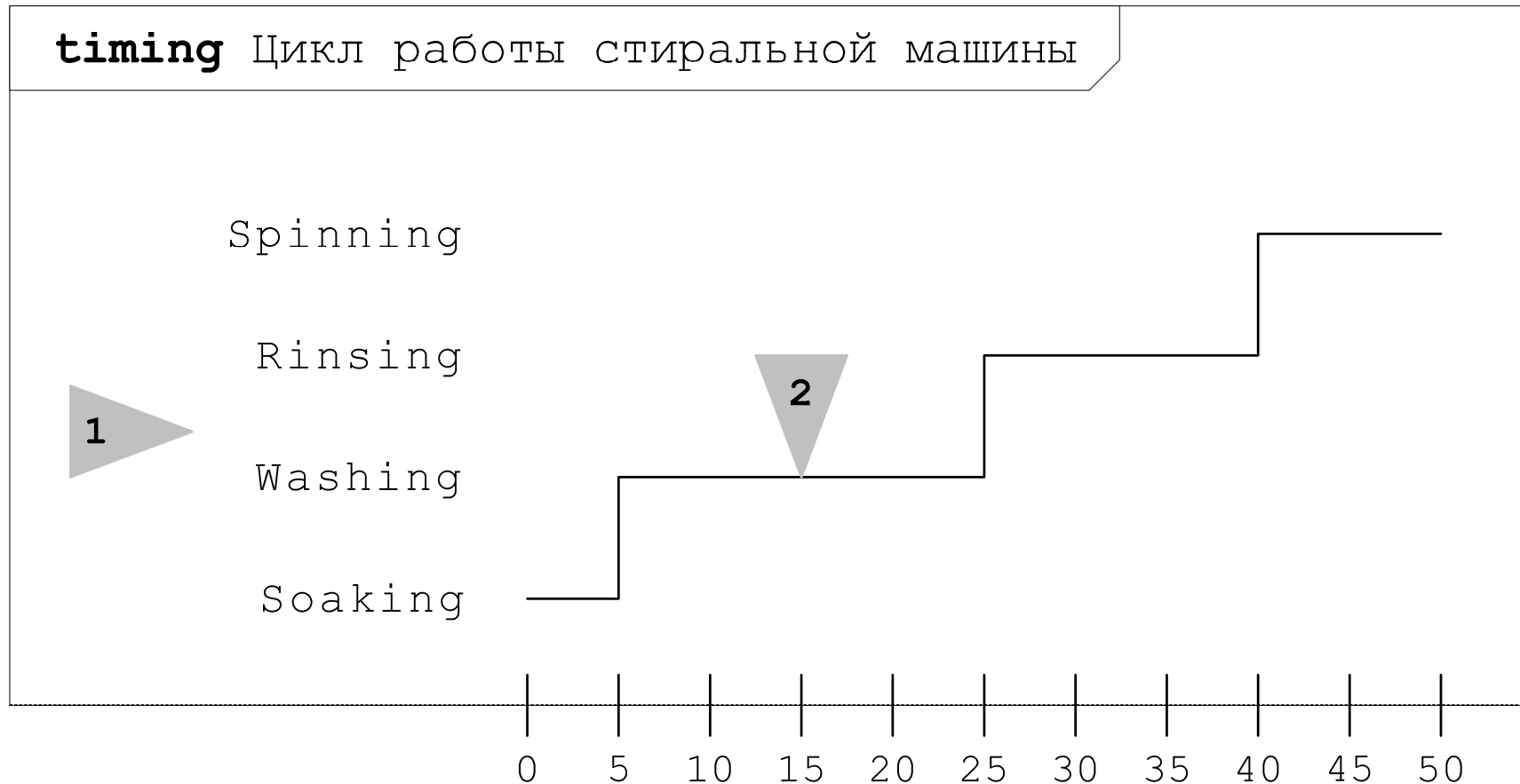


# Обзорная диаграмма взаимодействия



1) ссылка на взаимодействие

# Диаграмма синхронизации

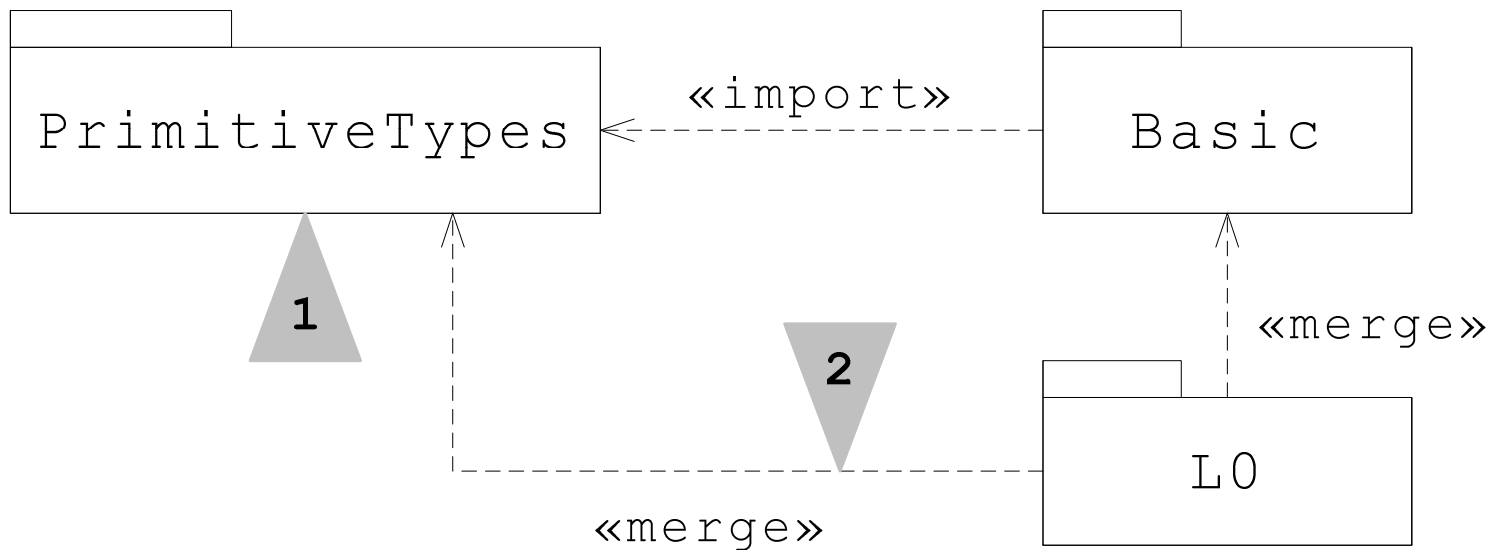


1) состояния

2) временная синхронизация состояний

# Диаграмма пакетов

**package** Пакеты UML верхнего уровня L0



1) пакет

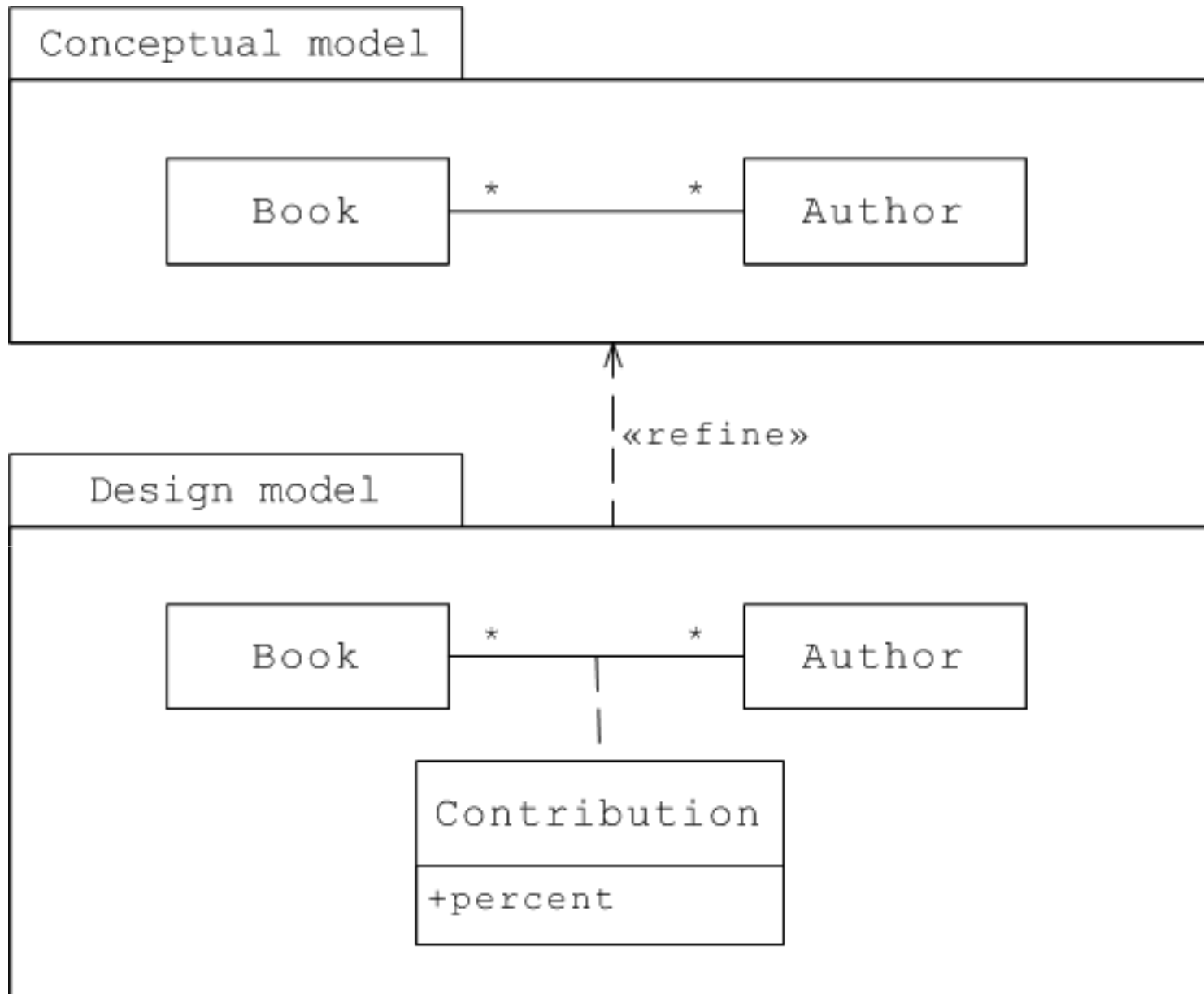
2) отношение зависимость

## 6. Модели и представления

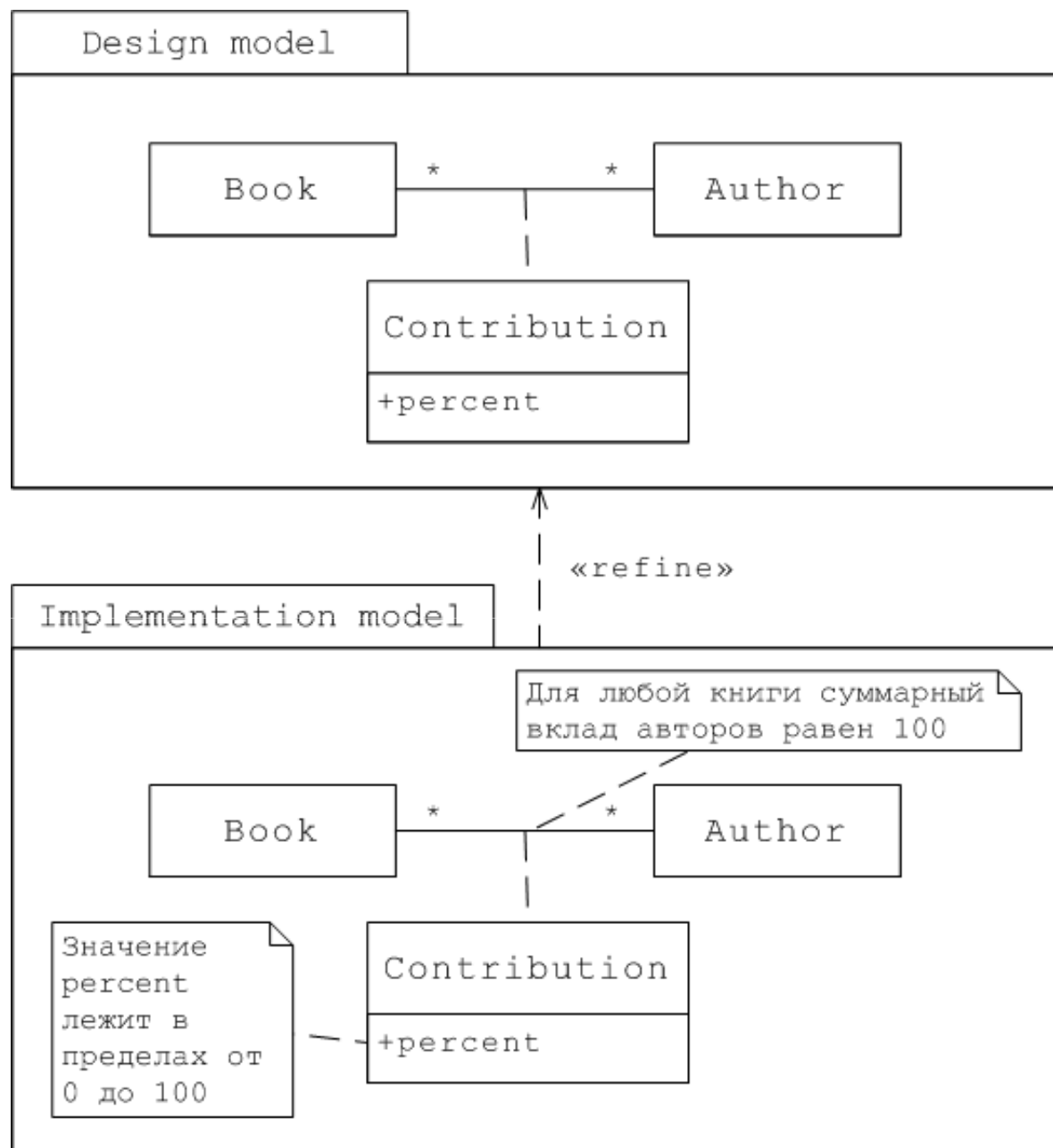
- Реальные модели сложны,  
сложную модель **НЕВОЗМОЖНО**  
обозреть **с одной точки зрения**

**Представление** — проекция  
(фильтрация) единого графа модели  
— средство  
логического структурирования модели

## Уровни моделей (i)

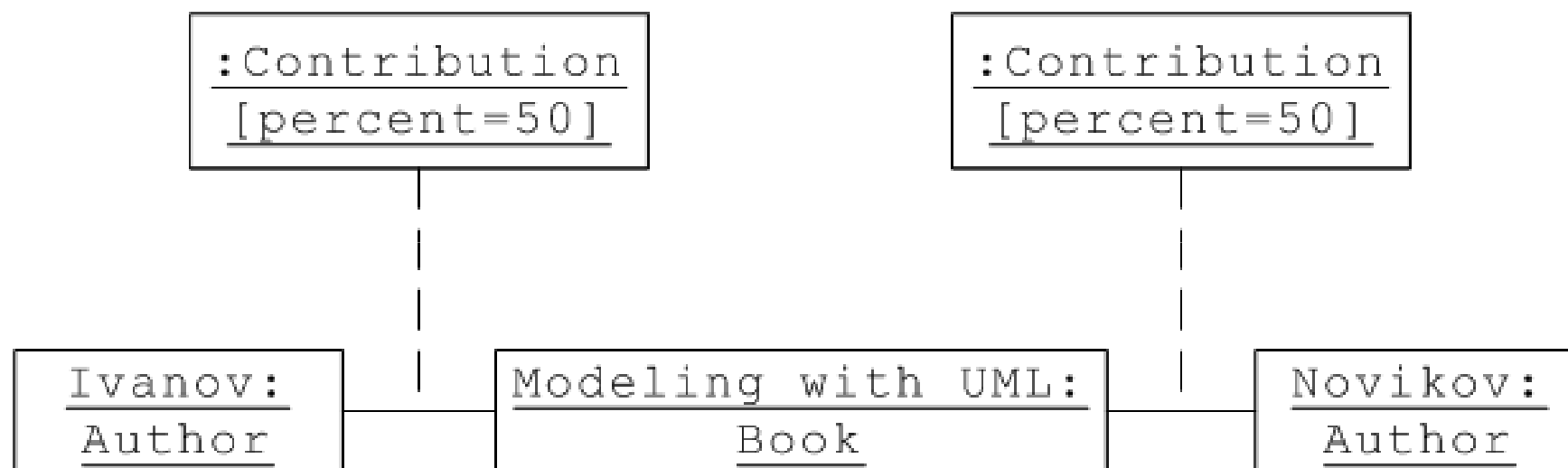


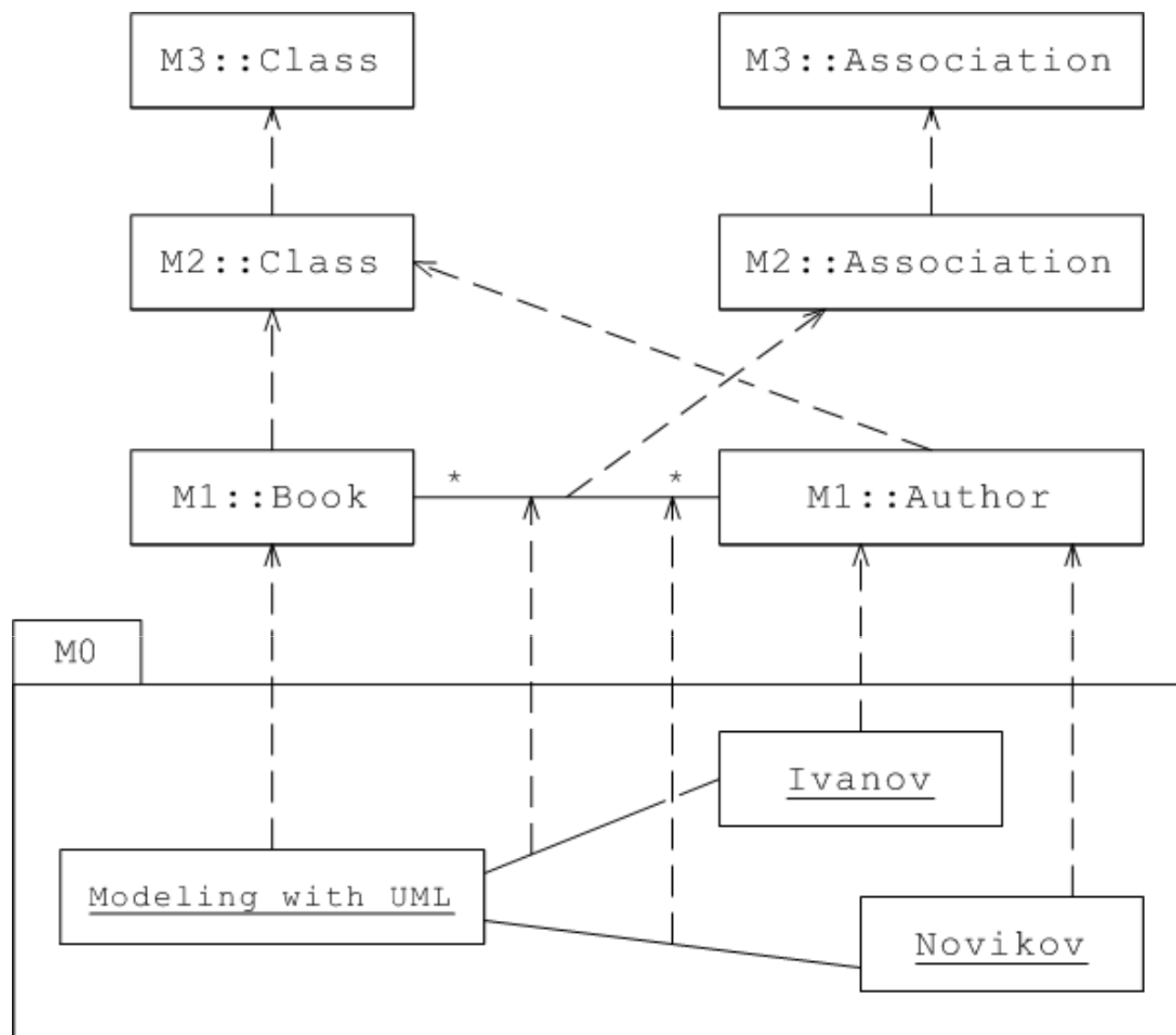
## Уровни моделей (ii)



# Уточняющая диаграмма объектов

Уровни  
моделей  
(iii)





Уровни  
моделей  
(iv)  
уровни  
мета-  
модели-  
рования

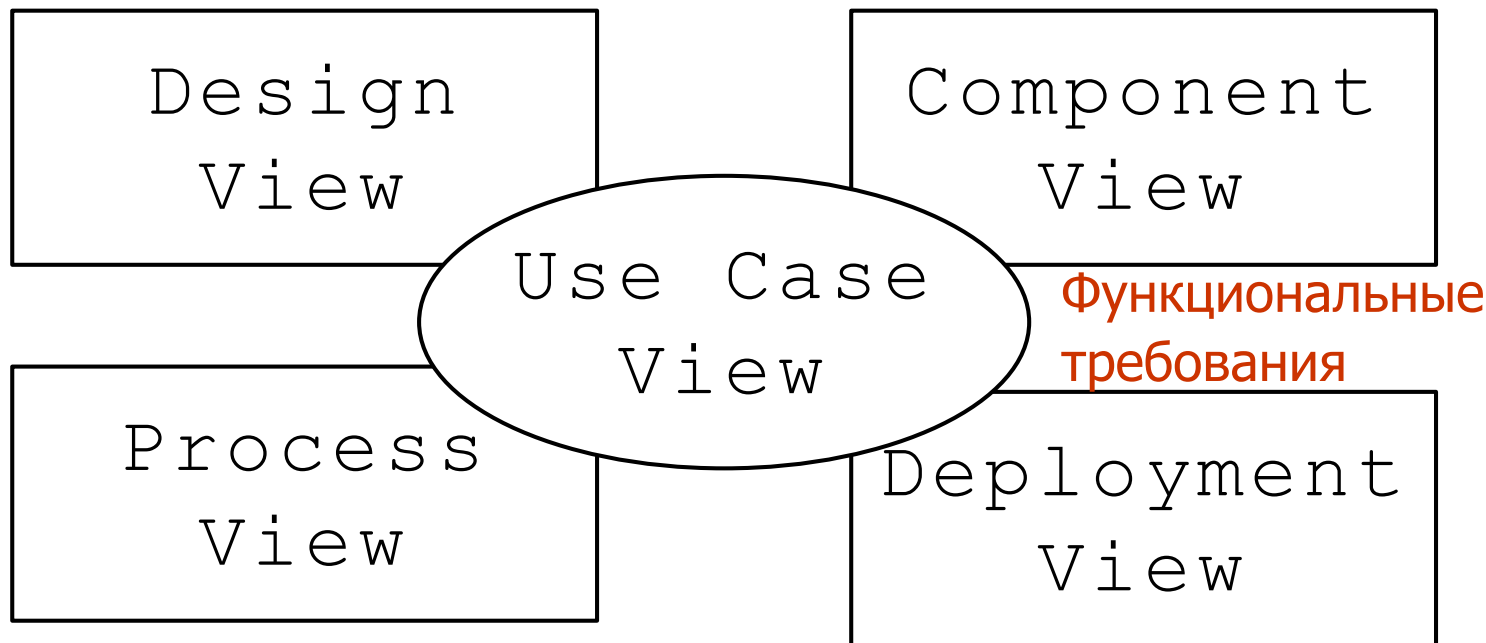
Все зависимости со стереотипом «instanceOf»



# Классические представления

Словарь предметной области

Сборка системы, управление конфигурацией



Производительность,  
масштабируемость, пропускная  
способность

Топология системы,  
распределение, установка

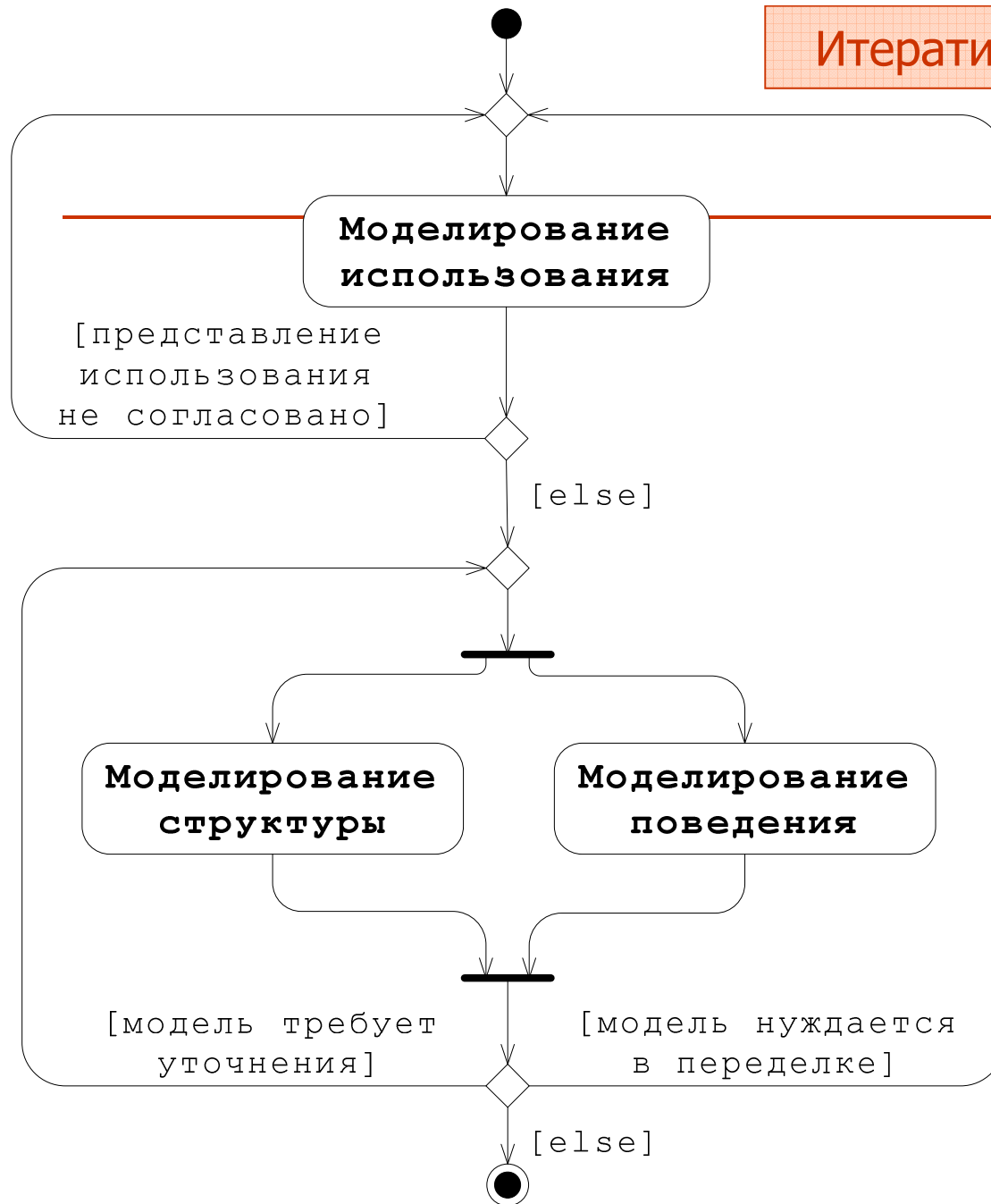


# Три представления



- Представление использования
  - ЧТО делает система
  - Диаграммы использования
- Представление структуры
  - ИЗ ЧЕГО состоит система
  - Диаграммы классов, компонентов и размещения
- Представление поведения
  - КАК работает система
  - Диаграммы автомата, деятельности и взаимодействия

## Итеративный процесс моделирования



# 7. Общие механизмы

---

Внутреннее представление = Спецификации  
(specifications)

Дополнения (adornments) UML 1 →  
Украшения (decorations) UML 2

Стандартные дихотомии (common divisions)

- Класс – объект
- Интерфейс – реализация

Механизмы расширения (extension mechanisms)

- Стереотипы (stereotypes)
- Именованные значения (tagged values)
- Ограничения (constraints)

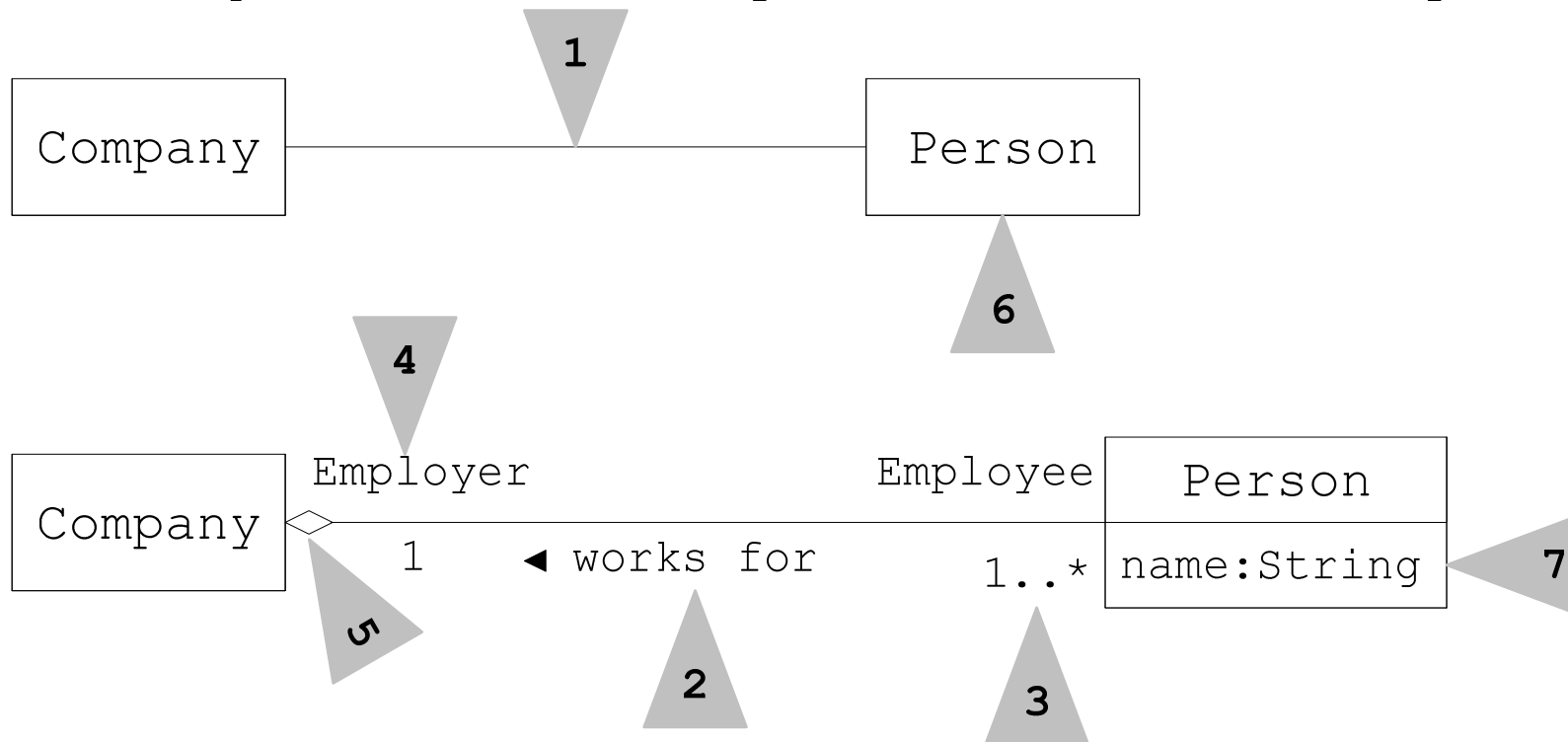
# Внутреннее представление и XMI



```

<?xml version="1.0" encoding="UTF-8" ?>
- <uml:Model xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
  xmlns:uml="http://www.eclipse.org/uml2/1.0.0/UML" xmi:id="_3c0YALFAEdqS_00jbZd_dg"
  name="VerySimpleProject">
  <ownedMember xmi:type="uml:Association" xmi:id="_3c0YAbFAEdqS_00jbZd_dg" name="Unknown
    Name" memberEnd="_3c0YA7FAEdqS_00jbZd_dg_3c0YB7FAEdqS_00jbZd_dg" />
- <ownedMember xmi:type="uml:Class" xmi:id="_3c0YArFAEdqS_00jbZd_dg" name="Class1">
- <ownedAttribute xmi:id="_3c0YA7FAEdqS_00jbZd_dg" type="_3c0YBrFAEdqS_00jbZd_dg"
  association="_3c0YAbFAEdqS_00jbZd_dg">
  <upperValue xmi:type="uml:LiteralUnlimitedNatural" xmi:id="_3c0YBLFAEdqS_00jbZd_dg"
    value="1" />
  <lowerValue xmi:type="uml:LiteralInteger" xmi:id="_3c0YBbFAEdqS_00jbZd_dg"
    value="1" />
  </ownedAttribute>
</ownedMember>
- <ownedMember xmi:type="uml:Class" xmi:id="_3c0YBrFAEdqS_00jbZd_dg" name="Class2">
- <ownedAttribute xmi:id="_3c0YB7FAEdqS_00jbZd_dg" type="_3c0YArFAEdqS_00jbZd_dg"
  association="_3c0YAbFAEdqS_00jbZd_dg">
  <upperValue xmi:type="uml:LiteralUnlimitedNatural" xmi:id="_3c0YCLFAEdqS_00jbZd_dg"
    value="1" />
  <lowerValue xmi:type="uml:LiteralInteger" xmi:id="_3c0YCbFAEdqS_00jbZd_dg"
    value="1" />
  </ownedAttribute>
</ownedMember>
</uml:Model>
  
```

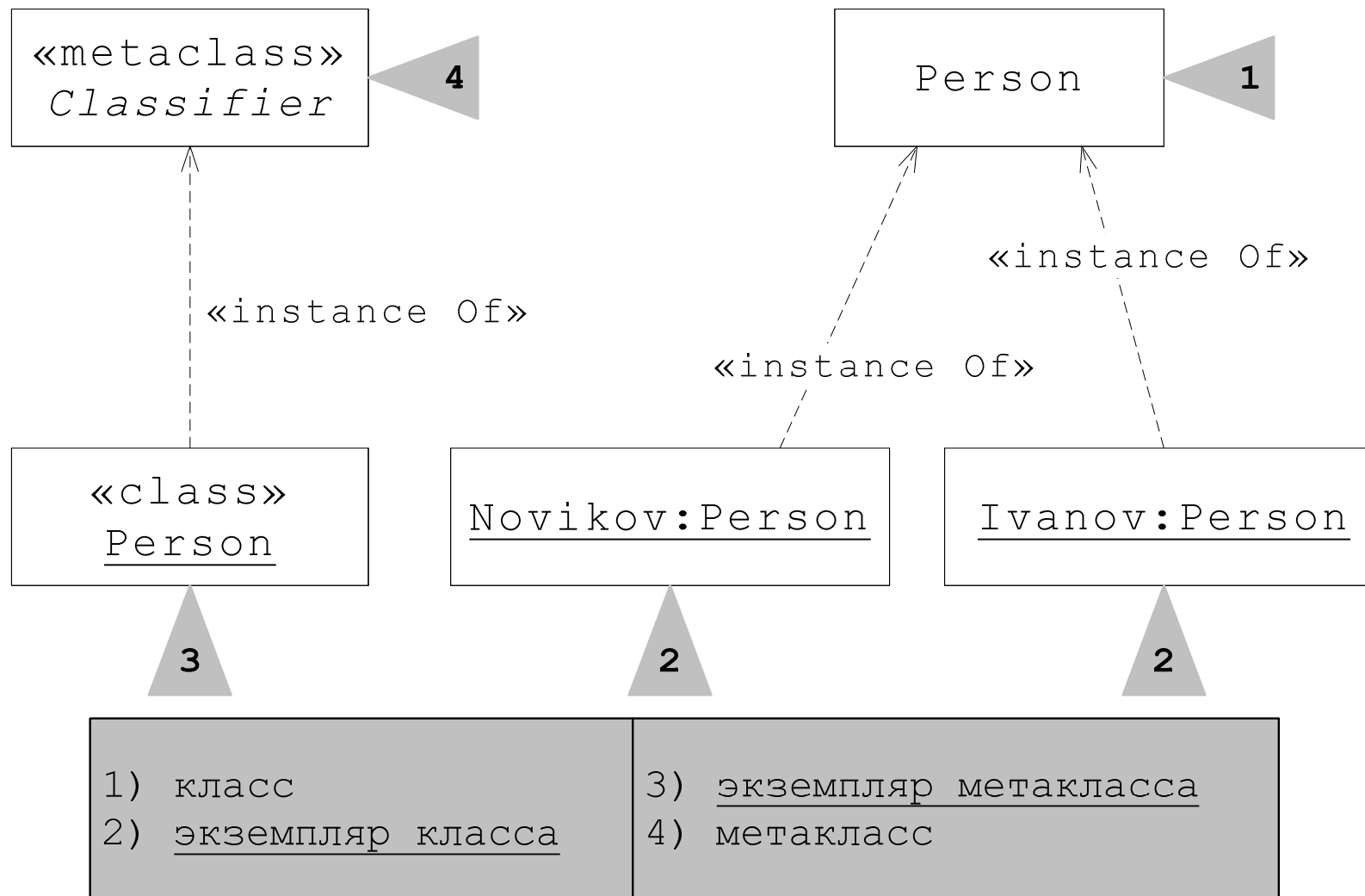
# Украшения (Дополнения)



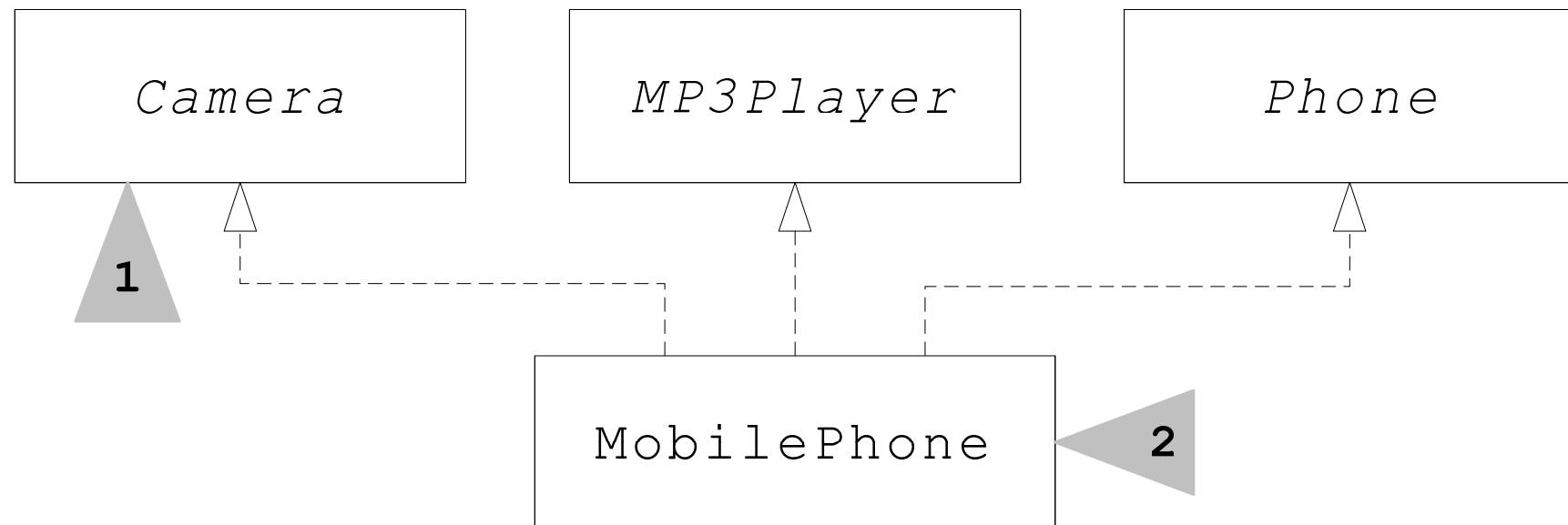
- 1) отношение ассоциация
- 2) имя ассоциации
- 3) кратность полюса
- 4) роль
- 5) агрегация

- 6) базовая нотация класса
- 7) атрибут

# Стандартные дихотомии. Класс-объект



# Стандартные дихотомии. *Интерфейс-реализация*



- 1) интерфейс
- 2) реализация



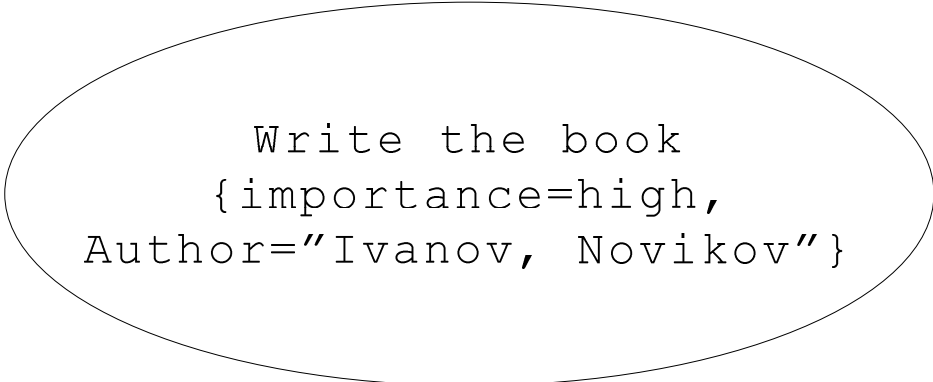
# Механизмы расширения (i)

Механизмы расширения — встроенный в язык способ изменить язык

— позволяют  
определять новые элементы модели на  
основе существующих управляемым и  
унифицированным способом

# Механизмы расширения (ii)

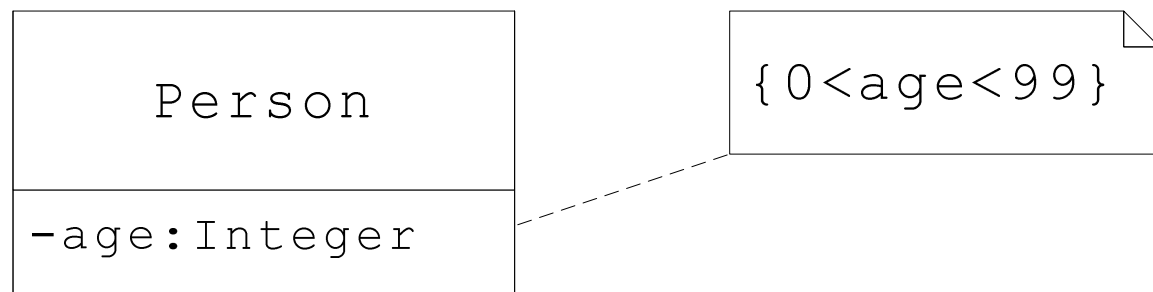
- Помеченное  
значение: имя  
свойства и  
значение свойства



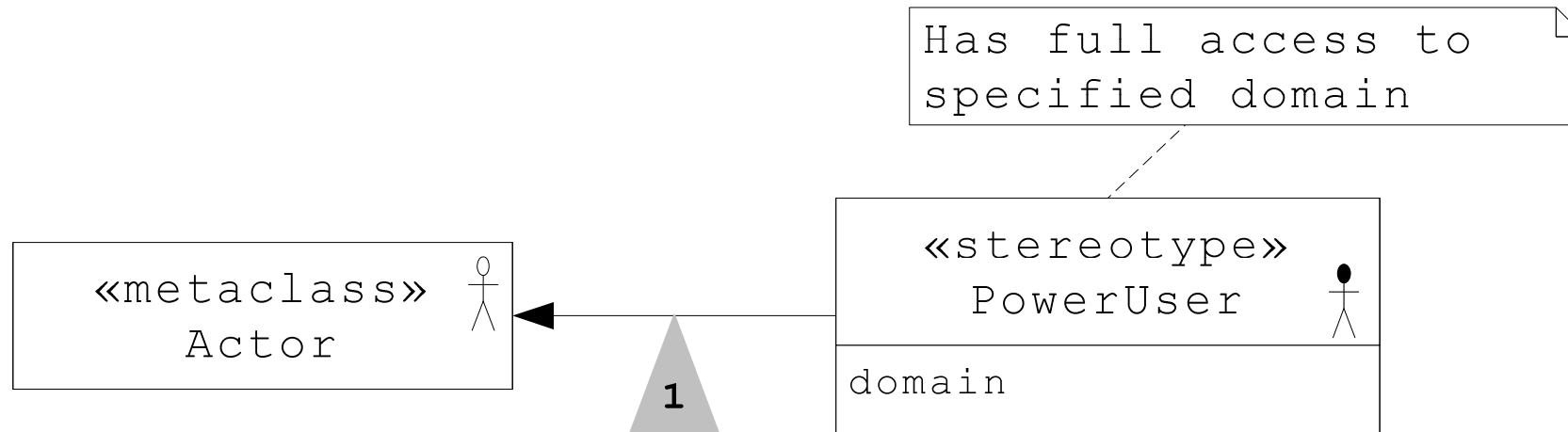
```
usecaseDiagram
    usecase "Write the book" as UC1
    note for UC1 "importance=high, Author='Ivanov, Novikov'"
```

Write the book  
{importance=high,  
Author="Ivanov, Novikov"}

- Ограничение:  
логическое  
утверждение



# Механизмы расширения (iii)

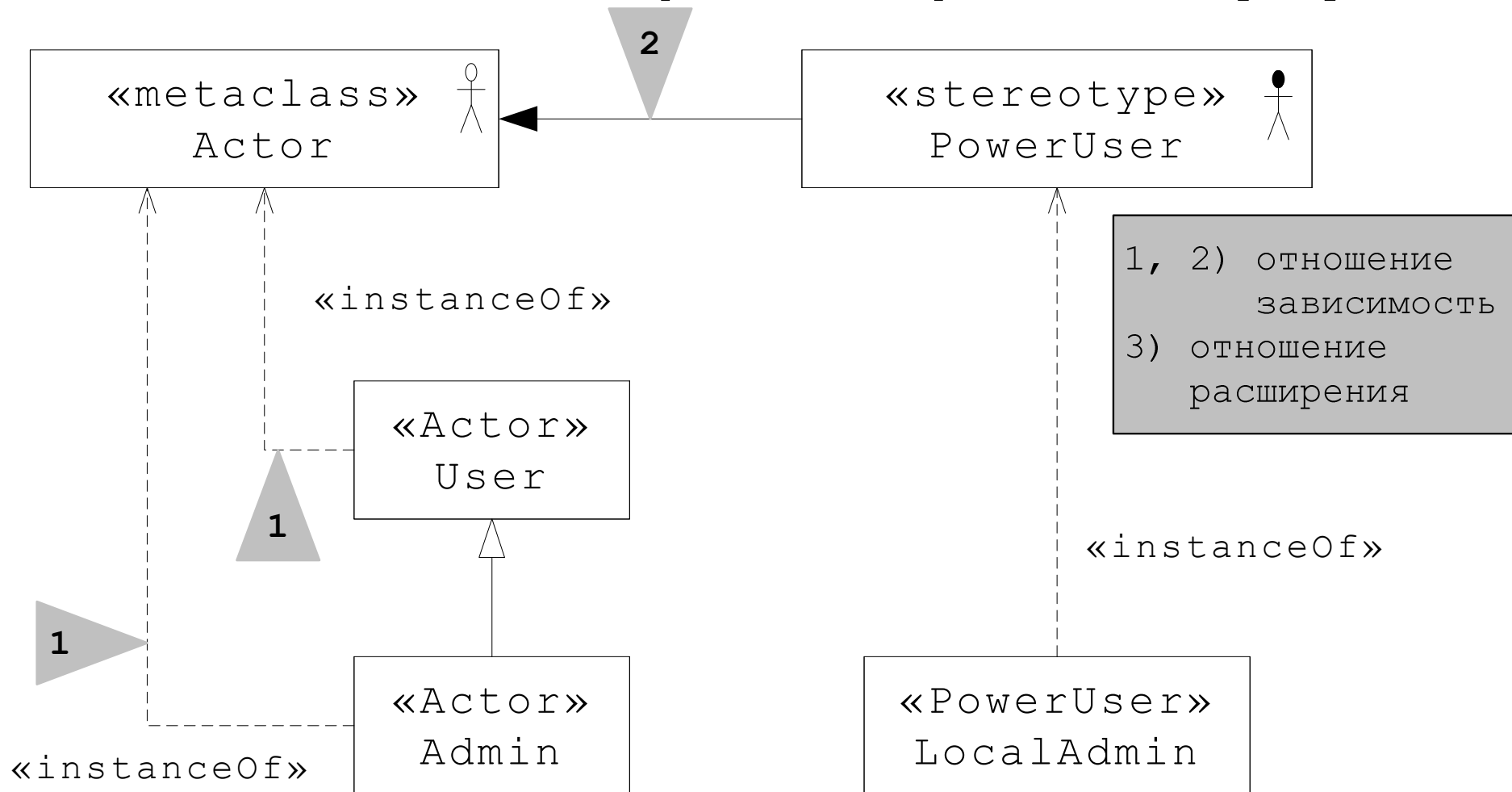


«PowerUser»  
Administrator  
{domain=LAN}

1) отношение расширения

**Стереотип:**  
определение  
нового элемента  
моделирования  
на основе  
существующего

# Механизмы расширения (iv)

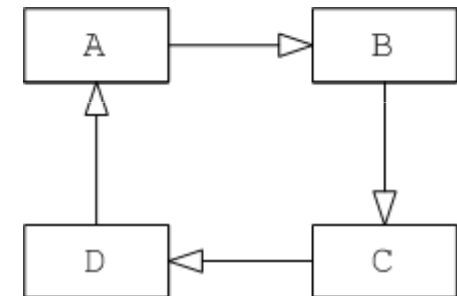


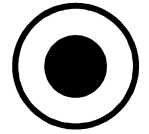
≠ отношение обобщения

# Общие свойства модели

---

1. Синтаксическая правильность
  - Проверяет инструмент
2. Семантическая непротиворечивость
  - Проверяем мы
3. Полнота
  - Недостижима и не нужна
4. Точки вариации семантики (semantic variation points)
  - Статическая или динамическая классификация? Зарыто глубоко





## 8. Выводы

- UML — это **формальный графический** объектно-ориентированный **язык моделирования**, который необходимо освоить
- UML имеет **нотацию**, **семантику** и **прагматику**, которые нужно знать и использовать с учетом особенностей реальной задачи и инструмента
- Модель UML состоит из описания **сущностей** и **отношений** между ними
- Элементы модели группируются в **диаграммы** и **представления** для наилучшего описания моделируемой системы с различных точек зрения
- В случае необходимости элементы UML могут быть **расширены** и **переопределены** средствами самого языка