
Проектирование ИС (часть 2)

Тема 1: Язык UML

Объем лекций по теме: 12 часов

Лектор: Щеголева Людмила Владимировна

Тема 1: Язык UML

Содержание

- ❑ Введение
- ❑ Диаграммы
- ❑ Последовательность построения диаграмм
- ❑ CASE – системы для построения диаграмм

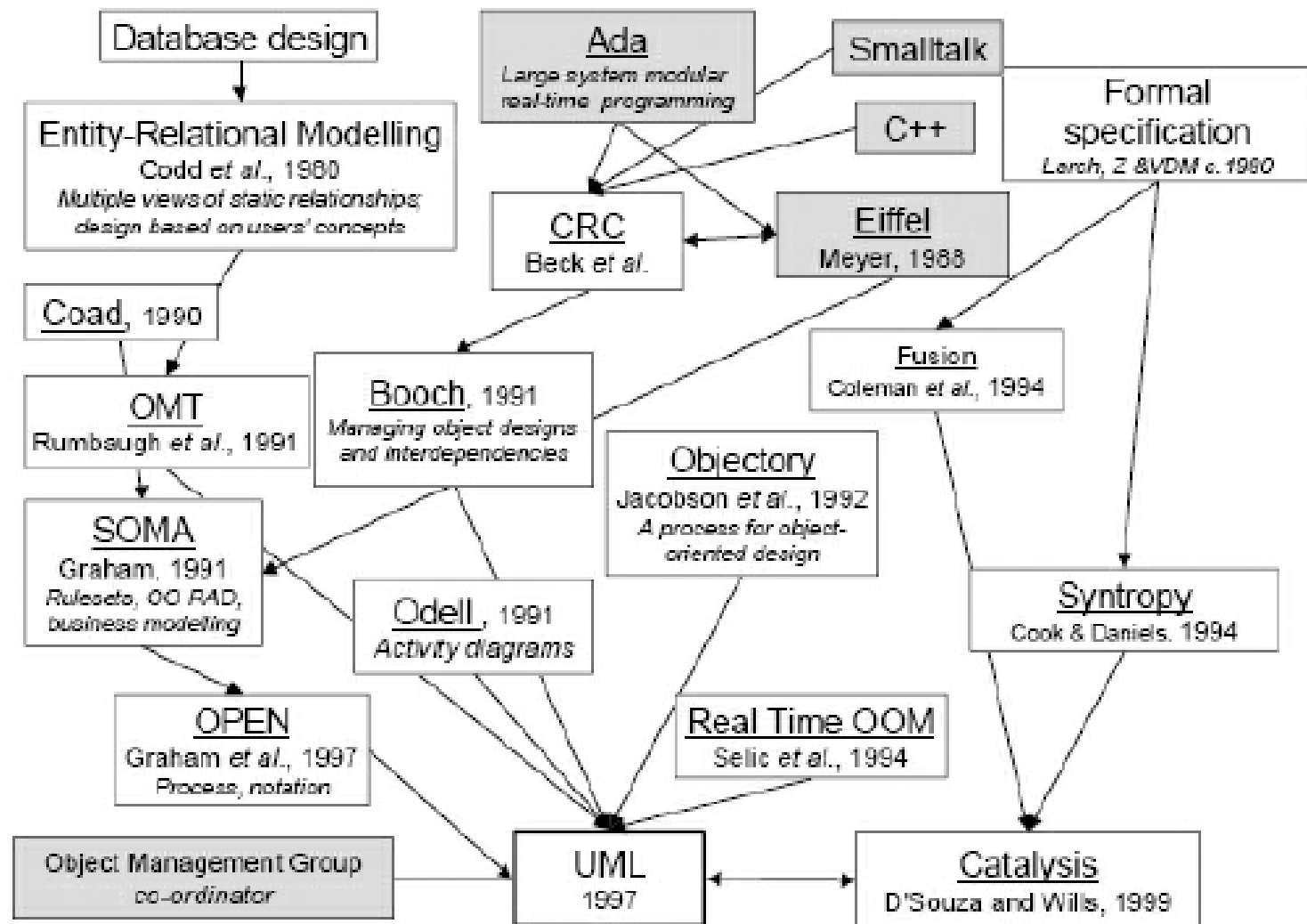
Введение

Тема 1: Язык UML

Введение

- UML (Unified Modeling Language) – Унифицированный Язык Моделирования
- Разработан группой объектного проектирования OMG (Object Management Group)
- Получил статус отраслевого стандарта

Введение



Авторы UML

- Гради Буч (Grady Booch)
- Джеймс Румбах (James Rumbaugh)
- Айвар Якобсон (Ivar Jacobson)

Первичные цели создания UML

- ❑ Предоставить пользователям готовый к использованию язык визуального моделирования
- ❑ Предоставить механизмы расширения и специализации
- ❑ Быть независимым от определенного языка программирования и процесса разработки
- ❑ Интегрировать лучший практический опыт разработок

Диаграммы языка UML

Тема 1: Язык UML

Диаграммы языка UML

- ❑ сценариев (use case diagram)
- ❑ классов (class diagram)
- ❑ состояния (statechart diagram)
- ❑ активности (activity diagram)
- ❑ последовательности (sequence diagram)
- ❑ коммуникации (collaboration diagram)
- ❑ компонентов (component diagram)
- ❑ топологии (deployment diagram)

Диаграммы языка UML

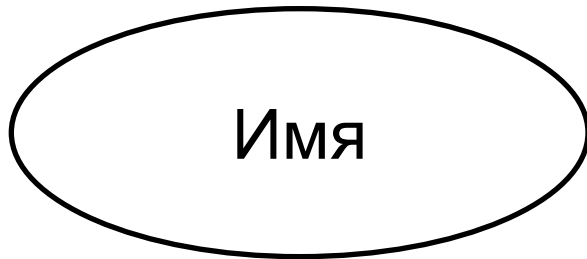
- ❑ композитная структурная диаграмма
- ❑ обзорная диаграмма взаимодействия
- ❑ временная диаграмма
- ❑ диаграмма пакетов

Диаграмма сценариев

- Диаграммы сценариев описывают функциональное назначение системы (то, что система будет делать в процессе своего функционирования)
- Диаграммы сценариев являются исходной концептуальной моделью системы в процессе ее проектирования и разработки

Диаграмма сценариев: элементы

Сценарий



Сценарий – фрагмент поведения ИС без раскрытия его внутренней структуры

Сценарий – сервис, который информационная система предоставляет пользователю (актеру)

Диаграмма сценариев: сценарий

Пример



Диаграмма сценариев: элементы

Актер



Имя

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

Диаграмма сценариев: актер

Пример



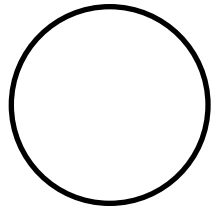
Дежурный
администратор



Менеджер

Диаграмма сценариев: элементы

Интерфейс



Имя

Интерфейс определяет совокупность операций, которые обеспечивают необходимый набор сервисов для актера

Диаграмма сценариев: элементы

Примечание



Примечание предназначено для включения в модель произвольной текстовой информации, имеющей непосредственное отношение к контексту разрабатываемого проекта

Диаграмма сценариев: примечание

Пример

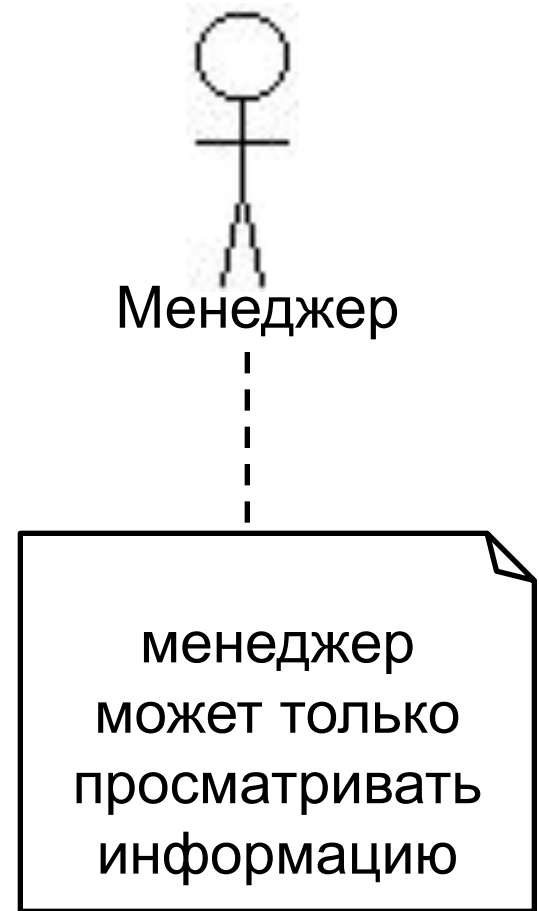


Диаграмма сценариев: отношения

- ❑ отношение ассоциации (association)
- ❑ отношение включения (include)
- ❑ отношение расширения (extend)
- ❑ отношение обобщения (generalization)

Диаграмма сценариев: ассоциация

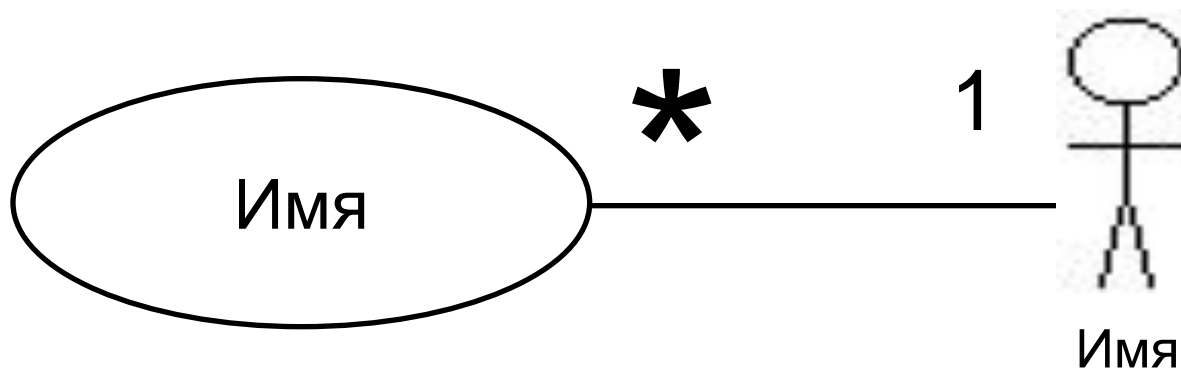


Диаграмма сценариев: ассоциация

Пример

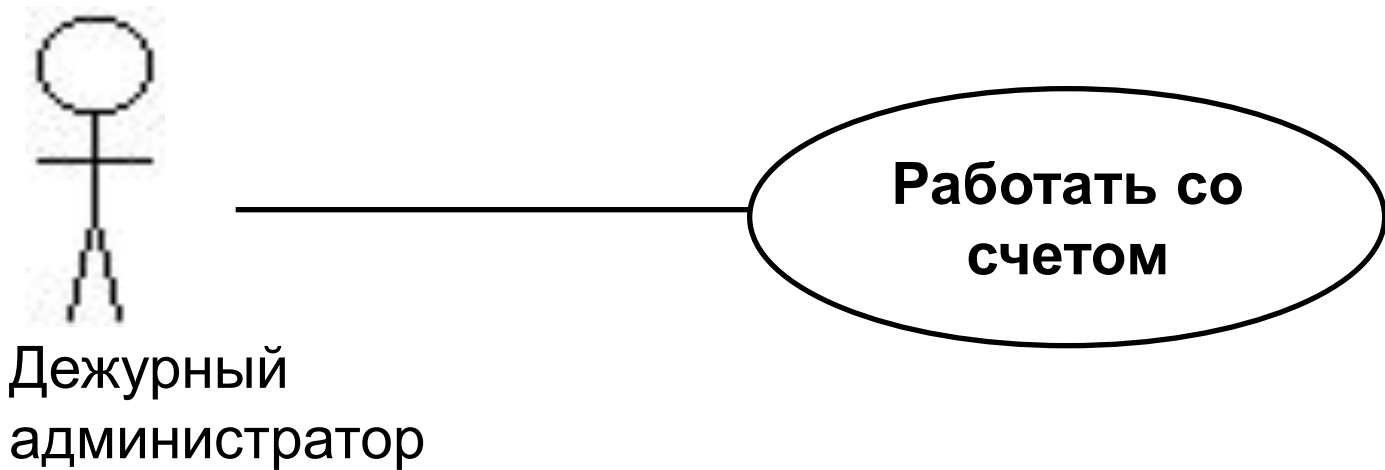
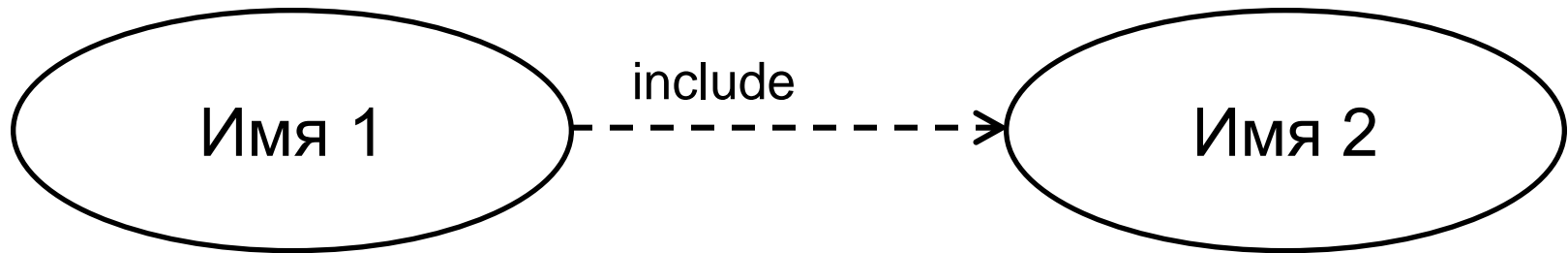


Диаграмма сценариев: включение



Сценарий 1 включает сценарий 2

Диаграмма сценариев: включение

Пример

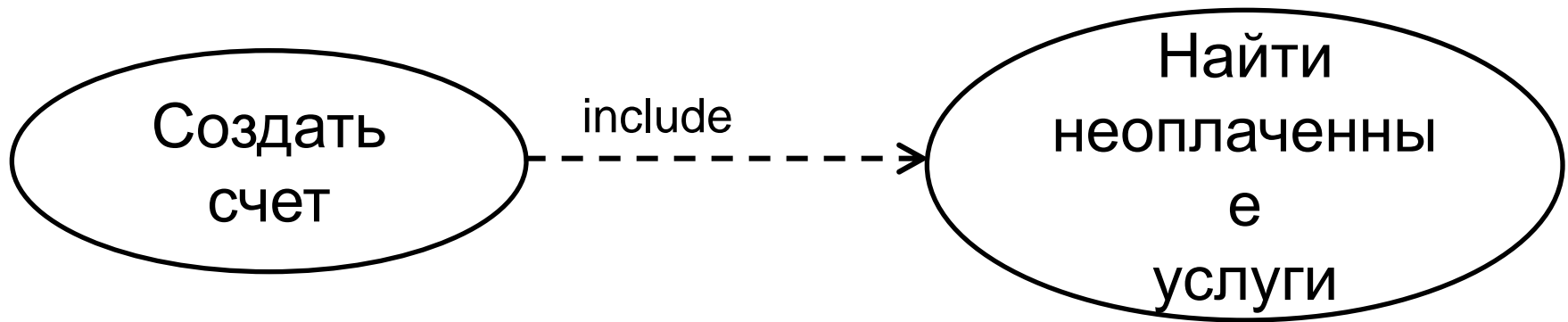
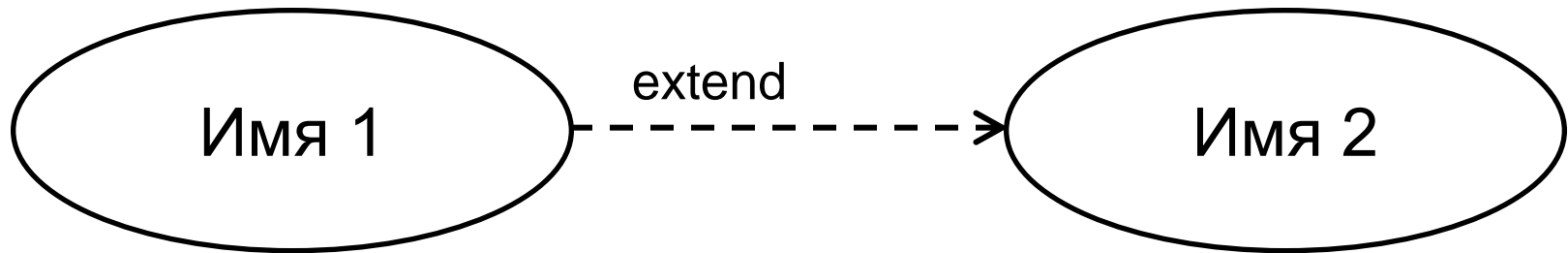


Диаграмма сценариев: расширение



Сценарий 1 расширяет сценарий 2

Диаграмма сценариев: расширение

Пример

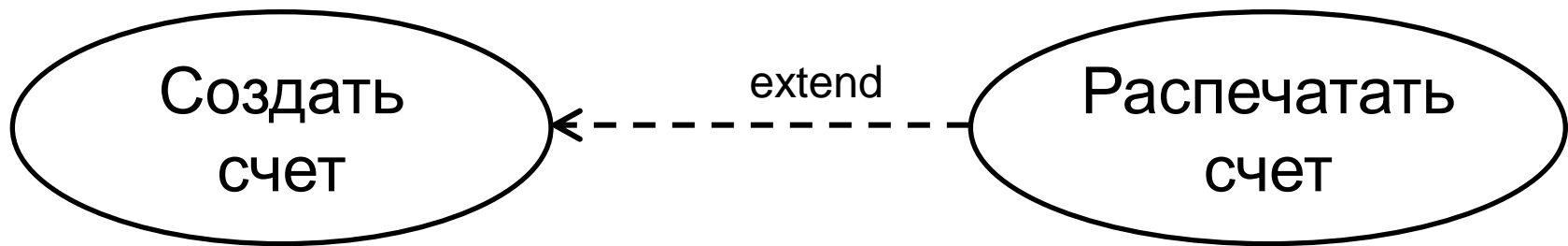
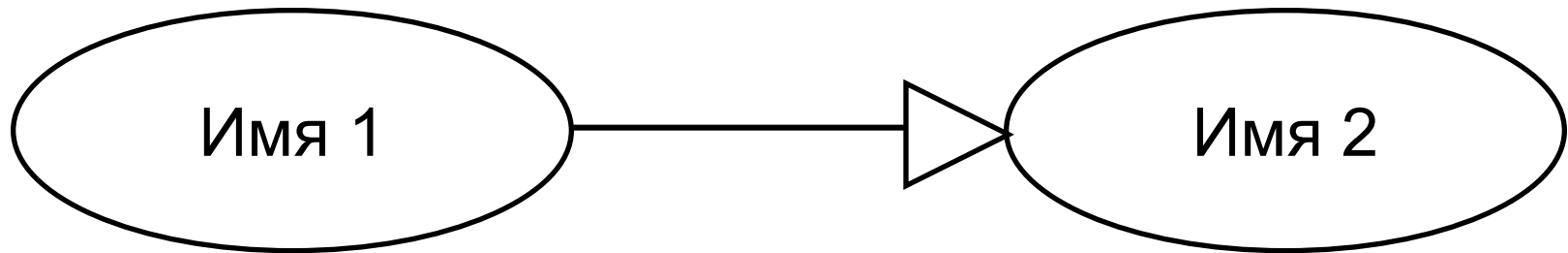


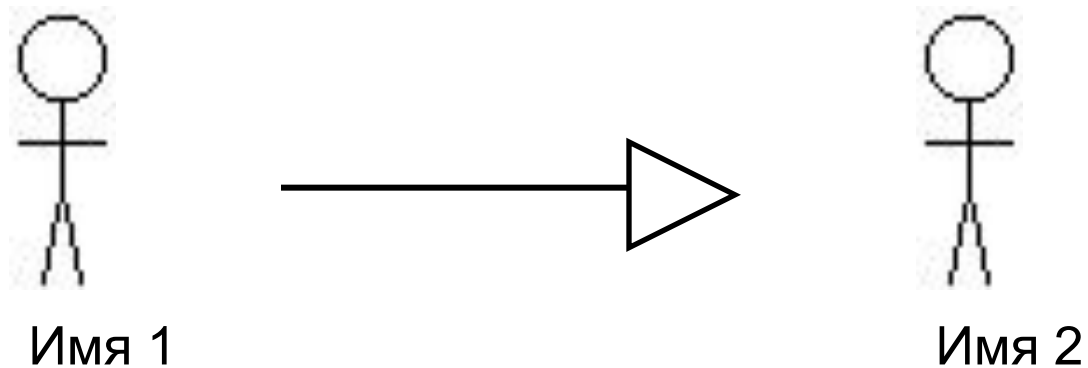
Диаграмма сценариев: обобщение



Сценарий 2 обобщает сценарий 1

Диаграмма сценариев: обобщение

Пример



Актер 2 обобщает Актера 1

Диаграмма сценариев: интерфейс

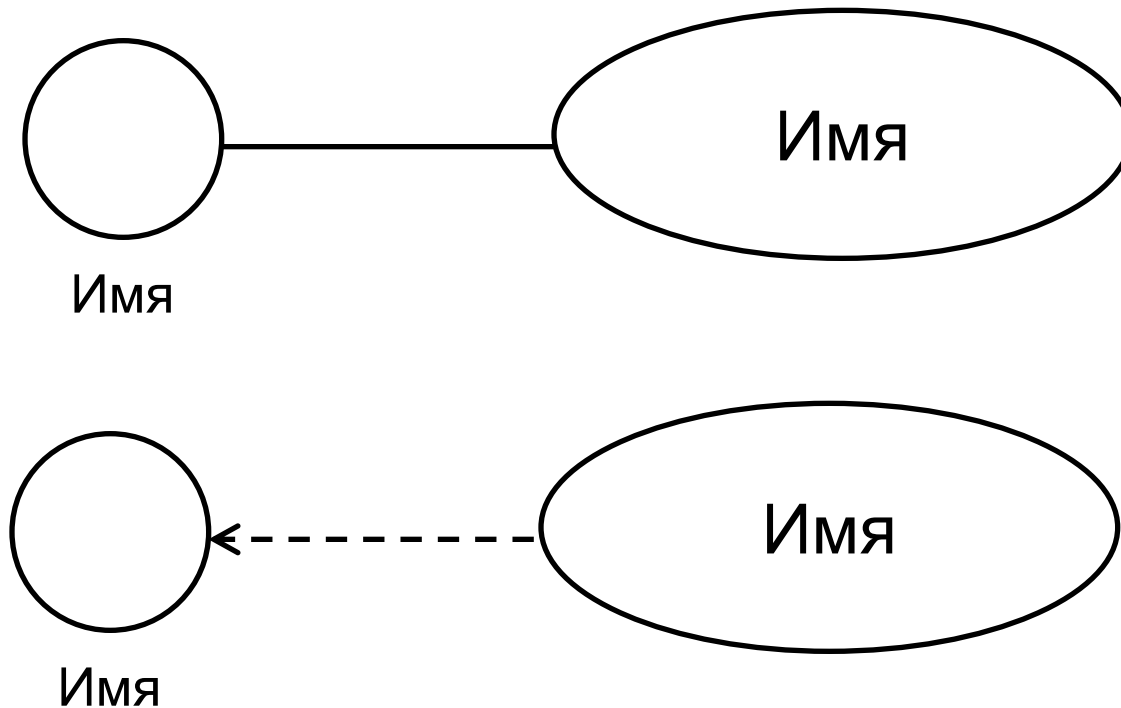


Диаграмма сценариев: интерфейс

Пример

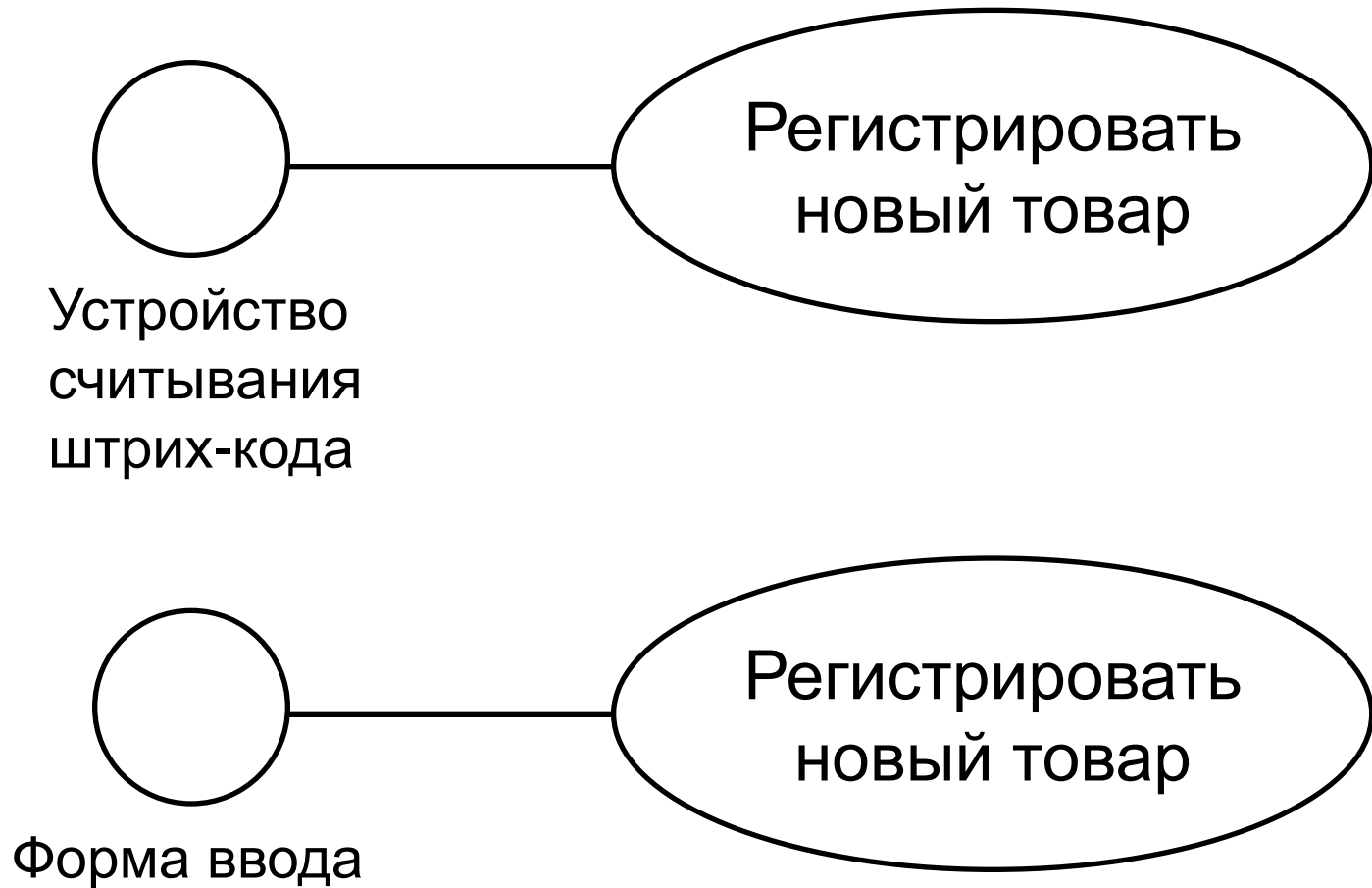


Диаграмма сценариев

Пример

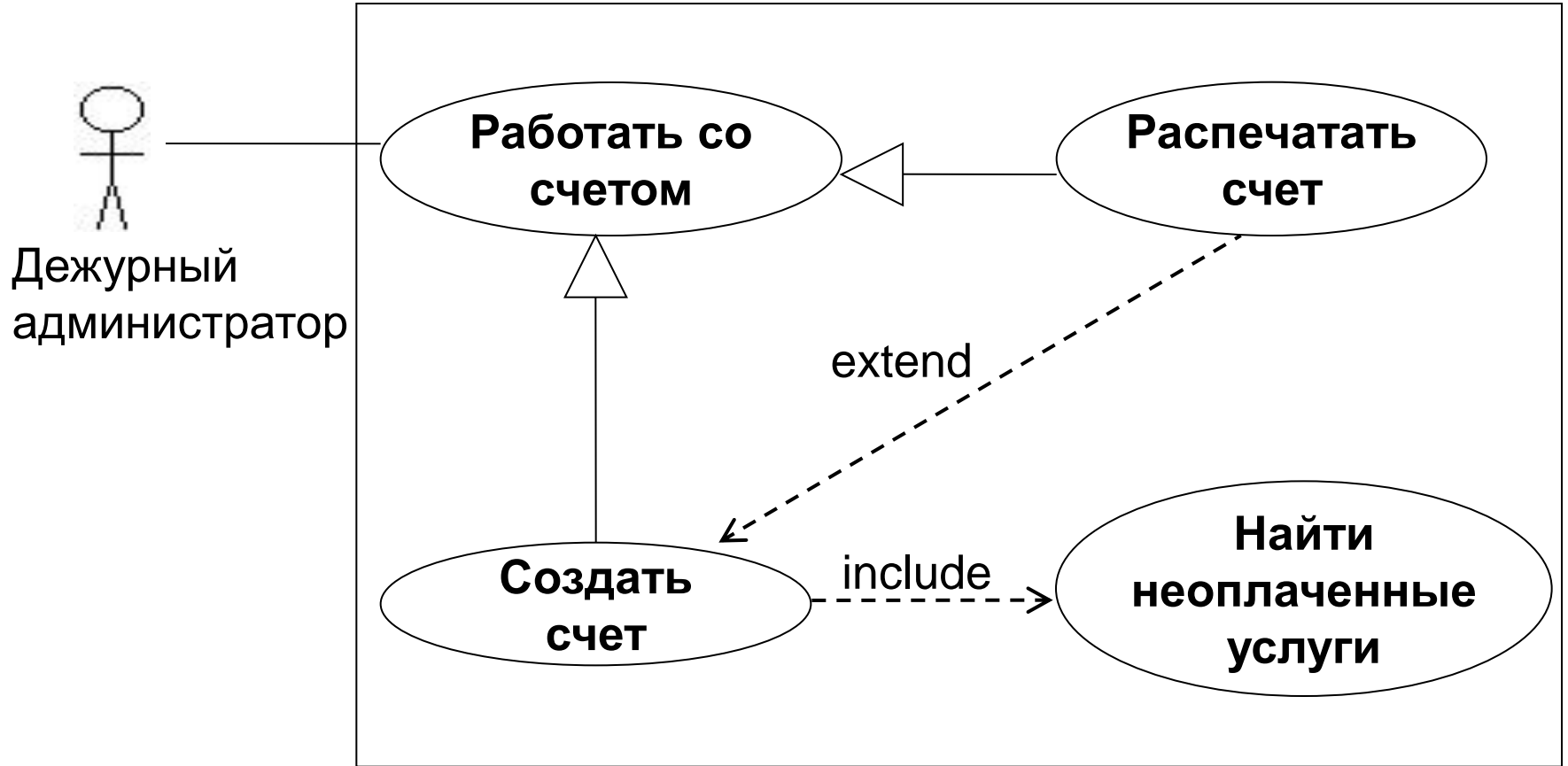


Диаграмма сценариев

Вопросы

- ❑ Что описывает диаграмма сценариев?
- ❑ Перечислить основные элементы диаграммы.
- ❑ Перечислить типы отношений на диаграмме сценариев.
- ❑ Привести примеры сценариев и отношений между сценариями.

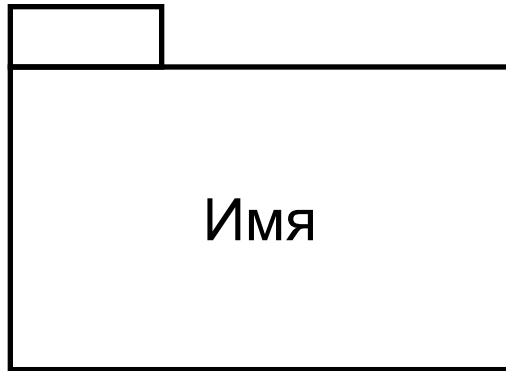
Задание

- Составить диаграммы сценариев для учебной ИС «Гостиница»

Диаграмма классов

- Диаграмма классов предназначена для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования

Диаграмма классов: элементы



Пакет

Пакет – способ организации элементов модели.
Каждый элемент модели принадлежит только одному пакету.

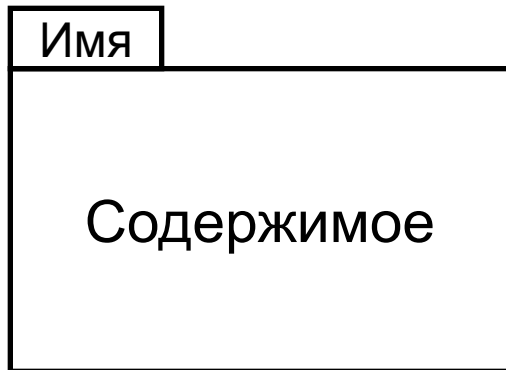


Диаграмма классов: пакет

Пример

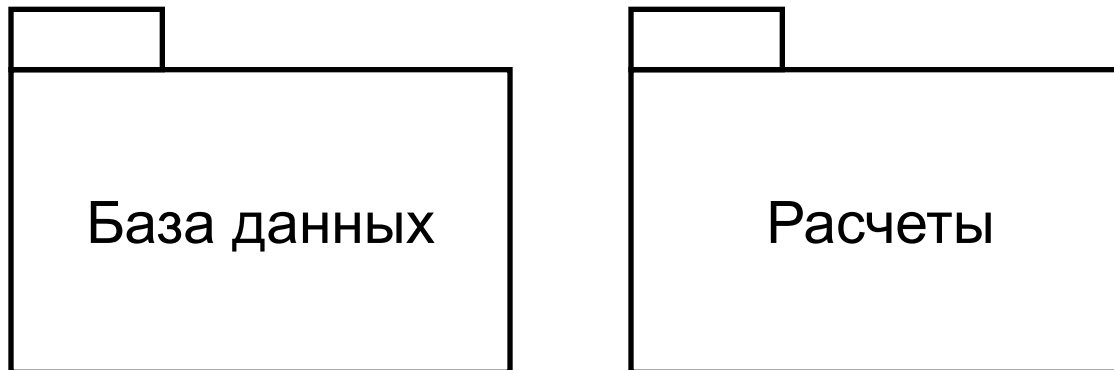


Диаграмма классов: элементы

Класс



Класс – обозначает множество объектов, которые обладают одинаковой структурой, поведением и отношениями с объектами из других классов

Диаграмма классов: элементы

Свойство

<квантор видимости> <имя> [<кратность>] :
<тип> = <исходное значение>

Диаграмма классов: свойство

<квантор видимости>

- «+» общедоступный (public) – атрибут доступен или виден из любого другого класса пакета, в котором определена диаграмма
- «#» защищенный (protected) – атрибут недоступен или невиден для всех классов, за исключением подклассов данного класса
- «–» закрытый (private) – атрибут недоступен или невиден для всех классов без исключения

Диаграмма классов: свойство

<кратность>

количество атрибутов данного типа, входящих в состав класса

записывается: [нижняя_граница1 .. верхняя_граница1, ...]

нижняя_граница и верхняя_граница являются положительными целыми числами

в качестве верхней_границы может использоваться специальный символ «*», который означает произвольное положительное целое число

Диаграмма классов: кратность

Пример

- [0..1] – кратность атрибута может принимать значение 0 или 1. При этом 0 означает отсутствие значения для данного атрибута
- [1..*] – кратность атрибута может принимать любое положительное целое значение
- [1..5] – кратность атрибута может принимать любое значение из чисел: 1, 2, 3, 4, 5.
- [1..3,5,7..*] – кратность атрибута может принимать любое значение из чисел: 1, 2, 3, 5, а также любое целое значение большее или равное 7

Диаграмма классов: свойство

<тип> – представляет собой выражение, семантика которого определяется языком спецификации модели

<исходное значение> – служит для задания некоторого начального значения для соответствующего атрибута в момент создания отдельного экземпляра класса

Диаграмма классов: свойство класса

Пример

- + color: RGB = (192, 192, 192)
- # navigable: boolean = TRUE
- + goal: enum(gTest, gWork) = gWork
- id: integer
- + name [1..2]: string

Диаграмма классов: элементы

Метод

<квантор видимости><имя>

(<список параметров>):

<тип возвращаемого значения>

Диаграмма классов: метод

<параметр>

**<вид><имя> : <тип> = <значение по
умолчанию>**

Диаграмма классов: метод

<вид>

in – входной параметр

out – выходной параметр

inout – одновременно входной и выходной
параметр

Диаграмма классов: метод класса

Пример

- + создать()
- + нарисовать(in форма: Многоугольник = прямоугольник, in цвет_заливки: Color = (0,0,255))
- запросить_счет_клиента(in номер_счета: integer): Currency

Диаграмма классов

Пример

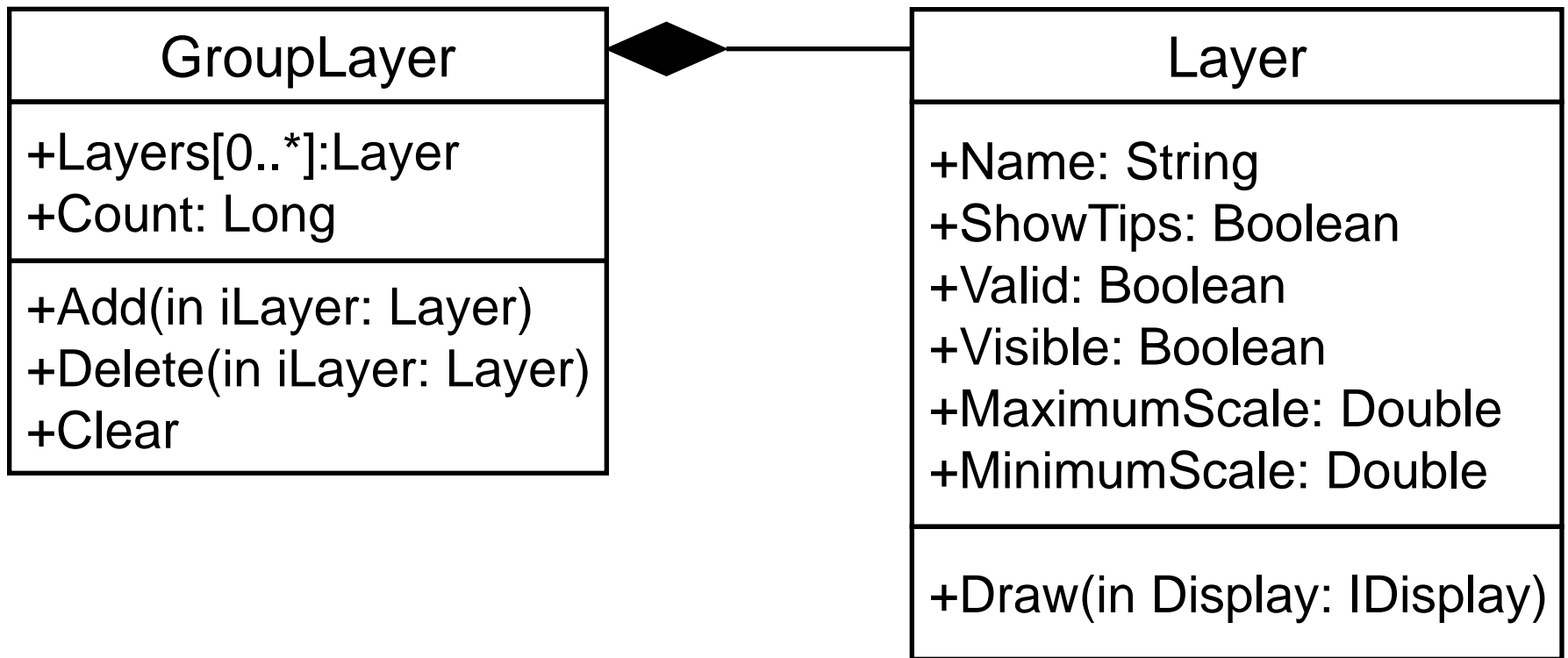


Диаграмма классов

Пример

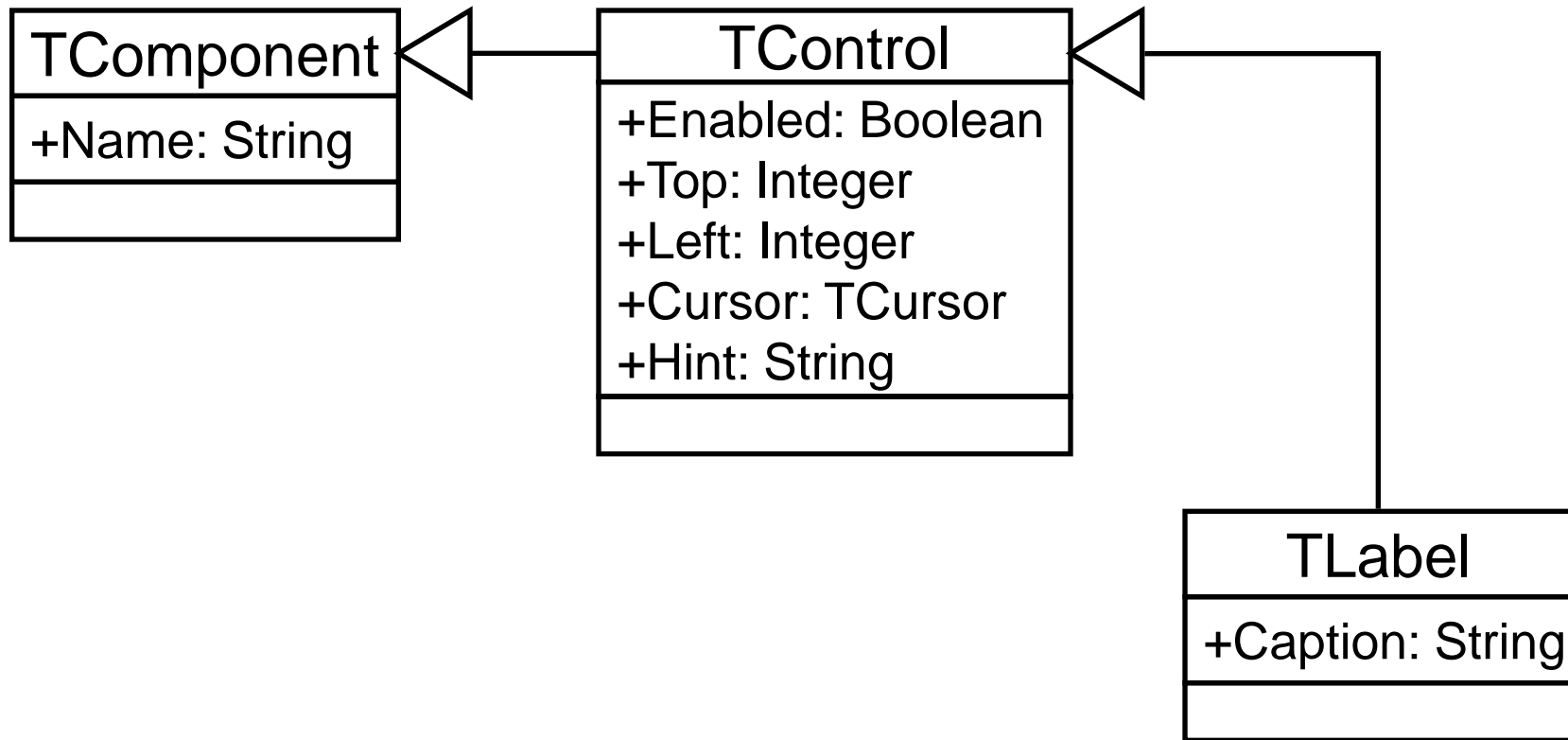
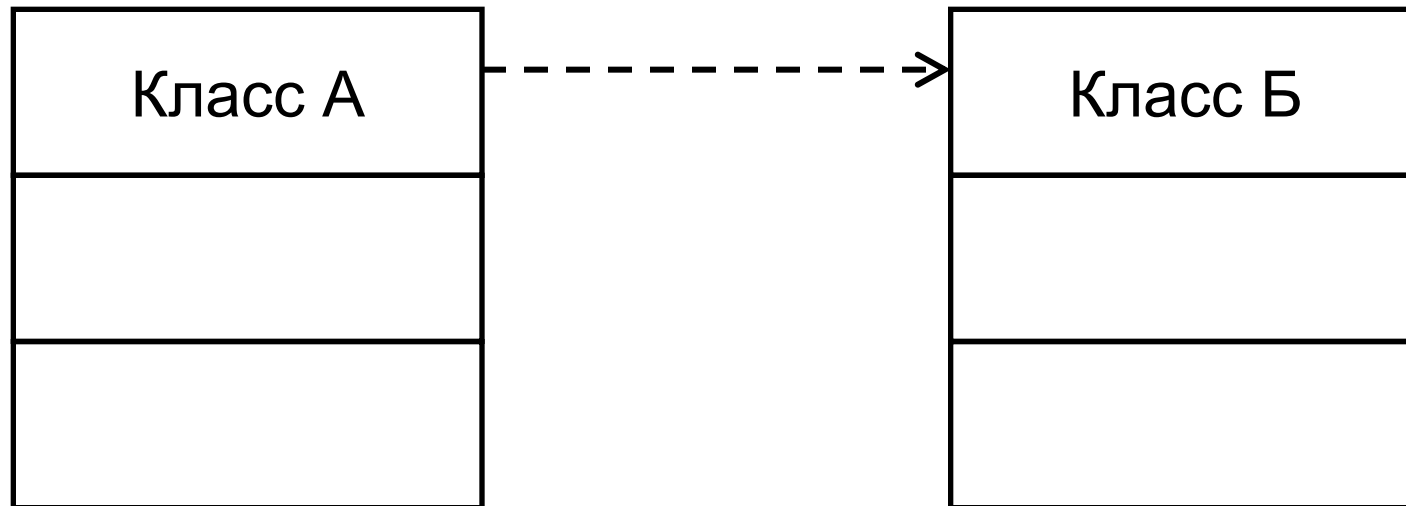


Диаграмма классов: отношения

- ❑ отношение зависимости (dependency)
- ❑ отношение ассоциации (association)
- ❑ отношение агрегации (aggregation)
- ❑ отношение композиции (composition)
- ❑ отношение обобщения (generalization)
- ❑ отношение реализации (realization)

Диаграмма классов: зависимость



Класс_А зависит от Класса_Б

Диаграмма классов: ассоциация



Диаграмма классов: ассоциация

Пример

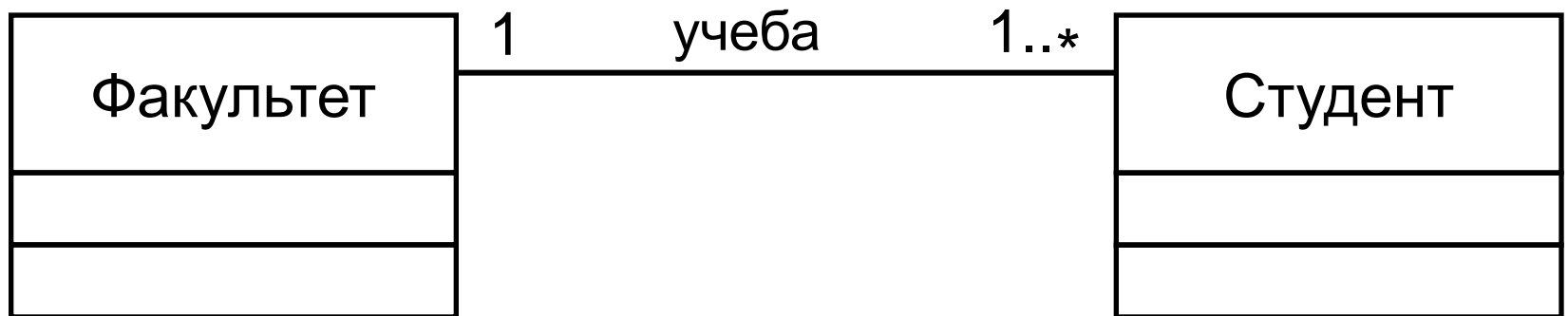


Диаграмма классов: ассоциация

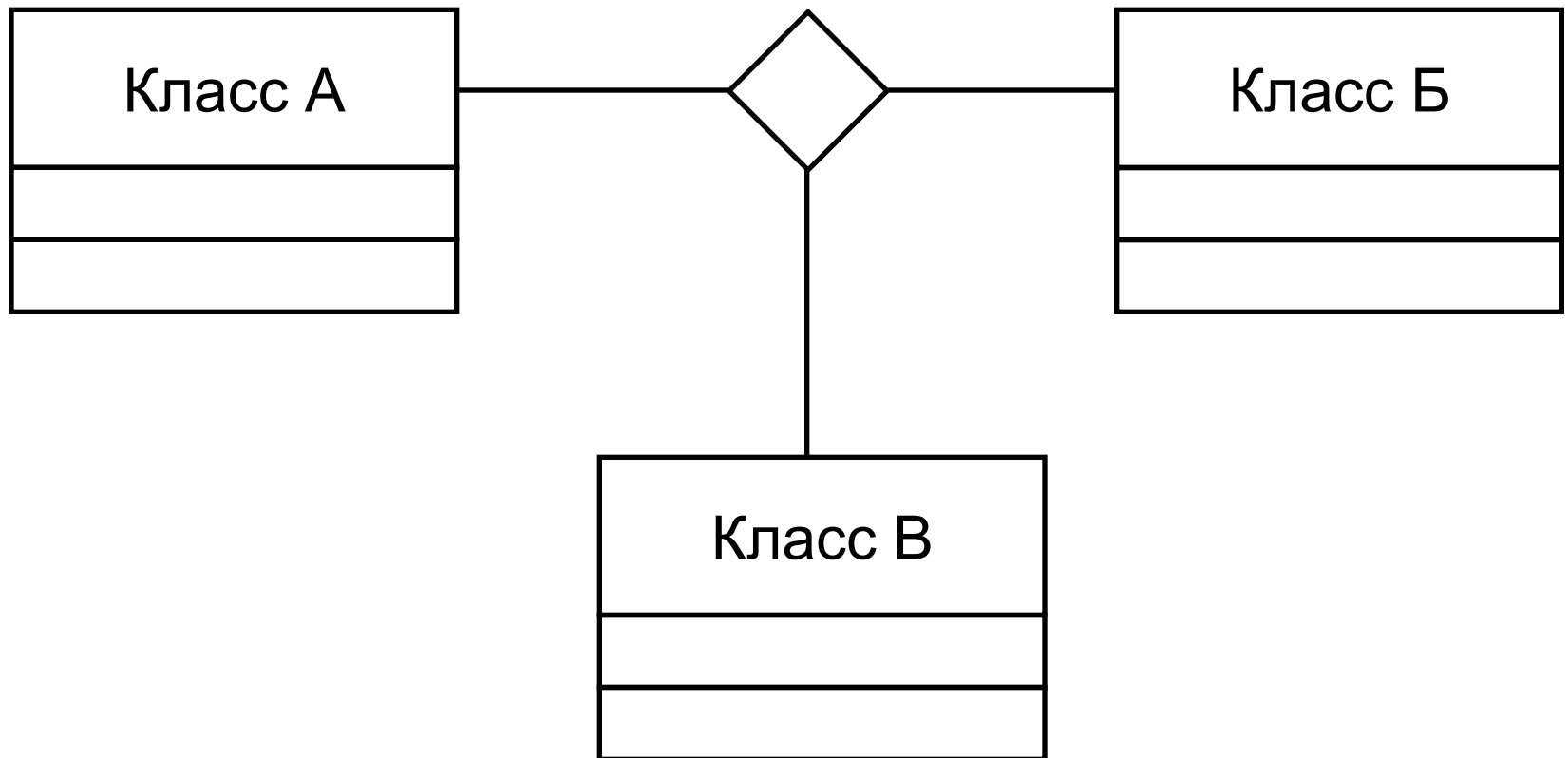


Диаграмма классов: ассоциация

Пример

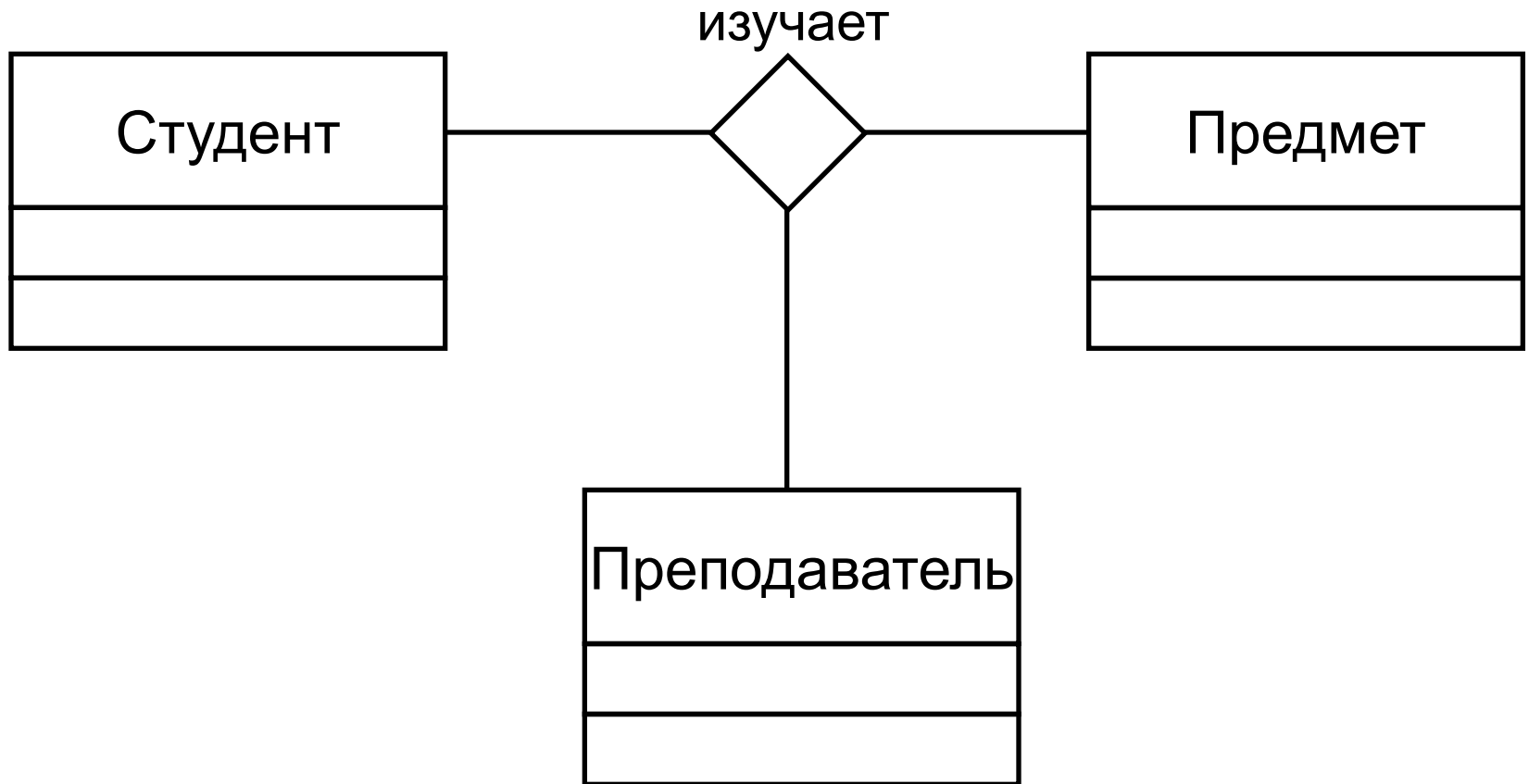


Диаграмма классов: агрегация

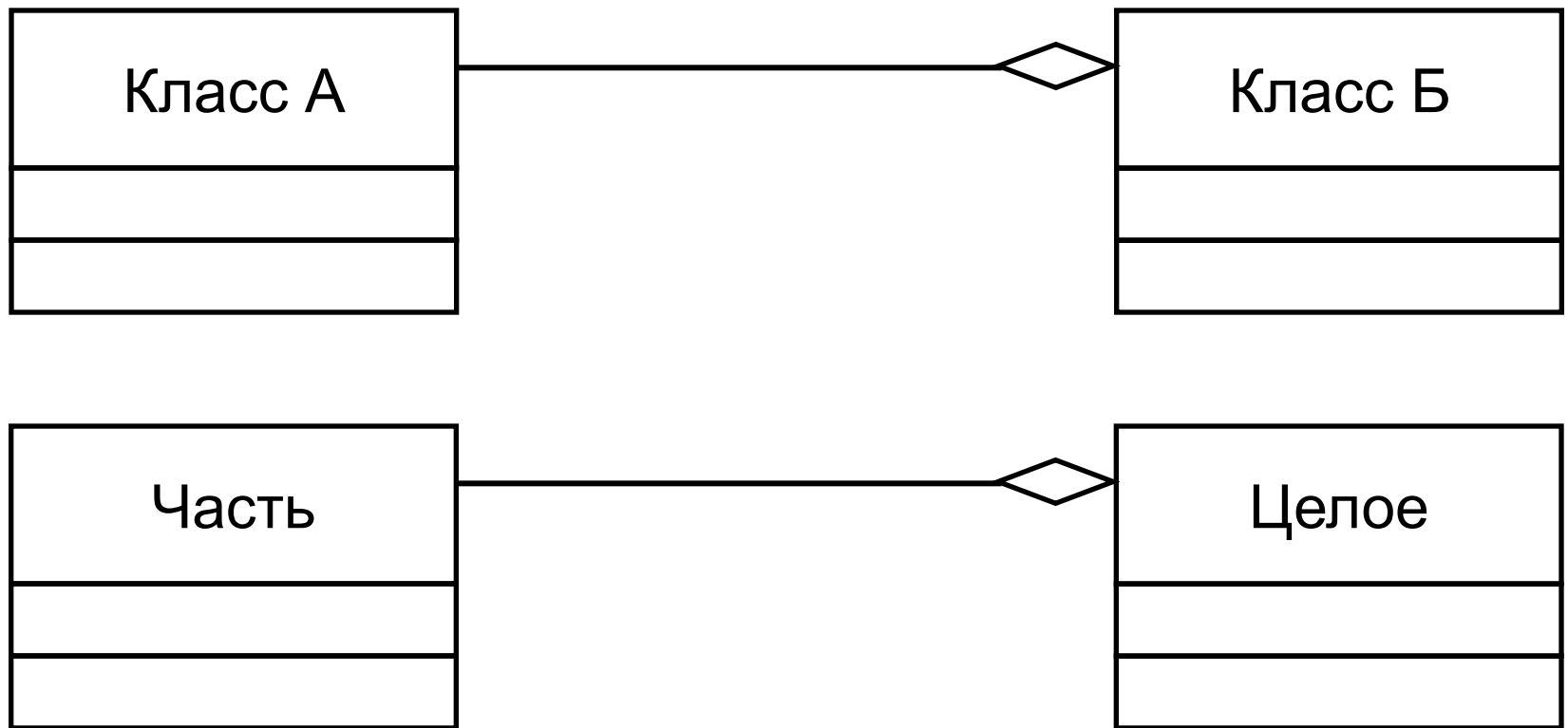


Диаграмма классов: агрегация

Пример

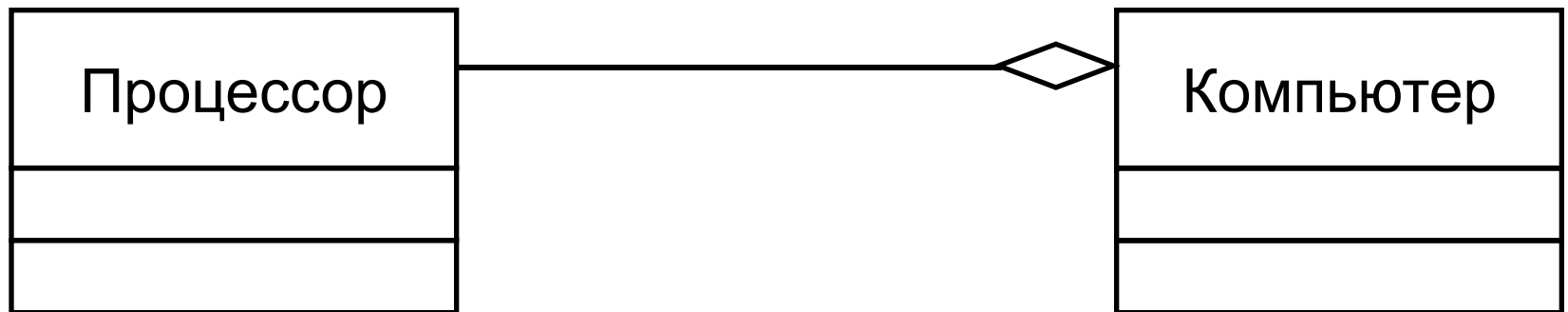


Диаграмма классов: композиция

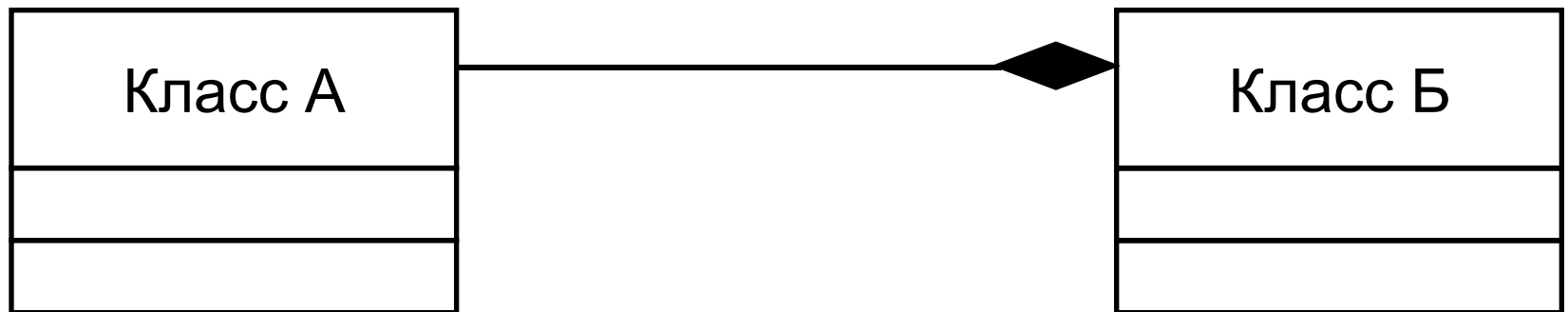


Диаграмма классов: композиция

Пример

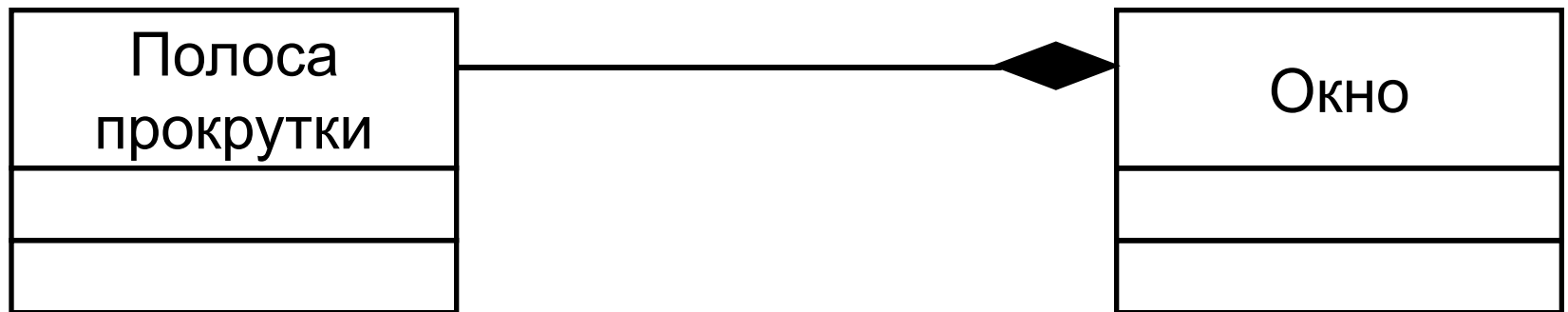


Диаграмма классов: обобщение

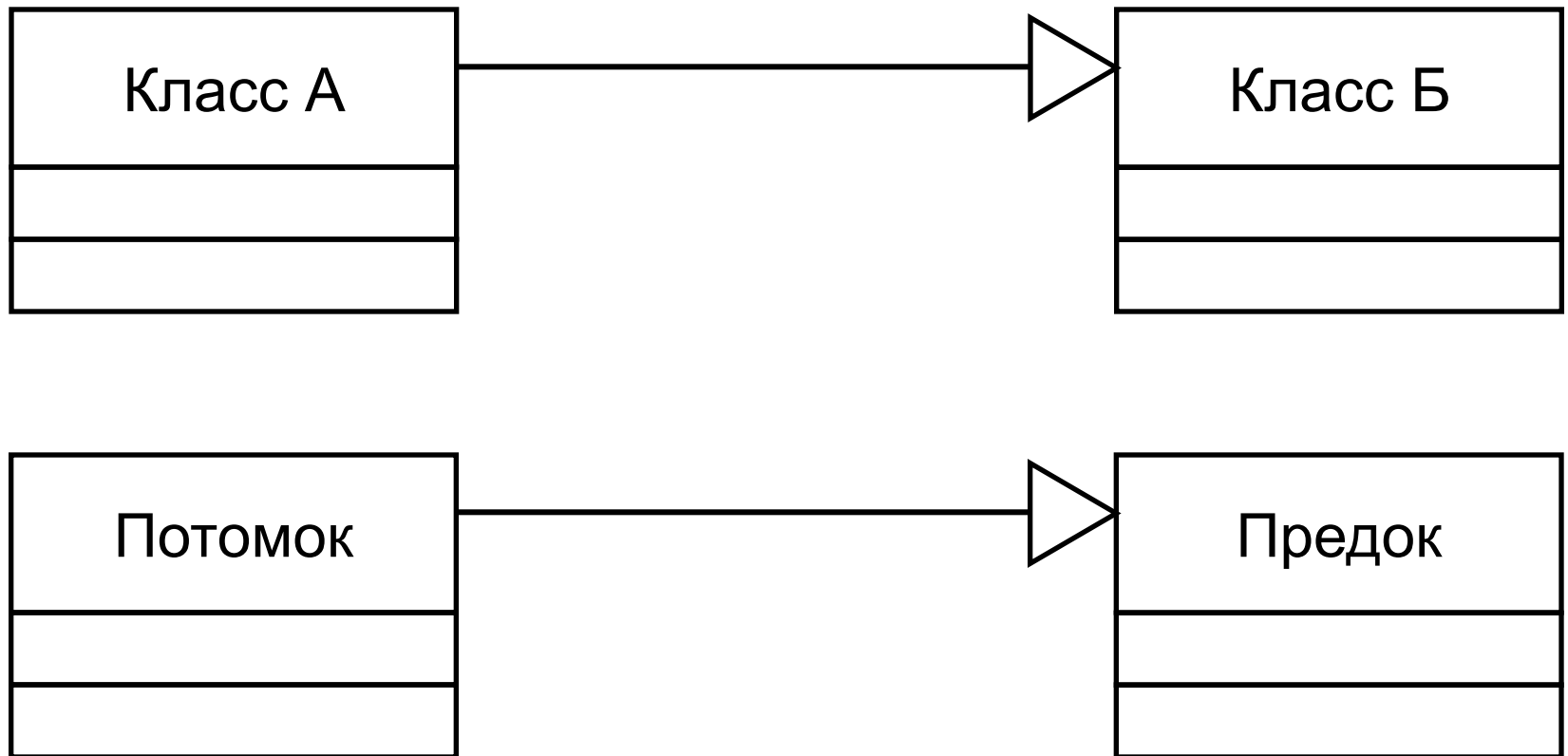


Диаграмма классов: обобщение

Пример

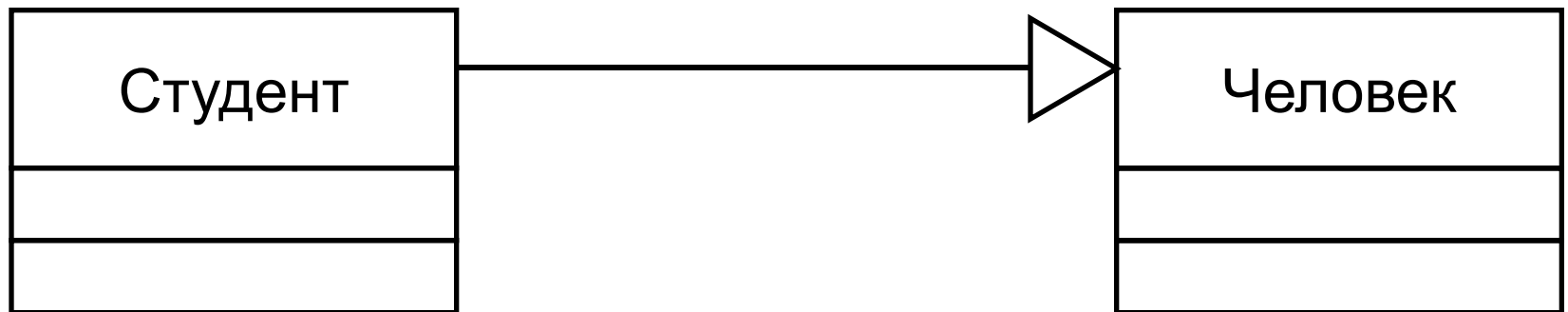


Диаграмма классов: элементы

Интерфейс



Интерфейс – набор операций, которые задают некоторые аспекты поведения класса и представляют его для других классов

Диаграмма классов: интерфейс

Пример

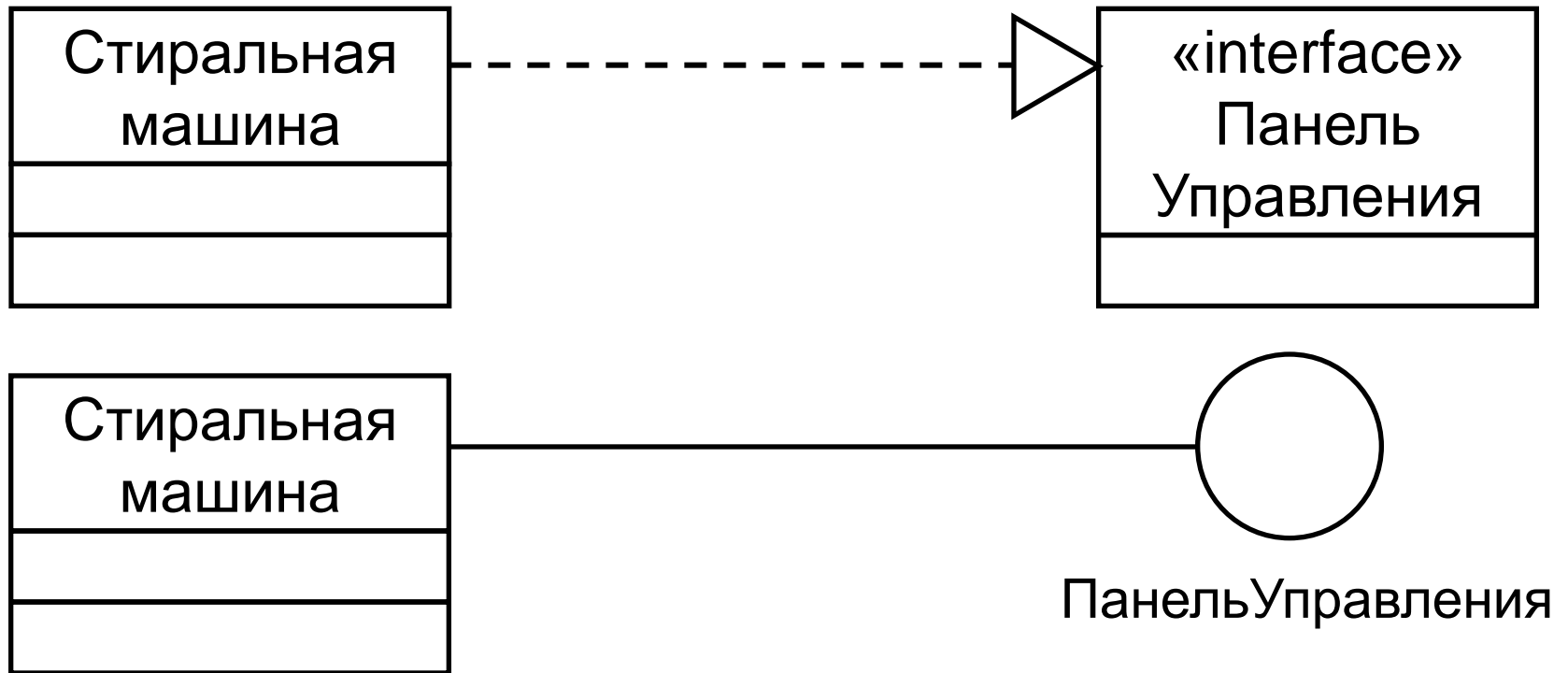


Диаграмма классов: интерфейс

Пример

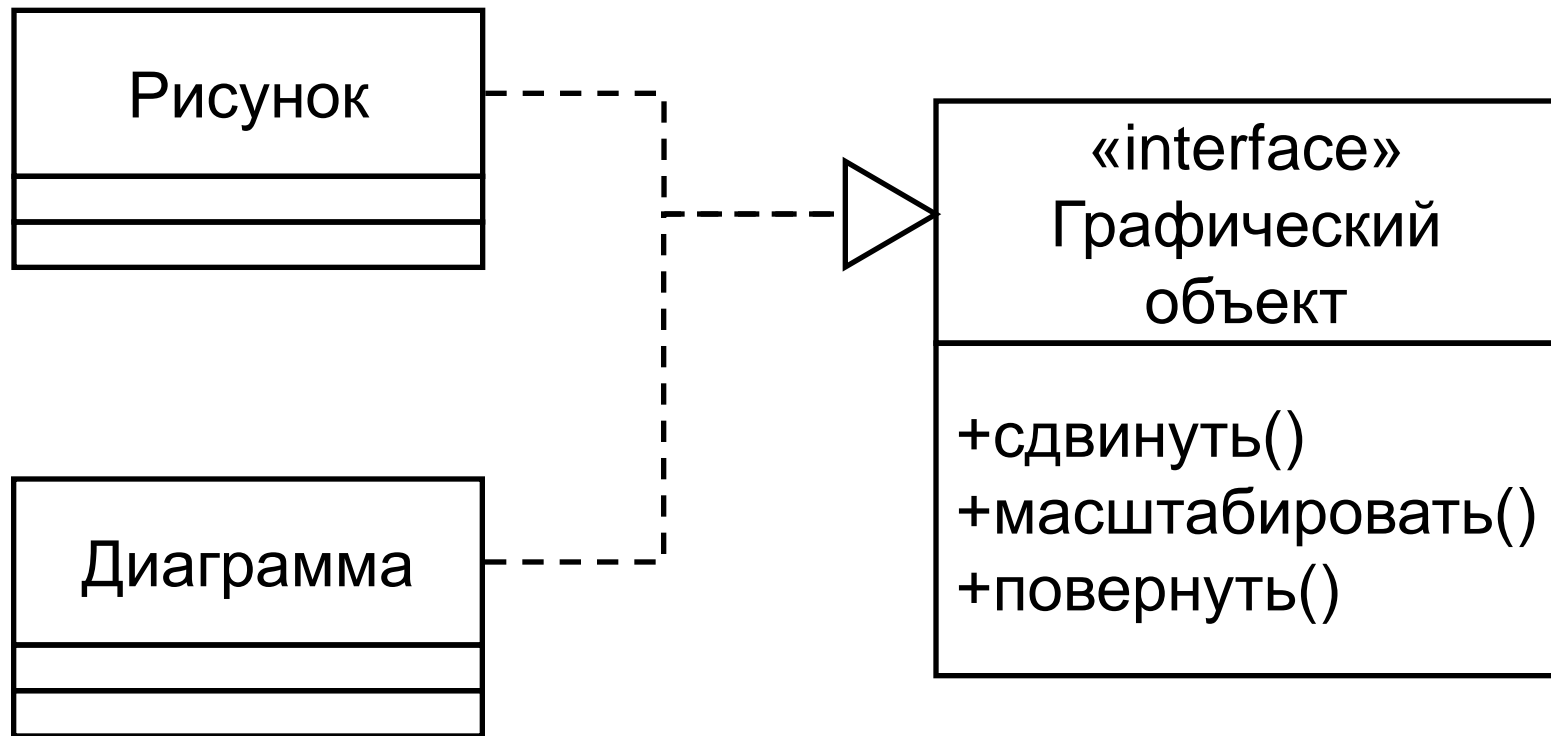
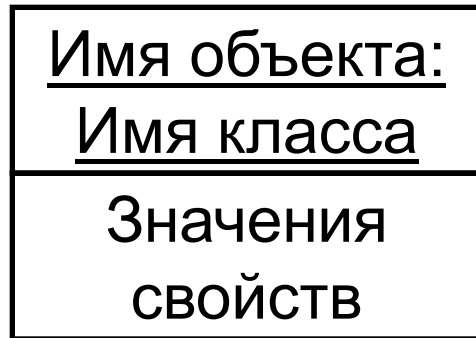


Диаграмма классов: элементы

Объект



Объект является отдельным экземпляром класса, который создается в процессе выполнения программы. Объект может иметь имя и конкретные значения свойств.

Диаграмма классов: объект

Пример



Диаграмма классов

Пример

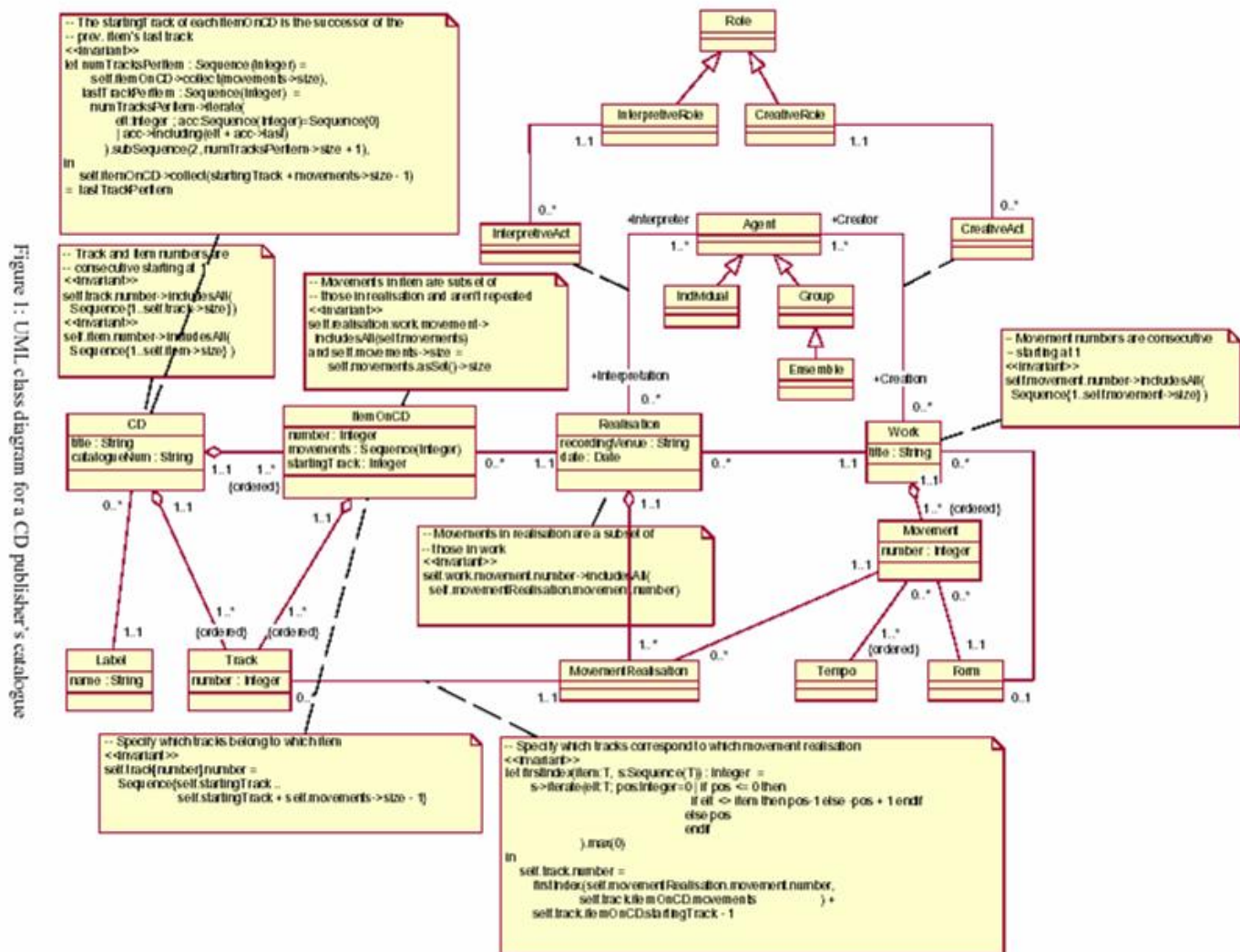


Диаграмма классов

Вопросы

- ❑ Что описывает диаграмма классов?
- ❑ Перечислить основные элементы диаграммы.
- ❑ Перечислить типы отношений.

Задание

- Спроектировать объектную базу данных для учебной ИС «Гостиница» и описать ее с помощью диаграммы классов

Диаграмма состояний: определение

- Диаграмма состояний описывает процесс изменения состояний только одного класса, а точнее – одного экземпляра класса, т. е. моделирует все возможные изменения в состоянии конкретного объекта
- Диаграмма состояний – конечный автомат

Диаграмма состояний: ограничения

- ❑ Переход из состояния в состояние происходит мгновенно
- ❑ История переходов из состояния в состояние не запоминается
- ❑ В каждый момент времени объект может находиться только в одном из своих состояний
- ❑ В любом состоянии объект может находиться как угодно долго
- ❑ Время на диаграмме состояний присутствует в неявном виде
- ❑ Количество состояний должно быть обязательно конечным
- ❑ Не должно быть изолированных состояний и переходов
- ❑ Не должно быть конфликтующих переходов

Диаграмма состояний: элементы

Состояние

Состояние – набор конкретных значений атрибутов объекта

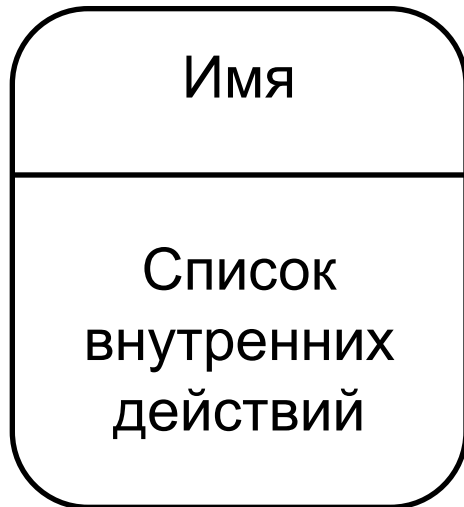


Диаграмма состояний: состояние

Действие

<метка> / <выражение действия>

<Метка>

entry – вход в состояние

exit – выход из состояния

do – деятельность в состоянии

include – вызов подавтомата

Диаграмма состояний: состояние

Пример

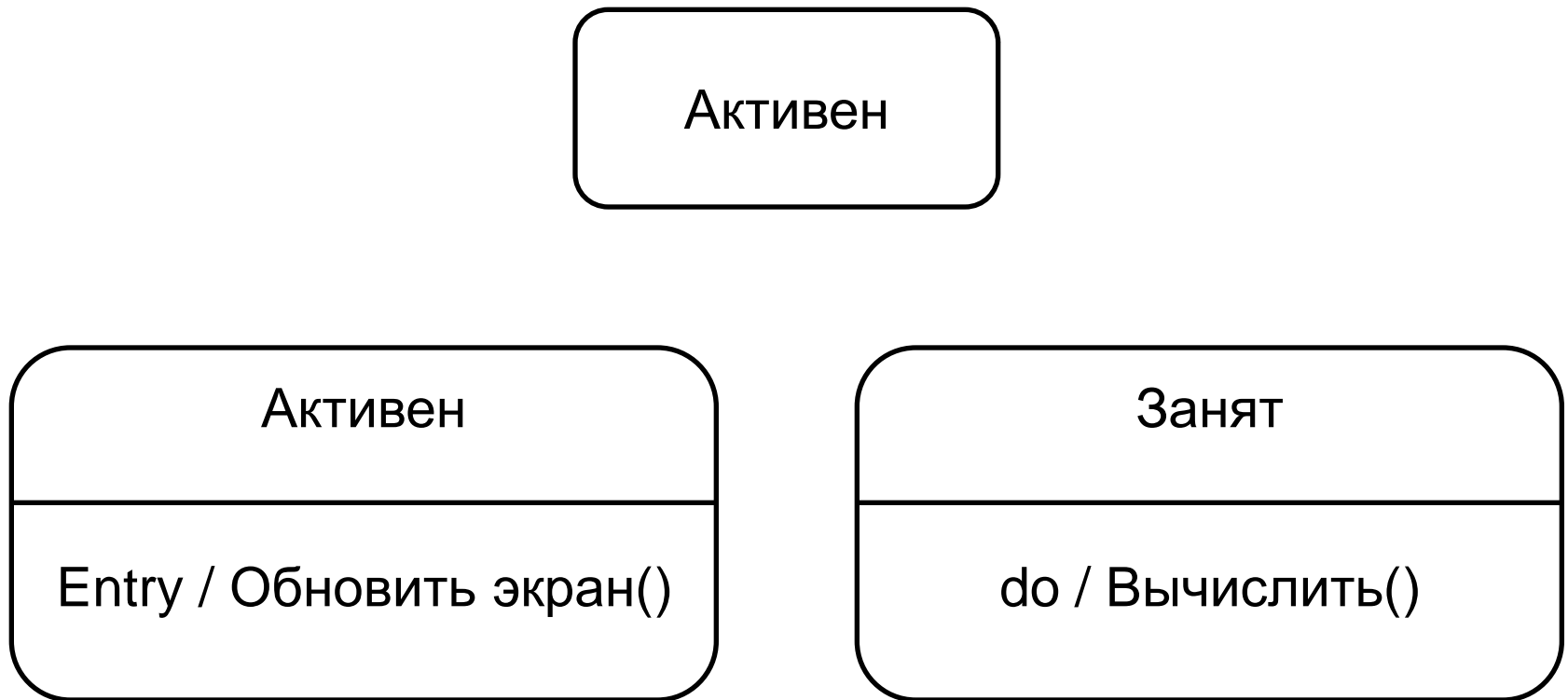
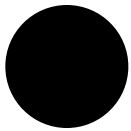
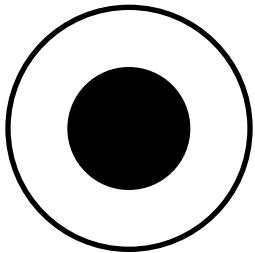


Диаграмма состояний: элементы



Начальное состояние



Конечное состояние

Диаграмма состояний: элементы

Переход



Переход осуществляется при наступлении некоторого события

Диаграмма состояний: переход

<Метка>

<сигнатура события>
[<сторожевое условие>]
/ <выражение действия>

Диаграмма состояний: метка

<сигнатура события>

<имя события> (<список параметров>)


[<сторожевое условие>]

– булевское выражение

Диаграмма состояний: переход

Пример

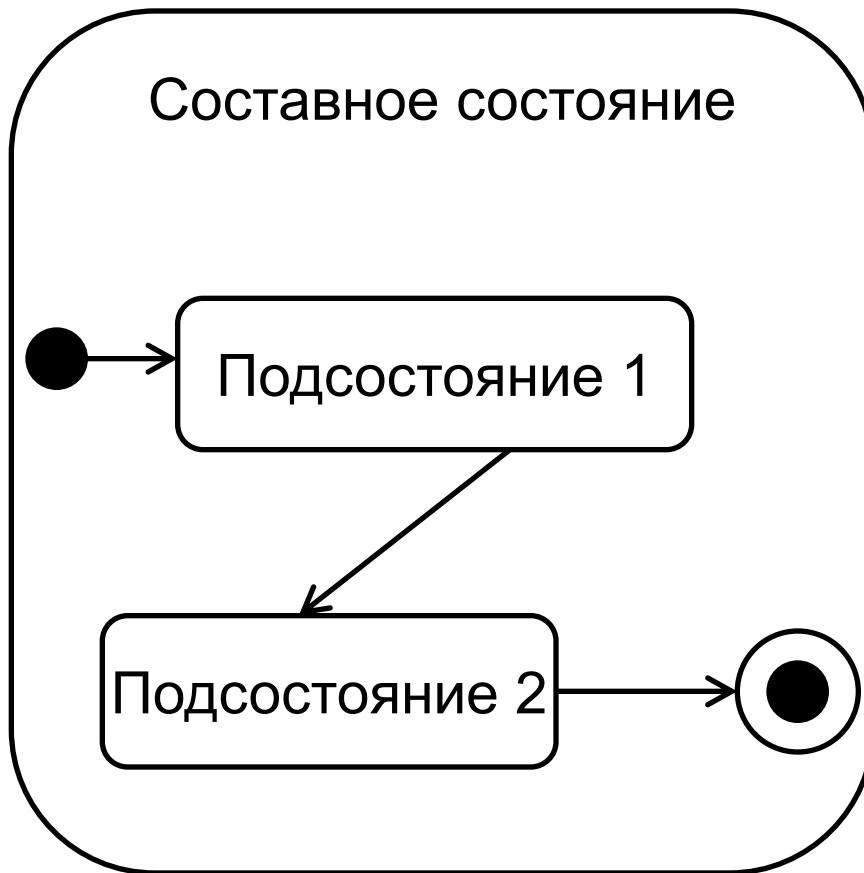
Нажатие клавиши (Клавиша) [Клавиша = «Свернуть»]



Получение сигнала / Установить соединение()



Диаграмма состояний: элементы



Составное состояние

Составное состояние
состоит из вложенных
в него подсостояний

Диаграмма состояний

Пример

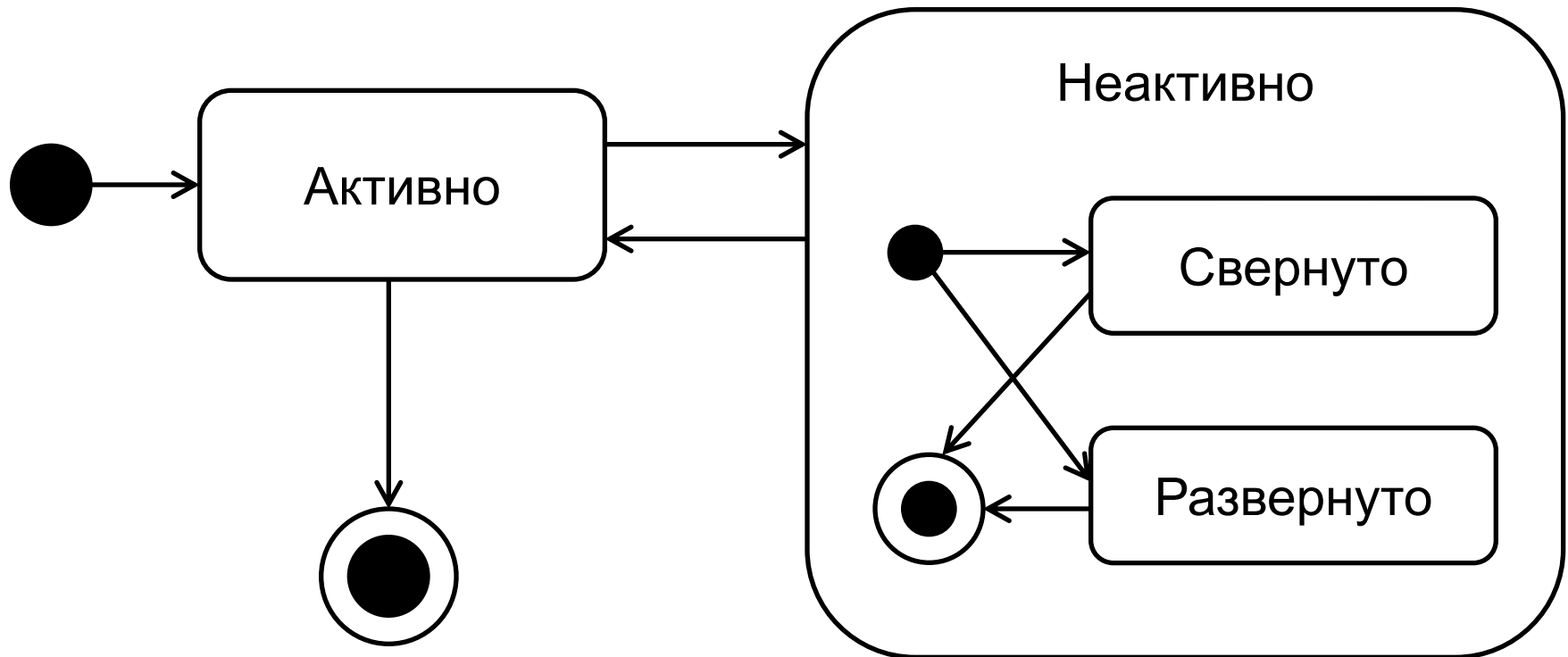


Диаграмма состояний

Вопросы

- ❑ Что описывает диаграмма состояний?
- ❑ Что такое состояние?
- ❑ Перечислить основные элементы диаграммы.

Задание

- Построить диаграмму состояний для класса «Карта брони» учебной ИС «Гостиница»

Диаграмма деятельности: определение

- Диаграмма деятельности описывает процесс выполнения действий, т.е. логику или последовательность перехода от одного действия к другому
- Диаграмма деятельности используется для моделирования бизнес-процессов

Диаграмма деятельности: элементы

Действие



Действие – операция, выражение, вычисления и т.д.

Диаграмма деятельности: действие

Пример

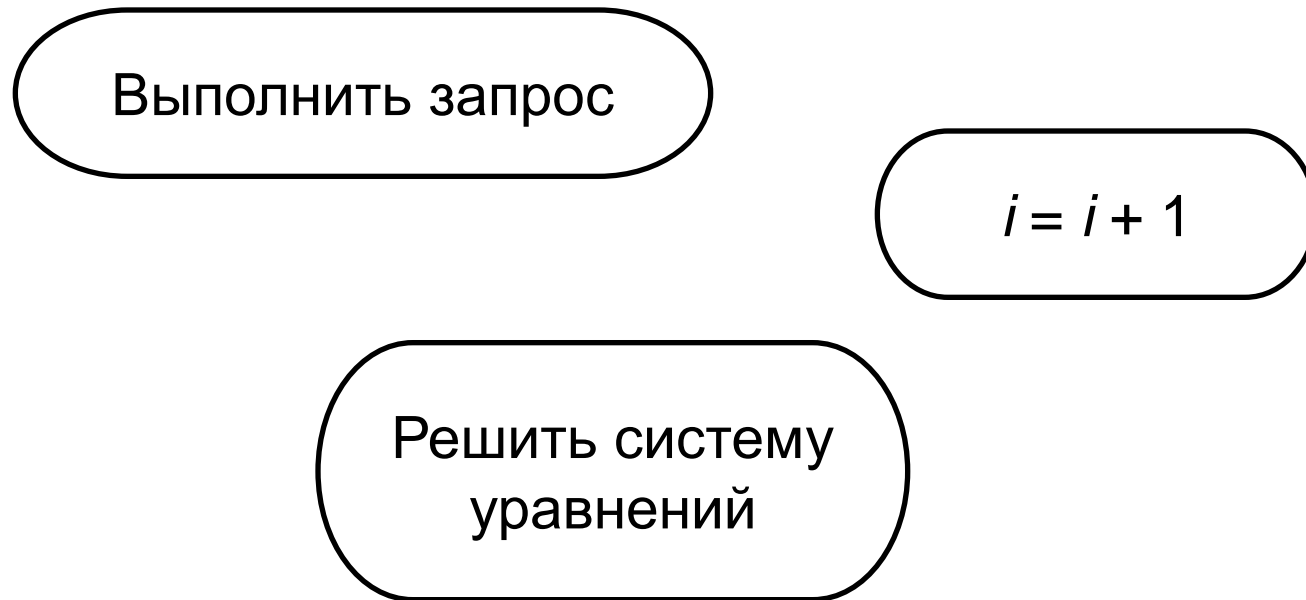
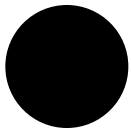
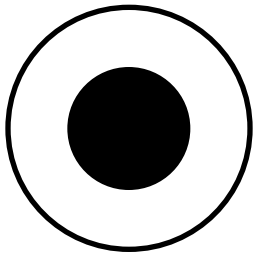


Диаграмма деятельности: элементы



Начало алгоритма



Конец алгоритма

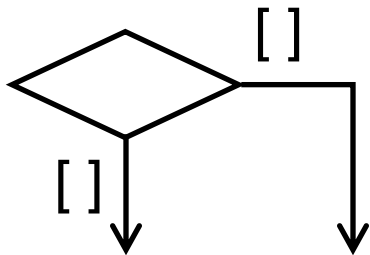
Диаграмма деятельности: элементы

Переход



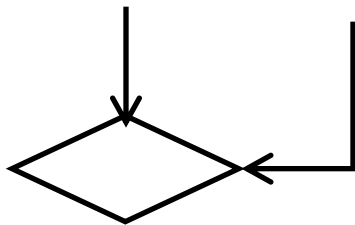
Переход срабатывает сразу
после завершения действия

Диаграмма деятельности: элементы



Ветвление

Ветвление – разделение на альтернативные ветви.



Соединение

Соединение – объединение альтернативных ветвей.

Диаграмма деятельности

Пример

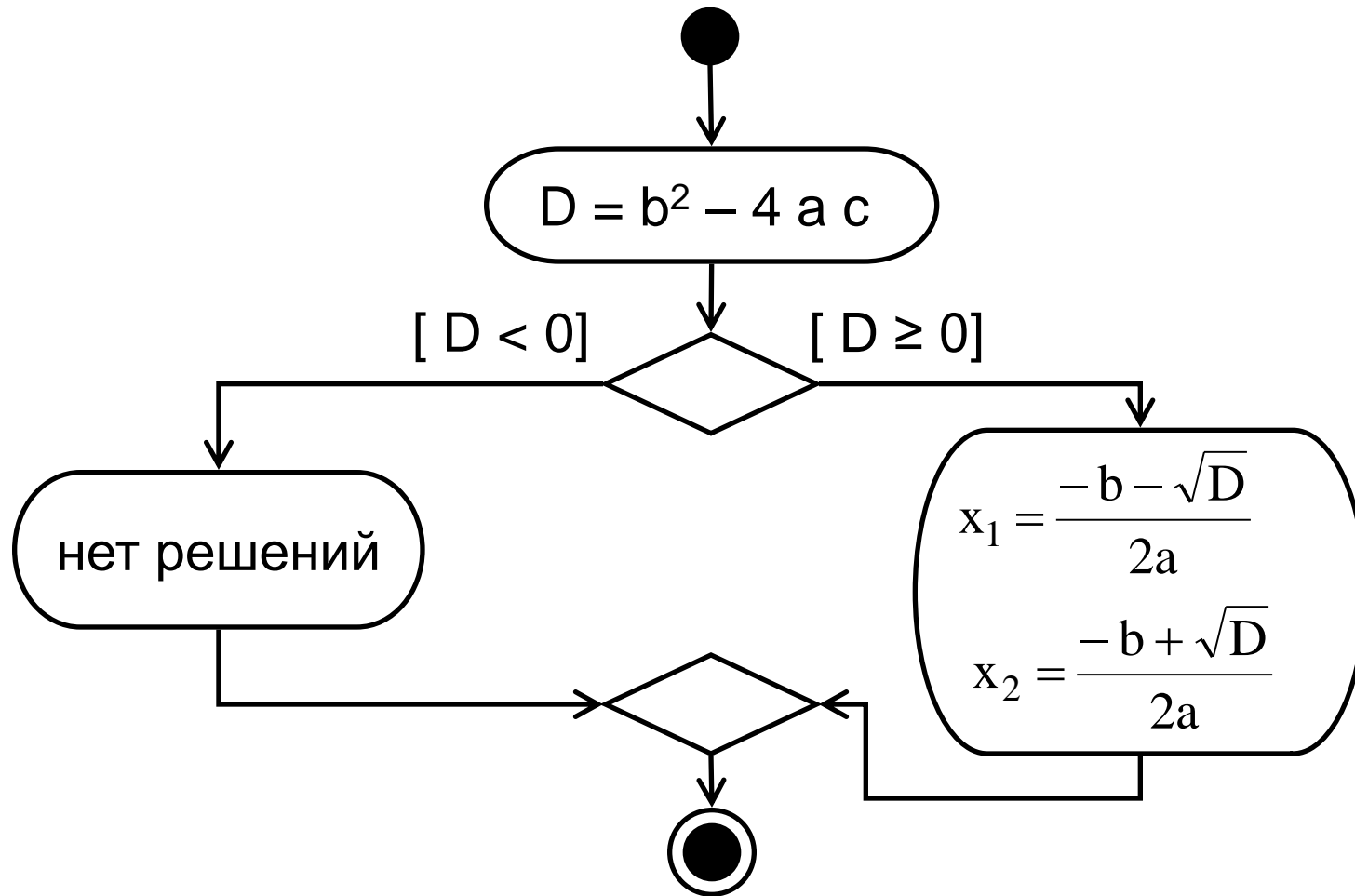
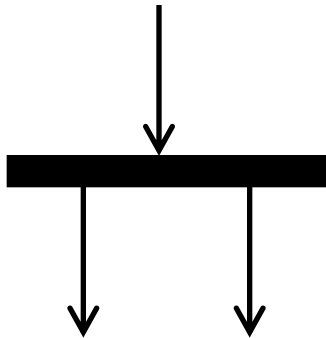
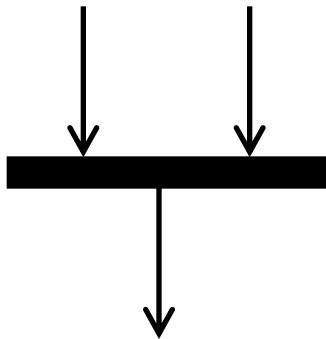


Диаграмма деятельности: элементы



Разделение

Разделение –
распараллеливание действий



Согласование

Согласование – переход к
следующему действию после
окончания всех согласуемых
действий

Задание

- Построить диаграмму деятельности для процесса «Приготовление напитка (с выбором из двух напитков)»

Диаграмма деятельности: элементы

Дорожка

Дорожка обозначает
исполнителя действий

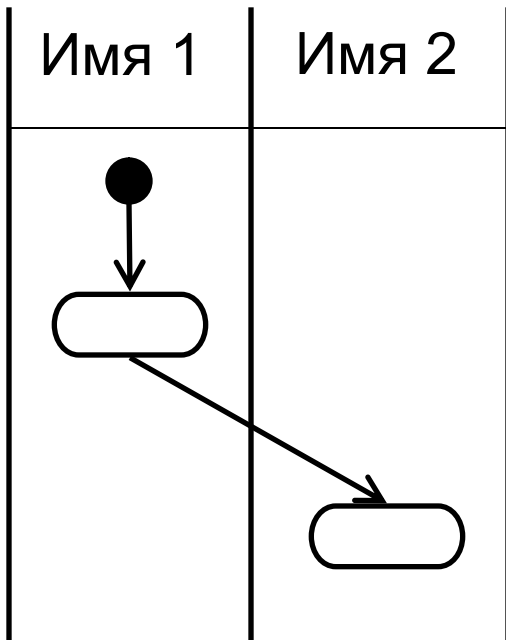
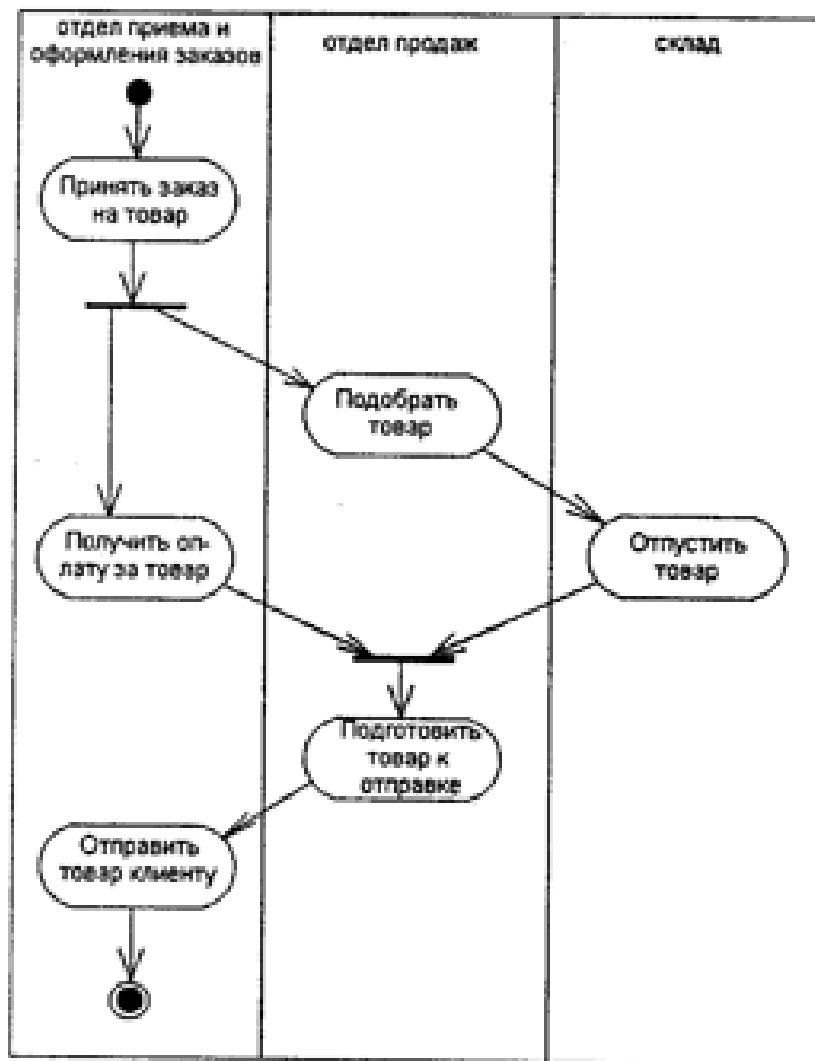


Диаграмма деятельности

Пример



Задание

- ❑ Построить диаграмму деятельности, описывающую алгоритм поиска минимального элемента в массиве
- ❑ Построить диаграмму деятельности, описывающую бизнес-процесс «Поселение клиента в гостинице»
- ❑ Построить диаграмму деятельности, описывающую алгоритм работы метода одного из классов учебной ИС «Гостиница»

Диаграмма деятельности

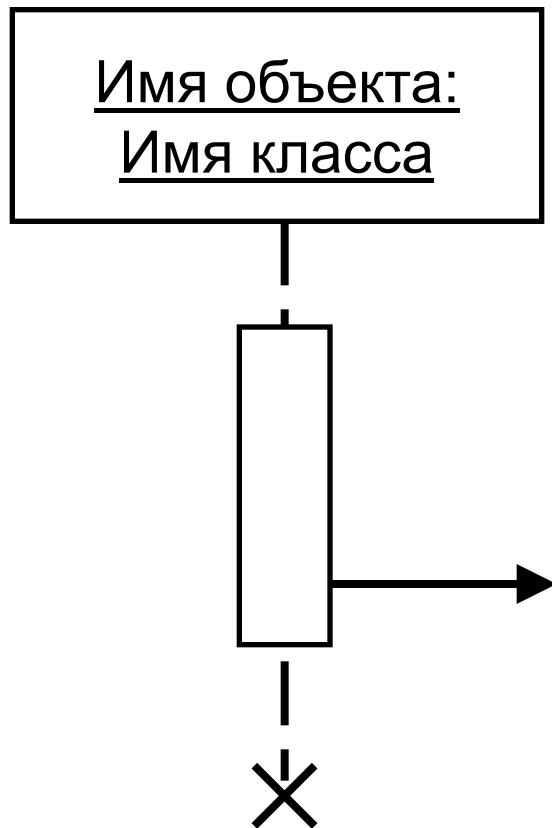
Вопросы

- ❑ Что описывает диаграмма деятельности?
- ❑ Перечислить основные элементы диаграммы.

Диаграмма последовательности: определение

- Диаграмма последовательности используется для представления временных особенностей передачи и приема сообщений между объектами

Диаграмма последовательности: элементы



Элементы

- ❑ Объект
- ❑ Линия жизни
- ❑ Фокус управления
- ❑ Сообщение
- ❑ Уничтожение объекта

Диаграмма последовательности: элементы

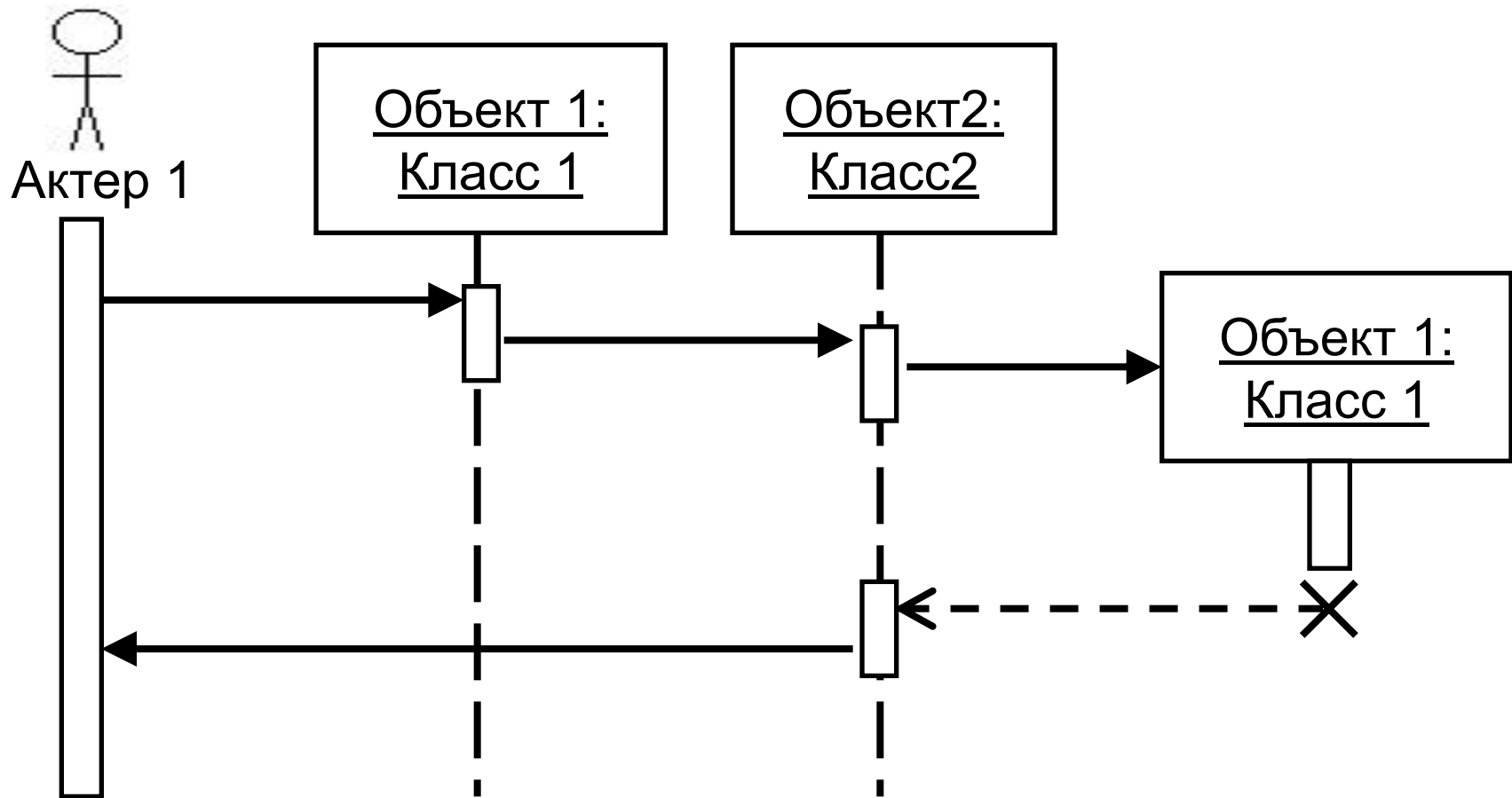


Диаграмма последовательности: элементы

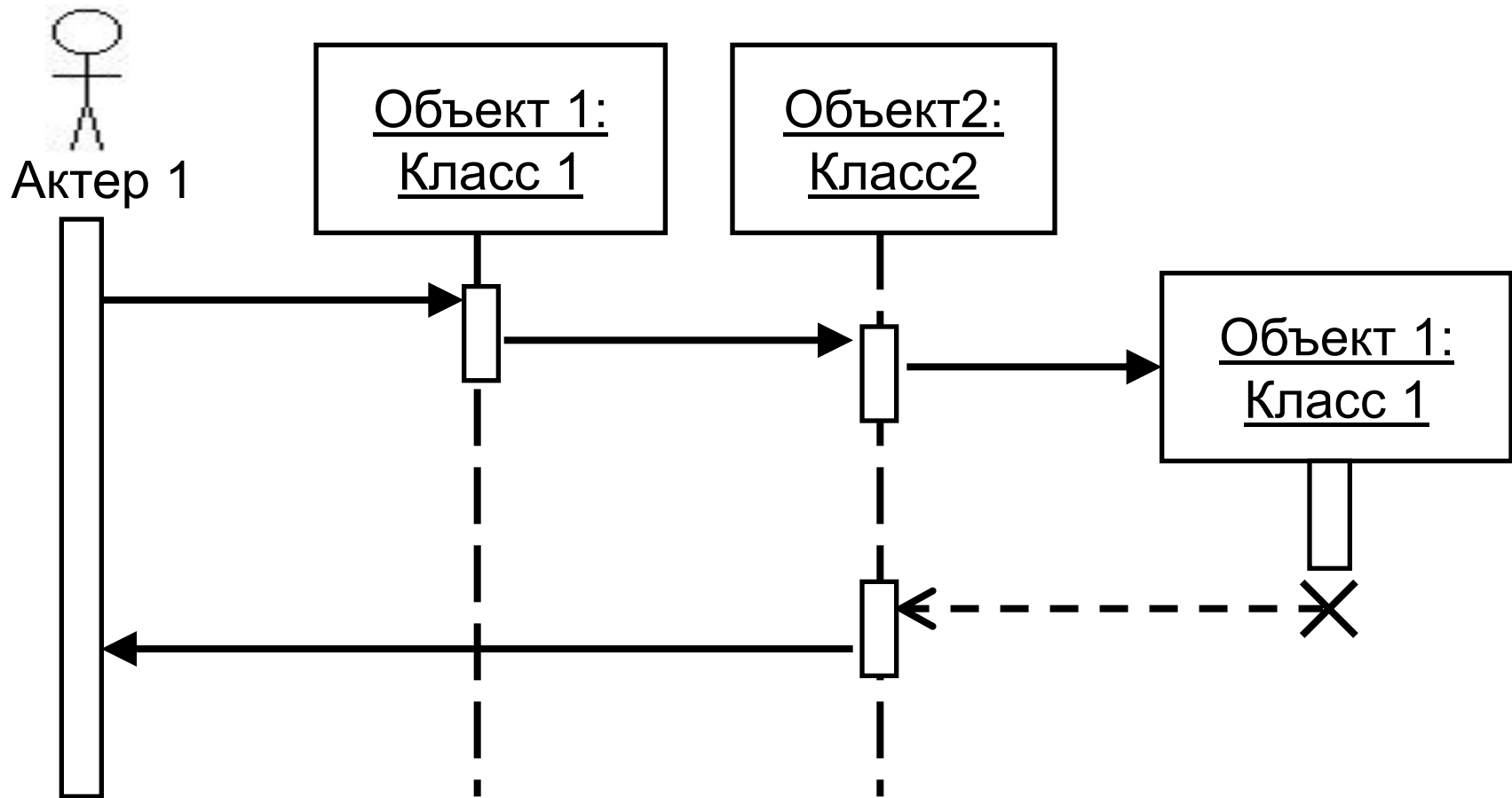


Диаграмма последовательности: элементы

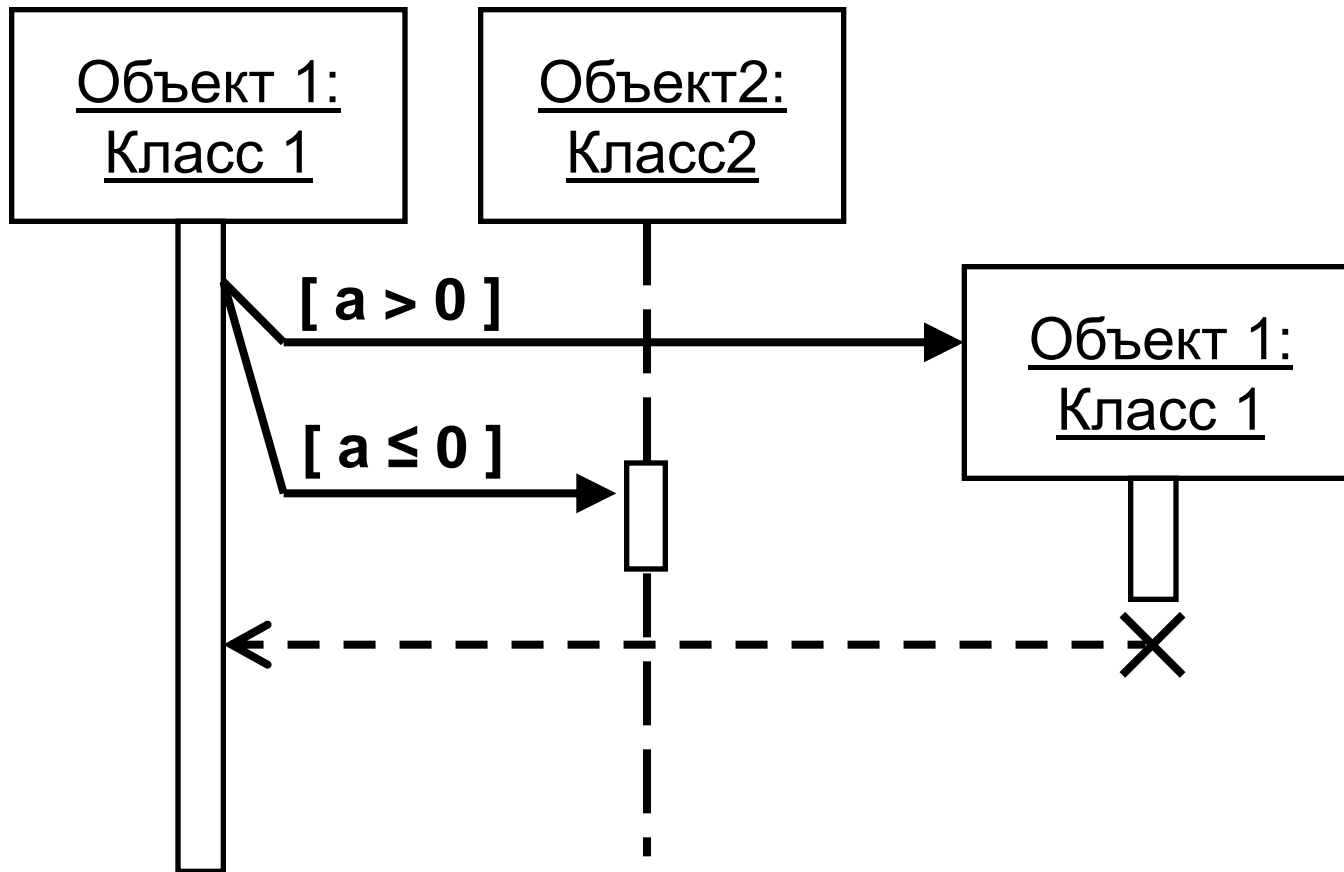


Диаграмма последовательности: элементы

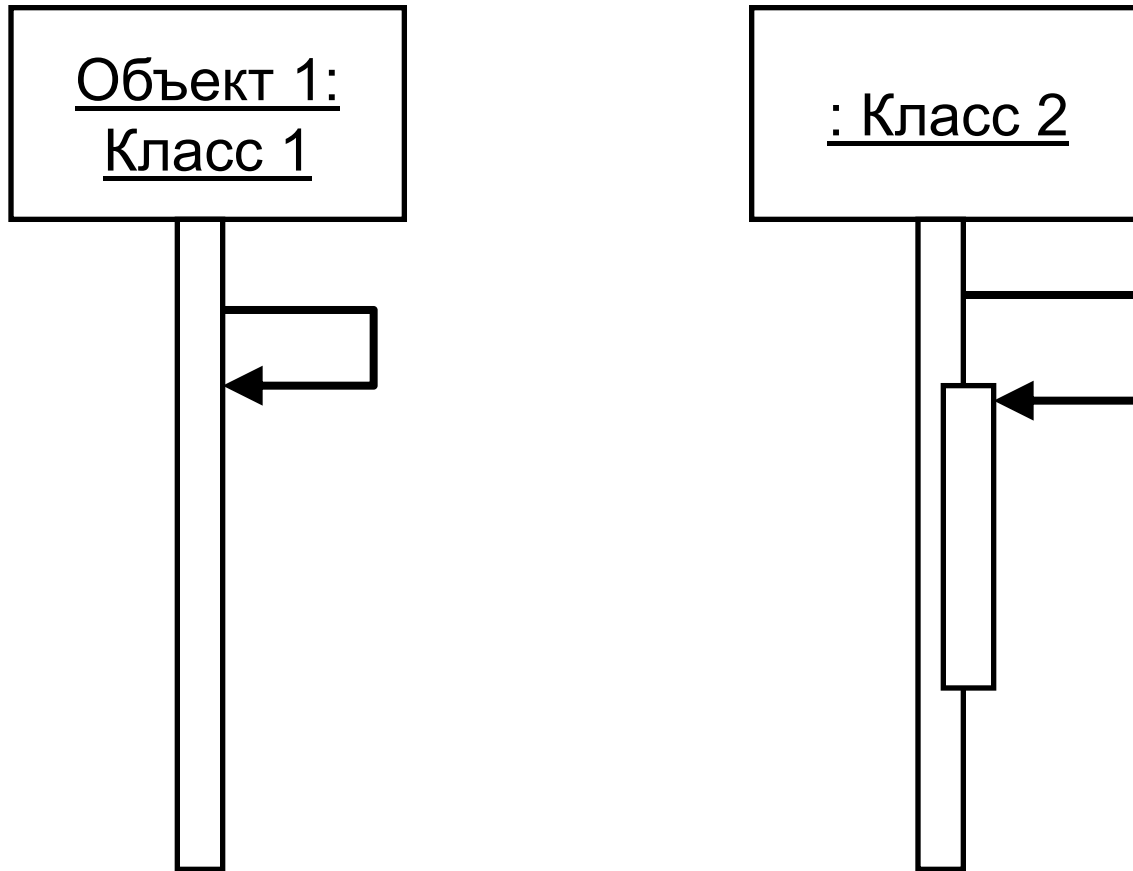


Диаграмма последовательности:

Типы сообщений

- ❑ Вызов процедуры
- ❑ Асинхронное сообщение
- ❑ Возврат из вызова процедуры

Диаграмма последовательности: элементы

Вызов



Один объект вызывает процедуру и ожидает, пока она не закончится.

Такое сообщение является синхронным.

Диаграмма последовательности: элементы

Асинхронное сообщение



Объект передает сообщение и продолжает выполнять свою деятельность, не ожидая ответа.

Диаграмма последовательности: элементы

Возврат



Объект передает сообщение об окончании выполнения процедуры.

Диаграмма последовательности: элементы

Метка



- ❑ стандартное сообщение
- ❑ имя функции
- ❑ граничное условие

Диаграмма последовательности: Стандартные сообщения

- ❑ «call»
- ❑ «return»
- ❑ «create»
- ❑ «destroy»
- ❑ «send»

Диаграмма последовательности

Пример

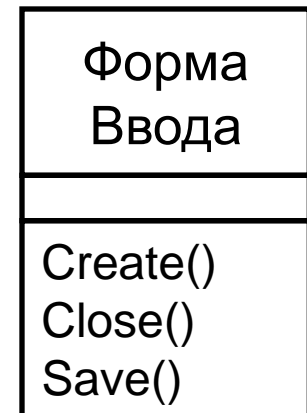
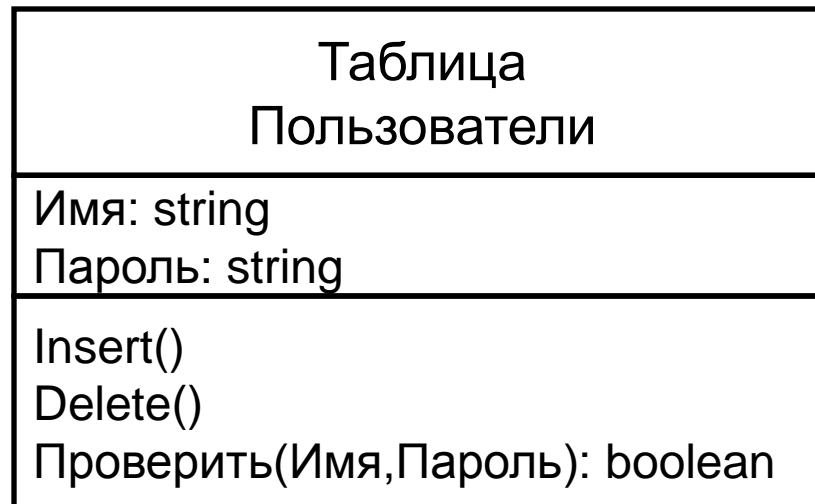
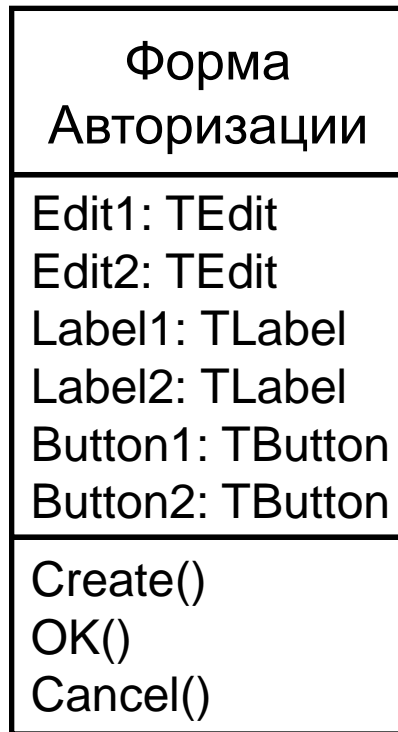


Диаграмма последовательности

Пример

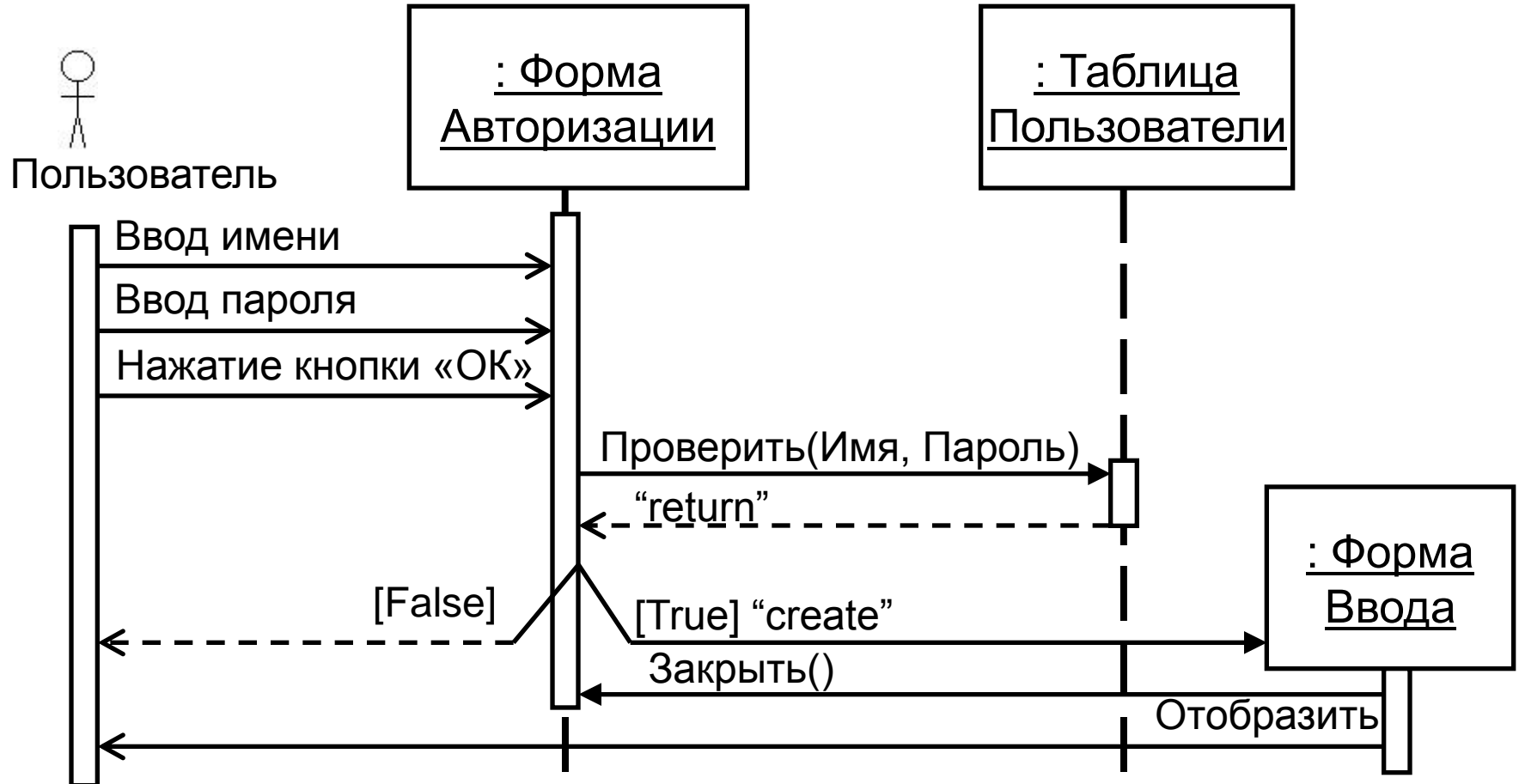


Диаграмма последовательности

Вопросы

- ❑ Что описывает диаграмма последовательности?
- ❑ Перечислить основные элементы диаграммы.
- ❑ Перечислить типы сообщений.

Задание

- Построить диаграмму последовательности для сценария «Выписать счет» учебной ИС «Гостиница»

Диаграмма коммуникации: определение

- Диаграмма коммуникации (кооперации) предназначена для спецификации структурных аспектов взаимодействия объектов

Диаграмма коммуникации: элементы

Элементы

- ❑ Объект
- ❑ Ассоциация
- ❑ Сообщение

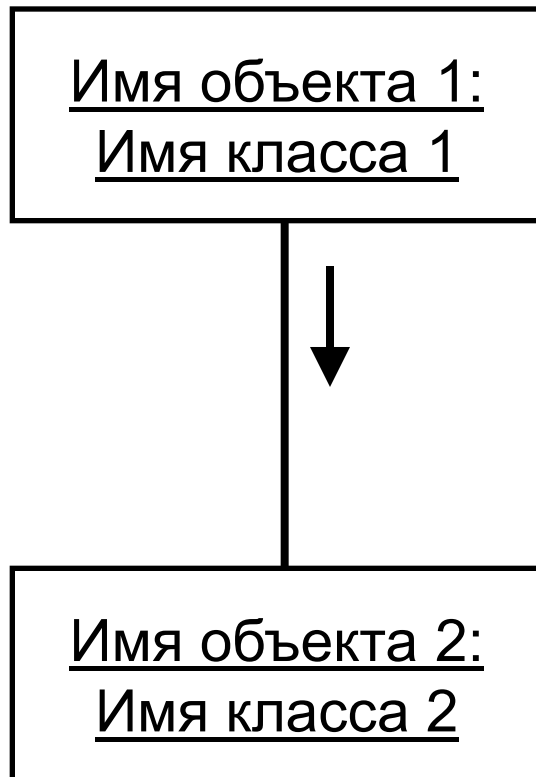


Диаграмма коммуникации

Пример

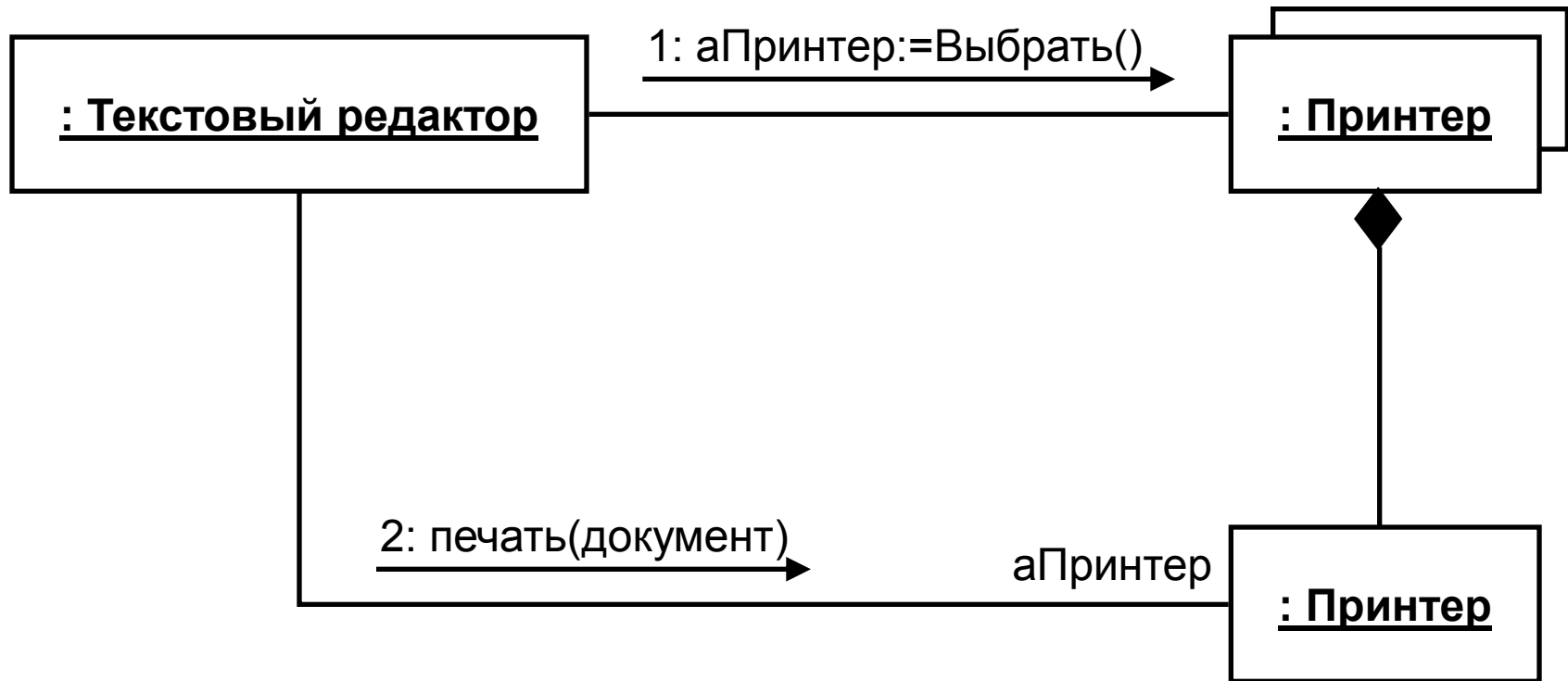


Диаграмма коммуникации

- Любую диаграмму последовательности можно преобразовать в диаграмму коммуникации, и наоборот

Диаграмма коммуникации

Пример

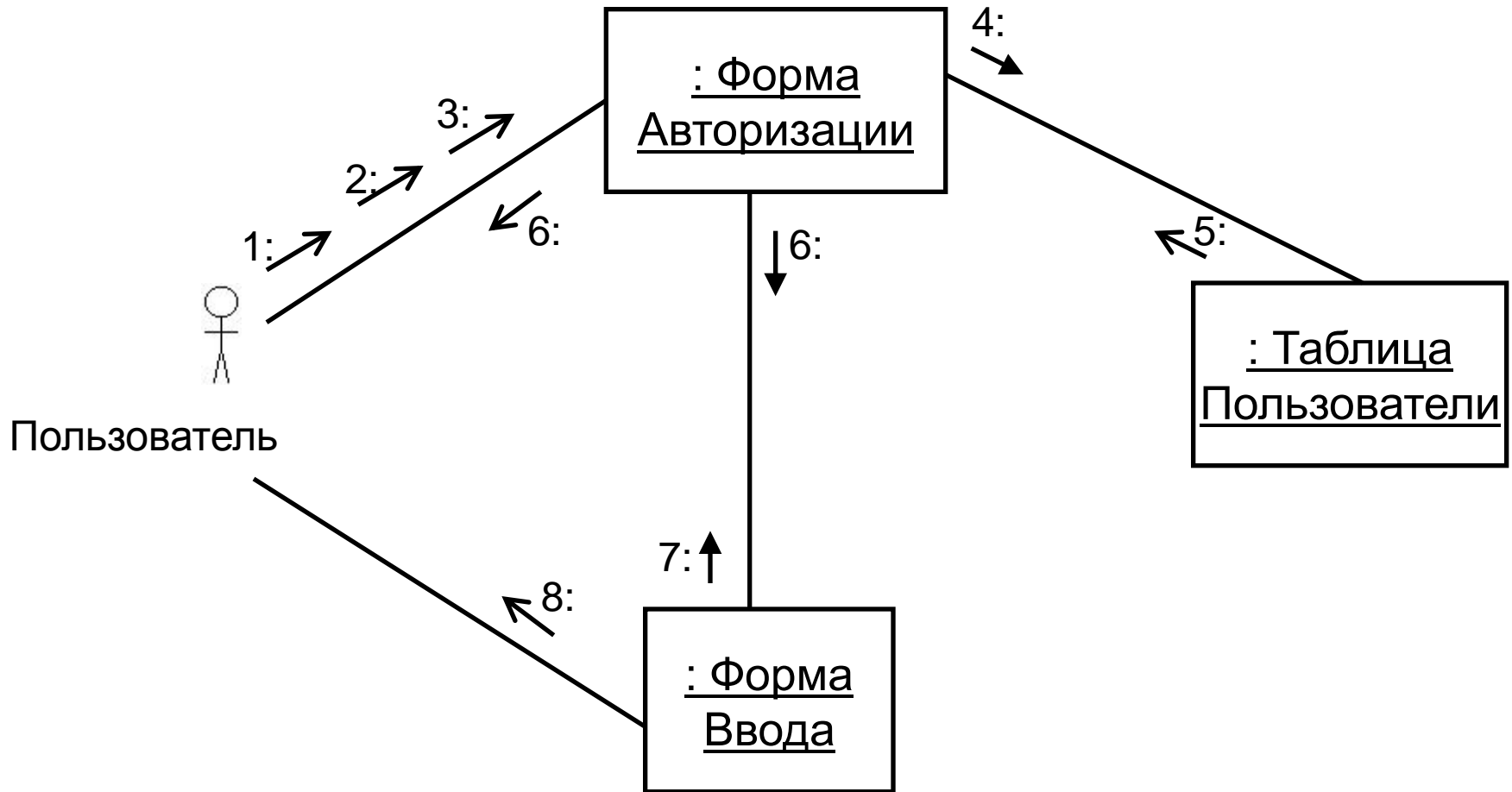


Диаграмма коммуникации

Вопросы

- ❑ Что описывает диаграмма коммуникации?
- ❑ Перечислить основные элементы диаграммы.

Задание

- Построить диаграмму коммуникации для сценария «Выписать счет» учебной ИС «Гостиница»

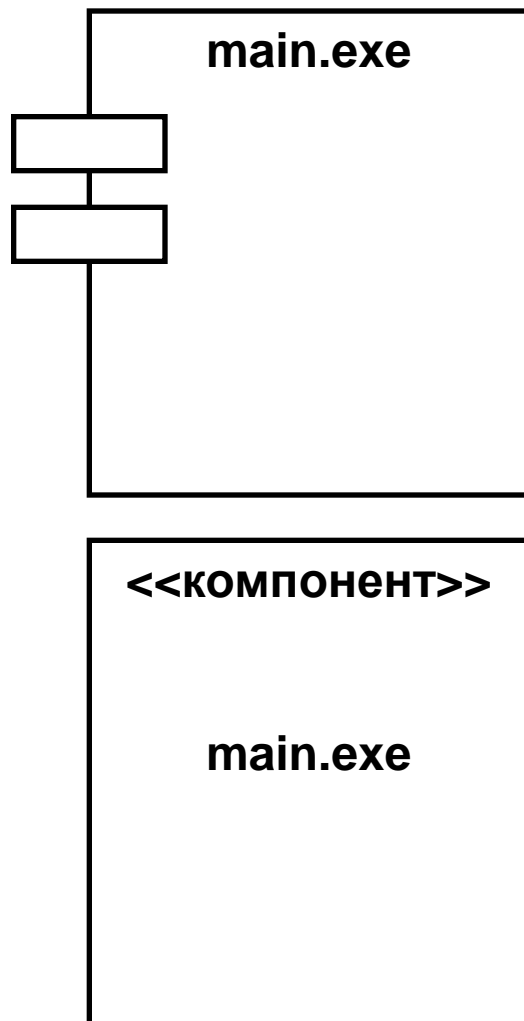
Диаграмма компонентов: определение

- Диаграмма компонентов описывает особенности физического представления системы

Цели построения диаграммы компонентов

- ❑ визуализация общей структуры исходного кода программной системы
- ❑ спецификация исполнимого варианта программной системы
- ❑ обеспечение многократного использования отдельных фрагментов программного кода
- ❑ представление концептуальной и физической схем баз данных

Диаграмма компонентов: элементы



Компонент — крупно модульный объект:

- ❑ исполняемый файл
- ❑ подсистема
- ❑ документ
- ❑ и др.

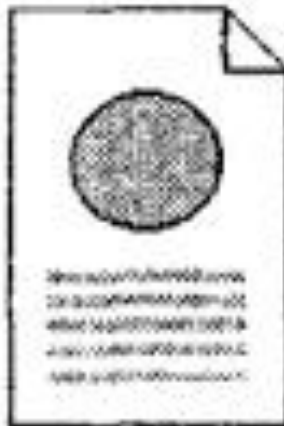
Диаграмма компонентов: КОМПОНЕНТЫ

dialog.dll



(a)

index.html



(б)

context.hlp



(в)

main.cpp



(г)

Диаграмма компонентов: интерфейс

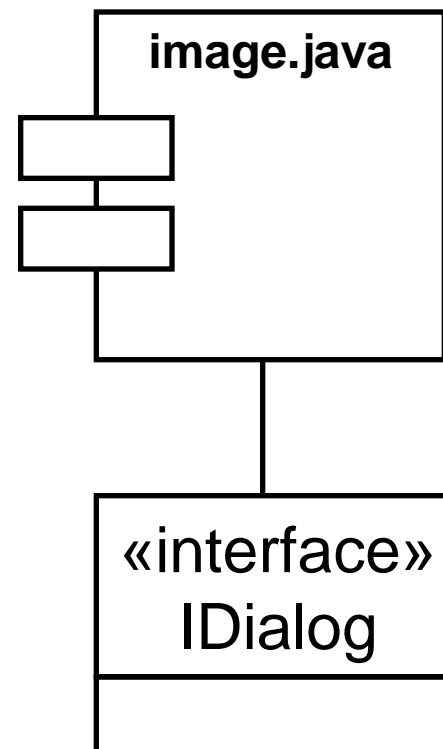
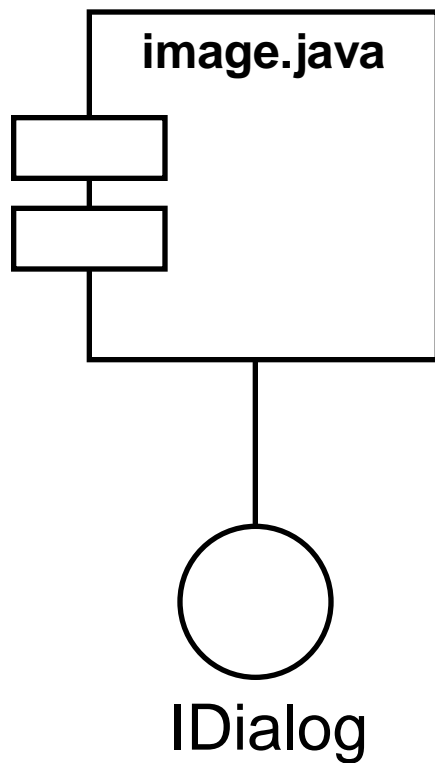


Диаграмма компонентов: интерфейс

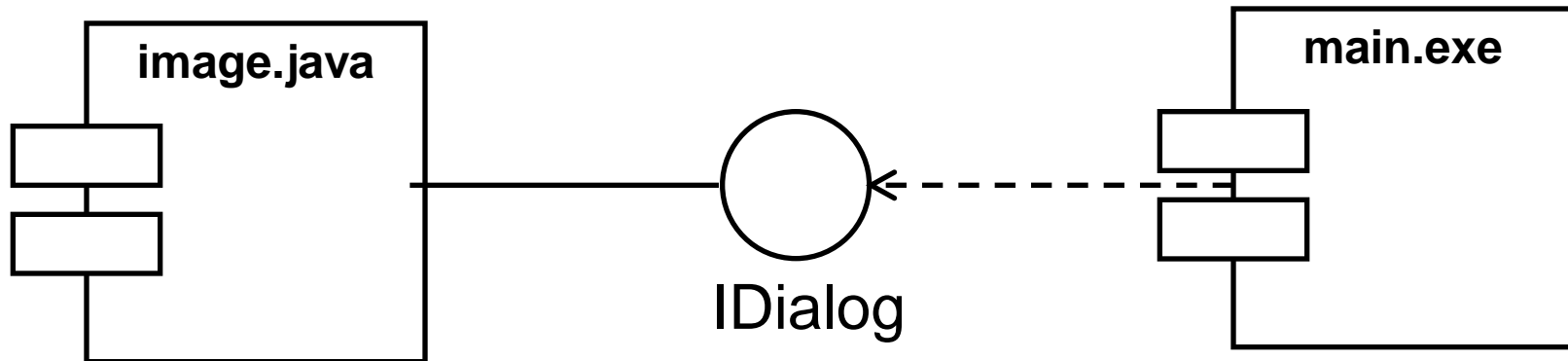


Диаграмма компонентов: зависимость

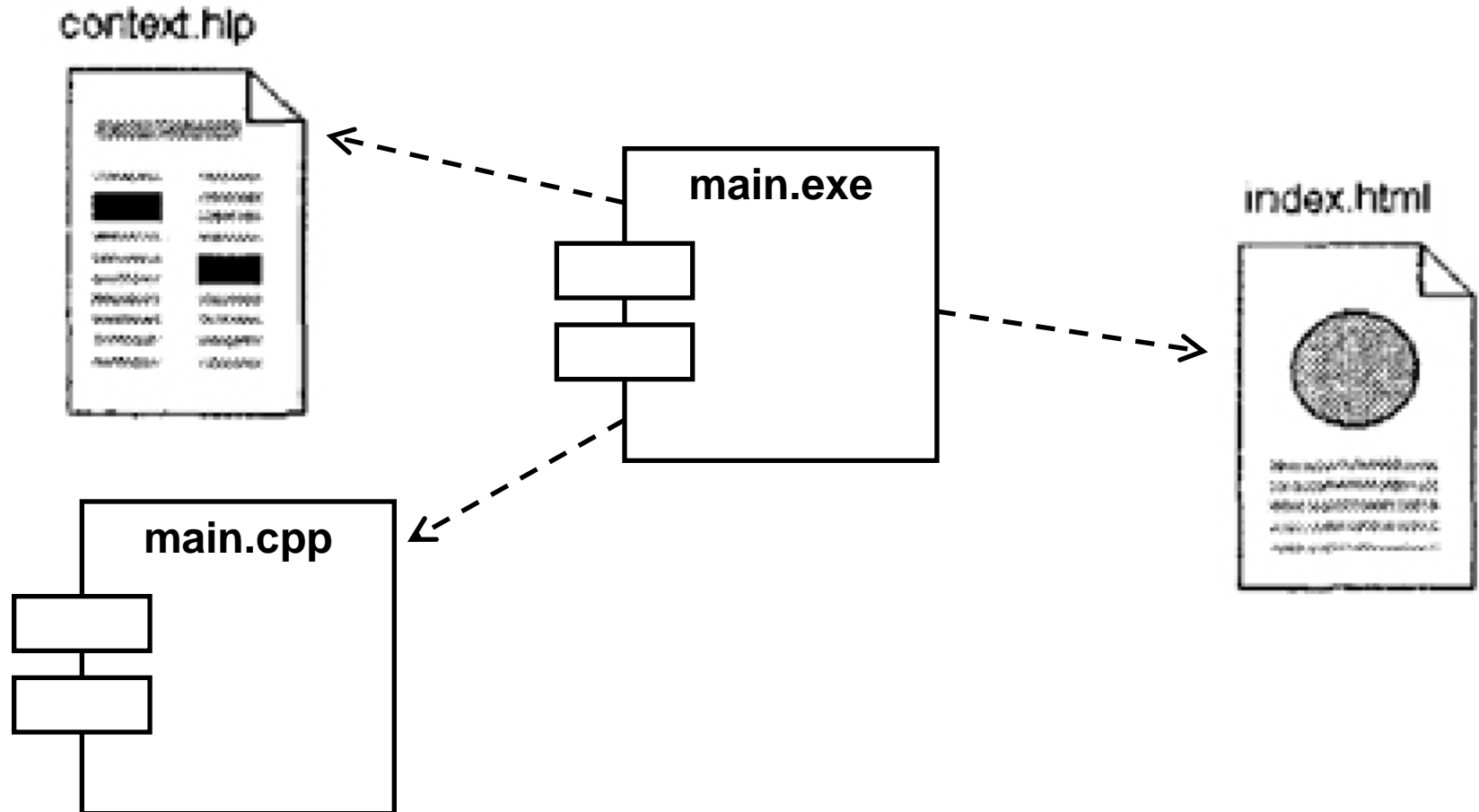


Диаграмма компонентов: зависимость

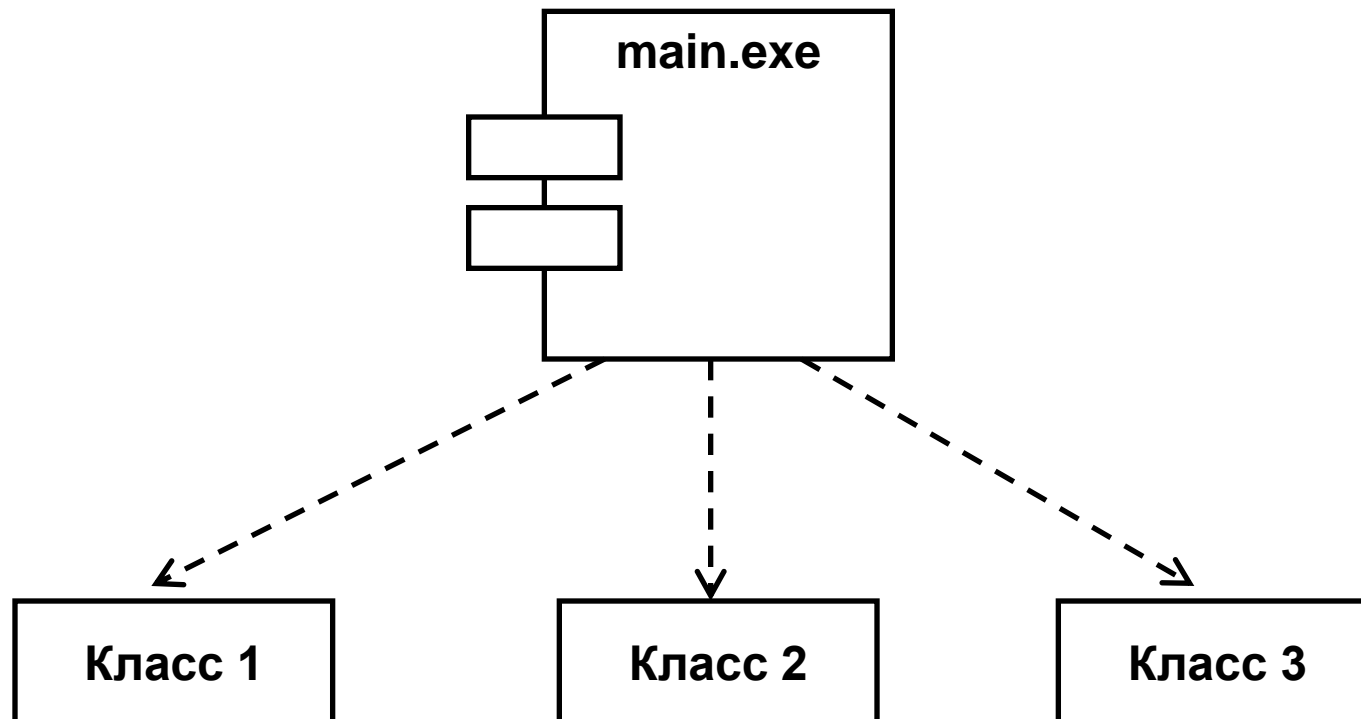


Диаграмма компонентов: реализация классов

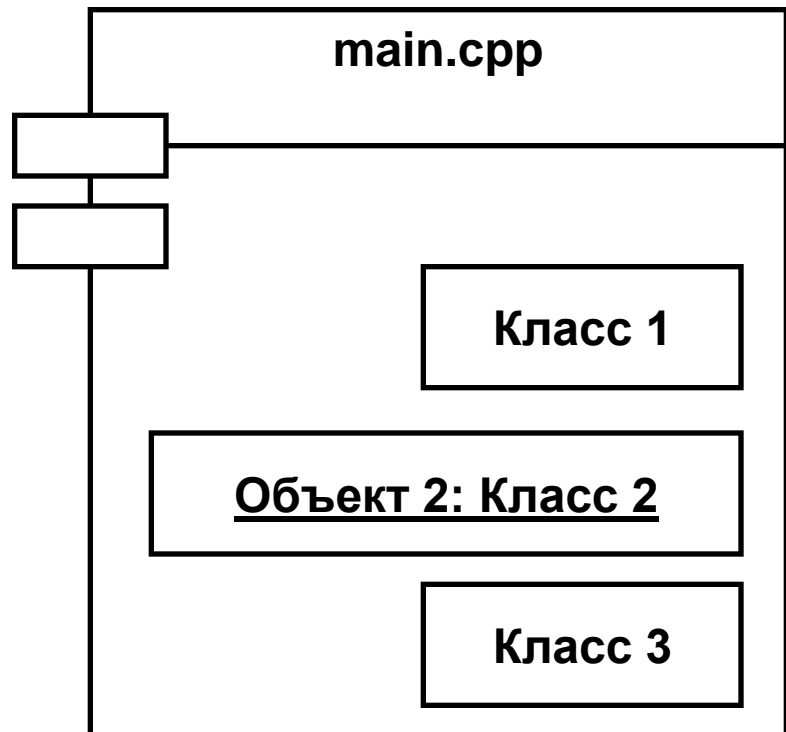
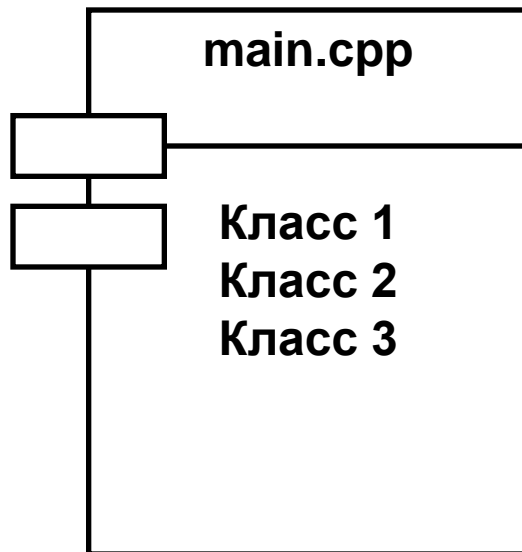


Диаграмма компонентов

Пример

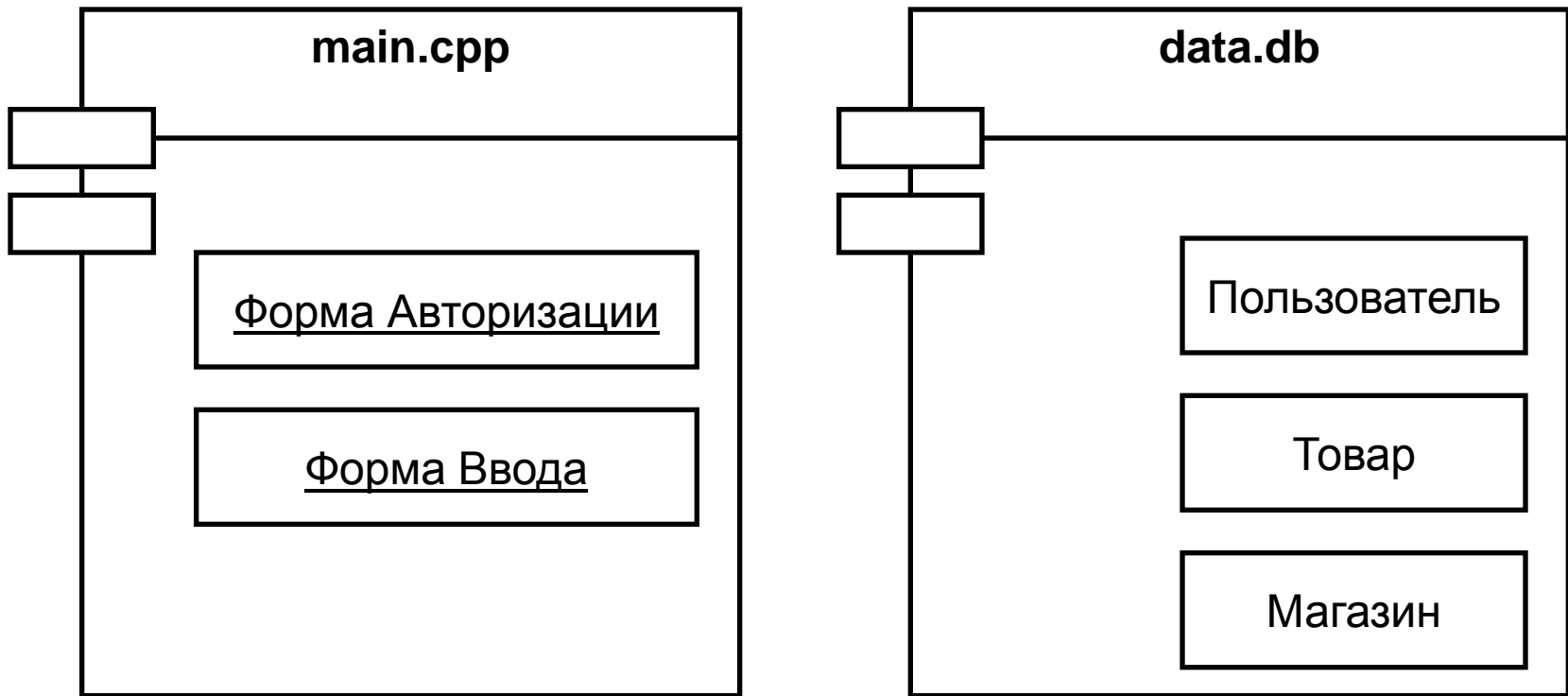


Диаграмма компонентов

Вопросы

- ❑ Что описывает диаграмма компонентов?
- ❑ Перечислить основные элементы диаграммы.
- ❑ Как на диаграмме компонентов отображается реализация классов, интерфейсов?

Задание

- Построить диаграмму компонентов для учебной ИС «Гостиница»

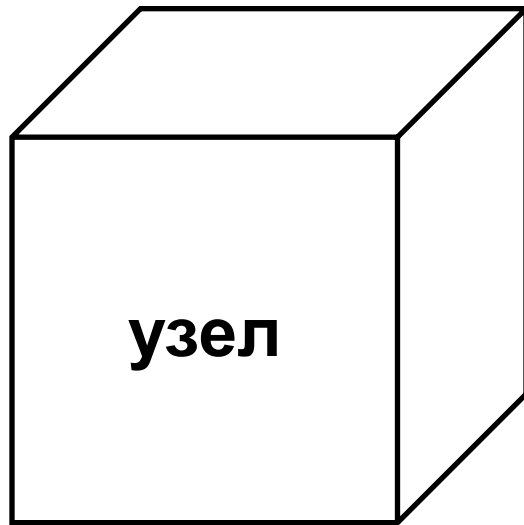
Диаграмма топологии: определение

- Диаграмма топологии применяется для представления общей конфигурации и топологии распределенной программной системы и содержит распределение компонентов по отдельным узлам системы

Цели построения диаграммы топологии

- ❑ определить распределение компонентов системы по ее физическим узлам
- ❑ показать физические связи между всеми узлами реализации системы на этапе ее исполнения
- ❑ выявить узкие места системы и реконфигурировать ее топологию для достижения требуемой производительности

Диаграмма топологии: элементы



Узел – физически
существующий элемент
системы :

- ❑ сервер
- ❑ рабочая станция
- ❑ принтер
- ❑ цифровая камера
- ❑ и др.

Диаграмма топологии: узлы

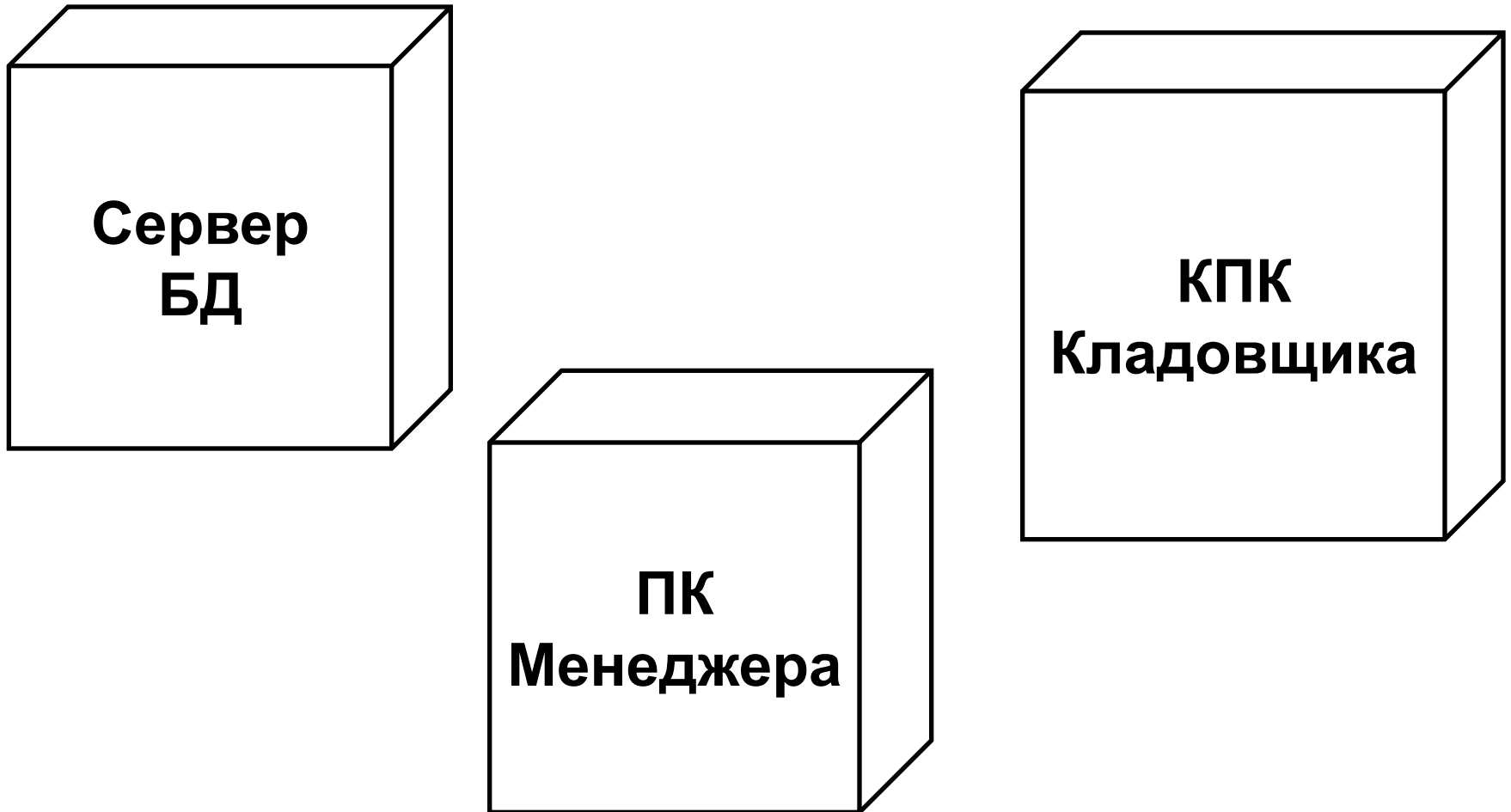


Диаграмма топологии:

- ассоциации
- КОМПОНЕНТЫ

Диаграмма топологии

Пример

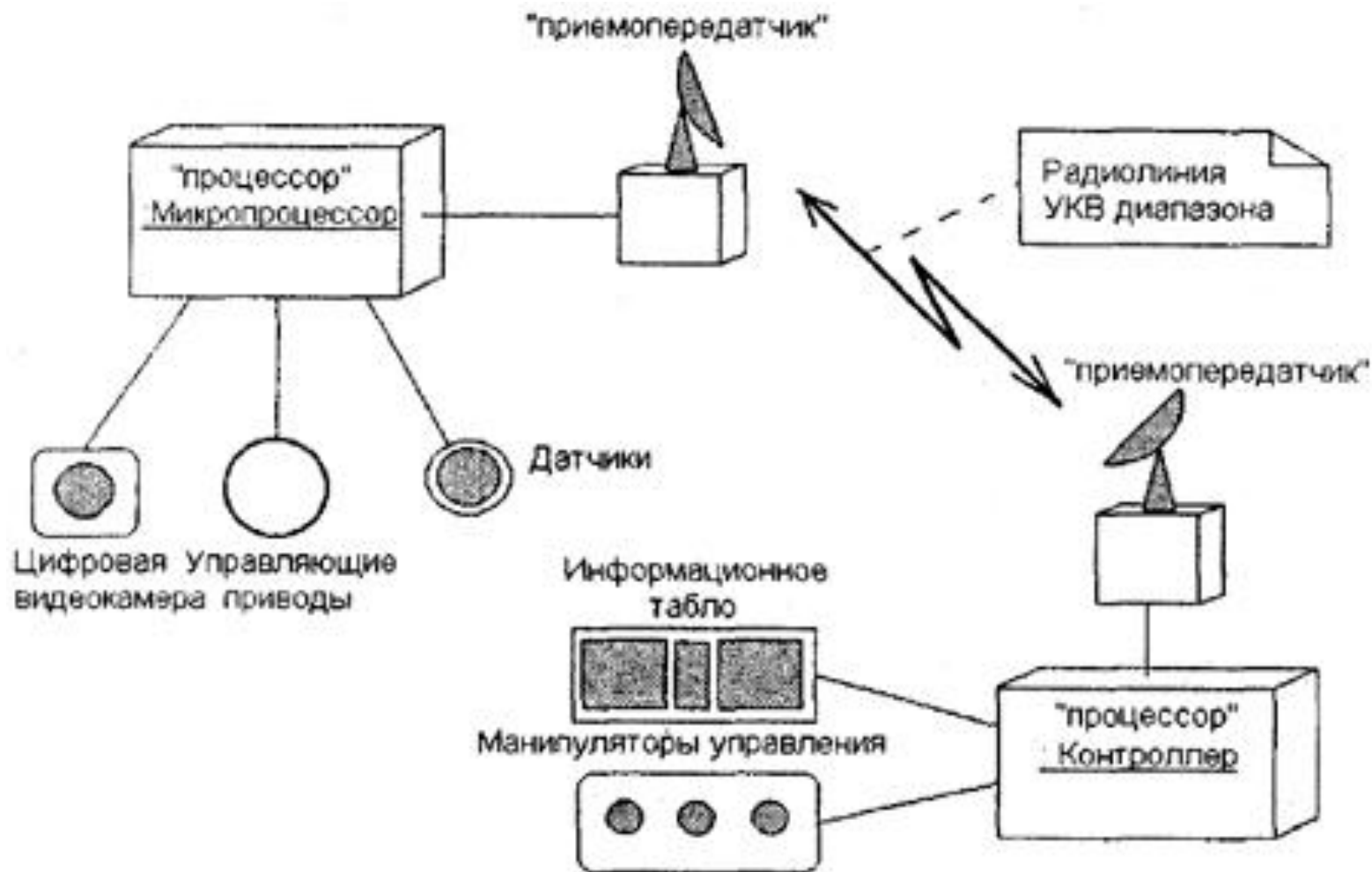


Диаграмма топологии

Вопросы

- ❑ Что описывает диаграмма топологии?
- ❑ Перечислить основные элементы диаграммы.

Задание

- ❑ Построить диаграмму топологии для учебной ИС «Гостиница»
- ❑ Построить диаграмму топологии для программной системы «автомобильная сигнализация»

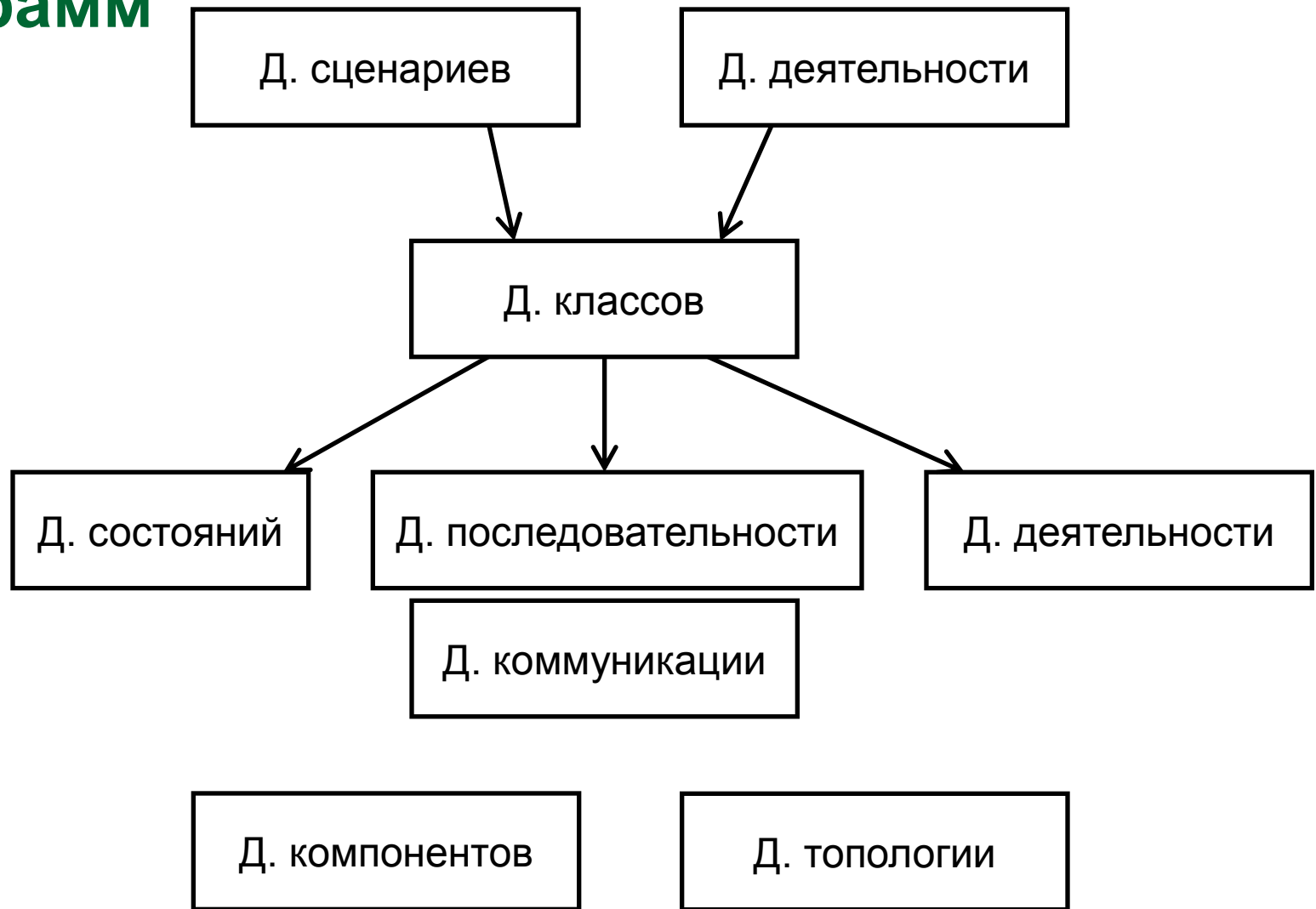
Последовательность построения диаграмм

Тема 1: Язык UML

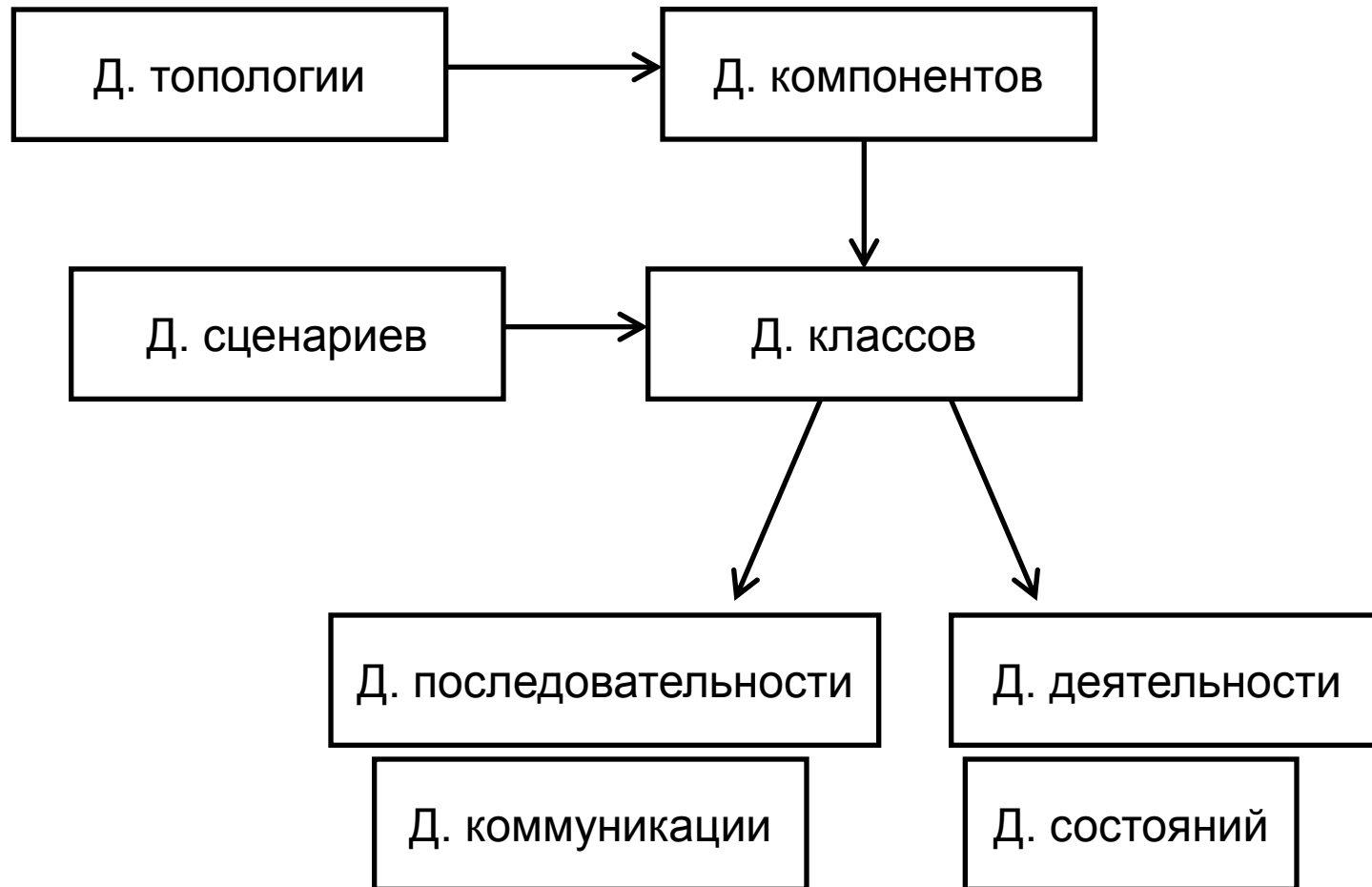
Последовательность построения диаграмм: способы

- ❑ от функций ИС
- ❑ от физической реализации

Последовательность построения диаграмм



Последовательность построения диаграмм



Диаграммы UML

Вопросы

- ❑ В какой последовательности разрабатываются диаграммы UML?

CASE – системы для построения диаграмм

Тема 1: Язык UML

CASE - системы

- CASE (Computer Aided Software Engineering) – программные средства, поддерживающие процессы создания и сопровождения ИС

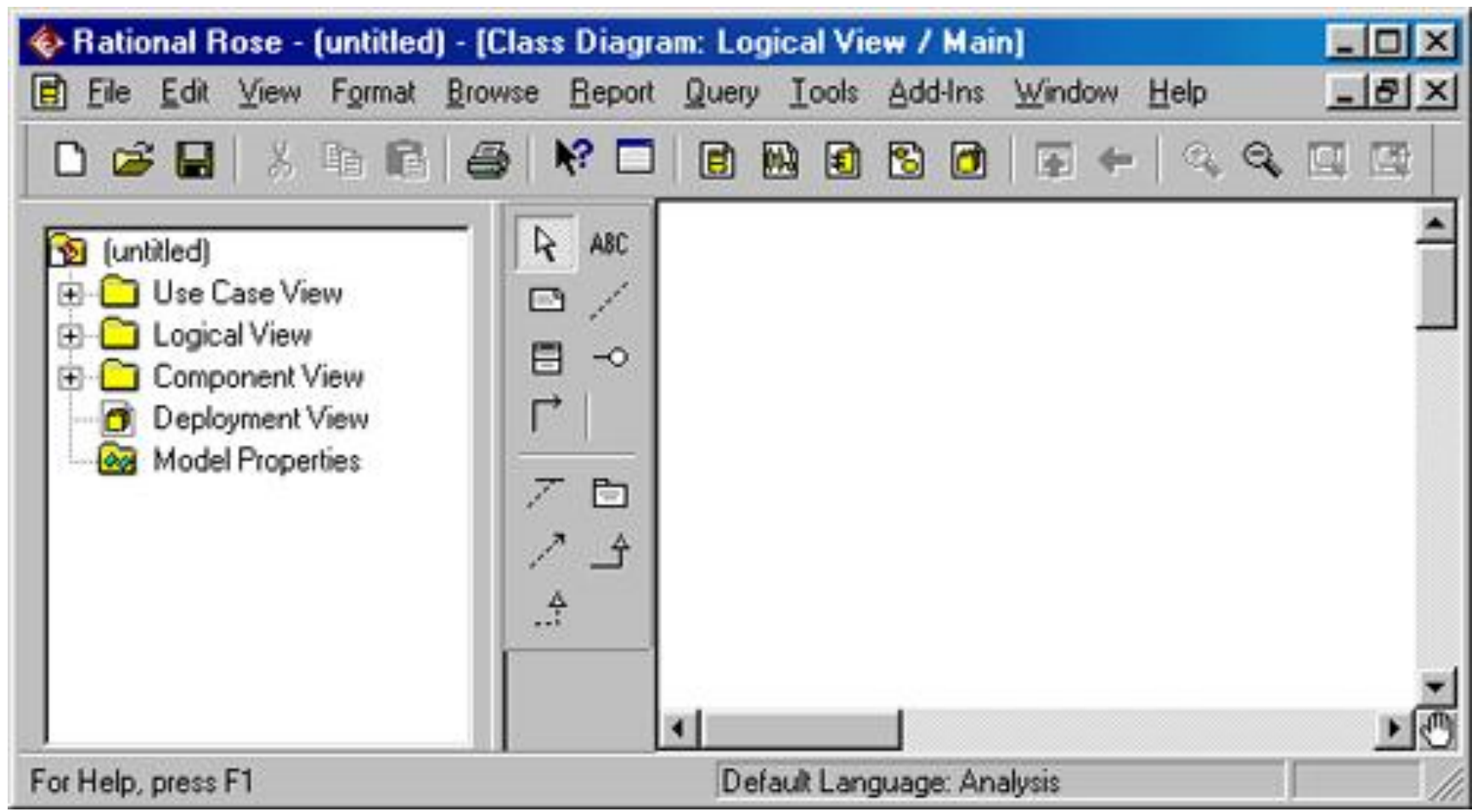
Rational Rose

- ❑ разработчик – Rational Software Corp.
- ❑ UML

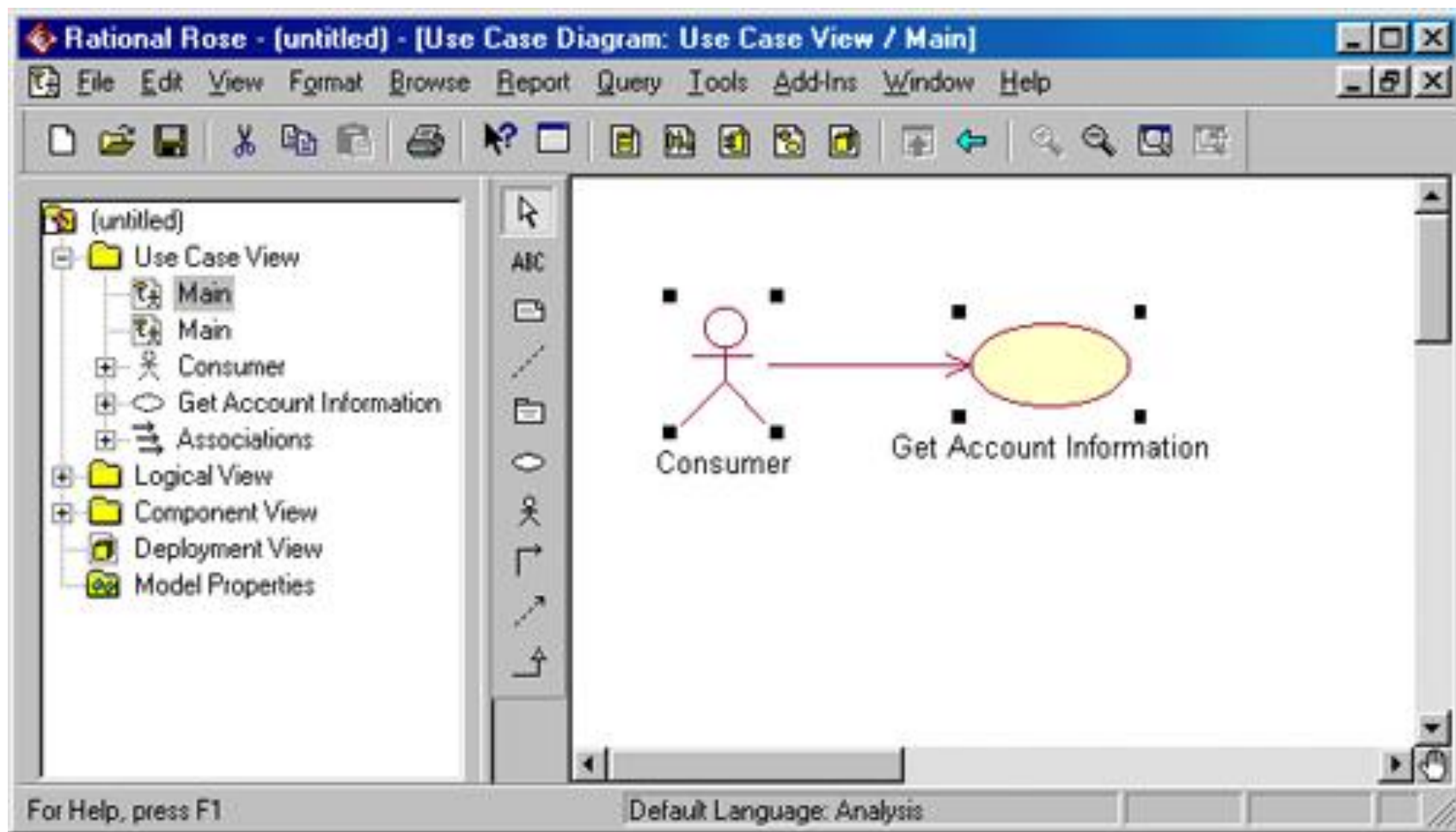
Rational Rose: генерация кода на языках

- ❑ Java
- ❑ C++
- ❑ VisualBasic
- ❑ и другие

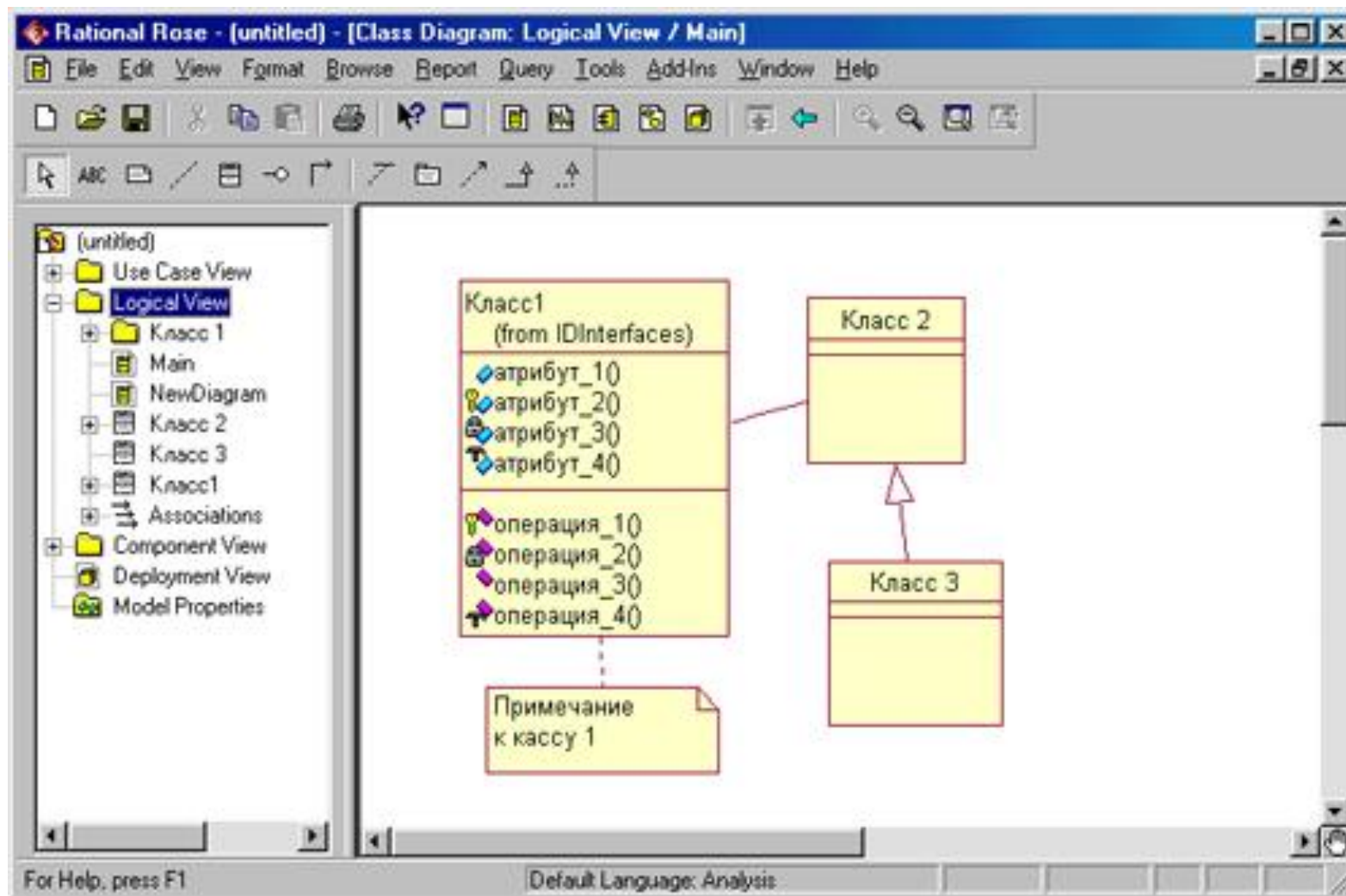
Rational Rose: внешний вид



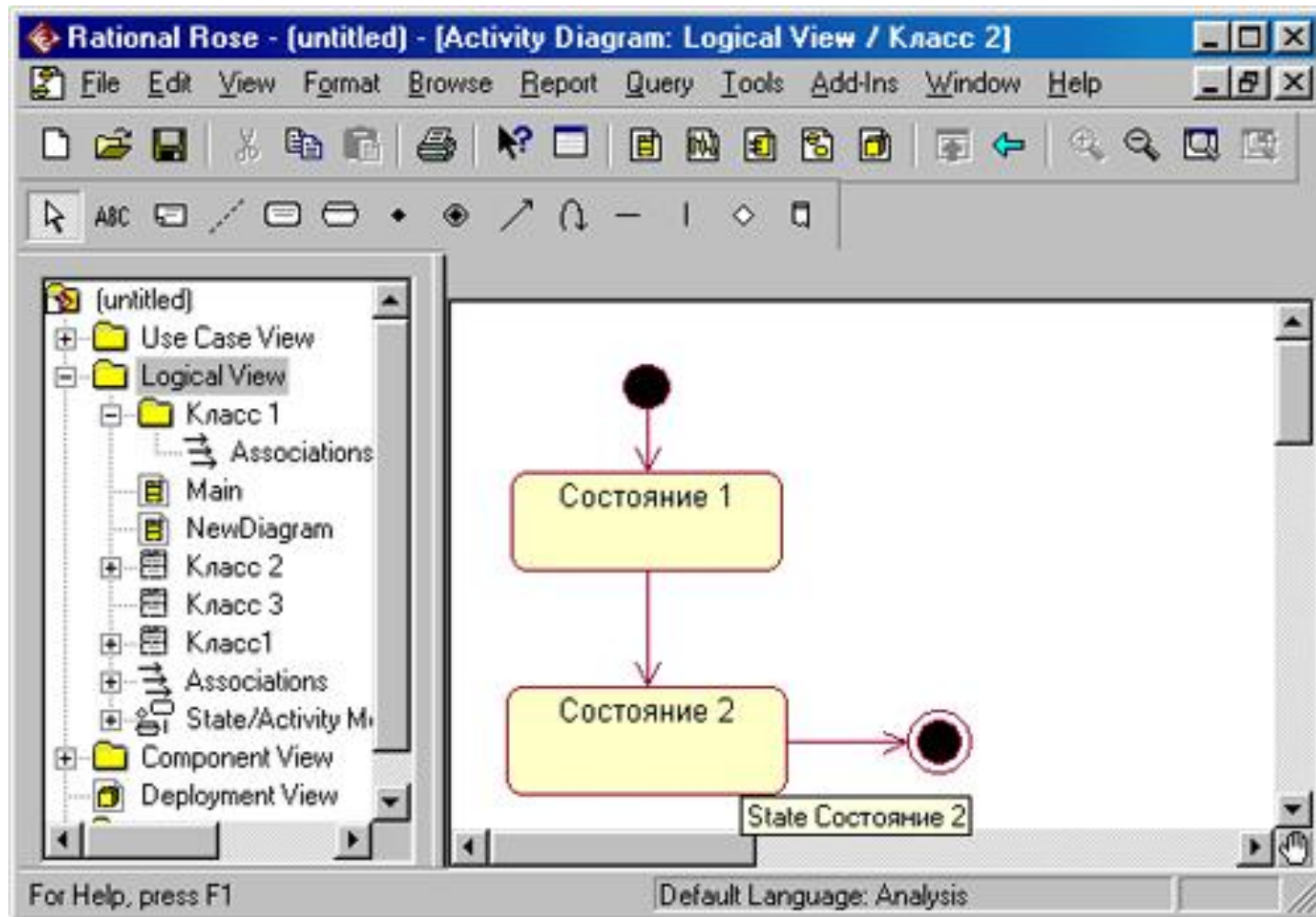
Rational Rose: диаграмма сценариев



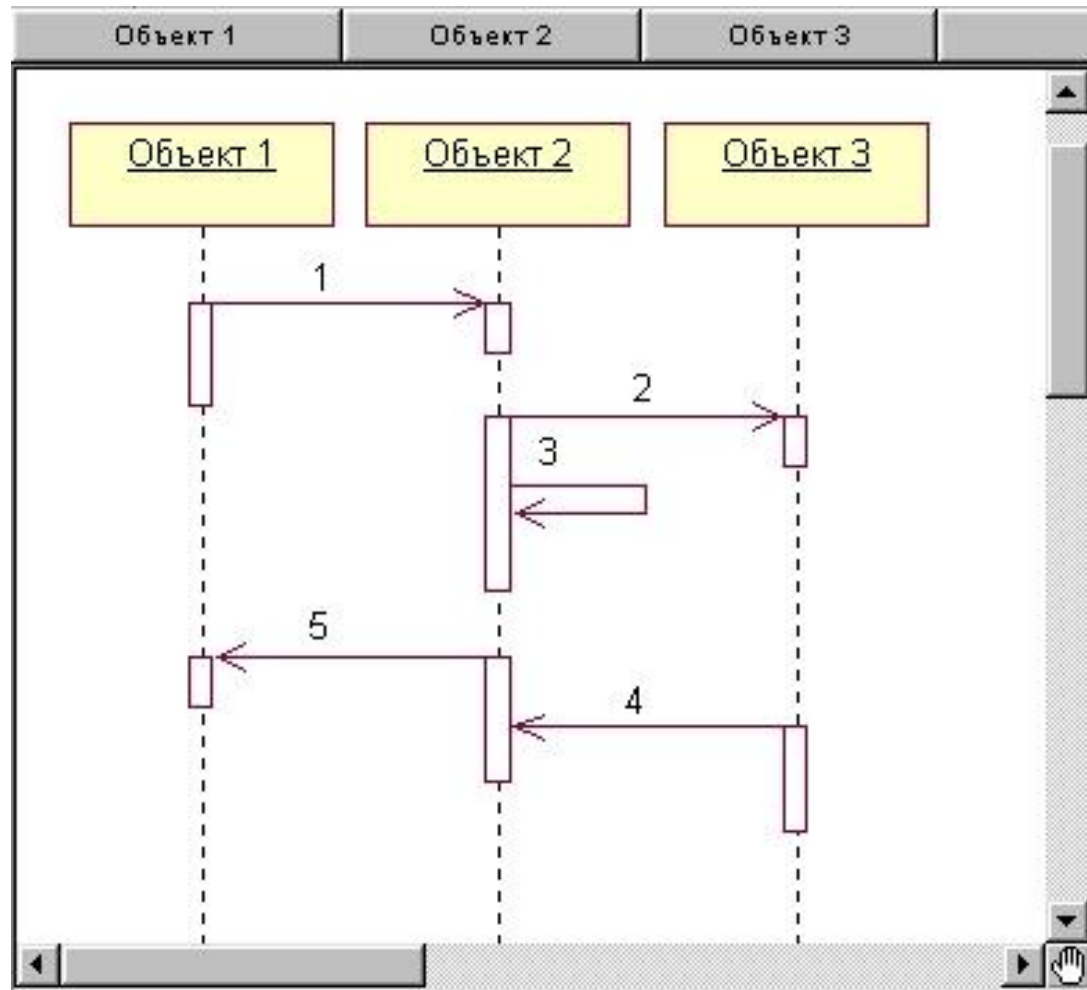
Rational Rose: диаграмма классов



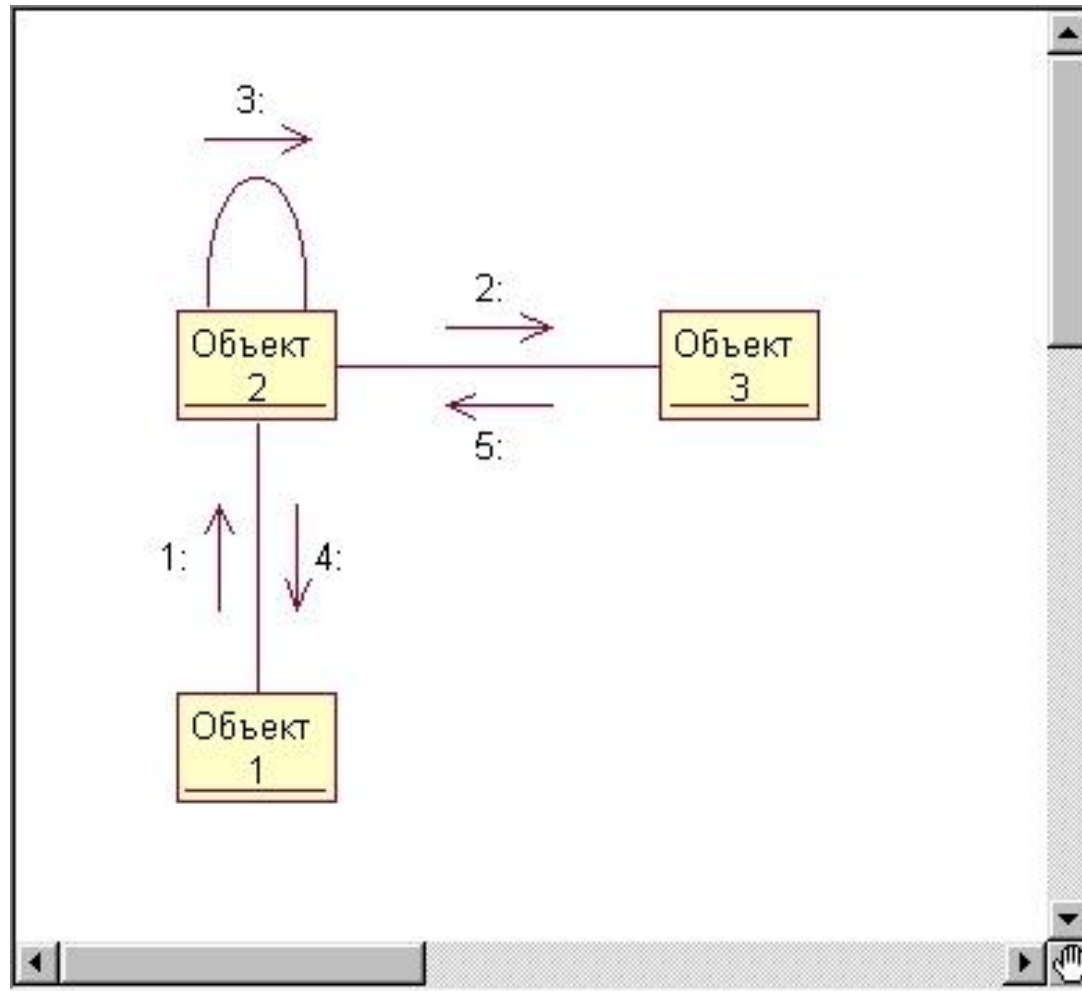
Rational Rose: диаграмма состояний



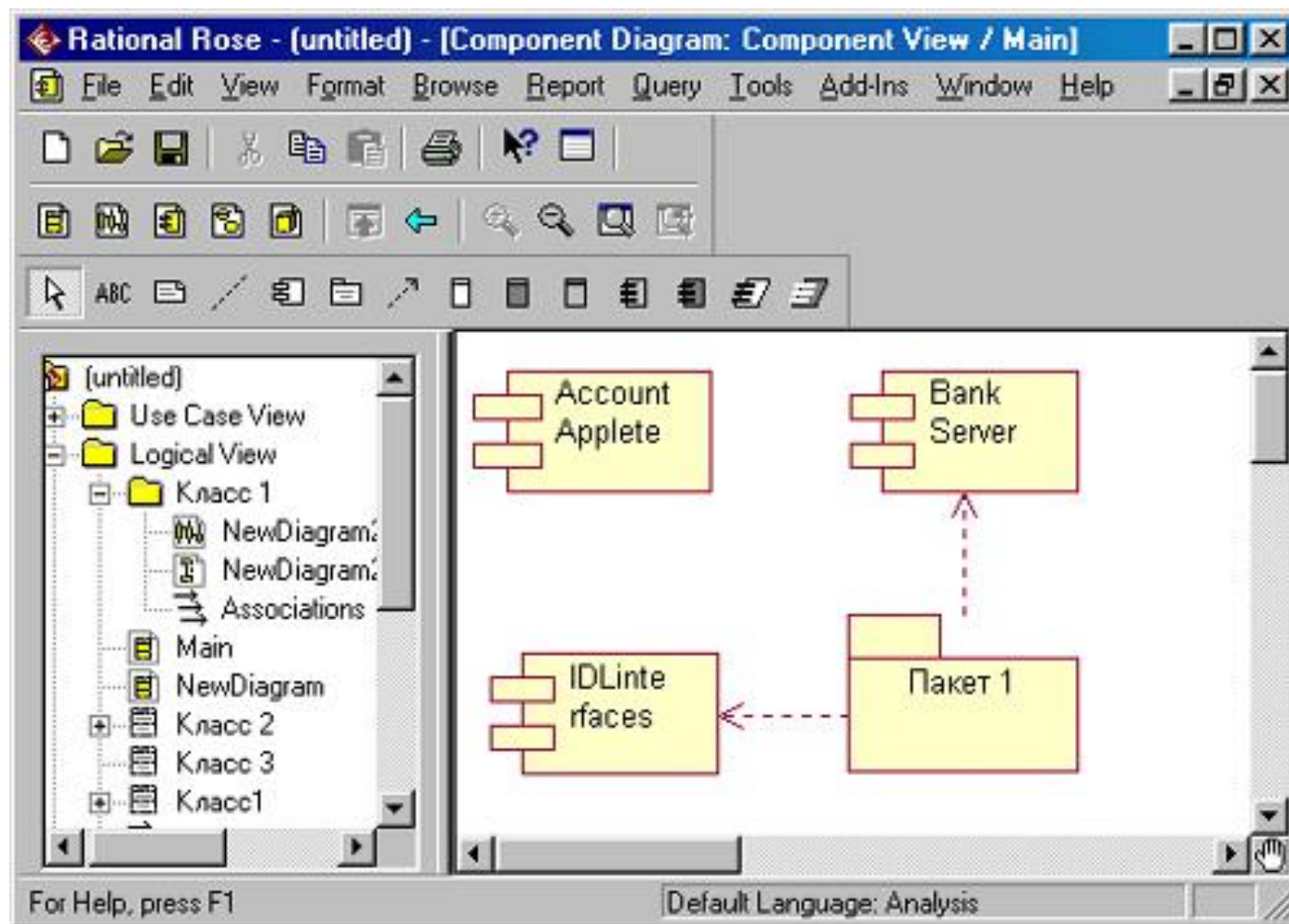
Rational Rose: диаграмма последовательности



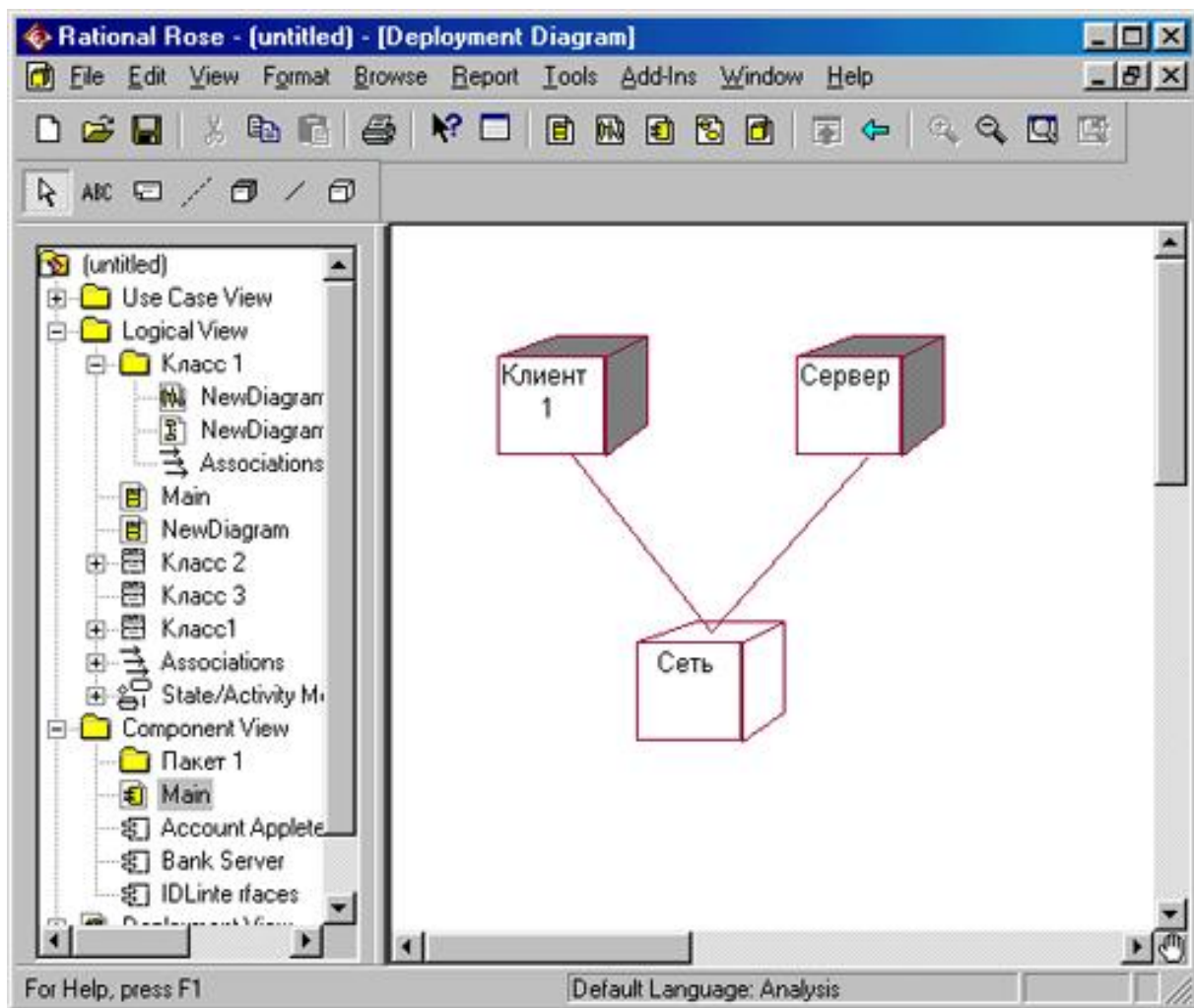
Rational Rose: диаграмма коммуникации



Rational Rose: диаграмма компонентов



Rational Rose: диаграмма топологии



Диаграммы UML

Вопросы

- ❑ Что такое CASE системы?
- ❑ Какими функциями обладают CASE системы?

Заключение

- ❑ UML – объектно-ориентированный метод разработки программного обеспечения
- ❑ UML включает 8 основных диаграмм (сценариев, классов, деятельности, состояний, последовательности, коммуникации, компонентов, топологии)
- ❑ CASE системы – программные средства, поддерживающие процессы создания и сопровождения ИС

Список литературы

- ❑ Орлов С. Технологии разработки программного обеспечения / С. Орлов. – СПб.: Питер, 2002. – 464 с.
- ❑ Буч Г. Язык UML. Руководство пользователя / Г. Буч, Дж. Рамбо, А. Джекобсон. – М.: ДМК, 2000. – 432 с.
- ❑ Буч Г. UML: специальный справочник / Г. Буч, Дж. Рамбо, А. Джекобсон – СПб: «Питер», 2001. – 656 с.
- ❑ Шмуллер Дж. Освой самостоятельно UML за 24 часа / Дж. Шмуллер. – М.: Издательский дом «Вильямс», 2005. – 416 с.
- ❑ Леоненков А. В. Самоучитель UML / А. В. Леоненков. – СПб.: БХВ-Петербург, 2006. – 432 с.
- ❑ Вендров А.М. CASE-технологии: Современные методы и средства проектирования информационных систем. – М.: Финансы и статистика, 1998. 176 с.