# Технологии и методы программирования

Часть 7

Ст. преподаватель
кафедры ПИВТ
Воронцова И.О.

2020 год

# Общие вопросы разработки программного обеспечения

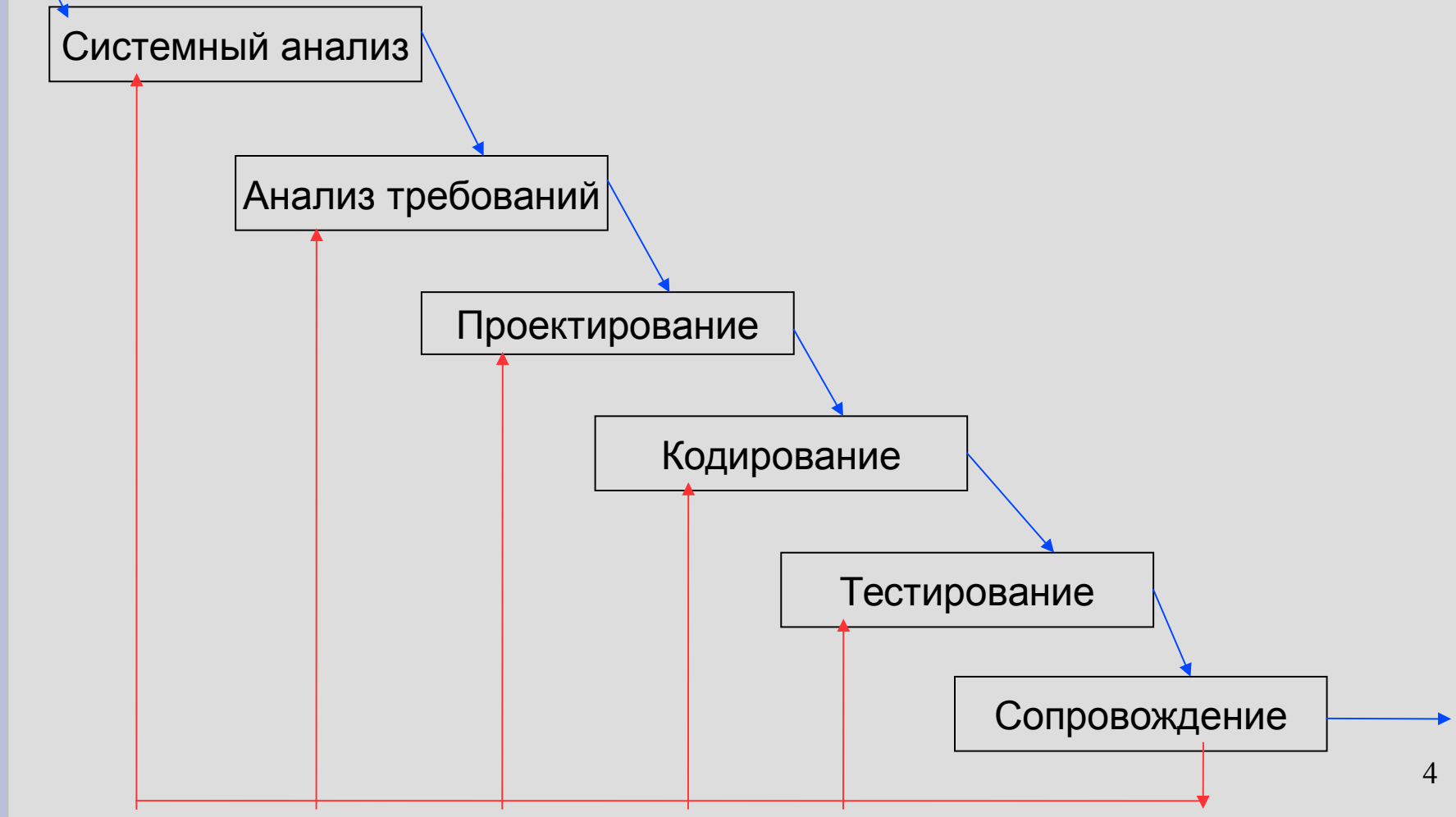
- Software development
- Особенности современного ПО:
  - Многофункциональность
  - Расширение области применения
  - Быстрая смена аппаратной базы
  - Требования к безопасности
- Software Engineering - Это интегрирование принципов математики, информатики и компьютерных наук с инженерными подходами в целях разработки и использования программного обеспечения
- Международный стандарт по программной инженерии SWEBOK (SoftWare Engineering Body of Knowledge) IEEE (Institute of Electrical and Electronics Engineering) 2004 год
- Международный стандарт по преподаванию программной инженерии в вузах Software Engineering. Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering ACM (Association for Computing Machinery) 2014 год.

# Структура и содержание SWEBOK

- *Software requirements* – программные требования
- *Software design* – дизайн (архитектура)
- *Software construction* – конструирование программного обеспечения
- *Software testing* - тестирование
- *Software maintenance* – эксплуатация (поддержка) программного обеспечения
- *Software configuration management (SCM)* – конфигурационное управление
- *Software engineering management* – управление проектом (PMBOK)
- *Software engineering process* – процессы программной инженерии
- *Software engineering tools and methods* – инструменты и методы
- *Software quality* – качество программного обеспечения

# Жизненный цикл ПО

- Водопадная модель по Уинстону Ройсу

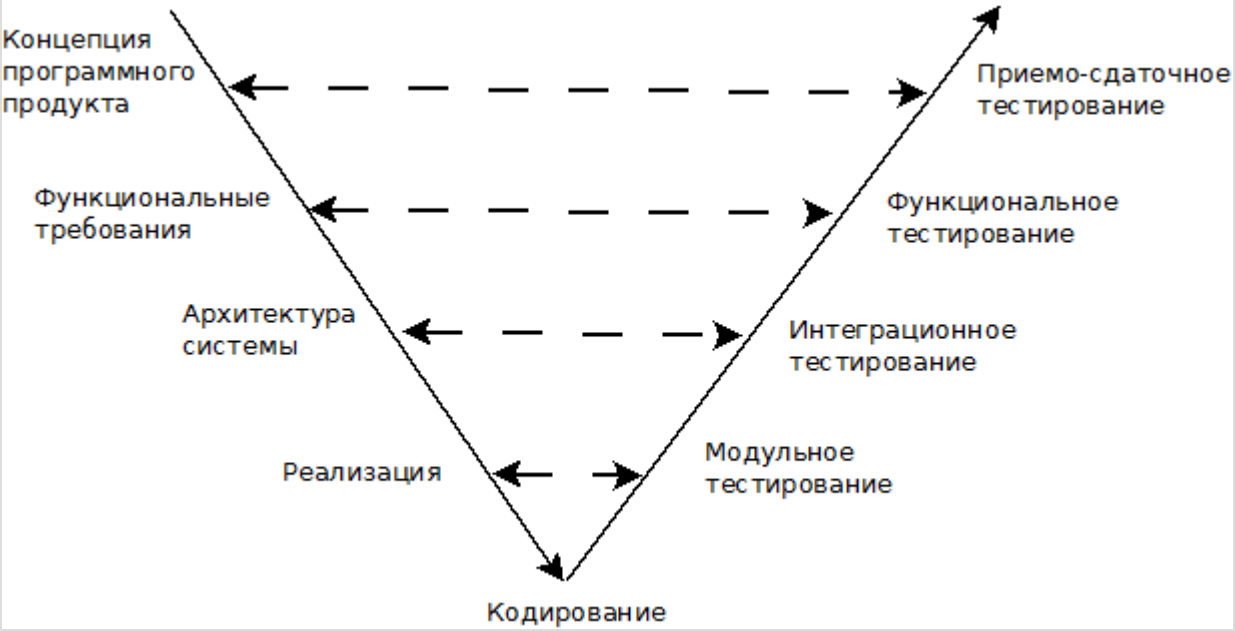


# Стратегии разработки ПО

1. Водопадная (линейная).
2. Инкрементная.
3. Итеративная.
4. Эволюционная.

# Водопадная стратегия

Основана на водопадной модели жизненного цикла.
Вариант: V-модель разработки.



# Инкрементная стратегия

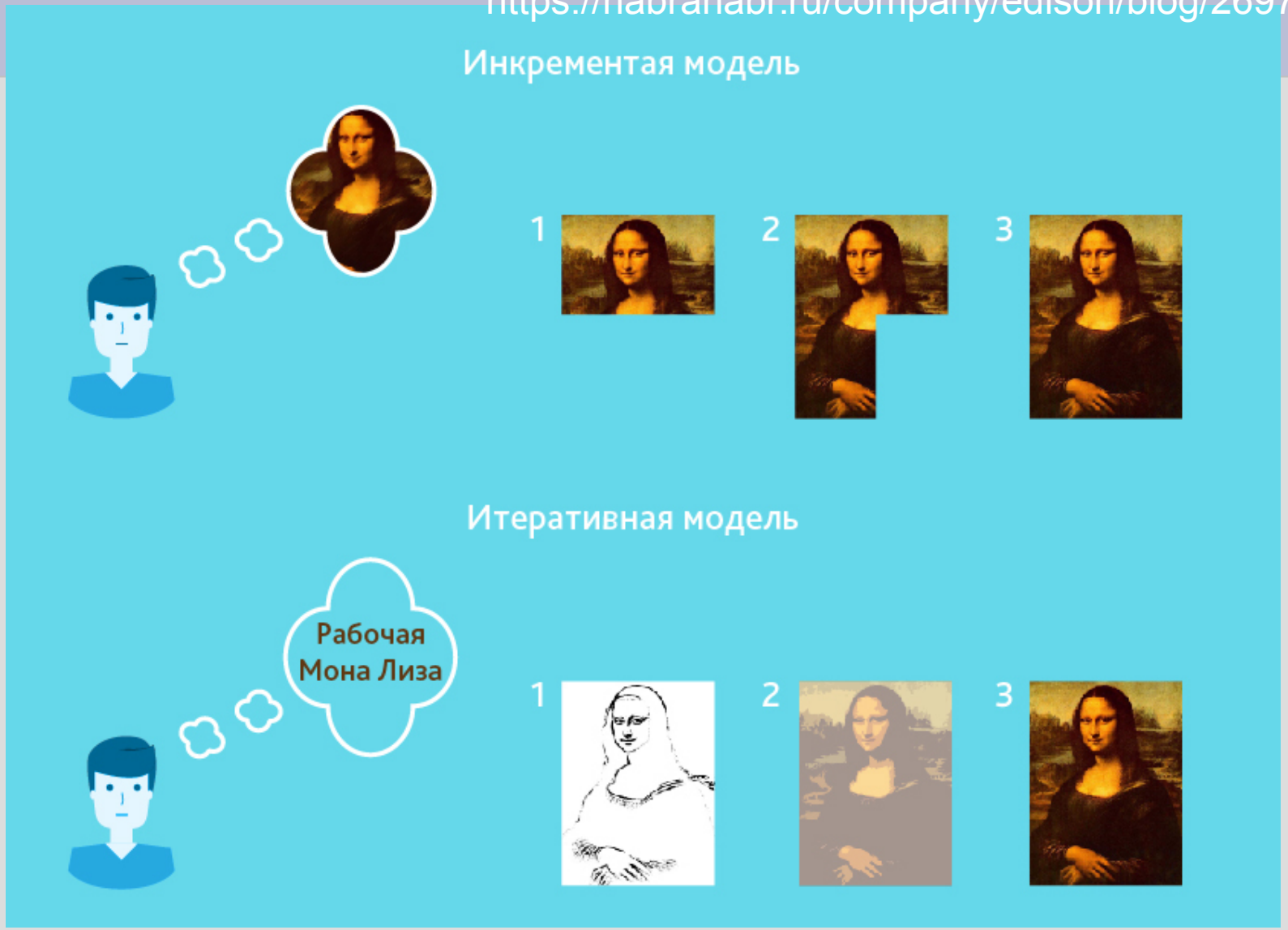
- Инкрементная стратегия – запланированное улучшение программного продукта.
  - Инкрементная модель.
  - Модель быстрой разработки RAD (Rapid Application Development) : компонентная модель.

# Модель быстрой разработки

- Достоинства:
  - Обеспечивает уменьшение времени выполнения итерации.
  - Модель может быть использована для информационных систем.
- Недостатки:
  - Применима для приложений, которые можно декомпозировать на отдельные модули.
  - Не обеспечивают высокую производительность ПО.
  - Не применимы в условиях высокого технического риска.

# Итеративная стратегия

https://habrahabr.ru/company/edison/blog/269789/

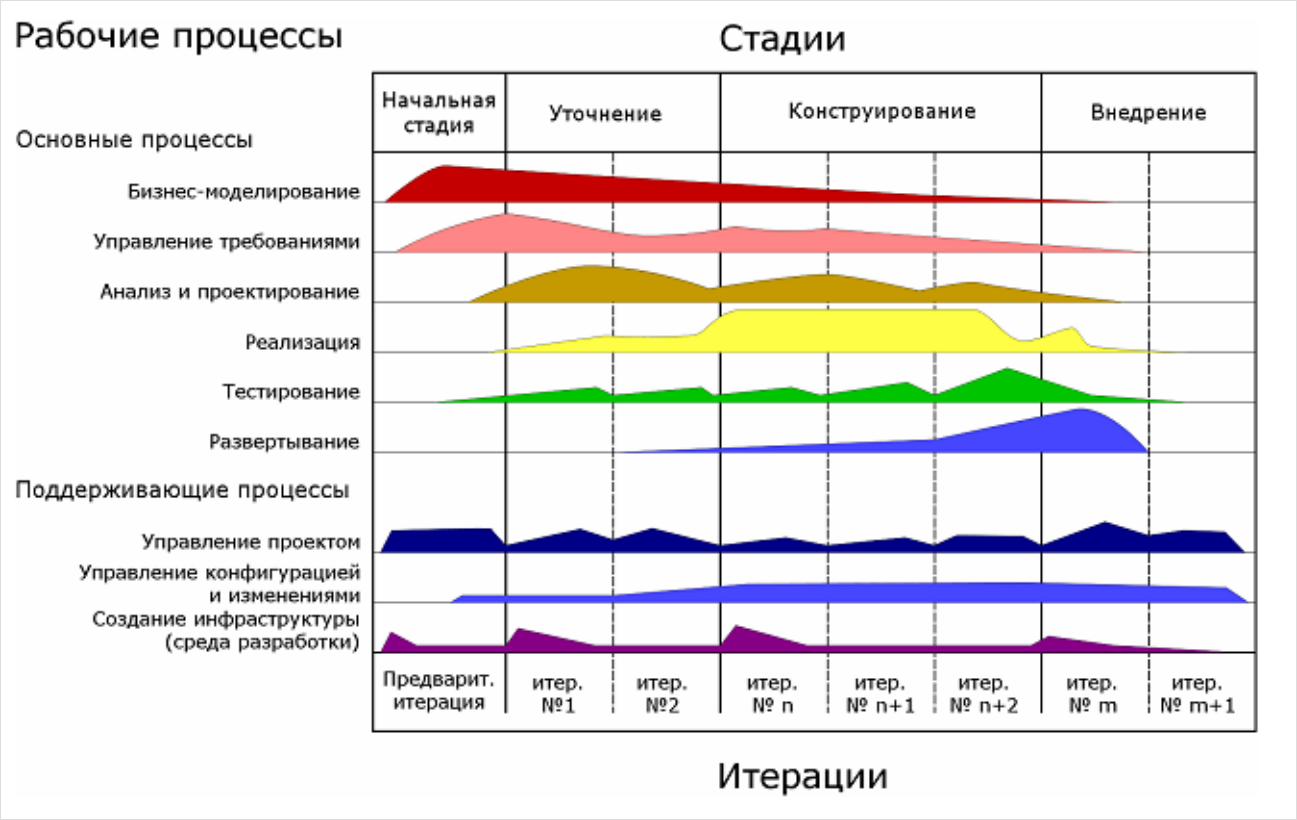


# Rational Unified Process (RUP)

Разработана Rational Software.

Методология основана на итеративной стратегии.

Используется как в небольших проектах, так и в больших и сложных.



# Эволюционная стратегия

- Эволюционная стратегия – если в начале разработки известны не все требования.
  - Спиральная модель.
  - Модель быстрой разработки RAD (Rapid Application Development): компонентная модель.
  - Методология Agile.
    - XP-процесс.
    - Scrum.
    - Kanban.
    - Lean.

# Спиральная модель

- 1988 г., Гарри Боэм

- Планирование
- Анализ риска

Линия принятия решений

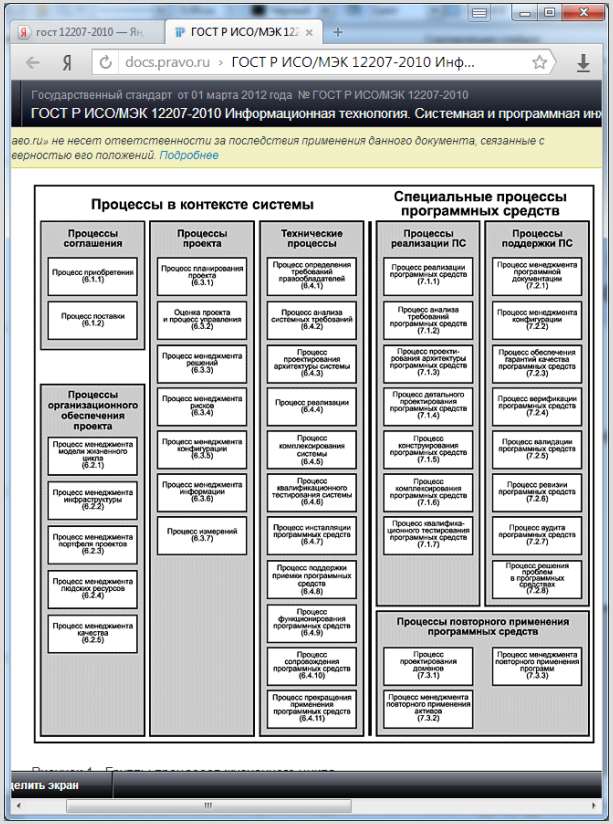
- Оценивание заказчиком
- Конструирование
- Переход к более полной версии

# XP-процесс

- eXtreme Programming – экстремальное программирование
- 1999 г., Кент Бек
- Основная задача – устранить высокую стоимость изменений ПО
- Используется в малых и средних по численности группах разработчиков (10 человек различной квалификации)
- Характеристики:
  - Малое время итерации (2 недели)
  - Малое приращение функциональности на итерации
  - Выход новой реализации – 2 месяца
  - Непрерывное и опережающее тестирование
  - Парное владение кодом

# Стандарты России в области программной инженерии

**ГОСТ Р 12207-2010 (ИСО/МЭК 12207-2008)** ИНФОРМАЦИОННАЯ ТЕХНОЛОГИЯ. СИСТЕМНАЯ И ПРОГРАММНАЯ ИНЖЕНЕРИЯ. ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА ПРОГРАММНЫХ СРЕДСТВ



# Стандарт ГОСТ Р 12207-2010

1. Процессы соглашения.
2. Процессы организационного обеспечения проекта.
3. Процессы проекта.
4. Технические процессы.
5. Процессы реализации программных средств.
6. Процессы поддержки программных средств.
7. Процессы повторного применения программных средств.

# Сбор и анализ требований к программному продукту

Этапы:

**Определение концепции продукта.**

Документ о концепции и границах проекта.

**Сбор требований.**

Документ о вариантах использования.

**Анализ требований.**

Модели анализа.

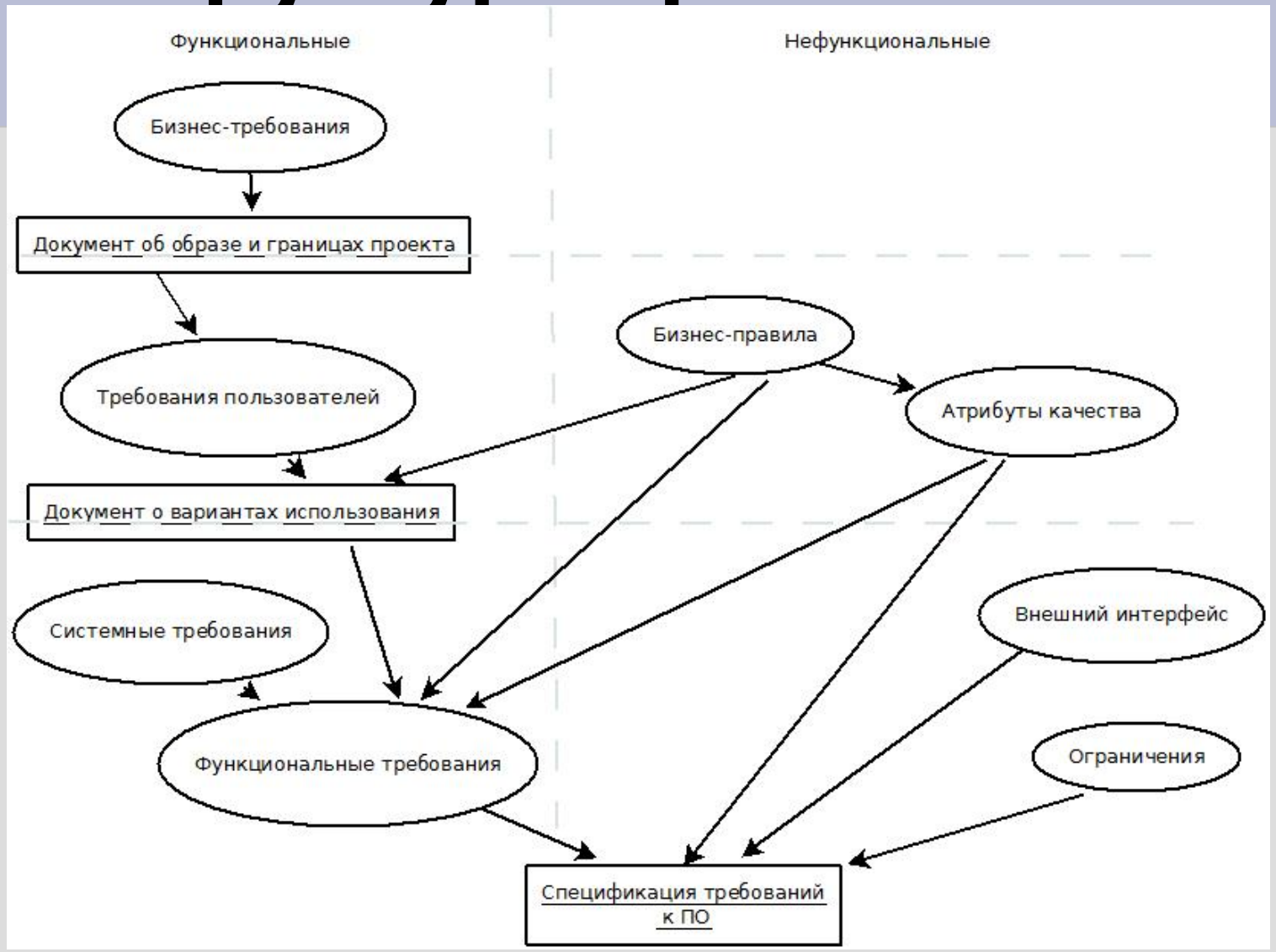
Спецификация требований к программному продукту.

**Итог: техническое задание и переход к этапу проектирования программного продукта.**

# Участники разработки требований

- Заказчики или инвестор
- Пользователи (подкласс заказчиков)
- Сотрудники правового отдела
- **Аналитики требований**
- Разработчики
- Тестировщики
- Технические писатели
- Менеджер проекта
- Производственники (внедрение)
- Сотрудники отдела продаж
- Сотрудники отдела технического обслуживания

# Структура требований




Карл Вигерс, Разработка требований к программному обеспечению

# Разработка концепции продукта

Участники этапа:

- бизнес-аналитик,
- инвестор (заказчик).

Цель: выработка единого видения проекта.

Документирование: документ о концепции и границах проекта (устав ПП, Product Vision Document, Market Requirement Document).

Содержат информацию о высокоуровневых требованиях и возможностях продукта, ориентировочные сроки реализации и бюджет.

Служит для: сделать вывод о целесообразности разработки.

# Сбор бизнес-требований для продукта

**Продукт под заказ**

- Определение исходных стимулов
- Определение целей продукта и критериев успеха
- Определение потребностей клиента
- Обзор конкурентов

**Продукт для открытого рынка**

- Определение исходных стимулов
- Обзор конкурентов
- Определение целевого сегмента рынка
- Определение потребностей клиентов
- Определение целей продукта и критериев успеха

# Исходные стимулы

- Потребность рынка.
- Производственная необходимость.
- Потребность заказчика.
- Технический прогресс.
- Юридические ограничения и нормы.

# Цели продукта и критерии успеха

- Финансовые

  - Освоить __ % рынка за __ месяцев.
  - Достигнуть объема продаж __ , дохода __.
  - Сэкономить __ на обслуживании системы.

- Нефинансовые

  - Разработать базовую технологическую основу для организации.
  - Соответствовать законодательной базе.
  - Повысить рейтинг организации.

# Обзор конкурентов

- Список проблем, которые должны быть решены в продуктах данного типа (что делает продукт).
- Список конкурентов, предлагающих продукты данного типа.
- Список продуктов конкурентов для обзора.
- Документация к продуктам или сами продукты.
- Список возможностей продуктов конкурентов (как работает продукт).
- Обобщение информации по конкурентам.

# Пример списка возможностей конкурентов

- Сравнительная характеристика XML-анализаторов

| Характеристика | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Проверка соответствия «заголовок-текст» | + | - | + | + | + |
| Проверка структуры документа | + | + | + | + | + |
| Интерфейс SAX | + | + | - | + | + |
| Интерфейс DOM | + | - | - | + | + |
| Открытый исходный код | - | + | - | - | - |
| Платформа Java | + | + | - | + | + |
| Платформа Windows | через Java-интерфейс | через Java-интерфейс | базовая платформа | через Java-интерфейс | через Java-интерфейс |
| Лицензия на коммерч. использование | предост. | предост | распростр. с приложениями ... | распростр. в виде комплекта разработчика | предост |

# Документ о концепции и границах проекта

- Бизнес-требования
  - Исходные данные
  - Возможности бизнеса
  - Бизнес-цели и критерии успеха
  - Положение о концепции
  - Бизнес-риски
  - Предположения и зависимости
- Рамки и ограничения проекта
  - Основные функции
  - Объем первоначальной и последующих версий
  - Ограничения и исключения
- Бизнес-контекст
  - Профили заинтересованных лиц
  - Приоритеты проекта
  - Особенности развертывания (операционная среда)

# Сбор требований пользователей

- Аналитик требований
- Выбор типичных представителей групп пользователей
- Опрос типичных представителей пользователей
- Семинар пользователь-разработчик
- Наблюдение за пользователями на рабочих местах
- Макетирование (создание прототипа)
  - Позволяет устранить неопределенность в требованиях заказчика
  - Позволяет проверить понимание задачи исполнителем (программистом)
  - Виды макета:
    - Рисунок
    - Макет в виде исполняемого файла

# Результат недостаточного взаимопонимания заказчика и разработчика

# Представление требований пользователя на основе варианта использования

- Идентификатор
- Имя (глагол+объект)
- Источник (автор)
- Дата создания
- Профиль пользователя
- Приоритет
- Частота использования
- Родительское бизнес-требование
- Предусловие (начальное состояние)
- Цель и результат
- Последовательность действий

# Представление требований в виде пользовательских историй

Пользовательская история — user story.
Содержание:
- текстовое описание функции;
- устные обсуждения истории;
- тесты.

Пример пользовательской истории:
- Пользователь может поместить свое резюме на веб-сайте.
- Пользователь может осуществить поиск вакансий.

# Пример карточки

лицевая сторона

Компания может оплатить публикацию вакансии с помощью кредитной карты.

Примечание: принимаются карты Visa, MasterCard, American Express.
Под вопросом Discovery.
Примечание к интерфейсу: поле для ввода типа карты не вводить, тип карты можно определить по первым двум цифрам ее номер

оборотная сторона

Тестировать с картами Visa, MasterCard, American Express. (+)
Тестировать с картами Dinner`s Club.(-)
Тестировать с просроченными платежными картами.
Тестировать с суммами до 100 $ и 100 $ и выше. Для добавления текста щёлкните мышью

# Представление системных событий и реакции на них

Таблица «событие-реакция»: управление кондиционером

| ID | Событие | Исходное состояние | Реакция системы |
|---|---|---|---|
| 1 | нажать на кнопку Вкл. | кондиционер включен пульт в сост. «включен» | кондиционер выключится |
| 2 | нажать на кнопку Вкл. | кондиционер выключен пульт в сост. «выключен» | кондиционер включится |
| 3 | нажать на кнопку Вкл. | кондиционер выключен пульт в сост. «включен» | кондиционер в прежнем сост. |
| 4 | нажать на кнопку Вкл. | кондиционер включен пульт в сост. «выключен» | кондиционер в прежнем сост. |

# Бизнес-правила

Типы бизнес-правил:

- факты,
- ограничения,
- активаторы операций,
- выводы,
- вычисления.

Источники бизнес-правил:

- законы государства,
- корпоративная политика,
- стандарты и нормативные документы,
- формулы,
- модели данных.

# Примеры бизнес-правил

Факт:
- Каждый товар имеет штрих-код.

Ограничение:
- Доставка всех заказов должна быть завершена между 10:00 и 14:00 по местному времени.

Активатор операции:
- После добавления покупателем товара в корзину предложить ему приобрести сопутствующие товары.

Вывод:
- Если товар данного вида на складе отсутствует, присвоить данному товару статус «нет в наличии».

Вычисления:
- Цена единицы товара снижается на 10 % при заказе более 6 единиц данного товара.

# Атрибуты качества (ГОСТ Р ИСО/МЭК 25010:2015)

- функциональная пригодность (правильность — соответствие спецификации, ТЗ; точность)
- надежность,
- удобство использования,
- защищенность,
- производительность,
- аппаратная и программная совместимость,
- переносимость,
- сопровождаемость.

# Спецификация требований

- Содержит функциональные требования.
- Каждое требование имеет уникальное имя и неизменно.
- Способы представления требований:
  - Структурированные описания на естественном языке
  - Графические описания
  - Формальные спецификации, использующие специальные языки (XML, DFD, ERD, STD, UML, блок-схемы)
- Стандарт IEEE 830-1998.

# Шаблон спецификации требований (Вигерс)

1. Введение.
2. Общее описание.
3. Функции системы.
4. Требования к данным.
5. Требования к внешним интерфейсам.
6. Атрибуты качества.
7. Требования по интернационализации и локализации.
8. Остальные требования.

Приложение А. Словарь терминов.

Приложение Б. Модели анализа.

# Техническое задание (ЕСПД. ГОСТ 19.201-78)

- Введение.
- Основание для разработки.
- Назначение разработки.
- Требования к программе.
- Требования к программной документации.
- Технико-экономические показатели.
- Стадии и этапы разработки.
- Порядок контроля и приемки.
- Приложения.

# ГОСТы, используемые при подготовке технического задания

ГОСТ 19.101-77 Виды программ и программных документов

ГОСТ 19.105-78 Общие требования к программным документам

ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом

ГОСТ 19.104-78 Основные надписи

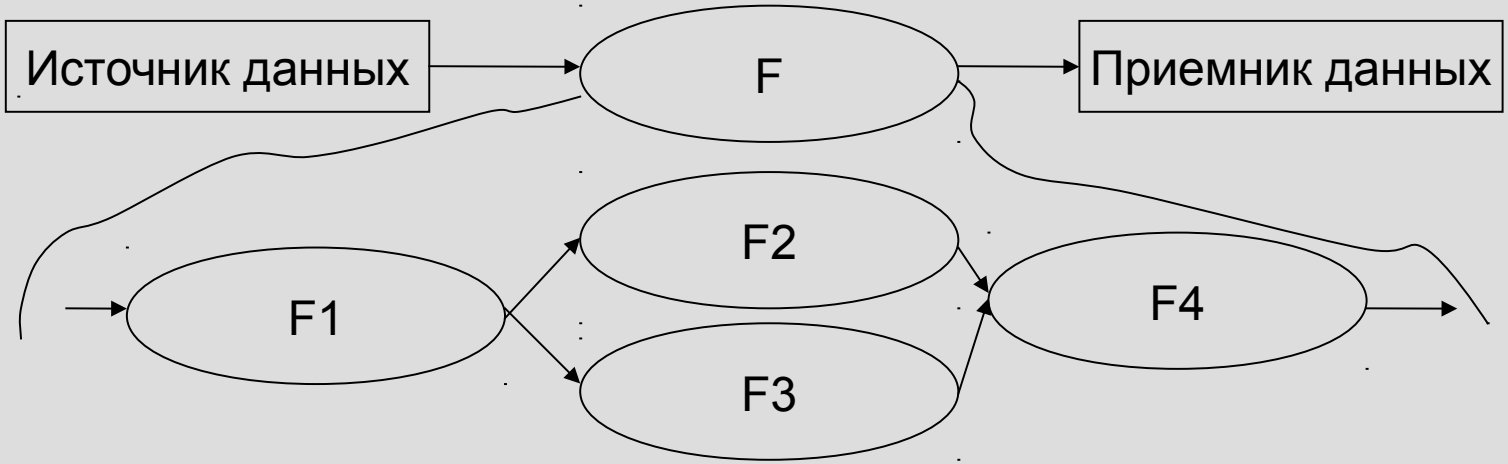
ГОСТ 34.602-89 Техническое задание на создание автоматизированной системы

# Методы анализа

- Процедурный подход
  - **Структурный анализ:**
  - **Диаграммы потоков данных (DFD)**
  - **Диаграммы «сущность-связь» (ERD)**
  - **Диаграммы состояний(STD)**
- Объектно-ориентированный подход
  - **Объектно-ориентированный анализ:**
  - **Диаграммы UML**

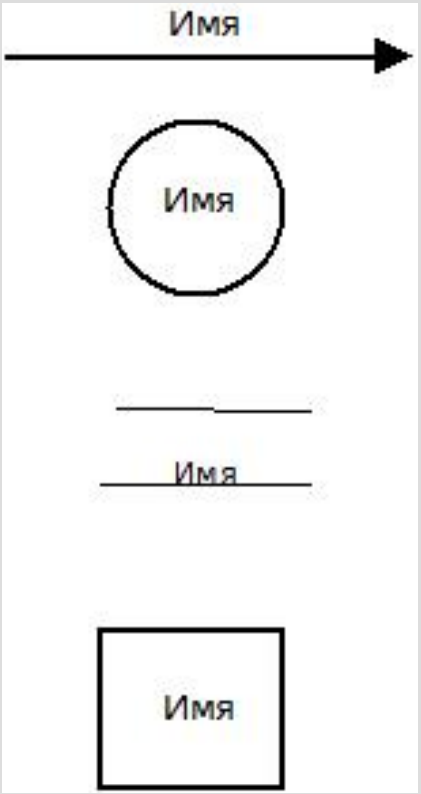
# Диаграммы потоков данных

- DFD — Data Flow diagram
- Основной метод: алгоритмическая декомпозиция задачи
- Элементы диаграммы: источник (потребитель) информации, процесс, поток данных (нотация Йодана, нотация Гейна-Сарсона)
- Любой процесс – преобразователь данных

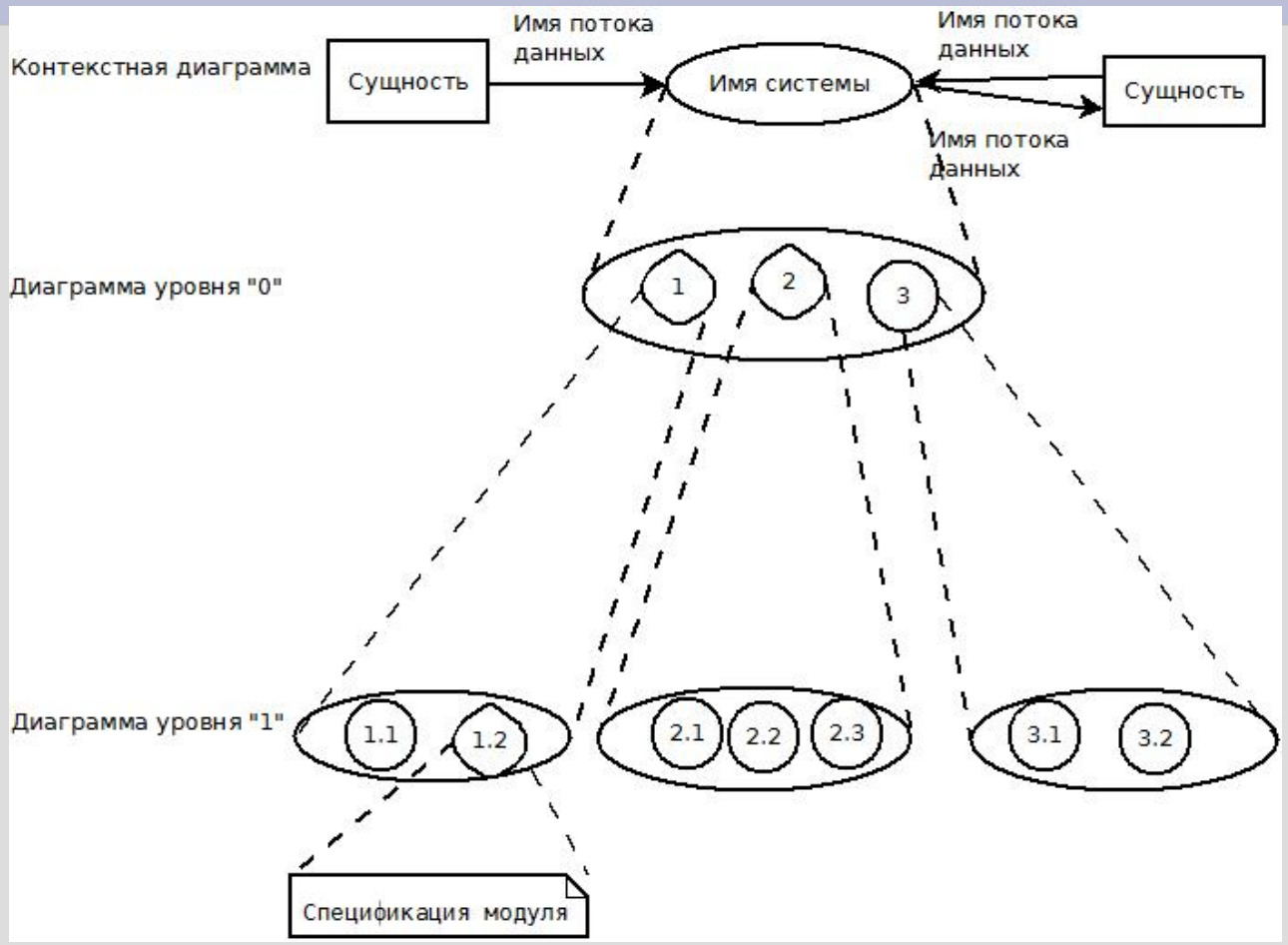


# DFD в нотации Йодана

- Элементы
  - поток данных
  - процесс
  - хранилище
  - внешняя сущность

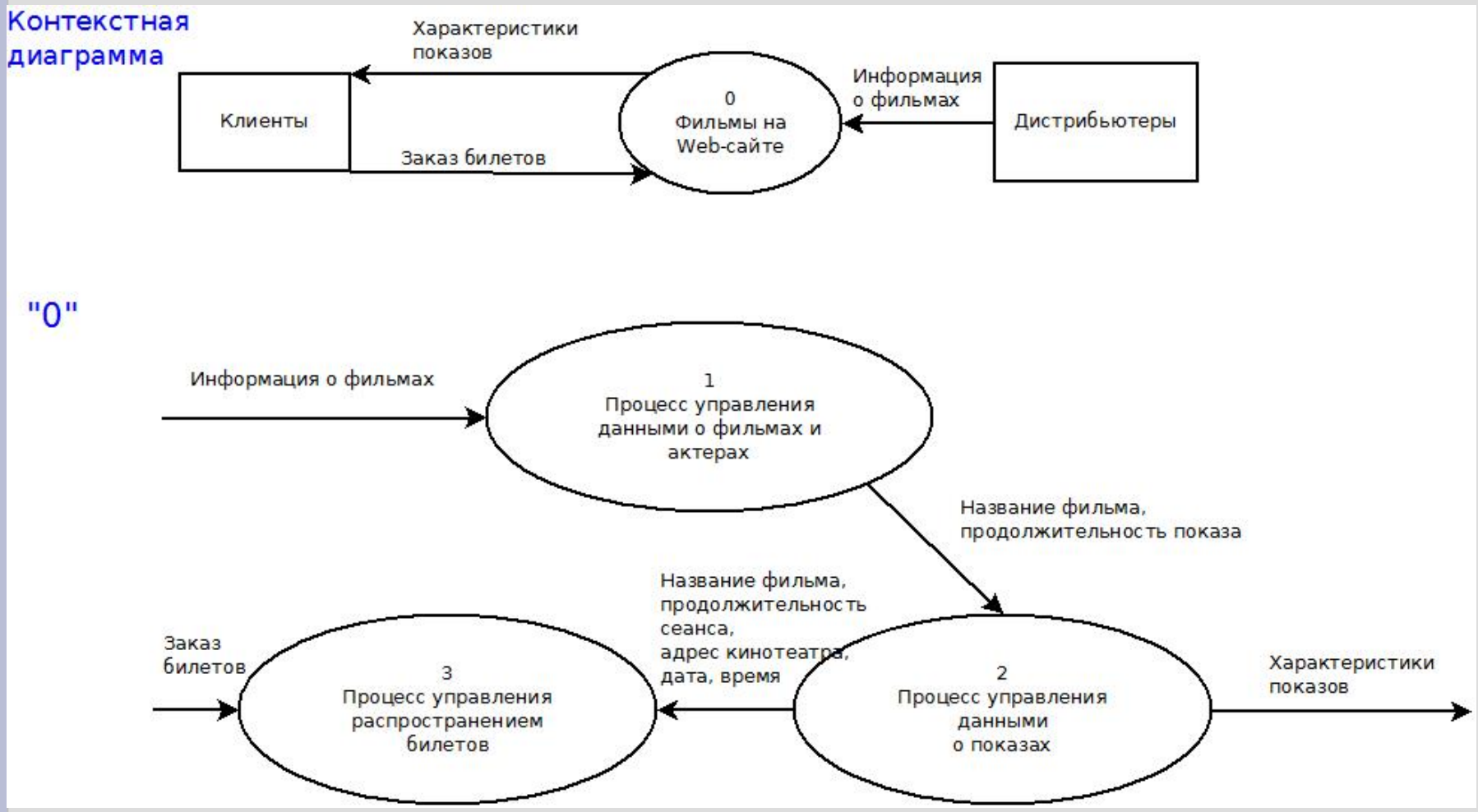


# Функциональная декомпозиция в DFD



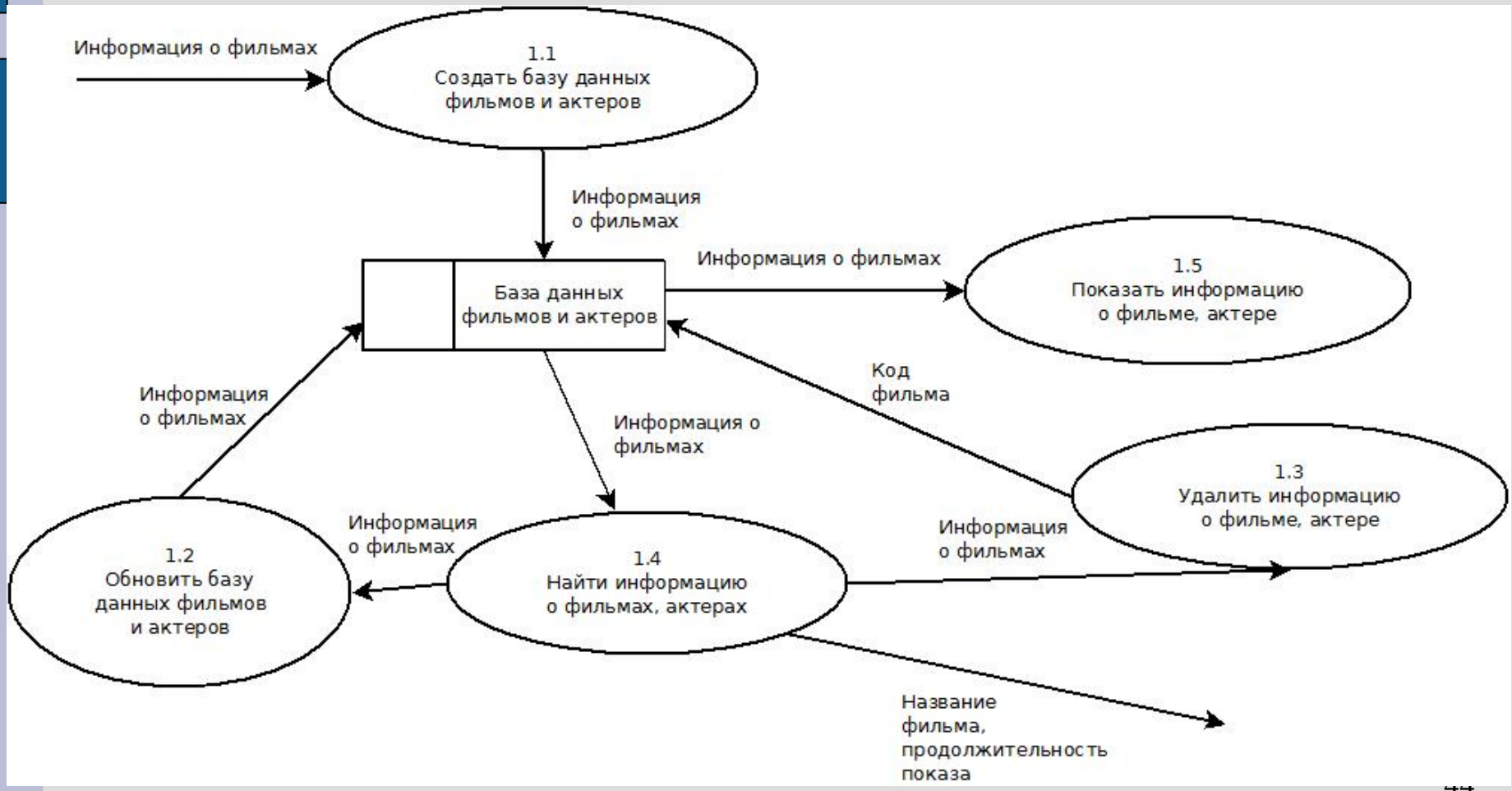
# Пример DFD (Йодан)

- Система «Фильмы на Web-сайте» (Л.А.Мацяшек, Б.Л.Лионг, Практическая программная инженерия, 2010)



# Пример DFD (Йодан)

- DFD 1 уровня

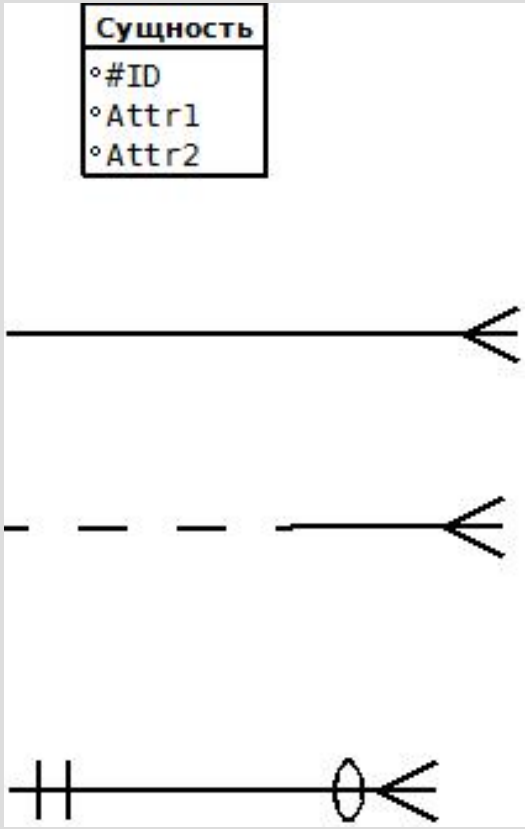


# Средства анализа данных

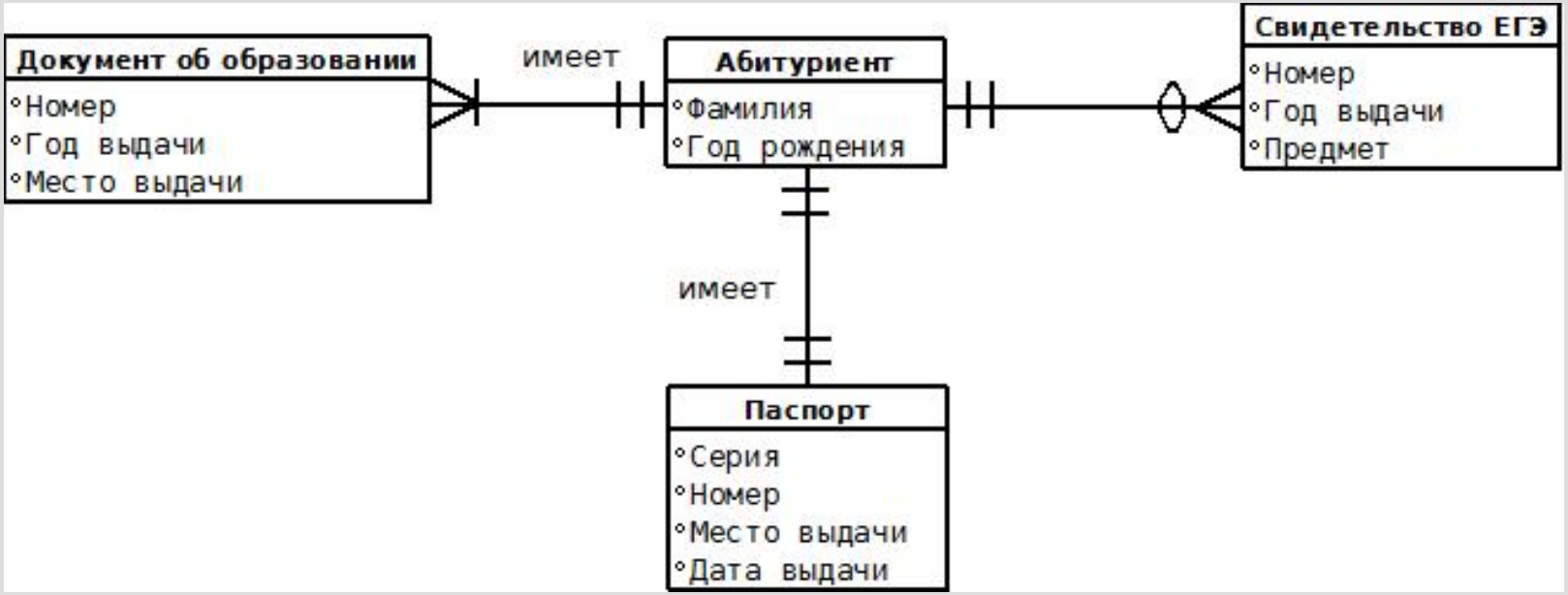
- Диаграммы «Сущность-связь»
- ERD — Entity-Relationship Diagrams
- Назначение: построение ER-моделей, не зависящих от типа СУБД (концептуальная модель)
- Нотация Чена, нотация Баркера
- Основные элементы:
  - **сущность,**
  - **экземпляр сущности,**
  - **атрибут,**
  - **связь.**

# ERD в нотации Баркера

- Ричард Баркер и др., 1981 год
- Публикация: Richard Barker (1990). CASE Method: Entity Relationship Modelling.
- Элементы
  - сущность и атрибуты
  - обязательная связь
  - необязательная связь
  - вариант необязательной связи



# Пример ERD (в нотации Баркера)



# Средства анализа поведения системы

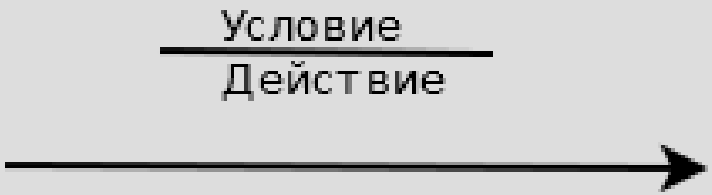
- Диаграммы переходов состояний
- STD — State Transition Diagrams
- Назначение: описание поведения системы во времени (спецификации управления)
- Основные элементы:
  - Состояние
  - Начальное состояние (только одно)
  - Завершающее состояние (любое количество)
  - Переход — определяется событием, управляющим переходом, и действием, выполняемым при переходе

# Обозначения STD

- Состояние

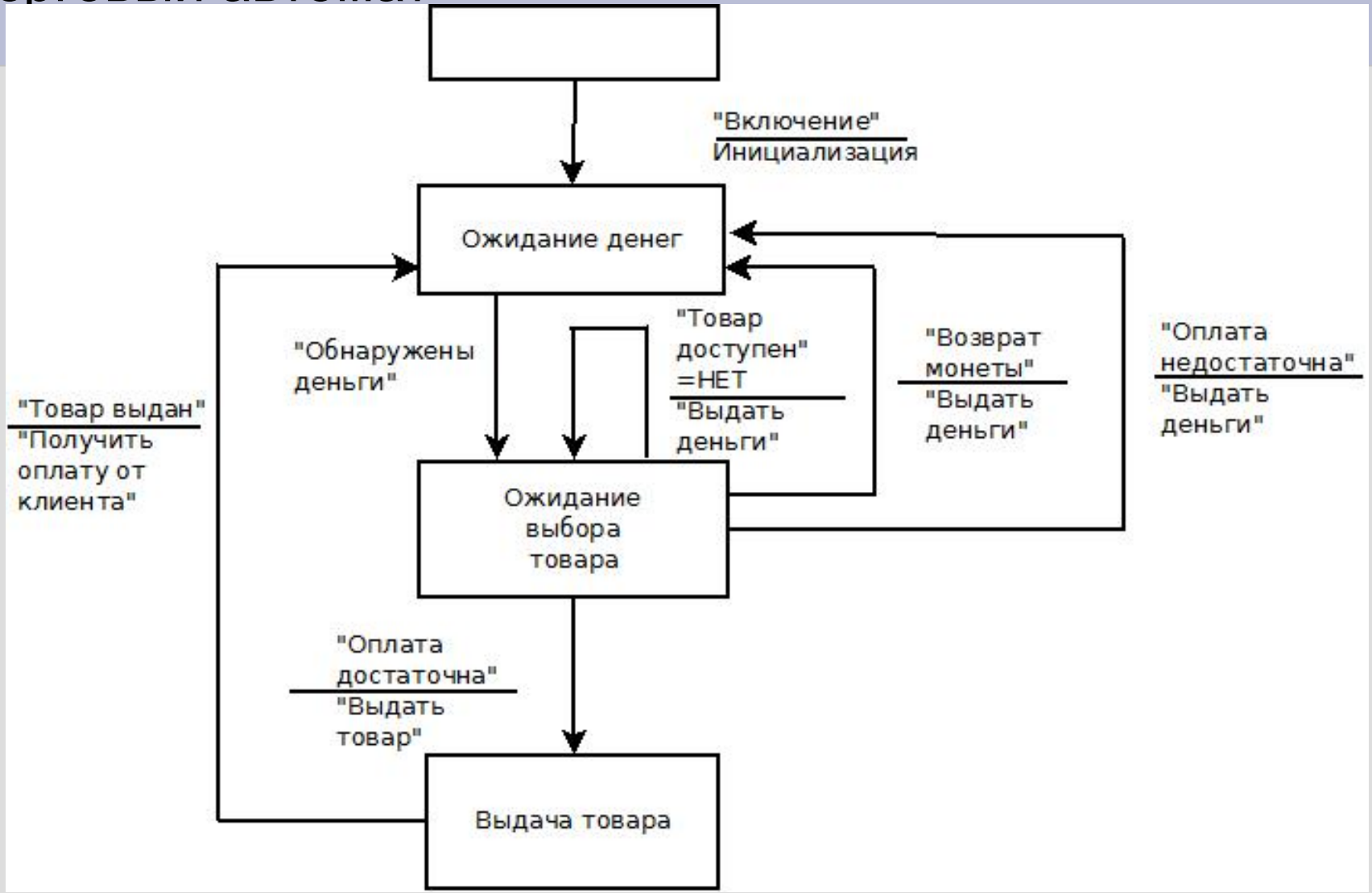

- Начальное состояние

- Переход



# Пример STD

- Торговый автомат



# Схемы по стандарту ГОСТ 19.701-90

- ГОСТ 19.701-90 (ИСО 5807-85) ЕСПД. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения
- Схема данных
- Схема программ (блок-схемы)
- Схема работы системы
- Схема взаимодействия программ
- Схема ресурсов системы

# Стандарты

- ГОСТ Р ИСО/МЭК 8631-94 Информационная технология. Программные конструктивы и условные обозначения для их представления. Описывает представление процедурных алгоритмов.
- ГОСТ 19781-90 Обеспечение систем обработки информации программное. Термины и определения. Разработан взамен ГОСТ 19781-83 и ГОСТ 19.004-80 и устанавливает термины и определения понятий в области программного обеспечения (ПО) систем обработки данных (СОД), применяемые во всех видах документации и литературы, входящих в сферу работ по стандартизации или использующих результаты этих работ.
- ГОСТ Р 51904-2002 Программное обеспечение встроенных систем. Общие требования к разработке и документированию.

# Стандарты языка UML

- Язык UML – Unified Modeling Language.
- Авторы: Гради Буч, Джеймс Румбо, Айвар Якобсон.
- OMG — Object Management Group.
- UML 2.4.1 принят в качестве международного стандарта ISO/IEC 19505-1, 19505-2:2011.
- Спецификации UML:
    - 1.3 (1999 год),
    - 1.4 (2001 год),
    - 1.5 (2003 год),
    - 2.0 (2005 год),
    - 2.4 (2011 год),
    - 2.5 (2015 год).

# Диаграммы UML

1. Диаграмма прецедентов (Use Case)
2. Диаграмма классов
3. Диаграмма объектов
4. Диаграмма последовательности (действий)
5. Диаграмма сотрудничества (кооперации)
6. Диаграмма схем состояний
7. Диаграмма деятельности
8. Диаграмма коммуникации
9. Диаграмма компонентов
10. Диаграмма размещения (развертывания)

# Структурные предметы UML

Класс


Объект

Кооперация

(сотрудничество)

# Структурные предметы UML (2)

Актер

Прецедент
(Use Case)

Интерфейс

# Структурные предметы UML (3)

Активный класс

Компонент

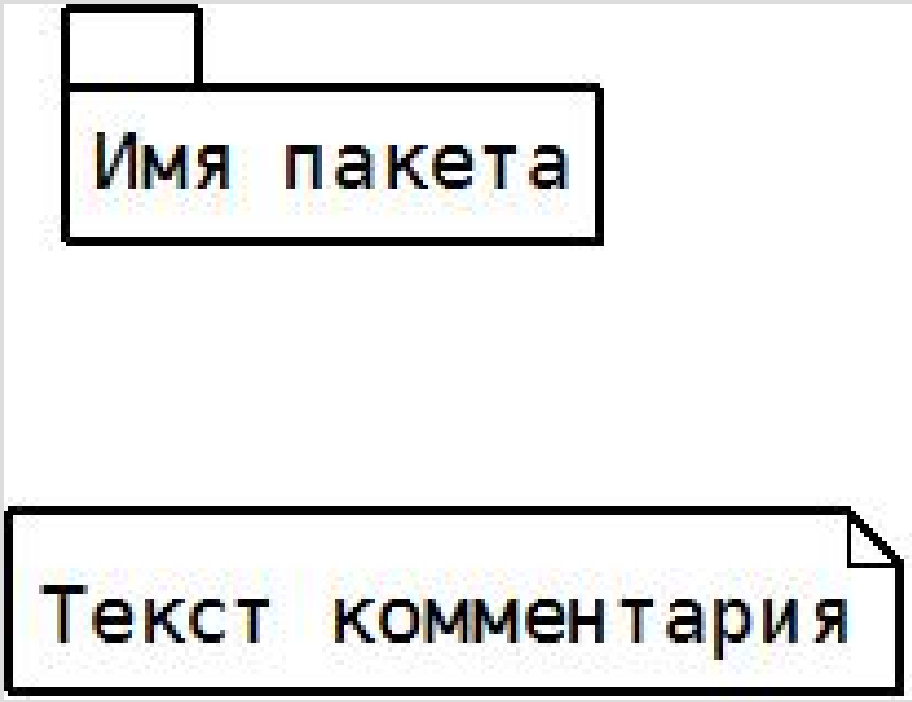
UML 1.2

UML 2.0

Узел

# Структурные предметы UML (4)

Пакеты




Комментарии (аннотации)

# Предметы поведения UML

Взаимодействие

Конечный автомат

Вид деятельности

# Отношения UML

Зависимость

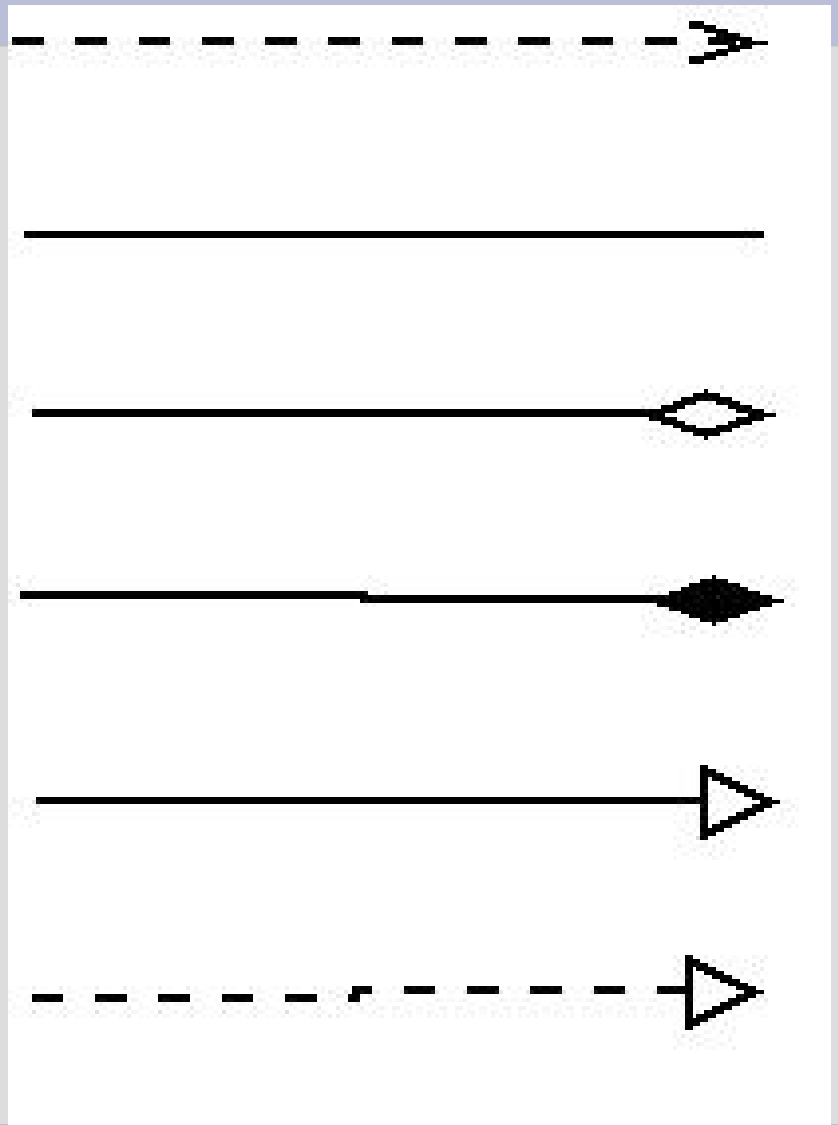
Ассоциация

Агрегация

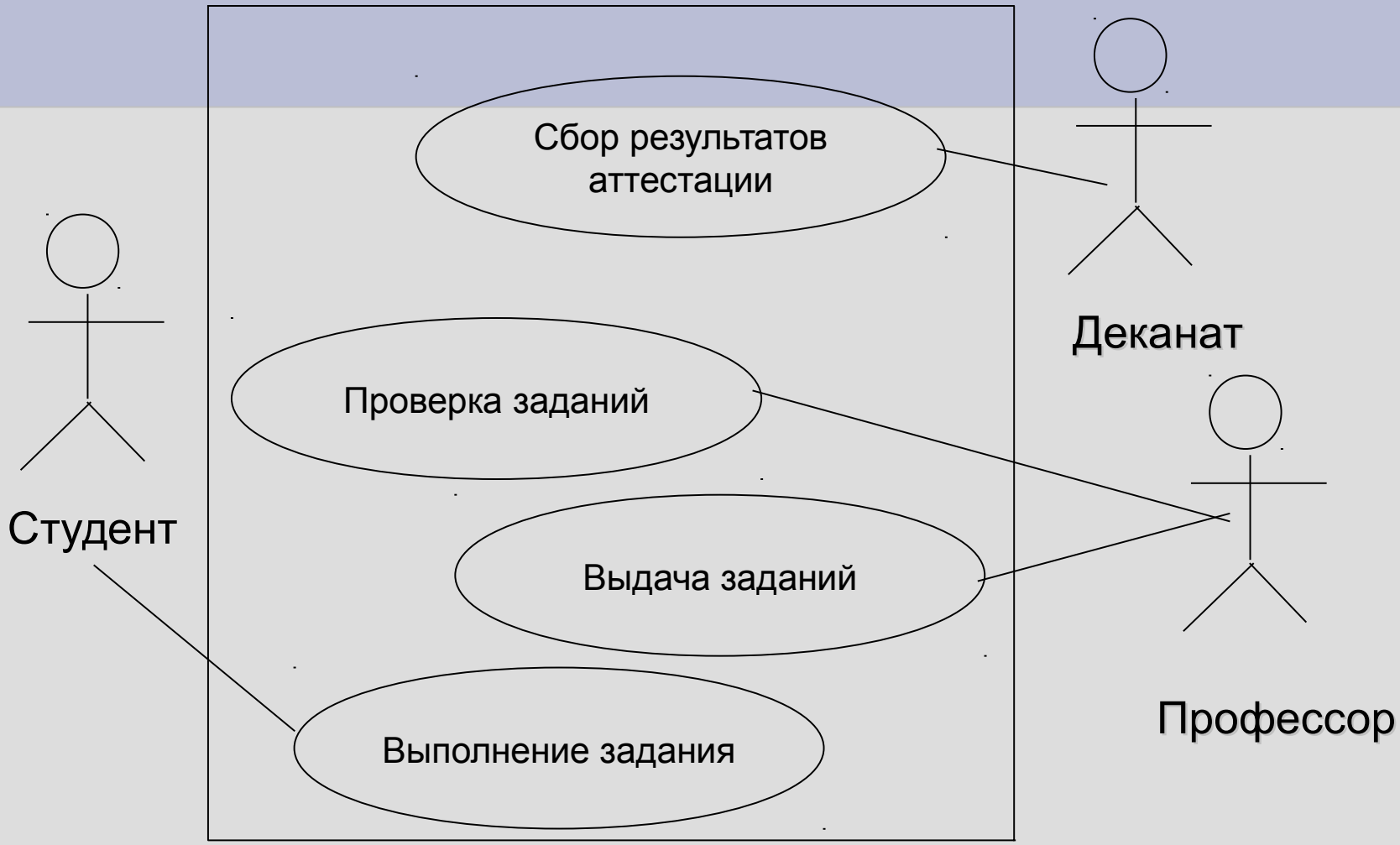
Композиция

Обобщение

Реализация



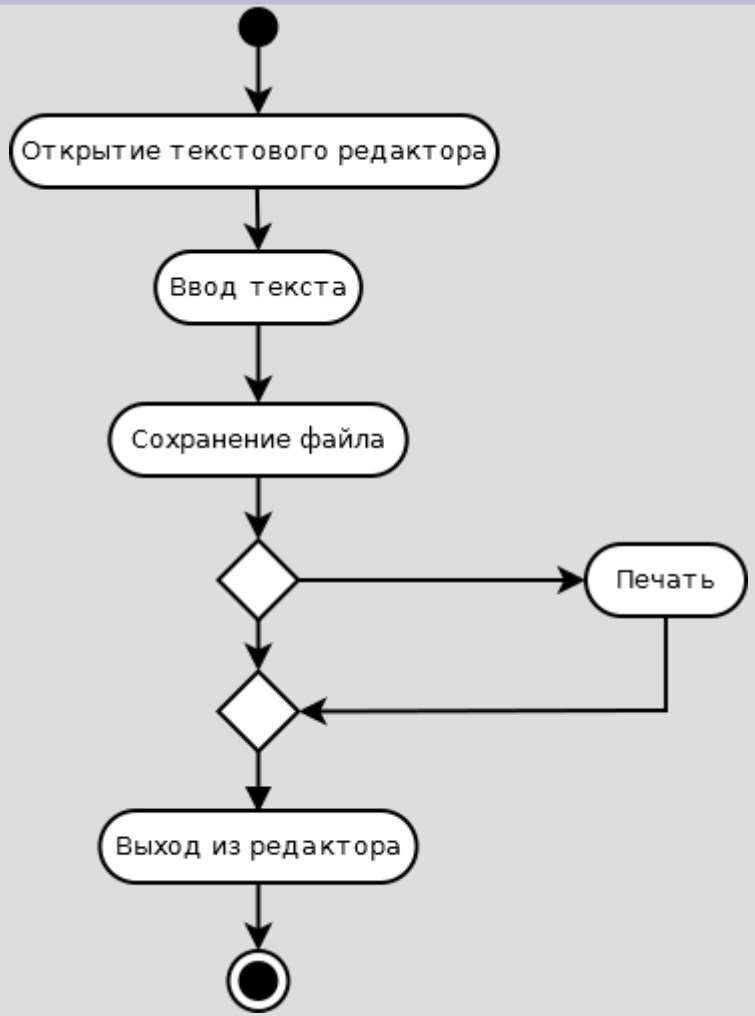
# Диаграмма Use Case



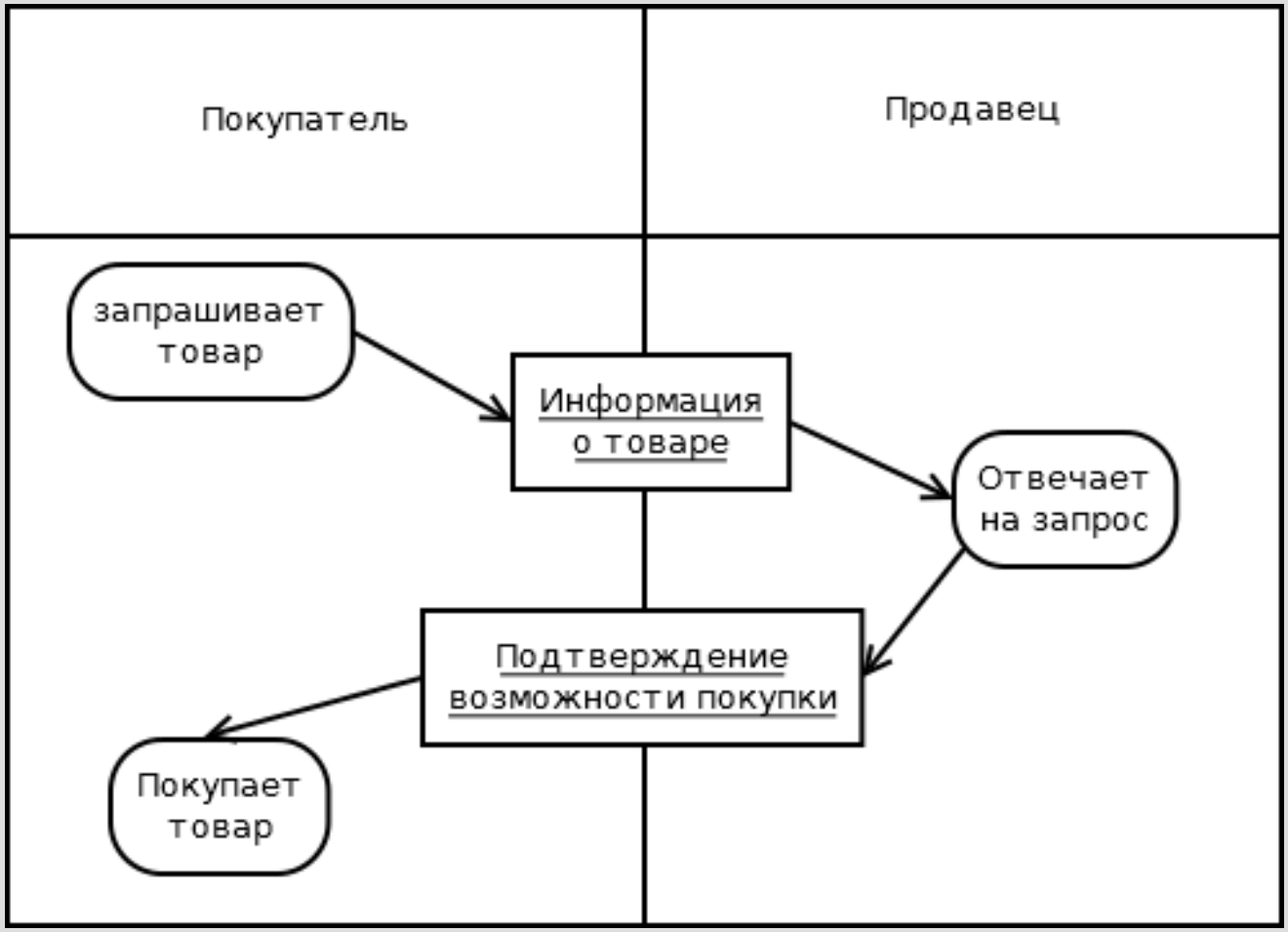

Между актером и прецедентом – ассоциативная связь

# Диаграмма Use Case (2)

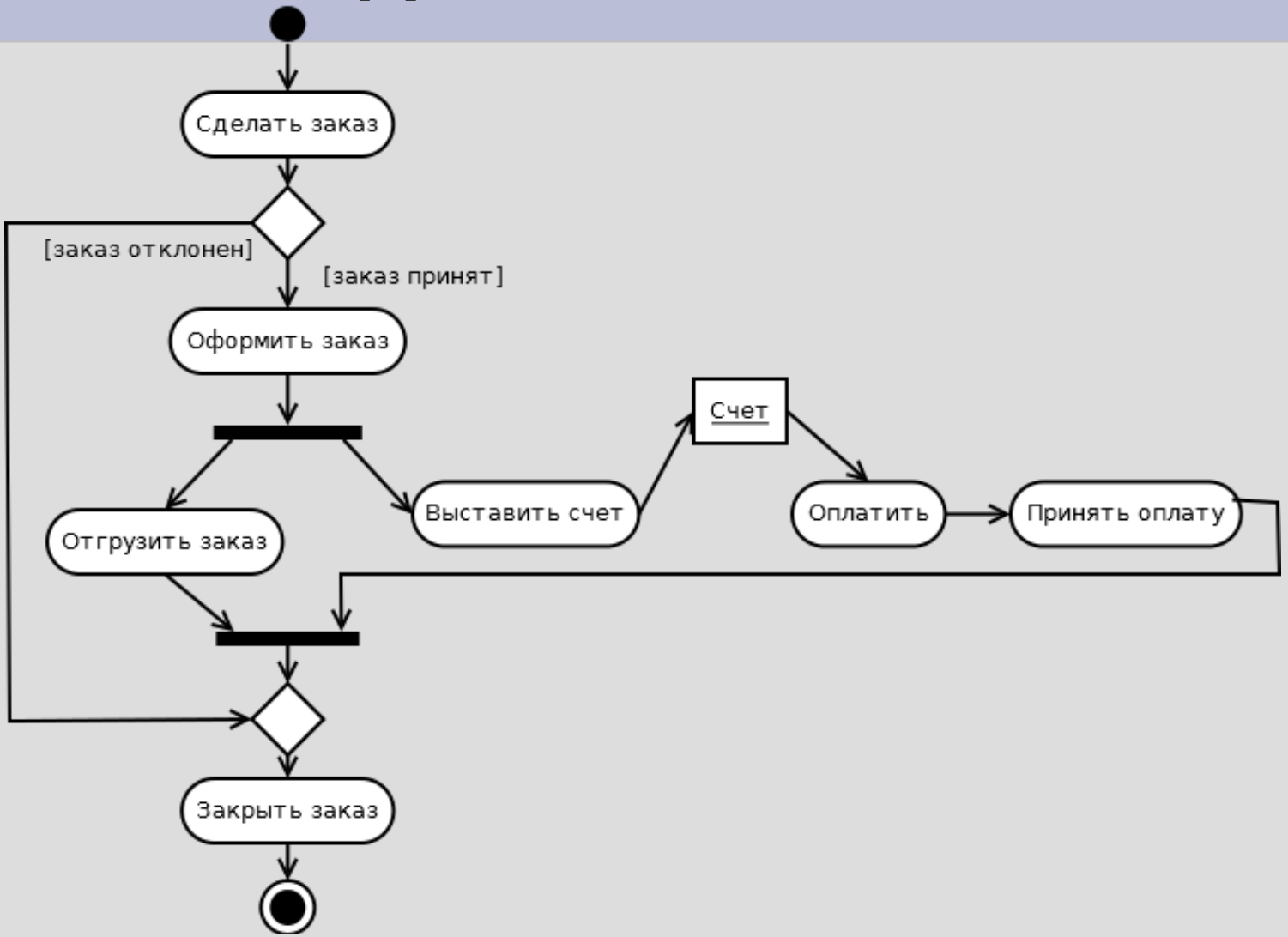
# Диаграмма деятельности



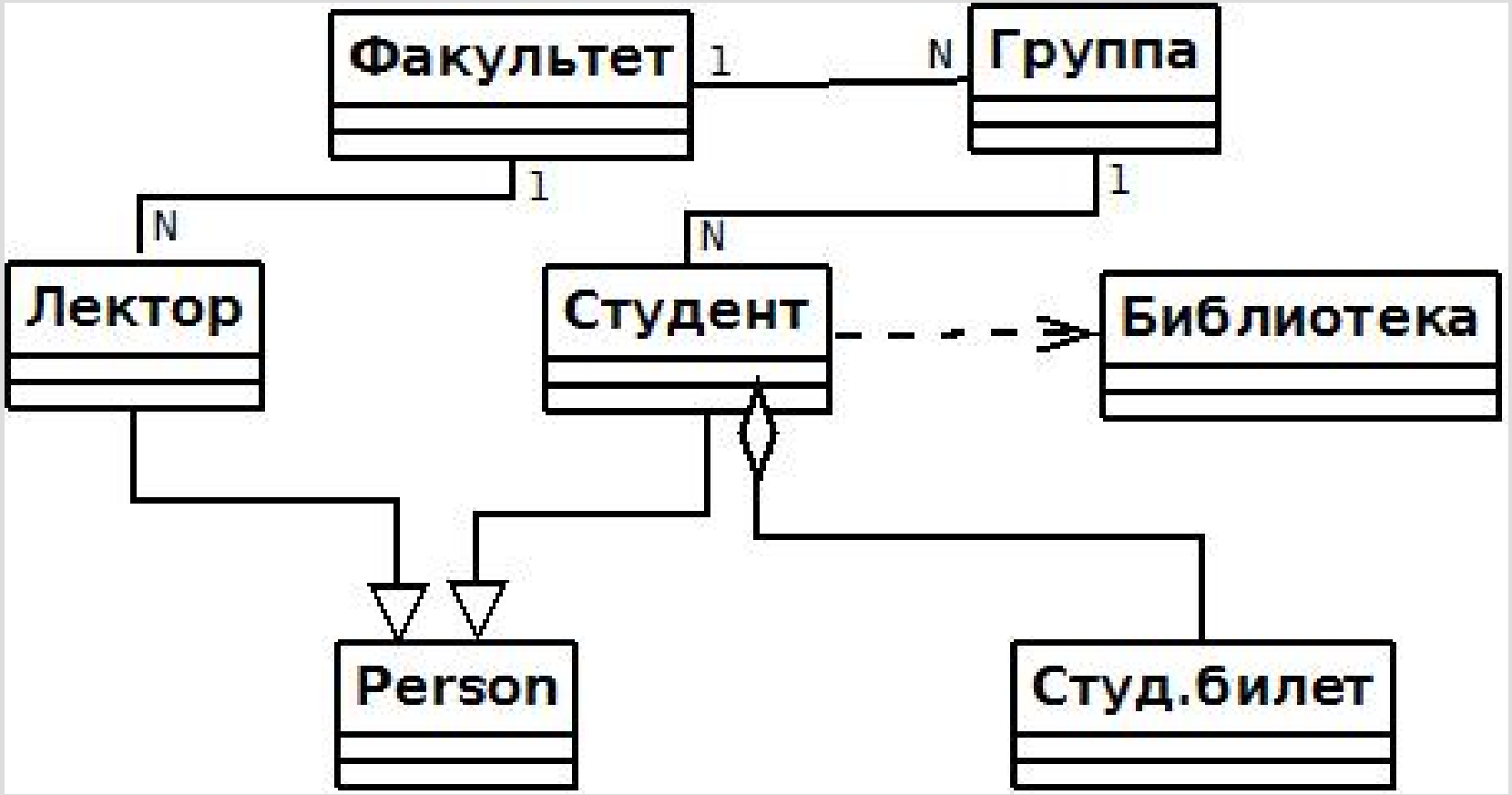
# Роли на диаграмме деятельности



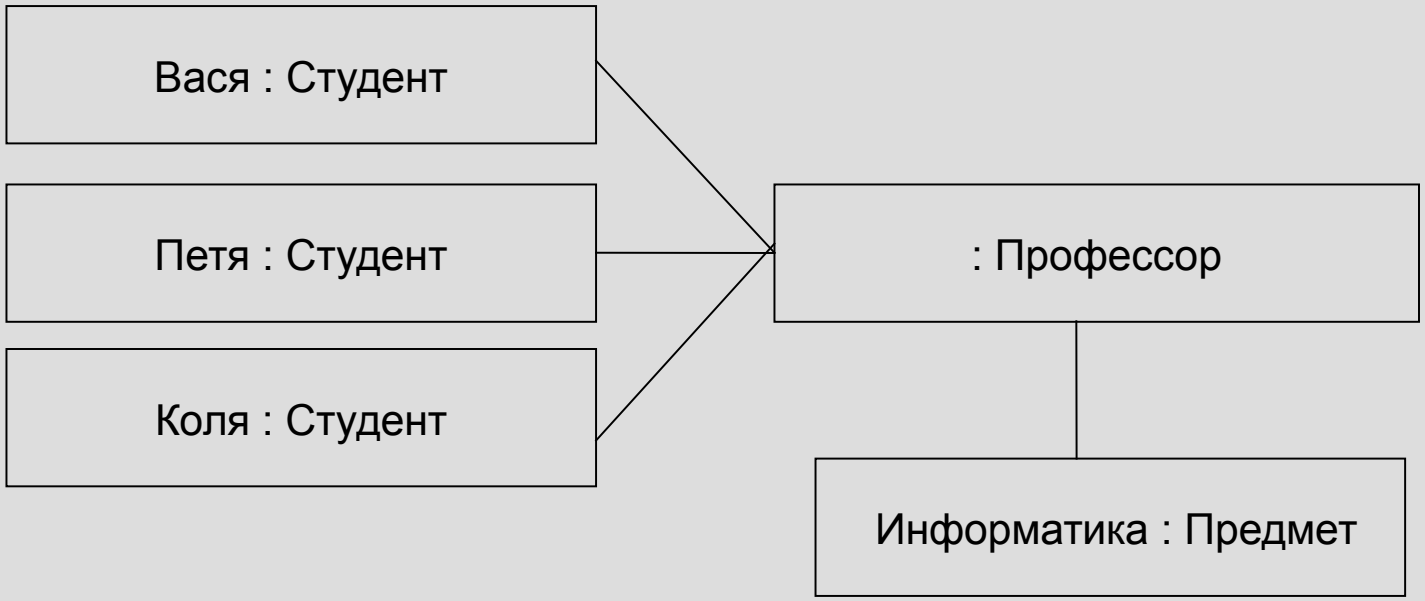
# Пример диаграммы деятельности



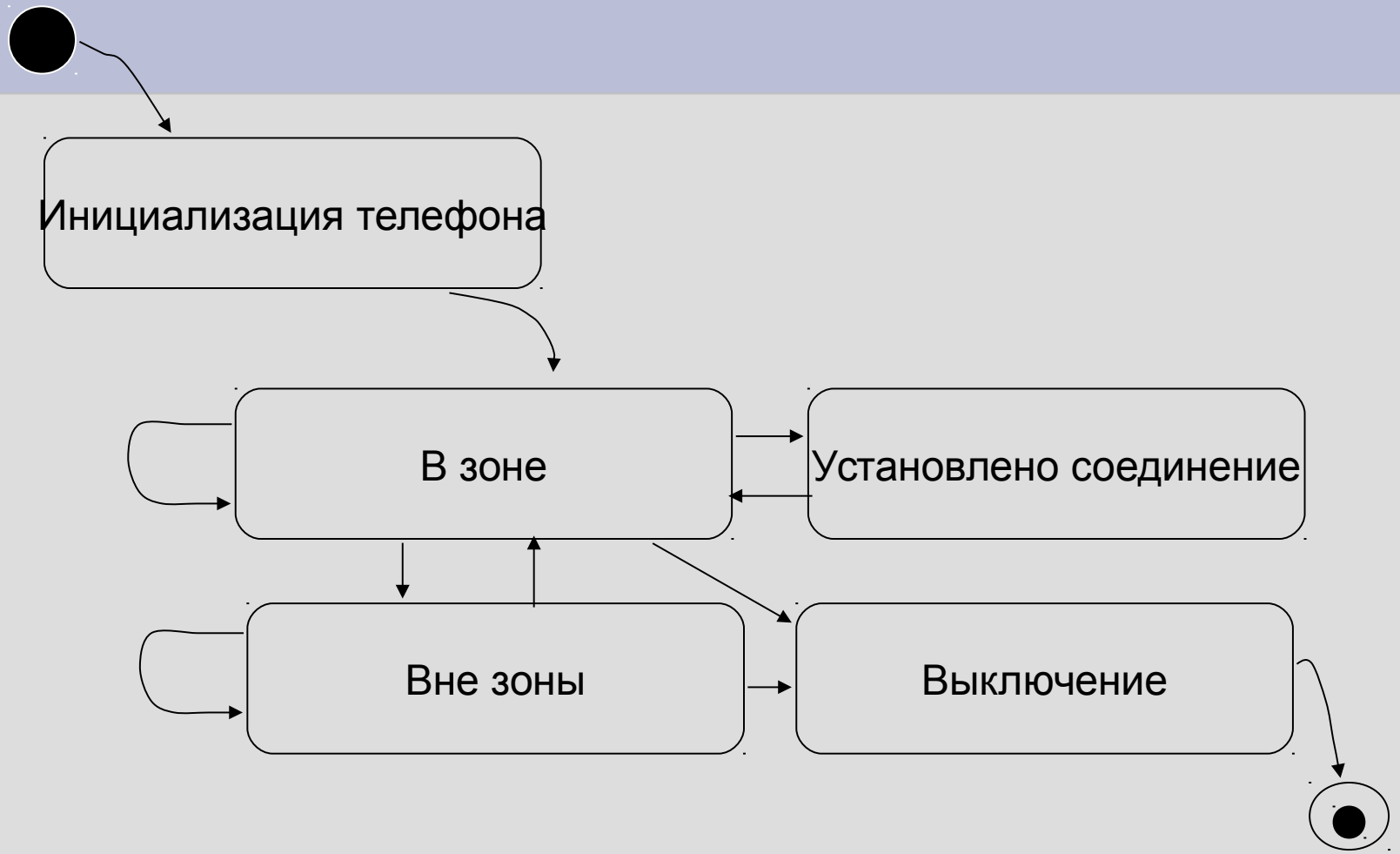
# Диаграмма классов



# Диаграмма объектов



# Диаграмма состояний

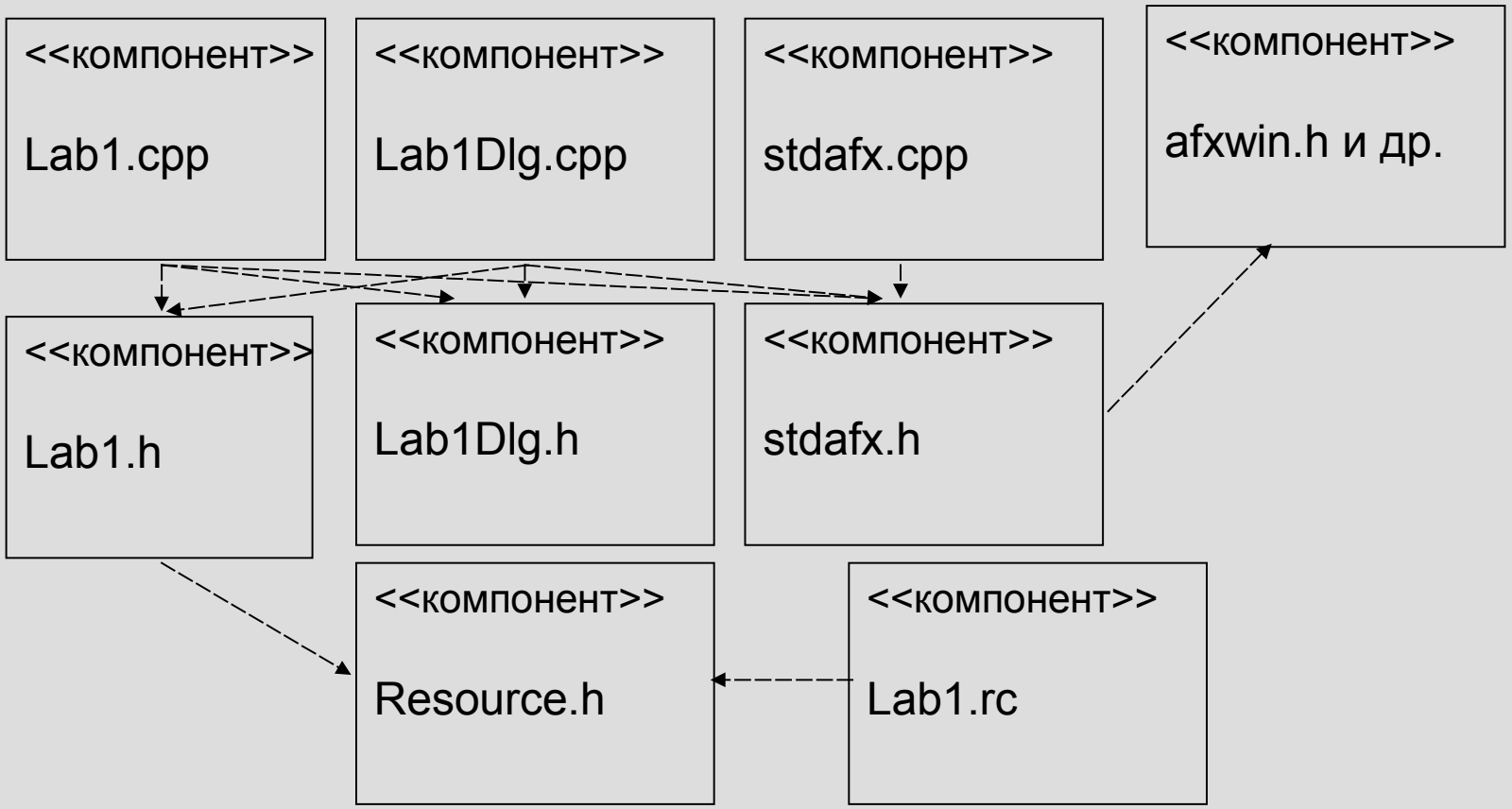


# Диаграмма последовательности

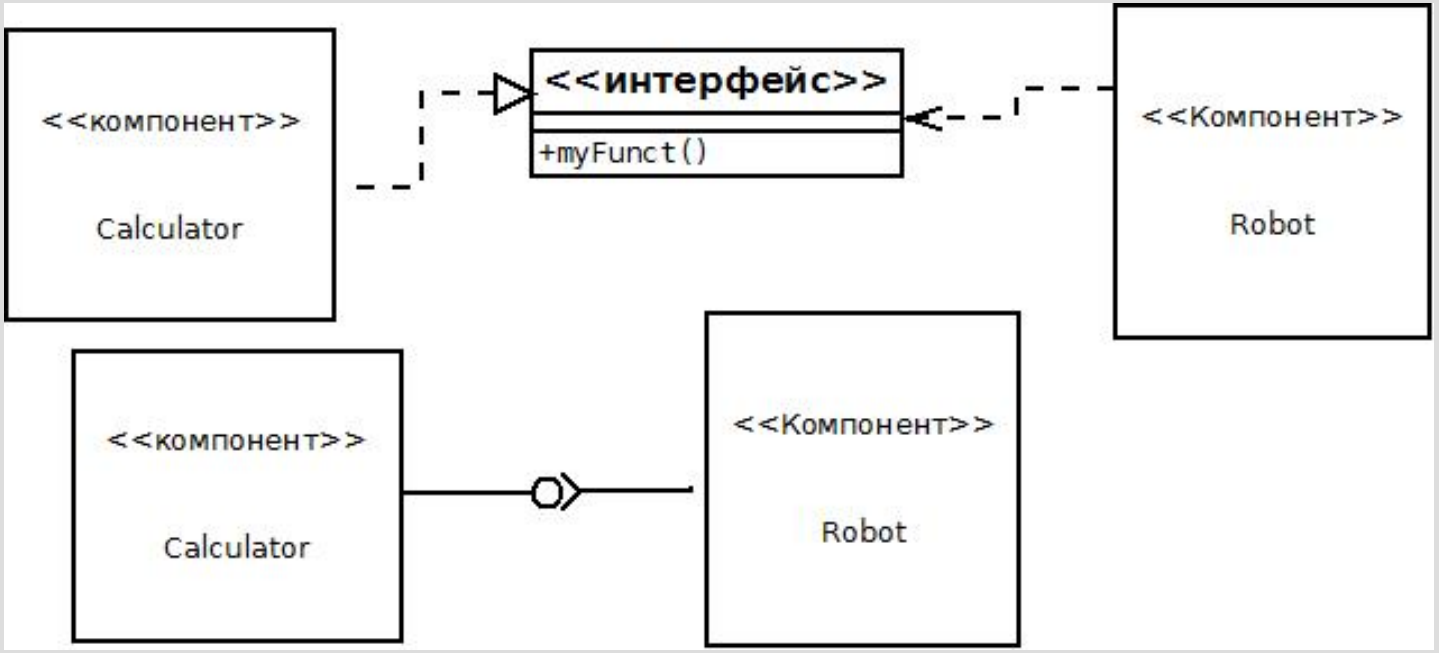
# Диаграмма компонентов

- Артефакт:
  - таблицы,
  - файлы данных,
  - исполняемые файлы,
  - документы,
  - динамически загружаемые библиотеки.
- Компонент:
  - программная реализация класса или нескольких классов

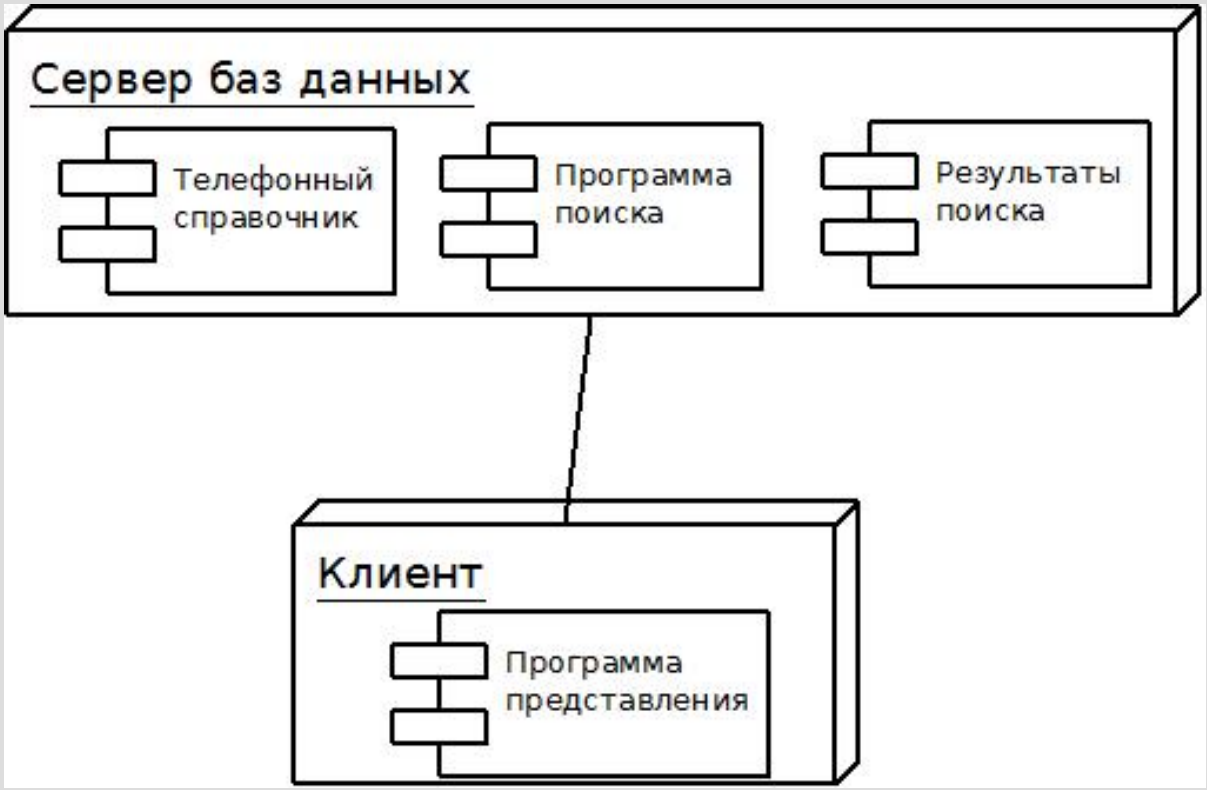

# Диаграмма компонентов



# Интерфейс


<<компонент>>
Calculator
<<интерфейс>>
+myFunct()
<<Компонент>>
Robot
<<компонент>>
Calculator
<<Компонент>>
Robot

# Диаграмма развертывания



# Вопросы

- Где можно применять диаграммы UML?
- Какие виды диаграмм наилучшим образом представляют структуру ПО?
- Какие виды диаграмм представляют описание ПО во времени?
- Какие виды диаграмм наилучшим образом представляют требования к ПО?

# Итоги анализа

- Итоги анализа – спецификация требований
- Спецификация требований + результаты планирования = техническое задание

# Методы проектирования

# Проектирование ПО

- Проектирование – это процесс, обратный анализу (синтез)
- На основе анализа формируются:
  - Информационная модель (данные)
  - Функциональная модель (действия)
  - Поведенческая модель (режимы работы системы)
- Направления проектирования:
  - Разработка данных
  - Разработка архитектуры
  - Процедурная разработка
  - Разработка интерфейсов пользователя
- Результаты проектирования используются на этапе кодирования

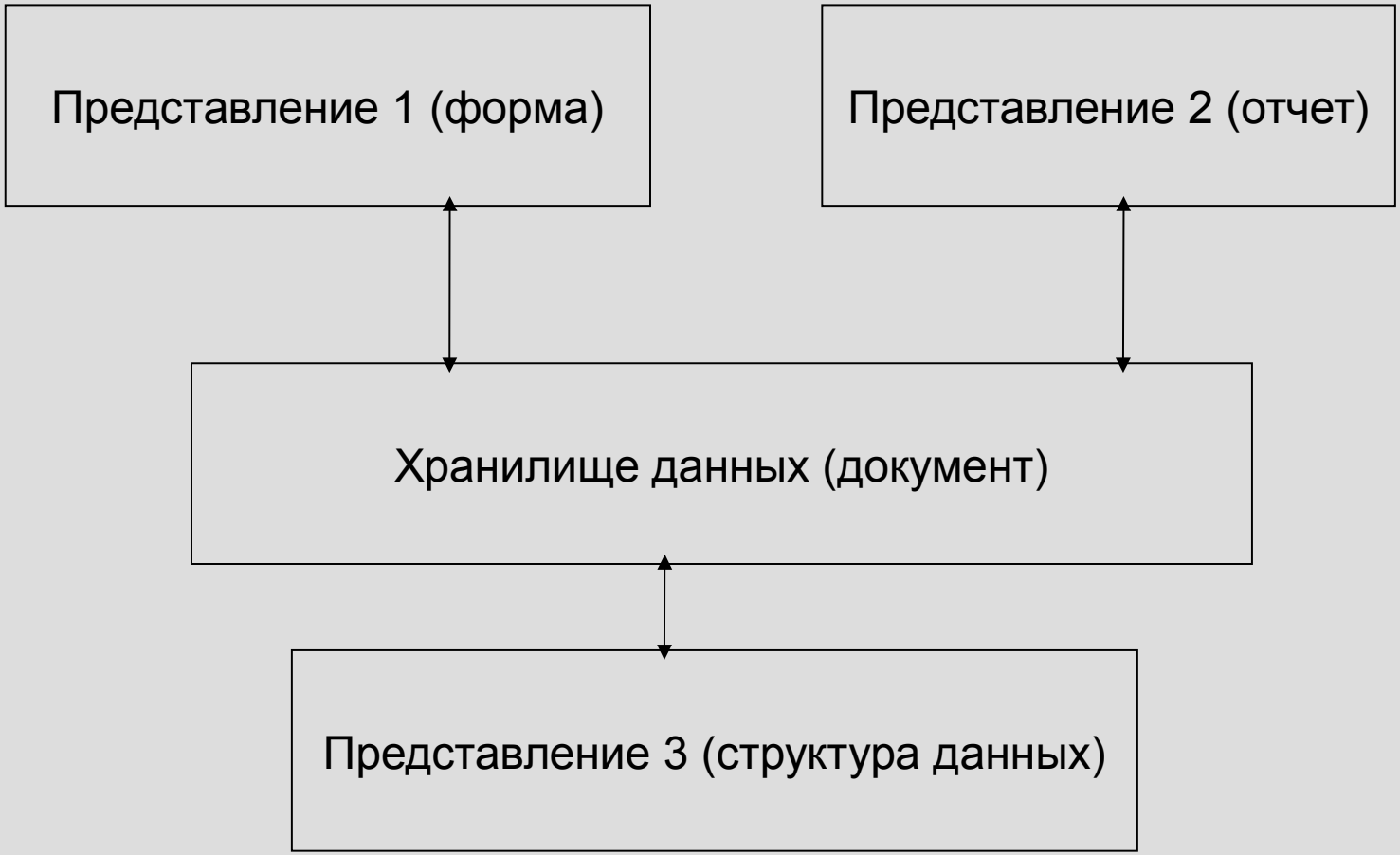
# Этапы проектирования

- Предварительное (абстракции уровня архитектуры)
  - Результат: структура системы (деление на подсистемы), модель управления (взаимодействие между подсистемами), модульная декомпозиция подсистем
- Детальное (структуры данных и алгоритмы)
- Интерфейс пользователя

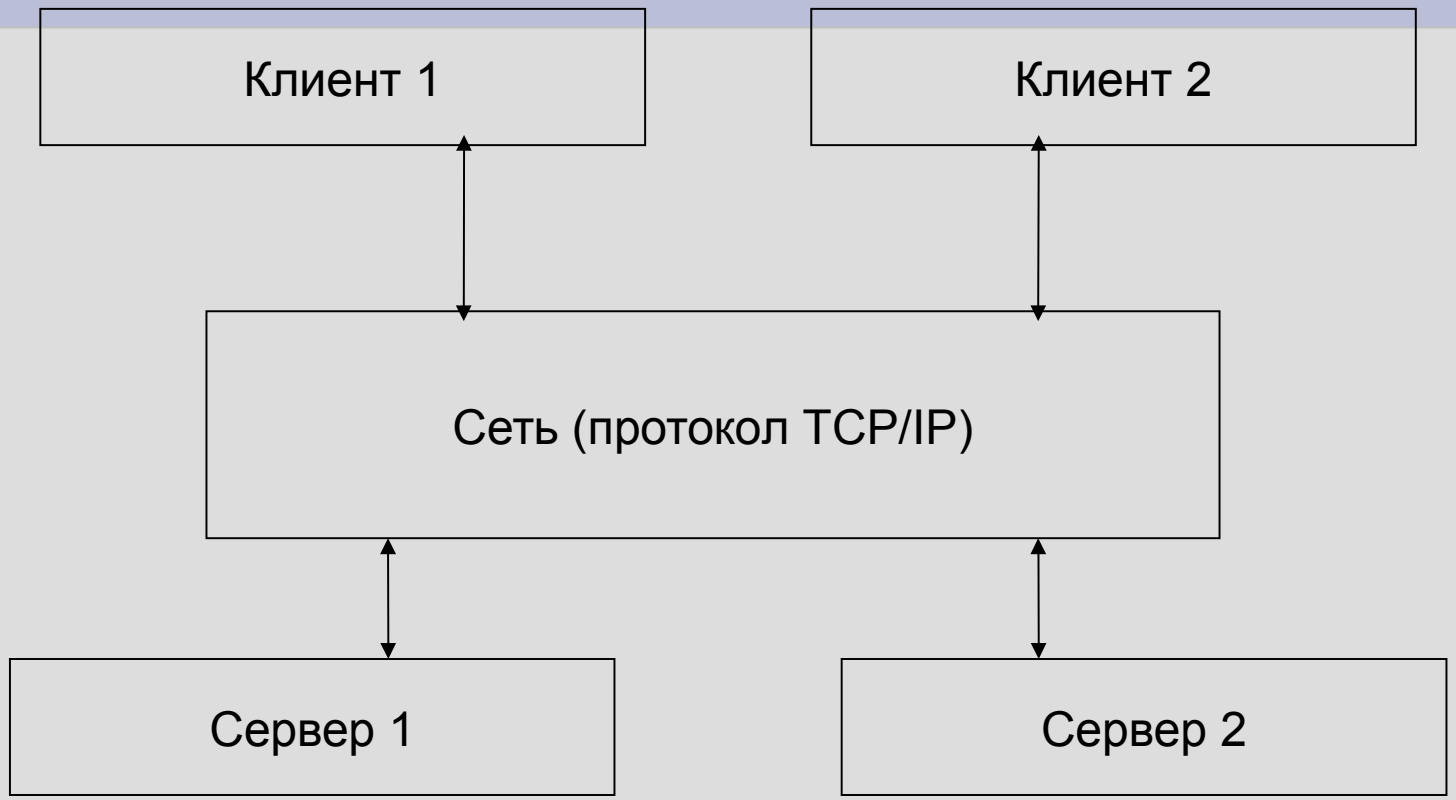
# Структура ПО

- Модель хранилища данных
- Модель клиент-сервер
- Трехуровневая модель
- Модель абстрактной машины
- Многоуровневая модель (программно-аппаратный стек)

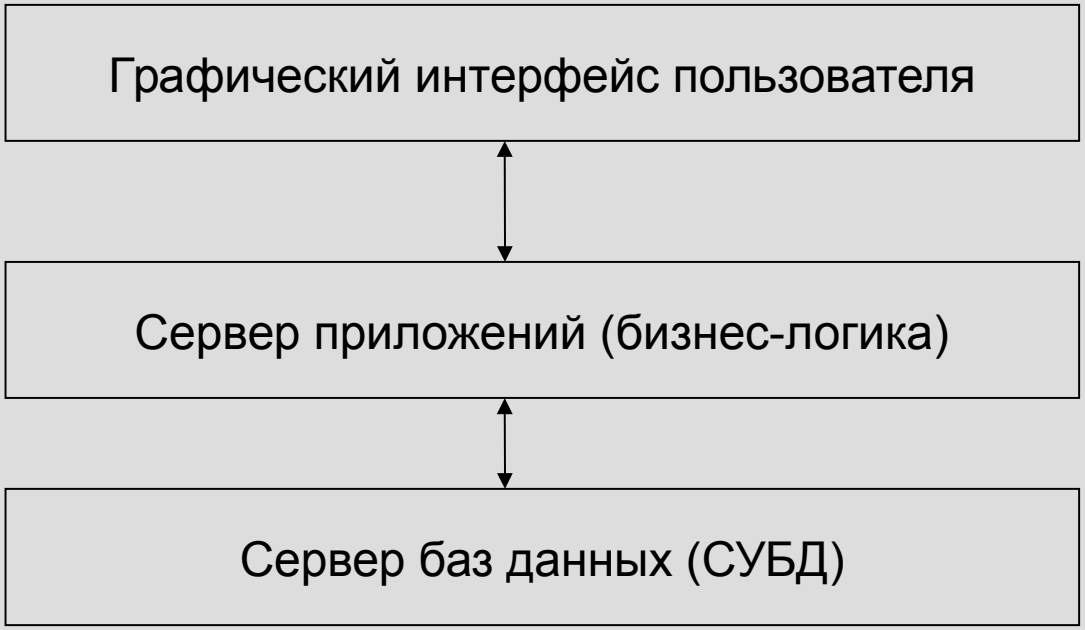
# Модель хранилища данных



# Модель клиент-сервер



# Трехуровневая модель



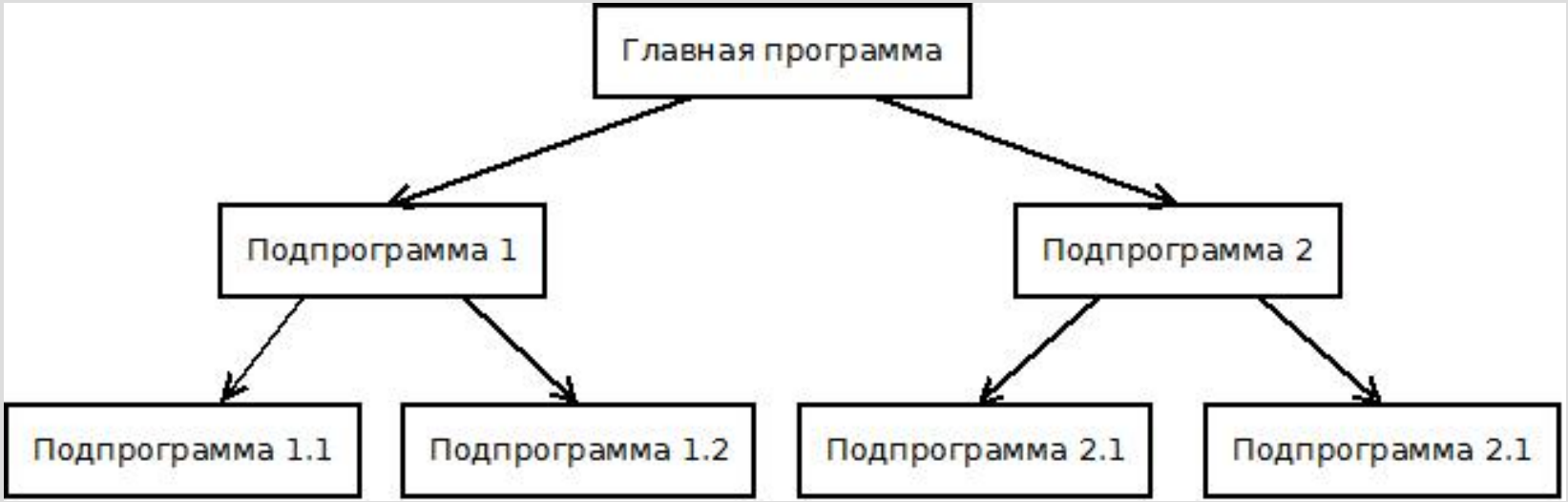
# Модель абстрактной машины

# Программно-аппаратный стек (многоуровневая структура)

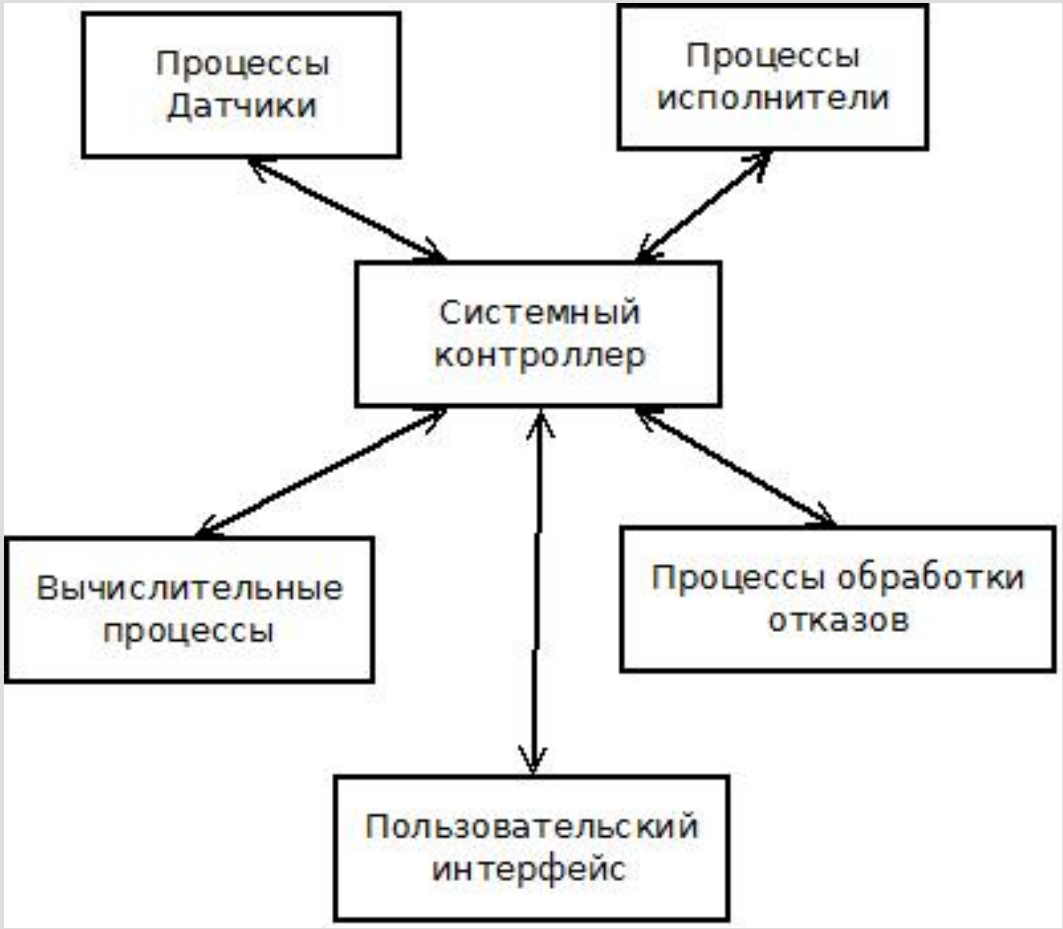
# Модели управления взаимодействия подсистем

- Модель централизованного управления
  - Модель «вызов-возврат»
  - Модель менеджера
- Модель событийного управления
  - Широковещательная модель
  - Модель управления прерываниями

# Модель «вызов-возврат»

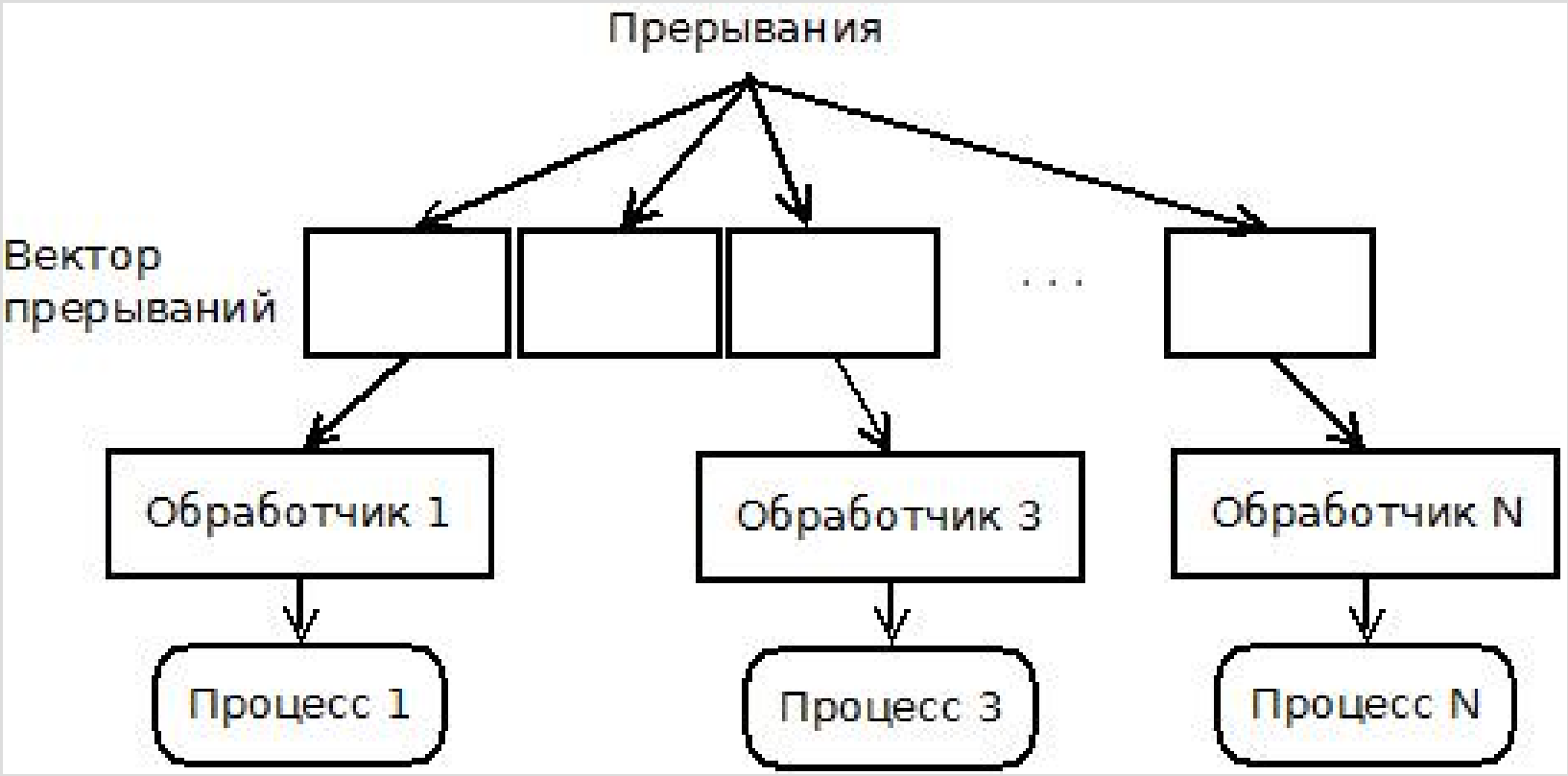


# Модель менеджера



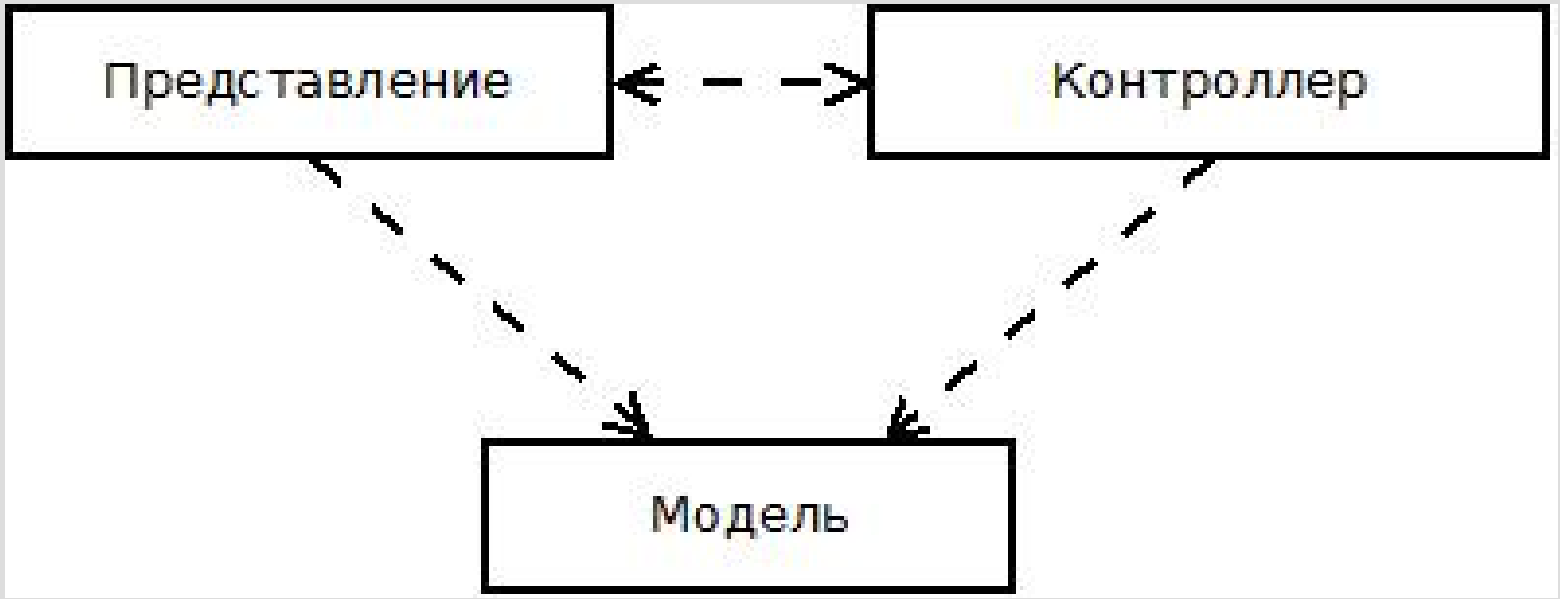
# Широковещательная модель

# Модель управления прерываниями



# Модульная декомпозиция ПО

Типовое решение для представления данных в Web:
MVC — Model / View / Controller



# Паттерны проектирования

## Примеры паттернов

Абстрактная фабрика (Abstract Factory)
Одиночка (Singleton)
Адаптер (Adapter)
Компоновщик ( Composite)
Фасад (Facade)
Посетитель (Visitor)
Наблюдатель (Observer)

# Паттерн Фасад (Facade)

- структурирует объекты,
- предоставляет унифицированный (простой) интерфейс вместо набора интерфейсов подсистемы.