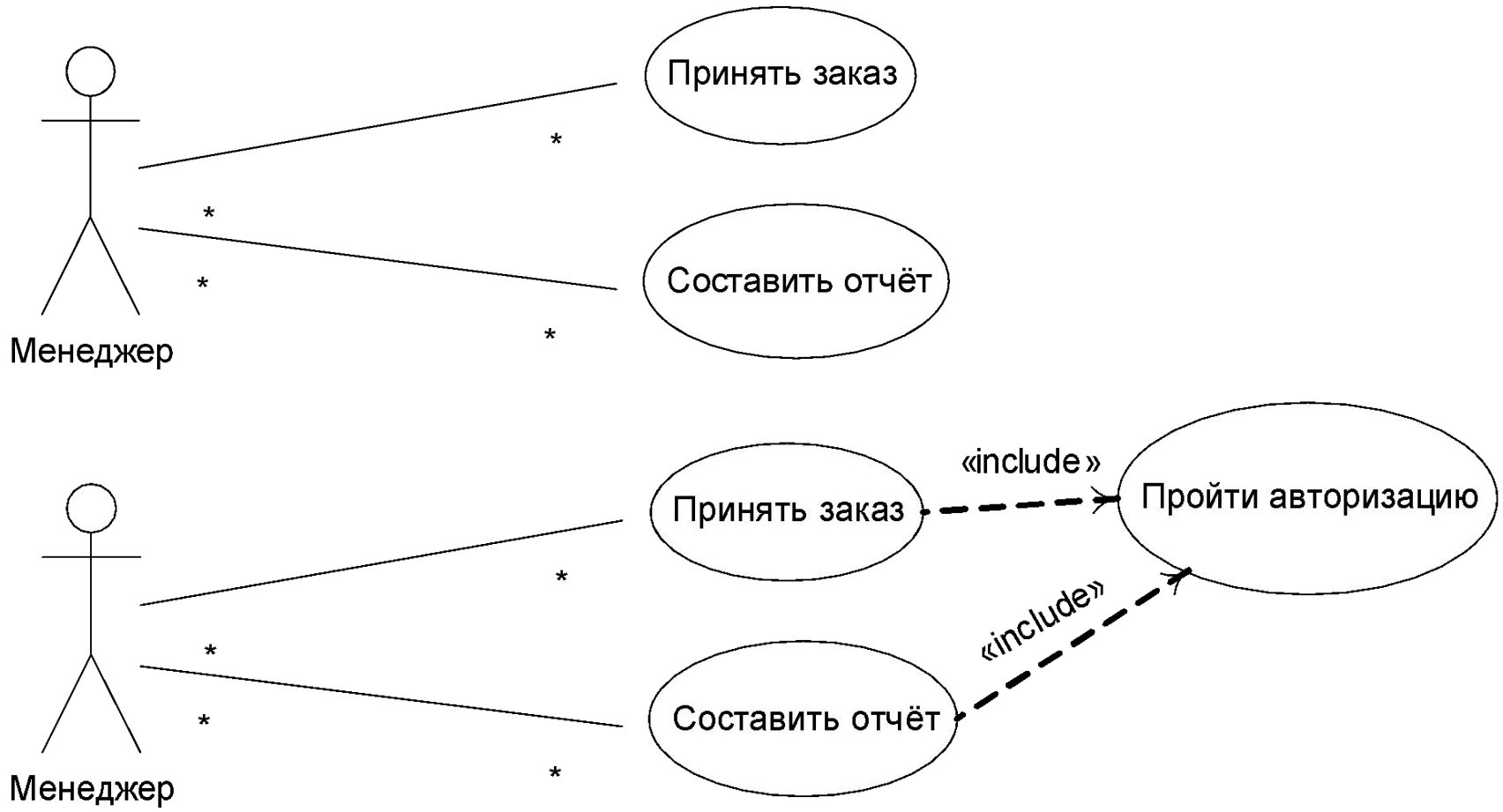
Анализ требований к информационным системам

Лекции 9 - 16

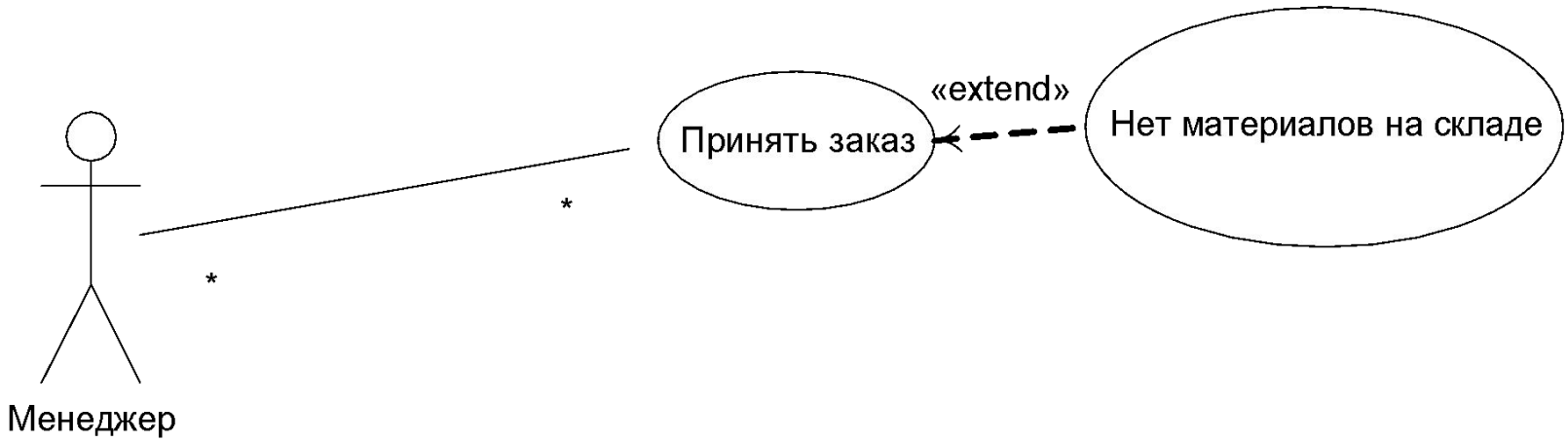
Моделирование требований

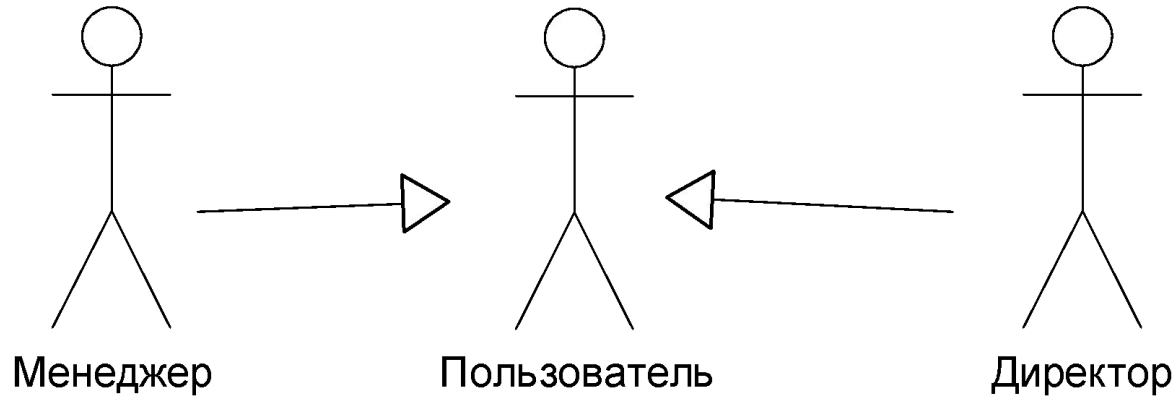
Лекция № 9

Use Case Diagram. Include



Use Case Diagram.
Extend & Generalization





Activities Diagram

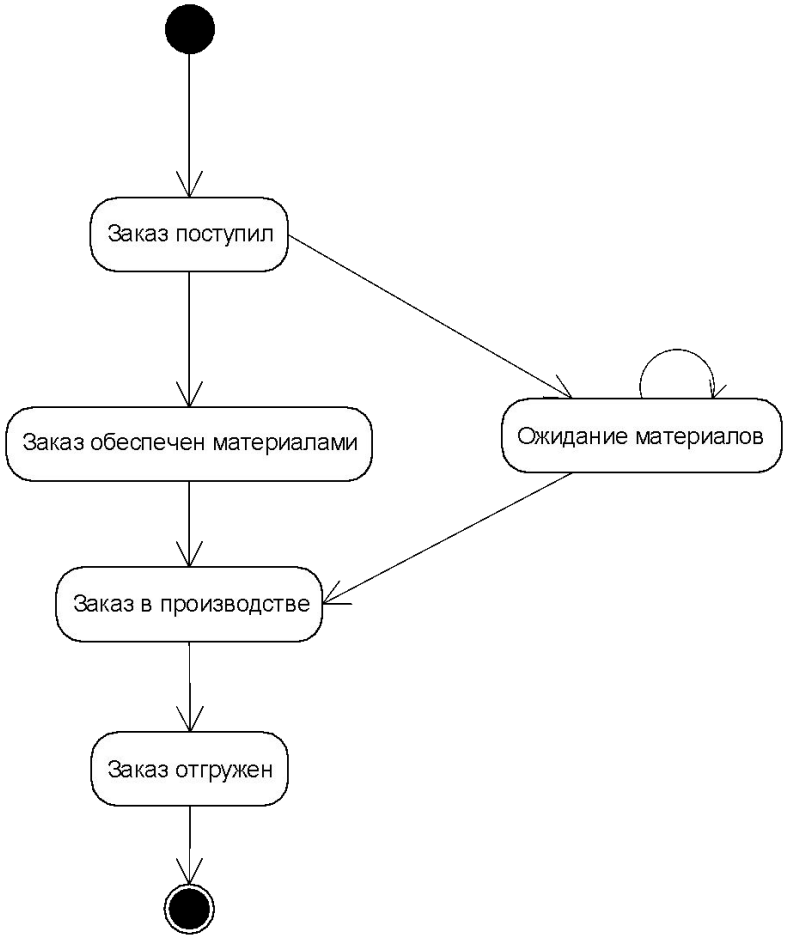
Основные компоненты описания системы:

- Функции (действия)
- Символы «старт» и «стоп»
- Потоки управления
- Разветвители
- Линейки синхронизации

Activities Diagram

- Диаграмма действий позволяет проиллюстрировать вариант использования с требуемой степенью подробности.
- Линейный вариант использования приводит к диаграмме действий с линейным потоком управления между действиями.
- Действия варианта использования с альтернативными сценариями реализуется через разветвители.
- Линейки синхронизации позволяют описывать такие сложные конструкции, как синхронизацию начала (окончания) параллельных во времени процессов.
- Помимо стандартного формата описания, UML предлагает вариант с «плавательными дорожками». Это удобно, когда в варианте использования участвуют несколько акторов.

State chart Diagram



Основные компоненты описания системы:

- Простые состояния,
- Составные состояния,
- Символы «старт» и «стоп»,
- Переходы,
- Линейки синхронизации.

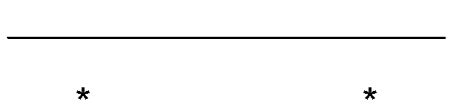
State chart Diagram

- Диаграмма состояний позволяет описать систему в более, чем одном варианте использования.
- Состояние может быть задано в виде набора конкретных значений атрибутов класса или объекта, при этом изменение их отдельных значений будет отражать изменение состояния моделируемого класса или объекта.
- Переход системы из состояния в состояние осуществляется при наступлении *событий*. При этом говорится, что переход *срабатывает.* Переход может быть безальтернативным, либо содержать альтернативы. Во втором случае переход обусловлен наступлением *сторожевых условий*. Наконец, событие может сопровождаться выражением действия, которое происходит в случае, если срабатывает переход.
- Можно объединить часть состояний в одно мета-состояние. При этом возможны переходы между подчинёнными состояниями, переходы между подчинённым и внешним состояниями и переходы между составным и внешним состоянием

Class Diagram

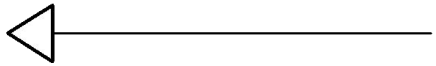
- Для создания диаграммы классов необходимо:
- Осуществить поиск классов (ключевых компонент проблемной области)
- Для каждого найденного класса определить его имя, основные атрибуты, операции и (или) ответственности
- Исследовать отношения найденных классов.
- Уровни абстракции классов:
- концептуальный уровень,
- уровень спецификации,
- уровень реализации.

Class Diagram

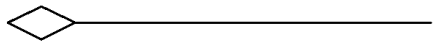


- ассоциация (именованная связь)

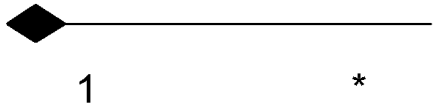
- зависимость (изменения в одном классе приводят к изменениям в другом)



- обобщение / генерализация (родовидовое отношение)

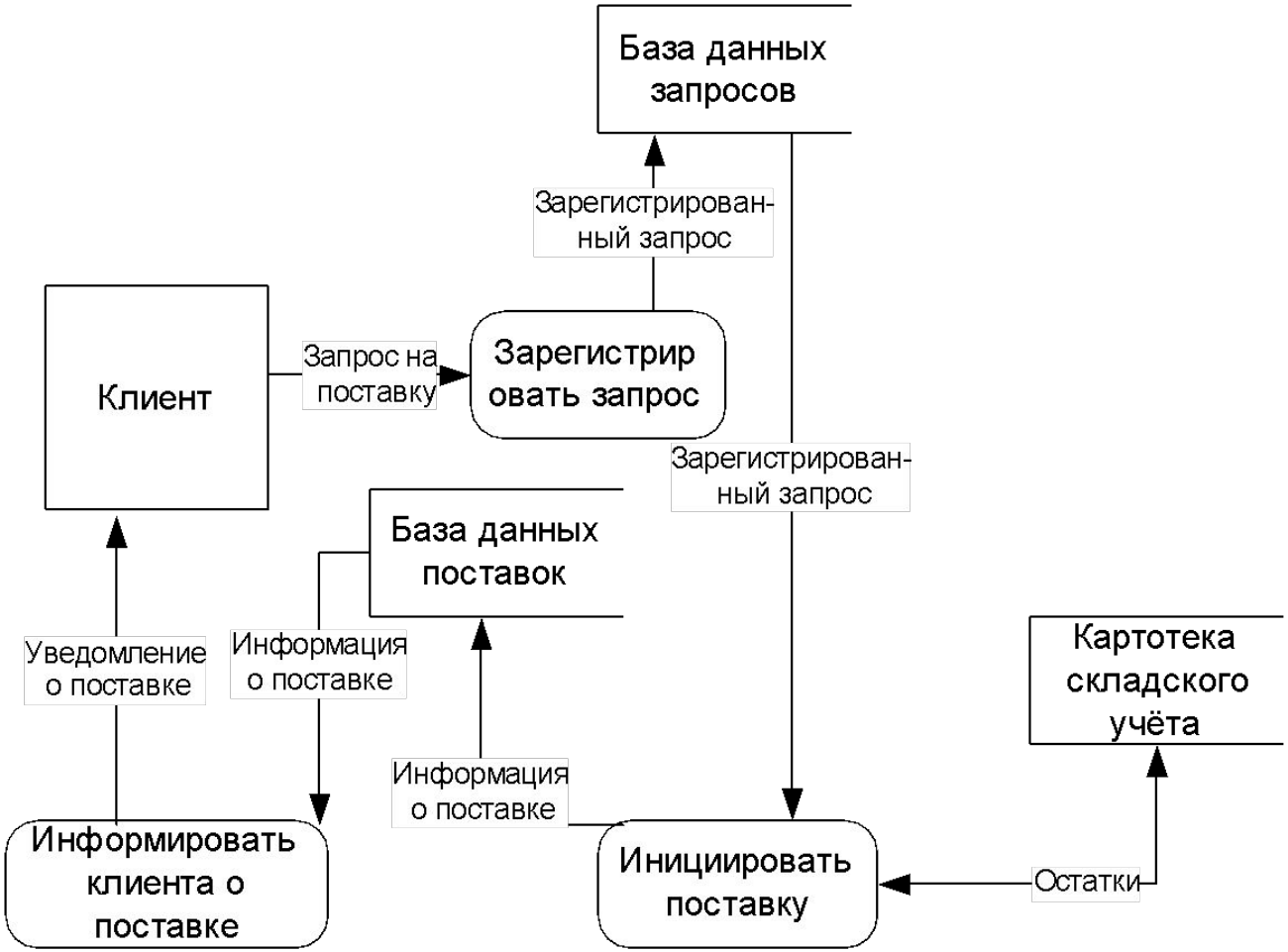


- агрегация (отношение «часть-целое»)



- композиция (отношение «часть-целое» », однозначно регламентирующее количество и состав частей целого

Data Flow Diagram



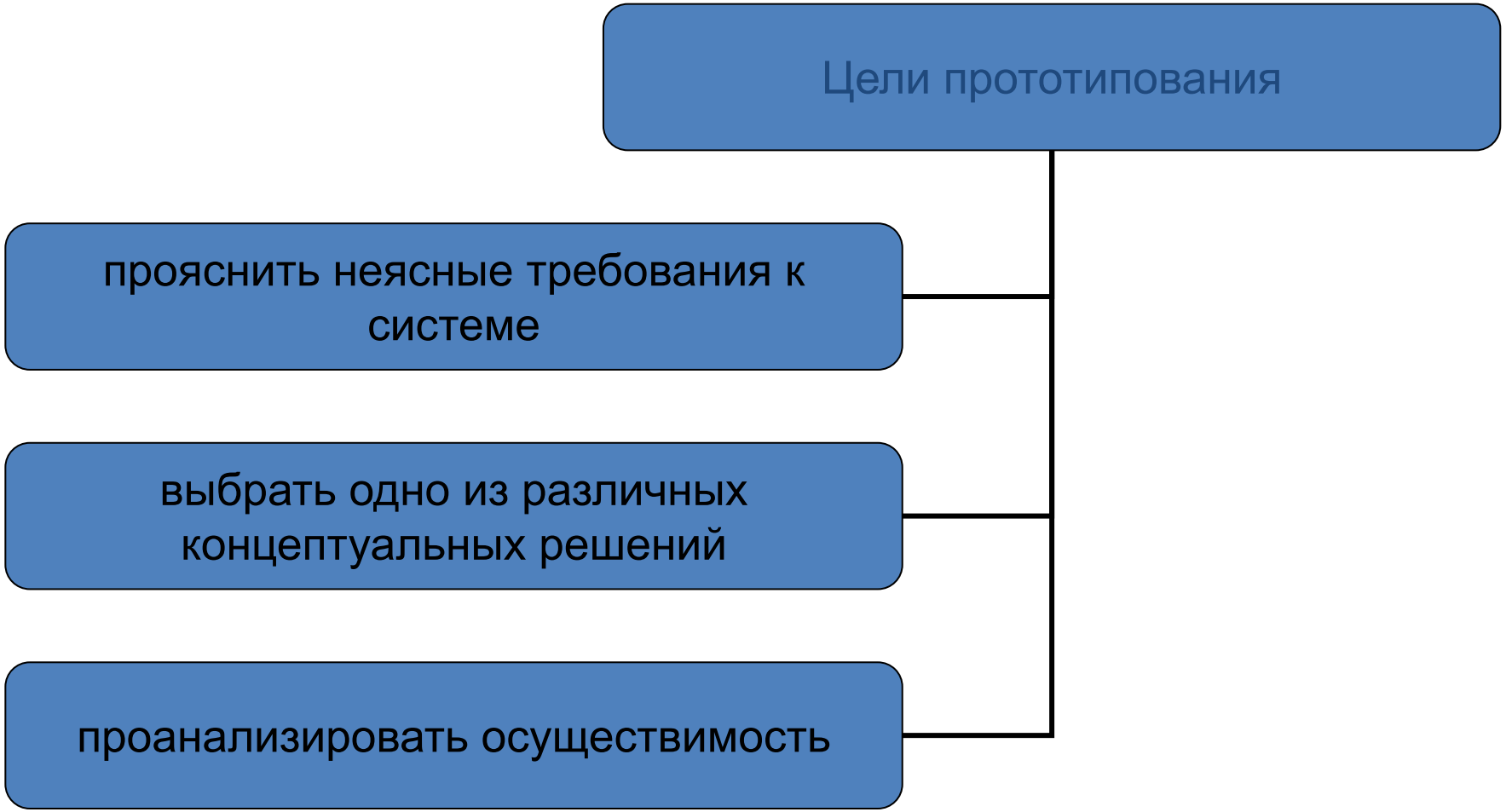
Data Flow Diagram

- Информационная система принимает извне *потоки данных*.
- Для обозначения элементов среды функционирования системы используется понятие *внешней сущности*.
- Внутри системы существуют *процессы* преобразования информации, порождающие новые потоки данных.
- Потоки данных могут поступать на вход к другим процессам, помещаться (и извлекаться) в *накопители данных*, передаваться к внешним сущностям.

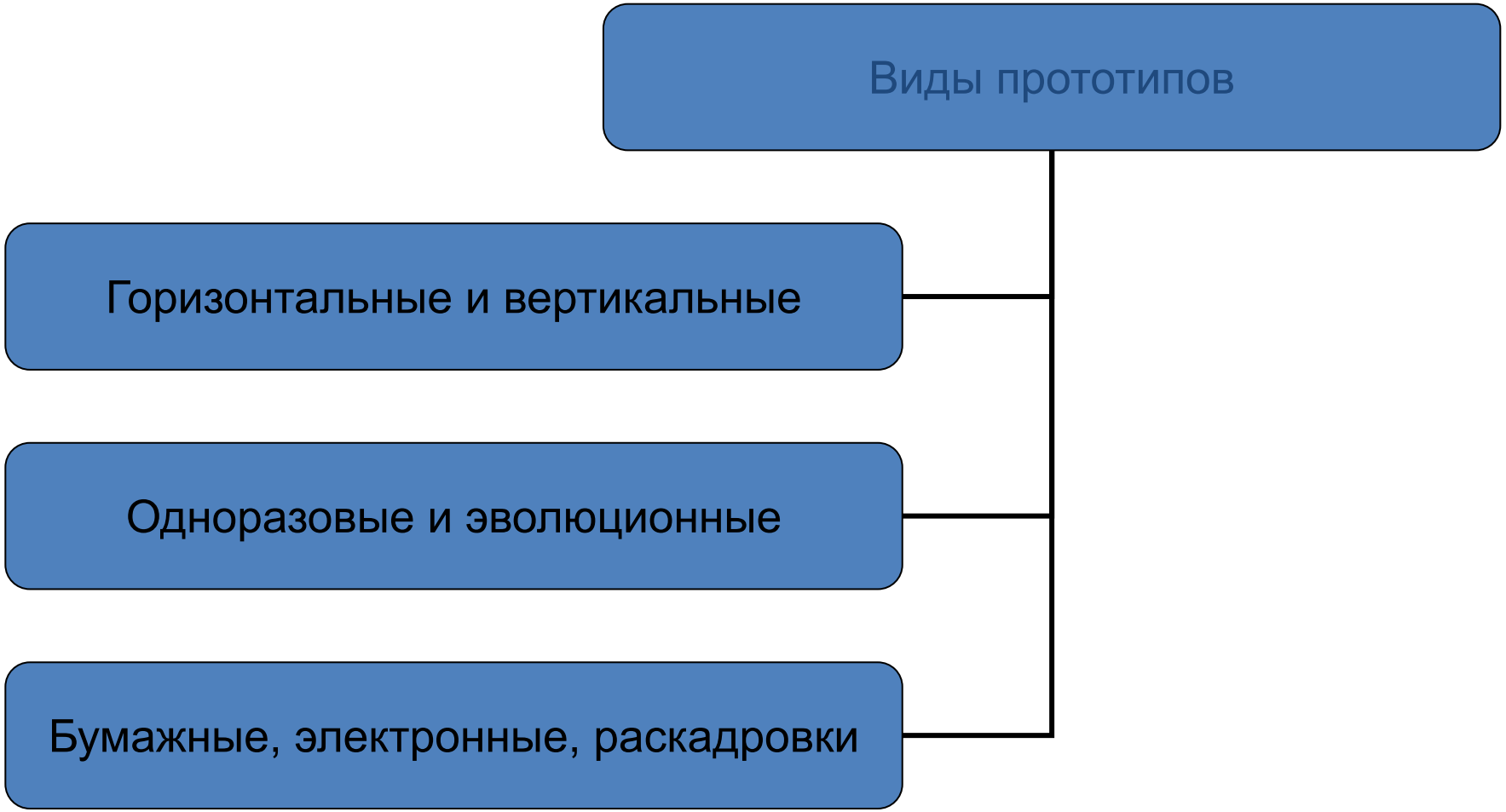
Расширенный анализ требований. Иллюстрированные сценарии и прототипы

Лекция № 10

Цели, требующие применения прототипов



Классификация прототипов



Горизонтальный прототип

- Горизонтальный или поведенческий прототип (horizontal prototype, behavioral prototype) моделирует интерфейс пользователя приложения, не затрагивая логику обработки и базу данных.
- Желательно реализовать ту часть кода, которая отвечает за перемещение между экранами в процессе исполнения вариантов использования, чтобы пользователь смог понять, как будет действовать система в ответ на его действия. Вся остальная функциональность имитируется.
- Горизонтальные прототипы следует использовать для достижения цели прояснения неясных, либо многоальтернативных требований.

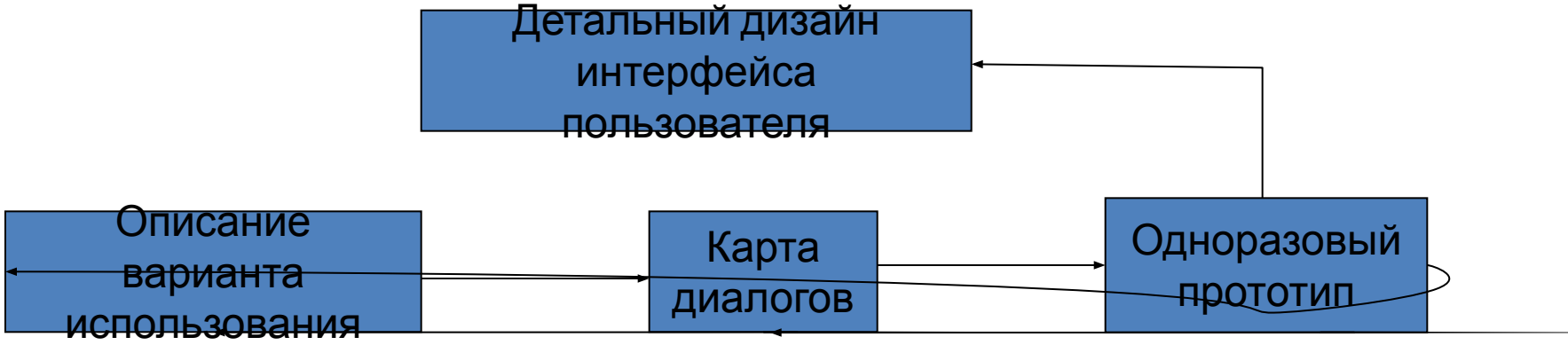
Вертикальный прототип

- Вертикальный или структурный прототип (vertical prototype, structural prototype) не ограничивается интерфейсом пользователя. Он реализует вертикальный «срез» системы, затрагивая все уровни её реализации. При создании такого рода прототипов рекомендуется использовать те языки и среды реализации, что и при изготовлении целевой системы (что, вообще говоря, совсем не обязательно для горизонтальных прототипов).
- Основные цели применения такого рода прототипов – анализ применимости, проверка архитектурных концепций.

Одноразовый прототип

- Одноразовый или исследовательский прототип (throwaway prototype, exploratory prototype) создаётся, когда нужно быстро промакетировать те или иные аспекты и компоненты системы.
- Одноразовый прототип должен создаваться быстро. При его разработке не следует уделять внимание вопросам повторного использования кода, качества, быстродействия, технологичности и т.п.
- В результате получается «сырой» код, который может содержать значительное количество дефектов. Необходимо принять меры к тому, чтобы фрагменты кода, реализующие такого рода прототипы, не стали частью целевой системы.

Переход от одноразового прототипа к детально проработанному UI



Эволюционный прототип

- Эволюционный прототип (evolutionary prototype) создаётся, как первое приближение системы, призванное стать впоследствии самой системой.
- Код эволюционного прототипа должен последовательно, в течении одной или более итераций, перерасти в код целевого приложения. Поэтому данный вид прототипов требует всего того, от чего следует отказаться при создании одноразовых прототипов: скрупулёзной разработки, применения технологических методов и приёмов, тестирования результатов и т.п.

Соотношение прототипов

	Одноразовые	*Эволюционные*
Горизонтальные	▪Прояснение и уточнение примеров использования и функциональных требований ▪Выявление пропущенных требований ▪Исследование возможных вариантов интерфейса пользователя	▪Реализация базовых вариантов использования ▪Реализация дополнительных вариантов использования по приоритетам ▪Реализация и доработка web-сайтов ▪Адаптация системы к быстро меняющимся требованиям бизнеса
Вертикальные	▪Демонстрация технической осуществимости	▪Реализация и наращивание ключевой клиент-серверной функциональности и уровней коммуникации ▪Реализация и оптимизация основных алгоритмов ▪Тестирование и настройка производительности

Бумажный прототип

- Бумажный прототип (paper prototype) – отличная альтернатива рассмотренным выше разновидностям электронных прототипов в случае, когда Разработчик ограничен в ресурсах. Его достоинства:
- 1. Заказчик не станет акцентировать внимание на цветовом решении, форме кнопок и т.п., отвлекаясь от анализа функциональности.
- 2. Заказчик никогда не скажет, глядя на бумажный интерфейс: «Да вы, я вижу, уже создали систему на 85%! Давайте закончим её в течении недели».

Раскадровка

- *Пассивная раскадровка.* Презентация, изготовленные при помощи средств электронного офиса (например, комбинации Microsoft Visio и Microsoft PowerPoint). В этом случае пользователь лишён свободы выбора, предоставляемой ему поведенческим прототипом. Но идею пошаговой смены экранов в процессе реализации сценария варианта использования вполне можно реализовать.
- *Активная раскадровка* является дальнейшим развитием понятия пассивной раскадровки, с применением средств анимации и т.п.
- *Интерактивная раскадровка* представляет собой электронный одноразовый горизонтальный прототип.

Иллюстрированные сценарии прецедентов

- Иллюстрированные сценарии прецедентов содержат дополнительные сведения – *аспекты применимости*, помогающие Разработчику лучше понять специфику проблемной области.
- Аспект применимости – информация, позволяющая расширить описание прецедента описаниями, конкретизирующими те или иные его особенности и, в конечном итоге, повысить степень комфортности пользователя.
- Виды аспектов применимости:
- ориентиры,
- средние значения атрибутов и объёмы объектов,
- средняя интенсивность использования.

Ориентиры

- Ориентиры – это описание опциональных функциональных возможностей системы. Отсутствие таких возможностей не приводит к фатальной неудаче. Присутствие – улучшает применимость, снабжая полезной информацией. Ориентиры следует расценивать не как требования, а как пожелания или рекомендации.

Прецедент с ориентиром

- В процессе выполнения прецедента менеджер по приёму заказов выбирает заказчика из клиентской базы, определяет товарные позиции из справочника и указывает их количество. Система отображает на мониторе наименование позиций, цену, сумму и количество на складе. Менеджер назначает скидку и определяет порядок оплаты. Система рассчитывает итоговую сумму [Менеджер должен иметь возможность видеть текущее сальдо расчётов с клиентом и данные по последним десяти сделкам со статистикой по дисциплине соблюдения договорных обязательств].

Средние значения атрибутов и объёмы объектов (СЗА&ОО)

- Данная информация позволяет оптимальнее построить пользовательский интерфейс и оценить на ранних стадиях проекта «узкие места» в обработке данных, которые могут повлиять на производительность системы.
- Так, при выборе из 2 возможностей лучше подойдёт элемент управления checkbox, при выборе, ограниченном 2-3 десятками позиций – выпадающий список, при многообразии, измеряемом тысячами вариантов, потребуются дополнительные средства фильтрации и поиска.

Прецедент со СЗА&ОО

- В процессе выполнения прецедента менеджер по приёму заказов выбирает заказчика из клиентской базы {до 10000 клиентов}, определяет товарные позиции из справочника {товары разбиты на 10 категорий, количество позиций в категории не превышает 500} и указывает их количество {до 100 позиций, средняя закупка – 8 позиций}. Система отображает на мониторе наименование позиций, цену, сумму и количество на складе. Менеджер назначает скидку и определяет порядок оплаты {на данный момент существуют 3 варианта порядка оплаты}. Система рассчитывает итоговую сумму.

Средняя интенсивность использования (СИИ)

- Средняя интенсивность использования позволяет выделить сценарии «массового» использования, в которых всё должно быть идеально (быстродействие, удобство пользования, минимум действий на выполнение операций). Например – интерфейс кассира в супермаркете.
- Другая крайность – сценарии, выполняемые от случая к случаю, не каждый день и не требующие особой оперативности (например, расчёт заработной платы за месяц). Эти данные позволяют, структурировать подачу информации, убрать из «главных» интерфейсов редко используемые опции и т.п.

Прецедент со СИИ

- Фрагмент описания потока событий ИСП для прецедента «Оформить заказ для нового клиента», расширенного объёмами и средними значениями объектов.
- В процессе выполнения прецедента менеджер по приёму заказов выбирает заказчика из клиентской базы (в 95% случаев), либо вызывается интерфейс регистрации нового клиента (в 5% случаев).

Документирование требований

Лекция № 11

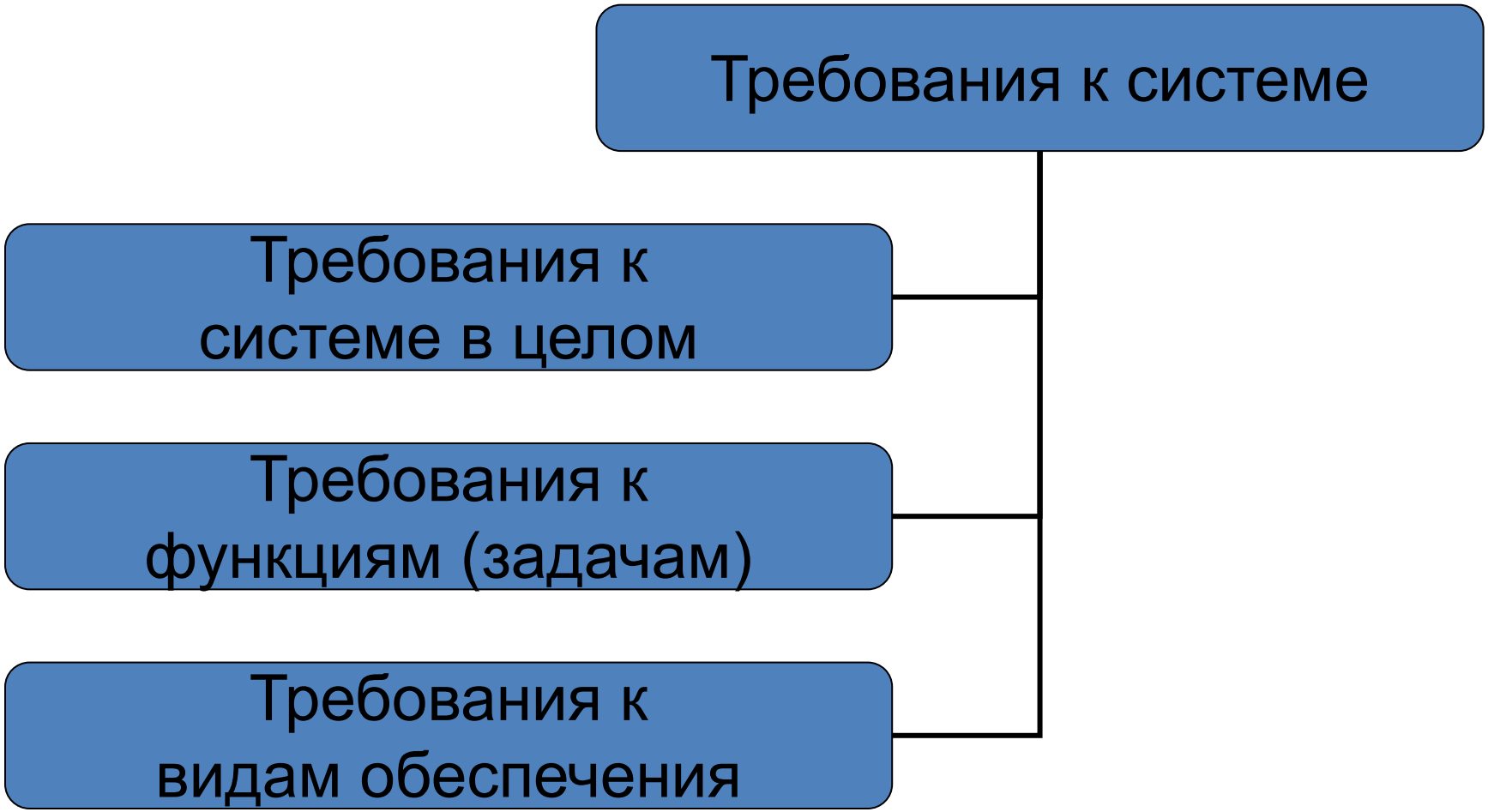
ГОСТ РФ

- ГОСТ 19.201-78 «Техническое задание, требования к содержанию и оформлению»

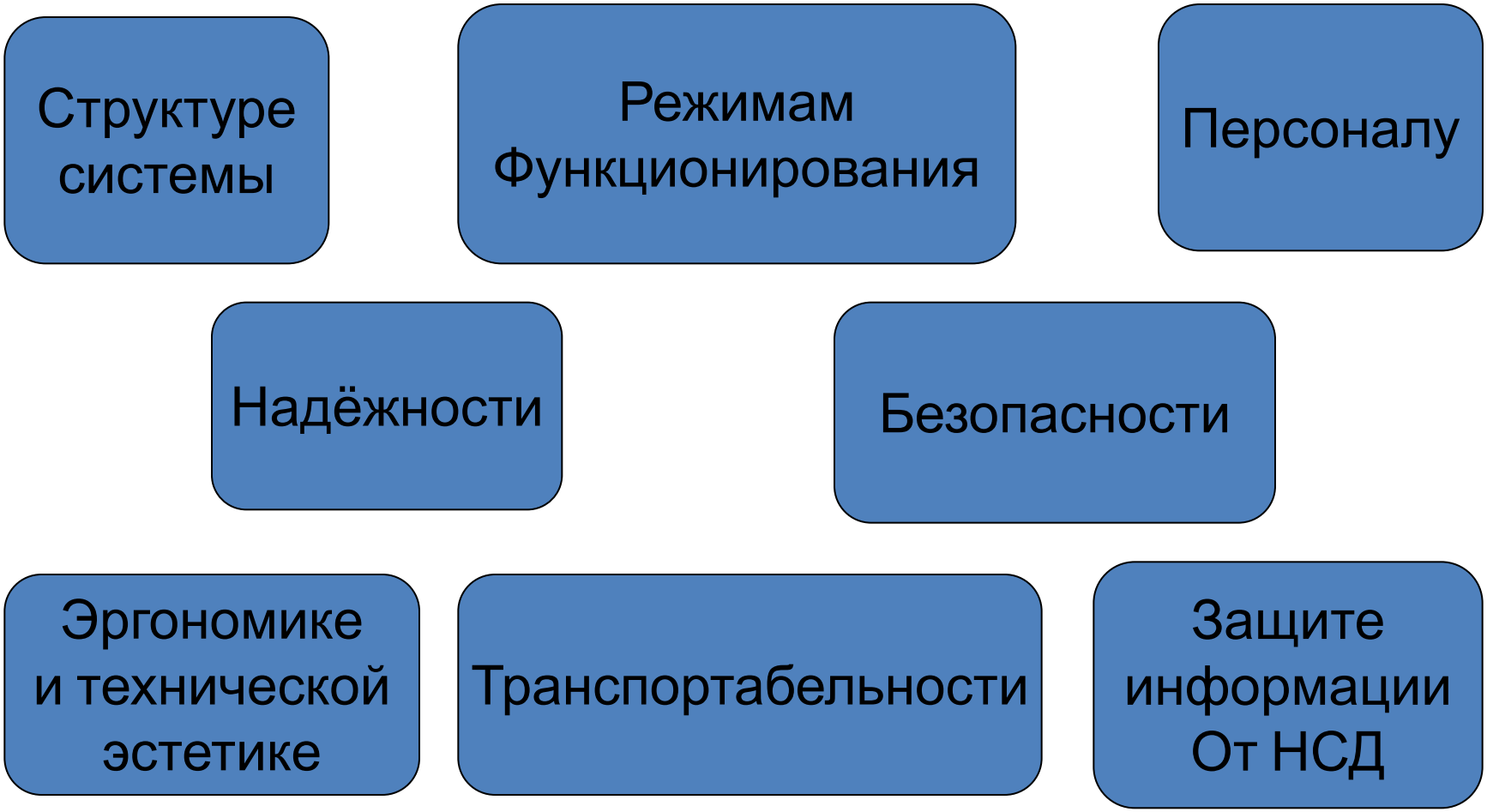
- ГОСТ 34.602-89 «Техническое задание на создание автоматизированной системы» (ТЗ на АС)

Структура ТЗ, ГОСТ 34.602-89

Требования к системе, ГОСТ 34.602-89



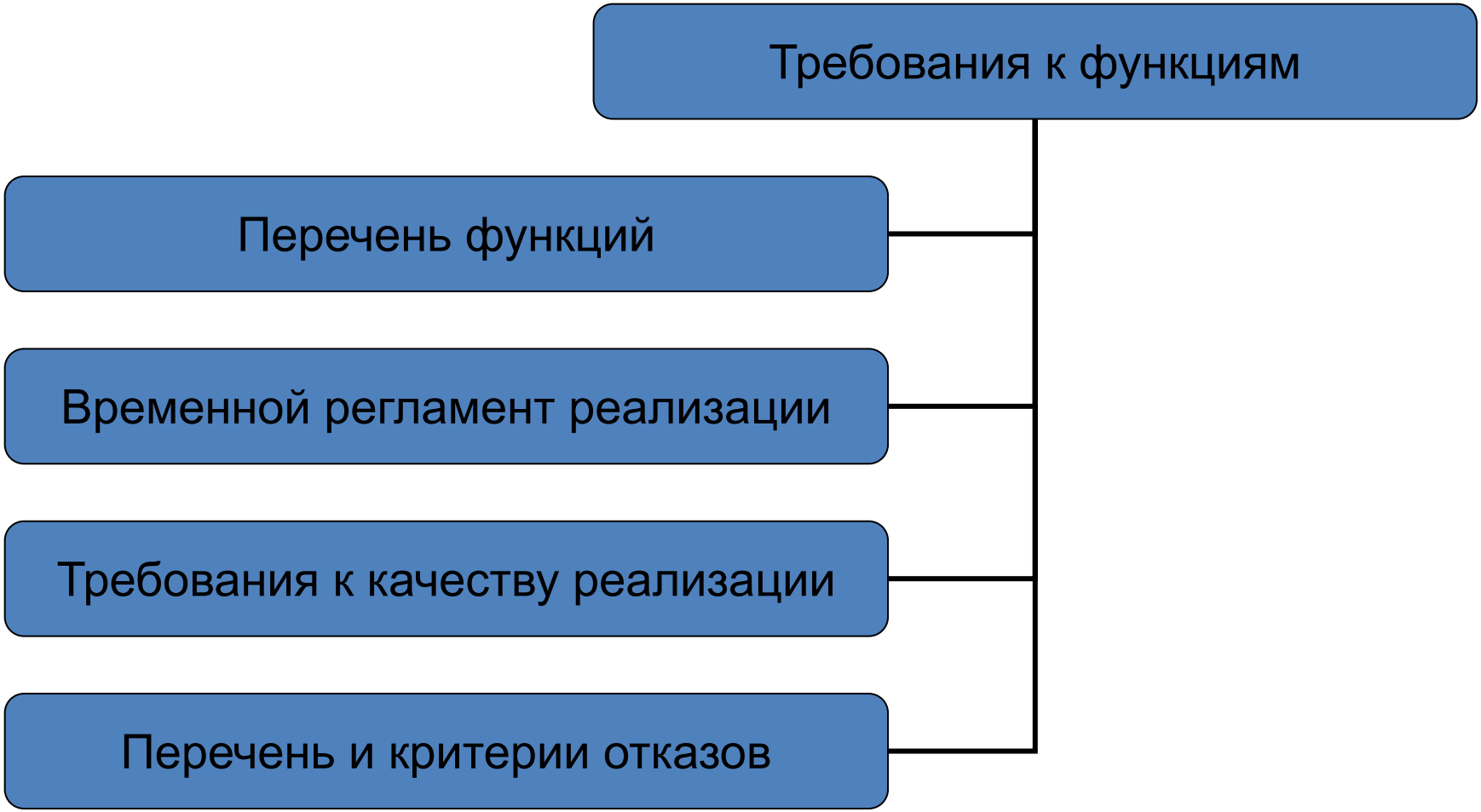
Требования к системе в целом, ГОСТ 34.602-89 (1)



Требования к системе в целом, ГОСТ 34.602-89 (2)

- Защите от внешних воздействий
- Эксплуатации, техническому обеспечению, ремонту и хранению
- Сохранности информации при авариях
- Патентной чистоте
- Стандарти-зации и унификаци и
- Показател и назначени я
- Дополни-тельные требования

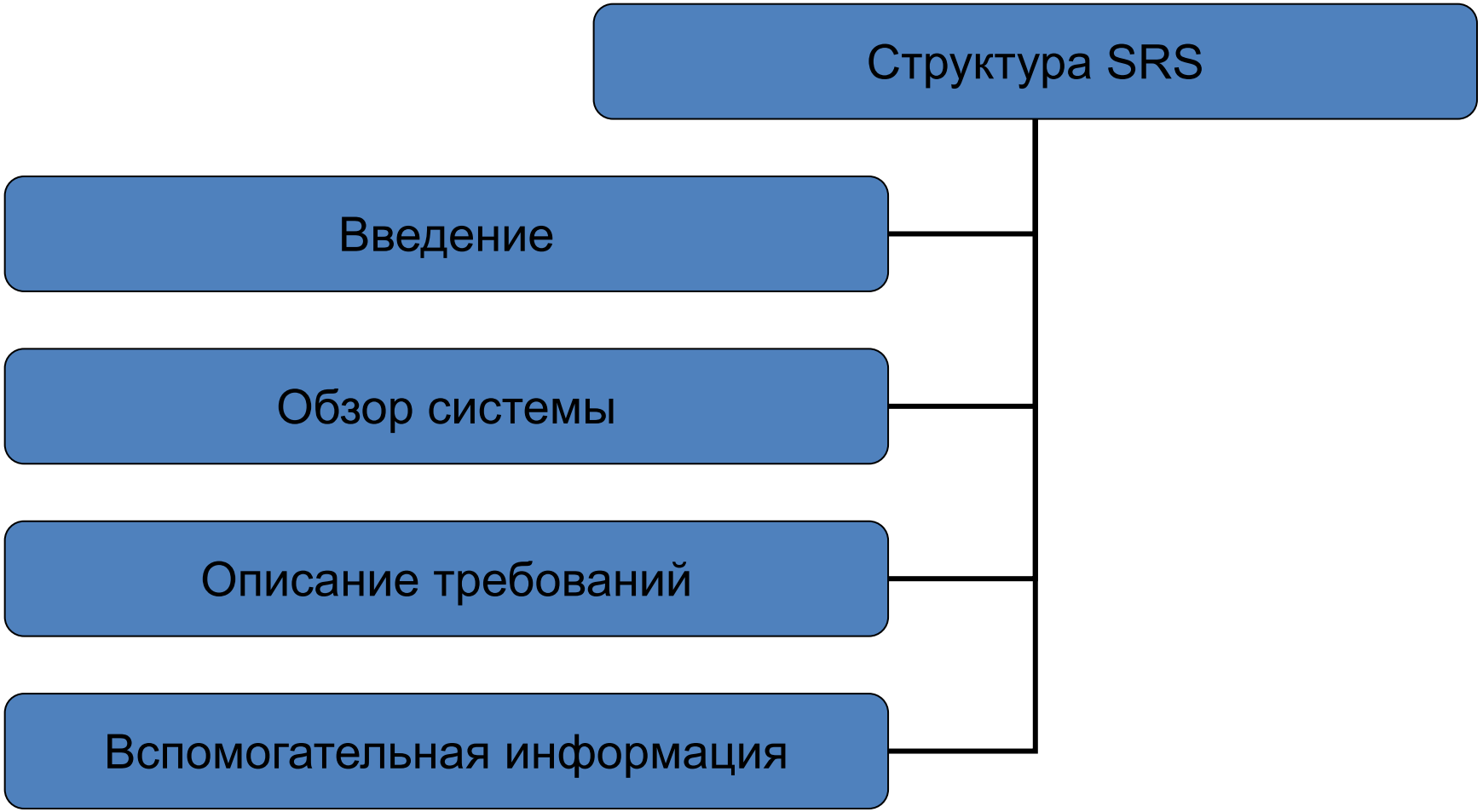
Требования к функциям, ГОСТ 34.602-89



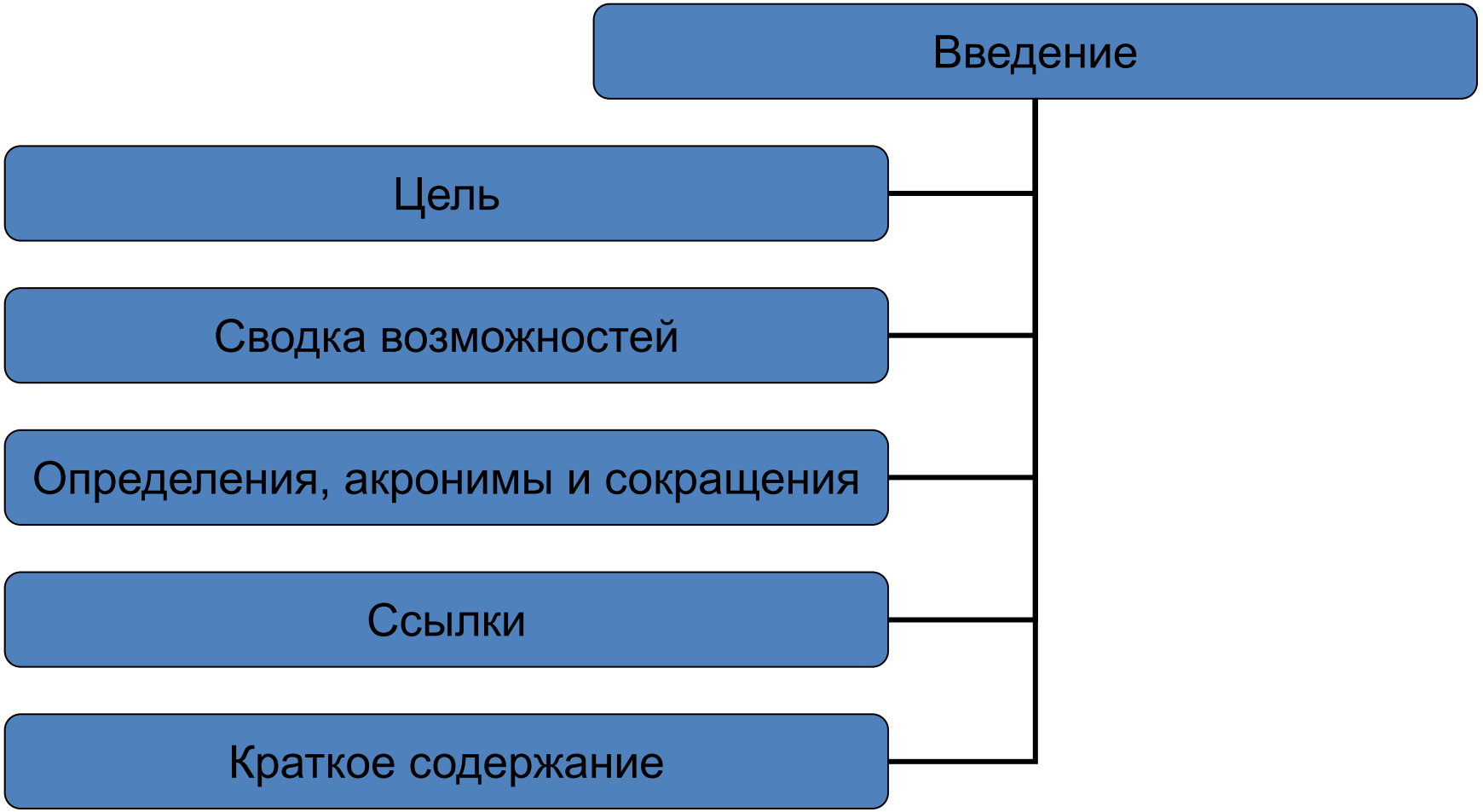
Требования к видам обеспечения, ГОСТ 34.602-89

- Математическое
- Лингвистическое
- Информационное
- Программное
- Методическое
- Организационное
- Техническое
- Метрологическое

Структура SRS, RUP



Введение; SRS, RUP



Обзор системы; SRS, RUP

- *Обзор прецедентов*. Содержит список имён и кратких описаний вариантов использования и акторов с иллюстрациями в виде диаграмм прецедентов.
- *Предположения и зависимости*. Данная секция описывает ключевые технические возможности, компоненты, подсистемы, связанные проекты, которые могут влиять на жизнеспособность разрабатываемой системы.

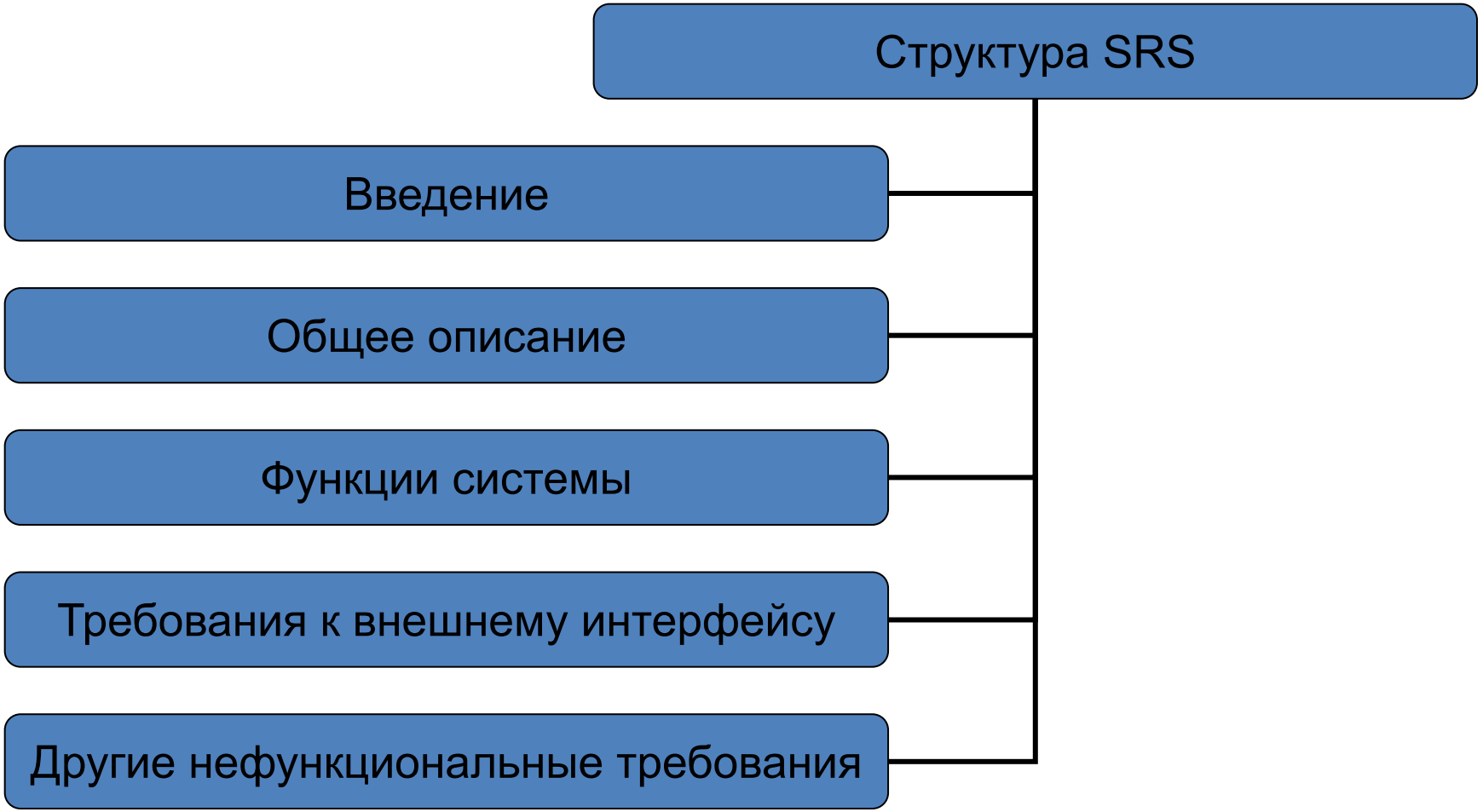
Предположения и зависимости

- Предположением (assumption) называется положение, которое считается истинным при отсутствии доказательства или определяющей информации.
- При определении зависимостей (dependencies) проекта от внешних факторов, необходимо проанализировать, какие новые операционные системы, регламенты бизнес-процессов, стандарты качества, информационные системы могут появиться на предприятии внедрения и как это может повлиять на функционирование изготовляемой АИС.

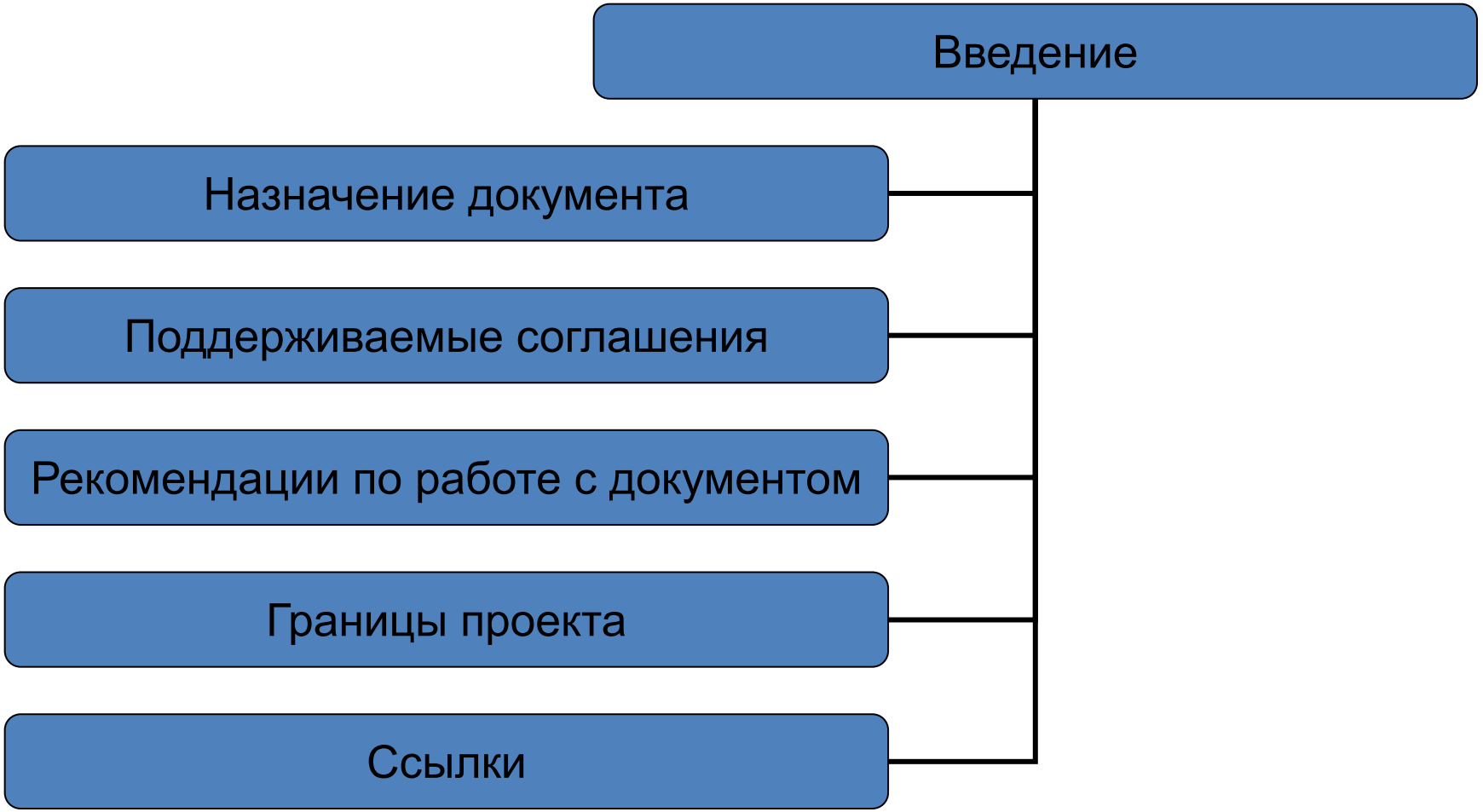
Описание требований; SRS, RUP

- *Описание вариантов использования*. Параграф содержит описание вариантов использования и связанных с ними нефункциональные требований, либо ссылки на соответствующие артефакты.
- *Специальные требования*. Параграф содержит описание функциональных требований (не описанных, как варианты использования), а также описание нефункциональных требований общего характера (не сопоставленных ни одному прецеденту в предыдущем разделе), либо ссылки на соответствующие артефакты.

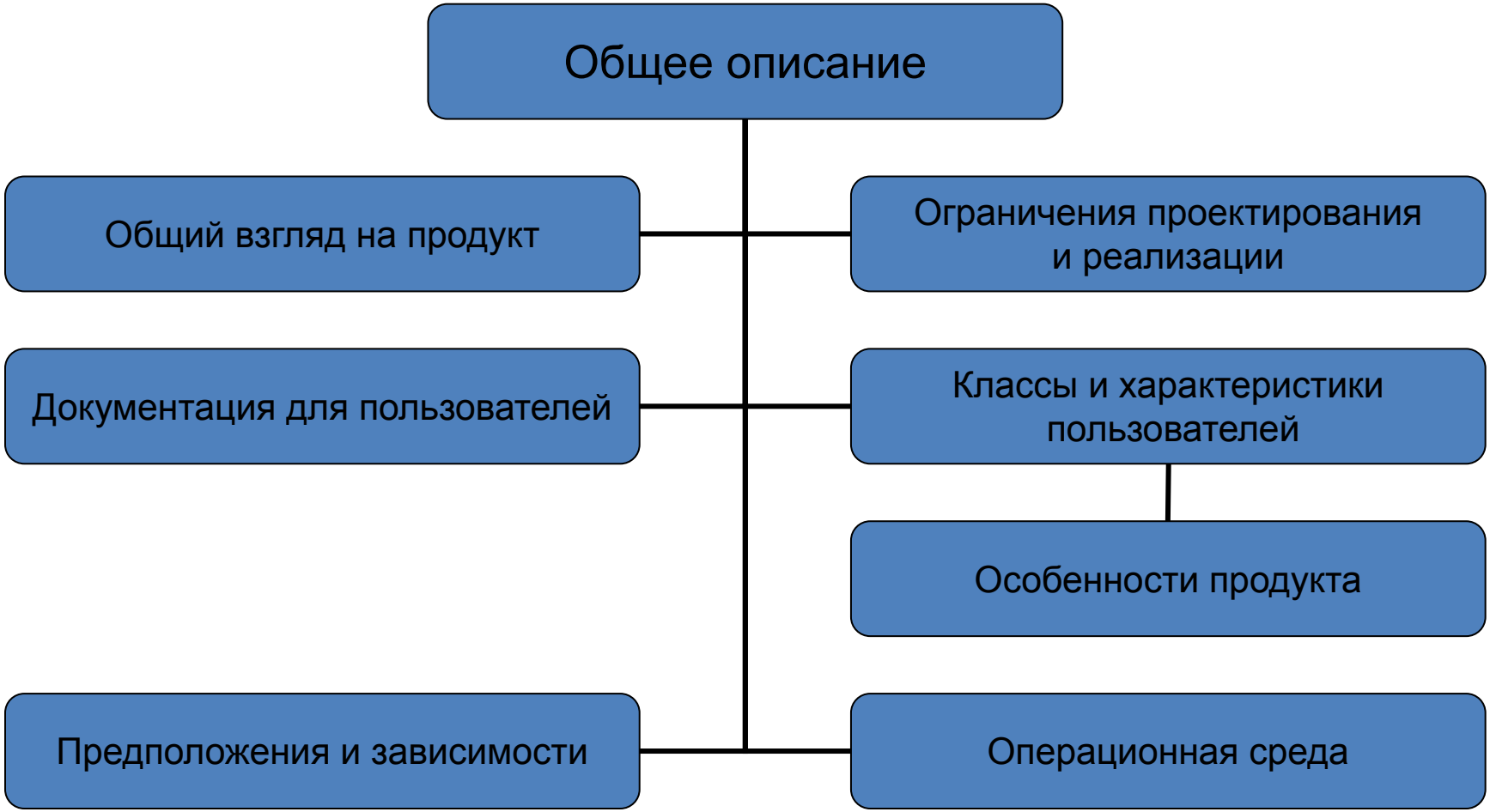
Структура SRS, на основе IEEE Standard 830-1998



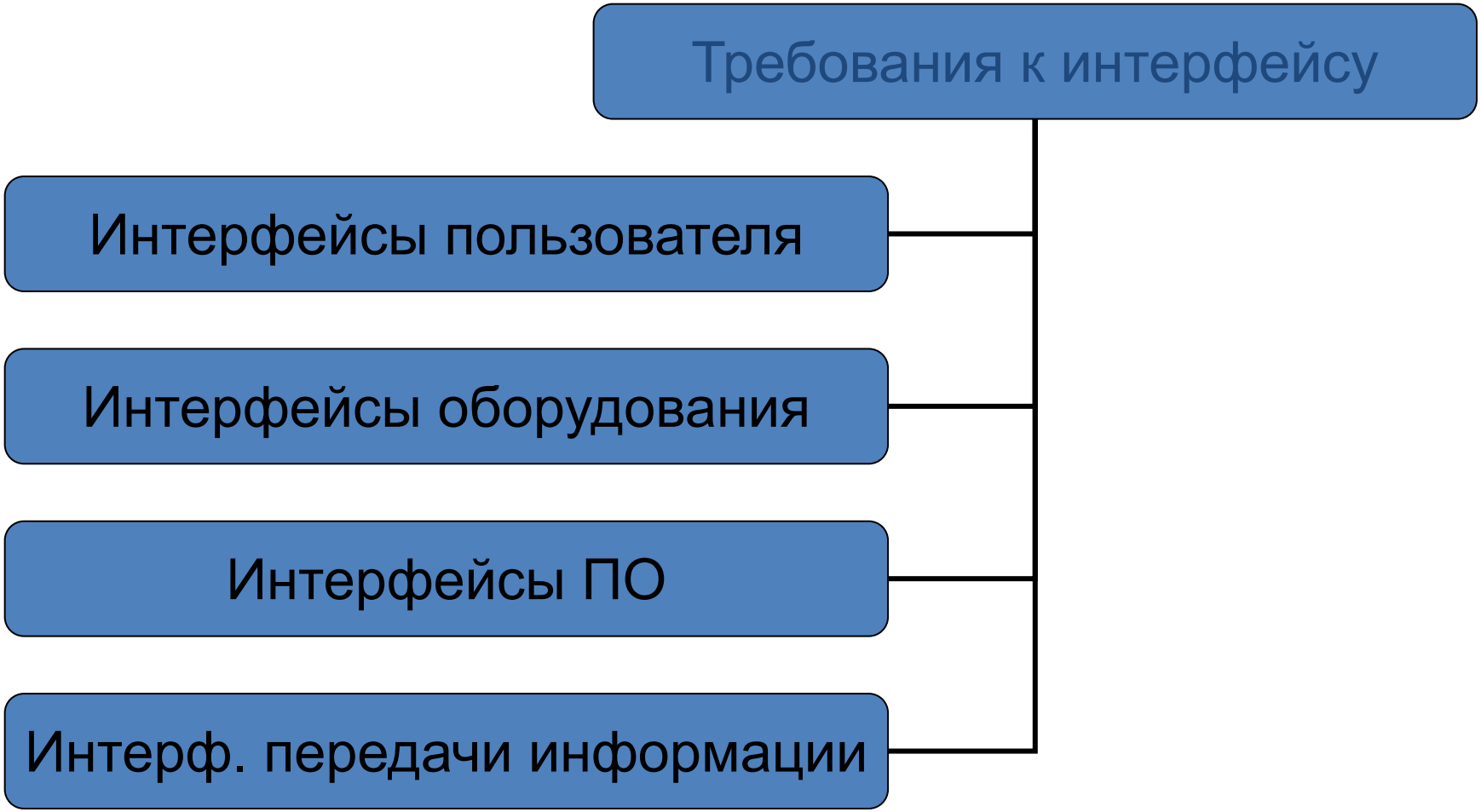
Введение; SRS; IEEE 830



Общее описание; SRS; IEEE 830



Требования к внешнему интерфейсу; SRS; IEEE 830



Назначение функциональной спецификации MSF

- инструкция команде разработчиков о том, что они должны будут создать
- основа для оценивания объема работы
- четкое соглашение с Заказчиком о том, что должно быть сделано
- основа для синхронизации работы всей проектной команды

Документирование требований в MSF

- В начале фазы проектирования проектная группа работает с проектными требованиями:
- бизнес-требования
- требования к эксплуатации
- системные требования
- требования пользователя
- Один из основных результатов фазы:
- функциональная спецификация