

# В чем проблема?

- Процент успешных проектов
- Запланировано гораздо больше чем удалось сделать
- Превышения бюджета
- Выполнена ненужная работа

26% Успешных  
46% Частично успешных  
28% Неудачны (провалы)

69% Превышение бюджета  
79% Превышение сроков

\$75bn Стоимость провальных проектов  
\$22bn Стоимость превышения бюджета

45% Функций систем никогда не используется !!!

# Причины провалов

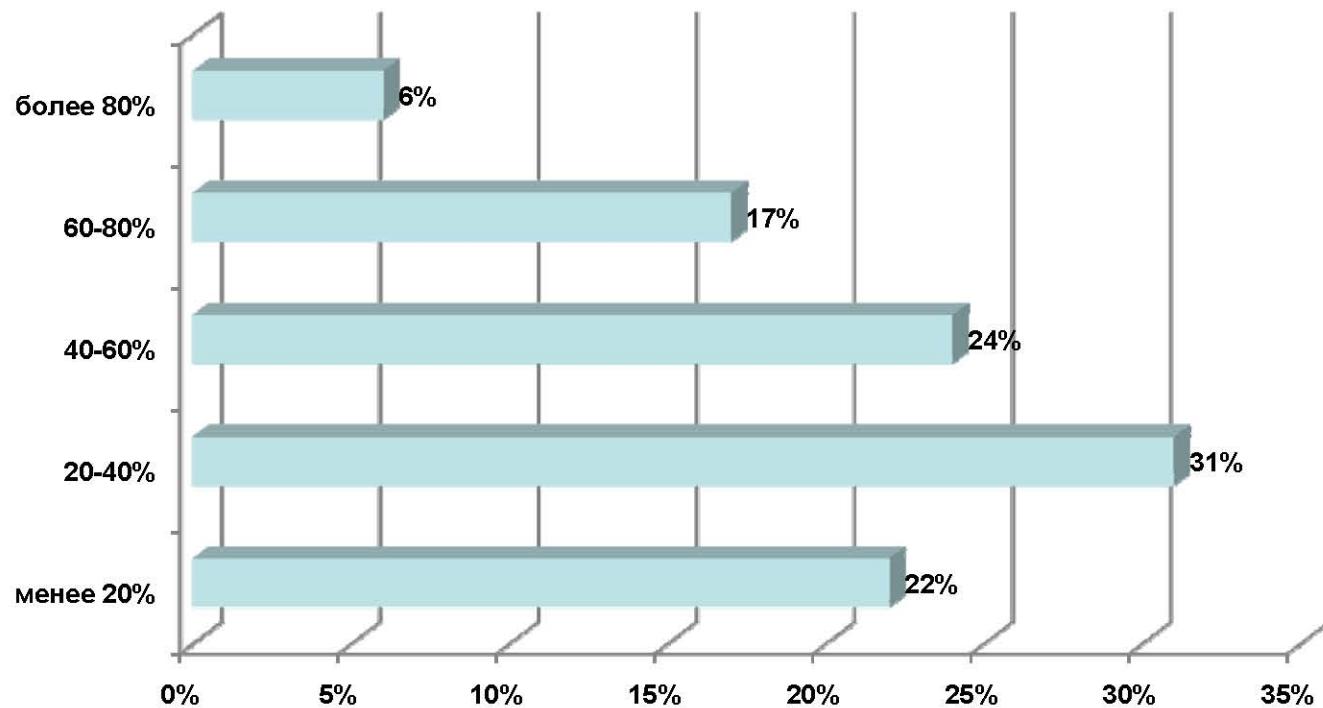
*Часть причин провалов проектов связаны с управлением проектом, часть причин связано с требованиями, но ни одна из причин **не является технической***

- Неполные или неоднозначные требования
- Низкое вовлечение пользователей в проект
- Недостаточно ресурсов
- Нереалистичные ожидания
- Недостаточная поддержка руководства
- Постоянно изменяющиеся, нестабильные требования
- Плохое планирование
- Проект перестает быть нужным
- Размер и сложность проекта

Источник: The Standish Group 1999

# Свежая статистика

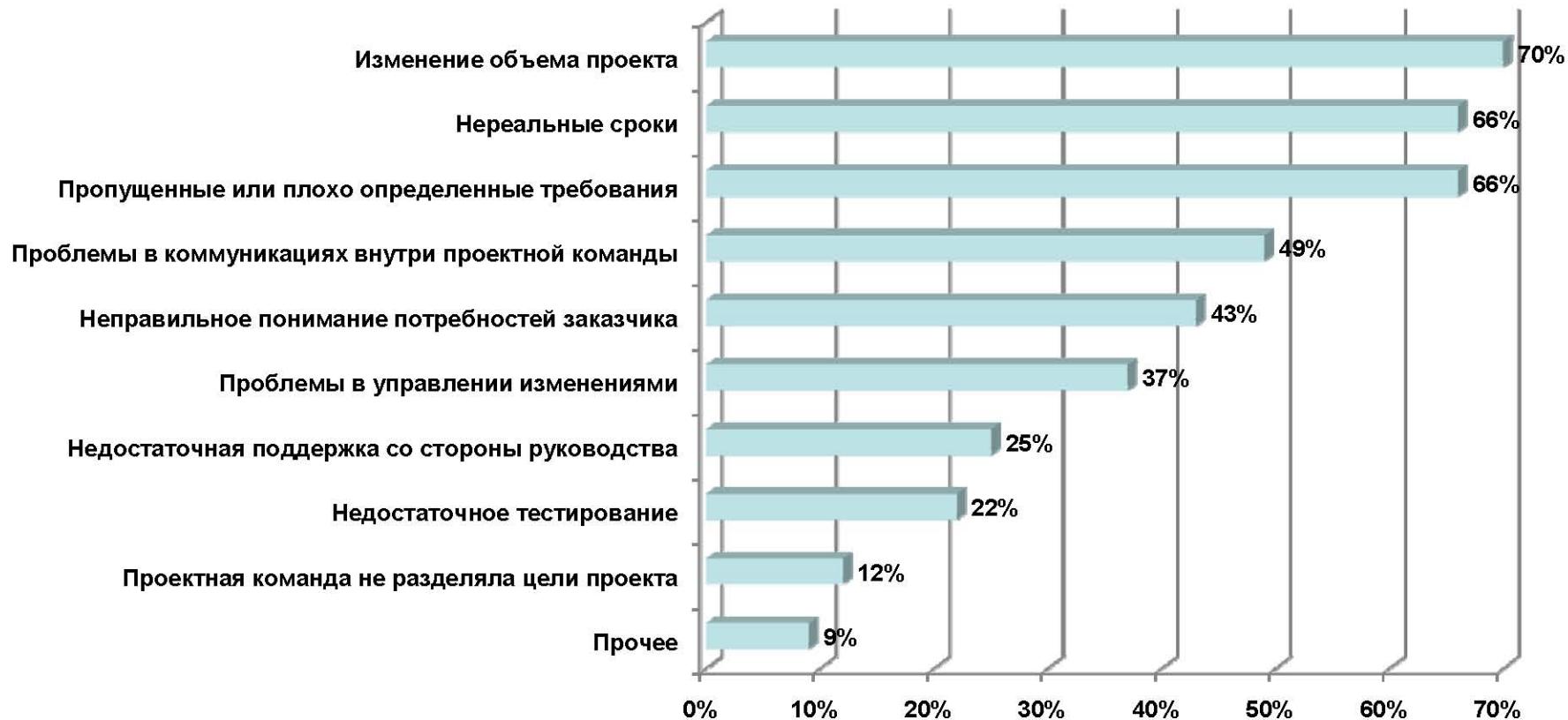
- Как часто проект или продукт выпускается вовремя в рамках бюджета, и тем набором возможностей, который был изначально запланирован?



Источник: The 2008 State of Requirements Management Report, © 2008 Jama Software 8

# Свежая статистика

- Что является причиной неудачных проектов?



Источник: The 2008 State of Requirements Management Report, © 2008 Jama Software 9

# Что такое требования и зачем нужны требования?

- Требования описывают потребности «заинтересованных сторон»
- Требования определяют то, что необходимо сделать, для того чтобы удовлетворить потребности «заинтересованных сторон»
- Требования нужны чтобы удовлетворить потребности «заинтересованных сторон»
- Требования нужны чтобы получить «правильный» продукт для выхода с ним на рынок в подходящее для этого время

# Зaintересованные стороны это:

- Государства, организации, отдельные люди или группы людей, а также системы, которые могут прямо или косвенно быть заинтересованы в результатах проекта, или если результаты проекта или ход его выполнения прямо или косвенно влияет на них (как положительно так и отрицательно).

# Область проблем и область решений



# Область проблем и область решений

Уровень требований	Область	Точка зрения	Цель
Требования заинтересованных сторон	Область проблем	Заказчик, Представитель заинтересованной стороны	Определяет что различные заинтересованные стороны желают достичь. Следует избегать конкретных решений.
Системные требования	Область решения	Пользователь системы, Аналитик	Абстрактно определяет как система будет удовлетворять пользовательским требованиям. Следует избегать точных описаний реализаций предлагаемых решений.
Системные спецификации (архитектура системы)	Область решения	Архитектор, Проектировщик	Определяет как конкретная архитектура системы будет удовлетворять системным требованиям.

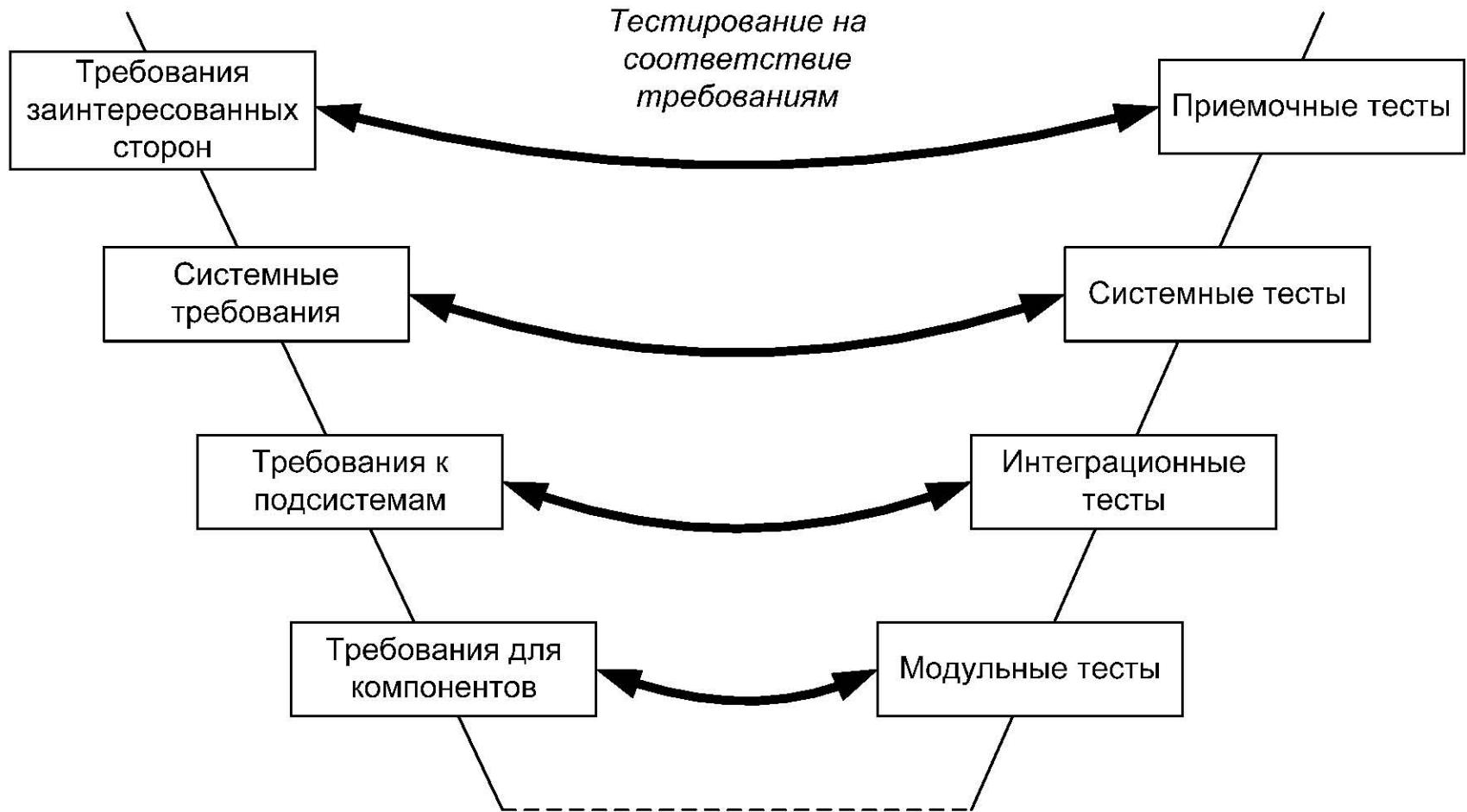
# Области применения требований

- Планирование
- Управление рисками
- Управление объемом проекта
- Проектирование
- Тестирование
- Управление изменениями

# Требования и качество

**Качество** – есть соответствие цели или соответствие требованиям – это обеспечение того, что удовлетворяет потребителя и в тоже время гарантирует, что нужды всех заинтересованных сторон учтены.

# Требования и процесс выполнения проекта



# Роль требований на каждом уровне реализации проекта



# Требования и моделирование

**Моделирование необходимо аналитику:**

- Как средство для обсуждения требований и их реализации с заинтересованными сторонами
- Для анализа разрабатываемой системы, чтобы убедиться в наличии требуемых системных свойств
- Для перехода между различными уровнями требований

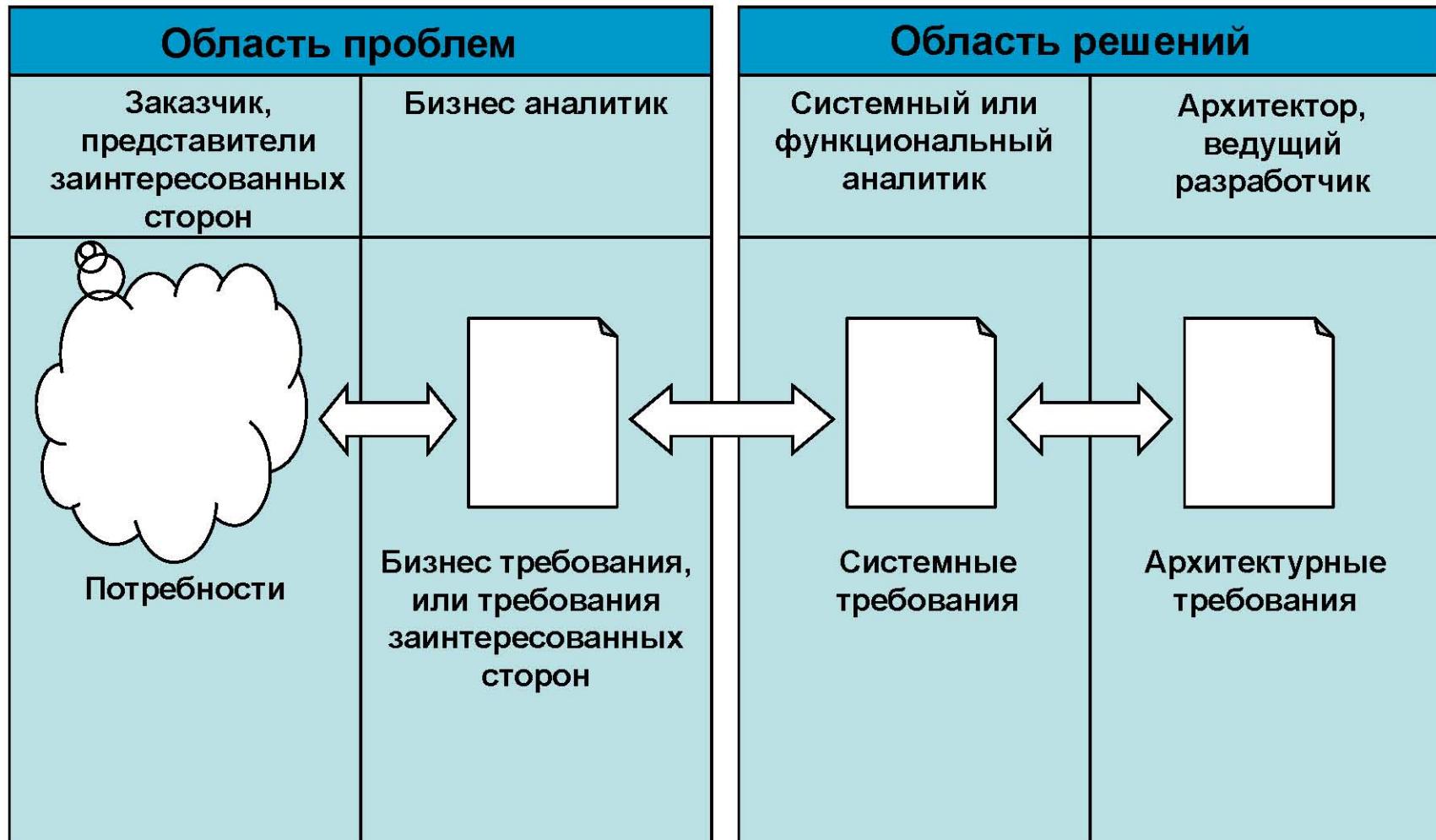
# Требования и тестирование

- **Тестирование системы:**
  - Приемочное тестирование
  - Функциональное тестирование
  - Нагрузочное тестирование и измерение производительности
  - Модульное тестирование
- **Тестирование требований:**
  - Проверки (review)
  - Формальные инспекции
  - Анализ требований с помощью моделирования
  - Согласования

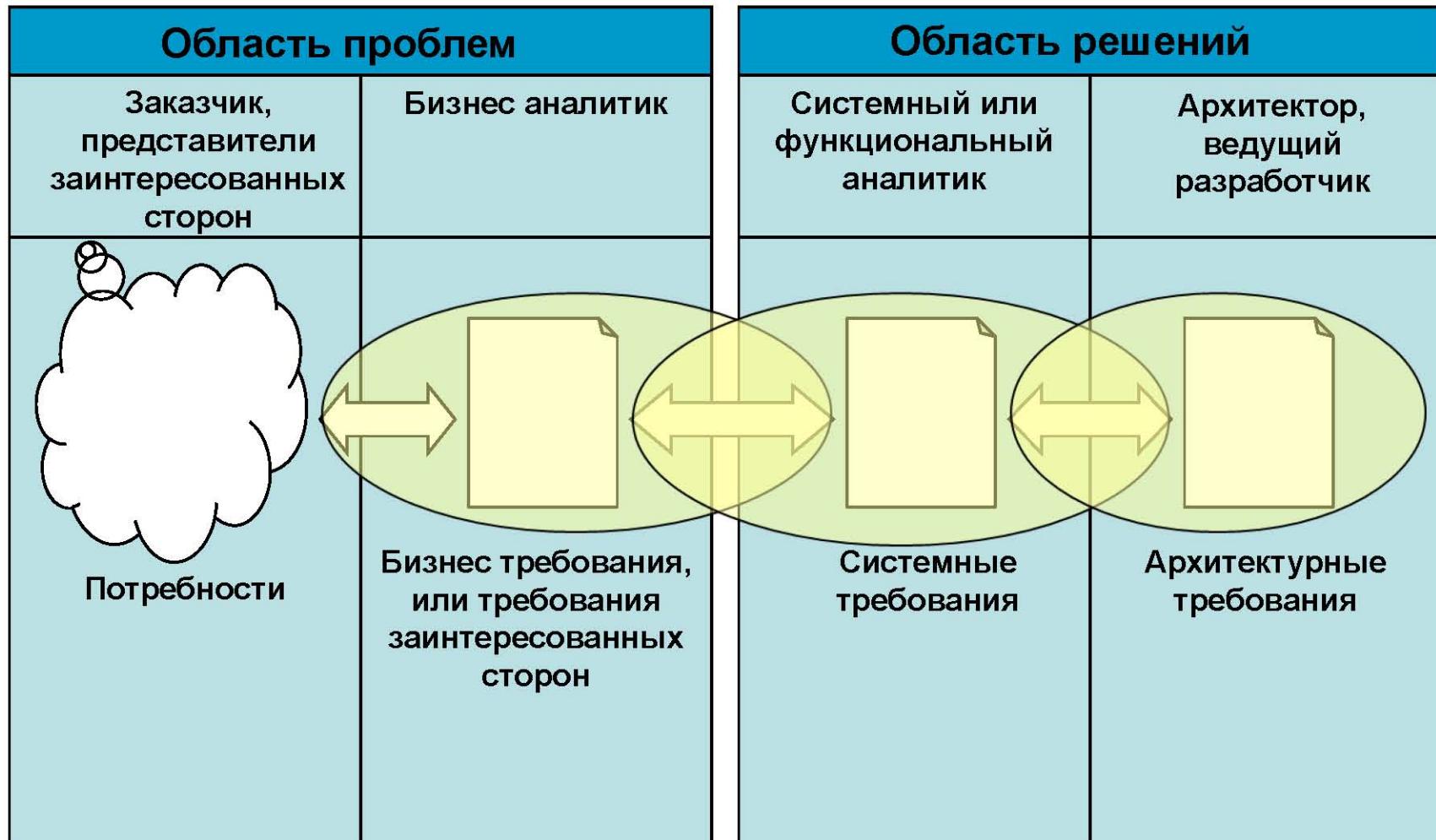
# Когда нет описания проблем

- недостаточное понимание настоящих проблем;
- невозможность определить рамки системы, и понять, что нужно включать, а что нет;
- преобладание разработчиков и исполнителей в дискуссиях о системе, поскольку единственное описание, существующее для системы, описывает ее в терминах реализации;
- невозможность нахождения наилучшего решения, из-за ограничений свободы в выборе решения.

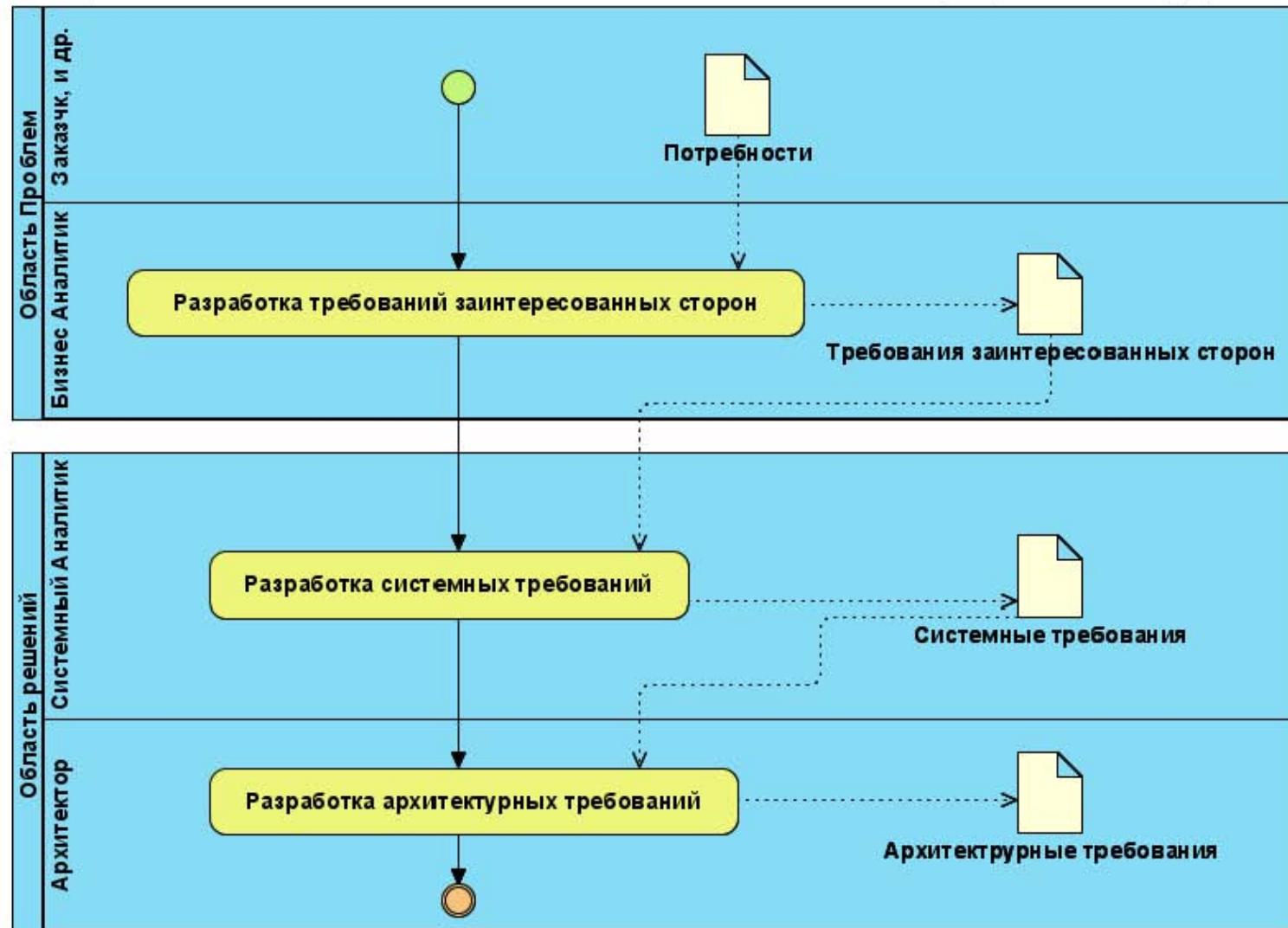
# Обобщенный процесс разработки требований для системы (пример)



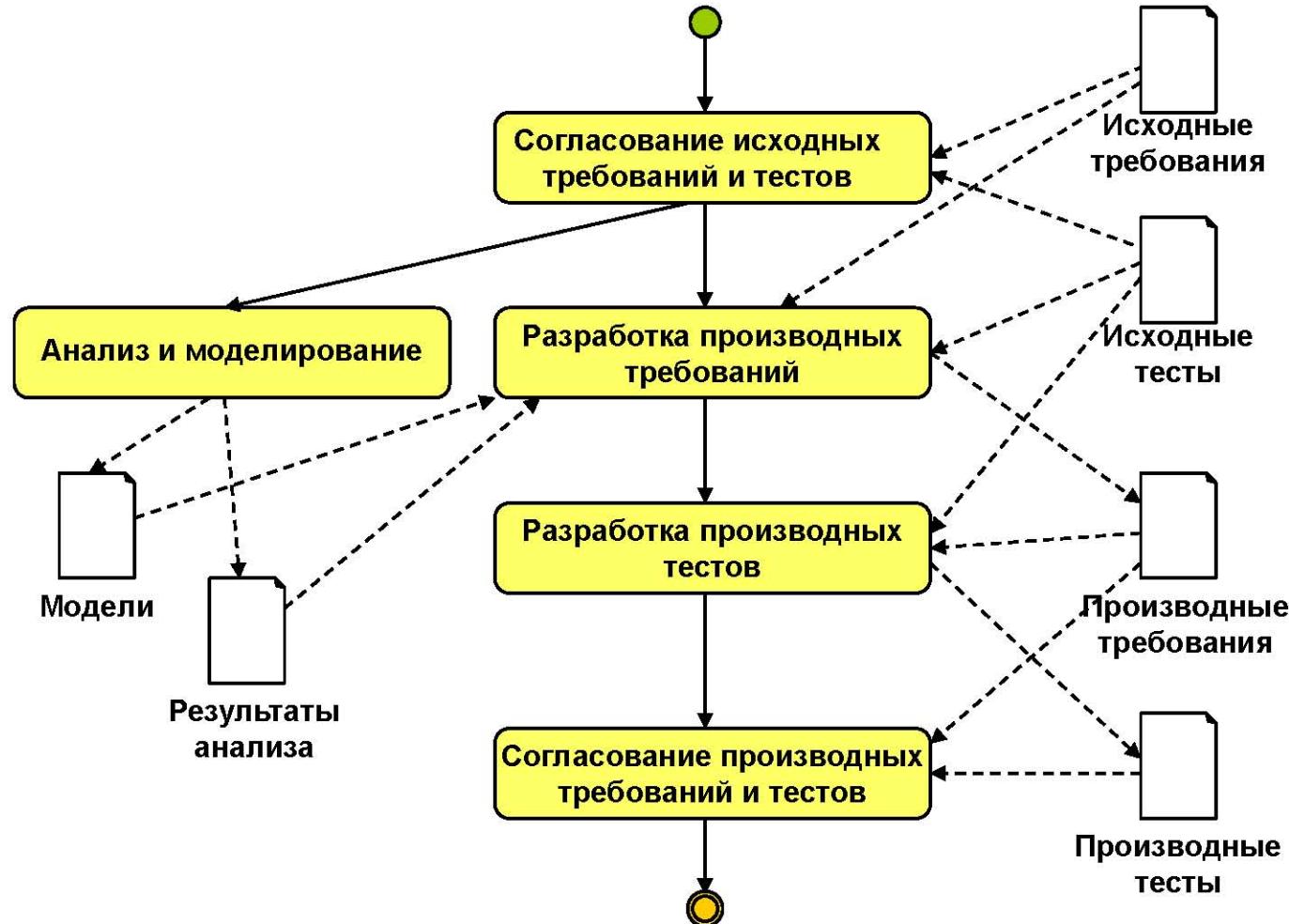
# Обобщенный процесс разработки требований для системы (пример)



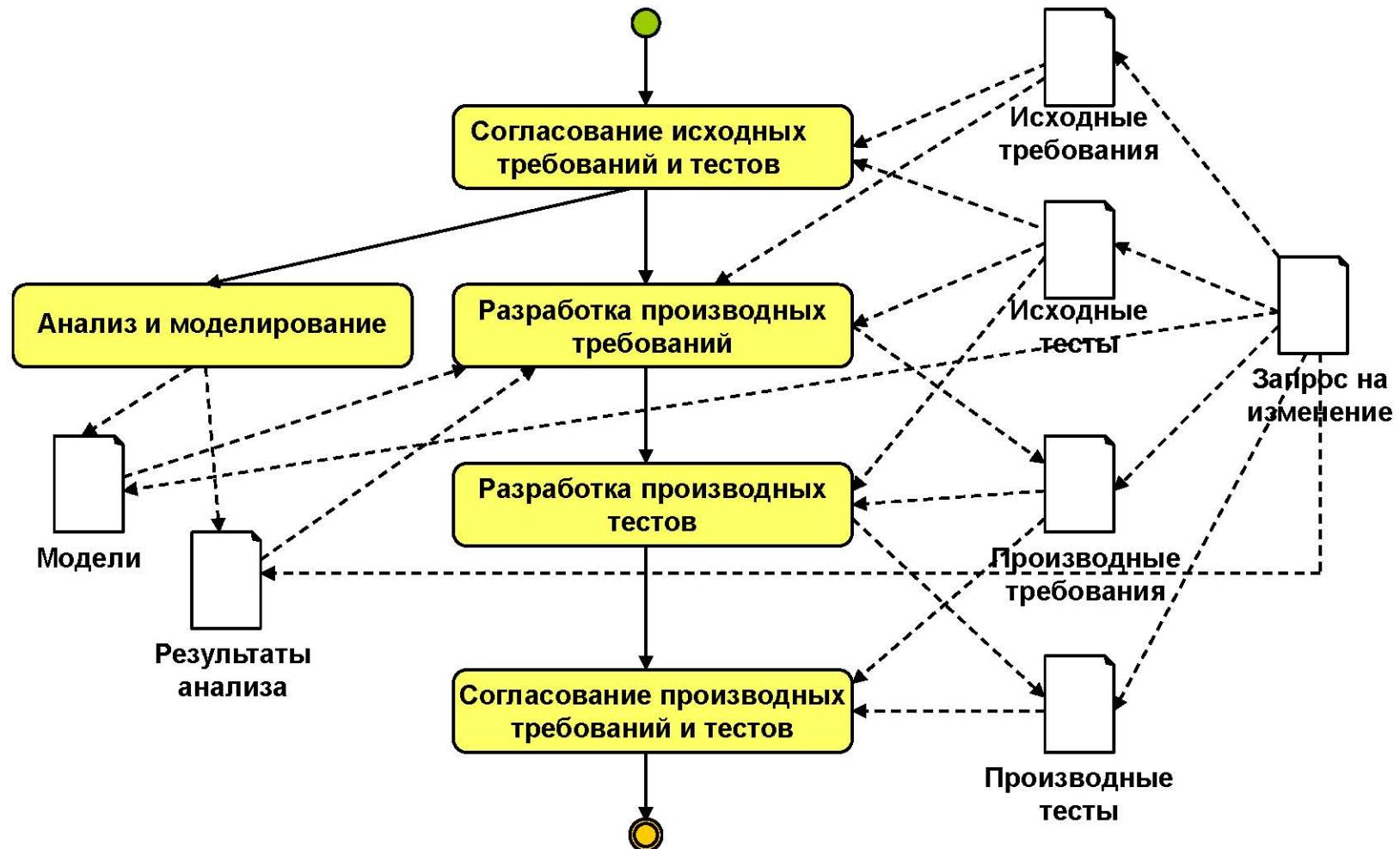
# Обобщенный процесс разработки требований для системы (пример)



# Общий процесс разработки требований для «идеального мира»



# Общий процесс разработки требований для «реального мира»



# Оценка требований

- Является ли требование полным?
- Является ли требование ясным?
- Является ли требование выполнимым?
- Является ли план тестирования и тесты понятным и приемлемыми?

# Причины отклонения требований

- Отсутствует часть информации – документах встречаются места оставленные для заполнения в дальнейшем (в документах на английском языке встречаются пометки типа TBA (to be agreed) – необходимо согласовать, TBC (to be complete) – необходимо завершить, TBD (to be decided) – необходимо принять решение).
- Недостаточно ясности – неоднозначность, противоречивость, путаница и т.д.
- Невозможно реализовать – никто не знает то, как это сделать.
- План проверки неприемлем.

# Действия характерные для каждого уровня

- согласование исходных требований с заказчиком;
- анализ исходных требований для определения рисков и потенциальных проблем, связанных с удовлетворением требований;
- создание одной или нескольких моделей для исследования возможных стратегий получения производных требований;
- Формирование производных требований при помощи результатов моделирования и анализа;
- разработка тестов для производных требований;
- согласование производных требований с командой (командами) ответственными за их реализацию;
- установление связей типа «удовлетворяет» между входными и производными требованиями;
- установление связей типа «проверяет» между производными требованиями и соответствующими тестами.

# Статусы требований

Требования характеризируются его тремя статусами:

- согласованность;
- удовлетворенность;
- проверяемость.

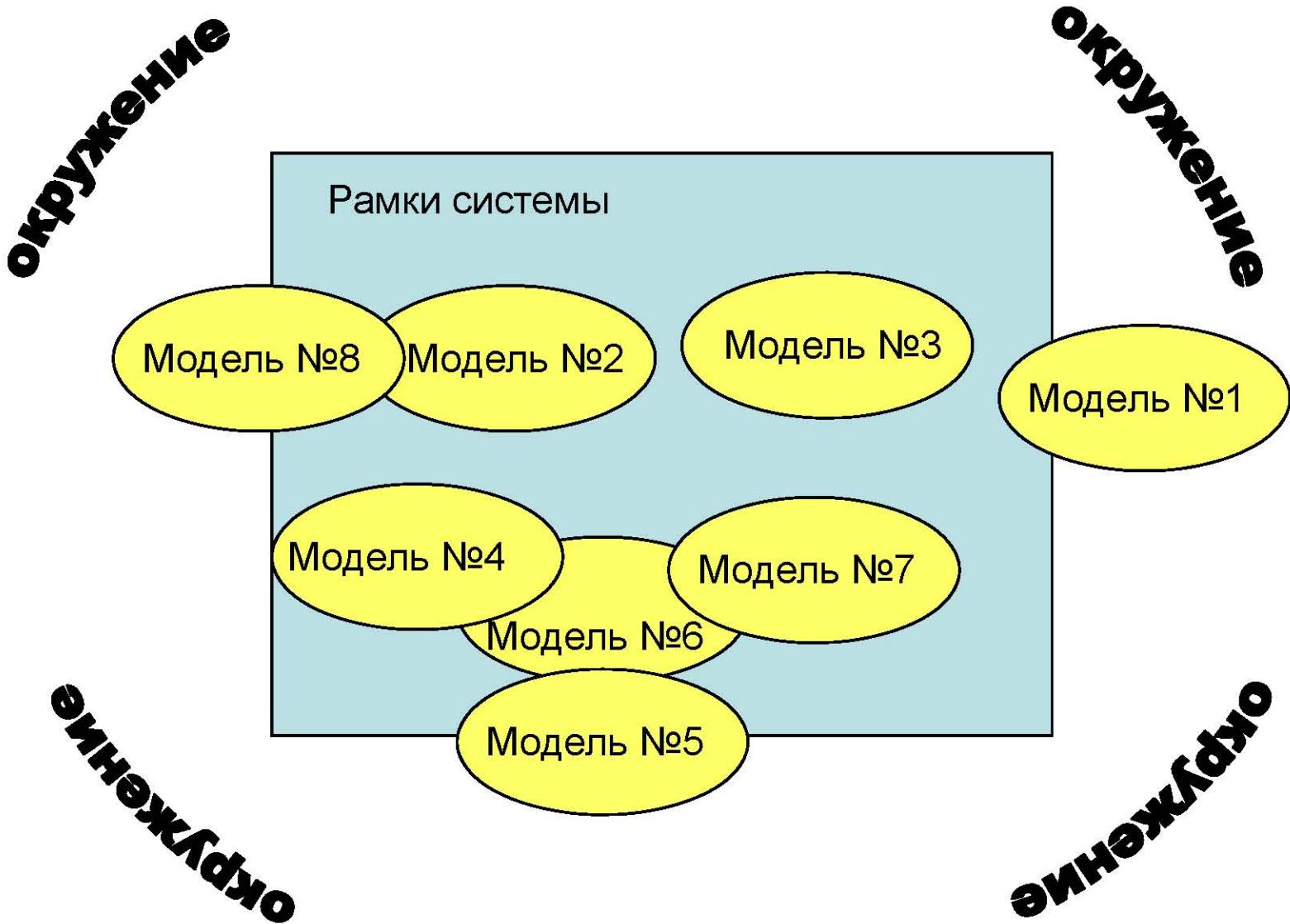
Идеальное состояние любого требования в любой системе заключается в том, что это требование:

- согласовано между заказчиком и исполнителем;
- имеет согласованную стратегию его проверки;
- удовлетворяется требованиями более низкого уровня (или спецификацией системы).

# Моделирование

- Можно смело утверждать, что моделирование это самая творческая часть работы аналитика или системного инженера.
- На практике не существует единственного «правильного» решения, поэтому модели меняются и развиваются на протяжении этапов разработки.
- Хорошая модель, это модель, помогающая общению.
- Модель может описывать структуру целой организации, или всего лишь одно определенное системное требование.
- Используемые методы моделирования определяются характером разрабатываемых систем, а также их областью их применения.

# Модели



# Преимущества которые дает моделирование

- Поощряет использование *точно определенной терминологии*, однозначность которой поддерживается в рамках разработки системы.
- Позволяет с помощью диаграмм получить *наглядное представление системных спецификаций и архитектуры* системы.
- Позволяет рассматривать *различные аспекты взаимодействия* системы с различных точек зрения.
- Поддерживает *системный анализ*.
- Позволяет проверить некоторые аспекты архитектуры системы с помощью динамических моделей.
- Позволяет *постоянно улучшать* систему путем уточнения архитектуры, поддерживая *генерацию тестов и исходного кода*.
- Позволяет свободно общаться *различным организациям* между собой используя стандартные нотации.

# Наиболее распространенные нотации

- Диаграммы потоков данных (DFD)
- Диаграммы сущность-связь (ERD)
- Диаграммы состояний (State Charts)
- Объектно-ориентированные нотации

# Объектно-ориентированные методы

- ОOA (Coad и Yourdon, 1991)
- ОМТ (Rumbaugh, 1991)
- Objectory (Jacobsen, 1993)
- Booch (Booch, 1994)
- UML (OMG, 1997)
- BPMN (BPMI, 2004)

# Методы основанные на перспективах

- Контролируемое формулирование требований (Controlled Requirements Expression (CORE))
- Техника структурного анализа и проектирования (Structured Analysis and Design Technique (SADT))
- Определение требований основанное на перспективах (Viewpoint-oriented Requirement Definition (VORD))

# Как соблюсти баланс?

Два очень важных аспекта должны находиться в постоянном балансе в процессе написания требований:

- требования должны быть удобными для чтения
- требования должны быть удобными для работы с ними

# Требования к документу

Документ (набор требований) должен позволять:

- однозначно идентифицировать каждое требование
- классифицировать каждое требование (по важности, по типу, по срочности)
- отслеживать статусы каждого требования
- определять атрибуты требованиям
- рассматривать каждое требование в контексте всего набора
- находить в наборе определенные требования по контексту, классификации или другим признакам
- устанавливать связи между требованиями и легко переходить по этим связям от одного требования к другому

# Структура документа

Хорошая структура требований позволяет:

- сократить количество требований;
- лучше понять большой объем информации;
- найти набор требований, относящийся к определенной теме или предмету;
- выявить пробелы и повторения;
- устранить противоречия между требованиями;
- управлять фазами (например, теми требованиями, которые были отложены);
- оценить требования (стоимость их реализации, риски);
- отклонить нежелательные требования;
- повторно использовать требования, в т.ч. в других проектах.

# Дополнительная информация

Документ с требованиями может также содержать следующую дополнительную информацию:

- *Историческая информация*, которая помогает представить требования в правильном контексте;
- *Внешнее окружение*, описывающее включающую систему, или как его часто называют «знания о предметной области»;
- *Определение объема требований* (что включено, а что нет, т.е. рамок проекта);
- *Определение терминологии* используемой в документе;
- *Описательный текст*, который служит для связи разделов документа между собой;
- *Описание заинтересованных сторон*;
- *Краткое описание моделей* используемых для получения требований;
- *Ссылки на другие документы*.

# Критерии законченного набора требований

- **полнота**: все необходимые требования записаны;
- **непротиворечивость**: не существует требований противоречащих друг другу;
- **отсутствие избыточности**: каждое требование сформулировано только один раз (нет повторов);
- **модульность**: требования близкие друг другу содержаться в одном разделе;
- **структурированность**: наличие ясной четкой структуры документа с требованиями;
- **удовлетворенность**: достигнут необходимый уровень покрытия требований связями типа «удовлетворяет»
- **тестируемость**: достигнут необходимый уровень покрытия требований тестами.

# Язык требований

Типы требований заинтересованных сторон:

- Возможности

**<Роль> должен иметь возможность <возможность>.**

**<Роль> должен иметь возможность <возможность>  
с(в) <показатель производительности> с <момент отсчета>  
находясь в <условия эксплуатации>.**

- Ограничения

**<Роль> не должен попадать под действие <соответствующе  
законодательство>.**

# Язык требований

Типы системных требований:

- Функции системы

**<Система> должна <выполняемая функция>  
не менее чем (с) <количество> <объект>  
функционируя в <условия эксплуатации>.**

- Ограничения

**<Система> должна <выполняемая функция> <объект>  
каждые <производительность> <единица измерения>.**

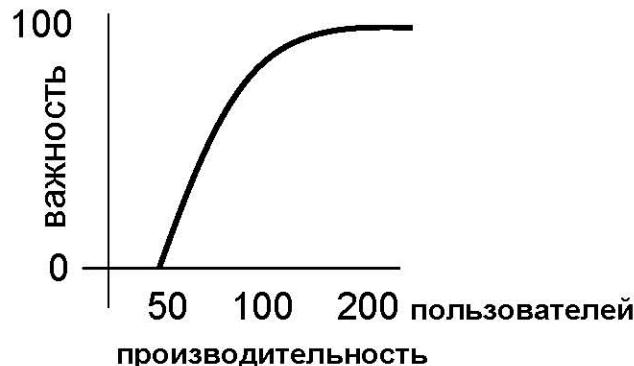
# Критерии для написания текста требований

- **атомарность**: каждое требование должно представлять собой один элемент иерархии требований, пригодный для установки связей с ним;
- **的独特性**: каждое требование должно иметь собственный уникальный идентификатор;
- **выполнимость**: требование должны быть технически реализуемо в установленные сроки, в рамках выделенного бюджета;
- **законность**: не должно противоречить применимому законодательству;
- **ясность**: должно быть понятно сформулировано;
- **точность**: должно быть точно и лаконично;
- **роверяемость**: должна существовать возможность проверки реализации каждого конкретного требования;
- **абстрактность**: не должно навязывать определенные технические решения, характерные для более низких уровней требований (спецификаций).

# Ключевые требования

- KURs (key user requirements – ключевые пользовательские требования) или
- KPIs (key performance indicators – ключевые показатели производительности)

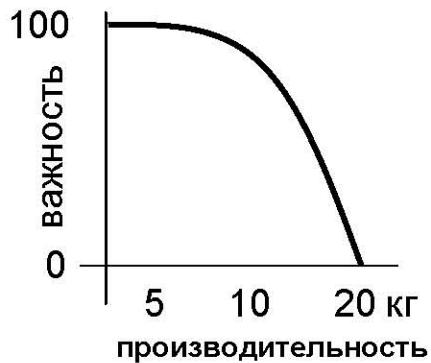
# Критерии важности/необходимости в требованиях



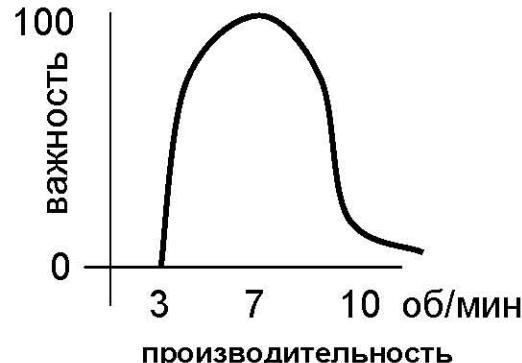
(а) максимизация



(б) определенная величина



(с) минимизация



(д) оптимизация

# Атрибуты требований

- *Идентификация*
  - Идентификатор
  - Название
- *Внутренние характеристики*
  - Основной тип
  - Качественный подтип
  - Тип продукта/процесса
  - Количественный/качественный тип
  - Фаза жизненного цикла
- *Приоритет и важность*
  - Приоритет
  - Важность
- *Источник и владелец*
  - Способ получения
  - Источник
  - Владелец
  - Согласовано
- *Контекст*
  - Набор требований/документ
  - Тема
  - Границы (рамки)
- *Проверка и утверждение*  
*(Verification and Validation, V&V)*
  - V&V метод
  - V&V стадия
  - V&V статус
  - Критерий успешности проверки
  - Критерий утверждения
  - Поддержка процесса
  - Статус согласования
  - Статус проверки
  - Статус удовлетворения
  - Статус рецензирования
- *Уточнение*
  - Необходимость
  - Комментарии
  - Вопросы
  - Ответы
- *Прочее*
  - Зрелось (стабильность)
  - Уровень риска
  - Оценочная стоимость
  - Фактическая стоимость
  - Релиз продукта

# Использование шаблонов

► Шаблон 34

<Система> должна <выполняемая функция> <объект>  
каждые <производительность> <единица измерения>.

Требование 347 + Шаблон 34

<система> = кофе машина  
<выполняемая функция> = производить  
<объект> = горячий напиток  
<производительность> = 10  
<единица измерения> = секунд

Требование 348 + Шаблон 34

<система> = кофе машина  
<выполняемая функция> = производить  
<объект> = холодный напиток  
<производительность> = 5  
<единица измерения> = секунд

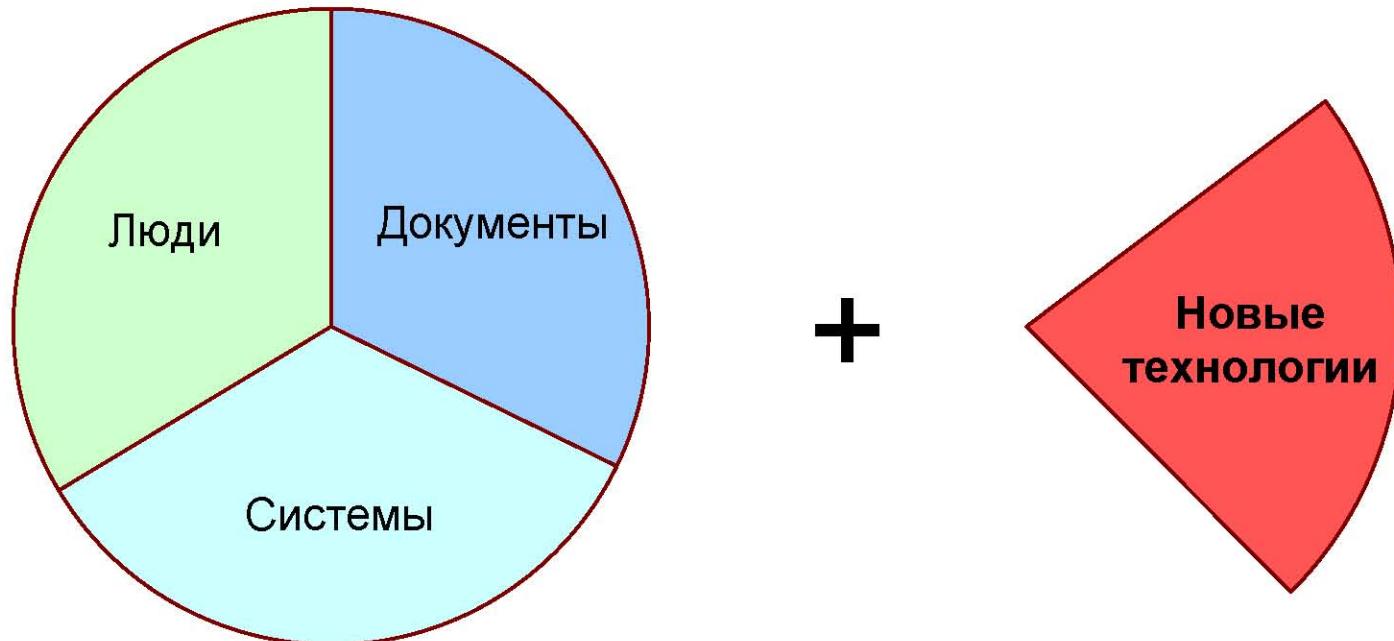
# Использование шаблонов

- *Возможность глобального изменения стиля:* для изменения формулировки определенных требований необходимо изменение только одного или нескольких определенных шаблонов.
- *Возможность более легкой обработки информации:* например, выделение всех «условий эксплуатации» в отдельный атрибут требований позволяет более удобно фильтровать и сортировать требования исходя из признака условий эксплуатации.
- *Возможность защиты конфиденциальной информации:* шаблоны могут быть использованы для защиты именно той части текста требования, которая должна быть защищена, в случае если требования содержат конфиденциальную или секретную информацию.

# Дополнительные аспекты работы требований

- Проблема использования естественного языка
- Словарь бизнес области
- Согласованная терминология в области решений
- Многоязычные проекты
- Однозначность различных распространенных естественных языков

# Источники требований



# Источники требований - Люди

- индивидуальные интервью;
- групповые интервью;
- семинары рабочих групп (workshops);
- опросы;
- наблюдение за работой людей;
- временное выполнение аналитиком текущей работы будущих пользователей системы.

# Интервью

- **До**
  - Согласование списка участников и организационная подготовка;
  - Подготовка вопросов для интервью (и их рассылка).
    - Структурированное или не структурированное интервью
    - Открытые или закрытые вопросы
  - Определить место проведения интервью
    - Переговорная/рабочее место/нейтральная территория
- **Во время**
  - Как сесть?
  - «Установить контакт»
  - Введение
  - Задавать вопросы и фиксировать ответы
    - Бумага или компьютер?
    - Самый главный вопрос «Зачем?»

# Интервью (продолжение)

- **Во время (продолжение)**
  - Вам нужно понять не только требуемые возможности, но и налагаемые ограничения
  - В конце спросить: «какие вопросы не заданы, но на них есть желание ответить?»;
  - Повторить коротко основные договоренности, чтобы убедиться в том что все правильно зафиксировано;
  - Сообщить план дальнейших действий.
- **После**
  - Прислать протокол (meeting minutes);
  - Начать писать требования как можно раньше.

# Групповые интервью

- *Вам необходима очень веская причина чтобы делать групповые интервью, если у вас ее нет, лучше их не делать.*
- Недостатки групповых интервью:
  - Часть людей может не прийти, так как решит, что за них скажут другие;
  - Концентрация не на существующих проблемах, а на личных амбициях;
  - Политические игры;
  - Страх показаться некомпетентным;
  - Страх перед присутствующим руководством.

# Семинары рабочих групп

- Возможно самый мощный способ сбора требований;
- Собирает всех представителей заинтересованных сторон для определения требованиями в небольшом но насыщенном интервале времени;
- Используется приглашенный ведущий для руководства мероприятием.

# Подготовка семинара

- Определение всех представителей заинтересованных сторон и участников со стороны команды разработки;
- Подготовительные материалы;
- Выбор подходящего времени и места;
- Выбор ведущего.

# Проведение семинара

- Ведение:
  - О месте проведения;
  - Повестка дня;
  - Правила проведения;
  - Текущее состояние проекта, цели семинара.
- Мозговой штурм;
- Определение возможностей и ограничений;
- Определение приоритетов и рамок проекта/фаз;
- Определение дальнейших шагов.

# Телефонные переговоры

- Вы теряете порядка 70% информации;
- Необходимо снабдить собеседника всеми материалами заблаговременно;
- Использование виртуального пространства для проведения телефонных интервью;
- Лучше использовать структурированное интервью и закрытые вопросы.

# Переписка

- Минусы:
  - Вы теряете до 90% информации;
  - Вы теряете время;
- Плюсы:
  - Необходима в рамках согласования и утверждения;
  - Проще с точки зрения организации – адресат может прочитать и ответить в любое удобное время;
  - У другой стороны есть время подумать и обсудить с коллегами прежде чем ответить.

# Опросы

- Помогают собрать требования когда представителей заинтересованных сторон много и не все они доступны;
- Помогают выделить представителей заинтересованных сторон;
- Помогают выявить наибольшее число требований при меньших затратах;
- Открытые и закрытые вопросы.

# Источники требований - Системы

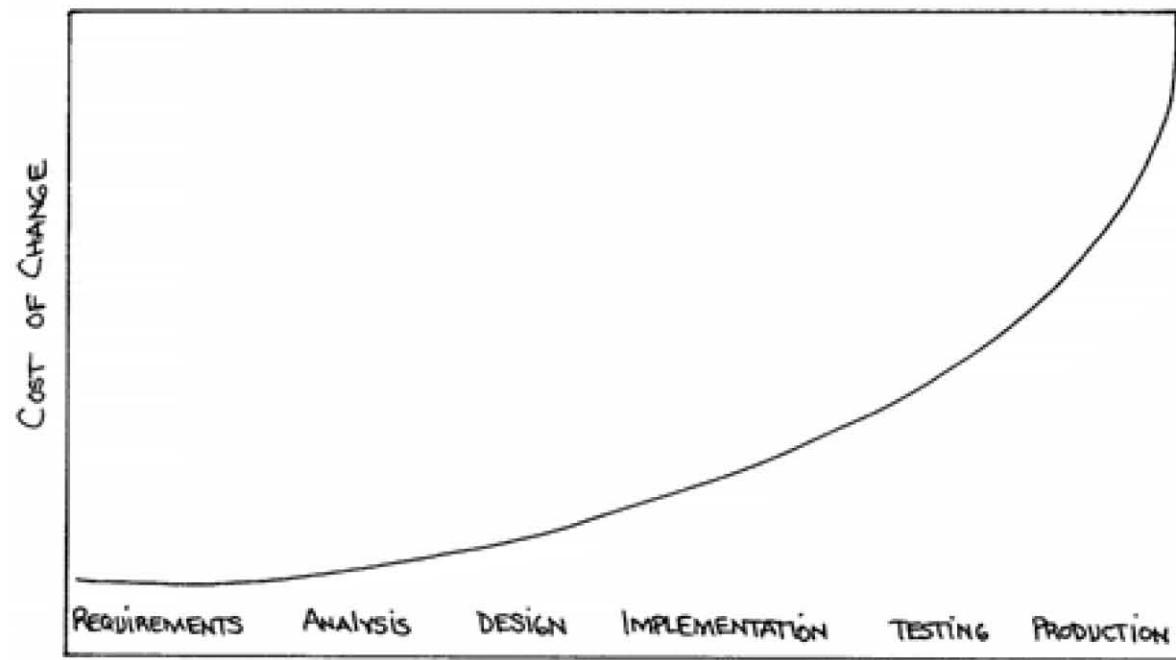
- системы, которые используются в настоящий момент и будут заменены разрабатываемой системой;
- системы, которые будут взаимодействовать с разрабатываемой системой;
- конкурирующие системы, продающиеся на рынке или разработанные в конкурирующих компаниях для выполнения аналогичных задач.

# Источники требований - Документы

- документация на существующие (заменяемые), аналогичные или конкурирующие системы;
- документация на системы, с которыми разрабатываемая система будет взаимодействовать;
- обзоры, рейтинги, аналитические отчеты и сравнения функциональности систем присутствующих на рынке;
- внутрикорпоративные стандарты, процедуры, регламенты, должностные инструкции, описание производственных процессов и т.п.;
- законодательство;
- отраслевые стандарты, ГОСТы и т.п.;
- международные стандарты;
- обращения в службу поддержки.

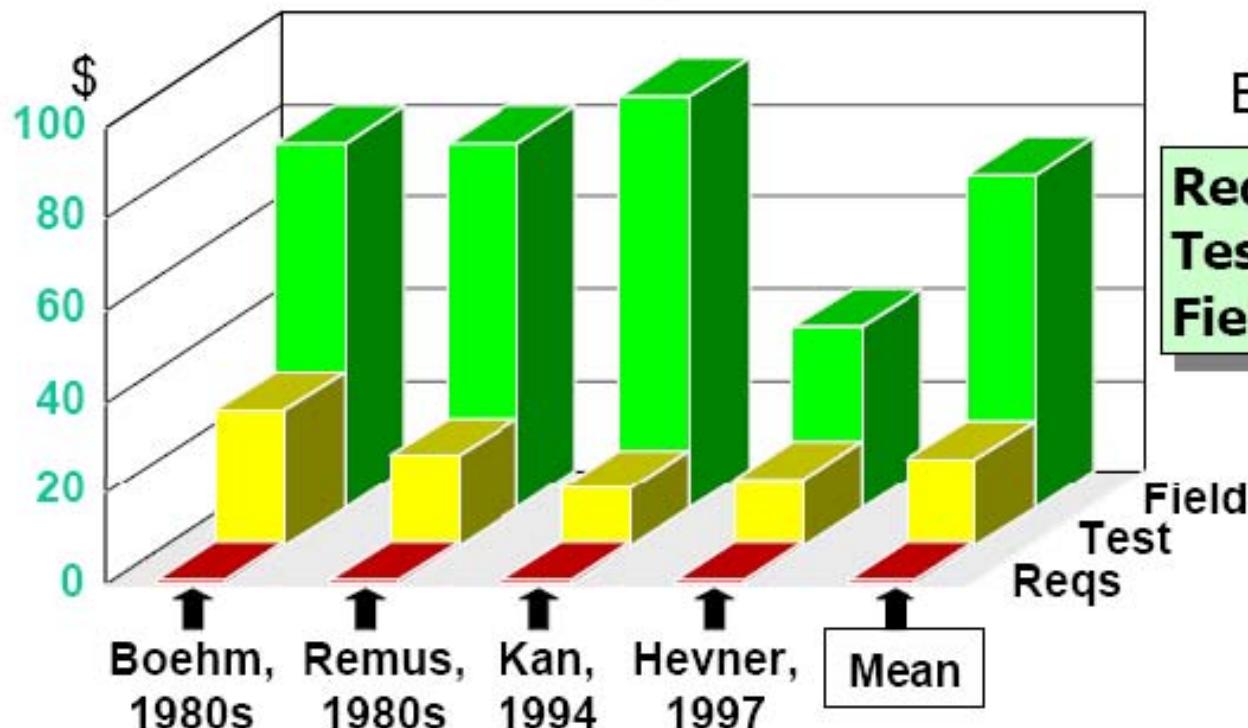
# Роль проверок

- Проверка документов (требований, моделей, спецификаций) позволяют избежать эффекта «падающего домино»
- Формальные инспекции документов и кода, по проверенным данным, позволяют снизить количество ошибок на 80%.
- Чем раньше удаётся устранить ошибки, тем больше удается сэкономить затраты на реализацию проекта.



# Стоимость ошибок

Cost of Defects by Phase Located



Example:

<b>Reqs</b>	\$200
<b>Test</b>	\$2,600 – \$6,000
<b>Field</b>	\$8,000 – \$18,000

# Роль аналитика

**В роли аналитика должно выступать незаинтересованное лицо обладающее:**

- умением эффективно взаимодействовать с заказчиками и поставщиками;
- высоким уровнем знаний в предметной области;
- ясно и четко излагать мысли (письменно);
- структурным и процессным мышлением
- умением работать в команде
- аккуратностью, позволяющей добиться высокой степени точности фиксирования информации и принятых решений
- высокой организованностью и мотивацией

# Семь правил чтобы избежать плохих требований

- *избегать хаос:* необходимо сконцентрироваться на самом важном, требование не должно быть похоже на роман;
- *избегать «лазеек»:* таких как «если это необходимо», поскольку такие «лазейки» делают требование бесполезным;
- *избегать помещать больше одного требования в один параграф:* зачастую можно определить, что в одном параграфе больше одного требования по наличию «и»;
- *избегать рассуждений;*
- *избегать нечетких слов:* обычно, в основном, часто, нормально, типично;
- *избегать использования неопределенных терминов:* удобный в использовании, универсальный, гибкий;
- *избегать принятие желаемого за требуемое:* 100 % надежный, приятный для всех пользователей, безопасный, подходящий для всех платформ, не должен никогда ломаться, обрабатывать все неожиданные сбои, быть готов к модернизациям для любых ситуаций которые могут возникнуть в будущем.

# Надежный путь разработки требований

- Определите структуру требований, и постоянно улучшайте ее в процессе работы с требованиями.
- Записывайте требования как можно раньше, даже если они далеки от совершенства.
- Как можно раньше определите атрибуты требований.
- Как можно быстрее подготовьте первую версию требований, для того чтобы получить отзывы.
- Улучшайте требования в процессе работы удаляя повторения, преждевременные технические решения и несогласованность.
- Постоянно обсуждайте требования и собирайте замечания, делайте как можно быстрее новые версии требований.
- Демонстрация пользователям гораздо лучше, чем «экспертный» анализ.

Основные правила для написания требований:

- Используйте простой прямой язык.
- Пишите требования, которые можно проверить.
- Используйте определенную и согласованную терминологию.
- В одном требовании пишите только одно требование.

# Три ключевых фактора



# Управление разработкой требований – сложное дело

- В отсутствии должного опыта люди, не понимают какой объем работы нужно выполнить, чтобы разработать требования
- Люди не понимают различия между требованиями заинтересованных сторон и системными требованиями
- Процесс управления требованиями зависит от типа организации
- Сложно оценить какой объем работы из запланированного выполнен
- Наличие большого количества изменений

# Основные типы организаций

- **Заказчики** – организации, покупающие системы и использующие их для своих собственных нужд.
- **Поставщики** – организации, разрабатывающие системы на заказ для конечных заказчиков, или для других поставщиков или продуктовых компаний.
- **Продуктовые компании** – организации, которые разрабатывают и продают готовые продукты.

# Основные этапы управления разработкой требований

- Планирование
- Контроль хода выполнение работы
- Управление изменениями

# Заказчик - планирование

- Концепция, Спонсор
- Определение полного списка заинтересованных сторон
- Сбор требований заинтересованных сторон (возможности и ограничения), способы сбора требований
- Атрибуты требований
- Проверки и согласования

# Заказчик – контроль разработки требований

- определение структуры спецификации требований;
- определение атрибутов каждого из требований;
- определение процесса рецензирования требований в соответствии с перечнем критериев рецензирования.

# Заказчик – контроль изменений

Разная степень формализации подхода к изменениям в зависимости от этапа реализации проекта:

- Внесение изменений без контроля
- Предупреждение о внесении изменений коллег
- Формальный процесс предложения, утверждения и внесения изменений

# Тендер глазами заказчика

- Причины проведения тендера:
  - нехватка внутренних ресурсов
  - нехватка экспертизы (скалируемости)
  - преимущества продукта
- Определение участников конкурса:
  - внутренние ресурсы
  - внешние ресурсы
  - поставщики продуктов
- Этапы тендера:
  - Исследование рынка
  - RFI (Request for Information – Запрос на предоставление информации)
  - RFP (Request for Proposal – Запрос на коммерческое предложение)
  - принятие решения

# Пример структуры RFI/RFP

- Конфиденциальность
- Введение
  - Цель проекта
  - Объем и рамки проекта
  - Зависимости, связи и ссылки на другие документы
- Представление компании Заказчика
- Инструкции участникам тендера:
  - Основные условия проведения тендера
  - Бюджет проекта и принципы оплаты работы
  - Право Заказчика вносить изменения в RFI/RFP
  - Право Заказчика на отклонения предложения
  - Требования к участникам тендера
  - Требования к ответам
  - Стоимость предлагаемого решения

# Пример структуры RFI/RFP

- График проведения тендера:
  - Отправка RFI/RFP
  - Подтверждение намерения участвовать в тендере
  - Вопросы и ответы на них
  - Ответы на RFI/RFP
  - Презентации ответов на RFI/RFP участниками тендера
  - Выбор Поставщика (короткого листа Поставщиков)
  - Заключение контракта (в случае выбора Поставщика)
- Критерии выбора Поставщика:
  - Экспертиза команды
  - Техническая сторона предложения
  - Стоимость
- Приложения:
  - Требования и вопросы к участникам тендера (о самой компании)
  - Требования (заинтересованных сторон или системные)

# Поставщик - Планирование

- Анализ полученных от Заказчика требований
- Вопросы и предложения Заказчику

Ваши вопросы и предложения могут попасть к вашим конкурентам! В этой ситуации вы можете:

- полностью проигнорировать проблему;
- сделать какое-то предположение (допущение), позволяющее устраниить проблему, и, зафиксировав его документально, двигаться дальше;
- принять решение, что, независимо от последствий, необходимо задать вопрос Заказчику.

- Оценка требований Заказчика
- Подготовка Предложения

# Поставщик - Планирование

- Необходимо сохранять все тендерные материалы, включая все, вопросы которые были не заданы, предположения, которые были сделаны, и т.п.
- Большой срок между проведением тендера и заключением контракта
- Разные команды на тендере и на реальном проекте
- Субподрядчики
- После заключение контракта:
  - Уточнение вопросов и проблемных требований
  - Планирование разработки детальных требований
  - Определение приоритетов и распределение по фазам

# Поставщик – контроль выполнения работ

- Анализ и оценка исходных требований
- Моделирование предлагаемого решения
- После заключения контракта:
  - Контроль в соответствии с планом проекта
  - Контроль субподрядчиков

# Поставщик – контроль изменений

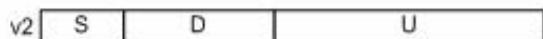
- Источники изменений:
  - заказчик;
  - поставщики (субподрядчики);
  - внутренние источники.
- Объекты контроля изменений:
  - входящие требования;
  - требования для поставщиков и субподрядчиков (исходящие требования);
  - допущения, предположения и интерпретации, сделанные командой, разрабатывающей коммерческое предложение.

# Продуктовая компания

- Заказчик и Поставщик в рамках одной организации (характерно так же для In-House разработки)
- Несколько модификаций продукта
  - Локализации
  - По полноте функционала
- Несколько версий одного продукта

# Несколько версий и модификаций одного продукта

Базовый продукт



Модификация А



Модификация В



- Базовый набор требований
- Требования характерные для версии
- Требования характерные для модификации

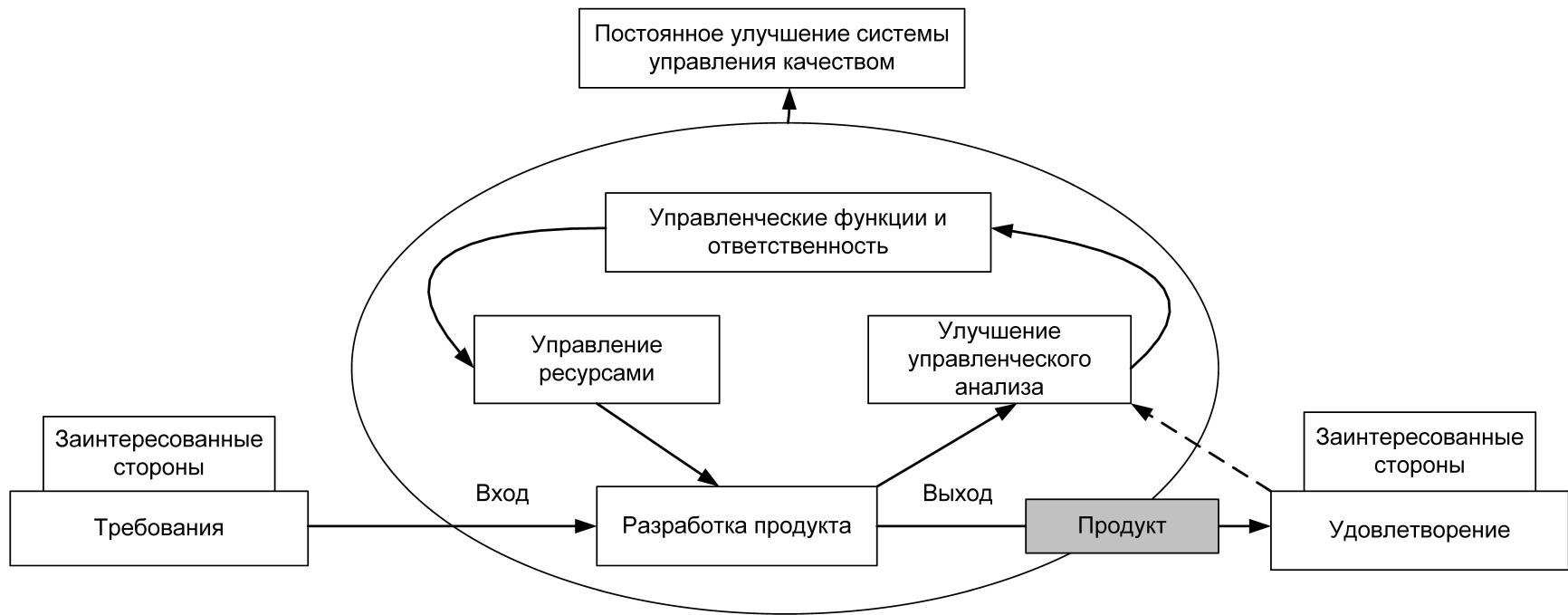
# CMM/CMMI - Уровни

№ уровня	Название уровня	Ключевые области процесса
1	Начальный	Если организация находится на этом уровне, то ключевых областей процессов для нее не предусмотрено
2	Повторяющийся	Управление программными конфигурациями.Обеспечение качества программных продуктов.Управление контрактами подрядчиков.Контроль за ходом проектов.Планирование программных проектов.Управление требованиями
3	Определенный	Экспертные оценки.Координация взаимодействий проектных групп.Инженерия программного продукта.Комплексный менеджмент ПО.Программа обучения персонала.Определение организационного процесса.Область действия организационного процесса
4	Управляемый	Менеджмент качества ПО.Управление процессом на основе количественных методов
5	Оптимизируемый	Управление изменением процесса.Управление технологическими изменениями.Предотвращение дефектов

# СММ/СММІ

- Метрики
  - Статус для каждого требования
  - Изменения требований и их количество
  - Количество изменений сделанных между версиями (метками)
- Процедуры
  - Процессы связанные с изменениями требований
  - Определение влияния изменения
  - Согласование изменений
  - Контроль за изменениями

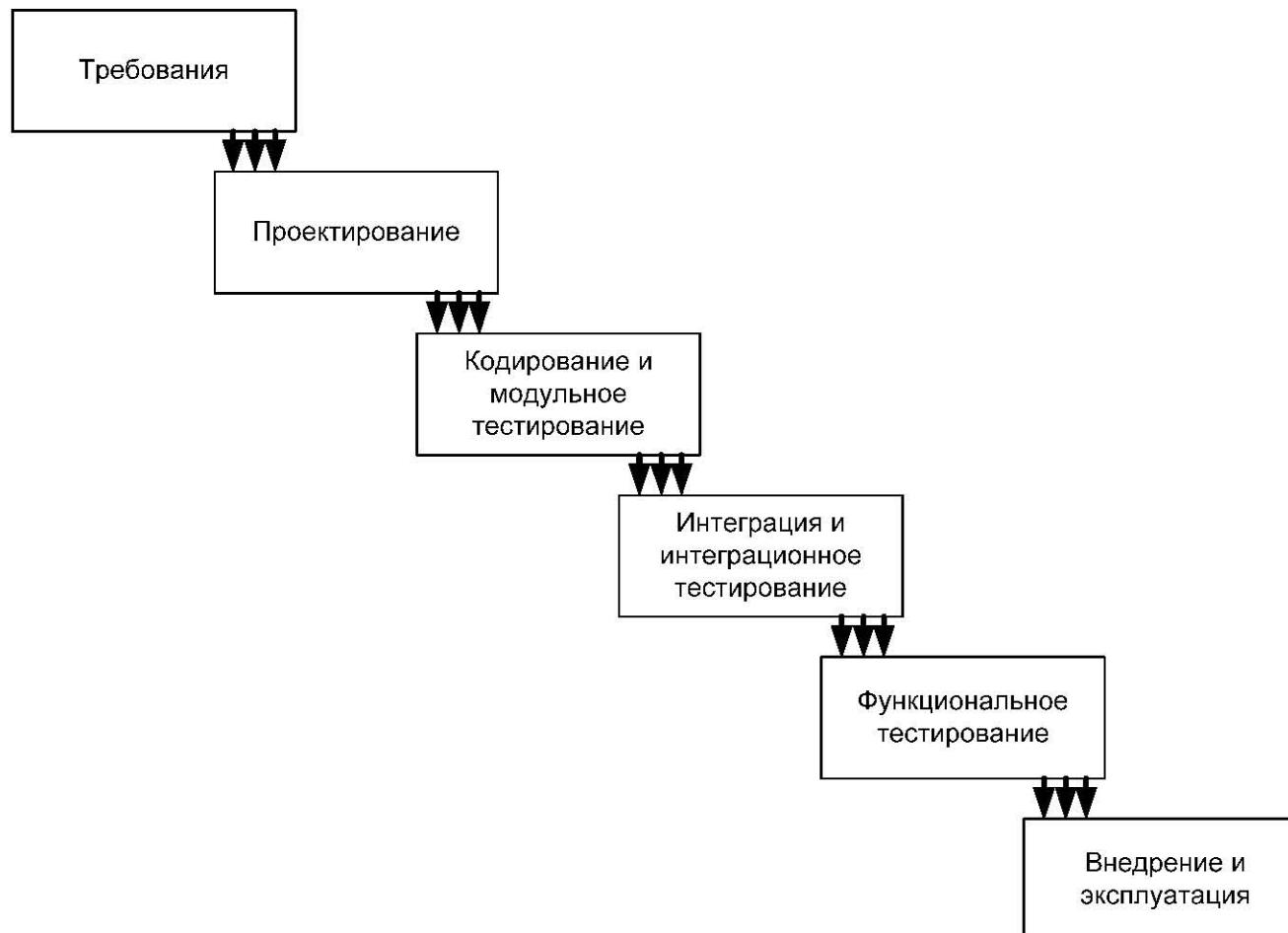
# ISO 9000:2000



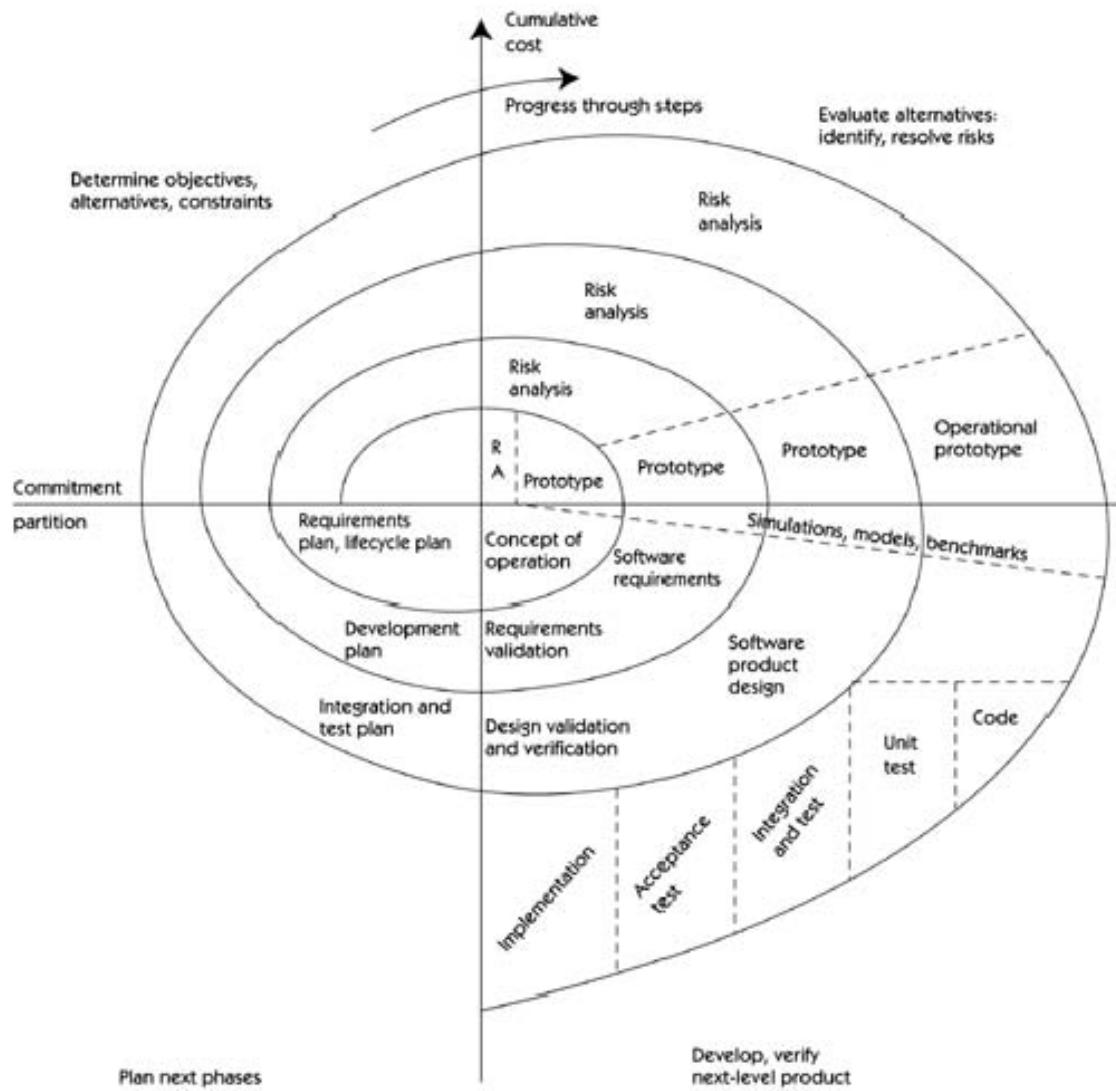
# СММ/СММІ и ISO 9000

Ни СММ/СММІ ни ISO 9000 не определяет то как необходимо разрабатывать и управлять требованиями. Они обязывают вашу организацию определить, внедрить эффективный процесс управления требованиями и неукоснительно его придерживаться.

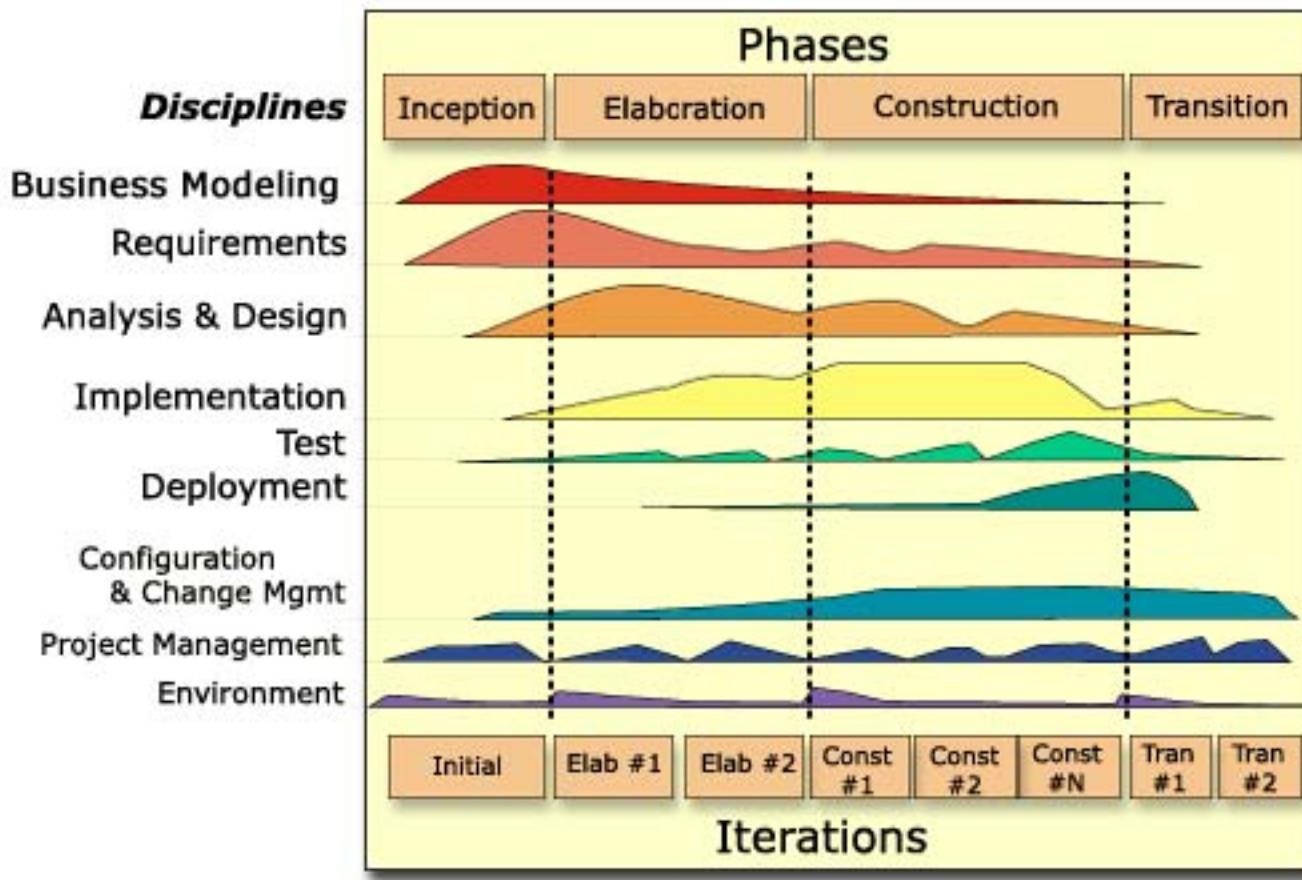
# Водопадная МОДЕЛЬ



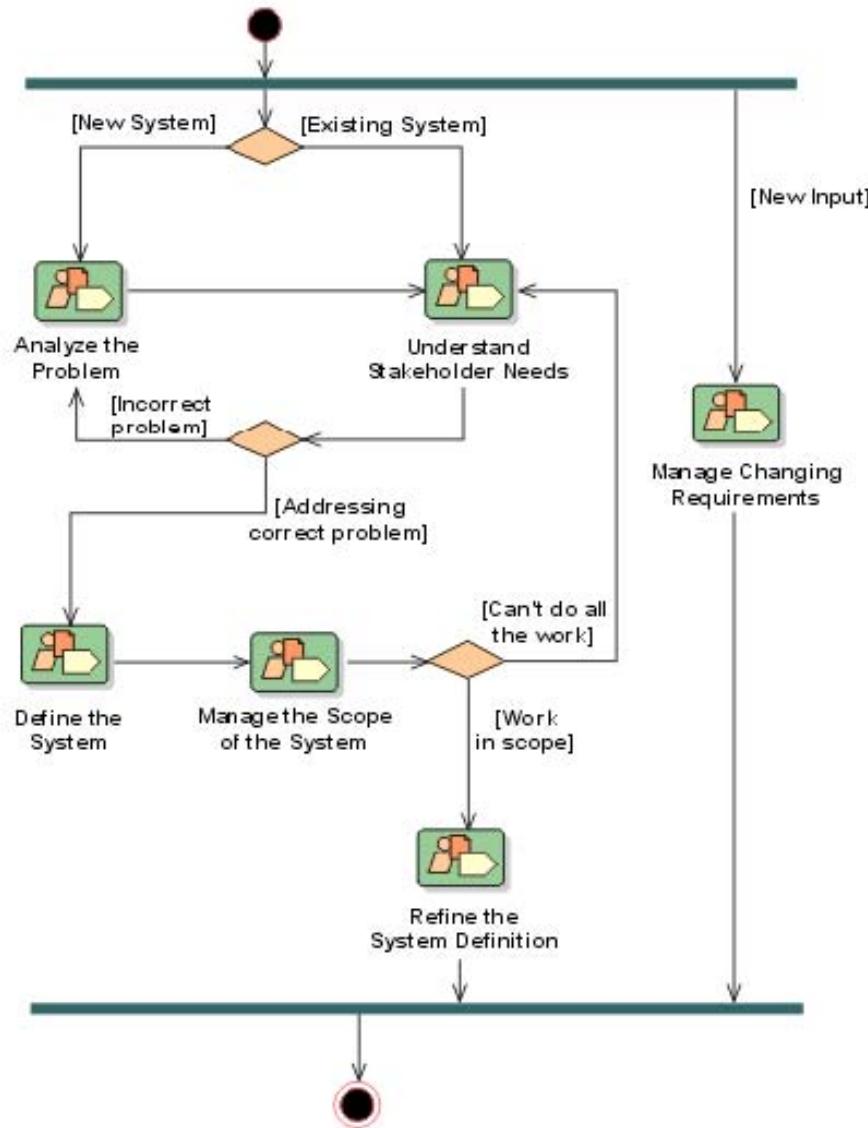
# Спиральная МОДЕЛЬ



# RUP – итеративная модель



# Требования в RUP



# Гибкие (agile) методологии

- eXtreme Programming
- Scrum
- DSDM
- Crystal
- ...



Nokia 888 гибкий телефон будущего

# Гибкие (agile) методологии – 12 практик

- **Игра в планирование** — Быстрое определение объема следующего релиза путем сопоставления приоритета со стороны бизнеса и оценок со стороны разработки. В случае если работа не соответствует плану, меняется план.
- **Маленькие релизы** — В эксплуатацию сначала выводится очень простая система с небольшим количеством функционала, через короткий промежуток времени выходит следующий релиз с дополнительным функционалом и т.д.
- **Метафора** — Позволяет всем разработчикам и участникам проекта понять как работает вся система с помощью небольшого предложения (истории).
- **Простая архитектура** — Система должна быть очень простой. Вся излишняя сложность должна устраняться как только находится.
- **Тестирование** — Программисты постоянно пишут модульные тесты (unit tests), до того как напишут код, тесты поддерживаются всегда в актуальном состоянии. Заказчик пишет приемочные тесты, которые помогают разработчикам лучше его понять.
- **Refactoring** — Программисты переписывают системный код, для того чтобы сделать архитектуру системы проще, гибче, эффективнее не изменяя при этом функционал системы.

# Гибкие (agile) методологии – 12 практик

- **Парное программирование** — Весь код системы пишется парами программистов, каждая из которых работает за одной машиной вместе(одновременно – только один человек имеет доступ к клавиатуре, другой помогает идеями и выявляет ошибки).
- **Совместное владение кодом** — Любой разработчик может изменить любую часть системы в любое время.
- **Постоянная интеграция** — Успешные сборка и построение системы несколько раз в день.
- **40 часовая рабочая неделя** — Правилом является работать не более 40 часов в неделю. Никогда переработки не повторяются две недели подряд.
- **Доступный заказчик** — представитель заказчика – реальный будущий пользователь системы сидит в одной комнате с командой разработчиков чтобы отвечать на вопросы, писать приемочные тесты, участвовать в планирование и т.д.
- **Стандарты программирования** — Программисты пишут код в соответствии с принятыми стандартами, чтобы он был проще и понятнее для всех членов команды.

# Сравнение формальных и гибких (agile) подходов к требованиям

Аспекты	Формальные	Гибкие
<b>Корректность (точность)</b>	x	x
<b>Непротиворечивость</b>	x	x
<b>Приоритеты требований</b>	+/-	x
<b>Необходимость (целесообразность)</b>	x	x
<b>Простота внесения изменений</b>	-	x
<b>Тестопригодность (возможно проверить)</b>	x	x
<b>Полнота</b>	x	-
<b>Однозначность интерпретации</b>	x	x
<b>Реализуемость</b>	x	x
<b>Покрытие связями (traceable)</b>	x	-

# Свежая статистика

- Какую модель процесса разработки ПО использует ваша команда?



Источник: The 2008 State of Requirements Management Report, © 2008 Jama Software 121

# Недостатки использования документов

- Трудно поддерживать в актуальном состоянии
- Проблемы с нотификацией заинтересованных лиц об изменениях
- Тяжело хранить дополнительную информацию
- Трудно связывать требования между собой и внешними объектами (требования других уровней, тесты, модели, задачи и т.п.)

# Причины использования средства управления требованиями

- Управление версиями и изменениями
- Атрибуты требований
- Связи между требованиями и другими объектами
- Возможность оценить прогресс выполнения работы
- Выделение части требований (сортировки и фильтры)
- Управление доступом
- Как средство командной работы и связи со всеми заинтересованными лицами

# Критерии выбора системы управления требованиями

- Возможность импорта существующих требований в систему
- Гибкая конфигурация атрибутов для различных типов требований
- Возможность создания версий и меток (*Versions and Baselines*)
- Возможность связи требований между собой
- Интеграция с другими системами возможность связи требований с объектами внутри этих систем
- Управление доступом
- Система экспорта и подготовки документов

# Критерии выбора системы управления требованиями

- Наличие системы запросов на изменение
- Возможность включать объекты (файлы, графику, диаграммы) в текст требований
- Возможность обсуждения требований
- Автоматические нотификации
- Web interface
- Удобство использования
- Примеры проектов, обучающие материалы, документация, доступность курсов

# Сравнение систем для управления требованиями

- INCOSE Requirements Management Tools Survey  
<http://www.paper-review.com/tools/rms/read.php>
- Requirements Tools  
<http://www.volere.co.uk/tools.htm>
- Requirements Management Tools A Qualitative Assessment  
[http://eprints.cs.vt.edu/archive/00000656/01/RM\\_Tools.pdf](http://eprints.cs.vt.edu/archive/00000656/01/RM_Tools.pdf)

# Свежая статистика

- Как ваша команда документирует и обсуждает требования?

