Требования к информационным системам

Бессарабов Н.В.

Использованы материалы книги

Ю.А. Маглинец “Анализ требований к автоматизированным информационным системам” – М. 2008

и презентация “Формирование требований при разработке ПО”

Требования к продукту и процессу его разработки

“В начале было Слово, и Слово было у Бога, и Слово было Бог.

Оно было в начале у Бога.

Все чрез Него нáчало быть, и без Него ничто не нáчало быть, что нáчало быть.”

Евангелие от Иоанна, гл. 1, стих 1-4

1. Требования к продукту – основной класс требований.
2. Требования к процессу разработки выдвигаются обычно для того, чтобы уменьшить риски заказчика.

Понятно, что регламентация процесса разработки приводит к дополнительным накладным расходам. Однако существуют ситуации, в которых необходимо самый детальный контроль процесса разработки.

Типичный случай -- выполнение оффшорного проекта. Разработчик может создать согласованный с заказчиком план работы с детализацией до дней и конкретных исполнителей. По нему разработчик выполняет ежедневные сборки, регрессионное тестирование разрабатываемых компонентов и тестирование продукта в целом.

Уровни требований к ПО

Требование - это "условие или возможность, которым должна соответствовать система" (русская редакция RUP).

"Требования – это исходные данные, на основе которых проектируются или создаются автоматизированные информационные системы" (Маглинец).

Уровни требований:

- На верхнем уровне находятся бизнес-требования (от топ-менеджеров).
- Второй уровень это требование пользователей.

Как правило, требования этого уровня формируется плохо, имеют противоречия, дублирование и плохую структуру.

- Третий уровень -- **функциональные требования**, которые определяют функции реализуемые системой, и **нефункциональные требования**.

Возможно появления противоречий в требованиях различных уровней. Так, требование полноты информации может противоречить требованию предоставления минимально необходимой для управления информации.

Правило "одна точка сбора": "Данные собираются там, где они появляются."

Функциональные и нефункциональные требования

Функция (feature) – предоставляемая системой услуга для удовлетворения одной или нескольких потребностей заинтересованных лиц.

Функциональные требования определяют поведение системы и устанавливают цели, задачи и предоставляемые сервисы. Описываются они с помощью предписывающих правил, например, “система должна позволять сформировать приходные и расходные накладные”. Другой способ -- это варианты использования Use Cases из стандарта UML.

Нефункциональные требования описывают атрибуты системы или же атрибуты системного окружения, задающие:

- особенности эксплуатации (реентерабельность, программная и аппаратная совместимость, адаптивность, свойства поддержки);
- производительность;
- свойства реализации (стандарты, языки, ОС и др.);
- атрибуты качества и надежности (частота сбоев, возможн. восстановления);
- интерфейсы, включая регламенты взаимодействия сущностей;
- атрибуты качества.

Интерфейсы могут быть аппаратными, программными и

Атрибуты требований

- Приоритет (высокий, средний, низкий); порядок очередности при планировании работ или при проектировании продукта.
- Статус (предложено, одобрено, утверждено, реализовано, верифицировано);
- Стоимость (высокая, средняя, низкая или числовое значение);
- Сложность реализации (высокая, средняя, низкая);
- Стабильность (высокая, средняя, низкая); отражает степень постоянства требования. Перечисляются все требования, которые могут быть изменены на протяжении ЖЦ.
- Исполнитель (группа, конкретный исполнитель или характеристики класса исполнителей).
- Возможность верификации - возможность проверки присутствия данного требования на разных этапах разработки.

Важно: система требований должна быть полной и согласованной.

Атрибуты качества

Основные **атрибуты качества**:

- применимость
- надежность,
- производительность,
- эксплуатационная пригодность.

Ограничения -- это формулировки условий модифицирующие требования или наборы требований которые сужают возможность решения по их реализации , выбор платформы и особенности развертывания.

Характеристика продукта в целом это набор связанных функциональных требований, которые обеспечивают возможности пользователя и удовлетворяют бизнес-целям.

Модель FURPS+

Модель FURPS+ это одна из классификаций требований. Используется в Rational Unified Process.

Аббревиатура FURPS составлена по первым буквам названий видов требований:

- Functionality (функциональные требования)
- Usability (требования к удобству работы)
- Reliability (требования к надежности)
- Performance (требования к производительности)
- Supportability (требования к простоте поддержки)

Знак плюс "+" в аббревиатуре FURPS+ означает, что учитываются дополнительные виды требований:

- ограничения на структуру проекта
- требования к реализации
- требования к интерфейсу
- физические требования.

Функциональные требования и удобство работы

Функциональные требования это:

- набор выполняемых функций
- возможности
- защита

Удобство работы:

- человеческий фактор
- эстетика
- последовательность пользовательского интерфейса
- электронная и контекстная справочные системы
- мастеры и агенты
- пользовательская документация
- учебные материалы

Надёжность

Надежность это:

- частота и серьезность сбоев
- возможность восстановления
- предсказуемость
- точность
- средняя продолжительность бесперебойной работы (MTBF)

Производительность

Требования к **производительности** накладывают определенные условия на функциональные требования. Например, для определенной операции можно задать следующие требования к производительности:

- скорость выполнения
- эффективность
- коэффициент готовности
- точность
- пропускная способность
- время отклика
- время восстановления
- используемые ресурсы

Удобство поддержки

Требования к **удобству поддержки** охватывают следующие вопросы:

- простота тестирования
- простота расширения
- простота адаптируемости
- простота обслуживания
- совместимость
- простота настройки
- простота обслуживания
- простота установки
- простота локализации

Требования к структуре проекта и реализации

Требования к структуре проекта охватывают вопросы, относящиеся к структуре системы. Их часто называют ограничениями на структуру.

Требования к реализации

Определяют особенности программирования и конструирования системы. Например:

- соответствие стандартам
- языки реализации
- правила в отношении целостности баз данных
- ограничения на ресурсы
- рабочие среды

Требования к интерфейсу. Физические требования

Требования к интерфейсу охватывают следующие вопросы:

- внешние объекты, с которыми должна взаимодействовать система
- ограничения на форматы, время ожидания и другие обстоятельства взаимодействия

Физические требования охватывают физические характеристики системы:

- материалы
- форма
- габариты
- вес

Эти требования могут применяться для описания особенностей аппаратного обеспечения, например, необходимых сетевых интерфейсов.

Требования к требованиям

- Полнота набора требований
- Ясность, недвусмысленность
- Корректность
- Непротиворечивость набора требований
- Проверяемость (верифицируемость, тестопригодность)
- Необходимость и полезность при эксплуатации
- Осуществимость
- Модифицируемость
- Трассируемость
- Ранжирование (упорядоченность)
- Наличие метрики

Полнота и ясность набора требований

Полнота не должна пониматься как в математической логике. Кроме того, только при каскадном подходе необходимо представлять будущее ПО до его создания и в каком то смысле полностью описать его. В рассматриваемой нами области полнота должна скорее пониматься в бытовом смысле как подробность описания, достаточная для достижения каких-то целей, обычно локальных.

В современных технологиях **полнота системы требований** всегда относительна.

Полнота отдельного требования означает, что описание требования не требует дополнительного уточнения или детализации.

Ясность или недвусмысленность требования достигается тогда когда все участники разработки начинают одинаково понимать требования.

На практике ясность достигается в процессе многочисленных консультаций, в ходе которых происходит согласование тезаурусов всех участников процесса.

Полезно составлять словарь употребляемых в разработке понятий, которые могут пониматься неоднозначно или нечётко.

Понятие ясность требования включает еще его прослеживаемость, начиная от первой формулировки вплоть до рабочих спецификаций программ.

Цитата из Вигерса: “Пишите документацию просто, кратко и чётко

Корректность, непротиворечивость и верифицируемость

Свойство **корректности** задает дихотомию: требование либо корректно, либо некорректно.

Необходимо также рассматривать взаимную корректность требований. Если два требования вступают в конфликт, то по крайней мере одно из них некорректно.

Вспомним, что требования характеризуется еще вертикальной и горизонтальный согласованностью. Иначе говоря, каждое требование не должно **противоречить** требованиям своего уровня иерархии и требованиям всех родительских уровней. В частности, требования пользователей не должны противоречить бизнес-требованиям,

а функциональные требования требованиям пользователей.

Свойства **верифицируемости** или возможность требования быть проверенным коррелируется со свойствами ясности и полноты. Если требование является полным, то есть в нем не опущены важные для реализации детали, и изложено на языке одинаково воспринимаемым всеми участниками процесса, то это свойство можно верифицировать.

Неверифицируемые требования не могут быть проверены. Поэтому они не должны включаться список требований к программному обеспечению,

который обычно является неотъемлемым компонентом договора на разработку ПО.

Необходимость и полезность при эксплуатации

Разграничение между свойствами требований "необходимость" и "полезность при эксплуатации".

Необходимым является требование, без выполнения которого невозможна либо затруднена реализация

автоматизированных бизнес-функция пользователей.

Любое свойство повышающее качество программного продукта можно считать **полезным**.

При обсуждении с заказчиком необходимости и полезности требования следует помнить о том, что большинство функциональных требований вытекают из требований бизнеса или пользователя. Но некоторые функциональные требования могут лежать вне компетенции заказчика.

Такие требования формулирует исполнитель, который представляет необходимость решений, не вытекающих непосредственно из бизнес-требований или требований пользователя.

Аргументы за или против включение требования будут лучше восприняты заказчиком если они будут излагаться в терминологии бизнеса заказчика.

Осуществимость и трассировка

Осуществимость требования можно оценить, сравнивая ценность требования, ресурсы потребные для его реализации, и учитывая предысторию работы по выявлению требований к ПО. Попросту говоря, если заказчик готов платить, а разработчик имеет возможность доработать проект с учетом изменений, то требование следует считать осуществимым.

Необходимо **поддерживать историю изменений требований**. Каждое требование должно быть записано только один раз. При необходимости следует использовать ссылки, но не дублировать требования.

Трассировка это прослеживание связей между требованием и связанными с ним моделями, документами, текстами программ.

Трассировка позволяет выявить:

- артефакты информационной системы, у которых нет связей ни с одним из требований.
- требования не включённые в список требований.
- требования, у которых нет связей с другими артефактами ИС.

Если трассировка выполнена, легче выяснить какие артефакты необходимо изменить при изменении требований.

Ранжирование, стабильность и метрики

Ранжирование требований задаёт их приоритеты, определяя тем самым соотношение затрат труда на их реализацию.

Стабильность это неизменяемость требования со временем.

Для некоторых требований, в первую очередь нефункциональных, могут задаваться количественные **метрики**, например, время отработки в секундах, средняя наработка на отказ в часах и т.д.

Предупреждение: Требования не должны содержать описания проекта или деталей проектирования и реализации, то есть, они должны отвечать на вопрос “что делать”, а не “как делать”.

Заметим, что упомянутые ранее (слайд 11) требования к структуре проекта это ограничениями на структуру, но не её описание.

Принятие проектных решений на стадии составления требований может быть губительным для проекта.

Причины неудач связанных с требованиями

Причины неудач:

- Неточность, денотационная неясность, недоопределённость , и неполнота в формулировках требований
- Недостаточное вовлечение заказчика в работу над проектом
- Недостаточное вложение ресурсов
- Плохое планирование и управление проектом
- Слишком частое или редкое изменение и фиксация требований
- Несовершенство используемой технологии
- Недостаточная поддержка со стороны руководства
- Недостаточно высокая квалификация разработчиков
- Отсутствие опыта специфичного для проекта
- Пренебрежение чужим или своим накопленным опытом

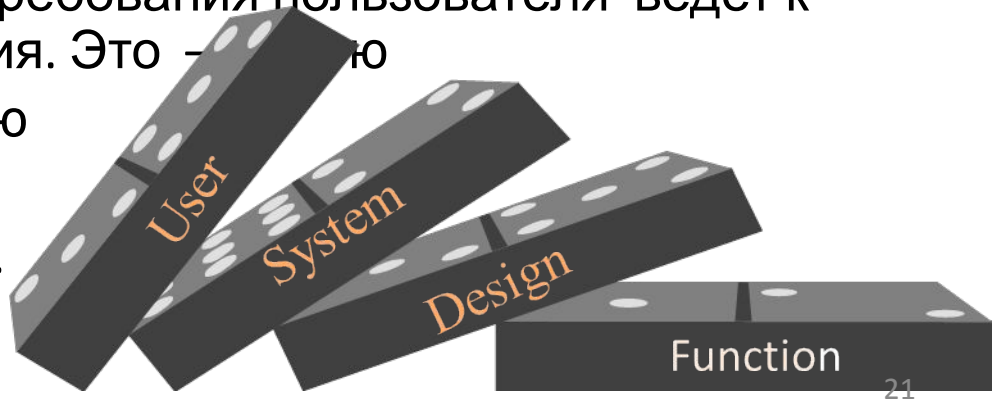
Причины неудач проектов. Эффект домино

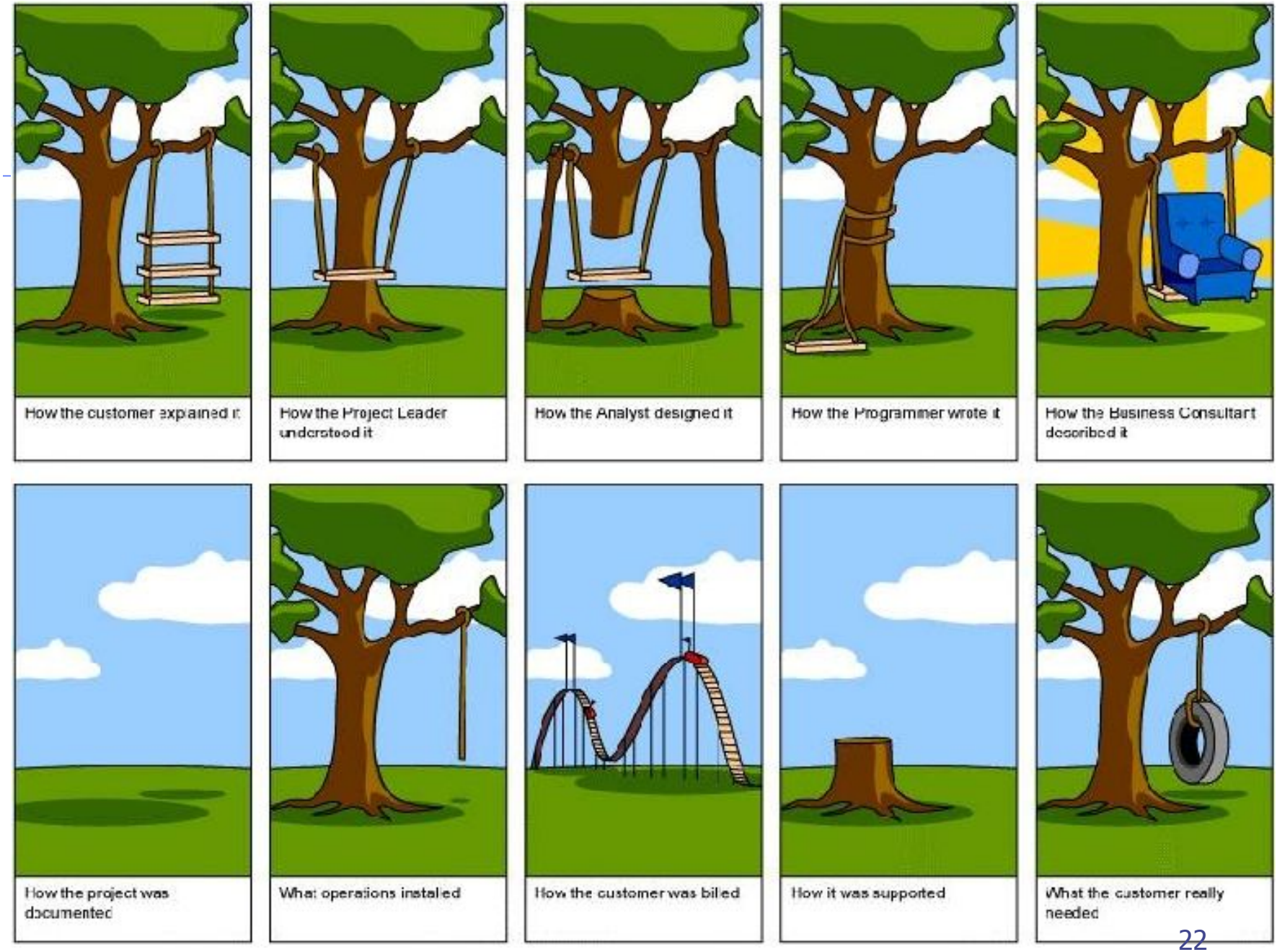
Ошибки в требованиях – самые дорогостоящие и самые распространенные ошибки. Составляют до 70% стоимости переделки продукта и до 30-40% всего бюджета проекта. Следует помнить, что:

- требования формулируются на естественном языке, приходят из разных источников, основывающихся на разных онтологиях/тезаурусах;
- требования разнотипны и могут выдвигаться в количестве, которое трудно контролировать;
- требования связаны между собой и другими проектными данными; требования изменяются на разных этапах ЖЦ ПО.

Неправильная работа с требованиями на любом из этапов жизненного цикла может вызвать эффект домино.

Например, пропуск (неучет) требования пользователя ведет к пропуску системного требования. Это – в свою очередь – приводит к отсутствию элемента дизайна, а потому и к отсутствию функциональности.




How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Контекст задачи анализа требований

Контекст задачи анализа требований определяется в первую очередь

1. Использованием моделей бизнеса (каких) либо словарей (тезаурусов)
2. Выбором методологии анализа бизнеса

После этого необходимо:

1. Выработать согласованное определение решаемой проблемы.
2. Выделить основные причины вызвавшие появление проблемы.
3. Выявить заинтересованных лиц и пользователей ПО.
4. Определить границы решения.
5. Выявить ограничения, которые необходимо наложить на решение.

Этап 1. Определение решаемой проблемы

Вопрос 1: Воздействием чего на что(кого) и результатом чего является проблема

Вопрос 2: Какое решение принимаем. Перечислить преимущества решения

Вопрос 3: Какие проблемы создаёт принятое решение (немного Голдратта)

Этап 1. Выделить основные причины вызвавшие появление проблемы.

Пример: Простая ИС

Этапы 3 и 4.Выявление заинтересованных лиц и пользователей, определение границ

Вопросы для выявления заинтересованных лиц и пользователей:

- Кто будет пользователем системы?
- Кто является заказчиком системы?
- Кто распоряжается деньгами?
- На кого окажут влияние введение системы?
- Кто будет оценивать и принимать систему?
- Существуют ли другие пользователи системы чьи потребности стоит учесть (во вторую очередь)?
- Кто будет сопровождать систему?
- Не забыли ли мы кого-нибудь?

Вопросы для определения границы системы:

- Кто будет управлять системой?
- Кто будет осуществлять сопровождение системы?
- Откуда система получает информацию?
- Какие внешние системы будут взаимодействовать с системой?

Этап 5. Выявление ограничений на решение

- Экономические (источники финансирования, конкуренция, себестоимость, лицензирование)
- Политические (отношения между подразделениями)
- Технические (используемые технологии, СУБД, операционные системы, условия эксплуатации)
- Системные
- Эксплуатационные
- Графики выполнения работ и ресурсы

Выявление требований

Методы выявления требований:

- Интервьюирование;
- Анкетирование;
- Совещания;
- Мозговой штурм и отбор идей;
- Прецеденты;
- Обыгрывание ролей;
- Создание прототипов, в том числе кликабельных.

Интервью

Преимущества:

- Малые затраты труда для респондента;
- Легко перестраивать ход опроса.

Недостатки:

- Сильно зависит от квалификации интервьюера;
- Высокая трудоёмкость.

Этапы:

- Подготовка.
- Проведение опроса.
- Завершение и отчёт.

Правила проведения интервью

“Никогда не задавай вопросов, на которые не знаешь всех ответов”

*Одно из пра*вил дипломата

- Длительность не более 1, 2 часов.
- Проводить не перед обедом и не перед концом рабочего дня.
- Четко представлять цель интервью.
- Объяснить свою роль респонденту перед началом интервью.
- Ограничить число прорабатываемых вопросов.
- Вопросы должны быть подготовлены и тщательно продуманы заранее
- Перед проведением интервью необходимо ознакомиться с предметной подобластью.
- Желательно познакомиться с имеющейся нормативной и другой документацией

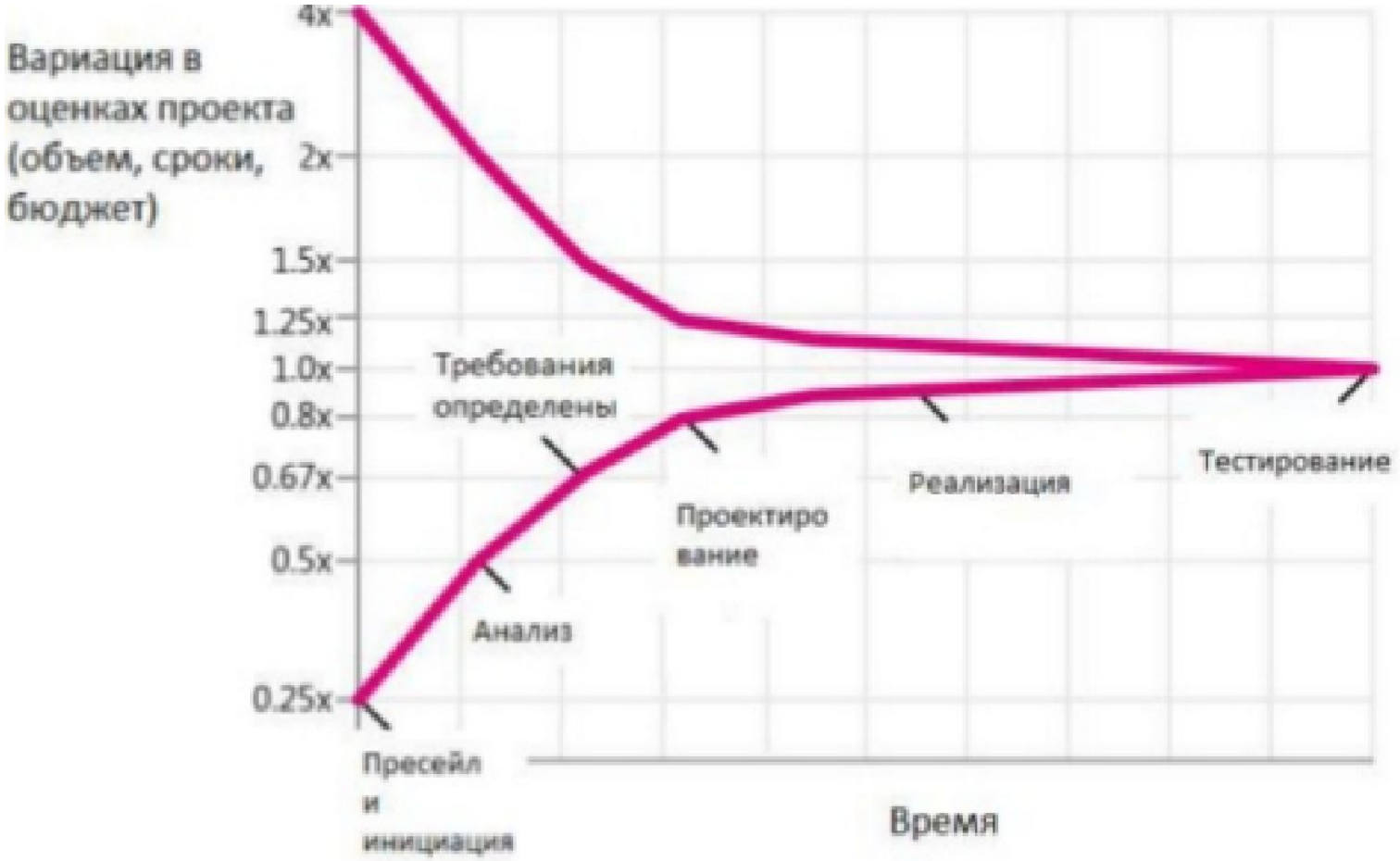
Что вы скажете респонденту перед интервью и некоторые правила проведения интервью

Перед интервью сообщим:

- Почему проводится это интервью
- Кто разрешил/приказал его проводить
- Кто еще будет проинтервьюирован (может быть!)
- Как и кто выбирал интервьюируемых
- Как будет использована полученная информация (осторожно!)
- Будет ли интервью анонимным
- Будет ли интервью отражено в отчете
- Какова будет обратная связь по итогам обработки результатов интервью
- Почему вам важно получить точную информацию

Некоторые правила:

- Не отвлекаться на темы, связанным с обсуждаемым предметом, но не входящие в тематику интервью
- Наладить доверительные отношения
- Не указывать интервьюируемому на его затруднения
- Давать респонденту время подумать
- Отделить факты от личных оценок
- Не иронизировать
- Концентрироваться на наиболее сложных аспектах
- Помнить, что эксперт не вы, а ваш респондент
- Не увеличивать длительность проведения интервью


Вариация в оценках проекта (объем, сроки, бюджет)
4x
2x
1.5x
1.25x
1.0x
0.8x
0.67x
0.5x
0.25x
Требования определены
Анализ
Проектирование
Реализация
Тестирование
Пресейл и инициация
Время

Программное обеспечение (ПО), как система, в свою очередь является подсистемой некоторой информационной системы