Оглавление

Введение в управление требованиями

Управление требованиями - это процесс, который включает идентификацию, выявление, документирование, анализ, отслеживание, приоритезацию требований, согласование требований, управление изменениями требований и уведомление заинтересованных лиц по планируемым изменениям. Данный процесс носит непрерывный характер на протяжении всего проекта.

Цель управления требованиями - гарантирование, что организация документирует, проверяет и удовлетворяет потребности и ожидания ее клиентов и внутренних или внешних заинтересованных лиц. Управление требованиями включает в себя поддержку требований, интеграцию требований и организацию работы с требованиями и сопутствующей информацией, которая появляется наряду с требованиями.

1. Проверка требований

После того, как требования собраны и задокументированы, необходимо проверить качество требований и разработанных документов. В проверке качества требований участвует вся команда разработки программного обеспечения, а также эксперты предметной или другие представители заказчика. Наиболее распространенным методом проверки спецификаций требований является проверка экспертом предметной области. Эксперт проверяет спецификацию на наличие предметных неточностей, противоречий. После проверки и исправления замечаний спецификация требований утверждается заказчиком и команда приступает к проектированию и разработке программного обеспечения. Проверка одним лишь экспертом часто приводит к тому, что спецификация не содержит ошибок в предметной области и весь функционал описан правильно, но, как показывает практика, требования могут быть сложно реализуемыми, не тестируемыми, дорогостоящими. Для выхода из подобной ситуации в проверке требований также должны участвовать инженеры по тестированию, системные архитекторы и руководитель проекта.

Для проверки требований рекомендуется также использовать руководства и контрольные списки, которые содержат рекомендации и критерии «хороших» требований.

1.1. Типичные проблемные ситуации процесса формирования и оценки требований

Двусмысленность требований

В ряду проблем и недостатков требований двусмысленность, является, пожалуй, наиболее критичным фактором риска проекта, закладываемого в фазе формирования требований. **Двусмысленность** (несоответствие свойству ясности, определённости) закладывает под проект «бомбу замедленного действия». На практике требование, сформулированное двусмысленным образом, может привести к различным его интерпретациям представителями Разработчика и Заказчика. Разработчик, руководствуясь своей интерпретацией, определит на её основе архитектурную основу, создаст аналитические и проектные модели и в конечном итоге создаст программный код. Как показывают исследования, цена исправления ошибки вырастает примерно на порядок при переходе между рабочими потоками (от анализа требований к проектированию, от проектирования к реализации и т.д.).

Тем самым, если не заложить средства на проверку требований на предмет двусмысленности в момент их формирования – существует риск непринятия готовой системы в момент приёмо-сдаточных испытаний, т.к. каждая из сторон будет придерживаться своей версией интерпретации требований, что ведёт к убыткам, судебным процессам и т.п. и тому есть масса примеров.

«Золочение» продукта

Под «золочением» понимают такие ситуации, когда разработчики добавляют функции, которых нет в спецификации, но им кажется, что это понравится пользователям. Зачастую же клиентам не нужны такие избыточные возможности, получается, что время, отведенное на реализацию, тратится впустую (конец цитаты).

Эта ситуация возникает в случае, когда, во-первых, в коллективе Разработчика присутствуют творческие личности (ведь далеко не всякая команда станет проявлять инициативу и делать сверх того, о чём её просили), во вторых – существует разрыв в прохождении информации от Заказчика к Разработчику. Инициативный разработчик «золотит» продукт из самых лучших побуждений, но, возможно, он плохо знаком с бизнес-процессом Заказчика и заложенные им «фичи» попросту не будут востребованы.

Другая сторона **«золочения»** заключается в том, что группа представителей Заказчика неоднородна по своей структуре и может возникнуть ситуация, когда представитель Заказчика, формулирующий «дорогие» требования, не обладает соответствующими полномочиями. Это – специфика российских предприятий, где часто всё бывает устроено существенно неформально.

Автору пришлось однажды присутствовать в одном очень показательном диалоге, где он участвовал в качестве Разработчика, сдающего продукт (АИС для склада материалов), ещё присутствовали пользователь новой системы и инвестор проекта.

> **Ирина (пользователь):** *мне неудобно работать в этой программе. Она не учитывает размеры баночек в литрах.*
> **Павел (инвестор):** *без этого спокойно можно обойтись, достаточно того, что есть учёт материалов по классификатору и 2 единицы измерения – в литрах и килограммах.*
> **Ирина:** *а мне этого недостаточно, я веду учёт в баночках!*
> **Павел:** *а я тут хозяин, здесь всё моё и мне твой баночный учёт не нужен!*
> **Ирина:** *а я всё равно буду работать, как я привыкла! Диалог длился около получаса и последнее слово осталось, конечно, за Павлом, программу переписывать не пришлось.*

Минимальная спецификация

Создавать полную документацию требований в соответствии с вышеизложенными принципами, или ограничиться наброском требований на 2-3 страницы, как это зачастую делает автор лекций в небольших проектах – как говориться, дело вкуса.

Однако, для работы «не по правилам», во-первых, должны быть объективные предпосылки, во-вторых – следует отдавать себе отчёт в выгодах и рисках этого выбора.

Минимальная спецификация уместна, если имеет место наличие одновременно трёх обстоятельств (объединение по «И»):

а) цена контракта и размеры проекта таковы, что разработка развёрнутого ТЗ экономически нецелесообразна;

б) коллектив Разработчика обладает достаточной степенью профессионализма и опыта выполнения проектов в смежных областях, чтобы уметь создавать по краткой спецификации продукт, который пройдёт приёмку Заказчиком;

в) между Заказчиком и Разработчиком существуют конструктивные отношения и обе стороны понимают и принимают риски мини-спецификации.

Другой вариант работы по мини-спецификации: Заказчик и Разработчик понимают, что создание развёрнутой спецификации оттягивает окончание выпуска готового продукта, что главная цель проекта – продукт, а не документация и готовы к плотному сотрудничеству в процессе его создания. Это – путь так называемых agile-методологий разработки ПО.

Основной риск применения мини-спецификации заключаются в том, что они базируются на человеческом факторе. Хорошие и конструктивные отношения между сторонами «на берегу» должны сохраниться на всём протяжении проекта, в противном случае у сторон возникнут существенные проблемы в формальном доказательстве того – что должна делать программа, т.к. мини-спецификация для этого недостаточно полна.

Пропуск типов пользователей

Корпоративные АИС создаются для того, чтобы быть использованными различными группами пользователей. Может сложиться ситуация, в которой в группу представителей Заказчика, участвующих в формировании требований, попадут наиболее инициативные персоны предприятия, которые, по всей видимости, смогут донести свой голос до представителей Разработчика. Те же категории пользователей, у которых не найдётся активных представителей, могут оказаться «за бортом» автоматизации. Именно эта ошибка формирования требований называется «пропуск классов пользователей». Чтобы её избежать, представитель Разработчика должен объективно оценить организационную структуру предприятия и его бизнес-процессы и вдумчиво подойти к выбору ключевых персон, проведение интервью с которыми поможет сформировать целостную картину требований к создаваемой АИС.

Методы и средства проверки требований

Наработано значительное количество методов и средств проверки требований. Они разнятся по ряду параметров. Так, различают:

- по широте анализа – просмотр (выборочная проверка) и сквозной контроль (тотальная проверка);
- по степени формализации – неофициальные процедуры, процедуры, проводимые по формальным правилам (инспекции, экспертизы);
- по составу группы проверки – с (без) участием автора, с (без) участием менеджера проекта, с (без) участием представителей внешних организаций;
- по используемым средствам – тексты требований, тестовые сценарии, критерии приемлемости, прототипы.

1.2. Характеристики хороших требований

Характеристики качества превосходных требований:

- **Полнота.** Как известно из теории искусственного интеллекта, неполнота – одно из фундаментальных свойств человеческого знания. При создании программных систем нам

приходится иметь дело с характеристиками ещё несуществующей системы. Идея о том, что необходимо сформулировать все требования полностью, т.е. исчерпывающим образом, до начала проектирования, а тем более – реализации системы, изжила себя вместе с так называемым каскадным подходом, который поддерживал последовательную модель реализации системы. Спиральный подход, на котором базируется большинство современных методологий, предусматривает поэтапное выделение и детализацию требований на всём протяжении цикла разработки системы.

Тем не менее, требование полноты предъявляется к требованиям, формулируемым к системе. Надо понимать, что данное требование – это скорее тенденция, цель, к которой нужно постараться максимально приблизиться на как можно более ранних стадиях проекта.

Требование полноты можно рассматривать в двух аспектах: полнота отдельного требования и полнота системы требований.

Полнота отдельного требования – свойство, означающее, что текст требования не требует дополнительной детализации, то есть в нём предусмотрены все необходимые нюансы, особенности и детали данного требования.

Полнота системы требований – свойство, означающее, что совокупность артефактов, описывающих требования, исчерпывающим образом описывает всё то, что требуется от разрабатываемой системы.

- **Ясность (недвусмысленность, определённость, однозначность спецификаций).** Каждый из совладельцев разрабатываемой системы обладает своим личным опытом восприятия событий внешнего мира. Слово, произнесённое вслух, вызывает индивидуальные ассоциации в семантическом пространстве каждого отдельного воспринимающего субъекта. То, что является ясным, допустим, для кардиохирурга, совсем необязательно будет таковым для специалиста в области программной инженерии.

Соответственно, требование обладает свойством ясности, если оно сходным образом воспринимается всеми совладельцами системы. На практике ясность требований достигается в том числе и в процессе консультаций, в ходе которых происходит «выравнивание тезаурусов» совладельцев системы. Хорошим подспорьем в этом служит согласованный сторонами глоссарий ключевых понятий предметной области.

К.Вигерс даёт следующий совет по повышению ясности документов: «Пишите документацию просто, кратко и точно, применяя лексику, понятную пользователям».

Ещё одной стороной понятия «ясность требования» является его. Требование, которое сформулировано ясно, может быть прослежено, начиная от того документа, где оно сформулировано впервые, вплоть до рабочих спецификаций.

- **Корректность и согласованность (непротиворечивость).** Корректность – одно из важнейших свойств требований. К. Вигерс в вводит понятие корректности требования через точность описания функциональности. В этом смысле корректность в определённой степени конкурирует с полнотой. Но есть и различие: если свойство полноты носит скорее качественный характер: абсолютная полнота представляет недостижимый идеал, к которому можно приближаться, то свойство корректности носит оценочный характер и задаёт дихотомию: каждое из требований

либо корректно, либо нет. Кроме того, можно рассуждать о взаимной корректности требований или согласованности (непротиворечивости): если два требования вступают в конфликт, значит – как минимум одно из них некорректно. В иерархии требований можно выделить вертикальную и горизонтальную согласованность. Иными словами, требования не должны противоречить, соответственно, требованиям своего уровня иерархии и требованиям «родительского» уровня. Так, требования пользователей не должны противоречить бизнес-требованиям, а функциональные требования – требованиям пользователя.

- **Верифицируемость (пригодность к проверке).** Признаки (свойства) требований, рассматриваемые в настоящей лекции, нельзя считать независимыми. В математической статистике такие признаки называются кореллируемыми. Так, свойство верифицируемости существенно связано со свойствами ясности и полноты: если требование изложено на языке, понятном и одинаково воспринимаемым участниками процесса создания информационной системы, причём оно является полным, т.е. ни одна из важных для реализации деталей не упущена – значит, это требование можно проверить. При этом в ходе проверки у сторон (принимающей и сдающей работу) не должно возникнуть неразрешимых противоречий в оценках. Так как хорошо сформулированные требования составляют основу успешного создания системы – роль верифицируемости трудно переоценить. Требования к системе представляют основу контракта между Заказчиком и Исполнителем и если данные требования нельзя проверить – значит и контракт не имеет никакого смысла, следовательно, успех или неудача проекта будут зависеть только от эмоциональных оценок сторон и их способности договориться, а это – слишком шаткая основа для осуществления работ.

- **Необходимость и полезность при эксплуатации.** Одни из самых субъективных и трудно проверяемых свойств требований.

Наиболее бесспорными требованиями следует считать бизнес-требования. Данные требования формулируют первые лица, представляющие Заказчика, и вряд ли кто-нибудь лучше них сможет сказать, каким условиям должна соответствовать создаваемая информационная система, чтобы соответствовать бизнес-целям предприятия. Тем не менее, если у представителя Исполнителя возникают сомнения в необходимости того или иного бизнес-требования, вызванные интуитивными соображениями, либо опытом внедрения информационных систем на аналогичных предприятиях, он должен проявить инициативу и собрать совместное совещание сторон. Аргументы в пользу отсутствия необходимости требования несомненно будут восприняты, особенно если они будут мотивированы в бизнес-терминологии Заказчика и подтверждены выкладками, прогнозирующими соотношение затрат на выполнение требования и ожидаемой от него эффективности.

Необходимость требований пользователя может вытекать из соответствующих бизнес-требований. Кроме того, требования пользователя могут мотивироваться эргономичностью продукта и особенностями функционирования его отдела (подразделения), недостаточно полно раскрытыми на предыдущем уровне иерархии требований.

Большинство функциональных требований вытекают из требований первых двух уровней. Другие функциональные требования могут лежать вне сферы компетенции Заказчика (который, вообще говоря, не обязан быть экспертом в области IT) и их должен сформулировать Исполнитель. Так, например, информационная система в процессе её использования может начать снижать свою

производительность из-за больших объёмов накапливаемых данных. Поэтому целесообразно заложить функции архивирования информации, переключения учётных периодов и т.п., необходимость которых следует не из особенностей бизнеса предприятия внедрения, а из общих принципов построения информационных систем.

Более слабой, чем «необходимость» формулировкой обладает свойство «полезность при эксплуатации». Разграничение между данными свойствами можно провести следующим образом. Необходимыми следует считать свойства, без выполнения которых невозможно, либо затруднено выполнение автоматизированных бизнес-функций пользователей; полезными при эксплуатации следует считать любые свойства, повышающие эргономические качества продукта.

- Осуществимость (выполнимость). Является в некоторой степени конкурирующим по введённым выше двум свойствам.

В принципе никто не мешает сформулировать требование, выполнимость которого ограничивается сегодняшним уровнем развития техники и технологии, хотя многое из того, что было невыполнимо десять лет назад, вполне выполнимо сегодня. Можно сформулировать требование, выполнимость которого ограничена научными представлениями о строении Вселенной, например – требование мгновенной передачи информации с земной станции на Марс, хотя и фундаментальные представления иногда меняются, пусть и не так быстро, как развитие IT-технологий.

Возвращаясь с небес на землю, отринем те требования, которые можно признать абсурдными и остановимся на тех, которые выполнимы принципиально. С точки зрения науки управления требованиями, далеко не все из них являются осуществимыми.

Выполнимость требования на практике определяется разумным балансом между ценностью (степенью необходимости и полезности) и потребными ресурсами. Так, если стоимость контракта на разработку информационной системы составляет $10000, а затраты на выполнение нового требования, возникшее в момент, когда проект выполнен наполовину, оценивается в $4000, является ли оно невыполнимым? Скорее всего, да, если Исполнитель докажет Заказчику новизну требования (требование не входило в согласованные спецификации) и сложность его исполнения. Но, если требование является критически важным, необходимым, но выпало из поля зрения при подписании контракта Заказчик готов выделить дополнительно финансирование, а Исполнитель – трудовые ресурсы – значит, требование выполнимо. Таким образом, требование осуществимости в ряде случаев также следует считать субъективным, а критерии его оценки лежат в области договорённостей между Заказчиком и Исполнителем.

Отличной иллюстрацией балансировки между ценностью и выполнимостью требований является так называемый треугольник компромиссов.

В качестве пояснения к рисунку приведём цитату из «белых страниц», размещённых Microsoft в открытом доступе. Хорошо известна взаимозависимость между ресурсами проекта (людскими и финансовыми), его календарным графиком (временем) и реализуемыми возможностями (рамками). Эти три переменные образуют треугольник. После достижения равновесия в этом треугольнике изменение на любой из его сторон для поддержания баланса требует модификаций на другой (двух других) сторонах и/или на изначально измененной стороне.

- **Модифицируемость.** Необходимо обеспечить возможность переработки требований, если понадобится, и поддерживать историю изменений для каждого положения. Для этого все они должны быть уникально помечены и обозначены, чтобы вы могли ссылаться на них однозначно. Каждое требование должно быть записано в спецификации только единожды. Иначе вы легко получите несогласованность, изменив только одно положение из двух одинаковых. Лучше используйте ссылки на первоначальные утверждения, а не дублируйте положения. Модификация спецификации станет гораздо легче, если вы составите содержание документа и указатель. Сохранение спецификации в базе данных коммерческого инструмента управления требованиями сделает их пригодными для повторного использования (конец цитаты).

- **Трассируемость.** Трассируемость требования определяется возможностью отследить связь между ним и другими артефактами информационной системы (документами, моделями, текстами программ и пр.). Отдельная траса представляет собой направленное бинарное отношение, заданное на множестве артефактов ИС, где первый элемент отношения представляет соответствующее требование, а второй – артефакт, зависимый от данного требования. На практике трассировки анализируются при посредстве графовых, либо табличных моделей.

Процесс трассировки позволяет, с одной стороны, выявить уже на стадии проектирования системы проектные артефакты, к которым не ведёт связь ни от одного из артефактов, описывающих требования, с другой – артефакты, описывающие требования, не связанные с проектными артефактами. В первом из случаев целесообразно убедиться в том, что проектный артефакт действительно имеет право на существование, а не является избыточным. Во втором случае необходимо проанализировать полезность выявленных требований: либо эти требования несут недостаточную полезную нагрузку и могут быть игнорированы, либо имеют место ошибки проектирования: пропущены соответствующие артефакты.

Другая цель трассировки – повысить управляемость проектом: при изменении отдельно взятого требования становится понятно – какие из проектных, рабочих и других артефактов подлежат изменению.

- **Упорядоченность по важности и стабильности.** Приоритет требования представляет собой количественную оценку степени значимости (важности) требования. Приоритеты требований обычно назначает представитель Заказчика. Разработчик, отталкиваясь от приоритетности требований, управляет процессом реализации информационной системы. Стабильность требования характеризует прогнозную оценку неизменности требований во времени.

- **Наличие количественной метрики.** Количественные метрики играют важную роль в верификации и аттестации информационных систем. В первую очередь это относится к нефункциональным требованиям, которые, как правило, должны иметь под собой количественную основу (запрос должен отрабатываться не более, чем ___ секунд; средняя наработка на отказ должна составлять не менее, чем ___ часов). Функциональные требования также могут расширяться количественными мерами при помощи так называемых аспектов применимости.

1.3. CheckList для проверки требований

Контрольный список требований — это инструмент для оказания помощи в определении задокументированы корректны, полны, не двусмысленны, последовательны, проверены и утверждены ли требования. Шаблон спецификации требований включает в себя примеры и может использоваться для документирования требований для вашего продукта или услуги, в том числе для приоритезации и утверждения. Адаптируйте спецификацию в соответствии с вашим проектом, организуйте соответствующие разделы таким образом, чтобы это работало лучше всего, и используйте этот контрольный список для записи решений о применимых областях. Каждый указанный ниже пункт, в зависимости от обстоятельств, должен быть задокументирован и включен в требования по рассмотрению пакета требований.

Применимо? (Да? Нет? Комментарий?)	**Требования, подлежащие сдачи**	**Основные вопросы и проблемы, которые необходимо учитывать**
		Описание Проекта
Да — Обязательное	Обзор проекта	Включает ли пакет имя проекта и все требования обзора пакета вкладчиков, а также имя рабочей группы (групп), которые будут владеть требованиями? Это может быть просто ссылка на Устав проекта.
Да — Обязательное	Основные заинтересованные стороны	Включает ли пакет список ключевых заинтересованных сторон с их рабочей группой и адресами электронной почты. Это может быть просто ссылка на Устав проекта.
Да — Обязательное	Рамки (границы) и бизнес-причина	Включает ли пакет краткое описание бизнес причины для этого проекта, а также что входит и что не входит в границы этого проекта. Определены ли целевая аудитория или основной заказчик? Это может быть просто ссылка на Устав проекта.
	Характеристики пользователя	Определены ли общие характеристики или профили предполагаемых пользователей?

	Предположения	Задокументированы ли допущения, которые влияют на требования.
	Препятствия и ограничения	Задокументированы ли технические, финансовые или деловые ограничения, которые могли бы ограничить варианты проектирования?
	Зависимости	Зависят ли требования от версий выпуска или функциональности других приложений/услуг? От других организационных изменений или нехватки ресурсов?
	Общий язык	Все ли сокращенные наименования или специализированные термины были включены в глоссарий или словарь данных?
		Требования к документации
	Функциональные требования	• Определены ли бизнес-правила? • Описаны ли действия по обработке входных и выходных данных? • Описана ли каждая функция, которая поддерживает входы или выходы? • Определена ли проверка достоверности данных на входах? • Описана ли точная последовательность операций? • Нужны ли конкретные реакции при возникновении нестандартных (нештатных) ситуаций? (например, переполнение, средства связи, ошибки обработки/восстановления) • Как влияют параметры? • Описаны ли взаимоотношения выходов к входам? (например, последовательности ввода/вывода, формулы для преобразования входов в выходы) • Описаны ли необходимые пользовательские интерфейсы? (например, формы экрана или организации, макеты отчетов, структура меню, ошибок и прочих сообщений, или функциональных клавиш) • Описаны ли явно нежелательные события/входы, наряду с необходимыми мерами реагирования?
	Производительность	• Выявлены ли статические и динамические численные требования по производительности? • Все ли требования по производительности измеримы? • Выявлены ли явные требования по задержке? • Измеримы ли требования по емкости? • Определены ли конкретные и измеримые требования в отношении доступности? • Определены ли конкретные и измеримые требования в отношении надежности?
	Управляемость и ремонтопригодность	• Существуют ли особые требования к управлению поставляемого продукта или услуги? • Существуют ли требования для медицинского мониторинга для продукта или услуги, условий отказа, обнаружения ошибок, регистрации и коррекции? • Существуют ли требования, в частности, связанные с

		простотой обслуживания? • Указаны ли нормальные и специальные операции?
	Удрбство использования	Определены ли юзабилити требования (требования по удобству использования)?
	Интерфейсы (Системы, сети, аппаратные средства) и интеграции	• Описан ли каждый требуемый интерфейс с другим продуктом или системой? • Описан ли каждый требуемый интерфейс с сетевым компонентом? • Описан ли каждый требуемый интерфейс с аппаратной составляющей или оборудованием? • Все ли входные, выходные и системные условия и их взаимодействия описаны? • Есть ли в документации ссылки на существующие интерфейсы? • Есть ли необходимость для требований, которые являются специфическими для данного сайта, таких как судов океанографии?
	Управление данными	Указаны ли требования к данным?
	Соответствие стандартам	Описаны ли требования, которые основаны на существующих стандартах, политиках, правилах или законах?
	Безопасность	Описаны ли требования к безопасности, в том числе авторизации и факторам аутентификации?
	Портативность	Должна ли система быть легко переноситься на другие хост-машины и/или операционные системы? Является ли экологическая независимость требованием?
	Существующие дефекты, которые необходимо решить	Существуют ли дефекты, которые должны быть решены с этим выпуском (версией)? Задокументированы ли они?
		Требования к процессам
	Отслеживаемость (трассировка)	Все требования пронумерованы или однозначно идентифицированы?
	Приоритетность	Определены ли приоритеты для каждого требования?
	Требования к утверждению	Все ли требования были одобрены и подтверждены спонсором?
	Краткость	Каждое ли требование является однозначным, только с одной интерпретацией?
	Непротиворечивость	Являются ли требования взаимно согласованными? Конфликтует или дублирует какое-нибудь требование другие требования?
	Полнота	• Каждре ли требование является правильным и полным? • Задокументированные требования фиксируют все высказанные потребности клиента? • Есть ли неустановленные потребности клиента, которые будут вызывать недовольство, если они не будут выполнены? • Есть ли неустановленные потребности клиента,

		которые, если встретятся, позволят сделать клиента более удовлетворенным? • Задокументированы ли все выявленные требования? • Конфликтуют ли какие-либо требования? • Есть ли особые соображения, не охваченные в требованиях?
	Бизнес-Сценарии и Варианты Использования	Были ли построены бизнес-сценарии для иллюстрации (или извлечения) требований?
	Передача рецензенту	Все ли рецензенты требований подтвердили готовность участвовать в рассмотрении требований? Существует ли план для обеспечения непрерывности обзора группой экспертов?
	Удаленные или отложенные требования	Были ли какие-то требования утверждены, но впоследствии были удалены? Были ли какие-либо требования известны, которые были отложены до будущих версий продукта? Идентифицированы ли эти требования?
	Тестирование и тестопригодность	• Каждое ли требование проверено? Отмечены ли непроверяемые (и не тестируемые) требования? • Могут ли требования служить основой для определения окончательной приемки продукта? • Какие типы тестирования и методологий тестирования предложены?
	Ясность	• Описаны ли требования в достаточной степени для разработки системы командой реализации, удовлетворяющей требованиям, а также для того, чтобы убедиться, что система удовлетворяет требованиям? • Являются ли спецификации требований понятными и читаемыми?
	Уместность/ пригодность	• Указывают ли требования, что должно быть сделано, а не описывают, как должны быть реализованы продукт или услуга? • Избегают ли требования указания конкретного дизайна?
	Планирование	• Существует ли план, чтобы решить каждую пометку «Будет определено позднее» в требованиях? • Существует ли план, чтобы проследить каждое требование к реализующемуся элементу (например, диаграмма дизайна или тестового примера)? • Используемы ли требования для повторного использования после внедрения? • Совместимы ли требования с более поздними фазами проекта? • Могут ли использоваться спецификации требований в качестве основы для выполнения проекта?
	Управление требованиями	• Хранятся ли документы с требованиями в соответствии с изменениями продукции? • Является ли управление изменениями системой для отслеживания изменений по требованиям? Соответствуют ли требования конфигурации? • Структурированы ли документы с требованиями в соответствии с изменениями?

		• Были ли документы с требованиями разработаны в соответствии с процессами разработки документации, которые были согласованы с проектной командой? • Облегчают ли документы с требованиями сбор данных о процессе управления требованиями?

1.4. Прототипирование

В общем случае, **прототипирование** подразумевает проверку инженерной интерпретации программных требований и извлечение новых требований, неопределенных или неясных на ранних итерациях сбора требований. Существует множество подходов к прототипированию, как с точки зрения детализации, так и того, чему уделяется внимание при прототипировании. Наиболее часто прототипы создаются для оценки способа реализации пользовательского интерфейса и проверки архитектурных подходов и решений.

При всей безусловной полезности прототипирования для обеспечения проверки требований и решений, необходимо понимать, что с прототипированием связан ряд вопросов способных привести к негативным последствиям или, как минимум, работам, требующим дополнительного времени и средств. Среди возможных негативных последствий прототипирования стоит выделить следующие:

- Смещение внимания с целевых функций прототипа и, как следствие, неудовлетворенность пользователей огрехами прототипа, отсутствием стоящей за ним реальной функциональности (для прототипов пользовательского интерфейса), ошибками в прототипе и т.п.
- Превращение прототипа в реальную систему за счет постоянного добавления новых свойств и функциональности "для проверки" – часто бывает нарушена архитектурная целостность, не обеспечена необходимая масштабируемость и качество получаемого программного продукта;

Здесь хотелось бы добавить и еще одну типичную проблему - переключение внимания заинтересованных лиц на эргономику и детали дизайна графического пользовательского интерфейса, при начальной цели построения прототипа для выявления функциональных и иных требований и наоборот. Проблема не во внимании пользовательскому интерфейсу, проблема в подмене, если так можно выразиться, функциональной составляющей пользовательским интерфейсом (вспомните, как часто вы сами говорили или слышали – "я не о 'кнопочках' и 'окошках', я о задаче ...").

Конечно, ясно, что эти факторы можно превратить и в положительные стороны прототипа. Кроме того, не стоит считать что прототип это всегда нечто, воплощенное в код. Прототипом пользовательского интерфейса может быть, например, просто "прорисованный" на бумаге или в электронной форме набор переходов между экранами/диалоговыми окнами системы (кстати, это подход, часто используемый в Agile-практиках, но отнюдь не заменяющий требований к системе).

Так или иначе, выбор того или иного метода прототипирования, да и самого факта такого способа проверки требований или технологических идей, должен основываться на временных и других имеющихся ресурсах, опыте в прототипировании и, конечно, степени сложности создаваемой программной системы.

1.5. Трассировка

Трассировка требований (tracing) - это установка связей требований с другими требованиями и проектными данными

Цель трассировки требований:

- Убедиться, что все требования к системе выполнены в процессе реализации
- Убедиться, что приложение делает только то, что предполагалось
- Облегчить управление изменениями

Одной из главных проблем сбора требований является проблема их изменения. Требования создаются итерационно путем постоянного общения представителей заказчиков с аналитиками и разработчиками будущей системы в целях выявления потребностей. Требования изменяются в зависимости от задач и условий их определения, а также постоянного уточнения на этапе заключения договора на создание системы. На момент заключения договора состав требований, их виды и свойства становятся более полными, т.е. соответствуют взглядам заказчика на создаваемую систему.

Одним из инструментов установления зависимости между сформулированными требованиями и их изменениями является трассировка, т.е. поддерживается развитие и обработка требований с прослеживанием идентифицированных связей, которые должны быть зафиксированы по двум направлениям - от источника требований к реализации и, наоборот. Выявляются причины появления разнообразных неточностей, добавлений и определяется необходимость внесения изменений в требования в одном из приведенных направлений.

Типы трассируемости требований:

Если после разработки некоторого рабочего продукта возникает потребность в изменении отдельных требований или необходимость проследить за происхождением внесенных требований в одном из направлений данной схемы трассирования, то уточняются связи между отдельными требованиями и элементами рабочих продуктов В случае трассирования требований от продукта, движение в обратном направлении, т.е. - к требованиям, можно выяснить, как написана каждая строка этого продукта и соответствует ли она отдельным атрибутам требований. Связи трассируемости требований помогают найти незапланированные и реализованные некоторые функции или фрагменты программ, не соответствующие заданным требованиям. И, наоборот, выявить нереализованные требования к функциональности. Взаимосвязи и зависимости между отдельными требованиями сохраняются, например, в таблице трассируемости, удаляются или модифицируются при различных изменениях.

Методы трассировки базируются на формальных спецификациях связей между элементами требований либо ограничиваются описаниями функций, ситуаций, контекста и возможных решений. **Основу трассировки составляют:**

- требования, которые изменяются при их формировании;
- некоторые детали выполнения функций в рабочем продукте системы, которые не предусматривались, но появились в связи с возникшей практической ситуацией;
- связи между различными моделями процесса проектирования системы на ЖЦ программного продукта и принятые решения о необходимости изменения требований в связи с появившимися недостатками в промежуточном продукте;
- информация о согласованных атрибутах требований на разных уровнях рассмотренной схемы трассирования и сохранение ее матрице трассирования;
- специальные системные требования, касающиеся повторного использования готовых компонентов или частей системы;
- результаты тестирования, по которым можно определить наиболее вероятные части кода, требующие проверки на наличие в них дефектов.

В матрице требований в строках указываются пользовательские требования, а столбцах - функциональные требования, элемент проектирования, вариант версии и др. В этих столбцах заполняются данные о степени выполнимости системных требований на каждом элементе создаваемого продукта. Механизм ссылок в таблице позволяет проверять связанные с каждым элементом продукта диаграммы вариантов использования, потоки данных, классы и др.

Процедура трассирования состоит в следующем:

- выбирается элемент (функция, фрагмент или некоторая часть) из матрицы трассирования требований, за которым проводится прослеживание на этапах ЖЦ;
- составляется список вопросов, по которым на каждом этапе проверяются связи при реализации требований в продукте, и если изменяется какое-то звено в цепочке требований, то может модифицироваться процедура разработки этого элемента на последующем этапе ЖЦ;
- проводится мониторинг статуса каждого требования на соответствие выполнения согласно принятому плану;
- уточнение ресурсов выполнения проекта при необходимости проведения изменений в требования и в элементы проекта.

Условием принятия решения о возможных модификациях требований и результатов промежуточного проектирования, является обновленная информация о связях между различными частями системы и первоначально заданными требованиями к ним.

Трассировка обеспечивает:

- ввод более сложных отношений вместо простых связей или специфических отношений;
- использование разных путей трассировки (между моделями или иерархическими связями);
- ведение базы данных объектов трассировки и отношений между ними.

Трассировка может быть выборочной для отдельных объектов или связанной с другими объектами, а также с возможными переходами от одной модели проектирования к другой путем проверки трансформации одних объектов в другие.

Документы трассировки требований:

- **Traceability Matrix** - Матрица трассировки. Показывает связь двух заданных типов требований.
- **Traceability Tree (Traced out of...)** - Дерево трассировки (трассировка от). Древовидное представление зависимости требований всех типов требований одного заданного типа.
- **Traceability Tree (Traced in to ...)** - Дерево трассировки (трассировка к). Древовидное представление зависимости требований одного заданного типа от всех типов требований.

Матрица трассировки — это таблица, в которой первая колонка содержит автоматизируемые функции бизнес процесса, а вторая колонка — соответствующие требования к функциям. На последующих этапах разработки АС к матрице трассировки справа пристраиваются две дополнительные колонки: функции системы и типовое проектное решение по реализации функций. Каждая строка этой таблицы и есть «трассировка функция — требования — . . .».

Матрица трассировки (отслеживания) требований представляет собой таблицу, которая связывает требования с их происхождением и отслеживает их на протяжении жизненного цикла проекта.

Применение матрицы трассировки (отслеживания) требований помогает удостовериться, что каждое требование увеличивает ценность бизнеса, связывая его с целями бизнеса и проекта. Это позволяет отслеживать требования на протяжении жизненного цикла проекта, что помогает удостовериться в том, что требования, одобренные в документах по требованиям, выполнены в конце проекта. Наконец, матрица отслеживания требований обеспечивает структуру для управления изменениями содержания продукта.

1.6. Документы и Типы Требований

Тип Документа	Описание	Тип Требования по Умолчанию
Stakeholder Requests (STR – Запросы Заинтересованных Лиц)	Ключевые запросы заинтересованных лиц	Stakeholder Request (STRQ – Запрос Заинтересованного Лица)
Vision (VIS - Концепция)	Полное описание системы и специфичные требования	Feature (FEAT – Функциональная Особенность)
Use Case Specification (UCS – Спецификация Сценариев Использования)	Описание Сценариев Использования	Use Case (UC - Сценарий Использования)
Glossary (GLS - Справочник)	Используется для хранения основных терминов	Glossary Item (TERM – Термин Справочника)
Supplementary Specification (SS – Дополнительная Спецификация)	Нефункциональные требования	Supplementary Requirements (SUPL- Дополнительные Требования)
Requirements Management Plan (RMP – План Управления Требованиями)	Данный документ	Нет требований

2. Содержание Процесса Управления Требованиями

Самое простое описание процесса управления требованиями содержит следующие основные пункты:

- Формирование плана управления требованиями.
- Сбор требований.
- Разработка документа Концепции (Vision).
- Создание сценариев использования (Use Cases).
- Дополнительная спецификация.
- Создание тестовых сценариев (Test Cases) из сценариев использования (Use Cases).
- Создание тестовых сценариев (Test Cases) из дополнительной спецификации.
- Проектирование системы.

Первый шаг (план управления требованиями) определяет пирамиду требований. На каждом из последующих семи шагов строится один элемент пирамиды.

Шаг	Тип Требований	Документы
Сбор требований	Потребности заинтересованного лица	Запросы заинтересованного лица
Разработка документа Концепции (Vision)	Функциональные особенности	Концепция
Создание сценариев использования (Use cases)	Сценарии использования, алгоритмы	Спецификация Сценариев Использования
Дополнительная спецификация	Дополнительные требования	Дополнительная спецификация
Создание тестовых сценариев (test cases) из сценариев использования (use cases)	Тестовые сценарии	Тестовые сценарии
Создание тестовых сценариев (test cases) из дополнительной спецификации	Тестовые сценарии	Тестовые сценарии
Проектирование системы	Диаграммы классов, диаграммы взаимодействий	UML-диаграммы

3. Формирование Плана Управления Требованиями

Одна из самых первых задач в проекте – это разработка Плана Управления Требованиями (Requirements Management Plan – RMP). RMP содержит в себе общие подходы к управлению требованиями в проекте. Документ детально описывает, каким образом создаются требования, как они упорядочиваются, изменяются и отслеживаются в течение жизненного цикла проекта.

Несколько вопросов, на которые может ответить документ RMP:

- Будет использоваться инструмент для управления требованиями?
- Какие типы требований будут присутствовать в проекте?
- Каковы атрибуты этих требований?

- Где будут создаваться требования – в базе данных или в документах?
- Между какими требованиями должна осуществляться трассировка?
- Какие документы необходимы?
- Какие требования и документы будут использоваться как контракт с заказчиком?
- Если часть проекта разрабатывается сторонними исполнителями, какие требования и документы будут использоваться как контракт со сторонними разработчиками?
- Нужно следовать RUP или какой-либо другой методологии?
- Нужны заказчику особые документы для осуществления разработки?
- Как будет осуществляться управление изменениями?
- Какой процесс будет гарантировать, что все требования будут выполнены и протестированы?
- Какие требования или представления необходимы для генерации отчетов?

4. Создание Сценариев Использования (Use Cases)

Наилучшая форма описания функциональных требований – это **сценарии использования (use cases).** Они извлекаются из функциональных особенностей.

Сценарий использования – это описание системы в терминах последовательности действий. Он должен иметь значимый результат или определенное значение для действующего лица (действующее лицо – это некто или нечто, взаимодействующее с системой).

Сценарии использования:

- Инициируются действующим лицом.
- Являются моделью взаимодействий между действующим лицом и системой.
- Описывают последовательность действий.
- Содержат в себе функциональные требования.
- Должны иметь некоторое значение для действующего лица.
- Представляют полный и имеющий смысл сценарий событий.

Назначение сценариев использования - способствовать соглашению между разработчиками, заказчиками и пользователями на тему того, что именно система должна делать. Сценарий использования становится своего рода контрактом между разработчиками и заказчиками. Он также является основой для реализации сценариев использования, которые играют большую роль в проектировании системы.

5. Дополнительная Спецификация

Дополнительная спецификация содержит нефункциональные требования (удобство использования, надежность, производительность, способность к сопровождению), а также некоторые функциональные требования, распространяющиеся за пределы системы, и вследствие этого, сложные для отображения в сценариях использования. Эти требования называют дополнительными требованиями. Они извлекаются из функциональных особенностей.

6. Создание Тестовых Сценариев (Test Cases) из Сценариев Использования

Как только все требования собраны, следует разработать способ проверить, правильно ли они реализованы в конечном продукте. Тестовые сценарии (test cases) показывают тестерам, какие шаги должны быть сделаны для того, чтобы протестировать все требования. На этом шаге следует концентрироваться на создании тестовых сценариев из сценариев использования.

7. Формирование Плана Управления Требованиями

План Управления Требованиями (Requirements Management Plan - RMP).

RMP содержит в себе общие подходы к управлению требованиями в проекте. Он описывает, каким образом создаются требования, как они упорядочиваются, изменяются и отслеживаются в течение жизненного цикла проекта. Он также описывает все используемые в проекте типы требований и их атрибуты.

8. Сбор требований

В каждой группе заинтересованных лиц выделяется, по крайней мере, один представитель, который будет отвечать за поступление требований. Этот человек должен быть уполномочен представлять группу, обладать соответствующими знаниями и быть доступным для связи с группой аналитиков.

Запросы заинтересованных лиц могут быть собраны с использованием различных методов:

- **Интервью** (индивидуальный диалог с заинтересованным лицом).
- **Анкеты** (набор вопросов, отправленный большому количеству заинтересованных лиц).
- **Семинары** (заинтересованные лица собираются на определенный период для интенсивных, насыщенных дискуссий).
- **Сценарии приложения** (использование визуальных/графических инструментов для демонстрирования поведения системы).
- **Ролевые игры** (каждому члену группы назначается определенная роль, обычно роль одного из пользователей).
- **Сессии с применением метода «мозгового штурма»** (Brainstorming sessions - групповой метод решения вопросов путем изложения идей в течение непродолжительных, интенсивных сессий).
- **Использование прототипов** (разработка прототипов для получения отклика о системе).
- **Сценарии использования** (Use Cases – взаимодействие между пользователем и системой, представленное в виде последовательности шагов).
- **Анализ существующих документов** (извлечение информации из документов Microsoft Word, электронной почты и записей).
- **Наблюдение и демонстрирование задач** (наблюдение за пользователями, выполняющими определенную задачу).
- **Анализ существующих систем** (сбор требований от морально устаревших заменяемых систем или от систем, разработанных в ходе конкуренции).

9. Документ Запросов Заинтересованного лица

Вся полученная в процессе сбора требований информация должна быть документально оформлена в документ **Запросов Заинтересованного Лица (Stakeholder Requests)**.

Можно создать один документ, который будет содержать в себе запросы всех заинтересованных лиц, один документ для каждого заинтересованного лица, либо можно собрать запросы нескольких заинтересованных лиц в одном документе.

Документ Запросов Заинтересованного Лица не содержит специально выделенного места для размещения требований типа STRQ (Stakeholder Requests или Запросы Заинтересованного лица). Требования могут быть вложены в качестве ответов на вопросы интервью, либо собраны в последнем параграфе, названным «Резюме Аналитика».

Главными атрибутами требования являются Text (Текст, обязательный атрибут) и Name (Название, не обязательный). Если Text – короткий (описание состоит из одного предложения), то Вам не надо создавать отдельное Name. Если описание Text длинное, тогда стоит создать краткое название для требования. Если указано Name, оно используется для идентификации требования. Если Name не указано, для идентификации используется Text. Для большей точности лучше использовать название для всех требований одного типа, либо не использовать название ни для одного из них. В нашем проекте мы создадим названия, потому что некоторые потребности заинтересованных лиц довольно обширные. Названия требований должны быть короткими, но содержательными.

10. Управление ожиданиями заинтересованных лиц

Задачи управления ожиданиями

- Заручиться поддержкой
- Корректировать ожидания
- Рассматривать тревоги/риски
- Решать проблемы
- Минимизировать отрицательное влияние
- Предвидеть реакции

Какие бывают ожидания?

- Ключевые факторы успеха
- Эффект продажи
- Продукт
- Процесс

Средства управления ожиданиями

1) Вовлечение сторон - "Модель и реализация процесса"
2) Выявление и корректировка ожиданий - "Навыки коммуникаций и управления"
3) Отслеживание изменений - "Инструменты"

От чего зависят ожидания?

- Изначальные договоренности
- Скрытые желания
- Понимание хода проекта и ощущение контроля
- Отношения между заинтересованными сторонами

Как формируются неправильные ожидания?

- Конфликтующие сообщения
- Жаргон
- Межкультурные различия

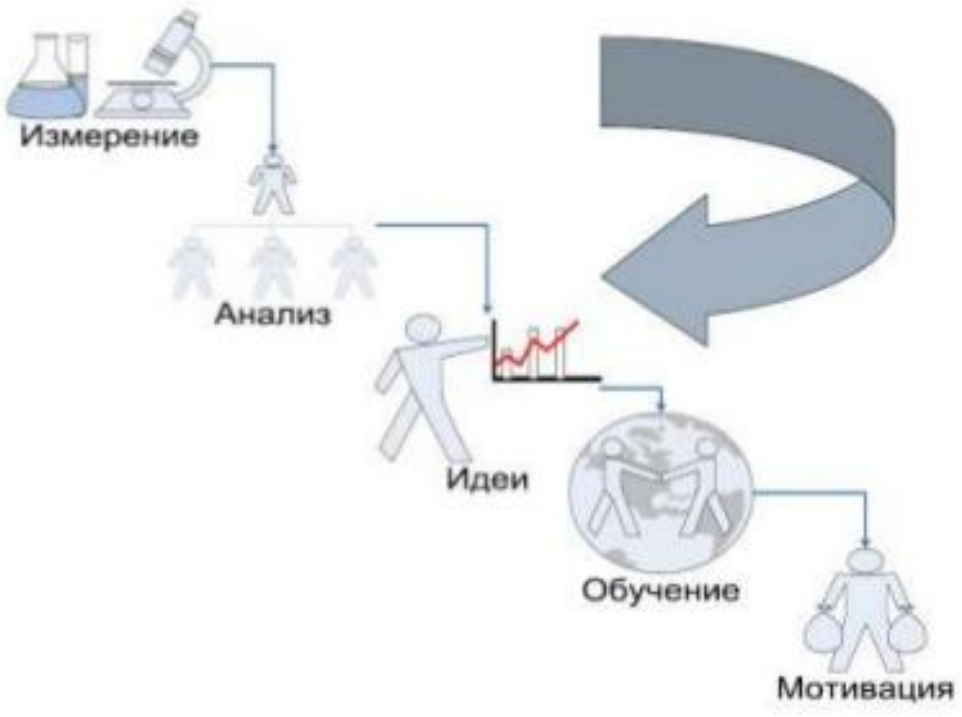
Как формировать правильные убеждения?

- "Делайте то, что проповедуете"
- Выявляйте коммуникационные предпочтения
- Слушайте убедительно
- Используйте принцип "Коломбо"
- Объясните когда и как сказать "НЕТ"

Общие принципы управления ожиданиями

- Выявляйте заинтересованных сторон
- Определяйте скрытые причины проведения проекта
- Вовлекайте клиента в процесс своей работы
- Информируйте все заинтересованные стороны и обучайте их
- Предлагайте им что-то взамен, когда корректируете ожидания

5 шагов к управлению ожиданиями



Управление ожиданиями с примерами

Представьте задачу: построить дом.

- Клиент думает: «Каменный»
- Архитектор: «Кирпичный»
- Строители: «Деревянный, игрушечный, железный»

Никто не знает, какого результата ожидать, пока не обсудит это со всеми участниками проекта.

Два самых первых и главных вопроса перед началом проекта:

1. Для чего и с какой целью делается проект?
2. Что ожидать в результате?

Управление ожиданиями это не только договоренность о результате и понимание того к чему стремиться. Управление ожиданиями это еще и границы ответственности, виды и способы отчетности, методы коммуникаций, измерение успеха и в целом договоренности о ходе работы, о взаимоотношениях с клиентом и с командой.

Вы, как менеджер проекта, договорились с клиентом о старте работ, передали задачи исполнителям. Проходит неделя, работа идет полным ходом, все прекрасно. Вдруг вам звонит клиент и буквально кричит о том, что все пропало, сроки сорваны, почему нет результатов. Вы в недоумении пытаетесь ему объяснить, что все не так, отрываете исполнителей от работы, требуете от них отчетов, ругаетесь на них за медлительность, наспех составляете отписку для клиента и теряете нервные клетки. Клиент все равно недоволен ходом работ и уже сомневается в успехе.

Проблема? Это различия в ожиданиях, которыми никто не управлял ни на старте проекта, ни по ходу работы.

Управление проектами — это, прежде всего, взаимоотношения с людьми. Поэтому многие ситуации аналогичны нашим личным отношениям. Проблема различий в ожиданиях распространяется повсеместно и очень глубоко, решая судьбы, разрывая браки. Женщина считает проявлением любви букет цветов утром у постели, мужчина считает проявлением любви выглаженные женой рубашки. Каждый ждет этого друг от друга и не получает, обвиняя в отсутствии чувств.

Управление ожиданиями — это договоренности. У каждого свои ожидания и если их не обсудить, то проблемы будут возникать ежечасно, ежедневно и всегда приведут к осложнениям.

Как управлять ожиданиями

Обозначить цели и границы работы

Четко и ясно. Недостаточно поставить задачу: «Выкопать яму 2 на 2 метра». Нужно уточнить: «В глубину 3 метра». Иначе мы имеем все шансы получить не тот результат и яму глубиной полметра.

Как бы не был прост этот совет, в нем может затаиться фатальная ошибка напрямую влияющая на результат. Типичная задача может иметь десяток различных целей и соответственно результатов,

особенно в долгосрочной перспективе. Пример из моей области. Казалось бы, что проще и стандартнее создания небольшого корпоративного сайта из пяти страниц. Но у такого сайта может быть несколько целей: привлечь новых клиентов, удержать лояльных клиентов, продать услуги, наладить коммуникации и так далее. Не всегда задача стоит в реализации всех этих целей. Иногда клиент фокусируется только на одной или двух (что часто самое верное решение, особенно с ограниченным бюджетом). Но клиент забывает это ясно и четко озвучить. Если не уточнить, то можем пойти к другой цели и результат не оправдает ожидания, хотя вроде как работа по факту будет выполнена, ведь сайт создан. Но он не будет приносить клиенту выгоду.

Обозначить, что можно и что нельзя делать

Договоренность о степени свободы и ограничениях, помогает в первую очередь избежать потери времени, бесполезных споров и патовых ситуаций.

Клиент ни о каких ограничениях явно не говорит и вы думаете, что у вас полный карт-бланш. Делаете на ваш взгляд невероятно красивую работу, вкладываете всю душу, показываете клиенту результат. И вдруг он заявляет: «Красный цвет не пойдет, он мне не нравится». А ваша работа полностью основана на красном цвете. Капризный клиент? Нет. Вы просто не уточнили ограничения до начала работы.

Сформулировать желания

Конечно же, в договоре обычно прописываются все условия, порядок работы, сроки и прочие моменты взаимоотношений. Но договор — это сухое формулирование фактов и юридические глупости. В договоре не всегда отражается реальное положение дел, и никогда не прописываются настоящие желания и эмоции.

Глядя на сумму договора и сроки работы, можете ли вы сказать насколько важен для клиента этот проект? Может быть очень важен, а может быть и нет. Может быть этот проект сейчас просто способ понять, стоит ли вообще иметь с вами дело, а в дальнейшем передавать вам более существенные задания. В договоре это не пропишешь. Или вот, что значит срок прописанный в договоре? Клиент просто молча с ним согласился, поверив в вашу оценку? Или может быть в день сдачи проекта и в зависимости от результата решится судьба клиента? Допустим он менеджер в компании и если он не представит в этот день проект руководству, то его уволят.

Договориться о методах и частоте коммуникаций

Некоторые клиенты обожают раз в час звонить менеджеру проекта, каждый день и общаться с ним обо всем в течение часа.

Чтобы избавить себя и клиента от ненужных проблем, важно до начала работы обговорить удобные для обеих сторон способы коммуникации. Причем не только в общих случаях, но и в отдельных ситуация. Например, каким способом связи должны решаться срочные вопросы. Это поможет решить проблемы по ходу работы. Весьма глупые проблемы, но тем не менее происходящие постоянно и у всех.

Сколько раз я видел как проекты доходят до кризиса из-за таких мелочей. Вы отправляете клиенту письмо по электронной почте с отчетом о работе. Клиент целый день не открывает почту, к вечеру начинает беспокоится и звонит вам. Вы уже расслабились, отключили телефон и даже не думаете

о надвигающейся буре. На утро клиент наконец дозванивается до вас, он возмущен, переживает и не понимает почему вы не ставите его в курс дела в назначенный для отчета день и не звоните ему, сообщая о результатах. Вы чувствуете всю бредовость ситуации, ведь письмо вчера послали и не ваша вина, что клиент не открывал почту, а о звонке не договаривались. Появляется злость и отчаяние, весь день коту под хвост, работать уже не хочется. Клиент же теряет к вам доверие.

Решить как часто и в какой форме делаются отчеты

Отчеты дело тонкое. Их мало кто любит, но часто они необходимы для отслеживания хода работы. Для некоторых клиентов, особенно, тех кто не сам принимает решение или принимает решение вместе с партнерами — отчеты важны. А форма отчетов и определенная регулярность еще более важны. Если клиент говорит своим партнерам или руководителям, что отчет представит в понедельник, а вы присылаете пару невнятных строк в письме во вторник — это настоящий удар ниже пояса.

Решить, как будет измеряться успех проекта в ходе работы и по ее окончании

Что может быть проще — работа сделана, вот и успех. Так бывает очень редко. Любой проект — это набор определенных показателей и качеств. Соответственно клиент и вы имеете свое представление о них. Если вы заранее не договоритесь о деталях, о способах измерения показателей при сдаче проекта то, велик шанс надолго погрязнуть в выяснении соответствия проекта качествам и показателям, доказывая друг другу свою точку зрения.

Таким образом, необходимо договариваться о степени успешности и завершенности проекта или отдельных его этапов, основываясь на измеримых качествах и показателях. Это очень действенное решение проблемы затянувшейся сдачи проекта и избежания массы недоразумений из-за различий в ожиданиях.

На примере, это может выглядеть так: задача построить дом. Можно построить дом с таким фундаментом, который через месяц развалится. Или с такой крышей, которая при первом снеге обрушится. Но по факту в день сдачи дом прекрасно стоит, ничего не перекошено, ничего не отваливается. И строители сделали свою работу в таком качестве, на которое хватило бюджета и их знаний. Получается в общем-то никто не виноват и задача выполнена. Но будет ли доволен клиент через пару месяцев, когда все рухнет? Ожидал ли он именно такого качества? Будут ли довольны строители, когда им придется бесплатно переделывать или еще хуже, идти в суд?

И ведь вся проблема решается до начала работ, обсуждением ожидаемых показателей и качеств. Причем, после такого обсуждения, когда клиент и исполнители договорились об измерении успешности проекта, очень часто пересматривается бюджет и сроки, в лучшую сторону и по инициативе клиента, который, обычно готов ради предсказуемого успеха идти на дополнительные затраты.

Навыки управления ожиданиями руководителя

1) Правило отсутствия неожиданностей.
2) Правило долгосрочного планирования.
3) Правило вовлечения руководителя.
4) Правило соломки.
5) Правило подсадной утки.
6) Правило обратной связи.

Матрица 2×2. Управление ожиданиями

Задача менеджера – сбалансировать потребности заказчика с целями команды (организации-разработчика). При этом менеджер может работать только с выявленными (communicated) требованиями и ожиданиями.

Достижение этого баланса – сама по себе достаточно сложная задача, так как цели разработчиков и заказчика «не всегда» совпадают. Но, что более важно, баланс явных ожиданий и целей – лишь вершина айсберга.

Например, недавно Слава Панкратов опубликовал статью насколько ощущаемые потребности участников команды отличаются от реальных. Для заказчиков аутсорсинговых проектов по разработке ПО, требования завершить проект в срок, в рамках бюджета и с заданным функционалом – это лишь часть реального спектра потребностей.

На 100% успешным будет проект с точки зрения и заказчика и команды, и на 100% успешной будет выполнена задача менеджера, если будут достигнуты неявные цели и удовлетворены неявные ожидания заказчика и команды. Особенно это важно в небольших проектах, где на первом плане реальные люди и их реальные взаимоотношения, а не формальные роли и процессы.

Особый интерес представляет частный случай этой матрицы, когда рассматриваются не все возможные ожидания, а ожидания о функциональности разрабатываемого продукта. Тогда ячейки будут выглядеть следующим образом:

И эта матрица иллюстрирует 3 критических процесса в разработке ПО, от которых непосредственно зависит успех проекта:

1. Выявление требований (переход от неявных требований заказчика к документированным)

2. Непосредственно разработку (процесс реализации системы в соответствии с архитектурой)

3. Формирование общего видения системы – обеспечение согласованности архитектуры системы с выявленными требованиями – одна из основных задач менеджера проекта.

Целью этих процессов является: обеспечение соответствия реального поведения системы реальным потребностям/ожиданиям заказчиков.

Пример содержания документа описания требований

Пример структуры плана управления требованиями

1. Выявление требований

- Исследования
- Интервью
- Семинар
- Создание прототипов
- Use Case

2. Анализ требований

- Уточнение требований
- Структуризация
- Установка приоритетов

3. Документирование требований

- Состав и распределение работ
- Спецификация требований
- Концепция эксплуатации
- Начальный план разработки ПО
- Критерии принятия работ

4. Управление изменениями требований

- Изменения требований
  - ✓ Предложение изменений
  - ✓ Анализ изменений
  - ✓ Принятие решений
  - ✓ Обновление требований
  - ✓ Обновление планов
- Контроль версий требований
- Прослеживание требований

Управление изменениями

Причины изменения требований

1. **Заказчик**
   - Не понравилось после просмотра
   - Передумал
   - Забыл
2. **Рынок**
   - Такой продукт уже не продать
   - Нужно выйти на рынок прямо сейчас, иначе этот продукт не продать
3. **Разработка**
   - Неточное определение границ проекта
   - Требования неправильно поняты
   - Требования плохо определены
   - Требования не были поняты
   - Сработали архитектурные риски

Условия возможности изменения требований:

- **Водопадные стратегии** – не возможно;
- **Инкрементные стратегии** – возможно с некоторыми ограничениями;
- **Эволюционные стратегии** – возможно.

Политика управления изменениями:

- Должен быть принят процесс контроля за изменениями;
- Все изменения должны следовать процессу или не рассматриваться;
- Для неутвержденных требований не выполняется никаких действий, кроме исследования осуществимости;
- Все запросы на изменение должны быть одобрены советом по управлению изменениями;
- Содержание запроса на изменение должно быть доступно всем заинтересованным лица проекта;
- Начальный текст запроса должен быть неизменным;
- Анализ воздействия должен проводиться для каждого изменения;
- Каждое одобренное изменение (добавленное требование) должно прослеживаться до запроса на изменение;
- Обоснование каждого одобрения на изменение должно быть задокументировано.

Анализ влияния изменения:

- Выявление последствий внесения изменения
- Определение всех сущностей (файлы, модели, артефакты, документы), которые нуждаются в модификации, если изменение будет принято
- Определение задач, необходимых для реализации изменения
- Оценка усилий для завершения этих задач
- Оценка нахождения этих задач на критическом пути проекта
- Оценка влияния на график работ

- Оценка влияния на стоимость
- Оценка приоритета изменения, учитывая
  - ✓ Достоинства
  - ✓ Недостатки
  - ✓ Затраты
  - ✓ Риски

Варианты решения на запрос об изменении требований

- Отложить низкоприоритетные требования
- Привлечь дополнительных сотрудников
- Краткосрочная сверхурочная работа
- Изменение графика работ
- Пожертвование качеством
- ОТКАЗ!

Управление состояниями требований

Определение состояний требований

- **Предложено** - Требование было выставлено авторизованным источником
- **Одобрено** - Требование было проанализировано и одобрено для определенной версии.
- **Реализовано** - Код, реализующий требование, был написан
- **Проверено** - Корректная функциональность донного требования была подтверждена версией продукта. Требование может быть прослежено до варианта тестирования.
- **Удалено** - Ранее одобренное требование было исключено из базисного списка. Причина удаления – задокументирована
- **Отклонено** - Предложенное требование – отклонено. Причина отклонения – задокументирована