Documentation testing

Тестирование документации

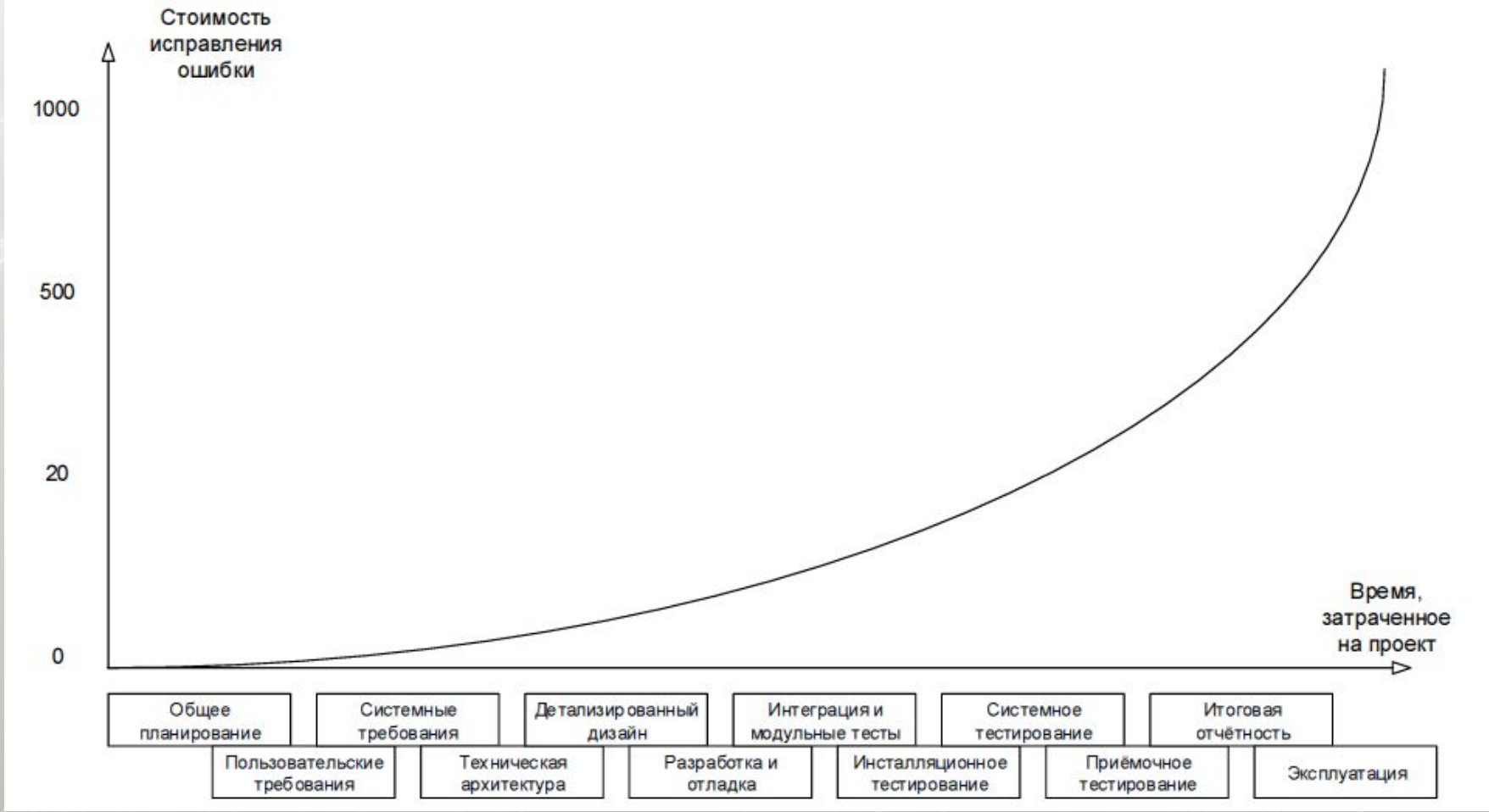
What is requirements?

Требование (***requirement***) —описание того, какие функциии с соблюдением каких условий должно выполнять приложение в процессе решения полезной для пользователя задачи.

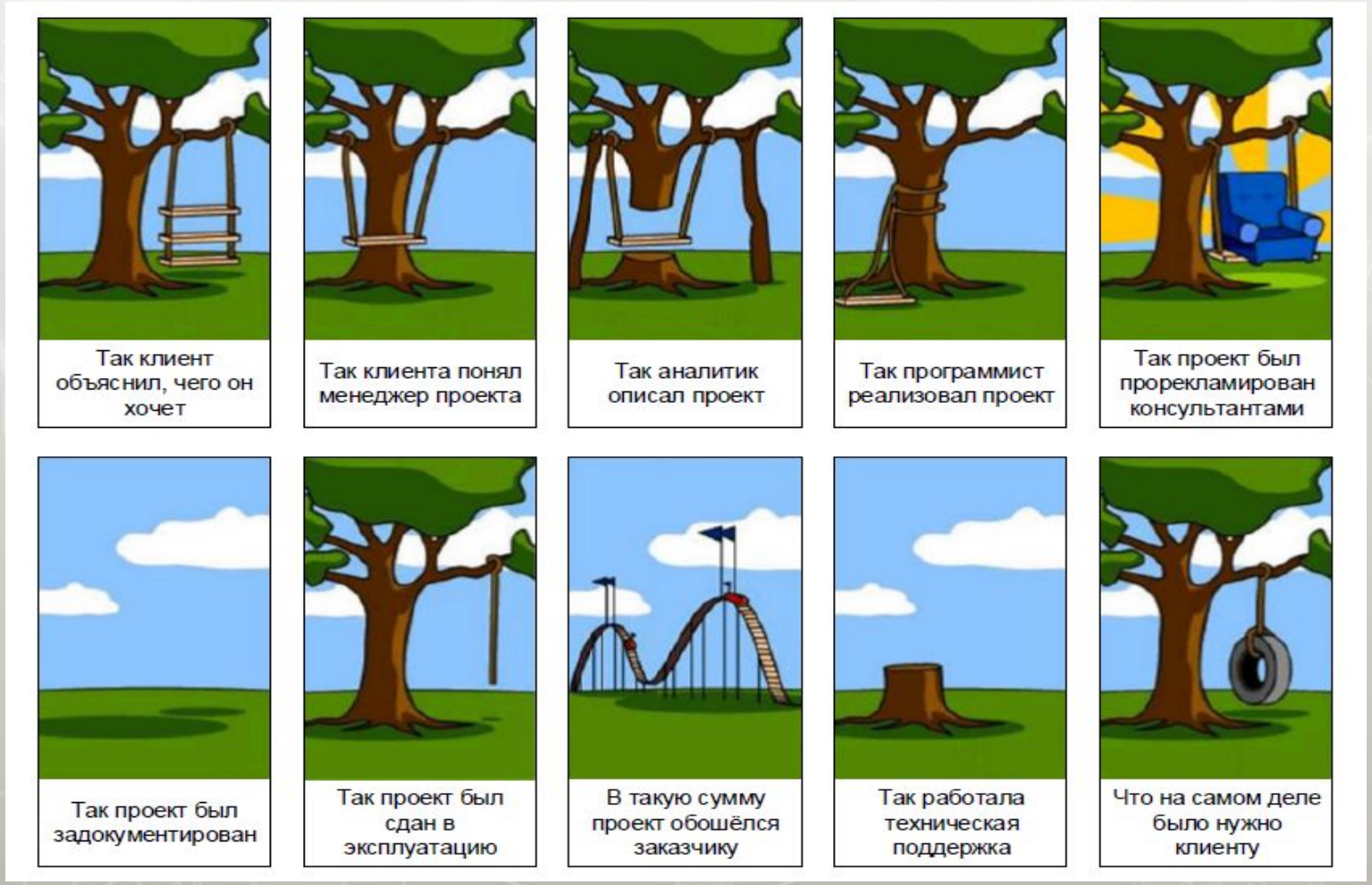
Роль требований в проекте:

- Позволяют понять, что и с соблюдением каких условий система должна делать.
- Предоставляют возможность оценить масштаб изменений и управлять изменениями.
- Являются основой для формирования плана проекта (в том числе плана тестирования).
- Помогают предотвращать или разрешать конфликтные ситуации.
- Упрощают расстановку приоритетов в наборе задач.
- Позволяют объективно оценить степень прогресса в разработке проекта.

Error correction costs



Typical project with poor requirements



Requirements types

В зависимости от времени и места её использования, документация бывает:

Продуктная документация (***product documentation***, ***development documentation***) - используется проектной командой во время разработки и поддержки продукта.

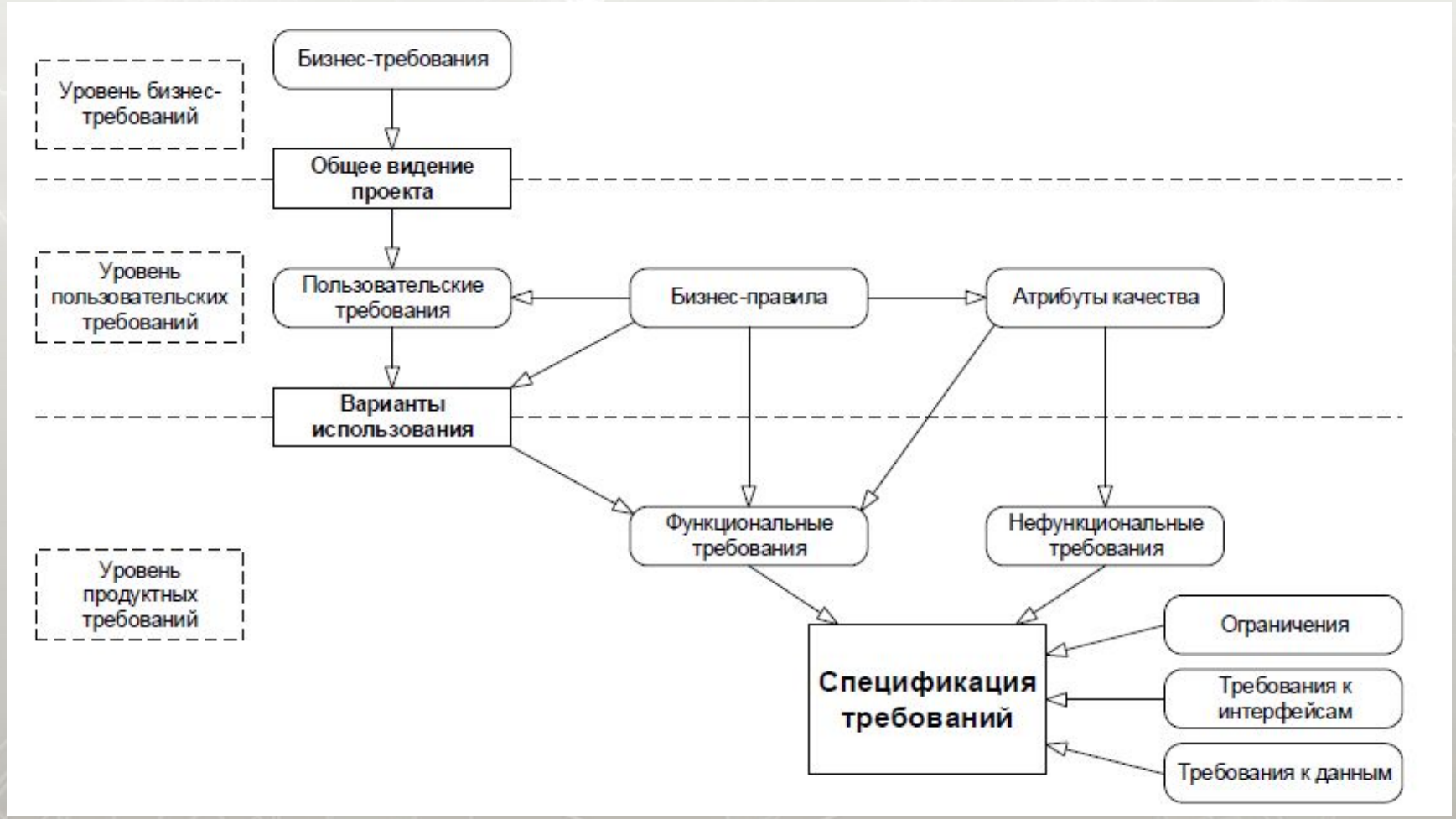
- **План проекта** (***project management plan***), в том числе и **тестовый план** (***test plan***)
- **Требования к программному продукту** (***product requirements document, PRD***)
- **Функциональные спецификации** (***functional specifications document***, ***FSD***; ***software requirements specification***, ***SRS***)
- **Архитектура и дизайн** (***architecture and design***)
- **Тест-кейсы** (***test cases***) и **наборы тест-кейсов** (***test suites***)
- **Технические спецификации** (***technical specifications***)

Requirements types

Проектная документация (***project documentation***) включает в себя как продуктную документацию, так и некоторые дополнительные виды документации и используется не только на стадии разработки, но и на более ранних и поздних стадиях (например, на стадии внедрения и эксплуатации).

- **Пользовательская и сопроводительная документация** (***user and accompanying documentation***)
- **Маркетинговая документация** (***market requirements document, MRD***)

Levels and types of requirements



Types of requirements

Бизнес-требования (***business requirements***) выражают цель, ради которой разрабатывается продукт (зачем вообще он нужен, какая от него ожидается польза). Результатом выявления требований на этом уровне является общее видение (***vision and scope***) — документ, который, как правило, представлен простым текстом и таблицами. Здесь нет детализации поведения системы и иных технических характеристик, но вполне могут быть определены приоритеты решаемых бизнес-задач, риски и т.п.

Примеры бизнес-требований:

- *Нужен инструмент, в реальном времени отображающий наиболее выгодный курс покупки и продажи валюты.*
- *Необходимо в два-три раза повысить количество заявок, обрабатываемых одним оператором за смену.*
- *Нужно автоматизировать процесс выписки товарно-транспортных накладных на основе договоров.*

Types of requirements

Пользовательские требования (***user requirements***) описывают задачи, которые пользователь может выполнять с помощью разрабатываемой системы (реакцию системы на действия пользователя, сценарии работы пользователя). Поскольку здесь уже появляется описание поведения системы, требования этого уровня могут быть использованы для оценки объёма работ, стоимости проекта, времени разработки и т.д. Пользовательские требования оформляются в виде вариантов использования (***use cases***), пользовательских историй (***user stories***), пользовательских сценариев (***user scenarios***).

Примеры пользовательских требований:

- *При первом входе пользователя в систему должно отображаться лицензионное соглашение.*
- *Администратор должен иметь возможность просматривать список всех пользователей, работающих в данный момент в системе.*
- *При первом сохранении новой статьи система должна выдавать запрос на сохранение в виде черновика или публикацию.*

Types of requirements

Бизнес-правила (***business rules***) описывают особенности принятых в предметной области (и/или непосредственно у заказчика) процессов, ограничений и иных правил. Эти правила могут относиться к бизнес-процессам, правилам работы сотрудников, нюансам работы ПО и т.д.

Примеры бизнес-правил:

- *Никакой документ, просмотренный посетителями сайта хотя бы один раз, не может быть отредактирован или удалён.*
- *Публикация статьи возможна только после утверждения главным редактором.*
- *Подключение к системе извне офиса запрещено в нерабочее время.*

Types of requirements

Атрибуты качества (***quality attributes***) расширяют собой нефункциональные требования и на уровне пользовательских требований могут быть представлены в виде описания ключевых для проекта показателей качества (свойств продукта, не связанных с функциональностью, но являющихся важными для достижения целей создания продукта — производительность, масштабируемость, восстанавливаемость). Атрибутов качества очень много, но для любого проекта реально важными является лишь некоторое их подмножество.

Примеры атрибутов качества:

- *Максимальное время готовности системы к выполнению новой команды после отмены предыдущей не может превышать одну секунду.*
- *Внесённые в текст статьи изменения не должны быть утеряны при нарушении соединения между клиентом и сервером.*
- *Приложение должно поддерживать добавление произвольного количества неиероглифических языков интерфейса.*

Types of requirements

Функциональные требования (***functional requirements***) описывают поведение системы, т.е. её действия (вычисления, преобразования, проверки, обработку и т.д.) В контексте проектирования функциональные требования в основном влияют на дизайн системы.

Примеры функциональных требований:

- *В процессе инсталляции приложение должно проверять остаток свободного места на целевом носителе.*
- *Система должна автоматически выполнять резервное копирование данных ежедневно в указанный момент времени.*
- *Электронный адрес пользователя, вводимый при регистрации, должен быть проверен на соответствие требованиям RFC822.*

Types of requirements

Нефункциональные требования (***non-functional requirements***) описывают свойства системы (удобство использования, безопасность, надёжность, расширяемость и т.д.), которыми она должна обладать при реализации своего поведения. Здесь приводится более техническое и детальное описание атрибутов качества. В контексте проектирования нефункциональные требования в основном влияют на архитектуру системы.

Примеры нефункциональных требований:

- *При одновременной непрерывной работе с системой 1000 пользователей, минимальное время между возникновением сбоев должно быть более или равно 100 часов.*
- *Ни при каких условиях общий объём используемой приложением памяти не может превышать 2 ГБ.*
- *Размер шрифта для любой надписи на экране должен поддерживать настройку в диапазоне от 5 до 15 пунктов.*

Types of requirements

Ограничения (***limitations, constraints***) представляют собой факторы, ограничивающие выбор способов и средств реализации продукта.

Примеры ограничений:

- *Все элементы интерфейса должны отображаться без прокрутки при разрешениях экрана от 800x600 до 1920x1080.*
- *Не допускается использование Flash при реализации клиентской части приложения.*
- *Приложение должно сохранять способность реализовывать функции с уровнем важности «критический» при отсутствии у клиента поддержки JavaScript.*

Types of requirements

Требования к интерфейсам (***external interfaces requirements***) описывают особенности взаимодействия разрабатываемой системы с другими системами и операционной средой.

Примеры требований к интерфейсам:

- *Обмен данными между клиентской и серверной частями приложения при осуществлении фоновых AJAX-запросов должен быть реализован в формате JSON.*
- *Протоколирование событий должно вестись в журнале событий операционной системы.*
- *Соединение с почтовым сервером должно выполняться согласно RFC3207 («SMTP over TLS»).*

Types of requirements

Требования к данным (***data requirements***) описывают структуры данных (и сами данные), являющиеся неотъемлемой частью разрабатываемой системы. Часто сюда относят описание базы данных и особенностей её использования.

Примеры требований к данным:

- *Все данные системы, за исключением пользовательских документов, должны храниться в БД под управлением СУБД MySQL, пользовательские документы должны храниться в БД под управлением СУБД MongoDB.*
- *Информация о кассовых транзакциях за текущий месяц должна храниться в операционной таблице, а по завершении месяца переноситься в архивную.*
- *Для ускорения операций поиска по тексту статей и обзоров должны быть предусмотрены полнотекстовые индексы на соответствующих полях таблиц.*

Types of requirements

Спецификация требований (***software requirements specification, SRS***) объединяет в себе описание всех требований уровня продукта и может представлять собой весьма объёмный документ (сотни и тысячи страниц).

Альтернативные названия:

- бизнес-требования (***business requirements document***)
- функциональная спецификация или «спека» (***functional spec***)
- документ с требованиями (***requirements document***) или просто требования (***requirements***).

Requirements management tools - инструменты, которые поддерживают запись требований, атрибутов требований (например, приоритет, ответственного) и аннотацию, и облегчают управление изменениями.

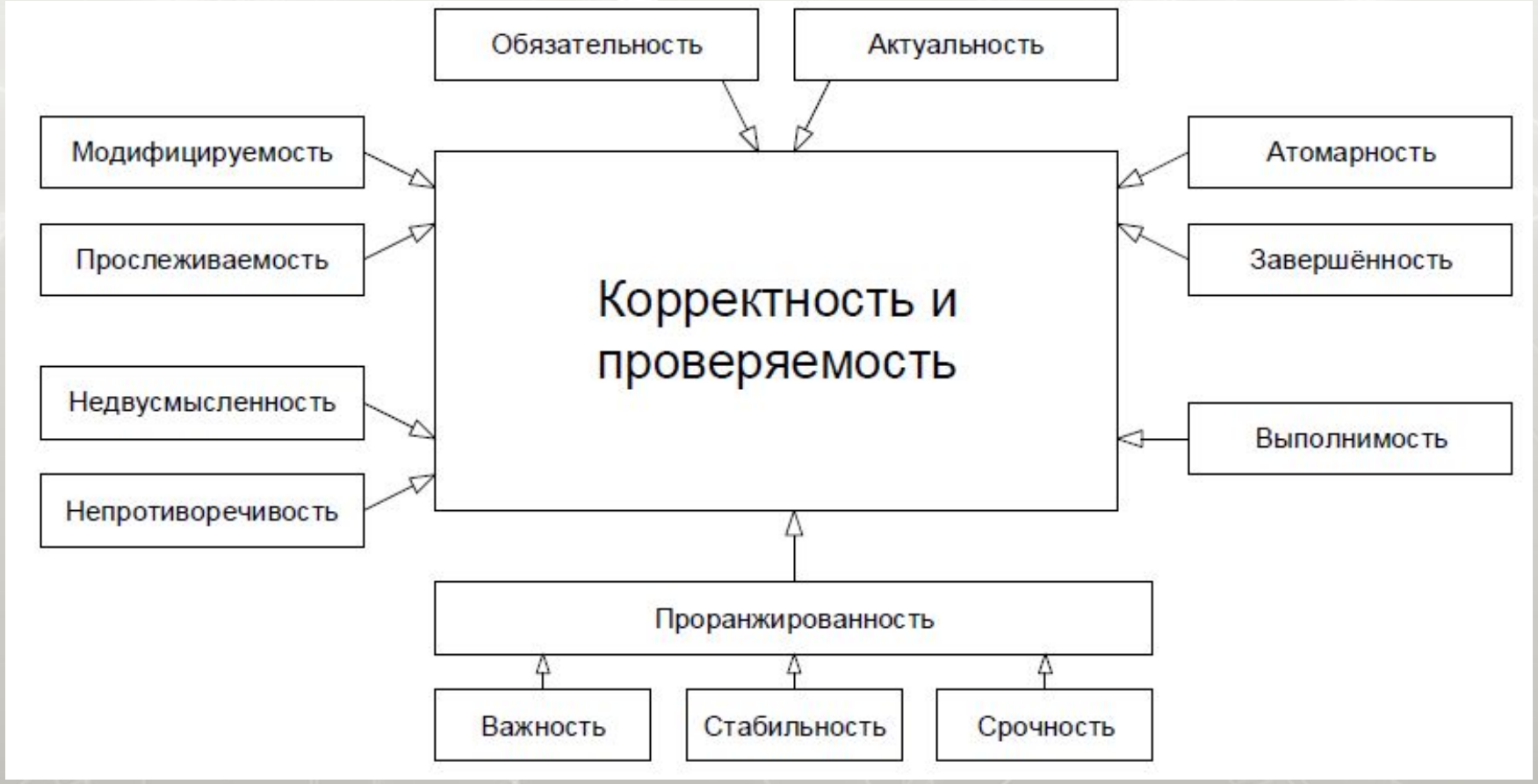
Agile Manager by Hewlett-Packard Enterprise

Blueprint by Blueprint Software Systems, Inc.

Enterprise Architect by Sparx Systems

JIRA Software (Confluence) by Atlassian

Qualitative requirements characteristics



Requirements characteristics

Завершённость (***completeness***). Требование является полным и законченным с точки зрения представления в нём всей необходимой информации, ничто не пропущено по соображениям «это и так всем понятно».

Типичные проблемы с завершённостью:

- Отсутствуют нефункциональные составляющие требования или ссылки на соответствующие нефункциональные требования (например: «*пароли должны храниться в зашифрованном виде*» — каков алгоритм шифрования?).
- Указана лишь часть некоторого перечисления (например: «*экспорт осуществляется в форматы PDF, PNG и т.д.*» — что мы должны понимать под «и т.д.»?).
- Приведённые ссылки неоднозначны (например: «*см. выше*» вместо «*см. раздел 123.45.b*»).

Requirements characteristics

Атомарность, единичность (***atomicity***). Требование является атомарным, если его нельзя разбить на отдельные требования без потери завершённости и оно описывает одну и только одну ситуацию.

Типичные проблемы с атомарностью:

- В одном требовании, фактически, содержится несколько независимых, например: «*кнопка “Restart” не должна отображаться при остановленном сервисе, окно “Log” должно вмещать не менее 20-ти записей о последних действиях пользователя*»
- Требование допускает разночтение в силу грамматических особенностей языка, например: «*если пользователь подтверждает заказ и редактирует заказ или откладывает заказ, должен выдаваться запрос на оплату*».
- В одном требовании объединено описание нескольких независимых ситуаций, например: «*когда пользователь входит в систему, ему должно отображаться приветствие; когда пользователь вошёл в систему, должно отображаться имя пользователя; когда пользователь выходит из системы, должно отображаться прощание*»

Requirements characteristics

Непротиворечивость, последовательность (***consistency***). Требование не должно содержать внутренних противоречий и противоречий другим требованиям и документам.

Типичные проблемы с непротиворечивостью:

- Противоречия внутри одного требования (например: «*после успешного входа в систему пользователя, не имеющего права входить в систему…*» — тогда как он успешно вошёл в систему, если не имел такого права?)
- Противоречия между двумя и более требованиями, между таблицей и текстом, рисунком и текстом, требованием и прототипом и т.д. (например: «*712.a Кнопка “Close” всегда должна быть красной*» и «*36452.x Кнопка “Close” всегда должна быть синей*» — так всё же красной или синей?)
- Использование неверной терминологии или использование разных терминов для обозначения одного и того же объекта или явления (например: «*в случае, если разрешение окна составляет менее 800x600…*» — разрешение есть у экрана, у окна есть размер).

Requirements characteristics

Недвусмысленность (***unambiguousness, clearness***). Требование описано без использования жаргона, неочевидных аббревиатур и расплывчатых формулировок и допускает только однозначное объективное понимание. Требование атомарно в плане невозможности различной трактовки сочетания отдельных фраз.

Типичные проблемы с недвусмысленностью:

- Использование терминов или фраз, допускающих субъективное толкование, например: «*приложение должно поддерживать передачу больших объёмов данных*»
- Вот утрированный пример требования, звучащего очень красиво, но совершенно нереализуемого и непонятного: «*В случае необходимости оптимизации передачи больших файлов система должна эффективно использовать минимум оперативной памяти, если это возможно*».
- Использование неочевидных или двусмысленных аббревиатур без расшифровки, например: «*доступ к ФС осуществляется посредством системы прозрачного шифрования*» и «*ФС предоставляет возможность фиксировать сообщения в их текущем состоянии с хранением истории всех изменений*»

Requirements characteristics

Недвусмысленность (***unambiguousness, clearness***)

- Формулировка требований из соображений, что нечто должно быть всем очевидно, например: «*Система конвертирует входной файл из формата PDF в выходной файл формата PNG*»

Перечень слов и выражений, которые можно считать верными признаками двусмысленности: адекватно (***adequate***), быть способным (***be able to***), легко (***easy***), обеспечивать (***provide for***), как минимум (***as a minimum***), быть способным (***be capable of***), эффективно (***effectively***), своевременно (***timely***), применимо (***as applicable***), если возможно (***if possible***), будет определено позже (***to be determined, TBD***), по мере необходимости (***as appropriate***), если это целесообразно (***if practical***), но не ограничиваясь (***but not limited to***), быть способно (***capability of***), иметь возможность (***capability to***), нормально (***normal***), минимизировать (***minimize***), максимизировать (***maximize***), оптимизировать (***optimize***), быстро (***rapid***), удобно (***user-friendly***), просто (***simple***), часто (***often***), обычно (***usual***), большой (***large***), гибкий (***flexible***), устойчивый (***robust***), по последнему слову техники (***state-of-the-art***), улучшенный (***improved***), результативно (***efficient***).

Requirements characteristics

Выполнимость (***feasibility***). Требование технологически выполнимо и может быть реализовано в рамках бюджета и сроков разработки проекта.

Типичные проблемы с выполнимостью:

- Так называемое «озолочение» (***gold plating***) — требования, которые крайне долго и/или дорого реализуются и при этом практически бесполезны для конечных пользователей, например: «*настройка параметров для подключения к базе данных должна поддерживать распознавание символов из жестов, полученных с устройств трёхмерного ввода*».
- Технически нереализуемые на современном уровне развития технологий требования, например: «*анализ договоров должен выполняться с применением искусственного интеллекта, который будет выносить однозначное корректное заключение о степени выгоды от заключения договора*».
- В принципе нереализуемые требования, например: «*система поиска должна заранее предусматривать все возможные варианты поисковых запросов и кэшировать их результаты*».

Requirements characteristics

Обязательность, нужность (***obligation***) и **актуальность** (***up-to-date***). Если требование не является обязательным к реализации, оно должно быть просто исключено из набора требований. Если требование нужное, но «не очень важное», для указания этого факта используется указание приоритета. Также исключены (или переработаны) должны быть требования, утратившие актуальность.

Типичные проблемы с обязательностью и актуальностью:

- Требование было добавлено «на всякий случай», хотя реальной потребности в нём не было и нет.
- Требованию выставлены неверные значения приоритета по критериям важности и/или срочности.
- Требование устарело, но не было переработано или удалено.

Requirements characteristics

Прослеживаемость (***traceability***). Прослеживаемость бывает **вертикальной** (***vertical traceability***) и **горизонтальной** (***horizontal traceability***). Вертикальная позволяет соотносить между собой требования на различных уровнях требований, горизонтальная позволяет соотносить требование с тест-планом, тест-кейсами, архитектурными решениями и т. д.

Для обеспечения прослеживаемости часто используются специальные инструменты по управлению требованиями (***requirements management tool***), и/или матрицы прослеживаемости (***traceability matrix***).

Типичные проблемы с прослеживаемостью:

- Требования не пронумерованы, не структурированы, не имеют оглавления, не имеют работающих перекрёстных ссылок.
- При разработке требований не были использованы инструменты и техники управления требованиями.
- Набор требований неполный, носит обрывочный характер с явными «пробелами».

Requirements characteristics

Модифицируемость (***modifiability***). Это свойство характеризует простоту внесения изменений в отдельные требования и в набор требований. Можно говорить о наличии модифицируемости в том случае, если при доработке требований искомую информацию легко найти, а её изменение не приводит к нарушению иных описанных в этом перечне свойств.

Типичные проблемы с модифицируемостью:

- Требования не атомарны и непрослеживаемы, а потому их изменение с высокой вероятностью порождает противоречивость.
- Требования изначально противоречивы. В такой ситуации внесение изменений (не связанных с устранением противоречивости) только усугубляет ситуацию, увеличивая противоречивость и снижая прослеживаемость.
- Требования представлены в неудобной для обработки форме (например, не использованы инструменты управления требованиями, и в итоге команде приходится работать с десятками огромных текстовых документов).

Requirements characteristics

Проранжированность по важности, стабильности, срочности (***ranked for importance, stability, priority***).

Типичные проблемы с проранжированностью состоят в её отсутствии или неверной реализации.

Важность (***importance***) характеризует зависимость успеха проекта от успеха реализации требования. Проблемы повышают риск неверного распределения усилий проектной команды, направления усилий на второстепенные задачи и конечного провала проекта из-за неспособности продукта выполнять ключевые задачи с соблюдением ключевых условий.

Стабильность (***stability***) характеризует вероятность того, что в обозримом будущем в требование не будет внесено никаких изменений. Проблемы повышают риск выполнения бессмысленной работы по совершенствованию, реализации и тестированию требований, которые в самое ближайшее время могут претерпеть кардинальные изменения.

Срочность (***priority***) определяет распределение во времени усилий проектной команды по реализации того или иного требования. Проблемы повышают риск нарушения желаемой заказчиком последовательности реализации функциональности и ввода этой функциональности в эксплуатацию.

Requirements characteristics

Корректность (***correctness***) и **проверяемость** (***verifiability***). Фактически эти свойства вытекают из соблюдения всех вышеперечисленных (они не выполняются, если нарушено хотя бы одно из вышеперечисленных).

Проверяемость подразумевает возможность создания объективного тест-кейса (тест-кейсов), однозначно показывающего, что требование реализовано верно и поведение приложения в точности соответствует требованию.

К типичным проблемам с корректностью также можно отнести:

- опечатки (особенно опасны опечатки в аббревиатурах, превращающие одну осмысленную аббревиатуру в другую также осмысленную, но не имеющую отношения к некоему контексту; такие опечатки крайне сложно заметить);
- наличие неаргументированных требований к дизайну и архитектуре;
- плохое оформление текста и сопутствующей графической информации, грамматические, пунктуационные и иные ошибки в тексте;
- неверный уровень детализации;
- требования к пользователю, а не к приложению (например: «пользователь должен быть в состоянии отправить сообщение» — увы, мы не можем влиять на состояние пользователя).

Requirements testing techniques

Взаимный просмотр (**peer review**). Взаимный просмотр («рецензирование») является одной из наиболее активно используемых техник тестирования требований и может быть представлен в одной из трёх следующих форм (по мере нарастания его сложности и цены):

- **Беглый просмотр** (***walkthrough***) может выражаться как в показе автором своей работы коллегам с целью создания общего понимания и получения обратной связи, так и в простом обмене результатами работы между двумя и более авторами с тем, чтобы коллега высказал свои вопросы и замечания. Это самый быстрый, самый дешёвый и наиболее широко используемый вид просмотра.
- **Технический просмотр** (***technical review***) выполняется группой специалистов. В идеальной ситуации каждый специалист должен представлять свою область знаний. Просматриваемый продукт не может считаться достаточно качественным, пока хотя бы у одного просматривающего остаются замечания.

Requirements testing techniques

Взаимный просмотр (**peer review**)

- **Формальная инспекция** (***inspection***) представляет собой структурированный, систематизированный и документируемый подход к анализу документации. Для его выполнения привлекается большое количество специалистов, само выполнение занимает достаточно много времени, и потому этот вариант просмотра используется достаточно редко (как правило, при получении на сопровождение и доработку проекта, созданием которого ранее занималась другая компания).

Requirements testing techniques

Вопросы - повторное использование техник выявления требований (задавание вопросов). Можно спросить представителей заказчика, можно обратиться к справочной информации, к более опытным коллегам при условии, что у них имеется соответствующая информация, ранее полученная от заказчика.

Главное, чтобы ваш вопрос был сформулирован таким образом, чтобы полученный ответ позволил улучшить требования.

Good and Bad questions

Плохое требование	Плохие вопросы	Хорошие вопросы
«Приложение должно быстро запускаться».	Насколько быстро? А если не получится быстро? Всегда?	• Каково максимально допустимое время запуска приложения, на каком оборудовании и при какой загруженности этого оборудования операционной системой и другими приложениями? • На достижение каких целей влияет скорость запуска приложения? • Допускается ли фоновая загрузка отдельных компонентов приложения? • Что является критерием того, что приложение закончило запуск?
«Опционально должен поддерживаться экспорт документов в формат PDF».	Любых документов? В PDF какой версии должен производиться экспорт? Зачем?	• Насколько возможность экспорта в PDF важна? • Как часто, кем и с какой целью она будет использоваться? • Является ли PDF единственным допустимым форматом для этих целей или есть альтернативы? Допускается ли использование внешних утилит (например, виртуальных PDF-принтеров) для экспорта документов в PDF?
«Если дата события не указана, она выбирается автоматически».	А если указана? А если дату невозможно выбрать автоматически? А если у события нет даты?	• Возможно, имелось в виду, что дата генерируется автоматически, а не выбирается? • Если да, то по какому алгоритму она генерируется? • Если нет, то из какого набора выбирается дата и как генерируется этот набор? • P.S. Возможно, стоит использовать текущую дату?

Requirements testing techniques

Тест-кейсы и чек-листы. Хорошее требование является проверяемым, а значит, должны существовать объективные способы определения того, верно ли реализовано требование.

Исследование поведения системы. Тестированию подвергается не одно требование, а целый набор.

Рисунки (графическое представление). Чтобы увидеть общую картину требований целиком, очень удобно использовать рисунки, схемы, диаграммы, интеллект-карты (карта мыслей, диаграмма связей, ***mind-map***) и т.д

Прототипирование. Является следствием создания графического представления и анализа поведения системы.

Business requirements

Необходим инструмент для автоматического приведения кодировок текстовых документов к одной. (**File Converter**)

- В каких форматах представлены текстовые документы (обычный текст, HTML, MD, что-то иное)?
  Понятия не имею, я в этом не разбираюсь.
- В каких кодировках приходят исходные документы?
  В разных.
- В какую кодировку нужно преобразовать документы?
  В самую удобную и универсальную.
- На каких языках написан текст в документах?
  Русский и английский
- Откуда и как поступают текстовые документы (по почте, с сайтов, по сети, как-то иначе)?
  Это неважно. Поступают отовсюду, но мы их складываем в одну папку на диске, нам так удобно
- Каков максимальный объём документа?
  Пара десятков страниц
- Как часто появляются новые документы (например, сколько максимум доку-ментов может поступить за час)?
  200–300 в час
- С помощью чего сотрудники просматривают документы?
  Notepad++

Business requirements level

Суть проекта: разработка инструмента, устраняющего проблему множественности кодировок в текстовых документах, расположенных в локальном дисковом хранилище.

Цели проекта:

- Исключение необходимости ручного подбора кодировок текстовых документов.
- Сокращение времени работы с текстовым документом на величину, необходимую для ручного подбора кодировки.

Метрики достижения целей:

- Полная автоматизация определения и преобразования кодировки текстового документа к заданной.
- Сокращение времени обработки текстового документа в среднем на 1–2 минуты на документ за счёт устранения необходимости ручного подбора кодировки.

Риски:

- Высокая техническая сложность безошибочного определения исходной кодировки текстового документа.

File Converter User requirements

Пользователь (в данном проекте) - человека, настраивающего работу приложения на сервере.

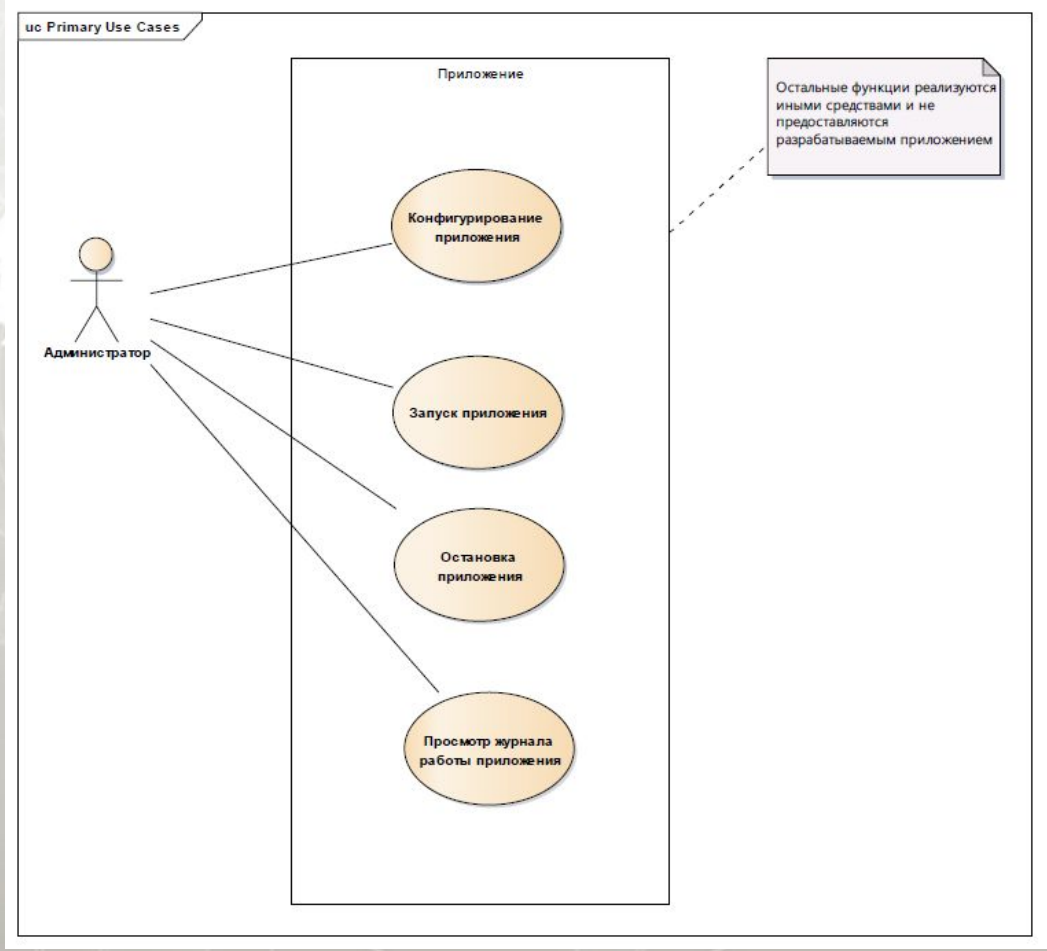


Диаграмма вариантов использования

User requirements level

Системные характеристики

СХ-1: Приложение является консольным.

СХ-2: Для работы приложение использует интерпретатор PHP.

СХ-3: Приложение является кроссплатформенным.

Пользовательские требования

ПТ-1: Запуск и остановка приложения.

ПТ-1.1: Запуск приложения производится из консоли командой «PHP converter.php параметры».

ПТ-1.2: Остановка приложения производится выполнением команды Ctrl+C.

ПТ-2: Конфигурирование приложения.

ПТ-2.1: Конфигурирование приложения сводится к указанию путей в файловой системе.

ПТ-2.2: Целевой кодировкой является UTF8.

ПТ-3: Просмотр журнала работы приложения.

ПТ-3.1: В процессе работы приложение должно выводить журнал своей работы в консоль и лог-файл.

ПТ-3.2: При первом запуске приложения лог-файл создаётся, а при последующих — дописывается.

User requirements level

Бизнес-правила

БП-1: Источник и приёмник файлов

БП-1.1: Каталоги, являющиеся источником исходным и приёмником конечных файлов не должны совпадать.

БП-1.2: Каталог, являющийся приёмником конечных файлов, не может быть подкаталогом источника.

Атрибуты качества

АК-1: Производительность

АК-1.1: Приложение должно обеспечивать скорость обработки данных 5 МБ/сек.

АК-2: Устойчивость к входным данным

АК-2.1: Приложение должно обрабатывать входные файлы размером до 50 МБ включительно.

АК-2.2: Если входной файл не является текстовым, приложение должно произвести обработку.

User requirements level

Системные характеристики

- *СХ-1: Приложение является консольным.*
- *СХ-2: Для работы приложение использует интерпретатор PHP. ???*
- *СХ-3: Приложение является кроссплатформенным.*

Пользовательские требования

- *Также см. диаграмму вариантов использования.*
- *ПТ-1: Запуск и остановка приложения.*
  - *ПТ-1.1: Запуск приложения производится из консоли командой "~~PHP~~ php converter.php параметры".*
  - *ПТ-1.2: Остановка приложения производится выполнением команды Ctrl+C в окне консоли, из которого было запущено приложение.*
- *ПТ-2: Конфигурирование приложения.*
  - *ПТ-2.1: Конфигурирование приложения сводится к указанию путей в файловой системе.*
  - *ПТ-2.2: Целевой кодировкой является UTF8.*
- *ПТ-3: Просмотр журнала работы приложения.*
  - *ПТ-3.1: В процессе работы приложение должно выводить журнал своей работы в консоль и лог-файл.*
  - *ПТ-3.2: При первом запуске приложения лог-файл создаётся, а при последующих – дописывается.*

Author
1) Какая минимальная версия интерпретатора PHP поддерживается приложением?
2) Существует ли некая специфика настройки интерпретатора PHP для корректной работы приложения?

Author
Должна ли в руководстве пользователя быть описана процедура установки и настройки интерпретатора PHP?

Author
Formatted: Russian

Author
Какие ОС должны поддерживаться? В чём цель кроссплатформенности?

Author
Какие параметры передаются скрипту при запуске? Какова реакция скрипта на:
- Отсутствие параметров.
- Неверное количество параметров.
- Неверные значения каждого из параметров.

Author
Путей к чему?

User requirements analyzing

Системные характеристики

СХ-1: Приложение является консольным.

СХ-2: Для работы приложение использует **интерпретатор PHP**.

- ***Какая минимальная версия интерпретатора PHP поддерживается приложением?*** **(5.5.x)**
- ***Существует ли некая специфика настройки интерпретатора PHP для корректной работы приложения?*** **(Наверное, должен работать mbstring)**
- ***Настаиваете ли вы на реализации приложения именно на PHP? Если да, то почему.*** **(Да, только PHP. У нас есть сотрудник, который его знает)**
- ***Должна ли в руководстве пользователя быть описана процедура установки и настройки интерпретатора PHP?*** **(Нет)**

СХ-3: Приложение является кроссплатформенным.

- ***Какие ОС должны поддерживаться?*** **(Любая, где работает PHP)**
- ***В чём вообще цель кроссплатформенности?*** **(Мы ещё не знаем, на чём будет работать сервер.)**

User requirements analyzing

Пользовательские требования

Также см. диаграмму вариантов использования.

ПТ-1: Запуск и остановка приложения.

ПТ-1.1: Запуск приложения производится из консоли командой «PHP *(**возможно, здесь опечатка: должно быть php (в нижнем регистре)**)* **(Да, OK)** converter.php параметры».

- ***Какие параметры передаются скрипту при запуске?*** **(Каталог с исходными файлами, каталог с конечными файлами)**
- ***Какова реакция скрипта на:***
  - ***отсутствие параметров***; **(Пишет хелп)**
  - ***неверное количество параметров***; **(Пишет хелп и поясняет, что не так.)**
  - ***неверные значения каждого из параметров***. **(Пишет хелп и поясняет, что не так.)**

ПТ-1.2: Остановка приложения производится выполнением команды Ctrl+C *(**предлагаем дополнить это выражение фразой «в окне кон-соли, из которого было запущено приложение»**)* **(Да, OK.)**.

ПТ-2: Конфигурирование приложения.

ПТ-2.1: Конфигурирование приложения сводится к указанию путей в файловой системе.

- ***Путей к чему?*** **(Каталог с исходными файлами, каталог с ко-нечными файлами.)**

User requirements analyzing

ПТ-2.2: Целевой кодировкой является UTF8.

- ***Предполагается ли указание иной целевой кодировки, или UTF8 используется в качестве целевой всегда?*** **(Только UTF8, других не надо)**

ПТ-3: Просмотр журнала работы приложения.

ПТ-3.1: В процессе работы приложение должно выводить журнал своей работы в консоль и лог-файл.

- ***Каков формат журнала?*** **(Дата-время, что и с чем делали, что получилось. Гляньте в логе апача, там нормально написано)**
- ***Различаются ли форматы журнала для консоли и лог-файла?*** **(Нет.)**
- ***Как определяется имя лог-файла?*** **(Третий параметр при за-пуске. Если не указан — пусть будет converter.log рядом с php-скриптом)**

ПТ-3.2: При первом запуске приложения лог-файл создаётся, а при по-следующих — дописывается.

- ***Как приложение различает свой первый и последующие запуски?*** **(Никак)**
- ***Какова реакция приложения на отсутствие лог-файла в случае, если это не первый запуск?*** **(Создаёт. Тут идея в том, чтобы оно не затирало старый лог — и всё)**

User requirements analyzing

Бизнес-правила

БП-1: Источник и приёмник файлов

БП-1.1: Каталоги, являющиеся источником исходны**м** *(**опечатка, исходных**)* **(Да)** и приёмником конечных файлов, не должны совпадать.

- ***Какова реакция приложения в случае совпадения этих каталогов?*** **(Пишет хелп и поясняет, что не так)**

БП-1.2: Каталог, являющийся приёмником конечных файлов, не может быть подкаталогом источника *(**предлагаем заменить слово «источника» на фразу «каталога, являющегося источником исходных фай-лов»**).* **(Не надо, непонятно становится)**

User requirements analyzing

Атрибуты качества

АК-1: Производительность

АК-1.1: Приложение должно обеспечивать скорость обработки данных 5 МБ/сек.

- ***При каких технических характеристиках системы?*** **(i7, 4GB RAM)**

АК-2: Устойчивость к входным данным

АК-2.1: Приложение должно обрабатывать входные файлы размером до 50 МБ включительно.

- ***Какова реакция приложения на файлы, размер которых превышает 50 МБ?*** **(Не трогает.)**

АК-2.2: Если входной файл не является текстовым, приложение должно произвести обработку.

- ***Обработку чего должно произвести приложение?*** **(Этого файла. Не важно, что станет с файлом, лишь бы скрипт не умер.)**

Product requirements analyzing

Для удобства формируется единый документ, который интегрирует в себе как пользовательские требования, так и детальные спецификации. Документация на продуктном уровне пригодна для того, чтобы разработчики смогли реализовать приложение, а тестировщики — протестировать его.

Errors in requirements testing

Типичные ошибки при анализе и тестировании требований

- Изменение формата файла и документа
- Отметка того факта, что с требованием всё в порядке.
- Описание одной и той же проблемы в нескольких местах
- Написание вопросов и комментариев без указания места требования, к которым они относятся
- Задавание плохо сформулированных вопросов
- Написание очень длинных комментариев и/или вопросов
- Критика текста или даже его автора
- Категоричные заявления без обоснования
- Указание проблемы с требованиями без пояснения её сути
- Плохое оформление вопросов и комментариев
- Описание проблемы не в том месте, к которому она относится
- Ошибочное восприятие требования как «требования к пользователю»
- Скрытое редактирование требований
- Анализ, не соответствующий уровню требований