Topic 5. Requirements Analysis. Estimation

Содержание:

1. Software Requirements
2. Analysis Software Requirements
3. Requirements Documentation
4. Estimation

Software Requirements - совокупность утверждений относительно атрибутов, свойств или качеств программной системы, подлежащей реализации. Создаются в процессе разработки требований к программному обеспечению, в результате анализа требований.

By levels:

Бизнес-требования — определяют назначение ПО, описываются в документе о видении (vision) и границах проекта (scope).

Пользовательские требования — определяют набор пользовательских задач, которые должна решать программа, а также способы (сценарии) их решения в системе. Пользовательские требования могут выражаться в виде фраз утверждений, в виде способов применения (use case), пользовательских историй (user story), сценариев взаимодействия (scenario).

Функциональные требования — охватывают предполагаемое поведение системы, определяя действия, которые система способна выполнять. Описывается в системной спецификации.

By characters:

1. Функциональный характер — требования к поведению системы.

- Бизнес-требования
- Пользовательские требования
- Функциональные требования

2. Производные требования — требования, которые подразумеваются или преобразованы из высокоуровневого требования. Например, требование для большего радиуса действия или высокой скорости может привести к требованию низкого веса.

1. **Нефункциональный характер** — требования к характеру поведения системы.

- **Бизнес-правила** — определяют ограничения, проистекающие из предметной области и свойств автоматизируемого объекта (предприятия)
- **Системные требования и ограничения** — определения элементарных операций, которые должна иметь система, а также различных условий, которым она может удовлетворять. К системным требованиям и ограничениям относятся:
  - — Ограничения на программные интерфейсы, в том числе к внешним системам
  - — Требования к атрибутам качества
  - — Требования к применяемому оборудованию и ПО
- **Требования к документированию**
- **Требования к дизайну и юзабилити**
- **Требования к безопасности и надёжности**
- **Требования к показателям назначения** (производительность, устойчивость к сбоям и т. п.)
- **Требования к эксплуатации и персоналу**
- **Прочие требования и ограничения** (внешние воздействия,

- Федеральное и муниципальное отраслевое законодательство (конституция, законы, распоряжения)
- Нормативное обеспечение организации (регламенты, положения, уставы, приказы)
- Текущая организация деятельности объекта автоматизации
- Модели деятельности (диаграммы бизнес-процессов)
- Представления и ожидания потребителей и пользователей системы
- Журналы использования существующих программно-аппаратных систем
- Конкурирующие программные продукты

Полнота (отдельного требования и системы требований) — требование должно содержать всю необходимую информацию для его реализации. В него включается вся информация об описываемом параметре, известная на момент описания. Система требований также не должна содержать невыявленных и не определенных требований. Причины неполноты описания следует явно объявлять.

Однозначность — требование должно быть внутренне непротиворечиво и все работающие с ним должны понимать его одинаково. Требования следует выражать просто, кратко и точно, используя известные термины. Обычно базовые знания читателей спецификации требований к ПО различаются. Поэтому в ее состав нужно включить раздел с определением понятий прикладной области, используемых при определении требований. Пример, неоднозначного требования. «Период обновления экрана должен быть не менее 20 сек.»

Корректность отдельного требования и согласованность (непротиворечивость) системы требований — требование не должно содержать в себе неверной, неточной информации, а отдельные требования в системе требований не должны противоречить друг другу.

Необходимость — требование должно отражать возможность или характеристику ПО, *действительно* необходимую пользователям, или вытекающую из других требований.

Осуществимость — включаемое в спецификацию требование должно быть выполнимым при заданных ограничениях операционной среды. Осуществимость требований проверяется в процессе анализа осуществимости разработчиком. В частности, для нефункциональных требований проверяется возможность достижения указанных численных значений при существующих ограничениях.

Проверяемость — проверяемость требования означает, что существует конечный и разумный по стоимости процесс ручной или машинной проверки того, что ПО удовлетворяет этому требованию. Каждое требование (особенно нефункциональное) должно содержать достаточно информации для однозначной проверки его реализации. Иначе, факт реализации будет основываться на мнении, а не на анализе, что приведет к проблемам при сдаче готового ПО. Для атрибутов качества (как мы помним, отдельной разновидности нефункциональных требований) критерием проверямости можно считать наличие численных значений характеристик качества продукта или системы.

Анализ требований — это процесс сбора требований к программному обеспечению, их систематизации, документированию, анализа, выявления противоречий, неполноты, разрешения конфликтов в процессе разработки программного обеспечения.

Анализ требований включает три типа деятельности:

- **Сбор требований:**
общение с клиентами и пользователями, чтобы определить, каковы их требования.

- **Анализ требований:** определение, являются ли собранные требования неясными, неполными, неоднозначными или противоречащими; решение этих проблем.

- **Документирование требований:** требования могут быть задокументированы в различных формах, таких как простое описание, сценарии использования, пользовательские истории, или спецификации процессов.

Традиционный способ документировать требования — это создание списков требований. В сложной системе такие списки требований могут занимать сотни страниц.

Соответствующей метафорой может быть чрезвычайно длинный список покупок в магазине. Такие списки крайне неэффективны в современном анализе, хотя используются и по сей день.

Преимущества:

• Обеспечивает контрольный список требований.
• Обеспечивает договор между заказчиками и разработчиками.
• Для большой системы может обеспечить описание высокого уровня.

Недостатки:

• Такие списки могут занимать сотни страниц.
• Перечисляют отдельные требования абстрактно:
• Лишает возможности видеть, как требования связываются между собой или работают вместе.
• Мешает верно расположить требования по приоритетам.
• Увеличивается вероятность неверного трактования требований.
• Трудно убедиться, что у вас есть все необходимые требования.
• Ложное чувство взаимопонимания между заинтересованными лицами и разработчиками.

Прототипы — макеты системы. Макеты дают возможность пользователям представить систему, которая ещё не построена. Опытные образцы помогают пользователям представить, на что будет похожа система, и облегчают пользователям принятие проектных решений, не дожидаясь окончания постройки системы.

Вариант использования (*Use Case*) — техника для документации потенциальных требований для создания новой системы или изменения существующей. Каждый вариант описывает один или несколько способов взаимодействия системы с конечным пользователем или другой системой, для достижения определённой цели. Варианты использования обычно избегают технического жаргона, предпочитая вместо этого язык конечного пользователя или эксперта в данной области. Они часто создаются совместно специалистами по сбору требований и заинтересованными лицами.

Спецификация требований программного обеспечения (*Software Requirements Specification, SRS*) является полным описанием поведения системы, которая будет создана. Она включает ряд сценариев использования, которые описывают все виды взаимодействия пользователей с программным обеспечением. Сценарии использования также известны как **функциональные требования**. В дополнении к сценариям использования, спецификация программного обеспечения также содержат нефункциональные (или дополнительные) требования. **Нефункциональные требования** — требования, которые налагают дополнительные ограничения на систему.

Оценка – один из наиболее часто встречающихся тасков в IT индустрии: программисты оценивают продолжительность разработки, тестировщики оценивают время тестирования, менеджеры оценивают общее время разработки проекта. Люди, координирующие нас, просят точных оценок, но мы все понимаем, что эстимейты - это только предсказание которое лишь возможно истинно.

Таким образом задача — дать как можно более аккуратную оценку стоимости проекта, максимально приближенную к реальности, которая поможет успешно закрыть сделку и в то же время не заставит всю команду работать сутками за гроши.

1. Разделяй и властвуй – этот старый принцип отлично подходит к нашей ситуации. Вы не можете оценить что-то, что вы не можете разбить на части.

2. Найти ориентир – вспомнить подобные задачи.

3. Обработка рисков:

- Реализация подобных задач;
- Работа с данной платформой/средой;
- Необходимые тулзы/знаний/ресурсы;
- Правильность понятия задач/требований/процесса разработки;
- Правильность планирования;
- Наличие документации.

FAST (Facilitated Application Specification Techniques - технология упрощенной спецификации приложения), представляющей собой специальный тип собеседований с заказчиком, который облегчает выявление его требований.

Таблица 2.1. Характерные особенности избранных методов статического тестирования [28]

	Инспекции	Сквозной контроль	Экспертные оценки
Презентатор	Любой, но не автор	Любой	Нет
Число участников	1-6 человек	5 или более	1 или два
Подготовка	Да	Только презентатор	Нет
Сбор данных	Да	Не обязательно	Нет
Отчет по результатам	Да	Не обязательно	Словесный комментарии или комментарий по электронной почте
Достоинства	Максимальная эффективность	Большее число участников	Минимальные затраты
Недостатки	Наибольшие затраты	Обнаруживает меньше дефектов, чем другие способы	Обнаруживает минимальное число дефектов по сравнению с другими способами

Инспекции

Основной организационной формой инспекции является совещание, на котором рабочий продукт анализируется с целью обнаружения дефектов. Каждый участник специально готовится к совещанию, а само совещание проводится в соответствии со специальным набором правил. Обнаруженные дефекты документируются, итоги совещания публикуются. Практика показала высокую эффективность таких сообщений с точки зрения обнаружения дефектов. Данные, опубликованные Бремом [28], показывают, что если на инспекции приходится 20% трудозатрат на программирование, то из инспектируемого программного кода будет изъято примерно 80% ошибок на уровне программных модулей.

Сквозной контроль

Сквозной контроль представляет собой менее формальное мероприятие, чем инспекции, в том смысле, что ни от одного из его исполнителей не требуется специальной подготовки, за исключением разве что презентатора. Кроме того, никаких итоговых отчетов при этом не требуется. Поскольку сквозной контроль формализован в меньшей степени, нежели инспекции, то он может охватывать больше материала. В то же время он не обладает такой эффективностью обнаружения и документирования дефектов, как инспекции.

Экспертные оценки

Экспертные оценки требуют немного больше, чем просто передача рабочего продукта коллеге по работе и выслушивание его мнения по этому поводу. Экспертная оценка может быть выражена словесно либо передана по электронной почте. Формальные процедуры экспертных оценок отсутствуют, эффективность поиска и документирования дефектов, свойственная этому методу, меньше, чем эффективность двух других методов.