Анализ и управление требованиями

Определение, виды, способы сбора и формализация

Определение понятия “требование”

Стандарт IEEE :

Требования к программной системе – это:

1. Функциональность, необходимая пользователю для решения проблемы или достижения цели.
2. Функциональность, которая должна быть получена (достигнута) системой или ее компонентами для соответствия контракту, стандарту, спецификации или другим формальным документам.
3. Документальное представление пп. 1 – 2.

Определение понятия “требование”

Стандарт ISO 12207:

Разработчик должен установить и документально оформить следующие требования к программным средствам:

- функциональные и технические требования, включая производительность, физические характеристики и окружающие условия, под которые должен быть, создан программный объект;
- требования к внешним интерфейсам программного объекта;
- квалификационные требования;
- требования безопасности, включая требования, относящиеся к методам эксплуатации, сопровождения, воздействию окружающей среды и травмобезопасности персонала;

и т.д.

Виды требований

Все требования разбиваются на три уровня:

1. *Бизнес-требования.* Бизнес-требования определяются целями и политикой организации их высказывают те, кто финансирует проект.
2. *Требования пользователей.* Определяют цели и задачи, которые позволит решить система, или что пользователи смогут делать с помощью системы. Пользовательские требования должны соответствовать бизнес-требованиям в противном случае их не следует включать в проект.
3. *Системные требования.* Определяют функциональность и характеристики системы, которую должны построить разработчики, для того чтобы пользователи смогли выполнить свои задачи (в рамках бизнес-требований).

Каждая система требований (бизнес-требования, требования пользователей и системные требования.) включает в себя функциональные и нефункциональные требования

Функциональные требования

Функциональные требования определяют функции, которые выполняет система, и зависят от потребностей пользователей и типа решаемой задачи.

Нефункциональные требования

Нефункциональные требования определяют характеристики и ограничения системы и не связаны непосредственно с функциональными требованиями.

Нефункциональные требования делят на:

требования к продукту

требования к процессу

внешние нефункциональные требования

Нефункциональные требования к продукту

Определяют его эксплуатационные качества, т.е. определяют то, насколько хорошо будет работать система. Часто такие характеристики называются *атрибутами* или *факторами качества программ*.

Стандарт ISO 9126 предлагает оценивать программную продукцию по **6 *характеристикам качества***, рекомендуя использовать **21 *показатель* (*подхарактеристику*) *качества***.

Атрибуты качества

Для пользователей

- производительность
- надежность и доступность
- безопасность
- удобство и простота обслуживания

Для разработчиков

- легкость сопровождения и эксплуатации
- повторное использование

Нефункциональные требования к процессу

определяют ограничения, связанные с использованием определенных технологий и стандартов разработки(язык программирования, методы разработки), и ограничения реализации (сроки изготовления)

Внешние нефункциональные требования

определяют взаимодействие проектируемой системы с другими системами, требования по квалификации персонала, юридические требования, логистические требования, требования среды, этические, экологические и т.п. требования.

Требования, которых быть НЕ ДОЛЖНО

Спецификация требований не должна содержать деталей проектирования или реализации (кроме известных ограничений). Иными словами, требования должны отвечать на вопрос: «что должна делать система», абстрагируясь от вопроса «как она это должна делать». Стремление принимать детальные проектные решения на этапе анализа требований – одно из типичных «ловушек», типичных для неопытных команд разработчиков.

Свойства требований

Основные характеристики

полнота

корректность

необходимость

осуществимость

приоритет

недвусмысленность

непротиворечивость (согласованность)

проверяемость

Выявление требований

Требования к ПС делят на:

требования, определяемые *предметной областью*;

требования, определяемые *пользователями* системы.

Методы выявления требований

1. **Опрос** (интервью)
    a. подготовка вопросов
    b. определение групп лиц
    c. проведение опроса
    d. определение последующих действий

Методы выявления требований

2. **Совместные семинары**

- определение правил
- наличие видения проекта
- темы обсуждения
- ограничения по времени
- отбор участников

Методы выявления требований

3. Мозговой штурм

- распределение ролей
- определение правил
- ограничения по времени
- голосование
- обработка результатов

Методы выявления требований

4. **Сценарии**

Сценарий – это способ описания структуры задачи, представляющий собой повествовательный рассказ о совершаемых действиях, происходящих в данных временных рамках и в данном контексте

Сценарий событий включает: ·

- Описание начального состояния системы.
- · Описание нормального протекания событий.
- · Описание исключительных ситуаций и способов их обработки.
- · Сведения о других действиях, которые можно выполнять во время реализации сценария.
- · Описание конечного состояния системы.

Методы выявления требований

Вариант использования описывает поведение системы при ее ответах на запрос одного из участников (пользователей) системы, называемого *основным действующим лицом*, в различных условиях

Различные модели поведения, или сценарии, развертываются в зависимости от определенных запросов и условий, при которых делались эти запросы. Вариант использования собирает вместе эти сценарии.

Способ представления вариантов использования зависит от знаний и умений людей (пользователей и разработчиков), средством связи между которыми, они служат. Это может быть текстовая форма, блок-схемы, диаграммы язык программирования и т.д.

При написании варианта использования нужно помнить о трех понятиях, которые служат основой при написании варианта и любого его предложения. Это следующие понятия:

1. *Область действия*, которая определяет, насколько велика разрабатываемая система.
2. *Основное действующее лицо* – участник, который обращается к системе, чтобы она обеспечила достижение его цели.
3. *Уровень*, который определяет, насколько высока эта цель.

Рекомендации

- Начните со стратегического варианта. От него будут ветвиться варианты использования уровня цели пользователя.
- Именуйте варианты использования с помощью коротких глагольных конструкций, объявляющих цель, которая должна быть достигнута.
- На каждом шаге четко определяйте действующее лицо и его намерения.
- Для описания альтернативного поведения применяйте расширения, а не предложения типа «если … то … иначе» в теле главного варианта использования.
- Для записи шагов варианта использования применяйте только предложения в настоящем времени, с глаголом действия в действительном залоге, и описывающее как действующее лицо успешно достигает цели, продвигающей весь процесс.

Применение модели MSC UML

Диаграмма использования призвана ответить на вопрос: что делает система во внешнем мире?

Методы выявления требований

5. Выявление требований на основе различных точек зрения. Метод VORD

Точка зрения – это источник информации о системных данных. В этом случае точка зрения – основа для построения модели создания и использования данных в системе.

Точка зрения – это особая часть модели системы и на основе различных точек зрения могут быть построены, например, модели сущность-связь или модели конечного автомата системы.

Точка зрения – это получатель системных сервисов. В этом случае точка зрения (точнее ее носитель) является внешним по отношению к системе и помогает определить данные, необходимые для выполнения системных сервисов и управления ими.

Метод VORD состоит из четырех основных этапов:

1. Идентификация точек зрения (“мозговой штурм”)
2. Структурирование точек зрения.
3. Документирование точек зрения.
4. Отображение системы точек зрения.

Идентификация точек зрения

решение следующих задач:

- идентификация потенциальных опорных точек зрения;
- идентификация системных сервисов;
- определение входных системных данных;
- определение нефункциональных требований;
- выявление управляющих событий и исключительных ситуаций

Результат: документ, идентифицирующий ОТЗ и сервисы системы

Структурирование

- определение связей сервисов с точками зрения
- определение иерархии наследования точек зрения

Документирование:

- детализация
- выделение входных, выходных и управляющих данных
- построение схемы

Методы выявления требований

6. Этнографический подход

Для определения среды функционирования системы и учета ее в требованиях разработчик должен «погрузиться» в эту среду, каждодневно наблюдая и фиксируя все реальные действия, выполняемые (потенциальными) пользователями, такой подход называется *этнографическим*

Анализ требований

Анализ требований (Requirement Process)

SWEBOK:

- Requirements Elicitation (Извлечение требований),
- Requirements Analysis (Анализ требований в узком смысле),
- Requirements Specification (Специфицирование требований),
- Requirements Validation (Проверка требований).

Анализ требований (Requirement Process)

]. RUP :

- § Analyze the Problem (Анализ проблемы),
- § Understand Stakeholder Needs (Понимание потребностей совладельцев),
- § Define the System (Определение системы),
- § Manage the Scope of the System (Управление контекстом системы),
- § Refine the System Definition (Уточнение определения системы).

Анализ требований (Requirement Process)

- Формирование видения;
- Выявление требований;
- Классификация и спецификация требований;
- Расширенный анализ требований (моделирование и прототипирование);
- Документирование требований;
- Проверка требований;
- Управление требованиями;
- Совершенствование процесса работы с требованиями.

Цели АТ

- добиться одинакового понимания с заказчиком и пользователями о том, что должна делать система;
- дать разработчикам наилучшее понимание требований к системе;
- определить границы системы;
- определить интерфейс пользователя и системы.

Результат АТ

набор артефактов

Это могут быть текстовые документы, модели UML, либо других языков моделирования, прототипы программного обеспечения.

Кто и для кого?

Специалист по АТ – постановка задачи, определение рамок проекта,

Представитель заказчика – постановка задачи, определение рамок проекта, контроль работы исполнителя, приёмка результатов работы.

Архитектор системы – разработка архитектуры, проектирование подсистем

Программист – разработка программного кода.

Тестировщик – составление тест-плана, тестовых сценариев.

Менеджер проекта – планирование и контроль исполнения работ.