**Требования. Варианты использования. Сценарии**

В предыдущих лекциях мы обсуждали оценки пользовательского интерфейса, и одним их показателей качественного интерфейса – это положительный ответ на вопрос: Достиг ли пользователь своих целей при использовании данной программной системой? Задачу специфицирования целей и потребностей пользователей на ранних этапах проектирования разработчики решают на этапе извлечения требований.

**Требования – это то, для чего предназначен и должен быть создан программный продукт, или аппаратный продукт, или услуга, или что либо иное, что мы собираемся построить или разработать.**

**Требования существуют независимо от того, обнаружили ли их или нет, записаны ли они или нет.**



Очевидно, что продукт никогда не будет правильным, если он не соответствует требованиям, поэтому таким образом можно думать о требованиях как о некоем естественном законе, и задача разработчика их обнаружить. Тем не менее, деятельность по требованиям не связана главным образом с написанием документа о требованиях. Деятельность по выявлению требований сосредотачивается главным образом на понимании бизнес-проблемы и предоставлении решения для неё.

**Настоящее искусство обнаружения требований – это обнаружение настоящей проблемы.**

Как только это будет сделано, будет и основа для выявления требований и выбора между альтернативными решениями.

**Как правило, требования представляют собой систему текстов, видов и моделей. Одним из таких видов является диаграмма вариантов использования или use-case диаграмма.**

Ни одна грамотно построенная система не существует в изоляции: она взаимодействует с действующими лицами (людьми или системами), которые используют её для достижения некоторой цели, ожидая от неё определённого поведения.

Вариант использования специфицирует это ожидаемое поведение субъекта (системы или её части), – он описывает **последовательности действий**, включая их варианты, которые субъект осуществляет для достижения действующим лицом определённого результата.

Варианты использования применяются для выражения требуемого поведения разрабатываемой системы, без описания реализации этого поведения. Они позволяют разработчикам, конечным пользователям и экспертам в предметной области достичь взаимопонимания, а кроме того, помогают удостовериться в правильности архитектурных решений и проверять систему по ходу её разработки.

В процессе создания системы варианты использования реализуются с помощью коопераций, элементы которых работают совместно для достижения целей каждого из них. Хорошо структурированные варианты использования описывают только существенные аспекты поведения и не являются ни слишком обобщёнными, ни чересчур подробными [Буч Г., Рамбо Д., Якобсон И. Язык UML. Руководство пользователя].

Хорошо структурированные варианты использования не являются, ни слишком обобщёнными, ни чересчур подробными.

Традиционно рассмотрим определения нового понятия «вариант использования».

Вариант использования – это описание возможных последовательностей взаимодействий между рассматриваемой системой и внешними участниками, связанные с конкретной целью.

**Use Case (вариант использования, ВИ, Прецедент, юскейс)** — это сценарная техника описания взаимодействия. С помощью Use Case может быть описано и пользовательское требование, и требование к взаимодействию систем, и описание взаимодействия людей и компаний в реальной жизни.

Отсутствует тот факт, что система несёт ответственность перед заинтересованными сторонами системы, а также ответственность является частью работы по написанию требований.

Вариант использования – это описание взаимодействий и обязанностей системы, «обсуждаемой системы» или «проектируемой системы», с внешними агентами или субъектами.

Варианты использования охватывают только «поведенческие» аспекты системы, то есть функциональные требования. Бизнес-правила, глоссарий, целевые показатели производительности, требования к процессам и многие другие вещи просто не попадают в категорию поведения.

Названия вариантов использования обеспечивают потрясающую структуру, на которой можно повесить многие другие детали проекта, такие как приоритет, группировка, даты выпуска, статус, требования к производительности и интерфейсные каналы. Они также связывают воедино куски информации, которые в противном случае было бы трудно связать друг с другом, такие как характеристики пользователя, бизнес-правила и требования к формату данных. Пока они не находятся в вариантах использования напрямую, они связаны с вариантами использования. Варианты использования выступают в роли ступицы колеса, а другие, различные виды информации действуют как спицы, ведущие в разных направления.

Именно по этим причинам варианты использования часто рассматривают как **центральный элемент** требований.

|  |
| --- |
| **Вариант использования** – это формулировка цели, которую основной участник имеет в отношении заявленных обязанностей системы, и сбор возможных сценариев между обсуждаемой системой и внешними участниками, показывающий, как цель основного участника может быть достигнута или **может быть неудачной**.  **Сценарий** – это последовательность взаимодействий, которые происходят при определённых условиях, с целью достижения цели основного действующего лица и достижения определённого результата в отношении этой цели. Взаимодействия начинаются с инициирующего действия и продолжаются до тех пор, пока цель не будет достигнута или отменена, и система выполняет все свои обязанности в отношении взаимодействия. |

Варианты использования — это вид документации, который можно использовать в работе в различных ситуациях, например, если требуется:

* описать рабочий процесс в бизнесе.
* сконцентрировать усилия на обсуждении принципиальных требований к разрабатываемой системе, а не на подробном их описании.
* описать функциональные требования к системе.
* документировать проект системы.
* написать документ в небольшой компактной группе или в большой распределённой группе.

**ВИ фиксирует соглашение между участниками системы о её по­ведении. ВИ описывает поведение системы при её ответах на за­прос одного из участников, называемого основным действующим лицом, в раз­личных условиях.**

Основное действующее лицо инициирует взаимодействие с сис­темой, чтобы добиться некоторой цели. Система отвечает, соблюдая интересы всех участников. Различные модели поведения, или сценарии, развёртываются в зави­симости от определённых запросов и условий, при которых делались эти запросы. Вариант использования собирает вместе эти сценарии.

ВИ представлены большей частью в текстовой форме, хотя возможны блок-схемы, циклограммы, сети Петри или языки программирова­ния. При нормальных обстоятельствах они служат средством связи между лицами, часто не имеющими специальной подготовки. Поэтому простой текст обычно являет­ся наилучшим выбором.

Вариант использования как форма описания стимулирует обсуждение проекти­руемой системы в группе разработчиков. Одна команда разработчиков может с по­мощью варианта использования документировать действительные требования, другая — окончательный проект. Все вышеперечисленное может применяться как для большой системы масштаба компании, так и для небольшой системы, например для фрагмента прикладной программы. Одинаковые основные правила создания ва­риантов использования применимы в любой ситуации, даже если документация име­ет различные уровни детализации и технические подробности.

Когда варианты использования документируют бизнес-процессы организации, рассматриваемая система и является этой органи­зацией. Участники — это акционеры компании, заказчики, поставщики, органы го­сударственного управления. Основные действующие лица — это заказчики компании и, возможно, их поставщики.

Хорошо написанный ВИ легко читается. Он состоит из пред­ложений, написанных в единой грамматической форме (простых шагов действий), в результате которых действующее лицо достигает цели или передает информацию другому действующему лицу. Описание ВИ – это, по сути, упражнение в написании эссе на естественном языке со всеми трудностями в формулировании «хорошего», которое приходит с естественным языком прозаического письма вообще. Главное, что сценарий использования не обязательно должен быть «лучшим», чтобы быть «полезным».

**Для написания хорошего варианта использования необходимо освоить три понятия, которые применяются к каждому предложению варианта использования и ко всему ВИ в целом**

* **Область действия (Scope): какова на самом деле рассматриваемая система?**
* **Основное действующее лицо (Primary actor): у кого есть цель?**
* **Уровень (Level): какой уровень имеет эта цель?**

Для дальнейшего разговора о ВИ необходимо освоить следующие понятия

1. Действующее лицо (Actor) : кто-то (или что-то), обладающий поведением.
2. Участник (Stakeholder): кто-то (или что-то), проявляющий интерес к поведению рассматриваемой системы.
3. Основное действующее лицо (Primary actor): участник (некто или нечто), инициирующий взаимодействие с рассматриваемой системой для достижения некоторой цели.
4. Вариант использования (Use Case): соглашение относительно поведения рассматриваемой системы.
5. Область действия (Scope): идентифицирует рассматриваемую систему.
6. Предусловия и гарантии (Pre-condition and guarantees): то, что должно быть истинным до и после реализации варианта использования.
7. Основной сценарий (Main success scenario): вариант, в котором не **возникает никаких ошибок.**
8. **Расширения (Extension): различные отклонения от основного сценария.**
9. **Номера расширений соответствуют номерам шагов основного сценария, в которых обнаруживаются данные отклонения (например, шаги 4a и 4b указы­вают на две отличные от основного сценария ситуации, которые могут возник­нуть на шаге 4).**

**Действующее лицо (Актор (Actor))**

Вариант использования представляет собой текстовое описание шагов, которые выполняет действующее лицо (actor).

Акторы – внешние субъекты, взаимодействующие с Системой

Актор – это что-либо (кто-либо) имеющее поведение. Актором может быть человек, компания или организация, компьютерная программа или компьютерная система, аппаратное или программное обеспечение или и то, и другое.

Более правильное слово будет **«роль»**.

**«Роль»** будет означать роль, которую играет конкретный человек, когда он использует систему, например, «создатель счета», «получатель заказа» или «менеджер». В идеале мы должны говорить о «типе актора» или «роли», чтобы указать, что мы имеем в виду категорию человека, возможно, описание должности («клерк») или отношения («клиент»).

Сама рассматриваемая система – особый актор. Мы обычно называем его по имени или называем «обсуждаемая система», «проектируемая система». Это не будет первичный актор или вторичный актор. Это просто актор, которым мы можем воспользоваться в определенные моменты.

**Подсистемы СУ, как внутренние субъекты**. Отдельный объект в программном обеспечении также является внутренним действующим лицом. Обычно мы намеренно пишем сценарии использования так, чтобы SuD был неоткрытым (черным) ящиком, в который мы не можем заглянуть внутрь. Внутренние участники тщательно не упоминаются, так как обычно они не известны на момент написания требований, или мы хотим специально исключить детали дизайна.

**Основной субъект варианта использования.** Единственный первичный субъект варианта использования - это внешний субъект, цель которого пытается удовлетворить вариант использования, - субъект, действие которого инициирует действие варианта использования.

**Заинтересованные стороны или Стейкхолдеры (Stakeholders)**

Заинтересованная сторона - это человек, который заинтересован в поведении варианта использования, даже если он никогда не взаимодействует напрямую с системой. Каждый основной участник, конечно, является заинтересованным лицом. Но есть заинтересованные стороны, которые не имеют особых случаев использования для себя. Они могут вообще никогда не взаимодействовать напрямую с системой, но все же имеют право заботиться о том, как система ведет себя. Примеры включают владельцев системы или компании и регулирующие органы. Бизнес-правила отражают интересы различных заинтересованных сторон в вариантах использования.

Совет директоров банка не взаимодействует напрямую с банкоматом в стене, кроме как с обычными клиентами. Служба внутренних доходов не вводит в банкомат ничего особенного, так же как Департамент страхования федерального правительства не вводит ничего особенного в компьютер страховой компании.

Заинтересованные стороны не отображаются непосредственно в тексте варианта использования. Вариант использования - это описание поведения, и заинтересованная сторона не действует в случае использования, кроме как в роли первичного или вторичного субъекта. Интересы заинтересованных сторон отражаются в поведении системы, проверках и проверках, которые она выполняет, журналах, которые она создает, и действиях, которые она выполняет. Все это, как говорится, заинтересованные стороны важны для требований системы. Вариант использования это форма контракта, в которой согласовываются возможные конфликтующие интересы всех заинтересованных сторон. Указанное поведение обсуждаемой системы должно охватывать взаимоприемлемый набор их особых интересов. Достаточно освоить две идеи:

* + Заинтересованная сторона не является основным участником, если только эта заинтересованная сторона / субъект не взаимодействует напрямую с системой в этом качестве.
  + Поведение проектируемой системы должно защищать интересы заинтересованных сторон, которые не присутствуют для защиты своих интересов.

Мы часто обнаруживаем, что в ИТ-отделе есть сценарии использования компьютерной безопасности, в отделе маркетинга - сценарии использования рекламы, а у клиента - сценарии использования основных функций системы.

Вариант использования описывает различные наборы взаимодействий, которые могут происходить между различными внешними агентами или субъектами, в то время как основной Актор преследует эту цель. В нем также описываются обязанности проектируемой системы, не вдаваясь в методы реализации или компоненты системы. Каждая возможная последовательность взаимодействий называется сценарием. Вариант использования собирает все сценарии, связанные с этой целью этого основного субъекта, в том числе тех, в которых цель достигнута, и тех, в которых цель должна быть оставлена.

|  |
| --- |
| **Покупка ценных бумаг через Интернет**  **Основное действующее лицо:** покупатель  **Область действия:** персональные консультанты / финансовый пакет (РАР)  **Уровень:** цель пользователя  **Участники и интересы:**  Покупатель — хочет купить ценные бумаги, причем они должны автоматически попасть в портфель PAF.  Биржевое агентство — хочет получить полную информацию о покупке.  **Предусловие:** программа PAF у пользователя уже открыта.  **Минимальные гарантии:** в наличии достаточно регистрационной информации, чтобы PAF могла обнаружить несоответствие и запросить у пользователя дополнительные данные.  **Гарантия успеха:** удаленный Web-сайт подтвердил покупку; регистрационные файлы и портфель пользователя обновлены.  **Основной сценарий:**   1. Покупатель выбирает покупку ценных бумаг через Сеть. 2. PAF получает от пользователя адрес нужного сайта. 3. PAF подключается к сайту, сохраняя контроль над процессом. 4. Покупатель выбирает и покупает ценные бумаги на данном сайте. 5. PAF перехватывает ответы Web-сайта и обновляет портфель покупателя. 6. PAF показывает покупателю новое состояние портфеля.   **Расширения:**  2a. Покупатель запросил Web -сайт, не поддерживаемый PAF.  2a1. Система получает от покупателя новое предложение с возможностью отменить вариант использования.  3a. Отказ любого рода в Сети во время установки:  3a1. Система сообщает покупателю о неудаче, даёт совет и возвращается на предыдущий шаг.  3a2. Покупатель либо отказывается продолжать этот вариант использования либо делает новую попытку.  4а. Во время транзакции покупки компьютер выходит из строя или выключается:  4а1. (Что здесь нужно делать?)  4b.web-сайт не подтверждает покупку, а задерживает ее:  4b1. PAF регистрирует задержку, устанавливает таймер, чтобы запросить покупателя о выходе.  5а. web-сайт не возвращает необходимую информацию о покупке:  5а1. PAF регистрирует отсутствие информации о том, совершилось ли в портфеле покупателя обновление для зависшей покупки. |

|  |
| --- |
| Вариант использования 2  **Получить страховую компенсацию за автомобильную аварию**  **Основное действующее лицо:** истец  **Область действия:** страховая компания (MyInsCo)  **Уровень:** обобщенный  **Участники и интересы:**  Истец — хочет получить максимально возможную сумму.  Компания MylnsCo — хочет заплатить как можно меньше.  Министерство страхования — хочет убедиться, что соблюдены все нормы и правила.  **Предусловие:** отсутствует.  **Минимальные гарантии:** MylnsCo регистрирует заявление и все действия.  **Гарантия успеха:** истец и MylnsCo приходят к соглашению о сумме страхового возмещения. Истец получает оговоренную сумму.  **Триггер:** истец представляет заявление на рассмотрение.  **Основной сценарий:**   1. Истец представляет на рассмотрение заявление с обоснованием. 2. Страховая компания проверяет законность страхового полиса истца. 3. Страховая компания назначает агента для расследования страхового случая. 4. Страховая компания проверяет, укладываются ли все детали в нормативы полиса. 5. Страховая компания выплачивает истцу страховое возмещение и закрывает дело.   **Расширения:**  1a. Предоставленные данные не полны:  1a1. Страховая компания запрашивает недостающую информацию.  1a2. Истец предоставляет недостающую информацию.  2а. Полис истца недействителен:  2a1. Страховая компания отклоняет заявление, извещает истца, документирует все действия и прекращает дело.  За. На этот момент нет свободных агентов.  За1. (Что в этом случае делает страховая компания?)  4а. Обстоятельства аварии противоречат основным правилам полиса:  4а1. Страховая компания отклоняет заявление, извещает истца, документирует все действия и прекращает дело.  4b. Обстоятельства аварии противоречат второстепенным правилам полиса:  4b1. Страховая компания начинает переговоры с истцом относительно суммы платежа. |
|  |

Этот вариант использования кажется коротким и простым, но, очевидно, его можно развернуть на большое количество более низких уровней. Значение, обеспечиваемое наличием взаимодействия, описанного на этом уровне, заключается в том, что оно легко просматривается и может быть расширено при необходимости. Шаги 2 и 4 приводят к деталям конструкции страховой компании.

**Требования и варианты использования**

Если вы пишете варианты использования как требования, имейте в виду:

*Это действительно требования.* Не следует превращать их в другую форму требований к поведению системы. Написанные надлежащим образом, они точ­но описывают, что должна делать система.

*Это не все требования.* Они не детализируют внешние интерфейсы, форма­ты данных, бизнес-правила и сложные формулы. Они устанавливают только часть (возможно, треть) всех требований, которые вам необходимо собрать, очень важную, но лишь часть.

**Схема требований**

|  |
| --- |
| Раздел 1. Цель и область действия  la. Что представляют собой общая область действия и цель?  lb. Участники (Кого это интересует?)  lc. Что входит в область действия и что нет?  Раздел 2. Используемые термины/Глоссарий  Раздел 3. Варианты использования  За. Основные действующие лица и их общие цели  3b. Варианты использования для бизнес-процессов  3c. Системные варианты использования  Раздел 4. Используемая технология  4a. Какие технологические требования предъявляются к данной системе?  4b. С какими системами будет взаимодействовать данная, каковы требования?  Раздел 5. Другие требования  5a. Процесс разработки  Q1. Кто участвует в проекте?  Q2. Какие оценки проекта будут отражены (простой, ранний, быстрый или гибкий)?  Q3. Какая обратная связь или прозрачность проекта нужна пользователям и организаторам?  Q4. Что мы можем купить, что должны построить, с кем конкурируем?  Q5. Какие еще существуют технологические требования (тестирование, установка и т.д.)?  Q6. От чего зависит развитие проекта?  5b. Бизнес-правила  5c. Производительность  5d. Эксплуатация, безопасность, документация 5е. Использование (простота использования)  5f. Сопровождение и мобильность  5g. Нерешенные или отложенные вопросы  Раздел 6. Людские резервы, правовые, политические, организационные вопросы  Q1. Как влияют людские резервы на функционирование системы?  Q2. Какие существуют правовые и политические требования?  Q3. Какие последствия для людей будет иметь создание этой системы?  Q4. Каковы требования к обучению?  Q5. Какое влияние оказывает система на окружающее сообщество? |

Вариантам использования посвящен только Раздел 3 требований. Это не во­обще все требования, а только требования к поведению. Бизнес-правила, глосса­рий, производительность, технологические требования и многое другое не попадает в категорию поведения. Для этих аспектов нужны собственные разделы.

В простейшем случае взаимодействие - это просто отправка сообщения. В программном обеспечении вызов функции, такой как «Печать (значение)», является простым взаимодействием. Это просто одно сообщение.

Взаимодействие также может быть последовательностью взаимодействий. Мы идем к машине кока-колы и вставляем две четверти, только чтобы сказать, что нам нужны точные изменения. Наше «взаимодействие» с машиной было следующим:

* покупатель вставляет квартал
* покупатель вставляет квартал
* покупатель нажимает колу
* машина говорит: «Требуется точное изменение»
* покупатель ругается, толкает монету
* машина возвращает две четверти
* покупатель занимает четверть

Простая последовательность действий состоит из простых взаимодействий или отдельных сообщений.

Однако он также может состоять из последовательностей, сложенных вместе. Всю вышеприведенную последовательность можно объединить в одну фразу: «Я пытался купить кока-колу в автомате (но она требовала точных изменений)». Такое сведение воедино последовательностей полезно для сокращения описания длительного взаимодействия.

До сих пор у нас было, что сценарий использования представляет собой набор возможных сценариев взаимодействия для достижения цели.

Нам нужно добавить два предложения:

* + Все взаимодействия связаны с одной и той же целью.
  + Взаимодействия начинаются с инициирующего события и заканчиваются, когда цель достигнута или отменена, и система выполняет свои обязанности в отношении взаимодействия.

|  |
| --- |
| **Вариант использования: получить оплату за автомобильную аварию**  **Сфера дизайна**: страховая компания («MyInsCo»)  **Основной участник:** истец  **Главный сценарий успеха**  1. Истец подает претензию с обоснованием данных.  2. Страховая компания подтверждает, что истец владеет действующим полисом  3. Страховая компания назначает агента для рассмотрения дела  4. Агент проверяет, что все детали соответствуют принципам полиса  5. Страховая компания платит истцу  **Расширения:**  1a. Представленные данные являются неполными:  1a1. Страховая компания запрашивает недостающую информацию  1a2. Истец предоставляет недостающую информацию  2а. Истец не имеет действительного полиса:  2a1. Страховая компания отклоняет претензию, уведомляет истца, фиксирует все это, прекращает производство по делу.  3a. На данный момент нет доступных агентов  3a1. (Что здесь делает страховая компания?)  4а. Авария нарушает основные принципы полиса:  4a1. Страховая компания отклоняет претензию, уведомляет истца, фиксирует все это, прекращает производство по делу.  4b. Несчастный случай нарушает некоторые незначительные правила полиса:  4b1. Страховая компания начинает переговоры с заявителем относительно степени оплаты. |

Этот вариант использования кажется коротким и простым, но, очевидно, его можно развернуть на большое количество более низких уровней. Значение, обеспечиваемое наличием взаимодействия, описанного на этом уровне, заключается в том, что оно легко просматривается и может быть расширено при необходимости. Шаги 2 и 4 приводят к деталям конструкции страховой компании.

## Пример use case (https://habr.com/ru/post/468267/)

Как выглядит сценарий, на примере авторизации на сайте через email.

|  |
| --- |
| (Системный) Авторизоваться на сайте для доступа в личный кабинет. ~~ (уровень моря)  **Контекст:** Не авторизованный клиент авторизуется на сайте, чтобы сайт его узнал и показывал персональную для него информацию: историю просмотров, покупок, текущее количество бонусных баллов и пр, используя email как логин.  **Уровень:** цель пользователя **Основное действующее лицо:** клиент (посетитель нашего интернет-магазина) **Область действия:** Взаимодействие клиента с сайтом интернет-магазина **Заинтересованные лица и интересы:**   * маркетолог хочет, чтобы максимальное число посетителей сайта были идентифицированы для большего охвата персональных рассылок, * специалист безопасности хочет чтобы не было случаев несанкционированного доступа к персональным данным посетителя, включая попытки подбора пароля для одного аккаунта или поиска аккаунта с слабым паролем, * злоумышленник хочет получить доступ к бонусам жертвы, * конкуренты хотят оставить негативные отзывы на товары, * ботнет хочет получить базу клиентов магазина и с помощью атаки добиться неработоспособности сайта.   **Предусловия:** посетитель должен быть не авторизован. **Минимальные гарантии:** посетитель узнает факт успешной или неуспешной попытки авторизации. **Гарантии успеха:** посетитель авторизован.  **Основной сценарий:**   1. Клиент запускает авторизацию. 2. Система подтверждает, что клиент не авторизован и нет превышения количества неуспешных попыток авторизации с данной сессии (поиск слабого пароля у множества аккаунтов) по «Правилу безопасности №23». 3. Система увеличивает счетчик количества попыток авторизации. 4. Система отображает клиенту форму авторизации. 5. Клиент вводит email и пароль. 6. Система подтверждает наличие клиента с таким email в системе и совпадение пароля и не превышение количества попыток входа в данный аккаунт по «Правилу безопасности №24». 7. Система авторизует клиента, добавляет историю просмотра и корзину данного сеанса с последним сеансом этого аккаунта клиента. 8. Система отображает сообщение успешности авторизации и переходит на шаг сценария, из которого клиент прервался на авторизацию. При этом данные на странице перегружаются с учетом персональных данных аккаунта.   **Расширения (Альтернативные сценарии) :**  2.a. Клиент уже авторизован:   2.а.1. Система уведомляет клиента о факте осуществленной ранее авторизации и предлагает либо прервать сценарий, либо перейти к шагу 4, при этом если шаг 6 успешно пройден, то 7 шаг выполняется с уточнением:  2.а.7. Система деакторизует клиента под старым аккаунтом, авторизует клиента под новым аккаунтом, при этом история просмотра и корзина данного сеанса взаимодействия остается в старом аккаунте, в новый не переходит. Далее переход к шагу 8.  2.b Количество попыток авторизации превысило порог по «Правилу безопасности №23»:  2.b.1 Переход к шагу 4, на форме авторизации дополнительно отображается капча  2.b.6 Система подтверждает корректный ввод капчи  2.b.6.1 Капча введена неправильно:  2.b.6.1.1. система увеличивает счетчик неуспешных попыток авторизации и под данный аккаунт  2.b.6.1.2. система отображает сообщение о неуспешности и возвращается к шагу 2  6.a. Аккаунта с данным email не обнаружено:  6.а.1 Система показывает сообщение о неуспешности и предлагает на выбор, либо переходод к шагу 2, либо переход к сценарию «Регистрация пользователя» с сохранением введенного email,  6.b. Пароль от аккаунта с данным email не совпадает с введенным:  6.b.1 Система увеличивает счетчик неуспешных попыток входа в данный аккаунт.  6.b.2 Система отображает сообщение о неуспешности и предлагает на выбор, либо переход к сценарию «Восстановление пароля» либо переход к шагу 2.  6.c: Счетчик попыток входа в данный аккаунт превысило порог по «Правилу безопастности №24».  6.c.1 Система отображает сообщение о блокировке входа в аккаунт в течении X минут и переходит к шагу 2. |

**Рекомендации по написании Use Case**

1. Акторы и цели. Перечислите, каких участников и каких целей будет поддерживать система. Просмотрите этот список на предмет точности и полноты. Расставляйте приоритеты и назначайте командам и релизам. Теперь у вас есть функциональные требования к первому уровню или 1-битной точности.

2. Основные сценарии успеха. Для выбранных вами сценариев использования напишите заинтересованные стороны, триггер и основной сценарий успеха. Их вы можете просмотреть в черновом варианте, чтобы убедиться, что система действительно отвечает интересам заинтересованных сторон. Это 2-й уровень точности по функциональным требованиям. Это довольно легкий материал для черчения, в отличие от следующих двух шагов, которые занимают много энергии.

3. Условия отказа. Мозговой штурм всех неудач, которые могут произойти. Составьте этот список полностью, прежде чем понять, как система должна обрабатывать их все. Еще раз, заполнение следующего шага, обработка ошибок, потребует гораздо больше энергии, чем просто перечисление ошибок. Люди, которые сразу начинают писать обработку ошибок, часто испытывают недостаток энергии, необходимой для завершения перечисления всех условий отказа.

4. Обработка ошибок. Наконец, напишите, как система должна реагировать на каждый сбой. Это часто сложная, утомительная и удивительная работа. Это удивительно, потому что, довольно часто, вопрос о неясном бизнес-правиле всплывает во время этого письма. Или обработка ошибок неожиданно покажет нового участника или новую цель, которую необходимо поддерживать.

Прочие же, особенно исследователи и разработчики CASE (Computer Aided Software Engineering) средств, посчитали, что неформальные артефакты будут слишком неполными, и что им нужен математический базис и поддержка в CASE средствах. Они разработали специальные языки, нотации и программы, с помощью которых варианты использования превратились в строгие и скрупулезные описания. Этот подход снискал куда меньше последователей. Людям был нужен простой способ выражения своих соображений относительно будущей системы. Они не хотели использовать формальные программные средства. Более того, им казалось, что дополнительная работа по формализации вариантов использования не дает ощутимых результатов.

Формалисты столкнулись с ещё одной проблемой: как учесть все варианты поведения, с которыми может столкнуться система? Как-то меня спросили: «Если я разрабатываю автомат для продажи конфет, как мне специфицировать, какие монетки засунет туда покупатель? Он может положить три монетки по 25 центов, а может и 15 пятицентовиков. Или сначала положит 25 центов, а потом десять пятицентовиков? Или в обратном порядке? Или еще как-нибудь?» Я ответил: «Нужно написать: человек кладет в автомат деньги» (А. Кокберн, 1999).

**Цели действующего лица**

При описании вариантов использования очень важно связать их с целями, которые ставят перед собой действующие лица системы. Это не даст свести варианты использования к описанию функциональности системы, о которой в первую очередь пекутся программисты. Нет, подходить надо с другой стороны: что именно хочет от системы пользователь? Это и есть причина, по которой он взаимодействует с программой. Если программа дает пользователю возможность достичь поставленной цели, то она тем самым приносит ему максимальную пользу.

Получается, что вариант использования должен представлять собой структуру из двух частей:

* описание последовательности действий, когда все идет, как надо,
* и небольшие описания того, как ведет себя система, когда происходит какой-то сбой.

В общем случае, с помощью Use Case может описываться взаимодействие двух или большего количества участников, имеющее конкретную цель:

* покупка товара в магазине (Покупатель-Продавец);
* отправка письма по электронной почте (Адресант-Почтовый клиент);
* запрос страницы браузером (Браузер-Web-сервер).

**Примеры**

**Описание функции:** Зарегистрироваться на сайте. Сайт должен предоставлять пользователю возможность зарегистрироваться. Для регистрации пользователь должен заполнить форму. После регистрации сайт должен отправить на e-mail пользователя подтверждение о регистрации.

Та же функция только в формате Use Case (UC). Зарегистрироваться на сайте.

|  |
| --- |
| Структура ВИ  Цель  Действующие лица  Предусловия  Максимальные и минимальные гарантии  Основной сценарий  Расширения (альтернативные сценарии) |

**Нехороший пример :**

|  |
| --- |
| **Действующее лицо:** пользователь сайта  **Предусловия:** пользователь находится на главной странице сайта  Пользователь нажимает кнопку «Зарегистрироваться».  Сайт отображает форму регистрации.  Пользователь заполняет поля формы и подтверждает регистрацию.  Сайт проверяет правильность заполнения формы.  Если поля заполнены неверно, то сайт сообщает пользователю об ошибках заполнения формы.  Если поля заполнены верно, то сайт регистрирует пользователя и отправляет на его e-mail письмо с подтверждением регистрации.  Пользователь нажимает кнопку «Зарегистрироваться».  Сайт отображает форму регистрации.  Пользователь заполняет поля формы и подтверждает регистрацию.  Сайт проверяет правильность заполнения формы.  Если поля заполнены неверно, то сайт сообщает пользователю об ошибках заполнения формы.  Если в базе уже есть пользователь с таким же e-mail , то сайт сообщает пользователю о том, что пользователь с таким e-mail уже зарегистрирован.  Если поля заполнены верно, то сайт регистрирует пользователя и отправляет на его e-mail письмо с подтверждением регистрации. |

***Исправленный вариант использования***:

|  |
| --- |
| **Действующее лицо:** пользователь сайта  **Предусловия:** пользователь находится на главной странице сайта  **Основной сценарий:**  1.a. Пользователь нажимает кнопку «Зарегистрироваться»  2.a.Сайт отображает форму регистрации.  3.a. Пользователь заполняет поля формы и подтверждает регистрацию.  3.a.1 Сайт подтверждает правильность заполнения формы.  3.a.2 Сайт регистрирует пользователя и отправляет на его e-mail письмо с подтверждением регистрации.  **Альтернативные сценарии:**  4a. Есть ошибки заполнения формы:  4.а.1. Сайт сообщает об ошибках заполнения формы.  4.а.2. Переход на шаг 3.  4б. Пользователь с таким e-mail уже зарегистрирован:  4.б.1. Сайт сообщает о том, что пользователь с таким e-mail уже зарегистрирован.  4.б.2. Переход на шаг 3. |

**Как у альтернативных сценариев могут быть так называемые расширения, которые также указываются в карточке ВИ.**

Расширения Альтернативные сценарии

|  |
| --- |
| 2-4а. Условие… :  2-4а1. Действие…  2-4а2. Действие…  \*а. Условие… :  \*а1. Действие…  \*а2. Действие… |

**В чем преимущество описания Use Case?**

1. Дают представление о поведении системы. Сравните с функцией: Сайт должен предоставить пользователю возможность зарегистрироваться
2. Понятны заказчикам и разработчикам
3. Позволяют описать множество альтернатив (исключений)
4. Список вариантов использования – перечень функциональности системы
5. Позволяют описывать функционал итеративно (перечень Use Cases - > Краткие описания - > Основные потоки -> Расширения)

Пример. **Регистрация пассажира на рейс.**

### registration

|  |
| --- |
| **Система:** Система регистрации пассажира на рейс  **Основное действующее лицо:** Пассажир  **Цель:** Зарегистрироваться на рейс  **Триггер:** Пассажир решает зарегистрироваться на рейс и заходит на страницу регистрации сайта  **Результат:** Информация о регистрации пассажира сохранена. У пассажира есть посадочный талон |

*Поведение элемента Use Case описывается потоком событий.*

|  |
| --- |
| **Основной поток событий** |
| |  |  |  |  | | --- | --- | --- | --- | | **№ шага** | **Действующее лицо** | **Шаг** | **Комментарий** | | 1 | Система | Запрашивает фамилию и код бронирования |  | | 2 | Пассажир | Вводит фамилию и код бронирования | Код бронирования:  7 символов, заглавные латинские и цифры.  Например: MTDTC1 | | 3 | Система | Проверяет, что приобретен билет на этот рейс на имя этого Пассажира |  | | 4 | Система | Сохраняет информацию о регистрации Пассажира на рейс |  | | 5 | Система | Подтверждает Пассажиру, что он зарегистрирован на рейс |  | | 6 | Система | Выводит для печати посадочный талон |  | | 7 | Пассажир | Отправляет посадочный талон на печать |  | |

### Пример текста варианта использования

|  |
| --- |
| **Покупка товара**  **Главный успешный сценарий:**  1. Покупатель просматривает каталог и выбирает товары для покупки.  2. Покупатель оценивает стоимость всех товаров.  3. Покупатель вводит информацию, необходимую для доставки товара (адрес, доставка на следующий день или в течение трех дней).  4. Система предоставляет полную информацию о цене товара и его до­ставке.  5. Покупатель вводит информацию о кредитной карточке.  6. Система осуществляет авторизацию счета покупателя.  7. Система подтверждает оплату товаров немедленно.  8. Система посылает подтверждение оплаты товаров по адресу элект­ронной почты покупателя.  **Расширения:**  За. Клиент является постоянным покупателем.  3a.1 Система предоставляет информацию о текущей покупке и ее цене,  а также информацию о счете.  3a.2: Покупатель может согласиться или изменить значения по умолча­нию, затем возвращаемся к шагу 6 главного успешного сценария.  6а. Система не подтверждает авторизацию счета.  . 1: Пользователь может повторить ввод информации о кредитной карте или закончить сеанс. |

**Варианты использования и поток событий**

**Вариант использования описывает, *что* делает система (или подсистема, или класс, или интерфейс),** но не указывает, *как* она это делает. В процессе моделирования всегда важно разделять внешнее и внутреннее представления. Можно специфицировать поведение варианта использования, описав поток событий в текстовой форме, понятной постороннему читателю. В описании должны присутствовать указание на то, как и когда вариант использования начинается и заканчивается, когда он взаимодействует с действующими лицами, какими объектами они обмениваются, а также упоминание основного и альтернативного потоков поведения.

Например, в контексте системы банкомата можно описать вариант использования «Проверка пользователя»:

* *основной поток событий:* вариант использования начинается, когда система запрашивает у клиента его персональный идентификационный номер (PIN). Клиент (Customer) вводит его с клавиатуры. Завершается ввод нажатием клавиши Enter. Затем система проверяет введенный PIN и, если он правилен, подтверждает ввод. Этим вариант использования заканчивается;
* *исключительный поток событий:* клиент может отменить транзакцию в любой момент, нажав клавишу Cancel. Это действие запускает вариант использования заново. Никаких изменений со счетом клиента не производится;
* *исключительный поток событий:* клиент может в любой момент до нажатия клавиши Enter стереть свой PIN и ввести новый;
* *исключительный поток событий:* если клиент вводит неправильный PIN, вариант использования перезапускается.

Если это происходит три раза подряд, система отменяет всю транзакцию и не позволяет данному клиенту работать с банкоматом в течение 60 с.

**Варианты использования и сценарии**

Обычно в начале работы вы будете описывать поток событий варианта использования в текстовой форме. Однако по мере уточнения требований к системе вам при изображении потоков понадобятся диаграммы взаимодействий, позволяющие представить их графически. Как правило, вы будете применять диаграмму последовательности для описания основного потока варианта использования, а ее вариации – для исключительных потоков.

Желательно отделять основной поток от альтернативных, поскольку вариант использования описывает не одну, а множество последовательностей, и выразить все детали интересующего вас варианта использования в виде одной последовательности невозможно. Например, в системе управления человеческими ресурсами присутствует вариант использования Hire employee (Нанять работника). Существует множество разновидностей этой основной бизнес-функции. Вы можете переманить работника из другой компании (наиболее общий сценарий), перевести сотрудника из одного подразделения в другое (что часто случается в транснациональных компаниях) или нанять иностранца (особый случай, регулируемый специальными правилами). Каждый из этих вариантов описывается своей последовательностью.

Hire employee описывает набор последовательностей, каждая из которых представляет один поток из всех возможных вариаций. Такая последовательность называется сценарием. *Сценарий* (scenario) – это конкретная последовательность действий, иллюстрирующих поведение. Сценарии по отношению к вариантам использования – то же самое, что экземпляры по отношению к классам, поскольку сценарий – это в основном один экземпляр варианта использования.

### Use Case для проектировщика интерфейсов

Профессиональный проектировщик интерфейсов всегда предпочитает ориентироваться на целеориентированные сценарии работы пользователя с системой, нежели на описание отдельных функций, к которым должен иметь доступ пользователь. Это позволяет проектировщику:

* делать специализированный интерфейс для каждой роли пользователя,
* выводить на первый план интерфейса элементы, соответствующие более приоритетным целям пользователя,
* делать интерфейс более лаконичным и простым для восприятия, увеличивая скорость обучения.

**Ограничения метода**

* 1. Use Case не обеспечивают полноту всех функциональных требований, если в систему должна быть заложена сложная бизнес-логика, т.е. обработка информации в системе зависит не только и не столько от действий пользователей, сколько от внутренних правил взаимодействия объектов. Например, работа с системами типа «тасктрекер» задается достаточно простыми и стандартными Use Case: «Создать задачу», «Назначить задачу», «Пометить задачу, как выполненную». Однако тасктрекеров существует огромное множество, и это оправдано тем, что в каждом есть свои возможности по заданию жизненных циклов задач, их типов и взаимосвязей. И эту внутреннюю логику работы с задачами нет смысла описывать в виде Use Case.
  2. То же касается бухгалтерских программ, систем поддержки принятия решения, профессиональных систем для проектирования и дизайна. Use Case важны для них, но не покроют и пятой части требований к функциональности.
  3. Большое количество различных интерфейсов пользователя для одной и той же функциональности. Это требование современности. Возьмем, например оплату коммунальных платежей. Пятнадцать лет назад было достаточно одного интерфейса, через который оператор вводил оплату. Сейчас же такие системы покидают рынок. Одна и та же система для одной и той же функциональности может предоставлять интерфейсы через инфомат, смартофон, ПК, планшетник. Это разные интерфейсы. Можно, конечно описывать юзкейсы для каждого интерфейса отдельно, но часто эффективней описать базовые правила, а для каждого из вариантов конечного устройства обслуживания или задать общие правила (для сенсорных устройств размер кнопки не менее, чем …) или разрабатывать прототипы интерфейсов. Прототипы интерфейсов - тоже модель. Весьма удобная.

**Против вариантов использования**

Авторы Экстремального Программирования остались верны идее неформальных сценариев, у которых нет никакой заданной структуры. Кент Бек (Kent Beck) создал термин «рассказ пользователя» (user story). Именно им и стали называть такие неформальные требования к системе. Рассказ пользователя состоит из нескольких фраз, записанных на маленькой бумажной карточке. В них пользователь объясняет, что именно он хочет от системы. В ХР рассказ пользователя представляет собой не требование к системе, а некое напоминание о будущем обсуждении этой функциональности с заказчиком. Таким образом, на карточке достаточно записать ровно столько информации, чтобы и заказчики, и программисты поняли, что именно им нужно будет обсудить в дальнейшем.

Ларри Константайн (Larry Constantine) рассматривает описание сценариев с точки зрения проектирования пользовательского интерфейса. Он обнаружил, что для этого ему совершенно не нужно описывать внутреннее поведение системы или ее взаимодействие с второстепенными действующими лицами. Для проектирования интерфейсов вполне достаточно текста, записанного в две колонки. В первой пишут то, что пользователь пытается сделать, во второй - как реагирует на его действия система. Такая простейшая структура позволяет создать интерфейс, отвечающий задачам пользователя. Чтобы избежать путаницы, Константайн переименовал «вариант использования» (use case) в «вариант задачи» (task case), так как варианты использования традиционно представляют собой спецификацию системы.

Существует три уровня детализации в описании вариантов использования: краткий, обычный и полный.

*Краткое описание* представляет собой от двух до четырех предложений, которыми определяется весь вариант использования. Он хорошо умещается в ячейке электронной таблицы (в других ячейках вы можете описать приоритетность, технические особенности реализации, порядковый номер версии и прочую информацию, касающуюся планирования).

*Обычное описание* состоит уже из нескольких абзацев текста, в которых дается вся вышеописанная информация.

*Полное описание* - это вместительный шаблон, в котором есть поля для описания заинтересованных сторон, минимальных гарантий, постусловий, правил и регламентов, ограничений производительности и т.д.

**Не для дизайна интерфейса**

Варианты использования не стоит применять для описания пользовательского интерфейса, несмотря на то, что их форма и позволяет это делать.

* Во-первых, вариант использования - это документ, описывающий требования к системе, а дизайн пользовательского интерфейса производится опытными специалистами, которым нужно сначала объяснить, что и как должна делать система.
* Во-вторых, дизайн интерфейса - очень непостоянная материя, он часто меняется. Если описывать интерфейс с помощью вариантов использования, то получится, что это описание нужно будет очень часто менять, гораздо чаще, чем это может делать любая команда разработчиков (по крайней мере, из тех, что я видел).

Для описания пользовательского интерфейса можно найти более удобные формы описания:

* диаграммы переходов между состояниями,
* варианты задач,
* скриншоты и т.д.