GeekBrains

Введение в бизнес-анализ

# Заинтересованные лица и участники проекта

# На этом уроке

1. Классифицируем типы и роли заинтересованных лиц.
2. Разберёмся с классами и ролями пользователей в проекте.
3. Поговорим о том, как выявлять заинтересованных лиц.
4. Изучим подход к сбору требований через варианты использования (use case).

# Оглавление

# Заинтересованные лица

Как мы узнали на прошлом уроке, **заинтересованные в проекте лица** (stakeholders) — это отдельные лица, группы или организации, которые активно вовлечены в проект, на которых влияет результат проекта и которые сами могут влиять на этот результат.

Если раскрыть определение более подробно, **стейкхóлдеры** — заинтересованные стороны, причастные стороны, участники работ, роли в проекте — это лица или организации, имеющие права, долю, требования или интересы относительно системы или её свойств, удовлетворяющих их потребностям и ожиданиям (ISO/IEC/IEEE 15288:2015).

Другие определения:

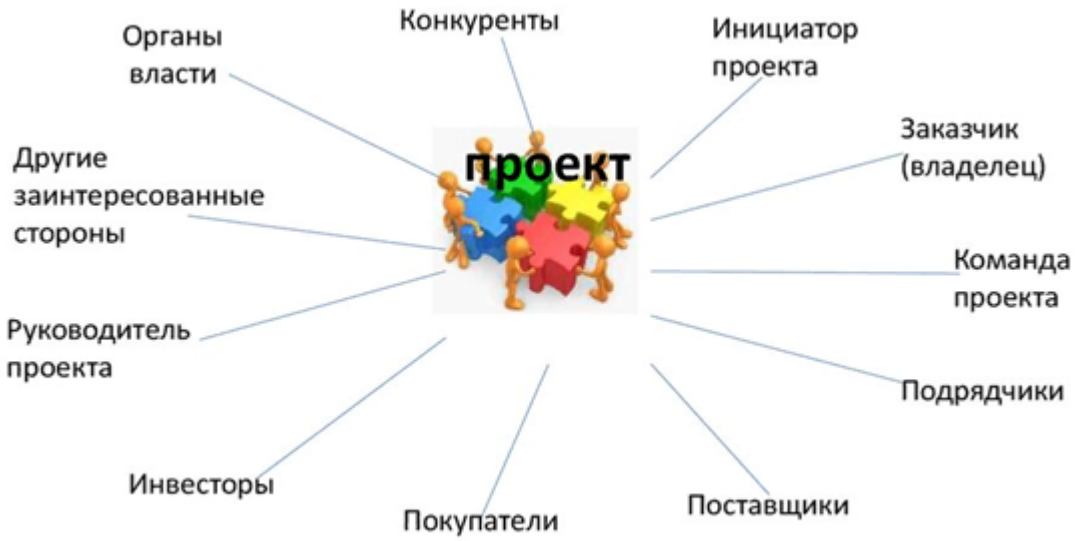
1. Индивидуум, команда, организация или их группы, имеющие интерес в системе (ISO/IEC 42010).
2. Люди, группы или организации, которые могут влиять на систему или на которых может повлиять система (OMG Essence).
3. Лицо, группа или организация, которая может влиять, на которую могут повлиять или которая может воспринимать себя подвергнутой влиянию решения, операции или результата проекта (PMBOK).
4. Лицо или организация, которая может воздействовать на осуществление деятельности или принятие решения, быть подверженной их воздействию или воспринимать себя в качестве подверженной воздействию (ISO 9000:2015).

Стейкхолдеры обеспечивают возможности для системы и выдвигают требования к системе. Также они рассматриваются в контексте процесса принятия решений как люди или организации, зависящие от результатов принимаемых решений.

Нужно заранее определить, кто заинтересован в принятии решений. Это часто упускают из виду. Однако как только решение будет объявлено или реализовано, все, кто хоть как-то был затронут этим решением, выскажут своё мнение.

# Типы заинтересованных лиц

Исчерпывающего списка типов или групп стейкхолдеров не существует, так как для разных целевых систем они могут значительно различаться. Например, в отдельно взятом проекте с участием органов власти стейкхолдеры могут быть такими:




Ниже приведены примеры наиболее распространённых типов (групп) стейкхолдеров, которые упоминаются в стандартах ГОСТ Р ИСО/МЭК 15288-2005, ISO/IEC 29148:2011, ГОСТ Р ИСО/МЭК 12207-2010, OMG Essence, своде знаний по системной инженерии SWEBOK и учебниках по системной инженерии. Под стейкхолдером здесь подразумевается организация или физическое лицо.

1. **Приобретающая сторона, или покупатель** (acquirer) приобретает или получает (procures) продукт или услугу от поставщика. Это может быть покупатель, заказчик, владелец, оптовый покупатель.
2. **Заказчик, или клиент** (customer) получает продукт или услугу.
3. **Разработчик** (developer) выполняет задачи разработки, включая анализ требований, проектирование, тестирование в течение всего жизненного цикла.
4. **Поставщик** (supplier) вступает в соглашение с приобретающей стороной на поставку продукта или услуги.
5. **Пользователь** (user) извлекает пользу в процессе применения системы.
6. **Производитель** (producer) отвечает за выполнение работы, за расписание, бюджет и распределение ресурсов, чтобы удовлетворить клиента.
7. **Сопровождающая сторона** (maintainer) выполняет поддержку системы на одном или нескольких этапах жизненного цикла, осуществляет деятельность по сопровождению.
8. **Ликвидатор** (disposer) выполняет ликвидацию (изъятие и списание) системы и связанных с нею эксплуатационных и поддерживающих служб.
9. **Аккредитор, или инспектор** (accreditor) проверяет систему на соответствие требованиям в процессе сдачи системы в эксплуатацию.
10. **Регулирующий орган** (regulatory bodies) проверяет систему на соответствие требованиям в процессе эксплуатации.
11. **Остальные** — персонал поддержки (supporters), инструкторы (trainers), операторы (operators) и другие.

# Роли заинтересованных лиц

Ход проекта существенно зависит от намерений, действий, информирования и поддержки нужд его участников и заинтересованных лиц. Обычно в рамках каждого процесса существует несколько ролей заинтересованных сторон. У каждого участника может быть более одной роли.

Основные роли заинтересованных в проекте сторон:

1. **Спонсор проекта** предоставляет ресурсы для проекта или формирует инвестиционно привлекательную ситуацию вокруг него. Очень важно определить спонсора до начала проекта, поскольку именно с ним нужно решать все вопросы, касающиеся обеспечения проекта ресурсами.
2. **Менеджер проекта** отвечает за управление проектом, то есть обеспечение наилучших его результатов в рамках заданных ограничений. Это не руководитель проекта, в функции которого входит лидерство: наличие авторитета, умение вовлекать людей в проект, создавать команду, вести людей за собой. Менеджер в классическом понимании не обязан быть лидером, он выполняет только административные обязанности.
3. **Лидер проекта** — наиболее авторитетный человек в команде проекта. К его мнению прислушиваются, он принимает большинство технических решений по ходу проекта. Часто лидер и менеджер проекта — одно лицо.
4. **Заказчик** получает результаты проекта в собственность того или иного вида.
5. **Пользователи** непосредственно используют результаты проекта в своей деятельности.
6. **Организация-исполнитель** выполняет проект и несёт ответственность перед заказчиком за его выполнение. Может быть создана для реализации одного конкретного проекта и состоять только из его команды.
7. **Проектная команда** — коллектив специалистов, объединённых для достижения общих целей и решения поставленных перед ними задач в течение жизненного цикла проекта. Каждый включённый в команду специалист обладает специфическими компетенциями и выполняет определённые функции. В проектную команду входят сотрудники организации-исполнителя, менеджер и лидер проекта.
8. **Команда управления проектом** — часть команды проекта, непосредственно участвующая в деятельности по управлению проектом. Это менеджер проекта и лидер проекта. Также туда могут входить секретарь менеджера, эксперты, помогающие в принятии решений и т. д.


Взаимоотношения между заинтересованными лицами проекта

**Источники влияния** — лица, группы лиц или организации, не связанные напрямую с проведением проекта и использованием его результатов, но имеющие свои интересы в том или ином развитии событий в проекте и способные повлиять на него. Это государственные и общественные организации, компании-конкуренты, средства массовой информации, отдельные лица, чью жизнь проект как-то затрагивает, служащие связанных с проектом организаций, способные повлиять на принимаемые решения, или те, чей карьерный рост может зависеть от проекта, и прочие.

Проектная команда должна идентифицировать заинтересованные стороны, определить их требования и ожидания, до возможной степени управлять их влиянием на проект, чтобы обеспечить его успешность.

# Пользователи

Отдельная группа заинтересованных лиц и участников проекта — пользователи. Это те, кто будет непосредственно работать с системой. В самом начале работы над проектом необходимо определить и охарактеризовать различные классы пользователей, чтобы узнать требования представителей всех важных классов.

**Пользователь** (user) — лицо или организация, которая использует действующую систему для выполнения конкретной функции.

## Классы пользователей

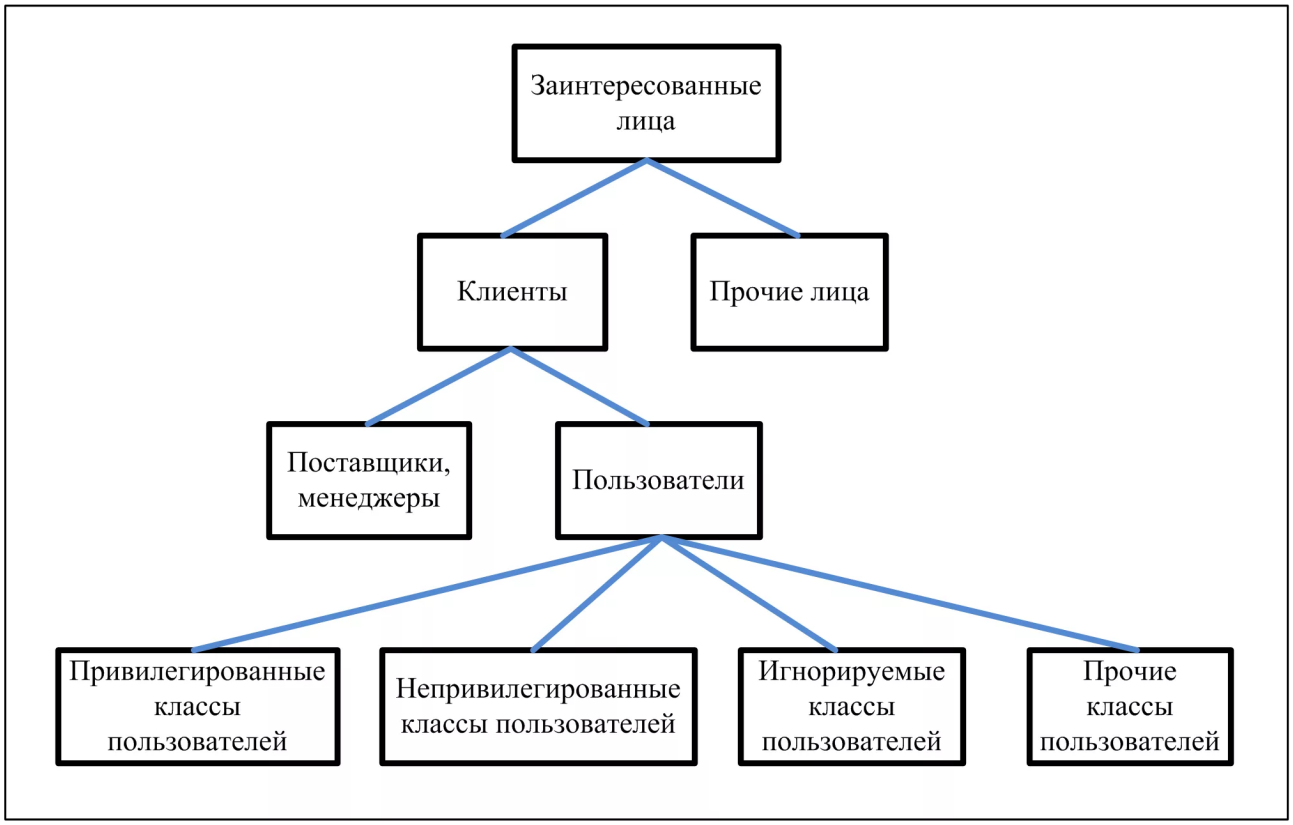
Пользователей продукта можно подразделять по таким признакам:

- частота использования продукта;
- опыт в предметной области и опыт работы с компьютерными системами;

- требуемая им функциональность;
- задачи, которые им приходится решать;
- права доступа к системе, например «обычный пользователь», «гость» или «администратор».

На основе этих признаков формируются классы пользователей. Некоторых сотрудников можно отнести к нескольким классам. Например, администратор иногда работает с системой, как рядовой пользователь. Конечные пользователи ПО — это потенциальные кандидаты на отдельные классы пользователей.

Одни классы пользователей важнее других. При принятии решения о приоритетах или поиске компромисса между требованиями, которые выдвинуты различными классами пользователей, мнение привилегированных классов имеет первостепенное значение.




**Привилегированные классы** — это группы пользователей, работа которых с продуктом определяет, способствует он достижению заявленных бизнес-целей или нет. Это не означает, что обязательно следует включать в привилегированные классы заинтересованных лиц, оплачивающих разработку системы (они, вполне вероятно, вообще не являются её пользователями), или тех, кто имеет большое политическое влияние.

**Непривилегированные классы** — это те пользователи, которые по причинам безопасности, конфиденциальности или правовым причинам не работают с продуктом (Cause & Lawrence, 1999).

**Игнорируемые классы** — это остальные классы пользователей, которые можно проигнорировать. Они получат то, что получится, то есть при разработке системы не нужно учитывать их интересы.

**Прочие классы** — пользователи, мнения которых при определении требований к продукту имеют примерно одинаковое значение.

У каждого класса пользователей есть свой набор требований, сформировавшийся в ходе выполнения задач. Кроме того, появляются различные нефункциональные требования, например удобство применения. Они влияют на проектирование пользовательского интерфейса.

Например, **новичков** волнует, насколько легко научиться работать (или вспомнить принципы работы) с продуктом. Такие клиенты предпочитают графические интерфейсы, упорядоченное представление информации на экране, подробные подсказки, мастеров и согласованность с другими приложениями, с которыми они уже знакомы.

**Опытных пользователей** больше интересует лёгкость использования и эффективность работы. Они ценят клавиатурные сокращения, макросы, возможности настройки, панели инструментов, возможности создания сценариев. В некоторых случаях даже предпочитают интерфейс командной строки графическому интерфейсу пользователя.

Важно помнить о **классах косвенных, или вторичных пользователей**. Они могут не напрямую обращаться к приложению, а работать с его данными и сервисами через другие приложения или отчёты. Однако даже опосредованный клиент всё равно остаётся клиентом.

Классы пользователей не обязательно состоят из людей. **Сторонние приложения и аппаратные компоненты** можно рассматривать в качестве дополнительных классов пользователей, с которыми взаимодействует система.

Например, система впрыска топлива в автомобиле представляет собой пользовательский класс ПО, встроенного в систему управления двигателем. Система впрыска топлива не может говорить сама за себя, поэтому аналитику придётся выяснить у инженера, разработавшего систему впрыска, требования к ПО, которое управляет процессом.

## Выявление классов пользователей

Один из полезных способов определения классов называется **«от расширения к сжатию»** (expand then contract) (Gottesdiener, 2002). Для начала нужно придумать как можно больше классов пользователей, даже если их будет слишком много. Важно не пропустить какой-либо класс, иначе это аукнется позже.

Следующий этап — выявить группы с похожими потребностями. Их можно объединить в один класс или рассматривать как несколько подклассов одного крупного класса пользователей. В списке отдельных классов не должно быть больше 15 пунктов.

Здесь могут оказаться полезными следующие вопросы:

1. Кто пользователи системы?
2. Кто заказчик (экономический покупатель) системы?
3. На кого ещё окажут влияние результаты работы системы?
4. Кто будет оценивать и принимать систему, когда она будет представлена и развёрнута?
5. Существуют ли другие внутренние или внешние пользователи системы, чьи потребности необходимо учесть?
6. Кто будет заниматься сопровождением новой системы?
7. Не забыли ли мы кого-нибудь?

Классы пользователей, их отличитель-ные черты, меру ответственности и физическое расположение в спецификации требований к ПО нужно задокументировать.

### Для чего нужна классификация пользователей

Например, одна компания, разрабатывающая ПО для примерно 65 корпоративных клиентов, рассматривала каждого из них как отдельного пользователя со своими потребностями. Описание требований с позиции каждого клиента сильно затруднило бы работу. Напротив, классификация клиентов по шести классам значительно упростила сбор требований для будущих версий продукта.

**Внимание!** *Не все заинтересованные в проекте лица на самом деле будут работать с продуктом, поэтому класс пользователей — это группа людей, которым следует предоставить возможность выразить своё мнение о создаваемом продукте.*

# Сбор требований

## Применение вариантов использования

**Вариант использования** (use case) продукта описывает последовательность взаимодействия системы и внешнего действующего лица.

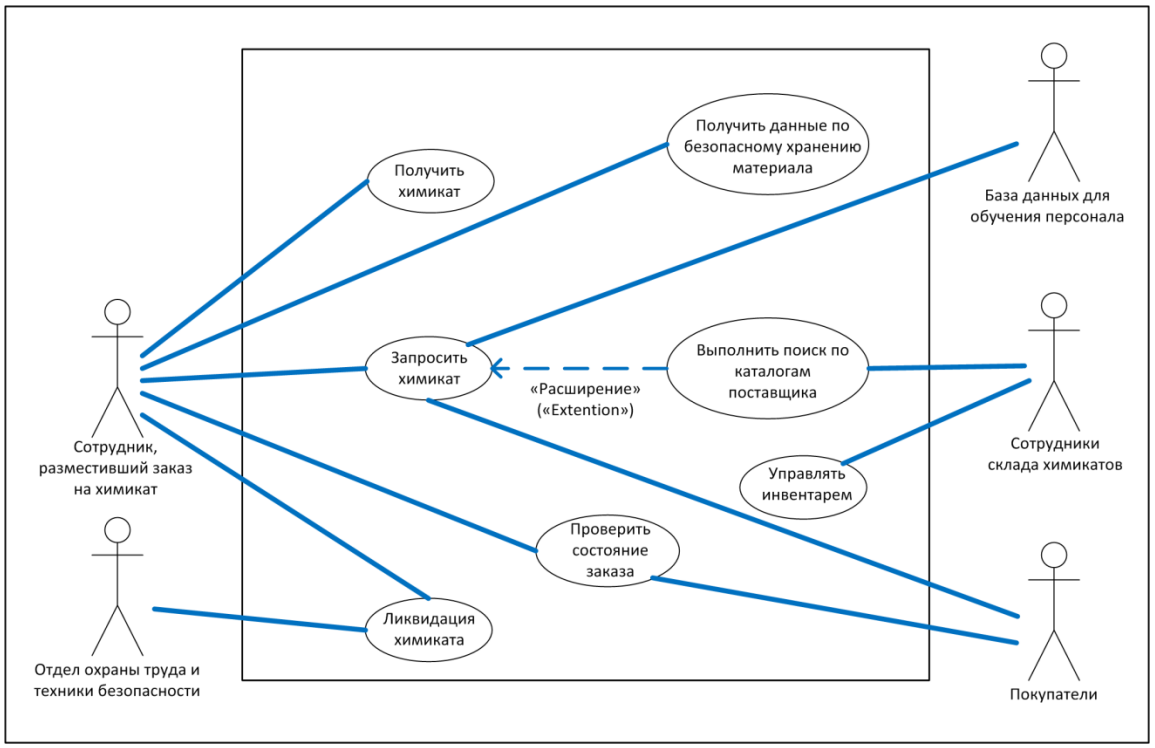
**Действующее лицо** (actor) — это человек, другая система ПО или аппаратное устройство, взаимодействующее с системой для достижения некой цели (Cockburn, 2001).

Ещё одно название действующего лица — **роль пользователя**. Это роль, которую члены одного или нескольких классов пользователей могут играть по отношению к системе (Constantine & Lockwood, 1999).

Например, в варианте использования Chemical Tracking System «Запрос химиката» (см. рисунок) участвует действующее лицо «Сотрудник, разместивший заказ на химикат». Класса пользователей

Chemical Tracking System с таким именем не существует. И химики, и специалисты, работающие на складе химикатов, могут запрашивать химикаты, поэтому члены любого из этих классов могут играть роль сотрудника, разместившего заказ на химикат.

Диаграммы вариантов использования (use case diagrams) позволяют получить визуальное представление о требованиях пользователей.




Фрагмент диаграммы варианта использования Chemical Tracking System

На рисунке показан фрагмент диаграммы варианта использования Chemical Tracking System, где применяются нотации UML (Booch, Rumbaugh & Jacobson, 1999; Armour & Miller, 2001).

Прямоугольник показывает границы системы. Линии соединяют каждое действующее лицо (нарисованный человечек) с вариантами использования (эллипсами), с которыми это лицо взаимодействует. Прямоугольник отделяет некоторые внутренние элементы высокого уровня системы (варианты использования) от внешних действующих лиц.

Понятие «вариант использования» пришло из мира объектно-ориентированного программирования. Оно годится и для проектов, где применяются любые приёмы разработки, поскольку пользователям безразлично, как именно создаётся ПО.

Варианты использования меняют традиционный подход к сбору информации. Пользователей не спрашивают **что, с их точки зрения, должна делать система**, а выясняют, **какие задачи собирается с её помощью решать пользователь.**

Цель такого подхода — описать все подобные задачи. До включения каждого варианта использования в утверждённую версию требований заинтересованные в проекте лица проверяют, не выходит ли он

за границы проекта. Теоретически в конечный набор вариантов использования должна входить вся желаемая функциональность системы. На практике же вряд ли удастся добиться стопроцентного результата. Однако варианты использования помогут решить эту задачу полнее, чем какой-либо другой приём сбора информации.

## Варианты использования и сценарии использования

**Вариант использования** (use case) — это отдельное, независимое действие, которое действующее лицо может выполнить для получения определённого значимого результата.

Один вариант использования может охватывать несколько схожих задач с одинаковыми целями. Следовательно, это набор связанных между собой сценариев использования, где **сценарий** — это отдельный пример варианта использования.

Можно начать сбор требований с абстрактных вариантов использования, а затем на их основе создать конкретные сценарии. Или, наоборот, перейти от сценария к более широкому варианту использования.

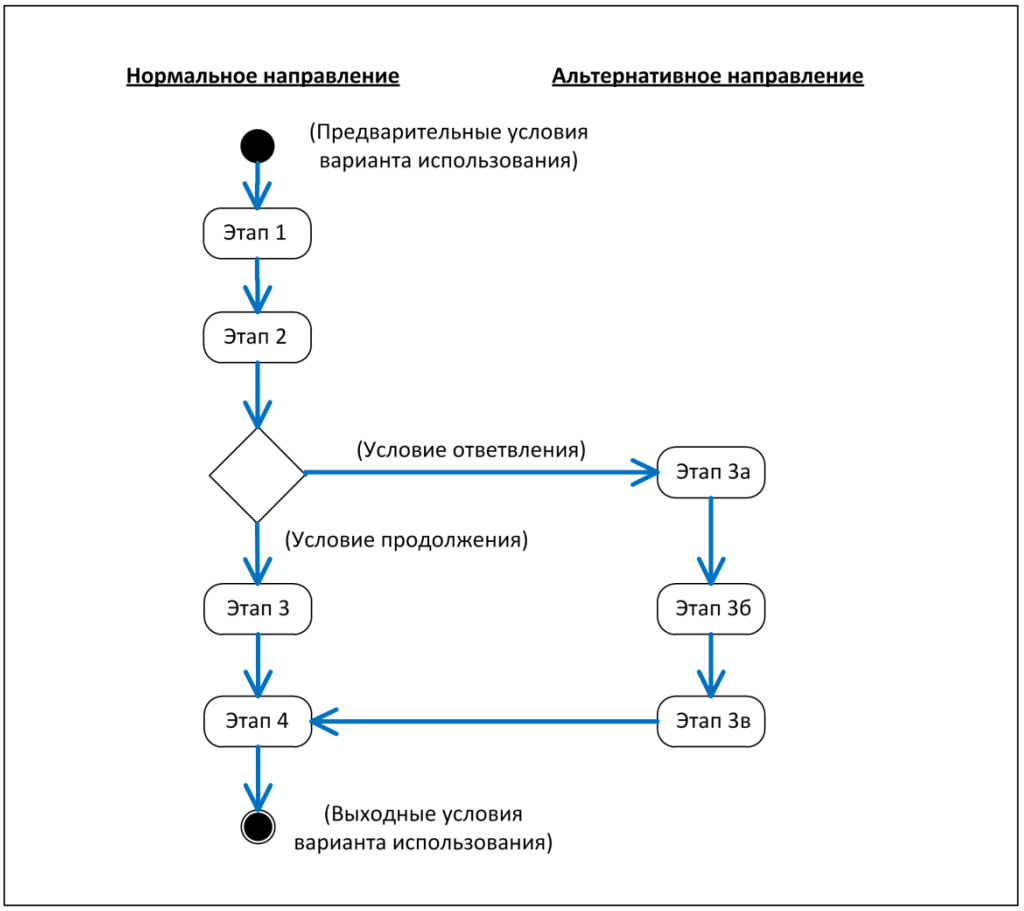
Важные элементы описания варианта использования:

1. Уникальный идентификатор.
2. Имя, кратко описывающее задачи пользователи в формате «глагол + объект», например «разместить заказ».
3. Краткое текстовое описание на естественном языке.
4. Список предварительных условий, которые должны быть выполнены до начала разработки варианта использования.
5. Выходные условия, описывающие состояние системы после успешного завершения разработки варианта использования.
6. Пронумерованный список действий, иллюстрирующий последовательность этапов взаимодействия лица и системы от предварительных условий до выходных условий.

## Нормальное направление варианта использования

**Нормальное направление развития** (normal course) — это основное или базовое направление варианта использования, основной или главный успешный сценарий, благоприятный путь.

Нормальное направление для варианта использования «Запрос химиката» — запрос химиката, который есть на складе. UML-диаграмма иллюстрирует диалоговый поток при нормальном и альтернативном развитии варианта использования:




## Альтернативные направления варианта использования

**Альтернативное направление** (alternative courses) — это другой допустимый сценарий из варианта использования. Также он может называться **вторичным сценарием** (secondary scenarios) (Schneider & Winters, 1998).

Вторичные сценарии также могут привести к успешному выполнению задания и удовлетворяют выходным условиям варианта использования. Однако это вариации решения задачи или диалоговой последовательности, необходимой для решения задачи. В определённой точке принятия решений в диалоговой последовательности нормальное направление может перейти в альтернативное, а затем вернуться обратно в нормальное. Хотя большинство вариантов использования можно описать простым языком, блок-схема или UML-диаграмма позволяет визуально представить логическое развитие сложного варианта использования, как показано на рисунке.

**Блок-схемы и диаграммы взаимодействия показывают точку принятия решений и условия, при которых основное направление развития событий переходит в альтернативное.**

Например, альтернативное направление для варианта использования «Заказать химикат» — это «Заказать химикат у поставщика». В обеих ситуациях конечная цель действующего лица одна и та же — запрос химиката, поэтому оба сценария входят в один вариант использования. Некоторые этапы альтернативного направления аналогичны этапам нормального направления, но для завершения альтернативного направления необходимо выполнить особые действия.

В данном случае пользователь может искать необходимый химикат по каталогам поставщиков. Если альтернативное направление само по себе — автономный вариант использования, можно расширить нормальное направление, включив этот отдельный вариант использования в нормальный поток. На диаграмме вариант использования «Запросить химикат» был расширен вариантом использования «Выполнить поиск по каталогам поставщика». Кроме того, сотрудники склада химикатов используют «Поиск по каталогам поставщиков» как отдельный вариант.

Иногда несколько вариантов использования имеют общие наборы этапов. Чтобы избежать их повторения в каждом варианте использования, следует выделить отдельный вариант с общей функциональностью и указать, что он включён в другие варианты использования как подвариант. Эта процедура аналогична вызову общей подпрограммы в компьютерной программе.

## Исключения вариантов использования

**Исключение** (exception) — это условие, препятствующее успешному завершению варианта использования.

Для варианта использования «Запросить химикат» существует одно исключение — «Химиката нет в продаже».

Если в процессе сбора информации не указать, как обрабатывать исключение, то возможны два пути:

1. Разработчики предложат лучший, по их мнению, способ обработки исключений.
2. При генерации пользователем неверного условия произойдёт сбой системы, так как никто не предусмотрел такой ситуации.

Но сбои системы не входят в список требований пользователей.

Иногда исключения рассматриваются как тип альтернативного направления (Cockburn, 2001), однако эти понятия следует разделять. Не обязательно реализовывать каждое альтернативное направление для варианта использования. Можно отложить его реализацию до следующего выпуска.

Однако необходимо и крайне важно реализовать исключения, из-за которых завершение сценариев может оказаться неуспешным. Любой, кто когда-либо занимался программированием, знает, что часто

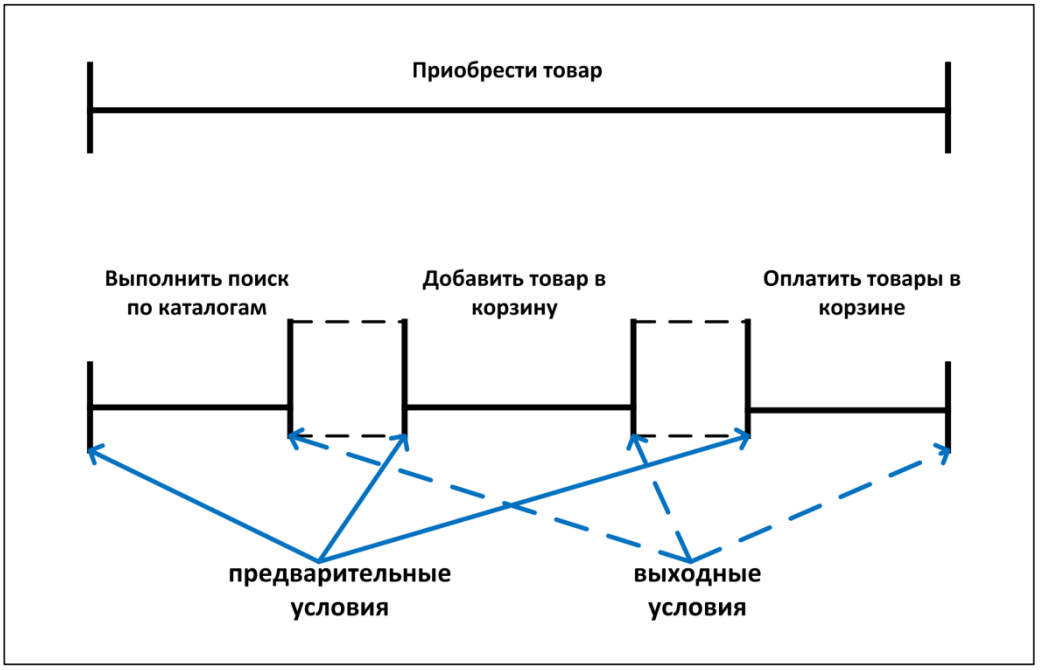
именно на работу над исключениями приходится большая часть программирования. Многие недостатки конечного продукта присущи именно обработчикам исключений или происходят из-за их отсутствия. Указать условия исключений в ходе сбора информации по требованиям — верный способ создать устойчивые к сбоям продукты.

## Последовательность вариантов использования

После каждого варианта использования система должна быть в таком состоянии, чтобы пользователь мог приступить к следующему варианту использования немедленно. То есть выходные условия одного варианта использования должны удовлетворять предварительным условиям следующего за ним варианта.

Например, в приложении обработки транзакций каждый вариант использования должен оставлять систему в состоянии готовности к следующей транзакции. Предварительные условия и выходные условия каждой транзакции варианта использования должны вставать в один ряд.

Для коммерческого веб-сайта предлагаются варианты использования «Поиск по каталогу», «Добавить товар в корзину» и «Оплатить товары в корзине». Если можно выполнить все эти действия по отдельности, то все они считаются независимыми вариантами использования. Кроме того, можно выполнить все три указанные операции подряд, назвав этот один большой вариант использования «Приобрести товар», как показано на рисунке:




Предварительные условия и выходные условия определяют границы отдельных вариантов использования, которые можно связать в цепочку для выполнения объёмной задачи.

## Выявление вариантов использования

Можно выявить варианты использования несколькими способами (Ham, 1998; Larman, 1998):

- сначала определить действующих лиц, а затем бизнес-процессы, в которых каждое лицо участвует;
- выразить бизнес-процессы в терминах определённых сценариев, обобщить сценарии в варианты использования и определить действующие лица для каждого варианта;
- определить внешние события, на которые система должна реагировать, а затем соотнести эти события с действующими лицами и определёнными вариантами использования;
- определить вероятные варианты использования на основе функциональных требований,
- если какие-либо требования невозможно проследить до какого-либо варианта использования, определить, нужны ли они.

Название варианта использования должно указывать на решаемую задачу, поэтому при формулировке названий необходимо использовать глагол. **Что клиент хочет:** просмотреть, напечатать, заказать, отправить по электронной почте, исправить, удалить или создать данные по безопасному хранению материала?

Иногда предложенный вариант использования — это всего лишь одно действие, которое выполняется как часть варианта использования, например «сканировать штрих-код». Аналитик должен выяснить, какую цель подразумевал пользователь, когда предлагал вариант использования. Например, спросить: «Сканируя штрихкод на контейнере с химикатами, что вы пытаетесь сделать?» Предположим, в ответ он услышит: «Это необходимо, чтобы я мог отправить этот химикат в свою лабораторию». Следовательно, настоящий вариант использования — «Отправить химикат в лабораторию». Сканирование штрих-кода — это только один из этапов взаимодействия действующего лица и системы, передающей химикат в лабораторию.

## Документирование вариантов использования

Есть два способа представить этапы взаимодействия пользователя и системы, которые и составляют основу варианта использования:

1. Составить пронумерованный список этапов с указанием того, какой элемент (система или определённое действующее лицо) выполняет каждый шаг. Аналогично описать альтернативные направления и исключения, для которых показан этап нормального направления развития, когда появляется ответвление, или этап, где возможно исключение.

2. Описать процесс в виде таблицы из двух столбцов. Действия лица показать в левом столбце, а реакцию системы — в правом. Цифры указывают на последовательность этапов диалога. Эта схема отлично работает, когда с системой взаимодействует только один пользователь.

Фрагмент описания варианта использования «Запросить химикат»:

| **Идентификатор варианта использования** | **Вариант использования 1** |
|---|---|
| **Название варианта использования** | **«Запросить химикат»** |
| Действующее лицо | Сотрудник, разместивший заказ на химикат |
| Описание | Сотрудник, разместивший заказ на химикат, указывает в запросе необходимый химикат, вводя его название или номер идентификатора химиката или импортируя его структуру из соответствующего графического средства.<br>Система выполняет запрос, предлагая новый или использованный контейнер с химикатом со склада или позволяя создать запрос на заказ у стороннего поставщика |
| Предварительные условия | 1. Личность пользователя аутентифицирована.<br>2. Пользователь имеет право запрашивать химикаты.<br>3. База данных по запасам химикатов в данный момент подключена |
| Выходные условия | 1. Запрос сохраняется в Chemical Tracking System.<br>2. Запрос был отправлен по электронной почте на склад химикатов или поставщику |
| Нормальное направление развития варианта использования | 1.0 «Запросить химикат со склада»<br>1. Сотрудник указывает требуемый химикат.<br>2. Система подтверждает, что такой химикат доступен.<br>3. Система перечисляет контейнеры с необходимым химикатом, имеющиеся на складе.<br>4. Сотрудник может просмотреть историю любого контейнера.<br>5. Сотрудник выбирает определенный контейнер или просит отправить запрос поставщику (альтернативное направление 1.1).<br>6. Сотрудник вводит остальную информацию, чтобы завершить запрос.<br>7. Система сохраняет запрос и отправляет его по электронной почте на склад химикатов |
| Альтернативное направление развития варианта использования | 1.1 «Запросить химикат у поставщика» (ответвление после этапа 5)<br>1. Сотрудник ищет химикат по каталогам поставщика.<br>2. Система отображает список поставщиков, где также указаны размеры, класс и цена контейнеров.<br>3. Сотрудник выбирает поставщика, размер, класс и количество контейнеров.<br>4. Сотрудник вводит остальную информацию, чтобы завершить запрос.<br>5. Система сохраняет запрос и отправляет его по электронной почте поставщику |
| Исключения | 1.0.И.1 — «Химикат недоступен» (на этапе 2)<br>1. Система отображает сообщение «Химикат не существует».<br>2. Система спрашивает сотрудника, хочет ли он запросить другой химикат или выйти из программы.<br>3а. Сотрудник просит запросить другой химикат.<br>4а. Система заново начинает нормальное направление развития варианта использования.<br>3б. Сотрудник решает выйти из системы. |

| | |
|---|---|
| | 4б. Система завершает вариант использования.<br>1.0.И.2. «Химиката нет в продаже»(на этапе 5).<br>1. Система отображает сообщение «Нет поставщика этого химиката».<br>2. Система спрашивает сотрудника, хочет ли он запросить другой химикат или выйти из программы.<br>3а. Сотрудник запрашивает другой химикат.<br>4а. Система заново начинает нормальное направление развития варианта использования.<br>3б. Сотрудник решает выйти из системы.<br>4б. Система завершает вариант использования |
| Включение | Вариант использования 22 — «Просмотреть хронологию контейнера» |
| Приоритет | Высокий |
| Частота использования | Используется примерно 5 раз в неделю каждым химиком, 100 раз в неделю каждым работником склада |
| Бизнес-правила | Бизнес-правило 28 — «Только сотрудник, получивший разрешение заведующего лаборатории, может запрашивать химикаты» |
| Специальные требования | 1. Система должна импортировать химические структуры в стандартной зашифрованной форме из любых средств, поддерживающих рисование химических структур |
| Предположения | 1. Импортированные химические структуры должны быть достоверными |
| Замечания и вопросы | 1. Выяснить, нужно ли одобрение руководства для запроса химиката, относящегося к уровню 1 списка опасных химикатов. Дата выполнения: 04.01.03. |

При документировании вариантов использования необходимо сосредоточиться на выявлении важнейших вариантов использования.

**Важнейший вариант использования** (essential use case) — упрощённое, обобщённое, абстрактное, не зависящее от технологии и реализации описание одной задачи или взаимодействия, в котором воплощена цель или намерения, лежащие в основе взаимодействия (Constantine & Lockwood, 1999).

То есть следует сосредоточиться на задаче, которую пользователю необходимо выполнить, и возможностях системы для выполнения этой задачи. Важнейшие варианты использования характеризуются более высоким уровнем абстракции, чем конкретные варианты использования (concrete use cases), которые описывают определённые действия, предпринимаемые пользователем для взаимодействия с системой.

Например, два способа начала реализации пользователем варианта использования «Запросить химикат»:

1. **Конкретный вариант использования** — «Введите номер ID химиката».
2. **Важнейший вариант использования** — «Укажите необходимый химикат».

Формулировка на важнейшем уровне позволяет пользователю выполнить задачу многими способами: ввести номер ID химиката, импортировать химическую структуру из файла, нарисовать структуру на экране с помощью мыши, выбрать химикат из списка и многое другое. Независимые от реализации

важнейшие варианты использования более пригодны для повторных применений, чем конкретные варианты использования.

Например, если пользователь говорит: «По умолчанию это должно быть…», значит, он описывает нормальное направление развития варианта использования. Фраза «Необходимо, чтобы пользователь также смог...» предполагает обсуждение альтернативного направления. Многие условия исключений можно выяснить с помощью вопроса «Что должно произойти, если в текущий момент БД отключена?» или «Что, если этого химиката нет в продаже?»

Когда вариант использования описан и никто уже не предлагает дополнительных вариантов, исключений или специальных требований, можно переходить к следующему варианту использования. Не стоит рассматривать все варианты использования за одно марафонское обсуждение или точно определять все детали каждого варианта использования. Обычно они рассматриваются многократно и уточняются.

## Преимущества применения вариантов использования

Способ с применением вариантов использования облегчает расстановку приоритетов требований. Высшим приоритетом обладают те функциональные требования, которые созданы на основе вариантов использования с высшим приоритетом.

Высший приоритет назначается по следующим причинам:

- варианты использования описывают один из основных бизнес-процессов, активизируемых системой;
- многие пользователи часто обращаются к ним;
- их запросил привилегированный класс пользователей;
- они предоставляют возможности, необходимые для соответствия требованиям;
- функции других систем зависят от их наличия.

С помощью варианта использования можно выявить важные объекты предметной области и их взаимоотношения. Разработчики, использующие объектно-ориентированные методы дизайна, могут преобразовать варианты использования в объектные модели, такие как диаграммы классов и диаграммы последовательностей.

Но не стоит тратить много времени на обсуждение деталей вариантов использования, которые не будут реализованы в ближайшие месяцы или годы. Вероятнее всего, они изменятся ещё до начала сборки ИС. По мере того как с течением времени изменяются бизнес-процессы, уточняются задачи, реализуемые посредством определённых вариантов использования. Если проследить функциональные требования, дизайн, код и тесты вплоть до их истоков — до пожеланий клиента, будет легче внести эти изменения бизнес-процессов во всю систему.

# Риски и ошибки при применении вариантов использования

**Слишком много вариантов использования**. В данном случае вряд ли удастся записать каждый из них на соответствующем уровне абстракции. Не стоит создавать отдельный вариант использования для каждого возможного сценария. Лучше включить нормальное направление развития, альтернативные направления и исключения в виде сценариев в один вариант использования. Как правило, количество вариантов использования превышает количество бизнес-требований и функций, однако функциональных требований обычно бывает намного больше, чем вариантов использования.

**Очень сложные варианты использования**. Создавать длинные описания — не самое лучшее решение. Нужно выбрать один успешный способ выполнения варианта использования с одной комбинацией правильных и ложных значений для различных логических решений и назвать его нормальным направлением. Другие успешные логические ответвления определить как альтернативные направления и назначить исключения для обработки неудачных ответвлений. Альтернативных направлений может быть множество, однако каждое должно быть кратким и понятным.

**Включение пользовательского интерфейса в варианты использования**. Варианты использования должны описывать то, что пользователям необходимо выполнить с помощью системы, а не то, как это будет выглядеть на экране. Правильная формулировка на этой стадии — «система предоставляет выбор», а не «система отображает раскрывающийся список». Наброски экрана и карты диалогов (диаграммы архитектуры интерфейса) стоит применять для визуального представления взаимодействия исполнителя и системы, а не для утверждения дизайна.

**Включение определений данных в варианты использования**. Определения элементов данных и структур трудно отыскать, когда они включены в описание вариантов использования. Возможны повторы определений и рассинхронизация, когда один экземпляр изменяется, а остальные — нет. Лучше собрать их в словарь данных, созданный для всего проекта.

**Варианты использования, которые непонятны пользователям**. Если пользователи не видят связи описания вариантов использования со своими бизнес-процессами или задачами, то варианты использования следует подкорректировать. Их нужно составлять с точки зрения клиентов, а не системы. Следует попросить клиентов проверить их. Излишне сложные варианты использования, скорее всего, говорят о том, что они не нужны.

**Новые бизнес-процессы**. Пользователям трудно придумывать новые варианты использования, если ПО создаётся для поддержки процесса, которого ещё нет. В этом случае сбор информации по требованиям — это изобретение нового бизнес-процесса. Рискованно ожидать, что с помощью новой информационной системы будет создан эффективный бизнес-процесс. Необходимо выполнить модернизацию бизнес-процесса перед описанием новой информационной системы.

# Практическое задание

Вспомним задание предыдущего урока. Нам понадобится список заинтересованных лиц и бизнес-требований.

1. Классифицируйте типы и роли заинтересованных лиц. Выделите классы пользователей относительно списка требований. Проверьте, все ли требования закрыты.
2. Опишите основной вариант использования относительно списка требований. Нарисуйте диаграмму вариантов использования. Выделите нормальное направление, альтернативное направление, опишите исключения.

Дополнительное задание* (*не влияет на оценку преподавателя*):

1. Опишите ещё один основной вариант использования к предыдущему кейсу.
2. Выделите нормальное направление, два альтернативных направления и два исключения.

# Глоссарий

**UML** (сокр. от англ. Unitied Modeling Language) — унифицированный язык моделирования.

**Альтернативное направление** (alternative course) — это другой допустимый сценарий из варианта использования. Также он может называться вторичным сценарием (secondary scenario).

**Важнейший вариант использования** (essential use case) — упрощённое, обобщённое, абстрактное, не зависящее от технологии и реализации описание одной задачи или взаимодействия, в котором воплощена цель или намерения, лежащие в основе взаимодействия».

**Вариант использования** (use case) — это отдельное, независимое действие, которое действующее лицо может выполнить для получения определённого значимого результата.

**Действующее лицо** (actor) — это человек, другая система ПО или аппаратное устройство, взаимодействующее с системой для достижения некой цели (Cockburn, 2001).

**Исключение** (exception) — это условие, препятствующее успешному завершению варианта использования.

**Нормальное направление развития** (normal course) — это основное или базовое направление варианта использования, основной или главный успешный сценарий, благоприятный путь.

**ПО** — программное обеспечение.

**Пользователь** (user) — лицо или организация, которая использует действующую систему для выполнения конкретной функции.

**Роль пользователя** — это роль, которую члены одного или нескольких классов пользователей могут выполнять по отношению к системе.

**Стейкхолдеры** (stakeholders) — лица или организации, имеющие права, долю, требования или интересы относительно системы или её свойств, удовлетворяющих их потребностям и ожиданиям.

**Сценарий** — это отдельный пример варианта использования.

# Используемые источники


1. Определение и типы стейкхолдеров.
2. Заинтересованные в проекте лица и их роли.
3. Анализ требований по Вигерсу. Классы пользователей.
4. Основные источники получения информации о потребностях клиента.
5. Как понять требования пользователей.