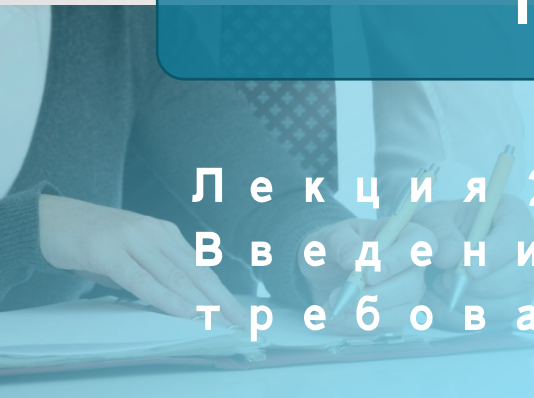
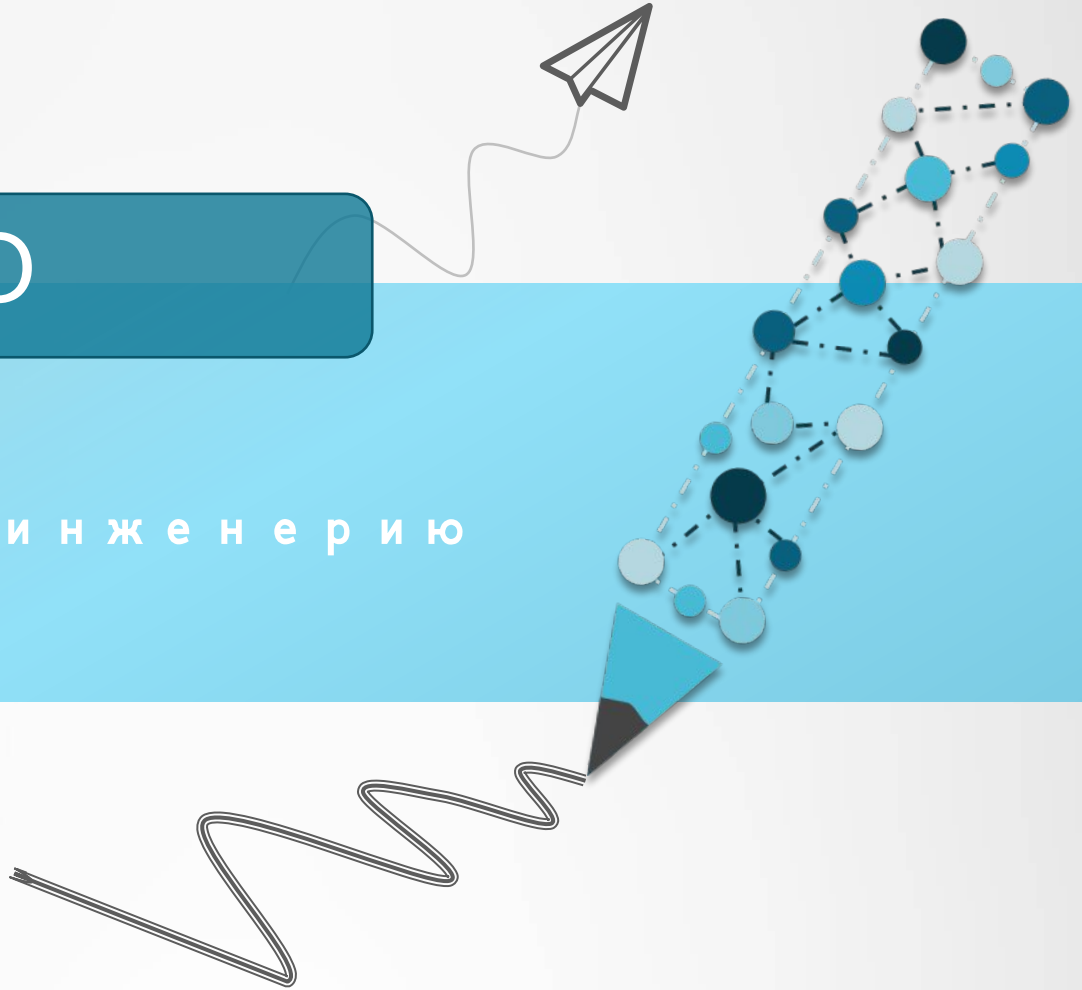
ТРПО

Лекция 2
Введение в инженерию требований





План

- Формирование и классификация требований
- Методы сбора требований
- SRS
- Моделирование (IDEF, UML, ARIS)

Бизнес-анализ — это структурированное изучение проблемы, имеющей отношение к бизнесу. Бизнес-анализ проводится, чтобы лучше понять проблему, а затем оценить, что требуется для ее устранения.

Бизнес-кейс/коммерческое предложение. Коммерческая цель или выгода является причиной выполнения проекта и может официально фиксироваться в документе, называющемся бизнес-кейсом, или коммерческим предложением. Этот документ обычно включает финансовые показатели (например, прирост выручки, снижение издержек и т.п.) или другие параметры (например, повышение уровня обслуживания потребителей, повышение мотивированности персонала и т.д.).

Сбор требований

Прежде чем начинать проект, обязательно нужно знать, какой результат (продукт) вы хотите получить. И порой этот продукт необходимо описать самым тщательным образом. Иными словами, нужно знать, какие требования заказчик предъявляет к продукту. **Полный набор этих требований называют каталогом требований, или спецификацией.**

Крупные и сложные проекты обычно насчитывают тысячи требований. Бизнес-анализ как раз и позволяет выявлять проблемы и определять, что требуется для их преодоления. В крупных проектах, таких, как разработка программного обеспечения, сбор требований является одним из важнейших этапов жизненного цикла проекта,

Сбор требований

Для выявления требований проводится **серия структурированных интервью с заказчиками**, которые позволяют точно определить их пожелания к готовому продукту.

Попытка напрямую узнать у заказчика, какие результаты ему нужны, может закончится крахом: заказчик станет выдвигать все новые и новые требования, так что вы просто будете не в силах их удовлетворить.

Помните, любое требование влияет на продолжительность и стоимость проекта.

Сбор требований

Соответственно, получая подробный список требований, вам нужно знать, являются ли они:

- **обязательными**, т.е. без них проект будет считаться незавершенным, а заказчик останется неудовлетворенным. Если готовый продукт не соответствует всем обязательным требованиям, это — провал;
- **желательными**. Эти требования не обязательны, но важны для заказчика, поэтому их стараются соблюдать, за исключением тех случаев, когда они влекут за собой неприемлемое увеличение стоимости или продолжительности проекта;

необязательными — это те требования, без которых заказчик вполне может обойтись и которые удовлетворяются только по

Кратко о процессе формирования требований

Функциональные требования — это требования к системе.

Бизнес-требования — эквивалентно бизнес-целям*.

Между ними — **Пользовательские требования**, User Requirements.

Пользовательские требования формулируются в терминах предметной области, а **функциональные требования** — в терминах системы.

*Бизнес-цель - это главный фактор, побуждающий к выполнению, реализации проекта. Формирование цели происходит на стратегическом уровне, то есть цель связана с глобальными задачами и одновременно с самим проектом.

Кратко о процессе формирования требований

Бизнес-процессы — самое начало работы.

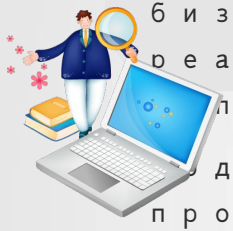
Например, можно рассмотреть процессы RUP*/MSF* (упрощенная последовательность):

1. Бизнес-моделирование
2. Выявление требований
3. RUP*: Анализ и проектирование, MSF:* концептуальный, логический и физический дизайн
4. Реализация
5. Тестирование
6. Опытно-промышленная эксплуатация
7. Support и развитие системы

Кратко о процессе формирования требований

***Microsoft Solutions Framework (MSF)** – представляет общую методологию разработки и внедрения IT решений.

Особенность этой модели состоит в том, что благодаря своей гибкости и отсутствию жестко навязываемых процедур она может быть применена при разработке весьма широкого круга IT проектов. Эта модель сочетает в себе свойства двух стандартных производственных моделей: каскадной (waterfall) и спиральной (spiral). Модель процессов покрывает весь жизненный цикл создания решения, начиная с его отправной точки и заканчивая непосредственно внедрением. Такой подход помогает проектным группам сфокусировать свое внимание на бизнес–отдаче решения, поскольку эта отдача становится реальной лишь после завершения внедрения и начала пользования продукта.



Модель процессов MSF учитывает постоянные изменения проектных требований. Она исходит из того, что

Кратко о процессе формирования требований

***Rational Unified Process (RUP)** – методология разработки программного обеспечения, использующая итеративную модель разработки.

В конце каждой итерации проектная команда должна достичь запланированных на данную итерацию целей, создать или доработать проектные артефакты и получить промежуточную, но функциональную версию конечного продукта.

Итеративная разработка позволяет быстро реагировать на меняющиеся требования, обнаруживать и устранять риски на ранних стадиях проекта, а также эффективно контролировать качество создаваемого продукта.

Кратко о процессе формирования требований

***Extreme Programming (XP) –** гибкая (agile) методология разработки программного обеспечения.

XP успешно применяется на проектах среднего размера, в которых заранее сложно составить формальное техническое задание.

XP отходит от традиционного процесса создания программной системы и вместо единоразового планирования, анализа и проектирования с расчетом на долгосрочную перспективу при XP все эти операции реализуются постепенно в ходе разработки, добиваясь тем самым значительного сокращения стоимости изменений в проекте.

Кратко о процессе формирования требований

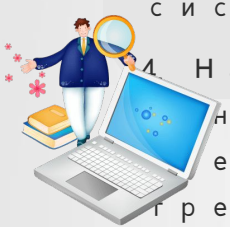
Если говорить упрощенно, то:

1. От заказчика поступает начальная концепция системы (в нескольких предложениях что они хотят, что это позволит достигнуть и т.д.) — по сути это и есть бизнес-требования.

2. Приступаем к моделированию бизнес-процессов, которые хотим автоматизировать (тут в помощь нам ARIS, IDEF0/IDEF3, UML), возможно, строим дополнительную модель (оптимизированную), в которой будут прописаны бизнес-процессы после автоматизации.

3. Получаем от заказчика требования к разрабатываемой системе (это будут пользовательские требования).

4. На основе пользовательских требований формулируем функциональные требования к системе (пользовательские требования — не единственный источник функциональных требований).



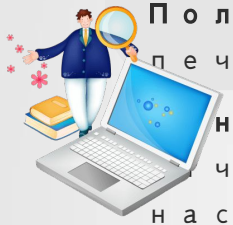
Кратко о процессе формирования требований

Типовая структура требований выглядит как «Система должна … /утверждение о необходимом функциональном поведении системы/» или «система должна позволять … /утверждение о возможности, предоставляемой пользователю или внешней системе/.

Например:

«Система должна вести журнал всех действий пользователя» или «Система должна позволять создавать новые Проекты».

Пример различий между пользовательскими и функциональными требованиями:



Пользовательское: «Система должна выводить отчеты на печать»

Функциональное: «Система должна обеспечивать вывод отчетов на печать, обеспечивать возможность выбора и настройки локального или сетевого принтера, выбора

Кратко о процессе формирования требований

Пользовательские и функциональные требования как правило связаны между собой. Это необходимо для отслеживания зависимостей требований друг от друга. В системах управления требованиями (например, Borland CaliberRM, TelelogicDoors, Rational RequisitePro) для этого есть так называемые «матрицы трассировки», на которых графически стрелками показываются зависимости между требованиями.

Важно сохранять пользовательские требования для хранения их в первоначальном виде, отслеживания источника их возникновения (вплоть до конкретного лица), расстановки их приоритетов (с точки зрения пользователя) и т.д.

Формирование и анализ требований

Анализ предметной области. Аналитики должны изучить предметную область, где будет эксплуатироваться система.

Сбор требований. Это процесс взаимодействия с лицами, формирующими требования. Во время этого процесса продолжается анализ предметной области.

Классификация требований. На этом этапе бесформенный набор требований преобразуется в логически связанные группы требований.

Разрешение противоречий. Без сомнения, требования многочисленных лиц, занятых в процессе формирования требований, будут противоречивыми. На этом этапе определяются и разрешаются противоречия такого рода.

Назначение приоритетов. В любом наборе требований одни из них будут более важны, чем другие. На этом этапе совместно с ами, формирующими требования, определяются наиболее жные требования.

Проверка требований. На этом этапе определяется их полнота, последовательность и непротиворечивость.

Подготовка к интервью по сбору требований у заказчика

Подготовка к интервью

- Определите интервьюируемое лицо
- Организуйте предварительную встречу, телефонный звонок
- Установите ЦЕЛЬ, время, место, участников, возможность записи и использования данных, порядок проведения всех интервью
- DO YOUR HOMEWORK: подготовьте первичную гипотезу, опросник, пропланируйте ход интервью

Проведение интервью

- Будьте пунктуальны
- Имейте при себе документы для пропуска в здание и визитки (если Вы выезжаете к клиенту)
- Следите за временем и управляйте диалогом
- Проясните все, что непонятно, просите все материалы, которые вам могут потребоваться
- Установите дату согласования протокола и т.п.
- Поблагодарите за уделенное время

Дальнейшие действия

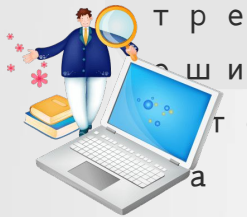
- Подготовьте протокол
- Структурируйте все свои записи В ТОТ ЖЕ ДЕНЬ
- Внесите изменения в уже имеющуюся у Вас гипотезу
- Отправьте на согласование протокол и исправленную версию вашей гипотезы в оговоренное время
- АКТУАЛИЗИРУЙТЕ материалы по итогам согласования

Аттестация требований

Аттестация должна продемонстрировать, что требования действительно определяют ту систему, которую хочет иметь заказчик.

Проверка требований важна, так как ошибки в спецификации требований могут привести к переделке системы и большим затратам, если будут обнаружены во время процесса разработки системы или после введения её в эксплуатацию.

Стоимость внесения в систему изменений, необходимых для устранения ошибок в требованиях, намного выше, чем исправление ошибок проектирования или кодирования. Причина в том, что изменение требований обычно влечёт за собой значительные изменения в системе, после внесения которых она должна пройти



Компания – Бизнес-требования

Поговорим о классификации и описании требований на пути от бизнеса к технической реализации.

Компания — Бизнес-требования

Источники: Топ-менеджмент компании

Документ: Бизнес-требования (обоснование потребностей инициативы)

Ответственный: Менеджер проекта

Состав бизнес-требований может отличаться на практике. Обычно включаются следующие пункты:

1. Описание контекста и списка ключевых заинтересованных лиц
2. Описание целей создания системы и критериев достижения
3. Ключевые бизнес-требования к решению и их приоритеты
4. Существующие и возможные ограничения на решение

Контекст — это то, что стало причиной создания системы, какая ситуация была в компании, какая проблема и как пришли к тому, что систему надо делать.

Пользователь — Требования пользователя

Пользовательские требования — описание на естественном языке (плюс поясняющие диаграммы) функций, выполняемых системой, и ограничений, накладываемых на неё.

Источники: Пользователь

Документ: Пользовательские требования/Требования к ПО

Ответственный: Системный аналитик

Эти требования должны определять только внешнее поведение системы, избегая по возможности определения структурных характеристик системы. Пользовательские требования должны быть написаны естественным языком с использованием простых таблиц, а также наглядных и понятных диаграмм.

Проблемы при формировании пользовательских требований

Отсутствие чёткости изложения. Иногда нелегко изложить какую-либо мысль естественным языком чётко и недвусмысленно, не сделав при этом текст многословным и трудно читаемым.

Смешение требований. В пользовательских требованиях отсутствует чёткое разделение на функциональные требования, на системные цели и проектную информацию.

Объединение требований. Несколько различных требований к системе могут описываться как единое пользовательское требование.

Системные требования

Системные требования описывают свойства и методы всех объектов системы. Программирование – это разработка и реализация структур данных и алгоритмов. Для разработки системы программисту необходимо знать структуры данных, необходимые для реализации системы, и алгоритмы (бизнес-правила/процедуры/пакеты обработки данных), которые ими манипулируют. Системные требования — детализированное описание системных функций и ограничений, которое иногда называют функциональной спецификацией. Она служит основой для ключения контракта между покупателем стемы и разработчиками ПО.

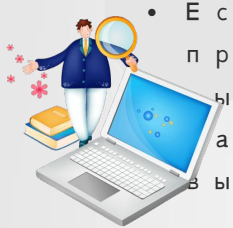
Системные требования — это более

Функциональные требования

Функциональные требования — это перечень сервисов, которые должна выполнять система, причём должно быть указано, как система реагирует на те или иные входные данные, как она ведёт себя в определённых ситуациях и т.д. В некоторых случаях указывается, что система не должна делать.

Стандартные формы для спецификирования функциональных требований:

- Описание функции или объекта.
- Описание входных данных и их источники.
- Описание выходных данных с указанием пункта их назначения.
- Указание, что необходимо для выполнения функции.
- Если это спецификация функции, необходимо описание предварительных условий (предусловий), которые должны выполняться перед вызовом функции, и описание заключительного условия (постусловия), которое должно быть выполнено после завершения выполнения функции.
- Описание побочных эффектов (если они есть).



Функциональные требования

Функциональные требования (functional requirements) определяют функциональность ПО, которую разработчики должны построить, чтобы пользователи смогли выполнить свои задачи в рамках бизнес-требований.

Иногда они называются требованиями поведения (behavioral requirements), они содержат положения с традиционным «должен» или «должна»: «Система должна по электронной почте отправлять пользователю подтверждение о заказе».

Нефункциональных требований

Нефункциональные требования — описывают характеристики системы и её окружения, а не поведение системы. Здесь также может быть приведён перечень ограничений, накладываемых на действия и функции, выполняемые системой.

Они включают временные ограничения, ограничения на процесс разработки системы, стандарты и т.д.

Нефункциональные требования не связаны непосредственно с функциями, выполняемыми системой. Они связаны с такими интеграционными свойствами системы, как надёжность, время ответа или размер системы. Кроме того, нефункциональные требования могут определять ограничения на систему, например на пропускную способность устройств ввода-вывода, или форматы данных, используемых в системном интерфейсе.

Нефункциональных требований

Нефункциональные требования основываются на бюджетных ограничениях, учитывают организационные возможности компании-разработчика, возможность взаимодействия разрабатываемой системы с другими программными и вычислительными системами, а также такие внешние факторы, как правила техники безопасности, законодательство о защите интеллектуальной собственности и т.п.

Нефункциональные требования описывают цели и атрибуты качества. Атрибуты качества (quality attributes) представляют собой дополнительное описание функций продукта, выраженное через описание его характеристик, важных для пользователей или разработчиков. К таким характеристикам относятся:

- легкость и простота использования;
- легкость перемещения;
- целостность;
- эффективность и устойчивость к сбоям;
- нешние взаимодействия между системой и внешним миром;
- ограничения дизайна и реализации. Ограничения (constraints) касаются выбора возможности разработки внешнего вида и структуры продукта.

Требования предметной области

Требования предметной области характеризуют ту предметную область, где будет эксплуатироваться система. Эти требования могут быть функциональными и не функциональными.

Эти требования отображают условия, в которых будет эксплуатироваться программная система.

Они могут быть представлены в виде новых функциональных требований или в виде ограничений на уже сформулированные функциональные требования или в виде указаний, как система должна выполнять вычисления.

Невыполнение требований предметной области может привести к выходу системы из строя.

Требования к продукту

Требования к продукту описывают эксплуатационные свойства программного продукта. Это требования к производительности системы, объёму необходимой памяти, надёжности (определяет частоту возможных сбоев в системе), переносимости системы на разные компьютерные платформы и удобству эксплуатации.

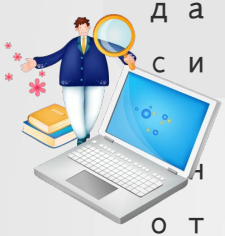
Организационные требования

Организационные требования отображают политику и организационные процедуры заказчика и разработчика ПО. Включают стандарты разработки программного продукта, требования к реализации ПО (т.е. к языку программирования и методам проектирования), выходные требования, которые определяют сроки изготовления программного продукта, и сопутствующую документацию.

Требования к интеграции

Требования к интеграции описывают низкоуровневый интерфейс взаимодействия новой системы с несколькими другими системами компании. Цель данного документа обосновать и формализовать выбор метода интеграции. Документ содержит в себе описание методов и способов интеграции с внешними системами, сервисами.

Интеграция приложений – это технологические процессы, используемые для организации обмена данными между различными информационными системами с помощью средств интеграции, предоставляемыми приложениями. К средствам интеграции, предоставляемыми приложениями относятся API функции, пакеты обработки и



Интеграция через ESB

Интеграция через ESB (Enterprise Service Bus, «Сервисная шина предприятия») применяется для обеспечения информационных систем возможностями для взаимодействия с сервисами. Использование этого метода интеграции приложений обеспечивает слабую связанность между информационными системами, так как системы взаимодействуют не напрямую, а через сервисы, размещенные на сервисной шине предприятия.

Интеграция через ESB

Основными функциями ESB являются:

- Физическое размещение сервисов.
- Размещение адаптеров к информационным системам.
- Предоставление канала для взаимодействия информационных систем.
- Обеспечение независимости от адресов, протоколов и интерфейсов при взаимодействии систем.
- Трансформация данных при взаимодействии.

Маршрутизация сообщений.

Обеспечение инфраструктурной функциональности, включая мониторинг,

Интеграция точка-точка

Интеграция приложений напрямую, является методом интеграции, при котором взаимодействие между системами происходит без применения универсального централизованного посредника, такого, как сервисная шина предприятия (ESB).

Интеграция данных

Интеграция данных – это процессы пакетного обмена данных между информационными системами, с помощью средств баз данных этих систем или экспорта–импорта файлов.

Задачи интеграции данных:

- Передачи больших объемов данных, включающая логику преобразования данных.
- Синхронизация (репликация) данных информационных систем

Интеграция ETL (Extract, Transform, Load) характеризуется следующим сценарием:

На платформе ETL пишется процесс, который

1. С помощью средств доступа к БД 1ой системы забирает из таблиц 1ой системы данные
2. С помощью средств и ресурсов БД 1ой или 2й системы или своих собственных механизмов осуществляет преобразование к структурам таблиц 2й системы
3. Загружает данные в таблицы БД 2й системы.

Интеграция данных

Файловый обмен характеризуется следующим сценарием:

1) Приложение, которому требуется передать данные другому приложению, сохра¬няет их в файле.

2) Разрабатывается интеграционное решение, которое преобразует формат файла в формат, требуемый другим приложением. (В частном случае для этого дорабатывается одна из интегрируемых систем)

3) Приложение которому нужны данные, загружает подготовленный файл.

Требования к пользовательскому интерфейсу

Пользовательский интерфейс — часть программной системы. Требования к пользовательскому интерфейсу могут быть разбиты на две группы:

- требования к внешнему виду пользовательского интерфейса и формам взаимодействия с пользователем;
- требования по доступу к внутренней функциональности системы при помощи пользовательского интерфейса.

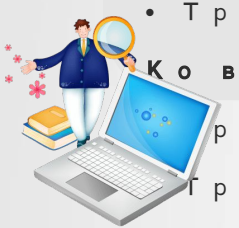
К первой группе можно отнести следующие типы требований:

- Требования к размещению элементов управления на экранных формах
- Требования к содержанию и оформлению выводимых сообщений
- Требования к форматам ввода

Ко второй группе относятся следующие типы требований:

Требования к реакции системы на ввод пользователя

Требования к времени отклика на команды пользователя



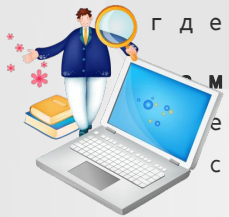
Управление требованиями

Управление требованиями — это процесс управления изменениями системных требований. Процесс управления требованиями выполняется совместно с другими процессами разработки требований. Начало этого процесса планируется на то же время, когда начинается процесс первоначального формирования требований, непосредственно процесс управления требованиями должен начаться сразу после того, как черновая версия спецификации требований будет готова.

С точки зрения разработки требования можно разделить на два класса.

Постоянные требования. Это относительно стабильные требования, которые исходят из основной деятельности организации и касаются непосредственно предметной области, где будет эксплуатироваться система.

Изменяемые требования. Эти требования отображают изменения, сделанные во время разработки системы или после ввода её в эксплуатацию.



Классификация изменяемых требований

Непостоянные требования — Требования, которые изменяются из-за изменений в окружении системы

Неожиданно возникающие требования — Требования, которые появляются во время разработки системы. В процессе проектирования может возникнуть необходимость добавления новых требований

Непрямые требования — Требования, которые являются результатом внедрения компьютерной системы, способной изменить организационные процессы и показать новые способы работы, которые приведут к новым системным требованиям

Вторичные требования — Требования, которые зависят от особенностей данной системы или от бизнес-проблем организации

Процесс управления изменениями

Анализ проблем изменения спецификации. Процесс начинается с определения проблем в требованиях или с прямого предложения внесения изменений. На этой стадии проблема или предложенные изменения анализируются для проверки их обоснованности. Затем могут быть сделаны более определённые предложения относительно изменений в требованиях.

Анализ осуществимости и расчёт их стоимости. Эффект от внесения предложенного изменения оценивается с использованием оперативного контроля. Стоимость изменений оценивается двумя показателями: стоимостью внесения изменения в спецификацию и стоимостью внесения изменений в структуру системы и непосредственно в программный код. По окончании этого этапа принимается решение, продолжать или нет внесение изменений в систему.

Реализация изменений. Реализация изменений в системной спецификации, структуре системы и программном коде.

Кто читает документацию

Заказчики системы. Определяют требования, проверяют специфицированные требования на соответствие требованиям заказываемой системы. Они могут вносить изменения в спецификацию.

Руководство компании-разработчика. Используя спецификацию для расчёта цены системы и для планирования процесса разработки системы.

Разработчики системы. Используют спецификацию в процессе разработки системы.

Инженеры, тестирующие систему. Используют спецификацию при разработке тестов, необходимых для аттестации системы.

Инженеры поддержки системы. Спецификация помогает разобраться в системе и понять, как взаимодействуют её отдельные компоненты.

Как правильно сформулировать и контролировать цель проекта?

Как и у всех путешествий, у проекта улучшения процессов должна быть цель. Если не определить конкретных целей по улучшению, люди не смогут работать согласованно, а вы не сможете сказать, есть ли движение вперед, не сможете определять приоритеты задач и сказать, когда цель достигнута.

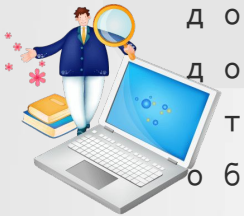
Метрика — измеримая характеристика проекта, продукта или процесса.

Ключевые показатели производительности (KPI) — это метрики, привязанные цели и служащие мерилом продвижения проекта к достижению определенной цели или результата. Набор KPI-показателей может отображаться на контрольной панели, показывая приближение к целям.

Как правильно сформулировать и контролировать цель проекта?

При определении целей по совершенствованию процессов нужно иметь в виду два обстоятельства.

- Во-первых, надо помнить, что совершенствование процесса ради самого совершенствования бессмысленно. Поэтому спросите себя, действительно ли достижение цели даст искомый рост бизнес-ценности.
- Во-вторых, не стоит разочаровывать членов команды, ставя цели, которые нереально достичь, поэтому нужно хорошенько подумать, достижима ли поставленная цель в вашей среде. Чтобы цель улучшения была разумной, ответ на оба вопроса должен быть положительным.



Как правильно сформулировать и контролировать цель проекта?

Если вы выбрали реалистичные KPI для своих целей, но не видите признаков прогресса по истечении разумного времени, нужно провести расследование:

- Правильно ли были проанализированы проблемы и выявлены их первопричины?
- Выбрали ли вы действия по улучшению, непосредственно направленные на эти первопричины?
- Был ли реалистичным план реализации этих действий по улучшению? Был ли план реализован, как планировалось?
- Изменилось ли что-то со времени исходного анализа, что должно было заставить переориентировать действия команды по улучшению?

Действительно ли члены команды приняли новые приемы работы и прошли период обучения, чтобы начать активно применять их на практике?

- Были ли поставлены реалистичные цели, которые команда

Документы процесса разработки и управления требованиями

Высокопроизводительные проекты отличаются эффективными процессами на всех этапах создания требований: выявления, анализа, спецификации, проверки и управления. Для облегчения выполнения этих процессов каждой организации необходим набор документов процесса (process assets).

Любой процесс определяют выполняемые действия и получаемые результаты; документы процесса помогают команде выполнять процессы последовательно и эффективно. Эти документы позволяют участникам проекта понять, какие шаги им следует предпринять и каких результатов от них ждут.

Документы процесса разработки и управления требованиями

Документы процесса для разработки требований	Документы процесса управления требованиями
• Процесс разработки требований • Процедура назначения требований • Процедура назначения приоритетов требованиям • Документ о концепции и границах • Шаблон вариантов использования • Шаблон спецификации требований к ПО • Контрольный список рецензирования требований	• Процесс управления требованиями • Процедура отслеживания состояния требований • Процесс управления изменениями • Шаблон устава совета по управлению изменениями • Контрольный список анализа влияния, оказываемого изменениями • Процедура отслеживания требований

Документы процесса разработки и управления требованиями

Документы процесса разработки требований:

- **Процесс разработки требований** описывает, как определить и классифицировать заинтересованных в проекте лиц в предметной области, а также как планировать действия по выявлению требований. Кроме того, здесь перечислены различные документы, касающиеся требований, модели, которые, как ожидается, будут созданы при выполнении проекта. Процесс разработки требований также определяет особенности анализа и проверки требований.
- **Процедура назначения требований** описывает, как назначать высокоуровневые требованиям к продукту конкретным подсистемам при разработке систем, состоящих из программных и аппаратных компонентов или множественных программных подсистем.

Процедура назначения приоритетов требований описывает приемы и инструменты, используемые для

Документы процесса разработки и управления требованиями

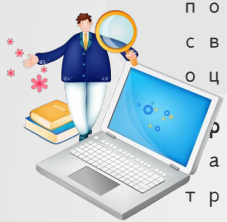
- **Шаблон документа о концепции и границах проекта** помогает куратору проекта и бизнес-аналитику осмысливать бизнес-цели, метрики успех, концепцию продукта и другие элементы бизнес-требований.
- **Шаблон вариантов использования** задает стандартный формат для описания задач, которые пользователям необходимо выполнять с помощью программы.
- **Шаблон спецификации требований к ПО** представляет собой структурированный, последовательный способ организации функциональных и нефункциональных требований. Подумайте о возможности применения более чем одного шаблона, чтобы учесть различные типы и масштабы проектов, выполняемые вашей организацией.

Контрольный список рецензирования требований. Официальное рецензирование документов, содержащих требования, — мощное средство повышения качества ПО. В списке рецензирования описано много типичных ошибок,

Документы процесса разработки и управления требованиями

Документы процесса управления требованиями

- **Процесс управления требованиями** описывает действия работающей над проектом команды для различения версий требований, определения базовых версий, работой с изменениями требований, различными версиями документации по требованиям, учетом и отчетностью о состоянии требований и накоплением информации по отслеживанию.
- **Процедура отслеживания состояния требований** подразумевает мониторинг состояния каждого функционального требования и отчетность по нему.
- **Процесс управления изменениями** определяет пути предложения, передачи, оценки и разрешения нового требования или его модификации.
- **Шаблон устава совета по управлению изменениями.** Устав совета по управлению изменениями описывает состав, функции и рабочие процедуры этого совета.
- **Контрольный список анализа последствий изменений в требованиях.** Анализ последствий помогает вам рассмотреть задачи, побочные эффекты и риски, связанные с реализацией каждого изменения в требованиях, а также оценить объем работы по задачам.
- **Процедура отслеживаемости требований** определяет, кто предоставляет данные по отслеживаемости, которые позволяют отслеживать связи требований с другими артефактами проекта, кто их собирает и управляет ими и где они хранятся.



Формирование и классификация требований

Сбор требований — это один из самых важных этапов процесса создания любой информационной системы, будь то десктопное, веб или мобильное приложение или же просто доработка уже существующего решения. Прежде, чем начать собирать требования, необходимо выявить всех заинтересованных лиц (стейкхолдеров), которые будут пользоваться системой. Чем точнее будет этот список, тем полнее будут требования.

Формирование и классификация требований

Стейкхолдерами могут быть любые физические лица и/или организации, которые активно участвуют в нашем проекте, и чьи интересы могут быть затронуты не только в процессе создания системы, но и непосредственно по завершению самого проекта. Ими могут быть менеджеры, начальники отделов, директора, любые сотрудники организации, которые будут хоть как-то взаимодействовать с готовым решением, и чьи требования (пожелания, идеи, потребности, проблемы) будем собирать.

Существует множество различных техник сбора требований, которые помогут лучше понять, что же хочет заказчик.

Анкетирование

Данный способ подразумевает под собой составление листа-опросника (анкеты, брифа), который может содержать открытые (требуют от опрашиваемого сформулировать его ответ) и закрытые (требуют от опрашиваемого выбрать ответ из предложенных вариантов) вопросы.

Анкетирование используется для того, чтобы подтвердить или детализировать ранее известные требования, выбрать параметры для решений.

Самым известным примером анкетирования может быть “Бриф на разработку сайта” — анкета содержащая список основных требований и информацию о будущем сайте.

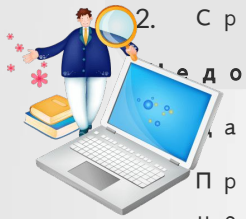
Преимущества:

1. Высокая скорость получения результатов.
2. Сравнительно небольшие материальные затраты.

Недостатки:

Данный метод не подходит для выявления неявных требований.

При составлении опросника физически невозможно учесть все необходимые вопросы.



Интервью

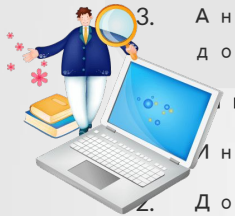
Этот метод известен многим, своего рода беседа “по душам "с заинтересованным лицом, тет-а-тет. Необходимо задавать открытые вопросы для получения информации и закрытые для того, чтобы подтвердить или опровергнуть конкретные варианты требований.

Данный способ применяется, в основном, для получения информации по какой-либо конкретной теме и/или для уточнения требований.

Многим может показаться этот способ достаточно легким, но это не так. Провести хорошее интервью достаточно сложно. Вы должны гибко реагировать на реакцию интервьюируемого и в случае необходимости изменять порядок заготовленных вопросов или их формулировку. Не забудьте включить диктофон во время интервью или вести заметки.

Из плюсов:

1. Возможность задавать вопросы в произвольной последовательности.
2. Возможность использовать вспомогательный материал.
3. Анализ невербальной реакции опрашиваемого человека, позволит сделать дополнительный вывод о достоверности его ответов.



инусов:

Интервью отнимает достаточно много времени и сил.

Дополнительной сложностью является получение одинаковых ответов от интервьюируемых.

Автозапись

Данный метод подразумевает под собой работу с записями, письмами (электронными письмами), а также с любым другим документом, автором которого является Заказчик или конечный пользователь (т.е. стейкхолдер).

В действительности это может быть и документ и наговоренная на диктофон последовательность действий или самая обычная салфетка, на которой заказчик набросал свои идеи/проблемы/пожелания, которые необходимо превратить в полноценные требования, согласовать и передать в разработку.

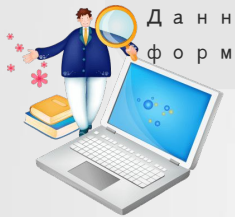
Примером такого метода может быть работа с концепцией или видением проекта (сам Заказчик любит называть это — «ТЗ»), которую он прислал вам на момент начала работ по аналитике.

Преимущество:

Помогает лучше понять сложные процедуры или процессы.

Недостаток:

Данный метод сильно зависит от опыта Заказчика, а также от его умения формулировать и выражать свои мысли.



Изучение существующей документации

Данная методика может быть использована при наличии в организации документации, которая может помочь в определении потребностей Заказчика. Примеры документации включают в себя: регламенты, описания процессов, структура организации, спецификации продукта, различные процедуры, стандарты и инструкции, шаблоны документов и т.д.

Выявленные требования являются основой для дальнейшего анализа и должны быть детализированы.

Данная методика применима, например, при автоматизации устоявшихся в организации регламентированных бизнес процессов.

Плюс:

Быстрое получение информации.

Минус:

Данный способ не применим при наличии в компании только базовых документов (или при их полном отсутствии) или, если в компании Заказчика не поддерживается актуальность документации.

Повторное использование спецификации

Повторно использовать спецификации можно в том случае, если есть уже завершенные один или несколько подобных проектов.

Техническое задание, подготовленное на предыдущем проекте может быть использовано для другого проекта с целью сократить продолжительность сбора, анализа и разработки требований, что позволит быстрее начать разработку.

Например, ТЗ для интернет магазинов похожи друг на друга и содержат одинаковые требования.

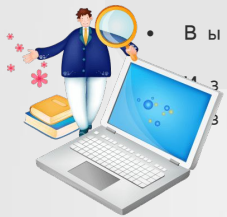
В большинстве случаев только часть документации актуальна для нового проекта, поэтому потребуется тщательная проверка требований на соответствие текущим целям и задачам Заказчика.

Преимущество:

Сокращение времени на разработку документации.

Недостатки:

- Высокая стоимость первого проекта.
- Излишняя детализация требований, может привести к их дорогостоящим изменениям в будущем.



Представитель заказчика в компании разработчика

Один из наиболее эффективных методов сбора требований, поскольку позволяет получать от представителя Заказчика своевременную оценку прогресса, корректности реализации, в короткие сроки получать обратную связь (фидбек) и дополнительную информацию для корректировки и разработки требований.

Метод часто применяется для сбора и управления требованиями при итерационной разработке, позволяет оперативно собирать, согласовывать и дорабатывать требования.

В дополнение можно сказать, что наличие представителя заказчика в компании разработчика является одним из главных правил Agile.

Преимущество:

Быстрое получение обратной связи и информации от Заказчика.

Недостатки:

- Достаточно высокая цена для Заказчика.
- Затраты по времени на адаптацию сотрудника.

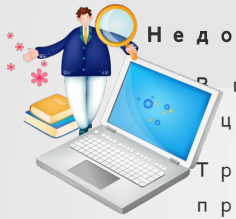
Работа «в поле»

Работа «в поле» состоит из наблюдения за деятельностью и процессами будущих пользователей системы, и в определении требований, основанных на этом наблюдении. Если говорить проще, то это наблюдение за тем, как работают пользователи, и документирование процесса, задач и результатов их деятельности.

Метод позволяет избежать проблем, связанных с трудностями стейкхолдеров в описании и выражении своих потребностей. В некоторых случаях процесс наблюдения может сопровождаться «интервьюированием» пользователей для уточнения особенностей и деталей их работы и задач. В процессе наблюдения можно так же выявить пути оптимизации бизнес процессов Заказчика.

Преимущества:

1. Позволяет наглядно увидеть проблему и разработать наиболее оптимальный вариант ее решения.
2. Помогает наиболее точно собрать требования, наблюдая за работой сотрудников.



Недостатки:

В процессе наблюдения могут быть упущены некоторые альтернативные сценарии бизнес процесса.

Трудно применим на секретных предприятиях или опасных (вредных) производствах.

Обучение

Обучение — это процесс, в котором Заказчик или любой другой человек из организации Заказчика, знающий процесс, обучает аналитика по принципу учитель — ученик.

Метод полезен, когда процессы и деятельность сотрудников Заказчика трудно описать с помощью других методов или Заказчик не может предоставить адекватное описание требований.

Обучение позволяет лучше понять сложные бизнес-процессы, а также преодолеть трудности, связанные с нехваткой абстрактного мышления и самовыражения у будущих пользователей системы.

Преимущество:

Позволяет понять сложный бизнес процесс, что позволяет предложить наилучшее решение.

Недостатки:

- Высокая стоимость и длительность.
- Метод неприменим на опасных (вредных) производствах.

Мозговой штурм

Мозговой штурм — наиболее часто используемый метод получения требований, которые связанны с новыми или плохо изученными направлениями деятельности организации Заказчика или функциями системы.

Он позволяет собрать множество идей от различных заинтересованных лиц (стейкхолдеров) в кратчайшие сроки и практически бесплатно.

Во время мозгового штурма участники «накидывают» любые идеи, касающиеся решения данной проблемы.

С помощью этой методики можно проработать несколько различных вариантов решения заданной проблемы, а также разрешить конфликты требований.

Плюсы:

Позволяет генерировать множество (в том числе и нестандартных) вариантов решений, а также позволяет участникам развивать идеи друг друга.

Минусы:

- Участники мозгового штурма должны быть мотивированы на идеи.
- Трудно применим в распределенных командах.

Совещание

Совещание — встреча, ориентированная на обсуждение конкретных вопросов, которые были определены и озвучены участникам заранее.

На такие встречи привлекаются люди, которые придерживаются различных точек зрения по текущей проблеме и могут помочь описать требования, основываясь на взглядах с разных сторон. В процессе совещания уточняется общий список требований, выявляются скрытые требования и решаются конфликты требований.

Совещания являются одной из ключевых практик в Agile, т.к. в них участвуют все стороны, заинтересованные в развитии проекта и решении проблемы.

Плюсы:

Позволяет развить и детализировать требования, определить приоритеты.

Недостатки:

- Сложности в организации встречи, если команда географически разделена, могут возникнуть трудности с присутствием всех необходимых людей на совещании.

  Консенсус необязательно будет достигнут.

Use case

Use cases или варианты использования позволяют собрать и сформировать функциональные требования от лица участников Диаграммы вариантов использования определяют границы решения и показывают связи с внешними системами и участниками.

Метод позволяет детализировать требования с точки зрения пользователей, а также помогает уточнить и систематизировать функционал, который требуется реализовать.

Плюсы:

Позволяет проработать все варианты развития сценария (основной и альтернативные сценарии)

Минусы:

Метод не применим для сбора нефункциональных требований.

Вывод

Комбинирование методик позволяет повысить эффективность сбора требований, а так же избежать их «потери».

При сборе требований необходимо помнить, что важны не только функциональные требования (ЧТО делает система), но и нефункциональные (КАК система это делает).

Тщательно собранные требования минимизируют риски проекта, т.к. позволяют сформировать четкий и понятный базис для разработки системы.

SRS

Как правило, заказчики и разработчики говорят на разных языках. Клиент представляет внешнее поведение системы: что она будет делать и как с ней будут работать конечные пользователи. Программисты же думают о продукте с точки зрения его внутренних характеристик.

Понять друг друга им помогает бизнес-аналитик, он превращает потребности клиента в требования, а требования в задачи для разработчиков. Первоначально это делается путем составления спецификаций требований к программному обеспечению (Software requirements specification или SRS).

Что такое SRS?

Software requirements specification — один из самых важных документов в разработке программного обеспечения. Он описывает работу ПО, его функции и нагрузки. Проще говоря, SRS предоставляет всем участникам дорожную карту для проекта.

Спецификация требований программного обеспечения описывает функциональные и нефункциональные требования. Часто в документ включают варианты использования, которые иллюстрируют, как пользователь будет взаимодействовать с системой.

Преимущества SRS

- Software requirements specification является основой проекта. Документ закладывает базу, которой будут следовать все участники команды разработки.
- Спецификации требований к программному обеспечению — это способ более четкой коммуникации. Этот инструмент помогает быть уверенным в том, что все участники процесса правильно понимают друг друга.
- Написание SRS также может минимизировать общее время и затраты на разработку. Команды разработчиков встроенных систем особенно выигрывают от использования SRS.
- Такая документация помогает избежать дальнейших улучшений и изменений в проекте, которые задерживают завершение или приводят к дополнительным расходам.

Как выглядит SRS

Структура SRS изменяется в зависимости от проекта, но всегда включает функциональные и нефункциональные требования. Есть шаблоны, по которым составляется структура спецификации требований к ПО, но нет строгих правил. Поэтому для стандартных шаблонов изменения скорее необходимы.

Примеры:

Роль

Если система предполагает несколько ролей, то под каждую
ем, как работа

Роль: ученик

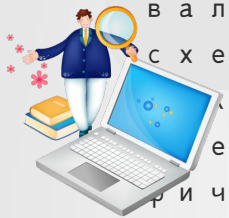
Роли и права доступа

Функция	Авторизованный пользователь	Неавторизованный пользователь
Авторизация в системе	-	+
Восстановление пароля	-	+
Выход из системы	+	-
Регистрация	-	+

Как выглядит SRS

Блок/фича

Функциональности обычно представляем в виде блоков или таблицы, которая включает в себя три раздела — это пользовательская история, бизнес-правила (UseCases) и валидация (на схеме казываем, что ебования к риче выполнены).



Авторизация через форму на сайте

Пользовательская история	**Я, как неавторизованный, но зарегистрированный пользователь, должен иметь возможность войти в личный кабинет, чтобы начать использовать сервис.**

Бизнес-правила	• ученик заходит на сайт → нажимает кнопку “Войти” → появляется окно входа
	• ученик вводит логин и пароль
	  ○ если ученик забыл пароль, то он нажимает кнопку “Восстановить пароль”:
	  ■ система отправляет на email ученика письмо с ссылкой для восстановления пароля
	  ■ ученик переходит по ссылке из письма → задает новый пароль → нажимает кнопку “Сохранить” → переходит на главную страницу сайта → нажимает кнопку “Войти” → появляется окно входа
	  ■ ученик вводит логин и пароль
	• Система проверяет зарегистрирован ли пользователь с указанными данными:
	  ○ зарегистрирован → ученик авторизуется в системе и переходит в личный кабинет
	  ○ незарегистрирован → ученик переходит к форме регистрации

Валидация	

Поле	Условие	Сообщение об ошибке
Все поля	Обязательные	Пожалуйста, заполните все поля, чтобы войти в личный кабинет
Email	Пользователь существует в системе	Неверный email или пароль
	email соответствует формату:	Неверный формат email

Как выглядит SRS

Пользовательская история

Этот раздел отображает сценарий использования конкретной фичи. Подробнее о пользовательских историях можно узнать здесь.

Регистрация

Пользовательская история	**Я, как неавторизованный и незарегистрированный пользователь, должен иметь возможность зарегистрироваться, чтобы начать использовать сайт.**

Как выглядит SRS

UseCases (Бизнес-правила)

Внутри пользовательских историй мы размещаем бизнес-правила или UseCases. Это перечень условий, при котором фича будет работать так, как нужно клиенту.

Бизнес-правила	• ученик попадет на сайт двумя способами:
	1) самостоятельно заходит на сайт;
	2) переходит на сайт по ссылке, которую получает по email, через месенджеры или соцсети от владельца курса/менеджера по продажам;
	• на главной странице сайта ученик нажимает кнопку “Регистрация” → переходит к форме регистрации:
	  ○ ученик может заполнить форму регистрации
	  ○ ученик может использовать данные соцсети → кнопка соцсети
	• ученик должен принять условия пользовательского соглашения (текст соглашения доступен по ссылке):
	  ○ если ученик принимает условия соглашения:
	  ■ появляется окно с сообщением “Мы отправили Вам письмо. Пожалуйста, проверьте Вашу почту” и ссылкой на почтовый сервис пользователя
	  ■ на email приходит письмо с ссылкой → ученик переходит по ссылке → подтверждает адрес эл. почты
	  ■ ученик переходит на сайт → нажимает кнопку “Зарегистрироваться” → переходит в личный кабинет
	  ○ если ученик НЕ принимает условия соглашения:
	  ■ регистрация на платформе невозможна → ученик переходит на главную страницу сайта

Как выглядит SRS

Разработкой такого документа, как правило, занимаются бизнес-аналитики. Уже было сказано о том, что часто заказчик вовлечен в процесс, чтобы уже на ранних этапах формировался продукт, который точно будет соответствовать ожиданиям. Опыт создания SRS в сотрудничестве с клиентом тоже полезен — это позволяет вносить изменения в фичи, когда они еще на «бумаге», а не в разработке.

Как написать хорошую SRS?

Хорошая SRS должна соответствовать нескольким ключевым характеристикам.

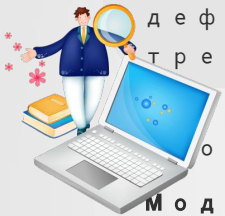
Верный: Важно убедиться, что SRS всегда отражает функциональность и технические характеристики продукта.

Однозначный: Лучше быть излишне конкретным, чем двусмысленным. SRS не является литературным шедевром, поэтому даже самые элементарные стилистические правила можно игнорировать во имя ясности.

Завершенный: Никогда не стоит упускать какую-либо функцию, запрошенную клиентом.

Последовательный: Все сокращения и определения должны использоваться единообразно во всем SRS.

Рейтинг по важности и/или стабильности: Время часто является дефицитным ресурсом в процессе разработки, поэтому ранжирование требований по их важности и стабильности – хорошая идея.



веряемый: Для каждого требования должен быть свой метод оверки.

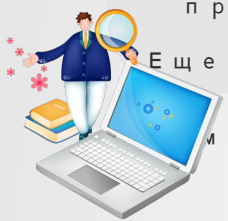
Модифицируемый: Изменения в требованиях следует вносить систематически, а их влияние на другие требования следует

Зачем мы используем SRS?

Наличие четкого набора требований гарантирует, что команда разработчиков создаст программное обеспечение, отвечающее потребностям клиента. SRS поможет оценить стоимость работ и охватить объем проекта. Он также дает программистам представление о технологическом стеке, который им понадобится, и помогает планировать работу.

Но это еще не все:

- Дизайнеры получают представление о проекте через документы SRS, чтобы они могли адаптировать дизайн к варианту использования.
- Тестировщики получают рекомендации по созданию тестовых примеров, соответствующих потребностям бизнеса.
- На основании SRS можно составить содержательную презентацию для инвесторов: бизнес-процессы легко визуализировать для грамотной презентации проекта.



Еще SRS важен, потому что это единый источник информации, который предотвращает недопонимание как между менеджером проекта и командой, так и между заказчиком и аутсорс-компанией.

Моделирование (IDEF, UML, ARIS). Понятие модели

Модель представляет искусственный, созданный человеком объект любой природы (умозрительный или материально реализованный), который замещает или воспроизводит исследуемый объект.

Процесс построения, изучения и применения моделей называется моделированием.

Модель — упрощенный, приближенный образ, который отражает наиболее существенные (с точки зрения цели моделирования) свойства оригинала.

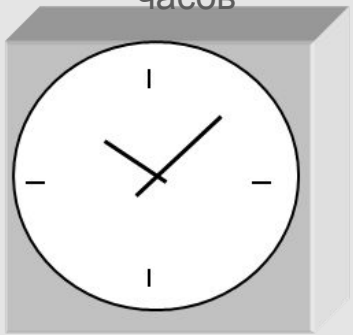
Соответствие модели оригиналу называется адекватностью модели.

Адекватность включает требования полноты и точности (правильности). Требования должны выполняться в той мере, которая достаточна для достижения цели.

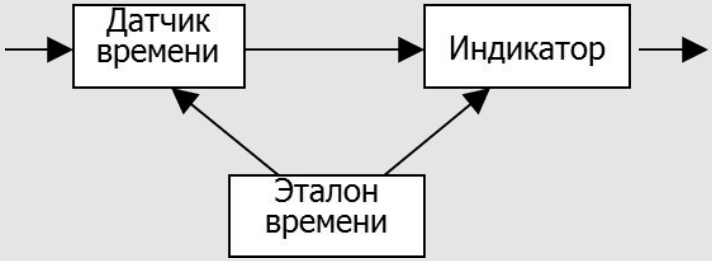
Моделирование (IDEF, UML, ARIS). Понятие модели

Для одного и того же объекта может быть построено множество различных моделей, отвечающих различным целям.

Модель внешнего вида часов



Структурная схема часов



Моделирование (IDEF, UML, ARIS). Понятие модели

Виды подобия:

- прямое (макет, фотография),
- косвенное (подобие по аналогии),
- условное (на основе соглашений).

Процесс моделирования имеет свойство динамичности: модели развиваются, уточняются, переходят одна в другую.

Классификация моделей

Познавательные (объяснительные) модели отражают уже существующие объекты.

Нормативные (прагматические) модели отражают объекты, которые должны быть осуществлены.

Градации нормативных моделей: от референтной (для целого класса объектов) до модели конкретного объекта.

Классификация моделей

Статические модели не учитывают временной фактор.

Динамические модели отражают изменения объекта, происходящие с течением времени.

Классификация моделей

Материальные модели построены из реальных объектов.

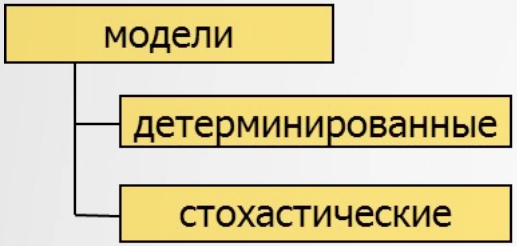
Абстрактные модели — это идеальные конструкции, выполненные средствами мышления, сознания.

Классификация моделей

Декларативные модели отражают свойства, структуры, состояния объектов.

Процедурные модели отражают процедурное, операционное знание.

Классификация моделей



Детерминированные модели отражают процессы и явления, не подверженные случайностям.

Стохастические – отражают случайные процессы, описываемые вероятностными характеристиками и статистическими закономерностями.

Классификация моделей

Формализованные модели могут не иметь смысловой интерпретации.

В содержательных моделях сохраняется семантика моделируемого объекта.

Языки описания моделей

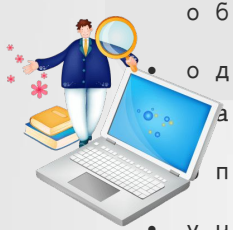
Языки описания моделей: аналитические, численные, логические, теоретико-множественные, лингвистические, графические.

Графические модели (схемы, диаграммы, графики, чертежи) – наглядны.

Нотация — система условных обозначений (знаков) и правил их использования, принятая в конкретной методологии.

Требования к нотации:

- простота — простой знак предпочтительнее сложного;
- наглядность — хотя бы отдаленное сходство с оригиналом;
- индивидуальность — достаточное отличие от других обозначений;
- однозначность — нельзя обозначать одним символом различные объекты;
- определенность — четкие правила использования модели;
- учет устоявшихся традиций.



Содержание модели бизнеса

В модели бизнеса отражают:

- функции, которые бизнес–система должна выполнять — что она делает, для кого, с какой целью;
- процессы, последовательность отдельных шагов процессов (работ, операций);
- организационные структуры, обеспечивающие выполнение процессов;
- материальные и информационные потоки, возникающие в ходе выполнения процессов;
- данные, необходимые при выполнении процессов, и отношения между этими данными.

Методы моделирования бизнеса
Структурные методы

Основаны на последовательной декомпозиции системы на все более мелкие подсистемы.

Методы моделирования бизнеса
Структурные методы

Принципы структурного подхода:

«разделяй и властвуй» — разбиение сложных проблем на множество меньших задач, легких для понимания и решения;

иерархическое упорядочивание – организация составных частей проблемы в иерархические древовидные структуры.

Две группы методов: моделирующие функциональную структуру и структуру данных

Наибольшее распространение получили методологии:

- IDEF0 – функциональные модели, основанные на методе SADT;
- IDEF1X – диаграммы данных «сущность–связь» (ERD);
- IDEF3 — диаграммы потоков работ (Work Flow Diagrams);
- DFD — диаграммы потоков данных (Data Flow Diagrams).

Методы объектно-ориентированного моделирования

Предназначены для создания моделей систем с целью их последующей реализации в виде объектно–ориентированных программ

Методы объектно-ориентированного моделирования

Наиболее известные методы:

- Booch’93 Г. Буча,
- OMT Дж. Румбаха
- OOSE А. Джекобсона
- UML (Unified Modeling Language) – на основе Booch’93, OMT, OOSE

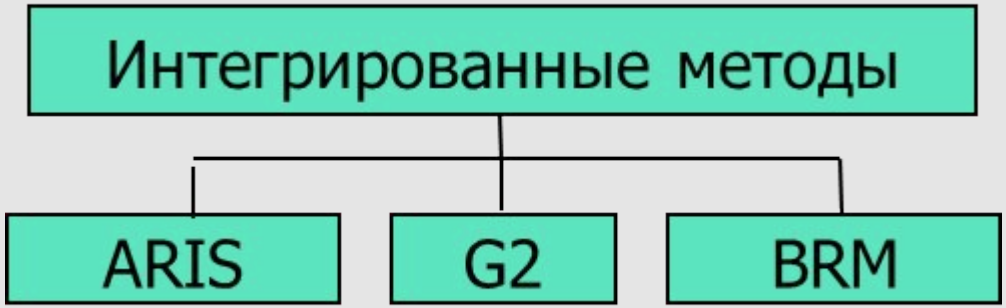
Главным структурообразующим элементом является объект.

В программировании объект — это структура, объединяющая данные и процедуры.

В модели бизнеса объекты – это участники бизнес–процесса (активные объекты) и пассивные объекты (материалы, документы), над которыми выполняют действия активные объекты.

Методы имитационного моделирования

Позволяют имитировать на компьютере (с помощью специальных программ) процессы функционирования реальной системы (в режиме сжатого времени или пошаговом режиме).



Методы имитационного моделирования

Наиболее распространенные методы:

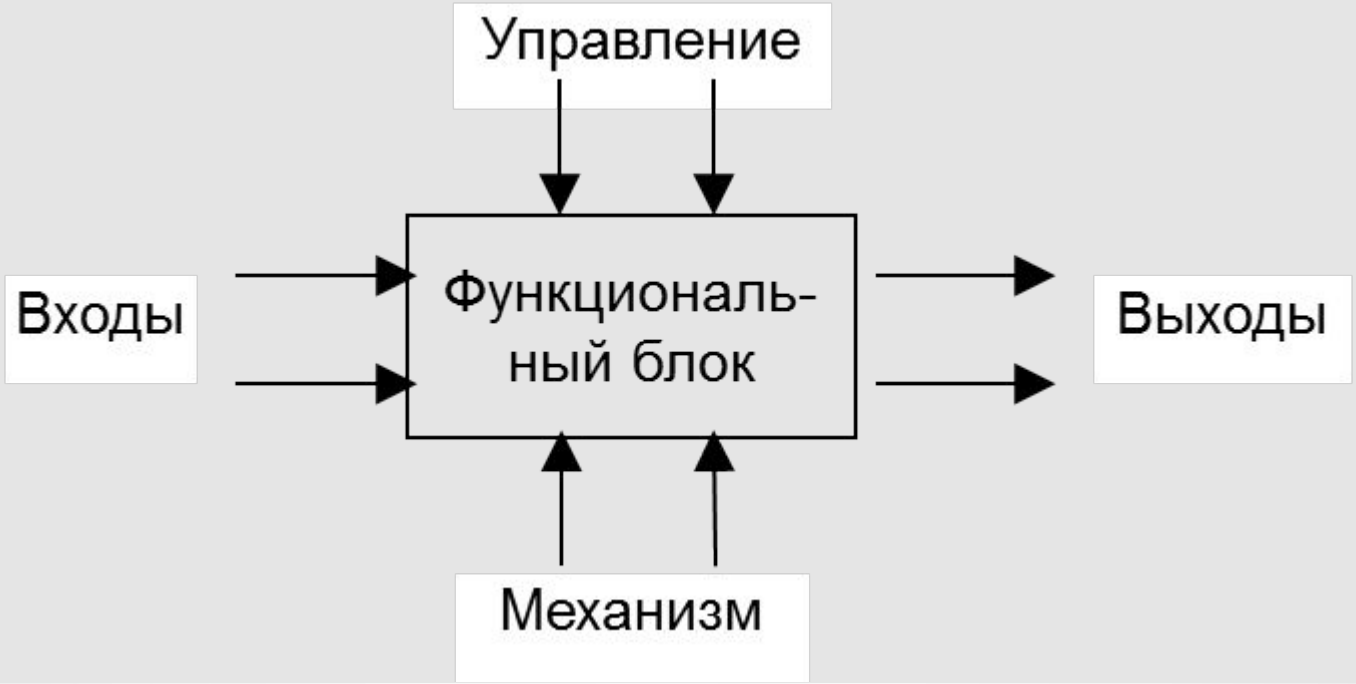
- сети Петри и раскрашенные сети Петри (CPN, Colored Petri Nets);
- GPSS (General Purpose Simulating System) – унифицированный язык имитационного моделирования;
- SIMAN (SIMulation ANalysis) – язык визуального моделирования.

Интегрированные методы

Интегрированные методы моделирования объединяют различные виды моделей – структурного анализа, объектно–ориентированные, имитационные и др.

- ARIS (Architecture of Integrated Information System) позволяет отражать в единой интегрированной модели: оргструктуры, функции, данные, процессы. Использует множество типов моделей.
- G2 — методология создания динамических интеллектуальных систем позволяет моделировать процессы с использованием знаний эксперта.
- BRM (Business Rules Management) – методология управления бизнес–правилами.

Структурные методологии
Методология IDEF0



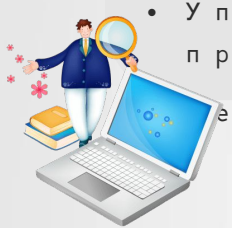
Структурные методологии
Методология IDEF0

Методология IDEF0 базируется на методе SADT (Structured Analysis and Design Technique) Росса, предназначенном для структурированного представления функций системы и анализа системных требований.

IDEF0-модель состоит из диаграмм и фрагментов текста. На диаграммах все функции системы и их взаимодействия представлены как блоки (функции) и дуги (отношения).

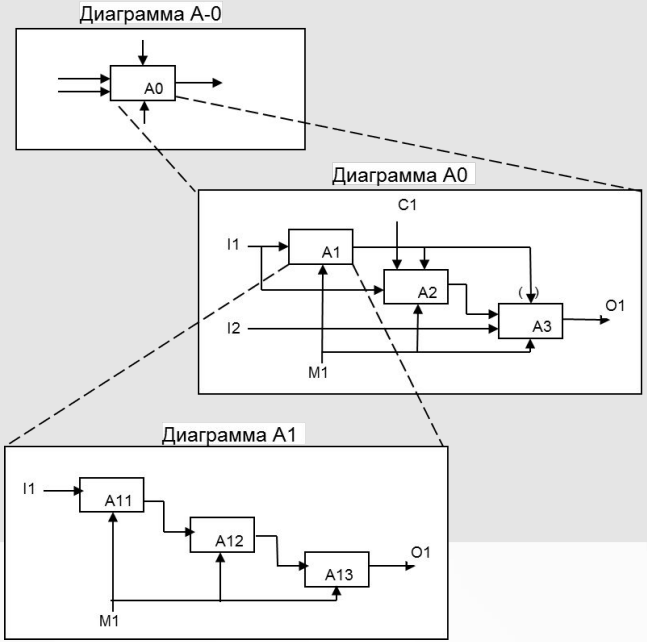
Основные элементы модели:

- Функциональный блок (Activity) - преобразование (активность);
- Выходы (Output) - результат преобразования;
- Входы (Input) — объекты, которые преобразуются в Выходы;
- Управление (Control) — информация, как происходит преобразование;
- Механизм (Mechanism) - объекты, осуществляющие преобразование.



Структурные методологии
Методология IDEF0

Функциональный блок может быть декомпозирован — представлен в виде совокупности других взаимосвязанных блоков, которые детально описывают исходный блок.



Структурные методологии
Методология IDEF0

Таким образом, **IDEF0-модель** состоит из набора иерархически связанных диаграмм

На диаграмме блоки соединяются дугами: выходные дуги одних блоков могут являться входами (управлением, механизмом) других.

Дуги с одним свободным концом имеют источник или получатель вне диаграммы. Для обозначения внешних дуг используются буквы:

- I (Input),
- C (Control),
- O (Output) и
- M (Mechanism).

Типы связей между блоками:

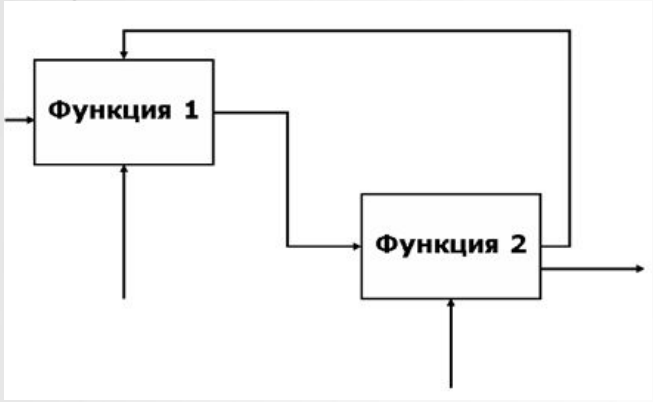
Выход-вход

Выход-управление

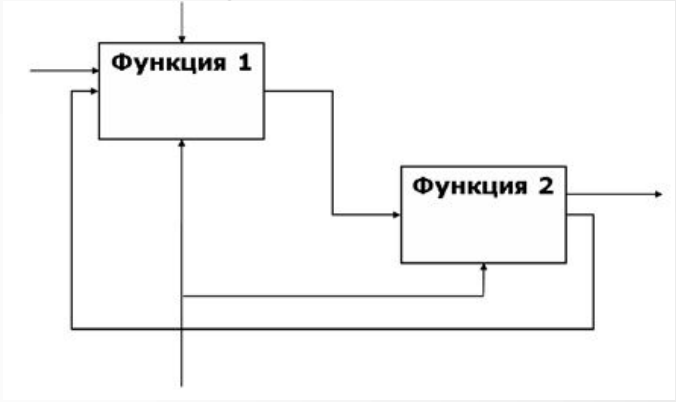
Выход-механизм

Типы связей между блоками:

Обратная связь по управлению



Обратная связь по входу



Объектно-ориентированный язык UML

Язык UML был разработан для создания моделей информационных систем (ИС) с целью их последующей реализации в виде объектно-ориентированных программ.

Все представления о модели сложной системы фиксируются в виде диаграмм – специальных графических конструкций (схем, графов).

Имеется 8 основных типов диаграмм UML, отражающих различные аспекты: процессы, выполняемые системой (предоставляемые пользователю сервисы), последовательность выполняемых системой алгоритмических операций, структуру программных объектов, их взаимодействие (обмен сообщениями) и т.д.

В настоящее время язык UML применяется не только для создания ИС, но и для анализа и перепроектирования бизнес-процессов:

- вместо моделей процессов ИС строятся модели бизнес-процессов,
- вместо программных объектов в моделях отражаются объекты бизнес-процессов (исполнители, продукция, услуги и т.д.),
- вместо окружения ИС (пользователей ИС) моделируется окружение бизнеса (поставщики, партнеры, клиенты).