

Министерство науки и высшего образования РФ
Сибирский государственный университет науки и
технологий имени М.В. Решетнева

Лабораторный практикум по дисциплине
«Управление жизненным циклом ИС»

Для магистрантов направлений
09.04.01 «Информатика и вычислительная техника» и
09.04.04 «Программная инженерия»

Красноярск, 2021

Оглавление

Введение.....	6
1 Лабораторная работа № 1 «Разработать описание проекта в терминах системного анализа»	7
1.1 Введение	7
1.2 Основы теории прикладного системного анализа.....	7
1.2.1 Определения.....	7
1.2.2 Когда возникает проблема?	7
1.2.3 Алгоритм системного анализа.....	8
1.3 Структура отчета.....	9
2 Лабораторная работа № 2 «Описание бизнес-процессов объекта автоматизации».....	10
2.1 Введение	10
2.2 Основы теории моделирования бизнес-процессов.....	10
2.2.1 Цели моделирования бизнес-процессов.....	10
2.2.2 Стадии моделирования бизнес-процессов	10
2.2.3 Виды моделирования бизнес-процессов	11
2.2.4 Принципы моделирования бизнес процессов.....	11
2.3 Нотации моделирования бизнес-процессов	12
2.3.1 IDEF0	12
2.4 BPMN	13
2.5 Структура отчета.....	15
3 Лабораторная работа № 3 «Описать требования к ИС»	16
Введение.....	16
3.1 Требования к ТЗ	16
3.1.1 Часть 1 – «Содержание задания».....	16
3.1.2 Часть 2 – «Исходные данные к проекту»	17
3.1.3 Часть 3 – Календарный план выполнения работ.....	21
3.2 Структура отчета.....	22
4 Лабораторная работа № 4 «Разработать проект системы в UML»	23
4.1 Введение	23
4.2 Виды диаграмм UML.....	23
4.2.1 Нотация UML для описания логики проекта.....	23
4.2.2 Диаграмма прецедентов — Use-case diagram	24
4.2.3 Диаграмма классов — Class diagram	25
4.2.4 Диаграмма активностей — Activity diagram	25
4.2.5 Диаграмма последовательности — Sequence Diagram	26
4.2.6 Диаграмма развертывания — Deployment Diagram	26
4.3 Структура отчета.....	26
5 Лабораторная работа № 5 «Тестовые сценарии системы»	28
5.1 Введение	28
5.2 Описание тест-кейса	28
5.2.1 Стандартные атрибуты тест-кейса.....	28

5.2.2	Пример оформления (один ожидаемый результат).....	28
5.2.3	Преимущества и недостатки тест-кейсов.....	29
5.2.4	Примеры оформления (несколько ожидаемых результатов).....	29
5.2.5	Несколько вариантов вводимых данных.....	30
5.2.6	Результаты для нескольких шагов из кейса.....	30
5.2.7	Несколько проверок после одного сценария.....	31
5.3	Требования к отчету.....	32
6	Лабораторная работа № 6 «Подготовить материалы к SCRUM-проекту».....	33
6.1	Введение.....	33
6.2	Теоретическое введение в SCRUM.....	33
6.2.1	Суть SCRUM.....	33
6.2.2	Роли в SCRUM.....	33
6.2.3	Требования к продукту.....	34
6.2.4	Этапы SCRUM.....	34
6.2.5	3 ключевых «оружия» на проекте.....	35
6.2.6	На чем держать фокус?.....	35
6.3	Разработка бэклога. Пользовательские истории.....	35
6.3.1	Что такое User Stories?.....	35
6.3.2	Для чего применяется User Story?.....	35
6.3.3	Структура User Story.....	36
6.3.4	User Story: примеры.....	37
6.4	Требования к отчету.....	37
7	Лабораторная работа № 7 «План внедрения системы».....	38
	Введение.....	38
7.1	Общие сведения о разработке программного обеспечения.....	38
7.2	Процесс управления разработкой программного обеспечения.....	38
7.2.1	Планирование проекта разработки программного обеспечения.....	39
7.2.2	Общие сведения о требованиях к информационным системам.....	40
7.2.3	План проекта.....	42
7.2.4	Контрольные отметки этапов работ.....	43
7.2.5	График работ.....	43
7.2.6	Диаграммы процессов и временные диаграммы.....	44
7.3	Требования к отчету.....	47
8	Лабораторная работа № 8 «Разработать регламент сопровождения системы».....	48
8.1	Введение.....	48
8.2	Общие сведения по сопровождению ИС.....	48
8.2.1	Постановка задачи.....	48
8.2.2	Аспекты, отражаемые в регламенте сопровождения.....	48
8.2.3	Фиксация и мониторинг всех поступивших заявок.....	49
8.2.4	Универсальность регламента.....	49
8.3	Требования к отчету.....	49
9	Лабораторная работа № 9 «Разработать план управления качеством».....	50
9.1	Введение.....	50
9.2	Общая информация по управлению качеством в ИТ-проектах.....	50
9.2.1	Понятие качества.....	50
9.2.2	Показатели качества.....	50

9.2.3	Стандарты качества	50
9.3	Структура плана управления качеством.....	51
9.3.1	Предписания.....	51
9.3.2	Политика обеспечения качества	51
9.4	Порядок работы над планом управления качеством.....	51
9.5	Требования к отчету	51
10	Лабораторная работа № 10 «Отчет по управлению конфигурацией»	52
10.1	Введение	52
10.2	Процесс и инструменты управления конфигурацией	52
10.2.1	Общие соображения	52
10.2.2	Процесс управления конфигурацией программного обеспечения	52
10.2.3	Необходимость управления конфигурацией	52
10.2.4	Задачи процесса управления конфигурацией	53
10.3	Версионный контроль в ИТ-проектах	53
10.3.1	Системы версионного контроля.....	53
10.3.2	Конфигурационные единицы	53
10.3.3	Ветвление и слияние версий.....	53
10.4	Требования к отчету	54
11	Лабораторная работа № 11 «Разработка бизнес-архитектуры по А.Остервальдеру».....	55
11.1	Введение	55
11.2	Описание модели Остервальжера-Пинье	55
11.2.1	Что такое бизнес-модель Остервальдера?.....	55
11.2.2	Что нужно сделать перед построением бизнес-моделей?	55
11.2.3	Структура модели Остервальдера.....	56
11.3	Порядок заполнения бизнес-модели Остервальдера.....	56
11.3.1	Потребительские сегменты (Customer Segments).....	56
11.3.2	Ценностные предложения (Value propositions)	57
11.3.3	Каналы сбыта (Channels).....	57
11.3.4	Отношения с клиентами (Customer relationships).....	57
11.3.5	Потоки доходов (Revenue streams).....	58
11.3.6	Ключевые ресурсы (Key resources)	59
11.3.7	Ключевые деятельности (Key activities).....	59
11.3.8	Ключевые партнёры (Key partners)	59
11.3.9	Структура издержек (Cost structure)	60
11.4	Требования к отчету	60
12	Лабораторная работа № 12 «Разработка архитектурной модели предприятия»	61
12.1	1. Введение	61
12.2	Методология управления архитектурой предприятия	61
12.2.1	Основные принципы методологии АП.....	61
12.2.2	Слои, аспекты и объекты архитектуры предприятия	62
12.2.3	Деятельность по созданию и использованию моделей АП.....	63
12.2.4	Базовая часть	63
12.3	Требования к отчету	64
13	Лабораторная работа № 13 «Разработать регламент компании-разработчика ПО»	65
13.1	Введение	65

13.2	Процесс регламентации.....	65
13.2.1	Понятие регламентации	65
13.2.2	Цели регламентации.....	65
13.2.3	Проблемы регламентации.....	65
13.2.4	Типовая структура регламента бизнес-процесса.....	67
13.2.5	Особенности регламентации бизнес-процессов верхнего уровня.....	67
13.2.6	Особенности регламентации бизнес-процессов операционного уровня	68
13.2.7	Содержание регламента бизнес-процессов операционного уровня.....	68
13.3	Структура регламента бизнес-процесса	68
13.4	Учет индивидуальных особенностей организации при регламентации.....	70
13.5	Требования к отчету	70
Заключение.....		71
Список литературы.....		73

Введение

Прогресс в области развития и применения информационных систем немыслим без управления их жизненным циклом. Поэтому бакалаврам и магистрам, занимающимся исследованиями, сервисным обслуживанием и разработкой информационных систем, необходимы практические навыки управления их жизненным циклом.

Дисциплина «Управление жизненным циклом информационных систем» предназначена для ознакомления студентов с процедурами управления жизненным циклом информационных систем для направлений подготовки магистра 09.04.01 «Информатика и вычислительная техника» и 09.04.04 «Программная инженерия».

В результате выполнения лабораторных работ студенты получают следующие компетенции:

- умение формулировать требования к информационным системам;
- умение грамотно составлять техническое задание;
- иметь навыки применения программных продуктов для моделирования информационных систем.

Комплекс лабораторных работ состоит из 13 тем. Занятия должны проводиться в лабораториях, оснащённых компьютерной техникой во время лабораторных работ, а также в процессе самостоятельной работы студентов.

1 Лабораторная работа № 1 «Разработать описание проекта в терминах системного анализа»

1.1 Введение

Прикладной системный анализ стремительно становится неотъемлемой компетенцией современного специалиста. Не имеет значения о какой отрасли идет речь. Не важен уровень занимаемой должности. Системный подход помогает быстро и результативно разобраться в структуре и функционировании системы, выявить характер взаимного влияния элементов системы - преимущественно скрытый и неочевидный, а иногда даже парадоксальный. Инструменты системного анализа позволяют разработать удобную модель, которая поможет осмыслить прошлое, проанализировать настоящее и построить сценарий достижения желанного будущего.

1.2 Основы теории прикладного системного анализа

1.2.1 Определения

Система – это средство достижения цели пользователями системы (стейкхолдерами)

Система может быть специально создана или приспособлена для достижения цели.

Система обязательно имеет одно или несколько целевых свойств. Если целевых свойств несколько, то они могут иметь разную ценность (разную значимость) для пользователя системы.

Проблема – возникает, когда система не позволяет достичь поставленных перед ней целей:

- целевое свойство не достигает намеченного уровня или
- целевое свойство ухудшается или может ухудшиться.

Системный анализ – это структурированный подход к достижению цели посредством выявления и решения проблем.

1.2.2 Когда возникает проблема?

Активно используемая система притягивает к себе проблемы как магнит. Вот наиболее типичные проблемы, сопровождающие систему на протяжении всего периода ее жизненного цикла:

- Создание новой системы
- вывод на рынок новой продукции (товар, услуга),
- запуск нового оборудования,
- создание нового общественного движения,
- создание новой технологии и т.п.

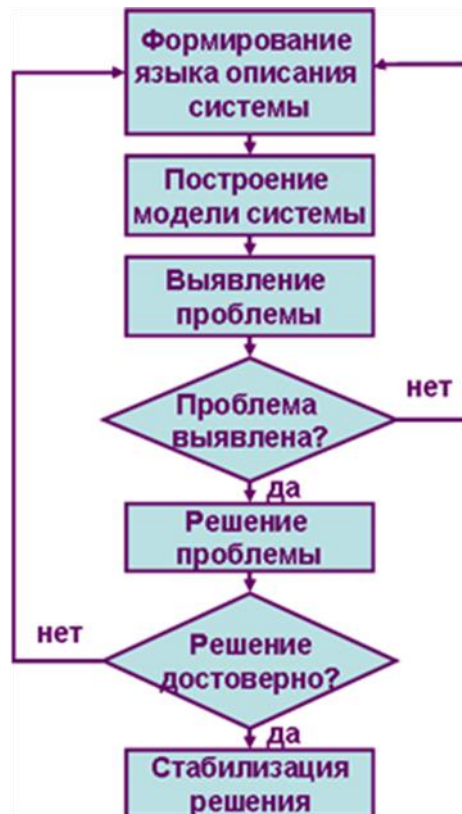
Изменение (улучшение) существующей системы - обычно связано с изменением целей пользователя системы.

- увеличение объемов производства на существующем оборудовании,
- повышение уровня лояльности потребителей,
- увеличение доли рынка,
- сокращение затрат на существующем оборудовании, и т.п.
- изменение внешних по отношению к системе условий
- изменение рынков (поставщики, потребители, рынок труда, страхование, транспортные услуги ...),
- изменение требований регулирующих органов, и т.п.

Практически всегда само существование системы обуславливает существование проблемы: сложно представить себе собственника или пользователя системы, которые не желали бы сделать свою систему более эффективной, управляемой, надежной и безопасной.

1.2.3 Алгоритм системного анализа

Прикладной системный анализ является методикой или, в некотором смысле, "технологией" улучшения систем. Как и любая технология, системный анализ предполагает последовательное выполнение определенных операций. Каждая операция имеет цель, после выполнения каждой операции ожидается вполне определенный результат. Ниже приведен алгоритм прикладного системного анализа и обозначены основные цели его этапов.



- Формирование языка описания системы
 - определение терминов описания системы
 - определение целевых свойств системы на языке описания системы
- Построение модели системы
 - установление взаимосвязей между ожиданиями стейкхолдеров и целевыми свойствами системы
 - установление взаимосвязей между внешними и внутренними факторами и параметрами системы
- Выявление проблемы
 - поиск критических параметров - параметров системы, негативно влияющих на ее целевые свойства
- Проверка: Проблема выявлена?
 - выявление корреляции значений критических параметров системы и отклонений ее целевых свойств от желаемых значений
- Решение проблемы
 - выявление коренных причин проблемы системы
 - определение действий по улучшению системы

- внедрение действий по улучшению
- Проверка: Решение достоверно?
 - выявление корреляции значений критических параметров системы и улучшения ее целевых свойств
- Стабилизация решения
 - стандартизация улучшающих действий, эффективность которых доказана
 - отслеживание корреляции состояния критических параметров системы и достигнутого результата

Качество выполнения системного анализа подтверждается эффективностью действий по изменению системы, приводящими к решению проблемы системы. Поэтому для полноценного выполнения системного анализа невозможно ограничиться только "анализом системы" (понять природу проблемы), а нужно также найти правильные улучшающие действия и воплотить их жизнь— создать или изменить систему и убедиться в эффективности предпринятых действий.

1.3 Структура отчета

- Титульный лист
- Введение
- Основная цель улучшаемой системы
- Основные задачи улучшаемой системы.
- Описание предметной области.
- Рекомендуемые сущности (таблицы данных)
- Система (определение)
- Взаимодействующие системы
- Компоненты (элементы, подсистемы) системы
- Процессы – динамическое изменение системы во времени
- Состояние – положение системы относительно других её положений (набор оцениваемых показателей)
- Системный эффект (синергия) – связи между подсистемами
- Цель – желаемые будущие состояния системы в заданный момент времени
- Граница системы – любые материальные и нематериальные ограничители, отделяющие систему от внешней среды
- Главная проблема владельца системы
- Список стейкхолдеров
- Языки конфигулятора
- Заключение

2 Лабораторная работа № 2 «Описание бизнес-процессов объекта автоматизации»

2.1 Введение

Моделирование бизнес-процессов является одним из методов улучшения качества и эффективности работы организации. В основе этого метода лежит описание процесса через различные элементы (действия, данные, события, материалы и пр.) присущие процессу. Как правило, моделирование бизнес-процессов описывает логическую взаимосвязь всех элементов процесса от его начала до завершения в рамках организации. В более сложных ситуациях моделирование может включать в себя внешние по отношению к организации процессы или системы.

Моделирование бизнес-процессов позволяет понять работу и провести анализ организации. Это достигается за счет того, что модели могут быть составлены по различным аспектам и уровням управления. В больших организациях моделирование бизнес-процессов выполняется более подробно и многограннее, чем в малых, что связано с большим количеством кросс-функциональных связей.

Обычно для моделирования бизнес-процессов применяются различные компьютерные средства и программное обеспечение. Это облегчает управление моделями, отслеживание в них изменений и позволяет сократить время анализа.

2.2 Основы теории моделирования бизнес-процессов

2.2.1 Цели моделирования бизнес-процессов

Конечная цель моделирования бизнес-процессов заключается в том, чтобы добиться улучшения работы. Для этого в ходе анализа основное внимание уделяется повышению ценности результатов процесса и снижению стоимости и времени выполнения действий.

Моделирование бизнес-процессов преследует несколько целей:

- во-первых, это цель описания процессов. За счет моделирования можно проследить, что происходит в процессах от начала, до завершения. Моделирование позволяет получить «внешний» взгляд на процессы и определить улучшения, которые повысят их эффективность.
- во-вторых, нормирование процессов. Моделирование бизнес-процессов задает правила выполнения процессов, т.е. то, каким образом они должны быть выполнены. Если следовать установленным в моделях правилам, руководящим указаниям или требованиям, то можно достичь желаемой производительности процессов.
- в-третьих, установление взаимосвязей в процессах. Моделирование бизнес-процессов устанавливает четкую связь между процессами и требованиями, которые они должны выполнять.

2.2.2 Стадии моделирования бизнес-процессов

Моделирование бизнес-процессов, как правило, включает в себя выполнение нескольких последовательных стадий. Т.к. конечной целью моделирования является улучшение процессов, то оно охватывает и «проектную» часть работы, и работы по внедрению моделей процессов.

Состав стадий, которые включает в себя моделирование бизнес-процессов следующий:

- выявление процессов и построение исходной модели «как есть». Для того чтобы улучшить процесс, необходимо понимать, как он работает в данный момент. На этой стадии определяются границы процесса, выявляются его ключевые

элементы, собираются данные о работе процесса. В результате создается исходная модель процесса «как есть». Эта модель не всегда адекватно отражает работу процесса, поэтому модель этой стадии можно назвать «первым драфтом» или исходной моделью «как есть».

- пересмотр, анализ и уточнение исходной модели. На этой стадии выявляются противоречия и дублирование действий в процессе, определяются ограничения процесса, взаимосвязи процесса, устанавливается необходимость изменения процесса. В результате формируется окончательный вариант модели «как есть».
- разработка модели «как должно быть». После анализа существующей ситуации, необходимо определить желаемое состояние процесса. Это желаемое состояние представляется в модели «как должно быть». Такая модель показывает, как процесс должен выглядеть в будущем, включая все необходимые улучшения. В ходе этой стадии моделирования бизнес-процессов и разрабатываются такие модели.
- тестирование и применение модели «как должно быть». Эта стадия моделирования связана с внедрением разработанной модели в практику деятельности организации. Модель бизнес-процесса проходит апробацию, и в нее вносятся необходимые изменения.
- улучшение модели «как должно быть». Моделирование бизнес-процессов не ограничивается только созданием модели «как должно быть». Каждый из процессов по ходу работы продолжает изменяться и совершенствоваться, поэтому модели процессов должны регулярно пересматриваться и улучшаться. Эта стадия моделирования связана с постоянным улучшением процессов и улучшением модели бизнес-процессов.

2.2.3 Виды моделирования бизнес-процессов

Моделирование бизнес-процессов может иметь различную направленность. Это зависит от того, какие проблемы предполагается решить с его помощью. Учет абсолютно всех воздействий на процесс может значительно усложнить модель и привести к избыточности описания процесса. Чтобы этого избежать, моделирование бизнес-процессов разделяют по видам. Вид моделирования выбирается в зависимости от исследуемых характеристик процесса.

Для целей совершенствования процесса применяют следующие виды моделирования:

- Функциональное моделирование. Этот вид моделирования подразумевает описание процессов в виде взаимосвязанных, четко структурированных функций. При этом строгая временная последовательность функций, в том виде, как она существует в реальных процессах, не обязательна.
- Объектное моделирование - подразумевает описание процессов, как набора взаимодействующих объектов – т.е. производственных единиц. Объектом является какой-либо предмет, преобразуемый в ходе выполнения процессов.
- Имитационное моделирование – при таком виде моделирования бизнес-процессов подразумевается моделирование поведения процессов в различных внешних и внутренних условиях с анализом динамических характеристик процессов и с анализом распределения ресурсов.

Разделение моделирования по видам выполняется для упрощения работы и концентрации внимания на тех или иных характеристиках процесса. При этом для одного и того же процесса могут быть применены различные виды моделирования. Это позволяет работать с одним видом моделей независимо от других.

2.2.4 Принципы моделирования бизнес процессов

Моделирование бизнес-процессов основывается на ряде принципов, которые дают возможность создать адекватные модели процессов. Их соблюдение позволяет описать множество параметров состояния процессов таким образом, чтобы внутри одной модели компоненты были тесно взаимосвязаны, в то время как отдельные модели оставались в достаточной степени независимыми друг от друга.

Главными принципами моделирования бизнес-процессов являются следующие:

- Принцип декомпозиции – каждый процесс может быть представлен набором иерархически выстроенных элементов. В соответствии с этим принципом процесс необходимо детализировать на составляющие элементы.
- Принцип сфокусированности – для разработки модели необходимо абстрагироваться от множества параметров процесса и сфокусироваться на ключевых аспектах. Для каждой модели эти аспекты могут быть свои.
- Принцип документирования – элементы, входящие в процесс, должны быть формализованы и зафиксированы в модели. Для различных элементов процесса необходимо использовать различающиеся обозначения. Фиксация элементов в модели зависит от вида моделирования и выбранных методов.
- Принцип непротиворечивости – все элементы, входящие в модель процесса должны иметь однозначное толкование и не противоречить друг другу.
- Принцип полноты и достаточности – прежде чем включать в модель тот или иной элемент, необходимо оценить его влияние на процесс. Если элемент не существенный для выполнения процесса, то его включение в модель не целесообразно, т.к. он может только усложнить модель бизнес-процесса.

2.3 Нотации моделирования бизнес-процессов

2.3.1 IDEF0

IDEF0 (Integration Definition for Function Modeling) – нотация описания бизнес-процессов. Основана на методологии SADT.

SADT (Structured Analysis and Design Technique, технология структурного анализа и проектирования) - графические обозначения и подход к описанию систем. Разработка SADT началась в 1969 году и была опробована на практике в компаниях различных отраслей (аэрокосмическая отрасль, телефония и т.д.). Публично появилась на рынке в 1975 г и получила очень широкое распространение в мире.

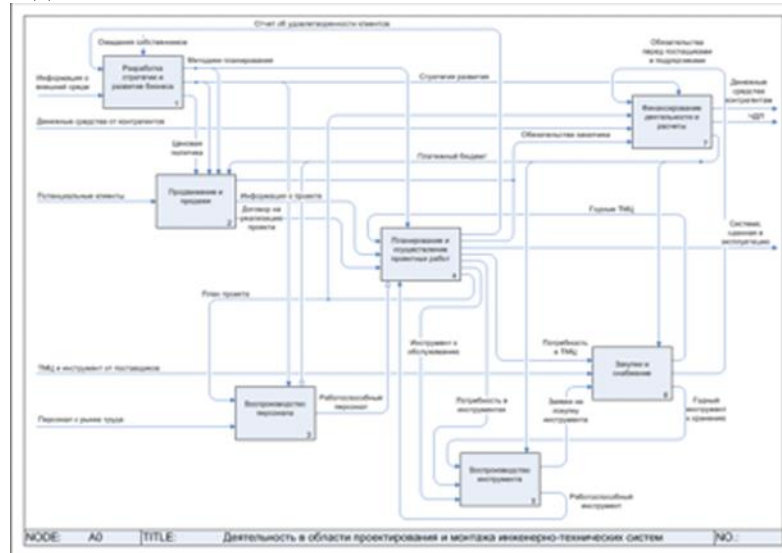
IDEF0 является результатом программы компьютеризации промышленности, которая была предложена ВВС США. Автоматизация деятельности предприятий потребовала соответствующих методик и инструментов. Перед тем, как разрабатывать программное обеспечение, необходимо было четко и понятно описать бизнес-процессы (нельзя автоматизировать хаос). Инструменты, разработанные для задач программирования, так же могут быть полезны и для задач менеджмента. Нотация может быть использована для моделирования широкого круга автоматизированных и неавтоматизированных систем.

Идея IDEF0 лежит в том, что бизнес-процесс отображается в виде прямоугольника, в которой входят и выходят стрелки.



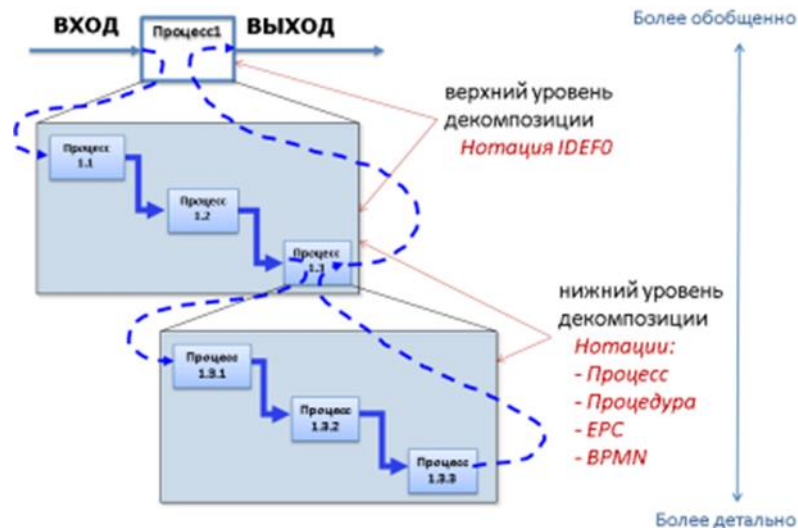
Для IDEF0 имеет значение сторона процесса и связанная с ней стрелка:

- слева входящая стрелка – вход бизнес-процесса – информация (документ) или ТМЦ, который будет преобразован в ходе выполнения процесса;
- справа исходящая стрелка – выход бизнес-процесса – преобразованная информация (документ) или ТМЦ;
- сверху входящая стрелка – управление бизнес-процесса – информация или документ, который определяет как должен выполняться бизнес-процесс, как должно происходить преобразование входа в выход;
- снизу входящая стрелка – механизм бизнес-процесса – то, что преобразовывает вход в выход: сотрудники или техника. Считается, что за один цикл процесса не происходит изменения механизма.



Пример диаграммы бизнес-процесса в нотации IDEF0

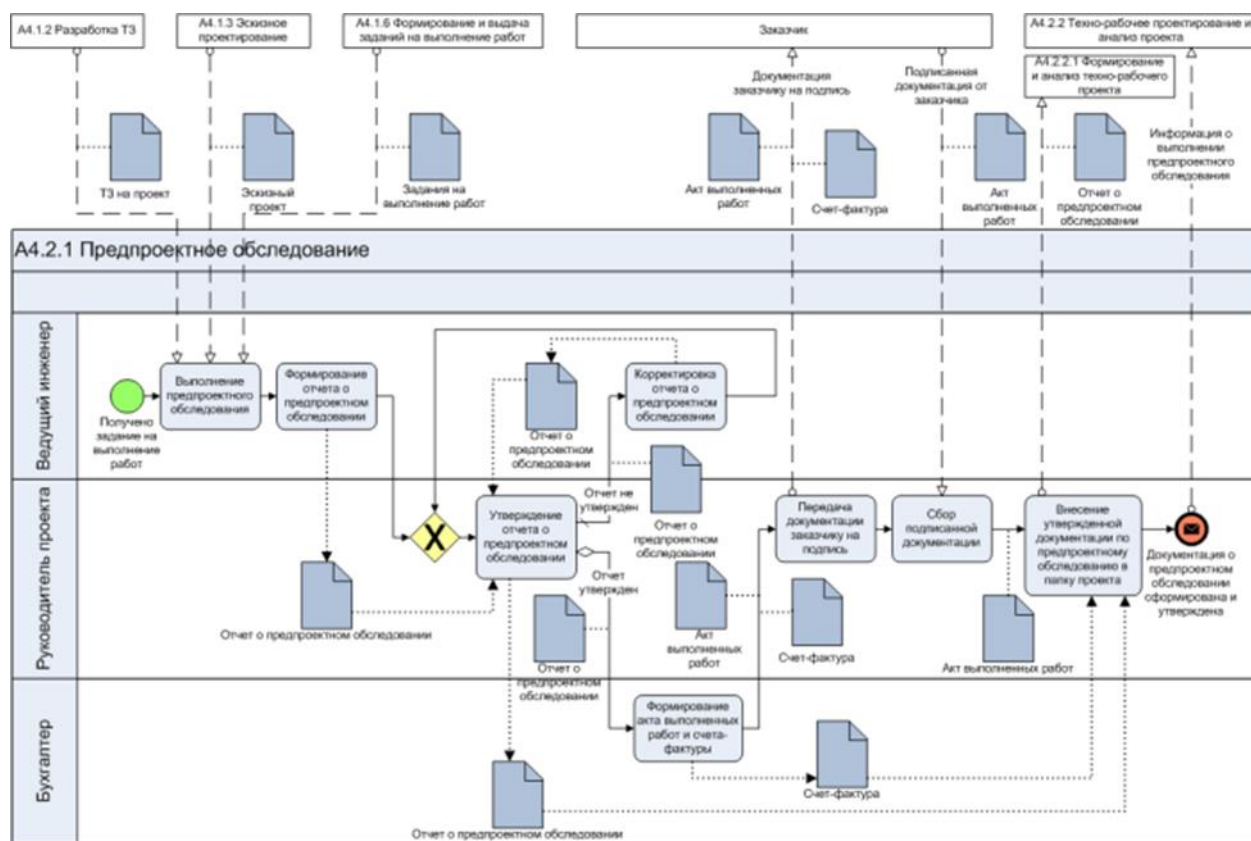
Бизнес-процесс может быть разбит (декомпозирован) на составляющие подпроцессы. IDEF0 используется для описания модели бизнес-процессов на верхнем уровне



2.4 BPMN

BPMN (Business Process Model and Notation) – модель бизнес-процессов и нотация. Используется для задач описания пошагового выполнения бизнес-процессов.

Можно сказать, что BPMN взяла часть идей от **Процедуры** (дорожки субъектов), часть от **EPC** (детальное описание алгоритма выполнения процесса) и углубила возможности детального описания бизнес-процесса.



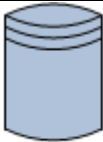
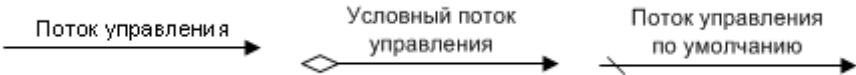
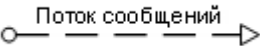


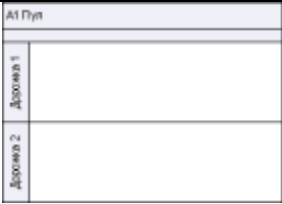
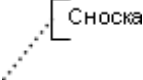
В итоге получился инструмент с широкими возможностями описания детального алгоритма выполнения процесса, который чаще всего применяют:

- Бизнес-аналитики для задач анализа бизнес-процессов,
- Бизнес-аналитики и технические специалисты для задач подготовки процесса к автоматизации.

Данная нотация обладает большим количеством графических элементов (более 60).

В BPMN выделяют 5 основных категорий элементов:

Название символа BPMN	Изображение
Элемент потока:	
○ Процессы	<div>Задача</div> <div>Подпроцесс</div>
○ События	<div>Стартовое событие</div> <div>Промежуточное событие</div> <div>Конечное событие</div>
○ Шлюзы (логические операторы)	<div>Параллельный шлюз</div> <div>Эксклюзивный шлюз</div> <div>Неэксклюзивный шлюз</div> <div>Комплексный шлюз</div> <div>Эксклюзивный шлюз по событиям</div>
Данные:	
○ Объекты	<div>Объект данных</div> <div>Набор объектов</div>

○ Базы данных	 База данных
Соединяющие элементы:	
○ Потоки управления	
○ Потоки сообщений	
○ Ассоциации	
Зоны ответственности:	
○ Пулы	
○ Дорожки	
Артефакты (сноски)	

2.5 Структура отчета

- Титульный лист
- Введение
- Краткое описание улучшаемой системы
- Контекстная модель IDEF0 для улучшаемой системы
- Модель 1 уровня IDEF0 (декомпозиция контекстной модели)
- По одной модели BPMN для каждого процесса модели 1 уровня
- Заключение

3 Лабораторная работа № 3 «Описать требования к ИС»

Введение

Во время лабораторного практикума должна быть разработана программная система (в дальнейшем **проект**) средней сложности.

Тема проекта согласуется с преподавателем (в дальнейшем это руководитель проекта) на первом занятии, в течение первой работы в соответствии с темой проекта команда студентов разрабатывает техническое задание (ТЗ) по форме, приведенной в приложении А. Разработку ТЗ можно считать началом первой фазы ЖЦ будущей ПС.

Техническое задание в дальнейшем должно стать основным документом, по которому студенты ведут разработку проекта. Любые изменения ТЗ на систему должны быть согласованы с преподавателем и заверены его подписью.

3.1 Требования к ТЗ

Техническое задание выполняется в соответствии с ГОСТ 34.602-89, в целом можно говорить, что оно состоит из трех частей:

- 1) содержание задания, в котором перечисляются все основные составные этапы выполнения проекта;
- 2) исходные данные к проекту;
- 3) календарный план выполнения работ.

3.1.1 Часть 1 – «Содержание задания».

Данная часть ТЗ фиксирована, в ней перечислены основные составные этапы выполнения проекта. При разработке ПС в рамках лабораторного практикума и в дальнейшем при выполнении курсового проекта будут использоваться две основные методологии: объектно-ориентированного анализа и проектирования (ООАП) и методология структурного проектирования.

ООАП (Object-Oriented Analysis/Design) - технология разработки программных систем, в основу которых положена объектно-ориентированная методология представления предметной области в виде объектов, являющихся экземплярами соответствующих классов [4]. Методология ООАП тесно связана с концепцией автоматизированной разработки программного обеспечения (Computer Aided Software Engineering, CASE) и языком моделирования UML (Unified Modeling Language).

Методология объектно-ориентированного анализа и проектирования используется при выполнении лабораторной работы №2 – описании и анализе предметной области, где

студенты должны выявить взаимосвязи между основными информационными объектами ПС, определить их характеристики и порядок взаимодействия.

Методология структурного проектирования применяется на первой фазе проектирования (лабораторная работа №3) при разделении системы на подсистемы, здесь используется принцип проектирования «сверху вниз». За каждым студентом закрепляется определенный перечень работ (обычно это разработка отдельной подсистемы), выполнение которых в дальнейшем контролирует преподаватель.

3.1.2 Часть 2 – «Исходные данные к проекту»

Часть включает в себя следующие подразделы:

1. Характеристики объекта автоматизации (или управления);
2. Требования к информационному обеспечению.
3. Требования к техническому обеспечению.
4. Требования к программному обеспечению.
5. Общие требования к проектируемой системе.
6. Перечень дополнительных работ (если необходимо).

Характеристики объекта автоматизации. Здесь указываются общие характеристики объекта автоматизации, характерные для рассматриваемой предметной области:

- а) полное название объекта (ов);
- б) условия его функционирования;
- в) *количественные и качественные показатели* объекта, которые являются ограничениями процесса функционирования.

В качестве примера рассмотрим проект «Автоматизированная система составления и разгадывания линейного кроссворда по выбранной теме», ТЗ на который приведено в приложении Б.

Понятие «объект автоматизации» в явном виде в ГОСТ 34.602-89 нигде не определено, но если внимательно прочитать п. 2.4.1. «В подразделе «Назначение системы» указывают вид автоматизируемой деятельности (управление, проектирование и т. п.) и перечень объектов автоматизации (объектов), на которых предполагается ее использовать»¹, то из него следует, что под «объектами автоматизации» авторы понимали вовсе не процессы. Будем считать, что объектом автоматизации может быть только материальный (интеллектуальный) объект - организация, магазин, цех, отдел, и так далее.

¹ ГОСТ 34.602-89, с. 3.

Комментарии к примеру. Из названия темы следует, что в данном случае объект автоматизации – это линейный кроссворд, а виды автоматизируемой деятельности – это *процессы*:

- 1) составления/ генерирования кроссворда;
- 2) разгадывания кроссворда;
- 3) работы со словарем понятий.

Обращаю Ваше внимание на то, чем составление кроссворда отличается от генерирования: в первом случае пользователь вручную составляет кроссворд (добавляет понятия, удаляет понятия и т.п.), во втором (генерирование) - кроссворд составляется системой автоматически в соответствии с теми настройками, которые выполнил пользователь. Процесс составления во многом дублирует функции, которые будет выполнять пользователь при редактировании кроссворда, поэтому процесс редактирования кроссворда не нужно выделять отдельно.

Для каждого процесса в ТЗ должны быть указаны количественные показатели-ограничения, для того, чтобы их правильно выбрать, необходимо знать начальные сведения о структуре самого объекта, каким образом будет проходить его построение и разгадывание. Отправной точкой является определение линейного кроссворда (ЛК). Итак, ЛК – это «цепочка слов, которая строится *методом стыкования*, где последняя буква первого слова является первой буквой второго и т. д. В чайнворде, как и в кроссворде, используются только имена существительные в именительном падеже и единственном числе». Так как ЛК строится из слов, то необходимо указать ограничение на длину слова: минимальную длину слова можно определить равную 3, а максимальную – 15, потому что чаще всего кроссворды будут составляться на бытовые темы и сам ЛК не должен быть очень простым. Чтобы ЛК было интересно разгадывать, он не должен быть очень коротким, поэтому минимальное количество слов, например, 5, а максимальное – 15. Идем дальше: «слова в чайнворде не пересекаются, а только *стыкуются* друг с другом. Иногда цепочку слов изгибают для придания сетке причудливой формы. Длинная изогнутая цепочка может неоднократно пересекать саму себя, как слова в кроссворде, такая головоломка обычно называется кроссчайнвордом». Исходя из этого, можно задать следующее ограничение – на форму отображения ЛК: обычная (линейная), спираль, змейка, W-образная. «В линейных кроссвордах слова могут перекрываться не только одной, но и двумя или тремя буквами, поэтому их длина указывается в скобках при определении к слову», эта часть описания ЛК дает еще одно ограничение: количество букв в пересечении - от 1 до 3. В качестве пожелания, заказчик отметил, что ЛК необходимо строить в двух режимах: ручном и автоматическом (генерация кроссворда), из этого следует еще одно ограничение –

составление кроссворда осуществляется с привязкой к словарю понятий. Для того чтобы системы была более универсальной (необходимо обеспечить создание тематических кроссвордов или на общие области знаний), заказчик предложил загружать в систему внешние словари понятий и обеспечить ему возможность редактировать их содержимое. При разгадывании кроссворда могут возникнуть затруднения, поэтому (в соответствии с пожеланиями заказчика) в системе должна быть организована система подсказок, количество которых можно связать с количеством слов – не менее 1 и не более 10% от количества слов.

Требования к информационному обеспечению. Разработка информационного обеспечения (ИО) – наиболее важная часть проекта, она может оказать существенное влияние на весь процесс разработки, поэтому уже на стадии разработки ТЗ необходимо определить:

- 1) на основании каких документов разрабатывается методическое и информационное обеспечение системы (нормативные и другие документы);
- 2) перечень исходных данных:
 - а) какие массивы данных используются и в каких форматах;
 - б) на каких носителях эти данные будут поставляться в систему;
- 3) перечень выходных данных:
 - а) какие массивы данных будут являться результатом работы ПС;
 - б) какие документы будут представлены пользователю и в каком виде (указывается вид носителя) и с какой периодичностью;
 - в) какие требования по целостности данных и их защите должны быть выполнены в проектируемой системе.

Особо должны быть выделены файл-серверные и клиент-серверные части информационного обеспечения, если таковые имеются.

Комментарии к примеру. Для разработки данной системы никаких нормативных документов не требуется (стандартов, инструкций и т.п.), поэтому необходимо только сослаться на информацию, где определена структура и свойства ЛК, а также требования к его построению. Также необходимо определить требования по входным и выходным данным. Большинство параметров ЛК будут задаваться пользователем в режиме диалога: он обязательно должен подключить словарь понятий, из которого будут формироваться задания для кроссворда. В данном случае «словарь понятий» – это текстовый файл определенной структуры (каждая строка файла – это понятие и его расшифровка), который должен загружаться в систему из внешней памяти (с любого логического диска), с которым пользователь должен иметь возможность работать дополнительно. Обязательное условие

заказчика – возможность создания коллекции ЛК, поэтому в системе должна быть предусмотрена возможность сохранения наиболее интересных кроссвордов в файл. На данном этапе еще трудно определить структуру этого файла (она должна учитывать и возможность дальнейшего разгадывания кроссворда с помощью данной системы), поэтому можно написать, что «структура файла определяется в процессе проектирования». Обязательным условием составления любого типа кроссвордов является его целостность (для данного случая - это отсутствие пустых клеток в середине ЛК), поэтому это обязательно должно быть записано в требованиях.

Требования к техническому обеспечению. Здесь формулируются ограничения по составу технических средств автоматизации с указанием конкретных типов оборудования и ЭВМ или их составляющих, используемых в проекте, если они заранее известны. Иначе в этом разделе указывается, что состав комплекса технических средств системы определяется в процессе проектирования системы.

Требования к программному обеспечению. Здесь приводится перечень используемых системных и прикладных программных средств, включая операционную систему, систему программирования, систему управления базами данных (в случае необходимости) и другие инструментальные средства (например, среда проектирования) с точным наименованием версий, если они заранее известны. Иначе указывается, что состав программного обеспечения определяется в процессе проектирования системы. Дополнительно могут быть указаны требования по совместимости разрабатываемого программного обеспечения с существующими системами.

Общие требования к проектируемой системе. В данной части ТЗ отдельно выделяется подраздел «*Функции, реализуемые системой*». В нем приводится подробный перечень функций, которые должна выполнять проектируемая система (или подсистема) в процессе ее эксплуатации. Отдельно должны быть выделены функции ввода данных, их обработки, передачи, хранения, а также формирования отчетов с выдачей на экран или печатающие устройства, функции управления, работа со справочниками и различные сервисные (обслуживающие систему) функции. Формулировка функций должна быть однозначной и конкретной, так как именно она является основой приемки проекта руководителем и проверки на полноту и качество реализованной системы или подсистемы.

Комментарии к примеру. Функции, которые должна выполнять данная система, определяются в первую очередь видами автоматизируемой деятельности, которые были определены в п.2.1 ТЗ, часть из них уже была выявлена в ходе обсуждения ограничений на ЛК. Среди неявных функций, про которые не должны забывать разработчики, это:

- а) Визуализация процессов работы с кроссвордом;

б) Выдача сведений о системе (справочные данные о системе и о том, как с ней работать).

Нужно отметить, что многие перечисленные в ТЗ функции, не раскрываются подробно, их необходимо будут детализировать при разработке функциональной спецификации (лабораторная работа №5).

В других подразделах оговариваются специальные технические требования, предъявляемые к системе:

- по быстродействию (времени реакции на выполнение наиболее важных функций);
- по режиму работы (диалоговый/интерактивный, автоматический);
- по точности (в случае, если в системе производятся математические расчеты, требующие минимизации вычислительных погрешностей, или используются внешние информационные источники (датчики, измерители и т.п.));
- по достоверности;
- по условиям функционирования (диапазон температур, относительная влажность, давление, наличие в атмосфере пыли, вредных примесей и т.д.),

а также все другие количественные и качественные показатели, определяющие эффективность функционирования системы. Кроме того, в данном разделе указываются санитарные правила и нормы (СанПин 2.2.2./2.4.2198-07 [6]) и ГОСТы, требования которых необходимо учитывать при разработке такого класса систем, с учетом того, что системы разворачиваются на средствах вычислительной техники [7, 8].

3.1.3 Часть 3 – Календарный план выполнения работ.

Технология RAD, как уже говорилось выше, требует жесткого следования плану-графику работ, поэтому в ТЗ оговариваются ключевые задания, по которым преподаватель должен проводить обязательный контроль. Каждый из перечисленных этапов должен завершаться *полностью готовой документацией*, согласованной с заказчиком (руководителем). Невыполнение в срок какого-либо из этапов может привести либо к сдвигу «контрольных точек» по оставшимся этапам, либо к незавершению проекта в срок.

В заключение хотелось бы отметить, что процесс составления ТЗ на систему:

1. требует от разработчиков коллективных обсуждений и принятия ответственных решений;
2. позволяет выявить наиболее «узкие» места проекта и оценить возможные риски;
3. дает возможность команде разработчиков распределить между собой все виды выполняемых работ, сосредоточив в дальнейшем усилия на концептуальных аспектах проекта;

4. определить наиболее приоритетные функции, которые будут составлять каркас системы.

3.2 Структура отчета

- Титульный лист
- Введение
- Общие сведения
- Назначение и цели создания (развития) Системы
- Характеристика объекта автоматизации
- Требования к системе
- Состав и содержание работ по созданию (развитию) системы
- Порядок контроля и приемки Системы
- Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу
- Требования к документированию
- Источники разработки
- Заключение

4 Лабораторная работа № 4 «Разработать проект системы в UML»

4.1 Введение

Unified Modeling Language (UML) — унифицированный язык моделирования. Расшифруем: modeling подразумевает создание модели, описывающей объект. Unified (универсальный, единый) — подходит для широкого класса проектируемых программных систем, различных областей приложений, типов организаций, уровней компетентности, размеров проектов. UML описывает объект в едином заданном синтаксисе, поэтому где бы вы не нарисовали диаграмму, ее правила будут понятны для всех, кто знаком с этим графическим языком — даже в другой стране.

Одна из задач UML — служить средством коммуникации внутри команды и при общении с заказчиком. Давайте рассмотрим возможные варианты использования диаграмм.

- Проектирование. UML-диаграммы помогут при **моделировании архитектуры больших проектов**, в которой можно собрать как крупные, так и более мелкие детали и нарисовать каркас (схему) приложения. По нему впоследствии будет строиться код.
- Реверс-инжиниринг — создание UML-модели из существующего кода приложения, обратное построение. Может применяться, например, на проектах поддержки, где есть написанный код, но **документация** неполная или отсутствует.
- Из моделей можно извлекать текстовую информацию и генерировать относительно удобочитаемые тексты — документировать. Текст и графика будут дополнять друг друга.

4.2 Виды диаграмм UML

4.2.1 Нотация UML для описания логики проекта

Как и любой другой язык, UML имеет собственные правила оформления моделей и синтаксис. С помощью графической нотации UML можно визуализировать систему, объединить все компоненты в единую структуру, уточнять и улучшать модель в процессе работы. На общем уровне графическая нотация UML содержит 4 основных типа элементов:

- фигуры;
- линии;
- значки;
- надписи.

UML-нотация является де-факто отраслевым стандартом в области разработки программного обеспечения, ИТ-инфраструктуры и бизнес-систем.

В языке UML есть 12 типов диаграмм:

- типа диаграмм представляют **статическую структуру** приложения;
- типов представляют **поведенческие аспекты** системы;
- 3 представляют **физические аспекты** функционирования системы (диаграммы реализации).

Некоторые из видов диаграмм специфичны для определенной системы и приложения. Самыми доступными из них являются:

- Диаграмма прецедентов (Use-case diagram);
- Диаграмма классов (Class diagram);
- Диаграмма активностей (Activity diagram);
- Диаграмма последовательности (Sequence diagram);
- Диаграмма развёртывания (Deployment diagram);

- Диаграмма сотрудничества (Collaboration diagram);
- Диаграмма объектов (Object diagram);
- Диаграмма состояний (Statechart diagram).

4.2.2 Диаграмма прецедентов — Use-case diagram

Диаграмма прецедентов использует 2 основных элемента:

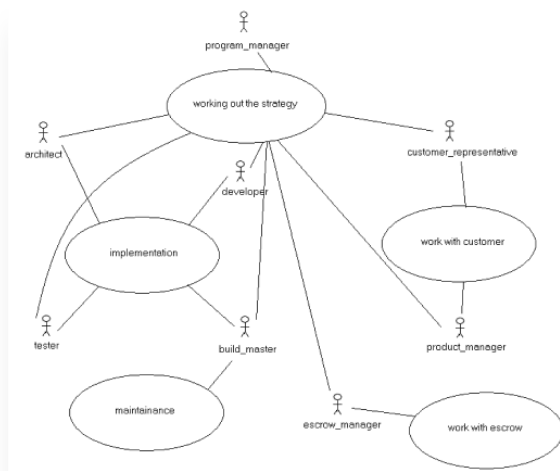
1) Actor (участник) — множество логически связанных ролей, исполняемых при взаимодействии с прецедентами или сущностями (система, подсистема или класс). Участником может быть человек, роль человека в системе или другая система, подсистема или класс, которые представляют нечто вне сущности.

2) Use case (прецедент) — описание отдельного аспекта поведения системы с точки зрения пользователя. Прецедент не показывает, "как" достигается некоторый результат, а только "что" именно выполняется.

Рассмотрим классический студенческий пример, в котором есть 2 участника: студент и библиотекарь. Прецеденты для студента: ищет в каталоге, заказывает, работает в читальном зале. Роль библиотекаря: выдача заказа, консультации (рекомендации книг по теме, обучение использованию поисковой системы и заполнению бланков заказа).

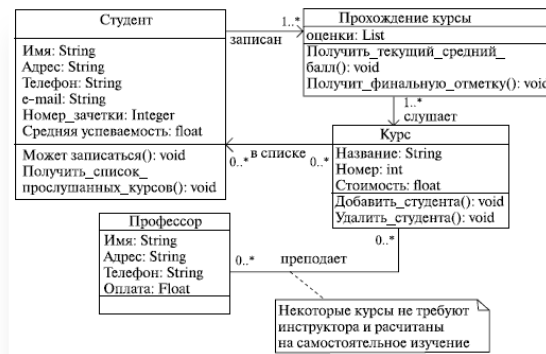


Второй пример немного сложнее. Видим, что одно и то же лицо может выступать в нескольких ролях. Например, product manager у нас работает над стратегией и больше ничем не занимается, архитектор работает над стратегией и занимается внедрением, build master занимается тремя вещами одновременно, и так далее. По такой схеме мы можем проследить, какая из ролей связана с какими прецедентами.



4.2.3 Диаграмма классов — Class diagram

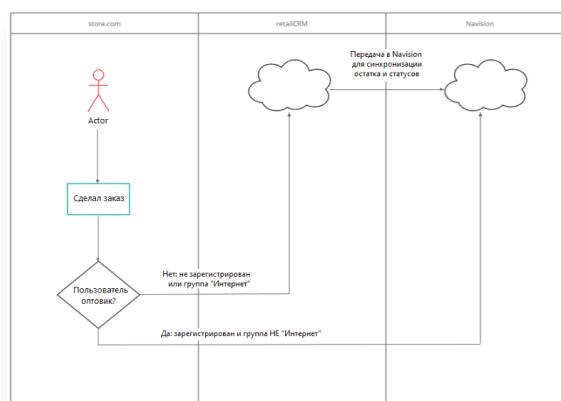
Класс (class) — категория вещей, которые имеют общие атрибуты и операции. Сама диаграмма классов является набором статических, декларативных элементов модели. Она дает нам наиболее полное и развернутое представление о связях в программном коде, функциональности и информации об отдельных классах. Приложения генерируются зачастую именно с диаграммы классов. Рассмотрим на простом примере ниже:



Для класса "студент" есть таблица, содержащая атрибуты: имя, адрес, телефон, e-mail, номер зачетки, средняя успеваемость. И также показаны связи данной сущности с другими: прохождением курса, какой курс слушает, кто профессор. В этом примере также добавляются функции, которые могут быть применены к сущности "студент".

4.2.4 Диаграмма активностей — Activity diagram

Диаграмма активностей описывает динамические аспекты поведения системы в виде блок-схемы, которая отражает бизнес-процессы, логику процедур и потоки работ — переходы от одной деятельности к другой. По сути, мы рисуем алгоритм действий (логику поведения) системы или взаимодействия нескольких систем. Ниже — пример подобной диаграммы для интернет-магазина.



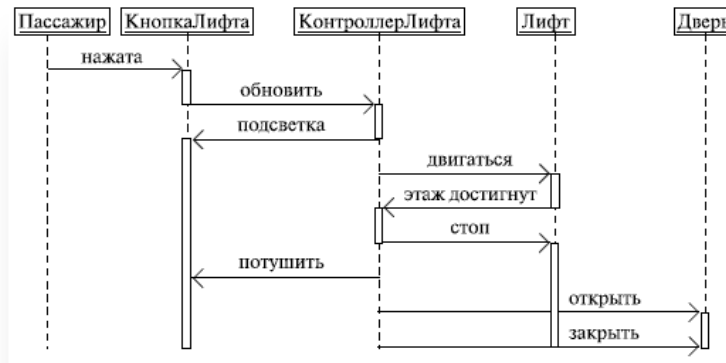
Evergreen

Диаграмма активностей для сайта магазина максимально доступно объясняет, какие есть интеграции в системе. Актер (в нашем случае — покупатель), зашедший на сайт, делает заказ. Далее у нас происходит разветвление: проверяем, является ли пользователь оптовиком (Да/Нет). Если он не зарегистрирован в системе и не оптовик, заказ отправляется в retailCRM. Если пользователь зарегистрирован, его заказ попадает в Navision. При этом между retailCRM и Navision происходит синхронизация остатка и статусов.

Эту базовую диаграмму мы можем дополнить, расширить, она может выступить частью документации и дает общее представление о работе системы.

4.2.5 Диаграмма последовательности — Sequence Diagram

Используется для уточнения диаграмм прецедентов — описывает поведенческие аспекты системы. Диаграмма последовательности отражает взаимодействие объектов в динамике, во времени. При этом информация принимает вид сообщений, а взаимодействие объектов подразумевает обмен этими сообщениями в рамках сценария.



4.2.6 Диаграмма развертывания — Deployment Diagram

Диаграмма развертывания отображает графическое представление инфраструктуры, на которую будет развернуто приложение: топологию системы и распределение компонентов по ее узлам, а также соединения — маршруты передачи данных между узлами. Диаграмма помогает более рационально организовать компоненты, от чего зависит в числе прочего и производительность системы, а также решить вспомогательные задачи, например, связанные с безопасностью.



4.3 Структура отчета

- Титульный лист
- Введение
- Краткое описание улучшаемой системы

- Функциональные требования к разрабатываемой системе в виде модели UML Use-Case
- По одной модели Activity diagram (State chart diagram) для 2-3 Use-Case
- По одной модели Sequence diagram для 2-3 Use-Case
- Диаграмма классов для разрабатываемой системы
- Модель компонентов для разрабатываемой системы
- Модель размещения для разрабатываемой системы
- Заключение

5 Лабораторная работа № 5 «Тестовые сценарии системы»

5.1 Введение

Тест-кейс — это проверка. "Выполни тест-кейс по вводу отрицательных значений" = проведи проверку такую-то и проверь, что результат будет такой-то.

См также: [Тест-кейс проверяет, а не доверяет!](#)

Устоявшегося русско-язычного определения нет, помните об этом. Главное — понимать суть.

Тест-кейс — это такое описание проверки работы системы, которое может выполнить **любой** человек из команды, будь то тестировщик, разработчик, аналитик или даже бизнес-заказчик.

Набор тест-кейсов называется тестовым набором (test suite).

Иногда этот набор некорректно называют тест-планом. Тест-план — это именно план: когда, что, зачем, какими ресурсами. (*тут будет ссылка на статью про тест-план*)

5.2 Описание тест-кейса

5.2.1 Стандартные атрибуты тест-кейса

1. **Номер** — уникальный идентификатор тест-кейса. Его удобно использовать для одинакового понимания, о какой проверке идет речь (например, дать ссылку в баге).
2. **Название** — краткое описание **сути** проверки. Должно помещаться в твиттер и быть понятным! Кратко, но емко.
3. **Предварительные шаги** — описание действий, которые необходимо выполнить, но прямого отношения к проверке они не имеют (например, зарегистрироваться в системе для проверки создания элемента). Если предварительных шагов нет, то секция не заполняется.
4. **Шаги** — описание действий, необходимых для проверки (например, создание элемента).
5. **Ожидаемый результат (ОР)** — сама проверка: что мы ожидаем получить после выполнения шагов ("*Элемент создан*").

См также:

[Правила написания предварительных шагов в тест-кейсах](#) — подробнее про пред шаги

[Вложил в тест-кейс аттач? Поясни его!](#) — и про вложения

5.2.2 Пример оформления (один ожидаемый результат)

Есть внутренний сайт компании, которая проводит интернет "Самый_лучший_в_своем_роде" — www.test.ru. Тестовый стенд, на котором проверяются доработки перед выкладкой в PROD (он же *production*, окружение для пользователей) находится по другому адресу — www.dev_test.ru.

Примечание: www.test.ru — абстрактное обозначение некоего сайта, не надо туда заходить и искать эту систему 😊. Приводится здесь, чтобы показать ошибки в написании тест-кейсов.

На сайте можно заводить карточки обслуживаемых зданий и карточки их жильцов. Карточки создает администратор, на тестовой машине всегда есть пользователь с правами админа, логин / пароль — admin / 1. При входе на тестовый сервер есть дополнительная авторизация, чтобы туда не могли попасть люди "извне", с логином и паролем test / test.

Тест-кейс № 1. Создание жильца без ФИО.

Шаги

1. Зайти на сайт www.dev_test.ru (логин - test, пароль - test).
2. Войти под учеткой администратора (логин - admin, пароль - 1)
3. Перейти на вкладку "Жильцы".
4. Нажать на кнопку "Создать карточку жильца".
5. Нажать на кнопку "Сохранить", не заполняя никакие данные.

Ожидаемый результат

Появляется сообщение об ошибке "Заполните обязательные поля, отмеченные *", карточка не сохраняется.

5.2.3 Преимущества и недостатки тест-кейсов

Преимущества: тест-кейсы можно доверить выполнять новичку или призванному на помощь коллеге из другого отдела, который ничегошеньки о проекте не знает. Дополнительных вопросов с его стороны будет по минимуму — все и так (должно быть 😊) понятно!

Недостатки (вытекают один из другого):

1. *Очень много копиасты много копиасты копиасты.* В примере выше заполняется поле "ФИО". Тест-кейсы "ввести в поле только символы, только числа, строку нулевой длины и т. д." будут очень похожи друг на друга, первые шаги одинаковые и, положив руку на сердце, будут копиаститься. Попробуйте написать хотя бы три тест-кейса на один функционал и сразу увидите эту проблему.
2. *Сложно поддерживать.* Представьте, что вкладку "Жильцы" переименовали в "Заказчики". Чтобы актуализировать тест-кейсы, надо внести изменения в сотни сценариев, что утомительно даже в режиме "*Ctrl + C, Ctrl + V*".
3. *Неактуальное состояние.* Тест-кейсы копиастятся друг от друга, и часто в них остаются неактуальные части из исходного кейса, которые забыли изменить.

Последний недостаток перечеркивает достоинства. Тестировщик, который уже год как работает на проекте, поймет и неактуальный кейс, тем более если выполняет их подряд, начиная с первого. А тестировщик, который ничего о проекте не знает и получил пару кейсов из середины тестового набора, не сможет понять, о чем в них идет речь.

Чтобы тест-кейсы честно выполняли свою роль, их надо поддерживать, периодически проверять на правильность и дорабатывать... Это отнимает очень много времени и сил.

Чтобы упростить этот процесс, могут быть использованы тест-кейсы с одним сценарием выполнения, но несколькими входными параметрами и разными ожидаемыми результатами. Фактически мы получаем мини чек-листы с предварительными шагами.

См также:

[Тест-кейсы – зло! Или все-таки нет?](#) — перевод статьи John Andrews о преимуществах тест-кейсов.

[Когда применять тест-кейсы](#) — а когда они не нужны

5.2.4 Примеры оформления (несколько ожидаемых результатов)

Рассматриваем все тот же абстрактный сайт www.test.ru. Допустим, что поле "ФИО" по ТЗ решили ограничить 40 символами (*тут будет ссылка почему так не надо делать*).

Когда говорят о нескольких ожидаемых результатах, это может означать:

- Даны несколько вариантов вводимых данных и для каждого прописан свой ожидаемый результат.

- Несколько шагов, а не только последний, содержат ожидаемый результат.
- После одного сценария выполняются сразу несколько проверок.

5.2.5 Несколько вариантов вводимых данных

Тест-кейс № 2. Создание жилья, проверка поля "ФИО".

Шаги:

1. Зайти на сайт www.dev_test.ru (логин - test, пароль - test).
2. Войти под учеткой администратора (логин - admin, , пароль - 1)
3. Перейти на вкладку "Жильцы"
4. Нажать на кнопку "Создать карточку жилья".
5. Заполнить поле ФИО (см "Ожидаемый результат")
6. Нажать на кнопку "Сохранить".

Ожидаемый результат

Вводимое значение	Ожидаемый результат
Киселева Ольга Евгеньевна	Ок, карточка сохраняется
<Оставить поле пустым>	Ошибка – «Заполните обязательные поля, отмеченные *», карточка не сохраняется
2*4*6*8*11*14*17*20*23*26*29*32*35*38*41*	Ошибка – «Максимальная длина поля – 40 символов, введено - 41», карточка не сохраняется. (Такую строку легко сформировать с помощью инструмента perlclip)
&*%#(^\$@*&	Ошибка – «Поле ФИО может содержать только буквы русского алфавита» (см. статью <i>про идиотов и ограничения</i>), карточка не сохраняется
Kiseleva Olga Evgenievna	Ошибка – «Поле ФИО может содержать только буквы русского алфавита» (см. статью <i>про идиотов и ограничения</i>), карточка не сохраняется
...	...

Для этого варианта тест-кейса запись в виде таблички: данные – результат — наше всё!

5.2.6 Результаты для нескольких шагов из кейса

Другой вариант записи тест-кейса с несколькими ожидаемыми результатами — когда результаты пишутся на разные пункты шагов выполнения проверки, то есть на разные этапы сценария.

Тест-кейс № 3. Создание жилья с ~~жудым~~ полным ФИО.

Шаги:

1. Зайти на сайт www.dev_test.ru (логин - test, пароль - test).
2. Войти под учеткой администратора (логин - admin, пароль - 1)
3. Перейти на вкладку "Жильцы"
4. Нажать на кнопку "Создать карточку жилья".
5. Ввести корректные ФИО, например, "Иванов Иван Иванович".
6. Нажать на кнопку "Сохранить".

Ожидаемый результат

1. Открывается окно ввода логина / пароля с соответствующими полями для ввода, кнопкой "Войти" и сообщением "Для входа в систему введите, пожалуйста, свои данные".
2. Вход в систему успешно осуществлен. В правом верхнем углу отображается надпись "Здравствуйте, admin". Открыта главная страница сайта.
4. Открылась страница "Создание нового жилья" с полями "Фамилия", "Имя" и "Отчество" и кнопкой "Сохранить".
6. Окно с информацией о жилье закрывается и отображается общий список, в котором присутствует новая карточка. Эту карточку можно открыть и на ней отображаются введенные данные, то есть в поле ФИО указано "Иванов Иван Иванович".

Ну что, приятно читать такой тест-кейс?)) Думаю, что в таком виде не очень. Обычно же читаешь и сразу выполняешь. Выполнил пункты 1,2,3... 10... А потом БАЦ, и в ожидаемом результате читаешь результат для пункта 1! Но подождите, я уже на пункте 6!

Это надо снова повторять тест-кейс, но теперь уже сравнивая результат. И глаза такие прыг-скок туда-сюда. Прочитали пункт 1 — прыг в результаты. Проверили — прыг обратно к шагам, ищем пункт 2. Выполнили — прыг глазами в результаты. Ищем там пункт 2, проверяем... Прыг снова в шаги, ищем пункт 3... Ну вы поняли.

Поэтому если уж очень хочется писать ОР на каждый шаг, сделайте это, пожалуйста, в виде таблички:

Тест-кейс № 3. Создание жилья с полным ФИО

Шаг	Ожидаемый результат
1. Зайти на сайт www.dev_test.ru (логин - test, пароль - test).	Открывается окно ввода логина / пароля с соответствующими полями для ввода, кнопкой "Войти" и сообщением "Для входа в систему введите, пожалуйста, свои данные".
2. Войти под учеткой администратора (логин - admin, пароль - 1)	Вход в систему успешно осуществлен. В правом верхнем углу отображается надпись "Здравствуйте, admin". Открыта главная страница сайта.
3. Перейти на вкладку "Жильцы"	Открылась страница со всеми жильцами, отсортированными в порядке поднимки (самые актуальные сверху)
4. Нажать на кнопку "Создать карточку жилья"	Открылась страница "Создание нового жилья" с полями: "Фамилия", "Имя" и "Отчество" и кнопкой "Сохранить".
5. Ввести корректные ФИО, например: "Иванов Иван Иванович".	Данные вводятся
6. Нажать на кнопку "Сохранить".	Окно с информацией о жилье закрывается и отображается общий список, в котором присутствует новая карточка. Эту карточку можно открыть и на ней отображаются введенные данные, то есть в поле ФИО указано "Иванов Иван Иванович".

5.2.7 Несколько проверок после одного сценария

Тест-кейс № 4. Создание жилья с самым полным ФИО.

Шаги:

1. Зайти на сайт www.dev_test.ru (логин - test, пароль - test).
2. Войти под учеткой администратора (логин - admin, , пароль - 1)

3. Перейти на вкладку "Жильцы"
4. Нажать на кнопку "Создать карточку жильца".
5. Ввести корректные ФИО, например, "Иванов Иван Иванович".
6. Нажать на кнопку "Сохранить".

Ожидаемый результат

1. Окно с информацией о жильце закрывается и отображается общий список, в котором присутствует новая карточка.
2. Эту карточку можно открыть.
3. В открытой карточке отображаются введенные данные, то есть в поле ФИО указано "Иванов Иван Иванович".

5.3 Требования к отчету

- Титульный лист
- Введение
- Варианты использования (Use-Case) улучшаемой системы
- По одной тест-кейсу для 3-4 Use-Case
- Заключение

6 Лабораторная работа № 6 «Подготовить материалы к SCRUM-проекту»

6.1 Введение

Agile – гибкий подход к управлению проектами помогает командам эффективно реагировать на изменения, быстрее и с меньшими затратами разрабатывать уникальные, соответствующие требованиям заказчиков продукты и услуги, которые дают конкурентное преимущество в условиях постоянно меняющегося рынка.

Agile может реализовываться разными способами. Один из них – это фреймворк SCRUM.

6.2 Теоретическое введение в SCRUM

6.2.1 Суть SCRUM

В SCRUM работа над проектом ведется короткими циклами (спринтами), в ходе которых постоянно наращивается функционал продукта/услуги (создаются готовые результаты — инкременты).

Каждый готовый результат демонстрируется заказчику, пользователям и другим заинтересованным сторонам для получения обратной связи и корректировки требований.

6.2.2 Роли в SCRUM

Владелец продукта (Product owner) отвечает за то, чтобы делать правильные вещи (бизнес-ценность)

- Представляет интересы заказчика и других заинтересованных сторон в проекте
- Помогает команде понять требования к продукту (включая критерии приемки) и определить их приоритеты на основе бизнес-ценности
- Участвует в приемке готовых результатов (инкрементов).

Команда разработки (Development team) отвечает за то, чтобы делать вещи правильно (качество разработки)

- Планирует работу в спринте (распределяет задачи между членами команды)
- В ходе спринтов создает готовый продукт (инкремент)
- Предоставляет готовые результаты для ревью заинтересованным сторонам
- Анализирует свою эффективность и вырабатывает меры по ее повышению.

SCRUM мастер (Scrum master) отвечает за то, чтобы делать их быстро (эффективность команды)

- Ведет и является фасилитатором Scrum мероприятий
- Помогает команде анализировать свою эффективность и вырабатывать меры по ее повышению
- Помогает выстраивать взаимодействия между командой, владельцем продукта и внешними заинтересованными сторонами
- Помогает соблюдать правила Scrum.

Факторы успеха Agile-команды

Agile команда

- ▶ Кросс-функциональная
- ▶ Работает в одном месте
- ▶ Стабильна в проекте
- ▶ С выделенными ресурсами
- ▶ Отвечает за процесс и качество
- ▶ Автономность



Поведение

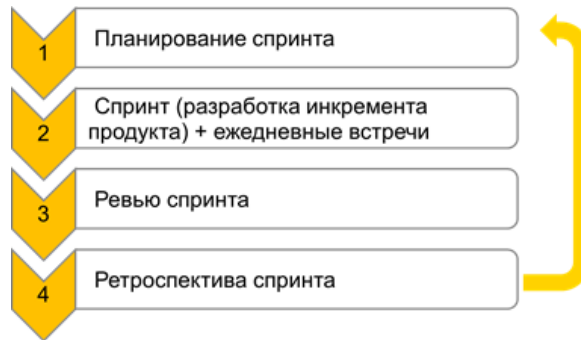
- ▶ Адаптивность
- ▶ Гибкость
- ▶ Кооперация
- ▶ Независимость
- ▶ Любопытство
- ▶ Взятие риска

6.2.3 Требования к продукту

Беклог продукта (Product backlog) — это приоритизированный список требований, зафиксированных в виде коротких описаний (пользовательских историй).

Он является единственным источником требований для разработки продукта и любых изменений, которые могут быть внесены в него.

6.2.4 Этапы SCRUM



6.2.4.1 Этап — Планирование спринта (Sprint Planning)

Цель этапа – определение и уточнение требований (пользовательских историй), включенных в текущий спринт, распределение и планирование работы по созданию инкремента продукта.

Участники – команда разработки, владелец продукта, SCRUM мастер.

Периодичность – в начале каждого нового спринта.

6.2.4.2 Этап — Спринт (Sprint)

Цель этапа – выполнение работы по реализации требований (пользовательских историй), создание готового продукта (инкремента).

Участники – команда разработки, SCRUM мастер и владелец продукта (для уточнения пользовательских историй и критериев приемки).

Ежедневные встречи (Daily Scrum)

Цель встречи – синхронизация действий команды разработки и создания плана работы на ближайшие сутки.

Участники – члены команды разработки, при необходимости — SCRUM мастер.

Периодичность – ежедневно, в начале дня (15 мин).

6.2.4.3 Этап — Ревью спринта (Sprint review)

Цель этапа – демонстрация созданного в ходе спринта готового продукта (инкремента) и получение обратной связи.

Участники – команда, владелец продукта и ключевые заинтересованные стороны, приглашенные владельцем продукта.

Периодичность – в конце каждого спринта.

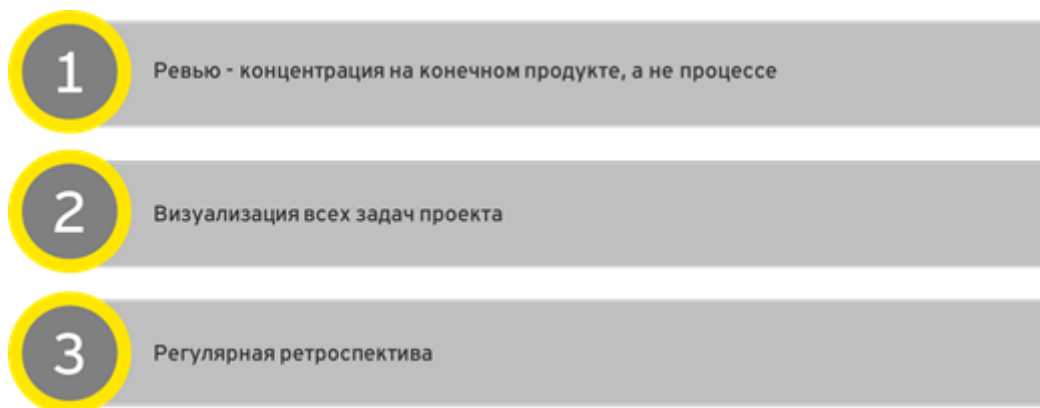
6.2.4.4 Этап — Ретроспектива спринта (Sprint retrospective)

Цель этапа – совершенствование процесса разработки и определение тех улучшений, которые будут реализованы командой в следующем спринте.

Участники — команда разработки, скрам мастер, владелец продукта (если необходимо).

Периодичность встречи — в конце каждого спринта после ревью.

6.2.5 3 ключевых «оружия» на проекте



6.2.6 На чем держать фокус?



6.3 Разработка бэклога. Пользовательские истории

6.3.1 Что такое User Stories?

Работа с аудиторией — одна из важнейших составляющих создания продуктов, потому что именно пользователи будут определять его успех. Следовательно, необходимо понимать, как пользователи будут взаимодействовать с вашим [приложением](#) или сайтом. Чтобы лучше в этом разобраться, существует User Story.

Пользовательская история, или User Story — это описание функций продукта простым языком, составленное с точки зрения пользователя. Она помогает понять, какую пользу клиенту принесет функционал приложения, еще на [этапе аналитики проекта](#). Пользовательские истории служат неким контекстом для разработчиков: те понимают, чего хочет от продукта конечный пользователь, и работают более целенаправленно. Эти истории состоят из нескольких предложений и не углубляются в детали: они отражают суть и фокусируются на главном.

6.3.2 Для чего применяется User Story?

Пользовательские истории помогают сосредоточиться на потребностях пользователя: как он будет использовать приложение? Чего он ждет от продукта? Как поведет себя в той или иной ситуации? Таким образом, ответы на эти вопросы помогут

создателям продукта решать настоящие проблемы клиентов. Вот несколько главных задач, для которых необходимо использовать User Stories:

- организовать работу. Когда проект разбит на части, связанные с пользовательскими историями, каждая из них представляет собой цельную и понятную задачу. Так разработчики могут фокусироваться на каждой из них и получать измеримый результат. Так, например, мы описывали пользовательские требования с помощью дискретных User Stories для разработки платформы для онлайн-обучения. Это помогло клиенту в условиях ограниченного бюджета гибко управлять приоритетами в разработке, добавляя или исключая User Story.
- сохранить фокус на пользователе. Конечно, разработка включает себя десятки сложных задач, связанных с техническими, финансовыми и другими вопросами. Однако юзер стори — это постоянное напоминание команде о тех, для кого этот продукт создается, и направляют их работу в нужное русло. Кто такой юзер вашего приложения и как вы можете быть ему полезны? Ответы на эти вопросы помогут вам составить качественную user story.
- сплотить команду. Несмотря на то, что у каждого есть свои задачи (дизайн, тестирование, разработка), каждый понимает конечную цель и видит себя частью целого. Только работая сообща, можно достичь нужного пользователю результата, и пользовательские истории дают четкое понимание этого аспекта работы.
- найти свежие решения. Команда старается придумать самый приятный и интересный способ решить задачу пользователя. Часто это приводит к появлению новых интересных идей и их воплощению. Результат — полезный и уникальный продукт.

6.3.3 Структура User Story

Ваша пользовательская история будет уникальной, поэтому вы можете создать свой особенный способ ее изложения. Однако есть стандартные элементы создания пользовательской истории, которые помогут вам лучше всего “прочитать мысли” пользователя и понять его способ мышления. Эти элементы включают в себя:

Заголовок	—	Краткое описание истории
ФОРМУЛИРОВКА ИСТОРИИ:		
Я как	—	Роль
Хочу	—	Функционал
Для того чтобы	—	Выгода

Когда вы разобрались с основными моделями поведения конечных пользователей, необходимо более подробно описать их действия. Как они будут заказывать еду в вашем приложении? Что будут искать на сайте университета? По каким критериям будут искать врача? Отталкивайтесь от того, что у вас есть, и постарайтесь как можно точнее представить поведение пользователей. Примерно так выглядит схема сценария использования вашего продукта:

*сценарий: Заголовок
дано [контекст]
и [ещё немного контекста] ...
когда [событие]*

тогда [результат]
и [ещё один результат]...

С помощью User Stories вы сможете приступить к созданию продукта более обдуманно. Формулировка функциональных требований станет проще, вы уже будете видеть конечный результат, и достичь его с этим пониманием будет проще.

6.3.4 User Story: примеры

Представим, что нужно разработать телемедицинское приложение, в котором врачи смогут проводить онлайн-консультации и отслеживать показатели пациента. Давайте для начала сформулируем несколько основных примеров User Stories для пациента, который пользуется этим приложением:

- **я как** пациент **хочу** созваниваться с врачами по видеосвязи, **чтобы** иметь возможность обсудить вопросы здоровья;
- **я как** пациент **хочу** иметь собственную медкарту, **чтобы** хранить всю информацию о своем здоровье в одном месте;
- **я как** пациент **хочу** оплачивать консультации в приложении, **чтобы** не искать для этого сторонние сервисы и обезопасить свои средства.

Вторая, не менее важная часть приложения — для врачей. У них есть свои потребности, поэтому и функционал будет отличаться. Вот примеры юзер стори для врачей:

- **я как** врач **хочу** созваниваться с пациентами по видеосвязи, **чтобы** иметь возможность работать из дома;
- **я как** врач **хочу** управлять своим расписанием, **чтобы** равномерно распределять нагрузку;
- **я как** врач **хочу** получать от пациентов медиафайлы, **чтобы** изучать результаты исследований и более качественно лечить людей.

Для написания User Stories необходимо понимать потребности клиентов. При этом количество и содержание ваших пользовательских историй зависит от разных факторов: целевая аудитория, функционал приложения и даже бюджет на разработку. Не бойтесь экспериментировать и придумывать разнообразные юзер стори. Даже неудачные варианты помогут вам сузить фокус и сосредоточиться на главных потребностях пользователя.

6.4 Требования к отчету

- Титульный лист
- Введение
- Варианты использования (Use-Case) улучшаемой системы
- Бэклог продукта для всех вариантов использования системы
- Бэклог первого спринта разработки системы
- Заключение

7 Лабораторная работа № 7 «План внедрения системы»

Введение

Цель работы — научиться выполнять планирование программного: продукта, выполнять анализ осуществимости задачи, распределять обязанности в команде разработчиков, составлять графики работ.

7.1 Общие сведения о разработке программного обеспечения

Существует разница между профессиональной разработкой ПО и любительским программированием. Процесс создания профессионального ПО является субъектом бюджетной политики организации. Возникает необходимость управления проектом разработки программного обеспечения.

Руководители проектов призваны спланировать все этапы разработки программного продукта. Они также должны контролировать ход выполнения работ и соблюдения всех требуемых стандартов. Постоянный контроль за ходом выполнения работ необходим для того, чтобы процесс разработки не выходил за временные и бюджетные ограничения. Хорошее управление не гарантирует успешного завершения проекта, но плохое управление обязательно приведет к его провалу. Это может выразиться в задержке сроков сдачи готового ПО, в превышении сметной стоимости проекта и в несоответствии готового ПО спецификации требований.

Руководитель программного проекта не видит процесс "роста" разрабатываемого ПО. Он может полагаться только на документацию, которая фиксирует процесс разработки программного продукта.

7.2 Процесс управления разработкой программного обеспечения

Работы, выполняемые в проекте по созданию ПО существенно зависят от организации, где выполняется разработка ПО, и от типа создаваемого программного продукта. Но всегда можно выделить следующие: Написание предложений по созданию ПО; Планирование и составление графика работ по созданию ПО; Оценивание стоимости проекта; Подбор персонала; Контроль за ходом выполнения работ; Написание отчетов и представлений.

Первая стадия программного проекта может состоять из написания предложений по реализации этого проекта. Предложения должны содержать описание целей проектов и способов их достижения. Они также обычно включают в себя оценки финансовых и временных затрат на выполнение проекта. При необходимости здесь могут приводиться обоснования для передачи проекта на выполнение сторонней организации или команде разработчиков.

На этапе планирования проекта определяются процессы, этапы и полученные на каждом из них результаты, которые должны привести к выполнению проекта. Реализация этого плана приведет к достижению целей проекта. Определение стоимости проекта напрямую связано с его планированием, поскольку здесь оцениваются ресурсы, требующиеся для выполнения плана.

Контроль за ходом выполнения работ (мониторинг проекта) — это непрерывный процесс, продолжающийся в течение всего срока реализации проекта. Руководитель должен постоянно отслеживать ход реализации проекта и сравнивать фактические и плановые показатели выполнения работ с их стоимостью. В течение реализации проекта обычно происходит несколько формальных контрольных проверок хода выполнения работ по созданию ПО. Такие проверки должны дать общую картину хода реализации проекта в целом и показать, насколько уже разработанная часть ПО соответствует целям проекта.

Руководители проектов обычно обязаны сами подбирать исполнителей для своих проектов. В идеальном случае профессиональный уровень исполнителей должен соответствовать той работе, которую они будут выполнять в ходе реализации проекта. Однако во многих случаях руководители должны полагаться на команду разработчиков, которая далека от идеальной. Такая ситуация может быть вызвана следующими причинами:

- Бюджет проекта не позволяет привлечь высококвалифицированный персонал. В таком случае за меньшую плату привлекаются менее квалифицированные специалисты.
- Бывают ситуации, когда невозможно найти специалистов необходимой квалификации, как в самой организации-разработчике, так и вне ее. Например, в организации "лучшие люди" могут быть уже заняты в других проектах.
- Организация хочет повысить профессиональный уровень своих работников. В этом случае она может привлечь к участию в проекте неопытных или недостаточно квалифицированных работников, чтобы они приобрели необходимый опыт и поучились у более опытных специалистов.

Таким образом, почти всегда подбор специалистов для выполнения проекта имеет определенные ограничения и не является свободным. Вместе с тем необходимо, чтобы хотя бы несколько членов группы разработчиков имели квалификацию и опыт, достаточные для работы над данным проектом. В противном случае невозможно избежать ошибок в разработке ПО.

Руководитель проекта обычно обязан посылать отчеты о ходе его выполнения как заказчику, так и подрядным организациям. Это должны быть краткие документы, основанные на информации, извлекаемой из подробных отчетов о проекте. В этих отчетах должна быть та информация, которая позволяет четко оценить степень готовности создаваемого программного продукта.

В группе по разработке ПО выделены следующие роли: руководитель – общее руководство проектом, написание документации, общение с заказчиком ПО; системный аналитик – разработка требований (составление технического задания, проекта программного обеспечения); тестер – составление плана тестирования и аттестации готового ПО (продукта), составление сценария тестирования, базовый пример, проведение мероприятий по плану тестирования; разработчик – моделирование компонент программного обеспечения, кодирование.

7.2.1 Планирование проекта разработки программного обеспечения

План, разработанный на начальном этапе проекта, рассматривается всеми его участниками как руководящий документ, выполнение которого должно привести к успешному завершению проекта. Этот первоначальный план должен максимально подробно описывать все этапы реализации проекта.

Процесс планирования начинается, исходя из описания системы, с определения проектных ограничений (временные ограничения, возможности наличного персонала, бюджетные ограничения и т.д.). Эти ограничения должны определяться параллельно с оцениванием проектных параметров, таких как структура и размер проекта, а также распределением функций среди исполнителей. Затем определяются этапы разработки и то, какие результаты документация, прототипы, подсистемы или версии программного продукта) должны быть получены по окончании этих этапов. Далее начинается циклическая часть планирования. Сначала разрабатывается график работ по выполнению проекта или дается разрешение на продолжение использования ранее созданного графика. После этого проводится контроль выполнения работ и отмечаются расхождения между реальным и плановым ходом работ.

План проекта должен четко показать ресурсы, необходимые для реализации проекта, разделение работ на этапы и временной график выполнения этих этапов. В некоторых

организациях план проекта составляется как единый документ, содержащий все виды планов, описанных выше. В других случаях план проекта описывает только технологический процесс создания ПО. В таком плане обязательно присутствуют ссылки на планы других видов, но они разрабатываются отдельно от плана проекта.

Детализация планов проектов очень разнится в зависимости от типа разрабатываемого программного продукта и организации-разработчика. Но в любом случае большинство планов содержат следующие разделы:

1. Введение. Краткое описание целей проекта и проектных ограничений (бюджетных, временных и т.д.), которые важны для управления проектом.
2. Организация выполнения проекта. Описание способа подбора команды разработчиков и распределение обязанностей между членами команды.
3. Анализ рисков. Описание возможных проектных рисков, вероятности их проявления и стратегий, направленных на их уменьшение.
4. Аппаратные и программные ресурсы, необходимые для реализации проекта. Перечень аппаратных средств и программного обеспечения, необходимого для разработки программного продукта. Если аппаратные средства требуется закупать, приводится их стоимость совместно с графиком закупки и поставки.
5. Разбиение работ на этапы. Процесс реализации проекта разбивается на отдельные процессы, определяются этапы выполнения проекта, приводится описание результатов ("выходов") каждого этапа и контрольные отметки.
6. График работ. В этом графике отображаются зависимости между отдельными процессами (этапами) разработки ПО, оценки времени их выполнения и распределение членов команды разработчиков по отдельным этапам.
7. Механизмы мониторинга и контроля за ходом выполнения проекта. Описываются предоставляемые руководителем отчеты о ходе выполнения работ, сроки их предоставления, а также механизмы мониторинга всего проекта.

План должен регулярно пересматриваться в процессе реализации проекта. Одни части плана, например, график работ, изменяются часто, другие более стабильны. Для внесения изменений в план требуется специальная организация документопотока, позволяющая отслеживать эти изменения.

7.2.2 Общие сведения о требованиях к информационным системам

Проблемы, которые приходится решать специалистам в процессе создания программного обеспечения, очень сложны. Описание функциональных возможностей и ограничений, накладываемых на систему, называется требованиями к этой системе, а сам процесс формирования, анализа, документирования и проверки этих функциональных возможностей и ограничений – разработкой требований.

Требования подразделяются на пользовательские и системные. Пользовательские требования – это описание на естественном языке (плюс поясняющие диаграммы) функций, выполняемых системой, и ограничений, накладываемых на неё. Системные требования – это описание особенностей системы (архитектура системы, требования к параметрам оборудования и т.д.), необходимых для эффективной реализации требований пользователя.

Разработка требований — это процесс, включающий мероприятия, необходимые для создания и утверждения документа, содержащего спецификацию системных требований. Процесс разработки требований должен начинаться с анализа осуществимости. Началом такого анализа является общее описание системы и ее назначения, а результатом анализа — отчет, в котором должна быть четкая рекомендация, продолжать или нет процесс разработки требований проектируемой системы. Другими словами, анализ осуществимости должен осветить следующие вопросы:

- Отвечает ли система общим и бизнес-целям организации-заказчика и организации-разработчика?

- Можно ли реализовать систему, используя существующие на данный момент технологии и не выходя за пределы заданной стоимости?
- Можно ли объединить систему с другими системами, которые уже эксплуатируются?

Критическим является вопрос, будет ли система соответствовать целям организации. Если система не соответствует этим целям, она не представляет никакой ценности для организации.

Выполнение анализа осуществимости включает сбор и анализ информации о будущей системе и написание соответствующего отчета. Сначала следует определить, какая именно информация необходима, чтобы ответить на поставленные выше вопросы. Например, эту информацию можно получить, ответив на следующее:

- Что произойдет с организацией, если система не будет введена в эксплуатацию?
- Какие текущие проблемы существуют в организации и как новая система поможет их решить?
- Каким образом система будет способствовать целям бизнеса?

Требует ли разработка системы технологии, которая до этого не использовалась в организации?

Далее необходимо определить источники информации. Это могут быть менеджеры отделов, где система будет использоваться, разработчики программного обеспечения, знакомые с типом будущей системы, технологи, конечные пользователи и т.д.

После обработки собранной информации готовится отчет по анализу осуществимости создания системы. В нем должны быть даны рекомендации относительно продолжения разработки системы. Могут быть предложены изменения бюджета и графика работ по созданию системы или предъявлены более высокие требования к системе.

Эффективное управление программным проектом напрямую зависит от правильного планирования работ, необходимых для его выполнения. План помогает менеджеру предвидеть проблемы, которые могут возникнуть на каких-либо этапах создания ПО, и разработать превентивные меры для их предупреждения или решения. План, разработанный на начальном этапе проекта, рассматривается всеми его участниками как руководящий документ, выполнение которого должно привести к успешному завершению проекта. Этот первоначальный план должен максимально подробно описывать все этапы реализации проекта.

Кроме разработки плана проекта, на менеджера ложится обязанность разработки других видов планов. Эти виды планов кратко описаны в таблице 7.1.

Таблица 7.1 – Виды планов

План	Описание
План качества	Описывает стандарты и мероприятия по поддержке качества разрабатываемого ПО
План аттестации	Описывает способы, ресурсы и перечень работ, необходимых для аттестации программной системы
План управления конфигурацией	Описывает структуру и процессы управления конфигурацией
План сопровождения ПО	Предлагает план мероприятий, требующихся для сопровождения ПО в процессе его эксплуатации, а также расчет стоимости сопровождения и необходимые для этого ресурсы
План по управлению персоналом	Описывает мероприятия, направленные на повышение квалификации членов команды разработчиков

В листинге 1 показан процесс планирования создания ПО в виде псевдокода. Здесь сделан акцент на том, что планирование — это итерационный процесс. Поскольку в процессе выполнения проекта постоянно поступает новая информация, план должен регулярно пересматриваться. Важными факторами, которые должны учитываться при разработке плана, являются финансовые и деловые обязательства организации. Если они изменяются, эти изменения также должны найти отражение в плане работ.

Листинг 1 – Процесс планирования проекта

```
Определение проектных ограничений
Первоначальная оценка параметров проекта
Определение этапов выполнения проекта и контрольных отметок
while пока проект не завершится или не будет остановлен loop
    Составление графика работ
    Начало выполнения работ
    Ожидание окончания очередного этапа работ
    Отслеживание хода выполнения работ
    Пересмотр оценок параметров проекта
    Изменение графика работ
    Пересмотр проектных ограничений
    if (возникла проблема) then
        Пересмотр технических или организационных параметров проекта
    end if
end loop
```

Процесс планирования начинается с определения проектных ограничений (временные ограничения, возможности наличного персонала, бюджетные ограничения и т.д.). Эти ограничения должны определяться параллельно с оцениванием проектных параметров, таких как структура и размер проекта, а также распределением функций среди исполнителей. Затем определяются этапы разработки и то, какие результаты документация, прототипы, подсистемы или версии программного продукта) должны быть получены по окончании этих этапов. Далее начинается циклическая часть планирования. Сначала разрабатывается график работ по выполнению проекта или дается разрешение на продолжение использования ранее созданного графика. После этого (обычно через 2-3 недели) проводится контроль выполнения работ и отмечаются расхождения между реальным и плановым ходом работ.

Большинство менеджеров проектов не думают, что реализация их проектов пройдет гладко, без всяких проблем. Желательно описать возможные проблемы еще до того, как они проявят себя в ходе выполнения проекта. Поэтому лучше составлять "пессимистические" графики работ, чем "оптимистические". Но, конечно, невозможно построить план, учитывающий все, в том числе случайные, проблемы и задержки выполнения проекта, поэтому и возникает необходимость периодического пересмотра проектных ограничений и этапов создания программного продукта.

7.2.3 План проекта

План проекта должен четко показать ресурсы, необходимые для реализации проекта, разделение работ на этапы и временной график выполнения этих этапов. В некоторых организациях план проекта составляется как единый документ, содержащий все виды планов, описанных выше. В других случаях план проекта описывает только технологический процесс создания ПО. В таком плане обязательно присутствуют ссылки на планы других видов, но они разрабатываются отдельно от плана проекта.

План, описанный выше, принадлежит именно к последнему типу планов. Детализация планов проектов очень разнится в зависимости от типа разрабатываемого программного продукта и организации-разработчика. План должен регулярно пересматриваться в процессе реализации проекта. Одни части плана, например график работ, изменяются часто, другие более стабильны. Для внесения изменений в план требуется специальная организация документопотока, позволяющая отслеживать эти изменения.

7.2.4 Контрольные отметки этапов работ

При планировании процесса определяются контрольные отметки— вехи, отмечающие окончание определенного этапа работ. Для каждой контрольной отметки создается отчет, который предоставляется руководству проекта.

Обычно по завершении основных больших этапов, таких как разработка спецификации, проектирование и т.п., заказчику ПО предоставляются результаты их выполнения, так называемые контрольные проектные элементы. Это может быть документация, прототип программного продукта, законченные подсистемы ПО и т.д.

Для определения контрольных отметок весь процесс создания ПО должен быть разбит на отдельные этапы с указанным "выходом" (результатом) каждого этапа. Например, на рис. 1 показаны этапы разработки спецификации требований в случае, когда для ее проверки используется прототип системы, а также представлены выходные результаты (контрольные отметки) каждого этапа. Здесь контрольными проектными элементами являются требования и спецификация требований.

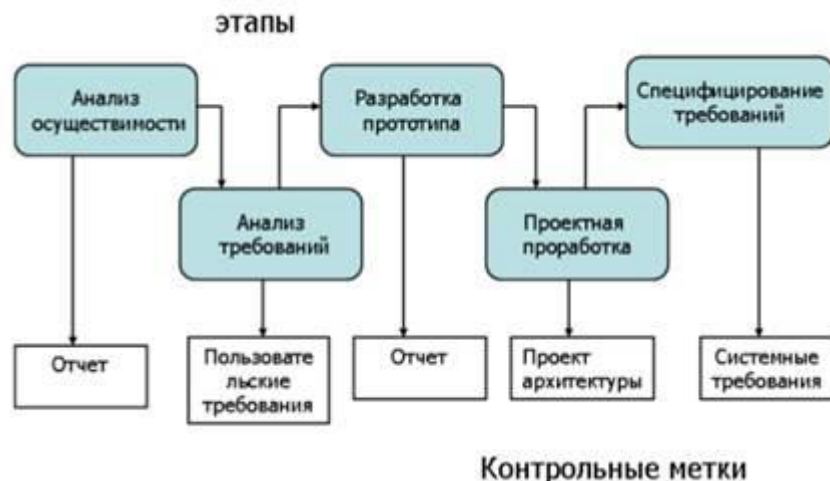


Рисунок 7.1 – Этапы процесса разработки спецификации

7.2.5 График работ

Составление графика – одна из самых ответственных работ, выполняемых менеджером проекта. Здесь менеджер оценивает длительность проекта, определяет ресурсы, необходимые для реализации отдельных этапов работ, и представляет их (этапы) в виде согласованной последовательности. Если данный проект подобен ранее реализованному, то график работ последнего проекта можно взять за основу для данного проекта. Но затем следует учесть, что на отдельных этапах нового проекта могут использоваться методы и подходы, отличные от использованных ранее.

Если проект является инновационным, первоначальные оценки длительности и требуемых ресурсов наверняка будут слишком оптимистичными, даже если менеджер попытается предусмотреть все возможные неожиданности.

В процессе составления графика (рисунок 7.2) весь массив работ, необходимых для реализации проекта, разбивается на отдельные этапы и оценивается время, требующееся

для выполнения каждого этапа. Обычно многие этапы выполняются параллельно. График работ должен предусматривать это и распределять производственные ресурсы между ними наиболее оптимальным образом.

Длительность этапов обычно должна быть не меньше недели. Если она будет меньше, то окажется ниже точности временных оценок этапов, что может привести к частому пересмотру графика работ. Также целесообразно (в аспекте управления проектом) установить максимальную длительность этапов, не превышающую 8 или 10 недель. Если есть этапы, имеющие большую длительность, их следует разбить на этапы меньшей длительности.

7.2.6 Диаграммы процессов и временные диаграммы

Кроме временных затрат, менеджер должен рассчитать другие ресурсы, необходимые для успешного выполнения каждого этапа. Особый вид ресурсов — это команда разработчиков, привлеченная к выполнению проекта. Другими видами ресурсов могут быть необходимое свободное дисковое пространство на сервере, время использования какого-либо специального оборудования и бюджетные средства на командировочные расходы персонала, работающего над проектом.

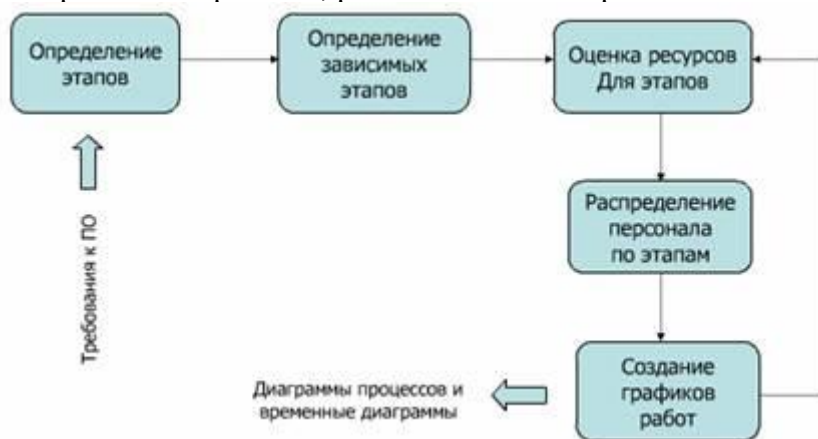


Рисунок 7.2 – Процесс составления графика работ

Существует хорошее эмпирическое правило: оценивать временные затраты так, как будто ничего непредвиденного и "плохого" не может случиться, затем увеличить эти оценки для учета возможных проблем. К исходным расчетным оценкам рекомендуется добавлять 30% на возможные проблемы и затем еще 20%, чтобы быть готовым к тому, что невозможно предвидеть.

График работ по проекту обычно представляется в виде набора диаграмм и графиков, показывающих разбиение проектных работ на этапы, зависимости между этапами и распределение разработчиков по этапам.

Временные и сетевые диаграммы

Временные и сетевые диаграммы полезны для представления графика работ. Временная диаграмма показывает время начала и окончания каждого этапа и его длительность. Сетевая диаграмма отображает зависимости между различными этапами проекта. Эти диаграммы можно создать автоматически с помощью программных средств поддержки управления на основе информации, заложенной в базе данных проекта.

Рассмотрим этапы некоего проекта, представленные в таблице 3.2, из которой, в частности, видно, что этап Т3 зависит от этапа Т1. Это значит, что этап Т1 должен завершиться прежде, чем начнется этап Т3. Например, на этапе Т1 проводится компонентный анализ создаваемого программного продукта, а на этапе Т3 — проектирование системы.

На основе приведенных значений длительности этапов и зависимости между ними строится сетевой график последовательности этапов (рисунок 3.3). На этом графике видно, какие работы могут выполняться параллельно, а какие должны выполняться последовательно друг за другом. Этапы обозначены прямоугольниками. Контрольные отметки и контрольные проектные элементы показаны в виде овалов и обозначены (как и в таблице 3.2) буквой М с соответствующим номером. Даты на данной диаграмме соответствуют началу выполнения этапов. Сетевую диаграмму следует читать слева направо и сверху вниз.

Таблица 7.2 – Этапы проекта

Этап	Длительность (дни)	Зависимость	Этап	Длительность (дни)	Зависимость
T1	8		T7	20	T1 (M1)
T2	15		T8	25	T4 (M5)
T3	15	T1 (M1)	T9	15	T3, T6 (M4)
T4	10		T10	15	T5, T7 (M7)
T5	10	T2, T4 (M2)	T11	7	T9 (M6)
T6	5	T1, T2 (M3)	T12	10	T11 (M8)

Если для создания сетевой диаграммы используются программные средства поддержки управления проектом, каждый этап должен заканчиваться контрольной отметкой. Очередной этап может начаться только тогда, когда будет получена контрольная отметка (которая может зависеть от нескольких предшествующих этапов). Поэтому в третьем столбце таблицы 7.2 приведены контрольные отметки; они будут достигнуты только тогда, когда будет завершен этап, в строке которого помещена соответствующая контрольная отметка.

Любой этап не может начаться, пока не выполнены все этапы на всех путях, ведущих от начала проекта к данному этапу. Например, этап T9 не может начаться, пока не будут завершены этапы T3 и T6. Отметим, что в данном случае достижение контрольной отметки M4 говорит о том, что эти этапы завершены.

Минимальное время выполнения всего проекта можно рассчитать, просуммировав в сетевой диаграмме длительности этапов на самом длинном пути (длина пути здесь измеряется не количеством этапов на пути, а суммарной длительностью этих этапов) от начала проекта до его окончания (это так называемый критический путь). В нашем случае продолжительность проекта составляет 11 недель или 55 рабочих дней. На рисунке 7.3 критический путь показан более толстыми линиями, чем остальные пути. Таким образом, общая продолжительность реализации проекта зависит от этапов работ, находящихся на критическом пути. Любая задержка в завершении любого этапа на критическом пути приведет к задержке всего проекта.

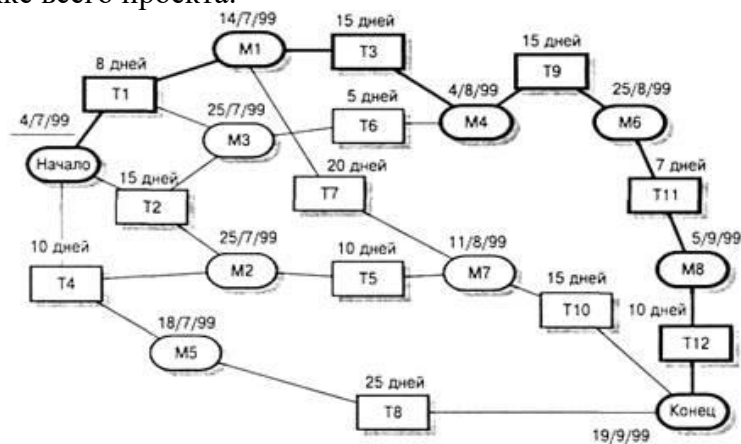


Рисунок 7.3 – Сетевая диаграмма этапов

Задержка в завершении этапов, не входящих в критический путь, не влияет на продолжительность всего проекта до тех пор, пока суммарная длительность этих этапов (с учетом задержек) на каком-нибудь пути не превысит продолжительности работ на критическом пути. Например, задержка этапа Т8 на срок, меньший 20 дней, никак не влияет на общую продолжительность проекта. На рис. 4 представлена временная диаграмма, на которой показаны возможные задержки на каждом этапе.

Сетевая диаграмма позволяет увидеть в зависимости этапов значимость того или иного этапа для реализации всего проекта. Внимание к этапам критического пути часто позволяет найти способы их изменения с тем, чтобы сократить длительность всего проекта. Менеджеры используют сетевую диаграмму для распределения работ.

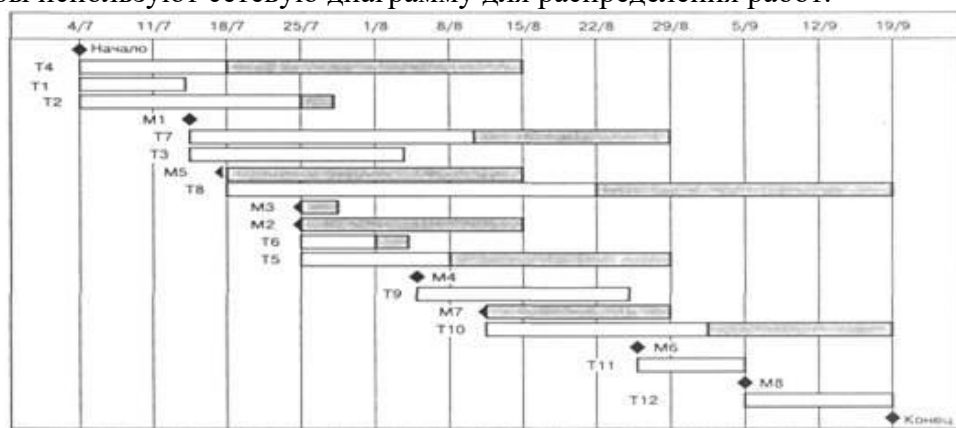


Рисунок 7.4 – Временная диаграмма длительности этапов

На рисунке 7.4 показано другое представление графика работ. Это временная диаграмма (иногда называемая по имени ее изобретателя диаграммой Ганта) может быть построена программными средствами поддержки процесса управления. Она показывает длительность выполнения каждого этапа и возможные их задержки (показаны затененными прямоугольниками), а также даты начала и окончания каждого этапа. Этапы критического пути не имеют затененных прямоугольников; это означает, что задержка с завершением данных этапов приведет к увеличению длительности всего проекта.

Подобно распределению времени выполнения этапов, менеджер должен рассчитать распределение ресурсов по этапам, в частности назначить исполнителей на каждый этап. В таблице 7.3 приведено распределение разработчиков на каждый этап, представленный на рисунке 7.4.

Таблица 7.3 – Распределение исполнителей по этапам

Этап	Исполнитель	Этап	Исполнитель	Этап	Исполнитель
T1	Джейн	T5	Мэри	T9	Джейн
T2	Анна	T6	Анна	T10	Анна
T3	Джейн	T7	Джим	T11	Фред
T4	Фред	T8	Фред	T12	Фред

Приведенная таблица может быть использована программными средствами поддержки процесса управления для построения временной диаграммы занятости сотрудников на определенных этапах работ (рисунке 7.5). Персонал не занят в работе над проектом все время его реализации. В течение периода незанятости сотрудники могут быть в отпуске, работать над другими проектами, проходить обучение и т.д.

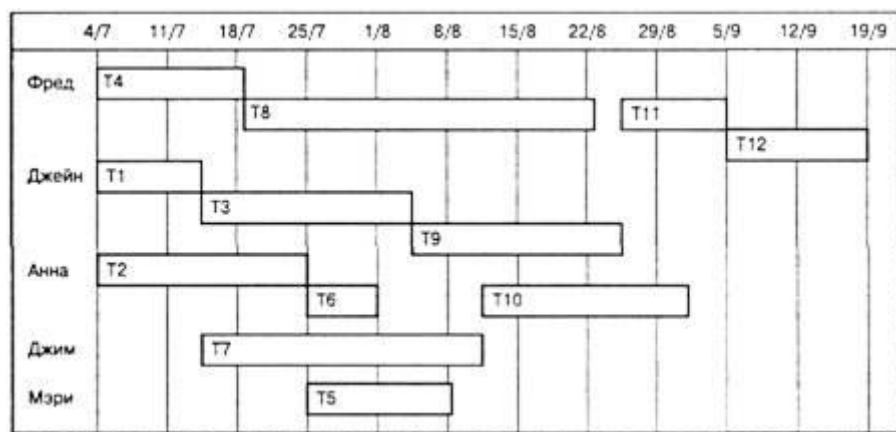


Рисунок 7.5 – Временная диаграмма распределения работников по этапам

7.3 Требования к отчету

- Титульный лист
- Введение
- Состав и структура задач плана внедрения системы
- Состав и расценки на ресурсы плана внедрения системы
- Календарный план внедрения системы
- Заключение

8 Лабораторная работа № 8 «Разработать регламент сопровождения системы»

8.1 Введение

Несмотря на то, что разработка регламентов взаимодействия в процессе поддержки и сопровождения корпоративных бизнес-приложений в промышленной эксплуатации находится, как правило, в зоне компетенции ИТ-подразделений организаций, суть поддержки и сопровождения информационных систем будет не лишним знать и понимать делопроизводителям, документоведам и архивистам. В настоящей работе рассмотрение этого вопроса будет продолжено. Однако рамки будут расширены: мы уделим внимание сопровождению корпоративных бизнес-приложений в целом, включая поддержку и сопровождение систем электронного документооборота, системы электронного архива, кадровых информационных систем и т.д., причем в период промышленной эксплуатации.

8.2 Общие сведения по сопровождению ИС

8.2.1 Постановка задачи

Проект завершен. Система электронного документооборота (или, к примеру, система электронного архива) принята компанией в промышленную эксплуатацию. Сотрудники компании остаются «один на один» с системой. Сотрудников исполняющей организации, готовых прийти на помощь, как в период опытной эксплуатации, уже нет рядом. Конечно, возникнут вопросы: «А вдруг что-то случится? Что делать?». Именно для того, чтобы СЭД работала стабильно, а инциденты, возникающие при работе с СЭД, устранялись быстро и новые пользователи обучались работе с СЭД, необходим внутренний нормативный документ, регулирующий вопросы поддержки и сопровождения информационных систем.

8.2.2 Аспекты, отражаемые в регламенте сопровождения

Предлагаемый ниже регламент устанавливает требования и порядок взаимодействия при поддержке эксплуатации и сопровождения корпоративных бизнес-приложений (КБП), а именно: закрепляет права и ответственность участников взаимодействия, уровни принятия решений и служит для обеспечения оперативного и эффективного взаимодействия между всеми его участниками.

Кроме того, регламентом фиксируются такие аспекты, как:

- Уровни поддержки и сопровождения КБП. Заявки (сообщения от пользователей об инцидентах в КБП) могут быть различной сложности и объединять их не стоит, т.к. для отработки заявок может потребоваться привлечение специалистов различного уровня. Предлагаемые в данном регламенте уровни поддержки и сопровождения:
- Уровень службы поддержки эксплуатации. В компетенции данной службы – вопросы администрирования информационных систем (ИС), обеспечение стабильной работы, исправление технических неполадок в работе ИС, консультаций пользователей по вопросам эксплуатации ИС и т.д. В предлагаемом регламенте служба поддержки эксплуатации – это подразделение сервисной организации, предоставляющее услуги по поддержке бизнес-приложения (т.е. внешней организации).
- Уровень службы сопровождения. Данная служба решает вопросы установки и настройки новых версий ИС, обновлений и дополнений. В предлагаемом регламенте служба сопровождения – это подразделение сервисной организации, предоставляющее услуги по сопровождению бизнес-приложения (т.е. также внешней организации).

- Уровень совета по изменениям. Совет по изменениям обладает необходимыми полномочиями для принятия решений о внесении изменений в то или иное КБП.

В предлагаемом регламенте совет состоит из представителей организаций – пользователей конкретного бизнес-приложения, эксперта от СС, а также представителей служб сопровождения смежных корпоративных бизнес-приложений, уполномоченных принимать решения по внесению изменений в данное бизнес-приложение.

8.2.3 Фиксация и мониторинг всех поступивших заявок.

Ни одна заявка от пользователя не должна быть «пропущена». Все действия по заявке должны быть зафиксированы. Таким образом, при повторении у пользователя ситуации, указанной в заявке, будет ясно, что необходимо делать или не делать, а при возникновении аналогичных ситуаций у других пользователей их проще будет решать. Конечно, такую статистику удобнее вести в какой-нибудь программе. В крайнем случае подойдет и Excel, но это будет уже сложнее. В предлагаемом регламенте вся информация по инцидентам в информационных системах ведется системе SD(Service Desk).

8.2.4 Универсальность регламента.

Принципы поддержки и сопровождения информационных систем универсальны, т.е. подходят и для СЭД и, к примеру, для ERP-системы. Поэтому нет необходимости разрабатывать отдельный регламент поддержки и сопровождения для каждой информационной системы компании.

8.3 Требования к отчету

- Титульный лист
- Введение
- Уровни технической поддержки и сопровождения
- Параметры задач в системе управления задачами
- Создание заявок в системе управления задачами
- Выполнение заявок
- Прочие процессы взаимодействия
- Заключение

9 Лабораторная работа № 9 «Разработать план управления качеством»

9.1 Введение

Качество является отличительной характеристикой, определяющей, в какой мере выполненный проект отвечает ожиданиям сторон, заинтересованных в его успешной реализации. Причем, данная формулировка применима не только к проекту в целом, но и к конечным его результатам и даже к отдельным компонентам конечных результатов. Под таким углом зрения качество становится показателем или характеристикой, которая определяет приемлемость того или иного компонента. Если, например, наш новый проект связан с выпуском шоколадных трюфелей, и каждая конфета должна весить ровно 60 г, то любая из них, имеющая меньший или больший вес, не будет отвечать установленному нами стандарту качества. Такие конфеты следует отбраковывать и упаковывать в коробки под другой торговой маркой.

9.2 Общая информация по управлению качеством в ИТ-проектах

9.2.1 Понятие качества

Качество зависит от масштаба, времени и затрат. В нашем примере с шоколадными трюфелями можно было бы сократить затраты, закупая более дешевый порошок какао. Однако в этом случае снизится качество шоколадных трюфелей, на что, конечно же, обратят внимание клиенты. Наверняка существует возможность и ускорить производство таких конфет, но это также повлияет на их вкусовые качества, а значит, снизится качество продукта в целом. Вывод из сказанного таков: если вам предлагают сделать что-либо дешевле или быстрее, то вы практически гарантированно снизите качество конечного результата.

9.2.2 Показатели качества

В качестве критерия приемлемости можно использовать показатели качества. План управления качеством детализирует спецификации и критерии, используемые для определения полноты и правильности соответствующего конечного результата (или любого другого элемента проекта). Он, конечно же, зависит от типа выполняемого проекта, т.е. план, разработанный для проекта, исследующего новый фармацевтический препарат, будет существенно отличаться от плана, разработанного для проекта строительства кондоминиума. Самыми важными элементами, которые должны быть освещены в плане управления качеством, являются спецификации конечных результатов и критерии соблюдения этих спецификаций. Помимо обязательных компонентов при составлении плана важно также учитывать элементы, перечисленные ниже.

9.2.3 Стандарты качества

Под стандартами мы подразумеваем директивы, правила или характеристики, которых необходимо придерживаться при производстве продукта соответствующего проекта. Определенные стандарты может установить ваша организация или предусматривать отрасль, в которой вы работаете. Стандарты, в отличие от предписаний, предусматриваются не всегда, но если они установлены, их обязательно следует придерживаться.

Институт управления проектами (PMI) немало потрудился над разработкой стандартов и правил, которые обеспечивают высокое качество управления проектами. Стандарты, разработанные PMI, могут служить ярким примером отраслевых стандартов.

9.3 Структура плана управления качеством

9.3.1 Предписания

Предписания обязательны к исполнению. Правительства, учреждения и организации наделены правом издавать предписания. Нарушение предписаний грозит вам (или вашей организации) штрафными санкциями, судебными исками или даже распоряжениями о прекращении работы.

9.3.2 Политика обеспечения качества

Политики обеспечения качества разрабатываются организацией и, как правило, обнародуются высшим руководством организации. Они включают инструкции, касающиеся стандартов качества проектов.

9.4 Порядок работы над планом управления качеством

Прежде чем приступить к выполнению проекта, обязательно ознакомьтесь со всеми стандартами, предписаниями и политиками, касающимися выполнения проектов в вашей организации, и либо включите их в свой план управления качеством проекта, либо сошлитесь на них в этом плане.

Ваш план управления проектом также может включать план управления качеством проекта. Вы имеете возможность сослаться на любые стандарты или предписания, необходимые для самого плана. Например, ваша организация может требовать, чтобы всеми проектами, бюджет которых превышает 5 млн долл., управляли только руководители проектов, имеющие сертификат специалиста по управлению проектами (Project Management Professional — PMP). Кроме того, в плане управления качеством проекта необходимо зафиксировать любые предписания, которыми вы будете пользоваться при создании документов планирования проекта. Например, вы можете потребовать, чтобы формулировка масштаба проекта утверждалась определенными членами высшего руководства вашей организации.

Процесс обеспечения качества также должен быть отражен в плане управления качеством проекта. Этот процесс должен описывать, каким образом непосредственные исполнители проекта будут обеспечивать соблюдение соответствующих стандартов и критериев качества. Процесс обеспечения качества может использоваться руководителем проекта, поставщиками и всеми остальными, кто несет ответственность за качество проекта. Аудиты качества также могут быть частью процесса обеспечения качества. Такие аудиты проводятся с целью выявления неэффективных или экономически неоправданных процессов в выполняемом проекте и обеспечения соблюдения соответствующих стандартов и предписаний.

9.5 Требования к отчету

- Титульный лист
- Ключевые показатели качества ПО
- Распределение ответственности за задачи управления качеством
- Чек-лист внедрения плана управления качеством
- Заключение

10 Лабораторная работа № 10 «Отчет по управлению конфигурацией»

10.1 Введение

Конфигурационное управление (англ. software configuration management, SCM) в программной инженерии — комплекс методов, направленных на систематический учёт изменений, вносимых разработчиками в программный продукт в процессе его разработки и сопровождения, сохранение целостности системы после изменений, предотвращение нежелательных и непредсказуемых эффектов, формализацию процесса внесения изменений.

В целом, конфигурационное управление отвечает на вопрос: «Кто-то уже сделал нечто, как нам это воспроизвести?»

Изначально управление конфигурацией применялось не в программировании. Под конфигурацией понимался состав деталей конечного продукта и «взаимное расположение частей» физического изделия. Таким образом, конфигурацией можно управлять, контролируя документы, описывающие конечный продукт, требования к нему, всю его проектную и технологическую документацию.

В связи с высокой динамичностью сферы разработки ПО, в ней конфигурационное управление особенно полезно. К процедурам можно отнести создание резервных копий, контроль исходного кода, требований проекта, документации и т. д. Степень формальности выполнения данных процедур зависит от размеров проекта, и при правильной её оценке данная концепция может быть очень полезна.

10.2 Процесс и инструменты управления конфигурацией

10.2.1 Общие соображения

Инструменты управления конфигурацией позволяют выполнять изменения и развертывания быстрее, повторяемыми, масштабируемыми, предсказуемыми и способными поддерживать желаемое состояние, которое переводит контролируемые активы в ожидаемое состояние.

Некоторые инструменты управления конфигурацией используют модель извлечения, в которой агент, установленный на серверах, периодически запускается, чтобы извлечь последние определения из центрального репозитория и применить их к серверу. Другие инструменты используют push-модель, когда центральный сервер запускает обновления для управляемых серверов.

10.2.2 Процесс управления конфигурацией программного обеспечения

Управление конфигурацией программного обеспечения определяется как процесс систематического управления, организации и контроля изменений в документах, кодах и других объектах в течение жизненного цикла разработки программного обеспечения. Сокращенно - SCM. Основная задача - повысить производительность труда с минимальными ошибками.

10.2.3 Необходимость управления конфигурацией

Основными причинами внедрения системы управления конфигурацией программного обеспечения являются:

- Есть много людей, работающих над программным обеспечением, которое постоянно обновляется

- Это может быть случай, когда несколько версий, ветвей, авторов участвуют в программном проекте, и команда географически распределена и работает одновременно
- Изменения в требованиях пользователей, политике, бюджете, расписаниях должны быть учтены
- Программное обеспечение должно работать на разных машинах и в операционных системах
- Помогает развивать координацию между заинтересованными сторонами
- Процесс SCM также полезен для контроля затрат, связанных с внесением изменений в систему

10.2.4 Задачи процесса управления конфигурацией

Любые изменения в элементах конфигурации программного обеспечения влияют на конечный продукт. Следовательно, изменения в элементах конфигурации должны контролироваться и управляться.

Задачи в процессе SCM:

- Идентификация конфигурации
- Исходные условия
- Изменение управления
- Учет состояния конфигурации
- Аудит конфигурации и обзоры

10.3 Версионный контроль в ИТ-проектах

10.3.1 Системы версионного контроля

Система контроля версий – это программное обеспечение, позволяющее создавать версии элементов и работать с этими версиями, как с самостоятельными элементами. В англоязычных источниках используется термин version control systems, сокращенно VCS. Работа с версиями предполагает как создание самих версий, так и структуры для их хранения. Как правило, это или цепочки, или деревья.

10.3.2 Конфигурационные единицы

Прежде чем работать с элементами и их версиями, надо эти элементы создать, т.е. дать указать системе контроля версий взять имеющиеся объекты реального мира и поместить их под свой контроль. Вместе с самим элементом всегда создается и его первая версия.

Чаще всего в качестве элементов для контроля версий выступают:

- файлы;
- директории;
- hard- и softlinks.

Внутри системы контроля сами элементы могут размещаться по-разному – это зависит от архитектуры VCS. Пользователю важно лишь знать, что элемент помещается внутри хранилища и работа с ним идет с помощью команд выбранного инструментария.

10.3.3 Ветвление и слияние версий

Как уже было сказано, системы контроля должны предоставлять структуры для хранения версий. Самым распространенным представлением подобной структуры является дерево версий. Это такая организация версий элемента, при которой на основе любой

версии элемента конфигурации может быть создано несколько наборов последовательностей его версий. При этом отдельный набор версий, происходящий из произвольной версии, называется веткой. И поскольку ветка содержит версии, то каждая из версий может быть источником для создания других веток. Короче, дерево.

Название модели говорит само за себя: у растений (элементов) появляются почки и листья (версии), из них, в свою очередь, ветки. На ветках – листья (другие версии) и другие ветки. На них, опять же, произрастает всё та же растительность. В результате растёт дерево, у которого крона – это множество версий. Один элемент – одно дерево.

Зачем нужна вся эта конструкция? Неужели нельзя просто наращивать версии одну за другой? Конечно, можно. Однако это сразу ограничит возможности использования подобной системы. Если версии появляются одна за одной, то в один момент времени создать новую версию сможет только один из пользователей, работающих с системой, остальные вынуждены будут подождать. Более того, когда появится новая версия, каждому надо будет соединить свои изменения с текущими наработками. И так – пока все желающие не поместят свои наработки в цепочку версий. При этом каждый должен будет убедиться, что слияние версий не привело к поломке системы. И, кроме того, пока все изменения не будут помещены таким образом под контроль, всем из ожидающих придется сохранять промежуточные результаты где-то локально, не смешивая с тем, что находится в настоящий момент в работе. И ладно, если пара человек работает над десятком элементов – они всегда смогут договориться. А если масштабы гораздо больше? Добавим десяток человек (даже не увеличивая количества элементов) – и подобные простые цепочки полностью застопорят работу. В общем, линейная структура версий порождает множество сложностей.

Итак, ясно, что без веток не обойтись. Но ведь не растить же ветку на малейший чих разработчика? Посмотрим, в каких же случаях отращаются ветки. Типовые примеры веток таковы:

- ветка для запроса на изменения — заводится для версий, создаваемых в ходе работы по запросу на изменение («девелоперская», или «сиарная», ветка);
- интеграционная ветка — служит промежуточным хранилищем для процесса стабилизации;
- релизная ветка — для выкладывания версий при стабилизации конфигурации (см. соответствующий раздел первой части статьи). Какие-то версии на ветке могут быть в дальнейшем объявлены частью базовой конфигурации;
- отладочная («дебажная») ветка — для кратковременного хранения версий, в основном, для целей проверки каких-либо решений.

10.4 Требования к отчету

- Титульный лист
- Введение
- Структура конфигурационных единиц
- Информация о структуре учебного проекта в системе управления вериями
- Карта развития версий по учебному проекту
- Заключение

11 Лабораторная работа № 11 «Разработка бизнес-архитектуры по А.Остервальдеру»

11.1 Введение

Стратегическое планирование важно для каждого бизнеса. Один из самых популярных инструментов — бизнес-модель Остервальдера. Он простой и эффективный, подходит как для развивающихся, так и уже давно работающих компаний. В этой статье поговорим о истории появления модели и подробно разберем каждый блок.

11.2 Описание модели Остервальдера-Пинье

11.2.1 Что такое бизнес-модель Остервальдера?

Бизнес-модель Остервальдера (Business Model Canvas) — инструмент стратегического управления, используемый для описания бизнес-моделей новых или уже работающих предприятий. Представляет собой схему из 9 блоков, описывающих разные бизнес-процессы организации.

Модель создали Александр Остервальдер и Ив Пинье. Подробное описание схемы они дали в книге «Alexander Osterwalder & Yves Pigneur: The Business Model Generation».

Работающие фирмы используют модель для поиска новых точек роста, анализа конкурентов и определения лучших практик развития бизнеса. Существует заблуждение, что инструмент применяют в стартапах и маленьких фирмах, но на самом деле его используют такие «гиганты», как IBM, Ericsson, Deloitte и многие другие.

На стадии планирования стартапа применение Канвас затруднительно. Заполнение всех блоков возможно, когда найдены поставщики и партнеры, определены каналы сбыта и подсчитаны издержки.

Создатели инструмент (Остервальдер и Пинье) в своей научной работе рекомендуют предпринимателям не ограничиваться составлением одной модели. Для поиска оптимального варианта задавайте себе сложные вопросы, учитывайте различные сценарии развития компании и тогда сможете выбрать лучшую бизнес-модель, которая окажет положительный эффект на развитие бизнеса.

11.2.2 Что нужно сделать перед построением бизнес-моделей?

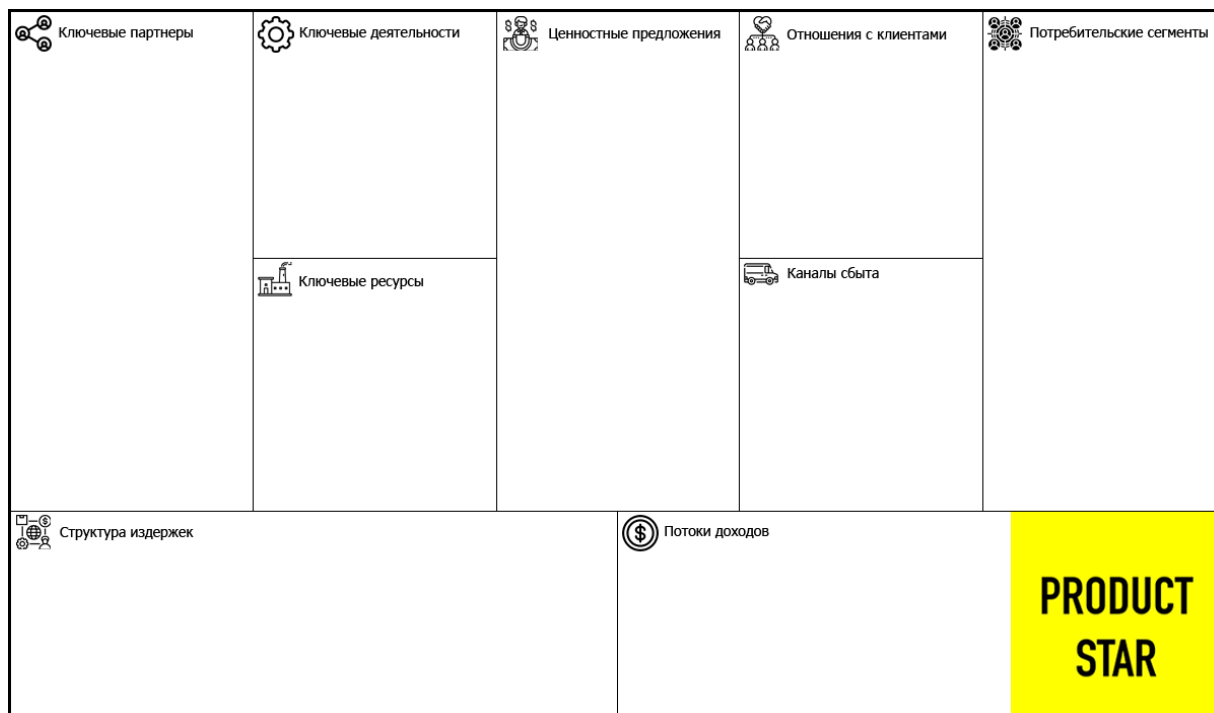
Перед составлением первой бизнес-модели сделайте следующее:

1. Проанализируйте разные направления деятельности. Регистрируясь в качестве индивидуального предпринимателя, создавая компанию или планируя новый продукт, выбирают несколько видов деятельности. Как правило, исходят из планов на будущее: чем планируют заниматься. На этом этапе подумайте, в каких сферах будете работать с наибольшей долей вероятности.
2. Выберите приоритетное направление. Из обилия рассматриваемых сфер деятельности выберите приоритетную — ту, от которой планируете получать большую часть прибыли. Вокруг нее выстраивайте долгосрочную стратегию развития бизнеса.
3. Составьте ассортимент предлагаемой продукции. Проанализируйте рынок, узнайте, какие товары пользуются спросом, а какие продаются плохо. Для достижения наиболее эффективного результата рассматривайте первую группу продукции, от второй откажитесь.
4. Продумайте и сформируйте рекламную стратегию, выберите инструменты продвижения продукции и методы конкурентной борьбы. Подумайте о маркетинговой стратегии: как продвигать товар на рынке, кому продавать (ваша целевая аудитория), на какие конкурентные преимущества обращать внимание клиентов, чтобы обойти конкурентов и т.д.

Если говорить проще, перед составлением первых бизнес-моделей Остервальдера на руках должен быть хотя бы минимальный план по развитию компании: основное направление деятельности, ассортимент товаров, маркетинговая стратегия и т.п.

Канвас поможет раскрыть стратегические задумки и найти новые точки роста. Если же делать план, не имея представления о бизнесе, толку не будет.

11.2.3 Структура модели Остервальдера



Модель состоит из 9 блоков:

1. Потребительские сегменты.
2. Ценностные предложения.
3. Каналы сбыта.
4. Отношения с клиентами.
5. Потоки доходов.
6. Ключевые ресурсы.
7. Ключевые виды деятельности.
8. Ключевые партнеры.
9. Структура издержек.

Далее подробно рассмотрим принципы заполнения блоков в такой последовательности.

11.3 Порядок заполнения бизнес-модели Остервальдера

11.3.1 Потребительские сегменты (Customer Segments)

Сегмент — группа людей с общей проблемой или потребностью. В этом блоке описывайте все сегменты, на которых ориентирована деятельность компании. От точности определения сегментов зависит успешность рекламных кампаний в будущем.

Например, продаете бухгалтерский софт. Сегмент — бухгалтера в коммерческих организациях. В таком случае рекламная кампания пройдет успешно. Но если ошибиться и определить в качестве сегмента менеджеров, получите низкую конверсию и потерянный рекламный бюджет.

В заполнении блока помогут ответы на вопросы:

- Для кого создаем продукты (оказываем услуги, производим товар и т.п.)?

- Как можно охарактеризовать наших потребителей одним словом?
- Можно ли быстро найти целевую аудиторию и поговорить с ней о предлагаемом продукте?
- Кто из разных групп покупателей наиболее важен?

11.3.2 Ценностные предложения (Value propositions)

Определите, почему потребители должны покупать ваш продукт (товар, услугу), а не обращаться к конкуренту. Ценностное предложение должно решать какую-то проблему клиента или закрывать одну из потребностей. Как правило, оно представляет собой совокупность товаров или услуг для конкретного сегмента.

Подумайте, в чем заключается главная ценность вашего продукта для потребителей. Если с этим возникают проблемы, обратитесь к аудитории, пусть они сами расскажут, какие конкретно проблемы им удалось решить после покупки/заказа. В общем, задача определить, чем вы лучше конкурентов.

В заполнении блока помогут ответы на вопросы:

- Какую ценность мы предоставляем потребителям?
- Какие проблемы помогаем им решить?
- Какие потребности удовлетворяем?
- Из чего состоит продукт/товар/услуга?

11.3.3 Каналы сбыта (Channels)

Каналы сбыта — точки контакта с потребителями. К ним относится все от информирования до послепродажного обслуживания. Если затрудняетесь с заполнением блока, воспользуйтесь шаблонными каналами сбыта:

- Информирование. Как доносится до потребителя ценностное предложение?
- Оценка. Как позиционируется продукт на фоне конкурентов?
- Продажа. Как происходит продажа?
- Доставка и адаптация. Какими методами осуществляется доставка до клиента и формирование первого позитивного впечатления о товаре?
- Обслуживание. Как обеспечивается послепродажное обслуживание?

Все каналы сбыта очень важны. Не думайте, что контакт с клиентом заканчивается на продаже. Постоянно «касайтесь» его после сделки, чтобы побудить на повторную покупку. Но чтобы это работало, придется спланировать, как минимум, эти 5 каналов сбыта.

В заполнении блока помогут ответы на вопросы:

- Какие каналы взаимодействия позволят пообщаться с нашими клиентами?
- Как мы взаимодействуем с ними сейчас?
- Какие из них наиболее эффективны?
- Какие наиболее выгодны?

11.3.4 Отношения с клиентами (Customer relationships)

Отношения с клиентами — методы взаимодействия с потребителями. Подумайте, как вы строите общение с целевой аудиторией? И строите ли вообще? Выделяют несколько типов взаимоотношений с клиентами:

- персональная поддержка;
- самообслуживание;
- бесплатное или условно-бесплатное пользование;
- совместное создание;
- индивидуальное или групповое обучение.

Перед заполнением блока также подумайте, какие задачи стоят перед бизнесом в данный момент? В зависимости от этого отношения с клиентами могут развиваться по нескольким сценариям:

- Привлечение для разовой продажи (например, автомобильный салон).
- Удержание для регулярного сотрудничества (например, сервис технического обслуживания).
- Классификация для работы с потребителями определенного типа (например, водители внедорожников или спортивных автомобилей).

То есть сначала определить текущие задачи, а затем думать о типе отношений с клиентами и как с ними взаимодействовать.

В заполнении блока помогут ответы на вопросы:

- Каких отношений ждут клиенты?
- Какие отношения есть сейчас?
- Почему отношения стали такими?
- Сходятся ли они с текущей бизнес-моделью?

11.3.5 Потоки доходов (Revenue streams)

В этом блоке опишите, как бизнес зарабатывает деньги. Существует несколько способов формирования потоков доходов:

- Продажа товаров. Самый популярный источник заработка компаний. Сводится к продаже товара/продукта конечному потребителю, дилерским сетям и т.п.
- Плата за использование услуги. Клиент пользуется услугой и платит за время или объем. Например, создание сайта в соответствии с техническим заданием. Организация определяет необходимое количество времени для выполнения заказа, исходя из этого формируется конечная стоимость.
- Оплата подписки. Фиксированная плата за использование чего-либо на протяжении определенного промежутка времени. Например, месячная подписка за доступ к онлайн-кинотеатру.
- Аренда. Временное использование актива по фиксированной ставке без уплаты его полной стоимости. Например, дата-центр сдает в аренду сервера. Потребитель не платит полную стоимость оборудования, а лишь некоторую часть в зависимости от срока использования.
- Лицензия. Доход возникает в результате временной передачи клиенту интеллектуальной собственности.
- Комиссия. Бизнес выступает в роли посредника и получает за это определенную комиссию. Например, площадка для продажи сайтов взимает 3% от суммы каждой сделки за то, что выступает в роли гаранта и защищает обе стороны от мошенничества.
- Реклама. Продажа целевой аудитории рекламодателям. Например, размещение рекламных баннеров на страницах сайта.

Описанные потоки доходов имеют свои механизмы ценообразования, которые напрямую связаны с ценностным предложением.

В заполнении блока помогут ответы на вопросы:

- За что клиенты готовы платить?
- За что они платят сейчас?
- Каким образом они платят?
- Как они предпочли бы платить?
- Какую часть от общей прибыли приносит каждый поток?

11.3.6 Ключевые ресурсы (Key resources)

Опишите необходимые для функционирования и масштабирования бизнес-модели активы. Это должны быть ресурсы, которые помогают доносить по клиентам ценностное предложение, поддерживать с ними связь и получать прибыль от деятельности.

По каждому выделенному активу уточните источники получения: приобретение в собственность, аренда, заимствование у партнеров и т.п.

Ориентируйтесь на 4 популярные категории активов:

- Материальные ресурсы. Физические объекты — сырье, станки, транспортные средства, недвижимость, точки продаж и т.п.
- Интеллектуальные ресурсы. Знания — технологии, патенты, программный код, бренды, товарные знаки и т.п.
- Персонал. Люди — маркетологи, менеджеры, программисты, механики, столяры, маляры и т.п. Те, кто отвечают за создание продуктов, оказание услуг, производство.
- Финансы. Деньги — оборотные средства, кредиты, инвестиции и т.п.

В заполнении блока помогут ответы на вопросы:

- Какие ресурсы нужны для создания и реализации ценностных предложений? Отношения с нашими клиентами? Каналы сбыта?

11.3.7 Ключевые деятельности (Key activities)

В прошлом блоке вы описали материалы, используемые для создания ценностных предложений — ответили на вопрос «чем?». Теперь подумайте и опишите основные работы, осуществляемые для реализации ценностных предложений — ответьте на вопрос «как?». Иными словами — опишите операционную деятельность бизнеса.

Упростим задачу описанием классификаций основных видов деятельности:

- Производство. Если вы производите какой-то товар, подробно опишите процедуры закупки сырья, логистики до места производства, обеспечения бесперебойной работы оборудования, контроля качества конечной продукции.
- Решение проблем. Если бизнес-модель предусматривает оказание услуг для решения проблем потребителей, опишите подходы к работе, систему управления знаниями, проводимые исследования, обучение персонала, диагностические мероприятия, способы контроля удовлетворенности клиентов.
- Управление инфраструктурой. Если компания представляет собой сервис, платформу или приложение, опишите планирование и разработку новых функций, тестирование, поддержку пользователей, управление опытом потребителя.

В заполнении блока помогут ответы на вопросы:

- Что нужно делать для поддержания ценности продукта?
- Без чего компания не может существовать?
- Что необходимо делать регулярно для постоянного повышения качества работы?

11.3.8 Ключевые партнёры (Key partners)

Здесь все просто — перечислите компании, с которыми сотрудничаете на постоянной основе для создания ценностного предложения. У производителя хлеба —

поставщики муки, у игровой студии — школа повышения квалификации разработчиков, у онлайн-бухгалтерии — коллегия бухгалтеров для проверки правильности работы сервиса.

Если говорить проще, то партнеры — сторонние компании, которые поставляют недостающие или непрофильные части бизнес-модели.

В заполнении блока помогут ответы на вопросы:

- Партнерство с какими компаниями помогает снижать риски?
- Кто может стать нашим поставщиком?
- Какие виды деятельности можно передать партнерам без ущерба качества?

11.3.9 Структура издержек (Cost structure)

Последний блок очень важен. Вы должны учесть все затраты, которые несет компания при создании ценностных предложений. Заполняйте блок на основе определенных ранее ресурсов, основных видах деятельности и партнерах.

Дополнительно разделите издержки на фиксированные (которые не зависят от объемов производства) и фиксированные (которые зависят от объемов производства).

В заполнении блока помогут ответы на вопросы:

- Какие важные расходы мы несем для производства продукта?
- Какие ресурсы для нас наиболее дороги?
- Какие виды деятельности требуют наибольших затрат?

Бизнес-модель Остервальдера дает общее представление о бизнесе: что делаем, кому продаем, сколько тратим, откуда получаем прибыль и т.п. Инструмент пользуется популярностью из-за простоты. Одна страница, 9 блоков — специальных знаний в сфере стратегического планирования для заполнения таблицы не требуется. Обобщенная информация дает понять, где есть проблемы, с чем работать не стоит, а что, наоборот, можно улучшить.

11.4 Требования к отчету

- Титульный лист
- Введение
- Заполненная канва модели Остервальдера-Пинье
- Заключение

12 Лабораторная работа № 12 «Разработка архитектурной модели предприятия»

12.11. Введение

В настоящее время интерес к тематике архитектуры предприятия (АП) растет как в профессиональной, так и в академической среде. Методы и технологии АП позволяют работать со знаниями об устройстве компании и применяются в проектах трансформации и инжиниринга предприятий. По направлению АП пока нет профессиональных стандартов и общепризнанных учебных материалов. Наряду с динамичностью, новизной и междисциплинарностью АП принадлежит к числу практически-направленных дисциплин, что создает дополнительные ограничения для ее освоения в традиционной вузовской среде.

12.2 Методология управления архитектурой предприятия

12.2.1 Основные принципы методологии АП

Методология управления АП должна удовлетворять следующим требованиям:

- Иллюстрация идеи многослойности АП: взаимосвязь организационно-управленческих объектов с информационными системами и технологической инфраструктурой компании (предприятие как экономическая, социальная и техническая система);
- Рассмотрение организации с разных аспектов и ответы на комплекс общесистемных вопросов:
 - Почему?
 - Что?
 - Где?
 - Как?
 - Кто?
 - Когда?
- Использование только тех объектов и артефактов, которые необходимы для большинства сценариев использования АП;
- Прозрачность взаимосвязи артефактов с интересами заинтересованных сторон;
- Привязка моделей и методов АП к этапам типовых проектов по трансформации предприятия или ИТ-проектов (возможность использования проектного подхода в обучении, формирование компетенций через практическую деятельность);
- Максимальная согласованность с имеющимися стандартами и популярными методологиями управления АП (TOGAF, ArchiMate);
- Наличие базовой части (необходимый минимум для знакомства с дисциплиной) и дополнительных расширений для адаптации под конкретные образовательные программы;
- Иллюстрация пересечений с базовыми дисциплинами, которые традиционно преподаются студентам, изучающим АП: управление/моделирование бизнес-процессами, проектирование информационных систем, управление проектами;

- Связность частных моделей – все создаваемые частные модели взаимосвязаны в единой электронной модели; объекты, возникшие в одной частной модели, используются в другой.

12.2.2 Слои, аспекты и объекты архитектуры предприятия

Для структурирования АП предлагается выделять следующие слои и аспекты.

Слои:

- Бизнес-слой (или организационно-управленческий слой),
- Слой информационных систем,
- Технологический слой.
- Такой состав слоёв согласуется с TOGAF и ядром ArchiMate (Рис. 1).

Аспекты:

- Цели (смысл),
- Деятельность (функция),
- Структура (акторы),
- Объект деятельности.

Такой состав аспектов согласуется с триадой системного подхода и с ядром ArchiMate (Рис. 12.1).

Помимо слоев в предлагаемой методологии заложена метамодель, то есть, определены моделируемые объекты (Рис. 12.1, 12.2) и определены взаимосвязи между ними (Рис. 12.2).



Рис. 12.1. Слои, аспекты и объекты архитектуры предприятия

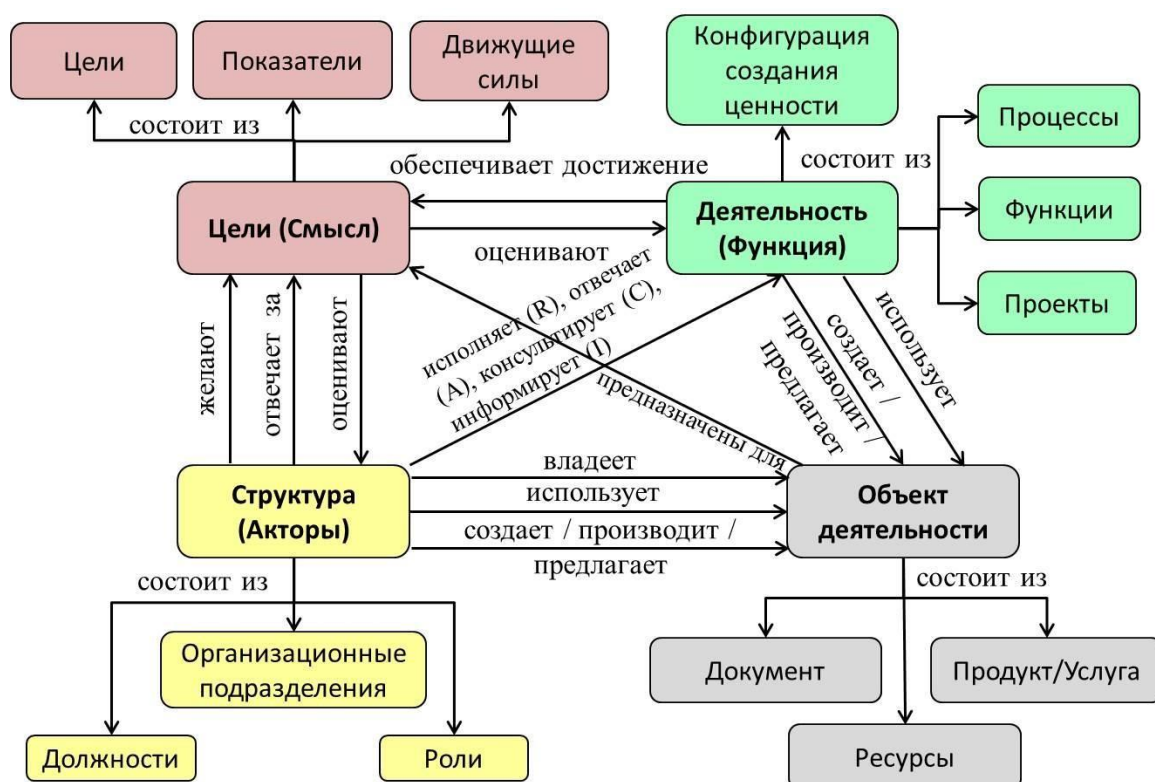


Рис. 12.2. Взаимосвязь объектов бизнес-слоя

12.2.3 Деятельность по созданию и использованию моделей АП

В реальных проектах деятельность по созданию и использованию моделей АП будет варьироваться в зависимости от сценариев использования АП (организационное развитие, цифровая трансформация и др.) и прочих ограничений (временных, ресурсных и т.п.). В учебных проектах состав работ и создаваемых моделей также будет варьироваться в зависимости от образовательной программы (её уровня, специализации, ...).

Состав задач управления АП легко представляется в виде проекта преобразования АП. Необходимость следовать жизненному циклу информационных систем при разработке ИТ-решений также укладывается в предложенный метод управления АП.

12.2.4 Базовая часть

Табл. 12.1. Задачи управления АП и создаваемые артефакты, базовая часть

Задачи управления АП	Создаваемые артефакты
1. Начальный этап	
Заинтересованные стороны и их интересы	Карта заинтересованных сторон
Цели и задачи проекта	Модель мотивации
2. Этап анализа существующей архитектуры предприятия	Модели состояния «как есть»
Общий взгляд на предприятие	Бизнес-модель «как есть»
Идентификация ценностного предложения (ЦП)	Дерево продуктов/услуг, Профиль ЦП
Выявление конфигурации создания ценности «как есть»	Модель создания ценности

Выявление архитектуры деятельности «как есть»	Функциональная модель Модель процесса
Выявление целей компании и показателей для их измерения	Дерево целей и/или карта стратегий, Таблица показателей,
Организационная структура и матрица ответственности «как есть»	Органиграмма Матрица ответственности
Анализ обработки и использования объектов деятельности «как есть»	Модель объектов деятельности (концептуальная модель бизнес слоя)
ИТ-архитектура «как есть» (существующие ИС и технологическая инфраструктура)	Модель использования ИС, Описание ландшафта приложений, Модель использования инфраструктуры
Общее представление об АП «как есть»	Верхнеуровневая (обзорная) модель АП
3. Этап проектирования целевой архитектуры предприятия	Модели состояния «как надо» с визуализацией изменений
Разработка видения целевой АП Разработка целевой АП с детализацией представлений по слоям <i>* на данном этапе может разрабатываться несколько альтернативных сценариев</i>	Бизнес-модель «как надо»; Верхнеуровневая (обзорная) модель АП «как надо»; Частные модели АП, которые будут затронуты изменениями (состав моделей как при описании текущего состояния), «как надо»
4. Этап реализации и перехода	
Планирование перехода между состояниями АП (как есть, как надо, промежуточное/-ые)	Диаграмма перехода
Формирование портфеля проектов развития	Карточки проектов трансформации и программ развития (предлагаемые инициативы)
Планирование реализации и перехода (см. управление проектами)	Календарный план проектов трансформации (например, в MS Project)
5. Этап оценки реализации архитектуры	

12.3 Требования к отчету

- Титульный лист
- Введение
- Описание бизнес-слоя (организационно-управленческого слоя)
- Описание слоя информационных систем,
- Описание технологического слоя
- Описание общей архитектурной модели, включая срезы As-Is и To-Be
- План внедрения разработанной архитектуры To-Be
- Заключение

13 Лабораторная работа № 13 «Разработать регламент компании-разработчика ПО»

13.1 Введение

Для чего предприятия регламентируют свои бизнес-процессы? Приводит ли это к заметному росту эффективности?

Если бизнес-процессы предприятия не регламентированы, то работа ведется на основе устоявшихся норм и правил, хранящихся в умах сотрудников. Результат достигается, и не важно, хорошо ли, плохо ли. При такой организации работы неизбежны серьезные потери различных видов ресурсов (финансовых, материальных, человеческих, временных).

Даже если документировать такую работу (описать работу, оформить регламент), важно понимать, что регламентация, не сопровождаемая анализом и изменениями существующих процессов, эффекта почти не дает.

13.2 Процесс регламентации

13.2.1 Понятие регламентации

ГОСТР ИСО 9001—2015:

4.4 Система менеджмента качества и ее процессы

4.4.2 Организация должна в необходимом объеме:

а) разрабатывать, актуализировать и применять документированную информацию для обеспечения функционирования процессов;

б) регистрировать и сохранять документированную информацию для обеспечения уверенности в том, что эти процессы осуществляются в соответствии с тем, как это было запланировано.

Для соответствующего выполнения бизнес-процессов разрабатывают регламенты и инструкции.

Регламентация – это установление подробных правил, определяющих порядок деятельности государственного органа, учреждения, организации и др. (Большая советская энциклопедия. — М.: Советская энциклопедия. 1969—1978)

Введем общие понятия регламента и инструкции.

Регламенты описывают процессы взаимодействия исполнителей, инструкции – порядок действий отдельных исполнителей при реализации бизнес-процессов.

13.2.2 Цели регламентации

Процессы нужно описывать и регламентировать для:

- Анализа проблем, узких мест, потерь при выполнении процессов с последующей разработкой и реализацией мероприятий по улучшению;
- Стандартизации деятельности, обеспечения четкой повторяемости процессов и возможности управления ими;
- Тиражирования опыта в другие организации (филиалы, новые предприятия);
- Накопления знаний и передачи их новым сотрудникам (при обучении, приеме на работу)
- Для проведения внутренних аудитов.

13.2.3 Проблемы регламентации

В российских компаниях вопрос регламентации решается часто формально. Сказывается отсутствие понимания целей и задач регламентации, адекватных методик и опыта, подготовленных специалистов.

Часто отсутствует предварительно построенная системы процессов (рис. 13.1), четко фиксирующая на всех уровнях рассмотрения взаимодействия и взаимозависимости регламентируемого бизнес-процесса.

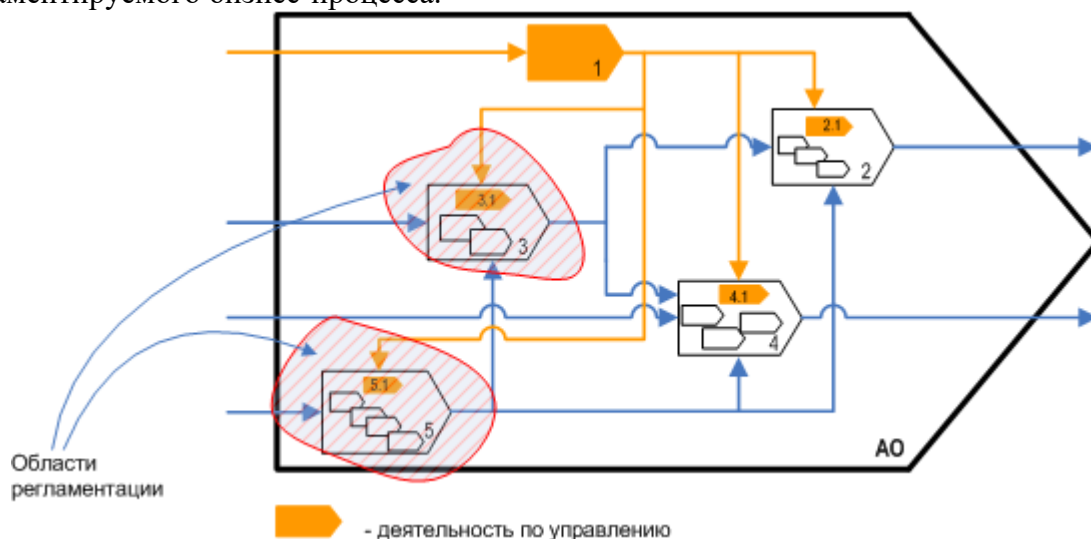


Рис.13.1 Области регламентации при наличии системы бизнес-процессов

Без предварительно построенной системы процессов традиционны следующие ошибки:

- смешиваются процессы разных уровней;
- неправильно определены границ процессов;
- регламентация проводится бессистемно, под влиянием возникшей потребности. Регламентируются хаотично выхваченные куски цепочек работ (см. рис. 13.2);
- управление процессами (деятельность по управлению) в регламентах либо не рассматривается совсем, либо о нем упоминается поверхностно;
- о входах и выходах в регламентах либо не говорится, либо они перечислены не полностью;
- одна и та же работа в разных регламентах может быть описана по-разному, в зависимости от точки зрения автора.

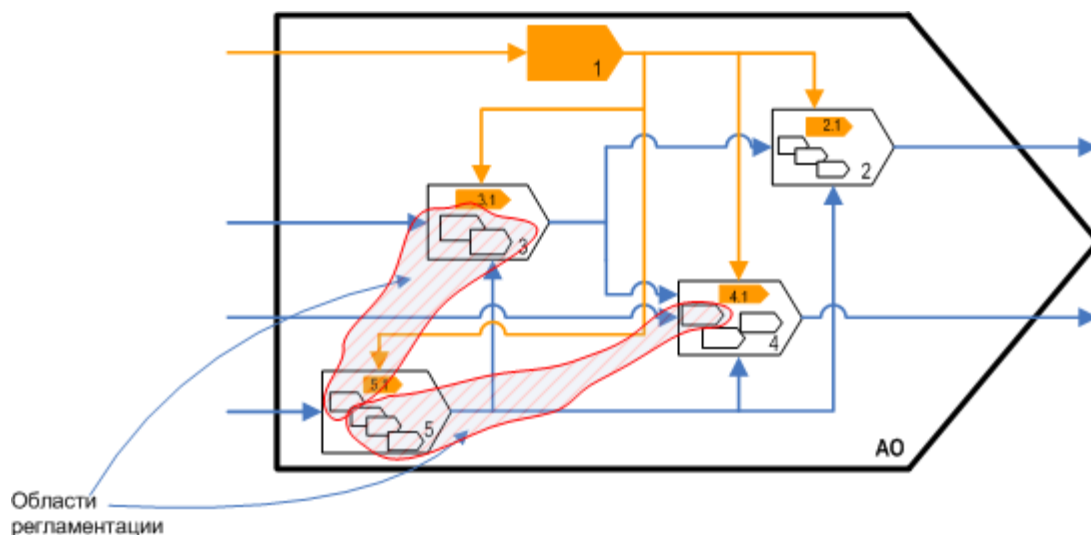


Рис.2 Области регламентации в случае отсутствия системы-бизнес-процессов

С такими регламентами бизнес-процессов невозможно работать ни с точки зрения улучшения, ни с точки зрения организации управления. Формальный подход к выделению и регламентации процессов приводит к созданию массы не работающих на практике, противоречащих друг другу, не нужных работникам документов.

13.2.4 Типовая структура регламента бизнеса-процесса

При регламентации бизнес-процесса в рамках внедрения процессного подхода обязательно следует учитывать основные элементы бизнес-процесса, то есть в регламенте должны быть отражены следующие моменты:

- деятельность по управлению процессом, в т.ч. владелец процесса и его полномочия, технология управления процессом (как минимум - планирование и отчетность), система показателей для управления процессом;
- входы процесса, в том числе требования к входам; события, инициирующие начало процесса;
- выходы процесса, в том числе требования к выходам (результатам); события, завершающие процесс;
- деятельность по преобразованию входов в выходы (технология выполнения процесса), т.е. ответственность персонала и выполняемые операции (структура и взаимосвязь операций);
- ресурсы, необходимые для выполнения процесса (персонал, инфраструктура, оборудование, информация, прочее).

Под регламентацией бизнес-процессов (процессов) верхних уровней понимается разработка, согласование, утверждение нормативно-методических документов, частично или полностью устанавливающих:

1. Порядок управления бизнес-процессом;
2. Порядок выполнения бизнес-процесса (входы, технология, выходы);
3. Требования к ресурсам, необходимым для его выполнения.

Под регламентацией бизнес-процессов операционного уровня понимается разработка, согласование и утверждение нормативно-методических документов, регламентирующих деятельность конкретных сотрудников на уровне выполняемых ими операций.

13.2.5 Особенности регламентации бизнес-процессов верхнего уровня

Что нужно описывать в регламентах процессов верхнего уровня (т.е. для крупных, масштабных процессов)? При регламентации бизнес-процессов верхнего уровня описывается деятельность руководителей по управлению процессом:

- Ответственность и полномочия руководителей;
- Технология управления бизнес-процессом (планирование, организация, контроль, мотивация, принятие решений по процессу);
- Систему показателей, используемых для управления процессом и отчетности перед вышестоящим руководством.

Управление невозможно без системы показателей, по которой мы оцениваем процессы. Для управления процессами не обязательно детально описывать технологии их выполнения. Технология выполнения бизнес-процесса верхнего уровня показывается в виде крупных блоков деятельности. Основная задача в данном случае состоит в том, чтобы отобразить структуру подпроцессов и распределение ответственности за них между руководителями более низкого уровня.

Важно:

- Создать систему процессов;
- Закрепить ответственность руководителя за их выполнение;
- Создать механизмы управления (включая систему показателей).

13.2.6 Особенности регламентации бизнес-процессов операционного уровня

При переходе от бизнес-процессов верхнего уровня к процессам операционного уровня акценты в части регламентации смещаются с регламентации управления на регламентацию технологии выполнения бизнес-процесса.

13.2.7 Содержание регламента бизнес-процессов операционного уровня

Регламентация операционной деятельности может быть выполнена, например, в виде:

- Операционной карты;
- Должностной инструкции
- Рабочей инструкции.

13.3 Структура регламента бизнес-процесса

Операционная карта процесса _____ (шаблон)

1. Наименование процесса вышестоящего уровня:

2. Ответственный за выполнение процесса:

3. Участники процесса:

4. Нормативные документы по процессу:

4.1

4.2

4.3

5. Плановые документы по процессу:

5.1

5.2

6. Отчетные документы по процессу

6.1

6.2

7. Показатели процесса

№	Наименование, ед.изм	Период контроля	Формула расчета

8. Границы процесса

Входы процесса		Выходы процесса	
№	Название	№	Название

9. Требования к ресурсам:

9.1

9.2

9.3

10. Длительность выполнения процесса:

11. Периодичность выполнения процесса:

12. Графическая модель процесса:



13.4 Учет индивидуальных особенностей организации при регламентации

При выполнении проекта регламентации бизнес-процессов в конкретной компании приведенная выше структура регламента может быть изменена как в сторону усложнения, так и в сторону упрощения.

К примеру, если некоторые процессы в организации вообще не регламентированы, но выполняются эффективно (с точки зрения руководителей компании), то не обязательно подробно описывать технологию их выполнения. Однако, описание технологии может потребоваться, если руководство компании собирается тиражировать опыт, открывая новые компании в других регионах и т.д.

Состав подпроцессов (операций) процесса может быть описан простым перечнем или таблицей без указания связей между подпроцессами. В некоторых случаях это вполне приемлемо. Для других процессов нужно не только указать состав подпроцессов (операций), но и связи между ними, например, в виде графической схемы взаимодействия.

13.5 Требования к отчету

- Титульный лист
- Введение
- Наименование процесса вышестоящего уровня:
- Ответственный за выполнение процесса:
- Участники процесса:
- Нормативные документы по процессу:
- Плановые документы по процессу:
- Отчетные документы по процессу
- Показатели процесса
- Границы процесса
- Требования к ресурсам
- Длительность выполнения процесса
- Периодичность выполнения процесса:
- Графическая модель процесса
- Заключение

Заключение

В практикуме представлен материал по жизненному циклу информационных систем, по основным моделям ЖЦИС, по этапам, а также по проектному управлению, по особенностям ИТ-проектов. Дана краткая характеристика системы проектного управления, принципы организации управления проектом, программой. Приведена информация о российских и международных стандартах и корпоративных методиках проектного управления.

Приведенные стандарты и корпоративные методики дают общее представление, каким путем можно прийти к запланированному результату, и опираясь на эти стандарты разработать свой план реализации.

Определены и разобраны в материалах лабораторных работ этапы жизненного цикла.

1. Планирование проекта.
 - Экспресс-обследование.
 - Технико-экономическое обоснование.
 - Оценка целесообразности проекта (Telos).
 - Выбор программного решения.
2. Анализ и постановка задачи.
 - Информационное обследование предприятия.
 - Описание бизнес-процессов.
 - Сбор требований.
 - Подготовка технического задания.
3. Проектирование.
 - Техническое проектирование.
 - Рабочее проектирование / Прототипирование при заказной разработке.
4. Разработка.
 - Закупка ПО.
 - Настройка конфигураций.
 - Создание ролей пользователей.
 - Миграция данных.
 - Разработка контрольного примера.
 - Тестовая эксплуатация.
 - Доработка по результатам тестирования.
 - Прием результатов испытаний.
5. Развертывание и внедрение.
 - Закупка и настройка требуемой ИТ-Инфраструктуры.
 - Ввод начальных остатков.
 - Обучение пользователей.
 - Развертывание системы на рабочих местах.
 - Основные виды тестирования.
 - Опытно-промышленная эксплуатация.
 - Приемо-сдаточные испытания.
6. Эксплуатация.
7. Сопровождение эксплуатации.
 - Авторский надзор.
 - Техническая поддержка.

- Постгарантийное сопровождение.
 - Обновление и релизы.
 - Увеличение производительности системы.
8. Модернизация.
- Стратегия управления LEGACY-Системами.
 - Виртуализация как стратегия модернизации решений.
 - Особенности проектов по модернизации.
9. Утилизация.

Успех работы над проектом зависит от точности определенной цели работы, от согласованной работы проектных команд как от заказчика, так и от исполнителя (подрядчика). В приложениях приведены образцы документы, которые используются на определенных фазах жизненного цикла информационных систем для организации процесса управления работой таких команд. Приведенные формы документов не претендуют на полноту. Они являются образцами документов. Каждая проектная команда может разрабатывать свои проектные документы, состав и полнота которых определяется целями и задачами проекта.

Список литературы

1. ГОСТ 34.601-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания.
2. ISO/IEC 12207:2008 Information technology – Software life cycle processes (Информационные технологии. Процессы жизненного цикла программного обеспечения).
3. ГОСТ Р ИСО/МЭК 12207-2010 «Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств».
4. ISO/IEC 15288 Systems engineering. System life cycle processes (Системотехника. Процессы жизненного цикла системы).
5. ГОСТ Р ИСО/МЭК 15288-2005 Информационная технология. Системная инженерия. Процессы жизненного цикла систем.
6. ISO/IEC 19510:2013 Информационные технологии. Модель и нотация процесса менеджмента объекта в групповом бизнесе.
7. ISO/IEC 19505:2012. Информационные технологии. Унифицированный язык моделирования группы по управлению объектами (OMG UML).
8. ГОСТ 19.201-78 Техническое задание, требования к содержанию и оформлению.
9. ГОСТ 34.602-89 Техническое задание на создание автоматизированной системы.
10. ГОСТ 34.201-89 Виды, комплектность и обозначение документов при создании автоматизированных систем.
11. ГОСТ 12119 ИТ. Пакеты программ. Требования к качеству и тестирование.
12. ГОСТ 12207 ИТ. Процессы жизненного цикла программных средств.
13. ГОСТ 14764 ИТ. Сопровождение программных средств.
14. ГОСТ 15504 ИТ. Оценка процессов.
15. ГОСТ 9126 ИТ. Оценка программной продукции. Характеристики качества и руководства по их применению.
16. ГОСТ серии 19 Единая система программной документации (ЕСПД).
17. IEEE 829 Стандарт документирования тестирования программного обеспечения и систем.
18. IEEE 1219 Standard for Software Maintenance.
19. CobiT (Control Objectives for Information and Related Technology).
20. ITIL (IT Infrastructure Library).
21. SWEBOOK (Guide to the Software Engineering Body of Knowledge).
22. PMBOK (Project Management Body of Knowledge).
23. PRINCE2 (PRojects IN Controlled Environments).
24. ISO 21500:2012 Guidance on project management.
25. ГОСТ Р ИСО 21500:2012 Международный стандарт по Управлению Проектами.
26. ГОСТ Р 54869-2011 Проектный менеджмент. Требования к управлению проектом.
27. ГОСТ Р 54871-2011 Требования к управлению программой.
28. ГОСТ Р 54870-2011 Требования к управлению портфелем проектов.
29. ISO 10006:2003 Quality management systems – Guidelines for quality management in projects.
30. ГОСТ Р ИСО 10006-2005 Системы менеджмента качества. Руководство по менеджменту качества при проектировании.
31. ISO 8402:94 Управление качеством и обеспечение качества.
32. ISO 9000:2005 Системы менеджмента качества. Основные положения и словарь.
33. ISO 9001:2008 Системы менеджмента качества. Требования.
34. ISO 9004:2009 Менеджмент с целью достижения устойчивого успеха организации. Подход с позиции менеджмента качества.

35. ISO 10001:2007 Менеджмент качества. Удовлетворенность потребителя. Руководящие указания по кодексу поведения для организаций.
36. ISO 10005:2005 Системы менеджмента качества. Руководящие указания по планам качества.
37. ISO 10006:2005 Системы менеджмента качества. Руководящие указания по менеджменту качества проектов.
38. ISO 10007:2003 Системы менеджмента качества. Руководящие указания по менеджменту конфигурации.
39. ISO/TR 10013:2001 Рекомендации по документированию систем менеджмента качества.
40. ISO/TR 10014:2006 Менеджмент качества. Руководящие указания по реализации финансовых и экономических выгод.
41. ISO 10015:1999 Управление качеством. Руководящие указания по обучению.
42. ISO 19011:2011 Руководящие указания по аудиту систем менеджмента.
43. ISO 10006:2003 Управление качеством. Руководящие указания по менеджменту качества проектов.
44. ГОСТ Р ИСО 10006-2005 Системы менеджмента качества. Руководство по менеджменту качества при проектировании.
45. ISO 15504:2004 Information Technology. Process Assessment / SPICE (Software Process Improvement and Capability Determination).
46. ГОСТ Р ИСО/МЭК 15504 Информационные технологии. Оценка процессов.
47. Contingency Planning Guide for Information Technology Systems (NIST).
48. Ананьин В., Зимин К. Ценность ИТ. Теория и практика. СIO congress Kirov. 28.04.2012. Режим доступа от 18.02.2013. [<http://bit.ly/139QkWJ>].
49. Блинов Д. Требования к системе: классификация FURPS+. Портал BeamTeam.ru, 2010г. [<http://beamteam.ru/2010/09/furps/>].
50. <http://www.novsu.ru/file/977849>.
51. <http://www.swell-services.be/cmsms/index.php?page=it-system-decommissioning>.
1. 52. http://2.bp.blogspot.com/-6w5M_Pf3Gq8/TskqA8pxBFI/AAAAAAAAAF9Y/YQPTqN6wy9A/s1600/image00.png
52. http://www.cnews.ru/promo/sap/forum/usp_project/Serikov.pdf.
53. http://www.intuit.ru/studies/courses/2196/267/print_lecture/6796.
54. <http://www.edu.dvgups.ru>.
55. http://tsconsulting.ru/upload/iblock/37b/37b27c616e326793c95eb98b325_a0748.pdf.
56. http://www.pqm-online.com/assets/files/standards/gost_r_iso_10006-2005.pdf.