

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Колледж экономики, управления и права

**Методические указания**

**по организации практических занятий и самостоятельной работы**

**по МДК.05.01 Проектирование и дизайн информационных систем**

**Специальность**

*09.02.07 Информационные системы и программирование*

*Квалификация Разработчик веб и мультимедийных приложений*

Ростов-на-Дону

2022

Методические указания по **МДК.05.01 Проектирование и дизайн информационных систем** разработаны с учетом ФГОС среднего профессионального образования специальности 09.02.07 Информационные системы и программирование, предназначены для студентов и преподавателей колледжа.

Методические указания определяют этапы выполнения работы на практическом занятии, содержат рекомендации по выполнению индивидуальных заданий и образцы решения задач, а также список рекомендуемой литературы.

Составитель (автор): Д.А. Косыченко преподаватель колледжа ЭУП

Рассмотрены на заседании предметной (цикловой) комиссии специальностей 09.02.04 Информационные системы (по отраслям), 09.02.05 Прикладная информатика (по отраслям) и 09.02.07 Информационные системы и программирование

Протокол № 9 от 30 июня 2022 г.

Председатель П(Ц)К \_\_\_\_\_\_\_\_\_\_\_ С.В. Шинакова

личная подпись

и одобрены решением учебно-методического совета колледжа.

Протокол № 7 от 30 июня 2022 г.

Председатель учебно-методического совета колледжа

\_\_\_\_\_\_\_\_\_\_\_ С.В. Шинакова

личная подпись

Рекомендованы к практическому применению в образовательном процессе.

Рецензенты:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

место работы должность ФИО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

место работы должность ФИО

**Содержание**

[Правила выполнения практических работ 4](#_Toc106898190)

[Практическая работа №1. Анализ предметной области различными методами 5](#_Toc106898191)

[Практическая работа №2. Изучение устройств автоматизированного сбора информации 8](#_Toc106898192)

[Практическая работа №3. Разработка модели архитектуры информационной системы 11](#_Toc106898193)

[Практическая работа №4. Обоснование выбора средств проектирования информационной системы 14](#_Toc106898194)

[Практическая работа №5. Построение организационных диаграмм в MS Visio 17](#_Toc106898195)

[Практическая работа №6. Построение организационных диаграмм в Elma Community Edition 30](#_Toc106898196)

[Практическая работа №7. ER-моделирование в нотации Чена 46](#_Toc106898197)

[Практическая работа №8. ER-моделирование в нотации IDEF1X 53](#_Toc106898198)

[Практическая работа №9. Функциональное моделирование IDEF0 59](#_Toc106898199)

[Практическая работа №10. Моделирование потоков данных DFD 68](#_Toc106898200)

[Практическая работа №11. Описание бизнес-процессов заданной предметной области 76](#_Toc106898201)

[Практическая работа №12. Моделирование и запуск бизнес-процесса в Elma Community Edition 84](#_Toc106898202)

[Практическая работа №13. Построение диаграммы Классов 103](#_Toc106898203)

[Практическая работа №14. Построение диаграммы Состояний 113](#_Toc106898204)

[Практическая работа №15. Построение диаграммы Деятельности 122](#_Toc106898205)

[Практическая работа №16. Построение диаграммы Последовательности 131](#_Toc106898206)

[Практическая работа №17. Оценка экономической эффективности информационной системы 135](#_Toc106898207)

[Практическая работа №18. Построение модели управления качеством 141](#_Toc106898208)

[Практическая работа №19. Реинжиниринг методом интеграции 143](#_Toc106898209)

[Практическая работа №20. Реинжиниринг бизнес-процессов методом горизонтального и/или вертикального сжатия 147](#_Toc106898210)

[Практическая работа №21. Разработка требований безопасности информационной системы 150](#_Toc106898211)

[Практическая работа №22. Проектирование спецификации информационной системы 157](#_Toc106898212)

[Практическая работа №23. Составление технического задания 159](#_Toc106898213)

[Практическая работа №24. Составление технического проекта 161](#_Toc106898214)

[Практическая работа №25. Составление руководства системного программиста 163](#_Toc106898215)

[Практическая работа №26. Разработка общего функционального описания программного средства по индивидуальному заданию 165](#_Toc106898216)

[Практическая работа №27. Разработка руководства по инсталляции программного средства по индивидуальному заданию 167](#_Toc106898217)

[Практическая работа №28. Разработка руководства пользователя программного средства 168](#_Toc106898218)

[Практическая работа №29. Изучение средств автоматизированного документирования 170](#_Toc106898219)

# Правила выполнения практических работ

Практические работы выполняются каждым обучающимся самостоятельно в полном объеме и согласно содержанию методических указаний.

Перед выполнением обучающийся должен отчитаться перед преподавателем за выполнение предыдущего занятия (сдать отчет).

Обучающийся должен на уровне понимания и воспроизведения предварительно усвоить необходимую для выполнения практических работ теоретическую и информацию.

Обучающийся, получивший положительную оценку и сдавший отчет по предыдущей практической работе, допускается к выполнению следующей работы.

Обучающийся, пропустивший практическое занятие по уважительной либо неуважительной причине, закрывает задолженность в процессе выполнения последующих практических работ.

**5 семестр**

# Практическая работа №1. Анализ предметной области различными методами

**Цель:** ознакомиться с процессом анализа предметной области и получить навыки по использованию методов анализа предметной области.

**Краткие теоретические сведения**

Главная цель объектного анализа - представить предметную область как множество объектов со свойствами и характеристиками, которые достаточны для их определения и идентификации, а также для задания поведения объектов в рамках выбранной системы понятий и абстракций. На произвольном шаге объектного анализа все понятия (сущности) ПрО - суть объекты. Каждый объект - это уникальный элемент, имеет, по крайней мере, одно свойство или характеристику и уникальную идентификацию во множестве объектов.

Предметная область сама является самостоятельным объектом или может быть объектом в составе другой предметной области.

Анализ ПрО проводится с помощью объектно-ориентированных методов и соответствующих стандартов. Конечная цель объектно-ориентированного анализа ПрО - определение объектной модели (ОМ) с помощью выделенных объектов, отношений между ними и их свойствами и характеристиками.

При построении модели ОМ в предметной области также выявляются функциональные задачи, формулируются требования к их проектированию и реализации. Требования, задачи и модель ОМ - необходимые условия построения архитектуры системы для анализируемой ПрО.

На данный момент известно более пятидесяти объектно-ориентированных методов анализа ПрО, которые прошли проверку практикой. Приведем некоторые основные из них:

метод объектно-ориентированного системного анализа OOAS (Object-Oriented system analysis), позволяющий выделить сущности и объекты ПрО, определить их свойства и отношения, а также построить на их основе информационную модель, модель состояний объектов и процессов представления потоков данных (dataflow) ;

метод объектно-ориентированного анализа OOA (Object-Oriented analysis), обеспечивающий моделирование ОМ и формирование требований к ПрО с помощью понятия "сущность-связь" (entity-relationship ER), спецификацию потоков данных и соответствующих процессов;

метод SD (Structured Design) структурного проектирования системы, данных и программ преобразования входных данных в выходные с помощью структурных карт Джексона;

методология объектно-ориентированного анализа и проектирования OOAD (Object-oriented analysis and design), которая основывается на ER-моделировании сущностей и отношений в объектной модели ПрО, она обеспечивает определение системы и организацию данных с использованием структурных диаграмм, диаграмм "сущность-связь" и матрицы информационного управления;

технология объектного моделирования OMT (Object Modeling Technique) включает в себя процессы (анализа, проектирования и реализации), набор нотаций для задания четырех моделей (объектной, динамической, функциональной и взаимодействия);

объединенный метод UML, включающий средства и понятия метода Г. Буча (объекты, классы, суперклассы), принципы наследования, полиморфизма и упрятывания информации об объектах, а также варианты использования метода Джекобсона для задания сценариев работы системы при выполнении задач ПрО и диаграммные средства взаимодействия объектов Румбауха;

метод определения распределенных объектов на основе объектной модели CORBA и набора сервисных системных компонентов общего пользования, обеспечивающих их функционирование в среде распределенных приложений;

метод генерации (generative) частей системы из семейства ПрО с помощью готовых объектов, аспектов, компонентов, программ многоразового использования и приложений, а также модели характеристик, в которой представлены функциональные и нефункциональные требования к семейству систем.

Наиболее используемая объектная модель ПрО реализована в системе CORBA. Каждый объект модели инкапсулирует некоторую сущность ПрО и определяет один или несколько сервисов (методов) ее реализации. Объекту соответствует одна или несколько операций обращения к методам. Объекты группируются в типы, а их экземпляры - в подтипы/супертипы.

Они инкапсулируют методы реализации, которые невидимы во внешнем интерфейсе, т.е. ОМ не содержит информации о способах реализации типа, а только о наличии его реализации. Во внешнем интерфейсе содержатся операции, которые вызывают методы объектов для их выполнения. Специализация типа определяется постепенно на этапах стратегии, анализа, проектирования и реализации объекта. Взаимодействие объектов осуществляет брокер объектных запросов и операций (Особенности системы CORBA изложены в "Интерфейсы, взаимодействие и изменение программ и данных" ).

Приведенная общая характеристика разновидностей объектно-ориентированных методов показывает, что они имеют много общих черт (например, ER-моделирование, Dataflow), а также свои специфические особенности. Каждый разработчик метода объектно-ориентированного анализа вводил необходимые новые понятия, которые зачастую семантически совпадали с аналогичными понятиями в других методах. Поэтому у авторов UML возникла идея объединить свои индивидуальные методы объектного анализа (Буча, Джекобсона и Рамбауха) для создания единого метода объектного моделирования UML.

**Методика выполнения**

**Задание № 1**

Ознакомиться с предложенным вариантом описания предметной области (согласно заданию индивидуального варианта).

**Задание № 2**

Проанализировать предметную область, уточнив и дополнив ее, руководствуясь собственным опытом, консультациями и любыми источниками (книгами, учебниками или интернет- источниками).

**Задание № 3**

Выполнить структурное разбиение предметной области на отдельные подразделения (подсистемы) согласно выполняемым ими функциям.

**Задание № 4**

Определить задачи и функции системы в целом и функции каждого подразделения (подсистемы).

**Задание № 5**

Продумать подробное описание работы каждого подразделения (подсистемы), алгоритмов и сценариев выполнения ими отдельных работ. Продумать виды входной и выходной информации для каждого подразделения (подсистемы).

**Задание № 6**

Описать схему работы будущей информационной системы, учитывая выделенные и описанные ранее подсистемы.

**Задание № 7**

Определить группу пользователей, для которой данная система будет более востребована.

Описать перечень функций системы, которые будут доступны данной группе пользователей.

**Задание № 8**

Расписать основные функциональные возможности администратора системы, как одного из пользователей системы.

**Варианты**

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №2. Изучение устройств автоматизированного сбора информации

**Цель работы:** изучение устройств автоматизированного сбора информации

**Краткие теоретические сведения**

Автоматизированные системы сбора данных в настоящее время являются общедоступным средством получения экспериментальной информации и связано это, в первую очередь, с широким распространением персональных компьютеров. Системы сбора данных находят применение для научных исследований, управления производственными процессами, мониторинга в промышленности, медицине, метеорологии, космонавтике и других областях человеческой деятельности. Автоматизированный сбор данных позволяет получить данные нового качества, которые невозможно получить иными средствами. Это результаты статистической обработки огромного числа измерений, полученных в цифровой форме, возможность регистрации случайно появляющихся событий с недостижимой ранее разрешающей способностью по времени и амплитуде, регистрация быстроизменяющихся процессов.

Основным отличием систем сбора данных от ПЛК является отсутствие в них алгоритма управления, т.е. отсутствие необходимости в мощном контроллере и языке МЭК 61131-3, а также наличие большого объема памяти для ведения архива. Хотя системы сбора данных можно построить на любом ПЛК, но в связи с указанными выше особенностями они занимают отдельный сегмент рынка и их выделяют в отдельную группу средств автоматизации.

Системы сбора данных могут применяться в реальном времени, например, для мониторинга (наблюдения) различных процессов, идентификации аварийных ситуаций в технологических процессах, а также могут применяться для архивирования данных, когда их обработка отделена от процесса сбора неопределенным интервалом времени. В системах реального времени текущие данные сохраняются в течение некоторого заданного времени в кольцевом буфере, откуда устаревшие данные вытесняются вновь поступившими. В архивирующих системах используются накопители информации большой емкости и данные обрабатываются после завершения сбора.

Архивирующие системы сбора данных (логгеры, самописцы) могут быть автономными устройствами, построенными на основе микроконтроллера (например, бортовые самописцы самолетов, электронные счетчики тепла или электроэнергии, портативные электрокардиографы). Данные, собранные логгерами, для обработки переносятся в компьютер с помощью, например, USB флэш-памяти или через последовательный интерфейс.

Системы сбора данных, построенные на основе компьютера, обычно являются стационарными и используют универсальное программное обеспечение, такое как Matlab, LabView, MS Excel, которое позволяет не только собрать данные, но и обработать их.

Для регистрации быстропротекающих процессов (с требуемой частотой отсчетов более 1 МГц) используются системы с параллельной шиной, в том числе платы для шины PCI компьютера. Компьютерные платы имеют ограниченное количество входов, что определяется компьютерным конструктивом, и требуют внешних клеммных блоков для подсоединения источников сигнала, создавая неудобства при монтаже системы.

Для регистрации медленных процессов удобнее внешние устройства, подключаемые к компьютеру через СОМ, USB или Ethernet порт. Внешние устройства отличаются также меньшим уровнем шумов, в то время как платы, вставляемых в компьютер, подвержены влиянию наводок от цифровых цепей компьютера.

Система сбора данных может быть распределенной, когда устройства ввода распределены территориально по объекту сбора данных, а полученные данные сходятся к единому накопителю и обработчику данных с помощью сетевых технологий. Сетевые (распределенные) системы сбора данных имеют свойство практически неограниченной наращиваемости числа каналов, однако имеют ограничение на скорость передачи данных по сети.

Входы систем сбора данных могут быть универсальными (потенциальными и токовыми), или специализированными (например, для термопар, для термопреобразователей сопротивления или для тензодатчиков). Системы со специализированными входами экономически более эффективны для потребителя. Универсальные входы используются совместно с измерительными преобразователями физических величин в ток или напряжение. Существуют также системы с гибридными входами, например, когда несколько входов принимают сигналы термопар, другие входы - сигналы тензодатчиков, третьи - сигналы термометров сопротивления и т. д.

ADVERTISEMENT

Входы могут быть дифференциальными, одиночными, цифровыми или дискретными (двоичными). Дифференциальные входы позволяют более эффективно подавлять внешние помехи, наводимые на кабель, передающий сигнал от датчика к модулю ввода [Денисенко]. Для передачи сигнала чаще всего используется напряжение в диапазоне 0...±5 В., 0...±10 В или ток 0...20 мА, 4...20 мА. Сигналы напряжения вырабатываются источниками напряжения и имеют высокую помехоустойчивость к емкостным наводкам, сигналы тока вырабатываются источниками тока и устойчивы к индуктивным наводкам [Денисенко]. Дискретные входы принимают логические сигналы ("0" или "1"), которые поступают от концевых выключателей, датчиков охранной или пожарной сигнализации, электромагнитных реле, датчиков наличия напряжения и т. п. Цифровые входы принимают сигналы от устройств с цифровым выходом, например, от цифровых датчиков температуры.

Основными параметрами систем сбора данных являются количество каналов, погрешность, динамическая погрешность, время установления или полоса пропускания, разрешающая способность, эффективное число разрядов, частота дискретизации, наличие гальванической изоляции входов и интерфейса, наличие защит от небрежного использования, перегрузок и перегрева.

Системы сбора данных обычно имеют 4, 8, 16, 32, 64 ... входа, которые опрашиваются по очереди или одновременно. Системы с одновременным опросом состоят из идентичных каналов, которые выполняют аналого-цифровое преобразование входной величины параллельно, т.е одновременно для всех каналов. Такие системы встречаются редко по причине высокой стоимости. Обычно опрос входов выполняется по очереди, с помощью коммутатора. Поэтому данные разных каналов оказываются сдвинутыми по времени на некоторую задержку, равную отношению периода опроса к количеству каналов.

Примером системы сбора данных может служить серия систем сбора данных RealLab! [Денисенко] построенная по модульному принципу, т.е. систему с необходимым количеством входов можно собрать из модулей - отдельных строительных блоков. Модули соединяются между собой с помощью промышленного интерфейса RS-485 и располагаются либо в общем монтажном шкафу, либо распределены по объекту сбора данных таким образом, чтобы уменьшить длину кабеля от датчика к модулю. Собранные данные в цифровой форме передаются по промышленной сети в центральный компьютер или контроллер. Модули RealLab! могут работать в стандартных сетях Modbus RTU или в стандартной де-факто сети DCON, имеют открытый протокол обмена. Каждый модуль в сети имеет свой адрес, поэтому для опроса модулей компьютер посылает им команду, содержащую адрес и код операции, которую необходимо выполнить.

**Задание № 1**

Изучить и описать технологии штрихового кодирования (Bar Code Technologies) сбора информации.

**Задание № 2**

Изучить и описать технологии радиочастотной идентификации (RFID – Radio Frequency Identification Technologies) сбора информации.

**Задание № 3**

Изучить и описать карточные технологии (Card Technologies) сбора информации.

**Задание № 4**

Изучить и описать технологии сбора данных (Data Communications Technologies).

**Задание № 5**

Изучить и описать технологии распознавания голоса, оптического и магнитного распознавания текста, биометрические технологии и некоторые другие.

**Задание № 6**

В зависимости от целей, сферы деятельности и располагаемых технических средств можно выделить методы сбора данных, применяемые:

1. в экономических информационных системах (например, маркетинга);
2. в геоинформационных системах;
3. в статистических информационных системах;
4. в информационных системах управления производственными процессами.

**Задание № 7**

Для заданной предметной области опишите устройства и методы автоматизированного сбора информации.

**Варианты**

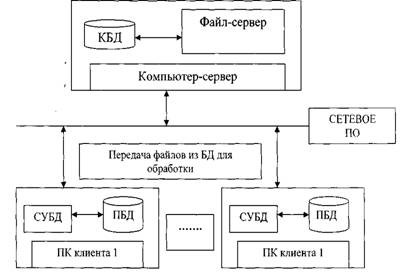
1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №3. Разработка модели архитектуры информационной системы

**Цель работы:** получение навыков разработки модели архитектуры информационной системы

**Краткие теоретические сведения**

Эффективность функционирования информационной системы во многом зависит от ее архитектуры. Самой простой архитектурой для реализации является архитектура "файл-сервер", но она же обладает и самым большим количеством недостатков, ограничивающих спектр решаемых ею задач. Простейшим случаем является случай, когда данные располагаются физически на том же компьютере, что и само приложение.



К существенным неудобствам, возникающим при работе с системой, построенной по такой архитектуре, можно отнести следующее: трудности при обеспечении непротиворечивости и целостности данных; существенная загрузка локальной сети передаваемыми данными; в целом, невысокая скорость обработки и представления информации; высокие требования к ресурсам компьютеров. При этом возникают следующие ограничения. невозможность организации равноправного одновременного доступа; пользователей к одному и тому же участку базы данных; количество одновременно работающих с системой пользователей не превышает пяти человек для ЛВС, построенной в соответствии со спецификацией 1 OBaseT (скорость обмена данными до 10Мб/с); При всем этом система обладает одним очень важным преимуществом - низкой стоимостью. Архитектура "файл-сервер" предусматривает концентрацию обработки на рабочих станциях. Основным преимуществом этого варианта является простота и относительная дешевизна. Подобное решение приемлемо, пока число пользователей, одновременно работающих с базой данных, не превышает 5-10 человек. При увеличении количества пользователей система может "захлебнуться" из-за перегруженности ЛВС большими потоками необработанной информации. Сервер, как правило, - самый мощный и самый надежный компьютер. Он обязательно подключается через источник бесперебойного питания, в нем предусматриваются системы двойного или даже тройного дублирования. В особо ответственных случаях можно подключить вместе несколько серверов так, что при выходе из строя одного из них в работу автоматически включится "дублер". Таким образом, при концентрации обработки данных на сервере надежность системы в целом ограничивается только материальными средствами, которые заказчики готовы вложить в техническое оснащение. Решение по автоматизации учета и управления в корпоративных структурах предполагает распределенную обработку данных, организацию параллельных вычислений, глубокое разграничение уровней доступа, возможность выбора различных операционных систем и серверных платформ. Если бизнес не велик, подобное решение оптимально. В ходе эксплуатации были выявлены общие недостатки файл-серверного подхода при обеспечении многопользовательского доступа к базе данных. Вся тяжесть вычислительной нагрузки при доступе к базе данных ложится на приложение клиента, что является следствием принципа обработки информации в системах "файл-сервер": при выдаче запроса на выборку информации из таблицы вся таблица базы данных копируется на клиентское место, и выборка осуществляется на клиентском месте. Локальные СУБД используют так называемый "навигационный подход", ориентированный на работу с отдельными записями. Не оптимально расходуются ресурсы клиентского компьютера и сети; например, если в результате запроса мы должны получить 2 записи из таблицы объемом 10000 записей, все 10000 записей будут скопированы с файл-сервера на клиентский компьютер; в результате возрастает сетевой трафик и увеличиваются требования к аппаратным мощностям пользовательского компьютера. В базе данных на файл-сервере гораздо проще вносить изменения в отдельные таблицы, минуя приложения. Эта возможность облегчается тем обстоятельством, что у локальных СУБД база данных - понятие более логическое, чем физическое, поскольку под базой данных понимается набор отдельных таблиц, сосуществующих в едином каталоге на диске. Все это позволяет говорить о низком уровне безопасности - как с точки зрения хищения и нанесения вреда, так и с точки зрения внесения ошибочных изменений. Недостаточно развитый аппарат транзакций для локальных СУБД служит потенциальным источником ошибок как с точки зрения одновременного внесения изменений в одну и ту же запись, так и с точки зрения отката результатов серий объединенных по смыслу в единое целое операций над базой, когда некоторые из них завершились неуспешно, а некоторые - нет; это может нарушать ссылочную и смысловую целостность базы данных. Недостатки настольных СУБД обычно проявляются не сразу, а лишь в процессе длительной эксплуатации, когда объем хранимых данных и число пользователей становятся достаточно велики - это приводит к снижению производительности приложений, использующих такие СУБД. Поскольку настольные СУБД не содержат специальных приложений и сервисов, управляющих данными, а используются для этой цели файловые сервисы операционной системы, вся реальная обработка данных в таких СУБД осуществляется в клиентском приложении, и любые библиотеки доступа к данным в этом случае также находятся в адресном пространстве клиентского приложения. Поэтому при выполнении запросов данные, на основании которых выполняется такой запрос, должны быть доставлены в то же самое адресное пространство клиентского приложения. Это и приводит к перегрузке сети при увеличении числа пользователей и объема данных, а также грозит иными неприятными последствиями, например разрушением индексов и таблиц. Недаром до сих пор популярны утилиты для "ремонта" испорченных файлов настольных СУБД. Недостатки архитектуры "файл-сервер" решаются при переводе приложений в архитектуру "клиент-сервер", которая знаменует собой следующий этап в развитии СУБД. Характерной особенностью архитектуры "клиент-сервер" является перенос вычислительной нагрузки на сервер базы данных (SQL-сервер) и максимальная разгрузка приложения клиента от вычислительной работы, а также существенное укрепление безопасности данных - как от злонамеренных, так и просто ошибочных изменений. БД в этом случае помещается на сетевом сервере, как и в архитектуре "файл-сервер", однако прямого доступа к базе данных (БД) из приложений не происходит. Функция прямого обращения к БД осуществляет специальная управляющая программа - сервер БД (SQL-сервер), поставляемый разработчиком СУБД.

**Задание № 1**

Спроектировать информационную систему на основе архитектуры «файл-сервер».

**Задание № 2**

Спроектировать информационную систему на основе архитектуры «клиент-сервер».

**Задание № 3**

Спроектировать информационную систему на основе многозвенной архитектуры «клиент-сервер».

**Варианты**

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №4. Обоснование выбора средств проектирования информационной системы

**Цель**: изучить основные характеристики СУБД и средств разработки приложений, провести их сравнительный анализ.

**Теоретические сведения**

Microsoft Office Access — реляционная система управления базами данных (СУБД) корпорации Microsoft. Входит в состав пакета Microsoft Office. Имеет широкий спектр функций, включая связанные запросы, связь с внешними таблицами и базами данных. Благодаря встроенному языку VBA, в самой Access можно писать приложения, работающие с базами данных.

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase.

Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями.

Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

MySQL — свободная реляционная система управления базами данных[7]. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB.

Interbase — реляционная система управления базами данных, разрабатывающаяся компанией Embarcadero, появилась в середине 1980-х годов, принадлежала самостоятельной одноимённой компании, Ashton-Tate, Borland. Код Interbase стал основой свободно распространяемой системы управления базами данных Firebird.

Delphi (произноситься Дельфи) — структурированный объектно-ориентированный язык программирования с жесткой привязкой к типам данных. В основном используется для прикладного программирования в среде Windows.

C++ Builder — программный продукт, инструмент быстрой разработки приложений (RAD), интегрированная среда программирования (IDE), система, используемая программистами для разработки программного обеспечения на языках программирования Си и C++.

RAD Studio – это программная среда, позволяющая пользователям на языках Object Pascal и C++ создавать, развертывать и обновлять приложения в самый быстрый способ за счет использования облачных сервисов.

**Задание**

1. Выполнить сравнительный анализ средств управления базами данных, заполнив таблицу приведенную ниже.

**Сравнительные характеристики СУБД**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Название** | **MS Access** | **MS SQL Server** | **MySQL** | **Oracle** |
| Версия |  |  |  |  |
| Фирма производитель |  |  |  |  |
| Поддерживаемые ОС |  |  |  |  |
| Аппаратные требования |  |  |  |  |
| Поддерживаемая модель данных |  |  |  |  |
| Формат файлов БД |  |  |  |  |
| Поддерживаемые объекты БД |  |  |  |  |
| Технология создания БД |  |  |  |  |
| Создание локальной БД |  |  |  |  |
| Поддержка сервера БД |  |  |  |  |
| Встроенный язык для разработки приложений |  |  |  |  |
| Поддержка ограничений целостности БД |  |  |  |  |
| Стандарт SQL |  |  |  |  |
| Передача данных в формат MS Excel, MS Word |  |  |  |  |
| Средства для получения отчетов |  |  |  |  |
| Разграничение прав доступа |  |  |  |  |
| Резервирование и восст. БД |  |  |  |  |
| Простота/сложность работы с СУБД |  |  |  |  |
| Поддержка Windows - инструмента |  |  |  |  |
| Средства поддержки транзакций |  |  |  |  |
| Простота/сложность работы с инструментальным средством |  |  |  |  |
| Возможность создания  Запускаемого файла |  |  |  |  |

1. Выполнить сравнительный анализ средств разработки приложений, заполнив таблицу, приведенную ниже.

**Сравнительные характеристики средств разработки приложений**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Параметр** | **Инструментальная среда** | | | |
| **Delphi** | **C++ Builder** | **Visual Studio** | **RAD Studio** |
| Название, версия, фирма производитель |  |  |  |  |
| Поддерживаемые операционные системы |  |  |  |  |
| Требования к аппаратному обеспечению |  |  |  |  |
| Встроенный язык |  |  |  |  |
| Поддержка стандарта SQL |  |  |  |  |
| Поддержка ООП |  |  |  |  |
| Механизмы доступа к БД |  |  |  |  |
| Наличие визуальных и не визуальных компонент для работы с БД |  |  |  |  |
| Средства построения отчётов |  |  |  |  |
| Наличие средств построения диаграмм. |  |  |  |  |
| Поддержка технологии Rapid Application Development. |  |  |  |  |
| Средства поддержки транзакций |  |  |  |  |
| Утилиты для работы с БД |  |  |  |  |
| Возможность создания исполняемого файла |  |  |  |  |
| Сложность или простота работы с инструментальным средством |  |  |  |  |

1. Ответить на контрольные вопросы.
2. Сделать вывод о полученном результате.
3. Сформировать отчет о проделанной работе.

**Контрольные вопросы**

* 1. Средства проектирования информационной системы? Понятие.
  2. Какие средства относятся к средствам проектирования информационных систем?
  3. Характеристика современных CASE-средств и их классификация
  4. Что такое системы управления базами данных?
  5. Что такое приложение?
  6. Средства разработки приложений? Понятие, примеры.

# Практическая работа №5. Построение организационных диаграмм в MS Visio

**Цель**: изучение основ создания организационных диаграмм в ручном и автоматическом режимах.

**Задачи**

Основными задачами практической работы являются:

* ознакомиться с теоретическими вопросами построения организационных диаграмм с помощью MS Visio;
* построить оргдиаграмму стандартными средствами;
* построить оргдиаграмму с помощью мастера диаграмм.

**Теоретические сведения**

***1. Общие сведения об организационных диаграммах***

**Организационная диаграмма** – это схема иерархии отчетности, которая используется для отображения отношений между сотрудниками, должностями и группами.

Организационные диаграммы могут быть как простыми схемами, так и большими и сложными на основе сведений из внешнего источника данных. Фигуры организационной диаграммы могут отображать основные сведения, например, имя и должность, или подробные сведения, например, отдел и учетный отдел. К фигурам организационной диаграммы можно даже добавлять рисунки.

**Организационная диаграмма** – это схематическое представление об иерархии внутри компании, а также распределении полномочий и ответственности между сотрудниками и отделами.

Как показывает опыт, применение руководителями, менеджерами данных схем значительно повышает эффективность управления предприятием.

Использование организационных диаграмм позволяет решить сразу несколько задач:

1. проанализировать состояние дел в компании на текущий момент,
2. вовремя обнаружить какую-то проблему в организации и системе управления,
3. избежать появления новых проблем и ошибок.

В зависимости от поставленных целей организационные диаграммы могут быть простыми или развернутыми.

***Простые диаграммы*** содержат краткую информацию о каждом из сотрудников (имя, должность и место в системе организационной иерархии), а также раскрывают численный состав персонала.

***Развернутые диаграммы*** несут дополнительные сведения о функциях и объектах управления сотрудников компании.

Организационные диаграммы позволяют визуально представить характер взаимоотношений внутри компании, благодаря чему удается своевременно обнаружить и разрешить возникающие в организации проблемы.

Основные **преимущества организационных диаграмм:**

1. Они позволяют выявить главных игроков в организации и проанализировать их отношения с остальным персоналом.
2. Сотрудники компании получают ясное представление о том, кто возьмет на себя ответственность при разрешении тех или иных организационных проблем в соответствии с корпоративными стандартами.
3. Повышают осведомленность сотрудников о функциях и профессиональном статусе каждого субъекта, работающего в компании, тем самым способствуя совершенствованию процесса коммуникации внутри фирмы.

Следует отметить, что организационные диаграммы имеют ряд ограничений. Например, отражая лишь структуру и формальный характер взаимодействия персонала, они не затрагивают личные взаимоотношения людей в коллективе. Кроме того, с помощью этих схем нельзя определить стиль менеджмента, выбранный руководством предприятия.

Организационные диаграммы описывают связи при помощи **трех типов графических элементов:**

1. ***Линия:*** указывает на прямые отношения между руководителями и подчиненными. Для описания взаимосвязей между различными иерархическими уровнями организации линии рисуют слева или справа от диаграммы.
2. ***Ступенька:*** служит для иллюстрации отношений между помощниками менеджера, а также связей между сферами деятельности, в которых помощники могут давать советы руководителю, но при этом их мнение не является авторитетным.
3. ***Функционал:*** показывает связи между должностями специалистов и сферами деятельности, в которых мнение специалистов является авторитетным для руководителя.

***Типы организационных структур***

1. ***линейная модель:*** каждый руководитель обеспечивает руководство нижестоящими подразделениями по всем видам деятельности;
2. ***функциональная модель:*** «одно подразделение = одна функция»;
3. ***линейно-функциональная модель:*** ступенчатая иерархическая;
4. ***процессная модель:*** «одно подразделение = один процесс»;
5. ***матричная модель:*** «один процесс или один проект = группа сотрудников из разных функциональных подразделений»;
6. ***множественная (смешанная).***

В ***линейной структуре*** управления каждый руководитель обеспечивает руководство нижестоящими подразделениями по всем видам деятельности. *Достоинство* – простота, экономичность, предельное единоначалие. Основной *недостаток* – высокие требования к квалификации руководителей. Сейчас практически не используется.

***Функциональная организационная структура*** – связь административного управления с осуществлением функционального управления.

***Линейно-функциональная структура*** – линейные руководители являются единоначальниками, а им оказывают помощь функциональные органы. Линейные руководители низших ступеней административно не подчинены функциональным руководителям высших ступеней управления. Она применялась наиболее широко.

*Преимущества:*

* четкая система взаимных связей внутри функций и в соответствующих им подразделениях;
* четкая система единоначалия – один руководитель сосредотачивает в своих руках руководство всей совокупностью функций, составляющих деятельность;
* ясно выраженная ответственность;
* быстрая реакция исполнительных функциональных подразделений на прямые указания вышестоящих.

*Недостатки:*

* в работе руководителей практически всех уровней оперативные проблемы («текучка») доминируют над стратегическими;
* слабые горизонтальные связи между функциональными подразделениями порождают волокиту и перекладывание ответственности при решении проблем, требующих участия нескольких подразделений;
* малая гибкость и приспособляемость к изменению ситуации;
* критерии эффективности и качества работы подразделений и организации в целом разные и часто взаимоисключающие;
* большое число «этажей» или уровней управления между работниками, выпускающими продукцию, и лицом, принимающим решение;
* перегрузка управленцев верхнего уровня;
* повышенная зависимость результатов работы организации от квалификации, личных и деловых качеств высших управленцев.

***Процессная модель.*** Истоки концепции управления процессами ведут к теориям управления, разработанным еще в XIX веке. В 80-х годах XIX-го века Фредерик Тейлор предложил менеджерам использовать методы процессного управления для наилучшей организации деятельности. В начале 1900-х годов Анри Файоль разработал концепцию реинжиниринга – осуществление деятельности в соответствии с поставленными задачами путем получения оптимального преимущества из всех доступных ресурсов.

*Преимущества процессных структур:*

* четкая система взаимных связей внутри процессов и в соответствующих им подразделениях;
* четкая система единоначалия – один руководитель сосредотачивает в своих руках руководство всей совокупностью операций и действий, направленных на достижение поставленной цели и получение заданного результата;
* наделение сотрудников большими полномочиями и увеличение роли каждого из них в работе компании приводит к значительному повышению их отдачи;
* быстрая реакция исполнительных процессных подразделений на изменение внешних условий;
* в работе руководителей стратегические проблемы доминируют над оперативными;
* критерии эффективности и качества работы подразделений и организации в целом согласованы и сонаправлены.

*Недостатки процессной структуры:*

* повышенная зависимость результатов работы организации от квалификации, личных и деловых качеств рядовых работников и исполнителей.
* управление смешанными в функциональном смысле рабочими командами – более сложная задача, нежели управление функциональными подразделениями;
* наличие в команде нескольких человек различной функциональной квалификации неизбежно приводит к некоторым задержкам и ошибкам, возникающим при передаче работы между членами команды. Однако потери здесь значительно меньше, чем при традиционной организации работ, когда исполнители подчиняются различным подразделениям компании.

***Матричная модель.*** Матричные структуры совмещают принципы построения функциональных и процессных систем. В этих структурах существуют жестко регламентированные процессы, находящиеся под управлением менеджера процесса. При этом деятельность осуществляется работниками, находящимися в оперативном подчинении менеджера процесса и в административном подчинении руководителя в функциональном «колодце».

*Преимущества*

* Комплексный подход к реализации проекта, решению проблемы;
* Концентрация усилий на решении одной задачи, на выполнении одного конкретного проекта;
* Большая гибкость структуры;
* Активизация деятельности руководителей проектов и исполнителей в результате формирования проектных групп;
* Усиление личной ответственности конкретного руководителя как за проект в целом, так и за его элементы.

*Недостатки*

* При наличии нескольких организационных проектов или программ проектные структуры приводят к дроблению ресурсов и заметно усложняют поддержание и развитие производственного и научно-технического потенциала компании как единого целого;
* От руководителя проекта требуется не только управление всеми стадиями жизненного цикла проекта, но и учет места проекта в сети проектов данной компании;
* Формирование проектных групп, не являющихся устойчивыми образованиями, лишает работников осознания своего места в компании;
* При использовании проектной структуры возникают трудности с перспективным использованием специалистов в данной компании;
* Наблюдается частичное дублирование функций.

***Смешанные структуры.*** Если применять различные модели организации деятельности в пределах отдельных бизнес-процессов, то можно использовать преимущества той или иной организационной модели. При этом для организации в целом будет применяться процессная организация основных структурных блоков, а в рамках отдельных блоков могут применяться различные модели.

**Методика выполнения**

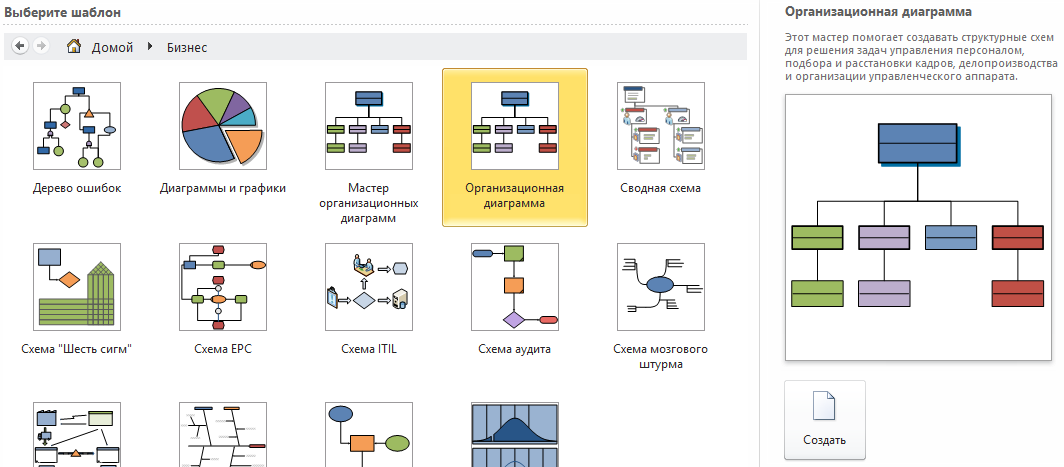
В качестве примера рассматривается процесс создания организационной диаграммы ВУЗа двумя способами:

1. вручную;
2. с помощью мастера.

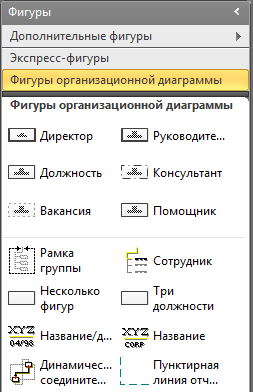
***1 Создание организационной диаграммы вручную***

Для построения организационной диаграммы вручную необходимо проделать следующие действия:

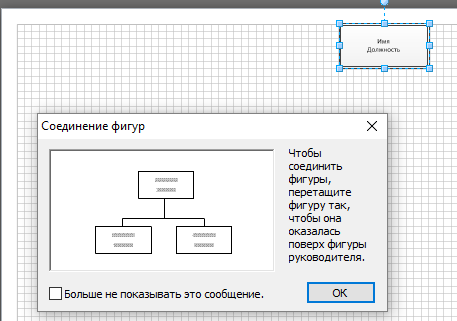
1. Запустить *MS Visio*.
2. В разделе *Выберите шаблон* выбрать категорию *Бизнес*, а затем *Организационная диаграмма* и нажать кнопку *Создать*.



В разделе фигур можно увидеть основные элементы, необходимые для построения организационной диаграммы.

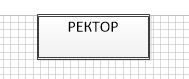


1. Перетащите фигуру *Директор* из набора фигур *Фигуры организационной диаграммы* в центр вверху страницы документа.
2. Надстройка организационных диаграмм отображает анимированное диалоговое окно, показывающее, как нужно добавлять в схему дополнительные фигуры.



1. Не снимая выделения с фигуры, при необходимости введите *ФИО*, затем нажмите клавишу ENTER и во второй строке введите *Должность*.

При создании организационной диаграммы ВУЗа можно ограничиться только указанием должностей.



1. Добавьте еще две фигуры Директор и расположите по обе стороны от фигуры Ректор. Соедините их используя Соединительную линию. В результате должна получиться схема.



1. Перетащите фигуру Руководитель на фигуру Директор. Затем, при необходимости, введите ФИО, нажмите клавишу ENTER и введите Должность.

Надстройка автоматически размещает новую фигуру под фигурой *Директор*.

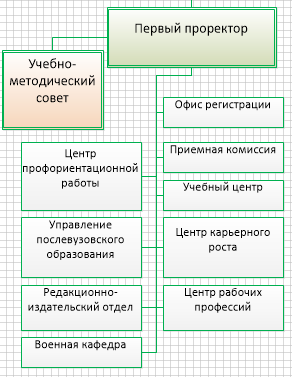
1. Повторите предыдущий шаг и обратите внимание, что надстройка разместила вторую фигуру руководителя сбоку от первой. Разместите все необходимые фигуры *Руководитель*.

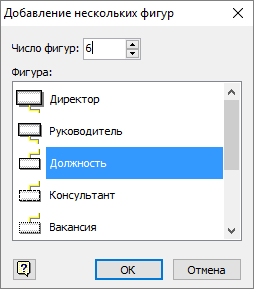


1. Используя фигуры *Руководитель* и *Несколько фигур* добавьте необходимые *Должности,* относящиеся к руководителю **Первый проректор**

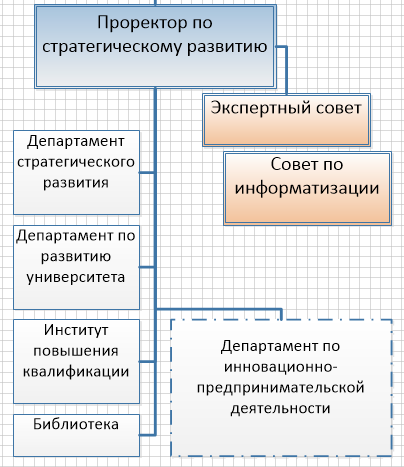
При использовании элемента *Несколько фигур* необходимо выполнить следующие действия:

* перетащить элемент Несколько фигур на необходимую фигуру;
* затем, в появившемся окне задать необходимые параметры, например, как на рисунке 9;
* нажать ОК.

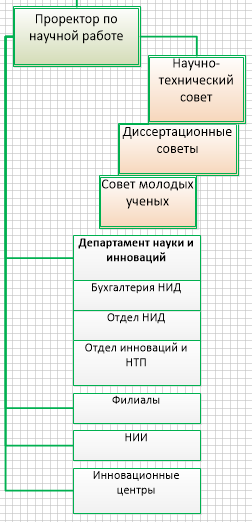


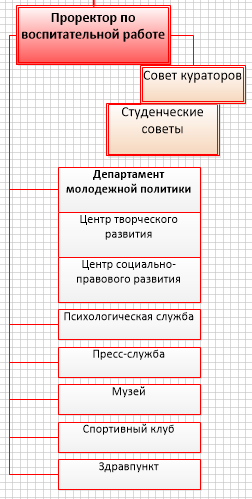


1. Повторите предыдущий шаг и создайте полную организационную диаграмму

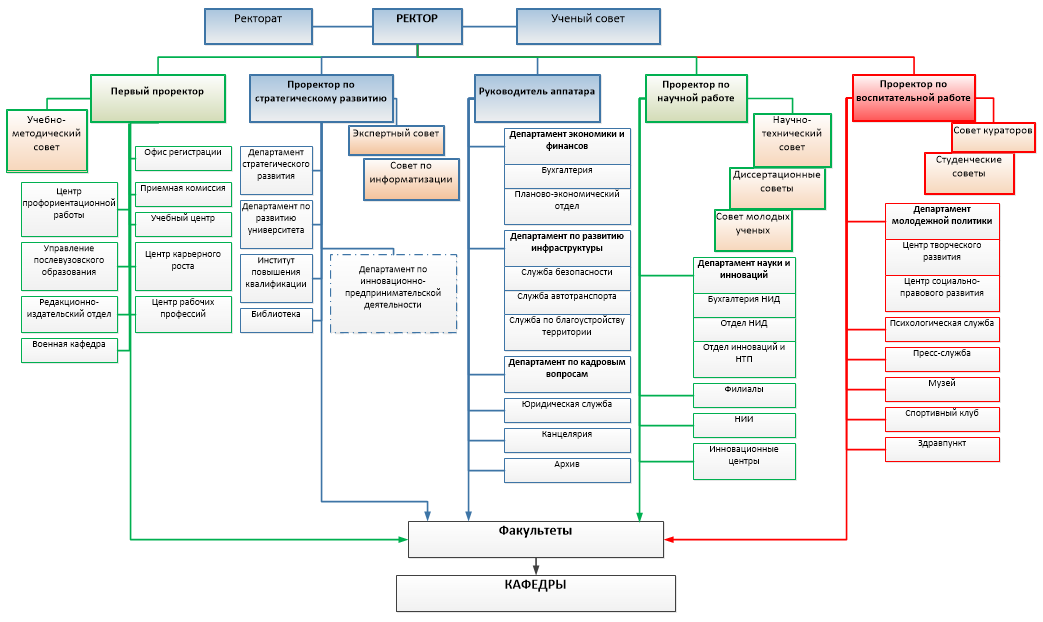








1. В результате выполнения всех описанных шагов организационная диаграмма ВУЗа должна принять вид

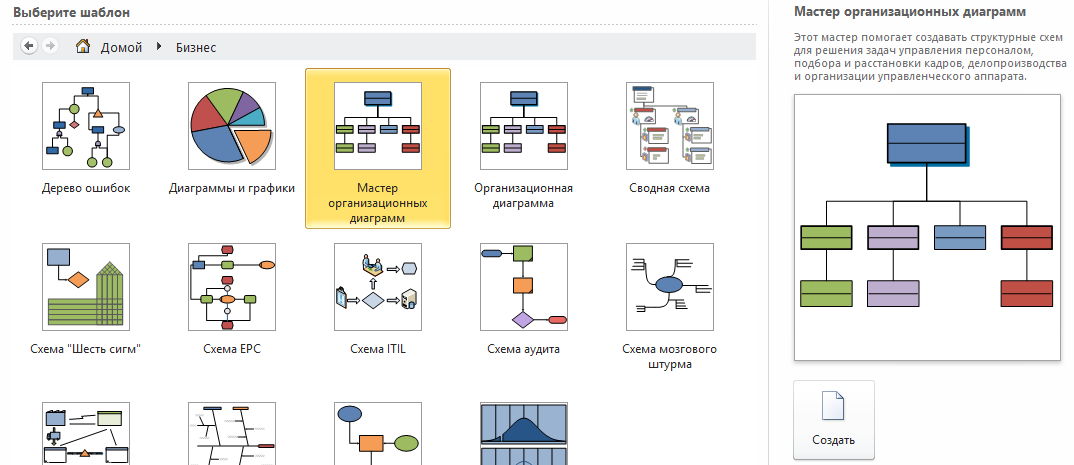


***2 Создание организационной диаграммы с помощью мастера диаграмм на основе данных***

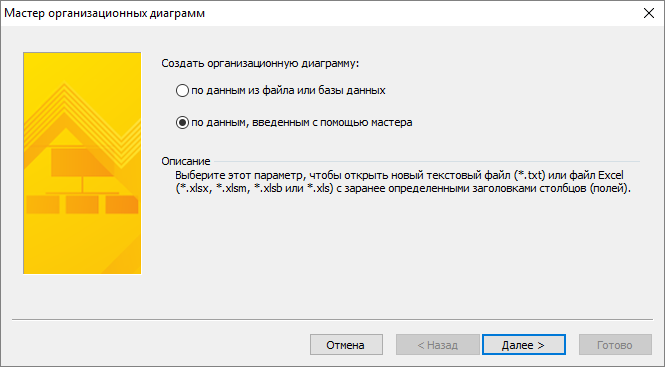
*Построение организационной диаграммы с помощью мастера рассмотрим также на примере ВУЗа, однако немного упростим структуру.*

Для построения организационной диаграммы с помощью мастера диаграмм необходимо проделать следующие действия:

1. Запустить *MS Visio*.
2. В разделе *Выберите шаблон* выбрать категорию *Бизнес*, а затем *Мастер организационных диаграмм* и нажать кнопку *Создать*.

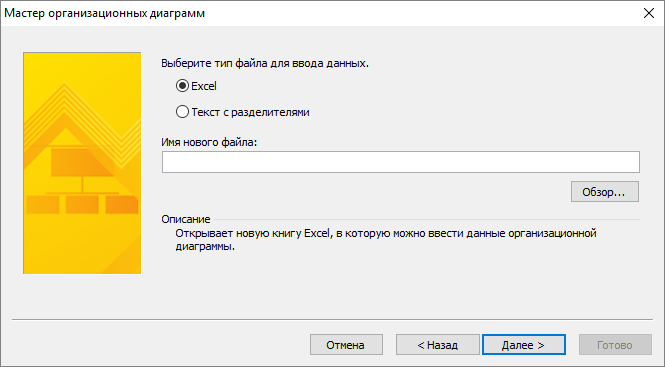


1. На первой странице мастера организационных диаграмм выберем переключатель *По данным, введенным с помощью мастера* (рис. 16). Обратите внимание – в описании этого переключателя подтверждается, что будет создан новый источник данных.

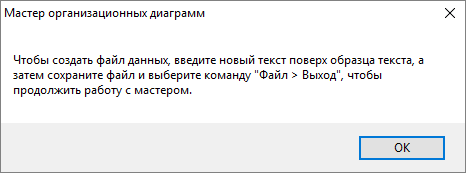


1. Далее необходимо нажать кнопку *Далее* и выбрать тип файла.

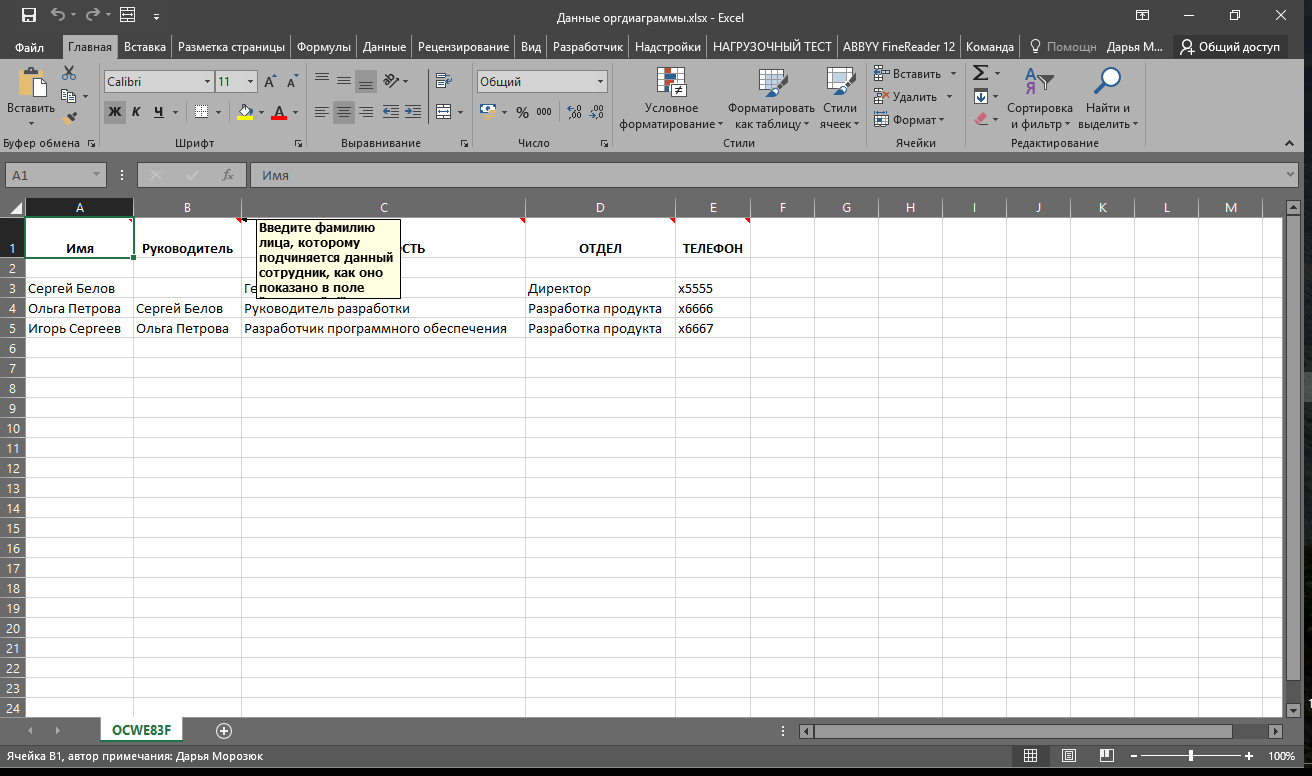
На странице выбора типа файла выберем переключатель *Excel* и щелкнем на кнопке *Обзор*. В открывшемся диалоговом окне выберем папку, в которую нужно сохранить файл, в поле *Имя файла* введем *Данные оргдиаграммы* и щелкнем на кнопке Сохранить. Имя файла отображается в поле *Имя нового файла*.



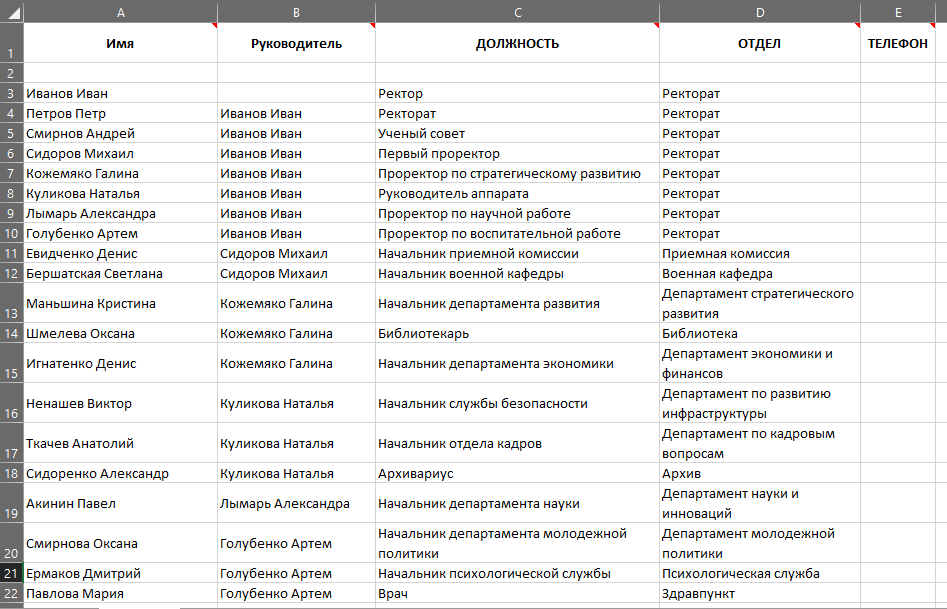
1. Щелкнем на кнопке *Далее*. MS Visio показывает на необходимость ввести свои данные поверх ранее введенных (рис. 18).



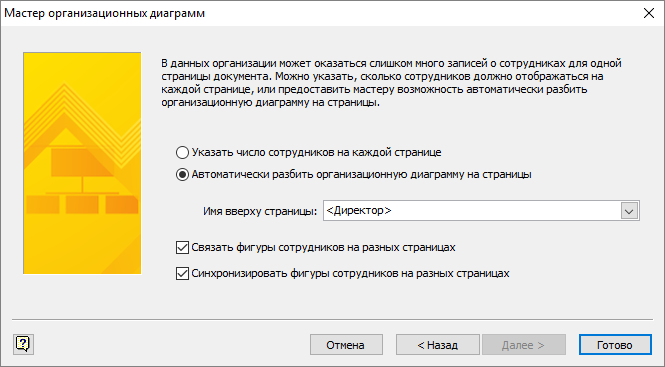
1. Щелкнем на кнопке *Ок*. В Excel отображается форматированная книга. ***На рисунке 19 показана книга Excel для ввода данных диаграммы, при этом заголовок каждого столбца включает примечание с инструкциями по вводу данных в этом столбце.***



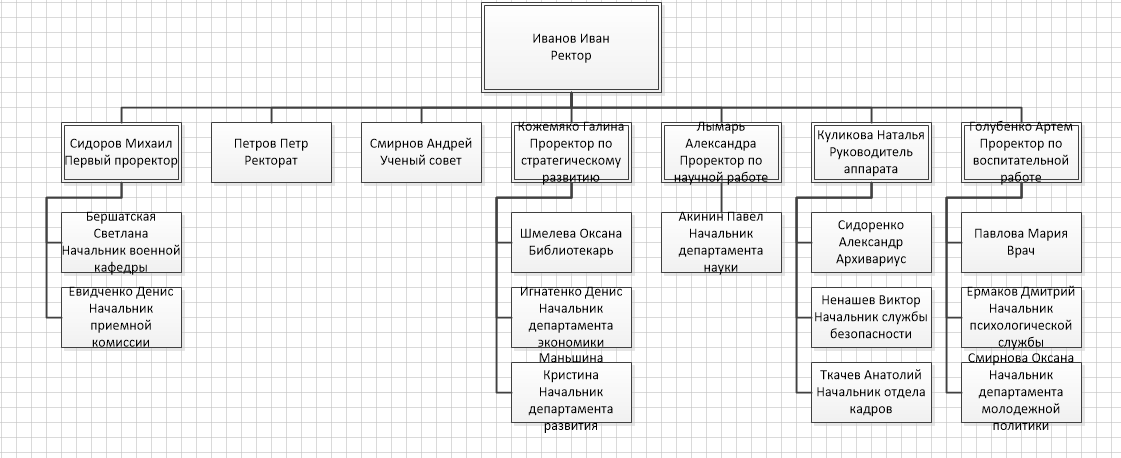
1. Далее вносим свои данные.



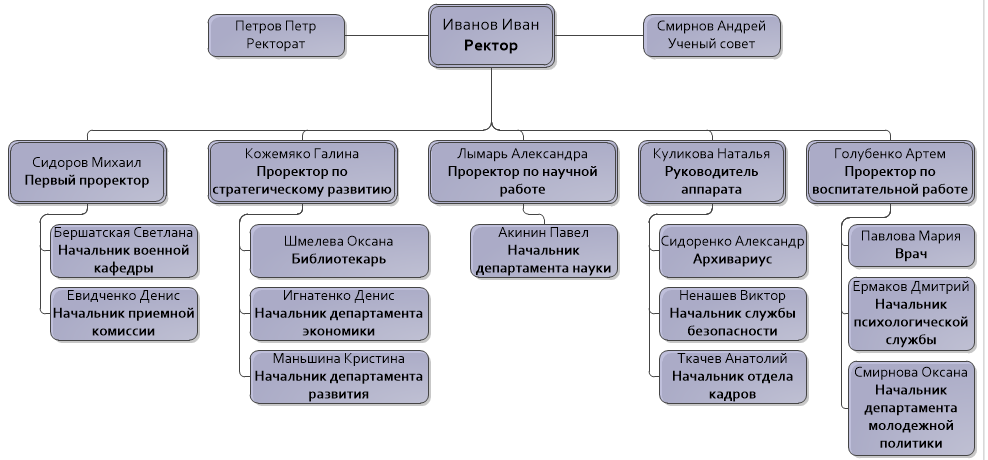
1. Закрываем Excel. После закрытия Excel происходит возврат в программу MS Visio, где открыта страница импорта фотографий мастера



1. Щелкнем на кнопке Готово, чтобы отобразить организационную диаграмму



1. На последнем шаге необходимо произвести форматирование диаграммы. В результате организационная диаграмма должна принять вид.



**Задание**

Построить организационную диаграмму в соответствии с вариантом.

Отчет по практическому занятию выполняется в формате MS Word, который содержит пошаговое описание процесса построения организационной диаграммы, а также скриншоты результатов согласно заданию.

**Варианты**

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

**Контрольные вопросы**

1. Что такое организационная диаграмма?
2. Способы построения оргдиаграмм в MS Visio?
3. Каковы принципы создания организационных диаграмм в MS Visio?
4. Какие существуют типы организационных структур? Перечислите их преимущества и недостатки.

# Практическая работа №6. Построение организационных диаграмм в Elma Community Edition

**Цель**: изучение основ создания организационных диаграмм с помощью CASE-средства ELMA.

**Теоретические сведения**

**Архитектура системы ELMA**

Система ELMA состоит из следующего набора программ:

1. Сервер ELMA;
2. Дизайнер ELMA;
3. ELMA Agent.

Сервер ELMA включает в себя набор связанных компонентов:

* базу данных;
* веб-приложение;
* сервер конфигураций;
* сервер оповещений.

Сервер ELMA является корневым элементом системы. Для начала работы с сервером его необходимо зарегистрировать.

Дизайнер ELMA – это приложение, которое устанавливается на компьютер пользователя и предоставляет удобный интерфейс для моделирования процессов, описания оргструктуры компании, а также создания конфигураций. Дизайнер ELMA также используется для выполнения ряда других задач. Для работы с Дизайнером его предварительно необходимо зарегистрировать.

ELMA Agent – это приложение, которое устанавливается на компьютер пользователя и в режиме реального времени показывает оповещения, которые сервер формирует для пользователя. Это позволяет ускорить реакцию пользователей на происходящие в системе события. Когда программа ELMA Agent запущена, в трее операционной системы (область рабочего стола рядом с часами, по умолчанию – правый нижний угол экрана) появляется его значок: ELMA Agent не требует регистрации для начала работы и может быть установлен на любое количество компьютеров в организации.

**Установка cервера и Дизайнера ELMA**

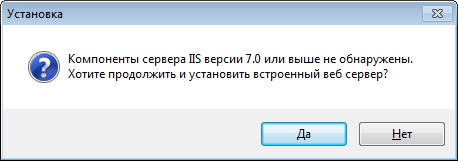
Установка сервера и Дизайнера ELMA осуществляется из единой программы-установщика; ELMA Agent устанавливается на компьютер отдельно (Об установке ELMA Agent читайте в руководстве пользователя системы ELMA). (Скачать бесплатную демо-версию всегда можно с сайта продукта - <http://www.elma-bpm.ru/download/index.html> )

Установка системы ELMA состоит из следующих шагов:

**Шаг 1.** Запуск программы установки системы ELMA.

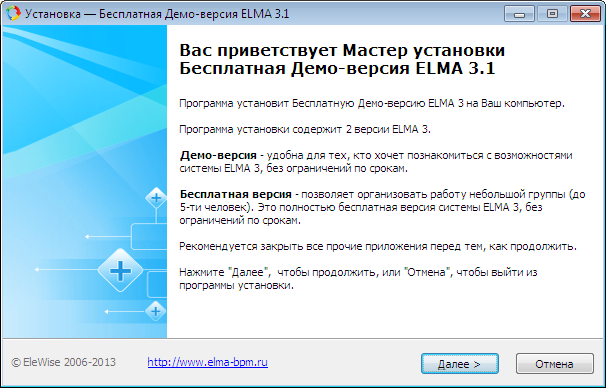
Установщик просканирует операционную систему на наличие установленного сервера IIS версии 7.0 или выше.

Если сервер еще не установлен, пользователю будет предложено установить встроенный веб-сервер из установщика системы ELMA. Чтобы подтвердить установку сервера, необходимо нажать кнопку Да. Отказаться от установки встроенного веб-сервера можно при помощи кнопки Нет – в этом случае установка системы ELMA будет прервана.



Если ранее установленный сервер IIS версии 7.0 или выше был обнаружен на компьютере, процесс установки системы ELMA автоматически перейдет к шагу 2.

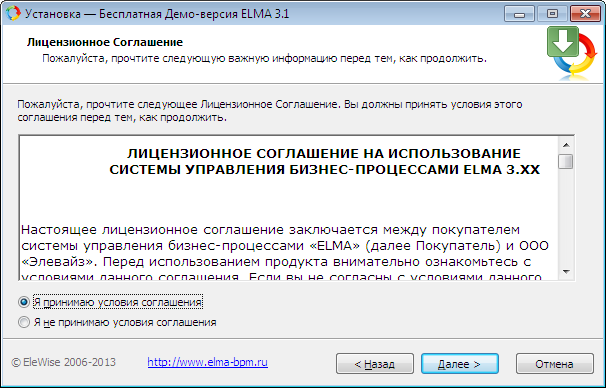
**Шаг 2.** Просмотрите приветствие Мастера установки ELMA и подтвердите необходимость установки.



По возможности закройте все работающие приложения и нажмите кнопку Далее для продолжения.

**Шаг 3.** Ознакомьтесь с текстом лицензионного соглашения.

После прочтения лицензионного соглашения, если вы согласны со всеми его пунктами, установите переключатель в положение Я принимаю условия соглашения. Для продолжения нажмите кнопку Далее.



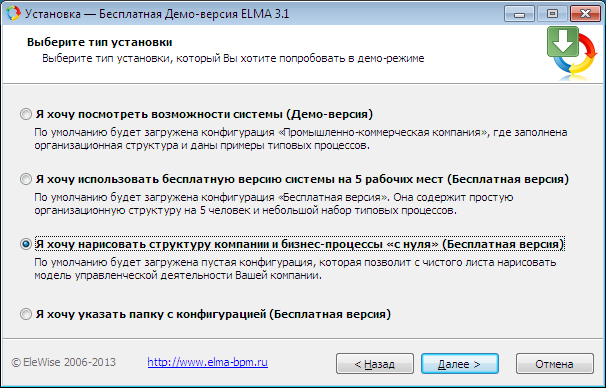
**Шаг 4.** Выберите тип установки:

Посмотреть возможности системы. По умолчанию будет загружена конфигурация "Промышленно-коммерческая компания", где заполнены организационная структура и даны примеры типовых процессов.

Использовать бесплатную версию системы на 5 рабочих мест. По умолчанию будет загружена конфигурация "бесплатная версия". Она содержит простую организационную структуру на 5 человек и небольшой набор типовых процессов.

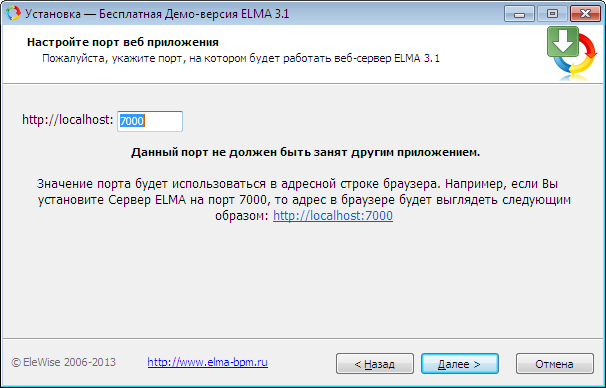
Нарисовать структуру компании и бизнес-процессы "с нуля". По умолчанию будет загружена конфигурация, которая позволит с чистого листа нарисовать модель управленческой деятельности Вашей компании.

Выбираем третий тип установки «Нарисовать структуру компании и бизнес-процессы "с нуля"»



**Шаг 5.** Настройте порт веб-приложения

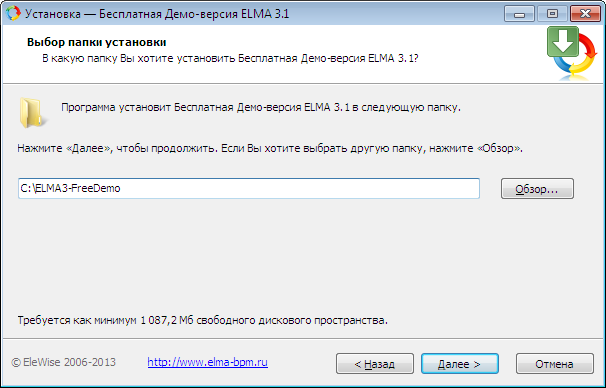
Здесь указывается порт, не занятый другими приложениями (По умолчанию 7000)



Значение порта будет использоваться в адресной строке браузера для входа в систему пользователям – http://127.0.0.1:7000

**Шаг 6.** Выберите папку, в которую будут установлены файлы cервера ELMA:

Рекомендуется использовать для установки папку по умолчанию, однако при желании вы можете выбрать любую другую. Папку установки можно указать вручную в соответствующем поле ввода или же воспользоваться кнопкой Обзор, чтобы указать нужную папку при помощи мыши в диалоговом окне обзора дисков операционной системы.

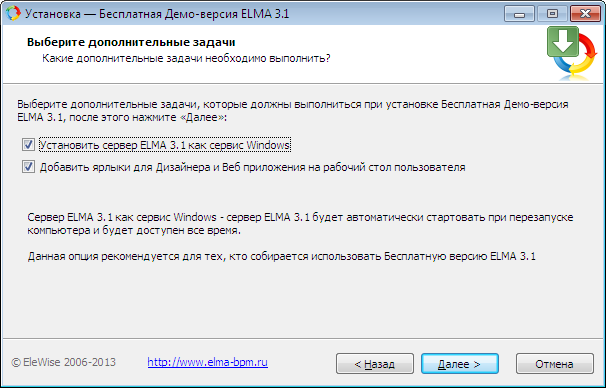


Для продолжения нажмите кнопку Далее.

**Шаг 7.** Выберите дополнительные задачи.

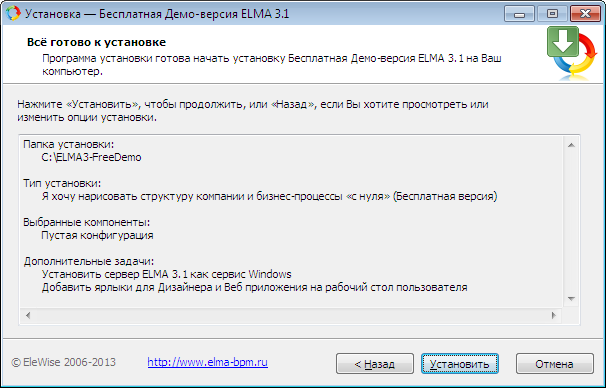
Установите флажок Добавить ярлыки на рабочий стол пользователя, чтобы в процессе установки системы ELMA на рабочем столе операционной системы пользователя автоматически были созданы ярлыки для быстрого запуска компонентов системы ELMA.

Для продолжения установки нужно нажать кнопку Далее.



**Шаг 8.** Подтвердите параметры установки.

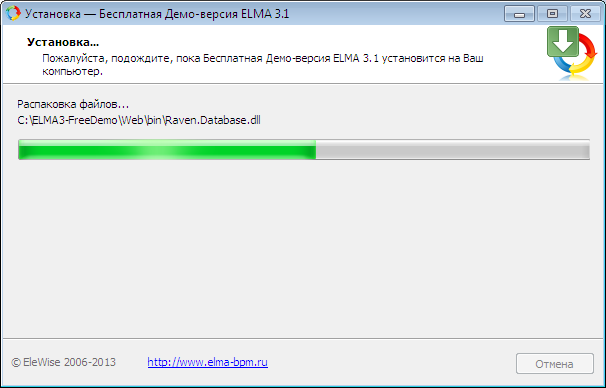
Проверьте правильность всех указанных в списке параметров. Для продолжения установки нужно нажать кнопку Установить.



**Шаг 9.** Подождите, пока закончится установка системы ELMA на ваш компьютер.

Процесс установки системы ELMA может занимать значительное время – дождитесь, пока копирование файлов не завершится.

О прогрессе установки можно судить по полосе прогресса. После завершения копирования файлов, переход на следующий шаг установки будет произведен автоматически.



**Шаг 10.** Подтвердите перезагрузку системы.

После завершения установки системы ELMA потребуется выполнить перезагрузку операционной системы. Рекомендуется выполнить перезагрузку немедленно, однако в случае крайней необходимости вы можете отложить этот процесс.

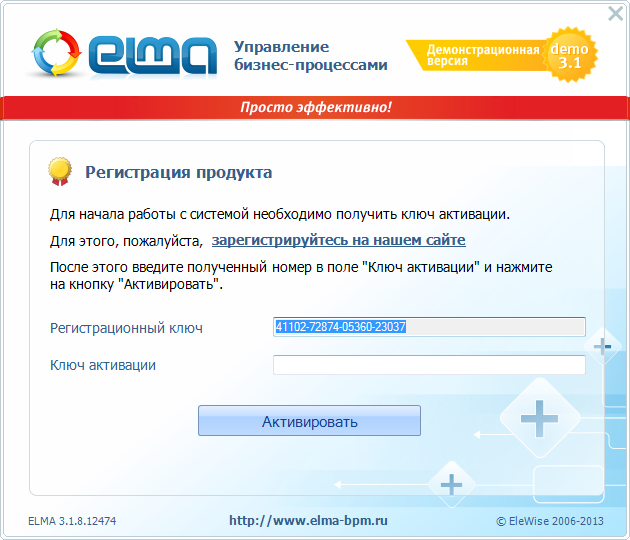
Чтобы отложить процесс перезагрузки системы, установите переключатель в положение Нет, я произведу перезагрузку позже. Позднее выполнить перезагрузку компьютера потребуется вручную.

Перед перезагрузкой системы не забудьте сохранить и закрыть все открытые для редактирования документы и файлы – в противном случае все не сохраненные данные будут потеряны.

Для завершения процесса установки нажмите кнопку Завершить.

**Регистрация продукта**

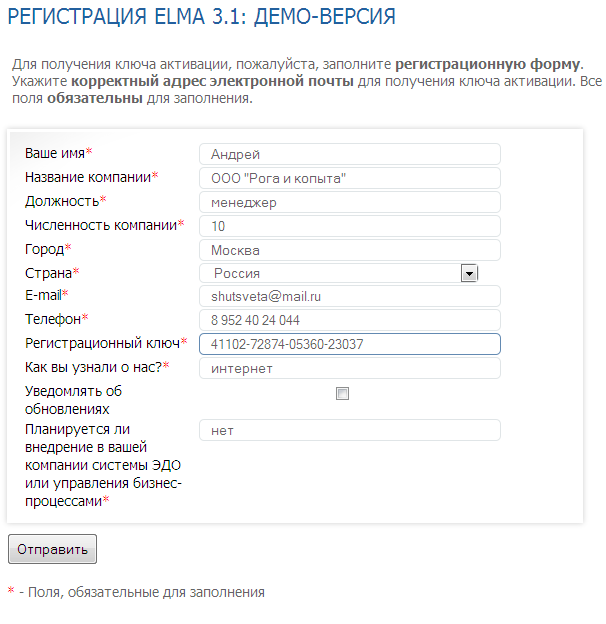
Для начала работы с системой необходимо получить ключ активации.



Для этого надо зарегистрировать систему на сайте разработчика - <http://www.elma-bpm.ru/download/register_demo_3.0.html>

Заполняем регистрационную форму. Указываем корректный адрес электронной почты для получения ключа активации. Заполняем все поля формы и нажимаем кнопку Отправить

Далее сайт выдает ключ активации и также Вы его получаете на свой e-mail



Вводим полученный ключ активации в форму и нажимаем кнопку «Активировать»

Если все сделано правильно программа выдаст окно, что система зарегистрирована

**Проверка установки сервера ELMA.**

Для проверки корректности установки сервера ELMA откройте веб-браузер на сервере и откройте страницу http://127.0.0.1:7000 (если вы указали другой порт для веб-сервера ELMA – используйте его). Через некоторое время (первый запуск занимает продолжительное время) система должна отобразить страницу для входа. В этом случае система установлена корректно.

После этого осуществляется первый запуск системы.

**Первый запуск сервера**

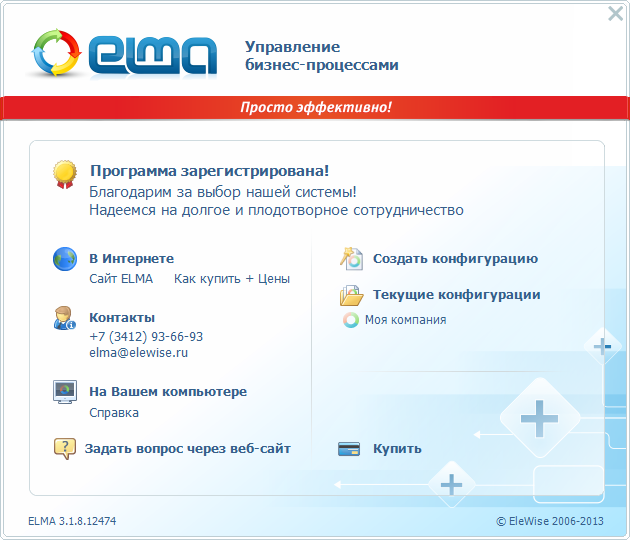
После завершения установки системы ELMA, в зависимости от установленной версии, в меню «Пуск» появится соответствующая папка: ELMA. При помощи содержимого данной папки можно выполнить запуск нужных компонентов системы ELMA:

* Дизайнер ELMA. Запустить программу Дизайнер ELMA, чтобы приступить к моделированию оргструктуры компании, бизнес-процессов, отчетов, маршрутов документов и настройке документооборота организации.
* Открыть ELMA в браузере. Открыть для работы веб-приложение системы ELMA
* Справка по ELMA. Перейти к изучению документации по ELMA
* Удалить ELMA. Если по какой-то причине требуется удалить ELMA из системы – сделать это можно при помощи соответствующей команды меню.

**Первый запуск Дизайнера ELMA**

После запуска Дизайнера ELMA откроется окно со списком доступных конфигураций – этот список можно увидеть ниже кнопки Текущие конфигурации в правой части окна запуска Дизайнера ELMA.

В окне запуска Дизайнера ELMA вы можете создать новую конфигурацию сервера или выбрать любую из уже существующих. Выберем конфигурацию «Моя компания»



Щелкните мышью по названию конфигурации в правой части окна запуска Дизайнера ELMA — откроется окно для ввода логина и пароля. По умолчанию можно зайти в конфигурацию под логином admin, пароль для администратора не задан. Нажмите кнопку ОК.

После прохождения авторизации и загрузки данных, система будет готова к работе. О том, как провести начальную настройку системы, смотрите далее.

**Знакомство с интерфейсом**

Веб-интерфейс

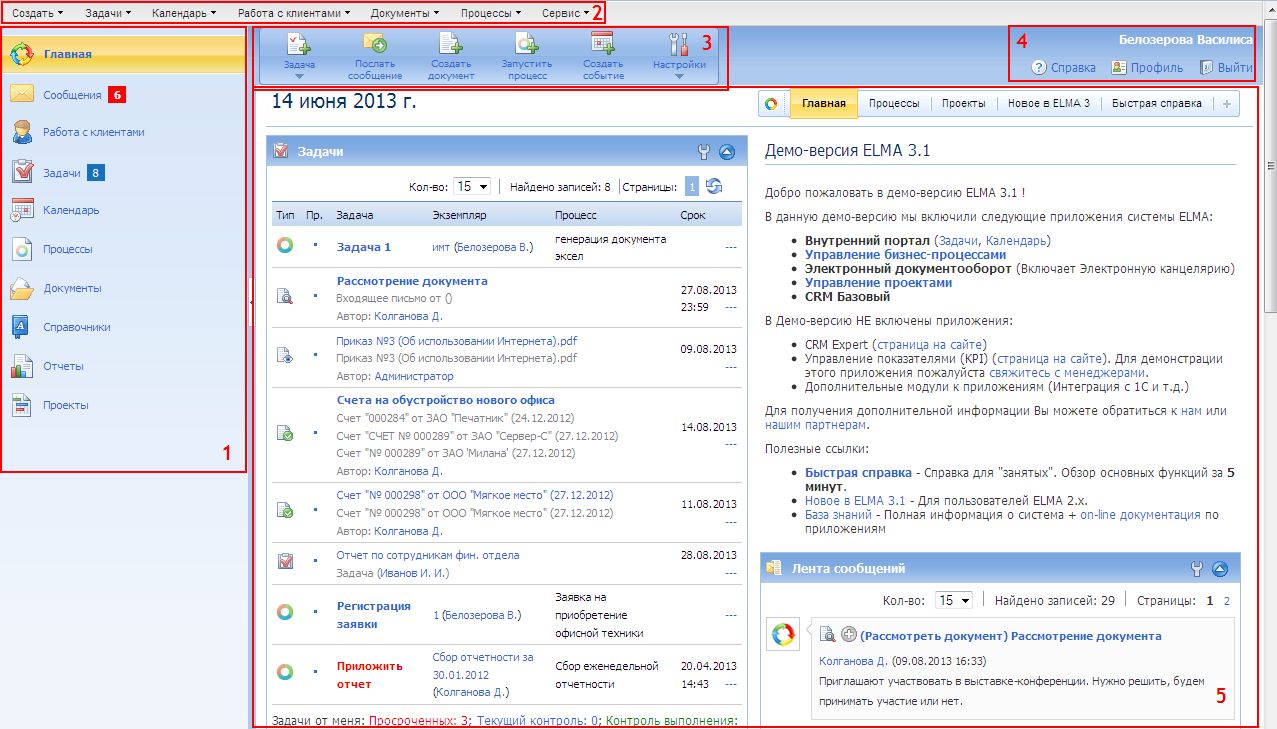
Основную часть времени при работе с системой ELMA пользователи проводят в веб-приложении ELMA.

Пользователь, который хорошо разобрался в интерфейсе веб-приложения и понял назначение его основных элементов, с легкостью освоит все остальные компоненты системы.

**Главная страница**

Первое, что видит пользователь, когда выполняет вход в веб-приложение системы – главную страницу.

Главная страница – это своего рода монитор системы ELMA, на котором пользователь видит все, что происходит в системе, состояние своих дел: сообщения, события в календаре, задачи, документы и т.д. Повседневную работу с веб-приложением ELMA рекомендуется начинать с просмотра главной страницы.



На главной странице находятся элементы навигации – перемещения по системе ELMA, элементы управления ею, а также настройки – каждый пользователь может настроить отображение главной страницы для себя так, как ему удобно.

На главной странице находятся четыре меню и наполнение самой страницы:

1. Боковое меню. Здесь перечислены все доступные для пользователя разделы веб-приложения ELMA. Набор пунктов этого меню зависит от конфигуации системы, настроек бокового меню и прав доступа конкретного пользователя. В процессе работы это меню может видоизменяться. Текущий раздел меню всегда подсвечивается желтым цветом, а ссылка на главную страницу всегда доступна в первой строчке бокового меню.
2. Горизонтальное меню. Так же меню работы с системой. Оно частично дублирует главное меню. Горизонтальное меню помимо команд перехода в различные разделы веб-приложения также содержит множество команд, например, таких, как создание записей в календаре.
3. Панель инструментов. Для каждой страницы она имеет свой набор кнопок в зависимости от назначения страницы, от ее функционала. Кнопки меню главной страницы системы в основном служат для создания различных объектов системы: создания задач для ваших подчиненных, создания записей в календаре, создания документов и т. д.
4. Меню пользователя. В зависимости от настроек ELMA состоит из двух или трех ссылок:

* Сменить пароль – в зависимости от настроек ELMA либо присутствует в личном меню, либо нет. При нажатии на эту ссылку, система ELMA предложит вам форму, в которой вы сможете сменить пароль.
* Настройки – присутствует в меню всегда. Эта ссылка ведет на страницу с вашими персональными настройками, где вы можете завести все данные о себе, настроить свою работу с документами и задачами, определить группы пользователей, настроить сертификаты – право электронной подписи.
* Выйти – присутствует в меню всегда. При помощи данной ссылки можно выйти из своего профиля и выполнить вход в систему ELMA под учетной записью другого пользователя системы.

1. Наполнение страницы. Здесь отображается наполнение страницы или раздела, выбранной с помощью любого меню. Около заголовка страницы справа находится кнопка вызова помощи. Она ведет на страницу помощи, относящуюся к текущему разделу.

Наполнение главной страницы составляют портлеты. Портлет – это небольшое окно с некоторой тематической актуальной информацией на текущий момент. Например, портлет Задачи в компактной форме сообщает вам о том, какие задачи вам предстоит выполнить. Отображаемую в портлете информацию можно настраивать, для настройки портлета нажмите кнопку в его заголовке, и в появившемся диалоговом окне укажите нужные параметры работы.

В системе ELMA существует большое количество стандартных портлетов, кроме того вы можете создавать собственные. Состав и расположение портлетов главной страницы можно настраивать по своему усмотрению.

В системе может быть создано несколько главных страниц. Каждая главная страница может содержать различный набор портлетов. Например, первая главная страница системы настроена для работы с документами, вторая настроена для управления проектами, третья – для контроля работы ваших подчиненных и т. п. Главные страницы создаются и настраиваются пользователями и администратором системы; существует возможность настроить уникальные главные страницы для отдельных пользователей системы.

Для переключения между главными страницами служит специальная лента, расположенная в правом верхнем углу главной страницы. Текущая главная страница выделена желтым цветом. Переключиться на другую главную страницу можно щелкнув мышью по соответствующей кнопке ленты.

c:\users\pr\desktop\курсы для вузов\главная страница лента.jpg

Подробнее о портлетах и их настройке читайте в руководстве пользователя системы ELMA

Остальные страницы веб-приложения системы ELMA отличаются от главной страницы в основном содержимым наполнения страницы и содержимым панели инструментов. В остальном их интерфейс идентичен.

**Меню системы**

При помощи горизонтального и бокового меню вы сможете переходить на всевозможные разделы системы, а также выполнять команды. Содержимое бокового и горизонтального меню во многом совпадают, но имеют немного различную специфику. Горизонтальное меню содержит в основном пункты, реализующие выполнение некоторых команд, в то время как боковое меню служит скорее для навигации по разделам системы.

Большинство пользователей предпочитает работать с боковым меню – его элементы имеют крупные удобные размеры.

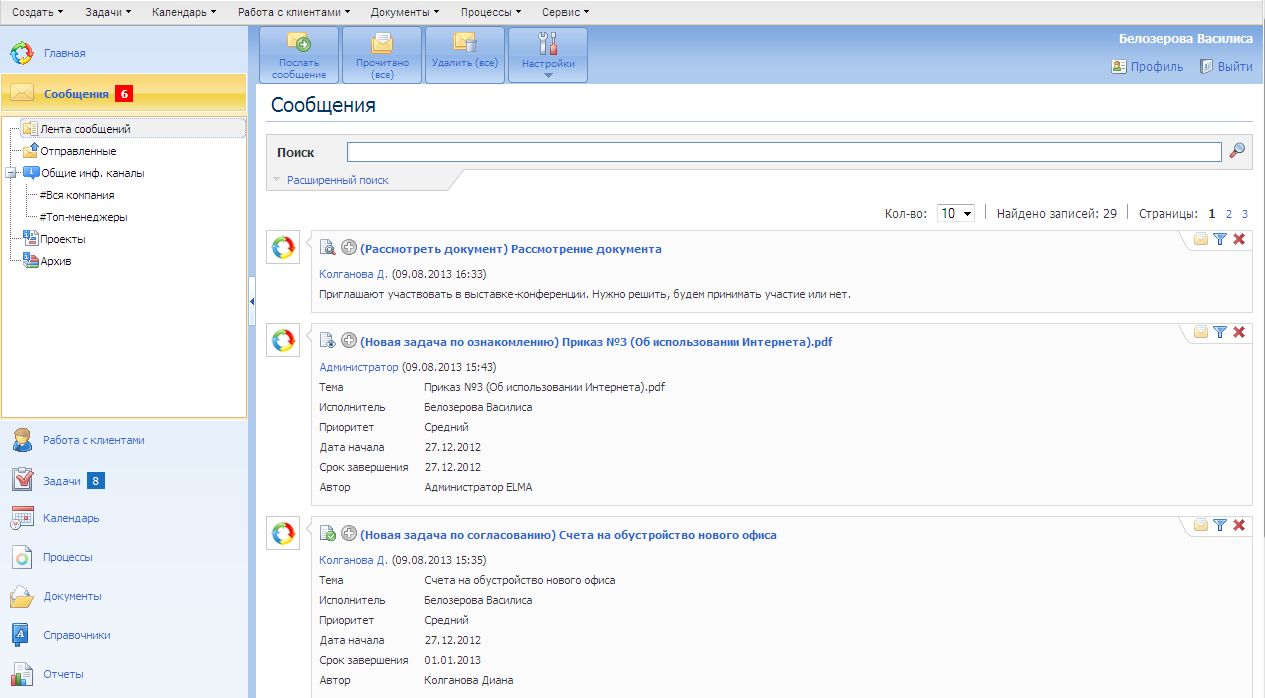
Горизонтальное меню представляет собой обычное многоуровневое меню – работа с подобными меню является привычным делом для пользователей операционных систем семейства Windows.

Подробно о содержимом меню системы, а также их настройке читайте в справке по Порталу системы.

Также, своего рода разновидностью меню является панель инструментов, которая есть на каждой странице системы ELMA. Ее содержимое для каждой страницы системы является уникальным. Вообще, здесь можно обнаружить крупные кнопки, которые позволяют быстро выполнить некоторое действие, специфичное для данного раздела. Например, находясь в разделе Документы на панели инструментов можно обнаружить кнопки создания и регистрации документов, отправки документов по маршруту и т. д.

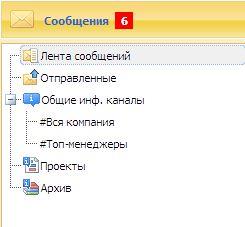
**Сообщения**

Очень важным разделом веб-приложения является раздел Сообщения – он предназначен для получения оповещений о разных событиях и изменениях в системе (например, оповещения о создании или изменении задачи, документа, комментарии к задаче или документу и др.), а также для получения сообщений от пользователей системы.



Данный функционал системы доступен всегда, независимо от того настроена ли возможность получать письма из системы на электронную почту или нет.

Попасть в раздел Сообщения можно из бокового меню выбором одноименного его пункта. В боковом меню вы увидите структуру раздела:



**Лента сообщений.** Данная страница содержит полный список послан­ных вам сообщений от системы ELMA. Сообщения и комментарии к ним отображаются в виде блога, поэтому вы сразу можете увидеть динамику об­суждений – разные уровни отступа сообщений позволяют сразу определить, какое сообщение является комментарием.

Чтобы оставить комментарий к некоторому сообщению ленты, нажмите кнопку в правом верхнем углу сообщения; в появившемся диалоговом окне укажите комментарий и нажмите кнопку Добавить комментарий.

Отправленные. На данной странице отображается список сообщений, которые были лично вами отправлены другим пользователям системы.

**Инф. каналы.** На данной странице можно просмотреть сообщения информационных каналов системы. Информационные каналы позволяют реализовать общение между определенной группой сотрудников на некоторую тему, например, обсуждение выполнения некоторого проекта. Таким образом, все обсуждения некоторой темы можно просмотреть в одном месте.

Также находясь в разделе Сообщения, вы можете написать сообщение другому пользователю системы. Для этого нажмите кнопку Послать сообще­ние панели инструментов. В появившемся диалоговом окне укажите имя получателей письма (их может быть несколько), тему письма и непосредственно, текст письма.

Если требуется послать сообщение в информационный канал, установите флажок напротив имени нужного информационного канала. Все пользователи, которым вы адресовали письмо, получат соответствующие уведомления и смогут его прочитать в своем разделе Сообщения.

**Построение организационной структуры**

Для чего нужна организационная структура в системе

Организационная структура является очень важным элементом в системе. Организационная структура используется:

При моделировании бизнес-процессов. Все участники бизнес- процессов назначаются только из организационной структуры.

При работе с модулем Задачи. Начальник видит все задачи своих подчиненных. Информация о том, кто кому подчиняется, берется из организационной структуры.

При работе с модулем Календарь. Начальник видит все кален­дарные планы своих подчиненных. Информация о том, кто кому подчиняется, берется из организационной структуры.

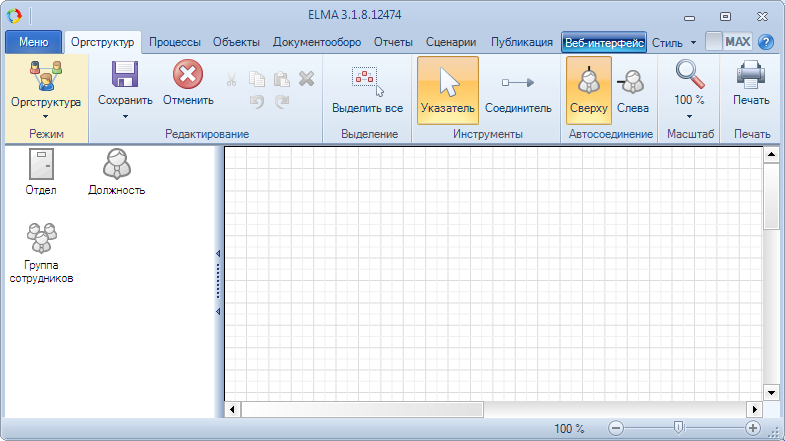
**Моделирование организационной структуры**

Для построения организационной структуры у вас должен быть от­крыт Дизайнер ELMA; выполните подключение к серверу ELMA.

При открытии Дизайнера ELMA вы по умолчанию окажетесь на вкладке Оргструктура, которая содержит инструменты для моделирования организа­ционной структуры компании. Центральная часть окна программы содержит разлинованную область, на которой хранится схема организационной струк­туры. В новой конфигурации оргструктура будет пуста.

Организационная структура описывает набор структурных подразделе­ний в организации и связи подчинения между ними.

В системе ELMA организационная структура должна иметь древовид­ную структуру. В самом верху находится, как правило, директор, либо гене­ральный директор компании.

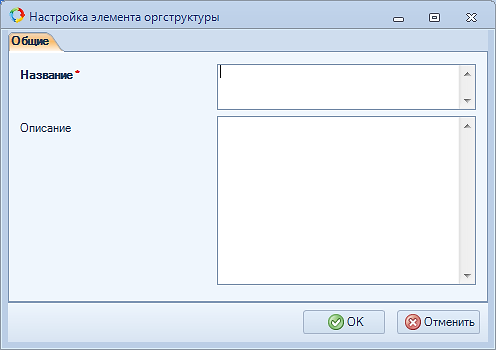


Для редактирования организационной структуры используется механизм Drag-and-Drop. Боковая панель вкладки содержит список возможных элементов оргструктуры. Всего их три: Отдел, Должность и Группа сотруд­ников.

Для того чтобы поместить элемент в оргструктуру, необходимо подвести указатель мыши к иконке элемента на боковой панели, нажать и, удерживая нажатой левую кнопку мыши, переместить иконку элемента на разлинованную область окна. Когда нужное место для элемента будет найдено, нужно отпустить левую кнопку мыши.

Начнем моделирование организационной структуры с генерального директора. Для этого переместим с боковой панели иконку Должность на пустое поле для моделирования организационной структуры.

Система попросит ввести название и описание для данной должности в иерархии организационной структуры. При моделировании оргструктуры, как правило, начинают с генерального директора организации или иного на­чальника высшего уровня.

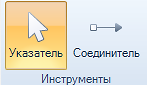


Далее для генерального директора надо добавить подчиненных. Это можно сделать двумя способами:

**Способ 1**

Чтобы назначить элементу, размещенному в оргструктуре, подчиненный элемент:

1. Убедитесь, что в группе Инструменты панели инструментов нажата кнопка Указатель.



1. Подведите указатель мыши к элементу оргструктуры. Слева, снизу и справа от элемента появится три небольших зеленых иконки, форма которых полностью повторяет иконки боковой панели окна: Отдел, Должность и Группа сотрудников.
2. Подведите указатель мыши к зеленой иконке элемента, который хотите сделать подчиненным.
3. Нажмите и удерживайте левую кнозампку мыши.
4. Перемещайте указатель мыши, пока значок подчиненного элемен­та оргструктуры не окажется на нужном месте.
5. Отпустите левую кнопку мыши.
6. В появившемся поле ввода укажите имя вновь созданного элемента оргструктуры.

Элементы оргструктуры, созданные таким способом, будут автоматически связаны линией подчинения с другими элементами оргструктуры.



**Способ 2**

Суть способа заключается в том, что сначала нужные элементы оргструктуры размещаются на схеме оргструктуры, а только затем указываются связи между ними:

1. Перетащите иконку нужного элемента с боковой панели на нужное место схемы оргструктуры – разлинованная часть окна. При этом данный участок должен быть свободен от других элементов.
2. Нажмите на кнопку Соединитель группы Инструменты панели инструментов.
3. Теперь вам потребуется вручную указать, какой элемент оргструктуры будет в иерархии выше, а какой ниже. Сначала требуется указать старший по должности элемент. Для этого подведите ука­затель мыши к данному элементу. Около его значка появятся небольшие изображения крестиков голубого цвета. При подведении указателя мыши к одному из таких крестиков, его изображение сменится изображением красного квадрата. Подведите указатель мыши к одной из таких областей.
4. Нажмите и удерживайте левую кнопку мыши.
5. Перемещайте указатель мыши. Пунктирная линия указывает на предполагаемое размещение линии подчинения. Подведите указатель мыши к элементу оргструктуры, который должен стать подчиняемым для первого. При подведении указателя мыши к элементу оргструктуры, около его значка появятся небольшие изображения крестиков голубого цвета. При подведении указателя мыши к одному из таких крестиков, его изображение сменится изображением красного квадрата. Подведите указатель мыши к понравившемуся голубому крестику.
6. Отпустите левую кнопку мыши. Линия подчинения вместо пунктира будет прочерчена сплошным цветом.

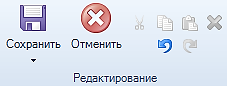
Точно так же можно добавить другие элементы организационной структуры.

Всего на организационной структуре могут быть элементы 3-х типов:

|  |  |
| --- | --- |
| c:\users\pr\desktop\курсы для вузов\орг. структура_должность.png | На **должность**может быть назначен ровно один сотрудник.  Типичные должности: директор, главный бухгалтер, секретарь и т.д. |
| c:\users\pr\desktop\курсы для вузов\орг. структура_группа_сотрудников.png | **Группа сотрудников**объединяет в себе группу пользователей, которые выполняют одну и ту же работу. Среди сотрудников в группе нет главного. **Группа сотрудников**используется при моделировании бизнес-процессов, когда не важно, кто из сотрудников выполнит операцию, а важно лишь, чтобы операция была выполнена в срок. В этом случае задача ставится всем сотрудникам из группы и к выполнению приступает тот, кто первым поставит пометку о начале работы над задачей. |
| c:\users\pr\desktop\курсы для вузов\орг. структура_отдел.png | **Отдел** состоит из:   1. Начальника отдела. 2. Сотрудники отдела.   Система позволяет назначить не более одного начальника отдела и любое количество сотрудников отдела. |

Все элементы на схеме организационной структуры должны быть объединены связями подчинения – «висячих» элементов быть не должно.

После создания организационной структуры, ее необходимо сохранить.

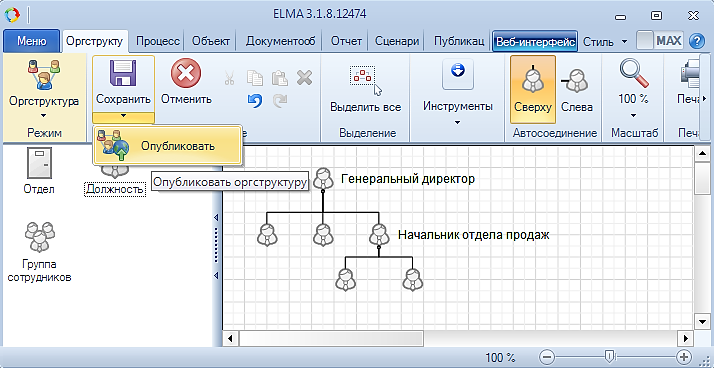


Для этого нажмите на основную часть многофункциональной кнопки Сохранить в группе Редактирование панели инструментов. Для того, чтобы из­менения в организационной структуре вступили в силу, ее необходимо Опубликовать.

Чтобы опубликовать оргструктуру, необходимо щелкнуть мышью по нижней части многофункциональной кнопки Сохранить в группе Редакти­рование панели инструментов. В появившемся контекстном меню выберите пункт Опубликовать.

**Публикация организационной структуры**

После того, как оргструктура была сохранена, ее еще нельзя будет использовать при работе с системой – предварительно оргструктуру требуется опубликовать.



Публикация организационной структуры делает доступными измене­ния, которые были сделаны с момента последней публикации. После публи­кации новой версии организационной структуры изменения будут доступны в следующих разделах:

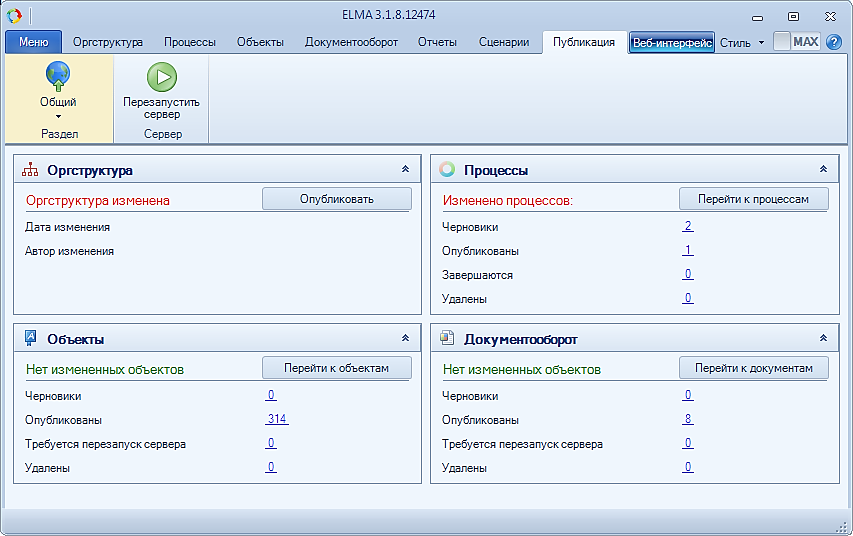
Управление учетными записями. Пользователей можно будет назначать на должности, созданные в новой организационной структуре.

При моделировании бизнес-процессов появится возможность работы с новой организационной структурой.

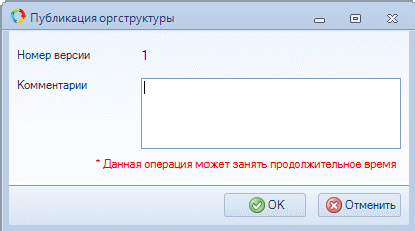
**Опубликовать организационную структуру можно двумя способами.**

**Способ 1.** Щелкните мышью по нижней части многофункциональной кнопки Сохранить в группе Редактирование панели инструментов. В появив­шемся контекстном меню выберите пункт Опубликовать.

**Способ 2.** Перейдите на вкладку Публикация Дизайнера ELMA.В блоке Оргструктура нажмите кнопку Опубликовать. Данная кнопка доступна в том случае, если с момента последний публикации организационная структура была модифицирована.



После нажатия на кнопку публикации оргструктуры появится диалоговое окно Публикация оргструктуры. Введите комментарий, который описывает изменения, сделанные в организационной структуре, и нажмите кнопку ОК.



Нарисуем простую оргуструктуру состоящую из директора, отдела продаж, бухгалтера и офис-менеджера и опубликуем её.

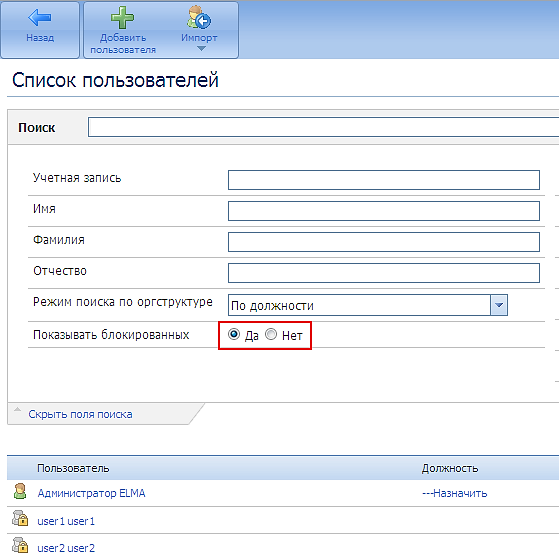
**Создание учетных записей**

После создания и публикации организационной структуры можно пере­ходить к созданию учетных записей пользователей.

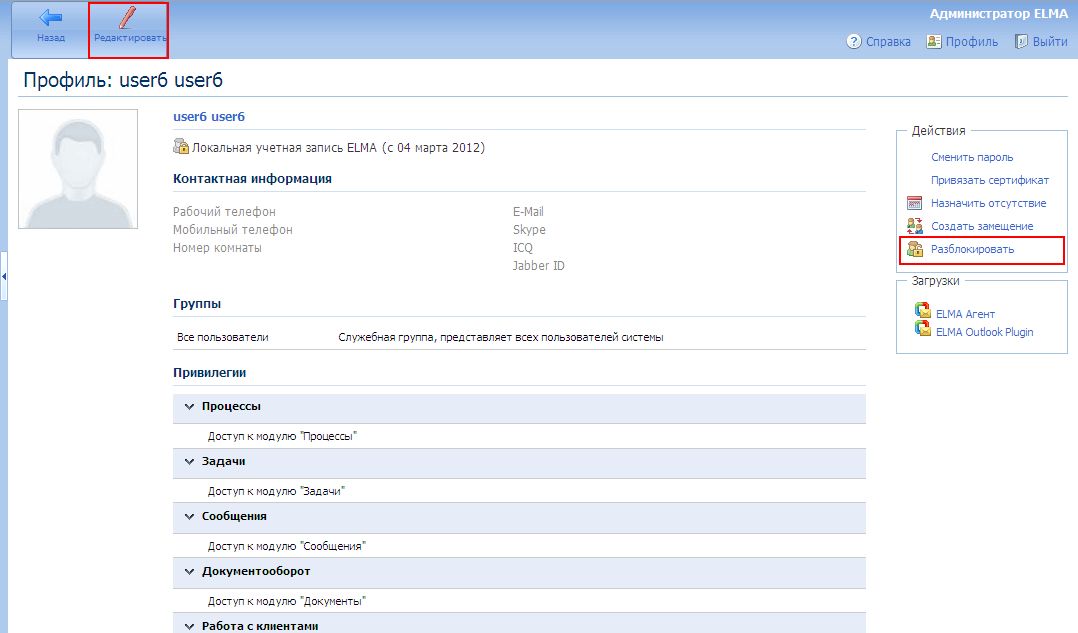
Для этого откройте веб-приложение ELMA (http://127.0.0.1:7000/ ), используя логин админи­стратора системы (Login: admin).

При помощи соответствующего пункта бокового меню перейдите в раз­дел Администрирование. В разделе Администрирование перейдите на стра­ницу Пользователи.

В бесплатной нельзя добавлять пользователей, по умолчанию они все заблокированы, чтобы их увидеть, надо в списке пользователей в расширенном поиске поставить "Показывать блокированных - да" и в Профиле заблокированного пользователя нажать Разблокировать.



Для создания/изменения новой учетной записи системы необходимо перейти в данного пользователя и нажать кнопку Редактировать.



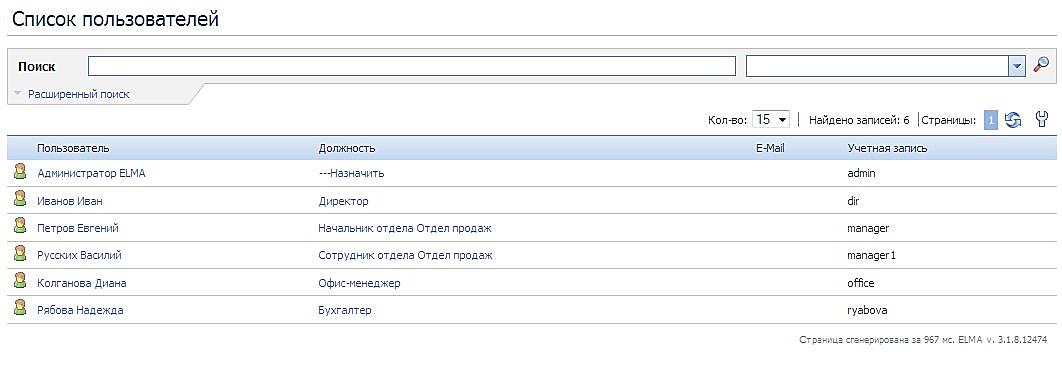
После нажатия на данную кнопку вы сможете заполнить информацию о пользователе. Вам обязательно нужно указать Учетную запись, Фамилию, Имя.

На вкладке Контакты вы можете указать контактные данные пользователя. Все поля данной вкладки заполняются по желанию.

Далее необходимо указать, какую позицию занимает данный пользователь в организационной структуре предприятия. Для этого просто нажмите на поле Должность и выберите в появившемся диалоговом окне нужную должность. Пользователь может занимать одновременно сразу несколько должностей.

Когда заполнение всех полей карточки пользователя будет завершено, нажмите кнопку Сохранить.

Для нашей оргструктуры заведем пользователей с должностями – Директор, Бухгалтер, Начальник отдела продаж, Менеджер по продажам и Офис-менеджер.



**Задание**

Построить организационную диаграмму в соответствии с вариантом.

Отчет по практическому занятию выполняется в формате MS Word, который содержит пошаговое описание процесса построения организационной диаграммы, а также скриншоты результатов согласно заданию.

**Варианты**

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №7. ER-моделирование в нотации Чена

**Цель:** Изучение методов построения логической модели базы данных выбранной предметной области

**Теоретические сведения**

В настоящее время существуют две группы методов проектирования схемы базы данных:

• методы с использованием диаграмм сущность-связь;

• методы функциональных и многозначных зависимостей.

Методы, основанные на использовании диаграмм сущность-связь, обеспечивают относительно простой способ формирования схемы базы данных. Однако после проектирования схемы базы данных требуется дополнительная проверка, находятся ли построенные таблицы в соответствующей нормальной форме (обычно схема базы данных приводится к третьей нормальной форме).

Методы с использованием функциональных и многозначных зависимостей являются более формализованными. Однако эти методы могут приводить к построению схемы базы данных с таблицами в труднопонимаемой со стороны пользователя форме. Кроме того, сами алгоритмы проектирования схемы имеют не полиномиальную временную сложность, что ограничивает их использование для баз данных с большим количеством атрибутов.

Модели сущность-связь основаны на выделении в предметной области, для которой осуществляется проектирование базы данных, различных типов объектов, информацию о которых требуется хранить в базе данных.

Набор однотипных объектов предметной области образует сущность. Между сущностями могут быть установлены информационные связи (зависимости), которые также могут быть учтены при проектировании схемы базы данных. Совокупность сущностей и связи между ними составляют информационную модель данных предметной области (Entity-Relationship диаграмму).

В настоящее время существует несколько приемов выделения сущностей и связей – нотации Чена, Мартина, Баркера, IDEF1X и т.д.

***Основные определения***

*Сущность (Entity)* – реальный либо воображаемый объект, имеющий существенное значение для рассматриваемой предметной области. Каждая сущность должна иметь наименование, выраженное существительным в единственном числе. Каждая сущность должна обладать уникальным идентификатором. Каждый экземпляр сущности должен однозначно идентифицироваться и отличаться от всех других экземпляров данного типа сущности.

*Атрибут (Attribute)* – любая характеристика сущности, значимая для рассматриваемой предметной области и предназначенная для квалификации, идентификации, классификации, количественной характеристики или выражения состояния сущности. Наименование атрибута должно быть выражено существительным в единственном числе.

*Связь (Relationship)* – поименованная ассоциация между сущностями, значимая для рассматриваемой предметной области.

***Нотация Чена***

В нотации Чена различают зависимые и независимые сущности.

Сущность называется ***независимой***, если каждый экземпляр сущности может быть однозначно идентифицирован без определения его отношений с другими сущностями.

Сущность называется ***зависимой*** от идентификаторов или просто зависимой, если однозначная идентификация экземпляра сущности зависит от его отношения к другой сущности.

Связь соединяется с ассоциируемыми сущностями линиями. Возле каждой сущности на линии, соединяющей ее со связью, цифрами указывается класс принадлежности.

Сущности и связи могут иметь атрибуты. Для каждой сущности находится атрибут (или набор атрибутов), значение которого однозначно определяет экземпляр сущности. Этот атрибут является ключом сущности.

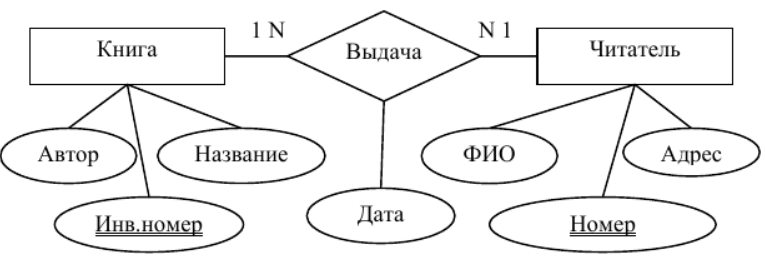
Связь также может иметь ключевой атрибут. В ряде случаев для удобства организации связей в состав атрибутов сущности вводится искусственный ключ (обычно число). Ключевой атрибут (набор атрибутов) на диаграмме отмечается двумя линиями снизу, внешние ключи отмечаются одной линией. В таблице представлены основные элементы, используемые для формирования ER-диаграммы в нотации Чена.

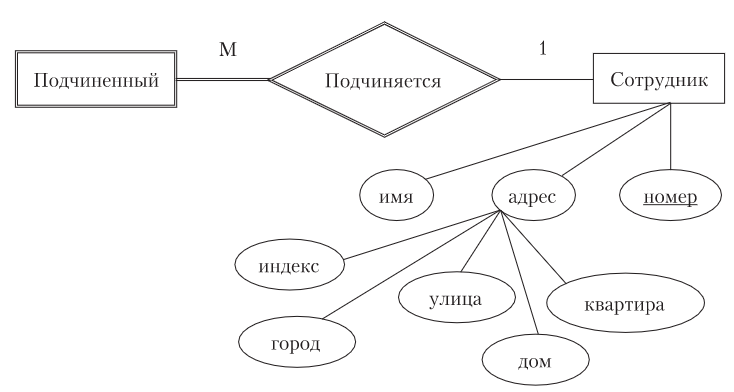
Таблица 1 – Элементы ER-диаграммы в нотации Чена

|  |  |
| --- | --- |
| **Элемент диаграммы** | **Описание** |
|  | Независимая сущность |
|  | Связь |
|  | Атрибут |
|  | Первичный ключ |
|  | Внешний ключ |

Связь имеет кардинальное число, определяющее, какое количество экземпляров одной сущности имеет информационную связь с экземплярами другой сущности.

На рисунках представлен пример ER-диаграммы в нотации Чена.





**Методика выполнения**

В качестве примера возьмем ***базу данных университета для описания учебного процесса.***

1. Проанализируйте предметную область.

База данных создаётся для информационного обслуживания сотрудников и студентов университета. БД должна содержать данные о сотрудниках, студентах, группах, предметах, оценках.

В соответствии с предметной областью система строится с учётом следующих особенностей:

* на одной должности работают несколько сотрудников;
* один сотрудник преподает несколько предметов;
* один предмет преподают несколько сотрудников;
* в одной группе несколько студентов;
* один студент получает оценки по нескольким предметам;
* по одному предмету оценку получают несколько студентов.

Для инфологического проектирования воспользуемся методом «сущность-связь». Для того, чтобы представить, как устроена предметная область нужно задать множество объектов реального мира (главная проблема что считать объектом). Объект – семантическое понятие, которое может быть полезно при обсуждении устройств реального мира. Сущность реального мира – объекты – не обязательно материальны – важно понятие существенно и различимо для других. Между объектами могут возникать связи трех видов:

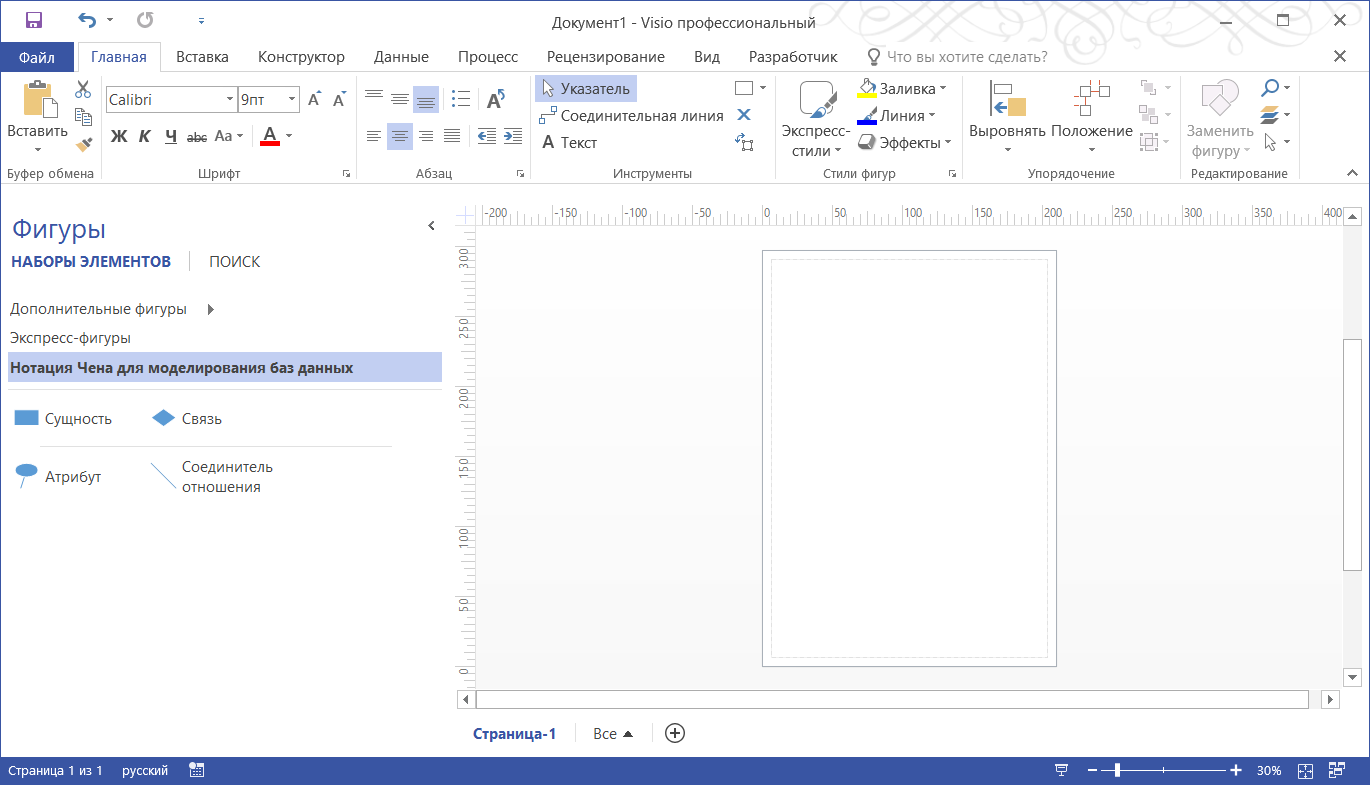
* один к одному 1:1 (пациент: место в палате);
* один к многим 1:n и многие к одному n:1 (абонент : номер телефона);
* многие ко многим n:n (пациент : хирург).

***Выделим базовые сущности этой предметной области:***

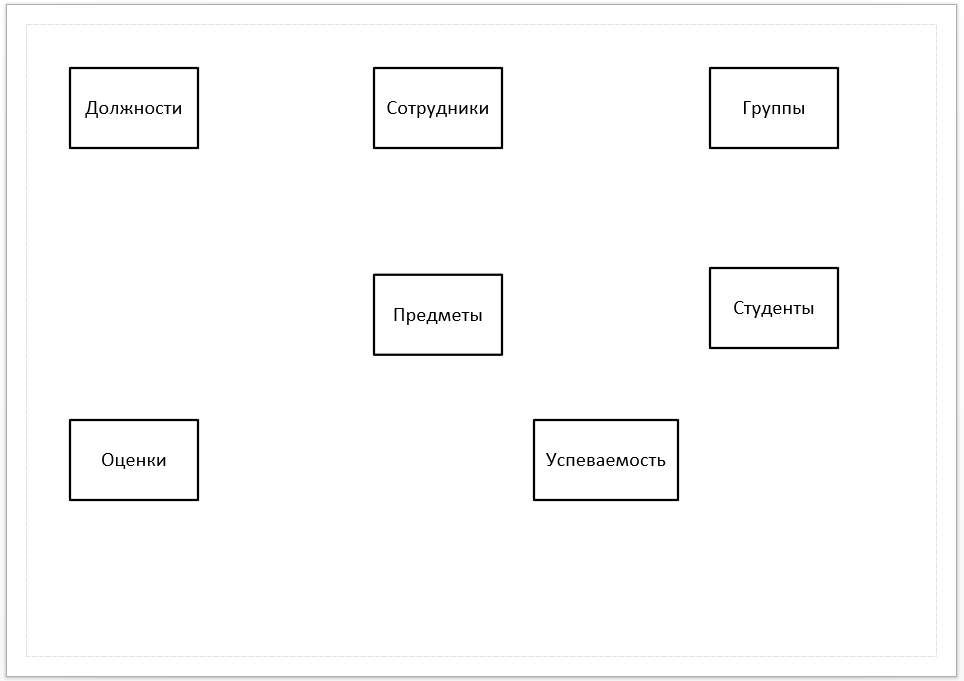
* Сотрудники. Атрибуты сотрудников – табельный номер (PK), ФИО, пол, дата рождения, адрес и телефон. Для сотрудника необходимо хранить код должности (FK).
* Должности. Атрибуты – ID (PK), название.
* Студенты. Атрибуты– номер зачетки (PK), ФИО, пол, дата рождения, адрес, телефон. Для студента необходимо хранить код группы (FK).
* Группы. Атрибуты– ID (PK), название.
* Предметы. Атрибуты – ID (PK), название. Для предмета необходимо хранить код сотрудника (FK).
* Оценки. Атрибуты – ID (PK), название.

Для отражения успеваемости в системе нужно учитывать по какому предмету каким студентом какая оценка получена. Для успеваемости необходимо хранить код студента (FK), код предмета (FK), код оценки (FK).

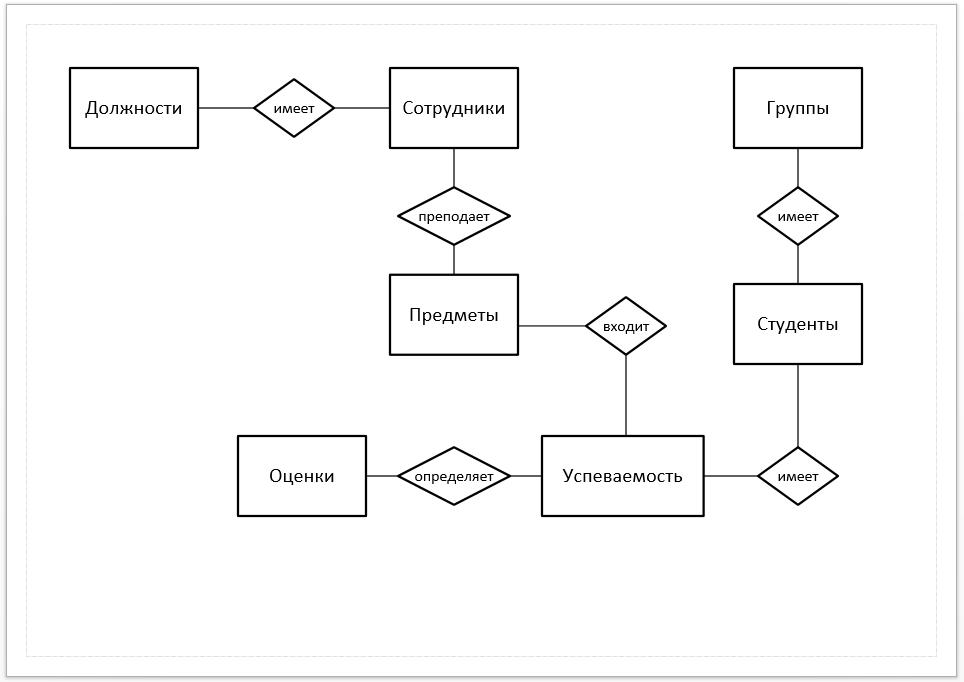
1. Запустите MS Visio.
2. На закладке выбора шаблона выберите категорию *Базы данных* и в ней элемент *Нотация Чена*. Нажмите кнопку *Создать* в правой части экрана.
3. Окно программы примет вид



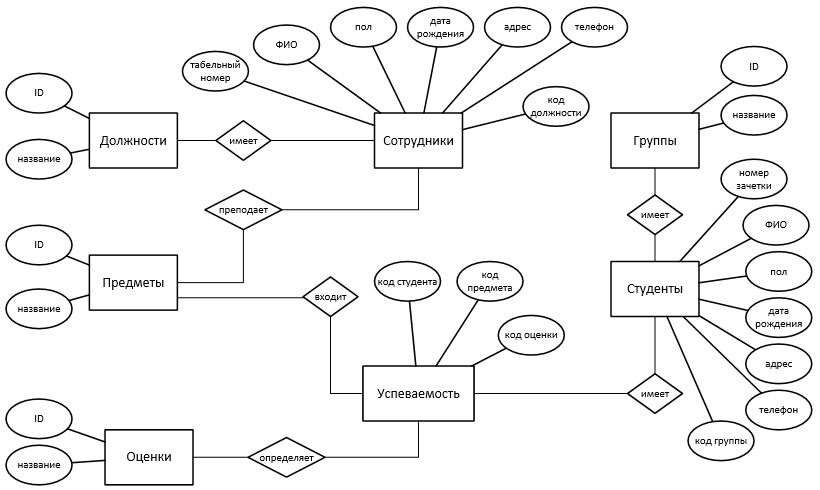
1. В соответствии с анализом предметной области необходимо добавить на рабочий лист четыре фигуры Сущность.
2. Для добавления текста фигуре необходимо дважды щелкнуть по ней левой кнопкой мыши.



1. Далее необходимо добавить связи между сущностями. Для этого необходимо использовать фигуру ромб.



1. На данном этапе получена общая диаграмма Чена. Для ее детализации необходимо добавить атрибуты ко всем сущностям. Атрибуты изображаются в виде овала

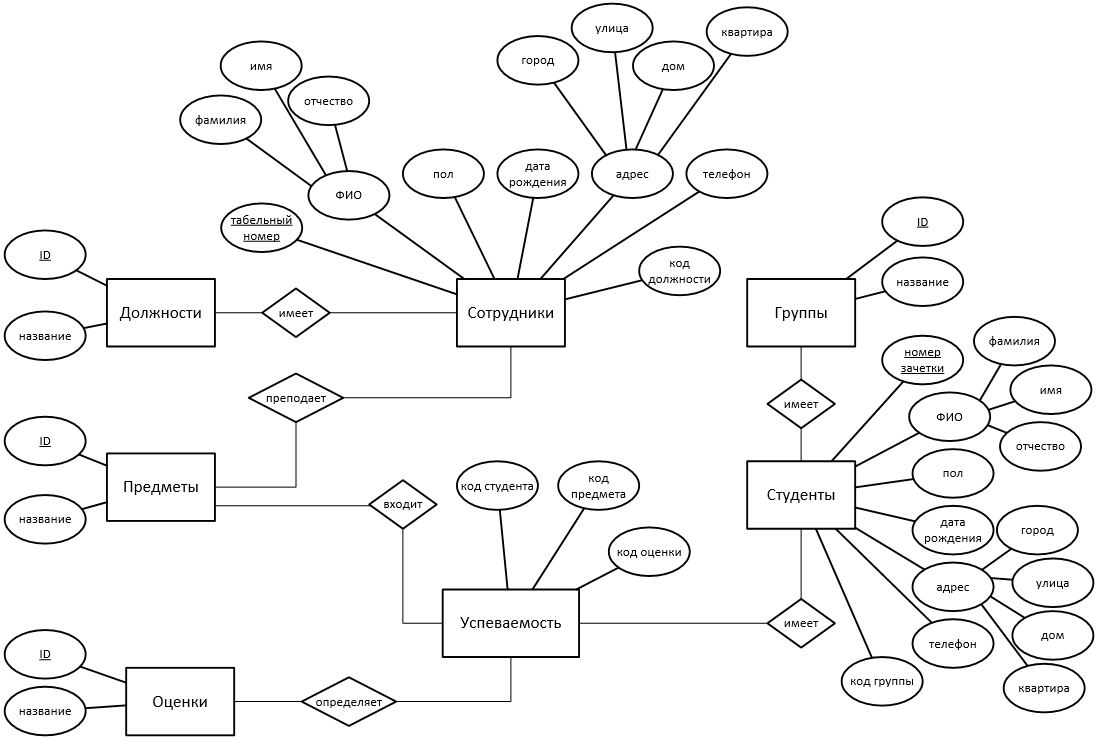


1. Далее необходимо разбить на дополнительные атрибуты ФИО и адрес для сотрудников и студентов, а также указать первичный и внешний ключ.

Для разбиения составного атрибута необходимо добавить фигуру овал и соединить с нужным атрибутом.

Для указания первичного ключа необходимо нажать правой кнопкой мыши на атрибуте и в контекстном меню выбрать пункт Задать первичный ключ.

Для указания внешнего ключа необходимо выбрать нужный атрибут и дополнительных параметрах шрифта выбрать двойное подчеркивание текста.



**Варианты**

Построить ER-модель для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

**Контрольные вопросы**

1. Группы методов проектирования схемы базы данных.
2. Приемы выделения сущностей и связей.
3. Основные определения ER-диаграмм.
4. Элементы диаграммы в нотации Чена.

# Практическая работа №8. ER-моделирование в нотации IDEF1X

**Цель:** освоение технологии построения информационной модели логического и физического уровней в нотации IDEF1X с использованием пакета Microsoft Visio.

**Теоретические сведения**

Методология IDEF1X – язык для семантического моделирования данных, основанный на концепции «сущность-связь».

Различают два уровня информационной модели: **логический** и **физический**.

**Логическая модель** позволяет понять суть проектируемой системы, отражая логические взаимосвязи между сущностями.

Различают 3 подуровня логического уровня модели данных, отличающиеся по глубине представления информации о данных:

* диаграмма сущность-связь (Entity-Relationship Diagram (ERD);
* модель данных, основанная на ключах (Key Based Model (KB);
* полная атрибутивная модель (Fully Attributed Model (FA).
  1. **Физическая модель** отражает физические свойства проектируемой базы данных (типы данных, размер полей, индексы). Параметры физической информационной модели зависят от выбранной системы управления базами данных (СУБД).

***Основные элементы информационной модели логического уровня***

*Сущности и атрибуты*

**Сущность** – это множество **реальных** или **абстрактных объектов** (людей, предметов, документов и т.п.), **обладающих общими атрибутами или характеристиками**. Любой объект системы может быть представлен только одной сущностью, которая должна быть уникально идентифицирована. *Именование сущности* осуществляется с помощью *существительного в единственном числе*. При этом имя сущности должно отражать **тип** или **класс** объекта, а не его **конкретный экземпляр** (например, **Студент**, а не **Петров**).

Любая сущность характеризуется набором атрибутов (**свойств**).

**Атрибут сущности** – характеристика сущности, то есть свойство реального объекта. Например, на рисуке атрибутами сущности **«Студент»** являются **«ID студента»**, **«Фамилия»**, **«Имя»**, **«Отчество»**, **«Дата поступления»** и **«Номер билета»**.

В свою очередь, *атрибуты сущности* делятся на 2 вида: *собственные* и *наследуемые. Собственные* атрибуты являются уникальными в рамках модели. *Наследуемые* атрибуты передаются от сущности-родителя при установке связи с другими сущностями.

**Первичный ключ (Primary Key, PK).** Каждая сущность должна обладать *атрибутом* или *комбинацией атрибутов*, чьи значения *однозначно определяют* каждый *экземпляр сущности*. Эти атрибуты образуют *первичный ключ* сущности.

**Внешний ключ (Foreign Key, FK)**. Если между двумя сущностями *имеется специфическое отношение* связи или *категоризации*, то *атрибуты*, входящие в *первичный ключ родительской* или *общей сущности*, *наследуются* в качестве *атрибутов сущностью-потомком* или *категориальной сущностью* соответственно. Эти атрибуты и называются внешними ключами. Наследуемый атрибут может использоваться в сущности в качестве части или целого первичного ключа, альтернативного ключа или не ключевого атрибута.

*Отношения в IDEF1X-модели*

При построении информационной модели различают следующие типы отношений между сущностями: *идентифицирующее*, *не идентифицирующее*, *не специфическое (многие-ко-многим) и отношения категоризации*.

**Мощность отношения** служит для обозначения отношения числа экземпляров родительской сущности к числу экземпляров дочерней.

***Нормализация данных***

*Нормализация* – это процесс проверки и реорганизации сущностей и атрибутов с целью удовлетворения требований к реляционной модели данных. Процесс нормализации сводится к последовательному приведению структур данных к нормальным формам – формализованным требованиям к организации данных.

Первая нормальная форма (1НФ). Сущность находится в первой нормальной форме тогда и только тогда, когда все атрибуты содержат атомарные значения. Среди атрибутов не должно встречаться повторяющихся групп, т.е. несколько значений для каждого экземпляра.

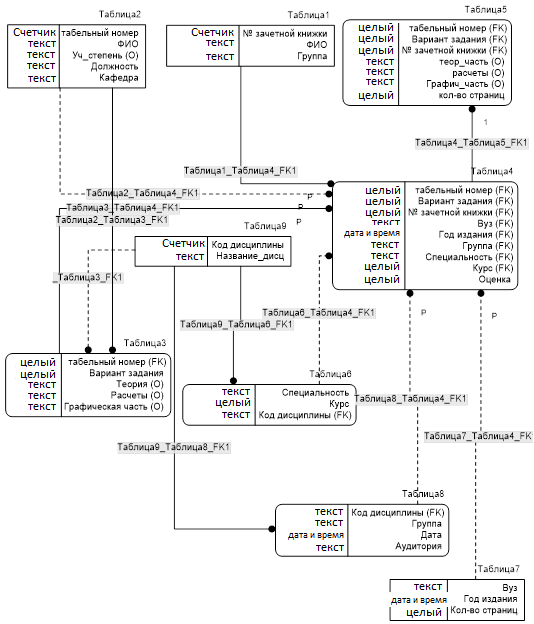
Вторая нормальная форма (2НФ). Сущность находится во второй нормальной форме, если она находится в первой нормальной форме, и каждый не ключевой атрибут полностью зависит от первичного ключа (не может быть зависимости от части ключа).

Третья нормальная форма (3 НФ). Сущность находится в третьей нормальной форме, если она находится во второй нормальной форме и никакой не ключевой атрибут не зависит от другого не ключевого атрибута (не должно быть зависимости между не ключевыми атрибутами).

**Пример модели логического уровня**

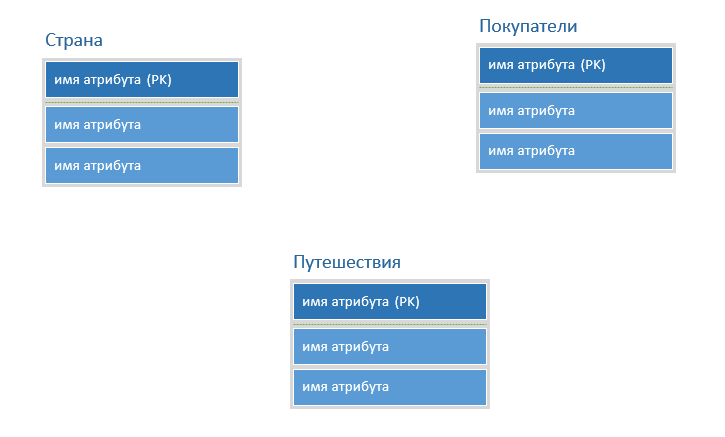


**Пример модели физического уровня**

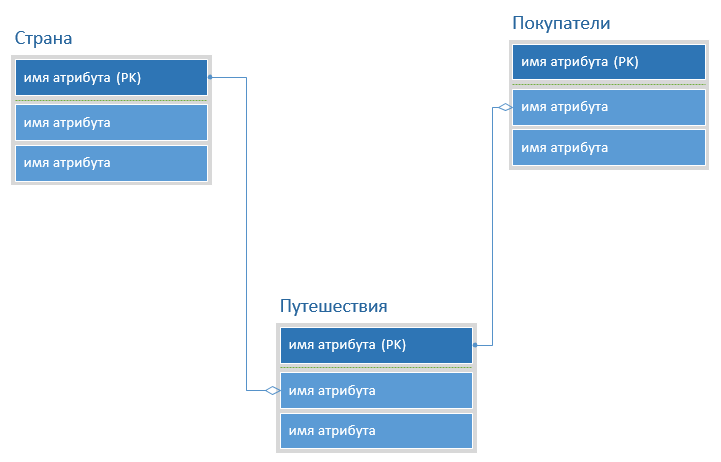


**Методика выполнения**

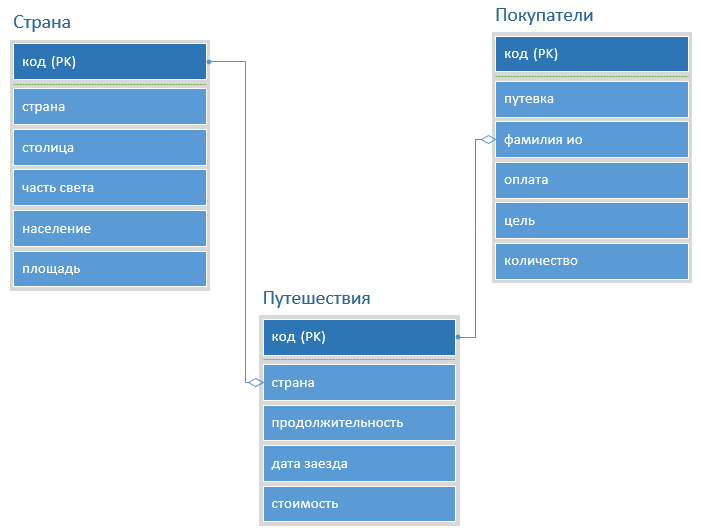
1. Запустите MS Visio.
2. В категориях шаблонов выберите *Базы данных* и в ней элемент *Нотация IDEF1X*. Нажмите кнопку *Создать*.
3. В соответствии с анализом предметной области необходимо добавить на рабочий лист три фигуры Сущность.
4. Для добавления текста фигуре необходимо дважды щелкнуть по ней левой кнопкой мыши.



1. Далее необходимо добавить связи между сущностями.

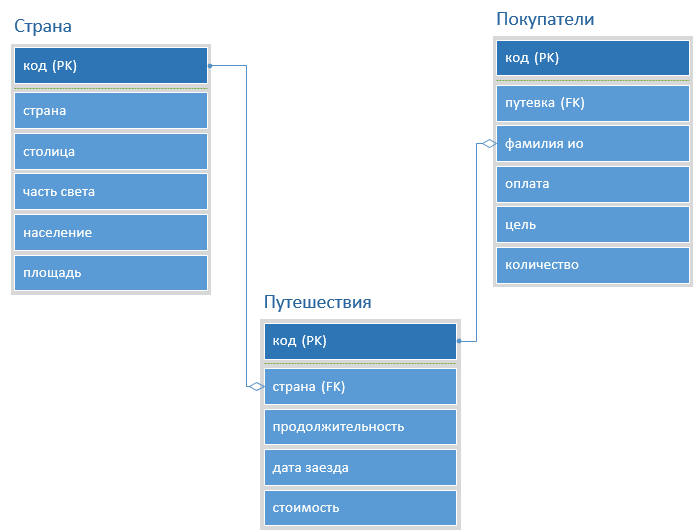


1. Добавить имена атрибутам (и сами атрибуты при необходимости).



1. Указать атрибуты, которые являются внешними ключами

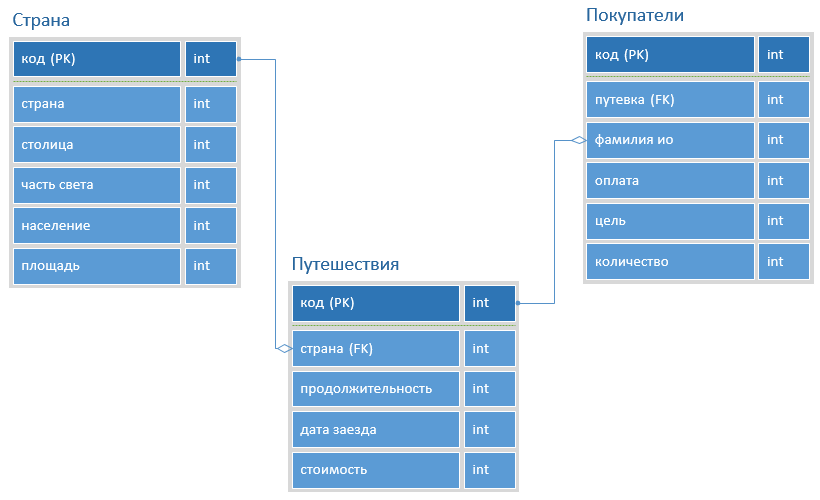
Для этого нужно выделить атрибут. Щелкнуть по нему правой кнопкой мыши и в контекстном меню выбрать пункт «Задать внешний ключ».



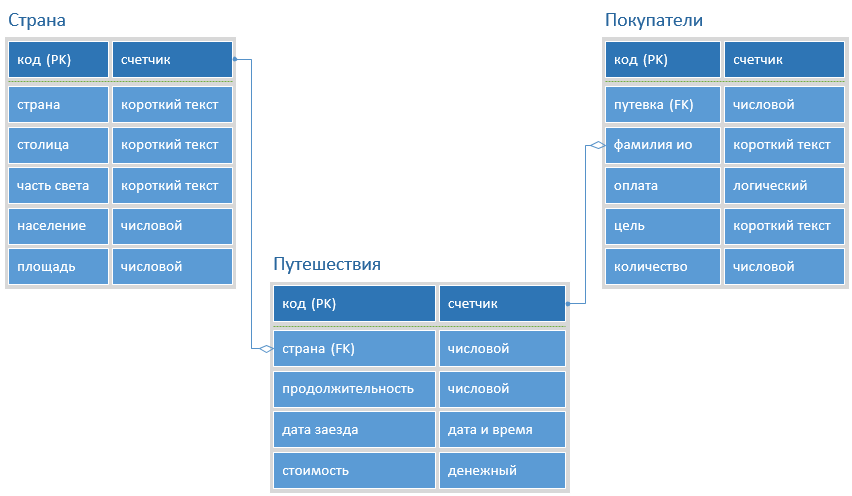
1. Создание физической модели

Для этого необходимо дублировать созданную модель на новый лист. Для этого щелкнуть правой кнопкой мыши по текущему листу и в контекстном меню выбрать пункт Дублировать.

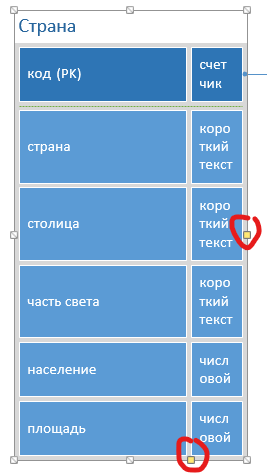
1. Затем на новом листе по очереди выбрать каждую сущность и в ее контекстном меню выбрать пункт Показать типы атрибутов.



1. Затем необходимо указать соответствующие типы данных, в зависимости от СУБД в которой будет выполняться реализация системы. Например, для MS Access модель примет вид:



Чтобы изменить размер ячейки, в которой указывается тип данных нужно выделить сущность и используя желтые точки изменить размер.



**Варианты**

Построить ER-модель для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

**Контрольные вопросы**

1. Элементы диаграммы в нотации IDEF1X.
2. Для чего предназначена диаграмма «сущность-связь»?
3. Что представляет собой нормализация?
4. В чем разница между логическим уровнем модели данных и физическим?

# Практическая работа №9. Функциональное моделирование IDEF0

**Цель:** получение навыков создания и редактирования функциональных моделей в нотации IDEF0.

**Теоретические сведения**

***1. Основные сведения по методологии IDEF0***

Модель в нотации IDEF0 представляет собой совокупность иерархически упорядоченных и взаимосвязанных диаграмм. Каждая диаграмма является единицей описания системы и располагается на отдельном листе.

**Цель моделирования.** Модель не может быть построена без четко сформулированной цели. Пример цели: «Описать функциональность предприятия с целью написания спецификаций ИС».

**Точка зрения.** Точку зрения можно представить как взгляд человека, который видит систему в нужном для моделирования аспекте. Как правило, выбирается точка зрения человека, ответственного за моделируемую работу в целом. Цель и точка зрения документируются.

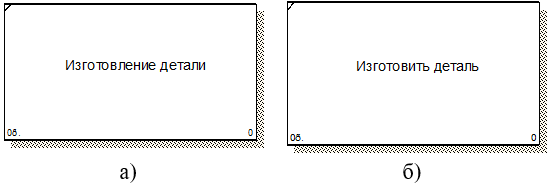
**Основные элементы IDEF0-модели**

В основе методологии IDEF0 лежат 4 основных понятия:

* функциональный блок;
* интерфейсная дуга (стрелка);
* декомпозиция;
* глоссарий.

1. **Функциональный блок**

Функциональные блоки обозначают поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты. Графически функциональные блоки изображаются в виде прямоугольников. Все блоки должны быть названы и определены. Имя функционального блока должно быть выражено сочетанием отглагольного существительного, обозначающего процесс, или глаголом:



Определение функционального блока заносится в глоссарий или словарь работ (Activity Dictionary).

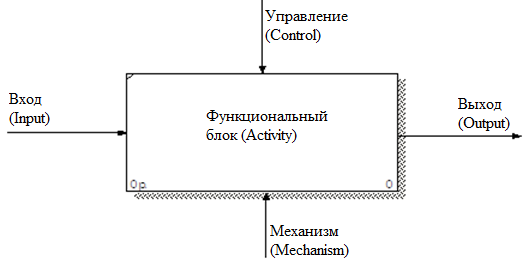
Все функциональные блоки модели нумеруются. Номер состоит из префикса и числа. Может использоваться префикс любой длины, но обычно используется префикс А. Контекстная (корневая) работа (функциональный блок) имеет номер А0.

1. **Интерфейсная дуга (стрелка – Arrow)**

Взаимодействие функциональных блоков с внешним миром и между собой описывается в виде интерфейсных дуг (стрелок). Стрелки представляют собой некую информацию и обозначаются существительными (например, «Заготовка», «Изделие») или именуемыми сочетаниями (например, «Готовое изделие»). Все стрелки должны быть определены. Определения заносятся в словарь стрелок – глоссарий (Arrow Dictionary).

**В IDEF0 различают 4 типа стрелок.**

Каждая стрелка имеет свое расположение относительно функционального блока.



Вход (Input) – материал или информация, которые используются или преобразуются работой для получения результата (выхода). Стрелка Input рисуется входящей в левую грань работы.

Управление (Control) – правила, стратегии, процедуры или стандарты, которыми руководствуется работа. Каждая работа должна иметь хотя бы одну стрелку управления. Рисуется как входящая в верхнюю грань работы.

Выход (Output) – материал или информация, которые производятся работой. Каждая работа должна иметь хотя бы одну стрелку выхода. Работа без результата не имеет смысла и не должна моделироваться. Изображается исходящей из правой грани работы.

Механизм (Mechanism) – ресурсы, которые выполняют работу, например, персонал предприятия, станки, устройства и т.д. Рисуется как входящая в нижнюю грань работы.

1. **Глоссарий**

Набор определений, ключевых слов и т.д., которые характеризуют каждый объект модели.

1. **Декомпозиция**

Разбиение системы на крупные фрагменты – функции, функции – на подфункции и т.д. до конкретных процедур.

**Модель может содержать 4 типа диаграмм:**

* контекстную (в каждой модели может быть только 1 контекстная диаграмма);
* декомпозиции;
* дерева узлов;
* только для экспозиции (FEO).

Контекстная диаграмма является вершиной древовидной структуры диаграмм и представляет собой общее описание системы и ее взаимодействия с внешней средой.

После описания системы в целом проводится разбиение ее на крупные фрагменты. Этот процесс называется функциональной декомпозицией, а диаграммы, которые описывают каждый фрагмент и взаимодействие фрагментов – диаграммами декомпозиции. После декомпозиции контекстной диаграммы проводится декомпозиция каждого большого фрагмента системы на более мелкие и т.д., до достижения нужного уровня подробности описания.

Диаграмма дерева узлов показывает иерархическую зависимость работ, но не взаимосвязи между работами.

Диаграммы для экспозиции (FEO) строятся для иллюстрации отдельных фрагментов модели, для иллюстрации альтернативной точки зрения либо для специальных целей.

Все диаграммы имеют нумерацию. Контекстная диаграмма имеет номер А-0, декомпозиция контекстной диаграммы – номер А), остальные диаграммы-декомпозиции – номера по соответствующему узлу (например, А1, А2, А21 и т.д.).

**Методика выполнения**

В качестве примера рассматривается процесс выполнения студентом курсовой работы (курсового проекта).

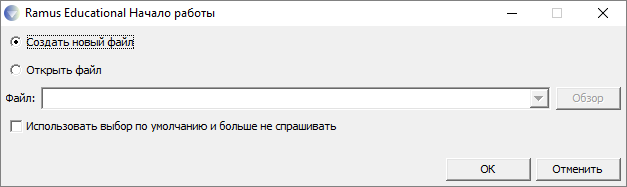
Программное обеспечение «Ramus» предназначено для использования в проектах, в которых необходимо описание бизнес-процессов предприятия. «Ramus» поддерживает методологии моделирования бизнес-процессов IDEF0 и DFD, а также имеет ряд дополнительных возможностей, призванных удовлетворить потребности команд разработчиков систем управления предприятиями.

«Ramus» обладает гибкими возможностями построения отчетности по графическим моделям, позволяющие создавать отчеты в форме документов, регламентирующих деятельность предприятия.

Ramus Educational имеет достаточно интуитивный интерфейс пользователя, позволяющий быстро и просто создавать сложные модели.

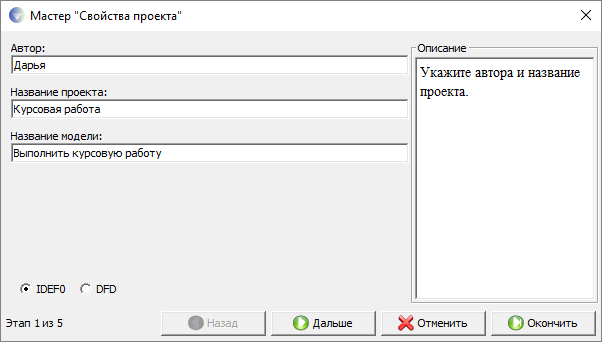
***1 Начало работы***

1. Запустите программу Ramus Educational. В появившемся окне предлагается создать новый проект или открыть уже существующий.



1. После нажатия на кнопку «ОК» осуществляется запуск мастера проекта.

* На первом шаге (рис. 5) в соответствующие поля необходимо внести сведения об авторе, названии проекта и модели, а также выбрать тип нотации модели (IDEF0 или DFD).



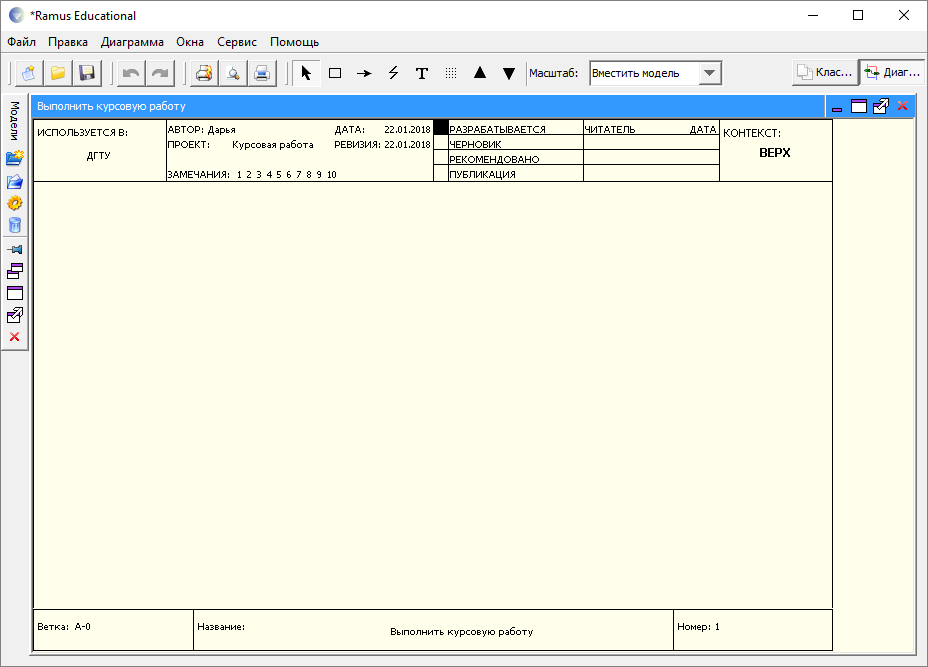
* На втором шаге вводится название организации, использующей данный проект.
* На третьем – дается краткое описание будущего проекта.
* Четвертый шаг позволяет создать несколько основных классификаторов (в данном случае можно пропустить этот шаг). Так как модели процессов реальных предприятий могут содержать значительное количество объектов (документы, персонал, функции и т.д.), то в Ramus предусмотрена возможность упорядочено хранить информацию об этих объектах в виде системы классификаторов. Классификация объектов упрощает поиск и обработку информации об объектах модели, а также и об объектах непосредственно не представленных на диаграммах процессов, но относящихся к процессам предприятия.
* На пятом, заключительном, предлагается выбрать те из созданных классификаторов, элементы которых будут содержаться в перечне собственников процессов (пропустить данный шаг).

При необходимости можно завершить работу мастера, нажав кнопку «Окончить».

После завершения работы мастера, откроется рабочее пространство «Диаграммы», в котором можно приступить к рисованию графической модели (рис. 6). В верхней части приводятся сведения о проекте, введенные пользователем посредством мастера диаграмм.

Программа Ramus Educational обладает гибким графическим интерфейсом, который можно настроить под нужды и предпочтения конкретного пользователя: ненужные окна можно закрыть/свернуть; можно менять их размеры и месторасположение; также можно группировать два и более окон в одном, при этом содержимое вложенных окон будет размещено на вкладках общего окна (данный функционал возможен не для всех комбинаций окон).

1. Сохраните созданную модель, выбрав опцию меню «Файл» – «Сохранить как».



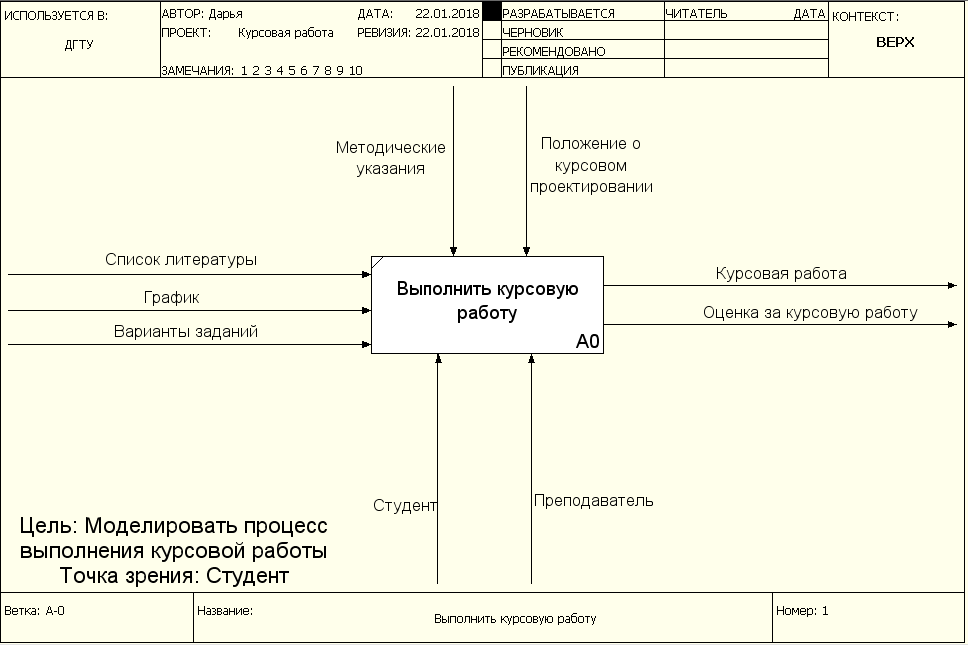
***2 Создание контекстной диаграммы***

1. На панели инструментов выберите пиктограмму функции () и мышью укажите месторасположение на рабочем пространстве.
2. Дайте данному функциональному блоку имя «Выполнить курсовую работу». Для этого дважды щелкните внутри блока.
3. Используя пиктограмму панели инструментов , создайте стрелки на контекстной диаграмме согласно Таблица 1.

Таблица 1 – Контекстная диаграмма

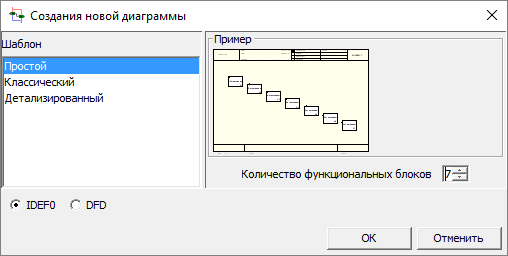
|  |  |  |
| --- | --- | --- |
| **Имя стрелки (Arrow Name)** | **Определение стрелки (Arrow Definition)** | **Тип стрелки (Arrow Type)** |
| График | График консультаций и сроки сдачи | Input |
| Список литературы | Источники информации для выполнения курсовой работы | Input |
| Варианты заданий | Список заданий на курсовую работу, подлежащий распределению между студентами | Input |
| Методические указания | Документ, содержащий указания по выполнению курсовой работы, описывающий содержание ее частей и основные требования | Control |
| Положение о курсовом проектировании | Документ, отражающий организационные требования по выполнению и сдаче курсовой работы | Control |
| Курсовая работа | Документ, являющийся основанием для получения оценки | Output |
| Оценка за курсовую работу | Результат выполнения курсовой работы | Output |
| Студент | Тот, кто выполняет курсовую работу | Mechanism |

1. В результате должна получиться контекстная диаграмма.

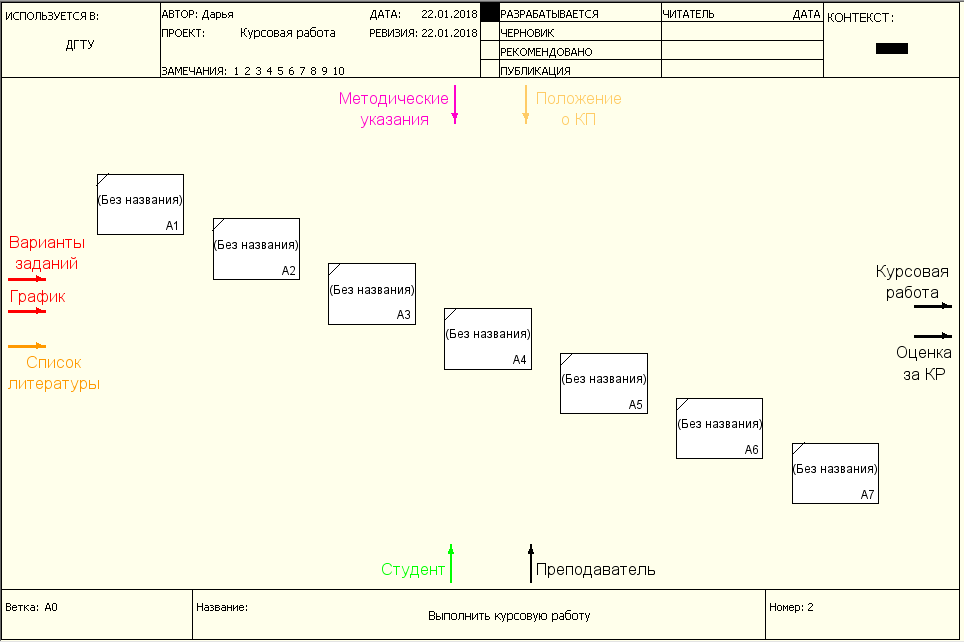


***3 Создание диаграммы декомпозиции***

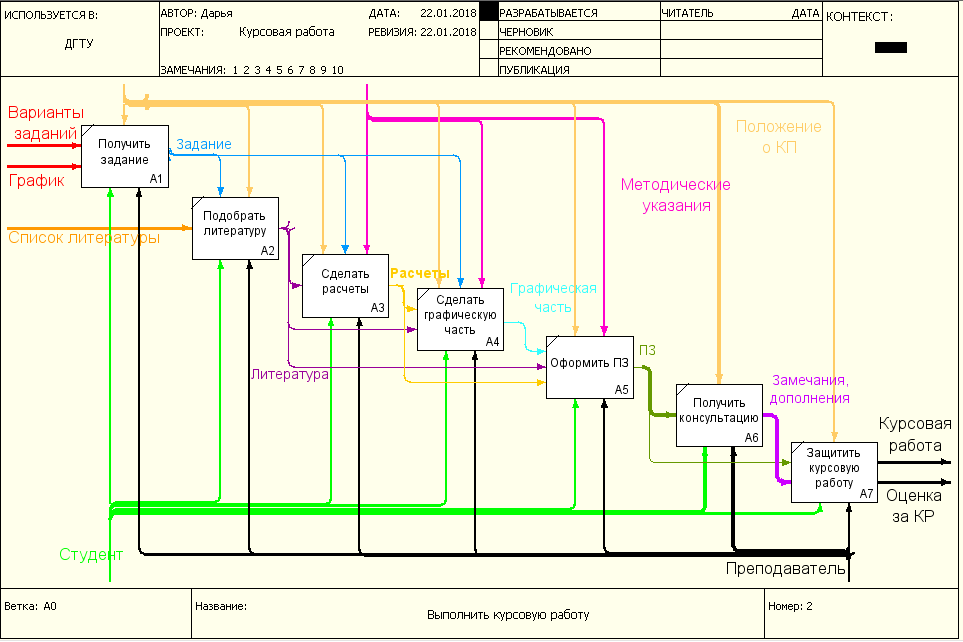
1. Выберите в палитре инструментов кнопку перехода на нижний уровень , в диалоговом окне «Создание новой диаграммы» установите количество функциональных блоков 7, укажите тип диаграммы (IDEF0) и нажмите кнопку ОК.



1. Автоматически будет создана диаграмма первого уровня декомпозиции с перенесенными в нее потоками родительской диаграммы.

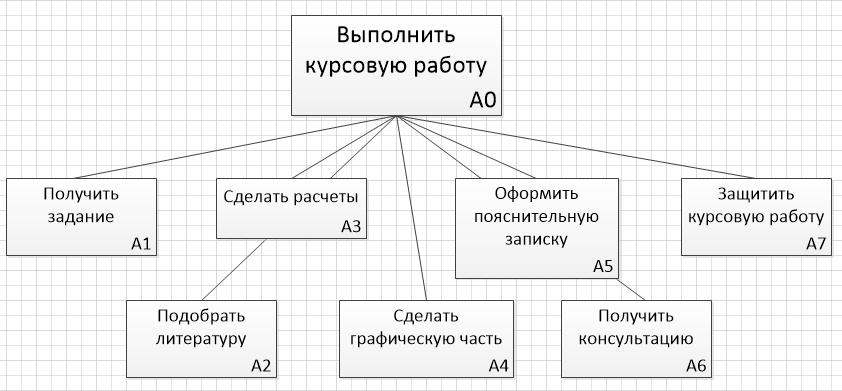


1. Выделите первую работу (функциональный блок), затем двойным щелчком мыши или, выбрав в контекстном меню пункт «Редактировать активный элемент», откройте окно свойств и внесите имя работы. Повторите операцию для оставшихся работ.
2. Выделив необходимый поток (стрелку) и, удерживая левую клавишу мыши, соедините его требуемым образом (через вход, управление, механизм или выход) с соответствующим функциональным блоком. В результате должна получиться детализирующая диаграмма.



**3. Создание дерева узлов**

Дерево узлов – это диаграмма, отображающая иерархию работ процесса. Ее необходимо построить с помощью Visio.



Для построения диаграммы:

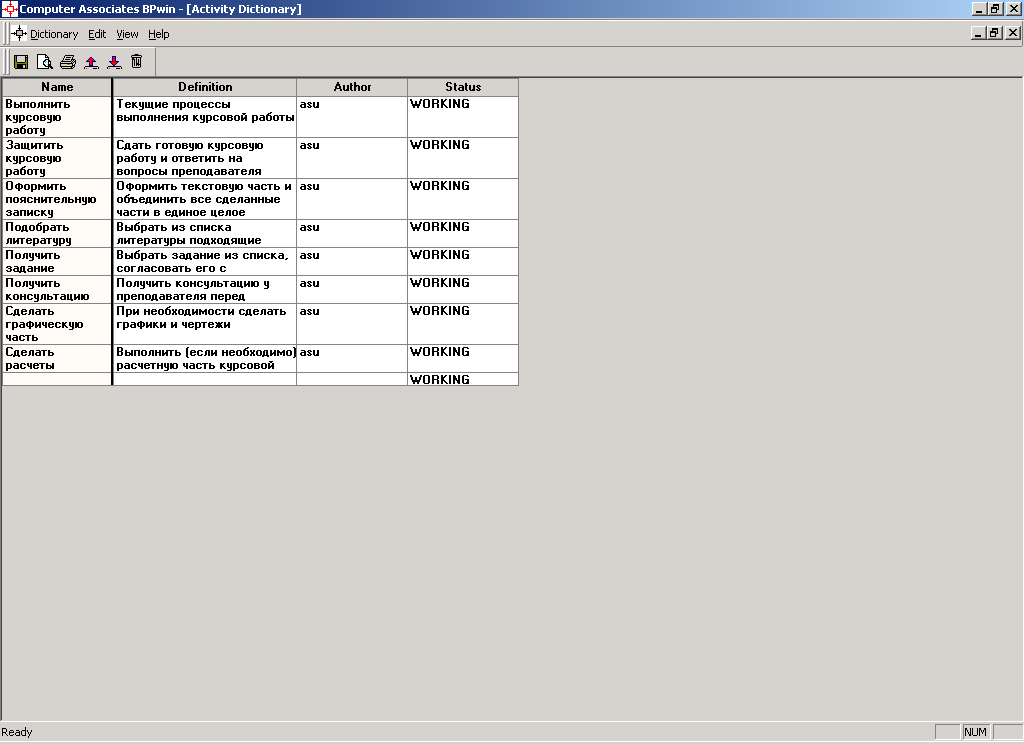
* создайте новую страницу;
* присвойте имя странице: дерево узлов;
* постройте дерево узлов, используя фигуры схемы IDEF0.

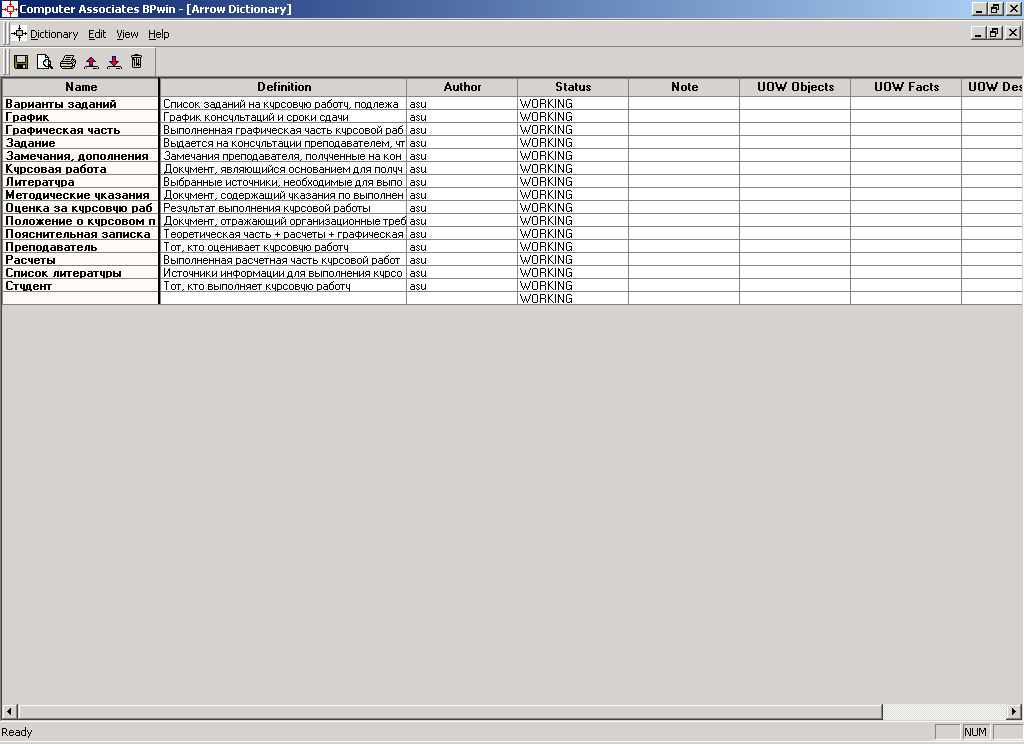
**4. Создание глоссария**

Глоссарий – это словарь ключевых слов, повествований, изложений, используемых при описании процесса.

Для построения глоссария:

* создайте документ Microsoft Office Word;
* создайте 2 таблицы: описание работ процесса, описание интерфейсных дуг процесса;
* наименование столбцов таблиц: имя (работы/дуги, описание);
* заполните таблицы в соответствии с ранее разработанной моделью процесса.





**Варианты**

Построить функциональную модель для следующих предметных областей

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

**Требования к построению модели**

1. На контекстной диаграмме необходимо указать точку зрения и цель моделирования.
2. Количество блоков любой декомпозиции не менее 3-х и не более 9.
3. Количество декомпозиций – 3 уровня декомпозиции.

**Контрольные вопросы**

1. Каковы цели функционального моделирования?
2. Назовите основные компоненты функциональной модели.
3. Какие виды интерфейсных дуг различают в IDEF0?
4. Для чего нужна цель и точка зрения?
5. Что такое функциональный блок?
6. Какие виды диаграмм может содержать функциональная модель?

# Практическая работа №10. Моделирование потоков данных DFD

**Цель:** изучение основных характеристик и основ работы с DFD-моделями в графическом редакторе.

**Теоретические сведения**

***1. Модель потоков данных***

DFD – data flow diagrams – диаграммы потоков данных – методология графического структурного анализа, описывающая внешние по отношению к системе источники и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ.

Диаграмма потоков данных – один из основных инструментов структурного анализа и проектирования информационных систем, существовавших до широкого распространения UML.

Для описания диаграмм DFD используются две нотации – Йордана (Yourdon) и Гейна-Сарсона (Gane-Sarson), отличающиеся синтаксисом.

***2. Основные элементы информационной модели логического уровня***

Согласно DFD источники информации (внешние сущности) порождают информационные потоки (потоки данных), переносящие информацию к подсистемам или процессам. Те в свою очередь преобразуют информацию и порождают новые потоки, которые переносят информацию к другим процессам или подсистемам, накопителям данных или внешним сущностям – потребителям информации.

При построении диаграмм различают элементы графической нотации, представленные в табл. 1.

Таблица 1 – Элементы графической нотации DFD

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Нотация Йордана** | **Нотация Гейна-Сарсона** |
| Поток данных |  |  |
| Процесс (система, подсистема) |  |  |
| Накопитель данных |  |  |
| Внешняя сущность |  |  |

**Поток данных** определяет информацию (материальный объект), передаваемую через некоторое соединение от источника к приемнику. Реальный поток данных может быть информацией, передаваемой по кабелю между двумя устройствами, пересылаемыми по почте письмами, магнитными лентами или дискетами, переносимыми с одного компьютера на другой и т. д.

Каждый поток данных имеет имя, отражающее его содержание. Направление стрелки показывает направление потока данных. Иногда информация может двигаться в одном направлении, обрабатываться и возвращаться назад в ее источник. Такая ситуация может моделироваться либо двумя различными потоками, либо одним – двунаправленным.

На диаграммах IDEF0 потоки данных соответствуют входам и выходам, но в отличие от IDEF0 стрелки потоков на DFD могут отображаться входящими и выходящими из любой грани внешней сущности, процесса или накопителя данных.

**Процесс** (в IDEF0 – функция, работа) представляет собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом.

Каждый процесс должен иметь имя в виде предложения с глаголом в неопределенной форме (вычислить, рассчитать, проверить, определить, создать, получить), за которым следуют существительные в винительном падеже, например:

* «Ввести сведения о клиентах»;
* «Рассчитать допускаемую скорость»;
* «Сформировать ведомость допускаемых скоростей».

Номер процесса служит для его идентификации и ставится с учетом декомпозиции. Вложенность процессов обозначается через точку.

Преобразование информации может показываться как с точки зрения процессов, так и с точки зрения систем и подсистем. Если вместо имени процесса «Рассчитать допускаемую скорость» написать «Подсистема расчета допускаемых скоростей», тогда этот блок на диаграмме стоит рассматривать, как подсистему.

**Накопитель (хранилище) данных** представляет собой абстрактное устройство для хранения информации, которую можно в любой момент поместить в накопитель и через некоторое время извлечь, причем способы помещения и извлечения могут быть любыми.

Накопитель данных может быть реализован физически в виде ящика в картотеке, области в оперативной памяти, файла на магнитном носителе и т.д.

Накопителю обязательно должно даваться уникальное имя и номер в пределах всей модели (всего набора диаграмм). Имя накопителя выбирается из соображения наибольшей информативности для разработчика. Например, если в качестве накопителей выступают таблицы проектируемой базы данных, тогда в качестве имен накопителей рекомендуется использовать имена таблиц. Таким образом, накопитель данных может представлять собой всю базу данных целиком, совокупность таблиц или отдельную таблицу. Такое представление накопителей в дальнейшем облегчит построение информационной модели системы.

**Внешняя сущность** (терминатор) представляет собой материальный объект или физическое лицо, выступающие как источник или приемник информации (например, заказчики, персонал, программа, склад, инструкция). Внешние сущности на DFD по смыслу соответствуют управлению и механизмам, отображаемым на контекстной диаграмме IDEF0.

Определение некоторого объекта, субъекта или системы в качестве внешней сущности указывает на то, что она находится за пределами границ проектируемой информационной системы. В связи с этим внешние сущности, как правило, отображаются только на контекстной диаграмме DFD. В процессе анализа и проектирования некоторые внешние сущности могут быть перенесены на диаграммы декомпозиции, если это необходимо, или, наоборот, часть процессов (подсистем) может быть представлена как внешняя сущность.

**Рекомендации по выполнению**

На основе модели предметной области, разработанной в практических занятиях №1 и 2, составить функциональную модель в нотации DFD.

Отчет по практическому занятию выполняются в формате MS Word, который содержит экранные формы моделей согласно заданию.

**Методика выполнения практического занятия**

В качестве примера рассматривается процесс выполнения студентом курсовой работы (курсового проекта).

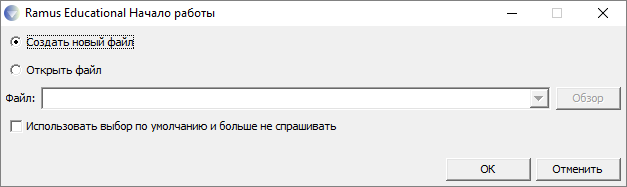
Программное обеспечение «Ramus» предназначено для использования в проектах, в которых необходимо описание бизнес-процессов предприятия. «Ramus» поддерживает методологии моделирования бизнес-процессов IDEF0 и DFD, а также имеет ряд дополнительных возможностей, призванных удовлетворить потребности команд разработчиков систем управления предприятиями.

«Ramus» обладает гибкими возможностями построения отчетности по графическим моделям, позволяющие создавать отчеты в форме документов, регламентирующих деятельность предприятия.

Ramus Educational имеет достаточно интуитивный интерфейс пользователя, позволяющий быстро и просто создавать сложные модели.

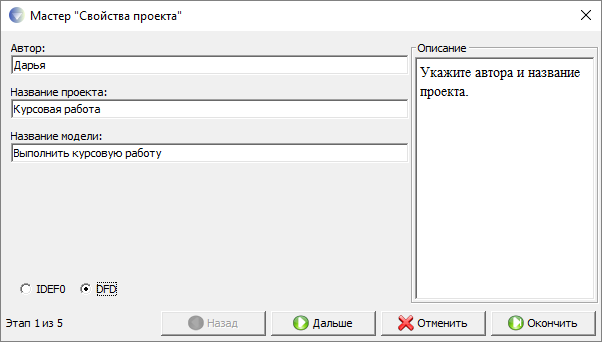
***1 Начало работы***

1. Запустите программу Ramus Educational. В появившемся окне предлагается создать новый проект или открыть уже существующий.

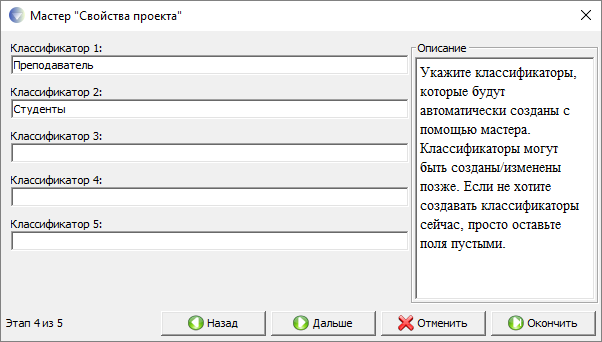


1. После нажатия на кнопку «ОК» осуществляется запуск мастера проекта.

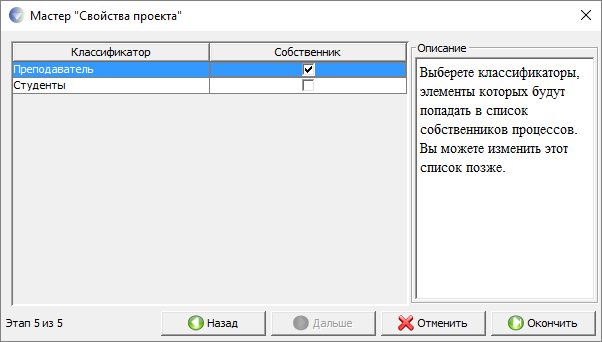
* На первом шаге в соответствующие поля необходимо внести сведения об авторе, названии проекта и модели, а также выбрать тип нотации модели (IDEF0 или DFD) – в данном случае – DFD.



* На втором шаге вводится название организации, использующей данный проект.
* На третьем – дается краткое описание будущего проекта.
* Четвертый шаг позволяет создать несколько основных классификаторов

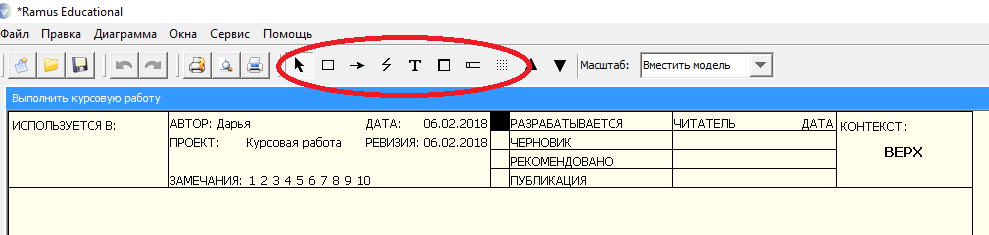


* На пятом, заключительном, предлагается выбрать те из созданных классификаторов, элементы которых будут содержаться в перечне собственников процессов.



При необходимости можно завершить работу мастера, нажав кнопку «Окончить».

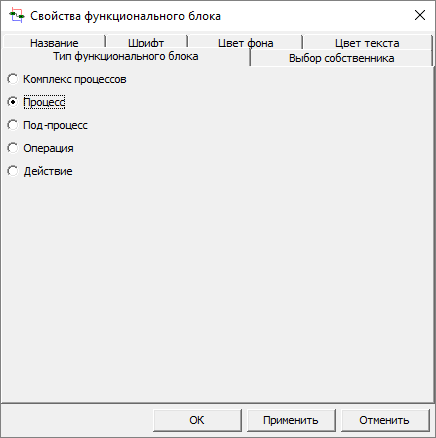
После завершения работы мастера, откроется рабочее пространство «Диаграммы», в котором можно приступить к построению графической модели. На панели инструментов, в верхней части окна рабочего пространства программы, содержатся элементы диаграммы потоков данных в нотации Gane-Sarson.



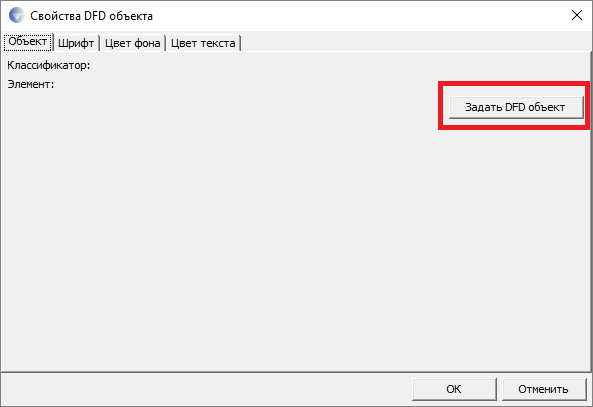
1. Сохраните созданную модель, выбрав опцию меню «Файл» – «Сохранить как».

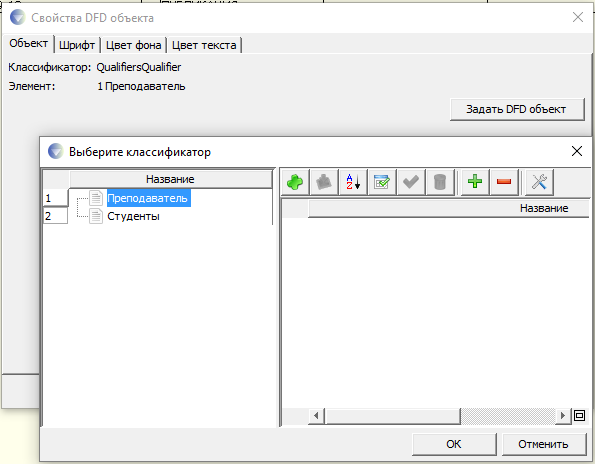
***2 Создание контекстной диаграммы***

1. На панели инструментов выберите инструмент создания процесса () и мышью укажите месторасположение на рабочем пространстве нового процесса.
2. Выделив процесс, выберите в контекстном меню опцию **«Редактировать активный элемент»**. В появившемся диалоговом окне на вкладке **«Название»** присвойте процессу имя «*Выполнить курсовую работу*»; на вкладке «**Тип функционального блока**» укажите тип элемента – «Процесс».

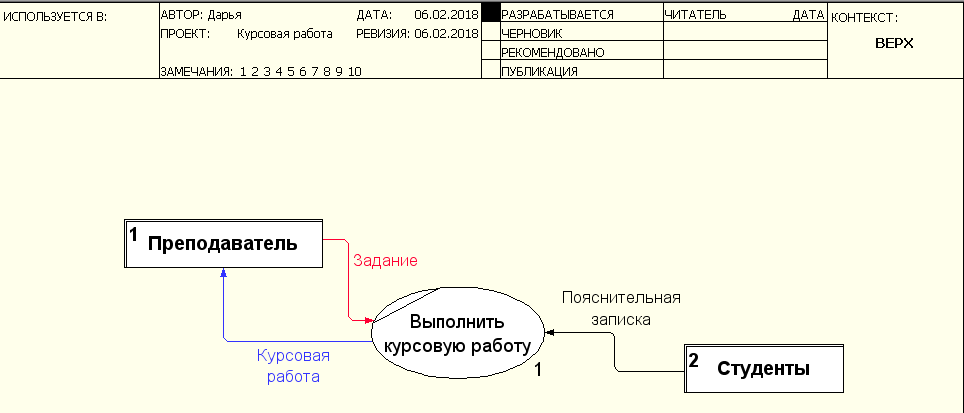


1. На панели инструментов выберите инструмент создания внешней сущности  и мышью укажите произвольное ее месторасположение в области построения.
2. В контекстном меню созданной внешней сущности выберите опцию **«Редактировать активный элемент»**, на вкладке **«Объект»** нажмите **«Задать DFD объект»**, после чего, в появившемся окне выделите классификатор *«Преподаватель»* и нажмите «ОК».
3. Повторяя действия предыдущего шага, добавьте внешнюю сущность *«Студенты»*.



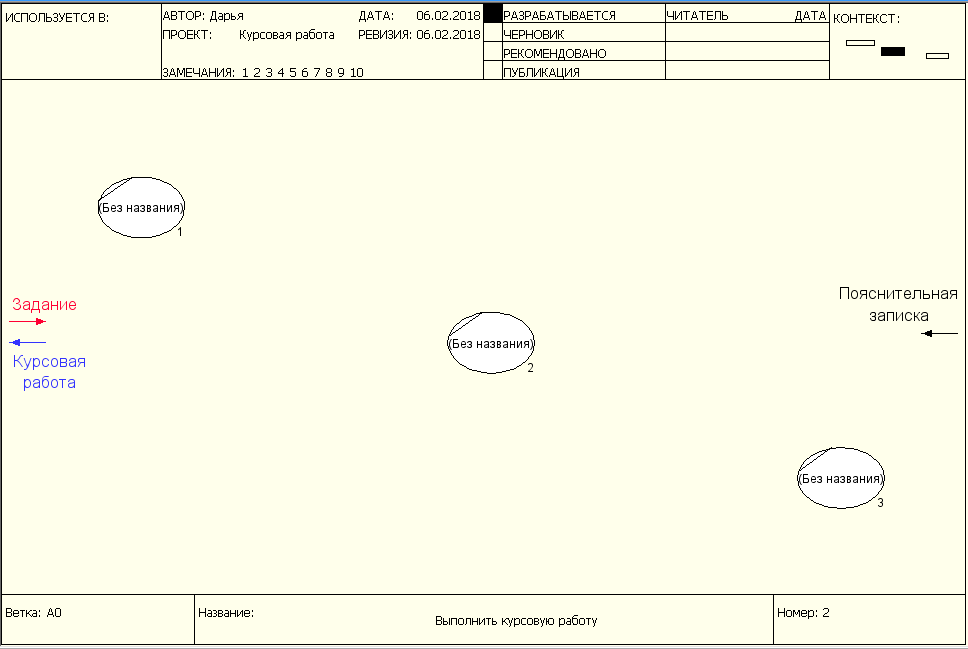


1. Выбрав на панели инструментов элемент , создайте стрелки на контекстной диаграмме. В результате должна получиться контекстная диаграмма, показанная на Рис. 9.



***3 Создание диаграммы декомпозиции***

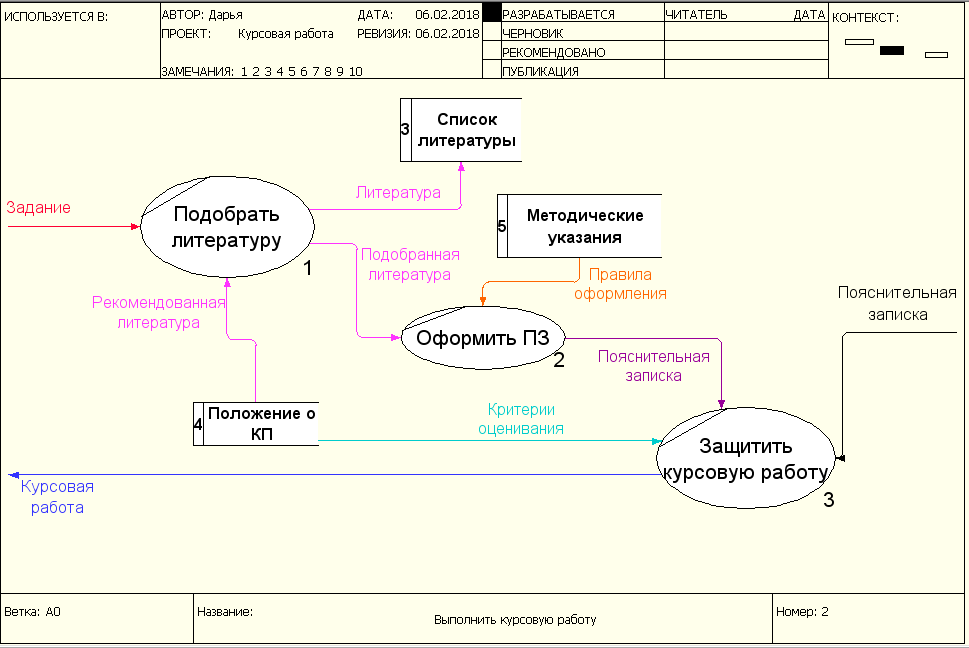
1. Выберите в палитре инструментов кнопку перехода на нижний уровень , в диалоговом окне **«Создание новой диаграммы»** установите количество функциональных блоков 3, укажите тип диаграммы (DFD) и нажмите кнопку ОК.
2. Автоматически будет создана диаграмма первого уровня декомпозиции с перенесенными в нее потоками родительской диаграммы.



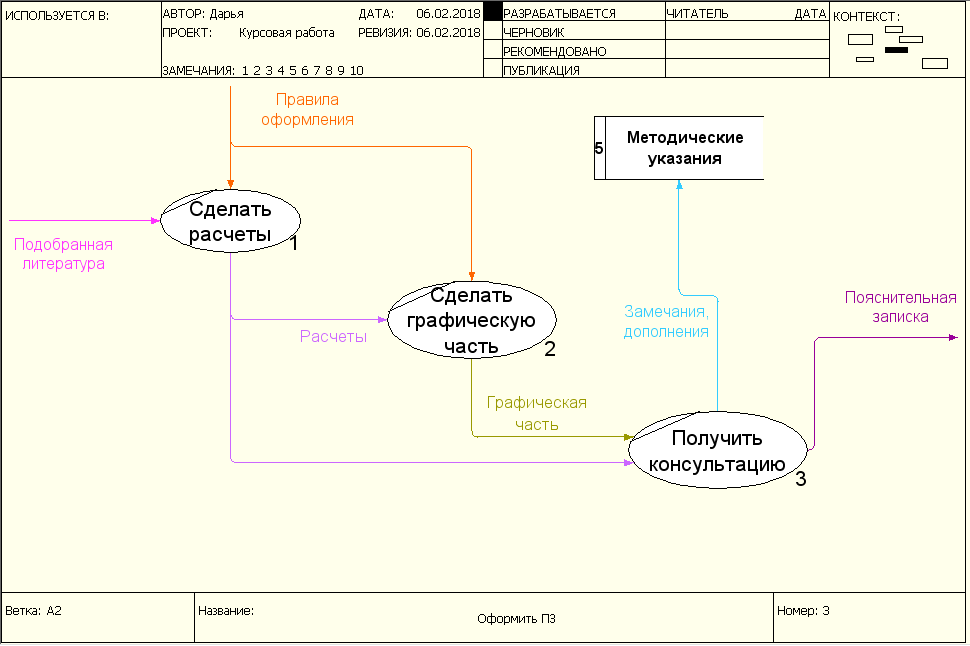
1. Двойным щелчком мыши на одном из процессов, или, выбрав в контекстном меню процесса пункт **«Редактировать активный элемент»**, откройте окно редактирования свойств и задайте процессу имя. Повторите операцию для оставшихся процессов.
2. Добавьте недостающие классификаторы для задания DFD объектов хранилищам данным. Для этого в меню выберите Окна -> Показать окно -> Классификаторы.

После этого нажмите на кнопку . Название классификатора можно ввести в созданную строку, дважды, медленно кликнув мышью по строке, или же нажав клавишу F2, предварительно выделив нужную строку мышью.

1. Добавьте хранилища данных, воспользовавшись кнопкой  палитры инструментов.
2. Выделив необходимый поток (стрелку) и, удерживая левую клавишу мыши, соедините его требуемым образом с соответствующим процессом. В результате должна получиться детализирующая диаграмма.



1. На основе описанных выше действий постройте диаграмму декомпозиций второго уровня для процесса *«Оформить ПЗ»*.



**Задание**

На основе модели предметной области, разработанной в практических занятиях №1 и 2, составить функциональную модель в нотации DFD.

Отчет по практическому занятию выполняется в формате MS Word, который содержит экранные формы моделей согласно заданию.

**Варианты**

Построить модель потоков данных для следующих предметных областей

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

**Требования к построению модели**

Количество блоков любой декомпозиции не менее 3-х и не более 9.

Количество декомпозиций – 3 уровня декомпозиции.

**Контрольные вопросы**

1. Каково назначение стандарта DFD?
2. В чем основные отличия стандартов IDEF0 и DFD?
3. Каким образом в MS Visio создается схема DFD? Какие для этого используются нотации?
4. Какова роль основных элементов в стандарте DFD?

# Практическая работа №11. Описание бизнес-процессов заданной предметной области

**Цель работы:** изучить методы описания бизнес-процессов с помощью языка UML

**Теоретические сведения**

Visual Paradigm for UML Community Edition – среда объектно-ориентированного проектирования на языке UML, распространяемая бесплатно для некоммерческого использования.

При первых запусках среда будет предлагать сообщить своё имя и e-mail для получения регистрационного кода. Получив код по электронной почте, следует активировать лицензию. Некоторые функции среды доступны только в её платных версиях. Начальное окно среды Visual Paradigm for UML Community Edition имеет вид, изображенный на рисунке 1.

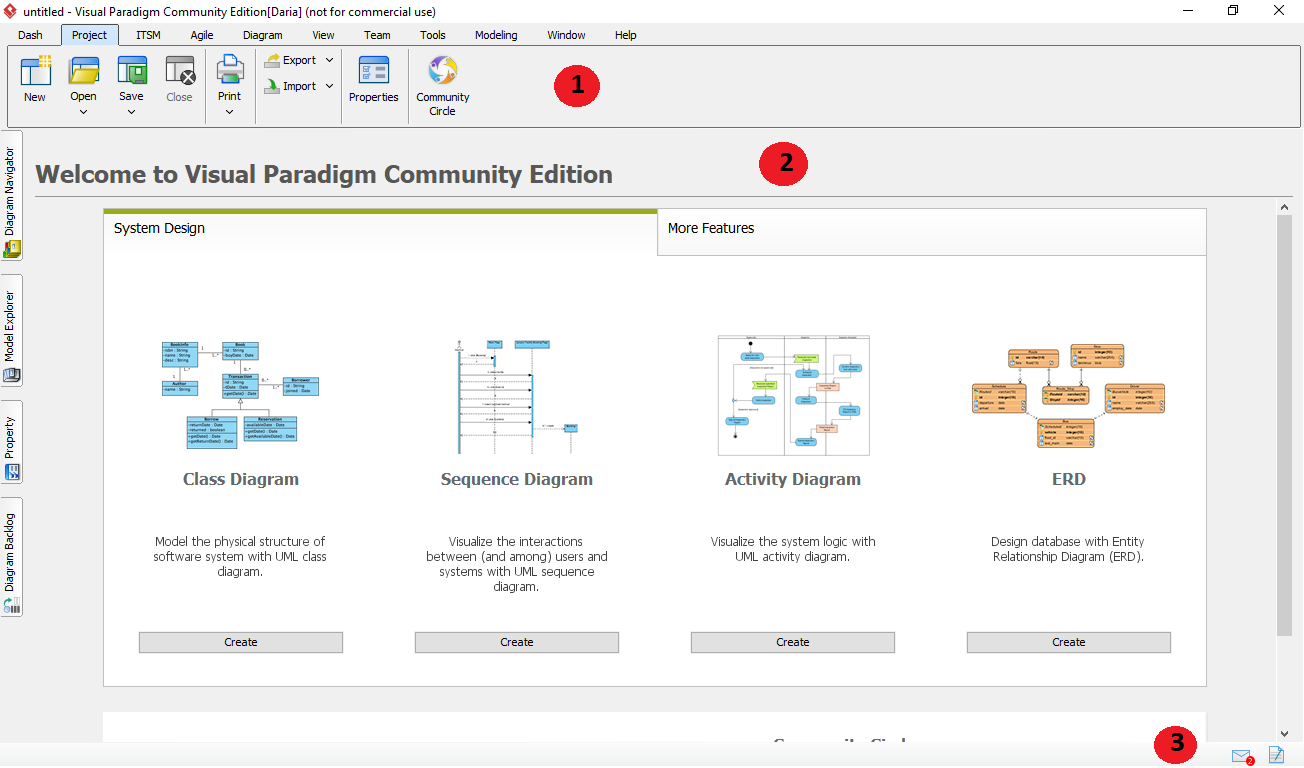


Рисунок 1 – Начально окно Visual Paradigm for UML CE

В таблице 1 приведено описание основных элементов интерфейса среды объектно-ориентированного проектирования Visual Paradigm for UML Community Edition.

Таблица 1 – Описание элементов интерфейса Visual Paradigm for UML CE

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Наименование** | **Описание** |
| 1 | Панель инструментов | Панель со вкладками, позволяющая выполнять основные операции по созданию проектов и работе с диаграммами в Visual Paradigm |
| 2 | Редактор диаграмм | Рабочая область, внутри которой отображается проектируемая диаграмма |
| 3 | Строка состояния | Строка, в которой отображаются уведомления |

***Сохранение и открытие проектов***

Для сохранения созданного проекта необходимо выбрать в панели инструментов *Project > Save* или *Project > Save as*… При первом сохранении проекта, среда проектирования предложит выбрать место для сохранения.

Для открытия ранее созданного проекта необходимо выбрать в панели инструментов *Project > Open* и указать пусть к файлу проекта, имеющему расширение *Visual Paradigm Project (\*.vpp)*.

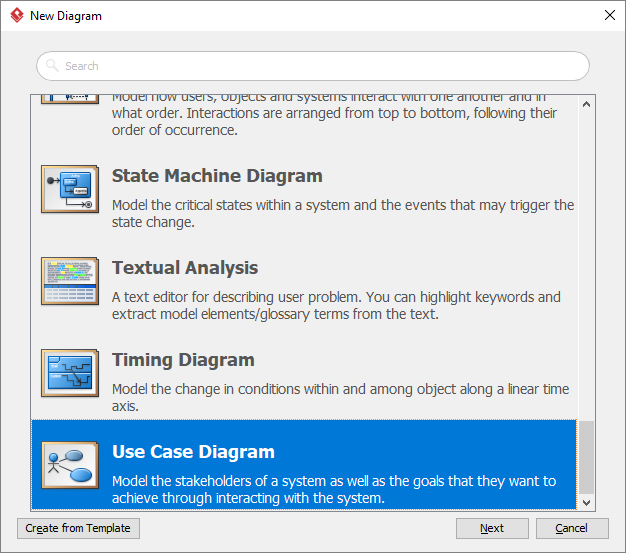
***Основы работы с диаграммами в Visual Paradigm for UML CE***

Далее описываются основные шаги по созданию диаграмм, добавлению фигур и установлению связей между ними.

1. Создание диаграмм

Рассмотрим в качестве примера процесс создания диаграммы вариантов использования. Для создания диаграммы прецедентов необходимо выполнить следующую последовательность действий.

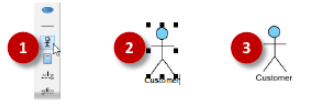
1. Выбрать *Diagram > New* а панели инструментов.
2. Начать набор названия диаграммы в строке поиске, в данном случае *use case diagram.*



1. Выбрать диаграмму в списке и нажать кнопку *Next.*
2. Задать имя диаграммы, место расположения и описание.
3. Нажать кнопку *Ok.*
4. Создание и соединение фигур
   1. Используя панель инструментов для работы с диаграммами

Далее рассмотрим процесс добавления фигуры *Актер* с помощью панели инструментов для работы с диаграммами.

1. Нажать на кнопку с изображением *Актера* в панели инструментов для работы с диаграммами.
2. Щелкнуть в рабочей области диаграммы чтобы создать актера и ввести его название.
3. Нажать в произвольном месте рабочей области либо нажать клавишу *Enter* на клавиатуре.



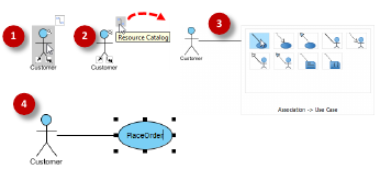
* 1. Используя каталог ресурсов

Переместив курсор мыши на фигуру в ее правом верхнем углу появляется значок, который предназначен для доступа к каталогу ресурсов.

Каталог ресурсов позволяет создавать новую фигуру, которая автоматически соединяется с существующей. Также его можно использовать для создания связи между двумя существующими фигурами.

Рассмотрим последовательность действий для создания варианта использования от существующего актера

1. Переместить курсор мыши на фигуру *Актер*.
2. Нажать на кнопку *Каталог ресурсов* в правом верхнем углу и потянуть указатель вправо.
3. Отпустить кнопку мыши и выбрать *Association > Use Case.*
4. Ввести название варианта использования.



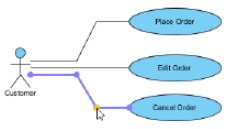
1. Изменение размера фигур

После нажатия на фигуру появляется несколько обработчиков изменения размера. Перетаскивая их можно изменять размер фигуры.

1. Добавление контрольных точек к соединениям

Для большинства диаграмм в качестве соединения по умолчанию используется «наклонное соединение», представляющее собой прямую линию. Добавление контрольных точек позволит изменить положение линии.

Для добавления контрольной точки необходимо навести курсор мыши на соединение нажать левую кнопку мыши и не отпуская ее переместить точку в нужное место.



***Примечание.*** Существует 5 типов соединений. Чтобы применить другой тип необходимо щелкнуть правой кнопкой мыши по линии и выбрать *Styles and Formatting > Connector Styles* и выбрать подходящий тип. Если необходимо изменить сразу все соединения нужно щелкнуть правой кнопкой мыши в свободном месте рабочей области и выбрать *Connectors* во всплывающем меню.

1. Изменение цвета фигуры

Для придания более выразительного внешнего вида диаграмме можно изменять цвет фигур.

Рассмотрим последовательность действий для изменения цвета варианта использования.

1. Щелкнуть правой кнопкой мыши по фигуре *Вариант использования* и в контекстном меню выбрать *Styles and Formatting > Formats…*
2. Открыть вкладку *Backgrounds* в окне *Formats.* Выбрать нужный цвет и нажать кнопку *Ok* чтобы применить выбранный цвет к фигуре.

***Экспорт диаграммы***

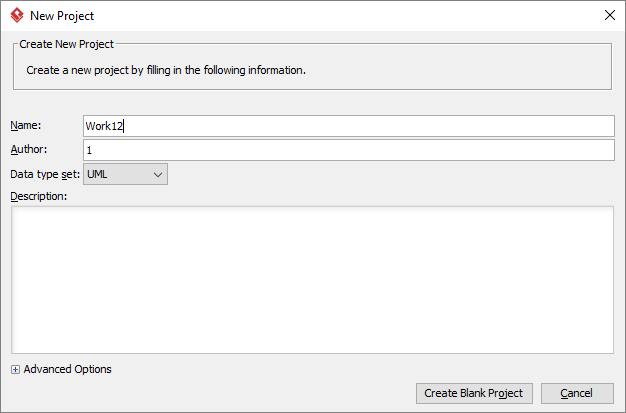
Готовую диаграмму можно полностью или частично копировать в формате изображения JDG или EMF в буфер обмена для экспорта в другое приложение.

Для этого следует выделить нужные объекты или всю диаграмму (Ctrl+A) и в контекстном меню выбрать команду *Copy > Copy to Clipboard as Image (JPG)*. Данная команда дублируется комбинацией клавиш Ctrl+Alt+C. После выполненных действий изображение диаграммы готово к вставке в другом приложении.

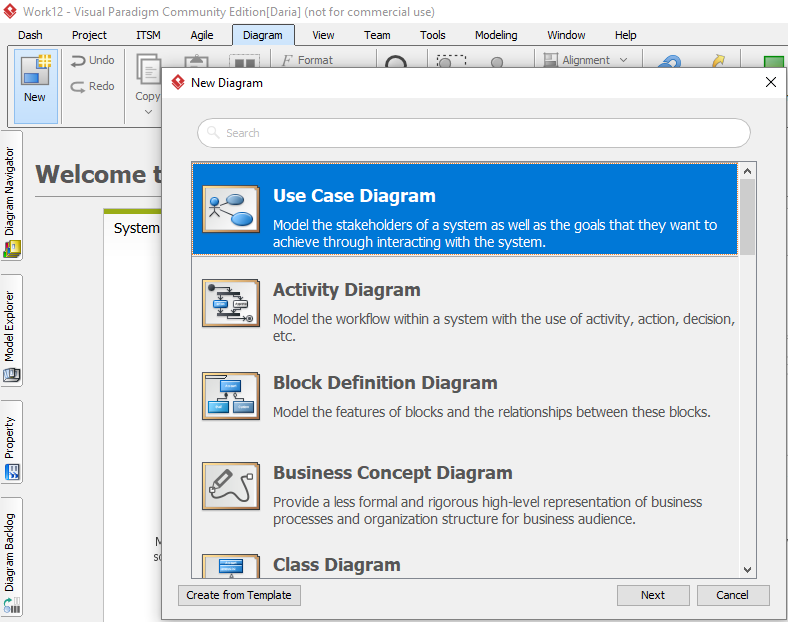
**Методика выполнения**

В качестве примера рассматривается моделирование системы продажи товаров по каталогу.

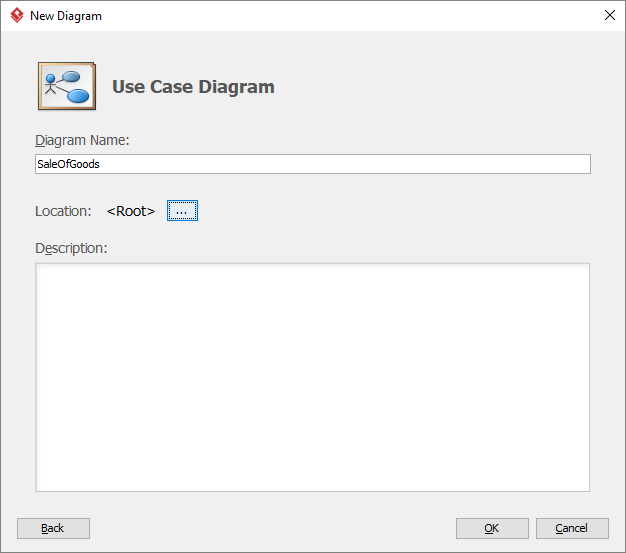
1. Запустите visual Paradigm for UML CE.
2. Создайте новый проект: *Project > New*. Введите название проекта и нажмите кнопку *Create Blank Project.*



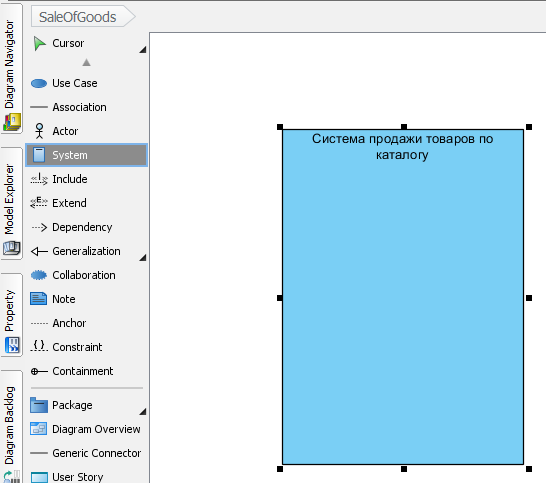
1. Создайте новую диаграмму вариантов использования (*Use Case Diagram)*: *Diagram > New*.



1. Присвойте имя диаграмме: *SaleOfGoods*.

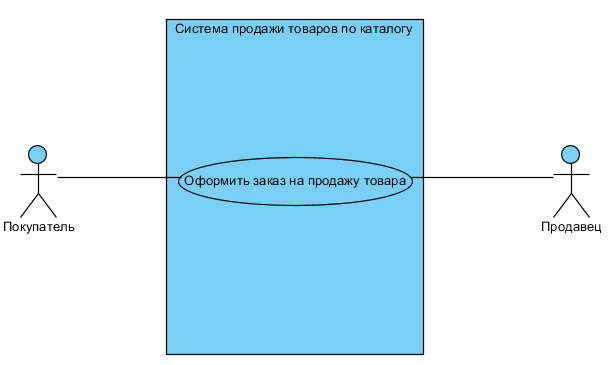


1. Добавьте фигуру *System.* Для этого в панели инструментов для работы с фигурами выберите нужную фигуру и щелкните в любом месте рабочей области. После этого переименуйте систему задав название: «Система продажи товаров по каталогу».

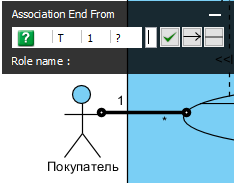


1. Далее добавьте актеров *Продавец* и *Покупатель* и вариант использования *Оформить заказ на покупку товара.*
2. Добавьте связь актеров с вариантом использования. Для этого: выделите актера > в правом верхнем углу откройте каталог ресурсов > в контекстном меню выберите use case > во вспомогательном окне начните вводить название ранее созданного прецедента «Оформить заказ на покупку товара» > нажмите кнопку Ok.

Результат выполнения.

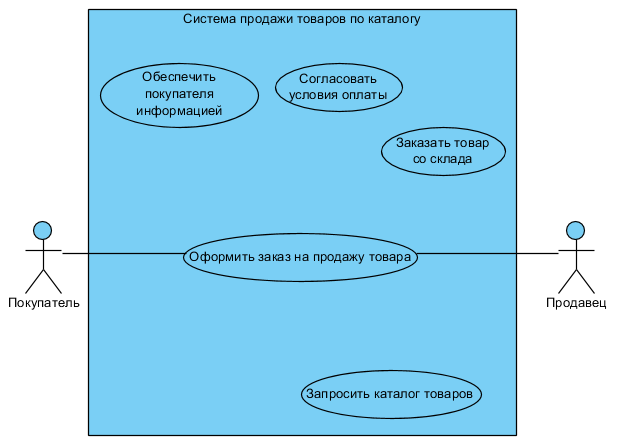


1. Далее необходимо установить кратность для отношения ассоциации между актерами и вариантом использования. Для этого необходимо навести курсор на связь и дважды щелкнуть левой кнопкой мыши по точке вначале линии. В появившемся контекстном окне необходимо задать кратность данной стороны отношения.

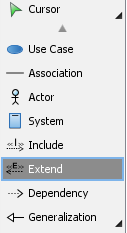


Для задания кратности второй стороне отношения нажмите кнопку со стрелкой вправо и проделайте те же действия. Для того, чтобы применить произведенные действия достаточно нажать кнопку с зеленой галочкой либо щелкнуть в любом пустом месте рабочей области.

1. Дополним исходную диаграмму. Для этого добавим 4 варианта использования «Обеспечить покупателя информацией», «Согласовать условия оплаты», «Заказать товар со склада» и «Запросить каталог товаров».

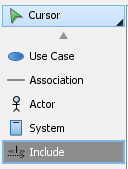


1. Установим связи между основным и дополнительными вариантами использования. «Запросить каталог товаров» связан с «Оформить заказ на продажу товара» отношением расширения. Для того чтобы установить этот тип соединения необходимо в панели инструментов для работы с диаграммами выбрать соответствующую фигуру.



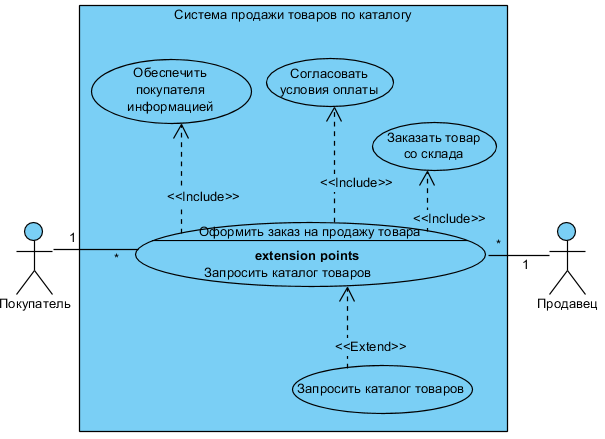
После чего соединить два варианта использования.

«Обеспечить покупателя информацией», «Согласовать условия оплаты», «Заказать товар со склада» связаны с «Оформить заказ на покупку товара» отношением включения. Для того чтобы установить этот тип соединения необходимо в панели инструментов для работы с диаграммами выбрать соответствующую фигуру.



После чего соединить варианты использования «Обеспечить покупателя информацией», «Согласовать условия оплаты», «Заказать товар со склада» с «Оформить заказ на покупку товара».

В результате выполнения всех вышеописанных действий диаграмма вариантов использования должна иметь вид.



**Задание**

Опишите бизнес-процессы предметной области с помощью UML в соответствии с вариантом

**Варианты**

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №12. Моделирование и запуск бизнес-процесса в Elma Community Edition

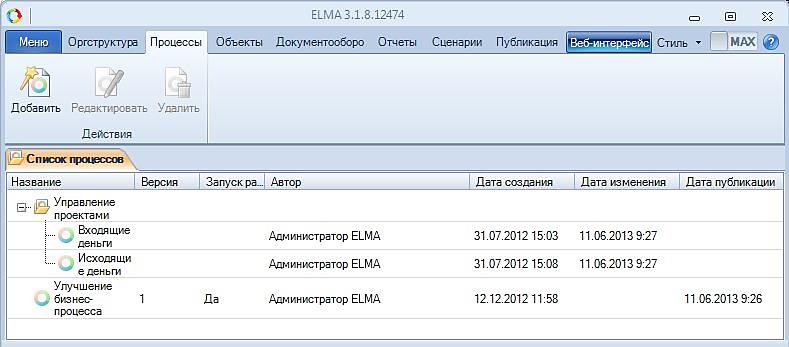
**Цель работы:** изучить методы описания бизнес-процессов с помощью CASE-средства ELMA

**Теоретические сведения**

**Создание и моделирование бизнес-процесса**

Дизайнер процессов ELMA позволяет визуально моделировать бизнес-процессы. Для моделирования используется язык описания BPMN.

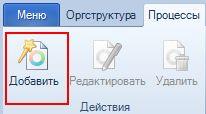
Создание бизнес-процесса осуществляется в Дизайнере ELMA – для это­го служит вкладка Процессы программы. Вкладка Процессы содержит вло­женные вкладки, первая из которых – Список процессов – содержит полный перечень процессов, созданных для данной конфигурации. В пустой конфи­гурации процессов пока еще нет.



Список процессов представлен в виде древа. В качестве ветвей дерева вы­ступают группы процессов – это своего рода папки, которые предназначены для хранения логически связанных бизнес-процессов. Рядом с именами па­пок находится значок в виде желтой папки, привычный пользователям опе­рационных систем семейства Windows.

Создать новый бизнес-процесс можно одним из следующих способов:

* Щелкнуть мышью по кнопке Добавить панели инструментов.



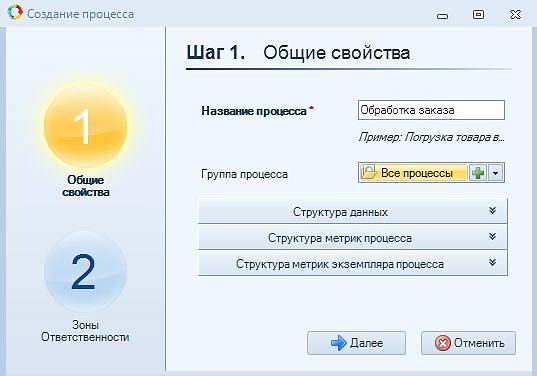
* Щелкнуть мышью по свободному участку списка процессов. В по­явившемся контекстном меню выбрать пункт Создать процесс.

Смоделируем бизнес-процесс «Обработка заказа», который может реализовываться в рамках продажи и аренды велосипедов через интернет-магазин

1. Стартовым событием процесса-примера является звонок или письмо от клиента на сайт компании (интернет-магазина).
2. Далее Офис-менеджер должна Зарегистрировать и обработать заявку
3. Дальше Сотрудник отдела продаж если заявка на аренду, то оформляет Заявку на аренду, если на покупку, то оформляет Заявку на покупку
4. После чего Сотрудник отдела продаж Проверяет наличие товара на складе
5. Если заказанный товар на складе есть, то Сотрудник отдела продаж бронирует товар на складе,
6. Либо, в случае отсутствия на складе, одновременно (предполагает использование параллельного шлюза) ^ Сотрудник отдела продаж делает заказ на фабрику-производителя и информирует клиента о задержке по выдаче товара. Ждем пока завершатся два этих действия (предполагает использование закрывающего параллельного шлюза) Далее приостанавливаем ход процесса пока не придёт заказанный с фабрики товар. Т.е. пока заказанный товар не придёт на склад магазина, формировать заявку на выдачу товара не начнут. (предполагает использование события-таймера)
7. Далее Сотрудник отдела продаж выдает товар по заказу и на этом процесс заканчивается

Можно сделать бизнес-процесс «Обработка заказа» более сложным, добавив Подпроцесс «Выдача товара по заказу».

Данные действия приведут к появлению диалогового окна Создание процесса, в котором вы при помощи мастера сможете определить параметры нового бизнес-процесса.

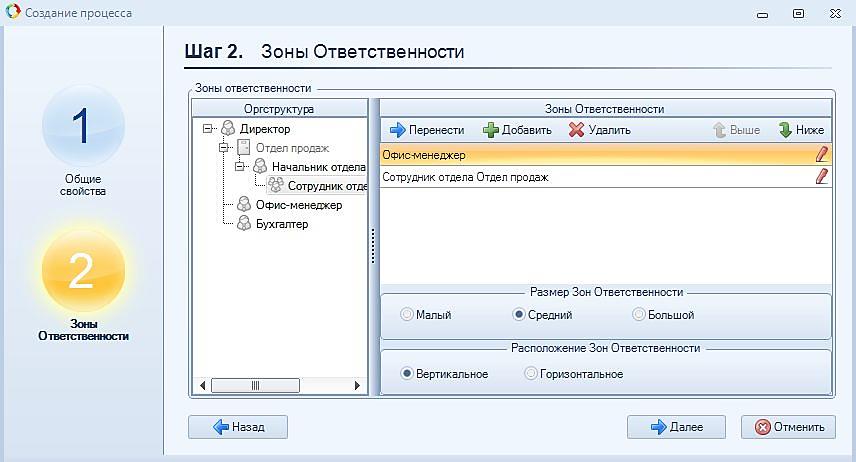


На **Шаге 1** укажите начальные настройки:

* Название бизнес-процесса\*. Данное имя будет использоваться не только для определения бизнес-процесса в списке процессоров Ди­зайнера ELMA, но и при вызове процесса из веб-приложения. Назовем бизнес-процесс «Обработка заказа»
* Группа бизнес-процесса – как уже говорилось ранее, все процессы организованны в виде древа для лучшей навигации. Например, процессы, связанные с логистикой, удобно помещать в группу Логистика. Группу можно выбрать из раскрывающегося списка; создать новую группу можно при помощи кнопки плюс – в этом случае откроется диалоговое окно Группа процессов. Чтобы поместить бизнес-процесс в корень древа, в раскрывающемся списке выберите пункт Все процессы.

Для продолжения нажмите кнопку Далее.

На **Шаге 2** необходимо указать участников бизнес-процесса. Участники бизнес-процесса выбираются из организационной структуры, созданной и опубликованной ранее. В нашем бизнес-процессе «Обработка заказа» будут участвовать Сотрудник отдела продаж и Офис-менеджер.



Добавить участника бизнес-процесса можно несколькими способами:

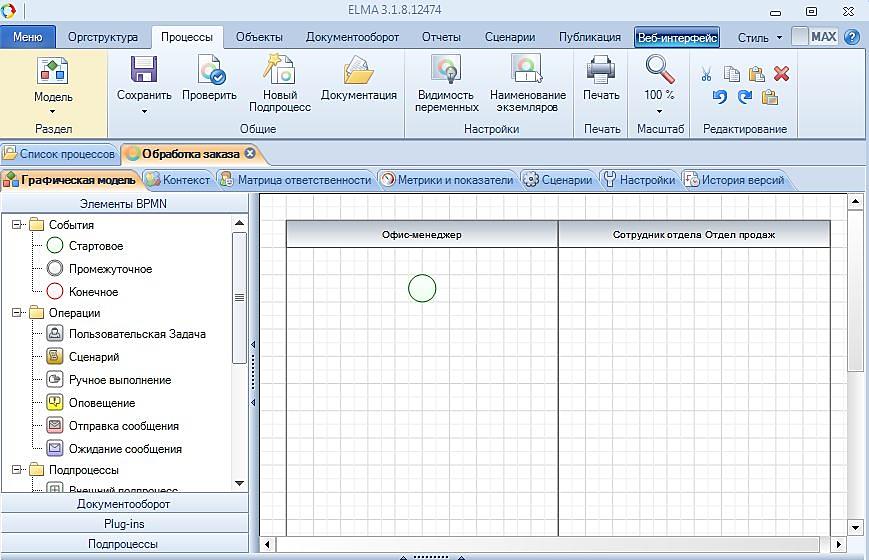
* Используя механизм Drag&Drop, перетащить участника из списка Оргструктура в список Зоны ответственности.
* Выделить элемент в списке Оргструктура и нажать кнопку Добавить, что находится над списком Зоны ответственности.
* Воспользоваться кнопкой Добавить, что находится над списком Зоны ответственности, и в появившемся диалоговом окне выбрать участника бизнес-процесса.

Последний способ позволяет более гибко настроить зону ответственности – вы сможете добавлять динамические зоны ответственности и зоны ответственности типа «бизнес-роль». Более подробно о зонах ответственности читайте в справке по Дизайнеру ELMA.

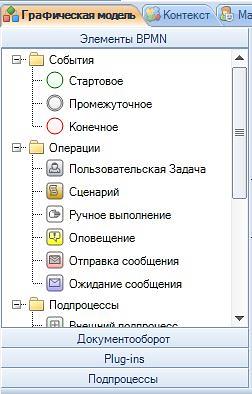
После составления списка участников бизнес-процесса нажмите кнопку Далее, чтобы завершить работу мастера создания бизнес-процесса. Дизайнер ELMA создаст «пустой» бизнес-процесс с зонами ответственности, установленными согласно списку участников. Для каждого открытого процесса на вкладке Процессы создается новая вложенная вкладка. При помощи мыши можно переключаться между открытыми процессами.

Итак, можно переходить к моделированию процесса.

После создания процесса мы видим на экране «пустой» процесс, где соз­даны зоны ответственности участников процесса и установлено лишь стартовое событие.



Следующий шаг – создание цепочки операций. Для того, чтобы поместить на графическую модель операцию необходимо перетащить ее с Панели операций.



Панель операций находится в левой части окна и содержит в виде списка все доступные операции системы. Операции разбиты на несколько вкладок и групп по смыслу их работы.

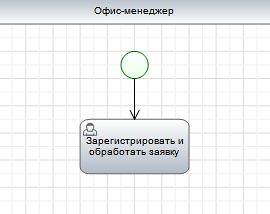
Для начала рассмотрим добавление операции типа ^ Пользовательская задача. Просто при помощи мыши перетащите значок операции Пользова­тельская задача с Панели операций на свободную область графической моде­ли в зону ответственности участника процесса, который отвечает за данную операцию. По умолчанию операция будет названа автоматически сгенериро­ванным именем, которое впоследствии можно изменить. Назовем нашу задачу офис-менеджеру «Зарегистрировать и обработать заявку»

Подобным образом вы можете добавить сколько угодно необходимых операций на графическую модель процесса. При помощи мыши вы также можете свободно перемещать уже существующие операции по графической модели с места на место.

После того, как операции процесса размещены на графической модели, следует соединить их переходами – эти линии связи позволят указать на­правление хода выполнения бизнес-процесса.

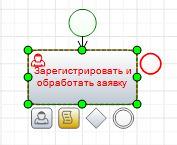
Для соединения двух операций следует подвести указатель мыши к краю первой операции, около указателя мыши появится небольшой красный квадрат, обозначающий точку выхода из операции. Теперь нажмите и удерживайте левую кнопку мыши.

Переместите указатель мыши к краю второй операции, около указателя мыши появится небольшой красный квадрат, обозначающий точку входа в операцию. Отпустите левую кнопку мыши. Линия перехода будет прочерчена. О направлении перехода позволяет судить стрелка на ее конце.



Таким образом, мы соединяем элементы на графической модели между собой. Некоторые операции могут иметь более одного входа и более одного вы­хода. Все элементы на графической модели, которые можно соединить между собой, соединяются схожим образом. На графической модели не должно остаться «висячих» операций, которые не имеют достаточного количества входов и выходов.

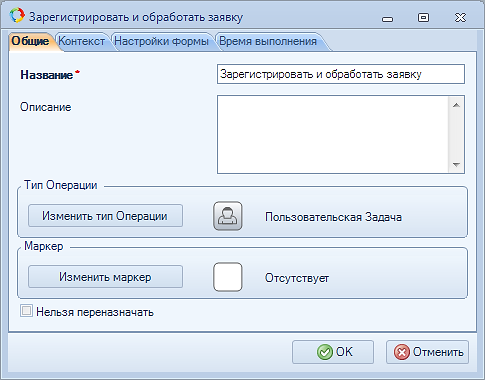
Существует еще один способ добавления операций на графическую модель процесса, позволяющий автоматически устанавливать связи между добавляемыми операциями: щелкните мышью по операции на графической модели процесса, от которой требуется создать следующую операцию.



Щелчок мышью по операции позволяет выбрать ее. При этом по краям выделенной операции появляются зеленые маркеры, перетаскивая мышью которые, можно изменять размеры операции на графической модели. Кроме того, за пределами объекта можно увидеть несколько небольших значков, которые представляют собой миниатюры операций, таких как Пользовательская задача, Сценарий, Шлюз, Промежуточное событие, Конечное событие.

Если при помощи мыши перетащить такой значок на нужное место на графической модели процесса, Дизайнер ELMA автоматически создаст новую операцию, уже соединенную с родительской. Подобный механизм вы уже могли видеть при моделировании организационной структуры.

Помещенные на графическую модель процессов операции имеют назва­ние, автоматически сгенерированное системой, кроме того они совершенно не настроены. Для того, чтобы перейти к настройке операции, просто либо дважды щелкните по блоку операции на графической модели, либо щелкните по блоку операции правой кнопкой мыши и в появившемся контекстном меню выберите пункт Настройки.



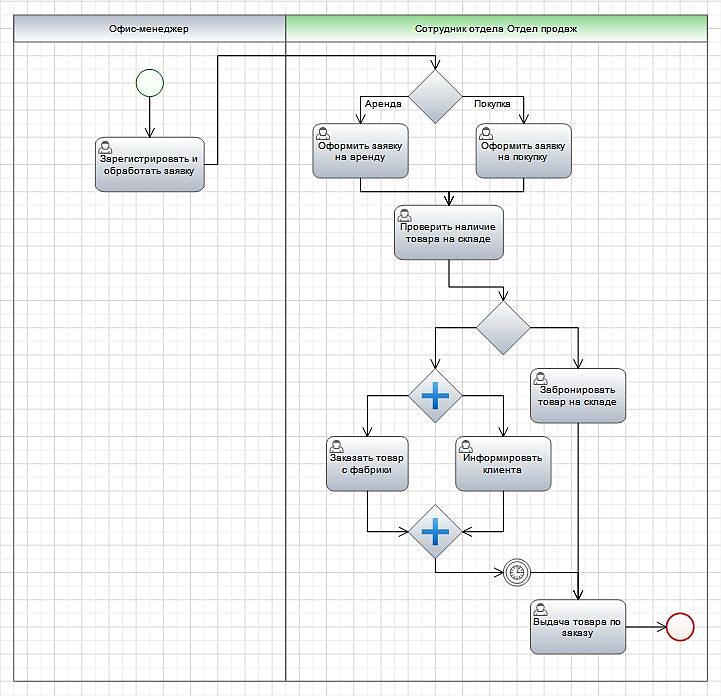
Система отобразит диалоговое окно настроек операции. Содержимое данного диалогового окна зависит от типа выбранной операции. В случае пользовательской задачи оно содержит три вкладки. Первая вкладка позволяет настроить название операции, которое будет отображаться на графической модели процесса, оставить описание к операции, изменить тип операции на другой, а также определить возможность переназначения данной задачи. Кнопка Изменить маркер позволяет настроить особенности хода выполнения данной операции.

Вкладка Контекст позволяет настроить, какие данные пользователь будет получать, а какие он должен будет сообщить системе при выполнении за­дачи.

Для нашего процесса – это ФИО заказчика, телефон, e-mail, адрес, марка велосипеда и аренда это велосипеда или покупка.

Итак, смоделируем весь бизнес-процесс «Обработка заказа»

При моделирование не забывайте писать правильно названия шлюзов (к примеру Далее) и указывать контекстные переменные которые будут использоваться в течении всего процесса.



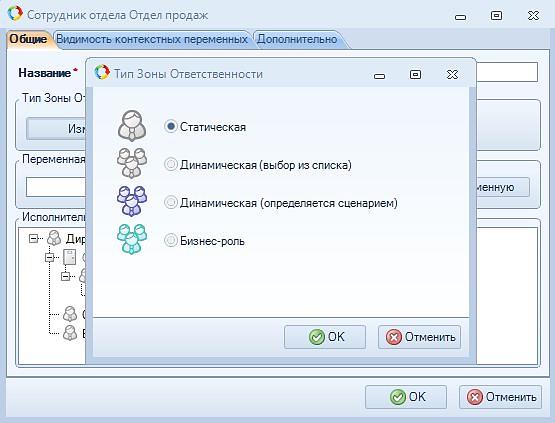
**Настройка зон ответственности**

Существуют случаи, когда в рамках бизнес-процесса нельзя определить к зоне ответственности кого-то одного пользователя. Например, требуется, чтобы исполнитель задачи определялся в ходе выполнения процесса.

Для разрешения подобных ситуаций в системе ELMA введены Динамические зоны ответственности. Суть динамической зоны ответственности проста: исполнитель по задачам в рамках данной зоны ответственности определяется во время исполнения бизнес-процесса. Определение исполнителя может происходить, как в ходе выполнения пользовательской задачи – например, ответственный пользователь выберет исполнителя из списка – так и при помощи программного кода – сценарием.

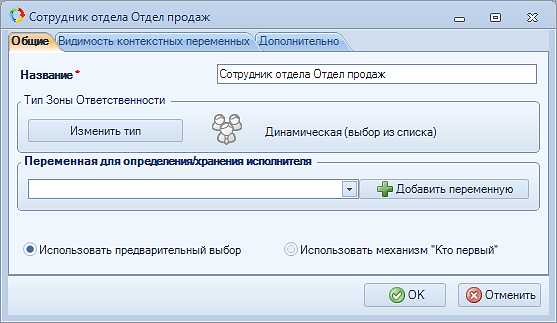
При создании бизнес-процесса можно было сразу определить наличие динамических зон ответственности. Если же вы этого не сделали – ничего страшного, создать динамическую зону ответственности в процессе можно в любой момент: вы можете добавить новую зону ответственности путем перетаскивания соответствующего элемента с Панели операций или же изменить тип уже созданной зоны ответственности.

Для изменения типа зоны ответственности либо дважды щелкните мышью по ее заголовку (изменим зону «Сотрудник отдела продаж», сделаем её динамической), либо щелкните правой кнопкой мыши по свободному участку зоны ответственности и в появившемся контекстном меню выберите пункт Настройки. Откроется диалоговое окно настройки зоны ответственности.

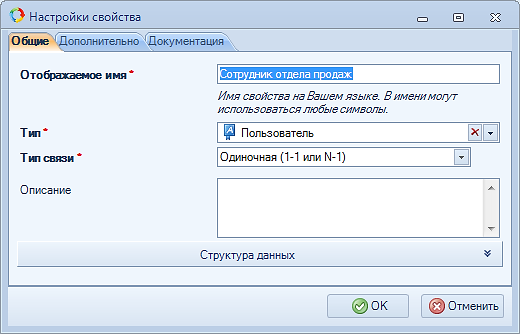


Далее нажмите кнопку Изменить тип в блоке Тип зоны ответственности. В появившемся диалоговом окне установите переключатель в положение Динамическая (выбор из списка) и нажмите кнопку ОК для сохранения настроек. Существуют также динамические зоны ответственности, в которых исполнитель определяется сценарием, более подробно о таких зонах ответственности читайте в справке по Дизайнеру ELMA.

Теперь от вас потребуется указать, в какой переменной системы будет производиться хранение исполнителя. Для этого служит раскрывающийся список блока Переменная для определения/хранения исполнителя. Если переменных такого типа в системе создано не было, а по умолчанию их быть не должно, список будет пуст.

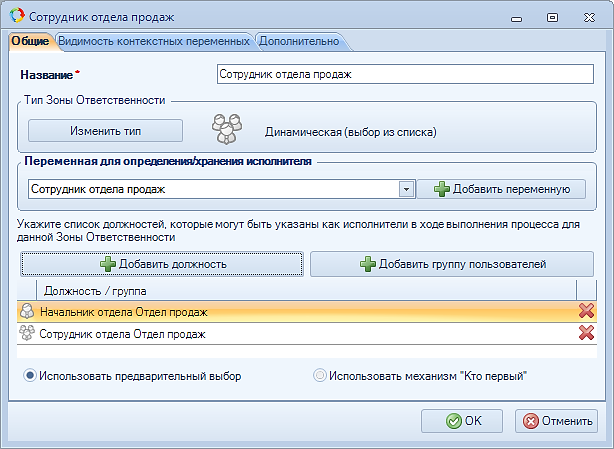


Для создания новой переменной используйте кнопку Добавить переменную. В появившемся диалоговом окне определите имя переменной и при на­личии желания некоторые другие параметры создаваемой переменной.



Теперь в выбранную переменную можно добавить пользователей или группы пользователей, из которых и будет выбираться исполнитель. По умолчанию переменная уже содержит группу Все пользователи, включаю­щую всех пользователей системы. Если вы хотите ограничить круг потенци­альных исполнителей, при помощи кнопки удалите группу Все пользователи, а затем добавьте новых пользователей или группы при помощи кнопок Добавить должность и Добавить группу.

Для нашего процесса добавим Начальника Отдела продаж и Сотрудника отдела продаж и Выберем механизм «Кто первый»



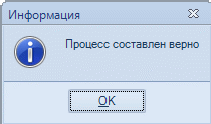
Сформировав таким образом список потенциальных исполнителей, нажмите кнопку ОК для сохранения настроек.

Подобно тому, как статическую зону ответственности вы изменили на динамическую, можно произвольно изменять тип любой зоны ответственно­сти.

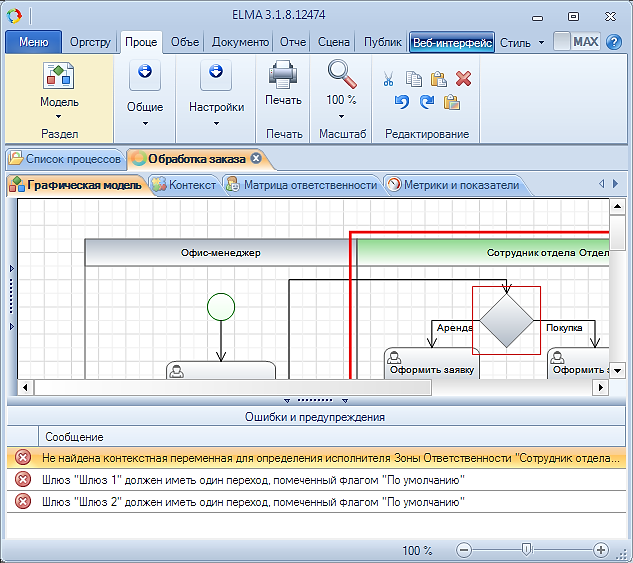
**Проверка бизнес-процесса и его публикация**

После моделирования процесса система предоставляет возможность проверить графическую модель процесса на корректность и наличие потенциальных ошибок.

Для проверки графической модели процесса используйте кнопку Проверить панели инструментов. В случае, если графическая модель составлена верно в соответствии со стандартом BPMN, система выдаст соответствующее уведомление:



В случае, если графическая модель содержит ошибки, система отметит цветной рамкой элемент, где предположительно содержится ошибка, и выведет список ошибок на панели Ошибки и предупреждения.



Желтой рамкой обводятся элементы, вызвавшие у системы подозрения – это обычное предупреждение, говорящее пользователю, что с объектом не все в порядке, но это не повлияет на ход процесса. Красной рамкой обводятся элементы, содержащие ошибки – запустить такой процесс не удастся.

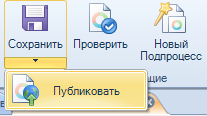
После того, как моделирование процесса завершено, и процесс отлажен, можно переходить к публикации процесса. Сохраненный в Дизайнере про­цесс нельзя будет использовать в веб-приложении системы, пока он не будет опубликован.

Публикация необходима в двух случаях:

1. Для нового процесса. Пока процесс не опубликован, с ним не смогут работать обычные пользователи.
2. При модификации существующего процесса. Публикация процесса делает доступными изменения, которые были сделаны с момента последней публикации.

Опубликовать бизнес-процесс можно двумя способами.

**Способ 1.** Нажмите нижнюю часть многофункциональной кнопки Со­хранить, в появившемся контекстном меню выберите пункт Опубликовать.



**Способ 2.** Перейти в Дизайнере на вкладку Публикация, где в блоке Про­цессы можно управлять публикацией процессов. (Подробнее о таком способе публикации читайте в руководстве пользователя)

После публикации изменения в процессе будут доступны для использования.

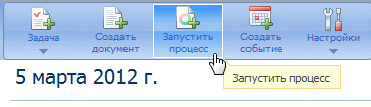
**Запуск бизнес-процесса**

Возможность запуска бизнес-процесса имеет пользователь, который имеет на это право согласно графической модели процесса. Кроме того, администратор системы может ограничить возможность запуска процесса лишь определенным кругом пользователей.

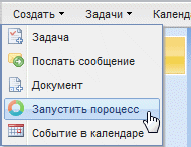
Для запуска нашего бизнес-процесса «Обработка заказа», нам надо зайти в веб-часть (http://127.0.0.1:7000/) под пользователем у которого должность офис-менеджер, у нас это Колганова Диана (логин office, пароль пустой)

Существует несколько способов запустить процесс:

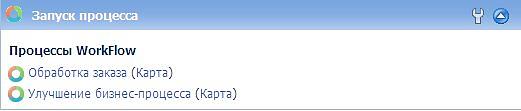
1. При помощи кнопки Запустить процесс панели инструментов глав­ной страницы системы;



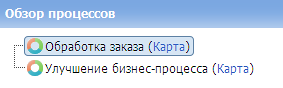
1. При помощи команды Создать – Запустить процесс горизонтального меню;



1. Выбрать нужный процесс с портлета Запуск процесса. Выбираем наш процесс «Обработка заказа»

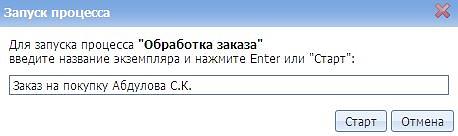


Первые два способа приводят к открытию диалогового окна Обзор процессов, в котором в виде древа представлены все процессы, которые имеет право запустить текущий пользователь. Для запуска процесса просто щелкните по нему мышью и подтвердите необходимость запуска в появившемся диалоговом окне.



Перед запуском процесса можно просмотреть его карту – она будет открыта на новой вкладке браузера. Карта процесса визуально напоминает графическую модель процесса, созданную в Дизайнере ELMA.

Когда процесс будет запущен, от вас первым делом потребуется указать имя экземпляра процесса и нажать кнопку Старт. Так как в системе один бизнес-процесс может быть одновременно запущен несколько раз, каждый его экземпляр должен иметь уникальное имя, позволяющее отличить один экземпляр от другого. Старайтесь давать экземплярам процесса простые и понятные названия, чтобы избежать возможной путаницы.

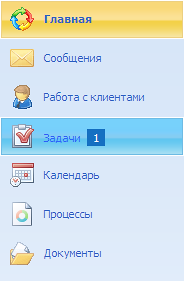


Запущенный процесс выполняется в соответствии с его графической моделью. Система будет автоматически назначать пользователям задачи, выполнять сценарии, принимать решения в шлюзах, посылать пользователям уведомления и совершать другие действия, предусмотренные при моделиро­вании бизнес-процесса.

**Выполнение пользовательских задач**

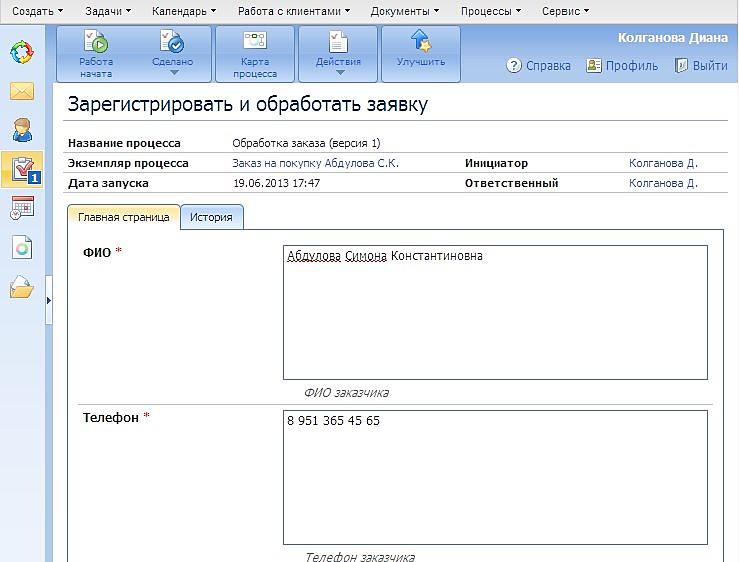
Если в ходе выполнения бизнес-процесса вы являетесь исполнителем пользовательской задачи, в соответствующий момент выполнения процесса вы получите соответствующее уведомление.

Во-первых, в разделе Задачи бокового меню число после названия раздела увеличится на некоторое количество единиц: если согласно бизнес-про­цессу вам назначается одна задача, то на единицу, если две задачи – на два и т.д. Перейдя в раздел Задачи можно просмотреть, какие именно задачи вам были назначены. Щелчок мышью по заголовку задачи позволяет перейти на страницу пользовательской задачи.



Щелчок мышью по заголовку задачи позволяет перейти на страницу пользовательской задачи.

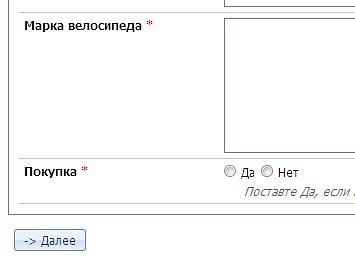
У нас, так как, первая задача по процессу «Обработка заказа» была у инициатора процесса – офис-менеджера, то пользовательская задача открывается автоматом после старта процесса.



В верхней части страницы пользовательской задачи отображается общая информация о процессе: имя бизнес-процесса (Обработка заказа), имя экземпляра (заказ на покупку Абдулова С.К.), дата запуска, имена инициатора процесса и пользователя, ответственного за выполнение процесса (Колганова Диана).

Основная часть страницы пользовательской задачи разделена на вклад­ки. На вкладке ^ Главная страница отображаются элементы, сформированные на основании контекста задачи. Здесь нам потребуется ввод определенных данных в соответствующие поля (ФИО, Телефон, E-mail, Адрес, Марка велосипеда, Покупка). Обязательные для заполнения поля помечены звездочкой – не заполнив их, вы не сможете доложить о выполнении задачи.

В нижней части вкладки Главная страница можно увидеть кнопки, при помощи которых можно доложить системе о выполнении задачи. Количество данных кнопок и их названия определяются количеством и именами переходов, выходящих из пользовательской задачи.



Панель инструментов страницы пользовательской задачи позволяет выполнить ряд специфических действий с задачей:

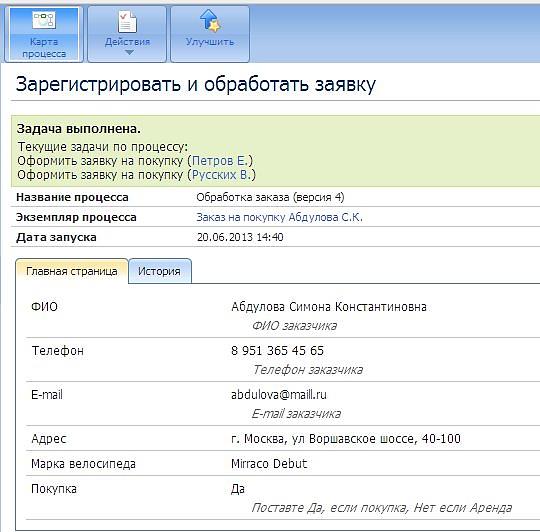
1. **Карта процесса.** Открывает карту процесса. На карте процесса можно просмотреть не только ее графическую модель, но и увидеть, на каком этапе находится выполнение процесса. Кнопка Карта процесса появляется на панели инструментов только у тех пользователей, кому даны права на просмотр карты процесса.
2. **Действия.** Данная многофункциональная кнопка позволяет получить доступ к следующим командам:

* **Сохранить**. Если вы пока не можете отчитаться о выполнении зада­чи, но уже заполнили некоторые поля страницы и хотите сохранить эти данные, используйте кнопку Сохранить панели инструментов. Это позволит временно закрыть страницу пользовательской задачи, а при следующем ее просмотре вы увидите ее в том же виде.
* **Назначить**. Позволяет отказаться от личного выполнения назначенной вам задачи и выбрать другого исполнителя. Данное действие возможно, если вы имеете на это права, а данная пользовательская задача может переназначаться со­гласно ее настройкам.
* **Задать вопрос.** Задать вопрос другому участнику данного бизнес-процесса.
* **Добавить комментарий.** Оставить комментарий по данному экземпляру бизнес-процесса.

Комментарии и вопросы, заданные по процессу его участниками, могут быть просмотрены на одноименных вкладках страницы пользовательской за­дачи или на странице экземпляра процесса.

Заполнив все поля формы задачи «Зарегистрировать и обработать заявку», нажмем кнопку Далее.

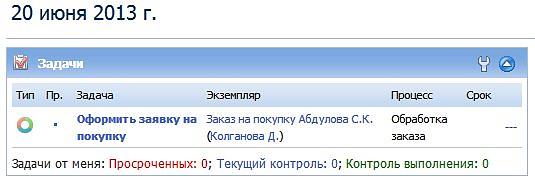
Система выдаст сообщение, что задача выполнена и укажет нам следующих исполнителей



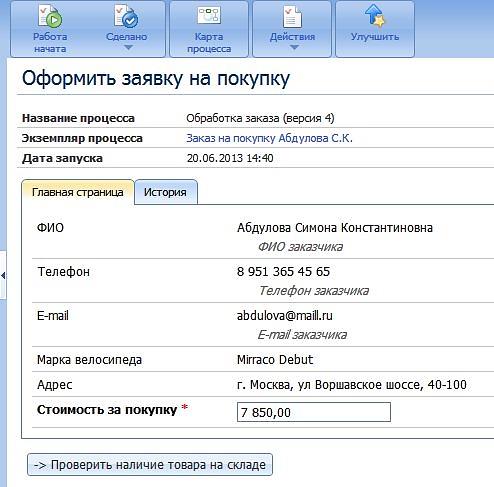
Задача согласно карте процесса уйдет на менеджеров отдела продаж, кто первый её откроет, тот и будет ответственный за выполнение задачи.

Зайдем в систему под первым менеджером Петровым Евгением – логин manager, пароль пустой.

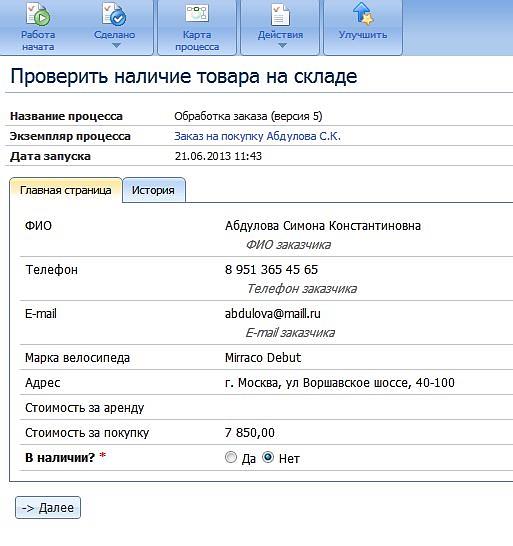
И увидим, что ему пришла задача по процессу «Оформить заявку на покупку»



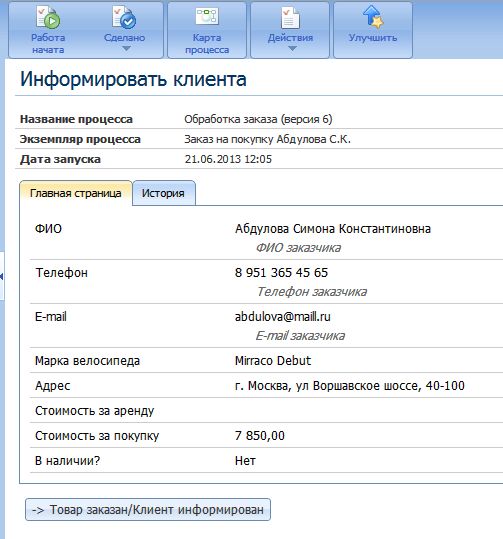
Перейдем в задачу и заполним все поля формы и отправим задачу дальше.



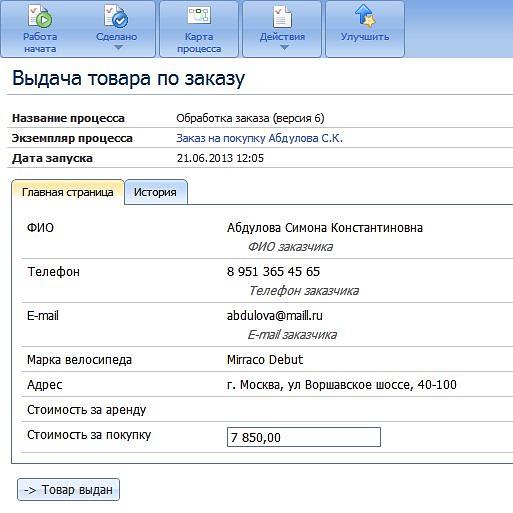
Далее Петрову Евгению также придет задача «Проверить наличие товара на складе», после того как он проверит есть ли данный товар на складе он отвечает к примеру что нет и мы двигаемся дальше по процессу.



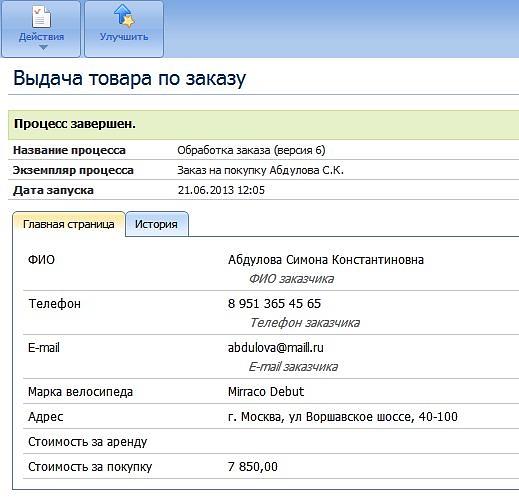
Далее одновременно приходит две задачи на Петрова Евгения, «Заказать товар с фабрики» и «Информировать клиента. После того как менеджер закажет велосипед необходимой марки с фабрики и информирует клиента по телефону и e-mail, что товар придет с задержкой. Он говорит что товар заказан и клиент информирован.



Далее приходит задача опять на менеджера Петрова Евгения – «Выдача товара по заказу»



И процесс завершается после того как товар будет выдан. Система выдаст сообщение, что Процесс завершен с информацией по процессу.



**Контроль и улучшение бизнес-процесса**

Система предоставляет возможность контролировать ход бизнес-процесса.

Все функции контроля можно разбить на две категории:

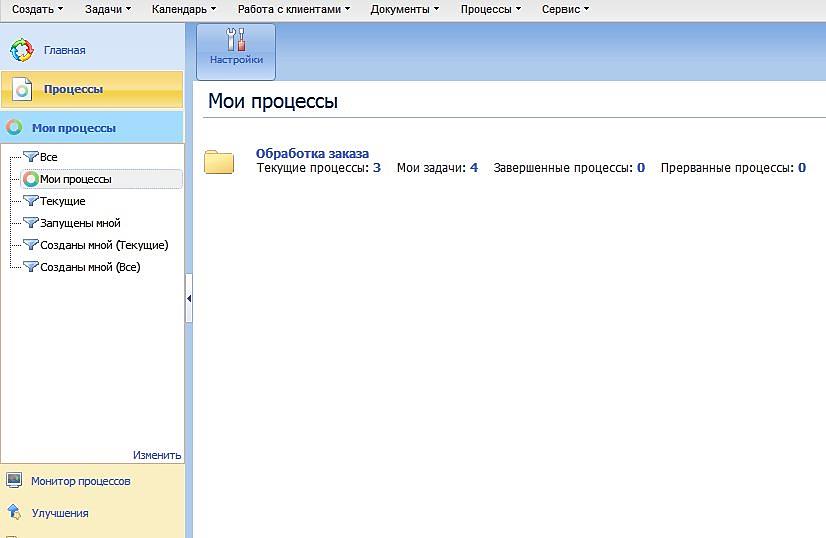
1. Оперативный контроль

2. Контроль по ключевым показателям

Подробнее рассмотрим возможности оперативного контроля хода процессов. Для этих целей служит страница Мои процессы и одноименный портлет, а также страница Монитор процессов.

**Мои процессы**

Страница Мои процессы, открыть которую можно при помощи команды Процессы – Мои процессы бокового или горизонтального меню, отображает список процессов, в которых принимает участие пользователь.



По умолчанию экземпляр процесса будет виден следующим пользователям:

* владелец процесса;
* текущий исполнитель;
* ответственный за исполнение;
* инициатор процесса.

Основную часть страницы Мои процессы занимает список процессов. Под каждым процессом отображается информация о нем:

* Текущие процессы – количество текущих экземпляров процесса;
* Мои задачи – количество задач по процессам, назначенных на текущего пользователя;
* Завершенные процессы – количество завершенных экземпляров процесса;
* Прерванные процессы – количество прерванных экземпляров процесса.

Щелкнув мышью по названию процесса, вы попадете на страницу процесса.

Щелкая мышью по голубым цифрам, обозначающим количество текущих, завершенных и прерванных экземпляров процесса и количество задач, вы будете попадать на соответствующие вкладки с информацией по разделу на странице процесса.

При помощи кнопки Настройки вы можете перейти на страницу настроек порядка отображения типов процессов на странице Мои процессы.

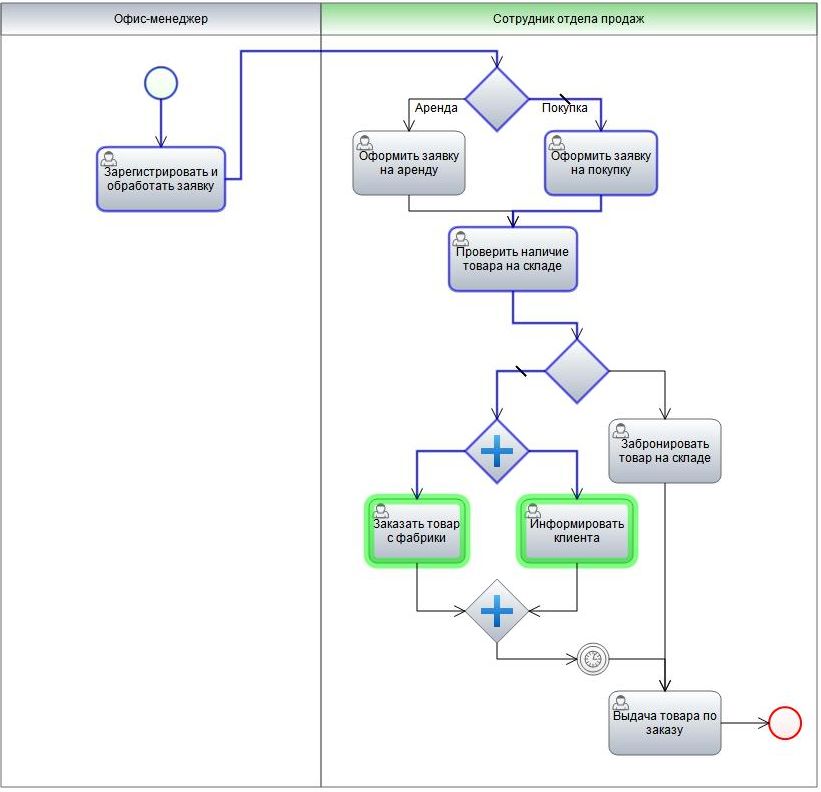
**Карта процесса**

Карта процесса представляет собой графическое отображение текущего состояния бизнес-процесса на графической модели процесса. Просмотрев карту выполняемого процесса, легко понять, на какой стадии находится выполнение.

Просмотр карты процесса осуществляется на новой вкладке браузера.

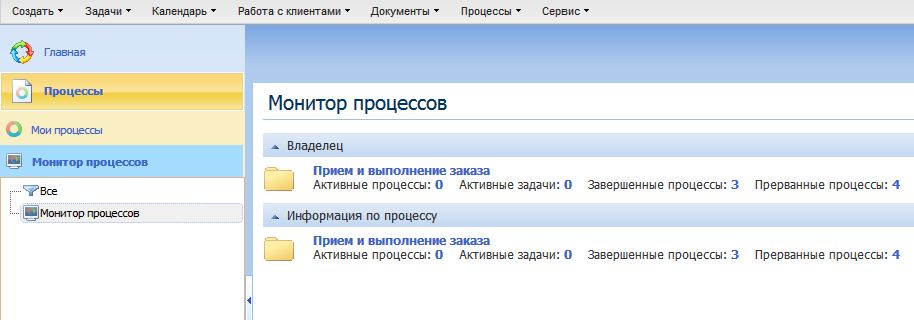
Находясь на странице пользовательской задачи, просмотреть карту процесса можно при помощи кнопки Карта процесса панели инструментов. Со страниц Мои процессы, Монитор процессов, а также с портлета Мои процессы просмотреть карту процесса можно при помощи кнопки Карта.

На карте процесса, запущенного на выполнение, зеленым цветом отмечен текущий шаг процесса, галочками помечены уже пройденные шаги.



**Монитор процессов**

Страница Монитор процессов представляет собой удобный инструмент для контроля хода процесса в целом – доступ к этой странице может быть получен при помощи команды Процессы – Монитор процессов бокового или горизонтального меню. Эта страница предназначена для анализа работы по процессам. На странице типа процесса представлена аналитическая информация по процессу.



На странице Монитор процессов отображаются типы процессов, в которых пользователь является владельцем, куратором или информируемым. В соответствии с этим все процессы на странице разбиты по вкладкам.

Под каждым процессом отображается:

* Активные процессы – количество выполняющихся в данный момент экземпляров процесса.
* Активные задачи – количество активных на данный момент задач по всем экземплярам процесса.
* Завершенные процессы – количество завершенных экземпляров процесса.
* Прерванные процессы – количество прерванных экземпляров процесса.

Щелкая мышью по голубым числам, обозначающим количество активных, завершенных и прерванных экземпляров процесса и активных задач, вы будете попадать на соответствующие вкладки с информацией по разделу на странице процесса.

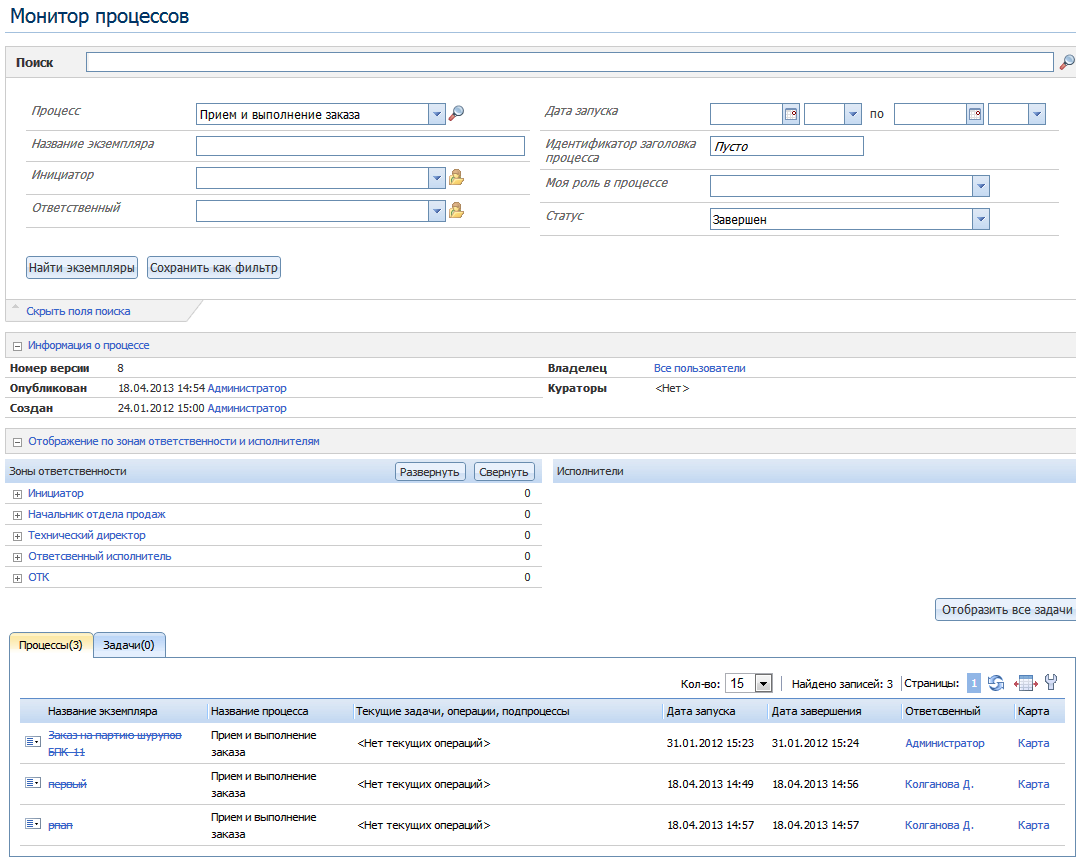
Щелчок мышью по названию процесса приведет к открытию страни­цы с полной информацией по всем экземплярам этого процесса. В заголовке страницы отображается название процесса. Остальное содержимое страницы представлено следующими объектами:

Первая сворачиваемая панель, носящая название Информация о процессе, содержит общую информацию о самом процессе: когда он был создан, когда опубликован, кто владелец, кто куратор процесса.

Вторая сворачиваемая панель, носящая название Общий фильтр, содержит фильтр для экземпляров процесса, при помощи которого можно отсеять нужные вам экземпляры процесса. Задайте условия фильтра и нажмите кнопку Искать. Внизу страницы отобразятся экземпляры процесса, соответствующие условиям фильтра.

Чуть ниже сворачиваемых панелей находится блок, на трех вкладках которого можно получить информацию об экземплярах процесса. На вкладке Активные процессы отображаются все запущенные в данный момент экземпляры процессов и информация по ним. На вкладке Завершенные процес­са отображаются экземпляры процессов, завершенных в штатном порядке согласно логике их выполнения. На вкладке Прерванные процессы отображаются экземпляры процессов, принудительно прерванные владельцем или другим лицом, имеющим на то права.

В самом низу страницы находится блок, содержащий две вкладки, при помощи которого можно в виде списка просмотреть информацию о текущих экземплярах процесса и задач. На вкладке Процессы отображается список текущих экземпляров процесса.



Щелкнув мышью по названию экземпляра процесса, вы попадете на страницу экземпляра процесса.

Щелкнув мышью по ссылке Карта, в новом окне откроется карта процесса и вы сможете ее просмотреть.

На вкладке Задачи отображается список текущих задач по экземплярам процесса. Щелкнув мышью по названию задачи, вы попадете на ее страницу.

**Улучшение и оптимизация бизнес-процессов**

В ходе работы достаточно часто появляются идеи по улучшению и оптимизации процессов. Система ELMA построена гибко и позволяет вносить изменения в процессы даже после внедрения этих процессов в работу.

Для внесения изменений необходимо открыть в Дизайнере ELMA конфи­гурацию на сервере и открыть для редактирования нужный бизнес-процесс из списка процессов.

Теперь внесите в бизнес-процесс нужные изменения, после чего необходимо будет опубликовать изменения. При публикации новой версии процесса в системе могут быть активные экземпляры старой версии бизнес-процесса.

Рассмотрим следующую ситуацию. Предположим, что в нашем процессе «Обработка заказа» были внесены изменения, и мы за последнюю задачу «выдача товара» поставили ответственного офис-менеджера. А в момент публикации у нас в работе были:

* Информирование клиента по заказу на покупку Абдуловой С.К;

Выдача товара по данному заказу пойдет без изменений. Т.е его будет выдавать менеджер отдела продаж. Если после публикации изменений процесса будет запущен процесс «Обработка заказа», то выдачу уже будет производить офис-менеджер.

Другими словами, экземпляры процесса, которые были запущены до момента публикации новой версии процесса, будут выполняться согласно старой версии процесса. По новой версии процесса будут выполняться экземпляры, созданные уже после публикации новой версии процесса.

**Пример карточки бизнес-процесса**

Процесс «Заявка на приобретение техники»

Сотрудник компании заполняет заявку на технику и отправляет заявку Администратору

Администратор рассматривает заявку и либо отклоняет её, указывая причину отказа сотруднику, после чего процесс заканчивается. Либо просит согласовать заявку Директором, директор может

* Отклонить заявку, указав причину отказа, после чего процесс заканчивается.
* Согласовать заявку, после чего ставится задача директору на выдачу денег, далее деньги выдаются и ставится задача на закупку техники Администратору.

Администратор закупает технику и далее ставится задача на получение техники Сотруднику компании, после того как техника получена сотрудником процесс заканчивается.

**Задание**

Составьте карточку с названием бизнес-процесса и его текстовым описанием.

Опишите и запустите бизнес-процессы предметной области в соответствии с вариантом

**Варианты**

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №13. Построение диаграммы Классов

**Цель**: изучение основ создания диаграмм классов на языке UML, получение навыков построения диаграмм классов, применение приобретенных навыков для построения объектно-ориентированных моделей определенной предметной области.

**Теоретические сведения**

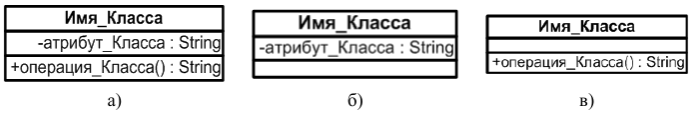
**Построение диаграммы Классов**

Диаграмма классов является одной из канонических диаграмм UML, создаваемой для визуализации структурированной статической модели предметной области. Этот вид диаграмм представляет собой графическое изображение объектов – классов с присущими им атрибутами, операциями и различных отношений между классами.

***1. Классы***

Класс (class) служит для обозначения множества объектов, обладающих функциональным набором одинаково описывающих параметров (атрибутов), реализуемых операций и однотипными отношениями с объектами других классов.

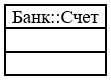
На диаграмме класс изображается габаритной прямоугольной рамкой, которая дополнительно может быть разделена горизонтальными линиями на секции, каждая из которых предназначена для указания имени, атрибутов (свойств) и реализуемых операций объектов данного класса.



Имя класса является обязательным элементом в его обозначении и должно быть уникальным (хотя бы в пределах пакета), имеющее непосредственное отношение к контексту моделируемой предметной области.

В соответствии с принятым в языке UML общим соглашением в качестве имени класса используются существительные и прилагательные, каждое из которых начинается с заглавной буквы, записанные без пробелов. Например, в качестве имен классов могут быть использованы профессиональные термины: «Сотрудник», «Компания», «Руководитель», «Клиент», «Продавец», «Менеджер», «Офис», «Покупатель», «Датчик\_Температуры» и др. Такое имена классов являются простыми.

Иногда возникает необходимость в явном указании пакета, к которому относится данный класс. С этой целью в условном обозначении перед именем класса указывается имя пакета и специальный символ разделитель – двойное двоеточие "::". Такое имя класса является квалифицированным. Текстовая строка имени класса в этом случае записывается в формате <Имя\_пакета>::<Имя\_класса>. Например, если определен пакет с именем «Банк», то имя класса «Счет» может быть записано так.

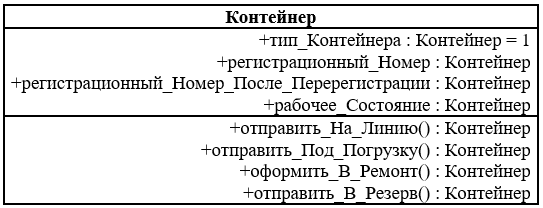


***2. Атрибуты классов***

Содержательной характеристикой класса является атрибут, содержащий множество значений, которые могут принимать отдельные объекты этого класса. При этом, класс может иметь любое число атрибутов или не иметь ни одного. Так, например, атрибутами класса «Усилитель» являются частотный диапазон, выходная мощность, коэффициент нелинейных искажений, уровень шума и т. д.

Запись атрибута также представляет собой отдельную строку текста, содержащую обязательное имя, в котором обычно каждое слово пишется с заглавной буквы, за исключением первого, например, name (имя) или birth\_Date (дата\_Рождения).

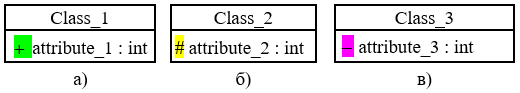
Например, указаны атрибуты класса Контейнер, в качестве которых выступают атрибут тип\_Контейнера, атрибут регистрационный\_Номер\_Контейнера, регистрационный\_Номер\_После\_Перерегистрации, рабочее\_Состояние.



Качественной характеристикой описания элементов класса является квантор видимости атрибута – потенциальная возможность других объектов модели оказывать влияние на отдельные аспекты поведения данного класса.

Эта характеристика может принимать одно из трех возможных значений и, соответственно, отображается при помощи специальных символов: символ "+" (public) обозначает атрибут с областью видимости типа общедоступный; атрибут с этой областью видимости доступен или виден из любого другого класса пакета, в котором определена диаграмма; например, для класса Class\_1 указан атрибут общедоступного типа, символ "#" (protected) обозначает атрибут с областью видимости типа защищенный; атрибут с этой областью видимости недоступен или невиден для всех классов, за исключением подклассов данного класса; например, для класса Class\_2 указан атрибут защищенного типа; символ "-" (private) обозначает атрибут с областью видимости типа закрытый; атрибут с этой областью видимости недоступен или невиден для всех классов без исключения; например, для класса Class\_3 указан атрибут защищенного типа;

Квантор видимости при описании атрибутов может быть опущен, что будет означать тот факт, что видимость атрибута не указывается.



***3. Операции классов***

Операция (operation) класса – это реализация услуги, которая может быть запрошена у любого объекта данного класса, чтобы вызвать определенное его поведение. Класс может иметь любое число операций либо не иметь ни одной. Так автомобиль может перемещаться по грунту, корабль – перемещаться по воде, компьютер – производить вычисления.

Представление полного синтаксиса записи операций класса также подчиняется определенным синтаксическим правилам: каждой операции класса соответствует отдельная строка, которая состоит из квантора видимости операции, обязательного имени операции, выражения типа возвращаемого операцией значения и, возможно, строки-свойства данной операции:

*< квантор видимости >< имя операции > (список параметров) : < выражение типа возвращаемого значения >{строка-свойство}*

Квантор видимости, как и в случае атрибутов класса, может принимать одно из трех возможных значений и, соответственно, также отображается при помощи специального символа.

Для именования операции используются короткие глагольные конструкции, описывающие некоторое поведение класса, которому принадлежит операция. Обычно каждое слово в имени операции пишется с заглавной буквы, за исключением первого.

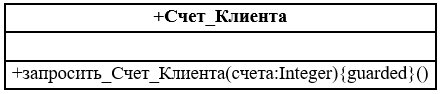
Например, запись +создать() – может обозначать абстрактную операцию по созданию отдельного объекта класса, которая является общедоступной и не содержит формальных параметров, запись +нарисовать(форма: Многоугольник = прямоугольник, цвет\_заливки: Color = (О, О, 255)) – может обозначать операцию по изображению на экране монитора прямоугольной области синего цвета, если не указываются другие значения в качестве аргументов данной операции.

*(список–параметров) содержит необязательные аргументы, синтаксис которых совпадает с синтаксисом атрибутов;*

*< выражение типа возвращаемого значения > является необязательной спецификацией и зависит от конкретного языка программирования;*

*{строка-свойство} показывает значения свойств, которые применяются к данной операции.*

Например, запись запросить\_Cчет\_Клиента(номер\_счета:Integer) – обозначает операцию по установлению наличия средств на текущем счете клиента банка. При этом аргументом данной операции является номер счета клиента, который записывается в виде целого числа (например, «123456»).



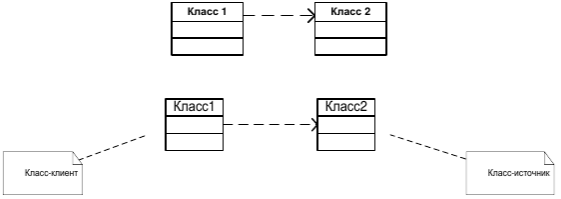
Квантор видимости для операции может быть опущен. В этом случае его отсутствие означает, что видимость операции не указывается.

***4. Отношения между классами***

Классы на диаграмме связываются различными типами отношений. При этом совокупность типов таких отношений фиксирована в языке UML и предопределена их семантикой.

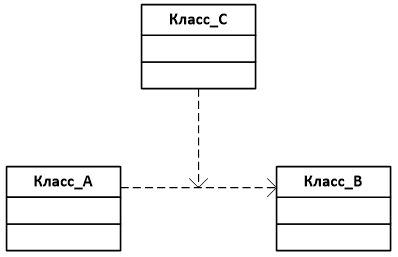
*4.1. Отношение зависимости*

**Отношением зависимости (dependency relationship)** называют связь по использованию, когда изменение в спецификации одного класса может повлиять на поведение другого. Отношение зависимости используется в такой ситуации, когда некоторое изменение одного элемента модели может потребовать изменения другого зависимого от него элемента модели. Отношение зависимости графически изображается пунктирной линией между соответствующими элементами со стрелкой на одном из ее концов, направленной к той сущности, от которой зависит данная сущность. Например, некая сущность Класс\_2 использует другую сущность Класс\_1.



В качестве класса-клиента и класса-источника зависимости может выступать множество элементов модели. В этом случае одна линия со стрелкой, выходящая от источника зависимости, расщепляется в некоторой точке на несколько отдельных линий, каждая из которых имеет отдельную стрелку для класса-клиента.

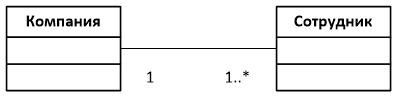
Например, если функционирование сущности Класс\_С зависит от особенностей реализации сущностей Класс\_А и Класс\_Б, то данная зависимость может быть изображена следующим образом.



*4.2. Отношение ассоциации*

**Ассоциацией (association relationship)** называется структурная связь, показывающая, что объекты одного класса некоторым образом связаны с объектами другого или того же самого класса.

Ассоциация может отображаться графически линией со стрелкой (маркером в виде треугольника), показывающей направление следования классов и кратность – количество объектов, связанных отношением. Отсутствие стрелки рядом с именем ассоциации означает, что порядок следования классов в рассматриваемом отношении не определен



Так, в примере кратность «1» для класса «Компания» означает, что каждый сотрудник может работать только в одной компании. Кратность «1..\*» для класса «Сотрудник» означает, что в каждой компании могут работать несколько сотрудников, общее число которых заранее неизвестно и ничем не ограничено.

Специальной формой или частным случаем отношения ассоциации является отношение агрегации, которое, в свою очередь, тоже имеет специальную форму – отношение композиции (см. пункт 3.4.4).

*4.3. Отношение агрегации*

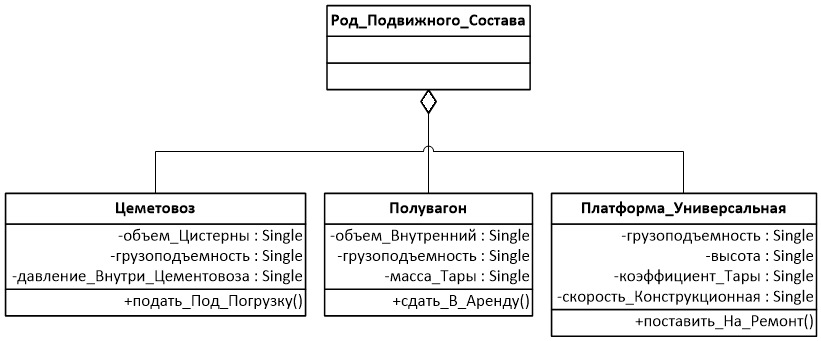
**Отношение агрегации (generalization relationship)** имеет место между несколькими классами в том случае, если один из классов представляет собой некоторую сущность, включающую в себя в качестве составных частей другие сущности.

Данное отношение имеет фундаментальное значение для описания структуры сложных систем, поскольку применяется для представления системных взаимосвязей типа «часть – целое».

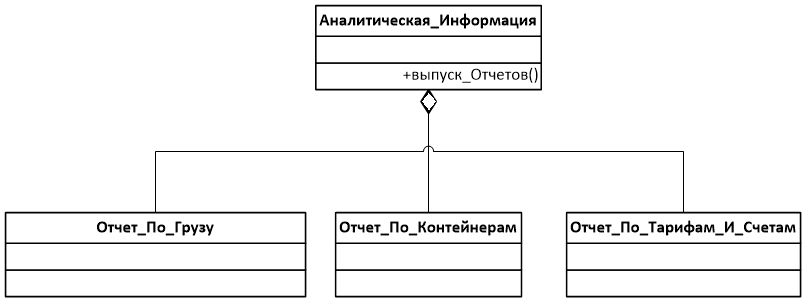
Это отношение по своей сути описывает декомпозицию или разбиение сложной системы на более простые составные части, которые также могут быть подвергнуты декомпозиции, если в этом возникнет необходимость в последующем.

Так, автомобиль состоит из кузова, двигателя, трансмиссии и т.п., а в состав приемопередающего устройства входят передатчик, приемник и антенно-фидерное устройство.

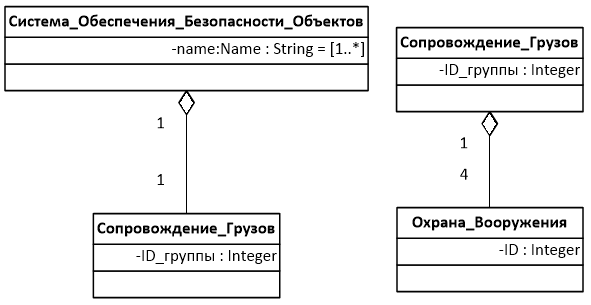
Графически отношение агрегации изображается сплошной линией, один из концов которой представляет собой геометрическую фигуру – ромб. Этот ромб указывает на тот из классов, который представляет собой «целое».



Примером отношения агрегации может служить деление класса Аналитическая\_информация на составные части: Отчет\_по\_грузу, Отчет\_по\_контейнерам, Отчет\_по\_тарифам.



Отношение агрегации обладает кратностью. Так, класс Система\_обеспечения\_безопасности\_объектов может содержать содержит одну подсистему Сопровождение\_грузов, которая в свою очередь может содержать, например, четыре класса Охрана\_вооруженная, каждый из которых может принадлежать лишь одному классу Сопровождение\_грузов.



*4.4. Отношение композиции*

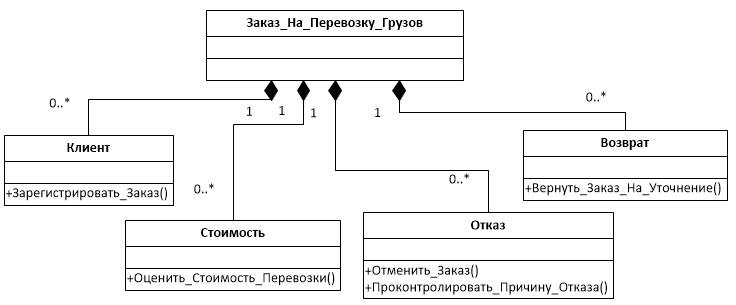
**Отношение композиции (realization relationship)** служит для выделения специальной формы отношения «часть-целое», при которой составляющие части в некотором смысле находятся внутри целого.

Специфика взаимосвязи между ними заключается в том, что части не могут выступать в отрыве от целого, т. е. с уничтожением целого уничтожаются и все его составные части.

Графически отношение композиции изображается сплошной линией, один из концов которой представляет собой закрашенный внутри ромб. Этот ромб указывает на тот из классов, который представляет собой класс-композицию или «целое».



Применительно к классу Заказ\_на\_перевозку\_грузов отношение композиции может иметь следующий вид.



*4.5. Отношение обобщения*

**Отношение обобщения (генерализация)** является обычным таксономическим отношением между более общим элементом (класс-предок) и более частным или специальным элементом (класс-потомок).

Применительно к диаграмме классов данное отношение описывает иерархическое строение классов и наследование их свойств и поведения. При этом предполагается, что класс-потомок обладает всеми свойствами и поведением класса-предка, а также имеет свои собственные свойства и поведение, которые отсутствуют у класса-предка.

На диаграммах отношение обобщения обозначается сплошной линией с треугольной стрелкой на одном из концов, направленной на более общий класс (класс-предок или суперкласс) от более специального класса (класса-потомка или подкласса).

Как правило, на диаграмме может указываться несколько линий для одного отношения обобщения, что отражает его таксономический характер. В этом случае более общий класс разбивается на подклассы одним отношением обобщения, например, так, как показано на рис. 14.

В этом случае данные отдельные линии изображаются сходящимися к единственной стрелке, имеющей с ними общую точку пересечения. Родительский Класс Отчет\_По\_Заказам\_На\_Перевозку имеет три потомка Отчет\_По\_Количеству\_Заказов, Отчет\_По\_Клиентам, Отчет\_За\_Период, которые наследуют структуру и поведение родительского класса.



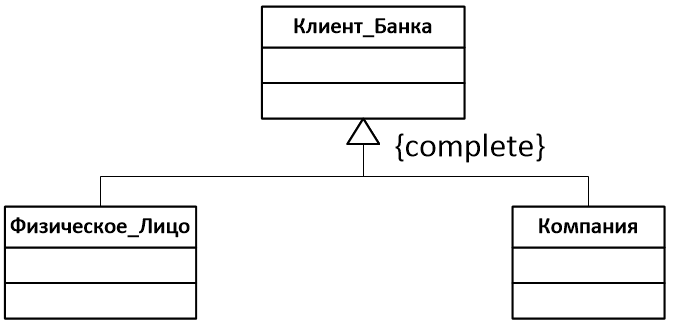
Для связей обобщения язык UML содержит ограничения. В большинстве случаев ограничение размещается рядом с элементом и заключается в фигурные скобки, например {complete}.

**В качестве ограничений** могут быть использованы следующие ключевые слова языка UML:

1. {complete} означает, что в данном отношении обобщения специфицированы все классы-потомки, и других классов-потомков у данного класса-предка быть не может.

Например, класс Клиент\_банка является предком для двух классов: Физическое\_лицо и Компания, и других классов-потомков он не имеет.

На соответствующей диаграмме классов это можно указать явно, записав рядом с линией обобщения данную строку-ограничение.



1. {incomplete} означает тот факт, что на диаграмме указаны в обобщении не все классы-потомки. В последующем, возможно, восполнить их перечень, не изменяя уже построенную диаграмму.
2. {disjoint} означает тот факт, что классы-потомки не могут содержать объектов, одновременно являющихся экземплярами двух или более классов.

В приведенном выше примере это условие также выполняется, поскольку предполагается, что никакое конкретное физическое лицо не может являться одновременно и конкретной компанией. В этом случае рядом с линией обобщения можно записать данную строку-ограничение.

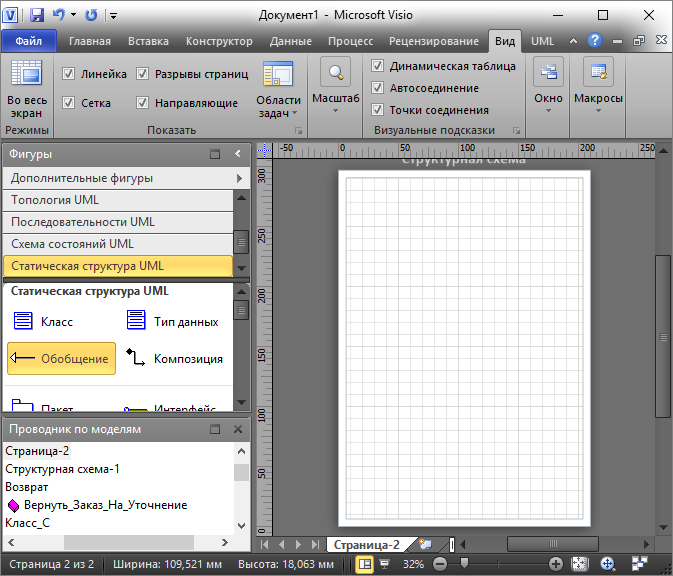
1. {overlapping} означает, что отдельные экземпляры классов-потомков могут принадлежать одновременно нескольким классам.

Например, класс Транспорт может быть специализирован путем создания подклассов Наземный\_Транспорт и Водный\_Транспорт, автомобиль – амфибия относится к обоим классам.

**Методика выполнения**

**Построение диаграммы Классов**

1. Запустите MS Visio.
2. На экране выбора шаблона выберите категорию *Программы и БД* и в ней элемент *Схема модели UML*. Нажмите кнопку *Создать* в правой части экрана.
3. Ознакомьтесь с элементами графического интерфейса и найдите обязательные панели инструментов **Фигуры**, содержащие категории **Деятельность UML, Взаимодействия UML, Компоненты UML, Топология UML, Последовательности UML, Схема состояний UML, Статическая структура UML, Сценарий выполнения UML,** **Проводник по моделям**, содержащий иерархическую структуру объектов Системы UML1, Рабочую область, ярлык Страница\_1, горизонтальную и вертикальную линейки



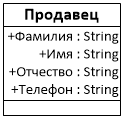
1. Установите следующие параметры страницы: **Ориентация** – Альбомная, **Автоподбор размера** – выключен, **Имя страницы** – *Диаграмма классов для системы продажи товаров по каталогу.*
2. Перейдите в категорию **Статическая структура UML**, ознакомьтесь с содержимым этой категории и найдите элементы: Класс, Пакет, Подсистема, Интерфейс, Метакласс, Двусторонняя ассоциация, Обобщение, Композиция, Примечание, Ограничение и др.
3. Создайте поэтапно статическую структуру классов UML, с помощью которой может быть сформирована некоторая функциональная часть системы, например, *Система продажи товаров по каталогу*. Для чего:

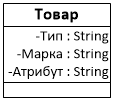
* Выберите структурные элементы (идентифицируйте классы), участвующие в организации продаж, например, *Продавец, Товар, Заказ, Заказ\_Оплата, Клиент, Корпоративный\_Клиент, Частный\_Клиент* и создайте предварительный вариант совокупности классов с указанием имен
* Установите для каждого класса атрибуты в соответствии с перечнем и содержательным описанием бизнес-процессов:

например, для класса *Продавец* в качестве атрибутов могут выступать данные: *фамилия, имя, отчество, телефон*. В данном случае все атрибуты видимы, принадлежат основному пакету *Продавец* .

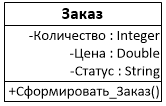
для класса *Товар* в качестве атрибутов могут выступать данные: тип, марка, артикул.



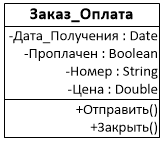




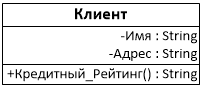
для класса *Заказ* в качестве атрибутов могут выступать данные: количество, цена, статус, а в качестве операций – сформировать заказ.



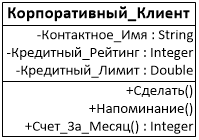
для класса *Заказ\_Оплата* в качестве атрибутов могут выступать данные: дата получения, проплачен, номер, цена, а в качестве операций – отправить, закрыть.



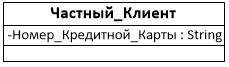
для класса *Клиент* в качестве атрибутов могут выступать данные: имя, адрес, а в качестве операций – кредитный рейтинг.



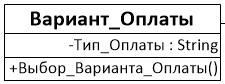
для класса *Корпоративный\_Клиент* в качестве атрибутов могут выступать данные: контактное имя, кредитный рейтинг, кредитный лимит, а в качестве операций – сделать, напоминание, счет за месяц.



для класса *Частный\_Клиент* в качестве атрибутов могут выступать данные: номер кредитной карты.



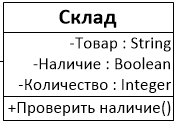
для класса *Вариант\_Оплаты* в качестве атрибутов могут выступать данные: тип оплаты, а в качестве операций – выбор варианта оплаты.



для класса *Каталог\_Товаров* в качестве атрибутов могут выступать данные: тип, марка, артикул, а в качестве операций – проверить наличие.

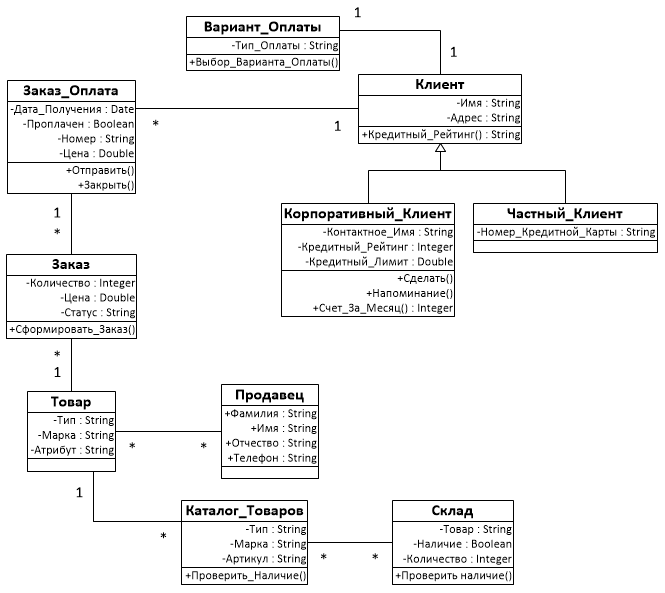


для класса *Склад* в качестве атрибутов могут выступать данные: товар, наличие, количество, а в качестве операций – Проверить наличие.

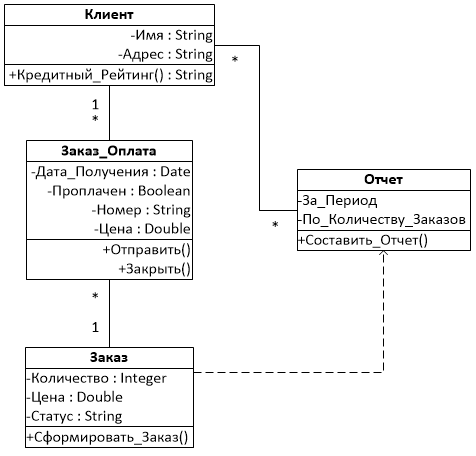


* Убедитесь, что все элементы наполнены адекватным содержанием и расположите все структурные элементы диаграммы наиболее оптимально на странице для установления отношений между ними.

В качестве примера на рис. 26 показан набор классов, описывающих реализацию системы продаж товаров по каталогу. Акцент сделан на классе *Клиент*, с которым связан класс *Заказ\_Оплата* посредством двусторонней ассоциации «один-ко-многим», *Вариант\_Оплаты* – двусторонней ассоциацией «один-к-одному» и классы *Корпоративный\_Клиент* и *Частный\_Клиент* посредством отношения обобщения. Классы *Заказ\_Оплата* и *Товар* связаны с классом *Заказ* посредством двусторонней ассоциации «один-ко-многим». Класс *Товар* связан с классом *Продавец* двусторонней ассоциацией «многие-ко-многим» и классом *Каталог\_Товаров* двусторонней ассоциацией «один-ко-многим». Класс *Каталог\_Товаров* связан посредством двусторонней ассоциации «многие-ко-многим» с классом *Склад.*



1. Создайте новую страницу с именем *Диаграмма классов учета клиентов*, и установите следующие опции: **Ориентация** – Альбомная, **Автоподбор размера** – выключен.
2. Идентифицируйте классы учета клиентов, осуществляющих заказы и создайте диаграмму классов с указанием их имен, атрибутов, операций, например.



**Задание практической работы**

По образцу построить диаграммы вариантов использования и классов.

**Задание самостоятельной работы**

В соответствии с индивидуальным вариантом, построить диаграммы вариантов использования и классов.

Перечень индивидуальных вариантов приведен в приложении А.

Отчет по практическому занятию выполняется в формате MS Word, который содержит экранные формы моделей согласно заданию.

**Контрольные вопросы**

1. Для чего используется язык UML?
2. Каково назначение диаграммы классов?
3. Назовите основные элементы диаграммы классов.
4. Какие виды связей доступны в диаграмме классов?
5. Для чего используется каждый вид связи?
6. Как создать диаграмму классов в VISIO?

# Практическая работа №14. Построение диаграммы Состояний

**Цель:** изучение основ создания диаграмм состояний на языке UML, получение навыков построения диаграмм состояний, применение приобретенных навыков для построения объектно-ориентированных моделей определенной предметной области.

**Краткие теоретические сведения**

***Диаграмма состояний*** описывает процесс изменения состояний только одного класса, а точнее – одного экземпляра определенного класса, т. е. моделирует все возможные изменения в состоянии конкретного объекта. При этом изменение состояния объекта может быть вызвано внешними воздействиями со стороны других объектов или извне. Именно для описания реакции объекта на подобные внешние воздействия и используются диаграммы состояний.

Диаграмма состояний описывает возможные последовательности состояний и переходов, которые в совокупности характеризуют поведение элемента модели в течение его жизненного цикла. Диаграмма состояний представляет динамическое поведение сущностей, на основе спецификации их реакции на восприятие некоторых конкретных событий.

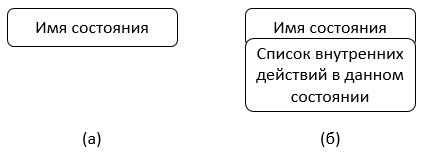
Хотя диаграммы состояний чаще всего используются для описания поведения отдельных экземпляров классов (объектов), но они также могут быть применены для спецификации функциональности других компонентов моделей, таких как варианты использования, актеры, подсистемы, операции и методы.

Диаграмма состояний по существу является графом специального вида, который представляет некоторый автомат. Понятие автомата в контексте UML обладает довольно специфической семантикой, основанной на теории автоматов. Вершинами этого графа являются состояния и некоторые другие типы элементов автомата (псевдосостояния), которые изображаются соответствующими графическими символами. Дуги графа служат для обозначения переходов из состояния в состояние. Диаграммы состояний могут быть вложены друг в друга, образуя вложенные диаграммы более детального представления отдельных элементов модели.



***Состояние***

Под состоянием понимается абстрактный метакласс, используемый для моделирования отдельной ситуации, в течение которой имеет место выполнение некоторого условия. Состояние может быть задано в виде набора конкретных значений атрибутов класса или объекта, при этом изменение их отдельных значений будет отражать изменение состояния моделируемого класса или объекта.



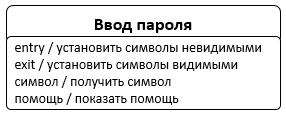
Секция «Список внутренних действий» содержит перечень внутренних действий или деятельностей, которые выполняются в процессе нахождения моделируемого элемента в данном состоянии. Каждое из действий записывается в виде отдельной строки и имеет следующий формат:

<метка-действия '/' выражение-действия>

**Метка действия** указывает на обстоятельства или условия, при которых будет выполняться деятельность, определенная выражением действия:

* ***entry*** – эта метка указывает на действие, специфицированное следующим за ней выражением действия, которое выполняется в момент входа в данное состояние (входное действие);
* ***exit*** – эта метка указывает на действие, специфицированное следующим за ней выражением действия, которое выполняется в момент выхода из данного состояния (выходное действие);
* ***do*** – эта метка специфицирует выполняющуюся деятельность («doactivity»), которая выполняется в течение всего времени, пока объект находится в данном состоянии, или до тех пор, пока не закончится вычисление, специфицированное следующим за ней выражением действия. В последнем случае при завершении события генерируется соответствующий результат;
* ***include*** – эта метка используется для обращения к подавтомату, при этом следующее за ней выражение действия содержит имя этого подавтомата.

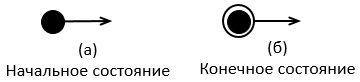
**Пример: Аутентификация входа**



***Начальное и конечное состояния***

***Начальное состояние*** представляет собой частный случай состояния, которое не содержит никаких внутренних действий (псевдосостояния). В этом состоянии находится объект по умолчанию в начальный момент времени. Оно служит для указания на диаграмме состояний графической области, от которой начинается процесс изменения состояний.

***Конечное (финальное) состояние*** представляет собой частный случай состояния, которое также не содержит никаких внутренних действий (псевдосостояния). В этом состоянии будет находиться объект по умолчанию после завершения работы автомата в конечный момент времени.



***Переход***

***Простой переход (simpletransition)*** представляет собой отношение между двумя последовательными состояниями, которое указывает на факт смены одного состояния другим. Пребывание моделируемого объекта в первом состоянии может сопровождаться выполнением некоторых действий, а переход во второе состояние будет возможен после завершения этих действий, а также после удовлетворения некоторых дополнительных условий. В этом случае говорят, что переход срабатывает, Или происходит срабатывание перехода. До срабатывания перехода объект находится в предыдущем от него состоянии, называемым исходным состоянием, или в источнике (не путать с начальным состоянием – это разные понятия), а после его срабатывания объект находится в последующем от него состоянии (целевом состоянии).

Переход осуществляется при наступлении некоторого события: окончания выполнения деятельности (doactivity), получении объектом сообщения или приемом сигнала. На переходе указывается имя события. Кроме того, на переходе могут указываться действия, производимые объектом в ответ на внешние события при переходе из одного состояния в другое. Срабатывание перехода может зависеть не только от наступления некоторого события, но и от выполнения определенного условия, называемого сторожевым условием. Объект перейдет из одного состояния в другое в том случае, если произошло указанное событие и сторожевое условие приняло значение «истина».

На диаграмме состояний ***переход изображается*** сплошной линией со стрелкой, которая направлена в целевое состояние. Каждый переход может помечен строкой текста, которая имеет следующий общий формат:

<сигнатура события>'['<сторожевое условие>']' <выражение действия>

***Событие***

***Событие*** представляет собой спецификацию некоторого факта, имеющего место в пространстве и во времени. Про события говорят, что они «происходят», при этом отдельные события должны быть упорядочены во времени. После наступления некоторого события нельзя уже вернуться к предыдущим событиям, если такая возможность не предусмотрена явно в модели.

События играют роль стимулов, которые инициируют переходы из одних состояний в другие. В качестве событий можно рассматривать сигналы, вызовы, окончание фиксированных промежутков времени или моменты окончания выполнения определенных действий. Имя события идентифицирует каждый отдельный переход на диаграмме состояний и может содержать строку текста, начинающуюся со строчной буквы.

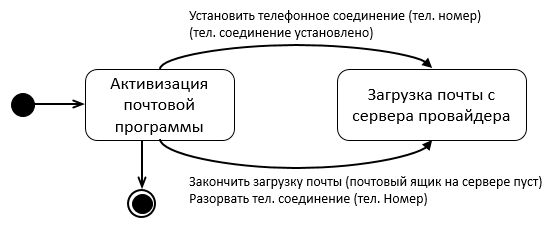
***Сторожевое условие***

***Сторожевое условие (guardcondition),*** если оно есть, всегда записывается в прямых скобках после события и представляет собой некоторое булевское выражение.

Если сторожевое условие принимает значение «истина», то соответствующий переход может сработать, в результате чего объект перейдет в целевое состояние. Если же сторожевое условие принимает значение «ложь», то переход не может сработать, и при отсутствии других переходов объект не может перейти в целевое состояние по этому переходу. Однако вычисление истинности сторожевого условия происходит только после возникновения ассоциированного с ним события, инициирующего соответствующий переход.

В общем случае из одного состояния может быть несколько переходов с одним и тем же событием-триггером. При этом никакие два сторожевых условия не должны одновременно принимать значение «истина». Каждое из сторожевых условий необходимо вычислять всякий раз при наступлении соответствующего события.

**Пример**

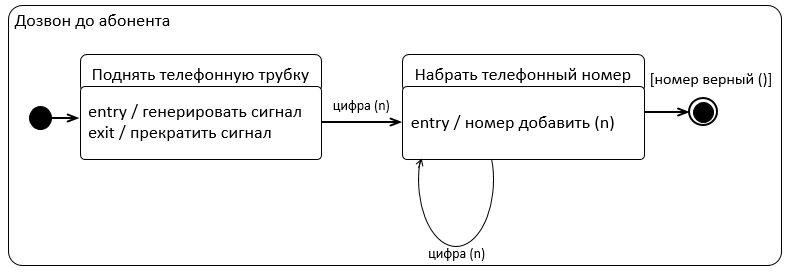


***Составное состояние и подсостояние***

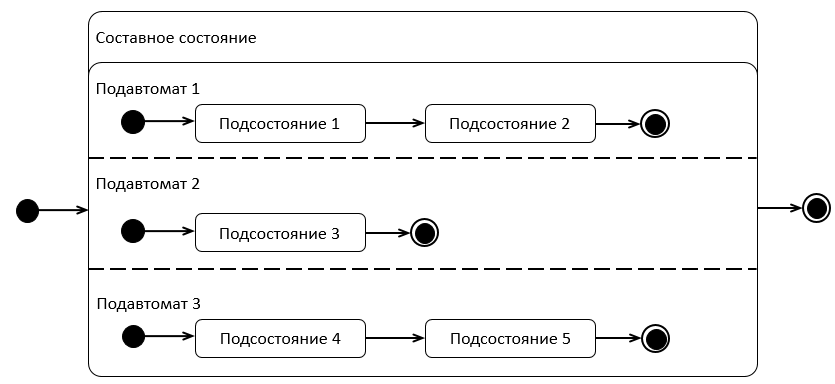
***Составное состояние (compositestate)*** – такое сложное состояние, которое состоит из других вложенных в него состояний. Последние будут выступать по отношению к первому как подсостояния (substate).



***Последовательные подсостояния (sequential substates)*** используются для моделирования такого поведения объекта, во время которого в каждый момент времени объект может находиться в одном и только одном подсостояний. Поведение объекта в этом случае представляет собой последовательную смену подсостояний, начиная от начального и заканчивая конечным подсостояниями. Хотя объект продолжает находиться в составном состоянии, введение в рассмотрение последовательных подсостояний позволяет учесть более тонкие логические аспекты его внутреннего поведения.



***Параллельные подсостояния (concurrentsubstates)*** позволяют специфицировать два и более подавтомата, которые могут выполняться параллельно внутри составного события. Каждый из подавтоматов занимает некоторую область (регион) внутри составного состояния, которая отделяется от остальных горизонтальной пунктирной линией. Если на диаграмме состояний имеется составное состояние с вложенными параллельными подсостояниями, то объект может одновременно находиться в каждом из этих подсостояний.



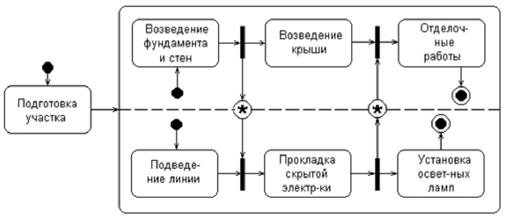
***Синхронизирующие состояния***

Поведение параллельных подавтоматов независимо друг от друга, что позволяет реализовать многозадачность в программных системах. Однако в отдельных случаях может возникнуть необходимость учесть в модели синхронизацию наступления отдельных событий. Для этой цели в языке UML имеется специальное псевдосостояние, которое называется синхронизирующим состоянием.

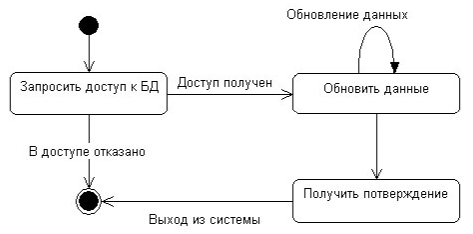
***Синхронизирующее состояние (synch state)*** обозначается небольшой окружностью, внутри которой помещен символ звездочки "\*". Оно используется совместно с переходом-соединением или переходом-ветвлением для того, чтобы явно указать события в других подавтоматах, оказывающие непосредственное влияние на поведение данного подавтомата.

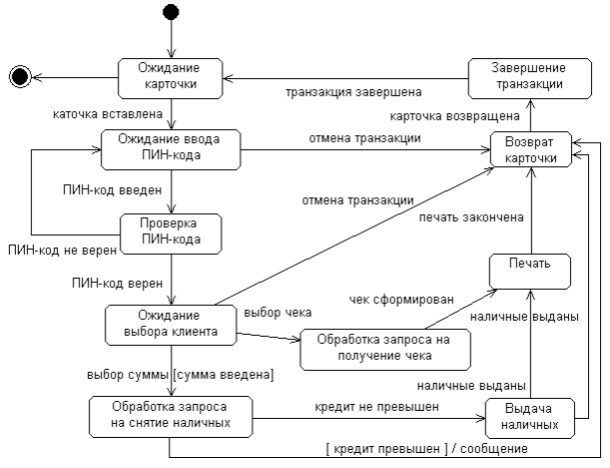
Для иллюстрации использования синхронизирующих состояний рассмотрим упрощенную ситуацию с моделированием процесса постройки дома. Предположим, что постройка дома включает в себя строительные работы (возведение фундамента и стен, возведение крыши и отделочные работы) и работы по электрификации дома (подведение электрической линии, прокладка скрытой электропроводки и установка осветительных ламп). Очевидно, два этих комплекса работ могут выполняться параллельно, однако между ними есть некоторая взаимосвязь.

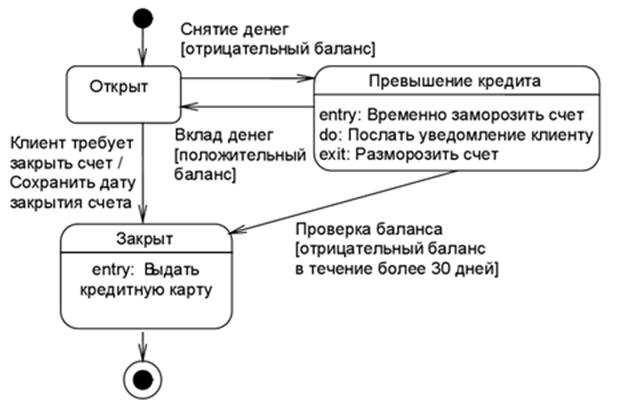
В частности, прокладка скрытой электропроводки может начаться лишь после того, как будет завершено возведение фундамента и стен. А отделочные работы следует начать лишь после того, как будет закончена прокладка скрытой электропроводки. В противном случае отделочные работы придется проводить повторно. Рассмотренные особенности синхронизации этих параллельных процессов учтены на соответствующей диаграмме состояний с помощью двух синхронизирующих состояний.



Примеры диаграмм состояний.



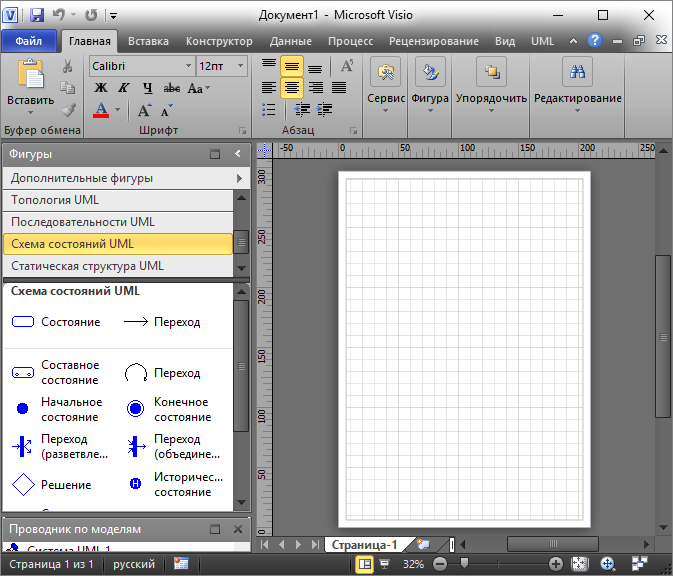




**Методика выполнения**

В качестве примера рассматривается моделирование системы продажи товаров по каталогу.

1. Запустите MS Visio.
2. На экране выбора шаблона выберите категорию *Программы и БД* и в ней элемент *Схема модели UML*. Нажмите кнопку *Создать* в правой части экрана.
3. Окно программы примет вид.



Клиент оформляет заказ. Класс Заказ имеет атрибут Статус. Проследим динамику движения заказов в системе с помощью ***диаграммы состояний, составленной для класса Заказ.***

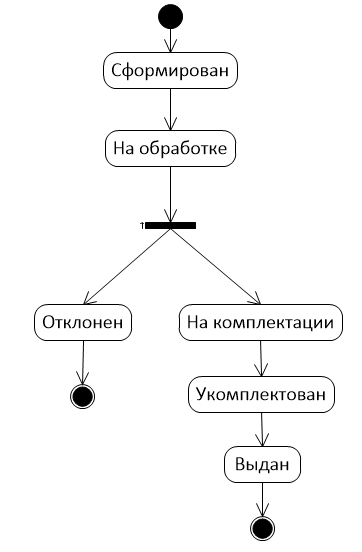
Данные о сформированном заказе поступают продавцу, который проверяет наличие товаров из заказа, проверяет оплату заказа, комплектует его и делает отметку о готовности. После оплаты заказа он выдается клиенту. Продавец делает отметку о том, что заказ выдан.

Если после проверки кредитного рейтинга клиента, он окажется отрицательным, то заказ будет отклонен.

Построим диаграмму состояний для класса Заказ. Для этого, в файле с диаграммой классов, созданной в практическом занятии 8, необходимо проделать следующие действия:

1. Щелкнуть правой кнопкой мыши по *классу Заказ*.
2. В контекстном меню выбрать пункт *Схемы*.
3. Т.к. в настоящее время уже созданных схем нет, нажать кнопку *Создать* и выбрать *Схема состояний*.
4. Переименовать созданный лист в *Схема состояний-Заказ*.
5. Построить диаграмму состояний для класса Заказ.

После формирования заказа он должен быть оплачен. Обработка заказа подразумевает проверку наличия товара и проверку оплаты. Переход в одно из состояний На комплектации, Укомплектован, Выдан означает смену Статуса заказа.



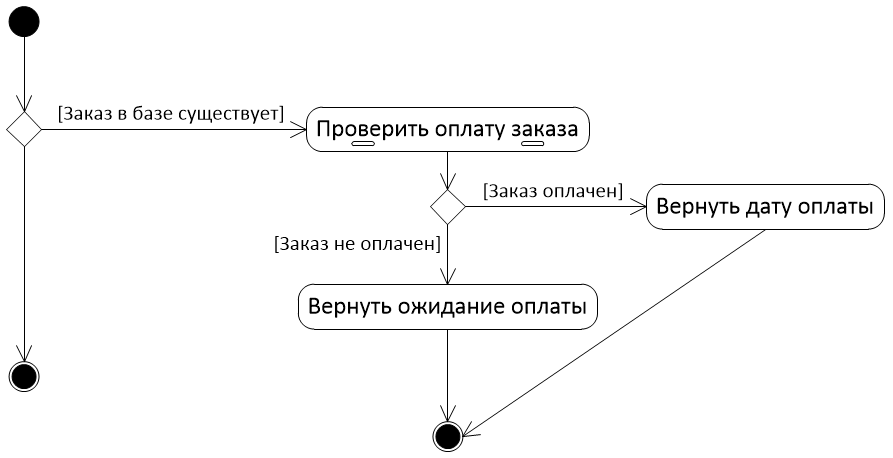
Далее опишем с помощью ***диаграммы состояний процесс оплаты заказа клиентом***, которому соответствует класс ЗаказОплата.

Построим диаграмму состояний для проверки оплаты заказа.

Чтобы проверить оплату заказа, необходимо определить, существует ли сам заказ. Результатом проверки оплаты заказа является вывод либо сообщения о произведенной оплате с параметрами (дата оплаты), либо сообщения об ожидании оплаты.

Событием, предшествующим проверке оплаты заказа, является занесение информации о заказе в базу данных заказов.

Чтобы построить диаграмму состояний для класса ЗаказОплаты, необходимо проделать действия, описанные в пунктах 1-4 построения диаграммы состояний для класса Заказ. Полученная диаграмма должна иметь вид, изображенный на рис. 15.



На этой диаграмме есть составное состояние ***«Проверить оплату заказа»***, т.к. оно включает в себя *проверку кредитного рейтинга клиента* и *проверку выбора варианта оплаты клиентом*.

Оплату заказа может произвести только клиент с положительным кредитным рейтингом, поэтому необходимым условием проверки оплаты заказа является проверка кредитоспособности клиента. Если клиент имеет отрицательный кредитный рейтинг, то заказ отклоняется, и на этом дальнейшие события не имеют смысла.

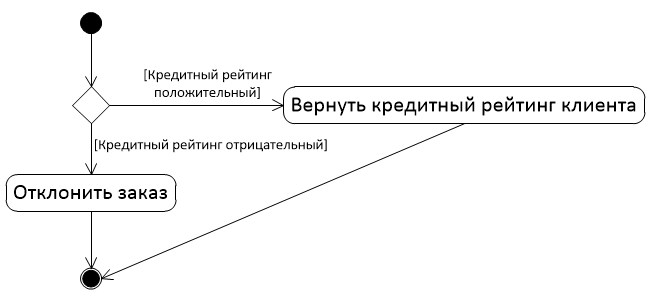
Если кредитный рейтинг клиента положительный, то необходимо проверить, выбрал ли клиент вариант оплаты. Событие, которое переводит систему в состояние ожидания выбора варианта оплаты клиентом, является получение сообщения о кредитоспособности клиента.

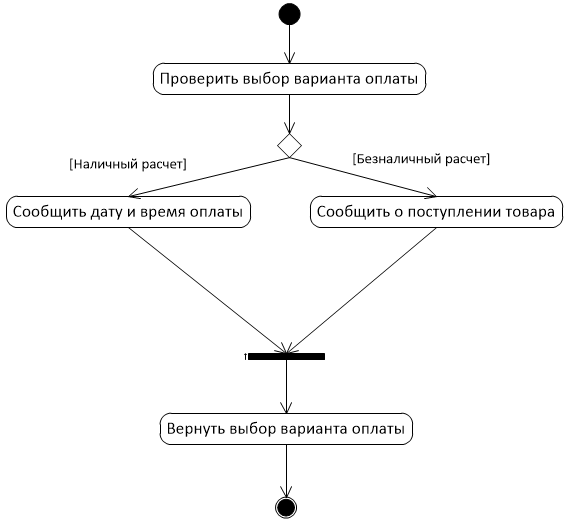
Оплата может быть произведена наличными средствами в магазине или с помощью безналичного расчета. В первом случае необходимо договориться с клиентом о дате и времени его прибытия в магазин. Во втором случае необходимо сообщить клиенту о наличии/поступлении товара. Событие, которое переводит систему в состояние ожидания оплаты, является выбор клиентом варианта оплаты.

Соответствующие диаграммы состояний имеют вид

Для создания диаграмм состояний, которые входят в состав составного состояния, нужно:

1. Щелкнуть правой кнопкой мыши по Составному состоянию и выбрать пункт Схема.
2. Либо, в Проводнике по моделям выделить название составного состояния и создать новую страницу с помощью кнопки .





**Задание практической работы**

По образцу построить диаграмму состояний.

**Задание самостоятельной работы**

В соответствии с индивидуальным вариантом, построить диаграмму состояний.

Перечень индивидуальных вариантов приведен в приложении А.

Отчет по практическому занятию выполняется в формате MS Word, который содержит экранные формы моделей согласно заданию.

**Контрольные вопросы**

1. Каково назначение диаграммы состояний?
2. Назовите основные элементы диаграммы состояний.
3. Как создать диаграмму состояний в VISIO?
4. В чем отличие диаграммы классов и состояний?

# Практическая работа №15. Построение диаграммы Деятельности

**Цель:** изучение основ создания диаграмм деятельности на языке UML, получение навыков построения диаграмм деятельности, применение приобретенных навыков для построения объектно-ориентированных моделей определенной предметной области.

**Краткие теоретические сведения**

**Построение диаграммы Деятельности**

При моделировании поведения проектируемой или анализируемой системы возникает необходимость не только представить процесс изменения ее состояний, но и детализировать особенности алгоритмической и логической реализации выполняемых системой операций. Традиционно для этой цели использовались блок-схемы или структурные схемы алгоритмов. Каждая такая схема акцентирует внимание на последовательности выполнения определенных действий или элементарных операций, которые в совокупности приводят к получению желаемого результата.

Алгоритмические и логические операции, требующие выполнения в определенной последовательности, окружают нас постоянно. Например, чтобы позвонить по телефону, нам предварительно нужно снять трубку или включить его. Для приготовления кофе или заваривания чая необходимо вначале вскипятить воду. Чтобы выполнить ремонт двигателя автомобиля, требуется осуществить целый ряд нетривиальных операций, таких как разборка силового агрегата, снятие генератора и некоторых других.

C увеличением сложности системы строгое соблюдение последовательности выполняемых операций приобретает все более важное значение. Если попытаться заварить кофе холодной водой, то мы можем только испортить одну порцию напитка. Нарушение последовательности операций при ремонте двигателя может привести к его поломке или выходу из строя. Еще более катастрофические последствия могут произойти в случае отклонения от установленной последовательности действий при взлете или посадке авиалайнера, запуске ракеты, регламентных работах на АЭС.

Для моделирования процесса выполнения операций в языке UML используются так называемые **диаграммы деятельности**. Применяемая в них графическая нотация во многом похожа на нотацию диаграммы состояний, поскольку на диаграммах деятельности также присутствуют обозначения состояний и переходов. Отличие заключается в семантике состояний, которые используются для представления не деятельностей, а действий, и в отсутствии на переходах сигнатуры событий. Каждое состояние на диаграмме деятельности соответствует выполнению некоторой элементарной операции, а переход в следующее состояние срабатывает только при завершении этой операции в предыдущем состоянии. Графически диаграмма деятельности представляется в форме графа деятельности, вершинами которого являются состояния действия, а дугами – переходы от одного состояния действия к другому.

Таким образом, диаграммы деятельности можно считать частным случаем диаграмм состояний. Основным направлением использования диаграмм деятельности является визуализация особенностей реализации операций классов, когда необходимо представить алгоритмы их выполнения. При этом каждое состояние может являться выполнением операции некоторого класса либо ее части, позволяя использовать диаграммы деятельности для описания реакций на внутренние события системы.

В контексте языка UML **деятельность (activity)** представляет собой некоторую совокупность отдельных вычислений, выполняемых автоматом. При этом отдельные элементарные вычисления могут приводить к некоторому результату или действию (action). На диаграмме деятельности отображается логика или последовательность перехода от одной деятельности к другой, при этом внимание фиксируется на результате деятельности. Сам же результат может привести к изменению состояния системы или возвращению некоторого значения.

***Состояние действия и деятельности***

***Состояние деятельности (activity state)*** – состояние в графе деятельности, которое служит для представления процедурной последовательности действий, требующих определенного времени. Переход из состояния деятельности происходит после выполнения специфицированной в нем ду-деятельности, при этом ключевое слово do в имени деятельности не указывается. Состояние деятельности не может иметь внутренних переходов, поскольку оно является элементарным.

Состояния деятельности могут быть подвергнуты дальнейшей декомпозиции, вследствие чего выполняемую деятельность можно представить с помощью других диаграмм деятельности. Состояния деятельности не являются атомарными, то есть могут быть прерваны. Предполагается, что для их завершения требуется заметное время.

Состояние деятельности можно представлять себе, как составное состояние, поток управления которого включает только другие состояния деятельности и действий.

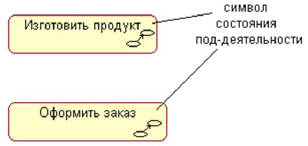
***Состояние действия (action state)*** является специальным случаем состояния с некоторым входным действием и, по крайней мере, одним выходящим из состояния переходом. Этот переход неявно предполагает, что входное действие уже завершилось. Состояние действия не может иметь внутренних переходов, поскольку оно является элементарным. Обычное использование состояния действия заключается в моделировании одного шага выполнения алгоритма (процедуры) или потока управления.

Графически состояние действия изображается фигурой, напоминающей прямоугольник, боковые стороны которого заменены выпуклыми дугами. Внутри этой фигуры записывается выражение действия (action-expression), которое должно быть уникальным в пределах одной диаграммы деятельности.



Действие может быть записано на естественном языке, некотором псевдокоде или языке программирования. Никаких дополнительных или неявных ограничений при записи действий не накладывается. Рекомендуется в качестве имени простого действия использовать глагол с пояснительными словами. Если же действие может быть представлено в некотором формальном виде, то целесообразно записать его на том языке программирования, на котором предполагается реализовывать конкретный проект.

Иногда возникает необходимость представить на диаграмме деятельности некоторое сложное действие, которое, в свою очередь, состоит из нескольких более простых действий. В этом случае можно использовать специальное обозначение так называемого **состояния поддеятельности (subactivity state)**. Такое состояние является графом деятельности и обозначается специальной пиктограммой в правом нижнем углу символа состояния действия (рис. 2). Эта конструкция может применяться к любому элементу языка UML, который поддерживает «вложенность» своей структуры. При этом пиктограмма может быть дополнительно помечена типом вложенной структуры.



Каждая диаграмма деятельности должна иметь единственное начальное и единственное конечное состояния. Они имеют такие же обозначения, как и на диаграмме состояний. При этом каждая деятельность начинается в начальном состоянии и заканчивается в конечном состоянии. Саму диаграмму деятельности принято располагать таким образом, чтобы действия следовали сверху вниз. В этом случае начальное состояние будет изображаться в верхней части диаграммы, а конечное – в ее нижней части.

***Переходы***

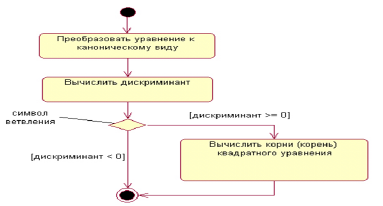
При построении диаграммы деятельности используются только такие переходы, которые срабатывают сразу после завершения деятельности или выполнения соответствующего действия. Этот переход переводит деятельность в последующее состояние сразу, как только закончится действие в предыдущем состоянии. На диаграмме такой переход изображается сплошной линией со стрелкой.

Если из состояния действия выходит единственный переход, то он может быть никак не помечен. Если же таких переходов несколько, то сработать может только один из них. Именно в этом случае для каждого из таких переходов должно быть явно записано сторожевое условие в прямых скобках. При этом для всех выходящих из некоторого состояния переходов должно выполняться требование истинности только одного из них. Подобный случай встречается тогда, когда последовательно выполняемая деятельность должна разделиться на альтернативные ветви в зависимости от значения некоторого промежуточного результата. Такая ситуация получила название ветвления, а для ее обозначения применяется специальный символ.

**Графически ветвление** на диаграмме деятельности обозначается небольшим ромбом, внутри которого нет никакого текста. В этот ромб может входить только одна стрелка от того состояния действия, после выполнения которого поток управления должен быть продолжен по одной из взаимно исключающих ветвей. Принято входящую стрелку присоединять к верхней или левой вершине символа ветвления. Выходящих стрелок может быть две или более, но для каждой из них явно указывается соответствующее сторожевое условие в форме булевского выражения.

В качестве **примера** рассмотрим фрагмент известного алгоритма нахождения корней квадратного уравнения. В общем случае после приведения уравнения второй степени к каноническому виду: необходимо вычислить его дискриминант. Причем, в случае отрицательного дискриминанта уравнение не имеет решения на множестве действительных чисел, и дальнейшие вычисления должны быть прекращены. При неотрицательном дискриминанте уравнение имеет решение, корни которого могут быть получены на основе конкретной расчетной формулы.

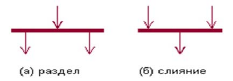
Графически фрагмент процедуры вычисления корней квадратного уравнения может быть представлен в виде диаграммы деятельности с тремя состояниями действия и ветвлением (рис. 3). Каждый из переходов, выходящих из состояния «Вычислить дискриминант», имеет сторожевое условие, определяющее единственную ветвь, по которой может быть продолжен процесс вычисления корней в зависимости от знака дискриминанта. Очевидно, что в случае его отрицательности, мы сразу попадаем в конечное состояние, тем самым завершая выполнение алгоритма в целом.



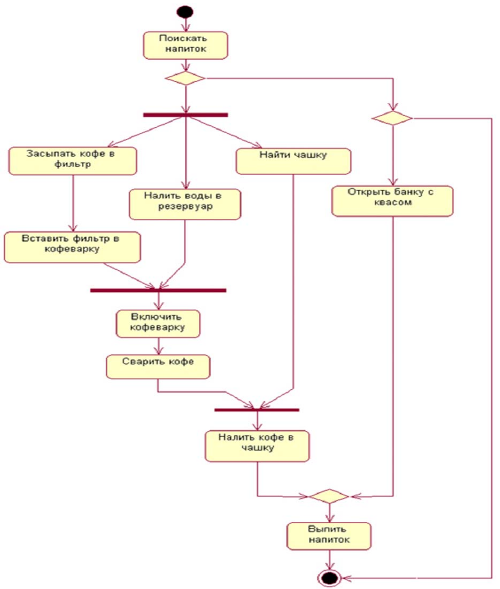
В рассмотренном примере, выполняемые действия соединяются в конечном состоянии. Однако это вовсе не является обязательным. Можно изобразить еще один символ ветвления, который будет иметь несколько входящих переходов и один выходящий.

Один из наиболее значимых недостатков обычных блок-схем или структурных схем алгоритмов связан с проблемой изображения параллельных ветвей отдельных вычислений. Поскольку распараллеливание вычислений существенно повышает общее быстродействие программных систем, необходимы графические примитивы для представления параллельных процессов. В языке UML для этой цели используется специальный символ для разделения и слияния параллельных вычислений или потоков управления. Таким символом является **прямая черточка**.

Такая черточка изображается отрезком горизонтальной линии, толщина которой несколько шире основных сплошных линий диаграммы деятельности. При этом разделение (concurrent fork) имеет один входящий переход и несколько выходящих. Слияние (concurrent join), наоборот, имеет несколько входящих переходов и один выходящий.



Для иллюстрации особенностей параллельных процессов выполнения действий рассмотрим **пример** с приготовлением напитка. Достоинство этого примера состоит в том, что он практически не требует никаких дополнительных пояснений в силу своей очевидности.



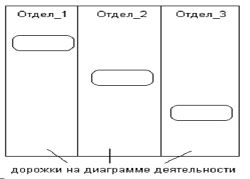
Хотя диаграмма деятельности предназначена для моделирования поведения систем, время в явном виде отсутствует на этой диаграмме. Ситуация здесь во многом аналогична диаграмме состояний.

Таким образом, диаграмма деятельности есть не что иное, как специальный случай диаграммы состояний, в которой все или большинство состояний являются действиями или состояниями поддеятельности. А все или большинство переходов являются переходами, которые срабатывают по завершении действий или под-деятельностей в состояниях источниках.

***Дорожки***

Диаграммы деятельности могут быть использованы не только для спецификации алгоритмов вычислений или потоков управления в программных системах. Не менее важная область их применения связана с моделированием бизнес-процессов. Действительно, деятельность любой компании (фирмы) также представляет собой не что иное, как совокупность отдельных действий, направленных на достижение требуемого результата. Однако применительно к бизнес-процессам желательно выполнение каждого действия ассоциировать с конкретным подразделением компании. В этом случае подразделение несет ответственность за реализацию отдельных действий, а сам бизнес-процесс представляется в виде переходов действий из одного подразделения к другому.

Для моделирования этих особенностей в языке UML используется специальная конструкция, получившее название ***дорожки (swimlanes)***. Имеется в виду визуальная аналогия с плавательными дорожками в бассейне, если смотреть на соответствующую диаграмму. При этом все состояния действия на диаграмме деятельности делятся на отдельные группы, которые отделяются друг от друга вертикальными линиями. Две соседние линии и образуют дорожку, а группа состояний между этими линиями выполняется отдельным подразделением (отделом, группой, отделением, филиалом) компании.



Названия подразделений явно указываются в верхней части дорожки. Пересекать линию дорожки могут только переходы, которые в этом случае обозначают выход или вход потока управления в соответствующее подразделение компании. Порядок следования дорожек не несет какой-либо семантической информации и определяется соображениями удобства.

В качестве примера рассмотрим фрагмент диаграммы деятельности торговой компании, обслуживающей клиентов по телефону. Подразделениями компании являются отдел приема и оформления заказов, отдел продаж и склад.

Этим подразделениям будут соответствовать три дорожки на диаграмме деятельности, каждая из которых специфицирует зону ответственности подразделения. В данном случае диаграмма деятельности заключает в себе не только информацию о последовательности выполнения рабочих действий, но и о том, какое из подразделений торговой компании должно выполнять то или иное действие.

Из указанной диаграммы деятельности сразу видно, что после принятия заказа от клиента отделом приема и оформления заказов осуществляется распараллеливание деятельности на два потока (переход-разделение). Первый из них остается в этом же отделе и связан с получением оплаты от клиента за заказанный товар. Второй инициирует выполнение действия по подбору товара в отделе продаж (модель товара, размеры, цвет, год выпуска и пр.). По окончании этой работы инициируется действие по отпуску товара со склада. Однако подготовка товара к отправке в торговом отделе начинается только после того, как будет получена оплата за товар от клиента и товар будет отпущен со склада (переход-соединение). Только после этого товар отправляется клиенту, переходя в его собственность.

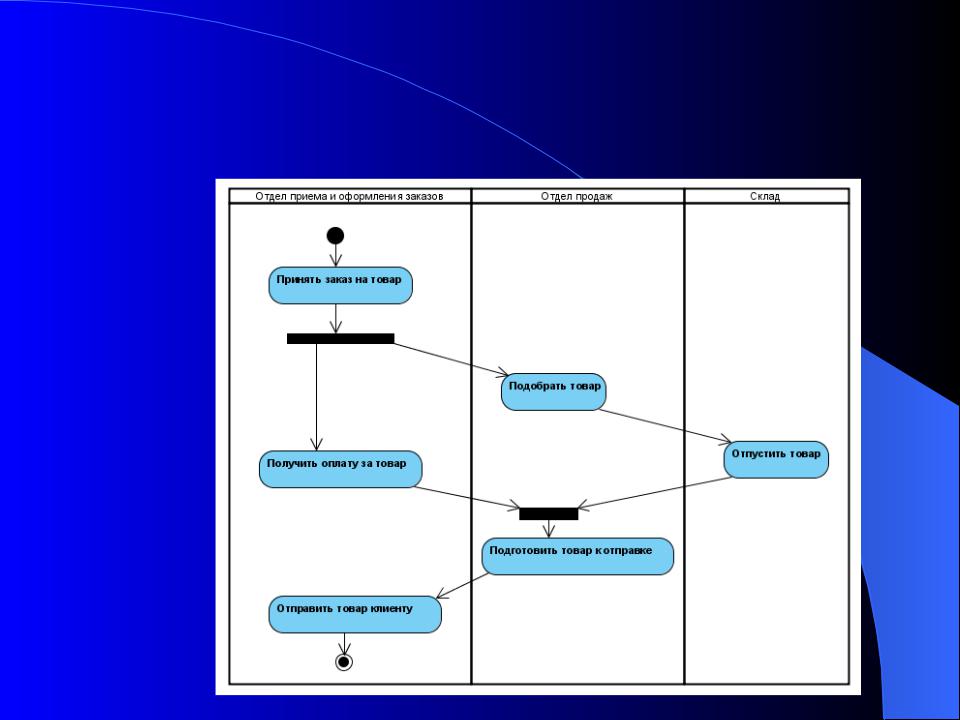
***Объекты***

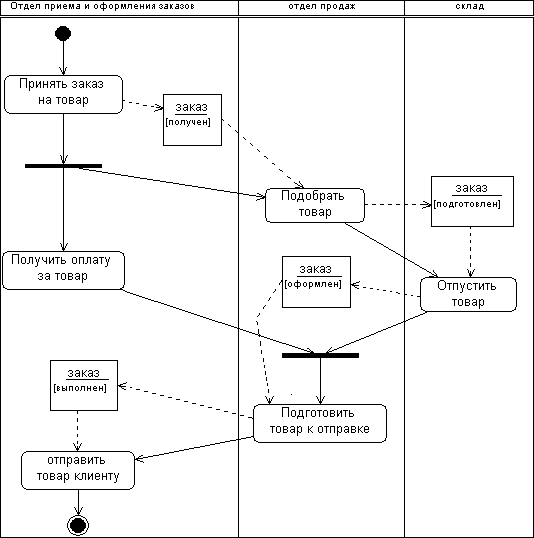
В общем случае действия на диаграмме деятельности выполняются над теми или иными объектами. Эти объекты либо инициируют выполнение действий, либо определяют некоторый результат этих действий. При этом действия специфицируют вызовы, которые передаются от одного объекта графа деятельности к другому. Поскольку в таком ракурсе объекты играют определенную роль в понимании процесса деятельности, иногда возникает необходимость явно указать их на диаграмме деятельности.

Для **графического представления объектов** используются прямоугольник класса, с тем отличием, что имя объекта подчеркивается. Далее после имени может указываться характеристика состояния объекта в прямых скобках. Такие прямоугольники объектов присоединяются к состояниям действия отношением зависимости пунктирной линией со стрелкой. Соответствующая зависимость определяет состояние конкретного объекта после выполнения предшествующего действия.

На диаграмме деятельности с дорожками расположение объекта может иметь некоторый дополнительный смысл. А именно, если объект расположен на границе двух дорожек, то это может означать, что переход к следующему состоянию действия в соседней дорожке ассоциирован с готовностью некоторого документа (объект в некотором состоянии). Если же объект целиком расположен внутри дорожки, то и состояние этого объекта целиком определяется действиями данной дорожки.

Возвращаясь к предыдущему примеру с торговой компанией, можно заметить, что центральным объектом процесса продажи является заказ или вернее состояние его выполнения. Вначале до звонка от клиента заказ как объект отсутствует и возникает лишь после такого звонка. Однако этот заказ еще не заполнен до конца, поскольку требуется еще подобрать конкретный товар в отделе продаж. После его подготовки он передается на склад, где вместе с отпуском товара заказ окончательно дооформляется. Наконец, после получения подтверждения об оплате товара эта информация заносится в заказ, и он считается выполненным и закрытым. Данная информация может быть представлена графически в виде модифицированного варианта диаграммы деятельности этой же торговой компании.



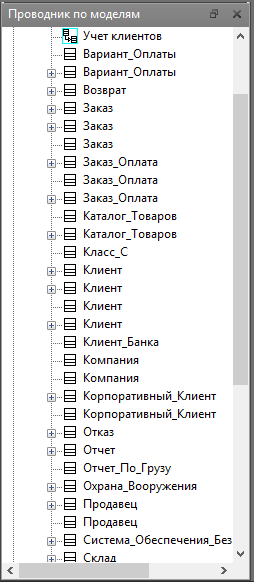


**Методика выполнения**

**Построение диаграммы Деятельности**

В качестве примера рассматривается моделирование системы продажи товаров по каталогу.

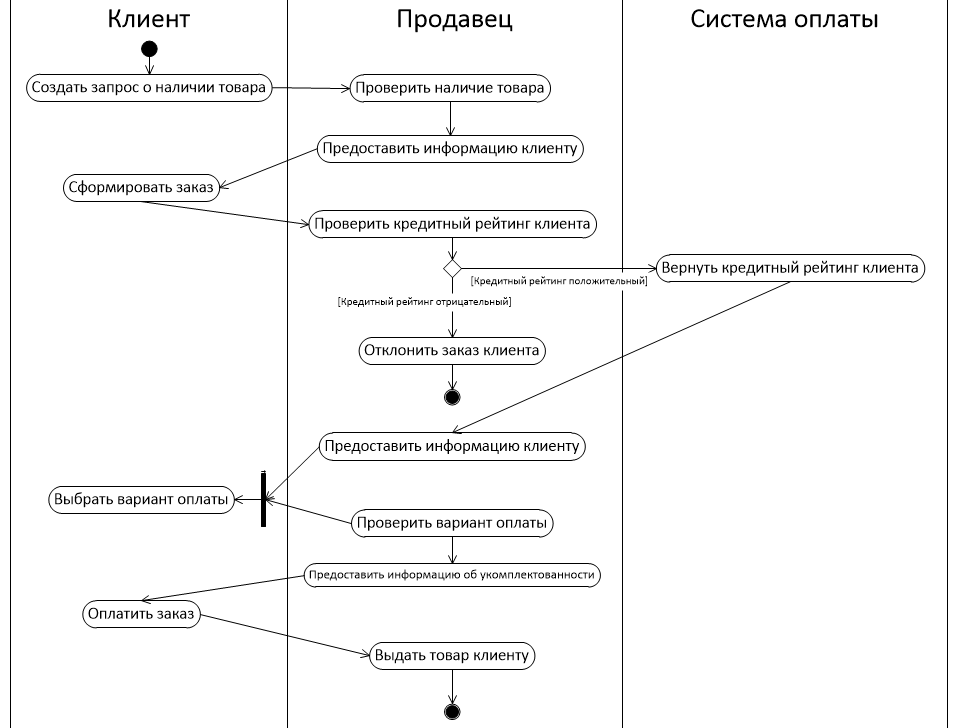
1. Запустите MS Visio.
2. **Откройте файл, созданный в предыдущей работе и содержащий диаграмму классов.**
3. В проводнике по моделям должны отображаться все классы и диаграммы, созданные ранее.



Опишем с помощью диаграммы деятельности процесс формирования заказа и выдачу товара. В бизнес-процессе участвуют 3 действующих лица: клиент, продавец и система оплаты. Следовательно, необходимо добавить 3 дорожки для распределения ответственности между этими лицами.

Для этого, в файле с диаграммой классов, созданной в практическом занятии 8, необходимо проделать следующие действия:

1. Щелкнуть правой кнопкой мыши по *классу Заказ*.
2. В контекстном меню выбрать пункт *Схемы*.
3. Нажать кнопку *Создать* и выбрать *Деятельность*.
4. Переименовать созданный лист в *Деятельность-Заказ*.
5. Построить диаграмму деятельности для класса Заказ. Для это выполните действия, описанные ниже.
   1. Добавить 3 элемента *Дорожка* и изменить их названия на *Клиент*, *Продавец* и *Система оплаты* соответственно.
   2. Добавить элементы *Начальное состояние*, *Конечное состояние, Состояние действия, Решение, Переход (объединение),* изменить их названия и задать расположение.



**Задание практической работы**

По образцу построить диаграмму состояний.

**Задание самостоятельной работы**

В соответствии с индивидуальным вариантом, построить диаграмму состояний.

Перечень индивидуальных вариантов приведен в приложении А.

Отчет по практическому занятию выполняется в формате MS Word, который содержит экранные формы моделей согласно заданию.

**Контрольные вопросы**

1. Дайте определение понятию «диаграмма деятельности».
2. Опишите назначение диаграммы деятельности.
3. Дайте определение понятиям «состояние деятельности» и «состояние действия». Графическое изображение состояния.
4. Приведите пример ветвления и параллельных потоков управления процессами на диаграмме деятельности.
5. Какие переходы используются на диаграмме деятельности?
6. Что представляет собой дорожка на диаграмме деятельности?
7. Как графически изображаются объекты на диаграмме деятельности?

# Практическая работа №16. Построение диаграммы Последовательности

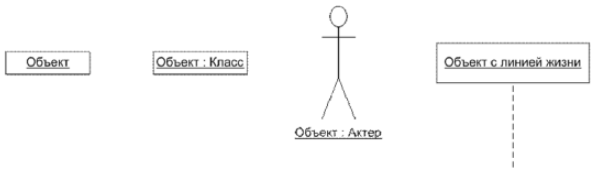
**Цель:** изучение основ создания диаграмм последовательностей на языке UML, получение навыков построения диаграмм последовательностей, применение приобретенных навыков для построения объектно-ориентированных моделей определенной предметной области.

**Краткие теоретические сведения**

**Построение диаграммы Последовательности**

Диаграммы последовательностей описывают взаимодействия множества объектов, включая сообщения, которыми они обмениваются.

В отличие от диаграммы классов, на которой изображаются абстрактные элементы в виде классов, на диаграмме последовательностей используются конкретные экземпляры классов – ***объекты***. Объекты отображаются прямоугольником без полей. Для того чтобы подчеркнуть, что это экземпляр абстрактной сущности, название объекта подчеркивается. При необходимости через двоеточие после названия можно указать сущность (класс) экземпляром которой является этот объект. Отметим, что объект может быть экземпляром не только класса, но и других абстракций, например, актера. Обратите внимание, что при указании в качестве классификатора актера изменится графическое обозначение объекта.

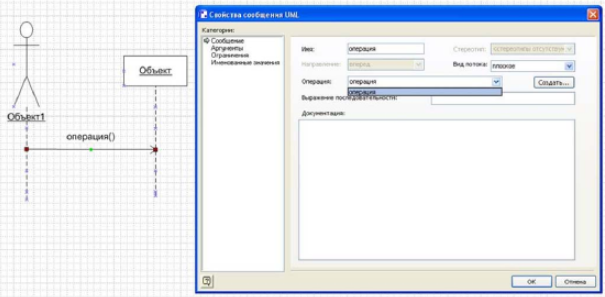


На диаграмме последовательностей у объекта может присутствовать ***линия жизни***, на которой отмечаются происходящие с объектом события. Линия жизни отображается пунктирной линией.

Между собой объекты могут быть связаны связями. ***Связь*** – это экземпляр отношения ассоциация, и имеет такое же графическое обозначение, что и ассоциация.

На диаграмме последовательностей объекты обмениваются сообщениями. ***Сообщение*** – это спецификация передачи данных от одного объекта другому, который предполагает какое-то ответное действие. Графически сообщение обозначается сплошной линией со стрелкой.

Часто операция вызывает какую-либо операцию в объекте. Очевидно, что класс, экземпляром которого является объект, должен иметь такую операцию. Привязка сообщения к операции класса объекта выполняется в свойствах сообщения.



При построении диаграммы классов обычно определяются только основные свойства сущностей, а такие детали, как операции, удобно создавать при построении диаграммы последовательности, для чего в свойствах сообщения UML есть кнопка создания операции.

Диаграммы последовательностей, как и другие диаграммы для отображения динамических свойств системы, могут быть выполнены в контексте многих сущностей UML. Они могут описывать поведение системы в целом, подсистемы, класса или операции класса и др. К сожалению, Visio недостаточно гибка в плане поддержки раскрытия содержания отдельных элементов с помощью других диаграмм. Например, кликнув правой кнопкой мыши по классу можно обнаружить, что для его описания можно создать лишь диаграммы классов, состояний и деятельности. Поэтому возможность привязать диаграмму последовательностей к элементу, который она реализует, средствами Visio невозможно, эту связь нужно подразумевать.

Диаграммы последовательностей будем делать в контексте прецедентов с диаграммы прецедентов, реализуя те функции, которые должна выполнять наша система.

При построении динамических диаграмм используется уже разработанная структура информационной системы. Для диаграммы последовательностей не нужно придумывать объекты, а достаточно определить, экземпляры каких классов участвуют в этом действии.

Определив необходимые объекты (как экземпляры классов, так и экземпляры актеров), вторым этапом построения диаграмм последовательностей определяются сообщения, пересылаемые между актерами. Фактически определяется последовательность шагов, для выполнения нужного действия.

**Методика выполнения**

**Построение диаграммы Последовательности**

1. В проводнике по моделям должны отображаться все классы и диаграммы, созданные ранее.

Построим диаграмму последовательности для варианта использования «Обеспечить покупателя информацией». Для этого добавим на диаграмму последовательности линии жизни и соотнесем объекты с актерами, инициирующими вариант использования «Обеспечить покупателя информацией», и с необходимыми классами.

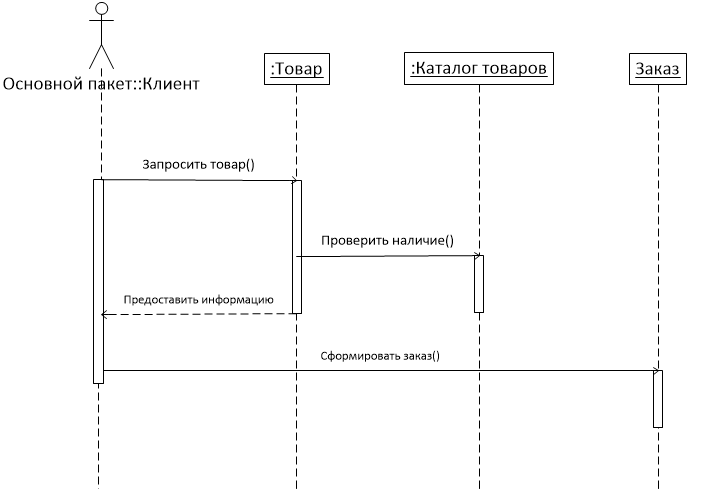
Для **добавления диаграммы последовательности в проект MS Visio** выполните следующие действия:

1. В проводнике по моделям найдите ветку «Основной пакет».
2. Нажмите по ней правой кнопкой мыши > Создать …
3. В контекстном меню выберите пункт «Схема последовательностей».

Добавим сообщения, которыми обмениваются объекты для исполнения варианта использования.

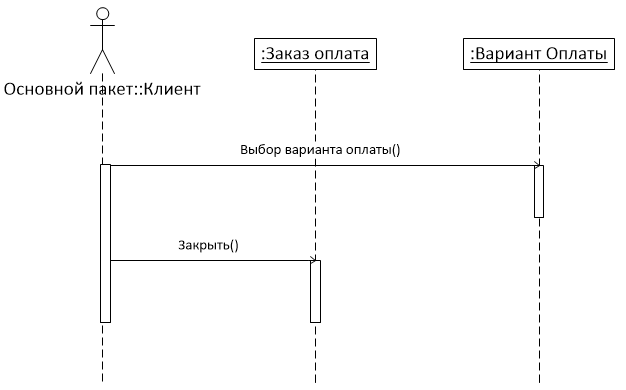
Если объект имеет операцию (посмотреть в практическом занятии №8 «Диаграммы классов» наличие операции у класса, которому принадлежит объект).

1. Из группы фигур «Последовательности UML» добавить три фигуры типа «Линия жизни объекта». Для изменения названия необходимо дважды щелкнуть левой кнопкой мыши по фигуре. Откроется окно свойств объекта и, если в данном файле нет ранее созданных классов, окно создания нового класса. В данном примере необходимо создать три класса «Товар», «Каталог товаров» и «Заказ» и соответственно три объекта с такими же названиями.
2. С помощью поиска фигур найти фигуру «Актер» и добавить ее в рабочую область. Двойным щелчком левой кнопки мыши задать имя «Клиент».
3. Добавить фигуру «Линия жизни» и соедините ее начало с фигурой «Клиент».
4. Протянуть все линии жизни вниз листа.
5. Добавить фигуры «Сообщение» и соединить, руководствуясь следующими принципами:
   1. Соединить фигурой «Сообщение» линию жизни клиента с линией жизни объекта товар. Двойным щелчком по сообщению открыть окно свойств и выбрать операцию запросить товар.
   2. Соединить фигурой «Сообщение» линию жизни клиента с линией жизни объекта заказ. Двойным щелчком по сообщению открыть окно свойств и выбрать операцию сформировать заказ.
   3. Соединить фигурой «Сообщение» линию жизни объекта товар с линией жизни объекта каталог товаров. Двойным щелчком по сообщению открыть окно свойств и выбрать операцию проверить наличие.
   4. Соединить фигурой «Сообщение (возврат)» линию жизни объекта товар и линию жизни клиента. Двойным щелчком по сообщению открыть окно свойств и задать текст сообщения «Предоставить информацию».
6. Добавить фигуры «Активация» и расположить их на диаграмме.

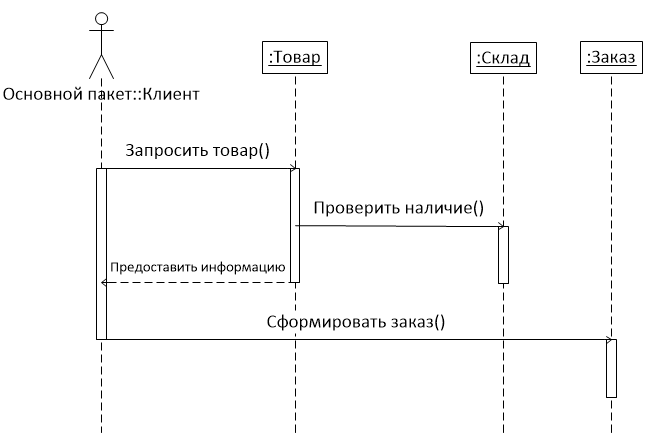


При построении диаграмм последовательностей можно вносить коррективы в диаграмму классов. Если объект класса получает новую операцию, то она добавляется в соответствующий класс на диаграмме классов как метод.

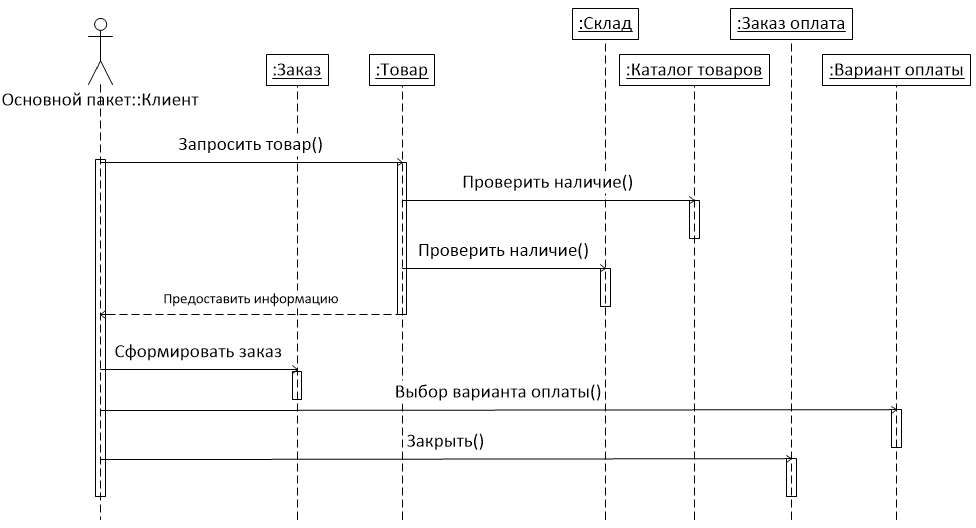
Построим диаграмму последовательности для варианта использования «Согласовать условия оплаты». Действия по построению диаграммы аналогичны построению диаграммы последовательности для варианта использования «Обеспечить покупателя информацией».



Построим диаграмму последовательности для варианта использования «Заказать товар со склада». Действия по построению диаграммы аналогичны построению предыдущих диаграмм последовательностей.



Построим диаграмму последовательности для системы продажи товаров по каталогу.



**Задание практической работы**

По образцу построить диаграмму состояний.

**Задание самостоятельной работы**

В соответствии с индивидуальным вариантом, построить диаграмму состояний.

Перечень индивидуальных вариантов приведен в приложении А.

Отчет по практическому занятию выполняется в формате MS Word, который содержит экранные формы моделей согласно заданию.

# Практическая работа №17. Оценка экономической эффективности информационной системы

**Цель работы:** изучение методов оценки экономической эффективности информационных систем

**Методика выполнения**

**Пример расчета экономической эффективности**

**Расчет затрат**

Разработка базы данных должна быть проведена в три стадии:

* разработка технического задания (далее – ТЗ);
* рабочее проектирование;
* внедрение.

На стадии разработки ТЗ должны быть выполнены перечисленные ниже работы:

* постановка задачи;
* определение и уточнение требований к техническим средствам;
* определение требований к системе;
* определение стадий, этапов и сроков разработки системы и документации;
* согласование и утверждение ТЗ.

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

* разработка системы;
* разработка программной документации;
* испытания системы.

На стадии внедрения должны быть выполнены подготовка и передача системы.

Для расчета временных затрат и координации проекта, разработчиками проекта был разработан календарный график проектных работ, показанный на рисунке 1.

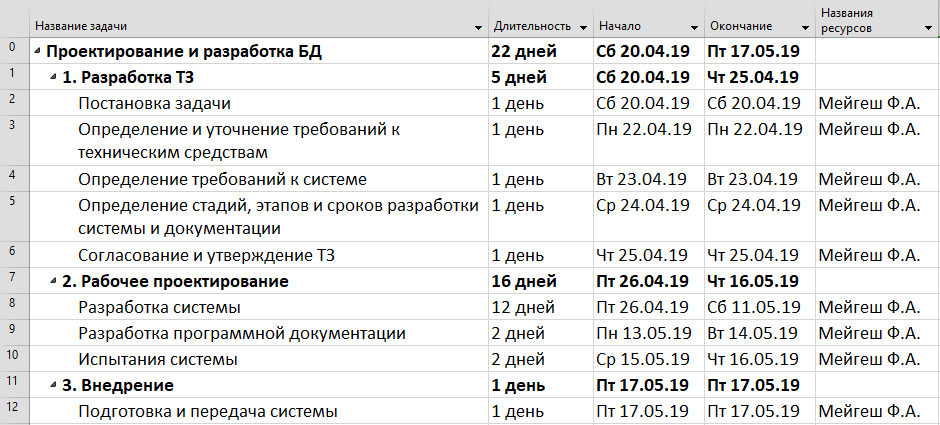


Рисунок 1 – Календарный график проектных работ

На разработку БД, согласно календарного графика работы, приходится 22 дня. Ответственным лицом за выполнение проектных работ является разработчик – студент группы СИС41 – Иванов И.И. Работы проводились в соответствии с техническим заданием. Начало проектных работ 20.04.2019, окончание – 17.05.2019.

Временные затраты на выполнение проекта сведены в таблицу 1 и составляют 132 часа.

Таблица 1 – Временные затраты на выполнение проекта

|  |  |  |  |
| --- | --- | --- | --- |
| Показатель | Длительность рабочего дня (час.) | Количество дней (дн.) | Количество часов (час.) |
| Затраты времени | 6 | 22 | 132 |

Рассчитаем сметную стоимость проекта. Капитальные затраты для проектных работ не производились. Оборудование, использованное для проектирования сети приведено в таблице 2.

Затраты на разработку проекта сети подразделяем на прямые и косвенные. Расчет выполним по отдельным статьям. Прямые статьи затрат сведены в таблицы 3-5, косвенные – в таблицу 6.

Таблица 2 – Оборудование, использованное для создания проекта сети

|  |  |  |
| --- | --- | --- |
| Вид оборудования | Первоначальная стоимость оборудования, руб. | Мощность, Вт/ч |
| Системный блок Acer Veriton ES2730G | 31 000 | 220 |
| Монитор Монитор LG 22MK400A-B | 5 500 | 15 |
| Модем TP-LINK TL-WR841N | 1 300 | 5 |
| Принтер HP OfficeJet 6950 | 4 500 | 27 |
| Итого: | 42 300 руб. | 267 Вт/ч |

Таблица 3 – Прямые статьи затрат

|  |  |  |
| --- | --- | --- |
| Наименование статьи затрат | Сумма (руб.) | Удельный вес (%) |
| Основная заработная плата персонала | 42 900 | 54,4 |
| Дополнительная заработная плата персонала | 17 160 | 21,8 |
| Страховые взносы 30% | 18 018 | 22,8 |
| Электроэнергия | 110,3 | 0,1 |
| Амортизационные отчисления | 705 | 0,9 |
| Итого: | 78 893,3 | 100% |

1. Основная заработная плата персонала.

В среднем зарплата у техника-программиста 39 000 р. (120 ч)

1. Дополнительная заработная плата персонала.
2. Страховые взносы.
3. Электроэнергия.

Количество потребляемой мощности возьмем из таблицы 2. Количество потребленной электроэнергии:

Стоимость 1 кВт = 3,13 р.

Сумма расходов на электроэнергию

1. Амортизационные отчисления.

Таблица 4 – Расчет амортизационных отчислений

|  |  |  |  |
| --- | --- | --- | --- |
| № | Оборудование | Срок гарантии | Первоначальная стоимость |
| 1 | Системный блок Acer Veriton ES2730G | 5 лет | 31 000 |
| 2 | Монитор Монитор LG 22MK400A-B | 5 лет | 5 500 |
| 3 | Модем TP-LINK TL-WR841N | 5 лет | 1 300 |
| 4 | Принтер HP OfficeJet 6950 | 5 лет | 4 500 |
|  | Итог: |  | 42 300 руб. |

Структура прямых затрат проекта изображена на рисунке 2.

В структуре прямых затрат проекта наибольшую долю занимает основная заработная плата и дополнительная заработная плата техника программиста, что характерно для наукоемких технологий.

Рисунок 2 – Структура прямых затрат проекта

Рассчитаем косвенные расходы необходимые для создания проекта.

Таблица 5 – Косвенные статьи затрат

|  |  |  |
| --- | --- | --- |
| Наименование статьи | Сумма (руб.) | Удельный вес (%) |
| Охрана труда и техника безопасности | 15 000 | 42,9 |
| Содержание отдела | 15 000 | 42,9 |
| Прочие расходы | 5 000 | 14,2 |
| Итого: | 35 000 | 100% |

В структуре косвенных затрат проекта наибольшую долю занимает затраты на содержание отдела и технику безопасности.

Сметная стоимость проекта определяется суммированием прямых и косвенных затрат. Цена формируется с учетом налогообложения и суммы прибыли на создание проекта. В таблице 6 приведена смета затрат на разработку базы данных для учета вычислительной и оргтехники школы на примере МБОУ ООШ №19.

Таблица 6 – Смета затрат

|  |  |
| --- | --- |
| Наименование затрат | Сумма (руб.) |
| Прямые затраты | 78 893,3 |
| Косвенные затраты | 35 000 |
| Полная себестоимость | 113 893,3 |
| Прибыль | 34 167,99 |
| НДС | 29 612,3 |
| Цена | 177 673,6 |

1. Прибыль.
2. НДС.
3. Цена.

Сметная стоимость проекта составляет 177 673,6 руб., стоимость проекта является конкурентоспособной в Ростовской области.

В структуре полной производственной себестоимости проекта наибольшая доля приходится на прямые затраты 69,27% в общей совокупности затрат.

**Расчет экономического эффекта от внедрения**

Далее просчитаем эффективность от внедрения разработанной БД.

Временные затраты при работе с информацией в школе занесем в таблицу 7.

Таблица 7 – Затраты времени

|  |  |  |
| --- | --- | --- |
| Операция | Затраты времени до внедрения (мин) | Затраты времени после внедрения (мин) |
| 1. Запись данных о ВТ | 10 | 5 |
| 1. Просмотр информации о ВТ | 15 | 2 |
| 1. Выборка данных о ВТ | 50 | 3 |
| Итого: | 75 | 10 |

Экономия составит:

Скорость обработки информации увеличивается в 7,5 раз.

Диаграмма изменения времени на обработку информации после внедрения проекта КС изображена на рисунке 7.

Рисунок 7 – Диаграмма изменения времени на обработку информации

Сотрудник, обрабатывающий информацию, согласно штатного расписания, имеет должностной оклад равный 30 000 руб. в месяц. Среднее количество рабочих дней в месяц составляет 20 дней.

1. Стоимость одного часа работы.
2. Эффективность работы сотрудника.
3. Окупаемость проекта.

Итоговые показатели эффективности проекта сведены в таблицу 8.

Таблица 8 – Сводные показатели эффективности проекта

|  |  |  |
| --- | --- | --- |
| Наименование | Ед. изм. | Показатель |
| Стоимость проекта (без НДС) | руб. | 148 061,29 |
| Скорость обработки информации | раз | 7,5 |
| Эффективность работы сотрудника | руб./день | 11 250 |
| Окупаемость | день | 13,16 |

Анализируя вышеприведенные расчеты можно сделать вывод, что поставленная цель, заключаемая в необходимости выведения на новый уровень эффективности БД, была достигнута.

**Задание**

Рассчитайте экономическую эффективность заданной информационной системы

**Варианты**

Для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №18. Построение модели управления качеством

**Цель работы:** получение навыков построения модели управления качеством

**Теоретические сведения**

Качество информационной системы — это совокупность свойств системы, обусловливающих возможность ее использования для удовлетворения определенных в соответствии с ее назначением потребностей. Количественные характеристики этих свойств определяются показателями.

Основными показателями качества информационных систем являются надежность, достоверность, безопасность.

Надежность — свойство системы сохранять во времени в установленных пределах значения всех параметров, характеризующих способность выполнять требуемые функции в заданных режимах и условиях применения.

Надежность информационных систем не самоцель, а средство обеспечения своевременной и достоверной информации на ее выходе. Поэтому показатель достоверности функционирования имеет для информационных систем главенствующее значение, тем более что показатель своевременности информации в общем случае охватывается показателем достоверности.

Достоверность функционирования — свойство системы, обусловливающее безошибочность производимых ею преобразований информации. Достоверность функционирования информационной системы полностью определяется и измеряется достоверностью ее выходной информации.

Безопасность информационной системы — свойство, заключающееся в способности системы обеспечить конфиденциальность и целостность информации, то есть защиту информации от несанкционированного доступа с целью ее раскрытия, изменения или разрушения.

Эффективность — это свойство системы выполнять поставленную цель в заданных условиях использования и с определенным качеством. Показатели эффективности характеризуют степень приспособленности системы к выполнению поставленных перед нею задач и являются обобщающими показателями оптимальности функционирования ИС, зависящими от локальных показателей, каковыми являются надежность, достоверность, безопасность.

Кардинальным обобщающим показателем является экономическая эффективность системы, характеризующая целесообразность произведенных на создание и функционирование системы затрат.

Стандартизация – деятельность, которая заключается в установлении положений для общего и многоразового использования относительно разрешения существующих и возможных проблем, и направленная на достижение оптимального уровня упорядочения на таких условиях.

Стандартизация - это установление и применение правил с целью упорядочения деятельности в определенной области на пользу и при участии всех заинтересованных сторон, в частности, для достижения всеобщей оптимальной экономии при соблюдении условий эксплуатации (использования) и требований безопасности. Стандартизация охватывает все стадии жизненного цикла продукции.

Стандарт – это нормативный документ, созданный на основе консенсуса и одобренный признанным органом. С помощью стандарта устанавливаются правила, установочные указания или характеристики разных видов деятельности или их результатов, предназначенные для общего и многоразового пользования. Стандарт направлен на достижение оптимального уровня упорядоченности в определенной сфере и доступен широкому кругу потребителей.

Международная система качества: стандарты ИСО.

Сегодня в условиях расширения международной торговли, повышения качества продукции перед предприятиями весьма остро стоит проблема внедрения стандартов ИСО серии 9000 и перехода от комплексных систем управления качеством к международным стандартам.

Существуют следующие отличия систем качества (по ИСО 9000) от комплексной системы управления качеством продукции:

* ориентация на удовлетворение требований потребителя;
* возложение ответственности за качество продукции на конкретных исполнителей;
* проверка потребителем производства поставщика;
* выбор поставщика комплектующих изделий и материалов;
* сквозной контроль качества продукции, начиная от материалов и кончая утилизацией продукции:
* маркетинг;
* организация учета и анализа затрат на качество;
* прослеживаемость материалов и комплектующих изделий по всему циклу производства;
* решение вопросов утилизации продукции после эксплуатации.

Разработаны следующие 5 стандартов ИСО серии 9000:

1. ИСО 9000 «Общее руководство качеством и стандарты по обеспечению качества. Руководящие указания по выбору и применению».

2. ИСО 9001 «Системы качества. Модель для обеспечения качества при проектировании и/или разработке, монтаже и обслуживании».

3. ИСО 9002 «Системы качества. Модель для обеспечения качества при производстве и монтаже».

4. ИСО 9003 «Системы качества. Модель для обеспечения качества при окончательном контроле и испытаниях».

5. ИСО 9004 «Общее руководство качеством и элементы системы качества. Руководящие указания».

**Задание № 1**

Привести национальные стандарты обеспечения качества автоматизированных информационных систем.

**Задание № 2**

Охарактеризовать международную систему стандартизации и сертификации качества продукции.

**Задание № 3**

Описать стандарты группы ISO.

**Задание № 4**

Привести методы контроля качества в информационных системах.

**Задание № 5**

Постройте модель управления качеством процесса изучения модуля «Проектирование и разработка информационных систем».

# Практическая работа №19. Реинжиниринг методом интеграции

**Цель:** освоить на практике особенности работы в среде деловой графики и демо-версии систем поддержки управления предприятием, научиться приемам построения моделей бизнес- процессов в различных нотациях.

**Теоретические сведения**

**Реинжиниринг методом интеграции**

**Основные этапы реинжиниринга компании (организации)**

В целом работа по реинжинирингу бизнеса состоит из двух основных этапов: ―обратного‖ и ―прямого‖ инжиниринга компании. Для того, чтобы изменить систему управления, ее нужно сначала описать и оценить. Поэтому  реинжиниринг начинается с описания и оценки ситуации ―»как есть». Для того чтобы понять, как и зачем менять, надо разработать цели и стратегию изменений, модель системы управления ―»как нужно». После этого реинжиниринг системы управления требует разработки плана действий по переходу из ситуации ―»как есть» в ситуацию ―»как надо». Поэтому в качестве необходимых шагов реинжиниринга компании можно выделить:

1. Выбор стратегических приоритетов компании для формулирования целей бизнес- реинжиниринга и определения наиболее важных бизнес-процессов компании.

2. Создание модели существующей компании ―»как есть» на основе моделирования бизнес- процессов и функциональной структуры компании до начала проведения изменений.

3. Анализ модели существующей компании и выявление узких мест в компании c точки зрения функциональной структуры компании и ее бизнес-процессов.

4. Разработка новой функциональной структуры и бизнес-процессов компании на основе методов бизнес-реинжиниринга

5. Разработка и организация использования поддерживающих информационных систем. При этом определяются требуемые ресурсы (оборудование, программное обеспечение) и при необходимости реализуется специализированная информационная система.

6. Переход компании на новую функциональную структуру и бизнес-процессы, то есть внедрение новой системы управления в практику.

Как правило, перечисленные шаги выполняются не последовательно, а частично параллельно, причем некоторые шаги повторяются несколько раз.

**Выбор вида реинжиниринга**

Различают реинжиниринг в узком смысле (перепроектирование отдельных бизнес- процессов) и широком смысле (полная реструктуризация организационной структуры и бизнес- процессов). Какой вариант выбрать компании зависит от целей реинжиниринга, которые в свою очередь формируются в зависимости от стратегических приоритетов компании.

Многие компании, задумав использовать методы бизнес-инжиниринга, ограничиваются

―обратным бизнес-инжинирингом, то есть построением и анализом модели бизнес-процессов

―»как есть», так как он представляет самостоятельную ценность, поскольку дает возможность выявлять узкие места компании, оценивать последствия увеличения (уменьшения) ресурсов в некотором подразделении, сравнивать между собой последствия различных вариантов и т. п.

**Бенчмаркинг**

Реинжиниринговые команды могут использовать бенчмаркинг (benchmarking). В сущности бенчмаркинг состоит в поиске компаний, которые делают что-то лучше всех, и в изучении того, как они этого добиваются, чтобы использовать эту информацию для проведения реинжиниринга собственной компании.

Проблема использования бенчмаркинга состоит в том, что он может ограничить мысленную работу реинжиниринговой команды рамками того, что у же сделано в отрасли. Используемый таким образом бенчмаркинг является средством только догнать конкурента, а не резко вырваться вперед. Бенчмаркинг способен, однако, обогатить реинжиниринговые команды идеями, особенно в том случае, когда в качестве образцов рассматриваются компании других отраслей.

**Методика выполнения**

**Реинжиниринг методом интеграции**

**Задание №1**

Анализ деловой ситуации:

Охарактеризуйте позицию организации на рынке.

1. Проведите аудит бизнес-процессов.

2. Оцените уровень непротиворечивости бизнес-требований к модулям информационной системы.

3. Какие инновационные технологии сферы ИТ требуется внедрить в бизнес- процессы организации?

4. Каковы перспективные направления реинжиниринга отдельных бизнес-процессов на предприятии?

**Пример описания**

Описание ситуации (кейса):

Компания "Эльдорадо" – крупнейшая сеть магазинов электроники и бытовой техники в России и ближнем зарубежье. Сегодня под брендом "Эльдорадо" работают 700 магазинов, расположенные во всех российских городах. Миссия "Эльдорадо" - помочь сделать правильный выбор и создать собственный яркий и комфортный мир, наполненный качественной техникой лучших мировых брендов. Компания стремимся предоставлять покупателям максимально широкий ассортимент самой современной техники по доступным ценам. Высококвалифицированный персонал, который всегда готов дать грамотную консультацию по любому вопросу, постоянное проведение специальных акций и мероприятий, способствующих еще более выгодным покупкам, качественное обслуживание, а также наличие огромного ассортимента – вот что отличает и выделяет магазины «Эльдорадо» в их сегменте рынка.  Магазин

«Эльдорадо» занимается реализацией товара бытовой техники. Каталог открывает перед покупателями огромный выбор товаров для дома, включая мелкую и крупную бытовую технику, фото и видео аппаратуру, сотовые телефоны и мобильные устройства, электрические и бытовые инструменты, аксессуары к бытовой технике и миллионы других сопутствующих товаров. Рассмотрим один из магазинов, в нем работают 8 продавцов. Продавцы помогают покупателю сделать правильный выбор техники по обустройству своего дома. Покупатель выбирает товар, бренд, модель, расцветку и комплектацию, дополнительно может приобрести необходимые аксессуары, доставку и установку товара по необходимому адресу. Продавец оформляет заказ, делает выписку на товар, заполняет гарантийный талон, приглашает покупателя пройти на кассу для оплаты. На кассе работают 2 кассира, они принимают оплату наличных денежных средств, по окончанию смены передают деньги инкассатору банка. Если товар приобретен с основного склада, клиент по желанию может забрать оплаченный товар сам или оформить доставку, установку техники в удобное для покупателя время по указанному адресу. Если товар приобретен с отдаленного склада (Новосибирск), клиенту необходимо подождать срок исполнения заказа – обычно одна-две недели. После оплаты покупки за наличный расчет необходимо подойти к сотрудникам информации, и оформить доставку (уточнить дату доставки, установки и адресные данные). На складе работают 2 кладовщика, каждый из которых «ведет» несколько заказов, и отслеживает движение товара. Помогают принимать машины с товаром и отгружать товар 2 грузчика. Директор занимается обучением персонала, еженедельно снимает остатки товара и заказывает с отдаленных складов товар, который заканчивается. Когда заказ готов, специалист по установке техники связывается с покупателем и договаривается о точном времени доставки, доставляет товар клиенту, устанавливает в необходимом месте и подписывает документы о выполнении работ у клиента. После доставки заказа специалист сдает документы бухгалтеру, который контролирует правильность расчетов и оформления.

**Задание №2**

1. Описать технико-экономические характеристики предприятия.

2. Разработка стратегии предприятия. Привести бизнесс-направления деятельности предприятия. Определить необходимость применения реинжиниринга.

3. Разработка предложений по усовершенствованию системы управления предприятием. Описание всех бизнес-процессов отдела или предприятия. Представить технологическую карту бизнес-процессов.

Технологическая карта бизнесс-процессов

|  |  |  |
| --- | --- | --- |
| Процесс | Информация | Механизм |
| Управление предприятием | Информация о внутренней и внешней среде предприятия | Директор |
| Управление снабжением | Информация о рынке поставщиков сырья и материалов. План деятельности предприятия | Служба снабжения |
| Управление финансами | Финансовая информация о предприятии. План деятельности предприятия. План закупок | Директор  Бухгалтерия |
| Управление производством | Заявки от покупателей, заказы. План закупленных сырья и материалов, годных к использованию. План деятельности предприятия | Директор  Служба производства  Отдел разработок |
| Управление сбытом | Запросы потребителей. Счет-фактура. План деятельности предприятия | Коммерческий отдел  Отдел сбыта и снабжения |

4. Построение функциональной модели, модели потоков данных существующих бизнес- процессов.

5. Анализ «узких мест».

6. Отбор бизнес-процессов для реинжиниринга.

7. Построение функциональной модели, модели потоков данных бизнес-процессов с предложениями по реинжинирингу.

8. Анализ экономической эффективности.

**Варианты**

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №20. Реинжиниринг бизнес-процессов методом горизонтального и/или вертикального сжатия

**Цель:** научиться использовать метод горизонтального и/или вертикального сжатия для реинжиниринга бизнес-процессов; осуществлять постановку задач по обработке информации; проводить анализ предметной области; осуществлять выбор модели и средства построения информационной системы и программных средств.

**Теоретические сведения**

**Реинжиниринг бизнес-процессов методом горизонтального и/или вертикального сжатия**

Вертикальное «сжатие» бизнес-процесса — сокращение уровней функциональной иерархии, задействованных в выполнении процедур процесса и принятии решений.

**Методика выполнения**

**Реинжиниринг бизнес-процессов методом горизонтального и/или вертикального сжатия**

**Задание №1**

Выполнить «сжатие» бизнес-процесса на основе представленных примеров.

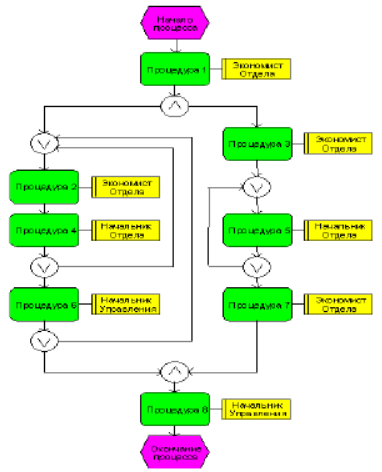
**Ход выполения**

Пример вертикального «сжатия» процесса приводится на следующих рисунках.

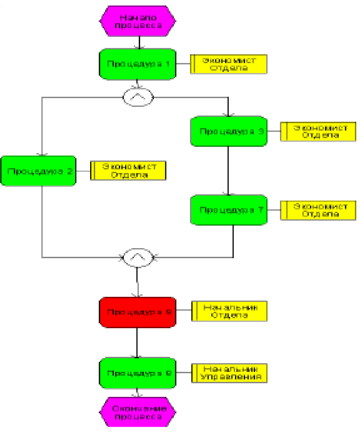
Представленная модель бизнес-процесса включает несколько итерационных согласований, в которых принимают участие Экономист, Начальник Отдела и Начальник Управления. Таким образом, для выполнения процесса необходимо участие сотрудников трех уровней функциональной иерархии. На практике, число уровней иерархии может достигать 8-10 (для крупных промышленных предприятий).

При таком числе уровней длительность выполнения бизнес-процесса значительно возрастает, а эффективность снижается.

На рисунке представлена модель, в которой устранены итерационные согласования, например за счет делегирования полномочий по принятию решений Экономисту Отдела. Начальник Отдела лишь просматривает и визирует документацию в конце процесса (процедура 9), после чего информации передается Начальнику Управления. Горизонтальное «сжатие» бизнес-процесса — устранение неэффективных процедур процесса, итерационных согласований на одном уровне иерархии, сокращение времени выполнения процедур процесса, времени простоя и т.д.



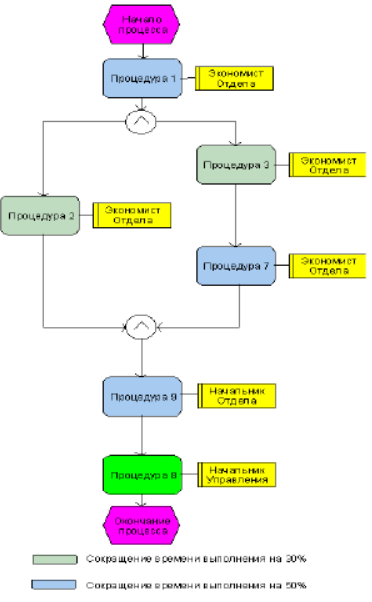
Модель процесса «как есть»



Модель процесса «как должно быть»

Сокращение времени выполнения функций. «Сжатия» бизнес-процесса можно добиться, используя как конкретные технические приемы реорганизации, так и радикально переосмысливая процесс в целом.

Последний подход рассмотрен, например, в работах «классиков» реинжиниринга — Хаммера и Чампи. Первый подразумевает рутинную, кропотливую работу по детальному анализу бизнес-процессов и разработке мер по их реорганизации. Выбор методов зависит от целей проекта и возможностей привлечения соответствующих ресурсов (например, возможности провести имитационное моделирование бизнес-процессов).



Модель процесса «как должно быть»

В отчете к практической работе необходимо представить следующее:

− схема «как есть»

− схема «как должно быть»

− описание к схемам.

**Варианты**

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №21. Разработка требований безопасности информационной системы

**Цель:** получение навыков разработки требований безопасности информационной системы.

**Теоретические сведения**

Безопасность ИС - защищенность системы от случайного или преднамеренного вмешательства в нормальный процесс ее функционирования, от попыток хищения (несанкционированного получения) информации, модификации или физического разрушения ее компонентов. Иначе говоря, это способность противодействовать различным возмущающим воздействиям на ИС.

Угроза безопасности информации- события или действия, которые могут привести к искажению, несанкционированному использованию или даже к разрушению информационных ресурсов управляемой системы, а также программных и аппаратных средств.

**Защита информации в информационных системах и информационных технологиях**

Сегодня можно утверждать, что рождается новая современная технология - технология защиты информации в компьютерных информационных системах и в сетях передачи данных. Реализация этой технологии требует увеличивающихся расходов и усилий. Однако все это позволяет избежать значительно превосходящих потерь и ущерба, которые могут возникнуть при реальном осуществлении угроз ИС и ИТ.

**Виды умышленных угроз безопасности информации**

*Пассивные*угрозы направлены в основном на несанкционированное использование информационных ресурсов ИС, не оказывая при этом влияния на ее функционирование. Например, несанкционированный доступ к базам данных, прослушивание каналов связи и т.д.

*Активные угрозы*имеют целью нарушение нормального функционирования **ИС**путем целенаправленного воздействия на ее компоненты. К активным угрозам относятся: вывод из строя компьютера или его операционной системы, искажение сведений в БД, разрушение ПО компьютеров, нарушение работы линий связи и т.д. Источником активных угроз могут быть действия взломщиков, вредоносные программы и т.п.

Умышленные угрозы подразделяются также на *внутренние*(возникающие внутри управляемой организации) и *внешние.*

Внутренние угрозы чаще всего определяются социальной напряженностью и тяжелым моральным климатом.

Внешние угрозы могут определяться злонамеренными действиями конкурентов, экономическими условиями и другими причинами (например, стихийными бедствиями). Широкое распространение получил *промышленный шпионаж*это наносящие ущерб владельцу коммерческой тайны незаконные сбор, присвоение и передача сведений, составляющих коммерческую тайну, лицом, не уполномоченным на это ее владельцем. *.*К основным угрозам безопасности информации и нормального функционирования ИС относятся:

\*утечка конфиденциальной информации;

\*компрометация информации;

\*несанкционированное использование информационных ресурсов;

\*ошибочное использование информационных ресурсов;

\*несанкционированный обмен информацией между абонентами;

\*отказ от информации;

\*нарушение информационного обслуживания;

\*незаконное использование привилегий.

*Утечка конфиденциальной информации --*это бесконтрольный выход конфиденциальной информации за пределы ИС или круга лиц, которым она была доверена по службе или стала известна в процессе работы. Эта утечка может быть следствием:

\*разглашения конфиденциальной информации;

\*ухода информации по различным, главным образом техническим, каналам;

\*несанкционированного доступа *к*конфиденциальной информации различными способами.

*Разглашение информации ее*владельцем или обладателем есть умышленные или неосторожные действия должностных лиц и пользователей, которым соответствующие сведения в установленном порядке были доверены по службе или по работе, приведшие кознакомлению с ним лиц, не допущенных кэтим сведениям.

Возможен *бесконтрольный уход конфиденциальной информации*по визуально-оптическим, акустическим, электромагнитным и другим каналам.

*Несанкционированный доступ --*это противоправное преднамеренное овладение конфиденциальной информацией лицом, не имеющим права доступа кохраняемым сведениям.

Наиболее распространенными путями несанкционированного доступа кинформации являются:

\*перехват электронных излучений;

\*принудительное электромагнитное облучение (подсветка) линий связи с целью получения паразитной модуляции несущей;

\*применение подслушивающих устройств (закладок);

\*дистанционное фотографирование;

\*перехват акустических излучений и восстановление текста принтера;

\*чтение остаточной информации в памяти системы после выполнения санкционированных запросов;

\*копирование носителей информации с преодолением мер защиты

\*маскировка под зарегистрированного пользователя;

\*маскировка под запросы системы;

\*использование программных ловушек;

\*использование недостатков языков программирования и операционных систем;

\*незаконное подключение к аппаратуре и линиям связи специально разработанных аппаратных средств, обеспечивающих доступ информации;

\*злоумышленный вывод из строя механизмов защиты;

\*расшифровка специальными программами зашифрованной: информации;

\*информационные инфекции.

Любые способы утечки конфиденциальной информации могут привести к значительному материальному и моральному ущербу как для организации, где функционирует ИС, так и для ее пользователей.

Большинство из перечисленных технических путей несанкционированного доступа поддаются надежной блокировке при правильно разработанной и реализуемой на практике системе обеспечения безопасности. Но борьба с информационными инфекциями представляет значительные трудности, так как существует и постоянно разрабатывается огромное множество вредоносных программ, цель которых -- порча информации в БД и ПО компьютеров. Большое число разновидностей этих программ не позволяет разработать постоянных и надежных средств защиты против них.

**Вредоносные программы**

Они классифицируются следующим образом:

*Логические бомбы-*используются для искажения или уничтожения информации, реже с их помощью совершаются кража или мошенничество. Манипуляциями с логическими бомбами обычно занимаются чем-то недовольные служащие, собирающиеся покинуть данную организацию, но это могут быть и консультанты, служащие с определенными политическими убеждениями и т.п.

*Троянский конь --*программа, выполняющая в дополнение к основным, т. е. запроектированным и документированным действиям, действия дополнительные, не описанные в документации. Троянский конь представляет собой дополнительный блок команд, тем или иным образом вставленный в исходную безвредную программу, которая затем передается (дарится, продается, подменяется) пользователям ИС.Этот блок команд может срабатывать при наступлении некоторого условия (даты, времени, по команде извне и т.д.). Запустивший такую программу подвергает опасности как свои файлы, так и всю ИС в целом. Троянский конь действует обычно в рамках полномочий одного пользователя, но в интересах другого пользователя или вообще постороннего человека, личность которого установить порой невозможно.

*Вирус --*программа, которая может заражать другие программы путем включения в них модифицированной копии, обладающей способностью к дальнейшему размножению.

Вирус характеризуется двумя основными особенностями:

1) способностью к саморазмножению;

2) способностью к вмешательству в вычислительный процесс (т. е. к получению возможности управления).

*Червь --*программа, распространяющаяся через сеть и не оставляющая своей копии на магнитном носителе. Червь использует механизмы поддержки сети для определения узла, который может быть заражен. Затем с помощью тех же механизмов передает свое тело или его часть на этот узел и либо активизируется, либо ждет для этого подходящих условий. Наиболее известный представитель этого класса -- вирус Морриса (червь Морриса), поразивший сеть Internet в 1988 г. Подходящей средой распространения червя является сеть, все пользователи которой считаются дружественными и доверяют друг другу, а защитные механизмы отсутствуют. Наилучший способ защиты от червя -- принятие мер предосторожности против несанкционированного доступа к сети.

*Захватчик паролей --*это программы, специально предназначенные для воровства паролей. При попытке обращения пользователя к терминалу системы на экран выводится информация, необходимая для окончания сеанса работы. Пытаясь организовать вход, пользователь вводит имя и пароль, которые пересылаются владельцу программы-захватчика, после чего выводится сообщение об ошибке, а ввод и управление возвращаются к операционной системе. Пользователь, думающий, что допустил ошибку при наборе пароля, повторяет вход и получает доступ *к*системе. Однако его имя и пароль уже известны владельцу программы-захватчика. Перехват пароля возможен и другими способами. Для предотвращения этой угрозы перед входом в систему необходимо убедиться, что вы вводите имя и пароль именно системной программе ввода, а не какой-нибудь другой. Кроме того, необходимо неукоснительно придерживаться правил использования паролей и работы с системой. Большинство нарушений происходит не из-за хитроумных атак, а из-за элементарной небрежности. Соблюдение специально разработанных правил использования паролей -- необходимое условие надежной защиты. угроза безопасность вредоносный вирус

*Компрометация информации*(один из видов информационных инфекций). Реализуется, как правило, посредством несанкционированных изменений в базе данных в результате чего ее потребитель вынужден либо отказаться от нее, либо предпринимать дополнительные усилия для выявления изменений и восстановления истинных сведений. При использовании скомпрометированной информации потребитель подвергается опасности принятия неверных решений.

*Несанкционированное использование информационных ресурсов, с*одной стороны, является последствиями ее утечки и средством ее компрометации. С другой стороны, оно имеет самостоятельное значение, так как может нанести большой ущерб управляемой системе (вплоть до полного выхода ИТ из строя) или ее абонентам.

*Ошибочное использование информационных ресурсов*будучи санкционированным тем не менее может привести к разрушению, утечке или компрометации указанных ресурсов. Данная угроза чаще всего является следствием ошибок, имеющихся в ПО ИТ.

*Несанкционированный обмен информацией между абонентами*может привести кполучению одним из них сведений, доступ ккоторым ему запрещен. Последствия -- те же, что и при несанкционированном доступе.

*Отказ от информации*состоит в непризнании получателем или отправителем этой информации фактов ее получения или отправки. Это позволяет одной из сторон расторгать заключенные финансовые соглашения техническим путем, формально не отказываясь от них, нанося тем самым второй стороне значительный ущерб.

*Нарушение информационного обслуживания --*угроза, источником которой является сама ИТ. Задержка с предоставлением информационных ресурсов абоненту может привести к тяжелым для него последствиям. Отсутствие у пользователя своевременных данных, необходимых для принятия решения, может вызвать его нерациональные действия.

*Незаконное использование привилегий.*Любая защищенная система содержит средства, используемые в чрезвычайных ситуациях, или средства которые способны функционировать с нарушением существующей политики безопасности.

Взлом системы- понимают умышленное проникновение в систему, когда взломщик не имеет санкционированных параметров для входа. Способы взлома могут быть различными, и при некоторых из них происходит совпадение с ранее описанными угрозами.

**Обеспечение информационной безопасности**

Обеспечение информационной безопасности (ИБ) – комплексная задача, решаемая параллельно по целому ряду направлений: правовому, организационному и техническому. Эффективное обеспечение ИБ возможно только при создании системы обеспечения информационной безопасности (СОИБ), охватывающей все направления деятельности организации и функционирующей на всех уровнях управления (стратегическом, тактическом и оперативном).

Создание СОИБ ведется с учетом специфики деятельности Заказчика, текущего уровня развития информационных технологий, учета планов развития в кратко и среднесрочной перспективе.

СОИБ представляет собой сложную организационно-техническую структуру, состоящую из:

* комплексной системы защиты информации – совокупности интегрированных программно-технических средств защиты информации, обеспечивающих защиту информационных ресурсов во всех информационных системах Заказчика и на всех этапах их жизненного цикла;
* совокупности процессов, обеспечивающего эффективную эксплуатацию и развитие комплексной системы защиты информации;
* персонала – сотрудников подразделений ИБ, занятых в процессах эксплуатации комплексной системы защиты информации, а также контроля над выполнением требований внутренних документов в области ИБ и прочих задачах обеспечения ИБ.

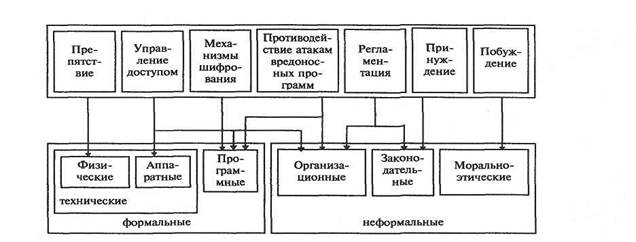
При создании СОИБ мы ориентируемся на ряд ключевых принципов, которым должна отвечать система обеспечения информационной безопасности:

* соответствие мер защиты, реализуемых СОИБ, реальным угрозам ИБ (которые определяются предварительно, по результатам аудита ИБ);
* поэтапное создание СОИБ с целью оптимизации финансовых затрат, сохраняющее при этом единую концепцию СОИБ;
* защита инвестиций – использование существующих средств и систем защиты информации для построения СОИБ, с целью снижения капитальных затрат (в том числе интеграция разнородных подсистем обеспечения ИБ путем создания единой интегрированной системы);
* централизованное управление и мониторинг, что снижает трудозатраты на эксплуатацию СОИБ;
* учет и эффективная интеграция с существующими процессами управления ИТ (при необходимости, с параллельной модернизацией средств и систем управления ИТ).

УЦСБ обладает большим опытом построения СОИБ для крупных предприятий топливно-энергетического комплекса, банковского сектора, промышленных предприятий и органов государственной власти, а также выполнения смежных задач:

* разработка концепции обеспечения ИБ;
* проведение аудитов ИБ;
* создание систем защиты отдельных элементов информационной инфраструктуры или комплексной системы защиты информации в качестве инфраструктуры будущей СОИБ;
* проведение тестирований на проникновение (так называемые пентесты или pentest).

**Методы и средства обеспечения безопасности информации**



*Препятствие* — метод физического преграждения пути злоумышленнику к защищаемой информации (к аппаратуре, носителям информации и т.д.).

*Управление доступом*— методы защиты информации регулированием использования всех ресурсов ИС и ИТ. Эти методы должны противостоять всем возможным путям несанкционированного доступа к информации. Управление доступом включает следующие функции зашиты:

• идентификацию пользователей, персонала и ресурсов системы (присвоение каждому объекту персонального идентификатора);

• опознание (установление подлинности) объекта или субъекта по предъявленному им идентификатору;

• проверку полномочий (проверка соответствия дня недели, времени суток, запрашиваемых ресурсов и процедур установленному регламенту);

• разрешение и создание условий работы в пределах установленного регламента;

• регистрацию (протоколирование) обращений к защищаемым ресурсам;

• реагирование (сигнализация, отключение, задержка работ, отказ в запросе и т.п.) при попытках несанкционированных действий.

*Механизмы шифрования* — криптографическое закрытие информации.

Эти методы защиты все шире применяются как при обработке, так и при хранении информации на магнитных носителях. При передаче информации по каналам связи большой протяженности этот метод является единственно надежным.

*Противодействие атакам вредоносных программ* предполагает комплекс разнообразных мер организационного характера и использование антивирусных программ. Цели принимаемых мер — это уменьшение вероятности инфицирования АИС, выявление фактов заражения системы; уменьшение последствий информационных инфекций, локализация или уничтожение вирусов; восстановление информации в ИС.

*Регламентация* — создание таких условий автоматизированной обработки, хранения и передачи защищаемой информации, при которых нормы и стандарты по защите выполняются в наибольшей степени.

*Принуждение* — метод защиты, при котором пользователи и персонал ИС вынуждены соблюдать правила обработки, передачи и использования защищаемой информации под угрозой материальной, административной или уголовной ответственности.

*Побуждение* — метод защиты, побуждающий пользователей и персонал ИС не нарушать установленные порядки за счет соблюдения сложившихся моральных и этических норм.

Вся совокупность технических средств подразделяется на **аппаратные** и **физические**.

*Аппаратные средства* — устройства, встраиваемые непосредственно в вычислительную технику, или устройства, которые сопрягаются с ней по стандартному интерфейсу.

*Физические средства* включают различные инженерные устройства и сооружения, препятствующие физическому проникновению злоумышленников на объекты защиты и осуществляющие защиту персонала (личные средства безопасности), материальных средств и финансов, информации от противоправных действий.

Примеры физических средств: замки на дверях, решетки на окнах, средства электронной охранной сигнализации и т.п.

*Программные средства* — это специальные программы и программные комплексы, предназначенные для защиты информации в ИС.

Многие из них слиты с программным обеспечением самой ИС.

Из средств программного обеспечения системы защиты выделим еще программные средства, реализующие механизмы шифрования (криптографии).

Криптография — это наука об обеспечении секретности и/или аутентичности (подлинности) передаваемых сообщений.

*Организационные средства* осуществляют регламентацию производственной деятельности в ИС и взаимоотношений исполнителей на нормативно-правовой основе таким образом, что разглашение, утечка и несанкционированный доступ к конфиденциальной информации становятся невозможными или существенно затрудняются за счет проведения организационных мероприятий. Комплекс этих мер реализуется группой информационной безопасности, но должен находиться под контролем первого руководителя.

*Законодательные средства* защиты определяются законодательными актами страны, которыми регламентируются правила пользования, обработки и передачи информации ограниченного доступа и устанавливаются меры ответственности за нарушение этих правил.

*Морально-этические средства* защиты включают всевозможные нормы поведения, которые традиционно сложились ранее, складываются по мере распространения ИС и ИТ в стране и в мире или специально разрабатываются. Морально-этические нормы могут быть неписаные (например, честность) либо оформленные в некий свод (устав) правил или предписаний. Эти нормы, как правило, не являются законодательно утвержденными, но поскольку их несоблюдение приводит к падению престижа организации, они считаются обязательными для исполнения. Характерным примером таких предписаний является Кодекс профессионального поведения членов Ассоциации пользователей ЭВМ США.

**Задание № 1.**

Определите цели и задачи системы защиты информации.

**Задание № 2**

Перечислите факторы, влияющие на организацию системы защиты информации.

**Задание № 3**

Определите дестабилизирующие воздействия на информационную систему и способы их нейтрализации.

**Задание № 4**

Напишите программу по подсчету общей вероятности нарушения безопасности объекта, подсчитываемой по формуле

https://documents.infourok.ru/3ad1f2dc-146c-4c4e-a065-1c649a8f157c/0/image001.jpg

где к – число угроз; n – число нарушителей; Рi – вероятность появления субъекта i-го типа; p(j/i) – условная вероятность того, что субъект i-го типа выберет для реализации угрозу j-го типа; qн1– вероятность несрабатывания средств обнаружения; qн2 – вероятность несрабатывания средств отражения; а – постоянная величина, характеризующая "скорость" реализации угрозы, tот – время, которым располагает субъект угрозы, если tот = 0 – угроза не реализуется.

**Задание № 5**

Разработайте требования безопасности информационной системы в соответствии с вариантом

**Задание № 6**

Выберите методы и средства защиты информации для исследуемой информационной системы.

**Варианты**

Для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

**6 семестр**

# Практическая работа №22. Проектирование спецификации информационной системы

**Цель**: получение навыков проектирования спецификации информационной системы

Спецификация требований программного обеспечения (англ. Software Requirements Specification, SRS) — структурированный набор требований (функционал, производительность, конструктивные ограничения и атрибуты) к программному обеспечению и его внешним интерфейсам. (Определение на основе IEEE Std 1012:2004) Предназначен для того, чтобы установить базу для соглашения между заказчиком и разработчиком (или подрядчиками) о том, как должен функционировать программный продукт.

Может включать ряд пользовательских сценариев (англ. use cases), которые описывают варианты взаимодействия между пользователями и программным обеспечением.

Пользовательские сценарии являются средством представления функциональных требований. В дополнение к пользовательским сценариям, спецификация также содержит нефункциональные требования, которые налагают ограничения на дизайн или реализацию (такие как требования производительности, стандарты качества, или проектные ограничения).

В стандарте ISO/IEC/IEEE 29148:2011, который пришел на смену устаревшему IEEE 830, содержится рекомендации к структуре и методам описания программных требований — «Recommended Practice for Software Requirements Specifications».

Пример организации структуры SRS, на основе стандарта ISO/IEC/IEEE 29148:2011

1. Введение
   1. Цели
   2. Соглашения о терминах
   3. Предполагаемая аудитория и последовательность восприятия
   4. Масштаб проекта
   5. Ссылки на источники
2. Общее описание
   1. Видение продукта
   2. Функциональность продукта
   3. Классы и характеристики пользователей
   4. Среда функционирования продукта (операционная среда)
   5. Рамки, ограничения, правила и стандарты
   6. Документация для пользователей
   7. Допущения и зависимости
3. Функциональность системы
   1. Функциональный блок X (таких блоков может быть несколько)
      1. Описание и приоритет
      2. Причинно-следственные связи, алгоритмы (движение процессов, workflows)
      3. Функциональные требования
4. Требования к внешним интерфейсам
   1. Интерфейсы пользователя (UI)
   2. Программные интерфейсы (SI)
   3. Интерфейсы связи и коммуникации (CI)
5. Нефункциональные требования
   1. Требования к производительности
   2. Требования к сохранности (данных)
   3. Требования к безопасности системы
6. Приложения

**Задание**

Составить спецификацию требований к ИС в соответствии с вариантом

**Варианты**

Для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №23. Составление технического задания

**Цель работы:** знакомство со стандартами, регламентирующими жизненный цикл разработки ИС. Приобретение практических навыков при разработке программного документа Техническое задание.

**Теоретические сведения**

Для обеспечения потребностей по разработке программного обеспечения (ПО) ИС необходимо взаимосвязанное совершенствование технических решений, технологий проектирования и программирования, инструментальных средств, а также обучения специалистов.

Стандарты для процессов, инструментальных средств и данных, которые отражают лучшую практику и защищают от неблагоприятных последствий, играют определенную роль в обеспечении указанных потребностей, в частности, поддерживая доверие потребителя к продуктам, услугам и технологиям разработки ПО.

Использование стандартов при разработке ПО ИС позволит обеспечить:

* повышение надежности;
* повышение эффективности применения и снижение затрат в сфере сопровождения программных средств (ПС);
* увеличение коэффициента повторного использования ПС общего назначения;
* повышение объективности оценок качества ПС;
* создание условий для конкуренции разработчиков на внутреннем рынке ПС;
* обеспечение конкурентоспособности отечественных ПС на мировом рынке.

Стандарты комплекса ГОСТ 34 на создание и развитие автоматизированных систем (АС) – обобщенные, но воспринимаемые как весьма жесткие по структуре жизненного цикла (ЖЦ) и проектной документации. Наиболее популярными можно считать ГОСТ 34.601-90 **(**Автоматизированные системы. Стадии создания) и ГОСТ 34.602-89 (Техническое задание на создание АС). Введение единой, достаточно качественно определенной терминологии, наличие достаточно разумной классификации работ, документов, видов обеспечения и др. способствует более полной и качественной стыковке разных систем, что особенно важно в условиях, когда разрабатывается все больше сложных комплексных АС.

В последние годы в стране идет интенсивная работа по гармонизации государственных стандартов Российской Федерации с международными стандартами ISO в области создания нормативной базы управления жизненным циклом программных средств и информационных систем. В основе стандартизации - ГОСТ Р ИСО/МЭК 12207-99 "Информационная технология. Процессы жизненного цикла программных средств", который является аутентичным переводом международного стандарта ISO/IEC 12207: 1995-08-01.

Техническое задание (ТЗ) на АС - утвержденный в установленном порядке документ, определяющий цели, требования и основные исходные данные необходимые для разработки АС и содержащий предварительную оценку экономической эффективности.

ТЗ содержит основные технические требования, предъявляемые к изделию и исходные данные для разработки; в ТЗ указываются назначение объекта, область его применения, стадии разработки документации, её состав, сроки исполнения и т. д., а также особые требования, обусловленные спецификой самого объекта либо условиями его эксплуатации. Как правило, ТЗ составляют на основе анализа результатов, полученных в ходе предпроектного обследования.

Как инструмент коммуникации в связке общения заказчик-исполнитель, техническое задание позволяет:

А) Обеим сторонам:

* представить готовый продукт;
* выполнить проверку готового продукта по пунктам (приёмочное тестирование – проведение испытаний);
* уменьшить число ошибок, связанных с изменением требований в результате их неполноты или ошибочности (на всех стадиях и этапах создания, за исключением испытаний).

Б) Заказчику:

* осознать, что именно ему нужно, опираясь на существующие на данный момент технические возможности и свои ресурсы;
* требовать от исполнителя соответствия продукта всем условиям, оговорённым в ТЗ.

В) Исполнителю:

* правильно понять суть задачи, показать заказчику «технический облик» будущего программного изделия или АС;
* спланировать выполнение проекта и работать по намеченному плану;
* отказаться от выполнения работ, не указанных в ТЗ.

**Задание к выполнению**

В соответствии с вариантом задания, определяющим предметную область на основании ГОСТ 34.602-89, разработать документ Техническое задание.

**Варианты**

Для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №24. Составление технического проекта

**Цель работы:** знакомство со стандартами, регламентирующими жизненный цикл разработки ИС. Приобретение практических навыков при разработке программного документа Технический проект.

**Теоретические сведения**

Технический проект – стадия разработки конструкторской документации на изделие или стадия создания автоматизированной системы.

В более узком смысле под техническим проектом понимается совокупность технических документов, которые содержат окончательные проектные решения по изделию (системе).

Разработку технического проекта на материальные изделия осуществляют в соответствии с Единой системой конструкторской документации (ЕСКД), на автоматизированные системы – в соответствии с Комплексом стандартов на автоматизированные системы (ГОСТ 34 серии).

Работы по созданию (развитию) автоматизированной системы, выполняемые на стадии «Технический проект», регламентируются документом ГОСТ 34.601-90 и в общем случае содержат следующие этапы:

Разработка проектных решений по системе и её частям.

Разработка проектной документации на автоматизированную систему и её части.

Разработка и оформление документации на поставку изделий для комплектования автоматизированной системы и (или) технических требований (технических заданий) на их разработку.

Разработка заданий на проектирование в смежных частях проекта объекта автоматизации.

Перечень документов, создаваемых на стадии «Технический проект», определяется документом ГОСТ 34.201-89.

**Задание к выполнению**

В соответствии с вариантом задания, определяющим предметную область на основании ГОСТ 34.601-90, разработать документ Технический проект на создание АИС.

**Варианты**

Для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №25. Составление руководства системного программиста

**Цель работы:** изучение нормативно правовой документации, регламентирующей разработку документации на программные средства, приобретение навыков разработки руководства системного программиста ПС.

**Теоретические сведения**

Если программа более-менее проста, пользователь может установить ее себе на компьютер самостоятельно. Поддерживать ее работоспособность, например, выполнить переустановку, если возникнут какие-нибудь проблемы, он тоже, как правило, в состоянии.

Сложными, в том числе, серверными и сетевыми программными продуктами приходится заниматься квалифицированным специалистам, системным администраторам. Более того, во многих компаниях сотрудникам запрещено устанавливать на своих рабочих местах программы по своему усмотрению. Даже простые программы там ставит только системный администратор.

В обязанности системного администратора также входит поддержание работоспособности программ, используемых в рамках тех или иных систем. Эта деятельность может заключаться в периодической проверке логов, резервном копировании данных, замерах производительности, устранении различных технических проблем.

Руководство системного администратора — программный документ, предоставляющий специалисту информацию, необходимую для выполнения этой работы.

В ЕСПД специалист, сходный по обязанностям с современным системным администратором, называется системным программистом, а адресованный ему документ — руководством системного программиста.

**Типовая структура руководства системного программиста**. Структура руководства системного программиста, приведена в ГОСТ 19.503-79:

1. Общие сведения о программе
   1. Назначение программы
   2. Функции программы
   3. Минимальный состав технических средств
   4. Минимальный состав программных средств
   5. Требования к персоналу (системному программисту)
2. Структура программы
   1. Сведения о структуре программы
   2. Сведения о составных частях программы
   3. Сведения о связях между составными частями программы
   4. Сведения о связях с другими программами
3. Настройка программы
   1. Настройка на состав технических средств
   2. Настройка на состав программных средств
4. Проверка программы
   1. Описание способов проверки
   2. Методы прогона
   3. Контрольные примеры
   4. Результаты
5. Дополнительные возможности
   1. Описание дополнительных разделов функциональных возможностей программы
   2. Способы выбора дополнительных возможностей
6. Сообщения системному программисту

**Задание к выполнению**

В соответствии с вариантом задания, определяющим предметную область на основании ГОСТ 19.503-79, разработать документ Руководство системного программиста.

**Варианты**

Для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №26. Разработка общего функционального описания программного средства по индивидуальному заданию

**Цель**: получение навыков разработки общего функционального описания программного средства

**Теоретические вопросы**

Виды информационных систем, их назначение и состав. Технологии разработки информационных систем.

Методологии разработки программного обеспечения. Процесс разработки программного обеспечения.

Управление разработкой программного обеспечения. Проектирование информационных систем.

Этапы проектирования.

Задачи и результаты проектирования.

**Задание № 1**

Подготовить исходные данные для разработки информационной системы. Исходными данными для планирования являются:

* общее описание некоторой информационной системы (назначение, область применения, решаемые задачи, технологические особенности реализации и внедрения);
* ограничения и условия разработки (требования заказчика, возможности команды разработчиков, сроки разработки, бюджет проекта и т.д.).

**Задание № 2**

Составить эскизный план разработки информационной системы.

**Задание № 3**

Составить документ «Технический проект» с описанием проектных решений (архитектура системы, логическая структура базы данных, решения по реализации пользовательского интерфейса и т.д.).

**Задание № 4**

Составить документ «План тестирования» с описанием методики тестирования и контрольных тестов.

**Задание № 5**

Составить документ «План ввода информационной системы в эксплуатацию».

**Варианты**

Для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №27. Разработка руководства по инсталляции программного средства по индивидуальному заданию

**Цель**: получение навыков разработки руководства по инсталляции программного средства

**Теоретические вопросы**

Понятие дистрибутива. Виды дистрибутивов.

Типы инсталляции программного обеспечения.

Руководство по инсталляции программного средства.

**Задание № 1**

Разработать руководство по инсталляции программного средства для заданной информационной системы.

**Варианты**

Для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №28. Разработка руководства пользователя программного средства

**Цель**: получение навыков разработки руководства пользователя программного средства

Руководство пользователя — документ, назначение которого — предоставить людям помощь в использовании некоторой системы. Документ входит в состав технической документации на систему и, как правило, подготавливается техническим писателем.

Большинство руководств пользователя помимо текстовых описаний содержат изображения. В случае программного обеспечения, в руководство обычно включаются снимки экрана, при описании аппаратуры — простые и понятные рисунки. Используется стиль и язык, доступный предполагаемой аудитории, использование жаргона сокращается до минимума либо подробно объясняется.

Структура и содержание документа Руководство пользователя автоматизированной системы регламентированы подразделом 3.4 документа РД 50-34.698-90. Структура и содержание документов Руководство оператора, Руководство программиста, Руководство системного программиста регламентированы ГОСТ 19.505-79, ГОСТ 19.504-79 и ГОСТ 19.503-79 соответственно.

Комплекс стандартов и руководящих документов на автоматизированные системы (ГОСТ 34)

* РД 50-34.698-90 АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ. ТРЕБОВАНИЯ К СОДЕРЖАНИЮ ДОКУМЕНТОВ
* Единая система конструкторской документации (ЕСКД) определяет документ «Руководство по эксплуатации» и другие документы:
* Единая система программной документации (ЕСПД) определяет документы «Руководство оператора», «Руководство по техническому обслуживанию» и их структуру:
* ГОСТ 19.101-77 Виды программ и программных документов
* ГОСТ 19.105-78 Общие требования к программным документам
* ГОСТ 19.505-79 Руководство оператора. Требования к содержанию и оформлению
* ГОСТ 19.508-79 Руководство по техническому обслуживанию. Требования к содержанию и оформлению

**Задание № 1**

Разработать руководство пользователя программного средства.

**Варианты**

Для следующих ИС

1. «Спортивный комплекс»
2. «Почта»
3. «Рыболовецкая компания»
4. «Агентство недвижимости»
5. «Пункт проката автомобилей»
6. «Рекламное агентство»
7. «Туристическая фирма»
8. «Авторемонтная мастерская»
9. «Санаторий»
10. «Редакция журнала»
11. «Фотостудия»
12. «Ювелирная мастерская»
13. «Кадровое агентство»
14. «Студия звукозаписи»
15. «Хлебопекарня»
16. «Страховая компания»
17. «Паспортный стол»
18. «Станция техобслуживания»
19. «Кинотеатр»
20. «Туристическая фирма»
21. «Регистратура поликлиники»
22. «Санаторий»
23. «Отдел кадров»
24. «Автосалон»
25. «Ателье»
26. Аптека»
27. «Библиотека»
28. «Гостиница»
29. «Детский сад»
30. «Кинологический клуб»
31. «Медицинская страховая компания»
32. «Школа»

# Практическая работа №29. Изучение средств автоматизированного документирования

**Цель**: получение навыков разработки руководства пользователя программного средства

Система автоматизации документооборота, система электронного документооборота (СЭД) — автоматизированная многопользовательская система, сопровождающая процесс управления работой иерархической организации с целью обеспечения выполнения этой организацией своих функций. При этом предполагается, что процесс управления опирается на человеко-читаемые документы, содержащие инструкции для сотрудников организации, необходимые к исполнению.

Выделение данной категории автоматизированных систем (как и понятие документооборота) характерно для постсоветского пространства, близкие к ней глобальные классы программного обеспечения — системы управления записями и ECM-системы.

Основные понятия электронного документооборота

Документооборот — движение документов в организации с момента их создания или получения до завершения исполнения или отправления (ГОСТ Р 7.0.8-2013); комплекс работ с документами: приём, регистрация, рассылка, контроль исполнения, формирование дел, хранение и повторное использование документации, справочная работа.

Электронный документооборот (ЭДО) — единый механизм по работе с документами, представленными в электронном виде, с реализацией концепции «безбумажного делопроизводства».

Машиночитаемый документ — документ, пригодный для автоматического считывания содержащейся в нём информации, записанный на магнитных, оптических и других носителях информации.

Электронный документ (ЭД) — документ, созданный с помощью средств компьютерной обработки информации, который может быть подписан электронной подписью (ЭП) и сохранён на машинном носителе в виде файла соответствующего формата.

Электронная подпись (ЭП) — аналог собственноручной подписи, являющийся средством защиты информации, обеспечивающим возможность контроля целостности и подтверждения подлинности электронных документов.

Основные принципы электронного документооборота

Однократная регистрация документа, позволяющая однозначно идентифицировать документ.

Возможность параллельного выполнения операций, позволяющая сократить время движения документов и повышения оперативности их исполнения

Непрерывность движения документа, позволяющая идентифицировать ответственного за исполнение документа (задачи) в каждый момент времени жизни документа (процесса).

Единая (или согласованная распределённая) база документной информации, позволяющая исключить возможность дублирования документов.

Эффективно организованная система поиска документа, позволяющая находить документ, обладая минимальной информацией о нём.

Развитая система отчётности по различным статусам и атрибутам документов, позволяющая контролировать движение документов по процессам документооборота и принимать управленческие решения, основываясь на данных из отчётов.

Классификация систем электронного документооборота

Универсальные «коробочные» СЭДО:

* стандартный набор функций;
* невозможность полного соответствия потребностям конкретной организации;
* низкие временные затраты на приобретение и установку;
* относительно низкая стоимость;
* необходимость приобретения лицензии на каждое внедряемое рабочее место.

Индивидуально разрабатываемые СЭДО:

максимально персонифицированная система;

* большие временные затраты;
* высокая стоимость разработки;
* сопутствующие расходы: затраты на обучение сотрудников, покупку нового оборудования и программного обеспечения.

Комбинированные СЭДО:

* базовая платформа, к которой разрабатываются необходимые дополнительные модули;
* полное соответствие нуждам предприятия;
* небольшие временные затраты на разработку и внедрение;
* стоимость включает: цену базовой платформы и стоимость индивидуальной доработки, зависящей от сложности заказа;
* передача заказчику прав на продукт;
* простота освоения и использования;
* полная локализация;
* удобный интерфейс;
* взаимодействие с существующими офисными приложениями.

**Задание № 1**

Изучить и описать интерфейс СЭД МикроДок Документооборот