

Министерство сельского хозяйства Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Брянский государственный аграрный университет»

Институт энергетики и природопользования
Кафедра информатики, информационных систем и технологий

Федькова Н.А.

Методология и технология проектирования информационных систем

Методическое пособие

Брянская область, 2022

УДК 004.415.2 (07)

ББК 32.965.7

М 54

Методология и технология проектирования информационных систем: методическое пособие / сост.: Н. А. Федькова. – Брянск: Изд-во Брянский ГАУ, 2022. – 52 с.

Издание окажет помощь студентам направления подготовки 09.04.03 Прикладная информатика при выполнении лабораторных работ по дисциплине «Методология и технология проектирования информационных систем».

Рекомендовано к изданию.

Рецензенты: старший преподаватель кафедры информационных систем и технологий Ульянова Наталья Дмитриевна.

© Брянский ГАУ. 2022

© Издательство Брянского ГАУ, 2022

© Федькова Н.А., 2022

Содержание

Введение	4
Ramus - кроссплатформенная система моделирования и анализа бизнес-процессов.....	5
Стандарт IDEF0	6
Диаграмма потоков данных.....	11
Моделирование данных	12
Лабораторная работа №1 Создание контекстной диаграммы	14
Лабораторная работа №2 Создание классификатора и диаграммы декомпозиции	17
Лабораторная работа №3 Создание диаграммы декомпозиций второго уровня	21
Лабораторная работа №4 Создание диаграммы DFD.....	23
Лабораторная работа №5 Анализ и описание информационных потоков предметной области «Приёмная комиссия ВУЗа»	24
Введение в нотацию BPMN	25
События и шлюзы в BPMN	28
Инструменты персонализации в BPMN: задачи, зона ответственности	33
Практическое использование подпроцессов в BPMN	38
Средства оповещения в BPMN	42
Использование артефактов и данных в BPMN	46
Список использованных источников	51

Введение

Пособие направлено на изучение современных методов и средств проектирования информационных систем. Предусматривается изучение CASE-средств, как программного инструмента поддержки проектирования информационных систем (ИС). Курс предусматривает изучение: состава и структуры различных классов экономических ИС как объектов проектирования; современных технологий проектирования ИС и методик обоснования эффективности их применения; содержания стадий и этапов проектирования ИС и их особенностей при использовании различных технологий проектирования; целей и задач проведения предпроектного обследования объектов информатизации; методов моделирования информационных процессов предметной области; классификацию и общие характеристики современных CASE-средств.

Научной основой курса являются методологии системного анализа и моделирования, позволяющие на этапе создания информационной системы решить следующие основные задачи:

- обеспечение требуемой функциональности системы и адаптивности к изменяющимся условиям ее функционирования;
- проектирование реализуемых в системе объектов данных;
- проектирование программ и средств интерфейса (экранных форм, отчетов), которые будут обеспечивать выполнение запросов к данным;
- учет конкретной среды или технологии реализации проекта, а именно: топологии сети, конфигурации аппаратных средств, используемой архитектуры, параллельной обработки, распределенной обработки данных и т.п.

Программой курса предусматривается изучение CASE-инструментов поддержки проектирования информационных систем. Практикум дисциплины включает в себя задания для освоения учащимися инструментальных средств разработки и анализа функциональных и информационных моделей деятельности экономических объектов (предприятий и учреждений), являющихся основой проектирования информационных систем.

Дисциплина «Методология и технология проектирования информационных систем» имеет целью ознакомить учащихся с информационными технологиями анализа сложных систем и основанными на международных стандартах методами проектирования информационных систем, обучить студентов принципам построения функциональных и информационных моделей систем, проведению анализа полученных результатов, применению инструментальных средств поддержки проектирования экономических информационных систем.

Ramus - кроссплатформенная система моделирования и анализа бизнес-процессов

Ramus - инструмент для моделирования бизнес-процессов и автоматизированной разработки системы регламентирующей документации, кроссплатформенное программное обеспечение, предназначенное для построения систем управления предприятием.

Ramus полностью поддерживает методологию моделирования бизнес-процессов IDEF0 и DFD, а так же имеет ряд дополнительных возможностей призванных удовлетворить потребности команд разработчиков систем управления предприятиями.

Данный программный продукт создан с целью стать основным инструментом бизнес-аналитиков в проектах по построению или реорганизации систем управления предприятием. К таковым могут относиться: проекты по реинжинирингу бизнес-процессов, проекты внедрения процессного управления, проекты построения системы менеджмента качества, проекты построения системы управления знаниями и т.п.

Основными возможностями Ramus являются:

- моделирование процессов (согласно нотаций IDEF0 и DFD);
- разработка систем классификации и кодирования предприятия с внутренними перекрёстными связями, которая также тесно увязывается и с моделями процессов;
- формирование отчётности по моделям и системе классификации, в том числе и отчётности в форме такой регламентирующей документации как должностные инструкции и регламенты процессов;
- генерация сайта, который призван обеспечить доступ к данным моделей процессов, системы классификации и кодирования, а также к разнообразнейшей отчётности через веб-интерфейс.

Ramus имеет редактор диаграмм **IDEF0** и **DFD** эргономичность которого находится на уровне не ниже чем у аналогичных продуктов, имеющих схожие редакторы. Это проявляется в более лёгкой и быстрой навигации по модели, в более «умном» поведении объектов диаграмм, в поддержке шаблонов диаграмм, в возможности быстрого исправления допущенных ошибок, в том числе и в возможности отмены действий.

Так как, модели процессов реальных предприятий могут содержать многие тысячи разнообразнейших объектов (документы, персонал, функции и т.д.), то в Ramus предусмотрена возможность упорядочено хранить информацию об этих объектах в виде системы классификаторов.

Классификатор в Ramus – это систематизированный перечень наименований объектов с присвоенными кодами.

Классификация объектов значительно упрощает поиск и обработку информации об объектах модели, а так же и об объектах непосредственно на диаграммах процессов не представленных, но, так или иначе, относящихся к процессам предприятия.

Каждый элемент системы классификации, кроме собственно названия, может иметь дополнительные атрибуты, в которых можно упорядочено хранить разнообразнейшую информацию об объекте.

Стоит отметить, что для создания качественной и информативной отчётности по модели, крайне необходимо, чтобы вся информация проекта содержалась упорядочено в виде системы классификации.

Для генерации отчётности в Ramus присутствует редактор отчётности.

Наличествуёт поддержка шаблонов отчётов в формате XML которые могут быть экспортированы из файла или импортированы в файл.

Совокупность моделей, классификаторов, матричных проекций и отчётов, имеющих отношение к одному и тому же предприятию в дальнейшем будем называть Проект.

Просмотр всей информации Проекта может быть осуществлён через веб-браузер. Для этого разработан веб-сервер, который выводит информацию Проекта в виде набора HTML страниц, или же, попросту говоря, в виде сайта.

Это существенно упрощает использование и развёртку Ramus, так как избавляет от необходимости установки клиентской версии Ramus на АРМах пользователей, которые имеют доступ только на чтение информации Проекта. Всей или некоторой информации Проекта, что определяется настройками прав доступа.

К любому элементу системы классификации и кодирования можно прикреплять файлы, которые будут доступны для скачивания с сайта Проекта.

Использование технологии Java, при реализации программных модулей, позволяет использовать Ramus под разными видами операционных систем и аппаратных платформ (MS Windows, Linux, Mac OS, и т.д....).

Ramus может использоваться в **файловом (локальном) и сетевом вариантах**.

Сетевая версия Ramus позволяет распределять доступ пользователей к данным.

Сетевая версия Ramus использует стандартизированные протоколы обмена данными, что позволяет интегрировать Ramus с другими системами.

Но и без использования сетевой версии можно разделить работу над Проектом между несколькими разработчиками путём использования функции расщепления Проекта.

В Ramus включена поддержка нескольких языков графического интерфейса пользователя. Язык интерфейса зависит от региональных настроек операционной системы.

Кроме всего прочего, Ramus поддерживает возможность расширения функциональности с использованием сценариев на языке программирования JavaScript.

Стандарт IDEF0

IDEF0 — Function Modeling — методология функционального моделирования и графическая нотация, предназначенная для формализации и описания

бизнес-процессов. Отличительной особенностью IDEF0 является её акцент на соподчинённость объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временная последовательность.

Основные элементы и понятия IDEF0

Графический язык IDEF0 удивительно прост и гармоничен. В основе методологии лежат четыре основных понятия.

Первым из них является понятие **функционального блока** (Activity Box). Функциональный блок графически изображается в виде прямоугольника и олицетворяет собой некоторую конкретную функцию в рамках рассматриваемой системы. По требованиям стандарта название каждого функционального блока должно быть сформулировано в глагольном наклонении (например, «производить услуги», а не «производство услуг») или отглагольным существительным (например, автоматизация учёта услуг)

Типы функциональных блоков в Ramus:

- 1) комплекс процессов;
- 2) процесс;
- 3) под-процесс;
- 4) операция;
- 5) действие.

Каждая из четырех сторон функционального блока имеет своё определенное значение (роль), при этом:

1. Верхняя сторона имеет значение «Управление» (Control).
2. Левая сторона имеет значение «Вход» (Input).
3. Правая сторона имеет значение «Выход» (Output).
4. Нижняя сторона имеет значение «Механизм» (Mechanism).

Каждый функциональный блок в рамках единой рассматриваемой системы должен иметь свой уникальный идентификационный номер.

Вторым «китом» методологии IDEF0 является понятие **интерфейсной дуги** (Arrow). Также интерфейсные дуги часто называют потоками или стрелками. Интерфейсная дуга отображает элемент системы, который обрабатывается функциональным блоком или оказывает иное влияние на функцию, отображенную данным функциональным блоком.

Графическим отображением интерфейсной дуги является однонаправленная стрелка. Каждая интерфейсная дуга должна иметь свое уникальное наименование (Arrow Label). По требованию стандарта, наименование должно быть оборотом существительного.

В IDEF0 различают пять типов стрелок:

Управление (Control) — правила, стратегии, процедуры или стандарты, которыми руководствуется работа (функциональный блок). Каждая работа должна иметь хотя бы одну стрелку управления. Стрелка управления рисуется как входящая в верхнюю грань работы. Управление влияет на работу, но не преобразуется работой. Если цель работы — изменить процедуру или стратегию, то такая процедура или стратегия будет для работы входом. В случае воз-

никновения неопределенности в статусе стрелки (управление или вход) рекомендуется рисовать стрелку управления.

Вход (Input) — материал или информация, которые используются или преобразуются работой для получения результата (выхода). Допускается, что работа может не иметь ни одной стрелки входа. Каждый тип стрелок подходит к определенной стороне прямоугольника, изображающего работу, или выходит из нее. Стрелка входа рисуется как входящая в левую грань работы. Очень часто сложно определить, являются ли данные входом или управлением. В этом случае подсказкой может служить информация о том, перерабатываются/изменяются ли данные в работе или нет. Если изменяются, то, скорее всего, это вход, если нет — управление.

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. Стрелка выхода рисуется как исходящая из правой грани работы.

Механизм (Mechanism) — ресурсы, которые выполняют работу, например, персонал предприятия, станки, устройства и т. д. Стрелка механизма рисуется как входящая в нижнюю грань работы. По усмотрению аналитика стрелки механизма могут не изображаться в модели.

Вызов (Call) — специальная стрелка, указывающая на другую модель работы. Стрелка вызова рисуется как исходящая из нижней грани работы. Стрелка вызова используется для указания того, что некоторая работа выполняется за пределами моделируемой системы.

С помощью интерфейсных дуг отображают различные объекты, в той или иной степени определяющие процессы, происходящие в системе. Такими объектами могут быть элементы реального мира (детали, вагоны, сотрудники и т. д.) или потоки данных и информации (документы, данные, инструкции и т. д.).

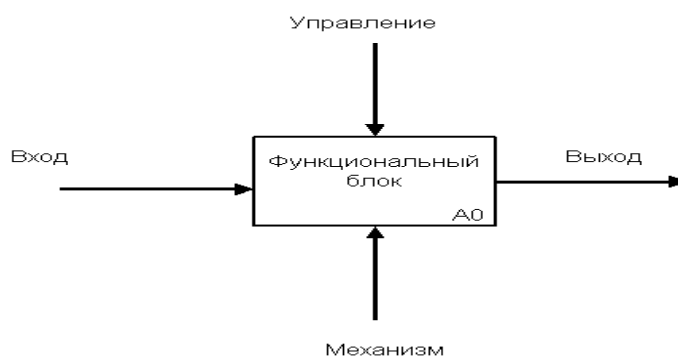


Рисунок 1 – Функциональный блок

В зависимости от того, к какой из сторон подходит данная интерфейсная дуга, она носит название «входящей», «исходящей» или «управляющей». Кроме того, «источником» (началом) и «приемником» (концом) каждой функциональной дуги могут быть только функциональные блоки. Необходимо отметить, что любой функциональный блок по требованиям стандарта должен иметь по крайней мере одну управляющую интерфейсную дугу и одну исходящую.

Обязательное наличие управляющих интерфейсных дуг является одним из главных отличий стандарта IDEF0 от других методологий классов DFD (Data Flow Diagram) и WFD (Work Flow Diagram).

Третьим основным понятием стандарта IDEF0 является **декомпозиция** (Decomposition). Принцип декомпозиции применяется при разбиении сложного процесса на составляющие его функции. При этом уровень детализации процесса определяется непосредственно разработчиком модели.

Декомпозиция позволяет постепенно и структурировано представлять модель системы в виде иерархической структуры отдельных диаграмм, что делает ее менее перегруженной и легко усваиваемой.

В процессе декомпозиции, функциональный блок, который в контекстной диаграмме отображает систему как единое целое, подвергается детализации на другой диаграмме. Получившаяся диаграмма второго уровня содержит функциональные блоки, отображающие главные подфункции функционального блока контекстной диаграммы и называется дочерней (Child diagram) по отношению к нему (каждый из функциональных блоков, принадлежащих дочерней диаграмме соответственно называется дочерним блоком – Child Box). В свою очередь, функциональный блок - предок называется родительским блоком по отношению к дочерней диаграмме (Parent Box), а диаграмма, к которой он принадлежит – родительской диаграммой (Parent Diagram). Каждая из подфункций дочерней диаграммы может быть далее детализирована путем аналогичной декомпозиции соответствующего ей функционального блока.

Важно отметить, что в каждом случае декомпозиции функционального блока все интерфейсные дуги, входящие в данный блок, или исходящие из него фиксируются на дочерней диаграмме. Этим достигается структурная целостность IDEF0 – модели.

Наглядно принцип декомпозиции представлен на рис. 2.

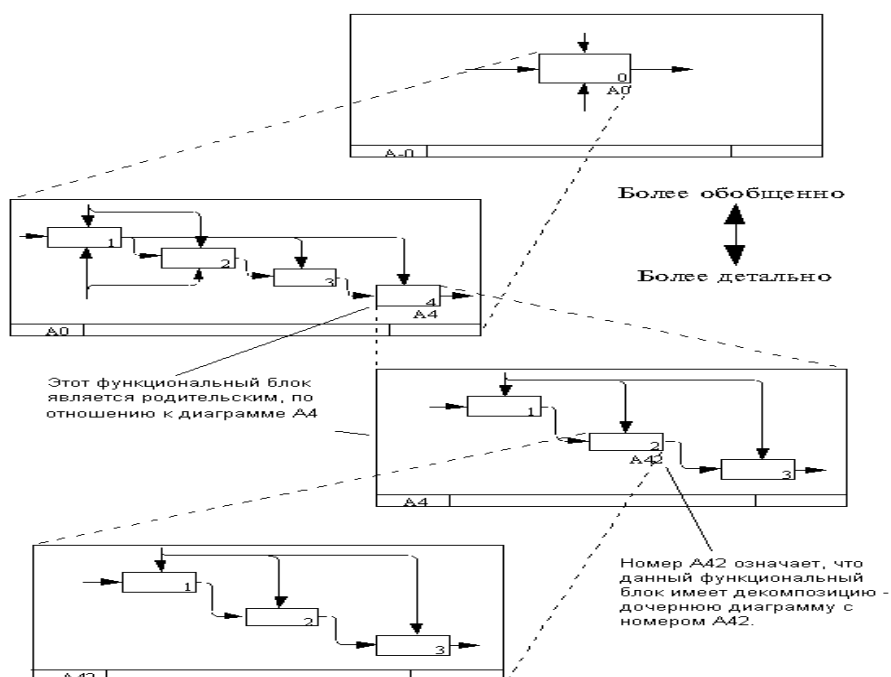


Рисунок 2 – Декомпозиция функционального блока

Принципы ограничения сложности IDEF0-диаграмм. Обычно IDEF0-модели несут в себе сложную и концентрированную информацию, и для того, чтобы ограничить их перегруженность и сделать удобочитаемыми, в соответствующем стандарте приняты соответствующие ограничения сложности:

- Ограничение количества функциональных блоков на диаграмме тремя-шестью. Верхний предел (шесть) заставляет разработчика использовать иерархии при описании сложных предметов, а нижний предел (три) гарантирует, что на соответствующей диаграмме достаточно деталей, чтобы оправдать ее создание;

- Ограничение количества подходящих к одному функциональному блоку (выходящих из одного функционального блока) интерфейсных дуг четырьмя.

Разумеется, строго следовать этим ограничениям вовсе необязательно, однако, как показывает опыт, они являются весьма практичными в реальной работе.

Разновидности IDEF.

1. IDEF0 - методология функционального моделирования. С помощью наглядного графического языка IDEF0, изучаемая система предстает перед разработчиками и аналитиками в виде набора взаимосвязанных функций (функциональных блоков - в терминах IDEF0). Как правило, моделирование средствами IDEF0 является первым этапом изучения любой системы.

2. IDEF1 – методология моделирования информационных потоков внутри системы, позволяющая отображать и анализировать их структуру и взаимосвязи.

3. IDEF1X (IDEF1 Extended) – методология построения реляционных структур. IDEF1X относится к типу методологий “Сущность-взаимосвязь” (ER – Entity-Relationship) и, как правило, используется для моделирования реляционных баз данных, имеющих отношение к рассматриваемой системе.

4. IDEF2 – методология динамического моделирования развития систем. В связи с весьма серьезными сложностями анализа динамических систем от этого стандарта практически отказались, и его развитие приостановилось на самом начальном этапе. Однако в настоящее время присутствуют алгоритмы и их компьютерные реализации, позволяющие превращать набор статических диаграмм IDEF0 в динамические модели, построенные на базе “раскрашенных сетей Петри” (CPN – Color Petri Nets).

5. IDEF3 – методология документирования процессов, происходящих в системе, которая используется, например, при исследовании технологических процессов на предприятиях. С помощью IDEF3 описываются сценарий и последовательность операций для каждого процесса. IDEF3 имеет прямую взаимосвязь с методологией IDEF0 – каждая функция (функциональный блок) может быть представлена в виде отдельного процесса средствами IDEF3.

6. IDEF4 – методология построения объектно-ориентированных систем. Средства IDEF4 позволяют наглядно отображать структуру объектов и заложенные принципы их взаимодействия, тем самым позволяя анализировать и оптимизировать сложные объектно-ориентированные системы.

7. IDEF5 – методология онтологического исследования сложных систем. С помощью методологии IDEF5 онтология системы может быть описана при помощи определенного словаря терминов и правил, на основании которых могут быть сформированы достоверные утверждения о состоянии рассматриваемой системы в некоторый момент времени. На основе этих утверждений формируются выводы о дальнейшем развитии системы и производится её оптимизация. В рамках этой статьи мы рассмотрим наиболее часто используемую методологию функционального моделирования IDEF0.

Диаграмма потоков данных

DFD — общепринятое сокращение от англ. Data Flow Diagrams — диаграммы потоков данных. Так называется 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 (data flow diagram, DFD) — один из основных инструментов структурного анализа и проектирования информационных систем, существовавших до широкого распространения UML. Несмотря на имеющее место в современных условиях смещение акцентов от структурного к объектно-ориентированному подходу к анализу и проектированию систем, «старинные» структурные нотации по-прежнему широко и эффективно используются как в бизнес-анализе, так и в анализе информационных систем.

Исторически сложилось так, что для описания диаграмм DFD используются две нотации — Йордана (Yourdon) и Гейна-Карсона (Gane-Sarson), отличающиеся синтаксисом.

Информационная система принимает извне потоки данных. Для обозначения элементов среды функционирования системы используется понятие внешней сущности. Внутри системы существуют процессы преобразования информации, порождающие новые потоки данных. Потоки данных могут поступать на вход к другим процессам, помещаться (и извлекаться) в накопители данных, передаваться к внешним сущностям.

Модель DFD, как и большинство других структурных моделей — иерархическая модель. Каждый процесс может быть подвергнут декомпозиции, то есть разбиению на структурные составляющие, отношения между которыми в той же нотации могут быть показаны на отдельной диаграмме. Когда достигнута требуемая глубина декомпозиции — процесс нижнего уровня сопровождается мини-спецификацией (текстовым описанием).

Кроме того, нотация DFD поддерживает понятие подсистемы — структурного компонента разрабатываемой системы.

Нотация DFD — удобное средство для формирования контекстной диаграммы, то есть диаграммы, показывающей разрабатываемую АИС в коммуникации с внешней средой. Это — диаграмма верхнего уровня в иерархии диаграмм DFD. Ее назначение — ограничить рамки системы, определить, где заканчивается разрабатываемая система и начинается среда. Другие нотации, ча-

сто используемые при формировании контекстной диаграммы — диаграмма SADT, Диаграмма вариантов использования.

External Entity (Внешняя сущность) представляет собой материальный предмет или физическое лицо, представляющее собой источник или приемник информации, например, заказчики, персонал, поставщики, клиенты, склад. Определение некоторого объекта или системы в качестве внешней сущности указывает на то, что она находится за пределами границ анализируемой ИС. В процессе анализа некоторые внешние сущности могут быть перенесены внутрь диаграммы анализируемой ИС, если это необходимо, или, наоборот, часть процессов ИС может быть вынесена за пределы диаграммы и представлена как внешняя сущность. Внешняя сущность обозначается квадратом, расположенным как бы "над" диаграммой и бросающим на нее тень, для того, чтобы можно было выделить этот символ среди других обозначений.

Process (Системы и подсистемы) - при построении модели сложной ИС она может быть представлена в самом общем виде на так называемой контекстной диаграмме в виде одной системы как единого целого, либо может быть декомпозирована на ряд подсистем. Процесс представляет собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом. Физически процесс может быть реализован различными способами: это может быть подразделение организации (отдел), выполняющее обработку входных документов и выпуск отчетов, программа, аппаратно реализованное логическое устройство и тд.

Моделирование данных

CA ERwin® Data Modeler Community Edition — это бесплатное средство моделирования, которое является базовой версией флагманского продукта CA ERwin Data Modeler Standard Edition. Это решение предоставляет множество базовых функций моделирования данных с ограничением до 25 объектов моделей, в том числе для проектирования и создания баз данных, сравнения моделей, определения стандартов и др.

Решение CA ERwin Data Modeler Community Edition помогает организациям управлять сложной инфраструктурой данных с помощью следующих основных возможностей:

- Визуализация сложных структур данных. Модели данных создаются автоматически, что дает простое графическое представление для визуализации сложных структур баз данных.
- Создание проектов баз данных. Создавайте проекты баз данных непосредственно на основе визуальных моделей, что позволяет повысить эффективность и уменьшить число ошибок.
- Определение стандартов. Повторно используемые стандарты, такие как шаблоны моделей, домены, стандарты именования, улучшают качество и эффективность.

- Сравнение моделей и баз данных. Функция Complete Compare осуществляет сравнение моделей, скриптов и баз данных, выводя все различия на экран (в версии Community Edition данные доступны только для чтения).

- Отчетность и публикация. Простой, интуитивно понятный интерфейс Report Designer позволяет создавать отчеты в виде текстовых и HTML-файлов, которые могут содержать диаграммы и метаданные.

CA ERwin Data Modeler Community Edition — это простое в использовании средство для управления небольшими базами данных и получения навыков моделирования баз данных. Моделирование данных повышает производительность за счет простоты в использовании графической среды, которая упрощает проектирование и обслуживание баз данных, автоматизирует множество трудоемких задач и улучшает коммуникацию в организации, помогая повысить эффективность и качество данных, сокращая при этом расходы.

Совместная работа над моделями данных позволяет реализовать эффективную коммуникацию между лицами, отвечающими за бизнес и технические вопросы, помогая обеспечивать согласование бизнес-требований с технической реализацией бизнес-данных. Визуальные инструменты проектирования позволяют разработчикам баз данных решать задачи проектирования и устранять проблемы до каких-либо существенных вложений ресурсов, что помогает организации быстрее реагировать на растущие потребности бизнеса, выявляя влияние изменений на информационные активы и обеспечивая быструю реакцию на непрерывные изменения и быстрый рост среды данных.

Поддерживаемые среды

CA ERwin Data Modeler Community Edition работает в следующих средах:

- Windows XP
- Windows 2003 и 2008 Server SP2
- Windows Vista
- Windows 7
- Windows 8

и поддерживает следующие среды баз данных:

- Oracle
- SQL Server
- DB2 UDB
- MySQL
- ODBC
- Sybase

ERwin имеет два уровня представления модели - **логический** и **физический**.

Логический уровень - это абстрактный взгляд на данные, на нем данные представляются так, как выглядят в реальном мире, и могут называться так, как они называются в реальном мире, например, «Постоянный клиент», «Отдел» или «Фамилия сотрудника». Объекты модели, представляемые на логическом уровне, называются сущностями и атрибутами (подробнее о сущностях и атрибутах будет рассказано ниже). Логическая модель данных может быть построе-

на основе другой логической модели, например, на основе модели процессов (см. VPwin). Логическая модель данных является универсальной и никак не связана с конкретной реализацией СУБД.

Физическая модель данных, напротив, зависит от конкретной СУБД, фактически являясь отображением системного каталога. В физической модели содержится информация о всех объектах БД. Поскольку стандартов на объекты БД не существует (например, нет стандарта на типы данных), физическая модель зависит от конкретной реализации СУБД. Следовательно, одной и той же логической модели могут соответствовать несколько разных физических моделей. Если в логической модели не имеет значения, какой конкретно тип данных имеет атрибут, то в физической модели важно описать всю информацию о конкретных физических объектах - таблицах, колонках, индексах, процедурах и т.д.

Лабораторная работа №1 **Создание контекстной диаграммы**

В качестве примера рассматривается деятельность промышленной компании. Компания занимается сборкой и продажей настольных компьютеров и ноутбуков. Компания не производит компоненты самостоятельно, а только собирает и тестирует компьютеры.

Деятельность компании состоит из следующих элементов:

- менеджер по работе с клиентами принимают заказы клиентов;
- операторы группируют заказы по типам клиентов;
- операторы собирают (согласно заказов) и тестируют компьютеры;
- операторы упаковывают компьютеры согласно заказам;
- кладовщик отгружает клиентам заказ.

Компания использует приобретенную бухгалтерскую ИС, которая позволяет оформить заказ, счет и отследить платежи по счетам.

1. Запустите программу Ramus. После запуска программы на экране появится окно начала работ Выберите опцию "**Создать**" и нажмите "**ОК**".

1. Внесите имя автора, название проекта, название модели и выберите опцию "IDEF0" (рис. 3).

2. На следующем шаге укажите, что модель используется "отделом стратегического планирования и развития" (рис. 4).

3. В описании проекта укажите "Это учебная модель, описывающая деятельность компании" (рис. 5), перейдите к следующему шагу.

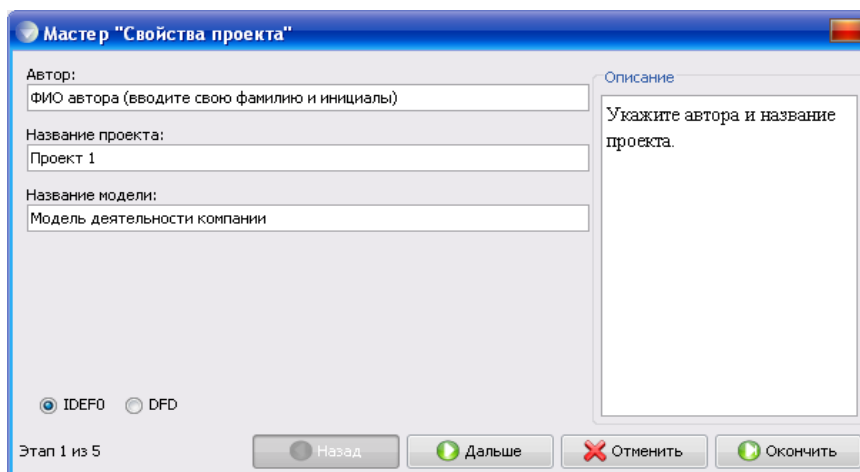


Рисунок 3 – Внесение первоначальных сведений о проекте (этап 1)

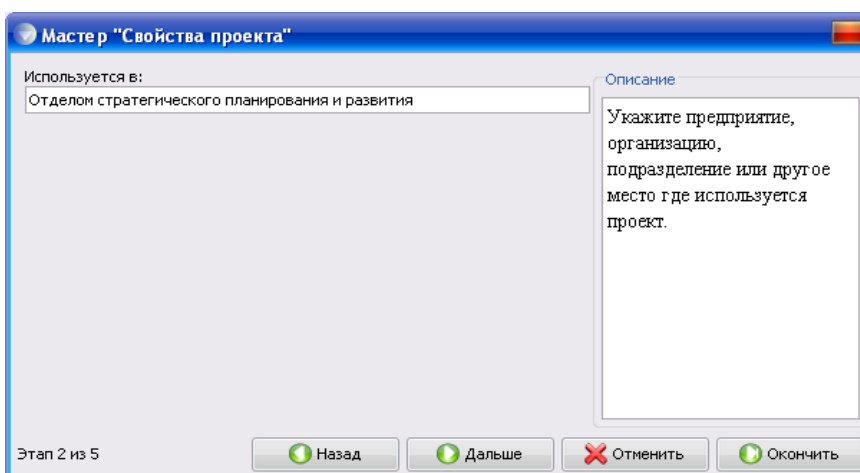


Рисунок 4 – Внесение первоначальных сведений о проекте (этап 2)

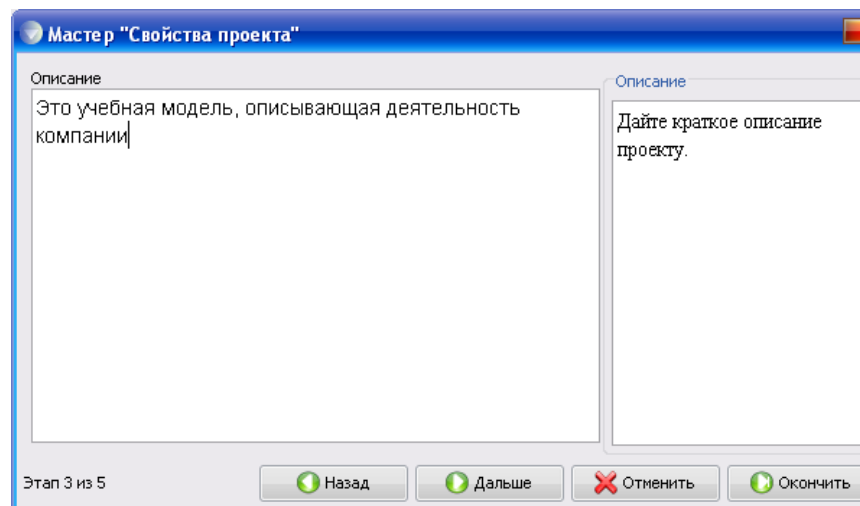


Рисунок 5 – Внесение первоначальных сведений о проекте (этап 3)

5. Раздел "классификаторы" оставьте незаполненным и нажмите "Дальше".
6. В следующем диалоговом окне нажмите "Окончить" и перейдите к рабочему интерфейсу программы.

Через меню Диаграмма/Свойства модели можно отредактировать метаданные модели, а именно: название модели, описание, место ее использования.

Активируйте окно модели, кликнув на область моделирования. Создайте контекстную диаграмму, нажав на соответствующую кнопку выбора элемента и после этого щёлкнув по пустой области построения диаграммы.

Перейдите в режим редактирования контекстной диаграммы, нажав правой кнопкой мыши на объекте и выбрав опцию "Редактировать активный элемент". В закладке "Название" введите "Деятельность компании".

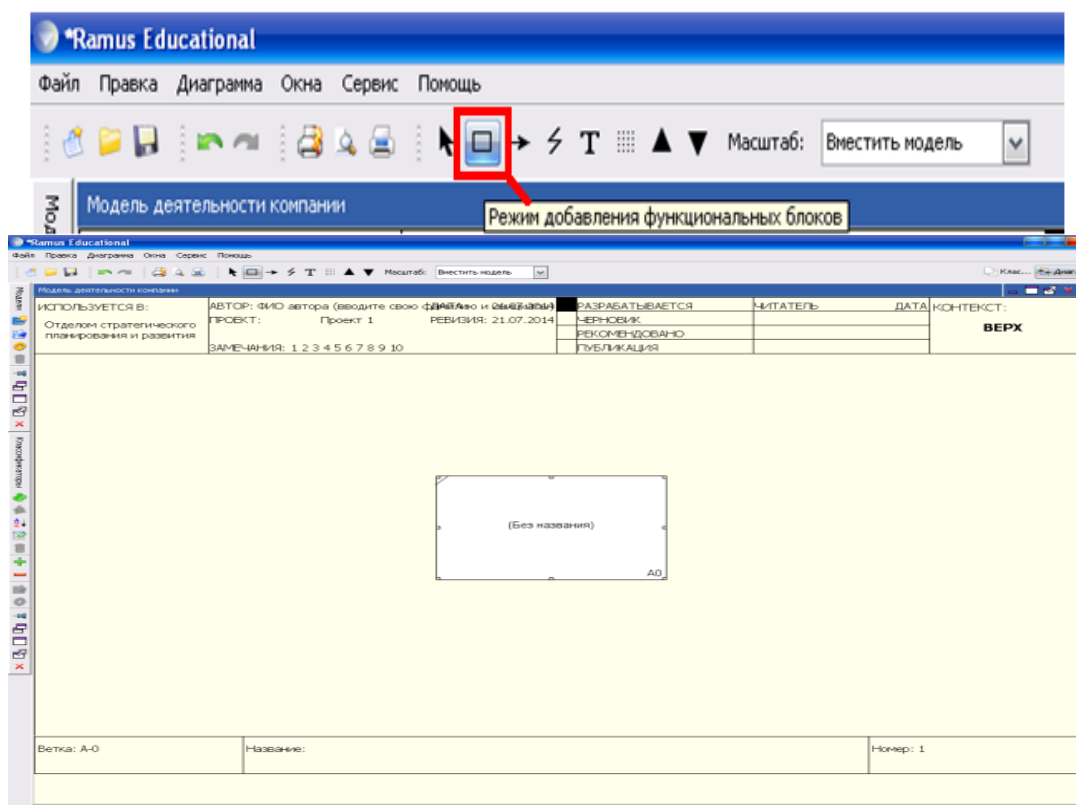


Рисунок 6 – Результат выполнения указанных действий

Создайте стрелки на контекстной диаграмме в соответствии с информацией, приведенной в таблице 1.


Для создания стрелок необходимо перейти в режим построения стрелок с помощью кнопки , навести курсор на исходную точку стрелки (левая, верхняя и нижняя граница области построения модели или правая граница контекстной диаграммы), после того, как область будет подсвечена черным цветом, кликнуть один раз и аналогичным образом обозначить конец стрелки (правая, верхняя и нижняя граница контекстной диаграммы или правая граница области построения модели). Перемещать стрелки и их названия можно по принципам стандартного механизма drag&drop.

Таблица 1 – Описание стрелок контекстной диаграммы

Название	"Смысловая нагрузка"	Тип
Персонал	Обслуживание системы (реализация управления деятельностью компании на всех этапах выполнения работ)	Механизм
Звонки клиентов	Запросы информации, заказы, необходимость технической поддержки и т.д.	Вход
Правила и процедуры	Правила продаж, инструкции по сборке, процедуры тестирования, критерии производительности, различные законодательные акты, ГОСТы, стандарты и т. д.	Управляющее воздействие
Документы на получение проданных товаров со склада	Договор на заключение сделки купли-продажи, счёт на оплату, накладная (по которой производится выдача товаров со склада), гарантийный талон.	Выход

После того как все стрелки созданы необходимо выровнять их относительно блока диаграммы - для этого вызвать контекстное меню (щелкнув по блоку работ) и выбрать Центровать присоединённые стрелки.

На рис. 7 представлен результат построения контекстной диаграммы.

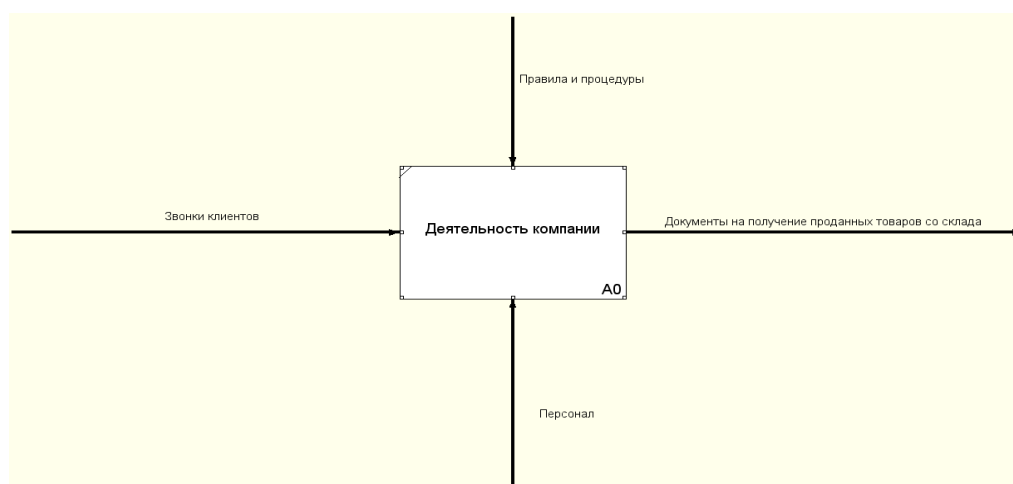


Рисунок 7 – Контекстная диаграмма

Лабораторная работа №2

Создание классификатора и диаграммы декомпозиции

Создание классификатора

Для того что бы создать классификатор необходимо перейти в соответствующий режим, для чего нужно щелкнуть ЛКМ по кнопке Классификаторы в правом верхнем углу (на панели инструментов).

Создайте элемент классификатора «Роли», в который внесите следующие элементы:

- менеджер по работе с клиентами;
- персонал производственного отдела;
- кладовщик.

Результаты выполнения указанных действий представлены на рис. 8.

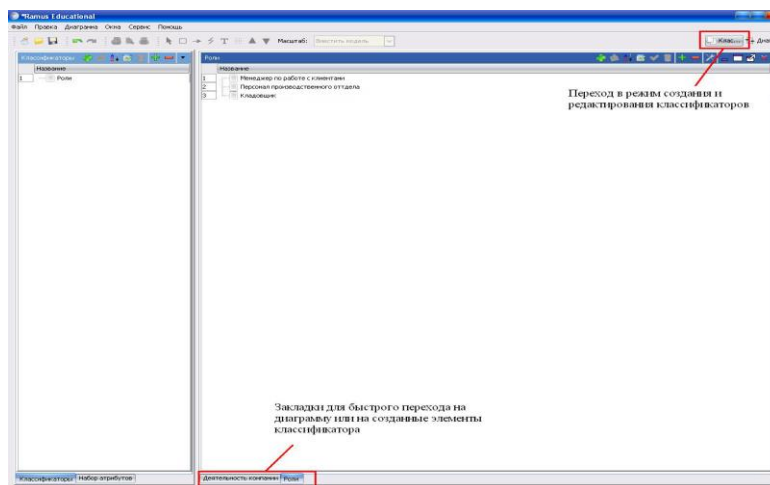


Рисунок 8 – Создание классификатора

Воспользовавшись закладкой в нижней части окна программы перейдите к контекстной диаграмме.

Создание диаграммы декомпозиций

1. Выберите кнопку перехода на уровень ниже ▼ на панели инструментов.
2. В диалоговом окне укажите число работ на диаграмме нижнего уровня - "3", а нотацию декомпозиции – *IDEF* (рис. 9), затем нажмите ОК. Автоматически будет создана диаграмма декомпозиции.

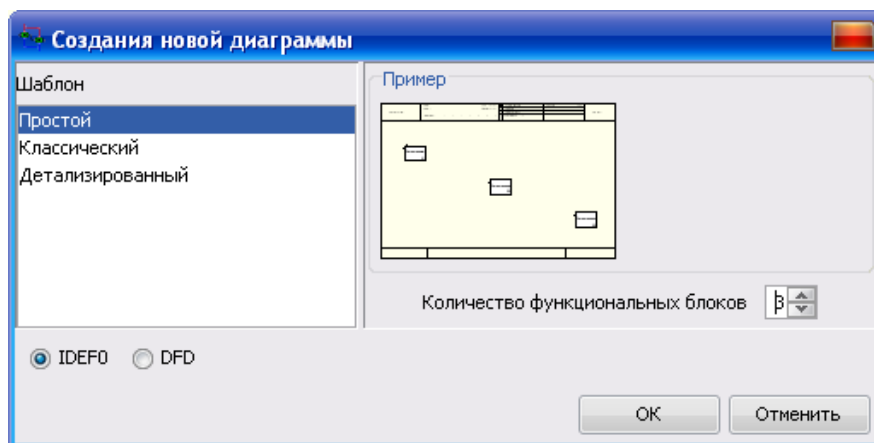


Рисунок 9 – Диалоговое окно декомпозиции работ

3. Правой кнопкой мыши щелкните по 1-ой работе, выберите "Редактировать активный элемент" и на вкладке "Название" укажите имя работы. Повторите операцию для всех трех работ.

4. Перейдите в режим рисования стрелок. Произведите связывание граничных стрелок с функциональными объектами, как показано на рис. 10. Для связывания граничных стрелок наводите курсор на сами стрелки, а не на границы области построения моделей.

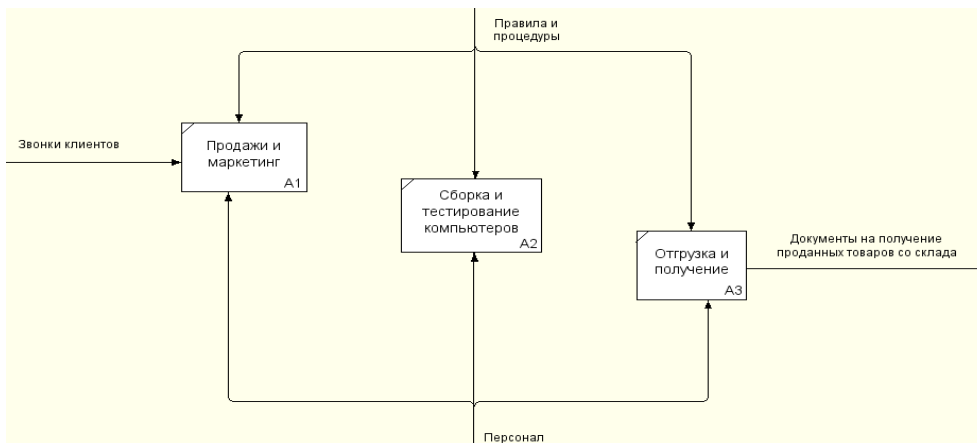


Рисунок 10 – Связывание граничных стрелок на диаграмме декомпозиции

5. Правой кнопкой мыши щёлкните по ветви стрелки "Сборка и тестирование компьютеров", переименуйте ее в "Правила сборки и тестирования" (рис. 12).

Далее необходимо конкретизировать роли персонала, отвечающего за выполнение конкретного блока работ. Правой кнопкой мыши щелкните по ветви стрелки механизма работы "Продажи и маркетинг" и в открывшемся диалоговом окне на закладке «Поток» сначала нажмите Очистить, а потом Добавить. Выберите «Роли» и поставьте галочку возле «Менеджер по работе с клиентами» (рис. 11) и нажмите Ок.

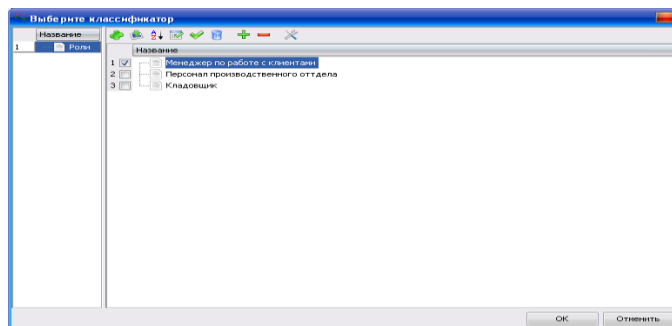


Рисунок 11 – Назначение названия стрелки из классификатора

Аналогично обозначить все остальные стрелки механизма (рис. 12).

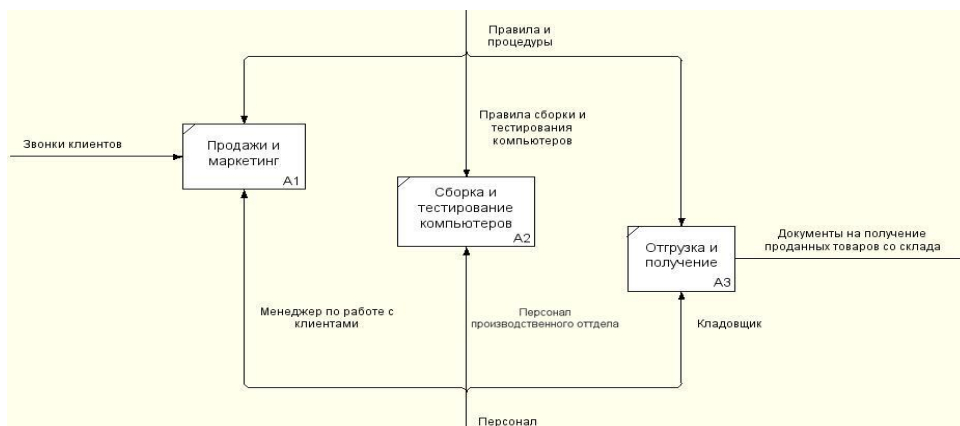


Рисунок 12 – Присвоение названий ветвям стрелок диаграммы декомпозиции

6. Создайте новые внутренние стрелки, как показано на рисунке (рис.13).

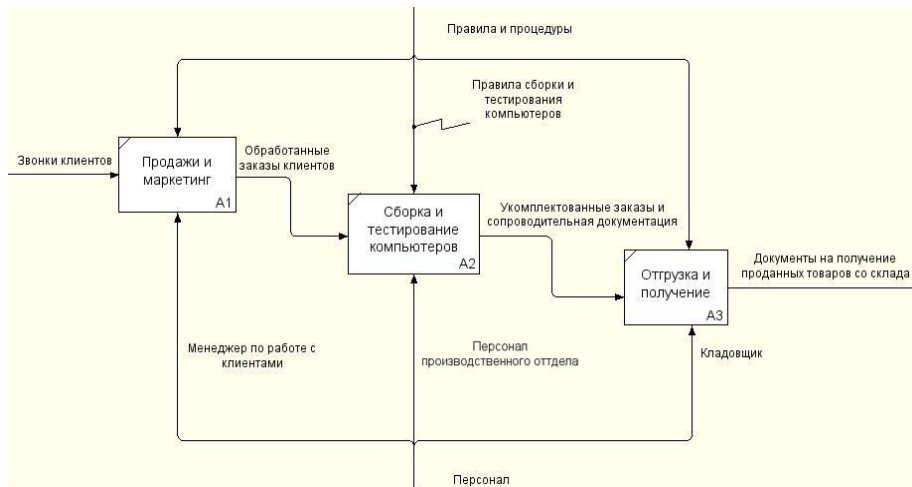


Рисунок 13 – Внутренние стрелки диаграммы декомпозиции

7. Создайте новую граничную стрелку "Маркетинговые материалы", выходящую из работы "Продажи и маркетинг" Эта стрелка автоматически не попадает на диаграмму верхнего уровне и имеет квадратные скобки у окончания —□—. Щелкните правой кнопки мыши по квадратным скобкам и выберите в контекстном меню "Туннель" одну из двух опций: Создать стрелку и Обозначить туннель круглыми скобками, в нашем случае - первый вариант (рис. 14).

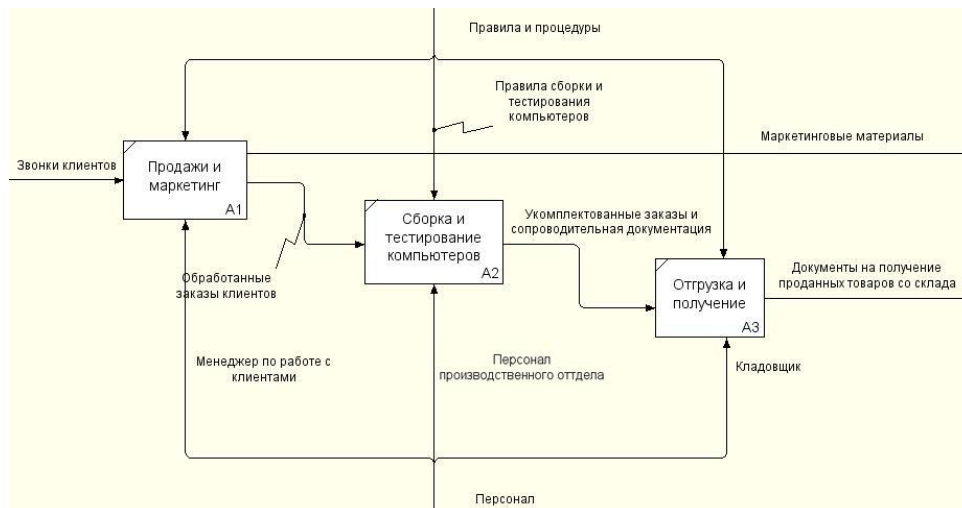


Рисунок 14 – Результат туннелирования стрелок

После этого необходимо доработать классификатор, добавив в Персонал производственного отдела две роли: Диспетчер и Тестировщики (рис. 15).

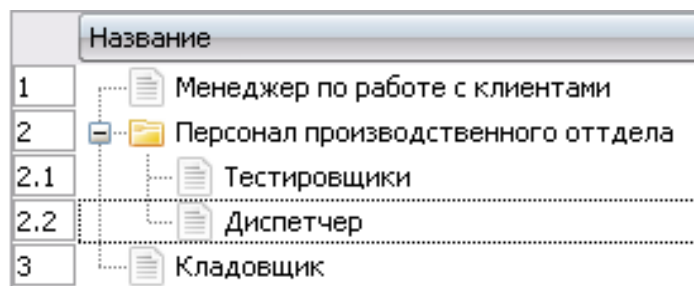


Рисунок 15 – Редактирование классификатора

Лабораторная работа №3

Создание диаграммы декомпозиций второго уровня

Декомпозируем работу "Сборка и тестирование компьютеров". В результате проведенного анализа получена следующая информация о процессе:

Производственный отдел получает заказы от отдела клиентов по мере их поступления.

- диспетчер координирует работу сборщиков, сортирует заказы, группирует и дает указания на отгрузку компьютеров, когда они готовы;
- каждые 2 часа диспетчер группирует заказы - отдельно для настольных компьютеров и ноутбуков - и направляет их на участок сборки;
- сотрудники участка сборки собирают компьютеры согласно спецификациям заказа и инструкциям по сборке. Когда группа компьютеров, соответствующая группе заказов, собрана, она направляется на тестирование. Тестировщик тестируют каждый компьютер и, в случае необходимости, заменяет неисправные компоненты.
- Тестировщики направляют результаты тестирования диспетчеру, который на основании этой информации принимает решение о передаче компьютеров, соответствующих группе заказов, на отгрузку.

1. На основе информации из таблиц 2 и 3 внесите новые работы и стрелки на диаграмму декомпозиции А2.

2. Далее следует разнести стрелки, перенесённые с диаграммы верхнего уровня (как показано на рис. 16).

Таблица 2 – Описание функциональных блоков диаграммы декомпозиции А2

Название функционального блока	Описание
Отслеживание расписания и управление сборкой и тестирование	Просмотр заказов, установка расписания выполнения заказов, просмотр результатов тестирования, формирования групп заказов на сборку и отгрузку
Сборка настольных компьютеров	Сборка настольных компьютеров в соответствии с инструкциями и указаниями диспетчера
Сборка ноутбуков	Сборка ноутбуков в соответствии с инструкциями и указаниями диспетчера
Тестирование компьютеров	Тестирование компьютеров и компонентов. Замена неработающих компонентов.

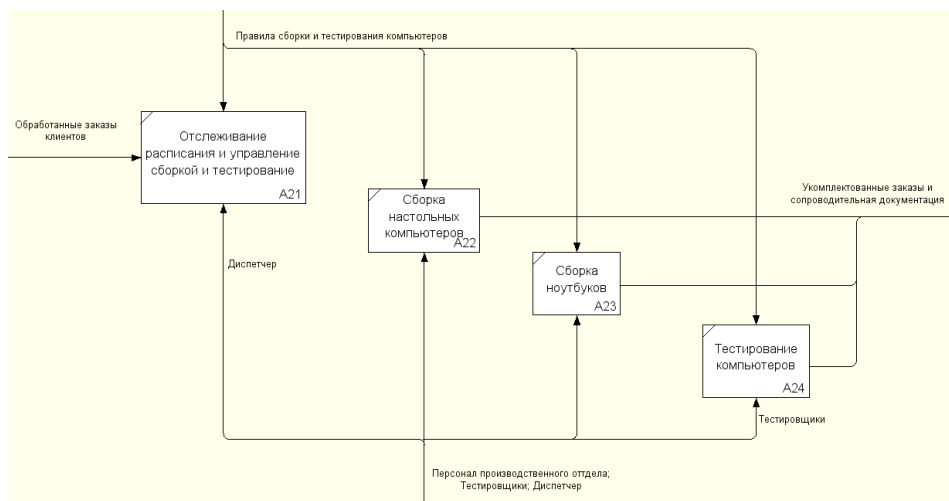


Рисунок 16 – Связывание граничных стрелок на диаграмме декомпозиции A2

Таблица 2 – Описание стрелок диаграммы декомпозиции A2

Название стрелки	Начало стрелки	Тип начала стрелки	Окончание стрелки	Тип окончания стрелки
Диспетчер (назначить из классификатора)	Персонал производственного отдела	Механизм (ветка стрелки)	Отслеживание расписания и управление сборкой и тестированием	Механизм
Заказы на настольные компьютеры	Отслеживание расписания и управление сборкой и тестированием	Выход	Сборка настольных компьютеров	Вход
Заказы на ноутбуки	Отслеживание расписания и управление сборкой и тестированием	Выход	Сборка компьютеров	Вход
Компоненты	Туннелированная стрелка	Вход	Сборка настольных компьютеров	Вход
			Сборка ноутбуков	Вход
			Тестирование компьютеров	Вход
Настольные компьютеры	Сборка настольных компьютеров	Выход	Тестирование компьютеров	Вход
Ноутбуки	Сборка ноутбуков	Выход	Тестирование компьютеров	Вход
Результаты тестирования	Тестирование компьютеров	Выход	Отслеживание расписания и управление сборкой и тестированием	Вход
Тестировщики (назначить из классификатора)	Персонал производственного отдела	Механизм (ветка стрелки)	Тестирование компьютеров	Механизм
Указание передать компьютеры на отгрузку	Отслеживание расписания и управление сборкой и тестированием	Выход	Тестирование компьютеров	Управляющее воздействие

3. Произведите туннелирование и связку граничных стрелок, если это необходимо. Результат представлен на рис. 17.

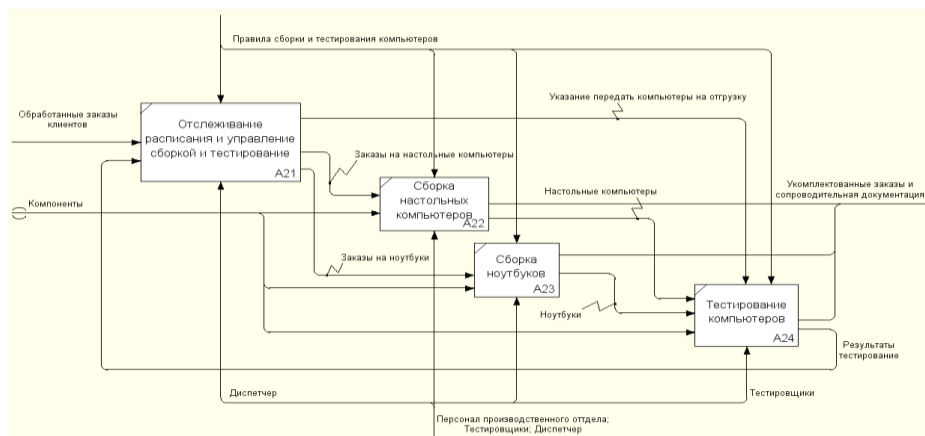


Рисунок 17 – Результат декомпозиции процесса «Сборка и тестирование»

Лабораторная работа №4 Создание диаграммы DFD



1. Создайте контекстную диаграмму процесса "Оформление заказов" (Файл / Новый проект).

2. Декомпозируйте созданную контекстную диаграмму "Оформление заказов", для чего в диалоговом окне выберите количество элементов декомпозиции - 2, тип диаграммы - **DFD**. Нажмите "OK" и внесите в диаграмму DFD имена работ:

- Проверка и внесение клиента
- Внесение заказа

3. Создайте классификаторы:

- Список клиентов
- Список продуктов
- Список заказов
- Заявки на заказ

4. Внесите в модель соответствующие хранилища данных при помощи кнопки  (рис. 18), а также внешнюю ссылку "Заявки на заказ", используя кнопку .

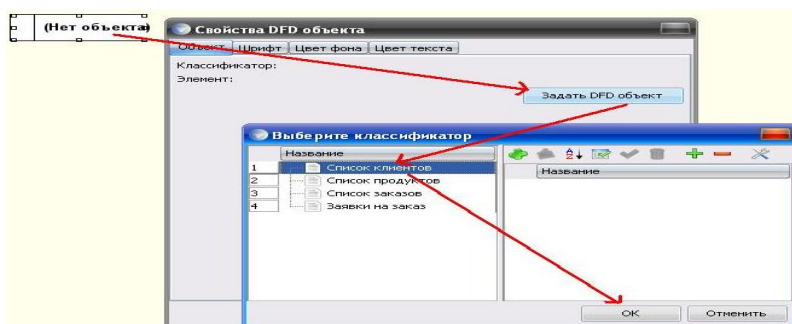


Рисунок 18 – Внесение хранилища данных

5. На основе следующей информации постройте DFD-модель процесса "Оформление заказов":

- Процесс "Оформление заказов" состоит из двух подпроцессов: проверка и внесение клиентов, и внесение заказов. Для выполнения этих процессов необходим список клиентов, список продуктов и для регистрации результатов выполнения процессов реестр списка заказов. Проверка и внесение клиентов в базу данных клиентов осуществляется на основе информации из заявок на заказ, а также после анализа информации в списке клиентов.

- Внесение заказов производится только при наличии информации о соответствующем клиенте в списке клиентов и только на те товары, которые занесены в список продуктов компании. Существуют возможность использовать ранее созданные заказы, сохраненные в списке заказов.

- Имейте в виду, что связь между некоторыми функциональными объектами и хранилищами данных может быть двунаправленной (исходящая и входящая стрелки).

6. Сверьте построенную Вами модель с моделью на рис. 19.

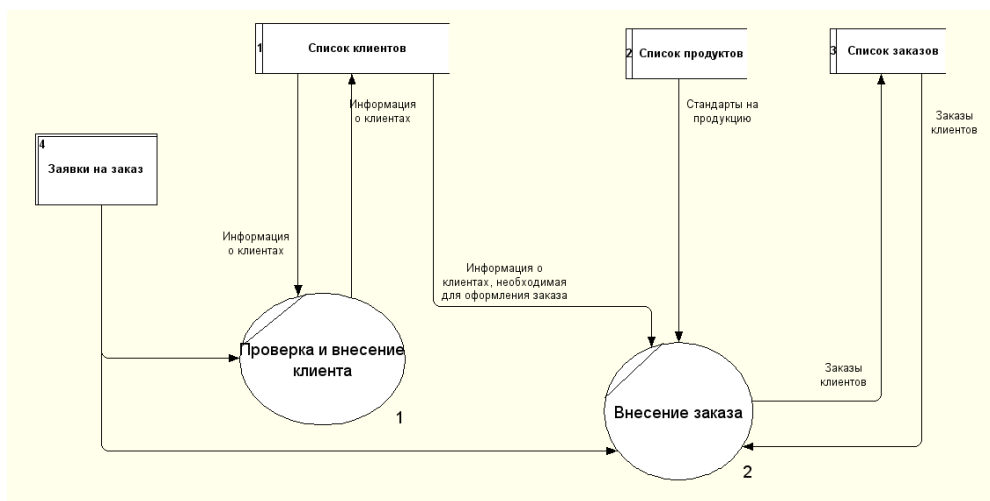


Рисунок 19 – DFD-диаграмма декомпозиции процесса оформления заказа

Лабораторная работа №5

Анализ и описание информационных потоков предметной области «Приёмная комиссия ВУЗа»

Задание: требуется представить процесс приёма, регистрации и дальнейшей обработки документов абитуриентов в виде диаграмм методологии IDEF0.

Схема документооборота, анализируемого участка учёта представлена на рис. 26.

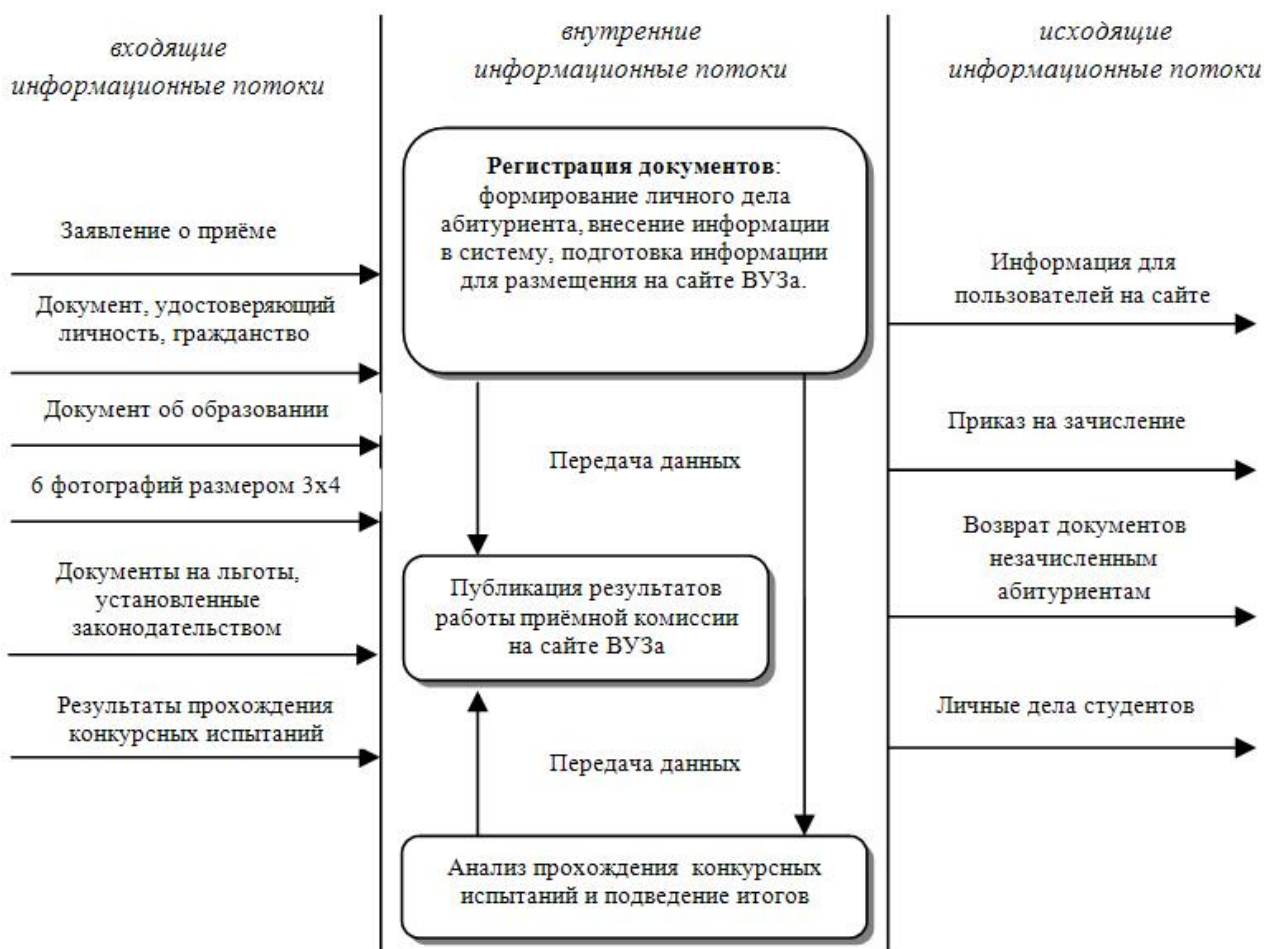


Рисунок 26 – Схема документооборота

Ведение в нотацию BPMN

Нотация по моделированию бизнес-процессов BPMN (The Business Process Modeling Notation) - это новый стандарт для моделирования бизнес-процессов и сетевых услуг, который впервые был выпущен BPMN Notation Working Group в мае 2004 года. Последняя версия нотации BPMN 2.0 вышла в 2010 году. Оригинальная спецификация (на английском языке) изготовлена группой компаний «Object Management Group».

Нотация BPMN описывает условные обозначения для отображения бизнес-процессов в виде диаграмм бизнес-процессов. BPMN ориентирована как на технических специалистов (разработчиков, ответственных за реализацию процессов), так и на бизнес-пользователей (бизнес-аналитиков, создающих и улучшающих процессы) и менеджеров, следящих за процессами и управляющих ими. Следовательно, BPMN призвана служить связующим звеном между фазой дизайна бизнес-процесса и фазой его реализации. Для этого язык использует базовый набор интуитивно понятных элементов, которые позволяют определять сложные семантические конструкции.

Людям, занимающимся бизнесом, крайне удобно работать с бизнес-процессами, отображаемыми в виде блок-схем. Множество бизнес-аналитиков

проектируют и описывают бизнес-процессы компаний с помощью простых диаграмм в нотации BPMN, т.к. язык нотации понятен даже на уровне пользователя. При этом модели процессов, описанных в нотации BPMN, являются ИСПОЛНЯЕМЫМИ (т.е. реализуются в любой BPM-системе), а не только документируются. Для детального описания процессов существуют программные решения, которые способны преобразовать диаграммы в исполняемые процессы, эти процессы затем могут быть запущены и работать в реальном времени.

Одной из причин создания BPMN явилась необходимость построения простого механизма для проектирования и чтения как простых, так и сложных моделей бизнес-процессов. Для удовлетворения двух этих противоречащих требований был применен подход систематизации графических элементов нотации по категориям. Результатом явился небольшой перечень категорий нотаций, позволивший людям, работающим с диаграммами BPMN, без труда распознавать основные типы элементов и осуществлять корректное чтение схем.

С точки зрения легкости чтения и понимания процессов нотация BPMN 2.0 вне конкуренции. Моделирование в BPMN осуществляется посредством диаграмм с небольшим числом графических элементов. Это помогает пользователям быстро понимать логику процесса.

Любой процесс, описанный в нотации BPMN, представляет собой последовательное или параллельное выполнение различных действий (операций) с указанием определённых бизнес-правил. Рассмотрим простой пример процесса «Обработка заказа», который может реализовываться в рамках продажи и аренды велосипедов через интернет-магазин.

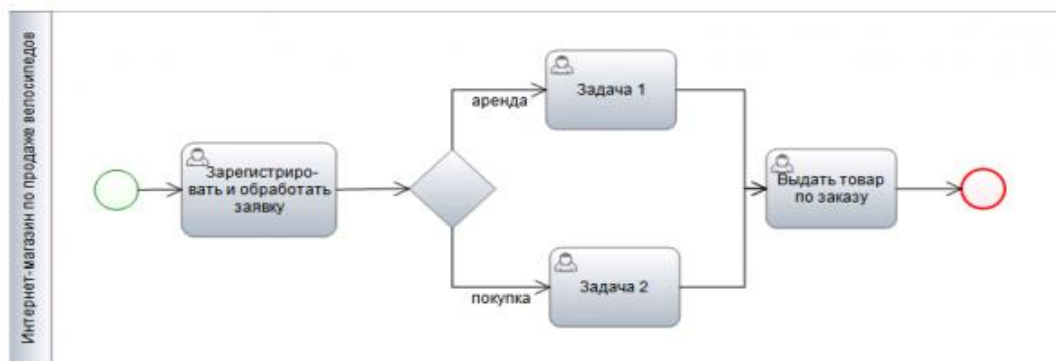


Рисунок 27 – Процесс «Обработка заказа»

Чтение процесса всегда начинается со Стартового события (зеленого кружка).



Рисунок 28 – Стартовое событие

Как видно из названия, Стартовое событие указывает на то, в какой точке берет начало тот или иной процесс. В контексте потока операций Стартовое событие является начальной точкой в процессе; это означает, что никакой входящий поток операций не может быть соединен со стартовым событием. Стартовое событие в нотации BPMN изображается в виде круга со свободным центром.

Примечание: стартовым событием процесса-примера является звонок или письмо от клиента на сайт компании (интернет-магазина).

Далее от Стартового события выполнение процесса идет по линиям (Поток операций) до Конечного события (красный кружок), их может быть несколько.

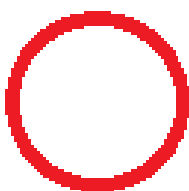


Рисунок 29 – Конечное событие

Конечное событие указывает на то, в какой точке завершается тот или иной процесс. В контексте Потока операций Конечное событие завершает ход Процесса; это означает, что никакой Исходящий поток операций не может быть соединен с Конечным событием.

Конечное событие представляет собой круг, выполненный одиночной, жирной линией. Толщина линии должна быть жирной настолько, чтобы без труда можно было отличить Конечное событие от Стартового.

Примечание: в приведённом примере Стартовое и Конечное события для большего удобства различаются так же по цвету. Конечное событие процесса «Обработка заказа» отображает завершение процесса – выдачей заказанного товара.

Вся логика работы (ход) процесса выражается во всевозможных элементах, расположенных между Стартовым и Конечным событием. Основным элементом, отражающим деятельность, выполняемую внутри процесса, являются Действия. Действия – это точки выполнения работ в ходе Процесса. Они относятся к выполняемым элементам Процесса BPMN. Действие может быть, как элементарным, так и неэлементарным (составным).

Элементарное Действие выражается в выполнении одной единственной Задачи. Графически Задача изображается в виде прямоугольника с закругленными углами. Самой распространённой Задачей является типичная для технологического процесса задача, где человек участвует в качестве исполнителя. Такие Задачи называются Пользовательскими.

Примечание: в рамках процесса «Обработка заказа» основными действиями являются задачи «Зарегистрировать и обработать заявку», «Оформить заявку на покупку» и «Оформить заявку на аренду».

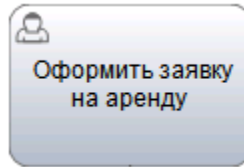


Рисунок 30 – Пользовательская задача

Другой элемент нотации, часто используемый в описании процессов – Шлюзы (Условия). Графический элемент Шлюза представляет собой небольшой ромб, используемый во многих нотациях схем бизнес-процессов для изображения ветвления и знакомый большинству инструментов моделирования. Фактически Шлюз - есть совокупность входов и выходов.



Рисунок 31 – Шлюз

Шлюзы используются для контроля расхождений и схождений потока операций в рамках процесса. Термин шлюз подразумевает пропускное устройство, которое либо позволяет осуществлять переход через шлюз, либо нет.

Примечание: в приведённом примере, в зависимости от желания клиента (купить или арендовать велосипед), заявка оформляется в формате покупки либо аренды соответственно. В данном процессе Шлюз указывает, что процесс может пойти только в одном из описанных направлений, т.е. либо покупка, либо аренда.

Даже самый сложный процесс, описанный в нотации BPMN, легко читается любым бизнес-пользователем, владеющим знанием основных элементов Процесса BPMN.

События и шлюзы в BPMN

Второй Урок практического курса BPMN посвящён рассмотрению следующих графических элементов спецификации BPMN и их использованию при описании бизнес-процессов: События и Шлюзы (исключающие, не исключающие и параллельные).

Расширим ранее приведённый пример процесса (Урок 1) с дополнительными действиями.

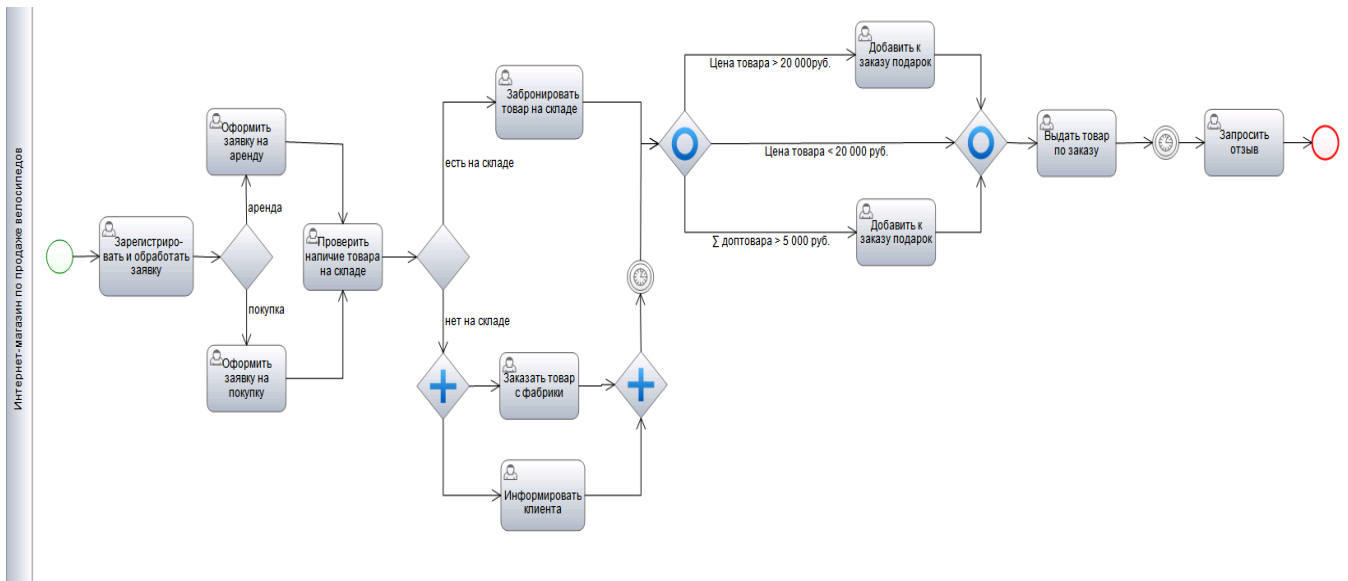


Рисунок 32 – Процесс «Обработка заказа»

Кроме Стартового и Конечного события, в описании бизнес-процессов используются Промежуточные события. Промежуточное событие влияет на ход процесса, однако, не может являться началом или завершением процесса и само по себе не является полноценным действием. Примерами Промежуточных события являются: ожидание определённого времени, события, письма.

Промежуточное событие изображается в виде круга со свободным центром. Для отличия от Стартового и Конечного типов Событий, изображение круга Промежуточное событие выполнено двойной тонкой линией.

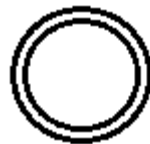


Рис. 2.1. Промежуточное событие

BPMN выделяет несколько типов каждого События: Сообщение, Таймер, Эскалация и другие - их достаточно большое количество. Полный перечень можно посмотреть в нотации BPMN (русский перевод) раздел 10.4. «Событие». Для определения типа События используются различные маркеры, позволяющие отличить данный тип События от другого.

Маркер – это специальный значок, рисуемый в центре круга События. Он влияет на характер работы События.

Для выбора типа события необходимо из контекстного меню выбрать пункт – Event type



- простое **Стартовое событие**.



- **Стартовое событие** –таймер позволяет запустить процесс по таймеру в определённый момент времени (пример с совещанием в 9.00час.), т.е. меняется характер запуска бизнес-процесса.



- **Стартовое событие-сообщение** показывает, что от участника поступает Сообщение, которое инициирует запуск Процесса (например, от клиента приходит оплата товара, приходит сообщение об оплате и запускается процесс выдачи товара).



- **простое Конечное событие**. Данный тип Конечного события не подразумевает какой-то определенный результат.



- **Конечное событие-сообщение** служит для указания того, что Участник отправил Сообщение в момент завершения Процесса.



- **обычное Промежуточное событие**.



- **маркер часов** показывает, что используется событие-таймер. В ход процесса при этом останавливается на определённое время.



- **маркер конверта** определяет **Промежуточное событие** типа «Сообщение», которое используется для отправки сообщения другому участнику Процесса. Данный вид Промежуточного события может стать инициатором Стартового события-сообщения другого процесса.

Наиболее часто для описания бизнес-процессов применяют Промежуточное событие-таймер, которое позволяет моделировать моменты времени, периоды и таймауты. Промежуточное событие данного типа графически изображается с аналоговыми часами внутри круга.

Промежуточное событие-таймер используется для того, чтобы приостановить ход процесса до определенного времени либо задать определённую цикличность выполнения действия (например, планирование и информирование о совещании каждую неделю в понедельник в 9.00 час.).

Примечание: в приведённом примере процесса используется промежуточное событие-таймер с целью приостановить ход процесса пока не придёт заказанный с фабрики товар. т.е. пока заказанный товар не придёт на склад магазина, формировать заявку и рассматривать вопрос о добавлении подарка не начнут. Таймер (Промежуточное событие-таймер) позволяет обозначить на диаграмме процесса это ожидание.

Второй пример использования промежуточного события-таймер в рамках процесса - ожидание момента, когда заказанный товар придёт клиенту; после чего с ним необходимо связаться, чтобы выяснить удовлетворённость

клиента работой интернет-магазина либо попросить оставить отзыв на сайте компании-продавца (обратная связь).

Ещё один элемент BPMN, используемый в рамках предыдущего Урока 1 – **Шлюз (Условия)**, так же имеет несколько вариаций. В первом примере (Урок 1) был использован **Исключающий Шлюз «ИЛИ»** (Эксклюзивный). Исключающие **Шлюзы** включаются в состав бизнес-процесса для разделения Потока операций на несколько альтернативных маршрутов. Для процесса с исключаяющим типом **Шлюза** может быть выбран лишь один из предложенных маршрутов (поэтому определяется как **Шлюз «ИЛИ»**).



Рисунок 34 – Исключающий Шлюз

Графический элемент *Исключающий Шлюз* не имеет внутренних маркеров.

Условие можно представить себе в виде вопроса, который появляется в какой-то точке процесса и предполагает несколько вариантов ответов. Каждый из предлагаемых ответов связан с определённым направлением потока операций.

Примечание: в процессе-примере Исключительный Шлюз используется два раза: первый Шлюз определяется вопросом «Покупают велосипед или арендуют?» и в зависимости от ответа процесс протекает либо в одном направлении «Оформить заявка на покупку», либо в другом – «Оформить заявку на аренду»; второй Шлюз определяется вопросом «Есть ли заказанный товар на складе?», по результатам которого товар либо бронируют на складе, либо, в случае отсутствия на складе, делают заказ на фабрику-производитель. Обе задачи выполняться одновременно не будут.

Второй тип **Шлюза**, так же часто используемый в описании процессов – **Параллельный Шлюз «И»**. Данный тип **Шлюза** используется для создания параллельных маршрутов и их синхронизации (объединения).



Рисунок 35 – Параллельный Шлюз

Графический элемент *Параллельный Шлюз* содержит внутренний маркер, выполненный в виде знака «+», что позволяет отличить данный тип Шлюза от других.

С помощью *Параллельного Шлюза* параллельные маршруты создаются без необходимости проверки каких-либо условий. При разветвлении все исходящие потоки (маршруты) активизируются одновременно. Закрывающий Шлюз

используется для синхронизации, т.е. он ожидает завершения выполнения всех входящих ветвей (маршрутов) и только затем активирует выходной поток.

Примечание: В рассматриваемом примере (Рис.32) Параллельный Шлюз разъединяет процесс на два параллельных маршрута с операциями «Заказать товар на фабрике» и «Информировать клиента», которые выполняются одновременно, не исключая друг друга: сотрудник может связаться с фабрикой и заказать нужный велосипед, и предупредить клиента о задержке по выдаче товара. Второй - закрывающий Параллельный Шлюз - используется для синхронизации потока операций, т.е. обязательно ожидается завершение обоих действий.

Ещё один, но менее распространённый тип Шлюза – *Неисключающий (Неэксклюзивный) Шлюз «И/ИЛИ»*. *Неисключающие Шлюзы* используются для разделения потока операций на несколько альтернативных и параллельных маршрутов.

Для данного экземпляра процесса может быть выбран лишь один из предложенных маршрутов – или параллельный (т.е. оба маршрута выполняются параллельно) или альтернативный (т.е. процесс пойдёт только по одному, соответствующему условиям, маршруту).



Рисунок 36 – Неисключающий Шлюз

Графический элемент *Неисключающий Шлюз* содержит внутренний маркер, выполненный в виде круга, что позволяет отличить данный тип Шлюза от других.

Примечание: В процессе-примере (Рис.32) описаны несколько возможных маршрутов потока операций, разделённые Неисключающим Шлюзом:

Два возможных альтернативных маршрута:

1. *При условии, что сумма заказанного товара (велосипеда) больше 20 000 руб., то к заказу добавляется подарок. При этом дополнительного товара заказано не было. Маршрут идёт только в одном направлении.*

2. *Если сумма товара меньше 20 000руб., и в заказе доптовара нет, то подарок не добавляется. Маршрут также идёт только в одном направлении.*

Два возможных параллельных маршрута:

1. *Если велосипед заказан на сумму больше 20 000руб., и дополнительно заказаны товары на сумму от 5 000руб., то клиент получит два подарка. Ход процесса идёт параллельно по двум маршрутам.*

2. *Если сумма велосипеда меньше 20 000руб., но дополнительные товары превышают 5000 руб., то клиент получает подарок. Также процесс проходит по двум направлениям параллельно. Закрывающий Неисключающий Шлюз, также, как и Параллельный Шлюз, синхронизирует потоки операций.*

В нотации BPMN описывается несколько типов **Шлюзов**, однако здесь мы привели три наиболее распространённые при описании бизнес-процессов элементы Условия.

Инструменты персонализации в BPMN: задачи, зона ответственности

Третий Урок практического курса BPMN посвящён рассмотрению следующих графических элементов спецификации BPMN и их использованию при описании бизнес-процессов: **Пул**, **Дорожка**, а также более подробно будет рассмотрен уже знакомый вам элемент нотации **Задача**.

Для отображения взаимодействия между участниками бизнес-процесса в нотации BPMN используются элементы **Пул** и **Дорожка**. Однако, опциональность каждого элемента довольно разнообразна и сложна в понимании. Чаще, при описании процессов в BPMS-системах (системах управления бизнес-процессами) в определение элемента Пул вкладывают понятие области процесса (совокупность всех действий и ответственных за их выполнение лиц). Пулы не отражают конкретные внутренние процессы участников, они скорее показывают глобальные взаимодействия и зависимости между участниками процесса.

Пул и **Дорожки** в рамках одного потока операций в нотации BPMN называются Оркестровкой, т.е. представляют собой диаграмму (схему), показывающую последовательность выполнения действий в рамках одного процесса.

Разработчики BPMS-систем элемент **Дорожка** часто используют в качестве внутренних ролей (Зоны Ответственности), что представляет собой распределение обязанностей среди участников процесса (менеджер, директор и т.п.). В области одного **Пула** могут находиться несколько **Дорожек** (участников процесса).

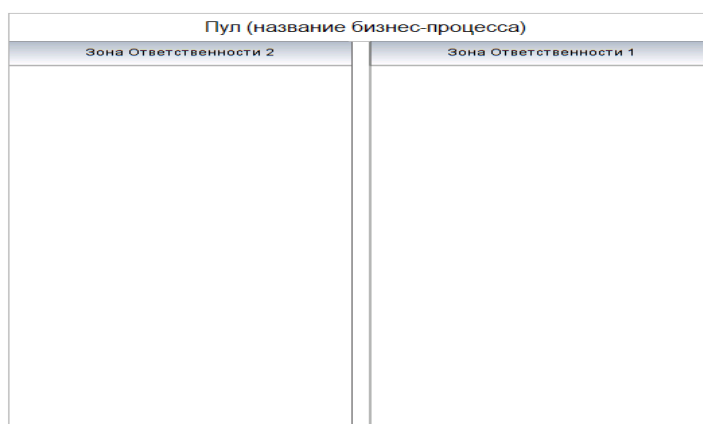


Рисунок 37 – Графическое изображение Пула и Дорожек (Зон Ответственности).

Дорожка представляет собой прямоугольник, в котором описываются все действия ответственного за выполнение задач лица. Дорожки в нотации BPMN могут располагаться как вертикально, так горизонтально.

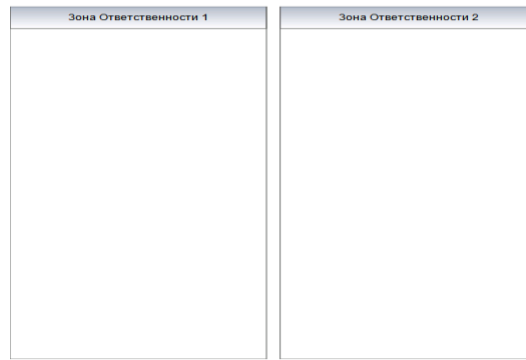


Рисунок 38 – Возможное расположение **Дорожек** в процессе

В случае если в системах BPMS каждый бизнес-процесс описывается отдельно (на одном листе описывается один процесс), то **Пул**, чаще всего, не визуализируют.

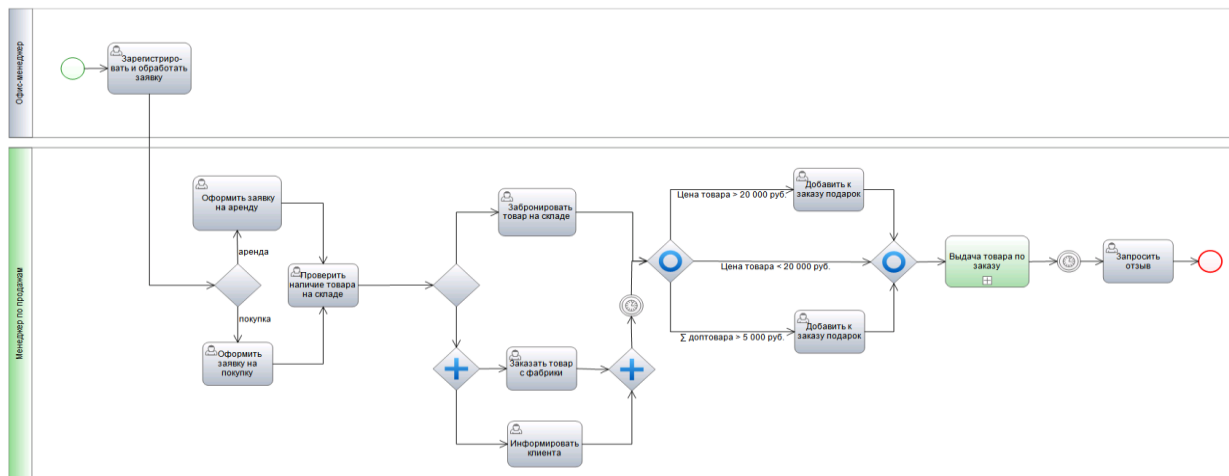


Рисунок 39 – Процесс «Обработка заказа»

Примечание: в расширенном примере-процессе (рис.39) выделены две **Дорожки** – участники процесса «Офис-менеджер» и «Менеджер по продажам». Офис-менеджер отвечает за регистрацию и обработку заказа. Далее процесс переходит в зону ответственности Менеджера по продажам. Разработчики BPMS для дополнительного удобства сделали возможность задавать зону ответственности динамической, т.е. она не определяет конкретного сотрудника, а лишь показывает роль (должность) ответственного за выполнение задач. В рамках примера «Менеджер по продажам» задаётся динамической зоной, т.к. менеджеров в компании может быть много, а задачи будут выполняться одним из этих сотрудников.

Дорожка фактически является зоной ответственности участника: любой элемент, помещенный в дорожку, выполняется исполнителем, прописанным в заголовке дорожки. Так, например, на рисунке 5 видно, что поток операций переходит из дорожки «Офис-менеджер» в дорожку «Менеджер по продажам». **Задачу**, находящуюся в дорожке «Офис-менеджера» исполняет именно офис-менеджер. В одном процессе может быть неограниченное количество **Дорожек**.

Таким образом, можно описать всех участников процесса – поток операций будет определять, какие задачи, кем, в какой момент, и в каком порядке будут выполняться в рамках процесса.

Ещё один элемент нотации BPMN, который хотелось бы рассмотреть более подробно в рамках данного Урока – **Задача**. **Задача**, как элементарное **Действие** процесса, рассматривалась в контексте первого Урока. Однако там был задействован основной тип **Задач** при описании бизнес-процессов **Пользовательская Задача**.

BPMN выделяет несколько типов **Задач**, что позволяет описывать различия в присутствии им поведении, характерные для каждого из типов. **Тип Задачи** определяется маркерами внутри графического элемента.

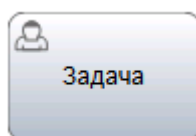


Рисунок 40 – Пользовательская задача

Графически пользовательская **Задача** отображается в виде прямоугольника с закругленными углами, который выполнен одинарной тонкой линией, и отличается от других типов **Задач** наличием маркера в виде верхней части фигуры человека.

Пользовательская Задача представляет собой задачу, типичную для технологического процесса (упорядоченной последовательности взаимосвязанных действий), где человек выступает в роли исполнителя и выполняет **Задачи** при содействии других людей или программного обеспечения.

Ещё один тип **Задач**, требующих участие людей – **Ручное выполнение**.

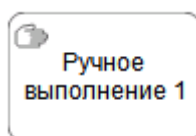


Рисунок 41 Задача – Ручное выполнение

Графически **Ручное выполнение** отображается как прямоугольник с закругленными углами, границы которого выполнены одинарной тонкой линией. Содержит маркер в виде руки, позволяющий отличать данный тип **Задач** от других.

Ручное выполнение представляет собой **Задачу**, выполнение которой подразумевает действия человека и исключает использование каких-либо автоматизированных механизмов исполнения или приложений. **Ручное выполнение** не поддается управлению никаким механизмом выполнения бизнес-процесса. Такой тип **Задач** можно отнести к неуправляемым, т.е. к **Задачам**, начало и за-

вершение выполнения которых не отслеживается механизмами выполнения бизнес-процесса. В случае BPMС-систем (систем управления бизнес-процессами), это предполагает собой некоторое действие, которое исполнитель выполняет за рамками системы.

Примером такого типа **Задач** может служить установка телефонного аппарата на территории заказчика специалистом по обслуживанию телефонов или проведение совещания.

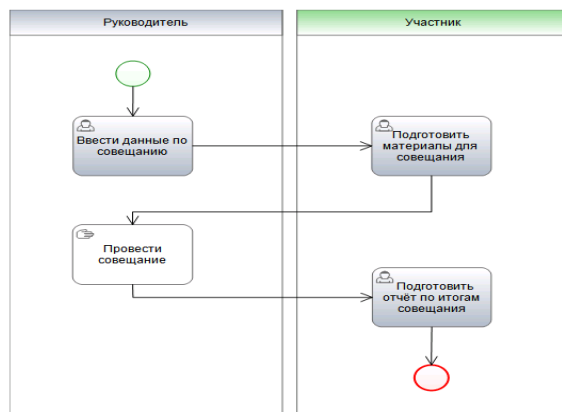


Рисунок 42 – Использование Задачи – Ручное выполнение в описании бизнес-процессов

Примечание: На рисунке 42 приведён простой пример использования задачи **Ручное выполнение** в рамках процесса организации и проведения совещания в компании. Руководитель компании запускает процесс, заполняя данные по совещанию: дата, время, тематика, участник совещания (выбирает сотрудника, с которым планируется собрание). В данном примере для простоты рассмотрен вариант проведения совещания руководителя только с одним сотрудником компании. Сотруднику назначается задача «Подготовить материалы для совещания». Руководитель проводит совещание и, т.к. эта задача исполняется без подключения автоматизации, на диаграмме процесса действие выполнено в виде графического элемента **Ручное выполнение**. По итогам совещания сотрудник подготавливает отчёт и на этом процесс завершается.

BPMN также описывает тип **Задач**, которые выполняются без участия человека. К таким **Задачам** относится **Задача-сценарий**.

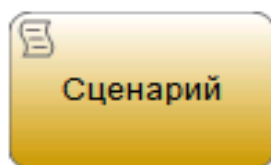


Рисунок 43 – Задача-сценарий

Графически **Задача-сценарий** отображается как прямоугольник с закругленными углами. Содержит маркер, позволяющий отличать данный тип **Задач**

от других. Маркер расположен в левом верхнем углу фигуры Задачи данного типа.

Задача-сценарий обозначает выполнение в процессе некоторого автоматизированного действия. То есть оно выполняется без конкретного исполнителя средствами самой ВРМС-системы. Например, это может быть подсчет каких-либо данных - подсчет сумм, введенных пользователем. Такая работа прекрасно выполняется компьютером без участия человека. ВРМС-система не только делает это быстрее, но еще и никогда не ошибается в расчетах.

Впрочем, в общем случае **Задача-сценарий** позволяет не только выполнять арифметические действия - это может быть любое автоматизированное действие, выполняемое ВРМС-системой без участия человека.

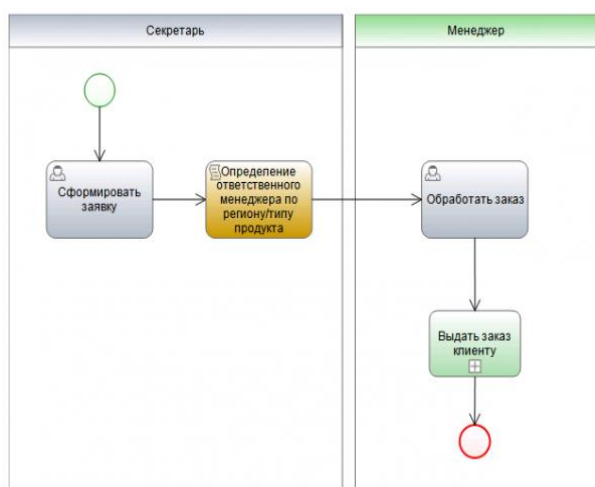


Рисунок 44

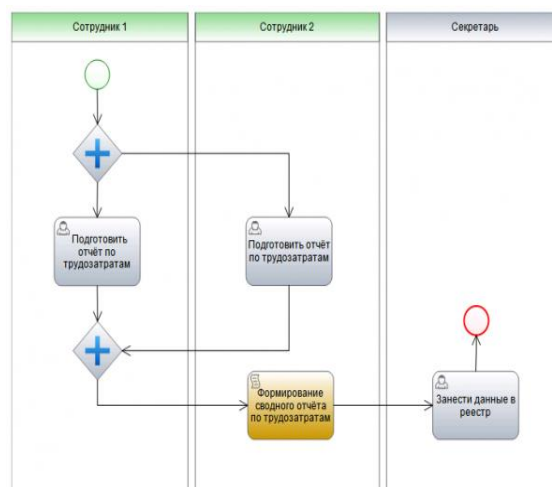


Рисунок 45

Использование **Задачи-сценария** в описании бизнес-процесса

Примечание: На рисунках 44 и 45. показано использование **Задачи-Сценария** в различных ситуациях. Рис.44 описывает процесс компании, когда клиент оставляет заказ на сайте компании или непосредственно секретарю, который формирует заявку. После этого система автоматически определяет менеджера компании, который отвечает либо за данный тип продукта, либо за регион, из которого заказчик сделал заказ. Затем система формирует задачу определённому менеджеру и далее процесс обработки и выдачи заказа идёт по разработанному маршруту.

Рис.45 отображает процесс сбора информации по отработанным часам сотрудниками компании для дальнейшего расчёта заработной платы. Сотрудникам компании параллельно приходят задачи на подготовку отчёта по трюдозатратам. После выполнения всеми сотрудниками задач, система автоматически производит подсчёт и формирует единый сводный отчёт по всем сотрудникам. Этот отчёт направляется секретарю для внесения в реестр с последующим расчётом оплаты труда по отработанным часам.

В приведённых процессах-примерах (Рис.39. и Рис.44) использован новый элемент ВРМН – **Подпроцесс**, который будет рассмотрен в рамках следующих Уроков.

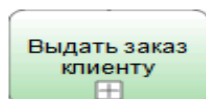


Рисунок 46 – Задача – Подпроцесс

Подпроцесс графически изображается в виде прямоугольника с маркером «+».

В нотации BPMN описывается несколько типов Задач, однако здесь мы привели три наиболее распространённые при описании бизнес-процессов.

Для проверки усвоенного материала, предлагается описать процесс «Предоставление отпуска сотруднику». Постарайтесь учесть все возможные условия, направления потоков операций, используя по максимуму элементы BPMN, рассмотренные в рамках двух Уроков.

Практическое использование подпроцессов в BPMN

Четвёртый Урок практического курса BPMN посвящён рассмотрению одного элемента спецификации BPMN – **Подпроцесс** и его использованию при описании бизнес-процессов.

В практике описания бизнес-процессов элемент нотации BPMN **Подпроцессы** используется в основном в двух случаях:

1. Для декомпозиции и повышения читаемости и наглядности схем (диаграмм);
2. Для описания повторяющихся действий. Единоразово описанный **Подпроцесс** может многократно вызываться (использоваться) внутри различных процессов.

Рассмотрим первый случай использования **Подпроцессов** – Декомпозиция процесса. Довольно часто при описании бизнес-процессов компании для наглядности используют схемы (диаграммы), отражающие верхние уровни организации работы. В этом случае диаграмма отображает «суть» процессов и нацелена на понимание логики процесса без знания деталей. Примером такого бизнес-процесса верхнего уровня может служить – процесс «Найм персонала». На верхнем уровне этот процесс будет выглядеть следующим образом:

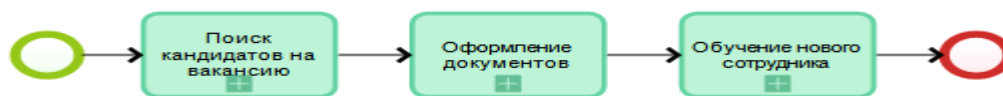


Рисунок 47 – Процесс верхнего уровня «Найм персонала»

Такая «прорисовка» процесса легка для восприятия любого бизнес-пользователя, т.к. отображает только последовательность основных действий в рамках процесса без утяжеления информацией. Любая схема (диаграмма) процесса представляет собой последовательность функциональных блоков, декомпозиция которых позволяет создать процесс верхнего уровня. При этом каждый **Подпроцесс** описывается уже на более низком уровне с полной детализацией

элементов BPMN (активностей, условий и исполнителей). **Подпроцессы** являются комплексными задачами в рамках основного процесса. Однако стоит отметить, что **Подпроцессы**, как элемент BPMN, являются не самостоятельными задачами, а лишь отсылкой к другому процессу.

Наиболее часто встречается тип **Подпроцессов** – Свёрнутый, т.е. процесс со скрытыми деталями, который позволяет облегчить визуализацию бизнес-процессов.

Свёрнутый **Подпроцесс** графически изображается в виде прямоугольника с маркером «+».

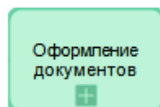


Рисунок 48 – Графическое изображение Задачи – Свёрнутый Подпроцесс

Декомпозиция процесса (разбивка на подпроцессы) позволяет моделировать и вносить изменения в рамках каждого **Подпроцесса**, не изменяя весь основной процесс целиком.

При детализации каждого отдельного **Подпроцесса** описываются необходимые условия выполнения: участники, активности, бизнес-правила и т.д. Такой процесс описывается в рамках одной оркестровки, что позволяет облегчить чтение и внесение изменений в процесс.

При детализации **Подпроцессов** приведённого примера процесса «Найм персонала» получим следующие процессы:

Поиск кандидатов на вакансию.

Оформление документов нового сотрудника.

Обучение нового сотрудника.

Рассмотрим каждый **Подпроцесс** отдельно.

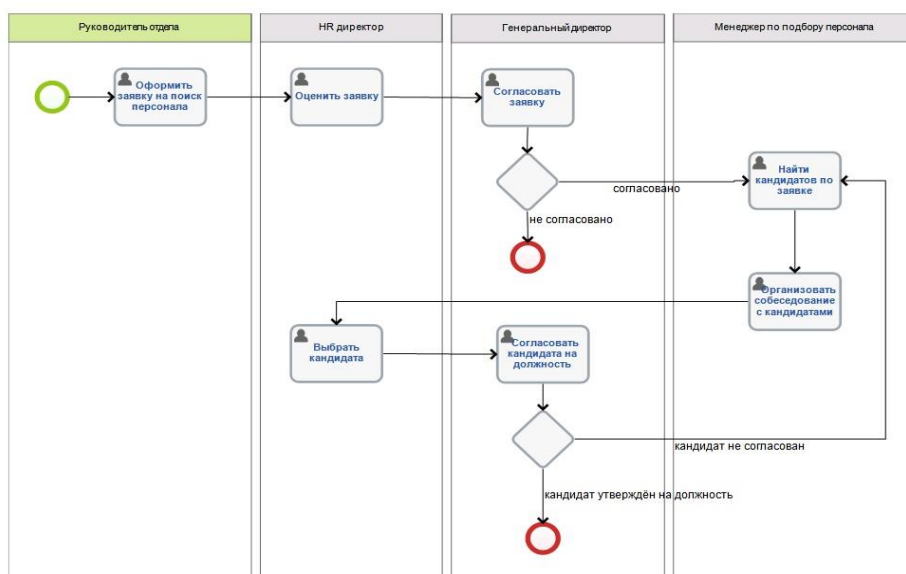


Рисунок 49 – Подпроцесс «Поиск кандидатов на вакансию»

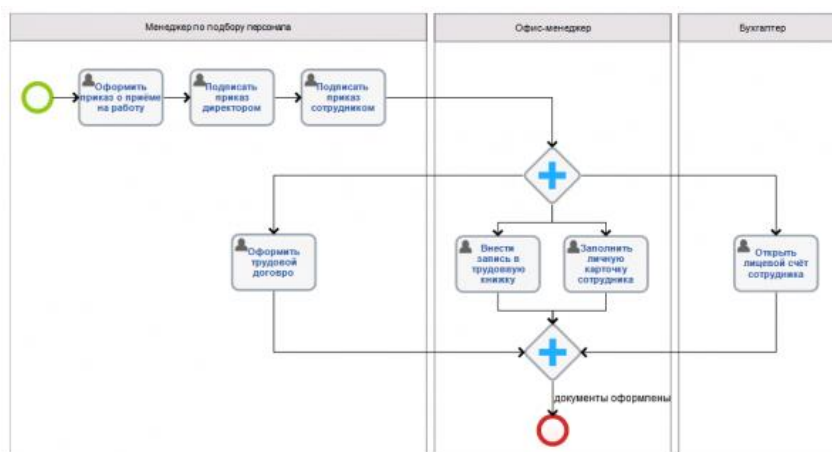


Рисунок 50 – Подпроцесс «Оформление документов»

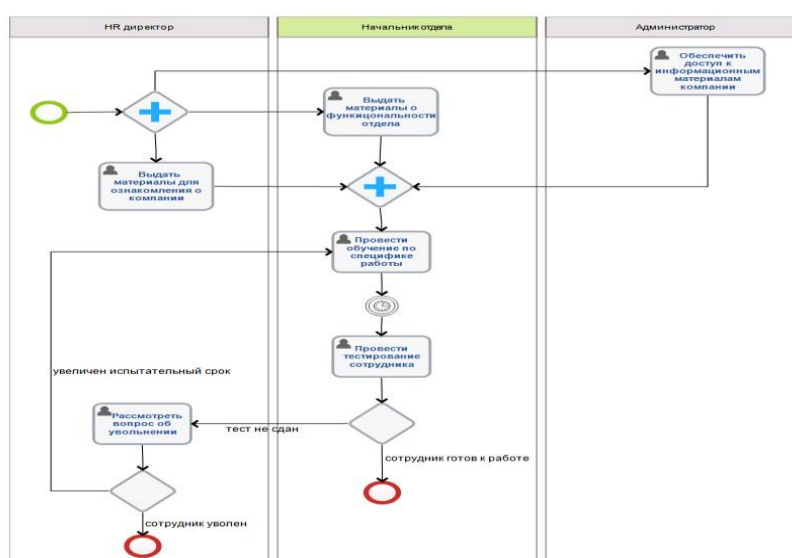


Рисунок 51 – Подпроцесс «Обучение нового сотрудника»

Вот таким образом можно описать довольно большой бизнес-процесс компании. А теперь представьте, если все активности и исполнители процесса «Найм персонала» будут отображены в рамках одной оркестровки. Сделать это сложно, а «читать» процесс будет ещё сложнее. Поэтому, используя декомпозицию (разбивку на подпроцессы) при описании сложных, но важных процессов компании, вы получаете продукт (процесс), который будет понятен любому бизнес-пользователю и легко изменяемый при моделировании и совершенствовании в будущем.

Ещё одним большим плюсом при использовании Подпроцессов является возможность их повторного использования. В рамках одного основного процесса могут повторяться одни и те же действия. Подпроцессы позволяют ссылаться на один и тот же Подпроцесс (функциональный блок) сколько угодно раз в одном бизнес-процессе и в абсолютно разных по сути процессах.

При внесении изменений в Подпроцесс нет необходимости перерисовывать все процессы, ссылающиеся на данный Подпроцесс. Сопровождением и актуализацией Подпроцесса занимаемся только его владелец, что позволяет

сократить время внесения изменений в процессы, снизить риск ошибок и иметь постоянно актуальные решения. **Изменения вносятся в одном месте и один раз!**

Повторно-используемый **Подпроцесс** используется для вызова предопределенного **Подпроцесса**. Примером повторно-используемого **Подпроцесса** может служить процесс «Информирования контрагентов» в рамках основных процессов «Выпуск нового продукта» и «Открытие нового филиала».



Рисунок 52



Рисунок 53

Пример использования
повторно-используемого
Подпроцесса



Рисунок 54 – Повторно-используемый Подпроцесс

Примечание: При появлении нового информационного канала или механизма рекламирования в Подпроцессе «Информирование контрагентов» изменения вносятся один раз и только в данный Подпроцесс, не затрагивая основные процессы компании.

В нотации BPMN рассматривается ещё один способ отображения **Подпроцесса** – Развёрнутый **Подпроцесс**.

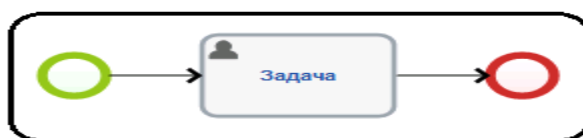


Рисунок 55 – Графически элемент Развёрнутый Подпроцесс

Развернутый **Подпроцесс** используется для более компактного отображения группы действий с использованием минимума деталей.

В BPMN также описаны различные типы **Подпроцессов**. Один из них мы описали – это повторно-используемый **Подпроцесс**.

Менее распространены в практике BPMN – **Подпроцессы Ad-Нос** (Спонтанный), **Событийный Подпроцесс**, **Транзакция**.

Событийным Подпроцессом называется специфический **Подпроцесс**, используемый внутри Процесса (Подпроцесса). Отличие такого Подпроцесса от стандартного состоит в том, что стандартный **Подпроцесс** в качестве триггера использует Поток операций, а **Событийный Подпроцесс** - Стартовое событие. Всякий раз, когда какое-то Стартовое событие запускается во время выполнения родительского Процесса, запускается и **Событийный Подпроцесс**.

Событийный Подпроцесс изображается в виде прямоугольника с закругленными углами, выполненный тонкой пунктирной линией.

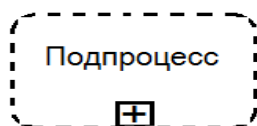


Рисунок 56 – Графический элемент Событийный Подпроцесс (Свёрнутый)

Транзакцией называется специфический тип **Подпроцесса**, который демонстрирует определенное поведение, контролируемое посредством протокола транзакции. Граница графического элемента **Транзакция** выполнена двойной линией.

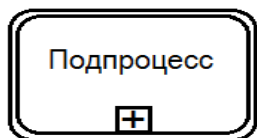


Рисунок 57 – Графический элемент Транзакция (Свёрнутый Подпроцесс)

Спонтанным Подпроцессом называется особый тип **Подпроцесса**, представляющий собой группу действий, взаимоотношения между которыми не поддаются строго регламентированным правилам. Для Процесса определяется набор Действий, однако, их последовательность и количество выполнений определяются исполнителями этих действий.

Графический элемент **Спонтанный Подпроцесс** содержит маркер, выполненный в виде знака тильды и располагающийся в центре нижней части фигуры **Подпроцесса**.



Рисунок 58 – Графический элемент Спонтанный Подпроцесс.

Средства оповещения в BPMN

Пятый Урок практического курса BPMN посвящён рассмотрению графических элементов спецификации BPMN и их использованию при описании биз-

нес-процессов: **Сообщения, Потоки Сообщений, Отправка и Получение сообщений.**

Основным типом бизнес-коммуникаций являются **Сообщения**, передаваемые и получаемые от одного участника другому в рамках процесса. **Сообщение** представляет собой содержимое (информацию) диалога между двумя участниками.

Сам же диалог (процесс обмена информацией) между двумя участниками отображается в виде **Потока сообщений**. **Поток сообщений** используется для отображения того порядка, в котором происходит обмен сообщениями (информацией, данными) между двумя заинтересованными сторонами, готовыми как отсылать, так и получать эти сообщения. В спецификации BPMN заинтересованные стороны представлены пулами (бизнес-объекты или бизнес-роли).

Графический элемент **Потока сообщений** представляет собой стрелку со свободным концом, выполненной в виде пунктирной чёрной линии (пунктир позволяет легко отличить **Поток сообщений** от **Потока операций**, выполненного стрелкой в виде непрерывной чёрной линии).



Рисунок 59 – Графическое представление Потока сообщений

Поток сообщений может соединять только два разных пула (участников процесса) между собой или элементы, расположенные внутри этих пулов. Однако, **Поток сообщений** не может соединять два элемента, расположенные внутри одного и того же пула.

Например, в процессе «Приём заказа» взаимодействие (обмен информацией) между Клиентом и Менеджером компании может отображаться следующей схемой:

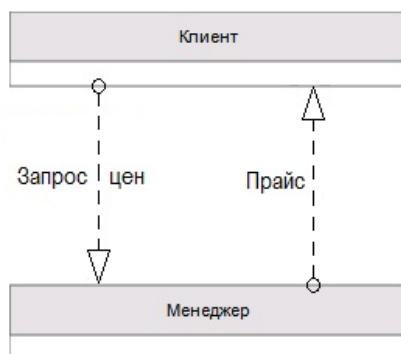


Рисунок 60 – Поток сообщений между участниками процесса

Стрелка используется для отображения межпроцессного взаимодействия - для связи элементов потока со свернутыми пулами. При необходимости **Поток** может быть именованным. **Поток сообщений** не отображает ход выполнения процесса, а показывает передачу сообщений или объектов из одного процесса в другой процесс или внешнюю ссылку.

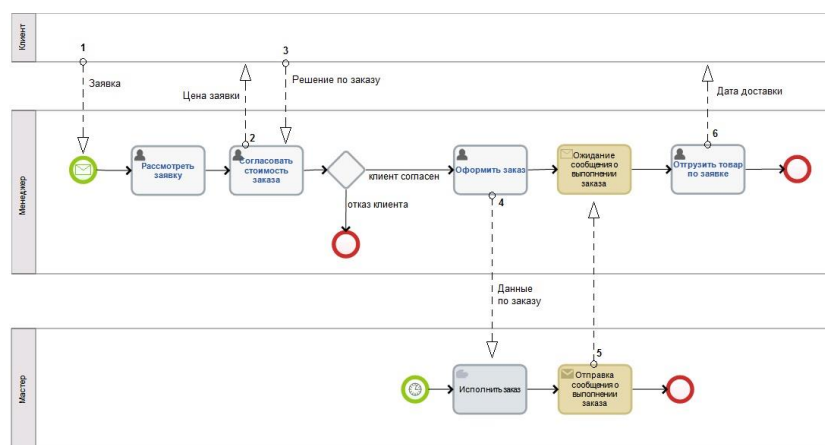


Рисунок 61 – Примеры использования Потока Сообщений

На рис. 61 представлены примеры использования потоков сообщений:

1. **Поток сообщений** представляет механизм запуска процесса: **Поток сообщений 1** выходит из внешнего процесса (или внешней ссылки) и входит в стартовое Событие рассматриваемого процесса.

Примечание: Когда Клиент оставляет запрос на сайте компании по интересующему продукту (или отправляет письмо по почте) стартует процесс «Обработка заказа» - отображается в виде Стартового события с маркером Сообщения (более подробно с данным элементом нотации можно ознакомиться в рамках Практического курса BPMN Урока 2).

2. **Поток сообщений** используется для передачи сообщений или объектов от одного действия рассматриваемого процесса во внешний процесс (или внешнюю ссылку): **Поток сообщений 2 (б)** выходит из задачи менеджера «Согласовать стоимость заявки» и входит во внешний процесс (или внешнюю ссылку);

Примечание: После анализа заявки Менеджер компании отправляет цену заявки Клиенту на согласование. Также после подготовки товара к отправке менеджер отправляет сообщение Клиенту о готовности заказа и дату доставки. Как и в первом примере, внешней ссылкой выступает пул – Клиент.

3. **Поток сообщений** используется для передачи сообщений или объектов из внешнего процесса (или внешней ссылки) в одно из действий рассматриваемого процесса: **Поток сообщений 3** выходит из внешней ссылки (пул - Клиент) и входит в задачу менеджера «Согласовать стоимость заказа».

Примечание: Клиент рассматривает стоимость заявки и, соглашаясь или отказываясь от заказа, отправляет сообщение о своём решении Менеджеру.

4. **Поток сообщений** используется для передачи сообщений или объектов от одного из действий внешнего процесса в одно действие рассматриваемого процесса (соединение элементов разных процессов): **Потоки сообщений 4 и 5** показывают обмен сообщениями между задачами разных процессов «Обработки заказа» и «Исполнения заказа».

Варианты 1, 2 и 3 использования **Потока сообщений** с участием Клиента применяется только в рамках аналитических диаграмм. Такие схемы не являются исполняемыми, т.к. Клиент не является участником BPMN систем.

Поток Сообщений может быть расширен с целью показать **Сообщение**, поступающее от одного участника к другому:

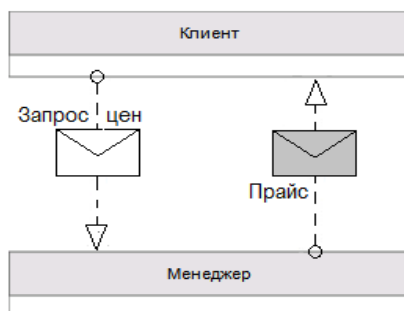


Рисунок 62 – Поток Сообщений между участниками процесса с присоединённым сообщением

Если **Сообщение** отражает содержание диалога (взаимодействия) между участниками процесса, то непосредственно сам диалог осуществляется через действия (задачи) – **Отправку и Получение сообщений**.

Получение сообщений представляет собой простую Задачу, суть которой заключается в получении сообщения, которое должно поступить от внешнего участника процесса (имеющего отношение к данному бизнес-процессу). Задача считается выполненной в случае, если сообщение было получено хотя бы один раз.

Графически **Получение сообщений** отображается в виде прямоугольника с закругленными углами (установленное в BPMN отображение графического элемента Задача) и отличается от других типов Задач наличием маркера в виде конвертика без заливки

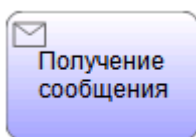


Рисунок 63 – Графический элемент Получение Сообщения

Отправка сообщений представляет собой Задачу, суть которой заключается в отправке сообщения внешнему участнику процесса (имеющему отношение к данному бизнес-процессу). Задача считается выполненной в случае, если сообщение было отправлено хотя бы один раз.

Графически **Отправка сообщений** отображается в виде прямоугольника с закругленными углами (установленное в BPMN отображение графического элемента Задача) и отличается от других типов Задач наличием маркера в виде конвертика с темной заливкой:

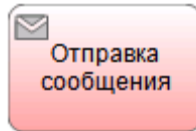


Рисунок 64 – Графический элемент Отправка Сообщения

Для проверки усвоенного материала Уроков 4 и 5, предлагается описать процессы одного из самых распространённых кадровых процессов компании – «Командировка сотрудника». Постарайтесь учесть все варианты протекания процесса, используя по возможности элементы BPMN, рассмотренные в рамках двух Уроков (Подпроцессы и Получение/Отправка сообщений).

Использование артефактов и данных в BPMN

Шестой Урок практического курса BPMN посвящён рассмотрению графических элементов спецификации BPMN и их использованию при описании бизнес-процессов: **Артефакты, Данные и Ассоциации**.

Язык BPMN позволяет разработчикам моделей указывать дополнительную информацию о процессе, не связанную непосредственно с потоками операций или потоками сообщений данного процесса. BPMN предлагает использование элементов нотации – **Артефакты, Данные**, и соединяющие элементы – **Ассоциации**. Рассматриваемые в Уроке элементы нотации не являются исполнительными и служат для облегчения читаемости и анализа моделируемых бизнес-процессов.

Артефакты используются для введения дополнительной информации по процессу. Существует два стандартных артефакта: **Группа и Текстовая аннотация** (в версии 1.2 нотации BPMN Данные входили в **Артефакты**, в BPMN 2.0 – уже выделены в отдельную категорию элементов). Однако разработчики BPMS систем или инструменты моделирования могут добавить столько Артефактов, сколько требуется.

Рассмотрим использование **Группы** как элемента моделирования процессов. Графическое изображение элемента **Группа** изображается прямоугольником с закругленными углами, граница которого – штриховая линия с точками. **Группа** позволяет объединять различные действия, но не влияет на поток управления в диаграмме.



Рисунок 65 – Графическое изображение Группы

Группа предназначена для группировки графических элементов, принадлежащих одной и той же категории. Такая группировка не оказывает влияния на поток операций. На диаграмме бизнес-процесса название категории, к кото-

рой принадлежат сгруппированные элементы, отображается в качестве названия **Группы**. Такого рода группировка может использоваться в целях составления документации или при проведении анализа.

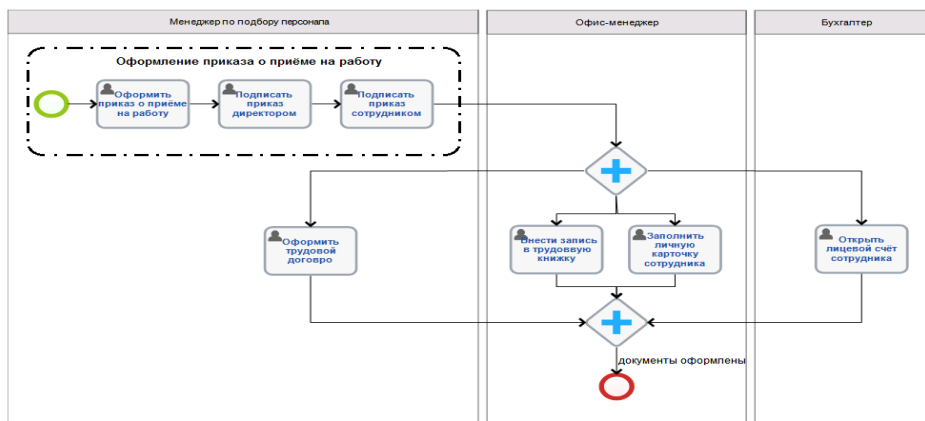


Рисунок 66 – Использование Группы в рамках процесса

Примечание: в приведённом процессе «Оформление документов на нового сотрудника» (Урок 4 Практических курсов BPMN) в Группу соединены задачи по подготовке и подписанию приказа о приёме на работу. Как видно, такое выделение области диаграммы (группировка элементов) носит только смысловую нагрузку (показывая логическую взаимосвязь, не изменяя при этом ход процесса).

Группа не является действием (задачей, подпроцессом) или одним из элементов потока (событием, шлюзом), поэтому данный графический элемент не может быть соединен с потоком операций или с потоком сообщений. Ограничения использования пулов и дорожек не распространяются на использование **Групп**. Это означает, что для объединения элементов диаграммы **Группа** может простирается за границы пула. В таком качестве **Группа** используется для отображения действий, являющихся частью масштабных взаимоотношений типа В2В. Текстовая аннотация – второй стандартный **Артефакт** нотации BPMN - представляет собой механизм, при помощи которого разработчик модели может добавлять на диаграмму дополнительную информацию, являющуюся важной для конечного пользователя диаграммы. **Текстовые аннотации** используются для уточнения значения элементов диаграммы (добавления комментариев, пояснений и другой текстовой информации) и повышения её информативности и лёгкость для понимания любым бизнес-пользователем.

Графический элемент **Текстовая аннотация** представляет собой негерметичный прямоугольник, выполненный одинарной линией.

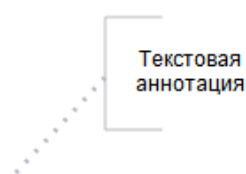


Рисунок 67 – Графический элемент Текстовая аннотация

Текстовая Аннотация может быть присоединена к определенному элементу на диаграмме при помощи **Ассоциации**, однако, он не оказывает влияния на ход Процесса. Текст, ассоциированный с **Текстовой аннотацией**, располагается в пределах данного графического элемента.

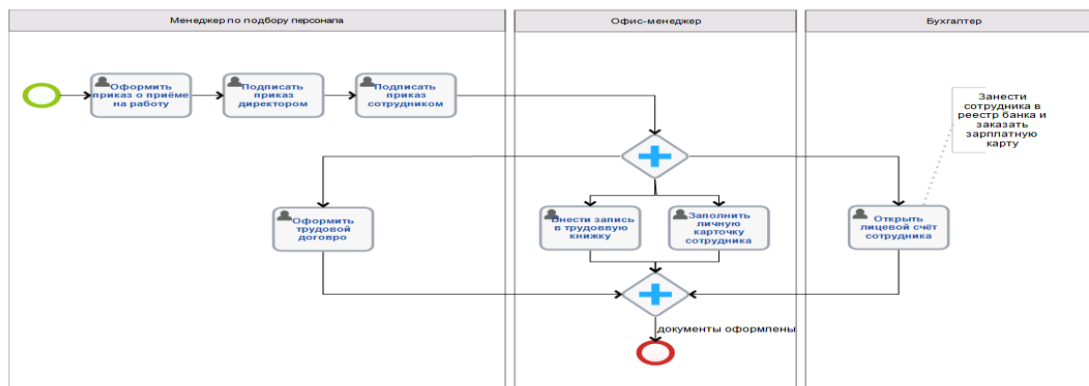


Рисунок 68 – Использование Текстовой аннотации в описании процесса

Примечание: в процессе «Оформление документов на нового сотрудника» Текстовая аннотация позволяет уточнить действия бухгалтера при выполнении задачи «Открыть лицевой счёт сотрудника».

Другой элемент нотации BPMN (соединяющий) – **Ассоциация** - используется для установки соответствия между какой-либо информацией и **Артефактом** и элементами потока (события, действия, шлюзы). Текстовые объекты, а также графические объекты, не относящиеся к элементам потока, могут соотноситься с элементами потока или потоком операции с помощью **Ассоциации** (см. рис.69).

Графическое изображение **Ассоциации** представляет собой пунктирную линию.



Рисунок 69 – Графический элемент Ассоциация

При необходимости **Ассоциация** может указывать направление потока (например, потока Данных). Тогда графический элемент **Ассоциации** отображается со стрелкой.



Рисунок 70 – Графический элемент направленная Ассоциация

В практике описания бизнес-процессов **Ассоциация** используется для соединения указанного пользователем текста (**Текстовой аннотации**), **Данных** (Объектов данных, Хранилище данных – см. далее) с элементами потока.

Традиционным требованием к моделированию процессов является возможность моделирования компонентов (физических или информационных), которые создаются, управляются и используются в ходе выполнения процесса.

Важным аспектом этого требования является возможность сбора введённых данных, а также запроса этих данных и управления ими.

VRMN выделяет несколько элементов, предназначенных для хранения и передачи компонентов в ходе выполнения процесса: **Объекты данных** и **Хранилище данных**. Обычно такие элементы относят к «связанным с компонентами».

Графическое представление элемента **Объект данных** имеет вид листа документа с загнутым углом.



Рисунок 71 – Графическое изображение Объекта данных

Объект данных представляет собой информацию, которая обрабатывается в ходе процесса. Они не влияют непосредственно на последовательный поток или поток сообщений процесса, но обеспечивают информацию о том, какие действия требуют выполнения и/или что они производят. **Объект данных** привязан к контексту процесса: он изображается внутри процесса или подпроцесса. **Объект данных** процесса существует только в интервале времени от момента запуска данного экземпляра процесса до его завершения. При отмене выполнения данного экземпляра процесса все находящиеся в нём экземпляры Объектов данных становятся неактивны. Соответственно, при завершении или отмене выполнения экземпляра процесса, доступ к Объектам данных этого экземпляра процесса из другого внешнего процесса невозможен.

В нотации VRMN 2.0 (в отличие от предыдущей версии нотации) вводится новое понятие **Хранилище данных** для моделирования постоянной памяти. Данный объект используется процессом для записи и извлечения данных как, например, базы данных. Сохраненная информация будет действительна даже после завершения выполнения экземпляра процесса.

Графический элемент **Хранилища данных** изображается следующим образом:



Рисунок 72 – Графический элемент Хранилища данных

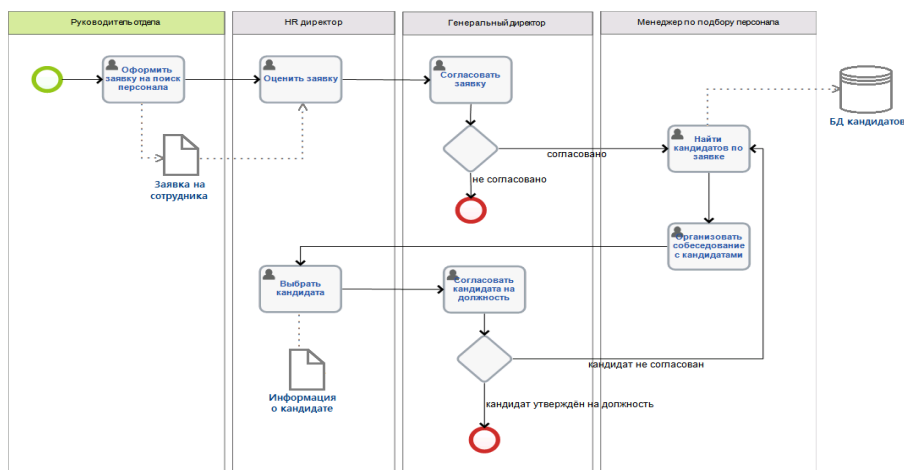


Рисунок 73 – Пример использования Ассоциаций в моделировании процессов

Примечание: на рис.74. рассмотрен пример процесса «Поиск кандидатов на вакансию» с использованием элементов нотации Данные и Ассоциации. Данные в рамках процесса показывают, какая информация является результатом исполнения задач (заявка на сотрудника, информация о кандидате) или используется при выполнении задачи (заявка на сотрудника, База данных кандидатов). Заявка на сотрудника и информация о кандидате являются Объектами данных, База данных (БД) кандидатов – Хранилищем данных.



Рисунок 74 – Межпроцессное взаимодействие через данные

В нотации BPMN элемент Хранилище данных используется для моделирования межпроцессного взаимодействия через данные, что невозможно исполнить с помощью Объектов данных, которые применяются только в рамках одной оркестровки (процесса/подпроцесса).

Список использованных источников

1. Информационные системы и технологии в экономике / Т.П. Барановская, В.И. Лойко, М.И. Семенов, А.И. Трубилин. М.: Финансы и статистика, 2009.
2. Вендров А.М. Проектирование программного обеспечения экономических информационных систем: учебник. М.: Финансы и статистика, 2011.
3. Гайдамакин Н.А. Автоматизированные информационные системы, базы и банки данных. М.: Гелиос АРВ, 2008.
4. Грекул В.И. Проектирование информационных систем. М.: Интернет-Университет Информационных Технологий, 2008.
5. Дубейковский В.И. Практика функционального моделирования с APFusion Process Modeler. М.: ДИАЛОГ МИФИ, 2011.
6. Ильина О.П. Информационные технологии бухгалтерского учета. Питер, 2002.
7. Калянов Г.Н. CASE структурный системный анализ (автоматизация и применение). М.: ЛОРИ, 1996. 242с.
8. Казин Ф.А., Тойвонена Н.Р. Проектный менеджмент в вузе. Учебные кейсы / под ред. Ф.А. Казина [Электронный ресурс]. СПб.: НИУ ИТМО, 2012. 182 с. – Режим доступа: <http://window.edu.ru/resource/221/7822> СПб.: НИУ ИТМО, 2012.
9. Информатизация бизнеса / А.М. Карминский, С.А. Арминский, П.В. Нестеров, Б.В. Черников. М.: «Финансы и статистика», 2010.
10. Карпова Т. Базы данных. Питер, 2012.
11. Кетков Ю.Л., Кетков А.Ю. Практика программирования: Visual Basic, C++Builder, Delphi. СПб.: БХВ Петербург, 2012.
12. Лещев Д.В. Создание интерактивного WEB-сайта: учебный курс. СПб.: Питер, 2010.
13. Маклаков С.В. Создание информационных систем с APFusion Modeling Suite. М.: ДИАЛОГМИФИ, 2010.
14. Никифоров С.В. Введение в сетевые технологии. М.: Финансы и статистика, 2011.
15. Омельченко Л.Н. Visual FoxPro 8. СПб.: БХВ Петербург, 2010.
16. Орлов С.А. Технологии разработки программного обеспечения: учебник для вузов. СПб.: Питер, 2010.
17. Петров В.Н. Информационные системы: учебник для вузов. СПб.: Питер, 2012.
18. Тельнов Ю.Ф. Проектирование экономических информационных систем: учебник; под ред. Ю.Ф. Тельнова. М.: Финансы и статистика, 2011.
19. Федоров А., Елманова Н. Базы данных. М. Компьютер пресс, 2011.
20. Чекалов А.П. Базы данных: от проектирования до разработки приложений. СПб.: БХВ-Петербург, 2010.
21. Черемных С.В., Семенов И.О., Ручкин В.С. Структурный анализ систем: IDEF-технологии. М.: Финансы и статистика, 2005.
22. Ярочкин В.И. Информационная безопасность. М.: Летописец, 2010.

Учебное издание

Федькова Н.А.

Методология и технология проектирования информационных систем

Методическое пособие

Редактор Адылина Е.С.

Подписано к печати 21.11.2022 г. Формат 60x84 ¹/₁₆.

Бумага офсетная. Усл. п. л. 3,02. Тираж 25 экз. Изд. №7418

Издательство Брянского государственного аграрного университета
243365 Брянская обл., Выгоничский район, с. Кокино, Брянский ГАУ