МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра вычислительной техники

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«6» 09 2021г.

ПРОЕКТИРОВАНИЕ ИС

Методические указания по выполнению практических работ по дисциплине «Проектирование информационных систем» для студентов направления подготовки бакалавров 09.03.01 Информатика и вычислительная техника, направленность (профиль) " Интеллектуальные системы в цифровой экономике "

Курск 2021

УДК 004.82 (075.8)
Составитель: Т.И.Лапина

Рецензент
Кандидат технических наук, доцент Е.А.Петрик

Проектирование информационных систем: методические указания по выполнению практических работ / Юго-Зап. гос. ун-т; сост.: Т. И. Лапина, Курск, 2021. 63с.: ил. 56, табл. 6, Библиогр.: с. 5.

Содержат краткие теоретические сведения о методах разработки моделей при проектированию информационных систем на основе методологии структурного анализа и проектирования SADT и объектно-ориентированной методологии с использованием нотаций языка UML. Методические указания соответствуют требованиям программ по направлениям подготовки бакалавров 09.03.01 Информатика и вычислительная техника, направленность (профиль) " Интеллектуальные системы в цифровой экономике "

Предназначены для студентов направления подготовки бакалавров 09.03.01 Информатика и вычислительная техника, направленность (профиль) " Интеллектуальные системы в цифровой экономике " дневной и заочной форм обучения.

Текст печатается в авторской редакции

Подписано в печать Формат 60х84 1/16.
Усл. печ. л. . Уч. – изд. л. .Тираж 100 экз. Заказ. Бесплатно.
Юго - Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

Практическая работа №1

Предпроектное обследования объекта автоматизации

1. Цель работы

Приобретение навыков использования методов проведения предпроектного обследования объекта автоматизации и анализа полученных материалов, формирования требований к проектируемой ИС, распределения ролей в группе разработчиков, оформления технического задания на проектирование.

2. Основные теоретические положения

Целью анализа предметной области является рассмотрение существующего состояния предметной области, характеристик объекта и системы управления, выявление и анализ проблем предметной области, наличие компьютеризированных информационных технологий, состав средств компьютерной техники и программного обеспечения, оценка их достаточности и эффективности для решения задач информатизации (автоматизации).

В данном разделе следует отразить следующие характеристики выбранной предметной области:

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ
- 1.1 Организационно-экономическая характеристика компании
- 1.2 Экономический анализ компании, выявление и анализ целей и проблем
- 1.3 Бизнес- и информационные процессы компании
- 1.4 ИТ-инфраструктура компании
- 1.5 Постановка задачи проектирования ИС

В качестве предметной области (объекта автоматизации или информатизации) может выступать:

- организация или ее подразделение (предприятие, учреждение, фирма, и т. п.),
- отдельный вид деятельности (бизнес-процесс).

Организационно-экономическая характеристика предметной области должна включать:

– наименование, организационная форма, юридический статус и миссию организации (необходимо выяснить миссию организации, оценить правильность ее формулировки и, если надо, дать свою формулировку),

– его организационную схему компании (с указанием общей численности работающих), пример организационной схемы приведен на рис.1;

– краткую характеристику технико-экономических аспектов подразделений.

Аспектами описания компании являются: основные задачи; тип производства (услуг); номенклатура готовой продукции (услуг); номенклатура материалов и ресурсов.

Необходимо установить базовые экономические и другие показатели, характеризующие деятельность организации (например, прибыль, рентабельность, число обслуживаемых клиентов, и т. п.).

Пример целей и задач компании приведен на рис.1.

Рисунок 1 – Пример целей и задач компании

Пример организационной схемы компании представлен на рис.2.

При анализе проблем предметной области следует сделать акцент на проблемах и недостатках, устранение которых предполагается осуществить в проекте, например:

– невозможность расчета показателей, необходимых для управления объектом из-за сложности вычислений или большого объема информации;
– высокая трудоемкость обработки информации (привести объемно-временные параметры);
– низкая оперативность, снижающая качество управления объектом;
– невысокая достоверность результатов решения задачи из-за дублирования потоков информации;
– несовершенство процессов сбора, передачи, обработки, хранения, защиты целостности и секретности информации и процессов выдачи результатов расчетов конечному пользователю и т. д.

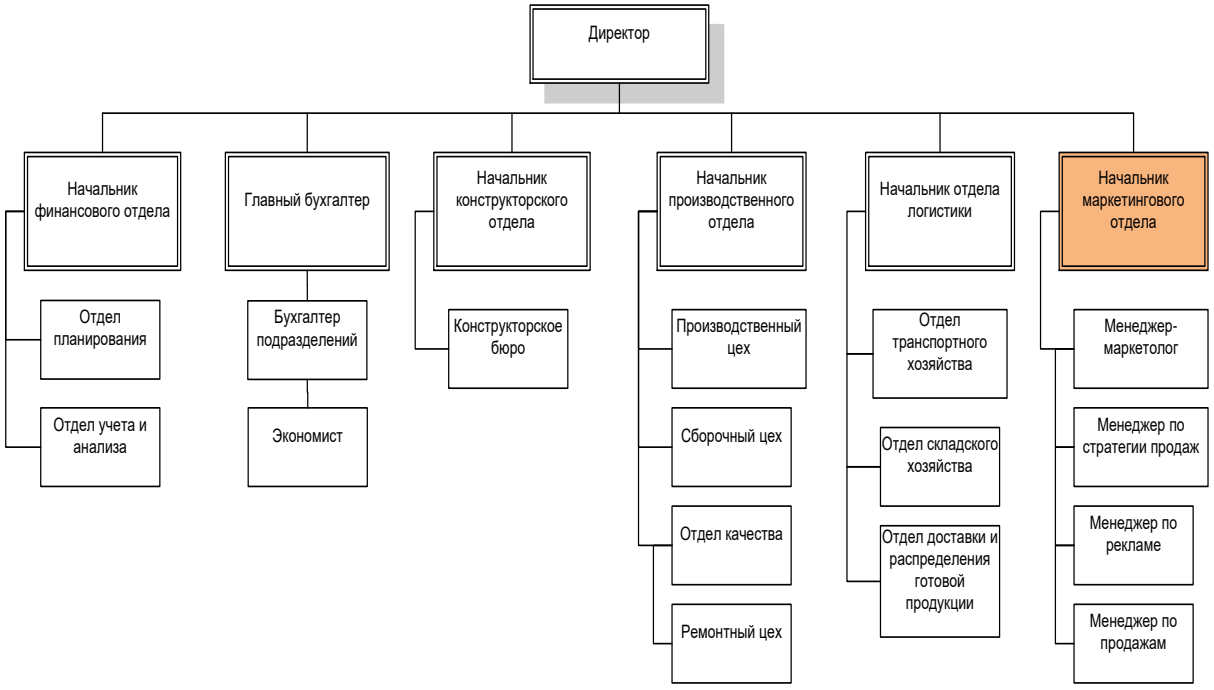


Рисунок 2 – Пример организационной схемы

В области информационного обеспечения управления рассматриваются потребности субъекта управления в экономической и другой информации для принятия управленческих решений, а также имеющиеся проблемы информационного обеспечения. Необходимо описать источники информации, способы ее хранения, передачи и переработки (используя методологию DFD). При этом следует:

- описать существующую технологию выполнения выбранной для рассмотрения функции управления (или комплекса функций), т. е. указать на следующее: особенности расчета показателей; перечни и источники используемых входных документов; методы и технические средства, применяемые для их обработки; методы защиты конфиденциальной информации (сведений, составляющих коммерческую или производственную тайну);
- выявить основные недостатки, присущие существующей практике управления и обработки экономической информации.

Описание задачи должны быть выполнено в виде единого связного текста и может сопровождаться диаграммами структурного системного анализа и обобщающими таблицами или разъясняющими схемами.

Для выполнения анализа объекта управления и решаемой задачи рекомендуется использовать методологии IDEF0, IDEF3, DFD, UML.

При описании ИТ-инфраструктура организации необходимо:

– идентифицировать существующие ИС и описать бизнес-процессы, которые они поддерживают;
– дать описание сетевой архитектуры, компьютерной техники и средств телекоммуникаций;
– описать системное и прикладное программное обеспечение;
– описать работу ИТ-подразделений и служб.

Сетевая архитектура. Сетевая архитектура представляет собой множество технических средств: сервера, клиентские устройства доступа, каналы связи. Необходимо рассмотреть, в случае наличия, существующую локальную вычислительную сеть, оборудование, структурированную кабельную сеть и ее атрибуты. Необходимо указать наличие доступа к внешним телекоммуникациям (в частности, выход в Internet), параметры подключения. Примеры сетевой архитектуры представлены на рис.3.

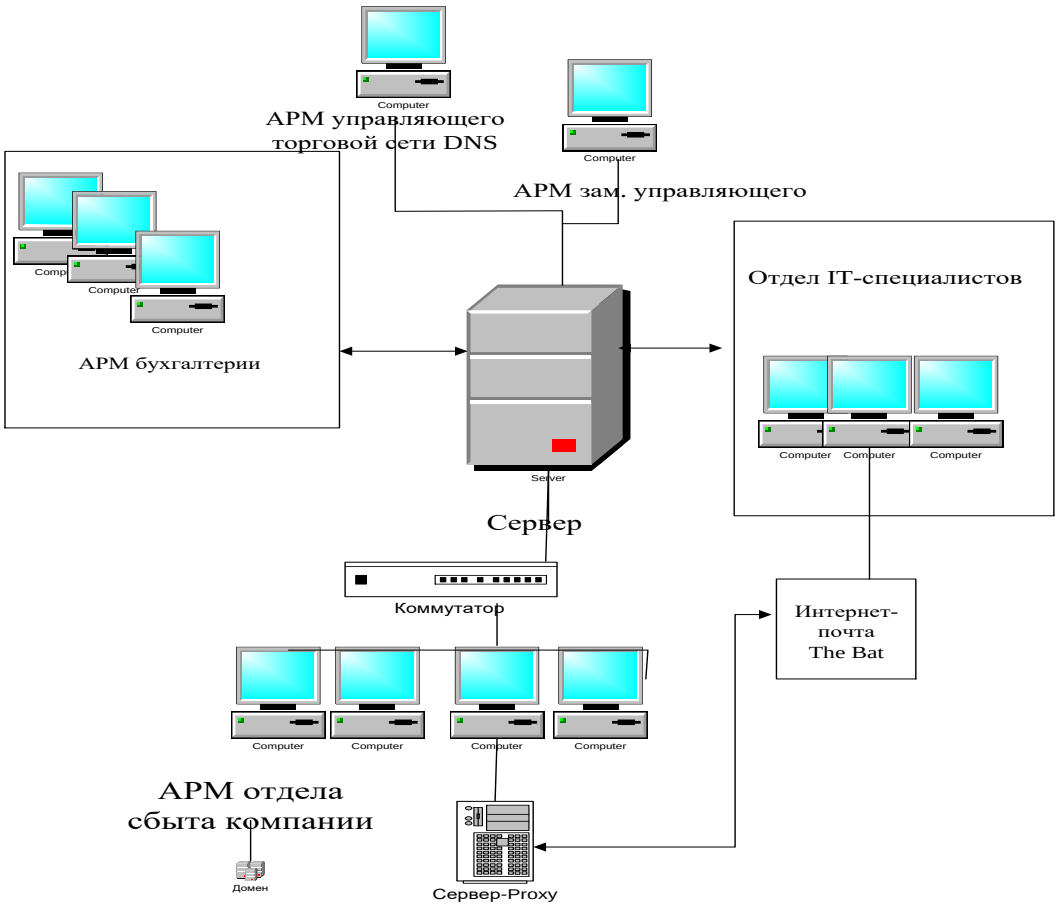


Рисунок 3 - Пример сетевой архитектуры

Компьютерные средства. Необходимо кратко описать компьютерные средства, используемые в организации (подразделении). Примером такого описания может быть с таблица 1:

Таблица 1 – Компьютерные средства, используемые в компании

Компьютерная техника	Количество
Компьютеры всего	**12**
из них: компьютеры в ЛВС	8
серверы	2
несвязанных ЛВС	1
Принтеры	**5**
из них лазерные	4

Далее следует привести обобщенные параметры компьютеров, например таблица 2:

Таблица 2 – Характеристика средств компьютерной техники, используемых компанией

Наименование	Цена, руб.	Количество, ед.	Итого, тыс.руб.
монитор ЖК 17"LG L1752 H-BF Flatron (LCD 1280x1024)	6306	10	63.060
Сист.блок Futjitsu P4-630-3.0 Ггц/DDR 512Mb-2*256Mb/80Gb	16276	10	162.760
Клавиатура Logitech Deluxe 250 Y-SAF76	276	10	2.760
Мышь Logitech Optical<M-SBF96>	296	10	2.960
Стоимость установки и обучения Итого	23154		23.154 16276

Программная архитектура. Программную архитектуру целесообразно формировать исходя из существующих программных продуктов, которые функционируют на предприятии (рис. 3).

Необходимо указать имеющиеся специализированные или офисные программы, бухгалтерские, складские и другие информационные системы. Необходимо показать, для решения каких задач они используется, в виде таблицы 3:

3. Контрольные вопросы

1. В чем заключается методика предпроектного обследования?
2. Какие существуют универсальные методы, пригодные для обследования всех функциональных звеньев предприятия?
3. Какие существуют характеристики документа?
4. Каким образом производится кодирование полученной документации?
5. На каких уровнях проводится обследование аспектов деятельности предприятий?
6. Что включает информационная база данных?
7. В каких направлениях выполняется информационный анализ предметной области?
8. Цель анализа полученной информации.

Практическая работа №2
Разработка ТЗ на проектирование ИС

1. Цель работы

Целью лабораторной работы является изучение рекомендаций стандартов на составление технического задания:

- ГОСТ 19.201-78 ЕСПД. Техническое задание. Требование к содержанию и оформлению; ГОСТ 24.201-79 Документация на АСУ. Требование к содержанию документа «Техническое задание»;
- ГОСТ 34.602-89. Информационная технология. Автоматизированные системы. Техническое задание на создание автоматизированной системы;
- IEEE 830-1998 «Методика составления спецификаций требований к программному обеспечению». Получение практических навыков в разработке и структуризации требований в пределах одной проблемной области.

2. Краткие теоретические сведения. Стандарты разработки

Техническое задание является исходным материалом для создания информационной системы или другого продукта. Поэтому техническое задание (сокращенно ТЗ) в первую очередь должно содержать основные технические требования к продукту и отвечать на вопрос, что данная система должна делать, как работать и при каких условиях.

Как правило, этапу составления технического задания предшествует проведение обследования предметной области, которое завершается созданием аналитического отчета. Именно аналитический отчет (или аналитическая записка) ложится в основу документа Техническое задание.

Если в отчете требования заказчика могут быть изложены в общем виде и проиллюстрированы UML-диаграммами, в техническом задании следует подробно описать все функциональные и пользовательские требования к системе. Чем подробнее будет составлено техническое задание, тем меньше спорных ситуаций возникнет между заказчиком и разработчиком во время приемочных испытаний.

Таким образом, техническое задание является документом, который позволяет как разработчику, так и заказчику представить конечный

продукт и впоследствии выполнить проверку на соответствие предъявленным требованиям.

Руководствующими стандартами при написании технического задания являются ГОСТ 34.602.89 «Техническое задание на создание автоматизированной системы» и ГОСТ 19.201-78 «Техническое задание. Требования к содержанию и оформлению». Первый стандарт предназначен для разработчиков автоматизированных систем, второй для программных средств (разницу между данными сериями мы обсуждали в статье «Что такое ГОСТ»).

Итак, ниже мы представляем список и описание разделов, которые должно содержать.

Таблица 1 – ГОСТы технического задания

ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению	**ГОСТ 34.602.89 Техническое задание на создание автоматизированной системы**
1. Введение	1. Общие сведения
2. Основания для разработки	
3. Назначение разработки	2. Назначение и цели создания системы
	3. Характеристика объекта автоматизации
4. Требования к программе или программному изделию	4. Требования к системе
4.1. Требования к функциональным характеристикам	4.2. Требования к функциям (задачам), выполняемым системой
	4.1. Требования к системе в целом
	4.1.1. Требования к структуре и функционированию системы
	4.1.3. Показатели назначения
4.2. Требования к надежности	4.1.4. Требования к надежности
	4. 1.5. Требования к безопасности
	4. 1.6. Требования к эргономике и технической эстетике
4.3. Условия эксплуатации	4.1.2. Требования к численности и квалификации персонала системы и режиму его работы
	4. 1.9. Требования к защите информации от несанкционированного доступа

	4. 1.10. Требования по сохранности информации при авариях
	4. 1.11. Требования к защите от влияния внешних воздействий
	4. 1.12. Требования к патентной чистоте
	4. 1.13. Требования по стандартизации и унификации
4.4. Требования к составу и параметрам технических средств	4. 1.8. Требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы
4.5. Требования к информационной и программной совместимости	
4.6. Требования к маркировке и упаковке	
4.7. Требования к транспортированию и хранению	4. 1.7. Требования к транспортабельности для подвижных систем
4.8. Специальные требования	4. 1.14. Дополнительные требования
	4.3. Требования к видам обеспечения
5. Требования к программной документации	8. Требования к документированию
6. Технико-экономические показатели	
7. Стадии и этапы разработки	5. Состав и содержание работ по созданию системы
8. Порядок контроля и приемки	6. Порядок контроля и приемки системы
	7. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие
	9.Источники разработки

Опираясь на таблицу, приведенную выше, можно выделить основные разделы технического задания:

- Общие сведения о системе (программе);
- Назначение, цели и задачи системы (программы);
- Требования к системе (функциональные требования, пользовательские требования, требования к системе в целом и тд);
- Требования к видам обеспечения;
- Требования к документированию;
- Стадии и этапы разработки;

– Порядок контроля и приемки системы (программы).

2.1 Общие сведения

Данный раздел документа Техническое задание должен содержать полное наименование системы и все варианты сокращений, которые будут использованы при разработке документации.

Пример:

«В данном документе создаваемая информационная система называется «Единое окно доступа к образовательным ресурсам», сокращенно ЕО. Систему Единое окно доступа к образовательным ресурсам далее в настоящем документе допускается именовать Единое окно или Система.»

Также сюда следует включить подразделы сообщающие реквизиты организаций участвующих в разработке (Заказчика и Исполнителя).

В подразделе «Основания для разработки» документа Техническое задание перечисляются основные документы, на основании которых выполняются данные работы. Например, для системы, выполняемой по заказу Правительства страны или другого Государственного органа, должны быть указаны законы, указы и постановления Правительства.

Далее следует указать сроки начала и окончания работ и сведения об источнике финансирования. Данная информация может быть указана и в конце технического задания в разделе с указанием стадий и этапов работ.

Неотъемлемой частью документа Техническое задание также должен быть список терминов и сокращений. Термины и сокращения лучше представить в виде таблицы с двумя столбцами «Термин» и «Полная форма».

Термины и сокращения располагаются в алфавитном порядке. В первую очередь принято давать расшифровку русскоязычным терминам и сокращениям, потом англоязычным.

3. Контрольные вопросы

1. Перечислите стандарты на разработку технического задания на проектирование информационной системы?
2. Из каких частей состоит описание технического задания?
3. Что входит в стандарт спецификаций?

Правктичекая работа №3
ПОСТРОЕНИЕ МОДЕЛИ БИЗНЕС-ПРОЦЕССОВ. МЕТОДОЛОГИЯ IDEF0

1. Цель работы

Приобретение навыков использования методологии IDEF0 при разработке бизнес-модели компании, освоение приемов построения бизнес-модели предметной области в редакторе MicrosoftVisio/

2. Основные теоретические положения

В ходе реализации программы интегрированной компьютеризации производства была разработана методология IDEF0 (Integrated Definition Function Modeling).

Методология IDEF0 представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели объекта какой-либо предметной области. Функциональная модель IDEF0 отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями.

Модель в IDEF0 представлена совокупностью иерархически упорядоченных и логически связанных диаграмм, а также текста документации и словарей, связанных друг с другом с помощью перекрестных ссылок.

Основу методологии IDEF0 составляет графический язык описания бизнес-процессов (функциональные блоки и интерфейсные дуги) (рисунок 1).

1. Функциональный блок

Функциональные блоки обозначают поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты.

Графически функциональные блоки изображаются в виде прямоугольников. Имя функционального блока должно быть выражено сочетанием отглагольного существительного,

обозначающего процесс, или глаголом, например, Учет заказов клиентов, Изготовление продукции, и.т.д. (рис. 1):

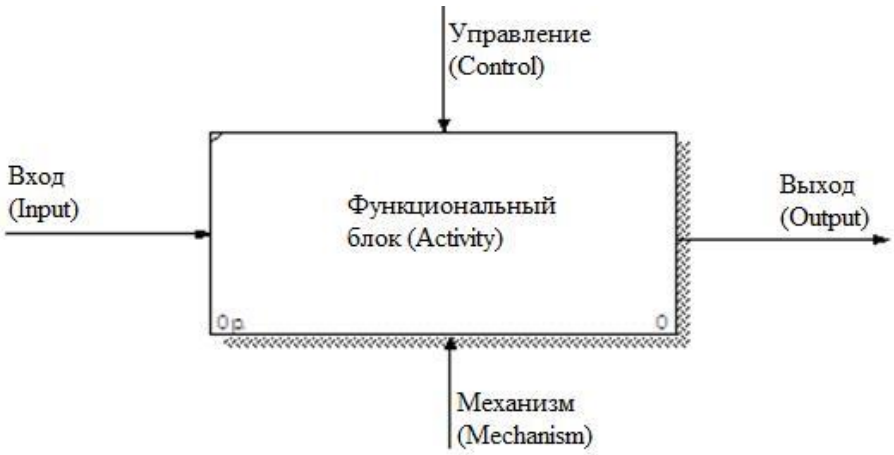


Рисунок 1– Функциональный блок

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

2. Интерфейсная дуга (стрелка – Arrow)

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

В IDEF0 различают 4 типа стрелок (рис.1). Каждая стрелка имеет свое расположение относительно функционального блока.

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

Управление (Control) – правила, стратегии, процедуры или стандарты, которыми руководствуется работа.

Каждая работа должна иметь хотя бы одну стрелку управления. Рисуется как входящая в верхнюю грань работы.

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

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

Модель может содержать 4 типа диаграмм:

- контекстную диаграмму (в каждой модели может быть только 1 контекстная диаграмма);
- диаграмму декомпозиции;
- диаграмму дерева узлов.

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

После описания системы в целом проводится разбиение ее на фрагменты. Этот процесс называется функциональной декомпозицией, а диаграммы, которые описывают каждый фрагмент и взаимодействие фрагментов – диаграммами декомпозиции.

После декомпозиции контекстной диаграммы проводится декомпозиция каждого фрагмента системы на более мелкие и т.д., до достижения требуемого уровня подробности описания.

Диаграмма дерева узлов показывает иерархическую зависимость диаграмм декомпозиции.

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

4. Порядок выполнения работы в MS VISIO

Для построения функциональной модели бизнес-процесса, используя MS Visio, необходимо запустить программу.

В открывшейся программе выбрать: Файл – Фигуры – Блок-схема – Фигуры схемы IDEF0.

1. Запустите MS Visio.
2. На закладке выбора шаблона выберите категорию *Блок-схема* и в ней элемент *Схема IDEF0*. Нажмите кнопку *Создать* в правой части экрана.
3. Окно программы примет вид, подобный рис. 2.

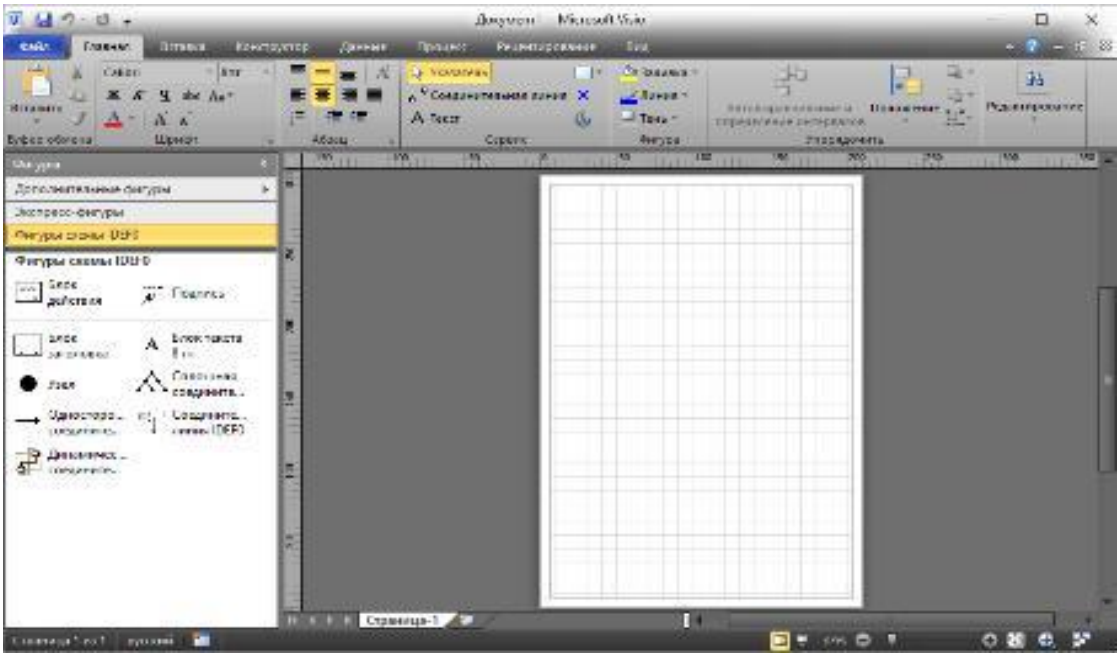


Рисунок 2 – Окно программы с выбранным шаблоном

4. Создание мастерской страницы.

4.1. Для удобства переведите страницу в альбомный вид: Конструктор – Параметры страницы – Ориентация – Альбомная;
4.2. Перетащите Блок заголовка на пустую страницу, удерживая нажатой правую кнопку мыши;
4.3. Заполнить поле «Заголовок» (рисунок 3), предложенное в открывшемся окне: внести номер контекстной диаграммы и имя рассматриваемого процесса, в данном случае: А-0 деятельность отдела по работе с заказами.

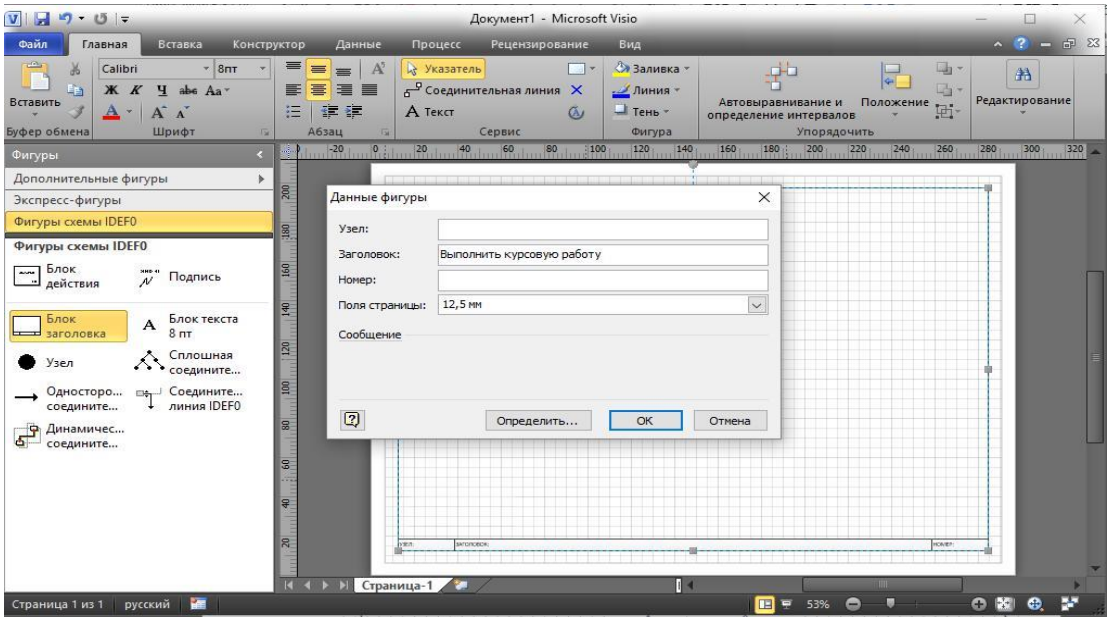


Рисунок 3 – Мастерская страница

4.4. Далее, имя заголовка фигуры «*Блок заголовка*» должно соответствовать номеру и названию задачи., В область диаграммы (поле *Блока заголовка*) внесите *Блок действия*. В открывшемся окне «Данные фигуры» внесите имя процесса и идентификатор процесса (рис. 4).

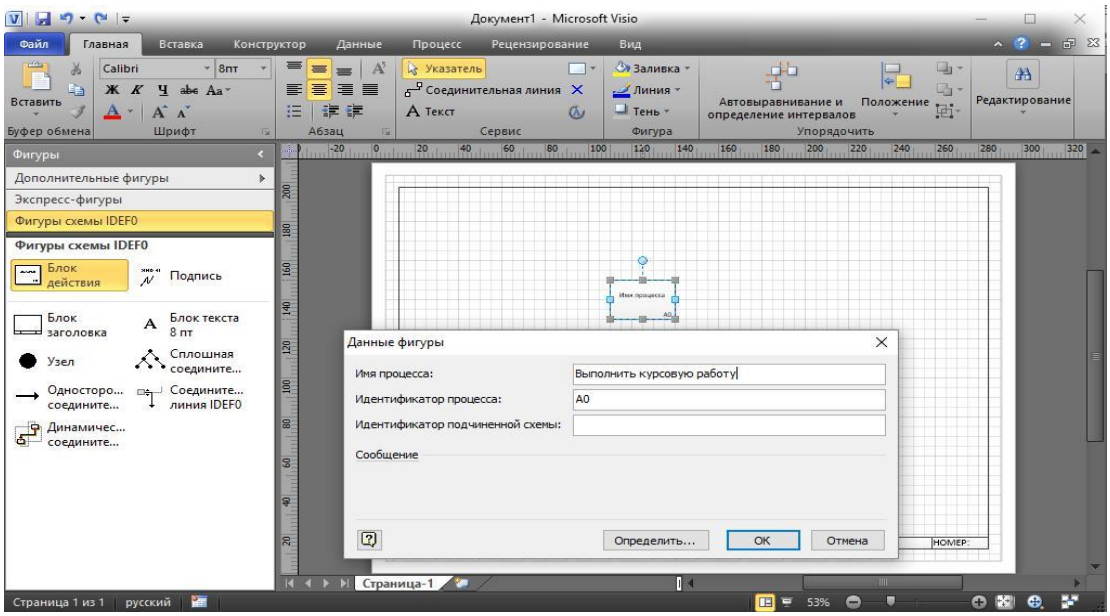


Рисунок 5 – Блок действия

С использованием блока *Односторонняя соединительная линия* создайте стрелки на контекстной диаграмме . Чтобы добавить текст необходимо дважды щелкнуть по стрелке.

Пример выполнения предыдущих пунктов представлен на рис. 6.

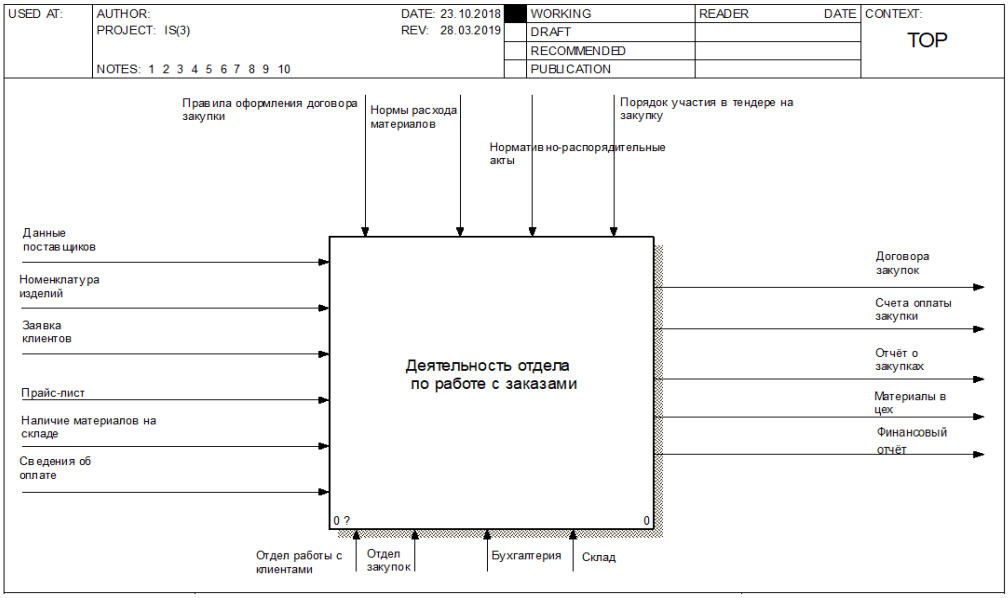


Рисунок 6 – Контекстная диаграмма

4.5. Создание диаграммы декомпозиции

1. Для построения декомпозиции диаграммы создайте новую страницу путем нажатия правой кнопкой мыши внижнем левом углу окна на ярлык Страница 1. Выбрать пункт Добавить страницу (рис.7)

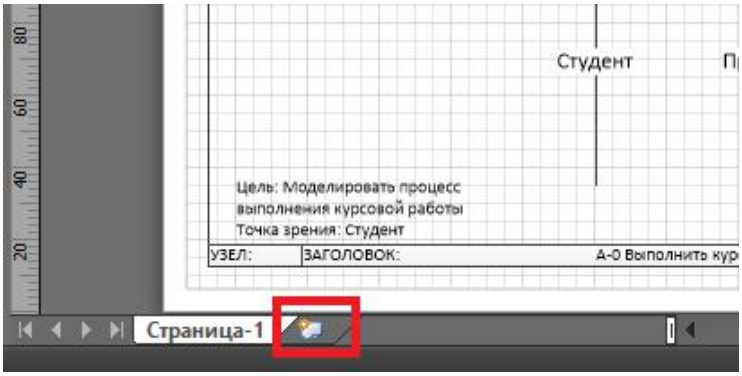


Рисунок 7– Добавление страницы

2. Переименуйте страницы в соответствии с уровнем декомпозиции, например: А-0, А1 и т.д.
3. Распределите работы диаграммы декомпозиции в области Блока заголовка.
4. Распределите стрелки для диаграммы декомпозиции в соответствии с контекстной диаграммой.

Итог выполнения вышеописанных шагов представлен на рис. 8.

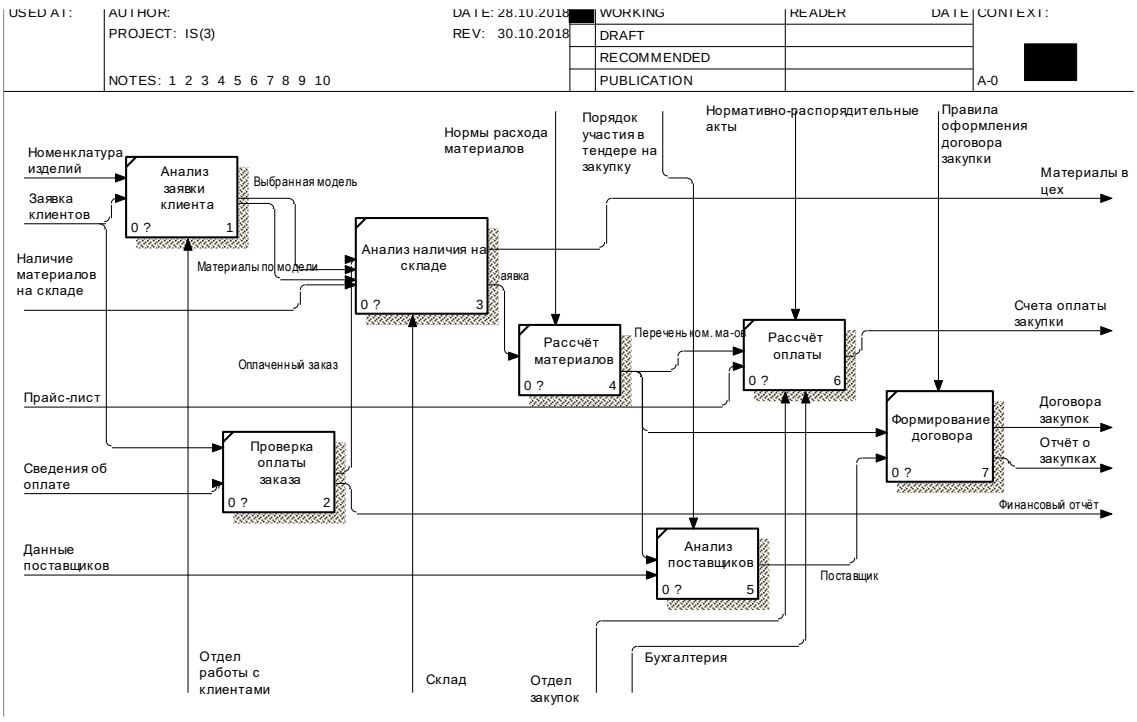


Рисунок 8 – Диаграмма декомпозиции

5. Задание к лабораторной работе

Построить модель бизнес-процессов выбранной предметной области.

6. Контрольные вопросы

1. Что такое функциональная модель бизнес-процесса?
2. Какие конструктивные элементы используются для построения функциональной модели?
3. Как представляется поток материальных, информационных, финансовых объектов?
4. Как трактуется и представляется управление выполнением функций?
5. Как представляются исполнители бизнес-процессов?
6. Как отражается использование информационной системы в бизнес-процессе?

7. Что такое ICOM метки и как они используются?
8. Что такое туннельные дуги и как они используются?
9. Что такое главный путь бизнес-процесса и как он отражается?
10. Как трактуются и представляются разветвления и соединения путей бизнес-процесса?
11. Как трактуются и представляются циклы в бизнес-процессе?

Практическая работа №4

Построение модели потоков данных в нотации IDEF3 и DFD

1. Цель работы

Выполнить построение диаграмм по методологии DFD.

Задачи работы: получить навыки построения диаграмм потоков данных в методологии SADT и нотациях IDEF3 и DFD редакторе Microsoft Visio.

2. Основные теоретические положения

Диаграммы потоков данных (**Data flow diagramm**, **DFD**) являются средством моделирования функциональных требований к проектируемой системе и используются для описания документооборота и обработки информации. В соответствии с данными методами модель системы определяется как иерархия диаграмм потоков данных, описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи пользователю. Диаграммы верхних уровней иерархии (контекстные диаграммы) определяют основные процессы или подсистемы с внешними входами и выходами.

Они детализируются при помощи диаграмм нижнего уровня. Такая декомпозиция продолжается, создавая многоуровневую иерархию диаграмм, до тех пор, пока не будет достигнут уровень декомпозиции, на котором процессы становятся элементарными, и детализировать их далее невозможно.

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

DFD описывает:

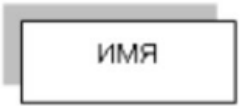
- функции обработки информации (работы, процессы);
- документы (стрелки, **arrow**), объекты, сотрудников или отделы, которые участвуют в обработке информации;
- внешние ссылки (**external references**), которые обеспечивают интерфейс с внешними объектами, находящимися за границами моделируемой системы;
- таблицы для хранения документов (хранилище данных, **data store**).

Работы обычно именуются по названию системы, например, **«Система обработки информации»**.

В **DFD** стрелки могут сливаться и разветвляться, что позволяет описать декомпозицию стрелок. Каждый новый сегмент сливающейся или разветвляющейся стрелки может иметь собственное имя.

В **DFD** номер каждой работы может включать префикс, номер родительской работы **A** и номер объекта. Номер объекта – это уникальный номер работы на диаграмме. Накопитель данных имеет префикс **D** и уникальный номер, например, **D5**.

Таблица 1 – Объекты диаграммы DFD в нотации Гейна-Сарсона

Наименование	Назначение
Процессы 	Представляют собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом. Физически процессы могут быть реализованы различными способами: это может быть подразделение организации (отдел), выполняющее обработку входных документов и выпуск отчетов; программа; аппаратно-реализованное логическое устройство и т. д. Изображаются прямоугольниками со скругленными углами
Внешние сущности 	Представляют собой материальные объекты или физические лица, представляющие собой источник или приемник информации, например, заказчики, персонал, поставщики, клиенты, склад. Определение некоторого объекта или системы в качестве внешней сущности указывает на то, что они находятся за пределами границ анализируемой системы. Изображаются в виде прямоугольника с тенью и обычно располагаются по краям диаграммы
Потоки данных 	Поток данных определяет информацию, передаваемую через некоторое соединение от источника к приемнику. Реальный поток данных может быть информацией, передаваемой по кабелю между двумя устройствами; пересылаемыми по почте письмами; магнитными лентами или дискетами, переносимыми с одного компьютера на другой и т. д. Поток данных на диаграмме изображается линией, оканчивающейся стрелкой, которая

	показывает направление потока. Каждый поток данных имеет имя, отражающее его содержание
Накопитель данных номер | имя	Абстрактное устройство для хранения информации, которую можно в любой момент поместить в накопитель и через некоторое время извлечь, причем способы помещения и извлечения могут быть любыми. Накопитель данных может быть реализован физически в виде микрофиши, ящика в картотеке, таблицы в оперативной памяти, файла на магнитном носителе и т.д. Накопитель данных в общем случае является прообразом будущей базы данных, и описание хранящихся в нем данных должно быть увязано с информационной моделью (ERD)

3. Порядок выполнения работы в MS VISIO

1.Запустите MS Visio.

2.На закладке выбора шаблона выберите категорию Программы и БД и в ней элемент Схема модели потоков данных. Нажмите кнопку

3.Создать в правой части экрана.

Окно программы примет вид, подобный рис. 1

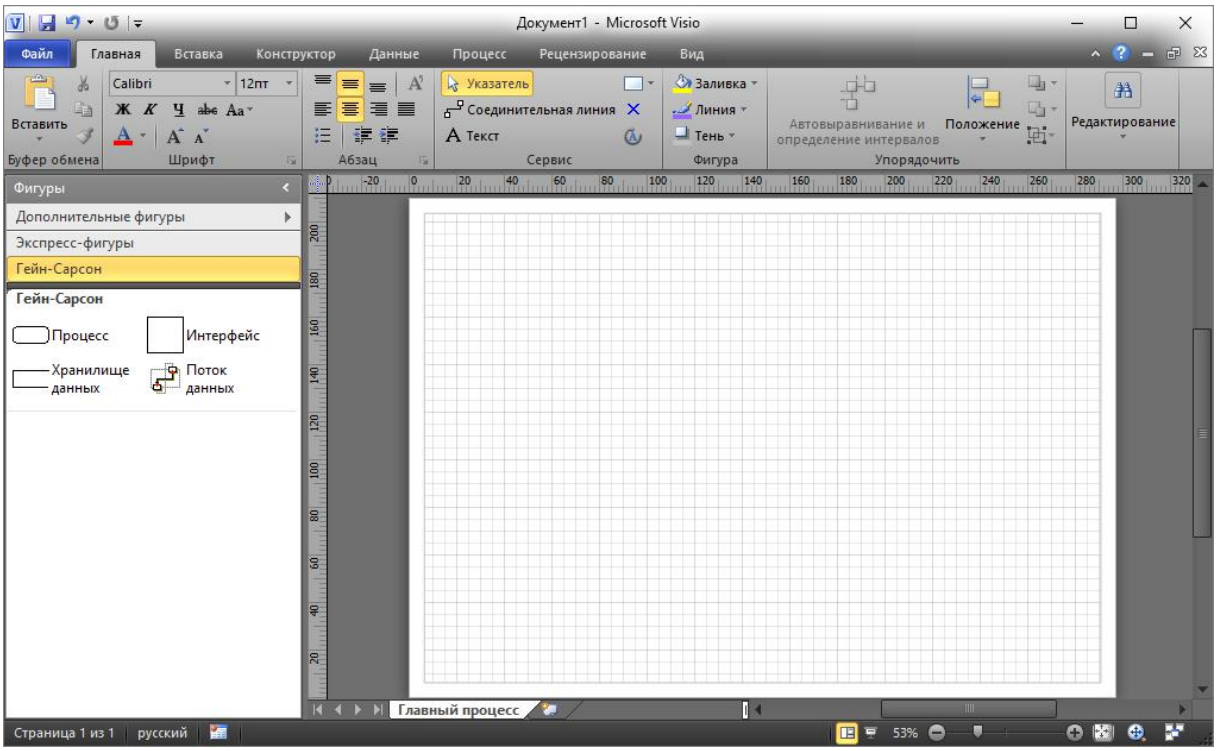


Рисунок 1 – Окно программы с выбранным шаблоном

4.Создание диаграммы.

4.1. Для удобства переведите страницу в альбомный вид: Конструктор – Параметры страницы – Ориентация – Альбомная;

4.2. Перетащите блок «Процесс» на пустую страницу, удерживая нажатой правую кнопку мыши;

4.3. Добавьте имя и номер процесса (рисунок 2), в данном случае: A0 Учет и анализ заявок клиентов;

4.4. Далее, добавьте «Внешние сущности (интерфейс)» и потоки данных.

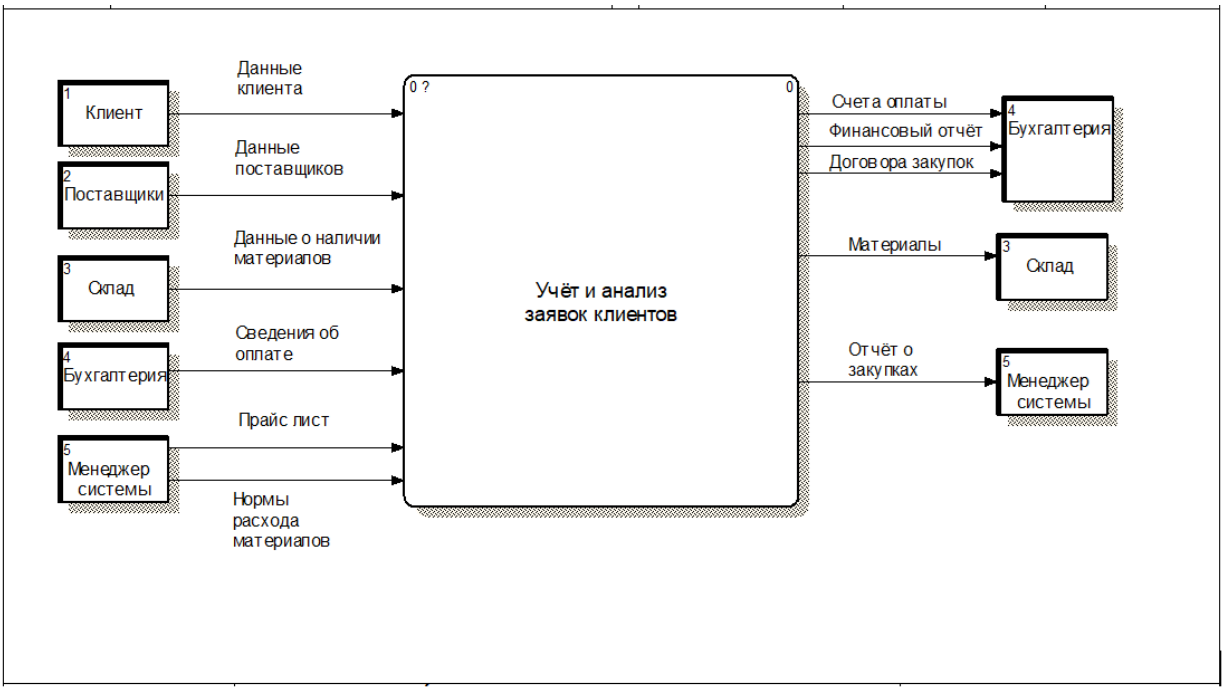


Рисунок 2 – Контекстная диаграмма в нотации DFD

4.5. Создание диаграммы декомпозиции.

Для построения декомпозиции диаграммы создайте новую страницу (рис. 3)

Рисунок 3 – Добавление страницы

Переименуйте страницы в соответствии с уровнем декомпозиции, например: 1-й уровень, 2-й уровень и т.д.

Распределите процессы, интерфейсы и хранилища данных диаграмм Свяжите объекты диаграммы DFD стрелками (потоками данных) как показано на рисунке 4.

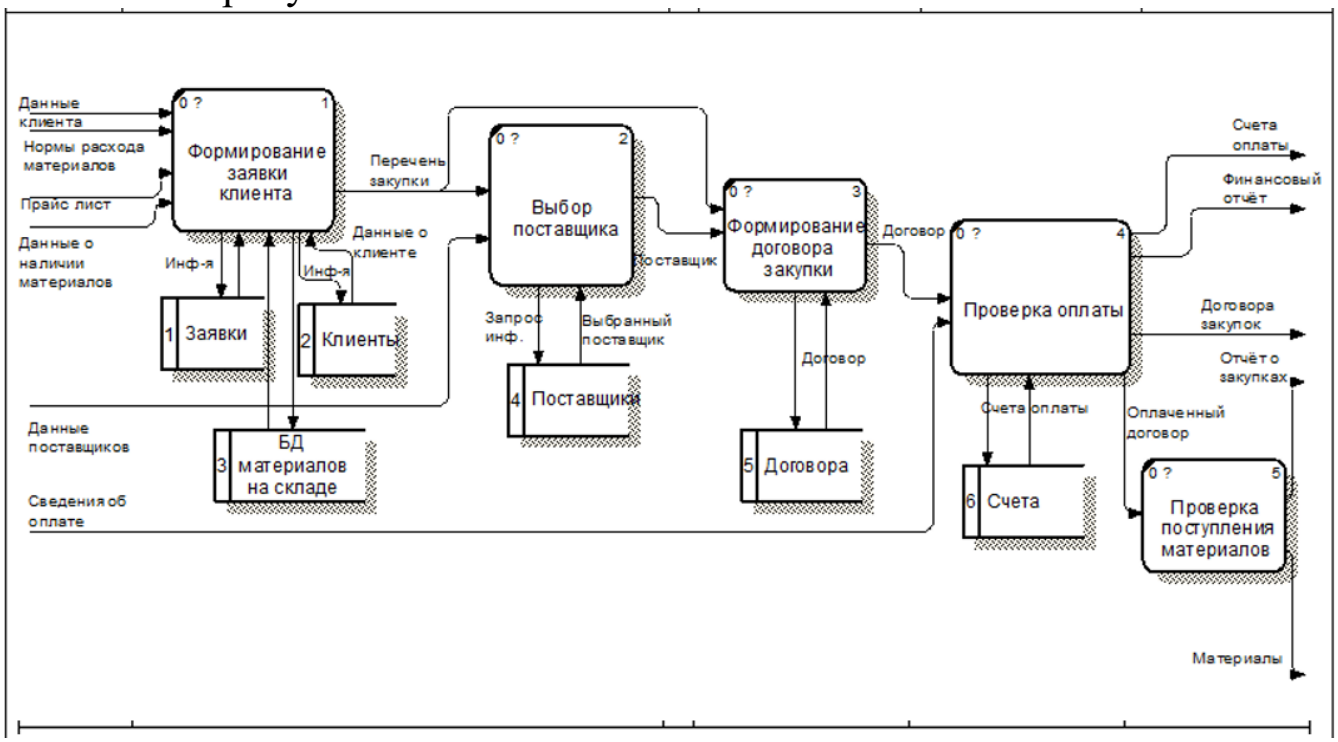


Рисунок 4 – Диаграмма декомпозиция процесса

Далее аналогично может быть осуществлена декомпозиция блоков A1-A5.

5. Задание к лабораторной работе

Построить модель потоков данных выбранной предметной области.

6. Контрольные вопросы

1. Каково назначение стандарта DFD? В чем основные отличия стандартов IDEF3 и DFD?
2. Каким образом в MS Visio создается схема DFD? Какие для этого используются нотации?
3. Какова роль основных элементов в стандарте DFD? Какие конструктивные элементы используются для построения модели потоков данных?

4. Как представляется поток материальных, информационных, финансовых объектов?
5. В чем отличие функциональной модели и модели потоков данных?

Практическая работа №5

Построение модели данных в нотации IDEF1x

1. Цель работы

Целью работы является освоение технологии построения информационной модели логического и физического уровней в нотации IDEF1X с использованием пакета Microsoft Office Visio.

2. Краткие теоретические сведения

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

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

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

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

- диаграмма сущность-связь (Entity-Relationship Diagram (*ERD*));

- модель данных, основанная на ключах (Key Based Model (*KB*));

- полная атрибутивная модель (Fully Attributed Model (*FA*)).

Физическая модель отражает физические свойства проектируемой базы данных (типы данных, размер полей, индексы). Параметры физической информационной модели зависят от выбранной системы управления базами данных (СУБД).

Сущность – это множество **реальных** или **абстрактных объектов** (людей, предметов, документов и т.п.), **обладающих общими атрибутами или характеристиками**. Любой объект системы может быть представлен только одной сущностью, которая должна быть уникально идентифицирована. *Именование сущности* осуществляется с помощью

существительного в единственном числе. При этом имя сущности должно отражать **тип** или **класс** объекта, а не его **конкретный экземпляр** (например, **Студент**, а не **Петров**) (рис. 1).

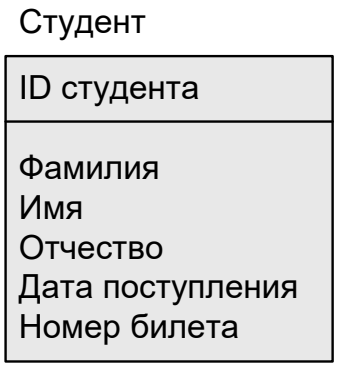


Рисунок 1– Графическое представление сущности «Студент» в MS Office Visio

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

Атрибут сущности – характеристика сущности, то есть свойство реального объекта. Например, на рис. 3.1 атрибутами сущности **«Студент»** являются **«ID студента»**, **«Фамилия»**, **«Имя»**, **«Отчество»**, **«Дата поступления»** и **«Номер билета»**.

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

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

Внешний ключ (Foreign Key, FK). Если между двумя сущностями *имеется специфическое отношение* связи или *категоризации*, то *атрибуты*, входящие в *первичный ключ родительской* или *общей сущности*, *наследуются* в качестве *атрибутов сущностью-потомком*

или *категориальной сущностью* соответственно. Эти атрибуты и называются внешними ключами.

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

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

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

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

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

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

3. Порядок выполнения работы

Информационная модель может быть построена на основе функциональной модели (в нотации IDEF0). В этом случае названия всех

интерфейсных дуг IDEF0-модели заносятся в *пул* – список потенциальных сущностей.

3.1. Создание логической модели «сущность-связь»

1. Запустите MS Office Visio.

2. На закладке выбора шаблона выберите категорию *Программное обеспечение и базы данных* и в ней элемент *Схема модели базы данных*. Нажмите кнопку *Создать* в правой части экрана.

3. Установите необходимые параметры страницы (масштаб, ориентация страницы).

4. MS Office Visio поддерживает различные нотации моделей баз данных. Для того чтобы задать нотацию IDEF1X, необходимо выбрать пункты меню *База данных* → *Параметры* → *Документ*. В открывшемся окне на вкладке *Общие* установить переключатель в меню *Набор символов* на *IDEF1X*. Меню *Имена, видимые на схеме* позволяет указать, какие имена атрибутов сущности будут отображены на диаграмме (концептуальные, физические или оба варианта одновременно). В данном случае для логического представления информационной модели необходимо выбрать пункт *Концептуальные имена* (рис.2).

В закладке *Отношение* окна *Параметры документа базы данных* в меню *Показывать* нужно отметить галочкой пункт *Мощность*, в меню *Отображение вида* выбрать пункт *Показывать вербальную фразу*, снять галочку в пункте *Обратный текст* (рис.2). Данные настройки позволят отобразить имя и мощность связи в модели.

5. Для того чтобы создать сущность, необходимо перетащить элемент  на рабочее поле. Переход в режим редактирования сущности осуществляется двойным щелчком по сущности или по нажатию правой кнопки мыши и выбора пункта меню *Свойства базы данных*.

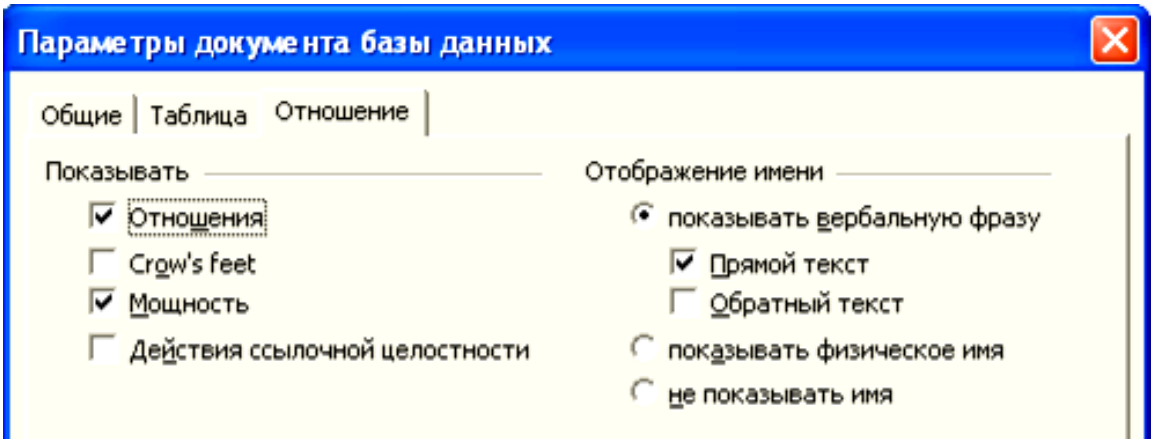


Рисунок 2 – Настройка параметров модели

Чтобы задать имя сущности, в окне *Свойства базы данных* нужно выбрать категорию *Определение*, снять галочку в пункте *Синхронизация имен при вводе* (в противном случае, физическое и логическое имя сущности будут совпадать, что по практическим соображениям не всегда удобно) и задать концептуальное имя сущности.

6. Далее необходимо установить связи между сущностями. В MS Office Visio по умолчанию используется *не идентифицирующее* отношение. Чтобы изменить **тип связи**, необходимо двойным щелчком по связи открыть окно *Свойства базы данных* и в категории в категории *Прочее* указать тип отношения (идентифицирующее, не идентифицирующее). В этой же категории указывается мощность связи (рис. 3).

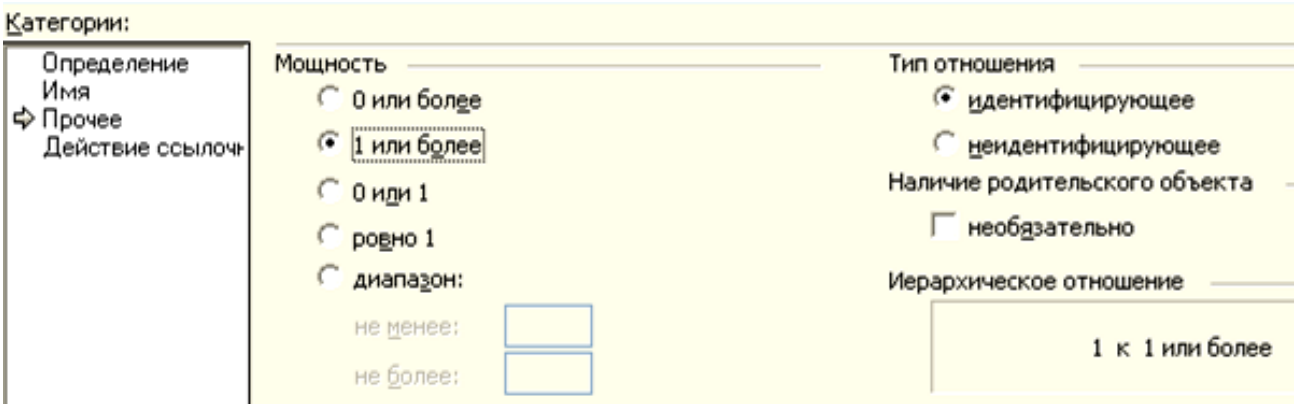


Рисунок 3 – Определение типа связи и мощности

Примечание. Кроме того, при не идентифицирующем отношении нужно указать, является ли наличие родительской сущности обязательным (т.е. может ли существовать экземпляр дочерней сущности, если не существует экземпляра родительской). Если наличие родительского объекта является необязательным, графически это отобразится в виде не закрашенного ромба со стороны родительской сущности.

Следующий шаг – в категории *Имя* в поле *Вербальная фраза* нужно указать имя отношения (рис. 4). Также можно указать имя связи в поле *Обратная фраза* для спецификации отношения потомок-родитель (в нашем случае обратная фраза отображаться не будет).

Примечание. Все изменения при закрытии окна свойств сохраняются автоматически.

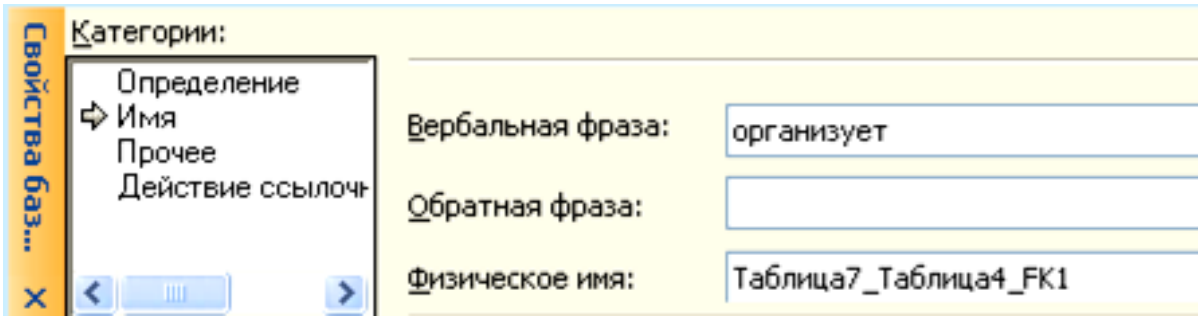


Рисунок 4 – Определение имени отношения

После определения имен, типов связей и задания мощностей получим информационную модель.

Пример информационной модели приведен на рис.5.

4. Контрольные вопросы

1. Для чего предназначена диаграмма «сущность-связь»?

2. Чем отличается полная атрибутивная модель от диаграммы «сущность-связь»?

3. Какие виды отношений существуют и чем они отличаются?

4. Приведите пример идентифицирующего отношения.

5. Приведите пример отношения полной категоризации.

6. Чем отличаются отношения полной и неполной категоризации?

7. Что представляет собой нормализация?

8. В чем разница между логическим уровнем модели данных и физическим?

Практическая работа №6
Построение программной и технологической моделей информационных систем

1. Цель работы

Разработать программную модель и технологическую архитектуру ИС:

- Функциональная архитектура приложения (дерево автоматизируемых функций)
- Разработка структура диалога приложения
- Дерево программных модулей
- Поведенческая модель (алгоритм функционирования системы)

2. Основные теоретические положения

Архитектура автоматизируемых бизнес-процессов **определяет функциональную архитектуру приложения (***совокупность функциональных подсистем и процедур***)** в виде набора операций, функций, задач *обработки информации*, обеспечивающих реализацию бизнес-процессов.

Функциональная архитектура приложения может быть представлена деревом функций предметной области.

Дерево автоматизируемых функций.

Следует привести иерархию функций управления и обработки данных, которые призван автоматизировать разрабатываемый программный продукт. При этом можно выделить и детализировать два подмножества функций: а) реализующих служебные функции (например, проверки пароля, ведения календаря, архивации баз данных и др.), б) реализующих основные функции ввода первичной информации, обработки, ведения справочников, ответов на запросы и др. (рис. 1)

Разработка структура диалога приложения.

В этом пункте следует выбрать способ описания диалога. Как правило, применяется два способа описания диалога.

Первый предполагает использование табличной формы описания. Второй использует представление структуры диалога в виде орграфа,

вершины которого могут быть перенумерованы, а описание его содержания в соответствии с нумерацией вершин, либо в виде экранов, если сообщения относительно просты, либо в виде таблицы. Пример приведен на рис.2.

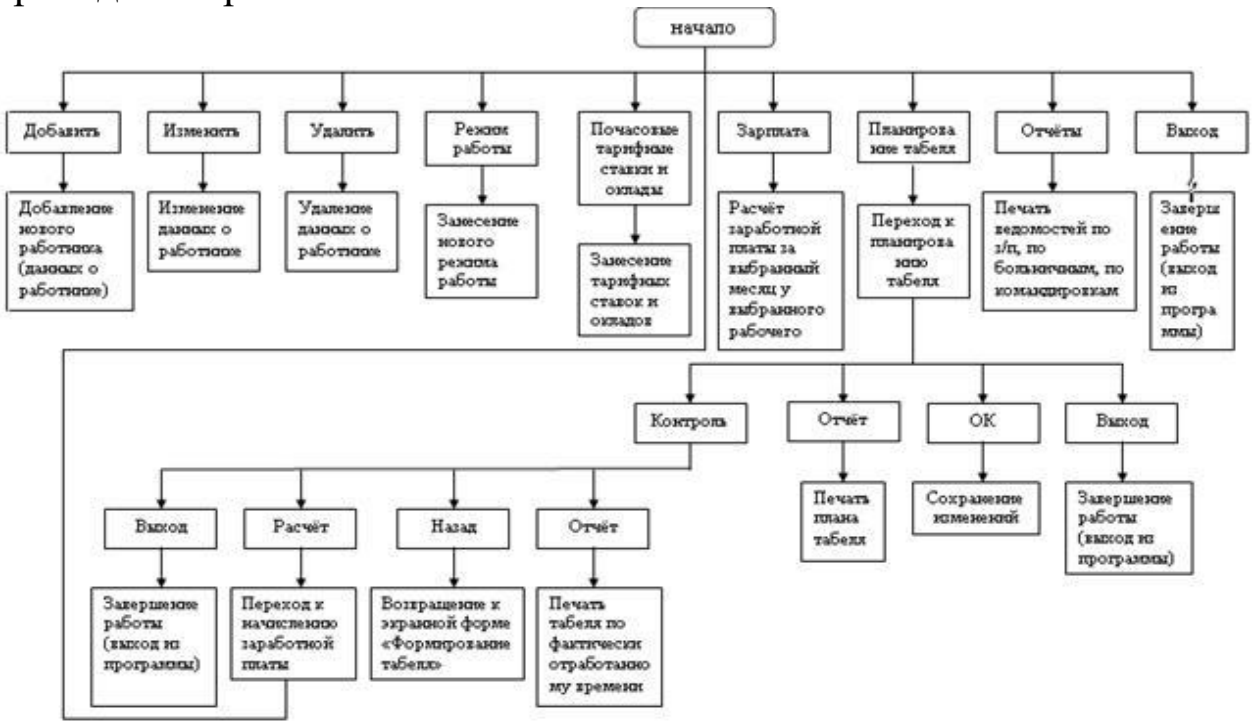


Рисунок 1– Пример функциональной архитектуры

Рисунок 2– Пример структуры диалога

Дерево программных модулей. На основе результатов, полученных выше, строится дерево программных модулей, отражающих структурную схему пакета, содержащей программные модули различных классов:

– выполняющие служебные функции;
– управляющие модули, предназначенные для загрузки меню и передачи управления другому модулю;
– модули, связанные с вводом, хранением, обработкой и выдачей информации.

Схема взаимосвязи программных модулей и информационных файлов отражает взаимосвязь программного и информационного обеспечения ИС. Пример на рис.3.

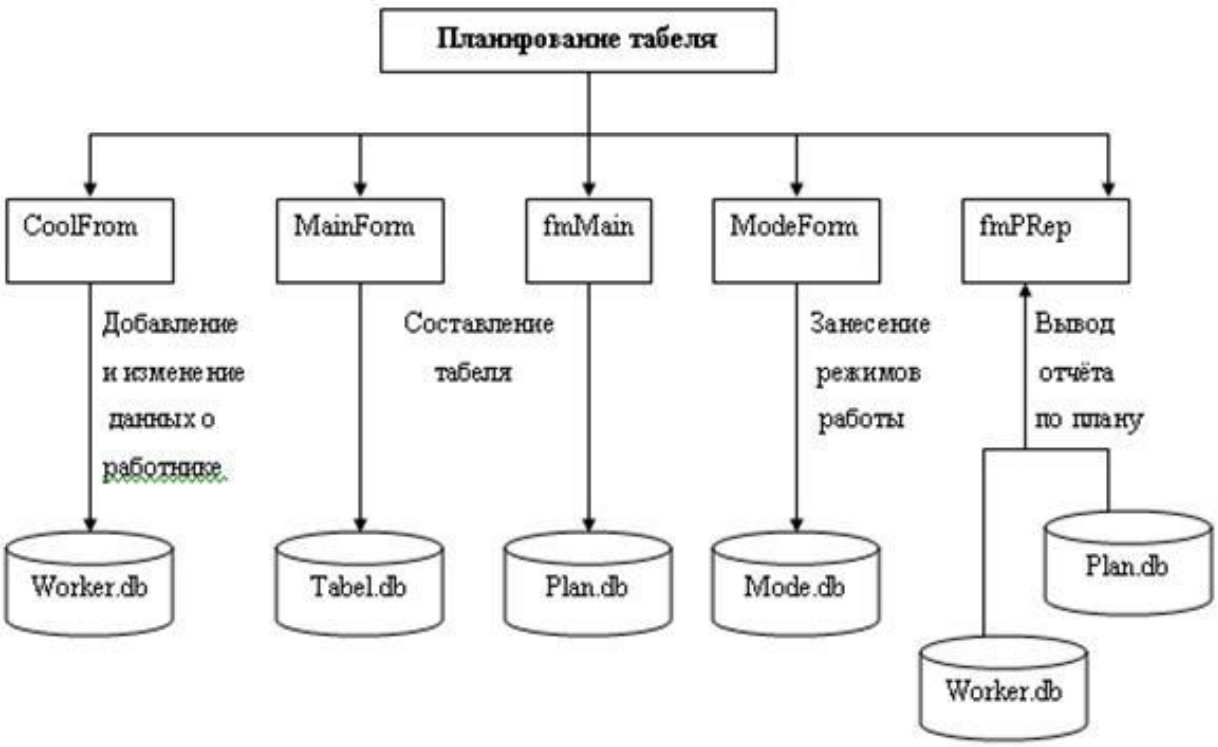


Рисунок 3– Дерево программных модулей

В данном пункте необходимо для каждого модуля указать идентификатор и выполняемые функции, например, в виде:

Идентификатор модуля	Выполняемые модулем функции

MainForm	Выбор пунктов главного меню
NewAppl	Поступление заявки на получение кредита
Add	Занесение клиента в базу данных

Поведенческая модель системы.

Наиболее известными и популярными способами и методологиями разработки поведенческих моделей являются:

– Блок-схемы алгоритмов;
– EPC –диаграммы (методология ARIS);
– Методология BPMN.

3.Задание к лабораторной работе

Разработать:

1. Функциональную архитектуру приложения.
2. Структуру диалога приложения.
3. Дерево программных модулей.
4. Алгоритм функционирования системы.

4.Контрольные вопросы

1. В чем заключается описание функциональной архитектуры?
2. Что отражает структуру диалога приложения?
3. Каким образом описывается и документируется алгоритм функционирования системы?
4. Каким образом производится кодирование полученной документации?
5. В чем отличие структуры диалога приложения от дерева программных модулей?

Список литературы

1. Абрамов, Г. В. Проектирование информационных систем [Электронный ресурс] : учебное пособие / Г. В. Абрамов, И. Медведкова, Л. Коробова. - Воронеж : Воронежский государственный университет инженерных технологий, 2012. - 172 с.
2. Золотов, С. Ю. Проектирование информационных систем [Электронный ресурс] : учебное пособие / С. Ю. Золотов. - Томск : Эль Контент, 2013. - 88 с.
3. Вендров, А.М. Проектирование программного обеспечения [Текст] : учебник / А.М. Вендров.— М: Финансы и Статистика, 2006. — 352с.
4. Лапина, Т. И. Информационные системы. Проектный практикум к выполнению и защите ВКР бакалавра по направлению 09.03.02 Информационные системы, 09.03.03 Прикладная информатика/ Т. И. Лапина//Юго-Западный гос. ун-т, ЗАО «Университетская книга»– Курск, 2016.–99с.
5. Стасышин, В. М. Проектирование информационных систем и баз данных [Электронный ресурс] : учебное пособие / В. М. Стасышин. - Новосибирск : НГТУ, 2012. - 100 с.

Практическая работа №7

Разработка модели функционирования ИС на основе диаграмм вариантов использования языка UML

1. Цель работы

Изучение метода объектно-ориентированного моделирования систем с использованием диаграмм вариантов использования унифицированного языка моделирования UML. Разработка диаграммы вариантов использования.

2. Основные теоретические положения

Унифицированный язык моделирования UML (Unified Modeling Language) представляет собой язык для определения, представления, проектирования и документирования программных систем, организационно-экономических систем, технических систем и других систем различной природы. UML содержит стандартный набор диаграмм и нотаций самых разнообразных видов. Стандарт UML версии 2.1 предлагает следующий набор диаграмм для моделирования:

- ***диаграммы вариантов использования (use case diagrams)*** - для моделирования бизнес-процессов организации и требований к создаваемой системе;
- ***диаграммы классов (class diagrams)*** - для моделирования статической структуры классов объектов системы и связей между ними;
- ***диаграммы поведения системы (behavior diagrams):***
  - ***диаграммы взаимодействия (interaction diagrams):***

    диаграммы последовательности (sequence diagrams) и ***кооперативные диаграммы (collaboration diagrams)*** - для моделирования процесса обмена сообщениями между объектами;
  - ***диаграммы состояний (statechart diagrams)*** - для моделирования поведения объектов системы при переходе из одного состояния в другое;
  - ***диаграммы деятельностей (activity diagrams)*** - для моделирования поведения системы в рамках различных вариантов использования, или моделирования деятельностей;
- ***диаграммы реализации (implementation diagrams):***
  - ***диаграммы компонентов (component diagrams)*** - для моделирования иерархии компонентов (подсистем) системы;
  - ***диаграммы размещения (deployment diagrams)*** — для моделирования

физической архитектуры системы.

Элементы экрана интерфейса Rose - это браузер, окно документации, панели инструментов, окно диаграммы и журнал. Их назначение заключается в следующем:

- браузер (browser) - используется для быстрой навигации по модели;
- окно документации (documentation window) - применяется для работы с текстовым описанием элементов модели;
- панели инструментов (toolbars) - применяются для быстрого доступа к наиболее распространенным командам;
- окно диаграммы (diagram window) - используется для просмотра и редактирования одной или нескольких диаграмм UML;
- журнал (log) - применяется для просмотра ошибок и отчетов о выполнении различных команд.

На рис. 1 показаны различные части интерфейса Rose.

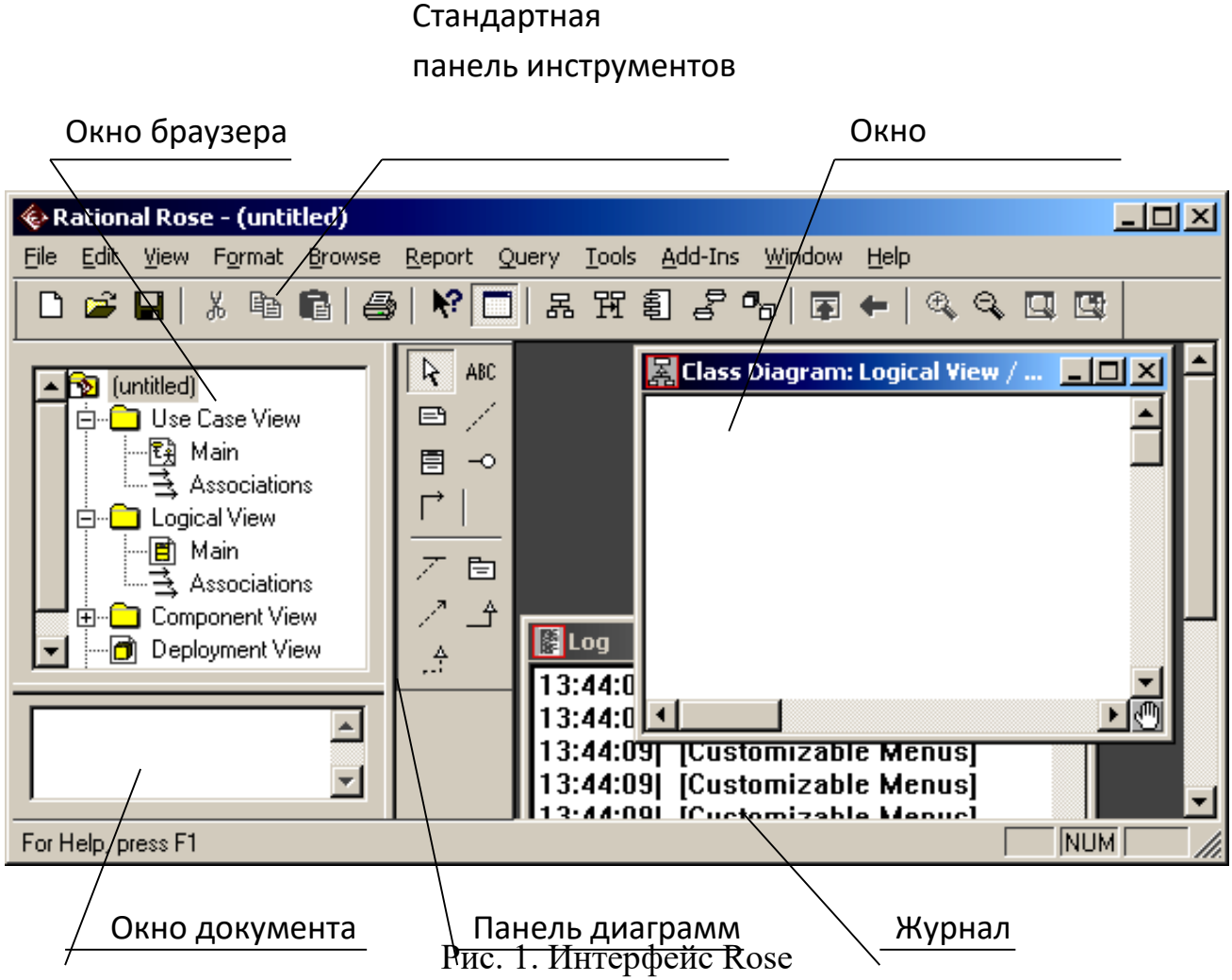


Рис. 1. Интерфейс Rose

Браузер - это иерархическая структура, позволяющая осуществлять навигацию по модели. Все, что добавляется к ней - действующие лица, варианты использования, классы, компоненты, будет показано в окне браузера.

С помощью браузера можно:

- добавлять к модели элементы;
- просматривать существующие элементы модели;
- просматривать существующие связи между элементами модели;
- перемещать элементы модели;
- переименовывать эти элементы;
- добавлять элементы модели к диаграмме;
- связывать элемент с файлом или адресом Интернета;
- группировать элементы в пакеты;
- работать с детализированной спецификацией элемента;
- открывать диаграмму.

Браузер поддерживает четыре представления (***view***): представление вариантов использования, компонентов, размещения и логическое представление. Все они и содержащиеся в них элементы модели описаны ниже.

Организация браузера представляет собой древовидную структуру. Каждый элемент модели может содержать другие элементы, находящиеся ниже его в иерархии. Знак «–» около элемента означает, что его ветвь полностью раскрыта. Знак «+» - что его ветвь свернута.

На рис. 3 изображено представление вариантов использования в браузере Rose, где:

Main – главная диаграмма вариантов использования;

Bank Officer, ***Credit System***, ***Customer*** – действующие лица;

Change PIN, ***Deposit Funds***, ***Make Payment***, ***Transfer Funds***, ***View Balance***, ***Withdraw Money*** – варианты использования;

Associations – связи.

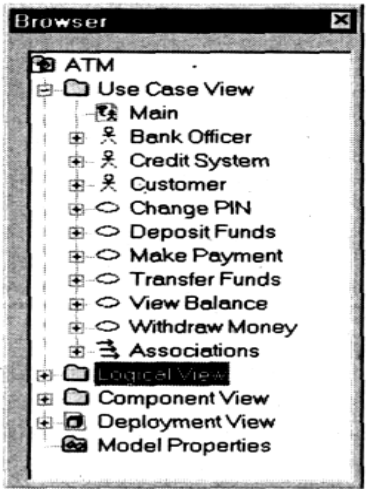


Рис. 3. Представление вариантов использования

Варианты использования

Исходя из потребностей действующих лиц выделяются следующие варианты использования:

- ***Login*** (Войти в систему).
- ***Register for Courses*** (Зарегистрироваться на курсы).
- ***View Report Card*** (Просмотреть табель успеваемости).
- ***Select Courses to Teach*** (Выбрать курсы для преподавания).
- ***Submit Grades*** (Проставить оценки).
- ***Maintain Professor Information*** (Вести информацию о профессорах).
- ***Maintain Student Information*** (Вести информацию о студентах).
- ***Close Registration*** (Закрыть регистрацию).

Упражнение 2. Создание вариантов использования в среде Rational Rose

Для того чтобы поместить вариант использования в браузер:

1. Щелкните правой кнопкой мыши по пакету представления вариантов использования в браузере.
2. Выберите в появившемся меню пункт ***New > Use Case***.
3. Новый вариант использования под названием ***NewUseCase*** появится в браузере. Слева от него будет видна пиктограмма варианта использования UML.
4. Выделив новый вариант использования, введите его название.

Результат выполнения упражнения показан на рис.5.

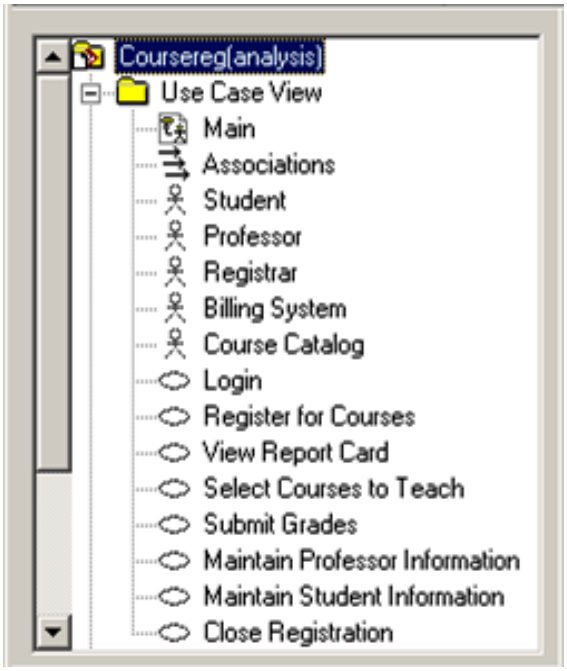


Рис. 5. Представление вариантов использования в браузере

Диаграмма вариантов использования

Создайте диаграмму вариантов использования для системы регистрации. Требуемые для этого действия подробно перечислены далее. Готовая диаграмма вариантов использования изображена на рис. 6.

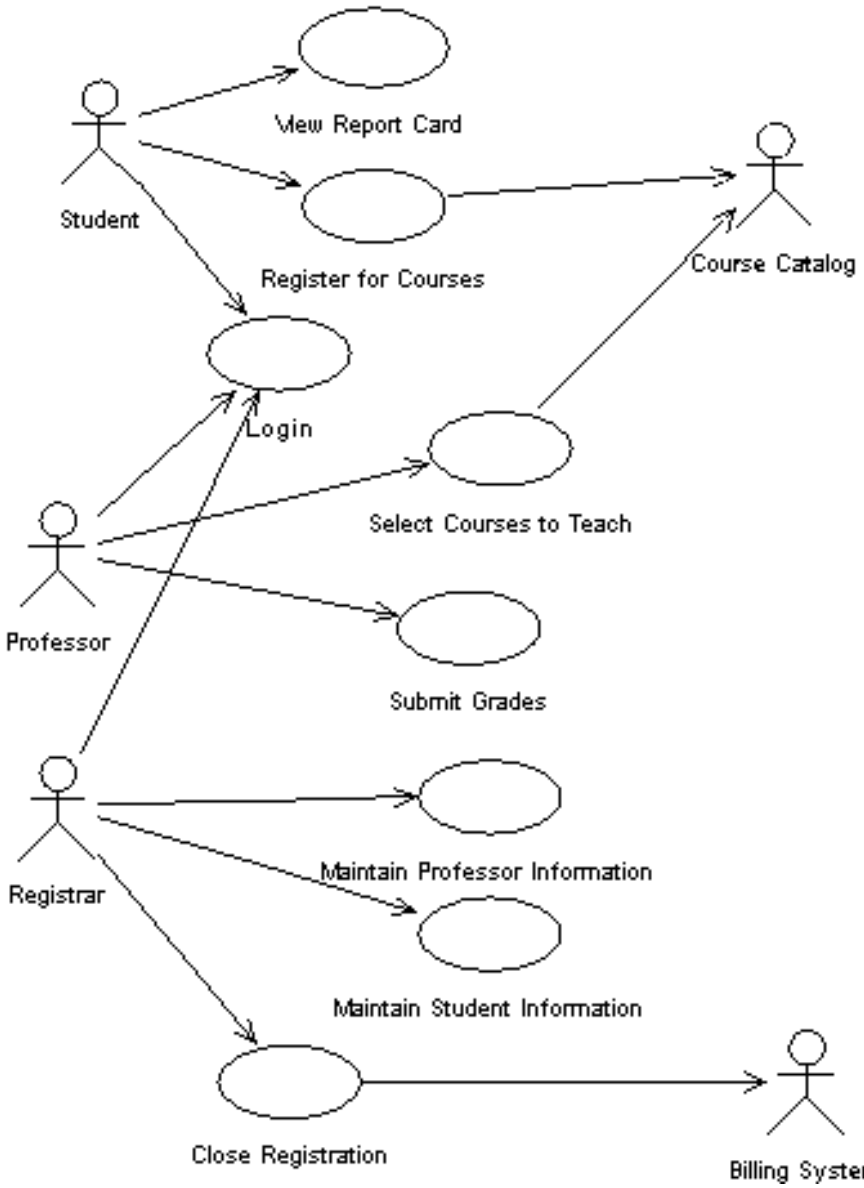


Рис. 6. Диаграмма вариантов использования для системы регистрации

В среде Rose диаграммы вариантов использования создаются в представлении вариантов использования. Главная диаграмма (***Main***) предлагается по умолчанию. Для моделирования системы можно затем разработать необходимое количество дополнительных диаграмм.

3 Порядок в ыполнения работы

Изучить раздел 1 и 2.

1. Создать концептуальную модель системы в соответствии с вариантом задания.
   a. Выполнить постановку задачи. Для этого:
      - ознакомится с предметной областью по теме работы;
      - выяснить субъективные цели и задачи (желаемый результат) у заказчика (его роль выполняет преподаватель);
      - провести анализ полученных данных;
      - создать содержательное описание системы.
   b. Разработать глоссарий проекта.
2. Создать разработанную в п. 3 модель вариантов использования.

Контрольные вопросы

1. **Для чего используется язык UML?**
2. **Какие диаграммы входят в состав языка UML?**
3. **В чем смысл варианта использования?**
4. **Каково назначение диаграмм вариантов использования:**
5. **Назовите основные свойства вариантов использования.**
6. **Назовите основные компоненты диаграмм вариантов использования.**
7. **Что такое «действующее лицо»?**
8. **Какую роль могут играть действующие лица по отношению к варианту использования?**
9. **Каким образом анализ внешних событий позволяет определить варианты использования системы?**

Практическая работа №8

Разработка диаграмм классов с использованием языка UML

1. Цель работы

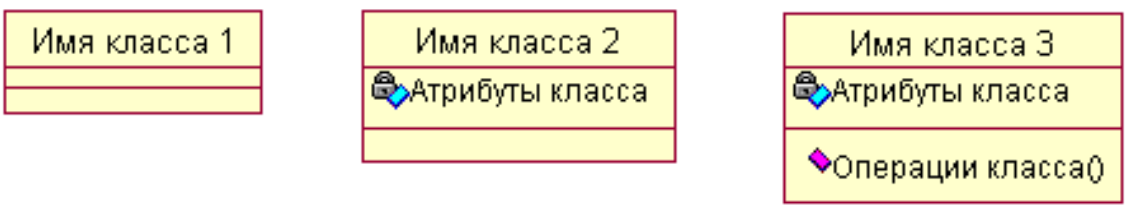
Изучение моделирования систем на логическом уровне с использованием диаграмм классов унифицированного языка моделирования UML.

2.Основные теоретические положения

Диаграмма классов ***(class diagrams)*** служит для представления статической структуры модели системы в терминах классов объектно-ориентированного проектирования. Диаграмма классов определяет типы классов системы и различного рода статические связи, которые существуют между ними. На диаграммах классов изображаются также атрибуты классов, операции классов и ограничения, которые накладываются на связи между классами.

Класс

Класс (***class***) в языке UML служит для обозначения множества объектов, которые обладают одинаковой структурой, поведением и отношениями с объектами из других классов. Графически класс изображается в виде прямоугольника, который дополнительно может быть разделен горизонтальными линиями на разделы или секции (рис. 1). В этих разделах могут указываться имя класса, атрибуты (переменные) и операции (методы).



а б в

Рис. 1. Графическое изображение класса на диаграмме классов

Обязательным элементов обозначения класса является его имя. На начальных этапах разработки диаграммы отдельные классы могут обозначаться простым прямоугольником с указанием только имени соответствующего класса (рис. 1, а). По мере проработки отдельных компонентов диаграммы описания классов дополняются атрибутами (рис. 1, б) и операциями (рис. 1, в).

Стереотипы классов

Стереотипы - это механизм, позволяющий разделять классы на категории. В языке UML основными стереотипами являются: ***Boundary*** (граница), ***Entity*** (сущность) и ***Control*** (управление).

Граничные классы (***boundary classes***) - это классы, которые расположены на границе системы и окружающей среды. Они включают все формы, отчеты, интерфейсы с аппаратурой (такой, как принтеры или сканеры) и интерфейсы с другими системами.

Классы-сущности (***entity classes***) отражают основные понятия (абстракции) предметной области и, как правило, содержат хранимую информацию. Обычно для каждого класса-сущности создают таблицу в базе данных.

Управляющие классы (***control classes***) отвечают за координацию действий других классов. Обычно у каждого варианта использования имеется один управляющий класс, контролирующий последовательность событий этого варианта использования.

Атрибуты

Атрибут (***attribute***) класса - это элемент информации, связанный с классом. Атрибут служит для представления отдельного свойства или признака, который является общим для всех объектов данного класса. Например, у класса ***Company*** (Компания) могут быть атрибуты ***Name*** (Название), ***Address*** (Адрес) и ***NumberOfEmployees*** (Число служащих). Атрибуты записываются во второй сверху секции прямоугольника класса.

Операции

Операция (***operation***) – это некоторый сервис, который предоставляет каждый экземпляр или объект класса по требованию своих клиентов (других объектов, в том числе и экземпляров данного класса).

Интерфейсы.

Интерфейс (***interface***) является специальным случаем класса, у которого имеются только операции и отсутствуют атрибуты. Для его изображения используется специальный графический символ — прямоугольник класса со стереотипом <<***interface***>> (рис. 2). При этом секция атрибутов у прямоугольника отсутствует, а указывается только секция операций.

а) б)

Рис. 2. Пример изображения интерфейса

3. Порядок выполнения работы

1. Изучить раздел 1 и 2.
2. Изучить раздел 3. Выполнить упражнения 1 – 4.
3. Разработать инфологическую модель системы в виде диаграммы классов в соответствии с вариантом задания.

4. Создать разработанную в п. 3 модель в среде Rational Rose.

5. Контрольные вопросы

1. Каково назначение диаграмм классов?
2. Для чего используется диаграмма классов на стадии анализа?
3. Для чего используется диаграмма классов на стадии проектирования?
4. Назовите основные компоненты диаграмм классов.
5. Назовите основные типы статических связей между классами.
6. Что представляет собой ассоциация?
7. В чем смысл множественности ассоциаций?;
8. В чем отличие атрибутов от ассоциаций?
9. Что такое признак видимости?
10. Что представляет собой операция класса?
11. В чем смысл обобщения?
12. Каково назначение ограничений на диаграммах классов?

Практическая работа №9

Диаграммы активности (activity diagrams) и последовательности (sequence diagrams) языка UML

1.Цель работы

Изучение моделирования систем на логическом уровне с использованием диаграмм классов унифицированного языка моделирования UML.

2. Основные теоретические положения

Взаимодействие между объектами в системе представляются *диаграммами взаимодействия (interaction diagrams)*. Диаграммы взаимодействия подразделяются на два основных типа диаграмм: *диаграммы последовательности (sequence diagrams)* и *диаграммы деятельности (activity)*.

Как правило, диаграмма взаимодействия используется для описания поведения в рамках одного варианта использования. На такой диаграмме изображается ряд объектов и те сообщения, которыми они обмениваются в рамках этого варианта использования.

Диаграммы последовательности несут в себе одну информацию, но выраженную разными способами. Диаграммы последовательности показывают взаимодействие объектов во времени и отражают последовательность происходящих событий.

1. Диаграммы последовательности (sequence diagrams)

Диаграммы последовательности имеют две размерности: вертикальная представляет время, горизонтальная - различные объекты. Оси могут меняться местами, так что ось времени может располагаться горизонтально, слева направо, а список объектов располагаться вертикально.

Объект на диаграмме изображается в виде прямоугольника на вершине вертикальной пунктирной линии, называемой *линией жизни объекта (lifeline)* Она представляет собой фрагмент жизненного цикла объекта в процессе взаимодействия. Если объект создается или уничтожается на отрезке времени, представленном на диаграмме, то его линия жизни начинается и заканчивается в соответствующих точках, в противном случае линия жизни объекта проводится от начала до конца диаграммы. Символ объекта рисуется в начале его линии жизни; если объект создается не в начале диаграммы, то сообщение о создании объекта рисуется со стрелкой, проведенной к

символу объекта. Если объект уничтожается не в конце диаграммы, то момент его уничтожения помечается большим крестиком "X". При сообщении, вызывающем уничтожение объекта (или самоуничтожение), в конце возвращается сообщение об уничтожении объекта. Линия жизни может разветвляться в две (и более) параллельные линии, показываемые условно. Каждая ответвляющаяся линия соответствует переходу в потоке сообщений. Линии жизни могут объединяться в некоторой последующей отметке.

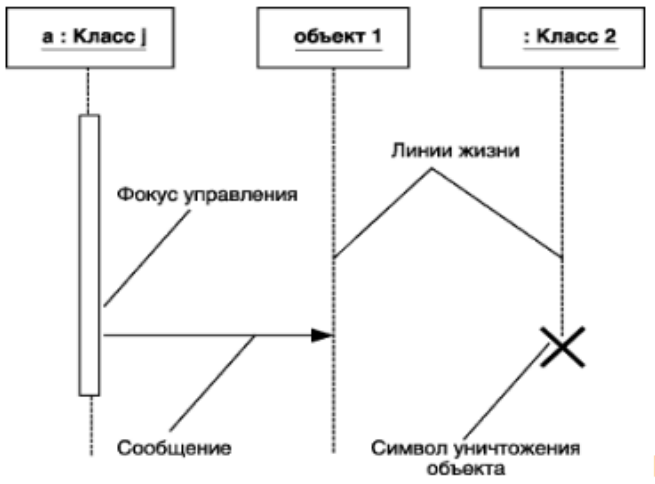


Рисунок 1 –Диаграмма последовательности

2. Построение диаграммы активности (Activity)

Диаграмма активности – позволяет моделировать **жизненный цикл объекта** в **виде** переходами из одного состояния (деятельности) в другое, т.е. отображают алгоритмы по преобразованию классов. Этот тип диаграмм позволяет проектировать алгоритмы поведения объектов с применением обозначений:

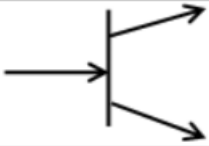
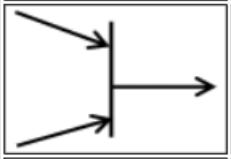
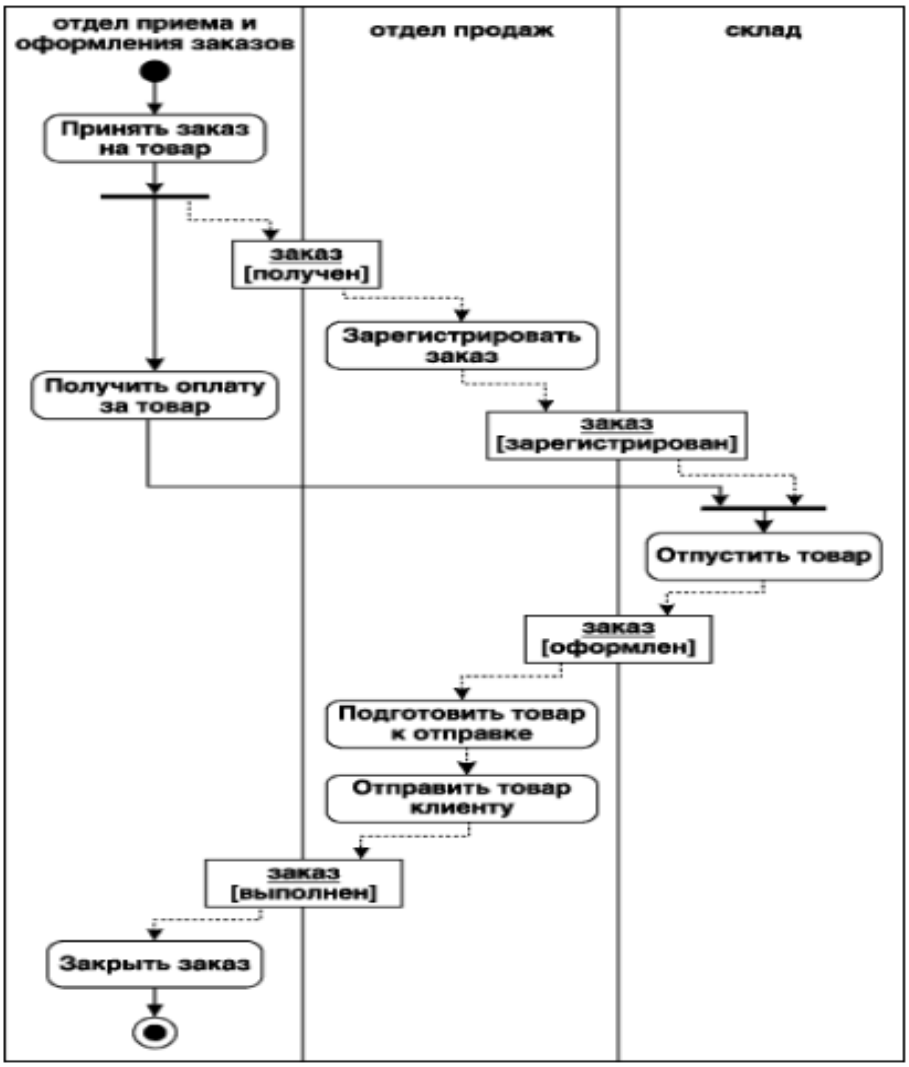
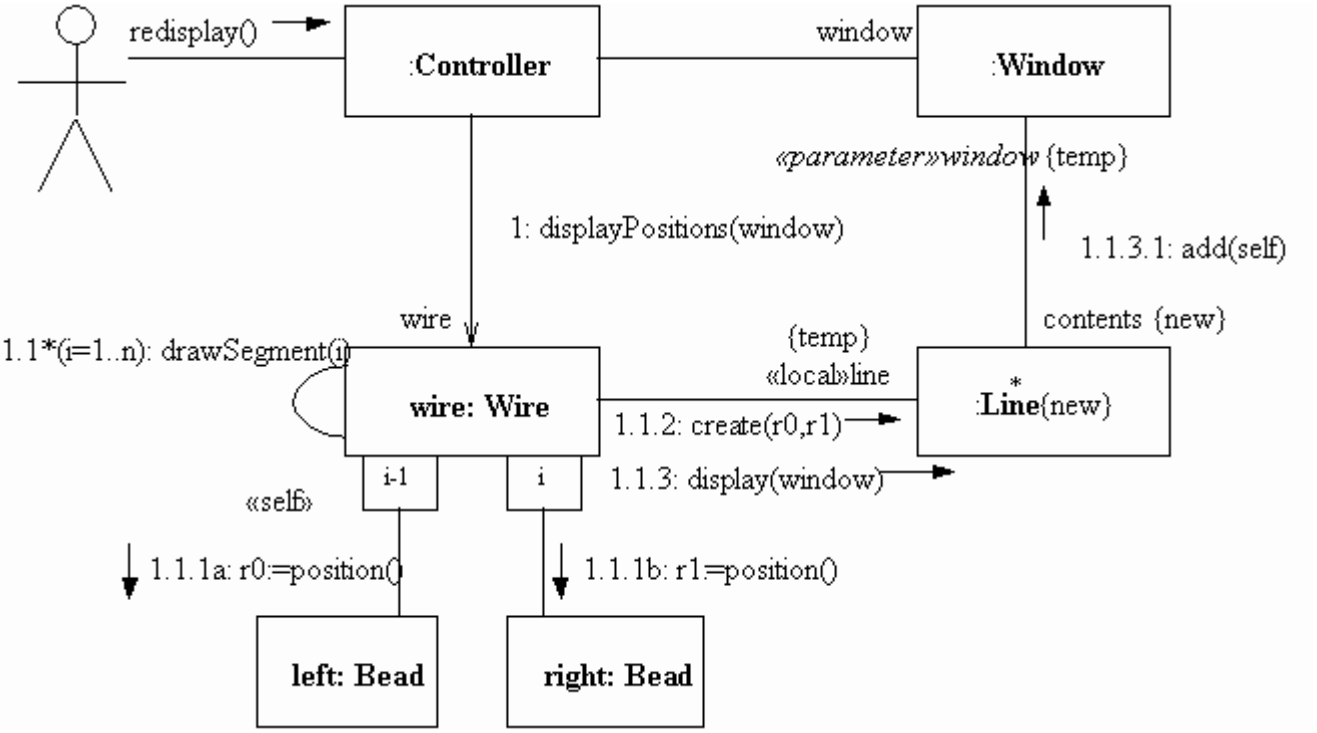
Обозначение	Наименование	
	на диаграмме автоматов	на диаграмме деятельности
	Начальное псевдосостояние (англ. initial pseudostate)	Начальный узел (англ. initial node)
	Конечное состояние (англ. final state)	Завершение деятельности (англ. activity final)
	Точка выхода (англ. exit point pseudostate)	Завершение потока (англ. flow final)
	Ветвление (англ. fork pseudostate)	Ветвление (англ. fork node)
	Соединение (англ. join pseudostate)	Соединение (англ. join node)
	Выбор (англ. choice pseudostate)	Слияние / решение (англ. merge / decision node)

Диаграмма активности позволяет разделить функции по исполнителям с использованием разделов диаграмм. Разделы группируют действия относительно какой-либо характеристики, например:



3. **Кооперативные диаграммы (collaboration diagrams)**

Кооперативные диаграммы (collaboration diagrams) предоставляют возможность пространственно располагать объекты. В отличие от диаграмм последовательности, на кооперативных диаграммах экземпляры объектов показываются в виде пиктограмм. На диаграмме отображаются лишь те объекты, что прямо или косвенно участвуют в выполнении данного варианта использования. Так же как на диаграмме последовательности, линии со стрелкой на конце обозначают сообщения, обмен которыми осуществляется в рамках данного варианта использования. Их временная последовательность, однако, указывается путем нумерации сообщений.



3. Выполнение работы

1. Изучить раздел 1 и 2.
2. Изучить раздел 3. Выполнить упражнения 1 – 4.
3. Разработать модель поведения системы в виде диаграмм активности и последовательности в соответствии с вариантом задания.
4. Создать разработанную в п. 3 модель в среде Rational Rose.

4. Контрольные вопросы

1. **Каково назначение диаграмм активности и последовательности?**
2. **Для чего используется диаграмма активности на стадии анализа?**
3. **Для чего используется диаграмма последовательности на стадии проектирования?**
4. **Назовите основные компоненты диаграммы последовательности.**
5. **Что представляет собой сообщение?**
6. **Что представляет собой диаграмма активности (деятельности)?**
7. **Назовите основные компоненты диаграммы деятельности ?**
8. **Каково назначение диаграммы кооперации?**

Практическая работа №10

Моделирования поведения системы на основе диаграмм состояний, взаимодействия (interaction diagrams) языка UML

5. Цель работы

Изучение моделирования систем на логическом уровне с использованием диаграмм состояний унифицированного языка моделирования UML.

6. Основные теоретические положения

Диаграммы состояний определяют все возможные состояния, в которых может находиться конкретный объект, а также процесс смены состояний объекта в результате наступления некоторых событий. Существует много форм диаграмм состояний, незначительно отличающихся друг от друга семантикой.

В качестве примера рассмотрим поведение объекта класса ***CourseOffering***. Он может находиться в открытом состоянии (возможно добавление данных о новом студенте) или в закрытом состоянии (максимальное количество студентов уже записалось на курс). Таким образом, конкретное состояние зависит от количества студентов, связанных с объектом ***CourseOffering***. Рассматривая каждый вариант использования, можно выделить еще два состояния: инициализацию (до начала регистрации студентов на курс) и отмену (курс исключается из расписания). Диаграмма состояний для класса ***CourseOffering*** приведена на рис. 1.

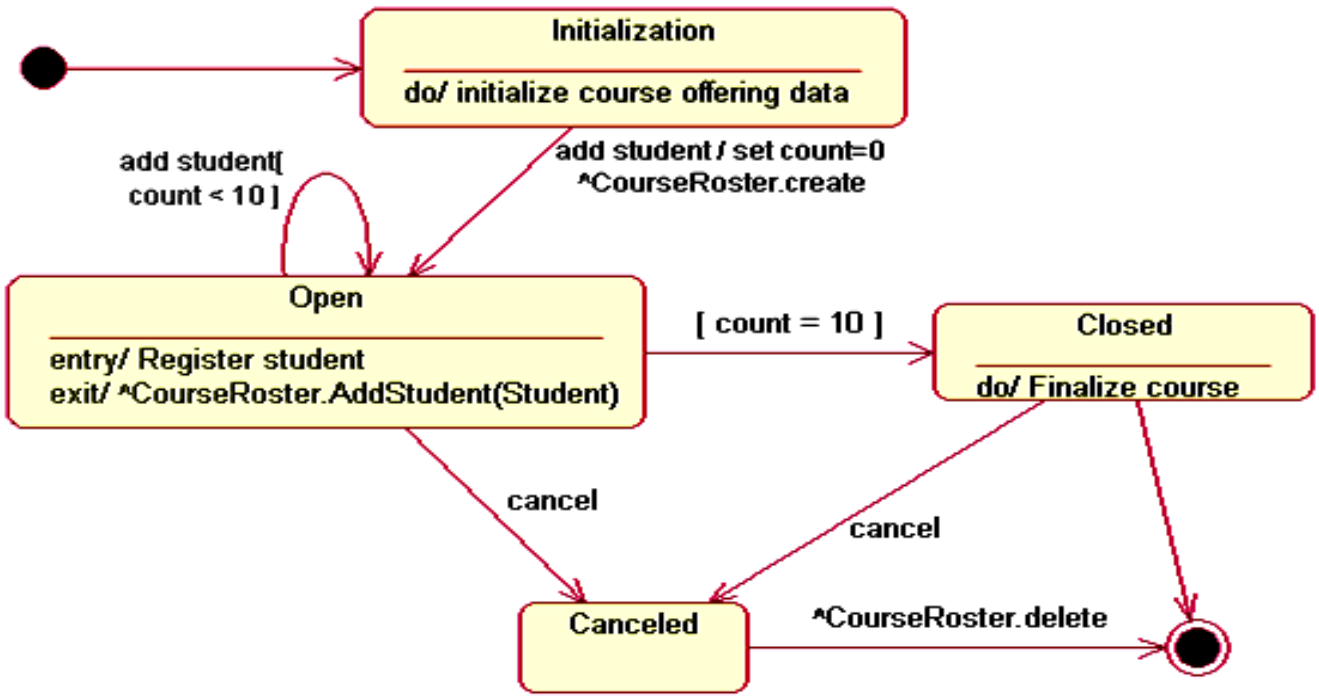


Рис. 1. Диаграмма состояний для класса CourseOffering

Упражнение 1. Создание диаграммы состояний для класса CourseOffering

Для создания диаграммы состояний:

1. Щелкните правой кнопкой мыши в браузере по нужному классу.

2. Выберите пункт ***New > Statechart Diagram*** в открывшемся меню.

Для того чтобы добавить состояние:

1. На панели инструментов нажмите кнопку ***State***.

2. Щелкните мышью на диаграмме состояний по тому месту, куда хотите поместить состояние.

Все элементы состояния можно добавить с помощью вкладки ***Detail*** окна спецификации состояния.

Для того чтобы добавить деятельность:

1. Откройте окно спецификации требуемого состояния.

2. Перейдите на вкладку ***Detail***.

3. Щелкните правой кнопкой мыши по окну ***Actions***.

4. Выберите пункт ***Insert*** в открывшемся меню.

5. Дважды щелкните по новому действию.

6. Введите действие в поле ***Actions***.

7. В окне ***When*** укажите ***Do***, чтобы сделать новое действие деятельностью.

Для того чтобы добавить входное действие, в окне ***When*** укажите ***On Entry***.

Для того чтобы добавить выходное действие, в окне ***When*** укажите ***On Exit***.

Послать событие можно с помощью следующих операций:

1. Откройте окно спецификации требуемого состояния.

2. Перейдите на вкладку ***Detail***.

3. Щелкните правой кнопкой мыши по окну ***Actions***.

4. Выберите пункт ***Insert*** в открывшемся меню.

5. Дважды щелкните по новому действию.

6. В качестве типа действия укажите ***Send Event***.

7. В соответствующие поля введите событие (***event***), аргументы (***arguments***) и целевой объект (***Target***). Для того чтобы добавить переход:

1. Нажмите кнопку ***Transition*** панели инструментов.

2. Щелкните мышью по состоянию, откуда осуществляется переход.

3. Проведите линию перехода до того состояния, где он завершается.

Чтобы добавить рефлексивный переход:

1. Нажмите кнопку ***Transition to Self*** панели инструментов.

2. Щелкните мышью по тому состоянию, где осуществляется рефлексивный переход.

Для того чтобы добавить событие, его аргументы, ограждающее условие и действие:

1. Дважды щелкните по переходу, чтобы открыть окно его спецификации.

2. Перейдите на вкладку ***General***.

3. Введите событие в поле ***Event.***

4. Введите аргументы в поле ***Arguments***.

5. Введите ограждающее условие в поле ***Condition***.

6. Введите действие в поле ***Action***.

Для отправки события:

1. Дважды щелкните по переходу, чтобы открыть окно его спецификации.

2. Перейдите на вкладку ***Detail***.

3. Введите событие в поле ***Send Event***.

4. Введите аргументы в поле ***Send Arguments***.

5. Задайте цель в поле ***Send Target***.

Для указания начального или конечного состояния:

1. На панели инструментов нажмите кнопку ***Start State*** или ***End State***.

2. Щелкните мышью на диаграмме состояний по тому месту, куда хотите поместить состояние.

3.Выполнение работы

4. Изучить раздел 1.

5. Изучить раздел 2. Выполнить упражнение 1.
6. Создать логическую модель системы в виде диаграмм состояний для вариантов использования созданных в соответствии с вариантом задания (лабораторная работа № 1).
7. Создать разработанную в п. 3 модель вариантов использования в среде Rational Rose.

4. Контрольные вопросы

10. Из каких основных элементов состоят диаграммы состояний?

11. В каком случае используются диаграммы состояний?

12. Чем отличаются диаграммы состояний и диаграммы деятельности?

13. Чем определяется переход объекта из одного состояния в другое?

Практическая работа №11
Диаграммы реализации: диаграммы размещения и компонентов

1.Цель работы

Изучение моделирования систем на логическом уровне с использованием диаграмм размещения и компонентов унифицированного языка моделирования UML.

2.Основные теоретические положения

Диаграмма развертывания (размещения) (deployment diagram) отражает физические взаимосвязи между программными и аппаратными компонентами системы. Она является хорошим средством для того, чтобы показать маршруты перемещения объектов и компонентов в распределенной системе. Каждый узел на диаграмме размещения представляет собой некоторый тип вычислительного устройства, в большинстве случаев — часть аппаратуры. Эта аппаратура может быть простым устройством или датчиком, а может быть и мэйнфреймом.

Диаграмма развертывания предназначена для визуализации элементов и компонентов программы, существующих только на этапе ее исполнения (***run-time***). При этом представляются только компоненты-экземпляры программы, являющиеся исполнимыми файлами или динамическими библиотеками. Те компоненты, которые не используются на этапе исполнения, на диаграмме развертывания не показываются. Так, компоненты с исходными текстами программ могут присутствовать только на диаграмме компонентов. На диаграмме развертывания они не указываются.

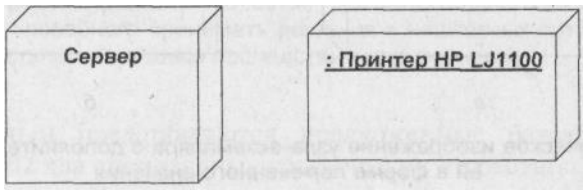
Цели, преследуемые при разработке диаграммы развертывания, следующие:

- указать размещение исполнимых компонентов программной системы по ее физическим узлам;

– показать физические связи между всеми узлами реализации системы на этапе ее исполнения;
– выявить узкие места системы и реконфигурировать ее топологию для достижения требуемой производительности.

Для обеспечения этих требований диаграмма развертывания разрабатывается совместно системными аналитиками, сетевыми инженерами и системотехниками. Далее рассмотрим отдельные элементы, из которых состоят диаграммы развертывания.

Узел (***node***) представляет собой некоторый физически существующий элемент системы, который может обладать некоторым вычислительным ресурсом. Графически узел на диаграмме развертывания изображается в форме трехмерного куба (строго говоря, псевдотрехмерного прямоугольного параллелепипеда).



а б

Рис. 1. Графическое изображение узла на диаграмме развертывания

Второй способ разрешает показывать на диаграмме развертывания узлы с вложенными изображениями компонентов (рис. 11.3, б). Важно помнить что в качестве таких вложенных компонентов могут выступать только исполняемые компоненты и динамические библиотеки.

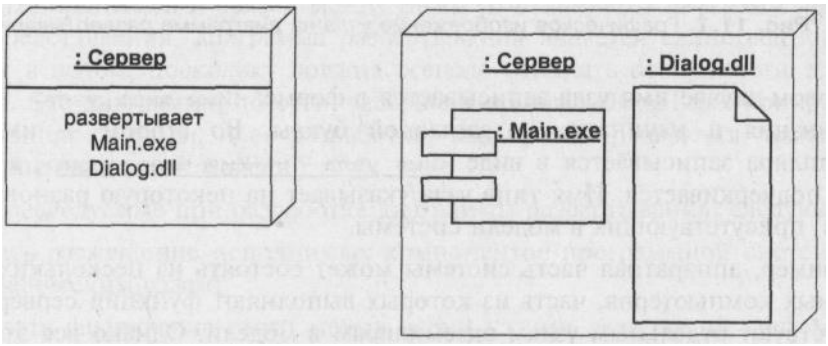


Рис.2. Варианты графического изображения узлов-экземпляров с размещаемыми на них компонентами

4. Порядок выполнения работы

1. Изучить раздел 1.
2. Изучить раздел 2. Выполнить упражнение 1.
3. Создать физическую модель системы в виде диаграммы размещения (развертывания) в соответствии с вариантом индивидуального задания (лабораторная работа № 1). Система должна иметь клиент-серверную архитектуру, должны быть отображены все необходимые устройства (принтеры и пр.). Для узлов должны быть отображены процессы и вид планирования процессов. С помощью примечаний представить развертывание компонентов на узлах диаграммы.

4. Контрольные вопросы

1. Для чего предназначена диаграмма компонентов?
2. Из каких основных элементов состоят диаграммы размещения?
3. В каком случае используются диаграммы размещения?
4. Какие обозначения используются в диаграммах размещения?

Список литературы

1. Лапина, Т. И. Информационные системы. Проектный практикум к выполнению и защите ВКР бакалавра по направлению 09.03.02 Информационные системы, 09.03.03 Прикладная информатика/ Т. И. Лапина//Юго-Западный гос. ун-т, ЗАО «Университетская книга»– Курск, 2016.–99с.
2. Золотов, С. Ю. Проектирование информационных систем [Электронный ресурс] : учебное пособие / С. Ю. Золотов. - Томск : Эль Контент, 2013. - 88 с.
3. Абрамов, Г. В. Проектирование информационных систем [Электронный ресурс] : учебное пособие / Г. В. Абрамов, И. Медведкова, Л. Коробова. - Воронеж : Воронежский государственный университет инженерных технологий, 2012. - 172 с.
4. Стасышин, В. М. Проектирование информационных систем и баз данных [Электронный ресурс] : учебное пособие / В. М. Стасышин. - Новосибирск : НГТУ, 2012. - 100 с.
5. Вендров, А.М. Проектирование программного обеспечения [Текст] : учебник / А.М. Вендров.— М: Финансы и Статистика, 2006. — 352с