

### Лабораторная работа 3.

#### Задание 1. Подготовить программные инструменты для построения функциональных моделей

Для выполнения работы используется программное средство Ramus.

Если программа не установлена на ПК, то самостоятельно произведите установку с сайта разработчика:

<https://github.com/Vitaliy-Yakovchuk/ramus>

<http://ramussoftware.com/>.

Для его работы требуется JDK8

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

#### Задание 2. Ознакомиться с правилами построения IDEF0-моделей (см. ниже).

Дополнительная информация - в файле [Руководство по IDEF0.pdf](#)

#### Методология функционального моделирования работ SADT

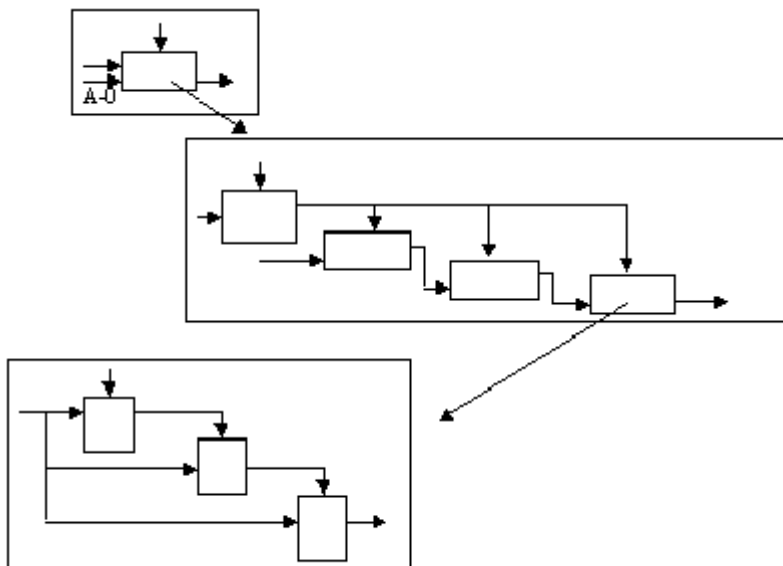
Методология SADT (Structured Analysis and Design Technique - технология структурного анализа и проектирования) разработана Дугласом Т. Россом в 1969-1973 годах. Технология изначально создавалась для проектирования систем более общего назначения по сравнению с другими структурными методами, выросшими из проектирования программного обеспечения. SADT - одна из самых известных и широко используемых методик проектирования. Новое название методики, принятое в качестве стандарта - IDEF0 (Icam DEfinition) - часть программы ICAM (Integrated Computer-Aided Manufacturing - интегрированная компьютеризация производства), проводимой по инициативе ВВС США.

Процесс моделирования в SADT включает сбор информации об исследуемой области, документирование полученной информации и представление ее в виде модели и уточнение модели. Кроме того, этот процесс подсказывает вполне определенный путь выполнения согласованной и достоверной структурной декомпозиции, что является ключевым моментом в квалифицированном анализе системы. SADT уникальна в своей способности обеспечить как графический язык, так и процесс создания непротиворечивой и полезной системы описаний.

В IDEF0 система представляется как совокупность взаимодействующих работ (или функций). Связи между работами определяют технологический процесс или структуру взаимосвязи внутри организации. Модель SADT представляет собой серию диаграмм, разбивающих сложный объект на составные части.

Каждый блок IDEF0-диаграммы может быть представлен несколькими блоками, соединенными интерфейсными дугами, на диаграмме следующего уровня. Эти блоки представляют подфункции (подмодули) исходной функции. Каждый из подмодулей может быть декомпозирован аналогичным образом. Число уровней не ограничивается, зато рекомендуется на одной диаграмме использовать не менее 3 и не более 6 блоков.

**Работы (activity)** обозначают поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты (изображается в виде прямоугольников). Каждая из работ на диаграмме может быть впоследствии декомпозирована.



Как отмечалось выше, в терминах IDEF0 система представляется в виде комбинации блоков и дуг. Блоки представляют функции системы, дуги представляют множество объектов (физические объекты, информация или действия, которые образуют связи между функциональными блоками). Взаимодействие работ с внешним

миром и между собой описывается в виде стрелок или дуг. С дугами связываются метки на естественном языке, описывающие данные, которые они представляют. Дуги показывают, как функции системы связаны между собой, как они обмениваются данными и осуществляют управление друг другом. Дуги могут разветвляться и соединяться. Ветвление означает множественность (идентичные копии одного объекта) или расщепление (различные части одного объекта). Соединение означает объединение или слияние объектов. Пять типов стрелок допускаются в диаграммах:

**Вход (Input)** - материал или информация, которые используются работой для получения результата (стрелка, входящая в левую грань).

**Управление (Control)** - правила, стратегии, стандарты, которыми руководствуется работа (стрелка, входящая в верхнюю грань). В отличие от входной информации управление не подлежит изменению.

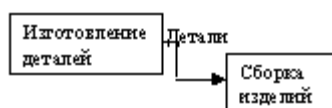
**Выход (Output)** - материал или информация, которые производятся работой (стрелка, исходящая из правой грани). Каждая работа должна иметь хотя бы одну стрелку выхода, т.к. работа без результата не имеет смысла и не должна моделироваться.

**Механизм (Mechanism)** - ресурсы, которые выполняют работу (персонал, станки, устройства - стрелка, входящая в нижнюю грань).

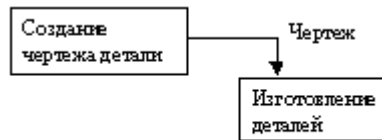
**Вызов (Call)** - стрелка, указывающая на другую модель работы (стрелка, исходящая из нижней грани).

Различают в IDEF0 пять типов связей работ.

**Связь по входу (input-output)**, когда выход вышестоящей работы направляется на вход следующей работы.



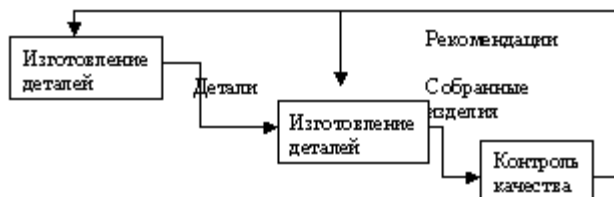
**Связь по управлению (output-control),** когда выход вышестоящей работы направляется на управление следующей работы. Связь показывает доминирование вышестоящей работы.



**Обратная связь по входу (output-input feedback),** когда выход нижестоящей работы направляется на вход вышестоящей. Используется для описания циклов.



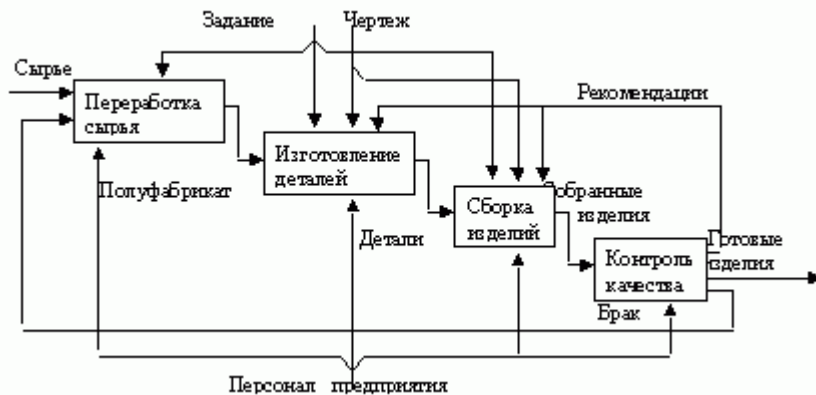
**Обратная связь по управлению (output-control feedback),** когда выход нижестоящей работы направляется на управление вышестоящей. Является показателем эффективности бизнес-процесса.



**Связь выход-механизм (output-mechanism),** когда выход одной работы направляется на механизм другой и показывает, что работа подготавливает ресурсы для проведения другой работы.



Из перечисленных блоков, как из отдельных кирпичиков, строится диаграмма. Пример SADT-диаграммы:



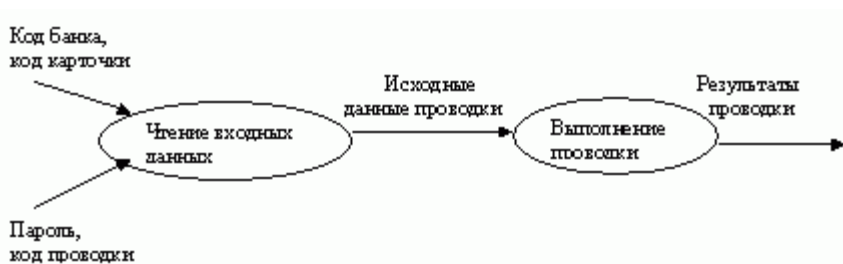
### Диаграммы потоков данных DFD (Data Flow Diagrams)

Диаграммы потоков данных используются для описания движения документов и обработки информации как дополнение к IDEF0. В отличие от IDEF0, где система рассматривается как взаимосвязанные работы и стрелки представляют собой жесткие взаимосвязи, стрелки в DFD показывают лишь то, как объекты (включая данные) движутся от одной работы к другой. DFD отражает функциональные зависимости значений, вычисляемых в системе, включая входные значения, выходные значения и внутренние хранилища данных. DFD - это граф, на котором показано движение значений данных от их источников через преобразующие их процессы к их потребителям в других объектах.

DFD содержит процессы, которые преобразуют данные, потоки данных, которые переносят данные, активные объекты, которые производят и потребляют данные, и хранилища данных, которые пассивно хранят данные.

**Процессы.** Процесс преобразует значения данных. Процессы самого нижнего уровня представляют собой функции без побочных эффектов (примерами таких функций являются вычисление суммы двух чисел, вычисление комиссионного сбора за выполнение проводки с помощью банковской карточки и т.п.). Весь граф потока данных тоже представляет собой процесс (высокого уровня). Про-

цесс может иметь побочные эффекты, если он содержит нефункциональные компоненты, такие как хранилища данных или внешние объекты. На DFD процесс изображается в виде эллипса, внутри которого помещается имя процесса; каждый процесс имеет фиксированное число входных и выходных данных, изображаемых стрелками.



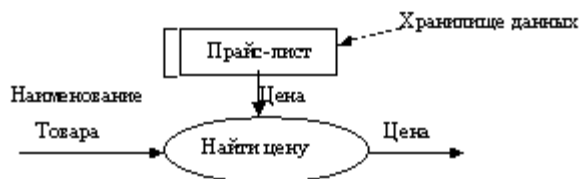
**Потоки данных.** Поток данных соединяет выход объекта (или процесса) с входом другого объекта (или процесса). Он представляет промежуточные данные вычислений. Поток данных изображается в виде стрелки между производителем и потребителем данных, помеченной именами соответствующих данных. Дуги могут разветвляться или сливаться, что означает, соответственно, разделение потока данных на части, либо слияние объектов.

**Активные объекты.** Активным называется объект, который обеспечивает движение данных, поставляя или потребляя их. Активные объекты обычно бывают присоединены к входам и выходам DFD.

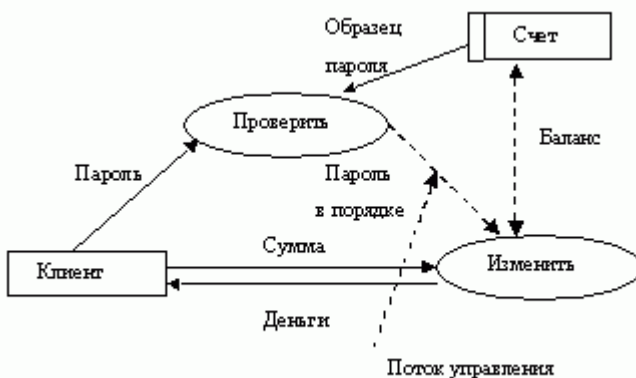


**Хранилища данных.** Хранилище данных - это пассивный объект в составе DFD, в котором данные сохраняются для последующего доступа. Хранилище данных допускает доступ к хранимым в нем данным в порядке, отличном от того, в котором они были туда помещены. Агрегатные хранилища данных, как, например, списки и таб-

лицы, обеспечивают доступ к данным в порядке их поступления, либо по ключам.



**Потоки управления.** DFD показывает все пути вычисления значений, но не показывает в каком порядке значения вычисляются. Решения о порядке вычислений связаны с управлением программой, которое отражается в динамической модели. Эти решения, вырабатываемые специальными функциями, или предикатами, определяют, будет ли выполнен тот или иной процесс, но при этом не передают процессу никаких данных, так что их включение в функциональную модель необязательно. Тем не менее, иногда бывает полезно включать указанные предикаты в функциональную модель, чтобы в ней были отражены условия выполнения соответствующего процесса. Функция, принимающая решение о запуске процесса, будучи включенной в DFD, порождает в DFD поток управления и изображается пунктирной стрелкой.



Первым шагом при построении иерархии DFD является построение контекстных диаграмм. Обычно при проектировании относительно простых информационных систем строится единственная контекстная диаграмма со звездообразной топологией, в центре которой находится так называемый главный процесс, соединенный с при-

емниками и источниками информации, посредством которых с системой взаимодействуют пользователи и другие внешние системы.

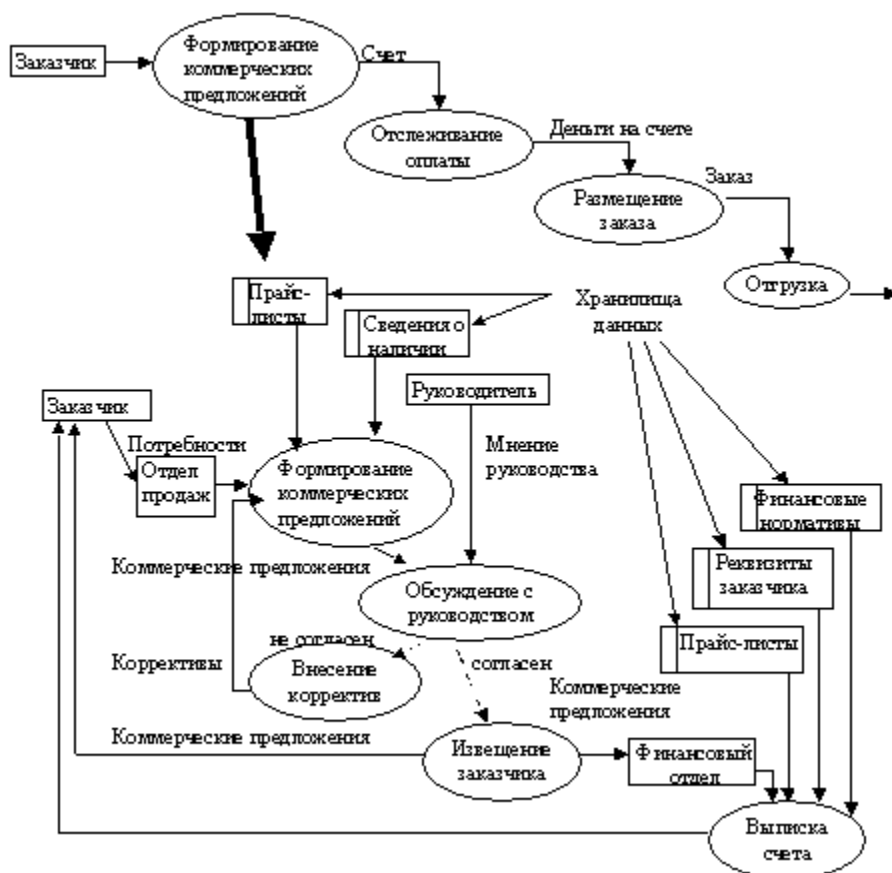
Если же для сложной системы ограничиться единственной контекстной диаграммой, то она будет содержать слишком большое количество источников и приемников информации, которые трудно расположить на листе бумаги нормального формата, и, кроме того, главный единственный процесс не раскрывает структуры распределенной системы.

Для сложных информационных систем строится иерархия контекстных диаграмм. При этом контекстная диаграмма верхнего уровня содержит не главный единственный процесс, а набор подсистем, соединенных потоками данных. Контекстные диаграммы следующего уровня детализируют контекст и структуру подсистем.

При построении иерархии DFD переходить к детализации процессов следует только после определения содержания всех потоков и накопителей данных, которое описывается при помощи структур данных. Структуры данных конструируются из элементов данных и могут содержать альтернативы, условные вхождения и итерации. Условное вхождение означает, что данный компонент может отсутствовать в структуре. Альтернатива означает, что в структуру может входить один из перечисленных элементов. Итерация означает вхождение любого числа элементов в указанном диапазоне. Для каждого элемента данных может указываться его тип (непрерывные или дискретные данные). Для непрерывных данных может указываться единица измерения (кг, см и т.п.), диапазон значений, точность представления и форма физического кодирования. Для дискретных данных может указываться таблица допустимых значений.

Ниже приведена диаграмма потоков данных верхнего уровня с ее последующим уточнением:





### Задание 3. Построить модели деятельности промышленной компании

а) Упражнения 1-2 обязательно, Упражнение 3 по желанию, см. ниже

б) Построить аналогичные модели в выбранной Вами предметной области по теме индивидуального проекта, провести декомпозицию как минимум до второго уровня включительно

---

#### Упражнения 1-3 к Заданию 3.

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

Деятельность компании подразумевает что:

- продавцы принимают заказы клиентов;
- операторы группируют заказы по типам клиентов;
- операторы собирают 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.

#### Упражнение 1. Создание контекстной диаграммы

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

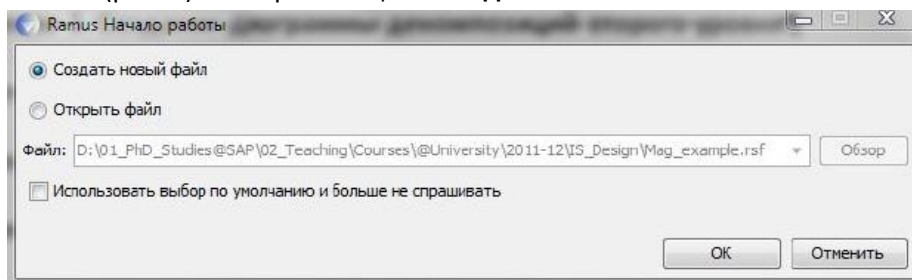


Рис. 1. Диалоговое окно начала работы в Ramus

2. Внесите имя автора, название проекта, название модели и выберите опцию **"IDEFO"**. На следующем шаге укажите, что модель используется **"отделом стратегического планирования и развития"**.

В описании проекта укажите **"Это учебная модель, описывающая деятельность компании"**, перейдите к следующему шагу.

3. Раздел **"классификаторы"** оставьте незаполненным и нажмите **"Дальше"**.



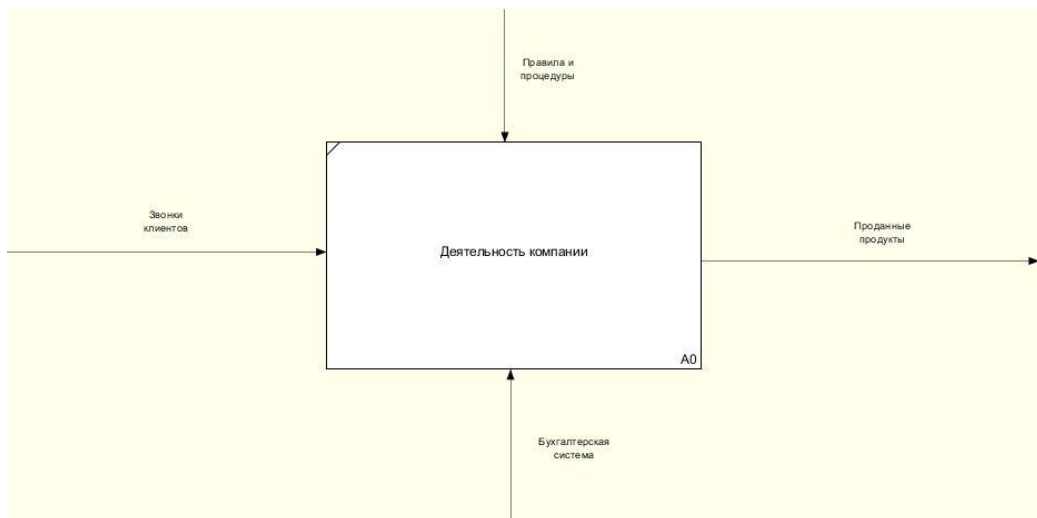
4. В следующем диалоговом окне нажмите "**Окончить**" и перейдите к рабочему интерфейсу программы.
5. Через меню **Диаграмма -> Свойства модели** можно отредактировать мета-данные модели, а именно: название модели, описание, место ее использования.
6. Активируйте окно модели, кликнув на область моделирования. Создайте контекстную диаграмму, нажав на кнопку .
7. Перейдите в режим редактирования контекстной диаграммы, нажав правой кнопкой мыши на объекте и выбрав опцию "**Редактировать активный элемент**". В закладке "**Название**" введите "**Деятельность компании**". Во вкладке "**Описание**" введите "**Текущие бизнес-процессы компании**". Обратите внимание, что вкладка "**Описание**" может быть недоступна в версии RAMUS Educational
8. 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
Бухгалтерская система	Оформление счетов, оплата счетов, работа с заказами	Механизм
Звонки клиентов	Запросы информации, заказы, техническая поддержка и т.д.	Вход
Правила и процедуры	Правила продаж, инструкции по сборке, процедуры тестирования, критерии производительности и т.д.	Управляющее воздействие
Проданные продукты	Настольные и портативные компьютеры	Выход

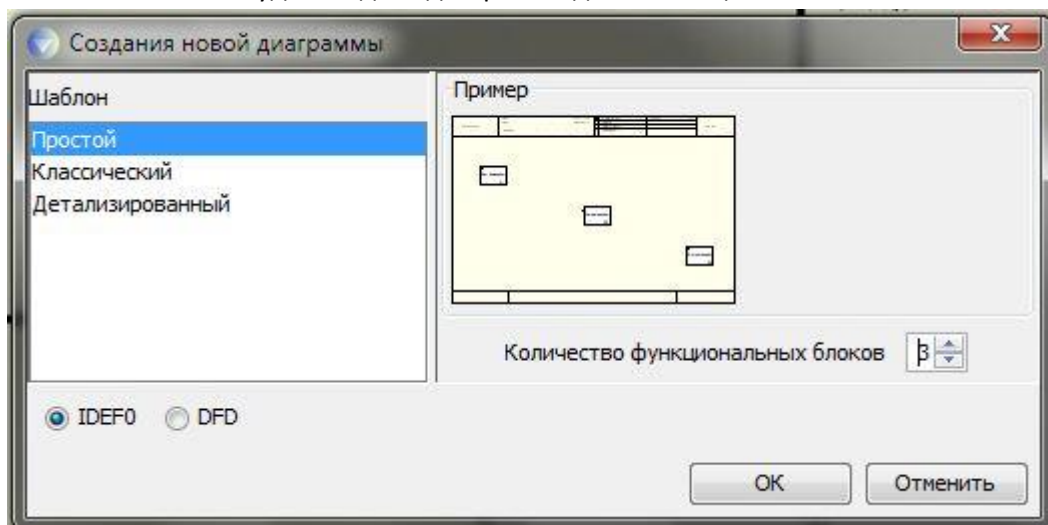
На рис. 2 представлен результат построения контекстной диаграммы по результатам Упражнения 1.



**Рис. 2.** Контекстная диаграмма (результат выполнения Упражнения 1)

## Упражнение 2. Создание диаграммы декомпозиций

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



**Рис. 3.** 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, а также внесите их описание в

соответствующую вкладку на основе данных таблицы (табл. 2). Обратите внимание, что вкладка "**Описание**" может быть недоступна в версии RAMUS Educational.

Таблица 2. Описание работ декомпозиции первого уровня

НАЗВАНИЕ	ОПИСАНИЕ
Продажи и маркетинг	Телемаркетинг, презентации, выставки
Сборка и тестирование компьютеров	Сборка и тестирование настольных и портативных компьютеров
Отгрузка и получение	Отгрузка заказов клиентам и получение компонентов от поставщиков

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

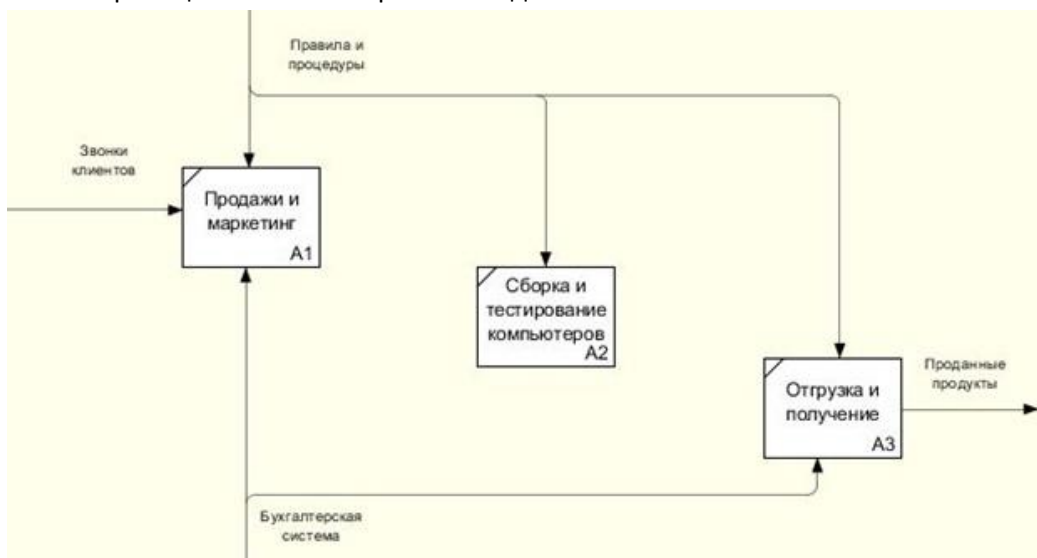


Рис. 4. Связывание граничных стрелок на диаграмме декомпозиции A0

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

5. Правой кнопкой мыши щелкните по ветви стрелки механизма работы "**Продажи и маркетинг**" и переименуйте ее в "**Система оформления заказов**" (рис.5)

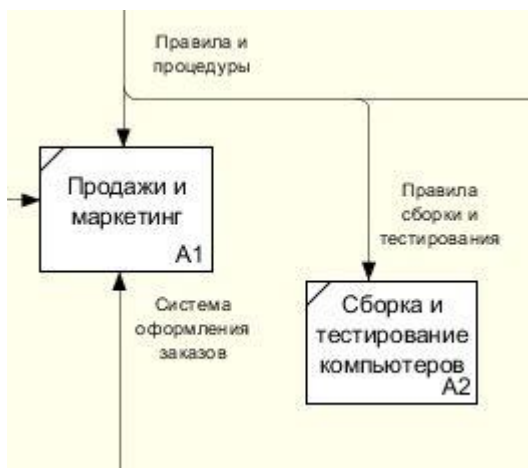


Рис. 5. Присвоение названий ветвям стрелок диаграммы декомпозиции A0

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

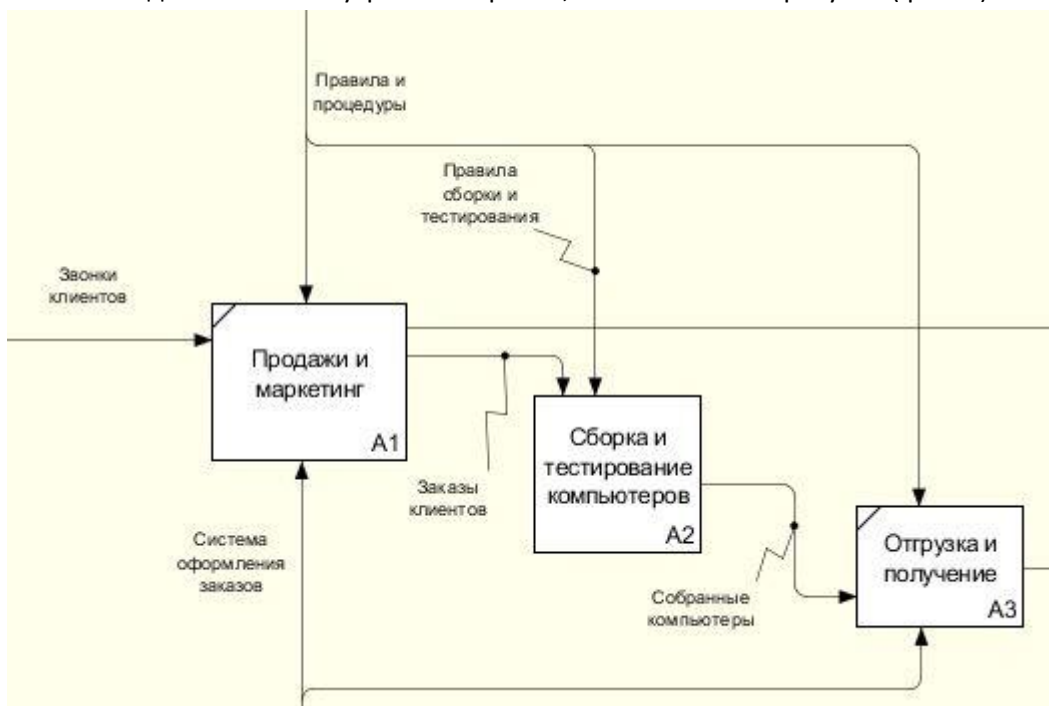
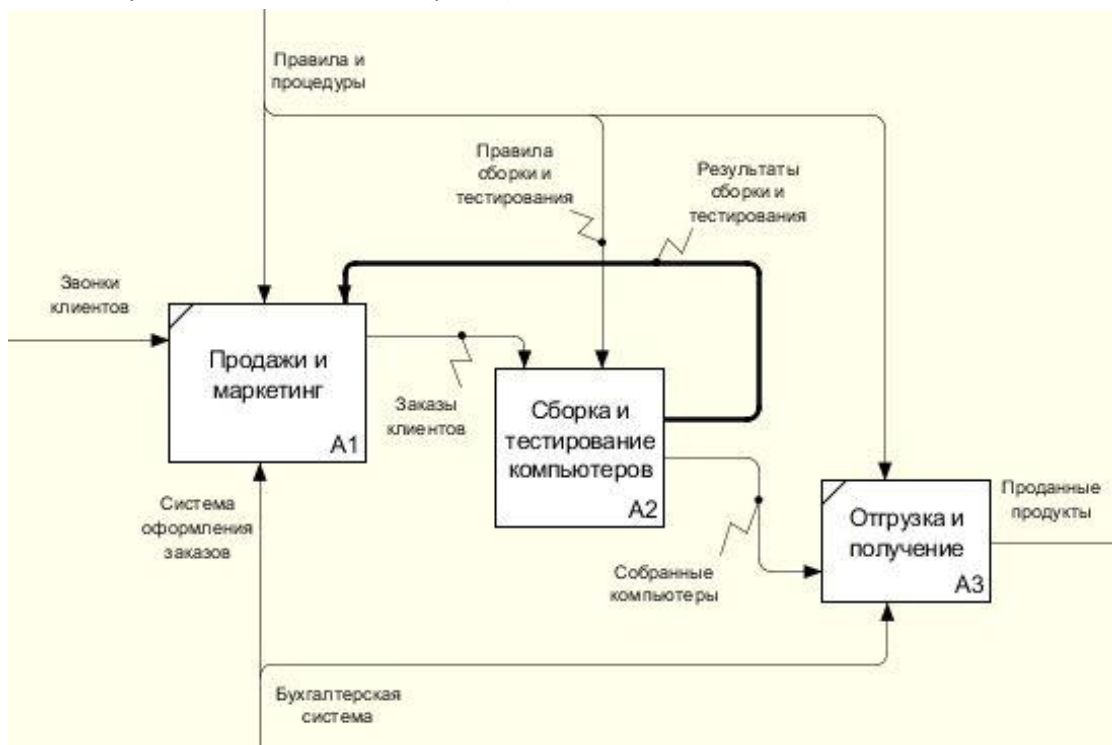


Рис. 6. Внутренние стрелки диаграммы декомпозиции A0

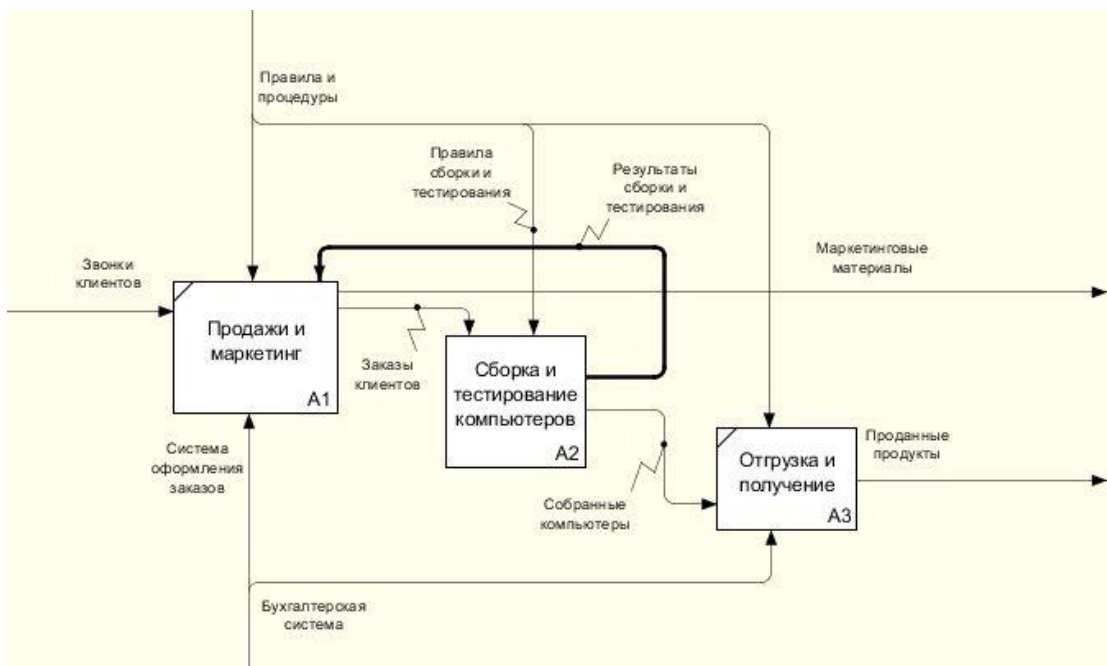
7. Создайте стрелку обратной связи (по управлению) "Результаты сборки и тестирования", идущую от работы "Сборка и тестирование компьютеров" к "Продажи и маркетинг". Измените стиль стрелки - толщину (правая кнопка мыши -> "Редактировать активный элемент" -> вкладка "Линия"). Методом drag&drop возможно переносить стрелки и их названия. При необходимости возможно установить "тильду" (опция контекстного

меню при нажатии на стрелке правой кнопкой мыши) для явной связи стрелки и подписи к ней ( рис. 7)



**Рис. 7.** Результаты редактирования стрелок на диаграмме декомпозиции A0

- Создайте новую граничную стрелку "**Маркетинговые материалы**", выходящую из работы "**Продажи и маркетинг**". Эта стрелка автоматически не попадает на диаграмму верхнего уровня и имеет квадратные скобки у окончания . Щелкните правой кнопкой мыши по квадратным скобкам и выберите в контекстном меню "**Туннель**" (см. рис. 8) одну из двух опций: "**Создать стрелку**" и "**Обозначить туннель круглыми скобками**", в нашем случае - первый вариант.



**Рис. 8.** Результат туннелирования стрелок

### Упражнение 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 тестирует 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 3 и 4 внесите новые работы и стрелки на диаграмму декомпозиции A2.



Таблица 3. Описание функциональных блоков диаграммы декомпозиции А2

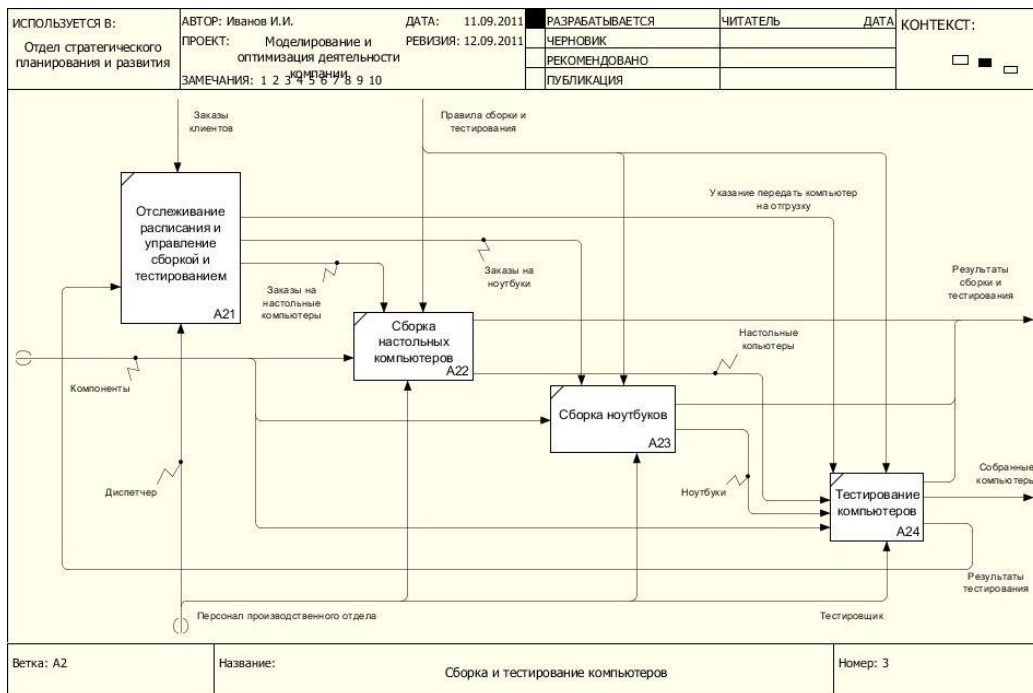
НАЗВАНИЕ ФУНКЦИОНАЛЬНОГО БЛОКА	ОПИСАНИЕ
Отслеживание расписания и управление сборкой и тестирование	Просмотр заказов, установка расписания выполнения заказов, просмотр результатов тестирования, формирования групп заказов на сборку и отгрузку
Сборка настольных компьютеров	Сборка настольных компьютеров 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.

Таблица 4. Описание стрелок диаграммы декомпозиции А2

НАЗВАНИЕ СТРЕЛКИ	НАЧАЛО СТРЕЛКИ	ТИП НАЧАЛА СТРЕЛКИ	ОКОНЧАНИЕ СТРЕЛКИ	ТИП ОКОНЧАНИЯ СТРЕЛКИ
Диспетчер	Персонал производственного отдела	Механизм (ветка стрелки)	Отслеживание расписания и управление сборкой и тестированием	Механизм
Заказы клиентов	Граница диаграммы	Управляющее воздействие	Отслеживание расписания и управление сборкой и тестированием	Управляющее воздействие
Заказы на настольные компьютеры	Отслеживание расписания и управление сборкой и тестированием	Выход	Сборка настольных компьютеров	Управляющее воздействие
Заказы на ноутбуки	Отслеживание расписания и управление сборкой и тестированием	Выход	Сборка компьютеров	Управляющее воздействие
Компоненты	Туннелированная стрелка	Вход	Сборка настольных компьютеров	Вход
			Сборка ноутбуков	Вход
			Тестирование компьютеров	Вход
Настольные компь-	Сборка настольных	Выход	Тестирование	Вход

ютеры	компьютеров		компьютеров	
Ноутбуки	Сборка ноутбуков	Выход	Тестирование компьютеров	Вход
Персонал производственного отдела	Туннелированная стрелка	Механизм	Сборка настольных компьютеров	Механизм
			Сборка ноутбуков	Механизм
Правила сборки и тестирования	Границы диаграммы		Сборка настольных компьютеров Сборка ноутбуков Тестирование компьютеров	Управляющее воздействие
Результаты сборки и тестирования	Сборка настольных компьютеров Сборка ноутбуков Тестирование компьютеров	Выход	Граница диаграммы	Выход
Результаты тестирования	Тестирование компьютеров	Выход	Отслеживание расписания и управление сборкой и тестированием	Вход
Собранные компьютеры	Тестирование компьютеров	Выход	Граница диаграммы	Выход
Тестировщик	Персонал производственного отдела		Тестирование компьютеров	Механизм
Указание передать компьютеры на отгрузку	Отслеживание расписания и управление сборкой и тестированием	Выход	Тестирование компьютеров	Управляющее воздействие

2. Произведите туннелирование и связку граничных стрелок, если это необходимо. Результат выполнения упражнения 3 представлен на рис. 9.





**Рис. 9.** Результат декомпозиции процесса Сборка и тестирование

#### Задание 4. Средствами Ramus построить DFD

а) для процедуры оформления заказа - выполнить *Упражнение 1*, см. ниже

б) построить аналогичные модели в выбранной Вами предметной области по теме индивидуального проекта (по крайней мере для двух деятельностей)

##### *Упражнение 1.*

1. Создайте новую контекстную диаграмму IDEF0 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 "**ОК**" 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 , а также внешнюю ссылку "**Заявки на заказ**", используя кнопку .
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*.

- Связь между некоторыми функциональными объектами и хранилищами данных может быть двунаправленной (исходящая и входящая стрелки).

6. Сверьте построенную Вами модель с моделью на рисунке

