

УО «ПОЛОЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ДОПОЛНИТЕЛЬНЫЕ МАТЕРИАЛЫ

**к лабораторным работам
по дисциплине «Базы данных»
для специальности**

1-40 01 01 Программное обеспечение информационных технологий

НОВОПОЛОЦК 2016

МАТЕРИАЛЫ ПОДГОТОВИЛА:

старший преподаватель кафедры технологий программирования
Бураченко Ирина Брониславовна

1 ЭТАП. ОСНОВЫ РАБОТЫ С CASE-СРЕДСТВОМ ALLFUSION PROCESS MODELER

AllFusion Process Modeler (далее *BPwin*) — CASE-средство для моделирования бизнес-процессов, позволяющая создавать диаграммы в нотации *IDEF0*, *IDEF3*, *DFD*. В процессе моделирования *BPwin* позволяет переключиться с нотации *IDEF0* на любой ветви модели на нотацию *IDEF3* или *DFD* и создать смешанную модель. *BPwin* поддерживает функционально-стоимостной анализ (ABC).

Работа с программой начинается с создания новой модели, для которой нужно указать имя и тип (рис. 1).

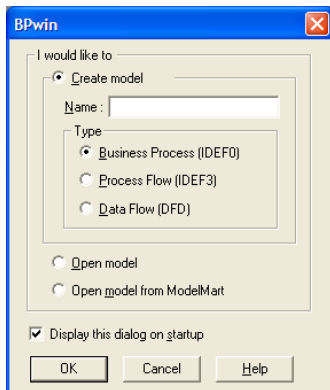


Рис. 1. Создание новой модели

От выбора типа модели зависит в каких нотациях можно производить декомпозицию работ. Так, если выбрать тип *Business Process (IDEF0)*, то в созданной модели можно производить декомпозицию работ в нотациях *IDEF0*, *IDEF3* и *DFD*; если выбран тип *Data Flow (DFD)* – в нотациях *DFD* и *IDEF3*; если же выбран тип *Process Flow (IDEF3)* – то только в нотации *IDEF3*.

После ввода имени модели и выбора ее типа *BPwin* сразу предложит задать параметры модели (рис. 2):

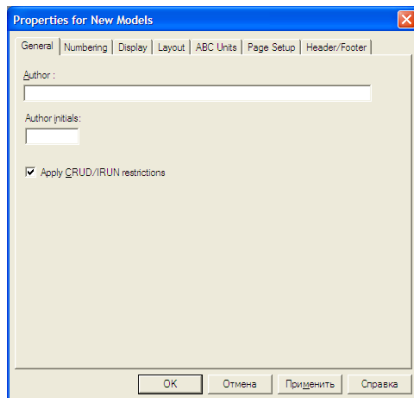


Рис. 2. Окно задания свойств модели

General – автор модели и его инициалы;

Numbering – формат нумерации работ и диаграмм и порядок ее отображения на диаграммах;

Display – список элементов отображения на диаграммах;

Layout – параметры расположения;

ABC Units – единицы функционально-стоимостного анализа;

Page Setup – параметры страницы;

Header/Footer – параметры верхнего и нижнего колонтитула.

После задания свойств модели появляется главное окно программы (рис. 3), состоящее из трех основных частей:

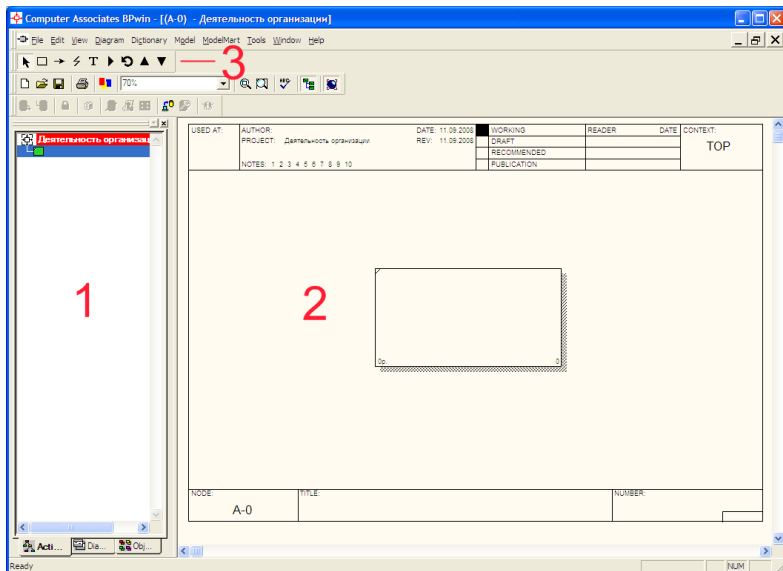


Рис. 3. Главное окно программы

1. Обзорщик модели (*Model Explorer*) – отображает структуру модели (имеющиеся диаграммы и их иерархию).
2. Основная часть – в ней отображаются диаграммы, с которыми ведется работа.
3. Панели инструментов, из которых наибольший интерес представляет панель инструментов *Model Toolbox*.

Примечание. В созданной модели с настройками по умолчанию некорректно отображаются русские символы. Чтобы устранить этот недостаток, необходимо подкорректировать используемые в модели шрифты. Для этого в меню *Model* → *Default Fonts* необходимо последовательно пройти по всем пунктам (рис. 4), выбрать в выпадающем списке *Script* значение *кириллический* и поставить галочку *Change all occurrences* (рис. 5).

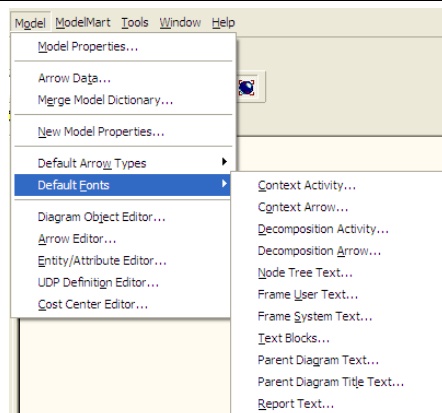


Рис. 4. Пункты меню, отвечающие за настройки шрифта

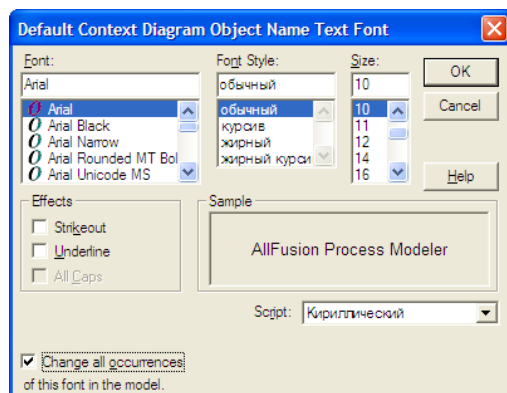













Рис. 5. Параметры шрифта

ПАНЕЛЬ ИНСТРУМЕНТОВ MODEL TOOLBOX

Данная панель инструментов отвечает за создание разнообразных графических элементов модели. В зависимости от типа текущей диаграммы набор кнопок на ней меняется.

Таблица 1. Вид и назначение кнопок Model Toolbox

Вид кнопки	Название кнопки	Назначение кнопки
	Pointer Tool	Превращает курсор в стрелку указателя для того, чтобы можно было выделять объекты
– IDEF0 – DFD – IDEF3	Activity Box Tool	Добавление на диаграмму новой работы

	Precedence Arrow Tool	Добавление на диаграмму новой стрелки
	Squiggle Tool	Связывание названия стрелки с самой стрелкой
	Text Tool	Добавление на диаграмму текста
	Diagram Dictionary Editor	Вызов окна менеджера диаграмм для просмотра имеющихся диаграмм по типам и переход к выбранной
	Go to Sibling Diagram	Переход между стандартной диаграммой, деревом узлов и FEO диаграммой
	Go to Parent Diagram	Переход к родительской диаграмме
	Go to Child Diagram	Переход к дочерней диаграмме
 – DFD	External Reference Tool	Добавление на диаграмму внешней сущности
 – DFD	Data store Tool	Добавление на диаграмму хранилища данных
 – IDEF3	Junction Tool	Добавление на диаграмму перекрестка
 – IDEF3	Referent Tool	Добавление на диаграмму объекта ссылки

Созданная модель уже содержит контекстную диаграмму с единственной работой («черный ящик») в той нотации, которая была выбрана на этапе создания модели. Теперь необходимо дать этой работе название и при необходимости задать ее свойства. Для этого нужно вызвать окно свойств работы, дважды щелкнув по ней мышью (рис. 6).

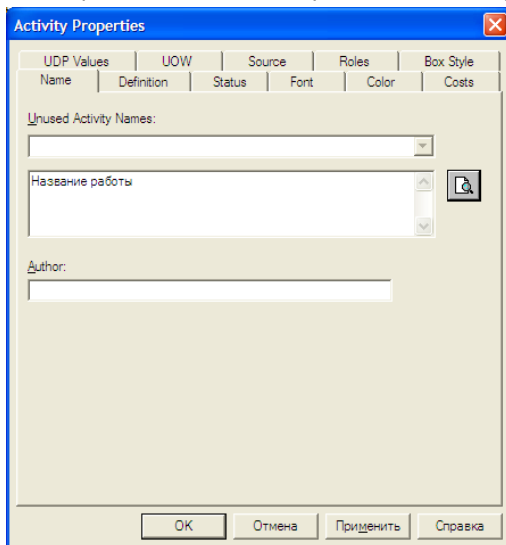


Рис. 6. Окно свойств работы

Далее необходимо разместить на диаграмме стрелки. Для этого следует нажать на *Model Toolbox* кнопку *Precedence Arrow Tool* (курсор примет форму крестика со стрелкой), щелкнуть по тому месту, откуда стрелка должна выходить и затем щелкнуть по тому месту, куда стрелка должна заходить (*BPwin* подсветит эти места при наведении на них курсора).

Для задания названия стрелки нужно нажать на *Model Toolbox* кнопку *Pointer Tool* и затем дважды щелкнуть по стрелке. В появившемся окне *Arrow Properties* название работы вводится в поле *Arrow Name* или выбирается из списка имеющихся названий стрелок.

После размещения стрелок на диаграмме можно проводить декомпозицию ее работ. Для этого следует нажать на *Model Toolbox* кнопку *Go to Child Diagram* и затем щелкнуть по работе, которую нужно декомпозировать. Появится окно, в котором необходимо выбрать в какой нотации проводить декомпозицию и количество дочерних работ (рис. 7).

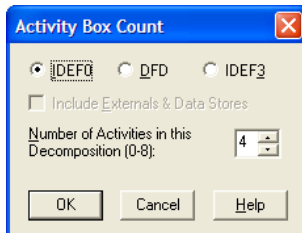


Рис. 7. Создание дочерней диаграммы

После создания дочерней диаграммы *BPwin* автоматически создаст указанное число работ и разместит граничные стрелки по краям диаграммы. Далее следует связать граничные стрелки со входами работ (при необходимости можно добавить новые граничные стрелки) и связать работы между собой. Дальнейшая декомпозиция работ проводится аналогичным образом.

ЭТАП 2. НАЧАЛО МОДЕЛИРОВАНИЯ. ПОСТРОЕНИЕ КОНТЕКСТНОЙ ДИАГРАММЫ В НОТАЦИИ IDEF0

Целью данной и большинства последующих работ является моделирование деятельности выбранного предприятия. Для этого будут применяться методологии:

- IDEF0 – методология функционального моделирования;
- IDEF3 – методология описания процессов;
- DFD – методология моделирования потоков данных;
- IDEF1X – методология моделирования данных.

Диаграммы в первых трех методологиях будут создаваться с помощью CASE-средства *AllFusion Process Modeler*, *IDEF1X* – с помощью *AllFusion ERwin Data Modeler*. Каждая диаграмма в нотациях *IDEF0*, *IDEF3*, *DFD* предназначена для описания одного или нескольких бизнес-процессов.

Бизнес-процесс – это устойчивая, целенаправленная совокупность взаимосвязанных видов деятельности (последовательность работ), которая по определенной технологии преобразует входы в выходы, представляющие ценность для потребителя.

Результатом моделирования бизнес-процессов является модель бизнес-процессов, которая относится к одному из трех типов:

- модель *AS-IS* (как есть) – модель текущей организации бизнес-процессов предприятия;
- модель *TO-BE* (как будет) – модель идеальной организации бизнес-процессов;
- модель *SHOULD-BE* (как должно бы быть) – идеализированная модель, не отражающая реальную организацию бизнес-процессов предприятия.

В примере будет создаваться модель *AS-IS*.

Перед началом построения диаграмм необходимо изучить выбранную предметную область. В этой и последующих работах в качестве предметной области будет выступать вымышленное предприятие по сборке и продаже настольных компьютеров и ноутбуков. Компания не производит комплектующие самостоятельно, а только собирает и тестирует компьютеры. Основные процедуры в компании:

- продавцы принимают заказы клиентов;
- сотрудники группируют заказы по типам компьютеров;
- сотрудники собирают и тестируют компьютеры;
- сотрудники упаковывают компьютеры согласно заказам;
- кладовщик отгружает клиентам заказы;
- снабженцы заказывают и доставляют комплектующие, необходимые для сборки.

Компания использует купленную бухгалтерскую информационную систему, которая позволяет оформить заказ, счет и отследить платежи по счетам.

Построение модели какой-либо системы в методологии IDEF0 начинается с определения **контекста моделирования**, который включает в себя субъекта моделирования, цель моделирования и точку зрения на модель.

Под **субъектом** понимается сама система, при этом необходимо точно установить, что входит в систему, а что лежит за ее пределами, другими словами, необходимо определить, что в дальнейшем будет рассматривать как компоненты системы, а что как внешнее воздействие.

Цель моделирования. Модель не может быть построена без четко сформулированной цели. Цель должна отвечать на следующие вопросы:

1. Почему этот процесс должен быть смоделирован?
2. Что должна показывать модель?
3. Что может получить читатель?

Точка зрения. Не смотря на то, что при построении модели учитываются мнения различных людей, модель должна строиться с единой точки зрения. Точку зрения можно представить как взгляд человека, который видит систему в нужном для моделирования аспекте. Точка зрения должна соответствовать цели моделирования. В течении моделирования важно оставаться на выбранной точке зрения.

В данной работе субъектом будет выступать само предприятие, а именно процессы, происходящие внутри него; цель моделирования – воспроизвести бизнес-процессы, происходящие на предприятии (модель *AS-IS*); точка зрения – с позиции директора как лица, знающего структуру предприятия в целом.

После определения контекста моделирования можно приступить к построению контекстной диаграммы (называемой еще «черным ящиком»). Данный тип диаграммы позволяет показывать, что подается на вход работы и что является результатом работы, без детализации ее составляющих. Данная диаграмма содержит только одну работу, которая будет представлять всю деятельность предприятия в целом (рис. 8).

USED AT:	AUTHOR: Fastowsky G. Eduard	DATE: 12.09.2007	WORKING	READER	DATE	CONTEXT:
	PROJECT: Computer Firm	REV: 01.02.2010	DRAFT			TOP
	NOTES: 1 2 3 4 5 6 7 8 9 10		RECOMMENDED			
			PUBLICATION			

Деятельность предприятия
по сборке и продаже
компьютеров и ноутбуков

Op. A0

NODE: A-0	TITLE: Деятельность предприятия по сборке и продаже ноутбуков	NUMBER:
--------------	---	---------

Рис. 8. Контекстная диаграмма

Любая *IDEFO* диаграмма состоит из прямоугольников, называемых работами (*activity*), и стрелок (*arrow*). Работа представляет собой некоторую конкретную функцию в рамках рассматриваемой системы. По требованиям стандарта название каждой работы должно быть выражено отглагольным существительным (например, «Изготовление детали», «Оформление заказа» и т.д.). Каждая из четырех сторон прямоугольника имеет свое определенное значение (рис. 9):



Рис. 9. Работа в *IDEFO*

- Вход – это потребляемая или изменяемая работой информация или материал;
- Выход – информация или материал, которые производятся работой;
- Управление – процедуры, правила, стратегии или стандарты, которыми руководствуется работа;
- Механизмы – ресурсы, которые выполняют работу (например, сотрудники, оборудование, устройства и т.д.).

Для рассматриваемого предприятия *входными стрелками* будут:

- Заказы клиентов – список компьютеров и их конфигурация, которые клиент желает приобрести;
- Комплектующие от поставщиков – комплектующие, полученные от поставщиков, из которых собираются компьютеры и ноутбуки.

Выходные стрелки:

- Готовая продукция – собранные компьютеры и ноутбуки;
- Заказы поставщикам – список комплектующих, которые предприятие закупает у поставщиков;
- Оплата за комплектующие – деньги поставщикам за комплектующие;
- Маркетинговые материалы – прайс-листы, рекламки и т.п.

Стрелки управления:

- Законодательство – различные законодательные документы, которыми руководствуется предприятие в процессе своей деятельности;
- Правила и процедуры – различные правила и процедуры, которыми руководствуется предприятие в процессе своей деятельности (например, правила сборки и тестирования компьютеров, процедура общения с клиентами и т.п.).

Стрелки механизмов:

- Бухгалтерская система;
- Персонал.

Итоговая контекстная диаграмма имеет вид (рис. 10):

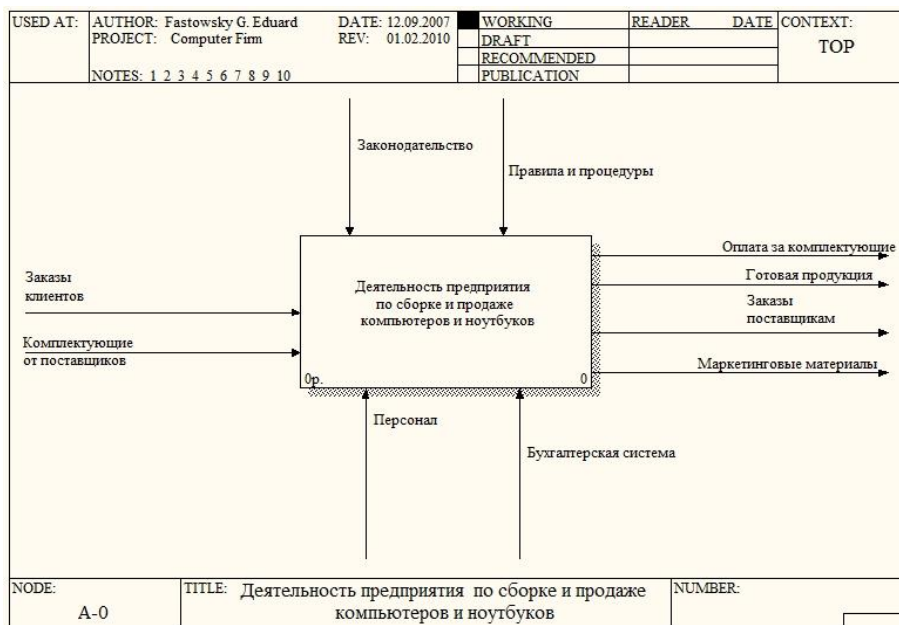


Рис. 10. Итоговая контекстная диаграмма

ЭТАП 3. ПОСТРОЕНИЕ ДИАГРАММЫ ДЕКОМПОЗИЦИИ ВТОРОГО УРОВНЯ В НОТАЦИИ IDEF0

На предыдущем этапе была построена контекстная диаграмма, содержащая только одну работу, которая описывает деятельность предприятия в целом, без детализации составляющих этой работы. Далее будут построены диаграммы декомпозиции первого и второго уровней в нотации *IDEF0*.

Декомпозиция – это разделение сложного объекта, системы, задачи на составные части, элементы.

С помощью диаграммы декомпозиции первого уровня покажем, из каких более мелких работ состоит работа «Деятельность предприятия по сборке и продаже компьютеров и ноутбуков». В данной работе были выделены следующие дочерние работы:

Таблица 2. *Виды дочерних работ*

Управление	Данная работа включает в себя общее управление предприятием, финансами, кадрами, бухгалтерию и т.п.
Продажи и маркетинг	Работа с клиентами, презентации, выставки, реклама, маркетинговые исследования и т.д.
Сборка и тестирование компьютеров	Сборка и тестирование настольных компьютеров и ноутбуков
Отгрузка и снабжение	Снабжение предприятия необходимыми комплектующими, хранение и отгрузка готовой продукции

После создания дочерней диаграммы первым действием соединим граничные стрелки с работами. Стрелку «Заказы клиентов» соединим с работой «Продажи и маркетинг», стрелку «Комплектующие от поставщиков» – с «Отгрузка и снабжение». Выходом работы «Управление» будет «Оплата за комплектующие», выходом «Продажи и маркетинг» – «Маркетинговые материалы». Стрелки «Заказы поставщикам» и «Готовая продукция» – выход работы «Отгрузка и снабжение».

Стрелка «Персонал» будет являться входом механизма всех четырех работ, а стрелка «Бухгалтерская система» – работ «Продажи и маркетинг» и «Отгрузка и получение». Стрелка «Правила и процедуры» будет входом управления всех четырех работ.

Любую ветвь стрелки также можно декомпонировать и дать ей свое название. Покажем это на примере ветки стрелки «Бухгалтерская система» для работы «Продажи и маркетинг». Назовем ее «Система оформления заказа». В AllFusion Process Modeler для более четкого указания какое название к какой стрелке относится, существуют несколько механизмов, одним из которых является *Squiggle* – стрелка в виде молнии, соединяющая название со стрелкой. Воспользуемся им для, чего щелкнем правой кнопкой по названию стрелки и выберем в выпадающем меню соответствующий пункт.

На данном этапе построения диаграммы выяснилось, что мы не учли такой важный фактор, как деньги, которые клиенты дают за готовую продукцию. Деньги клиентов – это вход работы «Деятельность предприятия по сборке и продаже компьютеров и ноутбуков». Добавим эту стрелку на диаграмму декомпозиции.

Если по каким-то причинам граничную стрелку дочерней диаграммы не следует показывать (например, на данной диаграмме она является несущественной, или чтоб не загромождать диаграмму), то ее можно просто удалить. Удалим стрелку «Законодательство».

Результат всех перечисленных действий показан на рис. 11.

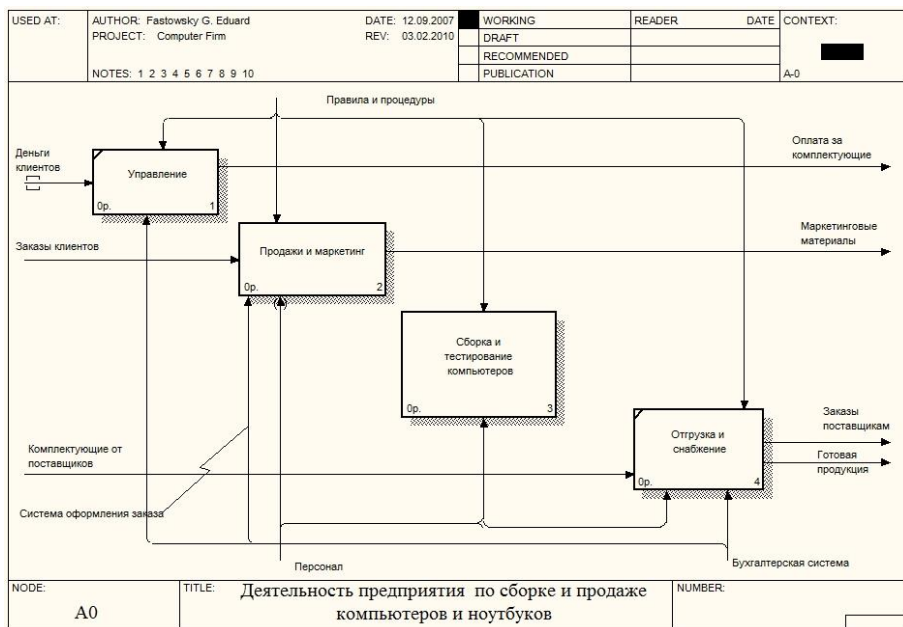


Рис. 11. Результат декомпозиции диаграммы

Если посмотреть на стрелку «Денег клиентов» диаграммы декомпозиции и на стрелку «Законодательство» контекстной диаграммы, то видно, что они окружены небольшими квадратными скобками. Это означает, что данная граничная стрелка является новой на диаграмме и ее нет на дочерней диаграмме (как в случае со стрелкой «Законодательство»), или же данная стрелка является новой на дочерней диаграмме и ее нет на родительской (как в случае со стрелкой «Денег клиентов»). От стрелок с квадратными скобками необходимо избавляться. Для этого есть два пути:

- добавить их на родительскую или дочернюю диаграмму, т.е. сделать граничной;
- затоннелировать.

Чтоб добавить такую стрелку на другую диаграмму или затоннелировать, нужно щелкнуть по квадратным скобкам правой кнопкой мыши и выбрать пункт меню «Arrow Tunnel». В появившемся окне следует выбрать один из двух вариантов: *Resolve it to border arrow* – сделать стрелку граничной, *Change it to resolved rounded tunnel* – затоннелировать стрелку. В данном случае мы решили обе стрелки затоннелировать (рис. 12 и рис. 13).

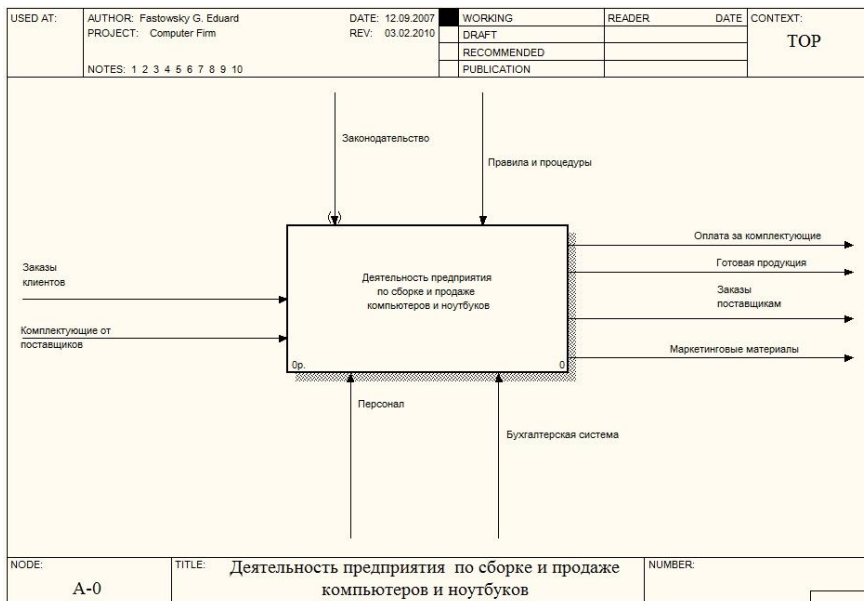


Рис. 12. Контекстная диаграмма с затоннелированной граничной стрелкой

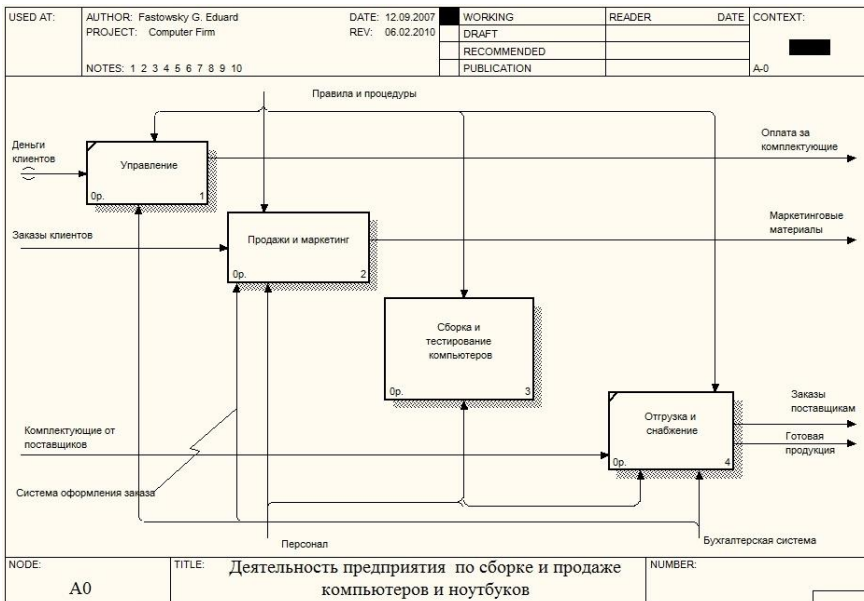


Рис. 13. Диаграмма декомпозиции с затоннелированной граничной стрелкой

После соединения граничных стрелок с работами следующим шагом соединим работы между собой. Поскольку работа «Управление» включает в себя общее управление предприятием, то одним из ее результатов будет являться «Управляющая информация», поступающая на вход управления всех остальных работ.

Работа «Продажи и маркетинг» получает на входе заказы клиентов (т.е. количество компьютеров и их конфигурация), информацию о которых она передает работе «Сборка и тестирование компьютеров» в качестве управляющей информации.

Работе «Сборка и тестирование» для своего функционирования необходимы комплектующие, которые она заказывает у работы «Отгрузка и снабжение» (выходная стрелка «Список необходимых комплектующих»). Собранные компьютеры она также передает работе «Отгрузка и снабжение» (выходная стрелка «Собранные компьютеры»). Информация о результатах сборки и тестирования необходима работе «Продажи и маркетинг» (выходная стрелка «Результаты сборки и тестирования»).

Результатом работы «Отгрузка и снабжение» будут необходимые комплектующие, которые поступают на вход работы «Сборка и тестирование компьютеров». Управление любого предприятия должно знать, что происходит на предприятии, чем занимается каждое подразделение и каковы результаты их работы, т.е. любая работа в идеале должна отчитываться о результатах своей деятельности перед управлением. Создадим стрелки выходов работ «Продажи и маркетинг», «Сборка и тестирование компьютеров» и «Отгрузка и снабжение» и соединим их со входом управления работы «Управление». «лоло»

Результат соединения работ между собой показан на рисунке 14:

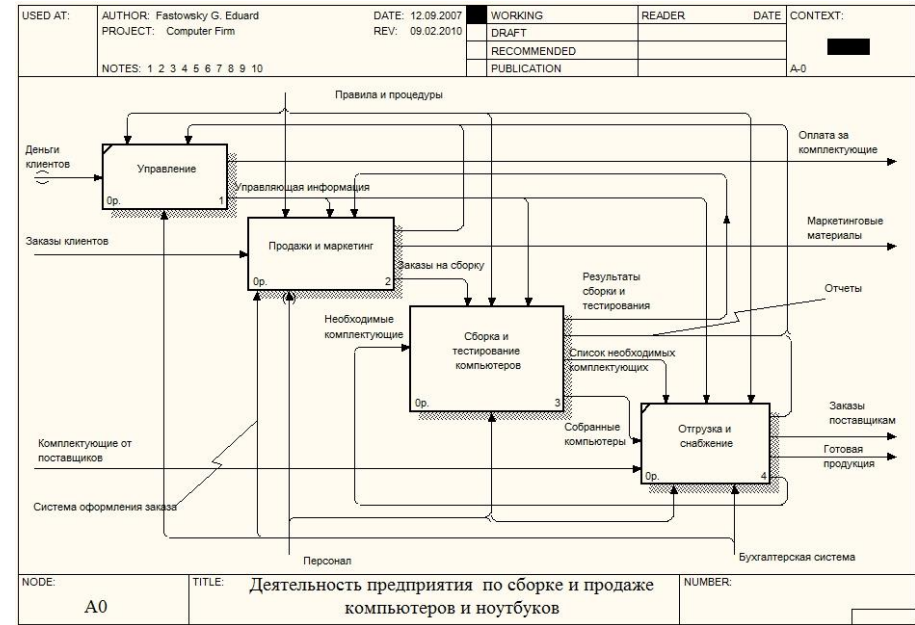


Рис. 14. Результат соединения работ

Если на диаграмме присутствует много работ и стрелок, то бывает затруднительно читать диаграмму. Для облегчения изучения диаграммы отдельные стрелки можно визуально выделить. Для зрительного выделения стрелки, соединяющей две работы, есть несколько механизмов:

- задать толщину стрелки;
- поменять цвет стрелки;
- добавить на стрелку дополнительные наконечники.

Толщина и цвет стрелки задаются в окне свойств стрелки, вызываемое двойным щелчком по стрелке. Вкладка «*Style*» отвечает за стиль стрелки, в том числе и за ее толщину («*Thickness*»), вкладка «*Color*» - за ее цвет. Для добавления на стрелку дополнительных наконечников следует щелкнуть правой кнопкой по стрелке и выбрать пункт меню «*Extra Arrowhead*».

Модифицируем диаграмму, визуально выделив некоторые стрелки. Итоговая диаграмма декомпозиции показана на рисунке 15:

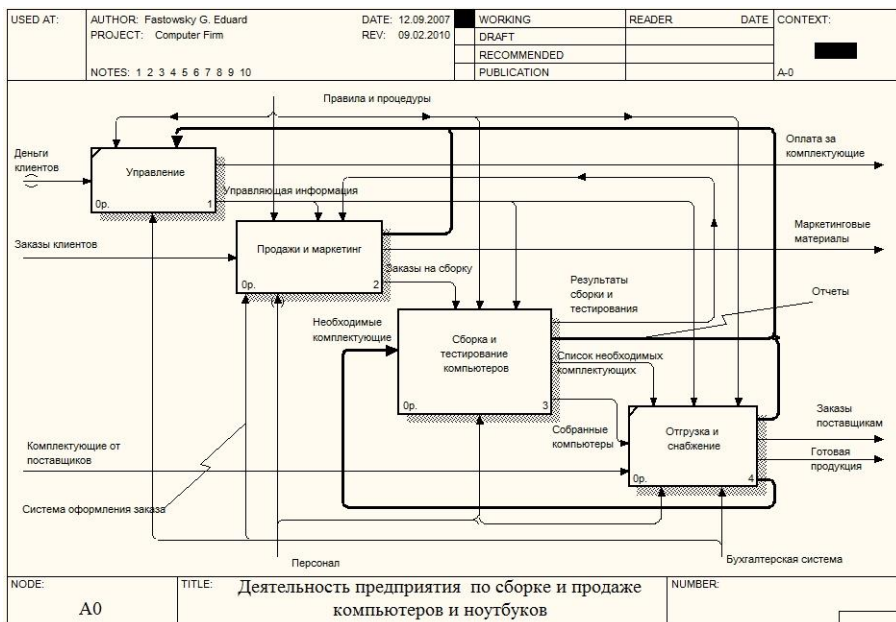


Рис. 15. Итоговая диаграмма декомпозиции первого уровня

ЭТАП 4. ПОСТРОЕНИЕ ДИАГРАММЫ ДЕКОМПОЗИЦИИ СЛЕДУЮЩЕГО УРОВНЯ В IDEF0

На данном этапе построим еще одну диаграмму декомпозиции в нотации IDEF0 – декомпозицию работы «Сборка и тестирование компьютеров» диаграммы A0.

В результате проведения экспертизы получена следующая информация.

1. Производственный отдел получает заказы клиентов от отдела продаж по мере их поступления.
2. Диспетчер координирует работу сборщиков, сортирует заказы, группирует их и дает указание на отгрузку компьютеров, когда они готовы.
3. Каждые 2 часа диспетчер группирует заказы – отдельно для настольных компьютеров и ноутбуков – и направляет на участок сборки.

4. Сотрудники участка сборки собирают компьютеры согласно спецификациям заказа и инструкциям по сборке. Когда группа компьютеров, соответствующая группе заказов, собрана, она направляется на тестирование.
5. Тестировщики тестируют каждый компьютер и в случае необходимости заменяют неисправные компоненты. Тестировщики направляют результаты тестирования диспетчеру, который на основании этой информации принимает решение о передаче компьютеров, соответствующих группе заказов, на отгрузку.

В данной работе мы выделили четыре дочерних работы: *«Отслеживание расписания и управление сборкой и тестированием»*, *«Сборка настольных компьютеров»*, *«Сборка ноутбуков»* и *«Тестирование компьютеров»*. Как и в предыдущей работе начнем с соединения граничных стрелок с работами.

Стрелка *«Необходимые комплектующие»* – это вход работ *«Сборка настольных компьютеров»* и *«Сборка ноутбуков»*.

Стрелки управления *«Управляющая информация»* и *«Заказы на сборку»* соединим с работой *«Отслеживание расписания и управление сборкой и тестированием»*, поскольку именно данная работа управляет всем процессом сборки и тестирования, а стрелку управления *«Правила и процедуры»* – с остальными тремя работами.

Персонал принимает участие во всех выделенных дочерних работах, поэтому заводим стрелку *«Персонал»* на вход механизма всех работ (при этом указав, что в первой работе участвует диспетчер, а в четвертой – тестировщик).

Список необходимых комплектующих – это один из результатов работ *«Сборка настольных компьютеров»* и *«Сборка ноутбуков»*. Результаты сборки и тестирования – это выходы работ *«Сборка настольных компьютеров»*, *«Сборка ноутбуков»* и *«Тестирование компьютеров»*. Компьютеры считаются собранными после того, как они успешно прошли тестирование, поэтому стрелка выхода *«Собранные компьютеры»* – выход работы *«Тестирование компьютеров»*. Различные отчеты формирует работа *«Отслеживание расписания и управление сборкой и тестированием»*.

Результат проделанных операций показан на рисунке 16.

После соединения граничных стрелок с работами следующим шагом соединим работы между собой. Поступающие заказы на сборку сортируются диспетчером, после чего они поступают на вход управления работ *«Сборка настольных компьютеров»* и *«Сборка ноутбуков»* (стрелки *«Заказы на настольные компьютеры»* и *«Заказы на ноутбуки»*, соответственно). Когда компьютеры собраны, диспетчер дает указание на их отгрузку (стрелка *«Указание передать компьютеры на отгрузку»*).

Собранные компьютеры (выходы работ *«Сборка настольных компьютеров»* и *«Сборка ноутбуков»*) должны быть протестированы, для чего они должны поступать на вход работы *«Тестирование компьютеров»* – стрелки *«Настольные компьютеры»* и *«Ноутбуки»*.

После тестирования компьютеров отчет (стрелка *«Результаты тестирования»*) направляется диспетчеру, который дает указание отгрузить компьютеры.

Итоговая диаграмма декомпозиции показана на рисунке 17.

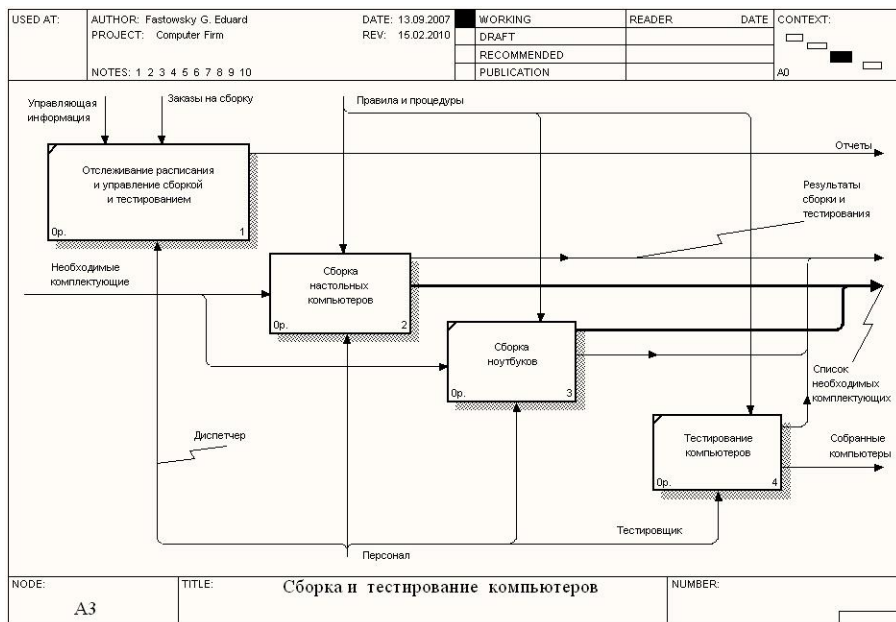


Рис. 16. Результат проделанных операций

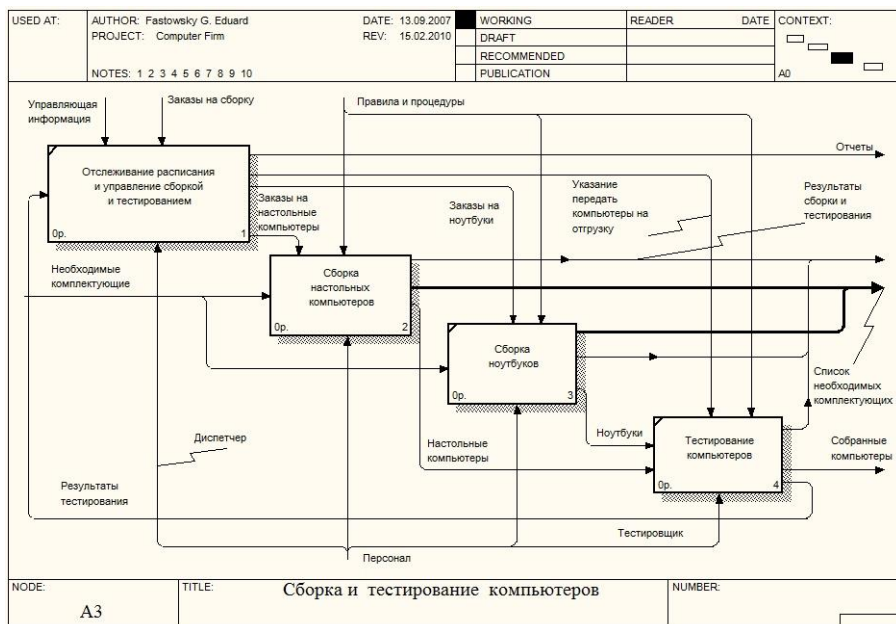


Рис. 17. Итоговая диаграмма

ЭТАП 5. ПОСТРОЕНИЕ ДИАГРАММЫ ДЕКОМПОЗИЦИИ В НОТАЦИИ IDEF3

IDEF3 – методология моделирования, использующая графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектов, являющихся частью этих процессов. *IDEF3* дает возможность аналитикам описать ситуацию, когда процессы выполняются в определенной последовательности, а также описать объекты, участвующие совместно в одном процессе.

Любая *IDEF3-диаграмма* может содержать работы, связи, перекрестки и объекты ссылки.

Работа (Unit of Work, activity). Изображается прямоугольником с прямыми углами (рис. 18) и имеет имя, выраженное отглагольным существительным, обозначающим процесс действия, одиночным или в составе фразы, и номер (идентификатор); другое имя существительное в составе той же фразы обычно отображает основной выход (результат) работы (например, «Изготовление изделия»). Все стороны работы равнозначны. В каждую работу может входить и выходить ровно по одной стрелке.



Рис. 18. Работа IDEF3

Связи. Связи показывают взаимоотношения работ. Все связи в *IDEF3* одноподнаправлены и могут быть направлены куда угодно, но обычно диаграммы *IDEF3* стараются построить так, чтобы связи были направлены слева направо. В *IDEF3* возможны три вида связей:

Таблица 3. Виды связей в IDEF3

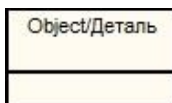
Изображение стрелки	Название	Описание
	Старшая (Precedence) стрелка	сплошная линия, связывающая единицы работ (UOW). Рисуется слева направо или сверху вниз. Показывает, что работа-источник должна закончиться прежде, чем работа-цель начнется
	Потоки объектов (Object Flow)	стрелка с двумя наконечниками, применяется для описания того факта, что объект используется в двух или более единицах работы, например когда объект порождается в одной работе и используется в другой
	Стрелка отношения (Relational Link)	пунктирная линия, используемая для изображения связей между единицами работ (UOW), а также между единицами работ и объектами ссылки. Значение задается аналитиком отдельно для каждого случая

Перекрестки (Junction). Окончание одной работы может служить сигналом к началу нескольких работ, или же одна работа для своего запуска может ожидать окончания нескольких работ. Перекрестки используются для отображения логики взаимодействия стрелок при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы. Различают перекрестки для слияния (*Fan-in Junction*) и разветвления (*Fan-out Junction*) стрелок. Перекресток не может использоваться одновременно для слияния и для разветвления.

Таблица 4. Типы перекрестков

Обозначение	Наименование	Смысл в случае слияния стрелок (Fan-in Junction)	Смысл в случае разветвления стрелок (Fan-out Junction)
	Асинхронное «И» (Asynchronous AND)	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены
	Синхронное «И» (Synchronous AND)	Все предшествующие процессы завершены одновременно	Все следующие процессы запускаются одновременно
	Асинхронное «ИЛИ» (Asynchronous OR)	Один или несколько предшествующих процессов должны быть завершены	Один или несколько следующих процессов должны быть запущены
	Синхронное «ИЛИ» (Synchronous OR)	Один или несколько предшествующих процессов завершены одновременно	Один или несколько следующих процессов запускаются одновременно
	Исключающее «ИЛИ» XOR (Exclusive OR)	Только один предшествующий процесс завершен	Только один следующий процесс запускается

Объект ссылки. Объект ссылки в *IDEF3* выражает некую идею, концепцию или данные, которые нельзя связать со стрелкой, перекрестком или работой. Они используются в модели для привлечения внимания читателя к каким-либо важным аспектам модели. При внесении объектов ссылок помимо имени следует указывать тип объекта ссылки (рис. 19).

**Рис. 19.** Объект ссылки

В данной лабораторной работе необходимо одну из работ, находящихся на диаграммах *IDEF0*, рассмотреть детально с помощью методологии *IDEF3*. При декомпозиции работы *IDEF0* (и *DFD*) нужно учитывать, что стрелки на диаграммах *IDEF0* или *DFD* означают потоки информации или объектов, передаваемых от одной работы к другой. На диаграммах *IDEF3* стрелки могут показывать только последовательность выполнения работ, т.е. они имеют другой смысл, чем стрелки *IDEF0* или *DFD*. Поэтому при декомпозиции работы *IDEF0* или

DFD в диаграмму *IDEF3* стрелки не мигрируют на нижний уровень. Если необходимо показать на дочерней диаграмме *IDEF3* те же объекты, что и на родительских диаграммах *IDEF0* или *DFD*, необходимо использовать объекты ссылки.

Проведем декомпозицию работы *Сборка настольных компьютеров* диаграммы АЗ «Сборка и тестирование компьютеров». Данная работа начинается, когда поступают заказы на сборку. Первым действием проверяется наличие необходимых для сборки комплектующих и заказ со склада отсутствующих. Далее комплектующие подготавливаются для последующей сборки (освобождение от упаковки, снятие заглушек и т.п.). Следующим шагом начинается непосредственно сам процесс сборки: установка материнской платы в корпус и процессора на материнскую плату, установка ОЗУ и винчестера. Данные действия выполняются всегда, независимо от конфигурации компьютера. Далее по желанию клиента могут быть установлены некоторые дополнительные комплектующие – DVD привод, ТВ-тюнер, кард-ридер. На этом сборка компьютера завершается. Следующим шагом идет установка операционной системы. По желанию клиента также может быть установлено дополнительное программное обеспечение. Последним действием составляется отчет о проделанной работе.

Выделим работу *Сборка настольных компьютеров* диаграммы АЗ «Сборка и тестирование компьютеров», нажмем на кнопку «Go to Child Diagram» панели инструментов и выберем нотацию *IDEF3*. Дочерние работы всегда можно добавить на диаграмму в процессе ее построения, поэтому число дочерних работ оставим по умолчанию. При создании дочерней диаграммы *BPM* переносит граничные стрелки родительской работы, их необходимо удалить и заменить на объекты ссылок. Заменяем стрелки «Заказы на настольные компьютеры», «Необходимые комплектующие», «Список необходимых комплектующих», «Настольные компьютеры» и «Результаты сборки» на объекты ссылок - кнопка «Referent» на панели инструментов, в появившемся окне выбрать переключатель «Arrow» и выбрать из списка нужное название (рис. 20):

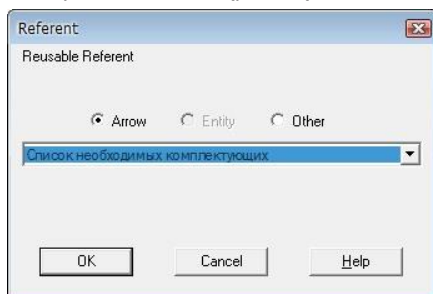


Рис. 20. Добавление объекта ссылки

Далее начинаем располагать на диаграмме работы, отражающие указанные выше действия, выполняемые при сборке компьютеров. Итоговая диаграмма декомпозиции работы в нотации *IDEF3* имеет вид:

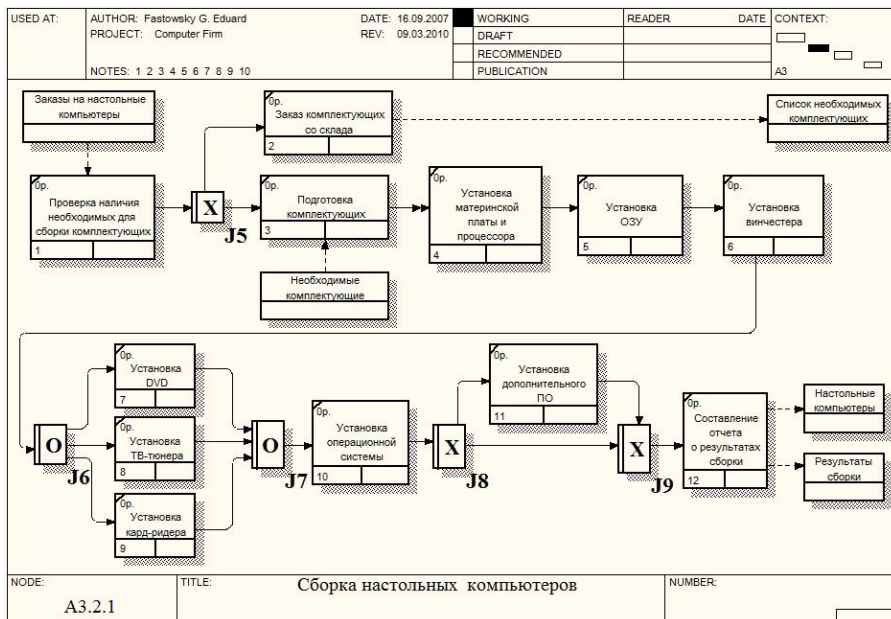


Рис. 21. Диаграмма декомпозиции

Рассмотрим основные особенности этой диаграммы. После проверки наличия необходимых для сборки комплектующих возможно одно из двух действий - или заказ со склада недостающих комплектующих, или, если все комплектующие в наличии, их подготовка. Поэтому мы поставили перекресток разветвления типа «Исключающее ИЛИ». Работы «Подготовка комплектующих» и «Установка материнской платы и процессора» соединены связью «Поток объектов». Тем самым мы показываем, что между этими работами передаются объекты. Все последующие работы соединяются связями «старшая стрелка», поскольку они только показывают последовательность действий над одними и теми же объектами.

После установки винчестера возможна установка DVD привода, ТВ-тюнера, кард-ридера или любая их комбинация. Поэтому мы поставили перекресток разветвления типа «Асинхронное ИЛИ». Такой же перекресток стоит и после завершения этих работ. Далее после установки операционной системы может быть установлено дополнительное ПО, или же сразу формируется отчет, поэтому мы поставили перекресток разветвления типа «Исключающее ИЛИ». За перекрестком разветвления типа «Исключающее ИЛИ» может следовать только такой же перекресток слияния, поэтому перед работой «Составление отчета о результатах сборки» мы поставили такой же.

ЭТАП 6. ПОСТРОЕНИЕ ДИАГРАММЫ ДЕКОМПОЗИЦИИ В НОТАЦИИ DFD

Диаграммы потоков данных (*Data flow diagram, DFD*) используются для описания документооборота и обработки информации. Подобно *IDEFO*, *DFD* представляет моделируемую систему как сеть связанных между собой работ. Их можно использовать как дополнение к модели *IDEFO* для более наглядного отображения текущих операций документооборота в корпоративных системах обработки информации. Главная цель DFD –

показать, как каждая работа преобразует свои входные данные в выходные, а также выявить отношения между этими работами. Любая DFD-диаграмма может содержать работы, внешние сущности, стрелки (потоки данных) и хранилища данных.

Работы. Работы изображаются прямоугольниками с закругленными углами (рис. 22), смысл их совпадает со смыслом работ *IDEF0* и *IDEF3*. Так же как работы *IDEF3*, они имеют входы и выходы, но не поддерживают управления и механизмы, как *IDEF0*. Все стороны работы равнозначны. В каждую работу может входить и выходить по несколько стрелок.



Рис. 22. Работа в DFD

Внешние сущности. Внешние сущности изображают входы в систему и/или выходы из нее. Одна внешняя сущность может одновременно предоставлять входы (функционируя как поставщик) и принимать выходы (функционируя как получатель). Внешняя сущность представляет собой материальный объект, 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. Внешние сущности изображаются в виде прямоугольника с тенью и обычно располагаются по краям диаграммы (рис. 23).



Рис. 23. Внешняя сущность в DFD

Стрелки (потоки данных). Стрелки описывают движение объектов из одной части системы в другую (отсюда следует, что диаграмма DFD не может иметь граничных стрелок). Поскольку все стороны работы в DFD равнозначны, стрелки могут начинаться и заканчиваться на любой стороне прямоугольника. Стрелки могут быть двунаправлены.

Хранилище данных. В отличие от стрелок, описывающих объекты в движении, хранилища данных изображают объекты в покое (рис. 24). Хранилище данных – это 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. Оно в общем случае является прообразом будущей базы данных, и описание хранящихся в нем данных должно соответствовать информационной модели (*Entity-Relationship Diagram*).

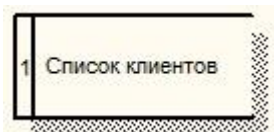


Рис. 24. Хранилище данных в DFD

Декомпозиция работы IDEF0 в диаграмму DFD. При декомпозиции работы *IDEF0* в *DFD* необходимо выполнить следующие действия:

- удалить все граничные стрелки на диаграмме *DFD*;
- создать соответствующие внешние сущности и хранилища данных;
- создать внутренние стрелки, начинающиеся с внешних сущностей вместо граничных стрелок;
- стрелки на диаграмме *IDEF0* затоннелировать.

Строго придерживаться правил нотации *DFD* не всегда удобно, поэтому *BPWin* позволяет создавать в *DFD* диаграммах граничные стрелки.

Построение диаграммы декомпозиции. Проведем декомпозицию работы *Отгрузка и снабжение диаграммы A0 «Деятельность предприятия по сборке и продаже компьютеров и ноутбуков»*. В этой работе мы выделили следующие дочерние работы:

- снабжение необходимыми комплектующими – занимается действиями, связанными с поиском подходящих поставщиков и заказом у них необходимых комплектующих;
- хранение комплектующих и собранных компьютеров;
- отгрузка готовой продукции – все действия, связанные с упаковкой, оформлением документации и собственно отгрузкой готовой продукции.

Выделим работу *Отгрузка и снабжение* диаграммы *A0 «Деятельность предприятия по сборке и продаже компьютеров и ноутбуков»*, нажмем на кнопку «Go to Child Diagram» панели инструментов и выберем нотацию *DFD*. При создании дочерней диаграммы *BPWin* переносит граничные стрелки родительской работы, их необходимо удалить и заменить на внешние сущности. Стрелки механизмов, стрелки управления «Правила и процедуры», «Управляющая информация» и стрелку выхода «Отчет» на дочерней диаграмме задействованы не будут, чтоб не загромождать диаграмму менее существенными деталями. Остальные стрелки заменим на внешние сущности – кнопка «External Reference Tool» на панели инструментов, в появившемся окне выбрать переключатель «Arrow» и выбрать из списка нужное название (рис. 25):

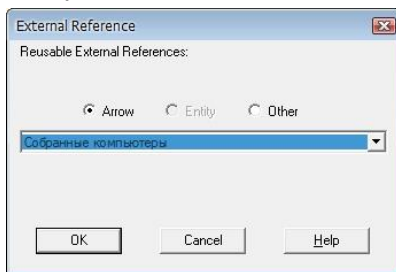


Рис. 25. Добавление внешней сущности

Далее разместим дочерние работы, свяжем их со внешними сущностями и между собой (рис. 26).

Центральной здесь является работа «Хранение комплектующих и собранных компьютеров». На ее вход поступают собранные компьютеры и полученные от поставщиков комплектующие, а также список необходимых для сборки компьютеров комплектующих. Выходом этой работы будут необходимые комплектующие (если они есть в наличии), список отсутствующих комплектующих, передаваемый на вход работы «Снабжение необходимыми комплектующими» и собранные компьютеры, передаваемые на отгрузку. Выходами работ «Снабжение необходимыми комплектующими» и «Отгрузка готовой продукции» будут, соответственно, заказы поставщикам и готовая продукция.

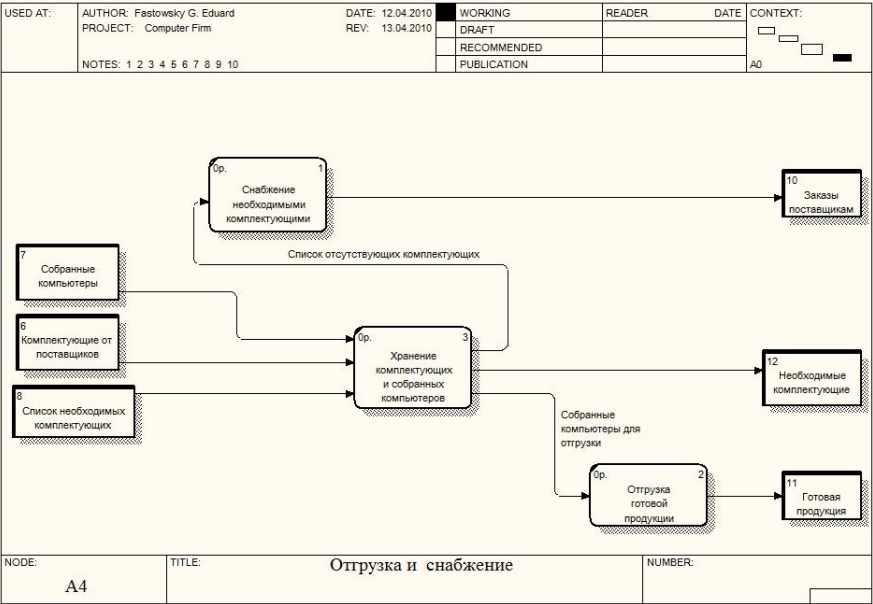


Рис. 26. Работы и внешние сущности

Следующим шагом необходимо определить, какая информация необходима для каждой работы, т.е. необходимо разместить на диаграмме хранилища данных (рис. 27).

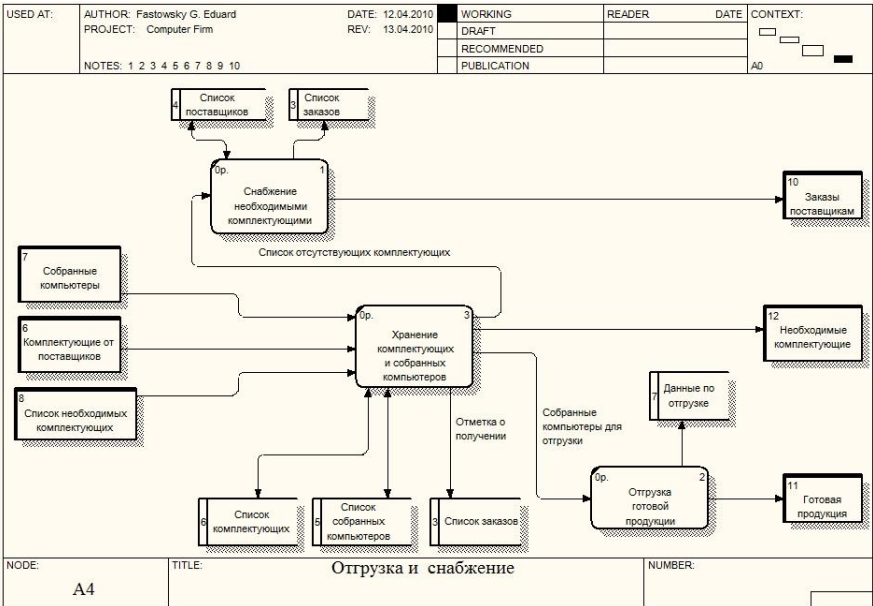


Рис. 27. Итоговая диаграмма декомпозиции

Работа «Снабжение необходимыми комплектующими» работает с информацией о поставщиках и с информацией о заказах, сделанных у этих поставщиков. Стрелка, соединяющая работу и хранилище данных «Список поставщиков» двунаправленная, т.к. работа может как получать информацию о имеющихся поставщиках, так и вносить данные о новых поставщиках. Стрелка, соединяющая работу с хранилищем данных «Список заказов» однонаправленная, т.к. работа только вносит информацию о сделанных заказах.

Работа «Хранение комплектующих и собранных компьютеров» работает с информацией о получаемых и выдаваемых комплектующих и собранных компьютерах, поэтому стрелки, соединяющая работу с хранилищами данных «Список комплектующих» и «Список собранных компьютеров» двунаправленные. Также эта работа при получении комплектующих должна делать отметку о том, что заказ поставщикам выполнен. Для этого она связана с хранилищем данных «Список заказов» однонаправленной стрелкой. Обратите внимание, что на DFD диаграммах одно и тоже хранилище данных может дублироваться.

Наконец, работа «Отгрузка готовой продукции» должна хранить информацию по выполненным отгрузкам. Для этого вводится соответствующее хранилище данных – «Данные по отгрузке».

Последним действием необходимо стрелки родительской работы затуннелировать (рис. 28):

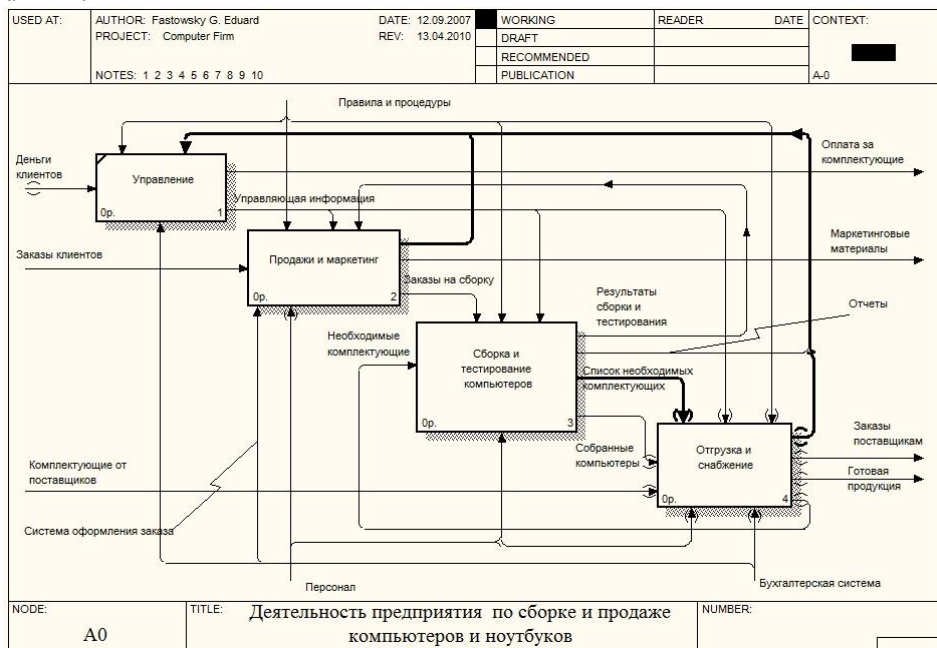


Рис. 28. Диаграмма IDEFO с затуннелированными стрелками работы «Отгрузка и снабжение»

ЭТАП 7. ПОСТРОЕНИЕ ФЕО ДИАГРАММ И ДИАГРАММ ДЕРЕВА УЗЛОВ ФЕО ДИАГРАММЫ

ФЕО (For Exposition Only) диаграммы (другое название – диаграммы только для экспозиции, описания) используются для иллюстрации альтернативной точки зрения, для

отображения отдельных деталей, которые не поддерживаются явно синтаксисом IDEF0. FEO диаграммы позволяют нарушить любое синтаксическое правило, поскольку эти диаграммы – фактически обычные картинки – копии стандартных диаграмм. Например, работа на FEO диаграмме может не иметь стрелок выхода или управления. AllFusion Process Modeler позволяет также строить FEO диаграммы для диаграмм в нотации DFD.

Для построения FEO диаграммы необходимо выбрать пункт меню *Diagram* → *Add FEO Diagram* и в появившемся окне выбрать диаграмму, на базе которой будет строиться FEO диаграмма (рис. 29).

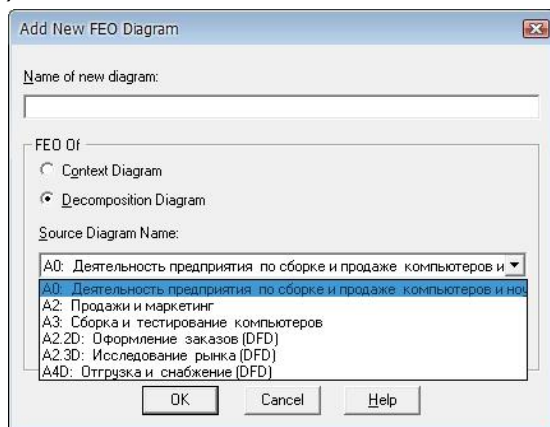


Рис. 29. Добавление FEO диаграммы

Созданная диаграмма будет точной копией родительской диаграммы и будет иметь номер, равный номеру родительской диаграммы + буква F. После создания диаграммы ее можно изменять. При этом изменения не будут влиять на родительскую диаграмму.

Для просмотра списка имеющихся FEO диаграмм нужно выбрать в *Обозревателе Модели* (Model Explorer) вкладку *Diagrams* (рис. 30).

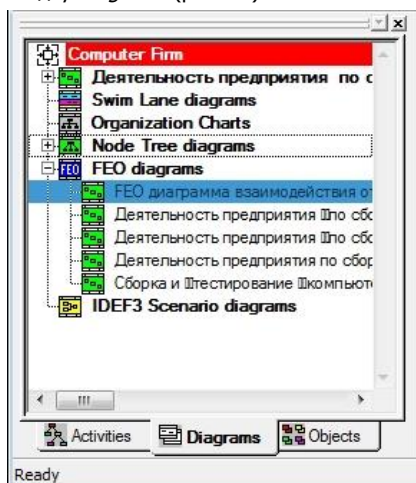


Рис. 30. Просмотр списка имеющихся FEO диаграмм

Построим FEO диаграмму для диаграммы декомпозиции второго уровня A0 «Деятельность предприятия по сборке и продаже компьютеров и ноутбуков» и покажем на ней как дочерние работы связаны между собой. Для этого создаем диаграмму, как показано выше, и удаляем на ней все граничные стрелки. Итоговая FEO диаграмма показана на рис. 31:

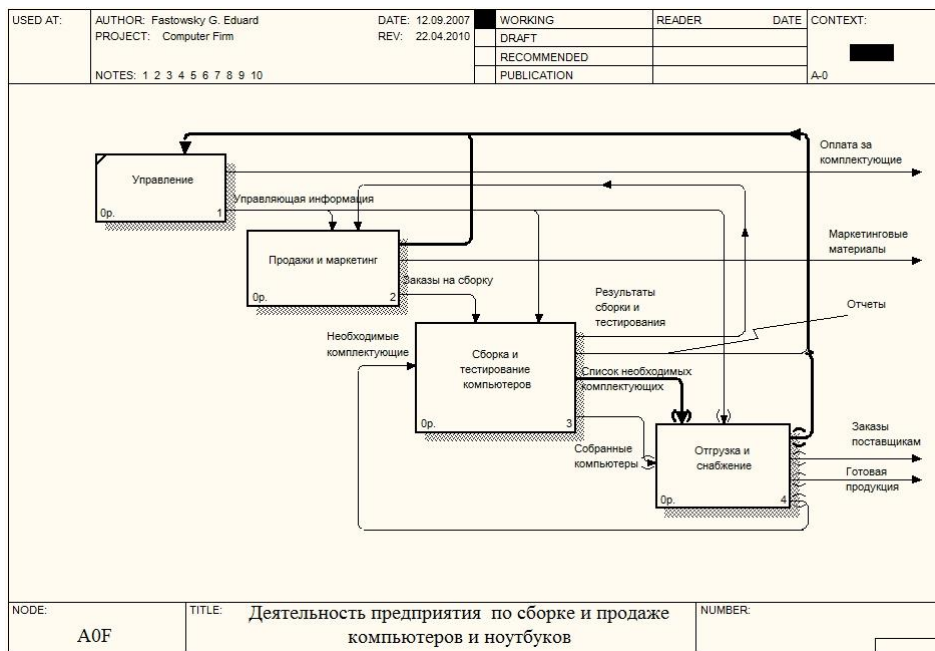


Рис. 31. FEO диаграмма

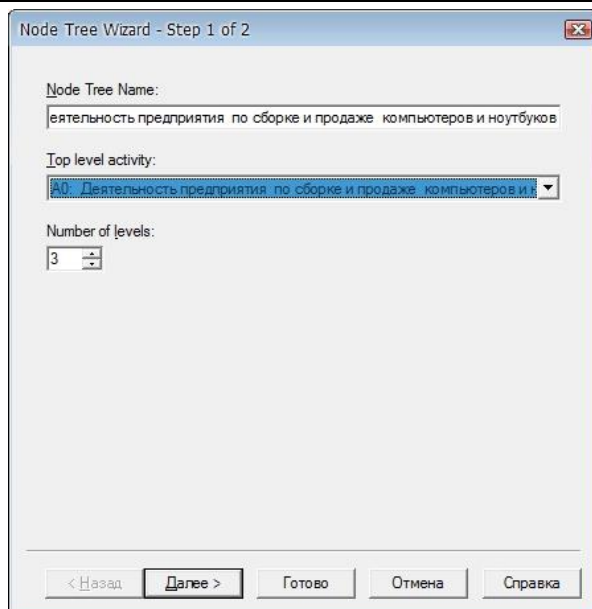
ДИАГРАММЫ ДЕРЕВА УЗЛОВ

Диаграмма дерева узлов показывает иерархическую зависимость работ, но не взаимосвязи между работами. В одной модели диаграмм дерева узлов может быть множество, поскольку дерево может быть построено на произвольную глубину и не обязательно с корня.

Для построения диаграммы дерева узлов необходимо выбрать пункт меню *Diagram* → *Add Node Tree*. Появляется мастер, с помощью которого диаграмма будет создана. На первом шаге (рис. 32) задается имя диаграммы дерева узлов, узел верхнего уровня и глубина дерева. Имя дерева узлов по умолчанию совпадает с именем работы верхнего уровня, а номер диаграммы генерируется автоматически как номер узла верхнего уровня + буква N.

На втором шаге мастера (рис. 33) задаются свойства диаграммы дерева узлов.

По умолчанию нижний уровень декомпозиции показывается в виде списка, остальные работы – в виде прямоугольников. Если необходимо отобразить все дерево в виде прямоугольников, то следует снять галочку возле опции «*Bullet last level*». Список всех созданных диаграмм дерева узлов можно посмотреть в Обозревателе Модели.



Node Tree Wizard - Step 1 of 2

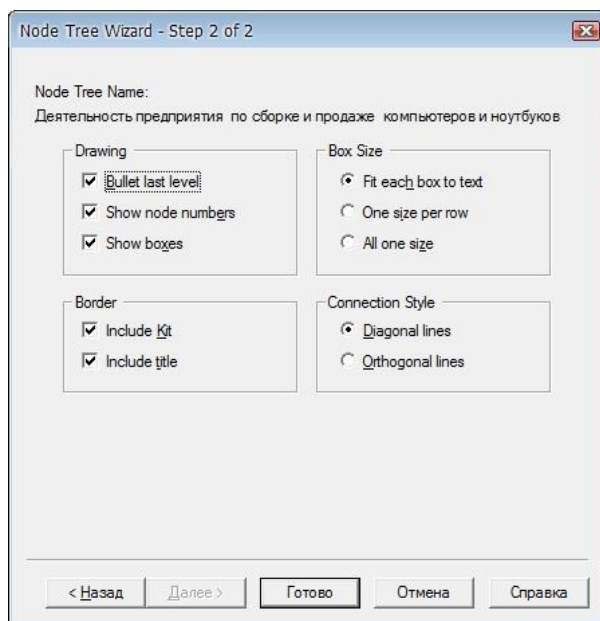
Node Tree Name:
Деятельность предприятия по сборке и продаже компьютеров и ноутбуков

Top level activity:
A0. Деятельность предприятия по сборке и продаже компьютеров и ноутбуков

Number of levels:
3

< Назад **Далее >** Готово Отмена Справка

Рис. 32. Создание диаграммы дерева узлов. Шаг 1



Node Tree Wizard - Step 2 of 2

Node Tree Name:
Деятельность предприятия по сборке и продаже компьютеров и ноутбуков

Drawing

- ☒ Bullet last level
- ☒ Show node numbers
- ☒ Show boxes

Box Size

- ☒ Fit each box to text
- ☐ One size per row
- ☐ All one size

Border

- ☒ Include Kit
- ☒ Include title

Connection Style

- ☒ Diagonal lines
- ☐ Orthogonal lines

< Назад Далее > **Готово** Отмена Справка

Рис. 33. Создание диаграммы дерева узлов. Шаг 2

Диаграмма дерева узлов для всех узлов модели показана на рис. 34:

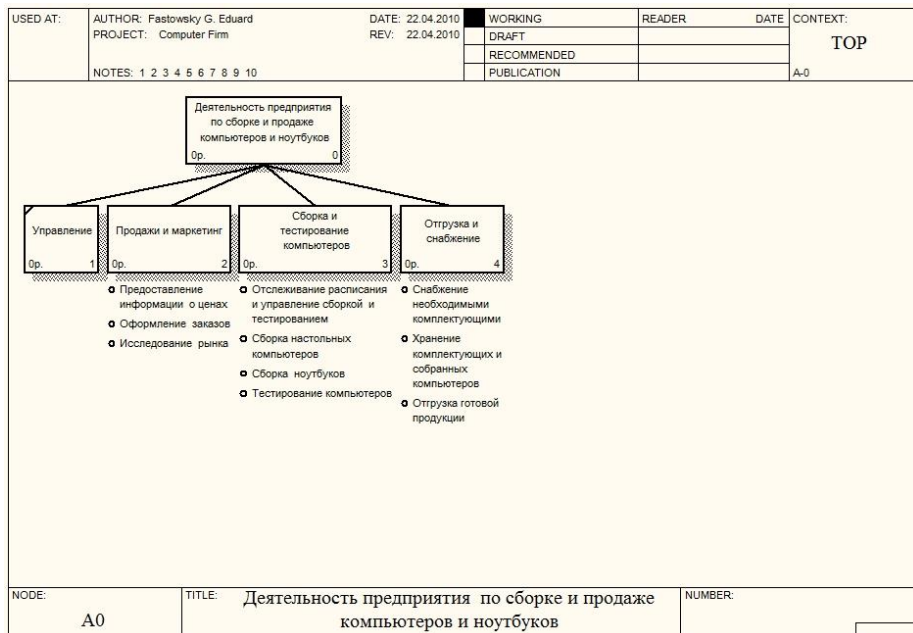


Рис. 34. Диаграмма дерева узлов

ЭТАП 8. ОСНОВЫ РАБОТЫ С ПРОГРАММНЫМ ПРОДУКТОМ ALLFUSION ERWIN DATA MODELER

CA ERwin Data Modeler (далее **ERwin**) – CASE-средство для проектирования и документирования баз данных, которое позволяет создавать, документировать и сопровождать базы данных, хранилища и витрины данных.

Работа с программой начинается с создания новой модели, для которой нужно указать тип и целевую СУБД (рис. 35).

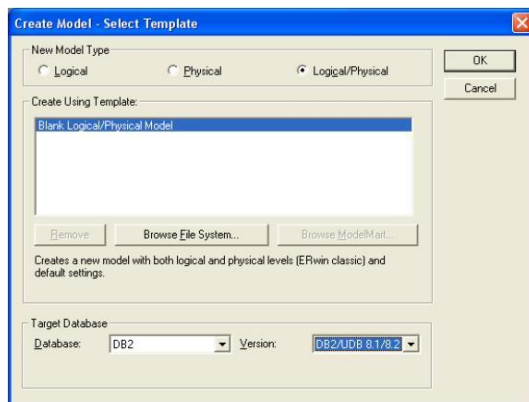


Рис. 35. Создание новой модели

ERwin позволяет создавать логическую, физическую модели и модель, совмещающую логический и физический уровни.

Логический уровень – это абстрактный взгляд на данные, на нем данные представляются так, как выглядят в реальном мире, и могут называться так, как они называются в реальном мире (например «Постоянный клиент», «Отдел» или «Заказ»).

Объекты модели, представляемые на логическом уровне, называются сущностями и атрибутами. Логическая модель данных является универсальной и никак не связана с конкретной реализацией СУБД.

Физический уровень зависит от конкретной СУБД. В физической модели содержится информация о всех объектах БД. Физическая модель зависит от конкретной реализации СУБД. Одной и той же логической модели могут соответствовать несколько разных физических моделей.

На логическом уровне ERwin поддерживает две нотации (IE и IDEF1X), на физическом – три (IE, IDEF1X и DM). Далее будет рассматриваться работа с ERwin в нотации IDEF1X.

Переключение между логической и физической моделями данных осуществляется через список выбора на стандартной панели (рис. 36).



Рис. 36. Переключение между уровнями

Примечание. В созданной модели с настройками по умолчанию некорректно отображаются русские символы. Чтобы устранить этот недостаток, необходимо подкорректировать используемые в модели шрифты. Для этого необходимо зайти в меню *Format* → *Default Fonts & Colors*, последовательно пройти по всем вкладкам, в качестве шрифта выбрав любой шрифт, название которого заканчивается на *CYR* (например, *Arial CYR*), и выставив переключатель *Apply To* в значение *All Objects*.

ЛОГИЧЕСКИЙ УРОВЕНЬ МОДЕЛИ ДАННЫХ

Для создания на логическом уровне сущностей и связей между ними предназначена панель *Toolbox*.

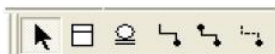


Рис. 37. Панель *Toolbox*

Таблица 5. Назначение кнопок панели *ToolBox*

Вид кнопки	Назначение кнопки
	Создание новой сущности. Для этого нужно щелкнуть по кнопке и затем по свободному месту на модели
	Создание категории. Для установки категориальной связи нужно щелкнуть по кнопке, далее - по сущности-родителю, и затем - по сущности-потомку.
	Создание идентифицирующей связи. Для связывания двух сущностей нужно щелкнуть по кнопке, далее – по сущности-родителю, затем – по сущности-потомку.
	Создание связи «многие ко многим»
	Создание неидентифицирующей связи

После создания сущности ей нужно задать атрибуты. Для этого нужно дважды щелкнуть по ней или в контекстном меню выбрать пункт *Attributes* (рис. 38).

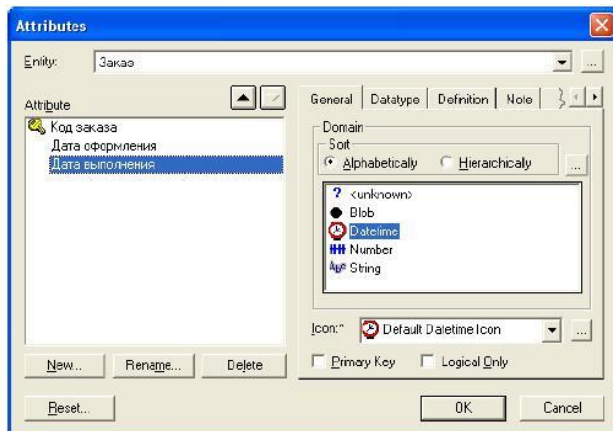


Рис. 38. Окно атрибутов выбранной сущности

В появившемся окне можно просмотреть и отредактировать информацию о созданных атрибутах, создать новые. Здесь же задается первичный ключ. Для создания нового атрибута следует нажать кнопку **New**. В появившемся окне можно выбрать тип атрибута (BLOB, дата/время, число, строка), задать имя атрибута (**Attribute Name**) и имя столбца (**Column Name**), который будет соответствовать атрибуту на физическом уровне (рис. 39).

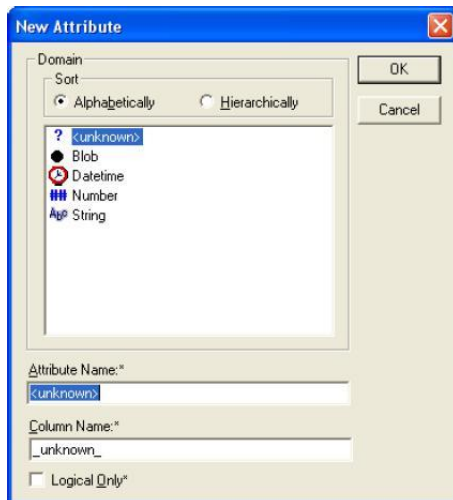


Рис. 39. Окно создания атрибута

После создания сущностей создаются связи между ними. При создании идентифицирующей связи атрибуты, составляющие первичный ключ сущности-родителя, мигрируют в состав первичного ключа сущности-потомка, при создании неидентифицирующей связи – просто в состав атрибутов сущности-потомка. Задать свойства

связи или поменять ее тип можно дважды щелкнув по ней или выбрав в контекстном меню пункт *Relationship Properties* (рис. 40). Здесь во вкладке *General* можно задать имя связи (в направлении родитель-потомок и потомок-родитель), мощность связи (ноль, один или больше; один и больше (P); ноль или один (Z); точно (конкретное число)), поменять тип связи. Во вкладке *RI Action* можно задать ограничения целостности.

Пример логической модели базы данных приведен на рис. 41.

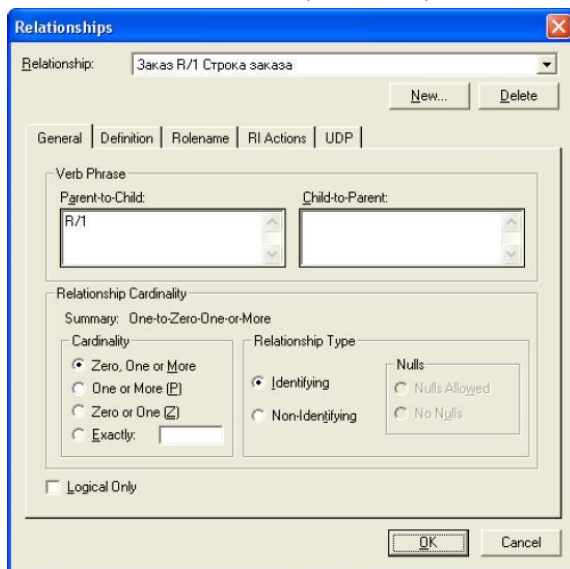


Рис. 40. Окно свойств связи



Рис. 41. Пример логической схемы БД

ФИЗИЧЕСКИЙ УРОВЕНЬ МОДЕЛИ ДАННЫХ

При переключении с логического уровня на физический автоматически будет создана физическая схема базы данных (рис. 42)

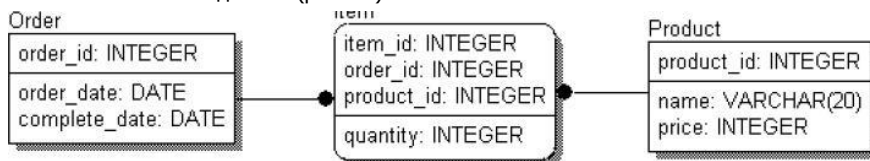


Рис. 42. Автоматически созданная физическая схема БД

Ее можно дополнить, отредактировать или изменить. Принципы работы с физической схемой аналогичны принципам работы с логической схемой.

По готовой физической схеме можно сгенерировать скрипты для выбранной СУБД. Для этого предназначен пункт меню *Tools → Forward Engineering/Schema Generation* (рис. 43).

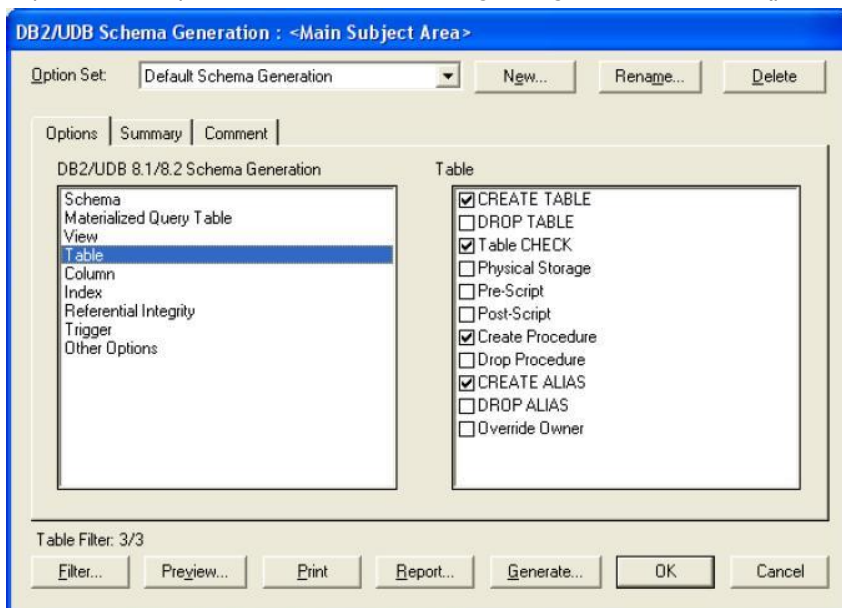


Рис. 43. Окно генерации SQL-скриптов для целевой СУБД

Здесь можно указать, какие именно скрипты следует генерировать, предварительно просмотреть их и непосредственно сгенерировать (при этом ERwin произведет подключение к целевой СУБД и в автоматическом режиме выполнит все SQL-скрипты).

ЭТАП 9. ПОСТРОЕНИЕ ЛОГИЧЕСКОЙ МОДЕЛИ ДАННЫХ ПРЕДМЕТНОЙ ОБЛАСТИ В НОТАЦИИ IDEF1X

На данном этапе необходимо построить в нотации IDEF1X в CASE-средстве ERwin Data Modeler логическую схему данных предметной области, бизнес-процессы которой моделировались в предыдущих работах.

Примечание. При построении модели можно ограничиться 5-6 сущностями.

IDEF1X

IDEF1X основан на подходе Чена и позволяет построить модель данных, эквивалентную реляционной модели в третьей нормальной форме. Нотация Чена и сам процесс построения диаграмм сущность-связь изучалась в курсе «Организация баз данных и знаний», поэтому здесь мы рассмотрим только отличия IDEF1X от нотации Чена.

Сущность (Entity) – реальный либо воображаемый объект, имеющий существенное значение для рассматриваемой предметной области. Каждая сущность должна иметь наименование, выраженное существительным в единственном числе. Каждая сущность должна обладать уникальным идентификатором. Каждый экземпляр сущности должен

однозначно идентифицироваться и отличаться от всех других экземпляров данного типа сущности.

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

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

В методе IDEF1X все сущности делятся на зависимые и независимые от идентификаторов. Сущность является независимой от идентификаторов или просто независимой, если каждый экземпляр сущности может быть однозначно идентифицирован без определения его отношений с другими сущностями. Сущность называется зависимой от идентификаторов или просто зависимой, если однозначная идентификация экземпляра сущности зависит от его отношения к другой сущности. Независимая сущность изображается в виде обычного прямоугольника, зависимая – в виде прямоугольника с закругленными углами.

В IDEF1X существуют следующие виды мощностей связей:

- N мощность – каждый экземпляр сущности-родителя может иметь ноль, один или более одного связанного с ним экземпляра сущности-потомка (по умолчанию);
- Р мощность – каждый экземпляр сущности-родителя должен иметь не менее одного связанного с ним экземпляра сущности-потомка;
- Z мощность – каждый экземпляр сущности-родителя должен иметь не более одного связанного с ним экземпляра сущности-потомка;
- конкретное число – каждый экземпляр сущности-родителя связан с некоторым фиксированным числом экземпляров сущности-потомка.

Связь изображается линией, проводимой между сущностью-родителем и сущностью-потомком, с точкой на конце линии у сущности-потомка. По умолчанию мощность связи принимается равной N. Если экземпляр сущности-потомка однозначно определяется своей связью с сущностью-родителем, то связь называется идентифицирующей, в противном случае – неидентифицирующей. Идентифицирующая связь изображается сплошной линией, неидентифицирующая – пунктирной линией.

В ERwin'e при установлении идентифицирующей связи атрибуты первичного ключа родительской сущности автоматически переносятся в состав первичного ключа дочерней сущности. Эта операция называется миграцией атрибутов. В дочерней сущности новые атрибуты помечаются как внешний ключ (FK). При установке неидентифицирующей связи атрибуты первичного ключа родительской сущности мигрируют в состав неключевых полей дочерней сущности.

Построение логической модели данных предприятия по сборке и продаже компьютеров и ноутбуков. Построение модели данных начинается с выделения сущностей данной предметной области. В нашем случае были выделены следующие сущности:

- клиент – человек, который покупает компьютеры;
- заказ – список компьютеров, которые покупает клиент;
- компьютер;
- комплектующие – то, из чего собирают компьютеры;
- сотрудник – сотрудник предприятия, собирающий конкретный компьютер.

Далее рассмотрим связи между сущностями:

Клиент—Заказ. Один клиент может делать несколько заказов. При этом если данные о клиенте имеются в базе данных, то он сделал минимум один заказ. Поэтому мощность связи – Р. Связь идентифицирующая, т.к. заказ без клиента существовать не может;

Заказ—Компьютер. В рамках одного заказа клиент может заказать несколько компьютеров, но как минимум заказ должен состоять из одного компьютера. Поэтому мощность связи – Р. Связь идентифицирующая, т.к. компьютер без заказа существовать не может;

Компьютер—Комплектующие. В состав одного компьютера входит много различных комплектующих; один и тот же тип комплектующего может входить в состав разных компьютеров. Мощность связи - много ко многим. В IDEF1X такой тип связи отсутствует, поэтому вводим промежуточную (ассоциативную) сущность – Конфигурация. Мощность связи между сущностями Компьютер и Конфигурация – Р, поскольку у любого компьютера должна быть конфигурация, мощность между сущностями Комплектующие и Конфигурация – N, поскольку какие-то комплектующие еще могут быть не установлены ни в один компьютер. Связь в обоих случаях идентифицирующая, т.к. конфигурация компьютера не может существовать без привязки к самому компьютеру и к комплектующим;

Комплектующие—Тип комплектующих. Поскольку перечень типов комплектующих, которые могут быть установлены в компьютер, ограничен, но используется очень часто, то мы приняли решение создать еще одну сущность – Тип комплектующих. Мощность связи – Р. Связь идентифицирующая;

Компьютер—Сотрудник. Каждый компьютер собирается каким-то одним сотрудником. Какие-то сотрудники могут собирать множество компьютеров. Мощность связи – N. Тип связи – неидентифицирующая, поскольку экземпляр сущности Компьютер уже может существовать, но за ним еще может быть не закреплён ни один сотрудник. Именно из этих же соображений в свойствах этой связи мы выбрали переключатель «Nulls Allowed» (на диаграмме это отображается в виде незакрашенного ромбика со стороны сущности-родителя).

Итоговая диаграмма показана на рис. 44:

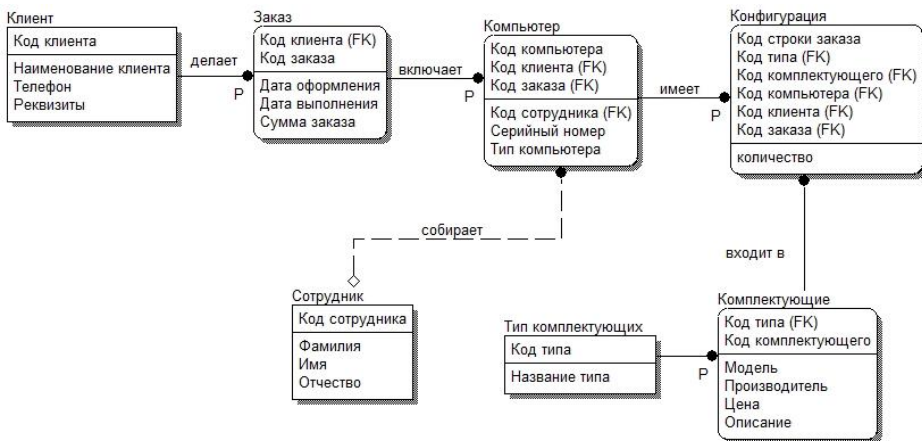


Рис. 44. Логическая модель данных предприятия по сборке компьютеров и ноутбуков