

Тема 3: Методология функционального моделирования SADT

Введение.

В настоящее время технологии разработки программного обеспечения (на базе графических моделей программ) представлены такими типами диаграмм:

- диаграммы IDEF;
- диаграммы UML;
- онтологии.

Все они являются языками проектирования сложных систем, т.е. — языками системного проектирования (СП).

Рассмотрим их детальнее.

1. Классификация графических языков автоматизации разработки программного обеспечения

Комплекс языков СП в настоящее время включает такие классы языков и инструментальных средств поддержки:

1) Методология функционального моделирования SADT. Ориентирована на задачу поддержки прямого построения описания требуемого объекта проектирования. Поддерживается графическими редакторами, работающими с различными типами диаграмм, отличающихся назначением (предметной областью). Можно назвать IDEF0, IDEF1, IDEFX, IDEF3, IDEF5 и т.д.

2) Средства, развивающие CASE-технологии проектирования программного обеспечения как инструмент проектирования объектов различных предметных областей. К данному направлению относятся языки UML, SYSML, ADA.

3) Средства создания предметно-ориентированных онтологий в инженерии знаний. Ориентированы на задачу извлечения знаний для построения баз знаний проектирования объектов различных предметных областей. Отличаются графическим интерфейсом и типом используемых отношений.

Можно назвать для примера редакторы онтологий Apollo, Protégé-2000, WebOnto, Ontology Editor, KBE Knowledge Base Editor, Knowledge Server и т.д.

Мы постараемся коснуться всех трех классов.

Задача построения соответствующего описания объекта проектирования на том или ином языке СП будет рассматриваться нами как задача построения дескриптивного описания модели.

Тема текущей лекции - основные положения методологии функционального моделирования SADT на примере диаграмм типа IDEF0.

2. Назначение и общие возможности

Методология SADT разработана Дугласом Россом. Методология SADT представляет собой совокупность методов, правил и процедур, предназначенных для построения структурно-функциональной модели объекта какой-либо предметной области.

Широко используется в проектировании бизнес-процессов, технических объектов, организационных систем и т.д.

Претендует на роль обязательного инструментального средства создания любых проектов на ранних стадиях.

Позволяет обеспечивать передачу моделей в системы более позднего проектирования (например, ARENA).

Функциональная модель SADT отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями.

Основные элементы этой методологии основываются на следующих концепциях:

1) **Графическое представление блочного моделирования.** Графика блоков и дуг SADT-диаграммы отображает функцию в виде блока, а интерфейсы входа/выхода представляются дугами, соответственно входящими в блок и выходящими из него. Взаимодействие блоков друг с другом описываются посредством интерфейсных дуг, выражающих "ограничения", которые в свою очередь определяют, когда и каким образом функции выполняются и управляются.

2) Строгость и точность. Выполнение правил SADT требует достаточной строгости и точности, не накладывая в то же время чрезмерных ограничений на действия аналитика. Правила SADT включают:

- Ограничение количества блоков на каждом уровне декомпозиции (правило 3-6 блоков, ограничение когнитивной сложности, т.к. человек при большем количестве факторов начинает их интуитивно делить на группы);
- Связность диаграмм (композиция в нумерации блоков);
- Уникальность меток и наименований (отсутствие повторяющихся имен);

- Синтаксические правила для графики (блоков и дуг);
- Разделение входов и управлений (правило определения роли данных).
- Отделение организации от функции, т.е. исключение влияния организационной структуры на функциональную модель.

Методология SADT может использоваться для моделирования широкого круга систем и определения требований и функций, а затем для разработки системы, которая удовлетворяет этим требованиям и реализует эти функции.

Для уже существующих систем SADT может использоваться для анализа функций, выполняемых системой, а также для указания механизмов, посредством которых они осуществляются.

Является удобным механизмом построения структурных моделей.

3. Состав функциональной модели

Результатом применения методологии SADT является модель, которая состоит из диаграмм (схем), фрагментов текстов и глоссария, имеющих ссылки друг на друга.

Диаграммы - главные компоненты модели, все функции ИС и интерфейсы на них представлены как блоки и дуги.

Место соединения дуги с блоком определяет тип интерфейса.

Управляющая информация входит в блок сверху.

Информация, которая *подвергается обработке (входная)*, показана с левой стороны блока.

Результаты (выход) показаны с правой стороны блока.

Механизм (человек или автоматизированная система), который осуществляет операцию, представляется дугой, входящей в блок снизу (рис.1.).

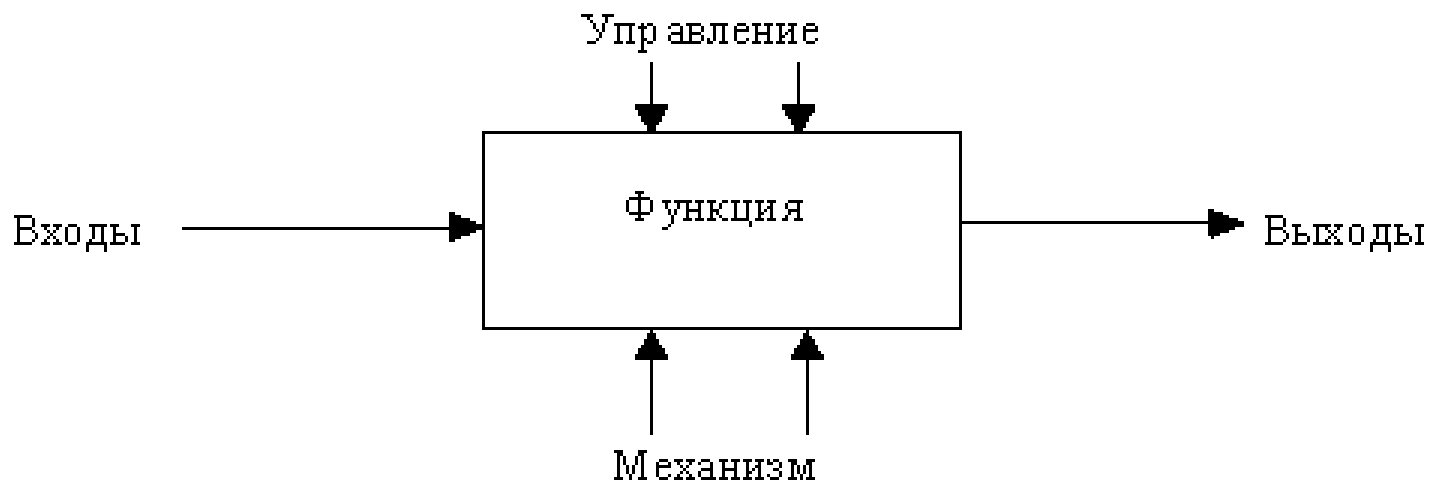


Рис 1. Функциональный блок и интерфейсные дуги

Механизмы (дуги с нижней стороны) показывают средства, с помощью которых осуществляется выполнение функций. Механизм может быть человеком, компьютером или любым другим устройством, которое помогает выполнять данную функцию (рисунок 2).

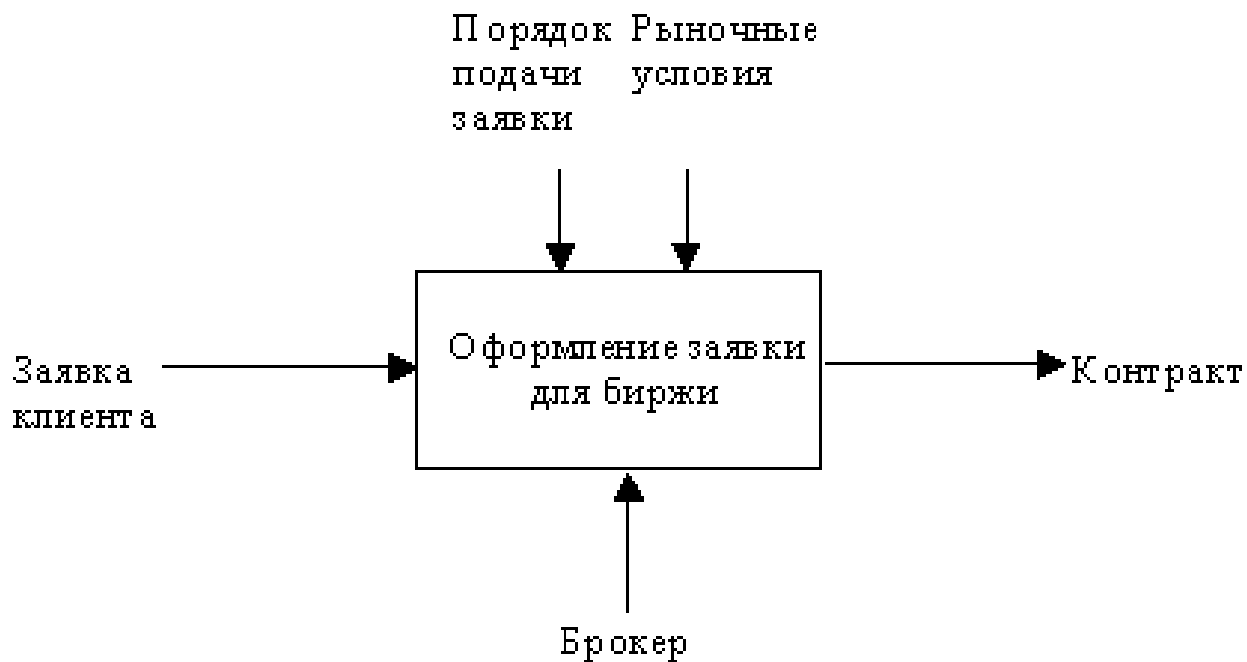
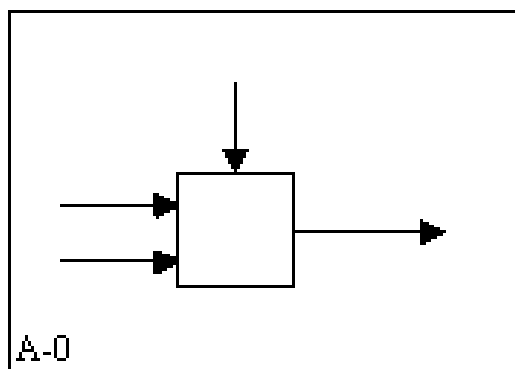


Рис. 2. Пример механизма

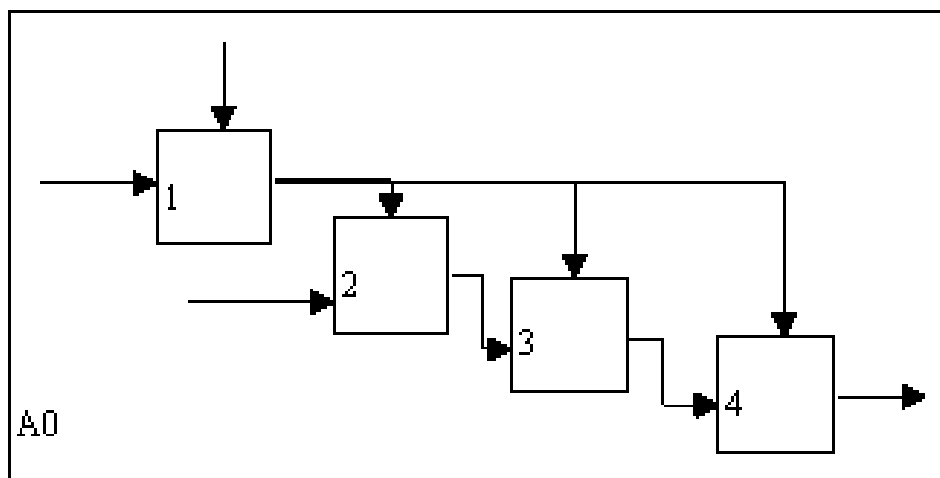
Пример 2: *блок* – проектирование дома,
управление – ГОСТы на проектирование,
входы – параметры желаемого проекта,
выход – готовый проект (чертежи),
механизм – компьютер, ArchiCAD, его базы
данных, проектировщик.

Одной из наиболее важных особенностей методологии SADT является постепенное введение все больших уровней детализации по мере создания диаграмм, отображающих модель. На рисунке 3 приведены четыре диаграммы и их взаимосвязи, показана структура SADT-модели. Каждый компонент модели может быть декомпозирован на другой диаграмме. Каждая диаграмма иллюстрирует "внутреннее строение" блока на родительской диаграмме.

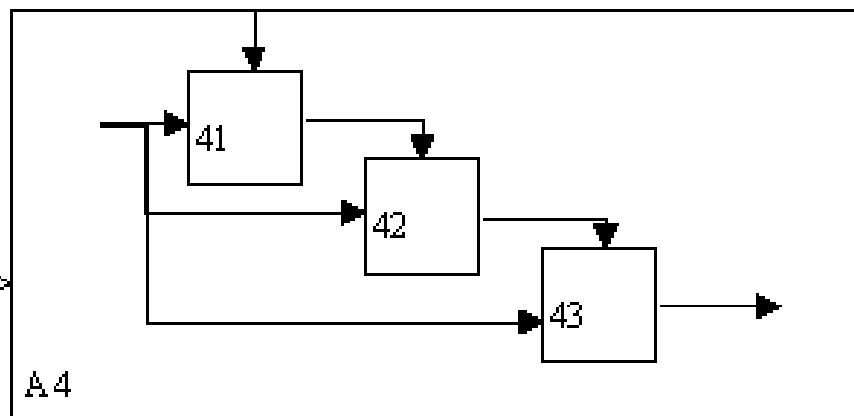


Более общее представление

Более детальное представление



Эта диаграмма является "родителем" этой диаграммы



*Рис.3. Структура SADT-модели.
Декомпозиция диаграмм*

4. Иерархия диаграмм

Построение SADT-модели начинается с представления всей системы в виде простейшей компоненты - одного блока и дуг, изображающих интерфейсы с функциями вне системы. Поскольку единственный блок представляет всю систему как единое целое, имя, указанное в блоке, является общим. Это верно и для интерфейсных дуг - они также представляют полный набор внешних интерфейсов системы в целом.

Затем блок, который представляет систему в качестве единого модуля, детализируется на другой диаграмме с помощью нескольких блоков, соединенных интерфейсными дугами.

Во всех случаях каждая подфункция может содержать только те элементы, которые входят в исходную функцию. Кроме того, модель не может опустить какие-либо элементы, т.е., как уже отмечалось, родительский блок и его интерфейсы обеспечивают контекст. К нему нельзя ничего добавить, и из него не может быть ничего удалено.

Другое название SADT – система для передачи понимания.

Модель SADT представляет собой серию диаграмм с сопроводительной документацией, разбивающих сложный объект на составные части, которые представлены в виде блоков. Детали каждого из основных блоков показаны в виде блоков на других диаграммах. Каждая детальная диаграмма является декомпозицией блока из более общей диаграммы. На каждом шаге декомпозиции более общая диаграмма называется *родительской* для более детальной диаграммы. —лек2----

Дуги, входящие в блок и выходящие из него на диаграмме верхнего уровня, являются точно теми же самыми, что и дуги, входящие в диаграмму нижнего уровня и выходящие из нее, потому что блок и диаграмма представляют одну и ту же часть системы.

Каждый блок на диаграмме имеет свой номер. Блок любой диаграммы может быть далее описан диаграммой нижнего уровня, которая, в свою очередь, может быть далее детализирована с помощью необходимого числа диаграмм. Таким путем формируется иерархия диаграмм.

Для того, чтобы указать положение любой диаграммы или блока в иерархии, используются номера диаграмм. Например, A21 является диаграммой, которая детализирует блок 1 на диаграмме A2. Аналогично, A2 детализирует блок 2 на диаграмме A0, которая является самой верхней диаграммой модели. На рисунке 4 показано типичное дерево диаграмм.

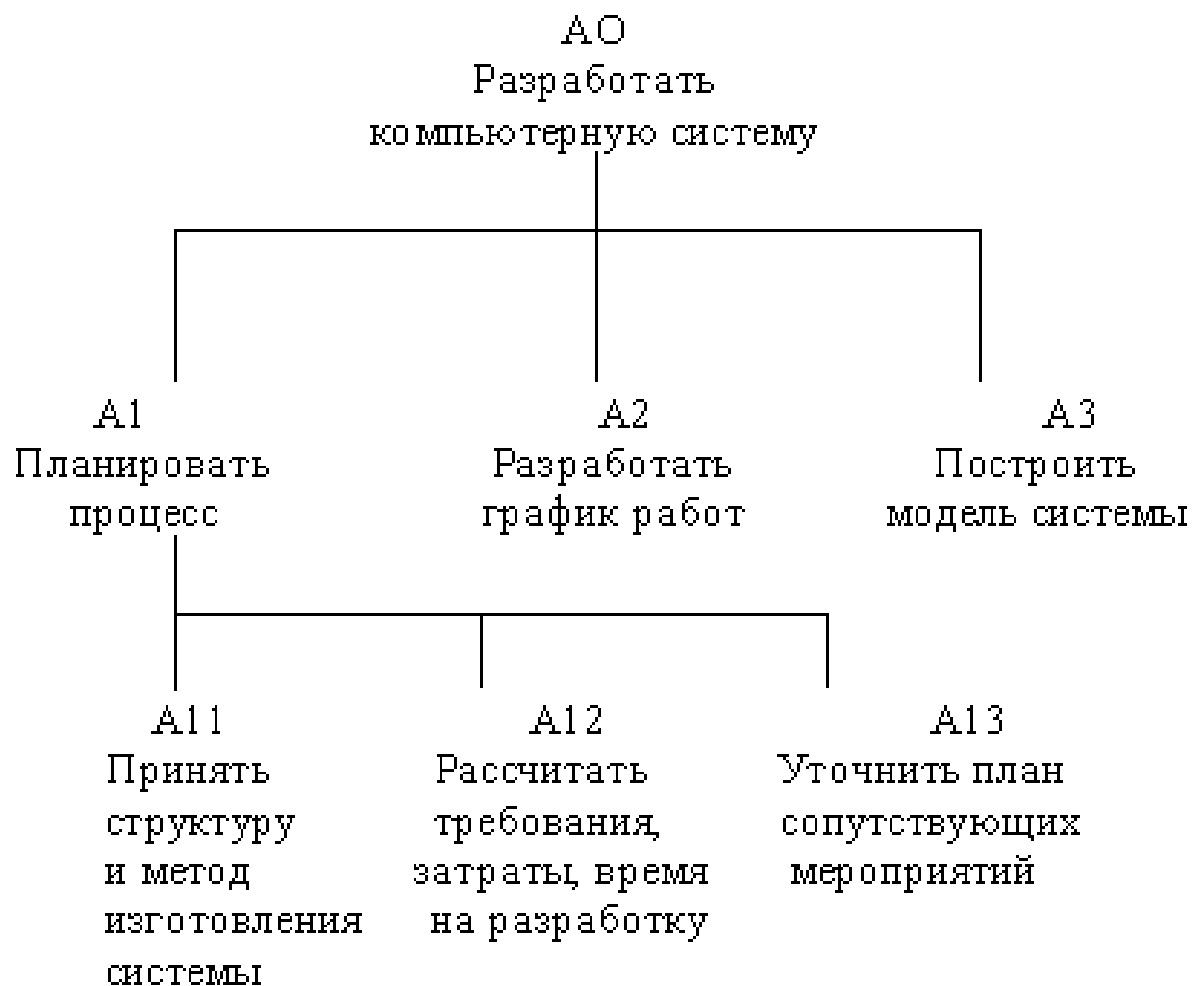


Рис. 4. Иерархия диаграмм

Некоторые дуги присоединены к блокам диаграммы обоими концами, у других же один конец остается неприсоединенным. Неприсоединенные дуги соответствуют входам, управлениям и выходам родительского блока. Источник или получатель этих пограничных дуг может быть обнаружен только на родительской диаграмме. Неприсоединенные концы должны соответствовать дугам на исходной диаграмме. Все граничные дуги должны продолжаться на родительской диаграмме, чтобы она была полной и непротиворечивой.

5. Типы связей между функциями

На SADT-диаграммах явно не указаны ни последовательность, ни время. Однако, обратные связи, итерации, продолжающиеся процессы и перекрывающиеся (по времени) функции могут быть изображены с помощью дуг.

Различают **семь типов** связывания.

1) Тип случайной связности. Связь между функциями мала или полностью отсутствует.

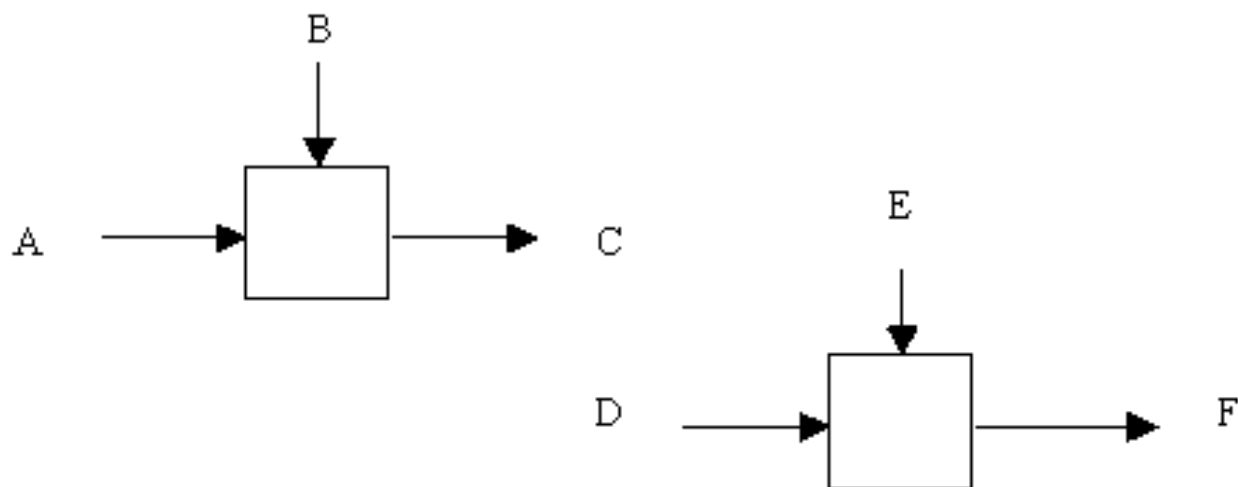


Рис. 8. Случайная связность

2) Тип логической связности. Данные и функции попадают в общий класс или набор элементов, но функциональных отношений между ними нет.

3) Тип временной связности. Функции связаны во времени, когда их данные используются одновременно или функции включаются параллельно, а не последовательно.

4) Тип процедурной связности. Функции выполняются в течение одной и той же части цикла или процесса.

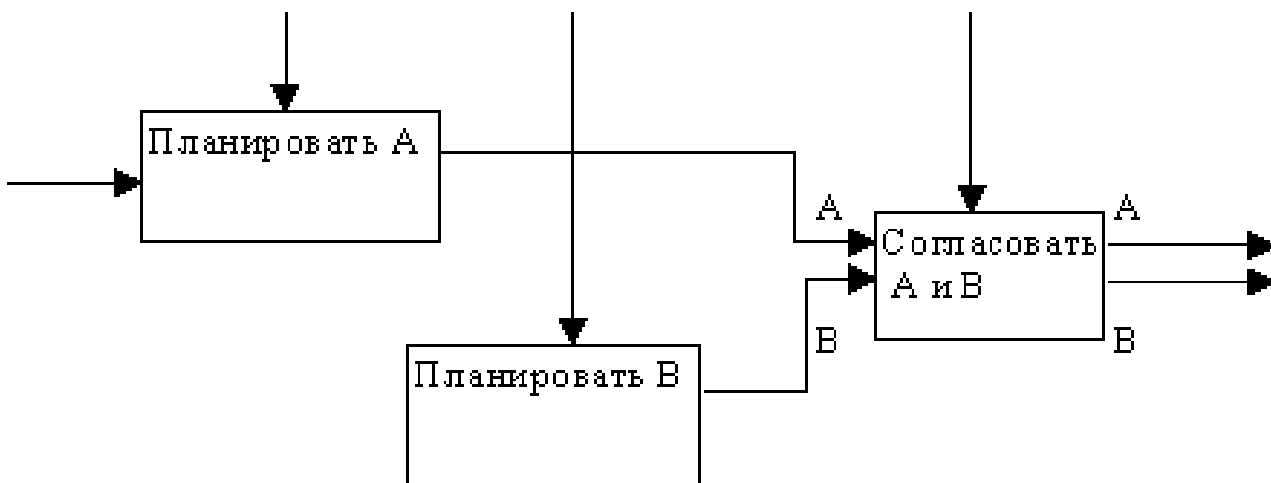


Рис. 9. Процедурная связность

5) Тип коммуникационной связности.

Блоки используют одни и те же входные данные и/или производят одни и те же выходные данные.

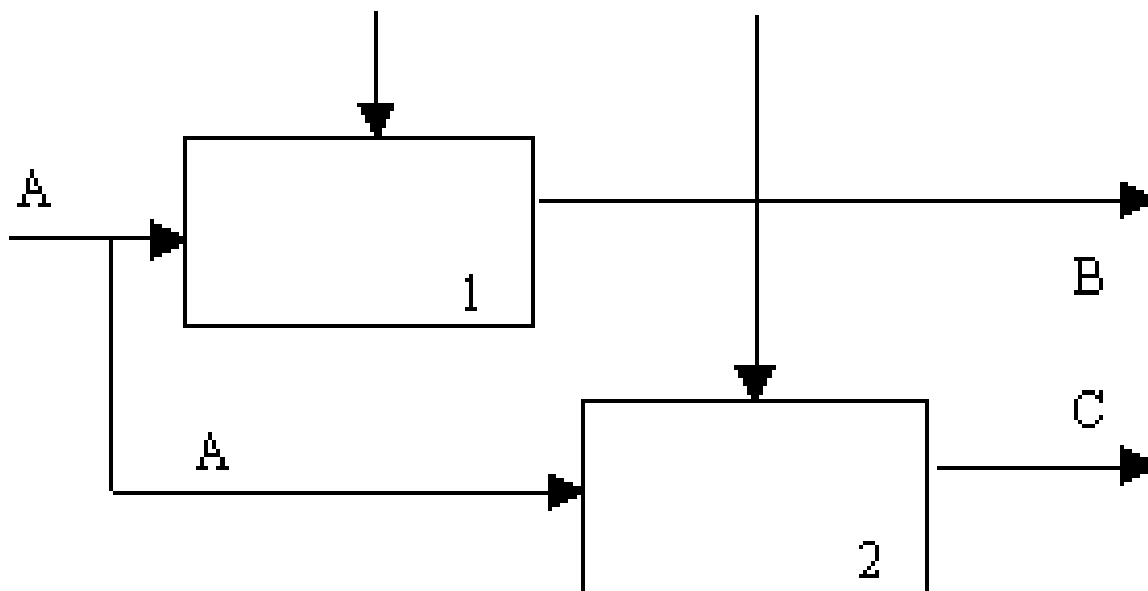


Рис.10. Коммуникационная связность

6) Тип последовательной связности.

Выход одной функции служит входными данными для следующей функции.

Моделирует причинно-следственные зависимости.

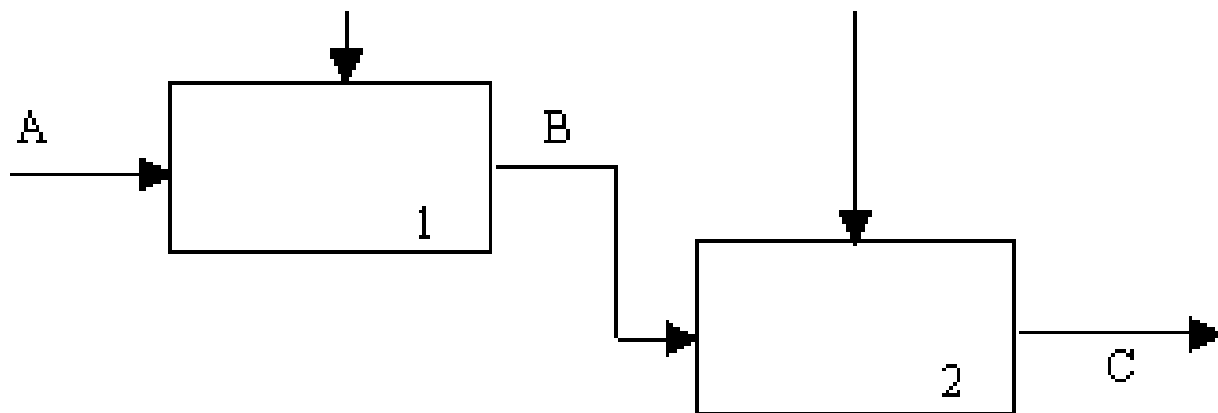


Рис. 11. Последовательная связность

7) Тип функциональной связности.
Отражает наличие полной зависимости одной функции от другой.

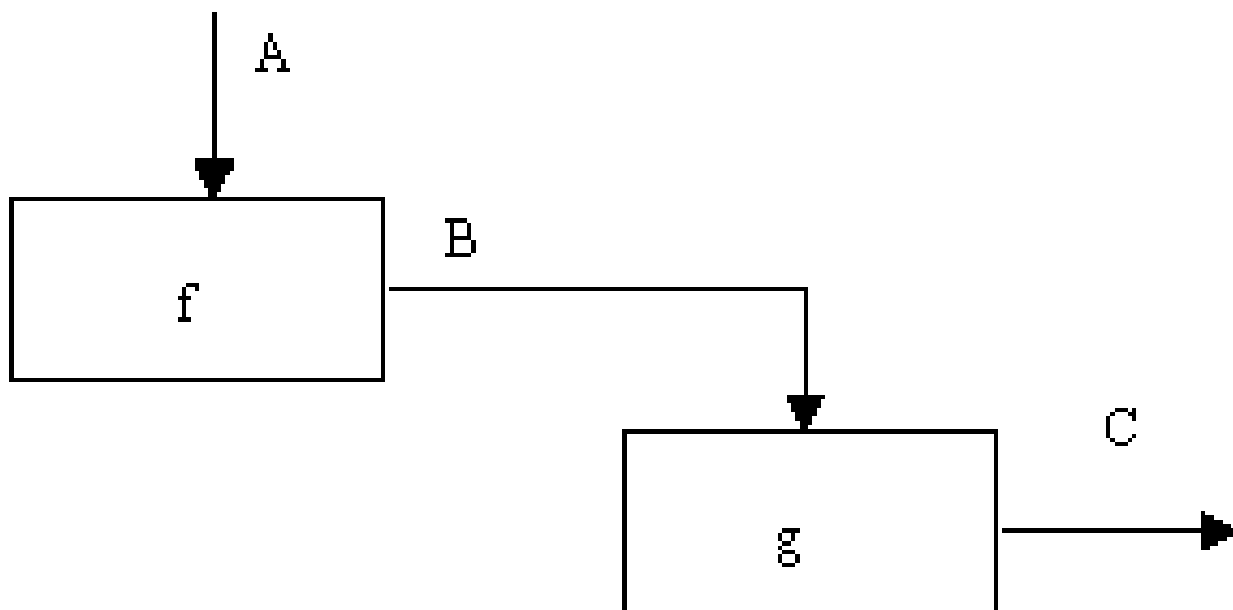


Рис. 12. Функциональная связность

Функциональная связь, показанная на рисунке 12, в математических терминах может иметь вид:

$$C = g(B) = g(f(A))$$

6. Процесс моделирования в SADT

Процесс моделирования в SADT является итеративной последовательностью шагов, приводящих к точному описанию системы.

Высокая эффективность SADT обусловлена разделением функций, выполняемых участниками создания SADT-проектов:

- *эксперты* являются источниками информации,
- *авторы* создают диаграммы и модели,
- *библиотекарь* координирует обмен письменной информацией,
- *читатели* рецензируют и утверждают модели,

- Комитет технического контроля принимает и утверждает модель.

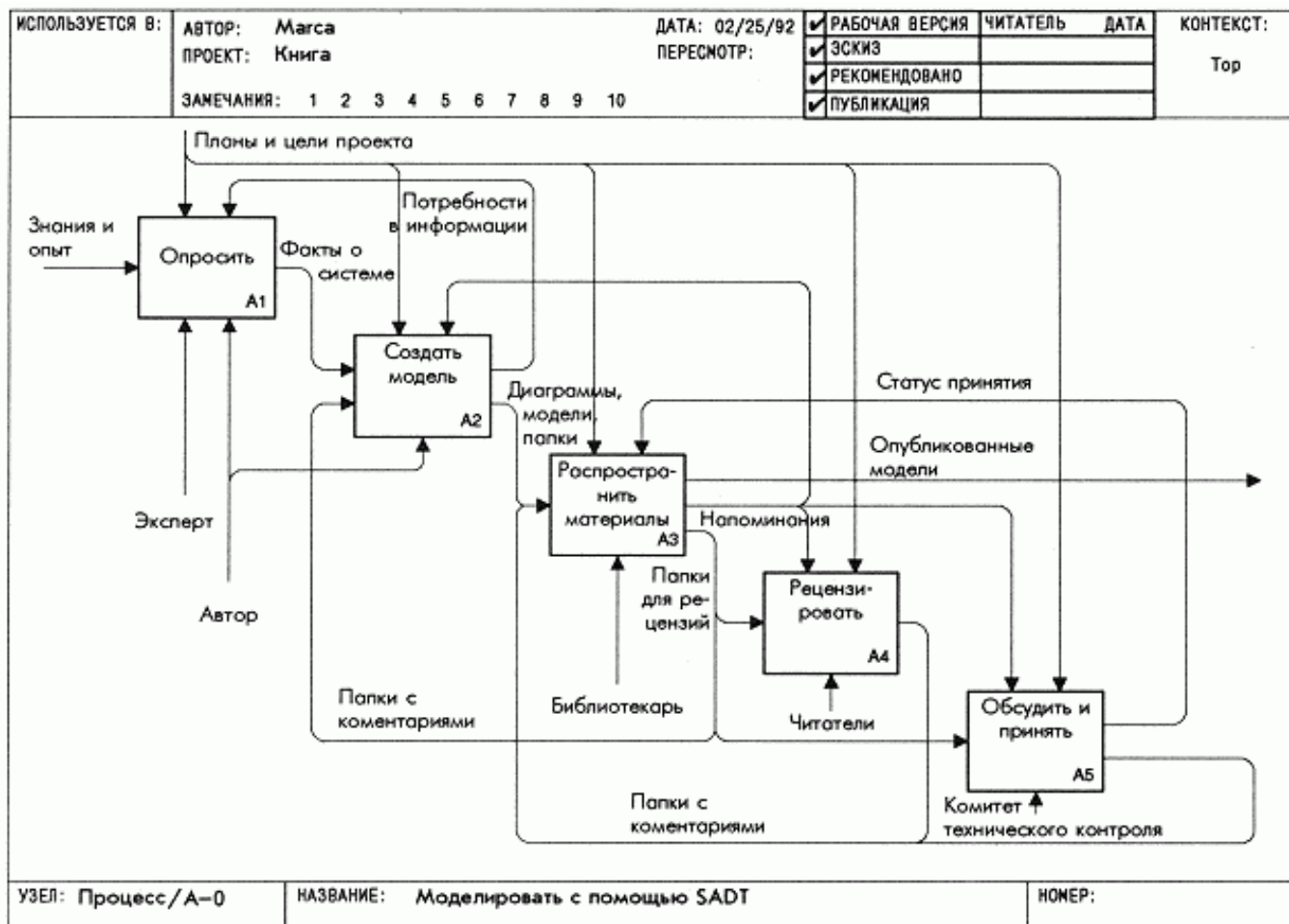


Рис. 1. Процесс создания SADT-модели, описанный с помощью SADT-диаграммы.

Рассмотрим процесс детальнее.

1. Получение знаний в процессе опроса

Сведения об изучаемой системе получают с помощью методики сбора информации - опросов или интервью экспертов. Независимо от конкретного источника информации SADT рекомендует руководствоваться определенной целью при его использовании. Это означает, что вы должны определить свои потребности в информации прежде, чем выбрать очередной источник.

2. Документирование полученных знаний

Это этап создания модели, на котором автор-аналитик документирует полученные им знания о данной проблемной области, представляя их в виде одной или нескольких SADT-диаграмм.

3. Корректность модели проверяется в процессе итеративного рецензирования

Итеративное рецензирование предполагает, что в процессе работы автор и эксперт многократно совещаются (устно и письменно) относительно достоверности создаваемой модели. Итеративное рецензирование называется циклом автор/читатель.

Цикл автор/читатель начинается, когда автор принимает решение распространить информацию о своей работе с целью получения отзыва о ней. Материал для распространения оформляется в виде "папок" - пакетов с результатами работы, которые обсуждаются другими специалистами-читателями.

Письменные замечания помещаются в папку в виде нумерованных комментариев. Папки с замечаниями являются обратной связью, которую авторы получают на свою работу. —лек3---

4. Координация процесса рецензирования

Библиотекарь играет роль наблюдателя за процессом рецензирования и является главным координатором процесса моделирования в SADT. Библиотекарь распространяет полученные от авторов папки, контролирует их движение, рассылает напоминания о своевременном возвращении авторам папок с замечаниями и о сроках ответов авторов на предложения читателей. Библиотекарь печатает законченные модели после того, как они одобрены и приняты к использованию.

5. Модели используются после их одобрения

Выделяется специальная группа людей - Комитет технического контроля. Комитет отвечает за точность модели и возможность ее использования в дальнейшем. Комитет следит за работой и ее соответствием конечным целям всего проекта.

7. Типы и назначения диаграмм в SADT

В настоящее время существуют различные типы диаграмм SADT, отличающиеся назначением. Можно назвать IDEF0, IDEF1, IDEF1X, IDEF2, IDEF3, IDEF4, IDEF5, IDEF6, IDEF8, IDEF9, IDEF14 и т.д.

IDEF0 — методика функционального моделирования — совокупность правил и процедур, предназначенных для **построения функциональной модели** объекта какой — либо предметной области. Функциональная модель объекта отображает производимые им действия и

связи между ними. Ранее нами рассмотрены все особенности диаграмм типа **IDEF0**.

Можно сказать, что диаграммы типа IDEF0 задают универсальный подход к моделированию любых предметных областей, а прочие типы диаграмм — специализированы для различных узких целей.

IDEF1X — методика информационного моделирования (моделирования данных). Цель его состоит в обеспечении разработчика системы **концептуальной схемой баз данных** в форме модели сущность — связь, предметной области,

которая может быть отображена в любую систему базы данных.

IDEF2 — методика поведенческого моделирования. Используется для определения **динамики функционирования сложных систем**. Поведенческая модель объекта описывает его функционирование как последовательность смены состояний. В основе данной методики лежат методы и модели имитационного моделирования систем, массового обслуживания сети Петри, а также теория конечных автоматов.

IDEF3 — методика моделирования процессов служит для создания таких моделей, в которых важно понять

последовательность выполнения действий и взаимозависимостей между ними.

IDEF4 — методика **объектно — ориентированного проектирования.** Предоставляет пользователю графический язык для изображения классов, диаграмм наследования, и др. элементов объектной модели.

IDEF5 — методика систематизации объектов приложений, направлена на представление **онтологической (принципиальной) структурной информации** приложения в удобном для пользователя виде. Для этого используется символические обозначения объектов, их

ассоциаций, ситуаций, и схемный язык описания отношений, классификация, часть – целое, переходов.

IDEF6 - методика, направленная на сохранение рационального **опыта проектирования ИС**, что способствует предотвращению повторных ошибок.

IDEF8 - методика, предназначенная для создания моделей, отображающих **взаимодействие человека и технической системы**.

IDEF9 - методика, предназначенная для **анализа имеющихся условий и ограничений**, в т.ч. **технических, юридических, и их влияние на**

принимаемые решения в процессе реинжиниринга.

IDEF14- методика **моделирования вычислительных сетей**, предназначена для графического представления и описания конфигураций, очередей сетевых компонентов, требований и надежности и т.д.

Тема: Диаграммы типа IDEF3.

1. Общее назначение IDEF3

IDEF3 является стандартом документирования технологических процессов, происходящих на предприятии, и предоставляет инструментарий для наглядного исследования и моделирования их сценариев.

Сценарием (Scenario) мы называем описание последовательности изменений свойств объекта, в рамках рассматриваемого процесса (например, описание последовательности этапов обработки детали в цеху и изменение её свойств после прохождения каждого этапа).

Исполнение каждого сценария сопровождается соответствующим документооборотом, который состоит из двух основных потоков:

- документов, определяющих структуру и последовательность процесса (технологических указаний, описаний стандартов и т.д.);
- документов, отображающих ход его выполнения (результатов тестов и экспертиз, отчетов о браке, и т.д.).

Для эффективного управления любым процессом, необходимо иметь детальное представление об его сценарии и структуре сопутствующего документооборота.

Средства документирования и моделирования IDEF3 позволяют выполнять следующие задачи:

- Документировать имеющиеся данные о технологии процесса, выявленные, скажем, в процессе опроса компетентных сотрудников, ответственных за организацию рассматриваемого процесса.

- Определять и анализировать точки влияния потоков сопутствующего документооборота на сценарий технологических процессов.

- Определять ситуации, в которых требуется принятие решения, влияющего на жизненный цикл процесса, например

изменение конструктивных, технологических или эксплуатационных свойств конечного продукта.

- Содействовать принятию оптимальных решений при реорганизации технологических процессов.

- Разрабатывать имитационные модели технологических процессов, по принципу "КАК БУДЕТ, ЕСЛИ..."

-

2. Типы диаграмм в IDEF3

Существуют два типа диаграмм в стандарте IDEF3, представляющие описание одного и того же сценария технологического процесса в разных ракурсах.

Диаграммы относящиеся к первому типу называются **диаграммами Описания Последовательности Этапов Процесса (Process Flow Description Diagrams, PFDD)**.

Диаграммы второго типа называются - **диаграммами Состояния Объекта в и его Трансформаций Процессе (Object State Transition Network, OSTN)**.

Предположим, требуется описать процесс окраски детали в производственном цеху на предприятии.

С помощью диаграмм PFDD документируется последовательность и описание стадий обработки детали в

рамках исследуемого технологического процесса.

Диаграммы OSTN используются для иллюстрации трансформаций детали, которые происходят на каждой стадии обработки.

На следующем примере, опишем, как графические средства IDEF3 позволяют документировать вышеуказанный производственный процесс окраски детали.

В целом, этот процесс состоит:

- непосредственно из самой окраски, производимой на специальном оборудовании;
- этапа контроля ее качества, который определяет, нужно ли деталь окрасить

заново (в случае несоответствия стандартам и выявления брака) или отправить ее в дальнейшую обработку.

2.1. Диаграммы типа PFDD

На рис.1 изображена диаграмма PFDD, являющаяся графическим отображением сценария обработки детали.

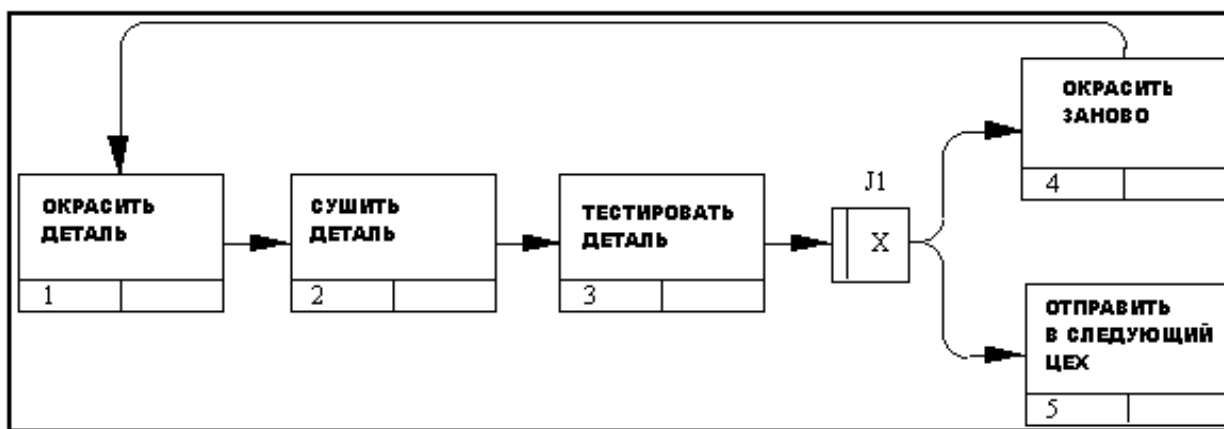


Рисунок 1. Пример PFDD диаграммы.

Сценарий, отображаемый на диаграмме, можно описать в следующем виде:

1) Деталь поступает в окрасочный цех, подготовленной к окраске.

2) В процессе окраски наносится один слой эмали при высокой температуре.

3) После этого, производится сушка детали, после которой начинается этап проверки качества нанесенного слоя.

4) Если тест подтверждает недостаточное качество нанесенного слоя (недостаточную толщину, неоднородность и т.д.), то деталь заново пропускается через цех окраски.

5) Если деталь успешно проходит контроль качества, то она отправляется в следующий цех для дальнейшей обработки.

Прямоугольники на диаграмме PFDD называются **функциональными элементами** или элементами поведения (Unit of Behavior, UOB) и обозначают событие, стадию процесса или принятие решения.

Каждый UOB имеет свое имя, отображаемое в глагольном наклонении и уникальный номер.

Стрелки или **линии** являются отображением перемещения детали между UOB-блоками в ходе процесса.

Линии бывают следующих видов:

- **Старшая** (Precedence) - сплошная линия, связывающая UOB. Рисуеться слева направо или сверху вниз.

- **Отношения** (Relational Link)- пунктирная линия, используемая для изображения связей между UOB

- **Потоки объектов** (Object Flow) - стрелка с двумя наконечниками используется для описания того факта, что объект (деталь) используется в двух или более единицах работы, например, когда объект порождается в одной работе и используется в другой.

Объект, обозначенный **J1** - называется **перекрестком** (Junction). Перекрестки используются для отображения логики

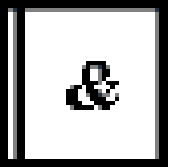
взаимодействия стрелок (потоков) при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы.

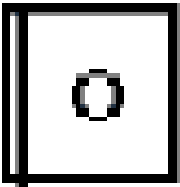
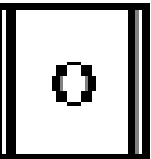
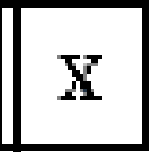
Различают перекрестки для слияния (Fan-in Junction) и разветвления (Fan-out Junction) стрелок.

Перекресток не может использоваться одновременно для слияния и для разветвления. При внесении перекрестка в диаграмму необходимо указать тип перекрестка. Классификация возможных типов перекрестков приведена в таблице 1.

----лек4--

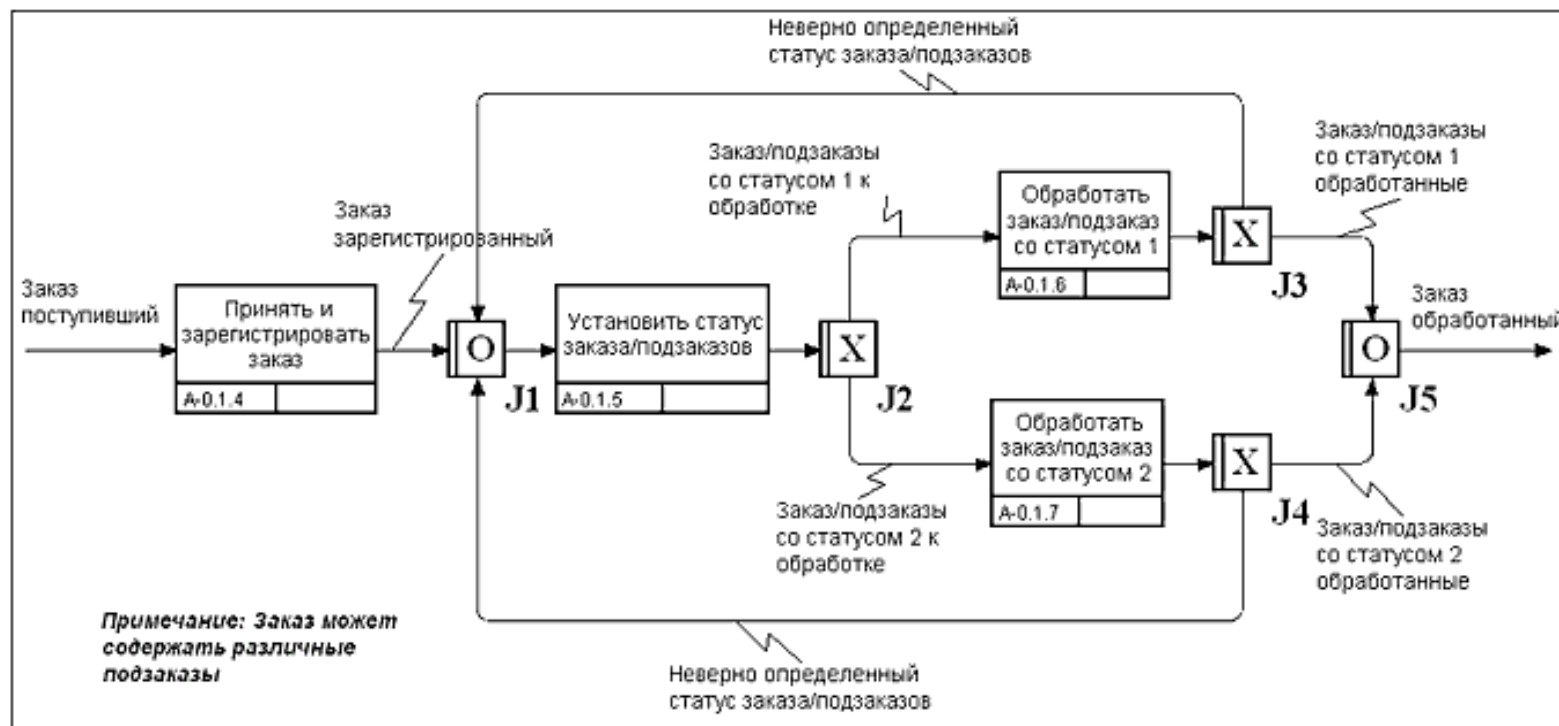
Таблица 1. Типы перекрестков:

Изображение	Наименование	Смысл в случае слияния стрелок	Смысл в случае разветвления стрелок
	Асинхронное И.	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены
	Синхронное И.	Все предшествующие процессы завершены <i>одновременно</i>	Все следующие процессы запускаются <i>одновременно</i>

	<p>Асинхронное ИЛИ.</p>	<p>Один или несколько предшествующих процессов должны быть завершены</p>	<p>Один или несколько следующих процессов должны быть запущены</p>
	<p>Синхронное ИЛИ.</p>	<p>Один или несколько предшествующих процессов завершаются <i>одновременно</i></p>	<p>Один или несколько следующих процессов запускаются <i>одновременно</i></p>
	<p>Исключающе е ИЛИ.</p>	<p>Только один предшествующий процесс завершен</p>	<p>Только один следующий процесс запускается</p>

Все перекрестки в PFDD диаграмме нумеруются, каждый номер имеет префикс "J".

На рис 3. приведена более детальная диаграмма PFDD.



Пример IDEF3-диаграммы

Рисунок 1. Пример более детальной PFDD диаграммы.

Каждый функциональный блок UOV может иметь последовательность **декомпозиций**, и, следовательно, может быть детализирован с любой необходимой точностью.

Под декомпозицией мы понимаем представление каждого UOV с помощью отдельной IDEF3 диаграммы. Например, мы можем декомпонировать UOV "Окрасить Деталь", представив его отдельным процессом и построив для него свою PFDD диаграмму.

При этом эта диаграмма будет называться **дочерней**, по отношению к изображенной на рис. 1, а та, соответственно **родительской**.

Номера UOB дочерних диаграмм имеют сквозную нумерацию, т.е., если родительский UOB имеет номер "1", то блоки UOB на его декомпозиции будут соответственно иметь номера "1.1", "1.2" и т.д.

Применение принципа декомпозиции в IDEF3 позволяет структурировано описывать процессы с любым требуемым уровнем детализации.

2.2. Диаграммы типа OSTN

Если диаграммы PFDD технологический процесс "С точки зрения наблюдателя", то другой класс диаграмм IDEF3 OSTN позволяет рассматривать тот же самый процесс "С точки зрения объекта".

На рис.2 представлено отображение процесса окраски с точки зрения OSTN диаграммы.

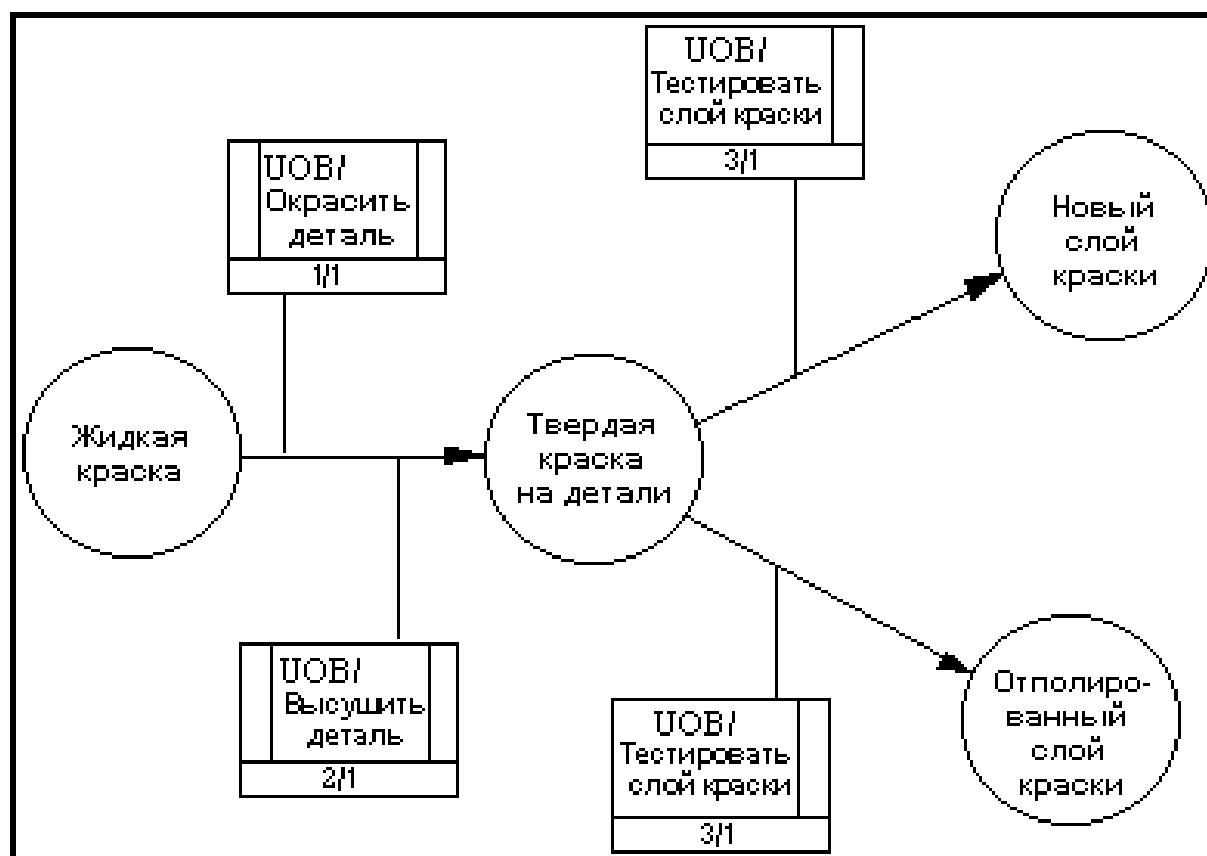


Рисунок 2. Пример OSTN диаграммы

Состояния объекта (в нашем случае детали) и **Изменение состояния** являются ключевыми понятиями OSTN диаграммы.

Состояния объекта отображаются окружностями, а их изменения направленными линиями. Каждая линия имеет ссылку на соответствующий функциональный блок UOB, в результате которого произошло отображаемое ей изменение состояния объекта.

Тема 5. Интерфейс SADT с системой имитационного моделирования Arena

1) Характеристика системы Arena

Одним из наиболее эффективных инструментов имитационного моделирования является система [Arena](#) компании [Systems Modeling](#).

Arena позволяет строить имитационные модели, проигрывать их и анализировать результаты такого проигрывания.

Основа Arena - язык моделирования SIMAN и анимационная система Cinema Animation. SIMAN - гибкий и

выразительный язык моделирования. Система Cinema animation используется для отображения результатов моделирования. Процесс моделирования организован следующим образом. Сначала пользователь шаг за шагом строит в визуальном редакторе системы Arena модель. Затем система генерирует по ней соответствующий код на SIMAN, после чего автоматически запускается Cinema animation.

Имитационная модель Arena включает следующие основные элементы: источники и стоки (Create и Dispose), процессы (Process) и очереди (Queue).

Источники - это элементы, от которых в модель поступает информация или объекты. Скорость поступления данных или объектов от источника задается статистической функцией.

Сток - это устройство для приема информации или объектов.

Очередь - это место, где объекты ожидают обработки.

Времена обработки объектов (производительность) в разных *процессах* могут быть разными. В результате перед некоторыми процессами могут накапливаться объекты, ожидающие своей очереди.

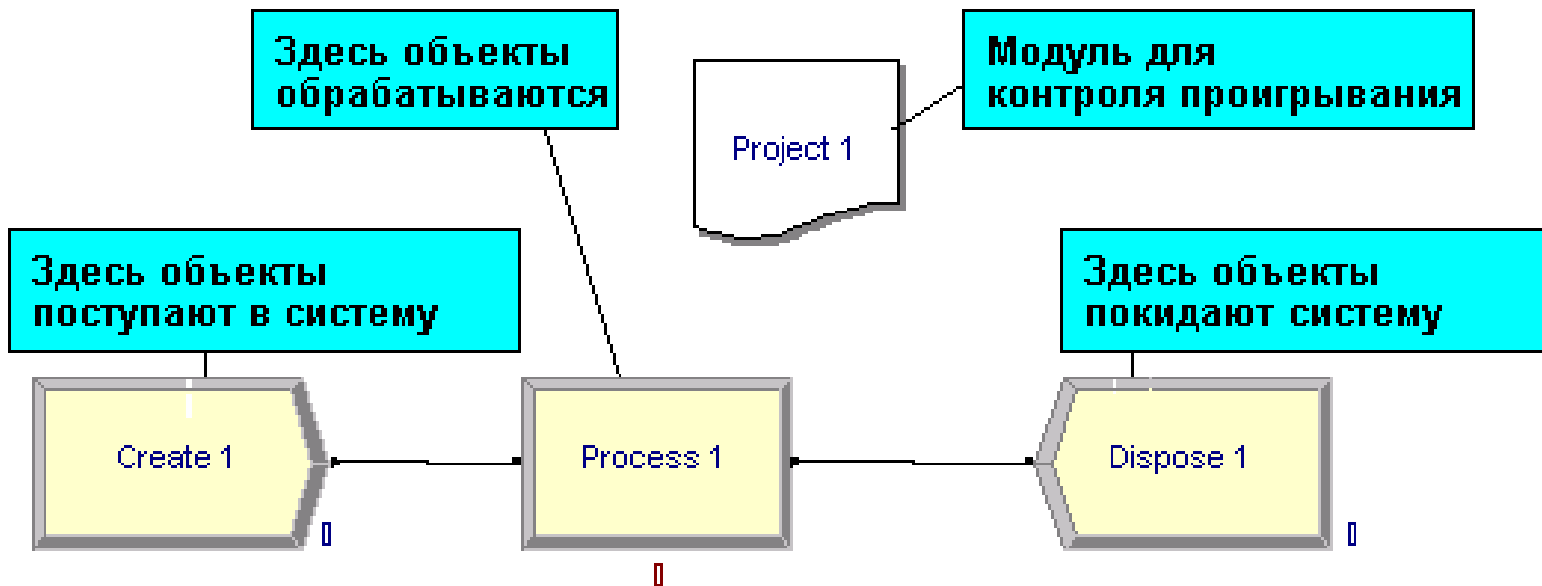


Рис. 1. Простейшая имитационная модель.

Для построения моделей Arena имеет палитру инструментов. Палитра инструментов содержит два типа модулей.

Модули типа “Flowchart” (в том числе Create, Dispose и Process) служат для отображения потоков объектов и могут быть перенесены на рабочее пространство

модели (аналог Simulink) с автоматическим формированием связей.

Модули типа “Data” (например Queue) не могут быть размещены в рабочее пространство модели и служат для настройки параметров модели.

2) Интеграция Vrpwin с системой Arena

Создавать имитационные модели без предварительного анализа бизнес-процессов не всегда представляется возможным. Не поняв сути бизнес-процессов предприятия бессмысленно пытаться оптимизировать конкретные технологические процессы.

Поэтому функциональные модели и имитационные модели не заменяют, а дополняют друг друга, при этом они могут быть тесно взаимосвязаны. Имитационная модель дает больше информации для анализа системы, в свою очередь результаты такого анализа могут стать причиной модификации модели процессов.

Наиболее целесообразно сначала создать функциональную модель, а затем на ее основе строить модель имитационную. Для поддержки такой технологии инструментальное средство функционального моделирования ВРwin имеет возможность преобразования

диаграмм **IDEF3** в имитационную модель Arena.

Поскольку имитационная модель имеет гораздо больше параметров, чем диаграмма IDEF3, в BPwin имеется возможность задать эти параметры с помощью свойств, определяемых пользователем (UDP).

Совместное использование инструмента построения функциональной модели BPwin и системы имитационного моделирования Arena позволяет наиболее эффективно оптимизировать технологические процессы практически в любой сфере деятельности.

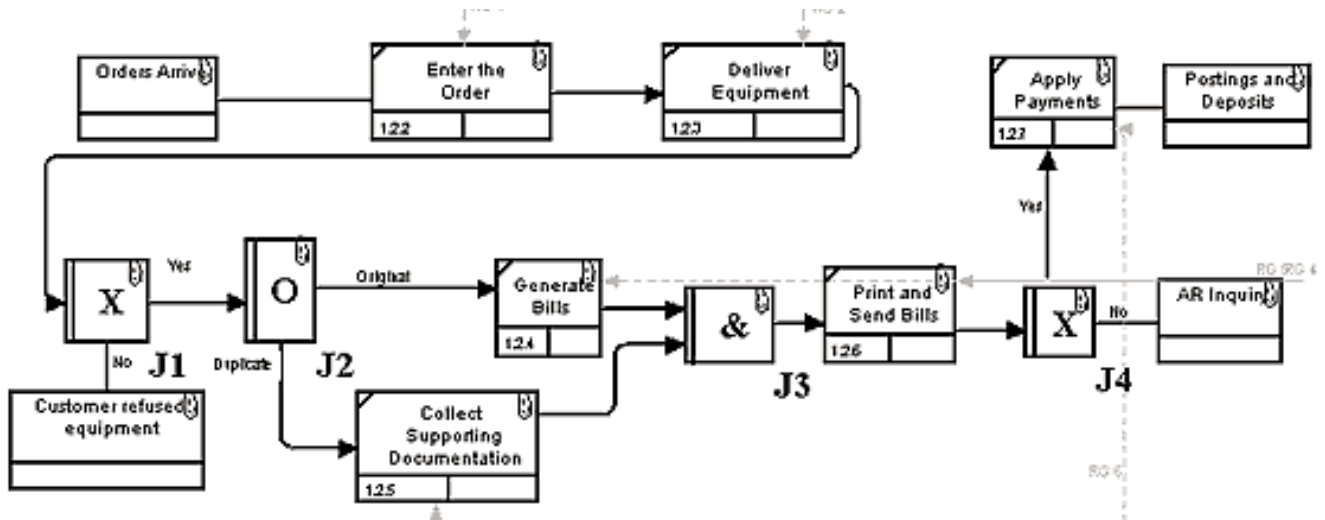


Рис. 5. Диаграмма IDEF3 – пример для иллюстрации экспорта в Arena.

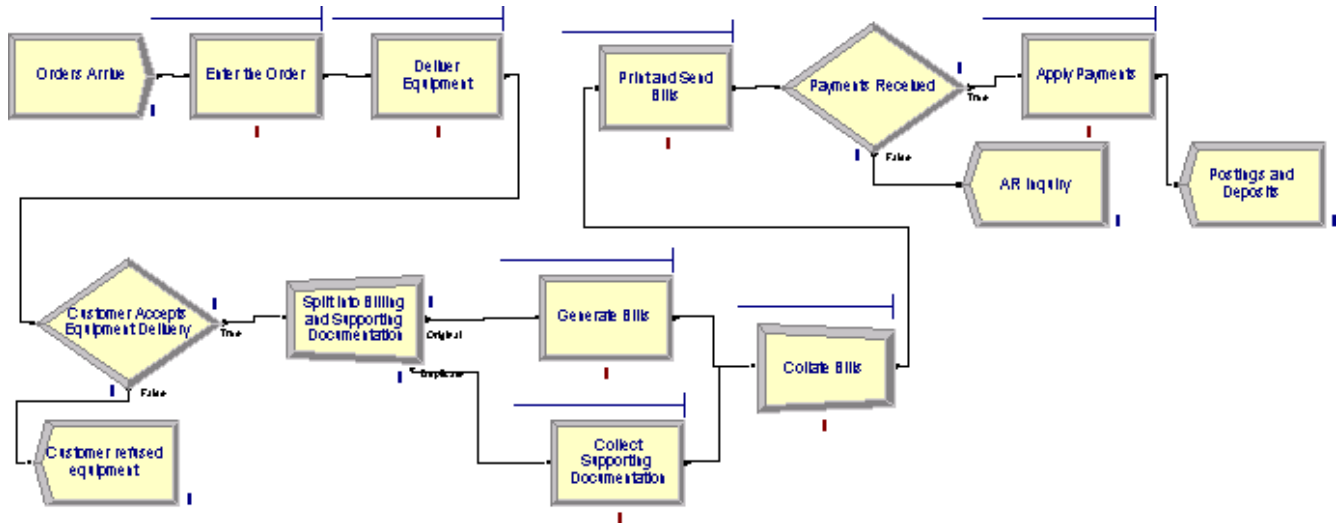


Рис. 6. Имитационная модель Arena – результат импорта из VProwin.

---лек5---

Тема: Пакет VpWin (AllFusion Process Modeler)

1) Порядок создания моделей IDEF0.

Рассмотрим порядок разработки средствами диаграмм IDEF0 моделей бизнес-процессов, имеющихся в вербальной модели для заданной предметной области.

Нажав правую кнопку на начальном блоке в разделе «font» выбрать русский язык и применить его ко всем блокам.

Нажав правую кнопку и выбрав «name», задать имя блока.

Аналогично сделать для всех стрелочек (связей).

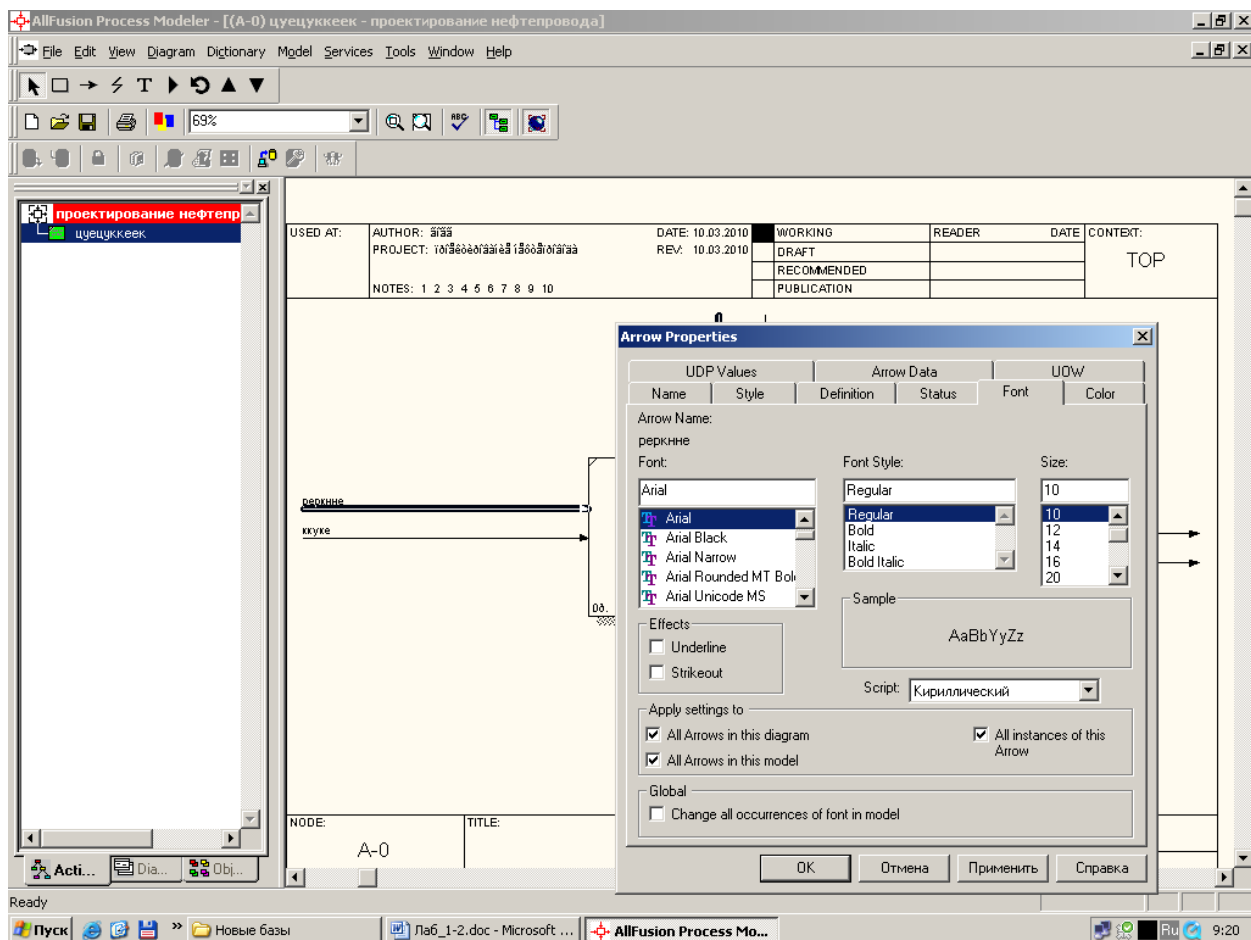


Рис. Пример установки русского языка для всех стрелочек.

Если нужно, что бы надписи на стрелочках были чуть в стороне, необходимо воспользоваться «молнией» на меню пиктограмм. Это позволит перетаскивать надпись с «молнией».

Примечание. В блоке всегда:

- сверху — входная управляющая информация (ГОСТ, приказы правила, методики и т.д.);
- слева — входные данные (сырье, материалы, требования ТЗ, т.е. мощность и т.д.);
- снизу — вход как механизм, т.е. материальные средства достижения цели (проектировщики, инженеры, рабочие,

станки, базы данных, компьютеры, лопаты и т.д.);

- справа – выход, т.е. результат работы (состав чертежей, спецификации к ним, структура сети, программы управления и т.д.).

Встав на имя блока в верхнем левом углу экрана, и нажав контекстную кнопку, выберем команду «decompose». Это позволит разбить исходный блок на нужное число подблоков.

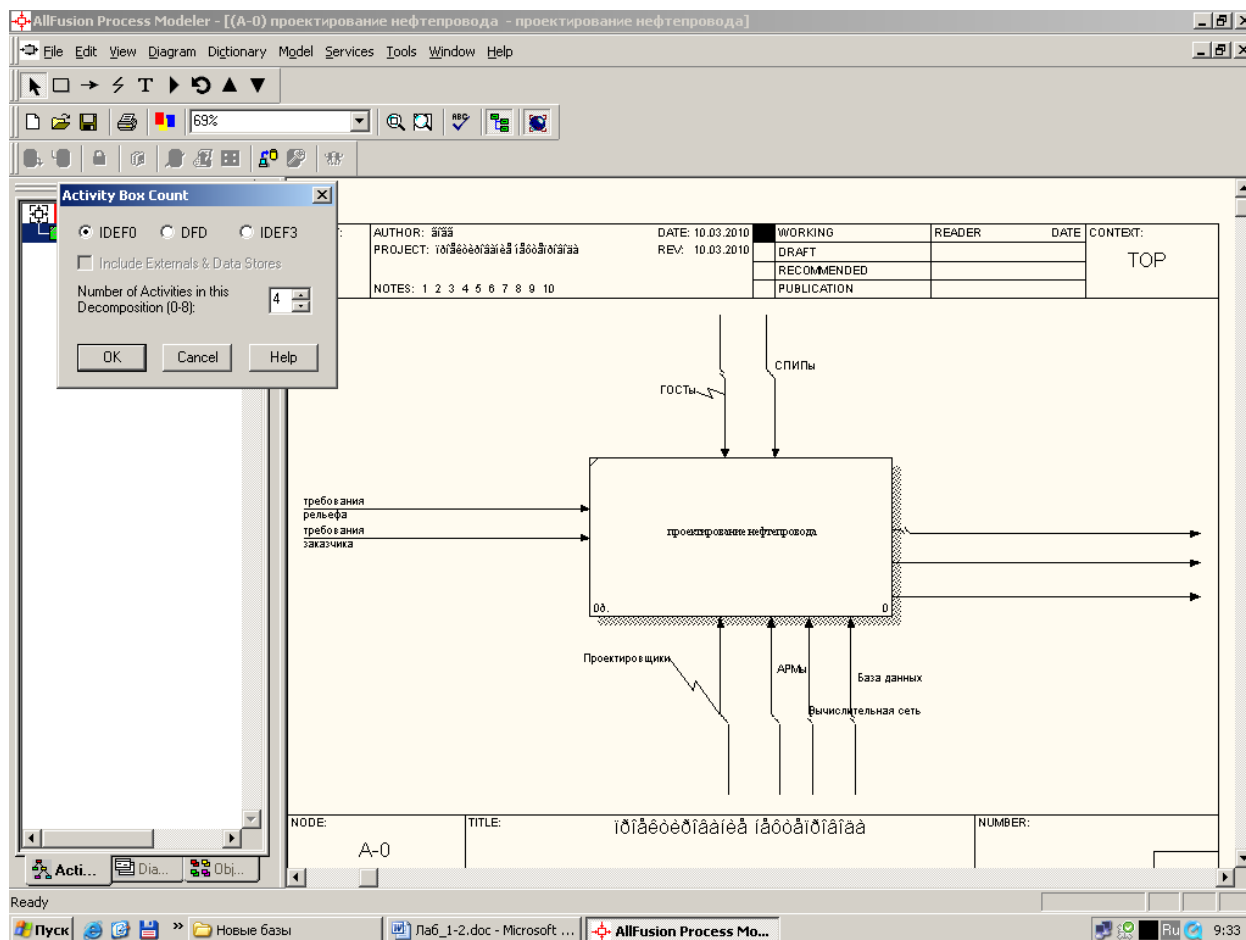


Рис. Декомпозиция блока.

Выбрав стрелочку (связь) на пиктографическом меню получим курсор в виде «крестика». Постараемся попасть на кончик входной стрелочки. Это позволит отправить ее на нужный подблок.

Выбрать форму блоков, исходя из их типа (работа с базой данных, работа с проектировщиком и т.д.)

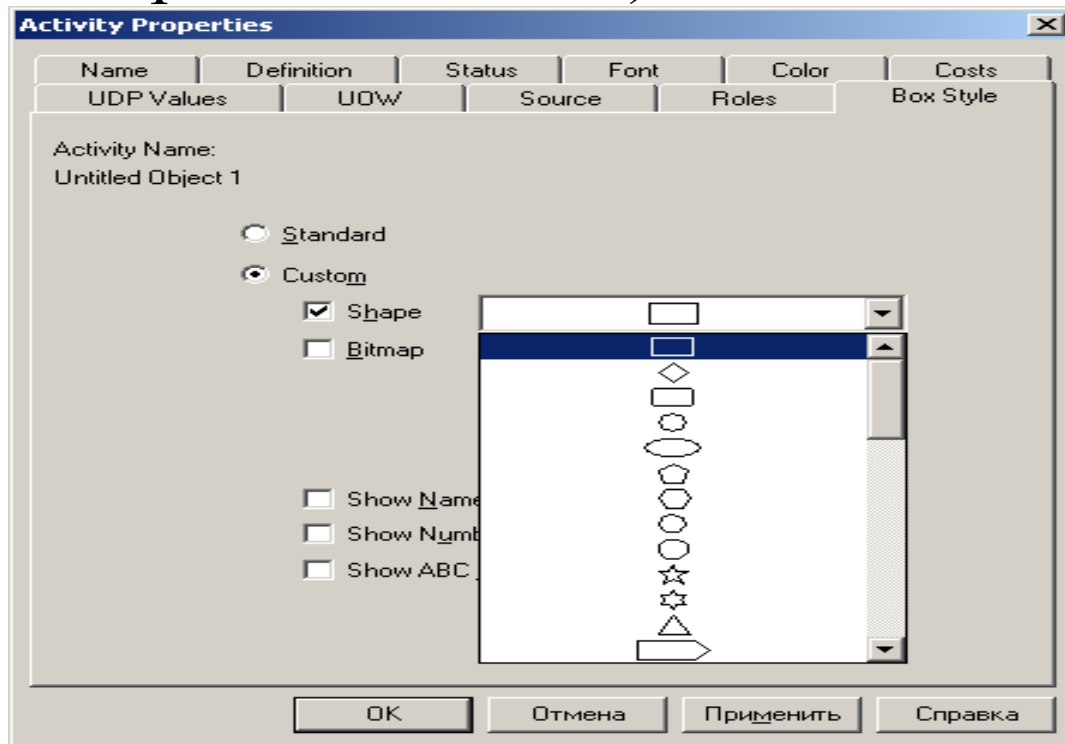


Рис. Меню выбора формы блоков.

Соответственно раскрасить блоки.

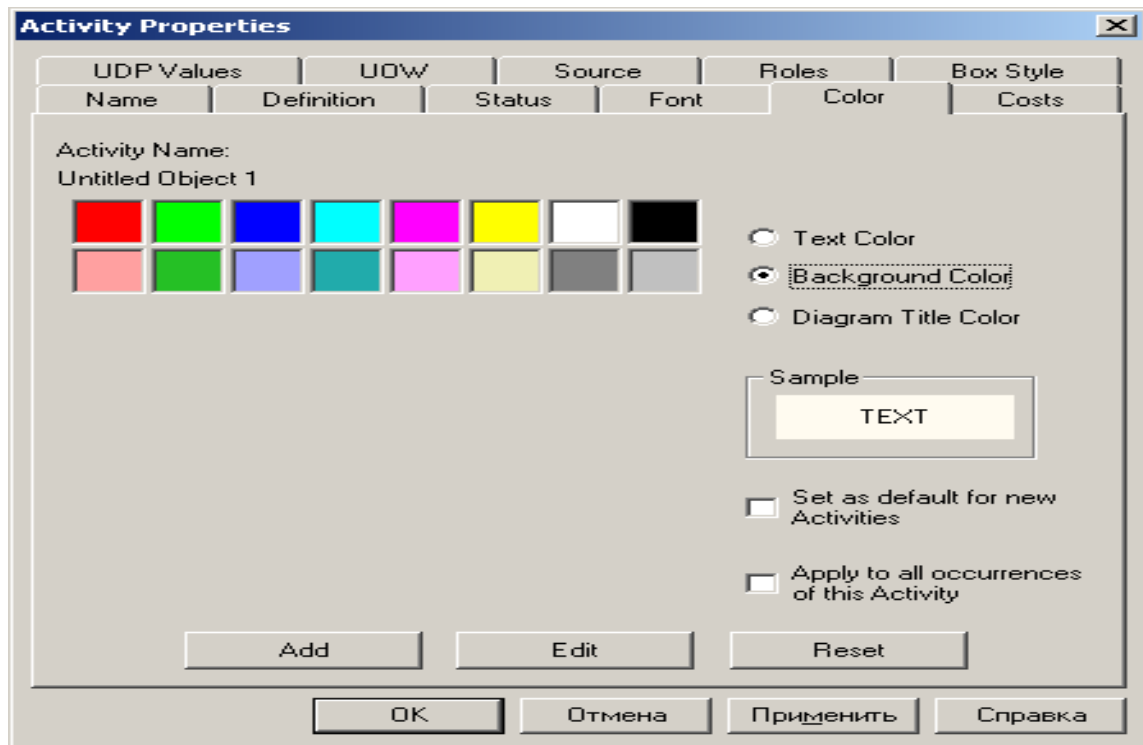


Рис. Меню

2) Порядок создания моделей IDEF3.

Порядок создания моделей ***IDEF3*** аналогичен порядку создания моделей ***IDEF0***.

Отличие - наличие спец. меню для вставки блоков – разветвителей – И, ИЛИ и т.д.

Тема: Диаграммы типа IDEF1.

1) Описание стандарта IDEF1.

IDEF1 был разработан как инструмент для анализа и изучения взаимосвязей между информационными потоками в рамках коммерческой деятельности предприятия.

IDEF1X - предназначен для разработки структуры реляционных баз данных и оперирующего с конкретными объектами физического мира.

Основной целью использования методологии IDEF1 есть исследование движения потоков информации и принципов управления ими на начальном этапе процесса проектирования

корпоративной информационно-аналитической системы.

Методология IDEF1 позволяет достаточно наглядно обнаружить "черные дыры" и слабые места в существующей структуре информационных потоков. Позволяет решить следующие задачи:

- Выяснить структуру и содержание существующих потоков информации на предприятии.
- Определить, какие проблемы, выявленные в результате функционального анализа и анализа потребностей, вызваны недостатком

управления соответствующей информацией.

- Выявить информационные потоки, требующие дополнительного управления для эффективной реализации модели.

При построении информационной модели проектировщик всегда оперирует с двумя основными глобальными областями.

Первой является *реальный мир*, или же совокупность физических и интеллектуальных объектов, таких, как люди, места, вещи, идеи и т.д., а также все свойства этих объектов и зависимости между ними.

Второй же является *информационная область*. Она включает в себя существующие информационные отображения объектов первой области и их свойств.

Информационное отображение не является объектом реального мира, но изменение его является следствием некоторого изменения соответствующего ему объекта реального мира.

Методология IDEF1 разработана как инструмент для исследования статического соответствия вышеуказанных областей и установления строгих правил и механизмов изменения объектов информационной

области при изменении соответствующих им объектов реального мира.

IDEF1 является аналитическим методом и используется преимущественно для выполнения следующих действий:

- Определения самой информации и структуры ее потоков, имеющей отношение к деятельности предприятия;
- Определение существующих правил и законов, по которым осуществляется движение информационных потоков, а также принципов управления ими.
- Выяснение взаимосвязей между существующими информационными потоками в рамках предприятия;

- Выявление проблем, возникающих вследствие недостатка качественного информационного менеджмента.

Основные преимущества IDEF1

Методология IDEF1 позволяет на основе простых графических изображений моделировать информационные взаимосвязи и различия между:

1. Реальными объектами;
2. Физическими и абстрактными зависимостями, существующими среди реальных объектов;
3. Информацией, относящейся к реальным объектам;

4. Структурой данных, используемой для приобретения, накопления, применения и управления информацией.

Другим отличительным свойством IDEF1 является широко развитая модульность.

Тема: Интеграция ВРwin с системой проектирования баз данных ERwin

Диаграммы **IDEF1X** в ВРwin предназначены для разработки структуры реляционных баз данных и оперируют с конкретными объектами физического мира.

Однако, имеется специализированная система, ориентированная на проектирование баз данных - ERwin, работающая сходным образом и поддерживающая интерфейс с ВРwin.

1) Общая характеристика процесса проектирования баз данных с ERwin

Модель данных представляется в ERwin как система взаимосвязанных ***сущностей*** различного типа, имеющих ***атрибуты***, которые, в свою очередь, обладают некоторыми ***значениями***.

Сущности представляют собой объекты, информацию о которых необходимо накапливать и сопровождать. Они являются "контейнерами" для организации и группировки бизнес-фактов (в данном случае – это таблицы БД).

Характеристики сущности определяются содержащимися в ней атрибутами.

Атрибуты сущности (*т.е. - поля таблицы*) представляют факты, касающиеся сущности, которые корпорация заинтересована накапливать и сопровождать.

Наиболее важные сущности обычно выявляются и фиксируются в документах в процессе некоторых рабочих сессий или индивидуальных интервью с экспертами в предметной области.

Сущности также могут выявляться в результате выполнения процесса приведения модели данных к некоторым стандартным формам представления, так называемой нормализации.

Суть нормализации — выявление и удаление многозначностей, избыточности и некорректности в модели данных путем создания новых сущностей.

Сущности делят на две основные группы: *зависимые и независимые*.

Зависимым сущностям для уникальной идентификации экземпляра требуется информация из других сущностей, независимым — нет.

В рамках двух основных групп сущностей выделяются более специализированные типы, имеющие некоторые особенности для поддержки конкретных видов отношений между основными и подчиненными сущностями

(*стержневые - независимые, кодовые - ключи, характеристические - зависимые*).

Каждая сущность должна включать один или несколько наборов атрибутов, являющихся **«кандидатами в ключи»**.

Кандидаты в ключи позволяют строить собственно *первичные ключи*, т.е. средства, призванные уникально идентифицировать конкретные **экземпляры** сущности (строки таблицы).

Кандидаты в ключи могут состоять из одного атрибута или из группы атрибутов. Если кандидатов в ключи не существует, или их трудно сопровождать, имеется возможность создать *искусственный первичный ключ*.

При определении первичных ключей важно обеспечить их уникальность и надежность с течением времени.

Для идентификации сущностей используются имена и описания. Имеются стандарты и соглашения об именовании, которые обеспечивают целостный подход к разработке имен и описаний.

2. Введение в реляционную диаграмму сущности

Для визуального представления сущностей и отношений между ними используются ERD-диаграмма (Entity Relational Diagram - реляционная диаграмма

сущности), основанная на нотации, используемой **ERwin**.

Существуют следующие методологии моделирования данных:

- 1) расширенный реляционный анализ (Extended Relational Analysis - ERA),
- 2) объектно-ориентированный подход (Object Oriented - OO)
- 3) объектно-ролевое моделирование (Object Role Modeling - ORM), и т.д.

Во всех названных методологиях присутствуют фундаментальные концепции ER. Методология ER-моделирования разработана П. Ченом в конце 1970-х годов. Для представления сущностей в

методологии ER используются прямоугольники.

В исходной ER-нотации Чена отношения содержат атрибуты. Равная возможность использования атрибутов как в сущностях так и в отношениях делает различие между сущностями и отношениями достаточно сложным.

С течением времени ER-подход изменялся и расширялся, но базовые концепции продолжали обеспечивать надежную основу для грамотного моделирования данных.

Сущности не предназначены для представления единичного объекта, они представляют набор экземпляров,

содержащих информацию, представляющую интерес с точки зрения их уникальности.

Конкретный экземпляр сущности представляется строкой таблицы и идентифицируется первичным ключом.

--лек6---

Сущность имеет следующие признаки:

- Она имеет имя и описание.
- Она представляет класс, а не единичный экземпляр абстракции.
- Ее конкретные представители (экземпляры) могут быть уникально идентифицированы.

- Она содержит логическую группировку атрибутов, представляющих информацию, интересную с точки зрения корпорации.

Независимая сущность не нуждается в информации из другой сущности для идентификации уникального экземпляра. Она представляется в **ERwin** в виде *прямоугольника*. Первичный ключ независимой сущности не включает в себя первичных ключей других сущностей.

Зависимая сущность должна привлекать информацию из другой сущности для идентификации уникального экземпляра. Она представляется на ER-диаграмме в виде *прямоугольника с закругленными углами*.

Первичный ключ зависимой сущности включает первичные ключи одной или более родительских сущностей.

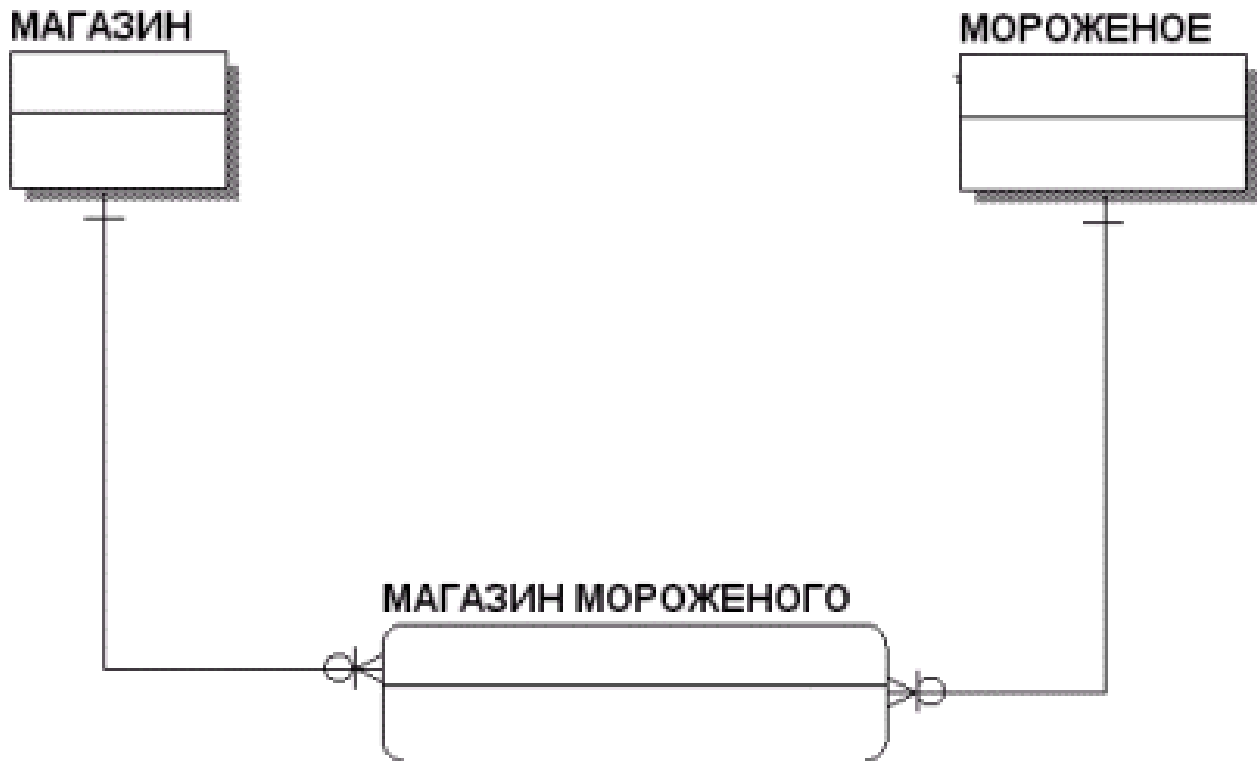


Рис. 1. Примеры сущностей для корпорации, торгующей мороженым.

Тут «магазин» и «мороженое» - независимые родительские сущности, «магазин мороженого» - дочерняя зависимая сущность.

Внутри квадрата могут быть описаны атрибуты сущности. Например, часть торта – его верхушка описывается сущностью.

ВЕРХУШКА

Идентификатор верхушки
Название верхушки
Описание верхушки

Фактически – это описание таблицы «ВЕРХУШКА», хранящей в

индексированных строчках в поле «названия верхушки» различные значения - формы верхушек.

Тут над чертой в сущности вводится индекс-идентификатор сущности, а под чертой – название значимого атрибута («название верхушки») и описание сущности в целом.

Идентификатор, или первичный ключ сущности это атрибут, позволяющий однозначно идентифицировать экземпляры, т.е. строки в сущности.

Имя, присваиваемое сущности («ВЕРХУШКА»), должно характеризовать экземпляры сущности. Имя должно быть понятным и общепринятым, осмысленным

для сообщества пользователей и экспертов предметной области. Имена должны отражать способ использования данных в рамках корпорации.

Имен обычно недостаточно. Каждая сущность нуждается в ясном, точном и полном *описании* или определении. Описание сущности должно объяснять смысл сущности и ее значение для корпорации.

3. Определение типов сущностей

И зависимые, и независимые сущности можно разделить на несколько типов:

- *Стержневые сущности* - основные или первичные сущности. Они

представляют наиболее важные объекты, информацию о которых следует хранить.

- *Коды/ссылки/классификаторы* - эти сущности содержат строки, определяющие набор значений или область определения для атрибута.

- *Характеристические сущности* - эти сущности бывают двух типов: исключающие и включающие.

- *Ассоциативные сущности* - эти сущности используются для разрешения отношений **многие-ко-многим**.

3.1. Стержневые сущности.

Стержневые сущности - наиболее важные корпоративные информационные

объекты. Стержневые сущности необходимо моделировать в виде масштабируемых и расширяемых контейнеров независимо от текущего способа ее использования. В нашем примере сущность МОРОЖЕНОЕ полностью вне контекста сущности МАГАЗИН и наоборот. Так что если в корпорации решат продавать МОРОЖЕНОЕ через новый канал сбыта, он может быть добавлен без изменений в других сущностях.

3.2. Кодовые сущности

Кодовые сущности всегда независимы. Их часто называют классификаторами или сущностями типов. Уникальные

экземпляры, представляемые кодовыми сущностями, определяют область определения для значений атрибутов, принадлежащих другим сущностям. Сущность «верхушка» — это кодовая сущность.

3.3. Характеристические сущности

Характеристические сущности всегда являются зависимыми и предназначены для хранения различных наборов значений атрибутов для экземпляров сущностей (т.е. это вариант обычной таблицы).

Характеристические сущности всегда имеют одну или более "равноправных" сущностей. Равноправные
характеристические сущности связаны с

родительской сущностью особым типом отношений, которые могут быть *исключающими или включающими*.

Равноправные характеристические сущности, которые находятся в исключающем отношении к родительской сущности, указывают на то, что только одна из равноправных сущностей содержит данные для некоторого экземпляра родительской сущности. Например:

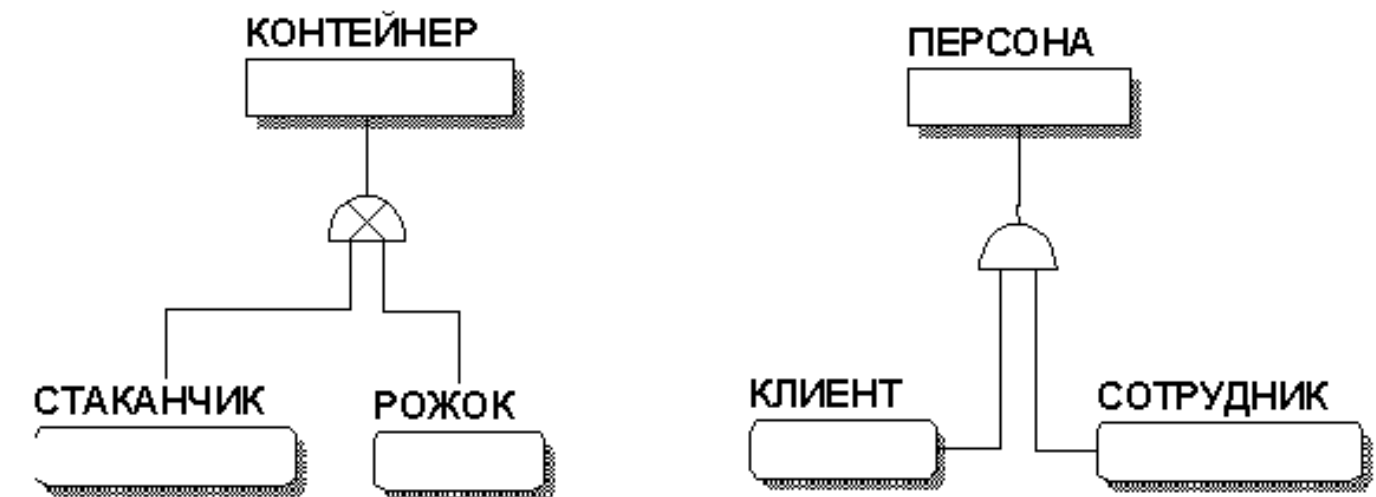


Рисунок 2.

На рисунке 2 представлена сущность КОНТЕЙНЕР и характеристические сущности РОЖОК и СТАКАНЧИК. Экземпляр КОНТЕЙНЕРА должен быть РОЖКОМ или СТАКАНЧИКОМ. КОНТЕЙНЕР не может быть одновременно и РОЖКОМ и СТАКАНЧИКОМ. Это

исключающие характеристические сущности.

Сущность ПЕРСОНА на рисунке 2 имеет две характеристические сущности СОТРУДНИК и КЛИЕНТ. Это пример включающих характеристических сущностей. Отсутствие (X) в символе характеристической сущности указывает на *включающее отношение*.

На практике СОТРУДНИК может быть КЛИЕНТОМ, а ПОСТАВЩИК может выступать в качестве КЛИЕНТА. Исключающие характеристические сущности не позволили бы одному экземпляру ПЕРСОНЫ содержать факты, общие для СОТРУДНИКА и КЛИЕНТА.