

Структурное проектирование ИС

Вопросы

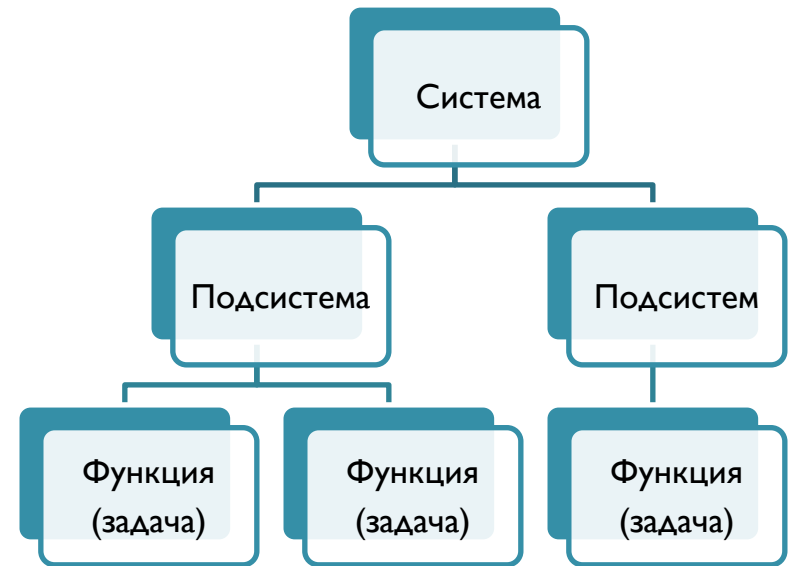
1. Сущность структурного подхода к проектированию ИС
2. Методология структурного анализа и проектирования SADT (нотация IDEF0, IDEF1X)
3. Методология структурного системного анализа Гейна –Сарсона (диаграммы потоков данных DFD)

Формальное определение метода проектирования

- Концепции и теоретические основы (структурный или объектно-ориентированный подход)
- Нотация – способ отображения моделей статической структуры и динамики поведения проектируемой системы (графические диаграммы, математическая формализация – множества, графы, сети Петри)
- Процедуры, определяющие практическое применение метода (последовательность и правила построения моделей, критерии, используемые для оценки результатов)

Сущность структурного подхода

Заключается в декомпозиции системы, которая производится следующим образом: система разбивается на функциональные подсистемы, которые делятся на функции, те – на задачи и так далее до конкретных процедур обработки данных.



Принципы структурного подхода

В основе структурного подхода лежат следующие принципы:

- принцип декомпозиции (научный метод, использующий структуру задачи и позволяющий заменить решение одной большой задачи решением серии меньших задач, разбить целое на части, решение по частям);
- принцип иерархического упорядочения функций (организация составных частей системы в иерархические древовидные структуры с добавлением новых деталей на каждом уровне);
- принцип абстрагирования (выделение существенных аспектов системы и отвлечение от несущественных) – метод «черных ящиков»;
- принцип непротиворечивости (обоснованность и согласованность элементов системы);
- принцип структурирования данных (данные должны быть структурированы и иерархически организованы).

Модель функции обработки данных в виде потока данных

$$\text{ВЫХОД} = F(\text{ВХОД})$$

Функция (Активность) – это преобразователь объектов (данных)

ВХОД-ОБРАБОТКА-ВЫХОД:

1. Исходные объекты (данные) входят в систему,
2. обрабатываются,
3. Результатные объекты (данные) выходят из системы.

Такая модель используется во всех структурных методологиях.

Функции взаимодействуют по входам и выходам: выход одной функции является входом другой функции



Методологии структурного анализа и проектирования

Методология структурного анализа и проектирования определяет шаги процесса, которые должны быть выполнены, их последовательность, правила распределения и назначения операций и средств.

В настоящее время успешно используются практически все известные методологии структурного анализа и проектирования, однако наибольшее распространение получили методологии:

- структурного анализа и техники проектирования SADT (Structured Analysis and Design Technique), Росс, Марка – МакГоун
- структурного системного анализа Гейна-Сарсона (Gane-Sarson),
- структурного анализа и проектирования Йодана/Де Марко (Yourdon/De Marko),
- развития систем Джексона (Jackson),
- информационного моделирования Мартина (Martin).

2. Методология структурного анализа и проектирования

Методология SADT (Structured Analysis and Design Technique) разработана Россом в семидесятые годы прошлого столетия и является методологией, ориентированной на описание взаимодействия функций (процессов).

С точки зрения SADT модель может основываться либо на функциях системы, либо на ее объектах (предметах). Соответствующие модели принято называть функциональными (активностными) моделями и моделями данных.

- **Функциональная модель** представляет с нужной степенью точности систему функций, которые в свою очередь отражают свои взаимодействия через объекты системы.
- **Модели данных** двойственны к функциональным моделям и представляют собой подробное описание объектов системы, связанных системными функциями.

Полная методология SADT заключается в построении моделей обоих типов для более точного описания сложной системы.

Развитие SADT

- 1969 год: началась работа над SADT
- 1973 год: реализация первого крупного приложения при разработке большого аэрокосмического проекта, SofTech, Inc.
- 1974 год: SADT была еще улучшена и передана одной из крупнейших европейских телефонных компаний
- 1975 год: Появление SADT на рынке после годовичного оформления в виде продукта
- С 1981 год SADT уже используется практически во всех сферах экономической деятельности
- В настоящее время – SADT в части нотации IDEF0 является стандартом

Диаграмма - основной рабочий элемент моделирования

Диаграмма является основным рабочим элементом моделирования.

Разработчик диаграмм и моделей обычно называется аналитиком, или, в терминологии SADT, **автором**.

Графика SADT позволяет отражать наборы функций и взаимодействия между ними.

В состав диаграммы входят:

- **блоки**, изображающие функции моделируемой системы,
- **дуги**, связывающие блоки вместе и изображающие взаимодействия и взаимосвязи между блоками
- на дугах отражаются потоки объектов (данных)
- объекты могут быть материальными (детали, изделия, услуги и др.), информационными (накладные, договоры, заказы и др.), финансовые (перечисления, начисления и др.)

Блоки

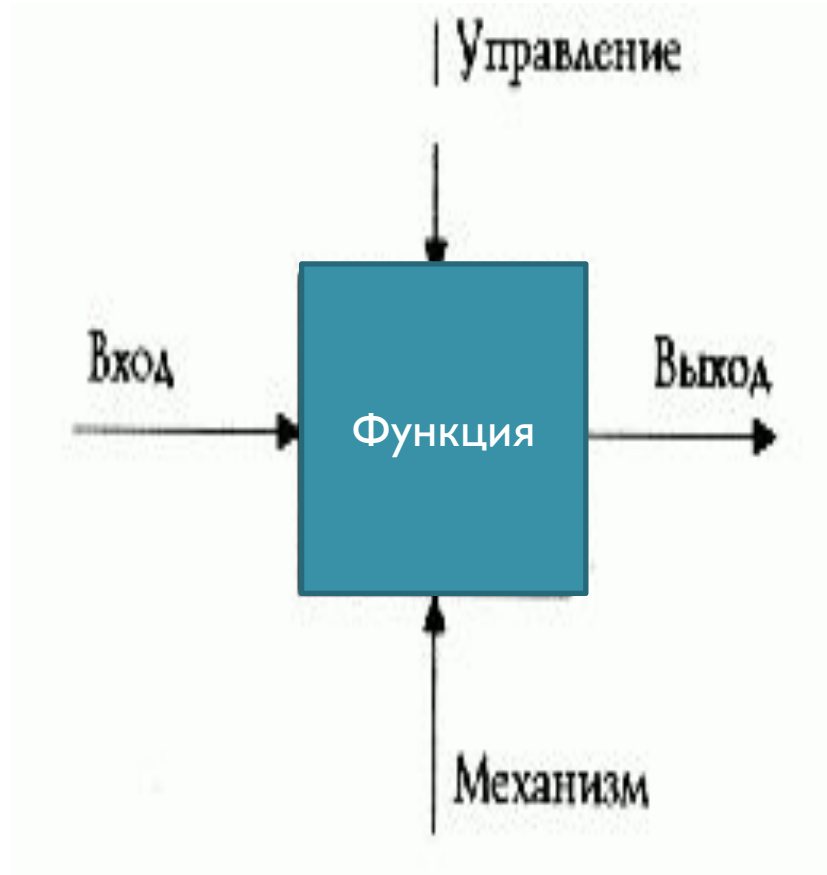
Блоки на диаграммах изображаются прямоугольниками и сопровождаются текстами на естественном языке, описывающим функции.

Блок представляет функцию или активную часть системы, поэтому названиями блоков служат глаголы или отглагольные существительные. Например, названиями блоков диаграммы выполнить задание являются: определить степень выполнения задания, выбрать инструменты, подготовить рабочее место, обработать на станке и собрать,

При этом каждая сторона блока имеет вполне определенное назначение:

- левая сторона предназначена для Входов (Input – I),
- верхняя сторона – для Управления (Control – C),
- правая сторона – для Выходов (Output – O),
- нижняя сторона – для Механизмов (Исполнителей) (Mechanism - M).

Схема моделирования Росса: SA-блок



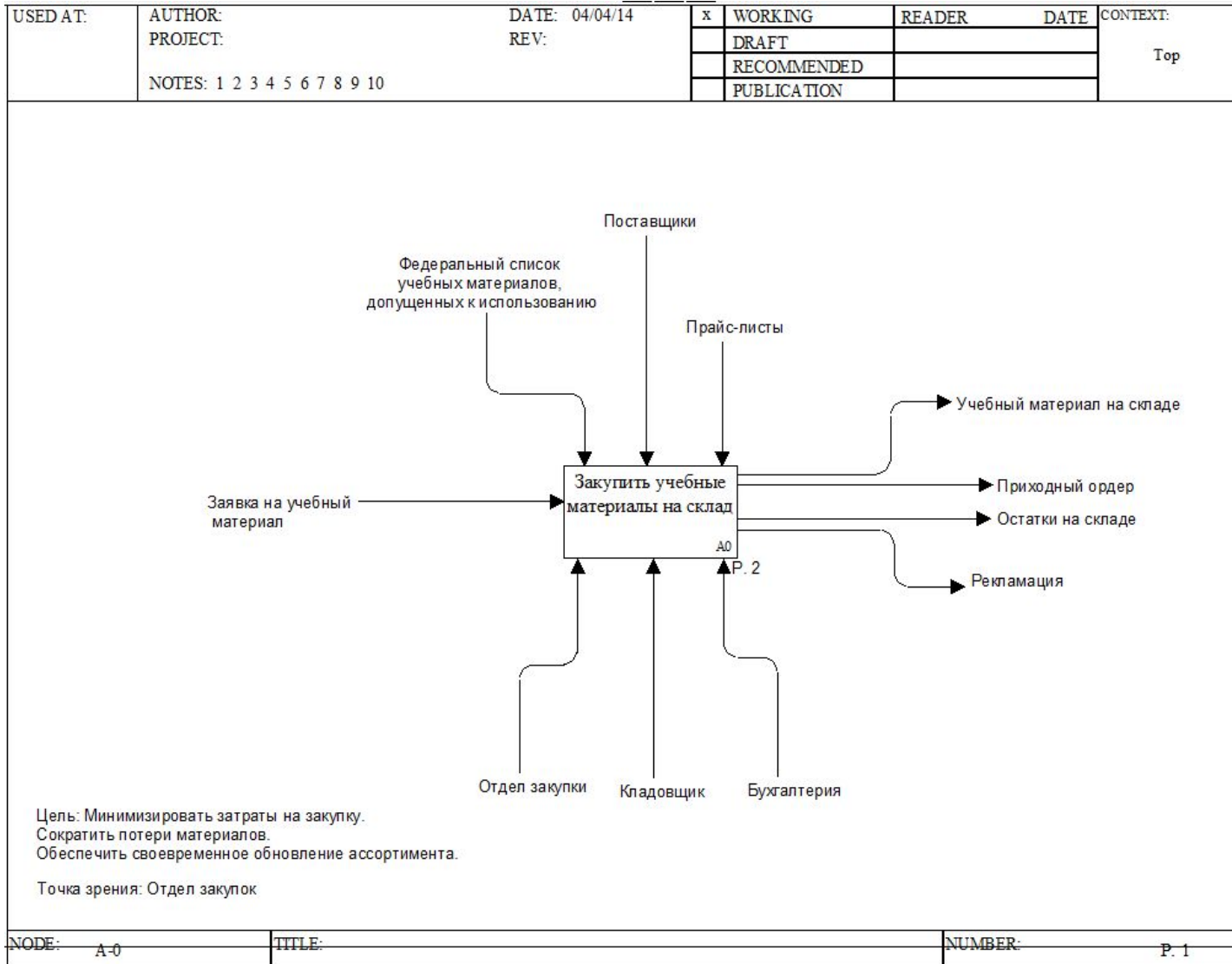
Контекстная диаграмма и ее роль

Один блок и несколько дуг на самом верхнем уровне используются для определения границы всей системы. Этот блок описывает общую функцию, выполняемую системой. Дуги, касающиеся этого блока, описывают главные управления, входы, выходы и механизмы этой системы.

Диаграмма, состоящая из одного блока и его дуг, определяет границу системы и называется контекстной диаграммой модели.

Таким образом, этот блок изображает границу системы: все, лежащее внутри него, является частью описываемой системы, а все, лежащее вне него, образует среду системы.

Методология SADT (IDEF0)



Цель модели

SADT-модель дает полное, точное и адекватное описание системы, имеющее конкретное назначение. Это назначение, называемое целью модели, вытекает из формального определения модели в SADT:

М есть модель системы **S**, если **М** может быть использована для получения ответов на вопросы относительно **S** с точностью **A**.

Таким образом, целью модели является получение ответов на некоторую совокупность вопросов. Эти вопросы неявно присутствуют (подразумеваются) в процессе анализа и, следовательно, они руководят созданием модели и направляют его. Это означает, что сама модель должна будет дать ответы на эти вопросы с заданной степенью точности.

Если модель отвечает не на все вопросы или ее ответы недостаточно точны, то мы говорим, что модель не достигла своей цели.

Вопросы для SADT-модели формулируются на самом раннем этапе проектирования, при этом основная суть этих вопросов должна быть выражена в одной-двух фразах.

Точка зрения

С определением модели тесно связана позиция, с которой наблюдается система и создается ее модель.

Поскольку качество описания системы резко снижается, если оно не сфокусировано ни на чем, SADT требует, чтобы модель рассматривалась все время с одной и той же позиции.

Эта позиция называется "точкой зрения" данной модели.

"Точку зрения" лучше всего представлять себе как место (позицию) человека или объекта, в которое надо встать, чтобы увидеть систему в действии.

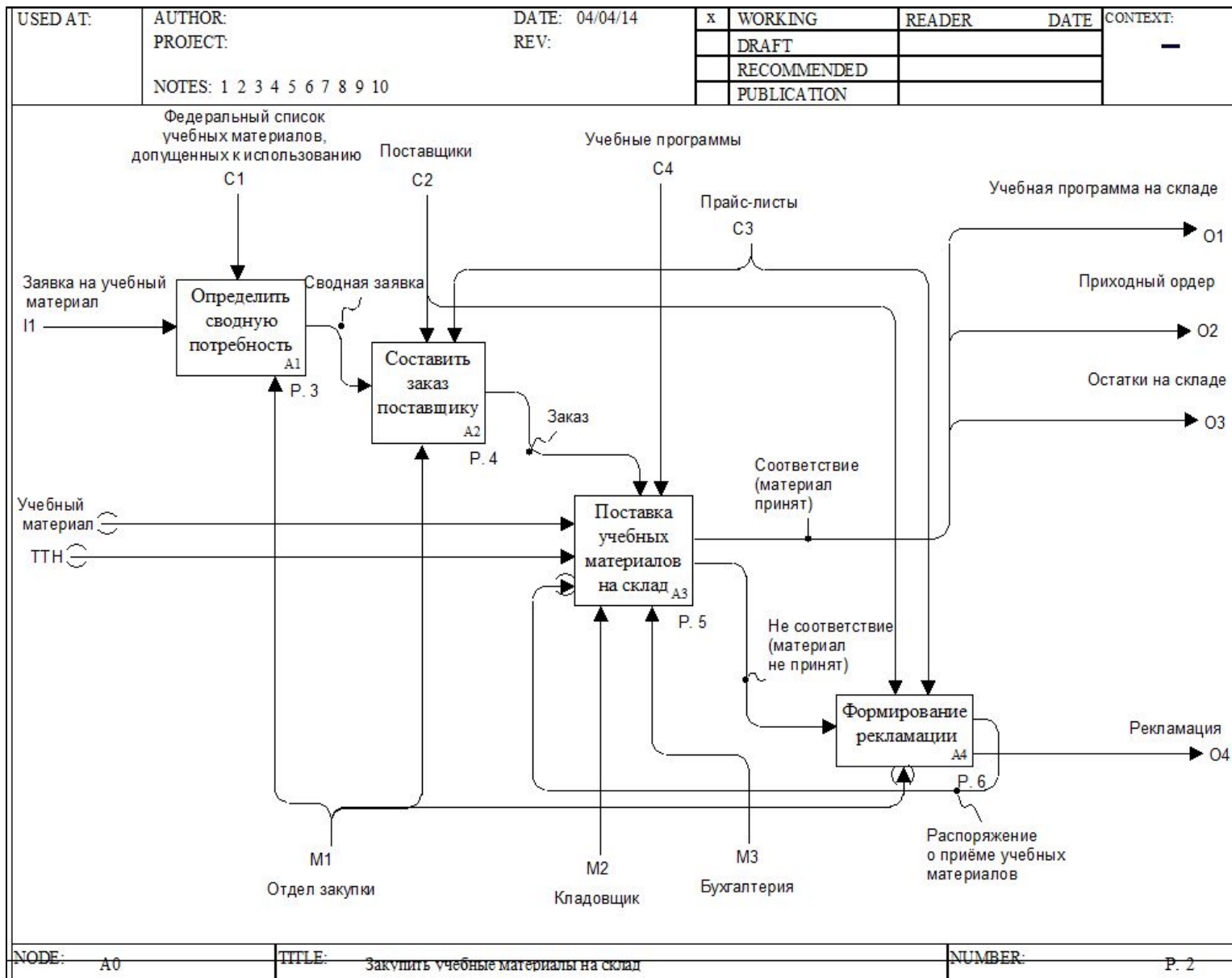
Субъект, управляющий процессом, часто называют владельцем процесса.

Правила синтаксиса моделей.

Декомпозиция диаграмм

- SADT-модель является иерархически организованной совокупностью диаграмм. Каждый функциональный блок на одной диаграмме может быть представлен подчиненной диаграммой, раскрывающей содержание этого блока.
- Диаграммы обычно состоят из трех-шести блоков, каждый из которых потенциально может быть детализирован на другой диаграмме.
- Каждый блок может пониматься как отдельный тщательно определенная подсистема. Разделение такого блока на его структурные части (блоки и дуги, составляющие диаграмму) называется декомпозицией и представляется отдельной диаграммой.
- Декомпозиция формирует границы, и каждый блок в SADT рассматривается как формальная граница некоторой части целой системы, которая описывается. Блок и касающиеся его дуги определяют точную границу диаграммы, представляющей декомпозицию этого блока. Эта диаграмма, называемая диаграммой с потомком, описывает все, связанное с этим блоком и его дугами, и не описывает ничего вне этой границы. Декомпозируемый блок называется родительским блоком, а содержащая его диаграмма - соответственно родительской диаграммой. Таким образом SADT-диаграмма является декомпозицией некоторого ограниченной системы.
- Принцип ограничения объекта встречается на каждом уровне.

Методология SADT (IDEF0)



Правила представления блоков

SADT требует, чтобы в диаграмме было не менее трех и не более шести блоков. **Эти ограничения поддерживают сложность диаграмм и модели на уровне, доступном для чтения, понимания и использования. Другими словами, SADT-диаграммы наглядны**

Блоки на диаграмме размещаются на «ступенчатой» схеме в соответствии с их доминированием, которое понимается как влияние, оказываемое одним блоком на другие.

Блоки должны быть пронумерованы в соответствии с их доминированием. Номера блоков служат однозначными идентификаторами для функций и автоматически организуют эти функции в иерархическую модель

Дуги

Дуги на SADT-диаграмме изображаются одинарными линиями со стрелками на концах.

Дуги в SADT представляют наборы предметов (данных) или объектов. При этом понятие объекта трактуется в широком смысле слова. Так как в SADT дуги изображают объекты, они описываются (помечаются) существительными или существительными с определениями.

Дуги обеспечивают взаимосвязь между блоками и могут состоять с функциями в четырех возможных отношениях: Вход, Выход, Управление, Механизм.

Дуги как инструмент взаимосвязи между блоками

- Входы преобразуются в Выходы,
- Управления ограничивают или предписывают условия выполнения (стандарты, инструкции, планы, нормативы, приказы). В некоторых случаях управляющие объекты могут рассматриваться и как входные, например, планы, нормативы могут участвовать в непосредственной обработке данных.
- Механизмы описывают средства или ресурсы, которые выполняют преобразования (люди, оборудование, технологии, программное обеспечение).

Взаимосвязь блоков

Взаимовлияние блоков может выражаться либо в пересылке Выхода к другой функции для дальнейшего преобразования, либо в выработке Управляющей информации, предписывающей, что именно должна делать другая функция.

Существуют пять типов взаимосвязей между блоками для описания их отношений:

- Управление,
- Вход,
- Обратная Связь по Управлению,
- Обратная Связь по Входу,
- Выход-Механизм.

Отношение Управления

Отношение Управления возникает тогда, когда Выход одного блока непосредственно влияет на блок с меньшим доминированием.

Отношение Входа

Отношение Входа
возникает тогда, когда
Выход одного блока
становится Входом для
блока с меньшим
доминированием.

Обратная связь по управлению

Обратная Связь по
Управлению возникает
тогда, когда Выход
некоторого блока влияет
на блок с большим
доминированием.

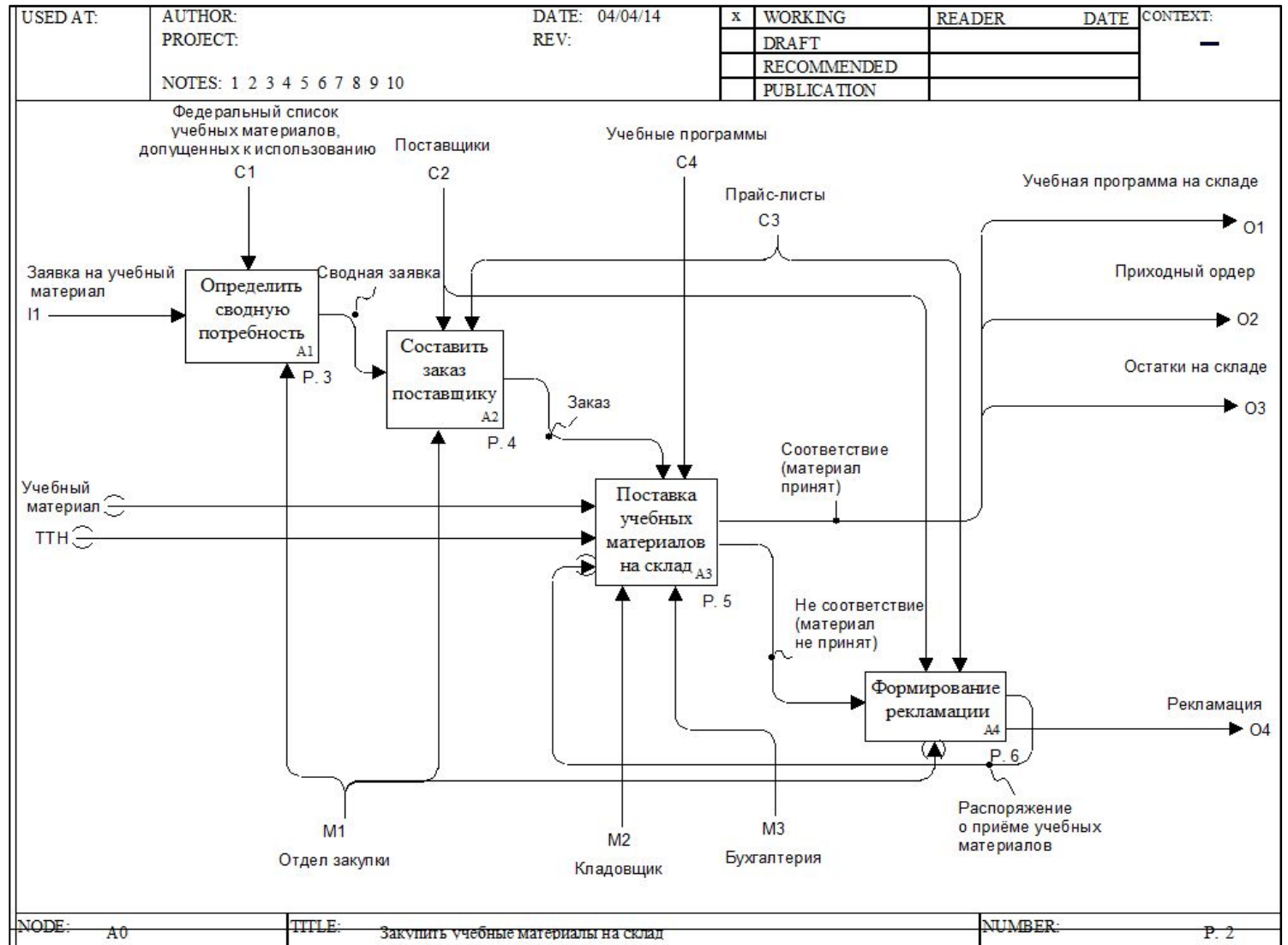
Обратная связь по входу

Обратная Связь по Входу имеет место, когда Выход одного блока становится Входом другого с большим доминированием.

Выход-механизм

Отношение Выход –
Механизм отражают
ситуацию, при которой
Выход одной функции
становится средством
достижения цели
другой функции

Пример связывания блоков



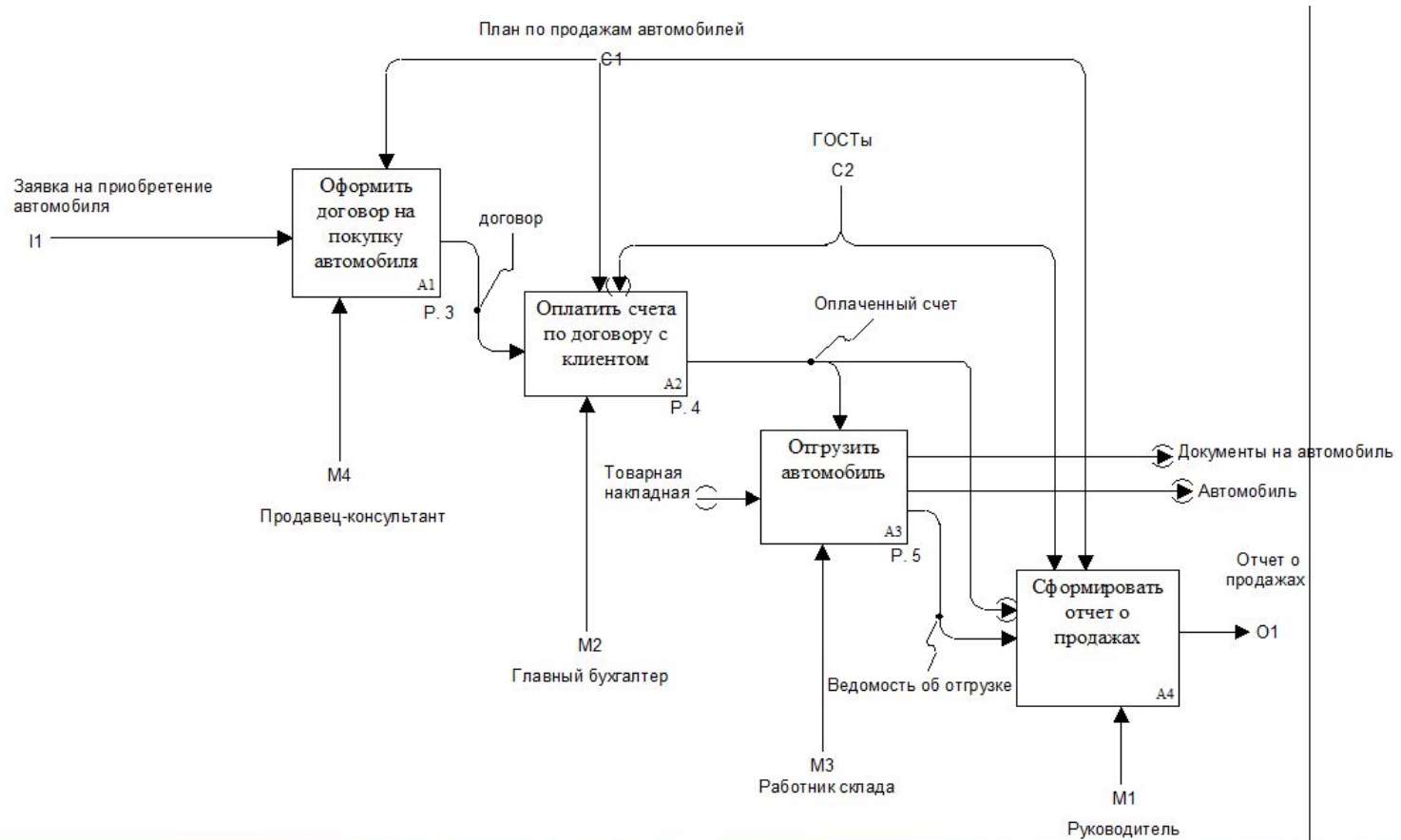
Туннельные дуги

Дуги могут быть туннельными:

- отражающие локальные объекты на диаграмме, которые не заданы на диаграмме вышестоящего уровня;

- отражающие общие объекты верхнего уровня, которые используются во всех блоках нижестоящего уровня.

Туннельные дуги



Дуга как сложный объект

Дуга в SADT редко изображает один объект. Обычно она символизирует набор объектов. Например, дуга, именуемая *рабочий комплект*, отражает *техническое задание*, *чертеж*, *план-график*, *некоторое сырье и заготовки*.

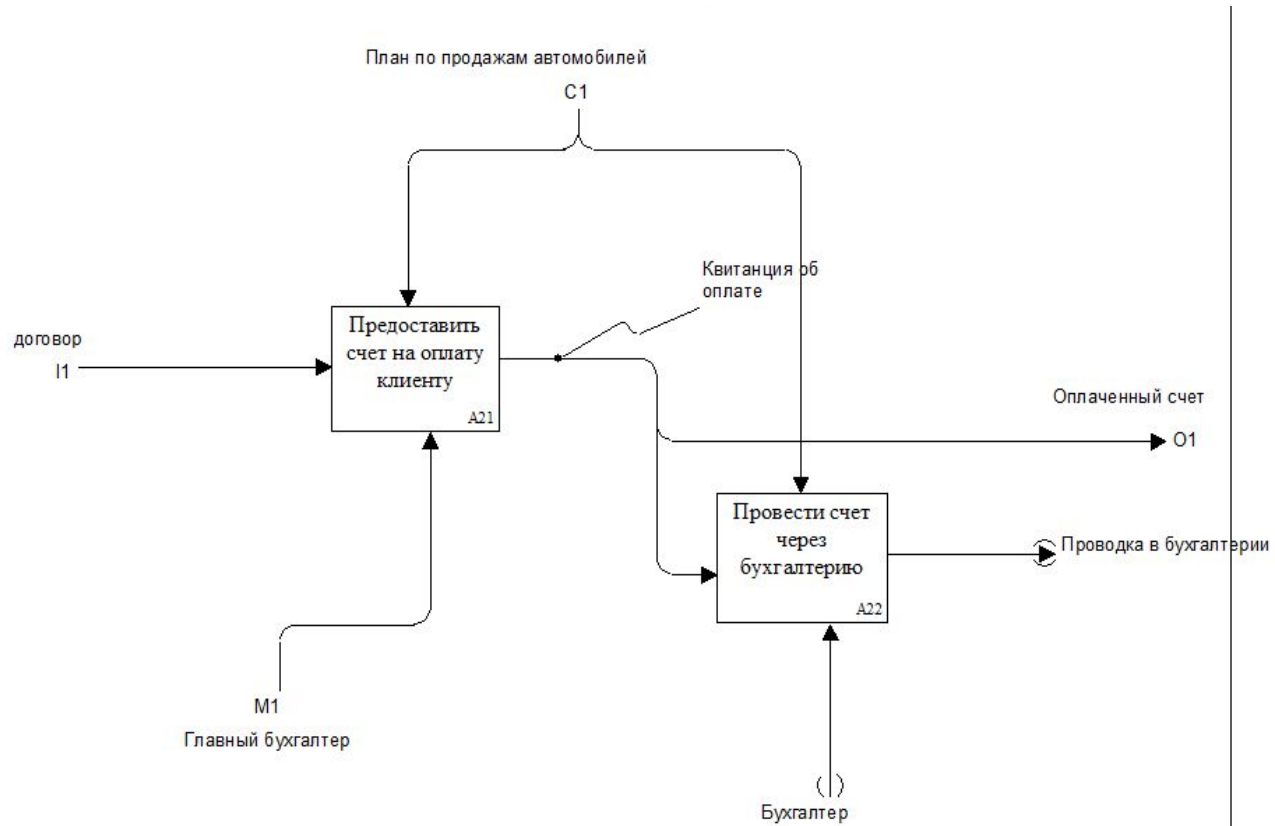
Так как дуги представляют наборы объектов, они могут иметь множество начальных точек (источников) и конечных точек (назначений). Поэтому дуги могут разветвляться и соединяться различными сложными способами.

Разветвление дуг

Разветвления дуг, изображаемые в виде расходящихся линий, означают, что все содержимое дуг или его часть может появиться в каждом ответвлении дуги. Дуга всегда помечается до разветвления, чтобы дать название всему набору. Кроме того, каждая ветвь дуги может быть помечена или не помечена в соответствии со следующими правилами:

- непомеченные ветви содержат все объекты, указанные в метке дуги перед разветвлением (т.е. все объекты принадлежат этим ветвям);
- ветви, помеченные после точки разветвления, содержат все объекты или их часть, указанные в метке дуги перед разветвлением (т.е. каждая метка ветви уточняет, что именно содержит ветвь).

Пример разветвления дуг

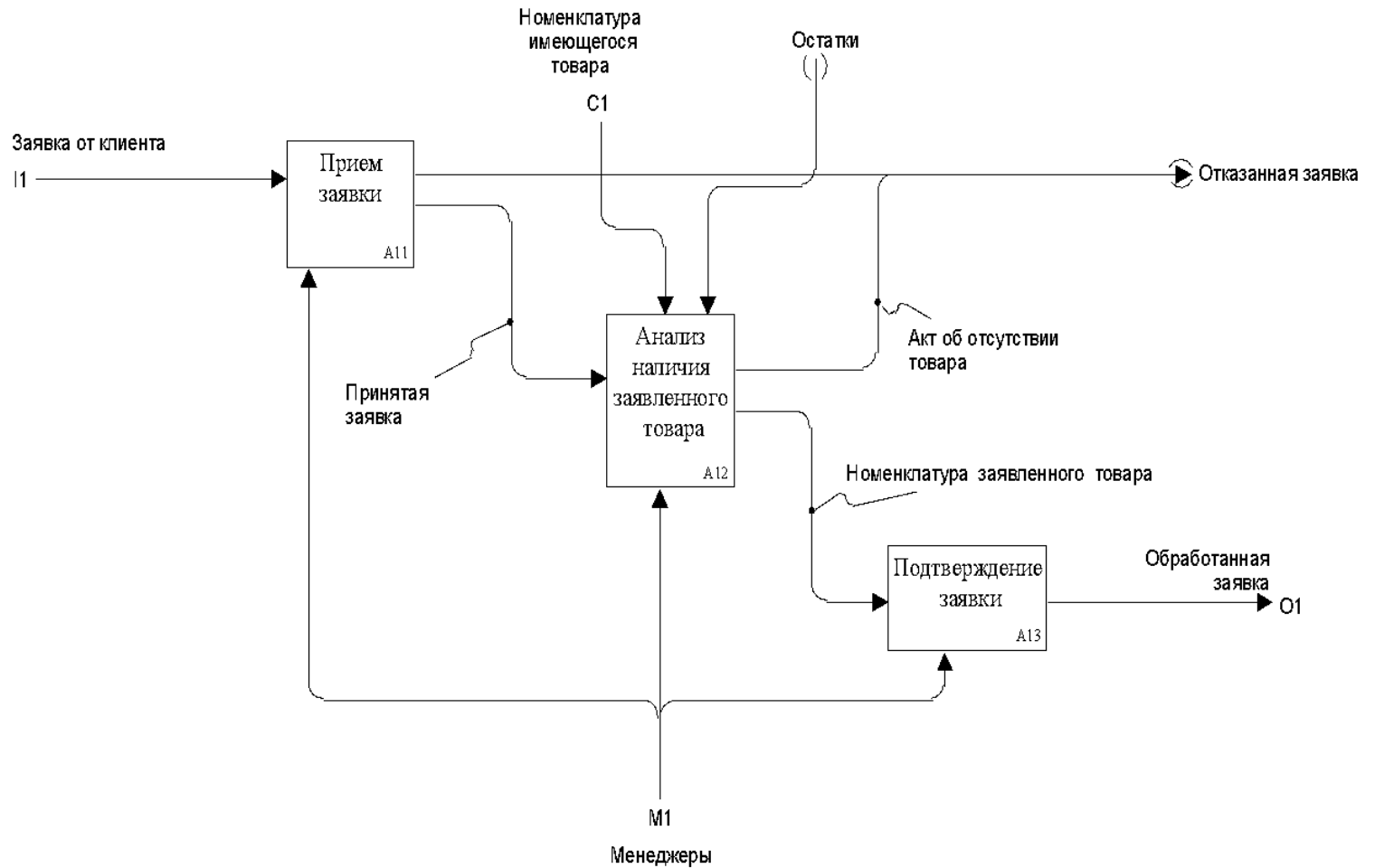


Слияние дуг

Слияние дуг в SADT, изображаемое как сходящиеся вместе линии, указывает, что содержимое каждой ветви идет на формирование метки для дуги, являющейся результатом слияния исходных дуг. После слияния результирующая дуга всегда помечается для указания нового набора объектов, возникшего после объединения. Кроме того, каждая ветвь перед слиянием может помечаться или не помечаться в соответствии со следующими правилами:

- непомеченные ветви содержат все объекты, указанные в общей метке дуги после слияния (т.е. все объекты исходят из всех ветвей);
- помеченные перед слиянием ветви содержат все или некоторые объекты из перечисленных в общей метке после слияния (т.е. метка ветви ясно указывает, что содержит ветвь).

Пример слияния дуг



ER-модель предметной области



Порядок построения модели

- **Процедурно-ориентированный подход** регламентирует первичность проектирования функциональных компонент по отношению к проектированию структур данных: требования к данным раскрываются через функциональные требования. *Сначала строится функциональная модель, на дугах выявляются объекты, которые затем должны быть перенесены в ER-модель*
- **При подходе, ориентированном на данные**, вход и выход являются наиболее важными - структуры данных определяются первыми, а процедурные компоненты являются производными от данных.

Сначала строится ER-модель, в которой определяется какие процессы существуют в предметной области. Затем строятся функциональные модели для каждого выявленного процесса, в которые переносятся объекты из ER-модели.

- **Параллельное проектирование** процессов и структур данных с согласованием функциональных моделей и ER-моделей

2. Методология структурного проектирования Гейна –Сарсона. Диаграммы потоков данных (DFD)

являются основным средством моделирования функциональных требований проектируемой системы.

С их помощью эти требования разбиваются на функциональные компоненты (процессы) и представляются в виде сети, связанной потоками данных.

Главная цель таких средств - продемонстрировать, как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами.

История создания

- Ларри Константайн (IBM) 1965, 1974 – структурное проектирование
- Hughee Aircraft Company – 1975, 1977 – интерактивная система графики структурных схем
- Гэйн К., Т. Сарсон – основали фирму Improved System Technologies. Первый CASE - инструмент STRADIS, 1976.
- Э. Йодан, Г. Маейрс, У. Стивенс, Т. Де Марко, В. Вайнберг. Компания Jordon inc. - 1975 г.

Оценка ЖЦ с использованием методов структурного анализа и проектирования:

5% - обследование, 35 % - анализ, 20 % - проектирование, 15 % - реализация, 25 % - остальное.

Методология Гейна-Сарсона

В основе данной методологии лежит построение модели ИС.

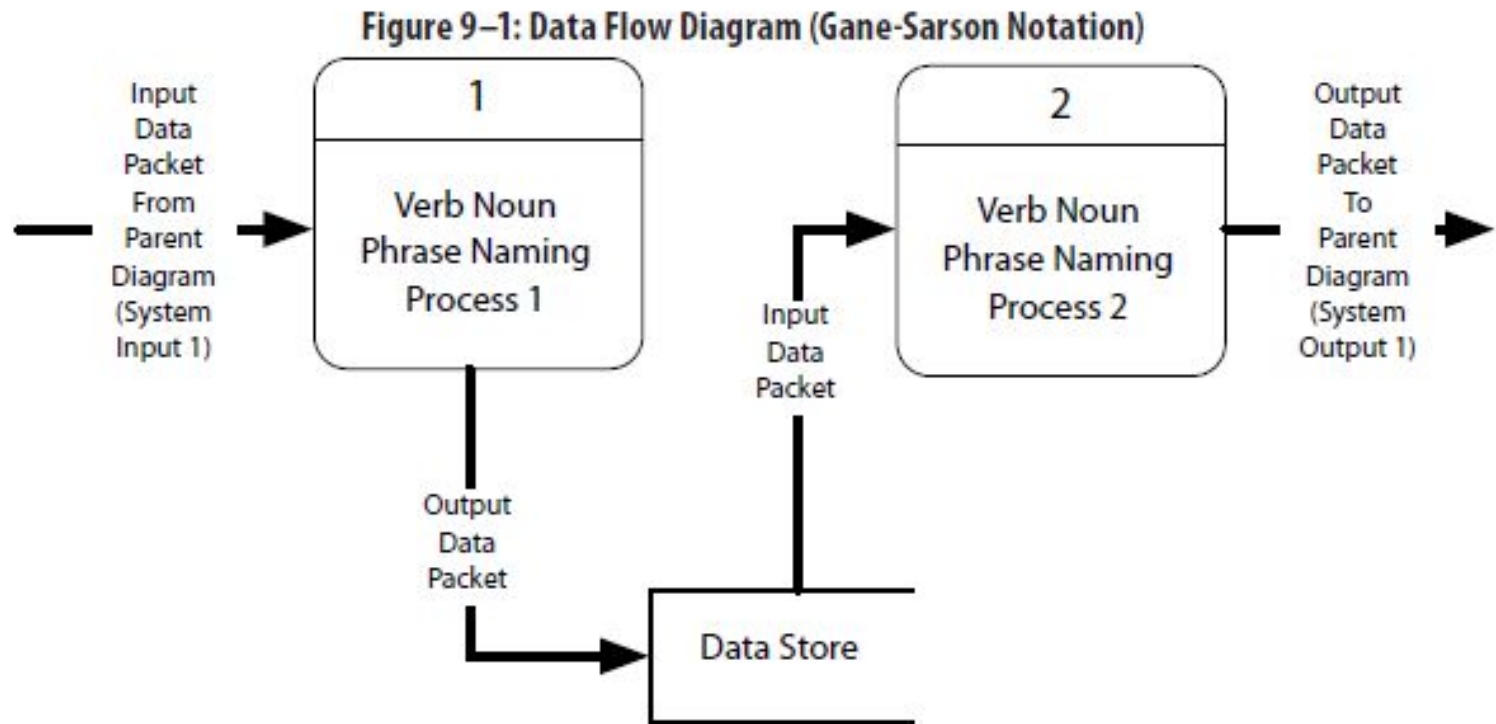
В соответствии с методологией модель системы определяется как иерархия диаграмм потоков данных – DataFlow Diagram (ДПД или DFD), описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи пользователю.

Диаграммы верхних уровней иерархии - контекстные диаграммы - определяют основные процессы или подсистемы ИС с внешними входами и выходами.

Они детализируются при помощи диаграмм нижнего уровня. Такая декомпозиция продолжается, создавая многоуровневую иерархию диаграмм, до тех пор, пока не будет достигнут такой уровень декомпозиции, на котором процесс становится элементарными и детализировать их далее нет необходимости.

Инструменты : Vantage Team Builder (Vestmount), Power Design (SAP)

Процесс преобразования данных



Основные компоненты DFD

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

Таким образом, основными компонентами диаграмм потоков данных являются:

- внешние сущности;
- системы/подсистемы;
- процессы;
- хранилища данных;
- потоки данных.

Внешние сущности

Внешняя сущность представляет собой материальный предмет или физическое лицо, представляющее собой источник или приемник информации, например, заказчики, персонал, поставщики, клиенты, склад. Может быть внешняя АС (подсистема)

Определение некоторого объекта или системы в качестве внешней сущности указывает на то, что она находится за пределами границ анализируемой ИС. В процессе анализа некоторые внешние сущности могут быть перенесены внутрь диаграммы анализируемой ИС, если это необходимо, или, наоборот, часть процессов ИС может быть вынесена за пределы диаграммы и представлена как внешняя сущность.

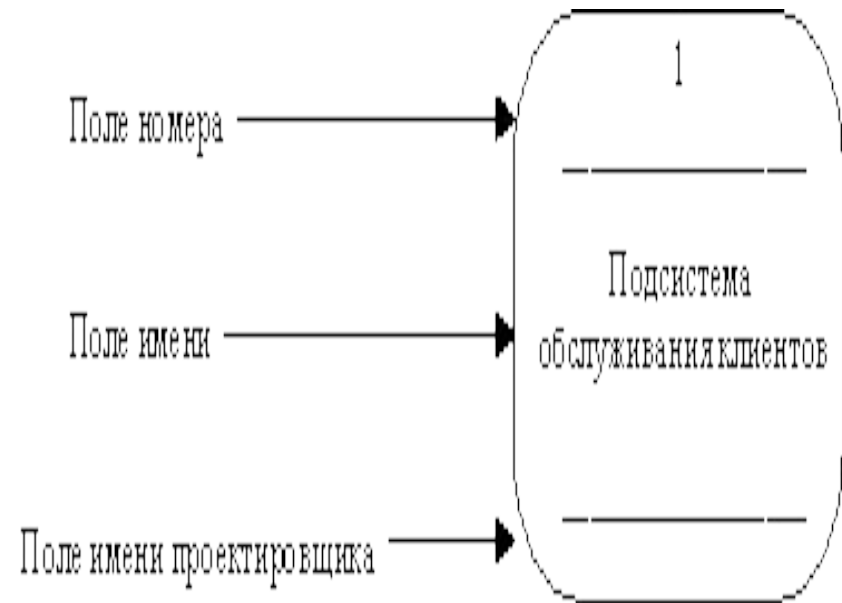
Внешняя сущность обозначается квадратом, расположенным как бы "над" диаграммой и бросающим на нее тень, для того, чтобы можно было выделить этот символ среди других обозначений:



Системы и подсистемы

При построении модели сложной ИС она может быть представлена в самом общем виде на так называемой контекстной диаграмме в виде одной системы как единого целого, либо может быть декомпозирована на ряд подсистем.

Номер подсистемы служит для ее идентификации. В поле имени вводится наименование подсистемы в виде предложения с подлежащим и соответствующими определениями и дополнениями.

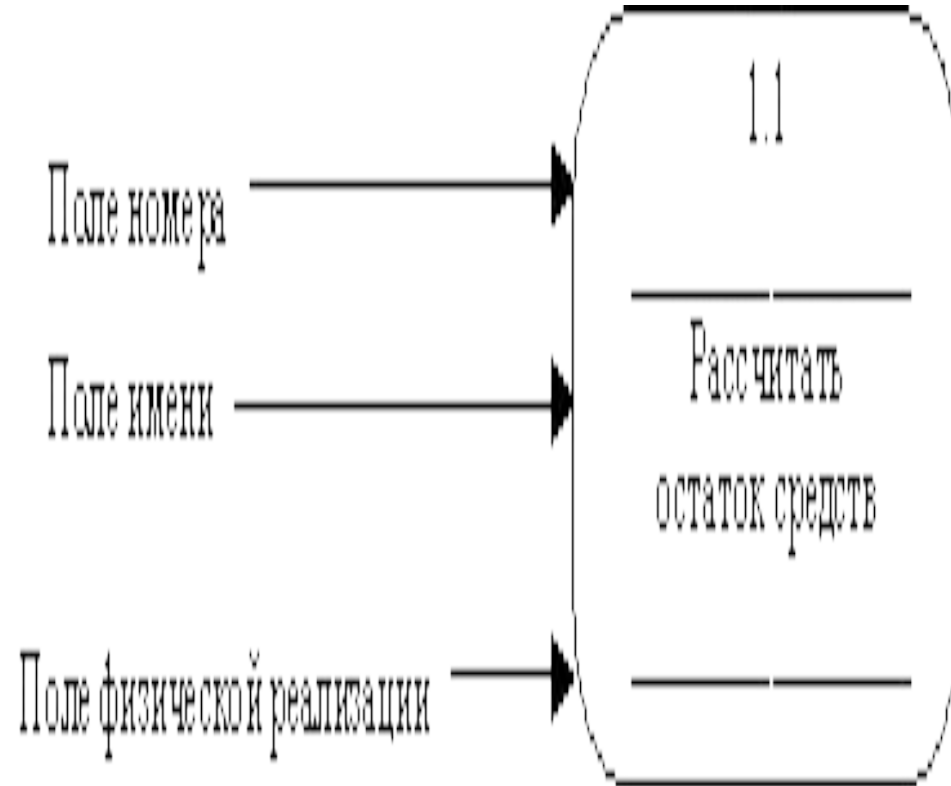


Процессы

Процесс представляет собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом. Физически процесс может быть реализован различными способами: это может быть подразделение организации (отдел), выполняющее обработку входных документов и выпуск отчетов, программа, аппаратно реализованное логическое устройство и т.д.

Номер процесса служит для его идентификации. В поле имени вводится наименование процесса в виде предложения с активным недвусмысленным глаголом в неопределенной форме (вычислить, рассчитать, проверить, определить, создать, получить), за которым следуют существительные в винительном падеже.

Информация в поле физической реализации показывает, какое подразделение организации, программа или аппаратное устройство выполняет данный процесс.



Хранилище данных

Накопитель данных представляет собой абстрактное устройство для хранения информации, которую можно в любой момент поместить в накопитель и через некоторое время извлечь, причем способы помещения и извлечения могут быть любыми.

Накопитель данных может быть реализован физически в виде микрофиши, ящика в картотеке, таблицы в оперативной памяти, файла на носителе и т.д.

Имя накопителя выбирается из соображения наибольшей информативности для проектировщика.

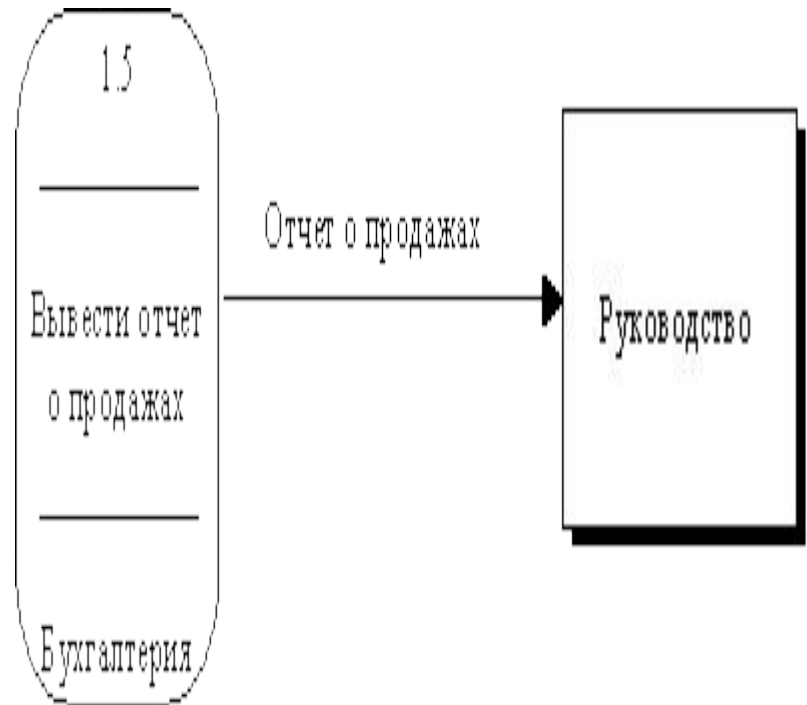
Накопитель данных в общем случае является прообразом будущей базы данных и описание хранящихся в нем данных должно быть увязано с информационной моделью.

D1	Получаемые счета
----	------------------

Поток данных

Поток данных определяет информацию, передаваемую через некоторое соединение от источника к приемнику. Реальный поток данных может быть информацией, передаваемой по кабелю между двумя устройствами, пересылаемыми по почте письмами, магнитными носителями, и т.д.

Каждый поток данных имеет имя, отражающее его содержание.



Построение контекстных диаграмм

Построение контекстных диаграмм является первым шагом при построении иерархии DFD. Обычно при проектировании относительно простых ИС строится единственная контекстная диаграмма со звездообразной топологией, в центре которой находится так называемый главный процесс, соединенный с приемниками и источниками информации, посредством которых с системой взаимодействуют пользователи и другие внешние системы.

Если же для сложной системы ограничиться единственной контекстной диаграммой, то она будет содержать слишком большое количество источников и приемников информации, которые трудно расположить на листе бумаги нормального формата, и кроме того, единственный главный процесс не раскрывает структуры распределенной системы. Признаками сложности (в смысле контекста) могут быть:

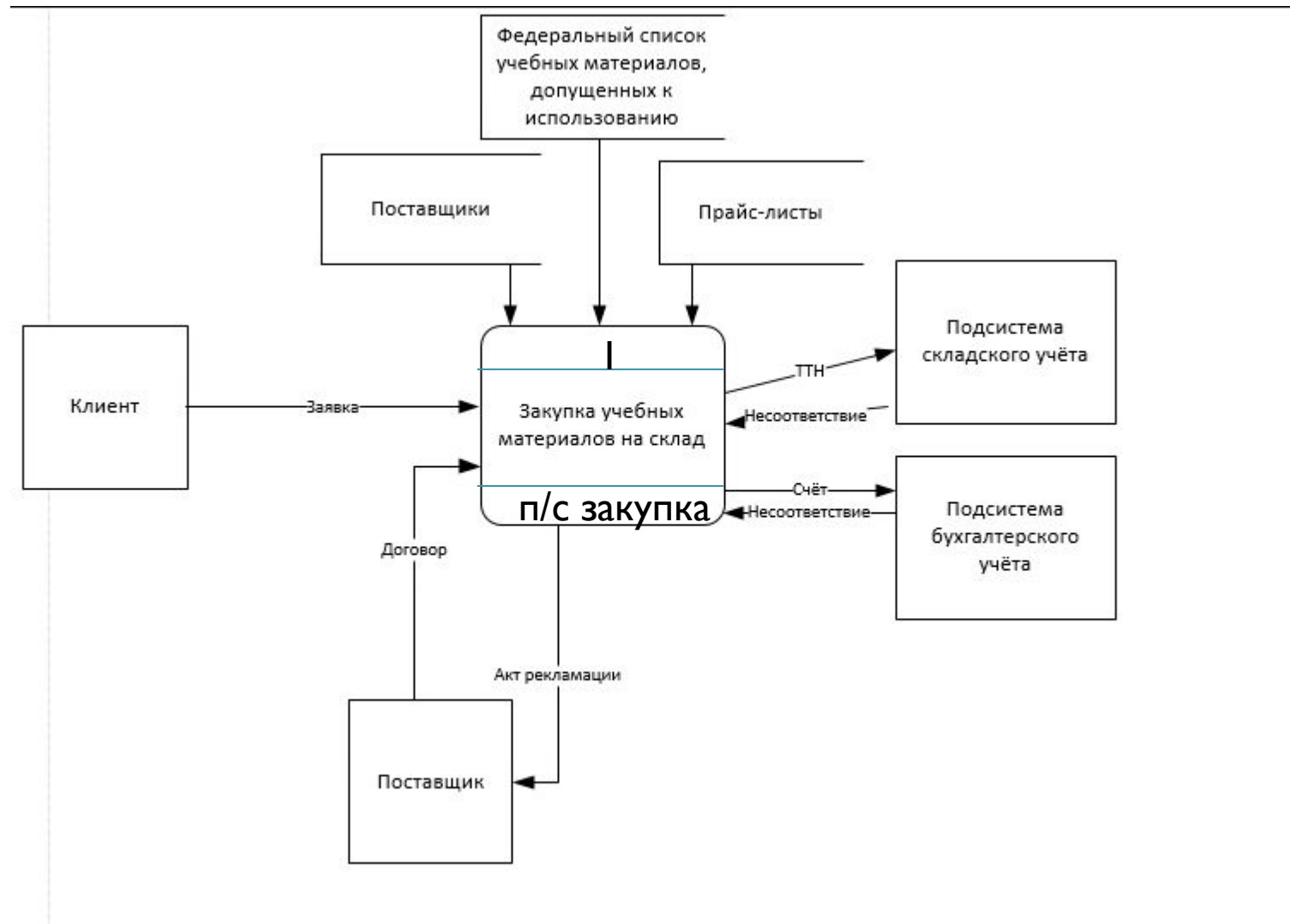
- наличие большого количества внешних сущностей (десять и более);

- распределенная природа системы;

- многофункциональность системы с уже сложившейся или выявленной группировкой функций в отдельные подсистемы.

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

Контекстная диаграмма



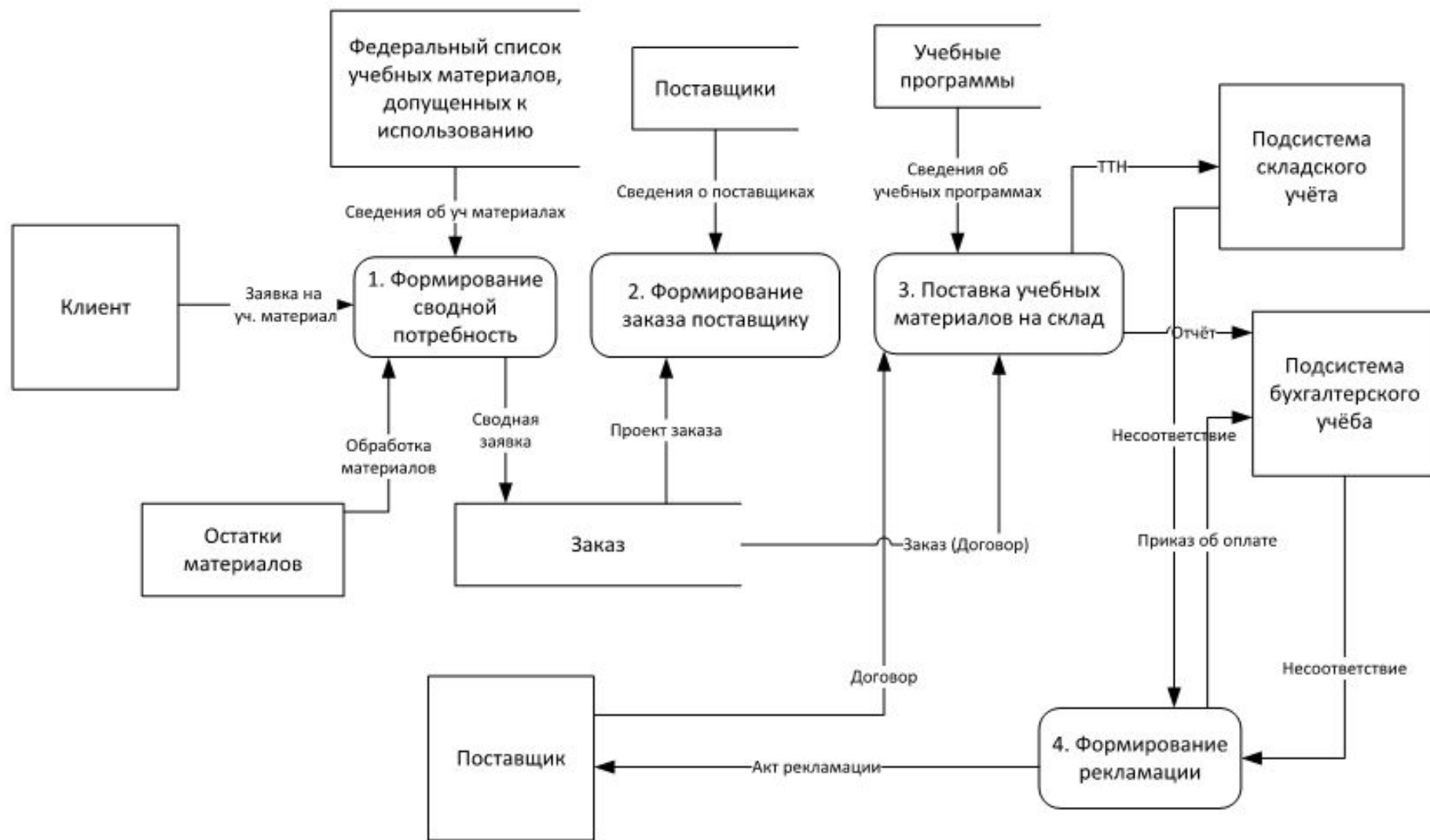
Декомпозиция контекстной диаграммы

Для каждой подсистемы, присутствующей на контекстных диаграммах, выполняется ее детализация при помощи DFD. Каждый процесс на DFD, в свою очередь, может быть детализирован при помощи DFD или миниспецификации. При детализации должны выполняться следующие правила:

- правило балансировки - означает, что при детализации подсистемы или процесса детализирующая диаграмма в качестве внешних источников/приемников данных может иметь только те компоненты (подсистемы, процессы, внешние сущности, накопители данных), с которыми имеет информационную связь детализируемая подсистема или процесс на родительской диаграмме;
- правило нумерации - означает, что при детализации процессов должна поддерживаться их иерархическая нумерация. Например, процессы, детализирующие процесс с номером I2, получают номера I2.1, I2.2, I2.3 и т.д.

Миниспецификация (описание логики процесса) должна формулировать его основные функции таким образом, чтобы в дальнейшем специалист, выполняющий реализацию проекта, смог выполнить их или разработать соответствующую программу.

Пример диаграммы DFD



Миниспецификация

Миниспецификация является завершением иерархии FD. Решение о завершении детализации процесса и использовании миниспецификации принимается аналитиком исходя из следующих критериев:

- наличия у процесса относительно небольшого количества входных и выходных потоков данных (2-3 потока);
- возможности описания преобразования данных процессом в виде последовательного алгоритма;
- выполнения процессом единственной логической функции преобразования входной информации в выходную;
- возможности описания логики процесса при помощи миниспецификации небольшого объема (не более 20-30 строк).

Пример мини-спецификации





Построение событийных цепочек процессов (eEPC – event process chain) – Методология ARIS

