Изучаемый курс

«ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ»

Преподаватель дисциплины:
Нефедова Людмила Петровна

Специальность:
09.02.07 информационные системы и программирование

Учебный материал

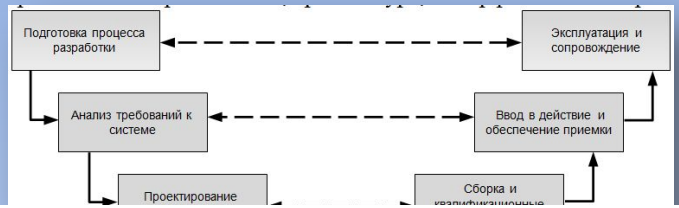
- «Осуществление интеграции программных модулей» Г.Н. Федорова учебник для студентов учреждений СПО - М.: «Академия», 2019.-288с. **Стр 36-60**.

1. «Технология разработки программных продуктов» А.В. Рудаков
2. «Технологии разработки программного обеспечения» С.А. Орлова

- Ресурс НПК на портале (платформа **MOODLE). «**Занятие 2.doc»
- Прочие Интернет-ресурсы

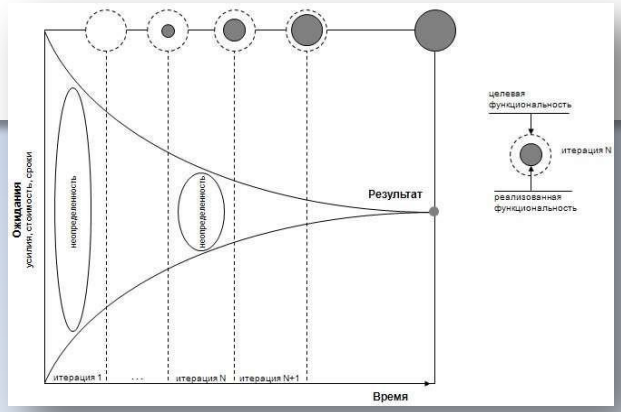
Занятие 7.

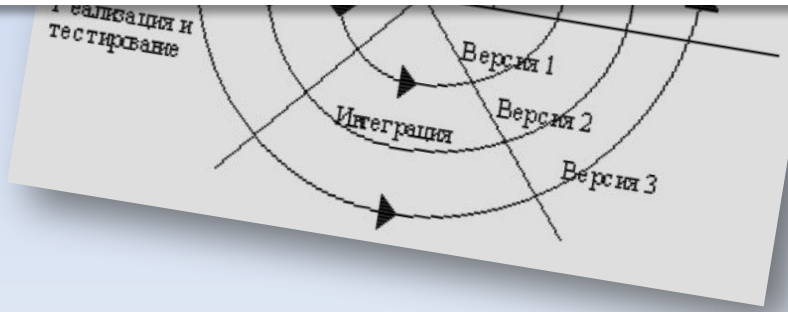
Методики проектирования ПО.



Разработка программного обеспечения начинается с **анализа требований** к нему. В результате анализа получают **спецификации** разрабатываемого программного обеспечения.

Далее строят **общую модель** его взаимодействия с пользователем или другими программами и конкретизируют его основные функции





Модели, используемые в качестве спецификаций, можно поделить на модели, используемые при **структурном** подходе и **объектно-ориентированном**. Каждой группе соответствуют определенные виды мод

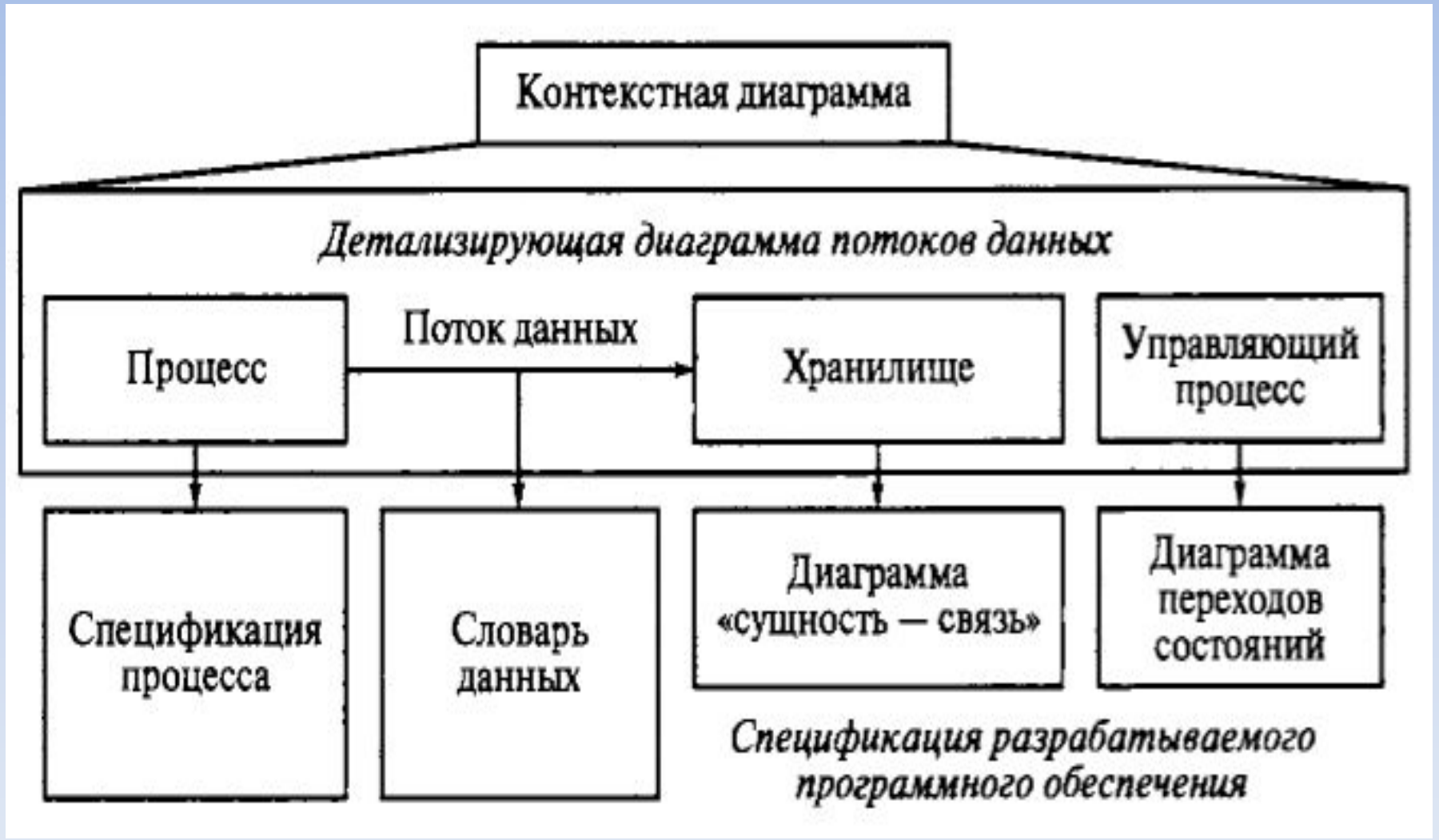
Анализ проектруемого ПО

При **структурном подходе** к программированию на этапе анализа и определения спецификаций разрабатывают три типа моделей: **модели функций, модели данных и модели потоков данных.**

Разные модели описывают проектируемое ПО с разных сторон, рекомендуется использовать сразу несколько моделей, разрабатываемых в виде диаграмм, и пояснять их текстовыми описаниями, словарями и т. п. Структурный анализ предполагает использование следующих видов моделей:

- **диаграмм потоков данных** (**DFD** - Data Flow Diagrams), описывающих взаимодействие источников и потребителей информации через процессы, которые должны быть реализованы в системе;
- **диаграмм «сущность-связь»** (**ERD** Entity-Relationship Diagrams), описывающих базы данных разрабатываемой системы;
- **диаграмм переходов состояний** (**STD** - State Transition Diagrams), характеризующих поведение системы во времени;
- **функциональных диаграмм** (методика **SADT**);
- **спецификаций процессов;**
- **словаря терминов.**

Комплексное представление программного обеспечения в виде совокупности моделей DFD, SADT и ERD



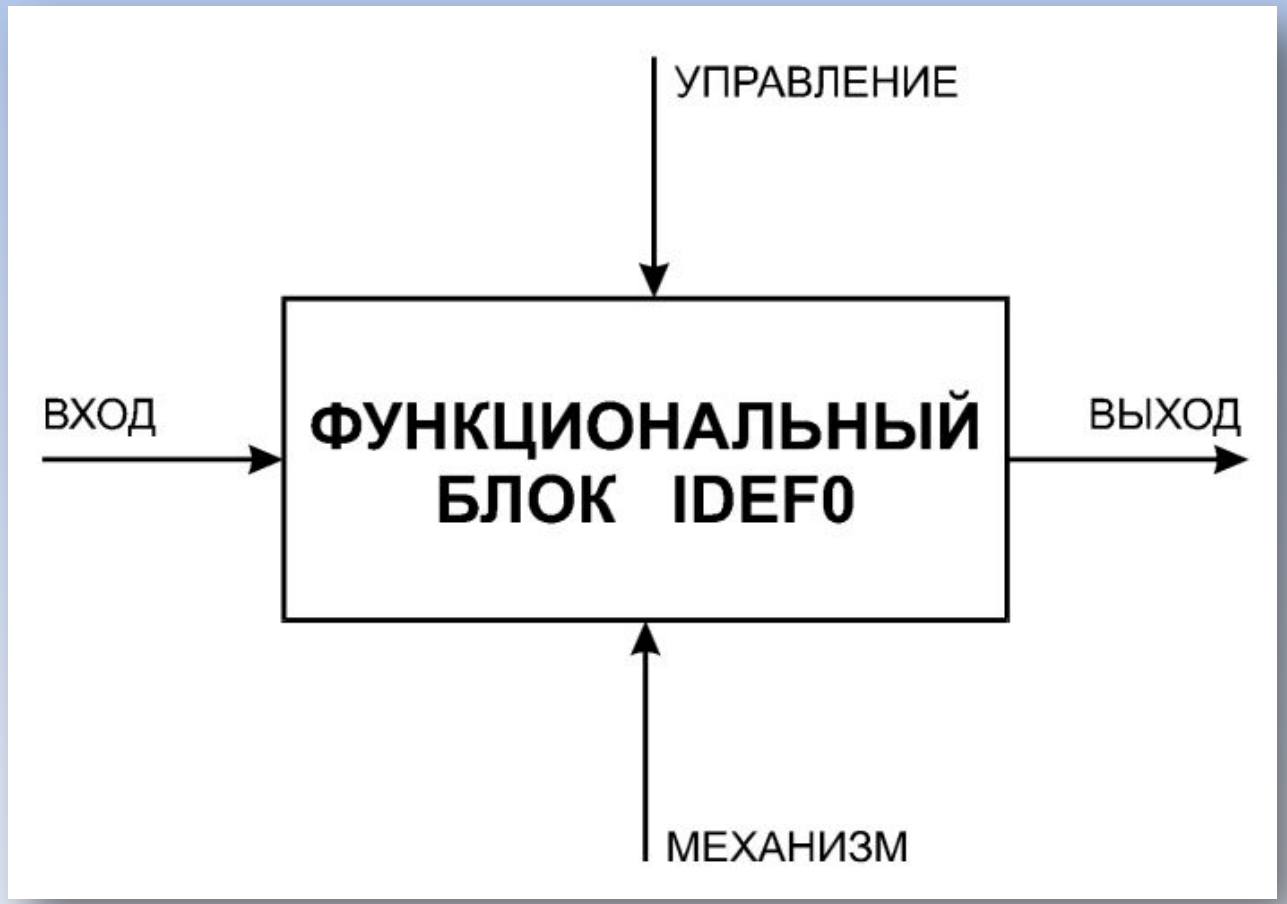
Методология функционального моделирования SADT

- Функциональные диаграммы отражают взаимные связи функций
- Создаются на ранних стадиях проектирования для определения основных функций и составных частей, устранения существенных ошибок.
- Результат = модель из диаграмм, текстов и глоссария

Методология SADT применима для:

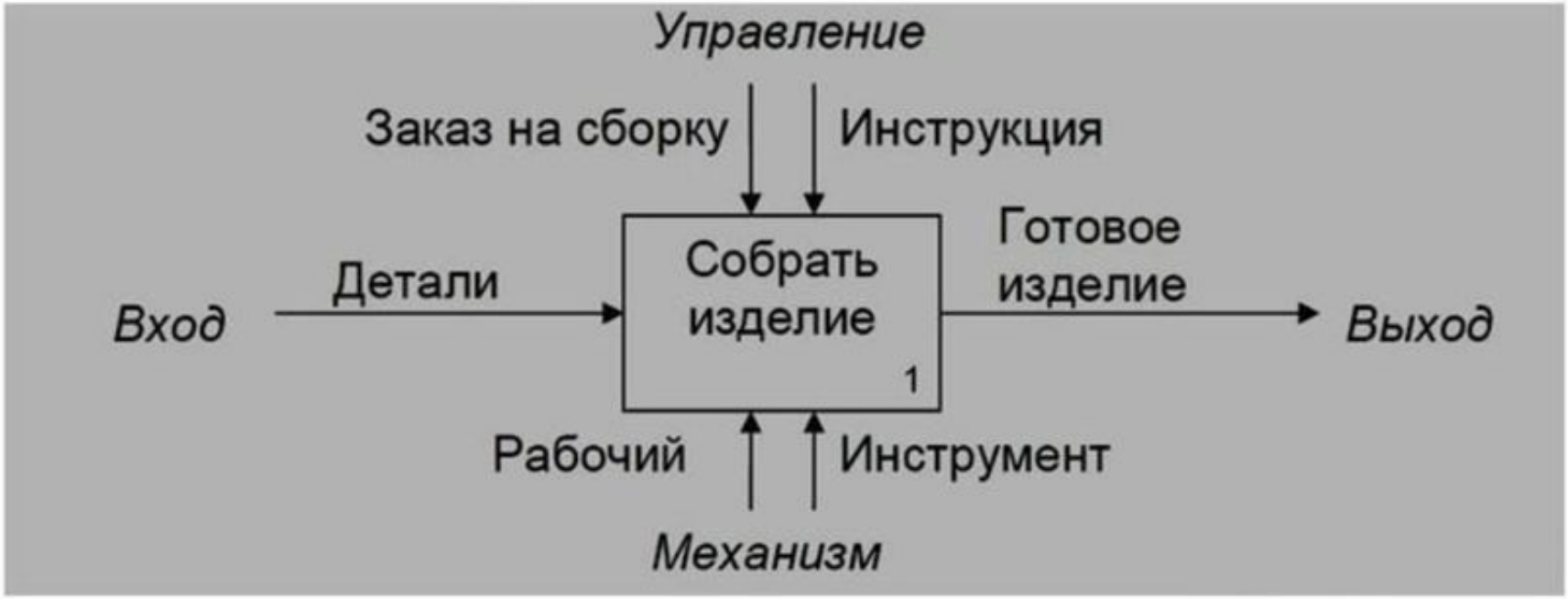
- моделирования предметной области ПО, определения требований и функций
- разработки программного обеспечения согласно требований

Диаграммы – главные компоненты модели. **Функции системы** – блоки, **интерфейсы** – дуги. Место соединения блока и дуги определяет тип интерфейса. **Управляющая система** входит в блок сверху, входная информация входит слева, результат - выход справа. **Механизм**(система, человек) **операции**– вход снизу.



Графические примитивы, термины SADT – EDEF0

Работы/ функциональные блоки (activity)	Поименованный процесс, функция или задача; название – глагол. Рисуется как прямоугольник
Вход (input)	Входные данные. Рисуется как стрелка в левую грань
Выход (output)	Выходные данные. Рисуется как стрелка из правой грани
Управление (control)	Правила, процедуры и стандарты, которыми руководствуется работа. Рисуется как стрелка в верхнюю грань
Механизм (mechanism)	Ресурсы, которые выполняют работу. Рисуется как стрелка в нижнюю грань



IDEF0

- методология функционального моделирования и графическая нотация, предназначенная для формализации и описания бизнес-процессов;
- отличительной особенностью IDEF0 является её акцент на соподчинённость объектов;
- в IDEF0 рассматриваются логические отношения между работами, а не их временна́я последовательность.

Пример диаграммы в формате IDEF0

- **Имя блока,** описывающее функцию, должно быть глаголом или глагольным оборотом.

Примеры имен функций: производить детали, наблюдать за выполнением.

- **Стрелки и их сегменты** помечаются существительными или оборотами существительных.

Примеры меток стрелок: менеджер, бюджет.

- Каждая сторона блока имеет своё **определенное значение** с точки зрения связи блок-стрелка.

- **Стрелка входа** - это материал или данные, которые преобразуются или расходуются функцией, чтобы создать то, что появится на ее выходе. Стрелка входа слева в блок. Функция может не иметь ни одной стрелки входа. Часто бывает сложно определить, являются ли данные входом, или управлением. Если данные изменяются или перерабатываются, это вход, если нет - управление.
- **Стрелка управления** - это правила, стратегии, процедуры, стандарты, которые определяют условия, необходимые функции, чтобы произвести правильный выход. Стрелка входи в верхнюю грань блока. Каждая функция должна иметь хотя бы одну стрелку управления. Управление влияет на функцию, но не преобразуется функцией. Если цель функции - изменить процедуру, то такая процедура будет для функции входом. В случае возникновения неопределенности в классифицировании стрелки (вход или управление)

- **Стрелка выхода** - это данные или материальные объекты, произведенные функцией. Стрелка выхода - выходящая из правой грани блока. Каждая функция должна иметь хотя бы одну стрелку выхода. Функция без выхода не имеет смысла и не должна моделироваться.
- **Стрелка механизма** - это ресурсы (персонал, техника, оборудование), поддерживающие выполнение функции. Стрелка механизма рисуется как входящая в нижнюю грань блока. Стрелка механизма может не изображаться на модели.
- **Стрелка вызова** - это стрелка, указывающая на другую модель. Стрелка вызова рисуется как исходящая из нижней грани блока. Такая стрелка используется как указание на то, что некоторая функция выполняется за пределами моделируемой системы.

IDEF0-модели состоят из трех типов документов: **графических диаграмм, текста, глоссария**. Эти документы имеют перекрестные ссылки друг на друга.

- **Графическая диаграмма** - главный компонент модели, содержащий блоки, стрелки, соединения блоков и стрелок. Диаграмма является единицей описания системы и располагается на отдельном листе.
- **Блоки** представляют основные функции моделируемого объекта. Эти функции могут быть декомпозированы на составные части и представлены в виде более подробных диаграмм. Процесс декомпозиции продолжается до тех пор, пока объект не будет описан на уровне детализации, необходимом для достижения целей конкретного проекта.
- **Диаграмма верхнего уровня** обеспечивает наиболее общее или абстрактное описание объекта моделирования. За этой диаграммой следует серия дочерних диаграмм, дающих более детальное

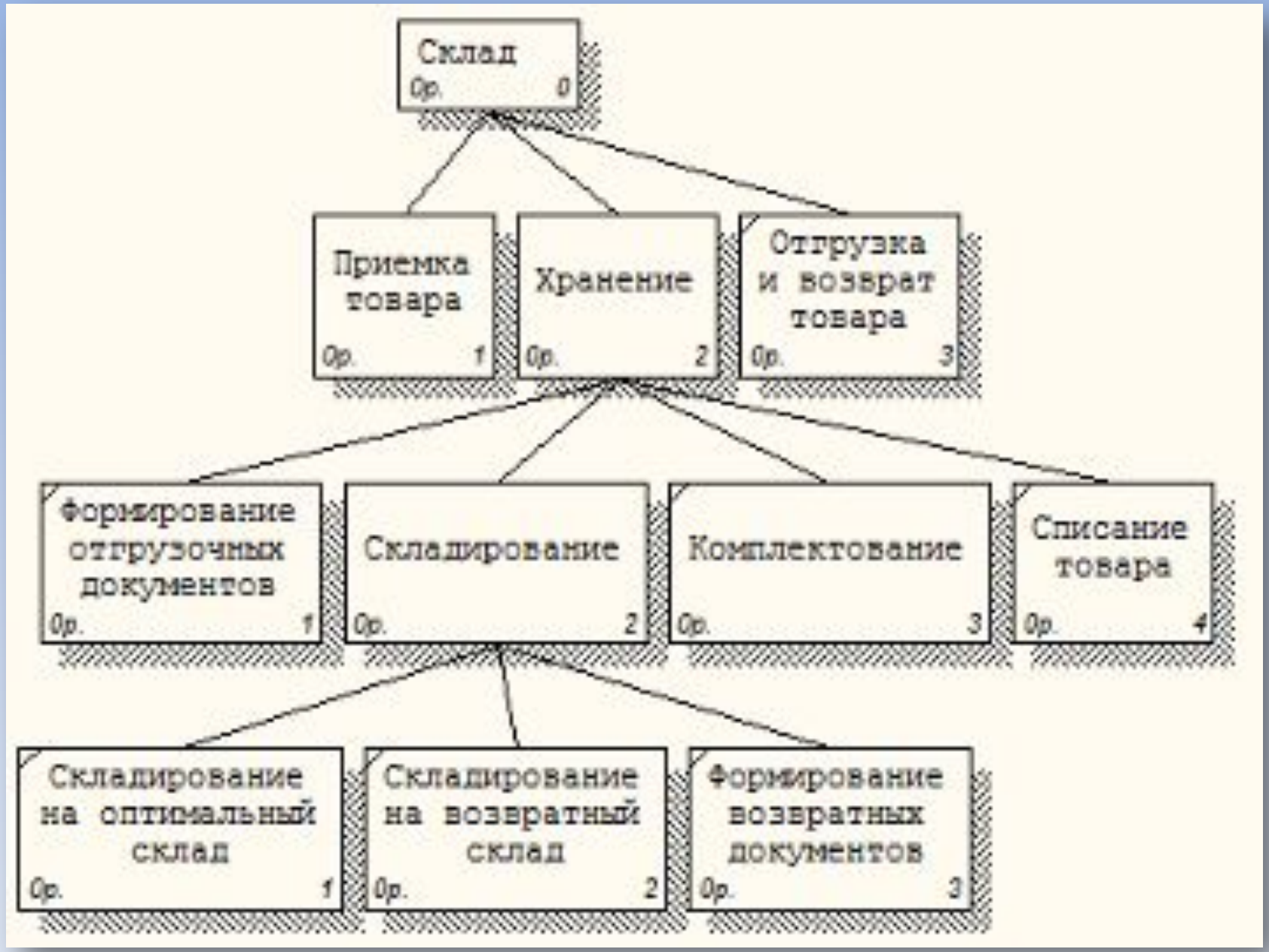
Модель может содержать следующие типы графических диаграмм: **контекстная, декомпозиции, дерева узлов, иллюстрации.**

- Каждая модель должна иметь контекстную диаграмму верхнего уровня, на которой объект моделирования представлен единственным блоком с граничными стрелками. Эта диаграмма называется **A-0**. Стрелки на этой диаграмме отображают связи объекта моделирования с окружающей средой.
- Диаграмма A-0 устанавливает область моделирования и ее границу. Необходимо установить, что входит в систему, а что лежит за ее пределами, то есть определить, что будет рассматриваться как компоненты системы, а что - как внешнее воздействие.
- В пояснительном тексте к контекстной диаграмме должна быть указана **цель построения** диаграммы и зафиксирована точка зрения. Цель - это краткая формулировка причины создания модели. Цель должна отвечать на вопросы: почему этот процесс должен быть смоделирован? что должна показать модель? что может получить читатель?

- **Точка зрения** - это указание на должностное лицо или подразделение организации, с позиции которого разрабатывается модель.
- Точка зрения определяет основное направление развития модели и уровень необходимой детализации. Четкое фиксирование точки зрения позволяет разгрузить модель, отказавшись от детализации и исследования отдельных элементов, не являющихся необходимыми, исходя из выбранной точки зрения на систему.

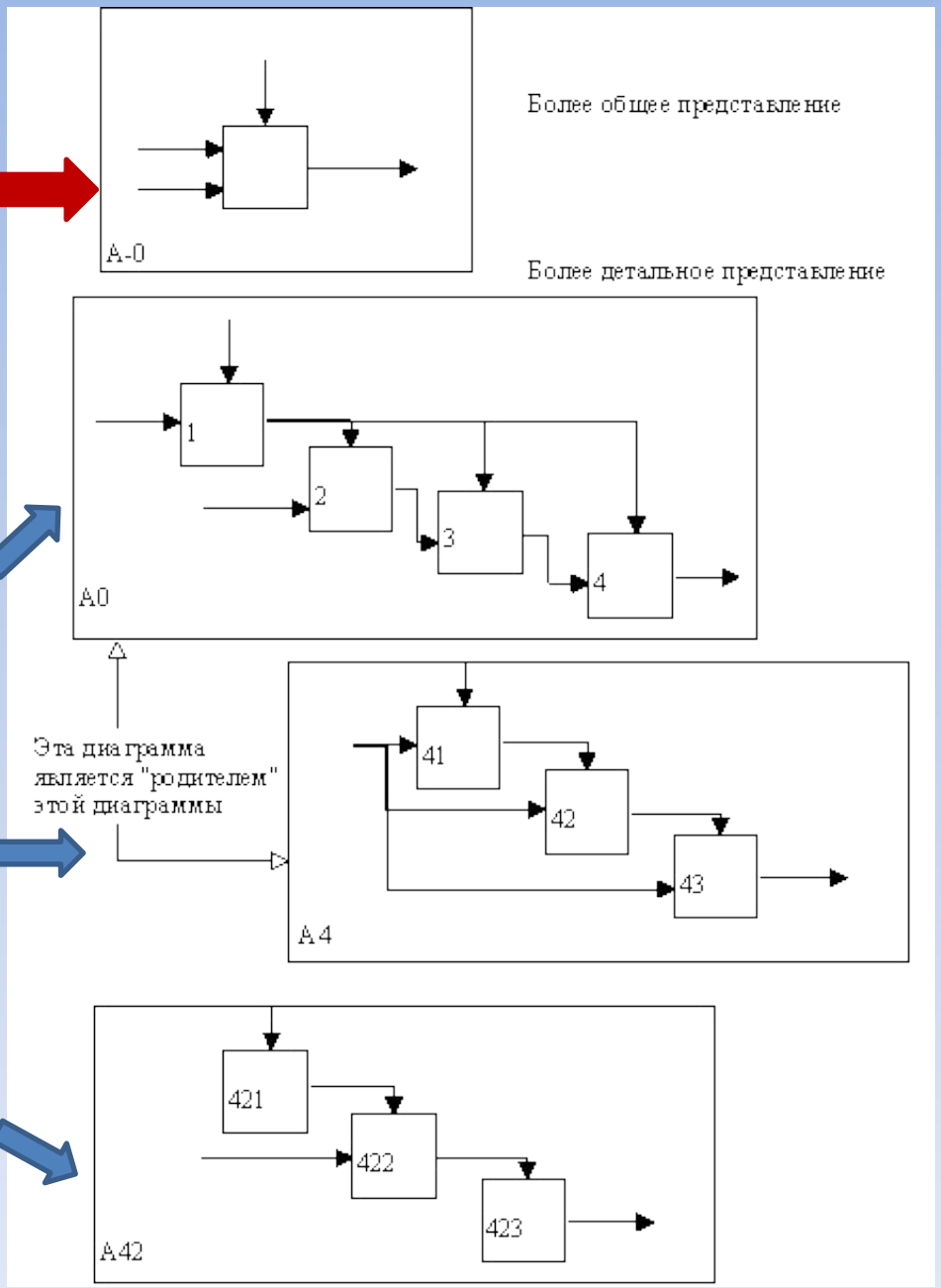
Например, модели одного и того же предприятия с точек зрения главного технолога и финансового директора будут существенно различаться: финансового директора не интересуют аспекты обработки сырья на производственном оборудовании, а главному технологу ни к чему прорисованные схемы финансовых потоков.

- Единственная **функция**, представленная на контекстной диаграмме верхнего уровня, может быть разложена на основные **подфункции** посредством создания дочерней диаграммы. В свою очередь, каждая из этих подфункций может быть разложена на составные части посредством создания дочерней диаграммы следующего, более низкого уровня. Каждая дочерняя диаграмма содержит дочерние блоки и стрелки, обеспечивающие дополнительную детализацию родительского блока.
- Дочерняя диаграмма, создаваемая при **декомпозиции**, охватывает ту же область, что и родительский блок, но описывает ее более подробно. Таким образом, дочерняя диаграмма как бы вложена в свой родительский блок.
- Диаграмме может быть поставлен в соответствие **текст**, представляющий собой краткий комментарий к содержанию диаграммы. Текст не должен использоваться для описания и без того понятных блоков и стрелок на диаграмме.
- **Глоссарий** - это список определений для ключевых слов, фраз, аббревиатур, связанных с блоками, стрелками или с моделью в целом. Глоссарий определяет термины, которые должны одинаково понимать все участники разработки.


Склад
Ор. 0
Приемка товара
Ор. 1
Хранение
Ор. 2
Отгрузка и возврат товара
Ор. 3
Формирование отгрузочных документов
Ор. 1
Складирование
Ор. 2
Комплектование
Ор. 3
Списание товара
Ор. 4
Складирование на оптимальный склад
Ор. 1
Складирование на возвратный склад
Ор. 2
Формирование возвратных документов
Ор. 3

Иерархия диаграмм

1. **Контекстная диаграмма** - диаграмма верхнего уровня – (представляет систему как единое целое)
2. **Диаграммы следующих уровней** (детализируют родительскую диаграмму, показывая



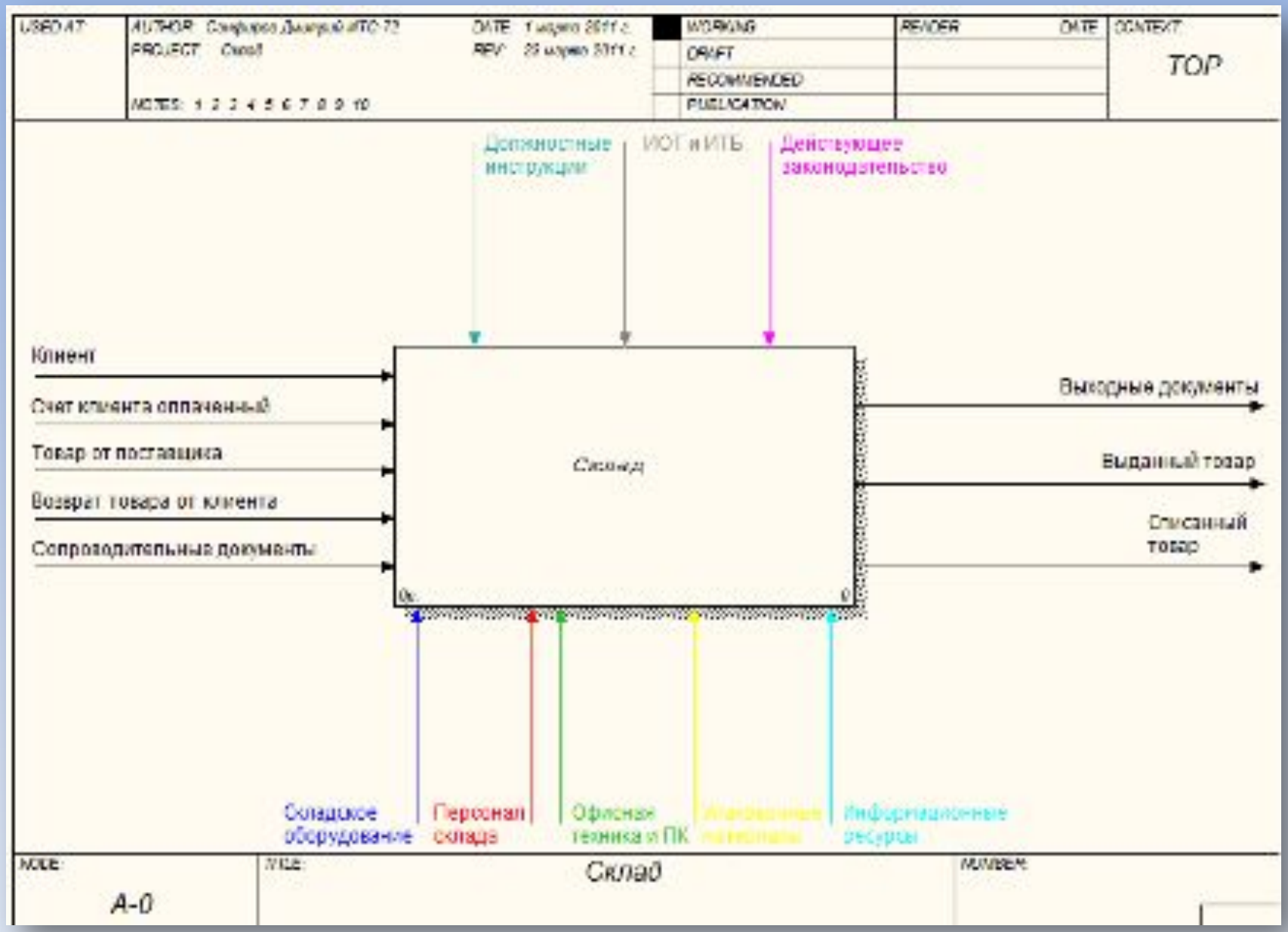
Этапы построения функциональной модели SADT:

1. Анализ предметной области (сбор информации об объекте и определение его границ);
2. Определение целей и задач, решаемых в ходе построения модели;
3. Построение, обобщение и декомпозиция диаграмм;
4. Критическая оценка, рецензирование и документирование;
5. Составление и завершение проектной документации.

Пример

Модель **Склад** описывает деятельность склада, а конкретно, выполняемые им функции:

- **Приемка товара**
- **Отгрузка и возврат товара**
- **Хранение**


USED AT
AUTHOR
PROJECT
DATE
REV
WORKING
DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT
TOP
NOTES: 1 2 3 4 5 6 7 8 9 10
Должностные инструкции
ИОТ и ИТБ
Действующее законодательство
Клиент
Счет клиента оплаченный
Товар от поставщика
Возврат товара от клиента
Сопроводительные документы
Склад
Выходные документы
Выданный товар
Списанный товар
Складское оборудование
Персонал склада
Офисная техника и ПК
Информационные ресурсы
NODE:
A-0
TITLE:
Склад
NUMBER:

Взаимодействие системы с окружающей средой - функционирование склада:

Входы (слева)

- Клиент
- Счет клиента оплаченный
- Товар от поставщика
- Возврат товара от клиента
- Сопроводительные документы

Выходы (справа)

- Выходные документы
- Выданный товар
- Списанный товар

Механизмы и управление (сверху)

- Действующее законодательство
- Должностные инструкции
- Инструкции по охране труда и технике безопасности

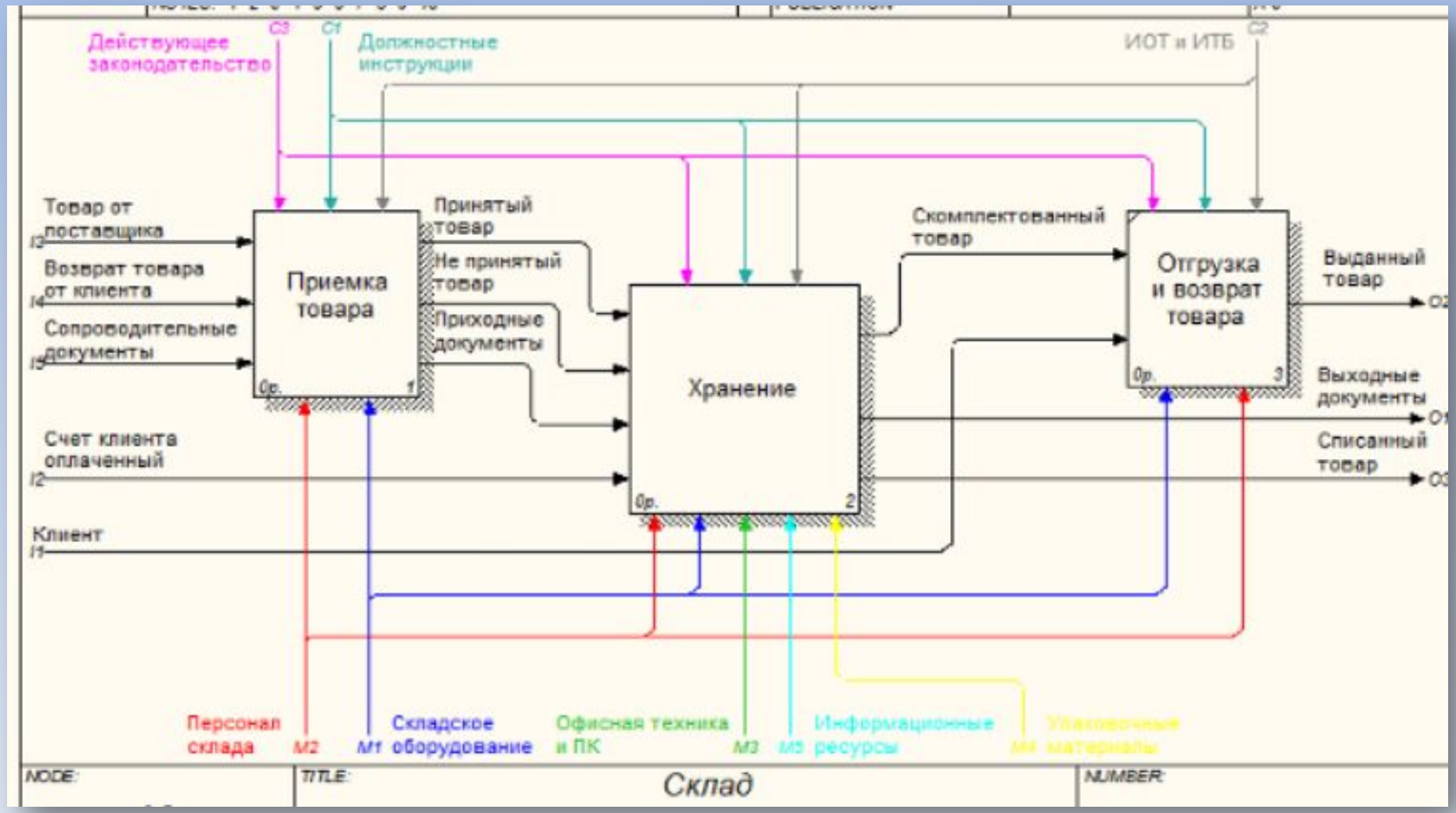
Ресурсы (снизу)

- Персонал склада
- Оборудование (складское и офисное)
- Информационные ресурсы
- Упаковочные (расходные) материалы

После описания контекстной диаграммы проводится **функциональная декомпозиция** - система разбивается на подсистемы и каждая подсистема описывается отдельно (диаграммы декомпозиции).

Затем каждая подсистема, при необходимости, разбивается на более мелкие и так далее до достижения нужной степени подробности. В результате такого разбиения, каждый фрагмент системы изображается **на отдельной диаграмме декомпозиции**

Диаграммы декомпозиции IDEF0. Подсистема Хранение



Диаграммы декомпозиции IDEF0

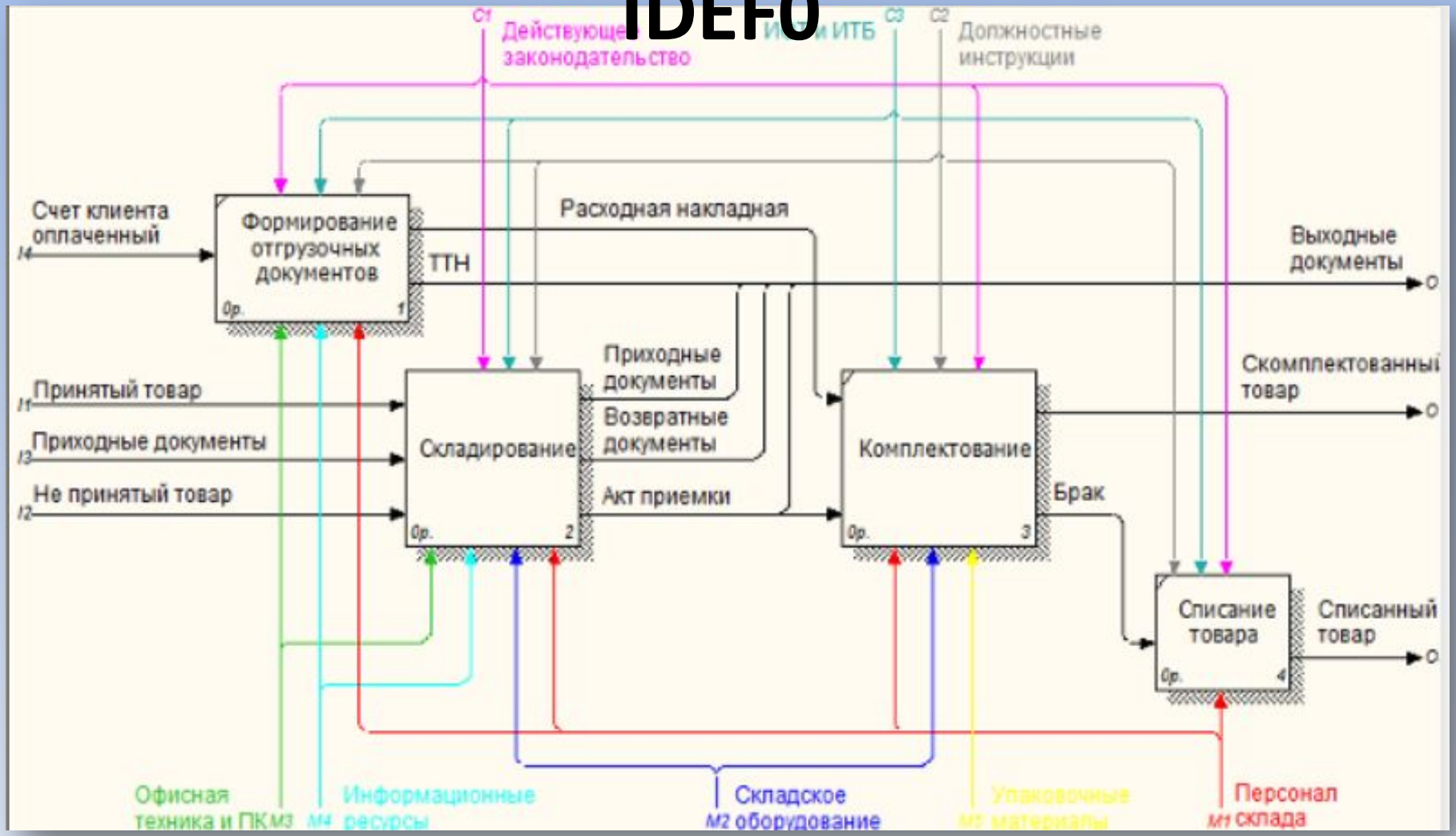
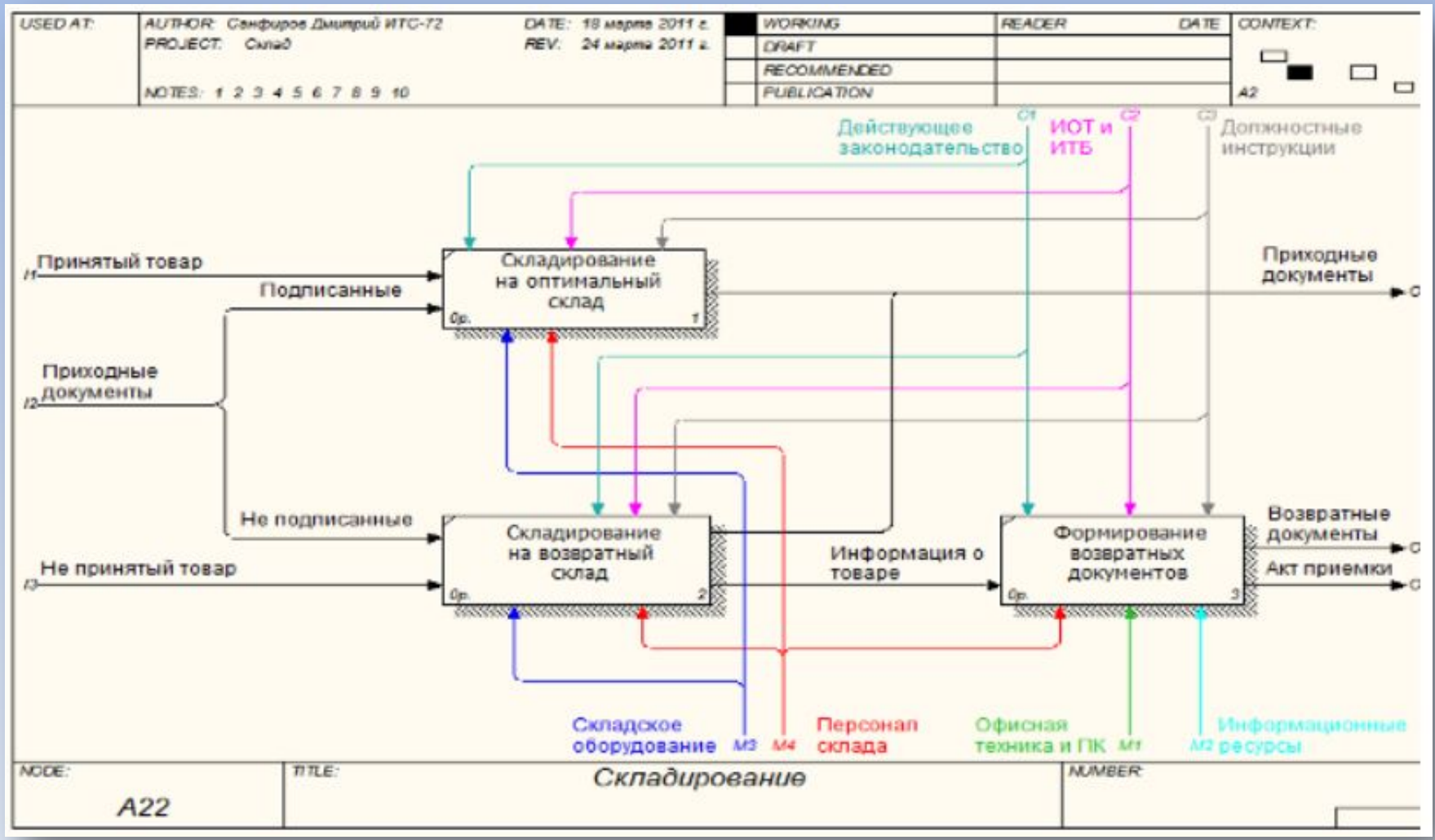


Диаграмма декомпозиции IDEF0. Складирование



Блоки на IDEF0-диаграмме размещаются по степени важности. В IDEF0 этот относительный порядок называется ***доминированием***. Доминирование понимается как влияние, которое один блок оказывает на другие блоки диаграммы.

Блоки принято располагать по диагонали диаграммы. **Наиболее доминирующий** блок обычно размещается в левом верхнем углу диаграммы, **наименее доминирующий** – в правом нижнем углу. Таким образом, топология диаграмм показывает, какие функции оказывают большее влияние на остальные.

Блоки на IDEF0-диаграмме должны быть пронумерованы. Нумерация блоков выполняется в соответствии с порядком их доминирования (1 – наибольшее доминирование, 2 – следующее и т.д.). Порядок доминирования (номер блока) располагается в правом нижнем углу функционального блока.

Пять типов взаимосвязей IDEF0

- В методологии IDEF0 используется ***пять типов взаимосвязей*** между блоками для описания их отношений: управление, вход, обратная связь по управлению, обратная связь по входу, "выход-механизм“
- ***Отношение управления*** возникает тогда, когда выход одного блока непосредственно влияет на работу блока с меньшим доминированием, т.е:
    - *Отношение входа*.
    - *Обратная связь по управлению*
    - *Обратная связь по входу*
    - *Связь "выход-механизм"*
- Дуга чаще символизирует набор объектов.

Декомпозиция бизнес-процесса на составляющие его операции в стандарте IDEF 0

Контекстная диаграмма модели

- Диаграмма, состоящая из **одного блока** и его дуг, определяющая границу системы, называется *контекстной диаграммой модели*.
- Все, что лежит внутри блока, является *частью описываемой системы,* а все, лежащее вне его, образует *среду системы*.
- **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.

Преимущества IDEF0

Идея IDEF0 состоит в том, что бизнес-процессы (функции реального объекта бизнеса) представляются как некие преобразования входного потока в выходной под контролем (управлением) управляющего потока с использованием для преобразования механизма.

- Основные преимущества IDEF0 состоят в следующем:
- полнота описания бизнес-процесса (управление, информационные и материальные потоки, обратные связи);
- комплексность при декомпозиции (мигрирование и туннелирование стрелок);
- возможность агрегирования и детализации потоков данных и информации (разделение и слияние стрелок);
- наличие жестких требований методологии, обеспечивающих получение моделей процессов стандартного вида;
- простота документирования процессов; соответствие подхода к описанию процессов в IDEF0 стандартам ISO 9000:2000.
- Отсюда и общее назначение IDEF0 - это перестройка структуры функций, которая позволит повысить производительность и эффективность системы.

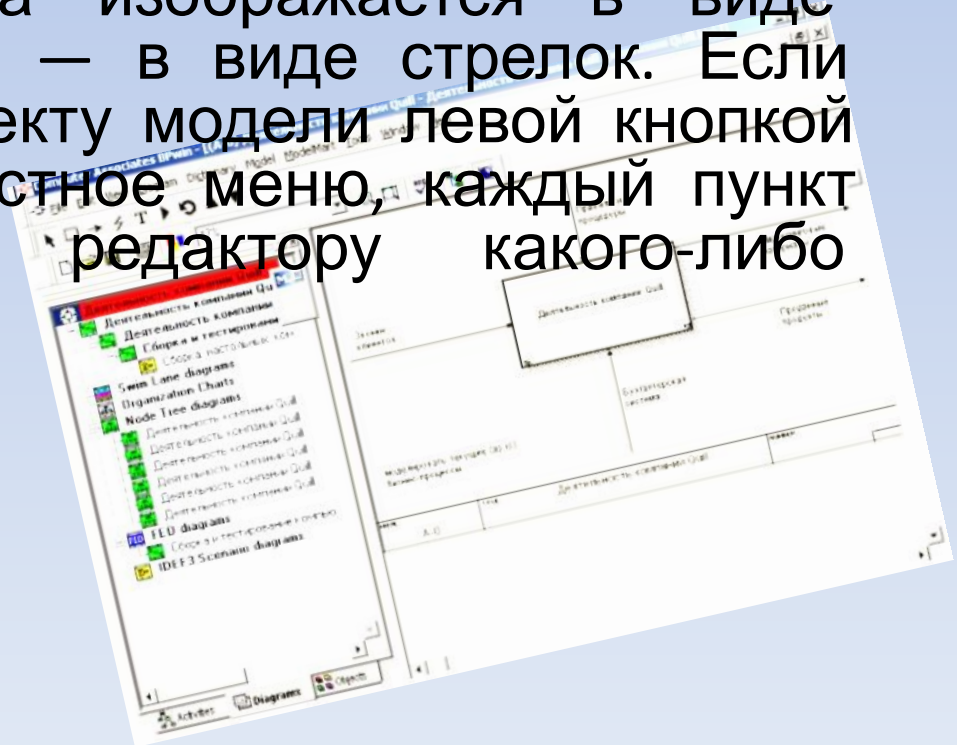
Среда построения диаграмм

Ramus — это программа, при помощи которой можно создавать визуальные диаграммы, используемые для наглядного отображения различных бизнес процессов. Данное решение будет крайне полезно на «мозговых штурмах» и собрания сотрудников предприятия. Помимо визуализации разных процессов и задач, создаваемые программой диаграммы также неплохо подходят для классификации и систематизации различных данных.

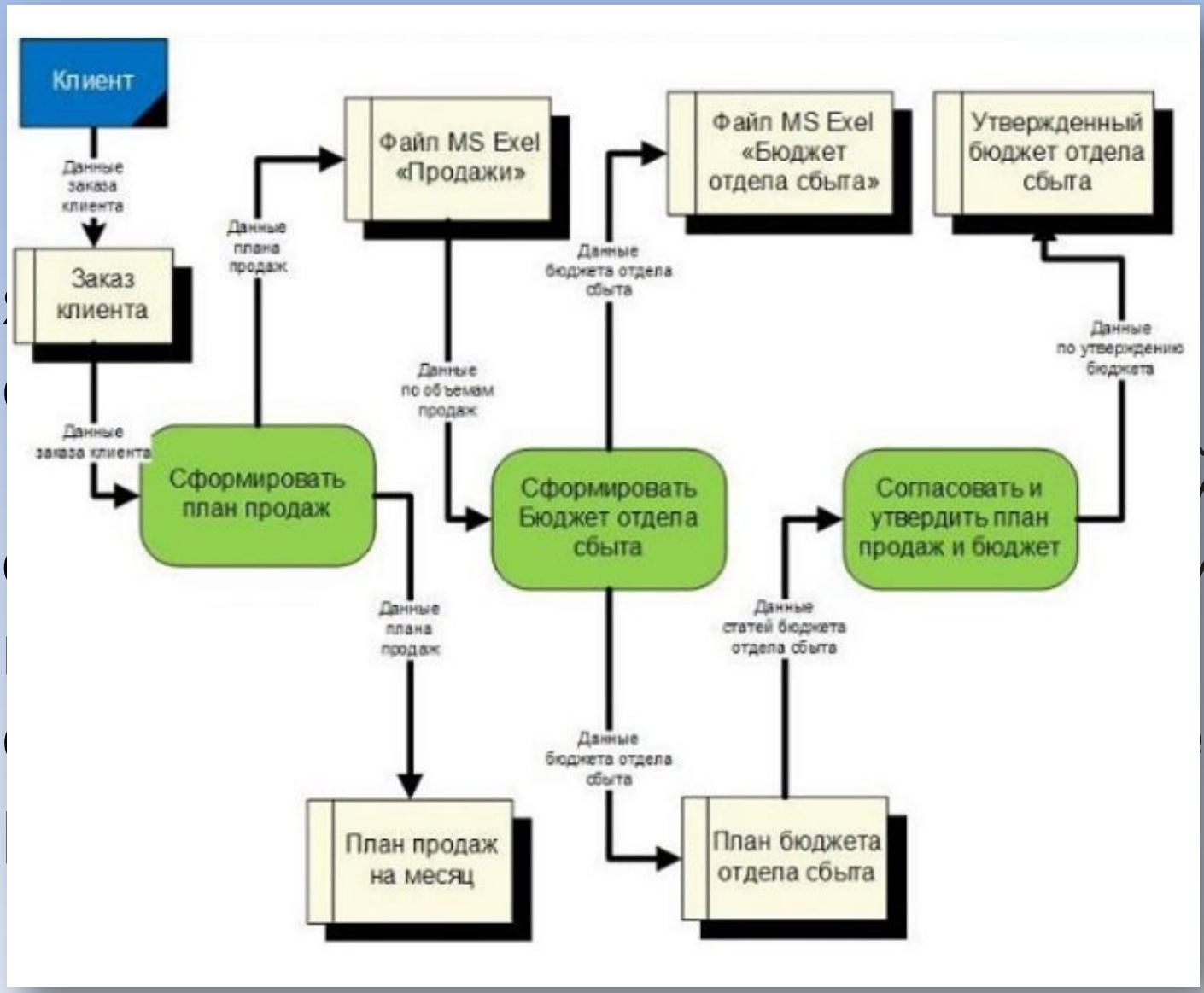
Главное преимущество Ramus Educational заключается в том, что она поддерживает сразу две популярных методологии: DFD и IDEF0.

BPWin - программа моделирования деловых процессов с помощью case-средств. BPwin поддерживает три методологии моделирования: функциональное моделирование (IDEF0); описание бизнес-процессов (IDEF3); диаграммы потоков данных (DFD).

Модель в BPwin рассматривается как совокупность работ, каждая из которых оперирует с некоторым набором данных. Работа изображается в виде прямоугольников, данные — в виде стрелок. Если щелкнуть по любому объекту модели левой кнопкой мыши, появляется контекстное меню, каждый пункт которого соответствует редактору какого-либо свойства объекта.



Диаграммы потоков данных



Диаграммы потоков данных (DFD)

- это способ представления процессов обработки информации.

Подобно IDEF0, DFD представляет систему как сеть процессов, связанных между собой с помощью стрелок.

В отличие от стрелок IDEF0, которые представляют собой жесткие взаимосвязи, стрелки DFD (потоки данных) показывают, как объекты (включая и данные) реально перемещаются от одной функции к другой. Это представление потока данных обеспечивает отражение в модели DFD таких физических характеристик системы, как движение объектов, хранение объектов, распространение объектов.

Диаграммы DFD обеспечивают удобный способ описания передаваемой информации как между частями моделируемой системы, так и между системой и внешним миром.

DFD используются для создания моделей информационного обмена организации, например, модели документооборота; при построении корпоративных

- Диаграммы потоков данных (DFD, Data Flow Diagramming) - это способ представления процессов обработки информации;
- Элементы модели: **процесс, поток, хранилище** - представляющие обработку, передачу и хранение данных (или материальных объектов).
- **Внешняя сущность** (External Reference) - изображается прямоугольником с тенью - материальный объект или физическое лицо, представляющее собой **источник** или **приемник** информации, который находится за пределами границы системы.

- **Поток данных** (Data Flow) - изображается стрелкой, описывает движение объектов (включая данные) от одного процесса к другому.
- **Процесс** - изображается прямоугольником со скругленными углами, это функция обработки информации, преобразующая входные потоки в выходные; совпадает со смыслом блоков IDEF0; имеет входы и выходы, не поддерживает управление и механизмы.
- **Хранилище данных** (Data Store) – прямоугольник с боковиной, описывает объекты (данные), которые необходимо сохранить в памяти прежде, чем использовать

Процесс

- В DFD **процессы** представляют собой **функции системы**, преобразующие входы в выходы. Процессы изображаются **прямоугольниками** со скругленными углами. Их смысл совпадает с блоком IDEF0 и единицами работы IDEF3. Так же, как и в IDEF3, они имеют входы и выходы, но не поддерживают управление и механизмы.
- **Каждый процесс должен быть именован глаголом с последующим дополнением. Кроме того, каждый процесс должен иметь уникальный номер**.
- Физически процесс может быть реализован различными способами: это может быть подразделение организации, выполняющее обработку входных документов и выпуск отчетов, программа, аппаратно реализованное логическое устройство.

Поток данных

- **Потоки** данных определяют информацию, передаваемую через некоторое соединение **от источника к приемнику**. Поток данных изображается линией, заканчивающейся стрелкой, которая показывает направление потока. Каждая стрелка имеет имя, отражающее его содержание.
- Реальный поток данных может быть информацией, передаваемой по кабелю между двумя устройствами, пересылаемыми по почте письмами, магнитными дискетами, переносимыми с одного компьютера на другой.
- Поскольку в DFD каждая сторона блока не имеет четкого назначения, как в IDEF0, **то стрелки могут подходить и выходить из любой грани блока**. В DFD также применяется двунаправленные стрелки для описания диалогов типа команда-ответ.

Хранилище данных

- В отличие от потока данных, описывающих объекты в движении, **хранилища данных** изображают объекты в покое. Хранилище представляет собой абстрактное устройство для хранения информации, которую можно в любой момент поместить в хранилище и через некоторое время извлечь, причем способы помещения могут быть любыми.
- Имя хранилища должно идентифицировать его содержимое и быть существительным.
- Хранилище физически может быть реализовано в виде ящика в картотеке, таблицы в оперативной памяти, файла на магнитном диске. То есть хранилище в общем случае является прообразом будущей базы данных.

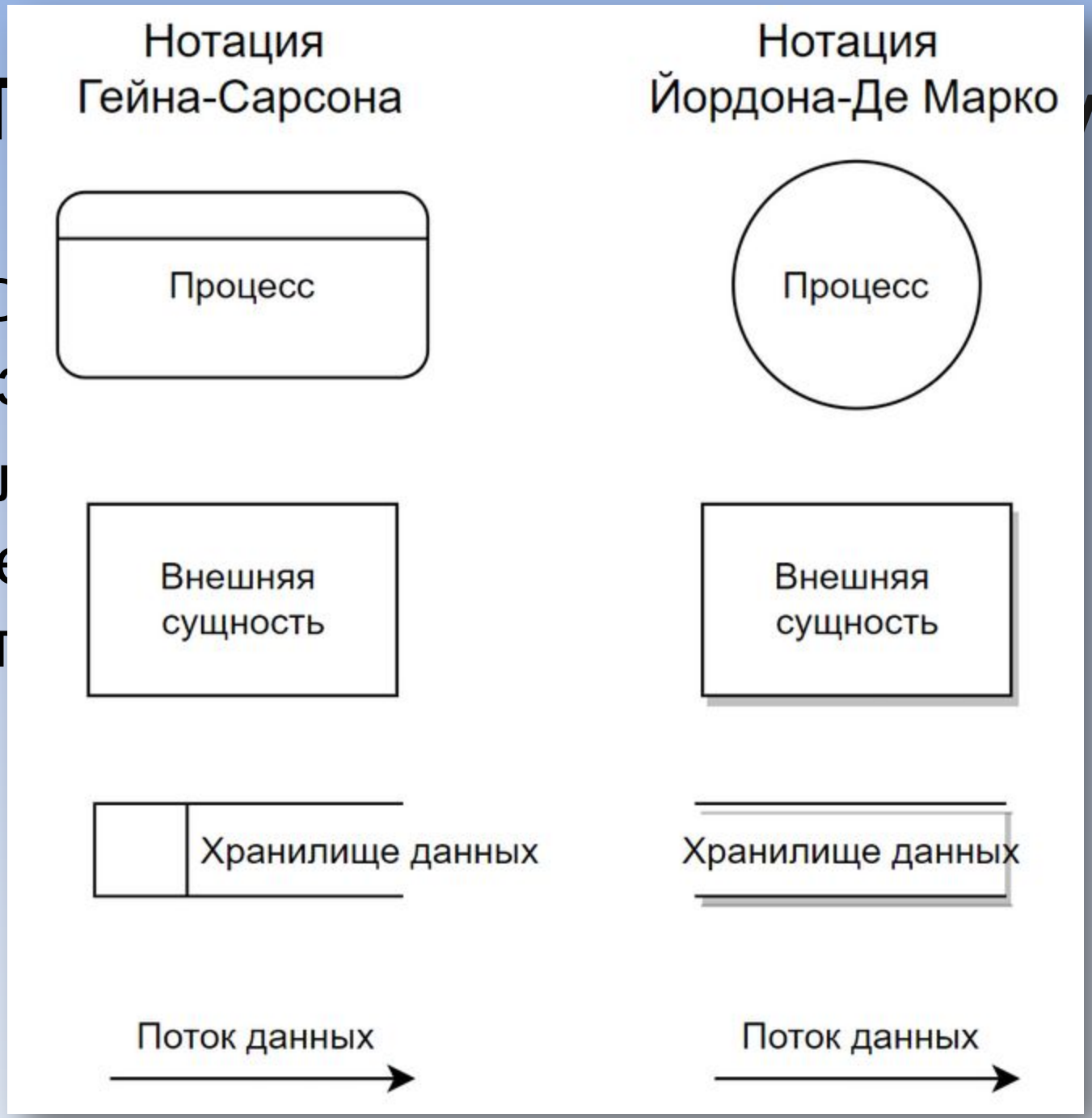
Внешняя сущность

- **Внешняя сущность** представляет собой материальный объект или физическое лицо, представляющее собой источник или приемник информации (например, заказчик, персонал, поставщик, клиенты, склад). Ее имя должно содержать существительные. Определение некоторого объекта или системы в качестве внешней сущности указывает на то, что она находится за пределами границ анализируемой системы.
- Внешняя сущность обозначаются в виде **прямоугольника** с тенью и обычно располагаются по краям диаграммы. Одна внешняя сущность может быть использована многократно на одной диаграмме. Обычно такой прием используют, чтобы не рисовать слишком длинных и запутанных стрелок.

Правила построения диаграмм:

- У каждого процесса должен быть как минимум один вход и один выход. Так как процесс в данной нотации – это обработка данных, то они должны поступать и исходить в заданном направлении.
- Процесс, связанный с обработкой данных, должен иметь внешний входящий поток. Запуск такого процесса возможен только при поступлении новой дополнительной информации, а не только накопленных в хранилище данных.
- Стрелки не могут проходить непосредственно между хранилищами, связь между ними возможна только через какой-либо процесса. Перемещать данные из одного хранилища в другое без дополнительной обработки – бессмысленно.
- Каждый процесс должен быть связан либо с другим процессом, либо с хранилищем данных. Процесс не может существовать самостоятельно, в нем нет смысла, если его результат никуда не передан.
- DFD-диаграмма предусматривает возможность декомпозиции крупных процессов на подпроцессы, которые будут подробно описаны. Возможно проведение декомпозиции до 3 – 4 уровней.

- В
раз
- Усл
эле
исп



Уровни диаграмм

- В зависимости от цели использования диаграммы можно отображать различные уровни детализации процесса.
- Для разговора и презентации процесса бизнес-пользователям и заказчикам, им важно понимать контекст и логику самого процесса, иногда нет смысла погружать их в технические моменты реализации.
- При разговоре с технической командой важно сделать акцент на реализации решения с технической точки зрения.

DFD диаграммы также можно делить на подобные уровни

- **Концептуальный (или контекстный)** уровень. Показывает общее описание процесса, который реализуется при потоке данных. Отображает абстрактно потоки данных, связанные с разными внешними сущностями

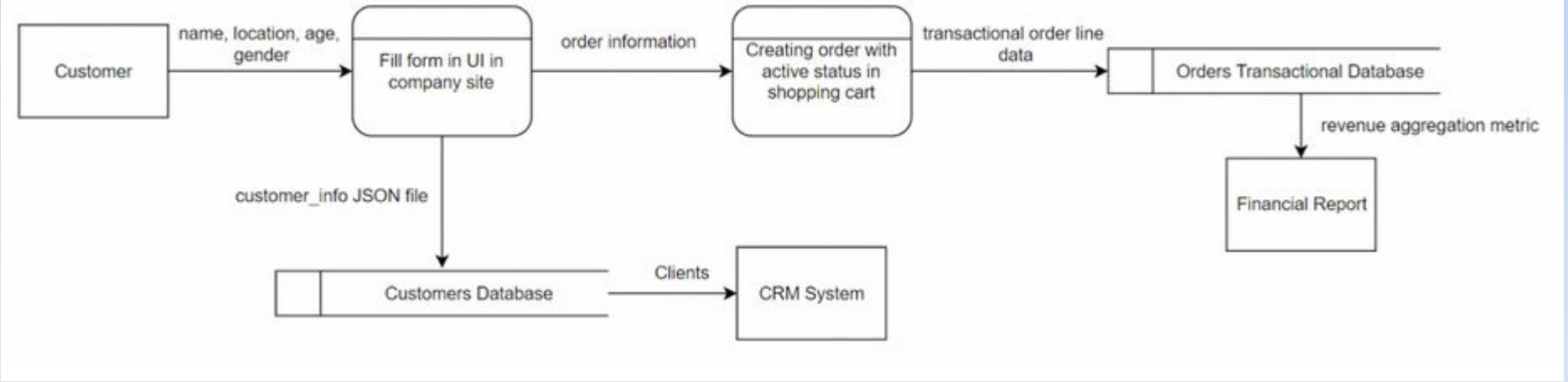
сущность **процесс** **хранилище**

- **Логический уровень**

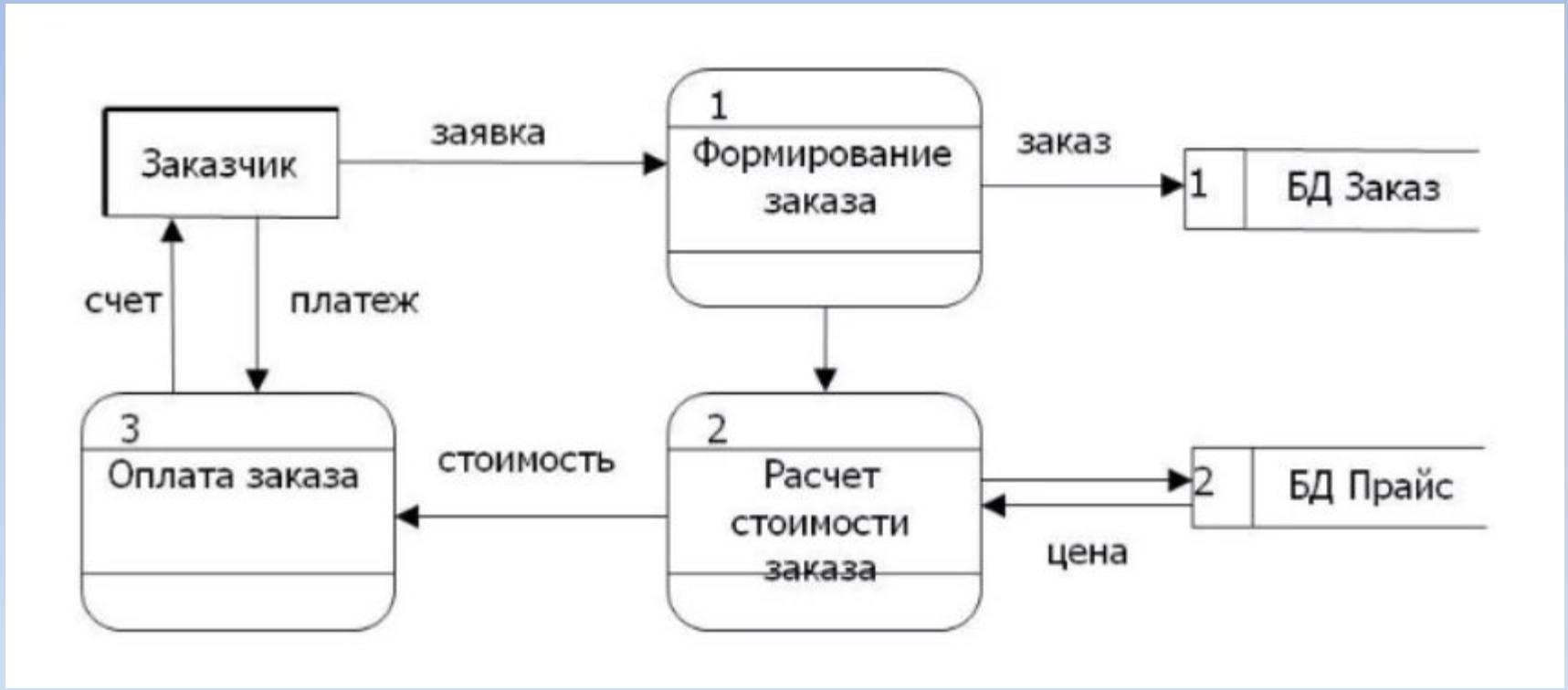
Отображает логику преобразования данных в системе в каждом процессе, описывает. Видны входные, промежуточные, выходные данные в каждом процессе, который протекает от внешней сущности до хранилищ данных. Больше указывает на вопрос “Что включает в себя процесс потока и обмена данными со стороны бизнеса?”

- **Физический уровень**

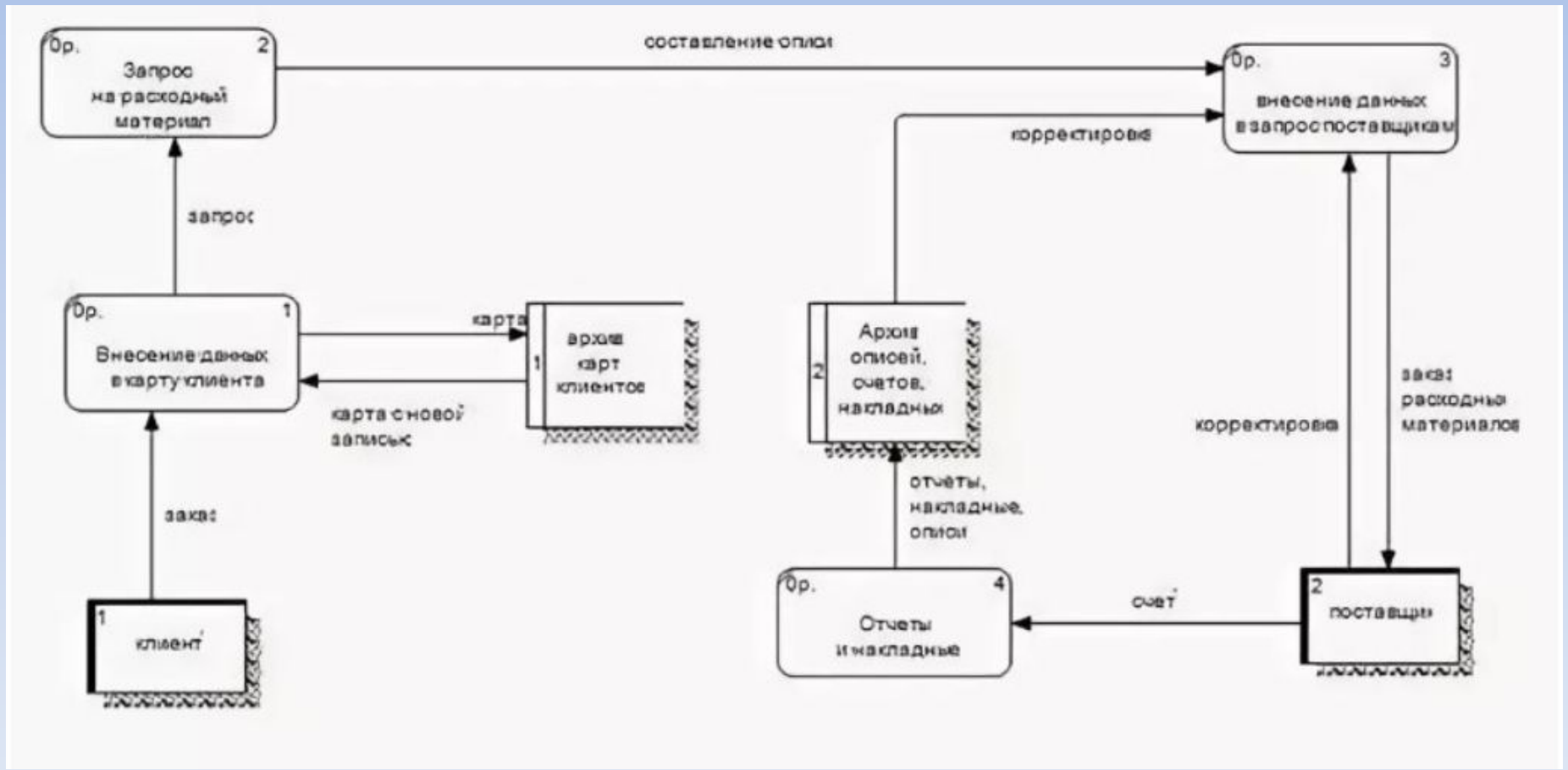
Включают точное отображение хранилищ данных, названий сущностей данных. Диаграмма физического уровня должна отвечать на вопрос “Как будет реализован процесс передачи и потока данных?”. Это программы и ручные процедуры.



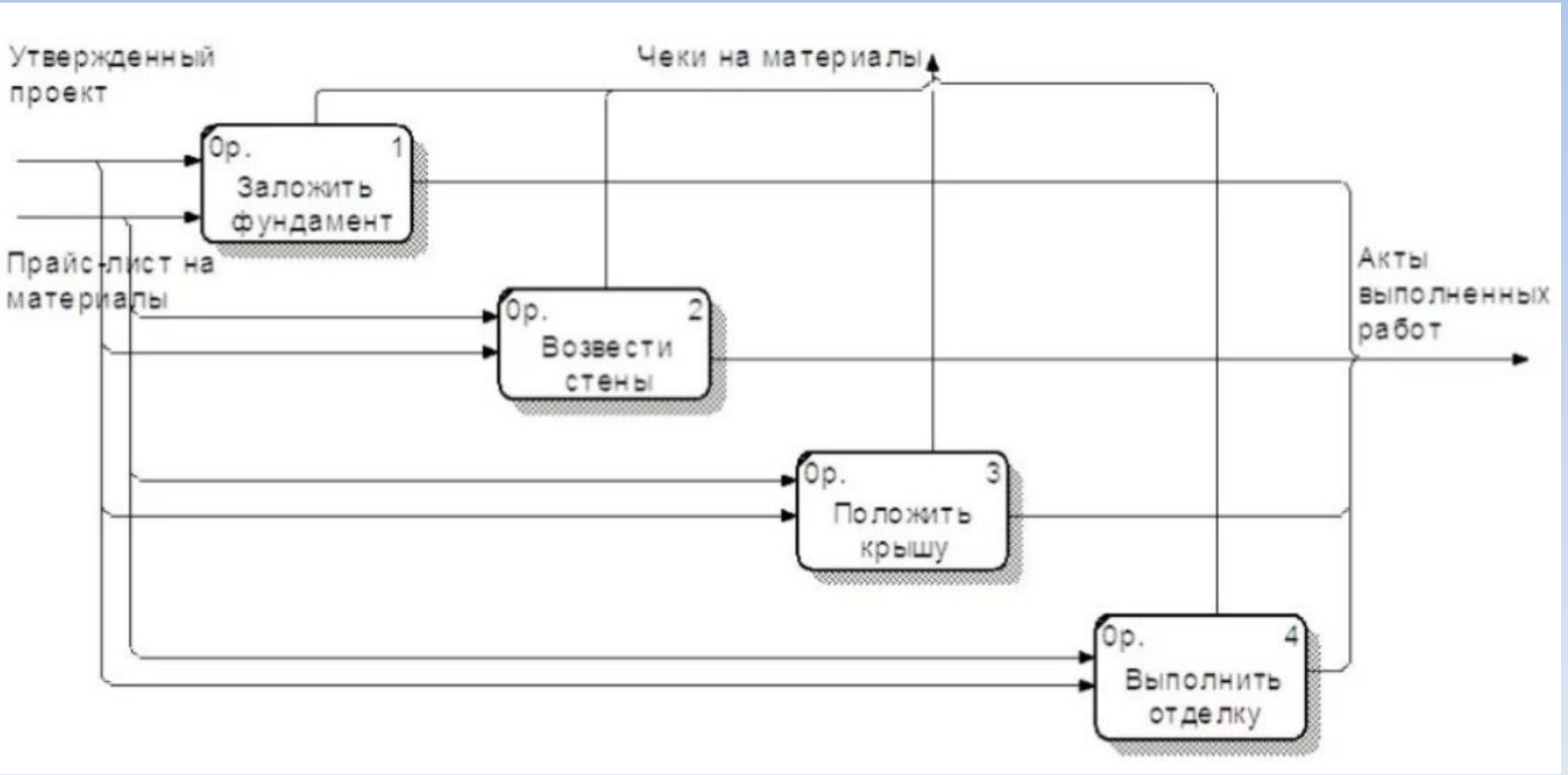
Примеры


Заказчик
заявка
1
Формирование заказа
заказ
1
БД Заказ
счет
платеж
3
Оплата заказа
стоимость
2
Расчет стоимости заказа
2
БД Прайс
цена

- Диаграмма DFD может носить **логический или физический характер.**
- На схеме ниже представлена **логическая схема**, в которой все внимание сконцентрировано на компании.



- **Физическая диаграмма**.
- Здесь рассмотрена конкретная система и представлены действия, необходимые для ее работы



Вопросы для самоконтроля

1. Какие модели описывают проектируемое ПО?
2. Каково их назначение?
3. Какой принцип нотации применяется в разных моделях?

- http://www.kgau.ru/istiki/umk/mbp/ch13.html#idp2177688
- http://www.interface.ru/fset.asp?Url=/CASE/defs91.htm?ysclid=l30ltsed1t
- https://akiselev87.files.wordpress.com/2011/03/d0b5d181d183d0b7.pdf
- https://intuit.ru/studies/courses/2195/55/lecture/1630
- https://youtu.be/AYfluzWo7wE

Построение диаграммы DFD

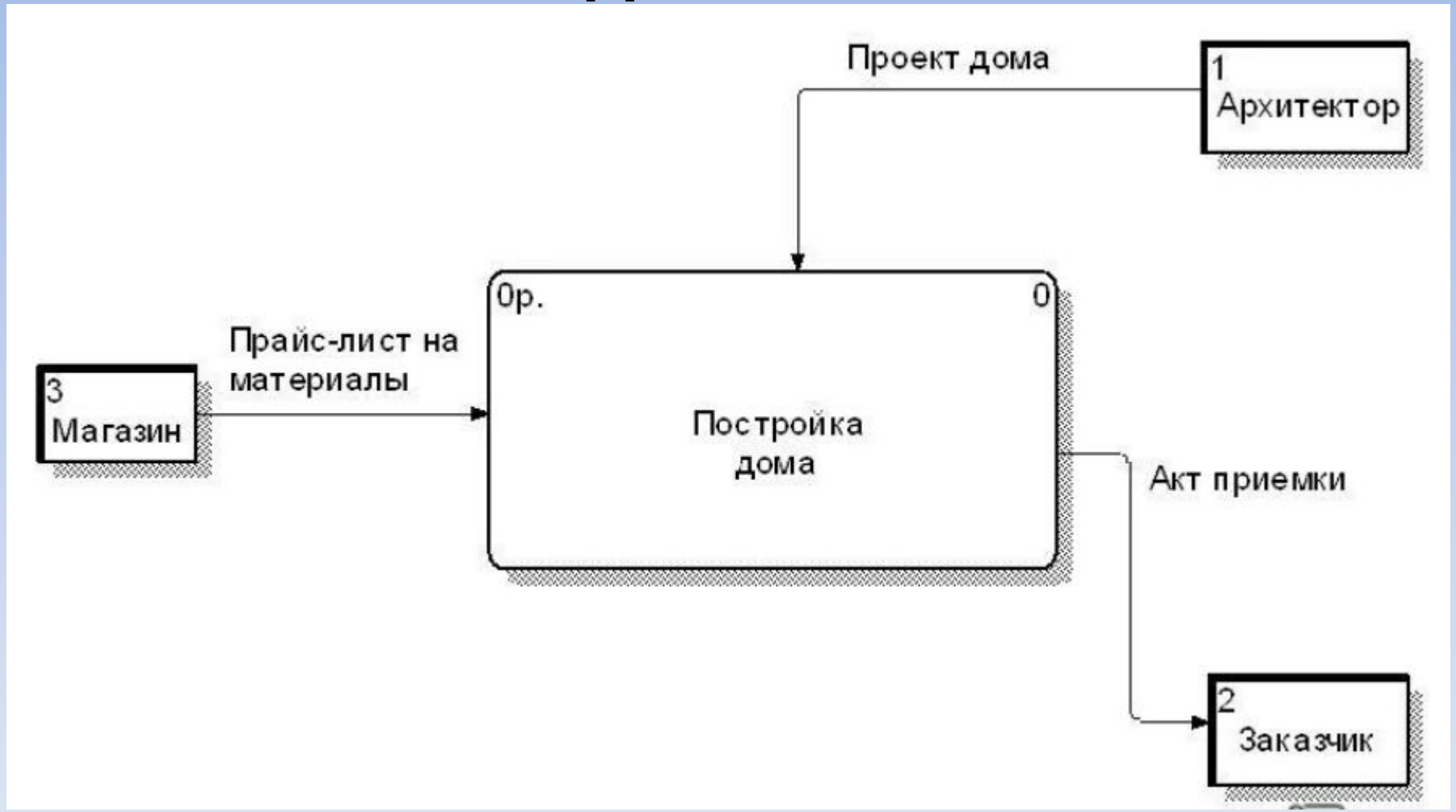
Диаграммы DFD могут быть построены с использованием структурного анализа, подобно тому, как строятся диаграммы IDEF0.

1. Сначала строится физическая модель, отображающая текущее состояние дел.
2. Затем эта модель преобразуется в логическую модель, которая отображает требования к существующей системе.
3. После этого строится модель отображающая требования к будущей системе.
4. И наконец, строится физическая модель, на основе которой должна быть построена новая система.

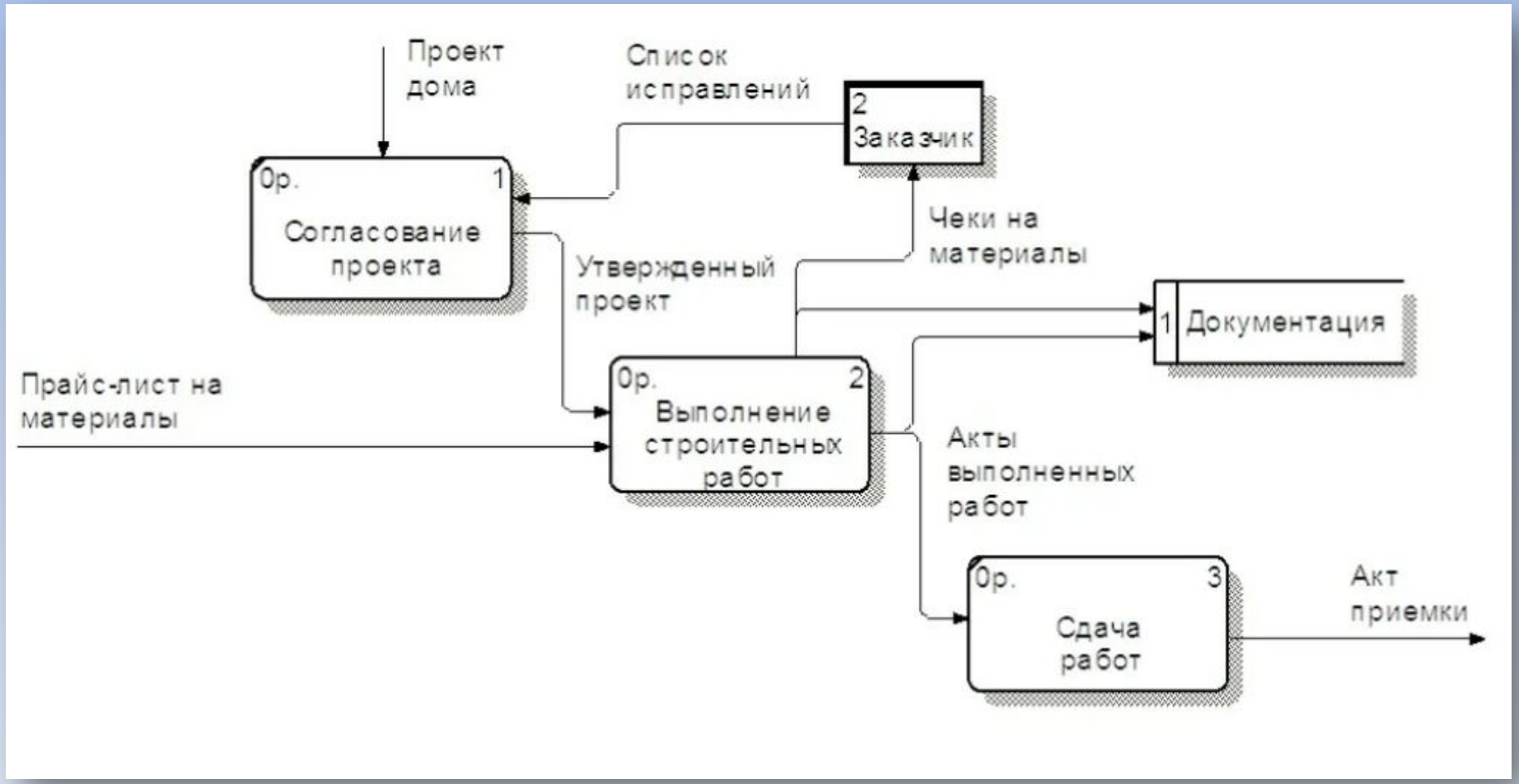
Другой подход называется событийным разделением, в котором для моделирования системы строится несколько моделей DFD.

1. Создается **логическая модель**, отображающая систему как совокупность процессов и описывающая, что система должна делать.
2. Затем строится **модель окружения**, которая описывает систему как объект, взаимодействующий с событиями из внешних сущностей. Эта модель содержит одну контекстную диаграмму и список событий. Контекстная диаграмма содержит один процесс, изображающий систему в целом и внешние сущности, с которыми система взаимодействует.
3. В конце создается **модель поведения**, показывающая, как система обрабатывает события. Эта модель состоит из одной диаграммы, в которой каждый блок изображает каждое событие из модели окружения. **Хранилища** могут быть добавлены для моделирования данных, которые необходимо запоминать между событиями. Потоки добавляются для связи с другими элементами.

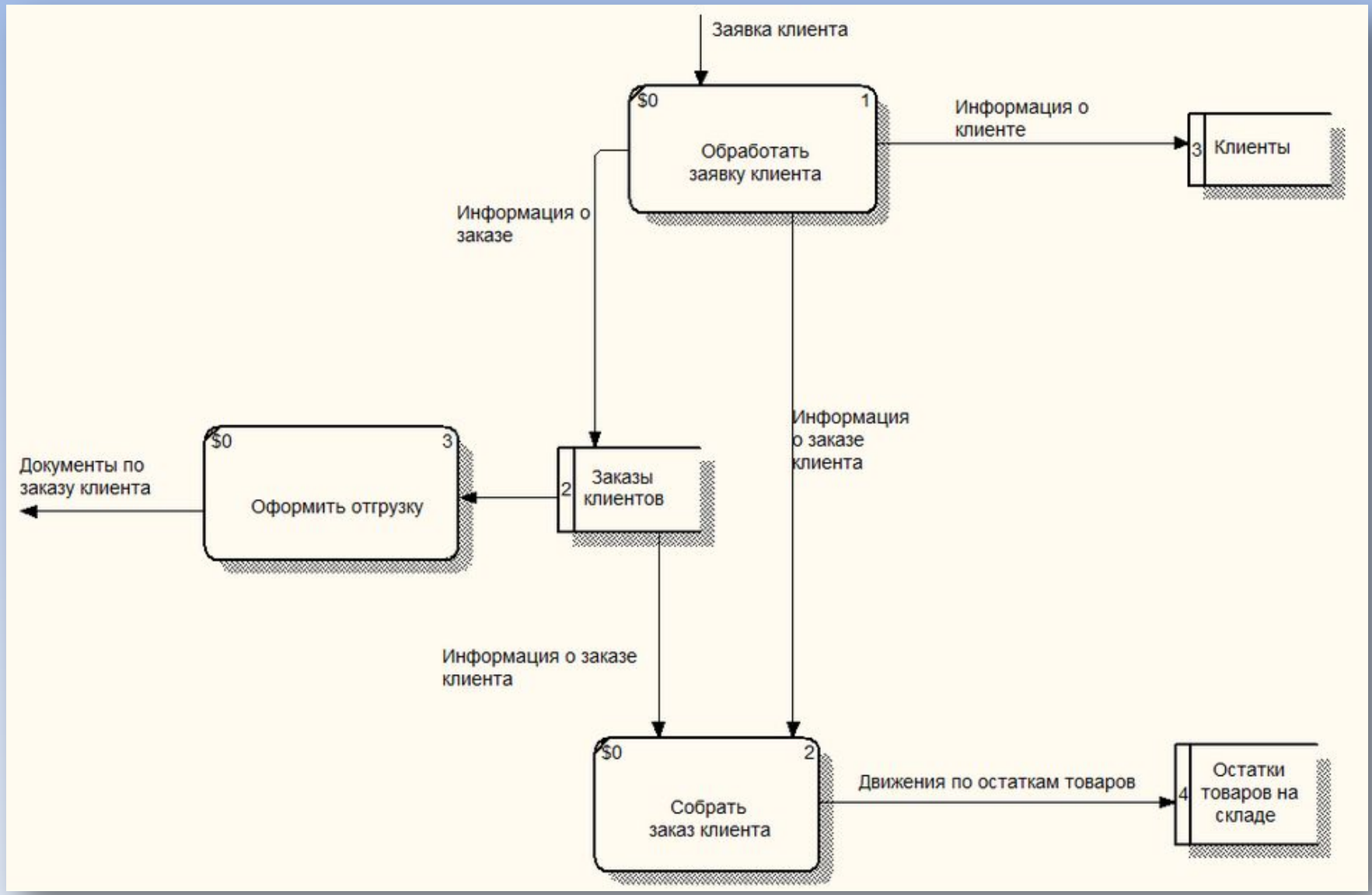
Контекстная диаграмма уровня системы. Постройка дачного домика



Постройка дачного домика



Заказ


Заявка клиента
A0
1
Обработать заявку клиента
Информация о клиенте
3
Клиенты
Информация о заказе
Информация о заказе клиента
A0
3
Оформить отгрузку
Документы по заказу клиента
2
Заказы клиентов
Информация о заказе клиента
A0
2
Собрать заказ клиента
Движения по остаткам товаров
4
Остатки товаров на складе