Дисциплина:
Проектирование и архитектура программных систем

Практическое занятие 3:
Нотации, языки моделирования структур ПС

Преподаватель:

Смирнов Константин Алексеевич

konst17@mail.ru

+7(981)-680-75-60

+7(921)-301-64-21

Модель программной системы

Под моделью ПС в общем случае понимается формализованное описание системы на определенном уровне абстракции. Каждая модель определяет конкретный аспект системы, использует *набор диаграмм и документов заданного формата*, а также отражает точку зрения и является объектом деятельности различных людей с конкретными интересами, ролями или задачами.

Цель модели: получение полного, точного и адекватного описания системы, имеющее конкретное назначение.

Модель ПС позволяет:

- справиться с растущей сложностью разрабатываемых программных систем;
- эффективно управлять разработкой в условиях изменяющихся требований;
- обеспечить взаимодействия между руководителем проекта, аналитиком, разработчиком, тестировщиком и пр.;
- обеспечить контроль изменений в процессе выполнения работ;
- избежать субъективности в оценке качества разрабатываемых продуктов.

Для ПС строятся два вида моделей.

Модели «AS-IS» («как есть») отражают существующее на момент обследования положение дел в организации и позволяют понять, каким образом функционирует данная организация. С помощью этой модели удобно выявлять узкие места и формулировать предложения по улучшению ситуации.

Модели «AS-TO-BE» («как должно быть») отражают представление о новых процессах и технологиях работы организации. Построение таких моделей будет более эффективно, если использовать результаты модели «AS-IS».

Сложные системы характеризуются выполняемыми процессами (функциями), структурой и поведением во времени. Для адекватного моделирования этих аспектов в автоматизированных ПС различают функциональные, информационные и поведенческие модели, пересекающиеся друг с другом.

Функциональная модель системы описывает совокупность выполняемых системой функций, характеризует, морфологию системы (ее построение) — состав функциональных подсистем, их взаимосвязи.

Информационная модель отражает отношения между элементами системы в виде структур данных (состав и взаимосвязи).

Поведенческая (событийная) модель описывает информационные процессы (динамику функционирования), в ней фигурируют такие категории, как состояние системы, событие, переход из одного состояния в другое, условия перехода, последовательность событий.

Графические (визуальные) модели представляют собой средства для визуализации, описания функциональной структуры системы, последовательности выполняемых действий, передачи информации между функциональными процессами, выявления отношений между данными. Для разных целей необходимы разные типы моделей.

Формальные (математические) модели: системы уравнений, статистические модели, модели математического программирования и др. Формальные модели универсальны. Одна и та же модель может описывать различные процессы. Достоинство таких моделей в том, что они позволяют с помощью формального аппарата вычислений находить решение. Недостаток: не все системы удается описать некоторой единой математической моделью. Как правило, этот тип моделей используется для расчета отдельных характеристик процессов.

Семантические модели сохраняют семантику (смысл, содержание) объекта. Примеры: дерево целей, модель организационной структуры, модель информационных коммуникаций компании и т.п. Описывают объекты, их свойства, состояния и поведение, отношения между объектами. Представляются в виде графов, диаграмм, таблиц, блок-схем, словесных описаний. Семантические модели незаменимы на ранних этапах проектирования сложных систем, т.к. позволяют представить общую картину, Главное свойство семантической модели – краткость и понятность. Семантическая модель может служить каркасом для построения других моделей.

Статические модели не учитывают временной параметр. Они отражают постоянные, устойчивые состояния объектов (систем, процессов), их состав, структуру, устойчивые внутренние и внешние связи.

Динамические модели отображают поток событий, т.е. изменение во времени состояний объектов системы (их параметров, характеристик, состояний). Последовательность взаимодействий объектов сложных систем позволяет представить функциональные особенности ИС в целом.

Разделяют **модели анализа и реализации.**

Модели анализа служат для определения требований, и совмещает в себе три модели:

- организационно-функциональную модель (отвечает на вопрос какая что делает и за что отвечает каждая часть ИС);
- процессно-ролевую модель (отвечает на вопрос кто-что-как-кому);
- модель структуры данных (отвечает на вопрос в каком виде, как и где хранятся данные, необходимые для функционирования ИС).

Модели реализации совмещают в себе следующие модели:

- функционально-технологическую модель (отвечает на вопрос что и как реализуется в ИС, содержит перечни функции ИС, способы реализации функций, средства реализации);
- процессно-ролевую модель (отвечает на вопрос какие функции выполняют различные субъекты ИС);
- количественную модель (отвечает на вопрос сколько необходимо ресурсов).

ОБЩИЕ ПРИНЦИПЫ МОДЕЛИРОВАНИЯ ИС

Этап построения модели ИС необходим для визуализации системы в ее текущем или желаемом состоянии. На этапе моделирования необходимо исследовать факторы, определяющие эффективность работы системы согласно выбранному критерию, выявить «узкие места», определить (качественно или количественно) эффективную стратегию управления системой, описать структуру системы и протекающие в ней процессы с возможностью генерации соответствующего программного кода.

Различные подходы к моделированию

Выбор методов построения моделей определяется целями проекта и в значительной мере влияет на весь его дальнейший ход.

Рациональный выбор возможен при понимании нескольких аспектов:

- целей проекта;
- особенностей разрабатываемой/внедряемой информационной системы;
- требований к информации необходимой для анализа и принятия решений в рамках конкретного проекта.

Выделяют два различных подхода к моделированию ИС: **структурный** и **объектно-ориентированный**.

Основными принципами **структурного подхода** к моделированию ИС являются:

- принцип «разделяй и властвуй» - принцип решения сложных проблем путем их разбиения на множество меньших, независимых задач;
- принцип иерархического упорядочивания – принцип организации составных частей проблемы в иерархические древовидные структуры;
- принцип абстрагирования – выделение существенных аспектов системы и отвлечение от несущественных;
- принцип формализации – необходимость строго методического подхода к решению проблемы;
- принцип непротиворечивости – обоснованность и согласованность элементов;
- принцип независимости данных – модели данных должны быть спроектированы и проанализированы независимо от процессов их логической обработки;
- использование графических нотаций.

Принципы **объектно-ориентированного подхода** к моделированию ИС следующие:

- принцип онтологизации системы;
- принцип декомпозиции;
- принцип инкапсуляции;
- принцип наследования;
- принцип полиморфизма.

Принцип онтологизации системы декларирует представление системы посредством классов, отражающих понятийную структуру предметной области в виде моделей (например, модель здания, модель уровневой системы образования и т.п.). Классы определяются атрибутами (количественными свойствами) и операциями (методами, поведенческими свойствами). Все множество экземпляров класса (объекты) также определяются через атрибуты и операции класса.

Принцип декомпозиции декларирует представление системы (предметной области или ИС) совокупностью взаимодействующих между собой объектов. Объекты являются экземплярами классов и взаимодействуют посредством передачи сообщений.

Принцип инкапсуляции – декларирует запрещение любого доступа к атрибутам объекта, кроме как через его операции (методы). В соответствии с этим принципом внутренняя структура объекта скрыта от пользователя, а любое его действие инициируется внешним сообщением, вызывающим выполнение соответствующей операции.

Принцип наследования - декларирует создание новых классов от общего к частному; новые классы сохраняют все свойства классов-родителей, а также содержат дополнительные атрибуты и операции, характеризующие их специфику.

Принцип полиморфизма - декларирует возможность работы с объектом без информации о конкретном классе, экземпляром которого он является.

Ключевыми преимуществами объектно-ориентированного подхода является возможность повторного использования (объектно- ориентированные системы могут быть легко собраны из ранее разработанных программных компонентов), а также расширяемость (системы будут легко расширяться без какой либо модернизации повторно используемых компонентов).

Визуальное моделирование

Визуальное моделирование - это способ построения графической модели с помощью зрительных абстракций, воспроизводящих понятие и объекты реального мира. Графические модели - представляют собой средства для визуализации, описания, проектирования и документирования архитектуры системы. Визуальное моделирование – это моделирование с использованием некоторой графической ***нотации***.

Нотацией называют систему условных обозначений для графического представления визуальных моделей.

Разработка визуальных моделей сложных систем, в виду значительного объема решаемых задач, должно опираться на специальные средства программной поддержки. Для визуального моделирования используются специализированные программно-технологические средства. Ими стали средства, реализующие CASE-методологию создания и сопровождения ИС.

CASE (Computer Aided Software Engineering) – методология разработки программного обеспечения, основанная на комплексном использовании компьютеров не только для написания исходного кода, но и для анализа и моделирования соответствующей предметной области.

CASE-средства (CASE-tools) – программное обеспечение, которое предназначено для разработки визуальных моделей программных систем и генерации исходного кода или структуры базы данных на некотором языке:

1-е поколение: генерация схем БД (Oracle Designer 2000, ERwin),

2-е поколение: генерация программного кода (Borland Together Designer 2005),

3-е поколение: прямая и обратная кодогенерация (IBM Rational Rose 2002/2003, Borland Together Developer 2005, Sparx Enterprise Architect),

4-е поколение: синхронизация программного кода и моделей (IBM Rational Software Architect , Borland Together Architect , Borland Development Studio),

5-е поколение Графические нотации визуального моделирования.

IDEF — методологии семейства ICAM (Integrated Computer-Aided Manufacturing) для решения задач моделирования сложных систем, позволяет отображать и анализировать модели деятельности широкого спектра сложных систем в различных разрезах.

Существуют разные виды IDEF - методологии, которые отличаются между собой полнотой информации отображаемой на диаграмме:

IDEF0 (Function Modeling) – диаграмма для создания функциональной модели,

IDEF1 (Information Modeling) – диаграмма для построения информационной модели для поддержки функций производственной системы или среды.

IDEF2 (Simulation Model Design) – данный метод позволяет построить динамическую модель меняющегося во времени поведения функций, информации и ресурсов производственной системы или среды. Данная модель используется редко.

- **IDEF3 (Process Description Capture)** – данный метод используется для сбора информации о состоянии моделируемой системы. IDEF3 состоит из двух методов. Process Flow Description (PFD) – описание процессов, с описанием того, как организована работа между различными элементами моделируемой системы. Object State Transition Description (OSTD) – описание переходов состояний объектов, с описанием того, какие существуют промежуточные состояния у объектов в моделируемой системе.
- **IDEF4 (Object-Oriented Design)** – данный метод объектно- ориентированного планирования был разработан для поддержки объектно- ориентированной идеологии.
- **IDEF5 (Ontology Description Capture)** – данный метод позволяет разрабатывать, изучать и поддерживать онтологию моделируемой системы. Термин «онтология» включает в себя каталог терминов области знаний; правила, объясняющие, как термины могут комбинироваться, создавая при этом корректные ситуации в области знаний и согласованные выводы, используемые в моделируемой системе.

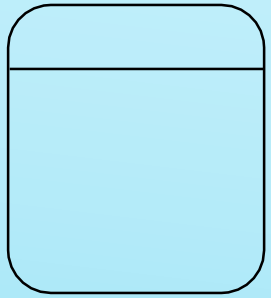
- **IDEF6 (Design Rational Capture Method) -** данный метод позволяет использовать рациональный опыт проектирования.
- **IDEF7 (Information System Auditing) -** данный метод описывает проведение методологии аудита информационной системы.
- **IDEF8 (User Interface Modeling)** – данный метод позволяет разрабатывать необходимые модели Графического Интерфейса Пользователя (Human-System Interaction Design). Метод предназначен для проектирования взаимодействия человека и технической системы.
- **IDEF9 (Business Constraint Discovery) -** данная модель предназначена для анализа имеющихся условий и ограничений (в том числе физических, юридических или любых других) и их влияния на принимаемые решения в процессе реинжиниринга.

- **UML 2.x -** Формальная спецификация последней версии UML 2.0 опубликована в августе 2005 года. Семантика языка была значительно уточнена и расширена для поддержки методологии.
- **Model Driven Development — MDD**. Последняя версия UML 2.5 опубликована в июне 2015 года.

- **DFD (Data Flow Diagram)** – диаграмма потоков данных, предложена Лари Константином в 70-е гг. XX века. Отображает пути, по которым циркулируют данные внутри системы, а также между системой и внешним миром. Как и в IDEF0 в основе – функции, выполняемые системой. Другое изображение – прямоугольники со скругленными краями.

НОТАЦИЯ DFD

Процесс



Выполняет какие-либо действия над данными, как например: создает, модифицирует, сохраняет, удаляет и т.д.

Хранилище данных

Используется для хранения данных.

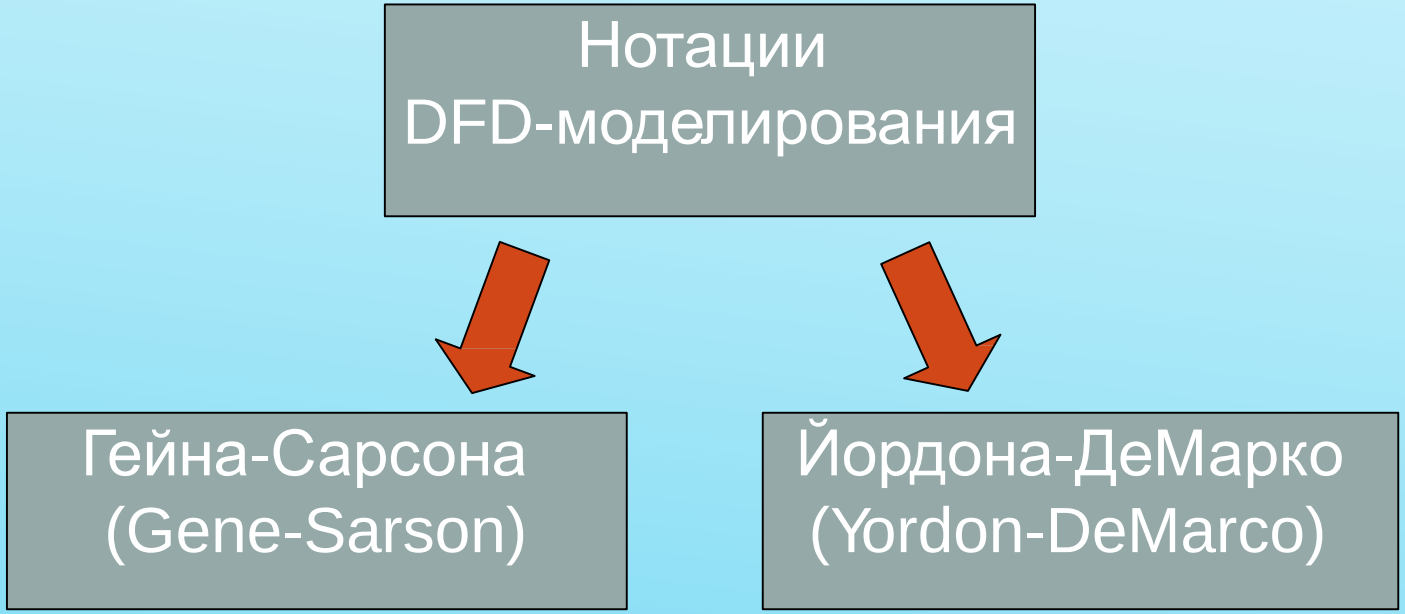
Внешняя сущность

Является источником или адресатом потока данных. Отображают внешние по отношению к системе сущности.

Поток данных

Потоки данных между процессами, хранилищами, внешними сущностями.

НОТАЦИИ, ИСПОЛЬЗУЕМЫЕ В DFD-МОДЕЛИРОВАНИИ

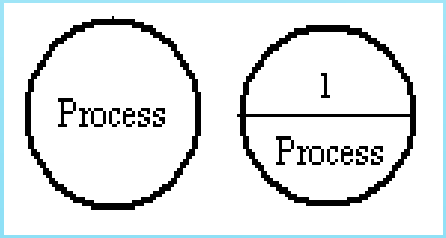


Примечание. В зависимости от используемой нотации графическое представление элементов диаграмм будет различным

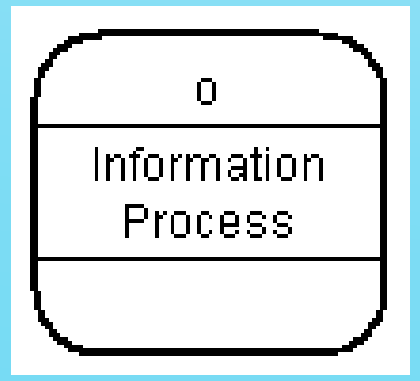
ДВЕ НОТАЦИИ DFD

Процесс
Трансформирует входящий поток данных в выходящий.

Нотация Йордона



Нотация Гейна-Сарсона

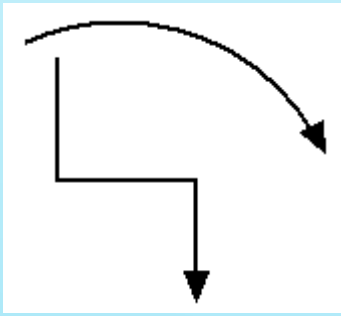


Хранилище данных
Используются для хранения данных в системе.

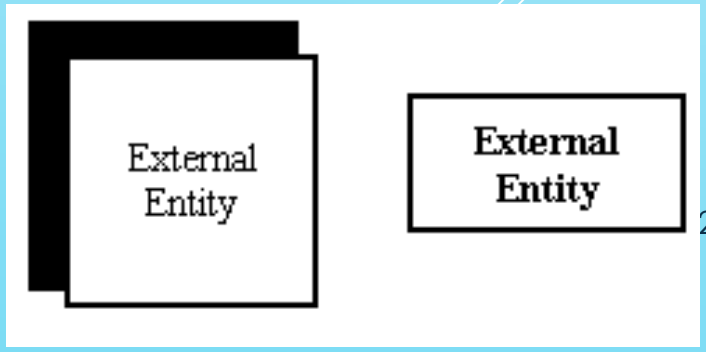
Нотация Йордона

Нотация Гейна-Сарсона

Поток данных
Стрелки, отображающие потоки данных..



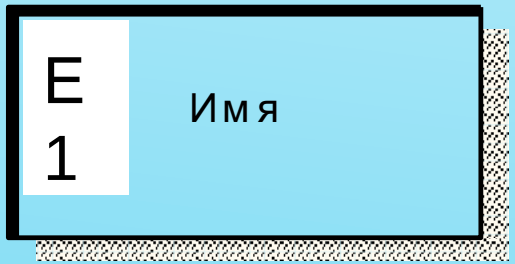
Внешняя сущность
Изображает сущности, находящиеся за рамками системы. Внешние сущности являются источниками или адресатами потоков данных системы.



ВНЕШНЯЯ СУЩНОСТЬ

- Представляет собой *материальный объект* или *физическое лицо*, являющееся источником или приемником информации (например, заказчики, клиенты, поставщики, склад, персонал, банк).
- Внешняя сущность находится за пределами анализируемой системы.
- Одна и та же внешняя сущность может быть использована многократно на одной или нескольких диаграммах.

Внешняя сущность в нотации Йордона-ДеМарко

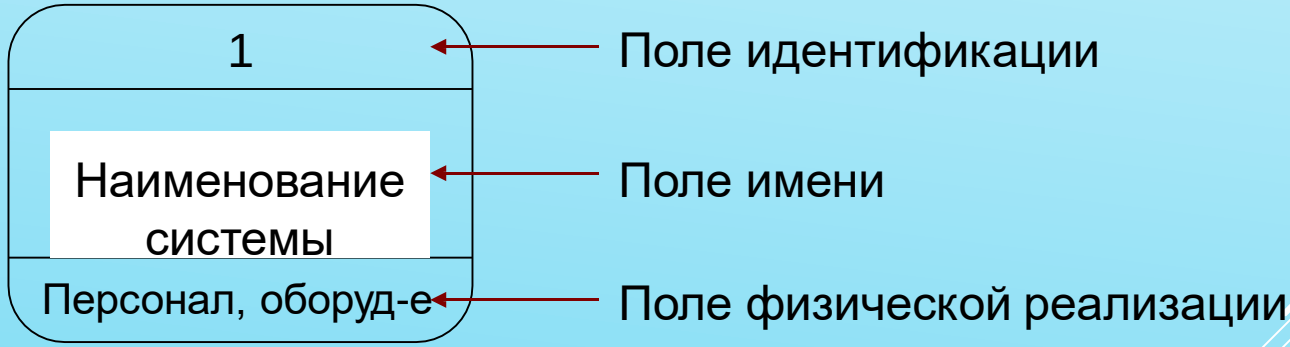


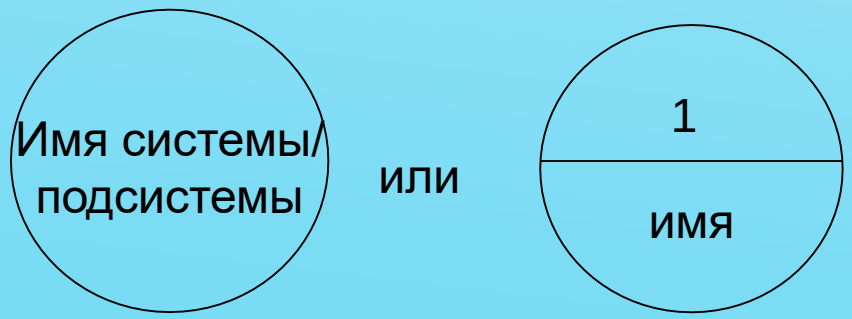
Внешняя сущность в нотации Гейна-Сарсона

СИСТЕМА И ПОДСИСТЕМА

- При построении модели сложной системы она может быть представлена в самом общем виде на так называемой контекстной диаграмме в виде одной системы, либо в виде ряда подсистем.
- *Наименование* системы/подсистемы представляется в виде словосочетания с отглагольным существительным (рассмотрение повестки дня, решение задачи, получение денег и т.п.).

Система/подсистема в нотации Гейна-Сарсона



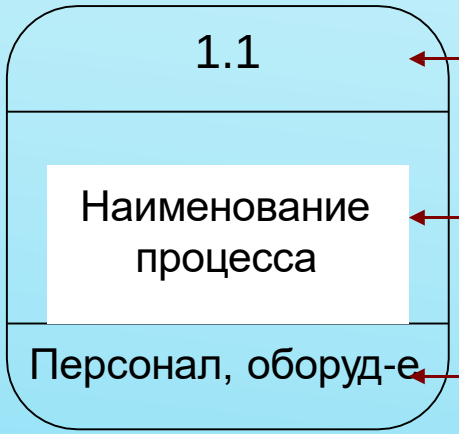


Система/подсистема в нотации Йордона-ДеМарко

ПРОЦЕСС

- Представляет собой *преобразование* входных потоков в выходные в соответствии с определенным алгоритмом.
- Примеры обработка входных документов и выпуск отчетности определенным подразделением, процессы физически реализованного устройства.
- Процесс *именуется* в виде словосочетания с активным глаголом в неопределенной форме, за которым следует существительное в винительном падеже.

ПРОЦЕСС

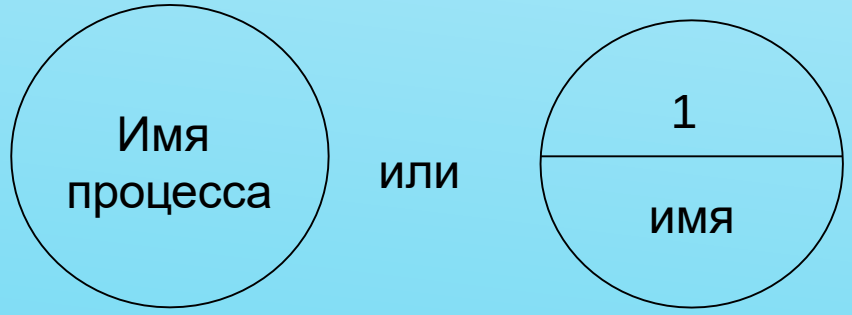


Поле идентификации

Поле имени

Поле физической реализации

Процесс в нотации Гейна-Сарсона



Процесс в нотации Йордона-ДеМарко

Процесс отличается от системы/подсистемы по полю наименования!!!!

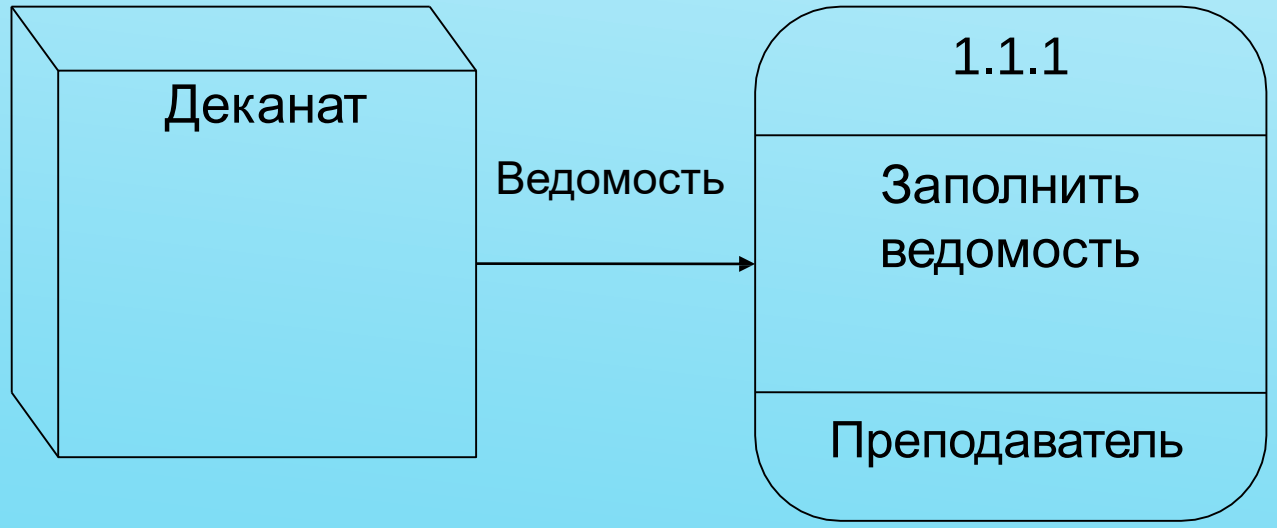
НАКОПИТЕЛЬ ДАННЫХ

Это абстрактное устройство для хранения информации, которую можно в любой момент поместить в накопитель и через некоторое время извлечь.

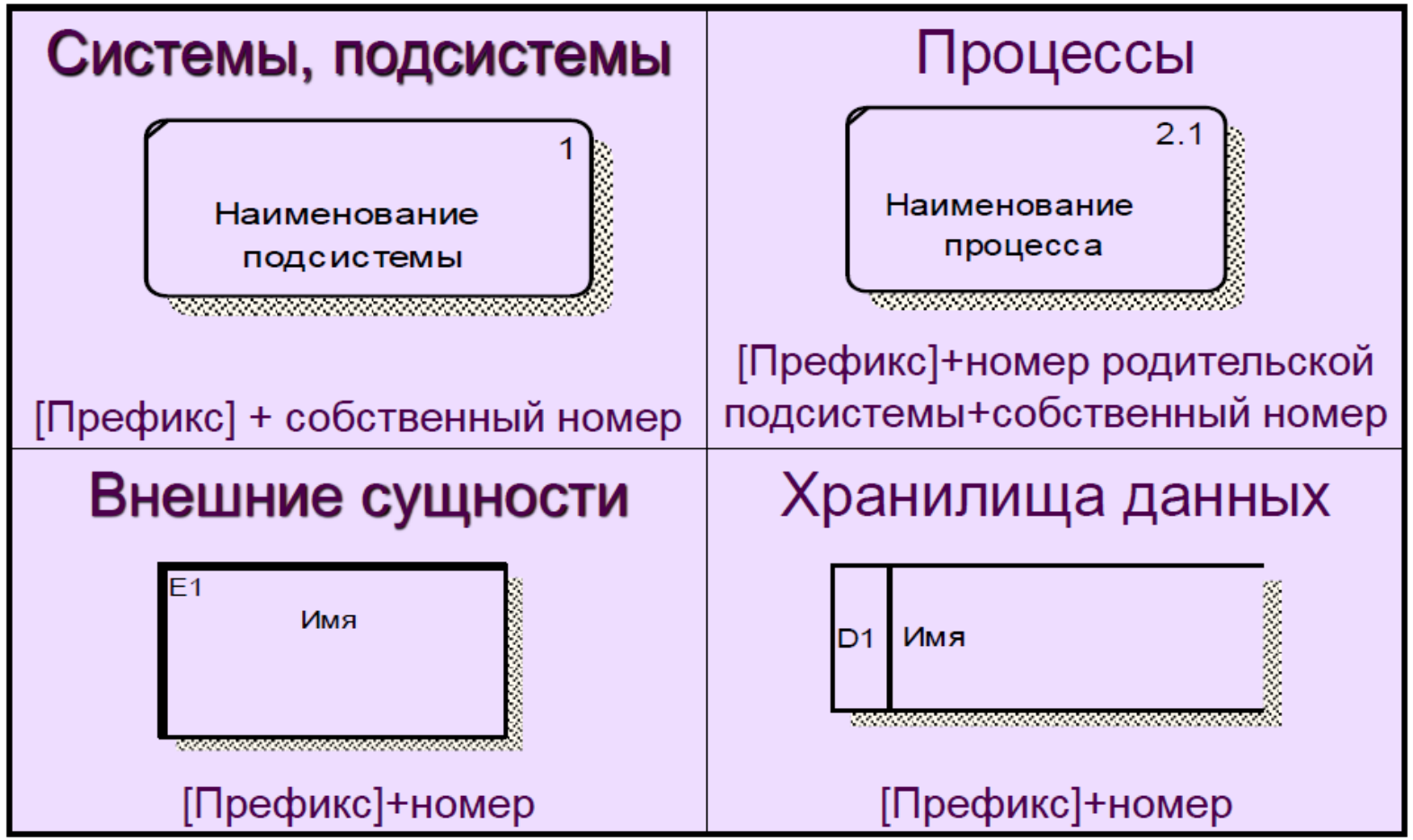
Примеры: ящик в картотеке, таблицы в ОЗУ, файл на электронном носителе.

ПОТОК ДАННЫХ

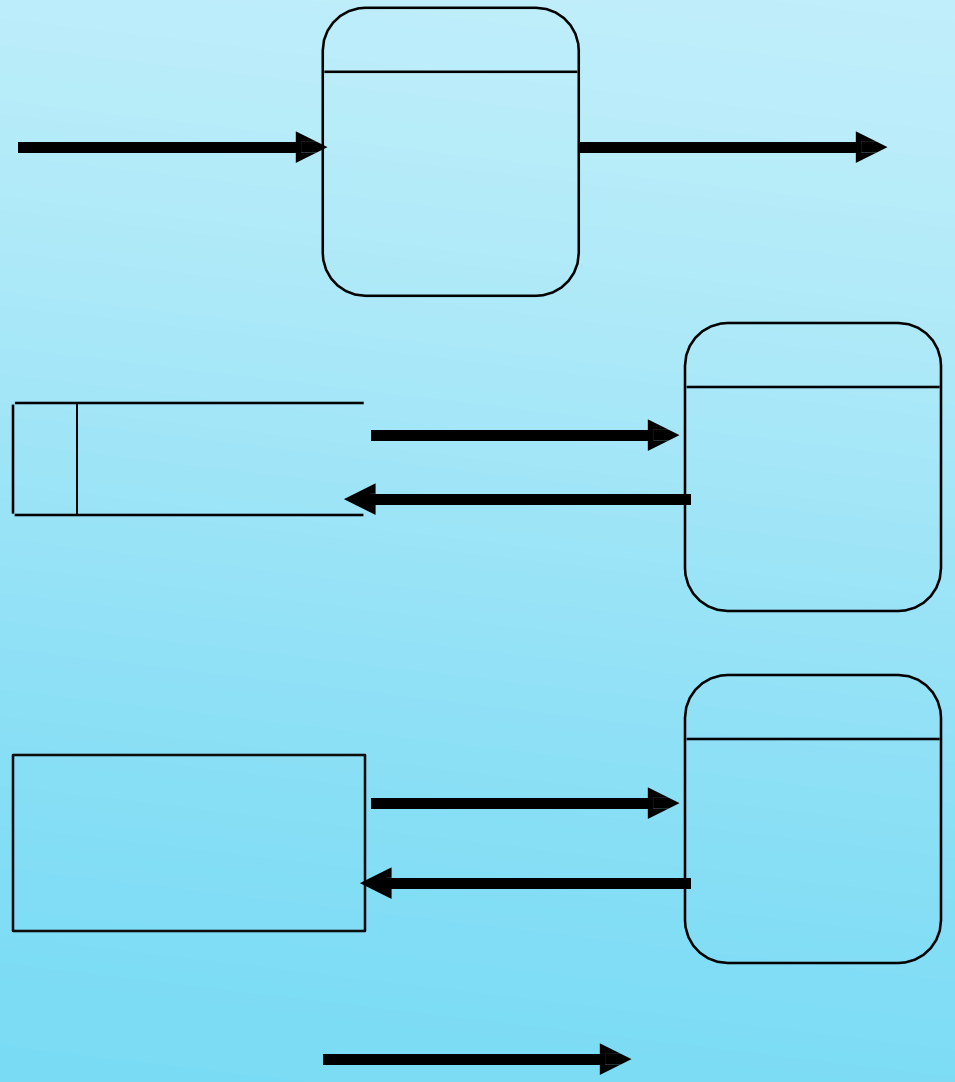
Определяет информацию, передаваемую через некоторые соединения от источника к приемнику. Реальный поток данных может быть информацией, передаваемой по кабелю между двумя устройствами, пересылаемыми по почте письмами и т.п.



НУМЕРАЦИЯ ОБЪЕКТОВ



ПРАВИЛА СОЗДАНИЯ ДИАГРАММ DFD

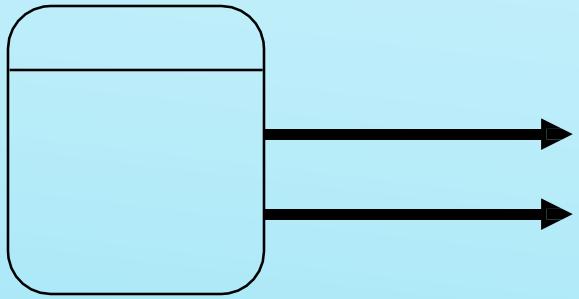


*Как минимум по **одному** потоку должны входить и выходить из каждого процесса или хранилища.*

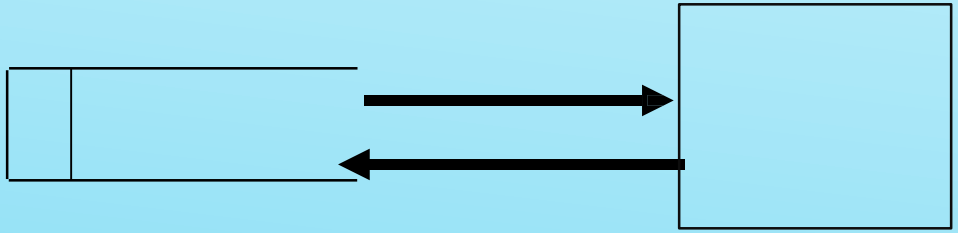
Хранилище и внешние сущности должны быть подсоединены к процессу (входящим, выходящим или обоими потоками).

Один поток должен идти только в одном направлении (все стрелки – однонаправленные).

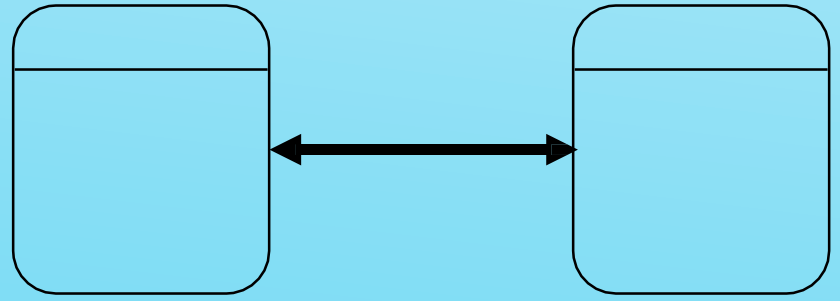
РАСПРОСТРАНЕННЫЕ ОШИБКИ DFD



У процесса есть выходящие потоки, но нет входящих.



Хранилище и внешний источник связаны напрямую.

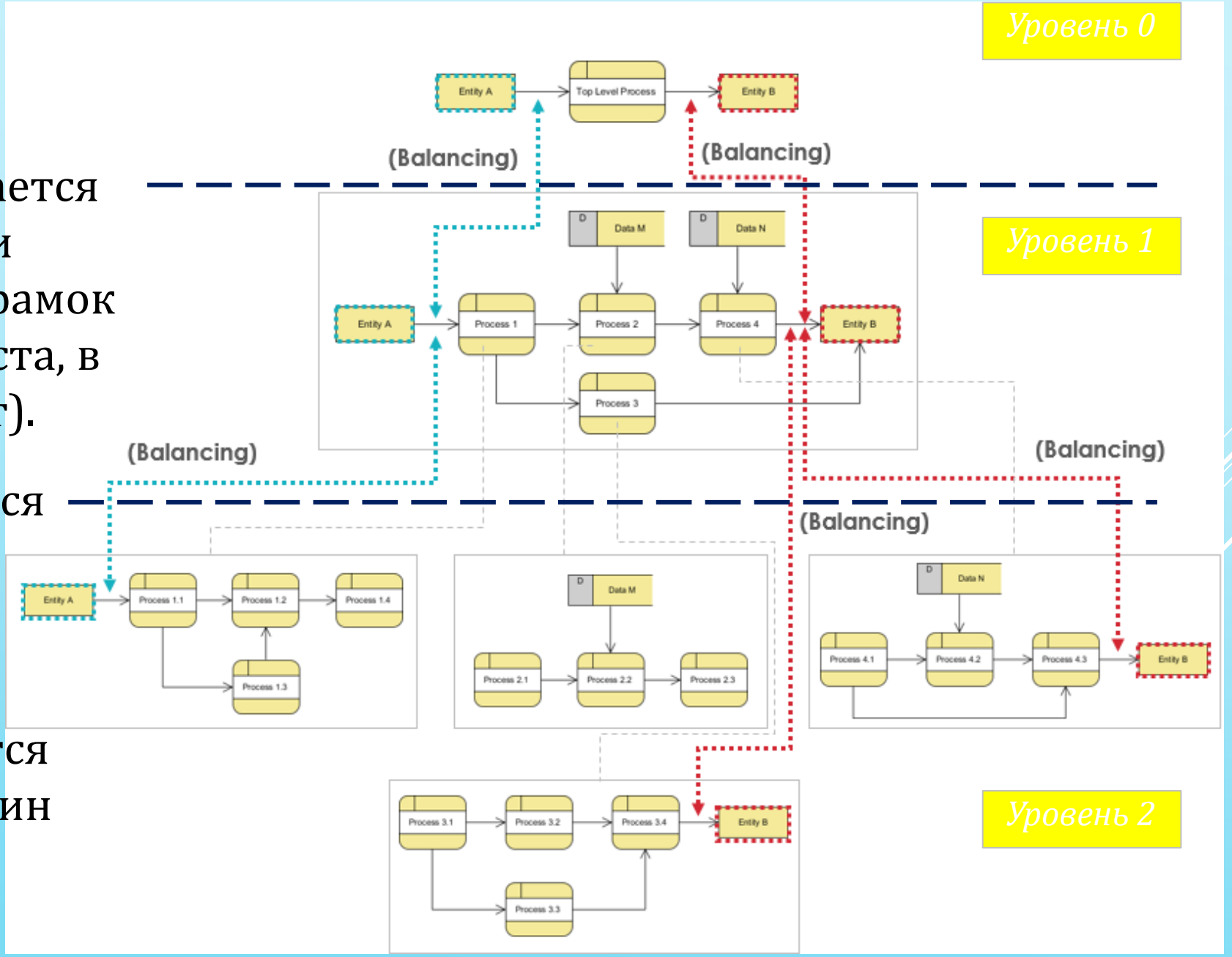


Поток идет одновременно в двух направлениях.

Хранилища связаны напрямую.

УРОВНИ DFD

- Первая диаграмма называется контекстной (уровень 0) и служит для обозначения рамок системы (задания контекста, в котором система работает).
- Диаграмма детализируется путем добавления новых уровней.
- На каждом новом уровне декомпозируется (строится отдельная диаграмма) один какой-то процесс с предыдущего уровня.



НУМЕРАЦИЯ УРОВНЕЙ

Как и в IDEF0 нумерация функциональных блоков имеет иерархических характер:

- Уровень 0 – 0
- Уровень 1 – 1, 2, 3, …
- Уровень 2 – 1.1, 1.2, …, 2.1, 2.2, …
- Уровень 3 – 1.1.1, 1.1.2, …, 1.2.1, 1.2.2, …

и т.д.

В отличии от IDEF0, рассматривающего систему как множество взаимопересекающихся действий, в названиях объектов DFD-диаграмм преобладают имена существительные. Контекстная DFD-диаграмма часто состоит из одного функционального блока и нескольких внешних сущностей. Функциональный блок на этой диаграмме обычно имеет имя, совпадающее с именем всей системы (рис.1). Добавление к диаграмме внешних ссылок не отменяет основного требования, что модель должна строиться с единственной точки зрения и иметь четко определенную цель и границы.

Рис.1

Функциональные блоки DFD моделируют некоторую функцию, которая преобразует сырье в какую-либо продукцию (или, в терминах JDEF, вход в выход). Хотя функциональные блоки DFD изображаются в виде прямоугольников с закругленными углами, они почти идентичны с функциональными блоками IDEF0 и действиями IDEF3. Как и действия IDEF3, функциональные блоки DFD имеют входы и выходы, рис. 2, однако не имеют управления и механизма исполнения, как IDEF0. В некоторых интерпретациях нотации DFD Гейна-Сарсона механизмы исполнения IDEF0 моделируются как ресурсы и изображаются в нижней части прямоугольника.

Рис.2

Внешние сущности обеспечивают необходимые входы для системы и/или являются приемниками для ее выходов. Одна внешняя сущность может одновременно предоставлять входы (функционируя как поставщик) и принимать выходы (функционируя как получатель). Внешние сущности изображаются как отбрасывающие тень прямоугольники (рис. 3) и обычно размещаются у краев диаграммы. Одна внешняя сущность может повторяться на одной и той же диаграмме несколько раз. Этот прием полезно применять для сокращения количества линий, соединяющих объекты на диаграмме.

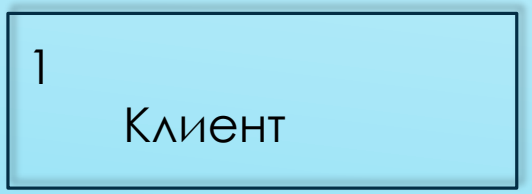


Рис.3 - Обозначение внешней сущности

Стрелки (потоки данных) описывают передвижение (потоки) объектов от одной части системы к другой. Поскольку все стороны обозначающего функциональный блок DFD прямоугольника равнозначны (в отличии от IDEF0), стрелки могут начинаться и заканчиваться в любой части блока. В DFD также используются двунаправленные стрелки, которые нужны для отображения взаимодействия между блоками (например, диалога типа «приказ – результат выполнения»).

На рис. 4 двунаправленная стрелка обозначает взаимный обмен информацией между департаментом маркетинга и рекламы и департаментом пластиковых карт.

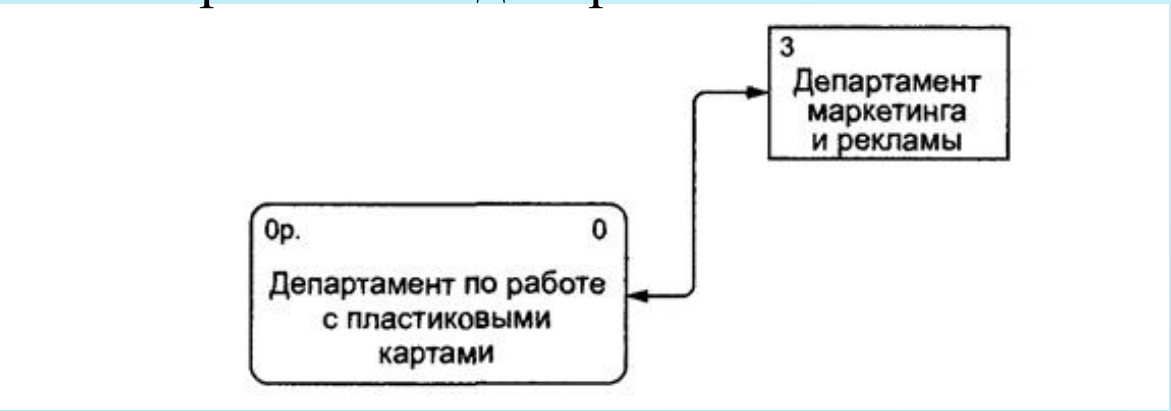


Рис. 4 - Двунаправленный поток между блоком и внешней сущностью

Ветвление и объединение. Стрелки на DFD-диаграммах могут быть разбиты (разветвлены) на части, и при этом каждый получившийся сегмент может быть переименован таким образом, чтобы показать декомпозицию данных, переносимых конкретным потоком (рис. 5).

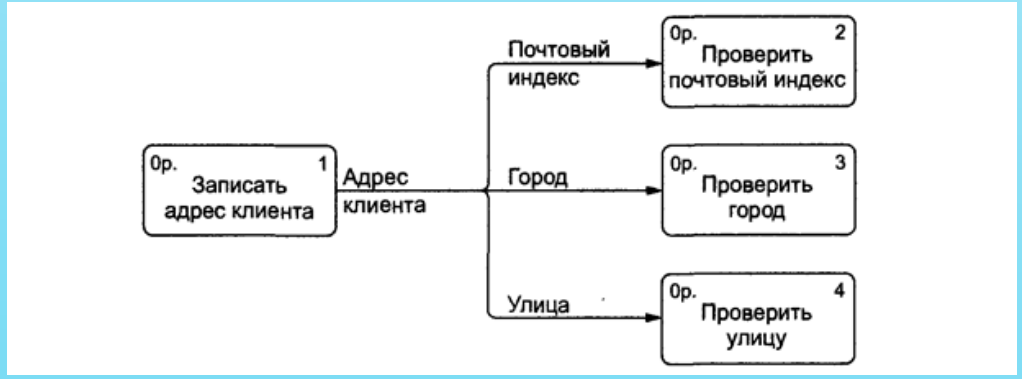


Рис. 5

Стрелки могут соединяться между собой (объединяться) для формирования так называемых комплексных объектов (рис.6).

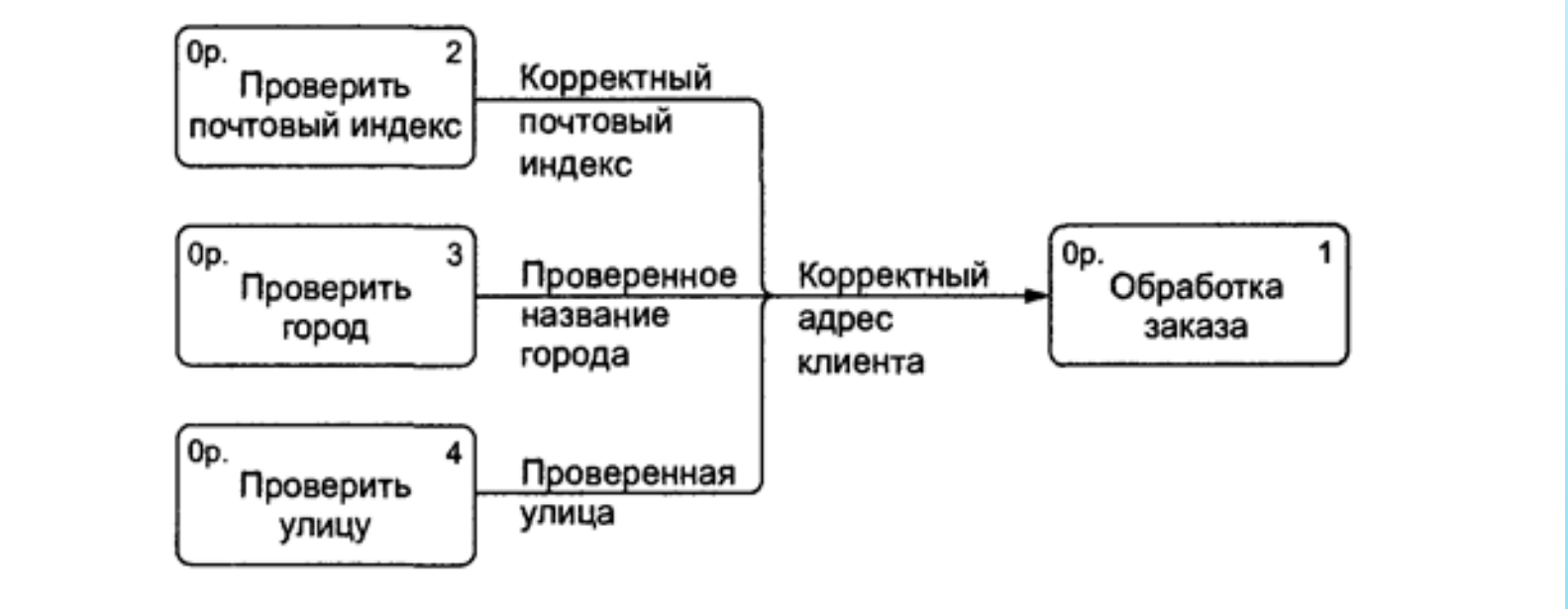


Рис. 6 - Объединение потоков в один

Нумерация объектов

Номер каждого функционального блока DFD может включать в себя префикс, номер родительской диаграммы и собственно номер объекта (рис. 7). Номер объекта уникальным образом идентифицирует функциональный блок на диаграмме. Номер родительской диаграммы и номер объекта в совокупности обеспечивают уникальную идентификацию каждого блока модели. Уникальные номера присваиваются также каждому хранилищу данных и каждой внешней сущности вне зависимости от расположения объекта на диаграмме. Каждый номер хранилища данных содержит префикс D (Data Store) и уникальный номер хранилища в модели (например, D3).

Аналогично, номер каждой внешней сущности содержит префикс E (External entity) и уникальный номер сущности в модели (например, E5).

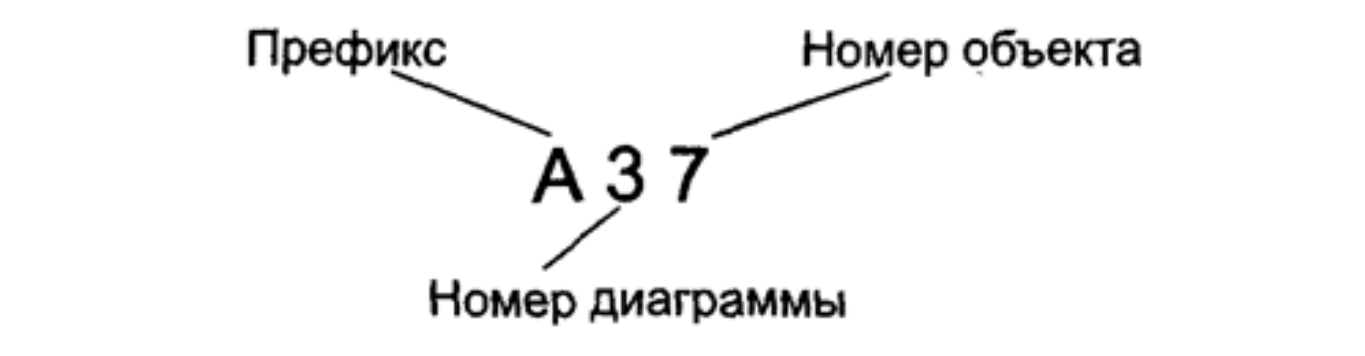


Рис. 7 - Компоненты номера функционального блока DFD

Задание:

1. Представьте процесс сдачи экзамена студентом преподавателю в виде нотации Йодана и Гейна-Сарсона

Рассмотрим процесс СДАЧА ЭКЗАМЕНА.

У нас есть две сущности СТУДЕНТ и ПРЕПОДАВАТЕЛЬ.

Описать графически потоки данных, которыми обменивается наша проектируемая система с внешними объектами.

Со стороны сущности СТУДЕНТ опишем информационные потоки:

Для сдачи экзамена необходимо, чтобы у СТУДЕНТА была ЗАЧЕТКА, а также, чтобы он имел ДОПУСК К ЭКЗАМЕНУ.

Результатом сдачи экзамена, т.е. выходными потоками будут ОЦЕНКА ЗА ЭКЗАМЕН и ЗАЧЕТКА, в которую будет проставлена ОЦЕНКА.

Со стороны сущности ПРЕПОДАВАТЕЛЬ информационные потоки следующие:

ЭКЗАМЕНАЦИОННАЯ ВЕДОМОСТЬ согласно которой будет известно, что СТУДЕНТ допущен до экзамена, а также официальный документ, куда будет занесен результат экзамена, т.е. ОЦЕНКА ЗА ЭКЗАМЕН, ПРОСТАВЛЕННАЯ В ВЕДОМОСТЬ.

Теперь детализируем процесс 1.СДАЧА ЭКЗАМЕНА. Этот процесс будет содержать следующие процессы:

Вытянуть билет

Подготовиться к ответу

Ответы на билет

Проставление оценки

Оформление : № группы, Ф.И.О. , номер, тема практического занятия, основной текст (структурированный, рисунки), выводы.

konst17@mail.ru