Лабораторная работа №1

Тема: построение блок-схем алгоритмов.

Цель работы: Получение практических навыков построения блок-схем с применением современных средств создания диаграмм.

Задание: Используя среду MS Visio, постройте блок-схему алгоритма процесса согласно полученному варианту, руководствуясь нормативными документами. (Используйте фигуры Visio Фигуры / Блок-схема / Фигуры простой блок-схемы).

Отчет по лабораторной работе должен содержать:

1. Фамилию и номер группы, задание
2. Описание алгоритма работы (текстовое)
3. Блок-схему алгоритма
4. Перечень других алгоритмов решения задачи (если они есть).

Основные теоретические сведения

Блок-схемы представляют собой совокупность действий или операций, изображенное в виде геометрических фигур. Переход от одного действия к другому обозначается направленной линией.

При составлении блок-схемы необходимо добавлять элементы сверху вниз последовательно друг за другом. При возникновении условий соблюдать древовидную иерархию. Блок-схема обязательно должна начинаться с элемента "Начало" и заканчиваться элементом "Конец", причем каждый из них должен быть употреблен только по одному разу.

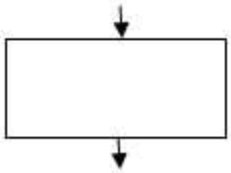
Составление блок-схем регламентируются следующими документами:

- *ГОСТ 19.701-90. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения.*
- *ГОСТ 19.002-80. Схемы алгоритмов и программ. Правила выполнения.*
- *ГОСТ 19.003-80. Схемы алгоритмов и программ. Обозначения условные графические*

Исходя из этих документов основными элементами блок-схемы являются:

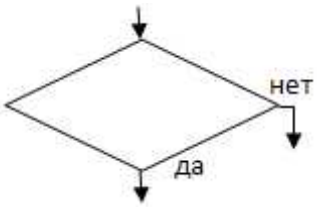
Терминатор

Обозначает начало или конец программы. Выделяет границы взаимодействия с внешней средой. Используется обычно с надписями "Начало","Конец" либо "Пуск","Остановка" строго по одному разу.



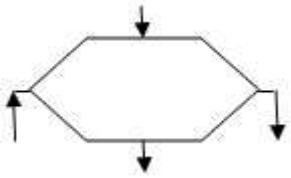
Процесс

Выполнение некоторой операции (арифметической, логической либо инойдругой), в результате которой каким-либо образом изменяются данные. Возможно объединение нескольких операций в один блок.



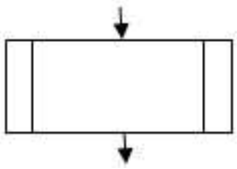
Решение

Выбор одного из двух возможных решений алгоритма. Внутри элемента расположено условие. Из углов ромба выходят возможные пути, обозначающиеся как "да","нет" либо "истина","ложь". В целях удобства чтения блок-схемы направление, отвечающее условию ("да"/"истина")выходит из нижнего угла ромба, противоположное из бокового. Возможно использования элемента для обозначения цикла repeat..until и while..do.



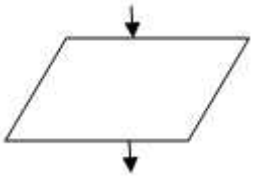
Модификация

Выполнение циклических команд for. Операции и действия цикла располагаются ниже элемента. При каждом шаге цикла программа возвращается к заголовку по левой стрелке. Выход из цикла производится по правой боковой стрелке.



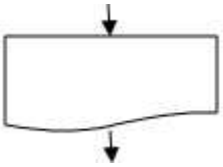
Предопределенный процесс

Обозначение процедуры, функции, модуля (части программы вне текущего последовательного кода).



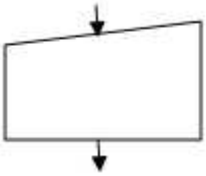
Данные

Осуществление обмена данными (ввод-вывод). Обобщенное представление обмена информацией без определенного типа носителя.



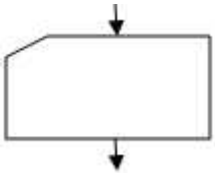
Документ

Вывод данных на бумажный носитель (печать на принтере).



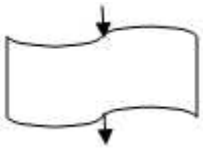
Ручной ввод

Неавтономный ввод данных с помощью клавиатруы.



Перфокарта

Ввод-вывод данных с перфокарты.

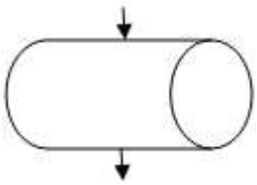


Перфолента

Ввод-вывод данных с перфоленты.

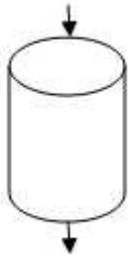
Запоминающее устройство с последовательным доступом

Обмен данными с магнитной лентой.



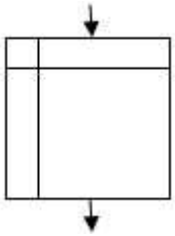
Запоминающее устройство с прямым доступом

Обмен данными с магнитным барабаном.



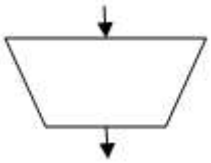
Магнитный диск

Ввод-вывод данных, носителем которых является магнитный диск.



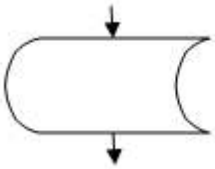
Оперативная память

Обмен данными с оперативно-запоминающим устройством (ОЗУ).



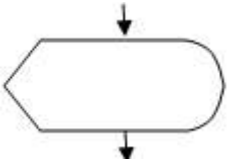
Ручное управление

Отображение процесса, выполняемого человеком.



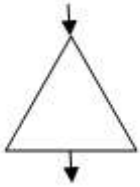
Сохраненные данные

Обмен данными при использовании запоминающего устройства, управляемого непосредственно процессором.



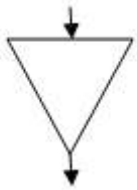
Дисплей

Отображение данных на мониторе, визуальных индикаторах.



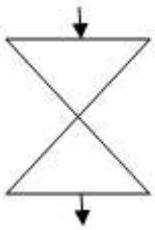
Извлечение

Выделение одного или несколько множеств из другого множества.



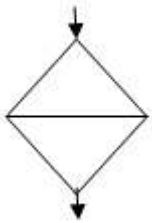
Слияние

Объединение одного или несколько множеств в общее множество.



Группировка

Объединение множеств с выделением некоторых других.



Сортировка

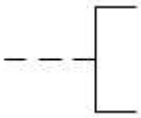
Упорядочивание множеств по заданному признаку.

Соединитель

Используется для обрыва линия связи в одном месте и продолжения в другом. Внутри элемента блок-схемы вводится уникальный идентификатор.

Межстраничный соединитель

Аналогичен предыдущему элементу блок-схемы, переносит линии связи с конца одной страницы в начало другой.



Комментарии

Пометка неактивной части программы.

Линия потока

Отображает пото данных, с возможным указанием

направления их передачи. Объединяет между собой элементы блок-схемы.

Пунктирная линия

Альтернативная связь между объектами. Используется также для обведения комментариев.

Параллельные действия

Синхронизация нескольких операций в программе единовременно.

Канал связи

Передача по каналам связи.

Пропуск

Пропуск элементов блок-схемы. Используется когда можно оставить часть программы без внимания.

Как правило, при составлении блок-схем используются только часть этих элементов, пренебрегая такими элементами как "ручной ввод", "дисплей" и им подобные, используя наиболее общее обозначение "данные".

Варианты задания

1. Передача данных в локальной сети, построенной по технологии Token Ring.
2. Передача данных в локальной сети, построенной по технологии Ethernet.
3. Передача данных в интернет с использованием технологии GPRS.
4. Передача данных в интернет с использованием технологии Wi-Fi.
5. Соединение абонентов телефонной сети (проводной).
6. Соединение абонентов мобильной связи.
7. Радиовещание.
8. Телевещание.
9. Факсимильная связь.
10. Телеграф.

Лабораторная работа №2

Тема: построение модели данных.

Цель работы: Получение практических навыков построения моделей данных с применением современных средств создания диаграмм.

Задание: Используя среду MS Visio, постройте ER-модель согласно полученному варианту.

Отчет по лабораторной работе должен содержать:

1. Фамилию и номер группы, задание
2. Описание данных, для которых строится модель (текстовое)
3. ERD
4. Примеры данных (не менее 10 строк для каждой сущности).

Основные теоретические сведения

Основные понятия модели Entity-Relationship (Сущность-Связи)

На использовании разновидностей ER-модели основано большинство современных подходов к проектированию баз данных (главным образом, реляционных). Модель была предложена Ченом (Chen) в 1976 г. Моделирование предметной области базируется на использовании графических диаграмм, включающих небольшое число разнородных компонентов. В связи с наглядностью представления концептуальных схем баз данных ER-модели получили широкое распространение в системах CASE, поддерживающих автоматизированное проектирование реляционных баз данных.

Основными понятиями ER-модели являются сущность, связь и атрибут.

Сущность - это реальный или представляемый объект, информация о котором должна сохраняться и быть доступна. В диаграммах ER-модели сущность представляется в виде прямоугольника, содержащего имя сущности. При этом имя сущности - это имя типа, а не некоторого конкретного экземпляра этого типа. Для большей выразительности и лучшего понимания имя сущности может сопровождаться примерами конкретных объектов этого типа.

Ниже изображена сущность АЭРОПОРТ с примерными объектами Шереметьево и Хитроу:

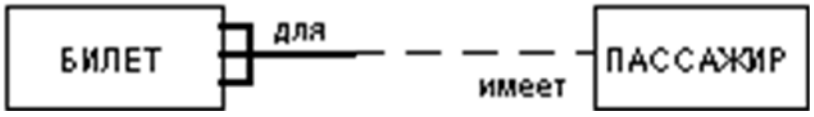
Каждый экземпляр сущности должен быть отличим от любого другого экземпляра той же сущности (это требование в некотором роде аналогично требованию отсутствия кортежей-дубликатов в реляционных таблицах).

Связь - это графически изображаемая ассоциация, устанавливаемая между двумя сущностями. Эта ассоциация всегда является бинарной и может существовать между двумя разными сущностями или между сущностью и ей же самой (рекурсивная связь). В любой связи выделяются два конца (в соответствии с существующей парой связываемых сущностей), на каждом из которых указывается имя конца связи, степень конца связи (сколько экземпляров данной сущности связывается), обязательность связи (т.е. любой ли экземпляр данной сущности должен участвовать в данной связи).

Связь представляется в виде линии, связывающей две сущности или ведущей от сущности к ней же самой. При это в месте "стыковки" связи с сущностью используются трехточечный вход в прямоугольник сущности, если для этой сущности в связи могут использоваться много (many) экземпляров сущности, и одноточечный вход, если в связи может участвовать только один экземпляр сущности. Обязательный конец связи изображается сплошной линией, а необязательный - прерывистой линией.

Как и сущность, связь - это типовое понятие, все экземпляры обеих пар связываемых сущностей подчиняются правилам связывания.

В изображенном ниже примере связь между сущностями БИЛЕТ и ПАССАЖИР связывает билеты и пассажиров. При том конец сущности с именем "для" позволяет связывать с одним пассажиром более одного билета, причем каждый билет должен быть связан с каким-либо пассажиром. Конец сущности с именем "имеет" означает, что каждый билет может принадлежать только одному пассажиру, причем пассажир не обязан иметь хотя бы один билет.

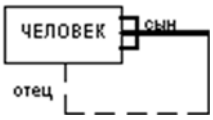


Лаконичной устной трактовкой изображенной диаграммы является следующая:

- Каждый БИЛЕТ предназначен для одного и только одного ПАССАЖИРА;
- Каждый ПАССАЖИР может иметь один или более БИЛЕТОВ.

На следующем примере изображена рекурсивная связь, связывающая сущность ЧЕЛОВЕК с ней же самой. Конец связи с именем "сын" определяет тот факт, что у одного

отца может быть более чем один сын. Конец связи с именем "отец" означает, что не у каждого человека могут быть сыновья.



Лаконичной устной трактовкой изображенной диаграммы является следующая:

- Каждый ЧЕЛОВЕК является сыном одного и только одного ЧЕЛОВЕКА;
- Каждый ЧЕЛОВЕК может являться отцом для одного или более ЛЮДЕЙ ("ЧЕЛОВЕКОВ").

Атрибутом сущности является любая деталь, которая служит для уточнения, идентификации, классификации, числовой характеристики или выражения состояния сущности. Имена атрибутов заносятся в прямоугольник, изображающий сущность, под именем сущности и изображаются малыми буквами, возможно, с примерами.

Пример: Уникальным идентификатором сущности является атрибут, комбинация атрибутов, комбинация связей или комбинация связей и атрибутов, уникально отличающая любой экземпляр сущности от других экземпляров сущности того же типа.

Нормальные формы ER-схем

Как и в реляционных схемах баз данных, в ER-схемах вводится понятие нормальных форм, причем их смысл очень близко соответствует смыслу реляционных нормальных форм. Заметим, что формулировки нормальных форм ER-схем делают более понятным смысл нормализации реляционных схем. Мы приведем только очень краткие и неформальные определения трех первых нормальных форм.

В первой нормальной форме ER-схемы устраняются повторяющиеся атрибуты или группы атрибутов, т.е. производится выявление неявных сущностей, "замаскиро-ванных" под атрибуты.

Во второй нормальной форме устраняются атрибуты, зависящие только от части уникального идентификатора. Эта часть уникального идентификатора определяет отдельную сущность.

В третьей нормальной форме устраняются атрибуты, зависящие от атрибутов, не входящих в уникальный идентификатор. Эти атрибуты являются основой отдельной сущности.

Более сложные элементы ER-модели

Мы остановились только на самых основных и наиболее очевидных понятиях ER-модели данных. К числу более сложных элементов модели относятся следующие:

- Подтипы и супертипы сущностей. Как в языках программирования с развитыми типовыми системами (например, в языках объектно-ориентированного программирования), вводится возможность наследования типа сущности, исходя из

одного или нескольких супертипов. Интересные нюансы связаны с необходимостью графического изображения этого механизма.

- Связи "many-to-many". Иногда бывает необходимо связывать сущности таким образом, что с обоих концов связи могут присутствовать несколько экземпляров сущности (например, все члены кооператива сообща владеют имуществом кооператива). Для этого вводится разновидность связи "многие-со-многими".
- Уточняемые степени связи. Иногда бывает полезно определить возможное количество экземпляров сущности, участвующих в данной связи (например, служащему разрешается участвовать не более, чем в трех проектах одновременно). Для выражения этого семантического ограничения разрешается указывать на конце связи ее максимальную или обязательную степень.
- Каскадные удаления экземпляров сущностей. Некоторые связи бывают настолько сильными (конечно, в случае связи "один-ко-многим"), что при удалении опорного экземпляра сущности (соответствующего концу связи "один") нужно удалить и все экземпляры сущности, соответствующие концу связи "многие". Соответствующее требование "каскадного удаления" можно сформулировать при определении сущности.
- Домены. Как и в случае реляционной модели данных бывает полезна возможность определения потенциально допустимого множества значений атрибута сущности (домена).

Эти и другие более сложные элементы модели данных "Сущность-Связи" делают ее существенно более мощной, но одновременно несколько усложняют ее использование. Конечно, при реальном использовании ER-диаграмм для проектирования баз данных необходимо ознакомиться со всеми возможностями.

В нашей лекции мы немного подробнее разберем только один из упомянутых элементов - подтип сущности.

Сущность может быть расщеплена на два или более взаимно исключающих подтипа, каждый из которых включает общие атрибуты и/или связи. Эти общие атрибуты и/или связи явно определяются один раз на более высоком уровне. В подтипах могут определяться собственные атрибуты и/или связи. В принципе подтипизация может продолжаться на более низких уровнях, но опыт показывает, что в большинстве случаев оказывается достаточно двух-трех уровней.

Сущность, на основе которой определяются подтипы, называется супертипом. Подтипы должны образовывать полное множество, т.е. любой экземпляр супертипа

должен относиться к некоторому подтипу. Иногда для полноты приходится определять дополнительный подтип ПРОЧИЕ.

Пример: Супертип ЛЕТАТЕЛЬНЫЙ АППАРАТ

Как полагается это читать? От супертипа: ЛЕТАТЕЛЬНЫЙ АППАРАТ, который должен быть АЭРОПЛАНОМ, ВЕРТОЛЕТОМ, ПТИЦЕЛЕТОМ или ДРУГИМ ЛЕТАТЕЛЬНЫМ АППАРАТОМ. От подтипа: ВЕРТОЛЕТ, который относится к типу ЛЕТАТЕЛЬНОГО АППАРАТА. От подтипа, который является одновременно супертипа: АЭРОПЛАН, который относится к типу ЛЕТАТЕЛЬНОГО АППАРАТА и должен быть ПЛАНЕРОМ или МОТОРНЫМ САМОЛЕТОМ.

Иногда удобно иметь два или более разных разбиения сущности на подтипы. Например, сущность ЧЕЛОВЕК может быть разбита на подтипы по профессиональному признаку (ПРОГРАММИСТ, ДОЯРКА и т.д.), а может - по половому признаку (МУЖЧИНА, ЖЕНЩИНА).

Получение реляционной схемы из ER-схемы

Шаг 1. Каждая простая сущность превращается в таблицу. Простая сущность - сущность, не являющаяся подтипом и не имеющая подтипов. Имя сущности становится именем таблицы.

Шаг 2. Каждый атрибут становится возможным столбцом с тем же именем; может выбираться более точный формат. Столбцы, соответствующие необязательным атрибутам, могут содержать неопределенные значения; столбцы, соответствующие обязательным атрибутам, - не могут.

Шаг 3. Компоненты уникального идентификатора сущности превращаются в первичный ключ таблицы. Если имеется несколько возможных уникальных идентификатора, выбирается наиболее используемый. Если в состав уникального идентификатора входят связи, к числу столбцов первичного ключа добавляется копия

уникального идентификатора сущности, находящейся на дальнем конце связи (этот процесс может продолжаться рекурсивно). Для именования этих столбцов используются имена концов связей и/или имена сущностей.

Шаг 4. Связи многие-к-одному (и один-к-одному) становятся внешними ключами. Т.е. делается копия уникального идентификатора с конца связи "один", и соответствующие столбцы составляют внешний ключ. Необязательные связи соответствуют столбцам, допускающим неопределенные значения; обязательные связи - столбцам, не допускающим неопределенные значения.

Шаг 5. Индексы создаются для первичного ключа (уникальный индекс), внешних ключей и тех атрибутов, на которых предполагается в основном базировать запросы.

Шаг 6. Если в концептуальной схеме присутствовали подтипы, то возможны два способа:

- все подтипы в одной таблице (а)
- для каждого подтипа - отдельная таблица (б)

При применении способа (а) таблица создается для наиболее внешнего супертипа, а для подтипов могут создаваться представления. В таблицу добавляется по крайней мере один столбец, содержащий код ТИПА; он становится частью первичного ключа.

При использовании метода (б) для каждого подтипа первого уровня (для более нижних - представления) супертип воссоздается с помощью представления UNION (из всех таблиц подтипов выбираются общие столбцы - столбцы супертипа).

Все в одной таблице	**Таблица - на подтип**
Преимущества	
Все хранится вместе. Легкий доступ к супертипу и подтипам. Требуется меньше таблиц	Более ясны правила подтипов. Программы работают только с нужными таблицами
Недостатки	
Слишком общее решение. Требуется дополнительная логика работы с разными наборами столбцов и разными ограничениями. Потенциальное узкое место (в связи с блокировками). Столбцы подтипов должны быть необязательными. В некоторых СУБД для хранения неопределенных значений требуется дополнительная память	Слишком много таблиц Смущающие столбцы в представлении UNION. Потенциальная потеря производительности при работе через UNION. Над супертипом невозможны модификации

Шаг 7. Имеется два способа работы при наличии исключающих связей:

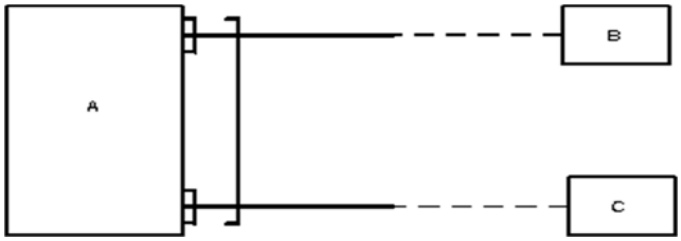
- общий домен (а)
- явные внешние ключи (б)

Если остающиеся внешние ключи все в одном домене, т.е. имеют общий формат (способ (а)), то создаются два столбца: идентификатор связи и идентификатор сущности. Столбец идентификатора связи используется для различения связей, покрываемых дугой исключения. Столбец идентификатора сущности используется для хранения значений уникального идентификатора сущности на дальнем конце соответствующей связи.

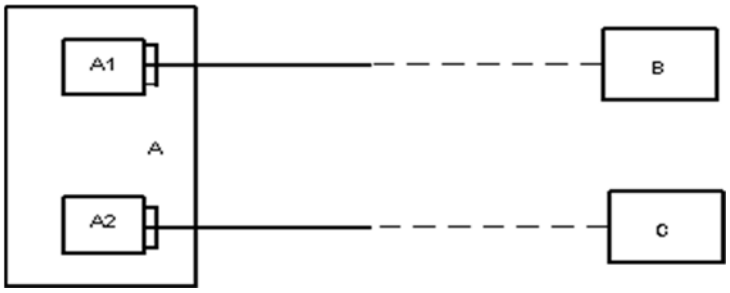
Если результирующие внешние ключи не относятся к одному домену, то для каждой связи, покрываемой дугой исключения, создаются явные столбцы внешних ключей; все эти столбцы могут содержать неопределенные значения.

Общий домен	**Явные внешние ключи**
Преимущества	
Нужно только два столбца	Условия соединения - явные
Недостатки	
Оба дополнительных атрибута должны использоваться в соединениях	Слишком много столбцов

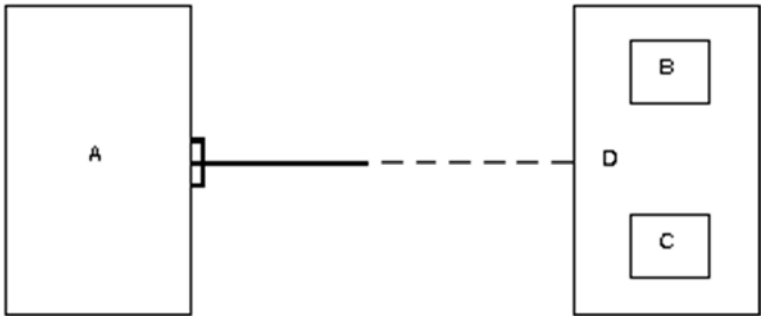
Альтернативные модели сущностей:



Вариант 1 (плохой)



Вариант 2 (существенно лучше, если подтипы действительно существуют)/ Вариант 3 (годится при наличии осмысленного супертипа D).



Варианты задания

1. Данный об абонентах сотовой связи.
2. Данный об абонентах телефонной компании.
3. Данный об абонентах кабельной компании.
4. Данные об аппаратной платформе локальных сетей компании.
5. Данные мониторинга локальной сети.
6. Данные о фирмах, предоставляющих коммуникационные услуги в регионе.
7. Данные о линиях связи в районе.
8. Данные о сетевом оборудовании.
9. Данные о стандартах, протоколах, нормативных документах, относящимся к телекоммуникационной отрасли.
10. Данные о программной платформе коммуникационных средств.

Лабораторная работа №3

Тема: моделирование потоков данных (процессов).

Цель работы: Получение практических навыков построения моделей потоков данных с применением современных средств создания диаграмм.

Задание: Используя среду MS Visio, постройте диаграмму потоков данных согласно полученному варианту. (Используйте Фигуры / Программное обеспечение и базы данных / Программное обеспечение / Гейн-Сарсон).

Отчет по лабораторной работе должен содержать:

1. Фамилию и номер группы, задание
2. Описание данных, для которых строится модель (текстовое)
3. Диаграмму потоков данных
4. Пример данных для модели (не менее пяти строк для каждой сущности).

Основные теоретические сведения

В основе данной методологии (методологии Gane/Sarson) лежит построение модели анализируемой ИС - проектируемой или реально существующей. В соответствии с методологией модель системы определяется как иерархия диаграмм потоков данных (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 становятся элементарными и детализировать их далее невозможно.

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;
- накопители 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. 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.

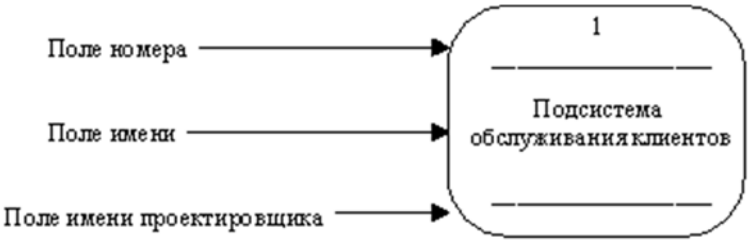
Внешняя сущность обозначается квадратом, расположенным как бы "над" диаграммой и бросающим на нее тень, для того, чтобы можно было выделить этот символ среди других обозначений:

Внешняя сущность

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.

Подсистема (или система) на контекстной диаграмме изображается следующим образом.



Подсистема

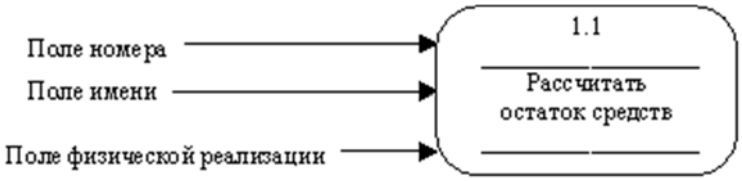
Номер подсистемы служит для ее идентификации. В поле имени вводится наименование подсистемы в виде предложения с подлежащим и соответствующими определениями и дополнениями.

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.

Процесс на диаграмме потоков данных изображается, как показано на рисунке 2.15.



Процесс

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, например:

- "Ввести сведения о клиентах";
- "Выдать информацию о текущих расходах";
- "Проверить кредитоспособность клиента".

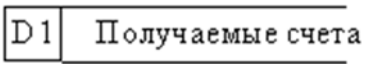
Использование таких глаголов, как "обработать", "модернизировать" или "отредактировать" означает, как правило, недостаточно глубокое понимание данного процесса и требует дальнейшего анализа.

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

Накопители 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 магнитном носителе и т.д. Накопитель данных на диаграмме потоков данных изображается, как показано на рисунке.



Накопитель данных

Накопитель данных идентифицируется буквой "D" и произвольным числом. Имя накопителя выбирается из соображения наибольшей информативности для проектировщика.

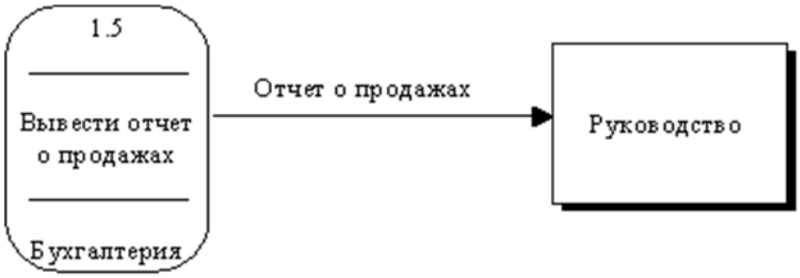
Накопитель данных в общем случае является прообразом будущей базы данных и описание хранящихся в нем данных должно быть увязано с информационной моделью.

Потоки 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 лентами или дискетами, переносимыми с одного компьютера на другой и т.д.

Поток данных на диаграмме изображается линией, оканчивающейся стрелкой, которая показывает направление потока. Каждый поток данных имеет имя, отражающее его содержание.



Поток данных

Построение иерархии диаграмм потоков данных

Первым шагом при построении иерархии ДПД является построение контекстных диаграмм. 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.

Иерархия контекстных диаграмм определяет взаимодействие основных функциональных подсистем проектируемой ИС как между собой, так и с внешними входными и выходными потоками данных и внешними объектами (источниками и приемниками информации), с которыми взаимодействует ИС.

Разработка контекстных диаграмм решает проблему строгого определения функциональной структуры ИС на самой ранней стадии ее проектирования, что особенно важно

для сложных многофункциональных систем, в разработке которых участвуют разные организации и коллективы разработчиков.

После построения контекстных диаграмм полученную модель следует проверить на полноту исходных данных об объектах системы и изолированность объектов (отсутствие информационных связей с другими объектами).

Для каждой подсистемы, присутствующей на контекстных диаграммах, выполняется ее детализация при помощи ДПД. Каждый процесс на ДПД, в свою очередь, может быть детализирован при помощи ДПД 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 12, получают номера 12.1, 12.2, 12.3 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.

Миниспецификация является конечной вершиной иерархии ДПД. 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).

При построении иерархии ДПД переходить к детализации процессов следует только после определения содержания всех потоков и накопителей данных, которое описывается при помощи структур данных. Структуры данных конструируются из элементов данных и могут содержать альтернативы, условные вхождения и итерации. Условное вхождение означает, что данный компонент может отсутствовать в структуре. Альтернатива означает, что в структуру может входить один из перечисленных элементов. Итерация означает вхождение любого числа элементов в указанном диапазоне. Для каждого элемента данных может указываться его тип

(непрерывные или дискретные данные). Для непрерывных данных может указываться единица измерения (кг, см и т.п.), диапазон значений, точность представления и форма физического кодирования. Для дискретных данных может указываться таблица допустимых значений.

После построения законченной модели системы ее необходимо верифицировать (проверить на полноту и согласованность). В полной модели все ее объекты (подсистемы, процессы, потоки данных) должны быть подробно описаны и детализированы. Выявленные недетализированные объекты следует детализировать, вернувшись на предыдущие шаги разработки. В согласованной модели для всех потоков данных и накопителей данных должно выполняться правило сохранения информации: все поступающие куда-либо данные должны быть считаны, а все считываемые данные должны быть записаны.

Варианты задания

1. Функционирование локальной сети (технология Token Ring).
2. Функционирование локальной сети (технология Ethernet).
3. Функционирование аппарата сотовой связи.
4. Функционирование цифровой АТС.
5. Функционирование маршрутизатора.
6. Функционирование сетевого коммутатора (switch).
7. Радиовещание.
8. Телевещание.
9. Факсимильная связь.
10. Телеграф.

Лабораторная работа №4

Тема: универсальный язык моделирования.

Цель работы: Получение практических навыков использования UML.

Задание: Используя среду MS Visio, постройте диаграммы UML согласно полученному варианту. (Используйте Фигуры / Программное обеспечение и базы данных / Программное обеспечение / соответствующая диаграмма).

Отчет по лабораторной работе должен содержать:

1. Фамилию и номер группы, задание
2. Описание данных, для которых строится модель (текстовое)
3. Диаграммы UML.

В качестве учебно-методического пособия используйте учебник М. Фаулера «Основы UML».

Варианты задания

1. Работа мобильного телефона.
2. Работа проводного телефона.
3. Работа цифровой АТС.
4. Работа по маршрутизатора.
5. Работа банкомата.
6. Работа сетевого коммутатора (switch).
7. Работа радиоприемника.
8. Работа телевизора.
9. Работа рации.
10. Работа спутникового приемника

Лабораторная работа №5

Тема: построение схемы сети.

Цель работы: Получение практических навыков построения диаграмм и схем сетей с применением современных средств создания диаграмм.

Задание: Используя среду MS Visio, постройте схему локальной сети УлГУ (с учетом всех корпусов, используйте Фигуры / Сеть / Сеть). Работа выполняется группой.

Отчет по лабораторной работе должен содержать:

1. Фамилии студентов и номер группы, задание
2. Описание данных, для которых строится модель (текстовое)
3. Схему сети.

Лабораторная работа №6

Тема: знакомство со средой имитационного моделирования AnyLogic.

Цель работы: Получение практических навыков построения имитационных моделей с применением современных средств создания диаграмм.

Задание: Используя среду AnyLogic, выполните предложенные обучающие задания.

В качестве учебно-методического материала, используйте руководство пользователя, введение в моделирование в среде AnyLogic и описание библиотек среды.

Лабораторная работа №7

Тема: моделирование систем массового обслуживания.

Цель работы: Получение практических навыков построения моделей потоков данных с применением современных средств создания диаграмм.

Задание: Используя среду AnyLogic, выполните обучающее задание №1 и постройте систему массового обслуживания согласно полученному варианту.

Отчет по лабораторной работе должен содержать:

1. Фамилию и номер группы, задание
2. Описание данных, для которых строится модель (текстовое)
3. Модель
4. Результаты исследования модели.

Варианты задания

1. «Аэропорт».
2. «Железная дорога».
3. «Торговый центр».
4. «Автозаправка».
5. «Интернет-кафе».
6. «Больница».
7. «Прокат автомобилей».
8. «Парикмахерская».
9. «Парк аттракционов».
10. «Автомастерская»