Кафедра прикладной математики и информационных технологий

Дисциплина «Архитектура ЭВМ и вычислительных сетей » специальность «Системный анализ и управление» – «220100.62»

Тема № 4 «Основы сетевой архитектуры»

Занятие № 4.2 «Модели сетевого взаимодействия»

Учебные цели

Дидактические:
- уяснить назначение и функции уровней эталонной модели.
Учебные:
- изучить общую характеристику эталонной модели открытых систем.
- изучить схему упаковки-распаковки данных.
Воспитательные:
- воспитывать у обучающихся стремление к углубленному изучению учебного материала.
- воспитывать у обучающихся стремление к самостоятельному изучению новых информационных технологий.

Содержание и порядок проведения занятия	Время, мин
ВОДНАЯ ЧАСТЬ: принять доклад о готовности группы к занятиям; проверить наличие обучающихся, внешний вид, готовность к занятиям. Объявить тему, цели и учебные вопросы занятия. Подчеркнуть актуальность и значимость темы для дальнейшей практической деятельности обучающихся.	5
ОСНОВНАЯ ЧАСТЬ	80
Учебные вопросы:	
. Общая характеристика эталонной модели открытых систем	15
. Назначение и функции уровней модели	15
. Схема упаковки-распаковки данных	20
. Модель «клиент-сервер» и ее модификации	10
. Сети и протоколы передачи информации	20
ЗАКЛЮЧИТЕЛЬНАЯ ЧАСТЬ: напомнить тему, цели занятий, ответить на поставленные вопросы, дать задание на самостоятельную работу, самоподготовку.	5

Литература

Основная:

.Информатика. Базовый курс. Учебник для вузов. Под ред. Симоновича; СПб.: Питер, -2014 г.
.Бройдо В.Л., Ильина О.П. Архитектура ЭВМ и систем: Учебник для вузов. – СПб.: Питер, 2009 г.
.Бройдо В. Л., Ильина О. П. Вычислительные системы, сети и коммуникации. Изд.: Питер, - 2011 г.
.Операционные системы. С. В. Синицын, А. В. Батаев. Учебник, изд. Академия, 2012 г.
.Операционные системы. Концепции построения и обеспечения безопасности. Ю. Ф. Мартемьянов, изд. Москва 2011 г.

Дополнительная:

.Максимов Н.В., Партыка Т.Л., Попов И.И. Архитектура ЭВМ и вычислительных систем: Учебник. – М.: ФОРУМ, 2006 г.
.Э. Таненбаум. Архитектура компьютера 4- е изд. Питер, 2005 г.

1. Общая характеристика эталонной модели открытых систем

Возникает потребность в упорядочении всех функций, связанных с передачей данных от одного компьютера к другому.

Регламентация функций передачи данных позволит упорядочить процесс разработки новых аппаратных и программных средств автоматизации. В этом случае удовлетворяется требование открытости сети, т.е. возможности подключения дополнительных элементов без изменения существующих. Помимо этого обеспечивается повышенная структурная и функциональная гибкость.

Обмен данными между программами различных компьютеров можно рассматривать как обмен между процессами, реализуемыми этими машинами.

Стремление к типизации побудило Международную организацию стандартов – (*International Standard Organization – ISO*) рекомендовать Эталонную модель открытых систем – ЭМОС (*Open System Interconnection – OSI*).

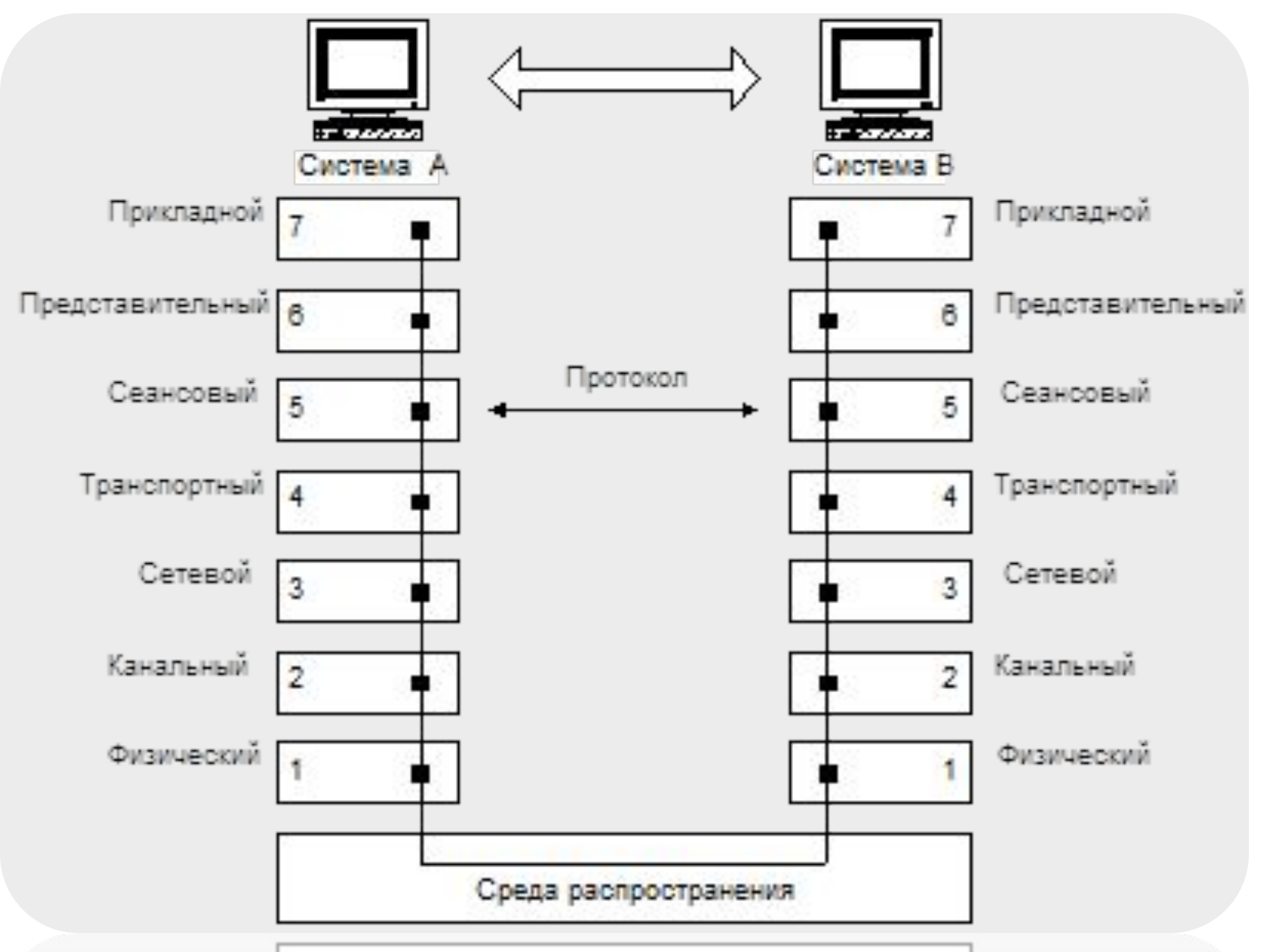
Общая характеристика модели. Основным понятием ЭМОС является *система*. Под системой здесь понимают иерархическую совокупность функций, реализуемых одной или несколькими ЭВМ, и предназначенную для выполнения предписанных ей в сети задач.

Каждая система является ***открытой***. Это означает, что независимо от особенностей аппаратной или программной реализации названные системы могут взаимодействовать друг с другом.

Система представляется многоуровневой иерархической структурой. На каждом уровне решается отдельная сетевая задача, обеспечивающая *сервис* вышестоящему уровню. Порождаемые этими задачами процессы, а также средства их решения объединяют понятием *объекты*. Все объекты приписаны соответствующим уровням.

Регламентации в ЭМОС подвергается порядок обмена данными между объектами. Взаимодействовать могут: объекты смежных уровней одной системы, одноуровневые объекты различных систем. В первом случае совокупность регламентирующих правил обмена данными называется *межуровневым интерфейсом*. Во втором случае эти правила носят название *протокола*.

Правила взаимодействия в ЭМОС допускают обмен данными между объектами одного уровня только через объекты смежного нижнего уровня. Последовательно используя названное ограничение к каждому из уровней, можно заметить, что маршрут обмена данными между объектами различных систем всегда проходит через самый нижний уровень. При этом среда распространения обеспечивает физическую связь между процессами.


Система A
Система B
Прикладной
7
Представительный
6
Сеансовый
5
Транспортный
4
Сетевой
3
Канальный
2
Физический
1
Протокол
Среда распространения

Два верхних уровня соответствуют прикладным процессам. Эти процессы выполняются в интересах пользователей и задаются их программами. Нижние 5 уровней определяют сетевой метод доступа. Точка раздела этих групп уровней называется портом. Через порты по логическим каналам осуществляется связь различных процессов. Процесс может быть одно- или многопортовым.

2. Назначение и функции уровней модели

Назначение ***прикладного*** уровня – выполнение любого информационно-расчетного процесса.

Основная задача ***представительного*** уровня – реализация процедур предоставления данных прикладным процессам. Этот уровень ответственен за преобразование данных, то есть он определяет их форматы, коды и структуры при выдаче их на сеансовый уровень и обратно – в сторону прикладного процесса. Также представительный уровень реализует следующие функции:

адресацию прикладных процессов;

сжатие и расширение данных;

сегментацию и объединение данных;

буферизацию данных.

Взаимодействие процессов представления организуется с помощью нижестоящего сеансового уровня.

Сеансовый уровень обеспечивает передачу данных по логическим каналам. Эта передача организуется в виде отдельных сеансов.

Транспортный уровень. Реализует подготовку данных к виду, пригодному для передачи методом коммутации пакетов.

Три нижних уровня реализуют функции сети передачи данных, обеспечивающей передачу данных между абонентами.

Сетевой уровень. Предназначен для выполнения процедур маршрутизации, то есть выбора маршрута передачи данных по линиям, связывающим узлы сети передачи данных.

Канальный уровень. Обеспечивает управление каналами передачи данных. Одна из основных функций этого уровня – повышение достоверности передачи данных.

Физический уровень. Реализует функции по управлению каналом связи, такие как установление, поддержание и разрушение физических соединений.

Как правило, функции физического и сетевого уровней реализуются в основном техническими средствами. На первом используются электронные схемы, на втором – контроллеры. На остальных уровнях используются программные средства, образующие сетевое программное обеспечение компьютеров.

.

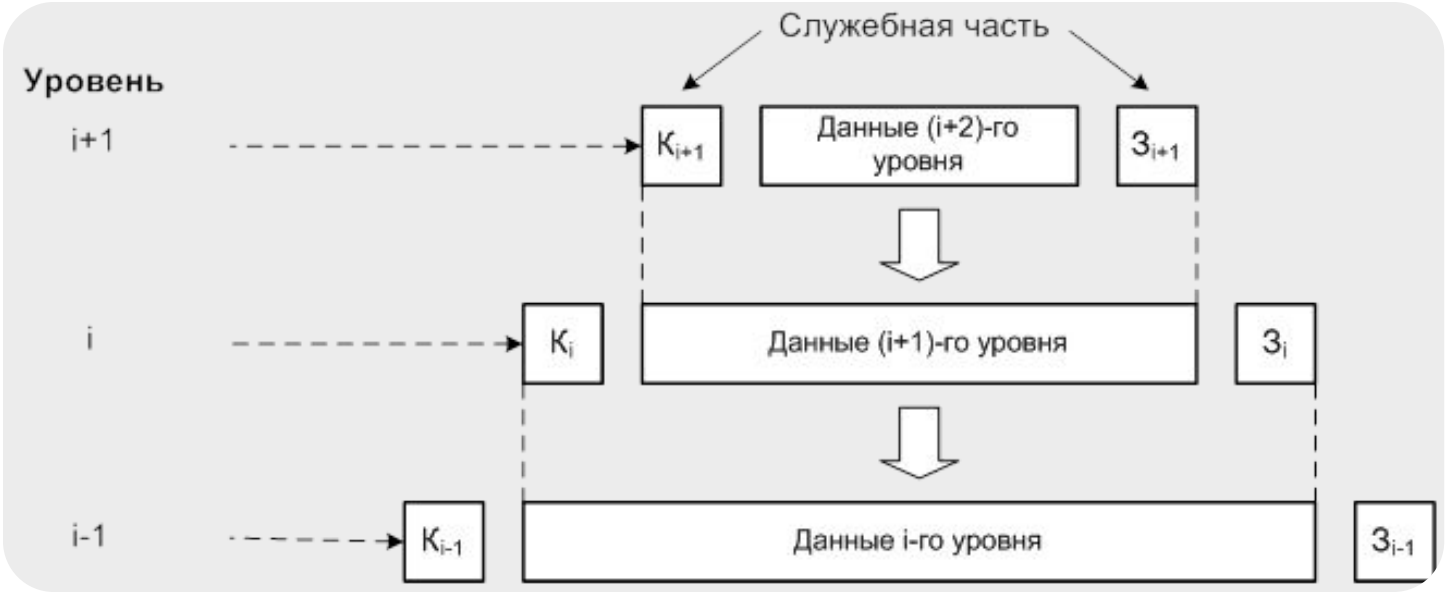
3. Схема упаковки-распаковки данных

Последовательное подключение различных уровней к процедуре передачи данных, а также требование независимости межуровневых интерфейсов имеет своим следствием тот факт, что данные, поступающие с более высокого смежного уровня, должны сопровождаться некоторой служебной информацией, которая только и подлежит декодированию на данном уровне. Вся остальная часть передаваемой информации для данного уровня должна быть «прозрачной», то есть бессодержательной.

Такая последовательная процедура добавления служебной информации к передаваемым данным получила наименование *упаковки* данных. На приемной стороне выполняется обратная процедура, состоящая в последовательном снятии служебной информации и называемая *распаковкой* или *вскрытием* данных.

На практике используют два способа работы со служебной информацией: **асинхронный и синхронный**.

При использовании ***асинхронного*** способа служебная часть информации представляется совокупностью двух блоков – *заголовка* (З) и *концевика* (К), размещаемых соответственно в начале и конце данных, сформированных смежным, старшим уровнем. Служебные блоки снабжаются специальными признаками, позволяющими выделить их среди других блоков.



Синхронная идентификация границ передаваемых данных позволяет упростить структуру служебной информации, ограничив ее состав одним типом блоков, например, только заголовком. Тогда для опознания конца данных на приемной стороне в заголовке необходимо указывать длину этих данных и отсчитывать ее после декодирования передней границы.

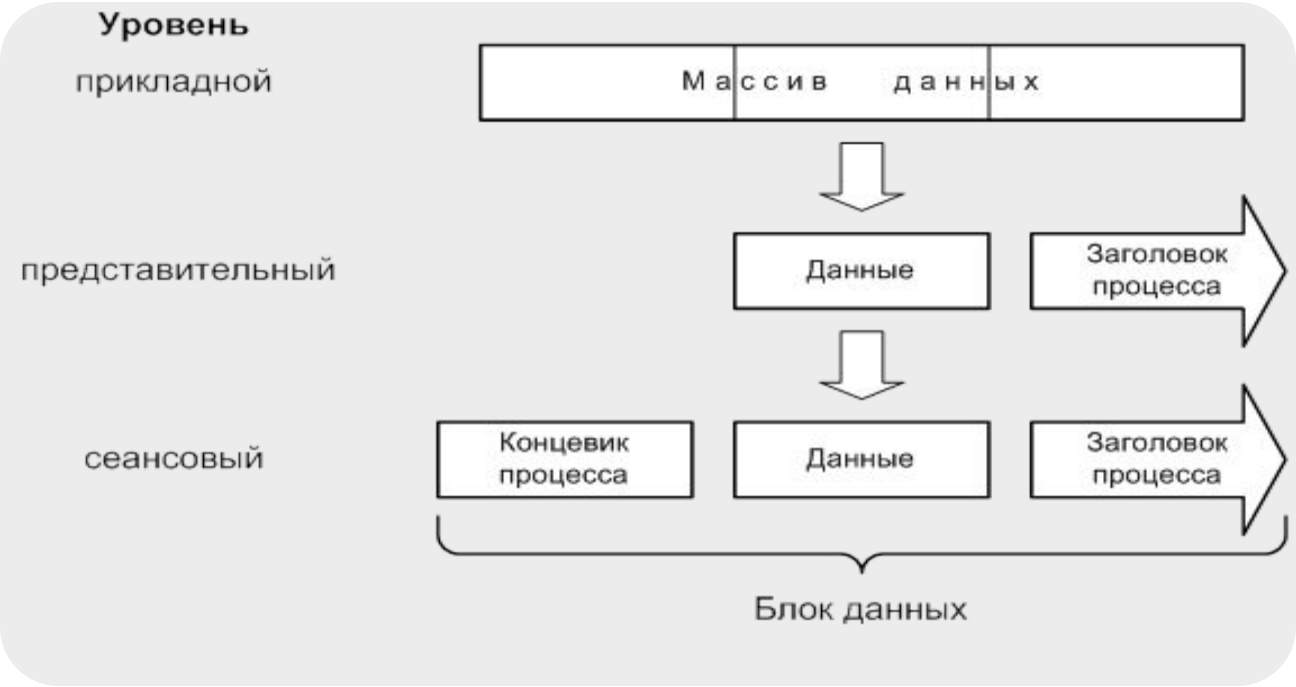
Схема упаковки данных при этом остается той же, что и в предыдущем случае за исключением того, что по мере продвижения данных от старших уровней к младшим осуществляется накопление лишь заголовков.

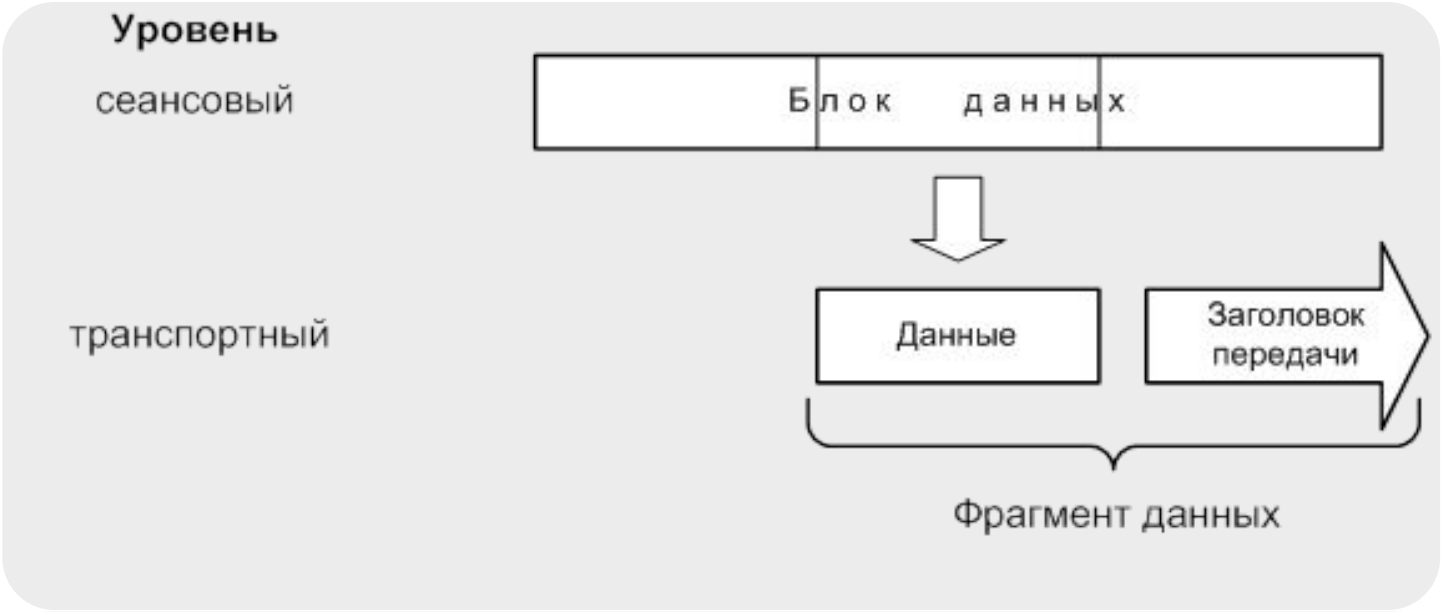
Задача состоит в том, чтобы передать некоторый массив данных на другой узел. Этот массив может иметь произвольную длину. Тогда при передаче этого массива представительскому уровню он сегментируется.

Представительный уровень добавляет к полученной части массива так называемый ***заголовок процесса***. В заголовке указываются идентификаторы исходного и конечного процессов, адреса взаимодействующих портов и имя массива.

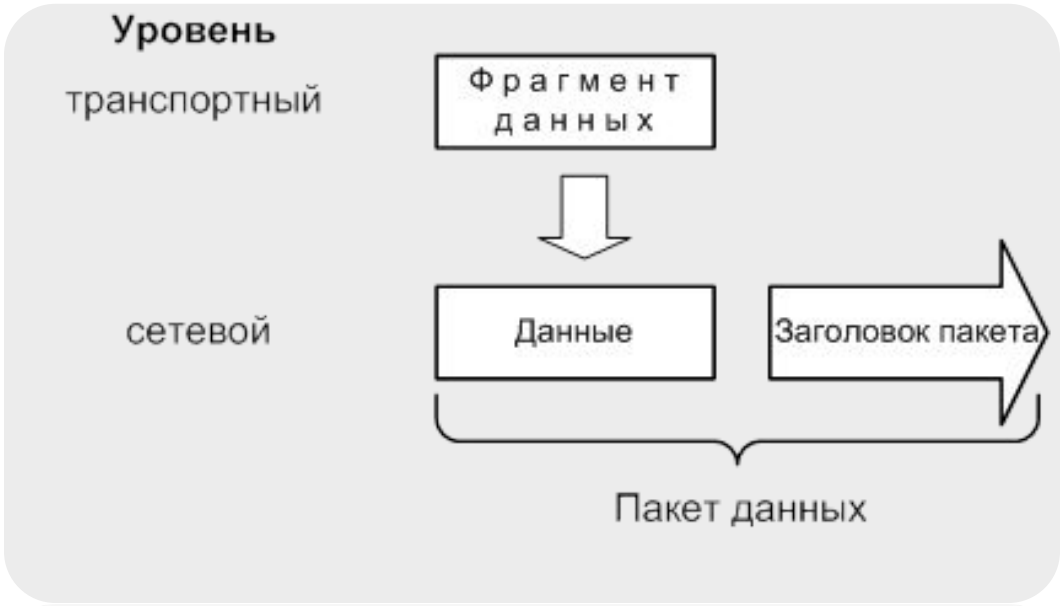
В таком виде данные передаются ***сеансовому*** уровню, который добавляет к ним *концевик процесса*. Основной информацией, содержащейся в концевике, являются проверочные символы, позволяющие выявлять ошибки на приемном конце, то есть после передачи данных в систему-адресат.

Полученная совокупность данных пользователя, заголовка и концевика процесса получила наименование *блока данных*.

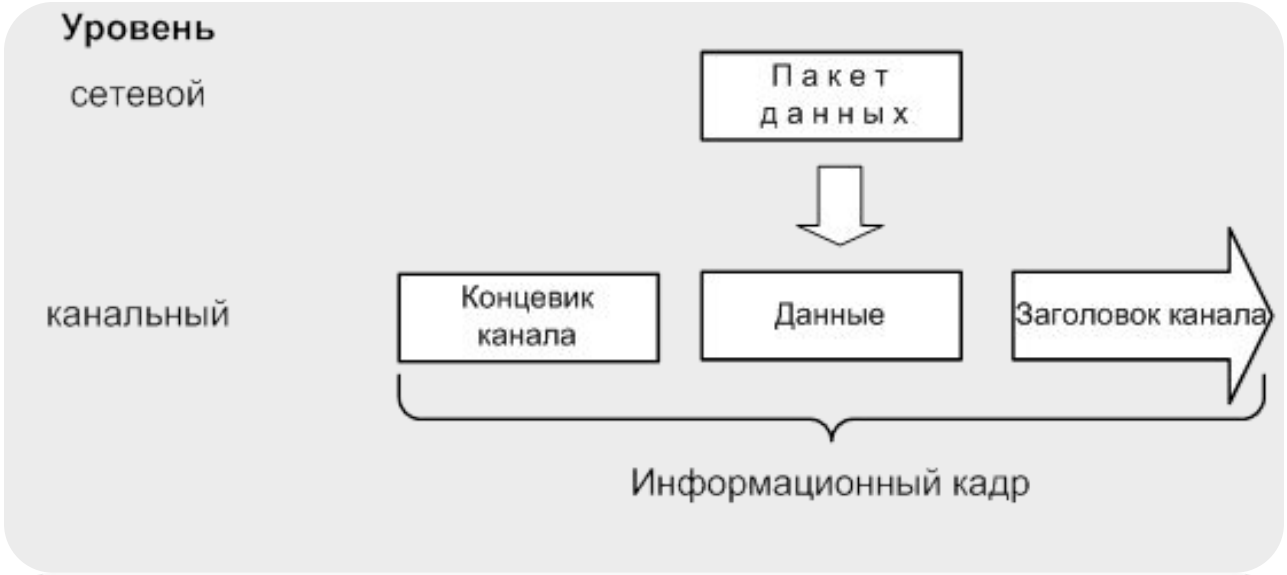




К каждому сегменту транспортный уровень добавляет служебную информацию в виде *заголовка передачи*. В результате получается *фрагмент данных*. В заголовке указывается тип массива, адреса взаимодействующих сеансовых объектов, идентификатор фрагмента. На приемной стороне заголовок передачи декодируется транспортным уровнем системы-получателя, после чего удаляется. Из поступающих фрагментов собирается *блок данных*, передаваемых сеансовому уровню.

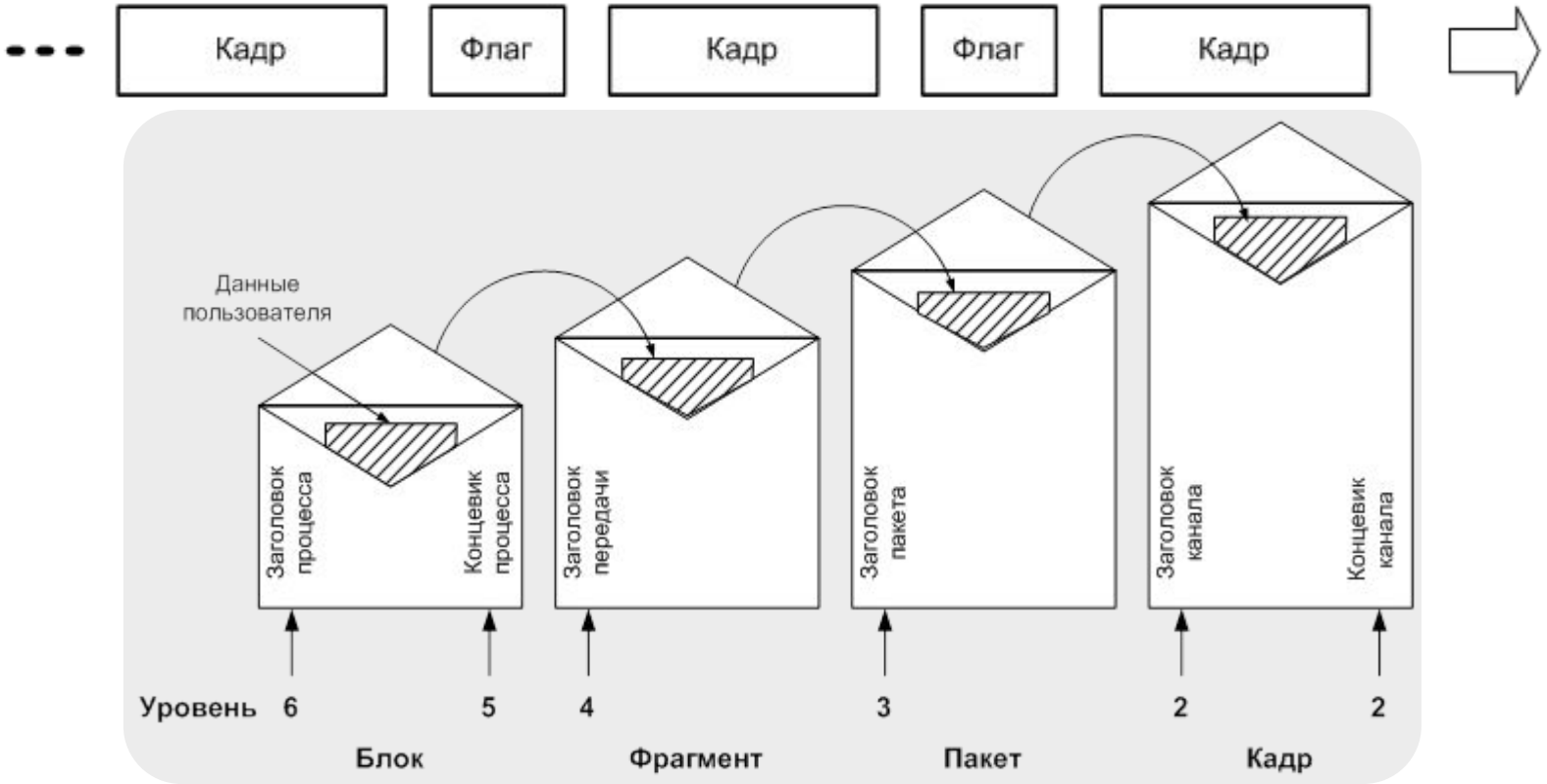


На ***сетевом*** уровне после выполнения процедуры маршрутизации к фрагменту данных добавляется *заголовок пакета*, что и приводит к образованию одноименной единицы данных.



Пакет данных после передачи его на ***канальный*** уровень обрамляется *заголовком* и *концевиком канала*. Этим завершается формирование основной единицы данных, подлежащей передаче по ***физическому*** каналу, – *информационного кадра*.

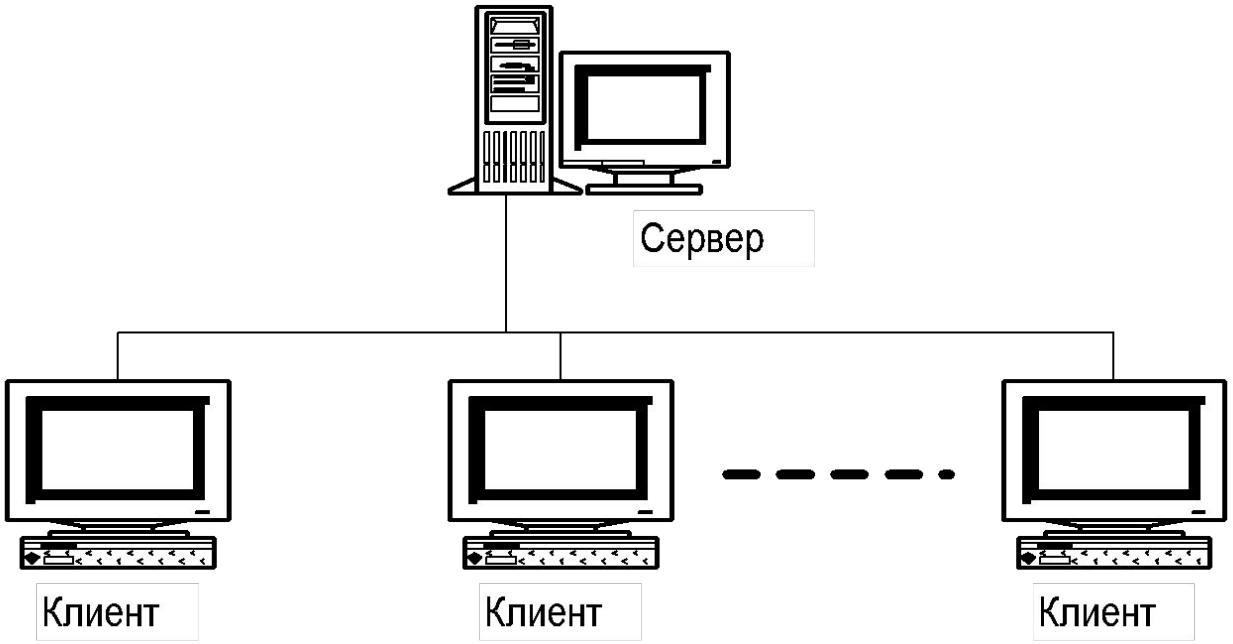
Служебная информация физического канала оформляется в виде *флага,* представляющего собой байт данных определенной конфигурации.



4. Модель «клиент-сервер» и ее модификации

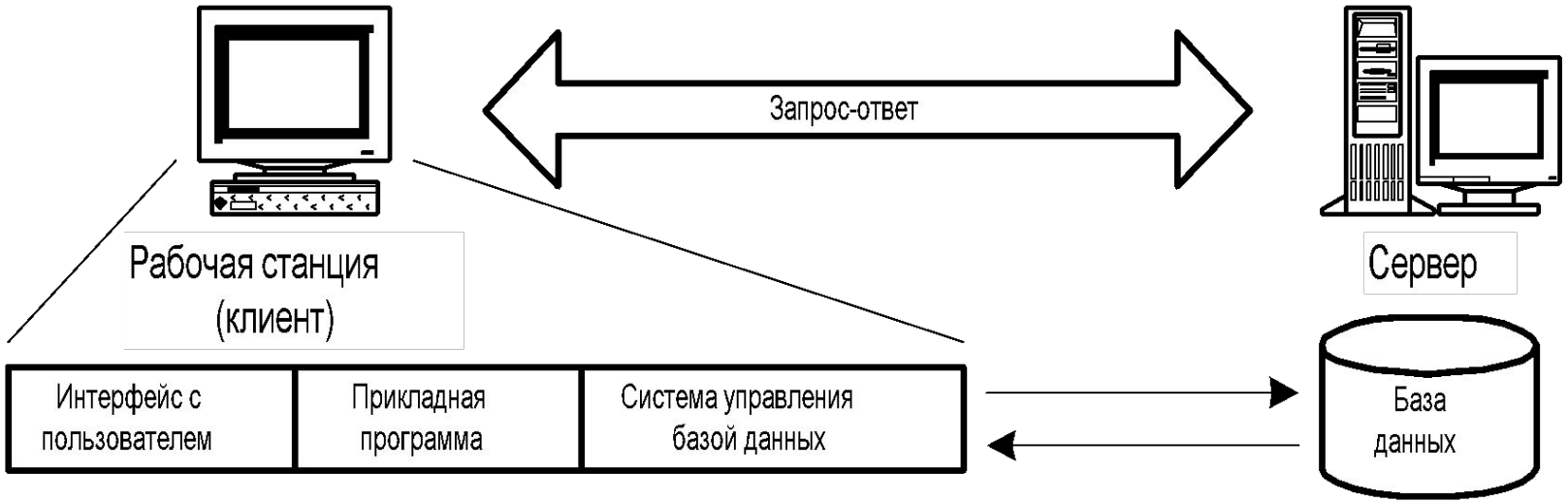
В случае организации доступа к информации, хранящейся в базах данных, различают несколько модификаций модели «клиент-сервер»:

- модель файлового сервера (*File Server* – FS);
- модель доступа к удаленным данным (*Remote Data Access* – RDA);
- модель севера базы данных (*DataBase Server* – DBS);
- модель сервера приложений (*Application Server* – AS).

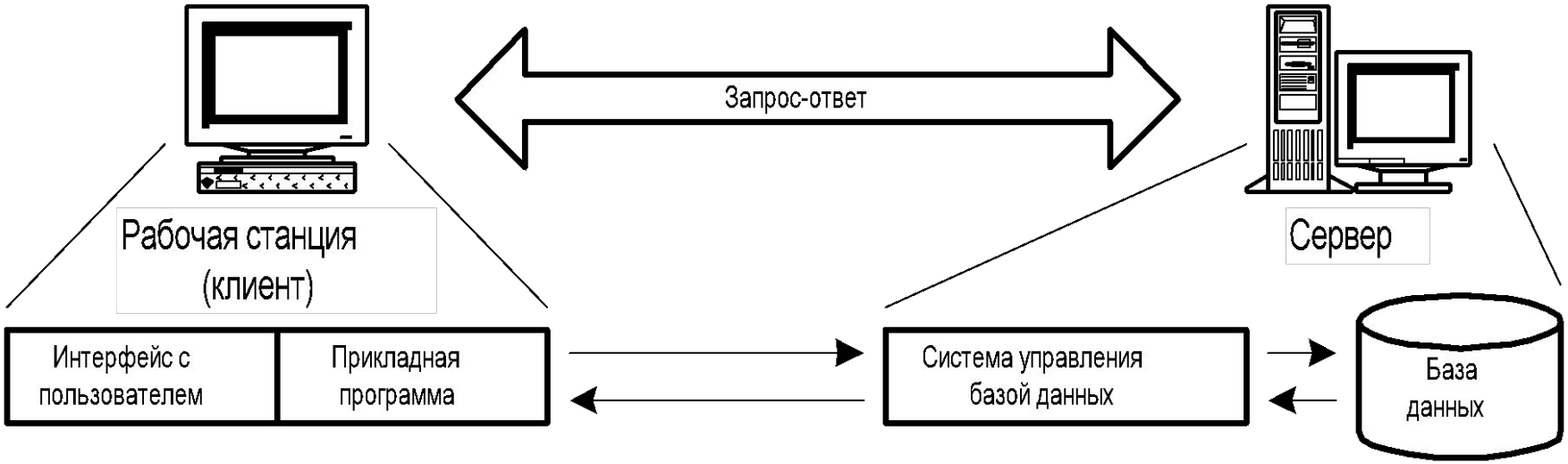


Модель файлового сервера (FS-модель) является базовой. Один из компьютеров в сети считается файловым сервером и предоставляет услуги по обработке файлов другим компьютерам. Файловый сервер работает под управлением сетевой операционной системы и осуществляет доступ к информационным ресурсам – файлам. На других компьютерах в сети функционирует приложение.

К недостаткам модели относят высокий сетевой трафик (передача множества файлов, необходимых приложению), узкий спектр операций манипулирования данными, отсутствие адекватных средств безопасности доступа к данным (защита только на уровне файловой системы)

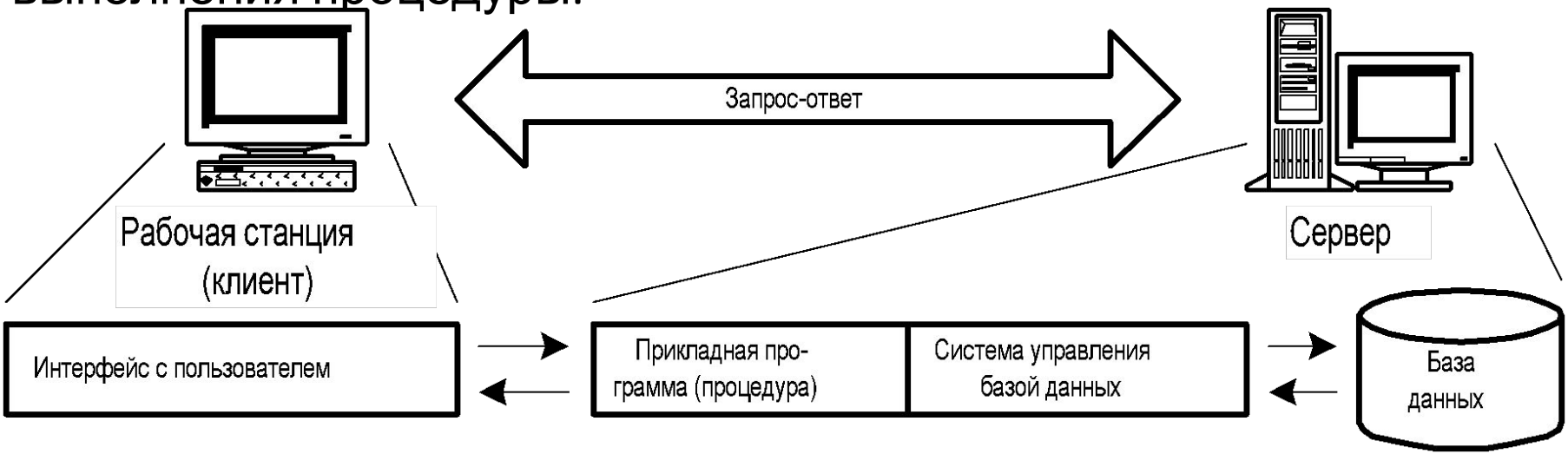


Более технологичная ***модель доступа к удаленным данным*** (RDA-модель) существенно отличается от FS-модели характером доступа к информационным ресурсам. Это обеспечивается операторами специального языка, такого как SQL. Клиент направляет запросы к информационным ресурсам (например, к базам данных) по сети удаленному компьютеру. На нем функционирует ядро системы управления базами данных, которое обрабатывает запросы, выполняя предписанные в них действия, и возвращает клиенту результат, оформленный как блок данных. При этом инициатором манипуляций с данными выступают программы, выполняющиеся на компьютерах-клиентах, в то время как ядру системы управления базами данных отводится пассивная роль – обслуживание запросов и обработка данных

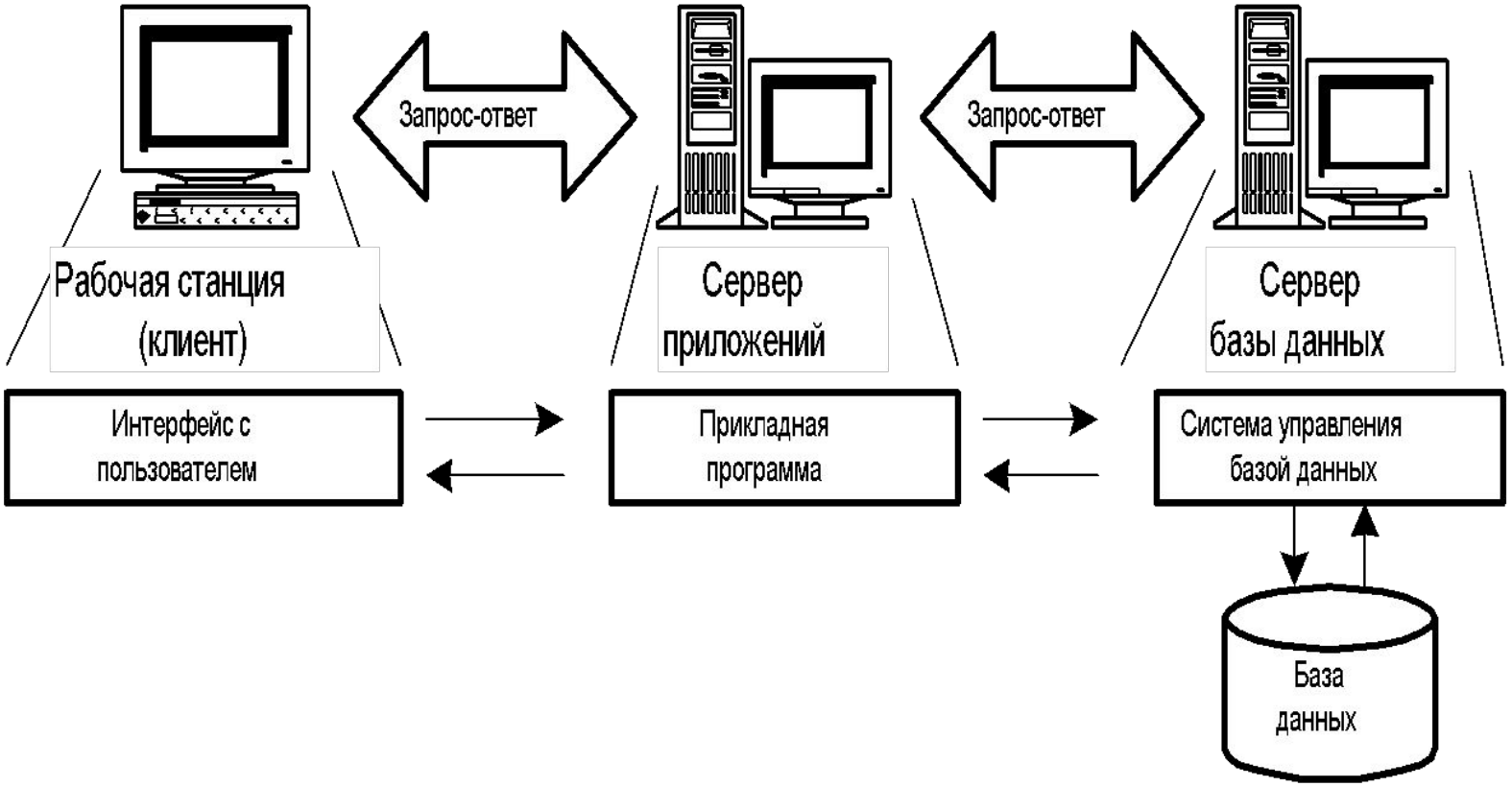


Наряду с RDA-моделью все большую популярность приобретает *модель сервера базы данных* (DBS-модель). Ее основу составляет механизм хранимых процедур – средство программирования SQL-сервера. Процедуры хранятся в словаре базы данных, разделяются между несколькими клиентами и выполняются на том же компьютере, где функционирует SQL-сервер.

Достоинства DBS-модели очевидны: это возможность централизованного администрирования прикладных функций на сервере, снижение трафика, возможность разделения процедуры между несколькими приложениями, экономия ресурсов компьютера за счет использования заранее созданного плана выполнения процедуры.



В ***модели сервера приложений*** (AS-модели) процесс, выполняющийся на компьютере-клиенте, как обычно отвечает за интерфейс с пользователем. Прикладные функции выполняются сервером приложения. Все операции над информационными ресурсами выполняются сервером баз данных.



5. Сети и протоколы передачи информации

На всех уровнях вычислительной сети от сети передачи данных до подключенных к ней вычислительных машин и терминалов используются разнообразные территориально рассредоточенные технические средства. Они предназначены для совместного выполнения задач пользователя сети. События, происходящие в столь сложной системе, отличаются тем, что заранее нельзя предсказать ни их характер, ни их последовательность, ни время их наступления.

Подчеркнем два очень важных обстоятельства.

Во-первых, устройства, подключенные к сети с коммутацией пакетов, должны иметь в своем составе некоторые средства обработки данных. Их необходимый минимум должен обеспечивать выполнение процедур согласования самого устройства с сетью. Устройства, не обладающие такими средствами обработки, должны подключаться к сети через устройства, отвечающие этому требованию.

Во-вторых, необходимо определить набор процедур, реализуемых в сети и в подключающихся к ней вычислительных машинах, обеспечивающих надежное выполнение задач пользователя. Эти процедуры осуществляют обмен информацией между территориально разнесенными техническими средствами и реализуются программным обеспечением. Взаимодействия двух таких процессов, которые обычно физически разнесены, обеспечивают реализацию информационного обмена через сеть

Два процесса, протекающих в различных географических точках, для координации своих действий и достижения «синхронизма» должны использовать обмен сообщениями. Такой обмен должен протекать в соответствии с тщательно разработанными процедурами. Эти процедуры называются ***протоколами***. Их основной особенностью является способность к работе в таких условиях, в которых последовательность и время наступления событий заранее неизвестны, и в процессе передачи возможны ошибки.

Понятие протокола уже было введено в предыдущих лекциях. Напомним, что под этим термином понимают *совокупность регламентирующих правил обмена данными между одноуровневыми объектами (процессами) различных систем* (ЭВМ).

Термин «протокол» также можно использовать для обозначения процедур обмена информацией между процессами не только в условиях сети, но и в многопроцессорных системах для управления взаимодействием параллельно выполняемых процессов в реальном масштабе времени, для управления несколькими устройствами. Кроме того, этот термин применим и в других системах, характеризующихся отсутствием, во-первых, фиксированных временных соотношений между наступлением событий и, во-вторых, взаимозависимостей между событиями и действиями, выполняемыми при их наступлении.

Функции протокола связаны с обменом сообщениями между процессами. Формат и содержание этих сообщений образуют *логические характеристики протокола*. Правила же выполнения процедуры определяют те действия, которые выполняют процессы, совместно участвующие в реализации протокола. Набор этих правил является *процедурной характеристикой протокола*. Используя эти понятия, мы можем теперь формально определить **протокол** как *совокупность логических и процедурных характеристик механизма связи между процессами*. Логическое определение составляет синтаксис, а процедурное – семантику протокола.

Функции протоколов сети передачи данных были частично рассмотрены в предыдущей лекции. Наиболее важными из них являются:

защита от ошибок;

управление потоком и защита от перегрузки;

выполнение операций по маршрутизации.

Защита от ошибок обеспечивает достоверность, как данных пользователя, так и управляющих сообщений, обмен которыми происходит при выполнении протоколов сети.

Управление потоком и защита от перегрузки дают возможность распределять ресурсы сети между большим числом пользователей, предоставляя каждому из них необходимые услуги без ущерба для работы сети.

Выполнение операций по маршрутизации и оптимизации использования ресурсов сети дает также большую степень доступности услуг сети путем обеспечения альтернативных маршрутов между двумя пунктами сети.

Несмотря на то, что перечисленные функции выполняются в базовой сети, некоторые из них должны дублироваться в протоколах, реализуемых в вычислительных машинах при их взаимодействии через сеть. Это необходимо по той причине, что сеть не управляет ходом выполнения задач или распределением ре сурсов этих вычислительных машин между задачами.

Протокол TCP/IP

Стек протоколов TCP/IP – фундамент сети Internet. Он неуклонно превращается в наиболее распространенный протокол сетевого и транспортного уровней для сетей всех размеров и конфигураций. Поэтому ему уделяется наибольшее внимание. Однако, протокол TCP/IP – это не только стек протоколов сетевого и транспортного уровней, но и полный набор протоколов, работающих на многих уровнях сетевой модели. Пакет протоколов TCP/IP включает также дополнительные компоненты, необязательные в процессах сетевой коммуникации, например, утилиты прикладного уровня, также входящие в состав пакета TCP/IP.

а) *Протокол сетевого уровня IP*

Как упоминалось раньше, на сетевом уровне выполняются задачи маршрутизации. В протоколах TCP/IP маршрутизация поддерживается путем применения IP-адресов, идентифицирующих сетевые устройства. Каждый компьютер, принтер (подключенный к сети), маршрутизатор или любое другое сетевое устройство имеет уникальный IP-адрес.

Каждый IP-адрес состоит из двух частей. Вместе они идентифицируют сеть, в которой расположено устройство, и само устройство. Один раздел IP-адреса представляет *сеть,* а другой – *хост* (отдельный компьютер).

Название улицы сообщает почтовой службе о том, в каком районе расположен дом. Название этой улицы присутствует в адресах многих домов, расположенных на ней.

Номер дома уникален для каждого отдельного дома, расположенного на этой улице. В городе есть много домов с таким номером, однако существует только один дом с названием улицы и номером дома, указанными в адресе.

Аналогично этому, один и тот же адрес сети имеют многие компьютеры, однако сочетание адреса сети и адреса хоста для каждого компьютера уникально (вернее, для каждого сетевого адаптера, ведь есть компьютеры с несколькими сетевыми адаптерами).

Например, первые три раздела IP-адреса 201.32.0.4 идентифицируют сеть. Разделы называются ***октетами***. Последний раздел идентифицирует сетевой интерфейс отдельного компьютера. Все компьютеры в этой подсети имеют один и тот же номер сети (идентификатор сети) – 201.32.0. Однако каждый компьютер имеет свой номер хоста, а .4 единственный в этой подсети.

б) *Классы IP-адресов*

Подобно любой обрабатываемой компьютером информации, IP-адрес состоит из двоичных цифр, т.е. битов. Человеку трудно работать с длинными строками из нулей и единиц, поэтому обычно IP-адреса записываются в *десятичном* формате. Однако IP-адрес в десятичной записи – это не десятичное число. Десятичным числом является каждый раздел, или октет IP-адреса. Октеты отделяются друг от друга точкой. Это не десятичная точка, отделяющая целую часть от дробной, а всего лишь условный символ, отделяющий октеты друг от друга.

Длина октета равна восьми битам, т.е. октет – это последовательность из восьми нулей или единиц. Иногда четыре октета обозначаются как w.x.y.z. В такой записи первый октет (правый) называется z-октетом, следующий – y-октетом и т.д.

IP-адрес состоит из четырех октетов по восемь битов каждый, всего – 32 бита. Это значит, что максимальное количество различимых IP-адресов равно 2^{32}, или 4 294 967 296. Типичный IP-адрес выглядит так: 192.168.1.12.

Деление IP-адресов на классы в зависимости от размера сети называется ***классовой адресацией***.

Традиционно в классе А первый октет служит адресом сети, а три остальных – адресом хоста. Первый (старший) бит первого октета в адресе класса А используется для идентификации класса, поэтому для идентификации сети остается только семь битов.

В адресе класса В для идентификации сети используются два первых октета, а для идентификации хоста – два вторых. Первые два октета содержат 16 бит, однако первые два из них используются для идентификации класса. Таким образом, для идентификации номера сети остается 14 бит.

В классе С первые три октета (24 бита) используются для идентификации сети, а последний – для идентификации хостов. Три первых бита идентифицируют класс, поэтому, чтобы получить количество битов, выделенных для идентификации сети, нужно из 24 вычесть 3. Другими словами для идентификации номера сети используется 21 бит.

Класс адресов	Количество сетей	Количество хостов на сеть	Старшие биты	Диапазон адресов первого октета	Количество битов в адресе сети
A	126	16777216	0	0-127	7
B	16384	65535	10	128-191	14
C	2097152	254	110	192-223	21
D			1110	224-239	28

Как видно из таблицы, доступны только 126 адресов класса A. К настоящему времени все они уже заняты. Они присвоены наиболее крупным корпорациям и учебным центрам, таким, как IBM, Hewlett Packard, Xerox, MIT (Massachusetts Institute of Technology), Columbia University, Digital Equipment Corporation, General Electric и Apple. В каждой из этих сетей индивидуальные номера можно присвоить 16 млн. хостов.

В приведенной схеме адресации можно создать 2 млн. сетей класса C, в каждой из которых может быть не более 254 адресов хостов. Адреса класса C выделяются провайдерам Internet.

Адреса класса B занимают промежуточное положение. Они присваивались главным образом большим компаниям, размер которых в то время был недостаточен для класса A. Компании Microsoft выделены адреса класса B.

Адреса класса D предназначены для *широковещательных сообщений,* т.е. для передачи одного сообщения одновременно многим получателям. Адрес класса D присваивается специальной группе компьютеров, в этом случае пакеты обрабатываются и распределяются широковещательными протоколами.

Итак, одна часть IP-адреса идентифицируетет сеть, а другая – компьютер (хост). Но где же расположены эти части в IP-адресе? Ответ на этот вопрос неоднозначен. Традиционно это зависит от класса сети, являющегося одновременно классом IP-адреса (существует и другой метод адресации, названный *бесклассовым.)*.

IP-адреса предоставляет организация IANA (Internet Assigned Numbers Authority). Первое время после появления Inrernet казалось логичным выделять IP-адреса компаниям и организациям блоками, потому что для коммуникации в Internet каждому компьютеру локальной сети нужен уникальный адрес Размер выделяемого блока адресов зависел от размера локальной сети. Большим компаниям нужны были большие блоки адресов, а маленьким – маленькие. ***Классы адресов*** назначались на основе размера локальной сети, т.е. количества хостов. В таблице показаны традиционные классы IP-адресов.

в) *Протоколы транспортного уровня: TCP и UDP*

Как отмечал в предыдущей лекции, транспортный уровень отвечает за обеспечение надежной связи одного компьютера с другим. Эта задача реализуется такими механизмами, как, например, подтверждение получения данных принимающим компьютером без потерь или повреждений.

Протоколы транспортного уровня предназначены также для идентификации сообщений, прибывающих на один и тот же компьютер. Различные приложения, установленные на одном компьютере, могут передавать и принимать сообщения одновременно. Чтобы разделять эти сообщения, в протоколах транспортного уровня используются *порты*.

Для идентификации передающего и принимающего компьютеров в TCP/IP используются логические адреса, состоящие из двух частей, т.е. IP-адреса. Но что получается, если два сетевых приложения, выполняющихся на одном компьютере, посылают или принимают сообщения одновременно? Протоколы должны различать эти сообщения. Для этого используются порты TCP и UDP.

Порт – это *точка логического соединения.* В транспортных протоколах TCP и UDP порты используются для идентификации конкретного приложения, передающего или принимающего сообщение.

Пакет TCP/IP содержит не один, а два протокола транспортного уровня.

TCP (Transmission Control Protocol – протокол управления передачей). Это протокол, ориентированный на установление соединения.

UDP (User Datagram Protocol – протокол пользовательских дейтаграмм). Этот протокол, *не* ориентирован на установление соединения.

Какой из этих двух протоколов используется для передачи конкретного сообщения, зависит от назначения и характера передаваемых данных. Протокол TCP используется, когда более важна надежность передачи данных, а UDP когда более важной характеристикой является производительность (скорость) коммуникации.

Протокол транспортного уровня TCP

Прежде чем начать передавать данные, TCP устанавливает между двумя сообщающимися компьютерами сеанс соединения. Для этого используются сообщения уведомления и ответа. Затем выполняются процедуры обнаружения и исправления ошибок, и данные разбиваются на пакеты.

В каждый пакет добавляется информация о нумерации пакетов, чтобы на принимающем конце их можно было собрать в правильной последовательности. Нумерация пакетов позволяет принимающему компьютеру обнаружить недостающие пакеты. Благодаря этим процедурам протокол TCP надежнее, чем UDP, однако выполнение дополнительных операций существенно снижает производительность.

Протокол транспортного уровня UDP

Протокол UDP не ориентирован на установление соединения. Не выполняется также нумерация пакетов данных, поэтому он более пригоден для передачи небольших сообщений, которые можно разместить в одном пакете. Протокол UDP не отслеживает также, что было передано и что получено. Однако в UDP выполняется проверка контрольной суммы, чем гарантируется правильность данных, поступивших на принимающий компьютер. Как и в TCP, в UDP, чтобы различить сообщения для или от разных приложений одного и того же компьютера, используются номера портов.

Протокол UDP не занимается нумерацией пакетов или обнаружением ошибок, поэтому его производительность высокая. Заголовок пакета UDP проще заголовка TCP.

Вопросы для самостоятельной работы:

Изучить:

Модель «клиент-сервер» и ее модификации.

схему упаковки-распаковки данных.

Протокол TCP/IP.

Протоколы IPX/SPX и NetBIOS/NetBEUI

Задание на самоподготовку:

Изучить:

Общую характеристику эталонной модели открытых систем.

Модель «клиент-сервер» и ее модификации.

Доработать конспект лекции.

Протоколы IPX/SPX и NetBIOS/NetBEUI

Вопросы для самоконтроля

Дать общую характеристику эталонной модели открытых систем.

Объяснить назначение и функции уровней модели структуре сетей.

Схему упаковки-распаковки данных.

Модель «клиент-сервер» и ее модификации.