1 Основные понятия

1.1 Определения компьютерных сетей и их компонентов

Рассмотрим самое общее определение современной компьютерной сети.

Компьютерной сетью (КС) назовем совокупность *компьютеров*, объединенных *сетью передачи данных* для обмена данными с целью *совместного решения* различных *прикладных задач* с использованием двух или более компьютеров сети.

Компьютерный сети также зачастую называют вычислительными сетями, что является отражением того факта, что изначально компьютерные сети создавались для решения вычислительных задач. В рамках настоящей книги мы будем использовать эти термины как эквивалентные.

В число интеллектуальных устройств кроме собственно компьютеров могут входить различные мобильные устройства (телефоны, планшеты), а также контроллеры (специальные процессоры) различных управляемых по сети объектов: “умных” домов и их подсистем, автомобилей, и других технических объектов различного назначения.

Сеть передачи данных является совокупностью *каналов передачи данных* и соединяющих их *коммуникационных устройств*.

Под компьютерами здесь понимаются не только традиционные компьютеры, но и мобильные устройства доступа к сети (смартфоны, планшеты и пр.), встроенные компьютеры различных технических устройств и вообще, любые устройства, оснащенные процессором, оперативной памятью, устройствами связи с внешним миром и операционной системой.

Каналы передачи данных могут иметь различную организацию, использовать различную *среду передачи цифрового сигнала* (т.е. данных) и иметь различную пропускную способность (скорость передачи данных).

К числу наиболее распространенных типов среды передачи цифрового сигнала относятся различные виды медных или волоконно-оптических кабельных систем, радиоэфир, воздушная оптическая среда. Среда передачи данных может быть *монопольно используемой* для организации одного канала между парой устройств (любое из них может быть как конечным интеллектуальным устройством, так и коммуникационным устройством) или разделяемой и обеспечивать создание на ее базе нескольких каналов.

В качестве примеров разделяемой среды передачи цифрового сигнала приведем радиоэфир и кабельную систему сегмента сети Ethernet, построенного с использованием одного или нескольких концентраторов (hub-ов), как это подробно рассматривается во 2-й главе. В качестве примера монопольно используемой среды передачи упомянем среду телефонный кабель, используемый для модемного подключения компьютера конечного пользователя к компьютерной сети.

Перейдем к рассмотрению характеристик пропускной способности канала передачи данных. Пропускная способность канала - это максимально достижимая скорость передачи данных через этот канал. Пропускная способность каналов измеряется в битах

в секунду (bit per second - bps), а также в Кило-, Мега-, Гига-, Терабит в секунду (K-, M-, G-, Tbps).

К числу коммуникационных устройств, используемых для “соединения” каналов передачи данных, относятся серверы (компьютеры, предоставляющие некоторые услуги другим компьютерам сети), маршрутизаторы (роутеры), коммутаторы (свитчи), мосты (бриджи) концентраторы (хабы), модемы для различных сред передачи, точки радиодоступа, мультиплексоры (все эти устройства с различной степенью подробности рассматриваются в настоящей книге) и некоторые другие более “экзотические” устройства. Различные каналы подключаются к сетевым портам (специализированным устройствам ввода-вывода) этих устройств. Отметим, что подавляющее большинство перечисленных устройств так же, как и “конечные”, пользовательские устройства доступа к сети являются специализированными компьютерами, ”заточенными” на решение коммуникационных задач.

Бегло рассмотрев техническую сторону организации компьютерных сетей, перейдем к рассмотрению цели их создания - совместному решений задач пользователей несколькими интеллектуальными устройствами компьютерной сети.

2 Сетевые протоколы и их многоуровневая организация

2.1 Определение и принципы уровневой организации сетевых протоколов

Коротко говоря, сетевыми протоколами называются правила взаимодействия компьютеров сети при совместном решении ими тех или иных задач. В русском языке слово “протокол” может иметь два различных значения. Так протоколом называют зачастую некоторый документ, фиксирующий процесс выполнения некоторого мероприятия (протокол собрания) или достигнутого соглашения (протокол о намерениях сторон). Но протоколом же называют и некоторый свод правил, регламентирующих взаимодействие сторон (дипломатический протокол). Именно последнее значение слова “протокол” и используется в термине “сетевой протокол”. И так же, как и в дипломатическом протоколе, для сетевого протокола любое отклонение какой-либо из взаимодействующих сторон от соблюдения определяемого протоколом свода правил может повлечь за собой полный разрыв каких-либо отношений, либо даже катастрофу (вплоть до военного конфликта, в случае нарушений дипломатического протокола, либо вывод на запредельный уровень сердечников ядерного реактора, управляемого через компьютерную сеть, при несоблюдении сетевых протоколов).

Приведем более формальное определение сетевого протокола.

Сетевой протокол – это совокупность правил, методов, стандартов и реализующих их аппаратных и программных средств, совместно обеспечивающих взаимодействие компьютеров в компьютерной сети.

Отметим, что указанная совокупность включает чрезвычайно широкий спектр правил, стандартов и пр. и не может быть реализована в аппаратуре и программном обеспечении без специальной структуризации этой совокупности. С одной стороны указанного спектра находятся “низкоуровневые”, неинтересные и обычно непонятные конечному пользователю (но необходимые для обеспечения возможности передачи между компьютерами каких-либо данных) требования к используемой среде передачи данных, способу представления цифрового сигнала некоторым сигналом этой среды, соединительным разъемам и аппаратной реализации приемо-передающих устройств и т.д. С другой стороны — “высокоуровневые” требования к языку общения пользователя с некоторым сетевым приложением и языку взаимодействия клиентской и серверной частей этого приложения.

Поэтому естественным способом структуризации сетевых протоколов, зачастую используемым при реализации других сложных программных систем, является их уровневая организация, в которой все множество сетевых протоколов разбивается на несколько (скажем на N) иерархически упорядоченных уровней. При этом пересылку данных через среду передачи могут выполнять лишь протоколы нижнего (1-го) уровня; с пользователем взаимодействуют лишь протоколы верхнего (N-го) уровня; а каждый из промежуточных уровней i взаимодействует лишь через стандартизованные интерфейсы пересылки сообщений (никаких глобальных переменных!) между этим уровнем и соседними с ним уровнями i-1 и i+1.

Под интерфейсом между парой уровней (верхним и нижним) может выступать совокупность параметризованных функций нижнего уровня, путем вызова которых осуществляется взаимодействие верхнего уровня с нижним. Обратное взаимодействие реализуется путем возврата значений вызываемых функций и их изменяемых параметров. Однако возможно взаимодействие уровней путём пересылки сообщений между ними. Именно так обычно организуется взаимодействие между уровнями, реализованными в ядре ОС (см. ниже).

Более детально: для пересылки информации через сеть i-й уровень ($i \in [2,N]$) “выполняет” требуемую пересылку путем “отправки” соответствующего сообщения своему соседу снизу, т.е. уровню i-1 (напомним, что фактически такая “отправка” обычно реализуется как передача параметров при вызове соответствующей функции следующего уровня); при получении информации из сети уровень j ($j \in [1, N-1]$) направляет эту информацию по направлению к пользователю путем отправки соответствующего сообщения уровню j+1 (обычно в виде результата выполнения функции текущего уровня вызывавшей его функции более высокого уровня и/или измененных значений параметров таких функций).

Обычно при уровневой разработке сложных систем ее ведут по принципу “снизу вверх”. Определяют на нижнем (первом) уровне некоторый набор функций, при помощи которого можно будет, не вдаваясь в несущественные для следующего (второго) уровня детали, запрограммировать функции второго уровня, за ним - третьего, и т.д. В случае сетевых протоколов довольно естественной функцией первого уровня представляется пересылка данных между смежными (непосредственно связанными средой передачи) компьютерами (как мы увидим дальше, на самом деле эта функция реализуется двумя смежными уровнями), на 2-м уровне — обеспечение транзитной передачи данных между

несмежными компьютерами через промежуточные с решением на каждом узле задачи маршрутизации, состоящей в выборе одного из нескольких возможных направлений пересылки. На третьем уровне могла бы решаться задача надежной доставки пакетов данных, включая повторную пересылку потерянных пакетов.

При рассмотренной выше организации взаимодействия уровней сетевых протоколов обеспечивается соблюдение двух принципов, известных как принципы построения открытых систем.

1) Реализация каждого из уровней полностью независима от реализации любых других уровней (при условии неизменности интерфейсов) этого и соседних с ним уровней и может быть изменена без нарушения работоспособности всей совокупности сетевых протоколов. Соблюдение этого принципа уменьшает объем независимо решаемых задач при изначальном создании сетевых протоколов и позволяет выполнять независимое развитие протоколов различных уровней.
2) Взаимодействие одноименных (имеющих одинаковый номер в иерархии) уровней различных компьютеров сети прозрачно по отношению к другим уровням. На обоих концах сети эти уровни, при их взаимодействии друг с другом, “видят” только интерфейс к более низкому уровню, воспринимаемый ими как интерфейс друг к другу.

Соблюдение указанных принципов при построении любых сложных систем сетевого программного обеспечения делает их открытыми для понимания и независимого изменения различных “частей” указанных систем.

Иллюстрация взаимодействия уровней сетевых протоколов приведена на рис. 0.1.

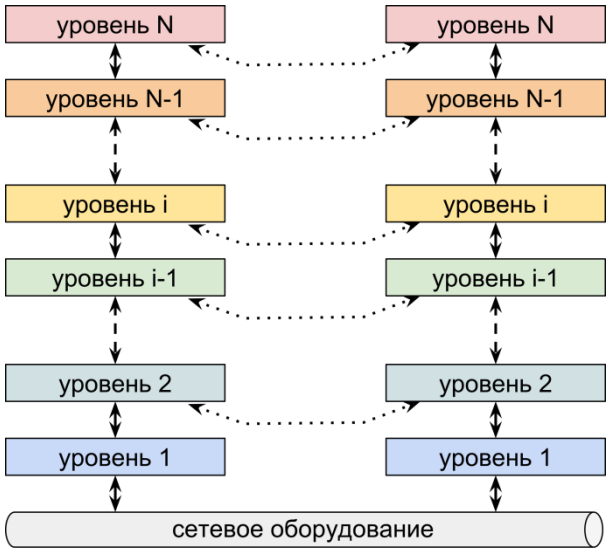

Рис. 0.1. Схема организации взаимодействия протоколов в многоуровневой модели

На этом рисунке представлена одновременно и картина реального взаимодействия модулей (стрелки, соединяющие соседние уровни в “столбцах” уровней, соответствуют интерфейсам между этими уровнями) и картина прозрачного взаимодействия одноименных уровней (точечные стрелки).

Рассмотренная нами схема взаимодействия уровней является абстрактной, применимой к любым протоколам с уровневой организацией. Но в силу своей абстрактности она не дает никакого представления ни о реальном количестве уровней, ни о функциональности каждого из уровней. Поэтому далее мы рассмотрим две конкретные модели: эталонную модель уровневой организации сетевых протоколов OSI/ISO и уровневую организацию протоколов семейства TCP/IP.

2.2 Эталонная модель уровневой организации сетевых протоколов OSI/ISO

Для начала расшифруем смысл аббревиатуры “OSI/ISO”. OSI (Open Systems Interconnection - взаимодействие открытых систем) - это название соответствующего стандарта, рекомендуемого в качестве эталона всем разработчикам новых сетевых протоколов. ISO (International Standard Organization - международная организация по стандартам) - это название организации, разработавшей указанный стандарт.

Организация ISO разрабатывала свою эталонную модель уровневой организации сетевых протоколов OSI одновременно с повсеместным внедрением протоколов TCP/IP. Поэтому обсуждаемая в пункте 1.7.3 близость модели OSI и уровневой организации протоколов TCP/IP совершенно не случайна.

В соответствии с эталонной моделью OSI множество всех сетевых протоколов разбивается на следующие 7 уровней, указанные нами в порядке сверху вниз:

7. Прикладной (Application layer)
6. Уровень представления данных (Presentation layer)
5. Сеансовый уровень (Session layer)
4. [redacted])
3. Сетевой уровень (Network layer)
2. Уровень каналов данных (Data link layer)
1. Физический уровень (Physical layer)

Охарактеризуем кратко каждый из уровней в порядке “снизу вверх”, указав основные функции, их форму реализации и соответствующие стандарты. При рассмотрении функций уровней мы будем называть *сетевыми устройствами* 1) компьютеры (смартфоны, планшеты и другие мобильные устройства мы отнесем к категории компьютеров) или 2) коммуникационные устройства, обеспечивающие пересылку данных между сегментами сети и называемые *маршрутизаторами*.

Физический уровень обеспечивает выполнение процесса передачи непрерывного битового потока данных между непосредственно связанными средой передачи данных сетевыми устройствами.

На этом уровне стандартизуются: используемая среда передачи данных (разновидности кабеля, радиосреда и др.) с её детальными характеристиками; используемые соединительные разъемы и методы аппаратной реализации; способы представления пересылаемого через среду передачи цифрового сигнала изменением некоторого параметра среды передачи (например, разностью электрического потенциала в медных кабелях, яркость светового луча в волоконно-оптическом кабеле, различной

частотой любого электромагнитного сигнала, представляющего значения “0” и “1” для разнообразных сред передачи).

Протоколы физического уровня реализуются в аппаратуре всех устройств, подключенных к сети. При этом разные технологии передачи данных реализуются различными, соответствующими технологии

Уровень каналов данных (или просто “канальный уровень”) в некоторых источниках именуют “уровнем звена связи”, что является далеко не самым удачным переводом англоязычного названия уровня. Основной задачей этого уровня является **потенциально ненадежная** доставка **корректных** пакетов данных (в отличие от битового потока) между сетевыми устройствами, непосредственно связанными средой передачи данных. Для обеспечения корректности передачи данных могут использоваться специальные методы избыточного кодирования (количество бит закодированного значения превосходит длину в битах самого этого значения) и/или иные способы проверки контроля корректности принятых из сети данных (такие, например, как контрольные суммы). При использовании методов избыточного кодирования в ряде случаев возможно восстановление корректных исходных значений по искаженным принятым битам. Если восстановление значения хотя бы одного байта из принятого пакета невозможна, отбрасывается весь пакет

Сетевой уровень обеспечивает ненадежную передачу пакетов данных между произвольными сетевыми устройствами, которые могут находиться как в одном и том же, так и в *различных сегментах и даже различных сетях*.

В случае, когда взаимодействующие сетевые устройства находятся в различных сегментах сети, средствами данного уровня решается задача маршрутизации пакетов, транзитом проходящих через соединяющее сегменты сетевое устройство в направлении “конечного” сетевого устройства. Задача маршрутизации, то есть выбора одного из нескольких потенциально возможных направлений, ведущих к цели, решается на каждом из упомянутых транзитных сетевых устройств, обычно именуемых маршрутизаторами.

Кроме своей основной задачи — маршрутизации пакетов — средства сетевого уровня могут разбивать пакеты на части (выполнять фрагментацию пакетов), если размер этих пакетов превосходит максимально возможную длину пакетов используемой среды передачи. Тогда при приеме таких пакетов выполняется их “сборка”, называемая дефрагментацией.

Средства сетевого уровня реализуются в виде модулей ядра операционной системы сетевого устройства.

Транспортный уровень *обеспечивает надежную передачу данных* между *парой программных процессов*, работающих на одном и том же, или на различных компьютерах, расположенных в произвольных частях сети. Надежность обеспечивается контролем доставки пакетов и повторной пересылкой “потерянных” пакетов. Поскольку последовательные пакеты, маршрутизируемые по разным маршрутам, могут в ходе пересылки “обгонять” друг друга, важной функцией этого уровня является управление последовательностью доставки пакета их получателю. И, наконец, размер пакета транспортного уровня может превосходить максимально допустимую длину пакета сетевого уровня. В этом случае на передающей стороне выполняется сегментация пакета (разбиение его на сегменты), а на принимающей — его десегментация.

Разные сегменты сети по пути следования пакетов имеют разную пропускную способность. Важной функцией протокола транспортного уровня является контроль перегрузки сети. В 1986 году после повсеместного внедрения надежного протокола транспортного уровня, в котором не было функции контроля, из-за потоков данных из быстрой сети национального научного фонда США NSFNet в медленный ARPANet весь интернет несколько месяцев страдал от постоянных перегрузок сети.

Средства транспортного уровня реализуются в виде модулей ядра операционной системы сетевого устройства.

Сеансовый уровень

И, наконец, **прикладной уровень**

2.3 Уровневая организация и стек протоколов TCP/IP

Протоколы TCP/IP были разработаны раньше эталонной семиуровневой модели OSI/ISO, и опыт разработки этих протоколов был весьма существенно учтен при разработке указанной модели. Модель протоколов семейства TCP/IP состоит из пяти уровней:

5. Прикладной (Application layer)
4. Транспортный уровень (Transport layer)
3. Межсетевой уровень (Internet layer или Internet Protocol - IP)
2. Уровень каналов данных (Data link layer)
1. Физический уровень (Physical layer)

Нетрудно заметить, что названия двух нижних уровней совпадают с OSI. Совпадают и их функции. И, поскольку для некоторых технологий четкой границы между этими уровнями нет, их обычно также объединяют в драйверный уровень. На этом уровне, в зависимости от сетевой технологии сегмента, подключенного к определенному сетевому интерфейсу, “прикрепляется” драйвер, обслуживающий интерфейсы к сегментам, построенных на базе соответствующей технологии. Одновременно с драйвером для каждого сетевого интерфейса используется модуль ARP одноименного протокола ARP (Address Resolution Protocol - протокол разрешения адресов), используемый для преобразования видов адресов, применяемых на более высоком IP-уровне, в адреса канального уровня. Уровень этого модуля несколько выше уровня соответствующего драйвера и взаимодействует только с этим драйвером.

Третий уровень модели TCP/IP называется межсетевым (Internet layer или Internet Protocol - IP). Вот мы и встретились впервые с одним из основных протоколов семейства TCP/IP, название которого содержится в название семейства. По своим функциям протокол IP модели TCP/IP идентичен протоколам сетевого уровня модели OSI/ISO. Но на межсетевом уровне (или IP-уровне) кроме протокола IP работает межсетевой протокол управляющих сообщений ICMP (Internet Control Message Protocol), основной задачей которого является контроль работоспособности протокола IP. На самом деле два этих протокола находятся в неразрывном единстве и даже определены в “соседних” стандартах RFC 791 и RFC 792 соответственно.

Транспортный уровень модели TCP/IP включает два независимых протокола TCP и UDP. TCP — это второй протокол, название которого стало составной частью названия семейства TCP/IP. Отметим, что протокол TCP (Transmission Control Protocol - протокол управления передачей) в точности соответствует спецификации транспортного уровня модели OSI/ISO, обеспечивая так называемый *виртуальный канал* между

взаимодействующими процессами. В отличие от него протокол UDP (User Datagram Protocol - дэйтаграммный протокол пользователя) обеспечивает ненадежное дейтаграммное соединение между взаимодействующими процессами, в котором могут происходить как потери пакетов, так и нарушения порядка следования пакетов. Как мы увидим в дальнейшем, для некоторых приложений такое ненадежное соединение оказывается более предпочтительным по сравнению с соединением через надежный, но требующий время для установления соединения и повторной доставки потерянных пакетов виртуальный канал.

Отметим, что программные модули всех рассмотренных выше уровней, включая транспортный реализованы в ядре ОС.

Поскольку более высокий уровень может взаимодействовать только с верхним из указанных уровней (транспортным), все сетевые ОС предоставляют пользователям совокупность системных вызовов (функций, выполняемых путём обращения к ядру ОС), обеспечивающих *интерфейсы транспортного уровня* для более высоких уровней. Эти интерфейсы рассматриваются в главе 6 настоящей книги.

Нетрудно заметить, что в уровневой модели протоколов TCP/IP отсутствуют сеансовый уровень и уровень представления данных. Если работающее на прикладном уровне приложение нуждается в услугах каких-либо из этих уровней, оно должно содержать внутри себя реализацию требуемых функций сеансового уровня и/или представления данных. В качестве примера приложения, реализующего “внутри себя” функции сеансового уровня, можно привести протокол защищенного удаленного терминала ssh, который в дополнение к своему основному функционалу может автоматически восстанавливать соединения, “разорвавшиеся” из-за ненадежной работы сети. В качестве примера приложения, реализующего некоторые функции уровня представления данных, можно привести протокол пересылки файлов ftp, предоставляющий возможности автоматического “приведения” формата пересылаемых символьных файлов к формату, используемому в операционной системе принимающего компьютера.

Отметим, что среди великого множества протоколов, работающих на прикладном уровне TCP/IP, есть протоколы, имеющие иерархическую структуру. В качестве примера приведем протокол сетевой файловой системы NFS, пользующийся услугами как протокола транспортного уровня UDP, так и прикладного протокола RPC (Remote Procedure Call - удаленный вызов процедур), работающего над транспортным протоколом TCP.

Общая схема взаимодействия протоколов семейства TCP/IP представлена на рис. 0.2. Фрагментами этого рисунка мы будем не раз пользоваться при рассмотрении различных протоколов тех или иных уровней.

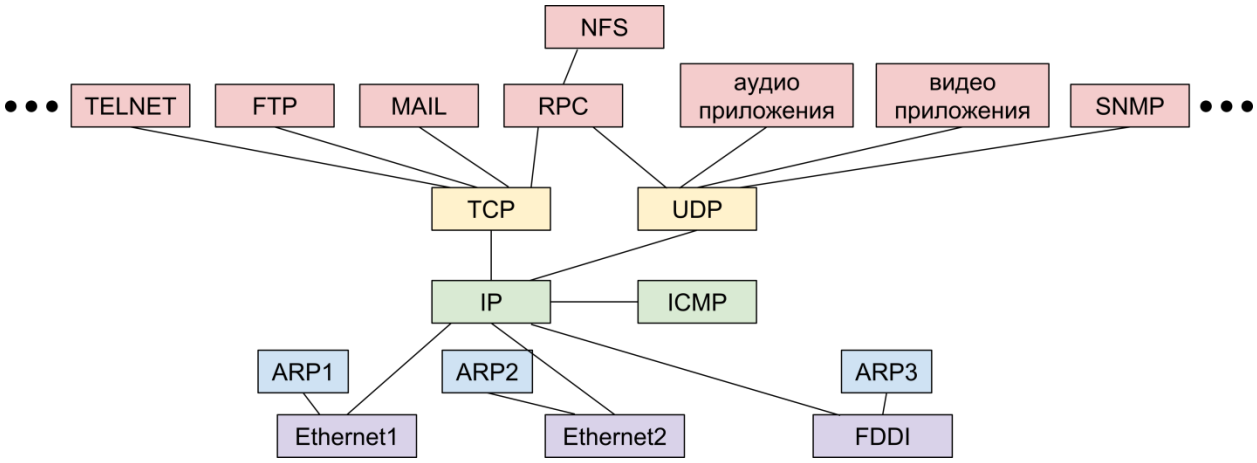

Рис. 0.2. Схема взаимодействия протоколов семейства TCP/IP

При отправлении пакетов данных в сеть с передающей стороны выполняется мультиплексирование (объединение) пакетов параллельно работающих протоколов в единый поток пакетов, пересылаемый через сеть передачи данных. На принимающем компьютере выполняется обратная процедура демультиплексирования (выделения пакетов, предназначенных конкретным протоколам), в ходе которой для каждого пакета строится цепочка протокольных модулей, через которую этот пакет последовательно поднимается по уровням. При построении этой цепочки используется принцип стека, на дно которого помещается драйверный модуль того интерфейса, через который поступил пакет. Затем модули добавляются в цепочку по принципу стека: при обработке пакета на текущем уровне принимается решение о том, какому модулю более высокого уровня адресован пакет (как это делается мы узнаем чуть позже), этот модуль заносится на верхушку стека и этому модулю передается пакет от расположенного “под ним” модуля. При взаимодействии двух конкретных приложений содержимое стека протоколов, участвующих в передаче пакета адресату однозначно определяется взаимодействующими приложениями. Поэтому сетевые специалисты зачастую говорят, что такие-то приложения взаимодействуют с использованием такого-то *стека протоколов*.

Рассмотрим несколько подробнее порядок пересылки информации через компьютерную сеть. Пусть браузер на клиенте, подсоединенном к интернету через WiMax-роутер, отправляет запрос к веб-серверу. В этой пересылке участвуют несколько промежуточных сетевых устройств: WiMax-устройства, маршрутизаторы провайдеров. На каждом из этих маршрутизаторов принятый пакет поднимается по стеку протоколов до модуля протокола IP и затем отправляется вниз по стеку протоколов выбранного для дальнейшей передачи пакета сетевого интерфейса. И лишь на конечных компьютерах информация проходит полный стек протоколов, спускаясь с вершины этого стека к его дну, на компьютере-источнике данных и, поднимаясь от дна стека к его вершине, на компьютере-получателе данных. Рассмотренная картина изображена на рис. 0.3.

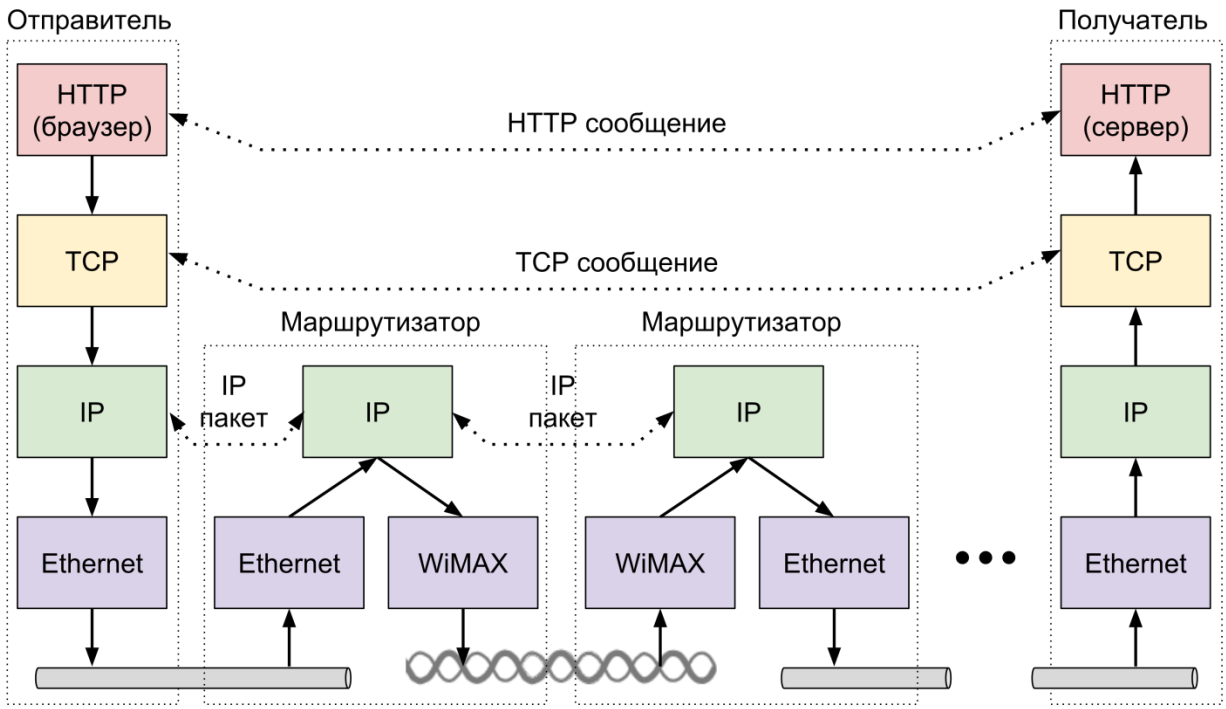

Рис. 0.3 Схема передачи через сеть информации между парой процессов

А теперь рассмотрим, каким преобразованиям подвергается в процессе ее пересылки информация, “путешествующая” через сеть между парой приложений. Приложение-отправитель направляет получателю некоторое *прикладное сообщение,* путем отправки его через интерфейс с протоколом транспортного уровня. Этот протокол инкапсулирует прикладное сообщение в новый пакет, называемый *транспортным сообщением* и содержащий кроме информации прикладного сообщения дополнительный заголовок транспортного уровня (ЗагТУ), содержащий, в частности, информацию о получателе (его компьютере и принимающем данные программном процессе), номере пакета и некоторую другую информацию. Сформированное таким образом транспортное сообщение передается на сетевой уровень, подвергаясь при этом очередной инкапсуляции в *IP-пакет*, содержащий в дополнение к полученному “сверху” транспортному сообщению заголовок IP-пакета (IP-заг), включающем адресную информацию, тип протокола транспортного уровня (TCP или UDP) и некоторые другие сведения. И, наконец, переданный IP-модулем драйверу IP-пакет инкапсулируется в пакет физического уровня, называемый *кадром* (frame) и пересылаемый через среду передачи данных. В ходе этой инкапсуляции в начало кадра добавляется его заголовок (ЗагК), а в конец - контрольная сумма всех слов содержания кадра. В заголовок кадра указывается адресная информация физического уровня, сведения о протоколе, чей пакет инкапсулирован в кадр (ARP, IP, ICMP) и некоторые другие сведения. Отметим, что транспортные сообщения, не помещающиеся в один IP-пакет, могут подвергаться разбиению на части, называемому сегментацией, а IP-пакеты, не помещающиеся в один кадр — аналогичному разбиению на части, называемому фрагментацией IP-пакетов.

Сформированные кадры пересылаются через среду передачи получателю, определяемому адресной информацией и при приеме этой информации автоматически

вычисляется контрольная сумма всех слов содержимого кадра. Полученное значение сравнивается с контрольной суммой из конца кадра, и при несовпадении этих значений кадр выбрасывается. Если же контрольные суммы совпали, начинается последовательная декапсуляция пакетов и их демультиплексирование — направление тому протоколу верхнего уровня, информация о котором содержится в заголовке пакета текущего уровня. В ходе “путешествия” наверх при необходимости выполняются требуемые десегментация и дефрагментация. “Путешествие” заканчивается прибытием полностью декапсулированного прикладного сообщения получающему его приложению. Этот процесс иллюстрируется на рис. 0.4.

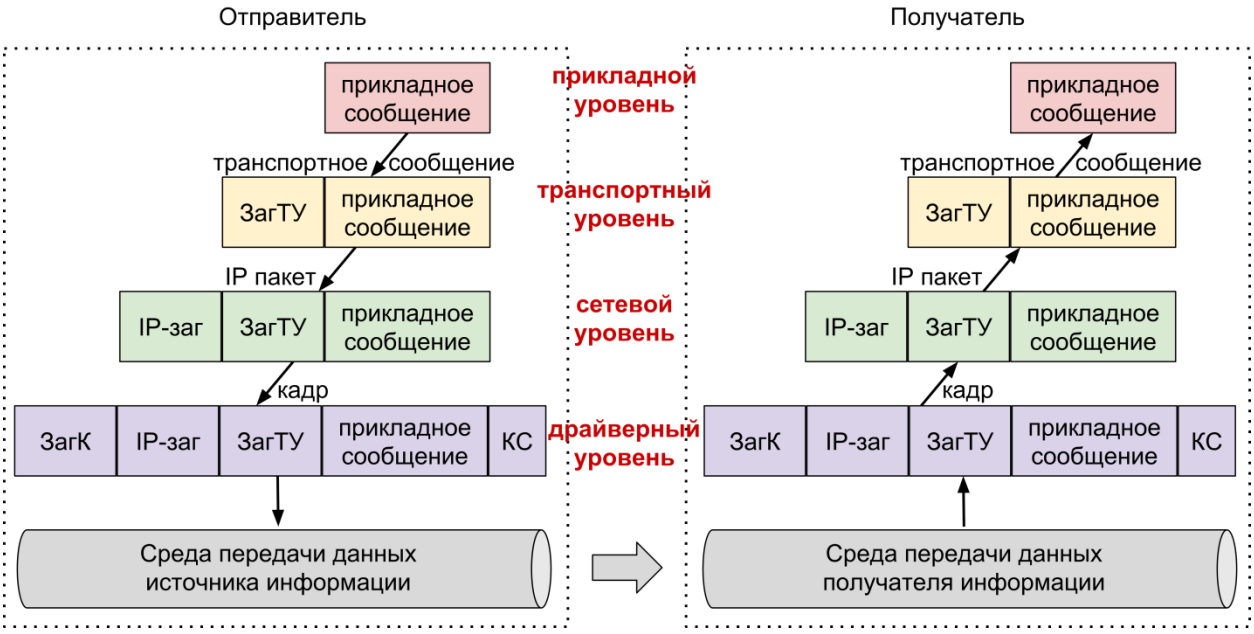

Рис. 0.4. Преобразование пакетов данных при пересылке через стеки протоколов

Рассмотренная картина может быть пояснена простой аналогией отправки писем по обычной (не электронной) почте. Отправитель письма на листе бумаги пишет содержание письма (аналог прикладного сообщения), которое он хочет направить адресату. Для того, чтобы сделать это, он обращается к следующему уровню (уровню почтового ящика), предварительно запечатав письмо в конверт и написав на нем адреса отправителя и получателя письма. Далее начинают работать уровни, невидимые для конечных пользователей почты. Из почтового ящика письма поступают в почтовые отделения, в которых они сортируются по индексам адресов получателей, упаковываются в ящики, предназначенные для отправки на определенных поездах и самолетах (чем не различные среды передачи?), на которых пишется соответствующая адресная информация, и доставляются в пункты назначений. Затем содержимое ящиков сортируется по отделениям связи, из которых письма поштучно приносятся в почтовые ящики получателей. И, наконец, получатель вынимает письмо из своего почтового ящика, открывает конверт и получает адресованное ему послание. Разумеется, что приведенная аналогия не идеально точна (письма и ящики с письмами не разрезаются на части и не склеиваются потом воедино). Тем не менее, на наш взгляд, она помогает понять суть происходящих процессов.