ФЕДЕРАЛЬНОЕ АГЕНТСТВО ВОЗДУШНОГО ТРАНСПОРТА РФ
Федеральное государственное образовательное учреждение высшего профессионального образования
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ГРАЖДАНСКОЙ АВИАЦИИ
Кафедра вычислительных машин, комплексов, систем и сетей

Н.И. РОМАНЧЕВА

СОВРЕМЕННЫЕ ИНТЕРНЕТ -ТЕХНОЛОГИИ

УЧЕБНОЕ ПОСОБИЕ

Москва- 2007

Печатается по решению редакционно-издательского совета Московского государственного технического университета ГА

Рецензенты: д-р. техн. наук А.В. Балдин (МГТТУ им. Н. Баумана)

д-р. техн. наук, доц. А.А. Егорова (МГТУ ГА)

Романчева Н.И.

Современные Интернет-технологии: Учебное пособие. - М.: МГТУ ГА, 2007.- 108 с.

Учебное пособие содержит материал второй части учебной дисциплины «Интернет-технологии в ГА», шифр ОПД.В.02, в которой рассматриваются современные Интернет-технологии, в том числе технологии передачи речи по IP-сетям и .NET. В конце каждого раздела учебного пособия приводятся контрольные вопросы, в приложении приведены тестовые вопросы.

Учебное пособие издается в соответствии с учебным планом для студентов специальности 230101 дневного обучения.

Рассмотрено и одобрено 20.03.2007г. на заседаниях кафедры ВМКСС и методического Совета по специальности 230101 20.03.2007 г.

С О Д Е Р Ж А Н И Е

ВВЕДЕНИЕ

В наши дни глобальной компьютеризации широкое внедрение сетевых коммуникаций способствует развитию информационных технологий. Ни одно предприятие не может существовать без сбыта своей продукции и услуг. В настоящее время уже понятно, что Интернет-технологии могут существенно повысить эффективность коммерческой деятельности в любой отрасли. Их растущая актуальность для всего современного информационного общества не могла ни отразиться и на деятельности предприятий гражданской авиации.

До недавнего времени сети с коммутацией каналов (телефонные сети) и сети с коммутацией пакетов (IP-сети) существовали практически независимо друг от друга и использовались для различных целей. Телефонные сети использовались только для передачи голосовой информации, а IP-сети соответствеено для передачи данных.

Обычные телефонные звонки требуют разветвленной сети связи телефонных станций, связанных закрепленными телефонными линиями, подвода волоконно-оптических кабелей и спутников связи. Выделенное подключение телефонной станции также имеет много избыточной производительности или времени простоя в течение речевого сеанса. Рассматриваемая в данном учебном пособии IP-телефония использует технологию сжатия голосовых сигналов и полностью использует емкость телефонных линий. Поэтому пакеты данных от разных запросов, и даже различные их типы, могут перемещаться по одной и той же линии в одно и тоже время.

Используя IP-сеть, можно обмениваться цифровой информацией для пересылки голосовых или факсимильных сообщений между двумя компьютерами в режиме реального времени. Применение Интернет позволит реализовать данную службу в глобальном масштабе. Можно сделать вывод, что за IP-телефонией, как за Web, последует новая волна IP-сервисов. Общей чертой этих сервисов будет осуществление IP-коммуникаций в реальном времени.

На сегодняшний день существует множество технологий и протоколов, позволяющих без труда соединять элементы распределенных систем между собой. Одна из наиболее известных технологий – DCOM, позволяющая эффективно осуществлять RPC-вызовы, передавать и принимать данные, распределять нагрузку между несколькими back-end серверами. Однако у систем, построенных на DCOM, есть очень важный недостаток, затрудняющий взаимодействие уровня представления и уровня бизнес-логики через Internet. Альтернативой DCOM при построении распределенных систем может служить использование Web-интерфейса на основе ASP.

Сегодня нельзя рассчитывать, что отдельное программное приложение будет самодостаточным - для эффективной работы ему, скорее всего, понадобятся данные или функциональность других информационных систем, расположенных, возможно, на других компьютерах и в других организациях. Интернет существенно поменял распространенные представления о внедрении и использовании информационных технологий. Можно выбрать самые разные модели использования информационных технологий - от создания собственной полномасштабной корпоративной информационной системы до аренды необходимых приложений в режиме ASP (Application Service Providing). Более того, .NET позволяет «смешивать» эти подходы, использовать их параллельно и переходить от одного к другому по мере надобности.

Microsoft .NET позволяет компоненту найти необходимые ресурсы на отдельном компьютере, в локальной сети или в Интернете и использовать их с минимальными накладными расходами. .NET также гарантирует компонентам:

-транзакционность (будут выполнены либо все связанные операции, либо ни одной);

-масштабирование и балансировку (копии компонентов и ресурсы для них выделяются и распределяются по серверам автоматически);

-безопасность (некорректная работа компонентов или связанных с ними внешних систем не может повредить среду .NET).

Кроме того, самые сложные понятия и действия можно представить в наглядной и удобной форме, понятной большинству пользователей без специального обучения.

В данном учебном пособии рассмотрены современные технологии передачи речи по IP-сетям и полнофункциональная технологическая платформа для поддержки корпоративных приложений в среде Интернет .NET.

В первой части в разделе «Транспортные технологии пакетной коммутации» рассматривает особенности передачи речевой информации, основные термины, протоколы, а также место IP- телефонии среди близких ей решений. Приводятся основные элементы и варианты построения IP-телефонных систем

В разделе «Стандарт ITU H.323» приведены различные подходы к построению сетей IP- телефонии: архитектура сети H.323, виды конференций в сетях H.323, а также сценарий установления соединения.

В следующих двух разделах первой части расмотрены сети на базе протоколов SIP, MGCP и MEGACO. Проводится сравнение подходов к построению сетей IP-телефонии.

В разделе «Технология MPLS» рассматриваются элементы архитектуры и преимущества технологии MPLS, схема коммутации и построение коммутируемого маршрута с использованием стека меток. Не остались без внимания и проблемы перехода к мультисервисным сетям.

Вторая часть учебного пособия «Технология .NET» рассматривает компоненты, средства, инструменты, спецификации для построения и сопровождения приложений на базе технологии .NET. Приводятся примеры современных решений. Рассмотрен протокол SOAP, использование WSDL- и WSML-файлов.

В разделе «Сервисы и продукты на платформе .Net» описаны общеязыковая среда выполнения Common Language Runtime (CLR) и набор базовых классов, радикально упрощающих разработку крупномасштабных приложений. Рассмотрен механизм языковой интеграции и приведено описание компонентной и корпоративной разработки с использованием .NET Framework. Кроме того, обсуждаются основы ключевых технологий .NET: работа с данными (ADO.NET) и XML, веб-службы (Web Services), веб-формы (Web Forms, ASP.NET)

В приложении данного учебного пособия приведен итоговый тест, который позволит проконтролировать уствоение материала.

Использование современных Интернет-технологий позволяет повысить производительности труда сотрудников, снизить затраты на обучение персонала и сопровождение информационной системы. Все это дает возможность быстрее освоить новые бизнесы, своего рода пропуск предприятия на новые рынки и сохранение инвестиций в ранее установленное ПО и оборудование.

Данное учебное пособие предназначено для студентов специальности 230101 и других специальностей для изучения вопросов, связанных с использованием современных Интернет-технологий в своей практической деятельности.

ЧАСТЬ 1. ТЕХНОЛОГИИ ПЕРЕДАЧИ РЕЧИ ПО IP-СЕТЯМ

Раздел 1. Транспортные технологии пакетной коммутации

1.1 **Место IP- телефонии среди близких ей решений**

До недавнего времени сети с коммутацией каналов (телефонные сети) и сети с коммутацией пакетов (IP-сети) существовали практически независимо друг от друга и использовались для различных целей. Телефонные сети использовались только для передачи голосовой информации, а IP-сети - для передачи данных.

Интерес к вопросам передачи речи по сетям передачи данных с пакетной коммутацией (VoIP) возник с тех пор, как стало очевидным, что коммутация каналов более не в состоянии удовлетворять растущие потребности рынка, обеспечивать активное внедрение новых и дополнительных услуг, снижение удельных затрат на расширение сетей.

В настоящее время стремительными темпами развиваются, наряду с традиционными методами импульсно-кодовой, дельта- и других цифровых видов модуляции, методы речеобразования с существенным сокращением избыточности, методы статистического уплотнения цифровых каналов, пакетов передачи и коммутации с использованием технологий FR, IP, ATM и др.

Транспортная технология ATM уже несколько лет успешно используется в магистральных сетях общего пользования и в корпоративных сетях, а сейчас ее начинают активно использовать и для высокоскоростного доступа по каналам xDSL (для небольших офисов) и SDH/Sonet (для крупных предприятий). Главные преимущества этой технологии - ее зрелость, надежность и наличие развитых средств эксплуатационного управления сетью. В ней имеются непревзойденные по своей эффективности механизмы управления качеством обслуживания и контроля использования сетевых ресурсов. Однако ограниченная распространенность и высокая стоимость оборудования не позволяют считать ATM лучшим выбором для организации сквозных телефонных соединений от одного конечного узла до другого.

Технологии Frame Relay суждено было сыграть в пакетной телефонии ту же роль, что и квазиэлектронным АТС в телефонии с коммутацией каналов: они показали пример эффективной программно управляемой техники, но имели ограниченные возможности дальнейшего развития. Пользователями недорогих услуг Frame Relay, обеспечивающих вполне предсказуемую производительность, стали многие корпоративные сети, и большинство из них вполне довольны своим выбором. В краткосрочной перспективе технология передачи речи no Frame Relay будет вполне эффективна для организации мультисервисного доступа и каналов дальней связи. Как

правило, на практике используются некоммутируемые соединения в режиме точка-точка, и сети Frame Relay распространены незначительно.

Широкополосная система доступа ADSL (Asymmetric Digital Subscriber Line)– это технология постоянного, некоммутируемого соединения абонента, одновременной передачи голоса и данных по обычным “медным” телефонным каналам. Передача данных по ADSL технологии производится по тому же кабелю, к которому подключен телефон абонента.

Основными преимуществами использования данной технологии связи являются:

- широкий комплекс телекоммуникационных услуг предоставляется одним оператором, что упрощает процедуры оплаты счетов и эксплуатацию оборудования;

- возможность надежной бесперебойной работы в сети Интернет, так как соединение по ADSL является более надежным, чем DialUp, к тому же отсутствует необходимость дозваниваться до провайдера для установки соединения.

- высокая скорость доступа и надежность связи по доступной цене:

- возможность организации нескольких дополнительных телефонных линий на существующей медной паре;

- возможность переноса существующего ADSL канала на новую телефонную линию с сохранением всех телефонных номеров, подключенных с использованием данного канала.

Цифровые сети с интеграцией услуг (ISDN) можно эффективно использовать для решения широкого касса задач по передаче информации в различных областях, например в телефонии. Цифровая технология обеспечивает высокое качество передачи речи и широкий набор дополнительных услуг для телефонии.

Две главные характеристики этой услуги - высокие скорости и невысокие цены. ISDN идеально подходит для малого и среднего бизнеса, но нередко используется частными лицами и крупными компаниями. Это экономичное решение для домашнего офиса позволяет организовать подключение нескольких устройств (телефон, факс) к одной линии ISDN. При этом обеспечивается одновременная работа с телефонией и факсимильной связью.

Основные преимущества по ISDN в сравнении с обычными телефонными линиями и каналами передачи данных:

- улучшенная помехоустойчивость передачи речи благодаря цифровому кодированию, вывод на дисплей дополнительной информации;

- передача страницы факса формата А4 менее, чем за 10 секунд, в 4 раза более высокая разрешающая способность;

- ISDN-соединение устанавливается практически мгновенно.

Несмотря на то, что IP и Ethernet с каждым годом все больше вытесняют традиционные сети телефонной связи, существует достаточно большое количество работающих сетей, которые требуют развития и модернизации. Многим компаниям экономически не целесообразно заменять весь парк традиционного TDM оборудования на IP/Ethernet VoIP-шлюзы, но развитие сети необходимо. Решение было найдено - ряд компаний разработал методику передачи TDM трафика по IP/Ethernet сетям. Соответственно, компании, строя новые сети на основе IP/Ethernet, теперь имеют возможность использовать уже существующие TDM сети, в качестве транспорта для них используя уже IP/Ethernet. Данная технология получила название TDMoIP.

Родоначальником данного вида решений была израильская компания RAD DataComunications и ее устройства семейства IPMux. Данные устройства позволяют упаковывать данные из потоков E1 в Ethernet кадры, включая сигнализацию, голосовые таймслоты, синхронизацию. Основной проблемой была передача синхронизации, но компании удалось разработать решение - синхронизация передается прозрачно через Ethernet - сеть.

Ряд компаний, в том числе Yoda, Redux разработали устройства, более экономично использующие полосу пропускания Ethernet для передачи E1, в отличие от IPMux. Это дало возможность использовать технологию передачи синхронных данных (TDM) в сетях радиодоступа Wi-Fi, в которых IPMux работал неустойчиво из-за большого количества потерянных пакетов, задержек и других особенностей Wi-Fi.

Технологии передачи речи по сетям IP (VoIP) и Frame Relay (VoFR) реализуют другой путь – интеграцию телефонов в существующие компьютерные сети. Пакетная телефония предоставляет наряду с традиционными услугами ТфОП (ожидание вызова, многоканальные номера, конференцсвязь, и т.д.) новые возможности:

- повышается эффективность использования полосы пропускания каналов за счет эффективных алгоритмов сжатия данных;
- расширяется управление сетью передачи всех типов трафика - речи, данных и видео;
- поддержка широко применяемых протоколов;
- предоставление конечным пользователям выбор способа телефонной связи, позволяя сократить расходы на междугородные звонки.

В настоящее время технология VoIP признана во всем мире наиболее прогрессивным методом передачи голосовой информации. При использовании технологии VoIP данные передаются по выделенным цифровым каналам. Эти каналы

имеют необходимую пропускную способность и используются только для передачи голосового трафика, что обеспечивает качественную голосовую связь.Телефонные сети, построенные на технологиях Voice over IP(VoIP), в несколько раз дешевле своих традиционных собратьев - сетей TDM. Занимая одну и ту же полосу пропускания, пользователь VoIP получает в два раза больше возможностей, а соответственно и услуг, чем при традиционной TDM-телефонии, т.е. при одинаковой арендной стоимости одной и той же полосы пропускания пользователь экономит минимум в 2 раза больше средств на связь. Это лишь один из плюсов использования VoIP технологий.

Благодаря компьютерно-телефонной интеграции (CTI) в последнее десятилетие в развитии телефонных сетей общего пользования (ТфОП) наметился очевидный прогресс, однако стоимость внедрения современных компьютерных технологий в сетях с коммутацией каналов слишком высока. Результаты сравнительного анализа, проведенного в [ ], в графической форме приведены на рисунке 1.1.

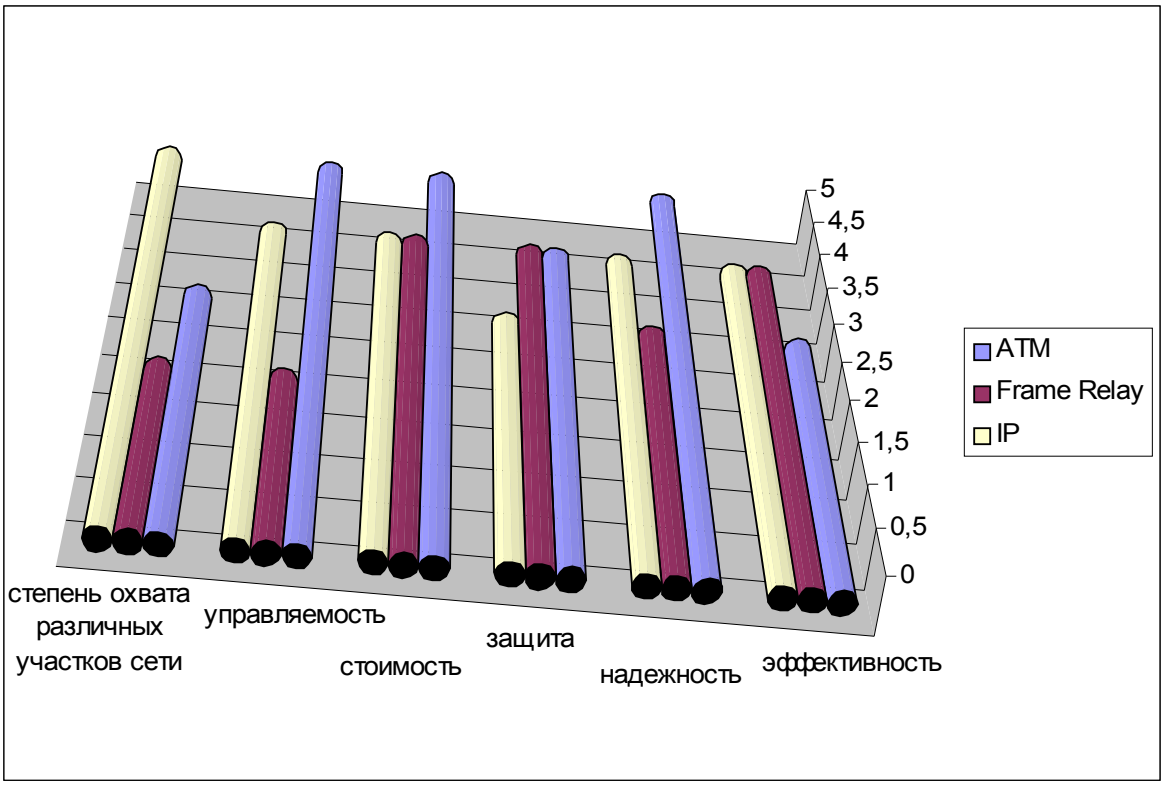


Рисунок 1.1 – Передача речи с использованием транспортных технологий

Основные технологии пакетной передачи речи - Frame Relay, ATM и маршрутизация пакетов IP - различаются эффективностью использования каналов связи, степенью охвата разных участков сети, надежностью, управляемостью, защитой информации и доступа, а также стоимостью.

Технология передачи речевой информации по сетям с маршрутизацией пакетов IP привлекает, в первую очередь, своей универсальностью - речь может быть преобразована в поток IP-пакетов в любой точке сетевой инфраструктуры: на магистрали сети оператора, на границе территориально распределенной сети, в корпоративной сети и даже непосредственно в терминале конечного пользователя. В конце концов, она станет наиболее широко распространенной технологией пакетной телефонии, поскольку способна охватить все сегменты рынка, будучи при этом хорошо адаптируемой к новым условиям применения. Несмотря на универсальность протокола IP, внедрение систем IP-телефонии сдерживается тем, что многие операторы считают их недостаточно надежными, плохо управляемыми и не очень эффективными. В расчете на порт стоимость систем IP-телефонии находится на уровне (или немного ниже) стоимости систем Frame Relay, и заведомо ниже стоимости оборудования ATM.

1.2 Основные понятия

Возможность передачи голосовых сообщений через сеть с пакетной коммутацией впервые была реализована в 1993 году и получила название *VoIP – Voice over Internet Protocol* - технология обмена голосовыми сообщениями по сетям Интернет и другим сетям, основанным на протоколе IP. Эта аббревиатура также используется как аналог терминов «IP-телефония», «Интернет –телефония», «цифровой телефон».

При единой технической базе IP-телефония сегодня разделяется в соответствии с приложениями на разные направления, поэтому данный термин может иметь различный подтекст в зависимости от того, к какому сегменту рынка он относится.

Приведем несколько определений, которые встречаются в литературе. Под *IP-телефонией* понимается технология, позволяющая использовать Интернет или любую другую IP-сеть в качестве средства организации и ведения международных и междугородних телефонных переговоров, передачи факсов, изображения в режиме реального времени [18].

В [21] под *IP-телефонией* понимается специальная область телефонной связи, интегрирующая методы и средства цифровой обработки сигналов, речевых технологий, управления вычислительными ресурсами на базе высоких технологий.

Сегодня *IP- телефония* является одним из наиболее сложных и системных приложений компьютерной телефонии (CTI), хотя часто эти два понятия ошибочно

отождествляются. IP-телефония позволяет использовать сети передачи данных с IP-протоколом в качестве носителя телефонных разговоров.

Интернет-телефония - технология, которая используется в Интернет для передачи речевых сигналов, ее можно рассматривать как частный случай IP-телефонии.

CTI (Computer Telephony Integration) - это технология, осуществляющая слияние двух независимо существующих телефонных и компьютерных систем.

Компьютерная телефония - отрасль, специализирующаяся на приложении компьютерного интеллекта к осуществлению и приему телефонных вызовов, а также, к другим сложным взаимодействиям. В первую очередь это реализация голосового соединения по каналам вычислительных сетей. К компьютерной телефонии относится также интерактивная обработка голоса, организация голосовой почты, алгоритмы распознавания речи и преобразования текста в речь. CTI позволяет использовать все преимущества компьютерной идеологии (стандарты, гибкость, совместимость, удобный и привычный интерфейс и т.д.) для управления телефонными соединениями. Основным достоинством CTI является открытость систем, т.е. вся компьютерная телефония основана на стандартах и, следовательно, системы CTI легко модифицируются, расширяются; достигается максимальная совместимость компонентов. Эти преимущества оказались настолько очевидными, что это обусловило быстрое развитие и широкое применение систем CTI

При сравнении технической литературы по CTI, IP-телефонии или VoIP можно сделать вывод, что данные технологии имеют одну и туже технологическую базу при наличии разных целей. Кроме того, оба направления так сильно интегрированы, что в зависимости от точки зрения, IP-телефония может быть частью CTI, и наоборот. Взаимосвязь данных технологий представлена на рисунке 1.2.

Как видно из рисунка 1.2, Интернет-телефония является частным случаем IP-телефонии, здесь в качестве линий передачи используются обычные каналы Интернет. При разговоре голосовые сигналы (слова, которые произносит человек) преобразуются в сжатые пакеты данных. После эти пакеты данных посылаются через Интернет другой стороне. Когда пакеты данных достигают адресата, они декодируются в голосовые сигналы оригинала.

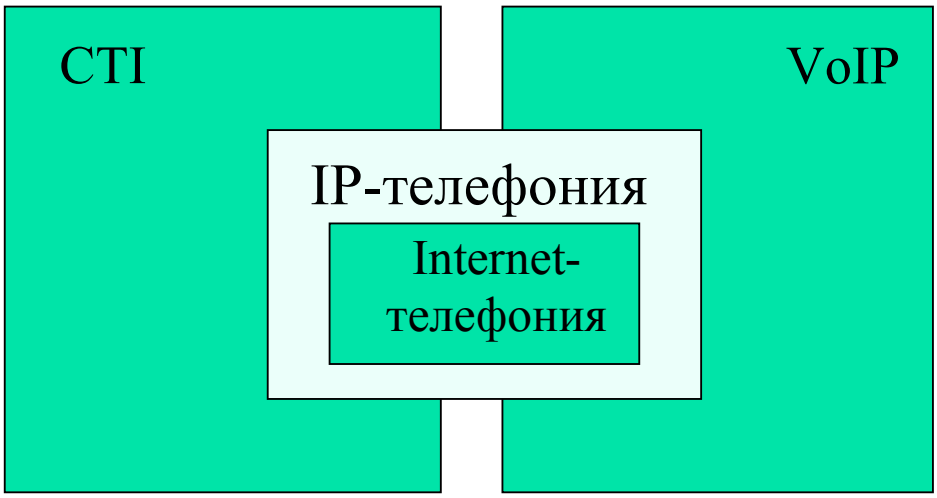


Рисунок 1.2 – Взаимосвязь технологий

В отношении сервисов и технологий между IP-телефонией и VoIP нет никакой разницы. Различные производители могут предпочитать один или другой термин, либо использовать их в равной степени. С точки же зрения решений термин «IP-телефония» более содержательный, так как он реализуется не только на уровне каналов передачи (как глобальных, так и локальных), но и на уровне абонентского оборудования.

Основными преимуществами технологии VoIP является сокращение требуемой полосы пропускания, что обеспечивается учётом статистических характеристик речевого трафика: блокировкой передачи пауз (диалоговых, слоговых, смысловых и др.), которые могут составлять до 40-50 % времени занятия канала передачи; высокой избыточностью речевого сигнала и его сжатием (без потери качества при восстановлении) до уровня 20-40 % исходного сигнала.

С другой стороны, трафик VoIP критичен к задержкам пакетов в сети, но обладает устойчивостью к потерям отдельных пакетов.

В соответствии с этим при передаче телефонного трафика по технологии VoIP должны учитываться жёсткие требования стандарта ISO9000 к качеству услуг, характеризующие качество установления соединения и качество соединения. Основным показателем качества в первом случае является время установления соединения. Во втором случае показателями качества являются сквозные (воспринимаемые пользователем) задержки и качество воспринимаемой речи.

Архитектура технологии передачи речи может быть упрощенно представлена в виде двух плоскостей. Нижняя плоскость - это базовая сеть с маршрутизацией пакетов IP, верхняя плоскость - это открытая архитектура управления обслуживанием вызовов.

Нижняя плоскость представляет собой комбинацию известных протоколов RTP/ UDP/IP - транспортный механизм для речевого трафика в сетях с маршрутизацией

пакетов IP. Рекомендации ITU-T допускают задержки в одном направлении не превышающие 150 мс.

Верхняя плоскость выполняет управление обслуживанием запросов связи. Управление обслуживанием вызова предусматривает принятие решений о том, куда вызов должен быть направлен, и каким образом должно быть установлено соединение между абонентами. Инструмент такого управления в ТфОП - телефонные системы сигнализации. В сетях с коммутацией пакетов ситуация более сложна. Сеть с мар-шрутизацией пакетов IP принципиально поддерживает одновременно целый ряд разнообразных протоколов маршрутизации. Такими протоколами на сегодня являются: RIP (Routing Information Protocol), IS-IS (Intermediate System-to-Intermediate System), OSPF (Open Shortest Path First), BGP (Border Gateway Protocol) и другие. Для IP-телефонии также разработан целый ряд стандартов и протоколов.

1.3 Стандарты IP -телефонии

Стандарты являются критическим фактором для мира IP-телефонии. IP-телефония должна обладать возможностью поддерживать совместную работу и обеспечивать информационную прозрачность с множеством стандартов связи, принятых в разных странах мира, также необходимо взаимодействие протоколов верхних уровней и приложений, начисление платы и др. Одна из наиболее важных областей стандартизации - протокол обмена сообщениями в IP-телефонии.

Для IP-телефонии чаще всего используется стандарты на основе H.323 и SIP, определяющие передачу видео- и аудио-информации по сетям с негарантированным качеством услуг, таким как Ethernet и IP.

Наиболее распространенным является протокол, специфицированный в рекомендации H.323 ITU-T. Он стал применяться раньше других протоколов, которых, до внедрения H.323 вообще не существовало. Протокол H.323 не привязан к какому-либо конкретному типу сети, однако H.323 нашел применение преимущественно именно в сетях на базе IP. Этот стандарт описывает несколько элементов, в том числе аудио- и видеокодеки (кодеры/декодеры), коммуникационные протоколы и синхронизацию пакетов. H.323 включает в себя:

- стандарты на видео кодер/декодеры;
- стандарты на голосовые кодер/декодеры;
- стандарты на общедоступные приложения;
- стандарты на управление вызовами;
- стандарты на управление системой.

Стандарты на видео кодер-декодеры не требуются для обработки телефонных звонков, но существуют внутри той же системы стандартов.

Технические требования к голосовым кодерам включают требования, такие как: малая полоса пропускания (8 kbit/s или меньше); высокое качество голоса; небольшие задержки; возможность реконструкции потерянных пакетов.

При передаче в режиме реального времени до 30% пакетов могут потеряться или опоздать. Приложение IP-телефонии должно возместить нехватку пакетов, восстановив потерянные данные. Сам алгоритм кодировки также оказывает влияние на восстановление данных. Сложные алгоритмы увеличивают стоимость необходимого оборудования. Наиболее популярным алгоритмом кодирования является G.723.1, основой которого являются методы сжатия речи MP-MLQ (Multipulse Maximum Likelihood Quantization) и ACELP. Они позволяет добиться весьма существенного сжатия речи при сохранении достаточно высокого качества воспроизведения.

Еще одна особенность состоит в том, что системы IP-телефонии должны иметь возможность поддерживать разные кодеры и добавлять новые по необходимости. Протокол H.323 был первоначально разработан для локальных вычислительных сетей, так что переменная ширина полосы частот и время задержки Интернет уменьшают полезность некоторых элементов H.323. По умолчанию голосовым кодер-декодером в стандарте H.323 является G.711. Однако ширина полосы частот в 64 kbps, требуемая в G.711, неприемлема при использовании в Интернет, т.к. большинство пользователей Интернета имеет канал заведомо меньшей ширины.

Особое положение занимает подгруппа стандартов для контроля вызовов, в том числе для установления соединения, управления потоками, контроля доступа, передачи служебных сообщений и т. п. Ключевым компонентом этой подгруппы является протокол управляющего канала H.245 для передачи разного рода служебной информации во время сеансов H.323. Он применяется для согласования конечными точками взаимоприемлемых параметров, открытия и закрытия логических каналов, передачи сообщений для управления потоками и других необходимых команд и запросов. Соединение же между двумя устройствами H.323 устанавливается и закрывается с помощью другого протокола данной подгруппы - протокола сигнализации вызова Q.931, а регистрация и контроль доступа, статусом устройств H. 323 осуществляются посредством третьего протокола этой подгруппы - RAS (Registration Admission Status).

Стек протоколов H.323, приведен на рисунке 1.3.

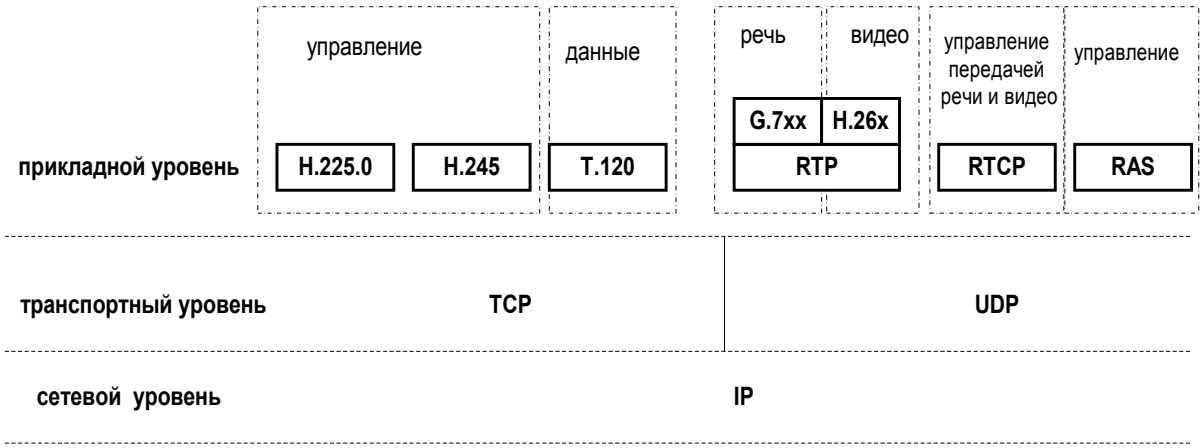


Рисунок 1.3 - Стек протоколов H.323

Для уменьшения стоимости шлюзов за счет реализации функций интеллектуальной обработки вызова в централизованном оборудовании с 1998 года разрабатывался протокол SGCP. В конце 1998 года рабочая группа MEGACO комитета IETF разработала протокол MGCP, базирующийся, в основном, на протоколе SGCP, но с некоторыми добавлениями в части OAM&P.

Рабочая группа MEGACO совершенствовала протокол управления шлюзами и разработала более функциональный, чем MGCP, протокол MEGACO. Его адаптированный к H.323 вариант (под названием Gateway Control Protocol) ITU-T предлагает в рекомендации H.248.

Другой протокол плоскости управления обслуживанием вызова - протокол инициирования сеанса SIP. Существенно, что SIP не является протоколом для передачи голосовых данных, так как он не связан с передачей данных какого-то определенного вида - в качестве передаваемой информации может выступать: речь (как в случае IP-телефонии), и музыка, и видео, и, например, текст. Кроме того, данный протокол позволяет организовывать сеансы коллективной работы над документами, какие поддерживаются в MS Exchange или Lotus Notes).

Тип данных определяется отдельным протоколом описания сеанса SDP (Session Description Protocol), который позволяет менять параметры сеанса по ходу обмена данными.

Очевидно, SIP существенно лучше, чем H.323, согласуется с пониманием IP-телефонии как массового глобального IP-сервиса, так как он способен обслуживать любые коммуникации в реальном времени поверх протокола IP.

1.4 Протоколы RTP и RTCP

Доставка данных в реальном масштабе времени является одним из важнейших факторов в эволюции глобальных сетей, причем существующая сетевая структура Интернет мало пригодна для массового использования приложений, работающих в

реальном времени. Эта проблема может быть решена, например, переходом к технологии ATM или применением дополнительных механизмов в существующих сетях.

Приложения, обеспечивающие передачу речевой и видеоинформации, используют сервис транспортного уровня без установления соединений (например, UDP). При этом каждое приложение может обеспечивать формирование полезной нагрузки пакетов специфическим образом, включая необходимые для функционирования поля и данные. Однако, как показал приведенный в [18] анализ, данные разной природы (речь, видео) имеют общие особенности, которые требуют обеспечения вполне определенной функциональности при их передаче по сети. Это позволяет сформировать некий общий транспортный уровень, объединяющий функции, общие для потоковых данных разной природы, и используемый всеми соответствующими приложениями, придав протоколу этого уровня статус стандарта. Комитетом IETF был разработан протокол транспортировки информации в реальном времени - Real-time Transport Protocol (RTP), RFC-1889 и RFC-1890, который стал основой практически для всех приложений, связанных с интерактивной передачей речевой и видеоинформации по сети с маршрутизацией пакетов.

Характерные для IP-сетей временные задержки и вариация задержки пакетов (джиттер) могут серьезно исказить информацию, чувствительную к задержке, например, речь и видеоинформацию, сделав ее абсолютно непригодной для восприятия. Отметим, что вариация задержки пакетов гораздо сильнее влияет на субъективную оценку качества передачи, чем абсолютное значение задержки.

Именно протокол RTP позволяет компенсировать негативное влияние джиттера на качество речевой и видеоинформации. Однако протокол RTP не имеет собственных механизмов, гарантирующих своевременную доставку пакетов или другие параметры качества услуг (это осуществляют нижележащие протоколы). Кроме того, функции исправления ошибок и управления потоком, которые обычно предоставляют транспортные протоколы, данный протокол не обеспечивает.

Обычно протокол RTF используется «поверх» любого сетевого соединения и не предъявляет никаких требований к сети кроме пакетного режима работы. Такой подход позволяет использовать RTP как в существующих сетевых соединениях, например, в Интернет, поверх UDP или TCP, так и в перспективных сетях ATM, Frame Relay и даже в специфических соединениях типа ADSL, широковещательных кабельных и спутниковых каналах.

Существует несколько серьезных причин, по которым транспортный протокол TCP плохо подходит для передачи чувствительной к задержкам информации. Во-первых, это алгоритм надежной доставки пакетов. Пока отправитель повторно передаст пропавший пакет, получатель будет ждать, результатом чего может быть недопустимое увеличение задержки. Во-вторых, алгоритм управления при перегрузке в протоколе TCP далеко не оптимален для передачи речи и видеоинформации. При обнаружении потерь пакетов протокол TCP уменьшает размер окна, а затем будет его медленно увеличивать. Однако передача речевой и видеоинформации осуществляется на вполне определенных, фиксированных скоростях, которые нельзя мгновенно уменьшить, не ухудшив качество предоставляемых услуг. Правильной реакцией на перегрузку для информационных потоков этих типов было бы изменение метода кодирования, частоты видеокадров или размера видеоизображения.

Протокол RTP предусматривает индикацию типа полезной нагрузки и порядкового номера пакета в потоке, а также применение временных меток. Отправитель помечает каждый RTP-пакет временной меткой, получатель извлекает ее и вычисляет суммарную задержку. Разница в задержке разных пакетов позволяет определить джиттер и смягчить его влияние - все пакеты будут выдаваться приложению с одинаковой задержкой.

Главная особенность RTP - это вычисление средней задержки некоторого набора принятых пакетов и выдача их пользовательскому приложению с постоянной задержкой, равной этому среднему значению. Однако следует иметь в виду, что временная метка RTP соответствует моменту кодирования первого дискретного сигнала пакета. Поэтому, если RTP-пакет, например, с видеоинформацией, разбивается на блоки данных нижележащего уровня, то временная метка уже не будет соответствовать истинному времени их передачи, поскольку они перед передачей могут быть установлены в очередь.

На рисунке 1.4 представлен основной заголовок RTP-пакета, содержащий ряд полей, которые идентифицируют такие элементы, как формат пакета, порядковый номер, источник информации, границы и тип полезной нагрузки:

V - поле версии протокола, 2 бита. Текущая версия протокола - вторая.

P - поле заполнения, 1 бит. Сигнализирует о наличии заполнения в конце поля полезной нагрузки. Заполнение применяется, когда приложение требует, чтобы размер полезной нагрузки был кратен, например, 32 битам.

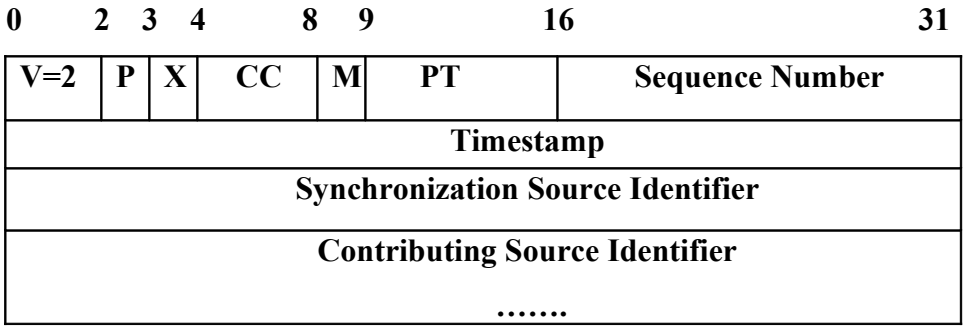


Рисунок 1.4 - Основной заголовок RTP -пакета

X - поле расширения заголовка, 1 бит. Служит для индикации того, что за основным заголовком следует дополнительный заголовок, используемый в экспериментальных расширениях протокола RTR

CC - поле отправителей, 4 бита. Содержит идентификаторы отправителей, чьи данные находятся в пакете, причем сами идентификаторы следуют за основным заголовком.

M - поле маркера, 1 бит. Обычно используется для указания границ потока данных. Смысл бита маркера зависит от типа полезной нагрузки. В случае передачи видеоинформации он определяет конец кадра, При передаче речевой информации маркер указывает начало периода активности после периода молчания.

PT - поле типа полезной нагрузки, 7 битов. Идентифицирует тип полезной нагрузки и формат данных, включая сжатие и шифрование, В стационарном состоянии отправитель использует только один тип полезной нагрузки в течение сеанса, но он может его изменить в ответ на изменение условий, если об этом сигнализирует протокол управления транспортировкой информации в реальном времени (Real-Time Transport Control Protocol).

Sequence Number - порядковый номер пакета, 16 битов. Каждый источник начинает нумеровать пакеты с произвольного номера, увеличиваемого затем на единицу с каждым переданным пакетом RTP. Псевдослучайное число, которое уникальным образом идентифицирует. Это позволяет обнаруживать потери пакетов и определять порядок пакетов с одинаковым временным штампом. Несколько последовательных пакетов могут иметь один и тот же штамп, если логически они порождены в один и тот же момент (например, пакеты, принадлежащие одному и тому же видеокадру).

Timestamp - временной штамп, 32 бита. Момент времени, в который был создан первый октет данных полезной нагрузки. Единицы, в которых время указывается в

этом поле, зависят от типа полезной нагрузки. Значение определяется по локальным часам отправителя.

Synchronization Source Identifier (SSRC) - поле идентификатора источника синхронизации, 32 бита. Источник в течение сеанса и не зависит от сетевого адреса. Это число играет важную роль при обработке порции данных, поступившей от одного источника.

Contributing Source Identifier (CSRC) - список полей идентификаторов источников, участвующих в создании RTP-пакета, 32 бита. Устройство смешивания информации (миксер) вставляет целый список SSRC идентификаторов источников, которые участвовали в построении данного RTP-пакета. Количество элементов в списке: от 0 до 15. Если число участников более 15, выбираются первые 15. Примером может служить речевая конференция, в которой передаются RTP-пакеты с речью всех участников - каждый со своим идентификатором SSRC. Они-то и образуют список идентификаторов CSRC. Вся конференция имеет общий идентификатор SSRC.

Доставка RTP-пакетов контролируется специальным протоколом RTCP (Real Time Control Protocol), RFC-1889.

Основной функцией протокола RTCP является организация обратной связи приемника с отправителем информации для отчета о качестве получаемых данных. Протокол RTCP передает сведения (как от приемника, так и от отправителя) о числе переданных и потерянных пакетов, значении джиттера, задержке и т.д. Эта информация может быть использована отправителем для изменения параметров передачи, например для уменьшения коэффициента сжатия информации с целью улучшения качества ее передачи.

1.5 Основные элементы IP-телефонии

Сеть IP-телефонии (согласно рекомендациям ITU-T H.323) представляет собой набор следующих устройств, соединенных по IP-сети:

- шлюз (gateway);
- диспетчер (gatekeeper);
- монитор (administration manager).

Шлюз – это основная и неотъемлемая часть архитектуры IP-телефонии, непосредственно соединяющая телефонную сеть(PBX/PSTN) с сетью IP.

Функции шлюза, являющиеся базовыми для технологии IP-телефонии:

- ответ на вызов вызывающего абонента PBX/PSTN;
- установление соединения с удаленным шлюзом;
- установление соединения с вызываемым абонентом PBX/PSTN;

- сжатие, пакетирование и восстановление голоса/данных.

Диспетчер - это дополнительное устройство, подключенное только к IP-сети и несущее в себе всю логику работы сети IP-телефонии.

Функции:

- аутентификация и авторизация абонента;
- распределение вызовов между шлюзами;
- биллинг или основанный на стандартах интерфейс к профессиональным системам биллинга третьих производителей).

Диспетчер необходим в любой сети IP-телефонии, содержащей более двух шлюзов.

Монитор - необязательный дополнительный модуль сети IP-телефонии, подключаемый только к IP-сети, используемый для удаленного конфигурирования и поддержки остальных устройств сети - шлюзов и диспетчеров.

Функции:

- интерфейс для удаленной настройки через IP-сеть параметров шлюзов и диспетчеров сети IP-телефонии.

Монитор является удобным средством конфигурирования и администрирования сети. В первых шлюзах для этого просто использовались стандартные сетевые приложения. Позднее в целях оптимизации работы производители оборудования IP-телефонии стали выпускать собственные приложения для этих целей.

Архитектура сети IP-телефонии представляет собой соединенные по IP-сети шлюзы в телефонную сеть, которые предоставляют непосредственный интерфейс абоненту и осуществляют кодировку, сжатие и пакетизацию голоса/данных и их восстановление. Весь механизм взаимодействия шлюзов и учет производится диспетчерами. Для удобства удаленного конфигурирования и администрирования сети может быть использован монитор. Эти три компонента у разных производителей могут называться по-разному, но все они выполняют функции, обобщенные выше.

1.6 Варианты построения IP-телефонных систем

Известны и практически реализуются две базовые схемы IP-телефонии. Первая связана с организацией телефонных переговоров между пользователями персональных компьютеров, оснащенных мультимедийным оборудованием и (или) специальными программными средствами. Вторая схема предусматривает использование шлюзов (рисунок 1.5). Шлюзы могут устанавливаться на серверах Интернет-провайдеров, городских телефонных станциях, серверах локальных вычислительных сетей, Web-серверах компаний, нуждающихся в организации служб технической поддержки, голосовых горячих линий и т.д..

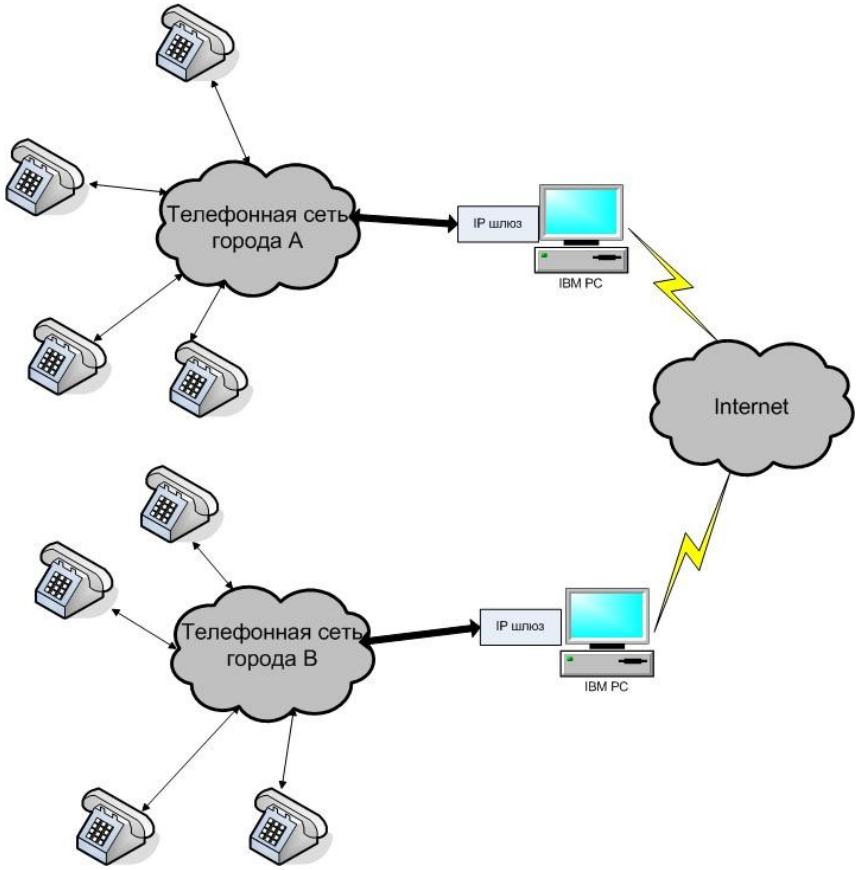


Рисунок 1.5 – Базовая схема IP-телефонии (второй вариант)

Голосовая связь через IP-сеть может осуществляться различными способами:

1) "Телефон - телефон". Для организации такой связи необходимо наличие определенных сетевых устройств и механизмов взаимодействия. Голосовой трафик передается через IP-сеть, как правило, на отдельном дорогостоящем участке. Устройствами, организующими взаимодействие, являются шлюзы, состыкованные, с одной стороны, с телефонной сетью общего пользования, а с другой - с IP-сетью. Голосовая связь в таком режиме имеет высокое качество, и пользоваться ею удобно. Для того чтобы воспользоваться этой услугой, необходимо позвонить провайдеру, обслуживающему шлюз, ввести с телефонного аппарата код и номер вызываемого абонента и разговаривать так же, как при обычной телефонной связи. Все необходимые операции по маршрутизации вызова выполнит шлюз;

2)"Компьютер - телефон". При этом способе чаще всего применяется корпоративная сеть, обслуживающая вызовы от компьютеров до шлюза, которые уже затем передаются по телефонной сети общего пользования. Корпоративные решения с использованием связи "компьютер-телефон" могут помочь сэкономить деньги. Конечному пользователю никакого дополнительного оборудования не требуется. Достаточно иметь под рукой телефон с возможностью тонального набора. Это нужно

для того, чтобы, дозвонившись до оператора, ввести свой код в тональном режиме, а дальше действия абонента ничем не отличаются от привычных.

3) "WEB - телефон". Еще одна новая услуга, которую предоставляют провайдеры IP-телефонии - это звонок с Веб-сайта или Surf&Call - решение компании VocalTec в области веб-телефонии, позволяющее осуществлять вызов, выбрав со страницы Интернет ссылку на имя вызываемого абонента. Это решение направлено, прежде всего, на расширение возможностей электронной коммерции. Surf&Call позволяет пользователям Интернет напрямую поговорить, например, с торговым представителем либо со специалистом технической поддержки интересующей его фирмы. Установление телефонного соединения происходит при нажатии курсором на соответствующую ссылку (например, название компании) на странице Интернет. При этом пользователю не требуется дополнительная телефонная линия или прерывание работы в Интернет.

Общий принцип действия телефонных серверов IP-телефонии следующий. С одной стороны, сервер связан с телефонными линиями и может соединиться с любым телефоном мира. С другой стороны, сервер связан с Интернет и может связаться с любым компьютером в мире. Сервер принимает стандартный телефонный сигнал, оцифровывает его (если он исходно не цифровой), значительно сжимает, разбивает на пакеты и отправляет через Интернет по назначению с использованием протокола Интернет (TCP/IP). Для пакетов, приходящих из Сети на телефонный сервер и уходящих в телефонную линию, операция происходит в обратном порядке. Обе составляющие операции (вход сигнала в телефонную сеть и его выход из телефонной сети) происходят практически одновременно, что позволяет обеспечить полнодуплексный разговор. На основе этих базовых операций можно построить много различных конфигураций.

1.7 Информационное представление речевого сигнала

Проведенный в различных исследовательских группах анализ качества синтезированной речи при передачи речевых данных через сеть Интернет показывает, что основным источником возникновения искажений, снижения качества и разборчивости синтезированной речи является прерывание потока речевых данных, вызванное потерями при передачи по сети либо превышением предельно допустимого времени доставки пакета с речевыми данными.

Процесс речевого диалога в системе Интернет с информационной точки зрения имеет следующие три фазы:

- соединение абонентов,

- обмен информацией,
- разъединение абонентов.

Во время первой и третьей фаз передаются только управляющие данные, и при этом происходит установление соединения. На протяжении второй фазы абоненты обмениваются как управляющими, так и информационными данными.

Основное требование к передаче командной информации - отсутствие ошибок передачи. Время доставки сообщений также играет немаловажную роль. Источником информационных данных является речевой сигнал, возможной моделью которого является нестационарный случайный процесс. Можно выделить следующие типы сигнальных фрагментов: вокализированные, невокализированные, переходные и паузы. При передаче речи в цифровой форме, каждый тип сигнала при одной и той же длительности и одинаковом качестве требует различного числа двоичных единиц (бит) для кодирования и передачи. Следовательно, скорость передачи разных типов сигнала также может быть различной. Поэтому передачу речевых данных в каждом направлении дуплексного канала следует рассматривать как передачу асинхронных логически самостоятельных фрагментов цифровых последовательностей (транзакций) с блочной (дейтаграммной) синхронизацией внутри транзакций, наполненной блоками различной длины.

Асинхронность транзакций позволяет с одной стороны оптимизировать трафик за счет снижения средней скорости передачи и с другой - скомпенсировать неидеальности канала передачи за счет относительной свободы в воспроизведении каждой транзакции.

Особенности функционирования каналов для передачи речевых данных, а также возможные варианты построения систем телефонной связи на базе сети Интернет, предъявляют ряд специфических требований к речевым кодекам (вокодера). Наиболее целесообразным для систем IP-телефонии является применение кодеков с переменной скоростью кодирования речевого сигнала. В основе кодека речи с переменной скоростью лежит классификатор входного сигнала, определяющий степень его информативности и, таким образом, задающий метод, кодирования и скорость передачи речевых данных. Наиболее простым классификатором речевого сигнала является Voice Activity Detector (VAD), который выделяет во входном речевом сигнале активную речь и паузы. При этом, фрагменты сигнала, классифицируемые как активная речь, кодируются каким-либо из известных алгоритмов (как правило, на базе метода Code Excited Linear Prediction - CRLP) с

типичной скоростью 4 - 8 Кбит/с. Фрагменты, классифицированные как паузы, кодируются и передаются с очень низкой скоростью (порядка 0.1 - 0.2 Кбит/с), либо не передаются вообще. Передача минимальной информации о паузных фрагментах предпочтительна.

Схемы более эффективных классификаторов входного сигнала детальнее осуществляют классификацию фрагментов, соответствующих активной речи. Это позволяет оптимизировать выбор стратегии кодирования (скорости передачи данных), выделяя для особо ответственных за качество речи участков речевого сигнала большее число бит (соответственно большую скорость), для менее ответственных - меньше бит (меньшую скорость). При таком построении кодеков могут быть достигнуты низкие средние скорости (2-4 Кбит/с) при высоком качестве синтезируемой речи.

Основным источником возникновения искажений, снижения качества и разборчивости синтезированной речи является прерывание потока речевых данных, вызванное потерями при передачи по сети, либо превышением предельно допустимого времени доставки пакета с речевыми. Поэтому одной из важнейших задач при построении вокодеров для IP-телефонии является создание алгоритмов компрессии речи толерантных к потерям пакетов.

1.8 Архитектура шлюза

Преобразование управляющей информации и данных, поступающих из одной сети (например, PSTN) в пакеты глобальной сети Интернет и обратно в сети IP-телефонии осуществляет шлюз. Причем такое преобразование не должно значительно исказить исходный речевой сигнал, а режим передачи обязан сохранить обмен информацией между абонентами в реальном масштабе времени.

Основные функции, выполняемые шлюзом при соединении типа "точка-точка" состоят в следующем: реализация физического интерфейса с коммуникационной сетью; детектирование и генерация сигналов абонентской сигнализации; преобразование сигналов абонентской сигнализации в пакеты данных и обратно; преобразование речевого сигнала в пакеты данных и обратно; соединение абонентов; передача по сети сигнализационных и речевых пакетов; разъединение связи. Большая часть функций шлюза в рамках архитектуры TCP/IP реализуются в процессах прикладного уровня. Управленческие задачи и связь с сетью осуществляется с помощью универсального процессора, а решения задач сигнальной обработки и телефонного интерфейса выполняются на цифровом процессоре обработки сигналов.

Очень важной является задача обнаружения и детектирования телефонной сигнализации. При использовании двухпроводных абонентских линий актуальной остаётся задача эхокомпенсации, особенность которой состоит в том, что компенсировать необходимо два различных класса сигналов - речи и телефонной сигнализации. Схема сигнальной обработки в шлюзе при подключении аналогового 2-х проводного телефонного канала PSTN показана на рисунке 1.6. Телефонный сигнал поступает на дифференциальную систему, которая разделяет приемную передающую часть канала. Далее сигнал передачи вместе с не отфильтрованной частью сигнала приема подается на аналого-цифровой преобразователь и превращается либо в стандартный 12-ти разрядный сигнал либо в 8-ми разрядный сигнал, закодированный по μ- либо A- закону.

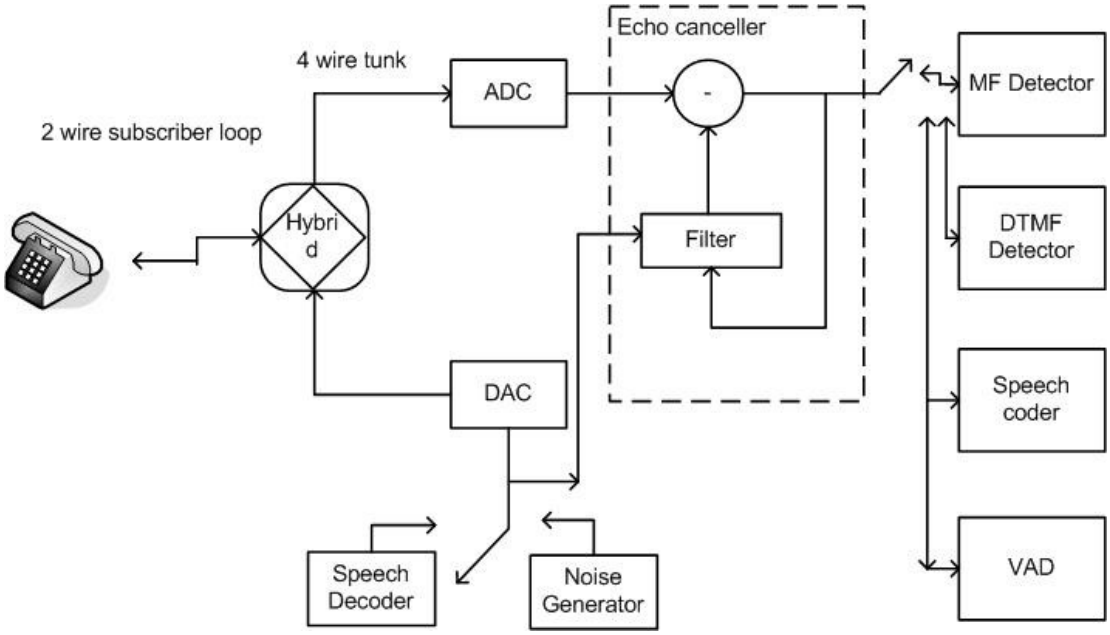


Рисунок 1.6 - Схема сигнальной обработки в шлюзе

В последнем случае обработка должна также включать соответствующий экспандер. В устройстве эхо-компенсации из сигнала передачи удаляются остатки принимаемого сигнала. Эхо-компенсатор представляет собой адаптивный нерекурсивный фильтр, длина памяти (порядок) которого и механизм адаптации выбираются такими, чтобы удовлетворить требованиям рекомендации МККТТ (ITU-T) G.165. Для обнаружения и определения сигналов внутриполосной телефонной сигнализации (MF сигналов), сигналов DTMF либо импульсного наборов используются детекторы соответствующих типов. В режиме сессии дальнейшая обработка входного сигнала происходит в речевом кодере. В анализаторе кодера сигнал сегментируется на отдельные фрагменты длительностью 30 мс и каждому входному блоку, состоящему из 240 отсчетов (1920 бит при A- либо μ-коде и 2880 бит

при 12-ти разрядном линейном коде), сопоставляется информационный кадр длиной 137 бит.

Часть параметров, вычисленная в анализаторе, используется в блоке определения голосовой активности (VAD - voice activity detector), который решает, является ли текущий анализируемый фрагмент сигнала речью или паузой. При наличии паузы информационный кадр может не передаваться в службу виртуального канала. Режим передачи паузных кадров следующий. На сеансовый уровень передается лишь каждый пятый кадр такого типа. Кроме того, при отсутствии речи для кодировки текущих спектральных параметров используется только 27 бит.

На приемной стороне из виртуального канала в логический поступает либо информационный кадр (длиной 137 или 27 бит), либо флаг наличия паузы. На паузных кадрах вместо речевого синтезатора включается генератор комфортного шума, который восстанавливает спектральный состав паузного сигнала.

Параметры генератора обновляются при получении паузного информационного кадра. Наличие информационного кадра длиной 137 бит включает речевой декодер, на выходе которого формируется 12-ти разрядный речевой сигнал. Для эхо-компенсатора этот сигнал является сигналом дальнего абонента, фильтрация которого дает составляющую электрического эха в передаваемом сигнале. В зависимости от типа цифро-аналогового преобразования сигнал может быть подвергнут дополнительной кодировке по А- либо μ- закону. Сложность состоит в том, что служебные сигналы могут перемешиваться с сигналами речи.

Реализация шлюза может различаться, в зависимости от базовой схемы системы IP-телефонии: для пользователей персональных компьютеров и пользователей телефонной сети, осуществляющих связь через Интернет.

Для первой схемы можно выделить два варианта реализации шлюза:

- программный - все процедуры делает персональный компьютер со встроенной звуковой картой;
- программно-аппаратный - используется специализированная DSP карта, выполняющая основные функции.

Первый вариант нашел воплощение в значительном множестве программных продуктов, выпускаемых различными фирмами (например, NetMeeting фирмы Microsoft). Второй вариант также в настоящее время достаточно широко распространен.

Для второй схемы существует только аппаратно-программная реализация, когда в систему входит набор специализированных DSP карт или модулей,

работающих, как правило, под управлением некоторого модуля центрального процессора (CPU). Первые продукты такого рода были реализованы на основе плат фирмы Dialogic и программного обеспечения, разработанного фирмой VocalTeeh. Подобные способы построения шлюзов удобны для офисных и в некоторых случаях для корпоративных применений, однако для крупных интернет-провайдеров и телекоммуникационных компаний, которым необходима установка многоканальных систем, такая реализация малопригодна из-за ненадежности функционирования и сложности обеспечения большого числа каналов.

Основной компонент для реализации аппаратного обеспечения шлюзов - это цифровые процессоры обработки сигналов (DSP). Первым и наиболее мощным DSP этого класса является TMS320C6201 фирмы Texas Instruments с производительностью до 1600 MIPs, на котором возможна реализация 16 и более голосовых каналов через IP.

Задачи, возникающие при реализации шлюзов, во многом аналогичны задачам, решаемым при создании современного станционного оборудования. Вместе с тем, имеется специфика, определяемая широким применением DSP (до десятка и более на одной плате) и особенностями используемых алгоритмов: если на плате шлюза совместно размещаются аналоговая и цифровая части, возникает задача электромагнитной совместимости; если эти части разнесены - задача их сопряжения.

При конструировании шлюза важно обеспечить согласование алгоритма с аппаратной частью - аппаратная часть должна рационально обслуживать алгоритм работы шлюза. Это при экономном использовании аппаратуры не всегда легко (возможно) сделать. Вместе с тем, допустимые модификации алгоритма (распараллеливание вычислений, оптимизация управления ресурсами, рациональный порядок вычислений и пр.) могут оказать заметное влияние на структуру аппаратной реализации и, в целом, обеспечить лучшее решение.

Контрольные вопросы

1) Поясните отличия терминов VoIP, CTI, IP-телефония.
2) Сформулируйте различия транспортные технологий пакетной коммутации.
3) Что лежит в основе технологии IP-телефонии? Назовите ее основные элементы.
4) Перечислите основные элементы IP-телефонии?
5) В чем особенность передачи речи через каналы Интернет?
6) Назовите основные стандарты и отличия протоколов нижней плоскости архитектуры IP-телефонии.
7) Перечислите способы голосовой связи через IP-сеть.
8) Поясните схему сигнальной обработки в шлюзе.

Раздел 2 Стандарт ITU H.323

2.1 Архитектура сети H.323.

Международным союзом электросвязи (ITU) в рекомендации H.323 [17] предложен первый на стандартизованной основе подход к построению сетей IP-телефонии. Сети на базе протоколов H.323 ориентированы на интеграцию с телефонными сетями и могут рассматриваться как сети ISDN, наложенные на сети передачи данных. В частности, процедура установления соединения в таких сетях IP-телефонии базируется на рекомендации Q.931 [18] и аналогична процедуре, используемой в сетях ISDN.

Рекомендация H.323 предусматривает довольно сложный набор протоколов, который предназначен не просто для передачи речевой информации по IP-сетям с коммутацией пакетов. Его цель - обеспечить работу мультимедийных приложений в сетях с негарантированным качеством обслуживания. Речевой трафик - это только одно из приложений H.323, наряду с видеоинформацией и данными. А так как ничего в технике (как и в жизни) не достается даром, обеспечение совместимости с H.323 различных мультимедийных приложений требует весьма значительных усилий. Например, для реализации функции переключения связи (call transfer) требуется отдельная спецификация H.450.2.

Вариант построения сетей IP-телефонии на базе H.323 хорошо подходит тем операторам местных телефонных сетей, которые заинтересованы в использовании сети с коммутацией пакетов (IP-сети) для предоставления услуг междугородной и международной связи. Протокол RAS, входящий в семейство протоколов H.323, обеспечивает контроль использования сетевых ресурсов, поддерживает аутентификацию пользователей и может обеспечивать начисление платы за услуги.

На рисунке 2.1 представлена архитектура сети на базе рекомендации H.323. Основными устройствами сети являются: терминал (Terminal), шлюз (Gateway), привратник (Gatekeeper) и устройство управления конференциями (Multipoint Control Unit - MCU).

Терминал H. 323 -оконечное устройство пользователя сети IP-телефонии, которое обеспечивает двухстороннюю речевую (мультимедийную) связь с другим терминалом H.323, шлюзом или устройством управления конференциями.

Шлюз IP-телефонии реализует передачу речевого трафика по сетям с маршрутизацией пакетов IP по протоколу H.323. Основное назначение шлюза - преобразование речевой информации, поступающей со стороны ТФОП, в вид, пригодный для передачи по сетям с маршрутизацией пакетов IP. Кроме того, шлюз

преобразует сигнальные сообщения систем сигнализации DSS1 и ОКС7 в сигнальные сообщения H.323 и производит обратное преобразование в соответствии с рекомендацией ITU H.246.

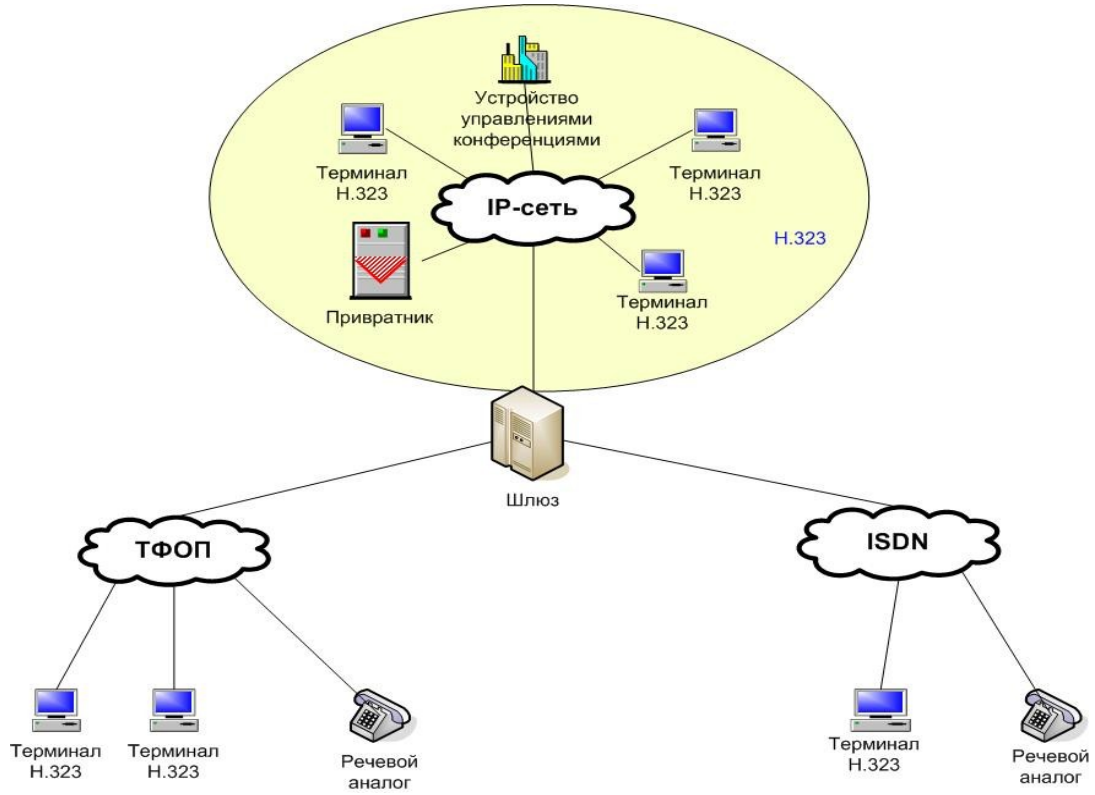


Рисунок 2.1 – Архитектура сети на базе H.323

В *привратнике* сосредоточен весь интеллект сети IP-телефонии. Сеть, построенная в соответствии с рекомендацией H.323, имеет зонную архитектуру, приведенную на рисунке 2.2. Привратник выполняет функции управления одной зоной сети IP-телефонии, в которую входят: терминалы, шлюзы, устройства управления конференциями, зарегистрированные у данного привратника. Отдельные фрагменты зоны сети H.323 могут быть территориально разнесены и соединяться друг с другом через маршрутизаторы.

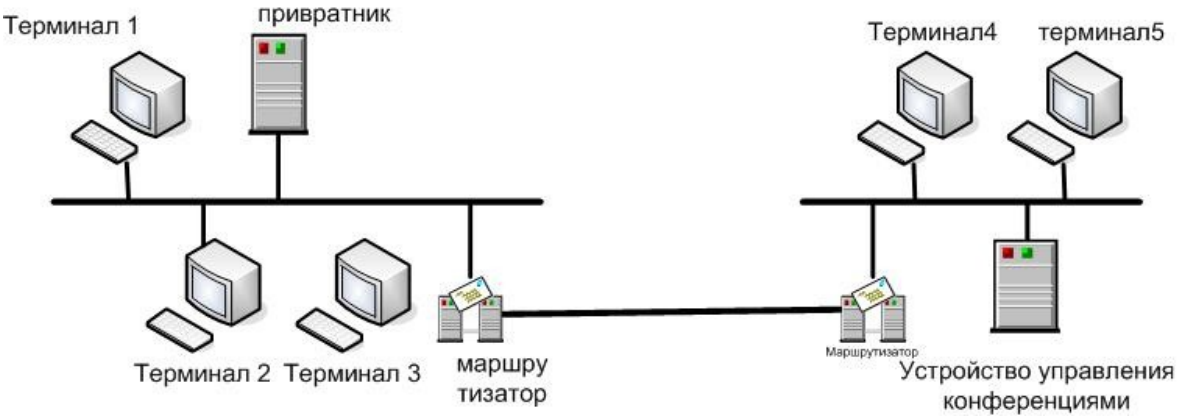


Рисунок 2.2 –Зона сети H.323

Наиболее важными функциями привратника являются:

- регистрация оконечных и других устройств;

- контроль доступа пользователей системы к услугам IP-телефонии при помощи сигнализации RAS;

- преобразование alias-адреса вызываемого пользователя (объявленного имени абонента, телефонного номера, адреса электронной почты и др.) в транспортный адрес сетей с маршрутизацией пакетов IP (IP-адрес плюс номер порта TCP);

- контроль, управление и резервирование пропускной способности сети;

- ретрансляция сигнальных сообщений H.323 между терминалами.

В одной сети IP-телефонии, отвечающей требованиям рекомендации ITU H. 323, может находиться несколько привратников, взаимодействующих друг с другом по протоколу RAS. Кроме основных функций, определенных рекомендацией H.323, привратник может отвечать за аутентификацию пользователей и начисление платы (биллинг) за телефонные соединения.

Устройство управления конференциями обеспечивает возможность организации связи между тремя или более участниками. Рекомендация H.323 предусматривает три вида конференции: централизованная (т.е. управляемая MCU, с которым каждый участник конференции соединяется в режиме точка-точка), децентрализованная (когда каждый участник конференции соединяется с остальными ее участниками в режиме точка-группа точек) и смешанная.

Преимуществом централизованной конференции является сравнительно простое терминальное оборудование, недостатком - большая стоимость устройства управления конференциями.

Для децентрализованной конференции требуется более сложное терминальное оборудование и желательно, чтобы в сети IP поддерживалась передача пакетов IP в режиме многоадресной рассылки (IP multicasting). Если этот режим в сети не поддерживается, терминал должен передавать речевую информацию каждому из остальных участников конференции в режиме точка-точка.

Устройство управления конференциями состоит из одного обязательного элемента - контроллера конференций (Multipoint Controller- MC), и, кроме того, может включать в себя один или более процессоров для обработки пользовательской информации (Multipoint Processor - MP). Контроллер может быть физически совмещен с привратником, шлюзом или устройством управления конференциями. Устройство управления конференциями может быть совмещено со шлюзом или привратником.

Контроллер конференций используется для организации конференции любого вида. Он организует обмен между участниками конференции данными о режимах, поддерживаемых их терминалами, и указывает, в каком режиме участники конференции могут передавать информацию, причем в ходе конференции этот режим может изменяться, например, при подключении к ней нового участника.

Так как контроллеров в сети может быть несколько, для каждой вновь создаваемой конференции должна быть проведена специальная процедура выявления того контроллера, который будет управлять данной конференцией.

При организации централизованной конференции, кроме контроллера MC, должен использоваться процессор MP, обрабатывающий пользовательскую информацию. Процессор MP отвечает за переключение или смешивание речевых потоков, видеоинформации и данных. Для децентрализованной конференции процессор не нужен.

Существует еще один элемент сети H .323 - *прокси-сервер H.323*. Этот сервер функционирует на прикладном уровне и может проверять пакеты с информацией, которой обмениваются два приложения. Прокси-сервер может определять, с каким приложением (H.323 или другим) ассоциирован вызов, и осуществлять нужное соединение. Прокси-сервер выполняет следующие ключевые функции:

- подключение через средства коммутируемого доступа или локальные сети терминалов, не поддерживающих протокол резервирования ресурсов (RSVP). Два таких прокси-сервера могут образовать в IP-сети туннельное соединение с заданным качеством обслуживания;

- маршрутизацию трафика H.323 отдельно от обычного трафика данных;

- обеспечение совместимости с преобразователем сетевых адресов, поскольку допускается размещение оборудования H.323 в сетях с пространством адресов частных сетей;

- защиту доступа (только для трафика H.323).

2.2 Протоколы сигнализации, входящие в семейство H.323

Протокол RAS (Registration, Admission, Status) обеспечивает взаимодействие оконечных и других устройств с привратником. Основными функциями протокола являются: регистрация устройства в системе, контроль его доступа к сетевым ресурсам, изменение полосы пропускания в процессе связи, опрос и индикация текущего состояния устройства. В качестве транспортного протокола используется протокол с негарантированной доставкой информации UDP.

Протокол H.225.0(Q.931) поддерживает процедуры установления, поддержания и разрушения соединения. В качестве транспортного протокола используется протокол с установлением соединения и гарантированной доставкой информации TCP.

По протоколу H.245 происходит обмен между участниками соединения информацией, которая необходима для создания логических каналов. По этим каналам передается речевая информация, упакованная в пакеты RTP/UDP/IP.

Выполнение процедур, предусмотренных протоколом RAS, является начальной фазой установления соединения с использованием сигнализации H.323. Далее следуют фаза сигнализации H.225.0 (Q.931) и обмен управляющими сообщениями H.245. Разрушение соединения происходит в обратной последовательности; в первую очередь закрывается управляющий канал H.245 и сигнальный канал H.225.0, после чего привратник по каналу RAS оповещается об освобождении ранее занимавшейся полосы пропускания.

2.3 Сценарий установления соединения в сети H.323

Сложность протокола H.323 демонстрирует рисунок 2.3, на котором представлен упрощенный сценарий установления соединения между двумя пользователями. В данном сценарии предполагается, что конечные пользователи уже знают IP-адреса друг друга. В обычном случае этапов бывает больше, так как в установлении соединения участвуют привратники и шлюзы.

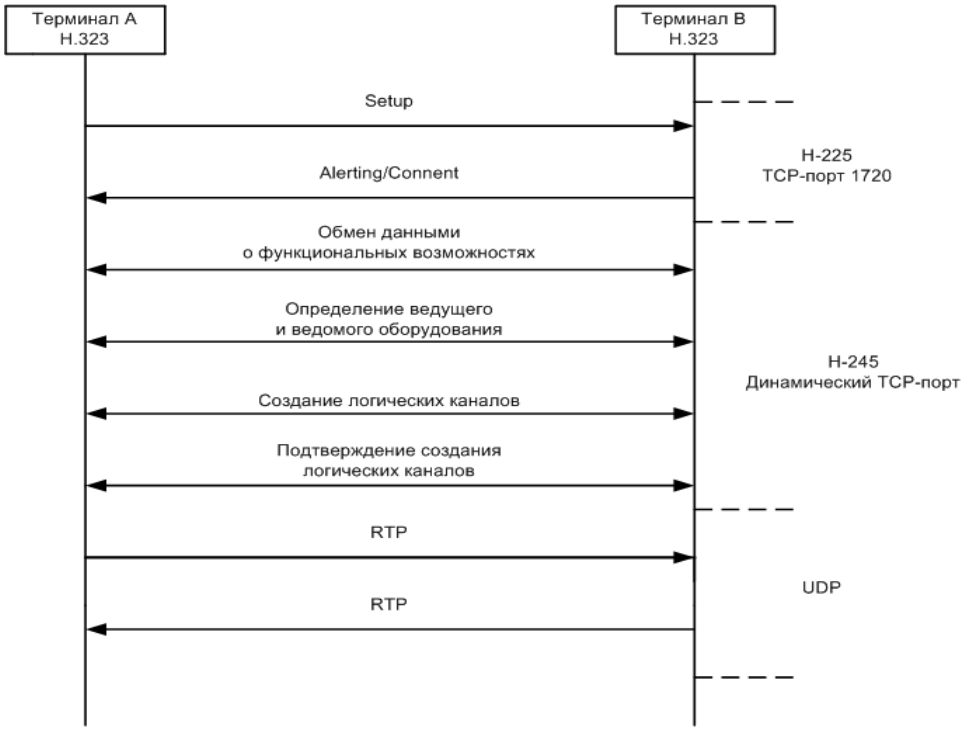


Рисунок 2.3- Упрощенный сценарий установления соединения

Сценарий установления соединения можно условно разбить на следующие этапы.

1) Оконечное устройство пользователя A посылает запрос соединения - сообщение SETUP - к оконечному устройству пользователя B на TCP-порт 1720.
2) Оконечное устройство вызываемого пользователя B отвечает на сообщение SETUP сообщением ALERTING, означающим, что устройство свободно, а вызываемому пользователю подается сигнал о входящем вызове.
3) После того, как пользователь B принимает вызов, к вызывающей стороне A передается сообщение CONNECT с номером TCP-порта управляющего канала H.245.
4) Оконечные устройства обмениваются по каналу H.245 информацией о типах используемых речевых кодеков (G.729, G.723.1 и т.д.), а также о других функциональных возможностях оборудования, и оповещают друг друга о номерах портов RTP, на которые следует передавать информацию.
5) Открываются логические каналы для передачи речевой информации.
6) Речевая информация передаётся в обе стороны в сообщениях протокола RTP; кроме того, ведется контроль передачи информации при помощи протокола RTCP

Приведенная процедура обслуживания вызова базируется на протоколе H. 323 версии 1. Версия 2 протокола H.323 позволяет передавать информацию, необходимую для создания логических каналов, непосредственно в сообщении SETUP протокола H.225.0 без использования протокола H.245. Такая процедура называется «быстрый старт» (Fast Start) и позволяет сократить количество циклов обмена информацией при установлении соединения. Кроме организации базового соединения, в сетях H.323 предусмотрено предоставление дополнительных услуг в соответствии с рекомендациями ITU H.450.x.

Следует отметить еще одну важную проблему - качество обслуживания в сетях H.323. Оконечное устройство, запрашивающее у привратника разрешение на доступ, может, используя поле *transportQoS* в сообщении ARQ протокола RAS, сообщить о своей способности резервировать сетевые ресурсы.

В рассмотренном сценарии оба пользователя включены в одну и ту же IP-сеть. В рамках проекта TIPHON предусматривается организация связи между абонентами IP-сети с учетом того, что вызов транзитом проходит через сеть коммутации каналов. В

качестве сети коммутации каналов может выступать как телефонная сеть общего пользования, так и ISDN, GSM и т.д.

2.4 Проект TIPHON

В проекте TIPHON (Telecommunication and Internet Protocol Harmonization over Networks) предполагается разработка новых стандартов только для тех областей связи, для которых действующие стандарты отсутствуют. Основная задача проекта, работа над которым была начата в 1997 году – решение проблем взаимодействия между сетями с маршрутизацией IP-пакетов и сетями с коммутацией каналов в части поддержки прозрачной передачи речевой и факсимильной информации, т.е. создание единой сетевой инфраструктуры, привлекательной для операторов различных видов связи. Обобщенная структура «сети сетей» представлена на рисунке 2.4.

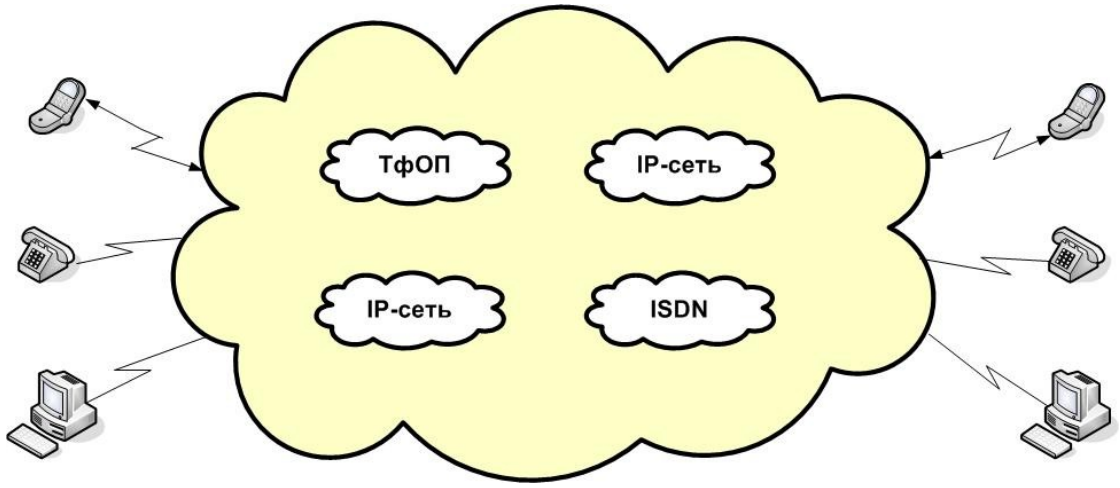


Рисунок 2.4 – Обобщенная структура сети TIPHON

В рамках проекта TIPHON сети, использующие различные технологии коммутации, имеют статус доменов «глобальной сети». В основу взаимодействия этих доменов положено обеспечение гарантированного качества обслуживания (QoS) и защиты межсетевых соединений. Таким образом, сеть TIPHON можно определить как сеть высшего уровня, поддерживающую предоставление услуг телефонной связи и базирующуюся на совокупности сетей более низкого уровня.

В основу проекта TIPHON положены следующие правила:

- терминалами TIPHON могут быть персональные компьютеры и обычные телефоны;
- интерфейс «человек-машина» (MMI) строится по аналогии с телефонным интерфейсом;
- пользователи могут менять точки доступа к услугам глобальной сети, при этом должно сохраняться QoS.

Главной целью проекта TIPHON является разработка механизмов взаимодействия и связанных с ними параметров для обеспечения мультимедийной связи с гарантированным качеством обслуживания между пользователями сетей с коммутацией каналов и сетей с маршрутизацией пакетов. При этом акцент делается на взаимодействие сетей, а не на отдельные сети, для чего и создаются соответствующие спецификации и стандарты, ориентированные на промышленные предприятия, операторские компании, администрации связи, органы сертификации и стандартизации и др.

Проект TIPHON предусматривает решение ряда технических задач, связанных с обеспечением приемлемого качества услуг телефонной связи. В число этих задач входит:

1) разработка эталонных конфигураций и функциональных моделей, требований к взаимодействию различных функциональных объектов, процедур управления соединением и протоколов;
2) преобразование адресов в формате E.164 в IP-адреса;
3) рассмотрение технических аспектов защиты;
4) изучение вопросов мобильности и обеспечения качества обслуживания.

2.5 **Модель сети TIPHON**

Одной из задач, решаемой в рамках рассматриваемого проекта, является разработка принципа декомпозиции шлюза.

Взятую за основу рекомендацию ITU-T H.323, спецификации TIPHON дополняют некоторыми обязательными процедурами, а также механизмами взаимодействия IP-сетей с ТфОП. Функциональная модель сети IP-телефонии, разработанная TIPHON, состоит из тех же компонентов, что и модель сети H.323, однако в ней предусмотрено разделение шлюза на три функционально-независимых объекта. Это шлюз сигнализации (SG), транспортный шлюз (MG) и контроллер транспортного шлюза (MGC).

Шлюз сигнализации служит промежуточным звеном сигнализации между IP-сетями и ТфОП. В задачи транспортного шлюза входит преобразование и/или перекодирование передаваемой информации. К транспортному шлюзу подключены ИКМ-тракты сети с коммутацией каналов, он также подавляет эхо, воспроизводит различные сообщения для абонентов, принимает и передает сигнал ыОТМР и т.д.

Контроллер транспортного шлюза MGC выполняет процедуры сигнализации H. 323, которые определены в рекомендациях ITU-T H.323, H.225 (RAS и Q.931) и H.245, а также преобразует сигнализацию ТФОП в сигнализацию H .323. Основная его задача -

управлять работой транспортного шлюза, т.е. осуществлять управление соединениями, использованием ресурсов, преобразованием протоколов и т.п.

Привратник отвечает за управление объектами сети, в частности, выполняет преобразование адресов (например, телефонных номеров в соответствующие IP-адреса) и маршрутизацию сигнальной информации. Привратник в модели сети TIPHON поддерживает все те функции, которые определены для него в рекомендации H.323. Но, помимо этого, он отвечает за начисление платы, взаиморасчеты, составление отчетов об использовании ресурсов и выполняет некоторые другие функции.

Следует особо подчеркнуть, что MGC - это объект, контролирующий работу транспортного шлюза. Управление соединениями в его функции не входит. Это - задача привратника, который выполняет ее в соответствии с рекомендацией ITU-T H. 323.

Разработанная в рамках проекта TIPHON модель сети, состоящая из функциональных элементов и интерфейсов (точек доступа) между ними, показана на рисунке 2.5.

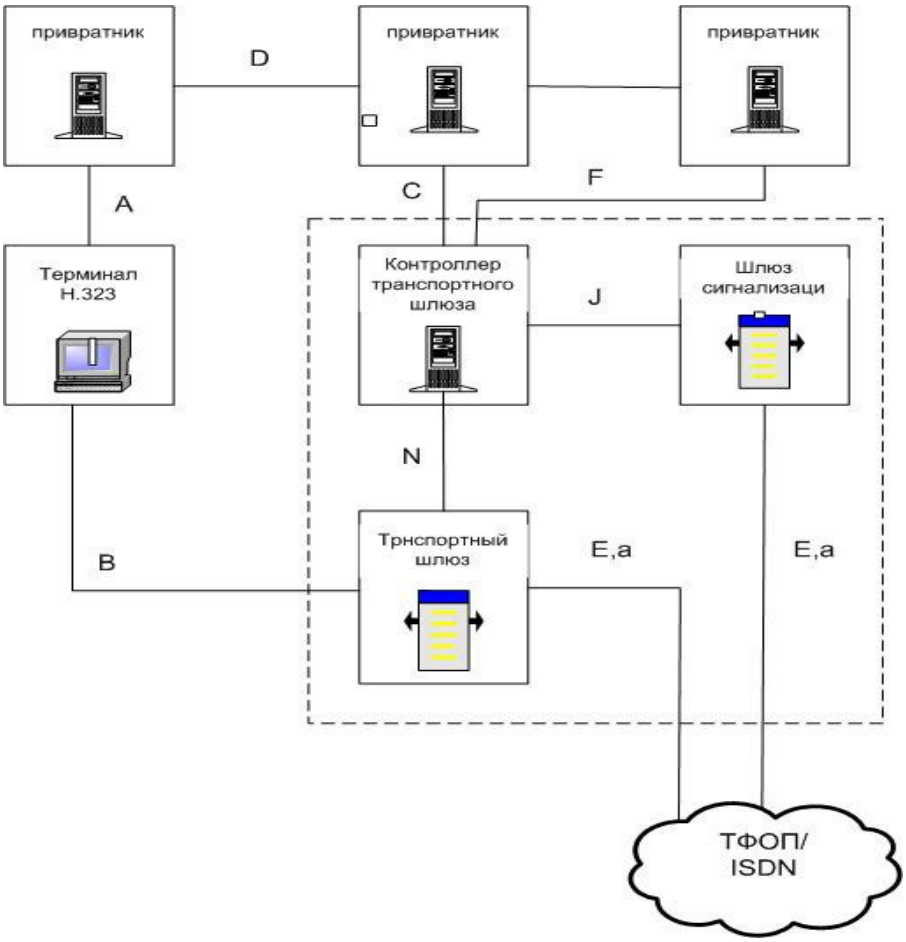


Рисунок 2.5 – Модель сети TIPHON

Чтобы соответствовать рекомендациям TIPHON, оборудование должно поддерживать эти интерфейсы. Так, интерфейс D предназначен для организации взаимодействия между привратниками, а интерфейс C - между контроллером шлюза MGC и привратником. Интерфейс N поддерживает взаимодействие между объектами MGC и MG. Они могут общаться на предмет создания, модификации и завершения соединений; определения требуемого формата информации; генерации акустических сигналов и различных речевых уведомлений; запроса отчетов о событиях, связанных с прохождением информационного потока. Показанные на рис. 2.8 функции поддержки (back-end) могут быть использованы для аутентификации, биллинга, преобразования адресов и других задач.

Смоделированный на основе трех описанных элементов распределенный шлюз воспринимается другими элементами сети как единая система.

Три элемента SG, MG, MGC могут физически не быть разделены, однако такое разделение дает определенные преимущества: использование трех отдельных объектов позволит обрабатывать больше вызовов, так как в этом случае разные функции распределяются по отдельным процессорам. В идеальном случае такие объекты должны иметь стандартные интерфейсы, что даст оператору возможность использовать продукцию разных фирм-производителей. В приведенной выше модели один шлюз сигнализации с целью более экономичного развертывания сети может быть использован для обслуживания большого числа транспортных шлюзов.

2.6 **Адресация сети TIPHON**

От решения задач адресации во многом зависят удобство пользования услугой, работа алгоритмов маршрутизации, обеспечение мобильности абонентов и т.д. Пользователям IP-сетей, как правило, адреса выделяются динамически, поэтому идентифицировать пользователей по их IP-адресам невозможно. Необходимо разработать новый механизм адресации, обеспечивающий технологическую прозрачность при преобразовании номера E.164 в IP-адрес.

Концепцией телефонной связи предусмотрено, что абонент сети ТфОП должен иметь возможность связаться с другим абонентом со своего телефона путём набора номера вне зависимости от того, к сети какого типа подключён адресат. Формат номера обычно соответствует рекомендации E.164. В настоящее время органами стандартизации разрабатываются механизмы преобразования телефонных номеров либо в IP-адреса, либо в унифицированные указатели ресурсов (URL).

Отображение телефонных номеров на IP-адреса создаёт проблему управления данными, так как пользователи имеют тенденцию перемещаться по сети Интернет и

входить в систему из разных мест, поэтому их IP-адрес регулярно изменяется. Предполагается, что сети IP-телефонии будут обслуживать сотни миллионов пользователей, и как следствие, гибкое и надёжное решение вопроса о том, каким образом должно выполняться регулярное обновление данных и как должны обрабатываться запросы, со всей очевидностью станет сложной проблемой.

Отображение телефонных номеров на URL немного упрощает проблему преобразования адресов путём использования Интернет-ярлыка для идентификации пользователя. Однако, как только телефонный номер преобразован в ярлык, последний должен быть преобразован в адрес поставщика услуг Интернет, который, в свою очередь, формирует окончательный IP-адрес получателя. Наличие такого большого количества стадий для поиска вызываемого абонента, будет существенно увеличивать время между набором номера вызывающим абонентом и получением им сигнала КПВ или зуммера «Занято».

В настоящее время органами стандартизации разрабатываются и другие механизмы, обеспечивающие надлежащую адресацию и маршрутизацию номеров E. 164. Вопрос преобразования номера телефонной сети общего пользования в IP-адрес представляется пока еще довольно сложным, и пути его решения разрабатываются не только рабочей группой в рамках проекта TIPHON, но и другими организациями, например IETF.

2.7 Классы обслуживания

Еще одним важным направлением работы TIPHON является вопрос о классах обслуживания. Конечный пользователь ожидает, что услуга передачи речевой информации будет предоставляться с хорошим качеством и высокой надежностью. Но такие примеры, как предоставление услуг сотовой связи стандарта GSM и микросотовой связи стандарта DECT, показали, что конечного пользователя удовлетворяет качество обслуживания, худшее по сравнению сТфОП или ISDN, до тех пор, пока он получает выгоду от использования новой услуги. В случае предоставления услуг сотовой связи - это мобильность терминала, а в случае IP-телефонии это могут быть низкая стоимость, возможности интеграции услуг в рамках единой сети

Для операторов очень привлекательна возможность предоставления услуг с разным уровнем качества (и, соответственно, с разными тарифами), причем поддерживаемым не только в пределах сети одного оператора, но и при связи между сетями разных операторов. Для этого в рамках проекта TIPHON определены

четыре класса обслуживания, каждый из которых гарантирует определенное качество, как при установлении соединения, так и во время сеанса связи.

Качество обслуживания при установлении соединения характеризуется, прежде всего, временем его установления, т.е. временем между набором абонентом последней цифры номера (или, например, команды ввода при наборе адреса на компьютере) и получением им ответного акустического сигнала.

Качество обслуживания во время сеанса связи определяется многими факторами, основными из которых являются сквозная временная задержка и качество сквозной передачи речи (оценивается методами экспертной оценки).

Набор рекомендаций H.323 не смог обеспечить серьезные улучшения для конечных пользователей. Она не смогла стать основной ни для разработки нового поколения конечных точек, ни для поддержки дополнительных видов обслуживания, подобных тем, что предоставляют традиционные учрежденческие АТС. Для того чтобы обеспечить реальные инновации на уровне конечных узлов, индустрия должна упростить процесс разработки новых приложений, предложив для этого стандартные программные интерфейсы и высокоуровневый инструментарий. Развитие средств компьютерно-телефонной интеграции показывает необходимость того, чтобы модель предоставления телефонных услуг строилась на базе служб сетей передачи данных - тогда она позволит быстро разрабатывать удобные и совместимые решения для сетей NGN.

Контрольные вопросы

1) Назначение и основные функции протокола H.323?
2) Перечислите основные элементы архитектуры сети на базе протокола H.323 .
3) Поясните функции основных элементов архитектуры сети на базе H.323.
4) Перечислите основные недостатки сети H.323.
5) Перечислите протоколы сигнализации, входящие в семейство H.323.
6) Для каких целей используется протокол RAS?
7) Какие цели и задачи стояли перед разработчиками проекта TIPHON?
8) Назовите основные элементы функциональной модели сети IP-телефонии на базе TIPHON.
9) Поясните назначение использования распределенного шлюза.
10) Поясните механизм преобразования телефонных номеров в IP-адреса/URL-адреса.
11) Каким образом обеспечивается качество обслуживания пользователей сети TIPHON?

Раздел 3 Протокол инициирования сеансов SIP

3.1 Назначение и принципы работы протокола SIP

Внедрить развитую поддержку речевых коммуникаций в среду передачи данных можно с помощью протоколов, ориентированных в первую очередь на предоставление услуг конечным пользователям. Созданные на их базе продукты должны легко интегрироваться в существующие сети, требуя лишь минимальной модификации сетевых инфраструктур, а сами протоколы - легко расширяться, причем так, чтобы добавление в них новых функций не нарушало работу систем, основанных на предыдущих версиях, и не требовало соответствующего одобрения зачастую конкурирующими друг с другом организациями по стандартизации. Всем этим критериям соответствует протокол SIP.

Протокол инициирования сеансов SIP (Session Initiation Protocol), разработанный комитетом IETF (спецификации протокола представлены в документе RFC 3261 [1]), является текст-ориентированным протоколом прикладного уровня и предназначается для организации, модификации и завершения сеансов связи: мультимедийных конференций, телефонных соединений и распределения мультимедийной информации.

К основным функциям протокола следует отнести: определение местоположение оконечной точки, передача сигнальной информации, приглашающей к взаимодействию, согласование параметров сессии для последующего её установления, разрушение сессии.

За основу были взяты протоколы, применяемые в таких IP-сервисах, как Web -протокол HTTP и электронная почта – протокол SMTP. Подход, используемый в SIP основан на том же подходе, что HTTP: запрос - ответ (request - reply). Все сообщения SIP текстовые, коды возврата аналогичны HTTP.

В основу протокола заложены следующие принципы:

- персональная мобильность пользователей. Пользователи могут перемещаться без ограничений в пределах сети, поэтому услуги связи должны предоставляться им в любом месте этой сети. Пользователю присваивается уникальный идентификатор, а сеть предоставляет ему услуги связи вне зависимости от того, где он находится. Для этого пользователь с помощью специального сообщения информирует сеть о своих перемещениях;

- масштабируемость сети характеризуется, в первую очередь, возможностью увеличения количества элементов сети при её расширении. Серверная структура сети, построенной на базе протокола SIP, в полной мере отвечает этому требованию;

- расширяемость протокола характеризуется возможностью дополнения протокола новыми функциями при введении новых услуг и его адаптации к работе с различными приложениями. Расширение функций протокола SIP может быть произведено за счет введения новых заголовков и типов сообщений;

- интеграция в стек существующих протоколов Интернет, разработанных IETF. Протокол SIP является частью сложной архитектуры, разработанной комитетом IETF. Эта архитектура включает в себя также протокол резервирования ресурсов (Resource Reservation Protocol, RSVP; RFC 2205), транспортный протокол реального времени (Real-Time Transport Protocol, RTP; RFC 1889), протокол передачи потоков в реальном времени (Real-Time Streaming Protocol, RTSP; RFC 2326 [10]), протокол описания параметров связи (Session Description Protocol, SDP; RFC 2327) и прочие. Однако функции протокола SIP не зависят ни от одного из этих протоколов;

- взаимодействие с другими протоколами сигнализации. Протокол SIP может быть использован совместно с протоколом H.323. Возможно также взаимодействие протокола SIP с системами сигнализации ТфОП - EDSS1 и ОКС7. Для упрощения такого взаимодействия сигнальные сообщения протокола SIP могут переносить не только SIP-адрес, но и телефонный номер формата E.164 или любого другого формата. Кроме того, протокол SIP, наравне с протоколами H.323 и ISUP, может применяться для обеспечения функционирования устройств управления шлюзами, в этом случае он должен взаимодействовать с протоколом MGCP или MEGACO;

- поддержка услуг интеллектуальной сети (преобразование имен, переадресация и маршрутизация).

Протокол SIP предусматривает организацию конференций трех видов:

- в режиме многоадресной рассылки, когда информация передается на один multicast-адрес, а затем доставляется сетью конечным адресатам;

- при помощи устройства управления конференции, к которому участники конференции передают информацию в режиме точка-точка, а оно, в свою очередь, обрабатывает ее (т.е. смешивает или коммутирует) и рассылает участникам конференции;

- путем соединения каждого пользователя с каждым в режиме точка-точка.

Протокол SIP дает возможность присоединения новых участников к уже существующему сеансу связи, т.е. двусторонний сеанс может перейти в конференцию.

Подход к построению сетей IP-телефонии на базе SIP проще в реализации, чем H323, но меньше подходит для организации взаимодействия с телефонными сетями. Кроме того, протокол SIP имеет следующие недостатки: не реализованы механизмы

управления потоками информации и предоставления гарантированного качества обслуживания, не предназначен для передачи пользовательской информации, в его сообщениях может переноситься информация лишь ограниченного объема. При переносе через сеть слишком большого сообщения SIP не исключена его фрагментация на уровне IP, что может повлиять на качество передачи информации.

3.2 Интеграция протокола SIP с IP-сетями

Важной особенностью протокола SIP является его независимость от транспортных технологий. В качестве транспорта могут использоваться протоколы X. 25, Frame Relay, AAL5, IPX и др. Структура сообщений SIP не зависит от выбранной транспортной технологии.

Сигнальные сообщения SIP могут переноситься не только протоколом транспортного уровня UDP, но и протоколом TCP. Протокол UDP позволяет быстрее, чем TCP, доставлять сигнальную информацию (даже с учетом повторной передачи неподтвержденных сообщений), а также вести параллельный поиск местоположения пользователей и передавать приглашения к участию в сеансе связи в режиме многоадресной рассылки. В свою очередь, протокол TCP упрощает работу с межсетевыми экранами, а также гарантирует надежную доставку данных. При использовании протокола TCP разные сообщения, относящиеся к одному вызову, либо могут передаваться по одному TCP-соединению, либо для каждого запроса и ответа на него может открываться отдельное TCP-соединение.

По сети с маршрутизацией пакетов IP может передаваться пользовательская информация практически любого вида: речь, видео и данные, а также любая их комбинация, называемая мультимедийной информацией. При организации связи необходимо известить встречную сторону, какого рода информация может приниматься (передаваться), алгоритм ее кодирования и адрес, на который следует передавать информацию. Таким образом, одним из обязательных условий организации связи при помощи протокола SIP является обмен между сторонами данными об их функциональных возможностях. Для этой цели чаще всего используется протокол описания сеансов связи - SDP (Session Description Protocol). Поскольку в течение сеанса связи может производиться его модификация, предусмотрена передача сообщений SIP с новыми описаниями сеанса средствами SDP.

Для передачи медиа-информации комитет IETF предлагает использовать протокол RTP, но сам протокол SIP не исключает возможность применения для этих целей других протоколов.

Многие стандарты никогда не воплощаются в успешные коммерческие продукты. К SIP это не относится. На рынке уже есть поддерживающие его шлюзы, серверы-посредники, терминалы. Внедрение протокола SIP сопровождается работой по дальнейшему развитию и расширению протокола. Одно из возможных новых применений SIP – это использование его в качестве протокола установления соединения в сотовых сетях третьего поколения (3G). Так, организация 3GPP (3rd Generation Partnership Project) уже приняла его в качестве сигнального протокола мобильной сети 3-го поколения. Протокол SIP является перспективным современным протоколом для предоставления широкого спектра телекоммуникационных услуг.

3.3 Адресация в сетях SIP

Для организации взаимодействия с существующими приложениями IP-сетей и для обеспечения мобильности пользователей протокол SIP использует адрес, подобный адресу электронной почты. В качестве адресов рабочих станций используются специальные универсальные указатели ресурсов - URL (Universal Resource Locators), так называемые SIP URL.

SIP-адреса бывают четырех типов:

- *имя@домен*,
- *имя@хост*,
- *имя@IP-адрес*,
- *№телефона@шлюз*.

Таким образом, адрес состоит из двух частей. Первая часть – это имя пользователя, зарегистрированного в домене или на рабочей станции. Если вторая часть адреса идентифицирует какой-либо шлюз, то в первой указывается телефонный номер абонента.

Во второй части адреса указывается имя домена, рабочей станции или шлюза. Для определения IP-адреса устройства необходимо обратиться к службе доменных имен - Domain Name Service (DNS). Если же во второй части SIP-адреса размещается IP-адрес, то с рабочей станцией можно связаться напрямую.

В начале SIP адреса ставится слово ’sip:’, указывающее, что это именно SIP-адрес, т.к. бывают и другие (например, ‘tel:’). Ниже приводятся примеры SIP-адресов:

sip: dfct@mail.ru.

sip: user1@170.52.0.215

sip: 111-22-33@sip-gateway.ru

Все SIP-номера находятся в единой сети и все звонки внутри этой сети бесплатны, таким образом можно совершенно бесплатно звонить в любую точку мира на SIP-номер.

3.4 **Сеть на базе протокола SIP**

Сеть на базе протокола SIP содержит элементы трех видов: агенты пользователя (User agents), прокси-сервер (Proxy server), сервер переадресации (Redirect server) (рисунок 3.1)

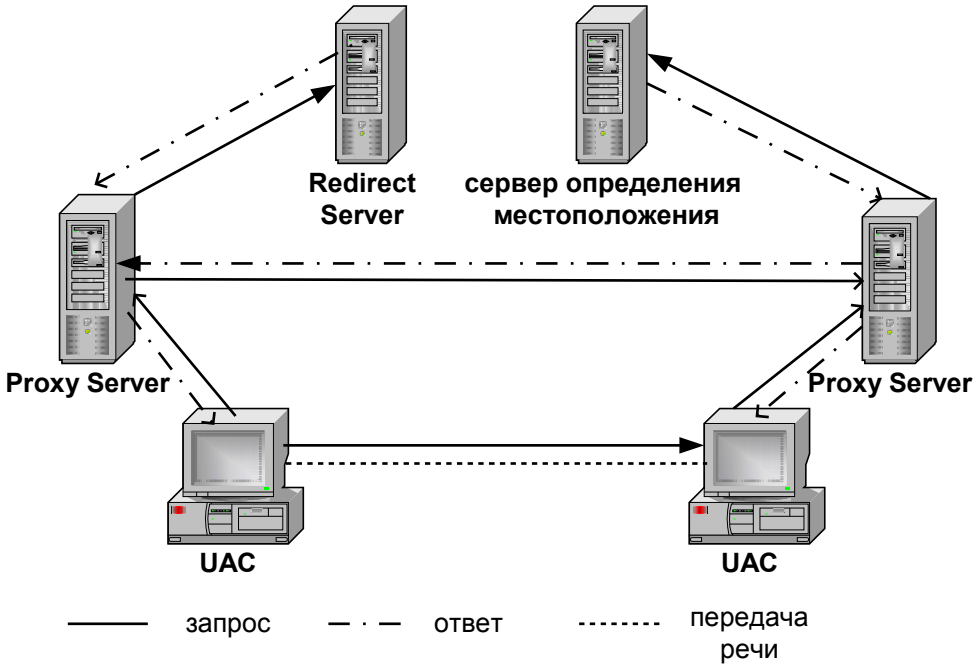


Рисунок 3.1 – Сеть на базе протокола SIP

Агенты пользователя являются приложениями терминального оборудования и включают в себя две составляющие: агент пользователя - клиент (User agent client – UAC) и агент пользователя- сервер (User agent server –UAS). UAC выступает в качестве вызывающей стороны, а UAS – в качестве вызываемой стороны.

Прокси-сервер содержит функции клиента (UAC) и сервера (UAS), действует «от имени других клиентов». Существует два типа сетевых серверов SIP: прокси-серверы (серверы посредники) и серверы переадресации. Серверы SIP могут работать как в режиме с сохранением состояний текущих соединений (statefull), так и в режиме без сохранения состояний текущих соединений (stateless). В отличии от привратника H.323, сервер SIP в режиме stateless может обслужить неограниченное количество пользователей.

Сервер переадресации определяет текущее местоположение вызываемого абонента и сообщает его вызывающему пользователю. Для определения текущего местоположения вызываемого абонента сервер переадресации обращается к серверу

определения местоположения, принципы работы которого RFC 3261 не специфированы.

3.5 Сценарий установления соединения

Диаграмма, изображённая на рисунке 3.2 представляет типичный пример обмена SIP-сообщениями между двумя пользователями A и B. В этом примере пользователь A использует SIP-приложение на своём ПК для вызова пользователя B через сеть Интернет. Абонент B принимает вызов на свой SIP-телефон. Два SIP прокси-сервера, отражённые на рисунке, действуют от имени пользователей, выполняя функции посредника при установлении сессии.

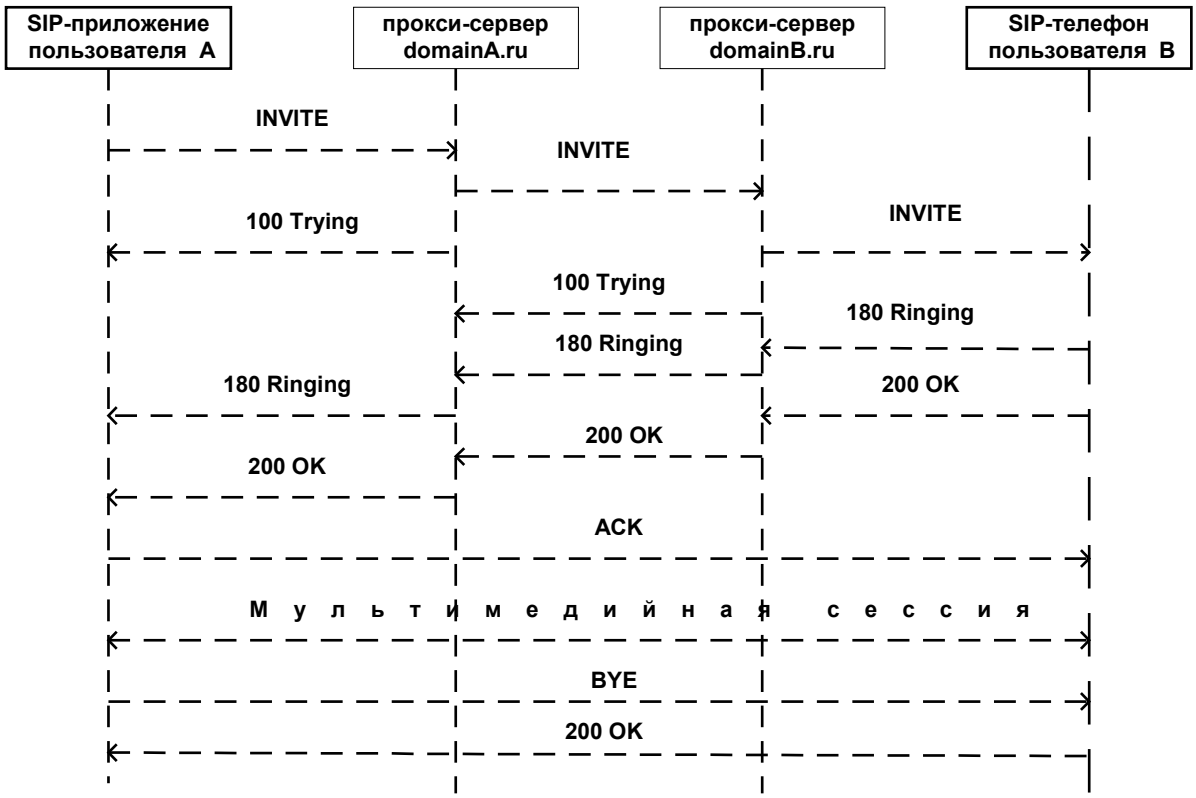


Рисунок 3.2- Взаимодействие элементов сети по протоколу SIP

Пользователь A вызывает пользователя B, используя его SIP-адрес.

Так как протокол SIP базируется на транзакционной модели «запрос/ответ», каждая транзакция состоит из запроса определённого типа, требующего выполнения каких-либо действий прокси-сервером, и, по меньшей мере, одного ответа.

В представленном сценарии транзакция создаётся при отсылке SIP-приложением пользователя A запроса INVITE, адресованного пользователю B. Запрос INVITE содержит группу заголовков. Заголовки обеспечивают дополнительную информацию о сообщении. Присутствующие в запросе INVITE заголовки включают уникальный идентификатор вызова, адрес места назначения, адрес отправителя,

информацию о типе сессии, которую желает установить пользователь A с пользователем B. Запрос INVITE может выглядеть следующим образом:

INVITE sip:userB@domainB.ru SIP/2.0
Via: SIP/2.0/UDP rn07.domainA.ru;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: User B <sip:userB@domainB.ru>
From: User A <sip:userA@domainA.ru>;tag=0708196022
Call-ID: a84b4c76e66710@rn07.domainA.ru
CSeq: 314159 INVITE
Contact: <sip:userA@rn07.domainA.ru>
Content-Type: application/sdp
Content-Length: 142

Первая строка сообщения содержит тип запроса (INVITE). Следующие строки представляют собой список заголовков:

- **Via** - содержит адрес (rn07.domainA.ru), на который пользователь A ожидает получить ответы на переданный запрос. Параметр «branch» идентифицирует данную транзакцию;

- **To** – содержит отображаемое имя (пользователь B) и SIP или SIPS URI (sip:userB@domainB.ru), на который был изначально направлен запрос;

- **From** – содержит отображаемое имя (пользователь A) и SIP или SIPS URI (sip:userA@domainA.ru), который указывает инициатора запроса. Заголовок включает параметр «tag», представляющий собой строку со случайным значением (*0708196022*), которая в целях идентификации была добавлена к URI SIP-приложением;

- **Call-ID** – содержит уникальный идентификатор для данного вызова, представляющий собой сочетание строки со случайным значением и имя или IP-адрес узла, на котором установлено SIP-приложение. Комбинация «tag» заголовка To, «tag» заголовка From и Call-ID полностью определяет отношения между равноправными SIP элементами пользователей A и B и носит название диалог;

- **CSeq** – содержит порядковый номер и тип запроса. Порядковый номер увеличивается на единицу для каждого нового запроса в диалоге;

- **Contact** – содержит SIP или SIPS URI, который представляет адрес, по которому возможно связаться с пользователем A в текущий момент времени. В отличии заголовка Via , который указывает другим элементам, куда отправлять ответы, значение заголовка Contact информирует другие элементы сети, куда отсылать будущие запросы;

- Max-Forwards – служит для ограничения числа пересылок, которые может осуществить запрос на пути к месту назначения;

- Content-Type – содержит тип тела сообщения;

- Content-Length – содержит длину тела сообщения в байтах.

Характеристики сессии, такие как тип медиа-информации, используемый кодек или частота дискретизации не описываются средствами SIP. Тело SIP сообщения содержит описание сессии, выполненное в формате другого протокола. Один из таких протоколов SDP. Это SDP-сообщение переносится SIP-сообщением так же, как документ прикреплённый к сообщению электронной почты.

Поскольку SIP-приложение пользователя A не располагает сведениями о местоположении пользователя B или имени SIP-сервера в домене domainB.ru, он посылает INVITE SIP-серверу, обслуживающему домен пользователя A, domainA.ru. Адрес SIP-сервера domainA.ru, может быть занесён в конфигурацию SIP-приложения пользователя A или, например, определён с помощью протокола динамической конфигурации узла DHCP. SIP-сервер domainA.ru является прокси-сервером, который получает запрос INVITE и посылает обратно SIP-приложению пользователя A ответ с кодом 100 (Trying). Этот ответ информирует о том, что INVITE был получен и прокси-сервер выполняет работу по маршрутизации запроса к месту назначения. Ответы в протоколе SIP характеризуются трёхзначным кодом и следующей за ним уточняющей фразой, что позволяет коррелировать этот ответ с отосланным запросом.

Прокси-сервер domainA.ru определяет местонахождение прокси-сервера в домене domainB.ru и пересылает ему запрос INVITE. Перед пересылкой, прокси-сервер добавляет дополнительное значение заголовка Via, которое содержит его собственный адрес (INVITE уже включает адрес пользователя A в первом значении заголовка Via). Прокси-сервер domainB.ru получает INVITE и передаёт обратно ответ с кодом 100 (Trying), указывающий прокси-серверу domainA.ru, что получил запрос INVITE и занимается его обработкой. Прокси- сервер обращается к базе данных (серверу определения местоположения), которая содержит текущий IP-адрес пользователя B. Прокси-сервер добавляет значение заголовка Via со своим адресом в INVITE и пересылает его на SIP-телефон пользователя B.

SIP-телефон пользователя B получает INVITE и оповещает пользователя о входящем вызове пользователя A. Оборудование пользователя B отправляет код 180 (вызов), который маршрутизируется обратно через два прокси-сервера в обратном порядке.

Когда оборудование пользователя A получает ответ с кодом 180, он передаёт эту информацию пользователю A, используя звуковой сигнал или отображая сообщение на экране терминала пользователя A.

В случае если пользователь B поднимает трубку, его SIP-телефон посылает ответ с кодом 200 (OK) для того, чтобы указать, что пользователь B принял вызов. Ответ с кодом 200 (OK) содержит тело сообщения с вариантом SDP-описания сессии, которую желает установить пользователь B с пользователем A. В результате происходит двухэтапный процесс обмена SDP-сообщениями: от пользователя A к пользователю B и обратно от пользователя B к пользователю A. Этот обмен обеспечивает возможности по согласованию параметров и базируется на простой модели *предложение/ответ* SDP-обмена. Если пользователь B не пожелал бы отвечать на звонок или был занят другим вызовом, то вместо ответа с кодом 200 был бы отослан ответ с кодом ошибки, что не привело бы к установлению сессии. Ответ с кодом 200 (OK) может выглядеть следующим образом:

SIP/2.0 200 OK
Via: SIP/2.0/UDP server10.domainB.ru
;branch=z9hG4bKnashds8;received=224.40.51.2
Via: SIP/2.0/UDP bigbox3.site3.domainA.ru
;branch=z9hG4bK77ef4c2312983.1;received=224.40.51.1
Via: SIP/2.0/UDP rn07.domainA.ru
;branch=z9hG4bK776asdhds ;received=192.0.2.3
To: User B <sip:userB@domainB.ru>;tag=a6c85cf
From: User A <sip:userA@domainA.ru>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.domainA.ru
CSeq: 314159 INVITE
Contact: <sip:userB@224.40.51.4>
Content-Type: application/sdp
Content-Length: 124

Первая строка ответа содержит код ответа (200) и Reason-Phrase - уточняющую фразу (OK). В последующих строках содержатся поля заголовков. Заголовки Via, To, From, Call-ID и CSeq копируются из запроса INVITE (в ответ включено три значения заголовка Via – одно добавлено SIP-приложением User A, второе – прокси-сервером domainA.ru и третье – прокси-сервером domainB.ru). SIP-телефон пользователя B добавляет параметр «tag» в заголовок To – он будет использоваться во всех последующих запросах и ответах данного вызова. Заголовок Contact содержит URI, который характеризует текущее местонахождение пользователя B. Заголовки Content-

Type и Content-Length описывают тело сообщения содержащее SDP-описание сессии пользователя B. Прокси-сервер также может послать INVITE одновременно нескольким терминалам, где может находиться пользователь. Такой тип параллельного поиска получил название размножение запросов.

В рассматриваемом случае ответ с кодом 200 маршрутизируется обратно через два прокси-сервера и приходит на SIP-приложение пользователя A, которое прекращает подачу сигнала КПВ и сообщает о том, что вызываемый пользователь принял вызов. В итоге, SIP-приложение пользователя A посылает сообщение подтверждения ACK для того, чтобы подтвердить принятие окончательного ответа (код 200). На рисунке 3.2 ACK посылается напрямую от SIP-приложения пользователя A SIP-телефону пользователя B, обходя прокси-серверы. Подтверждение завершает INVITE/200/ACK трехэтапное согласование, используемое для установления SIP сессии. Сессия между пользователем A и пользователем B теперь считается установленной, и они отсылают пакты с речевой информацией, используя формат, принятый при обмене SDP. В общем случае передающиеся по сквозному принципу речевые пакеты транспортируются по маршруту, отличному от пути следования SIP сигнальных сообщений.

В конце вызова (например, пользователь B первым вешает трубку) создаётся сообщение BYE. Это сообщение маршрутизируется напрямую SIP-приложению пользователя A в обход прокси-серверов. Пользователь A подтверждает получение BYE посылкой ответа с кодом 200 (OK), который завершает сессию BYE-транзакцию. Подтверждения ACK не посылается, так как используется только для подтверждения ответов на запрос INVITE.

Контрольные вопросы

1) Поясните использование протокола SIP в IP-телефонии.

2) На какой модели базируется протокол SIP?

3) Перечислите основные элементы сети IP-телефонии на базе протокола SIP .

4) Поясните функции основных элементов архитектуры сети на базе SIP.

5) Каковы основные преимущества SIP перед подходом, предложенным ITU в рекомендации H.323?

6) Перечислите основные недостатки сети с использованием SIP.

7) Поясните синтаксис SIP URL.

8) Для каких целей используется протокол SDP?

9) Каким образом создается однонаправленный разговорный канал для передачи вызывающему абоненту акустических сигналов, извещений?

Раздел 4 MGCP

4.1 Сеть на базе MGCP и MEGACO

Третий подход к построению сетей IP-телефонии, основанный на использовании протокола MGCP, также предложен комитетом IETF рабочей группой MEGACO.

При разработке этого протокола рабочая группа MEGACO основывалась на сетевой архитектуре, содержащей основные функциональные блоки трех видов (рисунок 4.1) [18]:

- шлюз (MG - Media Gateway), который выполняет функции преобразования речевой информации, поступающей со стороны ТфОП с постоянной скоростью передачи, в вид, пригодный для передачи по сетям с маршрутизацией пакетов IP (кодирование и упаковку речевой информации в пакеты RTP/UDP/IP, а также обратное преобразование);

- контроллер шлюзов (GA - Call Agent), которой выполняет функции управления шлюзами;

- шлюз сигнализации – (SG - Signaling Gateway), который обеспечивает доставку сигнальной информации, поступающей со стороны ТфОП, к контроллеру шлюзов и перенос сигнальной информации в обратном направлении.

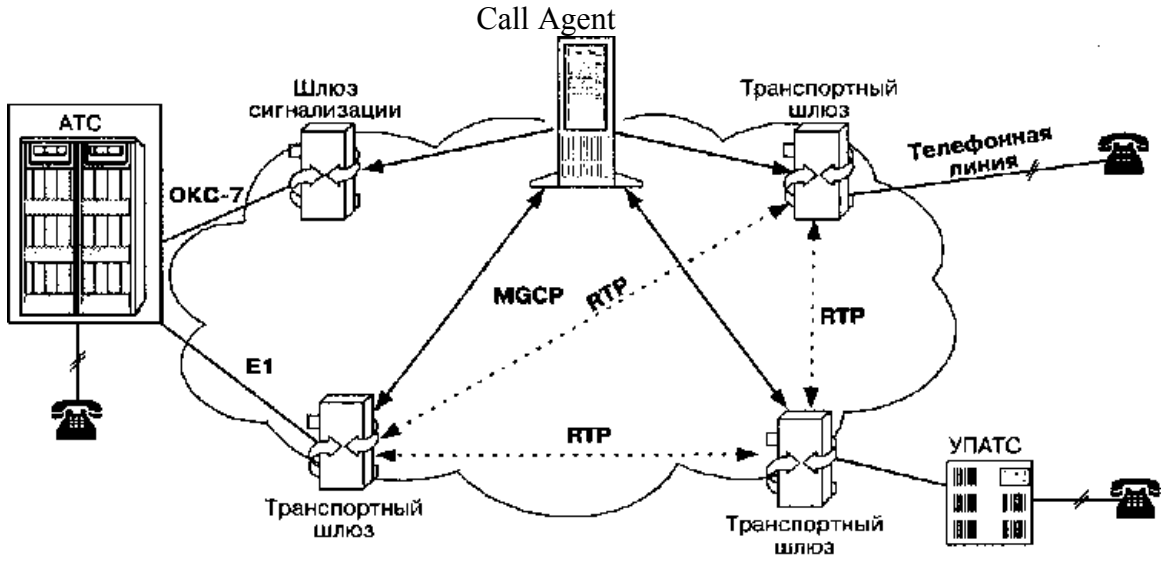


Рисунок 4.1-Архитектура сети на базе протокола MGCP

Таким образом, весь интеллект функционально распределенного шлюза сосредоточен в контроллере, функции которого могут быть распределены между несколькими компьютерными платформами.

Шлюз сигнализации выполняет функции транзитного пункта сети сигнализации ОКС7. Сами шлюзы выполняют только функции преобразования речевой информации. Один контроллер управляет одновременно несколькими шлюзами. В сети могут присутствовать несколько контроллеров. Предполагается, что они синхронизованы между собой и согласованно управляют шлюзами, участвующими в соединении. Вместе с тем, MEGACO не определяет протокола для синхронизации работы контроллеров. В работах [17, 18], посвященных исследованию возможностей протокола MGCP, для этой цели предлагается использовать протоколы H.323, SIP и др.

Сообщения протокола MGCP переносятся протоколом без гарантированной доставки сообщений UDP. Рабочая группа SIGTRAN комитета IETF разрабатывается механизм взаимодействия контроллера шлюзов и шлюза сигнализации.

Шлюз сигнализации должен принимать поступающие из ТфОП пакеты трех нижних уровней системы сигнализации ОКС7 (уровней подсистемы переноса сообщений MTP) и передавать сигнальные сообщения верхнего, пользовательского, уровня к контроллеру шлюзов. Шлюз сигнализации также должен уметь передавать по IP-сети приходящие из ТфОП сигнальные сообщения Q.931 .

Основное внимание рабочей группы SIGTRAN уделяется вопросам разработки наиболее эффективного механизма передачи сигнальной информации по IP-сетям. Предлагается использовать для передачи сигнальной информации протокол Stream Control Transport Protocol (SCTP), имеющий ряд преимуществ перед протоколом TCP, основным из которых является значительное снижение времени доставки сигнальной информации и, следовательно, времени установления соединения - одного из важнейших параметров качества обслуживания.

Протокол MGCP является внутренним протоколом для обмена информацией между функциональными блоками распределенного шлюза, который извне представляется одним шлюзом. Протокол MGCP является master/slave протоколом, т.е. контроллер шлюзов является ведущим, а сам шлюз - ведомым устройством, которое должно выполнять все команды, поступающие от контроллера Call Agent. Таким образом обеспечивается масштабируемость сети и простота управления сетью через контроллер шлюзов. Шлюзы требуют меньшей производительности процессоров, и не являются интеллектуальными устройствами и, следовательно, становятся менее дорогими. Кроме того, очень быстро вводятся новые протоколы сигнализации или дополнительные услуги, так как эти изменения затрагивают только контроллер шлюзов, а не сами шлюзы.

Третий подход, предлагаемый организацией IETF (рабочая группа MEGACO), хорошо подходит для развертывания глобальных сетей IP-телефонии, приходящих на смену традиционным телефонным сетям.

4.2 Сценарий установления соединения

Рассмотрим алгоритмы установления и завершения соединения с использованием протокола MGCP. На рисунке 4.2 представлен сценарий взаимодействия протокола MGCP с протоколом ОКС7. Шлюз сигнализации SG1 и SG2 совмещены с транспортными шлюзами TGW1и TGW2 соответственно.

От телефонной станции АТС-А к шлюзу сигнализации SG1 по общему каналу сигнализации поступает запрос соединения в виде сообщения IAM протокола ISUP. Шлюз SG1 передает сообщение IAM к контроллеру шлюзов, который обрабатывает запрос и определяет, что вызов должен быть направлен к АТС-Б посредством шлюза TGW2.

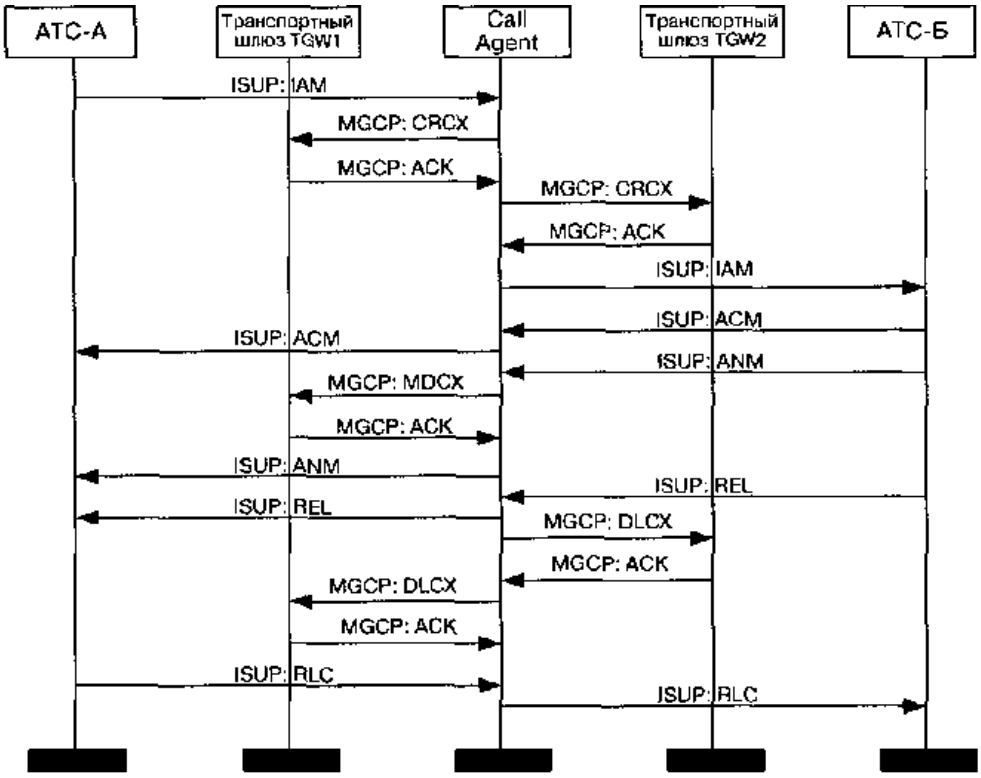


Рисунок 4.2 – Сценарий взаимодействие протокола MGCP с протоколом ОКС7

Контроллер резервирует порт шлюза TGW1 (разговорный канал). С этой целью он передает к шлюзу команду CreateConnection. Порт шлюза TGW1 может только принимать информацию (режим «recvonly»), так как он еще не осведомлен о том, по какому адресу и каким образом ему следует передавать информацию.

В ответ на эту команду шлюз TGW1 возвращает описание параметров сеанса связи. Приняв ответ шлюза TGW1, контроллер передает команду CRCX второму шлюзу TGW2 с целью зарезервировать порт в этом шлюзе.

Шлюз TGW2 выбирает порт, который будет участвовать в соединении, и подтверждает прием команды CRCX. При помощи двух команд CRCX создается однонаправленный разговорный канал для передачи вызывающему абоненту акустических сигналов или речевых подсказок и извещений. Порт шлюза TGW2 уже может не только принимать, но и передавать информацию, так как он получил описание параметров связи от встречного шлюза.

Далее контроллер шлюзов передает сообщение IAM к АТС-Б. На сообщение IAM станция АТС-Б отвечает подтверждением ACM, которое немедленно пересылается к станции АТС-А.

После того как вызываемый абонент примет вызов, АТС-Б передает к контроллеру шлюзов сообщение ANM.

Далее контроллер заменяет в шлюзе TGW1 режим «recvonly» на полнодуплексный режим при помощи команды MDCX. Шлюз TGW1 выполняет и подтверждает изменение режима.

Контроллер передает сообщение ANM к АТС-А, и далее начинается разговорная фаза соединения.

Завершение разговорной фазы происходит следующим образом. В сценарии, приведенном на рисунке 4.2, абонент Б дает отбой первым. АТС-Б передает через шлюз сигнализации сообщение REL к контроллеру шлюзов.

Приняв сообщение REL, контроллер шлюзов завершает соединение с вызванным абонентом. Шлюз подтверждает завершение соединения и передает к контроллеру собранные за время соединения статистические данные.

Контроллер шлюзов передает сообщение RLC к АТС-Б с целью подтвердить разъединение. Параллельно контроллер завершает соединение с вызвавшей стороной.

Шлюз TGW1 подтверждает завершение соединения и передает к контроллеру статистические данные сеанса

АТС-А подтверждает завершение соединения передачей сообщения RLC, после чего соединение считается завершенным.

Следует заметить, что алгоритм взаимодействия протоколов SIP и MGCP не сильно отличается от вышеописанного алгоритма.

Рабочая группа MEGACO комитета IETF продолжает работу по усовершенствованию протокола управления шлюзами, в рамках которой разработан более функциональный, чем MGCP, протокол MEGACO.

Международный союз электросвязи в проекте версии 4 рекомендации H.323 ввел принцип декомпозиции шлюзов. Управление функциональными блоками распределенного шлюза будет осуществляться контроллером шлюза - Media Gateway Controller - при помощи адаптированного к H.323 протокола MEGACO, который в рекомендации H.248 назван Gateway Control Protocol.

Сообщения протокола MEGACO отличаются от сообщений протокола MGCP, но процедуры установления и завершения соединений с использованием обоих протоколов идентичны и рассмотрены в [18].

4.3 Сравнение подходов к построению сети IP-телефонии

В настоящее время для построения хорошо функционирующих и совместимых с ТфОП сетей IP-телефонии подходят протоколы H.323 и MGCP. Протокол SIP несколько хуже взаимодействует с системами сигнализации, используемыми в ТфОП.

Подход, основанный на использовании протокола MGCP, обладает весьма важным преимуществом перед подходом, предложенным ITU в рекомендации H.323:

- поддержка контроллером шлюзов сигнализации ОКС7 и других видов сигнализации;
- прозрачная трансляция сигнальной информации по сети IP-телефонии.

В сети, построенной на базе рекомендации H.323, сигнализация ОКС7, как и любая другая сигнализация, конвертируется шлюзом в сигнальные сообщения H. 225.0(0.931).

Основным недостатком сети с использованием MGCP является незаконченность стандартов. Функциональные составляющие распределенных шлюзов, разработанные разными фирмами-производителями телекоммуникационного оборудования, практически несовместимы. Функции контроллера шлюзов точно не определены. Не стандартизированы механизмы переноса сигнальной информации от шлюза сигнализации к контроллеру и в обратном направлении. К недостаткам можно отнести также отсутствие стандартизированного протокола взаимодействия между контроллерами. Кроме того, протокол MGCP является протоколом управления шлюзами, но не предназначен

для управления соединениями с участием терминального оборудования пользователей (IP-телефонов). Это означает, что в сети, построенной на базе протокола MGCP, для управления терминальным оборудованием должен присутствовать привратник или сервер SIP.

Стоит также отметить, что в существующих приложениях IP-телефонии, таких как предоставление услуг международной и междугородной связи, использовать протокол MGCP (так же, как и протокол SIP) нецелесообразно в связи с тем, что подавляющее количество сетей IP-телефонии сегодня построено на базе протокола H.323. Оператору придется строить отдельную сеть IP-телефонии на базе протокола MGCP (или SIP), что связано со значительными капиталовложениями. В то же время, оператор связи, имеющий оборудование стандарта H.323, может присоединиться к существующим сетям IP-телефонии.

В проекте версии 4 рекомендации H.323 ITU-T ввел принцип декомпозиции шлюзов, использованный в сети на базе MGCP. Управление функциональными блоками распределенного шлюза будет осуществляться контроллером шлюза - MGC (Media Gateway Controller) при помощи ряда протоколов. Кроме того, предусмотрена также возможность прозрачной передачи сигнализации ОКС7 и других видов сигнализации по сетям IP-телефонии и обработка сигнализации всех видов привратником без преобразования в сигнальные сообщения H.225.0.

Контрольные вопросы

1) Перечислите основные элементы сети на базе MGCP.
2) Основное назначение протокола MGCP.
3) Каковы основные преимущества MGCP перед подходом, предложенным ITU в рекомендации H.323?
4) Перечислите основные недостатки сети с использованием MGCP.
5) В чем заключается принцип декомпозиции шлюзов?
6) Поясните функции основных блоков MG, SG, Call Agent.
7) Функции какого блока могут быть распределены между несколькими компьютерными платформами?
8) Каким образом создается однонаправленный разговорный канал для передачи вызывающему абоненту акустических сигналов, извещений?
9) Какой протокол транспортного уровня используется для переноса сообщений протокола MGCP?
10) Какие протоколы используются для синхронизации работы контроллеров в сети MGCP?

Раздел 5 Технология MPLS

5.1 Определение, назначение

Стандарт Multi-Protocol Label Switching (MPLS), предложенный Internet Engineering Task Force (IETF), предназначен для дальнейшего масштабирования Internet. Эта задача решается путем присоединения специальных меток к пакетам IP. Метки позволяют маршрутизаторам и коммутаторам пересылать трафик дальше на основе содержащейся в них информации, не анализируя внутренние поля каждого пакета.

Система меток, которыми снабжаются IP-пакеты в соответствии со спецификацией MPLS, ускоряет передачу корпоративного трофика no Internet, обеспечивает гарантированные уровни качества обслуживания в IP- и других сетях, снижает общие издержки на использование глобальных сетей. Схему срочной доставки пакетов можно представить следующим образом:

-пакет покидает корпоративную локальную сеть;

- оконечный маршрутизатор поставщика услуг снабжает пакет 20-битной меткой, содержащей информацию о маршруте, месте назначения и приоритете;

- пакет передается по Internet с меткой, которую считывают только оконечные устройства каждого из MPLS-доменов;

- пакет достигает оконечного устройства в сети назначения быстрее и в соответствии с классом качества обслуживания.

Спецификации MPLS разрабатывались для организации управления пакетами в территориально-распределенных сетях тем же способом, которым осуществляется манипулирование каналами в сетях ATM и управление маршрутами ячеек. Однако главное преимущество MPLS заключается в том, что клиенты могут обслуживать трафик точно также, как в сетях ATM, не прибегая при этом к сложным средствам ATM и не используя функции повышения качества обслуживания и структуру виртуальных каналов. В стандарте MPLS лучшим способом достигается сочетание высокой пропускной способности ATM с простотой и привычностью протокола IP. Корпорации получают возможность повысить качество обслуживания линий связи IP, управляя при помощи средств MPLS пересылкой пакетов по заранее определенным каналам и маршрутам, что обеспечивает нужную производительность для приложений, чувствительных к задержкам. Данный метод принципиально отличается от традиционного способа управления трафиком IP, когда пропускная способность и маршрут передачи информации заранее не известны и, таким образом, производительность становится совершенно непредсказуемой.

К достоинствам MPLS следует отнести упрощение топологии сетей с большим числом маршрутов. MPLS помогает уменьшить число уровней иерархии и избежать чрезмерной сложности процедур контроля, управления и обслуживания сети.

Еще одной положительной чертой MPLS является простота управления трафиком, не зависящая от особенностей конкретных компонентов, отвечающих за качество обслуживания. Контроль за информационными потоками в сетях MPLS реализован более эффективно и позволяет избежать перегрузок, устранить ограничения пропускной способности и исключить избыточные связи.

Однако далеко не всем крупным корпорациям требуется данная возможность, в оптимизации управления трафиком нуждаются лишь частные территориально-распределенные сети, построенные на основе смешанной топологии. Двухточечное соединение и сеть типа «звезда» и так поддерживают различные маршруты.

5.2 Схема коммутации

В основе MPLS лежит принцип обмена меток. Любой передаваемый пакет ассоциируется с тем или иным *классом эквивалентности пересылки* (Forwarding Equivalence Class, FEC), каждый из которых идентифицируется определенной меткой. Пакеты снабжаются метками - идентификаторами небольшой и фиксированной длины, которые определяют принадлежность каждого пакета тому или иному FEC. К одному FEC относятся пакеты всех потоков, пути следования которых через сеть MPLS (или через часть этой сети) совпадают, т.е. с точки зрения выбора очередного маршрутизатора пакеты этих потоков неразличимы. Таким образом, каждый FEC имеет свою систему меток.

Значение метки уникально лишь для участка пути между соседними узлами сети MPLS, которые называются также *маршрутизаторами, коммутирующими по меткам* (Label Switching Router, LSR). Метка передается в составе любого пакета, причем способ ее привязки к пакету зависит от используемой технологии канального уровня.

Маршрутизатор LSR получает топологическую информацию о сети, участвуя в работе алгоритма маршрутизации — OSPF, BGP, IS-IS. Затем он начинает взаимодействовать с соседними маршрутизаторами, распределяя метки, которые в дальнейшем будут применяться для коммутации. Обмен метками может производиться с помощью как *специального протокола распределения меток (*Label Distribution Protocol, LDP), так и модифицированных версий других протоколов сигнализации в сети (например, незначительно видоизмененных протоколов маршрутизации, резервирования ресурсов RSVP и др.).

Распределение меток между LSR приводит к установлению внутри домена MPLS *путей с коммутацией по меткам* (Label Switching Path, LSP). Каждый маршрутизатор LSR содержит таблицу, которая ставит в соответствие паре «входной интерфейс, входная метка» тройку «префикс адреса получателя, выходной интерфейс, выходная метка». Получая пакет, LSR по номеру интерфейса, на который пришел пакет, и по значению привязанной к пакету метки определяет для него выходной интерфейс. Значение префикса применяется лишь для построения таблицы и в самом процессе коммутации не используется. Старое значение метки заменяется новым, содержавшимся в поле «выходная метка» таблицы, и пакет отправляется к следующему устройству на пути LSP.

Вся операция требует лишь одноразовой идентификации значений полей в одной строке таблицы. Это занимает гораздо меньше времени, чем сравнение IP-адреса отправителя с наиболее длинным адресным префиксом в таблице маршрутизации, которое используется при традиционной маршрутизации.

Сеть MPLS делится на две функционально различные области — ядро и граничную область (рисунок 5.1).

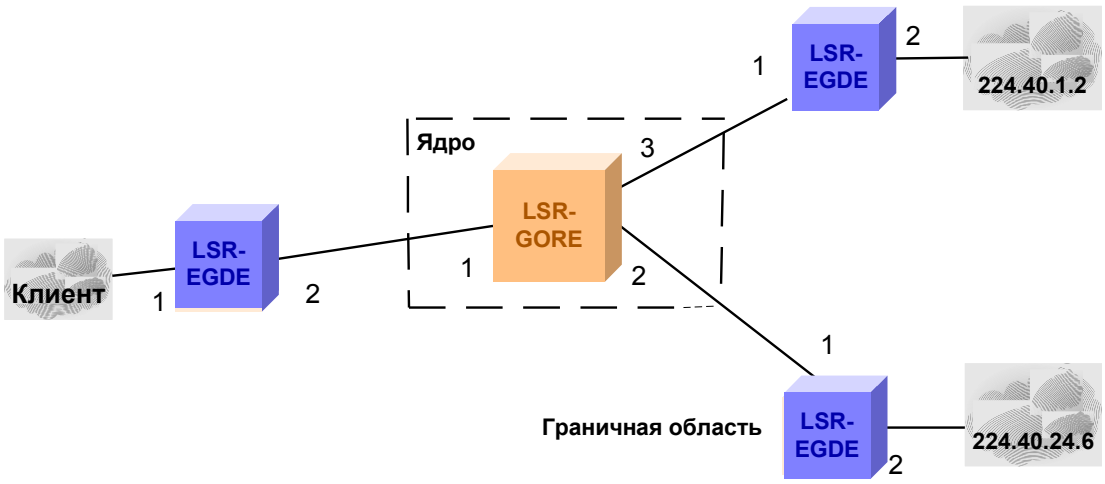


Рисунок 5.1 – Структура MPLS

Ядро образуют устройства, минимальным требованием к которым является поддержка MPLS и участие в процессе маршрутизации трафика для того протокола, который коммутируется с помощью MPLS. Маршрутизаторы ядра занимаются только коммутацией. Все функции классификации пакетов по различным FEC, а также реализацию таких дополнительных сервисов, как фильтрация, явная маршрутизация, выравнивание нагрузки и управление трафиком, берут на себя граничные LSR. В результате интенсивные вычисления приходятся на граничную область, а

высокопроизводительная коммутация выполняется в ядре, что позволяет оптимизировать конфигурацию устройств MPLS в зависимости от их местоположения в сети.

Таким образом, главная особенность MPLS — отделение процесса коммутации пакета от анализа IP-адресов в его заголовке. Очевидным следствием такого подхода является то, что очередной сегмент LSP может не совпадать с очередным сегментом маршрута, который был бы выбран при традиционной маршрутизации.

На установление соответствия пакетов определенным классам FEC могут влиять не только IP-адреса, но и другие параметры. Поэтому для пакетов, относящихся к различным потокам RSVP или имеющих разные приоритеты обслуживания, нетрудно реализовать, например, назначение различных путей коммутации по меткам. Каждый из классов FEC обрабатывается отдельно от остальных не только потому, что для него строится свой путь LSP, но и в смысле доступа к общим ресурсам (полосе пропускания канала и буферному пространству).

В результате технология MPLS позволяет очень эффективно поддерживать требуемое качество обслуживания, не нарушая предоставленных пользователю гарантий. Применение в LSR механизмов управления буферизацией и очередями, дает возможность оператору сети MPLS контролировать распределение ресурсов и изолировать трафик отдельных пользователей.

5.3 Элементы архитектуры

5.3.1 Метки и способы маркировки

Метка — это короткий идентификатор фиксированной длины, который определяет класс FEC. По значению метки пакета определяется его принадлежность определенному классу на каждом из участков коммутируемого маршрута.

Метка должна быть уникальной лишь в пределах соединения между каждой парой логически соседних LSR. Поэтому одно и то же ее значение может использоваться LSR для связи с различными соседними маршрутизаторами, если только имеется возможность определить, от какого из них пришел пакет с данной меткой. Другими словами, в соединениях «точка—точка» допускается применять один набор меток на интерфейс, а для сред с множественным доступом необходим один набор меток на модуль или все устройство.

Перед включением в состав пакета метка определенным образом кодируется. В случае использования протокола IP она помещается в специальный «тонкий» заголовок пакета, инкапсулирующего IP. В других ситуациях метка записывается в заголовок протокола канального уровня или кодируется в виде определенного значения VPI/VCI (в

сети ATM). Для пакетов протокола IPv6 метку можно разместить в поле идентификатора потока.

5.3.2 Стек меток

В рамках архитектуры MPLS вместе с пакетом разрешено передавать не одну метку, а целый их стек. Операции добавления/изъятия метки определены как операции на стеке (push/pop). Результат коммутации задает лишь верхняя метка стека, нижние же передаются прозрачно до операции изъятия верхней. Такой подход позволяет создавать иерархию потоков в сети MPLS и организовывать туннельные передачи. Стек состоит из произвольного числа элементов, каждый из которых имеет длину 32 бита: 20 бит составляют собственно метку, 8 отводятся под счетчик времени жизни пакета, один указывает на нижний предел стека, а три не используются. Метка может принимать любое значение, кроме нескольких зарезервированных.

5.3.3 Компоненты коммутируемого маршрута

Коммутируемый путь (LSP) одного уровня состоит из последовательного набора участков, коммутация на которых происходит с помощью метки данного уровня. Например, LSP нулевого уровня проходит через устройства LSR0, LSR1 LSR3, LSR4 и LSR5, при этом LSR0 и LSR5 являются, соответственно, входным (ingress) и выходным (egress) маршрутизаторами для пути нулевого уровня. Маршрутизаторы LSRl и LSR3 играют ту же роль для LSP первого уровня; первый из них производит операцию добавления метки в стек, а второй — ее изъятия. С точки зрения трафика нулевого уровня, LSP первого уровня является прозрачным туннелем. В любом сегменте LSP можно выделить верхний и нижний LSR по отношению к трафику. Например, для сегмента LSR4 — LSR» четвертый маршрутизатор будет верхним, а пятый - нижним.

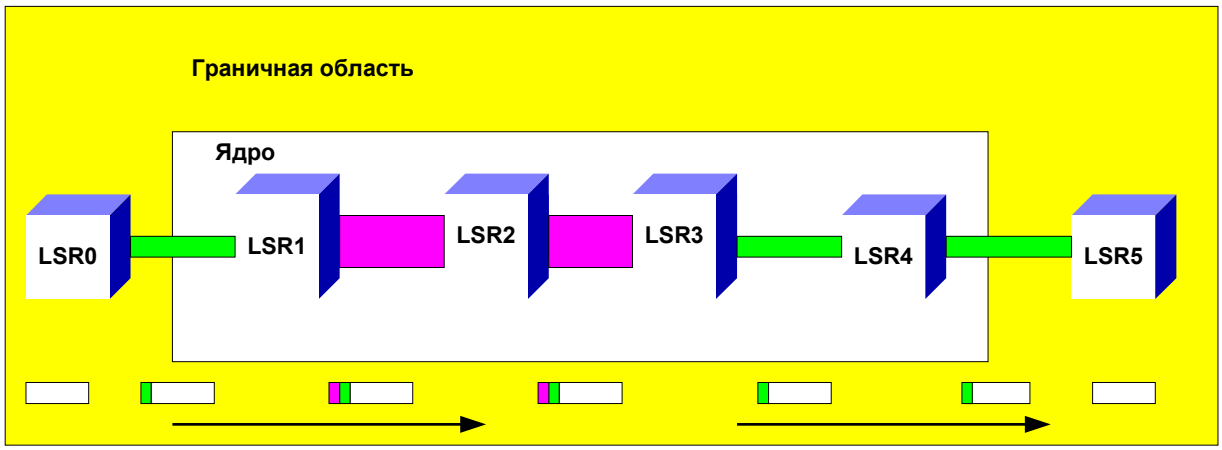


Рисунок 5.2- Компоненты коммутируемого соединения

5.3.4 Привязка и распределение меток

Под привязкой понимают соответствие между определенным классом FEC и значением метки для данного сегмента LSP. Привязку всегда осуществляет «нижний» маршрутизатор LSR, поэтому и информация о ней распространяется только в направлении от нижнего LSR к верхнему. Вместе с этими сведениями могут предаваться атрибуты привязки.

Обмен информацией о привязке меток и атрибутах осуществляется между соседними LSR с помощью протокола распределения меток. Архитектура MPLS не зависит от конкретного протокола, поэтому в сети могут применяться разные протоколы сетевой сигнализации. Очень перспективно в данном отношении - использование RSVP для совмещения резервирования ресурсов и организации LSP для различных потоков.

Существуют два режима распределения меток: независимый и упорядоченный:

- первый предусматривает возможность уведомления верхнего узла о привязке до того, как конкретный LSR получит информацию о привязке для данного класса от своего нижнего соседа;
- второй режим разрешает высылать подобное уведомление только после получения таких сведений от нижнего LSR, за исключением случая, когда маршрутизатор LSR является выходным для этого PEC.

Распространение информации о привязке может быть инициировано запросом от верхнего устройства LSR (downstream on-demand) либо осуществляться спонтанно (unsolicited downstream).

5.4 **Построение коммутируемого маршрута по протоколу LDP**

Рассмотрим, как система MPLS автоматически создает путь LSP в простейшем случае — с помощью протокола LDP. Архитектура MPLS не требует обязательного применения LDP, однако, в отличие от других возможных вариантов, он наиболее близок к окончательной стандартизации.

На первом этапе коммутирующие маршрутизаторы посредством многоадресной рассылки сообщений UDP определяют свое «соседство» (adjacency) в рамках протокола LDP.

Затем два соседних маршрутизатора обмениваются информацией для инициализации, в том числе о желательной версии протокола, режимах функционирования, продолжительности тайм-аутов, диапазонах меток и поддерживаемых типах.

После того как обе стороны согласятся об общем подмножестве функциональных параметров, пространство меток считается готовым к использованию, и процедура

назначения меток может быть начата. Кроме близости на канальном уровне, LDP может устанавливать связь между «логически соседними» LSR, не принадлежащими к одному каналу. Это необходимо для реализации туннельной передачи.

Далее LDP открывает транспортное соединение между участниками сеанса поверх TCP. По этому соединению передаются запросы на установку привязки и сама информация о привязке. Кроме того, участники сеанса периодически проверяют работоспособность друг друга, отправляя тестовые сообщения (keep alive message). Если после получения последнего контрольного сообщения проходит слишком много времени, то соединение разрывается.

На рисунке 5.3 приведен пример построения коммутируемого пути по протоколу LDP. Заполнение таблиц меток по протоколу LDP происходит следующим образом. Предположим, что выбран упорядоченный режим распределения меток LSP со спонтанным распространением сведений о привязке.

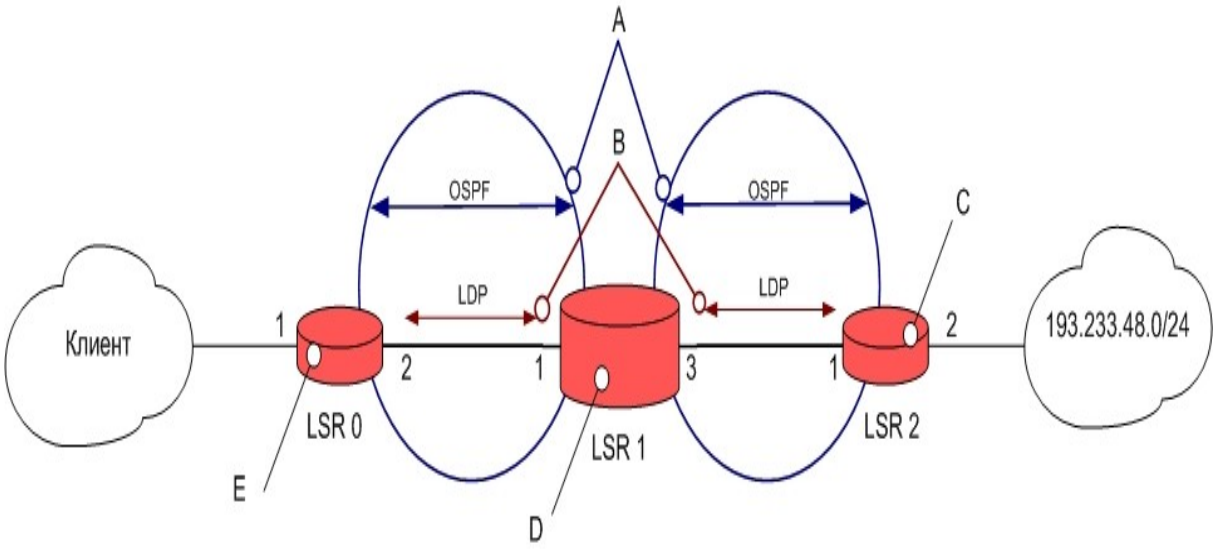


Рисунок 5.3- Построения коммутируемого пути по протоколу LDP

На стадии A каждое из устройств сети MPLS строит базу топологической информации, используя любой из современных протоколов маршрутизации, например OSPF.

На стадии B маршрутизаторы LSR применяют процедуру нахождения соседних устройств и устанавливают с ними сеансы LDP.

На стадии C LSR2 на основе анализа собственных таблиц маршрутизации обнаруживает, что он является выходным LSR для пути, ведущего к IP-сети 193.233.48.0. Тогда LSR2 ассоциирует класс FEC с пакетами, адрес получателя которых соответствует префиксу данной сети, и присваивает этому классу случайное значение метки - в нашем

случае 18. Получив привязку, протокол LDP уведомляет верхний маршрутизатор LSR (LSR1) о том, что потоку, адресованному сети с префиксом 193.233.48, присвоена метка 18. LSR1 помещает это значение в поле выходной метки своей таблицы.

На стадии D устройство LSR1, которому известно значение метки для потока, адресованного на префикс 193.233.48, присваивает собственное значение метки данному FEC и уведомляет верхнего соседа (LSR0) об этой привязке. Теперь LSR0 записывает полученную информацию в свою таблицу. После завершения данного процесса все готово для передачи пакетов из сети «клиента» в сеть с адресом 193.233.48,0, т.е. по выбранному пути LSP.

Спецификация класса FEC может содержать несколько компонентов, каждый из которых определяет набор пакетов, соответствующих данному классу. На сегодняшний день определены два компонента FEC: адрес узла (host address) и адресный префикс (address prefix). Пакет классифицируется как принадлежащий к данному классу FRC, если адрес получателя точно совпадает с компонентом адреса узла либо имеет максимальное совпадение с адресным префиксом. В рассмотренном примере узел LSR0 выполняет в процессе передачи классификацию пакетов, поступающих к нему из сети клиента, и (если адрес получателя в них совпадает с префиксом 193.233.48), присвоив пакету метку 33, отправляет его через интерфейс 2.

5.5 Преимущества технологии MPLS

Технология MPLS имеет ряд преимуществ:

- разделение функциональности между ядром и граничной областью сети;
- эффективное использование явного маршрута;
- отделение выбора маршрута от анализа IP-адреса дает возможность предоставлять широкий спектр дополнительных сервисов при сохранении масштабируемости сети);
- ускоренная коммутация (сокращает время поиска в таблицах);
- гибкая поддержка QoS, интегрированных сервисов и виртуальных частных сетей;
- сохранение инвестиций в установленное ATM –оборудование.

В настоящее время существуют два основных способа создания магистральных IP-сетей: с помощью IP-маршрутизаторов, соединенных каналами «точка—точка», либо на базе транспортной сети ATM, поверх которой работают IP-маршрутизаторы. Применение MPLS оказывается выгодным в обоих случаях. В магистральной сети ATM оно дает возможность одновременно предоставлять клиентам как стандартные

сервисы ATM, гак и широкий спектр услуг IP-сетей вместе с дополнительными услугами. Такой подход существенно расширяет пакет услуг провайдера, заметно повышая его конкурентоспособность. Тандем IP и ATM, соединенных посредством MPLS, способствует еще большему распространению этих технологий и создает основу для построения крупномасштабных сетей с интеграцией сервисов.

5.6 Проблемы перехода к мультисервисным сетям

На первый взгляд, для построения мультисервисной сети необходимо объединить существующие сети разных операторов (традиционные ТфОП, сети мобильной связи и IP-сети) в единую сеть. Это можно назвать конвергенцией существующих сетей разных операторов и технологий. Однако сегодня еще нет технологий, которые бы полностью удовлетворяли запросам перспективной мультисервисной сети. В то же время, технологические решения, способные стать ее основой, существуют уже сейчас, т. е. можно построить прообраз мультисервисной сети. В соответствии с устоявшейся терминологией назовем промежуточную сеть-прообраз конвергентной (К-сетью), а мультисервисную сеть следующего поколения - М-сетью.

Конвергентная сеть должна удовлетворять основным требованиям:

1) обеспечение передачи данных,

2) обеспечение передачи трафика реального времени,

3) обеспечение гарантированного качества обслуживания QoS,

4) управление современными сетями.

В качестве первого шага для транспортной сети следует подобрать такую технологию, которая удовлетворяла бы первым трем требованиям. Кроме этого, сеть должна «вписаться» в сегодняшнюю сетевую инфраструктуру и иметь возможность совершенствоваться. Логичнее всего использовать наиболее развитую сегодня технологию, т. е. TCP/IP, и взять за основу протокол IP. Он удовлетворяет первому требованию и благодаря технологии VoIP отвечает второму. Но протокол IP абсолютно не обеспечивает гарантированное качество обслуживания.

Рассмотрим технологию ATM – она, с одной стороны, удовлетворяет всем вышеназванным требованиям. Но, с другой стороны, возникает проблема стоимости оборудования ATM, значительно превышающей стоимость оборудования IP, и развитость сетей на основе IP сильно выигрывает по сравнению с ATM-сетями.

Рассматривая сегодняшние тенденции построения сетей, можно сделать вывод, что несомненным лидером в транспортных технологиях является MPLS. Сети MPLS строятся, в том числе и на территории СНГ. Технология MPLS вполне удовлетворяет перечисленным требованиям и лишена недостатков ATM. Стоимость оборудования

MPLS, по сравнению с ATM, ниже, а метод коммутации по меткам реализуется весьма удачно. Реализуется передача данных любого вида, MPLS может работать и «поверх» IP, и «поверх» ATM, обеспечивается гарантированное качество обслуживания. Разработки оборудования ведутся многими ведущими компаниями, например Cisco Systems, Alcatel, Avaya и др. Таким образом, реализация проекта транспортной сети на базе MPLS не должна вызвать особых затруднений.

В качестве второго шага создания мультисервисной сети необходимо рассмотреть варианты организации доступа. В конвергентной сети будут только те терминалы, которые есть сегодня - персональные компьютеры, телефонные аппараты, IP-телефоны. И, как следствие, в узле доступа должны быть реализованы технологии для любого терминального устройства: оптоволокно, например технология PON; радиодоступ - DECT, Bluetooth, Radio Ethernet и HIPERLAN2 - и, конечно же, доступ по медной паре, в частности, по технологии ISDN или xDSL. Безусловно, ни одна из перечисленных технологий не может в полной мере удовлетворить потребности мультисервисного доступа. Необходим некий абонентский концентратор, объединяющий все эти технологии.

Следующей задачей создания М-сети является обеспечение качества обслуживания. В сети MPLS эта проблема решается, но прежде чем передаваемая информация поступит в сеть MPLS, она будет находиться в общей IP- сети, не поддерживающей гарантированное QoS. Для обеспечения должного качества обслуживания трафика речевых и видеоприложений, необходим механизм, позволяющий приложениям информировать сеть о своих требованиях. На основе этой информации сеть может резервировать ресурсы или отказать приложению, вынуждая его либо пересмотреть требования, либо отложить сеанс связи.

Наиболее подходящим решением здесь можно считать протокол резервирования ресурсов RSVP (Resource Reservation Protocol). RSVP - это протокол сигнализации, который обеспечивает резервирование ресурсов для предоставления в IP-сетях услуг эмуляции выделенных каналов. Протокол позволяет запрашивать, например, гарантированную пропускную способность такого канала, предсказуемую задержку, максимальный уровень потерь. Но резервирование выполняется лишь в том случае, если имеются требуемые ресурсы.

Таким образом, совместное использование двух протоколов - RSVP на уровне доступа и MPLS на уровне транспортной сети - позволит предоставить пользователям конвергентной сети гарантированное качество обслуживания.

На начальном этапе ТфОП станет частью конвергентной сети, а на стыках между ТфОП и сетью IP/MPLS будут устанавливаться VoIP шлюзы - устройства, которые предназначены для преобразования речевой информации, поступающей со стороны ТфОП, в вид, пригодный для передачи по IP-сетям, и наоборот. Кроме того, в конвергентную сеть войдут сети IP-телефонии альтернативных операторов, построенные, например, на протоколах H.323 и SIP. Сегодня такие сети используются, в основном, для междугородной и международной связи, но в условиях конвергентной сети они станут альтернативой ТфОП.

Контрольные вопросы

1) Назначение стандарта MPLS.
2) В чем заключается главная особенность MPLS?
3) Назовите основные преимущества технологии MPLS.
4) Назовите основные функции ядра и граничной области сети MPLS.
5) Поясните термин «метка».
6) Что понимается под стеком меток?
7) Поясните механизм назначения меток.
8) Назовите способы распространения информации о привязке.
9) Что понимается под классом эквивалентности передачи?
10) Поясните термин «маршрутизатор, коммутирующий по меткам».
11) Что включает в себя LDP?
12) Перечислите этапы построения коммутируемого маршрута по протоколу LDP
13) Каким требованиям должна удовлетворять конвергентная сеть?
14) Перечислите основные задачи, решаемые при создании М-сетей.
15) Назовите технологию, которая удовлетворяла бы требованиям К-сети.
16) Какие протоколы обеспечивают гарантированное качество обслуживания пользователей К-сети?

ЧАСТЬ 2. ТЕХНОЛОГИЯ .NET

Раздел 1. Технология .Net

1.1 Предпосылки к созданию .Net

Миллионы людей в повседневной жизни, где бы они ни находились и какими бы устройствами ни пользовались, хотят получать все больше разнообразных услуг, основанных на современных информационных технологиях (включая новости, электронные магазины, онлайновые платежи, юридические консультации, страховки, офисные приложения и многое другое). Современное развитие Интернет и ее влияние на современную корпоративную культуру очевидно, дает такие возможности – у специалистов появились широкие возможности получения и использования деловой информации; возникли новые модели бизнеса (электронные биржи, электронные системы бронирования и продажи билетов, системы интеграции каналов сбыта, сообщества поставщиков и потребителей, электронные аукционы и др.), позволяющие сочетать глобальный охват рынка с персональным подходом к каждому клиенту. Руководители большинства предприятий на практике увидели, что информационные технологии непосредственно влияют на эффективность работы. Развертывание и внедрение сложной информационной системы сегодня обходится существенно дешевле и может быть выполнено значительно (иногда - в несколько раз) быстрее, чем это было еще несколько лет назад [21].

За последние годы произошло значительное увеличение пропускной способности сетей благодаря реализации многочисленных высокоскоростных каналов. Впервые появилась возможность организации по-настоящему распределенных систем, обслуживающих миллионы пользователей, например, приложение Napster представляет собой многофункциональный клиент, который взаимодействует со службой каталогов в Интернете и использует в качестве серверов компьютеры других пользователей этого приложения. Еще один пример распределенного приложения — система мгновенного обмена сообщениями, где многофункциональный клиент использует находящийся в Интернете «список приятелей» и взаимодействует с другими клиентами (Instant Messenger и Windows) в сети.

В то же время внедрение интернет-технологий в корпорациях проходило не всегда гладко. Упрощенные или узкоспециализированные решения, не способные адаптироваться к реальной рыночной ситуации, расти вместе с развитием бизнеса, взаимодействовать с другими информационными системами, вызывали немало разочарований. Стала очевидна настоятельная потребность в надежной

масштабируемой платформе для создания новых электронных интерактивных сервисов. Она должна:

- быть доступной самому широкому кругу предприятий — от малого бизнеса до транснациональных корпораций;
- отличаться гибкостью и широтой функций;
- работать на наиболее распространенном оборудовании;
- постоянно развиваться;
- быть удобной в изучении, развертывании и сопровождении.

Задача создания подобной платформы уникальна в целом ряде отношений:

- требуется опыт разработки самых различных решений - от серверных операционных систем и баз данных до офисных приложений и интернет-обозревателей;

- ее компоненты должны строиться в соответствии с открытыми стандартами и уметь взаимодействовать с другими внешними системами - от систем корпоративного планирования на мэйнфреймах и хранилищ данных на кластерах RISC/Unix до пользовательских интерфейсов карманных компьютеров и сотовых телефонов;

- ИТ-специалисты должны сосредоточиться на создании новых сервисов, освободившись от узкоспециальных работ (масштабировании, обеспечение надежности и совместимости низкоуровневых протоколов);

- создаваемые технологии должны использоваться для развертывания многих критически важных приложений в самых разных областях.

Для скорейшего развития распределенных систем нового поколения должны быть выполнены три условия:

1) Web-службы. Первое условие заключается в том, что все компоненты системы должны быть реализованы в виде веб-служб. Это в равной степени относится к компонентам программного обеспечения и к сетевым ресурсам (например, хранилищам).

2) Объединение и интеграция. Вторым условием является наличие простых и удобных способов объединения и интеграции веб-служб.

3) Простота и удобство работы пользователя. Третье условие — это наличие простой и удобной рабочей среды для конечных пользователей и потребителей.

Корпорация Microsoft считает, что платформа .NET позволит выполнить эти условия.

1.2 Определение и компоненты платформы .NET

Платформа .NET стимулирует развитие распределенных систем нового поколения и строится на признанных концепциях и стандартах. Платформу .NET образуют пять компонентов (рисунок 1.1): средства разработки; серверные системы; службы .NET Building Block Services — «строительные блоки»; программное обеспечение для устройств; специализированные рабочие среды. (реализованы в виде приложений на платформе .NET).

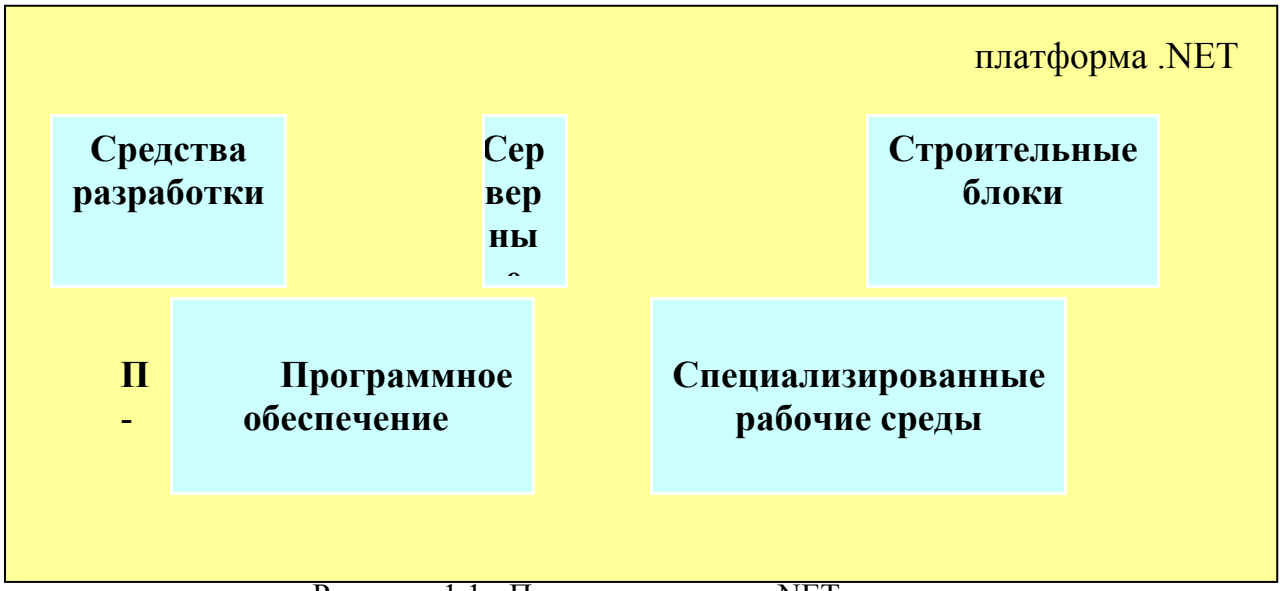


Рисунок 1.1 - Пять компонентов .NET

Первый компонент упрощает создание веб-служб. Он представлен платформой .NET Framework и набором инструментальных средств Visual Studio. .NET Framework и Visual Studio .NET обеспечивают самый простой, быстрый и эффективный способ разработки веб-служб.

Вторым компонентом .NET является набор серверов, отвечающих за объединение и интеграцию веб-служб. Эти серверные системы можно условно разделить на две категории. Первая включает программные продукты, которые обеспечивают базовые средства для работы с XML, например, Windows 2000, SQL Server 2000 и Exchange 2000. Использование языка XML является самым простым и «открытым» способом интеграции веб-служб. Вторая категория включает специальные серверные системы (такие как BizTalk Server), которые обеспечивает эффективные и универсальные возможности объединения и интеграции. Например, сервер BizTalk Server 2000 предлагает встроенный язык XLANG, позволяющий определять бизнес-процессы, транзакции и контракты и обеспечивающий глубокую интеграцию разнородных сред.

Третьим компонентом платформы .NET является набор служб, играющих роль «строительных блоков» (Building Block Service), которые повышают простоту и

удобство работы пользователя. Сегодня пользователям часто приходится вводить одни и те же учетные данные для доступа к веб-узлам и приложениям. Создатели .Net работают над созданием небольшого набора служб (таких как службы идентификации, оповещения и схематизированные хранилища), которые значительно упростят переход от одной службы к другой или переход из одной среды в другую. Такая интеграция имеет ключевое значение в мире распределенных вычислительных систем.

Службы — «строительные блоки» предлагают широкие возможности не только пользователям, но и разработчикам, позволяя им использовать готовые службы, доступные через Интернет.

Четвертый компонент платформы .NET представлен набором программного обеспечения для устройств и клиентских систем. Его роль заключается в том, чтобы предложить пользователю удобную и интегрированную среду для работы. Платформа .NET предполагает использование не одного устройства или клиента, а целого семейства дополняющих друг друга устройств. Объединяет эти устройства то, что все они являются «интеллектуальными». Чтобы поддержка этих устройств стала возможной, для них создается программное обеспечение с новыми функциями, которые делают работу пользователя более удобной и эффективной. Они запоминают вашу личную информацию и используют в качестве платформы для обработки данных Web-, а не отдельные серверы.

Пятым компонентом платформы .NET являются удобные рабочие среды, ориентированные на определенную категорию пользователей, которые интегрируют Web-службы и объединяют различные функциональные возможности. Предлагается несколько таких сред:

- MSN для потребителей;
- bCentral для предприятий малого бизнеса;
- Office для офисных работников;
- Visual Studio .NET для разработчиков.

Наиболее впечатляющая и наиболее обозримая часть .NET — новое семейство корпоративных серверов. Состав продуктов семейства .NET Enterprise Servers подобран так, чтобы удовлетворить в равной степени как практиков, развертывающих и сопровождающих информационные системы современных предприятий, так и теоретиков, заботящихся об элегантности концепций и перспективах развития отрасли.

Модульная структура .NET и ориентация на открытые стандарты обеспечивают широкий выбор «строительных блоков» и удобство сборки из них корпоративных

систем. Новая архитектура .NET гарантирует масштабируемость, необходимую самым крупным предприятиям.

1.3 Универсальные «строительные блоки»

Целый ряд разрабатываемых Microsoft сервисов предназначен для решения задач, типичных для большинства интранет— и интернет-систем. Появление стандартных сервисов не только значительно снизит трудоемкость разработки корпоративных информационных систем нового поколения, но и позволит создавать принципиально новые продукты и области бизнеса, в которых поставщики и потребители взаимодействуют через интернет.

Концепция «компонентного» программирования является развитием «объектно-ориентированного» подхода и позволяет создавать сложные информационные системы из многократно используемых «строительных блоков». Можно выделить следующие блоки:

- *Identity* - аутентификация, построенная на средствах Windows .NET (Whistler) и Passport .NET. Поддержка карточек доступа и биометрических устройств;

- *Notification and Messaging* - интегрированная поддержка всех типов сообщений, от корпоративных и интернет e-mail до факсов и SMS;

- *Personalization* - управление доступом и синхронизацией на основе групп, ролей и правил;

- *XML Store* - использование XML и SOAP для определения контента и функциональности, контроля полноты и корректности транзакций, взаимодействия с SQL Server, MSN Communities, NTFS и т.д.

1.4 Новые возможности среды

Используя интернет, сотни миллионов пользователей освоили базовые навыки работы с компьютерами и познакомились с несколькими важными метафорами — гипертекста, электронного почтового сообщения, адреса в интернете. Это хорошая база для построения на ее основе приложений и сервисов с интерфейсами, не требующими длительного освоения. В то же время Интернет нового поколения предполагает, что пользователи станут намного более активными. К новым возможностям следует отнести:

1) Natural Interface - технологии, обеспечивающие «естественное» общение с компьютером, включая распознавание и синтез речи, рукописный ввод и другие, сделают наиболее удобным использование самых различных устройств.

2) Universal Canvas -новая основанная на XML-схемах архитектура, превращающая Интернет из платформы «для чтения» в платформу для «чтения/записи». Кроме того, поддерживает для документа: создание, объединение источников, чтение, редактирование, комментирование, аннотирование, анализ.

3) Information Agent - представитель пользователя в Интернет, имеет историю, привычки и потребности конкретного пользователя, позволяет Интернет-сайтам и другим сервисам оптимально приспособиться к нему. Поддерживает P3P-технологии (приватные личные контакты). В отличие от существующих технологий персонализации, Information Agent остается под полным контролем пользователя.

4) Smarttags -обобщение технологии Intellisense на Web-контент. Позволяет «предугадывать» и корректировать действия пользователя. Наиболее известные примеры использования технологии Intellisense: «помощники» в Office и Internet Explorer, автозамена, автоформат, автозаполнение и т.д.

.NET предусматривает самые разнообразные способы использования различных цифровых устройств, но всегда, когда захочет пользователь, они «будут выглядеть как традиционный Web».

1.5 Сервисы и продукты на Платформе .NET

Microsoft .NET предлагает исключительно гибкую архитектуру сервисов. Они могут обеспечиваться отдельным компьютером, сервером в локальной сети, одним или несколькими серверами в Интернет.

Идея перейти от продажи приложений к подписке и аренде возникла уже достаточно давно, однако ее широкое внедрение тормозилось как консерватизмом потребителей, так и отсутствием необходимых технологий. .NET позволяет создавать компоненты и сервисы, которые могут адаптивным и прозрачным для пользователя образом настраиваться на разнообразные источники контента и обновлений кода. Хорошо спроектированный .NET-сервис «экранирует» подробности своей реализации от потребителя, передавая вопрос о выборе схемы его поставки от пользователей в компетенцию корпоративных ИТ-специалистов и лиц, принимающих бизнес-решения.

Можно выделить следующие сервисы и продукты:

- *Windows.NET* - новое поколение Windows, тесная интеграция с базовыми сервисами .NET, адаптируемость и расширяемость, высокая степень контроля со стороны пользователя;

- *MSN.NET* - контент и сервисы MSN облегчат решение производственных задач, обучение и отдых. Они позволят перейти от эпизодического использования разрозненных ресурсов к «полноценному присутствию в Сети;

- *Personal Subscription Services* -в дополнение к MSN.NET Microsoft планирует поддерживать и развивать самые различные сервисы — от деловых до игровых.

- *Passport.NET* - набор сервисов, упрощающих создание и использование сервисов электронной коммерции, избавляет пользователя от необходимости заводить множество паролей и логинов на разных сайтах, способствует повышению онлайновых безопасности покупок;

- Office.NET -Основанные на Universal Canvas средства создания и публикации документов, а также организации коллективной работы.Office.NET можно будет как купить, так и взять в аренду через Интернет;

- *Visual Studio.NET и Visual Studio for Applications* - универсальная расширяемая среда разработки для Платформы .NET. Быстрая разработка защищенных, масштабируемых, доступных из разных точек Web-приложений и сервисов, интегрирующих ключевые технологии .NET Framework, включая XML, SOAP, ADO.NET, ASP.NET, Windows DNA и другие;

- *bCentral for .NET* – ключевые подписные сервисы для начинающих и быстро растущих компаний, включая Web-хостинг, электронную почту, электронную коммерцию, управление сбытом CRM (Customer Relationship Management) и др.

Важная роль в идеологии .NET отводится комплексу Интернет-услуг, предоставляемых Microsoft в рамках ее грандиозной сети MS Network (MSN). MSN будет развиваться в двух направлениях. Во-первых, будет расширяться спектр услуг MSN. Во-вторых, Microsoft собирается предлагать свои технологии создания таких Интернет-услуг независимым разработчикам. Последнее направление сегодня представлено программной платформой .NET My Services (ранее проект имел кодовое название Hailstorm). .NET My Services открывает возможности централизованного хранения ресурсов (не только информационных, но и программных), доступных с любого удаленного клиентского устройства.

В качестве примера одного из таких ресурсов можно привести тривиальную адресную книгу. Кроме того, эта платформа позволяет создавать собственные варианты программного доступа к этим ресурсам подобных услуг с помощью соответствующего набора SDK.

Главными программными компонентами в комплексе этих средств являются .NET Framework и Visual Studio .NET. Именно поэтому точкой отсчета “эпохи .NET” большинство экспертов считают начало выпуска этих продуктов.

Говоря о перспективах .NET, нужно отметить два момента. С одной стороны, это действительно новая технологическая платформа, существенно отличающаяся от сегодняшней Windows. В упрощенном виде Windows = Win API + + COM, а .NET = CLR + XML Web Services. Понятно, что переход от одной платформы к другой будет происходить постепенно, более того, довольно долго они будут сосуществовать. Недаром архитектура .NET Framework реализована сегодня в виде отдельного дополнительного компонента. С другой стороны, .NET — это очевидный ответ Microsoft идеологии Java.

Общее мнение аналитиков по вопросу будущего данной технологий выразила компания Gartner: NET — это действительно новая платформа, а не очередная модификация Windows. Она очень тесно связана с нынешней архитектурой COM+, и, более того, две эти платформы будут еще длительное время существовать параллельно (хотя бы потому, что многих средств, реализованных с помощью COM+, в рамках .NET пока не существует). Тем не менее .NET — самостоятельная платформа со своими сильными и слабыми сторонами:

- она переносит фокус с отдельных Web-серверов или специализированных информационных систем на создание среды, обеспечивающей их эффективное взаимодействие между собой и с пользователями;

- Microsoft .NET развивает и объединяет концепции как операционных систем, так и Интернета, превращая построенную по открытым стандартам глобальную Сеть в операционную систему нового поколения. .NET позволяет приложениям и сервисам преодолевать ограничения отдельных физических устройств;

- разработчики могут расширять свои решения и инструментальные средства наиболее подходящими многократно используемыми компонентами, импортируемыми из Сети. Они могут сосредоточиться на проектировании элегантной архитектуры решения и не уделять столько внимания утомительному воспроизведению вспомогательных фрагментов кода, необходимого сегодня для простого связывания компонентов.

Контрольные вопросы

1) Что понимается под Платформой .NET?
2) Назовите основное отличие .NET от других современных технологий, в том числе COM+?

3) Перечислите основные компоненты технологии .NET.
4) Назовите главные программные компоненты технологии .NET.
5) Перечислите основные «строительные блоки» рассматриваемой технологии.
6) Какие условия способствовали возникновению .NET?
7) Какие новые возможности предоставляет данная технология?

Раздел 2. Технология ASP.NET

2.1 .NET Framework

Слова .NET и ASP.NET происходят от названия .NET Framework ("общая структура .NET") — набор объектов и планов (Blueprints), созданный компанией Microsoft для разработки приложений. ASP.NET работает благодаря .NET Framework.

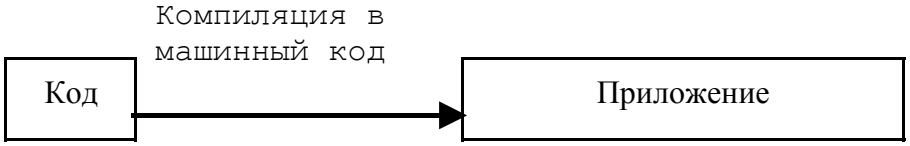
У всех приложений, разработанных в .NET Framework, и в том числе у приложений ASP.NET, есть некоторые общие возможности, обеспечивающие им совместимость, безопасность и стабильность. Рассмотрим эти возможности по одной.

Common Language Runtime (CLR) означает "среда выполнения общего языка", среда, которая управляет выполнением кода. Другими словами, в ней запускается и поддерживается любой написанный код.

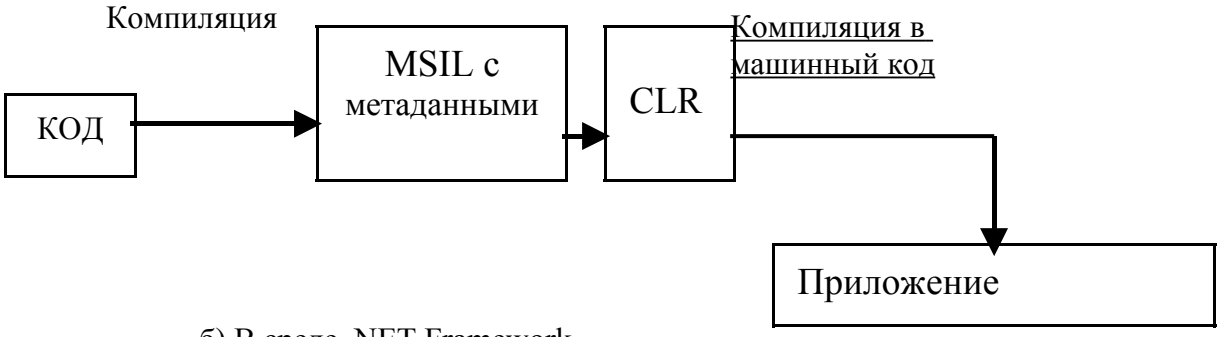
Традиционно при создании приложений пишется код на одном из языков программирования (таком, например, как Visual Basic), компилируете его в формат, понятный для компьютера (в единицы и нули), а затем отправляете откомпилированный код на выполнение. Обратите внимание, что компьютеры разных типов (например, PC и Macintosh) "говорят" на разных языках. Это значит, что для использования приложения на компьютере другого типа приложение необходимо перекомпилировать на язык этого компьютера. Но на .NET Framework дело обстоит несколько по-другому.

Конечно, при использовании .NET Framework и CLR все равно приходится писать код и заниматься его компиляцией. Однако вместо того чтобы компилировать его в то, что понимает компьютер, вы компилируете его в код языка, который называется Microsoft Intermediate Language (MS1L) - промежуточный язык от Microsoft. Такой язык является удобным средством представлять весь написанный вами код. Страницы ASP.NET также компилируются в код MSIL. При компиляции в этот код приложение создает так называемые метаданные. Они являются информацией, которая описывает само приложение. Метаданные рассказывают, что оно может делать, кому принадлежит и так далее.

Вместе с MSIL и метаданными появился новый класс языков программирования: C#, Cobol, Perl и так далее. Эти языки похожи на те, что были до них, но их выводом может быть код, такой как MSIL или обычный компилированный. Затем, когда нужно выполнять программу, за дело берется Common Language Runtime (CLR) и заново компилирует код — на этот раз на родной язык компьютера. Таким образом, код MSIL может выполняться на компьютере любого типа. CLR говорит на языках многих компьютеров и делает за вас всю компиляцию на эти языки. Так что, откомпилировав свое приложение одни раз, можно передавать его на любой другой компьютер. На рисунке 2.1 показано, в чем состоит разница между традиционным процессом компиляции и тем, что выполняется в .NET Framework..



а) Традиционный метод



б) В среде .NET Framework

Рисунок 2.1- Сравнительная схема компиляции

Используя метаданные, CLR ищет, каким образом следует запускать приложение. Благодаря этому установка программ становится очень легкой. Дело в том, что при традиционной установке информацию о приложении необходимо помещать в системный реестр или в центральное хранилище данных о приложениях. К сожалению, системный реестр можно вывести из строя любым изменением в приложении (перемещением его каталога, установкой нового компонента и так далее), и тогда приложение может и не работать так, как положено. Благодаря метаданным системный реестр становится ненужным. Вся необходимая информация хранится вместе с файлами приложения, так что все сделанные изменения вступают в силу автоматически

Код, который работает вместе с CLR, называется *управляемым кодом.* Такое название объясняется тем, что CLR управляет выполнением этого кода и дает определенные преимущества (такие, например, как управление ресурсами), и при этом разработчику не нужно делать ручную настройку. В свою очередь, код, который выполняется вне CLR, называется *неуправляемым.*

Кроме того, CLR может выполнять обработку ошибок, поддерживать безопасность, поддерживать согласованность версий (versioning) и развертывание приложений, а также интеграцию с различными платформами. А это означает, что для написания приложений .NET (и в том числе приложений ASP.NET) можно выбрать любой язык программирования.

2.2 Иерерхия классов

.NET Framework всегда держит при себе планы (Blueprints), то есть описания объектов. В свою очередь, объектами считается все, что находится в .NET Framework: страницы ASP.NET, окна сообщений и так далее. Такие объекты размещаются внутри логических групп, которые называются *пространствами имен* (namespace) (рисунок 2.2). Например, все объекты, относящиеся к базам данных, можно расположить в пространстве имен System.Data (‘Система.Данные’), а все объекты XML — в System.Xml (‘Система.XML’) и так далее. Такое объединение в группы очень полезно при создании библиотек объектов. Пространства имен используются при создании приложений ASP.NET.

System
–System.Data
–System.IO
–System.Net
- System.Net.TcpClient
-System.Net.TcpListener
-System.Net.WebRequest
-System.Net.WebResponse
–System.Reflection
–System.Security

Рисунок 2.2 – Иерерхия классов (отрывок)

Каждое пространство имен из структуры .NET — это, в сущности, коллекция схем. ASP.NET поставляется вместе с собственными схемами, но иногда этого набора бывает недостаточно. Поэтому, чтобы создавать дополнительные типы объектов, приходится обращаться из ASP.NET к другим наборам. Получить доступ к этим

дополнительным объектам и методам можно с помощью ключевого слова Import (импорт):

<%@ Import Namespace="System.Drawing" %>

Эта строка задает импорт всех классов из пространства имен System.Drawing (означает "Система.Чертеж"), например, font (шрифт) или image (изображение). Теперь при создании своих объектов можно использовать эти схемы.

Сборка – набор модулей, предназначенных для совместной работы или самодостаточное и готовое к установке приложение.

Компоненты сборки описываются в манифесте, который содержит: перечисление файлов сборки

–правила использования внешними потребителями типов и ресурсов, определенных в сборке

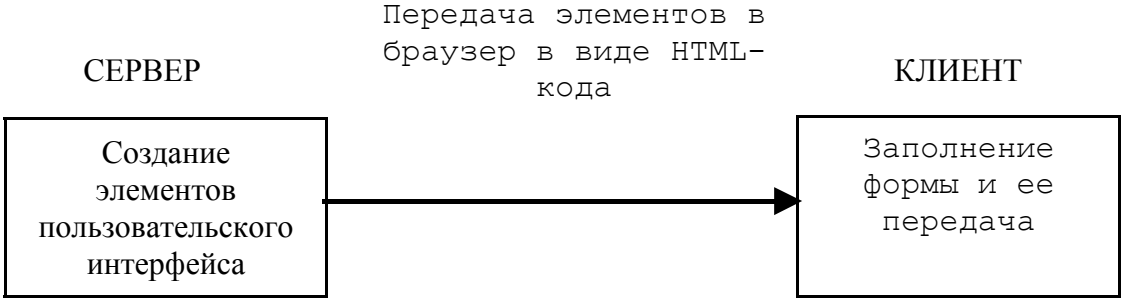
–определение адресации внешних использований типов и ресурсов к их реализации внутри сборки

Каждая сборка имеет уникальное имя: префикс, основанный на открытом ключе разработчика (для общих сборок), простое текстовое имя, номер версии, информация о локализации. Все общие сборки подписываются секретным ключом разработчика

2.3 Web-формы

Формы Web очень похожи на традиционные формы HTML. Разница между ними в том, что первые — это серверное средство, что означает возможность создания пользовательских элементов на сервере. В этом случае сервер полностью осведомлен о том, как выглядит интерфейс, какие функции он выполняет, какие данные следует ожидать и т.п.

На сервере пользователь создает объекты, называемые серверными элементами управления, которые представляют собой части интерфейса пользователя. В отличие от элементов формы HTML, эти объекты полностью контролируемы - они имеют свойства, события и методы, которыми можно манипулировать. Как только клиент запрашивает страницу, ASP.NET преобразовывает эти средства управления в HTML, который совершенно точно отображается в браузере (рисунок 2.3). Используя Web-формы, сервер знает о внешнем виде и функциях каждой конкретной формы



Передача данных обратно
на сервер

Рисунок 2.3

Посредством клиентского сценария, автоматически генерируемого ASP.NET, эти элементы управления предупреждают сервер всякий раз, когда что-либо происходит, например, нажатие на кнопку. Клиентский сценарий отсылает информацию на сервер в момент возникновения события, часто не сообщая об этом пользователю. Таким образом, сервер непрерывно уведомляется о клиентских процессах, что объединяет их друг с другом в одно целое. На практике это выглядит, как управляемая событиями модель. Поскольку сервер сам создает элементы управления, он также может запомнить, что вводилось в каждый элемент.

2.4 Серверные элементы управления

Серверные элементы управления являются составляющими пользовательского интерфейса Web-формы. В ASP.NET выделяется четыре типа серверных элементов управления:

- серверные элементы управления HTML,
- элементы Web,
- средства подтверждения,
- пользовательские элементы.

Элементы управления HTML представляют собой обычные элементы формы HTML, такие как текстовые поля ввода и кнопки, но создаются они на сервере, где ими можно управлять. Аналогичны им элементы Web, но они более функциональны и могут формировать сложный пользовательский интерфейс.

Средства подтверждения используются для проверки правильности пользовательского ввода данных.

Пользовательские элементы специально предназначены для реализации некоторой особой функциональности.

Все элементы управления, размещенные на сервере, имеют свойства, методы и события. Они намного функциональнее, чем традиционные элементы формы HTML, и существенно упрощают процесс формирования пользовательского интерфейса. При создании серверного элемента не нужно заботиться о написании кода HTML. При поступлении запроса на страницу элемент управления автоматически генерирует HTML, корректно отображаемый в браузере. Например, следующая строка создает сервере элемент управления Button:

<asp:Button text="Submit" runat="server"/>

В момент ответа на запрос клиенту выдается такой код HTML:

<input type="Submit" name="ctrll" value="Submit">

Эти две строки лишь похожи друг на друга. Первая выполняется только на сервере. Вторая строка — это то, что принимает клиент.

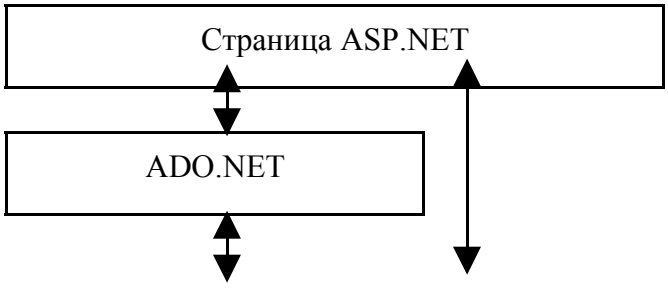
Также ASP.NET знает о возможностях каждого браузера и, следовательно, отсылает соответствующий код каждому из них. Например, если браузер и поддерживает динамический HTML (DHTML), то ASP.NET не будет отправлять ему такой код. Этот подход называется *низкоуровневой поддержкой* связи с тем, что ASP.NET может сглаживать вывод кода HTML для браузеров, которые не поддерживают современный уровень функциональности. В идеале, низкоуровневая поддержка должна отображать корректно i-ый элемент управления. Однако некоторые элементы будут отображаться неодинаково в различных браузерах, что связано с неидентичной трактовкой ими кода HTML. Тем не менее, при разработке под наиболее популярные версии браузеров низкоуровневая поддержка будет работать верно.

2.5 Работа с базами данных. ADO.NET

ADO.NET — новый этап в технологам ActiveX Data Objects (ADO), это модель доступа к данным, созданная специально для использования в Web. ADO.NET создает интерфейс ко всем совместимым с OLEDB источникам данных и обеспечивает соединение с ними, извлечение, управление и обновление данных. ADO.NET можно применять на удаленном компьютере, в распределенном приложении и в случае распределенных данных.

ADO.NET — это версия программного интерфейса ADO, приспособленная для более эффективного представления данных на ASP-страницах. Например, она полностью воспринимает язык XML и приспособлена для общения с XML-приложениями. ADO.NET предлагает много возможностей, которые значительно облегчают работу разработчика.

Страницы ASP.NET используют ADO.NET для связи с любым типом данных. Модель доступа к данным в ADO.NET и ASP.NET представлена на рисунке 2.4. Интерфейс ADO.NET полностью совместим с источниками данных, поддерживающими объектные технологии OLE для баз данных, ими, как язык SQL или механизм баз данных Jet.



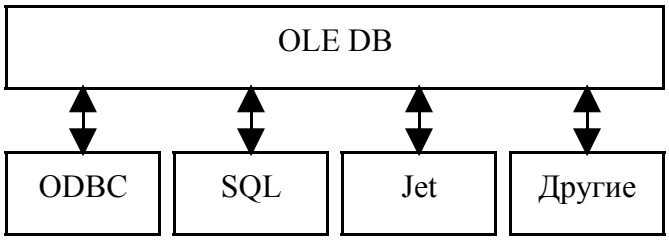


Рисунок 2.4- Модель доступа к данным в ADO.NET и ASP.NET.

ADO.NET состоит из двух ключевых частей: набора данных *DataSet* и средств управления *Managed Providers*. В наборе представлены данные, передаваемые между компонентами ADO.NET, а именно из памяти в страницы ASP.NET. Это механизм обработки данных вне их хранилища. Managed Providers служат в качестве промежуточного звена между набором DataSet и памятью и обеспечивают механизм для соединения, доступа, управления и извлечения информации из любой совместимой с OLE DB базы данных, например Microsoft Access.

Microsoft поставляет два типа Managed Providers в комплекте ADO.NET: SQL и OLE DB. Первый используется исключительно для взаимодействия с Microsoft SQL Server и обеспечивает все методы коммуникации между SQL-сервером и набором данных. Второй, OLE DB, служит связующим звеном между набором и совместимым с OLE DB источником данных.

Для соединения с SQL-сервером одноименный Managed Provider применяет протокол, называемый *tabular data stream* (табличный поток данных). Это наиболее эффективный и не зависящий от OLE DB, ADO или ODBC метод для соединения с SQL-сервером. Данный протокол управляется CLR. OLE DB-провайдер обеспечивает эффективное взаимодействие со всеми другими типами данных и, если есть необходимость, применяется для доступа к серверу SQL. Каждый провайдер имеет три компонента:

- интерфейс для соединения и управления базой данных, а также для взаимодействия с набором DataSet;
- поток данных для быстрого и эффективного доступа (похож на набор данных DataSet, но быстрее и с меньшим набором функций);
- объекты для соединения с базой данных и для исполнения ее специфических команд на нижнем уровне.

В центре интерфейса ADO.NET находится объект DataSet. Этот объект-концептуально новая идея, которая заменяет традиционный объект RecordSet в ADO. DataSet — это простой, постоянно находящийся в оперативной памяти способ хранения данных, который обеспечивает последовательную программную модель для доступа к

данным, независимо от их типа. В отличие от RecordSet, DataSet содержит иные наборы данных, включая ограничения, связи и даже несколько таблиц одноименно.

Данный объект содержит множество структур. В каждую структуру можно поместить несколько элементов. С каждым элементом в структуре можно производить различные операции — добавлять, изменять, просматривать и т.д. В этом заключаются основные принципы работы с DataSet.

DataTable – объект, который представляет отдельную таблицу базы данных. (DataSet содержит коллекцию таблиц в объекте TablesCollection). Объект DataTable полностью отображает соответствующую таблицу, включая связи между данными и ключевые ограничения. Он содержит две другие коллекции, Rows и Columns, которые отображают данные и логическую структуру таблиц.

2.6 Доступ к данным в среде ASP.NET

Можно выделить пять основных этапов получения данных, используя ASP.NET Web-страницы:

- Установить соединение с базой данных.
- Открыть созданное соединение.
- Заполнить компоненту DataSet требуемыми данными.
- Установить компоненту DataView для отображения данных.
- Связать серверный элемент управления с компонентой DataView, используя привязку данных.

Перед тем как получить доступ к данным необходимо импортировать два пространства имен, содержащие компоненты работы с данными (если используемая база данных работает под управлением СУБД Microsoft SQL Server, то используется пространство имен System.Data.SQLClient). Затем необходимо организовать соединение с базой данных. Далее происходит открытие соединения с базой данных и исполняется SQL команда, которая возвращает необходимые данные из таблицы. Это соответствует первому и второму шагам в процессе доступа к данным. Затем создается и заполняется возвращаемыми запросом данными объект типа DataSet. Это соответствует третьему шагу процесса.Потом данные привязываются к элементу управления типа DataList, который автоматически отображает данные.Далее остается просто создать элемент управления типа DataList. Элемент управления типа DataList использует шаблоны для форматирования данных и автоматически обрабатывает полученные записи. Он также получает размер окна браузера и подстраивает ширину колонок так, чтобы все колонки отображались в одном окне

Контрольные вопросы

1) Как осуществляется доступ к данным в среде ASP.NET?
2) Чем отличается процесс компиляции в среде .NET Framework?
3) Что понимается под пространством имен?
4) Назовите основные функции Managed Providers.
5) Основное значение объекта DataSet. Какие структуры содержит объект DataSet?
6) Перечислите основные типы серверных элементов управления в ASP.NET.
7) Как осуществляется доступ к данным в среде ASP.NET?

Раздел 3. Протокол SOAP

3.1 Определение и основные функции

На сегодняшний день существует множество технологий и протоколов, позволяющих объединять элементы распределенных систем между собой. Одна из наиболее известных технологий – DCOM, позволяющая эффективно осуществлять RPC-вызовы, передавать и принимать данные, распределять нагрузку между несколькими back-end серверами. Однако у систем, построенных на DCOM, есть очень важный недостаток, затрудняющий взаимодействие уровня представления и уровня бизнес-логики через Internet. Большинство современных сетевых экранов будут запрещать передачу таких пакетов из соображений безопасности. В распределенных системах для обеспечения взаимодействия разных уровней используется SOAP.

Simple Object Access Protocol (SOAP) – основанный на XML протокол, предназначенный для обмена информацией в распределенных системах. SOAP устанавливает стандарт взаимодействия клиент-сервер и регламентирует, как должен осуществляться вызов, передаваться параметры и возвращаемые значения. Для представления любой информации, передаваемой от клиента к серверу и наоборот, используется XML [20, 21].

SOAP нельзя рассматривать как полную замену DCOM, так как DCOM поддерживает специфические технологии, которые сложно или практически невозможно реализовать для SOAP-вызовов – управление временем жизни объектов (DCOM ping), передача объектных ссылок, COM-события. Поэтому в общем случае для перехода с DCOM на SOAP может потребоваться частичное (или даже полное) изменение архитектуры приложения.

К достоинству протокола SOAP следует отнести то, что он нейтрален к платформе, т.е. не накладывает ограничений на платформы, которые используются клиентом и сервером.

3.2 Элементы вызова SOAP

SOAP описывает преобразование в XML следующих элементов вызова:

- запрос (SOAP Request);
- отклик (SOAP Response);
- сообщение об ошибке (SOAP Fault).

1) SOAP Request. Вызов метода по протоколу SOAP преобразуется в XML следующим образом:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope ...>
 <SOAP-ENV:Body ...>
 <SOAPSDK4:Add ...>
   <x>1</x>
   <y>2</y>
 </SOAPSDK4:Add>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

В приведенном примере запроса вызывается метод Add, которому передаются два параметра: 1 и 2. Информация о том, какой метод и у какого объекта необходимо вызвать, передается в заголовке. В случае протокола HTTP заголовок может выглядеть следующим образом:

```
<HTTPHeaders>
<soapaction>"http://tempuri.org/Sample1/action/Adder.Add"</soapaction>
 <content-type>text/xml; charset="UTF-8"</content-type>
 <user-agent>SOAP Toolkit 3.0</user-agent>
 <host>aida:8080</host>
 <content-length>516</content-length>
 <connection>Keep-Alive</connection>
 <cache-control>no-cache</cache-control>
 <pragma>no-cache</pragma>
</HTTPHeaders>
```

В теге SoapAction указывается, какое действие необходимо выполнить на сервере.

2) SOAP Response. Ответ сервера содержит значения возвращаемых параметров.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
```

```
<SOAP-ENV:Envelope ...>
 <SOAP-ENV:Body ...>
 <SOAPSDK4:AddResponse ...>
   <Result>3</Result>
 </SOAPSDK4:AddResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

В данном случае сервер ответил на вызов метода Add с параметрами 1 и 2 значением 3.

3) SOAP Fault. В случае возникновения ошибок до или во время обработки запроса сервер возвращает информацию об ошибке. Ошибки могут быть двух типов:

- клиентские ошибки: неправильный запрос к серверу; запрос содержит неверные значения параметров; неправильно сформирован и т.д.;

- серверные ошибки. Ошибки могут быть связаны как с функционированием самого сервера (закончились свободные ресурсы), так и с обработкой запроса (компонент вернул ошибку).

Серверный компонент может использовать несколько стратегий для того, чтобы сообщить клиенту об ошибках, возникающих во время обработки запроса. Если метод возвращает код ошибки, SOAPServer предпримет попытку получить от компонента расширенную информацию об ошибке:

- ISoapError – если компонент поддерживает этот интерфейс, SOAPServer установит значение элементов отклика SOAPFault в соответствии с информацией, полученной вызовом методов интерфейса ISoapError.

- IErrorInfo – если компонент поддерживает стандартный интерфейс для предоставления информации об ошибке в COM, SOAPServer установит некоторые значения отклика SOAPFault в соответствии с информацией в IErrorInfo.

Пример серверной ошибки:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope ...>
 <SOAP-ENV:Body ...>
 <SOAP-ENV:Fault>
   <faultcode>SOAP-ENV:Server</faultcode>
   <faultstring>Can add no more numbers</faultstring>
   <faultactor>http://aida:8080/Sample1/Sample1.ASP</faultactor>
   <detail>
   <mserror:errorInfo ...>
```

```
    <mserror:returnCode>-2147467259 : Unspecified error
    </mserror:returnCode>
    <mserror:serverErrorInfo>
    <mserror:description>Can add no more numbers</mserror:description>
    <mserror:source>Sample1.Adder.1</mserror:source>
    </mserror:serverErrorInfo>
    <mserror:callStack>
    <mserror:callElement>
      <mserror:component>WSDLOperation</mserror:component>
      <mserror:description>Executing method Add failed
      </mserror:description>
      <mserror:returnCode>-2147352567 : Exception occurred.
      </mserror:returnCode>
    </mserror:callElement>
    ...
    </mserror:callStack>
   </mserror:errorInfo>
   </detail>
  </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

В теге *faultcode* указывается источник ошибки – клиент или сервер. *Faultstring* содержит строковое описание ошибки. *Faultfactor* – указывает URL, к которому обращался клиент. Тег *detail* содержит дополнительную информацию об ошибке (например, call stack). Спецификация SOAP не накладывает на нее каких-бы то ни было ограничений. SOAP Toolkit помещает в тег detail элемент <mserror:errorInfo ...>, содержащий следующие дочерние элементы:

- returnCode содержит код ошибки, которую вернул компонент;

- serverErrorInfo появится в случае, если серверный компонент поддерживает IErrorInfo, и будет содержать значения всех составляющих IErrorInfo – описание ошибки, источник, helpfile и helpcontext;

- callStack – коллекция элементов callElement – трассировка вызова по внутренним компонентам SOAPServer. Каждый callElement содержит имя компонента, код возврата, описание ошибки. С помощью этой информации можно увидеть, на каком из входящих

в SOAPServer компонентов обработка вызова завершилась с ошибкой, и как выглядела цепочка вызываемых компонентов.

Любой вызов SOAP включает в себя следующие этапы:

- разбор WSDL-файла и определение операции и параметров, которые необходимо передать на сервер;
- преобразование значений параметров в текстовую форму;
- формирование XML-запроса;
- передача запроса на сервер;
- получение отклика сервера;
- преобразование выходных параметров, разбор сообщения об ошибке.

За каждый из этапов отвечает специальный компонент SOAP Toolkit 3.0. За разбор WSDL-файла и предоставление информации о количестве и типах параметров отвечает компонент WSDLReader30, результатом работы которого являются WSDLOperation, WSDLPort, WSDLService. С помощью этих компонентов можно получить всю информацию, содержащуюся в WSDL-файле.

Преобразование параметров в текстовую форму осуществляется с помощью так называемых mapper-ов. В состав SOAP Toolkit mapper-ы для всех oleautomation-типов и некоторых других. Кроме того, есть Generic Custom Type Mapper, который подходит для большинства составных типов данных (структур). Для тех случаев, когда стандартные mapper-ы не подходят, можно использовать свои собственные.

Передача запроса на сервер и получение отклика осуществляется с помощью так называемого коннектора.

3.3 SOAP Toolkit

Для разработки распределенных приложений, использующих протокол SOAP, существует большое количество наборов компонентов, облегчающих процесс разработки и берущих на себя обработку различных этапов вызова и получения отклика от сервера, например, SOAP Toolkit.

Microsoft SOAP Toolkit, который в значительной степени облегчает процесс создания серверных компонентов SOAP, обеспечивающих прием и обработку запросов, и предоставляет стандартный Proxy, реализующий динамический IDispatch –интерфейс, что позволяет легко преобразовывать вызовы COM в SOAP-запросы.

SOAP Toolkit включает в себя следующие части:

- SOAP Listener – серверные компоненты, предназначенные для обработки входящих запросов. Представлены в 2-х разновидностях – ASP-страница и ISAPI-расширение;

- Proxy – клиентский компонент, позволяющий преобразовать вызов метода, осуществляемый через IDispatch-интерфейс, в SOAP-вызов;

- утилита для трассировки SOAP-вызовов, позволяет перехватывать вызовы к серверу и анализировать запросы и ответы сервера;

- WSDL-генератор. По библиотеке типов компонента генерирует файлы, необходимые для работы SOAP Listener'а и Proxy. В этих файлах находятся описания методов, параметров, типов параметров и т.д.;

- набор низкоуровневых компонентов, предназначенных для создания SOAP-запросов, преобразования различных типов данных в текстовый формат, приема и передачи пакетов по протоколу HTTP и преобразования ответа сервера;

- документация и примеры.

3.4 WSDL- файлы

WEB Service Description Language (WSDL) основан на XML и предназначен для описания тех сервисов, которые поддерживает сервер. Для каждого сервиса в WSDL-файле перечисляется набор операций, поддерживаемых данным сервисом, а также определяется формат сообщения, которого должен придерживаться клиент, чтобы сформировать правильный запрос.

WSDL-файл задает контракт между сервером и клиентом, который клиент обязан выполнять, чтобы быть обслуженным.

WSDL-файла включает следующие основные разделы:

1) Схема – <schema> – описывает сложные типы данных, используемые в параметрах методов.

```
<types>
 <schema ...>
 ...
 <complexType  name ='Recordset'>
  ...
 </complexType>
 </schema>
</types>
```

2) Описание SOAP-сообщений – <message>

```
<message name='TView.GetModules'>
 <part name='processID' type='xsd:int'/>
```

```
</message>

<message name='TView.GetModulesResponse'>
  <part name='Result' type='typens:Recordset'/>
</message>
```

3) Описание SOAP-портов и операций, которые поддерживаются этими портами:

```
<portType name='TViewSoapPort'>
  ...
  <operation name='GetModules' parameterOrder='processID'>
    <input message='wsdlns:TView.GetModules'/>
    <output message='wsdlns:TView.GetModulesResponse'/>
  </operation>
</portType>
```

4) Описание формата операций:

```
<binding name='TViewSoapBinding' type='wsdlns:TViewSoapPort' >
  ...
  <operation name='ShutdownMachine'>
    <soap:operation .../>
      <input>
        <soap:body ...parts='nFlags'/>
      </input>
      <output>
        <soap:body .../>
      </output>
    </sopa:operation>
  </operation>
</binding>
```

5) Описание сервисов и портов, входящих в эти сервисы:

```
<service name='TView' >
  <port name='TViewSoapPort' binding='wsdlns:TViewSoapBinding' >
```

```
  <soap:address location='http://aida/TView/TView.ASP'/>
 </port>
</service>
```

В WSDL-файле могут быть описаны один или несколько сервисов, для каждого из которых задан свой адрес – URL. WSDL-файл может находиться локально по отношению к клиенту или загружаться по протоколу HTTP (как в данном случае). Если WSDL-файл загружается по HTTP, и WEB-сервер требует аутентификации, имя пользователя и пароль необходимо передать через URL, например, http://user:pwd@aida/Sample1/Sample01.wsdl/ .Каждый сервис включает в себя один или несколько портов. Генератор WSDL-файлов создает один порт для каждого из указанных интерфейсов COM-объекта. Каждый порт включает в себя одну или несколько операций. Если сравнивать порт с интерфейсом, то операцию можно сравнить с методом интерфейса. Каждая операция включает несколько частей. Часть (part) можно сравнить с параметром метода.

3.5 WSML-файлы

WSML-файл описывает связь операций с конкретными методами COM-объектов, которые будут обслуживать запросы. Ниже приведены основные разделы WSML-файла.

1) Ссылки на используемые COM объекты:

```
<service name='TView'>
 <using PROGID='TView.TView.1' cachable='0' ID='TViewObject' />
```

2) Связь сложных типов данных с мэпперами, которые будут отвечать за преобразование сложных типов в XML:

```
<types>
        <type name='Recordset' ... targetPROGID='ADODB.Recordset'
iid='{00000535-0000-0010-8000-00aa006d2ea4}'/>
</types>
```

3) Описание связи между SOAP-операцией и методами COM-объекта:

```
<port name='TViewSoapPort'>
  <operation name='ShutdownMachine'>
   <execute uses='TViewObject' method='ShutdownMachine' dispID='1'>
```

```
    <parameter callIndex='1' name='nFlags' elementName='nFlags' />
   </execute>
  </operation>
</port>
```

На основе информации, содержащейся в этих файлах, компоненты SOAP Toolkit осуществляют подготовку SOAP-запросов, преобразование и передачу параметров, анализ ответа сервера.

Контрольные вопросы

1) Перечислите основные функции протокола SOAP.
2) Назовите основные достоинства протокола SOAP.
3) Каким образом описывается связь операций с конкретными методами COM-объектов?
4) В каком файле содержатся описания сервисов, поддерживаемых сервером?
5) Перечислите элементы вызова SOAP.
6) Назовите основные этапы вызова SOAP.
7) Какие основные разделы включает WSDL-файл?
8) С помощью каких компонентов можно получить всю информацию, содержащуюся в WSDL-файле?
9) Назначение SOAP Toolkit.
10) Поясните термин «мэппер».
11) Как осуществляется связь сложных типов данных с мэпперами?

Раздел 4. Язык программирования C#

4.1 Определение языка

С появлением технологии .NET. программирование выходит на новый уровень развития. Специально для данной технологии компания Microsoft разработала объектно-ориентированный язык программирования C# (Си Шарп), который является новейшим на сегодняшний день языком программирования. Язык C# был создан специально для технологии ASP.NET [19].

Он сочетает в себе преимущества уже существующих языков программирования и дополняет их удобными механизмами работы с технологией .NET.

Данный язык был разработан в 2000 году Андерсом Хейлсбергом, Скоттом Вилтамутом и Питером Гольде под эгидой Microsoft Research. Основным постулатом C# является высказывание: "всякая сущность есть объект". Язык основан на строгой

компонентной архитектуре и реализует передовые механизмы обеспечения безопасности кода.

C# - это полнофункциональный объектно-ориентированный язык, который поддерживает все три «столпа» объектно-ориентированного программирования: инкапсуляцию, наследование и полиморфизм. Он имеет прекрасную поддержку компонентов, надежен и устойчив благодаря использованию «сборки мусора», обработки исключений, безопасности типов.

Язык C# разрабатывался "с нуля" и вобрал в себя много полезных свойств таких языков, как C++, Java, Visual Basic, а также Pascal, Delphy и др. При этом необходимость обратной совместимости с предыдущими версиями отсутствовала, что позволило языку C# избежать многих отрицательных сторон своих предшественников.

Как и Java, язык C# разрабатывался для программирования приложений Интернет, и примерно 75% его синтаксических возможностей аналогичны языку программирования Java, 10% подобны языку программирования C++, а 5% – заимствованы из языка программирования Visual Basic. Объем новых концептуальных идей в языке C# составляет около 10%. Выделение и объединение лучших идей современных языков программирования делает язык C# не просто суммой их достоинств, а языком программирования нового поколения.

Существует несколько версий языка C#. Первая версия языка C# стандартизирована в ECMA и ISO .Вторая версия языка C# 2.0 вышла 7 ноября 2005 вместе с Visual Studio 2005 и .NET 2.0. Третья версия языка C# была представлена в сентябре 2005 в виде проекта спецификации C# 3.0 и бета-версии C# 3.0, устанавливаемой в виде дополнения к Visual Studio 2005.

Код на C# представляет собой последовательность операторов, каждый из которых заканчивается символом точки с запятой (;). Каждый оператор, как правило, является частью некоего блока, который обрамляется парой фигурных скобок ({ и }). Блоки могут быть вложенными друг в друга. После закрывающей блок фигурной скобки точка с запятой не ставится. Рекомендуется располагать каждый оператор в отдельной строке и соблюдать отступы в блоках. Язык C# чувствителен к регистру, то есть идентификаторы Console и console –это разные идентификаторы.

Для создания программы, как правило, используют среду разработки Microsoft Visual Studio 2005, хотя можно вводить тексты программ в любом текстовом редакторе и использовать компилятор командной строки C# - csc.

Синтаксис языка, структуру программы и другиме ключевые особенности

программ, написанных на C# рассмотрим на примере консольной программы, выводящей на экран консоли некоторый текст.

4.2 Синтаксис языка

Программа выводит на экран консоли список имен корневых узлов реестра Windows. Напомним, что Реестр (registry) - это база данных Windows, которая хранит самую разнообразную информацию о системе: какие устройства и программы в ней установлены, настройки установленных программ, список пользователей и их права, и многое другое. Реестр имеет иерархическую структуру и содержит несколько корневых узлов с конкретными именами. Например, узел HKEY_LOCAL_MACHINE содержит информацию об аппаратных устройствах компьютера, о системной памяти, драйверах и т. д. Узел HKEY_CURRENT_USER содержит настройки пользователя, который в текущий момент работает на компьютере: установки рабочего стола, переменные окружения, настройки приложений и т. п. Корневые узлы имеют подузлы, содержащие список параметров, которым можно задать определенные значения.

Библиотека классов .NET Framework предоставляет несколько объектов для работы с реестром. Например, класс Registry возвращает один из корневых узлов реестра в виде экземпляра класса RegistryKey, с помощью которого вы можете выполнять операции чтения из реестра или записи в него. Библиотека .NET Framework также содержит перечисление RegistryHive. Перечисление - это тип данных, позволяющий создать группу связанных между собой числовых констант. При этом константам задаются символьные имена.

Например, можно создать перечисление Seasons, которое будет содержать имена времен года:

enum Seasons { Summer, Autumn, Winter, Spring };

Рассмотрим код программы на языке C#.

```
/*
 * Программа выводит список имен корневых узлов реестра, которые заданы
 * в перечислении RegistryHive
  */
using System;
using System.Collections.Generic;
using System.Text;
using Microsoft.Win32;
namespace MyRootKeys
{
```

```
class Program
{
  static void Main(string[] args)
  {
    Type hives = typeof(RegistryHive);
    foreach (string s in Enum.GetNames(hives))
      Console.WriteLine(s);
  }
 }
}
```

Первые четыре строчки программы - это комментарий. Комментарии в C# бывают трех типов: многострочные, однострочные и XML-комментарии. В приведенном программе используется многострочный комментарий, который начинается с пары символов /* и заканчивается парой символов */. Все, что располагается между ними, считается комментарием.

Однострочный комментарий обозначается двумя символами слеша (//) и может располагаться в любом месте строки. Концом комментария считается конец строки.

XML-комментарий обозначается тремя символами слэша (///) и также заканчивается концом строки. Он предназначен для автоматического создания документации к вашей программе в среде разработки Visual Studio.

После комментариев следуют четыре предложения using:

```
using System;
using System.Collections.Generic;
using System.Text;
using Microsoft.Win32;
```

Эти предложения определяют, какие пространства имен будут использоваться в данной программе. Пространства имен позволяют обеспечить уникальность идентификаторов в программе и уменьшают конфликты между этими идентификаторами. Вся библиотека классов, поставляемая с Microsoft .NET Framework, поделена на ряд пространств имен.

Первое предложение - using System; - говорит о том, что мы будем использовать в тексте программы некие классы (или другие типы) из пространства имен System библиотеки классов .NET Framework. В нашей программе это будут классы Console и Type, определенные в пространстве имен System. Если бы не указали предложение using System;, то пришлось бы указывать полностью квалифицированное имя этого

класса System.Console.WriteLine(s). Пространство имен может состоять из нескольких идентификаторов, разделенных между собой точкой.

Два следующих пространства имен, указанные предложениями using и вставленные автоматически мастером генерации нового проекта, в данной программе не нужны, и можно удалить эти строки.

Четвертое предложение - using Microsoft.Win32 - указано потому, что в программе используется тип RegistryHive из этого пространства.

Следующим предложением - namespace - определяем для программы собственное пространство имен с именем MyRootKeys. namespace MyRootKeys

{ ...

}

Это имя совпадает с именем проекта, которое было указано при его создании. В свойствах проекта можно изменить это имя на любое. Определение пространства имен окружается фигурными скобками.

Далее идет определение класса Program:

class Program

{

...

}

Класс - это основная организационная структура объектно-ориентированных языков типа C#. Все операторы должны находиться внутри определения какого-либо класса. Невозможно написать какой-либо оператор вне класса. Определение класса заключается в фигурные скобки.

Класс Program содержит определение функции Main.

static void Main(string[] args)

{

}

Функция Main является точкой входа в программу, с которой начинается ее выполнение. Ключевое слово void указывает, что функция ничего не возвращает. Функция Main принимает параметр в виде массива строк с именем args. Данный параметр имеет смысл, когда пользователь запускает приложение и передаете ему в командной строке какие-либо параметры, которые затем передаются приложению в массиве args.

Функция Main имеет ряд вариантов (говорят, что функция перегружена):

static void Main(string[] args)

static int Main(string[] args)

static void Main()

static int Main()

Как видно, Main может принимать параметр в виде массива строк или не принимать ничего. Также функция Main может возвращать системе при своем завершении целое число (код завершения программы) или ничего не возвращать (void). Поскольку в данной программе не используются параметры командной строки, можно удалить args из определения Main.

Следующая строка –

Type hives = typeof(RegistryHive);

определяет переменную с именем hives, имеющую тип Type. Этой переменной присваивается значение выражения typeof(RegistryHive). Данное выражение возвращает информацию о типе RegistryHive в виде экземпляра класса Type. Тип Type - это один из базовых классов библиотеки классов .NET Framework, он интенсивно используется в реализации технологии отражения

RegistryHive - это перечисление, содержащее значения корневых узлов реестра Windows. Перечисление RegistryHive определено в пространстве имен Microsoft.Win32. Если не указать в начале файла предложение using Microsoft.Win32, то выражение typeof должно было бы выглядеть следующим образом:

typeof(Microsoft.Win32.RegistryHive).

В последних двух строчках программы используется цикл foreach:

foreach (string s in Enum.GetNames(hives))

Console.WriteLine(s);

Цикл foreach выполняет указанные пользователем действия над каждым элементом массива или коллекции. В данном случае с помощью функции Enum.GetNames(hives) получаем имена членов перечисления RegistryHive (имена корневых узлов реестра системы) в виде массива строк. Далее цикл foreach берет каждый член этого массива, присваивает его значение строковой переменной s и выводит эту строку на экран консоли. Вывод строк на экран осуществляется методом WriteLine класса Console, определенного в пространстве имен System. Метод WriteLine имеет около двух десятков перегруженных вариантов, каждый из которых принимает различное количество параметров различных типов. В рассматриваемом случае просто передаем этой функции имя члена перечисления RegistryHive в виде переменной s, которое и выводится на экран консоли.

Ниже приведены особенности, уникальные для C#:

а) Определения в namespace. При создании программы создается один или более классов в namespace. В нем же (вне класса) возможно объявление интерфейсов, enums и structs. Используя ключевые слова можно адресовать содержимое другого namespace.

б) Фундаментальные типы данных. В C# существует более широкое разнообразие типов данных чем в C, C ++ или Java. Типы - bool, byte, ubyte, short, ushort, int, uint, long, ulong, float, double, and decimal. Подобно Java, все типы имеют установленный размер. Подобно C и C ++ все типы могут быть знаковыми и без знаковыми. Подобно Java, char содержит 16-ти битный unicode символ. В C# новым типом данных является тип decimal, который может содержать до 28 десятичных цифр.

в) Два фундаментальных класса - класс object - базовый класс всех классов. Класс string - также базовый класс.

г) Ассемблирование - коллекция компилируемых классов и способность к выполнению других элементов языка, которые объединены в отдельном файле. Если это программа, файл имеет расширение EXE. Если это библиотека - DLL.

д) Признаки. Каждый член класса имеет признаки: public - доступен для всего кода; protected - доступен только от полученных классов; internal -доступен только при ассемблировании; protected internal - доступен только от полученных классов в пределах ассемблирования; private - доступен только из класса.

е) Прохождение аргумента. Методы могут объявляться для принятия некоторого числа аргументов. По умолчанию происходит передача значений фундаментальным типам данных. Ключевое слово ref может использоваться для передачи значения по определенной ссылке, которая позволяет возвращать значение. Ключевое слово out также вызывает переход по ссылке без передачи значения.

ж) Виртуальные методы. Прежде, чем метод в базовом классе будет переписан, он должен быть объявлен как virtual. Метод в подклассе, который будет переписан, должен быть объявлен с помощью ключевого слова override. Это предотвратит случайную перезапись метода. Данная особенность улучшает удобочитаемость и непринужденность обслуживания C# модулей.

з) Свойства. COM объект имеет свойства и поэтому каждый C# класс может использоваться как COM объект. C# позволяет определять свойства внутри любого класса. Внутри C# класса, каждому свойству дается имя и тип данных. Ключевые слова set accessor и get accessor используется для объявления выполняемого кода при чтении или обновлении свойства.

и) Индексатор. Индексатор подобен свойству за исключением того, что вместо имени для адресации члена класса используется индексированное значение внутри

квадратных скобок (как массив).

к) Определение версий. C# позволяет разработчикам поддерживать множество версий классов в двоичной форме, помещая их в различных namespace. Это позволяет как старым, так и новым версиям программного обеспечения запускаться одновременно. Наряду с этим в C# будет способность поддерживать и управлять множеством версий исходного кода.

л) Проверенная и непроверенная оценка. Проверенное выражение - выражение, которое выдает исключение при выходе за его пределы. Непроверенное выражение - выражение, которое не выдает исключение. Ключевые слова checked и unchecked используются для явного определения того, каким образом была выполнена оценка:

```
int j = checked(a * b);
int k = unchecked(a * b);
```

м) Явные и неявные преобразования. Подобно Java, C# учитывает неявное преобразование фундаментальных типов данных, пока нет вероятности потери данных (преобразование типа byte в int), но если есть вероятность потери данных (преобразование int в byte) выполняется явное преобразование. C# расширяет эту способность для других элементов программы, позволяя программисту определить как явные, так и неявные преобразования.

н) Внешне выполняемые методы. Методы в классе могут выполняться внешне. В следующем примере [19], статический метод RemoveDirectory выполняется в библиотеке под именем kernel32.dll:

```
class Path {
[DllImport("kernel32", setLastError=true)]
static extern bool RemoveDirectory(string name);
}
```

п) Итерация через члены коллекции.Утверждение foreach может использоваться для однократного выполнения блока кода для каждого члена списка или массива.

Контрольные вопросы

1) Перечислите особенности языка C#.
2) Какие среды можео использовать для создания программы на C#?
3) Как учитываются явные и неявные преобразования?
4) Поясните синтаксис определения пространства имен, которые будут использоваться в данной программе.
5) Назовите основные отличия языка C# от C++ и Java.

Список используемой литературы

1. Rosenberg J., Schulzrinne H., Camarillo G., Johnston A., Peterson J., Sparks R., Handley M., E. Schooler SIP: Session Initiation Protocol, RFC 3261
2. M.Handley, H. Schulzrinne, E. Schooler, J. Rosenberg SIP: Session Initiation Protocol, RFC 2543
3. J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart HTTP Authentication: Basic and Digest Access Authentication, RFC 2617
4. Donovan S. The SIP INFO Method, RFC 2976
5. J. Rosenberg, H. Schulzrinne Reliability of Provisional Responses in SIP, RFC 3262
6. A. B. Roach SIP - Specific Event Notification, RFC 3265
7. J. Rosenberg SIP UPDATE Method, RFC 3311
8. R. Sparks SIP Refer Method, RFC 3515
9. A. Johnston, A. Johnston, R. Sparks, C. Cunningham, K. Summers SIP Basic Call Flow Examples, RFC 3665
10. J. Rosenberg, H. Schulzrinne SIP: Locating SIP Servers, RFC 3263
11. J. Peterson Privacy Mechanism for SIP, RFC 3323
12. H. Schulzrinne, D. Oran, G. Camarillo The Reason Header Field for SIP, RFC 3326
13. G. Camarillo, W. Marshall, J. Rosenberg Integration of Resource Management and SIP, RFC 3312
14. C. Jennings, J. Peterson, M. Watson Private Extensions to SIP for Asserted Identity within Trusted Networks, RFC 3325
15. B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, D. Gurle SIP Extension for Instant Messaging, RFC 3428
16. D. Willis, B. Hoeneisen SIP Extension Header Field for Service Route Discovery During Registration, RFC 3608
17. Гольдштейн Б.С. -Протокол SIP. Справочник Серия: Телекоммуник. протоколы ВСС РФ.- Питер: BHV, 2005.
18. Гольдштейн Б.С. Пинчук А.В. Суховицкий А.Л. IP – телефония. М.:-Радио и связь, 2001.
19. Дейтел Х. C# . – Питер, BHV, 2006.
20. Тематический комплект Электронная библиотека по информационным технологиям.
21. http://ru.wikipedia.org/wiki/VoIP.

ПРИЛОЖЕНИЕ

Итоговый тест

1) Укажите неверное высказывание:

а) При организации телефонных переговоров по ВС необходимо передавать два типа информации: командную и речевую.

б) Основное требование к передаче командной информации - отсутствие ошибок передачи.

в) При передаче речевой информации проблема времени доставки пакетов по сети становится основной.

г) При передаче речевых пакетов используется достоверный транспортный протокол TCP.

д) Перед поступлением речевого потока на декодер, он восстанавливается с использованием протокола реального времени.

2) Укажите номер строки, содержащей неверное утверждение:

а) Спецификация класса FEC может содержать несколько компонентов, каждый из которых определяет набор пакетов, соответствующих данному классу.

б) Пакет принадлежит данному классу, если адрес получателя точно совпадает с компонентом адреса узла либо имеет максимальное совпадение с адресным префиксом.

в) Компонентами EFC являются: адрес узла и адресный префикс

г) MPLS базируется на IP

д) Архитектура MPLS требует обязательного применения LDP.

3) В соответствии с ITU-T H.323 сеть IP-телефонии представляет собой набор следующих устройств, соединенных по IP-сети:

а) gateway, gatekeeper, administration manager

б) gateway, gatekeeper, router

в) gateway, gatekeeper, administration manager, VoiIP

г) gateway, administration manager, TCP/IP

д) ПК, gateway, микрофон, наушники (колонки)

е) gatekeeper, administration manager, router

ж) нет правильного ответа.

4) Укажите неверное высказывание.

а) Шлюз- основная и неотъемлемая часть архитектуры IP-телефонии, непосредственно соединяющая телефонную сеть с IP-сетью.

б) Шлюзы различных производителей отличаются способом подключения к телефонной сети, аппаратной платформой и выполняемыми функциями.

в) Шлюз выполняет следующие функции: ответ на вызов абонента PBX/PSTN; установление соединения с удаленным шлюзом; установление соединения с вызываемым абонентом PBX/PSTN; сжатие, пакетирование и восстановление голоса (или факс-сигнала).

г) Диспетчер (Gatekeeper)- дополнительной устройство, подключенное только к IP-сети и несущее в себе всю логику работы сети IP-телефонии.

д) Функциями Gatekeeper является: аутентификация и авторизация абонента, распределение вызовов между шлюзом, биллинг (интерфейс с профессиональными биллинговыми системами).

5) SOAP Toolkit включает в себя следующие части:

А) серверные компоненты, предназначенные для обработки входящих запросов
Б) Утилита для трассировки SOAP-вызовов
В) WSDL-генератор
Г) Набор низкоуровневых компонентов, предназначенных для создания SOAP-запросов
Д) Proxy – клиентский компонент
Е)ASP-страница и ISAPI-расширение
Ж) WSDL- и WSML-файлы

1) А Б В Г Д Ж
2) Б В Г Е
3) А Б В Г Д Е
4) А Б Д Е Ж
5) А Б В Г Е Ж
6) В Г Е Ж

6) Укажите последовательность заполнения меток по протоколу LDP, при наличии 3-х LSR в сети MPLS:

1 LDP уведомляет верхний маршрутизатор (N-1) о том, что потоку N присвоена метка M.	**5** LSR_{N-1} присваивает собственное значение метки данному потоку.
2 LSR устанавливают сеансы LDP с соседними устройствами	**6** Каждое устройство сети MPLS строит базу топологической информации, используя OSPF
3 LSR_{N-1} уведомляет верхний маршрутизатор об этой привязке.	**7** Верхний маршрутизатор метку M помещает в поле выходной метки своей таблицы.
4 Выходной граничный маршрутизатор ассоциирует класс FEC с пакетами и присваивает случайное значение метки M.	**8** Верхний маршрутизатор помещает полученную информацию в поле своей таблицы.

а) 2-3-5-7 –8-6-1
б) 1-6-8-7-5-3-2
в) 8-1-5-3-6-4
г) 6-2-4-1-7-5-3-7
д) 4-1-2-7-5-6-3-7
е) 8-3-2-5-7 –1-6

7) Что является универсальными «строительными» блоками технологии .NET?

а) PERSONALIZATION, NOTIFICATION AND MESSAGING, XML STORE, NATYRALE INTERFACE
б) XML STORE, SMARTTAGS, IDENTIFY,
в) IDENTIFY, PERSONALIZATION, NOTIFICATION AND MESSAGING, XML STORE, UNIVERSAL CANVAS
г) UNIVERSAL CANVAS, NOTIFICATION AND MESSAGING, XML STORE,
д) IDENTIFY, PERSONALIZATION, NOTIFICATION AND MESSAGING, XML STORE
е) MSN.NET, PERSONALIZATION, , XML STORE, NATYRALE INTERFACE

8) Укажите номер правильного ответа для нижеприведенных утверждений.

1) C# обладает всеми основными свойствами языка для современных серверных приложений
2) C# объектно-ориентированный язык ,без множественного наследования.
3) C# объектно-ориентированный язык, с развитыми средствами безопасности.
4) C# объектно-ориентированный язык, со ссылками, но без указателей
5) C# объектно-ориентированный язык, с объектной обработкой исключений, с множественным наследованием

а) утверждения 1,3,5 - верные, 2,4 - ложные
б) утверждения 1,2, 4 - верные, 3,5 - ложные
в) утверждения 1,2,3 - верные, 4,5 - ложные
г) утверждения 3,4,5- верные, 1,2 - ложные
д) утверждения 2,3,4 - верные, 1,5 - ложные
е) утверждения 1-4 - верные, 5 - ложное

9) Основные разделы WSDL-файла:

а) <schema> <message> <portType><service name='TView'> <types> <binding> <service>

б) <schema> <message> <service name='TView'> <types> <port name='TViewSoapPort'>

в) <service name='TView'> <types> <portType><binding>

г) <schema> <message> <portType> <binding> <service>

д) <portType> <service name='TView'> <types> <port name='TViewSoapPort'> <message>

10) Укажите все неверные ответы (для сети H.323):

а) для установления соединения в сети используется только транспортный протокол UDP
б) для установления логических каналов используется динамический TCP-порт
в) для обмена данными о функциональных возможностях используется динамический TCP-порт

г) для определения ведущего и ведомого оборудования используется динамический TCP-порт
д) для установления начального соединения между абонентами используется TCP-порт 1720
е) для передачи активной речи используется протокол RTP
ж) для передачи активной речи используется протокол RSVP

1) а, б, в
2) а, д, ж
3) г, ж
4) б, в, е
5) а,ж
6) все ответы правильные
7) все ответы неправильные

11) Укажите правильную последовательность этапов получения данных в ASP.NET:

А) открыть созданное соединение
Б) заполнить компоненту DataSet требуемыми данными
В) установить соединение с БД
Г) установить компоненту DataView для отображения данных
Е) Используя привязку данных, связать серверный элемент управления с компонентой Data View
Ж) Импортирование пространства имен, содержащие компоненты работы с данными

1) А Б В Г Д Ж
2) В Б А Г Е
3) А Б В Г Д Е
4) А В Б Д Е Ж
5) Ж В А Б Г Е
6) А В Г Е Ж

12) Приведенный ниже пример содержит:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope ...>
<SOAP-ENV:Body ...>
<SOAPSDK4:Add ...>
<x>1</x>
<y>2</y>
</SOAPSDK4:Add>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

1) Вызов метода Add с параметрами 1 2 по протоколу SOAP
2) Ответ сервера на вызов метода Add значениями 1 2 по протоколу SOAP
3) Указания, какие действия необходимо выполнить на сервере, используя методы со значением 1,2 SOAP
4) Описание сложных типов данных, используемых в параметрах методов по протоколу SOAP

13) Какая строка не содержит переменных CGI-окружения:

1) REQUEST_METHOD, QUERY_STRING
2) CONTENT_TYPE, REMOTE_HOST
3) SCRIPT_NAME, SCRIPT_FILENAME
4) SERVER_PROTOCOL, USER_AGENT
5) SERVER_NAME, REMOTE_ADDR
6) SERVER_PORT, SERVER_SOFTWARE
7) REMOTE_USER, SERVER_PROTOCOL

14) Укажите неверное утверждение:

1) Заголовок CGI состоит из полей: Content-Type:, Location:, Status:
2) Заголовок CGI состоит как из стандартных полей: Content-Type:, Location:, Status: , так и произвольных.
3) Поле Status: позволяет вернуть CGI-скрипту статус обработки.
4) Если поле Status: не задано, то сервер подразумевает «200 OK»
5) Поле Location: означает редирект.
6) Заголовок CGI состоит как из стандартных полей: Content-Type:, Location:, Status: Server: , так и произвольных.
7) Поле Location: содержит URL ресурса, на который CGI-скрипт перенаправляет запрос

15) Укажите номер неверного высказывания:

1) assembly .NET – набор файлов, модулей и дополнительной информации, обеспечивающих простую установку приложений
2) манифест assembly позволяет скрыть от потребителя детали реализации, не требует привлечения таких средств как реестр.
3) каждый assembly имеет уникальное имя, состоящее из префикса, простого текстового имени, номера версии и информации о локализации.
4) манифест assembly содержит все файлы и модули, а также описание всех интерфейсов с внешним миром.
5) некоторые assembly могут иметь только простое текстовое имя, при использовании их как части другого приложения

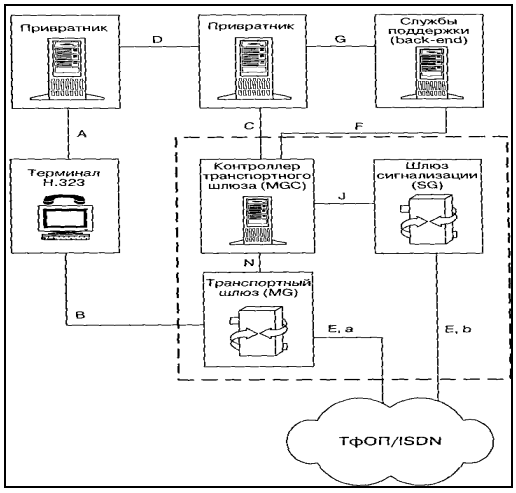
16) Укажите номер неверного высказывания:

1) MSIL можно рассматривать как ассемблер некоторой виртуальной машины.
2) MSIL представляет дополнительный уровень абстракции
3) MSIL имеет возможность обратного ассемблирования
4) как и Java bytecode MSIL имеет фиксированную разрядность платформы
5) извлечь из MSIL имена локальных переменных, констант и параметров при обратном ассемблировании может только разработчик при отладке

17) Базовым протоколом для всех приложений, связанных с интерактивной передачей речевой и видеоинформации по сети с маршрутизацией пакетов является:

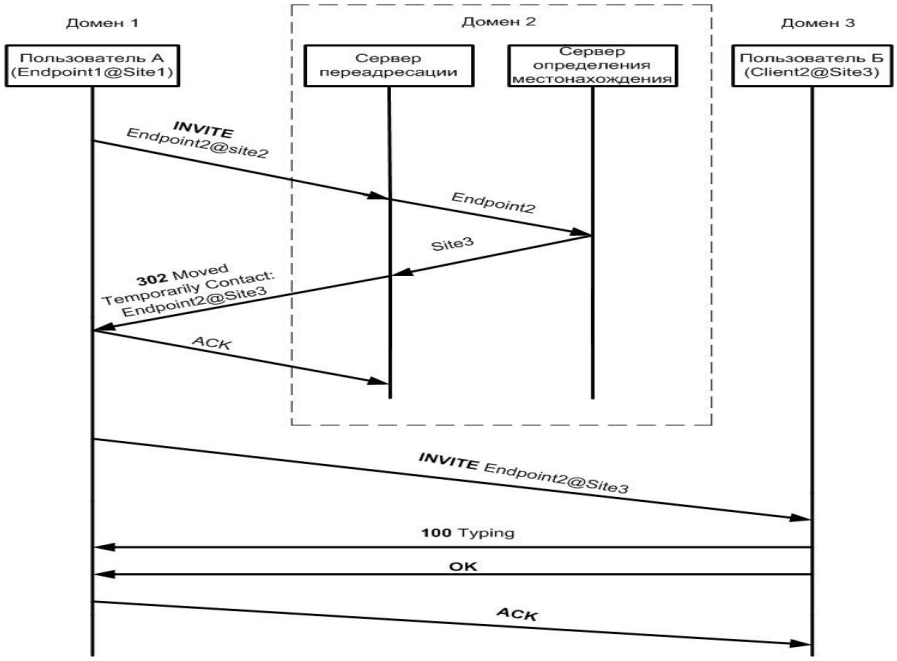
1) SIP
2) RTR
8) RTCP
9) MPLS
10) PIM
11) DVMRP
12) H.323

18) На рисунке представлена:



1) модель сети H.323
2) модель сети MGCP
3) обобщенная модель сети IP- телефонии
4) базовая модель сети IP-телефонии
5) модель сети SIP
6) модель TIPHON

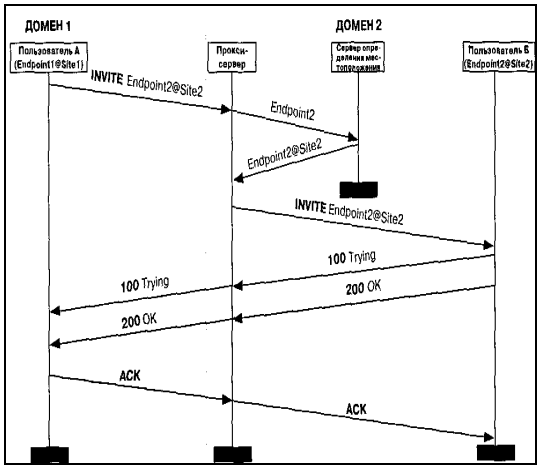
19) На рисунке представлен алгоритм установления соединения:



1) сети H.323
2) сети MGCP
3) сеть SIP с сервером переадресации
4) сети TIPHON
5) сети SIP с прокси-сервером

6) сети с сервером определения местоположения (базовая схема IP-телефонии)

20) На нижеприведенном рисунке представлен алгоритм установления соединения :



1) сети H.323
2) сети MGCP
3) сеть SIP с сервером переадресации
4) сети TIPHON
5) сети SIP с прокси-сервером
6) сети с сервером определения местоположения (базовая схема IP-телефонии)

21) Укажите неверное высказывание:

1) Сеть SIP содержит основные элементы трех видов: агенты пользователя, прокси-серверы, сервер переадресации
2) Существует три типа сетевых серверов SIP: прокси-серверы, серверы переадресации, сервер сигнализации (CS)
3) Сеть на базе MGCP содержит: шлюз, контроллер шлюзов, шлюз сигнализации.
4) Агенты пользователя SIP являются приложениями терминального оборудования и включают в себя две составляющие -агент пользователя – клиент (UAC) и агент пользователя-сервер (UAS)
5) Протокол MGCP является внутренним протоколом для обмена информацией между функциональными блоками распределенного шлюза.
6) Для передачи сообщений протокола MGCP используется протокол UDP.

22) Укажите все правильные высказывания.
BizTalk позволяет:

1) формировать системы электронного бизнеса неограниченного масштаба и сложности;
2) объединять разнородные внутрикорпоративные системы;
3) создавать электронные биржи;
4) сообщества партнеров с общим управлением процессами производства и сбыта;
5)объединять разнородные системы;
6)осуществлять эффективную публикацию и поиск информации в корпоративных интрасетях.