**Содержание**

Глава I. Структура глобальной сети Интернет 2

1.1. Подключение к сети Интернет 2

1.2. Адресация в Интернет 6

1.3. Семейство протоколов TCP/IP 10

Глава II. Интернет-технологии 16

2.3. Наиболее популярные Интернет-технологии 16

2.4. Технологии создания серверных частей Web-приложений 26

## Глава I. Структура глобальной сети Интернет

### 

### 1.1. Подключение к сети Интернет

В 1969 году Министерством обороны США была создана сеть, которая явилась предшественницей Интернета, — ARPANET. ARPANET была экспериментальной сетью. Она проектировалась дляисследований методов построения сетей, которые были бы устойчивы кчастичным повреждениям, получаемым, например,при бомбардировке, и способных в таких условиях продолжать нормальное функционирование. В ARPANET связь между компьютером-источником и компьютером-приемником (станцией назначения) присутствует постоянно. Создатели изначально предполагали, что сеть будет ненадежной, т.е. любая часть сети может исчезнуть в любой момент. Поддержка сети в рабочем состоянии возложена на связывающиеся компьютеры. Основной принцип постоения сети: любой компьютер связывается как равный с равным с любым другим компьютером.

Международной организации по стандартизации (International Standards Organization, ISO) нужны были годы, чтобы создать окончательный стандарт для компьютерных сетей, однако пользователи не ждали, а действовали. Активные пользователи стали инсталлировать программное обеспечение на все возможные типы компьютеров. Через несколько лет Интернет стал единственным приемлемым способом для связи разноплатформенных компьютеров. Такая схема подошла правительству и университетам, которые закупали компьютеры у различных производителей. Каждый закупал те компьютеры, которые ему нравились, и вправе был ожидать, что сможет работать по сети совместно с компьютерами других производителей.

Процесс улучшения сети идет постоянно. Большинство подобных изменений происходит незаметно для пользователей. На самом деле Интернет не просто сеть — это структура, объединяющая обычные сети. Интернет — это «сеть сетей». Что включает в себя Интернет? Ответ на этот вопрос меняется со временем. Раньше ответ был бы достаточно прост: все сети, работающие по протоколу IP и объединенные для формирования единой сети своих пользователей. Изначально в Интернет входили различные ведомственные сети, множество региональных сетей, сети учебных заведений и некоторые зарубежные (находящиеся за пределами США) сети.

В конце семидесятых Интернет привлек пользователей других сетей (например, BITNET, DECnet, Fidonet и т, д.), работающих не по протоколу IP. Они решили предоставлять услуги Сети своим клиентам и разработали методы подключения этих сетей к Интернету. Сначала эти подключения, названные шлюзами, служили только для передачи электронной почты. Однако некоторые из них разработали способы передачи и других услуг. Являются ли эти сети частью Интернета? И да, и нет. Все зависит от того, хотят ли они того сами.

Всемирная паутина — World Wide Web (WWW) появилась намного позже, в 1991 году. Ее автор — Тим Бернерс Ли из Европейского центра ядерных иссле­дований (CERN), расположенного в Женеве (Швейцария). Мало кому известная, появившаяся за счет энтузиазма, технология обеспечила лавинообразный рост популярности Интернета и появление того океана информации, который мы видим сейчас. Рубежом можно считать 1993 год, когда количество подключенных серверов превысило миллион. После этого пропали последние сомнения в перспективах сети сетей. Более подробно о WWW мы поговорим в четвертом модуле.

Для большинства пользователей мировой сети существует три основных информационных технологии Интернет: электронная почта (e-mail), файловые архивы FTP и World Wide Web. Чтобы воспользоваться этими полезными услугами, необходимо стать пользователем Интернет. В настоящее время существует множество способов подключения к сети. Способ доступа напрямую зависит от потребностей пользователя. Услуги подключения к сети предлагают огромное число провайдеров, но не всегда они могут удовлетворить потребности пользователей.

Как начать работу в сети? Какое оборудование для этого нужно? Постораемся подробно и осмысленно рассмотреть эти вопросы. Начнем обсуждение с выбора телекоммуникации, соединяющего пользователей между собой и/или с поставщиком Интернет-услуг. Наиболее распространенные способы соединения обычного пользователя с поставщиком сетевых услуг (провайдером) - это по коммутируемым линиям. Доступ по коммутируемым линиям телефонных сетей является пока самым доступным, дешевым и распространенным. Существует два способа подключения: dial-up connection (соединение с провайдером через телефонного оператора с максимальной пропускной способностью 56Кбит/c) и DSL (digital subscriber line - цифровая абонентская линия). Пропускная возможность DSL высока и в настоящее время этот способ подключения пользуется большой популярностью. Скорость загрузки электронных ресурсов сегодня не зависит от качества компьютера, а напрямую определяется характеристиками модема и коммутируемой линии. Следует сразу же отметить, что линии цифровой (иногда аналого-цифровой) телефонной связи существенно лучше привычной аналоговой телефонии.

Радикальным улучшением обозначено применение сотовой, спутниковой или оптоволоконной связи. Последняя сегодня реализует пропускную способность до 1 Гбит/сек. В перспективе получат распространение услуги Radio-Ethernet. Использование этого вида услуг на уровне абонентского доступа дает возможность удобного выхода в сеть Интернет через радиоканал в 2 Мбит/сек., который хотя и находится в совместном использовании несколькими абонентами, дает каждому достаточно удобный доступ, причем с возможностью голосовой связи. Абонент постоянно подсоединен к сети, запрос на нужную информацию посылается и обслуживается немедленно. Понятно, что для упомянутых здесь решений на совершенно другом (чем обозначено выше) уровне решается вопрос о выборе аппаратных приемно-передающих и компьютерных средств. Но при всем том для индивидуального пользователя в образовательных технологиях пока уместно ориентироваться на использование коммутируемых каналов и, отчасти, сотовой телефонии. При этом иногда удается добиваться гарантированной скорости передачи до 64-128 Кбит/сек. по линии основного и 1980 Кбит/сек. по линии первичного доступа.

Прежде чем говорить о информационных ресурсах Интернета, следует разобраться, каким образом можно подключиться к Всемирной сети. Ниже приведены основные способы, рис..



**Рис. . Основные способы доступа к Интернету**

Прямой (непосредственный) доступ к Интернету позволяет использовать самый полный спектр услуг Глобальной сети. Непосредственный доступ представляет собой прямой коннект локальной сети предприятия с Интернетом в качестве домена. Организация, имеющая прямой доступ, пользуется Сетью с высокой скоростью и высокой эффективностью постоянно, круглые сутки и практически 356 дней в году. Непосредственное соединение можно установить путем подключения локальной сети или рабочей станции к оптоволоконной магистрали или выделенному каналу связи при помощи соответствующего сетевого оборудования. Существует множество фирм, предлагающих такого рода доступ. Практически все серверы Интернета подключены к Всемирной сети с помощью именно этой технологии.

**1.1.1. «Доступ по вызову» (Dial-up Access)**

«Доступ по вызову» – это связь с Интернет-провайдером по коммутируемым телефонным каналам при помощи модема. Для того чтобы в течение месяца получать полный спектр услуг Интернета, достаточно иметь компьютер с установленной системой Microsoft Windows и любой модем (желателен со скоростью не ниже 14 400 bps). Достоинством такого вида доступа является относительно низкая стоимость услуг. Недостатками Dial-up соединения можно считать низкое качество и скорость канала передачи данных, высокую вероятность разрыва связи в самый неподходящий момент, длительное время, затрачиваемое пользователем на дозвон до провайдерского пула и авторизацию, и, наконец, занятость телефонной линии на протяжении всего сеанса работы в Интернете.

**1.1.2. Доступ по протоколам SLIP и РРР**

Использование такого вида доступа подразумевает наличие специального программного обеспечения, поддерживающего протоколы SLIP (Serial Line Internet Protocol) и РРР (Point to Point Protocol). Договорившись с поставщиком услуг о том, что работа в Интернете будет проходить именно по этим протоколам, предприятие подключается к Сети. После этого, как и при непосредственном доступе, в распоряжении предприятия абсолютно все ресурсы Глобальной сети, более того, оно входит в нее составной частью само, а не через сервер провайдера Интернета.

При работе со SLIP и РРР нет необходимости прокладывать в офис оптоволоконную магистраль, можно использовать выделенную коммутируемую телефонную линию, что гораздо дешевле. Единственным непременным условием для работы является то, что на компьютере должен быть установлен высокоскоростной модем.

SLIP является сетевым протоколом, позволяющим использовать для прямого выхода в Интернет обыкновенную телефонную линию и модем. Однако для работы потребуется специальное программное обеспечение. SLIP работает совместно с протоколом ТСР/IР как протокол более низкого уровня. Принимаемую через модем закодированную информацию SLIP анализирует и преобразует в IP-пакет, который передается программной оболочке более высокого уровня, например приложению класса клиент/сервер, которое, получив его, преобразует этот пакет соответственно директивам протокола TCP.

На отправку данная система работает так: SLIP получает от программы клиент/сервер сгенерированный ею IP-пакет, затем выбирает из него всю ценную информацию, преобразует ее в символьный код и отправляет в Интернет.

РРР – это более современная и совершенная версия протокола SLIP. Это более эффективный, быстрый и функциональный протокол. Соответственно, РРР более дорог, чем его аналог. Различий в алгоритмах обработки информации между ними практически нет. SLIP и РРР подходят для прямого подключения к Интернету, как локальных сетей, так и домашних компьютеров, используемых в качестве сервера. Такой доступ несоизмеримо дешевле непосредственного, но и намного дороже доступа по вызову. Тем не менее, здесь играет значительную роль скорость модема и пропускная способность телефонной линии. Именно эти факторы зачастую делают использование SLIP и РРР невозможным.

Существует еще один протокол, который называется CSLIP (Compressed SLIP), созданный фирмой Lawrence Berkley Labs для телефонных линий с малой пропускной способностью. CSLIP – это модификация SLIP. Выигрышь в скорости создается за счет универсального алгоритма сжатия заголовков IP-пакетов. На низкоскоростных линиях этот протокол позволяет увеличить эффективность связи почти в 6 раз и немного компенсировать ее слабые возможности.

Недостатком CSLIP является невысокая его надежность. Протокол сжимает заголовки IP-пакетов, начиная со второго и используя при этом информацию о заголовке из первого, несжатого пакета. Создается логическая последовательность IP-информации, и если одна из ее составляющих теряется, то есть один из пакетов последовательности был передан некорректно, CSLIP прерывает передачу и добросовестно начинает все заново. При плохой связи можно потерять на использовании CSLIP больше, чем при применении SLIP и РРР.

**1.1.3. Доступ UUCP**

Доступ UUCP (Unix-to-Unix CoPy) предназначен для пользователей платформ UNIX. UUCP используется для обмена файлами между двумя компьютерами. Возможности, предоставляемые данным видом доступа, ограниченны лишь работой с электронной почтой и новостями. Суть подключения UUCP сводится к тому, что на самом деле никакого Интернета на машине пользователя не будет. Доступом к Интернету владеет только провайдер, которой и предоставляет услуги UUCP. Компьютер пользователя соединяется в устанновленное время сервером провайдера и отправляет и принимает электронную почту. Дальше уже сетевые программы удаленного ПК будут пересылать корреспонденцию пользователя во Всемирную сеть.

**1.1.4. Подключение по технологии «coax at home»**

Технология «coax at home» позволяет получать доступ к Интернету, используя каналы кабельной телевизионной сети. В упрщенном виде такая структура строится следующим образом: используется подключение стандартного оборудования кабельного телевизионного центра, предназначенного для вещания, к Интернету через специальное устройство передачи данных, называемое головным модемом. Модем подключается посредством маршрутизатора к высокоскоростному каналу Интернета. После этого абоненту достаточно лишь установить на своем компьютере любую сетевую карту, поддерживающую протокол 10 Base-T или 100 Base-T, подключить ее к кабельному модему, а тот, в свою очередь, к расположенному в квартире антенному выходу. Основной элемент клиентской компьютерной системы в схеме кабельной сети – это кабельный модем, который представляет собой двунаправленный аналогово-цифровой преобразователь данных, использующий в процессе передачи информации принцип наложения на несущую частоту модулированного аналогового сигнала. Фундаментальное отличие данного аппарата от обыкновенного модема заключается в том, что кабельный модем не требует установки каких-либо драйверов, поскольку он подключается к компьютеру посредством сетевой карты и является абсолютно прозрачным для системы: машина считает, что она работает в локальной сети. Следовательно данному модему совершенно безразлично, какая операционная система проинсталлирована на пользовательском ПК. Необходимо лишь, чтобы эта система поддерживала возможность установки сетевой карты и настройки локальной сети. Для работы в Интернете абонент может применять любое стандартное программное обеспечение.

Главные преимущества доступа к Интернету по методу «coax at home» - это высокая стабильность соединения, отсутствие непредвиденных разрывов связи, а также то, что на протяжении всего сеанса работы во Всемирной сети телефонная линия остается свободной.

**1.1.5. Выбор поставщика информационных услуг**

Существует и другая сторона проблемы выхода в сеть Интернет: какой способ соединения предпочтителен с точки зрения выбора поставщика информационных услуг, или, точнее звена во всей цепи сетевого взаимодействия, к которому резонно подсоединиться.

В первом приближении существуют четыре основных различных способа соединения, причем большинство из них могут быть реализованы и для больших компаний, и для индивидуальных пользователей. Ниже приводится краткая характеристика существующих возможностей подключения к Интернет:

* **Прямое подключение (direct connection) к Интернет.** Подключение к магистральным каналам (backbone) Интернет осуществляется через специальный компьютер - шлюз (gateway). При этом способе возможен полный доступ ко всем услугам сети, но он считается достаточно дорогим в реализации и поддержке. Такой способ подключения приемлем для больших организаций.
* **Соединение через предоставленный шлюз.** При этом способе предполагается аренда чужого шлюза для полного доступа ко всем услугам сети. Такой способ может быть удобен для студентов, но вне вуза будет трудно найти организацию, которая позволит пользоваться своим шлюзом.
* **Использование прямого (direct) Интернет-провайдера.** При этом подключении компьютер пользователя соединяется со специальным шлюзом компании, обеспечивающей ограниченный или полный доступ к услугам Интернет. Отличие этих компаний от on-line-служб в том, что их роль состоит только в предоставлении шлюза в Интернет. Чаще всего Интернет-провайдеры такого типа пользуются выделенными соединениями по высокоскоростным каналам для обеспечения высокого качества сервиса. Пример прямого Интернет-провайдера - UUNET Technologies.
* **Использование косвенного (indirect) Интернет-провайдера.** В этом случае привлекают on-line-службу (типа Delphi или CompuServe) для получения доступа к некоторым или ко всем услугам Интернет. Данный способ доступа подходит для большинства частных лиц, хотя не все сервис-провайдеры предлагают все возможности Интернет.

Итак, каждый пользователь сам может подобрать для себя оптимальный вариант подключения к сети в зависмости от необходмых услуг, времени доступа и финансовых затрат.

### 

### 1.2. Адресация в Интернет

**1.2.1. Универсальный адрес сетевого ресурса**

Широкое использование компьютерных сетей неспециалистами на сегодня стало возможно благодаря разработке простых в применении средств доступа к многообразным ресурсам Интернет. В первую очередь это относится к появлению специальных программ-клиентов WWW, называемых браузерами, обладающих “дружественным” графическим интерфейсом (т.е. способом взаимодействия пользователя с программой).

Схема адресации в иерархически организованной файловой системе, характерной для таких популярных операционных систем, как DOS и Unix, позволяет однозначно идентифицировать заданный файл путем указания его времени и уникального адресного пути к нему.

**Пример:**

|  |  |
| --- | --- |
| В DOS C:\DOS\PROGR\FILE1.TXT | (Запись с именем file1.txt расположена в корне на диске C, в директории DOS, поддиректории PROGR) |
| В Unix  /users/data/Letters.html | (Запись с именем Letters.html расположена в корне, в директории users, поддиректории data) |

Важная для избежания ошибок при организации доступа к файлам разница в этих схемах состоит в использовании разных слэшей - прямого (/) в Unix и обратного (\) в DOS, а также чувствительность к верхнему и нижнему регистру, свойственная только Unix. За основу схемы адресации ресурсов в Сети принята нотация Unix, которая претерпела естественные расширения за счет приписывания к существующей схеме слева имени протокола доступа к заданному ресурсу, где расположен ресурс; а справа после служебных символов (#, ?) имени метки внутри файла или элементов поискового ресурса. В примерах

* http://www.dvgu.ru/users/data/Lessons.htm#Page1
* http://www.dvgu.ru/cgi-bin/proc?corn

содержится обращение по протоколу http к компьютеру с доменным именем www.dvgu.ru с попыткой доступа в первом случае к области файла Lessons.htm с меткой Page1, а во втором – реализация поискового запроса к программе proc с термином для поиска "corn" на том же сервере.

Отметим, что именно в таком виде и вводятся строки запроса на ресурсе в специально отведенном поле браузера, после чего нажатием клавиши Enter инициируется соединение и загрузка.

Итак, в основу построения адреса ресурса в Сети оказались заложены следующие понятия и принципы:

**Расширяемость** - новые адресные схемы должны были легко вписываться в существующий синтаксис URI (Uniform Resource Identifier - универсальный индикатор ресурса).

**Полнота** - по возможности, любая из существующих схем должна описываться посредством URI.

**Читаемость** - адрес должен быть легко читаем человеком, что вообще характерно для технологии WWW.

Расширяемость была достигнута за счет выбора определенного порядка интерпретации адресов, который базируется на понятии “адресная схема”. Идентификатор схемы стоит перед остатком адреса, отделен от него двоеточием и определяет порядок интерпретации остатка.

Полнота и читаемость порождали коллизию, связанную с тем, что в некоторых схемах используется двоичная информация. Эта проблема была решена за счет формы предоставления такой информации. Символы, которые несут служебные функции, и двоичные данные отображаются в URI в шестнадцатеричном коде и предваряются символом "%".

Прежде, чем рассмотреть различные схемы представления адресов, приведем еще один пример простого адреса URI:

http://lemoi.phys.dvgu.ru/wst/index.html

Перед двоеточием стоит идентификатор схемы адреса - "http". Это имя отделено двоеточием от остатка URI, который называется "путем". В данном случае путь состоит из доменного адреса машины, на которой установлен сервер HTTP, и пути от корня дерева сервера к файлу "index.html".

Кроме представленной выше полной записи URI существует упрощенная. Она предполагает, что к моменту ее использования многие параметры адреса ресурса уже определены (протокол, адрес машины в Сети, некоторые элементы пути).

При таких предположениях автор гипертекстовых страниц может указывать только относительный адрес ресурса, т.е. адрес относительно определенных базовых ресурсов.

**1.2.2. Схемы адресации сетевых ресурсов**

В RFC-1630 (Request for Comment - документы с таким названием содержат в себе материалы по Интернет-технологии, которые доведены до уровня стандарта или близки к этому уровню) рассмотрено восемь схем адресации Интернет и указаны две, синтаксис которых находится в стадии обсуждения.

**Схема HTTP.** Это основная схема для WWW. В схеме указываются ее идентификатор, адрес машины, TCR-порт, путь в директории сервера, поисковый критерий и метка.

Следует отличать понятие TCR-порта от физического разъема на задней стенке системного блока компьютера. В Интернет принято идентифицировать конкретную прикладную программу с определенным числом, или портом (это понятие абсолютно не связано с названием физического устройства ввода-вывода компьютера). Всякий раз передаваемый по сети от одного компьютера к другому пакет данных содержит информацию о том, какой именно протокол содержит информацию о том, какой именно протокол используется и с какой прикладной программой машины пытается установить связь. Номер порта и обозначает эту прикладную программу.

Приведем несколько примеров URI для схемы HTTP:

http://lemoi.phys.dvgu.ru/wst/test.html

Это наиболее распространенный вид URI, применяемые в документах WWW. Вслед за именем схемы (http) следует путь, состоящий из доменного адреса машины и полного адреса HTML-документа в дереве сервера HTTP.

В качестве адреса машины допустимо использование и IP-адреса:

http://144.206.160.40/test/test.html

Если сервер протокола HTTP запущен на другой, отличный от 80 порт TCP, то это отражается в адресе:

http://l44.206.130.137:8080/primorye/index.html

При указании адреса ресурса возможна ссылка на точку внутри файла HTML. Для этого вслед за его именем может быть указана метка внутри документа:

http://lemoi.phys.dvgu.ru/wst/test.html#first

Символ # отделяет имя документа от имени метки. Другая возможность схемы HTTP – дача параметров. Первоначально предполагалось, что в качестве параметров будут передаваться ключевые слова, но по мере развития механизма сервисных программ (скриптов) в качестве параметров стала передаваться и другая информация.

http://lemoi.phys.dvgu.ru/wst/isindex.html?keyword1+keyword2

В данном примере предполагается, что документ "isindex.html" - документ с возможностью поиска по ключевым словам. При этом в зависимости от поисковой машины (программы, реализующей поиск) знак "+" будет интерпретироваться либо как "AND", либо как "OR". Вообще говоря, "+" заменяет " " (пробел) и относится к классу неотображаемых символов. Если необходимо передать такой символ в строке параметров, то следует передавать в шестнадцатеричном виде его ASCII-код.

**Схема FTP**. Данная схема позволяет адресовать файловые архивы FTP из программ-клиентов World Wide Web. При этом программа должна поддерживать протокол FTP. В данной схеме возможно указание не только имени схемы, адреса FTP-архива, но и идентификатора пользователя и даже его пароля. Наиболее часто используется для доступа к публичным архивам FTP:

ftp://dvo.ru/pub/movie.avi

В этом случае записана ссылка на архив "dvo.ru" с идентификатором "anonymous" или "ftp" (анонимный доступ). Если есть необходимость указать идентификатор пользователя и его пароль, то можно это сделать перед адресом машины:

ftp://nobody:password@dvo.ru/users/local/pub

Здесь параметры отделены от адреса машины символом "@", а друг от друга – двоеточием. В некоторых системах можно указать и тип передаваемой информации, но данная возможность не стандартизирована. Следует также учитывать, что употребление идентификатора пользователя и его пароля не рекомендовано, так как данные передаются незашифрованными и могут быть перехвачены. Реальная защита в WWW осуществляется другими средствами и построена на других принципах.

**Схема Gopher**. Данная схема используется для ссылки на ресурсы распределенной информационной системы Gopher. Схема состоит из идентификатора и пути, в котором указывается адрес Gopher-сервера, тип ресурса и команда Gopher:

gopher://gopher.dvgu.ru:70:/7/kuku

В этом примере осуществляется доступ к gopher-серверу gopher.dvgu.ru через порт 70 для поиска (тип 7) слова "kuku". Следует заметить, что gopher-тип, в данном случае 7, передается не перед командой, а вслед за ней.

**Схема MAILTO**. Данная схема предназначена для отправки почты по стандарту RFC-822 (стандарт почтового сообщения). Общий вид схемы выглядит так:

mailto:samardak@dvgu.ru

**Схема TELNET**. По этой схеме осуществляется доступ к ресурсу в режиме удаленного терминала. Обычно клиент вызывает дополнительную программу для работы по протоколу telnet. При использовании этой схемы необходимо указывать идентификатор пользователя, допускается использование пароля. Реально доступ осуществляется к публичным ресурсам, и идентификатор и пароль являются общеизвестными, например их можно узнать в базах данных Hytelnet.

telnet://guest:password@apollo.dvgu.ru

**Схема FILE**. WWW-технология используется как в сетевом, так и в локальном режимах. Для локального режима используют схему FILE.

file:///C/text/html/index.htm

В данном примере приведено обращение к локальному документу на персональном компьютере MS-DOS или MS-Windows.

Существует еще несколько схем, которые на практике используются редко или находятся в стадии разработки, поэтому останавливаться на них не будем.

Из приведенных выше примеров видно, что спецификация адресов ресурсов URI является довольно общей и позволяет адресовать практически любой ресурс Интернет. При этом число ресурсов может расширяться за счет создания новых схем. Они могут быть похожими на существующие, а могут и отличаться от них. Реальный механизм интерпретации идентификатора ресурса, опирающийся на URI, называется URL (Uniform Resource Locator), и пользователи WWW имеют дело именно с ним.

**1.2.3. Регистрация имени домена**

Домены в различаются различаются по уровням иерархии, например в “lemoi.phys.dvgu.ru” lemoi – домен третьего уровня, phys – второго, dvgu – первого, а ru - верхнего. Создавая домен, необходимо зарегистрировать его в руководящей организации, тогда имя домена будет включено в имя ее домена. Домены верхнего уровня классифицируют организации по типам (используется в США): gov (government - государственные), edu (educational - образовательные), org (organization - организации), net (главные центры поддержки сети), mil (военные группы), int (международные), com (commercial - коммерческие), <country code> (любая страна, географическая единица). В 2004 году были приняты новые типы доменов верхнего уровня:

**biz** – домен для крупных и средних компаний, продвигающих свой бизнес;

**info** – информационный домен, который может быть использован для представления своей продукции или услуги в Интернет;

**name** – индивидуальный домен для представления персоналий в Интернете;

**coop** – домен для организаций стремящихся к кооперации;

**aero** – домен для авиакомпаний;

**museum** – домен для музеев.

Чтобы присоединиться ко всем, кроме государственного домена, необходимо иметь аргументы, соответствующие предъявляемым строгим требованиям. Включение в домен com гораздо проще, однако все-таки нужно правильно определить, к какому из доменов верхнего уровня относится организация.

Имя домена должно иметь смысл, легко запоминаться и вводиться с клавиатуры, а также не использоваться другой организацией в Интернете.

Выбранное подходящее имя регистрируется. Обычно для этого из области Registration Web-страницы InterNIC (http://www.internic.net/) получают текстовый бланк и заполняют его в любом редакторе или текстовом процессоре или заполняют форму WWW, используя программу просмотра Web.

Необходимо сообщить InterNIC о себе некоторые данные. Во-первых, кто будет контактировать с ней по административным, техническим или финансовым вопросам, касающимся домена. Во-вторых, имена и IP-адреса серверов DNS (Domain Name Service), поддерживающих домен.

Заполненная форма отсылается электронной почтой в InterNIC. Через некоторое время поступают два ответа: первый - подтверждение получения запроса, второй - разрешение на использование имени домена.

### 

### 1.3. Семейство протоколов TCP/IP

**1.3.1. Введение в TCP/IP**

Протокол в сетевых технологиях определяется как четко регламентированный свод правил, которые одинаково применимы для различных систем (программ, шлюзов, пакетов данных и др.), участвующих в передаче информации. Благодаря протоколам, взаимодействие этих систем происходит по заранее установленному сценарию.

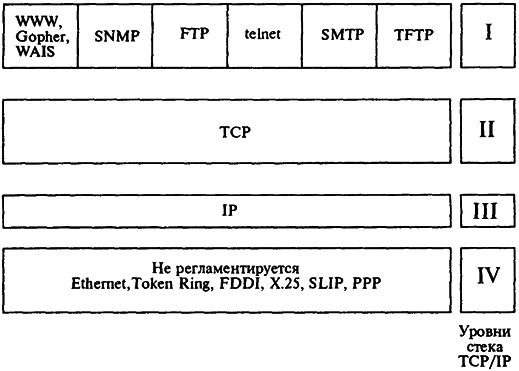
С целью лучшего понимания понятия протокола, рассмотрим один пример, который не имеет отношения к компьютерным сетям. Предположим, что в некотором регионе расположены два предприятия А и Б, которые сотрудничают друг с другом, рис. 10. Делавая активность предприятий базируется на многочисленных договоренностях и соглашениях, таких, например, как регулярные поставки продукции одного предприятия другому. Исходя из этой договоренностьи начальник отдела продаж предприятия А каждый месяц должен посылать сообщение начальнику отдела закупок предприятия В о том, сколько и какого товара они могут поставить в этом месяце. В ответ на это сообщение начальник отдела закупок отправляет заявку на нужный объем продукции во внутренний отдел производства. Такой порядок взаимодействия между начальниками или директорами в данном случае соответствует понятию «протокол уровня директоров». У начальников есть секрктари, через которых отправляются все сообщения.

После того как сообщения поступили секретарям, начальников не не интересуют способы отправки и дальнейшего передвижения письма - обычной или электронной почтой, факсом или курьером. Выбор способа передачи лежит на секретарях, которые решают этот вопрос, не уведомляя об этом начальство, так как их протокол взаимодействия завязан только на передаче сообщений, поступающих сверху, и не касается содержания этих сообщений. После отправки письма, секретари считают свою работу выполненной. Однако хороший секретарь может еще проверить получил ли адресат сообщение.

При решении разных проблем начальники могут взаимодействовать по другому сценарию, но это не отразится на работе секретарей, для которых не важно, какие сообщения отправлять, а важно, чтобы они дошли адресату. Другими словами, мы имеем два уровня – уровень начальников и секретарей, и каждый уровень имеет собственный протокол, который может быть изменен независимо от протокола другого уровня. Так как протоколы независимы друг от друга, то это дает возможность использовать многоуровневый подход.

Перейдя от примера к реальной сети, можно увидеть схожую картину. По мере движения пакета данных по сети на разных последовательных этапах его взаимодействия с другими сетевыми элементами отрабатывают протоколы разных уровней. Полная совокупность таких протоколов, используемых для успешного взаимодействия разных элементов в рамках сети данного типа, называют семейством или стеком. Интернет работает под семейством протоколов TCP/IP, которое имеет многоуровневую структуру. TCP/IP расшифровывается как Transmission Control Protocol/Интернет Protocol (Протокол управления передачей данных/Протокол Интернет).

Структура протоколов TCP/IP имеет четыре основных уровня и приведена на . рис.



**Рис.. Стек TCP/IP**

Самый нижний (уровень IV) соответствует уровню доступа к сети, который работает на основе популярных стандартных протоколах физического и канального уровня, таких, как Ethernet, Token Ring, SLIP, PPP и других. Протоколы этого уровня отвечают за пакетную передачу данных в сети на уровне аппаратных средств.

Следующий уровень (уровень III) обеспечивает межсетевое взаимодействие при передаче пакетов данных из одной подсети в другую. В качестве протокола в стеке задействован протокол IP.

Следующий уровень (уровень II) является основным и работает на базе протокола управления передачей TCP. Этот протокол необходим для надежной передачи сообщений между размещенными на разных машинах прикладными программами за счет образования виртуальных соединений между ними.

Все обозначенные выше уровни можно отнести к «уровню секретарей» из примера, описанного выше. Пользователя («начальника») больше всего интересует правильная работа самого верхнего уровня (уровень I), или «уровня начальников», который называется на языке стека TCP/IP прикладным.

Так как стек TCP/IP существует достаточно давно, то он включает в себя большое количество протоколов и сервисов прикладного уровня. Наиболее важными и популярными являются следующие протоколы: протокол передачи файлов FTP, протокол Telnet, протокол Gopher для доступа к ресурсам всемирного пространства GopherSpace, и самый известный протокол HTTP для доступа к удаленным гипертекстовым базам данных во всемирный паутине.

Классифицируем протоколы по характеру выполняемых задач.

**Транспортные протоколы**предназначены для управления передачей данных между двумя машинами:

* **TCP** (Transmission Control Protocol). Этот протокол поддерживает передачу данных, которая основана на логическом соединении между передающим и принимающим компьютерами.
* **UDP** (User Datagram Protocol). Этот протокол обеспечивает передачу данных без установления логического соединения. Другими словами данные отправляются без предварительно установленного соединения между компьютерами получателя и отправителя.

**Протоколы маршрутизации**предназначены для обрабатки адресов данных и определения наилучших путей до адресата. Кроме того, они обеспечивают разбиение больших сообщений на несколько маленьких сообщений, которые затем передаются последовательной цепочкой и компонуются в единое целое на компьютере-получателе:

* **IP** (Internet Protocol). Протокол обеспечивает фактическую передачу данных.
* **ICMP** (Internet Control Message Protocol). Протокол обрабатывает сообщения состояния для IP протокола, например, ошибки и изменения в сетевых аппаратных средствах, которые влияют на маршрутизацию.
* **RIP** (Routing Information Protocol). Один из нескольких протоколов для определения наилучшего маршрута доставки сообщения.
* **OSPF** (Open Shortest Path First). Альтернативный протокол для определения маршрутов.

**Протоколы поддержки сетевого адреса**используются для идентификации машины с уникальным номером и именем.

* **ARP (Address Resolution Protocol).** Протокол определяет уникальные числовые адреса машин в сети.
* **DNS (Domain Name System).** Протокол для определения числовых адресов по именам машин.
* **RARP (Reverse Address Resolution Protocol).** Протокол для определения адреса машин в сети, но способом, обратным ARP.

**Прикладные сервисы**представляют собой пользоватеьские программы, которые используются для получения доступа к различным услугам.

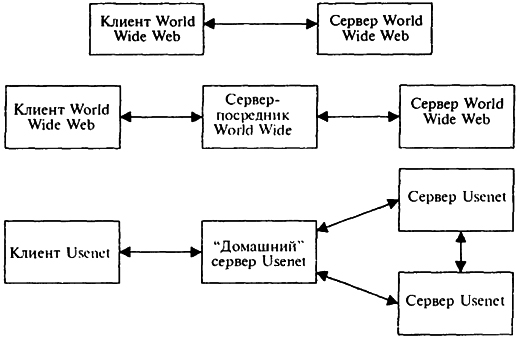
* **ВООТР (Boot Protocol).** Протокол загружает сетевую машину, читая информацию для начальной загрузки с сервера.
* **FTP (File Transfer Protocol).** Протокол передачи файлов между компьютерами.

**1.3.2. Сетевое взаимодействие «клиент-сервер»**

В основу взаимодействия компонентов информационных сервисов Сети в большинстве случаев положена модель «клиент-сервер». Как правило, в качестве клиента выступает программа, которая установлена на компьютере пользователя, а в качестве сервера - программа, установленная у провайдера. В данном контексте под провайдером понимаем организацию или частное лицо, которые ведут (поддерживают) информационные ресурсы.

При этом возможны два варианта организации самой информационной системы, которая обеспечивает доступ к информационному ресурсу. Большинство систем Интернет построены по принципу взаимодействия "каждый с каждым", например система World Wide Wed, т.е. каждый пользователь может напрямую взаимодействовать с каждым сервером без посредников. Такой подход позволяет упростить всю технологическую схему построения системы, однако приводит к порождению большого трафика в Сети. Альтернативный вариант построения системы, например системы Usenet, когда пользователь может взаимодействовать только со «своим» сервером и не может обратиться к произвольному серверу в Сети. Однако доступ он получает ко всей информации, которая присутствует в данной информационной системе, так как серверы обмениваются ею между собой.

В ряде случаев возможен выбор между первым способом реализации информационного обслуживания и вторым, например, это возможно в службе доменных имен DNS. Администратор сервера может настроить его для работы через другой сервер или непосредственно с программами-клиентами. Аналогично настраиваются и специальные серверы-посредники для различных информационных серверов Интернет. Несколько таких схем показано на рис.12.



**Рис.12. Различные схемы «Клиент-Сервер» для информационных сервисов Интернет**

Принципиальным различием между схемой с посредником и схемой Usenet является то, что при посреднике работа по доступу к ресурсу перекладывается на его плечи. При этом он будет устанавливать соединение с каждым сервером в сети. По схеме Usenet это делать не обязательно, так как информацию в принципе можно получить с любого сервера.

**1.3.3. Недостатки протоколов**

Проблемы защиты информации являются "врожденными" практически для всех протоколов и служб Интернет.

Система имен доменов (Domain Name System - DNS) представляет собой распределенную базу данных, которая преобразует имена пользователей и хостов в IP-адреса и наоборот. DNS также хранит информацию о структуре сети компании, например количестве компьютеров с IP-адресами в каждом домене. Одной из проблем DNS является то, что эту базу данных очень трудно "скрыть" от неавторизованных пользователей. В результате, DNS часто используется хакерами как источник информации об именах доверенных хостов.

FTP (File Transfer Protocol) обеспечивает передачу текстовых и двоичных файлов, поэтому его часто используют в Интернет для организации совместного доступа к информации. На FTP-серверах хранятся документы, программы, графика и любые другие виды информации. Некоторые FTP-серверы ограничивают доступ пользователей к своим архивам данных с помощью пароля, другие же предоставляют свободный доступ (так называемый анонимный FTP-сервис). Если вы используете опцию анонимного FTP для своего сервера, то должны быть уверены, что на нем хранятся только файлы, предназначенные для свободного распространения.

Sendmail - популярная в Интернет программа электронной почты, использующая для своей работы некоторую сетевую информацию, такую как IP-адрес отправителя. Перехватывая сообщения, отправляемые с помощью Sendmail, хакеры могут использовать эту информацию для нападений, например для спуфинга (подмены адресов).

SMTP (Simple Mail Transfer Protocol) - протокол, позволяющий осуществлять почтовую транспортную службу Интернет. Одна из проблем безопасности, связанная с этим протоколом, состоит в том, что пользователь не может проверить адрес отправителя в заголовке сообщения электронной почты. В результате хакер может послать в вашу сеть большое количество почтовых сообщений, что приведет к перегрузке и блокированию работы вашего почтового сервера.

Telnet - сервис Интернет, при осуществлении которого пользователи должны регистрироваться на сервере Telnet, вводя свое имя и пароль. После аутентификации пользователя его рабочая станция функционирует в режиме «тупого» терминала, подключенного к внешнему хосту. С этого терминала пользователь может вводить команды, которые обеспечивают ему доступ к файлам и возможность запуска программ. Подключившись к серверу Telnet, хакер может сконфигурировать его программу таким образом, чтобы она записывала имена и пароли пользователей.

Как уже говорилось, стек TCP/IP представляет собой набор протоколов, которые используются в Интернет и интрасетях для передачи пакетов между компьютерами. При передаче информация заголовков пакетов может подвергнуться нападению хакеров. Например, хакеры могут подменить адрес отправителя в своих пакетах, после чего они будут смотреться как пакеты, передаваемые авторизованным клиентом.

**1.3.4. Программное обеспечение для работы в сети**

Для использования информационных ресурсов Интернета необходимо определенное программное обеспечение. Для просмотра html-документов необходима программа, динамически обрабатывающая код HTML и отображающая web-страницы. Такая программа называется браузером (от англ. browser). Выделяют два основных класса браузеров: браузеры с поддержкой отображения графических элементов web-страниц и браузеры без такой поддержки. Большинство современных браузеров работают с графикой, включая flash анимацию и видео.

Самыми популярными браузерами являются Microsoft Internet Explorer и Netscape Navigator, предназначенные для работы в среде Microsoft Windows. Менее распространены Opera, Arache (работает под управлением DOS) и текстовый браузер для UNIX-совместимых платформ под названием Lynx.

Следует отметить, что браузеры имеют разные встроенные интерпретаторы HTML, из-за чего один и тот же html-документ может отображаться в них по-разному. Ощутимые различия наблюдаются не только между клиентским программным обеспечением разных производителей, но и между двумя версиями одной и той же программы. Самые существенные расхождения в алгоритмах обработки кода HTML имеются в двух наиболее популярных браузерах, созданных компаниями, которые никак не могут договориться между собой, чтобы достичь соответствия в отображении их продукцией интернетовских web-страниц. Речь об Интернет Explorer и Netscape Navigator.

## Глава II. Интернет-технологии

### 

### 2.3. Наиболее популярные Интернет-технологии

**2.3.1. Язык HTML**

Сервис World Wide Web объединяет каналы передачи связной информации по всему земному шару. Язык HTML (HyperText Markup Language) предоставляет простой формат для предоставления этой информации. Для безконфликтной работы необходимо, чтобы все программы, совместимые с WWW, могли поддерживать язык HTML. Сервис WWW использует HTTP протокол, который позволяет передавать кодированную информацию между клиентом и сервером.

Уже не раз в учебнике упоминалось, что гипертекст и гипермедиа являются для WWW фундаментальными технологиями. Если развивать идею гипертекста и включить в него графику, аудио и видео файлы, анимацию, то мы получим, так называемое, гипермедиа.

Язык разметки гипертекста HTML берет свое начало от языка стандартной обобщенной разметки (Standard Generalized Markup Language - SGML). Язык SGML используется для создания структурированных документов, а также языков для их разметки. SGML можно считать глобальным стандартом для описания языков разметки гипертекста, таких, как HTML. Документы SGML являются программно- и аппаратно-независимыми, они не привязаны к какой-либо конкретной программе, компьютеру или устройству вывода. SGML был одобрен ISO (International Standard Organization - Международная организация стандартов) в 1986 г. и является стандартом для многих коммерческих и государственных систем создания документов.

Документ HTML представляет собой обычный текстовый файл, который заключен между последовательностью команд форматирования.

Язык разметки HTML обычно используют для представления:

* новостей, почты, рекламы, текстов;
* меню с опциями;
* результатов запросов к базам данных;
* простых структурированных документов со встроенной графикой;
* обзоров имеющейся информации.

Для описания html-документа используется тэговая модель, которая описывает документ как совокупность тегов-контейнеров, внутри которых заключаются информационные объекты (контент). Язык гипертекстовой разметки - это система для разметки текста с помощью команд форматирования. Нечто, заключенное в <угловые скобки>, называется тэг, считается командой и не отображается.

Например, <Т> используется для начала отображения курсива и</Т> для окончания. Некоторые команды требуют наличия начального и конечного тэгов, как например как например <BR> (разрыв), этого не требуют. Интерпретация тэгов не зависит от регистра символов; <BR> и <br> интерпретируется одинаково.

Следующие шесть строк появляются практически во всех Web-страницах:

<HTML>

<HEAD>

</HEAD>

<BODY>

</BODY>

</HTML>

Поскольку тэги HTML не чувствительны к регистру, вы можете использовать любую схему преобразования букв в прописные. Некоторые атрибуты тэга чувствительны к регистру, и тогда они будут отображаться так, как этого требует стандарт.

* <HTML> и </HTML> отмечают начало и конец страницы.
* <HEAD> и </HEAD> отмечают начало и конец заголовка в неотображаемой браузером части страницы.
* <BODY> и </BODY> отмечают отображаемую область страницы.

Когда тэг имеет несколько параметров, они отделяются друг от друга символами пробела, табулятора, конца строки или любой их комбинацией, например:

<IMG SRC="picture.gif">

Первый параметр определяет тип тэга: *IMG* означает ссылку на изображение. *SRC="picture.gif*" показывает URL файла, в котором размещено изображение. Браузер отображает этот файл вместо тэга.

*IMG* и *SRC* разделены между собой пробелом. Ниже приведен несколько более сложный тэг:

<font face="Arial Narrow" size = 2 color=#FF0000>

Браузер отображает все, что расположено между этим тэгом и замыкающим последовательность <FONT> тэгом красным цветом и относительно малого размера. Без кавычек браузер определил бы атрибут FACE тэга <FONT> как Arial и потерял бы Narrow как нечто непонятное.

Магическая строка #FF0000 означает красный цвет, поскольку каждая пара шестнадцатиричных цифр отвечает за интенсивность красного, зеленого и голубого в суммарном цвете. FF (255 в десятичной нотации) означает наиболее интенсивный цвет; 00 - нулевой.

Ниже приводятся сайты на которых размещена полная документация по всем возможностям HTML.

**Microsoft**

*http://msdn.microsoft.com/workshop/entry.asp*

**Netscape**

*http://developer.netscape.com/library*/*documentation*/*communicator/*

**World Wide Web Consortium**

*http://www.w3.org/MarkUp*

**2.3.2. Таблицы каскадных стилей**

Даже с такими тэгами, как <В> (жирный), <I> (курсив) и <FONT> HTML поддерживает ограниченные возможности в отображении текста. В течении многих лет Web-дизайнеры требовали больших возможностей по управлению текстом и, наконец, дождались появления Cascading Style Sheets (CSS) - таблиц каскадных стилей.

Таблица стилей единожды описывает специфический формат, присваивает ему имя, и далее он употребляется там, где потребуется. В HTML вы можете определить стиль, например, так:

<style> <!-- .warning {font-family: Arial Narrow, sans-serif; font-size: 10pt; color: rgb(255,0,0)}

</style>

CSS-имена (селекторы), которые начинаются с точки - это определяемые пользователем имена классов. Селекторы, которые начинаются с букв, называются *типами* и переопределяют встроенные HTML-стили, такие как Р (параграф) или В (жирный шрифт).

Атрибуты класса определяют стиль по имени, которое назначается свойству между тэгами <STYLE> и </STYLE>. Атрибут STYLE определяет свойства CSS для тэга, который его содержит.

Мы определили стиль *warning,* который применяет для любого текста шрифт Arial Narrow, а если он недоступен, то использует любой рубленый шрифт, доступный браузеру. Текст отображается размером в 10 пунктов и красным цветом.

Заметьте, что синтаксис CSS несколько отличается от HTML. <STYLE> и </STYLE> тэги дают команду браузеру прекратить интерпретацию HTML и начать интерпретировать таблицы стилей.

* *<*!-- и --> - это тэги комментариев в HTML. Если браузер старой версии не может интерпретировать <STYLE> и </STYLE> тэги, то он пропустит все, что заключено в комментарии,
* Имена стилей не имеют параметров; CSS «знает», что *.warning -* имя стиля, расположенного в данной позиции.
* Параметры стилей заключаются в <угловые скобки>.
* Параметры стилей отделяются друг от друга точкой с запятой, а не пробелом.
* В качестве оператора присвоения используется двоеточие вместо знака равенства.
* Атрибут CLASS вызывает предопределенный CSS-стиль. Ниже приведен HTML-код, присваивающий все установки класса *warning* абзацу:

<Р CLASS="warning">

Атрибут STYLE определяет CSS атрибуты для ранжирования тэгов. Код, приведенный ниже, дает команду отображать текст в рамке шириной в один пиксел:

<border: 1px solid>

Свойства таблиц каскадных стилей подвергаются многократным переопределениям, прежде чем браузер отобразит текст. На самом верхнем уровне браузер имеет заданные по умолчанию свойства стиля для каждого тэга HTML. Они могут переопределяться в первую очередь свойствами стиля в секции <HEAD>, затем посредством атрибутов, назначенных *контейнерам,* таким как диапазоны, разделы, таблицы и абзацы, затем атрибутами, назначенными отдельным тэгам. Поэтому-то таблицы и называются *каскадными.*

Наследование обеспечивает другой пример каскадирования. Суть в том, что некоторые встроенные селекторы, например как TD, наследуют большинство своих стилевых свойств от других селекторов, таких как Р. И, следовательно, изменение свойств селектора Р повлечет за собой изменение селектора TD.Ссылка.

На приведенных ниже сайтах размещена документация о возможностях и использовании CSS.

**Microsoft:** *http://msdn.microsoft.com/workshop/author/css/reference/attributes.asp*

**World Wide Web Consortium:** *http://www.w3.org/Style/CSS/*

CSS понимает разные единицы измерения для различных устройств: рхдля пикселов, pt для точек, inи cm для дюймов и сантиметров, %для процентов и т. д.

Новшества в управляющих элементах CSS относятся не только к форматированию, но и к позиционированию. Следующие стилевые свойства размещают текст на 50 пикселов от верхнего края страницы и на 100 пикселов от левого края страницы:

<span style="position; absolute; top: 50px; left: 100px;">

И где я теперь?

</span>

**2.3.3. Использование шрифтов на электронных страницах**

В те давние времена, когда принтеры имели всего один шрифт, и тот только с прописными буквами, и выводили текст на перфорированную бумагу стандартного размера, проблем со шрифтами не было. Шрифты на первых персональных компьютерах воспринимались как экзотика, но как только речь зашла о совместимости и переносимости, шрифты стали вечной головной болью Web-дизайнеров и системных администраторов.

Первые текстовые процессоры под Windows позволяли определять шрифты, не инсталлируя их. Потом пользователи возмущались, что шрифты не отображаются. Шрифты стали проклятьем.

Когда Windows стали поддерживать TrueType-шрифты, стало полегче. Соответствующий диалог показываеттолькоте шрифты, которые могут быть отображены, и Windows может отобразить эти шрифты любым размером. CSS предоставляют широкие возможности управления шрифтами. К сожалению, вы никогда не знаете, какие именно шрифты установлены у пользователя, открывшего созданную вами Web-страницу. Круг замкнулся: мы определяем шрифты для ввода, но не для вывода. Шрифты должны размещаться непосредственно на Web-сайтах, чтобы посетители могли загрузить их при необ­ходимости. Тогда не придется конвертировать текст в графику, время загрузки уменьшится, а посетители увидят страницы такими, какими их создал автор. Однажды, я надеюсь, будет найден устраивающий всех способ загрузки шрифтов, так же, как сейчас существует способ загрузки изображений. А пока загляните на следующие Web страницы:

*http://www.microsoft.com/opentype -* домашняя страница Microsoft, посвященная шрифтам в Web. ОреnТуре - совместная инициатива Microsoft и Adobe, цель которой - разработать единый формат загружаемого шрифта, совместимый с PostScript и TrueType стандартами.

*http://www.truedoc.com/ -* домашняя страница TrueDoc, изобретения фирмы Bitstream Inc., основателя цифрового набора. Web-дизайнеры могут конвертировать шрифты на своих компьютерах в TrueDoc-файлы и размещать их на Web-сайтах. Удаленные браузеры загружают TrueDoc-файлы и используют их для отображения страниц с этих сайтов.

Одновременно с созданием TrueDoc-файлов вы определяете имя сайта одним из двух возможных способов; указывая абсолютный адрес *(http://www.whatever.com)* или относительный путь (-/whoever/). Таким образом, использование TrueDoc-файлов для просмотра других сайтов становится невозможным. Bitstream ввела это ограничение для того, чтобы хозяева каждого Web-сайта могли приобретать только те шрифты, которые они используют. К сожалению, если пользователь посещает несколько сайтов, разработанных с использованием одинаковых TrueDoc-шрифтов, шрифты будут загружаться многократно.

Bitstream поддерживает и распространяет несколько TrueDoc-шрифтов свободно, на тот случай, если вы не хотите указывать домашнюю страницу Sitstream как место размещения шрифтов. На их сайте также поддерживается компонент ActiveX, который делает Internet Explorer совместимым с TrueDoc.

*http://www.netscape.com/communicator/version4.0/dynfonts -* описывает то, как Netscape поддерживает загружаемые шрифты. В основном Netscape адаптировал TrueDoc-файлы и встроил их в Navigator 4.0 и выше.

*http://www.hexmac.com/index2.html -* домашняя страница утилиты Typograph, которая конвертирует шрифты в вашей системе в TrueDoc-формат. (Будьте аккуратны, поскольку не со всеми шрифтами разрешены такие преобразования).

**2.3.4. Язык VRML**

Формат HTML был разработан для представления в WWW статической текстовой и графической информации и позволил приблизить внешний вид WWW-страниц к виду хорошо иллюстрированных журналов. Но аппетит приходит во время еды, и у пользователей WWW возникло естественное желание получать информацию в виде подвижной и трехмерной графики, с которой они привыкли работать на локальных компьютерах (например, проводя досуг за компьютерными играми). Появившийся формат VRML (Virtual Reality Modelling Language, Язык моделирования виртуальной реальности) предназначен для представления подвижных объектов и трехмерного моделирования при работе с сервисом WWW.

Основная идея VRML состоит в том, что вместо готовых растровых изображений по сети от сервера к клиенту передается лишь набор инструкций и данных, позволяющий программе-клиенту синтезировать изображение. Это существенно снижает объем трафика и делает реальной передачу информации о подвижных трехмерных объектах.

В VRML входят описания основных графических примитивов, текстур поверхностей, источников освещения, перемещений. VRML позволяет передать анимацию и синхронизованный с изображением звук, а также организовать интерактивное взаимодействие с пользователем.

**2.3.5. Язык XML**

Технология XML представляет собой нечто большее, чем просто способ представления Web-страниц, с помощью XML набор документов превращается в базу данных. Содержимое документа XML располагается между стандартными тэгами; столь строгая структура кода позволяет всем приложениям без труда выбирать и использовать в своих целях это содержимое. Каждый документ XML становится хранилищем данных, к которому можно обращаться с запросами подобно тому, как можно было бы обратиться к любой БД. К сожалению, правила, регламентирующие порядок упаковки данных, на Web-страницах и методы обработки этих, данных до сих пор не определены и не систематизированы. Результат? Интернет сегодня представляет собой беспорядочную и причудливую смесь технологий HTML, JavaScript и Java на клиентских системах и крайне широкий набор компилируемых языков и языков сценариев на стороне сервера.

XML позволяет разобраться в этих нагромождениях и, упорядочив хаос, организовать его в единую унифицированную сеть. Данные, которые ранее были бессистемно разбросаны по страницам HTML, теперь размещаются на строго структурированных документах XML. Браузеры пятого поколения, выпущенные компаниями Netscape Communications и Microsoft, поддерживают спецификации XML и способны обрабатывать информацию гораздо эффективнее по сравнению со своими предшественниками, которые манипулировали лишь конструкциями HTML.

Эти данные правильно интерпретируются не только браузерами, но и другими XML-совместимыми приложениями. Новое поколение служб электронного обмена данными (EDI) способно связать при помощи средств XML различные бизнес-процедуры, определить соответствующие API-интерфейсы и форматы сообщений.

Некоторым же она представляется упрощенной версией доброго старого стандартного языка обобщенной разметки SGML, который, собственно, и положил начало HTML. И хотя и в том и в другом утверждении есть доля истины, XML нельзя считать обычным представителем когорты языков гипертекстовой разметки. Данная технология быстро превращается в основную движущую силу развития объектно-ориентированной Паутины.

Строгая дисциплина управления наборами документов лежит в основе большинства операций, требующих особой точности.

Хотя язык HTML создан на основе спецификаций SGML DTD, браузеры никогда не отличались их однозначной интерпретацией. Впрочем, на начальном этапе развития Интернет эту неопределенность можно было считать достоинством. Web-страницы не ограничивали полета творческой фантазии, и любой пользователь мог погрузиться в игру с Web. Но сейчас, когда HTML получил статус официального языка Сети, подобные вольности уже недопустимы.

XML придает технологии SGML дополнительную строгость и точность, не ограничивая возможность манипулирования огромным количеством HTML-страниц, накопленных в Интернет к сегодняшнему дню. Этого удалось добиться за счет упрощения правил определения DTD(спецификации определений типов документов -Document Type Definitions). Таким образом, чтобы добиться совместимости миллиардов уже размещенных в Интернет страниц HTML со спецификациями XML, достаточно приложить минимум усилий.

Вот, например, типичный фрагмент HTML:

**<img src=/img/fig1.jpg>**

Эквивалентная конструкция на языке XML будет выглядеть так:

**<img src="/img/fig1.jpg"/>**

Внесение небольших изменений превращает конструкции HTML в код XML. Заключение атрибута /img/fig1.jpg, представляющего собой ссылку на графический файл fig1.jpg, в кавычки и добавление в конце косой черты позволяет избежать двусмысленности при автоматическом синтаксическом анализе операторов XML. Страницу XML (так же, как и весь документ) можно считать базой данных, поскольку каждое идентифицируемое в процессе синтаксического анализа поле содержит специфическую информацию, которая распознается, обрабатывается и преобразуется в нужный вид другими приложениями.

Интернет Explorer 5.0 способен преобразовать страницу XML в объект, который можно непосредственно обрабатывать средствами таблиц стилей Extensible Stylesheet Language (XSL). Допускается также косвенная обработка за счет извлечения нужных фрагментов страницы при помощи сценариев Microsoft VBScript или ECMAscript и последующего встраивания их в модель Document Object Model браузера. Браузер Netscape версии 5.0 обладает аналогичными возможностями.

Однако браузер решает лишь часть задач. Web-узлы обслуживают массу других бизнес-процедур: с их помощью отслеживается доставка экспресс-отправлений, закупаются товары для офиса, проводятся операции с ценными бумагами. Появляются все новые и новые задачи; некоторые из них выполняются без всякого вмешательства человека.

Со временем XML найдет себе применение при определении служебных API-интерфейсов, предназначенных для интеграции информационных систем деловых партнеров.

На Web-узле английской компании Harvey Bowring Online, специализирующейся на страховании кредитов, используют инструментарий GlobalAccess, разработанный компанией D&B. На каждом этапе работы узла, как данные, так и протоколы запросов и ответов представляются в терминах XML. Это означает следующее. Система может работать где угодно. В технологии электронного обмена данными (EDI) для D&B нет ничего нового. Компания применяет ее уже в течение многих лет. Однако как D&B, так и ее клиенты не могут и, вероятно, не смогут получить глобальный доступ к сетям EDI.

Для обращения к данным применяется протокол HTTP. Поэтому никаких осложнений с межсетевыми экранами в этом случае не возникает.

Для доступа к данным браузер и приложения используют одни и те же унифицированные технологии. Гарантией совместимости является управление протоколами средствами XML DTD.

Интероперабельность. Несмотря на то, что система объединяет сервер WebMethods на узле D&B и набор инструментов D&B на узле Bowring, все ее составные части могут взаимодействовать и с другими средствами, поддерживающими XML.

Для развития объектно-ориентированной WWW предполагалось использовать технологии DCOM, CORBA и Интернет Inter-ORB Protocol. Однако им не удалось справиться с тем, что, оказалось, по силам XML - с его помощью XML можно не только проектировать Web-страницы, но и решать гораздо более сложные задачи.

**2.3.6. JavaScript**

Для сложных Web-страниц вам понадобится;

* отображать изменяющиеся данные, такие как текущее время пли дата;
* программировать переменное содержание в зависимости от даты, браузера пользователя или других условий;
* изменять внешний вид элементов страницы, если пользователь щелкнул мышью или провел курсор мыши над элементом.

Все, что связано с обработкой прерываний, требует программирования. Для простоты вы вставляете кусочки своих программ в последовательность HTML и CSS-кодов. Вот так:

<script language=”JavaScripts”>

<!-- document.write("URL=" + document.location); *//-->*

</script>

Тэг <SCRIPT> сообщает браузеру, что внутри HTML размещен код JavaScript. Тэг </SCRIPT> отменяет действие. <!-- и --> тэги сообщают браузерам, которые не могут интерпретировать <SCRIPT> и </SCRIPT>, что строки кода следует рассматривать как комментарии. Двойная косая черта (//) перед тэгом --> - знак комментария в языке JavaScript; без него JavaScript интерпретирует --> как ошибочный оператор.

При строгой типизации данных каждая переменная имеет один и только один тип данных, который не может быть изменен. Объявленное целым останется целым всегда, объявленное строкой останется строкой всегда. При *слабой* типизации данных, как в JavaScript, вы можете поместить любое значение в любую переменную, и переменная примет требуемый тип данных.

*Document.write -* метод, который прописывает HTML в Web-страницу как при программировании вручную. Прописанная «вручную» строка *URL=* дополняется содержанием переменной document.location (перемен­ная класса документ). В данном случае это адрес текущей Web-страницы.

Каждый JavaScript-оператор должен заканчиваться точкой с запятой. Отсутствие точки с запятой JavaScript считает ошибкой. Логические выражения должны быть заключены в круглые скобки, а блоки операторов -в фигурные скобки. Например, оператор *if* будет выглядеть так.

If (navigator.platform == “Win32”) {

dоcument.write("Windows");

winplat = "y";

}else {

document.write ("Other");

winplat = "n";

}

Этот фрагмент кода выполняет строки 2 и 3, если свойство *platform* объекта *navigator* равно Win32, и выполняет строки 5 и 6 в противном случае. JavaScript использует знак == как логический оператор эквивалентности и знак = в качестве оператора присваивания. Попытка сравнения с использованием знака равенства ( = ) - вторая причина ошибок в JavaScript.

Заметьте, что в третьей строке переменная *winplat* определяется при первом появлении. Если вы перепутаете имя переменной, никаких сообщений об ошибке не последует. JavaScript просто создаст новую переменную. Это третий источник ошибок в JavaScript, поскольку этот язык чувствителен к регистру. Для JavaScript переменные Cool и cool - разные.

Поскольку в JavaScript отсутствует строгое определение типов, переменные принимают тип в зависимости от того, что им присваивается. Если вы храните в переменной число, вы можете использовать переменную как числовую или как строковую. Величины, содержащие нечисловые сим­волы, могут использоваться только как строки.

Для создания объектов более сложных, чем простые переменные, JavaScript требует использования такого оператора:

curDt = new Date();

Создаваемый объект одновременно инициализируется текущей датой. Такой объект наследует множество полезных методов. Например, ниже показано, как использование метода *getDay* объекта *curDt* возвращает день недели:

curDay = curDt.getDay ()

На этих сайтах размещена полная документация по JavaScript

Microsoft: *http://msdn.microsoft.com/library/default.asp?url=/library/en-us/script56/html/js56jsoriJScript.asp*

Netscape: *http://developer.netscape.сот/library/documentation/communicator*

**2.3.7. VBScript**

В ответ на появление JavaScript Microsoft выпустила версию своего популярного языка программирования Visual Basic. В целом VBScript делает то же самое, что JavaScript, только программный код очень похож на Visual Basic. VBScript отличается от JavaScript следующим:

* в конце оператора не ставится точка с запятой;
* знак равенства используется как для присвоения значений переменным, так и для выполнения операции сравнения;
* блоки операторов выделяются не фигурными скобками, а парами ключевых слов *if...endif, do…loop*, и *while...wend.*

Так же как в JavaScript, переменные вводятся по мере необходимости, а их тип определяется по контексту. VBScript поддерживает почти все встроенные функции Visual Basic. На Web-страницах VBScript выглядит следующим образом:

<script language="VBScript">

<!--

document.write("URL=" + document.location)

-->

</script>

За подробной информацией о VBScript обратитесь на сайт *http://msdn.microsoft.com/library/default.asp?url=/library/en-us/script56/html/vtoriVBScript.asp*

Тэги <SCRIPT> и </SCRIPT> практически такие же, как и раньше, а язык определяется как VBScript. Тэги <!-- и --> – те же самые, но без знака комментариев //.

VBScript поддерживает *document.write,* метод в точности так же, как и JavaScript, но не требует ставить точку с запятой в конце оператора.

Основной недостаток VBScript состоит в том, что его поддерживает только Internet Explorer. Это уничтожает всякую надежду использовать одну и ту же страницу как для пользователей Netscape Navigator, так и для пользователей Internet Explorer. По этой причине использование VBScript для программирования на стороне клиента ограничено.

**2.3.8. Динамический HTML**

До появления версий 4.0 Internet Explorer и Netscape Navigator сценарии могли изменять содержание и внешний вид страниц, только используя метод *write.document* при загрузке страницы.

Любой сценарий, выполняемый позже, может посылать команды браузеру, такие как запрос на загрузку новой страницы, вывод окна предупреждения или изменение значения поля формы, но как только страница загружена, сценарий не может изменить HTML-код. Как же сделать Web-страницы интерактивными, если их невозможно изменить в ответ на действия пользователя?

Динамический HTML, частично реализованный в Netscape 4 и в большей степени в Internet Explorer 4, устраняет эти ограничения. С использованием динамического HTML сценарии могут вставлять блоки HTML, удалять и заменять их или изменять свойства объектов после отображения страницы на экране. Браузер автоматически обновляет новые свойства и (или) новый HTML-код.

Динамический HTML строится на двух принципах.

* Объектная модель документа (DOM) описывает способ организациии названия объектов в браузере, а также определяет, какие объекты и свойства могут быть изменены и какие значения они могут принимать.
* Событийная модель описывает способ передачи управления сценариям - какие действия посылают сценарии на выполнение.

Слабость динамического HTML состоит в том, что Internet Explorer и Netscape Navigator поддерживают очень разные модели документов и модели событий. Отчасти это следствие «войны браузеров», а отчасти связано с тем, что браузеры разрабатывались в разное время. В любом случае окончательным результатом следует считать разработку страниц с динамическим HTML, совместимых и с IE, и с Netscape.

За информацией о динамическом HTML обратитесь на сайт Microsoft: http://msdn.microsoft.com/workshop/author/dhtml/reference/dhtml\_reference\_entry.asp

Netscape: http://developer.netscape.com/library/documentalion/communicator/dynhtml/index.htm

В Internet Explorer почти все объекты на странице - ссылки, изображения, параграфы текста и т.д. могут реагировать на широкий спектр событий. Netscape Navigator поддерживает намного меньше событий, и при этом только для ссылок и форм.

**2.3.9. Java-апплеты**

Практически все современные браузеры способны отображать и выполнять Java-апплеты — специальные Java-приложения, которые пользователь получает в составе Web-страницы. Эти приложения нередко включаются в состав Web-страниц с целью добавления функциональности, которую сложно или невозможно реализовать с помощью скриптовых языков. Апплеты могут выполняться на всех платформах, для которых доступна виртуальная Java-машина.

Апплеты обычно создаются в соответствии с правилами, оговаривающими период их жизни и способы взаимодействия со своим окружением. Чаще всего эти способы весьма ограниченны (например, такие операции, как считывание и запись файлов, по умолчанию для апплетов запрещены; если же подобные операции необходимы, разрешения на их выполнение для конкретных апплетов и конкретных файлов описываются на клиентском компьютере; сетевой доступ из апплета возможен только к тому компьютеру, с которого он был загружен; запуск других приложений на компьютере пользователя из апплетов невозможен). Однако апплет способен считывать значения параметров (например, цвета, шрифтов, файлов с графическими изображениями, используемыми при выполнении апплета) с содержащей его Web-страницы и в соответствии с этими параметрами изменять свое поведение. Кроме того, параметры апплета можно менять динамически из кода на скриптовых языках, содержащихся в составе той же страницы.

Отметим, что, поскольку апплеты реализуют выполнение кода на компьютере клиента, они в определенной степени являются потенциально опасным содержимым. Именно поэтому все современные браузеры обладают доступными пользователю средствами ограничения возможностей выполнения апплетов.

**2.3.10. Элементы управления ActiveX**

Некоторые из современных браузеров (в частности, Microsoft Интернет Explorer) могут служить контейнерами для элементов управления ActiveX — специальных COM-серверов, выполняющихся в адресном пространстве браузера и также получаемых в составе Web-страницы.

С помощью элементов управления ActiveX, как и посредством Java-апплетов, можно реализовать любую функциональность, в том числе и неблагоприятную для компьютера пользователя, при этом, в отличие от Java-апплетов, при выполнении элементов управления ActiveX в общем случае нет никаких ограничений на доступ к файлам и иным ресурсам операционной системы и сети, а код, содержащийся в них, выполняется от имени загрузившего их пользователя. Как и Java-апплеты, элементы управления ActiveX могут считывать свои свойства с содержащей их страницы; кроме того, свойства элемента управления ActiveX можно менять динамически из кода на скриптовых языках, содержащихся в составе той же страницы; в том же коде можно обрабатывать события, возникающие в таких элементах управления.

Естественно, Microsoft Интернет Explorer обладает средствами ограничения возможностей выполнения элементов управления ActiveX, в том числе управления ими из кода на скриптовых языках. Однако для контроля безопасности их выполнения имеется еще одно средство, называемое электронной цифровой подписью. Цифровая подпись помещается внутрь элемента управления ActiveX, для чего требуется наличие соответствующего электронного сертификата. Электронная подпись, помимо сведений о фирме-производителе, содержит и другую полезную информацию. Так, например, если файл с элементом управления ActiveX после добавления электронной подписи был изменен, то об этом будет немедленно сообщено перед запуском такого элемента управления — при добавлении подписи к элементу управления ActiveX происходит вычисление контрольной суммы соответствующего файла. Отметим, однако, что в России в настоящее время нет авторизованных компаний, которые могли бы выдать электронный сертификат международного образца. Естественно, наличие электронного сертификата не гарантирует отсутствия потенциально опасного содержимого, но, по крайней мере, позволяет клиенту установить его источник.

Далее нам следует напомнить банальную истину, которая, как показывает практика, очевидна не для всех наших читателей. При работе с элементами управления ActiveX и Java-апплетами абсолютно бесполезно полагаться на антивирусное программное обеспечение (неважно, клиентское оно или серверное): признаков, характерных для вирусов (таких как способность внедряться внутрь исполняемых файлов и документов), подобные приложения, как правило, не содержат. Можно лишь запретить загрузку или выполнение соответствующего кода либо на уровне настроек браузера, либо на уровне корпоративных или персональных брандмауэров.

**2.3.11. Приложения Macromedia Flash**

Приложения Macromedia Flash являются сегодня наиболее популярным расширением функциональности Web-браузеров — с их помощью многие Web-дизайнеры придают своим сайтам интерактивность и оригинальность.

Модель безопасности приложений Flash основана на том, что Macromedia Flash Player, как и виртуальная Java-машина, выполняет приложения в ограниченном адресном пространстве, при этом выполняемые приложения не имеют доступа к файловой системе (кроме одного конкретного каталога, используемого Macromedia Flash Player для служебных целей) и другим ресурсам компьютера пользователя; исключение делается для микрофонов и видеокамер, однако пользователь должен дать разрешение на передачу данных, полученных с этих устройств. Доступ к сетевым ресурсам ограничивается доменом, с которого было получено приложение. Отметим, что приложения Flash также могут управляться с помощью кода JavaScript, присутствующего на той же странице. Сам Macromedia Flash Player для Microsoft Интернет Explorer является элементом управления ActiveX и использует возможности элементов управления ActiveX для доступа к свойствам приложений Flash из скриптовых языков.

Отметим, что помимо вышеперечисленных наиболее популярных средств расширения функциональности браузеров имеется и ряд других средств, реализованных обычно в виде так называемых модулей расширения (plug-in). Поскольку модули расширения также представляют собой исполняемый код, современные браузеры обладают средствами ограничения возможностей, связанных с их загрузкой и выполнением.

В заключение отметим, что перечисленные средства расширения функциональности HTML-страниц могут быть использованы и в динамических страницах, генерируемых серверными Web-приложениями. Так, в последнее время широкое распространение приобрели средства создания Web-приложений, выполняющихся под управлением Web-серверов и генерирующих динамические HTML-страницы с внедренным в них кодом на скриптовых языках, предназначенным для интерпретации браузером.

### 2.4. Технологии создания серверных частей Web-приложений

Cценарии для сервера позволяют получить доступ к файлам, базам данных и другим ресурсам, хранимым на сервере, а также к централизованным ресурсам сервера, таким как электронная почта или факс-служба.

Функционирование в непротиворечивой и управляемой среде - еще одно преимущество выполнения сценариев на сервере. Написанный код выполняется только на одной версии единственного сервера, а не на множестве версий множества браузеров.

Для использования сценариев на стороне сервера имеется три основных препятствия.

* Запуск скриптов на сервере зачастую требует получения специальных прав от Web-мастера или системного администратора.
* Для взаимодействия со сценариями, выполняющимися на сервере, пользователь должен щелкнуть мышкой на ссылке- или на кнопке на странице, а затем ожидать, когда сервер выполнит сценарий и перешлет ответ. Взаимодействие с использованием динамического HTML происходит быстрее.
* Для тестирования сценариев для сервера требуется иметь собственный WWW-сервер, предпочтительно того же типа, что и промышленный вариант.

По этой причине большинство разработчиков предпочитают программирование браузера.

**2.4.1. Активные серверные страницы - ASP**

Internet Information Server (IIS), WWW-сервер фирмы Microsoft, поддерживает версию VBScript для сервера, которая называется Active Server Pages (ASP). Она позволяет совмещать блоки кода, написанного на VBScript, и HTML. Каждый блок VBScript-кода выполняется на сервере, не передается пользователю и заменяется сгенерированным на сервере HTML-кодом.

ASP поддерживает встроенные объекты для входящих запросов, исходящих ответов, пользовательского сеанса, приложения и собственно сервера.

* Объект запрос хранит информацию о браузере клиента, переданной строке URL, полях форм и т. д.
* Объект *ответ* получает исходный HTML-код, коды статуса и некоторые команды. С помощью кодов статуса, например, можно заставить пользователя ввести имя пользователя и пароль. Команды могут отослать пользователя к другому URL.
* Объект *сервер* содержит информацию о сервере: его имя, версию программного обеспечения, состав программного обеспечения и так далее.
* Объект *сеанс* сохраняет информацию о каждом пользователе для дальнейшего взаимодействия. Пользователь идентифицируются по блоку данных, которые сервер посылает браузеру посетителя и ожидает такой же блок при следующем запросе. (Такой маленькитй блок идентификационной информации называется *cookie).*
* Объект приложение хранит информацию, общую для всех пользователей данного приложения. (Приложение - это все ASP-страницы в данном дереве каталогов).

Информацию по ASP вы можете найти в разделе «описание сценариев» на сайте

*http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnanchor/html/anch\_iis.asp*

Новейшей версией технологии Active Server Pages является ASP .NET, ключевая в архитектуре Microsoft .NET Framework. Основное отличие этой технологии от ASP с точки зрения архитектуры приложений заключается в том, что код, присутствующий на Web-странице, не интерпретируется, а компилируется и кэшируется, что, естественно, способствует повышению производительности приложений.

С помощью ASP.NET можно создавать Web-приложения и Web-сервисы, которые не только позволяют реализовать динамическую генерацию HTML-страниц, но и интегрируются с серверными компонентами и могут использоваться для решения широкого круга бизнес-задач, возникающих перед разработчиками современных Web-приложений.

В общем случае клиентом Web-сервера может быть не только персональный компьютер, оснащенный обычными Web-клиентами (например, Web-браузером), но и мобильные устройства, отличающиеся ограниченным размером экрана, малым объемом памяти, а нередко и невозможностью отображения графики. Для этих устройств существуют свои протоколы передачи данных (Wireless Access Protocol, WAP) и соответствующие языки разметки (WML, Wireless MarkupLanguage, СHTML, Compact HTML и т.п.). При этом необходимо передавать данные на мобильное устройство в соответствующем формате, для чего нередко создаются специальные сайты (например, поддерживающие WAP и WML). Более удобным представляется создание приложений, которые способны генерировать тот или иной код в зависимости от типа клиента. Именно такой подход и реализован в Microsoft ASP .NET.

**2.4.2. Perl**

Название языка расшифровывается как Practical Extraction and Report Language. По легенде, Ларри Вол изобрел Perl, когда писал программу для составления отчетов. Perl попадает под действие лицензии на свободно распространяемое программное обеспечение и постоянно развивается.

На Web-сервере Perl используется для запуска CGI-программ. Когда Web-сервер передает URL в Perl-программу, то он автоматически запускает Perl-интерпретатор, сообщает, какая программа должна быть запущена, и получает информацию о входящем запросе одним из двух способов.

* Через *стандартный ввод -* то есть через перенаправление ввода в командную строку MS DOS.
* Через *переменные* окружения - то есть через те элементы, которые можно видеть после ввода команды SET в командной строке DOS.

**Perl**-программы выводят HTML в стандартный вывод, аналогично тому как MS DOS-программы выводят информацию на экран. Web-сервер перенаправляет этот вывод удаленному пользователю.

**Perl,** так же как JavaScript и VBScript, имеет контекстно-определяемые переменные, которые преобразуются автоматически из числовых в строковые. Имена простых переменных начинаются с символа $, имена массивов начинаются с символа @.

Как и в JavaScript, операторы Perl должны заканчиваться точкой с запятой. Фигурные скобки определяют блоки операторов. Знак равенства (=) используется при присвоении значений переменным, а для сравнения величин следует использовать «==» или .eq.. При этом, если $а == $b означает сравнение чисел, то $а eq $b означает сравнение строк. Оператор <= предназначен для сравнения чисел, а "le" - сравнение символов. Перепутать эти операторы проще, чем забыть поставить точку с запятой в конце оператора,

Perl-программы часто требуют небольших изменений при перенесении с одной операционной системы на другую. По управлением UNIX-систем, к примеру, в первой строке определяется местоположение интерпретатора языка:

#!/usr/bin/perl

Конкретное место может меняться от системы к системе.

В Windows NT начальная строка совсем не требуется. Вместо этого ваш системный администратор должен установить связь между интерпретатором *Perl.exe* и расширением файла *.pl* в конфигурации Web-сервера.

**2.4.3. CGI**

Common Gateway Interface (CGI) является стандартным интерфейсом, который позволяет выполнять серверные приложения, вызываемые через URL. Входной информацией для таких приложений являются данные HTTP-заголовка либо тело запроса, в зависимости от применяемого протокола. CGI-приложение генерирует HTML-код, который возвращается браузеру. Не так давно широко использовался и термин «CGI-скрипт», происхождение которого объясняется тем, что подобные приложения писались на скриптовых языках типа Perl, выполняющихся, тем не менее, не в браузере, а на сервере. CGI-приложения создают с помощью любых средств разработки, генерирующих консольные приложения для операционной системы, под управлением которой функционирует Web-сервер.

Главная проблема любого CGI-приложения заключается в том, что при каждом клиентском запросе сервер загружает это приложение в отдельное адресное пространство, а затем осуществляет его выполнение и выгрузку. Из-за этой особенности ограничивается производительность приложений и возможность одновременной обработки большого количества клиентских запросов.