Система верстки TeX (LaTeX).

TeX — это система компьютерной вёрстки, предназначенная для набора научно-техническихтекстов высокого полиграфического качества. Пользователь задает текст и его структуру с помощью определенных команд на языке низкоуровневой разметки, а TeX сам форматирует документ.

LaTeX — наиболее популярный набор макрорасширений TeXa. Главная идея LaTeX состоит в том, что авторы должны думать о содержании, не беспокоясь о конечном визуальном облике. Готовя свой документ, автор указывает логическую структуру текста (разбивая его на главы, разделы, таблицы, изображения), а LaTeX решает вопросы его отображения. Это и является основным отличем TeX от MS Word. В первом случае используется парадигма WYSIWYM (What You See Is What You Mean), а во втором — WYSIWYG (What You See Is What You Get).

Преимущества LaTeX:

- Высокое качество подготовки научных текстов, работы с формулами;
- Автоматическая рубрикация документа, нумерация формул, рисунков и списка литературы;
- Пользователь сосредоточится на структуре, а не оформлении;
- Внесение изменений в визуальное представление документа не требует изменения самого документа.

Недостатки LaTeX:

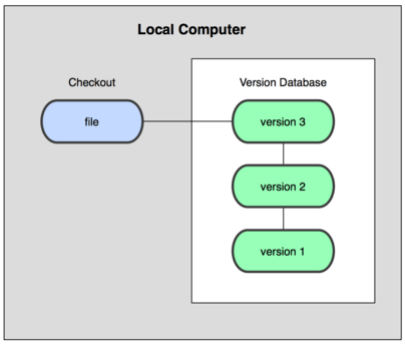
- Для работы с данным средством требуется знать язык разметки;
- Затуднена работа с рисунками;
- В MS Word более простой и понятный интерфейс: что мы видим, то и получаем.

Системы контроля версий.

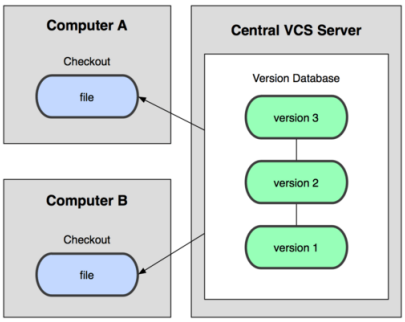
GitHub — крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки. Основан на системе контроля версий Git.

Git — распределённая система управления версиями, обладающая всеми преимуществами РСКВ.

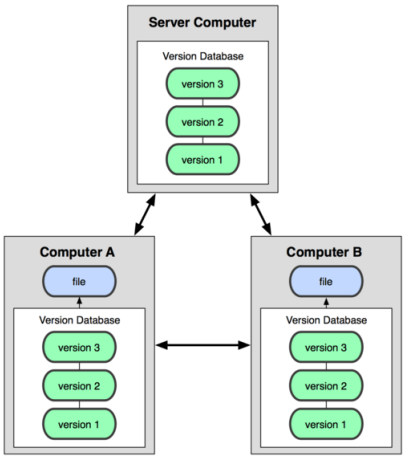
Существуют локальные, централизованные и распределенные системы контроля версий. **Локальные СКВ** содержат простую базу данных, в которой хранятся все изменения нужных файлов. Эта утилита основана на работе с наборами патчей между парами версий. Но данный подход не позволялет сотрудничать с разработчиками за другими компьютерами, поэтому были созданы **централизованные СКВ**.



Такие системы подразумевают центральный сервер, на котором хранятся все файлы. Центральный сервер — уязвимое место для всей системы. Есть риск потерять все данные, если поражается диск с центральной базой.



Решением этой проблемы являются **распределенные СКВ**. В этих системах существуют удаленные и локальные репозитории. Таким образом, клиенты не просто выгружают последние версии файлов, а копируют весь репозиторий, поэтому в случае отказа сервера база данных может быть восстановлена путем копирования клиентского репозитория.



Ядро Git представляет собой набор утилит командной строки с параметрами. Репозиторий Git представляет собой каталог файловой системы, в котором находятся файлы конфигурации репозитория, файлы журналов, хранящие операции, выполняемые над репозиторием, индекс, описывающий расположение файлов и хранилище, содержащее собственно файлы. По умолчанию репозиторий хранится в подкаталоге с названием «.git» в корневом каталоге рабочей копии дерева файлов, хранящегося в репозитории. Любое файловое дерево в системе можно превратить в репозиторий git, отдав команду создания репозитория из корневого каталога этого дерева.

Транспортные протоколы UDP и TCP.

Протоколы TCP и UDP относятся к транспортному уровню стека протоколов TCP/IP. Основная функция этих протоколов – передача данных между прикладными процессами, выполняющимися на узлах сети.

UDP

Протокол UDP позволяет приложениям отправлять датаграммы без установления соединений. Полезен в ситуациях, когда не требуется обеспечение надежности доставки данных.

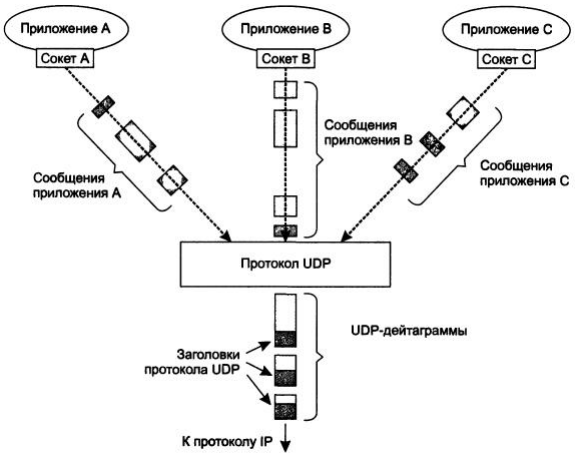
Заголовок UDP состоит из четырех полей по два байта:

- поле порта источника (source port);
- поле порта пункта назначения (destination port);
- поле длины (length);
- поле контрольной суммы UDP (checksum UDP).

Протокол UDP ведет для каждого порта две очереди: очередь пакетов, поступающих в данный порт из сети, и очередь пакетов, отправляемых данным портом в сеть. Процедура обслуживания протоколом UDP запросов, поступающих от нескольких различных прикладных сервисов, называется мультиплексированием.

Распределение протоколом UDP поступающих от сетевого уровня пакетов между набором высокоуровневых сервисов, идентифицированных номерами портов, называется демультиплексированием. На хосте-получателе протокол UDP принимает от протокола IP извлеченные из пакетов датаграммы. Полученные из IP-заголовка IP-адрес назначения и из UDP-заголовка номер порта используются для формирования UDP-сокета, од-

нозначно идентифицирующего приложение, которому направлены данные.

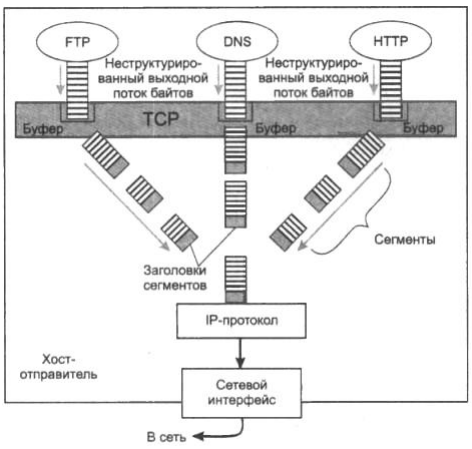


Датаграммы, передаваемые с помощью UDP, могут быть утеряны, дублированы, прийти не по порядку. Но простота этого протокола обеспечивает его быстродействие и компактность. Это широко применяется в клиент-серверных приложениях, когда клиент посылает запрос серверу и, если запрос или ответ теряется, пытается ещё раз через короткий промежуток времени.

TCP

Протокол TCP обеспечивает надежную транспортировку данных между прикладными процессами путем установления логического соединения.

На хосте-отправителе протокол TCP рассматривает информацию, поступающую к нему, как неструктурированный поток байтов. Поступающие данные буферизуются, из буфера вырезается часть данных, которая называется сегментом и снабжается заголовком.



Заголовок TCP имеет размер 20 байт и содержит:

- номер порта источника;
- номер порта назначения;
- последовательный номер - номер первого байта данных в сегменте;
- номер подтверждения - следующий ожидаемый последовательный номер;
- смещение данных - размер заголовка;
- резерв - резервируются для будущего использования;
- флаги - ACK (квитанция на принятый сегмент), URG, RST, SYN (первоначальное значение в поле номер последовательности) и т. д.
- размер окна -содержит количество байт, которое может быть послано

после байта, получение которого уже подтверждено;

- контрольная сумма - по ней определяется, был ли искажен пакет;
- указатель срочности;
- опции - необязательные значения, которые указываются при необходимости;
- заполнитель - здесь добавляется столько нулей, сколько необходимо, чтобы заголовок имел длину 32-бита.

Благодаря логическому соединению TCP следит, чтобы передаваемые сегменты не были потеряны, не были продублированы и пришли к получателю в том порядке, в котором были отправлены. При установлении соединения обе стороны передают друг другу следующие параметры: максимальный размер сегмента, размер окна, начальный порядковый номер байта, с которого она начинает отсчет потока байтов. Для установлении соединения также используются флаги ACK, SYN, RST и т. д.

В протоколе TCP правильность передачи каждого сегмента подверждается квитанцией от получателя. Для организации квитирования используется алгоритм скользящего окна. В данном методе источнику разрешается передавать определенное количество кадров до получения квитанции. Это количество называется размером окна.

При отправке кадра устанавливается тайм-аут. Если за установленное время квитанция на отправленный кадр не придет, то кадр считается утерянным и передается снова.

Использованные источники:

- http://www.chin28.narod.ru/d10.1.htm

- http://www.vanderboot.ru/tcp-ip/tcp-udp.php
- http://cdo.bseu.by/library/ibs1/net_l/tcp_ip/transp/tcp.htm
- http://iptcp.net/povtornaya-peredacha-i-skolzyashchee-okno.html

Сравнение производительности UDP и TCP при различных параметрах.

В таблице ниже приведены время, скорость и среднее количество потерянных пакетов для сравнения показателей UDP и TCP. В качестве передаваемых данных был использован файл размером 50 MB. Передача осуществлялась блоками по 1440 байт.

Протокол	Время, с	Скорость, Мбит/с	Среднее кол-во потерянных пакетов
UDP	$10,54 \pm 1,03$	38,09	182 из 35745
TCP	$13,30 \pm 0,34$	30,08	

На основании этой таблицы можно сказать, что скорость передачи по протоколу UDP выше, чем TCP. Но хотя UDP и выигрывает в скорости, мы теряем часть данных.

Был произведен ряд тестов при использовании файлов различных размеров, во всех тестах данные передавались блоками по 64000 байт.

Размер файла 10 MB.

Протокол	Время, с	Скорость, Мбит/с	Среднее кол-во потерянных пакетов
UDP	$0,06 \pm 0,01$	1333	0 из 161
TCP	$0,27 \pm 0,07$	296	

Размер файла 50 MB.

Протокол	Время, с	Скорость, Мбит/с	Среднее кол-во потерянных пакетов
UDP	$0,29 \pm 0,04$	1379	6 из 805
TCP	$1,01 \pm 0,18$	396	

Размер файла 80 MB.

Протокол	Время, с	Скорость, Мбит/с	Среднее кол-во потерянных пакетов
UDP	$0,41 \pm 0,06$	1561	52 из 1281
TCP	$0,31 \pm 0,01$	2065	

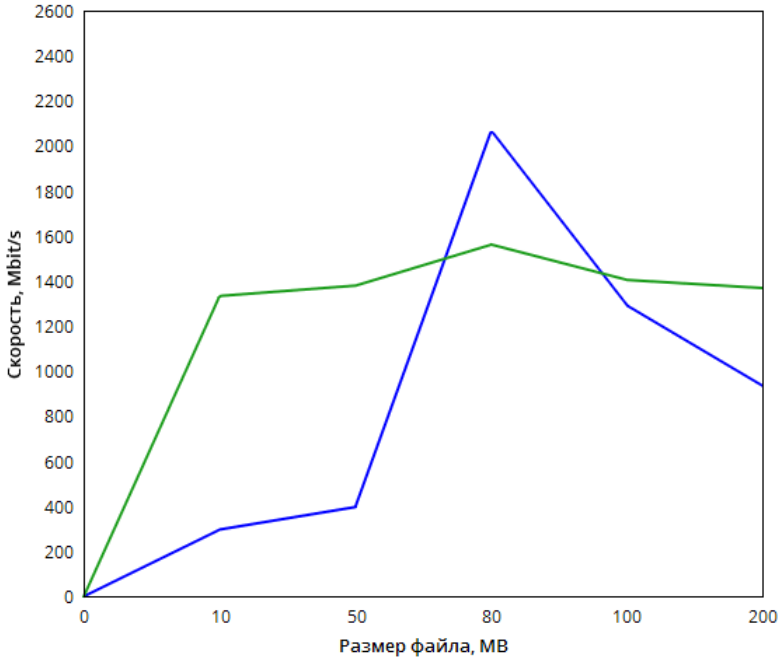
Размер файла 100 MB.

Протокол	Время, с	Скорость, Мбит/с	Среднее кол-во потерянных пакетов
UDP	$0,57 \pm 0,13$	1404	154 из 1609
TCP	$0,62 \pm 0,13$	1290	

Размер файла 200 MB.

Протокол	Время, с	Скорость, Мбит	Среднее кол-во потерянных пакетов
UDP	$1,17 \pm 0,14$	1368	1015 из 3218
TCP	$1,78 \pm 0,08$	933	

График зависимости скорости передачи данных от объема, построенный по данным таблиц.



По графику можно понять, что скорость передачи UDP не всегда выше скорости TCP. К тому же по таблицам видно, что чем больше размер файла, тем больше данных мы теряем, передавая их по протоколу UDP, в то время как TCP надежно доставляет все данные.

Можно сделать вывод, что при передаче небольших файлов скорость UDP значительно выше, чем TCP, при этом потери данных невелеки или же совсем отсутсвуют. Таким образом, при передаче маленьких файлов гораздо выгоднее использовать UDP. Для передачи же больших файлов следует использовать TCP, чтобы не потерять большой объем данных, при этом есть вероятность, что и скорость будет выше.