**Часть 1. Ответить на следующие вопросы:**

(Ответы должны быть краткими, но раскрывать суть понятий и ключевые моменты.

Ответ на один вопрос по объему – не менее 2-х страниц (но не более 3-х страниц) формата А4, шрифт -12)

1. Особенности сетевых и распределенных ОС.
2. Сетевые компоненты СОС.
3. Особенности организации файловой системы в СОС.
4. Особенности архитектуры сетевых операционных систем.
5. Особенности управления памятью в СОС.
6. Особенности планирования процессов и нитей в СОС.
7. Организация взаимодействия удаленных процессов.
8. Многоуровневая модель сетевого взаимодействия. Протоколы.Интерфейсы.
9. Способы адресации удаленных процессов. Маршрутизация. Синхронизация удаленных процессов.
10. Цели и задачи сетевого администрирования. Аудит пользователей и ресурсов.

Часть 2.

**Составить по 3 тестовых задания на каждый вопрос.** (Итого 30 тестовых заданий).

Задания оформить на странице А4 в электронном виде.

1. Особенности сетевых и распределенных ОС.

В зависимости от того, какой виртуальный образ создает операционная система, чтобы подменить им реальную аппаратуру компьютерной сети, различают сетевые ОС и распределенные ОС.

Сетевая ОС предоставляет пользователю некую виртуальную вычислительную систему, работать с которой гораздо проще, чем с реальной сетевой аппаратурой. В то же время эта виртуальная система не полностью скрывает распределенную природу своего реального прототипа, т.е. является виртуальной сетью.

При использовании ресурсов компьютеров сети пользователь сетевой ОС всегда помнит, что он имеет дело с сетевыми ресурсами и что для доступа к ним нужно выполнить некоторые особые операции, например, отобразить удаленный разделяемый каталог на вымышленную локальную букву дисковода или поставить перед именем каталога еще и имя компьютера, на котором тот расположен. Пользователи сетевой ОС обычно должны быть в курсе того, где хранятся их файлы, и должны использовать явные команды передачи файлов для перемещения файлов с одной машины на другую.

Работая в среде сетевой ОС, пользователь, хотя и может запустить задание на любой машине компьютерной сети, всегда знает, на какой машине выполняется его задание. По умолчанию пользовательское задание выполняется на той машине, на которой пользователь сделал логический вход. Если же он хочет выполнить задание на другой машине, то ему нужно либо выполнить логический вход в эту машину, либо ввести специальную команду удаленного выполнения, в которой он должен указать информацию, идентифицирующую удаленный компьютер.

Магистральным направлением развития сетевых операционных систем является достижение как можно более высокой степени прозрачности сетевых ресурсов. В идеальном случае сетевая ОС должна представить пользователю сетевые ресурсы в виде ресурсов единой централизованной виртуальной машины. Для такой операционной системы используют специальное название - распределенная ОС или истинно распределенная ОС.

Распределенная ОС, динамически и автоматически распределяя работы по различным машинам системы для обработки, заставляет набор сетевых машин работать как виртуальный унипроцессор. Пользователь распределенной ОС не имеет сведений о том, на какой машине выполняется его работа. Распределенная ОС существует как единая операционная система в масштабах вычислительной системы. Каждый компьютер сети, работающий под управлением распределенной ОС, выполняет часть функций этой глобальной ОС. Распределенная ОС объединяет все компьютеры сети в том смысле, что они работают в тесной кооперации друг с другом для эффективного использования всех ресурсов компьютерной сети.

В настоящее время практически все сетевые операционные системы еще очень далеки от идеала истинной распределенности. Степень автономности каждого компьютера в сети, работающей под управлением сетевой операционной системы, значительно выше по сравнению с компьютерами, работающими под управлением распределенной ОС.

В результате сетевая ОС может рассматриваться как набор операционных систем отдельных компьютеров, составляющих сеть. На разных компьютерах сети могут быть загружены одинаковые или разные ОС.

Например, на всех компьютерах сети может работать одна и та же ОС UNIX. Более реалистичным вариантом является сеть, в которой работают разные ОС, например, часть компьютеров работает под управлением UNIX, часть - под управлением NetWare, а остальные - под управлением Windows NT и Windows 98. Все эти операционные системы функционируют независимо друг от друга в том смысле, что каждая из них принимает независимые решения о создании и завершении своих собственных процессов и управлении локальными ресурсами. Но в любом случае операционные системы компьютеров, работающих в сети, должны включать взаимно согласованный набор коммуникационных протоколов для организации взаимодействия процессов, выполняющихся на разных компьютерах сети, и разделения ресурсов компьютеров между пользователями сети.

Если операционная система отдельного компьютера позволяет ему работать в сети, т.е. предоставлять свои ресурсы в общее пользование и/или потреблять ресурсы других компьютеров сети, то такая операционная система отдельного компьютера также называется сетевой ОС.

Таким образом, термин «сетевая операционная система» используется в двух значениях: во-первых, как совокупность ОС всех компьютеров сети и, во-вторых, как операционная система отдельного компьютера, способного работать в сети. Исходя из этого определения следует, что такие операционные системы, как, например, Windows NT, NetWare, Linux, Solaris, являются сетевыми, поскольку все они обладают средствами, которые позволяют их пользователям работать в сети.

2. Сетевые компоненты СОС.

К основным функциональным компонентам сетевой ОС можно отнести средства управления локальными ресурсами компьютера, которые реализуют все функции ОС автономного компьютера (распределение оперативной памяти между процессами, планирование и диспетчеризацию процессов, управление про­цессорами в мультипроцессорных машинах, управление внешней памятью, взаимодействие с пользователем и т.д.).

Сетевые средства, в свою очередь, можно разделить на три компонента:

* серверная часть ОС – средства предоставления локальных ресурсов и услуг в общее пользование;
* клиентская часть ОС – средства запроса доступа к удаленным ресурсам и услугам;
* транспортные средства ОС, которые совместно с коммуникационной системой обеспечивают передачу сообщений между компьютерами сети.

Упрощенно работа сетевой ОС происходит следующим образом. Предположим, что пользователь компьютера «А» решил разместить свой файл на диске другого компьютера сети - компьютера «В». Для этого он набирает на клавиатуре соответствующую команду и нажимает клавишу <Enter>. Программный модуль ОС, отвечающий за интерфейс с пользователем, принимает эту команду и передает ее клиентской части ОС компьютера «А».

Клиентская часть ОС не может получить непосредственный доступ к ресурсам другого компьютера, в данном случае к дискам и файлам компьютера «В». Она может только «попросить» об этом серверную часть ОС, работающую на том компьютере, которому принадлежат эти ресурсы. Эти «просьбы» выражаются в виде сообщений, передаваемых по сети. Сообщения могут содержать не только команды на выполнение некоторых действий, но и собственно данные, например, содержимое некоторого файла. Управляют передачей сообщений между клиентской и серверной частями по коммуникационной системе сети транспортные средства ОС. Эти средства выполняют такие функции, как формирование сообщений, разбиение сообщения на части (пакеты, кадры), преобразование имен компьютеров в числовые адреса, организация надежной доставки сообщений, определение маршрута в сложной сети и т.д. и т.п. Правила взаимодействия компьютеров при передаче сообщений по сети фиксируются в коммуникационных протоколах, таких как Ethernet, Token Ring, IP, IPX и пр. Чтобы два компьютера смогли обмениваться сообще­ниями по сети, транспортные средства их ОС должны поддерживать некоторый общий набор коммуникационных протоколов. Коммуникационные протоколы переносят сообщения клиентских и серверных частей ОС по сети, не вникая в их содержание.

На компьютере «В», на диск которого пользователь хочет поместить свой файл, должна работать серверная часть ОС, постоянно ожидающая прихода запросов из сети на удаленный доступ к ресурсам этого компьютера. Серверная часть, приняв запрос из сети, обращается к локальному диску и записывает в один из его каталогов указанный файл. Конечно, для выполнения этих действий требуется не одно, а целая серия сообщений, переносящих между компьютерами команды ОС и части передаваемого файла.

Очень удобной и полезной функцией клиентской части ОС является способность отличить запрос к удаленному файлу от запроса к локальному файлу. Если кли­ентская часть ОС умеет это делать, то приложения не должны заботиться о том, с локальным или удаленным файлом они работают, – клиентская программа сама распознает и перенаправляет (redirect) запрос к удаленной машине. Отсюда и название, часто используемое для клиентской части сетевой ОС, - редиректор. Иногда функции распознавания выделяются в отдельный программный модуль, в этом случае редиректором называют не всю клиентскую часть, а только этот модуль.

3. Особенности организации файловой системы в СОС.

Совокупность серверной и клиентской частей ОС, предоставляющих доступ к конкретному типу ресурса компьютера через сеть, называется сетевой службой. В приведенном выше примере клиентская и серверная части ОС, которые совместно обеспечивают доступ через сеть к файловой системе компьютера, образуют файловую службу.

Говорят, что сетевая служба предоставляет пользователям сети некоторый набор услуг. Эти услуги называют также сетевым сервисом (от англоязычного термина «service»). Хотя указанные термины иногда используются как синонимы, следует иметь в виду, что в некоторых случаях различие в значениях этих терминов носит принципиальный характер. Далее в тексте под «службой» мы будем понимать сетевой компонент, который реализует некоторый набор услуг, а под «сервисом» - описание того набора услуг, который предоставляется данной службой. Таким образом, сервис - это интерфейс между потребителем услуг и поставщиком услуг (службой).

Каждая служба связана с определенным типом сетевых ресурсов и/или определенным способом доступа к этим ресурсам. Например, служба печати обеспечивает доступ пользователей сети к разделяемым принтерам сети и предоставляет сервис печати, а почтовая служба предоставляет доступ к информационному ресурсу сети - электронным письмам. Способом доступа к ресурсам отличается, например, служба удаленного доступа - она предоставляет пользователям компьютерной сети доступ ко всем ее ресурсам через коммутируемые телефонные каналы. Для получения удаленного доступа к конкретному ресурсу, например к принтеру, служба удаленного доступа взаимодействует со службой печати. Наиболее важными для пользователей сетевых ОС являются файловая служба и служба печати.

Среди сетевых служб можно выделить такие, которые ориентированы не на простого пользователя, а на администратора. Такие службы используются для организации работы сети. Например, служба Bindery операционной системы Novell NetWare 3.x позволяет администратору вести базу данных о сетевых пользователях компьютера, на котором работает эта ОС. Более прогрессивным является подход с созданием централизованной справочной службы или, по-другому, службы каталогов, которая пред-назначена для ведения базы данных не только обо всех пользователях сети, но и обо всех ее программных и аппаратных компонентах. В качестве примеров службы каталогов часто приводятся NDS компании Novell. Другими примерами сетевых служб, предоставляющих сервис администратору, являются служба мониторинга сети, позволяющая захватывать и анализировать сетевой трафик, служба безопасности, в функции которой может входить, в частности, выполнение процедуры логического входа с проверкой пароля, служба резервного копирования и архивирования.

От того, насколько богатый набор услуг операционная система предлагает конечным пользователям, приложениям и администраторам сети, зависит ее позиция в общем ряду сетевых ОС.

Сетевые службы по своей природе являются клиент-серверными системами. Поскольку при реализации любого сетевого сервиса естественно возникают источник запросов (клиент) и исполнитель запросов (сервер), то и любая сетевая служба содержит в своем составе две несимметричные части - клиентскую и серверную. Сетевая служба может быть представлена в операционной системе либо обеими (клиентской и серверной) частями, либо только одной из них.

Обычно говорят, что сервер предоставляет свои ресурсы клиенту, а клиент ими пользуется. Необходимо отметить, что при предоставлении сетевой службой некоторой услуги используются ресурсы не только сервера, но и клиента. Клиент может затрачивать значительную часть своих ресурсов (дискового пространства, процессорного времени и т.п.) на поддержание работы сетевой службы. Принципиальной разницей между клиентом и сервером является то, что инициатором выполнения работы сетевой службой всегда выступает клиент, а сервер всегда находится в режиме пассивного ожидания запросов.

4. Особенности архитектуры сетевых операционных систем.

Компьютерная сеть – совокупность компьютеров, связанных коммуникационным оборудованием, работающих под управлением сетевой операционной системы. Операционная система компьютерной сети во многом аналогична ОС автономного компьютера – она также представляет собой комплекс взаимосвязанных программ, который обеспечивает диалог пользователя при работе в сети и реализует эффективный способ разделения ресурсов сети между всеми выполняемыми процессами в сети. Таким образом, сетевая операционная системы выполняет те же функции, что и обычная ОС и обеспечивает работу компьютерной сети, а именно:

- управляет передачей данных в сети;

- определяет способ хранения и методы доступа к файлам на сервере;

- управляет взаимодействием между сетью и подключенным к ней оборудованием;

- обеспечивает безопасный доступ к данным;

- поддерживает связь между несколькими сетями;

В результате, сетевая ОС может рассматриваться как набор операционных систем отдельных компьютеров, составляющих сеть. На разных компьютерах в сети установлены как правило, разные ОС, например Windows –2000, Windows NT, Linux. Но все эти ОС должны иметь в своем составе взаимно согласованный набор коммуникационных протоколов для организации взаимодействия процессов, выполняющихся на разных компьютерах сети, и разделения ресурсов компьютеров между пользователями сети. Сетевой ОС также называют ОС отдельного компьютера, которая имеет средства работы в сети.

Сетевая ОС функционально состоит из двух больших частей:

Средств управления локальными ресурсами компьютера (управление процессами, организация диалога пользователя, операции с файлами и т. д. );

Сетевых средств, в состав которых входит:

– Серверная часть ОС – предоставляет локальные услуги и ресурсы в общее пользование;

– Клиентская часть ОС – обработка запросов к удаленным ресурсам и услугам;

– Транспортные средства ОС – вместе с коммуникационной системой обеспечить передачу сообщений между компьютерами сети.

Упрощенно работа сетевой ОС происходит следующим образом. Предположим, что пользователь компьютера А решил разместить свой файл на диске другого компьютера сети - компьютера В.

Компьютер А данные Компьютер B

Порядок действий:

1. Пользователь компьютера А выполняет команды передачи данных на другой компьютер (мышью или набирает на клавиатуре);

2. Программный модуль ОС, отвечающий за диалог пользователя, принимает команду, определяет ее тип и передает ее клиентской части ОС компьютера А;

3. Клиентская часть компьютера А сама не может получить доступ к ресурсам компьютера В и посылает запрос на компьютер В, который передается транспортными средствами ОС;

4. Если запрос принят, то происходит передача данных, которые могут содержать либо набор команд, либо сами данные

Управляют передачей сообщений между клиентской и серверными частями по сети транспортные средства ОС, которые выполняют функции:

- Формирования данных для передачи – сообщений, т. е. добавляется служебная информация – адреса и т.д.

- Разбивку сообщения на части, если это необходимо;

- Преобразования символьного имени компьютера в числовой адрес;

- Проверку правильности передачи сообщения (проверку контрольной суммы );

- Определения маршрута передачи данных в сложной сети;

Для того, чтобы два компьютера в сети могли обмениваться сообщениями, транспортные средства должны поддерживать общие правила передачи информации (протокол). Коммуникационные протоколы переносят сообщения по сети, не вникая в их содержания.

Клиентская часть ОС должна уметь отличать запрос к удаленному файлу от запроса к локальному файлу. В этом случае приложения не должны заботиться о том, с каким файлом локальном или удаленным, они работают. Клиентская программа сама распознает и перенаправляет запрос к удаленному файлу. Клиентская часть или модуль, который выполняет функции распознавания, называется редиректор.

Клиентская часть ОС выполняет также преобразование форматов запросов к ресурсам. Запрос на доступ к ресурсам принимается в локальном формате, то есть в том который принят на локальном компьютере, в сеть запрос передается в формате, соответствующем требованиям серверной части ОС, работающей на компьютере, где расположен требуемый ресурс. Прием ответа от серверной части и преобразование его в локальный формат также выполняется клиентской частью ОС. Для приложения выполнение локального или удаленного запроса не различимо.

Компьютер, который занимается исключительно обслуживанием запросов других компьютеров – называется выделенным сервером, на нем устанавливается специальный вариант сетевой ОС – серверная ОС.

Компьютер, который обращается с запросами к другому компьютеру, - клиентский узел, на нем устанавливается клиентская ОС, которая позволяет

- Подключаться к сети;

- Передавать данные через сеть;

- Находить и использовать сетевые ресурсы, например принтер

Клиентское программное обеспечение не заменяет собой обычную ОС, а добавляет возможности пользоваться услугами сетевых служб, когда возникает необходимость.

Компьютер, который совмещает функции клиента и сервера – одноранговым узлом.

Многие сетевые ОС имеют два варианта выпуска: один предназначен для работы в качестве серверной ОС, другой – в качестве клиентской.

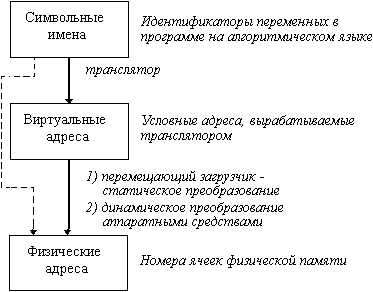
5. Особенности управления памятью в СОС.

Память является важнейшим ресурсом, требующим тщательного управления со стороны мультипрограммной операционной системы. Распределению подлежит вся оперативная память, не занятая операционной системой. Обычно ОС располагается в самых младших адресах, однако может занимать и самые старшие адреса. Функциями ОС по управлению памятью являются: отслеживание свободной и занятой памяти, выделение памяти процессам и освобождение памяти при завершении процессов, вытеснение процессов из оперативной памяти на диск, когда размеры основной памяти не достаточны для размещения в ней всех процессов, и возвращение их в оперативную память, когда в ней освобождается место, а также настройка адресов программы на конкретную область физической памяти.

Для идентификации переменных и команд используются символьные имена (метки), виртуальные адреса и физические адреса (рисунок 2.7).

Символьные имена присваивает пользователь при написании программы на алгоритмическом языке или ассемблере.

Виртуальные адреса вырабатывает транслятор, переводящий программу на машинный язык. Так как во время трансляции в общем случае не известно, в какое место оперативной памяти будет загружена программа, то транслятор присваивает переменным и командам виртуальные (условные) адреса, обычно считая по умолчанию, что программа будет размещена, начиная с нулевого адреса. Совокупность виртуальных адресов процесса называется *виртуальным адресным пространством*. Каждый процесс имеет собственное виртуальное адресное пространство. Максимальный размер виртуального адресного пространства ограничивается разрядностью адреса, присущей данной архитектуре компьютера, и, как правило, не совпадает с объемом физической памяти, имеющимся в компьютере.



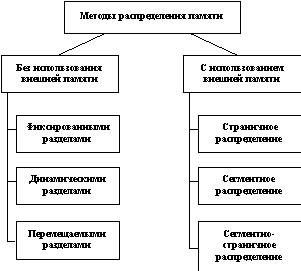
*Рис. 2.7. Типы адресов*

Физические адреса соответствуют номерам ячеек оперативной памяти, где в действительности расположены или будут расположены переменные и команды. Переход от виртуальных адресов к физическим может осуществляться двумя способами. В первом случае замену виртуальных адресов на физические делает специальная системная программа - перемещающий загрузчик. Перемещающий загрузчик на основании имеющихся у него исходных данных о начальном адресе физической памяти, в которую предстоит загружать программу, и информации, предоставленной транслятором об адресно-зависимых константах программы, выполняет загрузку программы, совмещая ее с заменой виртуальных адресов физическими.

Второй способ заключается в том, что программа загружается в память в неизмененном виде в виртуальных адресах, при этом операционная система фиксирует смещение действительного расположения программного кода относительно виртуального адресного пространства. Во время выполнения программы при каждом обращении к оперативной памяти выполняется преобразование виртуального адреса в физический. Второй способ является более гибким, он допускает перемещение программы во время ее выполнения, в то время как перемещающий загрузчик жестко привязывает программу к первоначально выделенному ей участку памяти. Вместе с тем использование перемещающего загрузчика уменьшает накладные расходы, так как преобразование каждого виртуального адреса происходит только один раз во время загрузки, а во втором случае - каждый раз при обращении по данному адресу.

В некоторых случаях (обычно в специализированных системах), когда заранее точно известно, в какой области оперативной памяти будет выполняться программа, транслятор выдает исполняемый код сразу в физических адресах.

Все методы управления памятью могут быть разделены на два класса: методы, которые используют перемещение процессов между оперативной памятью и диском, и методы, которые не делают этого (рисунок 2.8). Начнем с последнего, более простого класса методов.



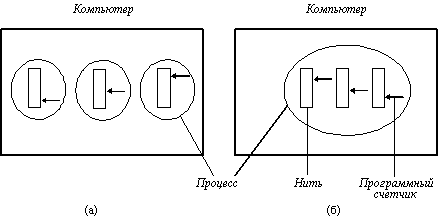
*Рис. 2.8. Классификация методов распределения памяти*

6. Особенности планирования процессов и нитей в СОС.

В традиционных ОС понятие нити тождественно понятию процесса. В действительности желательно иметь несколько нитей управления, разделяющих единое адресное пространство, но выполняющихся квазипараллельно.

Предположим, например, что файл-сервер блокируется, ожидания выполнения операции с диском. Если сервер имеет несколько нитей управления, вторая нить может выполняться, пока первая нить находится в состоянии ожидания. Это повышает пропускную способность и производительность. Эта цель не достигается путем создания двух независимых серверных процессов, потому что они должны разделять общий буфер кэша, который требуется им, чтобы быть в одном адресном пространстве.

На рисунке 3.9,а показана машина с тремя процессами. Каждый процесс имеет собственный программный счетчик, собственный стек, собственный набор регистров и собственное адресное пространство. Каждый процесс не должен ничего делать с остальными, за исключением того, что они могут взаимодействовать посредством системных примитивов связи, таких как семафоры, мониторы, сообщения. На рисунке 3.9,б показана другая машина с одним процессом. Этот процесс состоит из нескольких нитей управления, обычно называемых просто нитями или иногда облегченными процессами. Во многих отношениях нити подобны мини-процессам. Каждая нить выполняется строго последовательно и имеет свой собственный программный счетчик и стек. Нити разделяют процессор так, как это делают процессы (разделение времени). Только на многопроцессорной системе они действительно выполняются параллельно. Нити могут, например, порождать нити-потомки, могут переходить в состояние ожидания до завершения системного вызова, как обычные процессы, пока одна нить заблокирована, другая нить того же процесса может выполняться.



*Рис. 3.9. а) Три процесса с одной нитью каждый   
б) Один процесс с тремя нитями*

Нити делают возможным сохранение идеи последовательных процессов, которые выполняют блокирующие системные вызовы (например, RPC для обращения к диску), и в то же время позволяют достичь параллелизма вычислений. Блокирующие системные вызовы делают проще программирование, а параллелизм повышает производительность.

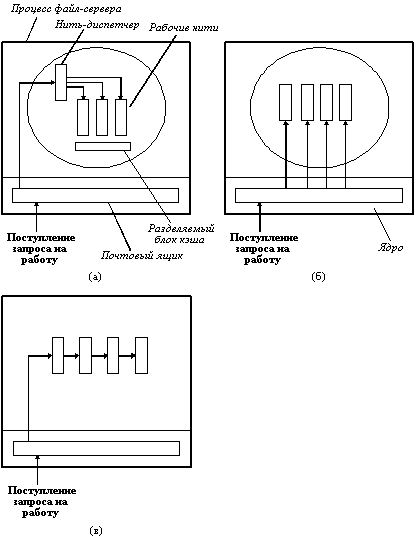
Один из возможных способов организации вычислительного процесса показан на рисунке 3.10,а. Здесь нить-диспетчер читает приходящие запросы на работу из почтового ящика системы. После проверки запроса диспетчер выбирает простаивающую (то есть блокированную) рабочую нить, передает ей запрос и активизирует ее, устанавливая, например, семафор, который она ожидает.

Когда рабочая нить активизируется, она проверяет, может ли быть выполнен запрос с данными разделяемого блока кэша, к которому имеют отношение все нити. Если нет, она посылает сообщение к диску, чтобы получить нужный блок (предположим, это READ), и переходит в состояние блокировки, ожидая завершения дисковой операции. В этот момент происходит обращение к планировщику, в результате работы которого активизируется другая нить, возможно, нить-диспетчер или некоторая рабочая нить, готовая к выполнению.

Структура с диспетчером не единственный путь организации многонитевой обработки. В модели "команда" все нити эквивалентны, каждая получает и обрабатывает свои собственные запросы. Иногда работы приходят, а нужная нить занята, особенно, если каждая нить специализируется на выполнении особого вида работ. В этом случае может создаваться очередь незавершенных работ. При такой организации нити должны вначале просматривать очередь работ, а затем почтовый ящик.

Нити могут быть также организованы в виде конвейера. В этом случае первая нить порождает некоторые данные и передает их для обработки следующей нити и т.д. Хотя эта организация и не подходит для файл-сервера, для других задач, например, задач типа "производитель-потребитель", это хорошее решение.

Нити часто полезны и для клиентов. Например, если клиент хочет растиражировать файл на много серверов, он может создать по одной нити для копирования на каждом сервере. Другое использование нитей клиентами - это управление сигналами, такими как прерывание с клавиатуры (del или break). Вместо обработки сигнала прерывания одна нить назначается для постоянного ожидания поступления сигналов. Таким образом, использование нитей может сократить необходимое количество прерываний пользовательского уровня.



*Рис. 3.10. Три способа организации нитей в процессе:   
а - модель диспетчер/рабочие нити; б - модель "команда"; в - модель конвейера*

Другой аргумент в пользу нитей не имеет отношения ни к удаленным вызовам, ни к коммуникациям. Некоторые прикладные задачи легче программировать, используя параллелизм, например задачи типа "производитель-потребитель". Не столь важно параллельное выполнение, сколь важна ясность программы. А поскольку они разделяют общий буфер, не стоит их делать отдельными процессами.

Наконец, в многопроцессорных системах нити из одного адресного пространства могут выполняться параллельно на разных процессорах. С другой стороны, правильно сконструированные программы, которые используют нити, должны работать одинаково хорошо на однопроцессорной машине в режиме разделения времени между нитями и на настоящем мультипроцессоре.

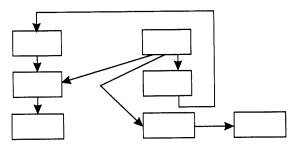
7. Организация взаимодействия удаленных процессов.

Все перечисленные выше цели объединения компьютеров в *вычислительные сети* не могут быть достигнуты без организации взаимодействия процессов на различных вычислительных системах. Будь то *доступ* к разделяемым ресурсам или общение пользователей через *сеть* – в основе всего этого лежит *взаимодействие удаленных процессов*, т. е. процессов, которые находятся под управлением физически разных операционных систем. Поэтому мы в своей работе сосредоточимся именно на вопросах кооперации таких процессов, в первую *очередь* выделив ее отличия от *кооперации процессов* в одной автономной вычислительной системе (кооперации *локальных процессов* ), о которой мы говорили в лекциях 4, 5 и 6.

1. Изучая взаимодействие *локальных процессов*, мы разделили средства обмена информацией по объему передаваемых между ними данных и возможности влияния на поведение другого процесса на три категории: сигнальные, канальные и *разделяемая память*. На самом деле во всей этой систематизации присутствовала некоторая доля лукавства. Мы фактически классифицировали средства связи по виду интерфейса обращения к ним, в то время как реальной физической основой для всех средств связи в том или ином виде являлось разделение памяти. Семафоры представляют собой просто целочисленные переменные, лежащие в разделяемой памяти, к которым посредством системных вызовов, определяющих состав и содержание допустимых операций над ними, могут обращаться различные процессы. Очереди сообщений и *pip*'ы базируются на буферах ядра операционной системы, которые опять-таки с помощью системных вызовов доступны различным процессам. Иного способа реально передать информацию от процесса к процессу в автономной вычислительной системе просто не существует. *Взаимодействие удаленных процессов* принципиально отличается от ранее рассмотренных случаев. Общей памяти у различных компьютеров физически нет. *Удаленные процессы* могут обмениваться информацией, только передавая друг другу пакеты данных определенного формата (в виде последовательностей электрических или электромагнитных сигналов, включая световые) через некоторый *физический канал* связи или несколько таких каналов, соединяющих компьютеры. Поэтому в основе всех средств *взаимодействия удаленных процессов* лежит передача структурированных пакетов информации или сообщений.
2. При взаимодействии *локальных процессов* и процесс–отправитель информации, и процесс-получатель функционируют под управлением одной и той же операционной системы. Эта же операционная система поддерживает и функционирование промежуточных *накопителей данных* при использовании *непрямой адресации*. Для организации взаимодействия процессы пользуются одними и теми же системными вызовами, присущими данной операционной системе, с одинаковыми интерфейсами. Более того, в автономной операционной системе передача информации от одного процесса к другому, независимо от используемого способа адресации, как правило (за исключением микроядерных операционных систем), происходит напрямую – без участия других процессов-посредников. Но даже и при наличии процессов-посредников все участники передачи информации находятся под управлением одной и той же операционной системы. При организации сети, конечно, можно обеспечить прямую связь между всеми вычислительными комплексами, соединив каждый из них со всеми остальными посредством прямых физических линий связи или подключив все комплексы к общей шине (по примеру шин данных и адреса в компьютере). Однако такая сетевая топология не всегда возможна по ряду физических и финансовых причин. Поэтому во многих случаях информация между *удаленными процессами* в сети передается не напрямую, а через ряд процессов-посредников, "обитающих" на вычислительных комплексах, не являющихся компьютерами отправителя и получателя и работающих под управлением собственных операционных систем. Однако и при отсутствии процессов-посредников удаленные процесс-отправитель и процесс-получатель функционируют под управлением различных операционных систем, часто имеющих принципиально разное строение.
3. Вопросы надежности средств связи и способы ее реализации, рассмотренные нами в лекции 4, носили для случая *локальных процессов* скорее теоретический характер. Мы выяснили, что физической основой "общения" процессов на автономной вычислительной машине является *разделяемая память*. Поэтому для *локальных процессов* надежность передачи информации определяется надежностью ее передачи по шине данных и хранения в памяти машины, а также корректностью работы операционной системы. Для хороших вычислительных комплексов и операционных систем мы могли забыть про возможную ненадежность средств связи. Для *удаленных процессов* вопросы, связанные с надежностью передачи данных, становятся куда более значимыми. Протяженные сетевые линии связи подвержены разнообразным физическим воздействиям, приводящим к искажению передаваемых по ним физических сигналов (помехи в эфире) или к полному отказу линий (мыши съели кабель). Даже при отсутствии внешних помех передаваемый сигнал затухает по мере удаления от точки отправления, приближаясь по интенсивности к внутренним шумам линий связи. Промежуточные вычислительные комплексы сети, участвующие в доставке информации, не застрахованы от повреждений или внезапной перезагрузки операционной системы. Поэтому *вычислительные сети*должны организовываться исходя из предпосылок ненадежности доставки физических пакетов информации.
4. При организации взаимодействия *локальных процессов* каждый процесс (в случае прямой адресации) и каждый промежуточный объект для накопления данных (в случае *непрямой адресации*) должны были иметь уникальные идентификаторы – адреса – в рамках одной операционной системы. При организации *взаимодействия удаленных процессов* участники этого взаимодействия должны иметь уникальные адреса уже в рамках всей сети.
5. Физическая линия связи, соединяющая несколько вычислительных комплексов, является разделяемым ресурсом для всех процессов комплексов, которые хотят ее использовать. Если два процесса попытаются одновременно передать пакеты информации по одной и той же линии, то в результате интерференции физических сигналов, представляющих эти пакеты, произойдет взаимное искажение передаваемых данных. Для того чтобы избежать возникновения такой ситуации (*race condition*!) и обеспечить эффективную совместную работу вычислительных систем, должны выполняться условия взаимоисключения, прогресса и ограниченного ожидания при использовании общей линии связи, но уже не на уровне отдельных процессов операционных систем, а на уровне различных вычислительных комплексов в целом.

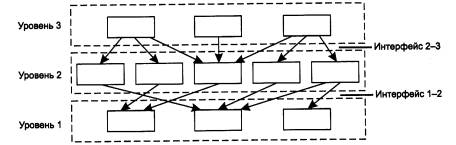
8. Многоуровневая модель сетевого взаимодействия. Протоколы.Интерфейсы.

Организация взаимодействия между устройствами в сети является сложной задачей. Как известно, для решения сложных задач используется универсальный прием - декомпозиция, то есть разбиение одной сложной задачи на несколько более простых задач-модулей (рис. 1.20). Процедура декомпозиции включает в себя четкое определение функций каждого модуля, решающего отдельную задачу, и интерфейсов между ними. В результате достигается логическое упрощение задачи, а кроме того, появляется возможность модификации отдельных модулей без изменения остальной части системы.



**Рис. 1.20.** Пример декомпозиции задачи

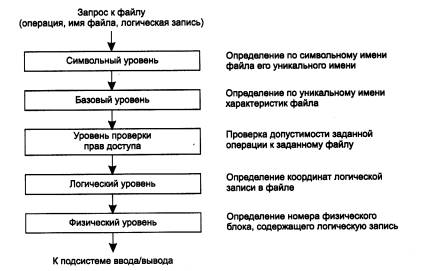
При декомпозиции часто используют многоуровневый подход. Он заключается в следующем. Все множество модулей разбивают на уровни. Уровни образуют иерархию, то есть имеются вышележащие и нижележащие уровни (рис. 1.21). Множество модулей, составляющих каждый уровень, сформировано таким образом, что для выполнения своих задач они обращаются с запросами только к модулям непосредственно примыкающего нижележащего уровня. С другой стороны, результаты работы всех модулей, принадлежащих некоторому уровню, могут быть переданы только модулям соседнего вышележащего уровня. Такая иерархическая декомпозиция задачи предполагает четкое определение функции каждого уровня и интерфейсов между уровнями. Интерфейс определяет набор функций, которые нижележащий уровень предоставляет вышележащему. В результате иерархической декомпозиции достигается относительная независимость уровней, а значит, и возможность их легкой замены.



**Рис. 1.21.** Многоуровневый подход - создание иерархии задач

Средства сетевого взаимодействия, конечно, тоже могут быть представлены в виде иерархически организованного множества модулей. При этом модули нижнего уровня могут, например, решать все вопросы, связанные с надежной передачей электрических сигналов между двумя соседними узлами. Модули более высокого уровня организуют транспортировку сообщений в пределах всей сети, пользуясь для этого средствами упомянутого нижележащего уровня. А на верхнем уровне работают модули, предоставляющие пользователям доступ к различным службам - файловой, печати и т. п. Конечно, это только один из множества возможных вариантов деления общей задачи организации сетевого взаимодействия на частные подзадачи.

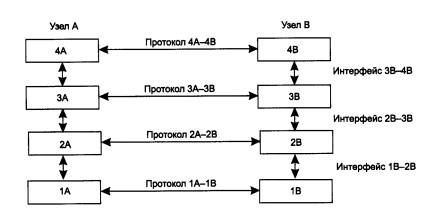
Многоуровневый подход к описанию и реализации функций системы применяется не только в отношении сетевых средств. Такая модель функционирования используется, например, в локальных файловых системах, когда поступивший запрос на доступ к файлу последовательно обрабатывается несколькими программными уровнями (рис. 1.22). Запрос вначале анализируется верхним уровнем, на котором осуществляется последовательный разбор составного символьного имени файла и определение уникального идентификатора файла. Следующий уровень находит по уникальному имени все основные характеристики файла: адрес, атрибуты доступа и т. п. Затем на более низком уровне осуществляется проверка прав доступа к этому файлу, а далее, после расчета координат области файла, содержащей требуемые данные, выполняется физический обмен с внешним устройством с помощью драйвера диска.



**Рис. 1.22.** Многоуровневая модель файловой системы

Многоуровневое представление средств сетевого взаимодействия имеет свою специфику, связанную с тем, что в процессе обмена сообщениями участвуют две машины, то есть в данном случае необходимо организовать согласованную работу двух «иерархий». При передаче сообщений оба участника сетевого обмена должны принять множество соглашений. Например, они должны согласовать уровни и форму электрических сигналов, способ определения длины сообщений, договориться о методах контроля достоверности и т. п. Другими словами, соглашения должны быть приняты для всех уровней, начиная от самого низкого - уровня передачи битов - до самого высокого, реализующего сервис для пользователей сети.

На рис. 1.23 показана модель взаимодействия двух узлов. С каждой стороны средства взаимодействия представлены четырьмя уровнями. Процедура взаимодействия этих двух узлов может быть описана в виде набора правил взаимодействия каждой пары соответствующих уровней обеих участвующих сторон. Формализованные правила, определяющие последовательность и формат сообщений, которыми обмениваются сетевые компоненты, лежащие на одном уровне, но в разных узлах, называются *протоколом*.



**Рис. 1.23.** Взаимодействие двух узлов

Модули, реализующие протоколы соседних уровней и находящиеся в одном узле, также взаимодействуют друг с другом в соответствии с четко определенными правилами и с помощью стандартизованных форматов сообщений. Эти правила принято называть *интерфейсом*. Интерфейс определяет набор сервисов, предоставляемый данным уровнем соседнему уровню. В сущности, протокол и интерфейс выражают одно и то же понятие, но традиционно в сетях за ними закрепили разные области действия: протоколы определяют правила взаимодействия модулей одного уровня в разных узлах, а интерфейсы - модулей соседних уровней в одном узле.

Средства каждого уровня должны отрабатывать, во-первых, свой собственный протокол, а во-вторых, интерфейсы с соседними уровнями.

Иерархически организованный набор протоколов, достаточный для организации взаимодействия узлов в сети, называется *стеком коммуникационных протоколов*.

Коммуникационные протоколы могут быть реализованы как программно, так и аппаратно.

9. Способы адресации удаленных процессов. Маршрутизация. Синхронизация удаленных процессов.

К числу наиболее фундаментальных вопросов, связанных с логической организацией *взаимодействия удаленных процессов*, можно отнести следующие:

1. Как нужно соединять между собой различные вычислительные системы физическими линиями связи для организации *взаимодействия удаленных процессов*? Какими критериями при этом следует пользоваться?
2. Как избежать возникновения race condition при передаче информации различными вычислительными системами после их подключения к общей линии связи? Какие алгоритмы могут при этом применяться?
3. Какие виды интерфейсов могут быть предоставлены пользователю операционными системами для передачи информации по сети? Какие существуют модели *взаимодействия удаленных процессов*? Как процессы, работающие под управлением различных по своему строению операционных систем, могут общаться друг с другом?
4. Какие существуют подходы к организации адресации *удаленных процессов*? Насколько они эффективны?
5. Как организуется доставка информации от компьютера-отправителя к компьютеру-получателю через компьютеры-посредники? Как выбирается маршрут для передачи данных в случае разветвленной сетевой структуры, когда существует не один вариант следования пакетов данных через компьютеры-посредники?

Разумеется, степень важности этих вопросов во многом зависит от того, с какой точки зрения мы рассматриваем *взаимодействие удаленных процессов*. Системного программиста, в первую очередь, интересуют интерфейсы, предоставляемые операционными системами. Сетевого администратора больше будут занимать вопросы адресации процессов и выбора маршрута доставки данных. Проектировщика сетей в организации – способы соединения компьютеров и обеспечения корректного использования разделяемых сетевых ресурсов. Мы изучаем особенности строения и функционирования частей операционных систем, ответственных за *взаимодействие удаленных процессов*, а поэтому рассматриваемый перечень вопросов существенно сокращается.

Выбор способа соединения участников сетевого взаимодействия физическими линиями связи (количество и тип прокладываемых коммуникаций, какие именно устройства и как они будут соединять, т. е. топология сети) определяется проектировщиками сетей исходя из имеющихся средств, требуемой производительности и надежности взаимодействия. Все это не имеет отношения к функционированию операционных систем, является внешним по отношению к ним и в нашем курсе рассматриваться не будет.

В современных сетевых вычислительных комплексах решение вопросов организации взаимоисключений при использовании общей линии связи, как правило, также находится вне компетенции операционных систем и вынесено на *физический уровень* организации взаимодействия. Ответственность за корректное использование коммуникаций возлагается на сетевые адаптеры, поэтому подобные проблемы мы тоже рассматривать не будем.

Из приведенного перечня мы с вами подробнее остановимся на решении вопросов, приведенных в пунктах 3–5.

Понятие протокола

Для описания происходящего в автономной операционной системе в лекции 2 было введено основополагающее понятие "процесс", на котором, по сути дела, базируется весь наш курс. Для того чтобы описать *взаимодействие удаленных процессов* и понять, какие функции и как должны выполнять дополнительные части *сетевых операционных систем*, отвечающих за такое взаимодействие, нам понадобится не менее фундаментальное понятие –*протокол*.

"Общение" *локальных процессов* напоминает общение людей, проживающих в одном городе. Если взаимодействующие процессы находятся под управлением различных операционных систем, то эта ситуация подобна общению людей, проживающих в разных городах и, возможно, в разных странах.

Каким образом два человека, находящиеся в разных городах, а тем более странах, могут обмениваться информацией? Для этого им обычно приходится прибегать к услугам соответствующих служб связи. При этом между службами связи различных городов (государств) должны быть заключены определенные соглашения, позволяющие корректно организовывать такой обмен. Если речь идет, например, о почтовых отправлениях, то в первую очередь необходимо договориться о том, что может представлять собой почтовое отправление, какой вид оно может иметь. Некоторые племена индейцев для передачи информации пользовались узелковыми письмами – поясами, к которым крепились веревочки с различным числом и формой узелков. Если бы такое письмо попало в современный почтовый ящик, то, пожалуй, ни одно отделение связи не догадалось бы, что это – письмо, и пояс был бы выброшен как ненужный хлам. Помимо формы представления информации необходима договоренность о том, какой служебной информацией должно снабжаться почтовое отправление (адрес получателя, срочность доставки, способ пересылки: поездом, авиацией, с помощью курьера и т. п.) и в каком формате она должна быть представлена. Адреса, например, в России и в США принято записывать по-разному. В России мы начинаем адрес со страны, далее указывается город, улица и квартира. В США все наоборот: сначала указывается квартира, затем улица и т. д. Конечно, даже при неправильной записи адреса письмо, скорее всего, дойдет до получателя, но можно себе представить растерянность почтальона, пытающегося разгадать, что это за страна или город – "кв.162"? Как видим, доставка почтового отправления из одного города (страны) в другой требует целого ряда соглашений между почтовыми ведомствами этих городов (стран).

Аналогичная ситуация возникает и при общении *удаленных процессов*, работающих под управлением разных операционных систем. Здесь процессы играют роль людей, проживающих в разных городах, а сетевые части операционных систем – роль соответствующих служб связи. Для того чтобы *удаленные процессы* могли обмениваться данными, необходимо, чтобы сетевые части операционных систем руководствовались определенными соглашениями, или, как принято говорить, поддерживали определенные *протоколы*. Термин*"протокол"*уже встречался нам в лекции 13, посвященной организации ввода-вывода в операционных системах, при обсуждении понятия "шина". Мы говорили, что понятие шины подразумевает не только набор проводников, но и набор жестко заданных *протоколов*, определяющий перечень сообщений, который может быть передан с помощью электрических сигналов по этим проводникам, т. е. в*"протокол"*мы вкладывали практически тот же смысл. В следующем разделе мы попытаемся дать более формализованное определение этого термина.

Необходимо отметить, что и *локальные процессы* при общении также должны руководствоваться определенными соглашениями или поддерживать определенные *протоколы*. Только в автономных операционных системах они несколько завуалированы. В роли таких *протоколов* выступают специальная последовательность системных вызовов при организации взаимодействия процессов и соглашения о параметрах системных вызовов.

Различные способы решения проблем 3–5, поднятых в предыдущем разделе, по существу, представляют собой различные соглашения, которых должны придерживаться сетевые части операционных систем, т. е. различные *сетевые протоколы*. Именно наличие *сетевых протоколов* позволяет организовать *взаимодействие удаленных процессов*.

10. Цели и задачи сетевого администрирования. Аудит пользователей и ресурсов.

Современные корпоративные *информационные системы* по своей природе всегда являются распределенными системами. Рабочие станции пользователей, серверы приложений, серверы баз данных и прочие сетевые узлы распределены по большой территории. В крупной компании офисы и площадки соединены различными видами коммуникаций, использующих различные технологии и сетевые устройства. Главная задача сетевого администратора — обеспечить надежную, бесперебойную, производительную и безопасную работу всей этой сложной системы.

Будем рассматривать *сеть* как совокупность программных, аппаратных и коммуникационных средств, обеспечивающих эффективное распределение вычислительных ресурсов. Все сети можно условно разделить на 3 категории:

* локальные сети (LAN, Local Area Network);
* глобальные сети (WAN, Wide Area Network);
* городские сети (MAN, *Metropolitan Area Network*).

*Глобальные сети* позволяют организовать взаимодействие между абонентами на больших расстояниях. Эти сети работают на относительно низких скоростях и могут вносить значительные задержки в передачу информации. Протяженность глобальных сетей может составлять тысячи километров. Поэтому они так или иначе интегрированы с сетями масштаба страны.

Городские сети позволяют взаимодействовать на территориальных образованиях меньших размеров и работают на скоростях от средних до высоких. Они меньше замедляют передачу данных, чем глобальные, но не могут обеспечить высокоскоростное взаимодействие на больших расстояниях. Протяженность городских сетей находится в пределах от нескольких километров до десятков и сотен километров.

*Локальные сети* обеспечивают наивысшую скорость обмена информацией между компьютерами. Типичная *локальная сеть* занимает *пространство* в одно здание. Протяженность локальных сетей составляет около одного километра. Их основное назначение состоит в объединении пользователей (как правило, одной компании или организации) для совместной работы.

*Механизмы* передачи данных в локальных и глобальных сетях существенно отличаются. *Глобальные сети* ориентированы на соединение — до начала передачи данных между абонентами устанавливается соединение (*сеанс*). В локальных сетях используются методы, не требующие предварительной установки соединения, — пакет с данными посылается без подтверждения готовности получателя к обмену.

1. Планирование сети.

Несмотря на то, что планированием и монтажом больших сетей обычно занимаются специализированные компании-интеграторы, сетевому администратору часто приходится планировать определенные изменения в структуре сети — добавление новых рабочих мест, добавление или удаление сетевых протоколов, добавление или удаление сетевых служб, установка серверов, разбиение сети на сегменты и т.д. Данные работы должны быть тщательно спланированы, чтобы новые устройства, узлы или протоколы включались в сеть или исключались из нее без нарушения целостности сети, без снижения производительности, без нарушения инфраструктуры сетевых протоколов, служб и приложений.

1. Установка и настройка сетевых узлов (устройств активного сетевого оборудования, персональных компьютеров, серверов, средств коммуникаций).

Данные работы могут включать в себя — замену сетевого адаптера в ПК с соответствующими настройками компьютера, перенос сетевого узла (ПК, сервера, активного оборудования) в другую подсеть с соответствующим изменениями сетевых параметров узла, добавление или замена сетевого принтера с соответствующей настройкой рабочих мест.

1. Установка и настройка сетевых протоколов.

Данная задача включает в себя выполнение таких работ — планирование и настройка базовых сетевых протоколов корпоративной сети, тестирование работы сетевых протоколов, определение оптимальных конфигураций протоколов.

1. Установка и настройка сетевых служб.

Корпоративная сеть может содержать большой набор сетевых служб. Кратко перечислим основные задачи администрирования сетевых служб:

* + установка и настройка служб сетевой инфраструктуры (службы DNS, DHCP, WINS, службы маршрутизации, удаленного доступа и виртуальных частных сетей);
  + установка и настройка служб файлов и печати, которые в настоящее время составляют значительную часть всех сетевых служб;
  + администрирование служб каталогов (Novell *NDS*, Microsoft Active Directory), составляющих основу корпоративной системы безопасности и управления доступом к сетевым ресурсам;
  + администрирование служб обмена сообщениями (системы электронной почты);
  + администрирование служб доступа к базам данных.

1. Поиск неисправностей.

Сетевой администратор должен уметь обнаруживать широкий спектр неисправностей — от неисправного сетевого адаптера на рабочей станции пользователя до сбоев отдельных портов коммутаторов и маршрутизаторов, а также неправильные настройки сетевых протоколов и служб.

1. Поиск узких мест сети и повышения эффективности работы сети.

В задачу сетевого администрирования входит анализ работы сети и определение наиболее узких мест, требующих либо замены сетевого оборудования, либо модернизации рабочих мест, либо изменения конфигурации отдельных сегментов сети.

1. Мониторинг сетевых узлов.

Мониторинг сетевых узлов включает в себя наблюдение за функционированием сетевых узлов и корректностью выполнения возложенных на данные узлы функций.

1. Мониторинг сетевого трафика.

Мониторинг сетевого трафика позволяет обнаружить и ликвидировать различные виды проблем: высокую загруженность отдельных сетевых сегментов, чрезмерную загруженность отдельных сетевых устройств, сбои в работе сетевых адаптеров или портов сетевых устройств, нежелательную активность или атаки злоумышленников (распространение вирусов, атаки хакеров и др.).

1. Обеспечение защиты данных.

Защита данных включает в себя большой набор различных задач: резервное копирование и восстановление данных, разработка и осуществление политик безопасности учетных записей пользователей и сетевых служб (требования к сложности паролей, частота смены паролей), построение защищенных коммуникаций (применение протокола IPSec, построение виртуальных частных сетей, защита беспроводных сетей), планирование, внедрение и обслуживание инфраструктуры открытых ключей (PKI).

Тестовые задания:

1. Предоставляет пользователю некую виртуальную вычислительную систему, работать с которой гораздо проще, чем с реальной сетевой аппаратурой.

*Сетевая ОС.*

2.Динамически и автоматически распределяя работы по различным машинам системы для обработки, заставляет набор сетевых машин работать как виртуальный унипроцессор.

*Распределенная ОС.*

3. Сетевая ОС предоставляющая пользователю сетевые ресурсы в виде ресурсов единой централизованной виртуальной машины.

*Распределенная ОС или истинно распределенная ОС.*

4. Средства предоставления локальных ресурсов и услуг в общее пользование.

*Серверная часть ОС*

5. Средства запроса доступа к удаленным ресурсам и услугам.

*Клиентская часть ОС.*

6. Средства ОС, которые совместно с коммуникационной системой обеспечивают передачу сообщений между компьютерами сети.

*Транспортные средства ОС.*

7. Совокупность серверной и клиентской частей ОС, предоставляющих доступ к конкретному типу ресурса компьютера через сеть.

*Сетевая служба.*

8. Сетевая служба предоставляет пользователям сети некоторый набор услуг.

*Сетевой сервис.*

9. Сетевая служба может быть представлена в операционной системе …

*Клиентской и серверной частями, либо только одной из них.*

10. Совокупность компьютеров, связанных коммуникационным оборудованием, работающих под управлением сетевой операционной системы.

*Компьютерная сеть.*

11. Набор операционных систем отдельных компьютеров, составляющих сеть.

*Сетевая ОС.*

12. Компьютер, который занимается исключительно обслуживанием запросов других компьютеров.

*Выделенный сервер.*

13. Для идентификации переменных и команд используются …

*Символьные имена (метки), виртуальные адреса и физические адреса.*

14. Совокупность виртуальных адресов процесса.

*Виртуальное адресное пространство.*

15. Все методы управления памятью могут быть разделены на два класса: …

*Методы, которые используют перемещение процессов между оперативной памятью и диском, и методы, которые не делают этого.*

16. В традиционных ОС понятие процесса тождественно …

*Понятию нити.*

17. Для чего нужны нити?

*Решение нескольких задач одновременно, что повышает пропускную способность и производительность.*

18. Могут ли в многопроцессорных системах нити из одного адресного пространства могут выполняться параллельно на разных процессорах?

*Да.*

19. Средства обмена информацией по объему передаваемых между ними данных и возможности влияния на поведение другого процесса делятся на три категории: …

*Сигнальные, канальные и разделяемая память.*

20. Если два процесса попытаются одновременно передать пакеты информации по одной и той же линии, то в результате …

*Произойдет взаимное искажение передаваемых данных.*

21. Для того чтобы избежать возникновения такой ситуации используется термин …

*Race condition.*

22. Определяет набор функций, которые нижележащий уровень предоставляет вышележащему.

*Интерфейс.*

23. Формализованные правила, определяющие последовательность и формат сообщений, которыми обмениваются сетевые компоненты, лежащие на одном уровне, но в разных узлах.

*Протокол.*

24. Иерархически организованный набор протоколов, достаточный для организации взаимодействия узлов в сети, называется …

*Стеком коммуникационных протоколов.*

25. В современных сетевых вычислительных комплексах решение вопросов организации взаимоисключений при использовании общей линии связи, как правило, также находится вне компетенции операционных систем и вынесено на … уровень.

*Физический уровень.*

26. Что позволяет организовать взаимодействие удаленных процессов.

*Наличие сетевых протоколов.*

27. Процессы, взаимодействующие в рамках одной ОС.

*Локальные процессы.*

28. Главная задача сетевого администратора …

Обеспечить надежную, бесперебойную, производительную и безопасную работу всей этой сложной системы.

29. Совокупность программных, аппаратных и коммуникационных средств, обеспечивающих эффективное распределение вычислительных ресурсов.

*Сеть.*

30. Что включает в себя мониторинг?

*Наблюдение за функционированием сетевых узлов и корректностью выполнения возложенных на данные узлы функций.*