Мориков Д. А. ПО-215А

Назаров, Широков “Современные ОС”

***Реферат и презентация “ОС Linux”***

**1к**

**Понятия и функции операционных систем**

**Операционная система (ОС)** – это комплекс программ, обеспечивающий организацию вычислительного процесса на ЭВМ, взаимодействие пользователя с вычислительной системой и управляющей ресурсами вычислительной системы.

**Вычислительная система (ВС)** – это программно-аппаратный комплекс, предоставляющий пользователю какие-либо услуги.

Функции ОС:

1. Управление процессами, задачами
2. Управление ресурсами
3. Взаимодействие с пользователем

**Иерархическая структура программно-аппаратных средств компьютера**

{ Прикладные программы }*-* ***пользователь***

{ Утилиты Компиляторы Редакторы Интерпретаторы команд } -***Программист***

{ Машинный язык Микроархитектура Физич.устр-ва. } - ***Разработчик ОС***

Задачи ОС:

1. Увеличение пропускной способности ЭВМ
2. Уменьшение времени реакции системы на запросы пользователя
3. Упрощение работы разработчиков ПО и сотрудников обслуживающего персонала ЭВМ

Так же ОС позволяет обеспечить выполнение следующих функций:

1. Автоматический запуск программ
2. Согласование совместной работы всех устройств компьютера
3. Распределение ресурсов компьютера
4. Организация записи, считывания, хранения и удаления информации с носителей

**Операционная среда** – это программная среда, образуемая ОС, определяющая интерфейс прикладного программирования, как множество системных функций и сервисов (системных вызовов) которые предоставляются прикладным программам.

Основные требования к разработке ПО ЭВМ:

1. Модульность
2. Гибкость, адаптируемость
3. Минимальное вмешательство человека
4. Функциональная избыточность, функциональная избирательность
5. Параметрическая универсальность

**ОС** – это набор программ, контролирующих работу прикладных программ и системных приложений, и выполняющих роль интерфейса между пользователями, программистами, прикладными программами, системными приложениями и аппаратным обеспечением компьютера.

**2к**

**Классификация ОС**

**ДЗ Привести примеры ОС по классификациям**

По назначению:

* Универсальные
* Специализированные

По способу загрузки:

* Загружаемые ОС
* Постоянно находящиеся в памяти

По особенностям алгоритмов управления ресурсами: (по алгоритмам управления процессором)

1. Поддержка многозадачности (многопрограммности)
   1. Однозадачные (однопрограммные)
   2. Многозадачные (многопрограммные)
2. Поддержка многопользовательского режима
   1. Однопользовательские
   2. Многопользовательские
3. Виды многопрограммной работы
   1. Невытесняющая многопрограммность
   2. Вытесняющая
4. Многопроцессорная обработка
   1. Без поддержки мультипроцессирования
   2. С поддержкой мультипроцессирования

По области использования и форме эксплуатации

**Системы пакетной обработки**

*Предназначены для решения задач вычислительного характера, не требующих получения быстрого результата*

*Основным критерием их эффективности является максимальная пропускная способность при максимальной загрузке всех ресурсов компьютера, поэтому в этих системах пользователь отстранен от компьютера.*

**Системы разделения времени**

*Обеспечивают удобство и эффективность работы пользователя, который может вести диалог с программой, системы предназначены для решения задач требующих быстрого получения результатов.*

**Системы реального времени**

*Предназначены для управления техническими процессами (объектами), где существует предельное время на выполнение программ, управляющих объектом.*

Основной критерий эффективности таких систем – **реактивность**.

По аппаратной платформе: (типу вычислительной техники)

* ОС для смарт карт
* Встроенные ОС
* ОС для персональных компьютеров
* ОС для мини ЭВМ
* ОС для мейнфреймов
* Серверные ОС
* Кластерные ОС

**Кластер** – это слабосвязанная совокупность нескольких вычислительных систем, работающих совместно, для выполнения общих приложений и представляющихся пользователю единой системой.

**История развития ОС**

1. 1945 – 1955

Ламповые машины, отсутствие ОС

За пультом управления мог находиться только 1 человек, одна и та же группа людей участвовала и в проектировании, и в эксплуатации, и в программировании. Программа вводилась в память машины с помощью панели переключателя или с колоды перфокарт.

ВС выполняла одновременно только одну операцию, в конце периода появляется первое системное ПО - 1951-1952 – прообразы первых компиляторов с символических языков. Все программы выполнялись строго последовательно. ВС имеют высокую стоимость и низкую эффективность использования.

1. 1955-начало 1960 годов

Компьютеры на основе транзисторов. Пакетные ОС

Вместо ламп применяются транзисторы, поэтому снижается потребление электроэнергии, стоимость эксплуатации и обслуживания, совершенствуется система охлаждения.

Начинается использование ЭВМ коммерческими фирмами, развиваются алгоритмические языки, появляются первые компиляторы, библиотеки подпрограмм, редакторы связи, упрощается процесс программирования, изменяется процесс выполнения программ – пользователь приносит программу с входными данными и указывает необходимые ресурсы, программа в виде колоды перфокарт получает название “задания”, оператор загружает задания в память машины и запускает на выполнение, таким образом происходит разделение персонала на операторов, пользователей, программистов и.т.д.

Так как смена ресурсов вызывает приостановку программы, а в результате процессор простаивает, то для повышения эффективности использования компьютера, задания имеющие схожие ресурсы начинают собирать вместе, создавая пакет заданий.

Появляются системы пакетной обработки, которые просто автоматизируют запуск одной программы из пакета за другой, увеличивая коэффицент загрузки процессора, был разработан формализованный язык управления заданиями.

Таким образом, системы пакетной обработки стали прообразом современных ОС и были первыми системными программами для управления вычислительным процессом.

1. Начало 60-ых – 1980г.

Компьютеры на основе интегральных микросхем. Первые многозадачные ОС.

Происходит переход от отдельных полупроводниковых элементов типа транзисторов к интегральным микросхемам

Система становится более дешевой, надежной, повышается производительность процессоров.

Повышение эффективности использования процессорного времени мешает низкая скорость работы механических устройств ввода и вывода. Поэтому вместо прямого чтения данных с перфокарт в память начинают выполнять предварительную запись пакета заданий сначала на ленту, затем на диск.

Стечением времени операции ввода, вывода начинают выполняться на том же ПК, котором происходят вычисления. – такая техника получает название **Спулинг** или **подкачки-откачки данных.**

Это повлекло за собой разработку аппарата прерываний для извещения процессора об окончании этих операций.

При обработке пакета заданий на магнитном диске, появилась возможность выбора очередного выполняемого задания.

Пакетные системы начинают заниматься планированием заданий. Появляется и развивается мультипрограммирование – пока одна программа выполняет операцию ввода-вывода, процессор выполняет другую программу, при окончании операции процессор возвращается к выполнению первой программы.

**Это потребовало аппаратной поддержки в виде следующих механизмов:**

- Реализация защитных механизмов, в виде появление привилегированных и непривилегированных команд, а так же защиты памяти

- Наличие прерываний

- Развитие параллелизма в архитектуре компьютера

**Роль ОС в организации мультипрограммирования:**

- Организация интерфейса между прикладной программой и ОС с помощью системных вызовов.

- Организация очереди из заданий в памяти и выделение процессора одному из заданий, т.е. планирование использования процессора.

- Сохранение содержимого регистров и структур данных

- Разработка стратегий управления памятью

- Разработка средств коммуникаций для обмена данными

- Организация хранения информации в виде файлов и обеспечение доступа к конкретным пользователям

- Разрешение конфликтных ситуаций при работе с различными ресурсами и разработка средств синхронизации

- Расширение систем мультипрограммирования стали системы разделения времени, в которых процессор переключается между задачами, не только на время операций ввода-вывода, но и просто по прошествии интервала времени. Переключения настолько частые, что пользователь может взаимодействовать со своими программами интерактивно.

- Параллельно внутренней эволюции - проходила и внешняя.

- Появляется идея создания семейств программно-совместимых машин, работающих под управление одной и той же ОС.

1. С 1980 года

ПК. Классические, сетевые и распределенные системы.

В основу элементной базы ложатся большие интегральные схемы, возрастает степень интеграции устройств, снижается стоимость микросхем, компьютер становится доступнее отдельному человеку, а не только корпорациям.

Так как компьютер используется обычным пользователем – это привело к “дружественному” программному обеспечению.

В середине 80-ых с развитием сетей появляются компьютеры под управлением сетевых или распределенных ОС.

В сетевых ОС пользователи могут получить доступ к ресурсам другого компьютера, но они должны знать об их наличии и уметь это сделать, каждая машина в сети работает под управлением своей локальной ОС имеющей дополнительные средства для работы с сетью.

Распределенная ОС – внешне выглядит как обычная автономная система, пользователь не знает, и не должен знать, где хранятся его файлы и выполняются программы, на локальной или удаленной машине.

**В процессе эволюции ОС выполняла шесть основных функций:**

- Планирование заданий и использования процессора

- Обеспечение программ средствами коммуникации и синхронизации

- Управление памятью

- Управление файловой системой

- Управление вводом-выводом

- Обеспечение безопасности

Каждая из приведенных функций обычно реализована в виде подсистемы, являющейся структурным компонентом ОС.

NEXT пара – ТЕСТ Классификации и история развития ОС

**Домашняя работа**

1. **По назначению:**

* Универсальные Debian
* Специализированные **(Be OS)**

1. **По способу загрузки:**

* Загружаемые Windows
* Постоянно находящиеся в памяти Kernel

**3) По особенностям алгоритмов управления ресурсами**

3.1) Многозадачность **(Unix, Windows, MS DOS)**

* Однозадачность MSDOS
* Многопрограммность Windows

3.2) Поддержка многопользовательского режима

* Однопользовательский **(MS DOS)**
* Многопользовательские **(Linux, Windows)**

3.3) Виды многопрограммной работы

* Не вытесняющая многопрограммность Net Ware
* Вытесняющая многопрограммность Linux

3.4) Многопроцессорная обработка

* Без поддержки мультипроцессирования Windows 7
* С поддержкой мультипроцессирования Windows 8,10

1. **По области использования и форме эксплуатации**

* Системы полетной обработки **(ОС ЕС)**
* Системы разделения времени (**FreeRTOS**)
* Системы реального времени **(RT 11)**

**Системы пакетной** обработки предназначены для решения задач вычислительного характера не требующих быстрого получения результатов.

Основным критерием по эффективности является их максимальная пропускная способность при максимальной загрузке всех ресурсов компьютера, поэтому в этих системах пользователь отстранен от компьютера.

**Системы разделения времени** обеспечивают удобства и эффективность работы пользователя, который может вести диалог с программой.

Системы предназначены для решения задач требующих быстрого получения результатов.

Система реального времени предназначена для управления техническими процессами (объектами) где существует предельное время на выполнение программ управления объектом

 Основной критерий эффективности таких систем – реактивность

По аппаратной платформе: (типу вычислительной техники)

* ОС для смарт карт **(THE)**
* Встроенные ОС **(Bios)**
* ОС для персональных компьютеров **(windows, macos)**
* ОС для мини ЭВМ **(Unix)**
* ОС для мейнфреймов **(Z/OS)**
* Серверные ОС **(windows server, novell netware)**
* Кластерные ОС **(Linux, forceBSD)**

**Основные принципы и концепции построения ОС**

1. Основные принципы

* Модульность – функционально законченная часть системы

ОС строится из множества программных модулей – функционально законченных элементов системы, выполненных в соответствии с принятыми межмодульными интерфейсами.

Модуль предполагает легкий способ его замены другим при наличии заданных интерфейсов.

Выделяют: **привилегированные**, **повторно входимые** и **реентерабельные модули**, так как они позволяют более эффективно использовать ресурсы вычислительной системы.

Свойства реентерабельности достигается за счёт механизмов динамического выделения памяти под переменные для нового вычислительного процесса (задачи).

***Программа называется реентерабельной***, если она разработана таким образом, что одна копия команд в памяти может быть совместно использована несколькими пользователями или процессами.

**Принцип модульности** отражает технологические и эксплуатационные свойства систем.

Является одним из основных в UNIX системах

**Принцип особого режима работы** - ядро ОС и низкоуровневые драйверы работают в специальном режиме работы процессора

**Причины**:

* 1. Повышается надежность выполнения вычислений
  2. Ряд функций должен выполняться исключительно централизованно под управлением ОС (чаще всего ввод-вывод)

**Принцип виртуализации** – используется практически в любой ОС

Пример – понятие виртуальной машины

Позволяет абстрагироваться от конкретных ресурсов, максимально обобщить их свойства и работать с некоторой абстракцией включающий в себя основные особенности.

**Главное достижение** принципа виртуализации – возможность выполнения в ОС приложений, разработанных для другой ОС.

Так же, одним из аспектов, является независимость программ от внешних устройств.

**Принцип мобильности**

**Мобильность** (переносимость) – возможность и легкость переноса ОС на другую аппаратную платформу.

**Мобильная ОС** – обычно разрабатывается с помощью специального языка высокого уровня, обеспечить переносимость достаточно сложно, так как архитектуры разных процессоров могут сильно различаться.

Для обеспечения мобильности создан стандарт на интерфейс прикладного программирования – **POSIX**

**POSIX –** интерфейс прикладного программирования для переносимых ОС.

**Принцип совместимости –** это способность ОС выполнять программы написанные для другой системы или для более ранних версий данной ОС

**Выделяют:** двоичную совместимость и совместимость на уровне исходных текстов приложений

**Двоичная совместимость** достигается если исполняемая программа запускается в другой ОС. Для этого необходимы совместимость на уровне команд процессора, совместимость на уровне системных вызовов и библиотечных вызовов

Совместимость на уровне исходных текстов требует наличия соответствующих трансляторов на уровне системного ПО и совместимости на уровне библиотек и системных вызовов.

Одним из средств обеспечения совместимости программных и пользовательских интерфейсов является соответствие стандартам **POSIX**

**Принцип генерируемости –** исходное представление центральной системной управляющей части ОС должно обеспечивать возможность настройки, исходя из конкретной конфигурации, конкретного вычислительного комплекса и круга решаемых задач.

**Генерация ОС** – это ее сборка (компоновка) из отдельных программных модулей. В результате генерации получаются скомпонованные двоичные коды ОС и построенные системные таблицы в которых отражаются конфигурация компьютера.

**Процесс генерации** выполняется с помощью специального генератора и соответствующего входного языка для этой программы.

Настройку процесса генерации упрощает принцип мобильности.

**Принцип открытости** (**расширяемости системы)** – этот принцип означает доступность ОС для анализа как пользователем, так и системным специалистом.

При изменении основная часть ОС может оставаться неизменной и добавляться новые службы или модифицироваться старые.

**Принцип обеспечения безопасности вычислений** – правила безопасности определяют такие свойства, как защита ресурсов одного пользователя от других и установление квот по ресурсам, для предотвращения захвата одним пользователем всех системных ресурсов.

Так же для поддержки принципа используется механизм учетных записей для обеспечения защиты информации от несанкционированного доступа.

Механизм предполагает проведение **аутентификации** пользователя при его регистрации на ПК и последующую авторизацию определяющий уровень полномочий пользователя. Каждая учетная запись может входить в одну или несколько групп.

**Концепции построения ОС**

**Монолитное ядро –** компоненты ОС являются не самостоятельными модулями, а составными частями одной большой программы.

**Монолитное ядро –** это набор процедур, каждое из которых может вызвать другую.

**Монолитное ядро –** это такая схема ОС все её компоненты, являются составными частями одной программы используют общую структуру данных, и взаимодействии друг с другом, путём непосредственного вызова процедур.

**Для монолитной ОС** ядро совпадает со всей системой.

Для многих ОС с монолитным ядром, сборка ядра выполняется отдельно для каждого компьютера, на котором устанавливается ОС.

При этом можно выбрать список оборудования и программных протоколов, поддержка которых будет включена в ядро.

Это старейший образ организации ОС.

**Многоуровневые системы.** В этих системах вся ее структура разбита на ряд более мелких уровней, с четко определенными связями между ними. Эти системы хорошо реализуются, тестируются и модифицируются, но они сложнее в разработке чем **монолитные**, и менее эффективные чем **монолитные**.

1968 год – **первая многоуровневая машина** (THE) – имела 6 уровней.

**Виртуальная машина.** Эмуляция реальной конфигурации компьютера.

При попытке обращения к виртуальному комплектующему устройству происходит системный вызов реальной ОС, которая и выполняет действия. Это позволяет каждому пользователю загрузить свою ОС на виртуальную машину.

**Недостаток:** менее эффективны реальных ОС

**Преимущества:** использование на одной системе программ, написанных для разных ОС.

**Микро-ядерная архитектура.** В таких ОС можно выделить центральный компактный модуль, относящийся к супервизорной части системы.

Модуль имеет очень небольшие размеры и выполняет небольшое количество управляющих функций.

**Микроядро** – это минимальная, главная часть ОС которая является основой модульной и переносимых расширений.

**Основная идея технологии микроядра** – это создание необходимой среды верхнего уровня иерархии, из которой можно легко получить доступ ко всем функциональным возможностям уровня аппаратного обеспечения.

**Микроядро** является стартовой точкой для создания всех остальных модулей системы, выполняющих сервисную функцию.

**Микроядро** обеспечивает только пять типов сервисов:

1. Управление виртуальной памятью
2. Поддержка заданий и потоков
3. Взаимодействие между процессами
4. Управление поддержкой ввода-вывода и прерываниями
5. Сервисы процессора

**Основное достоинство**: высокая степень модульности ядра ОС и упрощение процесса отладки компонентов ядра.

**Основная сложность**: необходимость тщательного проектирования

Пример – система реального времени **QNX**

**Смешанные системы.**

**Д\З КОНСПЕКТ НА ТЕМУ: “СЕТЕВЫЕ ОС”**

**Структура, понятия, сетевые службы и сервисы, достоинства и недостатки.**

**Сетевые ОС**

(Network Operating System – NOS) – это комплекс программ, обеспечивающих обработку, хранение и передачу данных в сети.

Сетевая операционная система выполняет функции прикладной платформы, предоставляет разнообразные виды сетевых служб и поддерживает работу прикладных процессов, выполняемых в абонентских системах. Сетевые операционные системы используют клиент-серверную, либо одно ранговую архитектуру. Компоненты NOS располагаются на всех рабочих станциях, включенных в сеть.

NOS определяет взаимосвязанную группу протоколов верхних уровней, обеспечивающих выполнение основных функций сети. К ним, в первую очередь, относятся:

адресация объектов сети;

функционирование сетевых служб;

обеспечение безопасности данных;

управление сетью.

При выборе NOS необходимо рассматривать множество факторов. Среди них:

набор сетевых служб, которые предоставляет сеть;

возможность наращивания имен, определяющих хранимые данные и прикладные программы;

механизм рассредоточения ресурсов по сети;

способ модификации сети и сетевых служб;

надежность функционирования и быстродействие сети;

используемые или выбираемые физические средства соединения;

типы компьютеров, объединяемых в сеть, их операционные системы;

предлагаемые системы, обеспечивающие управление сетью;

используемые средства защиты данных;

совместимость с уже созданными прикладными процессами;

число серверов, которое может работать в сети;

перечень ретрансляционных систем, обеспечивающих сопряжение локальных сетей с различными территориальными сетями;

способ документирования работы сети, организация подсказок и поддержек.

Функции и характеристики сетевых операционных систем (ОС).

Различают ОС со встроенными сетевыми функциями и оболочки над локальными ОС. По другому признаку классификации различают сетевые ОС одноранговые и функционально несимметричные (для систем “клиент/сервер”).

Основные функции сетевой ОС:

управление каталогами и файлами;

управление ресурсами;

коммуникационные функции;

защита от несанкционированного доступа;

обеспечение отказоустойчивости;

управление сетью.

Управление каталогами и файлами в сетях заключается в обеспечении доступа к данным, физически расположенным в других узлах сети. Управление осуществляется с по-мощью специальной сетевой файловой системы. Файловая система позволяет обращаться к файлам путем применения привычных для локальной работы языковых средств. При обмене файлами должен быть обеспечен необходимый уровень конфиденциальности обмена (секретности данных).

Управление ресурсами включает обслуживание запросов на предоставление ресурсов, доступных по сети.

Коммуникационные функции обеспечивают адресацию, буферизацию, выбор на-правления для движения данных в разветвленной сети (маршрутизацию), управление потоками данных и др. Защита от несанкционированного доступа — важная функция, способствующая поддержанию целостности данных и их конфиденциальности. Средства защиты могут раз-решать доступ к определенным данным только с некоторых терминалов, в оговоренное время, определенное число раз и т.п. У каждого пользователя в корпоративной сети могут быть свои права доступа с ограничением совокупности доступных директорий или списка возможных действий, например, может быть запрещено изменение содержимого некоторых файлов.

Отказоустойчивость характеризуется сохранением работоспособности системы при воздействии дестабилизирующих факторов. Отказоустойчивость обеспечивается применением для серверов автономных источников питания, отображением или дублированием информации в дисковых накопителях. Под отображением обычно понимают наличие в системе двух копий данных с их расположением на разных дисках, но подключенных к одному контроллеру. Дублирование отличается тем, что для каждого из дисков с копиями используются разные контроллеры. Очевидно, что дублирование более надежно. Дальнейшее повышение отказоустойчивости связано с дублированием серверов, что однако требует дополнительных затрат на приобретение оборудования.

Управление сетью связано с применением соответствующих протоколов управления. Программное обеспечение управления сетью обычно состоит из менеджеров и агентов. Менеджером называется программа, вырабатывающая сетевые команды. Агенты представляют собой программы, расположенные в различных узлах сети. Они выполняют команды менеджеров, следят за состоянием узлов, собирают информацию о параметрах их функционирования, сигнализируют о происходящих событиях, фиксируют аномалии, следят за трафиком, осуществляют защиту от вирусов. Агенты с достаточной степенью интеллектуальности могут участвовать в восстановлении информации после сбоев, в корректировке параметров управления и.т.п

**Структура**

1. В сетевой операционной системе отдельной машины можно выделить несколько частей.
2. Средства управления локальными ресурсами компьютера: функции распределения оперативной памяти между процессами, планирования и диспетчеризации процессов, управления процессорами, управления периферийными устройствами и другие функции управления ресурсами локальных ОС.
3. Средства предоставления собственных ресурсов и услуг в общее пользование – серверная часть ОС (сервер). Эти средства обеспечивают, например, блокировку файлов и записей, ведение справочников имен сетевых ресурсов; обработку запросов удаленного доступа к собственной файловой системе и базе данных; управление очередями запросов удаленных пользователей к своим периферийным устройствам.
4. Средства запроса доступа к удаленным ресурсам и услугам – клиентская часть ОС (редиректор). Эта часть выполняет распознавание и перенаправление в сеть запросов к удаленным ресурсам от приложений и пользователей. Клиентская часть также осуществляет прием ответов от серверов и преобразование их в локальный формат, так что для приложения выполнение локальных и удаленных запросов неразличимо.
5. Коммуникационные средства ОС, с помощью которых происходит обмен сообщениями в сети. Эта часть обеспечивает адресацию и буферизацию сообщений, выбор маршрута передачи сообщения по сети, надежность передачи и т.п., т. е. является средством транспортировки сообщений.

**Архитектура ОС**

ОС состоит из модулей.

Модуль – это функционально законченный элемент системы, выполненный в соответствии с принятыми межмодульными интерфейсами.

Все модули делят на две группы:

1. Модули выполняющие основные функции (ядро)
2. Модули выполняющие вспомогательные функции ОС

Ядро (kernel) – это часть ОС резидентная в оперативной памяти.

При начальной загрузке ядро считывается с диска в память начиная с нулевого адреса. При перенастройке системы ядро перестраивается и сохраняется в новом файле. Оно содержит системные программы, выполняющие управление ресурсами компьютера, управление памятью, доступом к диску и.т.п.

Ядро взаимодействует с аппаратной часть компьютера, изолируя прикладные программы от особенностей ее архитектуры. Ядро минимизировано и не выполняет ни одной функции непосредственно служащей пользователю.

**Компоненты ядра:**

1. Подсистема управления процессами

Процесс – это отдельная программа в стадии выполнения. Программа – это статический объект, представляющий собой файл с кодами и данными. Для каждого вновь создаваемого процесса ОС генерирует системные информационные структуры, которые содержат данные о потребностях процесса в ресурсах вычислительной системы, а так же о фактически выделенных ему ресурсов. Таким образом, процесс можно также определить, как некоторую заявку на потребление системных ресурсов. Чтобы процесс мог быть выполнен, ОС должна назначить ему область памяти, в которой будут размещены коды и данные процесса, а также представить ему процессорное время.

В мультипрограммных ОС одновременно могут существовать несколько процессор. Часть процессов порождается по инициативе пользователей (пользовательские), а другие инициализируются самой ОС для выполнения своих функций (системные).

Процессы могут одновременно претендовать на одни и те же ресурсы. В этом случае ОС поддерживает обслуживание очередей заявок к ресурсам.

Важной задачей ОС является защита ресурсов, выделенных одному процессу, от остальных процессов. Особенно тщательно защищаемым ресурсом является память. Память, отведённая под коды и данные процесса называется **адресным пространством.**

На протяжении периода существования процесса его выполнение может быть многократно прервано и продолжено. Для возобновления выполнения процесса, необходимо восстановить его среду.

Эта информация называется **контекстом** процесса (состояние регистров, коды ошибок, указатели на открытые файлы и пр.)

Для реализации сложных программных комплексов, часть необходимо организовать его выполнение в виде параллельных процессов и как следствие, ОС должна предоставить средства для межпроцессного взаимодействия.

Таким образом, подсистема управления процессами планирует выполнение процессов, т.е. распределяет процессорное время между несколькими одновременно существующими процессами, занимается созданием и уничтожением процессов, обеспечивает процессы необходимыми ресурсами, поддерживает синхронизацию процессов, а также обеспечивает взаимодействие между процессами.

1. Подсистема управления памятью

Память является для процесса таким же важным ресурсом, как и

процессор, так как процесс может выполняться процессором только в

в том случае, если его коды и данные находится в оперативной памяти.

Функции подсистемы управления памятью:

* + Распределение существующей в данной момент памяти, между процессами
  + Настройка виртуальных адресов на физические адреса
  + Загрузка кодов и данных процессов в отведенную память
  + Защита областей памяти каждого процесса

Одним из наиболее популярных способов управления памятью является виртуальная память. Наличие механизма виртуальной памяти позволяет программиста писать программы так, как будто в его распоряжении имеется оперативная память большого объема, часто превышающий объём имеющейся физической памяти. При размещении такой программы в памяти подсистема виртуальной памяти производит трансляцию виртуальных адресов, в физические.

Защита памяти – это способность предохранять выполняемую задачу от ошибочного обращения к памяти, назначенной другим.

1. Подсистема ввода-вывода и управления внешними устройствами

ОС экранирует сложность реальной аппаратуры от программиста. Обеспечивает поддержку широкого спектра драйверов, их динамическую загрузку и выгрузку. Программа, управляющая конкретной моделью внешнего устройства называется драйвером. Созданием драйверов занимаются как разработчики ОС, так и специалисты компаний, выпускающих внешние устройства.

1. Файловая система

Организует представление о данных, хранящихся на внешнем

Накопителе, в виде файла – неструктурированной последовательности байтов, имеющей символьное имя. Для удобства работы с файлами группируются в каталоги. Пользователь может с помощью ОС выполнять поиск файла по имени, удаление, вывод, изменение и сохранение.

1. Подсистема защиты данных и администрирования

* Отказоустойчивость ОС, защита сбоев и отказов аппаратуры и ошибок программного обеспечения
* Идентификация и аутентификация пользователей при входе в систему
* Разграничение прав доступа к данным
* Аудит ОС, фиксация всех событий
* Криптографические функции

1. Интерфейс прикладного программирования

Предназначен для использования прикладным программистом

Системных ресурсов компьютера и реализуемых ОС разнообразных системных функций и процедур. API описывает совокупность функций и процедур, принадлежащих ядру или надстройкам ОС.

Существует несколько способов вариантов реализации API:

* Реализация на уровне модулей ОС
* Реализация на уровне системы программирования
* Реализация на уровне внешней библиотеки

Как правило, не стандартизированы. В каждом конкретном

Случае набор вызовов API определяется, прежде всего, архитектурой ОС и ее назначением.

Обычно ядро оформляется в виде программного модуля некоторого специального формата, отличающегося от формата пользовательских приложений.

Поскольку некоторые компоненты ОС оформлены как обычные, т.е. в виде исполняемых модулей стандартного для данной ОС формата, то часто бывает очень сложно провести грань между ОС и приложениями.

**Файловая система** – это набор спецификаций (правил, протоколов) и соответствующее им ПО.

Они отвечают за создание, уничтожение, организацию, чтение, запись, модификацию и перемещение файловой информации.

А также за управлением доступа к файлам и за управление ресурсами.

**Термин файловая система** определяет принципы доступа к данным, организованным в файлы.

**Архитектура ОС**

**Ядро и модули**

Ядро оформляется в виде программного модуля некоторого специального формата.

Вспомогательные модули оформляются либо в виде приложений, либо в виде библиотек процедур.

Вспомогательные модули ОС делят на группы:

1. Утилиты – это программы решающие отдельные задачи управления и сопровождение компьютерной системы. (сжатие дисков, дефрагментаторы)
2. Системные обрабатывающие программы (текстовые, графические, компоновщики)
3. Программы предоставления доп.услуг.
4. Библиотеки процедур разного назначения упрощающие разработку приложений (библиотека мат. Функций функций ввода\вывода, и др)

Вспомогательные модули загружаются в оперативную память только на время своего выполнения, то есть являются транзитными.

Наиболее универсальными подходами к разработке ОС являются:

1. Модульная организация
2. Функциональная избыточность
3. Функциональная избирательность
4. Параметрическая универсальность
5. Концепция многоуровневой иерархической вычислительной системы
6. Разделение модулей на две группы по функциям (ядро и вспомогательные)
7. Разделение модулей ОС на две группы по размещению в памяти вычислительной системы (резидентные и транзитные)
8. Реализация двух режимов работы вычислительной системы – привилегированный режим (режим ядра, режим супервизора), пользовательский (режим задачи)
9. Ограничение функций ядра (а значит и количества модулей ядра), до минимального количества самых необходимых важных функций

В многослойной структуре принято выделять:

1. Средства аппаратной поддержки ОС – они участвуют в организации вычислительного процесса (Средства прерываний, средства поддержки привилегированного режима, средства поддержки виртуальной памяти, системный таймер, средство защиты памяти и др.
2. Машинно-зависимые модули ОС, в которых отражается специфика аппаратной платформы ПК. Он экранирует вышележащие слои ОС от особенностей аппаратуры.
3. Базовые механизмы ядра выполняют наиболее простые операции (переключение контекстных процессов, перемещение страниц между памятью и диском и.т.п)
4. Менеджеры ресурсов – выполняют стратегические задачи по управлению ресурсами вычислительной системы (менеджеры процессов ввода\вывода, оперативной памяти и файловой системы.)
5. Интерфейс системных вызовов – это верхний слой ядра ос, взаимодействует с приложениями и системными утилитами. – предоставляет доступ к ресурсам системы в удобной форме, без указаний деталей их физического расположения

Повышение устойчивости ОС обеспечивается переходом ядра в привилегированный режим. При этом происходит замедление выполнения системных вызовов. Системный вызов привилегированного ядра выполняет переключение процессора из пользовательского режима в привилегированный. А при возврате в приложение обратно к переключению.