04.09.2020

Зинаида Алексеевна

Литература – Батаев А.В. Операционные системы и среды (2017)

**История развития операционных систем**

Первый этап 1945-1955 год. Ламповые машины. Операционные системы отсутствовали.

Второй этап 1955-начало 60х. Полупроводниковые элементы – транзисторы. Происходит разделение персонала на программистов, операторов и специалистов по эксплуатации. Формируются задания на перфокартах, которые передаются вычислительной машине на выполнение. Для повышения производительности вычислительной машины формируется пакет заданий, которыми должна управлять управляющая программа. Появляются современные системы пакетной обработки, которые автоматизируют запуск одной программы из пакета за другой. Система пакетной обработки – прообраз современных ОС.

Третий этап начало 60х-1980 год. Интегральные микросхемы. На смену перфокартам приходят магнитные ленты – устройства последовательного доступа, информация считывается в том порядке, в котором была записана. На смену магнитным лентам приходят магнитные диски – устройства прямого доступа, информацию можно считывать в любом порядке, это приводит к тому, что можно планировать задания. Появляется мультипрограммирование – одна программа выполняется операцию ввода-вывода и процессор не простаивает.

Для реализации мультипрограммирования должна быть соответствующая аппаратная поддержка:

* Защитные механизмы памяти
* Наличие привилегированных и непривилегированных команд
* Наличие привилегированного пользовательского режима поддержки
* Наличие прерывания

Для реализации мультипрограммирования в операционную системы должны быть заложены механизмы:

* Системные вызовы
* Планирование заданий
* Сохранение контекста прерванной задачи
* Синхронизация и коммуникация процессов
* Системные вызовы

*Система разделения времени* – процессор переключается между задачами по истечению определённого интервала времени.

1969 год – разработка UNIX.

Четвертый этап. 1980 – наше время. Большие интегральные схемы (БИС). Появляются персональные компьютеры и классические сетевые распределенные операционные системы.

В 1983 году протокол TCP/IP стал стандартом.

07.09.2020

Назначение и функции операционных систем

Назначение ОС – организация вычислительного процесса

Операционная система – это совокупность программных средств, которые обеспечивают управление аппаратной частью компьютера, прикладными программами, а также взаимодействие их между собой и пользователем.

Задачи операционной системы:

* Увеличение пропускной способности (многозадачность)
* Уменьшение времени реакции системы на запросы пользователей
* Упрощение работы разработчиков, обслуживающего персонала (ПО, языки программирования, различные сервисные программы)

Функции ОС:

* Предоставление пользователю или программисту вместо реальной аппаратуры компьютера – расширенные виртуальные машины (высокоуровневые функции)
* Управление ресурсами (неразделяемые ресурсы, разделяемые ресурсы). Ресурс – любой физический и логический компонент системы. (логические компоненты – сетевые подключения, таблица выполняемых процессов; физические – процессор, оп).

Неразделяемые ресурсы - могут быть использованы на заданном отрезке времени только одним процессом. (файл, открытый на запись).

Разделяемые ресурсы – использование этого ресурса несколькими процессами.

Виды управлений:

* Управление процессами – создание, уничтожение процессов.
* Управление памятью – защищает адресное пространство процессов, выделяет память процессам, организует виртуальную память.
* Управление файлами (файловая система размещает данные на диске, что позволяет организовать работу диска)
* Управление внешними устройствами через драйверы
* Защита данных администрирования
* 1 рубеж защиты – login/password
* 2 рубеж защиты – аудит ОС (все события, происходящие в системе)
* Резервное копирование данных
* Пользовательский интерфейс (взаимодействие пользователя -ПК)

Требования к современным ОС

* Модульность
* Расширяемость – внесение изменений и дополнений без нарушения целостности системы
* Переносимость – код операционной системы должен легко переноситься с одного типа процессора на другой
* Совместимость – должна иметь средства для выполнения прикладных программ, написанных ля других операционных систем.
* Надежность и отказоустойчивость – должна быть защищена от внутренних и от внешних ошибок
* Высокая производительность

Классификация ОС

* 1 класс – по числу одновременно выполняемых задач. Однозадачные системы – в каждый момент времени может существовать не более, чем один пользовательский процесс, но одновременно с этим могут работать системные процессы. Многозадачные (MS DOS, Windows, Linux) – обеспечивают одновременное выполнение нескольких пользовательских процессов. Эта система решает проблему распределения ресурсов и конкуренции за эти ресурсы. В многозадачных системах отсутствуют защитные механизмы и средства коммуникации.
* 2 класс – по числу одновременно работающих пользователей.
* Однопользовательские – полный доступ ко всем системам компьютера.
* Многопользовательские – несколько пользователей могут одновременно использовать ресурсы компьютера. Многопользовательские системы являются многозадачными, но не наоборот.
* 3 класс – по типу доступа к компьютеру.

Системы пакетной обработки – формируется пакет, который предоставляется вычислительной машине, пользователь не взаимодействует с ПК.

Система разделения времени – переключение между задачи осуществляется по истечению определенного интервала времени.

Системы реального времени – должны обеспечить гарантированное время ответа на внешние события. Бывают мягкие и жесткие.

* Жесткие – недопустимость никаких задержек при никаких условиях. Бесполезность результатов при опозданиях. Катастрофа при задержке реакции. Цена опоздания бесконечно велика. QNX - ОС реального времени.
* Мягкие системы – за опоздание результатов приходится платить, но цена не слишком велика.
* 4 класс – по типу средств вычислительной техники для управления ресурсами которых предназначена ОС.
* Однопроцессорные ОС – все задачи выполняются на одном ПК
* Многопроцессорные ОС – команды выполняются на разных ПК. Параллельное выполнение команд.

Многопроцессорные ОС разделяются на ассиметричные и симметричные.

* Симметричные – на каждом процессе функционирует одно и тоже ядро, и задача может быть выполнена на любом процессоре.
* Ассиметричные – процессоры неравноправны. Существует главный процессор и подчиненные.
* Сетевые ОС – распределяют взаи
* Распределенные – распределяется работа по различным ПК
* 5 класс – специализированные. Бытовая техника, телефоны, машины.

Виды ОС: Dos, OS/2, Linux, Windows.

Архитектура персонального компьютера

Современные компьютеры построены на основе разработанных принципов Джона Фон Неймана:

* Принцип программного управления – программа состоит из набора команда, которые выполняется процессором автоматически друг за другом в определенной последовательности;
* Принцип однородности памяти – программы и данные хранятся в одних и тех же ячейках памяти;
* Принцип адресности – основная память состоит из пронумерованных ячеек и процессору в производный момент времени доступа любая ячейка.

Микропроцессор характеризуется быстродействием (кол-во операций, выполняемое процессором в секунду), разрядность (макс кол-во разрядов двоичного кода, которые могут передаваться и обрабатываться одновременно), архитектура – система команд, которые может выполнять микропроцессор, и кол-во ядер.

Микропроцессор состоит из:

* АЛУ - арифметико-логическое устройство, которое выполняет все арифметические и логические операции над числовой и символьной информацией;
* УУ – устройство управления, формирует и подает во все блоки компьютера в нужные моменты времени управляющие импульсы, которые поступают с генератора тактовой частоты.
* МПП – микропроцессорная память, служит на кратковременные хранения записи и выдачи информации, непосредственно используемой в вычислениях в ближайшее время работы компьютера. Строится на регистрах – быстродействующие ячейки памяти.
* Системная шина – обеспечивают сопряжение и связь всех устройств между собой. Обеспечивает 3 направления:
* Между микропроцессором (CPU) и основной памятью
* Между микропроцессором (CPU) и портами ввода-выводы
* Между основной памятью и портами ввода-вывода внешних устройств
* Включает в себя 3 шины:
* Шина адреса – выбор устройства или ячейки памяти, куда перемещаются или откуда считываются данные
* Шина данных – перемещаются данные между различными устройствами
* Шина управления – по ней передаются сигналы (чтения или записи), определяющие характер обмена информации.

ПЗУ (ROM) – предназначено для хранения информации и выдачи ее. Энергонезависимое. В ПЗУ хранятся: тестирующие программы, универсальный загрузчик.

ОЗУ (RAM) – энергозависимое. Предназначено для записи, считывания и хранения выполняемых программ и данных.

КЭШ – используются при обмене данными между процессором и оперативной памятью для компенсации разницы скорости обработки информации процессором и менее быстродействующей оперативной памятью.

VRAM – хранитпрервсвся изображение в закодированном виде

ВЗУ – служит для долговременного хранения данных

Порты ввода-вывода – служат для обеспечения обмена информацией компьютера с внешними устройствами

COM - последовательный порт, передаются электрические импульсы, несущие информацию в машинном коде следовательно, один за другим.

LPT – параллельный порт, одновременная передача 8 электрических импульсов, несущих информацию в машинном коде

USB – универсальная последовательная шина

08.09.2020

**Принцип построения операционных систем**

**Структуры операционных систем**

Модули ОС:

* Ядро – модули, выполняющие основные функции ОС
* Модули, выполняющие вспомогательные функции ОС

Базовые функции ядра ОС:

* Управление процессами
* Управление памятью
* Управления устройства ввода-вывода
* Обеспечивает решение задачи организации вычислительного процесса (переключение контекстов, загрузка/выгрузка страниц, обработка прерываний)
* Поддержка приложений, создание для них *прикладной программной среды* (открытие и чтение файла, вывод информации на дисплей и т.п.)

Ядро является резидентным – постоянно находится в оперативной памяти.

Приложения взаимодействуют с программным обеспечением с помощью интерфейса прикладного программирования.

*Вспомогательные модули* ОС являются транзитными (располагаются в оперативной памяти только на время выполнения своих функций), оформлены как обычные приложение и включают в себя:

* Архивирование информации
* Дефрагментация данных на диске
* Поиск необходимого файла и т.п.

К вспомогательным модулям относятся:

* Утилиты – решают отельные задачи управления и сопровождения ОС
* Системные обрабатывающие программы – текстовые, графические редакторы, калькуляторы и т.д.
* Программы предоставления пользователю дополнительных услуг – специальный вариант пользовательского интерфейса
* Библиотеки процедур – модули различного назначения, упрощающие разработку приложений

*Вспомогательные модули* обращаются к функциям ядра ОС посредством системных вызовов

Процессор поддерживает 2 режима работы: пользовательский (user mod) и привилегированный (kernel mod)

Ядро работает в привилегированном режиме, вспомогательные модули и приложения работают в пользовательском режиме

Типы ядер ОС:

* Монолитное ядро – все компоненты ОС работают в одном адресном пространстве, все компоненты ядра являются составными частями одной программы, используют общие структуры данных и взаимодействуют друг с другом путем непосредственного вызова процедур. Ядро совпадает со всей системой. (UNIX)

Достоинства:

* скорость работы

Недостатки

* сбой в одном из компонентов ядра приводит к нарушению работоспособности системы;
* при добавлении новых компонентов системы требуется перекомпиляция ядра.
* Модульное ядро – современная усовершенствованная модификация архитектуры монолитных ядер ОС.

Отличия: не требуется перекомпиляция при изменении состава.

* Микроядро – ядро минимизировано. Выполняет только основные функции управления процессами и минимальный набор для работы с оборудованием. Большинство компонентов ОС оформлены в виде приложений, которые называются сервисами и работают в пользовательском режиме

Достоинства:

* Устойчивость к сбоям оборудования, ошибками, компонентам системы
* Высокая степень модульности ядра ОС
* Загружать и выгружать новые драйверы не перезапуская систему
* Надежность

Недостатки:

* Передача данных между процессами требует накладных расходов
* Экзо ядро – модифицированная структура микроядра. Набор основных компонентов ядра минимизирован, все выполняется как пользовательские процессы.
* Нано ядро – выполняет лишь одну задачу – обработку аппаратных прерываний (keykos)
* Гибридное ядро – сочетает в себе особенности монолитных и микро-ядер. Основные компоненты помещаются ядро, а все остальные оформлены в виде приложений и выполняются в пользовательском режиме.

**Архитектуры ОС**

* ОС на основе монолитного ядра
* Многослойная структура (многоуровневая) – нижний слой (аппаратура), средний слой (ядро), верхний слой (утилиты, приложения). Пользовательский режим. Вся система разбивается на слои и нижележащий слой является основой для вышележащего слоя. Достоинства: хорошо тестировать и модифицировать. Недостатки: Сложно определить функции слоев.

Структура ядра:

* 1 слой *средства аппаратной поддержки* - аппаратное устройства компьютера, которое непосредственно учувствует в вычислительном процессе;
* 2 слой *машинно-зависимые* - отражают специфику аппаратной платформы, этот слой экранизирует верхние слои от особенностей аппаратуры, обеспечивает переносимость с одной платформы на другую (BIOS);
* 3 слой *базовые механизмы ядра* - не принимают решения распределения ресурсов, только обрабатывают принятые решения;
* 4 слой *менеджер ресурсов* – управление ресурсами вычислительной системы
* 5 слой *интерфейс системных вызовов* – взаимодействие с приложениями, с различными системными утилитами, образуя интерфейс прикладного программирования.
* Микро-ядерная архитектура ОС – в привилегированном режиме работает небольшая част ОС – микроядро.

Микроядро защищено от остальных частей ОС и приложений, в состав микроядра сходят машинно-зависимые модули, а также модули, выполняющие базовые функции ядра по управлению процессами, обработки прерываний, управлению виртуальной памятью, пересылки сообщений, управление устройств ввода вывода. Все остальные высокоуровневые функции ядра оформляются в виде приложений, работающих в пользовательском режиме. Взаимодействие между компонентами и приложениями ОС осуществляется путем передачи сообщений через микроядро.

Компоненты ОС, выполняющиеся в пользовательском режиме, называются серверами ОС. Их задача обслуживание запросов локальных приложений и других модулей ОС.

Клиент, которым может быть либо прикладная программа, либо другой компонент ОС запрашивает выполнение некоторой функции у соответствующего сервера, посылая ему сообщения

Достоинства:

* Переносимость
* Расширяемость
* Надежность
* Поддержка распределённых вычислений

Недостаток – производительность ниже, чем в классической системе, выполнение системного вызова сопровождается 4-мя переключениями, а в классической ОС только 2-мя.

* Смешанная ОС. Большинство современных ОС использует различные комбинации этих подходов (монолитную, микро-ядерную). Linux – монолитная система с элементами микро-ядерной архитектуры.

**Виды прерываний. Обработка прерываний**

*Прерывание* – это ситуация, приводящая к временному или окончательному прекращению выполнения последовательности команд одной и переходу к выполнению команд другой программы.

Классификация прерывания:

* Внешние (аппаратные) – в результате поступления сигналов от аппаратных устройств. Они являются асинхронными.
* Внутренние (исключения) – происходят синхронно выполнению программы при появлении аварийной ситуации в ходе исполнения некоторой инструкции программы. Процессор при переходе на выполнение исключительной ситуации сохраняет часть своего состояния перед выполнением текущей команды.

Исключительные ситуации возникают синхронно с работой процессора, но непредсказуемо для программиста, если только тот специально не заставил процессор делить некоторое чисто на ноль.

Прерывания бывают *исправимые* и *неисправимые.* К *исправным* относятся исключительно ситуации, как отсутствие нужной информации в оперативной памяти. *Неисправные* исключительные ситуации обычно возникают в результате ошибок в программах. Обычно операционная система реагирует на такие ситуации завершением программы, вызывавшей исключительную ситуацию.

* Программные (системные вызовы) – когда одна программа требует сервис со стороны другой программы. Программные прерывания – это удобный способ вызова процедур ОС.

*Аппаратные прерывания* обрабатываются драйверами

*Внутренние прерывания* обрабатываются системными модулями ядра

*Программные прерывания* обрабатываются процедурами ОС, обслуживающими системные вызовы.

Способы выполнения прерываний:

* Векторный – в процессор передается номер вызываемой процедуры обработки прерывания
* Опрашиваемый – процессор вынужден последовательно опрашивать потенциальные источники запроса прерывания

Каждому возможному *прерыванию процесса* соответствует некоторый фиксированный адрес физической оперативной памяти.

*Каждое прерывание* имеет свой номер и с ним связана определенная подпрограмма.

*Программа прерывания* – это адрес ячейки памяти, где находится программа по обработке данного прерывания

Если произошла прерывание, то:

* управление передается ОС;
* ОС запоминает состояние прерванного процесса;
* ОС анализирует тип прерывания и передает управление соответствующей программе обработки этого прерывания;
* После обработки прерывания процессор возвращается к выполнению прерванного процесса либо начинает обслуживать процесс с наивысшим приоритетом.

10.09.2020

**Структура сетевой и распределенной ОС**

Сетевые ОС – комплекс программных модулей, предназначенный для повышения эффективности аппаратных ресурсов компьютера путем рационального управления его ресурсами и разделения ресурсов между множеством выполняемых в сети процессов.

К сетевым возможностям можно отнести:

* Поддержка сетевого оборудования
* Поддержка сетевых протоколов
* Поддержка протоколов маршрутизации
* Поддержка фильтрации сетевого трафика
* Поддержка доступа к удаленным ресурсам по сети

Основные компоненты сетевой ОС:

* Средства управления локальными ресурсами компьютера
* Сетевые средства:
* Серверная часть
* Клиентская часть
* Транспортные средства

Сетевая служба – совокупность серверной и клиентской частей ОС, предоставляющих доступ к конкретному типу ресурса компьютера через сеть

Сервис – интерфейс между потребителем услуг и поставщиком услуг

Подходы к построению сетевых ОС:

* Сетевые службы глубоко интегрированы в ОС
* Сетевые службы объединены в виде некоторого набора программные модулей оболочки
* Сетевые службы разрабатываются и поставляются в виде отдельных программные модулей

Типы сетевых ОС:

* В зависимости от распределения функций между компьютерами, они могут выступать в роли выделенного сервера или клиентского узла

Сеть может быть построена по следующим схемам:

* На основе компьютеров, совмещающих функции клиента и сервера – одно ранговая сеть
* На основе клиентов и сервисов – сеть с выделенными серверами
* Сеть, включающая узлы разных типов – гибридная сеть.

В сетях с выделенными серверами используются серверные ОС для серверов. Отличия серверных ОС от одно ранговых ОС:

* Поддержка мощных аппаратных платформ
* Поддержка большого числа одновременно выполняемых процессов
* Мощные средства для администрирования сети
* Сложный набор сетевых служб

Для передачи сообщений по сети используется протокол. (TCP/IP)

Для преобразования ip-адреса в символьное имя и обратно используется служба доменных имен – DNS

**Интерфейс**

Интерфейс – система связи и взаимодействие устройства компьютера. Также средство взаимодействие пользователей с ОС компьютера.

Интерфейс пользователя – элементы и компоненты программы, которые способны оказать влияние на взаимодействие пользователя с программным обеспечением:

* Средства изображения информации
* Командные режимы
* Устройство и технологии ввода данных
* Диалоги
* Взаимодействие между пользователем и компьютером
* Обратная связь с пользователем

Виды пользовательского интерфейса:

* В период пакетной обработки заданий, интерфейс реализовывался с применением специальных носителей информации – перфокарт. Интерфейс командой строки (CLI)
* Взаимодействие пользователя с компьютером осуществляется путём ввода с клавиатуры команд
* В современных операционных системах командный режим обеспечивается с помощью эмуляторами. (cmd.exe(32), Command.com (16), exterm (linux))
* Для операционных систем с алфавитно-цифровым интерфейсом разрабатывались специальные оболочки (NC.EXE , VC.COM)

Графический интерфейс – характерно управление с помощью указывающего устройства. яя  
Интеллектуальный интерфейс – приближен к человеческой форме общения.

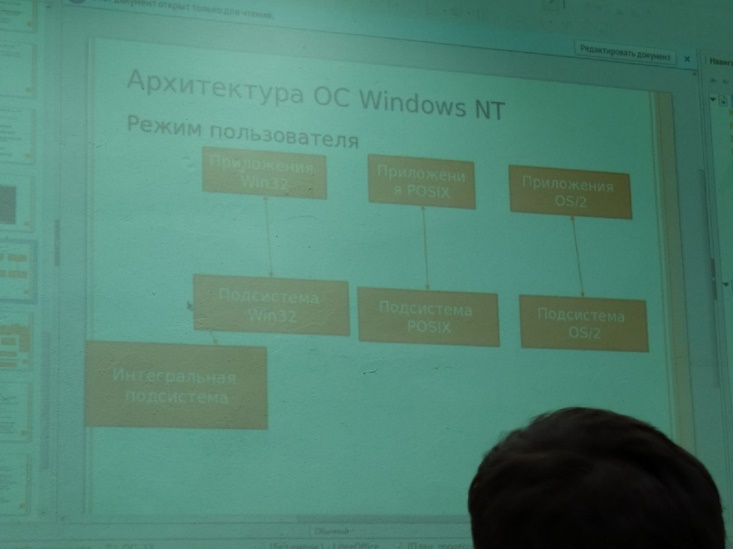
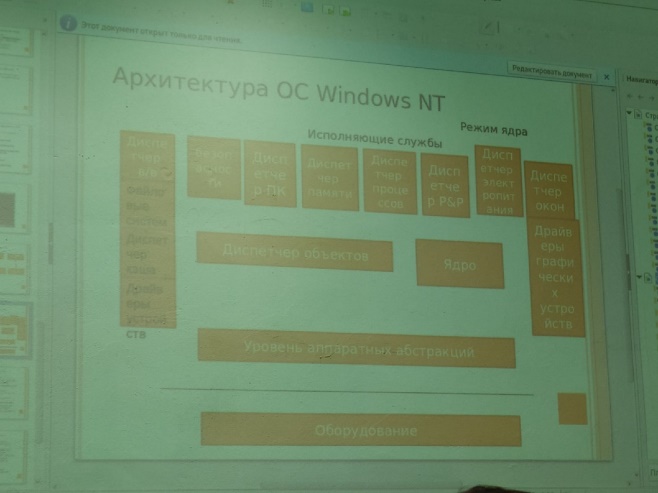
Программный интерфейс – интерфейс прикладного программирования

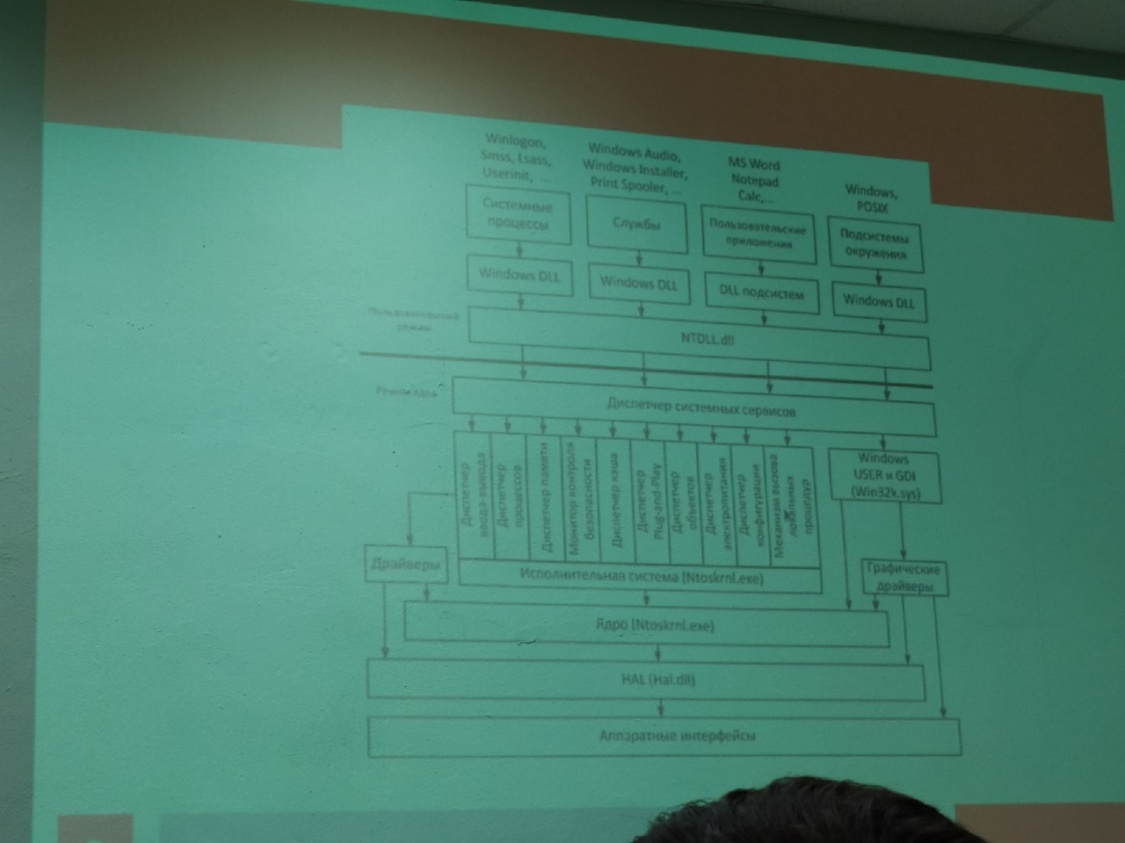
При написании приложений программист пользуется набором сервисных функции ОС, которые называются интерфейсом прикладного программирования. От пользователя эти функции скрыты за оболочкой алфавитно-цифрового или графического пользовательского интерфейса. Приложение выполняет обращение к функциям IP с помощью системных вызовов.

*Операционное окружение* – это среда, в которой пользователь запускает программу.

11.09.20

**Архитектура ОС Windows**

Режим пользователя:



В пользовательском режиме запускаются приложения пользователя, некоторые компоненты ОС. Это непривилегированный режим работы, приложение напрямую не могут обратиться к аппаратуре, только через исполняющие службы

Подсистема Win32позволяет запускать приложения 16, 32 разрядные приложения виндус и ДОС

POSIX приложение запускается для UNIX

Интегральная подсистема – управляет процессом регистрации

Режим ядра осуществляет доступ к данным системе и оборудования. Работает 3 службы: управляющие программы, ядро, уровень аппаратных абстракций

Ядро – планирует все системные операции, организует очереди данных, направляет их к микропроцессору и обрабатывает данные, отсылая их по соответствующему маршруту. Ядро планирует и управляет выполнением потоков согласно приоритетам, отвечает за синхронизацию нескольких процессов. Уровень аппаратных абстракций обеспечивает переносимость системы.

Диспетчер памяти отображает адреса виртуальной памяти в физические страницы ОЗУ. Диспетчер Plug and Playдля настройки параметров работы. Диспетчер электропитания контролирует питание на уровне ОС

Диспетчер окон и интерфейс графических устройств управляет системой окон, отвечает за отображение графики.

Диспетчер объектов реализует правила для сохранения, защиты и наименования объектов.

14.09.2020

hall.dll – уровень аппаратных абстракций.

Структура linux:

1. Пользователи
2. Стандартное ПО
3. Стандартная библиотека
4. ОС UNIX
5. Аппаратура

**Процессы**

Процесс – минимальный программный объект, обладающий собственными системными ресурсами.

Программа – это план действий, а процесс - это само действие

Понятие процесса включает:

* Программный код
* Данные
* Содержимое стека
* Содержимое адресного и других регистров процесса

Классификация процессов:

* По временным характеристикам: интерактивные, пакетные, реального времени
* По геологическому признаку: родительские и дочерние
* По времени развития: последовательные, параллельные
* По принадлежности к ОС: системные и пользовательские
* По связности: взаимосвязанные, изолированные, конкурирующие

Для управления процессами ОС используют системные данные, которые существуют в течение всего времени выполнения процесса. Совокупность этих данных называет *контекстом процесса*. *Контекст* определяет состояние процесса в заданный момент времени.

Для возобновления прерванного процесса необходимы системные данные (состояние регистров, программного счетчика, режим работы процессора и так далее).

Контекст процесса включает в себя:

* Пользовательский контекст – содержимое памяти, данных, стека, буферов ввода-вывода
* Регистровый контекст – содержимое аппаратных регистров
* Контекст системного уровня – структура данных ядра, связанные с процессом, которые характеризуют процесс.

Дескриптор процесса – системные данные, которые используются ОС для идентификации процесса.

Каждый процесс в ОС представлен в ОС набором данных, называемых таблицей управления процессом, блоков управления процессом.

Информация, хранящаяся в БУП:

* Текущее состояние процесса
* Уникальный идентификатор процесса
* Указатели памяти процесса
* Указатели выделанных ресурсов для процесса
* Область сохранения регистров, где хранится точка остановки процессора
* Приоритет процесса

Команда ps aux позволяет посмотреть процессы в linux

Операции над процессами:

* Создание и удаление процессов;
* присвоение имени процессу;
* включение этого процесса в список имен процессов, известных системе; назначение начального приоритета процессу;
* формирование БУП;
* выделение процессу начальных процессов
* Удаление процесса – удаление его из системы, при этом возвращаются ресурсы ОС, имя процесса удаляется из системных списков, таблица управления процессами освобождается и передается другому процессору
* Возобновление процесса – это операция по подготовке процесса начать выполнение с той точки, в которой он был прерван
* Изменение приоритета процесса
* Блокирование процесса

При запуске системы создается несколько процессов, некоторые из являются высокоприоритетными, а некоторые являются фоновыми, они не связаны с конкретным пользователем, но выполняют особые функции. Они активизируются только в определённый момент по мере выполнения задач и называются демонами.

В unix подобных операционных системах новый процесс создается с помощью системного вызова fork(). Этот запрос создает дубликат, вызываемого процесса, дочерние родительские процессы являются идентичными, и отличаются только идентификатором. Затем дочерний процесс выполняется системный вызов для изменения образа памяти и запуска новой программы

В Windows для создания процесса и запуска программы используется CreateProcess. После создания нового процесса для создание нового процесса родительские дочерние процессы имеют собственный адресные пространства, но дочерний процесс может использует одинаковые ресурсы с родительским процессом.

Каждый новый созданный процесс является дочерним предыдущему. Дочернему достаются от родительского процесса переменные, регистры. После вызова fork, как только родительские данные скопированы, последующее изменение в одном из процессов не влияет на другой.

Завершение процесса – в windows (exit), linux (Exitprocess)

Уничтожение процесса kill (unix), terminateProcess (Windows)

При завершении освобождаются все используемые системные ресурсы, завершаются дочерние процессы.

Процесс может находиться в состоянии зомби, если он не занимает процессорное время, не освобождает структуры ядра, в строке таблица процесса он остается.

Процесс может быть сиротой, если родительский процесс завершиться раньше дочернего, и он автоматически усыновляется процессом unit.

Процесс может быть в состоянии сна, которое невозможно прервать. Характеризуется тем, что не реагирует на системные вызовы и уничтожается только после перезагрузки системы.

События, приводящие к завершению процесса:

* Плановые завершения
* Выход по известной ошибке
* Выход по неисправной ошибке
* Уничтожение другим процессом

Состояние процесса:

* Готовность
* Выполнение
* Блокировка (ожидание)

**Понятие потоков**

Процесс состоит из потоков.

Поток – это последовательность команд процесса, которые выполняются независимо друг от друга и используют общие ресурсы одного процесса.

ОС является многопоточным и многозадачным. Все потоки в рамках одного процесса имеют одно и то же адресное пространство.

Классификация потоков:

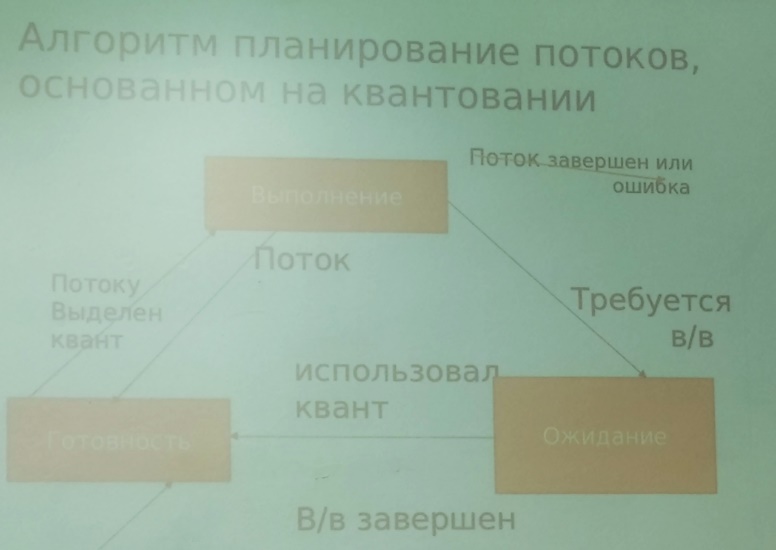
* По многозадачной модели – вытесняющая многозадачность и не вытесняющая/кооперативная многозадачность
* И по уровню реализации

**Невытесняющие алгоритмы** – активный поток выполняется до тех пор, пока он сам, по собственной инициативе, не отдаст управление ОС

**Вытесняющие алгоритмы** – решение о переключении процессора с одного потока на выполнение другого потока принимается ОС

Алгоритмы планирования - каждому потоку поочерёдно для выполнения предоставляется ограниченный непрерывный период процессорного времени – квант

Смена активного потока происходит, если:

* Потом завершился и покинул систему
* Произошла ошибка
* Поток перешел в состояние ожидания
* Исчерпан квант процессорного времени, отведенный данному потоку

Планировщик создает две очереди готовых потоков:

* 1-ая пришла в состояние готовности в результате исчерпания кванта
* 2-ая – потоки завершили ввод/вывод. Сначала будет просматриваться вторая очередь и, если она пуста, то квант будет выделяться из первой очереди.

Алгоритмы планирования, основанные на приоритетах

Приоритет – это число, характеризующее степень привилегированности потока при использовании ресурсов вычислительной машины

Чем выше приоритет, тем выше привилегии. Чем выше привилегии потока, тем меньше времени он будет проводить в очередях.

Приоритет процесса назначается ОС при его создании

Приоритет может изменятся в течении жизни потока:

* Либо по инициативе самого потока, когда он обращается с соответствующим вызовом к ОС
* Либо директивно администратором системы в зависимости от важности работы
* Либо назначаются самой ОС в зависимости от ситуации в системе, вэ том случае их называют **динамическими**
* Но приоритет может оставаться **фиксированным** на протяжении всей жизни потока
* Приоритет потока связан с приоритетом процесса, в рамках которого выполняется данный процесс.

Виды приоритетного планирования

Обслуживание с относительными приоритетами – выбирается поток с наивысшим приоритетом и активный поток выполняется до тех пор, пока он сам не покинет процессор

Обслуживание с абсолютными приоритетами – выбирается поток с наивысшим приоритетом, если в очереди готовых потоков появился с более высоким приоритетом, то прерванный поток переходит в состоянии готовности

**Реализация потоков**

Смешанные алгоритмы планирования

В основе планирования лежит квантование, но величина кванта и/или порядок выбора потока из очереди готовых определятся приоритетами потоков

В ОС windows реализовано планирование, основанное на квантовании, но сочетается с динамическими абсолютными приоритетами.

Есть 2 основных места реализации набора потоков пользовательском пространстве и в ядре

Реализация набора потоков в пользовательском пространстве. Весь набор потоков находится в одном пространстве. Об этом ядру ничего не известно

Использование потоков на уровне пользователя применимо для любой ОС

Достоинства:

* Для управления потоками процессу не нужно переключаться в режим ядра
* Использование потоков на уровне пользователя применимо для любой ОС
* Набор процедур, работающих на уровне приложения

Недостатки:

при выполнении ОС системного вызова блокируется не только данный поток, но и другие потоки данного процесса.

Так как ядро закрепляется за каждым процессом только 1 процессор то несколько потоков 1 процесса не могут выполняться одновременно даже в случае многопроцессорной системы.

Реализация потоков в ядре: нет таблицы потоков в каждом процессе, у ядра есть таблица потоков в которой отслеживаются все потоки, имеющиеся в системе.

Недостаток: тратится время на переключение между потоками внутри одного процесса за счет переходов в режим ядра

Гибридная реализация. Часть потоков работает на уровне ядра, а часть на уровне пользователя

**Взаимодействие процессов**

Заключается в передаче данных между процессами или совместном использовании ресурсов

Синхронизация заключается в согласовании скоростей процессов путем приостановки процесса до наступления некоторого события и последующей его активизации при наступлении этого события.

Многократное выделение процессу процессора называется диспетчеризация.