# Тема: «Операционные системы»

***Операционная система*** — программа, которая обеспечивает возможность рационального использования оборудования компьютера удобным для пользователя образом.

**Операционная система, как виртуальная машина**

При разработке ОС широко применяется абстрагирование, которое является важным методом упрощения и позволяет сконцентрироваться на взаимодействии высокоуровневых компонентов системы, игнорируя детали их реализации. Таким образом операционная система представляется пользователю виртуальной машиной, с которой проще иметь дело, чем непосредственно с оборудованием компьютера.

**Операционная система, как менеджер ресурсов**

ОС предназначена для управления всеми частями весьма сложной архитектуры компьютера. Таким образом операционная система, как менеджер ресурсов осуществляет упорядоченное и контролируемое распределение процессоров и других ресурсов между различными программами.

**Операционная система, как защитник пользователей и программ**

Если вычислительная система допускает совестную работу нескольких пользователей, то возникает проблема организации их безопасной деятельности. Необходимо обеспечить сохранность информации на диске, чтобы никто не смог удалить или повредить чужие файлы. Кроме того, нельзя разрешать программе одних пользователей произвольно вмешиваться в работу программ других пользователей. Всю эту деятельность совершает ОС.

**Операционная система, как постоянно функционирующее ядро**

*Операционная система* — это программа, которая постоянно работает на компьютере и постоянно взаимодействует со всеми прикладными программами. Однако, во многих современных ОС постоянно работает лишь часть её, которую принято называть ядром.

*Выделим основные функции, которые выполняют классические операционные системы:*

1. *Планирование заданий и использование процессора.*
2. *Обеспечение программ средствами коммуникации и синхронизации.*
3. *Управление памятью.*
4. *Управление файловой системой (FAT32, NTFS).*
5. *Управление вводом/выводом.*
6. *Обеспечение безопасности.*

**Системный вызов**

В любой ОС есть механизм, который позволяет обращаться к ядру операционной системы. *Системный вызов* – интерфейс между ОС и программой. Они [системные вызовы] создают, удаляют и используют различные объекты, главные из который – процессы и файлы.

***Файл***– поименованная последовательность битов, находящаяся в некоторой области памяти.

Пользовательская программа запрашивает сервис у ОС, осуществляя системный вызов. Имеются библиотеки процедур, которые загружают машинные регистры определёнными параметрами и осуществляют прерывание процессора. После этого управление передаётся обработчику данного вызова, который входит в ядро ОС.

Цель таких библиотек – сделать системный вызов похожим на обычный вызов от программы. Для того, чтобы обработка данного вызова была более приоритетной.

Системные вызовы так же называются программными прерываниями. Системный вызов обрабатывается в привилегированном режиме или режиме ядра. *Системный вызов (программное прерывание)* – синхронное событие, которое может повторяться при выполнении одного и того же программного кода.

**Прерывания**

Прерывание (hardware interrupt) – это событие, генерируемое внешним (по отношению к процессору) устройством. Посредством аппаратных прерываний центральный процессор информируется о том, что произошло какое-либо событие, требующее немедленной реакции (пользователь нажал клавишу, щёлкнул мышкой и т.п.). Или аппаратура сообщает о завершении асинхронной операции ввода/вывода (принтер закончил печать, закончено чтение данных с диска в основную память и т.д.).

Запрос к базе данных, обновление ОС

Важный тип аппаратных прерываний *– прерывание таймеров*, с помощью который ОС осуществляет планирование процессов. *Аппаратное прерывание* – асинхронное событие, оно возникает вне зависимости от того, какой код исполняется процессором в данный момент.

**Исключительные ситуации**

— это события, возникающие в результате попытки выполнения программой команды, которая по каким-то причинам не может быть выполнена до конца. Например:

* попытка доступа к ресурсу при отсутствии достаточных привилегий;
* обращение к отсутствующей странице памяти;

Исключительные ситуации являются синхронными событиями, которые возникают в контексте выполняемой задачи. Они делятся на исправимые и неисправимые.

**Файловые системы**

Главная задача **файловой системы** – скрыть особенности ввода/вывода и дать

программисту простую абстрактную модель файлов независимых от устройств. Все

операции с файлами реализуются с помощью системных вызовов.

# Архитектуры ОС

***Монолитное ядро****.* Представляет собой набор процедур, каждая из которых может вызвать каждую. Все процедуры работают в привилегированном режиме. Таким образом, *монолитное ядро* — это такая схема ОС, при которой все её компоненты являются составными частями одной программы, используют общие структуры данных и взаимодействуют с друг другом путём непосредственного вызова процедур.

Для *монолитной ОС* ядро совпадает со всей системой.Ядро является единой программой и перекомпиляция — это единственный способ добавить в него новые компоненты или исключить неиспользуемые.

***Микроядерная*** *архитектура.* Это такая архитектура, при которой, когда большинство составляющих ОС являются самостоятельными программами. В этом случае взаимодействие между ними обеспечивает специальный модуль ядра, называемый *микроядром*. *Микроядро* работает в привилегированном режиме и обеспечивает:

1. взаимодействие между программами,
2. планирование использования процессора,
3. первичную обработку прерываний,
4. операции ввода/вывода,
5. базовое управление памятью.

Основное достоинство данной архитектуры – это высокая степень модульности ядра, которая упрощает добавление в него новых компонентов. В такой ОС, можно не прерывая её работы загружать и выгружать новые драйверы, файловые системы и т.д.

Минус — вносятся дополнительные накладные расходы, связанные с передачей сообщений, что влияет на производительность системы.

***Многоуровневая (слоёная) архитектура.***

5. Интерфейс пользователя.

4. Управление вводом-выводом.

3. Драйвер устройства связи оператора и консоли.

2. Управление памятью.

1. Планирование задач и процессов.

0. Hardware.

***Многоуровневая*** *архитектура.* Идея в том, чтобы разбить всю вычислительную систему на ряд более мелких уровней с хорошо определёнными связями между ними. Так, чтобы объект уровня N могли вызывать только объекты уровня N – 1. Чем ниже уровень, тем более привилегированные команды и действия может выполнять модуль.

Первая подобная система была реализована Эдсгером Дейкстра в 1968г. в городе Эйндховен. Она называлась THE (Technical Highschool of Eindhoven).

*Плюсы:*

* Такие системы хорошо реализуются.
* Слоёные системы хорошо тестируются.
* Многоуровневые системы хорошо модифицируются.

*Минусы:*

* *Существуют серьёзные накладные расходы, когда нужно пройти от верхнего уровня до нижнего.*
* *Уделяется много времени и внимания правильному распределению функционала между уровнями.*

***Виртуальная машина.*** Если ОС реализует виртуальную машину для каждого пользователя, то такая виртуальная машина предстаёт как копия всего hardware в вычислительной системе, включая процессор, привилегированные и непривилегированные команды. Устройства ввода/вывода, прерывания и т.д. При попытке обратиться к такому виртуальному железу на уровне привилегированных команд в действительности происходит системный вызов реальной операционной системы, которая и производит все необходимые действия. Такой подход позволяет каждому пользователю загрузить собственную ОС на виртуальную машину и делать с ней всё, что захочет.

Недостатком виртуальной машины является снижение эффективности и, как правило, они очень громоздкие.

***Смешанная(гибридная) система.*** В большинстве случаев современные ОС использует различные комбинации вышеперечисленных подходов. Например, ядро ОС Linux представляет собой *монолитную* систему с элементами *микроядерной* архитектуры. При компиляции ядра можно разрешить динамическую загрузку и выгрузку очень многих компонентов ядра.

**Понятие процесса**

Программа в процессе исполнения является динамическим, активным объектом. Для выполнения программы ОС должна:

1. Выделить определённое кол-во оперативной памяти.
2. Закрепить за ней определённые устройства ввода/вывода или файлы.

***Процесс —*** *это* некоторая совокупность набора исполняющихся команд, ассоциированных с ним ресурсов, и текущего момента его выполнения, находящуюся под управлением ОС.

Вход 🡪 [Процесс не исполняется] 🡨выбран для исполнения/приостановка 🡪 [Процесс исполняется] 🡪 выход

(Рождение) 🡪допуск к планированию (Ожидание) 🡪 событие произошло (Готовность) 🡨прерывание/выбран для исполнения 🡪 (Исполнение) 🡪 завершение работы (Закончил выполнение)/ожидание события🡪 (Ожидание)

*Операции над процессами.*

* Создание процесса – завершение процесса.
* Приостановка процесса (перевод из исполнения в готовность) – запуск процесса (из состояния готовность в исполнение).
* Блокировка процесса (перевод из исполнения в ожидание) – разблокирование процесса (перевод из ожидания в исполнение).
* Изменение приоритета процесса.

Операции и процессы могут быть одноразовыми и многоразовыми.