
Операционные системы 2025



Введение в операционные системы


лектор — доцент
ПОВТАС БГТУ им. В.Г. Шухова
Островский Алексей Мичеславович



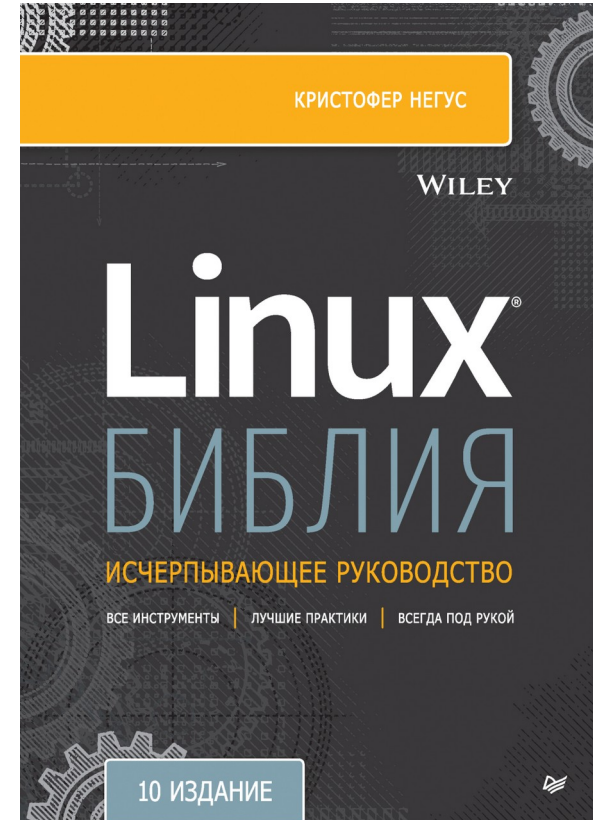
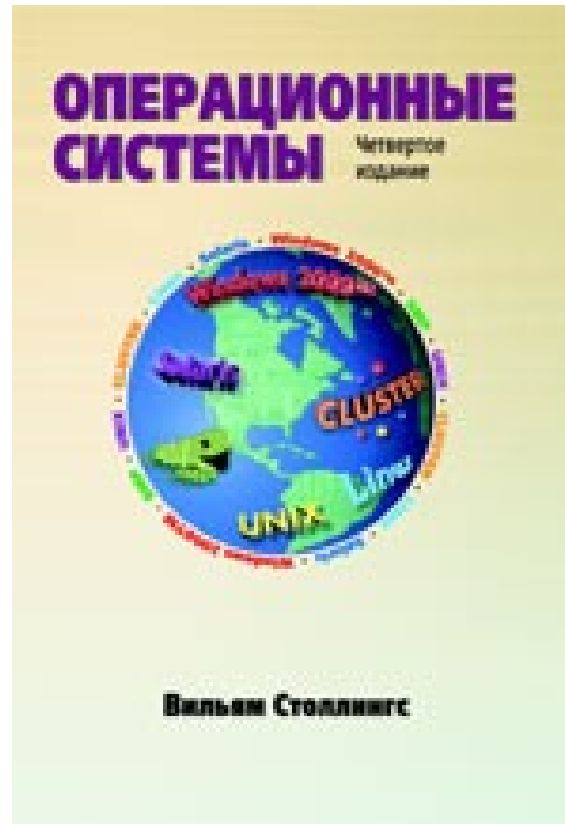
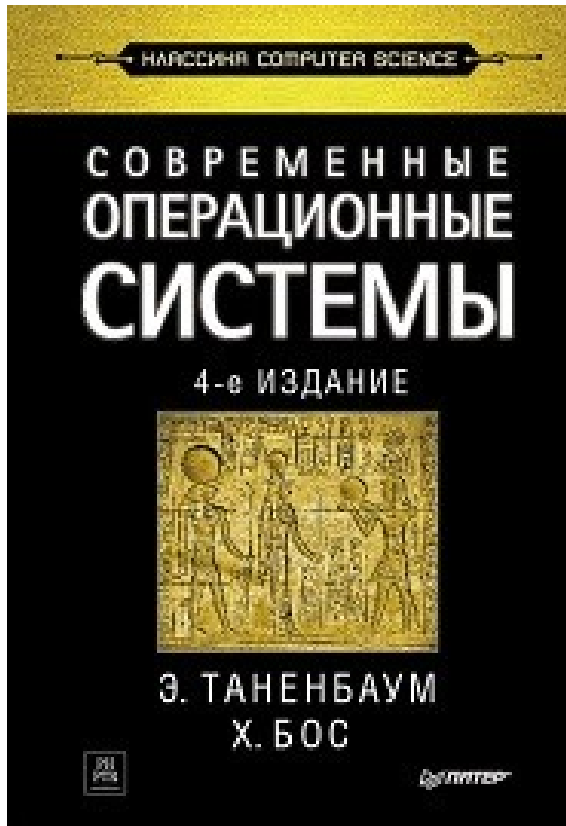


ЦЕЛЬ КУРСА

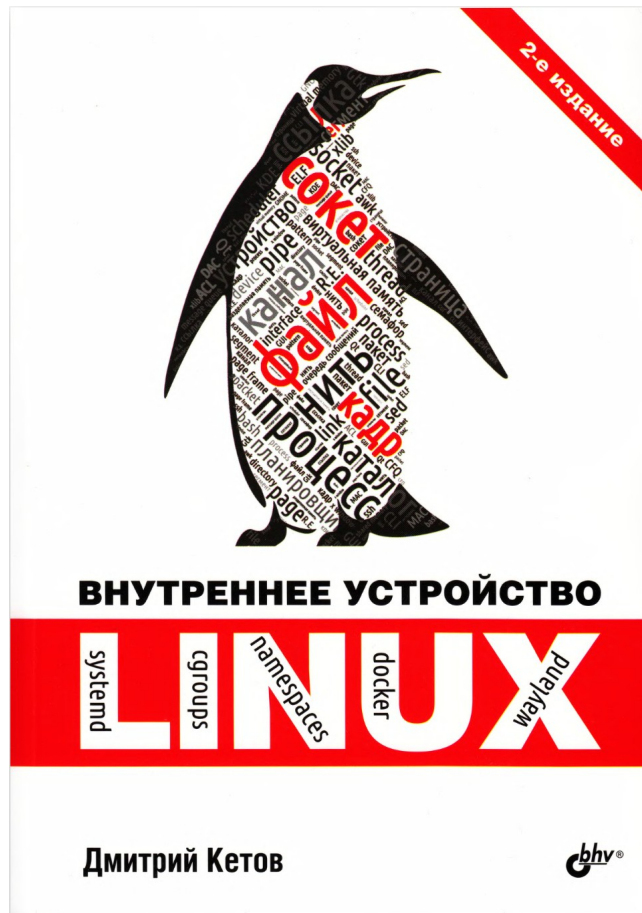
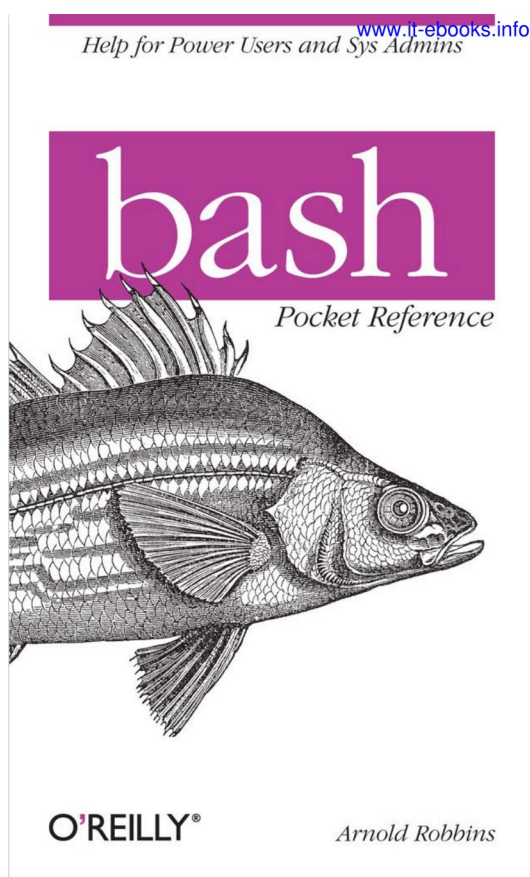
Состоит в овладении:

- 1) Знаниями, относящимися к принципам работы современных операционных систем (на примере ОС Linux [Mint]).
 - 2) Умениями рационально использовать ресурсы операционных систем (на примере ОС Linux) для выполнения задач в профессиональной деятельности.
 - 3) Навыками разработки, установки и администрирования компонентов операционных систем.
- 

ОСНОВНЫЕ КНИГИ



ДОПОЛНИТЕЛЬНЫЕ КНИГИ



ОПРЕДЕЛЕНИЕ

Операционная система (ОС) — это комплекс специального программного обеспечения, который управляет взаимодействием пользователей, прикладных программ с аппаратным обеспечением компьютера, обеспечивая эффективную, стабильную, защищенную работу всех функций, рационально распределяя и изолируя системные ресурсы. Иначе говоря, ОС предоставляет своего рода абстракцию над аппаратным обеспечением для упрощения разработки прикладных программ, осуществляя контроль и планирование использования таких ресурсов как процессорное время, память, устройства ввода-вывода, внешние накопители и т.д.

ЭВОЛЮЦИЯ ОС

1. **Пакетная обработка** (1950-е — 1960-е гг.): Первые ОС были ориентированы на пакетную обработку данных, что позволило пользователям готовить задания на перфокартах и загружать их в систему. ОС управляла очередью задач, минимизируя время простоя оборудования.
 2. **Временное разделение** (1960-е — 1970-е гг.): С появлением концепции времени разделения (timesharing) пользователи могли одновременно работать на одном компьютере. ОС начала поддерживать многозадачность, распределяя время процессора между различными задачами.
-

ЭВОЛЮЦИЯ ОС

3. **Персональные компьютеры** (1980-е гг.): Появление персонального компьютера изменило парадигму ОС. Появились такие ОС, как MS-DOS и Windows, ориентированные на одиночного пользователя и взаимодействие с пользователем через графический интерфейс.

4. **Сетевые и распределенные системы** (1990-е гг.): С развитием компьютерных сетей возникла необходимость в ОС, поддерживающих сетевые функции и распределённые вычисления. ОС, такие как UNIX и Windows NT, начали интегрировать сетевые функции и поддержку многоядерных процессоров.

ЭВОЛЮЦИЯ ОС

5. **Мобильные и встраиваемые системы** (2000-е — наст. время):
Появление смартфонов и планшетов вызвало необходимость создания специализированных ОС, таких как iOS и Android, ориентированных на мобильные устройства и сенсорные интерфейсы.

6. ??? ИИ-платформы или что-то иное

КЛАССИФИКАЦИЯ ОС

По числу пользователей: однопользовательские (например, ранние версии Windows и macOS) и многопользовательские (например, современные версии Linux, Windows Server).

По числу задач: однозадачные (ранние версии MS-DOS) и многозадачные (например, современные операционные системы, такие как Windows 11, Linux, macOS).

По интерфейсу: текстовые (системы с командной строкой, такие как Linux в режиме терминала) и графические (Windows, macOS, Android).

КЛАССИФИКАЦИЯ ОС

По назначению: серверные (Windows Server, Ubuntu Server), настольные (Windows 11, macOS, Ubuntu), мобильные (Android, iOS), встраиваемые (например, FreeRTOS, RTEMS, Android Things).






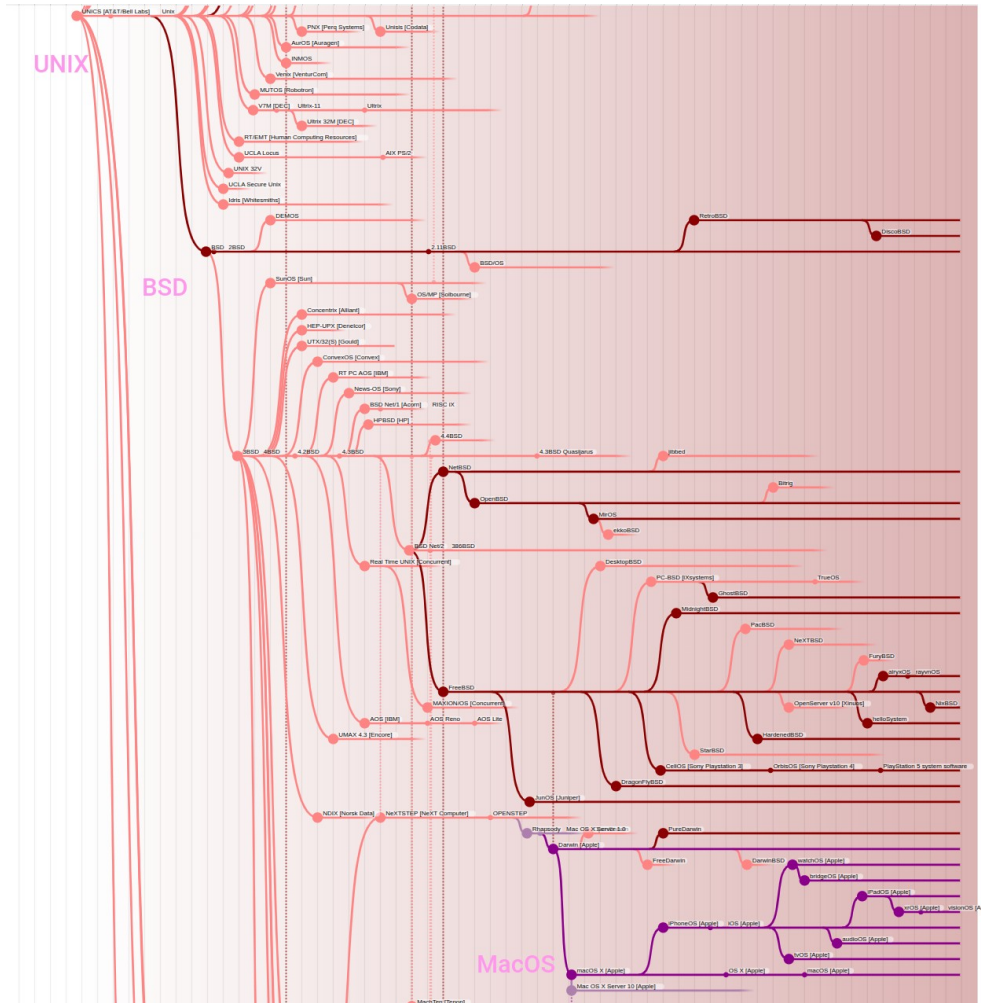
СПЕЦИАЛЬНЫЕ. ОС РЕАЛЬНОГО ВРЕМЕНИ

Жесткие ОС РВ (например, VxWorks, QNX), где недопустимы пропуски временных ограничений, поскольку это может привести к критическим последствиям.

Мягкие ОС РВ (например, Linux с патчем PREEMPT-RT), где временные ограничения важны, но не критичны.

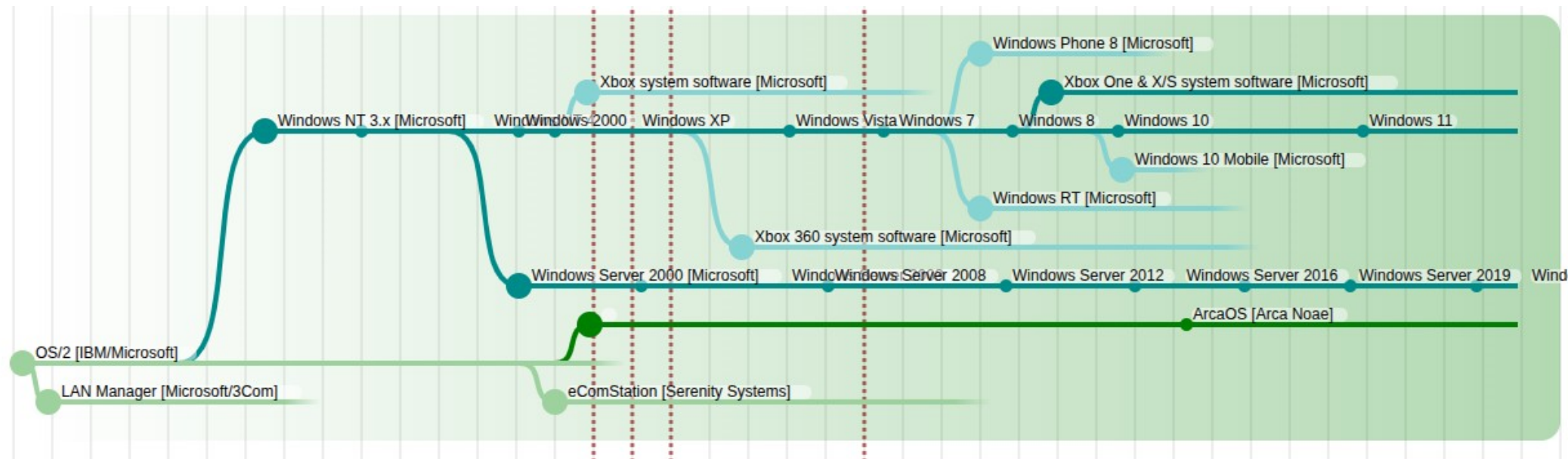


ЭВОЛЮЦИЯ MAC OS

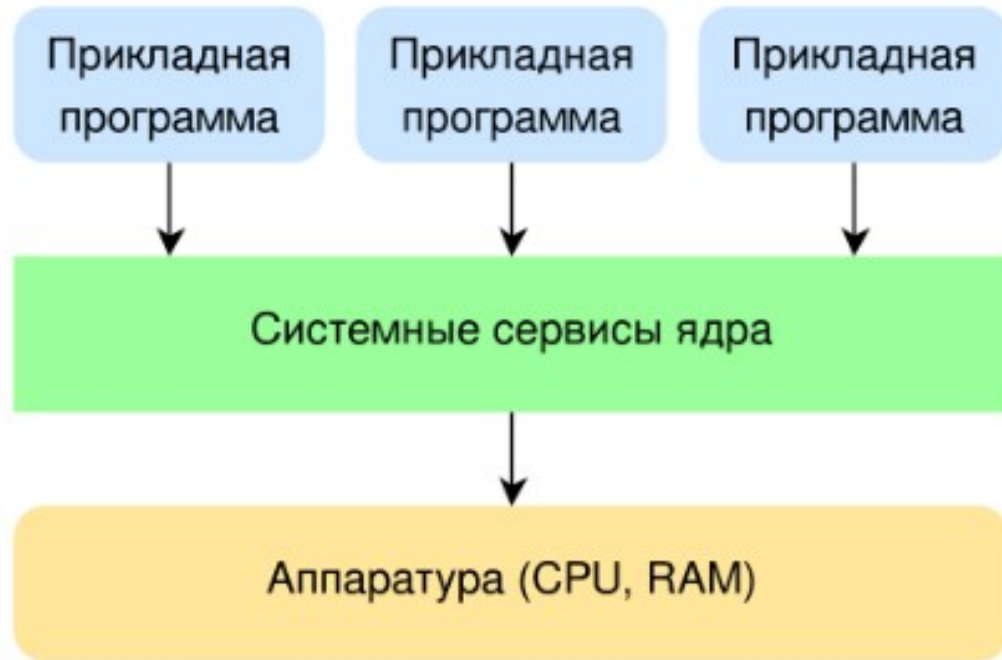




ЭВОЛЮЦИЯ WINDOWS

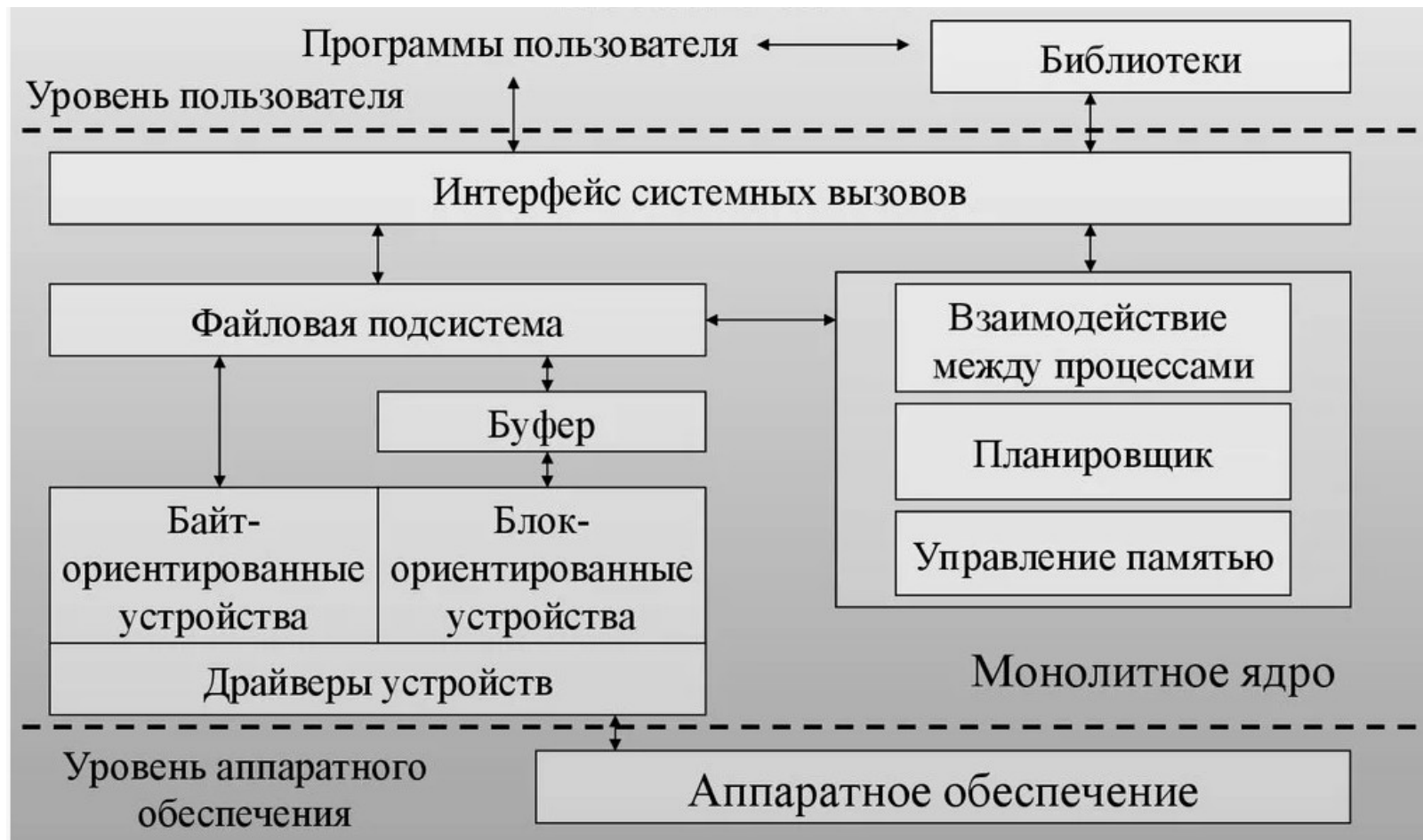


АРХИТЕКТУРА ОС. МОНОЛИТНАЯ

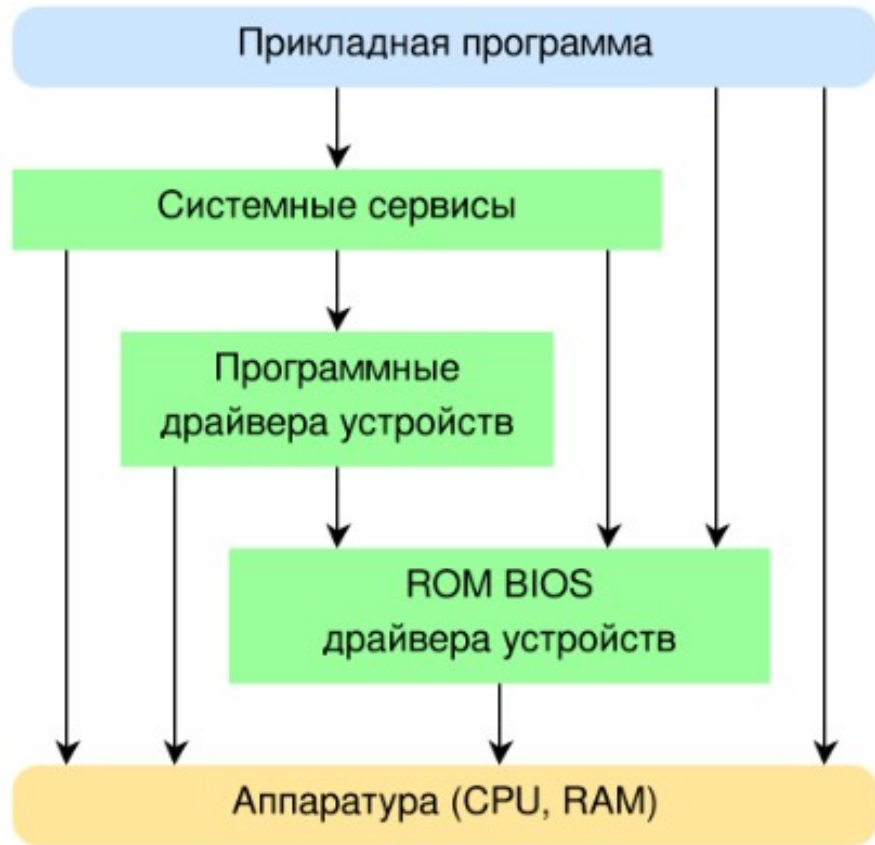


Пример: ОС Линукс с монолитным ядром и возможностью динамической загрузки/выгрузки драйверов и расширений. Здесь всё ядро (драйверы, управление памятью, сетевой стек, планировщик, файловые системы) работает в едином адресном пространстве.

АРХИТЕКТУРА ОС ЛИНУКС

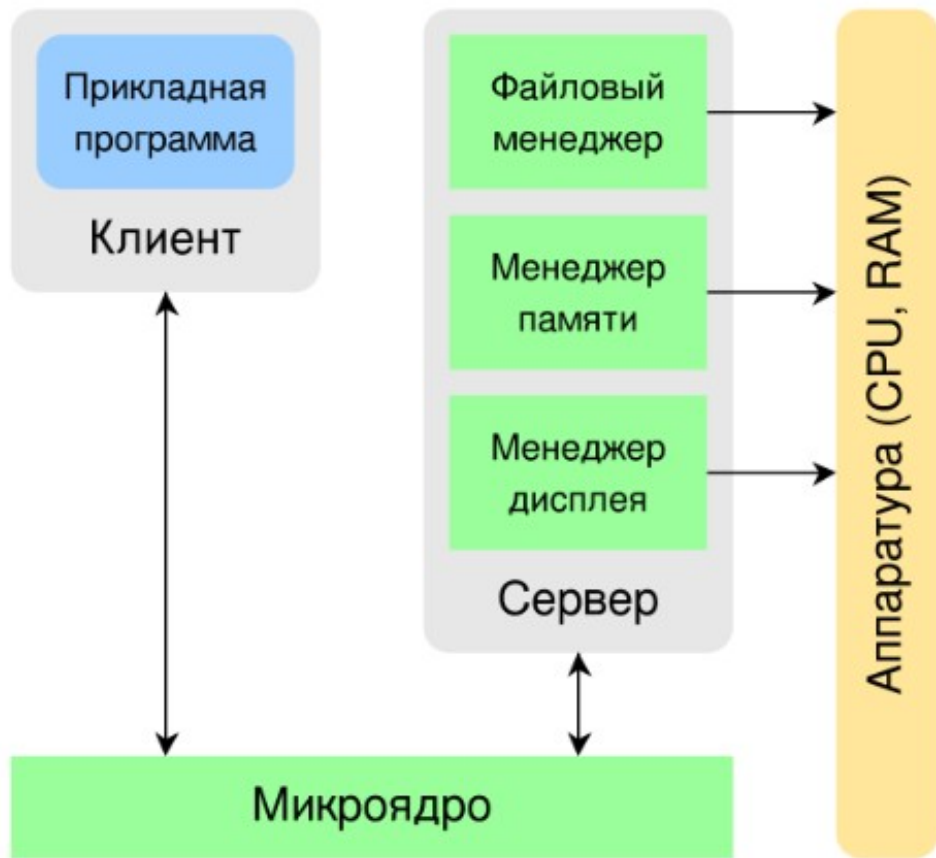


АРХИТЕКТУРА ОС. СЛОЕВАЯ



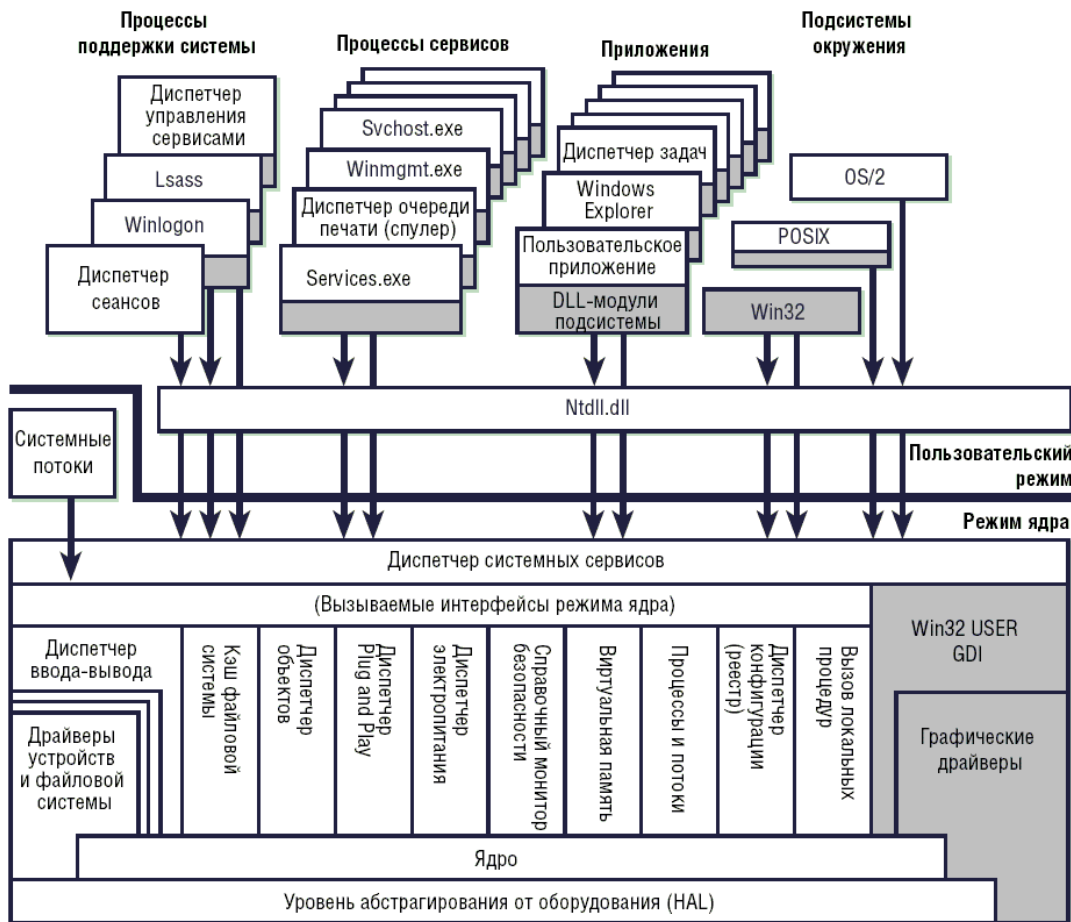
ОС THE system (1968) — это операционная система со слоевой архитектурой. Ядро разделено на несколько уровней (слоёв), каждый из которых решает собственный круг задач и может обращаться только к нижележащему слою. Такая архитектура обеспечивала простоту верификации и модульность, но имела низкую гибкость и производительность.

АРХИТЕКТУРА ОС. КЛИЕНТ-СЕРВЕР



ОС QNX Neutrino — это операционная система с клиент-серверной архитектурой на основе микроядра. Микроядро (обычно ~12-32 КБ) выполняет только минимальные функции (планирование, синхронизация, межпроцессное взаимодействие), а все остальные подсистемы (файловая система, драйверы устройств, менеджер памяти, графическая подсистема) реализованы в виде отдельных серверных процессов в пространстве пользователя. Прикладные программы (клиенты) взаимодействуют с этими серверами через механизм сообщений. Такая архитектура обеспечивает высокую надёжность и отказоустойчивость, так как сбой одного сервиса не приводит к краху всей системы, но при этом вносит накладные расходы на межпроцессное взаимодействие.

АРХИТЕКТУРА WINDOWS 11



ОС Windows 11 — это операционная система с гибридной архитектурой ядра (Windows NT). Ядро сочетает черты монолитного и микроядерного подхода: оно включает в себя основные подсистемы (диспетчер процессов и потоков, диспетчер памяти, подсистемы безопасности, драйверы устройств, файловые системы, сетевой стек), которые выполняются в режиме ядра, но при этом часть сервисов и подсистем (Win32, POSIX, OS/2, службы Windows, графическая подсистема) функционирует в режиме пользователя как отдельные процессы и окружения. Прикладные программы работают поверх пользовательских подсистем (например, Win32 API), которые взаимодействуют с ядром через системные вызовы и динамические библиотеки (ntdll.dll).



ФУНКЦИИ ОС

Управление процессами: создание, планирование и завершение процессов, управление состояниями процессов и их приоритетами.

Управление памятью: распределение и освобождение оперативной памяти, виртуальная память, управление страницами и сегментами.

Управление устройствами: взаимодействие с внешними устройствами, драйверами устройств и их программирование.





ФУНКЦИИ ОС

Управление файлами: создание, удаление, копирование, перемещение и защита файлов, управление файловой системой.

Обеспечение безопасности и защиты: управление правами доступа, аутентификация пользователей, защита данных и системных ресурсов.

Обработка ошибок: обнаружение, регистрация и восстановление после ошибок.



ОС КАК ВИРТУАЛЬНАЯ МАШИНА

Операционная система создаёт абстракцию аппаратных ресурсов, предоставляя пользователям и программам виртуальную машину, которая упрощает взаимодействие с компьютером. Эта виртуальная машина скрывает сложности реального аппаратного обеспечения, предоставляя удобный интерфейс для выполнения программ и доступа к ресурсам.



ОС КАК СИСТЕМА УПРАВЛЕНИЯ РЕСУРСАМИ

Операционная система также рассматривается как система управления ресурсами, распределяющая процессорное время, память, устройства ввода-вывода и другие ресурсы между различными программами и пользователями. Она решает, какие программы должны получить доступ к ресурсам, в каком порядке и на каких условиях. Это позволяет избегать конфликтов и перегрузок, обеспечивая эффективное и справедливое использование ресурсов. ОС использует различные алгоритмы планирования для оптимального распределения процессорного времени, управления памятью и другими ресурсами.



ИНТЕРФЕЙС ОС ДЛЯ ПРИКЛАДНОГО ПРОГРАММИРОВАНИЯ

Операционные системы предоставляют интерфейсы прикладного программирования (**API**), которые позволяют разработчикам программировать приложения, взаимодействующие с ОС. API обеспечивают стандартный способ доступа к функциям ОС, таким как работа с файлами, управление процессами, работа с сетью и другие.

Примеры популярных API:

- **WinAPI** для Windows
- **POSIX** для UNIX и Linux систем





UNIX

UNIX был разработан в 1969 году Кеном Томпсоном, Деннисом Ритчи и другими сотрудниками Bell Labs как операционная система для многозадачных компьютеров. Изначально UNIX создавался для упрощения работы с компьютером и был написан на ассемблере, но позже переписан на языке C, что сделало его портируемым на различные платформы. Благодаря своей модульности, гибкости и простоте архитектуры, UNIX стал основой для многих современных операционных систем, таких как Linux и macOS. Он также повлиял на развитие стандартов POSIX и других ключевых технологий в мире операционных систем.



ЛИНУКС. УБУНТУ

Ubuntu — это популярный дистрибутив Linux, основанный на Debian и разработанный компанией Canonical. Первый релиз состоялся 20 октября 2004 года, и с тех пор дистрибутив обновляется каждые шесть месяцев. Основная цель Ubuntu — предложить простой и удобный в использовании операционный продукт, который подходит как для серверов, так и для настольных компьютеров. Ubuntu поддерживается международным сообществом разработчиков и распространяется с акцентом на свободное программное обеспечение.

ЛИНУКС. Минт

Linux Mint — это популярный дистрибутив Linux, основанный на Ubuntu (а ранее и на Debian), ориентированный на удобство и простоту для конечного пользователя. Первый релиз состоялся в 2006 году, и с тех пор дистрибутив развивается с акцентом на стабильность, интуитивно понятный интерфейс и готовность к работе «из коробки». Основная цель Linux Mint — предоставить пользователю операционную систему с привычным рабочим окружением, где всё уже настроено и доступно без дополнительных усилий. Mint поддерживается сообществом разработчиков и активно используется как на домашних компьютерах, так и в образовательной и рабочей среде.

ПОПУЛЯРНЫЕ ДИСТРИБУТИВЫ

Page Hit Ranking		
Data span:		
Last 6 months		
Go		
Rank	Distribution	HPD*
1	CachyOS	3646▲
2	Mint	2623▼
3	MX Linux	1742▲
4	Debian	1638▲
5	EndeavourOS	1519▼
6	Pop!_OS	1259▼
7	Manjaro	1118▲
8	Ubuntu	1071▼
9	Fedora	1028▬
10	Zorin	831▲
11	openSUSE	795▬
12	Nobara	711▼
13	AnduinOS	696▲
14	elementary	586▬
15	NixOS	563▬
16	Bluestar	522▲
17	KDE neon	498▼
18	Garuda	486▲
19	TUXEDO	478▼
20	antiX	469▬
21	AlmaLinux	427▬
22	BigLinux	415▲
23	Kali	411▬

POSIX vs WinAPI

POSIX — это стандарт для обеспечения совместимости между разными UNIX-подобными ОС (Linux, macOS, BSD). Это делает разработку кроссплатформенных приложений проще. Контроль над системой, благодаря открытой природе. Исходный код ядра и системных компонентов.

WinAPI — это проприетарный API, предназначенный только для Windows. Пользователи и разработчики не имеют полного доступа к исходному коду.



POSIX vs WinAPI

POSIX — ориентирован на серверные, встраиваемые, высоконагруженные системы. Производителен при работе с процессами, сетевыми операциями и файловыми системами.

WinAPI — больше ориентирован на настольные и коммерческие приложения.



POSIX vs WinAPI

POSIX предоставляет гибкие и мощные инструменты для работы с процессами и потоками, такие как `fork()`, `exec()`, `pthread` и другие. Создание новых процессов и управление ими не ресурсоемкое.

В **WinAPI** механизмы работы с процессами и потоками немного сложнее и не столь гибки. Например, создание процесса в Windows требует больше ресурсов, чем аналогичный вызов в Linux.

POSIX vs WinAPI

Linux (**POSIX**) имеет мощные инструменты для автоматизации через командную строку и скрипты (bash и др.). Возможность писать скрипты для выполнения системных задач делает Linux гибкой и удобной системой для администраторов и разработчиков.

В Windows (**WinAPI**) командная строка (CMD, PowerShell) менее интегрирована с системой, и для сложных задач может потребоваться использование сторонних инструментов.

POSIX vs WinAPI

В **POSIX** системах, таких как Linux, модель управления правами пользователей и процессов более гибкая и прозрачная. Файловые права доступа и разделение привилегий на уровне ядра и пользователей четко определены.

В **Windows** сложная система списков ACL затрудняет понимание того, почему определённый пользователь имеет или не имеет доступ к ресурсу. В POSIX таких сложностей меньше, поскольку права доступа проще и легко проверяются. В отличие от POSIX, где есть всего три основных категории прав (чтение, запись, выполнение), Windows позволяет задавать десятки различных видов разрешений (например, удаление, изменение атрибутов файла, создание новых объектов и т.д.). Это даёт большую гранулярность, но может запутать пользователей и администраторов.

POSIX

POSIX (Portable Operating System Interface) — это набор стандартов, разработанных IEEE (Institute of Electrical and Electronics Engineers) для обеспечения совместимости между операционными системами. POSIX определяет интерфейсы и функции, которые операционные системы должны предоставлять для обеспечения переносимости приложений. Основной целью POSIX является стандартизация интерфейсов между программным обеспечением и операционной системой, чтобы программы, написанные для одной POSIX-совместимой ОС, могли быть легко перенесены на другую.



POSIX

Включает в себя такие аспекты, как:

Системные вызовы (например, для управления процессами, работы с файлами и сети).

Библиотечные функции (например, для работы со строками, математических операций и управления памятью).

Утилиты командной строки (например, grep, awk, sed).

Скриптовый язык командного интерпретатора (например, sh).

