Лекция 2.
Компоненты ОС.
Ядро. Виды ядра

Архитектура ОС

Какой-либо единой архитектуры ОС не существует, но существуют универсальные подходы к структурированию.

Общий подход к структурированию ОС – разделение её компонентов на 3 группы:

- ✔ **Ядро**
- ✔ **Драйверы устройств**
- ✔ **Оболочка (интерфейс)**

Архитектура ОС

переводит команды с языка программ на язык машинных кодов

оболочка, с помощью которой пользователь «общается» с компьютером.

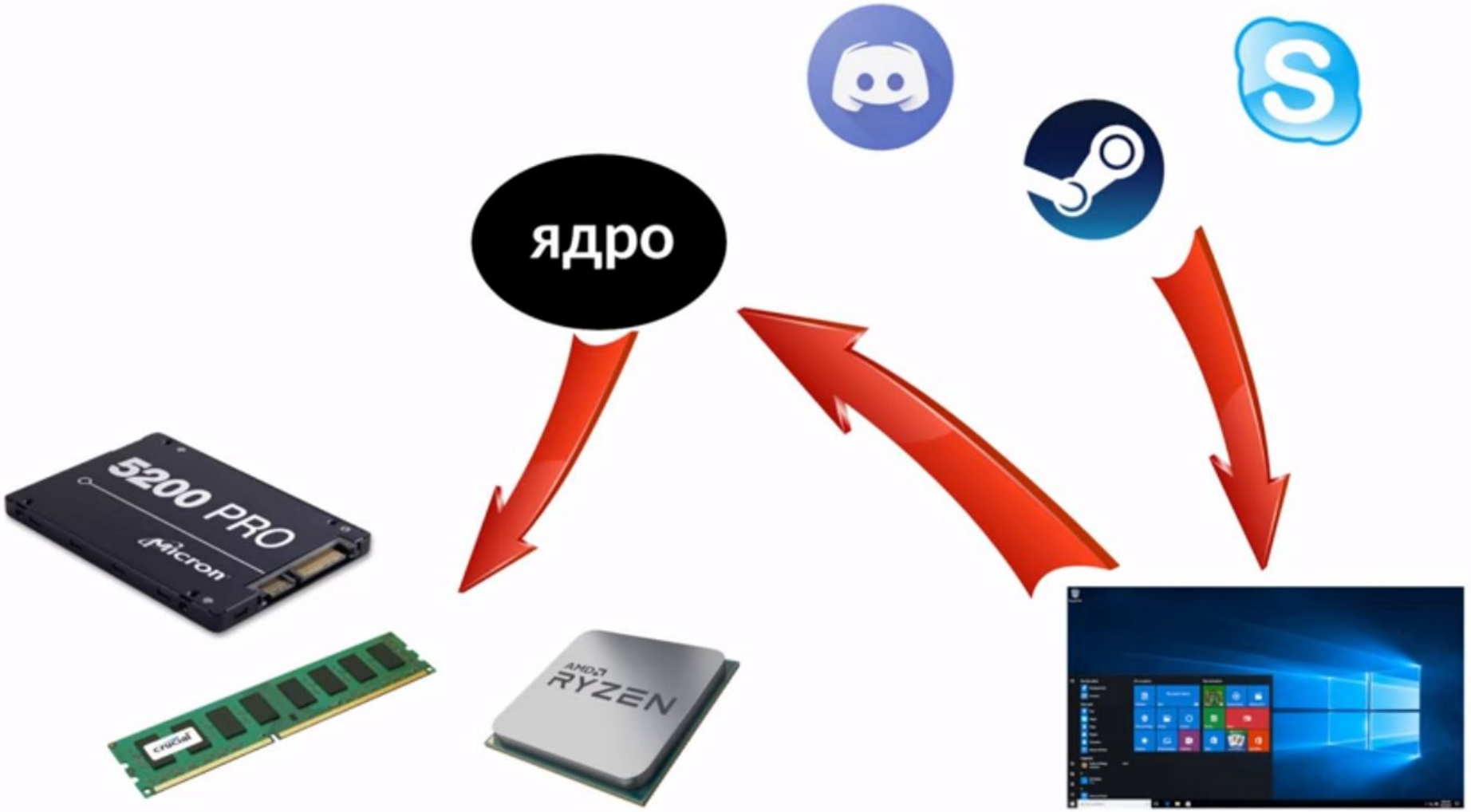
программы, управляющие устройствами

Ядро ОС

Ядро — **центральная часть операционной системы (ОС),** обеспечивающая приложениям координированный доступ к ресурсам компьютера, таким как процессорное время, память и внешнее аппаратное обеспечение. Также обычно **ядро предоставляет сервисы файловой системы и сетевых протоколов**.

Как правило, ядро предоставляет такой **доступ исполняемым процессам** соответствующих приложений **за счёт использования механизмов межпроцессного взаимодействия и обращения приложений к системным вызовам** ОС.

Ядро ОС



Драйверы

- **Драйвер (англ. driver)** – компьютерное программное обеспечение, с помощью которого другое программное обеспечение (операционная система) получает доступ к аппаратному обеспечению некоторого устройства. Обычно с операционными системами поставляются драйверы для ключевых компонентов аппаратного обеспечения, без которых система не сможет работать.

Оболочка ОС

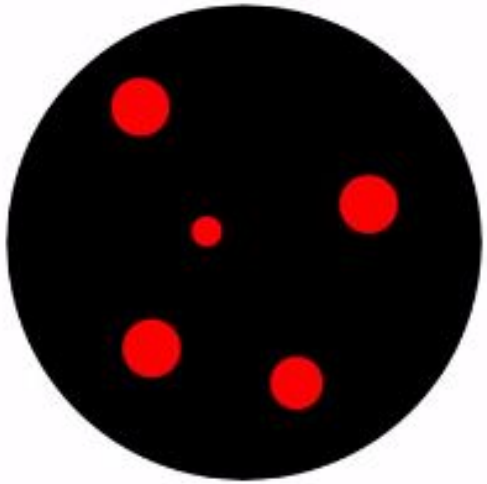
- **Оболочка операционной системы (от англ. shell «оболочка»)** – интерпретатор команд операционной системы, обеспечивающий интерфейс для взаимодействия пользователя с функциями системы. Оболочка – это самый внешний уровень операционной системы.

Типы архитектур ядер ОС

1. Монолитное ядро
2. Модульное ядро
3. Микроядро
4. Экзоядро
5. Наноядро
6. Гибридное ядро

Монолитное ядро

Монолитное ядро - ядро, все части которого работают в одном адресном пространстве.



- пространство ядра
- драйвера

Монолитное ядро

- компоненты ОС являются **составными частями** одной большой программы, а не самостоятельными модулями
- используют **общие структуры данных**
- **набор процедур**, каждая из которых может вызвать каждую

Монолитное ядро

Достоинства:

- скорость работы
- упрощённая разработка модулей
- богатство предоставляемых возможностей и функций
- поддержка большого количества разнообразного оборудования

Монолитное ядро

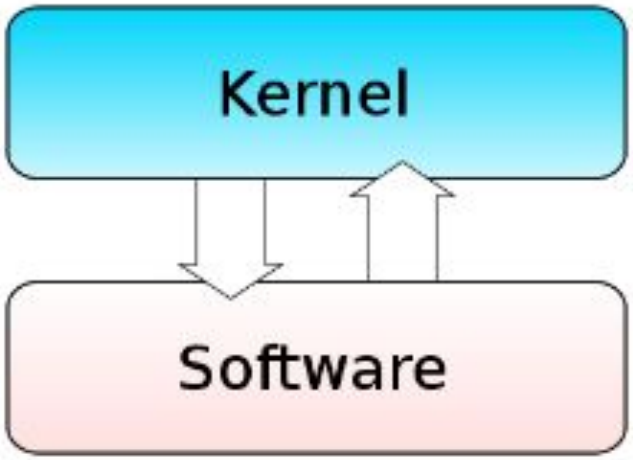
Недостатки:

- поскольку всё ядро работает в одном адресном пространстве, сбой в одном из компонентов может нарушить работоспособность всей системы
- присутствие в ядре лишних компонентов крайне нежелательно, так как ядро всегда полностью располагается в оперативной памяти

Монолитное ядро

Примеры:

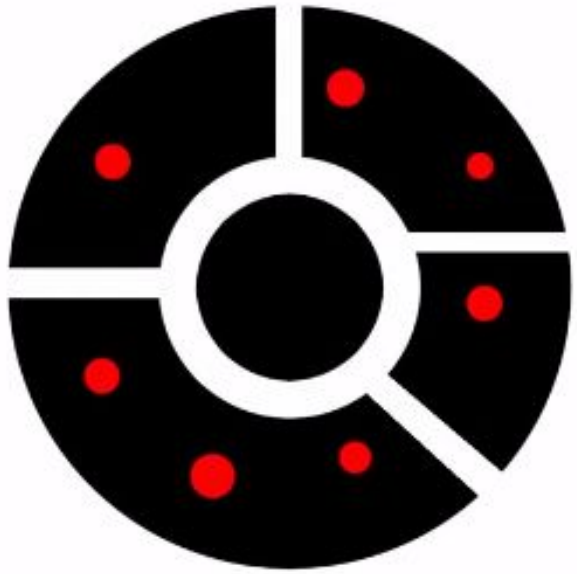
- Традиционные ядра UNIX(такие как BSD),
- ядра Linux
- ядро MS-DOS.



UNIX®

Модульное ядро

Модульное ядро — усовершенствованная модификация архитектуры монолитных ядер



- пространство ядра
- драйвера

Модульное ядро

- Модульность ядра осуществляется на уровне бинарного образа, а не на архитектурном уровне ядра

- Все модули ядра работают в адресном пространстве ядра и могут пользоваться всеми функциями, предоставляемыми ядром.

Модульное ядро

Достоинства:

- модульные ядра предоставляют тот или иной **механизм подгрузки модулей ядра**, поддерживающих то или иное аппаратное обеспечение (например, драйверов)
- **не требуется** многократная полная **перекомпиляция** ядра при работе над какой-либо его подсистемой или драйвером
- модули позволяют легко **расширить возможности** ядра по мере необходимости
- модульные ядра предоставляют особый **программный интерфейс** (API) для связывания модулей с ядром

Модульное ядро

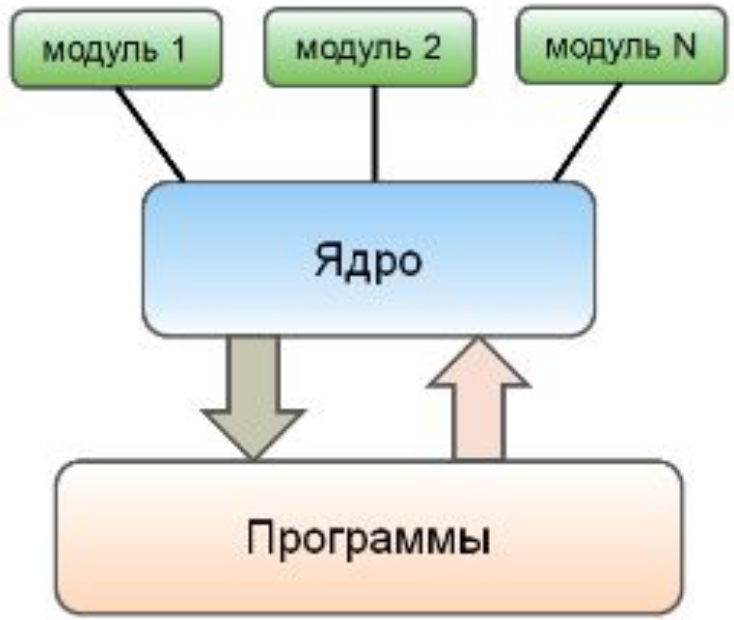
Недостатки:

- не все части ядра могут быть сделаны модулями. **Некоторые части ядра** всегда обязаны присутствовать в оперативной памяти и **должны быть жёстко «вшиты»** в ядро
- не все модули допускают динамическую подгрузку (без перезагрузки ОС)
- на модули ядра накладываются определённые **ограничения в части используемых функций** (например, они не могут пользоваться функциями стандартной библиотеки C/C++ и должны использовать специальные аналоги, являющиеся функциями API ядра)

Модульное ядро

Примеры

- UNIX
- Linux



Микроядро

Микроядро — ядро операционной системы, реализующее минимальный набор функций.

Микроядро

- предоставляет только элементарные функции управления процессами и минимальный набор абстракций для работы с оборудованием
- бóльшая часть работы осуществляется с помощью специальных пользовательских процессов, называемых сервисами
- перенесение значительной части системного кода на уровень пользователя и одновременная минимизация ядра

Микроядро

- микроядро работает в привилегированном режиме и обеспечивает:

✔ взаимодействие между программами,

✔ планирование использования процессора,

✔ первичную обработку прерываний,

✔ операции ввода-вывода

✔ базовое управление памятью

- остальные компоненты ОС взаимодействуют друг с другом путем передачи сообщений через микроядро

Микроядро

Сервисные процессы (в UNIX - "демоны") используются в различных ОС для решения задач:

- запуска программ по расписанию (UNIX и Windows NT),
- ведения журналов событий (UNIX и Windows NT),
- централизованной проверки паролей и хранения пароля текущего интерактивного пользователя в специально ограниченной области памяти (Windows NT).

Тем не менее не следует считать ОС микроядерными только из-за использований такой архитектуры.

Решающим критерием "микроядерности" является **размещение всех или почти всех драйверов и модулей в сервисных процессах**, иногда с явной невозможностью загрузки любых модулей расширения в собственно микроядро, а также разработки таких расширений

Микроядро

Достоинства:

- ✔Устойчивость к сбоям оборудования, ошибкам в компонентах системы
- ✔существенно упрощает добавление в ядро новых компонентов; можно, не прерывая работы ОС, загружать и выгружать новые драйверы, файловые системы и т. д.
- ✔упрощается процесс отладки компонентов ядра, так как новая версия драйвера может загружаться без перезапуска всей операционной системы
- ✔Компоненты ядра операционной системы ничем принципиально не отличаются от пользовательских программ, поэтому для их отладки можно применять обычные средства

Микроядро

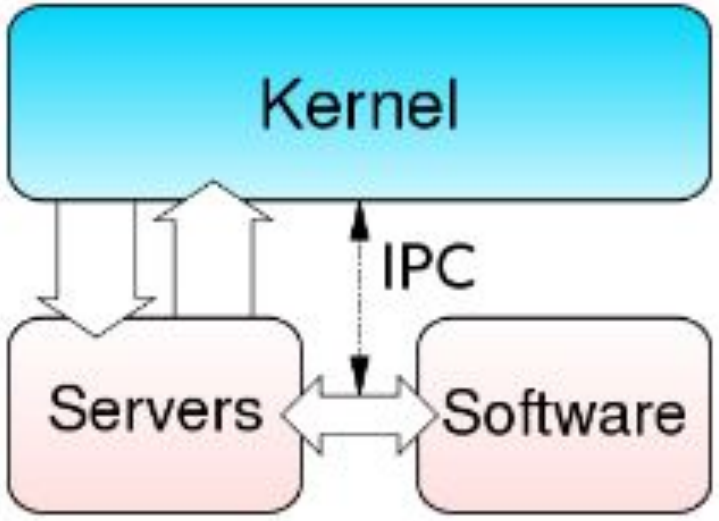
Недостатки:

- Передача данных между процессами требует накладных расходов.
- необходимость очень аккуратного проектирования с целью минимизации взаимодействия между компонентами ОС

Микроядро

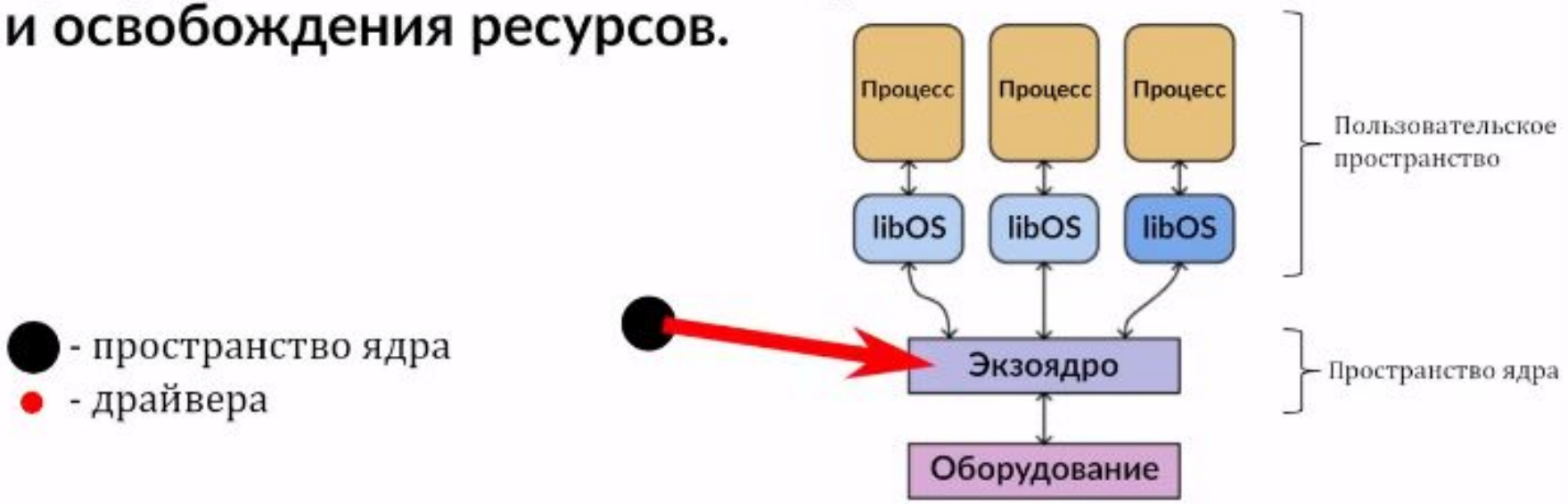
Примеры:

- Symbian OS
- AmigaOS
- MorphOS



Экзоядро

Экзоядро — ядро предоставляющее функции для взаимодействия между процессами и безопасного выделения и освобождения ресурсов.



Экзоядро

- предполагается, что API для прикладных программ будут предоставляться внешними по отношению к ядру библиотеками

- возможность доступа к устройствам на уровне контроллеров, что позволит, например, иметь доступ к диску на уровне секторов диска, что положительно скажется на быстродействии

Экзоядро (принципы)

1. **Экзоядро не абстрагирует ресурсы**. Это делают непривилегированные прикладные библиотеки - "библиотечные операционные системы" (libOS, library operating system). еализована своя libOS - сугубо в необходимых масштабах.
2. **Ресурсами должны управлять сами приложения**. Обязанности экзоядра сведены к выполнению минимума функций, связанных:

- с защитой именования, выделения, высвобождения и разделения ресурсов,
- с контролем прав доступа.

Экзоядро (принципы)

3. **Интерфейсы ресурсов должны быть как можно ближе к "железу"**. Чем ниже уровень интерфейса, тем меньше требуется вмешательства со стороны экзоядра и тем больший контроль над ресурсом получают приложения.

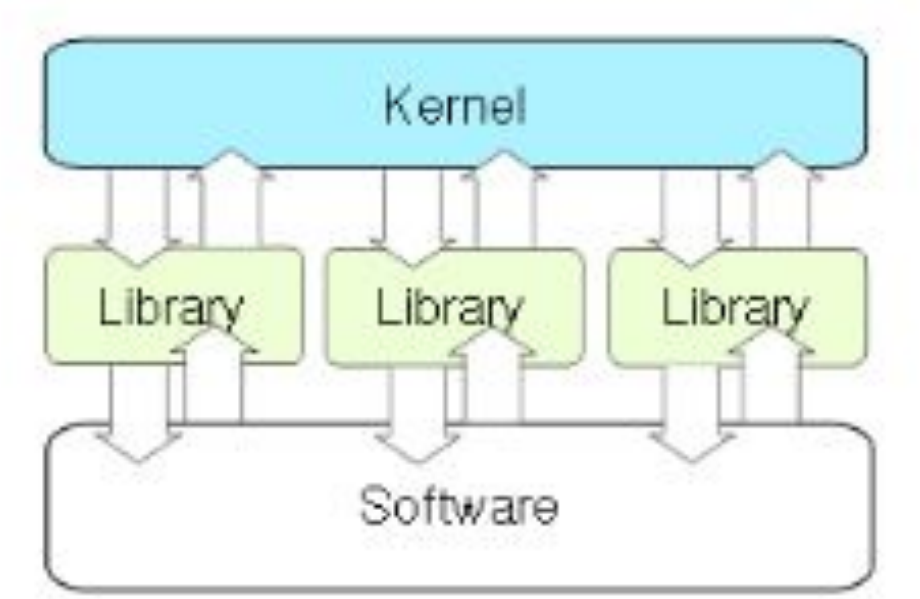
В идеальной ситуации интерфейсы и есть "железо".

4. **Прикладное управление ресурсами реализуется аппаратной частью.** Экзоядро стремится безопасно экспортировать все аппаратные операции.

Экзоядро

Пример:

XOK/ExOS
XOK
ExOS



Экзоядро

Достоинства:

Уменьшает абстрагируемость ресурсов в результате чего повышается надежность, приспособляемость, производительность, гибкость ОС

Недостатки:

Экспериментальная ОС

Наноядро

Наноядро — архитектура ядра, в рамках которой упрощённое и минималистичное ядро выполняет одну задачу — обработку аппаратных прерываний

- пространство ядра
- драйвера

Наноядро

После обработки прерываний от аппаратуры наноядро, в свою очередь, посылает информацию о результатах обработки (например, полученные с клавиатуры символы) вышележащему ПО.

Примеры:

KeyKOS — самая первая ОС на наноядре (1983г).
Symbian OS v8.1- S60 2rd Edition (Nokia N70, Nokia N90)

Гибридное ядро

Гибридное ядро — модифицированное микроядро, позволяющие для ускорения работы запускать модули ОС в пространстве ядра.

Гибридное ядро

Гибридное ядро, по сути представляет собой модификацию микроядра, которая позволяет ускорить работу системы за счёт запуска второстепенных программ в программной области ядра.

Имеют «гибридные» достоинства и недостатки.

Пример: Windows 10