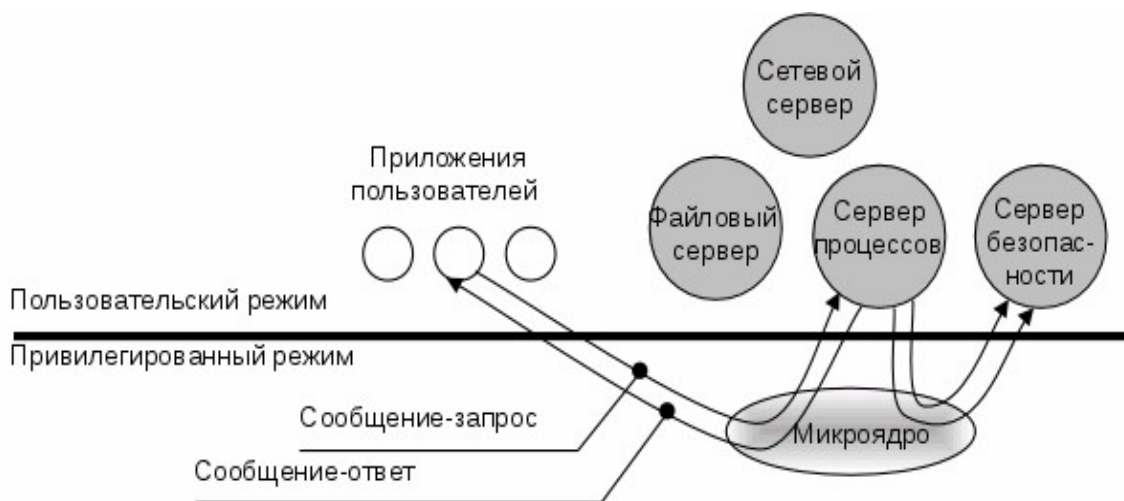


Реализация системного вызова в ОС с микроядерной архитектурой

Однозначного решения о переносе в пользовательский режим тех или иных системных функций не существует. В общем случае как пользовательские приложения оформляются многие менеджеры ресурсов.

По определению, основным назначением такого приложения является обслуживание запросов других приложений (создание процесса, выделение памяти, проверка прав доступа и т.д.). Поэтому менеджеры ресурсов, вынесенные в пользовательский режим, называются *серверами* ОС. Одной из главных задач микроядра является поддержка взаимодействия серверов.



Клиент (прикладная программа либо другой компонент ОС) посылает соответствующему серверу сообщение-запрос на выполнение некоторой функции.

Непосредственная передача этого сообщения серверу невозможна, так как каждое приложение работает в своем адресном пространстве. В качестве посредника выступает микроядро, выполняющееся в привилегированном режиме и имеющее доступ к адресным пространствам всех приложений. Микроядро передает сообщение нужному серверу, сервер выполняет запрошенную операцию и результат, снова через посредство микроядра, возвращается клиенту с помощью другого сообщения.

Такая схема обработки запроса соответствует модели клиент-сервер, где микроядро выполняет роль транспортных средств.

Схематично механизм обращений к функциям ОС, оформленным в виде серверов:



Преимущества и недостатки микроядерной архитектуры

ОС, основанные на концепции микроядра, в высокой степени удовлетворяют большинству требований, предъявляемых к современным ОС:

- единообразные интерфейсы;
- простота расширяемости;
- высокая гибкость;
- возможность переносимости;
- высокая надежность;
- поддержка распределенных систем;
- поддержка объектно-ориентированных ОС.

Основным недостатком микроядерной архитектуры является снижение производительности по сравнению с классическим вариантом. Так, при классической организации выполнение системного вызова требует двух переключений режимов «привилегированный – пользовательский», а при микроядерной – четырех. При обращении к часто используемым функциям

работа приложений существенно замедляется. По этой причине микроядерный подход не получил широкого распространения.

Обработка системного вызова в микроядерной архитектуре

Схема смены режимов при выполнении системного вызова в ОС с микроядерной архитектурой выглядит, как показано на рисунке. Из рисунка ясно, что выполнение системного вызова сопровождается четырьмя переключениями режимов ($4t$), в то время как в классической архитектуре – двумя.

Следовательно, производительность ОС с микроядерной архитектурой при прочих равных условиях будет ниже, чем у ОС с классическим ядром.



Обработка системного вызова в классической архитектуре

Повышение устойчивости ОС обеспечивается переходом ядра в привилегированный режим. При этом происходит некоторое замедление выполнения системных вызовов. *Системный вызов* привилегированного ядра инициирует переключение процессора из пользовательского режима в привилегированный, а при возврате к приложению – обратное переключение.



Соотношение классической и микроядерной архитектуры



СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Астапчук, В. А. Корпоративные информационные системы: требования при проектировании: учебное пособие для вузов / В. А. Астапчук, П. В. Терещенко. – 2-е изд., испр. и доп. – Москва: Издательство Юрайт, 2021. – 113 с. – (Высшее образование). – ISBN 978-5-534-08546-4. – URL: <https://urait.ru/bcode/472111>
2. Гостев, И. М. Операционные системы: учебник и практикум для среднего профессионального образования / И. М. Гостев. – 2-е изд., испр. и доп. – Москва : Издательство Юрайт, 2021. – 164 с. – (Профессиональное образование). – ISBN 978-5-534-04951-0. – URL: <https://urait.ru/bcode/472333>
3. Гостев, И. М. Операционные системы: учебник и практикум для вузов / И. М. Гостев. – 2-е изд., испр. и доп. – Москва: Издательство Юрайт, 2021. – 164 с. – (Высшее образование). – ISBN 978-5-534-04520-8. – URL: <https://urait.ru/bcode/470010>
4. Нестеров, С. А. Базы данных: учебник и практикум для вузов / С. А. Нестеров. – Москва : Издательство Юрайт, 2021. – 230 с. – (Высшее образование). – ISBN 978-5-534-00874-6. – URL: <https://urait.ru/bcode/469516>
5. Попов И. И. Операционные системы, среды и оболочки : учебное пособие / Т.Л. Партыка, И.И. Попов. — 5-е изд., перераб. и доп. — М. : ФОРУМ : ИНФРА-М, 2022. — 560 с.
6. Рудаков, А. В. Операционные системы и среды : учебник / Рудаков А.В. — Москва : КУРС: ИНФРА-М, 2021. — 304 с. — (Среднее профессиональное образование). - ISBN 978-5-16-106301-9
7. Рудаков А. В. Операционные системы и среды : учебник / Рудаков А.В. — М.: КУРС: ИНФРА-М, 2022. — 304 с.



ТЕСТ №1 по ОП.01. Операционные системы и среды

Студента группы _____

ФИО _____

Дата _____

ТЕСТ

1. Дайте определение Архитектуры операционной системы

2. Выберите один правильный ответ:

- Модули ядра являются:
 - А. Резидентными
 - В. Временными
 - С. Транзитными
- Определение Транзитные означает:
 - А. загружаются в оперативную память только на время выполнения.
 - В. постоянно находятся в оперативной памяти
 - С. архивируются после завершения использования
- Привилегированный режим:
 - А. для работы ОС или ее частей (процессор может выполнять все возможные команды).
 - В. для работы приложений
 - С. для программ, решающие отдельные задачи управления и сопровождения компьютерной системы

3. Дополните предложение:

Наиболее общим подходом к структуризации операционной системы является разделение всех ее модулей на две группы:

4. Дайте определение: Ядро – это

5. Дополните фразу:

Для работы приложений (недоступны команды процессора, связанные с управлением аппаратным обеспечением, защитой оперативной памяти, переключением режимов работы процессора) используется _____ режим.

6. Основной недостаток монолитной архитектуры:

7. Соотнесите понятия:

I. Утилиты	А. загрузчики, отладчики, текстовые или графические редакторы
II. Библиотеки процедур различного назначения для разработки приложений	Б. калькулятор, некоторые игры
III. Программы, предоставляющие дополнительные услуги	В. математические функции, функции ввода-вывода
IV. Системные обрабатывающие программы	Г. программы, решающие отдельные задачи управления и сопровождения компьютерной системы (сжатие дисков, их проверка, дефрагментация; архивирование, сбор статистики и т.д.);

8. Какие виды архитектур существуют?

- 1.
- 2.
- 3.

9. Дополните фразу:

В монолитном ядре реализуются все основные функции операционной системы, и оно является, по сути, _____ программой, представляющей собой совокупность процедур - большой набор сервисных функций.

10. Архитектура ОС, основанная на *привилегированном ядре* и *приложениях пользовательского режима* является:

- A. Классической
- B. Монолитной
- C. Полилитной
- D. Стандартной

11. Нарисуйте многослойную архитектуру и подпишите ее части (в общем виде):

12. Главная идея микроядра:

- A. Вспомогательные модули являются транзитными (загружаются в оперативную память только на время выполнения)
- A. Группировка модулей в менеджеры обычно осуществляется по функциям основных подсистем ОС
- B. Взаимодействует непосредственно с приложениями и системными утилитами, образуя прикладной программный интерфейс ОС
- C. минимизировать само ядро, вынести как можно функциональности в режим пользователя (т.е. исполнять эту функциональность в виде обычных процессов)



Тема 2. Архитектура операционной системы

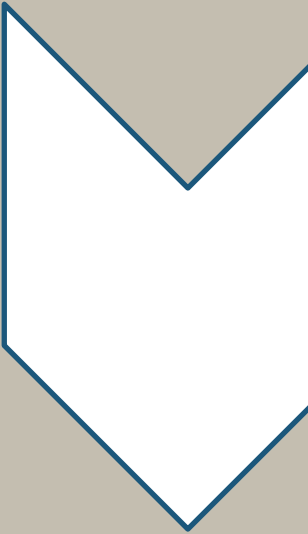
Определение архитектуры ОС

Архитектура операционной системы

- структурная и функциональная организация ОС на основе некоторой совокупности программных модулей

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

Структура операционной системы



Наиболее общим подходом к структуризации операционной системы является разделение всех ее модулей на две группы:

ядро

вспомогательные модули.

ЯДРО

Ядро

- ключевой, основной компонент операционной системы, именно в нем реализуется большая часть функциональности ОС

Основные функции:

- управление процессами;
- управление памятью;
- управление вводом-выводом и файловая система;
- интерфейс прикладного программирования для поддержки обращений к ядру из приложений.

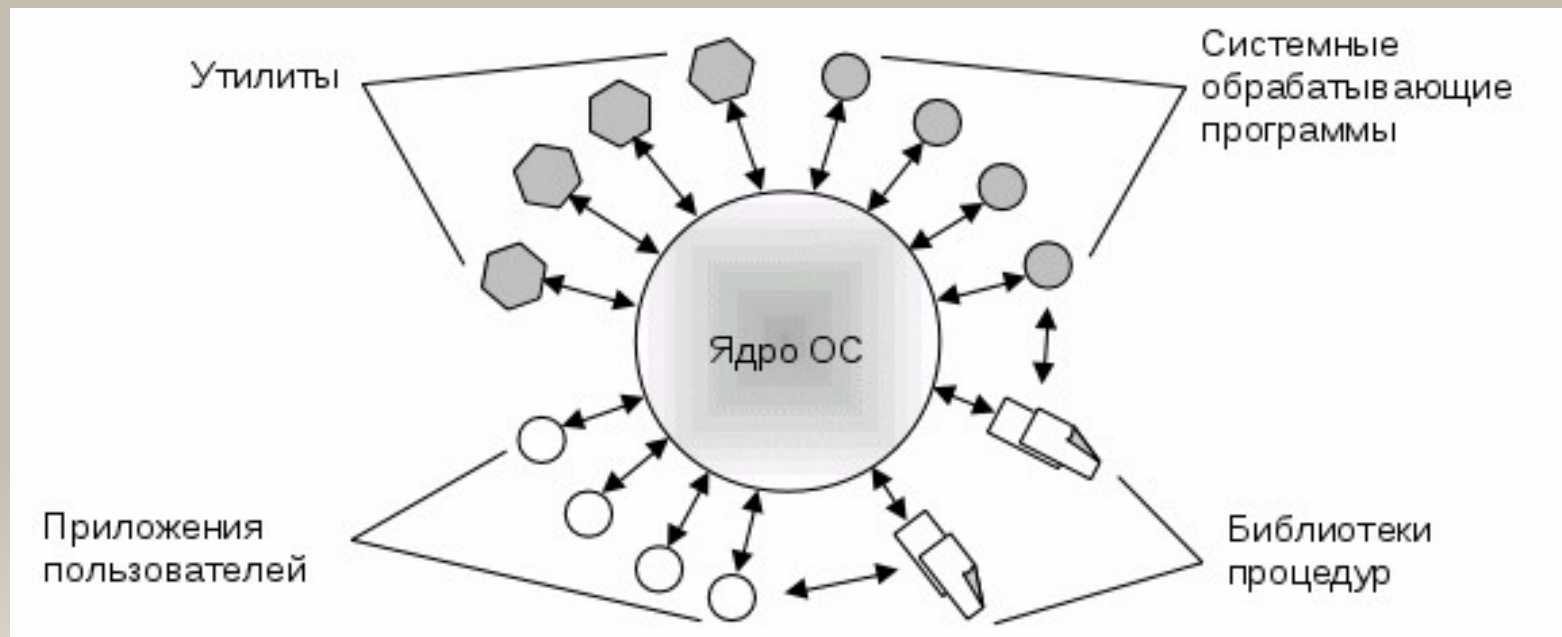
Резидентные

- Для обеспечения высокой скорости работы ОС модули ядра (все или большая часть), являются *резидентными*, т.е. постоянно находятся в оперативной памяти.

Вспомогательные модули

Менее
обязательные
Транзитные

- Выполняют полезные, но менее обязательные функции.
- Обращаются к функциям ядра посредством системных вызовов.
- Вспомогательные модули, в отличие от модулей ядра, являются **транзитными** - загружаются в оперативную память только на время выполнения.

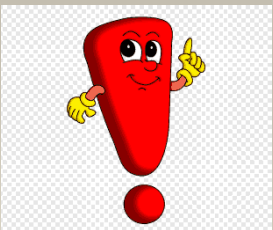


Виды вспомогательных модулей

- ***Утилиты*** – программы, решающие отдельные задачи управления и сопровождения компьютерной системы (сжатие дисков, их проверка, дефрагментация; архивирование, сбор статистики и т.д.);
- ***Системные обрабатывающие программы*** (загрузчики, отладчики, текстовые или графические редакторы);
- ***Библиотеки процедур*** различного назначения для разработки приложений (математические функции, функции ввода-вывода и т.д.);
- ***Программы, предоставляющие дополнительные услуги*** (калькулятор, некоторые игры).

Привилегированный и пользовательский режим

- **Пользовательский режим** (*user mode*) – для работы приложений (недоступны команды процессора, связанные с управлением аппаратным обеспечением, защитой оперативной памяти, переключением режимов работы процессора).
- **Привилегированный режим**, он же – **режим ядра** (*kernel mode*) – для работы ОС или ее частей (процессор может выполнять все возможные команды).



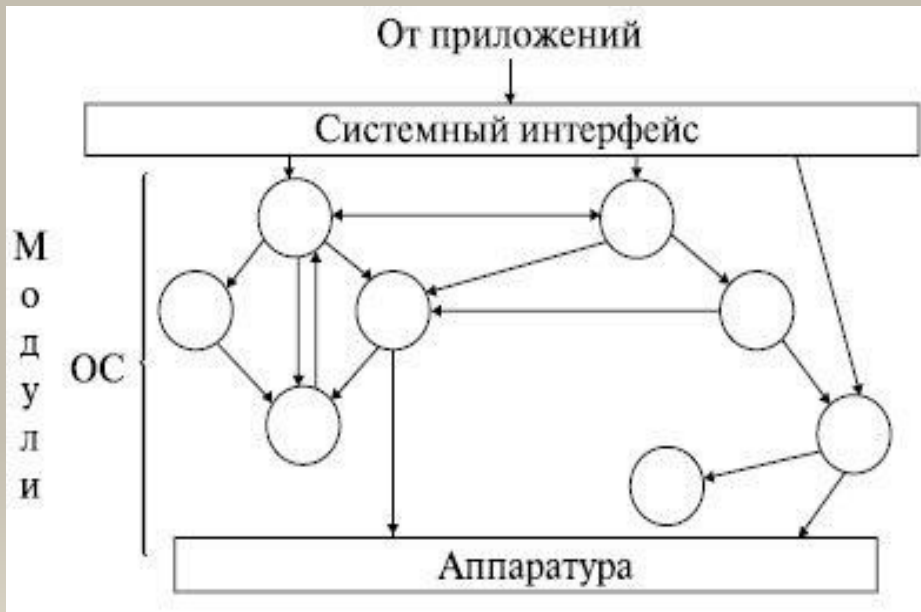
Понятие «ядро» и «привилегированный режим»
тесно связаны

Виды ЯДРА ОС



Монолитное ядро

- В монолитном ядре реализуются все основные функции операционной системы, и оно является, по сути, единой программой, представляющей собой совокупность процедур - большой набор сервисных функций.
- **Монолит** – все вместе, все библиотеки, сервисные функции в одном ядре.



Монолитное ядро содержит следующие базовые элементы:

- Планирование процессов
- Управление файловой системой
- Сетевое взаимодействие
- Драйверы устройств
- Управление памятью

Монолитное ядро + и -

Преимущества:

Производительность — в виду того, что количество переключений из контекста режима пользователя в режим ядра сведено к минимуму.

Недостатки:

Неустойчивость к сбоям — так как все базовые элементы и их работа выполняются в режиме ядра, и если хотя бы в одном модуле или блоке ядра произойдет какой-либо сбой, то ему будет подвержена вся ОС (все ядро), вариантов других нет, закончится все — перезапуском ОС.

Архитектура ОС, основанная на *привилегированном ядре* и приложениях *пользовательского режима* является **КЛАССИЧЕСКОЙ = МНОГОСЛОЙНЫМ** подход

- *Универсальный и эффективный способ декомпозиции сложных систем, базирующийся на следующих положениях:*

- Система представляется как иерархия слоев.
- Функции нижележащего слоя являются *примитивами* для построения более сложных функций вышележащего слоя.
- Взаимодействие слоев осуществляется через посредство функций *межслойного интерфейса*.
- Отдельный модуль может либо выполнить свою работу самостоятельно, либо обратиться к другому модулю своего слоя, либо обратиться к нижележащему слою через межслойный интерфейс.

Архитектура ОС, основанная на *привилегированном ядре* и приложениях пользовательского режима является **КЛАССИЧЕСКОЙ = МНОГОСЛОЙНОЙ** подход

- При таком подходе разработка системы осуществляется сверху вниз, от целей системы к их реализации.
- Сначала определяются функции слоев и межслойные интерфейсы, задающие общую структуру системы, а затем разрабатываются модули внутри слоев.



Многослойная структура ядра ОС

- Многослойный подход применим и к структуре ядра как сложного многофункционального комплекса



Многослойная структура ядра ОС

- **Средства аппаратной поддержки ОС** – аппаратные средства, прямо участвующие в организации вычислительных процессов: средства поддержки привилегированного режима, система прерываний, переключение контекстов процессов, трансляция адресов, защита памяти и т.п.
- **Машино-зависимые модули** – программные модули, в которых отображается специфика аппаратной платформы компьютера. В идеале этот слой полностью экранирует* вышележащие слои от особенностей аппаратуры, т.е. позволяет делать модули вышележащих слоев машинно-независимыми.
- На уровне HAL работа с устройством определенного типа (накопитель, видеоплата, мышь и т.п.) всегда описывается при помощи одного и того же заранее определенного набора функций. В случае, если устройство имеет иной набор функций (например, устаревший 3d-ускоритель может не поддерживать многих современных функций), драйвер обязан эмулировать* стандартные функции с тем, чтобы ОС могла не заботиться о том, какое конкретно устройство установлено.

* *Экранировать* - предохранять от посторонних воздействий

* *Эмуляция* — один из способов электронного архивирования устаревающих вычислительных систем.



Многослойная структура ядра ОС

- **Базовые механизмы ядра.** Модули этого слоя не принимают решений о распределении ресурсов, а только обрабатывают принятые на более высоком уровне решения. Выполняются наиболее примитивные операции ядра: программное переключение контекстов процессов, перемещение страниц между памятью и диском, диспетчеризация прерываний и т.п.
- **Менеджеры ресурсов.** Модули этого уровня реализуют управление основными ресурсами системы. Группировка модулей в менеджеры обычно осуществляется по функциям основных подсистем ОС: выделяются менеджеры процессов, ввода-вывода и файловой системы (могут быть объединены), оперативной памяти.
- **Интерфейс системных вызовов.** Взаимодействует непосредственно с приложениями и системными утилитами, образуя прикладной программный интерфейс ОС (API).

РЕЗЮМЕ Многослойной / классической / многоуровневой архитектуры

- Р Все компоненты ОС разделяются на модули, выполняющие основные функции ОС (ядро), и модули, выполняющие вспомогательные функции ОС.
- Р Вспомогательные модули оформляются либо в виде приложений, либо в виде библиотек процедур и функций.
- Р Вспомогательные модули являются транзитными (загружаются в оперативную память только на время выполнения). Модули ядра – резидентными (постоянно находящиеся в оперативной памяти).
- Р Устойчивость ОС повышается путем выполнения функций ядра в привилегированном режиме, а вспомогательных модулей ОС и пользовательских приложений - в пользовательском.

Многослойная классическая многоуровневая *архитектура* ОС **не лишена своих проблем.**

1. Дело в том, что значительные изменения одного из уровней могут иметь трудно предвидимое влияние на смежные уровни.
2. Кроме того, многочисленные взаимодействия между соседними уровнями усложняют обеспечение безопасности.

Поэтому, как *альтернатива* классическому варианту архитектуры ОС, часто используется **МИКРОЯДЕРНАЯ** архитектура ОС.

МИКРОЯДРО / МИКРОЯДЕРНАЯ архитектура

Микроядро

- ядро, содержащее только самые необходимые функции.

Идея:

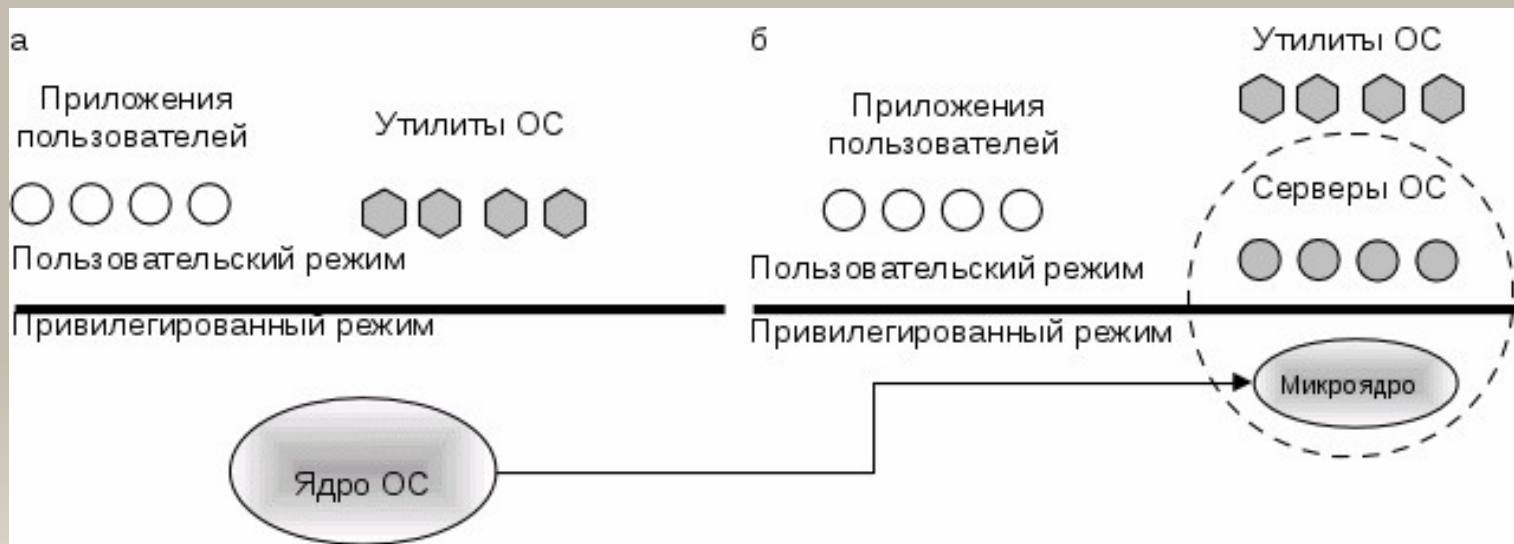
- минимизировать само ядро, вынести как можно функциональности в режим пользователя (т.е. исполнять эту функциональность в виде обычных процессов).

Многие сервисы
становятся
пользовательскими
процессами:

- Драйверы устройств,
- Файловые системы,
- Менеджер виртуальной памяти,
- Оконные системы графического интерфейса пользователя,
- Службы безопасности
- Данный подход популяризован ядром MACH («МАК»)
- На основе MACH сделаны, среди прочих, Mac OS X (комп. Apple), GNUHurd.

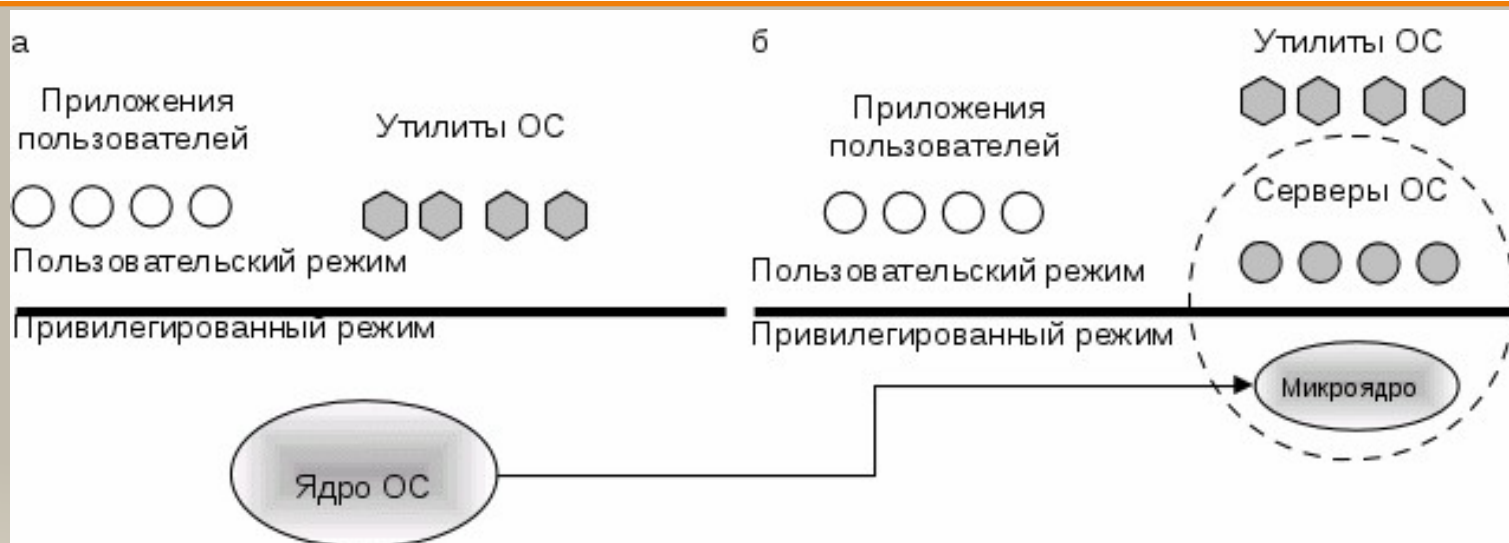
Суть МИКРОЯДЕРНОЙ архитектура

- В привилегированном режиме остается работать только очень небольшая часть ОС, называемая микроядром.
- Микроядро защищено от остальных частей ОС и приложений. В его состав входят машинно-зависимые модули, а также модули, выполняющие базовые механизмы обычного ядра.
- Все остальные более высокоуровневые функции ядра оформляются как модули, работающие в пользовательском режиме. Так, менеджеры ресурсов, являющиеся неотъемлемой частью обычного ядра, становятся "периферийными" модулями, работающими в пользовательском режиме.
- Таким образом, в архитектуре с микроядром традиционное расположение уровней по вертикали заменяется горизонтальным.
- Между собой они взаимодействуют как равноправные партнеры с помощью обмена сообщениями, которые передаются через микроядро.



Концепция МИКРОЯДЕРНОЙ архитектуры

- В привилегированном режиме работает только небольшая часть ОС – микроядро, защищенное от остальных частей ОС приложений.
- В состав функций микроядра включаются те функции ОС, которые трудно или невозможно выполнить в пространстве пользователя - это функции слоя базовых механизмов обычного ядра и ниже.
- Остальные, высокоуровневые функции ядра оформляются в виде приложений, работающих в пользовательском режиме. Соотношение классической и микроядерной архитектур приведено на ниже



(Перенос функций ядра в пользовательское пространство: а – классическая архитектура, б – микроядерная архитектура)

Реализация системного вызова в ОС с микроядерной архитектурой

- Однозначного решения о переносе в пользовательский режим тех или иных системных функций не существует. В общем случае как пользовательские приложения оформляются многие менеджеры ресурсов.
- По определению, основным назначением такого приложения является обслуживание запросов других приложений (создание процесса, выделение памяти, проверка прав доступа и т.д.). Поэтому менеджеры ресурсов, вынесенные в пользовательский режим, называются *серверами ОС*. Одной из главных задач микроядра является поддержка взаимодействия серверов.

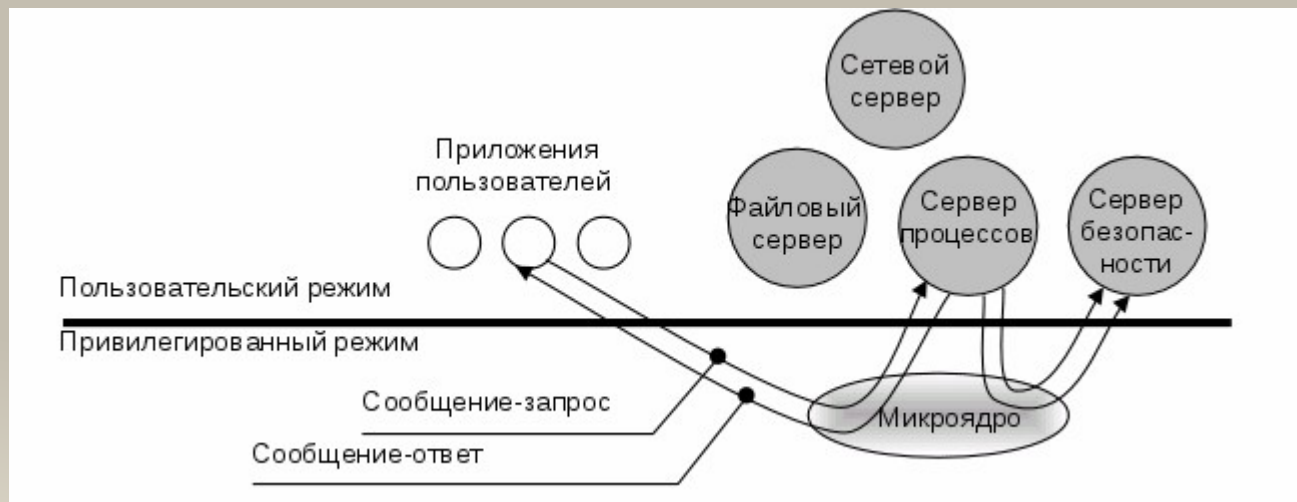


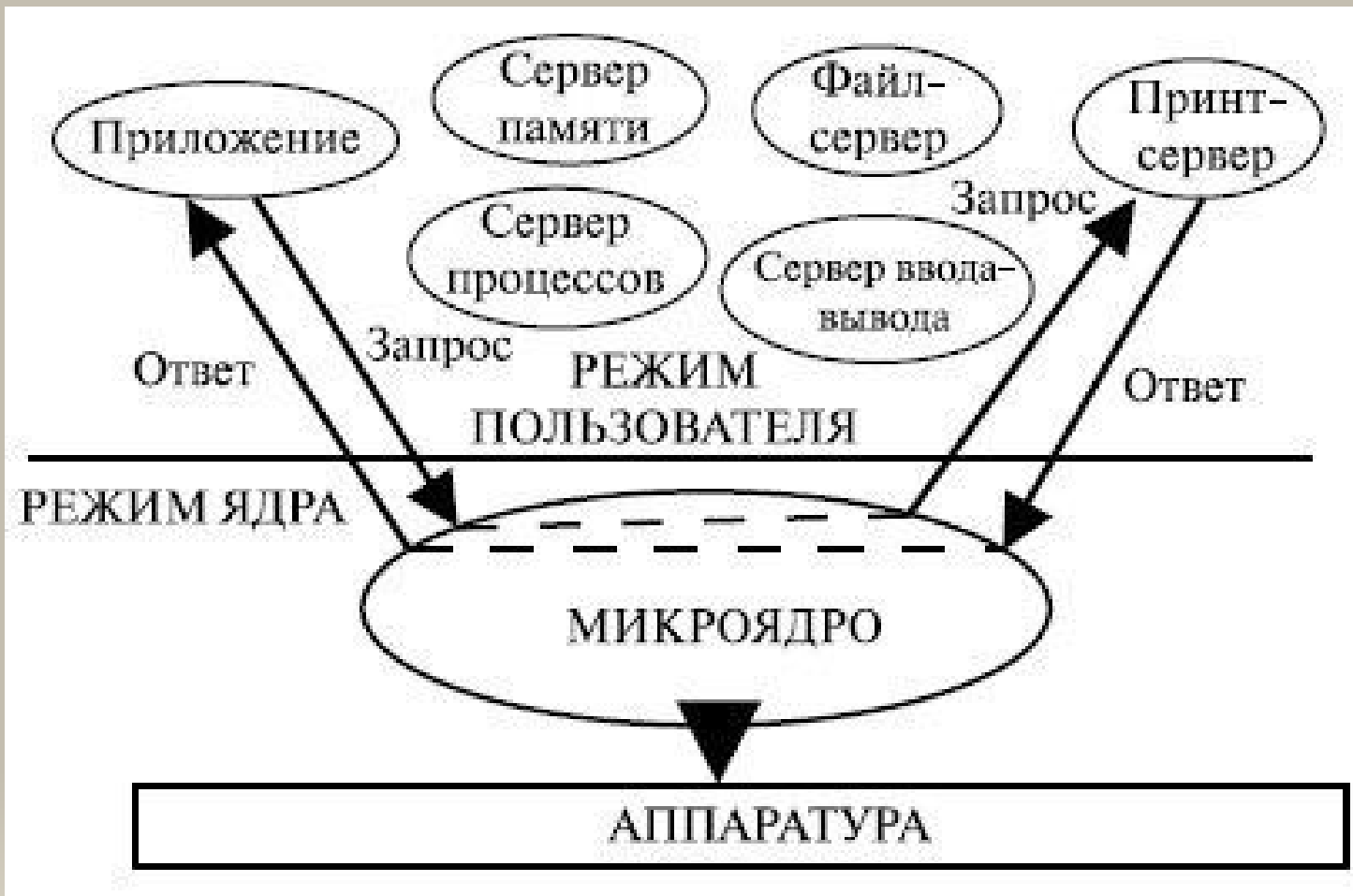
Схема обработки запроса соответствует модели КЛИЕНТ-СЕРВЕР, где микроядро выполняет роль транспортных средств.

- Клиент (прикладная программа либо другой компонент ОС) посылает соответствующему серверу сообщение-запрос на выполнение некоторой функции

- Непосредственная передача этого сообщения серверу невозможна, так как каждое приложение работает в своем адресном пространстве. В качестве посредника выступает микроядро, выполняющееся в привилегированном режиме и имеющее доступ к адресным пространствам всех приложений.

- Микроядро передает сообщение нужному серверу, сервер выполняет запрошенную операцию и результат, снова через посредство микроядра, возвращается клиенту с помощью другого сообщения.

Схематично механизм обращений к функциям ОС, оформленным в виде серверов



Микроядерная архитектура + и -

Преимущества:

ОС, основанные на концепции микроядра, в высокой степени удовлетворяют большинству требований, предъявляемых к современным ОС:

- единообразные интерфейсы;
- простота расширяемости;
- высокая гибкость;
- возможность переносимости;
- высокая надежность;
- поддержка распределенных систем;
- поддержка объектно-ориентированных ОС.

Недостатки:

Основным недостатком микроядерной архитектуры является снижение производительности по сравнению с классическим вариантом. Так, при классической организации выполнение системного вызова требует двух переключений режимов «привилегированный – пользовательский», а при микроядерной – четырех. При обращении к часто используемым функциям работа приложений существенно замедляется.

Обработка системного вызова в микроядерной архитектуре

- Схема смены режимов при выполнении системного вызова в ОС с микроядерной архитектурой выглядит, как показано на рисунке. Из рисунка ясно, что выполнение системного вызова сопровождается четырьмя переключениями режимов



Обработка системного вызова в классической архитектуре

- Повышение устойчивости ОС обеспечивается переходом ядра в привилегированный режим. При этом происходит некоторое замедление выполнения системных вызовов. *Системный вызов* привилегированного ядра инициирует переключение процессора из пользовательского режима в привилегированный, а при возврате к приложению – обратное переключение.



Соотношение классической и микроядерной архитектуры

