Дисциплина

«Основы организации операционных систем и эксплуатации программных средств специального назначения»

Содержание дисциплины

Общие сведения об операционных системах и программном обеспечении

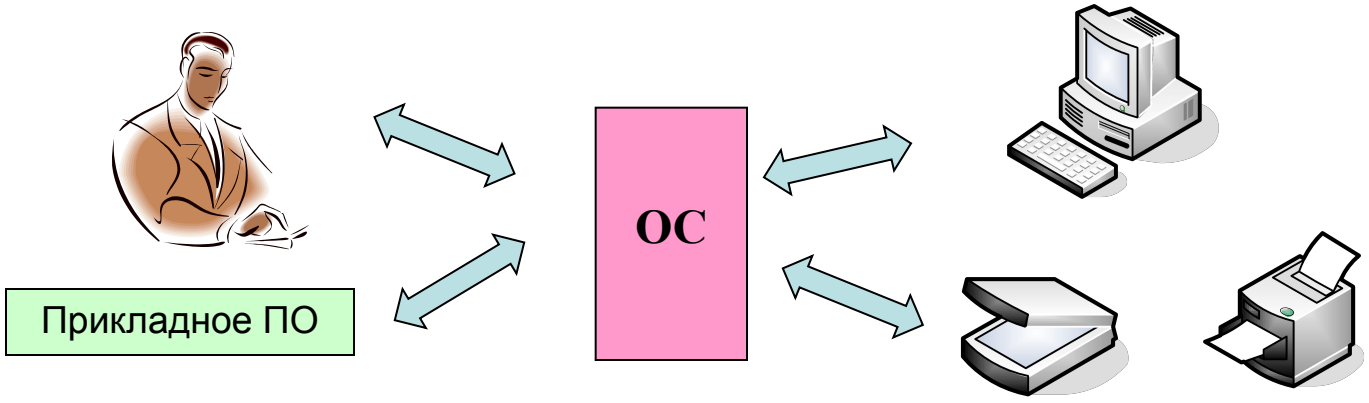
Операционные системы специального назначения

Разработка и применение специального программного обеспечения

Общие сведения об операционных системах и программном обеспечении

ОС ЭВМ - это комплекс взаимосвязанных программ, который действует как интерфейс между:

- приложениями и пользователями с одной стороны и
- аппаратурой ПК с другой стороны.



Группы функций ОС ЭВМ

- предоставление пользователю виртуальной машины:
  - ✔ Пользовательский интерфейс
  - ✔ Интерфейс прикладного программирования
  - ✔ Защита данных и администрирование
- управление ресурсами ПК:
  - ✔ процессоры, ОЗУ, НМД, НМЛ, таймеры и т.д.
  - ✔ принтеры, сетевые устройства, и т.д.

Подсистемы ОС

- По типу локальных ресурсов
  - ✔ Управления процессами
  - ✔ Управления памятью
  - ✔ Управления файлами
  - ✔ Управления внешними устройствами
- По задачам ко всем ресурсам
  - ✔ Подсистема пользовательского интерфейса
  - ✔ Подсистема администрирования
  - ✔ Подсистема защиты информации

Основные функции ОС

- Загрузка приложений в оперативную память и их выполнение;
- Стандартизованный доступ к периферийным устройствам (устройства ввода-вывода)
- Управление оперативной памятью
- Управление доступом к данным на энергонезависимых носителях (Жёсткий дискУправление доступом к данным на энергонезависимых носителях (Жёсткий диск, Компакт-диск и т. д.)
- Пользовательский интерфейс;
- Сетевые операции, поддержка стека протоколов

Дополнительные функции ОС

- Параллельное или псевдопараллельное выполнение задач (многозадачность);
- Взаимодействие между процессами: обмен данными, взаимная синхронизация;
- Защита самой системы, а также пользовательских данных и программ от злонамеренных действий пользователей или приложений;
- Разграничение прав доступа и многопользовательский режим работы (аутентификация, авторизация).

Архитектура ОС

- Какой-либо единой архитектуры ОС не существует
- Существуют универсальные подходы к структурированию
- Наиболее общий подход к структурированию ОС – разделение её модулей на 2 группы:
  - ✔ Ядро – модули, выполняющие основные функции ОС
  - ✔ Вспомогательные модули ОС

Архитектура ОС

- Ядро́ — 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 основополагающий элемент ОС, ядро представляет собой наиболее низкий уровень абстракции для доступа приложений к ресурсам системы, необходимым для его работы. Как правило, ядро предоставляет такой доступ исполняемым процессам соответствующих приложений за счёт использования механизмов межпроцессного взаимодействия и обращения приложений к системным вызовам ОС.

Типы архитектур ядер ОС

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

1. 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
- все части монолитного ядра **работают в одном адресном пространстве**

Монолитное ядро

- для монолитной ОС ядро совпадает со всей системой
- сборка ядра, то есть его компиляция, осуществляется отдельно для каждого компьютера, на который устанавливается операционная система
- старые монолитные ядра **требовали перекомпиляции** при любом изменении состава оборудования. Большинство современных ядер позволяют во время работы подгружать **модули**, выполняющие части функции ядра.
- можно выбрать список оборудования и программных протоколов, поддержка которых будет включена в ядро

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.

2.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. *Поэтому модульные ядра продолжают оставаться монолитными*

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, для обеспечения динамической подгрузки и выгрузки модулей

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

3.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

- большинство составляющих ОС программ являются самостоятельными программами
- взаимодействие между ними обеспечивает специальный модуль ядра, называемый микроядром
- 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
- Windows CE
- QNX, AIX, Minix, Mach
- ChorusOS
- AmigaOS
- MorphOS

4.Экзоядро

- предоставляет лишь **функции для взаимодействия между процессами и безопасного выделения и освобождения ресурсов**
- 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, а не файлов и кластеров, что положительно скажется на быстродействии

Экзоядро (принципы)

1. **Экзоядро не абстрагирует ресурсы**. Это делают непривилегированные прикладные библиотеки - "библиотечные операционные системы" (libOS, library operating system). В рамках каждого конкретного приложения может быть реализована 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": страницы физической и виртуальной памяти, блоки дисковой памяти, кванты времени процессора.

Виртуализация ресурса экзоядром допускается в двух случаях:

- ✔ когда ресурс не может быть распределен без нее,
- ✔ когда требуется большая эффективность эксплуатации ресурса.

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

Экзоядро

Пример:
XOK/ExOS (Дистрибутив amsterdam.lcs.mit.edu/exo).

XOK - небольшой безопасный распределитель ресурсов x86 совместимых машин (применяется механизм "безопасных связей" (secure bindings), отделяющий процедуру получения разрешения на эксплуатацию ресурса от фактического его использования)

ExOS - библиотечная операционная система - не что иное, как "Unix в библиотеке". Реализации Unix-абстракций были вычленены из ядра ОС 4.4BSD, благодаря чему они позволяют запускать без модификации многие приложения, включая достаточно сложные, вроде GCC, Csh, Perl, Vi, Telnet.

Экзоядро

Достоинства:

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

Недостатки:

В стадии экспериментов, остаётся академической игрушкой

5.Наноядро

Наноядро — архитектура ядра ОС, в рамках которой крайне упрощённое и минималистичное ядро выполняет лишь одну задачу — обработку аппаратных прерываний, генерируемых устройствами компьютера.

После обработки прерываний от аппаратуры наноядро, в свою очередь, посылает информацию о результатах обработки (например, полученные с клавиатуры символы) вышележащему программному обеспечению при помощи того же механизма прерываний.

Примеры:

KeyKOS — самая первая ОС на наноядре (1983г).

Symbian OS v8.1- S60 2rd Edition (Nokia N70, Nokia N90)

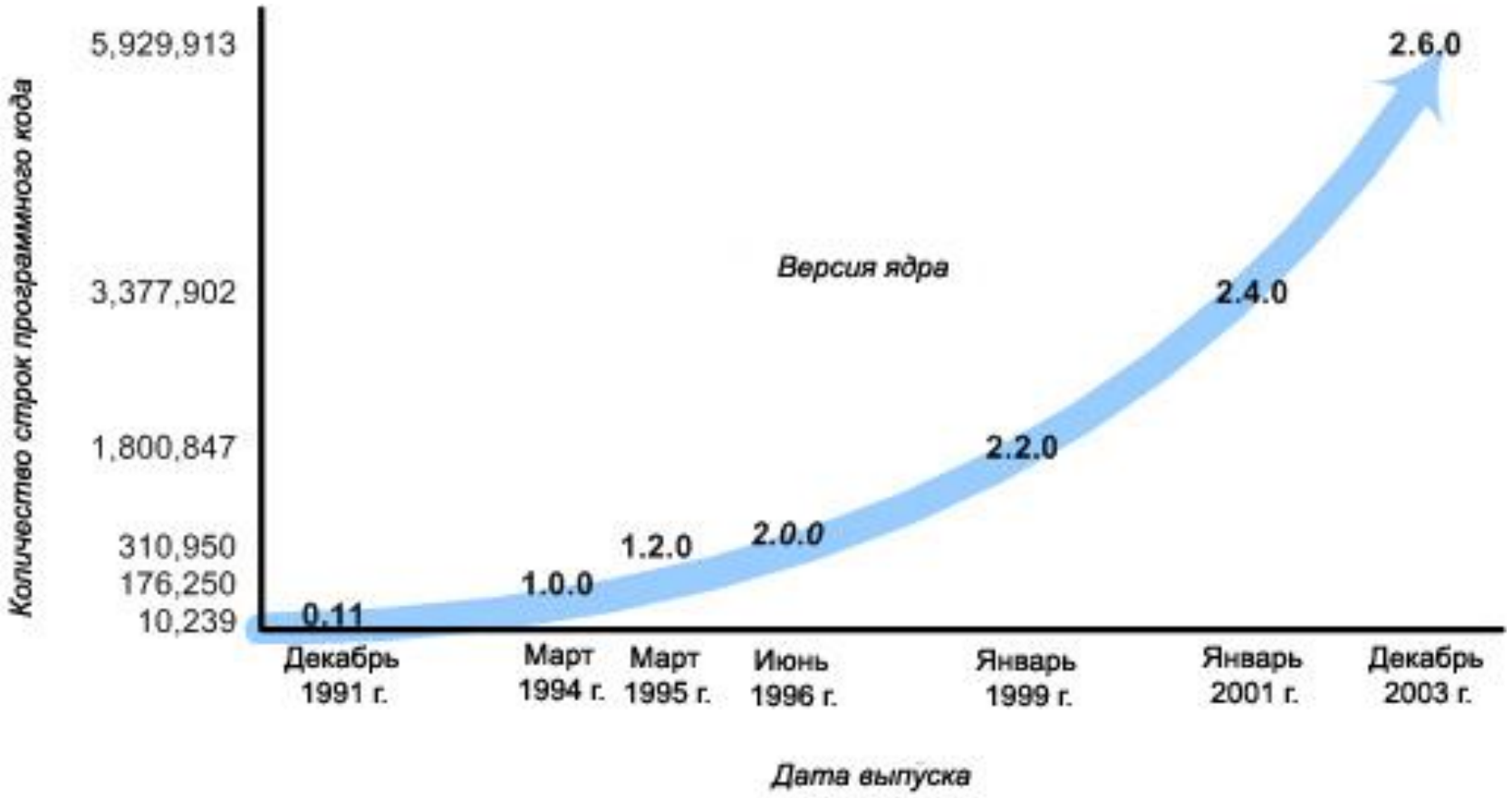
Symbian OS v9.1- S60 3rd Edition (Nokia N96)

Общие сведения об операционных системах и программном обеспечении

1. Общие сведения о LINUX

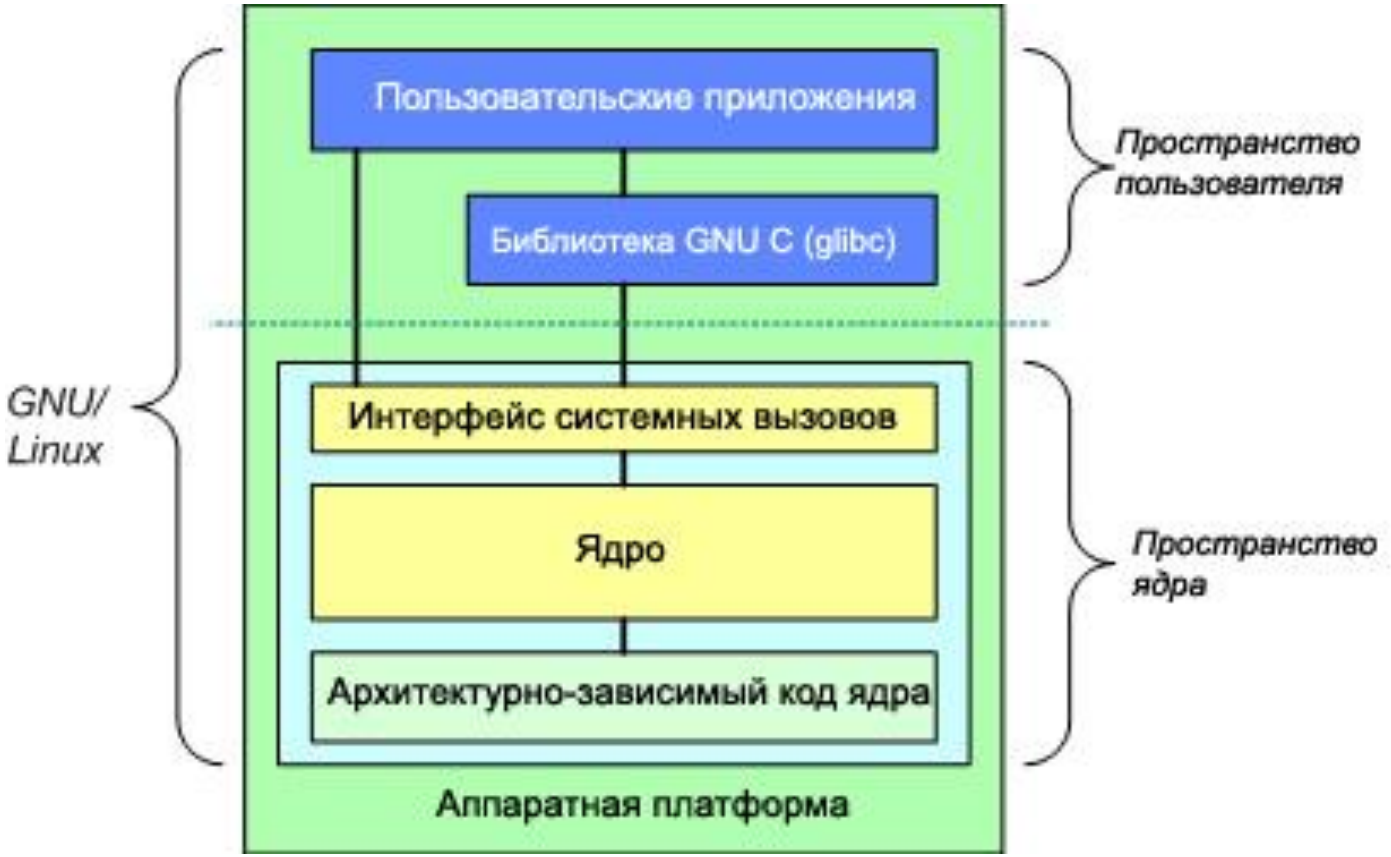
Ядро LINUX

Краткая история основных выпусков ядра Linux



Ядро LINUX

Архитектура операционной системы GNU/Linux



Ядро LINUX

Вариант архитектуры ядра Linux

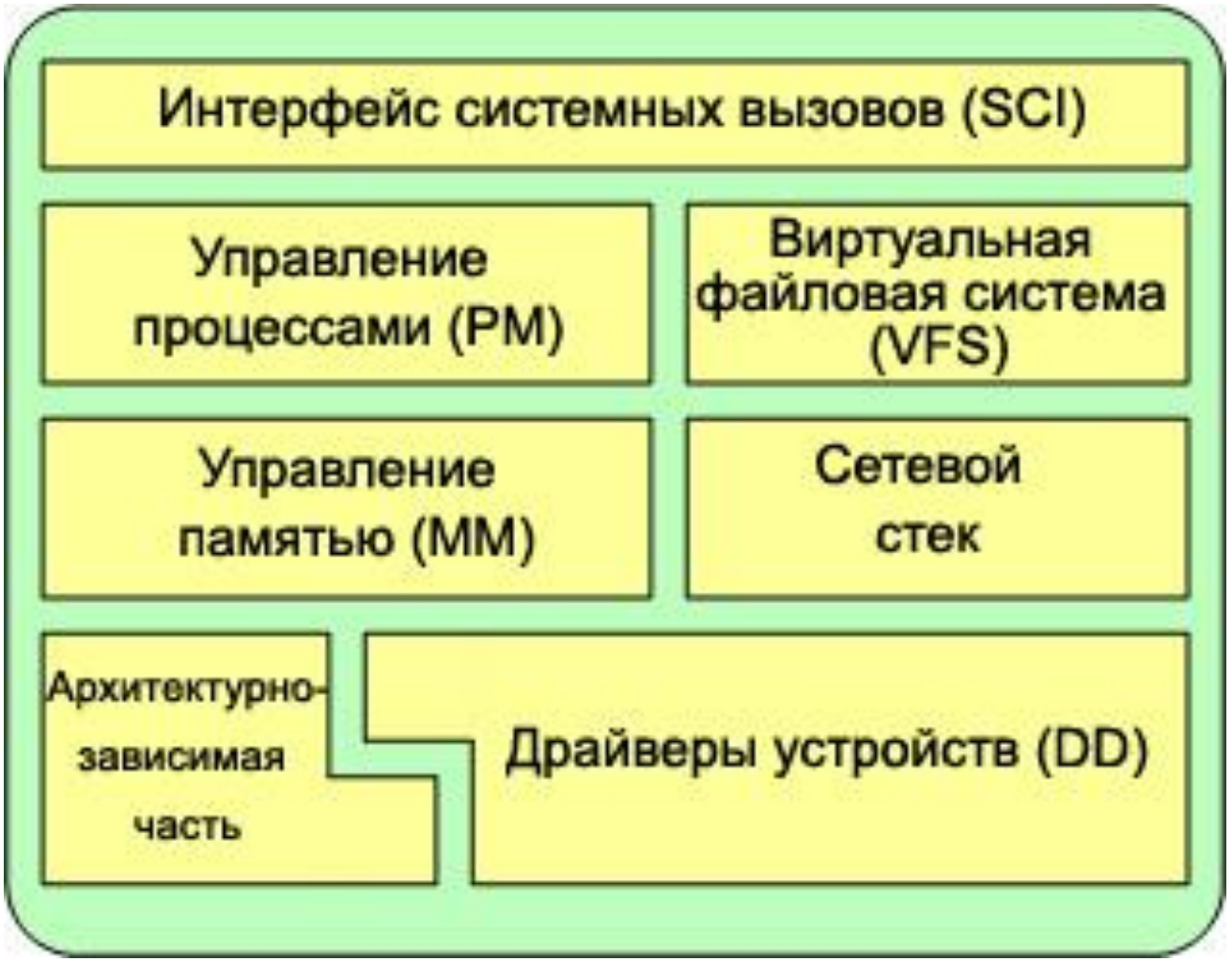
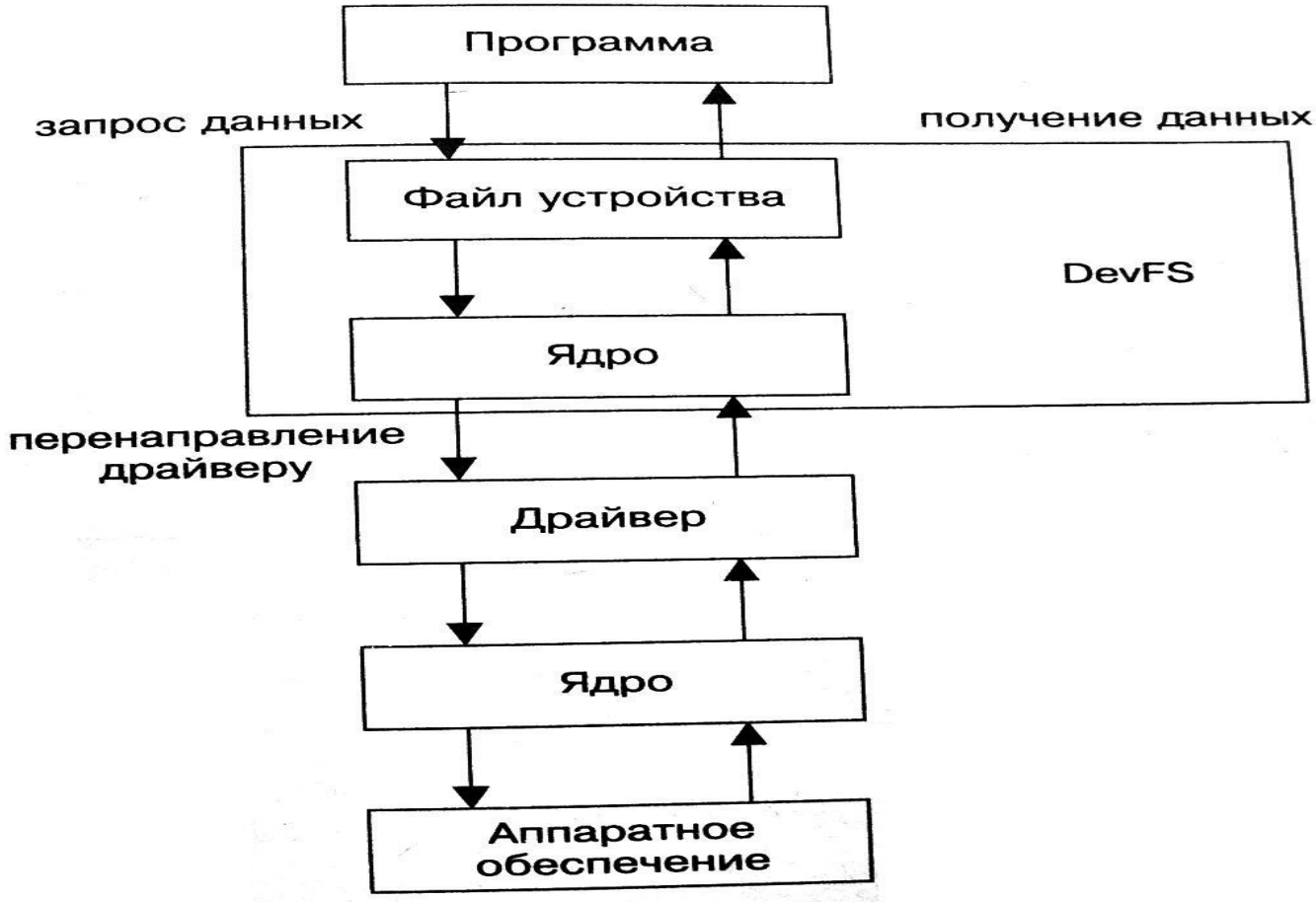


Схема получения данных из файлов устройств



Этапы загрузки

- Загрузка и инициализация ядра
- Обнаружение и конфигурирование устройств
- Создание процессов ядра
- Действия оператора (только при ручной загрузке)
- Выполнение сценариев запуска
- Работа в многопользовательском режиме

2. ОС и ПО в ВС РФ

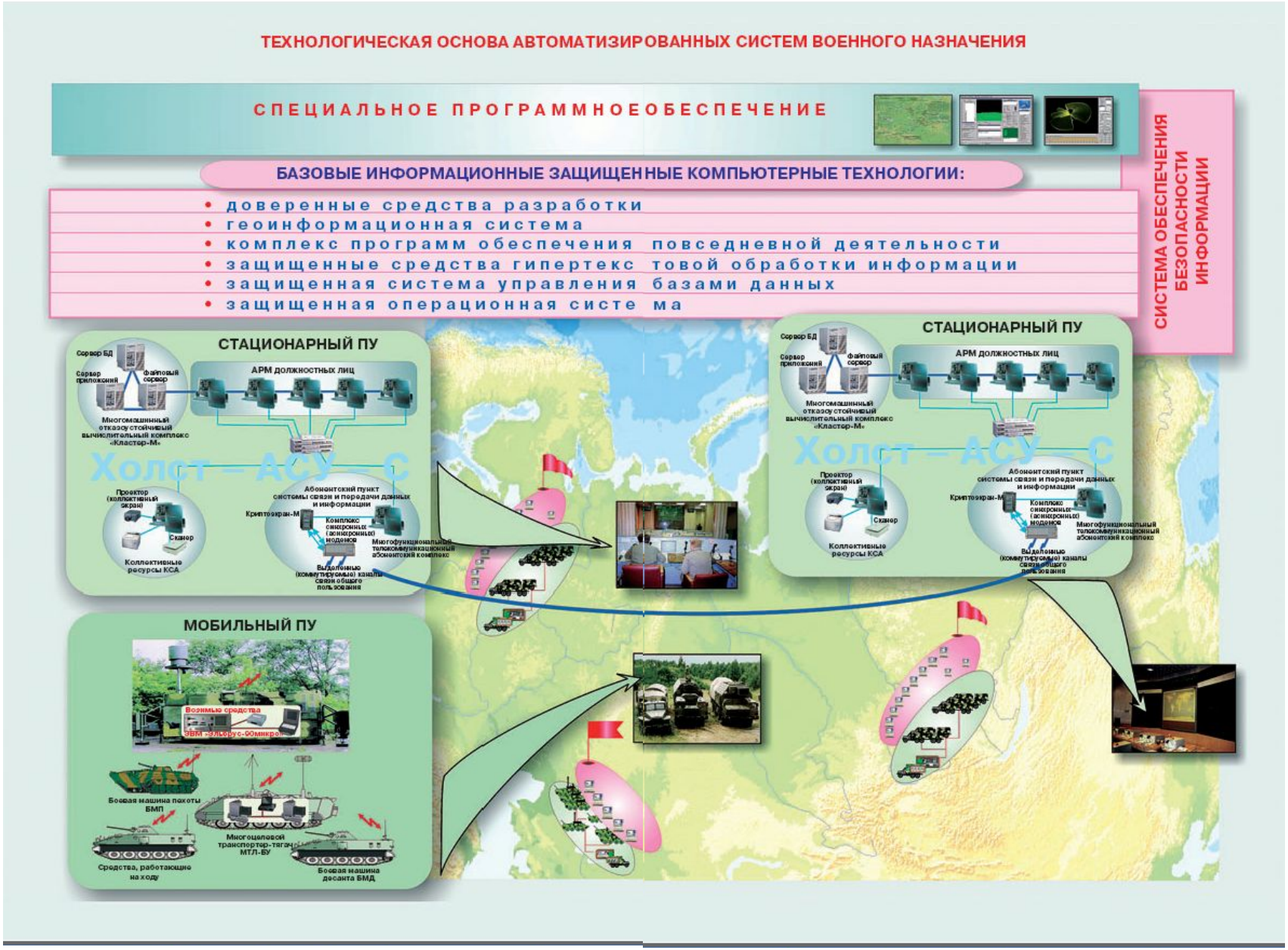
«Закупка программных средств является угрозой информационной безопасности»

"Программные продукты можно разделить на две категории: системные и прикладные. Я говорю только о сугубо системных вещах — это ОС и СУБД. Я не трогаю пакеты Word, Excel и прочие. Они не угрожают информационной безопасности и не могут разрушить базу данных"

Сергей Ковалевский,

дтн, академик РАЕН

Технологическая основа АС в ВС РФ



Отечественные ОС в ВС РФ

- МСВС (Мобильная Система Вооруженных Сил) – ОС общего назначения (ВНИИНС)
- ОМОНИМ - для построения защищенных стационарных систем для мэйнфреймов (ВНИИНС)
- ОЛИВИЯ - для построения защищенных автоматизированных систем мобильного базирования (ВНИИНС)
- PTS-DOS ("Физтех-софт")

Отечественные СУБД в ВС РФ

- "Паллада« (ВНИИНС) - защищенная объектно-ориентированная СУБД для АСУ ВС РФ мобильного базирования (работает с ОС МСВС и ОЛИВИЯ)
- "Линтер-ВС (ВНИИНС) - реляционная СУБД для АСУ ВС РФ (работает с ОС МСВС)

МСВС 3.0 - общие сведения

- МСВС 3.0 — защищенная многопользовательская многозадачная ОС с разделением времени, разработанная на основе Linux
- Платформы - Intel, MIPS и SPARC
- Особенность - встроенные средства защиты от несанкционированного доступа, удовлетворяющие требованиям ФСТЭК России по классу 2 средств вычислительной техники

МСВС 3.0 - общие сведения

Файловая система поддерживает:

- имена файлов длинной до 256 символов (возможность создания русскоязычных имен файлов и каталогов)
- символьные ссылки
- систему квот и списки прав доступа
- возможность монтирования файловых систем FAT и NTFS, а также ISO-9660 (компакт-диски)

МСВС 3.0 - общие сведения

Графическая система:

- на основе X Window
- поставляются два оконных менеджера - IceWM и KDE
- большинство программ в МСВС ориентировано на работу в графической среде