



МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ





Лекция № 2

Тема 1: Архитектура ОС

по курсу "Операционные системы"

Тема лекции: Архитектура Windows-подобных операционных систем

Лектор:

*Доцент кафедры ИС
кандидат технических наук, доцент
Голубничий Дмитрий Юрьевич*

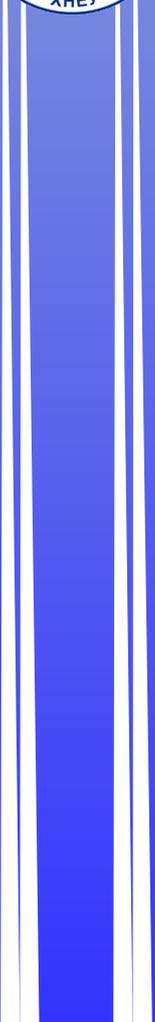
УЧЕБНЫЕ ВОПРОСЫ:

1.

Архитектура ОС Windows.

2.

Архитектура ОС ReactOS.



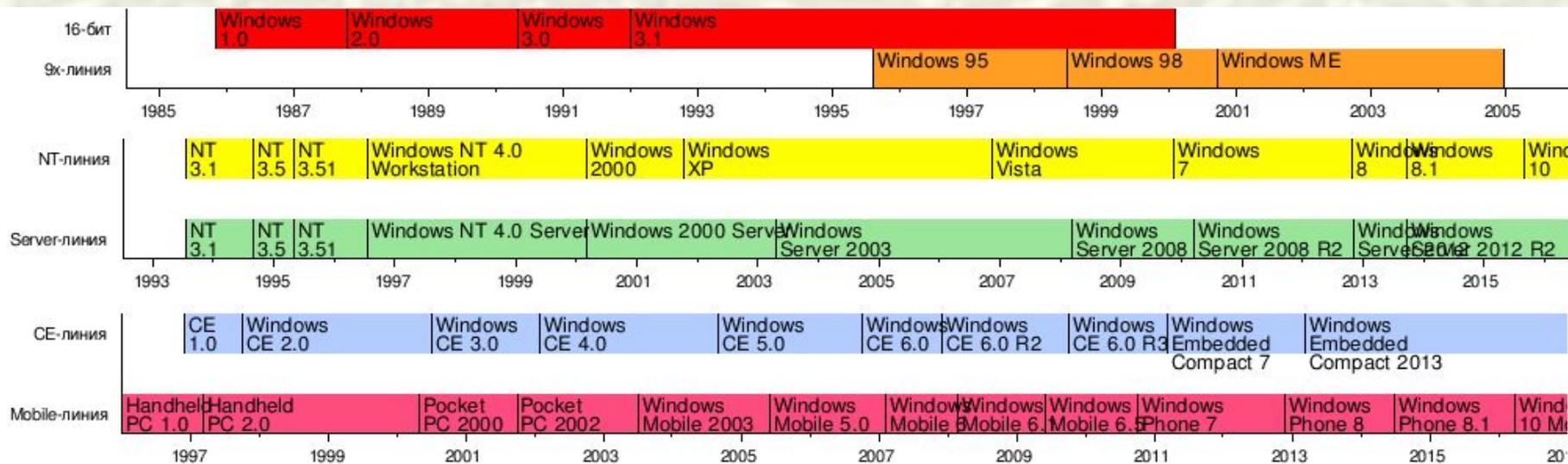
Windows

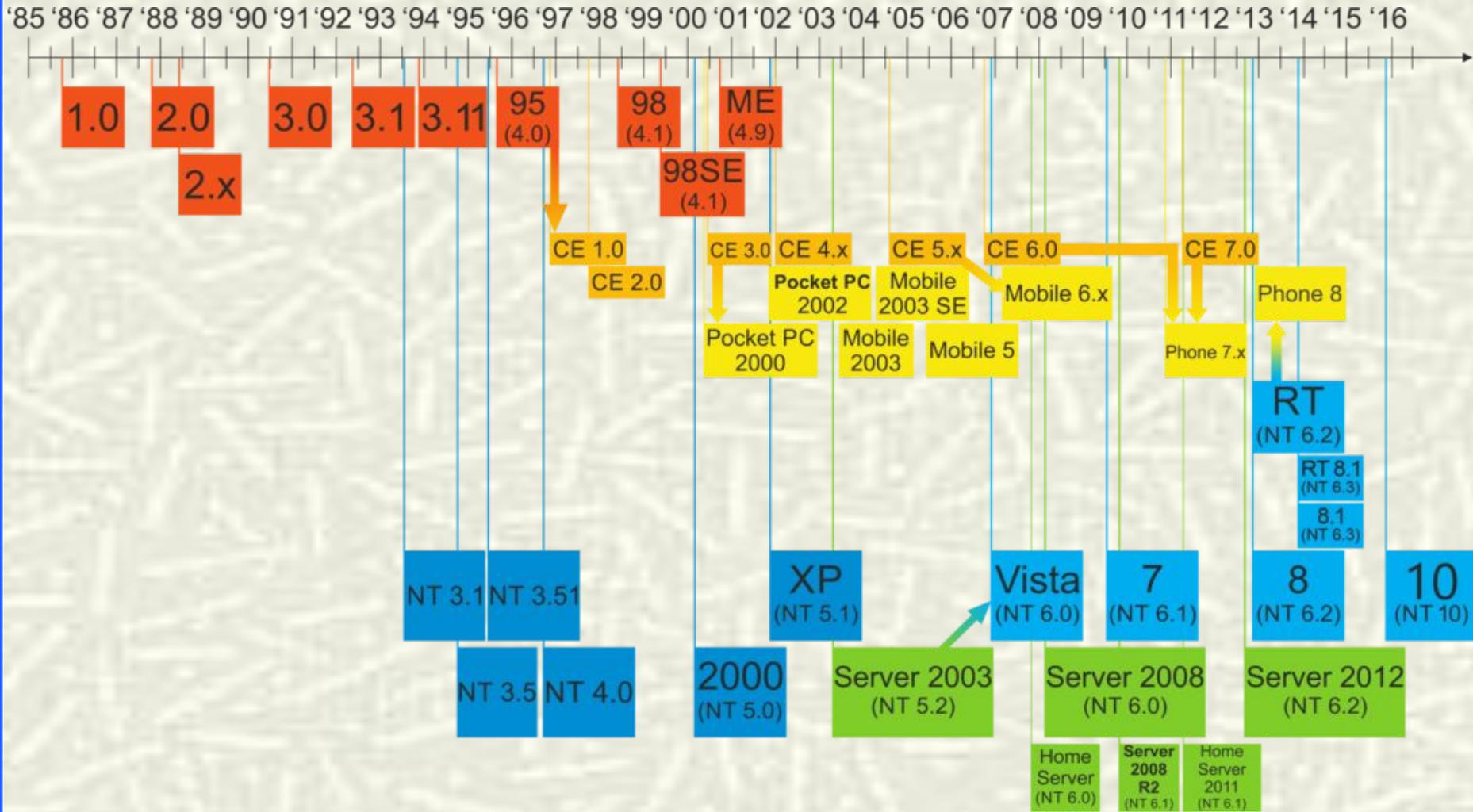


Эмблема Microsoft Windows

Последние 20 лет Windows — самая популярная (90 %) операционная система на рынке персональных компьютеров. Операционные системы Windows работают на платформах x86, x86-64, ARM, IA-64 (server). Существовали также версии для DEC Alpha, MIPS и PowerPC.

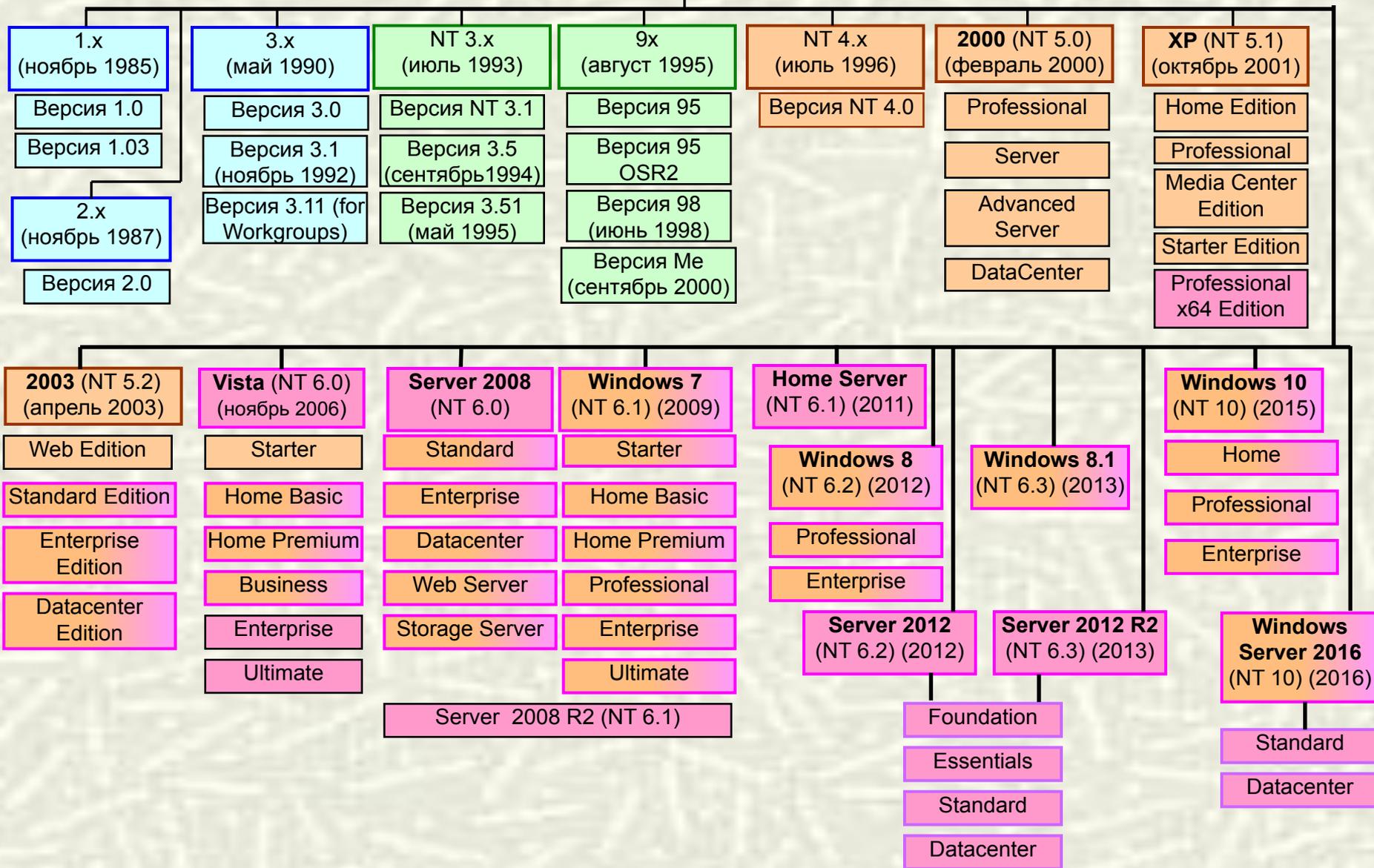
1. ОС WINDOWS





Клоны Windows

1. ОС WINDOWS



□ – 16-ти разрядные ОС

□ – 32-ти разрядные ОС с поддержкой
16-ти разрядности

□ – 32-ти разрядные ОС

□ – 64-ти разрядные ОС

□ – 32-ти и 64-ти разрядные ОС

Совместимость – возможность операционной системы выполнять приложения, разработанные для других операционных систем.

Виды совместимости:

1. На двоичном уровне (уровень исполняемой программы).
2. На уровне исходных текстов (уровень исходного модуля).

Вид совместимости определяется:

1. Архитектурой центрального процессора.
2. Интерфейсом прикладного программирования (API).
3. Внутренней структурой исполняемого файла.
4. Наличием соответствующих компиляторов и библиотек.

Способы достижения совместимости:

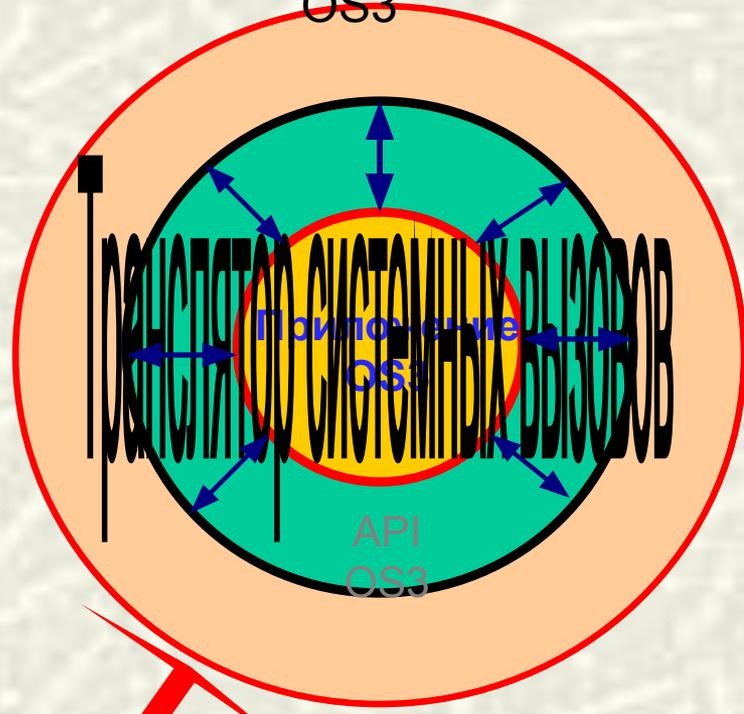
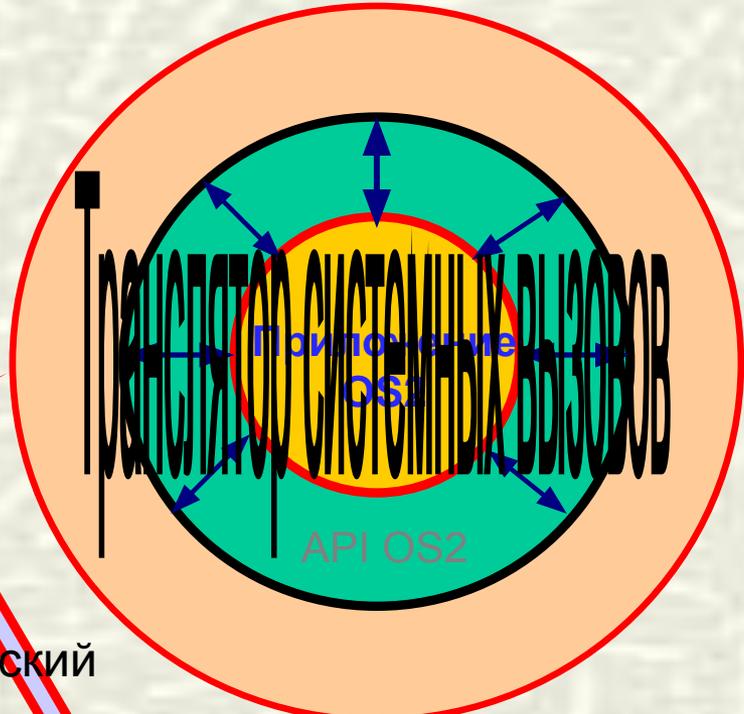
1. Эмуляция двоичного кода.
2. Трансляция библиотек.
3. Создание множественных прикладных сред различной архитектуры.

1. OS WINDOWS

Прикладная среда OS2

Прикладная среда OS3

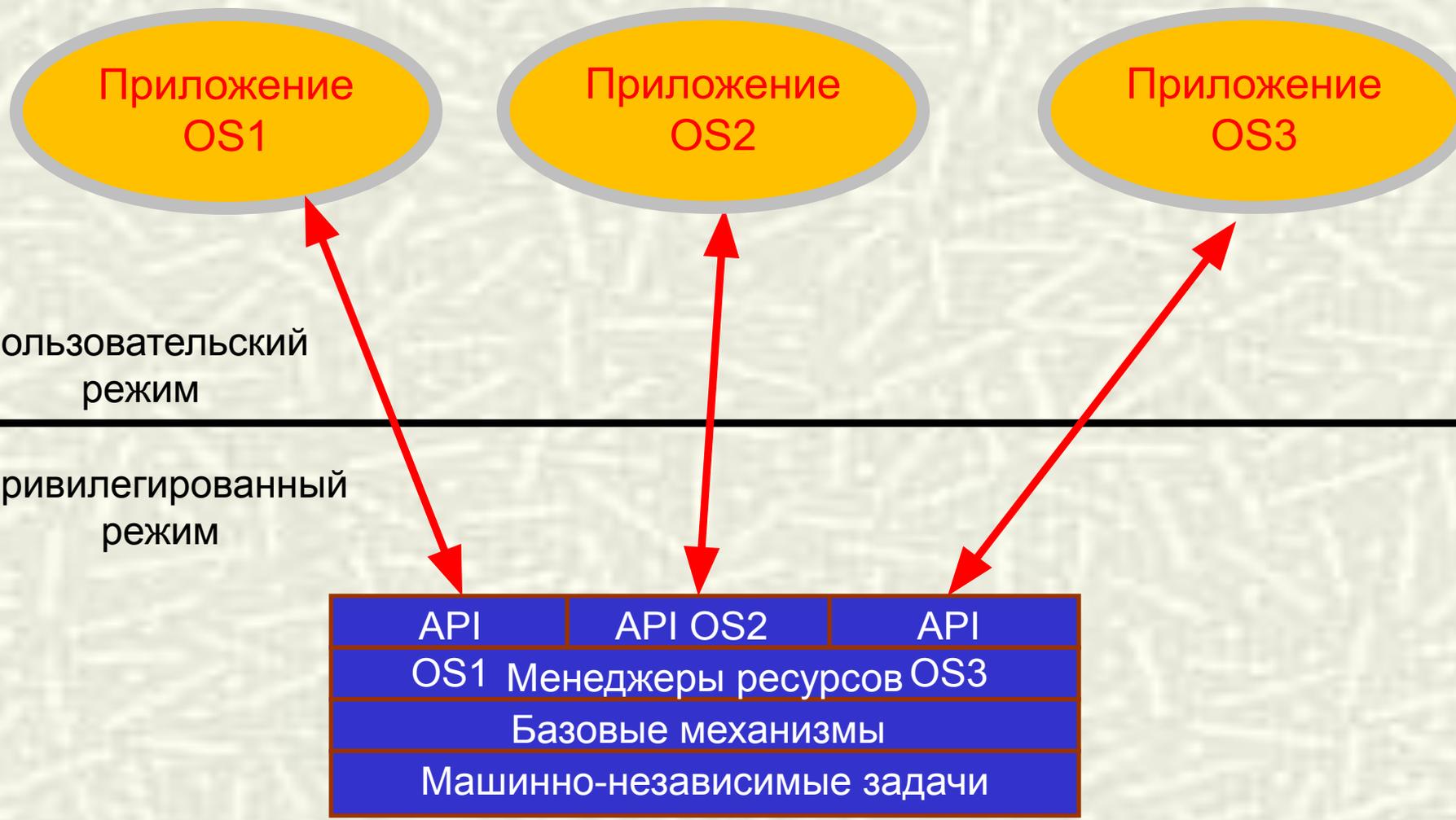
Обычное приложение OS1

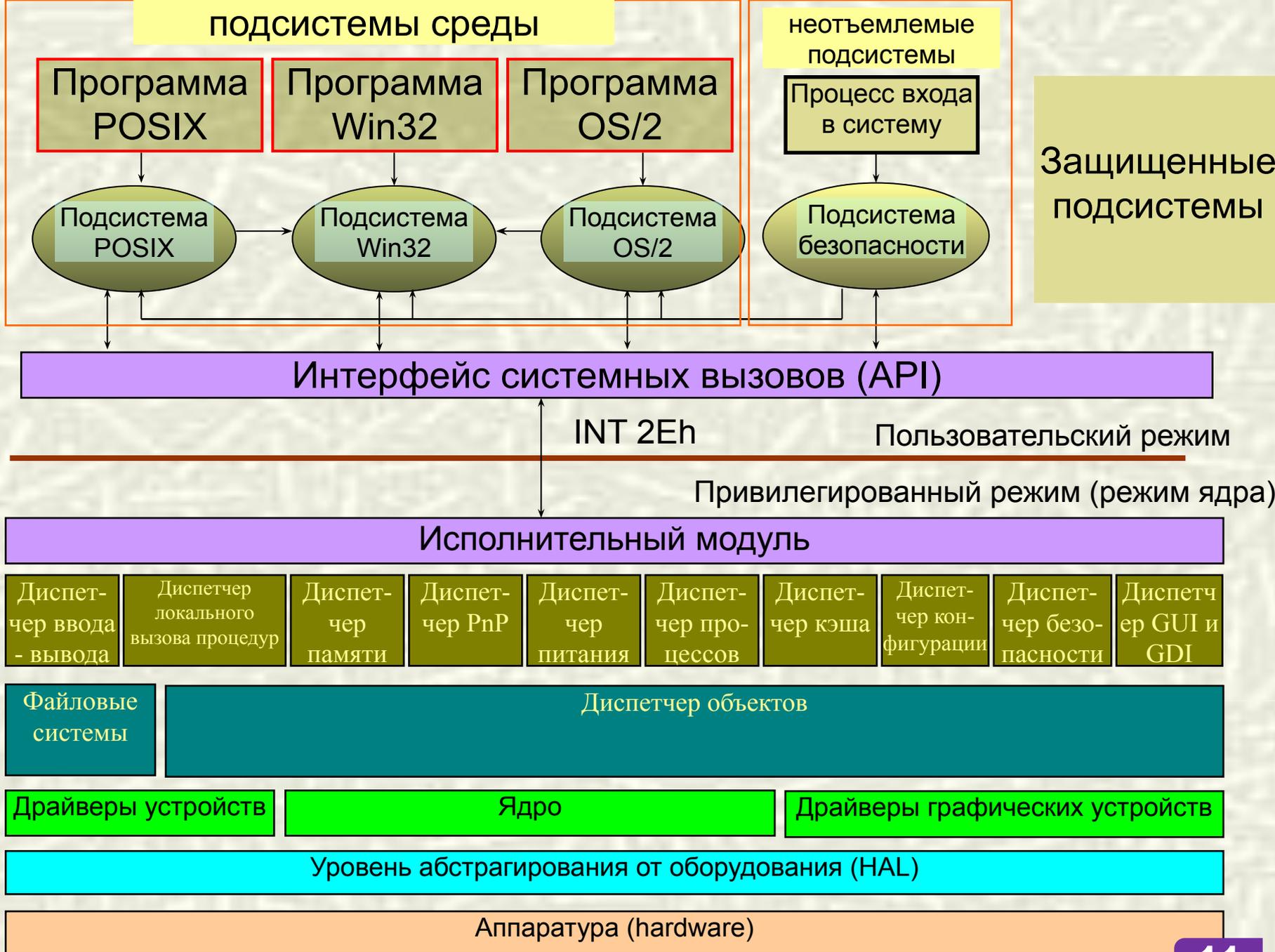


Пользовательский режим

Привилегированный режим

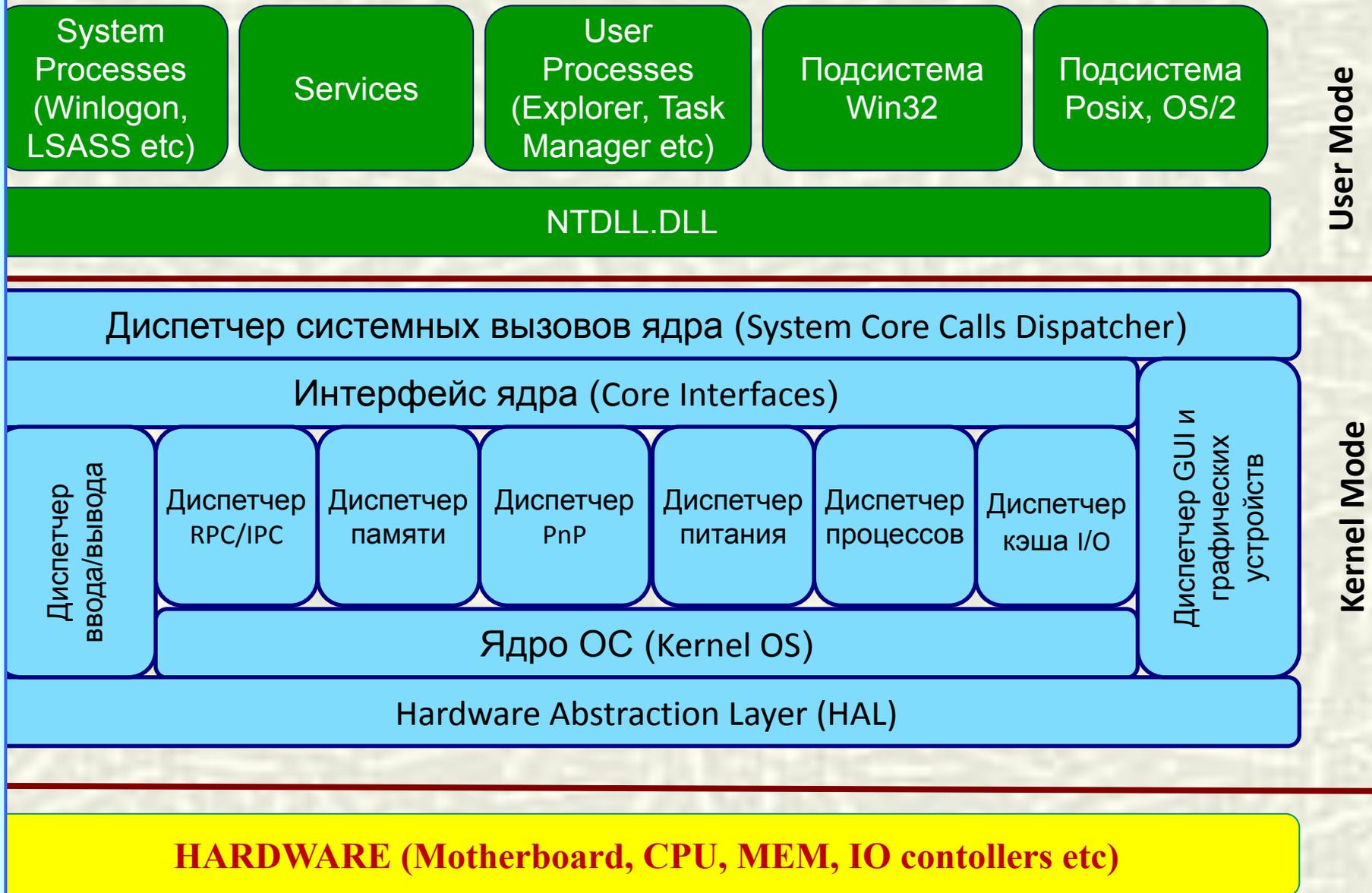
- API OS1
- Менеджеры ресурсов
- Базовые механизмы
- Машинно-независимые задачи





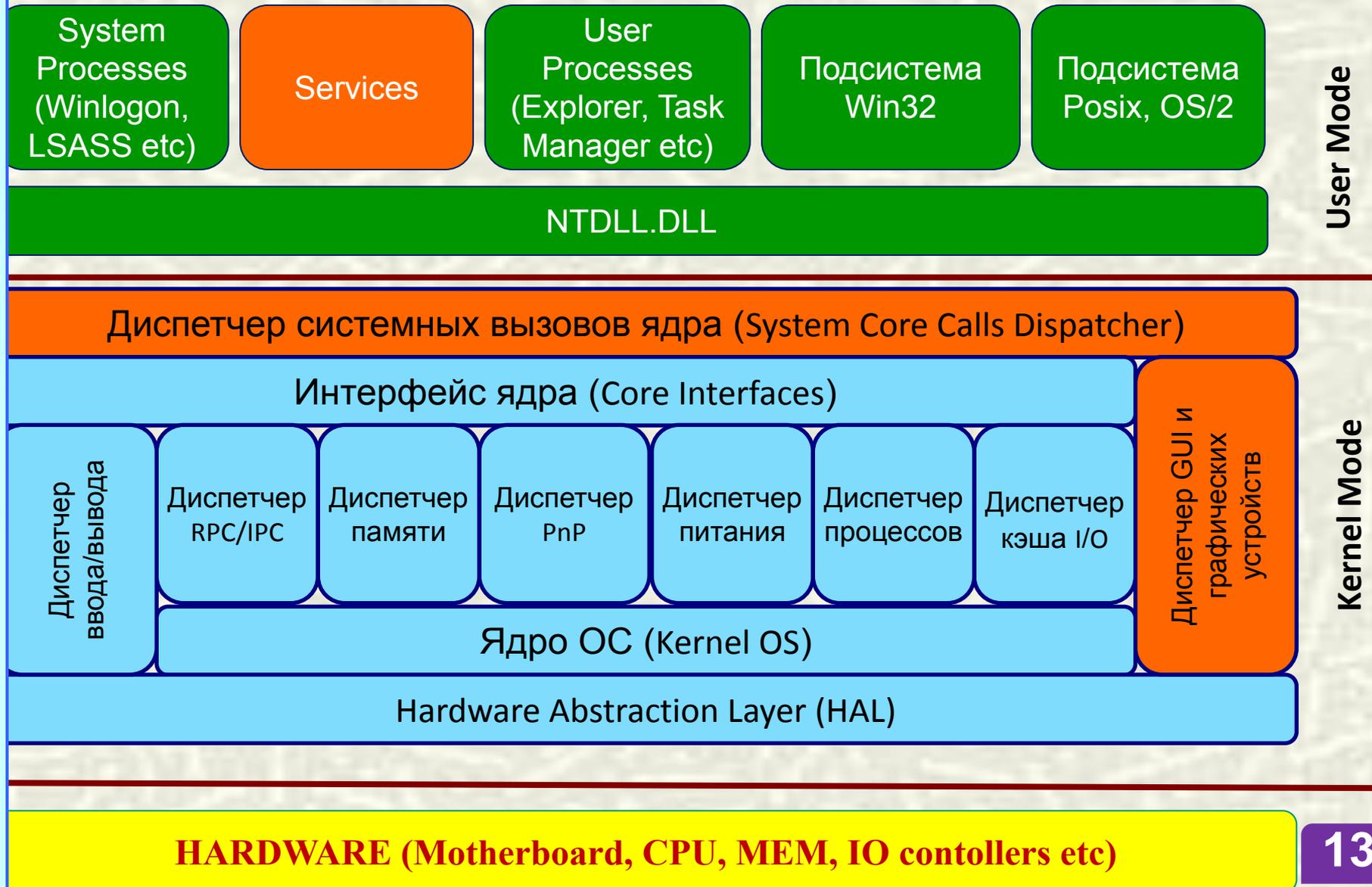
Архитектура Windows 2000

1. ОС WINDOWS



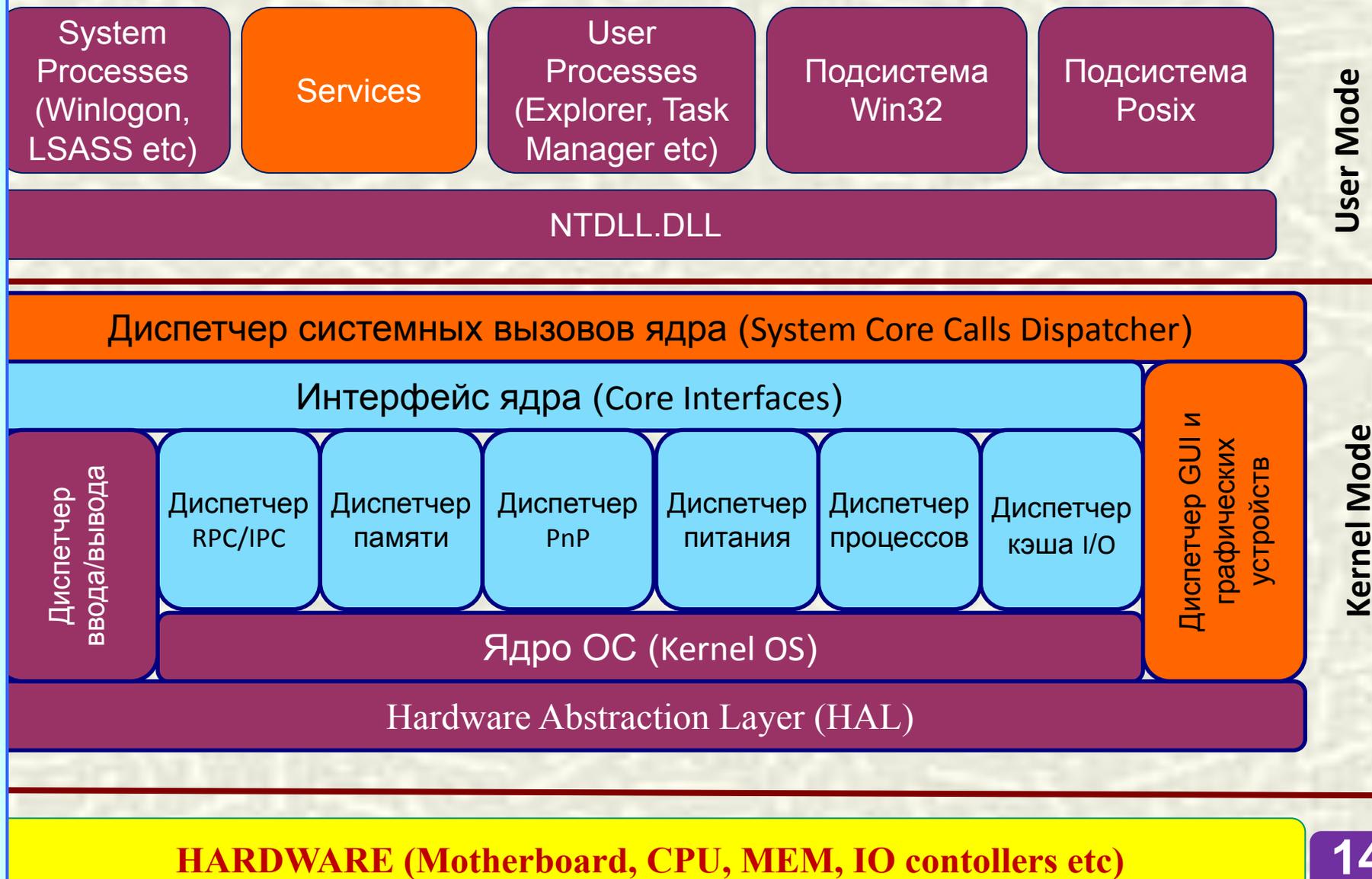
Архитектура Windows XP/ Windows Server 2003

1. ОС WINDOWS



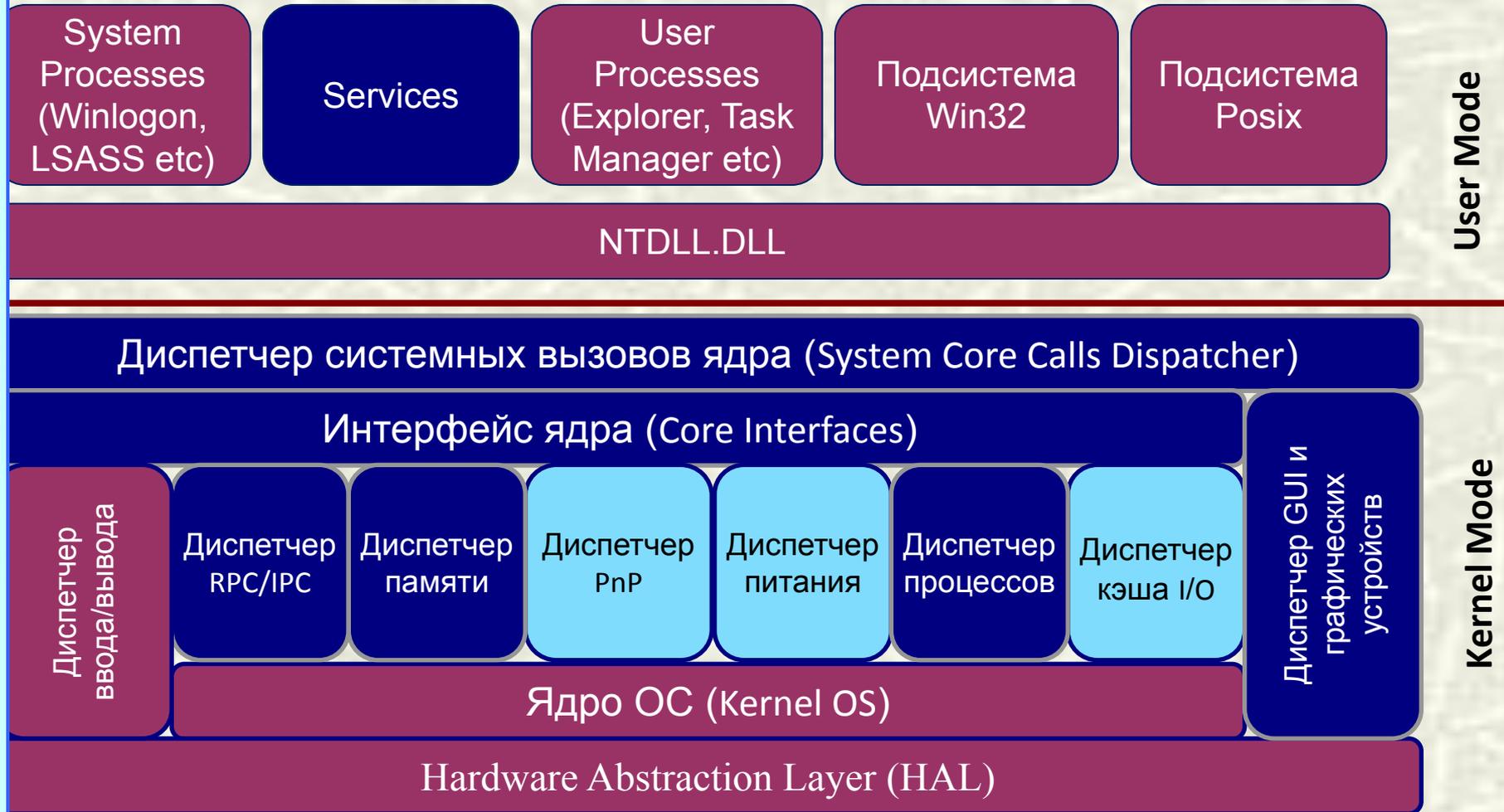
Архитектура Windows Vista / Windows Server 2008

1. ОС WINDOWS



Архитектура Windows 7 (10) / Windows Server 2008 R2

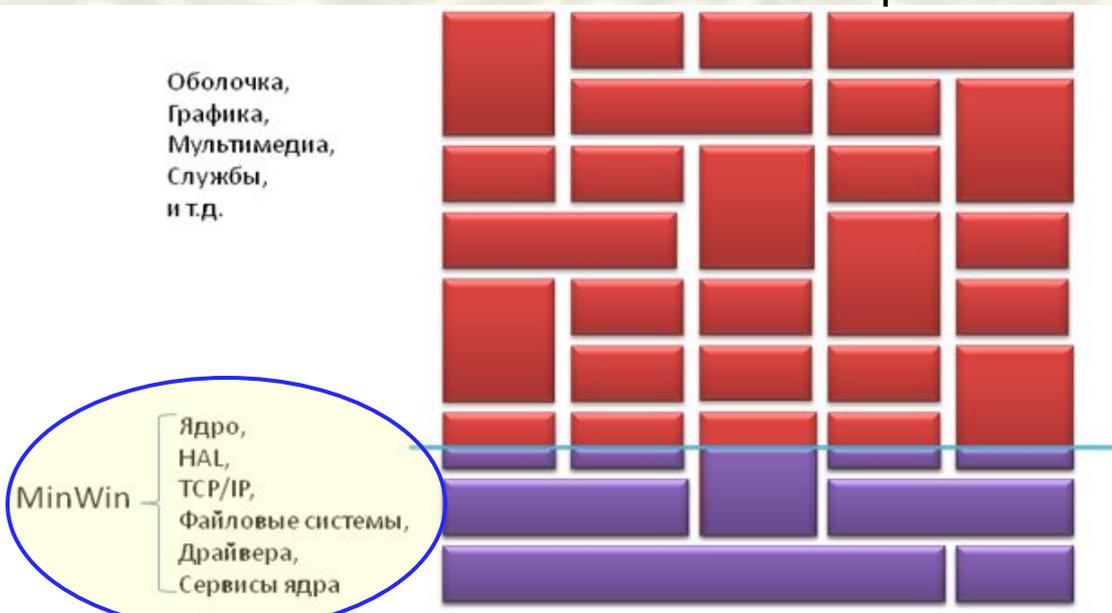
1. ОС WINDOWS



Архитектура Windows 7/ Windows Server 2008 R2

MinWin – минималистическая операционная система, состоящая лишь из ядра Windows и нескольких компонентов, таких как драйвера устройств, драйвера файловой системы, компоненты стека TCP/IP.

Wow64 – необязательная компонента режима Server Core



"NT Дэвида Катлера"

Размеры MinWin:

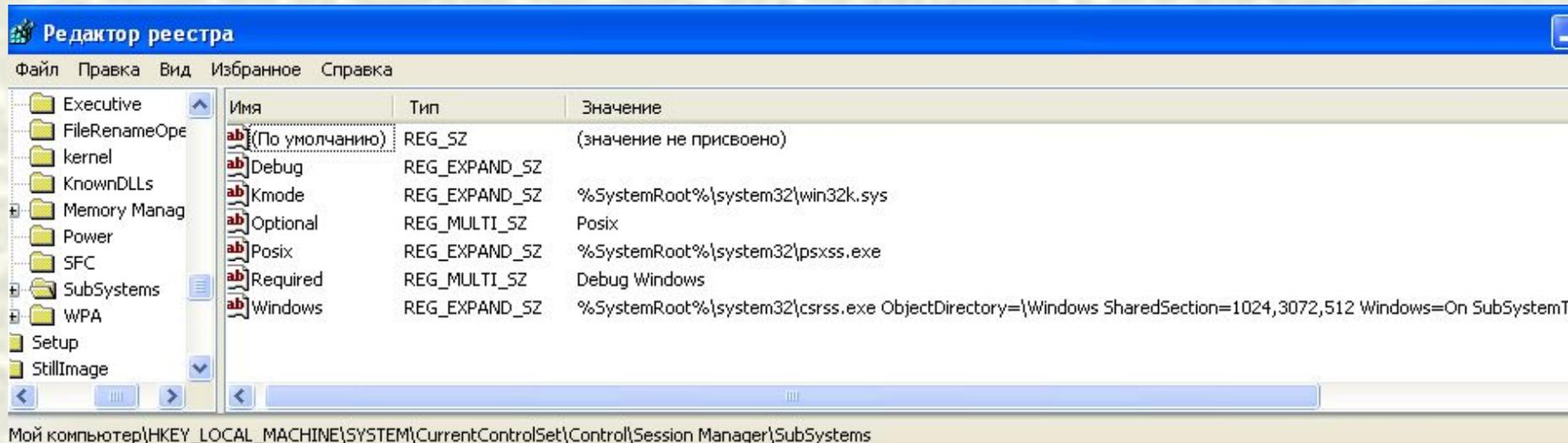
- ✓ 150 файлов,
- ✓ 25MB на HDD,
- ✓ 40MB в RAM

Рефакторинг DLL. Приложения за пределами MinWin используют обычные DLL. DLL перенаправляет запросы к MinWin API в MinWin DLL.

Пример: Kernel32.dll -> Kernelbase.dll
Advapi32.dll -> Kernelbase.dll

Стартовая информация защищенных подсистем хранится в разделе реестра:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\SubSystems.



Required – список подсистем, загружаемых при запуске системы.

Windows – указывается спецификация файла подсистемы Windows - csrss.exe

Debug остается незаполненным. Он используется для внутреннего тестирования и не выполняет никаких функций.

Optional указывает , что подсистемы OS/2 и POSIX запускаются по требованию

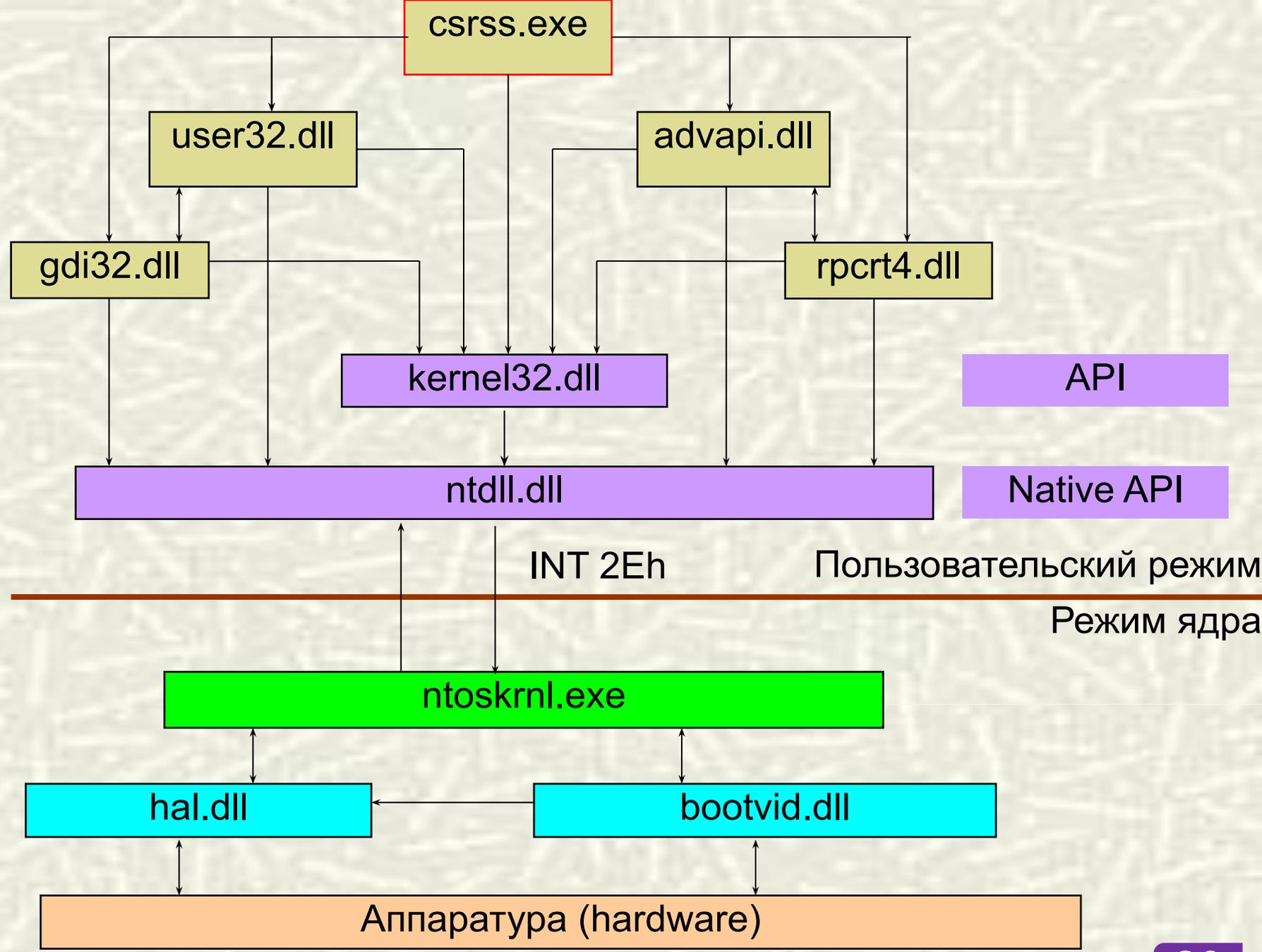
Posix указывается спецификация файла подсистемы POSIX, psxss.exe

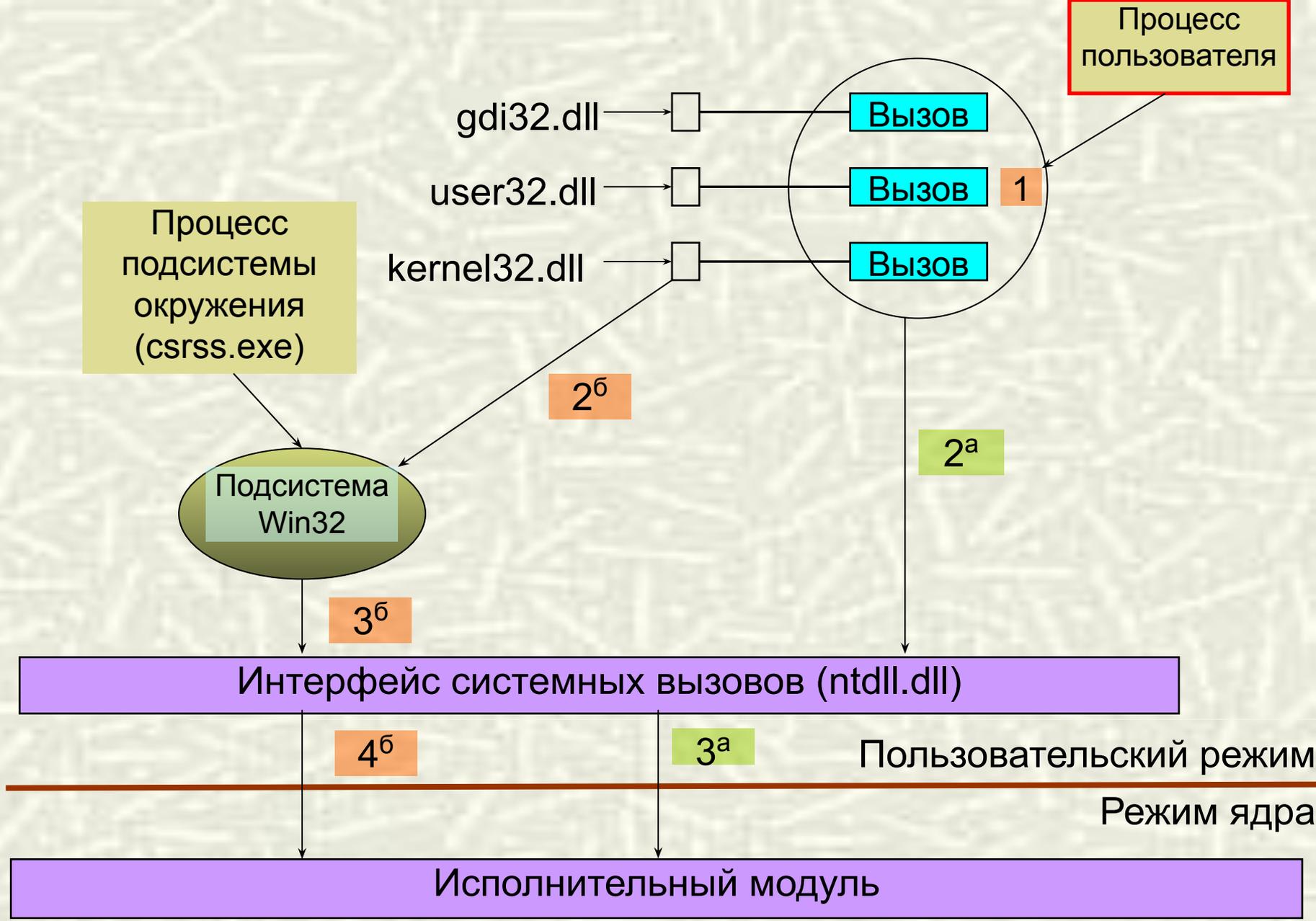
Kmode содержит имя файла той части подсистемы Windows, которая работает в режиме ядра, - win32k.sys

№ п/п	Подсистема исполнительной системы	Предназначение
1.	Диспетчер Объектов (Object Manager)	Управляет всеми известными операционной системе объектами (потоки, файлы, каталоги, семафоры, таймеры и т.д.), а также глобальным пространством имен.
2.	Диспетчер Ввода/Вывода (I/O Manager)	Служит интерфейсом между прикладными программами и драйверами устройств. Выступает каркасом для управления устройствами ввода/вывода и предоставляет общие службы ввода/вывода.
3.	Диспетчер Процессов (Process Structure)	Управляет процессами и потоками, включая их создание и завершение. Занимается не стратегиями, применяемыми по отношению к процессам, а механизмом для управления ими.
4.	Диспетчер Виртуальной Памяти (Virtual Memory Manager)	Определяет адресное пространство процесса и распределяет физическую память. Реализует архитектуру виртуальной памяти со страничной подкачкой по требованию ОС.
5.	Диспетчер Кэша (Cache Manager)	Реализует глобальный файловый кэш. Хранит блоки данных, которые использовались в последнее время, для ускорения доступа к ним.
6.	Диспетчер Безопасности (Security Reference Manager)	Реализует модель безопасности на основе Идентификаторов Безопасности (SID) и Списков Разграничительного Контроля Доступа (Discretionary Access Control List - DACL) . Реализует механизмы безопасности, удовлетворяющие требованиям класса C2 Оранжевой книги Министерства обороны США

Подсистемы Исполнительной Системы Windows и их предназначение

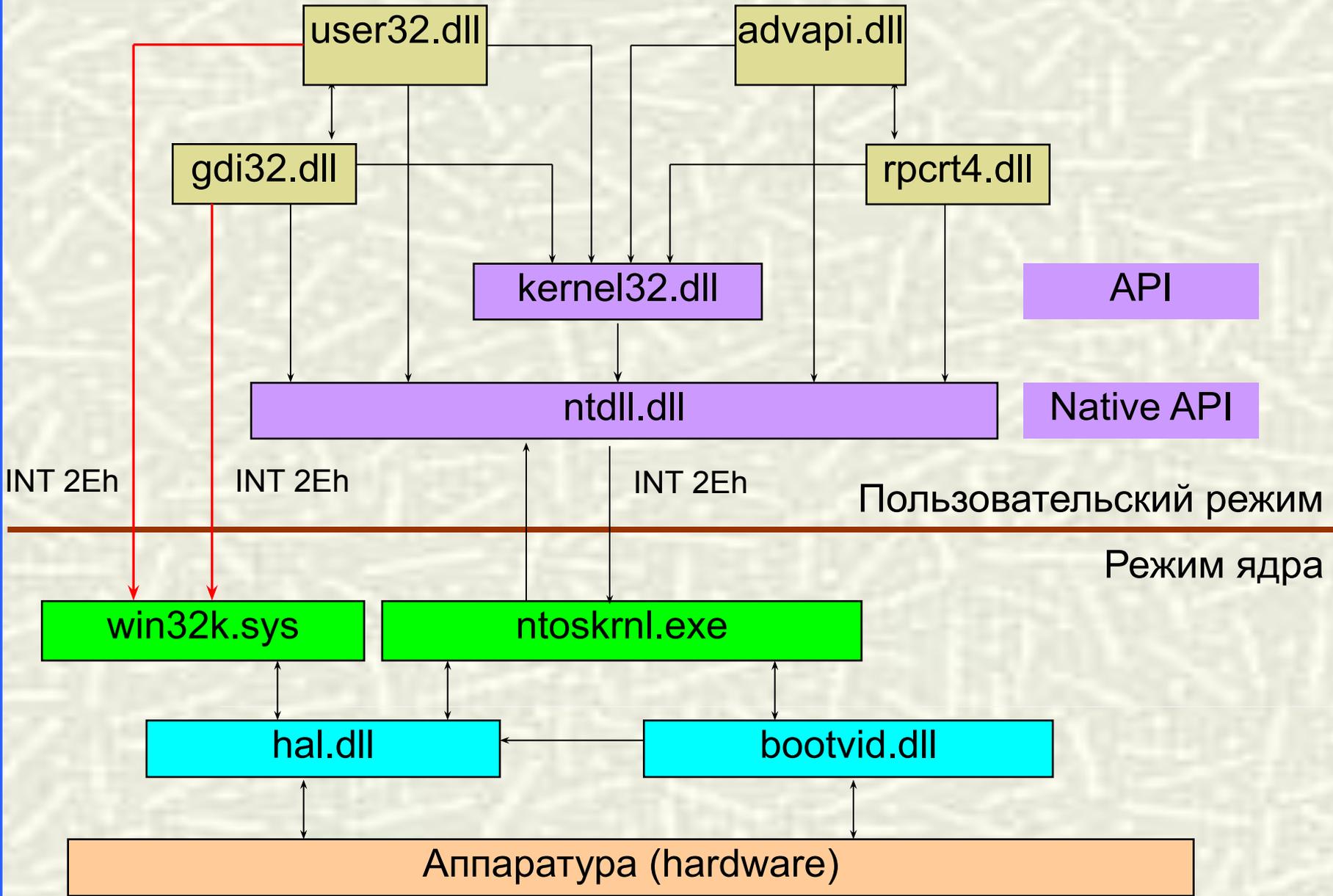
№ п/п	Подсистема исполнительной системы	Предназначение
7.	Диспетчер Plug-and-Play (Plug-and-Play Manager)	Уведомляет драйверы устройств о включении/отключении новых устройств. Для некоторых устройств проверка производится при загрузке системы, для других в произвольное время.
8.	Диспетчер Электропитания (Power Manager)	Контролирует состояние электропитания компьютера. Выключает монитор и диски, при отсутствии обращения к ним.
9.	Диспетчер Конфигурации (Configuration Manager)	Управляет Реестром. Добавляет новые записи и организует работу с ключами.
10.	Средство Вызова Локальных Процедур (Local Procedure Call - LPC) Facility)	Обеспечивает высокоэффективное взаимодействие между процессами и их подсистемами. Применяется при критичном выполнении системных вызовов, когда не используются стандартные механизмы межпроцессного взаимодействия (IPC)
11.	Поддержка среды Win32 (Win32 Support)	Реализует Win32-функции обмена сообщениям, окнами и рисования. Реализует интерфейс графических устройств (GDI).
12.	Исполнительный модуль (Executive Support)	Реализует управление очередями, системной областью памяти, обеспечивает системные рабочие потоки, а также предоставления интерфейса к исполняющей системе.





Различные маршруты выполнения вызовов Win32 API

1. ОС WINDOWS



Зависимости системных модулей, включая win32k.sys

Программная заглушка

Приложение:
Запрос на создание/открытие файла
CreateFile()

Kernel32.dll:
Вызов NtCreateFile()

ntdll.dll:
Вызов системного сервиса

```
NtCreateFile()  
moveax, 20h  
leaedx, [esp+4]  
int2Eh  
ret2Ch
```

INT 2Eh

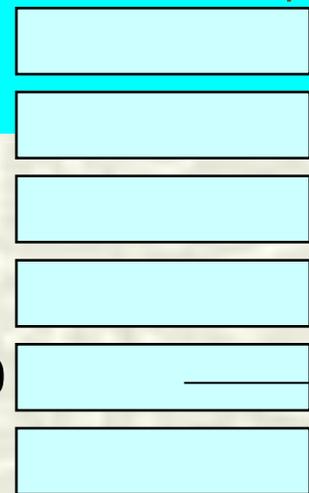
Пользовательский режим

Режим ядра

Обработчик ловушки

Диспетчер системных сервисов

Таблица дескрипторов системных сервисов



0x0020

Системный сервис создания/открытия файла



2. Архітектура ОС React OS



ReactOS — международный проект свободной и бесплатной операционной системы с открытым кодом. ReactOS не является точным клоном Windows, но операционной системой, **совместимой с приложениями и драйверами Microsoft Windows** (сейчас это Windows NT версии 5.x и выше — Windows 2000 и далее).

Разработчик	ReactOS Foundation
Семейство ОС	Windows NT
Последняя версия	0.4.10– 06 ноября 2018
Поддерживаемые платформы	x86, x86-64
Тип ядра	Гибридное ядро
Лицензия	GNU General Public License и др.
Состояние	Альфа-версия



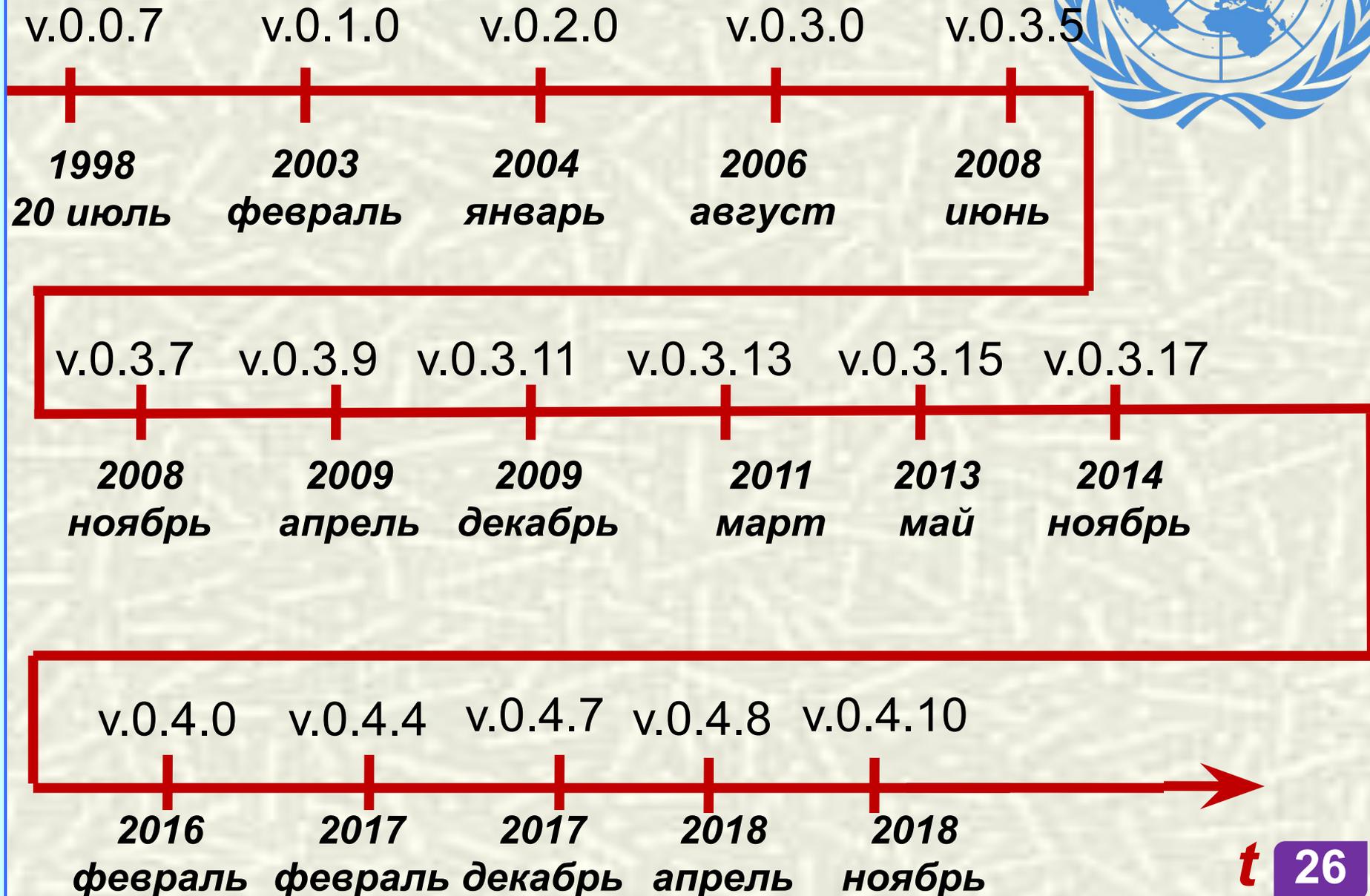
**Логотип операционной системы
ReactOS**

Для отстаивания интересов проекта был создан фонд “ReactOS Deutschland e.V.”, расположенный в Германии.

ИСТОРИЯ REACTOS



2. ОС REACTOS




My
Computer


My
Documents


My Network
Places


Recycle Bin


Applications
Manager


Command
Prompt


Read Me

ReactOS

ReactOS

ReactOS Version 0.4.9

Build 20180712-0.4.9-release-0-ga920a96.GNU_4.7.2

Reporting NT 5.2 (Build 3790: Service Pack 2)

C:\ReactOS

 Start

EN   5:45 PM

ФУНКЦИОНАЛЬНОСТЬ REACTOS

1. Большая часть ядра теперь **полностью совместима** с Windows 2003 Server SP1:
 - ✓ Исполнительная система (Executive)
 - ✓ Модули ядра (Планировщик, Диспетчер задач, Прерывания, и т.п.)
 - ✓ Уровень HAL
 - ✓ Локальный вызов процедур ядра,
 - ✓ Управление процессами и потоками
 - ✓ Поддержка системы ввода/вывода кроме PnP).
2. Другие части ядра полностью совместимы с NT-архитектурой: примечательный Диспетчер Cache, Менеджер Конфигурации (включая реестр) и Менеджер Памяти.
3. Поддержка приложения Win32 в основном, зависит от двух компонентов:
 - ✓ Win32k – режим ядра для GUI.
 - ✓ Win32 библиотеки (gdi32.dll, user32.dll, kernel32.dll, advapi32.dll) - взяты из соглашения Wine

Windows 95, 98 and ME



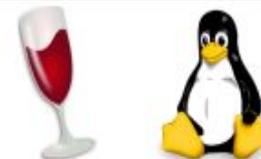
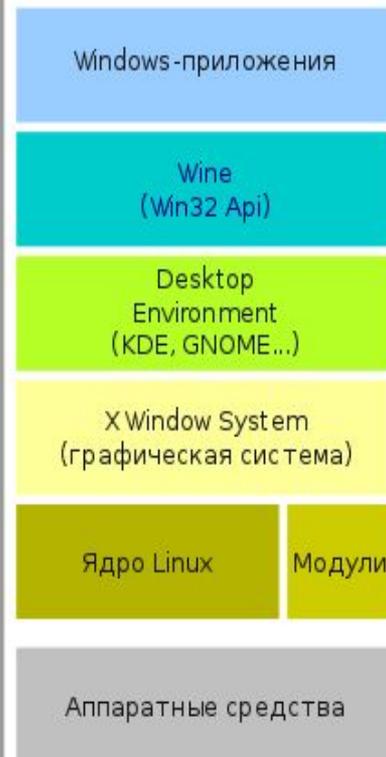
Windows NT, 2000, XP 2003, Vista, 7...



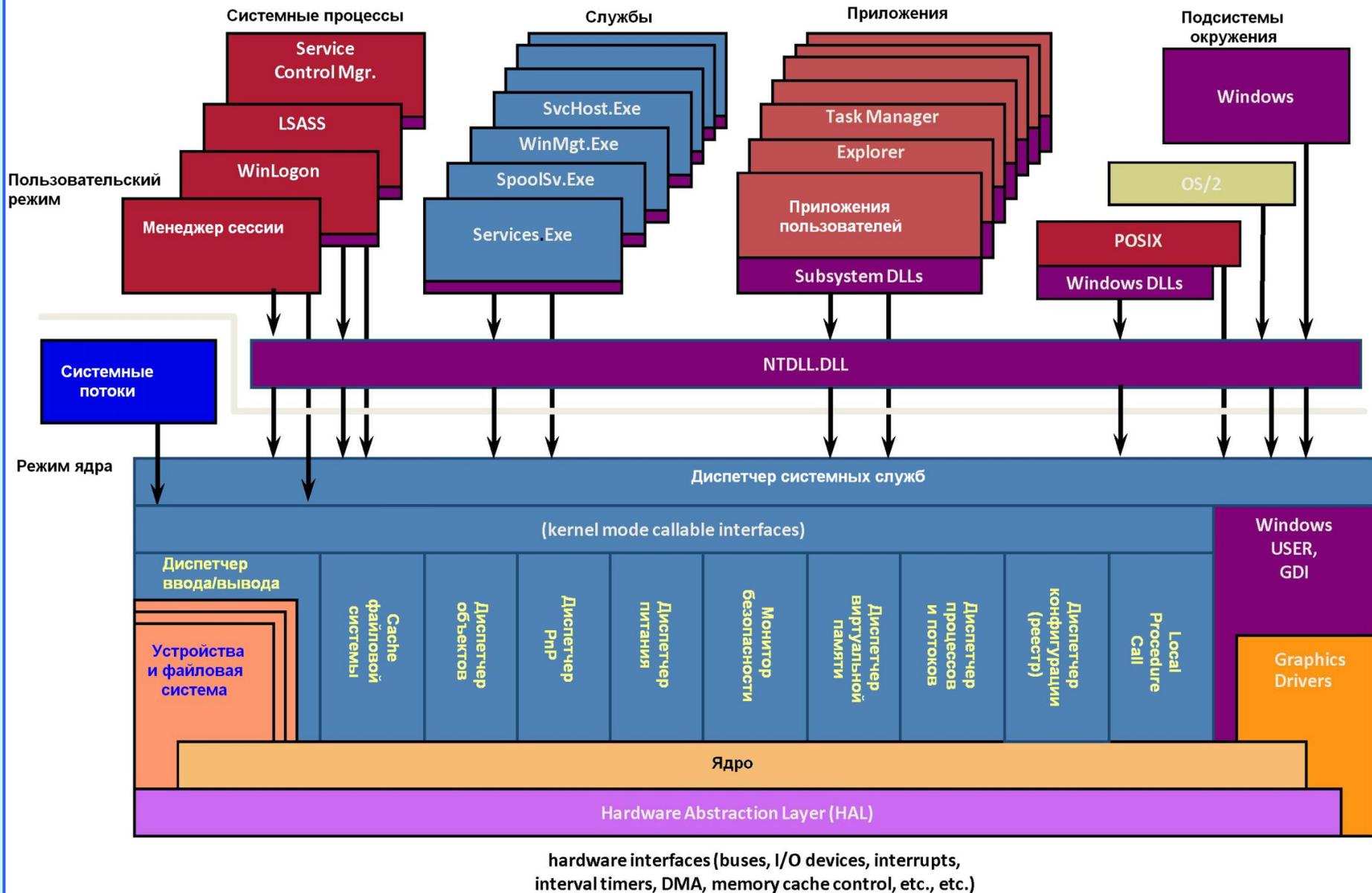
ReactOS



GNU / Linux with Wine



Архитектура ReactOS



Характеристика ядра ReactOS

Режим ядра ReactOS реализован:

- ✓ в одном большом модуле (ntoskrnl),
- ✓ в слое абстракции аппаратного обеспечения (HAL),
- ✓ наборе загружаемых модулей ядра (драйвера, и другие библиотеки).

Ядро написано в переносимом языке C++, компилируется с помощью MinGW и Microsoft Visual Studio

Таким образом для управления специфичными реализациями аппаратного обеспечения требуется HAL:

- ✓ прерывания,
- ✓ инициализация процессора,
- ✓ DMA, доступ к шинам PCI/ISA,
- ✓ таймеры, и т.д.

Некоторые драйвера специализированы для конкретного типа hardware (драйвер PCI, драйвер ATAPI, IDE, и т.д.).

Характеристика ядра ReactOS

В дереве исходников есть директория *Intoskrnl*, реализующая само ядро.

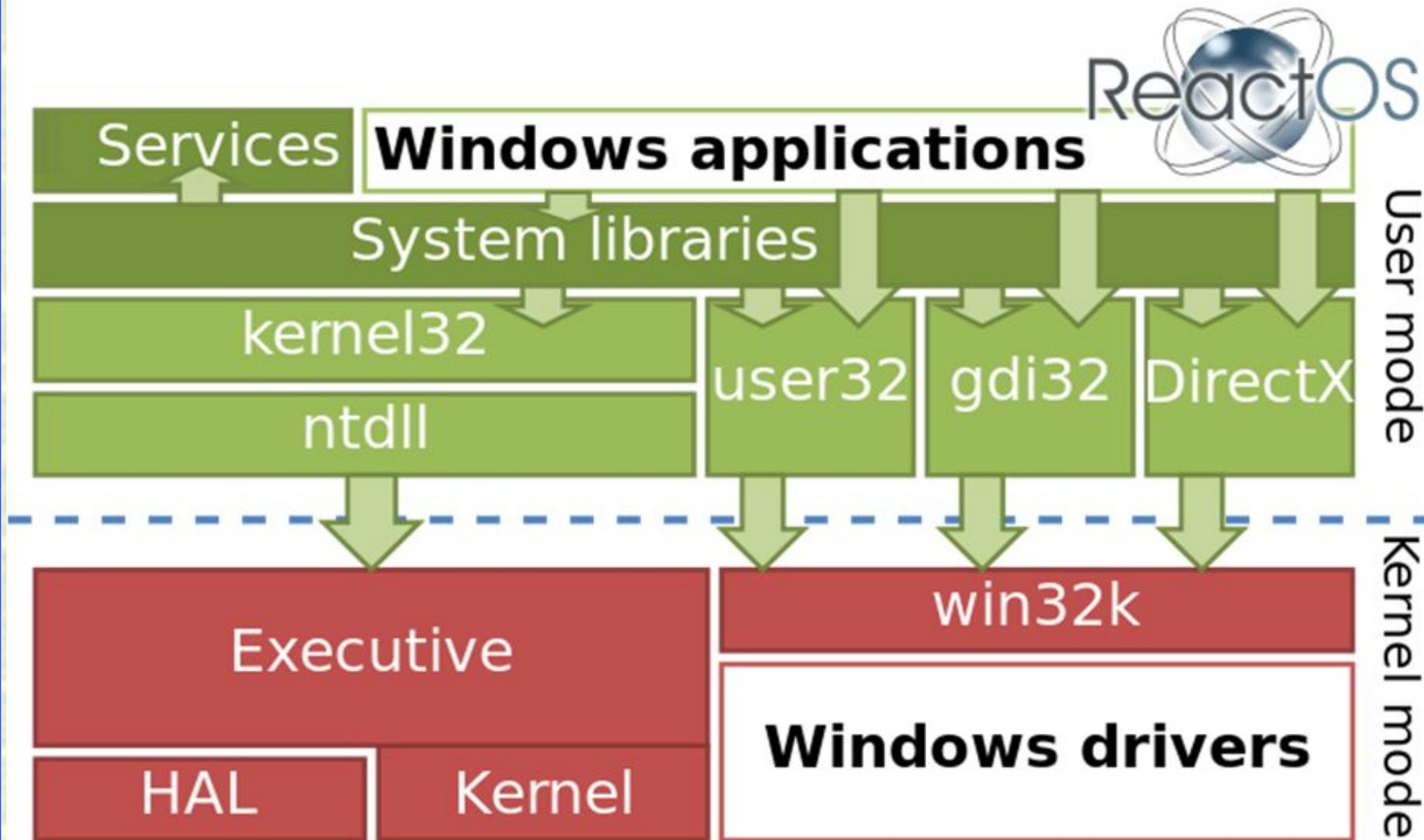
Также имеется:

- ✓ директория *hal* для общего кода HAL,
- ✓ *halx86*, *halxbox* для SMP специфичного кода (spinlocks и прерывания осуществлены в HAL).

Единственный порт системы ориентированный на другую архитектуру, над которым ведется работа – это PowerPC.

Менеджер PnP связывается с драйверами через *IRPs* (Пакеты Запроса ввода/вывода) и уведомления (события, операции) специфичны для ReactOS – на сегодня не реализованы в полном объеме.

NT загрузчик работает в страничном режиме с GDT, TSS. Загрузчик ReactOS работает в защищенном режиме и только ядро ответственно за системные структуры.



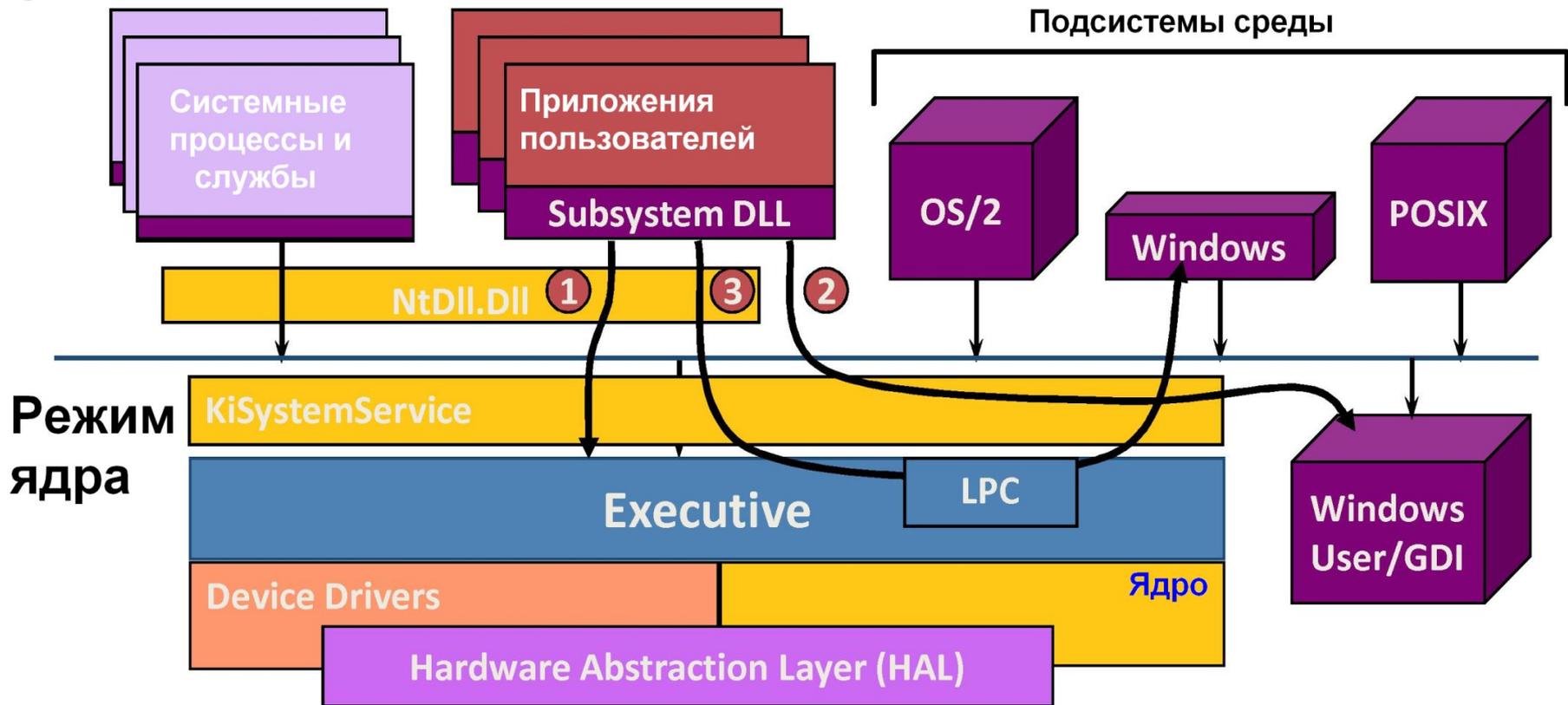
Архитектура ReactOS

Белым цветом обозначено стороннее программное обеспечение.

Зеленым – компоненты "пространства пользователя".

Красным – компоненты режима ядра.

Пользовательский режим



- ① большинство Windows Kernel APIs
- ② большинство Windows User and GDI APIs
- ③ немногие Windows APIs

Упрощенная архитектура ReactOS

Характеристика подсистем среды ReactOS

1. API DLLs

✓ для Windows: kernel32.dll, gdi32.dll, user32.dll, и т.п..

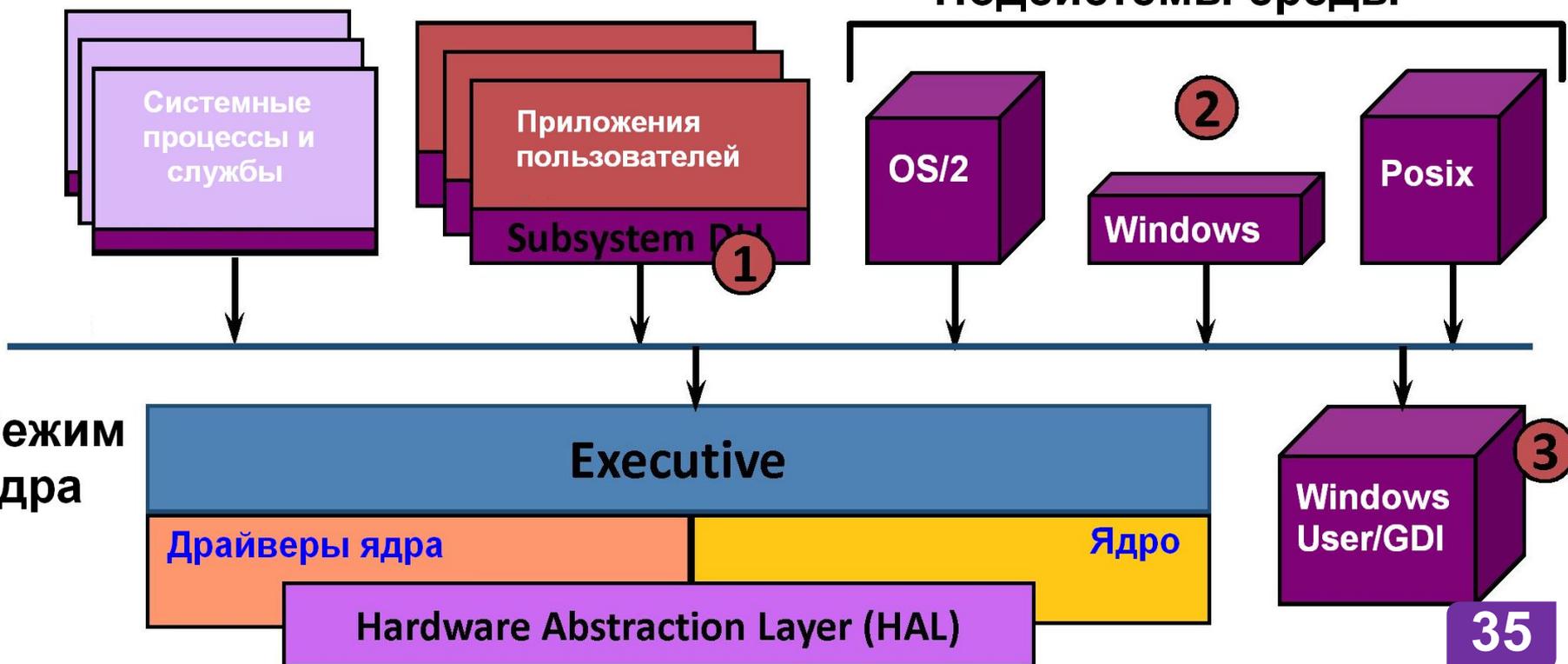
2. Процесс Подсистемы среды

✓ для Windows: csrss.exe (Client Server Runtime SubSystem)

3. Только для Windows: код режима ядра GDI

✓ win32k.sys - (этот код был прежде частью csrss.exe)

Пользовательский режим





**Виртуальное адресное пространство
пользовательского процесса в ReactOS**

Характеристика исполнительной системы ReactOS

Исполнительная подсистема (Executive) - верхний уровень ядра, представляющий сервис ядра подсистемам среды и другим серверам.

Диспетчер объектов обеспечивает поддержку объектно-базированной структуры ОС, представляющей ресурсы в виде объектов - абстрактных инкапсулированных типов данных.

Менеджер процессов отслеживает объекты процессов и потоков.

Менеджер виртуальной памяти выполняет формирование виртуального адресного пространства процесса и отображает виртуальные адреса в адресных пространствах процессов на физические страницы памяти.

Менеджер ввода-вывода обеспечивает независимый от устройств интерфейс ввода-вывода и отвечает за пересылку запросов на ввод-вывод соответствующим драйверам.

Менеджер безопасности проверяет права доступа к объектам по запросам других модулей Исполнительной системы и генерирует контрольные сообщения. Для получения информации о правах и передачи контрольных сообщений Менеджер безопасности взаимодействует с **Распорядителем локальной безопасности**.

Проблемы проектирования пользовательского режима в ReactOS

1. **Проблема обращения к ntdll.dll.** Нельзя использовать Wine для kernel32. ReactOS требует для своей реализации вызов ntdll.dll, какие затем выполняет эти системные вызовы, тогда как Wine предполагает осуществление функций API непосредственно в DLL.
2. **Проблема обращения к GDI и библиотеки USER32.** Технология NT требует непосредственного обращения к GDI через режим ядра (win32k.sys) . В ReactOS в подсистеме Wine делается это внутри GDI.
3. **Проблема поддержки h-заголовков.** Другие DLL непосредственно выполняются из подсистемы Wine с минимальными изменениями, чтобы поддерживать наши заголовки и формировать систему.
4. **Проблема совместимости с Windows NT.** DLL предполагают работать на уровне совместимости двоичных кодов с Windows.
5. **Проблема поддержки файловых систем.** В ReactOS доступны файловые системы **FAT32, ext2, ISO-9660 (CDFS), NTFS (read only), BtrFs, ReiserFS** и **UFS**.



Спасибо за внимание!