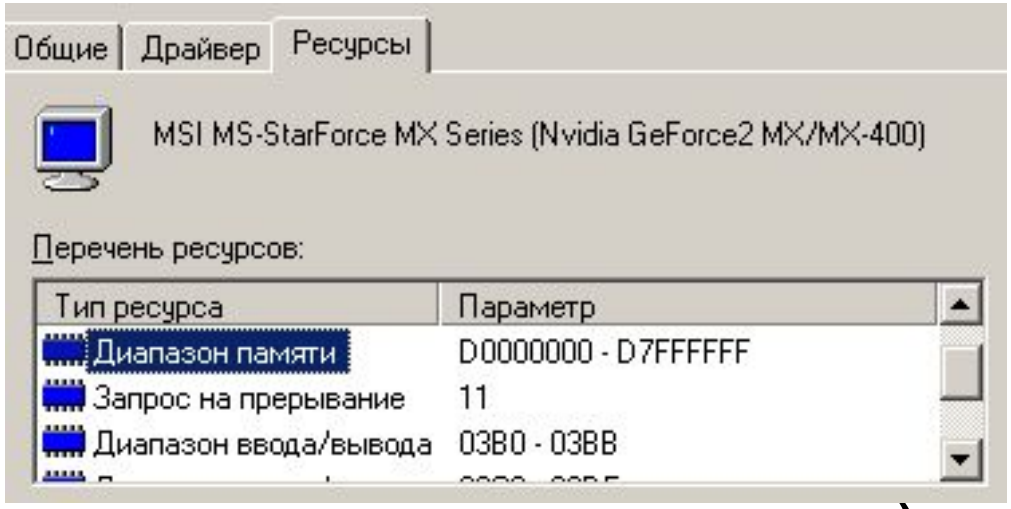
Операционные системы

Лекция 5

Общая организация Windows

Технология Plug & Play

- Диапазоны памяти
- Запрос на прерывание IRQ
- Диапазон портов в/в



- P&P BIOS
- ACPI (Advanced Configuration and Power Interface)

Загрузка и функционирование DOS

- BIOS, драйверы основных устройств) (Int 10h-1Fh)
  - 10h – видеосервис
  - 13h – дисковая подсистема
  - 14h – коммуникационные порты
  - 16h – клавиатура
  - 17h – порт принтера
  - 19h – перезагрузка системы
  - 1Ah – работа с часами
- IO.SYS – расширение базовой системы ввода-вывода. Настраивает устройства ввода-вывода
- MSDOS.SYS (config.sys) – ядро операционной системы (20h-2Fh)
  - 21h – основные функции
  - 23h – обработчик Ctrl-C
  - 25h – абсолютное чтение диска
  - 28h – мультизадачное прерывание
- Драйверы устройств (device=)
- COMMAND.COM (autoexec.bat) – командный процессор, обрабатывает внутренние и загружает внешние команды MS-DOS

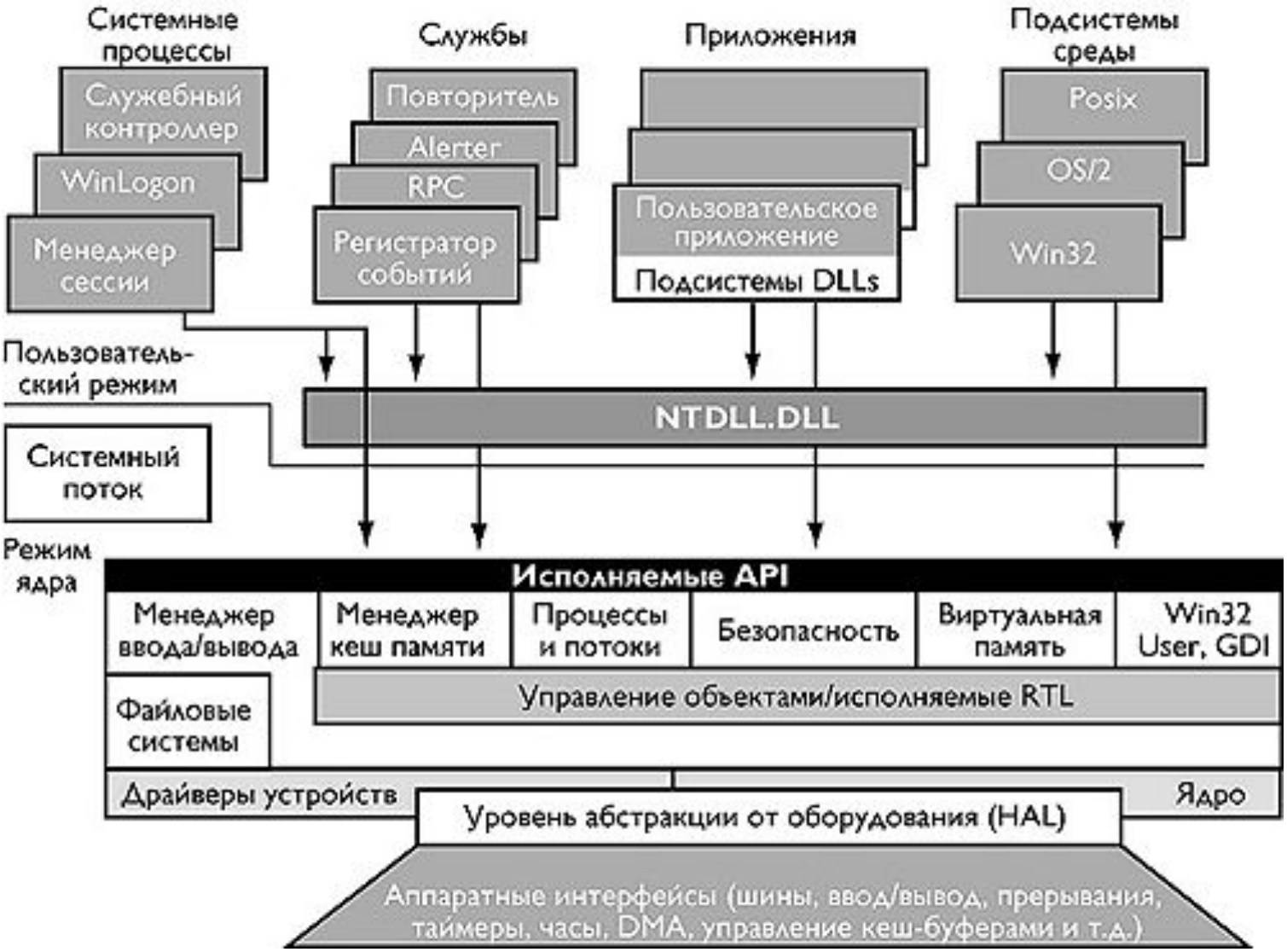
Загрузка Windows’98

- Загрузка DOS
- WIN.COM – переход в защищенный режим и загрузка графической подсистемы
- WININIT.EXE – загрузка Windows + основные DLL-библиотеки Windows
  - KRNL386.EXE – распределение памяти, управление задачами
  - USER.EXE – поддержка пользовательского интерфейса
  - GDI.EXE – интерфейс графических устройств
    - Дополнительные библиотеки: COMMDLG.DLL, QTIME.DLL
  - Драйверы: DRV, VxD (386)
- Оболочка: EXPLORER.EXE или PROGMAN.EXE

Ядро Windows 98

32	16
User 32 (USER32.DLL) Переадресация 32 разрядных вызовов 16-разрядному модулю	**User 16 (USER16.EXE)** Управление окнами и меню Windows 3.1 + новые методы (поддержка модели асинхронного ввода и т. д.)
GDI 32 (GDI32.DLL) Отображение шрифтов TrueType, подсистема печати, новая графическая подсистема	**GDI 16 (GDI16.EXE)** Графические методы Windows 3.1 + новые графические методы
Kernel 32 (KERNEL32.DLL) Управление потоками, синхронизация объектов, управление памятью, файловый ввод/вывод и т.д.	**Kernel 16 (KRNL386.EXE)** Инициализация Kernel 32

Структура Windows 2000/XP



Задачи, выполняемые ядром Windows 2000

- **Исполняемая часть NT** которая включает управление памятью, процессами, потоками, безопасностью, вводом/выводом, межпроцессорными обменами;
- **Ядро Windows NT** выполняет низкоуровневые функции операционной системы: диспетчеризация потоков, прерываний и исключений, синхронизация процессоров. Ядро также включает набор процедур и базовых объектов, используемый исполняемой частью для создания высокоуровневых конструкций;
- **Слой абстракции от оборудования (HAL - Hardware Abstraction Layer),** изолирует ядро, драйверы устройств и исполняемую часть NT от аппаратных платформ, на которых должна работать операционная система;
- **Драйверы устройств** включают как файловую систему, так и аппаратные драйверы, которые транслируют пользовательские вызовы функций ввода/вывода в запросы физических устройств ввода/вывода;
- **Функции графического интерфейса пользователя** работают с окнами, элементами управления и рисунками.

Исполняемая часть Windows 2000

- **Менеджер процессов и потоков управляет процессами и потоками.** Фактически потоки и процессы поддерживаются в NT нижележащим слоем. Исполняемая часть добавляет дополнительную семантику и функции к этим объектам нижнего уровня.
- **Менеджер виртуальной памяти** использует схему управления, при которой каждый процесс получает собственное достаточно большое адресное пространство, защищенное от воздействия других процессов. Менеджер памяти также обеспечивает низкоуровневую поддержку для менеджера кэш-памяти.
- **Монитор безопасности** проводит политику обеспечения мер безопасности на локальном компьютере, охраняя системные ресурсы и выполняя процедуры аудита и защиты объектов.
- **Система ввода/вывода** использует независимый от устройств ввод/вывод и отвечает за пересылку данных соответствующим драйверам для дальнейшей обработки.
- **Менеджер кэш-памяти** улучшает производительность системы ввода/вывода файлов, размещая читаемые с диска данные в основной памяти для ускорения доступа к ним, а также откладывая на короткое время запись измененных данных на диск.

Исполняемая часть Windows 2000

- Менеджер объектов, который создает, удаляет объекты и абстрактные типы данных, а также управляет ими. Объекты используются в Windows NT для представления таких ресурсов операционной системы, как процессы, потоки и объекты синхронизации.
- LPC передает сообщения между клиентским процессом и процессом сервера на том же самом компьютере. По сути, LPC - это оптимизированная версия известной процедуры удаленного вызова RPC (Remote Procedure Call), стандарта для организации взаимодействия процессов в архитектуре клиент/сервер.
- Широкий набор библиотечных функций общего типа: обработка строк, арифметические операции, преобразование типов данных, обработка структур.
- Процедуры распределения памяти, взаимообмен между процессами через память, два специальных типа объектов синхронизации - ресурсы и объекты fast mutex.

Ядро Windows 2000

Ядро NTOSKRNL.EXE выполняет большинство основных операций NT, определяющих порядок использования процессора: диспетчеризация потоков; диспетчеризация и обработка исключений; синхронизация работы процессоров; обеспечение базовых объектов ядра, которые используются исполняемой частью (и в некоторых случаях экспортируются в режим пользователя).

В отличие от остальной исполняемой части операционной системы, ядро никогда не выгружается из оперативной памяти, его выполнение никогда не прерывается другими потоками. Код ядра написан в основном на Си, а части, дающие наибольшую нагрузку на процессор, на языке Ассемблере.

Объекты ядра. Одна из функций ядра – обеспечение низкоуровневой базы для хорошо определенных примитивов операционной системы, которые обеспечивают работу компонентов высшего уровня. Ядро изолирует само себя от остальной части ОС, что позволяет вынести принятие политических решений из ядра, за исключением диспетчеризации потоков. Ядро использует набор простейших объектов, называемых объектами ядра, позволяющих управлять работой центрального процессора и порядком создания вычисляемых объектов. Большинство вычисляемых объектов включает в себя один или более объектов ядра, включая определенные ядром атрибуты. Один из наборов объектов называется объектами управления и включает объект процесса ядра, объект APC, объект процедуры отложенного вызова DPC (Deferred Procedure Call) и несколько объектов, используемых системой ввода/вывода (например, объект обработки прерывания).

Другой набор объектов ядра - объекты диспетчеризации, включает объекты синхронизации потоков, поток ядра, mutex, объекты события, семафора, таймера, таймера ожидания и ряд других.

Поддержка оборудования. Другой главнейшей задачей ядра является абстрагирование (или изоляция) исполняемой части и драйверов устройств от различий микропроцессорных платформ, на которых способна работать Windows NT: x86 и Alpha AXP. Специфичные для архитектуры функции (такие, как контекстное переключение потока) реализованы в ядре. Функции, которые могут отличаться от машины к машине, реализованы в составе HAL.

Ядро поддерживает набор интерфейсов, семантически идентичных для всех архитектур. Некоторые из интерфейсов реализованы по-разному для разных архитектур, однако, и идентичны внешне интерфейсы реализованы с помощью специфичного для архитектуры кода. Независимый от архитектуры интерфейс может быть вызван на любой машине, и его семантика будет той же, несмотря на то, зависит ли код от архитектуры или нет. Некоторые интерфейсы ядра (например, процедуры синхронизации SMP) реализованы в HAL, поскольку их реализация может изменяться даже внутри одного семейства компьютеров. В качестве примера зависящего от архитектуры кода можно назвать также поддержку кэша центрального процессора.

Абстракция от оборудования

Загружаемый модуль ядра HAL обеспечивает низкоуровневый интерфейс с аппаратной платформой, что позволяет скрыть такие зависимые от аппаратуры детали, как интерфейс ввода/вывода, контроллеры прерываний, механизм обмена данными между процессорами - любые аппаратно-зависимые и специфические для архитектуры функции.

Драйверы устройств – это загружаемые модули, которые работают в режиме ядра, обеспечивая интерфейс между системой ввода/ вывода и соответствующим оборудованием. Названия этих модулей обычно имеют расширение .SYS. Все они, как правило, написаны на Си (иногда C++) с использованием вызовов процедур HAL и могут быть переносимыми на уровне двоичного кода между платформами, поддерживаемыми NT. Имеется несколько типов драйверов устройств:

- **Драйверы, манипулирующие устройствами** (с использованием HAL) для записи выходных данных или получения входных данных от физических устройств или через сеть.
- **Драйверы файловой системы,** которые принимают запросы на файловый ввод/вывод и транслируют их в запросы ввода/вывода, связанные с конкретными устройствами.
- **Драйверы фильтров.** Примером могут быть драйверы поддержки зеркальных дисков, шифрования данных, перехвата ввода/вывода для дополнительной обработки данных перед передачей их на следующий уровень и т.д.
- **Сетевые драйверы,** которые передают и принимают удаленные запросы на ввод/вывод.
- Поскольку установка драйверов устройств является единственным способом добавить к системе пользовательский код, работающий в режиме ядра, то некоторые программисты могут рассматривать написание драйверов устройств как способ доступа к внутренним функциям и структурам данных операционной системы, недоступным из пользовательского режима.

Пользовательские процессы

Имеется четыре базовых типа пользовательских процессов.

- **Специальные процессы поддержки системы,** например, процесс регистрации пользователя и менеджер сессий, которые не являются службами NT.
- **Процессы сервера**, которые являются службами NT (аналог демонов в ОС Unix). Примером может быть регистратор событий (Event Logger). Многие дополнительно устанавливаемые приложения, такие как Microsoft SQL Server и Exchange Server, также включают компоненты, работающие как службы NT.
- **Подсистемы среды,** которые обеспечивают пользовательским приложениям среду других операционных систем. Windows NT поставляется с тремя подсистемами: Win32, Posix и OS/2 2.1.
- **Пользовательские приложения** одного из пяти типов: Win32, Windows 3.1, MS-DOS, Posix или OS/2 1.2.

Подсистемы среды и библиотеки DLL

- Windows NT имеет три подсистемы среды (Win32, Posix и OS/2 2.1), которые работают только на платформе x86. Подсистема Win32 специфична для Windows NT и не может работать вне нее.
- Каждая из подсистем обеспечивает пользовательским приложениям доступ к разным поднаборам служб Windows NT. Это означает, что некоторые вещи могут быть сделаны из приложения, построенного на одной подсистеме, и не возможны из приложения, построенного в другой подсистеме. Так, приложение для Win32 не может использовать функцию fork подсистемы Posix.
- Каждый исполняемый модуль связывается с одной и только одной подсистемой. Когда начинается выполнение модуля, изучается тип кода его заголовка, что позволяет определить подсистему среды для создания новых процессов.
- Пользовательские процессы не вызывают службы NT напрямую, а используют библиотеки динамических связей (DLL) соответствующей подсистемы среды. Роль библиотек, принадлежащих подсистеме среды, в том, чтобы транслировать документированные функции среды в соответствующие вызовы недокументированных служб NT. Эти библиотеки DLL экспортируют документированный интерфейс, который могут вызывать связанные с подсистемой программы. Например, библиотеки DLL подсистемы Win32 используют функции Win32 API. Библиотека DLL подсистемы Posix использует функции Posix 1003.1 API.

Подсистема Win32

Главные компоненты подсистемы Win32 - процесс подсистемы среды и драйвер режима ядра.

Процесс подсистемы среды поддерживает:

- консольные (текстовые) окна;
- создание и удаление процессов и потоков;
- работу виртуальной 16-разрядной DOS машины;
- иные функции (GetTempFile, DefineDosDevice, ExitWindowsEx и др.).

Драйвер режима ядра поддерживает:

- менеджер окон, который управляет отображением окон, выводом на экран, вводом с клавиатуры, от мыши и других устройств, а также передачей пользовательских сообщений приложениям;
- интерфейс графических устройств GDI (Graphical Device Interface), библиотека функций для вывода на графические устройства, для рисования текста, линий, фигур и манипуляций графическими объектами;
- зависимые от устройств драйверы графики, принтера и видеопорта;
- несколько библиотек DLL, которые транслируют документированные функции Win32 API в соответствующие недокументированные вызовы NTOSKRNL.EXE и WIN32K.SYS.

Приложения вызывают стандартные функции для создания окон и кнопок на дисплее. Менеджер окон передает эти запросы драйверам графических устройств через интерфейс графических устройств GDI, где они форматируются для вывода средствами конкретных устройств. GDI обеспечивает набор стандартных функций, позволяющих приложениям общаться с графическими устройствами, включая дисплеи и принтеры, без конкретных знаний о них. GDI интерпретирует запросы приложений на графический вывод и посылает их драйверам графических дисплеев. Этот интерфейс позволяет создавать код приложения, независимый от конкретных устройств и их драйверов

NTDLL.DLL

NTDLL.DLL – это специальная система поддержки DLL - библиотек. Она содержит два типа функций.

- **Первая группа функций обеспечивает интерфейс к службам NT,** которые могут быть вызваны из пользовательского режима. Существует более 200 таких функций, например NtCreateFile, NtSetEvent и т.д. Для каждой из них имеется точка входа в NTDLL.DLL с тем же именем. Внутренний код функции содержит специфичные для архитектуры команды, которые вызывают переход в режим ядра для обращения к реальным службам NT, код которых содержится в NTOSKRNL.EXE.
- **Вторая группа функций содержит большое количество функций поддержки:** загрузчик исполняемых модулей, коммуникационные функции для процессов подсистемы Win32, библиотека функций реального времени пользовательского режима, диспетчер вызовов асинхронных процедур APC (Asynchronous Procedure Call) пользовательского режима, диспетчер исключений.

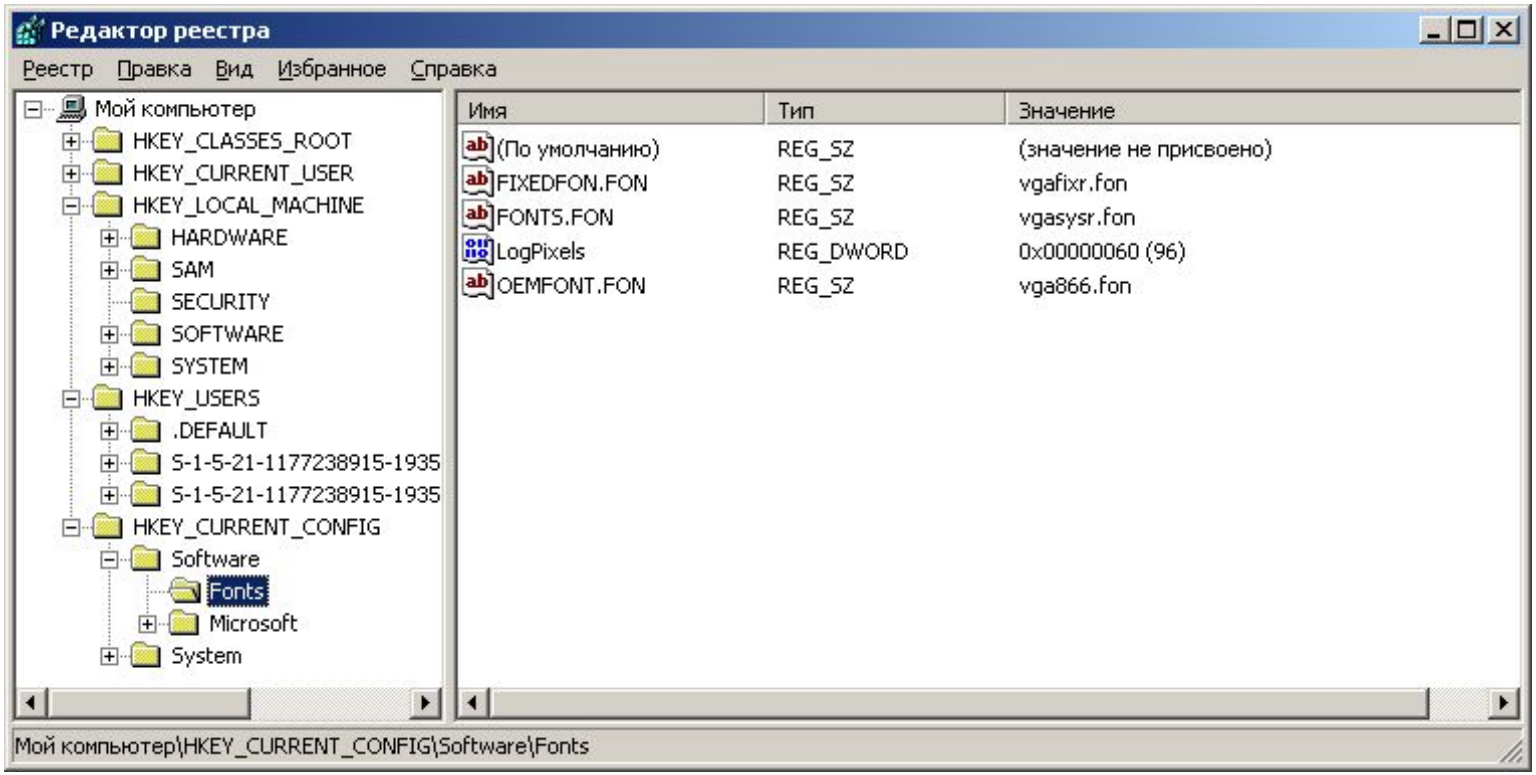
Загрузка системы

- **POST**
- **NTLDR.EXE**
  - Переключает процессор в линейный режим
  - Запускает мини-файловую систему
  - Читает boot.ini и выбирает ОС
  - Вызывает ntdetect.com для определения оборудования и помещает в HKEY_LOCAL_MACHINE\HARDWARE
  - Запускает ядро Ntoskrnl.exe
  - Управляющие параметры в HKLM\SYSTEM
  - Запускаются сервисы из HKLM\SYSTEM\CurrentControlSet\Services
  - Инициализация ядра из HKEY_LOCAL_MACHINE\HARDWARE
  - Драйверы в HKLM\SYSTEM\CurrentControlSet\Services*DriverName*

Основные модули

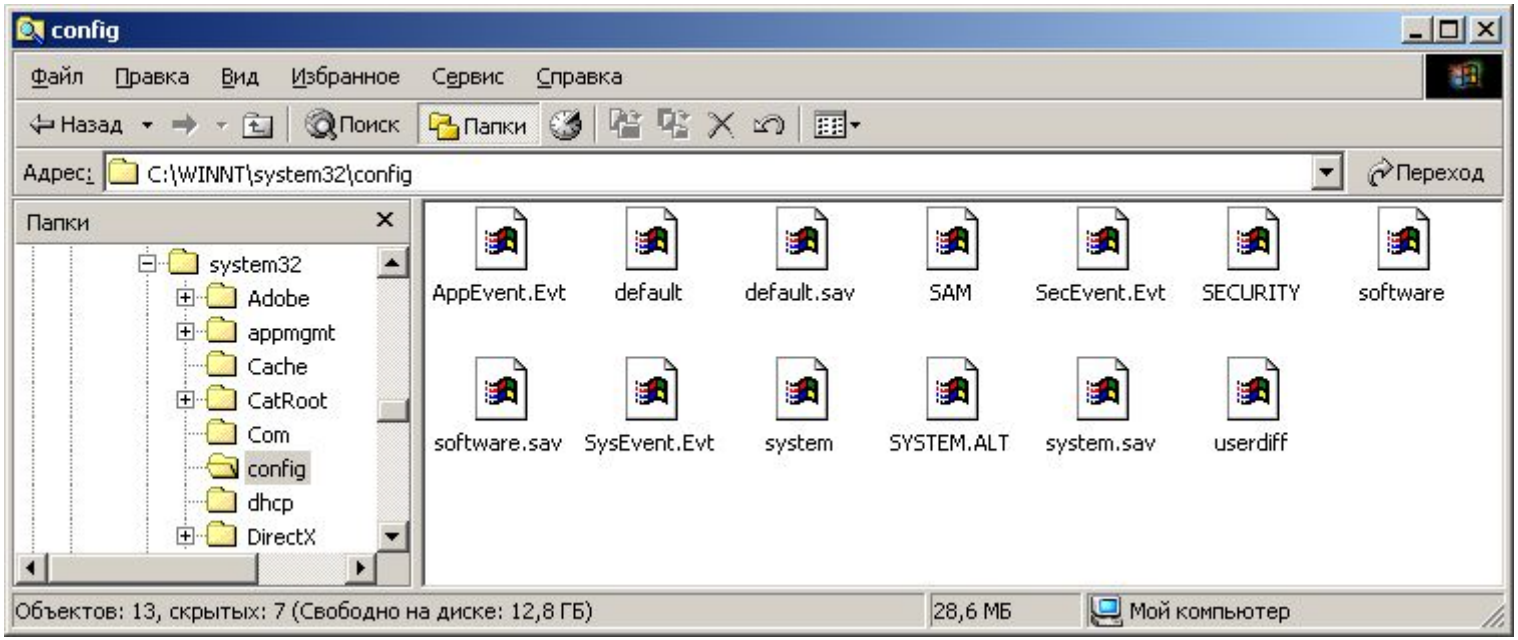
- **Smss.exe** – диспетчер сервисов
- **Winlogon.exe** – регистрация пользователей в системе
- **Svchost.exe**
  - Диспетчер авто-подключений удаленного доступа. Создает подключение к удаленной сети, когда программа обращается к удаленному DNS- или NetBIOS-имени или адресу.
  - Сетевые подключения.
  - Система событий COM+. Автоматическое распространение событий подписавшимся компонентам COM.
  - Съемные ЗУ
  - Телефония
  - Уведомление о системных событиях
  - Удаленный вызов процедур (RPC)
- **Spoolsv.exe** – Диспетчер очереди печати
- **Regsvc.exe** – Служба удаленного управления реестром
- **Mstask.exe** – Планировщик заданий
- **Taskmgr.exe** – Переключатель задач
- **Lsass.exe**
  - Агент политики IPSEC
  - Диспетчер учетных записей безопасности
- **Services.exe**
  - DHCP
  - DNS
  - P&P
  - Диспетчер логических дисков
  - Журнал событий
  - Клиент отслеживания изменившихся связей
  - Обозреватель компьютеров
  - Рабочая станция
  - Служба RunAs. Позволяет запускать процессы от имени другого пользователя.
  - Служба поддержки TCP/IP NetBIOS
- **Ntvdm.exe** – диспетчер виртуальных машин
- **Devldr32.exe** – загрузка драйверов устройств
- **Winmgmt.exe** – инструментарий управления Windows
- **cmd.exe**
- **inetinfo.exe**
  - Служба IIS Admin
  - Web-сервер
  - FTP-сервер
  - SMTP-сервер
- **internat.exe** – раскладка клавиатуры

Реестр


Редактор реестра
Реестр Правка Вид Избранное Справка
Мой компьютер
HKEY_CLASSES_ROOT
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
HARDWARE
SAM
SECURITY
SOFTWARE
SYSTEM
HKEY_USERS
.DEFAULT
S-1-5-21-1177238915-1935
S-1-5-21-1177238915-1935
HKEY_CURRENT_CONFIG
Software
Fonts
Microsoft
System
Имя
Тип
Значение
(По умолчанию)
REG_SZ
(значение не присвоено)
FIXEDFON.FON
REG_SZ
vgafixr.fon
FONTS.FON
REG_SZ
vgasysr.fon
LogPixels
REG_DWORD
0x00000060 (96)
OEMFONT.FON
REG_SZ
vga866.fon
Мой компьютер\HKEY_CURRENT_CONFIG\Software\Fonts

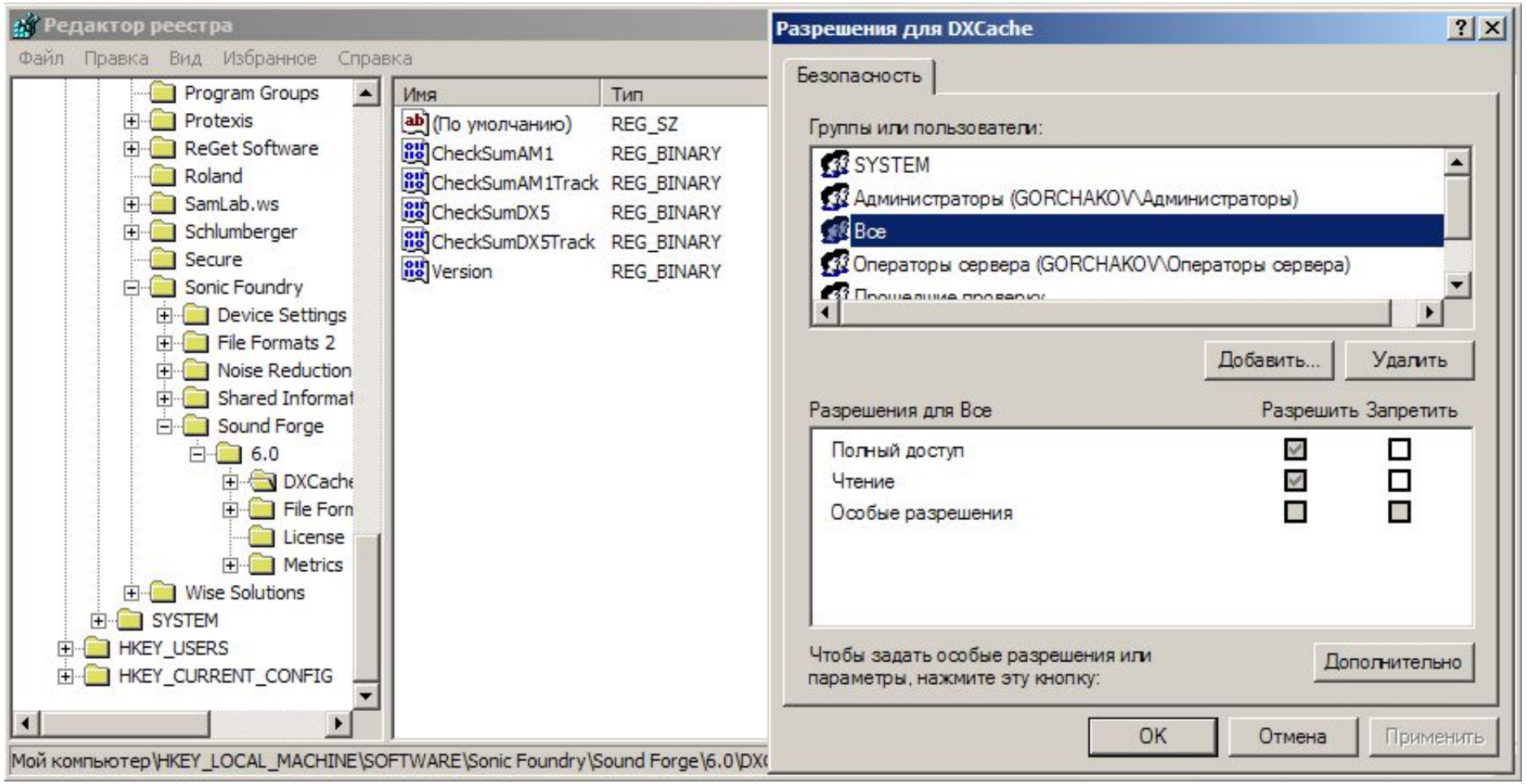
Хранение реестра в разных версиях Windows

- Windows 3.1 – reg.dat
- Windows'98 – system.dat, user.dat
- Хранение реестра в Windows 2000/XP



а также в "C:\Documents and Settings*UserName*\NTUSER.DAT" и NTUSER.DAT.LOG

Безопасность ключей реестра



Структура реестра

Разделы реестра

- HKEY_LOCAL_MACHINE – информация о компьютерной системе, содержит все остальные ветви
- HKEY_CLASSES_ROOT – ассоциации между приложениями и типами файлов, информация OLE, ассоциированную с объектами COM
- HKEY_CURRENT_CONFIG – текущий аппаратный профиль
- HKEY_CURRENT_USER – данные текущего пользователя
- HKEY_USERS – все профили пользователей

Типы файлов реестра

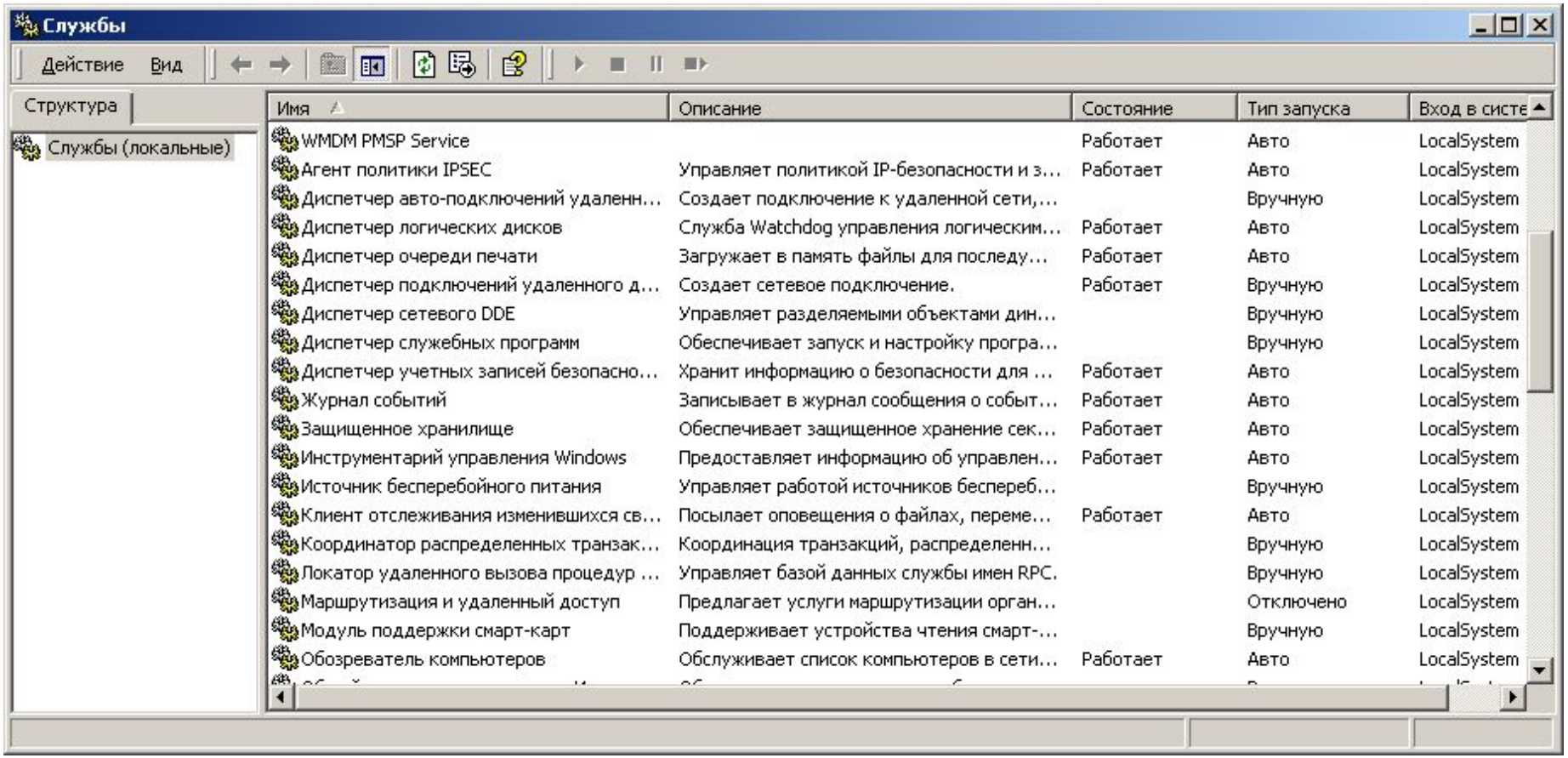
- Без расширения – копия куста
- alt – резервная копия HKEY_LOCAL_MACHINE\System
- log – журналы транзакций
- sav – копии кустов на момент завершения текстовой фазы установки

Резервирование реестра

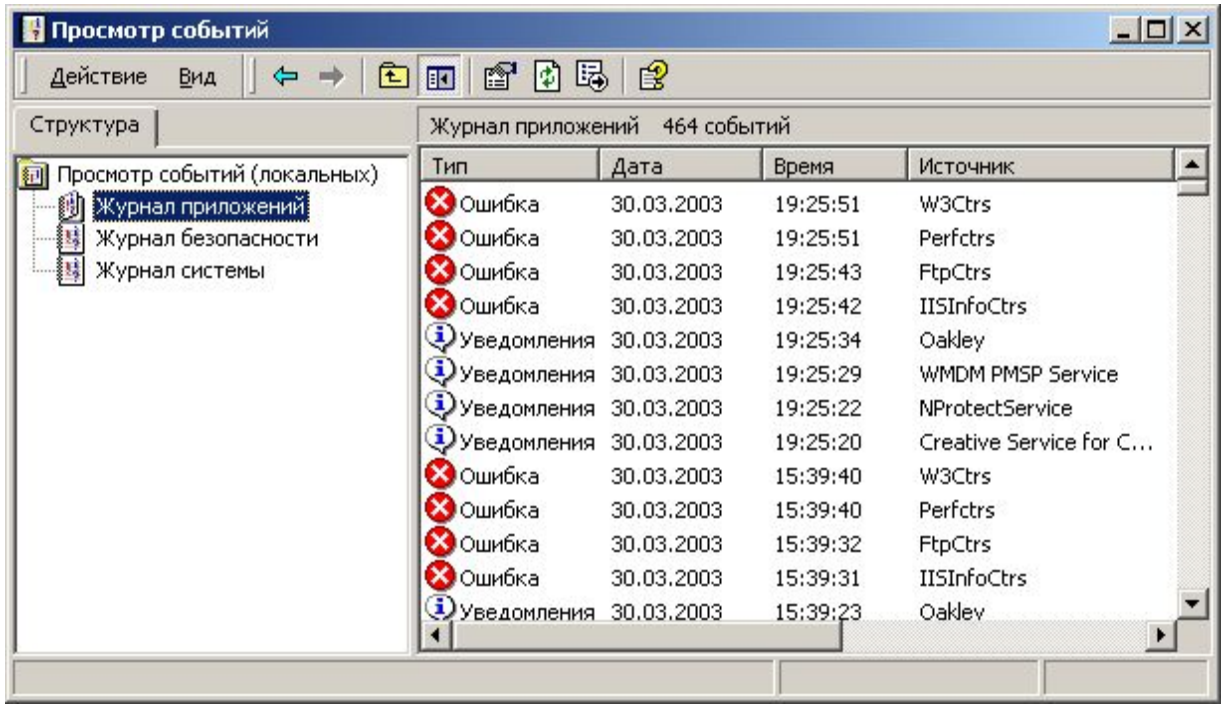
WINNT\system32\config:

- AppEvent.Evt, default, default.LOG, dafault.sav, SAM, SAM.LOG, SecEvent.Evt, SECURITY, SECURITY.LOG, Software, Software.LOG, Software.sav, SysEvent.Evt, System, SYSTEM.ALT, System.LOG, System.sav, userdiff

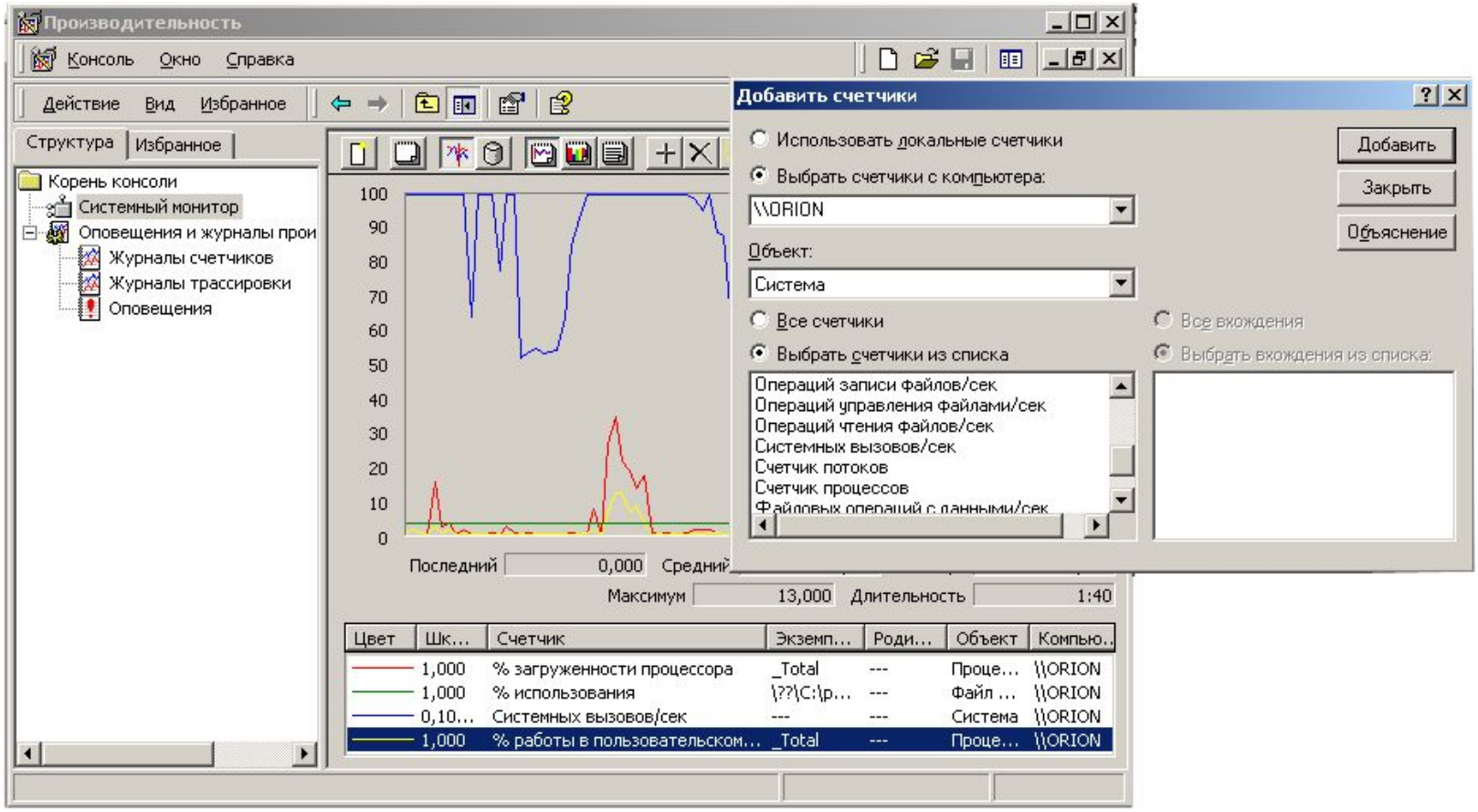
Настройка служб



Просмотр системных журналов



Просмотр производительности



Командная строка

```
D:\SASHA>help start
Запуск указанной программы или команды в отдельном окне.

START ["заголовок"] [/Dпуть] [/I] [/MIN] [/MAX] [/SEPARATE | /SHARED]
      [/LOW | /NORMAL | /HIGH | /REALTIME| /ABOVENORMAL | /BELOWNORMAL]
      [/WAIT] [/B]
      [команда/программа] [параметры]

    "заголовок" Заголовок окна.
    путь        Рабочий каталог.
    B           Запуск приложения без создания нового окна с отключением
                обработки сочетания клавиш ^C. Если приложение не
                обрабатывает сочетание клавиш ^C самостоятельно,
                единственным способом его прерывания является
                использование сочетания клавиш ^Break.
    I           Новой средой станет исходная среда, переданная
                cmd.exe, а не текущая среда.
    MIN         Запуск команды/программы в свернутом окне.
    MAX         Запуск команды/программы в развернутом окне.
    SEPARATE    Запуск 16-разрядной программы Windows в отдельной
                области памяти.
    SHARED      Запуск 16-разрядной программы Windows в общей
                области памяти.
    LOW         Запуск приложения с приоритетом IDLE.
    NORMAL      Запуск приложения с приоритетом NORMAL.
    HIGH        Запуск приложения с приоритетом HIGH.
    REALTIME    Запуск приложения с приоритетом REALTIME.
    WAIT        Запуск приложения с ожиданием его завершения.
    ABOVENORMAL Запуск приложения с классом приоритета ABOVENORMAL
    BELOWNORMAL Запуск приложения с классом приоритета BELOWNORMAL
    команда/программа
                Если это внутренняя команда cmd.exe или пакетный файл,
                обработчик команд (cmd.exe) запускается с ключом /K.
                Это означает, что окно не будет закрыто после завершения
```

Командные файлы

```
IF EXIST имя_файла. (
    del имя_файла.
) ELSE (
    echo имя_файла. missing.
)
```

Изменение команды IF при включении расширенной обработки команд:

IF [/I] строка1 оператор_сравнения строка2 команда

IF CMDEXTVERSION число команда

IF DEFINED переменная команда

где оператор сравнения принимает следующие значения:

EQL - равно NEQ - не равно

LSS - меньше LEQ - меньше или равно

GTR - больше GEQ - больше или равно

Программирование в Shell

```
var WshShell = WScript.CreateObject("WScript.Shell");
var WshSysEnv = WshShell.Environment("SYSTEM");
WScript.Echo(WshSysEnv("NUMBER_OF_PROCESSORS"));

var WSHShell = WScript.CreateObject("WScript.Shell");
intDoIt =  WSHShell.Popup("Текст в окне", 5, "Текст заголовка", 65);
WScript.Quit(0);

var WSHShell = WScript.CreateObject("WScript.Shell");
//Создает ключ HKCU\\MyRegKey со значением 'Ключ верхнего уровня'
WSHShell.RegWrite("HKCU\\MyRegKey\\", "Ключ верхнего уровня");
//Создает ключ HKCU\\MyRegKey\\Entry со значением 'Ключ второго уровня'
```