Лекция 2

ОПЕРАЦИОННИ СИСТЕМИ

гл. ас. д-р Р. Начева

Катедра “Информатика”, ИУ - Варна

Съдържание на лекцията

- Еволюция на ОС
- Архитектура на софтуерна система
- Системен и потребителски режим
- Приложен програмен интерфейс
- Видове архитектури на ОС
- Архитектура на Windows NT
- Процес на зареждане на ОС

Еволюция на ОС

Според A. Tanenbaum, Modern Operating Systems:

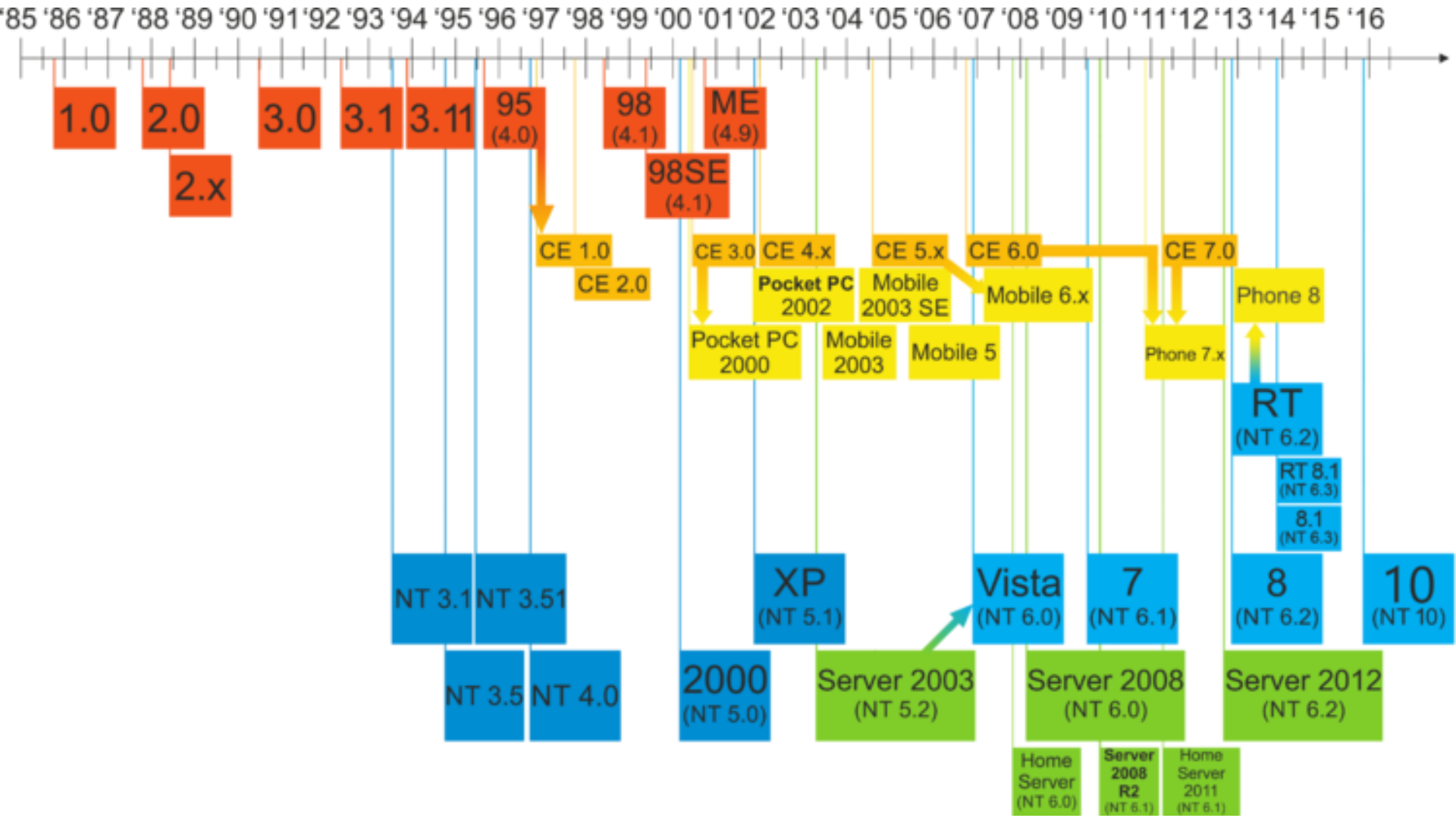
- *Първо поколение (средата на 40-те до средата на 50-те г. на 20 в.)* - първите електронни цифрови компютри нямат операционни системи. Програмите се въвеждат бит по бит в редове от механични превключватели (plug boards - табла). Не са се използвали програмни езици и операционните системи.
- *Второ поколение (средата на 50-те до средата на 60-те г. на 20 в.)* – в изследователските лаборатории на General Motors се внедряват първите операционни системи през 50-те години на миналия век за IBM 701.
- *Трето поколение (средата на 60-те до 80-те г. на 20 в.)* – появява се многозадачната работа; по-добро управление на ресурсите; концепцията за мултипрограмиране, при която няколко задачи се зареждат в основната памет едновременно; процесорът превключва от задача на задача при необходимост.

Еволюция на ОС

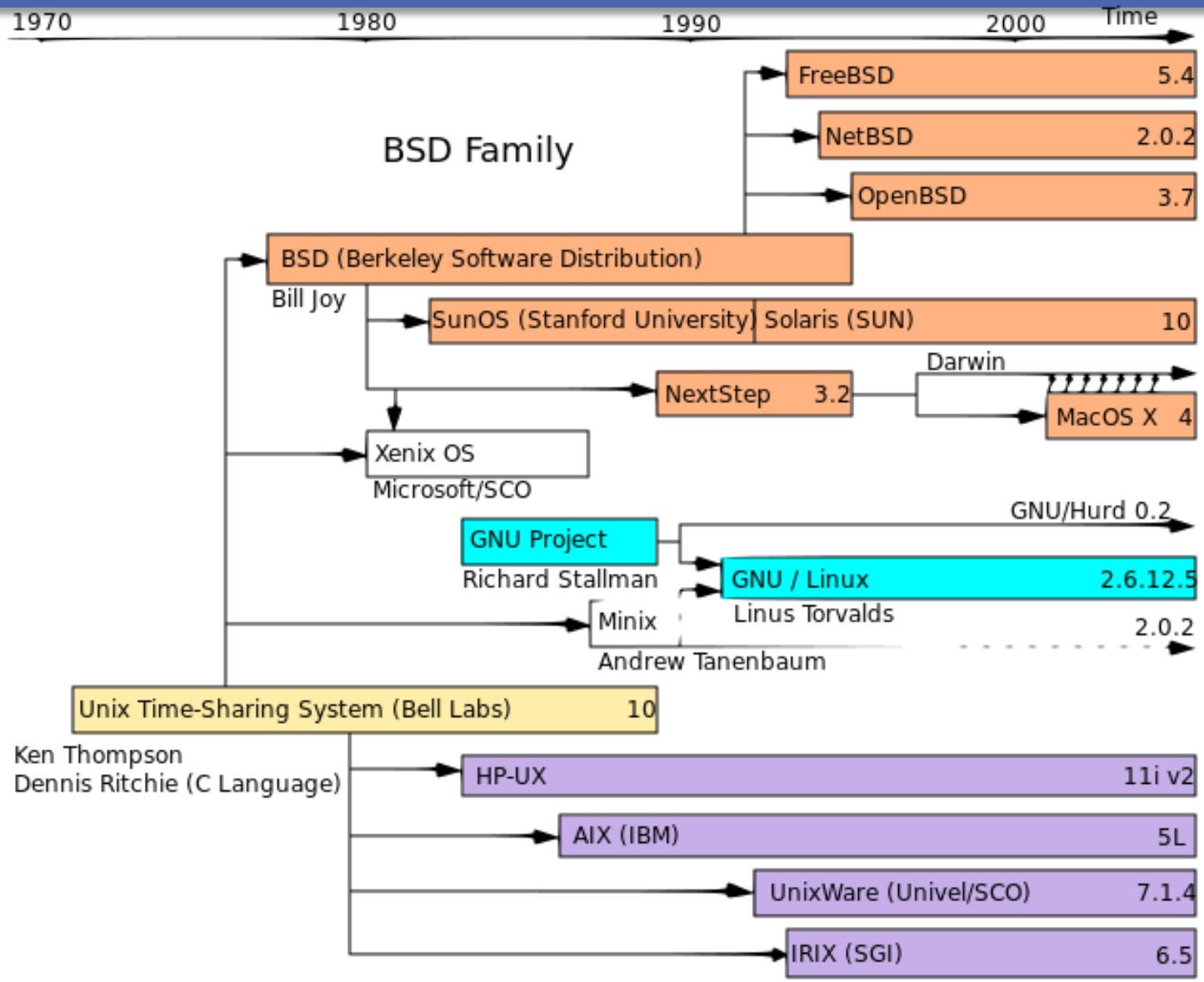
Според A. Tanenbaum, Modern Operating Systems:

- *Четвърто поколение (80-те г. на 20 в. досега)* - с разработката на големите интегралните схеми, чипове, въведена е операционна система в персоналния компютър и работната станция. Микропроцесорната технология еволюира - изграждат се мощни за времето си настолни компютри. Две операционни системи са доминирали при десктоп ОС: MS-DOS на Microsoft, Inc. за IBM PC и други машини, използващи процесора Intel 8088 и неговите наследници, и UNIX, която е доминираща при големите персонални компютри, използващи Motorola процесори 6899.

Еволюция на ОС: История на Windows



Еволюция на ОС: История на UNIX



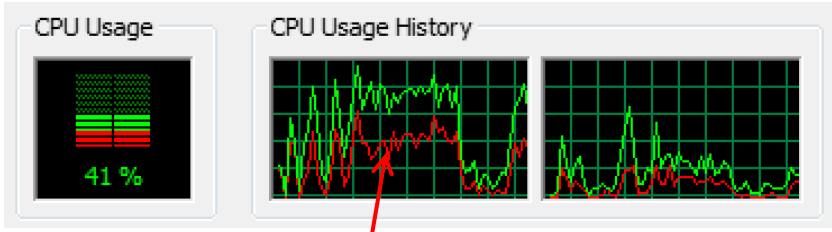
Архитектура на софтуерна система

- Архитектурата на софтуерната система описва основните ѝ компоненти, техните взаимоотношения (структури) и как взаимодействат помежду си.
- Нефункционалните решения се отделят от функционалните изисквания.
- Архитектурата служи като план за системата. Тя осигурява абстракция за управление на сложността на системата и създаване на механизъм за комуникация и координация между компонентите.
- Тя определя структурирано решение, което да отговаря на всички технически и оперативни изисквания, като същевременно оптимизира общите атрибути за качество като производителност и сигурност.

Системен и потребителски режим

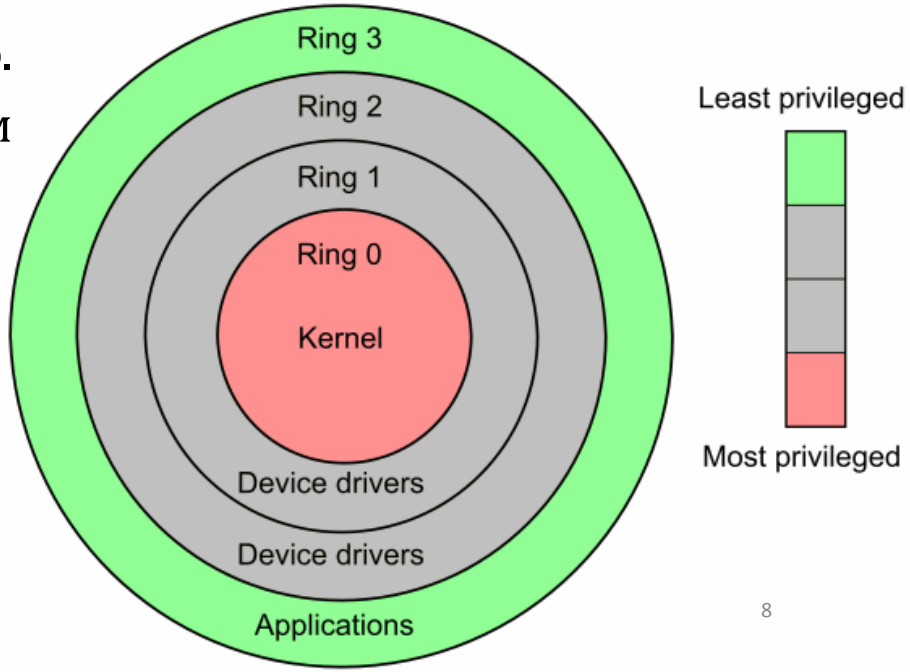
Системен режим (kernel mode, ring 0)

- изпълняващият се код има неограничен достъп до съответния хардуер. Той може да изпълни всяка инструкция за CPU и да посочи всеки адрес на паметта. Това е режим, запазен за най-надеждните функции на ОС, на най-ниско ниво. Прекъсванията в този режим спират работата на машината.



Системен режим (червено)

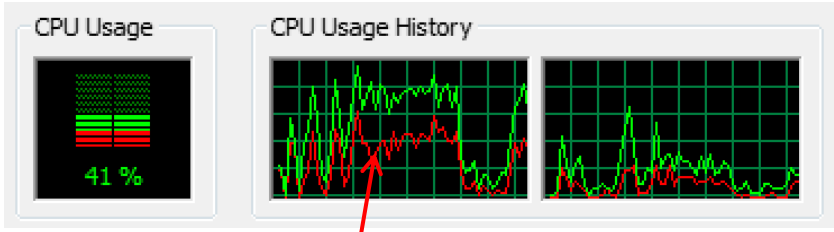
Нива на защита на изпълнявания от ОС код



Системен и потребителски режим

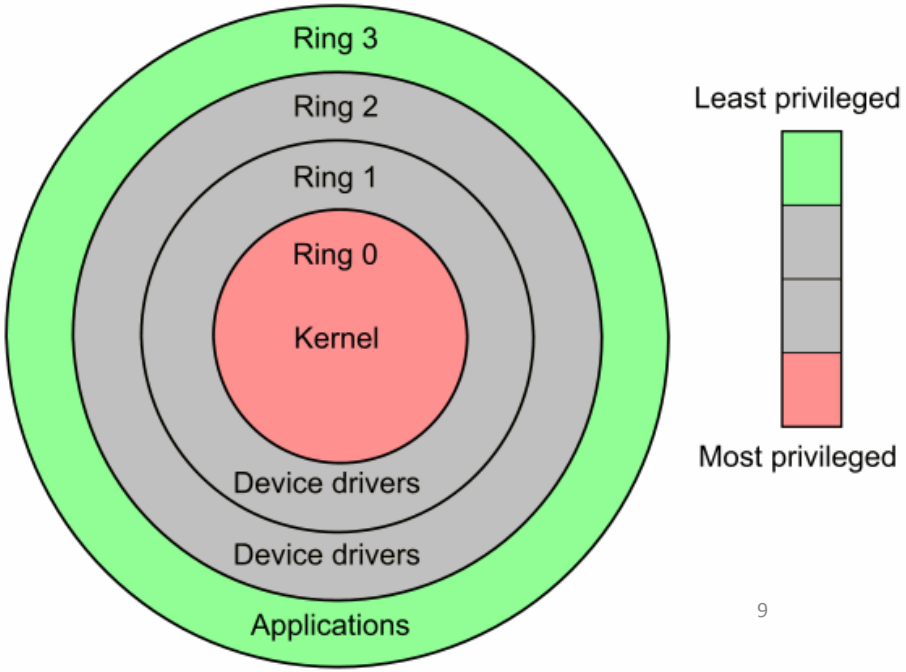
Потребителски режим (user mode, ring 3)

- Изпълняващият се програмен код няма възможност за директен достъп до хардуера или паметта. Кодът делегира на системните API достъпа до хардуер или паметта. Поради защитата, предоставена от този вид изолация, сривовете в потребителския режим винаги се възстановяват. По-голямата част от кода се изпълнява в потребителски режим.



Системен режим (червено)

Нива на защита на изпълнявания от ОС код



Приложен програмен интерфейс

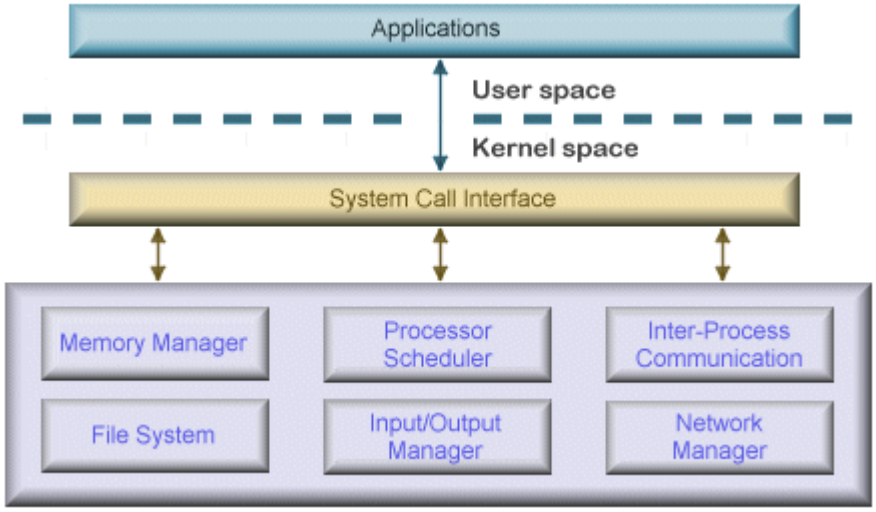
Application Programming Interface (API)

- Набор от команди, функции, протоколи и обекти, които програмистите могат да използват за създаване на софтуер или за взаимодействие с външна система. Той предоставя на разработчиците стандартни команди за извършване на общи операции. Интегрират се в т. нар. SDK (software development kit).
- Например Windows API предоставя на разработчиците контроли и елементи от потребителския интерфейс, като прозорци, ленти за превъртане (skroll bars) и диалогови прозорци. Също така предоставя команди за достъп до файловата система и извършване на файлови операции, като например създаване и изтриване на файлове. Освен това Windows API включва мрежови команди, които могат да се използват за изпращане и получаване на данни през локална мрежа или интернет.
- Вж. повече на Windows API Index.

Видове архитектури на ОС

Монолитен модел на ОС

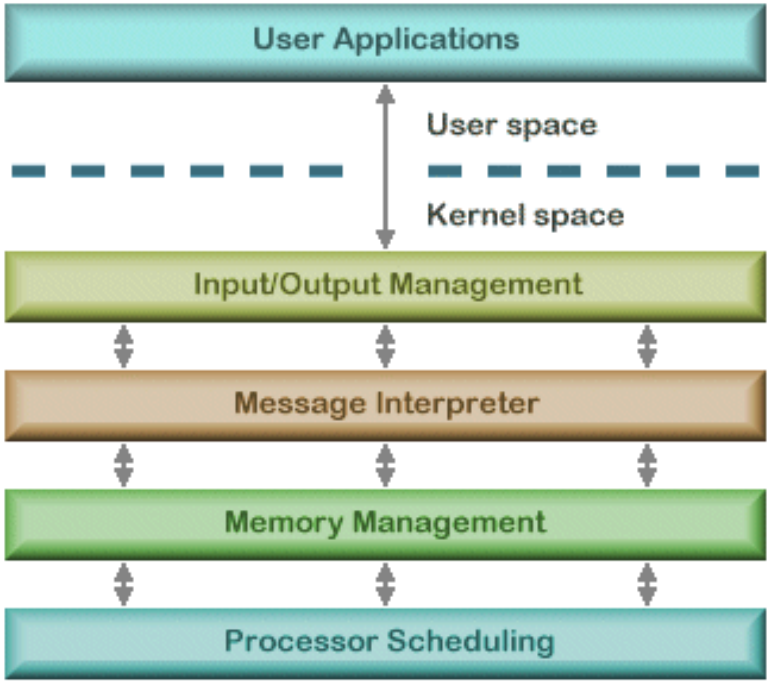
- В ранните монолитни системи, всеки компонент на операционната система се съдържа в ядрото, може да комуникира директно с всеки друг компонент и има неограничен достъп до системата. Въпреки, че това прави операционната система много ефективна, но означава и, че грешките са по-трудни за изолиране и има голям риск от сривове в ОС поради погрешен или злонамерен код.



Видове архитектури на ОС

Многослоен модел на ОС

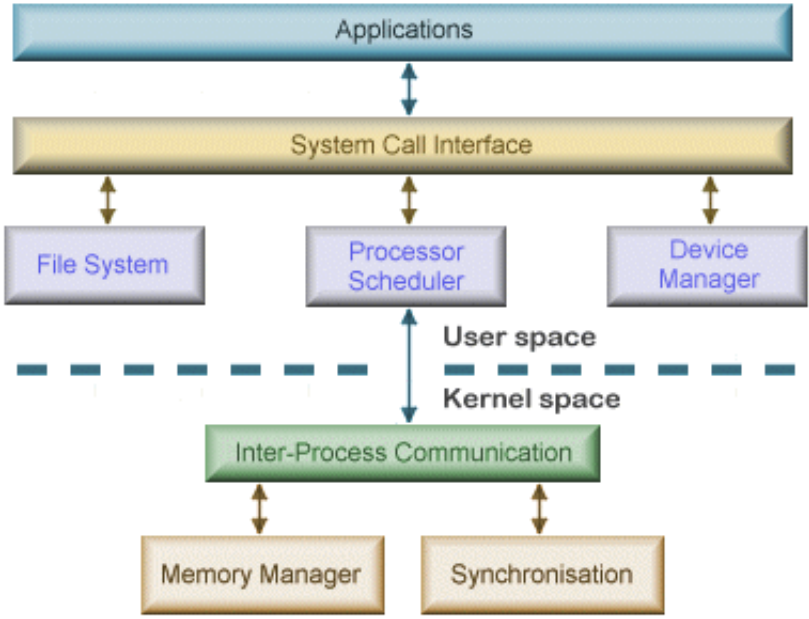
- Прилага модулен подход, който групира компоненти с подобна функционалност в слоеве, за управляване на сложността на ОС. В този вид архитектура всеки слой комуникира само със слоевете непосредствено над и под него, а слоевете от по-ниско ниво предоставят услуги на тези от по-високо ниво, използвайки интерфейс, който скрива тяхното изпълнение. Въпреки че този модулен подход налага структура и съгласуваност на ОС, опростявайки дебъгването и модифицирането, заявяването на услуги от потребителски процес може да премине през много слоеве преди да се обслужи. Ядрото все още е податливо на грешен или злонамерен код.



Видове архитектури на ОС

Модел на микроядро на ОС

- Включва само много малък брой услуги в ядрото в опит да се запази малко и мащабируемо. Услугите обикновено са: управление на паметта на ниско ниво, комуникация между процесите и основна синхронизация на процесите. Повечето компоненти на ОС, като управление на процеси и управление на устройства, се изпълняват извън ядрото с по-ниско ниво на системен достъп.
- Този тип архитектури са силно модулни. Сривът на работата на компонентите на ОС извън ядрото не влияе на стабилността на ОС.

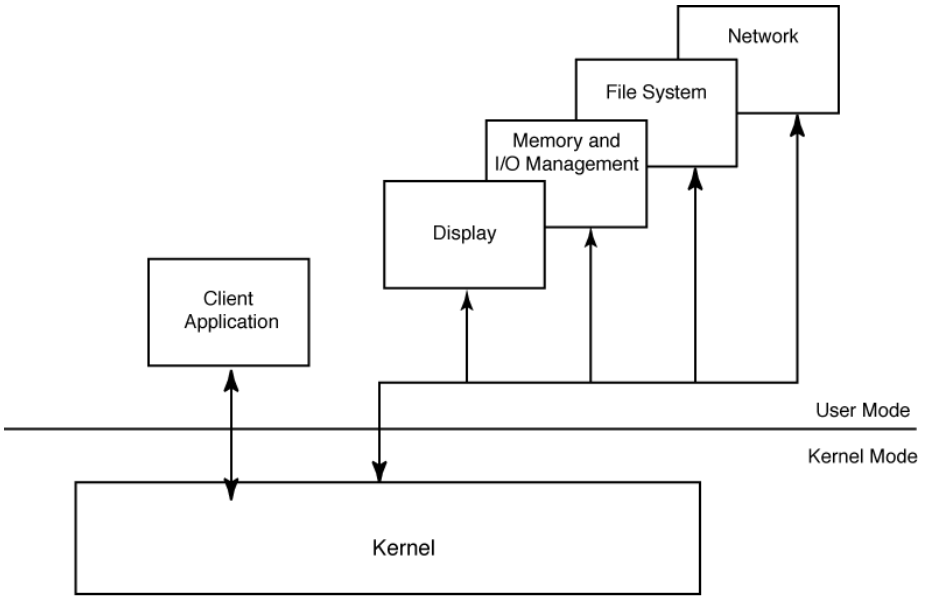


Недостатъкът е повишено ниво на комуникация между модулите, което може да влоши производителността на системата.

Видове архитектури на ОС

Модел "клиент – сървър" на ОС

- Комуникацията на компонентите е основана на парадигмата "клиент-сървър", отнасяща се до организацията на модулите на ОС. Когато даден процес се нуждае от изпълнение на конкретна услуга, той се счита за клиент. Сървърът е този модул, който отговаря на заявката.

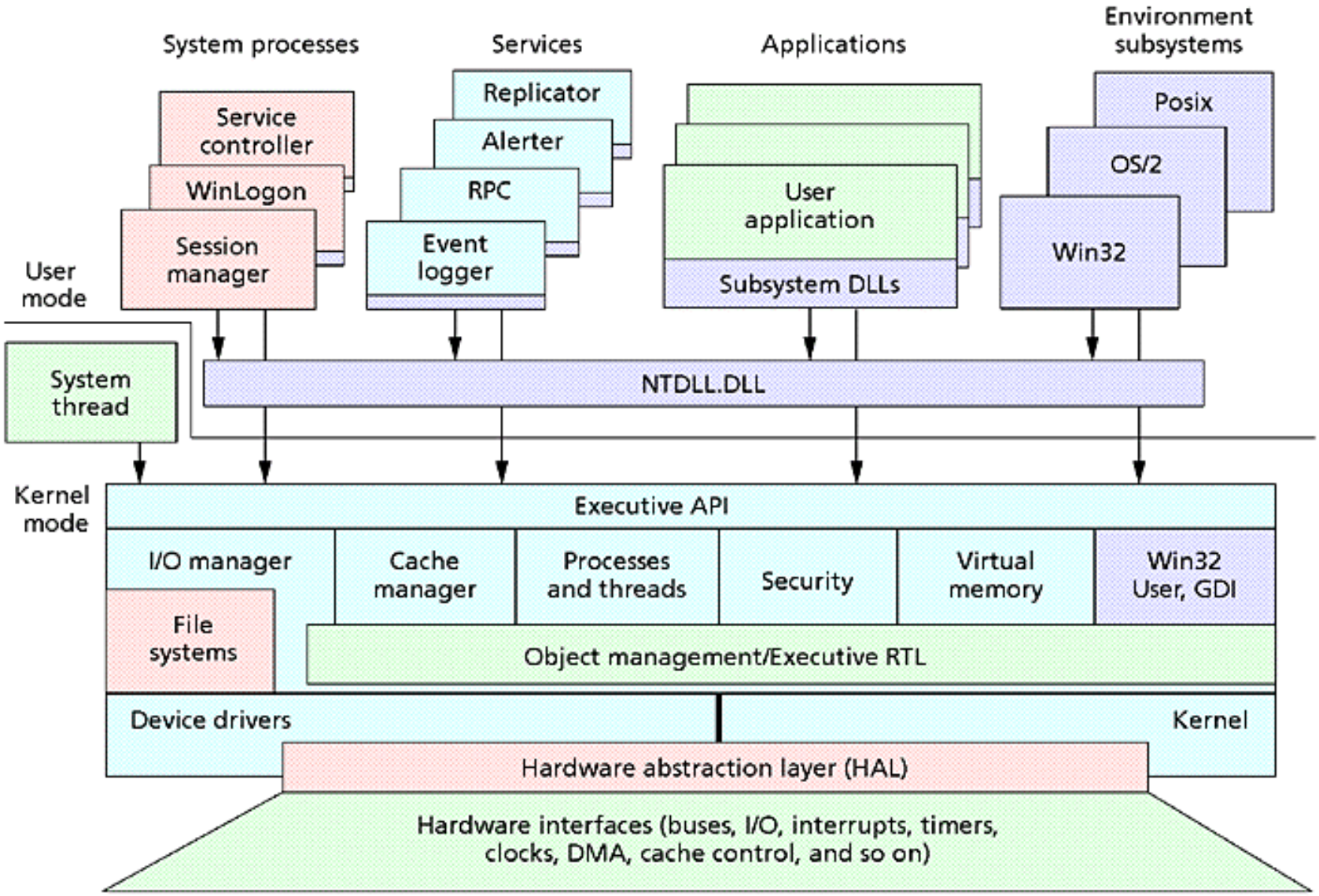


Архитектура на Windows NT

Windows NT (NT - New Technology) е:

- семейство операционни системи на Microsoft, първата версия от които е пусната през юли 1993 година;
- базирана на език от високо ниво;
- процесорно-независима (преносима);
- многозадачна;
- Многопотребителска.
- Включват се: NT 3.1 • NT 3.5 • NT 3.51 • NT 4.0 • 2000 • XP • Server 2003 • Vista • Home Server • Server 2008 • 7 • 2008 R2 • Windows 8 • Server 2012 • **Windows 10** • Server 2016 • Server 2019;
- Някои функционалности: Windows Shell, Windows API, Native API, Active Directory, Group Policy, Hardware Abstraction Layer, NTFS, BitLocker, Windows Store, Windows Update, and Hyper-V.

Архитектура на Windows NT



Архитектура на Windows NT

Потребителски режим на Windows NT:

- Непривилегирован режим на Windows NT и няма пряк достъп само до хардуера, ограничен достъп до паметта. Например, изпълнението на потребителските програми се осъществява в т. нар. пясъчници с наложени ограничения.
- Потребителският режим се състои от подсистеми, които могат да подават входно-изходни заявки към съответните драйвери за системен режим чрез Мениджъра за управление на В/И (I/O manager).
- Слоят за потребителския режим се състои от подсистемите на обкръжението, приложения и системни процеси и услуги.

Архитектура на Windows NT

Подсистеми на обкръжението:

- Компоненти на Windows NT, които поддържат работата на приложения от различни архитектури на операционната система. Осигуряват необходимата "среда", в която приложенията могат да работят. Те позволяват поддръжка на различни платформи за приложения, написани за различни операционни системи.
- Подсистемите на обкръжението са:
  - Подсистема Win32 за работа с 32-битови Windows приложения
  - OS / 2 подсистема за изпълнение на OS / 2 1.X символни приложения (не поддържа OS / 2 Presentation Manager GUI или Warp версии)
  - POSIX подсистема за работа с POSIX.1-съвместими приложения

Архитектура на Windows NT

Системни услуги:

- Управляват се от конзолата Services.msc.
- Това са компоненти на Windows NT, които позволяват създаването и управлението на процеси на фонов режим и обикновено стартират, когато машината се зареди.
- Не се стартират от крайния потребител и продължат да се изпълняват дълго след като потребителят е излязъл от системата (logoff).
- Услугите управляват разнообразни функции, включително мрежови връзки, звук на високоговорители, архивиране на данни, потребителски идентификационни данни и цветове на дисплея.
- Услугите на Windows изпълняват аналогични функции с демоните на UNIX.

Архитектура на Windows NT

Системни библиотеки:

- Dynamic-Link Library (DLL) е програмна библиотека употребявана от Microsoft Windows. Въведена е за намаляване на употребяваната RAM памет и на твърдия диск. Ако има програмен код, който се употребява повече от един път, той се обобщава в един файл (библиотека) и се зарежда само един път в оперативната памет.[1]
- Например DLL Comdlg32 се използва за изпълнение на функции, свързани с използване на диалогови прозорци.
- Чрез използването на DLL се намалява употребата на системните ресурси и се реализира модулния принцип на архитектурата на ОС.
- Могат да възникнат DLL конфликти, ако съществува зависимост между програма или друга DLL с дадена системна библиотека. Например необходимата DLL е обновена до по-нова версия; презаписана е с по-стара версия; премахната е от компютъра.

Източник: https://bg.wikipedia.org/wiki/Dynamic-Link_Library

Архитектура на Windows NT

NTDLL.dll:

- Намира се в C:\Windows\system32.
- Съдържа функции на ядрото на Windows NT (**ntoskrnl.exe**).
- Осъществява връзката между процесите от потребителския режим и системния режим.
- Използва Windows Native API за извикване на услуги от системен режим.
- Връзки с други библиотеки и процеси: https://www.webcitation.org/6D0rHn1dU?url=http://netcode.cz/img/83/nativeapi.html .

Архитектура на Windows NT

Системен режим:

- Режимът има пълен достъп до хардуера и системните ресурси на компютъра и изпълнява код в защитена зона в паметта. ОС контролира достъпа до приоритизиране на нишки, управление на паметта и взаимодействието с хардуера.
- Системният режим спира достъпа на услугите и приложенията, изпълнявани в потребителски режим, до критични области на операционната система, до които не би трябвало да имат достъп.
- Кодът, който се изпълнява в системен режим, включва: Windows Executive, който се състои от много модули, изпълняващи специфични задачи; ядрото, което предоставя услуги на ниско ниво, използвани от Windows Executive; слоя за абстракция на хардуера (HAL) и драйвери на ядрото.

Архитектура на Windows NT

Windows Executive:

- Занимава се с управление на обекти, входа и изхода, сигурността и управление на процесите. Модулите са разделени на няколко подсистеми, сред които са Cache Manager, Configuration Manager, I/O Manager, Local Procedure Call (LPC), Memory Manager, Object Manager, Process Structure and Security Reference Monitor (SRM).

Архитектура на Windows NT

Ядро на ОС:

- Основен софтуерен компонент на ОС;
- Има неограничен достъп до всички ресурси на системата;
- За Windows NT - ntoskrnl.exe.
- Управлява входно-изходните заявки от софтуера и ги транслира в инструкции за CPU и останалите хардуерни компоненти.
- Всички системни компоненти и библиотеки, които се изпълняват в системен режим, вкл. и ядрото на Windows NT, са описани на: https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/kernel-mode-managers-and-libraries.

Архитектура на Windows NT

Hardware Abstraction Layer (HAL):

- Софтуерният интерфейс между хардуера и останалата част от операционната система.
- HAL е реализирана като DLL и е отговорен за защитата на NT от спецификите на хардуера като контролели и входно-изходните интерфейси.
- Този слой прави NT преносима ОС, защото другите компоненти на ОС не се “интересуват” от физическата платформа, на която тя работи. Това е hal.dll.
- При персоналните компютри, HAL може основно да се счита за драйвер за дънната платка, който получава инструкции от компютърните езици на по-високо ниво при комуникация с компоненти на по-ниско ниво, но не позволява директен достъп до хардуера.

Процес на зареждане на ОС

Boot process

- След включване на компютъра BIOS инициализира хардуера;
- BIOS извиква съхранен код в MBR (Master Boot Record);
- MBR зарежда код от bootsector на активния дял;
- Bootsector зарежда и стартира bootloader от файловата система.

Въпроси за самопроверка

- Какво е архитектурата на софтуерната система?
- Избройте видове архитектури на ОС, които познавате?
- Какво е BIOS и предназначението му в процеса на зареждане на ОС?
- Какво представляват MBR и GPT? Какви са основните разлики между тях?

Допълнителна литература

- Major Operating Systems and historical evolution
- 8 ways to boot into Windows 10 Safe Mode
- Windows 10 : How to choose Diagnostic startup mode
- Repair Windows 10 using Automatic Repair
- FIX: KERNEL SECURITY CHECK FAILURE on Windows 10/8/8.1