Előadás_#10.

1. A Windows NT kialakulása

   A Windows NT (New Technology) a Microsoft cég új generációs, eredetileg vállalati felhasználókat megcélzó operációs rendszerének az elnevezése. A Windows NT operációs rendszerrel a Microsoft a DOS-, illetve Windows rendszereket kívánta felváltani. Szoftver fejlesztés azonban nem csak a Microsoft-nál történt, az IBM már 1987. áprilisában bemutatta az OS/2 nevű operációs rendszerének első verzióját. Igazi siker azonban csak az 1992-ben bemutatott valódi 32-bites OS/2 2.0 lett (mely képes volt Windows 3.1-es programok futtatására is), majd az 1994-ben bemutatott OS/2 3.0 Warp lett, mely hatékony és konzisztens rendszer volt. Egy többé kevésbé kényszerű együttműködés után a Microsoft 1990-ben szakított az IBM-mel és 1993. júliusában bemutatta saját 16-bites Windows NT 3.1 nevű operációs rendszerét. Ezek után a Microsoft fejlesztései is a 32-bites irányba indultak el, 1996. júliusában került bemutatásra a Windows NT 4.0. Az első említésre méltó 64-bites rendszer a Windows XP 64-bit volt, 2003 márciusában. A Microsoft asztali- és szerver operációs rendszerei ma is az NT technológiára épülnek.

   Az eddig megjelent NT alapú Windows verziók:

Verzió	Build	Megnevezés	Megjelenés
3.1	528	Windows NT 3.1	1993.07.27
3.5	807	Windows NT 3.5	1994.07.21
3.51	1057	Windows NT 3.51	1995.05.30
4.0	1381	Windows NT 4.0, 32-bit	1996.07.29
5.0	2195	Windows 2000	2000.02.17
		Windows 2000 Datacenter	2000.09.26
5.1	2600	Windows XP	2001.10.25
5.2	3790	Windows XP, 64-bit	2003.03.28
		Windows Server 2003	2003.04.24
		Windows Server 2003 R2	2005.12.06
6.0	600x	Windows Vista	2006.11.30
		Windows Server 2008	2008.02.27
6.1	760x	Windows 7	2009.10.22
		Windows Server 2008 R2	2009.11.22
		Windows Home Server 2011	2011.04.06
6.2	9200	Windows 8	2012.10.26
		Windows Server 2012	2012.09.04
6.3	9600	Windows 8.1	2013.10.18
		Windows Server 2012 R2	2013.10.18
10.0	10240	Windows 10	2015.07.29
1607	14393	Windows Server 2016	2016.10.12
1809	17763	Windows Server 2019	2018.11.13
11.0	22471	Windows 11	2021.10.05

A Windows NT megalkotásakor a Microsoft két alapvető szempontot kellett hogy figyelembe vegyen. Egyrészt az aktuális piaci igényeket, követelményeket ki kell elégíteni (ez lássuk be nem minden verziónál ment zökkenőmentesen), másrészt a továbbfejlesztési lehetőségek, és a saját (a piacot irányító, befolyásoló) célkitűzések is meg kell hogy jelenjenek a termékben.

2. Az első Windows NT kiadássokkal szemben támasztott követelmények

   Egy valós 32-bites (csak az Windows NT 4.0 óta), bármikor megszakítható (Preemptív) és újrahívható (Reentrant) [a rendszerhívásokat több alkalmazás is meghívhatja egyszerre, és ezek nem blokkolódnak, akkor sem, ha már valakit éppen kiszolgál az adott rendszerhívás], valamint virtuális memóriakezelésen alapuló operációs rendszer megvalósítása volt az alapvető cél. A legfontosabb követelmények a következők voltak:

   - Fusson különböző hardver architektúrákon.
   - Fusson multiprocesszoros környezetben.
   - Legyen skálázható az összes rendelkezésre álló erőforrás viszonylatában.
   - Vegye figyelembe a DCE (Distributed Computing Environment) ajánlásait, azaz legyen képes elosztott hardver környezetben is futni.
     [A DCE egy, a 90-es évek elején tett kísérletet a különböző hardver architektúrák szoftveres kompatibilitásának irányába.]
   - Legyen alkalmas a 16-bites MS-DOS és a Windows 3.1-es alkalmazások futtatására. (Ez a követelmény már a korai 32-bites időktől is csak kompromisszumok árán, és csak részben teljesült, a 64-bites verziók alatt pedig már közvetlenül nem is oldható meg.)
   - Teljesítse a POSIX 1003.1 szabványt.
     [Portable Operating System for UniX / formális neve: IEEE1003, hivatalos elnevezése ISO/IEC 9945).
   - Teljesítse az ipari környezetnek megfelelő biztonsági szabványokat.
   - Használjon UNICODE-ot a karakterek és stringek ábrázolására.
     [A UNICODE a karakterek gépi ábrázolásának szabványa. Mivel minden karaktert 16-biten ábrázol, így szinte minden nyelv ábécéjének teljes karakterkészletét lehetséges azonos kódolással használni. A UNICODE így lehetővé teszi az alkalmazások nyelvterülettől független elkészítését. Az első olyan, Microsoft által gyártott operációs rendszer, amely binárisan is egységes az egész világon, a Windows 2000 volt.]
     Az UNICODE előnyeit kihasználandó, az NT 4.0 a belső karakterábrázolásánál a Microsoft az UNICODE használta mellett döntött. Tekintettel arra, hogy az

UNICODE használata még egyáltalán nem volt általános, így a Windows NT 4.0 alatt futó alkalmazások jelentős része még nem használt UNICODE-ot. Ez az oka annak, hogy a Win32 API-ban (Application Programming Interface / Alkalmazásprogramozási interfész), mely például a karakterláncokat (String változókat) mint paramétereket közvetíti a folyamatok között, kétféle függvényt (Routine) kellett definiálni:

- széles: 16-bites UNICODE karakteres
- keskeny: 8-bites ANSI karakteres

A 8-bites ANSI függvények megvalósítása két egyszerű lépésben történik. Első lépés a String paramétereket átkódolása 16-bites UNICODE-ba, a második lépés pedig a függvény UNICODE-os megfelelőjének meghívása.

3. A Microsoft célkitűzései

A Microsoft az Windows NT alapú operációs rendszerek mindenkori minőségének, korszerűségének és továbbfejleszthetőségének biztosítása érdekében jelölték meg a következő célokat:

- A Windows NT kódja legyen könnyen továbbfejleszthető, vagyis kiterjeszthető. (Ez a kitétel nyilván csak a kód egyes részeire vonatkozik, azaz a kód csak részben nyílt, maga a teljes programkód nem publikus.)
- A kód legyen alkalmas az új hardver platformokon történő futtatásra is.
- A rendszer legyen több szempontból is megbízható, robusztus.
  - két program futása ne befolyásolja egymást
  - a rendszer összeomlást el kell kerülni
  - a belső komponensek együttműködése zökkenőmentes legyen
- Megfelelő kompatibilitás a meglévő rendszerekkel a felhasználói interfész, és alkalmazásprogramozási interfész (API) szintjén is. A kompatibilitás két alapvető szintje:
  - a Microsoft korábbi operációs rendszerei: MS-DOS, Windows 3.1
  - a nem Microsoft által készített, azonban széles körben elterjedt rendszerek: OS/2, UNIX, NetWare, stb.

  Mára ez a szempont gyakorlatilag is okafogyottá vált.
- A rendszer a hatékonysága legyen független a hardverkörnyezettől, vagyis bármelyik hardver platformon optimálisan tudja kihasználni a rendelkezésre álló erőforrásokat.

4. A Windows NT alapú rendszerek felépítésének fő jellemzői

   A Windows NT alapú operációs rendszerek felépítése szigorúan réteges szerkezetű, az alrendszereinek működése pedig kliens-szerver kiszolgálás alapú. Az egyes rétegek interfészeken igénybevételével kommunikálnak. Az eredeti elképzelés szerint a Windows NT fejlesztői a mikrokernel használata mellett döntöttek, de ahhoz, hogy a legtöbb szolgáltató folyamat felhasználói (User) módban futhasson, a hibrid kernel megfelelőbbnek bizonyult. A szolgáltatások egy részét nem célszerű és nem is hatékony (főleg a gyakori és nagyszámú környezetváltás miatt) felhasználói módba kényszeríteni. Azokat a szolgáltatásokat, melyek intenzíven használják a hardvert és futásuk gyorsasága az egész rendszer teljesítménye szempontjából kritikus, célszerű mindenképpen kernel módban futtatni. (Ilyen szolgáltatás például a cache menedzsment, a címtartomány menedzsment, a szálütemezés, a folyamatok közti kommunikáció, a virtuális memóriakezelés, az objektumkezelő, a biztonsági alrendszer, illetve a WIN32-es alrendszer grafikus támogatása.)

   A Windows NT alapú rendszerek fejlesztésekor alapvető volt az objektumorientált szemlélet. A Microsoft az egész operációs rendszert funkciójukban pontosan definiált objektumokból építette fel. Az objektumok közti kommunikáció szintén előre meghatározott interfészek igénybevételével történik.
   A Windows NT korai verzióiban, minden szándék és elhatározás ellenére, – a hardverközeli programozás miatt – csak az alábbi három objektumorientált tulajdonságot sikerült maradéktalanul megvalósítani:

   - Adatrejtés
     Az operációs rendszer objektumai csak saját adataikat érhetik el.
   - Interfész-használat
     Az objektumok előre definiált interfészek segítségével kommunikálnak.
   - Hierarchikus objektum szerkezet
     Kernel objektumok és executive objektumok használata.

A következő objektumorientált tulajdonságokat viszont a Windows NT alapú rendszerek nem valósítják meg:

- Polimorfizmus

  Azonos néven érhetünk el különböző objektumokat, mert ez esetben az azonosító környezetétől függ, hogy melyik objektumra hivatkozunk az adott azonosítóval. Szemléletes példa a polimorfizmusra például az, amikor egy objektumon belül azonos névvel, de különböző paraméterlistával definiálunk függvényeket. Ez esetben az, hogy melyik funkciót érjük el csak az aktuálisan használt paraméterek típusától függ.

- Öröklődés

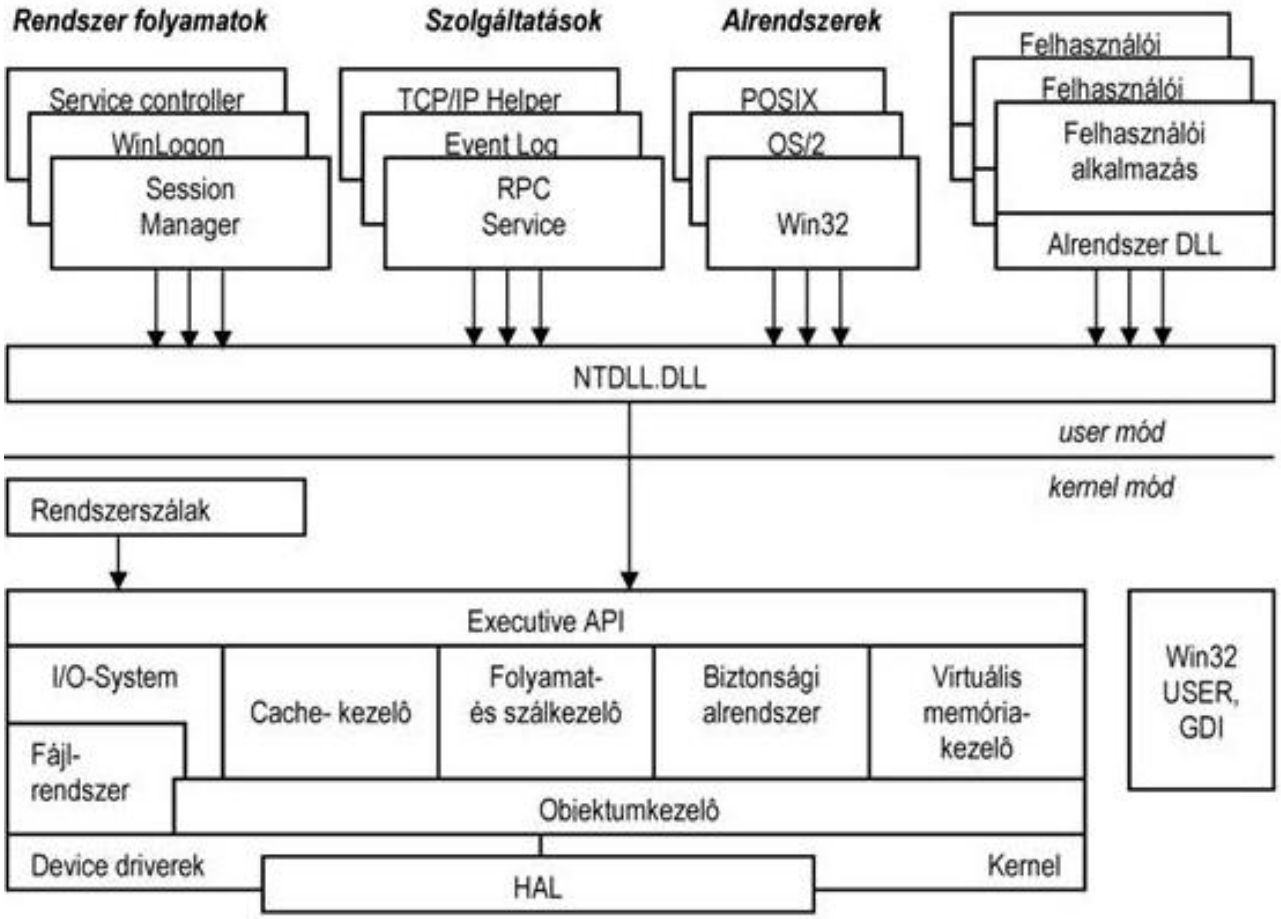
  Az objektumok hierarchikusan (is) származtathatók. Egy szülő objektum megfelelően definiált adatelemeit, függvényeit, attribútumait részben vagy egészben a származtatott objektum képes örökölni, így a későbbiekben maga is használhatja (vagy akár tovább is örökítheti) azokat.

  Fontos megjegyezni, hogy az NTFS használata óta az öröklődés már részben megvalósul!

- Dinamikus adattípus kötés

  Lehetőség van adattípusokkal paraméterezett objektumok definiálása is, de maga a dinamikus adattípus hozzárendelés gyakorlatilag mindig futásidőben történik, hiszen az egyes változókhoz csak ekkor rendelődik hozzá az aktuális adattípus.

5. A 32-bites Windows NT felépítésének komponensei



HAL

A HAL (Hardware Abstraction Layer) a számítógép hardverét teljesen elfedő, az operációs rendszer felépítését tekintve a legalacsonyabb szoftverréteg. A Számítógép-Architektúrák kurzuson megismertek szerint létezik ennél is hardverközelebbi szoftveres réteg – a BIOS illetve az UEFI – hiszen ezek technikailag az operációs rendszer és a hardver között helyezkedik el, viszont ezek nyilván nem részei az operációs rendszernek. Az operációs rendszer HAL fölött elhelyezkedő rétegei tehát csak a HAL-on keresztül érhetik el a hardvert. Minden Windows NT alapú operációs rendszer alatt használható CPU-hoz egyedi HAL-réteg készül, vagyis a HAL a használt CPU típusától függ. A telepítő állományok között így számos HAL-t találhatunk az \i386-os alkönyvtárában. Érdekesség, hogy a Vista előtti rendszerekben telepítéskor választódott ki, hogy milyen típusú HAL kerül felmásolásra. Többek között ezért sem volt biztos, hogy egy másik gépbe átrakva a HDD-t egyáltalán képes lesz-e a Windows elindulni. A Vista óta szerencsére ezzel a problémával már nem találkozunk.

A HAL az azonos architektúrájú CPU-knak az egyedi CPU modelltől függő funkcióit tartalmazza. A HAL feladata, hogy CPU független szolgáltatásokat nyújtson a többi réteg számára. Azonos CPU architektúra esetén – vagyis például az összes x86-os

processzoron futó Windows NT alapú rendszer között – csak a használatban lévő HAL-ban van különbség, a többi rétegben nincs (legalábbis nem szükségszerű). Nyilván ugyanez a helyezet x64 vagy RISC esetében is. Feladatát tekintve tehát a HAL egy olyan – az aktuális hardverre támaszkodó – virtuális gépként dolgozik, melynek funkcionalitása minden rendszerben azonos.
A perifériák, azaz csatlakoztatott külső hardverek esetében egy kicsit más a helyzet. Ezekhez nyilván egyedi illesztőprogramok tartoznak, melyek függhetnek az operációs rendszer konkrét típusától, illetve a CPU architektúrájától (x86 vagy x64) is. (Erről bővebben a device driverek réteg alatt lesz szó.)
(Az 1968-ban készült "2001 Űrodisszeia" című filmben szereplő HAL9000-es computer nevében a HAL természetesen mást jelentett, egy rövidítés volt, ez: Heuristically programmed ALgorithmic computer.)

Kernel

A kernel az operációs rendszer alapfunkcióit nyújtó komponense (pl. ütemezés, megszakításkezelés). A kernel a rendszernek az állandóan a memóriában (operatív tárban), azon belül is védett módban (azaz kernel módban) futó része.
A kernelben is előfordulnak hardver specifikus kódrészletek, hisz például a környezetváltás megvalósításához ismerni kell, hogy milyen regiszterei vannak a CPU-nak. Azonban amíg a HAL a CPU típusától függ, addig a kernel csak a CPU architektúrájától. Ez azt jelenti, hogy azonos architektúrájú CPU-k esetén a kernel is azonos. Például két x86-os rendszerben a kernelréteg is azonos, függetlenül attól, hogy konkrétan milyen CPU-t használunk. Nyilván ugyanez a helyezet x64 architektúra esetében is.
A kernelre (illetve a device driverek rétegre) épülő további komponensek már hardverfüggetlenek, azaz a rendszer további komponensei már gyakorlatilag hordozhatóak. A Windows NT alapú rendszerek hordozhatósága ebből következően úgy valósul meg, hogy a hardvert bármely CPU esetében hasonló interfészen keresztül elérhető szoftverrétegek fedik el. Az első két szoftverréteg tehát a HAL és a kernel. A kernelben realizálódnak a CPU architektúrától függő funkciói (például a környezetváltás és a szálkezelés).
Más szempontból is (legalább részben) hordozhatóak a Windows NT alapú rendszerek, hiszen – a UNIX rendszerekhez hasonlóan – hordozható programnyelven íródtak. A programkód legtöbb része C programnyelvben készült, de a hardver közvetlen kezelését végző (például a megszakítás kezelés), illetve a rendszer teljesítményét nagymértékben befolyásoló (például a környezetváltás) kódrészek már Assembly nyelven készültek.

A rendszer további rétegeinek a hardvertől történő függetlenítésén túl a kernelnek az is a feladata, hogy az operációs rendszer többi komponense által használható, jól definiált alapmodulok – az úgynevezett primitívek – végrehajtását lehetővé tegye.
A primitívek használatával megvalósítandó funkciók:

- szálütemezés (Thread Scheduling)
- trap-kezelés, megszakítás- és kivételkezelés
- multiprocesszor ütemezés
- a kernel objektumok kezelése

A kernel objektumok felépítésüket tekintve egyszerűbbek, mint a többi réteg objektumai. A gyors kezelhetőség érdekében ugyanis a kernel az egyes objektumok elérésekor nem végez ellenőrzést, hanem abból a logikából indul ki, hogy a rendszer többi komponense megfelelően használják az objektumokat.

Készülék meghajtók (Device Driverek)
A device driverek a I/O-alrendszer és a számítógép hardvere közötti kapcsolatot biztosítják, oly módon, hogy a hardvert nem közvetlenül, hanem a HAL rétegen keresztül érik el. A device driverek akár forráskódjukban is hordozhatók a különböző hardver architektúrák között, illetve akár binárisan is hordozhatók az azonos architektúrájú, de különböző típusú CPU-k esetén.
Ezen réteg vizsgálatakor nem szabad megfeledkezni arról, hogy az igények érkezhetnek a felsőbb rétegekből kiindulva – nyilván a felhasználó is generál itt realizálandó igényeket – illetve a másik irányból, azaz hardver oldaláról is. Ez a kétirányúság is az egyik oka a device driverek réteg és az I/O réteg szoros kapcsolatának.
A device driverek négy típusa a következő:

- Hardver driverek
  A hardver driverek az egyes hardver komponensek I/O csatornáit közvetlenül elérve teszik lehetővé a hardver használatát. (pl. BUS driverek: PCI, USB, FireWire, SATA)
- Fájlrendszer driverek
  A fájlrendszer driverek a fájlrendszer(ek) elérésére vonatkozó igények kiszolgálását végzik, I/O műveletekké alakítva át a kéréseket.

- Szűrő típusú device driverek
  Az NT alapú rendszerek által is használt réteg szerkezetű device driver struktúra lehetőséget kínál egyedi, speciális többletfunkciók megvalósítására. Ehhez arra van szükség, hogy a kívánt szoftveresen megvalósítandó funkciót megfelelő helyre, azaz a két érintett driver illetve réteg közé implementáljuk. Jellemző, hogy a szűrő típusú device driverek egy magasabb szintű rétegben lévő driver – például a fájlrendszer rétegé – és egy alacsonyabb szintű driver – a device driver rétegben található disc driver – közé ékelődnek. Ezen a módon valósulnak meg a szoftveres RAID funkciók, a kibővített illetve hibatűrő szoftveres lemezkezelés.
- Hálózati device driverek
  A hálózati device driverek feladata a számítógép hálózatokkal kapcsolatos igények megvalósítása, továbbítása. Ehhez minden esetben kérések megfelelő konverziójára van szükség.

Executive (az utolsó komponens a kernel módban)
Az executive a Windows NT alapú operációs rendszerek magas szintű alrendszereinek szolgáltatásait (memóriakezelés, biztonság) megvalósító réteg. Az adatokat objektumokban tárolja, melyeket a megfelelően definiált interfészeken keresztül, és csak leírókkal (Handle) lehet elérni.
Az executive réteg a következő komponenseket tartalmazza:

- Folyamat- és szálkezelő
- Virtuálismemória-kezelő
- Biztonsági alrendszer (Monitor)
- Cache kezelő
- I/O-rendszer kezelő
  Itt található Plug-And-Play kezelő is – ami az ábrán nem is szerepel, mivel csak a Windows 2000 óta része a rendszernek.

Az executive réteg kezeli le és szolgálja ki az (egyébként user módban futó) NTDLL.DLL rétegben definiált függvények igényeit az alsóbb rétegek felé, így biztosítva az operációs rendszer belső objektumai közötti kommunikációt.
Az executive réteg alapvető feladata az LPC (Local Procedure Call / Lokális eljáráshívás) szolgáltatás megvalósítása. Az LPC a Windows NT alapú operációs rendszerek IPC (Inter Process Communication / Folyamatok közötti kommunikáció) eszköze, azaz csak a Windows NT alapú rendszerek belső folyamatai közötti kommunikáció biztosítására szolgál. Az LCP igénybevételével válik egy felhasználói

objektum képessé arra, hogy egy másik felhasználói objektum adott függvényét meghívja. Azaz az LPC teszi lehetővé azt, hogy az egyes alkalmazások a hozzájuk tartozó alrendszer(ek) szolgáltatásait igénybe vehessék. Az executive réteg a rendszerfolyamatok és a szolgáltatások részére biztosít még további Run-time library függvényeket, valamint különböző támogató funkciókat megvalósító függvényeket is.

NTDLL.DLL (itt kezdődnek a user módú komponensek)
Az NTDLL.DLL egy speciális, dinamikusan kapcsolódó (kölcsön)könyvtár (Dinamically Linked Library / Dinamikusan kapcsolódó [kölcsön]könyvtár. A kölcsönkönyvtár fogalmi magyarázata egy korábbi fejezetben már szerepelt.) A Windows NT alapú operációs rendszerekben az NTDLL.DLL használata megkerülhetetlen a user mód és a kernel mód közötti kommunikáció lebonyolításához. Az executive rétegben már tárgyalt objektumok közötti kapcsolattartás, az LPC (Local Procedure Call / Lokális eljáráshívás) az, amely a megfelelő függvényhívások segítségével teszi lehetővé ezt a kommunikációt is. A felhasználói objektumok csak az NTDLL.DLL igénybe vételével érhetik el az operációs rendszer kernel módban működő részét, illetve a hardvert.
Magát a kommunikációt az NTDLL.DLL néhány egyszerű lépésben hajtja végre.

1. lépés Az NTDLL.DLL-hez függvényhívás érkezik valamelyik felsőbb rétegből.
2. lépés Az NTDLL.DLL ellenőrzi a függvényhívás hívás paramétereit. (Normál esetben nincs szükség hibakezelésre.)
3. lépés Végre kell hajtani egy user mód – kernel mód váltást.
4. lépés Az NTDLL.DLL átadja a rendszerhívást a system service dispatcher-nek.
5. lépés Az operációs rendszer így már képes meghívni a kért funkciót realizáló kernel módú függvényét.

NTDLL.DLL tartalmilag az executive által kiajánlott függvényeknek megfelelő függvény csonkokból áll, melyek ugyanolyan a paraméterezésűek, mint az executive-ban lévő párjuk. Ezen kívül tartalmaz számos további függvény az alrendszerek támogatására. Ezek közül a két legfontosabb:

- dinamikus memória kezelés (Heap)
- Image Loader

Az NTDLL.DLL-nek a fent leírtakon kívül létezik számos, a felhasználók számára nem dokumentált, illetve nem publikált függvénye is:
Nem dokumentált NTDLL.DLL függvények (Angol nyelvű)

Rendszerfolyamatok

A Windows NT alapú operációs rendszerek a független felhasználói folyamatként megvalósuló rendszerfunkciókat nevezik rendszerfolyamatnak. Nyilván ez esetben user módban futó folyamatokról van szó. Ennek ellenére a rendszerfolyamatok olyan alapvető részei az operációs rendszernek, hogy ezek nélkül a Windows NT alapú operációs rendszerek nem is lennének képesek működni.

A legfontosabb rendszerfolyamatok:

- SMSS (Session Manager SubSystem / Munkamenet kezelő alrendszer)
  Az smss.exe program indításával jön létre az SMSS folyamat. A továbbiakban ez a folyamat felelős az alkalmazások elindításáért. Az SMSS indítja el azokat az alrendszereket is, melyeknek az indított folyamat futásánál szükség van. Feladatai közé tartozik még az, hogy kapcsolatot teremtsen az általa indított folyamatok és a Debugger (hibakereső rutin) között, valamint az, hogy biztosítsa a környezeti változók definiálását és elérését.
- LOGON
  A winlogon.exe futtatása indítja a felhasználók beléptetéséért és kiléptetéséért felelős LOGON folyamatot, melyet a SAS (Secure Attention Sequence / Biztonsággal felügyelt programszakasz) billentyűkombinációk (alapesetben az CTRL-ALT-DEL) aktivizálnak. A beléptetés során a felhasználói azonosítót (username) és a jelszót (password) a LOGON olyan módon autentikálja, hogy először átadja azokat ellenőrzésre egy másik önálló folyamat, az LSASS (Local Security Authentication Server / Helyi biztonsági jogosultság ellenőrző) részére, melyet az lsass.exe indít. Ha az autentikáció sikeres, akkor a LOGON elindítja a userinit.exe programot. A userinit.exe a beléptetendő felhasználó adatainak ismeretében már képes beállítani a felhasználóhoz definiált környezetet. A környezet beállítása után a userinit.exe elindítja a már beléptetett felhasználóhoz rendelt Shell keretprogramot (alapesetben ez az explorer.exe).
- SERVICECONTROLLER
  A SERVICECONTROLLER egy kernelből indított folyamat, melyet az screg.exe aktivál, és amely a szolgáltatások automatikus indításáért és leállításáért felelős.

Egyszóval a rendszerfolyamatok felelősek azért, hogy miután elindult az operációs rendszer a felhasználók be tudjanak lépni, és el tudják kezdeni az erőforrások hatékony használatát.

Szolgáltatások

A szolgáltatásként hivatkozunk a Windows NT alapú operációs rendszerekben azokra a folyamatokra, melyek a kliens programok számára többletszolgáltatásokat nyújtanak. A feladatok összetettségéből következik, hogy jellemzően a szolgáltatásokat több folyamat együttesen szolgálja ki. A többletszolgáltatásokat igénylő kliens programok – melyek lehetnek akár felhasználói vagy akár rendszer programok – így szerver-kliens viszonyba kerülnek a szolgáltatásokkal, azaz az azokat megvalósító folyamatokkal.

A Windows NT felépítésének komponensei című ábrán szerepel az RPC (Remote Procedure Call / Távoli eljáráshívás) szolgáltatás, a különböző protokollokat megvalósító hálózati kapcsolatot biztosító szolgáltatás (TCP/IP Helper), illetve az operációs rendszer eseményeit naplózó (Event Logger / Eseménynapló) szolgáltatás. A Windows NT alapú operációs rendszerek természetesen számos további szolgáltatást is tartalmaznak.

Hasonlóan, mint a rendszerfolyamatok, a szolgáltatások is user módban futnak. A hasonlóság azonban itt véget is ér, hiszen a rendszerfolyamtok nélkül a Windows NT alapú operációs rendszerek nem képesek működni, addig a szolgáltatások (azaz az a szolgáltatásokat jelentő folyamatok) nélkül a rendszer – ha korlátozottan is – de működőképes marad. Ebből következik, hogy a szolgáltatásokat, a pillanatnyi igényeknek megfelelően a rendszer futása közben leállíthatjuk, illetve elindíthatjuk a Service Manager használatával. Lehetőség van az egyes szolgáltatások automatikus-, késleltetett- vagy kézi indítására, szüneteltetésére, vagy leállítására is. Az automatikus – azaz felhasználói beavatkozás nélküli – indítás azon programok esetében elengedhetetlen, melyeknek egy bejelentkezett felhasználó hiányában is futniuk kell. Ilyen például a távoli asztal (Terminal Services) vagy a fájlmegosztás (Server), melyek a számítógép távolból történő elérését teszik lehetővé.

A szolgáltatásokat a Service Contoller (SERVICES.MSC) regisztrálja és felügyeli, ennek segítségével tudja a felhasználó a kiválasztott szolgáltatást paraméterezni.

Alrendszerek

A Windows NT alapú operációs rendszerek korai változatai a különböző típusú applikációk, programok futtatását a megfelelő alrendszerek segítségével biztosították. Az idők folyamán – üzletpolitikai-, marketing- és korszerűségi okok miatt – a valamikori három alrendszerből (Win32, POSIX, és az OS/2) mára csupán a Win32 maradt meg, így közvetlenül már csak Windows-os alkalmazások futtathatók. A Windows 2000 óta kikerült az OS/2 alrendszer, a Windows XP óta pedig a POSIX alrendszer is. A Win32 alrendszer nemcsak hogy megmaradt, hanem a rendszer elválaszthatatlan része, nélküle az operációs rendszer nem is képes futni. A Win32 alrendszer egyes részei kernel módban futnak, például a grafikus megjelenésért felelős (Full Desktop Multi-User Win32 Driver / WIN32K.SYS) is, a USER32.DLL, KERNEL32.DLL, ADVAPI.DLL könyvtárak támogatásával. A fő ok a rendszer hatékonysága, ugyanis ezzel a megoldással módváltás nélkül oldható meg az executive, illetve kernel réteg által biztosított szolgáltatások függvényeinek az elérése. A WIN32 API-ban számos további grafikus funkció is definiálva van. A GDI (Graphical Device Interface / Grafikus eszközcsatoló) szabvány szerinti grafikus eszközök és nyomtatók kezelését a GDI32.DLL (WIN32 API része) biztosítja. Az itt definiált funkciók is a WIN32 alrendszer kernel módú részében (WIN32K.SYS) valósulnak meg.

A Win32 API hívások háromfélék lehetnek a megvalósításuk helye szerint:

- Közvetlenül az alrendszer DLL-ben megvalósított egyszerű függvények hívása, melyek végrehajtásához a rendszer más részeinek elérésére nincs is szükség.
- Magában az alrendszerben megvalósított függvényhívások, melyek hívását az alrendszer DLL továbbítja az NTDLL.DLL felé, ami az executive réteg LPC szolgáltatását igénybe véve, eljut a Win32 alrendszerhez, ami a kérést teljesíti.
- Amennyiben a rendszer egy másik, kernel módban futó rétege valósítja meg a hívást, az akkor is az executive réteghez kerül, ami továbbítja a megfelelő kernel rétegnek.

6. 32-bit vs 64-bit

   32-bites operációs rendszer esetén a Win32 alrendszer a 32-bites, a 16-bites és DOS alkalmazásokat is futtatja. 64-bites operációs rendszer esetében a visszafelé kompatibilitás a 16-bites, illetve a DOS alkalmazások estében megszűnt, azaz ezek közvetlenül nem futtathatók. Megfelelő emulátor szoftver (pl. DosBox) segítségével természetesen orvosolható ez az állapot.

   A 64-bites Windows NT alapú operációs rendszerek esetében a 64-bites programok futnak natív módban, azaz a 32-bites programok futtatásához van szükség egy további alrendszer, a WoW64 (Windows 32-bit on Windows 64-bit) igénybe vételére. A WoW64 alrendszer a funkcióit a következő ábra szerint, további DLL-ek igénybe vételével valósítja meg.