Operációs rendszerek

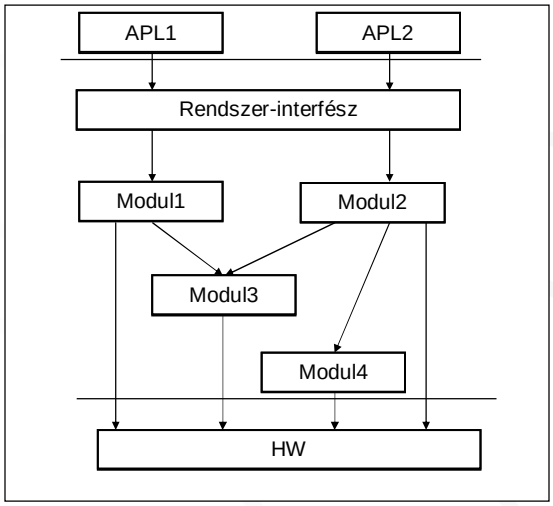
Windows NT

A Windows NT

- Felépítésében is új operációs rendszer: *New Technology* (*NT*)
- 32-bites Windows-os rendszerek felváltása
- Windows 2000: *NT alapú*

Operációs rendszerek felépítése

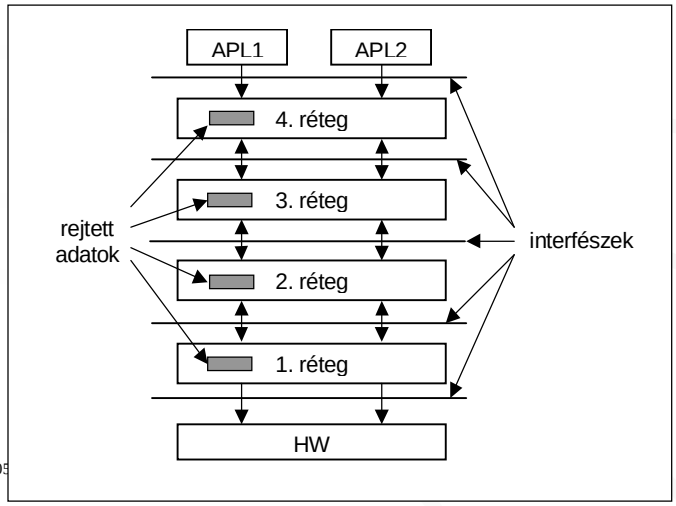
- Monolitikus kernel



Monolitikus kernel

- Közvetlen kommunikáció modulok között.
- ”Közösen" használt adatok.
- Nincs szabványos érintkezési felülete (interfésze) a moduloknak.
- Nem megbízható:
  egy hibás modul hibája továbbterjedhet a többi modul felé.
- Nehezen karbantartható, fejleszthető.

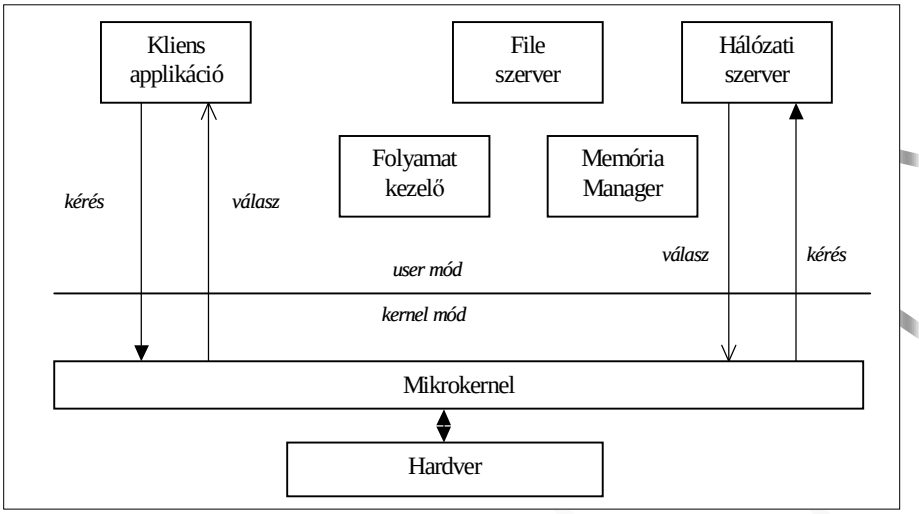
Réteg szerkezetű operációs rendszer I.



Réteg szerkezetű operációs rendszer II.

- A modulok rétegszerűen épülnek egymásra (objektum-orientált megközelítés).
- A kommunikáció jól definiált interfészeken keresztül bonyolódik.
- A rétegek csak a felettük illetve alattuk levő réteget érik el közvetlenül.
- Az interfészek használata:
  - *biztonságos: ellenőrizhető a kérés/hozzáférés jogossága,*
  - *megbízható: nem terjed a hiba,*
  - *karbantartható: fejlesztés, módosítás egyszerű.*

Kliens-szerver modellre épülő operációs rendszer I.



Kliens-szerver modellre épülő operációs rendszer II.

- Önálló, jól definiált funkciókat ellátó komponensek.
- Egymásra épülő szolgáltatások nyújtása, ill. azok használata.
- Mag az ún. *mikrokernel* (mikrokernel-es operációs rendszerek).
- Funkciója:
  - *Szolgáltatás kéréseket üzenetek formájában továbbítja az operációs rendszer moduljai számára.*
  - *Biztosítja a folyamatok számára a hardver elérését. (Hardver elérés csak mikrokernelen keresztül.)*

Kliens-szerver modellre épülő operációs rendszer III.

- A modulok önálló szerver folyamatok.
- Mikrokernel: privilegizált (kernel) mód.
- Folyamatok: felhasználói mód.
- Előny: rugalmas, biztonságos működés.
- Hátrány: nem hatékony hardver kihasználás (környezetváltások!).

  Pl.: *Mach* operációs rendszer.
  UNIX daemon folyamatok

Az NT felépítésének fő jellemzői

- Réteg szerkezet.
- Kliens-szerver működés (alrendszerek, szolgáltatások stb.).
- Szigorú interfész használat.
- Hatékony hardver kihasználás: user módban megvalósítható funkciók kernel módban vannak.

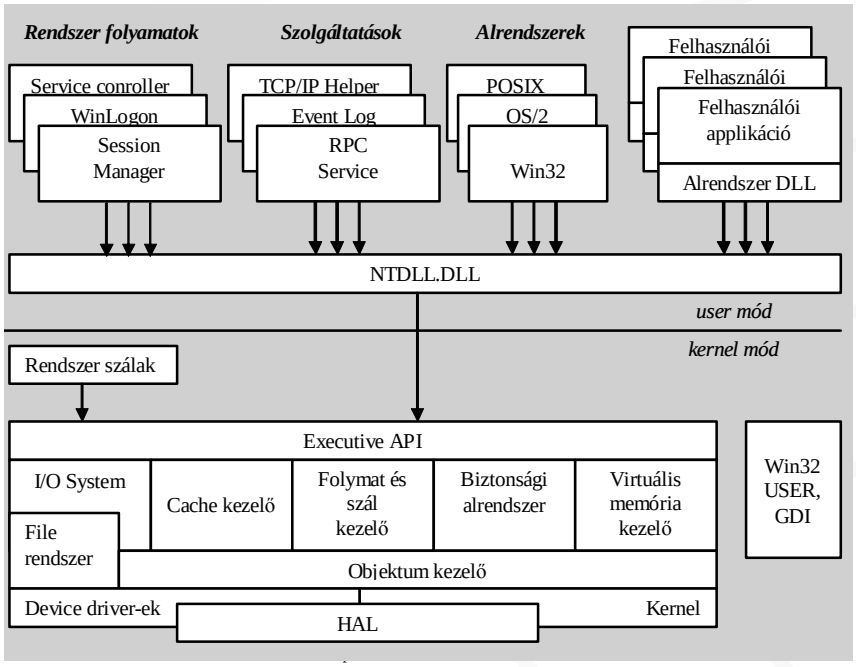
Az NT objektum-orientált szemlélete I.

- *adatrejtés*:
  Az operációs rendszer objektumai csak saját adataikat érhetik el.
- *interfész-használat*:
  Az objektumok formális interfészt használnak egymás elérésére.
- *hierarchikus objektum szerkezet*:
  kernel objektumok, executive objektumok.

Az NT objektum-orientált szemlélete II.

- A Windows NT ***nem*** valósítja meg a következő objektum-orientált tulajdonságokat:
  - *polimorfizmus*:
    Azonos néven különböző objektumok elérése.
  - *öröklődés*:
    Az objektumoknak hierarchikus származási rendszere.
  - *dinamikus adattípus-kötés*:
    Definiálhatóak adattípussal paraméterezett objektumok.

A Windows NT felépítés



HAL (Hardware Abstraction Layer)

- A hardvert teljesen elfedő legalacsonyabb szoftver réteg.
- Az aktuális hardverre támaszkodva egy *virtuális gépet* valósít meg.
- Hardver elérés csak ezen keresztül.
- Processzortól független szolgáltatásokat ad a többi rétegnek.

HAL II.

- Mindegyik processzorhoz külön HAL-réteg.
- Rendszer a telepítésekor választ HAL-t. (NT CD \i386–os alkönyvtára).
- Azonos architektúrájú processzorok esetén (pl. az x86 típusú processzorokon futó NT-k között) csak a HAL-ban van különbség.

Kernel

- Állandóan memóriában levő kernel (védett) módban futó komponens.
- A kernel a HAL-on kívül az egyetlen hardver-függő rész:
  - HAL a processzor típusától függ,
  - a kernel a processzor architektúrájától.
- A kernel által a nyújtott interface hardver független:
  - **à** a többi komponens hardver független.

NT hordozhatósága

- Hordozható operációs rendszer megvalósítása:
  - Hardver független interfésszel rendelkező szoftver rétegek:
    - A HAL, ill. Kernel feletti rétegek.
  - Hordozható programnyelv használata:
    - C, C++.

A Kernel funkciója

- Jól definiált működésű *alapmodulokat* (ún. *primitíveket*) biztosít az NT többi tagja számára.
- A primitívek a következő funkciókat valósítják meg:
  - *thread (szál) ütemezés, környezetváltás,*
  - *kivételkezelés, IT-kezelés,*
  - *trap-kezelés, multiprocesszor-ütemezés,*
  - *kernel objektumok kezelése.*

Kernel objektumok

- A kernel objektumok egyszerűek.
- Cél:
  - *gyors kezelés.*
- Megvalósítás:
  - *Nem végez ellenőrzést az egyes objektumok elérésekor, megbízik abban, hogy a rendszer tagjai helyesen használják az objektumokat.*

Device driver-ek (készülékmeghajtók)

- Az I/O alrendszer és a hardver közötti kapcsolat.
- Hardver elérés: HAL-rétegen keresztül. ("Hordozható" device driver-ek.)
- A device driver-ek négy típusát különböztetjük meg:
  - Hardver driver-ek
  - Fájlrendszer driver-ek
  - Szűrő típusú device driver-ek
  - Hálózat elérését biztosító device driver-ek

Device driver-ek típusai I.

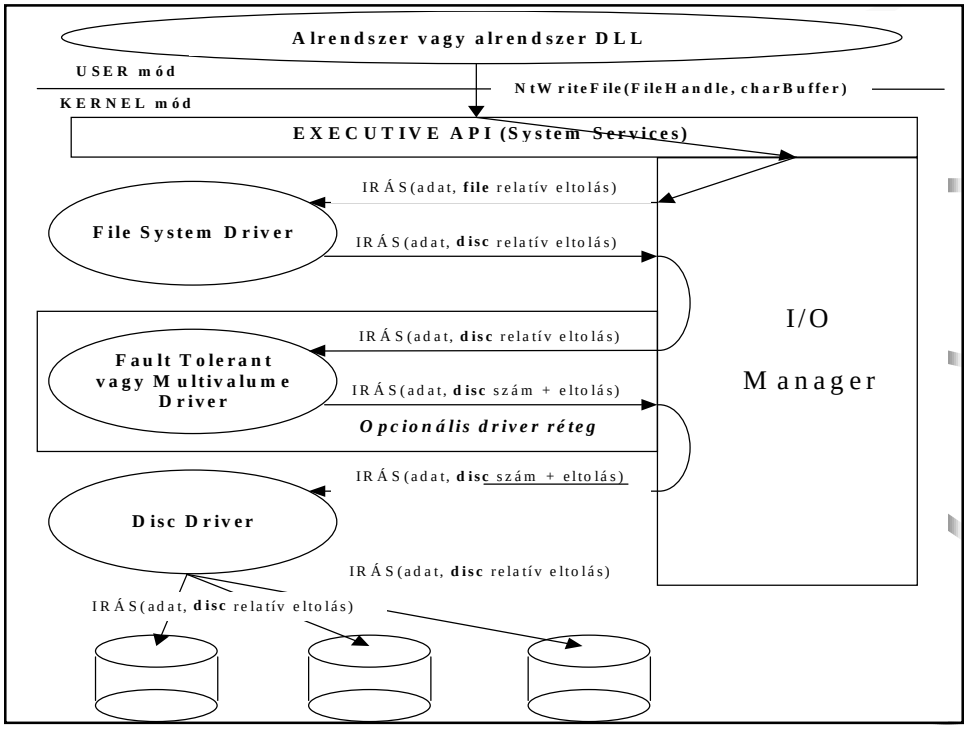
- Hardver driver-ek (alacsony szintű):
  - hardver egységek elérése, eszközök közvetlen be- és kimeneteit kezelik.
- Fájlrendszer driver-ek: (magas szintű)
  - Fájlrendszerek elérését biztosító kéréseket fogadnak el és I/O kérésekké transzformálják azokat.

Device driver-ek típusai II.

- Szűrő típusú device driver-ek:
  - Többletfunkciókat megvalósítása (réteg szerkezetű device driver struktúra).A magasszintű driver-ek és alacsony szintű driver-ek közé ékelődnek.
- Hálózat elérését biztosító device driver-ek:
  - hálózati kéréseket szolgálják ki, ill. továbbítják azokat.

Réteg szerkezetű device driver struktúra

- Különböző funkcionalitású driver-ek rugalmas használata.
- Igényeknek megfelelő rendszer építése.
- Különböző driver szintek:
  - A szintek funkcionalitása, ill. az adott funkcionalitáshoz tartozó interface jól definiált.
- Példa: Lemez írása.



Executive

Megvalósított függvények:

- elérhető alrendszerek hívások
- nem dokumentált alrendszerek hívások
- dokumenetált driver hívások
- nem dokumenetált belső (kernel, executive) hívások
- belső hívások

Executive önálló részei

- *Folyamat és szál kezelő*
  - *létrehozás, kezelés, leállítás*
- *Virtuális memória kezelő*
  - *memória gazdálkodás, cache kezelés támogatása*
- *Biztonsági alrendszer (monitor)*
  - *lokális gép objektum védelme*
- *Cache kezelő*
  - *file kezelő I/O gyorsítása*
- *I/O rendszer kezelő*
  - *eszköz független I/O*

Funkciói I.

- Az NT belső objektumai közötti kommunikáció biztosítása.
  - *LPC (Local Procedure Call, lokális eljárás hívás)*
    - *NT IPC (Inter Process Communication) eszköze.*
    - *Egy user objektum egy másik user objektum függvényét hívhatja meg.*
    - *PL.: Így érhetik el pl. az egyes alkalmazások a hozzájuk tartozó alrendszer szolgáltatásait.*

Funkciói II.

- Az NTDLL.DLL által definiált függvények hívásainak megvalósítása.
- A *run-time library* függvények megvalósítása.
- Rendszer processzek és a szolgáltatások számára támogató funkciókat megvalósító függvények.

Rendszer processzek (rendszer folyamatok)

- Operációs rendszer funkciókat megvalósító önálló folyamatok.
- Felhasználói módban futó folyamatok.
- Rendszer processzek szükséges részei a rendszernek.

Fontosabb rendszer folyamatok I.

- *SMSS* (*Session Manager*):

  A rendszer indításakor létrehozott folyamat, ami az applikációk elindításáért felelős.

  Feladatai:

  - *egyes alrendszerek elindítása, ha futásukra szükség van,*
  - *kapcsolat biztosítása a debugger és az általa futatott applikáció között*
  - *biztosítja a környezeti változók definiálásának és elérésének lehetőségét*

Fontosabb rendszer folyamatok II.

- *Logon*:
  - A felhasználók ki és beléptetését intéző folyamat.
  - Minden ún. *SAS* (*Secure Attention Sequence*) billentyű kombináció (alapesetben: CTRL-ALT-DEL) aktivizálja.
  - Beléptetéskor a user name és password kombinációt az önálló folyamatként futó *Local Security Authentication Server*-hez (*LSASS*) továbbítva ellenőrzi.
  - Ha az azonosítás sikeres, elindítja a *USERINIT.EXE* programot, ami beállítja a felhasználó által definiált környezetet, és elindítja a user által kért shell-t. (alapesetben: *EXPLORER.EXE*)

Fontosabb rendszer folyamatok III.

- *Service Controller*:
  - A szolgáltatások indításáért és leállításáért felelős folyamat.
  - user interface:
    - szolgáltatás állapota
    - szolgáltatás beállításai
    - szolgáltatás indítása/leállítása

Szolgáltatások

- Szolgáltató folyamatok:
  - kliens-szerver működés
  - szolgáltató folyamatok pl.:
    - RPC (Remote Procedure Call) szolgáltató
    - NT specifikus eseménynaplózó (event logger) szolgáltató.
    - …

Szolgáltatások tulajdonságai

- Felhasználói módban futó folyamatok.
- A szolgáltatásokat megvalósító folyamatok futása nélkül is képes működni az NT.
- A szolgáltatások a Service Manager segítségével elindíthatók és leállíthatók a rendszer működése közben.
- Egyszerű Win32-es alkalmazások, de együttműködnek a *Service Conroller* (*SERVICES.EXE*) folyamattal (regisztrál, indít leállít).
  - Elérés: *Control Panel* **à** *Services* ikonja

A szolgáltatások három elnevezése

- ahogy a szolgáltatás a Control Panel Services alpontján keresztül elérhető,
- ahogy Registry-ben szerepel,
- a szolgáltatást megvalósító futtatható program neve.

NTDLL.DLL

- Dinamikusan linkelhető könyvtár (*Dinamically Linked Library*, *-DLL*).
  - Interfész a felhasználói módú folyamatok számára
  - Minden felhasználói objektum az NTDLL.DLL-en keresztül éri el a környezetét (pl. LPC).
- Működés:
  - Hívási paraméterek ellenőrzése.
  - User–kernel módváltás.
  - NT kért függvényének meghívása.

Alrendszerek I.

- Applikációk futtatása: alrendszerek segítségével:
  - Win32 (szükséges),
  - POSIX (opcionális),
  - OS/2 (opcionális).
- Alrendszerek feladata:
  - Alkalmazások futtásához szükséges szolgáltatások nyújtsa.
  - Alkalmazások kontrollálása.

Alrendszerek II.

- Az alkalmazások és az NT interfésze: **alrendszer DLL** (programozói interface (*API*, *Application Programming Interface*).

Alrendszerek programozói interfésze

- Jól definiált publikus interface; a többi interface (pl. NTDLL.DLL) nem publikus, nem dokumentált.
- Az egyes alrendszerekhez tartozó API-k lényegesen különböznek egymástól. (A legszélesebb a Win32 API.)
- API funkciók: pl. ablakozás, szálak kezelése.
- Az applikáció csak egy alrendszerhez tartozhatnak, nem keveredhetnek egymással egy applikáción belül különböző alrendszerekhez tartozó hívások.

POSIX alrendszer I.

- Szigorúan a POSIX 1003.1-es szabványban rögzített tulajdonságokat valósítja meg.
- Lehetőség van:
  - fork-olásra,
  - hard link-ek definiálására,
  - interprocess kommunikációra,
  - folyamatok kezelésére, és
  - karakteres I/O kezelésére.

POSIX alrendszer II.

- **Nincs** lehetőség
  - szálak (thread-ek) létrehozására,
  - ablakkezelésre,
  - RPC-elérésére,
  - socket-ek használatára.

Win32 alrendszer I.

- A Win32 alrendszer:
  - 32-bites alkalmazások,
  - 16-bites és
  - DOS-os alkalmazások futtatása.
- Nélküle nem futhat az NT.

Win32 alrendszer II.

Az alrendszer egy része kernel módban fut. (*WIN32K.SYS*):

- Grafikus képernyő-kezelési funkciók. (A *USER32.DLL*, *GDI.DLL*, *KERNEL32.DLL*, *ADVAPI.DLL* könyvtárakban definiált funkciók.)
- Rendszer gyorsítása miatt kerültek ide, módváltás nélkül lehetséges az executive, ill. kernel-szolgáltatások (függvények) elérése.

Win32 alrendszer III.

Grafikus eszközözök és a printer kezelése szabványos felületen történik:

- A *GDI* (*Graphical Device Interface*), ill. a hozzá tartozó *GDI32.DLL* (WIN32 API része) teszi lehetővé. Ezek is a WIN32 alrendszer kernel módú részében (*WIN32K.SYS*) vannak megvalósítva

Win32 API-hívások megvalósítása I.

- Megvalósításuk helye szerint:
  - Az alrendszer DLL-ben: Egyszerű funkcionalitású függvények, a végrehajtásukhoz nincs szükség a rendszer más részeinek elérésére.

Win32 API-hívások megvalósítása II.

– Az alrendszerben:
  A hívást az alrendszer DLL továbbítja az *NTDLL.DLL* felé, ami az Executive réteg LPC szolgáltatását igénybe véve, eljut a Win32 alrendszerhez, ami a kérést teljesíti.
– Más (kernel módban futó) rétegben:
  A hívás az Executive réteghez kerül, ami továbbítja a megfelelő kernel komponens felé.

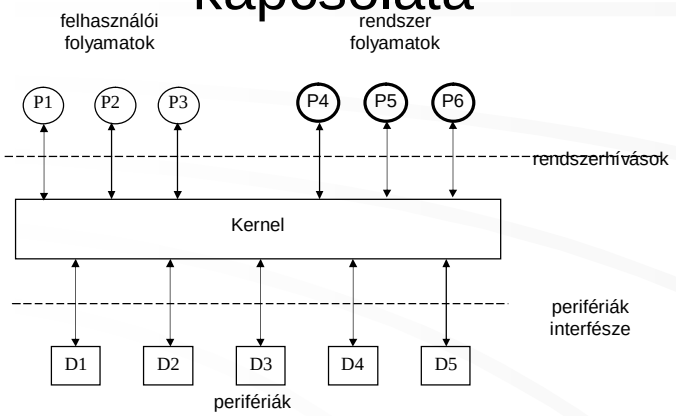
Operációs rendszerek

Folyamatok kezelése a UNIX-ban

Folyamatok a UNIX-ban

- Folyamat: multiprogramozott operációs rendszer alapfogalma - absztrakt fogalom.
- Gyakorlati kép: végrehajtás alatt álló program.
- Folyamat *virtuálisan* egyedül fut a rendelkezésre álló hardveren.
- Hardver erőforrások elérése:
  - operációs rendszeren keresztül.
- Műveletek:
  - erőforrás igénylés, várakozás.
- Operációs rendszer feladata:
  - *összehangolja az egyes folyamatok perifériás műveleteit,*
  - *elosztja a rendelkezésre álló hardver erőforrásokat.*

Folyamatok és a rendszer kapcsolata



Interfészek

OR interfészek:

- Programozói interfész (rendszerhívások):
  - Folyamat **à** OR
- Periféria interfész:
  - OR **à** I/O eszközök (perifériák)

Folyamatok párhuzamos végrehajtása

- egy fizikai processzor
- virtuális párhuzamos végrehajtás
- CPU ütemezés:
  gyors váltás, futó folyamatok számára szinte észrevétlen
- UNIX:
  időszeletet (*time slice* vagy *quantum*), prioritásos, preemptív (kiszorításos)

Folyamatok környezete

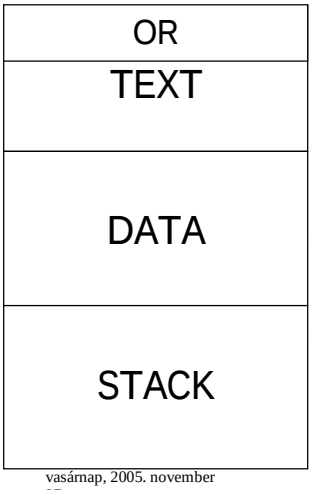
- Követelmény:
  - folyamatok megszakítása, ill. újraindítása
- Folyamatok környezete:
  a folyamat és az őt végrehajtó gép állapota (ill. annak leírása).
  - programszámláló (program counter)
  - regiszterek,
  - Memory Management Unit
  - használt erőforrások stb.

Folyamatok memóriája a UNIX-ban

Folyamatok memóriája

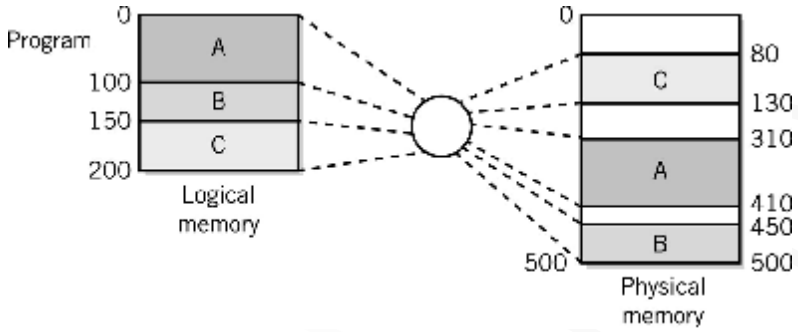
- Egyszerre több folyamat a fizikai memóriában.
- Fogalmak:
  - logikai memóriakép
  - fizikai memóriakép
  - virtuális memóriakezelés/tárcsere
- *OR feladata:*
  - memória cím–memória tartalom összerendelés
  - memória tartalom fizikai memóriába mozgatása

Memória használat

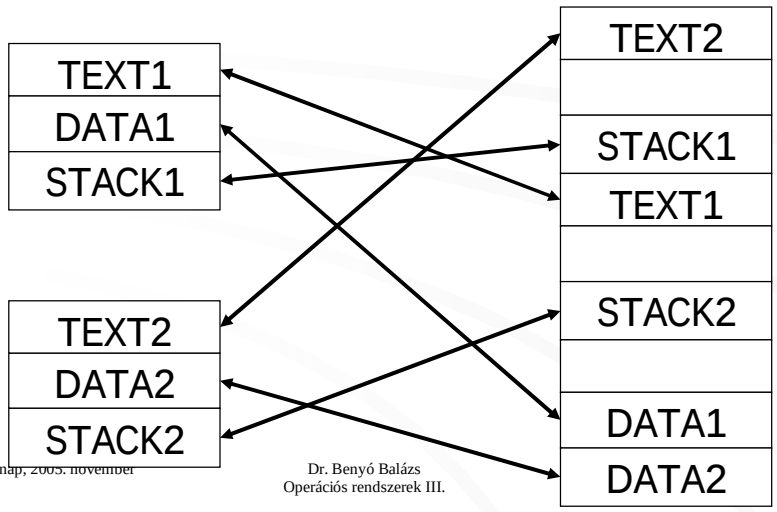


- program kód:
  - statikus
  - dinamikus (DLL)
- adatterületek:
  - inicializált (DATA)
  - nem inicializált (BSS)
  - heap (dinamikus adatok)
- veremtár:
  - dinamikusan növelhető terület

Logikai-fizikai címtranszformáció



Logikai-fizikai címtranszformáció



UNIX memória kezelés

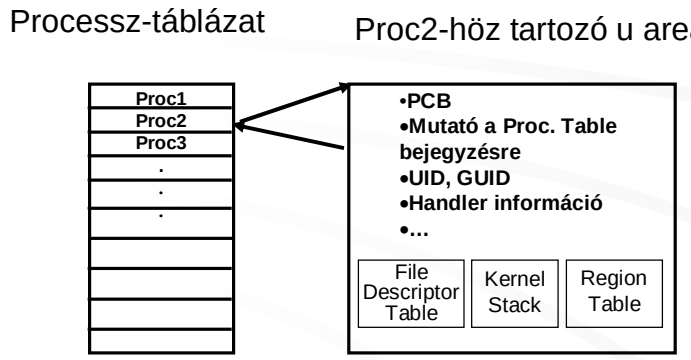
- Logikai – fizikai címtranszformáció:
  - folyamat leíró adatkban:
    - Region Table
    - Memory Management Unit

A folyamatok környezetének adatelemei UNIX-ban

A folyamatok környezetének adatelemei

- Minden adat, ami szükséges a folyamat újraindításához.
- Két adatstruktúrában tárolódik:
  - *u area (user area, felhasználói terület) processz memóriájában*
  - *proc (processz**)** struktúra rendszer memóriájában*
    - *Tárolás: hash tábla*

Folyamat környezet leíró adatok



A proc struktúra elemei I.

- processz azonosító (PID)
- user area-ra mutató referencia
- processz állapota:
  - kernel vagy user módban fut, várakozó vagy futásra kész, stb. (l. állapot diagramm)
- ütemezési információ
  - *p_cpu*, *p_usrpri*,…(l. ütemezés)
- ütemezésnél használt egyéb mutatók (pl. milyen prioritási kategóriában van)

A proc struktúra elemei II.

- signalok kezelésére vonatkozó információ:
  - *ignored-blocked-posted-handled* (l. signalok)
- MMU információ
  - milyen MMU regisztereket kell frissíteni a folyamat futtatásakor)
- a processzek különböző listáihoz aktív, zombie, szabad) tartozó linkek (l. állapot diagram)
- jelzőbitek

A proc struktúra elemei III.

- a PID alapján *hash táblában* történő tároláshoz szükséges mutatók
- a processzek hierarchiájában elfoglalt helyre vonatkozó információ:
  - processz szülője, processz gyermekei, stb.

A u area elemei I.

Egyszerű adatelemek:

- *PCB (hardver környezet, regiszterek)*
- *mutató a folyamat proc struktúrájára*
- *jogosítványok (valós és hatásos UID, GID)*
- *signal handler (kezelő) rutinok címei*
- *CPU használati statisztika (ütemezéshez)*
- *környezeti változók (aktuális könyvtár, kontroll terminál, keresési út, stb.)*

A u area elemei II.

Összetett adatelemek:

- *a folyamathoz tartozó memória (a felhasználói címtartomány) elérésére szolgáló táblázat (címtranszformációs tábla,* per-process Region Table*)*
- *fájl-leírók táblázata (per process File Descriptor Table)*
- *kernel módban használt veremtár (*per process Kernel Stack*)*

Összetett adatelemek

- Kernel Stack:
  - verem tár
  - kernel (privilégizált) módban futáskor
  - (rendszerhívások, IT stb.)
- Region Table:
  - MMU címtranszformációs táblázat
- File Descriptor Table:
  - A folyamat által megnyitott fájlok eléréséhez szükséges információ tárolása.

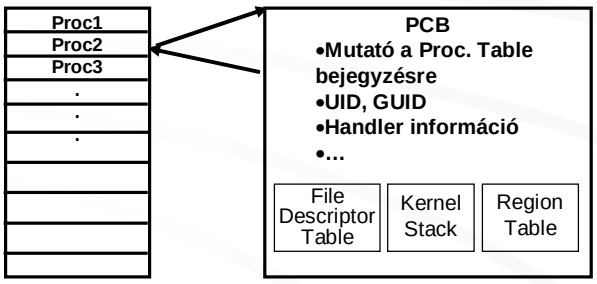
A hardver környezet (PCB)

- CPU regiszterei
- utasításszámláló (PC)
- stack pointer (SP)
- processzor-állapotleíró bitek (process status word), pl. carry (átvitel) bit
- memóriakezelő egység regiszterei (Memory Management Units, MMU)
- Lebegőpontos egység (Floating Point Units) regiszterei

Folyamat környezet leíró adatok

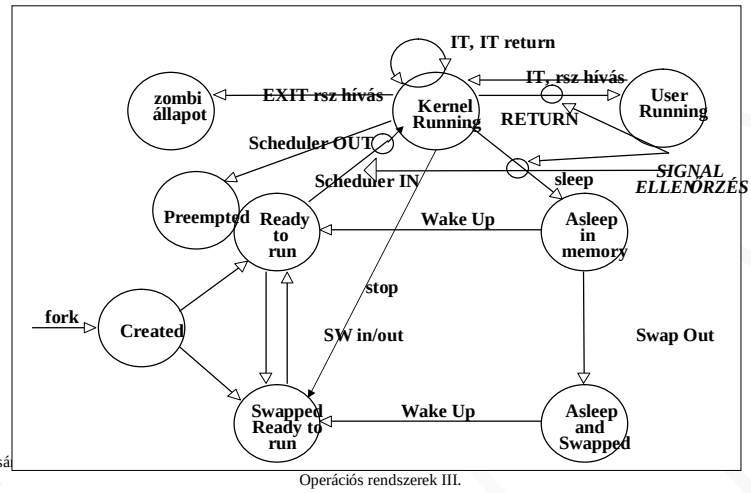
Processz-táblázat

Proc2-höz tartozó u area



Folyamatok állapot-átmeneti diagramja

Folyamatok állapot-átmeneti diagramja



Folyamatok állapot-átmeneti diagramja II.

Új folyamat keletkezése:

- *fork()* rendszerhívás
  - *gyermek folyamat*
    - *fork() visszatérési értéke: 0*
  - *szülő folyamatnak*
    - *fork() visszatérési értéke: gyerek PID-je*
- *Ősfolyamat: init (PID=0)*
- *scheduler (ütemező, PID=1)*

Folyamat indulása

```
int result=fork();
if(result==0)/*visszatérési érték vizsgálata*/
{
  printf("GYERMEK VAGYOK");
  exec("gyermek_kód",...);
  /*a gyermek végrehajtása*/
}
else
{
  if(result<0)
  {
    printf("HIBA");
  }
  else
  {
     printf("Anya proc:gyermekem PID-je:%d",result);
     /*ha a visszatérési érték >0, kiírja */
  }
}
```

Folyamatok állapotai I.

- *created (létrehozott) állapot*
- *ready to run (futásra kész) állapot*
- *kernel running*
- *user running*
- *asleep in memory (memóriában alvó)*
  - sleep *rendszerhívás*
  - *wake up átmenet*

Folyamatok állapotai II.

- *swapped out*
  - *swapped out and asleep*
  - *swapped out and ready to run*

Preemptív ütemezés:

- *kernel running* **à** *ready to run*
- *preemptív (erőszakosan megszakított, kiszorított) állapot*

Futás befejezése: zombie állapot

A UNIX OR feladatainak megvalósítása

Az OR feladatai

- A UNIX, mint operációs rendszer a feladatait a következő négy formában látja el:
- Kernel:
  - *rendszerhívások (system call*)
  - *kivétel* (*exception*)
  - *hardver megszakítások (hardware interrupt)*
  - *(call-out függvények)*
- Önálló folyamat:
  - *daemon folyamatok*

Rendszerhívások

- UNIX programozói interfésze. (*API, application programming interface)*
- Operációsrendszer szolgáltatásainak elérése:
  - erőforrás foglalás,
  - rendszerinformációk gyűjtése, elszámolás biztosítása,
  - védelmi és biztonsági mechanizmusok biztosítása.
- Privilegizált futási mód.
- Rendszer és folyamat erőforrásainak használata.

Kivétel (exception)

- Pl. nullával történő osztás, memóriaterületről történő kicímzés, stb.
- Utasítás végrehajtással szinkron események.
- Kivételek kezelése kernel feladat.

Hardver megszakítások (hardware interrupt)

- Perifériák (I/O egység) állapotának megváltozása.
- IT kezelés kernel feladat (csak rendszer erőforrásokkal!).
- Aszinkron módon fellépő eseményekre a rendszer reagálni tudjon
  - interrupt rutin lefuttatása,
  - nincs *környezetváltást* (*context switch*).

Az interrupt rutin által végzett tipikus műveletek

1. (végrehajtási mód váltása: user→ kernel)
2. Elmenti az aktuális processzor környezetet.
3. IT kiváltó esemény azonosítása.
4. IT kiszolgáló rutin meghatározása.
5. IT rutin hívása.
6. Környezet visszaállítása.
7. (végrehajtási mód váltás: kernel → user)

Daemon folyamatok

- Önálló folyamatokként megvalósított OR funkciók.
- Nem privilegizált módban futnak.
- Rendszerhívásokon keresztül érik el a kernel szolgáltatásait.

Folyamatok futási környezete I.

- Folyamatok logikai memóriaképe:
  - operációs rendszer által használt memória,
  - folyamat által használt memória.
- Memória védelme:
  - rendszer környezet (*system context*) és
  - felhasználói környezet (*user context*) megkülönböztetése Java VM (hordozható!), PC emulátorok

Folyamatok futási környezete II.

- Rendszer futási környezet:
  - operációs rendszer memóriaterületei,
  - kernel erőforrásai,
  - kernel stack
- Felhasználói futási környezet:
  - alkalmazás adatterületei,
  - user stack.

Folyamatok futási környezete III.

- A processzor különböző utasítás végrehajási módjai:
  - felhasználói (nem privilegizált) mód (user mode)
    - Néhány speciális utasítás végrehajtása és a memória adott részének elérése tiltott.
  - privilegizált, kernel mód (kernel mode).

Folyamatok futási környezete IV.

Környezetek érvényessége