Az Operációs rendszerek tárgy tervezett tanterve

ELTE Programtervező matematikus szak

Balogh Ádám, Lőrentey Károly, Nagy Tibor, Varga Balázs

2003. december 12.

A tárgy előfeltételei:

- Programozási környezet
- Számítógépek felépítése
- Elemi alkalmazások fejlesztése II.

1. Képesítési követelmények

A tárgyat elvégzett hallgatónak

- tisztában kell lennie az operációs rendszer szerepével és a tőle elvárható szolgáltatásokkal;
- értenie kell az operációs rendszerek működését, általános szerkezetét, ismernie kell az operációs rendszerekben használt legfontosabb módszereket, algoritmusokat;
- programfejlesztés során képesnek kell lennie hatékonyan alkalmazni az operációs rendszerek szolgáltatásait;
- minden feladathoz ki kell tudnia választani hozzá legmegfelelőbb operációs rendszert;
- képesnek kell lennie a konkrét operációs rendszerek műszaki dokumentációját a tárgy elvégzése során szerzett magasszintű összefüggések alapján hatékonyan feldolgozni, akár üzemeltetési, akár programozói feladatokat kell elvégeznie.

2. Az előadások tematikája

Az előadások tantervét az ACM/IEEE Computer Curricula 2001 figyelembevételével állítottuk össze. Az összesen nyolc témakör mindegyikével hozzávetőleg két-két előadáson keresztül kívánunk foglalkozni, a másodikat és a nyolcadikat kivéve, melyekre maximum egy-egy előadást szánunk.

Amikor csak lehetséges, a témakörök általános tárgyalása után néhány konkrét operációs rendszerben alkalmazott jellegzetes megoldás rövid bemutatásával illusztráljuk a tárgyalt problémák gyakorlatban is alkalmazott megoldásait.

A hallgatók várható nagy számára tekintettel a számonkérés előreláthatóan két fázisban, egy írásbeli és az azon túljutó hallgatók számára rendezett szóbeli vizsga meghirdetésével fog történni.

1. **Operációs rendszerek alapjai** (Balogh Ádám)
   - Az operációs rendszer céljai és feladatai
   - Az operációs rendszerek történetének áttekintése
   - Egy modern operációs rendszertől elvárt szolgáltatások
   - Tervezési célkitűzések (hatékonyság, megbízhatóság, hibatűrés, rugalmasság, hordozhatóság, biztonság, kompatibilitás), ezek ellentmondásossága
   - A védelmi feladatok, a hálózati és multimédia elvárások, a grafikus felhasználói felület hatásai az operációs rendszerek fejlődésére

- Tervezési alapelvek (monolitikus, rétegzett, moduláris, mikrokernel operációs rendszerek, alacsonyszintű párhuzamos virtuális gépek (IBM S/390), absztrakt virtuális gép (AS/400, .NET))
- Absztrakciók: folyamatok és erőforrások
- A felhasználói- és rendszerállapot és -védelem elve, átmenet felhasználói üzemmódból rendszerüzemmódba
- Beágyazott- és valós idejű rendszerek, osztott operációs rendszerek

2. **Alkalmazásfejlesztő felületek (API)** (Balogh Ádám)

   - Procedurális API-k, objektum-orientált API-k
   - Horizontális, vertikális jellegű API-k
   - Rétegezett, modularizált API-k
   - Middleware

3. **Párhuzamosság** (Lőrentey Károly)

   - Látszatpárhuzamosság, valódi párhuzamosság
   - A kölcsönös kizárás problémája
   - Holtpontok: okok, feltételek, megelőzés
   - A párhuzamosság kezelésének segédeszközei, folyamatok közötti kommunikáció (szemaforok, monitorok, állapotváltozók, randevú, üzenetküldés)
   - Gyártó-fogyasztó problémák és szinkronizáció
   - Többprocesszoros rendszerek
   - Példák: konkrét operációs rendszerek jellegzetes megoldásai

4. **Folyamatok, ütemezés** (Balogh Ádám)

   - Folyamatok, szálak, folyamat-hierarchiák
   - Folyamatállapotok, folyamattábla
   - Preemptív és kooperatív ütemezés
   - Ütemezők és ütemezési módszerek (round robin, prioritásos, garantált, sorsjáték, stb.)
   - Valósidejű operációs rendszerek problémái (határidők, garantált ütemezés)
   - Példák: konkrét operációs rendszerek jellegzetes megoldásai

5. **Memóriakezelés** (Balogh Ádám)

   - A fizikai memória és a memóriakezelő hardver áttekintése
   - Explicit rétegezés (overlayek), csere (swapping), partíciók
   - Lapozás (paging), szegmentálás
   - Lapkezelő algoritmusok (optimális, NRU, LRU, FIFO, stb.)
   - Gyorsítótárak működése és szerepe
   - Osztott memóriaterületek
   - Osztott programkönyvtárak, dinamikus kötés, programbetöltés, relokáció
   - Példák: konkrét operációs rendszerek jellegzetes megoldásai

6. **Fájlrendszerek** (Lőrentey Károly)

   - Fájlok: adat, metaadat, műveletek, pufferezés, soros és véletlen elérés
   - Könyvtárak: tartalom és szerkezet

- Fájlrendszerek: fel/lecsatolás, virtuális fájlrendszerek, naplózó (journaling) fájlrendszerek
- Fájlrendszerek implementációja, adatszervezési stratégiák
- Lemezkezelés, partíciók, RAID, logikai partícionálás
- Memória-leképezésű fájlok
- Speciális célú fájlrendszerek (pl. /proc, /sys a Linuxban)
- Példák: konkrét operációs rendszerek jellegzetes megoldásai

7. **Operációs rendszerek biztonsága és védelme** (Lőrentey Károly)

   - A rendszerbiztonság áttekintése, fontossága
   - Védelmi alapelvek (felhasználói azonosító, felhasználói csoportok, jogosultságok, jogosultság-delegáció)
   - A rendszer biztonsága és a felhasználó kényelme közötti ellentmondás
   - Autentikációs megoldások (jelszavak, kriptográfiai módszerek, központosított autentikáció)
   - Az autentikáció és autorizáció közti különbség
   - Memóriavédelem
   - Fájlvédelem
   - Kódolás, kriptográfia alkalmazása
   - Nyomonkövethetőség, naplózás
   - Példák: konkrét operációs rendszerek jellegzetes megoldásai

8. **Eszközök, perifériák programozása** (Lőrentey Károly)

   - Eszközök fajtái: karakter- és blokkalapú eszközök
   - Perifériák kezelése, szervezése
   - Az eszközök különbségeinek elfedése
   - Pufferezés, implementációs stratégiák
   - Megszakítások
   - Közvetlen memóriaelérésű eszközök
   - Hibakezelés
   - Példák: konkrét operációs rendszerek jellegzetes megoldásai

3. A gyakorlatok tematikája

A gyakorlatok célja (1) az előadáson elhangzott algoritmusok és adatszerkezetek elsajátításának segítése szemléltető programok készítésével, (2) a bemutatott rendszerszolgáltatások (pl. folyamatok közti kommunikációs eszközök) gyakorlati kipróbálása, illetve (3) egy konkrét operációs rendszer működésének közelebbi megismerése a rendszer a hallgatók által végrehajtott egyszerű módosításán keresztül.

A gyakorlatok GNU/Linux operációs rendszerrel felszerelt géptermeket igényelnek. Minden hallgatónak külön számítógépet kell biztosítanunk.

A gyakorlatok ideje alatt a gyakorlatvezetők irányításával kisebb programozási feladatokat oldanak meg a hallgatók. Minden gyakorlat elején kihirdetésre kerül az aznapi feladatsor, ezután a gyakorlatvezető részletes segítséget nyújt a megoldáshoz. A gyakorlat végére a hallgatók többsége a gyakorlatvezető útmutatásait követve el is tudja készíteni a feladatok megoldását. Az elkészült feladatokat a gyakorlatvezető regisztrálja. A gyakorlati jegy a beadott feladatok számának (és az

alább ismertetett szemléltető feladatnak) a függvénye. Az esetleg elmaradt feladatokat otthon kell megoldani, és a félév utolsó három óráján lehet bemutatni.

A gyakorlaton megoldandó feladatok mellett a gyakorlati jegybe beszámít egy a gyakorlatokon kívül, önálló munkával elkészítendő szimulációs program is, melynek témája az előadáson elhangzó valamelyik anyagrész bemutatása. A feladatokat a hallgatók a félév elején megkapják. Az értékelés a félév utolsó óráin történik, az elmaradt gyakorlati feladatok bemutatásával egy időben.

- 1. gyakorlat: Követelmények kihirdetése, házi feladat kiosztása
- 2–5. gyakorlat: Rendszerközeli programozás
- 6–10. gyakorlat: Kernelprogramozás
- 11–13. gyakorlat: Házi feladat és elmaradt programok bemutatása

Rendszerközeli programozás (4 gyakorlat)

Az operációs rendszerek tárgyalása nem lehet teljes az általuk nyújtott szolgáltatások közelebbi bemutatása nélkül. Az előadásokon számos, az operációs rendszer által biztosított eszközt bemutatunk, a gyakorlat második részében ezek közül kell néhányat a gyakorlatban is kipróbálniuk a hallgatóknak.

A harmadik résszel való összhang jegyében a gyakorlatokon a POSIX rendszerek C nyelvű alkalmazásfejlesztő felületét tárgyaljuk, de a hallgató a gyakorlatvezetővel történő egyeztetés után tetszőleges másik, a géptermekből elérhető operációs rendszert vagy programozási nyelvet is választhat.

- Folyamatkezelés (1. gyakorlat):
  1. Az aktuális folyamat lemásolása a `fork` rendszerhívással.
  2. A gyerekfolyamatban egy másik program elindítása (`exec`).
  3. A szülőfolyamatban várakozás a gyermek befejeződésére (`wait`).
  4. A gyerekfolyamat leállítása szignál küldésével.
  5. Szignálkezelés: a TERM szignál lekezelése a gyerekprocesszben.
- Folyamatok közötti kommunikáció (2–3. gyakorlat)
  1. Csövezetékek (pipe) létrehozása a gyerek–szülő közti kommunikációra.
  2. Ugyanez független folyamatokra, FIFO-val.
  3. Ugyanez Socket API-val
  4. System V: üzenetsor használata, üzenettípusok megkülönböztetésével
  5. System V: Osztott memória, szinkronizáció nélkül (termelő-fogyasztó probléma).
  6. System V: szemaforok, szinkronizációra.
- Fájlkezelés (4. gyakorlat)
  1. Fájlok metaadatainak lekérdezése (`stat`)
  2. Könyvtárak olvasása ('find -type f -print0 | xargs -0 ls -s' művelet implementációja
  3. Locking mechanizmusok (flock)
  4. Aszinkron be/kivitel (select)

Kernelprogramozás (5 gyakorlat)

Ebben a részben a hallgatók bepillantást nyernek egy valódi operációs rendszer belső működésébe: néhány egyszerű változtatást kell végrehajtaniuk a nyílt forrású Linux kernelen. A rész nem próbál teljeskörű áttekintést adni a kernel működéséről (ez már csak a rendelkezésre álló idő rövidsége miatt sem lenne kivitelezhető); a feladat inkább az operációs rendszer „demisztifikációja", annak bemutatása, hogy a kernel maga is csak egy program, melynek részei más programrendszerekhez hasonlóan egy átlagos programozó számára is megérthetők és szükség esetén javíthatók, módosíthatók.

Üzemeltetési (biztonsági és karbantartási) okokból a laboratóriumok gépeinek operációs rendszerét természetesen nem lenne szerencsés a hallgatók által írt változatokra cserélni; szerencsére lehetőség van a Linux kernel egy már futó kernelből történő rekurzív elindítására, mely ezeket a problémákat kiküszöböli. (User Mode Linux)

A Linux kernel komplexitása miatt szükség van arra, hogy a téma elején a gyakorlatvezető (kb. egy gyakorlat terjedelemben) bemutassa a kernel lefordításának menetét, a kész kernel kipróbálásának módját és áttekintést adjon a kernel forrásának szerkezetéről. Az anyagrész további ütemterve:

- 1. gyakorlat: Az User Mode Linux kernel lefordításának
- 2. gyakorlat: Kernelmodulkészítés, hello world kiírása a konzolra
- 3. gyakorlat: /proc/hello-world virtuális fájl létrehozása
- 4. gyakorlat: Karakteralapú eszköz készítése (/dev/hello-world-c)
- 5. gyakorlat: Blokkalapú eszköz készítése (/dev/hello-world-b)

Szemléltető programok (házi feladat)

A feladatot C++ nyelven javasoljuk megoldani, de a hallgatók a gyakorlatvezetővel történő egyeztetés után más nyelvet is választhatnak. Az alábbi feladatokat ajánljuk:

1. Holtpont-megelőzés: Gráf topológikus rendezésén alapuló modellezés
2. Holtpont-megelőzés: Bankár algoritmus szemléltetése
3. Holtpont-megelőzés: Erőforrás pályagörbék
4. Holtpont-felismerés: DAG tulajdonság ellenőrzése és kör megszűntetése
5. Kötegelt jellegű ütemezési algoritmusok összehasonlító szemléltetése
   - Legrövidebbet először
   - Prioritásos ütemezés
   - FIFO
   - Kétszintű ütemezés
6. Interaktív jellegű ütemezési algoritmusok összehasonlító szemléltetése
   - Round Robin ütemezés
   - Többszörös sorok
   - Sorsjáték ütemezés
7. Valós idejű ütemezési algoritmusok összehasonlító szemléltetése
   - Állandó arány

- Legrövidebb határidőt először
- Legkisebb lazaság

8. Memóriafoglalás algoritmusainak összehasonlító szemléltetése
   - First fit
   - Next fit
   - Best fit
   - Worst fit
   - Quick fit
9. Lapcserélési eljárások összehasonlító szemléltetése
   - NRU
   - FIFO
   - LRU
   - Második lehetőség
   - Óra algoritmus
10. Egy fájlrendszer működésének egyszerűsített szimulációja, könyvtárak nélkül: láncolt listás, fix blokkméretű helyfoglalás
11. Egy fájlrendszer működésének egyszerűsített szimulációja, könyvtárak nélkül: láncolt listás, folytonos helyfoglalás
12. Egy fájlrendszer működésének egyszerűsített szimulációja, könyvtárak nélkül: i-bögös (többlépcsős) helyfoglalási módszer
13. Egy fájlrendszer működésének egyszerűsített szimulációja, könyvtárak nélkül: kiegyensúlyozott fákon alapuló helyfoglalási módszer