Temos

1. OS sąvoka
2. OS kategorijos
3. Multiprogramavimo sąvokos
4. Virtualios mašinos sąvoka
5. Lygiagretūs procesai
6. Kritinė sekcija
7. Kritinės sekcijos apsauga
8. Deterio algoritmas
9. Semaforai
10. Procesų gamintojas ir naudotojas, sąveika
11. Įvedimo/išvedimo spoolerio bendra charakteristika
12. I/O spoolerio pagrindinis procesas
13. I/O spoolerio įvedimo ir skaitymo procesai
14. Dvejetainių ir bendrų semaforų sąryšis
15. Operacijų su semaforais realizacija
16. Procesų ir resursų sąvokos
17. Proceso deskriptorius
18. Resurso deskriptorius
19. Primityvas kurti procesą
20. Primityvas naikinti procesą
21. Primityvas stabdyti procesą
22. Primityvas kurti procesą
23. Primityvas aktyvuoti procesą
24. Primityvas keisti proceso prioritetą
25. Primityvas kurti resursą
26. Primityvas naikinti resursą
27. Primityvas prašyti resurso
28. Primityvas atlaisvinti resursą
29. Primityvas proceso planuotojas
30. Primityvas pasiruošusio proceso su aukščiausiu prioritetu nustatymas ir neaktyvaus procesoriaus radimas
31. Procesų būsenų schema.
32. Virtualios atminties sąvoka

33. Komandos vykdymo schema
34. Puslapinė organizacija
35. Polisegmentinė virtuali atmintis
36. Atminties skirstymo puslapiais strategijos
37. Atminties išskyrimas ištisinėmis sritimis
38. Kintamo dydžio ištisinių atminties sričių grąžinimas
39. Gijų samprata, vartotojo ir branduolio gijos
40. Mikrobranduolio architektūra
41. Įvedimo/Išvedimo procesai
42. Failų sistemos sąvoka
43. Failinės atminties įrenginių charakteristika
44. Failų deskriptorius, aktyvių failų katalogas
45. Užklausimo failinei sistemai pavyzdžiai
46. Failų sistemos hierarchinis modelis
47. Failų sistemos Įvedimo/Išvedimo posistemė
48. Bazinė failų valdymo sistema
49. Loginė failų valdymo sistema
50. Priėjimo metodai

1. OS – tai organizuota programų sistema, kuri veikia kaip interfeisas tarp kompiuterio ir vartotojo.

   OS sudaro visa tai, kas vykdoma supervizoriaus režime.

   Funkcijos: 1) suteikia OS vartotojams tam tikrą servisą 2) valdo resursų skirtymą efektyvų resursų naudojimą.

2. Grynosios OS kategorijos: 1) apdorojimo sistemos 2) laiko skirstymo 3) realaus laiko sistemos

3. Multiprogramavimas – savybė vienu metu vykdyti kelias užduotis.

   Tam reikia: 1) pertraukimo mechanizmo, 2) privilegijuoto (supervizoriaus) režimo 3) atminties apsaugos.

4. Virtuali mašina – virtuali realios mašinos kopija. Ji paslepia realios mašinos realizacija po virtualiais komponentais, kurie reikalingi kokioms nors užduotims atlikti Supaprastina nepatogią ir sudėtingą vartotojo sąsaja.

5. Lygiagretūs procesai – procesai vykstantys vienu metu ir nepriklausomai vienas nuo kito.

6. **Kritinė sekcija** (KS) - proceso kodo dalis, reikalaujanti bendrų resursų panaudojimo vienu metu. Kad išvengtų užblokavimo, du susiję procesai negali būti savo kritinėse sekcijose tuo pačiu metu. Visos kritinės sekcijos lygiavertės. Procesų vykdymo laikas bet koks. Programa gali būti užbaigta tik už kritinės sekcijos ribų.

7.

8. Deterio algoritmas – C_i = false. Įvykdomas noras vykdyti kritinę sekciją, jei kiti neturi ketinimų vykdyti kritinę sekciją. Bet gali būti, kad vienu metu norės vykdyti keletą procesų KS. Tada įvedamas bendras kintamasis EILĖ. Jei procesas neturi eilės vykdyti kritinę sekciją, jis atsistako šito savo noro.

9. Semaforai – sveikas neneigiamas kintamasis, su kuriuo galima atlikti dvi operacijas: wait (dar vadinama P) ir signal (dar vadinama V). Įėjimas į KS kontroliuojamas ***wait*** operacija; apie išėjimą iš KS signalizuoja ***signal***. Šios operacijos yra nedalomos ir vykdymo negalima nutraukti ar vykdymo metu kreiptis į tą semaforą. Kiekvienam semaforui sistema sudaro jo laukiančių procesų eilę. Kitas privalumas - OS perveda laukiantį procesą į, pvz., būseną "sustabdytas", t.y. kol vyksta laukimas procesoriaus laikas nenaudojamas.

   wait(s):
   if *s > 0*
   then *s := s - 1*
   else *{užblokuoti kviečiantį procesą};*
   wait *(s);{* ***KS*** *} ;* ***signal*** *(s)*

   signal(s):
   if *{yra užblokuotas procesas (-ai)}*
   then *{pradėti kurį nors procesą}*

else *s := s + 1;*

10. Procesas gamintojas sukuria informaciją ir patalpina ją į buferį, o lygiagrečiai veikia kitas procesas naudotojas paima informaciją iš buferio ir apdoroja. Naudojami tam du semaforai T – tuščių buferių semaforas U – užimtų buferių semaforas. Kadangi buferis bendras resursas abiem procesam, tai jis yra kritinė sekcija B, sauganti nuo kolizijų.
11. OS tai pat turi vadinamąją SPOOL sistemą periferinių sistemų on-line'e. Jis skirtas I/O įrenginių virtualizavimui.

    SPOOL – lygiagrečiai veikiančių procesų visuma. Buferio spoole organizuojami trys sąrašai: a) laisvi buferiai b) įvedimo buferiai c) išvedimo buferiai

    Proceso I f-ja: Paima buferį iš laisvų buferių sąrašo, pradeda skaitymo procesą ir duoda ja laisvą buferį.

    Proceso O f-ja: ima buferį iš išvedimo buferio sąrašo, paleidžiaW ir jam perduoda paimtą išvedimo buferį

12.

13.

14. Bet kuris semaforas gali būti išreikštas dvejetainiu semaforu.

    NS,M,D NS-kintamasis, M,D-dvejetainiai semaforai. P(S) – kritinę sekciją saugantis semaforas, kurio reikšmę modeliuoja NS.

15.

16. Procesas – operacinės sistemos primityvas, apibrėžiamas labai skirtingai – nuo „savarankiškai vykdoma programa“ iki „tai, kas sukuria fork(2) arba clone(2)“.

    Resursai yra tai, dėl ko procesas yra blokuojamas.

    Proceso sąvoka su dviem pagrindinėm charakteristikom:

    1) visų resursų rinkinio valdymas. Tai sudaro sudėtingą procesų kontekstą. Toks pakeitimas – ilgai trunkanti operacija. Iš kitos pusės net ir esant tam pačiam kontekstui, reikia vykdyti skirtingus kontekstus, todėl reikalinga skirtingų instrukcijų seka. Tokia instrukcijų valdymo seka vadinama gija.

17. F
18. f
19. f
20. f
21. f
22. f

23. f
24. f
25. f
26. f
27. f
28. f
29. f
30. f
31. f
32. Virtuali atmintis yra realios atminties modelis, kuris supaprastina adreso transliaciją. Jis turi savyje patogumo elementą. Jeigu anksčiau turėjo pats programuotojas rūpintis kaip skirstyt atmintį, o tam reikia aukštos kvalifikacijos. OS perėmus šį procesą, ji pateikia patogų interfeisą, kiekvienas vartotojas turi didelę atskirą virtualią atmintį.VR gali būti saugoma su RAM arba statiškai, arba dinamiškai. Yra svarbu, kad programos adresų nustatymas pagal fizinę vietą būtų atliekamas architektūriškai.
33. f
34. Puslapinė organizacija – konkretus VR įgyvendinimo būdas. Vienas VR puslapis atvaizduojamas į vieną atminties bloką. Čia fragmentacijos problema nėra aktuali. Antras būdas puslapinė organizacija su segmentacija. Architektūroje turi būti numatyta segmentų lentelės registras, kuris rodo į aktyvaus proceso segmentą, o jau jis turi savo PLR.
35. Galima apibrėžt ir daugelio segmentų virtualią atmintį. Tada jis būtų identifikuojamas nurodant segmentą ir žodį jame.
36. f
37. f
38. f
39. Visų resursų rinkinio valdymas sudaro gana sudėtingą procesų kontekstą. Toks pakeitimas – ilgai trunkantis operacija. Iš kitos pusės net ir esant tam pačiam kontekstui reikia vykdyti skirtingus kontekstus, todėl reikia skirtingų instrukcijų sekų. Tokia instrukcijų valdymo seka vadinama gijos (thread).Sukurti, sunaikinti, pereiti, komunikuoti tarp gijų yra daug greičiau nei tarp procesų. Gijos turi būsenas ir jų vykdymas gali būti, o kartais ir turi būti sinchronizuotas. Neturi pristabdymo būsenos. Su gijomis asocijuojamos operacijos. Operacijos: Create – kai sukuriamas procesas, automatiškai sukuriama ir gija. Gĩja gali sukurti naują giją proceso viduje. Finish – gijos darbo užbaigimas. Block. Unblock – patalpinama į pasiruošusių gijų sąrašą. Gijų blokavimas gali užblokuoti ir procesą.

    Vartotojo gijos. Šių gijų perjungimui nereikia branduolio įsikišimo.Gijų biblioteka nepriklauso nuo OS įsikišimo, tai taikomojo lygmens programa. O vartotojo gijų trūkumai – jei gija blokuoja procesą, blokuojamos ir kitos proceso

gijos.Vartotojo gijos negali išnaudoti daugiaprocesorinę sistemą.

Branduolio gijos. Jų valdymas atliekamas branduolyje. Jei taikomojoje programoje sukurtos gijos priklauso vienam procesui ir turi tą patį kontekstą, tai branduolys valdo visas gijas ir įveikia visus minėtus trūkumus, iššauktus vartotojo gijų. Branduolio gijų skaičius ribotas.

Jei persijungimai vyksta vartotojų gijų ribose, tai užtenka vartotojo gijų, jei kviečiamas sisteminis procesas, tai naudojamos branduolio gijos.

40. Mikrobranduolys yra OS nedidelė atminties dalis, įgalinanti OS modulinį plėtimą. Nėra vieningos mikrobranduolio sandaros. Problema yra driver'iai, juos reikia padaryt efektyvius. Į driver'į galima žiūrėti kaip į virtualų įrenginį, kuris palengvina įrenginio valdymą, pateikdamas patogesnį interfeisą. Kitas klausimas, kur vyksta procesai, ar branduolio erdvėj, ar už jo ribų. Pirmos OS buvo monolitinės, kur viena procedūra galėjo iškviesti bet kokią kitą procedūrą. Tai tapo kliūtimi augant OS. Buvo įvesta OS sluoksninė architektūra. Sudėtingumas nuo to nedingo. Kiekvienas sluoksnis gana didelis. Pakeitimai vienam sluoksnyje iššaukia pakeitimus ir gretimuose sluoksiniuose. Sunku kūrti versijas pagal konkrečią konfigūraciją. Sunku spręsti saugumo problemas dėl gretimų sluoksnių sąveikos.

    Mikrobranduolio sistemos atveju visi servisai perkelti į vartotojo sritį. Jie sąveikauja tarpusavyje ir su branduoliu. Tai horizontali architektūra. Jie sąveikauja per pranešimus, perduodamus per branduolį. Branduolio funkcija tampa pranešimo perdavimas ir priėjimas prie aparatūros.

    Mikrobranduolio architektūros pranšumai: 1) vieningas interfeisas 2) Išplečiamumas 3) Lankstumas 4) Pernešamumas(Portability) 5) Patikimumas 6) Tinkamumas realizuoti paskirtytas (išskirstytas) sistemas. Neigiama savybė – nepakankamas našumas, kalta pranešimų sistema, Jį reikia pernešti, perduoti, gavus atkoduoti. Atsiranda daug perjungimų tarp vartotojo ir supervizoriaus režimų.

41. I/O ir perdavimo valdymo sistema. Architektūra įgalina pertraukimus realizuoti kaip pranešimus. Branduolys atpažįsta pertraukimus, bet jų neapdoroja, o perduoda pranešimą vartotojo procesui, kuris atsako už interrupt'ą. Pačius įrenginius galima traktuoti kaip gijas, turinčias identifikatorius ir siunčiantiems pranešimus vartotojo erdvėje. I/O valdymui kiekvienam I/O įrenginiui sukuriamas jų aptarnaujantys procesai.

42. **Failų visuma** su visais jų tarpusavio ryšiais ir nusako failų sistemą. Talpumas – informacijos kiekis, kurį gali saugoti įrenginys. **Įrašo dydis** – informacijos kiekis, kuris gali būti įrašytas, atpažintas. **Failas** – turinti vardą sutvarkyta elementų seka. Failų sistemos funkcijos: 1) Užklausimų virtualiai failinei atminčiai konvertavimas į realią atmintį. 2) Failo info perdavimas tarp realios ir virtualios failinės atminties.

43. /*Tomai – diskeliai, cd-rom. Užlaikymas – timeout. Nustatymo laikas – galvučių perstūmimo laikas. Diskiniai įrenginiai : 1) diskinės atminties talpa 2) perdavimo, apsikeitimo greitis.

RAID – Redundant Array of Inexpensive (Independant) Disk. RAID0-RAID6 – RAID'ų standartai. */ ?

44. Failų deskriptorius FN – failo vardas, FA – failo padėtis įrenginyje. Nuosekli failų organizacija. L – ilgis, laikinas tipas, U – savininkas, {U} – naudotojai, READ – apsauga. Failų deskriptoriai saugomi diske kaip ir failai Aktyvių failų deskriptorius įtraukiamas į aktyvių failų katalogą (AFK) operatyvioje atmintyje. Vartotojo procesai – DBVS (duombazė)->priėjimo metodas->virtuali/loginė sistema <->OS procesai->bazinė/reali sistema->I/O sistema.
45. ęė
46. Hierarchinį modelį galima įsivaizduot kaip medį sudarytą iš failų, kai kurie iš jų yra direktorijos. Yra viena išimtis, kiekvienas failas ar direktorija gali save rasti priskirtą tik vienoje šakoje. Išimtis tik root direktorija. Ji nors nėra tiesioginiai sujungta su jokia šaka, tačiau yra fiktyviai prijungta prie failų sistemos. Kiekvienas žemiau hierarchijoje esantis failas yra žemesnio rango nei aukščiau stovintis.
47. I/O sistemą sudaro procesai valdantys I/O, kiekvienam įrenginiui atskiras procesas. Ryšys tarp procesų atitinka ryšį tarp įrenginių. Privalumai: 1) asinchroninis darbas, valdomas kaip nepriklausomas procesas ir įrenginių greičio skirtumai nereikalauja specialių OS veiksmų. 2) Ypatingos situacijos įrenginio darbe apdorojamos artimiausiame įr. lygyje.
48. Bazinė failų sistema transformuoja vidinį virtualaus failo adresą į išorinį (hardware) adresą. Darbui su failais turi funkcijas: SUKURTI (failo vardas, sritis), DALINTI(-||-,-||-) - sukuriamas dalinis failas, IŠPLĖSTI (-||-,-||-) - padidinti failo atmintį ATLAISVINTI (-||-,-||-)
49. Loginė failų valdymo sistema atvaizduoja failų vardus į identifikatorius. Loginė sistema reikalauja pranešimo apie darbo su failu pradžią, tada atidaro failą. Kiekvienas vartotojas sistemoje identifikuojamas vartotojo deskriptoriumi, kuriame yra informacija apie vartotojo failus. Loginė failų sistema pateikia komandas, kurioms realizuoti kreipiasi į bazinės failų sistemos komandas: REGISTRUOTI(f), SUKURTI (fv, [tomo vardas]), SUNAIKINTI (fv), ATIDARYTI (fv), UŽDARYTI (fv), SKAITYTI (fv, bl.nr, OA, bl.nr), RAŠYTI(-||-,-||-,-||-,-||-,-||-,-||-)
50. Priėjimo būdai: 1) tiesioginis – kai operuojama aparatūriniu įrenginio adresu. 2) nuoseklus – kai reikalingas visų kitų praeinamų įrenginių įrašų peržiūrėjimas. Ne tai/

    1) baziniai arba su eilėmis 2) tiesioginiai arba nuoseklūs.

    Bazinis – vartotojas pats turi dirbti su buferiu.

    Su eilėmis – visi įrašai siunčiami į programą iš eilės.

    Nuoseklūs – visi įrašai skaitomi iš eilės.

    Tiesioginis – galima kreiptis į bet kurią failo dalį.

http://vejas.pit.ktu.lt/~os/Atminties_valdymas.htm atmintis