VILNIAUS UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

**Operacinės sistemos**

Trečiasis praktinis darbas

„Multiprograminės operacinės sistemos projektas“

Darbą atliko:

Marek Rusakevič

Arnas Petruškevičius

3k, MI

TURINYS

1. **Procesai**
   1. Procesų būsenos
   2. Planuotojas
   3. Prioritetai
   4. Planuotojo veiksmų seka
   5. Proceso ir su juo susijusių klasių aprašai
   6. Procesų primityvai
   7. Sisteminiai ir vartotojiški procesai
2. **Resursai**
   1. Resursų aprašai
   2. Resurso primityvai
   3. Resurso paskirstytojas
3. **Procesų paketas**
   1. Procesas StartStop
   2. Procesas Read
   3. Procesas MainProc
   4. Procesas Loader
   5. Procesas JobGovernor
   6. Procesas VirtualMachine
   7. Procesas Interrupt
4. **Procesai**

Procesas – tai vykdoma programa, kartu su esamomis registrų reikšmėmis ir savo kintamaisiais. Kiekvienas procesas turi savo virtualų procesorių.

* 1. **Procesų būsenos**

Procesas gali gauti procesorių tik tada, kai jam netrūksta jokio kito resurso. Procesas gavęs procesorių tampa vykdomu. Procesas, esantis šioje būsenoje, turi procesorių, kol sistemoje neįvyksta pertraukimas arba einamasis procesas nepaprašo kokio nors resurso. Procesas blokuojasi priverstinai. Tačiau, jei procesas nereikalauja jokio resurso, iš jo gali būti atimamas procesorius, pavyzdžiui, vien tik dėl to, kad pernelyg ilgai dirbo. Tai visiškai skirtinga būsena nei blokavimasis dėl resurso.

Procesų būsenos:

* Vykdomas - turi procesorių
* Blokuotas - prašo resurso (išskyrus procesorių)
* Pasiruošęs – vienintelis trūkstamas resursas yra procesorius

Vykdomas

Blokuotas

Pasiruošęs

* 1. **Planuotojas**

Planuotojo paskirtis yra atimti procesorių iš proceso, peržvelgti pasiruošusių procesų sąrašą, išrinkti pasiruošusį procesą, kuris planuotojo manymu yra tinkamiausias, ir perduoti procesorių jam.

* 1. **Prioritetai**

Proceso prioritetas – tai proceso svarba, įvertinta kokioje nors skalėje. Procesas, turintis didesnį prioritetą, atsiduria arčiau procesų sąrašo pradžios. Visi procesų sąrašai yra rūšiuojami pagal proceso prioritetą, ir arčiau pradžios esantis procesas turi didesnę galimybę tapti vykdomu. Vartotojiški procesai turi kur kas mažesnį prioritetą nei sisteminiai.

* 1. **Planuotojo veiksmų seka**

Yra pasiruošusių procesų?

Einamasis procesas blokuotas?

Ne

Ne

Ypatinga situacija

Taip

Taip

Vykdomas procesas

Perduodamas valdymas šiam procesui

Imamas pirmas pasiruošęs procesas

Einamasis procesas įtraukiamas į blokuotų proc. sąrašą

* 1. **Proceso ir su juo susijusių klasių aprašai**

Proceso aprašas TProcess:

* FList: TProcessList; - procesų sąrašas, kuriam priklauso procesas. Tai gali būti pasiruošusių procesų sąrašas arba blokuotų, dėl kokio nors resurso, procesų sąrašas.
* FID: Integer;- Vidinis proceso vardas. Kiekvienas sisteminis objektas turi savo unikalų numerį.
* FSavedRegisters: TSavedRegisters; - Proceso išsaugota procesoriaus būsena. Naudojama perduodant valdymą procesui.
* FProcessor: Proc; nuoroda į procesorių
* FCreatedRes: TResourceList; - Proceso sukurti resursai.
* FOwnedRes: TElementList; - Procesui kūrimo metu perduoti resursai
* FState: TProcessState; - Proceso būsena.
* FPriority: Integer; - Proceso prioritetas. Pasirinkta sistema: kuo didesnė reikšmė, tuo procesas laikomas svarbesniu .
* FParent: TProcess; - nuoroda į procesą–tėvą
* FChildren: TProcessList; - procesų–vaikų sąrašas.
* FKernel:TKernel; - Nuoroda į branduolį
* FUserName: String; - Išorinis vardas. Naudojamas patogesniam proceso identifikavimui OS stebėjimo metu.

Procesų sąrašo aprašas TProcessList:

* FKernel: TKernel; - Nuoroda į branduolį
* FProcesses: array[ ]of TProcess: - Dinaminis procesų sąrašas.
* FCount: Integer ; Procesų skaičius procesų masyve.

Operacinės sistemos branduolys Tkernel:

* FProcesses: TProcessList; - Bendras visų, esančių sistemoje, procesų sąrašas
* FResources: TresourceList ;-Bendras visų, sistemoje esančių, resursų sąrašas
* FReadyProc: TProcessList; - Pasiruošusių procesų sąrašas
* FRunProc: TProcessList; - Einami procesai.
  1. **Procesų primityvai**

Procesų primityvų paskirtis – pateikti vienodą ir paprastą vartotojo sąsają darbui su procesais. Darbui su procesais skirti 2 primityvai:

* Kurti procesą - Šiam primityvui perduodama nuoroda į jo tėvą, jo pradinė būsena, prioritetas, perduodamų elementų sąrašas ir išorinis vardas. Pačio primityvo viduje vyksta proceso kuriamasis darbas. Jis yra registruojamas bendrame procesų sąraše, tėvo-sūnų sąraše, skaičiuojamas vidinis identifikacijos numeris, sukuriamas jo vaikų procesų sąrašas (tuščias), sukurtų resursų sąrašas ir t.t.
* Naikinti procesą - Pradedama naikinti proceso sukurtus resursus ir vaikus. Vėliau išmetamas iš tėvo sukurtų procesų sąrašo. Toliau išmetamas iš bendro procesų sąrašo ir, jei reikia, iš pasiruošusių procesų sąrašo. Galiausiai naikinami visi jam perduoti resursai ir proceso deskriptorius yra sunaikinamas.
  1. **Sisteminiai ir vartotojiški procesai**

Procesus galima suskirstyti į sisteminius ir vartotojiškus. Sisteminis procesas – tai procesas, atliekantis tam tikras sistemines funkcijas, pavyzdžiui, vartotojiško proceso palaikymą. Vartotojiškas procesas skirtas vartotojo užduočiai atlikti. Sisteminiai procesai yra kuriami paleidžiant sistemą, o naikinami – naikinant sistemą. Vartotojiški procesai yra sukuriami sisteminių procesų jau veikiant sistemai. Paprastai sisteminiai procesai turi turėti aukštesnį prioritetą nei vartotojiški.

1. **Resursai**
   1. **Resursų aprašai**

Resurso aprašas TResource:

* FID: Integer ; - vidinis resurso vardas. Kiekvienas sisteminis objektas turi savo unikalų numerį.
* FCreator: TProcess; -Nuoroda į procesą, sukūrusį šį resursą.
* FList: TElementList; - Nuoroda į resurso elementų sąrašą
* FWaitingProc : TProcessList; -Nuoroda į šio resurso laukiančių procesų sąrašą
* FWaitingCount: TIntegerList; -Nuoroda į šio resurso laukiančių procesų paprašytų resurso kiekių sąrašą.
* FKernel: TKernel ; - Nuoroda į branduolį.
* FResourceList: TResourceList;- nuoroda į visų resursų sąrašą
* FWaitingProcPoint: TList; - nuoroda į šio resurso laukiančių procesų resurso elementų rodyklių sąrašą.

Resursų sąrašo aprašas TresourceList:

* Count: Integer ; - sąraše esančių resursų kiekis.
* Kernel: TKernel ; - nuoroda į branduolį
* Resources: array[ ] of TResource ; - dinaminis masyvas, kurio elementai – resursai.

Resurso elemento aprašas TresElement:

* FElementList: TelementList- nuoroda į resurso elementų sąrašą, kuriame yra šis resurso elementas.
* FReceiver: TProcess;- procesas, kuris turi gauti šį resurso elementą. Jei šio lauko reikšmė lygi nil, tai laikoma, kad šį elementą gali gauti bet kuris procesas.
* FSender: TProcess; - procesas, atlaisvinęs šį resurso elementą.

Resurso elementų sąrašas TelementList:

* FResource: TResource; - Nuoroda į resursą, kuriam priklauso šis elementų sąrašas
* FCount: Integer; - Elementų skaičius sąraše. Šis skaičius nurodo, kiek resurso elementų yra laisvų (t.y. prieinamų procesams) iš viso.
* FElement:array [ ] of TResElement; - dinaminis elementų sąrašas.
  1. **Resurso primityvai**

Resursas turi keturis primityvus:

* Kurti resursą - Resursus kuria tik procesas. Resurso kūrimo metu perduodami kaip parametrai: nuoroda į proceso kūrėją, resurso išorinis vardas. Resursas kūrimo metu yra: pridedamas prie bendro resursų sąrašo, pridedamas prie tėvo suskurtų resursų sąrašo, jam priskiriamas unikalus vidinis vardas, sukuriamas resurso elementų sąrašas ir sukuriamas laukiančių procesų sąrašas.
* Naikinti resursą - Resurso deskriptorius išmetamas iš jo tėvo sukurtų resursų sąrašo, naikinamas jo elementų sąrašas, atblokuojami procesai, laukiantys šio resurso, išmetamas iš bendro resursų sąrašo, ir, galiausiai naikinamas pats deskriptorius.
* Prašyti resurso - Procesas, iškvietęs šį primityvą, yra užblokuojamas ir įtraukiamas į to resurso laukiančių procesų sąrašą. Sekantis šio primityvo žingsnis yra kviesti resurso paskirstytoją.
* Atlaisvinti resursą - Šį primityvą kviečia procesas, kuris nori atlaisvinti jam nereikalingą resursą arba tiesiog perduoti pranešimą ar informaciją kitam procesui. Resurso elementas, primityvui perduotas kaip funkcijos parametras, yra pridedamas prie resurso elementų sąrašo. Šio primityvo pabaigoje yra kviečiamas resursų paskirstytojas.
  1. **Resurso paskirstytojas**

Kaip kad procesorius yra skirstomas planuotojo, kiekvienas resursas taipogi yra skirstomas tam tikro paskirstytojo. Prašydamas resurso ar norėdamas jį atlaisvinti, procesas kreipiasi į atitinkamą resursų paskirstytoją, kuris privalo jį aptarnauti.

1. **Procesų paketas**

Mūsų modelyje užduoties keliui palaikyti bus naudojami šie procesai:

* StartStop - šakninis procesas, sukuriantis bei naikinantis sisteminius procesus ir resursus.
* Read - užduoties nuskaitymo iš įvedimo srauto procesas
* MainProc - Procesas valdantis JobGorvernor procesus.
* JobGovernor - virtualios mašinos proceso tėvas, tvarkantis virtualios mašinos proceso darbą
* Loader - iš išorinės atminties duomenys perkeliami į vartotojo atmintį
* VirtualMachine - procesas atsakantis už vartotojiškos programos vykdymą.
* Interrupt - procesas, apdorojantis virtualios mašinos pertraukimą sukėlusią situaciją.

Beveik visi procesai yra sukuriami sistemos darbo pradžioje proceso StartStop. StartStop nekuria tik 2 procesų – JobGorvernor (kiekvienai naujai vartotojo užduočiai MainProc kuria po naują procesą JobGorvernor) ir VirtualMachine, kurį kuria JobGorvernor.

Read

............

Interrupt

VirtualMachine

VirtualMachine

JobGovernor

JobGovernor

MainProc

* 1. **Procesas StartStop**

Šis procesas atsakingas už sistemos darbo pradžią ir pabaigą. Šio proceso paskirtis – sisteminių procesų ir resursų kūrimas.

Start

Sisteminių procesų naikinimas

Sisteminių procesų naikinimas

Blokavimas laukiant „END“ resurso

Sisteminių procesų inicializacija

Sisteminių resursų inicializacija

Stop

Procesas StartStop, gavęs procesorių, savo darbą pradeda sukurdamas visus sisteminius resursus. Sukūręs resursus, StartStop kuria procesus: Loader, MainProc, Interrupt ir Print. Sekantis StartStop etapas yra prašyti resurso “END”. Šioje vietoje StartStop blokuojasi ir laukia kol bus atlaisvintas pranešimas apie darbo pabaigą. Priklausomai nuo prioriteto anksčiau ar vėliau StartStop bus atblokuotas ir tęs darbą. Sekantis etapas yra sisteminių procesų naikinimas. Galiausiai yra naikinami sisteminiai resursai.

* 1. **Procesas Read**

Šį procesą kuria ir naikina procesas StartStop. Šio proceso paskirtis – gavus informaciją iš įvedimo srauto atlikti jos apdorojimą.

Įvedimo apdorojimas

Blokavimas laukiant „Iš vartotojo sasajos“ resurso

* 1. **Procesas MainProc**

Blokavimas laukiant „Užduotis būgne“ resurso

Vykdymo laikas ne 0 ?

Ne

Taip

Naikinamas procesas JobGorvernor, sukūręs gautąjį resursą

Kuriamas procesas JobGorvernor, suteikiant jam “Užduotis būgne” kaip pradinį resursą

Procesą MainProc kuria ir naikina procesas StartStop. Šio proceso paskirtis - kurti ir naikinti procesus JobGorvernor. Šis procesas pradeda savo veiklą prašydamas “Užduotis būgne” resurso, kurį atlaisvina procesas JobGorvernor. Gavęs reikalingą resursą, MainProc žiūri ar jo vykdymo laikas nėra 0. Nenulinė reikšmė – naujo JobGorvernor kūrimo ženklas, kai tuo tarpu nulinė reikšmė yra JobGorvernor naikinimo ženklas. Kurdamas JobGorvernor procesas suteikia jam “užduotis būgne” kaip pradinį resursą. Naikinamas būtent tas JobGorvernor, kuris atsiuntė “Užduotis būgne” resursą, su nuliniu vykdymo laiku. Atlikęs savo darbą MainProc vėl blokuojasi laukdamas “Užduotis būgne resurso”.

* 1. **Procesas Loader**

Blokavimasis laukiant “Pakrovimo paketo” resurso

Atlaisvinamas “Iš Loader” resursas, skirtas JobGorvernor procesui, sukūrusiam gautąjį “Pakrovimo paketo”resursą.

Duomenys yra kraunami iš iššorinės atminties

Procesas Loader pradeda savo darbą laukdamas “Pakrovimo paketo” resurso. Pakrovimo pakete yra laikoma informacija apie išorinės atmintį (ką reikės perkelti). Vėliau duomeny yra kopijuojami į atmintį. Tada yra sukuriamas bei atlaisvinamas pranešimas JobGorvernor apie darbo pabaigą. Pranešimas yra skiriamas būtent tam JobGorvernor, kuris atsiuntė “Pakrovimo paketo” resursą. Įvykdęs savo darbą procesas Loader vėl blokuojasi laukdamas “Pakrovimo paketas ” resurso.

* 1. **Procesas JobGovernor**

Procesą JobGorvernor kuria procesas MainProc. Proceso JobGorvernor paskirtis – kurti ir naikinti procesus VirtualMachine. Vienas JobGorvernor aptarnauja vieną virtualią mašiną.

Procesas JobGorvernor pradeda darbą laukdamas varotojo atminties, kur bus perkelta vartotojo užduoties programa. Gavęs “Vartotojo atmintis” resursą procesas atlaisvina resursą “Pakrovimo paketas”, skirtą Loader ir blokuojasi laukdamas pranešimo apie Loader darbo pabaigą. Dabar paprašoma dar vieno takelio vartotojo atminties, kur bus laikoma virtualios mašinos puslapių lentelė. Dabar puslapių lentelė yra užpildoma išskirtosios vartotojo atminties adresais. Dabar yra pasiruošta virtualios mašinos kūrimui, taigi sekantis žingsnis yra kurti procesą Virtuali mašina. Dabar JobGorvernor blokuosis tol, kol negaus pranešimo iš proceso Interrupt apie įvykusį pertraukimą proceso Virtuali mašina metu.

Gavęs pranešimą apie pertraukimą, JobGorvernor stabdo procesą Virtuali mašina. Yra tikrinama ar tai įvedimo - išvedimo pertraukimas. Jei tai nėra įvedimo – išvedimo pertraukimas, naikinamas procesas “Virtuali mašina”, atlaisvinama jos užimta vartotojo atmintis ir duodamas ženklas procesui MainProc apie tai, kad šis JobGorvernor jau baigė darbą ir turi būti sunaikintas. Ženklas - sukurtas ir atlaisvintas fiktyvus resursas “Užduotis būgne” su nuliniu vykdymo laiku. Kadangi šis procesas savo darbą jau baigė, tačiau dar nėra sunaikintas, jis blokuojasi laukdamas “Neegzistuojantis” resurso. Šio resurso JobGorvernor negaus, taigi jis blokuosis tol kol nebus sunaikintas.

Blokavimasis laukiant “Vartotojo atmintis” resurso

Atlaisvinamas “Pakrovimo paketas” resursas

Blokavimasis laukiant “Iš Loader” resurso

Ne

Taip

Ne

Taip

Naikinti procesą “Virtuali mašina”

Atlaisvinti “Vartotojo atmintis” resursą

Atlaisvinti fiktyvų “Užduotis būgne” resursą

Blokavimasis laukiant “Neegzistuojantis” resurso

Proceso “Virtuali mašina” aktyvavimas

Blokavimasis laukiant “Vartotojo įvedimas” resurso, skirto būtent šiam JobGorvernor

Neviršytas išvedimo limitas ?

Įvedimo – išvedimo pertraukimas ?

Proceso “Virtuali mašina” stabdymas

Blokavimasis laukiant “Iš Interrupt” resurso

Proceso “Virtuali mašina” kūrimas

Blokavimasis laukiant “Vartotojo atmintis” resurso, skirto puslapių lentelei laikyti

* 1. **Procesas VirtualMachine**

Procesą Virtuali mašina kuria ir naikina procesas JobGorvernor. Virtualios mašinos paskirtis yra vykdyti vartotojo užduoties programą. Procesų Virtuali mašina yra tiek, kiek yra procesų JobGorvernor. Virtualios mašinos darbo pradžioje procesorius yra perjungiamas į vartotojo režimą. Vėliau vykdoma virtualaus procesoriaus komandų interpretatoriaus programa. Ši programa veikia tol, kol sistemoje neaptinkamas pertraukimas. Tada virtuali mašina išsaugoja savo procesoriaus būseną, perduodamas valdymas pertraukimą apdorosiančioms programoms. Apdorojus pertraukimą valdymui sugrįžus į virtualią mašiną, yra atlaisvinamas resursas “Pertraukimas”, skirtas procesui Interrupt. Procesas virtuali mašina turi mažą prioritetą, todėl procesorių visų pirma gaus procesas Interrupt, kuris identifikuos pertraukimą ir perduos informaciją JobGorvernor, kuris atlikęs nustatytus veiksmus, priklausomai nuo situacijos arba aptarnaus arba sunaikins procesą “Virtuali mašina”.

Atlaisvinamas “Pertraukimas” resursas

Vykdoma vartotojo programa

Procesorius perjungiamas į vartotojo režimą

* 1. **Procesas Interrupt**

Procesą Interrupt kuria ir naikina procesas StartStop. Šio proceso paskirtis – reaguoti į pertraukimus, kilusius virtualios mašinos darbo metu. Procesas Interrupt savo darbo pradžioje laukia “Pertraukimas” resurso, kurį siunčia procesas virtuali mašina. Galiausiai yra kuriamas ir atlaisvinamas “IšInterrupt” resursas, kuris yra skirtas nustatytajam JobGorvernor procesui. Toliau procesas cikliškai blokuojasi vėl laukdamas”Pertraukimas” resurso.

Atlaisvinamas “Iš Interrupt”resursas skirtas konkrečiam JobGorvernor

JobGorvernor, atsakingo už pertrauktosios VM darbą, nustatymas

Pertaukimo tipo identifikavimas

Blokavimasis laukiant “Pertraukimas” resurso