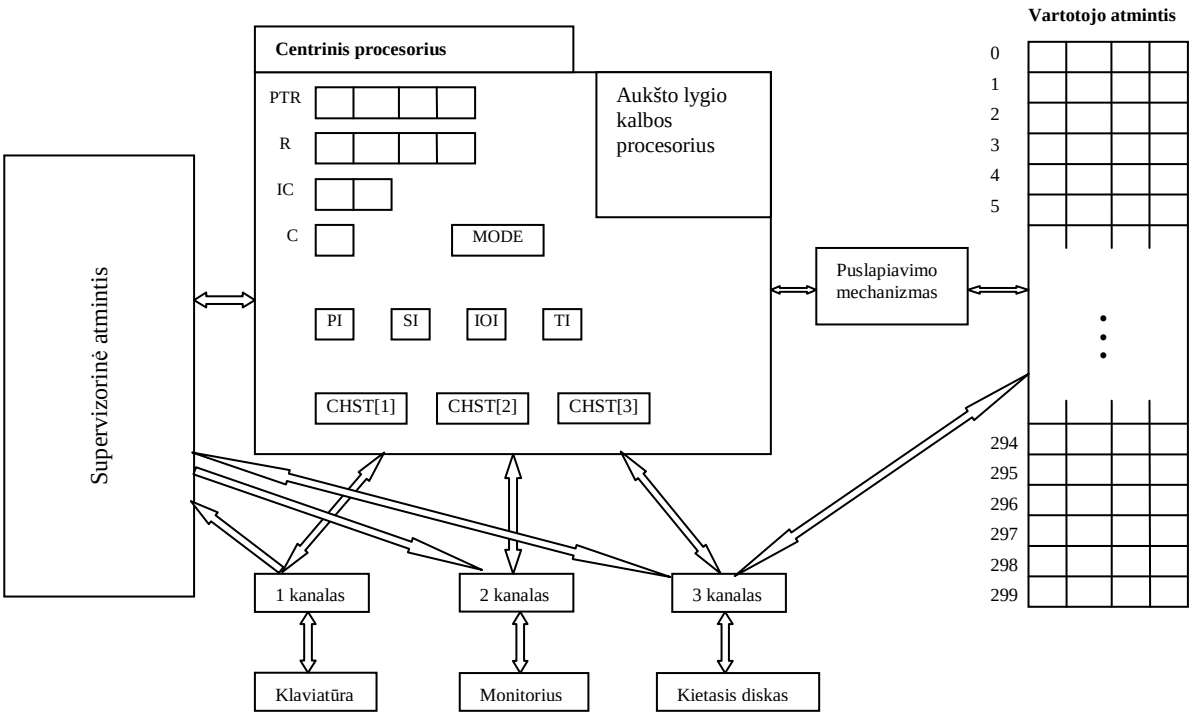
Operacines sistemos projektas

Interaktyvi operacinė sistema, kuri leidžia vartotojui interaktyviai paleisti pasirinktas programas, matyti jų išvedamus duomenis ekrane, įvesti reikiamus duomenis iš klaviatūros. Jei, vartotojui norint paleisti programą, pritrūksta atminties, sistema turi dalį vartotojo atminties puslapių perkelti į diską, t.y. naudoti swaping'ą.

Reali mašina



Centrinis procesorius:

- PTR – 4 baitų registras puslapių lentelei
- R – 4 baitų bendro naudojimo registras
- IC – 2 baitų virtualios mašinos atminties ląstelių skaitliukas
- C – 1 baito loginis
- MODE – procesoriaus darbo režimo požymio registras („S“ – supervizorius, „U“ – vartotojas)
- Pertraukimo registrai:
  - PI – programinio pertraukimo (1 baitas)
  - SI – supervizoriaus režimo perjungimo (1 baitas)
  - IOI – įvedimo/išvedimo pertraukimo (1 baitas)
  - TI – taimerio pertraukimo (1 baitas)
- CHST[i], i=1, 2, 3, – kanalų būsenos registrai (1 baitas (reikšmės 1 arba 0))

Pertraukimai:

Ar įvyko pertraukimas tikrinama po kiekvienos komandos. Kad pertraukimas įvyko sužinoma iš pertraukimo registrų reikšmių. Jei SI = 0, PI = 0 ir TI <> 0, tai pertraukimas neįvyksta. Kitaip:

1) SI = :

a. 1, tai pertraukimą iššaukė komanda GD
b. 2, tai pertraukimą iššaukė PD
c. 3, HALT

2) PI = :
   a. 1, tai neteisingas adresas
   b. 2, tai neegzistuojantis operacijos kodas
3) TI = 1
4) IOI = :
   a. 1, tai pertraukimas kilo 1 kanale
   b. 2 – pertraukimas kilo 2 kanale
   c. 4 – pertraukimas kilo 3 kanale

Jei pertraukimas kilo keliuose kanaluose, tai tos reiksmes susumuojamos.

Taimeris:
Naudojamas užduočių maksimaliam vykdymo laikui nustatyti.

Atmintis:
Vartotojo atminstis: Skirta virtualios mašinos atminčiai bei puslapių lentelei laikyti. Ji susideda iš 300 žodžių po 4 baitus. 10 žodžių laikysime bloku. Taigi Vartotojo atminti sudaro 30 blokų sunumeruotų nuo 0 iki 29, arba 300 žodžių sunumeruotų nuo 0 iki 299.

Supervisorinė atmintis: Skirta pačios operacinės sistemos poreikiams. Supervizorinėje atmintyje laikomi sisteminiai procesai, sisteminiai kintamieji, resursai.

Įvedimo ir išvedimo įrenginiai:
Įvedimo įrenginys – klaviatūra, išvedimo – monitorius.

Išorinė atmintis:
Joje bus saugomi į atmintį netilpę puslapiai (failas(-ai)) ir programų tekstai.

Procesai ir resursai

Procesai :
Procesas – tai vykdoma programa, kartu su esamomis registrų reikšmėmis ir savo kintamaisiais. Kiekvienas procesas turi savo virtualų procesorių Virtualios mašinos centrinis procesorius). Nors skirtumas tarp programos ir proceso nėra didelis, bet jis svarbus. Procesas – tai kokioje nors veiklumo stadijoje esanti programa. Tuo tarpu programa – tai tik tam tikras baitų rinkinys. Veiklumo stadiją apibūdina proceso aprašas – deskriptorius. Apraše ir yra laikomi visi procesui reikalingi parametrai; tokie kaip virtualaus procesoriaus registrų reikšmės, ar jam reikalingi kintamieji.

Procesų lygiagretus veikimas:
Virtualios mašinos veikia lygiagrečiai dėl pertraukimų, o sisteminiai procesai – blokuodamiesi, kai pritrūksta vieno ar kito resurso (statiniai resursai sukuriami OS darbo pradžioje proceso StartStop, žemesnio lygio procesai kuria ir naikina dinaminius resursus). Jei to paties resurso laukia keli procesai, pirmenybę turi procesas su aukštesniu prioritetu.

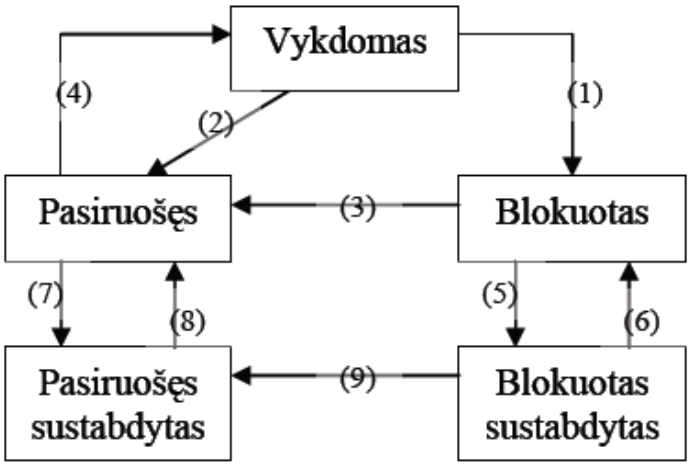
Procesų būsenos:
Procesas gali gauti procesorių tik tada, kai jam netrūksta jokio kito resurso. Procesas gavęs procesorių tampa vykdomu. Procesas, esantis šioje būsenoje, turi procesorių, kol sistemoje neįvyksta pertraukimas arba einamasis procesas nepaprašo kokio nors resurso (pavyzdžiui, prašydamas įvedimo iš klaviatūros). Procesas blokuojasi priverstinai (nes jis vis tiek negali tęsti savo darbo be reikiamo resurso). Tačiau, jei procesas nereikalauja jokio resurso, iš jo gali būti

atimamas procesorius, pavyzdžiui, vien tik dėl to, kad pernelyg ilgai dirbo.. Galime išskirti procesų būsenas:

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

Tačiau šių būsenų gali neužtekti. Gali susiklostyti tokia situacija, kad tam tikram procesui negalima leisti gauti procesorių, nors jis ir yra pasiruošęs. Toks procesas vadinamas sustabdytu.Galime papildyti būsenų sąrašą:

- **Sustabdytas** – kito proceso sustabdytas procesas.



1). Vykdomas procesas blokuojasi jam prašant ir negavus resurso.

2). Vykdomas procesas tampa pasiruošusiu atėmus iš jo procesorių dėl kokios nors priežasties (išskyrus resurso negavimą).

3). Blokuotas procesas tampa pasiruošusiu, kai yra suteikiamas reikalingas resursas.

4). Pasiruošę procesai varžosi dėl procesoriaus. Gavęs procesorių procesas tampa vykdomu.

5). Procesas gali tapti sustabdytu blokuotu, jei einamasis procesas jį sustabdo, kai jis jau ir taip yra blokuotas.

6). Procesas tampa blokuotu iš blokuoto sustabdyto, jei einamasis procesas nuima būseną sustabdytas.

7). Procesas gali tapti pasiruošusiu sustabdytu, jei einamasis procesas jį sustabdo, kai jis yra pasiruošęs.

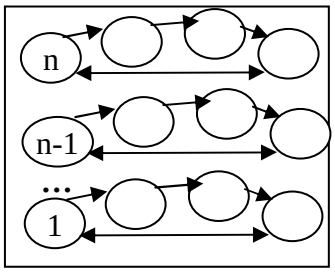
8). Procesas tampa pasiruošusiu iš pasiruošusio sustabdyto, jei einamasis procesas nuima būseną sustabdytas

9). Procesas tampa pasiruošusiu sustabdytu iš blokuoto sustabdyto, jei procesui yra suteikiamas jam reikalingas resursas.

Procesų prioritetai:

Vartotojiški procesai – laikini, skirti atlikti vartotojo užduotims (VirtualMachine).
Sisteminiai procesai – permanentiniai, užtikrinantys sistemos darbą (visi kiti).
Sisteminiai procesai turi aukštesnį prioritetą.

PPS (pasiruošusių procesų sąrašas)**:**
Vienodo prioriteto procesai apjungiami į atskirus sąrašus:



Resursai

Resursas yra tai, dėl ko varžosi procesai. Dėl resursų trūkumo procesai blokuojasi, gavę reikiamą resursą, procesai tampa pasiruošusiais. Resursus galima skirstyti į:

• Statinius resursus. Kuriami sistemos kūrimo metu. Tai mašinos resursai, tokie kaip procesorius, atmintis ar kiti resursai, kurie sistemos veikimo metu nėra naikinami. Šie resursai gali būti laisvi, kai nė vienas procesas jų nenaudoja, arba ne, kada juos naudoja vienas ar keli, jei tą resursą galima skaldyti, procesai.

• Dinaminius resursus. Kuriami ir naikinami sistemos darbo metu. Šie resursai naudojami kaip pranešimai. Kartu su jais gali ateiti naudinga informacija. Kartais šio tipo resursas pats yra pranešimas. Pavyzdžiui, esantis laisvas kanalo resursas žymi, kad bet kuris procesas gali naudotis kanalu. Jei jo nėra, procesas priverstas laukti, kol šis resursas taps prieinamu (bus atlaisvintas).

Resurso primityvai:

Resursas turi keturis primityvus:

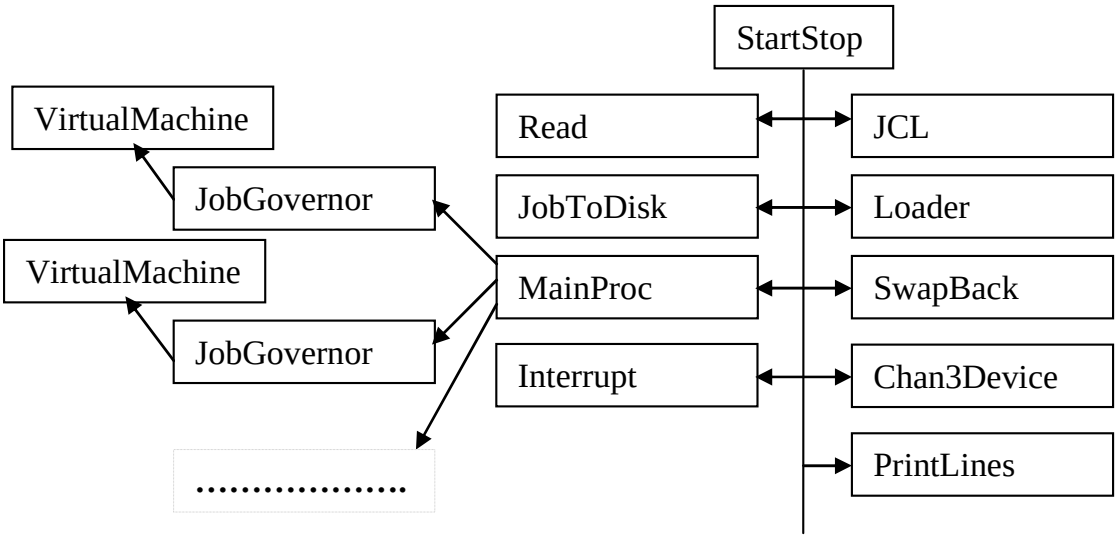
1. **Kurti resursą**. Resursus kuria tik procesas. Resurso kūrimo metu perduodami kaip parametrai: nuoroda į proceso kūrėją, resurso išorinis vardas. Resursas kūrimo metu yra: pridedamas prie bendro resursų sąrašo, pridedamas prie tėvo suskurtų resursų sąrašo, jam priskiriamas unikalus vidinis vardas, sukuriamas resurso elementų sąrašas ir sukuriamas laukiančių procesų sąrašas.

2. **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.

3. **Prašyti resurso.** Šį primityvą kartu su primityvu “atlaisvinti resursą” procesai naudoja labai dažnai. Procesas, iškvietęs šį primityvą, yra užblokuojamas ir įtraukiamas į to resurso laukiančių procesų sąrašą. Sekantis šio primityvo žingsnis yra kviesti resurso paskirstytoją.

4. **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.

Procesų hierarchija:



StartStop

Procesas *StartStop* atsakingas už sistemos darbo pradžią ir pabaigą. Įjungus kompiuterį šis procesas pasileidžia automatiškai. Šio proceso paskirtis – sisteminių procesų ir resursų kūrimas.

Veikimo schema:

START

Sisteminių resursų inicializacija

Sisteminių permanentinių procesų inicializacija

Blokavimasis laukiant darbo pabaigos

Blokavimasis laukiant išvedimo srauto resurso

Statistinės informacijos išvedimas

Sisteminių procesų naikinimas

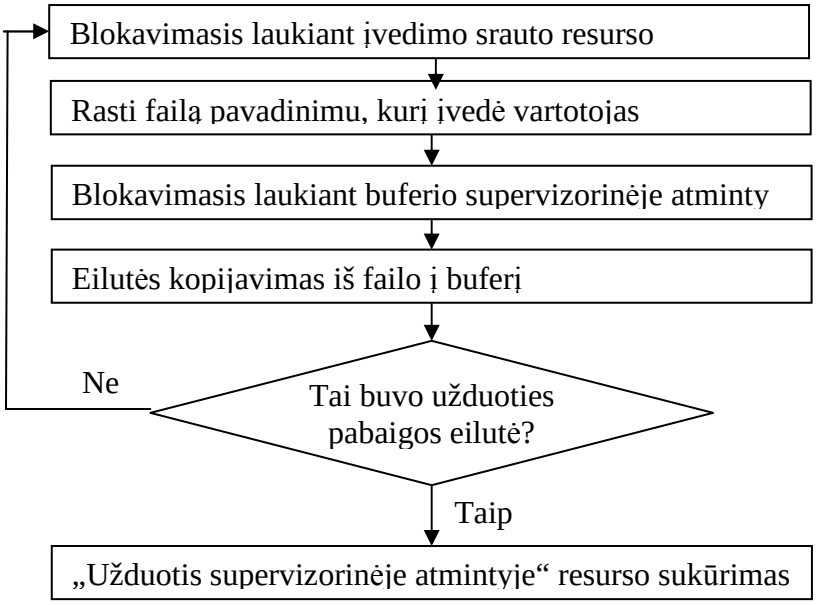
Sisteminių resursų naikinimas

STOP

Read

Read – užduoties nuskaitymo iš įvedimo srauto procesas.

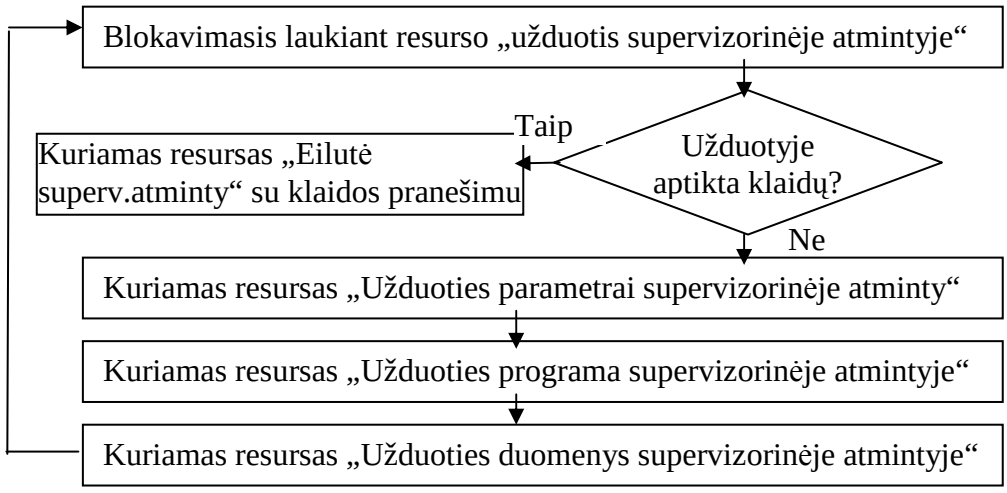
Veikimo schema:



JCL

JCL – procesas, interpretuojantis užduoties tekstą, išskiriantis programą, duomenis, parametrus ir užfiksuojantis užduoties formulavimo klaidas.

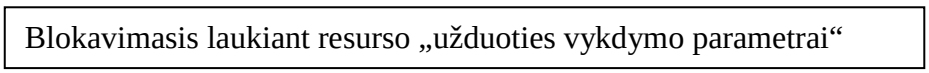
Veikimo schema:

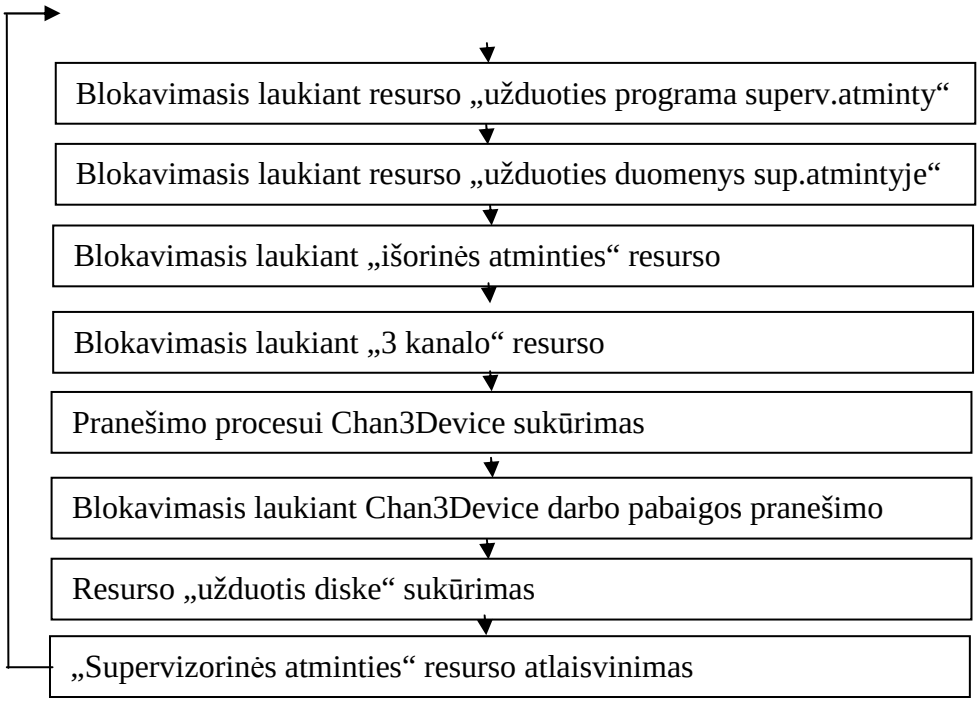


JobToDisk

JobToDisk – procesas, kuris patalpina užduotį iš supervizorinės į išorinę atmintį.

Veikimo schema:

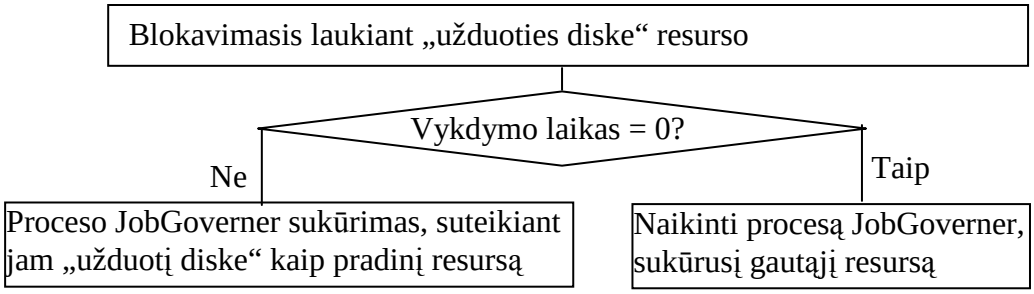




MainProc

MainProc – Procesas valdantis JobGorvernor procesus, t.y., kuriantis užduočių vykdymo aplinkas ir jas naikinantis.

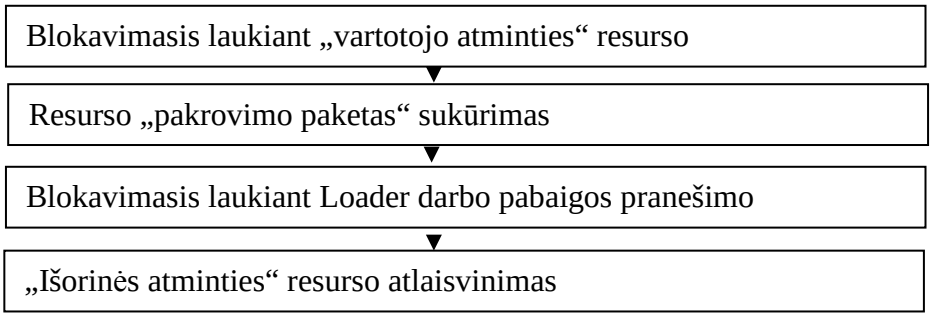
Veikimo schema:

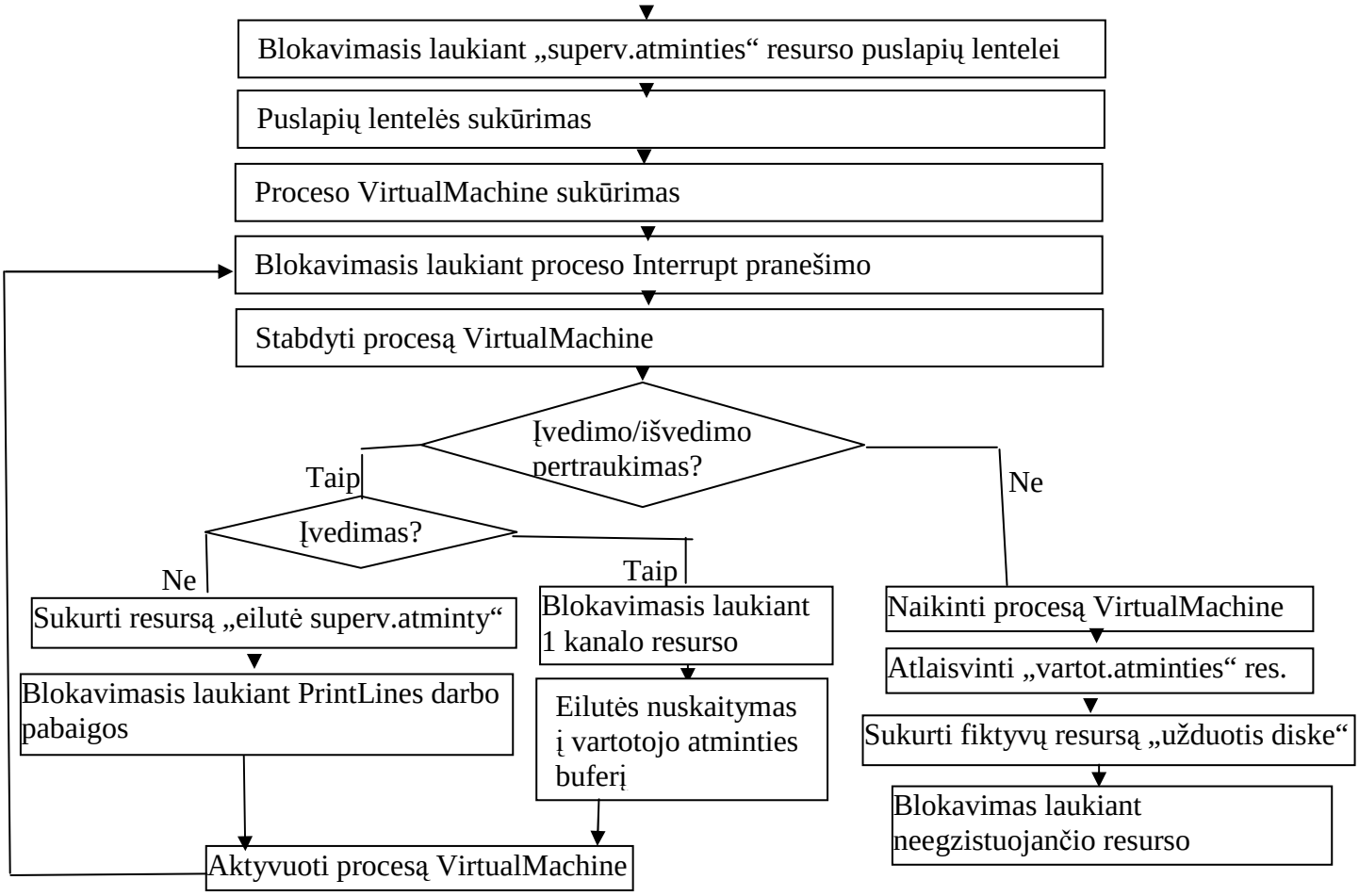


JobGovernor

JobGovernor – virtualios mašinos proceso tėvas, tvarkantis virtualios mašinos proceso darbą.

Veikimo schema:

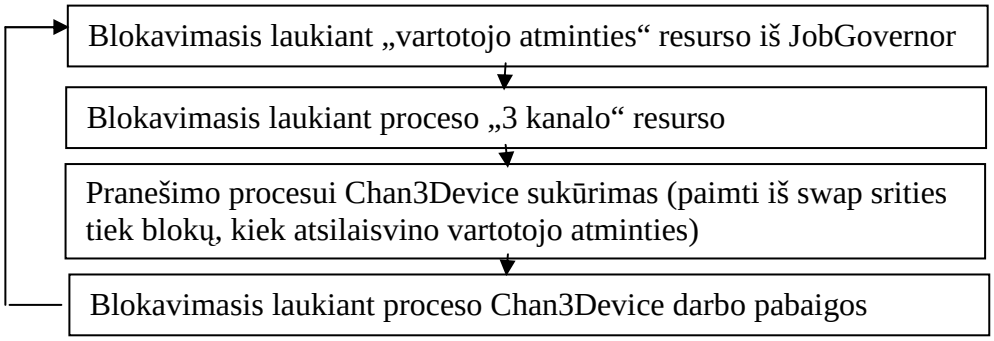




SwapBack

SwapBack – procesas, grąžinantis duomenis iš disko swap srities į vartotojo atmintį.

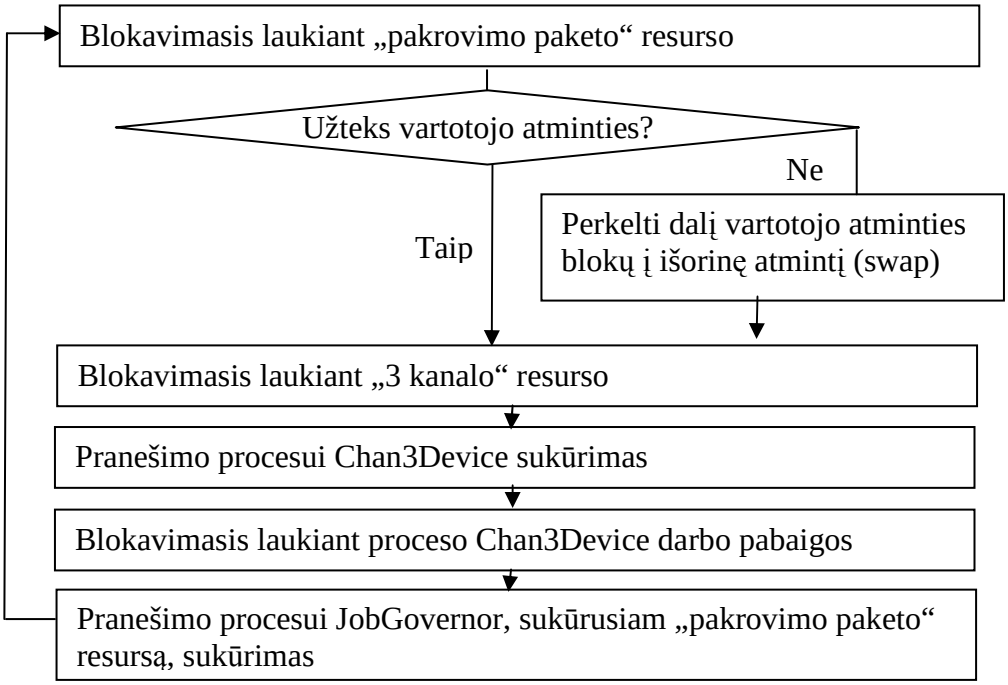
Veikimo schema:



Loader

Loader – iš išorinės atminties užduoties programos tekstas perkeliamas į vartotojo atmintį.

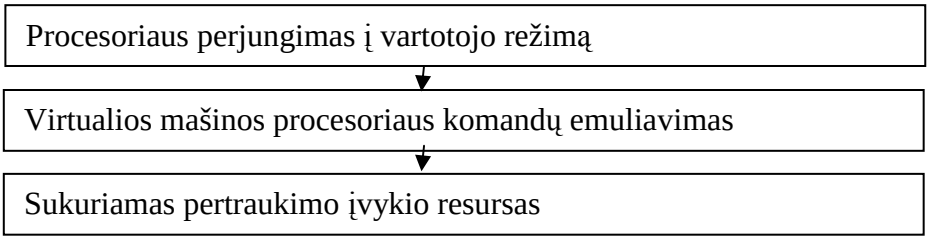
Veikimo schema:



VirtualMachine

Virtual Machine – procesas, atsakantis už vartotojiškos programos vykdymą.

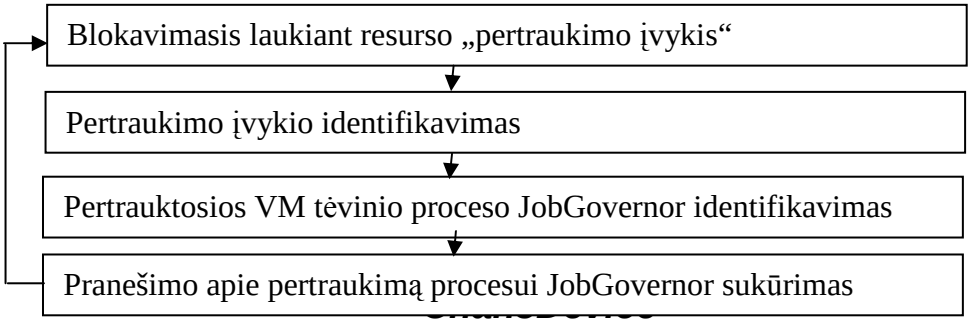
Veikimo schema:



Interrupt

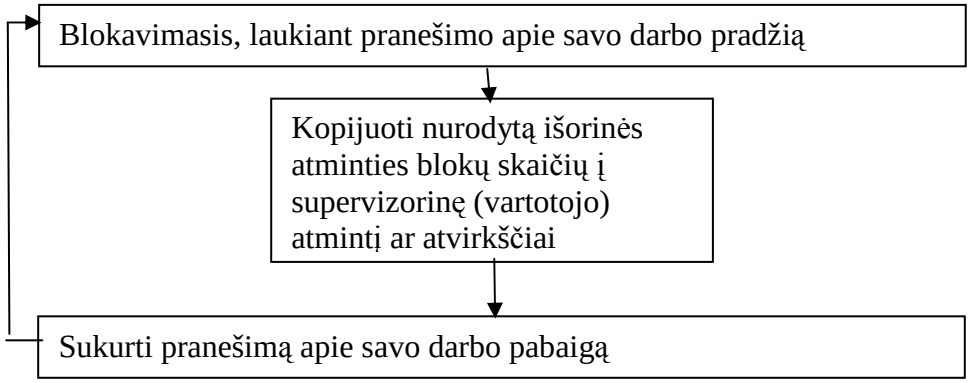
Interrupt – procesas, apdorojantis virtualios mašinos pertraukimą sukėlusią situaciją.

Veikimo schema:



Chan3Device – 3-iojo aparatūrinio kanalo valdymas, paslepiant jį nuo kitų procesų.

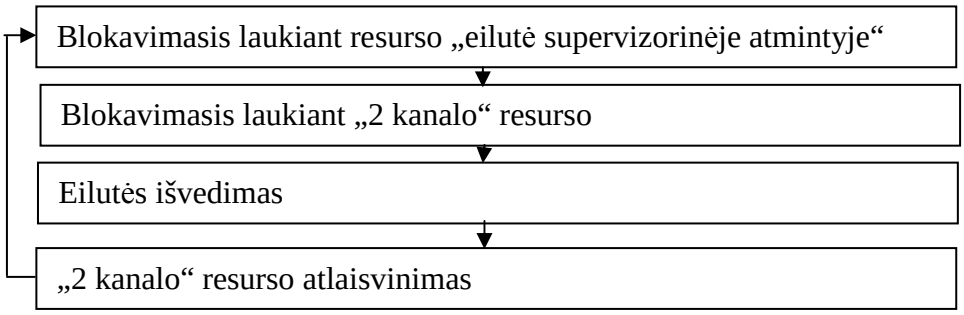
Veikimo schema:



PrintLines

PrintLines – į išvedimo įrenginį pasiunčiama eilutė iš supervizorinės atminties.

Veikimo schema:



Proceso deskriptoriaus struktūra:

Id	CPU	OA	R	SR	ST	PL	T	S	PR

Id: *String* // išorinis vardas.

CPU: { // procesoriaus einamoji būsena, dirbant su šiuo procesu:
mode: *char* // {s/u} – supervizorinis/vartotojiškas darbo režimas.
entry: *int* // (apibrėžtas sisteminiam procesui) – įėjimo taškas, nuo kurio bus pratęstas proceso darbas.
r: *byte[4]* // registras, atitinkantis VM registrą R.
ic: *short* // registras, atitinkantis VM komandų skaitliuką (du baitai).
c: *boolean* // registras, atitinkantis VM loginį trigerį:.
ptr: *byte[4]* // (apibrėžtas vartotojo procesui) – laukas puslapių lentelės registro reikšmei saugoti. (realizacijoje gali būti *int*).
clock: *int* // VM lokalaus laiko skaitliukas.
time: *int* // VM sutartinės darbo trukmės skaitliukas.
}

OA: *OABlocks* // procesui priklausanti operatyvioji atmintis.

R: *LinkedList* { // turimų resursų elementų sąrašas
  count: *int* // dalių kiekis (?)
  res_part: { // resurso dalies aprašymas
    ammount: *int* // išmatuojamojo resurso kiekis
    defin*: int[MAXINT]* // išmatuojamojo resurso dalių pasiekiamumas
    messid: String // laukiamo pranešimo/reikiamo kanalo identifikatorius
  }
}

SR: *LinkedList* { // sukurtų resursų deskriptorių sąrašas
  resource: *RD*
}

ST: *String* // {RUN, READY, BLOCK, READYS, BLOCKS} – proceso būsena

PL: *Queue* { // procesų sąrašas, kuriam šiuo metu priklauso procesas
  procid: *int*
  wanted: *res_part*: { // resurso dalies aprašymas
    ammount: *int* // išmatuojamojo resurso kiekis
    defin*: int[MAXINT]* // išmatuojamojo resurso dalių pasiekiamumas
    messid: String // laukiamo pranešimo/reikiamo kanalo identifikatorius
  }
}

T: *int* // tėvinio proceso vidinis vardas

S: *LinkedList* { // „sūnų“ sąrašas
  procid: *int*
}

PR: *int* // proceso prioritetas

Resurso deskriptoriaus struktūra:

Rid	PNR	K	PA	LPS	PASK

RId: *String* // resurso išorinis vardas

PNR: *boolean* // ar pakartotinio naudojimo resursas

K: *int* // sukūrusio proceso vidinis vardas

PA: { // resurso prieinamumo aprašymas
  msg: *boolean* // ar pranešimas, ar išmatuojamasis resursas
  list: *int[MAXINT]* // išmatuojamojo resurso blokų sąrašas
  number: *int* // kiek išmatuojamojo resurso elementų yra masyve list
  receiver: *int* // pranešimo gavėjo – proceso – vidinis vardas
}

LPS: *Queue* { // resurso laukiančių procesų sąrašas

procid: *int*
wanted: *res_part*: { // resurso dalies aprašymas
ammount: *int* // išmatuojamojo resurso kiekis
defin*: int[MAXINT]* // išmatuojamojo resurso dalių pasiekiamumas
messid: String // laukiamo pranešimo/reikiamo kanalo identifikatorius
}
}

PASK: *distributor()* // paskirstytojo funkcija

Procesoriaus deskriptorius PRD **:**

PROC

PROC: *int* // proceso, kurį vykdo procesorius, vidinis vardas

---

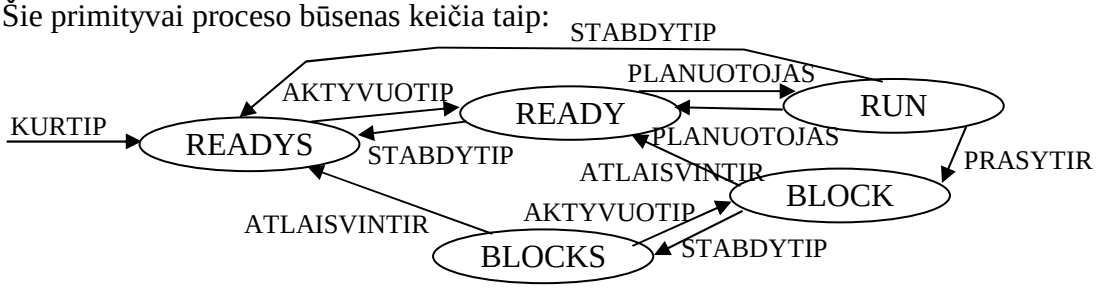
OS branduolys

Tai primityvai darbui su procesais ir resursais, nesudarantys atskirų procesų, o vykdomi juos kviečiančių procesų aplinkoje.

Darbo su procesais primityvai:
KURTIP, NAIKINTIP, STABDYTIP, AKTYVUOTIP, KEISTIPP

Darbo su resursais primityvai:
KURTIR, NAIKINTIR, PRASYTIR, ATLAISVINTIR

Šie primityvai proceso būsenas keičia taip:



Proceso dalis, dirbanti su primityvais ir bendrais kintamaisiais – kritinė sekcija. Ją vienu metu vykdyti gali tik vienas procesas. Kritinės sekcijos apsauga užtikrinama semaforais.

Resursų paskirstytojai:

Statinis rinkinys OS paprogramių, skirstančių resursus procesams, sujungtų į bloką. Resursas priskiriamas anksčiausiai jo paprašiusiam procesui, tai yra pirmajam iš resurso laukiančių procesų eilės. Prioritetų nėra, kadangi laikoma, jog modelis nebus labai apkrautas.

Pranešimų paskirstytojas
Įvedimo srauto paskirstytojas
OA paskirstytojas
Supervizorinės atminties paskirstytojas
Išorinės atminties paskirstytojas
Užduoties superv. atminty paskirstytojas
Užduoties parametrų paskirstytojas

Užduoties programos teksto paskirstytojas
Užduoties duomenų paskirstytojas
Užduoties diske paskirstytojas
3 kanalo paskirstytojas
Vartotojo atminties paskirstytojas
Pakrovimo paketo paskirstytojas
Pertraukimo įvykio paskirstytojas
Eilutės superviz. atmintyje paskirstytojas
2 kanalo paskirstytojas
Planuotojas

Planuotojas – tai procesoriaus resurso paskirstytojas. Jis dirba su branduolio primityvais, pasiruošusių procesų sąrašu, procesų deskriptoriais bei procesoriaus resurso deskriptoriumi.

Planuotojo veikimo schema: