Operativni Sistemi - Uvod

Veljko Petrović

Februar, 2024

Administrativni detalji

Kako se organizuje predmet

O predavaču

- Veljko Petrović
- pveljko@uns.ac.rs
- NTP 330

- Preliminarni termin konsultacija: Četvrtak 14:30, u kancelariji.
- **Uvek** je dobra ideja da se najavite za konsultacije.
- Možemo se dogovoriti i oko dodatnih termina za konsultacije u zavisnosti od mog i vašeg rasporeda i obaveza.
- Na predmetu učestvuju i prof. dr Dinu Dragan i prof. dr Dušan Gajić, koga bi trebalo da se sećate.

Asistenti

- Anja Delić
- Dunja Gojković
- Dane Milišić
- Branislav Ristić
- Radovan Turović
- Jovana Jovanović

Obaveštenja

- Jedino mesto gde možemo pouzdano da vam objavimo podatke je ACS sajt.
- Naročito važne su vesti o predmetu dostupne na http://www.acs.uns.ac.rs/sr/os
- Dok niste gotovi sa predmetom proveravajte ovo često da bi bili sigurni da ne propustite važne informacije.

Materijali

- Operativni Sistemi, Problemi i Struktura prof. Hajdukovića
- Sav materijal će biti dostpuan na ACS sajtu.
- Nemojte koristiti materijal prošlih generacija: svake godine bude promena.
- Glavni izvor za predmet je, predvidivo udžbenik.
- Takođe važni su ovi slajdovi koji često sadrže modernizacije, i adaptacije materijala kao i modernije primere.

O slajdovima

- Slajdovi su dostupni i kao interaktivan sajt dostupan iz svakog pretraživača i kao PDF dostupan na ACS sajtu
- I sajt i PDF su generisani iz istog izvora i ekvivalentni su, sa tim da sajt može da sadrži i, npr. animacije.
- Slajdovi za svako predavanje će biti dostupni pred čas i nalaziće se na:
  - **Za interaktivnu verziju** https://pveljko-ftn.github.io/predavanja-OS/01/slides.html gde se 01 menja sa brojem predavanja.
  - **Za PDF verziju** http://www.acs.uns.ac.rs/sr/node/237/4408649

O predavanjima

- Periodično se pušta papir.
- Prisustvo je obavezno, formalno, ali mnogo bitnije prisustvo je *korisno.*
- Nikada nije pogrešno vreme da se postavi pitanje ili napravi komentar.

O vežbama

- Automatska i ručna evidencija pristupa
- Apsolutno obavezno prisustvo
- Samostalna izrada zadataka
- Apsolutno se očekuje da se spremite za izradu zadataka pre svake vežbe
- Počinju od ove nedelje.
- Morate imati spremno okruženje za rad.
- Više o tome šta je 'okruženje za rad' kasnije danas.

Formiranje ocene

Broj bodova	Ocena
51-60	6
61-70	7
71-80	8
81-90	9
91-100	10

Odakle bodovi?

- Predispitne obaveze (do 70 bodova)
- Ispitne obaveze (do 30 bodova)

Predispitne obaveze

Obaveza	Opis	Bodovi
Test T1234	Konkurentno programiranje	do 40
Složeni oblik vežbi SOV	Konkurentno programiranje (konkurentni problem)	do 30

Predispitne obaveze

- Oba testa se rade na vežbama
- Imate *samo jednu priliku* da ih uradite tokom nastave, ali imamo pravo, za sada, da ponovimo predispitne obaveze
- To ponavljanje će biti posle kraja semestra ali mora biti pre granice za davanje bodova: dakle biće tokom leta.
- Ponavljanje se mora prijaviti i plaća se: to nije naša ideja niti naša želja, proceduralno je obavezno.

Kako položiti?

- Prolaznu ocenu možete imati samo ako važi svaki od sledećih uslova:
- T1234 + SOV ≥ 36
- Ispit ≥ 16
- T1234 + SOV + Ispit ≥ 51

Kako pasti?

- Ako imate manje od 36 bodova sa predispitnih obaveza onda su svi bodovi koje imate nevažeći i morate predmet slušati opet iduće godine. Nećete dobiti potpis.

- Nemojte dozvoliti da ovo budete vi, **molim vas.**
- Stvarno, ali stvarno je bitno da steknete svojih 36 bodova. Nećete imati previše šansi da to učinite.

Ispit

- Održava se u ispitnom roku
- Nosi najviše 30 bodova
- Namenjen je isključivo studentima koji na predispitnim obavezama imaju barem 36 bodova
- Integralni ispit obuhvata celo gradivo.
- Mora se prijaviti ispit.
- Radi se na papiru, ima četiri pitanja i traje 60 minuta.

Struktura ispita

- Prvo pitanje nosi 6 bodova, odnosi se na celo gradivo i biće bazirano na principu zaokruživanja. Ovo pitanje će uvek biti bazirano na listi pitanja koja ću objaviti, ali neće odgovarati tačno nijednom pitanju, zato što ona nisu na zaokruživanje.
- Drugo pitanje nosi 10 bodova, odgovara se u okviru jednog do dva pasusa i mora biti sa liste pitanja koja ću objaviti.
- Treće i četvrto pitanje nose zajedno 14 bodova i to tipično, ali ne garantovano, po 7 bodova i potpuno su slobodne forme: odnose se na celokupno gradivo predmeta bilo sa predavanja, udžbenika, slajdova i zahtevaju da se mogu povezati različite činjenice naučene u okviru predmeta i da se o njima može rezonovati.

Šta neće nikada biti na ispitu

- Na ispitu nikada neće biti bilo koje pitanje koje očekuje da se kod uči napamet.
- Ako se, nekim slučajem, u pitanju pomene klasa ili neki algoritam, onda se isključivo misli na namenu te klase odnosno algoritma, na to kako radi i zašto postoji ali nikada i nikako koje su metode, šta tačno nasleđuje ili bilo šta slično.
- Ovo garancija se odnosi na sva četiri pitanja i na sve rokove, zauvek.

Primer kompletnog ispita sa odgovorima

- 1. Koja su validna stanja binarnog semafora?

```
a) **0**    b) **1**    c) 2    d) SLOBODAN
```

```
e) ZAUZET   f)-1     g) 3
```

Primer kompletnog ispita sa odgovorima

- 2. Šta je mrtva petlja?
- Mrtva petlja nastaje kada više niti/procesa pristupaju deljenim resursima koji su međusobno zavisni i kojima se pristupa u režimu isključivosti. Ono što ovi uslovi omogućavaju jeste da se procesi/niti dovedu u situaciju gde za svaku nit/proces važi da čeka neku drugu nit/proces i grafik ovih zavisnosti formira zatvoren ciklus. To znači da niti/procesi nisu aktivne no, su u istanju čekanja i iz tog stanja nikako ne mogu izaći: odatle naziv 'mrtva petlja.'

Primer kompletnog ispita sa odgovorima

- 3. Kakav format diska bi izabrali za particiju koja čuva video snimke sisitema za video-nadzor pod uslovom da želite maksimum iskorišćenja prostora. Obrazložiti vaš odgovor.
- Koristio bih nekakav fajl sistem koji koristi kontinualne datoteke. Ovo je dobra ideja zato što onda nikako ne bih imao blokove, pokazivače, ili bilo šta slično. Samo metapodatke u deskriptoru datoteke i dužinu datoteke praćenu sadržajem datoteke, redom. Ne bih morao da se brinem o produženju datoteke pošto se ovo ne bi dešavalo. Eksterna fragmentacija bi, takođe bila relativno mali problem, budući da bih imao fajlove predvidive dužine. Kada bih birao veličinu bloka za ovaj disk, birao bih veliki blok, budući da ne očekujem male datoteke uopšte.

Primer kompletnog ispita sa odgovorima

- 4. Ako bi imali pristup direktno disku (blokovima) koliko bi vam minimalno trebalo pristupa disku da promenite vlasništvo nekom fajlu? Obrazložite vaš odgovor.
- Odgovor zavisi od fajl sistema, i okolnosti. Ako, kao na predavanjima, koristimo ext2fs, onda treba sigurno da pročitamo superblok (1 čitanje), a onda iz njega da dobijemo tabelu inoda (1 čitanje), pa onda u njoj da nađemo sadržaj root direktorijuma (1 čitanje), pa u root direktorijumu nađemo broj inode nekog, bilo kog fajla, pa onda nađemo sadržaj te inode (1 čitanje), a u toj inodi, odnosno deskriptoru fajla se čuva tabela pristupa. Tako da je odgovor oko, minimalno, 4 čitanja i jedno pisanje.

- (Napomena: Molim vas, vodite računa da je ovaj odgovor malo neprecizan i preskače barem jedno čitanje (za tačan odgovor, vidite pretavanje 08.4) ali da je ključna stvar za ovo specifično pitanje da je proces razmišljanja i rezonovanja korektan.)

Prepisivanje

- Ako se utvrdi da je neko prepisivao ili koristio bilo kakva nedozvoljena sredstva na proveri znanja bilo koje vrste, preduzeće se mere predviđenje za to pravilnikom fakulteta.
- Molim vas, molim vas nemojte. Oko svega možemo da se dogovorimo, sve može da se adaptira vašim potrebama, ali kada je prepisivanje u pitanju niko od vaših nastavnika na ovom predmetu nema smisao za humor.

ChatGPT

- Nemojte ni ovo.
- Korišćenje bilo kakvog automatskog metoda generisanja koda se smatra prepisivanjem.
- Šta više, dok ima mnogih legitimnih svrha ovog softvera, ne preporučuje se nikako da ga koristite da radi vaš posao za vas: svrha zadataka koje dobijate je obrazovna, tj. nije svrha da se napravi rešenje (već imamo rešenja!) nego da vi prođete kroz proces razumevanja rešenja.

ChatGPT

- Naročito, *naročito* nemojte da koristite da preko ChatGPT odgovarate na ispitna pitanja pa da iz toga učite.
- Jako jako puno vaših kolega je palo zbog ovoga.
- Takođe nemojte da učite iz bilo kakve skripte ili bilo kog izvora koji ne dolazi od mene.

Potpis

- Potpis zahteva barem 36 bodova na predispitnim obavezama. Svako ko ima 36 bodova ili više na predispitnim ispitima, dobiće potpis.
- Nema drugih zahteva

Tehničko okruženje

Kako biti spreman za vežbe i praćenje primera sa predavanja

Tehničko okruženje

- Na ovom kursu se koristi Linux.
- Iako se primeri mogu kompajlirati na Windows-u ovo *nije preporučeno.*
- Zašto? Naleteće te na suptilne nekompatibilnosti koje su takve da niste dovoljno dobri programeri da ih otklonite.
- Možda jeste dovoljno dobri programeri, u kom slučaju, svaka čast.
- Ovo nije prva generacija: svako ko je koristio Windows u ovom kursu je naleteo na problem.
- Ako hoćete da koristite Windows i spremni ste da budete za to odgovorni, WSL je verovatno najbolji izbor.

Tehničko okruženje

- Od softvera se koristi GCC (kao kompajler), GNU make i Visual Studio Code kao editor. Možete editovati kod ma kojim editorom: to neće uticati na vašu ocenu.
- Možete da instalirate Linux na vaš računar kao jedan od operativnih sistema i koristite to. Ali, ako vam je primaran OS Windows i ne želite da instalirate nešto novo, može i tako. Onda morate raditi u virtuelnoj mašini.

Virtuelna mašina

- Trebaće vam VirtualBox https://www.virtualbox.org/ koji se instalira kao i svaka druga aplikacija. Dostupan je za svaki operativni sistem.
- Dalje, trebaće vam ISO linux distribucije. Ako želite verziju koja je ista onoj u većini FTN laboratorija, to bi trebalo da je Ubuntu 20.04 LTS.
- Proces instalacije je bezbolan i trebalo bi da ne stvara ozbiljne probleme.
- Molim vas uradite ovo što pre

Uvod

Vrtoglava tura operativnog sistema

Zadatak operativnog sistema

- Operativni sistem:

  - Objedinjuje raznorodne delove računara tako što upravlja procesorom, kontrolerima i RAM-om.
  - Skriva od korisnika detalje funkcionisanja tako što pretvara računar od mašine koja rukuje bitima, bajtima i blokovima u mašinu koja rukuje **datotekama** i **procesima.**

Što baš datoteke i procesi?

- Datoteka predstavlja apstrakciju nad sposobnošću računara da čuva i pristupa nekakvim proizvoljnim podacima.
- Procesi predstavljaju apstrakciju nad sposobnošću računara da izvršava nezavisne tokove operacija nad podacima.

Zašto nam treba operativni sistem?

- Sa tačke gledišta onoga što se zaista dešava na nivou hardvera, datoteka kojoj pristupate preko mreže, na hard disku magnetno-rotacionog tipa, i na SSD hard disku su *potpuno različite stvari.*
- To znači da ako pišete softver koji treba da radi sa sve tri varijante vi morate, onda, napisati poseban kod za sve tri mogućnosti, plus još i kod za fajlove koji se nalaze na optičkim medijima, plus još podršku za razni hardver koji se može koristiti da ostvari sve ove stvari.
- Ovako je nekada odista bilo.

Zašto nam treba operativni sistem

- Tako da je jedna (veoma bitna) funkcija operativnog sistema da omogućava da softver koji pišete može da radi sa prijatnim apsrakcijama koje su uniformne umesto da direktno priča sa hardverom.
- To što stoji između vas i hardvera znači da operativni sistem može da uradi još dve jako bitne stvari:
  - Raspoređuje resurse između procesa koji se oko njih mogu takmičiti i omogućava sinhronizaciju
  - Omogućava *bezbednost.*

Pojam datoteke

- Datoteka ima:
  - Sadržaj (korisničke podatke)
  - Atribute (npr. veličina ili vreme kreiranja) – u deskriptoru datoteke
- Uloga datoteke:
  - Trajno čuvanje podataka.

  - Pristup podacima je čitanje i pisanje (kojima predstoji otvaranje i nakon kojih sledi zatvaranje datoteke).
- Što datoteke zatvaramo?

Pojam procesa

- Aktivnost procesa – angažovanje procesora na izvršavanju korisničkog programa.
- Slika procesa – adresni prostor procesa (naredbe, stek i podaci).
- Atributi – stanje, prioritet – čuvaju se u deskriptoru procesa.

Stanje i prioritet procesa

- Tipična stanja procesa su:
  - aktivan
  - čeka
  - spreman.
- Prioritet procesa određuje kada je proces aktivan:
- Skoro uvek je aktivan proces (odnosno procesi) sa najvišim prioritetom.
- Ako postoji nekoliko procesa sa najvišim prioritetom, vrši se raspodela procesorskog vremena između njih uz pomoć dva mehanizma:
  - Raspoređivanje između sistemskih niti
  - Kvantum/kvant

Sistemske niti

- Moderni procesori tipično imaju sposobnost istinskog paralelnog izvršavanja, tj. mogu stvarno da rade stvari u paraleli
- Ako budete radili sa mikrokontrolerima, recimo, onda to obično nije dostupno
- Koliko jedan računar može da radi stvari u paraleli se zove broj *sistemskih niti.*
- Vama je ovo možda poznato kao broj ‘jezgara’ ali jedan računar može imati više procesora sa više jezgara a jedno individualno jezgro može raditi više poslova.

Kvantum/kvant

- Na mašinama koje nemaju više sistemskih niti ili na mašinama koje prekardaše svoj broj sistemskih niti (vrlo čest slučaj) ili na mašinama koje štede struju pa rade sa manje jezgara koristi se kvantum mehanizam

- Ovo znači se procesorsko vreme jedne sistemske niti deli između više programa gde se oni smenjuju, a svako dobija malecnu količinu vremena (poznatu kao kvant)
- Ko drži kvant se smenjuje tako brzo da ljudski operater neće primetiti da se operacije ne izvršavaju u paraleli.
- Isticanje kvantuma regulišu prekidi sata.

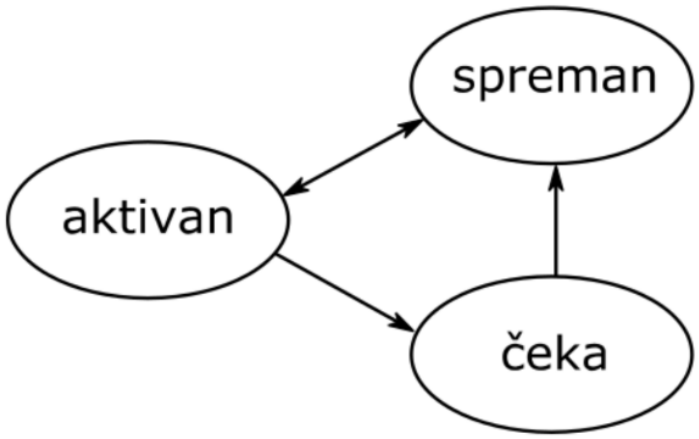
Terminološka zavrzlama

- Za potrebe ovog kursa ‘kvantum’ i ‘kvant’ je isto.
- Striktno kovoreći trebalo bi ‘kvant’ pošto je to adekvatan termin koji koristimo u fizici
- No, ‘kvantum’ vas podesća na taj termin na engleskom ‘quantum.’
- Biće puno engleskih termina na ovom kursu: to je neizbežno, engleski je *lingua franca* računarskih nauka.

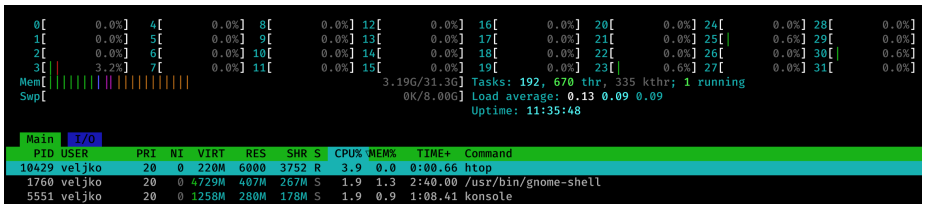
Stanje i prioritet procesa

- Aktivan proces prelazi u stanje “čeka” kada je nemoguć nastavak njegove aktivnosti (npr. UI radnja).
- Nakon tog čekanja prelazi u stanje “spreman”.

Stanje i prioritet procesa



Stanje i prioritet procesa, praktična ilustracija



Uloga procesa

- Procesi omogućuju bolje iskorišćenje računara (procesora) i njegovu bržu reakciju na dešavanje spoljnih događaja (npr. unos teksta).
- Istovremeno postojanje više procesa omogućuje da se procesor preključi sa aktivnog procesa na spreman proces kada aktivan proces prelazi u stanje “čeka”.
- Dobar primer ovakvog ponašanja je čekanje hitnog procesa na spoljni događaj (npr. unošenje teksta sa tastature).

Pojam niti

- Redosled naredbi programa (procesa) naziva se trag (trace) procesa.
- Proces je sekvencijalan ako je njegov trag poznat u vreme programiranja za dati ulaz.
- Trag sekvencijalnog procesa naziva se nit (thread) koja povezuje izvršavane naredbe u redosledu njihovog izvršavanja.

Mana sekvencijalnih procesa

- Mana sekvencijalnih procesa je da su neosetljivi na spoljne događaje (npr. editovanje teksta).
- Za editovanje su potrebne dve radnje (interakcija sa korisnikom i čuvanje unešenog teksta).
- Sekvencijalan editorski proces izvršava te dve radnje jednu za drugom:

Pseudokod 1

```
for(;;) {
  do_editor_command();
  if(time_to_save_data())
    save_data();
```

```
  }
}
```

Pseudokod nesekvencijalnog editorskog procesa

```
for(;;) {
      do_editor_command_in_foreground();
}
...
for(;;) {
     if(time_to_save_data())
     save_data_in_background();
}
```

Nesekvencijalan editorski proces (odvojene radnje)

- Prioritetnija radnja je posvećena interakciji sa korisnikom, a manje prioritetna pozadinska radnja je posvećena čuvanju teksta.
- Pod pretpostavkom da je hitna radnja zaustavljena, jer nema komandi od korisnika, pozadinska radnja može da se odvija sve dok, na primer, spoljni događaj poput pritiska dirke na tastaturi ne najavi početak interakcije sa korisnikom.

Nesekvencijalan editorski proces (odvojene radnje)

- Tada se zaustavlja pozadinska radnja, radi nastavljanja hitne radnje.
- Kada se obavi korisnička komanda u okviru hitne radnje, a hitna radnja se zaustavi u očekivanju nove komande, nastavlja se pozadinska radnja.
- Zahvaljujući preplitanju hitne i pozadinske radnje, u toku editiranja nema perioda bez odziva.
- Podrazumeva se da opisanom nesekvencijalnom editorskom procesu odgovaraju dve niti.

Nesekvencijalan editorski proces (odvojene radnje)

- Da bi opisano rukovanje nitima bilo moguće, prioritet, stanje i stek se ne vezuju za proces, nego za njegove niti.
- Znači svaka nit procesa ima svoj prioritet, svoje stanje, svoj stek, pa i svoj deskriptor.
- Za niti istog procesa se podrazumeva da nisu potpuno nezavisne, odnosno da sarađuju razmenom podataka.

- Tako, u slučaju nesekvencijalnog editorskog procesa, manje prioritetna nit se brine o čuvanju teksta koga pripremi prioritetna nit.

Konkurentni procesi

- Procesi sa više niti nazivaju se konkurentni procesi (konkurentni programi).
- U nekim okruženjima samo jedna od niti može biti “aktivna”, dok su ostale u stanju “spremna” ili “čeka”.
- U nekim (kao ono u kome mi pišemo) proizvoljan podskup niti može biti aktivan, ograničen samo mogućnostima računara.
- Preključivanje procesora sa jedne niti na drugu, uzrokuju redosled izvšavanja koji nije određen u vreme programiranja, što znači da je izvršavanje konkurentnih procesa u opštem slučaju stohastično zbog stohastične prirode dešavanja spoljnih događaja.

Terminološka zavrzlama

- Prvo, ‘konkurentnost’ u ovom kontekstu nema *nikavke* veze sa tim kako program uspeva na tržištu: reči su slučajno iste.
- Drugo, možda neki od vas razmišljaju da ovi ‘konkurentni’ programi zvuče neobično mnogo kao *paralelni* programi.
- Ovo su veoma slični termini ali se razlikuju suptilno.

Konkurentnost vs. paralelnost

- Konkurentni programi su programi koji imaju različite zadatke (niti) koje se izvršavaju sa preklapanjem u nekom periodu vremena.
- Paralelni programi imaju različite zadatke (niti) koje se izvršavaju jednovremeno u istom fizičkom trenutku. Paralelnost zahteva hardver koji to podržava, tj. podržava jednovremeno izvršavanje različitih poslova.
- Postoji razlika i u nameni, kako se terminologija koristi: konkurentni programi (tipično) izvršavaju više različitih zadatak sa preklapanjem u jednom periodu vremena. Paralelni programi (tipično) izvršavaju više delova jednog zadatka zbog ubrzanja.

Proces i nit

- Može da deluje zbunjujuće što imamo dva metoda za konkurentno programiranje

- Uprkos tome, proces i nit imaju odvojene funkcije iako su procesi, istorijski, korišćeni i da obezbede posao niti

Proces i nit

- Nit služi da u okviru *jednog procesa* omogući konkurentnost
- Tu je konkurentnost takva da između niti nema nikakve separacije: svaka nit ima pristup svim podacima svake niti za maksimalne performanse.
- Zato svaka nit ima svoj stek i svoj deskriptor, ali nema svoju sliku: to deli sa matičnim procesom.

Proces i nit

- Proces pruža konkurentnost i istorijski se ponekad koristio umesto niti, ali mu je glavna funkcija i osobina *separacija*
- Jedan proces ne može da 'zaviri' u drugi bez eksplicitne dozvole koja je vrlo kompleksna da se ostvari.
- Odnosno, kada se to desi, to bude bezbednosna mana i veliki problem.

Proces i nit

- Da li postoji razlog u modernom programu da se konkurentnost omogući kroz procese a ne kroz niti?
- Da! *Bezbednost.*
- Chrome izvršava svaki tab u posebnom procesu i to znači da je potencijalno maliciozan JS kod u jednom tab-u izolovan, višestruko, od nekog drugog tab-a gde je, recimo, vaš e-banking.

Struktura operativnog sistema

- Zadatak operativnog sistema je da upravlja fizičkim i logičkim delovima računara u okviru svog jezgra (kernela).
- Fizičkim delovima upravljaju moduli za rukovanje:
  - Procesorom
  - Kontrolerima
  - Radnom memorijom
- Logičkim delovima upravljaju moduli za rukovanje:
  - Datotekama
  - Procesima

Modul za rukovanje procesorom

- Zadatak ovog modula je preključivanje jedne niti na drugu.
- Moguće je preključivanje na niti u istom procesu ili u različitim procesima.
- Preključivanje procesora između niti istog procesa je brže nego preključivanje niti u okviru različitih procesa zato što se niti istog procesa nalaze u istom adresnom prostoru.
- Ovaj modul uvodi operaciju preključivanja.

Modul za rukovanje kontrolerima

- Zadatak ovog modula je upravljanje ulaznim i izlaznim uređajima koji su zakačeni za kontrolere.
- Modul se sastoji od niza komponenti nazvanih drajveri.

Drajveri

- Cilj drajvera jeste da uređaje predstavi u apstraktnom obliku sa jednoobraznim i pravilnim načinom korišćenja (primer drajvera diska).
- Drajveri uvode operacije ulaza i izlaza, u okviru kojih se rukuje preključivanjem (zaustavljanjem) niti koja je zatražila operaciju.
- Drajveri takođe rukuju i obradom prekida kao reakcijom na javljanje uređaja putem mehanizma prekida.

Modul za rukovanje radnom memorijom

- Zadatak ovog modula je da vodi evidenciju o slobodnoj radnoj memoriji radi zauzimanja i oslobađanja.
- U slučaju da podržava virtuelnu memoriju, ovaj modul se brine i o prebacivanju sadržaja između radne i masovne memorije.
- Ovaj modul uvodi operacije zauzimanja i oslobađanja.

Modul za rukovanje datotekama

- Zadatak modula za rukovanje datotekama je da omogući otvaranje i zatvaranje datoteka, kao i čitanje i pisanje njihovog sadržaja.
- Ovaj modul vodi evidenciju o blokovima (HDD/SSD) u kojima se nalaze sadržaji datoteka.
- Takođe ovaj modul vodi računa o prebacivanju sadržaja između radne i masovne memorije uz pomoću operacija čitanja (ulaza) i pisanja (izlaza), kao i bafera potrebnih za smeštanje sadržaja (modul za rukovanje memorijom).

- Pored ovog modul uvodi i operacije otvaranja i zatvaranja.

Modul za rukovanje procesima

- Zadatak ovog modula je da omogući stvaranje i uništavanje procesa, kao i stvaranje i uništavanje njihovih niti.
- Na ovaj način se uvodi višeprocesni i višenitni režim rada koji omogućava:
  - Bolje iskorišćenje procesora
  - Podršku većeg broja korisnika
  - Bržu reakciju na spoljne događaje
- Modul za rukovanje procesima poziva i operacije drugih modula (za upravljanje datotekama i memorijom).
- Modul za rukovanje procesima uvodi operacije stvaranja i uništavanja (procesa i niti).

Slojevit operativni sistem

modul za rukovanje procesima
modul za rukovanje datotekama
modul za rukovanje radnom memorijom
modul za rukovanje kontrolerima
modul za rukovanje procesorom

Slojeviti operativni sistem

- Za razliku od slojevitog operativnog sistema u praksi se uglavnom sreću monolitni operativni sistemi, koji nemaju hijerarjhijsku strukturu jer saradnja nije ograničena kao kod slojevitog OS.

Sistemski pozivi

- Svaki proces se nalazi u korisničkom sloju (gornjem sloju OS), i poseduje poseban adresni prostor koji se naziva korisnički prostor (user space).
- Operatvni sistem poseduje poseban adresni prostor koji se naziva sistemski prostor (kernel space).
- Zbog razdvojenosti ova dva prostora neophodno je uvođenje sistemskih poziva, radi poziva operacija OS.

Sistemski pozivi

- Sistemski pozivi zahtevaju korišćenje ASM naredbi i sakrivaju se unutar sistemskih potprograma (sistemskih operacija).
- Svaki proces u sistemskom prostoru ima svoj sistemski stek.
- Sistemski potprogrami obrazuju sistemsku biblioteku.

Sistemski pozivi

- Zahvaljujući sistemskim potprogramima, odnosno sistemskoj biblioteci, operativni sistem predstavlja deo korisničkog programa, iako za njega nije direktno linkovan.
- Istovremeno postojanje više procesa i nepredvidivost preključivanja, uzrokuje da je moguće da istovremeno postoji više procesa, koji su započeli, a nisu završili svoju aktivnost u okviru operativnog sistema, odnosno, čija aktivnost je zaustavljena unutar sistemskih operacija operativnog sistema.

Preplitanje izvršavanja tri sistemske operacije

Praktična ilustracija sistemskih poziva

```c
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdio.h>
int main(){
  unsigned char buff[32];
  ssize_t count = 0;
  int fd = open('open.c', O_RDONLY);
  while(count = read(fd, buff, 31)){
    buff[count] = '\0';
    printf("%s", buff);
```

```
  }
  close(fd);
}
```

Interakcija korisnika i OS

- Komande komandnog jezika omogućavaju korišćenje OS na interaktivnom nivou.
- Za interpretiranje i preuzimanje komandi komandnog jezika zadužen je poseban proces iz korisničkog sloja koji se zove interpreter komandnog jezika (shell).
- Interpreter komandnog jezika koristi OS na programskom nivou, jer u toku svog rada poziva sistemske operacije.

Pitanja

Šta smo naučili?

Pitanja

- Koje poslove obavlja operativni sistem?
- Šta obuhvata pojam datoteke?
- Šta se nalazi u deskriptoru datoteke?
- Šta omogućuju datoteke?
- Šta obavezno prethodi čitanju i pisanju datoteke?
- Šta sledi iza čitanja i pisanja datoteke?

Pitanja

- Šta obuhvata pojam procesa?
- Šta se nalazi u deskriptoru procesa?
- Koja stanja procesa postoje?
- Kada je proces aktivan?
- Šta je kvantum?
- Šta je sistemska nit?
- Šta se dešava nakon isticanja kvantuma?
- Po kom kriteriju se uvek bira aktivan proces?

Pitanja

- Koji prelazi su mogući između stanja procesa?
- Koji prelazi nisu mogući između stanja procesa?
- Šta omogućuju procesi?
- Šta karakteriše sekvencijalni proces?

- Šta karakteriše konkurentni proces?
- Šta ima svaka nit konkurentnog procesa?
- Koje su razlike između procesa i niti?

Pitanja

- Koju operaciju uvodi modul za rukovanje procesorom?
- Po čemu se razlikuju preključivanja između niti istog procesa i preključivanja između niti raznih procesa?
- Koje operacije uvodi modul za rukovanje kontrolerima?
- Šta je cilj drajvera?
- Koje operacije uvodi modul za rukovanje radnom memorijom?
- Koje operacije poziva modul za rukovanje radnom memorijom kada podržava virtuelnu memoriju?

Pitanja

- Koje operacije uvodi modul za rukovanje datotekama?
- Koje operacije poziva modul za rukovanje datotekama?
- Šta omogućuju multiprocesing i multithreading?
- Koje operacije uvodi modul za rukovanje procesima?
- Koje operacije poziva modul za rukovanje procesima?
- Koje module sadrži slojeviti operativni sistem?

Pitanja

- Šta omogućuju sistemski pozivi?
- Koje adresne prostore podržava operativni sistem?
- Šta karakteriše interpreter komandnog jezika?
- Koji nivoi korišćenja operativnog sistema postoje?