Operativni sistemi: interna struktura i principi dizajna

Poglavlje 2
Pregled operativnih sistema

Sedmo izdanje

William Stallings

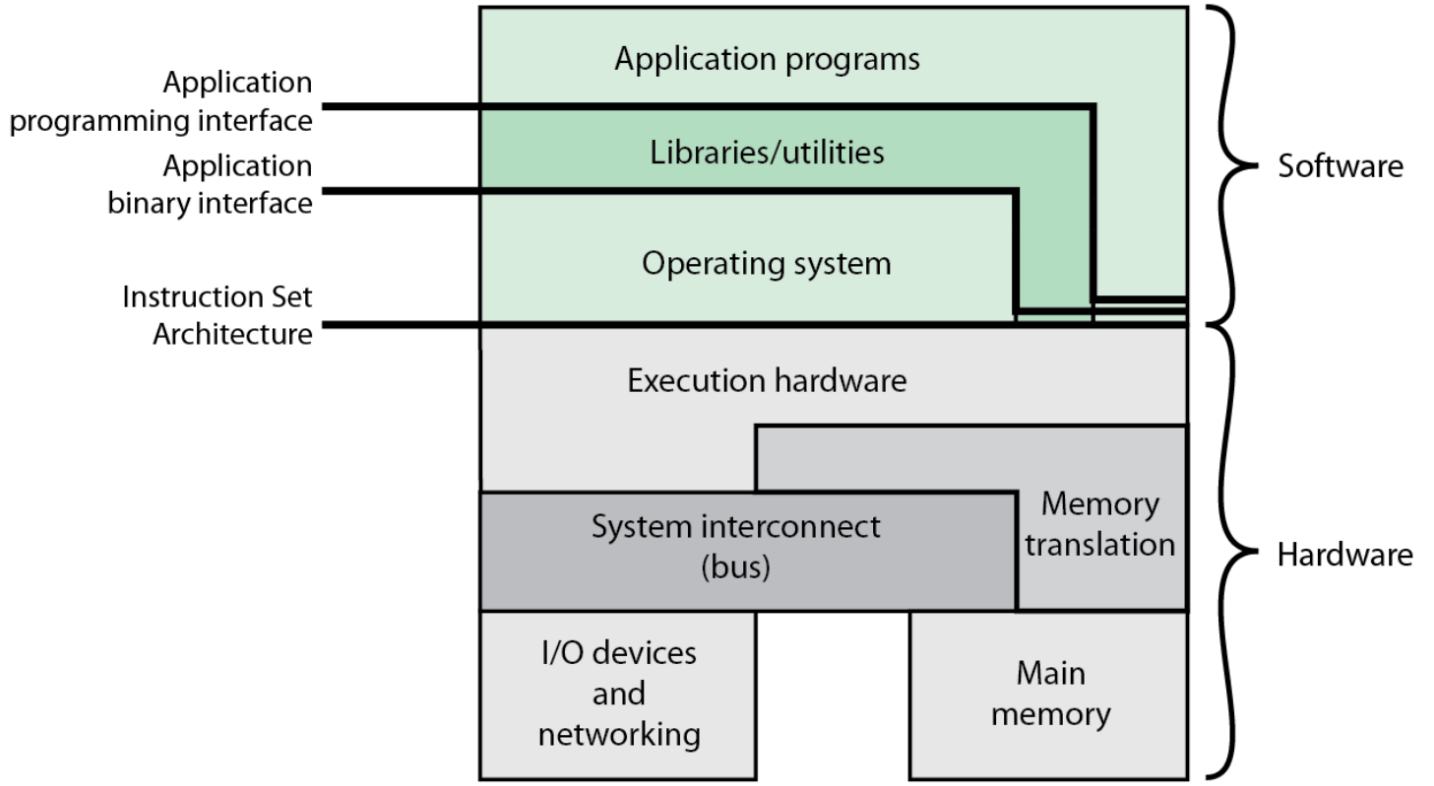
Operativni sistem

- Interfejs između aplikacija i računara
- Program koji upravlja izvršavanjem aplikativnih programa i dodelom sistemskih resursa

Glavni ciljevi OS-a:

- Pogodnost
- Efikasnost
- Mogućnost razvoja

Hardverska i softverska infrastruktura računara



Slika 2.1 Hardverska i softverska infrastruktura računara

Efikasnost: operativni sistem kao upravljač resursima

- Računar je skup resursa za prenos, skladištenje i obradu podataka
- OS je odgovoran za upravljanje ovim resursima
- OS ostvaruje kontrolu putem softvera



Operativni sistem kao softver



- Funkcioniše na isti način kao običan računarski softver
- Program ili skup programa koje izvršava procesor
- Često prepušta kontrolu i mora biti u stanju da je ponovo preuzme kako bi odlučio šta procesor sledeće treba da radi.

Operativni sistem kao upravljač resursima

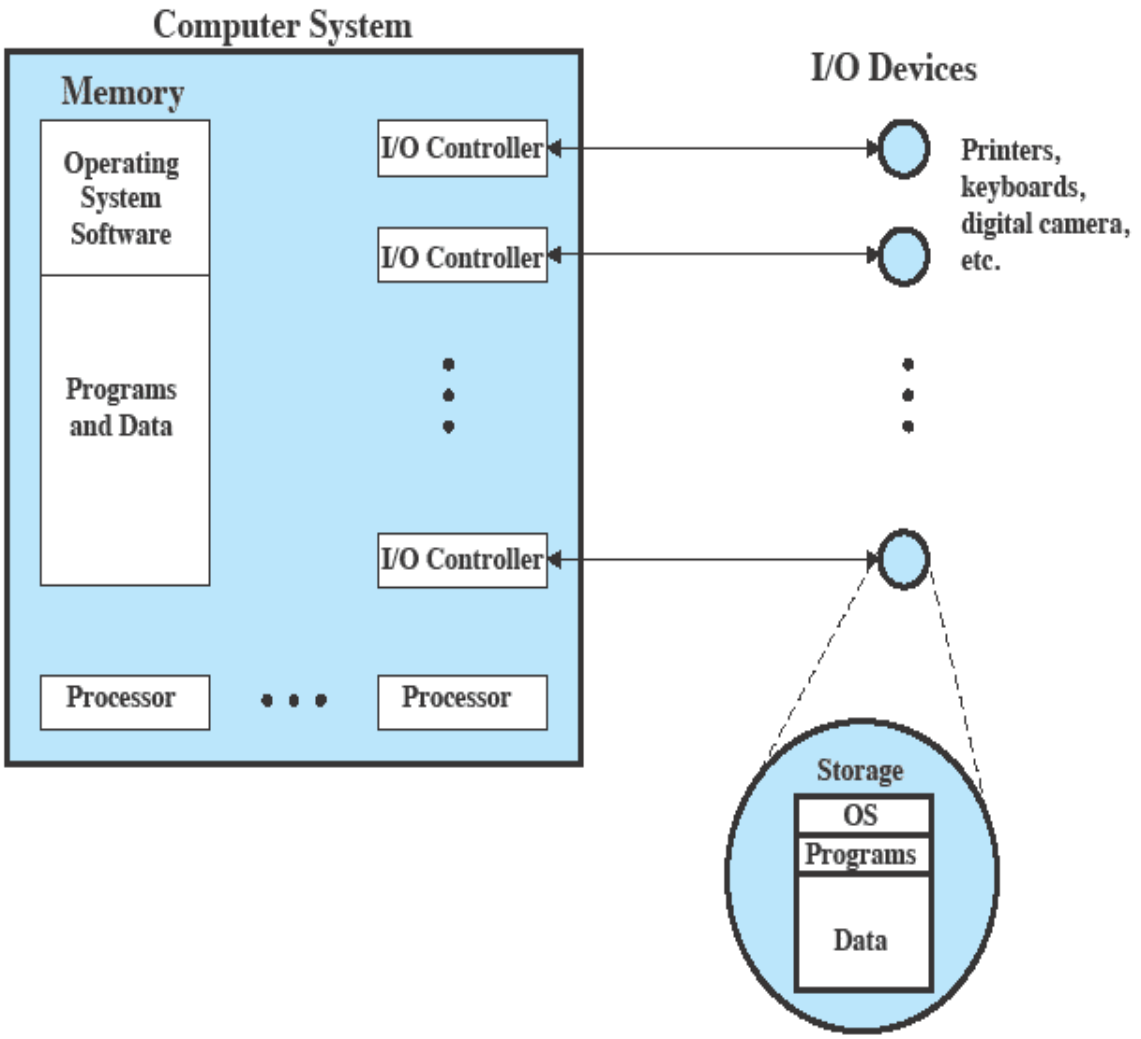
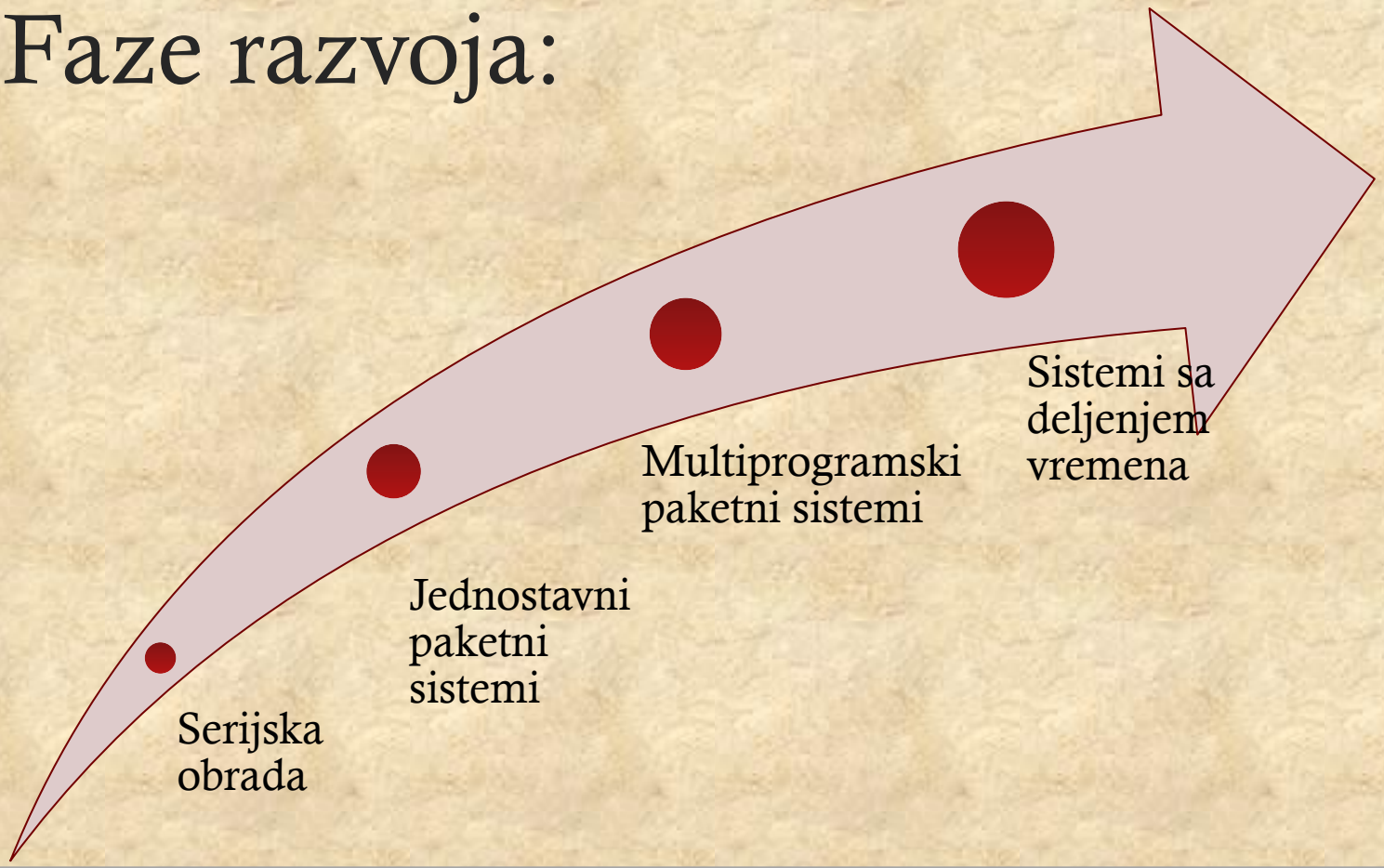
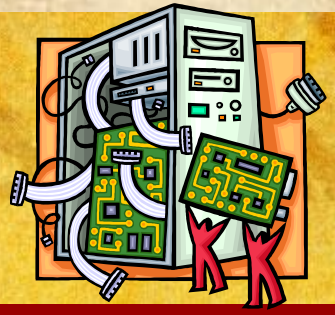


Figure 2.2 The Operating System as Resource Manager

Razvoj operativnih sistema

- Faze razvoja:





Serijska obrada

Najraniji računari:

- Bez operativnog sistema
  - programeri su radili direktno sa hardverom računara
- Računarima se upravljalo sa konzole koja je imala signalne lampice, prekidače, neki ulazni uređaj i štampač
- Korisnici su računaru pristupali „serijski“

Problemi:

- Raspoređivanje termina:
  - rad na računaru zakazivao se upisom na papirnu listu
    - dodeljeno vreme moglo je biti prekratko ili predugo, pa se vreme računara gubilo
- Vreme pripreme
  - znatno vreme trošilo se samo na pripremu programa za izvršavanje

Jednostavni paketni sistemi

- Rani računari su bili veoma skupi
  - bilo je važno maksimalno iskoristiti procesor
- Monitor (primitivni operativni sistem)
  - korisnik više nema direktan pristup procesoru
  - poslovi se predaju operateru računara, koji ih grupiše u pakete i stavlja na ulazni uređaj

Sa stanovišta monitora

- Monitor upravlja redosledom događaja
- *Rezidentni monitor* je softver koji je stalno u memoriji
- Monitor učitava posao i predaje mu kontrolu
- Posao vraća kontrolu monitoru

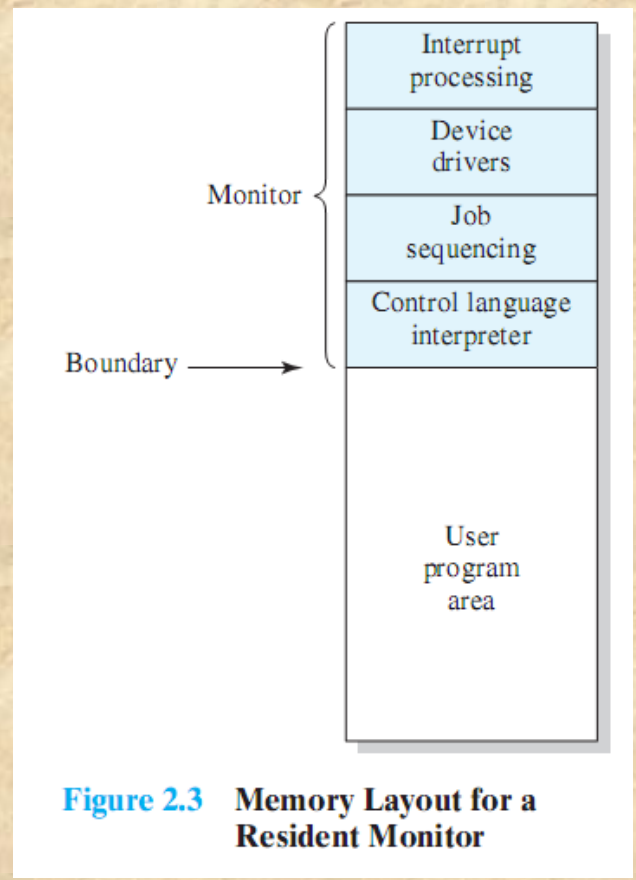


Figure 2.3 **Memory Layout for a Resident Monitor**

Poželjne hardverske karakteristike



Zaštita memorije monitora

- dok se korisnički program izvršava, ne sme da menja oblast memorije u kojoj je monitor

Tajmer

- sprečava da jedan posao monopolizuje sistem

Privilegovane instrukcije

- može ih izvršavati samo monitor

Prekidi

- daju OS-u veću fleksibilnost u upravljanju korisničkim programima

Režimi rada

Korisnički režim

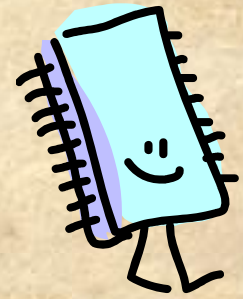
- korisnički program se izvršava u korisničkom režimu
- određene oblasti memorije su zaštićene od pristupa korisnika
- određene instrukcije se ne smeju izvršavati

Režim jezgra

- monitor se izvršava u režimu jezgra
- mogu se izvršavati privilegovane instrukcije
- može se pristupati zaštićenim oblastima memorije

Režijski troškovi jednostavnog paketnog sistema

- Procesor naizmenično izvršava korisničke programe i monitor
- Gubici:
  - deo glavne memorije sada se ustupa monitoru
  - deo procesorskog vremena troši monitor
- Uprkos režijskim troškovima, jednostavan paketni sistem poboljšava iskorišćenost računara.



Multiprogramski paketni sistemi

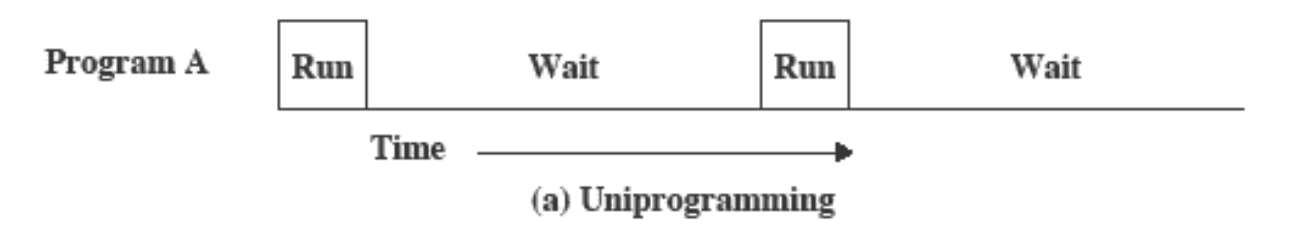
Read one record from file	15 μs
Execute 100 instructions	1 μs
Write one record to file	15 μs
TOTAL	31 μs

Percent CPU Utilization $= \frac{1}{31} = 0.032 = 3.2\%$

Figure 2.4 System Utilization Example

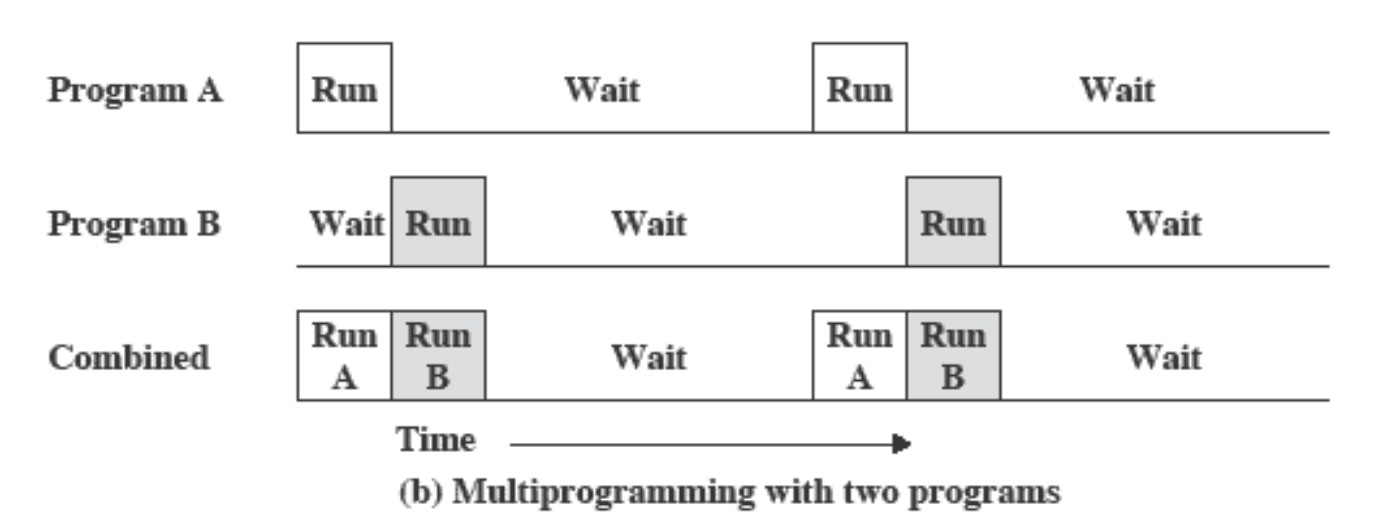
- Procesor je često neaktivan
  - čak i uz automatsko nizanje poslova
  - U/I uređaji su spori u poređenju s procesorom

Jednoprogramski rad



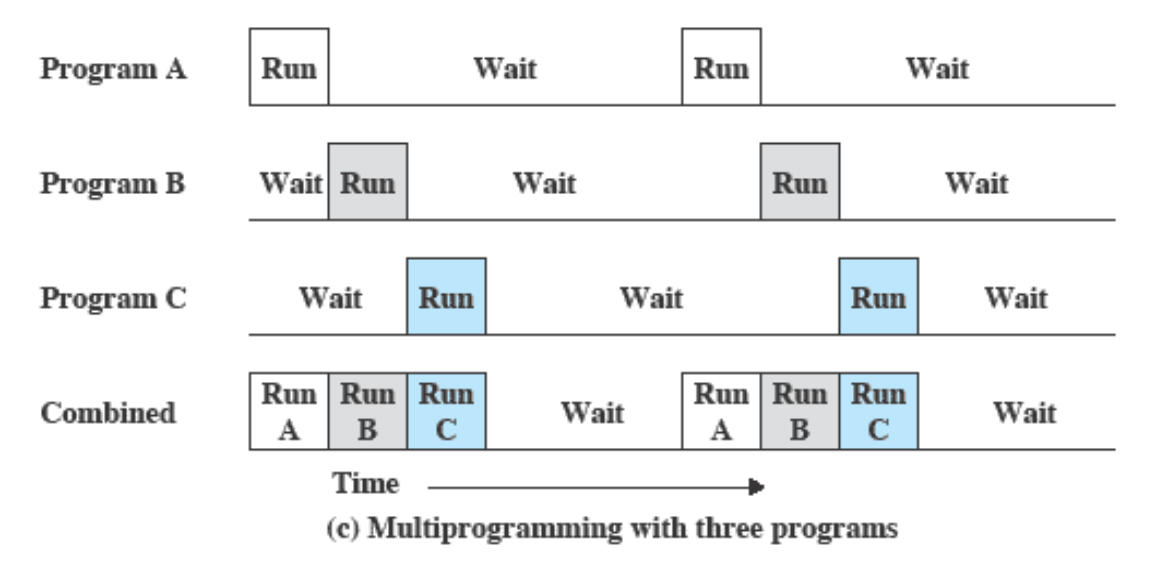
- Procesor neko vreme izvršava program dok ne naiđe na U/I instrukciju; tada mora da sačeka da se ta U/I instrukcija završi pre nego što nastavi

Multiprogramiranje



- Šta ako ima dovoljno memorije za OS (rezidentni monitor) i *dva* korisnička programa?
- Kada jedan posao mora da čeka na U/I, procesor može da pređe na drugi posao, koji možda ne čeka.

Multiprogramiranje



(c) Multiprogramming with three programs

- Multiprogramiranje
  - poznato i kao multitasking
  - memorija se proširuje tako da primi tri, četiri ili više programa, između kojih se procesor prebacuje

Primer multiprogramiranja

Table 2.1 Sample Program Execution Attributes

	JOB1	JOB2	JOB3
Type of job	Heavy compute	Heavy I/O	Heavy I/O
Duration	5 min	15 min	10 min
Memory required	50 M	100 M	75 M
Need disk?	No	No	Yes
Need terminal?	No	Yes	No
Need printer?	No	No	Yes

Uticaj na iskorišćenost resursa

	Uniprogramming	Multiprogramming
Processor use	20%	40%
Memory use	33%	67%
Disk use	33%	67%
Printer use	33%	67%
Elapsed time	30 min	15 min
Throughput	6 jobs/hr	12 jobs/hr
Mean response time	18 min	10 min

Tabela 2.2 Uticaj multiprogramiranja na iskorišćenost resursa

Sistemi sa deljenjem vremena

- Mogu se koristiti za obradu više *interaktivnih* poslova
- Procesorsko vreme se deli između više korisnika
- Poreklo: više korisnika istovremeno pristupa sistemu preko terminala, a OS prepliće izvršavanje korisničkih programa u kratkim intervalima (kvantumima) izračunavanja

Paketno multiprogramiranje nasuprot deljenju vremena

	Paketno multiprogramiranje	Deljenje vremena
Osnovni cilj	Puno korišćenje procesora	Minimalno vreme odziva
Izvor direktiva operativnom sistemu	Upravljačke komande priložene uz posao	Komande unete na terminalu

Tabela 2.3 Paketno multiprogramiranje nasuprot deljenju vremena

Kompatibilni sistemi sa deljenjem vremena

CTSS

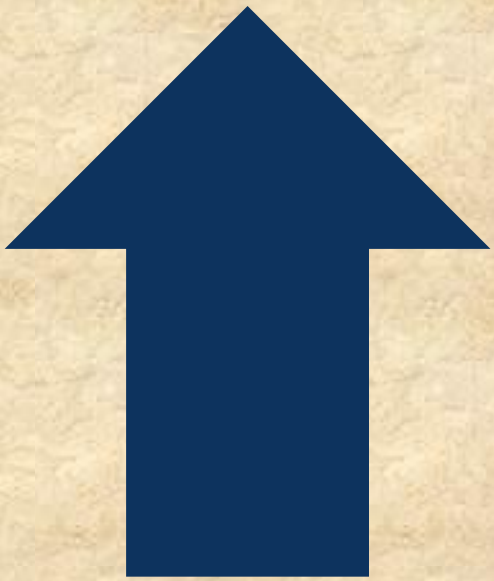
- Jedan od prvih operativnih sistema sa deljenjem vremena
- Razvila ga je na MIT-u grupa poznata kao Project MAC, za IBM 709/7094
- Radio je na računaru sa 32,000 *36*-bitnih reči glavne memorije, od čega je rezidentni monitor zauzimao 5000
- Da bi se pojednostavili i monitor i upravljanje memorijom, program se uvek učitavao od lokacije 5000-^{te} reči

Vremenski kvantumi

- Sistemski sat generiše prekide približno svakih 0.2 sekunde
- Pri svakom prekidu OS je preuzimao kontrolu i mogao je da dodeli procesor drugom korisniku
- U pravilnim vremenskim intervalima tekući korisnik je prekidan, a učitavan je drugi korisnik
- Programi i podaci prethodnog korisnika upisivani su na disk
- Kod i podaci programa prethodnog korisnika vraćani su u glavnu memoriju kada bi taj program ponovo došao na red

Glavna dostignuća

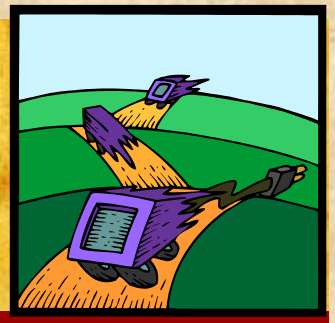
- Operativni sistemi su među najsloženijim softverskim sistemima ikada razvijenim



Glavna dostignuća u razvoju:

- Procesi
- Upravljanje memorijom
- Zaštita i bezbednost informacija
- Raspoređivanje i upravljanje resursima
- Struktura sistema

Proces



- Osnovni pojam u strukturi operativnih sistema

Proces se može definisati kao:

- program u izvršavanju
- instanca programa koji se izvršava
- entitet koji se može dodeliti procesoru i izvršavati na njemu
- jedinica aktivnosti koju karakterišu jedna sekvencijalna nit izvršavanja, tekuće stanje i pridruženi skup sistemskih resursa

Komponente procesa

- Proces se sastoji od tri komponente:
  - izvršni program
  - pridruženi podaci potrebni programu (promenljive, radni prostor, baferi itd.)
  - kontekst izvršavanja programa (ili „stanje procesa“)



- Kontekst izvršavanja je ključan:
  - to su interni podaci pomoću kojih OS nadgleda i kontroliše proces
  - obuhvata sadržaj raznih registara procesora
  - obuhvata informacije kao što su prioritet procesa i da li proces čeka na završetak određenog U/I događaja

Upravljanje procesima

- Celokupno stanje procesa u svakom trenutku sadržano je u njegovom kontekstu
- Nove funkcionalnosti mogu se projektovati i ugraditi u OS proširivanjem konteksta svim novim informacijama potrebnim za njihovu podršku

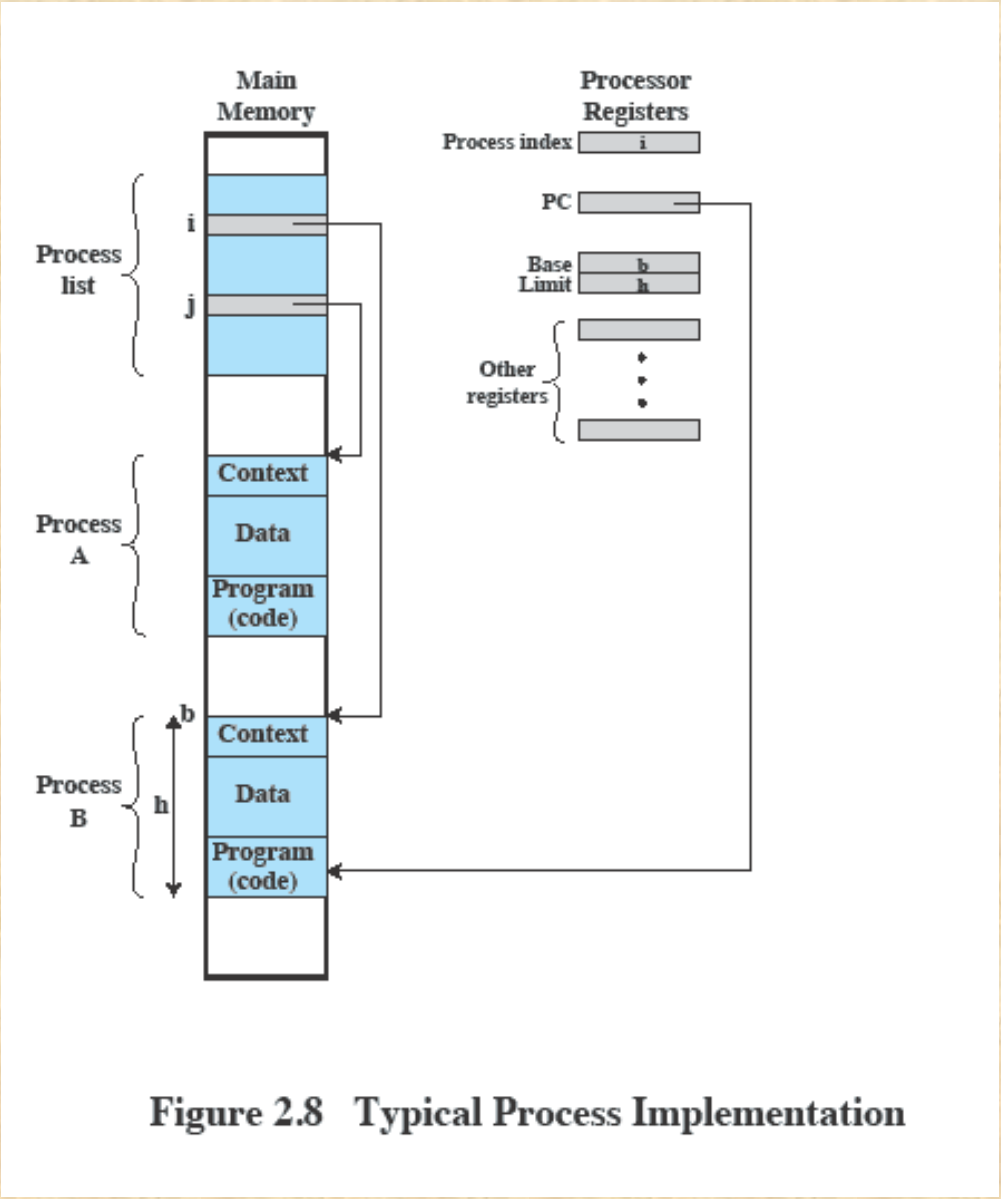


Figure 2.8 Typical Process Implementation

Višenitnost

- Tehnika u kojoj se proces koji izvršava aplikaciju deli na niti koje se mogu izvršavati konkurentno

Nit

- jedinica posla koja se dodeljuje procesoru
- obuhvata kontekst procesora i sopstvenu oblast podataka koja omogućava pozive potprograma
- izvršava se sekvencijalno i može se prekinuti

Proces

- skup jedne ili više niti i pridruženih sistemskih resursa
- programer ima veću kontrolu nad modularnošću aplikacije i vremenskim usklađivanjem događaja u aplikaciji

Simetrično multiprocesiranje (SMP)

- Termin koji označava hardversku arhitekturu računara, ali i ponašanje OS-a koje tu arhitekturu koristi
- Više procesa može da se izvršava paralelno
- Postojanje više procesora je transparentno za korisnika
  - ovi procesori dele istu glavnu memoriju i U/I resurse
  - svi procesori mogu da obavljaju iste funkcije
- OS se stara o raspoređivanju niti ili procesa na pojedinačne procesore i o sinhronizaciji između procesora

Prednosti SMP-a

Performanse	više procesa može da se izvršava istovremeno, svaki na drugom procesoru
Dostupnost	otkaz jednog procesora ne zaustavlja sistem
Postepeni rast	performanse sistema mogu se poboljšati dodavanjem još jednog procesora
Skaliranje	proizvođači mogu da ponude niz proizvoda koji se razlikuju po broju procesora u sistemu

Multiprogramm

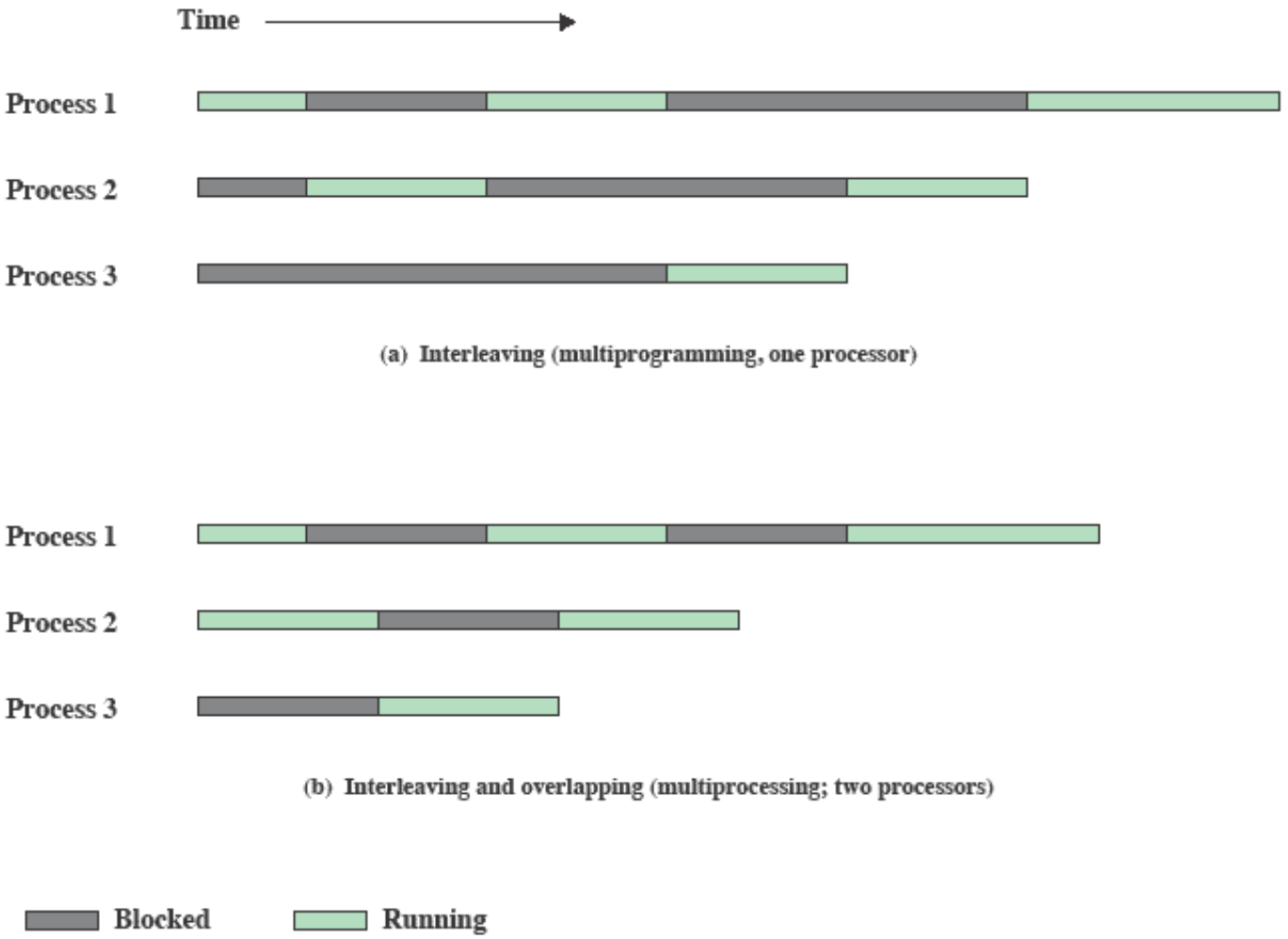


Figure 2.12 Multiprogramming and Multiprocessing

Virtuelne mašine i virtuelizacija

- Virtuelizacija
  - omogućava da jedan PC ili server istovremeno izvršava više operativnih sistema ili više sesija jednog OS-a
  - Svaki (gostujući) operativni sistem radi u **virtuelnoj mašini (VM)** i može da izvršava više aplikacija
  - Gostujući operativni sistemi rade kao da direktno komuniciraju sa hardverom, ali zapravo komuniciraju sa monitorom virtuelnih mašina (VMM), koji radi direktno na hardveru ili na operativnom sistemu **domaćina**



Koncept virtuelne mašine

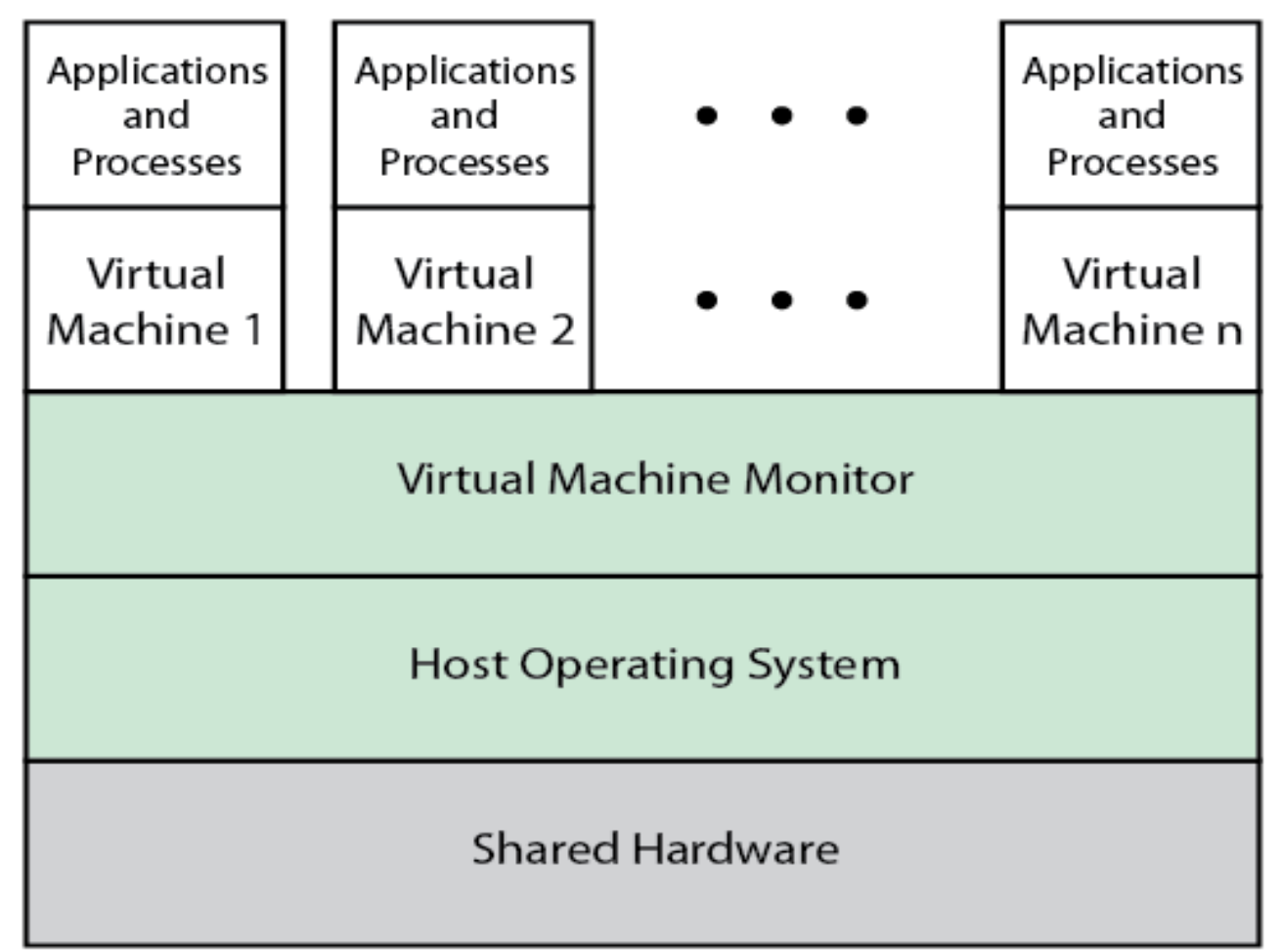


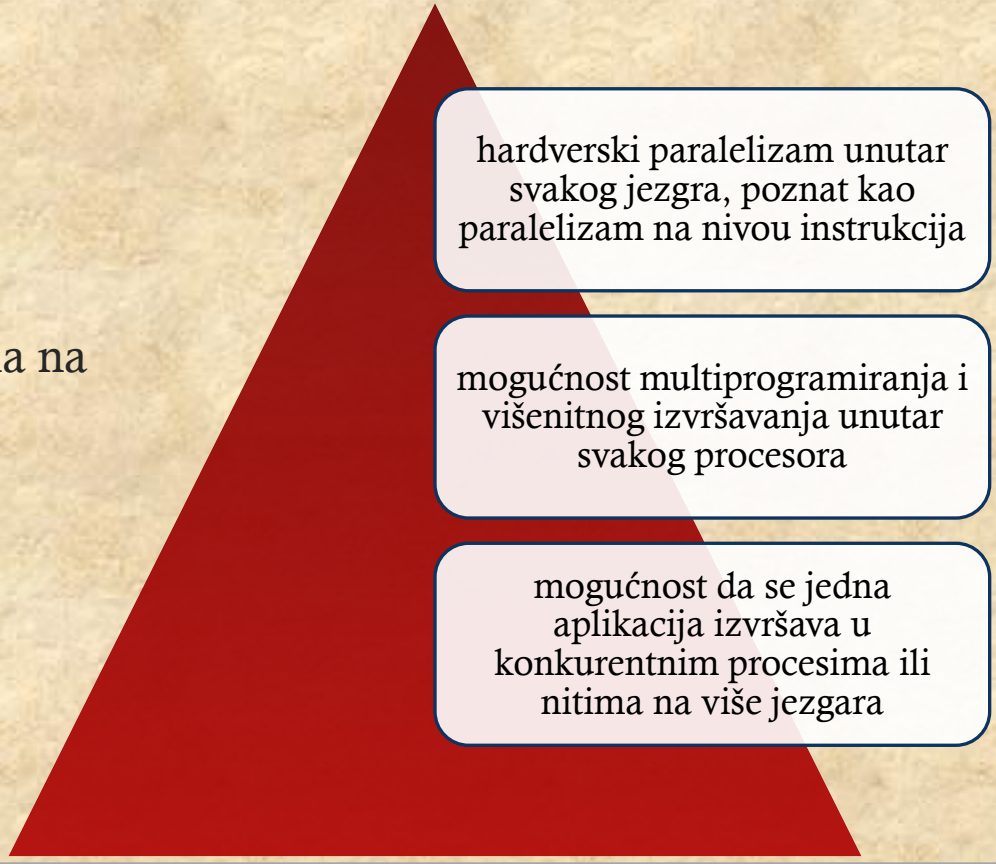
Figure 2.13 Virtual Memory Concept

Napomena: U nekim slučajevima, posebno kod servera, VMM radi direktno na hardveru. Ova slika prikazuje virtuelnu mašinu *na domaćinu*

Ispravka: treba da stoji Virtual MACHINE Concept

OS za višejezgarne sisteme

- Izazov pri projektovanju višejezgarnog sistema sa mnogo jezgara je da se efikasno iskoristi njegova procesorska snaga i da se inteligentno i efikasno upravlja znatnim resursima na čipu
- Potencijal za paralelizam postoji na tri nivoa:



Pregled sistema Windows

- **MS-DOS 1.0** objavljen 1981.
  - 4000 linija izvornog koda na asembleru
  - radio u 8 KB memorije
  - koristio mikroprocesor Intel 8086
- **Windows 3.0** izašao 1990.
  - 16-bitni
  - grafički interfejs (GUI)
  - implementiran kao sloj iznad MS-DOS-a
- **Windows 95**
  - 32-bitna verzija
  - doveo do razvoja sistema Windows 98 i Windows Me
- **Windows NT (3.1)** objavljen 1993.
  - 32-bitni OS sa podrškom za starije DOS i Windows aplikacije, kao i za OS/2
- **Windows 2000**
  - sadržao servise i funkcije za podršku distribuiranoj obradi
  - Active Directory
  - mogućnosti plug-and-play i upravljanja napajanjem
- **Windows XP** objavljen 2001.
  - cilj je bio da OS zasnovan na NT-u zameni verzije zasnovane na MS-DOS-u
- **Windows Vista** izašao 2007.
- **Windows Server** objavljen 2008.
- **Windows 7** izašao 2009. godine, kao i **Windows Server 2008 R2**
- **Windows Azure**
  - namenjen računarstvu u oblaku

Niti i SMP

- Dve važne karakteristike sistema Windows su podrška za niti i za simetrično multiprocesiranje (SMP)
  - Rutine OS-a mogu se izvršavati na bilo kom raspoloživom procesoru, a različite rutine mogu se istovremeno izvršavati na različitim procesorima
  - Windows podržava korišćenje više niti izvršavanja unutar jednog procesa. Više niti istog procesa može se istovremeno izvršavati na različitim procesorima
  - Serverski procesi mogu koristiti više niti da bi istovremeno obrađivali zahteve više klijenata
  - Windows obezbeđuje mehanizme za deljenje podataka i resursa između procesa i fleksibilne mogućnosti međuprocesne komunikacije

Tradicionalni UNIX sistemi

- Razvijeni su u Bell Labs-u, a 1970. godine proradili su na računaru PDP-7
- Preuzeli su mnoge ideje iz sistema Multics
- PDP-11 je bio prekretnica jer je prvi put pokazao da UNIX može biti OS za sve računare
- Sledeća prekretnica bilo je ponovno pisanje UNIX-a na programskom jeziku C
  - pokazalo je prednosti korišćenja jezika visokog nivoa za sistemski kod
- Prvi put je opisan u stručnom časopisu 1974. godine
- Prva široko dostupna verzija van Bell Labs-a bila je Verzija 6 iz 1976. godine
- Verzija 7, objavljena 1978. godine, predak je većine savremenih UNIX sistema
- Najvažniji sistem van AT&T-a bio je UNIX BSD (Berkeley Software Distribution)

Pregled sistema LINUX

- Nastao kao varijanta UNIX-a za IBM PC
- Linus Torvalds, finski student računarstva, napisao je prvu verziju
- Linux je prvi put objavljen na internetu 1991. godine
- Danas je to potpuno funkcionalan UNIX sistem koji radi na više platformi
- Besplatan je i izvorni kod je dostupan
- Ključ uspeha bila je dostupnost besplatnih softverskih paketa
- Izrazito modularan i lak za konfigurisanje

Modularno monolitno jezgro

- Većina UNIX sistema je monolitna (gotovo sva funkcionalnost OS-a nalazi se u jednom velikom bloku koda koji se izvršava kao jedan proces sa jednim adresnim prostorom)
- Sve funkcionalne komponente jezgra imaju pristup svim njegovim internim strukturama podataka i rutinama
- Linux to donekle poboljšava jer je strukturisan kao skup modula

Učitljivi moduli

- Relativno nezavisni blokovi
- Modul je objektna datoteka čiji se kod može povezati sa jezgrom i odvojiti od njega tokom izvršavanja
- Modul se izvršava u režimu jezgra u ime tekućeg procesa
- Imaju dve važne karakteristike:
  - Dinamičko povezivanje
  - Složivi moduli