

# OperativniSistemi

## II Semestar–2023/24–Vježbe

**Sedmica 1**

# HandoutzaVježbe

## Agenda:

* PAPP
* Turingova Mašina
* Arhitektura Osnovih Operativnih Sistema
* Sistemi za verzioniranje koda (VCS) – GIT
* Manuelna Kompilacija

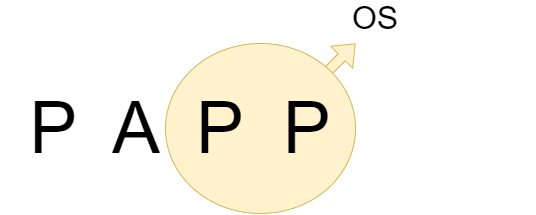
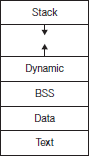
**Kontakt:**

(Asistenti na grupama)

### Problem – Algoritam – Program – Proces (PAPP)

PAPP model predstavlja vizuelnu reprezentaciju životnog ciklusa i misaonog procesa koji je potrebanza razumijevanje abstraktnih pojmova u svim poljima informacijskih tehnologija. Sve oblasti koje senalaze u oblasti Informacijskih tehnologija se mogu simulirati preko ovoga akronima. Oblastoperativnih sistema obuhvata ponašanje **programa** i njihovu transformaciju u **procese** te njihovu

sinhronizaciju i upotrebu sa stajališta inžinjera.

Kroz rad sa 

Programiranjem smo upoznati sa arhitekturom Programa, Nekakva osnovna vizuelna reprezentacijaprocesa koji možemo zamisliti u glavi se svodi trivializovan diagram poput ovoga:

### Turingova Mašina

Razumijevanje operativnih sistema se zasniva na razumijevanju interne strukture rada računara.Jedan od najboljih primjera za opisivanje rada računara je preko **Turingove Mašine** koja predstavljaabstraktni koncept generalnog modela računarske komputacije. Trivializovan pristup sa stajališta

Operativnih sistema opisuje ovu mašinu kroz nekoliko osnovnih komponenata:

* + Traka sa simbolima
  + Instrukcijski set
  + Glava koja vrši analizu

Traka je podijeljena u segmente na kojima se nalaze **simboli,** Glava se pozicionira iznad segmentatrake,registrujesimbol,pretražuje simboluInstrukcijskomsetu,provjeravastanjemašine(stanja

mašine su zadana u instrukcijskom setu), te glava djeluje po instrukcijama vezanim za simbol i stanjemašine zadanim u instrukcijskom setu ( pomjeranje lijevo ili desno na traci).

Ovaj model komputacije predstavlja generalni prikaza rada CPU i po logici izvršavanja nije ništadrugačiji od najmodernijih procesora na današnjem tržištu.

### Arhitektura Osnovnih Operativnih Sistema (LINUX)

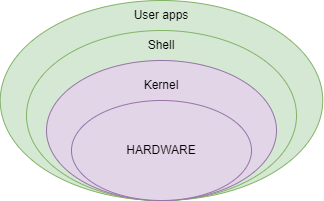
Močna turingova mašina daje vizuelni prikaz defincije pojmova programa i procesa.Nekakva osnovnadefinicija procesa jeste da je to „program koji se pokrene ili učita u obradivu memoriju“. Program

predstavlja naš kod koji ispišemo u bilo kojem programskom jeziku. Aktivacija njegova je ono što namdaje mogucnost da ga oslovljavamo kao **Proces**. Operativni sistemi generalno imaju 2 moda rada,a tosu: **Korisniči režim** i **privilegovani režim** ( User mode and Privilegued mode). Najlakša distrinkcija jeu tome što Korisnički dio nema pristup cijelom operativnom sistemu, dok privilegovani.Nekakav

tradicionalni način konceptualiziranja ovih koncepata je da svi programi koji pripadaju operativnomsistemu i koji su sklopu operativnog sistema te po ekstenziji CIJELI operativni sistem sam po sebi bezkorisnickih dodataka pripada Privliegovanom režimu rada, dok sve korisničke aplkacije se su ukorisničkom režimu rada.

Poznata riječ „KERNEL“ se odnosi na segment operativnog sistema koji je u **privilegovanom režimu.**

Makroskopska vizuelna apstrakcija arhitekture operativnog sistema se može prikazati na sljedećinačin:



Zelenom bojom je označen segmenat operativnog sistema koji je u **Korisničkom Režimu**, dok jeLjubičastom bojom označen segmenat operativnog sistema koji je u **Privilegovanom Režimu.**

Iz ovoga na slici može se intutivno zaključiti definicija Kernela kao **INTERFEJS** između korisničkogrežima rada i hardware-a računara. On se brine o drajverima, sistemskim pozivima dubokim

arhitekturama operativnih sistema koje bi nama kao Sistemskim inžinjerima bile previše komplekse zasvakodnevnu upotrebu.

### Sistemi za Verzioniranje koda (VCS)

Najbolji predstavnik VCS Sistema u današnjem vremenu je **GIT**. On omogučava stvari poput:

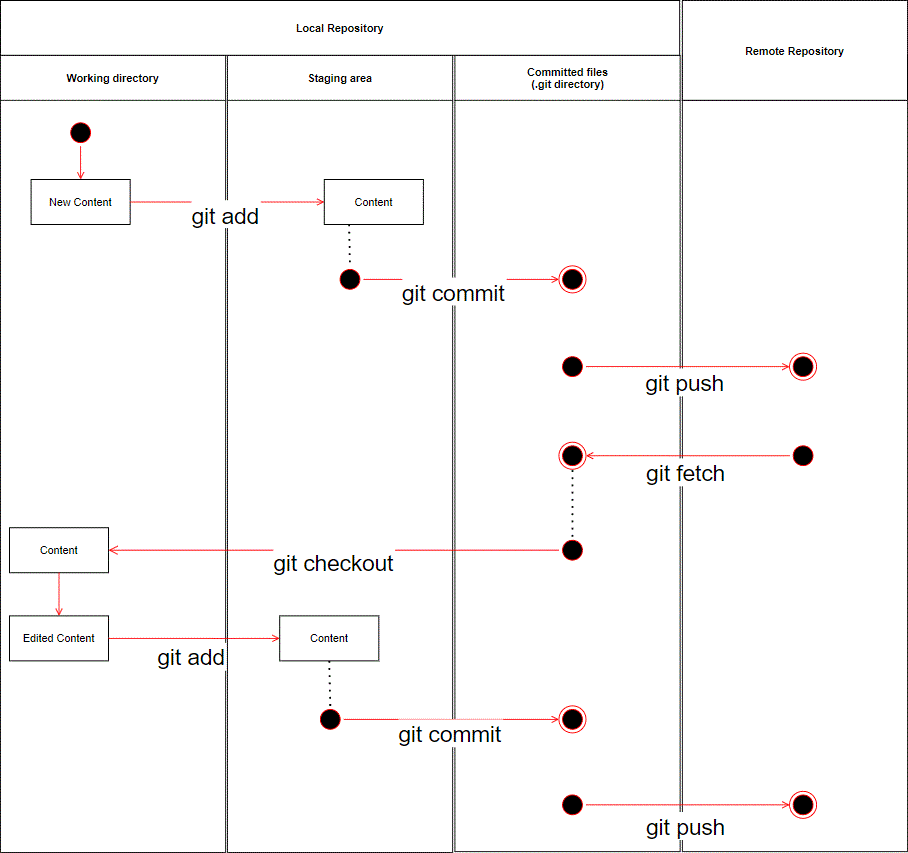
* + Sinhronizacije koda i napretka na projektima
  + Praćenje verzija programa
  + Protokol u slučaju nezgode ( Disaster recovery)
  + Može u ekstremnim situacijama zamijeniti sisteme za autorizaciju, te poslužiti kao RBAC( RoleBased Access Control)

Zasnovan je na principu rada sa repozitorijumima. Trivializovano . Repozitorij je jedinica mjere GIT-akoji služi kao spremnik za naš kod (analogno projektnim direktorijumima koje inačekoristimo).Komunikacija sa gitom se izvodi radom preko **shella** i koristi instacu shella pod nazivom**BASH (**Bourne Again Shell). Svaka komanda unutar Shella pocinje sa Git <command>. Spašavanje

progresa se radi pomoču sisteme pod nazivom „COMMIT“, sve promjene koje se „komitaju“ budu

spašene i dobiju jedinstveni short SHA256 kod kao jedinstveni identifikator koji nam može sluziti zavračanje koda. Svaki „komit“ zahtjeva poruku kao verbalni opis koji ide uz promjenu. Vizuelna

reprezentacija izgleda ovako:



Remote u zadanom dijagramu predstavlja repozitorij napravljen na internetu (CLOUD) na platformikoja vrši skladišetenje repozitorija poput GITHUB, BITBUCKET,GITLAB... Sinhronizacija izmeđju

remote-a i localnog repozitorija se vrši pomoču 2 komande GIT PUSH I GIT PULL koji dobavljajuposljednje komitove sa REMOTE-a i analogno šalju posljednji komit na REMOTE sa lokalnog

repozitorija.

### Manuelna Kompilacija

Da biste manuelno kompajlirali C++ kod, slijedite ove korake:

Otvorite uređivač teksta i napišite svoj C++ kod. Spremite ga sa ekstenzijom .cpp, na primjer,"myprogram.cpp".

Otvorite terminal i navigirajte do direktorija u kojem se nalazi vaš kod.

Kompajlirajte svoj kod koristeći C++ kompajler, kao što je g++. Na primjer, ako imate g++ kompajlerinstaliran na svom sistemu, možete kompajlirati svoj kod sljedećom naredbom:

* + - g++ myprogram.cpp -o myprogram

Ova naredba će kompajlirati vaš kod i stvoriti izvršnu datoteku nazvanu "myprogram" u istomdirektoriju.

Ako u vašem kodu postoje greške, kompajler će ih prikazati u terminalu. Morat ćete popraviti greškeprije nego što uspješno kompajlirate svoj kod.

Nakon što je vaš kod uspješno kompajliran, možete ga pokrenuti tako da u terminal upišete njegovoime, a zatim sve potrebne argumente. Na primjer, za pokretanje "myprogram" upisali biste:

* + - ./myprogram

Ovo će izvršiti vaš program i prikazati bilo kakav izlaz u terminalu.

Napomena: Tačne naredbe i sintaksa za kompajliranje C++ koda mogu varirati ovisno o vašemoperativnom sistemu i kompajleru koji koristite.

## Za sve eventualne primjedbe,komentare,sugestije obratiti se na mail:[email](mailto:adin.jahic2019@size.ba) predmetnog asistenta