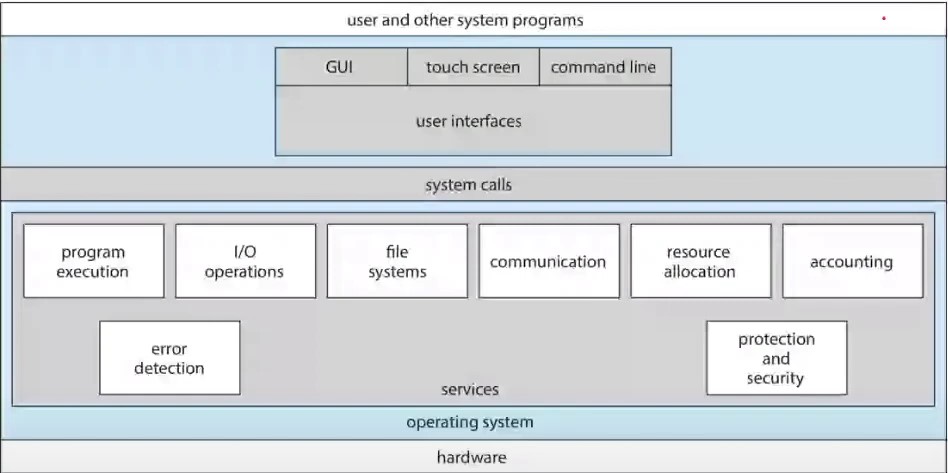
**Struktura e sistemeve operative**

Shërbimet kryesore ose obligative të cilat një sistem duhet ofruar janë : 1.*User Interface* – Secili OS ka një UI qoftë ai *Comand-Line-Interface*, *Graaphic-User-Interface, Touch-Screen*, kombinim etj. 2.*Ekzekutimi i programeve* – OS duhet të jetë në gjendje që të vendos programet në memorie dhe ti ekzekutoj ato qoftë në mënyrë normale ose abnormale (me errora). 3.*I/O operacionet* – OS duhet të disponojë një shërbim i cili mirret me pasije I/O. 4.*Manipulimi i Fajllave.* 5.*Komunikimi ­*– Midis dy a më shumë proceseve në të njëjtin kompjuter ose në kompjuter të lidhur në rrjetë për të ndërvepruar proceset duhet patjetër të komunikojnë në mes vete. 6.*Detektimi i Errorave* – OS duhet të ofrojë mekanizëm që merret me detektimin e errorave dhe për secilin error të ofrojë hapa se si të rregullohet ashtu që të sigorohet kompjutim kopitabil. OS gjithashtu ofron lehtësime në debugging. 7.*Logging* – Nënkupton evidentimin e aktivitetetve të përdoruesit karrshi resurseve kompjuterike. 8.*Protection & Security* – Kontrollimi i qasjes dhe mbrojtia nga rreziqet.

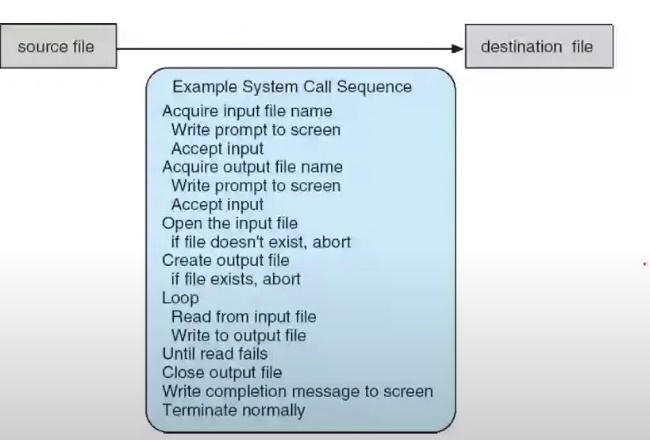


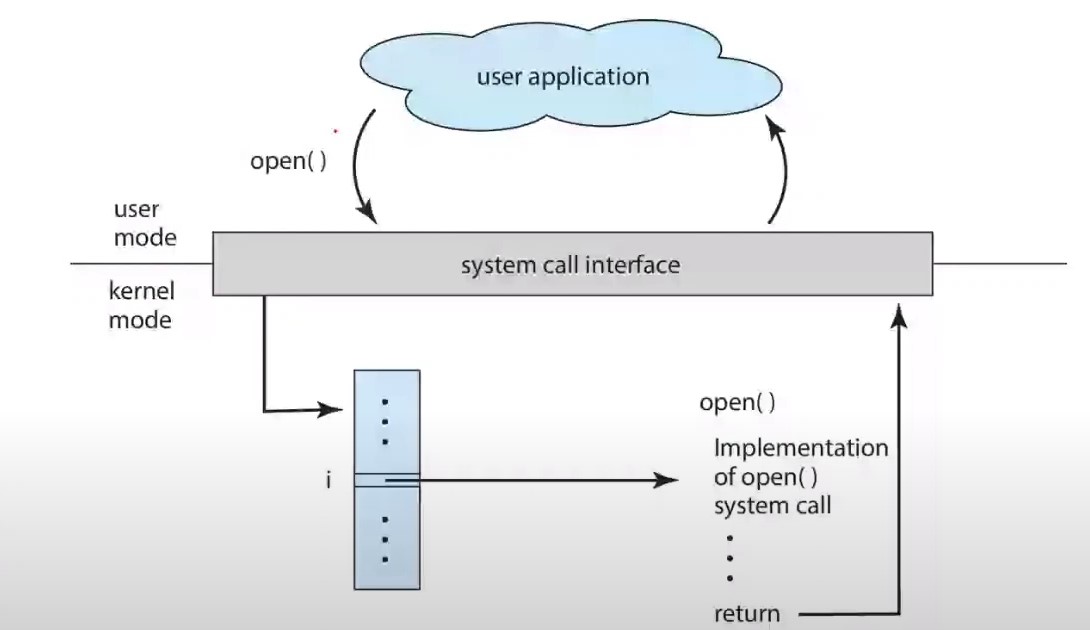
*Konzolla ku shkruhen komandat quhet shell*.

*Xerox ka krijuar gui e parë*.

Për të funksionalizuar Shërbimet që ofron OS përdoren *System Calls*. *System Calls* janë “hapa” që duhet kryer për të fituar nga shërbimi i OS ose për ta funksionalizuar atë.

System calls zakonisht janë të shkruajtura në gjuhtë si C/C++, dhe përdoren me anë të API. Ekzistojnë 3 API kryesorë për përdorimin e System Calls : *Win32, POSIX, Java API*.

Shembull : Përdorimi i shërbimit të manipulimit me fajlla për të kompjuar përmbajtien e një fajlli në një fajll tjetër. Mund të shohim numrin e madhë të System calls që duhet përdorur ashtu që të funksionalizohet një shërbim i thjeshtë i OS si kopjimi i përmbajties. Për me lehtësu këtë punë dhe mos me pasë nevoj mbajtien mend të gjitha këtyre System calls përdoret API, që në këtë rast një gjë e tillë mund të kryhej vetëm me thirrje të funksionit të API copy().

Shembull : Përdorimi i System calls me anë të API për të hapur një fajll.  Pa pasur nevoj të shkruajm System Calls, me anë të API i cili e bën konvertimin e thirrjes open() në System Calls e nevojshëm për të ekzekutar shërbimin e kërkuar në këtë rast hapjes së fajllit.

Mund të shfaqet nevoj që të dërgohet edhe parametra në thirrjet e API, për ta bërë këtë tri qasje : *Duke përodrur regjistrat, Duke përdorur memorien, Duke përdorur Stack*.

Qasje me memorie dhe me stack nuk e kufizon madhësinë e parametrave ose numrin e tyre ndërsa qasja me regjistra e kufzionë.

*System calls i karakterizojmë si* system calls për *:* 1.Kontrollin e proceseve, 2.Menagjimin e fajllave, 3.Menagjimin e paisjeve, 4.Mirëmbajtien e informacioneve, 5.Komunikim, 6.Protection.

*Komunikimi midis proceseve mund të bëhet me memorien e përbashkët ose me message passing*.

OS duhet të ketë *Programe të sistemit* për secilin shërbim që e ofron, gjdo *System program* është një API por jo gjdo API është System Program.

*Programet e sistemti/System Programs* që i ofron një OS mund ti ndajmë në : 1.File manipulation. 2.Status Information. 3.Program load and execute. 4.Communication. 5.Background Services(Daemon, subsystems). 6.Application Program. 7.Programming language support.

Ekzistojnë 4 arkitektura të Sistemeve operative : 1.*Arkitektura e thjeshtë* (Monotholike) – Shembull MSDOS, komplet OS llogaritet si një vargë, shtimi i gjërave të reja në të ose manipulimi i atyre ekzistuese është mangësi e madhe. 2.Ariktuektura shumë shtresore (Layered) : Ndahet OS në shtresa secila shtresë kujdeset për funksionimin e rregullt të nënshtresave të para.Për dallim nga ariktektura e thjeshtë në layered ndryshime mund të bëhen vetëm në shtresat pa pasur ndonjë problem ose pa ndryshuar gjith sistemin. Ka mangësi komunikimin midis shtresave të largëta. 3.*Arkitektura modulare* – Për ta mundësuar këtë lloj arkitekture na nevojtiet *mikrokernelli*. Mikrokernelli është nënjësie e kernellit që ofronë vetëm funksionet bazike, për të shtuar funksione të tjera duhet shtuar module në mikrokernell, pra për dallim nga layered dhe monothelikja shtimi i funksioneve të reja është i lejuar dhe komunikmi në mes moduleve është shumë më i mirë se tek shtreat layered. Mangësi mund të ketë kompleksitetin e implementimit. 4.*Hybrid* – Merrë të mirat nga 2 dhe 3, dhe përdoret nga shumica e sistemve operative moderne.