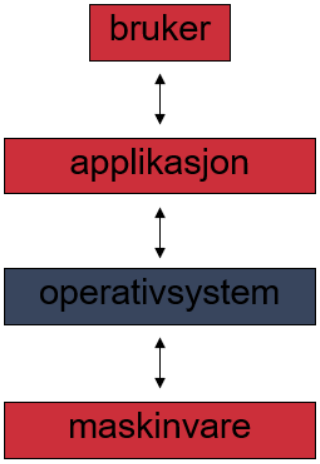
TK1100
FORELESNING 0x05 OS

1. OS

a) Forklar med egne ord hva et operativsystem er og hvorfor man trenger/ikke trenger et OS.

Her finnes det forskjellige svar da oppgaven sier at man skal forklare med egne ord. Det er viktig å få med at operativsystemer ligger mellom applikasjonen og maskinvaren, her fungerer operativsystemet som grensesnitt mellom applikasjonen og maskinvaren, for å at de skal klara av å kommunisere med hverandre. Operativsystemet tar «vekk» informasjonen som finnes i maskinvaren/hardwaren som brukeren ikke trenger eller skjønner noe av, og tilbyr derfor brukeren og applikasjonene en virtuell maskin som er enkel å bruke.

Operativsystemet fungerer også som en ressursadministrator, her er det operativsystemet som gir hvert program tid og plass på ressursene som finnes i maskinen, eks. CPU, minne etc.



b) Hva betyr abstraksjon i OS sammenheng?

Å abstrahere er å ta vekk detaljer som ikke er nødvendige å se for den aktuelle personen, i OS sammenheng så abstraherer operativsystemet detaljer fra maskinvaren for både brukeren og applikasjonene som kjører på maskinen. Det betyr at operativsystemet viser kun den informasjonen eller detaljene som brukeren/applikasjonen trenger for å kunne fungere eller bruke maskinen. For brukeren oppfattes abstraksjoner i form av symboler som er mulig å klikke på, for en applikasjon tilbyr operativsystemet systemkall som gjør det mulig for applikasjonen at kjøre det den er egnet for.

c) Hva er forskjellen på Monolithic kernel og Micro kernel?

Det er forskjellige måter en kjerne, og operativsystemet i seg selv, kan struktureres på. De to arkitekturene man ofte deler operativsystemer inn i er *Monolithic Kernel* og *Micro Kernel.*

En mikrokernel er en kjerne med minimal funksjonalitet. Den har kun det den trenger, som ofte vil det si at den kan administrere mine, prosessorer og interrupts, og evt andre små oppgaver som er nødvendig for systemet. En mikrokernel er ofte veldig lett å bygge på, utvide og endre. Samtidig har den noen faktorer som gjør den mer inefektiv enn andre kjerner.

En monolithic kernel fungerer derimot som en motpart til mikrokernelen. Den er en stor samling av masse funksjoner, der alt er linket sammen til et objekt. Den monolitiske kjernen kan ofte være veldig effektiv og kompleks, da alt er optimalisert på de beste måter, men når det er sagt er de ofte så store og komplekse at det er vanskelig å gjøre endringer. Det finnes mye kode som ingen har oversikt over, og det krasjer veldig lett.

Ofte sier man at de fleste operativsystemer starter som en mikrokernel, men ettersom man legger til mer funksjonalitet, likner den mer og mer på en monolotisk kjerne.

d) Hva er de viktigste funksjonene i operativsystemer?

- Brukergrensesnitt
- Applikasjons-kjøring
- Håndtering av ressurser
- Håndtering av maskinvare
- Håndtering av nettverk
- Sikkerhet

e) Forklar kort hva en ressurs er i sammenheng med et OS.

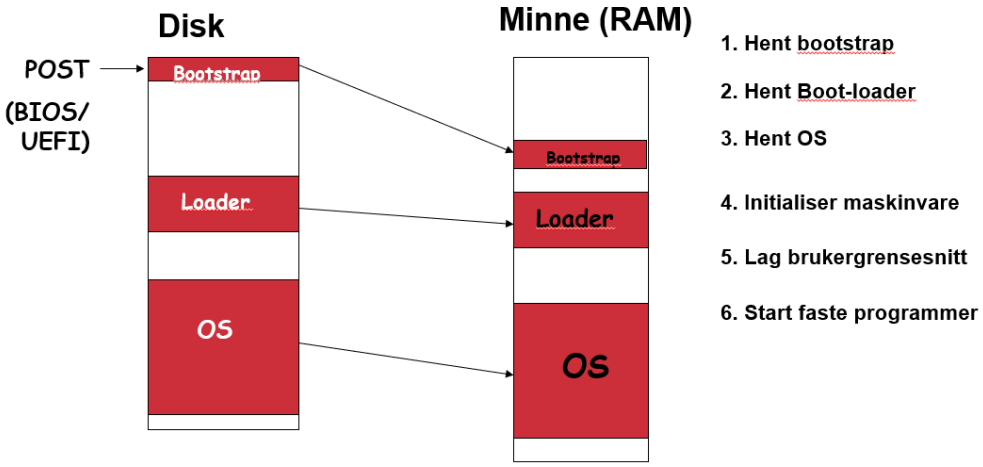
Operativsystemet fungerer som en ressursadministrator, det betyr at det er operativsystemet som sier når et program får tilgang til en ressurs og gir også plass til programmet på ressursen. En ressurs er et element i maskinen som programmet trenger å ta i bruk for å kunne bli kjørt, eks. prosessorer, minne, eksternt lager og I/O enheter.

2. Kjernen

a) Forklar hva som skjer i en oppstart av en maskin med operativsystem.

En såkalt bootstrapping er prosessen for å laste ned de grunnleggende delen i et program inn i minnet under en oppstart eller omstart av maskinen.

Se for deg at du skal starte din pc, når du trykker på knappen så hentes da bootstapen, boot-loadern og operativsystemet in til pc sin RAM. Etter det så sjekkes maskinvaren for å se at det ikke er noe feil med den, når det er gjort så lages brukergrensesnittet som er det vi ser på skjermen og til sist så starter de faste programmene seg (de som starter ved en oppstart uten at du som bruker fysisk setter i gang de).



b) Hva er en kjerne/kernel og hva har den for oppgaver?

Kjernen i en pc er et program, som ligger i sentrum av operativsystemet, og er et av de viktigste elementene for at en datamaskin skal fungere. Den starter opp maskinen, konfigurerer alt av hardware, og administrerer alle prosesser, tråder og ressurser som kjøres i pc-en. Uten kjernen ville ikke pc-en klart å skru seg på ordentlig, og den er kritisk for at du skal kunne bruke maskinen.

c) Hva er risikoene ved å kjøre på kjernemodus?

Når du kjører en maskin i *kjernemodus*, vil det si at du har de samme rettighetene som kjernen din har rettigheter til. Dette inkluderer tilgang til hele minnet, og rettigheter til å kjøre absolutt alle instruksjoner som er på maskinen. Dette vil si at det du velger å kjøre, likesågodt kan krasje hele pc-en din, bruke opp alt av minne, eller ødelegge hardware. Maskinen stopper deg ikke fra noen av disse tingene, siden du har fått rettighetene. Kjernemodus har de høyeste rettighetene man kan få på en pc, og er også kjent som “ring 0”.

d) Når skal man kjøre på bruker- vs kjernemodus i et OS?

De fleste vanlige applikasjoner kjører i brukermodus, og dette fordi det er mye tryggere å begrense rettighetene til en applikasjon fra å kunne tukle med hardware. Eksempler på slike applikasjoner: Word, Calculator, Chrome, World of Warcraft. Noen prosesser må likevel kjøre i kjernemodus, fordi det trenger å aksessere ressurser direkte. Eksempler på disse er: Operativsystem, BIOS, Drivere.

e) Hva er en prosess og en tråd, og hva er forskjellen mellom de?

En prosess er selve programmet som kjører med sine ressurser. En tråd er en del av programmet, og styrer de spesifikke instruksjonene. En prosess kan inneholde alt mellom en til mange tråder som gjør mange små oppgaver samtidig.

f) Hvilket/hvilke program på din datamaskin bruker mest av maskinen sin CPU akkurat nå?

Windows: Bruk taskmanager
OSX: Bruk Activity Monitor
LINUX/OSX: Bruk terminal med kommando fra slideserie

g) Hvorfor bruker operativsystemet Multitasking og hvordan fungerer kommunikasjonen her mellom OS og CPU? Hvordan hadde datamaskinen vært hvis det ikke fantes Multitasking?

Multitasking er når en maskin kan utnytte flere prosesser samtidig. Vi nevnte at en prosess kan ha flere tråder som kjører samtidig, men en datamaskin kan også ha flere prosesser som kjører samtidig. Hvis du vil kjøre flere prosesser samtidig, da trenger du multitasking. Multitasking oppnås ved at Operativsystemet sender en instruksjon til CPU-en. CPU-en kan bare behandle en instruksjon fra en prosess av gangen, per kjerne den har, så d

h) Hvilke tre ulike states finnes i “Context switching”?

- Running: prosessen som kjører akkurat nå
- Ready: Aktivisert prosess og venter i kø
- Blocked: prosess som venter på å bli aktivisert

i) Hva skjer hvis RAM-minnet ikke er stort nok?

I RAM-minnet så ligger alla program som kjøres, hvis det er så at RAM-minnet er for lite kommer noe av innholdet midlertidig legges på harddisken. Dette sier seg selv at nå kommer program kjøres veldig mye langsommere når de ligger data på harddisken også, det tar ekstrem mye lengre tid å hente data fra harddisken en fra RAM-minnet da RAM-en er koblet til North bridge som også er koblet direkte til CPU-en.

j) Hvordan vet et Operativsystem hvilken tråd som skal kjøres når?

Operativsystemet bruker seg av en type av kjøreplan for å vite når og hva som skal kjøres, her er det altså operativsystemet som bestemmer om hvilken tråd som får kjøre på prosessoren til hvilken tid.

k) Hvilke ulike deler har prosessen (tråden) i sitt minne?

- Code segment
- Data segment
- Stack segment
- System data segment (PCB)
- Evt. Flere stacker for trådene

l) Sant/usant: heapen i processen sitt minne er dynamisk.

Sant: at heapen er dynamisk betyr at den kan endre størrelse begge veier.

m) Nevn noen forskjeller mellom HEAP og STACK i en prosess sitt minne.

HEAP er dynamisk og vokser oppover, STACK er statisk og vokser nedover.

n) Hvorfor trenger hardware drivere og hva har det med OS å gjøre? Nevn også noen eksempel på drivere.

Drivere brukes som et grensesnitt mellom hardware og software. Operativsystemet trenger drivere for å kunne «forstå» hvordan det skal bruke hardwaret, hvar slags instrksjuner den kan sende til det osv. Uten drivere ville ikke operativsystemet klart å benytte hardware som blir plugget inn i PC-en (Grafikk-kort, mus, skjerm, vifter, CPU, etc.). Det finnes drivere for alle disse komponentene, men mange er «Plug'n'play» som vil si at du slipper å laste ned noe programvare for å bruke det. Det blir automatsikt lastet ned når du plugger det inn.

3. Skallet

a) «Signaler», «Flag» og «Redirects» bruker man i skjellet, forklar kort hva disse er.

Signaler: Når du kjører i et command line interface så er det mulig å gi signaler til operativsystemet, se dette som typer av kommando med hurtigtaster, for eksempel så kan du trykke `Ctrl-C` for å si «Aborter kjørende prosess».
Flag: Når du skal skrive in en kommando i command line interface har du mulighet å modifisere kommandot med hjelp av ulike flagg. Teste for eksempel `dir /?` (Windows) på din pc og se hva du får opp.
Redirects: Bruker ofte disse for å omdirigere output, for eksempel (finnes flere eksempel enn disse):

```
> skriver output til en fil
>> skriver output in på slutten av en fil
< tar innholdet i en fil
| tar output fra kommando og leverer den som input til
neste kommando
```

b) Ut ifra hvilket OS du har på din maskin, velg fremgangsmåte for WINDOIWS eller OSX/LINUX.

WINDOWS - *Utfør disse oppgavene i CMD (Command promt)*

1. Naviger til Dokumenter-mappen på maskinen din
2. List alle filene som finnes i dokument-mappen
3. Lag et directory(mappe)
4. Gå inn i mappen
5. Skriv in dette for å lage en fil: **notepad** kommandofil.txt
6. Nå skal et nytt notepad-vindu åpnes, skriv valgfri tekst i filen, lagre og lukk notepad-vinduet
7. Skriv ut innholdet i filen
8. Gå tilbake til CMD og kopier .txt-filen, du skal nå ha 2 filer
9. List innholdet i mappen på nytt, nå skal du ha **to filer i mappen**
10. Bytt navn på én av filene
11. Ta nå én av filene og flytt den ut til Dokumenter-mappen
12. Til slutt prøv deg frem med help kommando i CMD

OSX/LINUX - *Utfør disse oppgavene i Terminal.*

1. Naviger til Dokumenter-mappen på maskinen din
2. List alle filene som finnes i dokument-mappen
3. Lag et directory(mappe)
4. Gå inn i mappen
5. Skriv inn dette for å lage en ny fil: echo TK1104 er det beste faget > kommandofil.txt
6. Skriv ut innholdet i filen
7. Kopier kommandofil.txt
8. List innholdet i mappen på nytt, nå er det to filer i mappen
9. Bytt navn på én av filene
10. Flytt en av filene til Dokumenter-mappen
11. Til slutt prøv deg frem med man kommando i terminalen

WINDOWS	OSX/LINUX
dir	ls
type	cat
copy	cp
ren	mv
del	rm
rd	rmdir
find	grep
md	mkdir
cd	cd
help	man

4. Historikk

https://www.youtube.com/watch?v=26QPDBe-NB8
Se YouTube videoen, hent informasjon fra den og stoffet på slideserien, skriv siden 500 ord om OS og historien/utviklingen til OS. Bruk gjerne andre kilder til dette også.