

LUENTO 2

Käyttöjärjestelmän rakenne

Keskeiset käyttöjärjestelmien osa-alueet
Käyttöjärjestelmien kehittyminen
Nykykaikaisen käyttöjärjestelmän piirteitä



Keskeiset KJ:n osa-alueet

- Keskeisimmät osa-alueet (kirjan jaottelu)
 - 1) Prosessin käsite
 - 2) Muistinhallinta
 - 3) Tietoturva ja suojaukset
 - 4) Vuorottaminen ja resurssien hallinta
 - 5) Järjestelmän hierarkkinen rakenne

vrt. edellinen kuva

1) Prosessi

= Suoritettavaksi otettu ohjelma

- koodi muistissa (voi olla yhteiskäytössä)
- oma data-alue ja pino muistissa (muuttujat)
- kaikki ei välttämättä yhtäaikaan muistissa

+ KJ:n ylläpitämät hallinnolliset rakenteet

- prosessin kuvaaja PCB (Process Control Block)
 - CPU:n dataa prosessin suorittamisesta
 - prosessorin rekistereiden arvot
 - KJ:n dataa prosessin hallitsemiseksi
 - tunnistus, omistaja, prioriteetti ..
 - tietoja prosessin varaamasta muistista, tdsista, ...
 - prosessin tila (esim. odottaa siirännän valmistumista)
 - ...

Eräs toteutus

- KJ:llä prosessilista, jossa viitteet prosessin kuvaajiin (context)
- CPU:ssa rekisteri, jossa suoritettavan prosessin numero
- Prosessinvaihto: CPU A:lta B:lle
 - KJ talletti rekistereiden arvot A:n kuvaajaan
 - KJ latsi B:n kuvaajasta arvot rekistereihin

2) Muistinhallinta

- Suoritusaikainen tallennus
 - prosessit (ohjelmat+data) keskusmuistissa (primary memory)
- Pysyvä tallennus
 - tiedostot (ohjelmat+data) 'tukimuistissa' (secondary memory)
- Tilan varaaminen
 - KJ huolehtii automaattisesti
 - kirjanpito vapaista muistialueista / levylohkoista
 - kirjanpito varatuista muistialueista / levylohkoista
 - tarvittaessa KJ käyttää levyä muistin jatkeena (virtuaalimuisti)
- Suojaus ja käyttöoikeudet
 - prosessien eristäminen toisistaan
 - silti tuettava modulaarista ohjelmointia
 - koodin / datan yhteiskäyttö sallittua
 - muistinsuojaus, tdsien käyttöoikeudet

Muistinhallinta

Avainkäsite virtuaalimuisti

- Ei ota kantaa todellisen muistin määrään tai minne ohjelma muistissa sijoitetaan
- Suoritusaikana tarvittava osa koodista/datasta muistissa, loput levyllä
 - MMU huomaa puuttumisen
 - KJ lataa muistiin
- Ei näy sovelluksen ohjelmoijalle

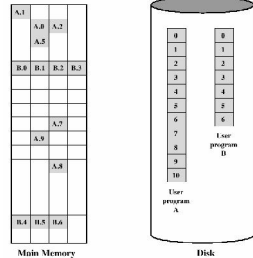
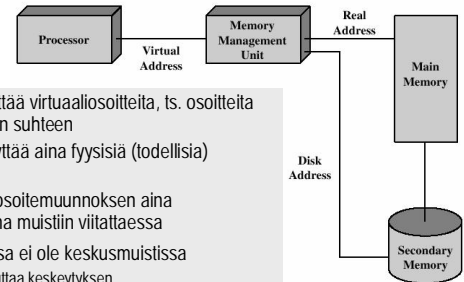


Fig 2.9 [Stal05]

Virtuaalimuisti



- Ohjelma käyttää virtuaaliosoitteita, ts. osoitteita ohjelman alun suhteen
- Laitteisto käyttää aina fyysisiä (todellisia) osoitteita
- MMU tekee osoitemuunnoksen aina suoritusaikana muistiin viitattaessa
 - MMU aiheuttaa keskeytyksen
 - jos muistissa ei ole vapaata tilaa, KJ vapauttaa jonkin alueen
 - KJ hakee levytä puuttuvan osan muistiin

Fig 2.10 [Stal05]

3) Tietoturva ja suojaus

- Käyttäjien tunnistus
 - Vain tunnuksen & salasanan tuntevat käyttäjät saavat ottaa istunnon koneeseen
 - Prosessilla aina omistaja
- Resurssien käyttöoikeus
 - Prosessi käyttää resurssia vain omistajan luvalla
 - Idstoihin liittyy omistaja ja käyttöoikeudet
 - vain omistaja voi muuttaa käyttöoikeuksia
 - Ohjelmat ja data suojattava toisilta ohjelmilta
 - erityisen tärkeää on suojata KJ sovelluksilta
 - MMU ja ajonaikainen osoitemuunnos
 - Resurssien yhteiskäyttö silti sallittava

4) Resurssien hallinta

- Resurssi?
 - CPU, muisti, tdsto, I/O-laite ...
 - CPU:n varaaminen (*allocation*) = vuorottaminen (*scheduling*)
- Milloin?
 - Prosessia käynnistettäessä, suoritettaessa, tapettaessa
 - KJ:n päätöksillä, prosessin (palvelu)pyyntöjen perusteella
- Vastausaika
 - Interaktiivisuus vs. tausta-ajo (eräajo)
 - KJ:n palveluprosessi vs. sovellus
- Tasapuolisuus
 - Samanlaisille prosesseille samantasoinen palvelu
- Tehokkuus
 - Maksimoi läpimenoaste, minimoi vastausajat
 - Palvele mahd. monta (samanaikaista) käyttäjää (sovellusta)

Vuorottamisen perusideoita

- Suoritukseen otetuille prosesseille **READY-jono (short term queue)**
 - vuorottaja* valitsee seuraavaksi suoritettavan prosessin (jonon ensimmäinen)
 - Round-Robin: uusi työ jonon loppuun ja CPU:ta pois tuleva työ jonon loppuun
- Suoritettavaksi ottamista odottaville prosesseille oma jono (**long term queue**)
 - ei liian monta prosessia yhtäaikaan READY-jonoon (moniajoaste)
- Kullakin tapahtumalla omat odotusjononsa
 - I/O-laitteet, semaforit, ajastimet, ...

Milloin?

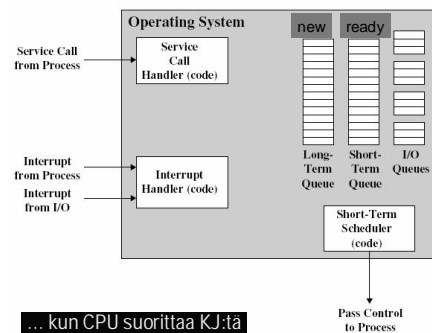


Figure 2.11 Key Elements of an Operating System for Multiprogramming

5) Hierarkkinen rakenne

- KJ muodostuu useista erillisistä tasoista
- Hierarkian ansiosta kokonaisuus jakaantuu helpommin hallittaviin osiin
 - suunnittelu, toteutus, testaus
- Kullakin tasolla oma osajoukko tehtävistä
 - ylemmillä tasoilla kehittyneimmät KJ:n palvelut
 - alemmilla tasoilla laiteläheisimmät toiminnot
- Taso tarjoaa palvelunsa ylemmälle tasolle, käyttää alemman/alempien tasojen palveluja
- Rajapinnat hyvin määriteltyjä
 - tason toteutusta voi muuttaa koskematta muihin tasoihin

KJ:n laiteriippumattomat palvelut

Level	Name	Objects	Example Operations
13	Shell	User programming environment	Statements in shell language
12	User processes	User processes	Quit, kill, suspend, resume
11	Directories	Directories	Create, destroy, attach, detach, search, list
10	Devices	External devices, such as printer, displays and keyboards	Create, destroy, open, close, read, write
9	File system	Files	Create, destroy, open, close read, write
8	Communications	Pipes	Create, destroy, open, close, read, write

KJ:n laiteriippuvat tasot

Level	Name	Objects	Example Operations
7	Virtual Memory	Segments, pages	Read, write, fetch
6	Local secondary store	Blocks of data, device channels	Read, write, allocate, free
5	Primitive processes	Primitive process, semaphores, ready list	Suspend, resume, wait, signal P / V

Muistinhallinta: MMU:n rakenteen huomiointi
 Laittejurit: ohjaimien ja esim. levyn rakenteen huomiointi
 Vuorottaja: rekistereiden talletus/palautus, synkronointiprimitiivit

Laitetasot Brown, Denning 1984

Level	Name	Objects	Example Operations
4	Interrupts	Interrupt-handling programs	Invoke, mask, unmask, retry
3	Procedures	Procedures, call stack	Mark stack, call, return
2	Instruction Set	Evaluation stack, micro-program interpreter, scalar and array data	Load, store, add, subtract, branch
1	Electronic circuits	Registers, gates, buses, etc.	Clear, transfer, activate, complement

Mitä laitteistopiirteitä tarvitaan KJ:n toteutuksessa?

KJ-toteutuksen vaikeat kohdat

- Toimintojen synkronointi / ajoitus
 - joskus odotettava että jotain muuta tapahtuu ennenkuin voi jatkaa
 - keskeytysten oikea priorisointi
 - laitteistosignaaliit tai ohjelmien toisilleen välittämät tiedot eivät saa kadota tai kahdentua
- Poissulkeminen (Mutual exclusion)
 - eräitä resursseja voi käyttää vain yksi kerrallaan
 - esim. kirjoitin tai yhteinen tdsto / data
- Lukkiutumisen / Nälkiintymisen (Deadlock, Starvation)
 - valkeaa havaita
 - odotettava resursseja, jotka ovat toisten hallussa
 - huono prioriteetti, ei saada ensinkään palvelua

Samanaikaisuuden hallinta
 Ch 5, 6.1-6 [Stal05]

kertaa RiO

Käyttöjärjestelmien kehittyminen, kehittäminen ja ylläpito

KJ:n kehittäminen ja ylläpito

- Laitteistot muuttuvat / uusia kehitetään
 - kytkimet, kortit, nauhat, levyt
 - merkkipohjaiset / graafiset päätteet
 - tuki virtuaalimuistille
 - muistin määrä kasvanut, väylät parantuneet, moniprosessorijärjestelmät, jne.
- Tietojenkäsittelytavat muuttuvat
 - interaktiiviset reaaliaikaiset järjestelmät
 - ikkunointiympäristöt
 - paikallisverkot ja Internet
 - kuvankäsittely
 - ...

KJ:n kehittäminen ja ylläpito

- Jatkuvan kehitystarpeen vuoksi
 - modulaarinen rakenne
 - selkeät liittymät eri osien välillä
 - mahd. oliopohjainen toteutus
 - private vs. public data
- Myös KJ:ssä puutteita ja virheitä
 - paikkopaketit (patches, service packages)
 - uudet KJ-versiot
- Milloin aika tehdä KJ uudelleen alusta?

KJ:n historia lyhyesti

Perusmallit:

- Eräajojärjestelmä, yksiajo (Batch System)
- Eräajojärjestelmä, moniajo (Multiprogramming, multitasking)
- Osituskäyttöjärjestelmä (Time-Sharing)

Nykyaikaistetut mallit:

- Moniprosessorijärjestelmä (Multiprocessor)
- Verkkokäyttöjärjestelmä (Networked systems)
- Hajautettu järjestelmä (Distributed system)
- Asiakas-palvelija malli (Client-Server)

KAIKKI TARJOAVAT SAMAT PERUSPALVELUT

Eräajo Yksiajojärjestelmä

Eräajo & yksiajo

- Ensimmäiset KJ:t 50-luvun puolivälissä
- Koneen muistissa yksink. monitoriohjelma
- Käyttäjä määritteli työnsä reikäkorteilla tai nauhalla (ns. kortinkuvat)
 - erätyö = ohjaukortit + ohjelma + data
- Operaattori työnsi kortit lukijaan ja käänsi vipua
- Ohjaukortit kertoivat milloin monitorin piti ladata muita palveluohjelmia (esim. kääntäjä)
- Vain yksi työ kerrallaan suoritettavana, uusi työ ajoin vasta kun edellinen valmis

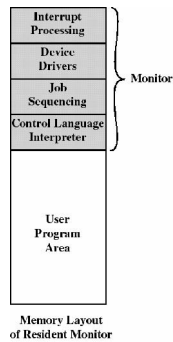
Työnohjauskieli (Job Control Language, JCL)

- Monitorille tarkoitettuja kortinkuvia
 - \$JOB parametrit**
 - \$FTN**
 - Ohjelmakortit**
 - ...
 - \$LOAD**
 - \$RUN**
 - Datakortit**
 - ...
 - \$END**
 - ...
 - Seuraava erätyö**
 - ...
- Esimerkkejä:
 - \$JOB uuden työn alkukortti
 - \$FTN lataa Fortran-kääntäjä ja anna kontrolli sille
 - \$LOAD lataa käännös muistiin
 - \$RUN aja juuri ladattu ohjelma

Monitori

- Jatkuvasti muistissa
- Luki kortinkuvan kerrallaan erätyötä suoritettavakseen
- Kun sovellus ladattu muistiin, suoritus hyppäsi sen alkuun
- Sovellusta suoritettiin kunnes
 - valmistui tai virhe
 - aika loppui
- Kontrolli jälleen monitorille
- Monitori luki seuraavan kortinkuvan

Kuva 2.3



Monitori ja siirräntä

- Monitori huolehti siirräntästä
 - siirräntän yksityiskohdat ei sovelluksen murheena
- I/O-käskey oli itseasiassa aliohjelmakutsu monitorin alueella olevaan koodiin
 - oma käskey, 'palvelupyynnö'
- Monitorin tarjoama palvelu
 - tarkasti, että sovellus ei vahingossa lukenut ohjauskorttia datakseen (-> liian vähän dataa?)
 - ohitti tarvittaessa kortteja, kunnes taas järkeväh ohjauskortti (-> liikaa dataa?)

Monitori ja laitteistopiirteitä (1)

- Muistinsuojaus
 - Monitori suojattava sovellukselta
 - CPU:n tarkistettava muistiosoitteet
 - laitteistossa kantarekisteri BASE
- Keskeytysmekanismi
 - hallittu kontrollin siirto monitorin ja sovelluksen välillä
 - bitti PSW:ssä, keskeytyskäsittelyn alku laitetointo
- Kellokeskeytys
 - ettei yksi sovellus valloittanut koko laitteistoa
 - viimeistään kello aiheutti keskeytyksen
 - kontrolli taas monitorille

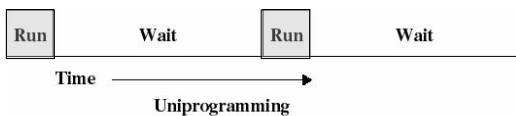
Monitori ja laitteistopiirteitä (2)

- Etuoikeutetut käskeyt (Privileged Instructions)
 - siirräntäkäskeyt
 - muistin rajarekisterin asettaminen
 - keskeytysten esto ja salliminen
 - jos sovellus yrittää käyttää näitä käskeyjä, tuloksena poikkeus 'tuntematon käskeykoodi'
- Etuoikeutettu vs. käyttäjätila (Supervisor/User mode)
 - vain laitteisto ja monitori voi asettaa (bitti PSW:ssä)
 - CPU suorittaa etuoikeutetun käskeyn vain, jos on etuoikeutetussa tilassa

Yksiajojärjestelmän heikkous

- Siirräntä erittäin hidasta verrattuna CPU:n nopeuteen
- CPU odottelee usein siirron valmistumista ennenkuin voi jatkaa sovelluksessa eteenpäin

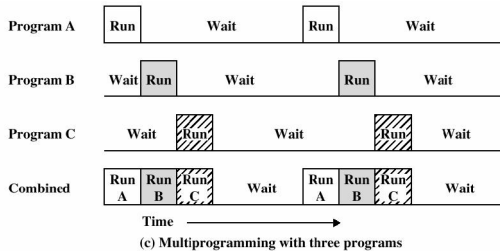
⇒ Huono CPU:n käyttöaste



Moniajojärjestelmä

Moniajojärjestelmä

- Suoritettavaksi useita sovelluksia
 - kun yksi odottaa esim. siirännän valmistumista, CPU suorittaa toista



Lisää laitteistovaatimuksia

- I/O-ohjain keskeyttää, kun siirräntä valmis
 - CPU voi suorittaa muuta siirron aikana
- MMU: suojaus ja ajonaik. osoitemuunnos
 - muistissa yhtäaikaan useita sovelluksia ja sovelluksen sijainti vaihtelee eri suor.kerroilla
 - Jos ei virtuaalimuistia
 - rajarekisteri LIMIT, kantarekisteri BASE
 - Jos virtuaalimuisti
 - sivutaulurekisteri PTR
 - osoitemuunnospuskuri TLB
 - sivunpuutoskeskeytys (page fault)

Lisävaatimuksia KJ:lle

- Prosessien hallinta
 - kirjanpitoa prosesseista = PCB:t
- Vuorottaminen
 - CPU toiselle prosessille, jos yksi jää odottamaan
 - tapahtumaohjattu tai aikaviipaleteknikka
 - prosessin tila: READY vs. BLOCKED
- Muistinhallinta
 - sovelluksille löydettävä tilaa muistista
 - kirjanpito vapaista ja varatuista alueista

Yksiajon ja moniajon vertailu

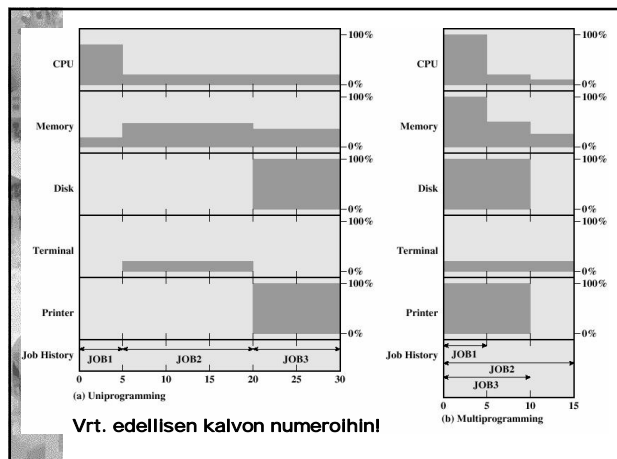
	JOB1	JOB2	JOB3
Type of job	Heavy compute	Heavy I/O	Heavy I/O
Duration	5 min	15 min	10 min
Memory required	50 K	100 K	80 K
Need disk?	No	No	Yes
Need terminal?	No	Yes	No
Need printer?	No	No	Yes

- Muistia 256 K
 - Tässä ei kilpailua oheislaitteistosta
- Taulukko 2.1.

Yksiajon ja moniajon vertailu

	Uniprogramming	Multiprogramming
Processor use	22%	43%
Memory use	30%	67%
Disk use	33%	67%
Printer use	33%	67%
Elapsed time	30 min	15 min
Throughput rate	6 jobs/hr	12 jobs/hr
Mean response time	18 min	10 min

Taulukko 2.2.



Osituskäyttö

- **Osituskäyttö huomioi muuttuneet käyttötavat:** interaktiivinen päätetyöskentely
 - käyttäjä voi käynnistää sovelluksen itse
 - syötteet näppäimistöä
 - tulostus näytölle
- Ihminen koneeseen verrattuna hidas
 - tyypillinen käyttäjä tarvitsee CPU-aikaa vain 2s/min
 - järjestelmässä voi olla esim. 30 yhtäaikaista käyttäjää, eikä yksi edes huomaa muiden läsnäoloa

Osituskäyttö

- Aikaviipalekello
 - vuorottelu ei pelkäästään siirrän odottelun perusteella
 - kullekin vuorotellen aikaviipale (esim. 50-100 ms), jotta voidaan taata kaikille siedettävät vasteajat
- Prioriteetit
 - osituskäytölle suurempi prioriteetti kuin erätöille tai taustalla ajettaviin töille
 - ettei käyttäjä hermostuisi päätteensä ääressä...

NYKYAIKAISEN KJ:N PIIRTEITÄ

Uutuuksia

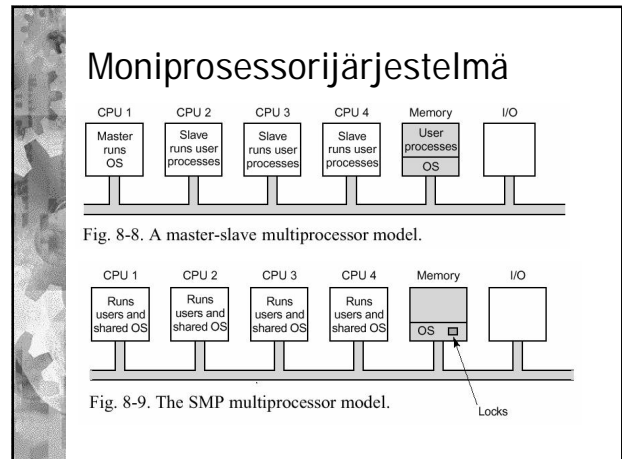
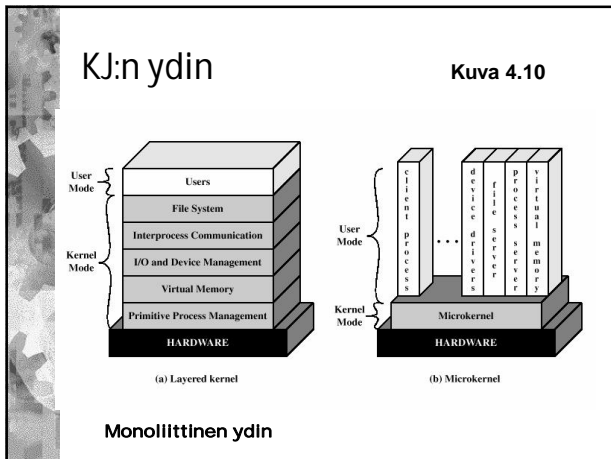
- Laitteistokehitys
 - moniprosessorijärjestelmät
 - nopeat verkot
 - nopeammat prosessorit
 - suurempi muisti, uudet talletusmediat
- Ohjelmistojen / käyttötapojen muutos
 - Asiakas/palvelija -malli
 - Internet ja WWW
 - Multimedia

Mikrokernel

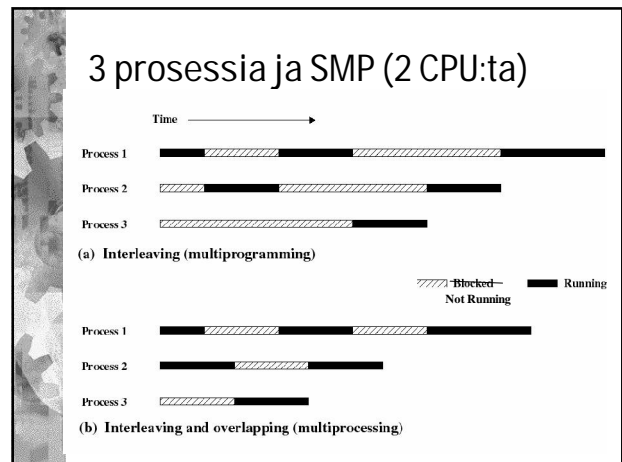
- Vain välttämättömät laiteoiminnot ytimeen, joka suoritetaan etuoikeutetussa tilassa
- keskeytyskäsitteilyn alkutoimet
 - mikä / kuka aiheutti?
- vuorottamisen laiteoiminnot
 - rekistereiden kopiointia
- muistinhallinnan laiteoiminnot
 - MMU:n asetukset, suojaus
- siirrän laiteoiminnot
 - ohjaimien käyttö, suojaus
- prosessien välinen sanomanvälitys
 - pyyntöjen välitys, kopiointia prosessien muistialueille

Mikrokernel

- Muut KJ:n palvelut 'tavallisina' prosesseina, jotka suoritetaan käyttäjätilassa
 - laiteajurit, tiedostojärjestelmä, virtuaalimuisti...
 - odottavat vuorottamista Ready-jonossa
 - eivät pääse suoraan käsiksi laitteistoon
- Toteutus perustuu sanomanvälitykseen
 - IPC, inter process communication
- Joustavuus, laajennettavuus, siirrettävyys ...
- Vrt. Monoliittinen ydin
 - KJ:n keskeiset toiminnot yhdessä ajomoduulissa
 - yleisempää, nopeampaa



- ### SMP – symmetric multiprocessing
- Koneessa useita CPU:ita
 - kaikki rakenteeltaan ja tehtäviltään samanlaisia
 - SMP, Symmetric Multiprocessing
 - aidosti rinnakkainen suoritus
 - kukin voi suorittaa KJ:tä tai sovellusta
 - Muu laitteisto yhteiskäytössä
 - muisti, väylät, I/O-laitteet
 - Useamman CPU:n mukanaolo ei vaikuta normaaliin ohjelmointiin
 - KJ:ssä sensijaan paljonkin uutta mietittävää
 - Tehokkuus, vikasietoisuus, laajennettavuus



- ### Verkkoyrjestelmä
- Useita erillisiä (mahd. erilaisia) solmukoneita
 - Kullakin koneella oma KJ ja omat prosessit
 - Mahd. yhteiskäytössä oleva tdstojärjestelmä
 - Globaali käyttäjien tunnistus
- Verkkoyrjestelmä
- Käyttäjä tuntee ja käyttää koneita nimeltä
 - Toisella koneella olevien tiedostojen käyttö
 - KJ:t voivat olla erilaisia eri koneissa

- ### Hajautettu yrjestelmä
- Useita erillisiä koneita
 - Kullakin koneella oma KJ ja omat prosessit
 - Mahd. yhteiskäytössä oleva tdstojärjestelmä
 - Globaali käyttäjien tunnistus
- Hajautettu yrjestelmä
- Käyttäjän ei tarvitse tuntea koneita nimeltä
 - KJ hoitaa mm. kuormantasauksen
 - Globaali KJ (kaikissa samanlainen)

Asiakas-palvelija malli

- Sovellus jaettu useampaan osaan
 - esim. WWW-palvelija ja selain (käyttöliittymä)
- Asiakas ja palvelija voivat sijaita eri koneissa
 - WWW-palvelija konehuoneen palvelimella, selainohjelma työhuoneen koneella
- tai samassa koneessa
 - ikkunamanageri ja sovellusohjelma
- Palvelija palvelee useita asiakkaita
- Sanomanvälitys
 - TCP/IP-protokolla, etäproseduurikutsu

Reaaliaikajärjestelmä

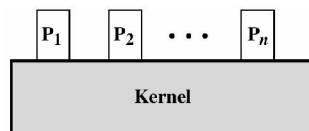
- Tarve reagoida ulkopuolisiin tapahtumiin
 - Ohjausjärjestelmät: laboratoriokeheet, teollisuus, lentoliikenne, teleliikenne, robotiikka
- Tapahtumat tulevat reaaliajassa
 - Ehdittävä käsitellä ennen uutta
- Hard Real-time vs. Soft Real-time
 - Ei saa ylittää aikarajoja (hard deadline) vs. yrittää parhaansa, saa joskus myöhästyäkin (soft deadline)
- Periodinen vs. aperiodinen (jaksollinen, epäsäännöllinen)
 - Ajallinen tai määrällinen säännöllisyys
 - Alku- ja/tai päättymisajalle aikaraja

KJ:N SUORITTAMISESTA

KJ:n suorittamisesta

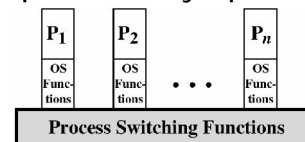
- Myös KJ eräs CPU:n suorittamista käskykokoelmista
- Käyttäjätilassa / etuoikeutetussa tilassa
- KJ:n osat käsittelevät yhteisiä data-alueita
 - melkein kaikki käyttävät PCB:tä (prosessin kuvaaja)
- Onko KJ myös prosessi?

j KJ etuoikeutetussa tilassa



- Prosessi vain käyttäjätilan käsite
 - KJ:n osat eivät jonota
- KJ:llä omat muistialueensa: koodi, data, pino
- KJ:n osat suoritetaan omillaan etuoik. tilassa
 - oikeus tehdä kaikkia KJ:n toimintoja kaikissa osissa
- ~ vanha monoliittinen KJ

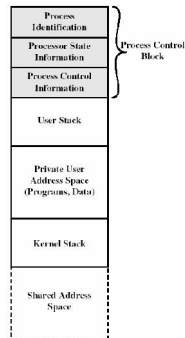
k KJ prosessin ympäristössä



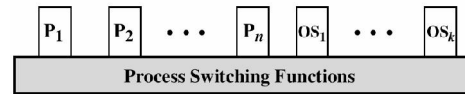
- KJ yhteiskäyttöisellä muistialueella
 - kaikkien prosessien osoitevaruudessa
- Prosessi itse suorittaa KJ:n rutiineja
 - hallittu siirtyminen keskeytyksellä, etuoikeutettu tila
- Kontrolli prosesseilta poissa vain, kun synkronointi tai vuorottaminen vaatii
- ~ uudempi monoliittinen KJ

KJ prosessin ympäristössä

- KJ:n koodi ja data yhteisellä muistialueella
- Prosessi käyttää kernel-pinoa, kun suorittaa KJ:n koodia, muulloin normaalia pinoaan
- Prosessi voi odottaa KJ:n koodissa
- Useita KJ:n osia voi olla yhtäaikaan kesken eri prosessien ympäristöissä
 - suoritukseen vuorottajan kautta



I KJ = joukko palveluprosesseja



- Monet KJ:n palveluista erillisiä prosesseja
 - odottavat Blocked/Ready-jonossa
 - kullakin oma osoiteavaruus
 - tarvittaessa etuoikeutetussa tilassa, erilaisia oikeuksia
- Vuorottaminen prosessien ulkopuolella
- Sanomanvälitys: pyyntö-vastaus mekanismi
 - palvelupyyntö: lähetä / vastaanota sanoma
 - sopii myös moniprosessori / hajautettuihin järjestelmiin
- Jos ytimessä vain laiteriippuvat toiminnot = mikrokernel