Linux sulautetuissa järjestelmissä

Henri Autio

Tampereen yliopisto
Tietojenkäsittelytieteiden laitos
Tietojenkäsittelyoppi
Pro gradu -tutkielma
Huhtikuu 2003

Tampereen yliopisto
Tietojenkäsittelytieteiden laitos
AUTIO, HENRI: Linux sulautetuissa järjestelmissä
Pro gradu -tutkielma, 55 sivua
Huhtikuu 2003

Sulautettuja järjestelmiä löytää nykyään kaiken tyyppisistä laitteista, joihin törmäämme jokapäiväisessä elämässämme. Ne on tarkoitettu kustannustehokkaaksi ratkaisuksi tiettyyn ongelmaan ja niiden laitteisto on rakennettu nimenomaan tiettyä tarkoitusta varten ja vastaavasti ohjelmistopuolella sovellusohjelmat on suunniteltu hyödyntämään mahdollisimman tehokkaasti kyseisen laitteiston tarjoamia ominaisuuksia.

Suurimmaksi ongelmaksi, jota vastaan sulautettujen järjestelmien suunnittelijat nykyään taistelevat, on muodostunut aika; Miten ehditään toteuttaa kaikki uudet ja monimutkaiset ominaisuudet laitteisiin, joita asiakkaat haluavat? Lyhyen kehityskaaren aikana on kyettävä reagoimaan nopeasti markkinoiden muutoksiin ja tämä asettaa omat vaatimuksensa sulautetun järjestelmän käyttöjärjestelmälle; sen on oltava joustava.

Tutkimuksessani paneudun sulautettuihin järjestelmiin Linux-käyttöjärjestelmän näkökulmasta ja tarkoitukseni on selvittää niitä asioita, jotka tekevät Linuxista kilpailukykyisen ja varteenotettavan vaihtoehdon sulautetun järjestelmän sieluksi, siis käyttöjärjestelmäksi. Lisäksi haluan valottaa niitä asioita, miksi on perusteltua valita Linux sulautettuun järjestelmään ja mitä mahdollisia rajoituksia sillä on sulautetun järjestelmän kannalta.

Avainsanat ja –sanonnat: Linux, sulautettu järjestelmä.

Alkusanat

Haluan kiittää Pro Gradu –tutkielmani ohjaajaa Jyrki Nummenmaata Tampereen Yliopiston tietojenkäsittelytieteiden laitokselta. Lisäksi haluan kiittää vaimoani Sandraa, vanhempiani sekä siskoani tuesta ja kannustuksesta, jota olen saanut opiskeluni aikana.

Tampereella 12. toukokuuta 2003

Henri

Sisältö

1 Johdanto

*Sulautetuista järjestelmistä (*embedded systems*)* on tullut avaintekijä modernissa yhteiskunnassa. Sulautettuja järjestelmiä löytää nykyään kaiken tyyppisistä laitteista [Ortiz, 2001]. Niitä on käytössä esimerkiksi reitittimissä ja kytkimissä, jotka mahdollistavat verkkojen, kuten Internetin, toiminnan. Niitä on lääketieteellisissä laitteissa, jotka pitävät potilaat hengissä, kopiokoneissa, TV:n kaukosäätimissä, pankkiautomaateissa ja monissa muissa laitteissa, joihin törmäämme jokapäiväisessä elämässämme. Käytännössä sulautetut järjestelmät ovat laitteita, jotka sisältävät tietokoneen sekä tarkoitukseen soveltuvan ohjelmiston. Sulautetun järjestelmän idea on siinä, että käyttäjän ei tarvitse tietää olevansa tekemisissä tietokoneen kanssa, eikä käyttäjä yleensä edes tiedosta sulautetun järjestelmän olemassa oloa laitteessa.

Sulautettu järjestelmä on tarkoitettu kustannustehokkaaksi ratkaisuksi tiettyyn ongelmaan. Sulautetussa järjestelmässä laitteisto on yleensä rakennettu nimenomaan tiettyä tarkoitusta varten, eikä siinä siksi ole ylimääräisiä laitteistokomponentteja. Vastaavasti ohjelmistopuolella sovellusohjelmat on suunniteltu hyödyntämään mahdollisimman tehokkaasti kyseisen laitteiston tarjoamia ominaisuuksia [Lombardo, 2001]. Sulautettujen järjestelmien laitteistokomponenttien, kuten suorittimien, keskusmuistin ja flash-muistin, hinnat ovat pudonneet huomattavasti viimevuosien aikana. Samaan aikaan sulautetuissa järjestelmissä ajettavien ohjelmistojen monimutkaisuus on kasvanut merkittävästi. Nämä kaksi asiaa yhdessä ovat aiheuttaneet sen, että suurin ongelma sulautettua järjestelmää rakennettaessa ei enää ole viimeisen vapaan tavun löytäminen, johon ohjelmisto saadaan mahtumaan. Suuremmaksi ongelmaksi, jota vastaan sulautettujen järjestelmien suunnittelijat nykyään taistelevat, on muodostunut aika. Miten ehditään toteuttaa kaikki uudet ja monimutkaiset ominaisuudet laitteisiin, joita asiakkaat jatkuvasti vaativat? Tämän päivän laitteisiin halutaan enemmän älyä ja toiminnallisuutta kuin aikaisemmin. Laitteisiin kohdistuvien vaatimusten kasvaminen on johtanut siihen, että sulautetun laitteen toteuttamiseen ei enää välttämättä riitä pelkkä yksinkertainen symbolisella konekielellä ohjelmoitava mikrokontrolleri, vaan avuksi tarvitaan laajempi kokonaisuus, jossa jokaista asiaa ei tarvitse tehdä alusta lähtien itse. On tärkeätä, että kehitystyössä voidaan hyödyntää mahdollisimman paljon jo olemassa olevia ohjelmisto-komponentteja.

Tutkimuksessani tulen käsittelemään asioita, jotka tulee ottaa huomioon suunniteltaessa sulautettua järjestelmää. Tutkimukseni painottuu sulautettuihin järjestelmiin Linux-käyttöjärjestelmän näkökulmasta ja tarkoitukseni on

selvittää niitä asioita, jotka tekevät Linuxista kilpailukykyisen ja varteenotettavan vaihtoehdon sulautetun järjestelmän sieluksi, siis käyttöjärjestelmäksi. Tutkimukseni kautta haluan valottaa sitä, miksi on perusteltua valita Linux sulautettuun järjestelmään ja mitä mahdollisia rajoituksia sillä on sulautetun järjestelmän kannalta. Lisäksi käsittelen reaaliaikaisuutta ja sitä miten Linux kykenee tarjoamaan reaaliaikaisuutta sitä tarvitseville sovellusohjelmille. Esittelen samalla kolme tärkeintä menetelmää, joilla Linuxin reaaliaika ominaisuuksia voidaan parantaa. Reaaliaikaisuus on erityisen tärkeässä asemassa sulautettujen järjestelmien kannalta siksi, että niissä on lähes poikkeuksetta tarvetta viiveettömään toimintaan. Normaalissa työpöytä- ja palvelinkäytössä tällaista tarvetta ei yleensä esiinny.

Tutkimusaiheestani ei ole tehty varsinaisia tutkimuksia, koska Linux sulautetuissa järjestelmissä on vielä uusi tulokas. Siksi lähteet, joihin tämä tutkimus viittaa, ovat pääsääntöisesti käytännönläheisiä raportteja Linuxin ominaisuuksista ja siitä miten Linuxia on tähän mennessä hyödynnetty sulautetuissa järjestelmissä. Lähdekirjallisuuden avulla olen pyrkinyt löytämään tutkimukseni kannalta oleellisia tarkastelukohtia aiheeseen. Olen pohtinut Linuxia ja sulautettuja järjestelmiä lähdemateriaaliin tuettuna sekä sitä, miten Linux-käyttöjärjestelmä soveltuu sulautettuihin järjestelmiin. Lisäksi olen pyrkinyt tuomaan esille omakohtaisia kokemuksiani vuodelta 2001, jolloin työskentelin Flander Oy:ssä Irma nimisessä hankkeessa, jossa suunnittelin sekä toteutin sulautettua Linuxia hyödyntävän laitteen ohjelmiston ja räätälöin sen käyttöjärjestelmän käsin. [Flander]

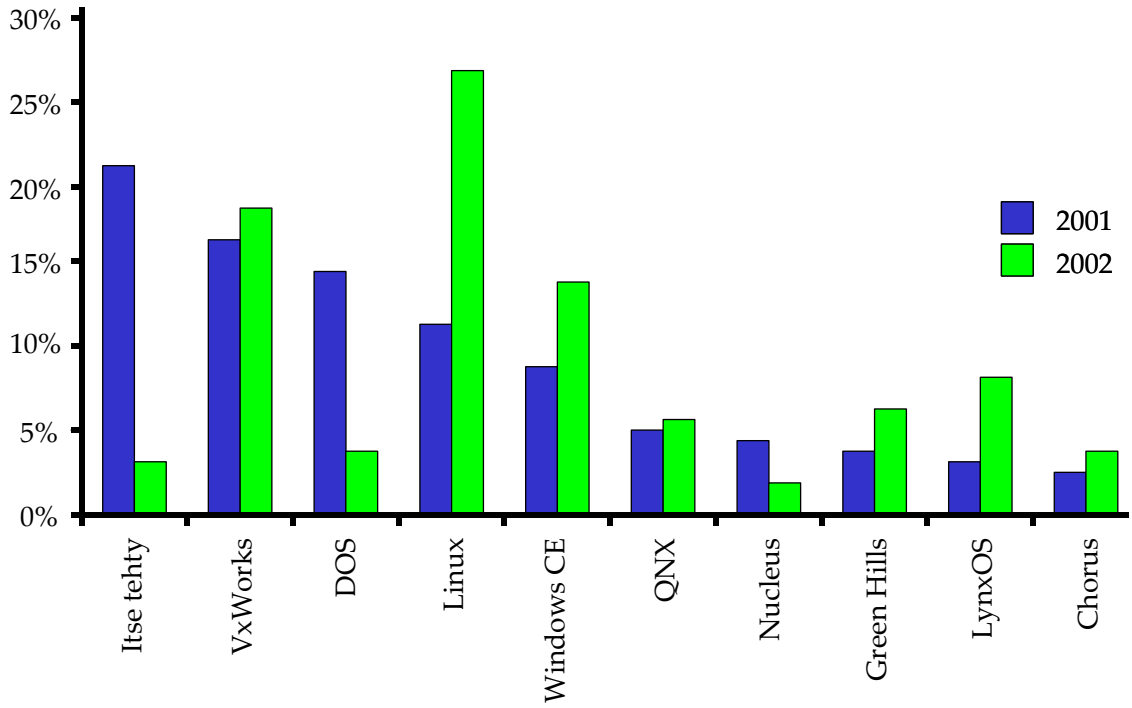
Tutkimus sisältää johdannon lisäksi viisi muuta lukua. Toisessa luvussa käsittelen lyhyesti tutkimuksen lähtökohtia. Kolmannessa luvussa esittelen Linuxin taustoja, miten tähän ollaan tultu. Neljännessä luvussa esittelen keskeisiä käsitteitä, jotka liittyvät erittäin läheisesti tutkimusaiheeseen ja joiden ymmärtäminen on tärkeää koko tutkimuksen ymmärtämisen kannalta. Viidennessä luvussa tarkastelen Linuxin ominaisuuksia sulautetun järjestelmän kannalta. Kuudennessa luvussa käyn läpi muutamia ratkaisuja, joilla Linuxin reaaliaikaominaisuuksia voidaan parantaa ja samalla tarkastelen niiden soveltuvuutta. Seitsemännen luvun aiheena on sulautetun Linuxin sovellukset, jossa esittelen Linuxin nykyisiä käyttökohteita sulautetuissa järjestelmissä. Tutkimuksen viimeinen kappale on varattu keskustelulle ja aiheen syvemmälle pohdinnalle sekä aiheen yhteenvedolle. Tutkimuksen varsinainen sisältö on neljässä viimeisessä luvussa.

2 Tutkimuksen lähtökohdat

Markkinoilla on jo tarjolla kymmeniä erilaisia kaupallisia sulautettuja käyttöjärjestelmiä. Kilpailu niiden välillä on erittäin kovaa, mutta selvää markkinajohtajaa ei vielä ole tunnistettavissa, toisin kuin työpöytä- sekä palvelinkäyttöjärjestelmien puolella.

Venture Development Corporationin tutkimuksen mukaan sulautettujen käyttöjärjestelmien myynti tulee kasvamaan kaksinkertaiseksi, 1.6 miljardiin euroon nykyisestä 752 miljoonasta eurosta, vuoteen 2005 mennessä. Markkinoiden voimakkaasti kasvava kysyntä on avannut pelikentät uusille yrittäjille, joihin myös Linux lukeutuu [Ortiz, 2001]. Erityisen mielenkiintoiseksi Linuxin tulemisen sulautettujen käyttöjärjestelmien markkinoille tekee se, että Linux on *vapaa niin sanotuista royaltymaksuista* (Royalty Free) eli sen käytöstä ei tarvitse maksaa tekijänkorvauksia.

Ohjelmistomarkkinoita analysoivan Evans Data Corporationin [Aug 17] syksyllä 2001 tekemän kyselyn mukaan Linux on noussut vahvaksi sulautettujen järjestelmien käyttöjärjestelmäksi. Kyselyyn osallistui yli 500 sulautettujen järjestelmien kehittäjää.



Kuva 1: Evans Data Corporation: Sulautettujen käyttöjärjestelmien käyttö.

Kuvasta 1 voimme todeta, että kyselyn tekohetkellä, vuonna 2001, suosituimmat käyttöjärjestelmät sulautetuissa järjestelmissä olivat:

1. Itse tehty
2. Wind River VxWorks ja VxWorks AE
3. DOS
4. Sulautettu Linux
5. Microsoft Windows CE
6. QNX
7. ATI Nucleus
8. Green Hills ThreadX ja Integrity
9. LynuxWorks LynxOS
10. Sun Microsystems Chorus

Lisäksi kuvasta 1 voimme nähdä, että Evans Data Corporationin kyselyn mukaan kehittäjät arvioivat, että vuonna 2002 he tulevat käyttämään sulautetuissa järjestelmissään seuraavia käyttöjärjestelmiä:

1. Sulautettu Linux
2. Wind River
3. Windows CE
4. LynxOS
5. BSD
6. Green Hills
7. QNX
8. DOS
9. Chorus
10. Itse tehty

Kyselystä saatujen tulosten mukaan Linux-käyttöjärjestelmän näkymät sulautettujen järjestelmien markkinoilla ovat erittäin lupaavat. Linuxin käytön on odotettavissa kasvavan tulevaisuudessa vieläkin suuremmaksi suhteessa muihin sulautettuihin käyttöjärjestelmiin.

3 Taustaa

3.1 UNIX™

UNIX on laitteistoriippumaton käyttöjärjestelmä, jonka kehitystyö alkoi vuonna 1969, jolloin Ken Thompson Bell's Laboratoryn tutkimusryhmästä aloitti moniajavan, monenkäyttäjän käyttöjärjestelmän kehittämisen. Myöhemmin tutkimusryhmään, Thompsonin seuraksi, liittyi Dennis Richie. Yhdessä he muiden tutkimusryhmän jäsenten avustuksella kehittivät esiversion UNIX-käyttöjärjestelmästä. Kun UNIX-käyttöjärjestelmä oli saatu julkaisukuntoon, kaikki suurimmat tietokonevalmistajat alkoivat toimittaa sitä laitteidensa mukana [Rusling, 1996 - 1999]. Loppu onkin tietokonealan historiaa. UNIX on tähän päivään mennessä saavuttanut huomattavan paljon jalansijaa ympäri maailmaa, niin työasemissa, kuin palvelin puolellakin.

Tällä hetkellä UNIXista on saatavilla lukuisia eri versioita. Kaupallisesti tunnetuimpia ja käytetyimpiä ovat muun muassa: Sun Microsystems Solaris, International Business Machines eli lyhyemmin IBM AIX, Digital Unix sekä Silicon Graphics eli SGI ja Irix. [Peltomäki & Linjama 1999]

3.2 Minix

Minix oli mikrotietokoneille tarkoitettu UNIX-yhteensopiva käyttöjärjestelmä. Minixin kehitti saksalainen professori Andrew S. Tanenbaum Intel 8086 mikroprosessorille. Sen ydin perustui niin sanottuun mikrokernel arkkitehtuuriin, joten sen ydin oli hyvin pieni kooltaan. Minix oli tarkoitettu ja kehitettiin alunperin opetuskäyttöä varten. Tanenbaum halusi näyttää oppilailleen miten oikea käyttöjärjestelmä toimii sisäisesti. Minix oli hyvin yksinkertainen käyttöjärjestelmä ja siksi myös hyvin rajoittunut, ja siinä oli paljon puutteita. Sen ainoa käytännön etu muihin sen ajan UNIX-käyttöjärjestelmiin oli se, että kuka tahansa sai Minix kaikki lähdekoodit itselleen ostamalla Tanenbaumin kirjan Operating System. Lisäksi Minix oli selvästi edullisempi kuin muut kaupalliset UNIX-käyttöjärjestelmät. Näiden kahden asian seurauksena Minixistä tuli lopulta hyvin suosittu tietokoneharrastelijoiden sekä opiskelijoiden keskuudessa. [Rusling 1996 - 1999; Peltomäki & Linjama 1999; Hasan]

3.3 Linux

Linux tai tarkalleen sanottuna *GNU*/Linux (GNU it's Not UNIX, GNU) on UNIX-yhteensopiva *avoimen lähdekoodin* (Open Source) ydin ja sen kehitystyön aloitti 1990-luvun alussa suomalainen Linus Torvalds. Linuxin juuret vievät aina UNIXin syntymävuoteen 1969 saakka. Linuxia voi kuka tahansa muokata ja levittää vapaasti GPL-lisenssin alaisena.

Yleisesti kun puhutaan Linuxista, tarkoitetaan koko järjestelmää, joka normaalisti sisältää nykypäivänä myös X-ikkunoinnin, työpöydän hallintaohjelmiston, kuten esimerkiksi KDE:n ja paljon muita erilaisia sovellusohjelmia. Linuxia on saatavilla valmiissa paketeissa usealta eri valmistajalta. Tällaista valmista kokonaisuutta kutsutaan nimellä *jakelupaketti* (distribution). Tunnetuimpia Linux jakelupaketteja ovat Red Hat, Mandrake ja SuSe, mutta saatavilla on lukuisia muitakin. Ero eri jakelupakettien välillä syntyy muun muassa asennuksen helppoudessa, tukipalveluissa sekä mukana toimitettavien sovellusohjelmien määrässä. Linux-*käyttöjärjestelmäksi* (Operating System, OS) tulee mielestäni kutsua vain Linuxin *ydintä* (Kernel) yhdessä muiden perustoiminnallisuuden mahdollistavien kirjastojen ja sovellusohjelmien kanssa, joihin voidaan laskea esimerkiksi lähes poikkeuksetta jokaisen ohjelman käyttämä C–kirjasto, sillä muiden ohjelmistokomponenttien olemassaolo ei ole edellytys Linuxin toiminnalle.

3.3.1 Arkkitehtuuri

Perusarkkitehtuuriltaan Linux noudattaa BSD UNIXin ratkaisuja, mutta myös joitain System V:n ominaisuuksia on otettu mukaan. Yksi tällaisista ominaisuuksia on muun muassa *prosessienvälinen kommunikaatio* (Inter-Process Communication, IPC) [Koski & Tervo, 1997] Linux on *monenkäyttäjän* (multiuser), *moniajava* (multitasking) sekä *moniprosessorointia* (multiprocesssing) tukeva käyttöjärjestelmä [Lennon, May 2001] eli se mahdollistaa usean samanaikaisen prosessin suorituksen ja käyttäjän työskentelyn samanaikaisesti. Luonnollisesti jokaisella käyttäjällä tulee olla oma näyttö sekä näppäimistö, mutta kaikki muut resurssit, kuten kovalevytila sekä prosessoriaika ovat jaettuja.

3.3.1.1 Modulaarisuus

Linux on *modulaarinen* käyttöjärjestelmä. Tämä tarkoittaa käytännössä sitä, että ydin ei tarjoa muuta, kuin rajapinnan sovellusohjelmien ja laitteiston

välille. Esimerkiksi komentotulkki ja graafinen käyttöliittymä on Linuxissa täysin erillinen sovellusohjelma eli se on kaikkien muiden ohjelmistokomponenttien lisäksi valittavissa täysin vapaasti.

3.3.1.2 Monoliittinen ydin

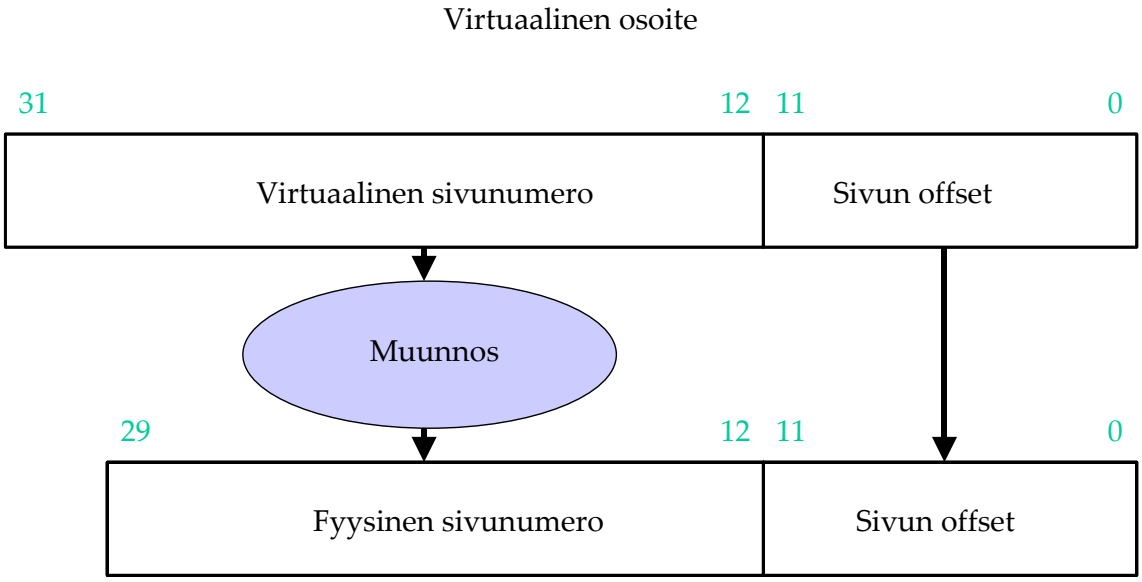
Linuxin ydin on *monoliittinen* (monolithic) eli rakenteeltaan yhtenäinen. Tämä tarkoittaa sitä, että kaikki käyttöjärjestelmän osat, joihin kuuluu *prosessien hallinta* (process management), *muistin hallinta* (memory management), *tiedostojärjestelmä* (file system) ja *laiteajurit* (driver) ovat osa yhtenäistä binääri–tiedostoa, joka suoritetaan kokonaisuudessaan yhdessä prosessissa. Linuxin rakenne on siis päinvastainen Minix mikro-ydin arkkitehtuuriin verrattuna, jossa suurin osa käyttöjärjestelmän eri osista suoritetaan erillisissä prosesseissa, ytimen ulkopuolella.

3.3.1.3 Moduulit

Ytimen *arkkitehtuurin* (architecture) yksi hienoimmista ominaisuuksia on ladattavien moduulien tuki. Vaikka Linuxin ydin on rakenteeltaan monoliittinen, se mahdollistaa *laajennusosien* (extensions) yhdistämisen *ajon aikana* (runtime) ytimeen, linkittämällä siihen lisäkoodia [Lennon, May 2001]. Tämä mahdollistaa ytimen toiminnallisuuden muuntamisen dynaamisesti ilman, että järjestelmää tarvitsee käynnistää uudelleen. Tällaisia ajon aikana ladattavia moduuleita ovat esimerkiksi laiteajurit.

3.3.1.4 Muistinsuojaus

Muistinsuojauksen (memory protection) etuna on, että huonosti käyttäytyvällä ohjelmalla on häviävän pieni todennäköisyys aiheuttaa muiden ohjelmien tai jopa koko järjestelmän kaatuminen. Muistinsuojaus perustuu siihen, että jokaiselle prosessille annetaan käyttöön oma virtuaalinen osoiteavaruus, jonka ansiosta mikään prosessi ei pysty osoittamaan toisen prosessin muistiosoitteisiin aiheuttaen ongelmia. Prosessin osoiteavaruudella tarkoitetaan muistialuetta, joka on varattu ainoastaan tälle prosessille. Kuva 2 havainnollistaa miten virtuaalinen muistiosoite on erotettu fyysisestä muistiosoitteesta. Muistiosoitteen muunnos tapahtuu käyttöjärjestelmän toimesta käyttäen apuna suorittimen muistinhallinta-yksikköä. [Rusling, 1996 - 1999]



Kuva 2: Virtuaalisen muistiosoitteen muunnos fyysiseksi muistiosoitteeksi. [George Mason University]

3.3.2 Kehitys

Torvalds oli yksi niistä tietokoneharrastajista, jotka joutuivat aikoinaan käyttämään Minixiä paremman käyttöjärjestelmän puutteessa. Hän, eikä moni muukaan, ollut Minixiin eikä sen tarjoamiin ominaisuuksiin tyytyväinen. Muut tarjolla olleet UNIXit olivat esimerkiksi Sun Microsystemsin kehittämä Solaris. Solariksen ja muiden kaupallisten UNIXien suurin ongelma oli ja on edelleenkin se, että ne toimivat pääsääntöisesti ainoastaan valmistajien omilla *RISC* (Reduced Instruction Set Computer) eli rajoitetun käskykannan laitteistokokoonpanoilla, jotka olivat silloin ja ovat nykyäänkin kalliimpia, kuin PC-laitteet. Ei pidä myöskään unohtaa sitä, että myös UNIX-käyttöjärjestelmästä joutui maksamaan kovan hinnan. Torvalds halusi kehittää paremman käyttöjärjestelmän, joka toimisi Intelin halvoilla x86-arkkitehtuurin suorittimilla ja joihin tavallisella ihmisillä olisi varaa. [Torvalds 2001]

"Linux is not portable. It uses 386 task switching etc. and it probably never will support any thing other than AT-hard disk, as that's all I have" [Torvalds, 1991]. Vapaasti suomennettuna yllä oleva tarkoittaa, että "Linux ei ole siirrettävissä muille alustoille. Se käyttää 386 tehtävänvaihtoa ja niin edelleen. Se ei tule koskaan tukemaan muita, kuin AT-kovalevyjä, koska minulle ei ole muita käytössä". Lausahdus on näin jälkeenpäin ajatellen todella huvittava, ottaen huomioon miten valtavasti Linux on kehittynyt tuosta ajasta tähän päivään

mennessä ja miten monella eri laitealustalla ja oheislaitekokoonpanolla se nykyään toimii.

Ensimmäinen virallisesti julkaistu Linux versio oli numeroltaan 0.02. Tämä Linux versio saatettiin kaikkien alan harrastajien saataville vuoden 1991 lokakuussa. Tämä versio oli vielä varsin vaatimaton sekä riittämätön ominaisuuksiltaan muuhun, kuin testikäyttöön. Linux osoitti kuitenkin jo ensi metreillä sellaisia vahvuuksia ja piirteitä muihin käyttöjärjestelmiin verrattuna, että sitä kokeilleet ihmiset alkoivat kiinnostua kasvavassa määrin asiasta ja alkoivat näkemään sen vaihtoehtoisena käyttöjärjestelmänä. Tuhannet harrastelijat ympäri maailmaa kasvavassa määrin kiinnostuivat Linuxista ja alkoivat käyttää sekä kehittämään sitä. Kiinnostus ja harrastamisen määrä kasvoi nopeasti, josta seurasi se, että Linuxin julkistamisen jälkeen kehitys on ollut valtavan nopeaa ja jo kaksi ja puolivuotta ensimmäisen version jälkeen oli valmiina versio 1.0. Tämä versio oli historian ensimmäinen Linux, joka oli riittävän kehittynyt vakavasti otettavaksi vaihtoehdoksi tuotantokäyttöön.. [Rousku 1997]

Suurimpana lisäyksenä, vuonna 1996 julkaistu, Linuxin versio 2.0 toi mukanaan tuen *rinnakkaisprosessoinnille* (Symmetric MultiProcessing, SMP) eli tuen monelle suorittimelle, joka on käytössä erityisesti palvelimissa. Tänään Linux on saavuttanut versionumeron 2.4 ja nyt mukana on tuki muun muassa *USB* (Universal Serial Bus), *BT* (Bluetooth), IrDA, IPv6, *WLAN* (Wireless Local Area Network), sekä lukuisille muille tämän päivän uusille tekniikoille ja standardeille.

Koska kukaan ei omista Linuxia, ei sen kehitystä valvo mikään yritys tai organisaatio. Jotta kehitystyö pysyisi kuitenkin jossain määrin hallinnassa, keräävät Linus Torvalds ja Alan Cox, muiden Linuxin pääkehittäjien kanssa, ytimeen tehdyt päivitykset sekä muutokset keskitetysti yhteen ja julkaisevat Linuxin viralliset versiot vapaaehtoisvoimin. Lisäksi Linus Torvalds omistaa itseoikeutetusti Linux –tuotemerkin oikeudet.

4 Keskeiset termit ja käsitteet

Erityisen tärkeässä asemassa Linuxin saavuttaman huomion, levinneisyyden ja kehittymisen kannalta ovat olleet avoin lähdekoodi ja GPL–lisenssi, jonka alaisuudessa Linuxia levitetään. Niiden tuominen esiin tässä on ensiarvoisen tärkeää, jotta ymmärtää mihin Linuxin filosofia ja kantava voima perustuu. Lisäksi haluan tuoda esille muutamia, käyttöjärjestelmän kannalta keskeisiä termejä ja käsitteiden määrityksiä, jotta lukijalle on selvää myöhemmissä luvuissa, asioita tarkemmin käsiteltäessä, mistä on kysymys.

4.1 Avoin lähdekoodi

Avoimen lähdekoodin määrittävän Open Source Definitionin mukaan avoimella lähdekoodilla ei tarkoiteta pelkästään lähdekoodin avointa saatavuutta, vaan ohjelmien on täytettävä seuraavat yhdeksän erityisehtoa:

1. *Vapaa levitysoikeus* (Free Distribution)
2. *Lähdekoodi* (Source Code)
3. *Johdannaiset teokset* (Derived Works)
4. *Lähdekoodin yhteenkuuluvuus* (Integrity of the Author's Source Code)
5. *Henkilöiden ja ryhmien syrjinnän kielto* (No Discrimination Against Fields of Endeavor)
6. *Toimialojen syrjinnän kielto* (No Discrimination Against Fields of Endeavor)
7. *Lisenssin levittäminen* (Distribution of License)
8. *Lisenssi ei saa olla tuotekohtainen* (License Must Not Be Specific to a Product)
9. *Lisenssi ei saa rajoittaa muita ohjelmia* (The License Must Not Restrict Other Software)

Avoimen lähdekoodin määrityksen koko virallinen suomennos on luettavissa kokonaisuudessaan Internetissä. [OpenSource.org]

4.2 GNU General Public License

Linux lähdekoodi on suojattu *GNU copyleftillä* (GNU General Public License, GNU GPL). Tiivistettynä lisenssiehdot sanovat, että ohjelmaa suojaa tekijän omistama tekijänoikeus, mutta tekijänoikeutta käytetään vain suojaamaan ohjelman vapaata levitystä. Kaikki GNU copyleftin suojaamien ohjelmien lähdekoodit on oltava vapaasti kaikkien saatavilla. Ohjelmien lähdekoodeihin

saa tehdä muutoksia, mutta muutettua versiota ei voi kaupallisesti hyödyntää jos tarkoituksena on puhtaasti myydä GPL-ohjelmistoa. Muutettujen ohjelmaversioiden lähdekoodien pitää olla vapaasti saatavilla. Ohjelmien levityskustannuksista voi periä maksun, mutta sisältö on edelleen ilmainen ja vapaasti levitettävissä sekä kopioitavissa. [Free Software Foundation, 1991]

GPL-lisenssin suomennettu, mutta toistaiseksi kuitenkin epävirallinen, versio on luettavissa kokonaisuudessaan Internetissä. [Free Software Foundation, 1991]

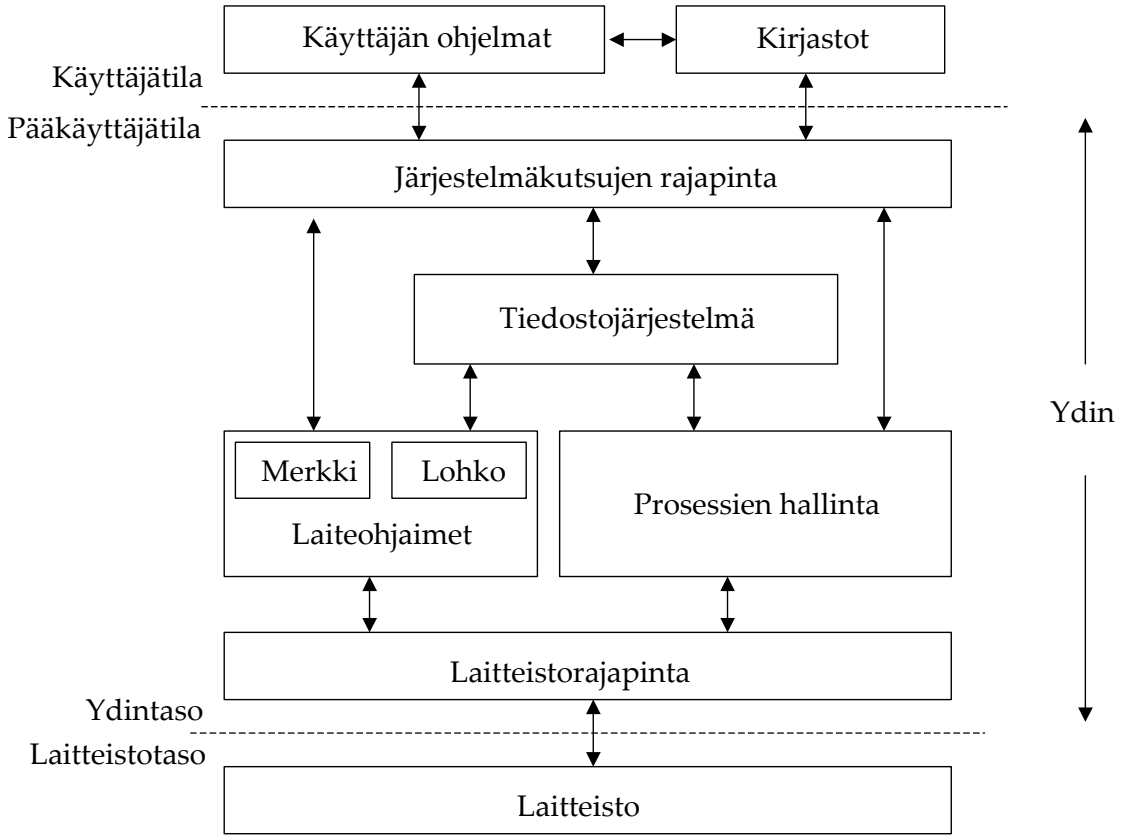
4.3 POSIX

Portable Operating System Interface X eli lyhemmin POSIX on *IEEE*:n (Institute of Electrical and Electronics Engineers) [IEEE] määrittelemä käyttöjärjestelmäriippumaton ohjelmointirajapinta [Koski & Tervo, 1997]. Sen tavoitteena on ohjelmien lähdekoodien siirrettävyys eli yhteensopivuus eri käyttöjärjestelmien ja laitealustojen välillä. Se määrittää miten käyttöjärjestelmän tulee toimia perusoperaatioiden lisäksi muun muassa signaloinnin ja *säikeiden* (thread) osalta [FSMLab RTLinux]. POSIXia ei ole kuitenkaan tarkoitettu pelkästään UNIXien väliseksi standardiksi, vaikka siitä yleensä puhutaan pelkästään niiden yhteydessä. Myös esimerkiksi Microsoft Windows NT tukee POSIX määrityksen mukaisia ohjelmointirajapintoja. [IEEE]

4.4 Käyttöjärjestelmä

Käyttöjärjestelmä on kokoelma järjestelmätason ohjelmia, jotka mahdollistavat *käyttäjätilan* (user mode) sovellusohjelmien suorittamisen. Käyttöjärjestelmän tärkein ja lähimpänä laitteistoa oleva osa on *ydin* (kernel). Käyttöjärjestelmän päätehtävä on kontrolloida laitteiston sekä sovellusohjelmien yleistä toimintaa ja tarjota sovellusohjelmille yhtenäinen *ohjelmointirajanpinta* (Application Programming Interface, API), jonka tarkoituksena on yksinkertaistaa ja yhtenäistää laitteiston osien käyttöä. Ohjelmointirajapintojen ansiosta jokaisen sovellusohjelmoijan ei tarvitse huolehtia jokaisesta perusasiasta itse. Käyttöjärjestelmän tehtäviin kuuluu myös jakaa laitteistoresurssit, kuten suoritin ja keskusmuisti, samanaikaisesti suoritettaville prosesseille. Lisäksi käyttöjärjestelmä tarjoaa sovellusohjelmille yleispalveluita, joita ovat esimerkiksi tiedostojenkäsittely sekä erilaiset *syöttö- ja tulostus* (Input/Output, I/O) operaatiot.

Kuvassa 3 on esitetty karkealla tasolla Linux-käyttöjärjestelmän ytimen tärkeimmät osat, ja se miten ydin liittyy laitteistoon sekä tarjoaa yhtenäisen järjestelmäkutsujen rajapinnan käyttäjätilan ohjelmille.



Kuva 3: Linux-käyttöjärjestelmän rakenne.

4.5 Laiteajuri

Laiteajurin tehtävänä on tarjota yhtenäinen, abstrakti liityntärajapinta käyttöjärjestelmän ja tietokoneeseen liitetyn oheislaitteen välille. Tyypillisesti laiteajuri ottaa vastaan syöttö- ja tulostuspyyntöjä käyttöjärjestelmältä ja ohjaa oheislaitetta annettujen ohjeiden mukaan [Husain & Parker 1996]. Laiteajureiden ansiosta sovellusohjelmoijan, joka haluaa käyttää tietyn tyyppistä käyttöjärjestelmän tarjoamaa palvelua, kuten tiedoston lukua, ei tarvitse ottaa kantaa siihen millainen ja millä tavalla jokin levykeasema on liitetty tietokoneeseen ja miten se toimii. Riittää kun ohjelmoija tietää minkä nimiseen tiedostoon ja missä hakemistossa hän haluaa toimenpiteensä kohdistaa, käyttöjärjestelmän ydin hoitaa kaiken muun.

Rubini & Corbet [2001] ovat jakaneet UNIXin laiteajurit kolmeen eri pääryhmään oheislaitteiden toimintatavan mukaan: *merkki-* (character) ja

lohkomuotoisiin (block mode) laitteajureihin sekä *verkkosovittimiin* (network interfaces).

Lohkomuotoiset laiteajurit ovat yleisimpiä ja ne käsittelevät ytimen *välimuistista* (cache) tulevaa ja sinne lähetettävää dataa tietyn suuruisissa lohkoissa. Alun perin lohkomuotoiset laiteajurit olivat suunniteltu käytettäväksi pelkästään levykeasemien yhteydessä. Niiden käyttö on lisääntynyt myös muissa laitteissa ja nykyään niitä käytetäänkin muun muassa synkronisten modeemien sekä joidenkin huippunopeiden kirjoittimien laiteajureissa.

Merkkimuotoiset laiteajurit eroavat kahdella tavalla merkittävästi lohkomuotoisista laiteajureista: Luku- ja kirjoitustoiminnot voidaan kohdistaa suoraan prosessin muistiavaruuteen ilman, että apuna tarvittaisiin ytimen välimuistia. Lisäksi luku- ja kirjoituspyynnöt välitetään yleensä suoraan merkkimuotoiselle oheislaitteelle. Esimerkiksi näytönohjaimen kuvamuistia on mahdollista käsitellä suoraan merkkimuotoisen laiteajurin kautta.

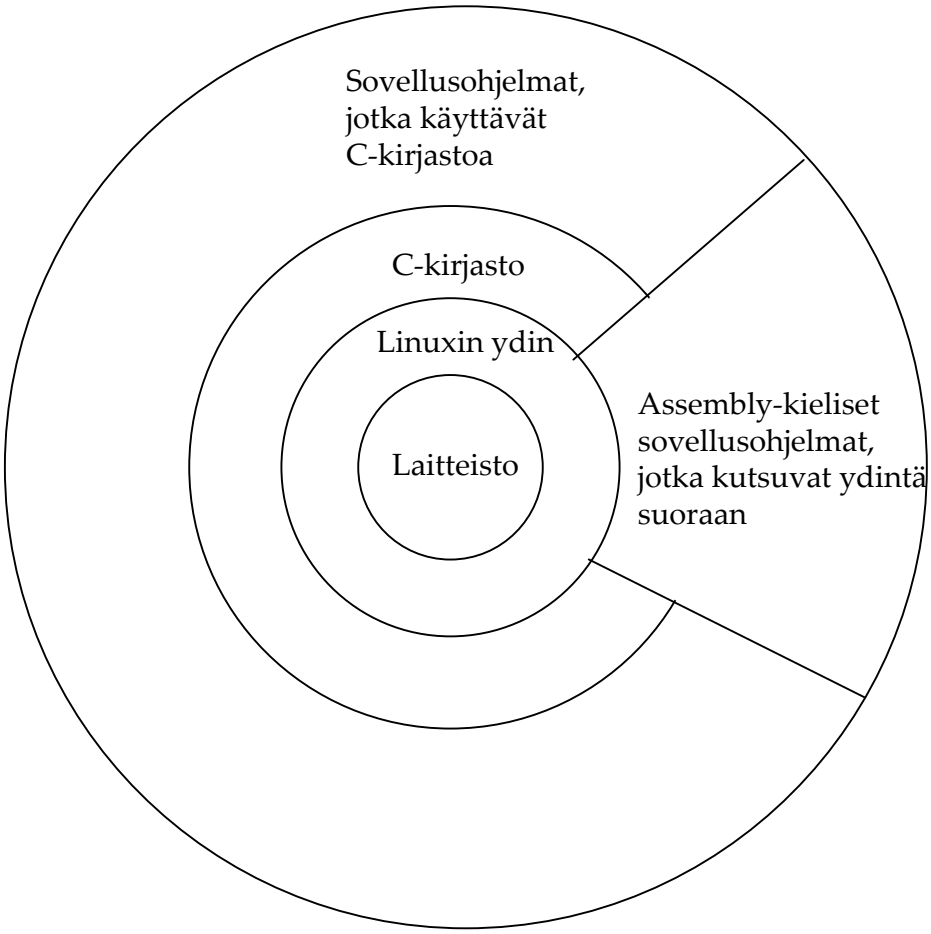
Verkkosovittimien tehtävänä on datapakettien lähettäminen ja vastaanottaminen käyttäen apuna ytimen verkkokerrosta. Sovitin ei ota kantaa miten paketti ja millä protokollalla se välitetään verkon yli vaan se ainoastaan hoitaa välitettävän tietopaketin kokoamisen ja purkamisen.

4.6 Kirjastot

POSIX-yhteensopivuus ohjelmakoodissa saavutetaan käyttämällä POSIX-standardin mukaisia funktio-kutsuja. Käyttöjärjestelmissä on tätä varten tarjolla C-*kirjasto* (library), jonka kautta ohjelmat kommunikoivat käyttöjärjestelmän kanssa. C-kirjaston tehtävä on muuntaa POSIX-kutsut käyttöjärjestelmän sisäiseen muotoon. Linuxissa tämä C-kirjasto on yleensä *glibc* (GNU C Library). Se tarjoaa suurimman osan UNIXien standardi-funktioista, kuten open(), close(), read(), write(), printf() ja niin edelleen.

Ohjelmakirjastojen yleinen tarkoitus on vähentää päällekkäisten ohjelmakoodien tarvetta keskittämällä useimmin käytetyt toimenpiteet jaettuihin kirjastoihin, jotka tarjoavat palveluitaan keskitetysti niitä tarvitseville sovellusohjelmille.

Kuvassa 4 on esitetty Linux-käyttöjärjestelmän kolme tärkeintä kerrosta: Linux-ydin, C-kirjasto sekä ohjelmistot, jotka käytävät C-kirjastoa. [Lombardo 2001]



Kuva 4: Linux-käyttöjärjestelmän tärkeimmät ohjelmistokerrokset. [Lombardo, 2001]

4.7 Sulautettu järjestelmä

Clark [Aug - Jun, 2000] määrittelee sulautetun järjestelmän tietokoneeksi, joka on integroitu isomman järjestelmän, joka ei ole yleiskäyttöinen tietokone, osaksi. Määritelmä on lyhyt ja ytimekäs, mutta siinä on yksi vakava puute; Se ei kerro mitä tarkoitetaan yleiskäyttöisellä tietokoneella. Määritelmä ei kerro meille, mikä on se todellinen ero näiden kahden järjestelmän välillä. Siksi onkin syytä selventää yleiskäyttöisen tietokoneen määritelmää.

Suurin ero yleiskäyttöisen, kuten esimerkiksi *PC*-laitteiston (Personal Computer), ja sulautetun järjestelmän välillä on *käyttöliittymä* (User Interface, UI). Yleiskäyttöiseen tietokoneeseen on yleensä liitettynä monitori, näppäimistö

ja hiiri tai jokin muu osoitinlaite, jolla sitä ohjataan. Sulautetussa järjestelmässä ei ole välttämättä käyttöliittymää laisinkaan tai se on hyvin erikoistunut käyttötarkoitustaan varten, kuten esimerkiksi painonappi, kosketusnäyttö tai suuri ohjauspaneeli. Joihinkin sulautettuihin laitteisiin on mahdollista kytkeä tavallinen näyttö tai näppäimistö, mutta se ei ole yleensä sen normaali käyttötapa. Ulkoisten lisälaitteiden liitäntämahdollisuus on tarkoitettu lähinnä väliaikaista käyttöä varten. Tällaisissa tapauksissa on monesti kyse laitteen huollosta tai vian etsinnästä.

Edellä mainittu tapa erottaa yleiskäyttöinen ja sulautettu järjestelmä toisistaan ei kuitenkaan ole vielä täysin riittävä. Ratkaisevin ero näiden järjestelmien välille syntyy siinä, että sulautettu järjestelmä pyrkii olemaan kustannustehokas ratkaisu tiettyyn ongelmaan. Se voi esimerkiksi suoriutua kymmenestä eri tehtävästä, mutta nämä kymmenen asiaa ovat ne, joihin se tulee ikinä kykenemään. Esimerkiksi autossa sulautettu järjestelmä voi säätää polttoaineen annostelua palotilaan tai kerätä kriittistä tietoa moottorin tilasta, mutta siinä kaikki. Yleiskäyttöiset tietokoneet ovat joka paikan höyliä, sillä ne toimitetaan tehtaalta ilman tiettyä päätehtävää. Niillä voi tehdä lukemattomia eri asioita, kuten pelata pelejä, surffata Internetissä tai vaikkapa kirjoittaa tutkielman. [Lombardo, 2001]

Yleiskäyttöisten tietokoneiden toiminnallisuutta on myös mahdollista laajentaa helposti laittamalla CD-levy CD-asemaan ja asentamalla uusi ohjelma koneen kovalevylle. Tällaista mahdollisuutta ei sulautetuissa järjestelmissä yleensä ole, pois lukien ohjelmistopäivityksistä johtuvaa tarvetta. Ohjelmistopäivityksillä pyritään poistamaan esimerkiksi järjestelmässä olevia tietoturvaongelmia tai muita mahdollisia ohjelmointivirheitä.

Monissa sulautetuissa järjestelmissä on yleensä myös tarvetta reaaliaikaisuuteen, johtuen niiden erikoistuneesta käyttötarkoituksesta. Tällaista vaatimusta yleiskäyttöisillä tietokoneilla ei ole, eivätkä niiden käyttöjärjestelmät edes mahdollista reaaliaikaisuutta.

4.8 Reaaliaikaisuus

Reaaliaikaisuudella tarkoitetaan sitä, että järjestelmän on pystyttävä vastaamaan ajallisiin ja ulkoisiin tapahtumiin riittävän nopeasti, ennalta tunnetulla viiveellä ja joka ikinen kerta. Reaaliaikaisessa järjestelmässä käyttöjärjestelmän on aina pidettävä huolta siitä, että kriittiset ja

reaaliaikaisuutta vaativat toimenpiteet tulevat tehdyiksi juuri sillä hetkellä, kun ne on tarkoitettu suorittaa.

Jotta reaaliaikaisuuden vaatimiin vasteaikoihin, jotka ovat luokkaa alle 15 mikrosekuntia [FSMLab RTLinux], olisi mahdollista päästä, on käyttöjärjestelmän pystyttävä asettamaan ulkoiset ja sisäiset tapahtumat salamannopeasti tärkeysjärjestykseen. Jos järjestelmä ei kykene suorittamaan annettua aikakriittistä tehtävää ajallaan, on se epäonnistunut tehtävässään.

4.8.1 Järjestelmän ennakoitavuus

Järjestelmän, jossa voimme ennakoida etukäteen huonoimman mahdollisen suoritustilanteen vasteajan, sanotaan olevan ennakoitava. Toisaalta järjestelmä, jonka vasteajan arvioiminen ennalta on vaikeaa tai mahdotonta esimerkiksi siksi, että se muuttuu jatkuvasti, kutsutaan ei-ennakoitavaksi järjestelmäksi. Reaaliaika-ohjelmistot vaativat ennakoitavaa käyttöympäristöä ja sellaisen ympäristön voi tarjota ainoastaan reaaliaikakäyttöjärjestelmä.

Yleiskäyttöisessä tietokoneessa on hyvin yleistä, että järjestelmä on ns. *varatussa tilassa* (busy), jonka käyttäjä havaitsee normaalisti esimerkiksi Microsoft Windows -käyttöjärjestelmässä kohdistimen paikalla olevasta tiimalasista. Tällöin käyttöjärjestelmä suorittaa jotain sellaista toimintoa, joka vaatii niin paljon *prosessointiaikaa* (CPU-time), että myös kaikki muut rinnakkaisprosessit järjestelmässä hidastuvat tai niiden suorittaminen on hetkellisesti täysin pysähdyksissä. Rinnakkainen suoritus eli *moniajo* (multitasking) ei siis yksinään takaa sitä, että järjestelmä pystyy reaaliaikaisuuteen, sillä huolimatta suoritustasoista, toimenpiteiden suoritus- ja *vasteaikaa* (response time) pituutta ei voida tietää ennalta. Vasteajan ennakoimisen tekee lähes mahdottomaksi se, että sen pituuteen vaikuttavat muun muassa ytimessä saman aikaisesti käsiteltävien *keskeytysten* (interrupts) määrä.

Reaaliaikaisuus on mahdollista toteuttaa yksinkertaisimmillaan käyttöjärjestelmässä, joka ei tue moniajoa laisinkaan. Tällaisesta käyttöjärjestelmästä on hyvä esimerkki jo käytöstä poistunut *DOS* (Disk Operating System) -käyttöjärjestelmä. Siellä pääohjelman silmukka on ainoa suoritettava prosessi ja se saa halutessaan kaikki järjestelmän resurssit vapaasti käyttöönsä, koska sillä ei ole muita prosesseja tai säikeitä kilpailemassa suoritusvuorosta ja resursseista.

4.8.2 Vasteaika

Kun suorituksen vasteaika on riittävän pieni, suoritusta voidaan sanoa reaaliaikaiseksi. Vasteajalla tarkoitetaan sitä aikaväliä, kun tehtävän suoritus käynnistetään, esimerkiksi ulkoisesta ärsykkeestä keskeytyksellä, siihen hetkeen, kun suoritus päättyy ja ohjelma on tuottanut halutun vasteen. [Dankwardt K, Jan 2002]

4.8.3 Pehmeä vs. kovareaaliaika

Reaaliaikajärjestelmät voidaan jakaa eri luokkiin niiden luotettavuusvaatimusten mukaan. Luokat ovat *pehmeän reaaliajan* (soft-realtime) järjestelmät sekä *kovan reaaliajan* (hard-realtime) järjestelmät [Mullender S., 1993]. Joissain yhteyksissä esiintyy myös kolmas luokka, jota kutsutaan nimellä *puolikova reaaliaika* (firm-realtime). Koska kaksi ensimmäistä ovat käytössä yleisempiä, jätetään puolikovan reaaliajan käsittely pois ja mainittakoon kuitenkin, että se eroaa pehmeästä ja kovasta reaaliajasta siten, että siinä on lyhyempi pehmeän-reaaliajan aikaraja sekä pitempi kovan-realiajan aikaraja.

Pehmeän reaaliajan järjestelmissä vasteajan ylityksellä hetkellisesti ei ole vakavia seurauksia. Järjestelmä voi toimia aivan normaalisti jos virheitä sattuu satunnaisesti. Käytännössä tämä voi tarkoittaa vaikka sitä, että puhelu yhdistyy vahingossa väärään numeroon [Mullender S., 1993], mutta mitään muita vakavampia seurauksia virheellä ei ole ja yleensä uusintayritys tuottaa paremman tuloksen. Tällaisiin ongelmiin laitteiden kanssa meistä jokainen on varmasti törmännyt jossain vaiheessa elämäänsä.

Kovan reaaliajan järjestelmissä vasteajan ylitys on aina vakava virhetilanne. Vasteaika on aina saavutettava eli järjestelmän on pystyttävä koska ja missä tilanteessa tahansa reagoimaan alle ennalta sovitun aikarajan puitteissa pyyntöihin. Järjestelmän ylittäessä aikarajan suorituksen aikana edes kerran voi aiheuttaa kohtalokkaita seurauksia. Kyseessä saattaa olla pahimmassa tapauksessa ihmishenkien menetys tai suuri taloudellinen katastrofi. [Dankwardt K, Jan 2002]

4.9 Moniajo

Moniajolla tarkoitetaan järjestelmää, joka mahdollistaa usean samanaikaisen prosessin (*Process)* suorittamisen käyttäen jaettuja laitteistoresursseja, kuten esimerkiksi yhtä suoritinta. Moniajo voidaan jakaa kahteen aliluokkaan.

Keskeyttävään (pre-emptive), jota kutsutaan joissain yhteyksissä myös aidoksi moniajoksi, sekä *vuoropohjaiseen* (co-operative) moniajoon. [Peltomäki & Linjama, 1999]

4.9.1 Vuoropohjainen moniajo

Vuoropohjainen moniajo perustuu siihen, että jokainen prosessi saa vuorollaan kaiken prosessointiajan itsellensä. Suoritettuaan haluamansa ajan omia tehtäviään prosessi luovuttavaa suoritusvuoron seuraavalle prosessille omaehtoisesti. Vuoropohjaisen moniajon käyttöjärjestelmässä huonosti käyttäytyvä prosessi voi omalla suoritusvuorollaan kaatuessaan lamauttaa muiden ohjelmien toiminnan ja sen seurauksena pahimmassa tapauksessa koko käyttöjärjestelmä voi jumiutua.

Tätä menetelmää ei enää käytetä kehittyneissä käyttöjärjestelmissä, mutta se oli aikoinaan hyvin suosittu. Nykyään vielä käytössä olevista käyttöjärjestelmistä muun muassa Windows 9x -sarjan sekä Macintosh System V 7.x käyttävät vielä vuoropohjaista moniajoa.

4.9.2 Keskeyttävä moniajo

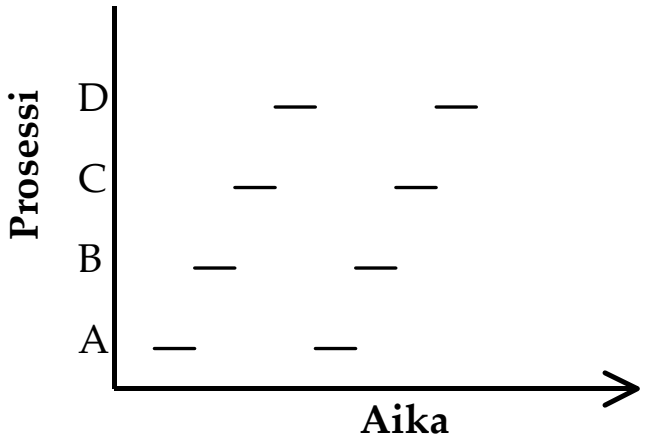
Keskeyttävässä moniajossa sovellusohjelmat pyytävät suoritusaikaa käyttöjärjestelmältä. Mikään yksittäinen prosessi ei voi ottaa käyttöjärjestelmän kaikkia resursseja yksin haltuunsa. Käyttöjärjestelmän tehtävä on hoitaa resurssien jakaminen prosessien kesken tasapuolisesti, ottaen huomioon niiden tärkeysjärjestyksen. Koska keskeyttävän moniajon toimintatapa on niin erilainen vuoropohjaiseen moniajoon nähden, tätä resurssien jakamista voi mielestäni kutsua resurssien lainaamiseksi, sillä käyttöjärjestelmä ei luovuta missään vaiheessa hallintaoikeutta prosessille, kuten vuoropohjaisessa moniajossa. Muun muassa Linuxin ydin perustuu keskeyttävään moniajoon.

Keskeyttävän moniajon tehokkuuden kannalta merkittävimmässä asemassa on käyttöjärjestelmän kyky jakaa suoritusaikaa prosessien välillä joustavasti, ottaen kuitenkin samalla huomioon niiden tärkeysjärjestyksen. Suoritusjärjestyksen määrää ytimen *aikataulutin* (scheduler).

4.9.3 Aito vs. näennäinen moniajo

Jos laitteistossa on vain yksi suorituksesta vastaava suoritin, riippumatta siitä miten moniajo on toteutettu järjestelmän moniajavassa

käyttöjärjestelmässä, tällaiset järjestelmät ovat käytännössä tarkemmin tarkasteltuna vain näennäisesti moniajavia. Tämä johtuu siitä, että yhden suorittimen järjestelmässä käyttöjärjestelmä vaihtaa suoritusvuorossa olevaa prosessia niin nopeasti, että pitemmässä aikaikkunassa asiaa tarkasteltaessa näyttää siltä kuin useampaa prosessia suoritettaisiin samanaikaisesti. Teknisesti tämä on täysin mahdotonta, sillä nykyiset *suorittimet* (processors) eivät kykene suorittamaan, kuin yhtä prosessia kerrallaan. Markkinoille on tosin juuri tullut ensimmäiset työpöytä- että palvelinkäyttöön tarkoitetut Intelin valmistamat P4 Northwood tuoteperheen suorittimet, jotka mahdollistavat kahden rinnakkaisen säikeen suorittamisen samanaikaisesti. Käyttöjärjestelmälle nämä HyperThreading-tekniikkaa hyödyntävät suorittimet näkyvät kahtena erillisenä loogisena suorittimena. Toistaiseksi tekniikan hyöty on kuitenkin jäänyt vaatimattomaksi. Sovelluksesta riippuen HyperThreading -tekniikalla saavutetaan 0 – 10 % suorituskyvyn kasvun vastaavaa ei-HyperThreading-suorittimeen verrattuna. [Muropaketti, 2002]



Kuva 5: Moniajossa suoritusaika jaetaan suoritettavien prosessien kesken aikaviipaleina.

Moniajo tarkoittaa käytännössä sitä, että jos prosessien suoritusta tarkastellaan riittävän pienessä aikaikkunassa, niin voidaan havaita että vain yksi prosessi on suoritusvuorossa kerralla. Prosessi saa tällöin vain pienen siivun *suoritusaikaa* (time slice) kerrallaan. Käyttöjärjestelmästä riippuen prosessi voi saada suoritusvuoron esimerkiksi 10 millisekunnin välein. Kuva 5 havainnollistaa, miten moniajossa prosesseja suoritetaan vuoronperään. Täydellisempään moniajoon päästään vasta, kun järjestelmässä on useampi suoritimia, jolloin prosesseja voidaan suorittaa aidosti samanaikaisesti eri suorittimilla. Optimaalisesti moniajavassa järjestelmässä on oltava yhtä monta suoritinta, kuin on suoritettavia prosesseja. Useamman suorittimen ollessa kyseessä puhutaan moniprosessoroinnista.

4.9.4 Moniprosessorointi

Moniprosessorointi on teknisesti erittäin hankalaa, sillä se vaatii käyttöjärjestelmän prosessien välistä kommunikointia. Pahimmassa tapauksessa se saattaa jopa hidastaa järjestelmän toimintaa yhdellä suorittimella varustettuun järjestelmään verrattuna. Tämä johtuu siitä, että prosessien välinen kommunikointi vaatii käyttöjärjestelmältä paljon työtä ja resursseja, joiden avulla prosessien kuormitus saadaan jaettua optimaalisesti suorittimien kesken. Lisäksi prosessien välinen kommunikointi vaatii suuria määriä tiedonsiirtoa, eikä kaikkia tehtäviä ole mahdollista pilkkoa ja suorittaa erillisissä säikeissä. Moniprosessoroinnin onnistumisen kannalta tärkein ehto on ohjelmiston tuki säikeille.

4.10 Aikataulutin

Moniajavan käyttöjärjestelmän aikatauluttimen tulee päättää mikä suoritusvuoroa odottava prosessi tulee suoritetuksi seuraavaksi. Aikataulutuksesta vastaavan algoritmin tulee minimoida suoritusaikaa odottavan prosessin odotusaika.

Yksinkertaisin aikataulutusalgoritmi on nimeltään *round-robin*. Se suorittaa prosesseja ennalta tunnetussa järjestyksessä ja aina kiinteällä aikasiivulla. Kun suoritusvuorossa oleva prosessi valmistuu tai aikasiivu päättyy, suoritusvuoron saa seuraava saman suoritustason prosessi, joka on valmiina (READY) odottamassa suoritusvuoroaan. Jos korkeimman suoritustason prosessi pyytää suoritusaikaa, sitä palvellaan seuraavan *prosessinvaihdon* (context switch) yhteydessä. Muita käytössä olevia aikataulutus-algoritmeja ovat muun muassa *FIFO* sekä *adaptive sporadic*.

Jos käyttöjärjestelmässä on käytössä *suoritustasot* (priority level), prosessin valinta tapahtuu niiden perusteella. Ohjelmia suoritetaan yleensä normaali-suoritustasolla. Jos usealla prosessilla on sama suoritustaso, prosessit joutuvat kilpailevat suoritusajasta keskenään. Korkeamman suoritustason omaavalla prosessilla on aina etuoikeus suoritusaikaan alemmalla suoritustasolla ajettaviin prosesseihin nähden. Alimmalla eli *lepotilan* (idle) -suoritustasolla ajettavat prosessit saavat käyttöönsä sen suoritusajan, joka korkeamman suoritustason omaavilta prosesseilta on jäänyt käyttämättä.

4.11 Suoritustasot

Nykyaikaisessa moniajavassa käyttöjärjestelmissä on yleensä käytössä suoritustasot, joiden avulla prosesseja asetetaan tärkeysjärjestykseen jakamalle ne eri luokkiin suoritustasojen perusteella. Suoritustasoja voi olla käyttöjärjestelmästä riippuen muutamasta useisiin satoihin. Karkeasti jaettuna prioriteettitasot ovat: *lepotila* (idle), *matala* (low), *normaali* (normal) ja *korkea* (high). Lisäksi joissain käyttöjärjestelmissä on tarjolla *reaaliaikainen* (realtime) suoritustaso. Viimeisin suoritustaso normaaleissa, yleiskäyttöön tarkoitetuissa, käyttöjärjestelmissä tarkoittaa käytännössä sitä, että muiden prosessien suoritus lopetetaan lähes kokonaan, jolloin järjestelmästä tulee lähes käyttökelvoton, sillä järjestelmä ei enää vastaa minkään muun alemman suoritustason prosessien pyyntöihin.

4.12 Säikeet

Moniajon mahdollistava tekniikka on suoritusajan jakaminen sovellusohjelmien eli prosessien kesken säikeisiin. Jokainen prosessi suoritetaan ytimen alaisuudessa omassa, muista erillään olevassa säikeessä.

Omassa säikeessä suoritettava prosessi voi, käyttöjärjestelmästä riippuen, luoda myös omia aliprosesseja säikeisiin ja näin parantaa ohjelmiston toimivuutta ja joustavuutta. Raskas ja pitkä toiminto voidaan suorittaa erillään omassa työsäikeessä, jolloin pääsäie ei jumiudu kokonaan pitkän tehtävän aikana ja pääohjelma voi jatkaa toimintaansa normaalisti.

4.13 Keskeytykset

Keskeytykset ovat oheislaitteelta tulevia signaaleja, jotka ilmoittavat, että oheislaite haluaa huomiota käyttöjärjestelmältä. Yleensä keskeytyksiä syntyy, kun oheislaite on suorittanut sille annetun tehtävän ja se on valmis suorittamaan seuraavaa [Husain & Parker 1996]. Kun keskeytys on *saatu kiinni* (asserted), suoritin lopettaa suorittamasta sillä hetkellä suorittamassa olevaansa tehtävää. Käyttöjärjestelmä ohjaa keskeytyksen laiteajurille, joka on rekisteröinyt itsensä tietyn keskeytyksen kuuntelijaksi. Tämän jälkeen kyseinen laiteajuri suorittaa keskeytyksen nostaneelle oheislaitteelle tarvittavat toimenpiteet.

5 Sulautettu Linux

Seuraavaksi tulen käsittelemään asioita, jotka tulee ottaa huomioon valittaessa Linuxia sulautetun järjestelmän käyttöjärjestelmäksi. Tällaisia asioita ovat muun muassa laitteisto ja laiteajurit sekä reaaliaikaisuuden tarve.

5.1 Laitteisto

Merkittävin muutos, joka on tapahtunut vuosien saatossa sulautettujen järjestelmien laitteistossa, on ollut siirtyminen yhä kasvavissa määrin PC-yhteensopiviin laitteistokomponentteihin. PC-laitteiston osia on, massatuotannosta johtuen, helposti saatavilla ja niiden hinnat on todella edullisia räätälöityihin erikoiskomponentteihin verrattuna. Tästä on seurannut se, että sulautettujen laitteiden hinnat ovat romahtaneet ja nykyään on mahdollista valmistaa edullisesti sellaisia sulautettuja laitteita, joita ei vielä hetki sitten olisi kannattanut edes suunnitella. [Epplin, 1997]

Sulautettu järjestelmä on mahdollista toteuttaa myös ei-PC-laitteistolla, jos niin halutaan. Suunnittelijoilla on lähes vapaat kädet, sillä Linux ei rajoita laitteistoalustan valintaa. Linuxia voi mielestäni kutsua laitteistoriippumattomaksi käyttöjärjestelmäksi, koska se toimii lähes kaikilla ja yleisimmillä nykyään käytössä olevilla eri laitealustoilla ja arkkitehtuureilla. Jos tämä määritelmä ei miellytä jotakin, on meidän tyydyttävä kutsumaan Linuxia rajoitetummin *moniarkkitehtuurisuutta* (multiarchitecture) tukevaksi käyttöjärjestelmäksi, sillä Linux-käyttöjärjestelmä on tähän päivään mennessä *sovitettu* (ported) muun muassa seuraaville laitteistoarkkitehtuureille: x86, Power PC, Alpha, Sparc, SMP, Motorola 68000, ARM, IPS, PalmPilot, PocketPC ja niin edelleen. Kuvassa 6 on Microsoftin PocketPC standardin mukainen Compaq iPAQ kämmentietokone, johon on asennettu Linux-käyttöjärjestelmä X-ikkunointijärjestelmän kanssa.

Jos Linuxia on tarvetta ajaa muilla, kuin jo entuudestaan tuetuilla laitealustoilla, niin siitäkään ei muodostu ylitsepääsemätöntä ongelmaa. Linux on sovitettavissa kaikille 32- ja 64-bittisille arkkitehtuureille, joissa on *sivutetun muistin tukiyksikkö* (Paged Memory Management Unit, PMMU), ja saatavilla on samalle arkkitehtuurille tarkoitettu C-kääntäjä, kuten esimerkiksi *GNU C Compiler* eli lyhemmin *gcc*. [Linux Kernel Archives] Linux onkin maailman sovitetuin tarjolla olevista käyttöjärjestelmistä [Torvalds, 1999]. Eräät Linuxia kehittävistä ja sille tukea tarjoavista organisaatioista, joihin muun muassa Red Hat kuuluu, tarjoavat niin sanottua sovituspalvelua, jonka avulla asiakas voi

halutessaan saada Linuxin toimimaan arkkitehtuurissa, jolle ei entuudestaan ole tarjolla tukea.

Kuva 6: Linux Compaq iPAQ PocketPC -laitteessa. [iPAQLinux.com]

5.2 Laiteajurit

Jos sulautettu laitteisto koostuu tunnetuista ja yleisesti käytössä olevista PC-laitteistokomponenteista, on oheislaitteen käyttöönoton kannalta tärkeimmän ohjelmisto-osan eli laiteajurin saaminen huomattavasti helpompaa, kuin harvinaislaatuisemmalle oheislaitteelle. Laitteistoa suunniteltaessa tämä asia kannattaa ottaa erityisen tarkasti huomioon.

Aina ei kuitenkaan ole mahdollista valita sellaisia oheislaitteita, joille on olemassa käyttövalmiit laiteajurit, johtuen esimerkiksi niiden erityisistä ominaisuuksista, omista tarpeista sekä harvinaislaatuisuudesta. Aivan kaikille oheislaitteille Linuxissa ei ole tukea valmiina vaikka se onkin yleisesti ottaen hyvin tuettu käyttöjärjestelmä yleisempien oheislaitteiden, kuten verkkokorttien, näytönohjainten, massamuistien ja niin edelleen, osalta.

Eteen saattaa tulla tilanne, jossa laitteeseen, johon Linux on tarkoitus sulauttaa, on valittu oheislaitteita, joille ei ole saatavilla suoraa tukea edes uusimmassa tarjolla olevassa Linuxin ytimen versiossa tai suoraan oheislaitteen valmistajalta. Jos vaihtoehtoista oheislaitetta, joka hoitaisi vastaavan asian, ei ole saatavilla tai ne ovat jostain muusta syystä, kuten esimerkiksi teknisiltä ominaisuuksiltaan epäkelpoja käytettäväksi omassa laitteessa, ainoaksi

vaihtoehdoksi tällaisissa tapauksessa jää oheislaiteajurin tekeminen itse tai sen teettäminen ulkopuolisella. Ulkopuolisia ohjelmiston tarjoajia kutsutaan *kolmansiksi osapuoliksi* (third party). Laiteajurin tekeminen itse vaatii huomattavan paljon tuntemusta käyttöjärjestelmän sekä oheislaitteen puolesta. Toisaalta laiteajurin teettäminen puolestaan vaatii paljon rahaa, jos apua pyydetään kaupalliselta yritykseltä. Linux-maailmassa ei ole kuitenkaan täysin poissuljettua se, että joku tekee tarvittaessa laiteajurin pelkästä pyynnöstä ja vieläpä aivan ilmaiseksi. Linuxin yksi kantavista voimista on nimittäin ollut aina yhteen hiileen puhaltaminen eli toisia autetaan mielellään yhteisen edun nimissä. Lisäksi tekijää motivoi monesti maine, jota hän saa suorituksellaan Linuxin eteen.

Johtuen laiteajurien laitteistoläheisyydestä, niiden tekeminen vaati erityisosaamista ja kokemusta. Suunnittelijan on tunnettava hyvin käyttöjärjestelmän rakenne, ytimen toiminta sekä oheislaite ja hänen on oltava erittäin kokenut ohjelmoija. Saattaa olla, että laiteajurin tekeminen on taloudellisesti sekä ajallisesti järkevämpää teettää mahdollisuuksien mukaan ulkopuolisella taholla, kuten esimerkiksi oheislaitteen valmistajalla, jos omat resurssit tai kokemus laiteajurin tekemiseen ovat riittämättömät.

Jos laiteajuri päätetään tai syystä tai toisesta joudutaan tekemään itse, nousee esille yksi Linuxin parhaista puolista: Avoin lähdekoodi. Maailmalla on tarjolla valtavasti referenssilähdekoodia. Voit ottaa esimerkiksi vastaavanlaisen valmiin laitteen laiteajurin lähdekoodin ja muokata sitä siten, että se vastaa omia tarpeitasi. Tässä tapauksessa joudutaan yleensä pyytämään valmistajalta tarkat määritykset sekä tekniset tiedot, jotka kertovat miten oheislaite toimii ja miten se saadaan liitettyä käyttöjärjestelmän ytimen alimpana olevaan ohjelmistokerrokseen. Kaikki valmistajat eivät halua saattaa oheislaitteidensa määrityksiä lainkaan julkisuuteen, niiden arkaluontoisuutensa takia tai niiden saanti voi olla rajoitettu ainoastaan suurille ja nimekkäille sopimusvalmistajille. Tästä asiasta minulla on henkilökohtaisesti kokemuksia VIA Technologyn osalta, josta sain kuulla ikävän vastauksen tiedustellessani emolevyni piirisarjaan liitetyn AC97-yhteensopivan äänipiirin tietoja, jonka *lähdekoodittomasti* (binary only) julkaistu laiteajuri ei suostunut toimimaan uudemman emolevyversioni kanssa Linuxissa. VIA:lta tarvitsemani tiedon ja referenssiajureiden avulla olisin saanut äänipiirin toimimaan Linuxissa.

Lähdekoodeja ei siis välttämättä ole aina julkisesti saatavilla, vaikka tuki oheislaitteelle on muuten olemassa Linuxissa. Tästä saattaa olla haitta tapauksissa, joissa laiteajurissa esiintyy ongelmia tai sitä olisi muuten tarvetta

muuttaa, jotta se vastaisi paremmin sulautetun järjestelmän vaatimuksia. Yleensä tällaiset lähdekoodittomat laiteajurit ovat myös valitettavan usein sidottu tiettyyn ytimen versioon, joka ei välttämättä vastaa sitä versiota, jota sulautetun järjestelmän suunnittelijat ovat ajatelleet käyttää. Linuxin ytimen versioita tulee nykyään ulos tiheää tahtia ja valitettavan usein lähdekoodittomat ajurit laahaavat yhteensopivuudessa monta ytimen versionumeroa perässä. Ajallisesti tämä voi tarkoittaa kuukausia. Tällaisessa tapauksessa ei ole muuta vaihtoehtoa, kuin turvautua valmistajien tarjoamiin palvelukanaviin. Ne ovat yleensä ruuhkaisia, joten muutoksien toteuttaminen ja tuen saaminen uusimmille ytimen versioille saattaa kestää todella kauan. Pahimmassa tapauksessa muutoksia ei toteuteta asiakkaan haluamassa ajassa tai ei koskaan.

5.3 Skaalautuvuus ja joustavuus

Koska Linux on modulaarinen käyttöjärjestelmä, kaikki ylimääräinen on erotettu sen ytimestä. Ytimen ulkopuolisia osia ovat muun muassa komentotulkki, yksinkertaiset tiedosto- ja levynkäsittelykomennot, kuten esimerkiksi *ls*, *cp*, *rm*, *mkdir* ja niin edelleen sekä kirjastot, joiden kautta sovellusohjelmat käyttävät käyttöjärjestelmän palveluita. Tämä mahdollistaa sen, että järjestelmän kokoonpano on täysin muunneltavissa omiin tarpeisiin. Parhaimmillaan sulautetussa Linux-järjestelmässä ei ole ytimen lisäksi muuta, kuin itse sulautettu sovellusohjelmisto [Gemmel & Highton, 2000], joka on mahdollista suorittaa järjestelmän ainoana käyttäjätason prosessina ilman komentotulkkia.

Vaikka Linuxin ydin on rakenteeltaan monoliittinen, on sen sisältämät ominaisuudet täysin muunneltavissa. Ytimeen ei tarvitse ottaa mukaan kuin niiden oheislaitteiden ajurit, jotka omassa laitteistossa on käytössä [Gemmel & Highton, 2000]. Esimerkiksi mp3-soittimessa ei ole tarvetta verkoille eikä verkkoprotokollille, *RAID* (Redundant Array of Independent Disks, alun perin Redundant Array of Inexpensive Disks) -kovalevyohjaimille eikä myöskään VGA-näytöille ja niin edelleen.

Ladattavien moduulien avulla perusydin voi olla useassa eri laitteessa täysin sama. Ajonaikana, kuten esimerkiksi laitteen käynnistyessä tarvittavat laiteajurit ladataan dynaamisesti muistiin ja linkitetään ytimeen sen perusteella mikä kyseisen laitteen oheislaitekokoonpano on sillä hetkellä.

Edellä mainittujen Linuxin ytimen ominaisuuden perusteella järjestelmän vaatiman tallennustilan ja keskusmuistin määrä voidaan säätää käyttökohteen

mukaan optimaalisesti. Sulautettua järjestelmää suunniteltaessa tulee kuitenkin ottaa huomioon, että Linuxin ydintä ei saa parhaassakaan tapauksessa mahdutettua pakattuna pienempään tilaan, kuin noin 400 kilotavua. Tämän päälle tulee lisäksi tarvittavien sovellusohjelmien vaatima tilan tarve. Tyypillisesti Linuxin ytimen koko vaihtelee 500 - 600 kilotavun välillä. Kokonainen Linux-käyttöjärjestelmä graafisine X-ikkunointijärjestelmineen on mahdollista saada toimimaan laitteistossa, jossa on pienimmillään neljä megatavua tallennustilaan sekä 16 megatavua keskusmuistia. Resurssien määrän tarve asettaa selkeän rajoituksen Linuxin käyttökohteille. Sitä ei voi käyttää sulautettuna käyttöjärjestelmänä järjestelmässä, jonka on mahduttava todella pieneen tallennustilaan ja on tultava toimeen äärimmäisen pienillä resursseilla. Pienellä resurssitarpeella tarkoitan järjestelmää, jonka on mahduttava megatavun tallennustilaan ja toimittava alle neljän megatavua keskusmuistilla. [LynuxWorks.A, 1987 - 2000]

5.4 Siirrettävyys

”Write once, run anywhere” eli vapaasti suomennettuna “ohjelmoi kerran ja aja missä vain” kuvaa hyvin Linuxia. Kuten totesin jo kappaleessa 5.1, jossa kerroin Linuxin moniarkkitehtuurisuuden tuesta, Linux on maailman sovitetuin käyttöjärjestelmä eli se toimii lähes poikkeuksetta missä ja millä tahansa laitealustalla. Tämä onkin yksi Linuxin suurimmista eduista. Jos sulautetun järjestelmän laitteistoarkkitehtuuria tai laitealustaa, jolla sitä on tarkoitus ajaa, ei ole lyöty lukkoon tai sitä halutaan muuttaa, on erittäin todennäköistä, että Linux on sovitettavissa myös tälle laitteistoarkkitehtuurille. Sovellusohjelmistoa ei tarvitse muuttaa uuden laitteistoarkkitehtuurin takia vaan riittää, kun sovellusohjelman lähdekoodit käännetään uudestaan kyseiselle laitteistoarkkitehtuurille. Jos sovellusohjelma toimii yhdellä Linuxin tukemalla laitteistoarkkitehtuurilla, niin se toimii myös kaikilla muilla alustoilla ilman mitään muutoksia lähdekoodeihin.

Linuxin siirrettävyyttä voi hyödyntää jo tuotekehitysvaiheessa siten, että vaikka *kohdealusta* (target), jolla sulautettua Linux-järjestelmää on tarkoitus ajaa, ei olisikaan vielä valmis, on ohjelmistoa mahdollista kehittää millä tahansa laitteistolla, jolla Linux toimii. Monesti kehitysalustana käytetäänkin esimerkiksi samaa PC-laitteistoa, jolla muuten työskennellään; Luetaan sähköpostit ja kirjoitetaan dokumentit. Tämä nopeuttaa osaltaan tuotekehityksen läpivientiä, aikaistamalla sulautetun järjestelmän pystyyn saattamista, jolloin järjestelmässä mahdollisesti olevien *virheiden etsintä* (debug) ja niiden poistaminen voidaan aloittaa mahdollisimman aikaisessa vaiheessa.

Samalla helpotetaan laitteiston kehitystä, sillä toisella laitteistolla etukäteen kehitettyä ja testattua ohjelmistoa oikealla laitteistolla testattaessa voidaan epäselvissä ongelmatilanteissa ainakin osassa virheistä ohjelmiston osuus sulkea pois mahdollisuuksien joukosta.

5.5 Reaaliaikaisuus

Koska yleiskäyttöön tarkoitetut käyttöjärjestelmät, joihin myös Linux lukeutuu, ovat alun perin suunniteltu työpöytä ja palvelin käyttöön, ei niiden ytimissä ole alun perin otettu huomioon reaaliaikaisuuden vaatimuksia. Käytännön totuus on siis se, että Linuxinkaan vakioydin ei pysty täyttämään reaaliaikakäyttöjärjestelmälle asetettuja vaatimuksia, huolimatta siitä, että se tarjoaa sovellusohjelmille keskeyttävää moniajoa.

Linux on suunniteltu tarjoamaan jokaiselle suoritettavalle *tehtävälle* (task), sen suoritustasosta riippumatta, tasapuolisesti suoritinaikaa ja muita sen tarvitsemia resursseja [Ortiz, 2001] siten, että järjestelmän kokonaissuorituskyky olisi mahdollisimman hyvä. Tämä sotii reaaliaikaisuutta vaativia järjestelmiä vastaan, sillä niissä on tarkoitus palvella tarvittaessa muutamaa prosessia mahdollisimman nopeasti ja kaikin mahdollisin keinoin, muiden prosessien kustannuksella. Ratkaisu tilanteeseen ei ole se, että prosessille asetetaan niin sanottu reaaliaikainen suoritustaso, sillä ongelma piilee syvemmällä käyttöjärjestelmän ytimessä ja ytimen toimintatavoissa hoitaa moniajo.

5.5.1 Ytimen ongelmat

Linux toteuttaa POSIX 1003.13 määrityksen mukaisen standardin; Miten monenkäyttäjän reaaliaikajärjestelmän tulee toimia muistin lukituksen, aikataulutuksen sekä reaaliaikasignaalien osalta. Määrityksen mukaan reaaliaikaisella suoritustasolla ajettava prosessi voidaan lukita keskusmuistiin, jotta sitä ei suoritusaikana sivutettaisi virtuaaliseen muistiin, kuten esimerkiksi kovalevylle. Aikataulutin puolestaan huolehtii siitä, että prosessit tulevat aina suoritettua ennalta tunnetussa järjestyksessä. Vaikka POSIX 1003.13 -määrityksen ansiosta järjestelmän suorituskyky paranee huomattavasti, ei käyttöjärjestelmä silti saavuta kaikkia kovaa reaaliaikaisuutta vaativien prosessin vaatimuksia, sillä korkeamman suoritustason prosessi voi tulla syrjäytetyksi ytimessä tapahtuvien toimenpiteiden takia [Varhol, 26 June 2000]. Tämä tarkoittaa myös sitä, että ytimen suorittaessa jotain tehtävää ei mitään

prosessien pyyntöjä pystytä käsittelemään, jolloin myös reaaliaikaprosessit joutuvat odottamaan.

Linuxin ytimen kykenemättömyys reaaliaikaisen käyttöjärjestelmän vaatimuksiin johtuu käytännössä siis siitä, että muiden käyttöjärjestelmien tapaan Linuxin ydin estää ajoittain kaikkien keskeytysten käsittelyn hyvinkin pitkäksi aikaa. Tästä aiheutuu se, että vaikka Linuxissa monien muiden käyttöjärjestelmien tapaan on tarjolla prosesseille reaaliaikainen suoritustaso, jonka omaava prosessi tulee aina suoritettua ennen muita suoritustasoja, tämäkin prosessi tulee kaikesta huolimatta syrjäytetyksi ytimen toimesta.

Koska Linux estää keskeytykset suorittaessaan toimintoja ytimen tasolla, ei järjestelmä enää kykene palvelemaan tulevia ulkoisia laitteisto- tai sisäisiä ohjelmisto-keskeytyksiä. Tämä puolestaan johtaa siihen, että mitä pidempään keskeytykset ovat estettyinä, sitä suuremmaksi kasvaa viive eli vasteaika, jonka sisällä järjestelmä pystyy vastaamaan prosessien pyyntöihin. [Dankwardt K, Jan 2002.A]

5.5.2 Ei-keskeyttävä ydin

Linuxissa suoritettava käyttäjätilan koodi on aina mahdollista keskeyttää ytimen toimesta. Tämä on käytännön edellytys, jotta moniajo toimisi hyvin yleiskäyttöisessä tietokoneessa. Ongelma muodostuu siinä, että korkeimmankin suoritustason prosessi voi tulla syrjäytetyksi myös silloin, kun sillä on kriittinen toimenpide kesken. Ydin keskeyttää kaikkien muiden prosessien toiminnan silloin, kun sen sisäisissä *tehtäväjonoissa* (task queue) ja niin sanotuissa *bottom half handlereissä* on tehtäviä odottamassa. Ne suoritetaan ytimen toimesta aina ennen käyttäjätilan ohjelmia.

Bottom half handlerillä tarkoitetaan mekanismia, jonka ansiosta kaikkea ytimen sisällä suoritettavaksi tarkoitettuja tehtäviä ei ole tarvetta suorittaa yhdellä kertaa vaan tehtäviä voidaan asettaa käyttöjärjestelmän sisällä pinoon odottamaan myöhäisempää suoritusajankohtaan. Hyvä esimerkki tästä on keskeytysten käsittely laiteajureissa: Laiteajurin ei ole tarkoitus käyttää liikaa aikaa toimenpiteiden suorittamiseen, jotta koko järjestelmän suorituskyky ei laske liikaa, sillä keskeytysten käsittelyn aikana ei voida suorittaa mitään muuta prosessia. Osa laiteajurin tehtävistä voidaan yleensä suorittaa hiljaisempana ajankohtana, jolloin suoritusvuoro voidaan luovuttaa nopeammin seuraavalle prosessille. [Rusling, 1996 - 1999]

Ratkaisun hyvänä puolena on, että sillä saavutetaan joustavuutta vaihdettaessa suoritusvuoroa tehokkaammin prosessien välillä, mutta samalla ytimen ajoituksien ennakoimisesta tulee vaikeampaa, sillä työjonoon kasautuvien tehtävien määrää ja tarkkaa suoritusajankohtaa on mahdotonta arvioida. Työjonoa on kuitenkin pakko purkaa jossain vaiheessa suorittamalla sinne kertyneitä tehtäviä. Tämä on edessä viimeistään tilanteessa, jossa käyttöjärjestelmällä on raskas kuormitus päällä ja työjonoon on ehtinyt kertyä useita suoritusvuoroa odottavia tehtäviä, jotka on pakko suorittaa.

Linuxin ytimessä on lisäksi paljon *ei-keskeytettävää* (non-preemptible) koodia. Kun käyttäjätilan prosessi tekee pääkäyttäjätason kutsun ja astuu tällaiseen osaan ydintä, kutsuvan prosessin suoritusta ei voida keskeyttää ennen kuin sen kutsu palaa takaisin lukitusta osasta. Samaan aikaan keskeytetty tai suoritusvuoroaan odottava korkeansuoritustason prosessi voi joutua odottamaan liian kauan. Ongelma on käytännössä siinä, että Linuxin ydin ei pysty keskeyttämään itseään edes tilanteessa, jossa matalan suoritustason prosessista tulee järjestelmäkutsu, kun samalla korkeamman suoritustason prosessia pitäisi ruveta palvelemaan [Dankwardt K, Jan 2002.B]. Tällaisessa tilanteessa suoritustasolla ei ole mitään painoarvoa.

Jos kriittinen prosessi on joutunut odottamaan vuoroaan liian kauan, on vasteaika kasvanut liian suureksi ja vahinko on päässyt käymään. Käytännössä tämä tapahtuu esimerkiksi silloin, kun käyttäjätilasta tehdään esimerkiksi read()-kutsu jollekin tulostus- tai syöttölaitteen laiteajurille tai jonkin oheislaitteen laiteajuri vastaanottaa keskeytyksen ja alkaa palvella tätä. Kun keskeytykset on estetty, ei silloin esimerkiksi sarjaportilta tulevaa dataa voida käsitellä ennen kuin ydin vapauttaa seuraavan kerran keskeytysten käsittelyn. Mikään ei myöskään takaa aivan varmasti sitä, että seuraavaksi palveltavaksi tuleva prosessi on juuri se minkä haluaisimme. Linux käsittelee käyttäjätilasta tulevia signaaleja eli *ohjelmallisia keskeytyksiä* (software interrupt) lähes samalla tavoin, kuin oheislaitteilta tulevia laitteistokeskeytyksiä; myös nekin joutuvat syrjäytetyksi ytimen toimesta.

5.5.3 Ennakoimattomuus

Reaaliaikaohjelmistot vaativat, että järjestelmän vasteaika on ennakoitavissa. Näin ei kuitenkaan ole Linux-käyttöjärjestelmässä. Tämä johtuu siitä, että Linuxin aikataulutin käyttää hyväkseen algoritmeja, joiden suoritusaika ei ole vakio. Linux–käyttöjärjestelmässä aikataulutin valitsee seuraavaksi suoritettavan prosessin käymällä läpi kaikki suorituslistalla

vuoroaan odottavat prosessit. Lopuksi suoritusvuoron saava prosessi aktivoidaan. Algoritmin suoritusaika on siis lineaarinen ja suoraan riippuvainen suorituslistalla olevien prosessien lukumäärästä. Algoritmin suoritusajalla ei siis ole ylärajaa, joten sen huonointa mahdollista tilannetta ei pystytä ennalta arvioimaan, jolloin ytimen vasteaika ei ole ennalta arvattavissa. Villejä arvauksia on mahdollista vetää hatusta esimerkiksi järjestelmän suorituskykyä mittaamalla ja tekemällä niistä tilastollisia päätelmiä. Ongelma on kuitenkin siinä, että hyvänkään arvion perusteella ei voida taata, että saatuun aikarajaan päästään joka ikinen kerta.

5.6 Sulautetun Linuxin jakelupaketit

Koska Linux on ilmainen ja vapaasti saatavilla lähdekoodeina, on tarjolla kymmeniä erilaisia muunnelmia jakelupaketeista. Tarjolla on muun muassa *palomuuriksi* (firewall) soveltuvia versioita, äärimmäisen *pienellä muistimäärällä* (small footprint) toimivia sekä reaaliaikasuoritukseltaan paranneltuja versioita. Lisäksi tarjolla on lukuisia versioita eri suoritinarkkitehtuureille sekä mikrokontrollereille. Ei pidä myöskään unohtaa, että on olemassa eri jakelupakettiversiot työpöytä-, palvelin- ja sulautettuihin järjestelmiin.

Sulautettua Linux-järjestelmää on mahdollista kehittää normaalissa työpöytäkäyttöön tarkoitetussa Linux-käyttöjärjestelmässä, eikä mikään estä myöskään ajamasta sulautettua Linux-laitetta tällaisella jakeluversiolla, sillä kehittäjällä on täydellinen vapaus valita itse ne tarvittavat kirjastot ja sovellusohjelmat, jotka hän tarvitsee mukaan laitteeseensa.

Valittaessa itselle sopivaa jakelupakettia kannattaa kuitenkin ottaa huomioon se seikka, että kaikkien eri jakelupakettien pohjalla on kuitenkin yksi ja sama Linux. Jakelupaketit koostuvat monilta osin samoista peruskomponenteista, kuten Linuxin ytimestä, komentotulkista ja tarvittavista kirjastoista sekä perustyökaluista. Mukana olevien osien versiot voivat vaihdella huomattavasti eri jakelupakettien välillä, jolloin toisessa jakelupakettiympäristössä linkitetty ohjelmisto ei välttämättä toimi toisessa Linux-ympäristössä ilman lähdekoodin uudelleen kääntämistä. Eroja jakelupakettien välillä syntyy lähinnä siitä, kuinka paljon erilaisia apuohjelmia, sovellusohjelmia ja työkaluja sen mukana toimitetaan.

Oikean jakelupaketin valinta onkin vaikea tehtävä, sillä tarjolla on niin monia eri vaihtoehtoja, joista valita. Koska oikean valinnan tekeminen riippuu hyvin voimakkaasti sovellusalueesta, mitään yleispätevää ohjetta valitsemiseen

ei ole tarjolla. Kannattaa tutustua mahdollisimman monen eri toimittajan Linux-jakelupakettiin, sillä oikealla valinnalla voi säästyä huomattavan paljon aikaa ja rahaa jos kohdalle osuu juuri laitteistoon ja käyttötarkoituksiin sopivilla ominaisuuksilla varustettu jakelupaketti.

Suurin osa tarjolla olevista jakelupaketeista on ilmaisia, mutta on olemassa myös kaupallisia jakelupaketteja. Merkittävin ero kaupallisten ja ei-kaupallisten jakelupakettien välillä syntyy siinä, että kaupallisten jakelupakettien tarjoajat antavat tuotteilleen tukea ja niiden mukana toimitetaan hyödyllisiä apuohjelmia, jotka helpottavat ja siten nopeuttavat kehittäjän työtä. Monesta kaupallisesta jakelupaketista on olemassa myös ilmaiseksi Internetistä ladattava versio, johon ei luonnollisesti kuulu mitään painettuja ohjekirjoja tai tukea.

5.6.1 Kaupallisia jakelupaketteja

Tällä hetkellä tarjolla olevia kaupallisia sulautetun Linuxin jakelupaketteja ovat: [LinuxDevices.com, Aug 2001]

- AMIRIX: Embedded Linux
- Coollogic: Coollinux
- Coventive: XLinux.
- Esfia: RedBlue Linux
- KYZO: PizzaBox Linux
- Lineo: Embedix
- LynuxWorks: BlueCat
- Mizi: Linu@
- MontaVista: Hard Hat Linux.
- Neoware: NeoLinux
- PalmPalm: Tynux
- Red Hat: Embedded Linux
- REDSonic: RedIce-Linux
- RidgeRun: DSPLinux.
- SysGo: ELinOS Embedded Linux
- TimeSys: Linux GPL
- Tuxia: TASTE
- Vitals System: vLinux

5.6.2 Avoimen lähdekoodin jakelupaketteja

Tällä hetkellä tarjolla olevia Avoimen Lähdekoodin sulautetun Linuxin jakelupaketteja. [LinuxDevices.com, Aug 2001]

- Embedded Debian Project
- ETLinux
- FREESCO
- Linux Router Project
- Linux-VR Project
- Linux On A Floppy (LOAF)
- Qplus
- Midori Linux
- uClinux
- μLinux
- PeeWeeLinux
- ThinLinux

5.6.3 Muita jakelupaketteja

Edellä luetellut jakelupaketit ovat koottu erityisesti sulautetun Linuxin tarpeiden mukaan. Linuxista on tarjolla myös lukuisia muita jakelupaketteja edellä mainittujen lisäksi, mutta niiden kaikkien luetteleminen ei tässä yhteydessä ole mielekästä. Muut jakelupaketit ovat normaaliin työpöytä- sekä palvelinkäyttöön tarkoitettuja jakelupaketteja, jotka voivat myös soveltua tiettyihin sulatettuihin sovelluksiin erinomaisesti.

Koska tällaisten jakelupakettien ensisijainen käyttökohde on suurella kiintolevytilalla ja keskusmuistilla varustetut yleiskäyttöiset tietokoneet, on kehittäjän tehtävä räätälöinti käsin sen suhteen, mitä ohjelmistokomponentteja järjestelmään halutaan ottaa mukaan. Mitään suoranaista teknistä estettä ei esimerkiksi työpöytäkäyttöön tarkoitetun Debian Linuxin käytölle sulautetussa järjestelmässä ole.

Myös näistä jakelupaketeista on tarjolla sekä kaupallisia, että ei-kaupallisia versioita. Kaupallisen version tukipalveluista ei tosin ole yleensä hyötyä suoranaisesti sulautetun järjestelmän kehittäjän kannalta, sillä tukipalvelut on tarkoitettu pääasiassa normaalin tietokoneen käyttäjälle. Tilanne muuttuu toiseksi jos järjestelmä koostuu yleisesti käytössä olevista laitteistokomponenteista, joille tukea on saatavilla.

6 Ratkaisuja reaaliaikaisuuden puutteeseen

Monta Vista Softwaren pääjohtaja Jim Ready on todennut, että kaikki sulautetut järjestelmät eivät vaadi käyttöjärjestelmältä kovaa reaaliaikaisuutta, vaan että perus Linuxin tarjoama suorituskyky on täysin riittävä. Tämä pitää paikkansa joissain tapauksissa, mutta on vahvasti sovelluskohteesta riippuvainen. Linux esimerkiksi sulautettuna mp3-soittimessa ei aseta juurikaan vaatimuksia reaaliaikaisuudelle. Käyttäjä tuskin huomaa sitä jos toiston käynnistävän painikkeen painamisesta kuluu esimerkiksi 100 millisekuntia siihen, ennen kuin laitteesta alkaa kuulumaan ääntä. Toisaalta kovaa reaaliaikaisuutta vaativassa järjestelmässä 100 millisekuntia voi olla kohtalokkaan pitkä aika, jos esimerkiksi lentokoneen ohjausjärjestelmän tietokoneelta kuluu sen verran liikaa aikaa lentäjän virheohjausliikkeeseen reagoimiseen ja sen korjaukseen.

Joissain tapauksissa reaaliaikaisuuden puute on mahdollista poistaa kokonaan tai järjestelmän reaaliaikasuorituskykyä voidaan parantaa merkittävästi laitteistoratkaisujen muutoksilla. Yksi tällaisista vaihtoehdoista on yksinkertaisesti kasvattaa järjestelmän suorituskykyä esimerkiksi nopeamman suorittimen avulla, jolloin yleisellä tasolla saavutetaan pienempi vasteaika. Ongelmien kiertäminen laitteistomuutoksilla on kuitenkin ratkaisu vain joihinkin yksittäistapauksiin. Se ei koskaan poista täysin riskitilanteen mahdollisuutta, jossa järjestelmä epäonnistuu tehtävässään, jos ulkoisia ja sisäisiä ärsykkeitä on hetkellisesti liikaa.

Paras tapa saada puute hoidettua on korjata ongelma sen alkulähteellä, eli Linuxin ytimessä. Viime aikoina juuri tähän ovatkin monet Linuxille tukea ja ohjelmistoja tarjoavista organisaatioista panostaneet huomattavan paljon voimavarojaan.

6.1 Reaaliaikapäivitykset ja jakelupaketit

Linuxille on tarjolla useita erilaisia ratkaisuja reaaliaikaisuuden saavuttamiseen, kuten seuraavista listoista voi nähdä. Ensinnäkin Linuxista on tarjolla useita kaupallisia ja ei-kaupallisia reaaliaikajakelupaketteja, joiden suorituskyky on pyritty optimoimaan reaaliaikatarpeita varten. Lisäksi Linuxille on tarjolla lukuisia reaaliaikapäivityksiä ja työkaluja, joilla voidaan muuttaa Linux tarjoamaan sovelluksille reaaliaikainen suoritusympäristö.

Alla on listattu LinuxDevices.comin lista tällä hetkellä tarjolla olevista Linuxin reaaliaika-jakelupaketteista sekä päivityksistä.

6.1.1 Kaupalliset reaaliaika-Linux-jakelupaketit

- FSMLabs: Open RTLinux
- Lineo: Embedix
- LynuxWorks: BlueCat
- MontaVista Software: Real-Time Solutions for Linux
- REDSonic: REDICE-Linux
- TimeSys: Linux/Real-Time

6.1.2 Avoimen lähdekoodin reaaliaika-Linux-jakelupaketit

- ADEOS
- ART
- KURT -- The KU Real-Time
- Linux/RK
- Linux-SRT
- QLinux
- RealTimeLinux.org
- RED-Linux
- RTAI
- RTLinux

6.1.3 Avoimen lähdekoodin reaaliaikapäivityksiä ja työkaluja Linuxille

- Molnar’s patches (Linux 2.2/2.4)
- Andrew Morton’s pre-emption point kernel patches (Linux 2.4)
- A Linux kernel pre-emptability enhancement
- A configurable Linux Real-time scheduler
- Real-time Linux common API
- EL/IX
- Linux Trace Toolkit
- Linux real-time characterization
- CarbonKernel RTOS simulator

6.2 Menetelmiä

Tässä kappaleessa esittelen yleisimmät menetelmät, joilla Linux voidaan muuntaa reaaliaikaisemmaksi käyttöjärjestelmäksi, kun se on perusytimellä varustettu.

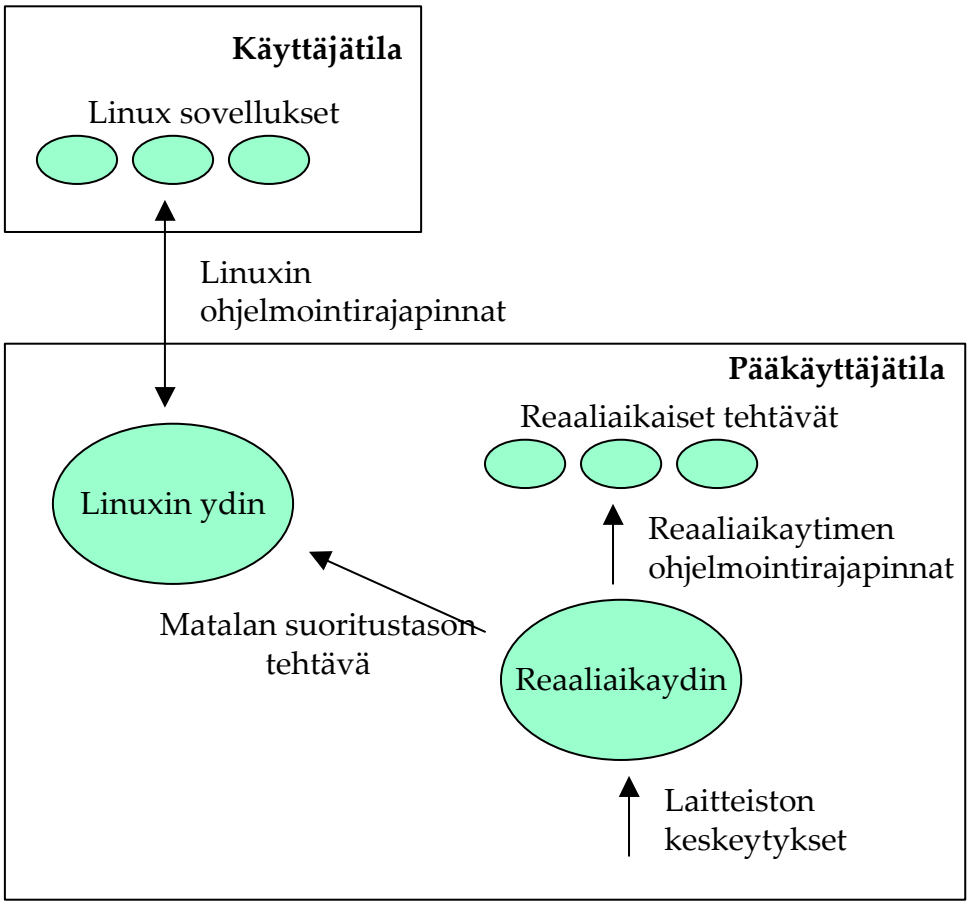
6.2.1 Kaksoisydin

Yleisin menetelmä saavuttaa reaaliaikavaatimukset Linuxissa on niin kutsuttu kaksoisydinten menetelmä, jossa Linuxin perusydintä muokataan vain vähän. Käytännössä Linuxin ytimeen lisätään pieni, vain muutaman sadan rivin pituinen kerros koodia keskeytysohjaimen ja ytimen väliin [Ivchenko, May 2001]. Lisätyn kerroksen tarkoitus on eristää Linuxin ydin fyysisestä laitteistosta. Kaksoisydinten menetelmää käyttävät muun muassa RTLinux ja RTAI. Jakelupaketissaan esimerkiksi LynuxWorks ja Red Hat käyttävät hyväkseen RTLinuxin teknologiaa ja puolestaan Lineo RTAI:n tarjoamaa teknologiaa.

Yksinkertaisuudessaan kaksoisydinten toimintaperiaate on se, että Linuxin (ei-reaaliaikainen) ydin laitetaan prosessiksi toisen, reaaliaikaisuuteen kykenevän ytimen alle. Reaaliaikaydintä ajetaan suoraan x86-suorittimessa ja täysin erillään Linuxin ytimestä. Varsinaista Linuxin ydintä ajetaan reaaliaikaytimen alaisuudessa samalla suoritustasolla muiden prosessien kanssa, alimmalla mahdollisella suoritustasolla eli käytännössä lepo-suoritustasolla. Linuxin ydin siis jakaa *tausta-ajossa* (background) suoritusajan reaaliaikaisuutta vaativien prosessien kanssa, jotka vaativat kovaa reaaliaikaisuutta. Linuxin ytimen on tyydyttävä ainoastaan pehmeään reaaliaikaisuuteen saaden suoritusajasta ainoastaan sen, joka jää yli reaaliaikaista käsittelyä vaativilta prosesseilta [Wiesner, March 2001]. Kuva 7 havainnollistaa, miten Linuxin ydintä suoritetaan matalan suoritustason tehtävänä reaaliaikaytimen alaisuudessa.

Erottamalla Linux täysin keskeytyksiä ohjaavasta laitteistosta kaksoisytimen ratkaisussa ja tarjoamalla sille virtuaalinen rajapinta laitteistoon, saadaan luotua kovan-reaaliajan järjestelmä, jossa on myös tarjolla kaikki Linuxin hyvät puolet siten, ettei mikään Linuxin alaisuudessa suoritettava asia muutu. Huonoa ratkaisussa on se, että ohjelmat, jotka halutaan tai on pakko suorittaa kovaa reaaliaikaisuutta vaativalla tasolla eli reaaliaikaytimen alaisuudessa, on sovitettava *sen omaan* (proprietary) ohjelmointirajapintaan sopivaksi. Tästä aiheutuu se, että reaaliaikaisuutta vaativat ohjelmat eivät ole enää Linux-

käyttöjärjestelmän kanssa yhteensopivia ohjelmia. Käytännössä tämä tarkoittaa siis sitä, että koodin uudelleenkäytettävyys muuttuu tältä osin paljon vaikeammaksi. Tätä voidaan pitää isonakin ongelmana, sillä kova kilpailu on johtanut tilanteeseen, jossa vanhoja koodeja tulisi pystyä käyttämään mahdollisimman paljon hyväksi uusia järjestelmiä suunniteltaessa. Lisäksi reaaliaikaiset prosessit eivät voi käyttää Linuxin järjestelmäkutsuja tai laiteajureita hyväkseen.



Kuva 7: Linuxin ja reaaliaikaytimen yhteistyö. Clark [Aug - Jun, 2000]

Yhteensopivuuden kanssa samanaikaisesti menetetään lisäksi yksi Linuxin tärkeimmistä järjestelmän vakauteen liittyvistä ominaisuuksista, nimittäin muistinsuojaus. Reaaliaikaiset tehtävät suoritetaan pääkäyttäjätilassa, muistiltaan suojaamattoman reaaliaikaytimen alaisuudessa, eikä Linuxin suojatussa käyttäjätilassa, jossa prosessien muistiavaruudet on erotettu toisistaan. Tästä aiheutuu se, että pienikin ohjelmointivirhe ohjelmistokoodissa voi aiheuttaa pahimmassa tapauksessa järjestelmän kaatumisen ja sen seurauksena vakavia ongelmia [Clark, Aug - Jun 2000]. Tällainen toiminta ei ole missään tilanteessa hyväksyttyä reaaliaikajärjestelmissä.

Reaaliaikakomponenttien kehitystyötä ja sen vaatimaa testausta ei voida myöskään suorittaa enää Linuxin alla, vaan ne on tehtävä reaaliaikaytimen alaisuudessa ja erityisesti sille tarkoitetuilla kehitystyökaluilla. Tällä on oma vaikutuksensa kehitystyön kulkuun, tehden siitä vaikeampaa.

6.2.2 Pienempi viive

Toisen hyvin suositun menetelmän, jolla Linuxin reaaliaikaongelma voidaan poistaa tai, jolla sitä voidaan ainakin merkittävästi parantaa, on niin sanottu *pienempään viiveeseen* (low-latency) pyrkivät ytimen päivitykset. Tähän ryhmään voidaan laskea kaikki menetelmät, joilla Linuxin ytimen kykyä vaihtaa suoritettava tehtävä parannetaan siten, että viive korkeamman suoritustason prosessien suoritukselle olisi mahdollisimman lyhyt.

Yksinkertaisin päivitys saada Linuxin ydin vaihtamaan tehtävää nopeammin ovat niin kutsutut *keskeytyspisteet* (preemption points), joiden tehtävänä on suorittaa tietyissä kohdissa eksplisiittisesti tarkistus siitä, onko korkeamman suoritustason tehtävä valmiina, odottamassa suoritusvuoroaan. [Dankwardt K, Jan 2002.B] Jos näin on, ydin siirtyy palvelemaan sitä välittömästi keskeytyspisteen kohdalla. Esimerkiksi Morton [Aug 2001] on suorittanut Linuxin ytimen eri osien suoritusaikojen mittauksia. Tulosten perusteella hän on löytänyt ne ongelmakohdat, joissa Linuxin ytimen suoritus kestää reaaliaikaisuuden vaatimuksiin nähden liian kauan. Mortonin tarjoamissa päivityksissä Linuxin ytimen lähdekoodeihin lisätään keskeytyspisteitä löydettyihin ongelmakohtiin.

Hyvin karkeasti yksinkertaistettuna Mortonin keskeytyspisteissä suoritetaan ohjelmaistauksen 1 mukainen toimenpide [Morton, Aug 2001] eli käytännössä ydin pakotetaan tarvittaessa keskeyttämään nykyisen tehtävän suoritus.

```
if (conditional_schedule_needed()) {
        spin_unlock(&dcache_lock);
        unconditional_schedule();
}
```

Ohjelmalistaus 1: Esimerkki Mortonin päivityksen keskeytyspisteestä.

Tämän ratkaisun hyvänä puolena voidaan pitää sitä, että se ei vaadi monimutkaisia muutoksia Linuxin lähdekoodeihin ja sillä saavutetaan joillekin

sovelluksille riittävä reaaliaikaisuuden taso. Kovaa reaaliaikaisuutta tällä ratkaisulla ei kuitenkaan saavuteta, vaan yleensä on tyydyttävä ainoastaan pehmeän reaaliajan vaatimuksiin. Menetelmän hyvänä puolena voidaan myöskin pitää sitä, että tällä ratkaisulla ei ole mitään vaikutusta sovellusohjelmien toimintaan eli käytännössä niihin ei tarvitse tehdä mitään muutoksia. Tämän ansiosta lähdekoodin uudelleenkäytettävyys ei vaarannu millään tavoin.

Ratkaisun huonoihin puoliin on laskettava ehdottomasti vähäisetkin tarvittavat muutokset Linuxin lähdekoodeihin. Päivityksen lisäyksen jälkeen ydin ei ole enää standardin mukainen Linux-ydin ja mahdollisten muiden tarvittavien tai haluttujen päivityksien sekä korjauksien lisäys reaaliaika-päivitettyyn ytimeen ei välttämättä ole enää mahdollista. Tällaisessa tilanteessa päivitykset on lisättävä käsin ja se voi merkitä ajallisesti hyvinkin pitkä projektia. Lisäksi päivitykset on tarkoitettu vain tietyille Linuxin versiolle ja se ei välttämättä toimi laisinkaan muun version kanssa, kuin mille se on alun perin suunniteltu. Päivityksiä ei myöskään ole tarjolla jokaiselle Linuxin versiolle ja tämä saattaa osaltaan rajoittaa Linux version valintaa. Päivitettäessä ydin eri versioon on samalla tehtävä reaaliaikapäivitys uudelleen.

Keskeytyspisteiden lisäyksen jälkeen huonoin keskeytysviive on pisin väli kahden eri keskeytyspisteen välillä eli viive on riippuvainen suoraan siitä miten ahkerasti keskeytyspisteitä on laitettu lähdekoodin eri kohtiin. Äärimmäisen hyvään keskeytysviiveeseen pyrittäessä keskeytyspisteitä on oltava suuri määrä. Pitää kuitenkin muistaa ottaa huomioon, että ehtolauseiden lisääminen ytimen lähdekoodeihin ei ole suoritusajan kannalta ilmaista. Mitä enemmän keskeytyspisteitä on, sitä suuremmaksi kasvaa ytimen viive, sillä jokaisessa tarkistuspisteessä joudutaan suorittamaan hieman lisäkoodia aikaisempaan verrattuna. Tästä voi olla seurauksena järjestelmän yleisen suoristustason laskeminen vaikka reaaliaikasuorituskyky onkin kasvanut korkeamman prioriteetin prosessien palvelun nopeutumisena. Clark [Aug - Jun, 2000].

Keskeytyspisteitä hyödyntää jakelupaketissaan esimerkiksi REDSonic.

6.2.3 Keskeyttävä ydin

Linux ytimen muuntaminen keskeyttäväksi mahdollistaa käyttäjätason prosessin keskeyttämisen myös silloin, kun se on suorittamassa järjestelmäkutsua [Dankward, Jan 2002.B]. Ratkaisu muistuttaa jossain määrin matalan viiveen päivityksiä, mutta tämän ratkaisun muutoksien tarkoitus on

mahdollistaa matalamman eli ei niin tärkeän prosessin suoritus välittömästi, kun korkeamman suoritustason prosessi tarvitsee suoritusvuoron. Lisäksi erona on se, että keskeyttävä ydin pyrkii estämään korkeamman suoritustason prosessin tulemisen syrjäytetyksi ytimen toimesta. Edellä lueteltujen ominaisuuksien ansiosta tällä ratkaisumallilla on mahdollista saavuttaa erinomainen reaaliaikainen suorituskyky.

Keskeyttävän ytimen ansiosta suoritusvuoro voidaan luovuttaa nopeammin korkeamman suoritustason omaavalle prosessille. Täysin mielivaltaisissa kohdissa keskeytystä ei kuitenkaan voida suorittaa turvallisesti, sillä joissain kohdissa ydintä joudutaan käsittelemään tietoa, johon saa päästä käsiksi vain yksi prosessi kerrallaan. Tällaisia alueita kutsutaan nimellä *kriittinen alue* (critical section). Nämä lohkot on suojattu Linuxin ytimessä rajaamalla lähdekoodi *spin lock* –kutsujen sisään. Menetelmä estää muita prosesseja pääsemästä etenemään lukituspisteestä eteenpäin silloin, kun joku muu prosessi on suorittamassa toimenpiteitään lukitun alueen sisäpuolella. [Dankward, Jan 2002.B]

MontaVistan ja TimeSysin tarjoamissa ratkaisuissa spin lock -kutsuja on muutettu siten, että keskeytyksien määrä on saatu vähenemään entisestään. Kun korkeamman suoritustason prosessi haluaa suoritusaikaa, aikataulutin keskeyttää alhaisemman suoritustason prosessin järjestelmäkutsun, jos muutetun spin lock –kutsun toimesta ei ole erikseen kerrottu, että keskeytys ei ole mahdollinen.

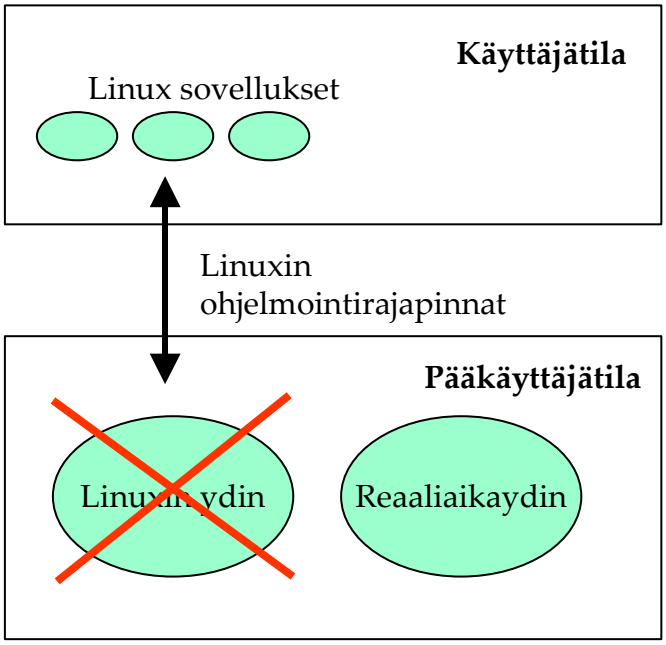
6.2.4 Ytimen korvaaminen

Neljäntenä tapana ratkaista Linuxin reaaliaikaongelma esittelen Lynx Real-Time Systems Inc.:n kehittämän kaupallisen LynxOS-ytimen. Tämä ratkaisumalli ei täysin istu tämän tutkimuksen aihealueeseen, mutta koska se on tällä hetkellä yksi tehokkaimmista tavoista tarjota Linux-sovellusohjelmille reaaliaikaista alustaa, niin käyn sen toimintaperiaatteen lyhyesti läpi.

Ratkaisun idea on siinä, että Linuxin ydin korvataan kokonaan toisella ytimellä eli LynxOS:llä, joka pystyy saavuttamaan kovanreaaliajan järjestelmille asetetut vaatimukset. Tässä ratkaisussa ei siis ole Linuxin ydintä laisinkaan, vaan se on korvattu kokonaan uudella ytimellä, jonka suunnittelussa on otettu alun perin huomioon reaaliaikaisuuden vaatimukset. LynxOS ei siis ole muunnelma Linuxin ytimestä. Mielenkiintoiseksi tämän ratkaisun tekee se, että

se alaisuudessa voidaan suorittaa kaikkia samoja sovellusohjelmia ja työkaluja kuin Linuxissa, eli LynxOS on rajapinnoiltaan täysin yhteensopiva Linuxin ytimen kanssa [LynyxWorks.B]. Kuva 8. havainnollistaa Linux ytimen korvaamista toisella, säilyttäen kuitenkin ohjelmointirajapintojen yhteensopivuus sovellusohjelmille samana.

LynxOS toteuttaa siis samassa paketissa täysiverisen reaaliaikaytimen ja standardin mukaiset rajapinnat, kuten Unix, Linux sekä POSIX. Näiden ominaisuuksien ansiosta LynxOS suoritusympäristönä ei vaadi erityisten reaaliaika-rajapintojen käyttöä, vaan Linux sovellusohjelmistot hyötyvät LynxOS-ytimen tuomista eduista automaattisesti.



Kuva 8: Linuxin ytimen korvaaminen. Clark [Aug - Jun, 2000]

LynuxWorksin ratkaisun ehdottoman hyvinä puolina on pidettävä sitä, että ohjelmistoihin ei tarvitse tehdä mitään muutoksia ja samalla pystytään tarjoamaan täydellisen hyvää reaaliaikaisuutta. Tämän ansiosta kehittäjät voivat hyödyntää täysipainoisesti tähän mennessä opittua Linux-sovellusohjelmointi kokemustaan ilman, että tarvitsee opetella uutta. Toisaalta tämä ratkaisu syrjäyttää täysin Linux-ytimen kaupallisella ja suljetulla ytimellä, jolloin kehittäjillä ei ole täydellistä hallintaa käyttöjärjestelmän sisällä tapahtuvien asioiden suhteen, joka kuitenkin on yksi Linuxin vahvimmista puolista. LynuxOS ei ole myöskään ilmainen. Vastapainoksi kaupallisuudelle LynuxWorks tarjoaa laajat tuki- ja koulutuspalvelut asiakkailleen.

Myös OnCore:lla on tarjolla vastaavanlainen ratkaisu, kuin LynuxWorksin LynxOS, heidän kaksoisydintä hyödyntävän Linux ratkaisunsa lisäksi.

6.3 Yhteenveto

Reaaliaikapäivitykset on tarkoitettu käytettäväksi pääsääntöisesti silloin, kun käyttöön valittu Linuxin ydin on niin sanottu perusversio, johon ei olla tehty vielä mitään muutoksia. Monet päivityksistä on myös yleensä sidottuja tarkasti tiettyyn Linux-versioon tai versiohaaraan, kuten esimerkiksi Linux 2.2 -sarjan ytimiin.

Reaaliaika Linux-jakelupaketit ovat valmiiksi koottuja ja optimoituja ajatellen reaaliaikajärjestelmien tarpeita. Tarjolla on lukuisia vaihtoehtoja, joista valita. Valmiiden reaaliaikajakelupakettien ratkaisut perustuvat samoihin perusmenetelmiin, jotka esittelin edellä, joten samaan suorituskykyyn on mahdollista päästä myös ilmaisen jakelupaketin avulla käyttäen hyväksi tarjolla olevia ilmaisia päivityksiä sekä työkaluja suorituskyvyn mittaamiseen.

Clarkin [Aug - Jun 2000] mukaan ainoa todellinen ratkaisu onkin suunnitella ydin, joka on täysin keskeyttävä; Kriittiset alueet, jotka käsittelevät jaettuja tietorakenteita käyttävät paremmin tilanteeseen soveltuvia synkronisointimetodeja, kuten esimerkiksi *opastimia* (semaphore) tai keskeytykset voidaan estää täksi ajaksi. Tietorakenteet on suunniteltava siten, että niiden käsittely on ennakoitavissa ja nopeaa. Tämä tarkoittaa esimerkiksi sitä, että aikatauluttimelta tulisi kulua aina vakioaika seuraavan suoritettavan prosessin valintaan ja suoritusvuoron vaihtamiseen sille, riippumatta siitä kuinka monta prosessia järjestelmässä on käynnissä.

Kaiken kaikkiaan jokaisella tarjolla olevalla menetelmällä saavutetut hyödyt ovat niin selvät, että Dankwardin [Jan 2002.B] mielestä on odotettavissa, että keskeyttävästä ytimestä on tulossa pikkuhiljaa Linuxin ytimen vakio ominaisuus. Tähän ajatukseen on helppo yhtyä, sillä esimerkiksi matalan viiveen –päivityksen lisäämisen jälkeen jo perus-Linuxissa on työpöytäkäytössä huomattavissa selvää joustavuuden ja suorituskyvyn parantumista, johtuen käyttöjärjestelmän kyvystä vaihtaa tehtäviä nopeammin. Keskeyttävän ytimen ansiosta voidaan saavuttaa viive, joka on vain parin millisekunnin luokkaa ja parhaassa tapauksessa vain kymmeniä mikrosekunteja.

Aivan ongelmattomia ratkaisut, joilla reaaliaikaisuuteen pyritään, eivät kuitenkaan ole. Jos ratkaisuksi valitaan kaksoisytimen menetelmä, niin

reaaliaikaprosesseja ei suoriteta Linux-ytimen alaisuudessa, jolloin menetetään Linuxin hyvä muistinsuojaus sekä laitteistorajapinnat. Lisäksi näiden ohjelmien lähdekoodit eivät ole Linux-yhteensopivia. Matalan viiveen päivityksissä samoin, kuin keskeyttävän ytimen menetelmissä joudutaan tekemään paljon muutoksia ytimen lähdekoodeihin, jolloin se ei enää ole standardin mukainen Linux-ydin. Lisäksi ongelmia saattaa muodostua laiteajureiden kohdalla mikäli niitä ei ole suunniteltu yhteensopiviksi moniprosessoroinnin kanssa. Keskeyttävässä ytimessä keskeytysten takia saattaa tapahtua myös sellaisia tilanteita, jossa laiteajurissa yritetään mennä kriittisten alueiden sisään samanaikaisesti, usean eri prosessin toimesta, vaikka järjestelmässä olisikin vain yksi suoritin. Siksi tilanteeseen on varauduttava etukäteen, varmistamalla kaikkien järjestelmän osien yhteensopivuus tällaisissa tilanteissa.

7 Sulautetun Linuxin sovelluksia

Tässä luvussa teen katsauksen jo olemassa oleviin Linux-pohjaisiin laitetyyppeihin, jotka voidaan laskea sulautettujen järjestelmien ryhmään erityisen käyttötarkoituksen, käyttöliittymän ja käytössä olevien resurssien suhteen.

7.1 PDA–laitteet sekä Web-padit

Kuva 9: Galleo Linux Multimedia Communicator. [Benenstein, July 2001]

Tarjolla olevia Linux-pohjaisia PDA-laitteita ja muunnelmia ovat muun muassa:

- Galleo Linux Multimedia Communicator – Isolla värinäytöllä varustettu Comminicator tyyppinen monitoimilaite (Kuv 9). Galleo sisältää muun muassa:
    - Kaksitaajuus puhelimen
    - Konqueror WWW-selaimen
    - Video- sekä mp3-soittimen
    - Java-tuen
    - USB sekä IrDA yhteydet
    - Videokameran
- Compaq iPAQ – Maailmalla toiseksi eniten myytyyn PDA-laitteeseen, Palm-laitteiden jälkeen, sovitettu Linux-käyttöjärjestelmä täydellisine apuohjelmineen.

- IBM e-LAP reference design – IBM:n valmistama *esimerkkilaite* (reference design) tekee muiden laitevalmistajien mahdolliseksi hyödyntää IBM:n aikaansaamia tuloksia Linux-pohjaisen PDA-laitteen kehittelyssä tarjolla olevan toimivan laitteen pohjalta.

Edellä lueteltujen laitteiden lisäksi tarjolla on myös lukuisten muiden tunnettujen valmistajien, kuten LG:n, Sharpin ja Hitachin kehittämiä Linux-pohjaisia PDA- sekä Web-pad laitteita.

7.2 Äly- ja Internetpuhelimet

Kuva 10: Motorola A760 Linux älypuhelin.

Linuxia on tähän mennessä sovellettu äly- ja Internetpuhelinten ryhmässä muun muassa seuraavan tyyppisillä laitteilla: [LinuxDevices]

- VoiceIP pöytäpuhelimissa
- IP Video -puhelimissa
- Internet puhelimissa
- Matkapuhelimissa

Uusimpana tulokkaana tähän laiteryhmään on tullut Motorolan juuri julkaisema älypuhelin (Kuva 10), joka hyödyntää käyttöjärjestelmänä Linuxia. [Motorola] Ensimmäisenä Linux-pohjaisena älypuhelimena Motorola A760:lla on paljon pelissä. Jos tuote menestyy, niin Linuxille on odotettavissa erittäin hyviä aikoja älypuhelimissa.

7.3 Audio ja video viihdelaitteet

Kuva 11: Nokia Media Terminal [Nokia]

Linuxia on tähän mennessä sovellettu audio ja video viihdelaitteet ryhmässä muun muassa seuraavan tyyppisillä laitteilla: [LinuxDevices]

- Digitaalinen videonauhuri
- Digitaalinen vastaanotin
- Mp3-soitin
- Autosoitin
- Internetradio
- Internet/TV

Muita valmistajia Nokian (Kuva 11) lisäksi, jotka käyttävät Linuxia verkkotuotteissaan ovat muun muassa Sony, NEC, Motorola sekä HP. Myös tällä osa-alueella uskon Linuxin valtaavan tulevaisuudessa yhä enemmän jalansijaa muilta käyttöjärjestelmiltä.

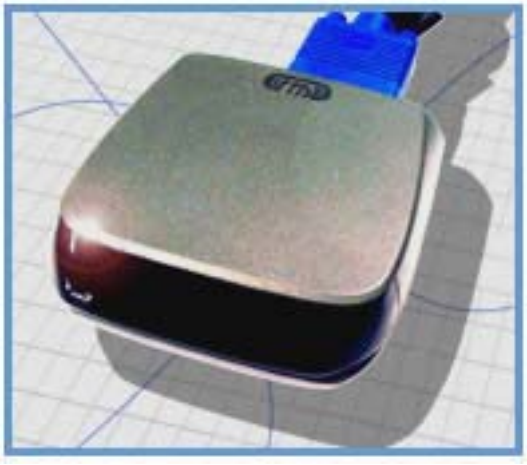
7.4 Verkkolaitteet

Kuva 12: Wireless Networks Inc.:n BlueLAN yhdyskäytävä [Wireless Networks Inc.]

Perinteisen Linux-pohjaisten laitteiden ryhmän muodostavat nimenomaisesti verkkolaitteet, kuten reitittimet, yhdyskäytävät, palomuurit, tulostinpalvelimet ja niin edelleen. Tässä ryhmässä Linux on ollut aina vahvoilla sekä yleiskäyttöisten, että sulautettujen järjestelmien puolella. Yhtenä syynä tähän on ollut se, että Linux-käyttöjärjestelmä on alun perin suunniteltu verkkokäyttöjärjestelmäksi ja siinä on erinomainen tuki TCP/IP:lle.

Valmistajia, jotka käyttävät Linuxia verkkotuotteissaan ovat muun muassa Hitachi, Ericsson sekä LinkSys.

7.5 Muita laitteita



Kuva 13: Irma [Flander]

Linuxia on sovellettu edellä esiteltyjen päälaiteryhmien lisäksi myös muun tyyppisiin laitteisiin, joita ovat muun muassa:

- Rannekello (IBM, Citizen)
- Robotti (ActiveMedia, Tokion yliopisto)
- Web-kamera (Axis, StarDot)
- Analogisen videokuvan verkkopalvelin (Sony)
- Auto (Dodge, DaimlerChrysler)

Yksikään näistä laitteista ei edusta ominaisuuksiltaan laiteryhmälle tyypillistä laitetta, sillä Linux-pohjaisuudella niihin on pyritty tuomaan mukaan enemmän älyä.

Viimeisenä laitteena haluan esitellä Linux-pohjaisen Irma – The Infrared Mobile Accessory laitteen, joka oli alun perin tarkoitettu pelkästään Nokia Communicator 9200-sarjan puhelimen lisävarusteeksi. Nykyään Irmasta on tarjolla myös muille laitteille tarkoitettuja muunnelmia. Laitteen avulla käyttäjä voi tallentaa PC:ltä PowerPoint esityksen diaesitykseksi esimerkiksi Communicatorin muistiin, josta ne voidaan helposti näyttää taskukokoisen Irman avulla suoraan Communicatorista videotykillä. Olin mukana Irma-projektissa, jossa suunnittelin ja toteutin muun muassa Irman sisällä olevan Linux ohjelmiston sekä hoidin Linuxin räätälöinnin Irman laitealustalle.

Lyhyt yhteenveto Irman ominaisuuksista:

- Paino 110 g
- Koko 99 x 90 x 25 mm
- Resoluutio 800 x 600 pikseliä
- IrDA 115 kbit/s
- 8 MB FLASH ROM, josta 4 MB käyttöjärjestelmän ja sovellusohjelmiston käytössä.
- RAM 32/64 MB
- Suoritin STPC ~80 MHz
- Linux 2.4, räätälöity vakioydin.
- Kirjastot RedHat 7.0:sta, paitsi glib-c BlueCat Linuxista.
- Sovellusohjelmisto ainoa käyttäjäprosessi

7.6 Yhteenveto

Kun tarkastellaan yleisesti tarjolla olevia Linux-pohjaisia sulautettuja laitteita ja niiden määrää, voidaan tehdä hyvin suoraviivainen johtopäätös sen asian suhteen, miten hyvin Linux soveltuu erilaisten laitteiden käyttöjärjestelmäksi. Mielestäni Linux on ominaisuuksiltaan erinomainen valinta sulautettujen laitteiden käyttöjärjestelmäksi. Tätä puoltaa esimerkiksi se, että jos vertaa Linux-pohjaisia laitteita muihin saman laiteryhmän eri käyttöjärjestelmällä varustettuihin laitteisiin, voi havaita välittömästi, ettei Linuxin valitsemisella laitteen käyttöjärjestelmäksi ainakaan ole merkitystä laitteen tarjoamiin ominaisuuksiin. Toisin sanoen Linux sinällään ei rajoita laitteen toiminnallisuutta.

8 Keskustelua

Sulautetut järjestelmät ovat nykyään hyvin ajankohtaisia, sillä laitteista on tulossa yhä älykkäämpiä ja monimutkaisempia. Linux-käyttöjärjestelmä on joustava ratkaisu sulautettuihin järjestelmiin ja se tarjoaa varteenotettavan vaihtoehdon muille sulautetuille käyttöjärjestelmille. Suurimmaksi puutteeksi nousee reaaliaikaisuuden puute, joka on korjattavissa esimerkiksi RTLinuxin kaltaisella ilmaisella päivityksellä ytimeen. RTLinux ratkaisee kovanreaaliajan ongelman, mutta tuo mukanaan uusia; Reaaliaikaisuutta vaativat sovellusohjelmat menettävät muun muassa Linux-yhteensopivuuden ja sen tarjoaman vakaamman alustan.

Linuxin joustavuutta puoltavat muun muassa ytimen rakenne sekä sen julkinen lähdekoodi. Lähdekoodin avoimuudella on erityisen suuri merkitys, koska tällöin järjestelmän suunnittelijalla on täydellinen kontrolli käyttöjärjestelmän toimintaan. Lähes kaikkiin Linuxin sovellusohjelmiin on myös saatavilla lähdekoodit, yleensä GPL-lisenssin alaisina. Se tarjoaa valtavan tietopankin, jonka turvin kehitystyötä on mahdollista nopeuttaa hyödyntämällä muiden valmiita lähdekoodeja. Open Source ja GPL-lisenssi saattavat olla kuitenkin joillekin este toteuttaa järjestelmäänsä Linuxilla, koska kaikkia kehitettyjä hienouksia ei haluta paljastaa kilpailijoille. Vastaavasti kilpailijan Linux-ratkaisut ovat vapaasti hyödynnettävissä. Markkinamielessä avoimet lisenssit ei ole kaikkien mieleen, sillä sen alaisilla ohjelmilla ei voi suoranaisesti rahastaa ja niitä saa kuka tahansa muokata tai levittää eteenpäin. Linuxia hyödyntävien yritysten liiketoiminta perustuukin yleensä asiakkaille tuotettuihin tukipalveluihin. Toisaalta sulautetun järjestelmän sovellusohjelmisto voi olla lähdekoodeiltaan suljettu vaikka käyttöjärjestelmänä olisikin käytössä avoimen lähdekoodin Linux.

Linux tuo oman suolansa yhä kiihtyville sulautettujen käyttöjärjestelmien markkinoille. Ennen lisenssimaksuihin huvenneet rahat, jotka ovat samalla nostaneet tuotteen hintaa loppukäyttäjälle, voidaan Linuxia sovellettaessa sijoittaa esimerkiksi kehitystyöhön. Näin tuotteista saadaan varmasti parempia tai tuotetta voidaan myydä asiakkaalle edullisempaan hintaan. Lisenssirahojen säästyminen mahdollistaa myös sen, että tuotetta voidaan myydä vaihtoehtoisesti paremmalla katteella, joka on liiketoiminnan ja Linuxille tarjottavan tuen sekä kehityksen kannalta erittäin suotavaa.

Yhtä laajaa laitearkkitehtuurin tukea ei ole myöskään tarjota muilla käyttöjärjestelmillä. Laitearkkitehtuurituki on tärkeä seikka arkkitehtuurin

valinnan vapauden kannalta. Lisäksi se helpottaa kehitystyötä ja testausta, sillä perinteisellä x86-työasemalla on mahdollista kehittää sulautettua ohjelmistoa, joka toimii lopullisessa tuotteessa esimerkiksi ARM-arkkitehtuurilla.

Yleisesti ottaen Linux sulautetun järjestelmän käyttöjärjestelmänä tarjoaa vankan rungon, jonka päälle oma tuote voidaan rakentaa helpommin ja nopeammin siten, että laitteistoresurssit saadaan hyödynnettyä mahdollisimman tehokkaasti. Linuxin valtava avoimen lähdekoodin ohjelmistokanta nopeuttaa osaltaan kehitystyötä.

Yhteenvetona Linuxin eduista sekä selvistä ongelmakohdista sulautetun järjestelmän käyttöjärjestelmänä: [Lehrbaum, Jan 2001]

Hyvää:

- Avoin lähdekoodi
  - Täydellinen räätälöitävyys omiin tarpeisiin
  - Mahdollisuus muutosten tekemiseen käyttöjärjestelmätasolla
  - Mahdollisuus nähdä miten asiat toimivat käyttöjärjestelmässä
  - Mahdollisuus käyttöjärjestelmän virheiden korjaamiseen välittömästi
  - Laaja ohjelmakanta tarjolla valmiina
  - Nopeuttaa kehitystyötä
- Hinta
  - Käyttöjärjestelmä ei maksa mitään
  - Ei lisenssimaksuja tai ajonaikaisia rojalteja
  - Alhaisemmat tuotantokustannukset
  - Organisaatiot keskittyvät tuen antamiseen
- Hyvin tuettu
  - Kolmansien osapuolien sovellukset
  - Erinomainen verkkotuki
  - Paljon ilmaisia ajureita sekä työkaluja saatavilla
- Tukee montaa eri arkkitehtuuria
  - Ei rajoita laitteistoa tiettyyn arkkitehtuuriin vaan voidaan valita haluttu lähes poikkeuksetta
- Toimii tarvittaessa pienillä muisti ja suoritin resursseilla
  - Alhaisemmat laitteistokustannukset
- Modulaarisuus
  - Ytimen kokoonpano täysin muunneltavissa laitteistotarpeiden mukaan
- Vakaa

- Hyvä muistinsuojaus
- Pitkät käyttöajat ilman uudelleenkäynnistämisiä

Huonoa:

- Aivan uusimmille laitteille ei välttämättä heti ajureita
  - Jotkut laitevalmistajat eivät toimita Linux-ajureita laisinkaan
  - Voi joutua tekemään ajurit itse
- Reaaliaikaisuuden puute
  - Ominaisuudet eivät ole vielä aivan yhtä kehittyneitä, kuin kaupallisissa high-end reaaliaikakäyttöjärjestelmissä
  - Tarjolla olevilla laajennuksilla ongelma voidaan kiertää, mutta tilalle tulee muita ongelmia, kuten ei-standardi Linux-API tai päivityksen muokkaama ei-standardi ydin, joka saattaa olla versio sidonnainen.
- Resurssien tarve
  - Ei tule koskaan sopimaan resurssivaatimuksilta äärimmäisen pieniin sulautettuihin järjestelmiin, joissa on tarvetta minimoida muistin ja tilan (RAM ja ROM) käyttö äärimmilleen, sillä ydintä ei saa käytännössä mahtumaan alle 400 kilotavun. Ajonaikainen keskusmuistin tarve on minimissään noin 4 megatavua.

8.1 Tulosten rajaus

Tutkimukseni rajoittui pelkästään Linuxiin ja sen sulautettujen järjestelmien kannalta keskeisten ominaisuuksien käsittelyyn. Tarkastelin asiaa pääsääntöisesti Linuxin ytimen kannalta, enkä koko käyttöjärjestelmän näkökulmasta, johon kuuluu myös käyttöliittymät ja apuohjelmat. Tutkimuksessa Linuxia ei ole varsinaisesti vertailtu muihin sulautettuihin käyttöjärjestelmiin, vaan johtopäätökset ja tulokset perustuvat täysin Linuxin tämän hetken käytännön ominaisuuksiin ja tiedossa oleviin nykyhetken mukaisiin vaatimuksiin sulautetuista järjestelmistä.

8.2 Jatkotutkimusaiheita

Linux-käyttöjärjestelmä sulautetun järjestelmän kannalta on todella laaja tutkimuskenttä. On vaikea käsitellä yksittäistä aihetta aihepiiristä, joka pitää sisällään niin paljon teknisiä asioita. Kun jokin käsite on selvitetty, huomaa, että se pitää sisällään tai vaatii tuekseen jälleen uuteen asiaa perehtymistä.

Sulautetun järjestelmän käyttöjärjestelmä tutkimusaiheena on äärimmäisen vaativa myös siksi, että se on muodostunut markkinoiden paineesta enemmän käytännön läheinen, kuin teoreettinen aihealue.

Tutkimuksen aikana rajautui muutamia selvästi muista asioista erillisiä aiheita, joita kannattaisi tarkastella tarkemmin tulevaisuudessa. Jatkotutkimusaiheiksi soveltuvia asioita ovat muun muassa:

- Millaisia sulautettuja käyttöjärjestelmiä on yleisesti ottaen tarjolla.
- Sulautettujen käyttöjärjestelmien ominaisuuksien tarkempi tutkimus.
- Miten eri käyttöjärjestelmät soveltuvat erilaisiin käyttötarkoituksiin.
- Sulautetun järjestelmän toteutus muutaman eri käyttöjärjestelmän avulla ja eri toteutuksien vertailu.
- Reaaliaikaisuuden todellinen tarve ja miten eri käyttöjärjestelmät todella pystyvät siihen.
- Linuxin reaaliaikapäivitykset.
- Linuxin reaaliaikajakelupaketit.
- Linuxin perusjakelupakettien soveltuvuus sulautettujen järjestelmien tarpeisiin.
- Perus-Linuxin räätälöinti sulautettuun järjestelmään.
- Graafista käyttöliittymää vaativat sulautetut Linuxin sovellusalueet (PDA –laitteet, matkapuhelimet)

9 Lähteet

[Benenstein, July 2001], ELJ Issue 4: Galleo Linux Multimedia Communicator, http://www.linuxjournal.com/article.php?sid=4761, Tarkistettu 6.4.2003

[Clark, Aug - Jun 2000], Running Linux applications in an embedded, real-time environment, *Dedicated Systems Magazine*, Sivut: 13 - 17, Dedicated Systems Experts

[Cylíax, Nov 1998], Embedded RT-Linux Part 1: General Introduction, *Circuit Cellar Inc*, No 100, Sivut: 49 - 50, 52, 54 - 55, Circuit Cellar Inc

[Dankwardt K, Jan 2002.A], ELJonline: Real Time and Linux, Part 1, http://www.linuxdevices.com/articles/AT5997007602.html, Tarkistettu 30.3.2003

[Dankwardt K, Jan 2002.B], ELJonline: Real Time and Linux, Part 2: the Preemptive Kernel, http://www.linuxdevices.com/articles/AT5503476267.html, Tarkistettu 30.3.2003.

[Dankwardt K, Jan 2002.C], ELJonline: Real Time and Linux, Part 3: Sub-Kernels and benchmarks, http://www.linuxdevices.com/articles/AT6320079446.html, Tarkistettu 30.3.2003.

[Epplin, Oct 1997], Linux as an Embedded Operating System, *Embedded Systems Programming*, Vol. 10, No 10, Sivut: 96 - 98, 100, 102, 104, Miller Freeman

[Evans Data Corporation, Aug 17 2001], Embedded Linux tops developers' 2002 wishlist, http://www.linuxdevices.com/news/NS5134111490.html, Tarkistettu 30.03.2003

[Flander], http://www.flander.net, Irma™ - the Infrared Mobile Accessory, Tarkistettu 3.4.2003

[FSMLab], *RTLinux*, http://fsmlabs.com/, Tarkistettu 30.03.2003.

[Free Software Foundation Inc.], GNU General Public License version 2, June 1991, http://www.gnu.org/copyleft/gpl.html, Tarkistettu 30.3.2003.

[Gemmel & Highton, 2000], Embedded ColdFire taking Linux on-board, *Embedded System Engineering*, vol 8, no 4, Sivut: 48, 50, 52, Electronic Design Automation, June - July 2000

[George Mason University] The Core of Information Technology, *Virtual Memory*, http://cne.gmu.edu/itcore/virtualmemory/vmideas.html, Tarkistettu 5.4.2003

[Hasan], *History of Linux*, http://bdlug.hypermart.net/articles/linuxhistory.html, Tarkistettu 3.4.2003

[Husain & Parker, Jan 1996], *Linux Unleashed 2nd Bk&Cs edition*, Sam's Publishing, USA

[IEEE], http://www.ieee.org, Tarkistettu 30.03.2003.

[iPAQLinux.com], http://www.ipaqlinux.com, Tarkistettu 30.3.2003.

[Ivchenko, May 2001], Real-Time Linux, *Embedded Systems Programming*, Vol. 14, No 5, Sivut: 35 - 57, CMP Media Inc

[Koski ja Tervo, 1997], *Linux tehokäyttäjän opas*, Suomen ATK -kustannus, Jyväskylä

[Kurt et al., 2001], *Linux Programming 2nd Edition*, Sam's Publishing, Indianapolis, USA

[Lennon, May 2001], Embedding Linux, *IEEE Review*, Vol. 47, Issue 3, Sivut: 33 - 37

[Lehrbaum, Jan 2001], What's so good about open source and Linux - in embedded? , http://www.linuxdevices.com/articles/AT8151978006.html, Tarkistettu 31.3.2003

[Linux Kernel Archives], http://www.kernel.org, Tarkistettu 27.04.2002.

[LinuxDevices.com, Aug 2001], The Embedded Linux Distributors Quick Reference Guide, http://www.linuxdevices.com/articles/AT2760742655.html, Tarkistettu 30.3.2003

[Lombardo, 2001], Embedded Linux, New Riders Publishing, Indianapolis

[LynuxWorks.A, 1987 - 2000], *LynuxWorks BlueCat Linux Release 3.0 User's Guide*, USA

[LynuxWorks.B], *LynuxWorks LynxOS*, http://www.lynuxworks.com/, Tarkistettu 6.4.2003

[Muropaketti, 2002], Intel Northwood 3,06 GHz, *Muropaketti*, http://www.muropaketti.com/artikkelit/cpu/nw3060/index.phtml, Tarkistettu 30.03.2003.

[Morton, Aug 2001], http://www.zip.com.au/~akpm/linux/, Linux patches, Tarkistettu 30.3.2003.

[Motorola], *Motorola Introduces World's First Java™ + Linux® Handset*, Lehdistötiedoite, http://www.motorola.com/mediacenter/news/detail/0,1958,2349_1920_23,00.html, Tarkistettu 6.4.2003

[Mullender S., 1993], Distributed Systems Second Edition, ACM Press New York, New York

[OpenSource.org], Open Source Definition Version 1.9, *http://www.opensource.org/docs/definition.html*, Tarkistettu 30.03.2003

[Ortiz Jr., Nov 2001], Embedded OS's gain the inside track, *Computer*, Vol. 34, No 11, Sivut: 14 - 16, IEEE Computer Soc.
[Peltomäki ja Linjama, 1999], *Linux-käyttäjän peruskirja*, Teknolit, Jyväskylä

[Pattersson & Hennessy, 1998], *Computer Organization & Design – The Hardware/Software interface*, Morgan Kaufmann Publishers, San Francisco

[Rubini & Corbet, June 2001], *Linux Device Drivers 2nd edition*, O'Reilly Online Catalog, http://www.xml.com/ldd/chapter/book/, Tarkistettu 30.03.2003.

[Rusling, 1996 - 1999], The Linux Kernel, Version 0.8-3, http://en.tldp.org/LDP/tlk/tlk.html , Tarkistettu 30.03.2003.

[Tietokone, 2002] Tietokone 4B/2002, Yritys-IT, Kämmenmikrot yrityskäytössä, Sivut 18 – 22, Sanoma Magazines Finland Oy

[Torvalds, Apr 1999], The Linux Edge, *Communications of the Acm*, Vol. 42, No 4, Sivut: 39 - 39, ACM

[Torvalds, 2001], *Just for Fun: The Story of an Accidental Revolutionary*, Harperbusiness, USA

[Varhol, 26 June 2000], Embedded Linux is starting to make sense, *Electronic Design*, Vol. 48, No 13, Sivut: 107 - 108, 110, 112, 114, 116, Penton Publishing

[Weinberg, Aug - Sept 2001], The Embedded Linux OS, *Embedded Systems Engineering*, Vol. 9, No 6, Sivut: 38 - 41, Electronic Design Automation

[Wiesner, March 2001], Harness robust debugging techniques to improve embedded Linux systems, *Electronic Design*, Vol. 49, No 6, Sivut: 104 - 112, Penton Mediaz