Teknillinen korkeakoulu
Elektroniikan, tietoliikenteen ja automaation tiedekunta
Tietoliikenne- ja tietoverkkotekniikan laitos

Risto Järvinen

Viestintärajapinta erityisvälitysverkoille

Diplomityö joka on jätetty opinnäytetyönä tarkastettavaksi diplomi-insinöörin tutkintoa varten.

Espoo, 23. marraskuuta 2009

Valvoja: Professori Jukka Manner

Ohjaaja: DI Juho Määttä

TEKNILLINEN KORKEAKOULU DIPLOMITYÖN TIIVISTELMÄ

Tekijä:	Risto Järvinen
Työn nimi:	Viestintärajapinta erityisvälitysverkoille
Päivämäärä:	23. marraskuuta 2009 **Sivuja:** 70
Tiedekunta:	Elektroniikan, Tietoliikenteen ja Automaation tiedekunta
Professuuri:	S-38 Tietoliikenne- ja tietoverkkotekniikan laitos (Comnet)
Työn valvoja:	Prof. Jukka Manner
Työn ohjaaja:	DI Juho Määttä

Tässä työssä käsitellään sähköisen viestinnän sovittamista erityisvälitysverkkoihin. Tietoverkkojen kehitys on edennyt asteelle, jossa verkkoyhteyksiä on saatavilla lähes kaikkialla ja lähes kaikissa laitteissa. Verkkoyhteyksien ominaisuudet kuitenkin vaihtelevat suuresti, ja monet eivät pysty sellaisenaan toimimaan Internetin osina. Erityisvälitysverkot ovat ympäristöjä, jossa Internetin protokollat eivät enää pysty toimimaan. Työn päämääränä oli kehittää rajapinta ja sen toteuttava ohjelmistokomponentti, jolla viestintäsovelluksia voidaan sovittaa viestintään haasteellisten verkkojen yli. Sähköposti valittiin rajapinnaksi sen soveltuvuuden ja standardiaseman vuoksi.

Työssä kehitettiin sulautettava, luotettava ja siirrettävä ohjelmistokomponentti, joten ohjelmiston toteutuksen suunnitteluun kiinnitettiin paljon huomiota. Ohjelmiston rakenteesta tehtiin tehokas ja suoraviivainen. Ohjelmiston kehityksen yhteydessä sen toteutusta testattiin ohjelmiston laadun ja yhteensopivuuden varmistamiseksi.

Avainsanat: sähköposti, haasteelliset verkot, erityisvälitysverkot, bundle, delay tolerant networking, SMTP, IMAP, DTN, välitys

HELSINKI UNIVERSITY OF TECHNOLOGY

ABSTRACT OF THE MASTER'S THESIS

Author:	Risto Järvinen		
Name of the thesis:	Communications Interface for Challenged Internets		
Date:	November 23, 2009	**Number of pages:**	70
Faculty:	Faculty of Electronics, Communications and Automation		
Professorship:	S-38 Department of Communications and Networking (Comnet)		
Supervisor:	Prof. Jukka Manner		
Instructors:	Juho Määttä, M.Sc.		

In this thesis we study merging electronic communications with challenged network systems. Evolution of data networks has reached a stage where network connectivity is ubiquitous almost everywhere and on almost every device. Though the abilities of different network connections vary dramatically. Many are not able to function as parts of the Internet. Challenged networks are some of the networks where Internet protocols can not function. The purpose of this work is to develop an interface, and an implementation of that interface, that normal messaging applications can use to communicate over challenged networks. Email was chosen as the interface because of it's flexibility and status as an Internet standard.

As part of this work, an embeddable, reliable and portable software component was implemented, with great focus on the design of the component. The software was designed to be efficient and straightforward. During the development, the software was tested for quality and compatibility purposes.

Keywords: email, challenged networks, bundle, delay tolerant networking, SMTP, IMAP, DTN

Alkusanat

Tämä diplomityö on tehty Teknillisen korkeakoulun Tietoliikenne- ja tietoverkkotekniikan laitokselle. Erityiskiitokset työn valvojalle, professori Jukka Mannerille ja työn ohjaajalle Juho Määtälle, sekä Mika Nupposelle, joiden väsymätön avustus piti työn tiellään. Kiitän myös Elektroniikan, tietoliikenteen ja automaation tiedekuntaa, sekä ylläpitoyksikköä että kansliaa, tuesta ennen ja työn aikana.

Otaniemi, Marraskuussa 23, 2009

Risto Järvinen

Sisältö

Lyhenteet

ADSL	Asymmetric Digital Subscriber Line, tietoliikenneyhteystekniikka.
AMHS	Aeronautical Message Handling System, ilmailuviestintäjärjestelmä.
ANSI	American National Standards Institute, Yhdysvaltalainen standardointiorganisaatio.
AODV	Adhoc On-demand Distance Vector, reititysprotokolla.
API	Application Programming Interface, sovellusohjelmointirajapinta.
ARPA	Advanced Research Projects Agency, Yhdysvaltojen puolustusministeriön tutkimusorganisaatio.
ASCII	American Standard Code for Information Interchange, merkistöstandardi.
CCEB	Combined Communications Electronics Board, sotilasviestinnän standardointiorganisaatio.
CNR	Combat Net Radio, radiojärjestelmä.
CSMA/CD	Carrier Sense Multiple Access / Collision Detect, monen käyttäjän lähiverkkotekniikka.
DCE	Distributed Computing Environment, ohjelmistojärjestelmä.
DSN	Delivery Status Notification, sähköpostin raportointimekanismi.
DTN	Delay Tolerant Networking, häiriönsietoiset verkkomenetelmät.
DVB	Digital Video Broadcasting, digitaalinen televisiotekniikka.
EDI	Electronic Data Interchange, tiedonsiirtomenettely.
EGNOS	European Geostationary Navigation Overlay Service, paikannustekniikka.
ETSI	European Telecommunications Standards Institute, eurooppalainen standardointijärjestö.

Flash-OFDM	Fast Low-latency Access with Seamless Handoff Orthogonal Frequency Division Multiplexing, langaton verkkotekniikka.
FSF	Free Software Foundation, vapaan ohjelmiston järjestö.
GCC	GNU Compiler Collection, ohjelmistokehityksen työkalupaketti.
GNU	GNU's Not Unix, vapaan ohjelmiston projekti.
GPS	Global Positioning System, paikannustekniikka.
GSM	Global System for Mobile communications, matkapuhelinstandardi.
HomePNA	Home Phoneline Network Alliance, tietoliikenneyhteystekniikka.
HTML	Hypertext Markup Language, dokumenttien kuvauskieli.
HTTP	Hypertext Transfer Protocol, WWW-siirtoprotokolla.
ICAO	International Civil Aviation Organization, Kansainvälinen siviili-ilmailujärjestö.
IANA	Internet Assigned Numbers Authority, kansainvälinen organisaatio.
IEEE	Institute of Electrical and Electronics Engineers, kansainvälinen organisaatio.
IETF	Internet Engineering Task Force, avoin standardointiorganisaatio, joka kehittää ja julkistaa Internet standardeja
IDL	Interface Definition Language, rajapintojen kuvauskieli.
IEC	International Electrotechnical Commission, kansainvälinen sähkötekniikan standardointijärjestö.
IMAP	Internet Mailbox Access Protocol, protokolla sähköpostiviestien noutamiseen.
IPN	Interplanetary Internet, planeettojen välinen Internet.
IRC	Internet Relay Chat, pikaviestintäprotokolla.
IrDA	Infrared Data Association, infrapunatekniikkojen erikoisryhmä.
ISO	International Organization for Standardization, kansainvälinen standardointijärjestö.
ITU-T	International Telecommunication Union Telecommunication Standardation Sector, kansainvälinen televiestinnän standardointiorganisaatio.
LAN	Local Area Network, lähiverkko.
LEMONADE	License to Enhance Message Oriented Network Access for Diverse Environments, sähköpostin kehitysprojekti.

LEO	Low Earth Orbit, matalan kiertoradan satelliittiluokitus.
LDAP	Lightweight Directory Access Protocol, hakemistoprotokolla.
LMTP	Local Mail Transport Protocol, sähköpostin siirtoprotokolla.
MANET	Mobile Adhoc Network, mobiiliverkkotekniikka.
MHS	Message Handling System, viestintäjärjestelmä.
MIDL	Microsoft Interface Description Language, rajapintojen kuvauskieli.
MIME	Multipurpose Internet Mail Extension, sähköpostin rakennelaajennus.
MMHS	Military Message Handling System, armeijan viestintäjärjestelmä.
MPEG	Moving Picture Experts Group, digitaalisen median asiantuntijaryhmä.
MSRPC	Microsoft Remote Procedure Call, viestintäprotokolla.
NAT	Network Address Translation, verkon osoitteenmuunnos-tekniikka.
NATO	North Atlantic Treaty Organization, Pohjois-Atlantin liitto, kansainvälinen puolustusliitto.
NNTP	Network News Transfer Protocol, uutisryhmien siirtoprotokolla.
OVT	Organisaatioiden välinen tiedonsiirto, tiedonsiirtomenettely.
PAN	Personal Area Network, lyhyen matkan lähiverkko.
PDH	Plesiochorous Digital Hierarchy, televerkkotekniikka.
PMR	Professional Mobile Radio, radiopuhelintyyppi.
POP	Post Office Protocol, protokolla sähköpostiviestien noutamiseen.
POSIX	Portable Operating System Interface for uniX, siirrettävä ohjelmointirajapinta.
PSK31	Phase Shift Keying 31 baud, radioamatöörien radiotekniikka.
QMTP	Quick Mail Transport Protocol, sähköpostin siirtoprotokolla.
RPC	Remote Procedure Call, ohjelmistojen viestintäprotokolla.
RFC	Request For Comments, julkaistu avoin muistio.
RTP	Real-time Transport Protocol, tosiaikaisen datan siirtoprotokolla.
RTTY	Radioteletype, radiokaukokirjoitus.
SDH	Synchronous Digital Hierarchy, televerkkotekniikka.
SMTP	Simple Mail Transfer Protocol, sähköpostin siirtoprotokolla.

SONET	Synchronous Optical Networking, televerkkotekniikka.
STANAG	Standardization Agreement, standardointisopimus.
SUSv3	Single Unix Specification versio 3, järjestelmästandardi.
TETRA	Terrestial Trunked Radio, radiopuhelintekniikka.
TMC	Traffic Message Channel, liikennetiedotusjärjestelmä.
UID	Unique Identifier, yksilöivä tunnus.
UCS	Universal Character Set, merkistönkoodausmäärittely.
UMTS	Universal Mobile Telecommunications System, matkapuhelinstandardi.
UTF	Unicode Transformation Format, merkistönkoodausmäärittely.
UUCP	Unix-to-Unix-Copy, protokolla viestien ja tiedostojen siirtoon.
WiMAX	Worldwide Interoperability for Microwave Access, langaton verkkotekniikka.
WLAN	Wireless LAN, langaton lähiverkko.
WPAN	Wireless Personal Area Network, langaton lyhyen matkan lähiverkko.
WWW	World Wide Web, hajautettu hypertekstijärjestelmä.
XMPP	Extensible Messaging and Presense Protocol, pikaviestintäprotokolla.
XML	Extensible Markup Language, tiedon esityskieli.

Luku 1

Johdanto

Työssä käsitellään sähköisen viestinnän sovittamista haasteellisen verkon välitysjärjestelmiin. Sähköisen viestinnän menetelmät ovat kasvaneet ja kehittyneet eri suuntiin, joten työssä pyritään rakentamaan erilaisten järjestelmien väliin sopiva välitysohjelmisto.

Sähköisen viestinnän merkitys yhteiskunnassa on kasvanut huomattavaksi viimeisen kahden vuosikymmenen aikana. Internet on kasvanut kuriositeetista olennaiseksi osaksi päivittäistä toimintaa. Huolimatta Internetin levinneisyydestä ympäri maailman sen kattavuus ei ole täydellinen. Monissa paikoissa maapallolla ei ole riittävää telepalveluverkkoa tarjoamaan Internet-palveluita, ja maapallon pinnalta lähdettäessä viestintälinkkien kaistat kapenevat ja kulkuajat kasvavat siinä määrin, että normaaleja protokollia ei voi käyttää.

Internetin kattavuutta laajentamaan kehitetään viiveriippumattomia verkkoja (Delay Tolerant Networking, DTN) [1], tarkoituksena rakentaa uudenlaisia sovelluksia, esimerkiksi planeettojen välinen Internet (Interplanetary Internet, IPN) [2]. Viiveriippumattomat verkot on tarkoitettu toimimaan heterogeenisissä verkoissa, joissa yhteydet eivät ole jatkuvia.

Toinen Internetin saatavuutta parantava uusi ratkaisu on ns. silmukkaverkotus (engl. mesh networking), jossa useampi verkkolaite tekee dynaamisesti yhteistyötä liikenteen välittämisessä. Silmukkaverkotusta käytetään yleensä kun tarvittavaa verkkoinfrastruktuuria ei ole, mutta silti halutaan kattava tavoitettavuus verkkoon.

Näitä ja vastaavia tekniikoita kutsutaan tässä yhteydessä erityisvälitysverkoiksi. Erityisvälitysverkot ovat itsenäisiä tai toimivat Internetin laajennuksina paikkoihin, jonne tavanomaiset yhteydet eivät sovellu. Koska erityisvälitysverkot eivät toimi aivan kuten perinteinen Internet, tulee sovelluksia muokata, jotta ne toimisivat tässä ympäristössä.

Sähköposti yhtenä merkittävimmistä Internetin palveluista on yksi helpoimpia muuntaa toimimaan erityisvälitysverkoissa, koska alkuperäiseltä rakenteeltaan Internetin vanhimmat protokollat on suunniteltu toimimaan verkossa, jossa tavoitettavuus oli usein heikko, verkon hierarkia monimutkainen ja siirtoajat pitkiä. [3] Tämä mahdollistaa sähköpostin sovittamisen rajapinnaksi erityisvälitysverkkojen yli tapahtuvaan kommunikointiin.

Sähköposti kehittyi monenkäyttäjän palvelinten sisäisestä viestinnästä, joka välitettiin tietoliikenneyhteyksiä pitkin muille palvelimille rypäsajoina. Internet-yhteyksien yleistyessä sähköpostiohjelmat erkanivat omiksi sovelluksikseen, jotka välittivät postit suoraan sähköpostipalvelimelta toiselle. Kiinteiden jatkuvasti verkossa olevien palvelimien myötä sähköpostin välitys yksinkertaistui ja rypäsajojen suorittaminen jäi enää soittoyhteyksien käyttäjille.

Tietokoneiden yleistyessä sähköpostin asiakassovellukset kehittyivät ja pienemmätkin tietokoneet pystyivät lukemaan sähköpostia vaikkeivät sisältäneetkään erillistä sähköpostipalvelinta. Nykyään liki kaikki Internetiin kytkettävissä olevat laitteet pystyvät vastaanottamaan sähköpostia.

1.1 Tavoitteet ja menetelmät

Tässä työssä tutkitaan sähköpostiviestinnän sovittamista erilaisiin välitysverkkoihin ja tuotetaan ohjelmistokomponentti, jota voidaan käyttää välitysjärjestelmäohjelmiston osana. Ohjelmistokomponentin tarkoituksena on toimia rajapintana, joka tarjoaa asiakassovelluksille perinteisen liittymän, mutta samaan aikaan mahdollistaa erityisvälitysverkkojen käyttämisen viestien välittämiseen.

Yhteensopivuus

Ohjelmiston tärkein ominaisuus on olla yhteensopiva Internet-standardien ja -sovellusten kanssa. Rajapinnat on yleensä standardoitu, mutta sovellukset usein laajentavat (de facto -standardeja) tai rikkovat osia standardeista. Yhteensopivuuden takaamiseksi suoritetaan kirjallisuustutkimus alan standardeihin ja toteutusohjeisiin. Ohjelmistoa testataan eri sovellustoteutuksia vastaan.

Toiminnan luotettavuus

Ohjelmisto suunnitellaan luotettavaksi viestijärjestelmän osaksi, jotta sitä voidaan käyttää autonomisena ja skaalautuvana komponenttina. Viestintäohjelmistot usein toimivat sulautetussa ympäristössä, joten resurssienkäytön suhteen pitää olla sääs-

teliäs. Toiminnan luotettavuus on yksi pääkriteereistä ratkaisumallia suunnitellessa. Luotettavuutta voidaan arvioida sovellustesteillä ja lähdekoodia ja sen suorittamista analysoivilla työkaluilla.

Laitteistoriippumattomuus

Ohjelmisto kehitetään mahdollisuuksien mukaan laitteistoriippumattomaksi. Laitteistoriippumattomuus tarkoittaa pitäytymisessä standardoiduissa rajapinnoissa, kuten POSIX-standardin määrittelemissä. Tapauksissa missä riippumattomuus ei ole mahdollista, tehdään toteutus eri rajapinnoille. Ohjelmiston ja sen osien laitteistoriippumattomuudesta tehdään arviointi. Riippumattomuuden toteamiseksi ohjelmisto pyritään testaamaan eri ohjelmistoalustoilla.

1.2 Rajaukset

Työssä toteutetaan erityisvälitysverkkoja käyttävän viestintäohjelmiston liityntäkomponentti. Komponentille ei toteuteta omaa käyttöliittymää. Työssä ei toteuteta erityisvälitysverkkojen siirtomenetelmiä, mutta suunnittelussa otetaan huomioon erilaisten välitysverkkojen erityistarpeita.

Työn analysoinnissa käsitellään ohjelmistojen laaduntarkkailua, mutta alueeseen ei paneuduta syvällisesti. Laaduntarkkailun osalta keskitytään vain työkaluihin, joita on kehitystyössä on käytetty.

1.3 Tulokset

Työssä suoritetaan tutkimus sähköisen viestinnän menetelmiin ja kehitetään tämän pohjalta rajapinta viestinnän sovittamiseen erityisvälitysverkkojen kanssa yhteensopivaksi. Rajapinta toteutetaan ohjelmistokomponenttina, jonka toiminta varmistetaan ohjelmistokehitystyökaluilla ja sovellusyhteensopivuustesteillä.

1.4 Työn rakenne

Työn rakenne on seuraava: Luvussa 2 käsitellään työhön liittyvä kirjallisuustutkimus. Valittu ratkaisumalli ja sen toteutukseen liittyvät valinnat käsitellään luvussa 3, jonka jälkeen luvussa 4 käsitellään toteutetun sovelluksen rakennetta, toimintaa ja valittuja komponentteja. Lopuksi luvussa 5 käsitellään toteutetun sovelluksen testauksen tuloksia, toteutuksessa tehtyjä kompromisseja ja sovelluksen kehitysmahdollisuuksia.

Luku 2

Viestintä tietoliikenneverkoissa

Tässä luvussa käydään läpi tietoliikenneverkkojen viestinvälitysrajapintoja ja toimintatapoja, sekä erityisvälitysverkkojen rakenteita ja toimintatapoja. Esittely painottuu asiakastekniikkaan ja yleisimpiin Internet-pohjaisiin tekniikkoihin. Tarkoituksena on antaa yleiskuva viestintämenetelmistä, niiden rajoituksista ja toteutuseroista. Eri viestintämenetelmien yhteensovittaminen on työssä toteutettavan ohjelmistokomponentin päätarkoitus, joten erilaisten järjestelmien rajoitusten ja mahdollisuuksien tunteminen tärkeää.

2.1 Internet-liittymät

Internet on saavuttanut globaalin vallitsevan aseman sähköisen viestinnän väylänä ja siten kattaa suurimman osan päivittäisestä viestinnästä. Internetin ydin on Internet Protocol (IP) [4]-protokolla, joka on pakettivälitteinen verkkotekniikka. Internetin toiminnan perusoletuksena on, että kaikki siihen kytketyt laitteet ovat jatkuvassa yhteydessä. Internetin reititysprotokollat osaavat kiertää katkenneiden yhteyksien ympäri. Täysin katkeamattomat yhteydet eivät luonnollisesti ole mahdollisia, mutta oletus on, että kaikki katkot ovat väliaikaisia ja yrittämällä vähän ajan kuluttua uudestaan vika olisi korjautunut. Yhteydet ovat päästä päähän, suoraan lähettävältä taholta kohdetaholle, välittäjien ollessa sovellustasolle läpinäkyviä.

Internet on yleistynyt kulutustuotteeksi ja Internet-palveluntarjoajat myyvät palveluna erilaisia tapoja liittyä Internet-verkkoon. Ensimmäisenä yleistynyt tällainen palvelu oli telepalveluista kehittyneet soittoyhteyteen perustuva Internet-liittymä (engl. dialup). Soittoyhteyksien ominaisuuksiin kuuluu yleensä, että käyttäjille jaetaan osoitteet dynaamisesti ilman varauksia, jolloin käyttäjällä ei ole pysyvää osoitetta josta hänet löytäisi. Soittoyhteyspohjaisissa liittymissä ei myöskään ole järjes-

telyä yhteyden avaamiseksi, jos käyttäjä on sulkenut yhteytensä, mutta Internetistä haluttaisiin ottaa häneen yhteyttä. Soittoyhteyteen perustuvien liittymien maksut muodostuvat yleensä yhteysajan mukaan, jolloin kulujen säästämiseksi yhteys katkaistaan jos ei ole yhteydellä ei ole ollut vähään aikaan toimintaa. [5]

Moderneja kaapeloituja Internet-liittymätyyppejä ovat Asymmetric Digital Subscriber Line (ADSL)- [6], Single-pair High-speed Digital Subscriber Line (SHDSL)- [7], kaapelimodeemi- [8] ja Home Phoneline Networking Alliance (HomePNA) [9] -järjestelmät. Koska näiden suorituskyky verrattuna vanhempiin soittosarjayhteyksiin oli suurempi näitä kutsutaan yleisnimellä laajakaistaiset Internet-liittymät. Nämä ovat ns. viimeisen mailin -ratkaisuja, joiden tarkoitus on toimia yhteytenä lähimpään tietoliikennekeskukseen, josta on nopeampi runkoyhteys eteenpäin. Kaikilla näillä on sama palvelunkuvaus; suhteellisen nopea siirtonopeus, jatkuva yhteys ja kiinteä kuukausimaksu. Osalla ratkaisuista on asymmetrinen siirtokapasiteetti: siirtämiseen runkoverkosta käyttäjälle varataan enemmän kapasiteettia kuin käyttäjältä verkkoon lähtelle liikenteelle. Osoitteet jaetaan yleensä dynaamisesti, mutta osassa liittymistä voidaan käyttää staattista osoitetta. Käyttöä rajoittavat lähinnä päätelaitteiden toimintavarmuus, mahdolliset häiriöt yhteyksissä ja palvelusopimuksen käyttöehdot.

Lähiverkot

Lähiverkoilla tarkoitetaan yleisesti kaikkia kiinteistön sisäisiä tietoliikenneverkkoyhteyksiä. Lähiverkkojen yleisin tarkoitus on jakaa Internet-yhteyksiä kiinteistön sisällä; sen sijaan että jokainen laite olisi kytketty omalla liittymällä Internetiin, laitteet jakavat lähiverkon kautta yhden Internet-liittymän. Tämän lisäksi lähiverkkoja käytetään usein mm. paikallisten resurssien, verkkolevyjen, tulostimien, yms. jakamiseen ja työryhmien ja organisaatioiden sisäisten tietojärjestelmien käyttämiseen.

Ehdottomasti yleisin lähiverkkotekniikka on Ethernet, joka on syrjäyttänyt markkinoilla kaikki kilpailijansa. Ethernet on pakettipohjainen tietoverkkojärjestelmä, joka on määritelty IEEE 802.3 -perheen standardeissa [10]. Ethernet-verkko käyttää välitysmedianaan joko kierrettyä parikaapelia tai valokuitua, joista jälkimmäistä kustannussyistä käytetään yleensä vain runkoyhteyksissä. Historiallisesti Ethernet käytti koaksiaalikaapelointia ja törmäyksen havaitsemismenettelyä (Carrier Sense Multiple Access / Collision Detection, CSMA/CD), mutta näistä on luovuttu kaksisuuntaisen (engl. full-duplex) viestinnän mahdollistaviin kaapeleihin siirryttäessä. Moderneissa kiinteistöissä on Ethernet-verkolle yhteensopiva yleiskaapelointi. Ether-

net-verkkojen siirtonopeus on koaksiaalikaapelissa 10 Mbit/s, parikaapelilla 10-1000 Mbit/s ja valokuiduilla 10-10000 Mbit/s. Kehysrakenteen pysyessä yhteensopivana varsinaisia siirtomenetelmiä on voitu kehittää tekniikan parantuessa.

Ethernet-verkot ovat hyvin luotettavia, siirtonopeudet hyvin suuret ja loppukäyttäjän laitteisto on hinnoiltaan halpaa. Kaapelointi voi muodostaa merkittävän kertakustannuserän, mikäli sitä ei ole kiinteistössä valmiina. Koska lähiverkot yleensä omistaa käyttävä yhteisö, niillä ei ole juoksevia kuluja laitteiston uusinnan ja sähkönkulutuksen lisäksi. Internet-liittymänä yhteyden laatua rajoittaa lähiverkon ulkoyhteytenä toimiva liittymä.

Langatonta lähiverkkoa (WLAN, Wireless Local Area Network) käytetään usein Internet-yhteyksien jakeluun kiinteistöissä, joissa ei ole tai on riittämätön yleiskaapelointi, tai jos kaapeleita halutaan mukavuussyistä välttää. WLAN tekniikkoja on standardoitu IEEE 802.11-sarjan standardeihin joista yleisimmät ovat IEEE 802.11b [11], IEEE 802.11g ja IEEE 802.11n. Langaton lähiverkko on yleensä kiinteästi kytketyn Internet-liittymän lisänä. Langatonta lähiverkkoa jaetaan myös ns. WLAN-kohdealueina (engl. hotspot), esimerkiksi kauppakeskuksen kiinteistön sisällä. Tällöin palvelu on yhdistetty kauppakeskuksen pääasiallisen toiminnan lisäarvopalveluksi.

Langattoman lähiverkon rajoittavina tekijöinä on tukiasemien sijoittelun muodostama peittoalue, suorituskyvyn vaihtelu peittoalueen sisällä ja mobiililaitteilla langattoman verkon käytön tehonkulutus. Langattomien lähiverkkojen asentaminen tapahtuu usein myös ilman koordinointia, jolloin tukiasemat voivat aiheuttaa toisilleen häiriöitä käyttämällä samoja taajuuskanavia. Lisäksi langattomat lähiverkot usein eivät anna käyttäjille julkisia IP-osoitteita, vaan käyttävät yksityisiä osoitteita osoitteenmuunnostekniikalla (engl. NAT, Network Address Translation) [12].

Mobiiliviestintäpalvelut

Mobiiliviestintäpalvelut ovat saavuttaneet merkittävän aseman viestinnässä. Euroopassa vallitsevat toteutus on Global System for Mobile communications (GSM) [13]. GSM-järjestelmän kehittäminen aloitettiin 80-luvulla ja otettiin ensimmäisenä maailmassa käyttöön Suomessa vuonna 1991. GSM on ns. toisen sukupolven (2G, 2nd Generation) mobiiliviestintäpalvelu.

Mobiiliviestintäpalvelut tarjoavat puhe- ja tekstiviestipalvelun lisäksi piiri- tai pakettikytkentäisiä Internet-liittymäpalveluja. Näissä palvelumaksu on joko kiinteä, siirrettyyn datamäärän mukaan (pakettikytkentäiset) tai käytetyn yhteysajan mukaan (piirikytkentäiset). Pääasialliset rajoittavat tekijät ovat mobiililaitteen akkuka-

pasiteetti, tukiasemien peittoalue ja viestinnän kustannukset. Tehonkulutuksen rajoittamiseksi mobiililaitteet yleensä minimoi ajan jonka mobiilidatayhteys on päällä. Puhelimen ollessa poissa päältä saapuvat puhe- tai datayhteysyritykset epäonnistuvat. Mikäli puhelimeen ei saa yhteyttä, tekstiviestit puskuroituvat tekstiviestipalvelukeskukseen, josta ne puretaan kun puhelin kytkeytyy takaisin päälle tai poistetaan tietyn ajan kuluttua.

Ominaisuuksiltaan ja suorituskyvyltään parannettuja ns. kolmannen sukupolven (3G) ratkaisuja on myös otettu käyttöön, ensimmäisenä näistä oli mobiiliviestintäverkko verkko Japanissa vuonna 2001, joka toimi pohjana Universal Mobile Telecommunications System (UMTS) -järjestelmälle [14]. 3G-verkkojen mobiiliviestintäpalvelut ovat samat kuin niiden edeltäjillä, mutta toteutustekniikan parantuminen on nostanut järjestelmän suorituskykyä ja siirtokapasiteettia. Mobiiliviestintäpalveluiden siirtokapasiteetti on yleensä asymmetrisesti jaettu: runkoverkosta käyttäjälle tulevalle liikenteelle on varattu suurempi siirtokapasiteetti. Parantunut suorituskyky on mahdollistanut tätä hyväksi käyttäviä uusia palveluita, kuten videopuhelut ja mobiililiittymän käyttämisen pääasiallisena Internet-liittymänä.

Kolmas globaalisti merkittävä, mutta vähemmän käytetty mobiiliviestintäpalvelu on satelliittipuhelinjärjestelmät, [15] joista Iridium [16] on vallitseva ratkaisu. Iridium-järjestelmä perustuu 66 matalan kiertoradan satelliitin (LEO, Low Earth Orbit) muodostamaan verkkoon, jolla saavutetaan globaali peittoalue. Satelliittipuhelinten palvelumalli on sama kuin muillakin mobiililaitteilla, mutta kapasiteetti hyvin paljon rajatumpi ja viestinnässä on pitkästä kulkumatkasta johtuen huomattavasti latenssia. Iridium-järjestelmä pystyy välittämään puheyhteyksiä, tekstiviestejä ja Internet-liikennettä, mutta Internet-liikenne tarvitsee erikoiskäsittelyn, koska tiedonsiirron edestakainen kulkuaikaviive (engl. Round-Trip Time, RTT) on keskimäärin 1.8 sekuntia [17]. Satelliittipuhelinpalvelu on myös huomattavan kallis verrattuna maanpäällisiin palveluihin, johtuen pienemmästä käyttäjämäärästä ja korkeista operointikustannuksista.

Mobiiliviestintäpalveluiden lisäksi on ns. langattomia laajakaistatekniikoita. Nämä ovat mobiiliverkkoja, jotka eivät perustu puhelinpalveluun, vaan ovat kehitetty välittämään pakettipohjaista Internet-liikennettä. Suomessa tällaisia tekniikoita käytössä on IEEE 802.16-sarjan standardeihin kuuluva Worldwide Interoperability for Microwave Access (WiMAX) [18] -standardin mukainen verkkotekniikka ja markkinointinimellä “@450” oleva verkkotekniikka, joka perustuu Fast Low-latency Access with Seamless Handoff Orthogonal Frequency Division Multiplexing (Flash-OFDM) [19] -tekniikkaan. @450-verkko tarjoaa koko maan kattavan peittoalueen. WiMAX tarjoaa nopeammat yhteysnopeudet, mutta on peittoalueeltaan rajoittu-

neempi. Kummankin palvelumalli on siirtonopeusrajan mukaan porrastettu kiinteä kuukausimaksu.

Edellä mainittujen julkisten mobiiliverkkojen lisäksi on myös ammattilaiskäyttöön tarkoitettuja PMR-verkkoja (Professional Mobile Radio). PMR-verkot ovat ensisijaisesti puhepohjaiseen ryhmäviestintään, mutta hiljattaiset toteutukset, kuten TETRA (TErrestial Trunked RAdio) [20], sisältävät myös tekstiviesti- ja piiri- ja pakettikytkentäiset tietoverkkoliittymät. PMR-verkot eivät yleensä ole kytketty puhelinverkkoihin, vaan toimivat näiden rinnalla. PMR-verkkojen tietoverkkoliittymät ovat yleensä vain sovelluskohtaisessa käytössä, esimerkiksi viranomaisten autot raportoivat sijaintinsa viestintäkeskukselle.

2.2 Muut tietoliikenneverkot

Kaikki tietoverkot eivät ole Internetiin kytkettyjä, tai edes perustu Internet-tekniikoihin. Tärkein Internetistä riippumaton tietoliikenneverkko ovat maailmanlaajuinen televerkko, joka koostuu sadoista eri teleoperaattoreiden televerkoista. Aiemmin kuvattujen Internet-liittymien lisäksi televerkot mahdollistavat datayhteydet liittymästä toiseen.

Poiketen Internet-tyylisestä pakettipohjaisesta viestinnästä, televerkkojärjestelmät yleensä käyttävät piirikytkentäistä viestintää, jossa yhteydelle varataan kiinteä jatkuvasti käytettävissä oleva kaista. Televerkkojärjestelmien tärkeimpiä suunnittelukriteerejä on pieni latenssi, sillä jo yli 200 ms viive aiheuttaa havaittavan häiriön puhekeskusteluun. Muita tärkeitä ominaisuuksia järjestelmän kannalta ovat kansallinen ja kansainvälinen yhteistoiminta, globaalisti tavoitettava osoiteavaruus ja resurssien käytön seuranta laskutusta varten. Yleiset televiestinnän runkojärjestelmät ovat Plesiochorous Digital Hierarchy (PDH) -järjestelmä ja tätä korvaavat teknisesti kehittyneemmät eurooppalainen Synchronous Digital Hierachy (SDH) ja Yhdysvaltojen Synchronous Optical Networking (SONET) -järjestelmät [21]. Perusidea näissä on hierarkisesti käsitellä ja varata siirtoresursseja käyttäen perusyksikkönä yhtä puhekanavaa. Näin televerkkojärjestelmästä voidaan varata tietty siirtokaista kahden pisteen väliseen yhteyteen. Käytännössä kaikkia televerkon osia voidaan käyttää osana Internetiä, kunhan televerkon yhteyden kummassakin päässä on järjestelmä, joka yhdistää yhteyden Internetiin. Heikkoutena verrattuna pakettipohjaiseen viestintään on, että hierarkisen televerkon yhteydet ovat usein staattisia ja käyttämätön kapasiteetti menee hukkaan. Osana ns. digitaalista konvergenssiä [22] televerkkoja on liitetty toimimaan osana IP-verkkoja, jolloin voidaan tehokkaammin jakaa kapasiteetti puheviestinnän ja Internet-liikenteen kanssa.

Vastaavasti mobiililiittymistä voidaan ottaa suoria yhteyksiä toisiin mobiililiittymiin, mutta tämä on resurssien käytön kannalta huono ratkaisu ja yleensä ei operaattoreiden sallima, koska liikennemääriä ei pysty seuraamaan. Mobiililiittymissä on myös tekstiviestipalvelu, joka on Internetistä riippumaton viestintätapa. Tekstiviestit lähetetään runkoverkossa olevaan viestikeskukseen, josta ne välitetään vastaanottajalle. Tekstiviestipalvelut ovat hyvin suosittuja ja pienen resurssien käyttönsä takia operaattoreille edullisia toteuttaa.

Julkisien tieto- ja televerkkojen lisäksi sähköistä langatonta viestintää voi suorittaa myös muista riippumattomilla radioviestintäjärjestelmillä. Käytännössä tämä tarkoittaa, että varataan taajuusalue sovelluksen radioviestintää varten ja kalustetaan sopivat laitteistot viestintään. Taajuusalueiden jakelua hoitavat kansalliset (Suomessa Ficora) ja kansainväliset järjestöt, mm. ITU ja ETSI. Taajuusalueilla on aina maantieteellisesti rajatut käyttöalueet ja lähetystehot. Taajuusalueiden hallinnoinnilla on tarkoitus taata kansainvälisesti eri radiojärjestelmille häiriötön toiminta. Kansalliset järjestöt suorittavat halutuimpien taajuusalueiden myynnin usein huutokaupalla. Osa taajuusalueista on vapaassa käytössä tiettyjä sovelluksia varten ja muutama vapaita kaikkeen käyttöön. Esimerkiksi WLAN-tekniikat käyttävät kaikkeen käyttöön vapaata taajuusaluetta, kun taas @450-verkko käyttää sovellukselle lisenssoitua taajuusaluetta. [23]

Taajuusalueen lisäksi radioviestintäjärjestelmä tarvitsee viestintään radiolaitteiston ja viestintäkäytännön. Radioviestintäjärjestelmä voi olla rakennettu yhdeltä-yhdelle tai yhdeltä-monelle -tyylisen viestinnän pohjalle. Käytännössä järjestelmät ovat pitkälti sovelluskohtaisia ja toteuttajakohtaisia, näistä esimerkkinä Yhdysvaltain armeijan Combat Net Radio (CNR), joka on määritelty standardissa MIL-STD-188-220 [24]. Radioviestintäjärjestelmän suorituskyky riippuu fyysisellä tasolla lähetystehoista, käytetyistä antenneista, vastaanottimen etäisyydestä lähettäjästä, lähetyksen kulkureitistä ja vastaanotetuista häiriöistä, sekä käytetystä radiokanavan jakomenettelystä.

Langatonta viestintää on myös mahdollista harrastaa ilman kaupallista toimintaa. Radioamatööreille on myönnetty lupa harrastaa radiotoimintaa tietyin ehdoin. Radioamatööritoiminta vaatii kansallisen organisaation myöntämän radioamatöörisertifioinnin ja kansallisten sekä kansainvälisten radioamatöörisäännösten noudattamista. Radioamatöörien viestintä on sääntöjen mukaan aina avointa, salaamatonta ja ei saa sisältää kaupallista toimintaa. Radioamatöörikäyttöön on varattu joukko taajuuskaistoja.

Radioamatöörit käyttävät morsetuksen ja suoran analogisen puhe- ja videoviestinnän lisäksi myös digitaalisia viestintäjärjestelmiä. Radioamatöörikäytössä olevia

digitaalisia viestintäjärjestelmiä on kahden tyyppisiä: suoraan kirjoitukseen perustuvat menetelmät, mm. RTTY (radioteletype, suom. radiokaukokirjoitus) ja PSK31 (Phase Shift Keying 31 baud), sekä pakettivälitteiset menetelmät, esimerkiksi PACTOR (latinaa, suom. sovittelija). Osoiteavaruutena kaikessa radioamatööriviestinnässä toimivat henkilö- tai asemakohtaiset tunnukset. Radioamatöörien viestinnän rajoituksina on käytettävissä oleva laitteisto, sallitut radiokaistat, häiriölähteet ja ilmakehän tila. Koska kaikkien radioamatöörilähetyksiä tekevien on oltava radioamatöörejä, tämä viestintämuoto on rajoittunut vain omistautuneiden harrastajien käytettäväksi. [25]

Sähköinen joukkoviestintä on viestinnän muoto, jossa kaikki lähetyksen peittoalueella voivat vastaanottaa samaa viestiä. [22] Sähköinen joukkoviestintä on ensisijaisesti yksisuuntaista, mutta jotkin sovellukset ovat käyttäneet paluukanavaa. Suomessa näitä palveluja ovat ääniradio- ja televisiolähetykset. Sähköisen joukkoviestinnän välitysmedia on ensisijaisesti maanpäällinen yleisradioverkko, mutta sitä välitetään myös satelliitti- ja kaapeliverkkoja pitkin. Sähköisen joukkoviestinnän pääasiallinen sisältö on ääni- ja videolähetykset, mutta näiden lisäksi voi myös lähettää muista viestejä. Suomessa ääniradiolähetyksissä käytetään Radio Data System (RDS) -järjestelmää ääniradiokanavien oheistiedon ja tiedotusviestien välittämiseen. Esimerkiksi Suomessa liikennetiedotteita (Traffic Message Channel, TMC) lähetetään RDS-signaalina radioliikenteen mukana. Suomessa televisiolähetykset (TV) tehdään Digital Video Broadcasting (DVB) -järjestelmällä [26]. Yhdessä DVB-lähetyskanavassa lähetetään useampi TV-kanava, koodattuna ISO/IEC MPEG-2 (Moving Picture Experts Group) Transport Stream -muodossa [27]. Lähetyskanavassa voi lähettää myös TV-kanavista riippumatonta materiaalia. Käytännössä näin on lähetetty esimerkiksi joidenkin DVB-vastaanottimien ohjelmistopäivityksiä.

Sähköisten joukkoviestintäjärjestelmien suurimmat puutteet ovat paluukanavan puuttuminen ja lähetysten kohdentaminen. Joukkoviestintä on yksisuuntaista ja kaikille osoitettua. Viestintälaitteet tarvitsevat jonkun toisen viestintätavan paluuviestien lähettämiseen. Lähetykset pystytään kohdentamaan vain osalle vastaanottajista, esimerkiksi salaamalla lähetykset. Tätä käytetään mm. maksullisten TV-kanavien toteuttamiseen. Salatut lähetykset kuitenkin kuluttavat kaikille jaettua lähetyskapasiteettia.

Eräänlaisen joukkoviestintäjärjestelmän muodostavat globaalit navigointijärjestelmät, Global Positioning System (GPS) [28] ja European Geostationary Navigation Overlay Service (EGNOS) [29]. Näissä järjestelmissä pääasiallinen tarkoitus on tarjota paikannuspalvelua koodatun signaalin avulla, mutta mukana voidaan siirtää myös muuta tietoa. GPS-järjestelmässä käytännössä siirretään satelliittien sijainti-

tietoa ja satelliittien atomikelloaikaa. Lähetyksillä on koko maapallon kattava peittoalue, mutta siirtokapasiteetti on mitätön, vain 50 bit/s.

2.3 Erityisvälitysverkot

Tässä luvussa käsitellään lyhyesti erityisvälitysverkkojen luonnetta, miten ne poikkeaa perinteisistä tietoliikenneverkoista, millaisia kehityssuuntia erityisvälitysverkot ovat ottaneet ja millaisia käytäntöjä on vakiintunut. Erityisvälitysverkot ovat erikoistuneita tietoliikenneverkkoja, joissa ei käytetä perinteisiä tele- ja tietoliikenneverkkojen ratkaisuja.

Tietoliikenneverkot ovat levittäytymässä alueille ja sovelluksiin, joissa perinteisiä televerkkoja ja infrastruktuuria ei ole saatavilla. Sellaisille alueille, joilla ei ole televerkkojen infrastruktuuria, on langattomista lähiverkoista rakennettu langattomia runkolinkkejä ja silmukkaverkkoja. Joissain sovelluksissa energiatehokkuuden ja luotettavuuden nimissä ei voida käyttää perinteistä runkoverkkoratkaisua.

Lyhyen matkan viestintään on kehitetty ns. Personal Area Network (PAN) ja Wireless Personal Area network (WPAN) -järjestelmiä. [21] PAN-verkkojen tarkoitus on yhdistää henkilökohtaisia laitteita toisiinsa, esimerkiksi mobiilipuhelin handsfree-laitteeseen. Yleisin PAN-verkkotekniikka on Universal Serial Bus (USB), ja yleisimmät WPAN-verkkotekniikat ovat radiotekniikkaan perustuvat Bluetooth [30] ja IEEE-standardiin 802.15.4 [31] perustuva ZigBee [32] ja infrapunatekniikkaan perustuva IrDA [33] (Infrared Data Association). PAN-verkkojen kantama on yleensä alle kymmenen metriä. Vaikka PAN-verkot itsessään eivät kanna pitkälle, niitä voidaan käyttää välittämään muita verkkoyhteyksiä.

Silmukkaverkoissa verkottuneilla laitteilla ei ole runkoverkkoa, vaan laitteet muodostavat siirtoreittinsä keskenään. Näitä välitysmenetelmiä kutsutaan yleisesti adhoc-reititysprotokolliksi. Silmukkaverkoista joiden solmut ovat liikkuvia käytetään termiä mobiilisilmukkaverkko (engl. Mobile Adhoc Network, MANET). Nämä verkkotekniikat ovat vahvasti kehityksen alla, mutta muutamia ratkaisuja on standardoitu, esimerkiksi Adhoc On-demand Distance Vector (AODV) -reititys [34], tai ollaan standardoimassa, esimerkiksi vedos-vaiheessa oleva IEEE 802.11s [35].

Yksi suosittu sovellusala silmukkaverkoille on ns. langattomat sensoriverkot, joissa yksinkertaiset verkottuneet mittalaitteet välittävät mittaustuloksiaan toistensa kautta. Tällöin tiedonkeruulaitteiston ei tarvitse pystyä saavuttamaan kaikkia mittalaitteita jatkuvasti, vaan mittalaitteet voivat välittää toistensa liikennettä. Tämä kasvattaa silmukkaverkon jäsenten käsittelemän tiedon määrää, mutta vähentää kokonaisuutena kaikkien mittalaitteiden tarvitsemia lähetystehoja. [36]

Yhteyksille, joissa normaalien siirtoprotokollien käyttäminen on mahdotonta liian suurten latenssien, korkean pakettihäviön tai lyhytaikaisten yhteysikkunoiden takia, on kehitetty viiveensietäviä siirtoprotokollia (engl. Delay Tolerant Networking, DTN). [2] Nämä protokollat perustuvat yleensä säilytä-ja-välitä -tyylisiin (engl. store-and-forward) ratkaisuihin. Viestit puskuroidaan DTN-verkon laitteiden kesken ja puskurikoot on valittu resurssien mukaan. Nämä siirtoprotokollat ovat myös vahvasti kehityksen alla, mutta joitain ratkaisuja on osittain standardoitu, esimerkiksi kokeiluasteella oleva IETF Bundle -protokolla [37]. Tärkeä sovellus viiveensietävälle siirtoprotokollille on äärimmäisen pitkän matkan tiedonsiirtojärjestelmät, joita käytetään avaruusluotainten kanssa. Näissä sekä siirron latenssi on suuri että siirtokaista pieni.

Erityisvälitysverkkojen etuna on tietoliikenneverkkojen laajentaminen uusiin sovelluksiin, ympäristöihin ja toimintatapoihin. Suurin erityisvälitysverkkojen haitta on että ne eivät sovi perinteisten verkkoyhteyksien malliin: yhteydet eivät ole jatkuvia, yhteydet eivät ole päästä päähän ehjiä, viestejä ei kuitata päästä päähän, siirroissa on suuri latenssi ja pienet siirtonopeudet ja jonotusajat voivat olla hyvin suuria. [1]

2.4 Internet viestintärajapinnat

Tässä luvussa esitellään Internetin viestintäprotokollien nykyistä tilaa, rakennetta ja kehitysaskeleita. Erityistä huomiota keskitetään ominaisuuksiin, joista on hyötyä erityisvälitysverkoissa. Luvussa käsitellään viestien esittämistä, säilömistä ja erilaisia viestintämenetelmiä.

2.4.1 Merkistöt

Sähköisen viestinnän pohjimmainen ongelma on kuinka esittää viesteissä käytettävän kielen abstraktit merkit. Tietokoneet käsittelevät kaikkia asioita numeroina ja merkeillä tai kirjaimilla ei ole itsessään suoraa merkitystä. Tätä varten on määritelty merkistöjä, jotka ovat koodikirja, jonka mukaan merkeille määritellään vastaavat lukuarvot, säilömistapa ja yleensä myös ulkoasu eli glyyfi. Jotta järjestelmät pystyisivät siirtämään viestejä siten, että viestin sisältö olisi tulkittavissa samalla tavalla, tulee sopia käytettävät merkistöt. [38]

Internet-viestinnän yleisin merkistöstandardi, jota käytetään yleisesti oletusarvona, on ASCII (lyhenne sanoista American Standard Code for Information Interchange, myös US-ASCII). ASCII-merkistö määrittelee 128 merkkiä, jotka kattavat

englannin kielen merkit (pienet ja suuret kirjaimet), numerot, joukon välimerkkejä ja kontrollikoodeja, ja niiden koodauksen 7-bittisiksi lukuarvoiksi. ASCII-merkistö on määritelty standardeilla ANSI X3.4, ISO-646-US ja ITU-T T.50. ASCII-merkistö on esitetty taulukossa 2.1. [39]

Taulukko 2.1: ASCII-merkistö, pystyrivit kolmen merkitsevämmän bitin mukaan.

	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	¨	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	´	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	IS4	,	<	L	\	l	|
1101	CR	IS3	-	=	M	]	m	}
1110	SO	IS2	.	>	N	^	n	~
1111	SI	IS1	/	?	O	_	o	DEL

ASCII-merkistö ei kuitenkaan sisällä esimerkiksi kaikkia skandinaavisten maiden käyttämiä merkkejä, eikä yleisemmin kuin murto-osan maailmassa käytettävien kielien merkistöistä. Tätä puutetta korjaamaan on määritelty lukuisia muita merkistöjä, kuten pohjoismaissa suosittu ISO-8859-1 tai ISO-Latin-1.

Unicode on merkistöstandardi, joka pyrkii kattamaan kaikki maailman kielet. Unicode on teollisuusstandardi, jonka kehittämistä The Unicode Consortium -järjestö ohjaa. Viimeisimmän (versio 5.1.0 tätä kirjoittaessa) määritelmän mukaan merkkiavaruus kattaa 100713 merkkiä, ja pystyy esittämään yhteensä 1114112 erilaista merkkiä (17 tasoa joissa kussakin 63356 merkkiä). Unicode-merkit voidaan koodata eri tavoin. Peruskoodauslähtökohtia ovat UTF (Unicode Transformation Format) ja

UCS (Universal Character Set). UTF-merkistöt määrittelevät kuinka kaikki Unicode-merkit voidaan koodata ja UCS-merkistöt esittävät kiinteän kokoisen koodauksen, joka esittää alijoukon Unicode-merkistöstä. Yleisimmät Unicode-koodaustavat on esitetty taulukossa 2.2. Vaihtelevan pituinen koodaus tarkoittaa, että yksittäiset Unicode-merkit koostuvat useista kiinteän kokoisista osamerkeistä. [40]

Taulukko 2.2: Yleisimmät Unicode-koodaustavat.

Koodaus	Kuvaus
UTF-8	8-bittinen, vaihtelevan pituinen, ASCII-yhteensopiva, yleisin.
UTF-7	7-bittinen, vaihtelevan pituinen.
UTF-16	16-bittinen, vaihtelevan pituinen.
UTF-32	32-bittinen, kiinteän pituinen.
UCS-2	16-bittinen, kiinteän pituinen, UTF-16 alijoukko.
UCS-4	32-bittinen, kiinteän pituinen, olennaisesti sama kuin UTF-32.

Unicode on yhdenmukaistettu rinnakkain kehitetyn ISO/IEC 10646 -standardin kanssa. Erona näiden kahden välillä on että ISO 10646 kattaa pelkästään merkistön määrittelyn (UCS-koodaukset) ja Unicode kattaa myös kaksisuuntaisten kirjoitusasujen (heprea, arabia), merkkijärjestyksen ja merkkien normalisoinnin määritykset.

2.4.2 Sähköposti

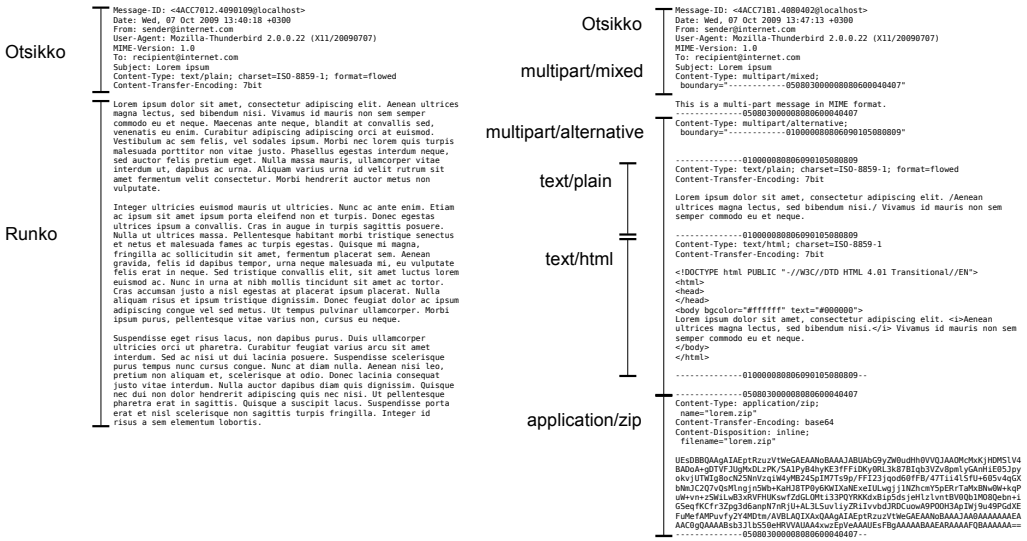
Tietoverkkojen ensimmäisiä ja suosituimpia sovelluksia on sähköposti (engl. email, electronic mail). Aluksi sähköpostit siirrettiin eräajoina palvelimelta toiselle, esimerkiksi Unix-to-Unix-Copy palvelua (UUCP) käyttäen. UUCP palvelu pystyi siirtämään tiedostoja, sähköpostia ja uutisryhmien viestejä. UUCP palvelu kopioi viestit koneelta toiselle yleensä soittoyhteyksiä käyttäen. Järjestelmä ei sisältänyt automaattista reititystä vaan sähköpostin tapauksessa sähköpostin kulkureitti ilmoitettiin sähköpostin osoiteosassa (!-notaatio). [41]

Sähköpostin ydinprotokolla on Simple Mail Transport Protocol (SMTP), joka kehittyi teollisuusstandardina vuosien varrella. SMTP-protokollaa käytetään viestien lähettämiseen asiakkaalta palvelimelle ja palvelimelta toiselle. Viimeisin SMTP protokollan määrittelevä IETF standardi on RFC 5321 [42], ja viimeisin sähköpostissa käytettävän viestirakenteen määrittelevä IETF standardi on RFC 5322 [43]. SMTP-protokollan toimintaa käsitellään lisää kohdassa sähköpostin toimittaminen.

Sähköpostin rakenne

Sähköpostiviesti koostuu tekstiriveistä, jotka päättyvät rivinvaihtoon (ASCII-koodit 13 ja 10) ja ovat korkeintaan 998 merkkiä pitkiä. Tekstiriveistä muodostuu otsikkotiedot ja valinnaisesti viestirunko. Otsikkotietorivit koostuvat otsikkotiedon nimestä, kaksoispisteestä (:) ja otsikkotiedon arvosta. Otsikkotietojen merkistö on ASCII, jos muiden merkistöjen merkkejä halutaan käyttää, ne tulee esittää koodattuna ASCII-yhteensopivaan merkistöön. Useamman rivin otsikkotiedot voidaan esittää siten, että seuraavien rivien alussa on välilyönti (ASCII-koodi 32) tai sarkain-merkki (ASCII-koodi 9).

Sähköpostiviesti koostuu kahdesta pääosasta: otsikoista ja rungosta. Otsikot määrittelevät sovitussa muodossa tietoja viestistä ja sen toimituksesta. Runko sisältää viestin sisällön. Sähköposti voi sisältää ns. MIME-rakenteen (Multipurpose Internet Mail Extension) [44], joka mahdollistaa useiden erilaisten sisältöjen siirtämisen yhdessä sähköpostiviestin rungossa. Esimerkkinä viestien rakenteesta kuvassa 2.1 kohdassa A on normaali sähköpostiviesti ja kohdassa B on MIME-rakenteellinen viesti, joka sisältää viestin paljaana tekstinä (text/plain) ja Hyper Text Markup Language (HTML) -muodossa (text/html), sekä liitteen zip-tiedostona.



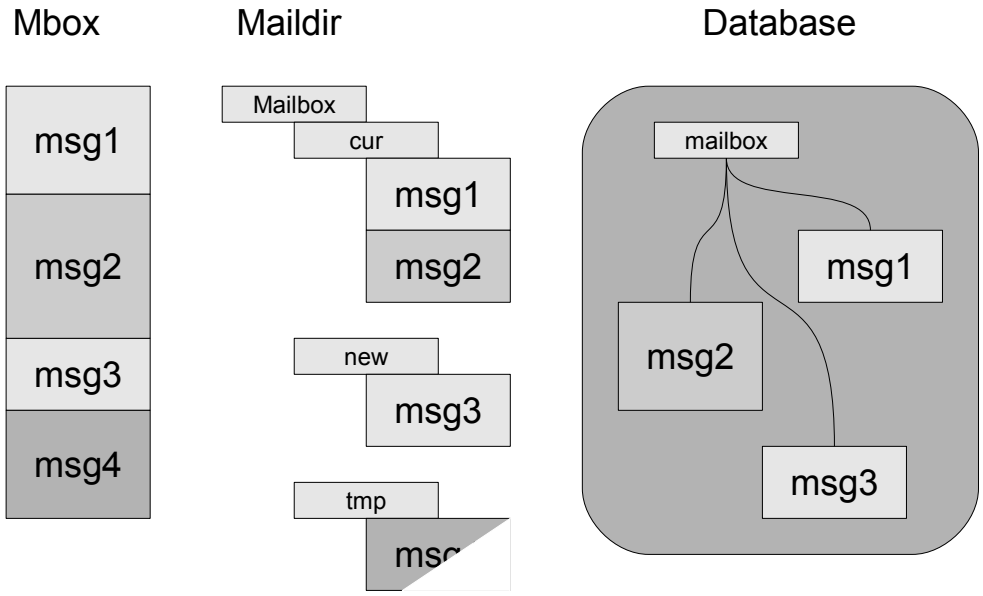
Kuva 2.1: Sähköpostiviestin rakenne, ilman ja MIME-rakenteen kanssa.

Sähköpostiviestin otsikkotietoihin myös voidaan sisällyttää monenlaisia lisäominaisuuksia, mutta vain harvat näistä ovat saavuttaneet standardin aseman. IETF standardit RFC 2076 [45] ja RFC 4021 [46] esittelee vastaavasti yleisiä käytössä olevia otsikkokenttiä ja erityisesti Internet Assigned Numbers Authority (IANA) or-

ganisaation hyväksymät otsikkokentät. Kaikki sähköpostisovellukset eivät toteuta näitä kaikkia, vaan otsikot, joita ei tueta, jätetään käsittelemättä.

Sähköpostien säilöminen

Sähköpostiviestien säilyttämiseen tietojärjestelmissä on useita tapoja. Yleisimmät näistä on mbox-tiedostot, maildir-rakenne ja tietokantajärjestelmät. Säilytystapoja on esitelty kuvassa 2.2. Mbox-tiedostoissa [47] sähköpostiviestit tallennetaan peräkanaa yhteen tekstitiedostoon, ns. from-rivin erottaessa viestit toisistaan. Mbox-tiedostojen rakenne kehittyi sovellusten tarpeen mukaan ja se on hyvin yksinkertainen luoda, mutta viestien määrän kasvaessa epäkäytännöllinen ylläpitää. Mbox-tiedostojen rakenteesta ei ole yksimielistä konsensusta vaan sovellukset ovat muokanneet rakennetta tarpeidensa mukaan.



Kuva 2.2: Erilaisia tapoja säilyttää postilaatikko.

Maildir-rakenne on hakemistorakenne, jossa sähköpostiviestit tallennetaan yksittäin, yksi viesti tiedostoa kohden. [48] Tämä malli soveltuu sähköpostille erinomaisesti, koska viestit toimitetaan yksittäisinä, toisistaan riippumattomina ja muuttumattomina. Maildir sisältää kolme alihakemistoa kutakin postilaatikon kansiota kohden; väliaikaiset (/tmp), uudet (/new) ja säilötyt (/cur) viestit. Toimintamalli on, että viestejä toimittavat ohjelmistot kirjoittavat viestit /tmp-hakemistoon ja kun kirjoitus on valmis, siirtävät ne /new-hakemistoon. /new-hakemistosta Maildir-rakennetta lukeva ohjelmisto siirtää ne /cur-hakemistoon, jonne ne jäävät pysy-

väissäilöön. Tarkoituksena on estää ettei toimitettavia viestejä vahingossa aleta käsittelemään kesken siirron. Tämä toimintatapa mahdollistaa sen, että postilaatikon käyttöä ei tarvitse synkronoida lukoilla toimitus- ja käsittelyohjelmien välillä.

Maildir-rakenne ei alunperin määritellyt tapaa tehdä alikansioita, mutta eri toteutukset ovat tehneet määritelmään laajennuksia. Eräs näistä on ns. "Maildir++"-rakenne, joka on kehitetty Courier IMAP-palvelimen ohella [49]. Päähakemiston ajatellaan olevan käyttäjän pääkansio ("INBOX") ja sen alihakemistot, joiden nimi alkaa pisteellä (".") ovat postilaatikon muita kansioita. Näillä hakemistoilla voi olla vastaavasti nimettyjä alihakemistoja, jotka ovat postilaatikon kansion alikansioita.

Maildir-rakenteeseen on myös määritelty tapa tallentaa lisätietoa viestitiedoston nimeen, mutta nimeämiskäytäntö ei ole riittävä kaikkiin tarpeisiin ja nimenmuutokset hankaloittavat postilaatikon yhtäaikaista käyttöä. Maildir-rakenteista postilaatikkoa käyttävät ohjelmat usein toteuttavat omia lisätietorakenteita sisältämään tiedon, jota pelkkä Maildir-rakenne ei pysty säilyttämään.

Sähköpostiviestejä voidaan myös säilöä tietokantajärjestelmiin. Tämä on hiljattainen, mutta vahva trendi. Tietokantajärjestelmät yksinkertaistavat viestisäilön toteuttamista, mutta monimutkaistuttavat järjestelmää kokonaisuutena. Tietokantoja sähköpostien säilyttämiseen käyttävät yleensä vain sähköpostipalvelimet IMAP- tai webmail-sovelluksissa. Asiakasohjelmat tallentavat viestit yhä Mbox- tai Maildir-muodossa. Tietokantajärjestelmä voi tallentaa sähköpostiviestien kaikki osat muokattavassa ja haettavassa muodossa, sekä indeksoida sisällön automaattisesti eri hakuparametrien pohjalta.

Sähköpostien toimittaminen

Sähköpostin siirtämiseen on monia protokollia, mutta Internet-käytössä liki kaikki siirto palvelimilta toisille tapahtuu SMTP-protokollalla. Muita protokollia käytetään lähinnä erikoiskäytössä, kuten UUCP historiallisessa käytössä, Local Mail Transport Protocol (LMTP) [50], jota käytetään sähköpostijärjestelmien sisäisessä viestienvälityksessä ja Quick Mail Transport Protocol (QMTP) [51], joka on qmail-ohjelmiston tukema, SMTP-protokollaa nopeampi protokolla.

SMTP-protokolla on tekstipohjainen ja sisältää mekanismin protokollalaajennusten mainostamiseen ja käyttöönottoon. Perusprotokolla sisältää yhteyden kättelyn lisäksi vain komennot lähettää sähköpostiviestejä kohdeosoitteille. Itse sähköpostiviesti on kokonaan asiakassovelluksen muodostama.

Internet sähköpostipalvelimet välittävät sähköpostiviestit toisilleen käyttäen ensisijaisesti SMTP-protokollaa. Sähköpostipalvelin halutessaan voi neuvotella siirtoa

varten SMTP-protokollan laajennuksia tai käyttää toista protokollaa (esimerkiksi QMTP), mutta ainoa taattu yhteensopivuustaso on pelkkä SMTP-protokolla. Jokainen siirtopolun palvelin lisää viestin alkuun tiedon viestin vastaanotosta, vikadiagnostiikkoja varten. Mikäli siirrossa tapahtuu virhetilanne, jota ei voida korjata, viestistä luodaan virheviesti, joka palautetaan lähettäjälle. [52] Viestin toimituksen raportointiin on kehitetty Delivery Status Notification (DSN)-laajennus, [53] mutta tämän täysipainoinen käyttö vaatii, että kaikkien viestien välittämiseen osallistuvien palvelinten täytyy toteuttaa tämä laajennus.

Riippumatta sähköpostin siirtoprotokollasta, kaikki sähköpostia vastaanottavat palvelimet toimittavat viestit käyttäjän omalle tilille palvelimella. Perinteisesti UNIX-käyttöjärjestelmissä sähköpostit säilytetään siihen varatussa mbox-tiedostossa (/var-/spool/mail -hakemistossa). Tämä kuitenkin vaatii että käyttäjä ottaa sähköpostien lukemiseksi pääteyhteyden UNIX-palvelimeen. Kehittyneemmät palvelimet pystyvät tallentamaan sähköpostit mbox-tiedostojen lisäksi Maildir-kansioihin tai suorittamaan viestien jakamista eri kansioihin. Erikoistuneemmat sähköpostipalvelimet säilyttävät käyttäjien viestit omassa tietokannassaan. Tavasta riippumatta sähköpostien säilöpaikkaa kutsutaan postilaatikoksi.

POP (eli POP3, Post Office Protocol version 3) [54] on yksinkertainen protokolla, joka toimittaa sähköpostiviestit palvelimen postilaatikosta käyttäjän omalla tietokoneellaan ajamaan asiakasohjelmaan. POP-protokollassa asiakasohjelma ottaa yhteyden POP-palvelimeen, joka on vastaanottanut ja säilönyt sähköpostiviestit käyttäjän tilille, ja vastaanottaa eräajona saapuneet viestit. Viestit voidaan samalla poistaa palvelimelta. POP on yksinkertainen viaksi asti; viestit ovat palvelimella vain jonona, jolloin on hankalaa pystyä tunnistamaan mitkä viestit ovat uusia. Useamman käyttäjän tapauksessa tämä menee vielä hankalammaksi. Yleiset käytännöt on joko siirtää viestit aina pois palvelimelta, jolloin kaikki palvelimella olevat ovat aina uusia, tai jättää kaikki viestit palvelimelle, jolloin uudet voidaan päätellä muistamalla edellisen tarkistuskerran viestien määrä.

IMAP (eli IMAP4, Internet Mailbox Access Protocol version 4) [55] on monipuolinen protokolla postilaatikon etäkäyttöön. IMAP-protokolla tukee POP-tyylistä erillistoimintaa, jossa postit siirretään käyttäjän koneelle sekä jatkuvaan yhteyteen perustuvaa suorakäyttöistä toimintaa. Suorakäyttöisessä toiminnassa on etuna, että käyttäjä saa välittömästi tietoa muutoksista postilaatikon tilassa. IMAP-protokolla pystyy myös käsittelemään useita postilaatikoita, jakamaan uutisryhmiä ja muuta sisältöä, tallentamaan viesteille status-arvoja (ns. lippuja, esimerkiksi “luettu”), suorittamaan viesteille osittaista noutoa ja suorittamaan hakuja palvelimen päässä. IMAP lisää postilaatikkoon synkronoinnin tarvitsemia konsepteja, kuten postilaati-

kon nimen lisäksi yksi yksilöivän identiteetin (“UIDvalidity”) ja viesteille vastaavan identiteetin (“UID”).

IMAP-protokollan monipuolisuus on myös sen suurin ongelma. Protokolla on kehittynyt inkrementaalisesti, pysyen jatkuvasti enimmäkseen taaksepäin yhteensopivana, ja saavuttanut muodon, jossa sitä on haastavaa toteuttaa täysin standardin mukaisesti. IMAP-protokolla sisällyttää myös IMAP-palvelimeen monia pakollisia ominaisuuksia, joiden toteuttaminen on monimutkaista. [56]

IMAP-protokollalle on tehty valinnaisia laajennuksia moninaisiin tarpeisiin. Laajennukset on sisällytetty protokollaan siten, että asiakasohjelma voi kysyä palvelimelta sen tukemia laajennuksia ja jos näitä on, niin asiakasohjelma voi käyttää niiden sallimaa laajempaa syntaksia. Olennaisin laajennuksista on ns. LEMONADE (License to Enhance Message Oriented Network Access for Diverse Environments) -laajennusperhe, jotka muodostavat profiilin, joka keventää mobiili-toteutusten vaatimuksia. Reaaliaikaisen viestinnän kannalta oleellisin on ns. “IDLE”-laajennus, joka mahdollistaa viestin asynkroniset saapumisilmoitukset postilaatikon pollaamisen sijaan. [57]

2.4.3 Uutisryhmät

Uutisryhmät ovat sähköpostin variantti, joka keskittyy laajojen keskusteluryhmien toteuttamiseen. Isojen keskusteluryhmien tapauksessa viestien tilaajien seuraaminen ja viestien levittäminen muodostuvat merkittäviksi ongelmiksi. Uutisryhmäohjelmistot kiertävät ongelmat muodostamalla oman levityshierarkiansa, joka perustuu nimettyihin uutisryhmiin. Poiketen sähköpostista ja postituslistoista, uutisryhmät jaetaan liki poikkeuksetta eräajoina määriteltyä jakeluhierarkiaa pitkin ja viestejä ei toimiteta käyttäjille, vaan ne säilötään uutisryhmäpalvelimissa.

Uutisryhmät kehittyivät sähköpostin rinnalla UUCP-protokollalla siirrettävinä ilmoitusviesteinä. Internetin ilmestyessä uutisryhmien välittämiseen kehittyi Network News Transfer Protocol (NNTP) -protokolla, joka korvasi nopeasti edeltäjänsä. NNTP-protokolla mahdollisti eräajojen ajamisen yksittäisten viestien tasolla ja helpotti viestienvälityshierarkian ylläpitämistä. NNTP-protokolla toimii sekä uutisryhmäpalvelimien että asiakasohjelmien välisenä viestiprotokollana. NNTP-protokollan viimeisin versio on määritelty IETF standardissa RFC 3977 [58].

Uutisryhmien viestit tallennetaan samassa muodossa kuin sähköpostiviestit, vain osoitetyypit ovat erilaisia. Uutisryhmäviesteissä kohdeosoitteina on uutisryhmien nimiä. IMAP-protokollaa voidaan käyttää myös lukemaan uutisryhmäviestejä.

Uutisryhmien suosio ja merkitys on viime vuosina tippunut Internet foorumei-

den korvattua ne pääosin. Syynä epäillään uutisryhmien avointen käyttöehtojen aiheuttamaa sisällön laadun romahtamista, roskapostin vallatessa alaa. Monet suuret Internet-palveluntarjoajat ovat luopuneet uutisryhmien välittämisestä. [59]

2.4.4 Pikaviestintä

Pikaviestintä on välitöntä kahden keskistä tai yhdeltä useammalle lyhyiden viestien vaihtamista. Verrattuna sähköpostiin pikaviestintäprotokollat ovat yksinkertaisempia, mutta vaativat kaikkien palvelunkäyttäjien olevan jatkuvasti yhteydessä välityspalvelimeen tai toisiinsa.

Ensimmäiset pikaviestintäsovellukset toimivat viestimällä saman monenkäyttäjän tietokoneen yhtäaikaiselta käyttäjältä toiselle. Tällainen oli esimerkiksi "talk"-sovellus, jolla UNIX-koneelle sisäänkirjautuneet käyttäjät pystyivät keskustelemaan keskenään. Tästä kehittyivät erillisiä palvelimia käyttävät pikaviestintäsovellukset, joista yksi vanhimpia ja yleisimpiä on Internet Relay Chat (IRC) [60].

IRC-protokolla on tekstipohjainen ja yksinkertainen. Yleinen mekanismi IRC-verkoissa on muodostaa palvelimista puurakenne ja välittää ryhmämuutokset kaikille palvelimille mutta viestit vain niille palvelimille, joilla on viestin kohderyhmän käyttäjiä. IRC-protokollaan on usein tehty sovelluskohtaisia muutoksia, jotka eivät ole yleistymisestä huolimatta päässeet standardin tasolle. IRC-protokollan puute on yhteisten sopimusten puuttuminen monista käytännöistä, esimerkiksi merkistön valinnassa. [61]

IRC-protokollan lisäksi pikaviestintään on olemassa lukuisia suljettuja protokollia. Esimerkkinä suljetuista pikaviestintäprotokollista voidaan mainita suosittu Microsoft Messenger. Suljetuilla protokollilla on yleensä vain yhden sovellusvalmistajan käyttäjiä, mikä rajoittaa palvelun saatavuutta. Osa suljetuista protokollista on takaisinmallinnettu (engl. reverse engineering), jolloin niistä on voitu tehdä muita toteutuksia.

Toinen avoin pikaviestintä protokolla on IETF-organisaation kehittämä Extensible Messaging and Presense Protocol (XMPP) [62]. XMPP-protokolla perustuu Extensible Markup Language (XML) -kieleen. Käyttäjät viestivät yhä vain palvelinten kanssa ja palvelimet kommunikoivat keskenään ilman rakennettua hierarkiaa. Jokainen käyttäjätunnus on sidottu palvelimeen, jolloin palvelimet tämän pohjalta tietävät mihin palvelimeen ottaa yhteyttä. XML-pohjaisuuden takia XMPP-protokolla on laajennettava, mutta viestinnässä XML-syntaksi muodostaa huomattavan osan kokonaisliikennemäärästä. [63] XMPP-protokolla tukee yhden käyttäjän liittymistä verkkoon useammilla eri tavoilla yhtäaikaisesti, sekä viestien jättämistä käyt-

täjän ollessa irti verkosta ja yleensäkin käyttäjän liittymistilan seuraamista.

2.4.5 Internet-puhelut

Internet-puhelut ovat kaksisuuntaisia ääni- tai videoviestintämenetelmiä. Yleinen kattotermi puhepohjaisille Internet-puheluille on Voice over IP (VoIP). Internet-puheluiden toteutukset yleensä keskittyvät kahteen osaan: yhteyden neuvotteluun ja viestintädatan siirtoon. Yleisimmät avoimet protokollat näiden toteuttamiseen ovat IETF-standardi Session Initiation Protocol (SIP) [64] ja ITU-standardi H.323 [65] yhteyksien neuvotteluun, sekä Real-time Transport Protocol (RTP) [66] viestidatan siirtoon. Esimerkkinä suljetuista Internet-puhelujärjestelmistä voidaan nimetä suosittu Skype.

Internet-puhelut voidaan neuvotella suoraan käyttäjältä käyttäjälle, mutta yleensä käytetään keskitettyjä välityspalvelimia, jolloin voidaan käyttää yksinkertaisempia yhteysnimiä ja käsitellä paremmin tilanteet, jossa yhteyden kohde ei ole saatavilla. Neuvottelun jälkeen pyritään muodostamaan suora yhteys käyttäjältä toiselle käyttäen vähälatenssista siirtoprotokollaa. Jos suora yhteys ei ole mahdollinen, voidaan käyttää välityspalvelimia, toteutuksesta riippuen.

Puhelinverkkoja vastaten Internet-puheluiden tärkein vaatimus on käyttäjältä käyttäjälle vähälatenssinen verkkoyhteys. Yhteyksien neuvottelu vaatii verkkoyhteyden käyttäjälle tai käyttäjän välityspalvelimelle. Yleinen ongelma Internet-puhelujärjestelmissä on pääsyltään rajatut verkot. Esimerkiksi palomuurilla suojattuun tai osoitteenmuunnostekniikkaa käyttävään verkkoon voi olla mahdoton neuvotella suoraa verkkoyhteyttä, jolloin on pakko käyttää välityspalvelinta myös viestintädatan välittämiseen.

2.4.6 Tiedostonsiirto

Tiedostojen siirto Internetissä muodostaa suurimman osan Internetin liikenteestä. Tiedostojen siirtämisen yleisin menetelmä on Hypertext Transport Protocol (HTTP) [67], jota käytetään ns. hyperteksti-muotoisten dokumenttien siirtämiseen. Hypertekstidokumentit voivat sisältää viitteitä muihin dokumentteihin. HTTP-protokolla on suositun World Wide Web (WWW) -palvelun perusta. Toinen yleinen tiedostonsiirtoprotokolla on File Transfer Protocol (FTP) [68]. Käytännössä Internet-käytössä HTTP-protokolla on korvannut kaikki kilpailijansa.

Tiedostonsiirron toimintamallissa asiakas lähettää siirtopyynnön palvelimelle, joka palauttaa vastauksena pyydetyn dokumentin tai virheilmoituksen, joka voi sisältää jatkotoimintaohjeita, esimerkiksi uudelleenohjauksen. Tiedostojen siirto tarvit-

see siis aina yhteyden palvelimelle, joka voi halutun dokumentin palauttaa. WWW-palvelussa on järjestelmän saatavuutta parannettu käyttämällä välityspalvelimia (engl. proxy server), jotka voivat vähentää usein pyydettyjen dokumenttien siirtämisen aiheuttamaa verkkokuormaa, ja monistamalla palvelimia erilaisin menetelmin. Nämä menetelmät kuitenkin pyrkivät vain takaamaan, että palvelu on jatkuvasti saatavilla, eivätkä siten auta tilanteessa, jossa päästä päähän verkkoyhteyttä asiakkaan ja palvelinten välillä ei ole. WWW-palvelun käytön parantamiseksi verkkoyhteyden ollessa huono on kehitetty erikoistuneita välityspalvelimia, mutta ratkaisut ovat hyvin erikoistuneita eivät takaa erinaisten WWW-palvelujen toimimista**??**.

Tiedostonsiirtomenetelmien päälle on WWW-palvelussa kehitetty korkeamman tason palveluita. Tämä on tehty käyttämällä WWW-selainsovellusta tulkitsemaan lähetetty dokumentti sovelluksena, joka viestii WWW-palvelimen kanssa. Yleisimmät tämän mallin sovellukset on toteutettu Java, Flash tai Javascript -ohjelmointikielillä. Muodostettu ohjelma sisällytetään, suoraan tai viitteenä, siirrettyyn dokumenttiin. Tällaisten ohjelmien heikkoutena on, että ne yleensä vaativat nopean ja luotettavan Internet-liittymän.

Tiedostojen siirtoon on myös kehitetty menetelmiä, jotka eivät tarvi keskitettyjä palvelimia. Näitä kutsutaan ns. vertaisverkko-järjestelmiksi (engl. peer-to-peer, P2P). Vertaisverkoissa jokainen vertaisverkon jäsen osallistuu tiedon siirtämiseen muodostamalla itselleen kopion siirrettävästä tiedostosta. Vertaisverkkomenetelmät eivät rajoitu pelkästään tiedostojen siirtämiseen, vaan niitä on käytetty myös pikaviestinnän ja Internet-puheluiden toteuttamisessa. Esimerkkeinä vertaisverkoista voidaan mainita ensimmäisenä suuren suosion saanut vertaisverkko Napster [69] ja anonyymiin julkaisuun keskittyvä Freenet [70].

Yleisesti vertaisverkkojärjestelmät ovat erityisvälitysverkkojen kaltaisia ratkaisuja, mutta lähtökohdiltaan ovat Internet-ympäristöön tarkoitettuja. Poiketen erityisvälitysverkoista, vertaisverkkosovelluksissa käyttäjä ei pääsääntöisesti osallistu niiden tiedostojen siirtoon, joita eivät ole itse vastaanottamassa.

2.4.7 Muita toteutuksia

ITU-T X.400

Sähköiseen viestintään tietoverkkojen yli on myös muita sähköpostin kanssa kilpailevia ratkaisuja. Viestintäjärjestelmien arkkityyppi on Internation Telecommunication Union -organisaation määrittelemä ITU-T X.400 Message Handling System (MHS). X.400 kehitettiin rinnan teollisuusstandardina kasvaneen sähköpostin kanssa, mutta X.400 oli alunperin kehitetty käyttämään OSI-verkkoa, joka oli harvemmin toteu-

tettu. X.400 standardille kehitettiin laajennus joka mahdollisti TCP/IP-protokollan käyttämisen [71], mutta tässä vaiheessa sähköposti oli jo voittanut enimmäkseen TCP/IP-pohjaiset markkinat.

X.400 standardista kehittyneitä järjestelmiä on yhä käytössä omissa markkinaraoissaan. Armeijan viestinnässä on omat MMHS (Military Message Handling System) -järjestelmät, joita ovat mm. Pohjois-Atlantin liiton (engl. the North Atlantic Treaty Organisation, lyh. NATO) standardoima STANAG 4406 [72] ja CCEB (Combined Communications Electronics Board) -järjestön standardi ACP 123 [73]. Ilmailualan viestinnässä on Kansainvälisen siviili-ilmailujärjestön (engl. International Civil Aviation Organization, lyh. ICAO) määrittelemä AMHS (Aeronautical Message Handling System) -järjestelmä. Organisaatioden väliseen tiedonsiirtoon on standardoitu Organisaatioiden välinen tiedonsiirto (OVT, engl. Electronic Data Interchange, lyh. EDI) -tekniikka.

Sähköposti on omaksunut monia ominaisuuksia X.400-järjestelmältä. X.400 on kattavampi ja tarkemmin määritelty standardi. Sähköposti kuitenkin voitti markkinoilla ja siten on saavuttanut aseman, jossa käyttäjät olettavat käyttämiltään viestijärjestelmiltä vähintään sähköpostin tasoista ja sähköpostin kanssa yhteensopivaa palvelua. X.400 on myös standardina hyvin suuri, joten useimmat sen toteuttajista ovat ns. leikanneet kulmia ja tehneet osittaisia toteutuksia, tai muokanneet järjestelmää epäyhteensopivilla tavoilla omaan kohdesovellukseensa.

Microsoft Exchange

Microsoft Exchange on kehitetty suljettuna sovelluksena käyttäen suljettuja muunnoksia julkisista standardeista. Microsoft Exchange -protokolla oli pitkään varjeltu yrityssalaisuus, mutta Euroopan Unionin oikeus pakotti Microsoftin julkaisemaan palvelinsovellusrajapintansa vuonna 2007. [74]

Microsoft Exchange alkoi vuonna 1996 X.400-pohjaisena viestijärjestelmänä, jossa oli tuki X.500-hakemistopalveluille. Exchange-palveluun kehitettiin seuraavaan versioon tuki SMTP-protokollalle, ja Microsoft Office -toimisto-ohjelmistopakettiin lisättiin tuki Exchange-palveluille. Kehityksen myötä Exchange-palveluun on lisätty täysi SMTP- ja IMAP-tuki, jaettujen kalenterien ja tehtävälistojen käyttö ja tiedostojen jakaminen.

Microsoft Exchange käyttää sisäisessä viestinnässään muunnosta Distributed Computing Environment / Remote Procedure Call (DCE/RPC) -protokollasta [75], nimeltä Microsoft Remote Procedure Call (MSRPC) [76], jonka päälle on rakennettu eri osajärjestelmien kommunikointiprotokollat. Samaa MSRPC-protokollaa käyte-

tään mm. Windows Domain -palvelinten asiakas/palvelin-viestinnässä. Exchange-järjestelmässä on 8 ydinprotokollaa, jotka käsittelevät kukin eri tietotyyppejä: viesti/liite-, kansio-, tietorakenne-, tiedonsiirto-, ominaisuus/siirto-, säilöntä-, taulukko- ja ilmoitus-protokollat. [77] Protokollien viestirakenne on kuvattu variaatiolla Interface Description Language (IDL) -kielestä [75], nimeltä Microsoft Interface Description Language (MIDL) [76]. Exchange tukee myös ns. standardipohjaisia (SMTP, IMAP, LDAP, jne) protokollia, mutta on lisännyt toteutuksiin omia laajennuksiaan.

Asiakassovellusten toteuttamisen yksinkertaistamiseksi Microsoft Windows käyttöjärjestelmän mukana jaetaan Simple Messaging Application Program Interface (Simple MAPI) -kirjasto, jota mm. Microsoft Outlook Express käyttää. Microsoft Office -ohjelmiston mukana tulee Microsoft Outlook -ohjelmisto ja sen käyttämä täysikokoinen Extended MAPI -kirjasto. [78]

Kokonaisuutena Microsoft Exchange -järjestelmä on hyvin monimutkainen. Microsoftin julkaisema dokumentaatio on yli 3000 sivua pitkä, ja käytännön toteutukset sisältävät monia, jopa ohjelmistoversiokohtaisia eroja, joille on päätynyt standardiksi asti määritellyt kiertotavat.

2.5 Yhteenveto

Tietoverkkojen siirtomenetelmien ominaisuudet vaihtelevat suuresti. Osa on kykeneviä sellaisenaan toimimaan Internetin osina, osa pystyy tähän rajoituksin, osa ei lainkaan. Erilaisten siirtomenetelemien ominaisuuksia on kerätty taulukkoon 2.3. Taulukon luvut ovat suuntaa-antavia, ja kannattaa tiedostaa että huippukantamalla järjestelmien suorituskyky on huonoimmillaan. Useimpien yhteystyyppien suorituskyky riippuu käyttötilanteesta ja teoreettisia maksimiarvoja ei käytännössä saavuteta. Riippumatta ominaisuuksistaan, kaikki luetellut siirtomenetelmät pystyvät välittämään viestejä.

Haasteelliset verkot alkavat siitä mihin taulukossa esitellyt ratkaisut loppuvat. Siinä missä Internetiin liitettävät verkot muodostavat yhden kokonaisuuden, muut verkot ovat olleet vain yksittäisiä saarekkeita. Haasteellisten verkkojen sovellukset pystyvät viestimään heterogeenisen, epäluotettavan ja hitaan verkon ylitse.

Viestinnän sovellukset ovat kehittyneet sekä lähtien tele- ja joukkoviestintäpalveluista, että lähtien Internetistä. Taulukossa 2.4 on kerättynä oleellisimpia viestintäsovelluksia, niiden käyttämiä tekniikoita ja protokollia. Internetissä käytetään lisäksi lukemattomia määriä sovelluksia ja toteutuksia, joita on mahdoton sisällyttää tähän. Voidaan kuitenkin havaita, että osaan sovelluksista ei ole Internet-pohjaisia ratkaisuja, ja vastaavasti osaa ei ole toteutettu muuten kuin Internet-palveluna. Ylei-

Taulukko 2.3: Siirtomenetelmien ominaisuudet.

Tekniikka	Huippukantama	Huippukapasiteetti	IP
Televerkko STM-64	Kaapeloitu	9.6 Gbit/s	Ei / PPP
Soittoyhteys V.92	Kaapeloitu	Jopa 56 kbit/s	Ei / PPP
ADSL2+	Kaapeloitu, 7 km keskukseen	Jopa 24 Mbit/s	Kyllä
Ethernet 1000Base-T	Kaapeloitu, 100 m kupari	1 Gbit/s	Kyllä
WLAN 802.11g	Langaton, 140 m ulkoilmassa	Jopa 54 Mbit/s	Kyllä
Bluetooth 2.1+EDR Class 2	Langaton, jopa 10 m	Jopa 2.1 Mbit/s	Ei / PPP
ZigBee	Langaton, jopa 100 m	250 kbit/s	Ei/Kyllä
IrDA (FIR)	Langaton, 1 m	4 Mbit/s	Ei / PPP
GSM (GPRS 2G)	Langaton, 35 km tukiasemasta	Jopa 80 kbit/s	Kyllä
GSM (EDGE 2G)	Langaton, 35 km tukiasemasta	Jopa 236.8 kbit/s	Kyllä
UMTS (HSDPA 3G)	Langaton, 8 km tukiasemasta	Jopa 14 Mbit/s	Kyllä
Iridium	Langaton, 781 km kiertorata	Jopa 3.8 kbit/s	“Kyllä”
WiMAX	Langaton, 50 km tukiasemasta	Jopa 70 Mbit/s	Kyllä
Flash-OFDM	Langaton, 25 km tukiasemasta	Jopa 5.3 Mbit/s	Kyllä
TETRA	Langaton, 58 km tukiasemasta	Jopa 14 kbit/s	Kyllä
PSK31	Langaton, 300 km (riippuu)	31 bit/s	Ei
PACTOR I	Langaton, 20 km (riippuu)	160 bit/s	Ei
DVB-T	Langaton, 100 km lähettimestä	22 Mbit/s	Ei
GPS	Langaton, 20200 km kiertorata	50 bit/s	Ei

simmät syyt sovellusten rajoittumiseen jompaan kumpaan ryhmään on sovelluksen tekniset vaatimukset, kustannukset ja historia.

Taulukko 2.4: Viestinnän sovellustyyppejä.

Sovellus	Tekniikka	Internet
Tekstiviesti	GSM, UMTS, TETRA	-
Hätätiedotus	Radio (RDS), DVB, TETRA, “GSM”	-
Puheyhteys	GSM, UMTS, TETRA, Internet (>64 kbit/s)	SIP, RTP, ..
Äänilähetys	Radio, DVB, Internet (>64 kbit/s)	HTTP, RTP, ..
Videolähetys	DVB, Internet (>1 Mbit/s)	HTTP, RTP, ..
Pikaviestintä	Internet	IRC, XMPP, ..
Sähköposti	Internet	SMTP, X.400, ..

Internetin viestintäprotokollat kehittyvät jatkuvasti. Sähköposti, uutisryhmät ja pikaviestintä ovat vain pieni osa Internetissä käytetyistä viestintämenetelmistä. Esitellyillä protokollilla on vahva standardiasema, lukemattomia käyttäjiä ja lukuisia käyttökohteita.

Luku 3

Päämäärät ja ratkaisumalli

Tässä luvussa käsitellään työn päämäärät täyttävän ratkaisumallin valintaa, sekä valitun ratkaisumallin toteuttamista ja analysointia. Toteuttamisen taustoista käsitellään ohjelmointiympäristöä, protokollien toteutustapoja ja suunnitteluvalintoja, korostaen valittuja muutoksia standardien konsepteista. Ohjelmistokomponentin toteuttaminen perustuu oleellisesti valittuun ohjelmistoympäristöön ja toteutustapaan. Lopuksi esitellään ohjelmiston laaduntarkkailua ja ohjelmiston testaussuunnitelma. Ohjelmistotestaamisen tarkoitus on varmistaa, että ohjelmiston toteutus on hyvänlaatuinen ja yhteensopiva muiden sovellusten kanssa.

3.1 Sovellusrajapinta

Työn päämäärä on tuottaa ohjelmistokomponentti, joka yhdistää perinteisen Internet-tietoverkon mihin vain erityisvälitysverkkoon. Tämä pyritään toteuttamaan valitsemalla parhaiten sopiva sovellusrajapinta. Rajapinnan tulee olla sekä yleinen ja avoin, jotta sillä olisi käytännön sovelluksia ja mahdollista tehdä riippumattomia toteutuksia, että kykenevä siirtämään kaikenlaista liikennettä millaiseen ympäristöön vaan. Rajapinnan vaatimuksia tarkastellaan erityisvälitysverkkojen ja sovellusten kannalta.

Erityisvälitysverkkojen käyttäessä, riippuen toteutuksesta ja tilanteesta, on mahdoton taata yhteyksille suorituskykyrajoja, kuten latenssia, siirtonopeutta tai saatavuutta. Rajapinnan tulee abstraktoida varsinainen siirtotapa siten että viestintää tekevien sovellusten ei tarvitse ymmärtää erityisvälitysverkkojen mekanismeja.

Sovellukset asettavat kukin omia rajoituksiaan käytettävälle rajapinnalle. Erityisvälitysverkkojen ominaisuudet asettavat omat rajat siirtomenetelmien mahdollisuuksille, mitä rajapinta ei välttämättä pysty peittämään. Sovellusten ominaisuuksia

ja niiden toteuttamisen vaatimuksia on vertailtu taulukossa 3.1. Sovellusten ominaisuudet on jaettu niiden käyttämään siirtomuotoon, yhteysmuotoon, osoitustapaan ja viestirakenteeseen.

Taulukko 3.1: Viestintäsovellusten ominaisuuksia.

Sovellus	Siirto	Yhteys	Osoitus	Viestirakenne	Muuta
Sähköposti	Eräajo	Asiakas-palvelin	Kohde, lista	MIME	
Uutisryhmät	Eräajo	Asiakas-palvelin	Ryhmä	MIME	
Pikaviestintä	Eräajo, virta	Asiakas-palvelin, asiakas-asiakas	Kohde, ryhmä	Teksti, XML	Läsnäolo Läsnäolo
Internet-puhelut	Virta	Asiakas-asiakas	Kohde, lista	Mediavirta	Reaaliaikaisuus
Tiedostonsiirto	Eräajo, virta	Asiakas-palvelin	Kohde	Raaka (MIME)	Vasteaika-vaatimus

Siirtomuotoina ovat joko tapa lähettää viestit yksittäisinä kokonaisuuksina eräajoissa tai muodostaa viestin siirrosta jatkuva yhteys. Eräajoa siirtomuotonaan käyttävät protokollat pystyvät erityisvälitysverkoissa toimimaan pääosin normaalisti. Virta-muotoiset protokollat voidaan toteuttaa jakamalla virta osiin ja siirtämällä osat eräajoina, kuten IP-protokolla tekisi normaalissa Internet-käytössä.

Yhteysmuotoina on joko asiakas-palvelin -tyylinen malli tai suora asiakkaalta toiselle oleva malli. Erityisvälitysverkkojen kannalta asiakas-asiakas-yhteydet ovat hankalia. Asiakas-palvelin-mallissa voidaan erityisvälitysverkko peittää toteuttamalla rajapintaan palvelin-ohjelmisto. Tämä tosin on mahdollista vain jos protokollassa ei ole suoranaisia vasteaikavaatimuksia toiminnoille.

Viestien osoituksessa käytetään kohdetta, listaa kohteista tai ryhmäkohdetta. Kohde on, riippuen sovelluksesta, henkilö tai joku muu tunniste. Kohteilla on alkuperäissovelluksessaan tietty merkitys, mutta nimiä voi yhteensovittaessa tulkita laajemmin, protokollan syntaksin rajoissa. Esimerkiksi sähköpostijärjestelmissä yleisesti on toteutettu ns. postituslista-palveluja, joissa yhden kohdeosoitteen viestit välitetään kaikille postituslistan jäsenille. Erilaisten sovellusten yhteensovittamisessa on hyödyllistä jos osoitus on mahdollisimman monipuolinen tai syntaksiltaan avoin.

Protokollat määrittelevät käyttämänsä viestirakenteet. Tämä rajoittaa viestijärjestelmän lävitse siirrettävien viestien rakennetta. Erilaisten sovellusten kannalta on hyödyllistä että viestirakenteet ovat yhteensopivia. Osa rakenteista sallivat moniosaisia viestejä ja sisältötyypin sisällyttämisen viestiin.

Protokollien tunnettujen ominaisuuksien pohjalta voidaan tarkastella eri sovellusten toteuttamista toisten sovellusten rajapinnoin. Työssä tutkituista protokollista suosituimmat, ja niiden kyky tukea muita sovelluksia, on esitetty taulukossa 3.2. Taulukossa on myös laskettu protokollien kattavuuden vertailu, siten että kunkin

sovelluksen täydestä kattavuudesta on annettu kaksi pistettä ja osittaisesta yksi.

Taulukko 3.2: Sovellusten rajapintojen yhteensopivuus.

	Toteuttava rajapinta				
Sovellus	SMTP / IMAP	NNTP	XMPP	SIP / RTP	HTTP
Sähköposti	Täysin	Osittain	Osittain	Osittain	Osittain
Uutisryhmät	Täysin	Täysin	Osittain	Osittain	Osittain
Pikaviestintä	Osittain	Ei	Täysin	Täysin	Ei
Internet-puhelut	Ei	Ei	Osittain	Täysin	Ei
Tiedostonsiirto	Täysin	Osittain	Osittain	Ei	Täysin
	7	4	6	6	4

Uutisryhmien toteuttaminen SMTP/-IMAP-protokollilla on mahdollista, koska IMAP-protokolla pystyy käsittelemään uutisryhmien tilaamisen ja uutisryhmäviestejä voidaan lähettää SMTP-protokollalla. Sähköposti tavallaan voi toimia pikaviestintänä, koska IMAP-protokollassa on IDLE-laajennus, jolla on mahdollista saada reaaliaikaisia ilmoituksia. Läsnäolon käsitettä tämä ei pysty esittämään. SMTP-protokollaa voidaan myös käyttää tiedostojen siirtämiseen, ja IMAP-protokollaa tiedostojen siirtoon ja jakeluun.

Uutisryhmäprotokollat toisaalta eivät sovellu sähköpostiin yhtä hyvin. Periaatteessa yksityistä postia voi mallintaa henkilökohtaisilla ryhmillä. Käytännössä tämän tyylisiä ratkaisuja ei ole tutkittu, koska sähköposti on toiminut rinnalla uutisryhmäpalvelujen kanssa.

HTTP-protokolla itsessään käyttää hyvin yksinkertaista toimintamallia, joka ei sovellu interaktiivisiin yhteyksiin. Jokainen HTTP-pyyntö vaatii vastauksen tietyssä ajassa, mikä tekee protokollan sopimattomaksi rajapinnaksi erityisvälitysverkoille. HTTP-protokollan päälle, WWW-palvelun kontekstissa, on tehty lukuisia erilaisten sovellusten toteutuksia. Nämä kaikki kärsivät siitä protokollan käytännöstä jossa esitettävä data siirretään yksittäisinä sivuina. Muutokset datassa vaativat sivun uudelleenlataamista. Tätä usein kierretään lisäämällä sivulle aliohjelmia, jotka käsittelevät ja läpinäkyvästi uudelleenhakevat sivun sisältöä. Käytännössä tällöin on HTTP-protokollan päälle rakennettu oma protokolla. Tällaista “protokollaa protokollan päällä” voidaan käyttää erityisvälitysverkkojen kanssa, mutta se on turhan monimutkainen ja vaatii käyttäjältä sivujen lukemiseen tarvittavan WWW-selaimen, mikä rajaa pois monia sulautettuja sovelluksia.

XMPP-protokolla soveltuu periaatteessa hyvin useimpiin sovelluksiin, mutta käytännössä XMPP-protokollan toteuttavat ohjelmistot keskittyvät vain pikaviestintään. Joustavan XML-rakenteensa avulla XMPP-protokollalla voidaan esittää moninaisia viestimuotoja ja jakelusääntöjä. XMPP-protokollaan on myös laajennuksia tiedostojen siirtoon ja Internet-puheluiden järjestämiseen. Käytännössä nämä menetelmät käyttävät XMPP-protokollasta riippumattomia siirtomenetelmiä, jotka neuvotellaan XMPP-yhteyden kautta, eli XMPP-protokolla itsessään ei näitä sovelluksia kata.

Internet-puheluprotokollat ovat sovellukseensa erikoistuneet, mutta ovat myös laajennettu hoitamaan useimpia muitakin sovelluksia. SIP-protokollaa voidaan käyttää pikaviestintämenetelmänä, ja se pystyy välittämään läsnäolon käsitteen. SIP-protokollalle on myös laajenteita, jotka toteuttavat sähköposti- ja uutisryhmä-tyylisiä sovelluksia. Nämä laajennukset toimivat laajentamalla SIP-protokollan pikaviestintämekanismia ja ovat tarkoitettu vain päästä-päähän tyyliseen viestintään. Varsinainen mediasiirtoprotokolla, RTP, odottaa aina saavansa vähälatenssisen päästä-päähän yhteyden, joten se ei sovellu erityisvälitysverkkoihin sellaisenaan.

Tutkimuksen pohjalta SMTP/IMAP-rajapinta on joustavin rajapintavaihtoehto. Käytännössä kaikkien rajapinnan sovellusten tulee siis sovittaa toimintamallinsa käyttämään sähköpostia viestinnässä, sovituin laajennuksin. Esimerkiksi pikaviestintä-tyylisen reaaliaikaisen viestinnän mahdollistamiseksi, sähköpostiasiakasohjelmistojen tulee käyttää IMAP IDLE-laajennusta. Sovellusten viestit tulee taltioida käyttäen sähköpostin RFC 5322 -standardin mukaista viestirakennetta. Rakenne on joustava ja sallii moniosaiset viestit ja erilaiset viestityypit.

3.2 Ohjelmointiympäristö

Ohjelmointiympäristöllä tarkoitetaan rajapintoja, joita käytetään ohjelmistoa kehittäessä. Nämä rajapinnat koostuvat ohjelmointikielestä ja ohjelmistokirjastoista, sekä ajoympäristöstä, joka koostuu laitteistosta, käyttöjärjestelmästä ja käyttöjärjestelmän ohjelmointirajapinnoista. Ohjelmointiympäristö yleensä valitaan ohjelmoijan taitojen, kehitettävän ohjelmiston tarpeiden ja ohjelmistoa suorittavan ajoympäristön rajoitusten mukaan.

Siirrettävyys

Siirrettävyys tarkoittaa ohjelmiston kykyä toimia eri ajoympäristöissä, eri käyttöjärjestelmien alaisuudessa ja eri prosessoriarkkitehtuureilla. Siirrettävyyttä yleises-

ti pidetään hyvänä ominaisuutena ohjelmistolla, koska se mahdollistaa ohjelmiston käytön useammissa ympäristöissä, mikä mahdollistaa sekä markkinakohderyhmän laajentamisen että laitekustannusten vähentämisen.

Siirrettävyyttä on yleensä hankala mitata, se joko toimii tai ei toimi, mutta se tulee ottaa huomioon ohjelmistoa kehitettäessä. Siirrettävyyttä rajoittavat eri käyttöjärjestelmien tarjoamat rajapinnat. Standardoituja rajapintojakin toteutettaessa usein ilmenee toteutuskohtaisia eroja, joilla on käytännön vaikutuksia, jollei niitä ota huomioon. [79]

Standardien noudattamisen lisäksi siirrettävyyteen voi pyrkiä toteuttamalla ohjelmisto modulaarisesti, jotta ongelmalliset osat voidaan eristää moduuliin, josta tehdään käyttöjärjestelmäkohtaiset toteutukset. Siirrettävyys pitää siis ottaa huomioon ohjelmiston suunnitteluvaiheessa ja pitää kokoajan mielessä.

Mikäli mikään näistä toimintatavoista ei ole mahdollista, voidaan ohjelmisto toteuttaa uudelleen eri ohjelmointiympäristöön. Tämä on työläs tapa kiertää siirrettävyyden puutetta, koska ohjelmisto joudutaan käytännössä kirjoittamaan uudestaan. Uudelleentoteutuksen yhteydessä on myös mahdollista refaktoroida lähdekoodia, esimerkiksi vaihtaa se käyttämään siirrettävämpiä rajapintoja.

Rajapinnat

Yhteensopivan ja siirrettävän ohjelman kehittämisessä tärkeää on käyttää sovellusohjelmointirajapintoja (API, Application Programming Interface), jotka löytyvät mahdollisimman monesta ohjelmiston tarkoitetun kohderyhmän käyttöjärjestelmästä. Rajapinnat kulkevat käsi kädessä käyttöjärjestelmien kanssa, mutta rajapintoja on standardoitu siirrettävyyden mahdollistamiseksi.

Parhaiten tunnettu ja laajalti tuettu sovellusohjelmointirajapinta on vuonna 1988 julkaistu IEEE Std 1003.1-1988, joka yleisemmin tunnetaan nimellä POSIX (Portable Operating System Interface for uniX) [80]. Nimensä mukaisesti POSIX-standardin juuret ovat UNIX-käyttöjärjestelmissä, mutta rajapinnat ovat toistettavissa muillakin käyttöjärjestelmillä. Viimeisin julkaisu on IEEE 1003.1-2004 (POSIX), joka on yhdistänyt The Open Group konsortion Single Unix Specification -standardin (SUSv3) ja muita teollisuuden alan de facto -standardeja. Olennaisesti sama standardi on julkaistu myös ISO/IEC-järjestöjen toimesta nimellä ISO/IEC 9945:2003/Cor 1:2004.

POSIX on kokoelma standardeja ja dokumentteja, jotka määrittelevät ohjelmointiympäristön peruskäsitteet, järjestelmän ohjelmointirajapinnat, komentokuoren ja komentokuoren työkalut, sekä POSIX-standardin perusteet ja taustat.

Käytössä olevista käyttöjärjestelmistä kaikki UNIX-pohjaiset toteuttavat suurimman osan POSIX-rajapinnoista. [81] Muut käyttöjärjestelmät toteuttavat yleensä vähintään osan POSIX-rajapinnoista ohjelmistojen siirrettävyyden parantamiseksi. Vastaavasti osa POSIX-rajapinnat toteuttavista käyttöjärjestelmistä tarjoaa myös muita rajapintoja, jotka ohittavat POSIX-standardin määrittelemät ominaisuudet.

Monilla käyttöjärjestelmillä, erityisesti sulautetuilla, on oma natiivi ohjelmointirajapintansa. Natiivilla rajapinnalla voidaan saavuttaa johonkin erityissovellukseen huomattavasti parempi suorituskyky tai suoraviivaisempi toteutus, kuin käyttämällä standardoituja rajapintoja.

Muita merkittäviä POSIX-yhteensopimattomia rajapintoja ovat Microsoft Windows -käyttöjärjestelmäperheen WinAPI-rajapintakokoelma [82] ja Sun Microsystemsin Java-tuoteperhe [83]. WinAPI on Windows-käyttöjärjestelmien päärajapinta ja siten merkittävä Windows-käyttöjärjestelmän yleisyyden takia. WinAPI on Microsoft Corporation -yhtiön kehittämä ja julkaisema.

Java-tuoteperhe sisältää ohjelmistoja ja määritelmiä, jotka yhdessä tarjoavat järjestelmän kehittää ja ajaa Java-ohjelmointikielellä tehtyjä sovelluksia käyttöjärjestelmäriippumattomasti. Käytännössä Javan käyttöjärjestelmäriippumattomuus on saavutettu toteuttamalla Javan ajonaikainen ympäristö eri alustoille. Sun tarjoaa itse toteutukset Solaris, Windows ja Linux käyttöjärjestelmille, mutta Java on toteutettu myös monille muille.

Java-ympäristöstä on useampia versioita eri tasoisille ympäristöille; Java Card sulautetuille älykorttilaitteille, Java Micro Edition (ME) rajoitetuille laitteille (kuten matkapuhelimet), Java Standard Edition (SE) yleiskäyttöisille tietokoneille ja laitteille, Java Enterprise Edition (EE) yrityssovelluksille. Versioiden ominaisuudet vaihtelevat, mutta ohjelmointikieli pysyy samana ja tarjotut ominaisuudet on toteutettu yhteensopivasti eri versioiden välillä. Yksinkertaisempien versioiden ohjelmien tulisi toimia suoraan laajemmilla versioilla.

Käytännössä Java-ympäristön käyttäminen tarkoittaa että ohjelmisto on kehitettävä Java-ohjelmointikielellä ja käyttökohteeseen on asennettava ajonaikainen Java-ympäristö. Vaihtoehtoisesti Java-ohjelma voidaan kääntää laitteistokohtaiseksi Java-kääntäjällä, jolloin ohjelma on kuin normaali käännetty ohjelma ja toimii pelkästään tällä laitteistolla.

Siirrettävyystekniikat

Siirrettävyyteen on toteutettu myös erikoisratkaisuja, kuten siirrettävyyskirjastoja ja virtualisointijärjestelmiä, jotka mahdollistavat ohjelmien muuntamisen siirrettä-

viksi vaikka ohjelmistoa ei olisi niin alunperin suunniteltu eikä toteutettu.

Siirrettävyyskirjastot toteuttavat standardoidut rajapinnat käyttöjärjestelmiin, joista ne puuttuvat. Siirrettävyyskirjastoa käytetään ohjelman käännösvaiheessa tarjoamaan puuttuvat rajapinnat ja ohjelma sitten ajonaikaisesti käyttää kirjastoa toteuttamaan puuttuvat ominaisuudet. Verrattuna natiiviin toteutukseen, siirrettävyyskirjasto aiheuttaa suorituskyvyn heikkenemistä, jonka suuruus riippuu rajapinnan toteutuksen monimutkaisuudesta. Lisäksi siirrettävyyskirjasto on oma ohjelmistokomponentti, joka lisää järjestelmän monimutkaisuutta ja resurssien käyttöä.

Yleisin siirrettävyyskirjastotyyppi on UNIX/POSIX-yhteensopivuuden käyttöjärjestelmään tarjoava kirjasto. Tällaisia ovat esimerkiksi Cygwin-kirjasto (Windows-alustalle) [84], ixemul.library-kirjasto (AmigaOS-alustalle) ja Interix-alijärjestelmä (Windows-alustalle) [85].

Virtualisointijärjestelmät mahdollistavat yhden käyttöjärjestelmän suorittamisen toisen käyttöjärjestelmän alaisuudessa. Tällöin kaikki virtualisoidun käyttöjärjestelmän rajapinnat ovat käytettävissä ja ohjelmistoja voidaan suorittaa ilman mitään muutoksia eri järjestelmän alaisuudessa.

Kevyimpänä virtualisointimallina on sovellustason virtualisointi, jossa ohjelmiston käyttöjärjestelmärajapinta virtualisoidaan (QEMU User Mode Emulation [86], Wine [87], User-mode Linux [88]). Tällöin ohjelmisto suoritetaan muuntamattomana, mutta kaikki interaktio ohjelmiston ja käyttöjärjestelmän välillä on virtualisoitu. Tämä on resurssinkäytöltään kevyin malli, mutta johtuen toteutettavien rajapintojen määrästä myös monimutkaisin toteuttaa. Jos emuloitu käyttöjärjestelmärajapinta eroaa natiivista rajapinnasta, voi aiheutua yhteensopivuusongelmia.

Tyypillisin virtualisointimalli on käyttöjärjestelmätason virtualisointi, jossa käyttöjärjestelmän ja tietokonelaitteiston rajapinta virtualisoidaan (VMware [89], Virtualbox [90]). Tällöin käyttöjärjestelmä ja ohjelmisto suoritetaan virtuaalikoneessa muuntumattomia, mutta käyttöjärjestelmän interaktio laitteiston kanssa on virtualisoitu. Koska ohjelmiston suorittamista varten joudutaan suorittamaan myös erillinen kopio käyttöjärjelmästä, aiheuttaa tämän mallinen virtualisointi merkittävästi resurssikuluja ja joissain tapauksissa suorituksen hidastumista.

Sekä siirrettävyyskirjastot että virtualisointi yleensä nähdään välttävinä tapoina taata ohjelmiston siirrettävyys, koska kumpikin ratkaisumalli aiheuttaa sovelluskohteeseen tarpeetonta monimutkaisuutta ja kuluttaa enemmän resursseja.

Protokollien toteutus

Tietoliikenneohjelmistojen toteutuksessa on oleellista suunnitella ohjelmiston rakenne. Rakenteen oleellisia valintoja ovat millaista siirräntärajapintaa (engl. I/O, Input/Output) käyttää ja miten ohjelmisto hoitaa useamman asiakkaan yhtäaikaisen palvelemisen. [91]

Siirräntärajapinta tarkoittaa ohjelmointirajapintaa, jota käytetään tiedon siirtämiseen tai signalointiin ohjelmistokomponenttien välillä. Siirräntärajapinnat käyttöjärjestelmissä voidaan jakaa kolmeen luokkaan; synkronisiin, reaktiivisiin ja proaktiivisiin. Synkronisissa suorittava ohjelma jää odottamaan vastausta käyttöjärjestelmältä. Tämä on yksinkertaisin malli ja yleensä riittämätön tietoliikenneohjelmistojen toteuttamiseen, mutta mahdollinen monisäikeisessä toteutuksessa. Reaktiivissa siirräntärajapinnoissa käyttöjärjestelmä tarjoaa ohjelmistolle ilmoituksen, kun rajapintaa voidaan käyttää. Proaktiivissa siirräntärajapinnoissa käyttöjärjestelmä hyväksyy siirräntäpyynnön ja ilmoittaa pyynnön valmistumisesta. Reaktiivisia rajapintoja kutsutaan myös estymättömiksi (engl. non-blocking) ja proaktiivisia rajapintoja kutsutaan myös asynkronisiksi. [92]

Tietoliikenneohjelmistojen olennainen vaatimus on kyky palvella montaa asiakasta kerrallaan ja säilyttää jokaiselle asiakkaalle oma protokollan konteksti. [93] Ohjelmisto voi jakaa asiakkaiden palvelemisen seuraavilla tavoilla:

1. jakamalla ohjelmiston useampaan prosessiin tai säikeeseen jossa yksi prosessi/säie palvelee yhtä asiakasta.

2. yksi prosessi/säie palvelee useampaa asiakasta seuraamalla eri asiakkaiden yhteyksien tilaa a) käyttäjätason säikeillä, b) käsittelyn jatkamisella (engl. continuation) tai c) tilakoneilla.

Suoritusympäristö ja -rajapinnat asettavat rajoituksia millaista rakennetta voidaan käyttää. Käyttöjärjestelmät eivät välttämättä tarjoa kaikkia rajapintoja, eikä kaikkien toimintatapojen toteutus ole luontevaa eri ohjelmointikielillä. POSIX standardi määrittelee rajapinnat, jotka kattavat synkronisen ja reaktiivisen siirräntärajapinnan. POSIX standardin myöhemmät laajennukset määrittelevät säikeiden toteuttamisen ja proaktiivisen siirräntärajapinnan. Useimmat käyttöjärjestelmät sisältävät omia laajennettuja rajapintoja, jotka ovat POSIX standardin rajapintoja tehokkaampia tai muutoin sopivampia käyttöjärjestelmän tapaan toteuttaa asioita.

3.3 Protokollat

Tässä luvussa esitellään työn puitteissa toteutettavat protokollat ja mitä suunnittelupäätöksiä kehitystyössä tehtiin. Protokollista käydään läpi yleinen toteutusmalli ja rakenteen olennaiset piirteet.

Taustatutkimuksen ja asetettujen päämäärien pohjalta ohjelmistoon päätettiin toteuttaa siirtorajapinta käyttäen sähköpostin protokollia. Sähköposti sopii ominaisuuksiltaan parhaiten sovelluksen vaatimuksiin. Sähköposti on universaalisti tuettu, ja rakenteeltaan joustava. Sähköposti pystyy lähettämään sekä pieniä, että suuria viestejä. Sähköpostin normaalissa toiminnassa sähköpostipalvelin ei tarvitse jatkuvia verkkoyhteyksiä, vaan on kykenevä selviämään verkkokatkoista. Sähköpostin epäideaalisuudet, kuten viestien monimutkainen säilöminen ja noutaminen, rajoittuvat rajapinnan yläpuolelle. Sisäisesti järjestelmä voi rakentaa viestit uudelleen kuljetusta varten.

Sähköpostin toteuttamisessa käytetään SMTP-protokollaa viestien lähettämiseen erityisvälitysverkkoihin ja IMAP-protokollaa viestien vastaanottamiseen. Viestit säilötään sähköpostimuodossa, josta ne voi tarvittaessa muuntaa muihin välitysmuotoihin. Rajapinnan sisällä viestit säilötään Maildir-rakenteeseen.

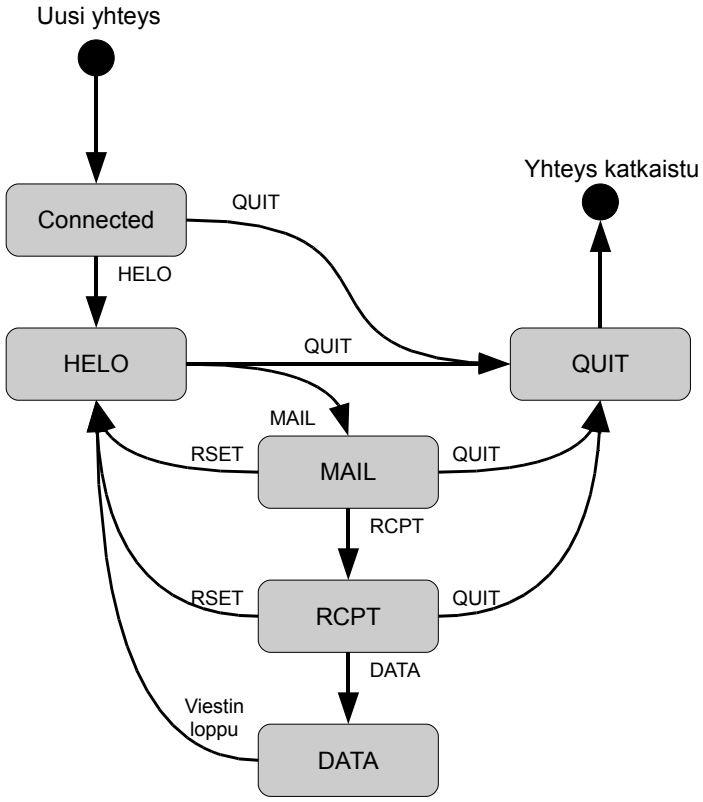
3.3.1 SMTP

SMTP-protokolla voidaan toteuttaa standardin mukaisesti tilakoneella, joka seuraa asiakkaan antamia komentoja. Ohjelmisto odottaa että asiakas on syöttänyt kokonaisen tekstirivin ennen käsittelyä. Jos puskurissa jo oleva osa ylittää pisimmän sallitun viestirivin pituuden, yhteys katkaistaan.

Protokollatilakone käsittelee syöterivin tulkitsemalla mikä protokollan komento on kyseessä, suorittaa komennon ja tekee tilasiirtymän. Mikäli komento ei ole sallittu sen hetkisessä tilassa, tulostetaan virheilmoitus. SMTP-protokollan tilakoneen tilakaavio on esitetty kuvassa 3.1. Tilat muodostuvat sähköpostiviestin lähettämisen vaiheista; ensiksi kuvataan lähettäjät ja vastaanottajat, sitten siirretään varsinainen viestidata. Nämä operaatiot voivat tapahtua vain peräkkäin ja vain tässä järjestyksessä.

Tilakoneessa on lisäksi ajastin, joka laukeaa jos asiakas ei lähetä syötettä riittävän pitkään aikaan. Ajastimen lauettua keskenjäänyt tapahtuma perutaan ja yhteys suljetaan.

Erityisvälitysverkkoon viestejä välittäessä halutaan hylätä mahdollisimman paljon virheellisiä viestejä heti alkuvaiheessa. SMTP-toteutus voi protokollan mukaan antaa välittömästi virheilmoituksen, jos palvelin pystyy heti sanomaan lähde- tai



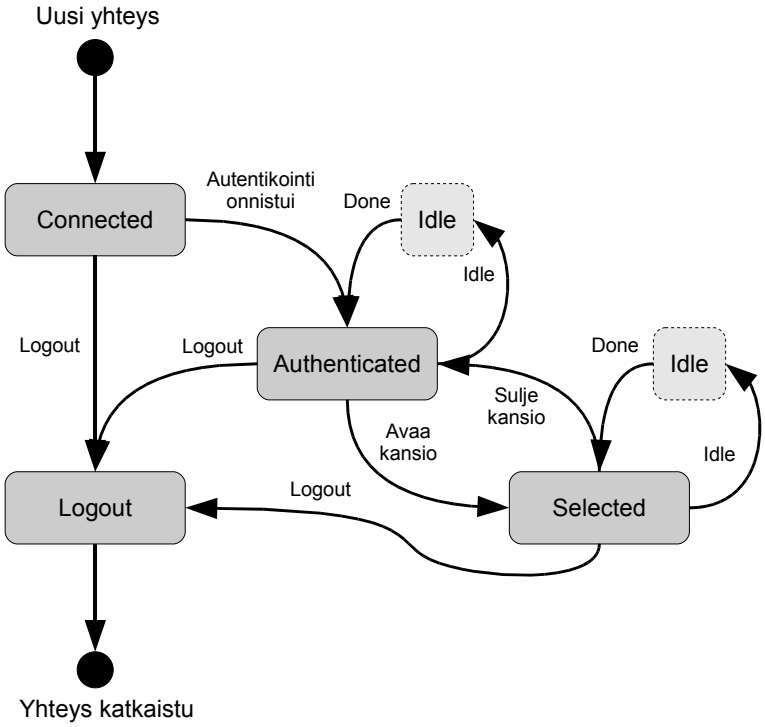
Kuva 3.1: SMTP-protokollan tilakone.

kohdeosoitteen tai viestirungon olevan viallinen. Tällöin käyttäjän asiakasohjelmisto ilmoittaa välittömästi virheen viestin lähettämisessä. Tämä monimutkaistaa viestin vastaanoton tulkintaa, mutta mahdollistaa paremman palautteen saamisen. Vaihtoehtoinen tapa on hyväksyä viesti toimitettavaksi, tulkita se välitysvaiheessa ja jos on ongelmia, lähettää takaisin virheilmoitus. Koska jälkimmäinen tapa välttää viestien sisällön tulkitsemisen protokollapalvelimessa, se mahdollistaa tietoturvan kannalta paremman ratkaisun toteuttamisen, mikä on tärkeää Internet-sovelluksissa [94].

Mikäli viestin välityksessä tapahtuu myöhemmässä vaiheessa virhe, voidaan tästä palauttaa ns. “bounce”-viesti, joka on sähköpostityylinen tapa kertoa toimitusvirheestä. [53] Kun virheviesti sisältää alkuperäisen viestin Message-id-arvon on se yhdistettävissä lähetettyyn viestiin.

3.3.2 IMAP

IMAP-protokolla toteutetaan pääosin standardin mukaiseksi tilakoneella, joka seuraa asiakkaan antamia komentoja. IMAP-protokolla on tekstipohjainen, joten ohjelmisto odottaa aina kokonaisen tekstirivin saapumisen ennen käsittelyä. IMAP-protokollalla on mahdollista siirtää kokonaisia sähköpostiviestejä komentojen mukana, joten tekstirivit täytyy puskuroida joustavasti. Kuvassa 3.2 on esitetty IMAP-protokollan tilakone. IMAP-protokollassa on vain muutamia komentoja, jotka aiheuttavat tilan muutoksen. Muut komennot suorittavat toimintonsa muuttamatta protokollan tilaa.



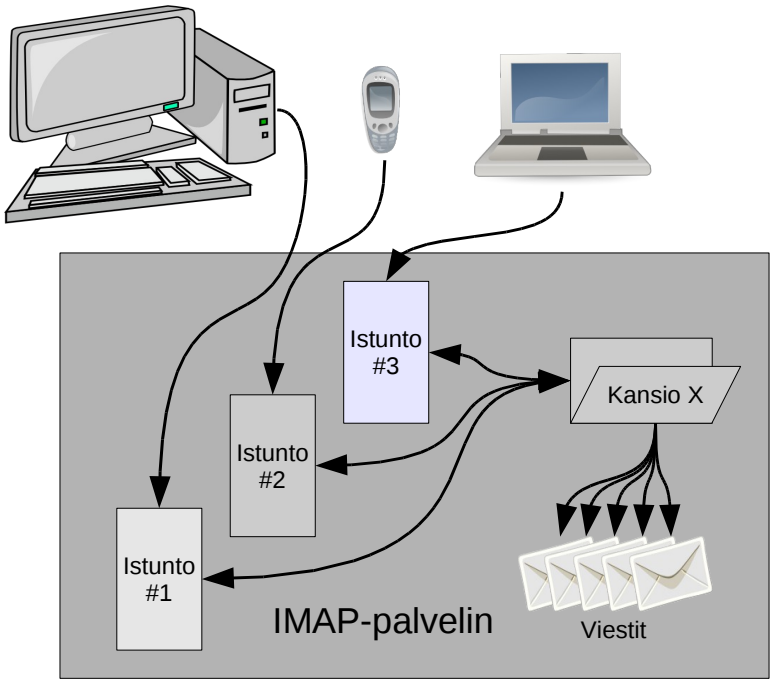
Kuva 3.2: IMAP-protokollan tilakone.

Tilakoneessa on lisäksi ajastin, joka laukeaa jos asiakas ei lähetä syötettä riittävän pitkään aikaan. Ajastimen lauettua yhteys suljetaan. IMAP-protokollassa on mahdollisuus siirtyä “IDLE”-komennon avulla tilaan, jossa käyttäjä vain passiivisesti vastaanottaa postilaatikon muutostietoja. Tässä tilassa ajastin on kytketty pois päältä.

IMAP-protokollan syntaksi on monimutkainen, joten toteutettaessa tulee siis olla tarkkana komentojen tulkinnassa ja validoinnissa. Erityisen haasteen muodosta-

vat monimutkaiset noudot (“FETCH”) ja haut (“SEARCH”), koska niiden toteuttaminen vaatii viestien rakenteen tulkkaamista. Koska viestisovellusta olisi tarkoitus käyttää sulautetussa ympäristössä rajoitetuilla resursseilla, päätettiin IMAP-protokollan haku-komento jättää toteuttamatta alkuvaiheessa. Hakutoiminnot voivat helposti vaatia kaikkien postilaatikon viestien lukemista ja tarkastelemista.

Toinen IMAP-protokollan ongelmakohta on postilaatikoiden yhtäaikainen käyttö. Tilannetta on esitelty kuvassa 3.3. Jokaisen IMAP-protokollan käyttäjä muodostaa oman ns. istunnon käytössään olevasta postilaatikon kansiosta ja IMAP-palvelimen tulee toimia käyttäjän istunnon mukaisesti. Muiden käyttäjien aiheuttamat muutokset kansiossa voidaan välittää käyttäjälle vain kun käyttäjä suorittaa komennon, jonka syntaksi sallii tämän. Siihen asti, että muutokset pystytään välittämään käyttäjälle, tulee IMAP-palvelimen totella komentoja aivan kuin muiden käyttäjien muutokset eivät olisi tehneet mitään muutoksia. Tämä aiheuttaa vaikeuksia erityisesti, kun suoritetaan pysyviä muutoksia, kuten poistetaan tai lisätään viestejä.



Kuva 3.3: IMAP-postilaatikon yhteiskäyttö.

Työn toteutuksessa IMAP-protokollan epäselvemmät kohdat on pyritty avaamaan mahdollisimman yksinkertaisesti. Postilaatikon kansiot keräävät muutokset ja jollei tyhjennystä erityisesti pyydetä, suorittaa poistettavaksi merkittyjen viestien poista-

misen vasta kun kaikki postilaatikkoa käyttävät asiakasyhteydet on suljettu. Tyhjennystä ("EXPUNGE") pyydettäessä poistetaan viestit ja merkitään asiakkaiden konteksiin viestit poistetuiksi. Koska tietoa muutoksesta ei voi toimittaa kaikille asiakkaille välittömästi, täytyy vastata kieltävästi kaikkiin komentoihin joiden tulos olisi muuttunut poistojen takia, kunnes saadaan välitettyä tieto poistoista. Tällöin asiakas ei pysty enää noutamaan tai vaikuttamaan viestiin, joka on poistettu, mutta ei myöskään tahattomasti tekisi muutoksia postilaatikon muihin viesteihin. [95]

Recent-lipun määritelmä standardissa on epätäsmällinen ja osittain hankala. Toteuttaessa käytettiin tapaa tulkita viestin sijaitseminen Maildir-rakenteen /new-hakemistossa merkiksi, että viesti on uusi ja siten omaa Recent-lipun. Tämän takia ei voida helposti avata postilaatikkoa pelkässä lukumoodissa, koska IMAP-standardi vaatii ettei lukumoodissa postilaatikon avaaminen saa vaikuttaa Recent-lippuihin. Yksinkertaisin toteutus on antaa Recent-lippujen muuttua standardin vastaisesti.

IMAP-protokollan yhtäaikaisen käytön tekee monimutkaisemmaksi osa komennoista, jotka muuttavat postilaatikon tilaa tavoilla, joita ei voi välittää muille käyttäjille. Näitä komentoja ovat esimerkiksi postilaatikon kansion poistaminen tai uudelleennimeäminen. Yksinkertaisin ratkaisu olisi kieltäytyä poistamasta tai nimeämästä kansioita, jotka ovat useamman käytössä, mutta tämä voi aiheuttaa tilanteen, jossa kansiota ei pysty poistamaan tarvittaessa. Suoraviivainen käytännön vaihtoehto on tilanteen sattuessa katkaista muiden postilaatikon käyttäjien IMAP-yhteys, minkä kuuluisi saada heidät ottamaan uuden yhteyden palvelimelle ja tällöin saavan tiedon muutoksista. Tämän lisäksi voi kansion poiston käsitellä siten, että sen poistanut käyttäjä ja kaikki sen jälkeen yhteyden ottavat uudet käyttäjät näkevät kansion poistettuna/nimettynä, mutta vanhat käyttäjät näkevät ja voivat käyttää sitä yhä vanhalla nimellään. Jälkimmäisin ratkaisu on kaikkein yhteensopivin, mutta hyvin monimutkainen toteuttaa. Toteutuksessa käytettiin toisena esiteltyä mallia, jossa katkaistaan käyttäjiltä yhteys jos postilaatikko menee tilaan, jota ei voida protokollan kautta kuvata. Yhteyden katkaisua käytetään myös yleisesti virhetilanteiden hoitamiseen, jossa käyttäjä on saanut protokollan tilan sekaisin.

Viestien säilömiseen käytetään muokattua Maildir-rakennetta. Jokaiselle käyttäjätunnukselle luodaan oma hakemisto, jonne säilötään käyttäjän sähköpostilaatikko, jokainen kansio omaksi Maildir-rakenteekseen. Käytetty Maildir-rakenne poikkeaa alkuperäisestä määritelmästä seuraavin tavoin:

- Tiedostonimissä käytetään kaksoispisteen (":") tilalla puolipistettä (";"), koska kaksoispistettä ei pysty käyttämään tiedostonimissä osalla käyttöjärjestelmistä. Tämä muutos tehtiin siirrettävyyden parantamiseksi.

- Tallennettujen viestien sisältöä ei muokata, mutta viestien IMAP-protokollan liput ja UID-luvut säilötään viestin tiedostonimeen. Tämä muutos yksinkertaistaa IMAP-palvelimen toteutusta, koska ei tarvitse ylläpitää erillistä tietokantaa UID-luvuista.

- Postilaatikon kansion yksilöivä UIDvalidity-luku tallennetaan tiedostoon nimeltä ".uidvalidity" Maildir-rakenteen juureen. Tämä muutos yksinkertaistaa IMAP-palvelimen toteutusta.

- Postilaatikon kansiohierarkia toteutetaan Maildir-määritelmästä poiketen: sen sijaan, että Maildir-päähakemisto olisi käyttäjän pääkansio "INBOX", käsitellään pääkansiota kuin mitä tahansa kansiota. Käytetty malli on siistimpi, koska se ei tarvitse poikkeuskäsittelyjä pääkansion käyttämisen vuoksi.

Vaikka Maildir on alunperin tarkoitettu toimimaan yhtäaikaisesti useamman eri sovelluksen käyttäessä sitä ilman lukkojärjestelmiä, IMAP-järjestelmien kanssa tämä ei ole täysin mahdollista. Ollakseen yhteensopiva ja tehokas, IMAP-järjestelmän täytyy hallita Maildir-rakenteen muutoksia ja tallentaa Maildir-rakenteeseen ylimääräistä tietoa. Jos useampi eri IMAP-palvelin käyttää samaa Maildir-rakennetta, tulee IMAP-palvelimien jakaa näiden kahden käyttäjän pääsy lukoilla. Jos Maildir-rakennetta käyttää vain yksi IMAP-sovellus, lukkoja ei tarvita. Tässä työssä toteutetussa ohjelmistossa vain yksi IMAP-palvelinprosessi käyttää Maildir-rakenteita, jolloin käytön hallinta suoritetaan prosessin sisäisesti.

3.4 Ohjelmistotestaus

Ohjelmistotestaaminen on osa ohjelmistojen laaduntarkkailua (engl. quality assurance, QA). Laaduntarkkailulla tarkoitetaan menettelyjä, joilla tuotteen laatu voidaan taata. Merkitys ja prosessit ovat samanlaiset niin ohjelmistoille kuin esimerkiksi autojen valmistukselle. Laaduntarkkailun prosessin käsittely ei mahdu tähän työhön mukaan. Tässä luvussa käsitellään mitä ohjelmistotestaamisen työkaluja on käytetty ohjelmiston kehityksessä. [96]

Työhön liittyen suoritettiin seuraavia ohjelmistotestaamisen menetelmiä: sovellusyhteensopivuustestausta, automatisoitua yhteensopivuustestausta ja yksikkötestausta avustusohjelmistoilla. Testien rakennetta ja suorittamista käsitellään tässä luvussa. Tuloksia analysoidaan luvussa 4.

Sovellusyhteensopivuus

Sovellusyhteensopivuustestaus on erityisesti Internet-standardien toteutuksessa tärkeä osa sovelluskehitystä. Tässä työssä yhteensopivuuden testauksessa käytetään synteettisiä testejä ja sovellustestejä. Testeistä kerättyjä tuloksia käytetään sovelluksen yhteensopivuuden arviointiin, tehtyjen kompromissien arviointiin ja jatkokehityksen suunnitteluun.

Yleisimmät SMTP- ja IMAP-rajapintoja käyttävät sovellukset ovat Microsoft Windows käyttöjärjestelmän mukana tuleva Outlook Express, Microsoft Office -ohjelmistopaketin mukana jaettava Outlook (tarkka malli riippuu Office-paketin versiosta) ja yleisin avoimen lähdekoodin sovellus, Mozilla Thunderbird, sekä UNIX-postiohjelmisto Mutt.

Testauksessa suoritetaan jokaisessa testattavassa ohjelmistossa samat toiminnot määrätyssä järjestyksessä. Testauksessa suoritettavat tehtävät on esitelty taulukossa 3.3. Toimintojen onnistumiset, epäonnistumiset ja poikkeamat käytöksessä kirjataan ylös.

Taulukko 3.3: Sovellustestit.

Tehtävä	Kuvaus
1	Sähköpostitunnuksen luominen.
2	Postilaatikon avaaminen.
3	Sähköpostiviestin lähettäminen omalle tunnukselle.
4	Itselleen lähetetyn viestin lukeminen.
5	Viestin lähettäminen jollekin toiselle tunnukselle.
6	Viestin lähettäminen toisella prioriteetillä.
7	Muualta vastaanotetun viestin lukeminen.
8	Tarkista postilaatikon muutosten ilmoitusnopeus.
9	Tarkista saman postilaatikkon käyttö useammalta yhteydeltä.
10	Postilaatikon kansion luominen ja tuhoaminen.
11	Postilaatikon kansion siirtäminen yhtäaikaisen yhteyden kanssa.

Koska sovellusyhteensopivuusviat saattavat johtua kumman tahansa ohjelmiston, joko kehitettävän tai vasten testattavan, toteutusvirheestä, tulee pitää huoli, että kehitettävä ohjelmisto toimii standardin tai yleisen käytännön mukaan. Mikäli testissä käytettävä sovellus toimii vastoin standardia, kuuluu hyviin tapoihin tehdä korjaus-

pyyntö sovelluksen kehittäjille. Tämä kuitenkaan ei ole käytännön ratkaisu, koska sovelluksen kaikki aikaisemmat käytössä olevat versiot sisältävät yhä havaitun puutteen. Vaihtoehtoisesti kehitettävään ohjelmistoon voidaan toteuttaa ns. kiertotapa, jolla tehdään esimerkiksi asiakassovelluskohtainen muutos, jolla saadaan ohjelmisto toimimaan ongelmallisen sovelluksen kanssa.

Automatisoidut testit

Automatisoitujen testien tarkoitus on suorittaa mekaanisesti rajapintojen toimintoja ja erikoistapauksia. Automatisoitu testaaminen on yleensä ohjelmistokehityksen perustyökalu, koska sitä voidaan toistaa välittömästi ja aina muutosten jälkeen, ns. regressiotestaus. Automatisoituja stressitestejä käytetään myös suorituskyvyn mittaamiseen toteutusmallin analysoimiseksi.

Testauksessa käytettävät automatisoidut työkalut ovat “ImapTest”, “Swaks” (Swiss Army Knife SMTP), “Xstress” ja “Fetchmail”. ImapTest ja Swaks ovat vastaavasti IMAP- ja SMTP-protokollatoteutusten testaamiseen. Xstress on SMTP-protokollan stressitestaustyökalu ja Fetchmail on IMAP-yhteensopiva sähköpostin noutotyökalu. Stressitestit suoritetaan erikokoisilla viesteillä ja eri määrillä yhtäaikaisia käyttäjiä. Oletusasetuksillaan ImapTest suorittaa satunnaisia sarjoja komentoja tietyllä tyyppijakaumalla, käyttäen monta rinnakkaista IMAP-yhteyttä. Tämän lisäksi ImapTest voi suorittaa käsikirjoitettuja useamman IMAP-yhteyden testejä. Kehityksen yhteydessä valittiin käsikirjoitetuista testeistä alajoukko, joka kattaa toteutetut ominaisuudet. Vastaavasti Swaks suorittaa viestin lähetyksen SMTP-protokollalla. Täysin itsenäistä viestien luontia Swaks ei tue, vaan vähintään kohdeosoite on aina määriteltävä. Lisäksi Swaks tukee SMTP-laajennuksien testaamista, mutta niitä tässä työssä ei käytetä.

Testaustyökalut

Työssä käytettiin kääntäjänä GNU Compiler Collection (GCC) [97] -ohjelmistopaketin C-ohjelmointikielen kääntäjää. Lähdekoodi käännettiin kääntäjän täysillä varoitusasetuksilla. Jos lähdekoodissa on virhe, joka rikkoo ohjelmointikielen syntaksia, tästä saadaan varoitus ja voidaan reagoida varoitukseen. Kääntäjä pystyy tunnistamaan vain selvät syntaksivirheet. Jos virhe on ohjelman logiikassa, ei kääntäjä tätä pysty ilmoittamaan. Kääntäjät saattavat joissain tilanteissa antaa varoituksia tilanteista, jotka ovat silti oikein toteutettu.

Kehitystyötä tehdessä ja ohjelmistotestejä suorittaessa ohjelmistoa suoritettiin simulaattorityökalulla, nimeltä Valgrind [98]. Simulaattori tulkitsee käännetyn ohjel-

man ja simuloi käytetyt käskyt. Tällöin se voi tarkastaa muistioperaatioiden käytön ja seurata muistin varaamista ja vapauttamista. Näin voidaan havaita muistin käyttö varattujen alueiden ohi sekä tilanteet, joissa ohjelma ei vapauta varaamaansa muistia, eli ns. vuotaa muistia. Tämä poimii ensisijaisesti virheitä muistin varausten käsittelyssä, mutta antaa palautetta myös tilanteista, joissa ohjelmisto lakkaa kokonaan toimimasta.

Ohjelmistotestien kattavuutta arvioitiin suorittamalla lähdekoodin peittoalueanalyysi ja profilointi. Työssä käytettiin GCC-ohjelmistopaketin peittoaluetyökalua nimeltä Gcov ja profilointityökalua nimeltä Gperf. Peittoalueanalyysin tarkoitus on analysoida ohjelman suorituksesta mitä alueita lähdekoodista suoritetaan. Tällöin voidaan arvioida kattaako ohjelmistotestit toteutuksen kaikki osat, ts. mikä on testien peittoalue. Peittoalueanalyysin pohjalta voidaan laajentaa testejä kattamaan mahdollisimman suuri osa lähdekoodista. Profiloinnin tarkoitus on analysoida mitkä osat ohjelman ajon aikana suoritetaan useimmin ja missä osissa kuluu eniten suoritusaikaa. Profilointitietojen pohjalta voidaan tunnistaa ohjelmiston pullonkaulat ja keskittyä kehitystyössä kehittämään näitä osia, tai muuttamaan ratkaisua siten että tietyltä pullonkaulalta vältytään.

3.5 Yhteenveto

Ohjelmistokehityksen ydinasioita on valita käytettävät rajapinnat ja toteutusmallit sovelluksen vaatimusten mukaan. Työssä tehdyn tutkimuksen pohjalta valittiin SMTP/IMAP-protokollat toteutettavaksi rajapinnaksi. Siirrettävyys ja siirrettävyystekniikat mahdollistavat kerran kehitetyn ohjelmiston käyttämisen eri ympäristöissä ja sovelluksissa. Ohjelmistotestaaminen toimii ohjelmistokehityksen apuvälineenä, osana laaduntarkkailua.

Tietoliikenneprotokollien toteuttaminen on yleensä hyvin vaativaa. Tilakoneet ovat suoraviivainen malli, jolla monimutkaisia interaktioita voidaan hallita. Esitellyillä malleilla ja rakenteilla voidaan toteuttaa järjestelmä, jonka toiminta on yhdenmukaista ja luotettavaa. Tältä pohjalta kehitetään ohjelmistokomponentti, joka integroi valitut protokollat osaksi viestintäjärjestelmää.

Luku 4

Toteutus

Tässä luvussa käsitellään toteutetun sovelluksen rakennetta ja käytettyjä komponentteja. Tämän jälkeen käydään läpi suoritetun ohjelmistotestauksen tulokset, joiden pohjalta arvioidaan sovelluksen yhteensopivuutta ja suorituskykyä, ja käytetyn testausjärjestelyn kattavuutta.

Sovelluksen kehitysympäristönä käytettiin Debian GNU/Linux -käyttöjärjestelmän [99] versiota 5.0.2 (“Lenny”). Debian GNU/Linux (lyhyesti Debian) on POSIX-yhteensopiva UNIX-tyylinen käyttöjärjestelmä, joka käyttää Linux-käyttöjärjestelmäydintä [100] ja GNU-käyttöjärjestelmäohjelmistoja [101].

Kehitystyökaluina käytettiin GNU Compiler Collection (GCC) -kääntäjätyökaluja C-kääntäjänä, ja Anjuta Integrated Development Environment (IDE) -kehitysympäristöä, sekä GNU Debugger (GDB) ja Valgrind -debuggaustyökaluja. Käännöstyön automatisointiin käytettiin CMake-työkalua.

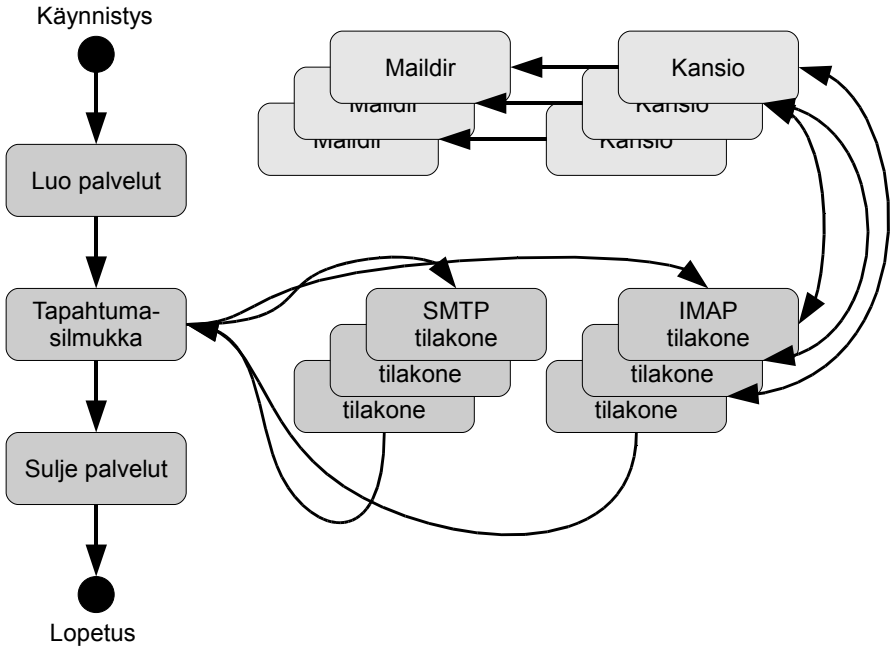
Automatisoidun testauksen yhteydessä sovellukselle suoritettiin koodin profilointi käyttäen gprof- ja gcov-työkaluja. Profiloinnin tarkoituksena on löytää suorituksessa eniten suoritusaikaa kuluttaneet osat koodia, jolloin voi keskittämällä resurssit näiden pullonkaulojen parantamiseen saadaan suurin käytännön etu. Profiloinnin yhteydessä suoritettiin synteettisten testien peittotarkastelu käyttäen gcov-työkalua. Peittotarkastelulla selvitetään testien riittävyyttä ohjelmiston testaamiseen.

Suorituskykytestauksessa käytettiin Intel Core i7 palvelimia, jotka vastaavat viimeisintä PC-tietokonetekniikkaa tätä työtä kirjoittaessa, VIA C7-M -pohjaista kannettavaa ja Intel Atom -pohjaista ns. “nettop” -tietokonetta. Jälkimmäiset kaksi ympäristöä vastaavat tehokkaamman pään sulautettuja laitteistoja.

4.1 Sovelluksen rakenne

Sovelluksen toteutusmalliksi suunnitteluperusteiden (nimetty luvussa 1.1) pohjalta valittiin Libevent-kirjaston [102] avulla toteutettu, yhtä prosessia käyttävä malli, jossa protokolla-asiakkaiden tilaa seurataan tilakoneilla. Libevent on korkean suorituskyvyn siirrettävyyskirjasto, joka yhdistää eri käyttöympäristöjen erilaisia toteutuksia yhtenäisen rajapinnan taa. Libevent yhdistää reaktiivisen siirrännän, signaalit ja ajastimet tapahtuma-pohjaiseen rajapintaan. Tämä soveltuu erinomaisesti verkkopalvelinsovellusten toteuttamiseen ja yksinkertaistaa merkittävästi toteuttamista.

Tilakone-toteutusmalli on siirrettävä ja yksinkertainen, jolloin se on myös nopea kehittää ja toimintavarma. Toteutusmallin rakennetta on esitelty kuvassa 4.1. Toteutusmalli sallii kaikkien protokollien toteuttamisen yhdellä ohjelmalla. Asiakasyhteydet yhdistetään protokollan mukaiseen tilakoneeseen yhteyden muodostamisen käsittelyssä. Tilakonemallit esiteltiin luvussa 3.3. Tämän lisäksi toteutusmalli ottaa huomioon IMAP-protokollan kansioiden yhteiskäytön, säilyttämällä kansioiden yhteiskäytöstä oman tietorakenteen.



Kuva 4.1: Palvelimen tilakone.

Käytetyssä toteutusmallissa on siis neljä peruskomponenttia: SMTP-protokollayhteydet, IMAP-protokollayhteydet, IMAP-postilaatikon avatut kansiot ja kan-

sioita vastaavat Maildir-rakenteet. SMTP-protokollakäsittelijä on riippumaton muista komponenteista ja siten seuraa SMTP-protokollan tilakonemallia hyvin tarkasti. IMAP-protokollakäsittelijä joutuu IMAP-protokollan tilakonemallin lisäksi seuraamaan IMAP-istunnon ja valitun postilaatikon kansion tilan päivityksiä. Jotta asiakasohjelmistot pystyisivät saamaan asynkronisia päivitystietoja kansion tilasta, kaikki kansion tilaa muuttavat operaatiot aiheuttavat päivitysoperaation, joka yrittää lähettää päivitykset jokaiselle kansiota käyttävälle IMAP-istunnolle.

Siirrettävyyden parantamiseksi ja vaivan säästämiseksi käytettiin GLib-yleiskirjastoa tietorakenne-primitiivien toteuttamiseen ja rajapintojen siirrettävyyden parantamiseen. GLib on GTK+-projektin kehittämä yleiskirjasto, jota käytetään pohjana GTK+-pohjaisissa graafisissa sovelluksissa.

4.2 Ohjelmistotestauksen tulokset

Tässä luvussa käydään läpi ohjelmistotestauksen tulokset. Ensiksi tarkastellaan automatisoitujen testien tuloksia ja mitä niistä voi päätellä. Lopuksi tarkastellaan sovellusyhteensopivuustestien tuloksia.

Automatisoiduilla testiohjelmistoilla suoritettiin ohjelmiston toimivuuden, yhteensopivuuden ja suorituskyvyn testaukset. SMTP- ja IMAP-protokollien suorituskykytestin tulokset ovat liitteessä A. Suorituskykytestit tehtiin kolmella eri laitteistolla: Intel Core i7 -tekniikka käyttävillä Intel Xeon E5520 prosessoreilla varustettulla palvelimella, Intel Atom 330 prosessorilla varustetulla nettop-tietokoneella ja VIA C7-M prosessorilla varustetulla kannettavalla tietokoneella. Testit suoritettiin toisella samanlaisella Intel Core i7 -palvelimella johon testattavat koneet oli kytketty 1 Gb/s Ethernet-verkkoyhteyksillä. Suorityskykytesteinä suoritettiin eri kokoisten viestien lähettämistä SMTP-protokollalla ja vastaanottamista IMAP-protokollalla. Tämän lisäksi testattiin yhtäaikaisten yhteyksien vaikutusta suorituskykyyn.

Liitteen A kuvissa A.1, A.2 ja A.3 on esitetty SMTP-protokollan suorituskykymittaus, jossa lähetettiin 10000 tietyn kokoista viestiä ja mitattiin kauanko testissä kesti. Testeissä havaittiin että SMTP-rajapinnan suorituskyky oli käytetyillä laitteistoalustoilla pienillä viesteillä tasainen yli 800 viestiä sekunnissa, joka laski lineaarisesti viestien koon kasvaessa. Prosessorin käytössä käyttäjän osuus kasvoi viestikoon kasvaessa, mikä johtunee viestin siirrossa tehtävästä viestin tekstiriveille tehdystä prosessoinnista (rivinvaihto korjataan, tarkastetaan onko kyseessä viestin viimeinen rivi).

Kuvissa A.7, A.8 ja A.9 on esitetty SMTP-protokollan suorituskykymittaus, jossa yhteensä 9600 viestiä lähetetään eri määrällä yhtäaikaisia lähettäjiä. Testeistä ha-

vaitaan suorituskyvyn huipun olevan noin 20 yhtäaikaisen yhteyden kohdalla. Yksittäisen yhteyden suorituskyvyn rajoittaa protokollan vaatimat synkroniset vaiheet, joissa asiakkaan on odotettava palvelimen vastausta. Yhteyksien suorittaminen yhtäaikaa vähentää protokollan aiheuttamia viiveitä, mutta vastaavasti kasvattaa muuta resurssinkäyttöä. Prosessorin käytöstä käyttäjän ja järjestelmän osuudet pysyvät liki kiinteinä, mikä antaisi olettaa että suorituskyvyn pullonkaulat voivat olla verkkoyhteydessä tai testausohjelmiston toteutuksessa.

IMAP-protokollan suorituskykymittauksen tuloksia on esitelty kuvissa A.4, A.5 ja A.6. Tässä testissä ladatttiin ja poistettiin 10000 tietyn kokoista viestiä IMAP-palvelimelta. Testeissä havaittiin IMAP-protokollan suorituskyvyn olevan käytetyillä laitteistoalustoilla kymmenien viestien luokkaa sekunnissa. Suorituskyky laski merkittävästi vasta suurilla (>64 kiB) viesteillä. Prosessorin käytöstä havaitaan että viestien lataaminen on palvelimelle hyvin intensiivistä työtä. Tämä liittyy IMAP-protokollan tapaan käsitellä postilaatikkoja, erityisesti silloin kun postilaatikossa on paljon viestejä.

Kuvissa A.10, A.11 ja A.12 on esitetty IMAP-protokollan suorituskykymittaus, jossa suoritetaan ImapTest-työkalulla IMAP-protokollan komentoja satunnaisesti, tietyn testijakauman mukaisesti. Testiohjelma seuraa käskyjen toimintaa protokollan korrektiuden kannalta ja laskee suoritettujen käskyjen määrät. Testeissä havaittiin, kuten SMTP-protokollan tapauksessa, että yksittäinen yhteys kärsii protokollan synkronisuuden aiheuttamasta suorituskyvyn laskemisesta, jolloin suorituskyvyn huippu saavutetaan vasta, laitteistosta riippuen, yli kahdeksan yhtäaikaisen yhteyden avulla. IMAP-palvelin pystyi joka laitteistolla suorittamaan satoja komentoja sekunnissa. Suorituskyvyn rajoittavana tekijänä vaikuttaa olevan käyttöjärjestelmän suorituskyky, järjestelmäoperaatioiden muodossa. Yhtäaikaisten yhteyksien toisilleen aiheuttamaa protokollatason kuormaa ei testistä pystytty vahvistamaan.

IMAP-protokollan yhteensopivuustestisarjan tulokset ovat liitteenä B. Testisarjassa suoritettiin kaikki ImapTest-ohjelman testit, paitsi “SEARCH”-komentoa ja laajennuksia, joita ohjelmisto ei toteuta, ensisijaisesti testaavat testit. Yhdestätoista testiryhmästä kuusi suoritettiin täysin onnistuneesti, neljässä jotkin komennot antoivat palautteen, jota testityökalu ei hyväksynyt. Neljässä epäonnistuneesta testissä viallisen palautteen antavia komentoja on yhteensä viisi, joista yksi johtuu toteuttamattomasta “SEARCH”-komennosta, kolme johtuu valinnaisesta laajennuksesta, jota ohjelmisto ei toteuta, mutta ohjelmisto silti käyttää, ja yksi “RECENT”-lipun käsittelystä.

Automatisoidun testauksen yhteydessä muodostettiin ohjelmasta profilointi- ja peittoaluetilastoja. Ohjelmiston testauksen peittoalueet on esitelty taulukossa 4.1.

Testejä ei ole erityisesti suunniteltu maksimoimaan peittoaluetta. Peittoaluetta analysoimalla havaittiin suurimman osan peittymättömistä funktioista joko kuuluvan virhetilanteiden käsittelyyn tai olevan ns. kuollutta koodia, joka on käännetty ohjelman mukana, mutta jota oikeasti ei kutsuta mistään. Koska kaikkia virhetilanteita ei pysty järkevästi tuottamaan, voidaan tehdä erillinen tukiohjelma, joka suorittaa ko. funktiot erillisessä keinotekoisessa ympäristössä ja siten aiheuttaa virhetilanteet.

Taulukko 4.1: Peittoaluetestin tulokset.

Tiedosto	Lähdekoodirivejä	Peittoalue
rfc2822.c	226	75.66%
mailbox.c	660	76.35%
imap_helper.c	74	76.36%
ai.c	41	85.37%
common.c	254	43.70%
imap.c	1016	67.52%
net.c	106	39.62%
smtp.c	183	71.04%

Profilointityökalu laski missä funktioissa ohjelmisto vietti eniten aikaa, mutta johtuen kehityksessä käytetyn koneen nopeudesta, ohjelmiston asynkronisesta rakenteesta ja siitä että testidatan käyttämät viestit olivat pieniä ja siten mahtuivat käyttömuistiin, yksikään funktio ei käyttänyt merkittävästi aikaa. Käytetyimmät funktiot olivat Maildir-rakenteen tarkistusfunktio ja tekstirivien kirjoittamiseen ja tulkitsemiseen käytetyt funktiot. Tämän pohjalta pystyttiin päättelemään, että yksittäinen raskain operaatio on IMAP-protokollan “LIST”-komento, jonka käy rekursiivisesti läpi kaikki postilaatikon kansiot.

Sovellustestien tulokset on esitetty taulukossa 4.2. Sovellustestit esiteltiin taulukossa 3.3. Taulukon merkintöjen tarkoitus on seuraava: “Ok” ominaisuus toimi kuten pitäisikin, “Huom.” ohjelmiston toiminnassa on mainitsemisen arvoinen poikkeama, “Ei” testattu ominaisuus ei toiminut. Testit suoritettiin rinnakkain neljä eri ohjelmistolla, käyttäen samaa käyttäjätiliä sähköpostipalvelimella.

Testissä 2 sovellusten toiminta erosi siten että Outlook Express ja Outlook 2003 postilaatikkoa avatessa esittivät vain pääkansion INBOX, ja muut kansiot täytyi lisätä erillisen komennon (“IMAP Folders”) kautta. Outlook 2003 pystyttiin asettamaan siten että se näyttää aina kaikki kansiot. Mutt ei esitä kansioita lainkaan vaan

Taulukko 4.2: Sovellustestien tulokset.

Tehtävä	Outlook Express 6	Outlook 2003	Thunderbird	Mutt
1	Ok	Ok	Ok	Ok
2	Huom.	Huom.	Ok	Huom.
3	Ok	Huom.	Ok	Ok
4	Ok	Ok	Ok	Ok
5	Ok	Huom.	Ok	Ok
6	Ok, Huom.	Ok, Huom.	Ok, Huom.	Ei, Huom.
7	Ok	Ok	Ok	Ok
8	Huom.	Ok	Ok	Ei, Huom.
9	Ok	Ok	Ok	Ei
10	Ok	Ok	Ok	Ei
11	Huom.	Huom.	Huom.	Ei

avaa aina yhden kansion. Oletettu toimintatapa oli että kaikki IMAP-palvelimen kansiot näkyisivät käyttäjälle.

Outlook 2003 ei testissä 3 suostunut lähettämään viestiä sähköpostiosoitteeseen, joka on muotoa “tunnus@kohde”, siten että “kohde” osassa ei ole pisteellä erotettuja osia. Todennäköisesti tämä on jonkin tason validointiominaisuus, mutta se estää silti standardien mukaisten osoitteiden käyttämisen. Sama vika ilmeni myös testissä 5.

Sähköpostin lähettämisessä eri prioriteeteillä ilmeni käytäntöeroja. Thunderbird käytti “X-Priority”-otsikkoa, Outlook Express otsikkoja “X-Priority” ja “X-MSMail-Priority”, ja Outlook 2003 otsikkoja “X-Priority”, “X-MSMail-Priority” ja “Importance”. Lisäksi Thunderbird tarjosi käyttäjälle viisi prioriteettitasoa, kun taas Outlookit vain kolme. Mutt ei tarjonnut viestien prioriteetin muuttamista, mutta otsikkokenttä oli mahdollista lisätä käsin.

Testissä 8 tarkasteltiin asiakassovellusten kykyä havaita asynkronisia muutoksia postilaatikoissa. Thunderbird ja Outlook 2003 käyttivät kahta IMAP-yhteyttä siten että toinen seurasi jatkuvasti “INBOX”-laatikon muutoksia. Outlook Express käytti myös kahta yhteyttä, mutta seurasi vain sitä kansiota, joka kullakin hetkellä oli käyttöliittymästä valittu.

Mutt ei seurannut muutoksia testissä 8. Koska tehtävä 9 liittyi tähän ominaisuuteen, se ei myöskään toiminut Mutt-ohjelmalla. Mutt tarkasti postilaatikon uudestaan tietyn ajan kuluttua ja muutoksia havaitessaan ilmoitti vain käyttäjälle, että

palvelimen tila eroaa näytetystä postilaatikosta. Vastaavasti testeissä 10 ja 11, Mutt-ohjelman käyttöliittymässä ei ole IMAP-kansioiden käsittelyyn liittyviä komentoja. Mutt-ohjelman toimintamalli vastaa ns. offline-toimintatilaa, jossa asiakasohjelma lataa eräajona postilaatikon sisällön ja käsittelee sitä kuin paikallista kansiota.

Testissä 11 yksikään sovellus ei toiminut täysin tyydyttävästi. Kansion siirtäminen aiheutti yksinkertaisen toteuttamistavan mukaisesti asiakasohjelmien laatikon sisältöä seuraavan IMAP-yhteyden katkaisemisen, mutta tämä ei saanut ohjelmistoja tarkastamaan muutoksia postilaatikossa. Ohjelmistot luulivat uudelleennimetyn kansion yhä olevan olemassa alkuperäisellä nimellään, ja raportoivat käyttäjälle palvelimen virheilmoitukset kun sitä yritti käyttää.

4.3 Yhteenveto

Sovelluksen käytännön toteutuksessa käytettiin yksinkertaista ja vaivalla suunniteltua mallia. Tietorakenteet ja toimintamalli on valittu toteuttamaan protokolla standardin mukaisesti. Kehitystyökalut ja automatisoidut testit varmistavat ettei järjestelmässä ole ohjelmistovirheitä.

Suorituskykytesteillä arvioitiin ohjelmiston suorituskykyä testien tilanteissa ja muutamalla eri laitteistoalustalla. Ohjelmistosta paikallistettiin muutamia rajoituksia. Osa rajoituksista on protokollasta johtuvia, mutta osaan voidaan ottaa kantaa ohjelmiston jatkokehityksessä.

Sovellusyhteensopivuustestit osoittavat kuitenkin että standardinmukainenkaan toteutus ei ole välttämättä ongelmaton. Sovelluksien tulkinnat protokollista ei myöskään ole aina täydellisiä. Sovellusten toteutuserot usein johtuvat ohjelmiston valitusta toimintatavasta, joka ei ole aina samanlainen kuin protokollaa kehittäessä ollaan tarkoitettu.

Luku 5

Analyysi ja yhteenveto

Tässä työssä käsitellään sähköisen viestinnän sovittamista erityisvälitysverkkoihin. Tietoverkkojen kehitys on edennyt asteelle, jossa verkkoyhteyksiä on saatavilla lähes kaikkialla ja lähes kaikissa laitteissa. Verkkoyhteyksien ominaisuudet kuitenkin vaihtelevat suuresti, ja monet eivät pysty sellaisenaan toimimaan Internetin osina. Erityisvälitysverkot ovat ympäristöjä, jossa Internetin protokollat eivät enää pysty toimimaan. Työn päämääränä oli kehittää rajapinta ja sen toteuttava ohjelmistokomponentti, jolla viestintäsovelluksia voidaan sovittaa viestintään haasteellisten verkkojen yli. Sähköposti valittiin rajapinnaksi sen soveltuvuuden ja standardiaseman vuoksi.

Työhön liittyen kehitettiin sulautettava, luotettava ja siirrettävä ohjelmistokomponentti, joten ohjelmiston toteutuksen suunnitteluun kiinnitettiin paljon huomiota. Ohjelmiston rakenteesta tehtiin tehokas ja suoraviivainen. Ohjelmiston kehityksen yhteydessä sen toteutusta testattiin ohjelmiston laadun ja yhteensopivuuden varmistamiseksi.

Työssä tehty toteutus on pääosin standardien mukainen. Toteutetut osat toimivat testien pohjalta standardien mukaisesti ja asiakasohjelmat pystyivät niitä käyttämään. Ohjelmiston yksinkertaistamiseksi ja kehityksen nopeuttamiseksi osia standardista oli jätetty toteuttamatta. Nämä ovat kuitenkin yhteensopivuuden nimissä kannattavia toteuttaa. Protokollien hankalampien kohtien ratkaisut aiheuttivat ongelmia, jotka tulivat ilmi sovellusyhteensopivuustesteissä. Näitä tulisi joko toteuttaa uudelleen tavalla joka olisi paremmin yhteensopiva tai etsiä kiertotapoja joilla asiakasohjelmistot saisi välttämään ongelmatapaukset.

Peittoaluetestien parantaminen vaatii virhetilanteiden luomista. Lisäksi käyttämättömäksi jäänyttä lähdekoodia voidaan poistaa lähdekooditiedostoista. Testauksessa Gprof-työkalu havaittiin riittämättömäksi suorituskyvyn analysointiin. Sille on

etsittävä vaihtoehto. Kokonaisuutena testien pohjalta voidaan todeta lähdekoodin laadun olevan tyydyttävä ja jatkokehityskelpoinen.

Jatkokehityksessä tulisi myös paneutua ohjelmiston skaalautumiskyvyn tutkimiseen. Erityisesti IMAP-protokolla havaittiin suorituskykytesteissä osassa sekä prosessoritehon että järjestelmäresurssien rajoittamaksi. Sulautettuna osana järjestelmälle voi aiheutua huonosti suoriutuvasta rajapinnasta haitallisia määriä kuormitusta, erityisesti automatisoitujen sovellusten, kuten sensoriverkkojen toimesta. Jatkokehityksessä tulisi analysoida tapoja keventää toteutuksen toimintamallia tai miten asiakasohjelmistot pystyvät näitä kulmatapauksia välttämään.

Yhden prosessin palvelinratkaisu on yksinkertainen, mutta raskaat operaatiot voivat kasvattaa palvelimen vasteajat merkittäviksi ja siten aiheuttaa suorituskyvyn heikkenemisen. Esimerkiksi viestien käsittelyn tiedosto-operaatiot voivat aiheuttaa huomattavia viiveitä protokollan muille osille. Tätä voi keventää siirtämällä postilaatikoiden käsittelyt omaan säikeeseen ja sopimalla tehtävänannolle ja postilaatikoiden tietorakenteiden lukitsemiselle sopivat mekanismit. Tämä tosin vaatii erillisen viestintämenetelmän kehittämiselle säikeiden välille, ja monimutkaistaa tilakoneita.

Kehityksessä käytettiin myös yksinkertaisia tietorakenteita järjestelmien toteuttamiseen. Näiden skaalautuminen on yleensä $O(N^2)$ luokkaa. Käytettyjen tietorakenteiden vaikutus tulisi selvittää profiloimalla suoritusta. Tietorakenteiden vaihtaminen skaalautuvampiin ei ole aina tehokkaampaa, koska se kasvattaa ohjelman monimutkaisuutta.

Viestien säilöntä Maildir-rakenteessa on pääosin käytännöllistä ja tehokasta, mutta asettaa merkittävän painon käyttöjärjestelmän tiedostojärjestelmälle. Yksittäisillä tiedostoilla on aina jonkin verran kiinteitä resurssikustannuksia (pienin levytilan varausyksikkö on usein 4 kiB), joiden suhde säilöttyyn dataan nähden nousee jos säilöttävät tiedostot ovat pääosin pieniä. Vaihtoehtona tälle, viestit voidaan tallentaa tietokantaan. Potentiaalinen hyöty ei ole ilmiselvä, ja muutos ohjelmistoon on merkittävä. Levykuormaa voi myös pyrkiä vähentämään säilyttämällä osia postilaatikoista ajon aikana muistissa.

Sovelluksen siirtäminen käytännössä UNIX-pohjaisista ympäristöistä muualle ei mahtunut työn aikatauluun. Suunnitteluvalinnat tehtiin siten, että siirtymän pitäisi olla mahdollinen, mutta tästä saadaan käytännön tuloksia vasta jatkokehityksessä.

Viitteet

[1] Kevin Fall. A Delay-Tolerant Network Architecture for Challenged Internets. In *SIGCOMM*, 2003. Saatavissa: `http://www.sigcomm.org/sigcomm2003/papers/p27-fall.pdf`.

[2] Scott Burleigh, Vinton Cerf, Robert Durst, Kevin Fall, Adrian Hooke, Keith Scott, and Howard Weiss. The Interplanetary Internet: A communications infrastructure for Mars exploration. In *IAC-02*, 2002. Saatavissa: `http://www.ipnsig.org/reports/IAF-Oct-2002.pdf`.

[3] Janet Abbate. *Inventing the Internet*. MIT Press, Cambridge (Mass), 1999. ISBN 978-0262011723. 268 s.

[4] J. Postel. Internet Protocol. RFC 791 (Standard), September 1981. Saatavissa: `http://www.ietf.org/rfc/rfc791.txt`. Updated by RFC 1349.

[5] Liikenne ja viestintäministeriö. Internet-liityntämuodot, 2009. Saatavissa: `http://www.laajakaistainfo.fi/teknologiat/index.php`. Viitattu `6.10.2009`.

[6] International Telecommunication Union. *Series G: Transmission systems and media, digital systems and networks, Asymmetric digital subscriber line (ADSL) tranceivers, ITU-T Recommendation G.992.1*. International Telecommunication Union, 1999.

[7] International Telecommunication Union. *Series G: Transmission systems and media, digital systems and networks, Single-pair high-speed subscriber line (SHDSL) tranceivers, ITU-T Recommendation G.991.2*. International Telecommunication Union, 2003.

[8] International Telecommunication Union. *Series J: Transmission of television, sound programme and other multimedia signals, Transmission systems for interactive cable television services, ITU-T Recommendation J.122*. ITU, 1998.

[9] International Telecommunication Union. *Series G: Transmission systems and media, digital systems and networks, Phoneline networking tranceivers - Foundation, ITU-T Recommendation G.989.1.* ITU, 2001.

[10] IEEE Standard for Information technology–Telecommunications and information exchange between systems–Local and metropolitan area networks–Specific requirements Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications - Section Three. *IEEE Std 802.3-2008 (Revision of IEEE Std 802.3-2005)*, pages 1–315, 2008. ISBN 07381-5797-9.

[11] Supplement To IEEE Standard For Information Technology- Telecommunications And Information Exchange Between Systems- Local And Metropolitan Area Networks- Specific Requirements- Part 11: Wireless LAN Medium Access Control (MAC) And Physical Layer (PHY) Specifications: Higher-speed Physical Layer Extension In The 2.4 GHz Band. *IEEE Std 802.11b-1999*, pages 1–90, 2000. ISBN 0-7381-1811-7.

[12] Larry L. Peterson and Bruce S. Davie. *Computer networks : a systems approach.* Morgan Kaufmann, San Francisco (CA), 2003. ISBN 1-55860-832-X (sid.).

[13] Michel Mouly and Marie-Bernadette Pautet. *The GSM system for mobile communications.* Mouly and Pautet, Palaiseau, 1992. ISBN 2-9507190-0-7.

[14] Heikki Kaaranen. *UMTS networks : architecture, mobility and services.* Wiley, Chichester, 2005. ISBN 0-470-01103-3.

[15] Regis J. Bates. *Broadband telecommunications handbook, 2nd ed.* McGraw-Hill, New York, 2002. ISBN 0-07-139851-1.

[16] Iridium Satellite Data Services White Paper. Technical report, Iridium Satellite, LLC., 2006. Saatavissa: `http://www.blueoceans.ca/uploads/solutions/1/IridiumDataServicesWhiteP% aper.pdf`. Viitattu `22.10.2009`.

[17] Margaret M McMahon. Measuring latency in Iridium Satellite Constellation data services. Technical report, ISDI Technologies, Inc., 2005. Saatavissa: `http://www.dodccrp.org/events/10th_ICCRTS/CD/papers/233.pdf`. Viitattu `26.10.2009`.

[18] IEEE Standard for Local and metropolitan area networks- Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems Amendment 2: Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands and Corrigendum 1. *IEEE Std 802.16-2005*, pages 1–822, 2006. ISBN 0-7381-4857-1.

[19] OFDM for Mobile Data Communications. Technical report, Flarion Technologies, Inc., 2003.

[20] The European Telecommunications Standards Institute (ETSI), TETRA03, Sophia Antipolis, France. *EN 300 392-1 - Terrestrial Trunked Radio (TETRA); Voice plus Data (V+D); Part 1: General network design, version 1.4.1*, 2009.

[21] Andrew S. Tanenbaum. *Computer Networks (International Edition)*. Prentice Hall, fourth edition, August 2002. ISBN 0130384887.

[22] Timo Leppinen. Viestintäverkkojen kehitys. Technical report, Telehallintokeskus, 2002. Saatavissa: `http://www.ficora.fi/attachments/suomiry/1158858972592/Viestintaverkko% jenkehitys_raportti.pdf`.

[23] Heikki E. Heinonen. *Tiimissä hamssiksi - Radioamatööritekniikan perusteita*. Suomen Radioamatööriliitto ry, 1997.

[24] Defense Information Systems Agency. Interoperability Standard for Digital Message Transfer Device Subsystems. Technical report, Department of Defense, 1993.

[25] R. Dean Straw, editor. *The ARRL Handbook for Radio Amateurs 2000*. ARRL - the national association for Amateur Radio, 77th edition, 1999.

[26] The European Telecommunications Standards Institute (ETSI), Sophia Antipolis, France. *Digital Video Broadcasting (DVB); Implementation guidelines for the use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream*, May 2004.

[27] International Standard, ISO/IEC/JTC1/SC29 WG11. *ISO/IEC 13818-1, Information technology – Generic coding of moving pictures and associated audio information: Systems*. International Organization for Standardization, Geneva, Switzerland, 1998. 182 s.

[28] United States of America, Department of Defense, Washington, DC. *Global Positioning System Precise Positioning Service Performance Standard*, February 2007.

[29] Fact Sheet 1: EGNOS explained. Technical report, European Space Agency, 2005. Viitattu 20.10.2009.

[30] Bluetooth SIG, Bellevue, Washington, USA. *Bluetooth specification Version 2.1 + EDR*, 2007. Saatavissa: http://www.bluetooth.com/NR/rdonlyres/F8E8276A-3898-4EC6-B7DA-E5535258% B056/6545/Core_V21__EDR.zip.

[31] IEEE Standard for Information technology- Telecommunications and information exchange between systems- Local and metropolitan area networks- Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). *IEEE Std 802.15.4-2006*, 2006. ISBN 0-7381-4996-9.

[32] Demystifying 802.15.4 and ZigBee. Technical report, Digi International, Inc., 2008.

[33] Stuart Williams and Iain Millar. The IrDA Platform. Technical report, HP Laboratories, Bristol, 1995. Saatavissa: http://www.irda.org/associations/2494/files/Publications/irda_platform% .pdf.

[34] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561 (Experimental), July 2003. Saatavissa: http://www.ietf.org/rfc/rfc3561.txt.

[35] Draft Standard for Information Technology - Telecommunications and Information Exchange Between Systems - LAN/MAN Specific Requirements - Part 11: Wireless Medium Access Control (MAC) and Physical layer (PHY) specifications: Amendment: Mesh Networking. *IEEE Unapproved draft P802.11s/D2.0*, 2008.

[36] Kay Römer and Friedemann Mattern. The Design Space of Wireless Sensor Networks. Technical report, Institute of Pervasive Computing, 2004. Saatavissa: http://www.vs.inf.ethz.ch/publ/papers/wsn-designspace.pdf. Viitattu 20.10.2009.

[37] K. Scott and S. Burleigh. Bundle Protocol Specification. RFC 5050 (Experimental), November 2007. Saatavissa: http://www.ietf.org/rfc/rfc5050.txt.

[38] Jukka Korpela. Datatekniikka ja viestintä, 2004. Saatavissa: `http://www.cs.tut.fi/~jkorpela/`. Viitattu `15.4.2009`.

[39] ISO. *ISO Standard 646, 7-Bit Coded Character Set for Information Processing Interchange*. International Organization for Standardization, Geneva, Switzerland, second edition, 1983. Saatavissa: `http://www.iso.ch/cate/d4777.html`. Also available as ECMA-6.

[40] The Unicode Consortium. *The Unicode Standard, Version 5.0*. Addison-Wesley Developers Press, Boston, MA, 2007. ISBN 0-321-48091-0.

[41] M.R. Horton. UUCP mail interchange format standard. RFC 976, February 1986. Saatavissa: `http://www.ietf.org/rfc/rfc976.txt`. Updated by RFC 1137.

[42] J. Klensin. Simple Mail Transfer Protocol. RFC 5321 (Draft Standard), October 2008. Saatavissa: `http://www.ietf.org/rfc/rfc5321.txt`.

[43] P. Resnick. Internet Message Format. RFC 5322 (Draft Standard), October 2008. Saatavissa: `http://www.ietf.org/rfc/rfc5322.txt`.

[44] N. Freed and N. Borenstein. Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. RFC 2045 (Draft Standard), November 1996. Saatavissa: `http://www.ietf.org/rfc/rfc2045.txt`. Updated by RFCs 2184, 2231, 5335.

[45] J. Palme. Common Internet Message Headers. RFC 2076 (Informational), February 1997. Saatavissa: `http://www.ietf.org/rfc/rfc2076.txt`.

[46] G. Klyne and J. Palme. Registration of Mail and MIME Header Fields. RFC 4021 (Proposed Standard), March 2005. Saatavissa: `http://www.ietf.org/rfc/rfc4021.txt`. Updated by RFC 5322.

[47] E. Hall. The application/mbox Media Type. RFC 4155 (Informational), September 2005. Saatavissa: `http://www.ietf.org/rfc/rfc4155.txt`.

[48] Dan J. Bernstein. Using maildir format, 2003. Saatavissa: `http://cr.yp.to/proto/maildir.html`. Viitattu `28.9.2009`.

[49] Sam Varshavchik. Maildir++, 2008. Saatavissa: `http://www.inter7.com/courierimap/README.maildirquota.html`. Viitattu `22.10.2009`.

[50] J. Myers. Local Mail Transfer Protocol. RFC 2033 (Informational), October 1996. Saatavissa: `http://www.ietf.org/rfc/rfc2033.txt`.

[51] Dan J. Bernstein. Quick Mail Transport Protocol (QMTP), 1997. Saatavissa: `http://cr.yp.to/proto/qmtp.txt`. Viitattu `9.10.2009`.

[52] K. Moore. Recommendations for Automatic Responses to Electronic Mail. RFC 3834 (Proposed Standard), August 2004. Saatavissa: `http://www.ietf.org/rfc/rfc3834.txt`. Updated by RFC 5436.

[53] K. Moore. Simple Mail Transfer Protocol (SMTP) Service Extension for Delivery Status Notifications (DSNs). RFC 3461 (Draft Standard), January 2003. Saatavissa: `http://www.ietf.org/rfc/rfc3461.txt`. Updated by RFCs 3798, 3885, 5337.

[54] J. Myers and M. Rose. Post Office Protocol - Version 3. RFC 1939 (Standard), May 1996. Saatavissa: `http://www.ietf.org/rfc/rfc1939.txt`. Updated by RFCs 1957, 2449.

[55] M. Crispin. INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1. RFC 3501 (Proposed Standard), March 2003. Saatavissa: `http://www.ietf.org/rfc/rfc3501.txt`. Updated by RFCs 4466, 4469, 4551, 5032, 5182.

[56] B. Leiba. IMAP4 Implementation Recommendations. RFC 2683 (Informational), September 1999. Saatavissa: `http://www.ietf.org/rfc/rfc2683.txt`.

[57] E. Burger and G. Parsons. LEMONADE Architecture - Supporting Open Mobile Alliance (OMA) Mobile Email (MEM) Using Internet Mail. RFC 5442 (Informational), March 2009. Saatavissa: `http://www.ietf.org/rfc/rfc5442.txt`.

[58] C. Feather. Network News Transfer Protocol (NNTP). RFC 3977 (Proposed Standard), October 2006. Saatavissa: `http://www.ietf.org/rfc/rfc3977.txt`.

[59] Sascha Segan. R.I.P Usenet: 1980-2008, 2008. Saatavissa: `http://www.pcmag.com/article2/0,2817,2326848,00.asp`. Viitattu `6.10.2009`.

[60] J. Oikarinen and D. Reed. Internet Relay Chat Protocol. RFC 1459 (Experimental), May 1993. Saatavissa: `http://www.ietf.org/rfc/rfc1459.txt`. Updated by RFCs 2810, 2811, 2812, 2813.

[61] Paul Mutton. *IRC Hacks*. O'Reilly Hacks, 2004. ISBN 0-596-00687-X.

[62] P. Saint-Andre. Extensible Messaging and Presence Protocol (XMPP): Core. RFC 3920 (Proposed Standard), October 2004. Saatavissa: `http://www.ietf.org/rfc/rfc3920.txt`.

[63] C. M. Sperberg-McQueen, Francois Yergeau, Eve Maler, Jean Paoli, and Tim Bray. Extensible Markup Language (XML) 1.0 (Fifth Edition). Technical report, W3C, 2008. Saatavissa: `http://www.w3.org/TR/2008/PER-xml-20080205`.

[64] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261 (Proposed Standard), June 2002. Saatavissa: `http://www.ietf.org/rfc/rfc3261.txt`. Updated by RFCs 3265, 3853, 4320, 4916, 5393.

[65] International Telecommunication Union. *Series H: Infrastructure of audiovisual services – Systems and terminal equipment for audiovisual services, Packet-based multimedia communications systems, ITU-T Recommendation H.323*. International Telecommunication Union, 2006.

[66] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 3550 (Standard), July 2003. Saatavissa: `http://www.ietf.org/rfc/rfc3550.txt`. Updated by RFC 5506.

[67] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (Draft Standard), June 1999. Saatavissa: `http://www.ietf.org/rfc/rfc2616.txt`. Updated by RFC 2817.

[68] J. Postel and J. Reynolds. File Transfer Protocol. RFC 959 (Standard), October 1985. Saatavissa: `http://www.ietf.org/rfc/rfc959.txt`. Updated by RFCs 2228, 2640, 2773, 3659.

[69] LLC. Napster. Napster, 2008. Saatavissa: `http://www.napster.com`. Viitattu `20.11.2009`.

[70] Ian Clarke. The Freenet Project, 2009. Saatavissa: `http://freenetproject.org`. Viitattu `20.11.2009`.

[71] M.T. Rose and D.E. Cass. ISO Transport Service on top of the TCP Version: 3. RFC 1006 (Standard), May 1987. Saatavissa: `http://www.ietf.org/rfc/rfc1006.txt`. Updated by RFC 2126.

[72] NATO, Brussels. *North Atlantic Treaty Organization (NATO) Standardization Agreement (STANAG) 4406, Military Message Handling System*, 1994.

[73] Combined Communications Electronic Board (CCEB), Allied Message Handling (AMH) International Subject Matter Experts (ISME) working group, US Military Comm.-Electronics Board, Washington, D.C. *Allied Communications Publication (ACP) 123, Common Messaging Strategy and Procedures*, 1994.

[74] Court of First Instance of the European Communities. Press Release No 63/07: Judgment of the Court of First Instance in Case T-201/04, Microsoft Corp. v Comission of the European Communities, 2007. Saatavissa: `http://curia.europa.eu/ en/ actu/ communiques/ cp07/ aff/ cp070063en.pdf`. Viitattu `6.10.2009`.

[75] The Open Group. *CAE Specification: DCE 1.1: Remote Procedure Call, Document Number: C706*. The Open Group, 1997.

[76] Microsoft Corp., Redmond, WA. *Remote Procedure Call Protocol Extensions*, 2008.

[77] Microsoft Corp., Redmond, WA. *Exchange Server Protocols Overview*, 2008.

[78] Microsoft Knowledge Base. KB 200018: Differences between CDO, Simple MAPI, and Extended MAPI, 2007. Saatavissa: `http://support.microsoft.com/kb/200018`. Viitattu `7.10.2009`.

[79] James D. Mooney. Bringing Portability to the Software Process. Technical report, West Virginia University, Dept. of Statistics and Computer Science, 1997. Saatavissa: `http://www.cs.wvu.edu/%7Ejdm/research/portability/reports/TR_97-1.pdf`. Viitattu `26.1.2009`.

[80] IEEE and The Open Group. *IEEE Std 1003.1, 2004 Edition : Standard for Information Technology - Portable Operating System Interface (POSIX)*. IEEE, 2004. ISBN 978-1931624437.

[81] Milo. Operating System Technical Comparison, 2007. Saatavissa: `http://www.osdata.com/`. Viitattu `16.4.2009`.

[82] Microsoft Corporation. Overview of the Windows API, 2009. Saatavissa: `http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winpr% og/winprog/overview_of_the_windows_api.asp`. Viitattu `17.4.2009`.

[83] James Gosling. *Java(TM) Language Specification, The (3rd Edition)*. Addison-Wesley, 2005. ISBN 978-0321246783. 688 s.

[84] Red Hat Inc. Cygwin Information and Installation, 2008. Saatavissa: `http://cygwin.com/`. Viitattu `20.4.2009`.

[85] Microsoft Corporation. Welcome to Subsystem for UNIX-based Applications, 2006. Saatavissa: `http://technet.microsoft.com/en-us/library/cc779522.aspx`. Viitattu `20.4.2009`.

[86] Fabrice Bellard. QEMU - Open source processor emulator, 2009. Saatavissa: `http://www.qemu.org`. Viitattu `27.10.2009`.

[87] Wine authors. Wine - Wine Is Not an Emulator, 2009. Saatavissa: `http://www.winehq.org`. Viitattu `27.10.2009`.

[88] User-mode Linux, 2009. Saatavissa: `http://user-mode-linux.sourceforge.net/`. Viitattu `27.10.2009`.

[89] Inc. VMware. VMware Business Infrastructure Virtualization, 2009. Saatavissa: `http://www.vmware.com`. Viitattu `27.10.2009`.

[90] Inc. Sun Microsystems. VirtualBox, 2009. Saatavissa: `http://www.virtualbox.org`. Viitattu `27.10.2009`.

[91] Dan Kegel. The C10K problem, 2006. Saatavissa: `http://www.kegel.com/c10k.html`. Viitattu `20.10.2008`.

[92] Jukka Manner. S-38.1600 UNIX-sovellusohjelmointi, opetusmateriaali. Technical report, Teknillinen korkeakoulu, 2009.

[93] George Varghese. *Network Algorithmics - An interdisciplinary approach to designing fast network devices*. Elsevier / Morgan Kaufmann, 2005. ISBN 0-12-088477-1.

[94] Dan J. Bernstein. The qmail security guarantee, 2005. Saatavissa: `http://cr.yp.to/qmail/guarantee.html`. Viitattu `17.4.2009`.

[95] M. Gahrns. IMAP4 Multi-Accessed Mailbox Practice. RFC 2180 (Informational), July 1997. Saatavissa: `http://www.ietf.org/rfc/rfc2180.txt`.

[96] Kshirasagar Naik and Priyadarshi Tripathy. *Software Testing and Quality Assurance*. Wiley, Hoboken, New Jersey, 2008. ISBN 978-0-471-78911-6.

[97] Free Software Foundation, Inc., Boston, MA. *Using the GNU Compiler Collection (GCC)*, 2008.

[98] Valgrind Developers. Valgrind - instrumentation framework for building dynamic analysis tools, 2009. Saatavissa: `http://valgrind.org/`. Viitattu `23.10.2009`.

[99] Debian Project. Debian - The Universal Operating System, 2009. Saatavissa: `http://www.debian.org`. Viitattu `8.10.2009`.

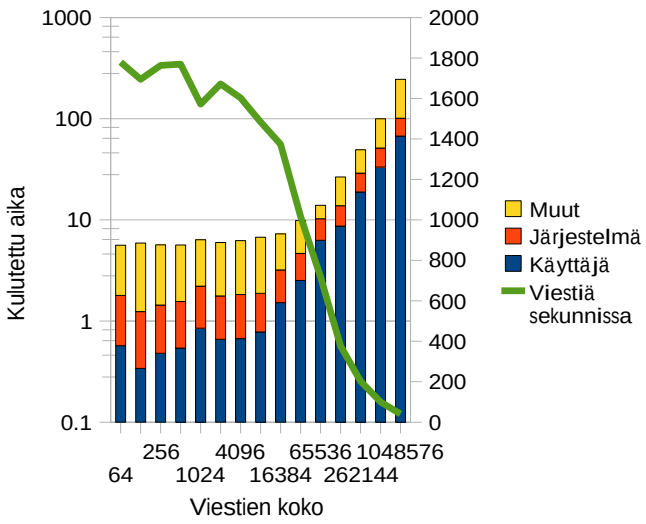
[100] Linux Kernel Organization Inc. The Linux Kernel Archives, 2009. Saatavissa: `http://www.kernel.org`. Viitattu `8.10.2009`.

[101] Free Software Foundation Inc. The GNU Operating System, 2009. Saatavissa: `http://www.gnu.org`. Viitattu `8.10.2009`.

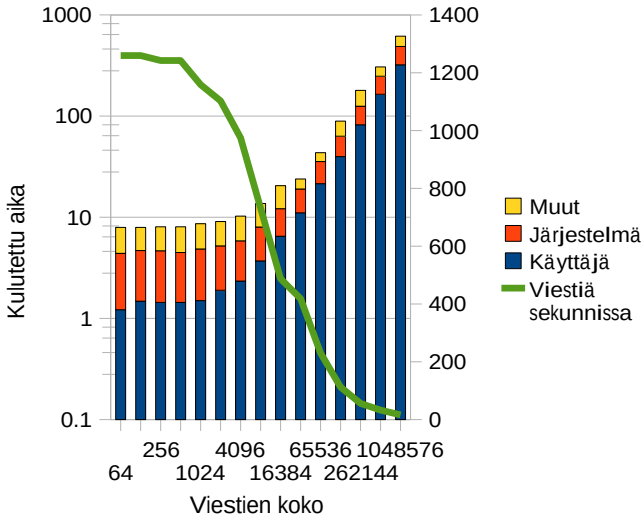
[102] Niels Provos. Libevent - an event notification library, 2009. Saatavissa: `http://monkey.org/~provos/libevent/`. Viitattu `12.6.2009`.

Liite A

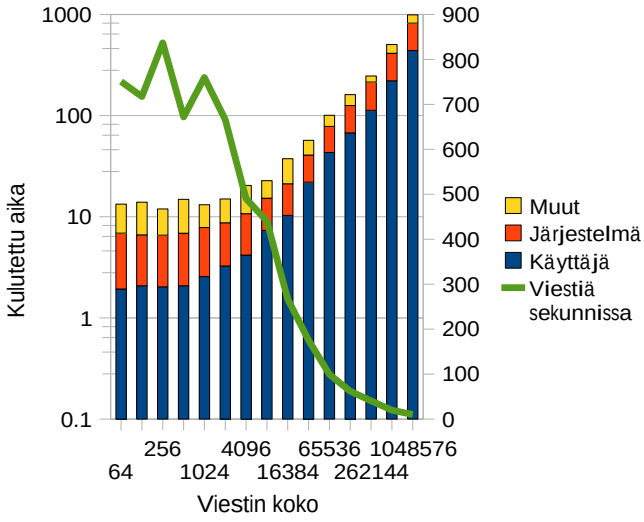
Suorituskykytestit



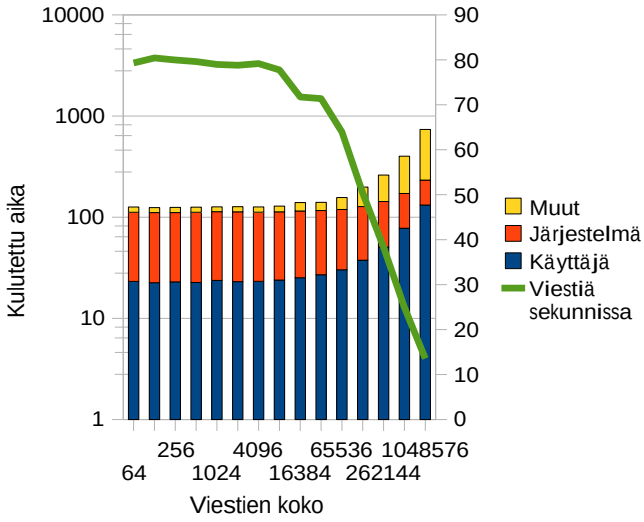
Kuva A.1: SMTP-suorituskyky, Intel Core i7, kun lähetetään 10000 viestiä. Viestien kokoa muutettiin 64 merkistä 1048576 merkkiin. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään siirtonopeutta viesteinä sekunnissa.



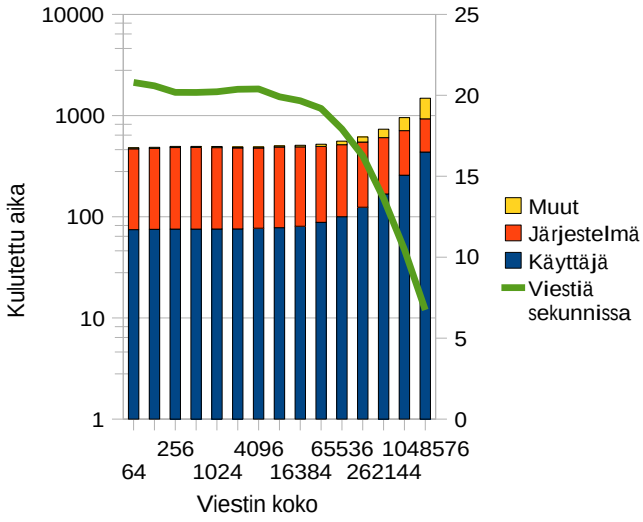
Kuva A.2: SMTP-suorituskyky, Intel Atom 330, kun lähetetään 10000 viestiä. Viestien kokoa muutettiin 64 merkistä 1048576 merkkiin. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään siirtonopeutta viesteinä sekunnissa.



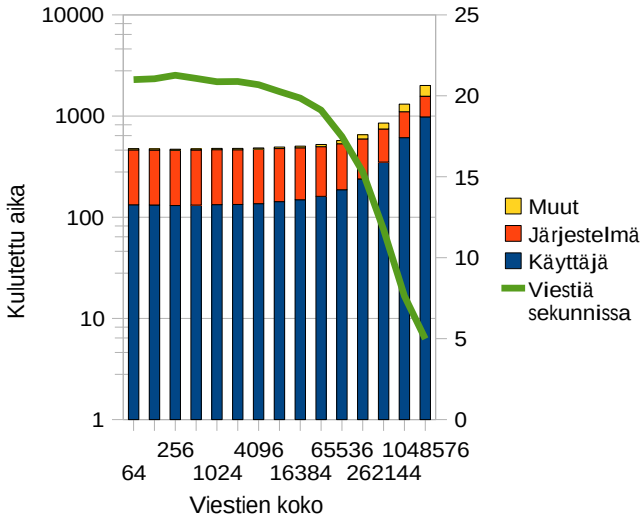
Kuva A.3: SMTP-suorituskyky, VIA C7-M, kun lähetetään 10000 viestiä. Viestien kokoa muutettiin 64 merkistä 1048576 merkkiin. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään siirtonopeutta viesteinä sekunnissa.



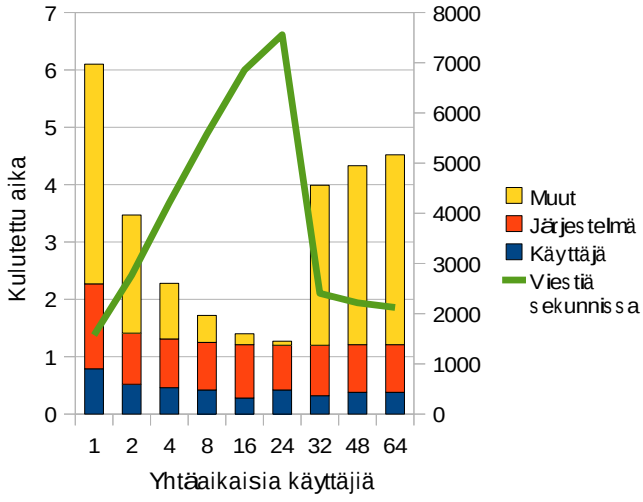
Kuva A.4: IMAP-suorituskyky, Intel Core i7, kun luetaan 10000 viestiä. Viestien kokoa muutettiin 64 merkistä 1048576 merkkiin. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään siirtonopeutta viesteinä sekunnissa.



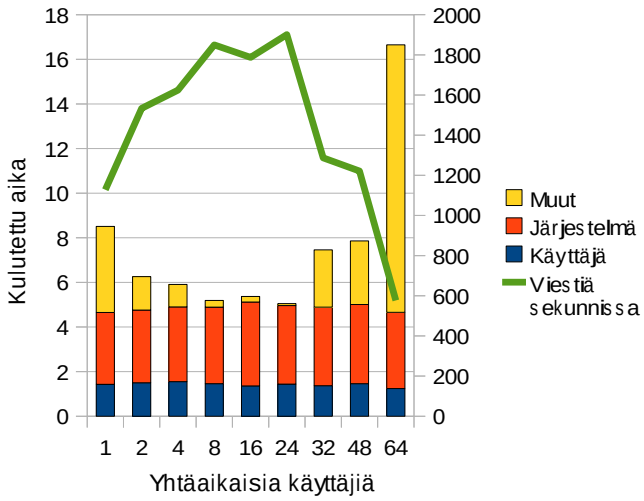
Kuva A.5: IMAP-suorituskyky, Intel Atom 330, kun luetaan 10000 viestiä. Viestien kokoa muutettiin 64 merkistä 1048576 merkkiin. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään siirtonopeutta viesteinä sekunnissa.



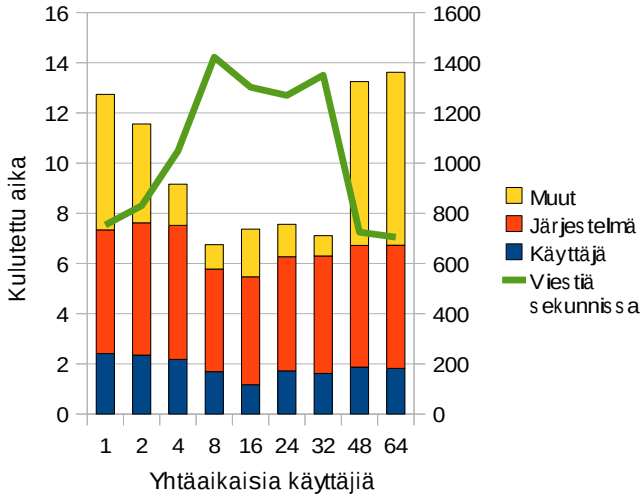
Kuva A.6: IMAP-suorituskyky, VIA C7-M, kun luetaan 10000 viestiä. Viestien kokoa muutettiin 64 merkistä 1048576 merkkiin. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään siirtonopeutta viesteinä sekunnissa.



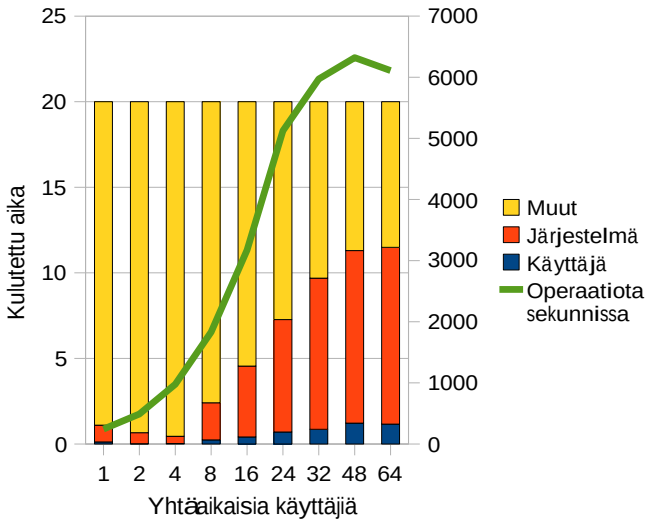
Kuva A.7: SMTP-suorituskyky, Intel Core i7, kun lähetetään yhteensä 9600 viestiä. Testiajot tehtiin enimmillään 64 yhtäaikaisella käyttäjäyhteydellä. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään siirtonopeutta viesteinä sekunnissa.



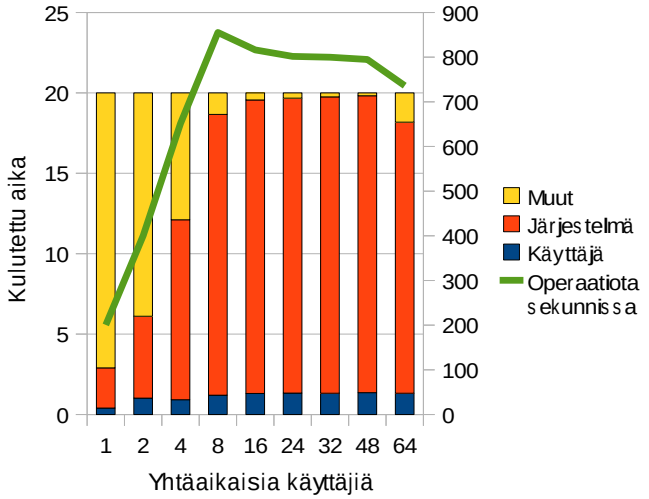
Kuva A.8: SMTP-suorituskyky, Intel Atom 330, kun lähetetään yhteensä 9600 viestiä. Testiajot tehtiin enimmillään 64 yhtäaikaisella käyttäjäyhteydellä. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään siirtonopeutta viesteinä sekunnissa.



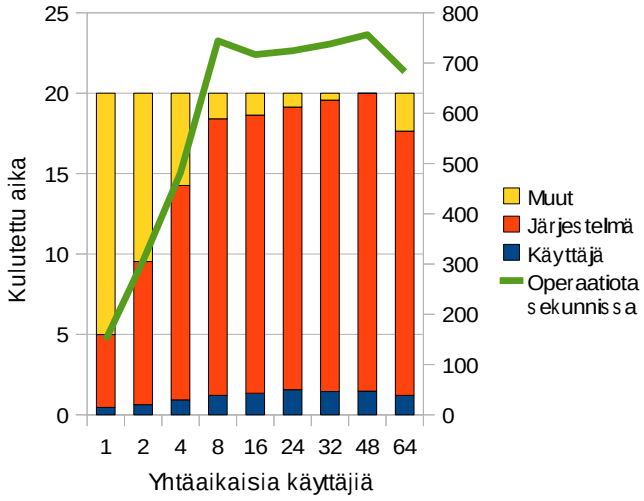
Kuva A.9: SMTP-suorituskyky, VIA C7-M, kun lähetetään yhteensä 9600 viestiä. Testiajot tehtiin enimmillään 64 yhtäaikaisella käyttäjäyhteydellä. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään siirtonopeutta viesteinä sekunnissa.



Kuva A.10: IMAP-suorituskyky, Intel Core i7, kun suoritetaan 20 sekuntia satunnaisia käskyjä. Testiajot tehtiin enimmillään 64 yhtäaikaisella käyttäjäyhteydellä. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään suoritusnopeutta operaatioina sekunnissa.



Kuva A.11: IMAP-suorituskyky, Intel Atom 330, kun suoritetaan 20 sekuntia satunnaisia käskyjä. Testiajot tehtiin enimmillään 64 yhtäaikaisella käyttäjäyhteydellä. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään suoritusnopeutta operaatioina sekunnissa.



Kuva A.12: IMAP-suorituskyky, VIA C7-M, kun suoritetaan 20 sekuntia satunnaisia käskyjä. Testiajot tehtiin enimmillään 64 yhtäaikaisella käyttäjäyhteydellä. Pylväät esittävät käytettyä suoritusaikaa, joka on jaettu käyttäjä-, järjestelmäaikaan ja muuhun käytettyyn aikaan. Viivalla esitetään suoritusnopeutta operaatioina sekunnissa.

Liite B

Protokollayhteensopivuus testi

```
$ ./imaptest user=xxxxxxxx host=localhost port=xxxx pass=xxxx test=tests
*** Test expunge2 command 17/19 (line 42)
 - failed: Expected tagged reply 'ok', got 'NO search not implemented'
 - Command (tag 2.11): search all

*** Test store command 5/13 (line 19)
 - failed: Missing 2 untagged replies (2 mismatches)
 - first unexpanded: 2 fetch ($!unordered=2 flags ($!ignore=\recent
 $!noextra $!unordered \seen $$hello $$world))
 - first expanded: 2 fetch ( flags (   \seen $hello $world))
 - best match: 2 FETCH (FLAGS (\Seen))
 - Command (tag 10.19): store 2,4 +flags ($$hello $$world)

*** Test select command 5/9 (line 18)
 - failed: Missing 1 untagged replies (1 mismatches)
 - first unexpanded: status $mailbox ($!unordered=2 messages 2 recent 2
 uidnext $uidnext uidvalidity $uidvalidity unseen 2)
 - first expanded: status imaptest ( messages 2 recent 2 uidnext 3
 uidvalidity 1256296965 unseen 2)
 - best match: status imaptest (MESSAGES 2 UNSEEN 2 RECENT 0 UIDVALIDITY
 1256296965 UIDNEXT 3)
 - Command (tag 12.13): status $mailbox (messages recent uidnext uidvalidity
 unseen)

*** Test select command 6/9 (line 22)
 - failed: Missing 2 untagged replies (3 mismatches)
 - first unexpanded: 2 recent
 - first expanded: 2 recent
```

```
 - best match: 2 EXISTS
 - Command (tag 12.14): select $mailbox

*** Test copy command 21/25 (line 45)
 - failed: Missing 1 untagged replies (1 mismatches)
 - first unexpanded: 4 fetch ($!unordered=2 flags ($!ignore=\recent
 $!noextra $!unordered \recent \flagged $$keyword1 $$keyword2) internaldate
 $date5)
 - first expanded: 4 fetch ( flags (   \recent \flagged $keyword1 $keyword2)
 internaldate 22-Feb-2008 17:06:23 +0200)
 - best match: 4 FETCH (FLAGS (\Flagged \Recent) INTERNALDATE "22-Feb-2008
 17:06:23 +0200")
 - Command (tag 20.10): fetch 4 (flags internaldate)

Info: 11 test groups: 4 failed, 1 skipped due to missing capabilities
Info: base protocol: 5/175 individual commands failed
Info: extensions: 0/0 individual commands failed
```