Liite 1. Ryhmän sisäinen ohjelmointiohje

Elementtien nimeäminen, kirjoittaminen ja kokoaminen

Paketit

Nimet ovat lyhyitä (yksisanaisia), ja ne kirjoitetaan pienellä alkukirjaimella.

Hierarkia Javan omissa paketeissa on *java.** tai *javax.**.

Ulkopuoliset osat nimetään domain-nimen mukaan, esimerkiksi *org.xml.**; Nippu-projektilla nimetään kuitenkin vain *nippu.**.

Samanlaiset luokat kerätään samaan pakettiin.

Luokat

Nimet alkavat isolla kirjaimella, esimerkiksi *Calendar*.

Monisanaiset kirjoitetaan yhteen, esimerkiksi *LinkedList*.

Käytetään mahdollisimman kuvaavia, kuitenkin järkevän pituisia nimiä.

Useimmiten kannattaa toteuttaa ainakin *equals* ja *toString*, ehkä *hashCode*.

Käytetään get- ja set-metodeja mieluummin kuin julkisia muuttujia.

Metodit

Nimet alkavat pienellä kirjaimella, esimerkiksi *equals, getTime*.

Nimeämisessä kannattaa pyrkiä kuvaavuuteen.

Vakiot

Nimet kirjoitetaan kokonaan isolla, esimerkiksi *DEFAULTFILE*.

Muuttujat

Nimet alkavat pienellä kirjaimella.

Luettavuutta helpottaa se, että muuttujat esitellään heti koodin alussa eikä keskellä koodia.

Ohjelmakoodin kommentointi

Javadoc-kommentointi aloitetaan tagilla /** ja lopetetaan tagilla */. Jokainen kommentti-rivi aloitetaan *-merkillä. Kommentti liittyy aina seuraavaan ohjelmaelementtiin (esimer-

kiksi luokkaan, metodiin, vakioon). Kommentoinnissa voi käyttää normaalia HTML-koodausta. Normaalin tekstin ja HTML-koodauksen lisäksi voidaan antaa @-tageilla erityismerkityksiä kommenteille. Alla on lista @-tageista, joita tulisi käyttää eri elementtien yhteydessä:

Luokat

@author: luokan tekijä

@version: luokan versionumero

Metodit

@param: parametrit, annetaan muodossa "*@param parametrin_nimi selitys*"

@return: mitä arvoja metodi palauttaa

@throws: mahdolliset poikkeukset, annetaan muodossa "*@throws poikkeuksen_nimi*"

Lisäksi esimerkiksi seuraavia @-tageja voidaan tarvittaessa käyttää:

@see: lisätietoja. Tekee dokumenttiin kentän "See also". Annetaan muodossa "*@see <kohde dokumentoinnissa tai html-linkki>*".

@deprecated: Mistä lähtien koodi on vanhentunut

Javadocin käyttö

Komento on muotoa:

```
javadoc [options] [packagenames] [sourcefiles] [classnames] [@files]
```

Suositeltava muoto Nipun Java-lähdekoodista dokumenttia tehtäessä:

```
javadoc -private -linksource -d /home/group/nippu/javadocs/ -classpath
/home/group/nippu/koodit/ paketit
```

Javadoc ei tue muutoksiin perustuvaa dokumentin luontia tai yksittäisten pakettien lisäämistä erikseen. Koko dokumentti on siis luotava aina kerralla. Kohtaan *paketit* on siis lueteltava kaikki käytössä olevat Java-paketit, esimerkiksi *nippu.reception*, *nippu.printingmanagement* ja niin edelleen.

Hyödyllisiä komentoriviparametreja:

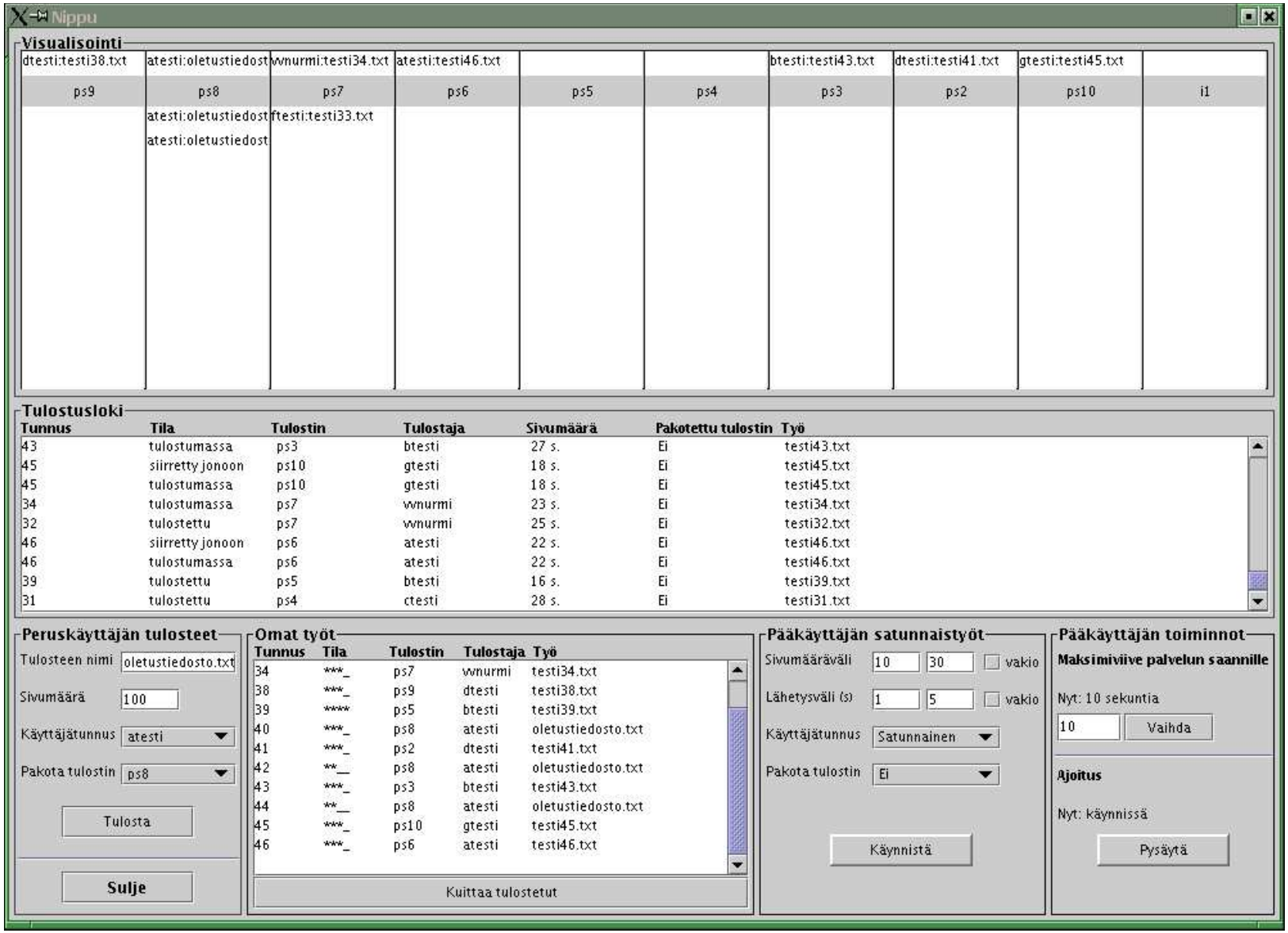
-help	luettelee kaikki mahdolliset parametrit
-private	tekee dokumentin kaikista (myös luokan sisäisistä) metodeista, käytettävä sisäisessä dokumentoinnissa
-protected	tekee dokumentin ainoastaan protected- ja public-metodeista, suositeltava jos haluaa julkaista luokan ulkopuolisille (oletus)
-d	määrittelee hakemiston, jonne dokumentointi tehdään
-classpath	tällä määritellään, missä käyttäjän omat luokat ovat
-linksource	tekee HTML-dokumentin myös itse koodista

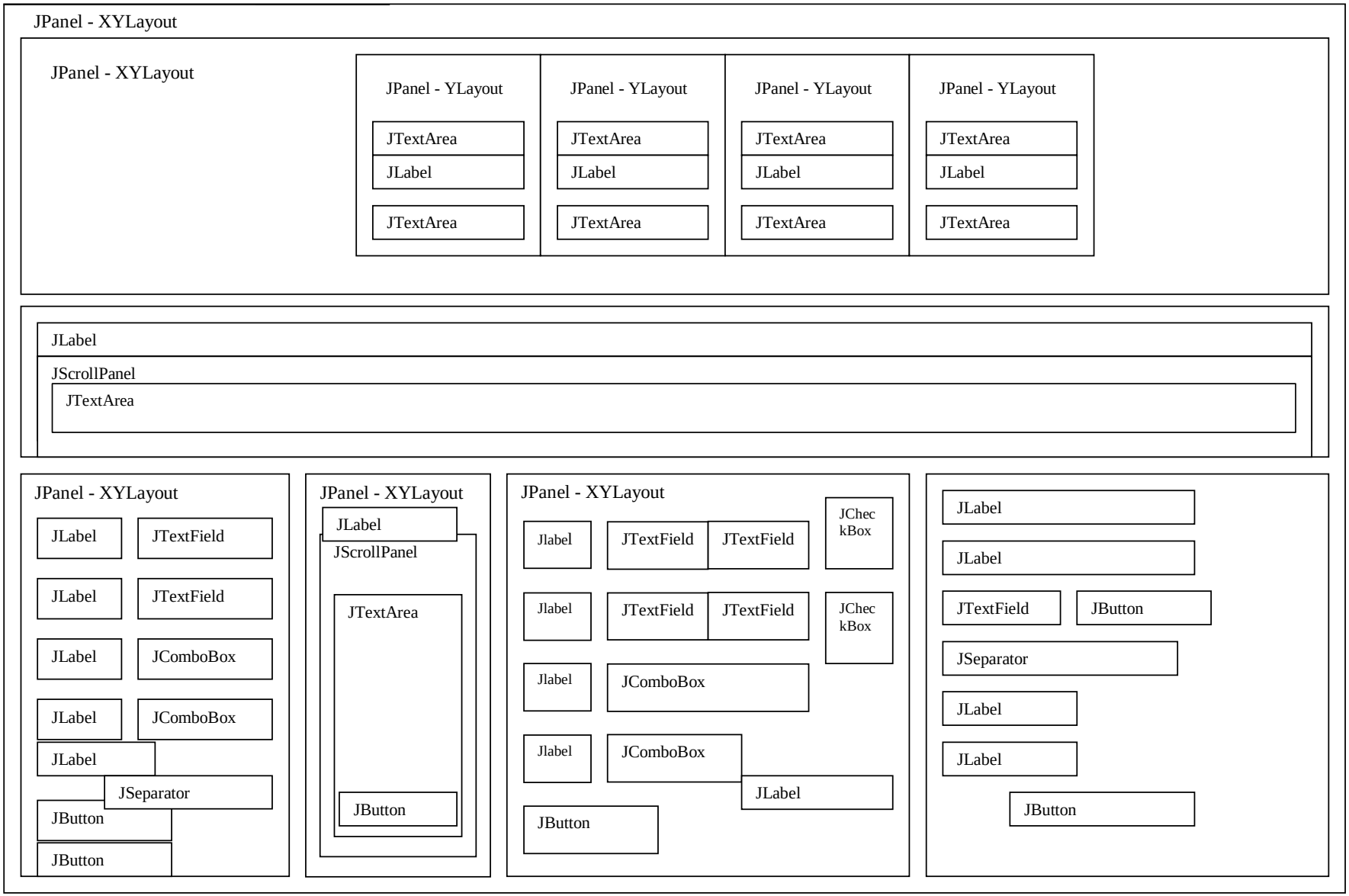
Liite 2. Pääkäyttäjän käyttöliittymä

Seuraavalla sivulla on kuva Nipun pääkäyttäjän käyttöliittymästä. Käyttöliittymän graafinen ulkoasu (esimerkiksi värit, kirjasimet) voi olla erilainen eri käyttöympäristöissä, mutta kuvasta ilmenevät käyttöliittymän toiminnot ja niiden sijainnit.

Käyttöliittymän kuvan jälkeen on Swing-komponenttikaavio, josta ilmenevät vastaavat Swing-komponentit.

Käyttöliittymästä kerrotaan liittymäkuvauksessa luvussa 3.3.1.


Nippu
Visualisointi
dtesti:testi38.txt
atesti:oletustiedost
wnurmi:testi34.txt
atesti:testi46.txt
btesti:testi43.txt
dtesti:testi41.txt
gtesti:testi45.txt
ps9
ps8
ps7
ps6
ps5
ps4
ps3
ps2
ps10
i1
atesti:oletustiedost
ftesti:testi33.txt
atesti:oletustiedost
Tulostusloki
Tunnus
Tila
Tulostin
Tulostaja
Sivumäärä
Pakotettu tulostin
Työ
43 tulostumassa ps3 btesti 27 s. Ei testi43.txt
45 siirretty jonoon ps10 gtesti 18 s. Ei testi45.txt
45 tulostumassa ps10 gtesti 18 s. Ei testi45.txt
34 tulostumassa ps7 wnurmi 23 s. Ei testi34.txt
32 tulostettu ps7 wnurmi 25 s. Ei testi32.txt
46 siirretty jonoon ps6 atesti 22 s. Ei testi46.txt
46 tulostumassa ps6 atesti 22 s. Ei testi46.txt
39 tulostettu ps5 btesti 16 s. Ei testi39.txt
31 tulostettu ps4 ctesti 28 s. Ei testi31.txt
Peruskäyttäjän tulosteet
Tulosteen nimi
oletustiedosto.txt
Sivumäärä
100
Käyttäjätunnus
atesti
Pakota tulostin
ps8
Tulosta
Sulje
Omat työt
Tunnus
Tila
Tulostin
Tulostaja
Työ
34 ***_ ps7 wnurmi testi34.txt
38 ***_ ps9 dtesti testi38.txt
39 **** ps5 btesti testi39.txt
40 ***_ ps8 atesti oletustiedosto.txt
41 ***_ ps2 dtesti testi41.txt
42 **__ ps8 atesti oletustiedosto.txt
43 ***_ ps3 btesti testi43.txt
44 **__ ps8 atesti oletustiedosto.txt
45 ***_ ps10 gtesti testi45.txt
46 ***_ ps6 atesti testi46.txt
Kuittaa tulostetut
Pääkäyttäjän satunnaistyöt
Sivumääräväli
10
30
vakio
Lähetysväli (s)
1
5
vakio
Käyttäjätunnus
Satunnainen
Pakota tulostin
Ei
Käynnistä
Pääkäyttäjän toiminnot
Maksimiviive palvelun saannille
Nyt: 10 sekuntia
10
Vaihda
Ajoitus
Nyt: käynnissä
Pysäytä


JPanel - XYLayout
JPanel - XYLayout
JPanel - YLayout
JTextArea
JLabel
JTextArea
JPanel - YLayout
JTextArea
JLabel
JTextArea
JPanel - YLayout
JTextArea
JLabel
JTextArea
JPanel - YLayout
JTextArea
JLabel
JTextArea
JLabel
JScrollPanel
JTextArea
JPanel - XYLayout
JLabel
JTextField
JLabel
JTextField
JLabel
JComboBox
JLabel
JComboBox
JLabel
JSeparator
JButton
JButton
JPanel - XYLayout
JLabel
JScrollPanel
JTextArea
JButton
JPanel - XYLayout
Jlabel
JTextField
JTextField
JChec kBox
Jlabel
JTextField
JTextField
JChec kBox
Jlabel
JComboBox
Jlabel
JComboBox
JLabel
JButton
JLabel
JLabel
JTextField
JButton
JSeparator
JLabel
JLabel
JButton

Liite 3. Analysoijaan liittyvää filosofista pohdintaa

Osa A

Palvelupyyntöavaruuden suhde kustannuksiin

Terttu tarjoaa analysoijalle kaksi tapaa ilmaista tulostustöiden samankaltaisuutta. Nämä ovat töiden yhdisteen kustannus suhteessa töiden erillisten kustannusten summaan ja töiden välinen etäisyys palvelupyyntöavaruudessa[1]. Näiden kahden tavan roolit eroavat toisistaan. Eräs palvelupyyntöavaruuden teoreettinen hyöty tulee esille, kun tarkastellaan Terttua yleisesti ajoittajana ja unohdetaan Nippuun liittyvät tulosteet ja tulostimet. Tertun palvelupyyntöavaruudessa lähekkäiset pisteet ovat palvelupyyntöjä, jotka muistuttavat paljon toisiaan. Avaruudesta voidaan valita myös sellaisia pisteitä, jotka eivät vastaa yhtään ajoittajalle annettua palvelupyyntöä. Nämä pisteet voidaan mieltää jonkinlaisiksi kompromisseiksi pyydetyistä palveluista. Piste, joka on hyvin lähellä useita palvelupyyntöjä olematta kuitenkaan yksikään niistä, voidaan mieltää hyväksi kompromissiksi näistä pyynnöistä. Joissakin tapauksissa ajoittaja voisi päättää siirtää suoritukseen tällaisen kompromissipisteen ja kuitata sillä isonkin joukon palvelupyyntöjä.

Tällainen toiminta tarvitsee oletuksen, että palvelupyynnöt eivät ole **eksakteja**. Eksaktilla palvelupyynnöllä tarkoitetaan tässä sitä, että se tulee suorittaa juuri sellaisena kuin se on ajoittajalle annettu. Jos palvelupyyntö sen sijaan ei ole eksakti, se voidaan mieltää pyynnöksi suorittaa jotain sen kaltaista. Esimerkki epäeksaktista palvelupyynnöstä on tilanne, jossa ajoittaja on radio-ohjelman juontaja ja sen asiakkaat ovat radio-ohjelman kuuntelijoita, jotka pyytävät juontajaa soittamaan tietynlaista musiikkia. Musiikkitoiveet esitetään (rock, jazz)-koordinaatiston pisteinä, joissa rock- ja jazz-musiikin mieleisyys voidaan määrittää reaalilukuna väliltä nollasta yhteen. Kun juontaja on saanut kylliksi samankaltaisia musiikkitoiveita, hän voi valita niistä hyvän kompromissin ja täyttää siten usean kuuntelijan toiveet samalla kertaa. Juontajan tekemän musiikkivalinnan ei kuitenkaan tarvitse vastata yhdenkään kuuntelijan varsinaista toivetta.

Tertun palvelupyyntöavaruus ja vuorikiipeilyalgoritmi mahdollistavat sen, että jos palvelupyynnöt ovat epäeksakteja, niin Terttu voisi suorittaa kompromisseja varsinaisten palvelupyyntöjen sijaan. Terttu ei kuitenkaan tee niin. Nipun kannalta tämä ei ole haitta, sillä Nipun palvelupyynnöt ovat eksakteja. Nippu-ohjelmistossa tulosteiden on tulostuttava juu-

1 Palvelupyyntöavaruus on Tertun käyttämä termi, joka vastaa Nipun termiä tulostustyöavaruus.

ri sellaisina kuin käyttäjät ne haluavat. Lisäksi tulostimen valinnassa on vaihtoehtoina vain äärellinen määrä erillisiä tulostimia, eikä niiden väleistä voi valita minkäänlaista kompromissitulostinta. Näin ollen palvelupyyntöavaruutta ei voida käyttää kompromissiratkaisujen tekemiseen. Tämän takia palvelupyyntöavaruuden pisteitä ei tarvitse koskaan kuvata takaisin palvelupyynnöiksi, joten avaruuden pisteiden ei tarvitse sisältää kaikkea palvelupyyntöihin liittyvää tietoa.

Nipussa on keskeisintä tulostustöiden mielekäs niputtaminen eli se, että ajoittaja yhdistelee työt halutulla tavalla. Analysoijan antamat kustannukset yksittäisille ja yhdistetyille töille taas ovat ainut asia, jolla Nipun Tertun ulkopuoliset komponentit voivat vaikuttaa siihen, miten yhdistely tapahtuu. Siksi kustannusten laskeminen on nähtävä ensisijaiseksi suhteessa siihen, miten tulostustöiden sijainti tulostustyöavaruudessa vaikuttaa ajoittajan toimintaan. Töiden on kuitenkin sijaittava avaruudessa, ja niiden sijainnit vieläpä vaikuttavat siihen, mitä töitä ajoittaja edes harkitsee yhdistettäväksi, ja siihen, mitkä työt siirtyvät tulostimille minkäkin työn mukana. Tästä syystä on tärkeää pyrkiä saamaan tulostustöiden koordinaatit kuvaamaan samoja asioita kuin mitä kustannusfunktio kuvaa. Samoin koordinaatisto ei saa estää yhdistyskelpoisten töiden yhdistämistä. Toisaalta koordinaatisto ei saa ujuttaa selkeään tulostustyönippuun kylkiäiseksi sellaisia töitä, jotka eivät kyseiseen selkeään nippuun yhdisty, mutta jotka voisivat hyvinkin yhdistyä johonkin toiseen nippuun, joka ei vain kuulu tämän ajoituspäätöksen alaisuuteen.

Osa B

Yksittäisen työn kustannuksen suhde kahden työn yhdisteen kustannukseen

Tässä luvussa pohditaan analysoijan tarjoamia kustannusfunktioita. Pohdintaa tarvitaan, sillä Tertun dokumentaatiossa ei näytetä selitettävän kovin tarkasti kustannusfunktioiden välisen suhteen luonnetta.

Tertun dokumentaation mukaan yksittäisen palvelupyynnön kustannuksen ja kahden palvelupyynnön yhdisteen kustannuksen merkitys ilmenee ainoastaan vertailussa

$$combinedCost(job1,\ job2) < cost(job1) + cost(job2) \quad .$$

Jos vertailun totuusarvo on tosi, palvelupyynnöt yhdistetään. Muutoin niitä ei yhdistetä. Tämän vertailun tausta-ajatuksena tuntuu olevan oletus

(1) $$combinedCost(job1,\ job2) = cost(fusion(job1,\ job2)) \quad .$$

Funktio *combinedCost* on siis kurkistus mahdollisen yhdistetyn palvelupyynnön kustannukseen. Periaatteessa funktio *combinedCost* voitaisiin sivuuttaa kokonaan ja käyttää sen sijasta funktioita *cost* ja *fusion* yhtälön (1) mukaan. Todennäköisimmin Terttu pyrkii tehokkuuteen funktion *combinedCost* käytöllä. Näin vältetään palvelupyyntöjen (mahdollisesti työläs) yhdistäminen tapauksissa, joissa vain todetaan, että yhdisteen kustannus on liian suuri eikä yhdistettä siksi tarvita.

Toinen Tertun kustannusfunktioiden tausta-ajatus vaikuttaa olevan, että palvelupyynnölle on laskettavissa kustannusta kuvaava lukuarvo suoraan palvelupyynnön tiedoista. Näin ollen funktio *cost* on toteutettavissa luontevalla tavalla ja funktio *combinedCost* voidaan toteuttaa käyttämällä yhtälöä (1) kuitenkin niin, että palvelupyyntöjen yhdistämistä ei varsinaisesti suoriteta, vaan *combinedCost* kykenee itse päättelemään yhdisteen kustannuksen laskemiseen tarvittavat tiedot suoraan parametreina annetuista palvelupyynnöistä.

Nippu-järjestelmässä on vaikeaa laskea tulostustyönipulle kustannusta vain nipun tietojen perusteella. Periaatteessa kustannus voitaisiin laskea esimerkiksi niin, että kustannuksen lähtöarvona on 1,0. Kustannus kerrottaisiin alle 1,0:n kertoimella – sitä pienemmällä, mitä enemmän töitä nipussa on. Kustannus kerrottaisiin myös kertoimella, joka riippuu nipun sisältämien tulostustöiden sijaintien keskihajonnasta – mitä suurempi hajonta, sitä suurempi kerroin. Tällaisen lähestymistavan vaikeus johtuu siitä, että kustannuksen laskukaavan ja järjestelmään haluttujen yhdistelemisominaisuuksien suhde on ihmiselle vaikea hahmottaa. Mielekkäiden laskukaavojen tuottaminen vaatisi todennäköisesti useita käytännön kokeiluja.

Tämän takia Nippu-järjestelmään ei olla kehitetty yleistä tulostustyönipun kustannuksen laskentafunktiota. Sen sijaan on kehitetty yleinen funktio *costMultiplier*, joka ilmaisee suhdetta

$$costMultiplier(job1,\ job2) = \frac{combinedCost(job1,\ job2)}{cost(job1) + cost(job2)} \quad .$$

Tämän funktion avulla voidaan toteuttaa Tertun vaatimat funktiot *cost* ja *combinedCost*. Nipussa onkin asetettu

$$\begin{cases} cost(job) = 1{,}0 \\ combinedCost(job1, job2) = costMultiplier(job1, job2) \cdot (cost(job1) + cost(job2)) \end{cases} \cdot$$

Liite 4. Asetustiedostojen mallit

Alla on malli jokaisesta Nippuun kuuluvasta asetustiedostosta. Tässä esimerkkitapauksessa on määritelty kaksi käyttäjää ja kaksi tulostinta. Töiden enimmäismäärä on sata, tosin tätä parametria ei tässä ohjelmistoversiossa käytetä. Analysoijan vakioista on asetettu oletusarvot. Kaikki komponentit sijaitsevat nyt samassa verkkoasemassa, jolloin IP-verkko-osoite on aina 127.0.0.1. Tiedostoissa on myös tyhjiä rivejä ja #-merkillä alkavia kommenttirivejä, jotka sivuutetaan tiedostoa luettaessa. Asetustiedostot on määritelty liittymäkuvauksessa luvussa 3.3.2.

Yhteinen asetustiedosto – nippu.conf

```
# tulostustöiden maksimimäärä
MAX_JOBS=100

# käyttäjien asetukset
USER_oopisk=2%3%5
USER_hhakkeri=1%2%3

# tulostinten asetukset
PRINTER_ps2=3%4%5%20
PRINTER_ps10=2%4%8%80

# analysoijan asetukset
SAMEUSERMULTIPLIER=0.3
MAXNUMBEROFPAGES=100
BIGBUNDLEMULTIPLIER=1.1
DISTANCELIMIT=20.0
MINDISTANCEMULTIPLIER=0.7
SAMEFORCEDPRINTERMULTIPLIER=1.0
COORDINATESTYLE=submitter
FASTPRINTERDISTANCE=50.0
```

Käyttöliittymän asetustiedosto – nippu-userinterface.conf

```
# vastaanoton verkkoparametrit
RECEPTION_LOCATION=127.0.0.1:7702

# tulostushallinnan verkkoparametrit
PRINTINGMANAGEMENT_LOCATION=127.0.0.1:7704
```

Vastaanoton asetustiedosto – nippu-reception.conf

```
# analysoijan verkkoparametrit
ANALYZER_LOCATION=127.0.0.1:7703

# ajoittajan verkkoparametrit
SCHEDULER_LOCATION=127.0.0.1:7700

# nippujen lähettämistapa:
PRINTJOBSENDING=bundle
```

Analysoijan asetustiedosto – nippu-analyzer.conf

```
# tulostushallinnan verkkoparametrit
PRINTINGMANAGEMENT_LOCATION=127.0.0.1:7704
```

Tulostushallinnan asetustiedosto – nippu-printingmanagement.conf

```
# ajoittajan verkkoparametrit
SCHEDULER_LOCATION=127.0.0.1:7701
```

Liite 5. Sanasto

Aidosti positiivinen tarkoittaa samaa kuin aidosti nollaa suurempi, eli positiivinen mutta ei nolla.

Akselit ovat tulostustyöavaruuden ulottuvuuksien nimiä.

Alkeistulostustyö tarkoittaa sellaista tulostustyötä, joka ei ole yhdiste muista töistä tai yhdisteistä.

Bundle on Nippu-ohjelmiston toteutuksessa keskeinen tietorakenne, joka sisältää yhtä tulostustyönippua koskevat tiedot (tietosuunnitelman luku 3.1.2). Bundle on englantia ja suomeksi nippu.

DNS (domain name service) on Internetissä käytettävä nimipalvelujärjestelmä, jolla verkkoasemille määritellyt DNS-nimet voidaan kuvata IP-verkko-osoitteiksi ja päinvastoin. DNS:ää käytetään verkkoaseman käyttöjärjestelmän tarjoaman rajapinnan kautta.

DNS-nimi on sellainen verkkoaseman nimi, joka voidaan kuvata kyseisen verkkoaseman IP-verkko-osoitteeksi DNS-nimipalvelun avulla. Asiaan liittyvä Internet-standardi (RFC) määrää, että sallittuja merkkejä DNS-nimissä ovat kirjaimet (a–z), numerot (0–9), tavuviiva ja piste.

FQDN-muoto (fully qualified domain name) tarkoittaa DNS-nimen kirjoitusmuotoa, jossa ovat mukana kaikki korkeamman tason verkkotunnukset pisteillä erotettuna, esimerkiksi melkki.cs.helsinki.fi. Paikallisen verkon DNS-nimi voidaan kirjoittaa myös lyhyessä muodossa, esimerkiksi melkki.

IP-verkko tarkoittaa sellaista tietoliikenneverkkoa, jossa tietoliikenteen verkkotason protokollana käytetään IP-protokollaa (Internet protocol). IP-protokolla on Internetissä ja useimmissa lähiverkoissa käytettävä verkkoprotokolla.

Java on yleiskäyttöinen, olioperustainen ohjelmointikieli.

Javadoc on Java-kehitystyökalujen mukana tuleva ohjelma, jolla voidaan automaattisesti luoda koodidokumentti Javadoc-syntaksin mukaan kommentoidusta Java-lähdekoodista.

Koordinaatti on tiettyyn tulostustyöavaruuden akseliin liittyvä lukuarvo. Analysoija määrittelee jokaiselle tulostustyölle N kappaletta koordinaatteja, jossa N on tulostustyöavaruuden ulottuvuuksien lukumäärä.

Linux on vapaasti saatava, unix-tyyppinen käyttöjärjestelmä.

Nippu on sekä ohjelmistotuotantoprojektin että projektin tuotteena syntyvän tulostuspalvelinohjelmiston nimi. Nippu-nimi valittiin, koska simuloitava käyttäjä saa tuotteen avulla useat tulosteensa yhdessä nipussa. Nippu-projekti toteutetaan Helsingin yliopiston tietojenkäsittelytieteen laitoksella kesällä 2003.

PrintJob on Nippu-ohjelmiston toteutuksessa keskeinen tietorakenne, joka sisältää yhteen alkeistulostustyöhön liittyvät tiedot (tietosuunnitelman luku 3.1.1).

Swing on graafisten käyttöliittymäkomponenttien kirjasto, jota käytetään Java-ohjelmoinnissa.

TCP-yhteys tarkoittaa IP-verkossa TCP-yhteysprotokollalla (transmission control protocol) muodostettua tietoliikenneyhteyttä. TCP on UDP:n ohella tavallisin yhteysprotokolla IP-verkoissa. TCP-protokolla takaa sovelluksen lähettämien pakettien pääsemisen perille ja pakettien keskinäisen järjestyksen säilymisen.

Terttu on ajoittajaydin, joka sisältyy keskeisenä komponenttina Nippuun. Terttuun kuuluvat myös visualisoija ja tietoliikennekomponentti. Terttu on toteutettu Helsingin yliopiston Terttu-ohjelmistotuotantoprojektissa kesällä 2002.

Tietoliikenne on tiedon välitystä eri ohjelmistokomponenttien välillä käyttäen erityisesti TCP/IP-verkkoyhteyksiä

Tietoliikennekomponentti käsittää kaiken Nipun tietoliikenteeseen liittyvän ohjelmarakenteen.

Tietoliikenneprotokolla on säännöstö, jota verkkoasemien käyttöjärjestelmien ja sovellusten on noudatettava, jotta niiden välinen yhteys olisi mahdollinen. Esimerkkejä tietoliikenneprotokollista ovat Nipun sovellustason protokolla (liittymäkuvauksen luku 3.3.3), yhteystason TCP-protokolla ja verkkotason IP-protokolla.

Tietoliikenneyhteys tarkoittaa kahden Nipun komponentin välistä tietoliikennettä erityisesti niin, että jos esimerkiksi vastaanotto on yhteydessä kahteen eri käyttöliittymään, on tällöin kyseessä kaksi eri tietoliikenneyhteyttä. Nipussa käytetään TCP-protokollan mukaisia tietoliikenneyhteyksiä.

Tili on tässä dokumentissa yleisesti käytetty lyhenne sanasta tietoliikenne.

Toimintaympäristö tarkoittaa sitä fyysistä tai simuloitua verkkoasemien muodostamaa verkkoa, jossa Nippu hallinnoi tulostusta.

Tulostaja tarkoittaa sellaista toimintaympäristössä toimivaa agenttia, joka synnyttää tulostuspyyntöjä.

Tuloste on tulostuskomennon suorittamisesta syntynyt lopputuote.

Tulostustyö on Nipun tai jonkin toimintaympäristön laitteen käsittelyssä oleva, tulostajan antama tulostuskomento. Tulostustyö voi olla yhdiste (eli nippu) kahdesta tai useammasta tulostustyöstä tai niiden yhdisteestä. Katso myös **alkeistulostustyö**.

Tulostustyöavaruus on N-ulotteinen koordinaatisto, johon analysoija sijoittaa tulostustyöt ajoittajan pyynnöstä. Ajoittaja tarkastelee töiden samankaltaisuutta sen perusteella, miten kaukana toisistaan ne ovat tulostustyöavaruudessa. Tulostustyöavaruus vastaa Tertun käsitettä palvelupyyntöavaruus.

Verkkoasema on yksittäinen tietoliikenneverkkoon liitetty laite, esimerkiksi tietokone tai tulostin.

Ympäristömuuttuja on käyttöjärjestelmän asettama käyttöympäristön parametrin sisältävä muuttuja, jonka arvon ajettava ohjelma voi lukea. Esimerkiksi unix-tyyppisissä järjestelmissä *USER*-ympäristömuuttuja sisältää käyttäjän käyttäjätunnuksen.

Liite 6. Javadoc-dokumentti lähdekoodista

Seuraavilla sivuilla on automaattisesti tuotettu Javadoc-dokumentti Nipun version 1.0 lähdekoodista. Dokumentin on tuottamisessa on käytetty Javadoc-ohjelmaan liitettävää PDFDoclet-lisäosaa. Javadoc-dokumentissa on oma sisäinen sivunumerointi ja lopussa sisällysluettelo.