

## **Käyttö- ja ylläpito-ohje**

PUSU-ryhmä

Helsinki 13.12.2007

Ohjelmistotuotantoprojekti

HELSINGIN YLIOPISTO  
Tietojenkäsittelytieteen laitos

**Kurssi**

581260 Ohjelmistotuotantoprojekti (9 op)

**Projektiryhmä**

Jussi Hynninen  
Jaakko Juvonen  
Paavo Koskinen  
Mikko Leino  
Janne Salo  
Vesa Tuomiaro

**Asiakas**

Johannes Korpela

**Johtoryhmä**

Kimmo Simola  
Juhani Haavisto (ohjaaja)

**Kotisivu**

<http://www.cs.helsinki.fi/group/pusu/>

**Versiohistoria**

Versio	Päiväys	Tehdyt muutokset
1.0	9.12.2007	Ensimmäinen versio

# Sisältö

<b>1</b>	<b>Johdanto</b>	<b>1</b>
<b>2</b>	<b>Sanasto</b>	<b>1</b>
<b>3</b>	<b>Käyttöohje</b>	<b>2</b>
3.1	Asennusohje . . . . .	2
3.1.1	Ohjelmiston hakemistorakenne . . . . .	3
3.1.2	Esimerkki konfiguraatioskriptin ajamisesta . . . . .	3
3.1.3	Käyttöliittymän salasanasuojaus (ei pakollinen) . . . . .	3
3.2	Konfiguraatiotiedostot . . . . .	4
3.2.1	Palvelimen konfiguraatiotiedostot . . . . .	4
3.2.2	pusu.properties . . . . .	5
3.2.3	log4j.properties . . . . .	5
3.2.4	Ylläpitokäyttöliittymä . . . . .	5
3.3	Käynnistysohje . . . . .	6
3.3.1	Esimerkkejä . . . . .	6
3.4	Toiminnot . . . . .	7
3.4.1	Palvelimen toiminnot . . . . .	7
3.4.2	Ylläpitokäyttöliittymä . . . . .	8
3.4.3	Palvelimen syötteen tarkastelu . . . . .	9
<b>4</b>	<b>Ylläpito</b>	<b>9</b>
4.1	Järjestelmävaatimukset . . . . .	9
4.2	Tarkennuksia suunnitelmiin . . . . .	10
4.3	Toteutumattomat vaatimukset ja suunnitelman osat . . . . .	10
4.4	Tunnetut ongelmat . . . . .	10
4.5	Koodin ylläpito . . . . .	10

4.5.1	SIP-toteutuksen vaihtaminen . . . . .	11
4.5.2	Informa-kirjaston vaihtaminen . . . . .	11
4.5.3	PostgreSQL-tietokannan vaihtaminen . . . . .	11
4.6	Jatkokehitysideoita . . . . .	11
4.6.1	Atom-syötteiden hakeminen . . . . .	11
4.6.2	RSS-syötteiden tuen laajentaminen . . . . .	12
4.6.3	Tietoturva . . . . .	12
4.6.4	Tietokantarajapinnan luominen . . . . .	12
4.6.5	SIP:n laajempi hyödyntäminen ja noudattaminen . . . . .	12
4.6.6	SIP:n vaihtaminen toiseen protokollaan . . . . .	12

# 1 Johdanto

RSS-syötteet toimivat nykypäivänä asiakaslähtöisesti siten, että asiakasohjelmat hakevat tietyin väliajoin palvelimelta uuden RSS-dokumentin, esimerkiksi uutisia. Tämä toiminta on erittäin tehokasta. Ensinnäkin asiakasohjelmien pitää osata veikata, koska tietoa kannattaa hakea ja hakiessaan RSS-dokumentin kaikki tarjolla olevat uutiset lähetetään kerralla, myös vanhat jo haetut. Tästä seuraa tyypillisesti se, että suurin osa palvelimelta haetuista artikkeleista on duplikaatteja, mikä aiheuttaa turhaa dataliikennettä. Erityisesti tämä ongelma korostuu silloin, kun RSS-syötteitä haetaan mobiililaitteilla hitaan ja kalliin datayhteyden ylitse.

PUSU-projektiryhmä on luonut RSS-syötteille uudenlaisen Push-palvelinohjelmiston, joka poistaa edellä mainitun duplikaattiongelman. Järjestelmään lisätään internetissä saatavilla olevia RSS-syötteitä ylläpitokäyttöliittymän kautta, minkä jälkeen niiden sisältämiä artikkeleita lähetetään automaattisesti eteenpäin asiakasohjelmille. Asiakasohjelma ilmoittaa järjestelmälle, mitä syötteitä hän haluaa seurata ja kuinka usein hänelle saa lähettää artikkeleita. Järjestelmä pitää kirjaa asiakasohjelmista siten, että se osaa lähettää ainoastaan uudet artikkelit. Täten asiakasohjelmalle ei lähetetä ollenkaan artikkelien duplikaatteja ja edellä mainittu turha tiedonsiirto järjestelmän ja asiakasohjelman välillä poistuu.

Järjestelmällä on ylläpitokäyttöliittymä, jonka kautta voidaan hallita muun muassa järjestelmän tukemia RSS-syötteitä. Järjestelmä on toteutettu Java-kielellä. Kontrolli- ja tiedonsiirtoprotokollana asiakasohjelman ja järjestelmän välillä käytetään SIP-protokollaa. Protokollaa ei toteutettu erikseen projektia varten, vaan käytettiin valmista avoimen lähdekoodin NIST-SIP -toteutusta (versio 1.2). RSS-syötteiden hakemiseen ja parsimiseen järjestelmä käyttää Informa-kirjastoa (versio 0.7.0).

Tämä dokumentti on tarkoitettu auttamaan ohjelmiston käyttäjiä ja jatkokehittäjiä. Dokumentissa kuvataan miten ohjelmisto asennetaan, miten sitä käytetään ja autetaan alkuun parannusten toteuttamisessa. Dokumentti kirjoitetaan suunnitteludokumentin pohjalta.

## 2 Sanasto

**Järjestelmä** PUSU-projektin tuottama palvelinohjelmisto ja ylläpitokäyttöliittymä.

**Artikkeli** Synonyymi RSS-artikkelille.

**Asiakas** Järjestelmän tilaaja. Tässä projektissa Johannes Korpela.

**Asiakasohjelma** Ohjelma, joka tilaa järjestelmältä RSS-syötteen/syötteitä.

**Asiakkuus** Asiakkuuteen kuuluu kaikki asiakasohjelman tilaukset.

**Client** Synonyymi asiakasohjelmalle.

**Tilaus** Tilaus on asiakkaan ilmaisema tahto vastaanottaa jonkin syötteen artikkeleita. Asiakas muodostaa tilauksen lähettämällä SUBSCRIBE-pyyntöä.

**RSS** XML-pohjainen standardi usein uutisten, blogien yms. julkaisemiseen. Termi viittaa aina RSS:n versioon 2.0, ellei toisin mainita.

**Atom** RSS:n kaltainen julkaisuformaatti, joka tarjoaa RSS:ää laajemmat ominaisuudet.

**RSS-dokumentti** RSS-muotoinen dokumentti.

**RSS-artikkeli** RSS-dokumentin sisältämä yksittäinen artikkeli. Koostuu item-elementistä ja sen sisällöstä.

**RSS-syöte** Palvelimen tarjoama RSS-dokumenttien virta.

**SIP** Protokolla loogisen yhteyden muodostamiseen tietoverkossa. Järjestelmä käyttää tätä asiakasohjelmien kanssa kommunikointiin.

**SUBSCRIBE-pyyntö** SIP:n laajennos, jolla asiakasohjelma voi pyytää tietoa vastaanottajan tilamuutoksista. Järjestelmässä asiakasohjelmat käyttävät SUBSCRIBE-pyyntöä tilatessaan RSS-syötteitä.

**NOTIFY-pyyntö** SIP:n laajennos, jolla SUBSCRIBE-pyyntöä vastaanottaja voi ilmoittaa tilamuutoksista pyytäjälle. Järjestelmä käyttää tätä mm. RSS-dokumenttien lähettämiseen asiakasohjelmalle.

**Informa** LPGL-lisenssin alainen Java-kirjasto, joka toteuttaa mm. RSS- ja Atom -syötteiden noutamiseen liittyvän toiminnallisuuden<sup>1</sup>.

**JAIN SIP** Yhteinen nimitys eräälle Javan SIP-rajapinnalle<sup>2</sup> ja sen toteutukselle.

**Log4j** Apache-projektin tekemä kirjasto, joka toteuttaa lokitiedostojen kirjoittamiseen liittyvän toiminnallisuuden<sup>3</sup>.

**PostgreSQL** BSD-lisenssin alainen tietokantajärjestelmä.

## 3 Käyttöohje

### 3.1 Asennusohje

Ohjelmisto asennetaan purkamalla paketti *pusu-dist.tar.gz* haluttuun polkuun, ajamalla ohjelmiston konfiguraatioskripti ja kopiaamalla *www-käyttöliittymä* haluttuun polkuun.

<sup>1</sup><http://informa.sourceforge.net/index.html>

<sup>2</sup><http://www.jcp.org/en/jsr/detail?id=32>

<sup>3</sup><http://logging.apache.org/log4j/1.2/index.html>

### 3.1.1 Ohjelmiston hakemistorakenne

```
pusu/
|--- bin/          - Käynnistystiedostot
|--- conf/         - Ohjelmiston konfiguraatiotiedostot
|--- lib/          - Ohjelmiston vaatimat kirjastot
|--- log/          - Ohjelmiston tuottamat lokitiedostot
|--- pusuUI/       - Järjestelmän www-pohjainen ylläpitokäyttöliittymä.
|--- setup.sh      - Konfiguraatioskripti
```

### 3.1.2 Esimerkki konfiguraatioskriptin ajamisesta

```
[~]$ tar xvfz pusu-dist.tar.gz
[~]$ cd pusu
[pusu]$ ./setup.sh
    Starting configuration of the PUSU server...
    Server specifics:
Enter IP for PUSU server (default '128.214.9.149'): 128.214.9.149
Enter port for PUSU server (default '4321'): 4321
    Database parameters:
Enter database address (IP or DNS, default 'localhost'):
Enter database port (default 12792):
Enter database name (default 'pusu'): pusudb
Database username and password are used to connect to the database.
A user with the information given will be created.
Enter database username (default 'pusuadmin'):
Enter database password:
Enter log4j initial loglevel for console (default FATAL):
Enter log4j initial loglevel for logfiles (default WARN):
    Creating and initializing the database...
Enter the PostgreSQL administrator password:
    Database created.
Creating /home/jmvhynni/pusu/conf/pusu.properties...
Setting file permissions of /home/jmvhynni/pusu/conf/pusu.properties (0600)...
Creating /home/jmvhynni/pusu/conf/log4j.properties...
Setting file permissions of /home/jmvhynni/pusu/conf/log4j.properties (0600)...
Enter the DNS name of the server that will be hosting the admin
interface (default alkokrunni.cs.helsinki.fi): db.cs.helsinki.fi
Enter the port your web server is using (default 8080):
creating pusuUI/config/config.php...

    Configuration complete!

Copy the directory pusuUI to the root of your web server.
The address for the server feed is http://db.cs.helsinki.fi:8080/pusuUI/
If you want to place the admin UI elsewhere, please edit
pusuUI/config/config.php accordingly
[pusu]$ mv pusuUI ~/public_html/cgi-bin/
[pusu]$ chmod -R a+rX ~/public_html/cgi-bin/pusuUI
```

### 3.1.3 Käyttöliittymän salasanasuojaus (ei pakollinen)

Jos ylläpitokäyttöliittymä näkyy julkisessa verkossa, on järkevää suojata se salasanal-  
la. *Seuraava ohje koskee Apache HTTP-palvelinta Linux-ympäristössä, salasanasuojaus  
muilla palvelinohjelmistoilla saattaa erota Apachesta.*

Käytä *htpasswd*-komentoa luodaksesi salasanatiedoston (esimerkissä nimeltään *.htpasswd* käyttäjätunnukseksi username):

```
htpasswd -c /full/path/to/somewhere/.htpasswd username
```

Mikäli mahdollista, sijoita salasanatiedosto hakemistoon, joka ei näy julkisessa verkossa (ts. jota HTTP-selain ei palvele).

Luo tämän jälkeen käyttöliittymän admin-hakemistoon tiedosto nimeltä *.htaccess* ja lisää sinne seuraavat rivit:

```
AuthType Basic
AuthName "PUSU Admin"
AuthUserFile /full/path/to/somewhere/.htpasswd
Require user username
```

Asetus *AuthName* määrittää tekstin, joka näkyy käyttäjän selaimessa, kun salasanaa suojatulle alueelle kysytään. *AuthUserFile*-asetuksen tulisi osoittaa äsken luotuun salasana-tiedostoon. *Require user* -asetuksella luetellaan sallitut käyttäjät välilyönnein eroteltuna. Nyt *.htaccess*-tiedoston sisältävä hakemisto ja kaikki sen alihakemistot ovat salasanasuojattuja. Varmista vielä, että Apachella on luku- ja suoritusoikeudet salasana- ja *.htaccess*-tiedostoihin.

## 3.2 Konfiguraatitiedostot

### 3.2.1 Palvelimen konfiguraatitiedostot

Palvelimen konfiguraatitiedostot sijaitsevat alihakemistossa *conf/*. Konfiguraatitiedostoja on kaksi, joista *pusu.properties* sisältää yleiset palvelinasetukset ja *log4j.properties* lokitukseen liittyvät asetukset.



### 3.2.2 pusu.properties

Asetus	Selitys
hostIP	IP-osoite tai DNS-nimi, jolla palvelin, jolla PUSU-ohjelmisto sijaitsee, näkyy ulkomaailmaan. <i>Pakollinen asetus.</i>
port	Portti, jossa PUSU-ohjelmisto kuuntelee sisääntulevia yhteyksiä. <i>Pakollinen asetus.</i>
dbAddress	Tietokantapalvelimen osoite (oletuksena 'localhost')
dbPort	Portti, jossa tietokantaohjelmisto kuuntelee yhteyksiä
dbName	Tietokannan nimi. <i>Pakollinen asetus.</i>
dbUserName	Tietokannan käyttäjätunnus. <i>Pakollinen asetus.</i>
dbPassword	Tietokannan salasana. <i>Pakollinen asetus.</i>
log4j.properties	Täysi polku tiedostoon log4j.properties.

### 3.2.3 log4j.properties

Ohessa on lueteltu vain olennaisimmat asetukset. Log4j:n toiminnasta voi lukea tarkemmin sen manuaalista<sup>4</sup>. Huomioi, että konfiguraatiotiedostossa määritellyjä lokitustasoja noudatetaan vain siihen asti, kunnes järjestelmä käynnistyessään lukee lokitustasonsa tietokannasta.

Asetus	Selitys
log4j.rootCategory	Mihin kaikkialle lokitetaan
log4j.appender.CONSOLE.Threshold	Konsoliin tulostettavien lokitietojen oletustaso
log4j.appender.LOGFILE.File	Täysi polku lokitiedostoon
log4j.appender.LOGFILE.Append	Käytetäänkö samaa lokitiedostoa uudelleen
log4j.appender.LOGFILE.Threshold	Lokitiedostoon tulostettavien lokitietojen oletustaso
log4j.appender.LOGFILE.MaxFileSize	Lokitiedoston maksimikoko
log4j.appender.LOGFILE.MaxBackupIndex	Säilytettävien lokitiedostojen määrä (kun MaxFileSize ylittyy)

### 3.2.4 Ylläpitokäyttöliittymä

Käyttöliittymän asetustiedot sijaitsevat käyttöliittymähakemiston alihakemistossa *conf* tiedostossa *config.php*. Asetukset on määritelty assiosiatiivisessa taulukossa *\$config*, jossa indeksinä toimii asetuksen nimi ja alkiona asetuksen arvo, siis esim. *\$config['setting'] = 'value'*;. Tämä taulukko on muiden käyttöliittymäskriptien (erityisesti *index.php* ja

<sup>4</sup><http://logging.apache.org/log4j/1.2/manual.html>

*admin/index.php*) käytössä sellaisenaan. Mahdolliset asetukset on lueteltu taulukossa alla.

Asetus	Selitys
db	Käytettävä tietokantajärjestelmä. Tällä hetkellä ainoa tuettu arvo on 'postgres'. Asetus on olemassa laajennettavuuden helpottamiseksi.
db_user	Tietokannan käyttäjänimi.
db_pw	Tietokannan salasana.
db_server	Tietokantapalvelimen osoite.
db_port	Tietokantapalvelimen käyttämä portti.
db_name	Käytettävän tietokannan nimi.
server_feed	Palvelimen syötteen nimi, ts. tietokannan subscriptions- taulun palvelimen syötettä kuvaavan rivin avain. Subscriptions-taulussa on oltava rivi, jolla on tämä avain.
server_article_url	Palvelimen syötteeseen lisättävien artikkeleiden linkin alkuosa, johon lisätään kunkin artikke- lin tunnus (tyypillisesti tunnusta käytetään GET- parametrina, esim. <a href="http://example.com/?guid=2">http://example.com/?guid=2</a> , missä <a href="http://example.com/?guid=">http://example.com/?guid=</a> on asetustiedostossa esiintyvä linkin alkuosa). Alkuosa+artikkelin tunniste on myös link- ki, joka palvelimen syötteen tilanneille asiakasohjelmille lähetetään kyseisen artikkelin linkkinä.

### 3.3 Käynnistysohje

Järjestelmä käynnistetään komennolla *pusu start*. Komento sijaitsee pusu-hakemiston bin-alihakemistossa. Vastaavasti *pusu stop* pysäyttää järjestelmän ja *pusu restart* käynnistää sen uudestaan. *pusu status* kertoo järjestelmän tilan. Käynnistämiseen käytetään Wrapper-ohjelmistoa, jonka dokumentaatiota<sup>5</sup> voi olla hyödyllistä tutkia. Käynnistyskomennon lo-ki löytyy tiedostosta *pusu.startup.log*, joka sijaitsee alihakemistossa log.

#### 3.3.1 Esimerkkejä

```
[pusu]$ bin/pusu start
Starting Pusu server...
[pusu]$ bin/pusu status
Pusu server is running (PID:18842).
[pusu]$ bin/pusu stop
Stopping Pusu server...
Stopped Pusu server.
[pusu]$ bin/pusu status
Pusu server is not running.
```

<sup>5</sup><http://wrapper.tanukisoftware.org/>

## 3.4 Toiminnot

Tässä kappaleessa kuvataan järjestelmän (palvelinohjelmisto, ylläpitokäyttöliittymä) toiminnot. Vaatimusmäärittelydokumentissa on useista toiminnoista tarkka yleiskuvaus sekä ulkoisten rajapintojen kuvaus. Suunnitteludokumentti ja ohjelmakoodi puolestaan sisältävät dokumentaatiota näiden toimintojen yksityiskohdista ja toteutuksesta.

### 3.4.1 Palvelimen toiminnot

Palvelimen toiminnot voidaan jakaa kahteen kategoriaan: asiakasohjelmien käyttämät toiminnot sekä ulkopuolisten syötteiden noutaminen. Näitä toimintoja käsitellään tarkemmin seuraavassa.

#### Toiminnot asiakasohjelman kannalta

Palvelin kommunikoi asiakasohjelmien kanssa ennalta määriteltujen rajapintojen avulla. Näitä rajapintoja ovat SIP-protokolla, Atom- ja RSS-standardit sekä vaatimusmäärittelydokumentissa kuvattu XML-pohjainen tapa välittää tilaustietoja ja syötelistoja. Palvelin ei ota kantaa asiakasohjelmien toteutukseen, ainoa vaatimus on, että asiakasohjelma käyttää SIP-protokollaa, erityisesti sen SUBSCRIBE/NOTIFY-laajennosta, ja lähettää palvelimelle SUBSCRIBE-pyyntöissään vaatimusmäärittelydokumentin mukaisia XML-dokumentteja.

Palvelin tarjoaa asiakasohjelmille seuraavat toiminnot: tilausten tekeminen, muokkaaminen ja lopettaminen, saatavilla olevien syötteiden kysely sekä syötteiden toimitus asiakasohjelmien ilmoittamien tilausasetusten mukaan. Näiden toimintojen käyttö asiakasohjelman näkökulmasta on kuvattu tarkemmin vaatimusmäärittelydokumentin luvuissa 4.1.2 ja 5 sekä palvelimen näkökulmasta suunnitteludokumentissa. Seuraavassa on tiivistelmä näistä toiminnoista.

Tilausten lopetus: asiakasohjelman tunnistetiedoilla (from-tag, to-tag, call-id) varustettu SUBSCRIBE-pyyntö, jossa Expires-otsakkeen arvoksi on asetettu 0. Sivuvaikutuksena asiakasohjelmalle lähetetään syötelistaus.

Tilausten tekeminen: asiakasohjelman tunnistetiedoilla varustettu SUBSCRIBE-pyyntö, jonka rungossa on mukana XML-muodossa tilausasetukset.

Tilausten muokkaaminen: asiakasohjelman tunnistetiedoilla varustettu SUBSCRIBE-pyyntö, jonka rungossa on mukana XML-muodossa tilausasetukset ja jonka tunnistetiedot liittyvät johonkin voimassaolevaan asiakkuuteen.

Saatavilla olevien syötteiden kysely: joko samalla tavalla kuin tilauksen lopetus tai, jos asiakasohjelma ei halua lopettaa voimassaolevia tilauksia, lähettämällä tilaukset päivittävä SUBSCRIBE-pyyntö, jonka runko-osassa ilmaistaan list-elementillä, että ensimmäisen NOTIFYn tulisi olla syötelistaus.

Syötteiden toimitus: RSS- tai Atom-dokumentteja sisältävät NOTIFY-pyyntöt asiakasohjelmille.

### **Syötteiden nouto**

Palvelin noutaa RSS-standardin mukaisia syötteitä ulkopuolisista palveluista HTTP-protokollaa käyttäen. Näistä syötteistä noudetut artikkelit tallennetaan palvelimen omaan tietokantaan, josta niitä välitetään asiakasohjelmille. Noudettavia syötteitä voi hallita ylläpitokäyttöliittymästä (ks. seuraava kappale).

### **3.4.2 Ylläpitokäyttöliittymä**

Järjestelmän ylläpitokäyttöliittymällä voi seurata ja hallita palvelimen toimintaa. Käyttöliittymässä on toiminnot palvelimen suoritusajakaisten asetusten muuttamiseen, vanhojen artikkelien poistamiseen tietokannasta, noudettavien syötteiden hallintaan, artikkelien kirjoittamiseen palvelimen syötteeseen sekä asiakkuuksien hallintaan. Kannattaa huomata, että ylläpitokäyttöliittymä ei ole suorassa yhteydessä itse palvelinohjelmistoon, vaan vaikuttaa sen toimintoon ainoastaan tietokannan välityksellä. Tästä syystä useiden asetusten, kuten lokitustason tai minimilähetysvälin, muuttaminen ei vaikuta järjestelmän toimintaan välittömästi.

Muokattavissa olevia järjestelmän suoritusajakaisten asetuksia ovat lokitustaso, NOTIFY-pyyntöjen minimilähetys ja yhdessä NOTIFY-pyyntöissä lähetettävien artikkelien maksimimäärä. Lokitustasoja on viisi, joista ääripää "debug" tuottaa eniten tietoa ja toinen ääripää "fatal" lokittaa vain järjestelmän toiminnan estävät virheet. Alempi (enemmän tietoa tuottava) lokitustaso sisältää kaikki ylemmät, esim. debug lokittaa debug-viestien lisäksi kaikki info-, warn-, error- ja fatal-tasoiset viestit.

NOTIFY-pyyntöjen minimilähetysväli on aikaväli, jota useammin palvelimelta ei lähde NOTIFY-pyyntöjä asiakasohjelmille (poislukien tilauksen tekemiseen vastauksena tulevat NOTIFY-pyyntöt, jotka lähetetään aina välittömästi SIP-protokollan mukaisesti). Käytännössä tämä tarkoittaa sitä, että järjestelmä tarkistaa asetuksen määrittämän ajan välein, voiko järjestelmässä oleviin tilauksiin lähettää uusia artikkeleita (ts. sallivatko tilausten asetukset sen kyseisellä hetkellä ja onko uusia artikkeleita saatavilla).

Yhdessä NOTIFY-pyyntöissä lähetettävien artikkelien maksimimäärä on globaali asetus, joka tarvittaessa rajoittaa yhdelle tilaukselle kerralla lähtevien artikkelien määrää. Jos jonkin tilauksen asetuksissa määritelty kerralla lähetettävien artikkelien maksimimäärä on suurempi kuin asetettu globaali maksimi, käytetään globaalia maksimia tilauksen oman asetuksen sijaan rajoittamaan kerralla lähtevien artikkelien määrää.

Ylläpitokäyttöliittymä mahdollistaa myös määriteltä rajaa vanhempien artikkelien poistamisen tietokannasta. Poistettavien artikkelien tulee olla vähintään vuorokauden vanhoja.

Syötteiden hallintaan tarjottavia välineitä ovat uusien syötteiden lisääminen, syötteiden tietojen (osoite ja päivitysväli) muokkaaminen ja syötteiden poistaminen. Syötteen poistaminen poistaa myös kaikki syötteen artikkelit tilaukset. Jokaisesta syötteestä näytetään myös статистиikkaa: tilaajien ja artikkelien lukumäärä sekä viimeisin päivitysajankohta ja syötteen tila. Huomaa, että päivitysajankohta ei välttämättä tarkoita aikaa, jolloin syöte on haettu, vaan aikaa jolloin syötteen tila on viimeksi tarkastettu. Varsinkin harvoin päivittymään asetettujen syötteiden kohdalla asia voi olla näin. Mahdollisia tiloja ovat "ok", "not yet updated"(syöte on uusi, eikä sitä ole ehditty vielä päivittää kertaakaan), "timeout"(syötteen viimeisin nouto aikakatkaistiin), "not found"(annetusta osoitteesta ei löytynyt mitään) ja "error"(jokin muu virhe syötteen noutamisessa). "Not found- ja "error"-tilassa olevat syötteet kannattaa yleensä poistaa järjestelmästä.

Ylläpitokäyttöliittymästä on mahdollista myös tarkastella järjestelmässä olevia asiakkuuksia. Näistä näytetään id-numero (server tag), eräs yhteyden identifioivista merkkijonoista (call-id), asiakasohjelman contact-tiedot (käytännössä ip-osoite ja portti) sekä asiakkuuden päättymishetki. Lisäksi kunkin asiakkuuden kohdalla listataan kyseisen asiakasohjelman tilaamat syötteet. Asiakkuuksia voi käyttöliittymän kautta myös poistaa, joskaan tämä ei useinkaan liene tarpeellista.

### 3.4.3 Palvelimen syötteen tarkastelu

Palvelimen syötteeseen lisättyjä artikkeleita voi tarkastella omalta sivultaan (käyttöliittymähakemiston etusivu, index.php). Myös ylläpitokäyttöliittymästä on linkki tälle sivulle. Jokaista artikkelia voi tarkastella myös erikseen omalla sivullaan, joka generoidaan dynaamisesti syötesivulle annettavan GET-parametrin avulla (ks. lisätietoja kohdasta Konfigurointitiedot).

## 4 Ylläpito

### 4.1 Järjestelmävaatimukset

Järjestelmä itsessään ei ole kovinkaan vaatelias suoritinajan tai muistin suhteen, vaan resurssien tarve riippuu siitä kuinka monia asiakkaita järjestelmän on palveltava ja kuinka montaa syötettä järjestelmä tarjoaa edelleen asiakkailleen.

Palvelin toimii PostgreSQL-tietokantajärjestelmän versiolla 7.4 tai uudemmalla, ja Java-ajoympäristöstä pitää olla vähintään versio 1.5. Ylläpitokäyttöliittymää käyttääksesi tarvitset HTTP-palvelimen ja PHP 5 -ajoympäristön. Nämä ohjelmistovaatimukset sanelevat myös laitteistovaatimukset.

## 4.2 Tarkennuksia suunnitelmiin

Toteutuksen aikana tulleet muutokset löytyvät päivitetyn suunnitteludokumentin versiosta 1.1. Kovinkaan merkittäviä muutoksia ei suunnitelmiin kuitenkaan jouduttu tekemään.

Atom-syötteiden hakemisesta luovuttiin, koska Informa-kirjasto ei kyennytkään niitä noutamaan kuten oli ymmärretty. Sekvenssikaavioihin tuli myös lukuisia pieniä muutoksia. Etenkin SIP-vastausten lähettäminen muuttui oleellisesti käyttämään omaa rajapintaansa SipResponse.

## 4.3 Toteutumattomat vaatimukset ja suunnitelman osat

Vaatimukset pystyttiin toteuttamaan lähes kokonaisuudessaan. Ainoana toteutumattomana vaatimuksena on Atom-syötteiden nouto, joka on vaatimusmäärittelydokumentissa 4.1.4 pienimmällä mahdollisella prioriteetilla 3.

## 4.4 Tunnetut ongelmat

- Informan toiminnasta johtuen syötteistä noudettavat uudet artikkelit eivät päivitty tietokantaan heti, kun ne on noudettu (ts. InputFeedListener-rajapinnan feedUpdated-metodia ei kutsuta), vaan vasta, kun syöte noudetaan seuraavan kerran. Tämä voi olla pieni haitta, jos kyseessä on harvoin noudettava syöte. Ongelma johtuu siitä, että Informa ilmoittaa ensin syötteen päivittyneen (mikä puolestaan saa aikaan feedUpdated-metodin kutsun, jos uusia artikkeleita on), ja vasta sitten uusien artikkelien löytyneen, jos niitä oli. Uudet artikkelit jäävät siis RssInputFeediin odottamaan seuraavaa päivityskierrosta. Ongelman voinee kiertää muokkaamalla InputFeedListener-rajapintaa siten, että feedUpdated-metodille annetaan vain yksi artikkeli kerrallaan, ja tätä metodia kutsutaan aina Informan ilmoittaman uuden artikkelin lisäystapahtuman yhteydessä.
- Informa ei välttämättä toimi oikein, jos noudettavassa RSS-syötteessä on jonkin link-elementin sisältönä epäkelpo URL. Tämä saattaa jopa kaataa Informan. Ainoat mahdolliset ratkaisut tähän ovat Informan päivittyminen (tai itse muokkaaminen) tai Informasta luopuminen.

## 4.5 Koodin ylläpito

Tässä kappaleessa esitellään muutamia kohtia ohjelmistosta, joita mahdollisesti halutaan muokata.

#### **4.5.1 SIP-toteutuksen vaihtaminen**

Palvelin on suunniteltu siten, että ainoastaan luokka SipHandler on riippuvainen nykyisestä SIP-toteutuksesta. Siis SipHandler on kirjoitettava uudestaan uutta SIP-toteutusta käyttämään. Tämä tarkoittaa myös SipResponse-rajapinnan toteuttamista, joka on tällä hetkellä toteutettu SipHandlerin sisäluokkana. Pienenä yksityiskohtana Starter-luokkaa pitää muistaa muokata SipHandlerin alustamisen osalta.

#### **4.5.2 Informa-kirjaston vaihtaminen**

Ainoa Informa-kirjastosta riippuvainen luokka on RssInputFeed. On siis kirjoitettava uusi InputFeed-rajapinnan toteutus hakemaan RSS-syötteitä. Oleellista on myös muistaa päivittää InputFeedFactory-luokka tuottamaan oikeita InputFeed-olioita.

#### **4.5.3 PostgreSQL-tietokannan vaihtaminen**

PusuDatabase-luokka ja ylläpitokäyttöliittymä ovat ne, joita joudutaan muokkaamaan tietokantajärjestelmän vaihtamisen takia. PusuDatabase-luokan konstruktorista pitää vaihtaa tietokannan osoite, ja SQL-lauseet on syytä tarkistaa. Ne toimivat melko todennäköisesti myös muissa tietokantajärjestelmissä. Tosin on käytetty sellaisia tietotyyppejä, joita ei välttämättä sellaisenaan löydy muista tietokantajärjestelmistä.

Asennuskriptissä ja konfigurointitiedostoissa on myös kohtia, jotka ovat sidottuja PostgreSQL-tietokantaan.

### **4.6 Jatkokehitysideoita**

Tässä kappaleessa esitellään muutamia ideoita, joilla ohjelmiston toiminnallisuutta voitaisiin laajentaa.

#### **4.6.1 Atom-syötteiden hakeminen**

Atom-syötteiden haun toteuttaminen vastaa melkolailla edellisen kappaleen Informa-kirjaston vaihtamista. Nyt on vain kirjoitettava uusi InputFeed-rajapinnan toteutus ja luotava InputFeedFactory-luokkaan toiminnallisuus luoda Atom-syötteen hakevia olioita.

On myös syytä huomata, että Atom-syötteillä ovat mahdollisesti monimutkaisempia kuin RSS-syötteet. Esimerkiksi Atom-syötteen sisältö voi olla binääristä. Tästä johtuen kunnollisen Atom-tuen lisääminen saattaa olla suurempi urakka kuin vain InputFeed-rajapinnan toteutus.

#### **4.6.2 RSS-syötteiden tuen laajentaminen**

Toistaiseksi palvelin ei tue kaikkia RSS-syötteiden ominaisuuksia tai kenttiä. Muutostarvetta ei ole mahdollista paikantaa mihinkään yhteen paikkaan ohjelmistossa.

#### **4.6.3 Tietoturva**

Tietoturvaan ei ole PUSU-projektissa kiinnitetty erityistä huomiota. Toki ollaan pyritty siihen, ettei ohjelmistossa olisi tietoturva-aukkoja, mutta yksi jatkokehitysidea voisi olla tietoturvan tarkistaminen ja vahvistaminen.

#### **4.6.4 Tietokantarajapinnan luominen**

Jatkokehityksen yhteydessä voi olla järkevää luoda kaksi rajapintaa, jotka PusuDatabase-luokka toteuttaa. Nämä rajapinnat vaatisivat fetcher- ja communicator-pakkausten tarvitsemien (osin samojen) metodien toteuttamisen. Tämä voisi olla nykyistä järkevämpi suunnitteluratkaisu ja myös helpottaa testaamista. Myös käytettävän tietokannan vaihtaminen näiden rajapintojen avulla lienee helpompaa.

#### **4.6.5 SIP:n laajempi hyödyntäminen ja noudattaminen**

Järjestelmää on kohtalaisen helppo laajentaa siten, että SIP-protokollan vapaaehtoisia osakkeita ja muita vastauskoodeja hyödynnetään. Esimerkkinä tästä on "423 Subscription Too Brief"-vastaus, joka voitaisiin lähettää liian lyhyen tilauksen (sekunnin luokkaa) tekeväälle asiakasohjelmalle. Myös joitain SIP:n suosituksia voidaan ottaa huomioon, esimerkiksi tilauksen lopettavan NOTIFYn lähettäminen asiakasohjelmille, joiden tilausta ei ole sen loppumisaikaan mennessä uudistettu.

#### **4.6.6 SIP:n vaihtaminen toiseen protokollaan**

Yksi mielenkiintoinen vaihtoehto jatkokehitykselle saattaisi olla SIP:n vaihtaminen johonkin toiseen protokollaan. Tämä tarkoittaisi tosin communicator-pakkauksen ja osan data-pakkausta uudelleen kirjoittamista.