# Käyttöohje

Boa Open Access

Helsinki 5.5.2006

Ohjelmistotuotantoprojekti

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

**Kurssi**

581260 Ohjelmistotuotantoprojekti (6 ov)

**Projektiryhmä**

Ilmari Heikkinen
Timo Hintsa
Erno Härkönen
Arto Vuori
Mikko Kautto

**Asiakas**

Olli Niinivaara

**Johtoryhmä**

Juha Taina
Riikka Kaven

**Kotisivu**

`http://www.cs.helsinki.fi/group/boa`

**Versiohistoria**

| Versio | Päiväys | Tehdyt muutokset |
|---|---|---|
| 1.0 | 5.5.2006 | Ensimmäinen versio |

# Sisältö

# 1 Johdanto

Boa on ohjelma, joka muuntaa CiteSeer-, DBLP-muotoista metadataa QS-muotoisiksi tiedostoiksi. Metadatalähteitä kutsutaan yhteisellä nimellä raakadatalähteet ja näiden sisältämää dataa raakadataksi. Ohjelma toimii Java-ympäristössä ja vaatii toimiakseen Java 2 Runtime Environment 5.0 -ympäristön tai tätä uudemman version.

# 2 Ohjelman toiminta

Boa koostuu muuntimista ja tulostimista. Muunnin käsittelee raakadataa ja antaa sitä tulostimille sopivassa muodossa.

Ohjelma lukee aluksi pääasetustiedoston, josta selviää muiden asetustiedostojen sijainti. Tämän jälkeen ohjelma lukee kaikki raakadatalähteiden asetustiedostot ja käyttäjien asetustiedostot.

Ohjelma siirtyy tämän jälkeen käsittelemään raakadatalähteitä yksitellen. Raakadatalähteen olemassa olevat tilatiedostot luetaan, jos raakadata lähde on käsitelty aikasemmin. Raakadatalähteen tila tallenetaan tilatiedostoihin lähteen käsittelyn jälkeen. Mahdolliset tulostustiedostot tallennetaan asetuksissa määriteltyyn hakemistoon.

Ensimmäisellä ajokerralla käyttäjän ja raakadatalähteen tiedot annetaan tulostimelle. Näitä ei tulosteta seuraavilla raakadatalähteen käsittelykerroilla. Tilatiedostoihin tallennetaan tieto käsitellyistä tiedostoista. Jokainen raakadatalähteen syötehakemistossa ollut tiedosto käsitellään vain kerran. Seuraavilla ohjelman ajokerroilla käsitellään vain hakemiston uudet tiedostot.

# 3 Esimerkkiajo

Esimerkkiajo, jossa luodaan CiteSeer-raakadatalähde, DBLP-raakadatalähde, käyttäjätiedot, ja ajetaan ohjelma.

## 3.1 Ensimmäinen askel: hakemistojen luonti

Luomme hakemistot pääasetustiedostolle, raakadatalähteiden asetustiedostoille, käyttäjätiedoille, tilatiedoille, ja yhden jokaista raakadatalähdettä kohden ulostulohakemistoineen.

```
mkdir settings
mkdir raakadatalähteet
mkdir käyttäjätiedot
mkdir tilatiedot
mkdir citeseer_raakadata
```

```
mkdir dblp_raakadata
mkdir citeseer_ulostulo
mkdir dblp_ulostulo
```

## 3.2 Toinen askel: asetustiedostojen luonti

Ensiksi luomme pääasetustiedoston nimeltä `settings/boa.cfg`.

Editoimme pääasetustiedoston näyttämään seuraavalta:

```
sourcedir = raakadatalähteet
userdir = käyttäjätiedot
datadir = tilatiedot
```

Seuraavaksi luomme käyttäjätietotiedoston itsellemme nimeltä `käyttäjätiedot/erkki.cfg` ja lisäämme siihen oman nimemme.

```
name = Erkki Esimerkki
```

Viimeiseksi luomme raakadatalähteiden asetustiedostot hakemistoon raakadatalähteet. CiteSeer:lle luomme tiedoston `raakadatalähteet/citeseer.cfg` ja muokkaamme sen esimerkiksi seuraavan näköiseksi:

```
name = Citeseer-lähteeni
user = erkki.cfg
transformer = citeseer
inputdir = citeseer_raakadata
outputdir = citeseer_ulostulo
datadir = tilatiedot
printer = qs
```

Vastaavasti tiedosto `raakadatalähteet/dblp.cfg` voisi näyttää seuraavalta:

```
name = DBLP-lähteeni
user = erkki.cfg
transformer = dblp
inputdir = dblp_raakadata
outputdir = dblp_ulostulo
datadir = tilatiedot
printer = qs
```

### 3.3 Kolmas askel: raakadatatiedostojen kopiointi lähdehakemistoihin ja ohjelman ajaminen

Antaaksemme ohjelmalle jotain pureksittavaakin, kopioimme CiteSeer-muotoisia tiedostoja hakemistoon `citeseer_raakadata` ja DBLP-muotoisia tiedostoja hakemistoon `dblp_raakadata`.

Nyt, kun kaikki on valmiina, ajamme ohjelman komennolla `java -jar Boa.jar`

Ohjelma käy läpi raakadatatiedostot ja kirjoittaa QS-muotoisia tiedostoja hakemistoihin `citeseer_ulostulo` ja `dblp_ulostulo`.

## 4 Ohjelman käynnistäminen

Ohjelma käynnistetään komentoriviltä komennolla:

```
java -jar Boa.jar
```

Ajamalla ohjelma ilman mitään parametreja käytetään pääasetustiedostona tiedostoa `settings/boa.cfg`. Ohjelman ajaminen tarvitsee aina pääasetustiedoston. Jos asetustiedostoa ei löydy, ohjelman suoritus keskeytyy.

Käyttäjä voi määritellä käytettävän pääasetustiedoston parametrilla *-m*. Pääasetustiedosto sisältää muiden asetustiedostojen sijainnit.

Esimerkki:

```
java -jar Boa.jar -m example/boa.cfg
```

Ohjelman tulostaa lyhyen käyttöohjeen parametrilla *-h*. Ohjelman suoritus pysähtyy ohjeen tulostuksen jälkeen.

Esimerkki:

```
java -jar Boa.jar -h
```

Parametrilla *-v* ohjelma kertoo suorituksestaan tarkemmin. Tätä parametria kannattaa käyttää kun halutaan tutkia ohjelmaa tarkemmin.

Esimerkki:

```
java -jar Boa.jar -m example/boa.cfg -v
```

## 5 Ohjelman käyttäminen

Kaikki tarvittavat asetukset tulee täyttää asetustiedostoihin ennen ohjelman käynnistämistä. Ohjelma käynnistetään komentoriviltä, jonka jälkeen se käsittelee asetustiedostois-

sa kuvatut raakadatalähteet ja antaa näistä saadut tiedot tulostimille. Ohjelman suoritus pysähtyy, kun kaikki raakadatalähteet saadaan käsiteltyä.

Jos raakadatalähteen asetuksissa oleva *outputfile* on määritetty, tulostetaan tulostustiedostojen lisäksi vielä raakadatalähteen tilasta kertova tiedosto samaan hakemistoon. Tiedostonimi saadaan *outputfile*-asetuksesta lisäämällä tiedostopääte `.txt`. Tästä tiedostosta voi lukea käsiteltyjen tiedostojen tilan.

Suorituksen jälkeen raakadatan syötehakemistoihin voi lisätä uusia raakadatatiedostoja. Nämä tiedostot käsitellään seuraavalla ohjelman suorituskerralla. Raakadatalähteen tilatiedostot kertovat tilan, josta suoritusta tulee jatkaa. Kirjoitettavien tulostustiedostojen nimessä on juokseva numero, joka kasvaa eräajojen välillä.

## 5.1 Ongelmatilanteet

Ongelmatilanteiden selvittämiseen kannattaa käyttää ohjelman käynnistyksen yhteydessä *-v* parametria. Tällöin ohjelma tulostaa tietoa suorituksen edistymisestä.

Jos ohjelman tilatiedostot ovat jostain syystä epäkunnossa, niiden poistaminen voi ratkaista ongelman. Tilatiedostot kertovat edellisistä ajokerroista muuntimille tarvittavaa tietoa.

Ensimmäisellä ajokerralla muodostetaan raakadatalähteistä riippumattomia tietoja varten tiedosto `IdFactory.state` pääasetustiedostossa määritettyyn *datadir*-hakemistoon. Tämä tiedosto sisältää koko ohjelman sarjanumeron ja juoksevan sarjanumeron uusille tietueille. Tiedoston voi poistaa vain, jos kaikki muut tilatiedostot poistetaan samalla.

Jokaisen raakadatalähteen tilatiedostot tallentuvat raakadatalähteen asetuksissa olevaan *datadir*-hakemistoon. Nämä tiedostot voi poistaa, jos koko raakadata lähteen aineisto halutaan käsitellä uudestaan. Yhden raakadatalähteen kaikki tilatiedostot tulee kuitenkin poistaa samalla kertaa. Vanhat tulostustiedostot on myös syytä poistaa, jos tilatiedostot poistetaan.

# 6 Asetustiedostot

Boa on komentoriviltä ajettava ohjelma. Ohjelman käyttö tapahtuu asetustiedostojen kautta. Nämä asetustiedostot määrittävät ohjelman tarvitsemat tiedot.

## 6.1 Pääasetustiedosto

Boa tarvitsee pääasetustiedoston, joka määrittää muiden asetustiedostojen sijainnin. Boa:n käyttämä pääasetustiedosto sisältää seuraavat pakolliset tiedot:

userdir = Hakemisto, josta luetaan käyttäjien asetustiedostot

sourcedir = Hakemisto, josta luetaan raakadatalähteiden asetustiedostot

datadir = Hakemisto, jonne ohjelma tallettaa raakadatalähteistä riippumattomat tilatiedot

Esimerkki pääasetustiedostosta:

```
userdir = example/users
sourcedir = example/sources
datadir = example/data
```

Esimerkissä kuvatulla asetustiedostolla Boa lukee käyttäjien asetustiedostot hakemistosta `boafiles/users`, raakadatalähteiden asetustiedostot hakemistosta `boafiles/sources` ja tulostaa tilatietonsa hakemistoon `boafiles/data`. Kaikkien hakemistojen tulee olla ennalta olemassa. Boa ei yritä luoda puuttuvia hakemistoja automaattisesti.

## 6.2 Käyttäjien asetustiedostot

Jokainen käyttäjä kuvataan omassa asetustiedostossaan. Nämä tiedostot tulee sijoittaa hakemistoon, joka on määritetty pääasetustiedostossa käyttäjien asetustiedostojen hakemistoksi. Kaikki tämän hakemiston tiedostot luetaan ohjelmaan ja ne tulkitaan käyttäjiksi.

Yksittäisen asetustiedoston sisältö on seuraava:

name = Käyttäjän nimi

Esimerkki käyttäjän asetustiedostosta:

```
name = Firstname Lastname
```

Esimerkissä kuvattu tiedosto määrittelee käyttäjän nimeltä Firstname Lastname.

Jokaiselle käyttäjälle tulee luodaan oma asetustiedosto. Asetustiedoston nimi kirjataan raakadatalähteen asetustiedoston *user*-asetuksen arvoksi.

## 6.3 Raakadatalähteiden asetustiedostot

Jokainen raakadatalähde kuvataan omassa asetustiedostossaan. Nämä tiedostot tulee sijoittaa hakemistoon, joka on määritelty pääasetustiedostossa raakadatalähteiden asetustietojen hakemistoksi. Kaikki tämän hakemiston tiedostot luetaan ohjelmaan ja ne tulkitaan raakadatalähteiksi.

Yhden raakadatalähdettä kuvaavan tiedoston rakenne on seuraava:

name = Sisältää lähteen sisältöä kuvaavan nimen

user = Sisältää käyttäjän tiedoston nimen ilman polkua

inputdir = Hakemisto, joka sisältää kaikki raakadatalähteen syötetiedostot

outputdir = Hakemisto, jonne muunnetut tiedostot kirjoitetaan

outputfile = Kirjoitettavien tiedostojen nimi. Nimeen lisätään tarvittavat tunnisteet automaattisesti.

datadir = Hakemisto, jonne tallennetaan tämän lähteen eräajon tilatiedostot.

transformer = Käytettävä muunnin. Vaihtoehdot ovat citeseer, dblp ja qs.

printer = Tulostin muunnettavalle datalle. Vaihtoehdot ovat console ja qs.

gzip = Kirjoitettavien tiedostojen pakkaus tapahtuu asettamalla asetus arvoon true.

cutsize = Kirjoitettavien tiedostojen suurin haluttu koko ilmoitetaan tavuina. Arvo 0 ei rajaa tiedoston suurinta kokoa.

Näistä asetuksista *name, user, inputdir, outputdir, datadir, transformer* ja *printer* ovat pakollisia. Jokainen asetus voi olla mainittu vain kerran. Jokaista raakadatalähdettä kohden on siis vain yksi raakadatan syötehakemisto ja yksi käyttäjä.

Esimerkki raakadatalähteen asetustiedostosta:

```
name = Citeseer XML
user = user1.cfg
inputdir = example/source1
outputdir = example/output1
outputfile = cs
datadir = example/data
transformer = citeseer
printer = qs
gzip = false
cutsize = 0
```

Esimerkissä annetaan aluksi nimi raakadatalähteelle *name*-asetuksessa. Tieto ohjelmaa ajavasta käyttäjästä välitetään *user*-asetuksella. Tälle annettu tiedostonimi sisältää käyttäjän tiedot. Tiedosto tulee löytyä pääasetustiedostossa määritetystä käyttäjätietohakemistosta.

Esimerkin syötehakemistona toimii inputdir arvona annettu hakemisto `example/source1`. Tässä hakemistossa olevat tiedostot käsitellään ja niistä muodostuva tulostus kirjoitetaan *outputdir*-asetuksessa annettuun hakemistoon `example/output1`. Kaikkien kirjoitettujen tiedosten nimi alkaa asetuksen *outputfile* sisältämällä arvolla `cs`. Raakadatalähteen tilatiedot tallentuvat *datadir*-hakemistoon, joka esimerkissä on `example/data`.

Esimerkin syötetiedostojen muodoksi on *transformer*-asetuksessa annettu `citeseer`. Käytettäväksi tulostimeksi on valittu *printer*-asetuksessa `qs`. Pakkauksen käytöstä ilmoitetaan asetuksessa *gzip*, joka esimerkissä on laitettu pois päältä. Asetus *cutsize* on asetettu arvoon `0`, jolloin kirjoitettavan tiedoston kokoa ei ole rajoitettu.

Jokainen raakadatalähde tarvitsee oman asetustiedoston.

Kaikki hakemistot tulee olla valmiiksi tehtynä.

Käyttäjätietotiedostojen tulee löytyä pääasetustiedossa mainitussa hakemistossa.

Muuntimet lukevat tiettyä tiedostotyyppiä ja välittävät saadut tiedot tulostimelle. Yhteen raakadatalähteeseen kuuluu vain yhden tyyppisiä tiedostoja. Raakadatalähdehakemistossa olevat tiedostot käsitellään raakadatatiedostoina.