Suunnitteludokumentti

Metaxa

Helsinki 15.12.2005
Ohjelmistotuotantoprojekti
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Kurssi

581260 Ohjelmistotuotantoprojekti (6 ov)

Projektiryhmä

Väinö Ala-Härkönen
Reima Halmetoja
Antti Laitinen
Kalle Pyykkönen
Oskari Saarekas
Tuomas Tanner
Juuso Vanonen

Asiakas

Olli Niinivaara

Johtoryhmä

Juha Taina
Joni Salmi

Kotisivu

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

Sisältö

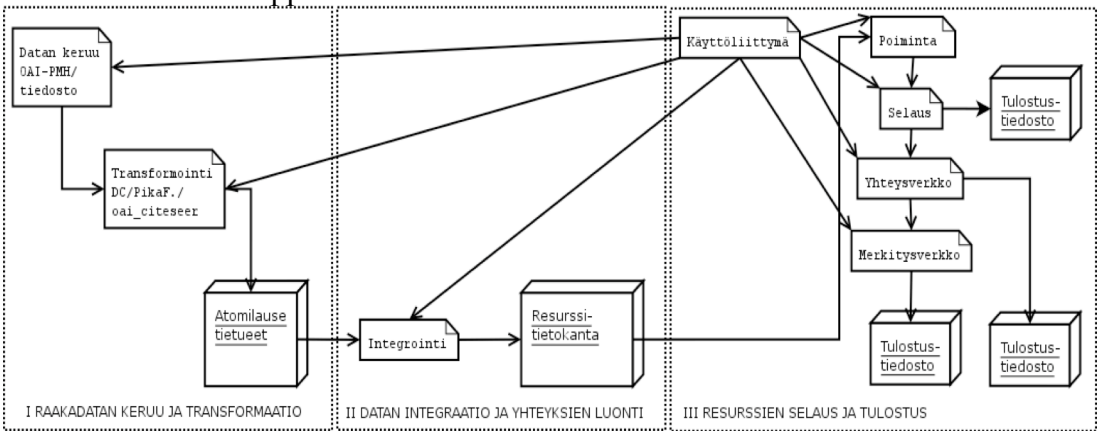
1. Johdanto

Tämä dokumentti on Metadatan hallinta (Metaxa) -projektin suunnitteludokumentti. Suunnitteludokumentissa kuvataan ohjelmiston yleisarkkitehtuuri, osajärjestelmien väliset rajapinnat sekä itse osajärjestelmien toiminta itse sillä tarkkuudella kun ne on katsottu tarpeelliseksi suunnitella ja dokumentoida koodin ulkopuolella. Käytännössä alunperin ideana on ollut se, että ohjelmistoa toteuttavat projektin jäsenet ovat voineet tehdä työtään mahdollisimman paljon toisistaan riippumatta.

Tätä dokumenttia on projektin päättyessä tarkennettu ja korjattu vastaamaan toteutunutta tilannetta – se toimii siis myös toteutusdokumenttina jatkokehittäjille jotka haluavat tietää mistä osista ohjelmisto rakentuu alkamatta käydä läpi ohjelmakoodia vaivalloisesti. Kannattaa myös huomata, että dokumentissa on kuvailtu myös ei-toteutettaviin ominaisuuksiin liittyviä tietorakenteita – esimerkiksi merkitykset resurssiverkossa. Nämä on jätetty sekä tietokantaskripteihin että luokkakaavioihin jatkokehittäjiä ajatellen, mutta niillä ei tässä ohjelmistoversiossa tehdä mitään.

2. Järjestelmäarkkitehtuuri

Ohjelmisto koostuu kolmesta erillisestä osajärjestelmästä - Raakadatan keruu ja transformointi, integraatio sekä selaus & tulostus. Nämä järjestelmät on kuvattu tarkemmin omissa kappaleissaan.



Kuva 1: järjestelmäarkkitehtuuri

Lista vaatimuksista, jotka kukin osajärjestelmä tulee toteuttamaan:

Raakadatan keruu ja transformointi -osajärjestelmän vaatimukset

K1.1 Valmiin metadatan tuonti

K1.1.1 XML Dublin Core harvestointi

K1.1.5 oai_citeseer-muotoisten tiedostojen tuonti

K1.1.6 XML Dublin Core-muotoisten tiedostojen tuonti

K1.2 Itse tehdyn metadatan tuonti

K1.2.1 Pikaformaatti-tiedostojen tuonti

K1.3.1 Raakadatan säilytys

K1.3.2 Atomilauseiden säilytys

K1.4 Lähteiden päivitys

Datan integraatio -osajärjestelmän vaatimukset

K2.1 Resurssiverkon integrointi

K2.2.1 Atomilauseiden valinta

K2.3 Integraation toistettavuus

K2.3.1 Useat resurssiverkot - käyttäjä voi valita mihin resurssiverkkoon dataa integroidaan

Selaus ja tulostus -osajärjestelmän vaatimukset

K2.3.1 Useat resurssiverkot - käyttäjä voi valita mitä resurssiverkoista tarkastellaan

K3.1.1 Resurssien hakeminen ominaisuuksien perusteella

K3.1.2 Resurssien hakeminen yhteyksien perusteella

K3.1.3 Hakuehtojen muokkaus käsin

K3.2 Resurssien selaus

K.3.2.3 Hakutulos tallennus "CSV-muodossa"

K3.2.7 Resurssiverkon puhdistus

K3.4.3 Yhteysverkon tallennus PAJEK-muodossa

3. Järjestelmän tietokantarakenne & osajärjestelmien tiedonvälitys

Järjestelmän tietokantarakenne toimii samalla eri komponenttien välisenä rajapintana. Raakadatan keruu -osajärjestelmä tuottaa transformoitua raakadataa, jota integraatio -osajärjestelmä syö ja muuttaa resurssiverkoksi. Selaus & tulostus -osajäjestelmä lukee luotua resurssiverkkoa ja muodostaa tästä visualisointeja ja tulosteita.

Tietokantarakenteen taulujen kuvauksiin on jätetty viittauksia sellaisiin vaatimuksiin, joita ei otettu mukaan lopulliseen toteutukseen. Myös näiden ei toteutettavien vaatimusten kuvaukset löytyvät vaatimusmäärittelystä. Nämä vaatimukset toimivat kuitenkin valitun tietorakenteen perusteluina ja ovat jatkokehityksen kannalta oleellisia.

3.1. Käytettävä tietokanta

- MySQL 5.0.16
- JDBC ajuri: MySQL Connector/J 3.1
- Merkistö: UTF-8

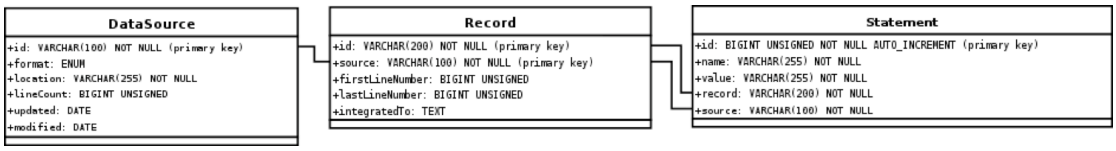
3.2. Järjestelmän yhteinen asetustiedosto

Järjestelmällä on yksi kaikkien komponenttien käyttämä properties -muotoinen asetustiedosto **dbconfig.properties**. Tiedoston merkistö on UTF-8. Java Properties-luokan kuvaus: http://java.sun.com/j2se/1.4.2/docs/api/java/util/Properties.html.

Tiedosto sisältää seuraavat tiedot:
jdbc_driver = jdbc ajurin luokkanimi
jdbc_url = jdbc urli joka määrittää yhteyden tietokantaan
jdbc_user = tietokannan käyttäjän nimi

jdbc_pass = käyttäjän salasana

3.3. Transformoidun datan tietorakenne



Kuva 2: Transformoidun datan tietorakenne

Transformoitu data sijaitsee omassa tietokannassaan erillään resurssiverkkokannoista. Datankeruuosajärjestelmä lisää uutta dataa tähän yhteen tietokantaan. Transformoidun datan vaadittavat ominaisuudet on määritelty vaatimusmäärittelyn luvussa 6.2.

Yhtäaikaisuus

Uuden tietueen syöttö tietokantaan tehdään atomisesti. Transaktio aloitetaan ennenkuin Record-taulun rivi luodaan ja päätetään kun viimeinen Recordiin liittyvä Statement-taulun rivi on luotu. Näin tiedonkeruu ja transformointi ei voi tuottaa virheellistä dataa integrointia varten. Jos integrointi ajetaan yhtäaikaisesti transformoinnin kanssa, integroitujen tietueiden tiedot ovat aina täydellisiä.

Taulujen väliset yhteydet

Taulujen väliset viiteavaimet toimivat cascades -periaatteella, eli jos taulun rivi poistetaan, poistetaan myös tähän riviin viittaavat muiden taulujen rivit.

DataSource

Tämä taulu kuvaa tietolähdettä. Jokaisella tietolähteellä on uniikki tunniste. Kun

datankeruu käynnistetään uudelle lähteelle, ensimmäiseksi luodaan lähteelle tämä taulu. Taulussa pidetään tietoa lähteen formaatista, sijainnista, raakadatarivien määrästä, viimeisestä päivityskerrasta ja viimeisestä kerrasta jolloin tietoa saatiin. Näitä tietoja käytetään valittaessa integrointiprosessiin lähteitä.

Kenttä	***Lisärajoite***	***Kuvaus***	***Vaatimukset***
id		Uniikki tunniste	
format	dcxml / quickformat_name / quickformat_document / oai_citeseer	Lähteen raakadataformaatti	K2.2.1, K2.1
location	URL	Lähteen sijainti	K2.2.1
linecount	>=0	Raakadatarivien määrä	K2.2.1
updated	>=modified	Viimeisin päivitys	K2.2.1, K1.4
modified		Viimeisin muutos	K2.2.1, K1.4

Record

Taulu kuvaa yhden raakadatatietueen. Taululla on id, joka on tietolähteen sisällä oleva uniikki tunniste. Recordiin merkitään myös mistä kohtaa raakadatatiedostoa kyseinen tietue löytyy, jotta tietueen alkuperäinen esitys saadaan palautettua.

Kenttä	***Lisärajoite***	***Kuvaus***	***Vaatimukset***
id		Uniikki tunniste	K1.1, K1.2
source		Viiteavain	
firstLineNumber	>= 0	Tietueen ensimmäisen rivin numero raakadatatiedostossa	K3.2
LastLineNumber	>= firstLineNumber, <= tietolähteen linecount	Tietueen viimeisen rivin numero raakadatatiedostossa	K3.2

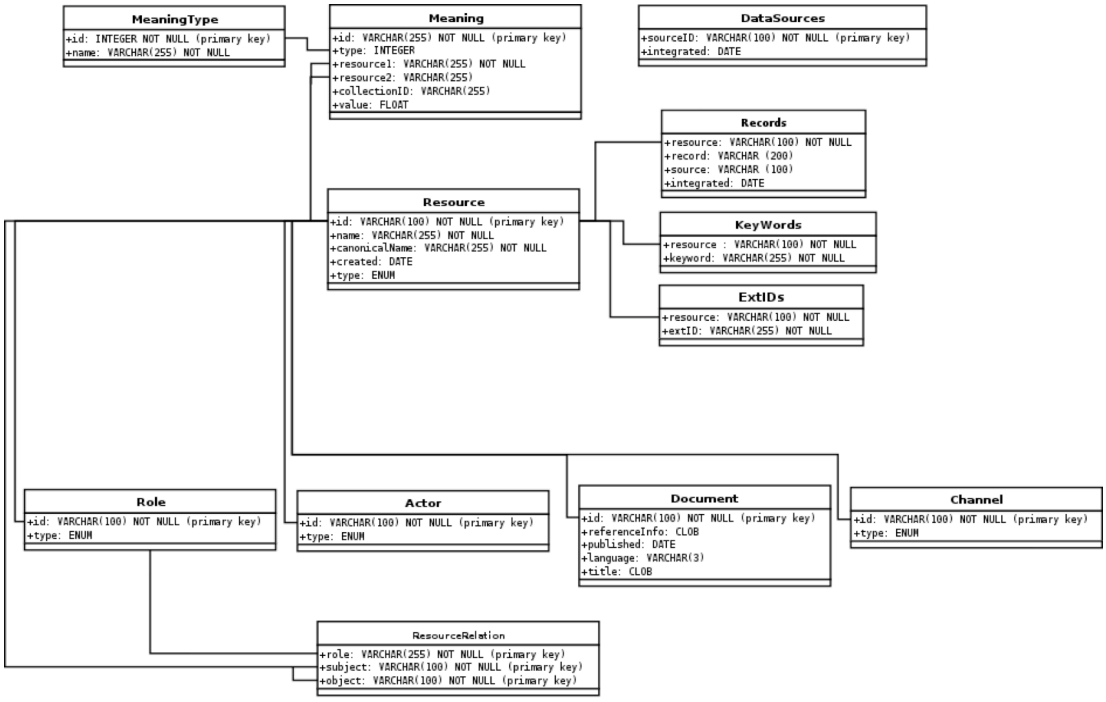
Kenttä	*Lisärajoite*	*Kuvaus*	*Vaatimukset*
integratedTo		Pilkuilla eroteltu lista resurssiverkoista joihin tietue on integroitu	K2.2.1

Statement

Tämä taulu kuvaa transformoituja atomilauseita. Record- ja source- kentät identifioivat mistä Recordista atomilause tulee. Uniikki id yksilöi jokaisen atomilauseen globaalisti. Näin resurssiverkossa on helppo viitata niihin atomilauseisiin mistä tietty resurssi koostuu. Name kertoo atomilauseen nimen, value arvon.

Kenttä	*Lisärajoite*	*Kuvaus*	*Vaatimukset*
id		Atomilauseen tunniste	
name	Yhteisen sanaston sana, kts. vaatimusmäärittely	Atomilauseen nimi	K2.1, K1.3.2
value		Atomilauseen arvo	K2.1, K1.3.2
record		Viiteavain	
source		Viiteavain	

3.4. Resurssiverkon tietorakenne



Kuva 3: Resurssiverkon tietorakenne

Resurssien ominaisuudet ja yhteydet, sekä merkitykset on määritelty vaatimusmäärittelydokumentissa.

Yhtäaikaisuus

Sekä uuden resurssin tai resurssiyhteyden luonti integrointivaiheessa, että resurssin tai yhteyden poisto selausvaiheessa on oman transaktionsa sisällä. Näin resurssien ja yhteyksien lisäys ja poisto voivat tapahtua yhtäaikaisesti.

Taulujen väliset yhteydet

Verkon resurssit ovat yhteydessä toisiinsa ResourceRelation taulun avulla. Jos jokin resurssi poistetaan, myös kaikki resurssiin liittyvät yhteydet poistetaan

ResourceRelation -taulusta. Resurssia poistettaessa poistetaan sekä specifin resurssityypin taulun rivi että yleinen Resource -taulun rivi.

3.4.1. Resurssit

Yhtä resurssia kuvaavat seuraavat taulut: Resource, Statements, Keywords, ExtID. Nämä taulut ovat kaikille resursseille yhteisiä. Statements, Keywords ja ExtID kuvaavat resurssien ominaisuuksia joita voi olla 0 tai useampi.

Resource

Taulu sisältää id -kentän joka on resurssin tunniste. Tämä on sama kuin vaatimusdokumentin näkyväTunniste. Emme siis käytä kahta rinnakkaista tunnistejärjestelmää, vaan vain asiakkaan haluamaa (jokseenkin luettavaa) tunnistetta. Jokaisella resurssilla on myös nimi & kanonisoitu nimi (Name & canonicalName) sekä luontipäivämäärä.

Kenttä	***Lisärajoitteet***	***Kuvaus***	***Vaatimukset***
id		Uniikki tunniste	K2.1, K3.4.1, K3.4.3, K3.4.4
name			K2.1, K3.1.1, K3.2.1, K3.3.1, K3.4.1
canonicalName			K2.1, K3.3.1
created		Luontipäivämäärä	K2.1, K3.1.1, K3.2.1
type	Actor / Document / Channel / Role	Kertoo resurssin tyypin	

Statements

Lista resurssiin liittyvistä atomilauseista.

Kenttä	*Lisärajoitteet*	*Kuvaus*	*Vaatimukset*
resource		Viittaus resurssiin	
statement		Resurssiin liittyvän atomilauseen ID	K2.1

Keywords

Lista resurssiin liittyvistä hakusanoista. Asiakkaan vaatimus oli, että ohjelman on tuettava kolmea hakusanaa. Tällä ratkaisulla täytämme vaatimuksen ja samalla teemme järjestelmästä laajennettavan & hakujen toteuttamisesta tehokasta.

Kenttä	*Lisärajoitteet*	*Kuvaus*	*Vaatimukset*
resource		Viittaus resurssiin	
keyword		Hakusana	K3.1.1, K3.3.1

ExtID

Taulu resurssin lisätunnisteille. (Raakadatassa voi olla resurssille useita tunnisteita).

Kenttä	*Lisärajoitteet*	*Kuvaus*	*Vaatimukset*
resource		Viittaus resurssiin	
extID		Resurssin lisätunnisteet	K2.1, K3.3.1

3.4.2. Resurssityypit

Resurssityyppejä on neljää erilaista: Actor, Document, Channel & Role. Nämä kaikki laajentavat yleistä Resource tyyppiä. Jokaista erityistä resurssityyppiä vastaa yksi Resource taulun rivi. Yhteys muodostetaan käyttämällä resurssityyppi-tauluissa samaa id -kentää kuin Resource -taulussa. Tällöin taulujen id-kenttä toimii viiteavaimena Resource-tauluun.

Actor

Tämä resurssi kuvaa toimijaa. Sen lisäkenttänä on vain type -toimijan tyyppi.

Kenttä	***Lisärajoitteet***	***Kuvaus***	***Vaatimukset***
id		Tunniste	
type	Henkilö / Organisaatio / Joku	Toimijatyyppi	K3.1.1, K3.2.1, K3.3.1

Document

Tämä resurssi kuvaa dokumenttia. Se sisältää seuraavat kentät: referenceID - pitkä tieteellisessä artikkelissa käytetty viittaustieto, published -päivämäärä, language - julkaisun kieli, title - otsikko kokonaisuudessaan.

Kenttä	***Lisärajoitteet***	***Kuvaus***	***Vaatimukset***
id		Tunniste	
referenceID		Pitkä tieteellisessä artikkelissa käytetty viittaustieto	K3.3, K3.3.1
published		Julkaisupäivämäärä	K3.1.1, K3.2.1, K3.3.1
language	ISO-koodi	Julkaisukieli	K3.1.1, K3.2.1, K3.3.1

Kenttä	*Lisärajoitteet*	*Kuvaus*	*Vaatimukset*
title		Otsikko kokonaisuudessaan	K3.1.1, K3.2.1, K3.3.1

Channel

Kanavaa kuvaava resurssi. Se sisältää vain type -kanavatyyppi lisäkentän.

Kenttä	*Lisärajoitteet*	*Kuvaus*	*Vaatimukset*
id		Tunniste	
type	Lehti / Konferenssi / Julkaisusarja / Raporttisarja / Tietokanta	Kanavatyyppi	K3.1.1, K3.2.1, K3.3.1

Role

Tämä resurssi kuvaa toimijan roolia. Sen lisäkenttänä on type -roolityyppi. Jokaiseen dokumenttiin liittyy yksi tietyntyyppinen rooli, mutta tähän rooliin voi liittyä useampi toimija. Roolit sijaitsevat toimijoiden ja muiden muiden resurssien yhteyksien välissä.

Kenttä	*Lisärajoitteet*	*Kuvaus*	*Vaatimukset*
id		Tunniste	
type	Tekijä / Julkaisija / Avustaja / Oikeuksienomistaja	Roolityyppi	K3.1.1, K3.2.1, K3.3.1

Alustusvaiheessa luodaan kaksi erityistä Role -resurssia: *BIBREF* ja *NONE*. BIBREF rooli kuvaa dokumenttiviittausta. NONE kuvaa roolin puuttumista (käytetään kanavien välisissä yhteyksissä). Näiden molempien erikoisroolien type kenttä on null.

3.4.3. Resurssien väliset yhteydet

ResourceRelation

Tämä taulu ilmaisee resurssien väliset yhteydet. ReferencingResource kertoo viittaavan resurssin, refererencedResource viitatun resurssin. Yhteyteen liittyy aina rooli. Rooleja on kahta erityistyyppiä. Id:llä 0 merkitään sitä ettei roolia käytetä - ns. null-rooli. Id:llä 1 oleva rooli merkitsee dokumenttien välistä viittausta.

Kenttä	***Lisärajoitteet***	***Kuvaus***	***Vaatimukset***
role		Rooli	K3.4.3
subject		Viite resurssiin johon rooli liittyy	K3.1.2, K3.4, K3.4.4, K3.3.1
object		Viite resurssiin	K3.1.2, K3.4, K3.4.4, K3.2.6, K3.3.1

Yhteyksien rajoitteet

Vaatimusdokumentin resurssiverkon kuvauksen mukaan vain seuraavat resurssien väliset yhteydet ovat sallittuja

subject	***role***	***object***	***huomautus***
Actor	Role	Document	
Actor	Role	Channel	
Document	1	Document	Dokumenttien välinen yhteys merkitään erityisellä BIBREF -roolilla
Channel	0	Document	Roolia ei ole, merkitään erityisellä NONE -roolilla

Näitä sallittuja yhteyksiä valvotaan integraatiovaiheessa, kun heuristiikat luovat yhteyksiä resurssien välillä.

3.4.4. Merkitysverkko

Merkitysverkko kuvataan kahdella tietokantataululla.

MeaningType

Tällä kuvataan yksi merkitystyyppi. Merkitystyypillä on näytettävä nimi ja numerotunniste. Näin nimen muuttaminen on helppoa eikä aiheuta suurta muutosta tietokantaan.

Kenttä	***Lisärajoitteet***	***Kuvaus***	***Vaatimukset***
id		Uniikki tunniste	
name		Merkitystyyppi	K3.5.*

Meaning

Meaning -taulun riveillä on oma tunniste, merkitystyyppi, tieto siitä mihin ryhmään merkitys kuuluu (collectionID) sekä merkityksen painoarvo.
Jos merkitys liittyy yhteen resursiin, laitetaan resource1 -kenttään viite tähän resurssiin. Jos merkityksellä kuvataan kahden resurssin välistä yhteyttä, resource1 merkitsee yhteyden alkua ja resource2 yhteyden kohdetta.

Kenttä	***Lisärajoitteet***	***Kuvaus***	***Vaatimukset***
id		Uniikki tunniste	K3.5
type		Merkityksen tyyppi	K3.5, K3.5.3.1

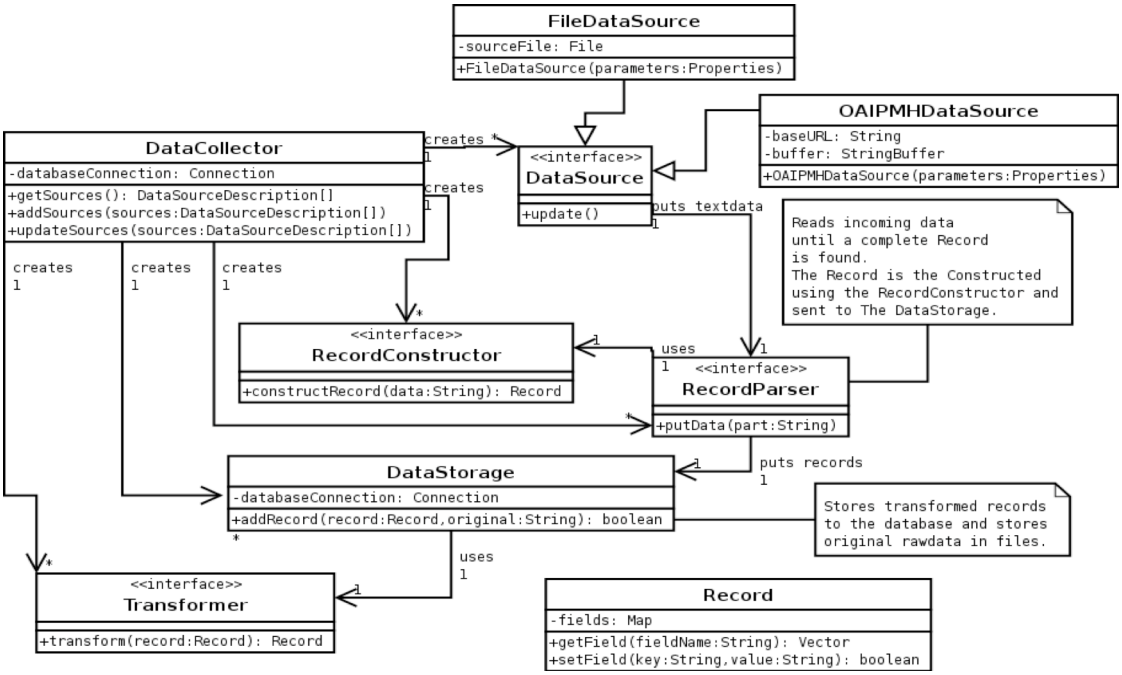
Kenttä	*Lisärajoitteet*	*Kuvaus*	*Vaatimukset*
resource1		Resurssi, johon merkitys liittyy tai yhteyteen liittyvän resurssin lähtöresurssi	K3.5.*
resource2		Yhteyteen liittyvän merkityksen kohderesurssi tai null	K3.5.*
collectionID		Merkitysryhmä	K3.5.3
value		Merkityksen painoarvo	K3.5.2

3.5. Datan formaatti create table -lauseina

Create table -lauseet, jotka luovat sekä raakadatatietokannan että resurssiverkot löytyvät ohjelmiston config-hakemistosta nimillä rawdata.sql ja resourcegraph.sql.

4. Osajärjestelmä 1: Raakadatan keruu ja transformointi

4.1. Luokkakaavio



Kuva 4: datankeruun luokkakaavio

Ylläolevassa luokkakaaviossa ei ole kuvattu kaikkia tämän osajärjestelmän luokkia ja niiden metodeita. Sen sijaan luokkakaavioon on hahmoteltu kaikki järjestelmän toiminnan kannalta tärkeimmät rajapinnat, luokat ja metodit. Kaavion tarkoituksena on antaa yleiskuva järjestelmän toiminnasta.

4.2. Pääkomponentit

Raakadatan keruu ja transformointi koostuu seuraavista pääkomponenteista:

- DataCollector -luokasta. Kontrolleri datan keruuta varten.
- DataSource -rajapinnasta jolle on toteutus jokaista lähdetyyppiä kohden (OAI-PMH, tiedosto jne.)

- RecordParser -rajapinnasta jolle on toteutus jokaista lähdetyyppiä tai dataformaattia kohden (Riippuu siitä kumpi määrittelee tietueiden erottelun).
- RecordConstructor -rajapinnasta jolle on toteutus jokaista dataformaattia varten. Kukin toteutus osaa luoda Record-olion jostakin dataformaatista (Record-olion selitys myöhemmin).
- DataStorage -luokasta jonka avulla RecordParser-tyyppiset instanssit tallettavat tuloksensa
- Transformer -rajapinnasta jolle on toteutus jokaista dataformaattia kohden. Kukin toteutus osaa muuntaa yhden dataformaatin yhteiseen atomilauseformaattiin.

Luokka: DataCollector

- Tarjoaa mahdollisuuden datalähteiden lisäämiseen, päivittämiseen ja poistamiseen.
- DataCollector luo kutakin järjestelmään lisättyä datalähdettä kohden yhden DataSource-, RecordParser-, RecordConstructor- ja Transformer-toteutuksen sekä yhden DataStorage instanssin.

Rajapinta: RecordParser

- Rajapinta, jonka toteutukset tuntevat jonkin lähdetyypin tai dataformaatin tavan erotella yksittäiset tietueet.

Rajapinta: RecordConstructor

- Rajapinta, jonka toteutukset osaavat luoda Record-olion jonkun dataformaatin mukaisesta tekstidatasta.

Rajapinta: DataSource

- Rajapinta, jonka toteutukset toteuttavat jonkin datalähdetyypin lukemisen. Datalähdetyyppejä ovat OAI-PMH ja tiedosto

Luokka: DataStorage

- Tarjoaa palvelun datan tallentamiselle. Kunkin datalähteen raakadata tallennetaan sellaisenaan tiedostoon ja yhtenäiseen muotoon transformoitu data tallennetaan tietokantaan.

4.3. Record-olioiden muoto

Record-olioiden kenttien avaimet nimetään seuraavasti:

- Sisäinen atomilausemuoto: avaimina käytetään suoraan yhteisen sanaston termejä (kts. vaatimusmäärittely), kapitalisointi kuten vaatimusdokumentin listassa

- Dublin Core: käytetään XML-tageissa yleensä esiintyvää muotoa (dc-nimiavaruus) seuraavasti:
    - DC 1.1 Simple: dc:title, dc:creator, dc:subject, dc:description, dc:publisher, dc:contributor, dc:date, dc:type, dc:format, dc:identifier, dc:language, dc:relation
    - DC 1.1. Qualified (niiltä osin kuin pitää tukea): dc:bibliographicCitation, dc:issued, dc:references, dc:isReferencedBy, dc:rightsHolder

- oai_citeseer: käytetään DC 1.1-tageja edellämainitusti ja sen lisäksi vaatimusdokumentin määrittelemät citeseer-laajennukset seuraavaan muotoon muokattuna, katso todelliset jäsennettävät tagit esimerkkidatasta:
    - identifier -> oai_citeseer:identifier
    - oai_citeseer:author attribuutti name: oai_citeseer:authorName
    - oai_citeseer:author alielementti affiliation: oai_citeseer:authorAffiliation

    - oai_citeseer:relation type="References" alielementti oai_citeseer:uri: oai_citeseer:relationReferences
    - oai_citeseer:relation type="Is Referenced By" alielementti oai_citeseer:uri: oai_citeseer:relationIsReferencedBy

- Pikaformaatti:
    - Pikaformaatti dokumenteille: documentqf:identifier, documentqf:creator, documentqf:title, documentqf:publisher, documentqf:publication, documentqf:publishedYear, documentqf:referenceIdentifier, documentqf:isReferencedBy, documentqf:references, documentqf:keywords
    - Pikaformaatti nimille: nameqf:actorUnknown, nameqf:person, nameqf:organization, nameqf:roleUnknown, nameqf:publisher, nameqf:channelUnknown, nameqf:magazine, nameqf:conference, nameqf:publicationSet, nameqf:reportSet, nameqf:database

4.4. Olioiden yhteistyö

- DataCollector luokan updateSources(DataSourceDescription[] sources) käynnistää kunkin taulukossa määritellyn DataSource -rajapintaluokan implementaatioiden update()-metodin. Kun update() -metodi on suoritettu ilman virheitä, niin DataCollector päivittää tietokantaan updated-sarakkeen arvon DataSource-taulussa kyseistä lähdettä vastaavalla rivillä. Jos päivityksessä ilmenee virheitä updated-saraketta ei päivitetä.
- DataSource lukee tietolähteestä dataa ja kutsuu RecordParserin putData(String part) -metodia, kun yksi puskurillinen on luettu tietolähteestä.
- RecordParserin putData()-metodi lisää saadun merkkijonon oman puskurinsa perään ja skannaa puskurin etsien kokonaista tietuetta.
    - Jos kokonaista tietuetta ei löydy, niin mitään ei tehdä.
    - Kun kokonainen tietue löytyy, jäsentää RecordParser sen tuntemansa RecordConstructor-toteutuksen avulla Record -olioksi ja kutsuu DataStoragen addRecord(Record record, String original) metodia.

        - RecordConstructor luo raakadatatietueelle lähteen sisällä yksikäsitteisen tunnisteen, jonka avulla myöhemmin noudetut samat tietueet voidaan tunnistaa.
- DataStorage tallentaa tietueet ja raakadatan seuraavasti
    - addRecord kutsuu Transformerin transform(Record) -metodia, joka transformoi olemassa olevan Recordin uudeksi yhteiseen sanastoon muutetuksi Recordiksi.
    - addRecord tekee haun kantaan tietueen tunnisteella
        - Jos ei löydy, niin tehdään INSERT ja tallennetaan raakadata tiedoston loppuun
        - Jos löytyy, käydään kaikki tietueeseen riippuvat Statement taulun atomilauseet läpi ja verrataan niitä Recordin vastaaviin arvoihin.
            - Jos kaikki arvot samoja ja atomilauseiden lukumäärä vastaa toisiaan, ei tehdä mitään (=tieto on duplikaatti)
            - Jos arvot eroavat tai atomilauseiden lukumäärä ei vastaa Recordin lukumäärää, vanhat atomilauseet uusilla Recordin atomilauseilla. Tällöin myös nollataan Record taulun integratedTo -kenttä (koska kyseinen tietue pitää integroida uudestaan)
            - Tällöin tallennetaan raakadata tiedostoon ja päivitetään firstLineNumber & lastLineNumber vastaamaan uutta tiedon sijaintipaikkaa.

4.5. Datalähdekohtainen asetustiedosto

Kutakin järjestelmään lisättyä datalähdettä kohden on olemassa yksi asetustiedosto. Luokka DataCollector on vastuussa näistä tiedostoista.

Tiedosto on seuraavanlainen:

Formaatti: xml

Sisältää seuraavat tiedot:

- Datalähteen tunniste
- Datalähteen sijainti
- Käytettävä DataSource-toteutus
- Käytettävä RecordParser-toteutus
- Käytettävä RecordConstructor-toteutus
- Käytettävä Transformer-toteutus

DTD:

```
<!DOCTYPE METADATASOURCE [

<!ELEMENT SOURCE(ID,LOCATION, DATASOURCE, RECORDPARSER,
RECORDCONSTRUCTOR, TRANSFORMER)>

<!ELEMENT ID (#PCDATA)>
<!ELEMENT LOCATION (ORIGINAL)>
<!ELEMENT ORIGINAL (#PCDATA)>

<!ELEMENT DATASOURCE(CLASS, PARAMETERS?)>
<!ELEMENT RECORDPARSER(CLASS, PARAMETERS?)>
<!ELEMENT RECORDCONSTRUCTOR(CLASS, PARAMETERS?)>
<!ELEMENT TRANSFORMER(CLASS, PARAMETERS?)>

<!ELEMENT CLASS(#PCDATA)>

<!ELEMENT PARAMETERS(PARAMETER+)>
<!ELEMENT PARAMETER(#PCDATA)>
<!ATTLIST PARAMETER NAME CDATA #REQUIRED>

]>
```

Esimerkkejä:

Esimerkki 1

```
<source>
        <id>http://oaipmh.somerepository.org/OAI</id>
        <location>
                <original>http://oaipmh.somerepository.org/OAI</original>
        </location>
        <datasource>
                <class>maito.datacollecting.oaipmh.OAIPMHDataSource</class>
                <parameters>
                        <parameter name="metadata_prefix">oai_dc</parameter>
                </parameters>
        </datasource>
        <recordparser>
                <class>maito.datacollecting.oaipmh.OAIPMHRecordParser</class>
        </recordparser>
        <recordconstructor>
                <class>maito.datacollecting.dcxml.DCXMLRecordConstructor</class>
        </recordconstructor>
        <transformer>
                <class>maito.datacollecting.dcxml.DCXMLTransformer</class>
        </transformer>
</source>
```

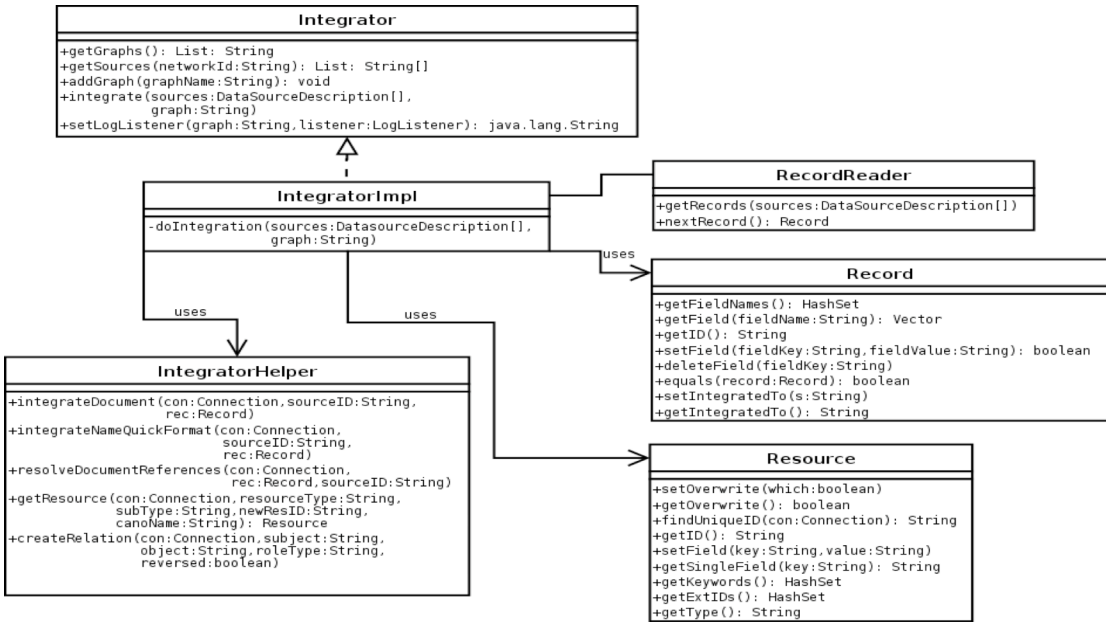
Esimerkki 2

```
<source>
        <id>file:///data/data.txt</id>
        <location>
                <original>file:///data/data.txt</original>
        </location>
        <datasource>
```

```
            <class>maito.datacollecting.file.FileDataSource</class>
        </datasource>
        <recordparser>
            <class>maito.datacollecting.quickformat.QuickformatRecordParser</class>
        </recordparser>
        <recordconstructor>
<class>maito.datacollecting.quickformat.QuickformatNameRecordConstructor</class>
        </recordconstructor>
        <transformer>
            <class>maito.datacollecting.quickformat.QuickformatTransformer</class>
        </transformer>
</source>
```

5. Osajärjestelmä 2: Metadatan integraatio

5.1. Luokkakaavio



Kuva 5: Integraation luokkakaavio

5.2. Pääkomponentit

Metadatan integraatio koostuu seuraavista pääkomponenteista:

Luokka: Integrator

- Rajapintaluokka käyttöliittymän ja integroinnin välillä

Luokka: IntegratorImpl

- Tarjoaa palvelut kannassa olevien resurssiverkkojen listaukseen, Resurssiverkon sisällön listaukseen ja uusien resurssiverkkojen lisäykseen
- Muodostaa tietokantayhteyden dbconfig.properties-tiedoston perusteella
- Lataa integraation asetukset fieldmap_*.properties-tiedostoista
- Käynnistää ja hallitsee integraatioprosessia
- Välittää integraation lokitiedot käyttöliittymälle

Luokka: IntegratorHelper

- Tarjoaa apumetodit integraation eri vaihdeiden toteuttamiseen.

Luokka: RecordReader

- Hakee annetusta lähteestä peräisin olevia tietueita atomilausekannasta ja palauttaa Record olioita käsiteltäväksi IntegratorImpl-luokalle

Luokka: Record

- Kuvaa yhtä raakadatatietuetta
- Tarjoaa aksessorit tietueen tietojen kyselyyn ja muokkaukseen

Luokka: Resource

- Kuvaa yhtä resurssia.
- Luokka tarjoaa metodit resurssin lataamiseen tietokannasta, sen tietojen päivittämiseen ja lukemiseen sekä resurssin tallentamiseen.
- Luokassa myös apumetodi, joka etsii tietokannasta uniikin muodon luokalle annetusta tunnisteesta

5.3. Asetustiedostot

Asetustiedostot kertovat mihin tietokannan tauluun ja mihin taulun kenttään kukin atomilausedatan lause kirjataan.

fieldmap_document.properties

Tiedostomuoto:
Raakadata-avain = Taulu.kenttä

fieldmap_actorchannel.properties

Tiedostomuoto:
Raakadata-avain = Resurssityyppi (Actor vai Channel),alatyyppi,roolityyppi

fieldmap_qnformat.properties

Tiedostomuoto:
Raakadata-avain = Taulu.kenttä,Resurssityyppi (Actor vai Channel),alatyyppi

5.4. Olioiden yhteistyö

- IntegratorImpl integrate -metodi käynnistää sisäisen doIntegration -metodinsa omassa säikeessään.
- doIntegration kontrolloi integrointiprosessia
- Aluksi luodaan tietokantayhteydet sekä raakadata-tietokantaan että resurssiverkkoon sekä alustetaan RecordReader.
- Kaikki integroitavat raakadatalähteet käsitellään yksi toisensa jälkeen
- Resurssiverkkoon lisätään tieto että tämä datalähde on integroitu
- Jokainen datalähteen tietue käydään läpi - tarkastetaan onko tietue jo integroitu tähän resurssiverkkon: jos on ohitetaan, jos ei lisätään tieto tietueseen
- Jos datalähde on tyyppiä quick_format_name suoritetaan tietueelle IntegratorHelperin integrateNameQuickFormat metodi joka suorittaa nimiformaatin integroinnin.
- Muutoin suoritetaan IntegratorHelperin integrateDocument joka integroi dokumentin ja luo tietueesta myös muut ei-dokumenttiresurssit
- Kun kaikki tietolähteen tietueet on integroitu, luodaan dokumenttiresurssien väliset yhteydet IntegratorHelperin integrateDocument -metodilla

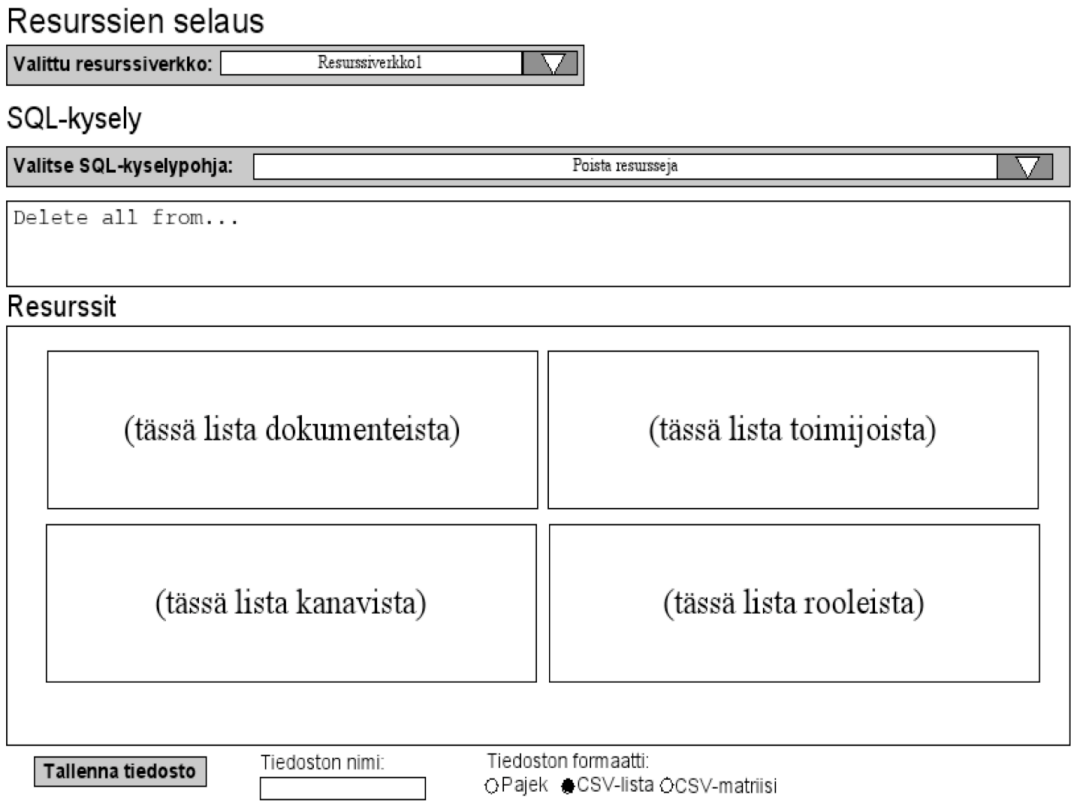
6. Osajärjestelmä 3: Metadatatiedon selaus ja tulostus

Tämä osajärjestelmä tarjoaa graafisen käyttöliittymän resurssitietokannan sisällön selaamiseen ja muokkaamiseen sekä näkymien tallentamiseen erilaisiin tiedostoformaatteihin. Osajärjestelmä lukee ja muokkaa integraatio-osajärjestelmän luomia resurssiverkkoja jotka ovat kukin omassa tietokannassaan.

6.1. Käyttöliittymä

Käyttöliittymä koostuu yhdestä ikkunasta, josta tietokantaan voidaan kohdistaa SQL-kyselyjä ja kyselyjen tuloksia voidaan tarkastella. Kyselyjen tulokset jaetaan neljään eri ali-ikkunaan, joista kukin sisältää yhtä tyyppiä tuloksena saaduista resursseista. Ikkuna tarjoaa myös mahdollisuuden resurssinäkymän tallentamiseksi tiedostoon ja mahdollisuuden valita tallennettavan tiedoston formaatti.

Ikkunassa on kaksi pudotusvalikkoa. Toisella voidaan valita valmiita SQL-kyselyitä pohjaksi eri tyyppisiä toimintoja varten. Näitä ovat esimerkiksi resurssien haku ja resurssien poisto. Toisella valikolla valitaan käytettävä resurssiverkko, eli toisin sanoen tietokanta, jota halutaan käyttää.



Kuva 6: käyttöliittymähahmotelma

Neljä resursseja näyttävää ali-ikkunaa ovat *javax.swing.JEditorPane*-luokan instansseja, jotta niiden sisältö voidaan luoda html-merkkauskielen avulla. Tämä onnistuu kutsumalla JEditorpane:n setContentType() -metodia ja annetaan sille parametriksi "text/html".

6.2. Käyttöliittymän SQL-kyselypohjat

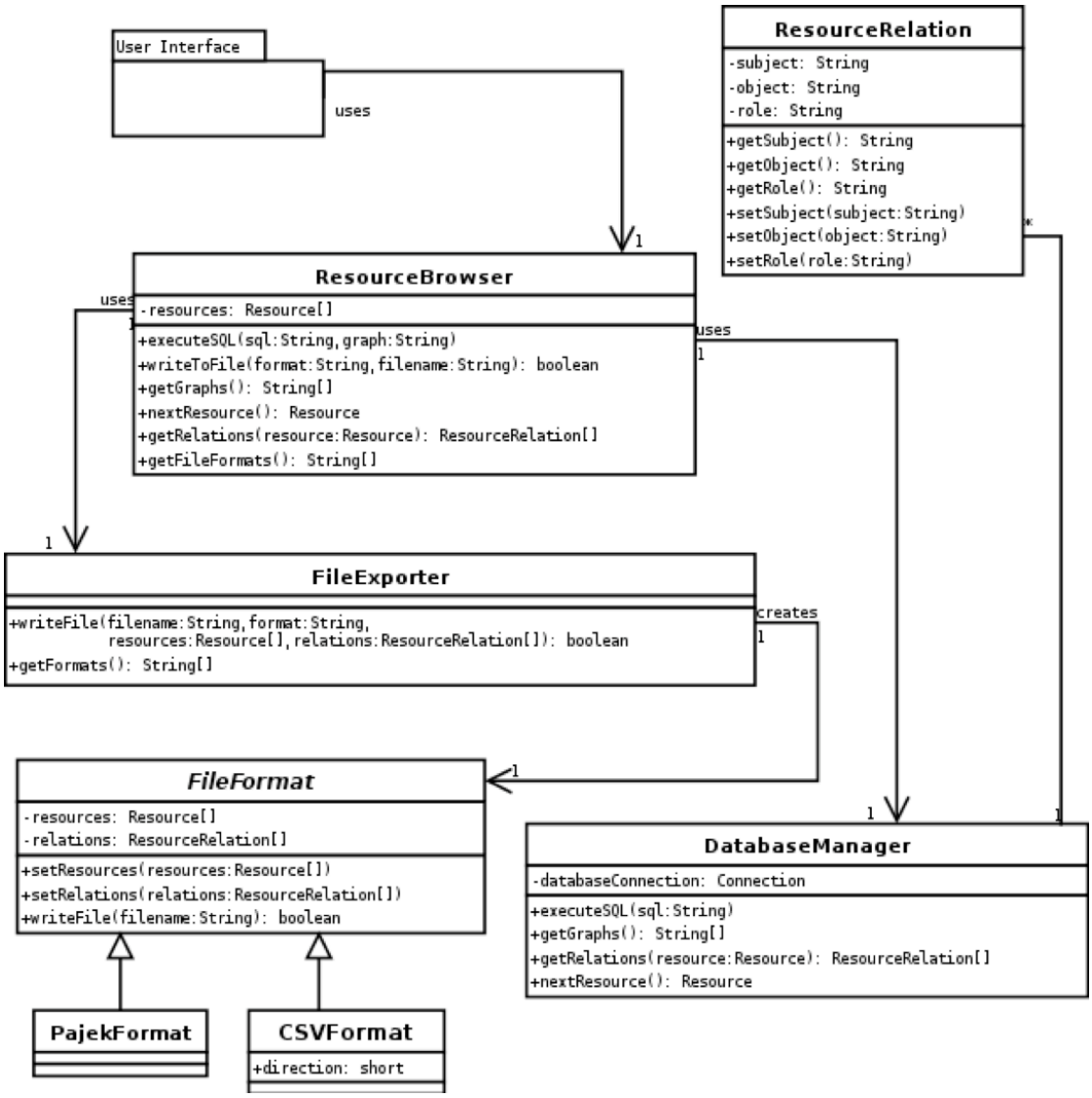
Kyselyt sijaitsevat konfiguraatiotiedostossa joka on xml-muotoinen. Se liittää eri käyttötarkoituksiin SQL-kyselyn. Käyttöliittymä lukee dropdown-menua varten tiedostosta kaikki erilaiset kyselyt.

Esimerkkitiedosto:

```
<examplequeries>
	<query>
		<menutext>Fetch all resources</menutext>
		<sql>
		<![CDATA[
			select * from Resource;
		]]>
		</sql>
	</query>
	<query>
		<menutext>Remove resources that are not related to other
		 resources</menutext>
		<sql>
		<![CDATA[
			delete from Resource where id NOT IN (select subject from
			 ResourceRelation);
		]]>
		</sql>
	</query>
</examplequeries>
```

Kukin menutext-elementti sisältää tekstin joka näytetään pudotusvalikossa kyselypohjaa valittaessa.

6.3. Logiikan toteuttavat luokat



Kuva 7: selauksen luokkakaavio

Luokkien tarkemmat kuvaukset:

ResourceBrowser

Käyttöliittymä suorittaa kaiken toimintonsa tämän luokan avulla. Luokka delegoi käyttöliittymän pyynnöt muille luokille. Tarjoaa:

- Kyselyn suorittamispalvelun jolla voi hakea tai tuhota resursseja (Jos kysely on muotoa "delete" luokka poistaa tietokannasta ja selaustuloksesta valitut resurssit. Jos kysely on muotoa "select", luokka valitsee tietokannasta kyseiset resurssit selaustulokseen.
- Tiedon tallennuspalvelun, jolla voi tallentaa resurssit tiedostoon.
- Resurssien palautuspalvelun joka palauttaa ResourceBrowser:ssa olevat resurssit
- Tallennusformaattien palautuspalvelun, joka palauttaa mahdolliset tallennusformaatit

Resource

Tämä luokka vastaa resurssitietokannan taulua Resource. Kukin luokan instanssi tietää oman tyyppinsä ja siten sisältää yhden Resource-taulun rivin tiedot sekä tiedot yhdestä Resource-taulua laajentavasta taulusta: Role, Actor, Document tai Channel.

ResourceRelation

Tämä luokka vastaa resurssitietokannassa olevaan ResourceRelation -taulua. Jokainen luokan instanssi kuvaa yhden rivin taulussa ja sisältää kaikki rivillä olevat tiedot.

DatabaseManager

Huolehtii tietokantayhteydestä ja SQL-kyselyiden suorittamisesta. Kun executeSQL() -metodilla on suoritettu jokin SQL-kysely, voidaan sen jälkeen kutakin tulosriviä vastaava Resource-olio hakea nextResource() -metodia käyttämällä. Kun jokainen tuloksen riveistä on palautettu Resource -oliona, palautetaan null. Mikäli suoritettu

kysely ei ole select-lause palautetaan myös null (jos kyseessä on esimerkiksi delete-lause).

FileFormat

Rajapinta jonka toteuttavat luokat edustavat kukin yhtä tiedostoformaattia johon resurssinäkymä voidaan tallentaa. setResources()- ja setRelations()-metodeilla voidaan asettaa tallennettavat resurssit ja yhteydet. writeFile() -metodi tallentaa kaikki lisätyt resurssit ja yhteydet (ne joiden subjekti- ja objektiresurssit löytyvät resurssilistasta) tiedostoon.

FileExporter

Huolehtii SQL-kyselyn aikaansaaman resurssinäkymän tallentamisesta tiedostoon. Luo FileFormat -rajapinnan toteuttavan luokan syöttäen sille talletettavat resurssit ja relaatiot.

Asetustiedosto (exportformats.properties) sisältää avaimina tuettavien formaattien nimet, jotka FileExporter tunnistaa ja palauttaa getFormats() -metodilla. Kunkin avaimen arvona on kyseisen formaatin FileFormat-rajapinnan toteuttavan luokan nimen.

6.4. Olioiden yhteistyö

- Käyttöliittymä käyttää ResourceBrowser -luokkaa kaikkien toimintojen suorittamiseksi ja datan saamiseksi.
- ResourceBrowser delegoi toimintojen suorittamisen FileExporter- ja DatabaseManager-luokille.
- FileExporter luo haluttua tiedostoformaattia vastaavan FileFormat-toteutuksen joka suorittaa varsinaisen tiedostoon tallentamisen.