TEKNILLINEN KORKEAKOULU
Viestintätekniikka
Diplomityö

Tekstiaineiston ontologiaperustainen indeksointi ja haku

Espoossa 28.3.2008

Matias Frosterus
matias.frosterus@tkk.fi
55491N

TEKNILLINEN KORKEAKOULU **TIIVISTELMÄ**
Automaatio- ja systeemitekniikan osasto

Tekijä, työn nimi

Matias Frosterus

Tekstiaineiston ontologiaperustainen indeksointi ja haku

Päivämäärä: 28.3.2008 Sivumäärä: 77 + 8

Osasto	Professuuri
Automaatio- ja systeemitekniikan osasto	AS-75 Viestintätekniikka
Työn ohjaaja Professori Eero Hyvönen	

Informaation lisääntyessä yhteiskunnassa vaaditaan sen tehokasta käsittelyä yhä enemmän ammattilaisten lisäksi myös tavallisilta käyttäjiltä. Tällöin luonnollinen pyrkimys on yksinkertaistaa ja automatisoida tiedonhakuprosessia mahdollisimman paljon, johon semanttisen webin tekniikat tarjoavat uusia mahdollisuuksia.

Tässä diplomityössä tutkittiin mahdollisuuksia dokumentin laajentamisen ja ontologisten käsitteiden hyödyntämisen kautta parantaa tiedonhakuprosessia tekstipohjaiseen aineistoon, kuten sanomalehtiarkistoon. Tätä tarkoitusta varten luotiin automaattinen annotointi ja hakusovellus Airo, joka suorittaa jonkin annetun ontologian pohjalta dokumentin laajennuksen. Tämä tapahtuu ontologisella käsiteklusteroinnilla, jossa jonkin käsitteen esiintyminen tekstissä nostaa myös ontologian hierarkiassa läheisten käsitteiden painoa kyseistä dokumenttia indeksoitaessa ja haettaessa.

Järjestelmän testit osoittivat, että käsitehaku yhdistettynä sanahakuun laskee haun tarkkuutta, mutta nostaa saantia. Sen sijaan hybridimenetelmä dokumentin- ja kyselyn laajennuksesta, jossa perinteisen sanahaun tuottamien dokumenttien käsitteillä suoritetaan laajentava haku, nosti saantia tarkkuuden kärsimättä. Luotu järjestelmä on ontologiariippumaton ja jokaisen ontologian tuottamat käsitteistykset talletetaan omaan indeksiinsä, jolloin niitä voidaan hakea erikseen.

Avainsanat: dokumentin laajennus, indeksointi, Semanttinen Web

HELSINKI UNIVERSITY OF TECHNOLOGY **ABSTRACT OF MASTER'S THESIS**
Department of Automation and Systems Technology

Author, Name of the Thesis

Matias Frosterus

Ontology-based indexing and retrieval of textual content

Date: March 28th, 2008 Number of pages: 77 + 8

Department	Professorship
Automation and Systems Technology	AS-75 Media Technology
Instructor	
Professor Eero Hyvönen	

With the increase of the overall amount of information in a society, more efficient utilization of information is required from professionals and laymen alike. Therefore it is natural to facilitate and automate the information retrieval process as much as possible, which is a central goal of the technologies of the Semantic Web.

This thesis studies the possibilities of using techniques such as document- and ontological query expansion to improve the information retrieval process of textual data such as articles of a newspaper archive. An automatic annotation and search application Airo was created for this purpose. It performs document expansion based on a given ontology by concept clustering. This means that the occurrence of an ontological concept increases the weights of other, hierarchically and otherwise related concepts in the search index.

Tests conducted on the system showed that concept-based search coupled with word-based search increases the yield while decreasing the precision. However, a hybrid system of document and query expansion, where the results of a traditional word-based search are expanded according to the concepts in the retrieved documents, increases the yield without adversely affecting the precision. The system works with arbitrary ontologies, and annotations from each ontology are stored in a separate index, so they are all independently searchable.

Key words: document expansion, indexing, Semantic Web

SISÄLLYSLUETTELO

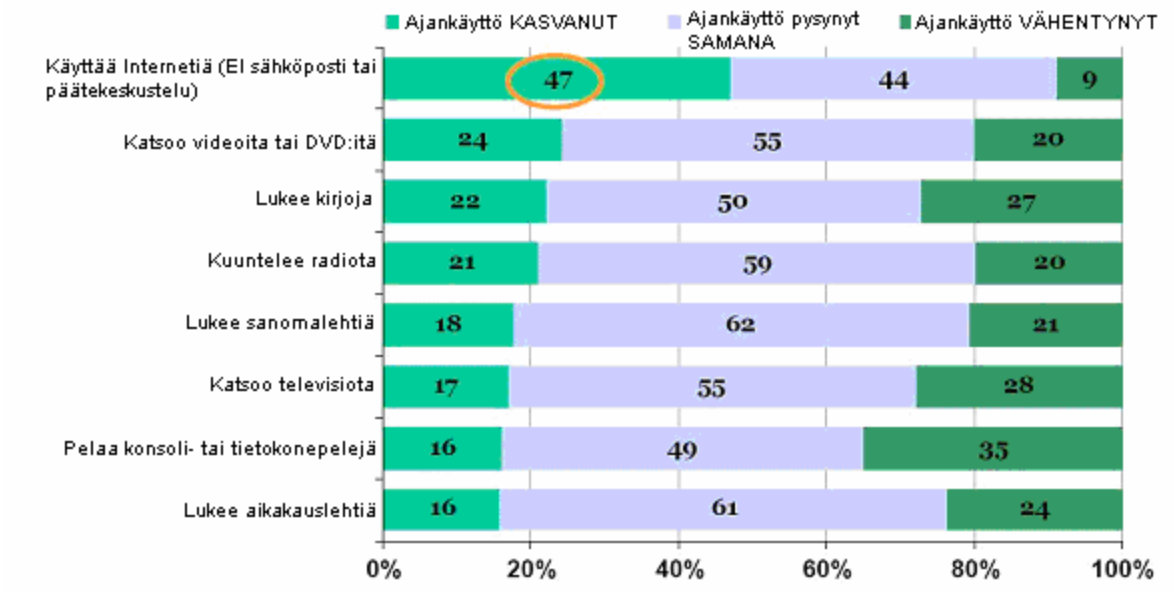
1 JOHDANTO

Viime vuosina Internetin merkittävyys tiedonhaun kannalta on yhä vain korostunut /44/. Tiedon määrä kasvaa kiihtyvällä vauhdilla ja WWW:stä on muodostunut yhä tärkeämpi kanava tämän tiedon käsittelemiseen ja löytämiseen. Hakukoneteknologioiden myötä tiedonhaku on helpottunut eikä vaadi enää samanlaista perehtymistä aiheeseen kuin ennen Internetiä. Kuitenkin tiedon lisääntyessä myös epäolennaisen tiedon määrä lisääntyy ja tyypilliseen, yhdellä tai kahdella hakusanalla suoritettuun hakuun saadaan usein vastaukseksi jopa miljoonia dokumentteja. Ongelmaksi onkin muodostunut oleellisen tiedon seulominen valtavista tulosjoukoista eikä niinkään haluttua aihetta käsittelevien dokumenttien löytäminen ylipäänsä /45/.

Perinteisesti olennaisten dokumenttien löytäminen suuresta tulosjoukosta on toteutettu lisäämällä alkuperäiseen kyselyyn tarkentavia hakusanoja eli manuaalisella kyselyn laajentamisella /13/. Tämä ei kuitenkaan ole yhtä yksinkertainen prosessi kuin alkuperäinen tiedonhaku ja vaatii asiantuntemusta niin hakuprosesseista kuin haettavana olevasta tietokannastakin. Tätä prosessia automatisoimaan on kehitetty erilaisia automaattisia ja interaktiivisia kyselynlaajennusalgoritmeja /13/, mutta näiden tehokkuutta aina rajaa se, ettei tietokone ymmärrä hakemansa tekstin sisältöä ja merkitystä.

Perinteisten hakumenetelmien ja nykyisen Internetin infrastruktuurin rinnalle ja korvaajaksi on kaavailtu semanttista webiä /3/, joka pyrkii tuottamaan koneymmärrettävää informaatiota lisäämällä tietoon sen merkityksen. Semanttiseen muotoon saatettu tietokanta sallii huomattavasti korkeamman tason automaation tiedon jäsentämisessä, koska kone kykenee tehokkaammin vastaamaan käyttäjän tarpeisiin. Tiedonhaussa tämä tarkoittaisi yhä paremmin käyttäjän toiveet täyttäviä, helpommin karsittavia ja relevantimpia tulosjoukkoja. /1/

Internetin ja muun sähköisen median suosion kasvaessa tarvitsevat myös perinteisemmät tiedonvälityskanavat kuten sanomalehdet uusia lähestymistapoja toimintansa jatkamiseen. Maailmanlaajuisesti sanomalehtien levikki on ollut jo pitkään laskussa /44/ ja kuva 1 esittää eri medioiden käyttöasteen muutoksen Yhdysvalloissa vuonna 2004. Tutkimuksessa kysyttiin eri medioihin liittyen käyttävätkö haastatellut enemmän, vähemmän vai saman verran aikaa päivässä edelliseen vuoteen verrattuna. Kuvassa on esitetty vaalean vihreällä enemmän aikaa käyttävien osuus, vaalean sinisellä ajankäytöltään muuttumattomien osuus ja tumman vihreällä vähemmän aikaa käyttävien osuus. Internetin nähdään olevan merkittävässä kasvussa, kun taas lähes kaikkien muiden medioiden käyttöaika on pysynyt lähes samana tai on laskussa.



Kuva 1. *Medioiden käytön muutos USA:ssa 2004 /44/*

Kuitenkin tarvetta asiantuntevasti toimitetulle aineistolla tulee aina olemaan. Tiedon ja tietolähteiden määrän kasvaessa tiedon luotettavuus nousee yhä tärkeämpään asemaan ja sanomalehtitoimituksilla on jo olemassa tehokas infrastruktuuri tiedon käsittelylle ja sen luotettavuuden varmistamiselle.

Useiden sanomalehtien arkistot on muunnettu suurelta osin digitaaliseen muotoon ja perinteisen paperijulkaisun oheen monet sanomalehdet julkaisevat myös verkkolehteä. Kaikki nykyään tuotettu materiaali on tyypillisesti digitaalisessa muodossa, josta sen arkistointi on periaatteessa vaivatonta. Näistä digitaalisista arkistoista on muodostunut arvokas informaation lähde laajalle käyttäjäjoukolle tiedonhallinnan ammattilaisista tavallisiin kuluttajiin, joista viimeksi mainituilla ei ole koulutusta tiedonhakuun. Tämän vuoksi laajojen arkistojen hakujärjestelmiä on kehitettävä sellaisiksi, että ne tuottaisivat yhä helpommin ja paremmin olennaista informaatiota tiedonhakijan taidoista riippumatta.

Tarkemmin määriteltynä ongelmia nykyiselle arkistohaulle tuottavat sanahaun ilmaisuvoiman riittämättömyys ja sanahakuprosessin voimakas iteratiivisuus /13/. Käytännössä hyviä tuloksia saadakseen on hakijan suoritettava haku moneen kertaan hakutermejä vaihdellen löytääkseen kaiken kannaltaan oleellisen tiedon ja rajatakseen pois epärelevanttien dokumenttien massan. Lisäksi sanomalehtiarkistojen ja toimitusten välinen yhteistyö on hankalaa, koska yhteiset standardit ovat harvassa ja usein uutisten tuottajien ja niitä arkistoivien välillä ei vallitse täydellistä yhteisymmärrystä toimintatavoista esimerkiksi artikkelien asiasanoitukseen liittyen.

Tässä diplomityössä käsitellään suurten tietomäärien automaattista sisällönkuvailua semanttisen webin edellyttämiin muotoihin sekä sen vaatimuksia ja mahdollisuuksia artikkelimuotoiselle uutisaineistolle. Työhön liittyen kehitettiin Java-pohjainen sovellus, Airo, jolla voidaan luoda perusta muulle ontologioita hyödyntävälle jatkokäsittelylle ja yhä tehokkaammalle aineiston hyödyntämiselle.

Tutkimuksen keskeiset tutkimusongelmat ovat:

- Sanahaun ilmaisuvoiman riittämättömyys: sanahakuun saadaan vastauksena vain sellaisia dokumentteja, joissa hakutermit esiintyvät sellaisinaan. Lisäksi, jos hakutermi vastaa useampaa käsitettä, ei haluttua käsitettä sisältäviä dokumentteja voida erottaa sellaisista, joissa termi esiintyy eri merkityksessä.
- Samaa aihetta käsitteleviä dokumentteja ei välttämättä saada vastauksena samoihin sanahakuihin
- Uutisaineistoja selailevaa henkilöä palvelevat linkit muihin artikkeleihin tai esimerkiksi ilmoituksiin tehdään termien, ei käsitteiden avulla, jos ylipäänsä mitään linkkien suosittelua suoritetaan

Nämä ongelmat pyritään ratkaisemaan ontologiaperustaisella indeksoinnilla ja semanttisen webin käytännöillä, jotka tarjoavat uudet joukon haasteita:

- Laajan, tekstimuotoisen aineiston semanttinen, ontologiapohjainen käsitteistäminen, joka täytyy voida suorittaa niin nopeasti, että sitä voidaan soveltaa miljoonista artikkeleista koostuviin tietokantoihin
- Käsitepohjainen dokumentinlaajennus ja -haku, eli miten käsitteet sijoitetaan dokumenttien yhteyteen ja miten niitä voidaan hyödyntää haussa mahdollisimman käyttäjäystävällisesti
- Semantiikan suomat mahdollisuudet uutisaineistolle, eli miten käsitteistettyä aineistoa voidaan hyödyntää siten, että sekä tiedon tuottajat, että käyttäjät hyötyvät siitä

Ensimmäisessä yllämainituista haasteista merkittävimmät ongelmat muodostaa aineiston laajuuden vaatima käsitteistyksen automaattisuus. Esimerkiksi uutistuottajien arkistot ovat liian laajoja manuaalisille tai edes puoliautomaattisille ratkaisuille yksinkertaisesti siksi, että arkistojen työntekijät ovat täysin työllistettyjä uusien artikkelien arkistoinnista, eikä kaikkien vanhojen artikkelien läpikäymisen vaatimia resursseja ole käytettäväksi.

Käsitepohjaisella dokumentinlaajennuksella tarkoitetaan käytännössä sitä, että indeksoitaviin dokumentteihin lisätään tiedot niissä esiintyvistä käsitteistä ja näiden ympäristöistä muodostaen semanttisia ryhmittelyjä. Tällä pyritään tuottamaan lisäinformaatiota indeksointia varten liittyen dokumentin varsinaiseen aiheeseen ja näin ollen parantaa dokumentin löydettävyyttä sitä haettaessa. Lisäksi pyritään luomaan näitä uusia mahdollisuuksia hyödyntävä hakukoneisto.

Kolmas haaste käsittelee spesifisti semantiikan suomia mahdollisuuksia uutisaineistojen kannalta. Uutisaineistoilla on joukko karakterisoivia piirteitä, kuten aihepiirin laajuus, aineiston päivittäinen kasvu ja uusien henkilöiden, paikkojen ja käsitteiden ilmaantuminen, jotka tarjoavat sekä haasteita, että mahdollisuuksia semanttiselle käsittelylle.

Tutkimus toteutettiin osana TEKES:in FinnONTO-projektia /69/ ja rahoittajana toimi Sanoma Data /70/, joka myös toimitti käytetyn aineiston.

Seuraavassa esitellään ensin asiasanastojen ja ontologioiden tarkoitus ja ominaisuudet, jonka jälkeen tarkastellaan semanttisen tiedon esittämiseen käytettyjä kieliä. Neljännessä luvussa perehdytään informaation indeksointiin ja hakumalleihin painottaen erityisesti kyselyn laajennusta. Viidennessä luvussa esitellään semantiikan suomia etuja ja mahdollisuuksia sanomalehtiarkiston kannalta. Kehitetty menetelmä, dokumentin laajennus käsite-

klusteroinnilla, esitellään luvussa kuusi ja menetelmän pohjalta luotu käytännön toteutus luvussa seitsemän. Luodun järjestelmän testaus kuvataan luvussa kahdeksan viimeisen luvun sisältäessä yhteenvedon.

2 ASIASANASTOT JA ONTOLOGIAT

2.1 Asiasanastot

2.1.1 Yleistä

Asiasanasto tarkoittaa kontrolloitua sanastoa, johon on koottu jonkin aihepiirin suhteen oleellisia termejä /62/. Asiasanastoja käytetään yleisesti kirjastoissa, museoissa, sanomalehtien arkistoissa jne. Tyypillisesti asiasanastot on organisoitu sanojen merkitysten mukaan tesauruksiksi, jolloin sanastossa kuvataan myös termien välisiä suhteita, joka helpottaa indeksointia ja toisiinsa liittyvien termien löytämistä sekä mahdollistaa päättelytekniikoiden ja -ohjelmien hyödyntämistä. /1, s.1/

2.1.2 Asiasanastojen rakenteesta

Tesaurus on keskeinen työkalu laajojen aineistojen luokittelussa ja asiasanoittamisessa, mutta käsiteltävän aineiston laajentuessa ja yleistyessä siirtyy tesaurusten käyttö puhtaasti tiedonkäsittelyn ammattilaisten harteilta myös maallikoiden ja tietokoneohjelmien käyttöalueelle. Tieto- ja tietämystekniikan näkökulmasta katsottuna asiasanastot ovatkin keskeisessä asemassa tiedon haussa, päättelyssä ja esittämisessä. /1, s.1-2/

Asiasanastoissa termien välisiä suhteita määritellään tyypillisesti kolmella eri suhderyhmällä /1, s.4-5/

- Ekvivalenssisuhteilla kuvataan synonyymit ja korvaavat- sekä vanhentuneet termit. Näin voidaan siis määrittää suositeltavat termit ja mahdollisesti myös erotella leksikaaliset variantit ja kvasisynonyymit aidosti eriävistä synonyymitermeistä.
- Hierarkkiset suhteet kuvataan laajempi termi- ja suppeampi termi-suhteilla, joilla voidaan luoda ala- ja yläkäsitteistä muodostuva, yksinkertainen luokkarakenne, jota voidaan täydentää koostumussuhteilla, sekä yksilö- ja luokkasuhteilla.
- Assosiatiivisilla suhteilla kuvataan muut suhteet rinnakkaistermimääreillä.

2.2 Ontologiat

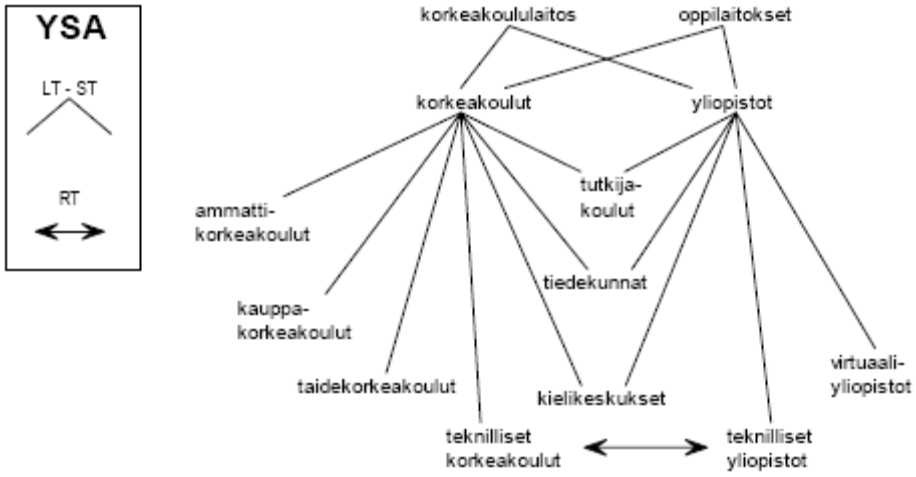
2.2.1 Yleistä

Ontologia /5/ on käytännössä kehittyneempi versio asiasanastosta, joka on erityisesti suunniteltu koneymmärrettävään muotoon. Ihmiselle riittävä termien esitystarkkuus ei vastaa koneen vaatimuksiin, koska ohjelmallisesti hyödynnettävissä olevassa tiedossa pitää kaikkien päättelyssä käytettävien suhteiden olla eksaktisti ja eksplisiittisesti määritellyt. Ontologisointi on siis periaatteessa asiasanaston suhdeverkoston tarkentamista ja täydentämistä. /1, s.2/

Formaali määritelmä ontologiasta on käsitteistämisen eksplisiittinen määrittäminen. Itse termi ontologia on lainattu filosofiasta, jossa se merkitsee olevan systemaattista määrittämistä ja tietämystekniikoihin sovellettaessa oleva siis tarkoittaa tietyn sanaston määrittämää konseptiavaruutta. /5/

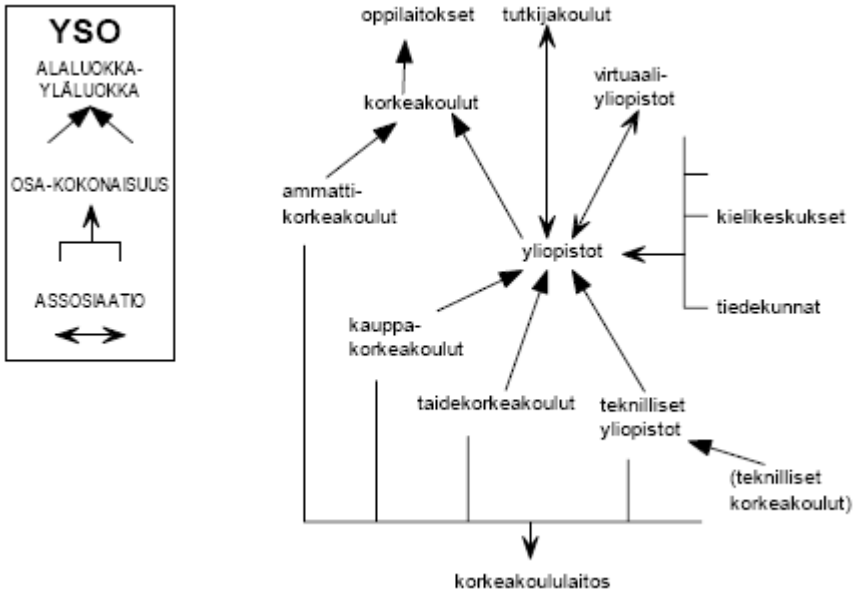
Tässä työssä sanalla termi viitataan jossakin dokumentissa esiintyvään, yksittäiseen termiin, ja sanalla käsite ymmärretään ontologista käsitettä, johon termi viittaa. Käsitteellä on paikkansa ontologisessa hierarkiassa, sillä on suositeltava merkkijonoesitys eli nimi (engl. label), mahdollisia vaihtoehtoisia nimiä sekä muita ontologisia suhteita toisiin hierarkian käsitteisiin.

Esimerkki ontologisoinnin, eli asiasanaston ontologiaksi muuttamisen, hyödyistä on Yleisen suomalaisen asiasanaston /72/ (YSA) ontologisoinnissa Yleiseksi suomalaiseksi ontologiaksi /71/ (YSO, ks. 2.2.6). Kuvassa 2 on esitetty osa YSA:n hierarkiaa, joka käsittelee korkeakouluihin liittyvää tietoa. /23/



Kuva 2. *Ote Yleisen suomalaisen asiasanaston hierarkiasta /23/*

YSA:ssa on laajempi termi-suppeampi termi -suhteita (LT-ST) ja rinnakkaistermisuhteita (RT). Näin ollen automaattinen päättelykone ei kykene esimerkiksi päättelemään kuvan 2 asiasanastosta, että yliopistot ovat korkeakouluja, koska yliopistojen laajempana terminä eivät ole korkeakoulut vaan astetta ylempänä hierarkiassa sijaitsevat oppilaitokset. Ammattikorkeakouluilla, tiedekunnilla ja kielikeskuksilla on kaikilla korkeakouluihin samanlainen suhde, jonka perusteella voidaan esimerkiksi kielikeskukset virheellisesti mieltää eräänlaisiksi korkeakouluiksi tai yliopistoiksi. Myös virtuaaliyliopistot ymmärrettäisiin koneellisessa päättelyssä jonkinlaisiksi yliopistoiksi, vaikka ne eivät ole erillisiä oppilaitoksia. /23/



Kuva 3. *Ote Yleisen suomalaisen ontologian hierarkiasta /23/*

Kuvassa 3 on esitetty osa YSO:n hierarkiasta, joka vastaa kuvan 2 YSA:n hierarkiaa. Tarkistamalla ja lisäämällä suhteita yhdellä sekä selkeästi määrittelemällä nämä suhteet luokkahierarkiaksi pelkkien LT-ST-termisuhteiden sijaan luodaan rakenne, jossa automaattinen päättely ei tuota enää vastaavia virheitä kuin asiasanastojen kohdalla. /23/

Asiasanastoihin verrattuna ontologia on yksi yhtenäinen kokonaisuus ja kaikkien käsitteiden keskinäiset suhteet on eksplisiittisesti määritelty. Kaikille käsitteille annetaan myös yksilöllinen URI ja näin erotetaan termit ja käsitteet, eli termien todelliset, yksiselitteiset merkitykset, toisistaan. Yksi termi voi siis viitata useampaan ontologiseen käsitteeseen eri puolilla hierarkiaa, toisin kuin asiasanastoissa, joissa määritelmällisesti termit ovat yksilöiviä. /23/

Ontologisointi lisää merkittävästi sanaston kehittämiseen vaadittavaa työmäärää, joten on tärkeää arvioida saavutettavat hyödyt, joita ontologisoinnin mahdollistama tiedonkäsittelyn automaatio kykenee tarjoamaan. Lisäksi on päätettävä tehdyn operaation laajuus eli muodostettavan hierarkkisen rakenteen syvyys ja sen hienojakoisuus. /1, s.2/

2.2.2 Ontologioiden peilaus ja yläontologiat

Ontologiat ovat aina sidottuja kohdealueeseensa ja esittävät sen ontologiaa hyödyntävien sovellusten vaatimalla tarkkuudella. Semanttisen webin keskeisenä ajatuksena on semanttisten verkkojen muodostaminen eri kohdealueiden metatietoja ja tietomalleja yhdistämällä mahdollistaen automaattisen päättelyn ja tietojen hyödyntämisen. /11, s.2/

Ontologioiden suora yhdistäminen ei kuitenkaan välttämättä ole täysin suoraviivaista. Esimerkiksi hierarkioiden yhteensovittaminen voi olla monimutkaista, jos samaa käsitettä on lähestytty eri näkökulmista ja eri tarkkuudella. Eräs mahdollinen ratkaisu eri ontologioiden yhdistämisen ongelmien minimoimiseksi on niin sanottujen yläontologioiden käyttö. Nämä

tarjoavat yhteisen yläkäsitteistön, jota lokaalit ontologiat voivat tarkentaa ja joka liittää termit yhtenäiseksi kokonaisuudeksi. /11, s.2/

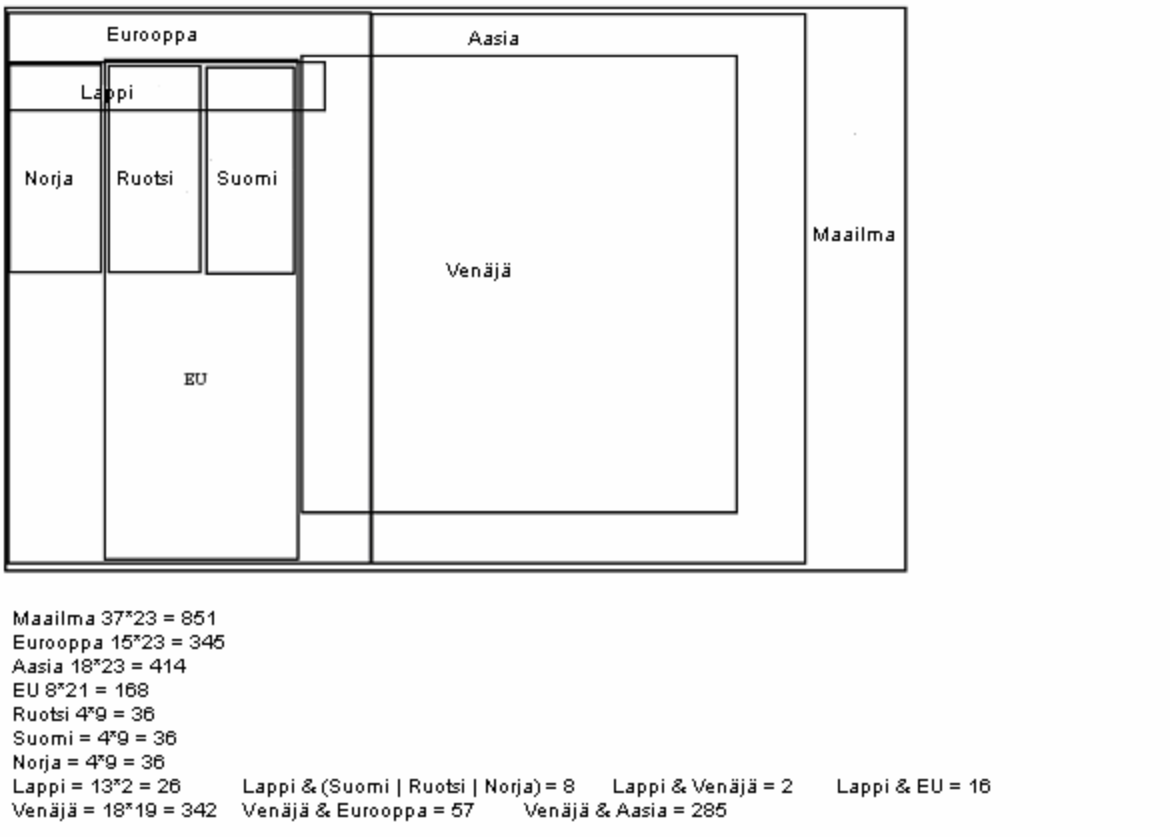
Esimerkiksi jossain yläontologiassa voi olla käsite autosta, joka sitten voidaan tarkentaa jossain teknisemmässä ontologiassa jakamalla se sähköautoihin ja polttomoottoriautoihin. Keskeinen käsite ontologioiden välisessä kanssakäymisessäkin on ontologioiden peilaus (engl. mapping) keskenään, eli kahdessa ontologiassa esiintyvien, samojen käsitteiden välille on voitava vetää yhteys. /55/

Jos ontologia ajatellaan matemaattisesti muodossa $O = (S,A)$, missä S sisältää ontologian piirteet, jotka kuvaavat ontologian sanaston, ja A sisältää ontologian aksioomat, eli sanaston tulkinnan jossain tietyssä kontekstissa, on täydellinen ontologinen peilaus ontologiasta $O_1 = (S_1,A_1)$ ontologiaan $O_2 = (S_2,A_2)$ ontologisten piirteiden muunnos $f : S_1 \rightarrow S_2$ siten, että $A_2 = f(A_1)$. Toisin sanottuna kaikki tulkinnat, jotka toteuttavat O_2:n aksioomat toteuttavat myös O_1:n käännetyt aksioomat. Osittainen peilaus on olemassa ontologiasta $O_1 = (S_1,A_1)$ ontologiaan $O_2 = (S_2,A_2)$, jos on olemassa aliontologia $O_1' = (S_1',A_1')$ (S_1' kuuluu joukkoon S_1 ja A_1' kuuluu joukkoon A_1), jolle on olemassa täydellinen peilaus O_1':stä O_2:een. /55/

2.2.3 Sumean tiedon esittäminen ontologioissa

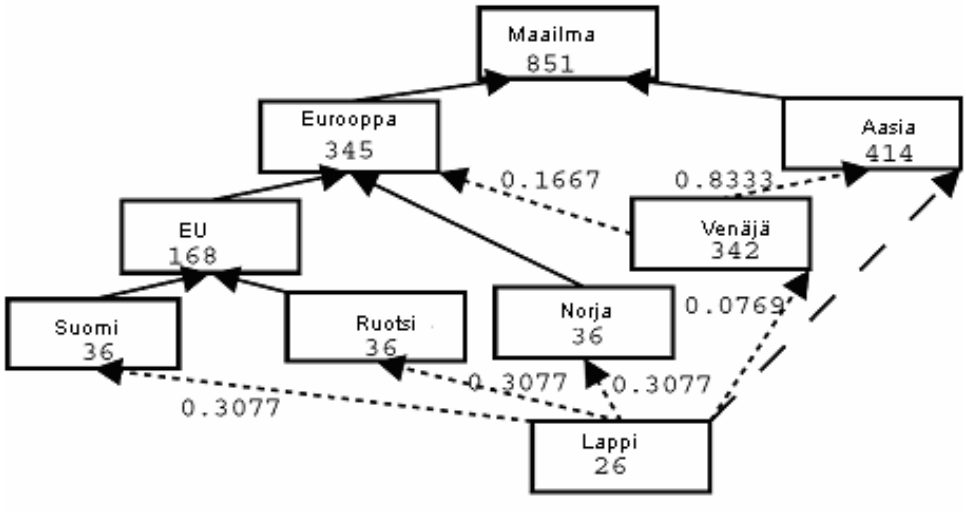
Sumealla (engl. fuzzy) tiedolla ymmärretään tässä tietoa, johon liittyy jonkinlainen painokerroin. Annotoinnissa tämä tarkoittaa sitä, että jokin asia annotoidaan useammalla kuin yhdellä ontologisella käsitteellä ja näille käsitteille annetaan painot, jotka tyypillisesti tuottavat yhteenlaskettuna summaksi 1. Ontologiassa puolestaan sumeus tarkoittaa sitä, että ontologisille suhteille annetaan painot, jotka edustavat joko osasuhteita tai todennäköisyyksiä. /63, s. 10-11, 36-37/

Oletetaan määritellyksi osasuhde partOf, joka kuvaa sitä, että tietyn käsitteen edustaja muodostaan osan jonkin toisen käsitteen edustajasta. Tällainen suhde on varsin tyypillinen, mutta binäärisessä muodossa se ei aina riitä mallintamaan todellisen maailman tilanteita oikein. Suora, binäärinen partOf-suhteisto ei kykene esimerkiksi ilmaisemaan sitä, että Lappi on osin Norjan, Ruotsin, Suomen ja Venäjän alueella, eikä millään tavoin kuvaamaan päällekkäisten alueiden kokoa. Kuvassa 4 on esitetty tämä tilanne Venn-diagrammilla. /39/



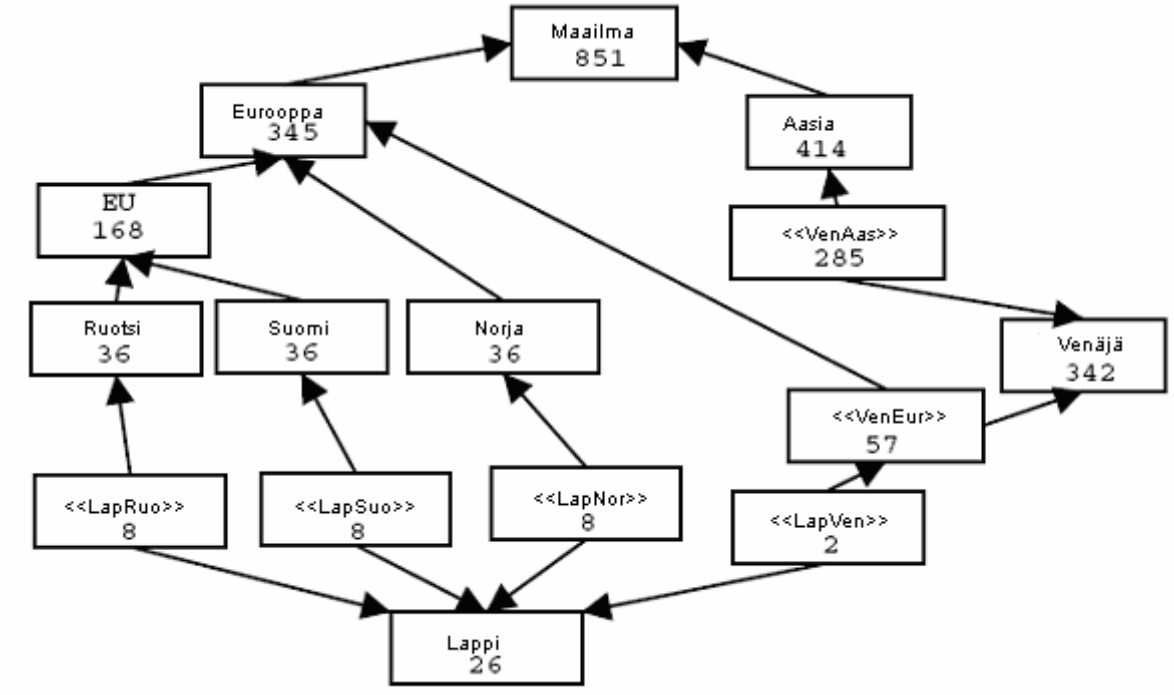
Kuva 4. *Venn-diagrammi, jossa joukko maita ja alueita sekä niiden päällekkäisyys ja koko /39/*

Jotta ontologioissa voidaan määrittää osittaisia aliluokka- ja osasuhteita, voidaan näihin lisätä sumeusarvo. Toisin sanottuna esimerkiksi partOf-suhteen tapauksessa jos luokka A sisältyy kokonaan luokkaan B (Suomi on kokonaan EU:n osa), annetaan aliluokkasuhteelle sumeusarvoksi yksi. Jos taas suhde on alle yksi (Lappi sisältyy Suomeen vain osittain), annetaan suhteelle arvo välillä nollasta yhteen siten, että luokan kaikkien aliluokkasuhteiden yhteenlaskettu sumeusarvo ei ylitä yhtä. Kuvassa 5 on esitetty taksonomia, jossa katkoviivat kuvaavat osittaisia osasuhteita ja pitkä katkoviiva edustaa erillisyyttä. Numerot katkoviivojen yhteydessä ovat suhteeseen liittyviä sumeusarvoja. /39/



Kuva 5. *Kuvan 4 Venn-diagrammia vastaava taksonomia /39/*

Jos kaarien itsessään ei haluta sisältävän sumeusarvoja, voidaan sumeiden kaarien sisälle luoda välikäsitteet, jotka kuvaavat kahden resurssin päällekkäisen osan. Jos luokasta A on olemassa sumeusarvon sisältävä kaari luokkaan B, luodaan A:n B:n väliin luokka C, joka sisältää A:sta sumeusarvolla kerrotun osan. Luokasta C vedetään tämän jälkeen ei-sumeat kaaret luokkiin A ja B. Tämä on esitetty kuvassa 6. /39/



Kuva 6. *Kuvan 5 taksonomia muutettuna kiinteän polun järjestelmäksi (engl. solid path structure) /39/*

Päällekkäisyyksien laskemiseksi taksonomian käsitteelle A tarvitaan tieto siitä, kuinka paljon B:llä on yhteistä A:n kanssa. Tämä voidaan laskea tilastollisia menetelmiä hyödyntäen yksinkertaisesti todennäköisyytenä P(B|A). Kun tämä lasketaan taksonomian kaikille käsitteille, saadaan tulokseksi kaikkien käsitteiden päällekkäisyydet. Teoriassa tämä ehdollinen

todennäköisyys voidaan laskea suoraan Venn-diagrammista, mutta käytännössä tämä on erittäin hankalaa. Toinen tapa laskea päällekkäisyys on muuttaa RDF(S)-graafi kuvan 6 mukaiseksi Bayes-verkoksi (engl. Bayesian Network), jossa käsitteiden päällekkäisyys lasketaan seuraavalla kaavalla: /39/

$$o = \frac{n(A \cap B)}{n(B)} \qquad (1)$$

Kaavassa 1 n(.) on käsitteen massa, joka esimerkiksi kuvan 6 tapauksessa viittaa maa-alueiden kokoon. Edellä mainittujen välikonseptien sumeusarvo on siis yksinkertaisesti leikkaus niiden konseptien massoista, joiden välillä se sijaitsee. /39/

2.2.4 Ontologiakielet

Ontologiakieleltä vaaditaan luonnollisesti kykyä toteuttaa ontologioiden hyvän esitettävyyden asettamat vaatimukset, jotka pääasiassa ovat seuraavat /3, s.110/ :

- Hyvin määritelty rakenne (syntaksi)
- Tehokas päättelytuki
- Formaali semantiikka
- Riittävä ilmaisuvoima
- Esityksen kätevyys

Hyvin määritellyn kieliasun edut ovat selvät: se vaaditaan, jotta kieli voisi olla konekäsiteltävää. Tehokas päättelytuki mahdollistaa automaattisen tiedonprosessoinnin, joka tehostaa suurten ontologioiden käsittelyä merkittävästi. Formaali semantiikka määrittelee tiedon merkitykset tarkasti ja on edellytys loogiselle päättelylle. Riittävän esityskyvyn kulloinenkin määritelmä riippuu luonnollisesti kielen käyttötarkoituksesta, mutta tyypillisesti vaaditaan ainakin asiasanastoihin verrattuna useamman erilaisen suhteen määrittelemistä. OWL (ks. 3.2) täyttää nämä vaatimukset vähintään kohtuullisesti. /3, s.110/

2.2.5 Asiasanaston muuttaminen ontologiaksi

Eräs mahdollinen lähestymistapa asiasanaston ontologisointiin on suorittaa se käsityönä, jolloin eräs mahdollisuus on jakaa prosessi karkeasti neljään vaiheeseen: syntaktiseen muunnokseen, semanttisten ekvivalenssisuhteiden erottamiseen ja määrittämiseen, hierarkkisten suhteiden täydentämiseen ja korjaamiseen sekä assosiatiivisten suhteiden tarkempaan määrittelyyn. Syntaktinen muunnos tarkoittaa yksinkertaisesti asiasanaston kielellistä muuntamista johonkin ontologiaformaattiin kuten OWL. /1, s.5/. Yleensä tähän käytetään jotain valmista ontologiaeditoria, kuten Stanfordin yliopiston kehittämää Protege-2000-ohjelmaa /12/.

Muut muutokseen vaadittava toimenpiteet liittyvät asiasanastojen ja ontologioiden käsitteellisiin eroihin. Ontologioissa käytettävän hierarkkisen järjestelmän on oltava yksiselitteinen, koska ontologioita hyödyntävät päättelykoneet vaativat toimiakseen yksiselitteiset suhteet käsitteiden välille. Useissa asiasanastoissa ei esimerkiksi tehdä eroa suppeamman käsitteen ja osasuhteessa olevan käsitteen välille. /56/ Toisin sanottuna käsitteet ”tuolin

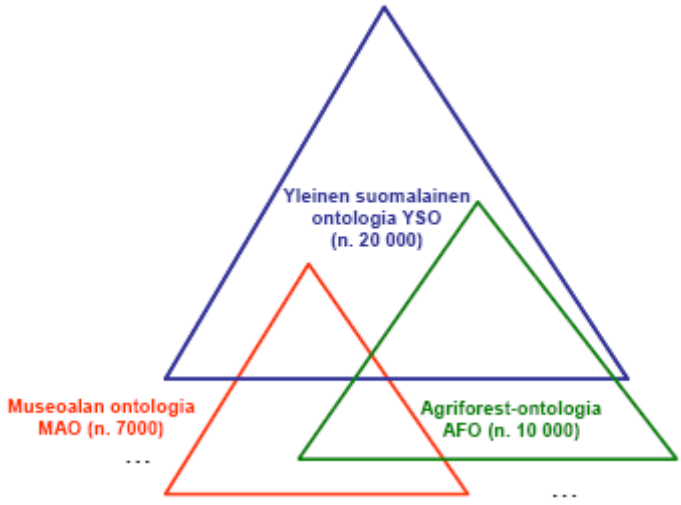
jalka” ja ”valtaistuin” saattavat molemmat olla saman alisteisen suhteen päässä tuolista, vaikka käsitteiden välisen suhteen merkitys on selvästi erilainen.

Toinen keskeinen tehtävä on hierarkian täydentäminen. Asiasanastot eivät välttämättä muodosta täydellistä rakennetta ja myös alisteisten suhteiden epämääräisyydestä johtuen saatetaan tarvita ryhmitteleviä käsitteitä sitomaan hierarkia yhdeksi kokonaisuudeksi. Tällöin joudutaan usein myös muodostamaan useita käsitteitä sanaston monimerkityksisistä termeistä. /56/ Esimerkiksi suomen sana ”kilpi” voi merkitä iskuilta suojaavaa puolustusvälinettä tai liikenteessä käytettävää merkkiä. Käsitteinä nämä eroavat toisistaan merkittävästi, jolloin termi joudutaan jakamaan kahdeksi, eri puolille hierarkiaa sijoittuviksi käsitteiksi.

Ontologisointi voidaan myös suorittaa puoliautomaattisesti, jossa asiasanaston hierarkia laajennetaan lisäämällä uusia suhteita olemassa olevien käsitteiden välille, mikä voidaan tehdä jotain toista hierarkiaa hyödyntäen, tai vaihtoehtoisesti louhimalla jostain dokumenttimassasta täydentäviä suhteita. Tämä menetelmä kuitenkin nojaa voimakkaasti olemassa olevien resurssien hyödyntämiseen ontologisoinnissa. /64/

2.2.6 Yleinen suomalainen ontologia YSO

Yleinen suomalainen ontologia eli YSO, on kansallisen Suomalaiset semanttisen webin ontologiat -hankkeen (FinnONTO, 2003-2007) puitteissa rakennettu yleisen tason ontologia, joka tarjoaa kansallisen yläkäsitteiden ontologian (top ontology). YSO toimii yhdistävänä tekijänä eri suomalaisille ontologioille, jotka täydentävät sitä sovelluskohtaisen tarpeen mukaan. Kuvassa 7 on esitetty esimerkki tästä yhdistävyydestä graafisessa muodossa, eli YSO:n suhde kahteen muuhun suomalaiseen ontologiaan. /23/



Kuva 7. *Yleisen suomalaisen ontologia toimii erilaisten alakohtaisten ontologioiden yhdistäjänä ja yläontologiana /23/*

YSO on OWL-kielellä esitetty, ontologisoitu versio Kansalliskirjaston laajassa käytössä olevasta Yleisestä suomalaisesta asiasanastosta YSA:sta. Ontologisointia varten YSA:an lisättiin joukko kokoavia yläkäsitteitä, jotta hierarkiasta saatiin yhtenäinen. Esimerkkejä tällaisista yläkäsitteistä on käsitteiden jako ylimmällä tasolla abstrakteihin, muuttuviin ja

pysyviin käsitteisiin, joista esimerkiksi viimeksi mainittu jakaantuu mm. fyysisiin objekteihin, ilmiöihin ja ajanjaksoihin. Tällaisia yläkäsitteitä ei tyypillisesti käytetä suoraan annotoinnissa, vaan ne vain täydentävät hierarkiaa ja mahdollistavat päättelykoneiden toiminnan. /23/

YSO sisältää myös erillisessä rakenteessa YSA:n koko sanaston, joka on muutettu SKOS-muotoon (Simple Knowledge Organization System /58/). Tämä mahdollistaa sen, että YSO:n käsitteistä löytyy suorat viittaukset alkuperäisiin YSA:n termeihin, joka helpottaa esimerkiksi YSA:lla annotoidun aineiston käyttämistä YSO:a hyödyntävien järjestelmien rinnalla. Lisäksi SKOS on standardi esitysmuoto, jota on helpompi prosessoida eteenpäin eri sovelluksissa ja siihen muunnettuna YSA:n termeillekin määritellään yksilöivät URI:t. /23/

Kun termit muutetaan käsitteiksi pitää monimerkityksisten termien ollessa kyseessä muodostaa useampia käsitteitä, jotka sijoittuvat eri puolille hierarkiaa. Esimerkiksi YSA:n termi ”kaakao” on jaettu kahteen käsitteeseen termin eri merkitysten mukaan. YSO:n käsitteen `kaakao_1` yläkäsitteen on `juomat`, kun taas käsitteen `kaakao_2` yläkäsitteenä on `nautintoaineet` ja assosiaatiivisena suhteena `suklaa`. Jos erottelua kahden samaan termiin liittyvän käsitteen välillä ei haluta tai voida tehdä, voidaan käyttää koostekäsitteitä, jotka ovat yksinkertaisesti unioni termin eri merkityksistä. Kaakaon tapauksessa saatetaan esimerkiksi ruokaa käsittelevissä dokumenteissa haluta tehdä tarkempi erottelu termin eri merkityksien välille, mutta yleisemmässä aineistossa se ei välttämättä ole tarpeen tai edes mielekästä.

3 SEMANTTISEN TIEDON ESITYS

3.1 RDF

3.1.1 Yleistä

RDF (Resource Description Framework) on World Wide Webin sisältöjen (resurssien) kuvaamiseen tarkoitettu kieli, jota kehittää World Wide Web Consortiumin (W3C) /2/. W3C on nykyisen World Wide Webin kehittäjän, Tim Berners-Leen perustama konsortio, joka pyrkii webin kehittämiseen ja käytettyjen teknologioiden standardointiin /4/. RDF on edennyt Recommendation-luokitukseen ja sitä käytetään laajalti. /2/

XML (Extended Markup Language) puolestaan on universaali metakieli merkintäkielien määrittelemiseksi ja se mahdollistaa sovellusten välisen kommunikoinnin ja metadatan vaihtamisen. XML ei kuitenkaan sisällä tukea informaation semantiikan (eli merkityksen) käsittelemiseksi. Esimerkiksi elementtien sisäkkäisyyden merkitystä ei ole yksiselitteisesti määrätty, vaan jokainen merkintäkielen kirjoittaja päättää sen itse. RDF:ssä on pyritty määrittämään yleispätevä semantiikka, joka tukee moninaisia tietosisältöjä. /3, s.61-62/

3.1.2 RDF:n tarkoitus – miksi RDF?

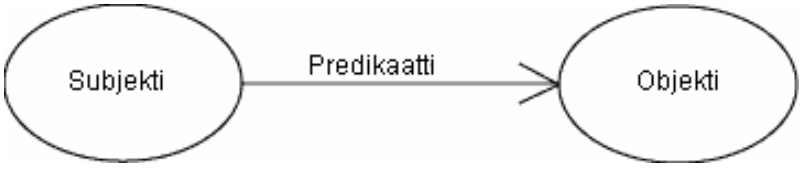
RDF on tarkoitettu koneymmärrettäväksi ja alustariippumattomaksi tiedonesitysstandardiksi, jota eri sovellukset voivat käyttää ja välittää edelleen merkitysten säilyessä. W3C määrittelee RDF:n motivaatioksi seuraavaa: /6/

- Formaatti, jolla voidaan esittää webin sisäistä metadataa web-resursseista ja niitä käyttävistä systeemeistä
- Avoin tietomalli suljetun sijaan
- Mahdollistaa koneymmärrettävän informaation käsittely ja hyödyntäminen sen luoneen ympäristön ulkopuolella
- Mahdollisuus yhdistää tietoa useista sovelluksista ja näin tuottaa uutta informaatiota
- Mahdollistaa web-agentti-pohjainen, automaattinen tiedonkäsittely

RDF perustuu yksinkertaiseen tietorakenteeseen, jota on helppo käsitellä ja tulkita ohjelmallisesti, mutta samalla se määrittelee formaalin semantiikan ja laajennettavissa olevan, URI-pohjaisen sanaston. Sille on olemassa XML-perustainen esitysmuotosuositus, mutta sinällään RDF on esitystavasta riippumaton. Kaikki nämä seikat yhdessä mahdollistavat Internetin skaalassa toimivan tietorakenteen, jossa tieto on laajalle hajautettu. Yleisesti ottaen RDF:ssä ei oleteta, että kaikki tieto mistä tahansa RDF-resurssista olisi saatavilla. Mahdollisuus osittaisen informaation hyödyntämiseen on ensiarvoisen tärkeä. /6/

3.1.3 RDF:n rakenne

RDF perustuu väittämiin (engl. statement), jotka antavat arvon jonkin tietyn resurssin tietylle ominaisuudelle. Väittämä on rakenteeltaan kolmikko (engl. triplet), joka muodostuu subjektista, predikaatista ja objektista. Subjekti määrää väittämän käsittelemän resurssin, predikaatti kuvattavan ominaisuuden tai suhteen ja objekti tämän arvon, joka voi olla toinen resurssi tai jokin kiinteä arvo (esim. string-muotoinen kirjainjono). Predikaattia kutsutaan joskus myös ominaisuudeksi (engl. property). Kuvassa 8 on esitetty tämä tietomalli graafisesti suunnattuna verkkona, jossa subjekti ja objekti muodostavat solmut ja predikaatti toimii suunnattuna kaarena (suuntaus aina subjektista objektiin). /6/



Kuva 8. *RDF:n rakenne verkkoesityksenä /6/*

Resurssit identifioidaan RDF:ssä web-tunnisteilla eli Uniform Resource Identifiereillä (URI), jotka muodostuvat XML-syntaksin mukaisesta nimiavaruusmäärittelystä (engl. namespace) ja paikallisesta nimestä (engl. local name). Nämä yhdessä määrittävät resurssille yksilöllisen tunnisteen, jonka uniikkius voidaan taata. /2/

Solmun URI kertoo mikä resurssi on kyseessä ja predikaatin URI määrittää kaaren merkityksen. Tämä predikaatin URI voi puolestaan toimia solmuna jollekin toiselle kolmikolle,

jos halutaan määritellä jotain itse ominaisuudesta. Tämä mahdollistaa resurssien välisten suhteiden tarkemman määrittelemisen ja rajaamisen. /6/

3.1.4 RDFS

RDFS, eli RDF Schema, on skeemakieli, jolla RDF:n terminologia määritellään. Nimi saattaa olla tavallaan harhaanjohtava, koska RDFS ei määrittele RDF-dokumenttien rakennetta, niin kuin XML Schema määrittelee XML-dokumenttien rakenteen, vaan sillä määritetään datamalleissa käytetty sanasto. /3, s.62/ RDFS:n rakenteet ovat validia RDF:ää, mutta niiden merkitykset poikkeavat osittain tavallisesta. RDFS on W3C:n suositusasteelle (engl. Recommendation) edennyt standardi. /2/

Käytännössä RDF mahdollistaa resurssien kuvailun yksinkertaisin väittämin, kun taas RDFS mahdollistaa kuvauksiin käytetyn sanaston määrittelyn. Kuvailtavat asiat jaetaan luokkiin, joiden edustajille voidaan asettaa määriteltäviä ominaisuuksia. Luokkiin kuuluvat asiat ovat ilmentymiä (engl. instance) luokastaan ja yksi resurssi voi olla samanaikaisesti useamman luokan instanssi. Luokat voivat myös muodostaa luokkahierarkian käyttäen aliluokkasuhteita, joissa luokkien ominaisuudet periytyvät. /2/

RDFS määrittelee joukon primitiivejä, joilla luokkien kuvailu tapahtuu. Näihin primitiiveihin kuuluvat mm. seuraavat: /38/

- rdfs:Class, joka on kaikkien luokkamuotoisten resurssien luokka. Itsessään rdfs:Class on rdfs:Classin instanssi.
- rdfs:subClassOf sallii aliluokkasuhteiden määrittämisen. Vastaavasti rdfs:subPropertyOf mahdollistaa aliominaisuuksien määrittelemisen.
- rdfs:Resource on RDFS:n ylin luokka. Kaikki RDF:llä kuvaillut asiat ovat rdfs:Resource-luokan instansseja ja kaikki luokat ovat sen aliluokkia.
- rdfs:Domain- ja rdfs:Range-ominaisuuksien avulla voidaan asettaa rajoituksia muille ominaisuuksille. rdfs:Domain määrittelee ne luokat, joiden instanssit voivat toimia ominaisuuden subjekteina ja rdfs:Range määrittää rajoituksia ominaisuuden arvoihin.
- rdfs:Literal on luokka literaaliarvoisille resursseille (esim. merkkijonot tai kokonaisluvut) ja tukee kielimäärittelyjä.
- rdfs:label on luokka ihmisluettavan nimen lisäämisen jollekin määriteltävälle resurssille ja rdfs:comment sallii huomioiden kirjoittamisen.

3.1.5 RDF:n ja RDFS:n rajoitteet

RDF ja RDFS mahdollistavat tietynasteisen ontologisen struktuurin kuvaamisen. Sanastohierarkian organisointi aliluokkien ja aliominaisuuksien (engl. subproperty) tasolla, sekä mahdollisuus rajoituksien kuvaamiseen arvoalueiden tasolla jättävät kuitenkin monia ontologisen ilmaisun kannalta oleellisia tarpeita täyttämättä. Tärkeimmät puutteet ovat seuraavat: /3, s.111/

- Ominaisuuksien vaikutusalueen laajuutta ei voi määrätä lokaalisti. rdfs:range määrittää ominaisuuden arvoalueet kaikille luokille, mikä tarkoittaa, että ominaisuudelle ei voida asettaa rajoituksia, jotka olisivat voimassa vain tietyille luokil-

le. Tällöin tarvitaan saman ominaisuuden esittämiseksi kaksi eri esitystä, jos rajoitteita halutaan asettaa lokaalisti. Esimerkiksi jos halutaan rajoittaa `syö`-ominaisuus kasvinsyöjissä kasveihin ja lihansyöjissä muihin eläimiin, tarvitaan kaksi eri ominaisuutta: `lihansyöjä_syö` ja `kasvinsyöjä_syö`.

- Luokille ei voida määritellä yhteispisteettömyyssuhdetta (engl. disjoint). Esimerkiksi mies on yhteispisteetön luokalle nainen, mutta RDFS:ssä onnistuu vain aliluokkasuhteiden määrittäminen.
- Luokkia ei voida yhdistää käyttäen Boolen loogisia operaatioita. Joskus uuden luokan luominen on helpointa yhdistäen muista luokista unionia, leikkausta tai komplementtia käyttäen, mikä ei kuitenkaan ole mahdollista RDFS:ssä. Esimerkiksi `henkilö`-luokka olisi eleganttia ilmaista `mies`- ja `nainen`-luokkien unionina.
- Kardinaalisuusehtoja ei voida määrittää, eli ei voida asettaa instanssien lukumäärää, jotka tietyllä luokalla pitää olla jossain suhteessa itseensä. Esimerkiksi ei voida asettaa ehtoa, että jokaiseen `henkilö`-luokan instanssiin liittyy `vanhempi`-ominaisuudella kaksi `henkilö`-luokan instanssia.
- Erilaisten erikoisominaisuuksien, kuten transitiivisuus (jos A:lla on transitiivinen ominaisuus kohti B:tä ja C:llä kohti A:ta, niin C:llä on tällöin kyseinen ominaisuus myös kohti B:tä), uniikkius (tietty ominaisuus voi esiintyä vain kerran yhden resurssin yhteydessä) tai inversio (tietty ominaisuus yhteen suuntaan määrä toisen ominaisuuden olemassaolon vastakkaiseen suuntaan), määritteleminen ominaisuuksille ei ole mahdollista RDFS:ssä.

3.2 OWL

3.2.1 Yleistä

OWL eli Web Ontology Language on W3C:n standardi koneymmärrettävien ontologioiden kehittämiseen. Se pohjaa RDF:ään, mutta sisältää enemmän mahdollisuuksia semanttisten suhteiden ja rajoitteiden esittämiseen. /7/

OWL perustuu varhaisempaan ontologiastandardiin, DAML + OIL:iin (DARPA Agent Markup Language + Ontology Interchange Language), joka oli myös RDF:n pohjalle rakennettu semanttinen merkintäkieli /8/. Joukko tutkijoita Euroopassa ja Pohjois-Amerikassa oli todennut olemassa olevan tarpeen ilmaisuvoimaisemmalle ontologiakielelle ja niinpä kahden mantereen tiedemiesten ehdotukset yhdistettiin DAML + OIL:iksi /3, s.109/. Sen perusajatus oli määrätä tietty, eksplisiittinen merkitys kieltä käyttäville kolmikoille /8/.

3.2.2 OWL:in tarkoitus

OWL on tarkoitettu ontologioiden kuvaamiseen koneymmärrettävässä muodossa. Se ottaa kantaa luvussa 3.1.6 esitettyihin RDFS:n puutteisiin ja täyttää luvussa 2.2.4 esitetyt tarpeet ontologiakielelle. /7/ OWL ei kuitenkaan kaikissa versioissaan ole suora laajennus RDF Schemaan, koska RDFS sisältää joitakin hyvin voimakkaita mallinnusprimitiivejä, kuten

`rdfs:Class` ja `rdf:Property`, jotka suoraan käytettyinä johtaisivat hallitsemattomiin laskennallisiin päättelyominaisuuksiin. /3, s.112/

Käytännössä OWL lisää RDFS:n sanastoa, jolla ominaisuuksia ja luokkia määritellään. Esimerkiksi luokkien väliset suhteet, kuten erillisyys tai samuus voidaan ilmoittaa, samoin kuin Boolen operaatioilla toisista luokista yhdistetyt luokat. Kardinaliteetti, ominaisuuksien symmetrisyys ja ominaisuuksien väliset inversiosuhteet rikastavat OWL:in ilmaisukykyä. /7/

Koska semanttisen webin visioon kuuluu mahdollisuus ontologioiden automaattiseen keräämiseen ja yhdistelyyn useista eri lähteistä, tukee OWL myös hajautettuja ontologioita, eli kaiken tiedon ei tarvitse sijaita samassa tiedostossa. OWL-ontologiat voivat olla rinnasteisia sisältäen toisista ontologioista eksplisiittisesti haettuja (engl. import) sisältöjä. /9/

3.2.3 OWL:in rakenne

OWL:ista on olemassa kolme laajuudeltaan eriävää versiota: /3, s.113/

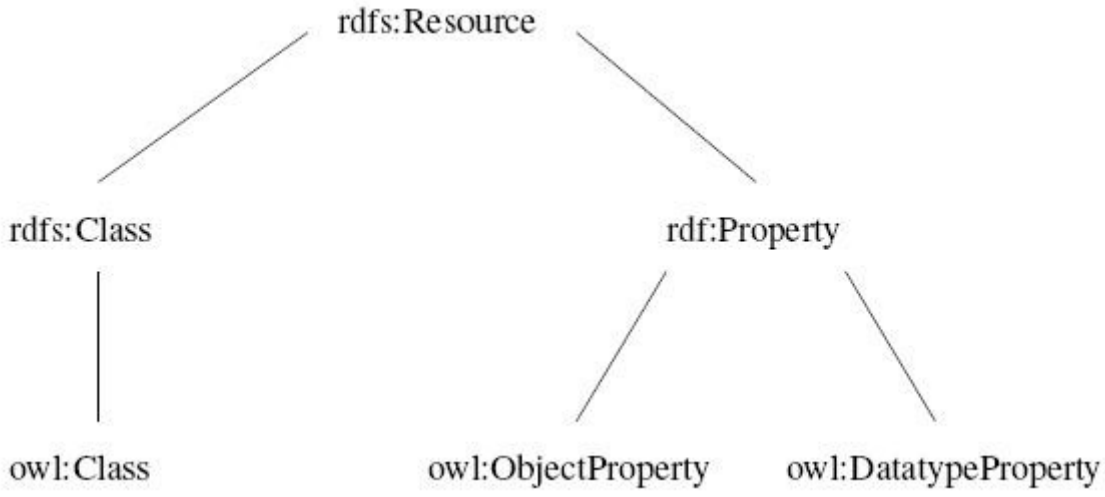
- Koko OWL-kielen sisällään pitävä versio, OWL Full, käyttää kaikkia OWL:in primitiivejä ja sallii näiden vapaan yhdistelyn RDF- ja RDFS-primitiivien kanssa. Tämä mahdollistaa myös jo määriteltyjen RDF- tai RDFS-primitiivien muokkaamisen lisäämällä niihin esimerkiksi kardinaalisuusehtoja. OWL Fullin etuina muihin OWL:in versioihin on sen sisältämien ominaisuuksien suomien mahdollisuuksien lisäksi suora RDFS-laajennus, joka sallii maksimaalisen yhteensopivuuden RDF-dokumenttien ja ohjelmistojen kanssa. Ongelmana kuitenkin on luvussa 3.2.2 esitetyn mukaisesti looginen ratkeamattomuus. /3, s.113/
- OWL DL (OWL Description Logic) on OWL Fullista rajattu alikieli, joka takaa päättelyketjujen laskennallisen toimivuuden (kaikki ketjut lasketaan rajallisessa ajassa tehokkaasti) ja se suunniteltiinkin kuvauslogiikkoja (engl. Description Logic) hyödyntäviä tutkimushaaroja varten /9/. Haittavaikutuksena on, että OWL DL ei ole täysin yhteensopiva RDF:n kanssa. Vaikka jokainen OWL DL-dokumentti onkin validia RDF:ää, pitää useimpia RDF-dokumentteja rajata joiltain osin ja laajentaa toisaalla, jotta niistä saataisiin valideja OWL DL-dokumentteja /3, s.113/.
- OWL Lite on OWL DL:stä edelleen yksinkertaistettu versio, joka on suunnattu sovelluksille, jotka tarvitsevat luokitteluhierarkiaa ja esimerkiksi yksinkertaisia kardinaalisuusehtoja (tuki arvoille 0 ja 1). Siitä puuttuu joukko OWL DL:ssä sallittuja ominaisuuksia (owl:oneOf, owl:unionOf, owl:complementOf, owl:hasValue, owl:disjointWith ja owl:DataRange) ja joillekin jäljelle jääneille ominaisuuksille on asetettu joitakin lisärajoituksia /7/. OWL Liten vahvuudet ovat sen helppous niin käyttäjiä kuin sovelluksiakin ajatellen, mutta luonnollisesti se sallii vain rajoitetun ilmaisun /3, s.114/.

Ontologioita OWL:illa toteutettaessa pitää miettiä ontologiaa hyödyntävien sovellusten tarpeet ja vaatimukset. Valinta OWL Liten ja OWL DL:n välillä riippuu tarpeesta käyttää DL:n laajempaa ilmaisua ja valittaessa OWL DL:n ja OWL Fullin välillä pitää ottaa huo-

mioon Fullin tarjoamat metamallinnusmahdollisuudet RDF Scheman suhteen ja toisaalta DL:n mahdollistama toimiva looginen päättely. /3, s.114/

Eri OWL:in versioiden välillä on tarkka määrittely niiden yhteensopivuudesta keskenään. Kaikki oikeanmuotoiset OWL Lite -ontologiat ovat oikein muodostettuja OWL DL -ontologioita, jotka puolestaan ovat oikeanmuotoisia OWL Full -ontologioita. Samoin logiikan puolella jokainen validi OWL Lite -päätelmä on validi OWL DL-päätelmä, joka puolestaan on validi OWL Full -päätelmä. Yhteensopivuus hierarkiassa alaspäin ei ole taattu. OWL Full voidaan ajatella RDF:n laajennuksena, kun taas OWL DL ja OWL Lite ovat laajennuksia rajoitetusta RDF:stä. /7/

OWL käyttää kuitenkin RDF:ää ja RDF Schemaa suurilta osin hyödykseen ja kaikki OWL:in version käyttävät RDF:n syntaksia. Kuvassa 9 on esitetty joidenkin mallinnusprimitiivien väliset aliluokkahierarkiat OWL:in ja RDFS:n välillä. /3, s.114/



Kuva 9. *Joidenkin mallinnusprimitiivien väliset aliluokkahierarkiat OWL:issa ja RDFS:ssä /10/*

Kuvasta 9 nähdään kuinka OWL:in mallinnusprimitiivit periytyvät suoraan RDF:n ja RDFS:n vastaavista. Teoriassa tämä mahdollistaisi yhteensopivuuden alaspäin siten, että OWL:ia hyödyntävät järjestelmät osaisivat automaattisesti käsitellä myös RDFS-dokumentteja, mutta käytännössä tämä toteutuu vain OWL Fullissa. /3, s.115/

OWL sisältää myös oletuksen avoimesta maailmasta (engl. open world assumption), jossa ei tehdä oletuksia tuntemattomasta. Suljetun maailman oletuksessa tuntemattomien asioiden oletetaan olevan loogisessa mielessä epätosia, mutta OWL:issa näin ei tehdä. /9/

3.2.4 OWL:in käyttö

Esimerkki OWL-kielellä toteutetusta ontologiasta on Yleinen suomalainen ontologia, eli YSO (ks. 2.2.6), josta seuraavassa katkelma:

```
<rdf:Description rdf:about="http://www.yso.fi/onto/yso#p7059">
    <yso-meta:associativeRelation rdf:resource="http://www.yso.fi/onto/yso#p7061"/>
```

```
    <om:overlappedBy
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.0</om:overlappedBy>
    <yso-meta:partOf rdf:resource="http://www.yso.fi/onto/yso#p788"/>
    <rdfs:subClassOf rdf:resource="http://www.yso.fi/onto/yso#p3733"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Nimitys käytössä
vuodesta 1989 lähtien, aiemmat katso toimiupseerit</rdfs:comment>
    <yso-meta:prefLabel xml:lang="en">warrant officers</yso-meta:prefLabel>
    <yso-meta:prefLabel xml:lang="sv">institutofficerare</yso-meta:prefLabel>
    <rdf:type rdf:resource="http://www.yso.fi/onto/yso-meta/2007-03-02#Concept"/>
    <yso-meta:prefLabel xml:lang="fi">opistoupseerit</yso-meta:prefLabel>
    <yso-meta:associativeRelation rdf:resource="http://www.yso.fi/onto/yso#p789"/>
    <yso-meta:semanticTag
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">yso_opistoupseerit</yso-
meta:semanticTag>
    <yso-meta:associativeRelation rdf:resource="http://www.yso.fi/onto/yso#p7060"/>
    <om:definedConcept rdf:resource="http://www.yso.fi/onto/ysa#Y13015"/>
    <om:overlaps rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.0</om:overlaps>
  </rdf:Description>
```

Kyseessä siis on käsite ”opistoupseerit”, jonka URI on http://www.yso.fi/onto/yso#p7059. Ensimmäinen rivi määrittelee subjektin ja lopuilla riveillä on subjektiin liittyviä predikaatti-objekti-pareja. Toisella rivillä määritellään, että ”opistoupseerit”-käsitteellä on assosiatiivinen suhde toiseen käsitteeseen, jonka URI on http://www.yso.fi/onto/yso#p7061 (kanta-aliupseerit). Kolmannella rivillä määritellään ominaisuus ”overlappedBy”, joka saa arvokseen liukuluvun 1,0.

Lisäksi käsitteelle määritellään mm. osasuhde (yso-meta:partOf), aliluokkasuhde (rdfs:subClassOf) ja tyyppi (rdf:type). Käsitteelle annetaan myös kolme erikielistä suositeltua nimeä (prefLabel), joiden kielimääreet sijaitsevat XML-syntaksin mukaisesti osana kolmikon predikaattia.

Seuraavassa esimerkki eräästä YSO:n ominaisuusmäärittelystä:

```
  <rdf:Description rdf:about="http://www.yso.fi/onto/yso-meta/2007-03-02#partOf">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:domain rdf:resource="http://www.yso.fi/onto/yso-meta/2007-03-02#Concept"/>
    <rdfs:range rdf:resource="http://www.yso.fi/onto/yso-meta/2007-03-02#Concept"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">partOf</rdfs:label>
  </rdf:Description>
```

Tässä määritellään osasuhde-ominaisuus, jonka URI on http://www.yso.fi/onto/yso-meta/2007-03-02#partOf ja jota käytetään ilmaisemaan, että kolmikon subjekti on objektin muodostaman kokonaisuuden osa. Esimerkiksi männällä voisi olla partOf-suhde moottoriin. Koodin toisella rivillä määritetään, että kyseisen ominaisuuden tyyppi on OWL-määrittelyn mukainen ObjectProperty, joka siis käytännössä tarkoittaa, että partOf-suhde on ObjectProperty-luokan instanssi. Kolmannella ja neljännellä rivillä ominaisuudelle annetaan domain- ja range-määreet, joiden mukaan sekä ominaisuuden subjektina, että objek-

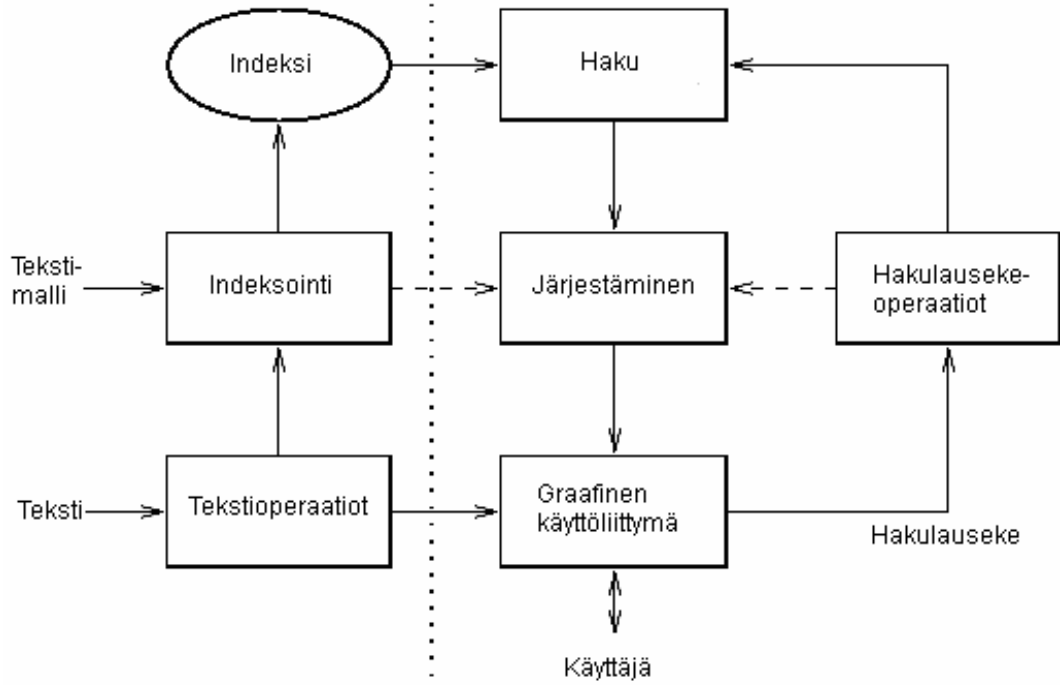
tina pitää olla Concept-luokan instanssi. Viidennellä rivillä on ominaisuuden merkkijonotyyppinen, ihmisluettava nimi.

4 TIEDON INDEKSOINTI JA HAKUMALLIT

4.1 Informaation indeksoinnista ja hakumalleista

4.1.1 Yleistä

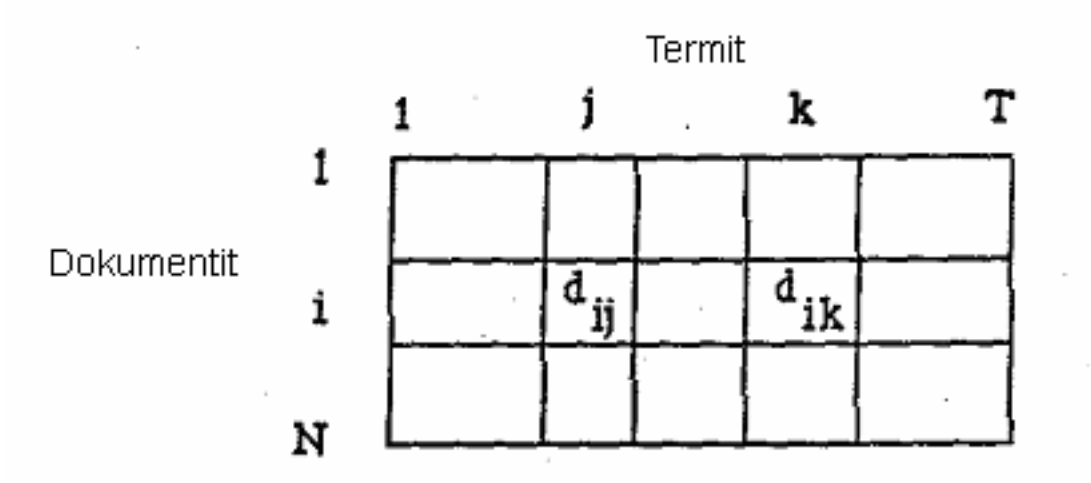
Informaation tallettaminen on turhaa, jos sitä ei voida hakea takaisin esiin. Tiedonhakuprosessilla (engl. information retrieval process) ymmärretään tässä koko prosessia indeksoinnista käyttäjän kyselyn muodostamiseen ja hakutulosten palauttamiseen. Tiedonhakuprosessi erotetaan tiedon selailusta, vaikka siinäkin käyttäjä tietyssä mielessä etsii vastausta johonkin informaatiotarpeeseen.



Kuva 10. *Tiedonhakuprosessi /18/*

Tiedonhaku voidaan mallintaa esimerkiksi kuvan 10 kaltaisesti. Ennen kuin tiedonhakuprosessi voidaan alustaa, pitää päättää itse tekstin esitysmuodosta ja kokoelmaan käytettävistä dokumenteista. Indeksin luonti on kriittinen osa hakuprosessia ja sen pääasiallinen tarkoitus on nopeuttaa järjestelmän vastauksia kyselyihin. Käyttäjä toimii käyttöliittymän kautta ja suorittaa kyselyn, jolle mahdollisesti tehdään joitain kyselyoperaatioita ja joka tämän jälkeen suorittaa itse haun indeksiin. Lopuksi tulokset järjestetään jollain tavalla ja esitetään käyttäjälle. /18/

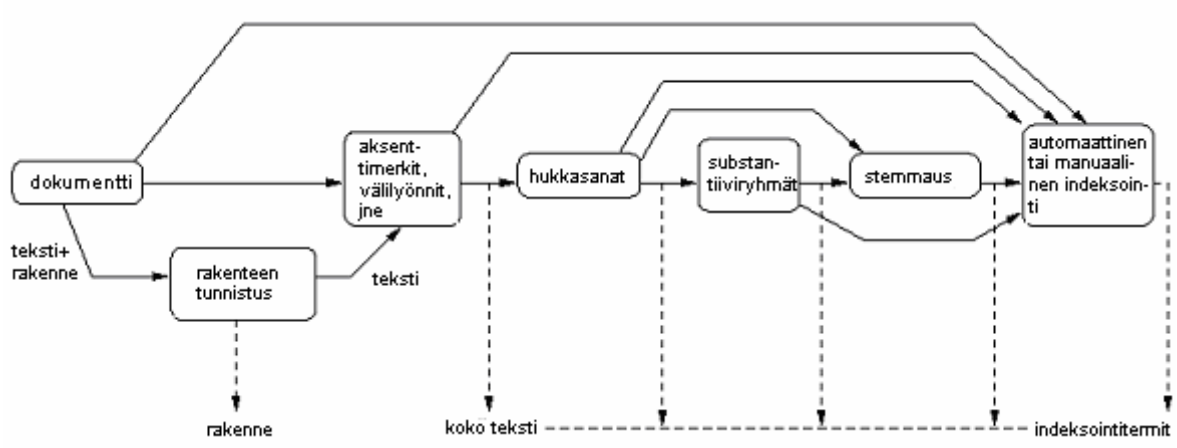
Tietoa talletetaan tyypillisesti kokoelmiin, joissa tieto on jaettu dokumenteiksi. Dokumentit koostuvat yleensä kentistä, kuten otsikko, tekijä, leipäteksti jne., jotka sisältävät itse tekstimuotoisen informaation. Kokoelmat voidaan esittää termidokumenttimatriisimuodossa, kuten kuvassa 11. /16/



Kuva 11. *Kokoelman C matriisiesitys /16/*

Kuvassa 11 sarakkeet vastaavat kokoelmasta löytyneitä termejä ja rivit vastaavat kokoelman muodostavia dokumentteja. Dokumentti D_i esitetään siis T:n pituisena, binäärisenä termivektorina $\{d_{i1},d_{i2},\ldots,d_{iT}\}$, missä alkio d_{ij} on yksi, kun dokumentti, jonka järjestysluku on j, sisältää termin, jonka järjestysluku on i. Muussa tapauksessa alkion arvo on nolla. /16/

Kaikkea tekstissä esiintyvää ei kuitenkaan yleensä indeksoida laskennallisen kuorman vähentämiseksi. Usein yleiset, indeksoinnin suhteen epäinformatiiviset käsitteen kuten partikkelit ja pronominit kerätään hukkasanojen (engl. stopword) listaan, joita ei indeksoida. Lisäksi sanat voidaan stemmata, eli etsiä sanarunko tai kehittyneemmin lemmata, eli palauttaa perusmuotoon, jotta kaikki saman termin eri taivutusmuodot indeksoituisivat samalle sarakkeelle, eivätkä muodostaisi omia sarakkeitaan matriisiin. /18/ Myöhempien hakujen kannalta tämä on erittäin oleellista, koska muuten taivutetut muodot eivät vastaa perusmuodolla tai toisella sijamuodolla tehtyihin kyselyihin, joka vähentää saantia hyvinkin merkittävästi suomen kaltaisissa, sijapäätteitä käyttävissä kielissä.



Kuva 12. *Dokumentin looginen esitys miten alkuperäisestä tekstistä päädytään indeksoitaviin termeihin /18/*

Kuvassa 12 on esitetty dokumentin looginen esitys, josta nähdään alkuperäisen tekstin indeksoitaviksi termeiksi muuttava muokkausprosessi. Sinällään koko prosessia ei ole vält-

tämätöntä käydä läpi loppuun saakka, vaan indeksi voidaan rakentaa jostain välimuodostakin. Tarkemmin eriteltynä kuvattu prosessi lähtee liikkeelle dokumentista, joka saattaa sisältää tekstin lisäksi myös rakenneinformaatiota esimerkiksi dokumentin esityksestä. Dokumentin teksti jatkaa eteenpäin ja siitä poistetaan välimerkit ja hukkasanat. Substantiiviryhmät ovat substantiiveja, jotka yhdessä muodostavat jonkin yksittäisen käsitteen (esimerkiksi Irakin sota) ja jotka ovat mielekkäitä indeksointitermejä kokonaisuuksina. Viimeiseksi voidaan vielä suorittaa stemmaus tai lemmaus ennen termien sijoittamista lopulliseen indeksiin. /18/

4.1.2 Boolen hakumalli

Boolen hakumalli (engl. Boolean retrieval model) perustuu kolmeen Boolen loogiseen operaattoriin: unioniin (OR), leikkaukseen (AND) ja komplementtiin (NOT). Kuvan 11 mukaisessa termidokumenttimatriisiesityksessä kyselyt on helppo esittää matemaattisessa muodossa. /16/

$$Q_{AND} = T_j \; AND \; T_k \tag{2}$$

Kaavassa 2 /16/ on esitetty kysely AND-operaattorin tapauksessa, joka voidaan siis esittää kahden termivektorin leikkauksena. Tulos palautetaan kaavan 3 /16/ mukaan.

$$D_{QAND} = \{d_i \mid (d_{ij} = 1) \cap (d_{ik} = 1)\} \tag{3}$$

Kaavassa 3 siis palautetaan dokumentti *i*, jos sen termivektorissa esiintyvät sekä termi *j*, että termi *k*. Vastaavat kaavat voidaan helposti esittää myös unionille ja komplementille. /16/

Boolen malli on helppo implementoida ja se on laskennallisesti tehokas, mikä on tehnyt siitä de facto standardin nykyisissä hakukoneissa. Mallin toinen vahvuus on sen ilmaukselinen voima ja selkeys. Valinnat määritellään yksikäsitteisesti, mutta toisaalta Boolen hakua voidaan rajata tai laajentaa monilla eri tavoilla. /14/

Boolen hakumallin heikkoutena on sen epäintuitiivisuus, mikä nostaa oppimiskynnyksen kohtuullisen korkeaksi. Boolen logiikassa käytetään arkikielestä tuttuja sanoja (ja, tai), joiden merkitys kuitenkin on hyvinkin erilainen. Tavallisessa kielessä ”A ja B” tarkoittaa tyypillisesti joukkoa, joka on suurempi kuin A. Samoin ’tai’ on arkikielessä tyypillisesti eksklusiivinen eikä inklusiivinen niin kuin Boolen logiikassa. Lisäksi täyden hyödyn saavuttamiseksi täytyy Boolen hakumallissa käyttää varsin suurta määrää sisäkkäisiä sulkuja, joiden hahmottaminen on aloittelijalle vaikeaa. /14/

Oppimiskynnyksen lisäksi Boolen hakumallin ongelmana on hienojakoisuuden puute. Leikkaus ei erittele sellaisen tapauksen välillä, jossa mikään ehto ei täyty ja sellaisen, jossa kaikki paitsi yksi täyttyvät. Vastaava, mutta käänteisenä, on totta myös unionin osalta. Tästä seuraa, että OR-operaattorin tarkkuus ja AND-operaattorin saanti ovat liian pienet. Perinteinen Boolen hakumalli ei myöskään tarjoa tapaa järjestää tuloksia relevanssin mukaan tai mallintaa epätarkkuutta hakutermeissä. /14/

Tavallisessa Boolen hakumallissa voidaan loogisten operaattorien lisäksi käyttää läheisyysvaatimuksia (tiettyjen sanojen pitää esiintyä lähekkäin tekstissä), kenttähakua, jolloin

haku kohdistetaan johonkin dokumentin tiettyyn kenttään (esim. otsikko, tekijä, jne.) ja lemmausta, jolloin hakuun vastaavat myös hakusanojen taivutetut muodot. /14/

Boolen hakumallista on kehitetty monia versioita, jotka pyrkivät paikkaamaan sen heikkouksia. Esimerkiksi Smart Boolean ottaa lähtökohdakseen tavallisen Boolen hakumallin ja pyrkii lisäämään siihen toiminnallisuutta, joka helpottaa tehokkaiden kyselyiden muodostamista tavallisen käyttäjän näkökulmasta. Smart Booleanissa käyttäjä muodostaa arkikielisen hakulauseen, joka muutetaan Boolen lausekkeeksi. Läheisyysvaatimuksilla, kenttähaulla ja lemmauksella saatavan tulossanan samankaltaisuuden perusteella voidaan myös muodostaa relevanssijärjestys hakutuloksille. Jos hakusanoja on esimerkiksi kaksi, arvostaa algoritmi sellaisen dokumentin, jossa nämä kaksi sanaa esiintyvät lähekkäin otsikossa haetussa taivutusmuodossa sellaisen dokumentin yli, jossa näin ei ole. /15/

Boolen mallia on myös kehitetty sumean logiikan avulla hienojakoisemmaksi ja vähemmän ehdottomaksi. P-normihaussa /15/ ideana on, että käsitteen ja dokumentin suhde ei ole binäärinen, vaan sumea, eli d_{ij} voi saada arvoja väliltä nollasta yhteen. P-normihaussa Boolen operaattorit sumeutetaan, eli niille annetaan P-arvo. Tämä arvo kertoo operaattorin ehdottomuuden, jonka soveltaminen voi olla laskennallisesti raskas operaatio. Lopuksi käyttäjälle esitetään tulosdokumentit laskevassa relevanssin todennäköisyysjärjestyksessä. /15/

4.1.3 TF-IDF

TF-IDF-painotus (term frequency – inverse document frequecy) pyrkii mallintamaan jäsenyysfunktiota (engl. membership function), joka määrittää millä painotuksella tietty dokumentti pitää indeksoida tietyn termin suhteen. TF-osan painotuksessa pyritään ottamaan huomioon se, että koko dokumentissa useaan kertaan esiintyvät termit karakterisoivat dokumenttia harvemmin esiintyviä termejä enemmän. IDF-osa puolestaan huolehtii siitä, että koko aineistossa yleisesti esiintyvät termit eivät dominoi painotuksien laskentaa. Kokeellisesti on osoitettu, että TF-IDF tuottaa paremman tuloksen haulle kuin binääriseen indeksointiin perustuva järjestelmä. /16/

TF-IDF:n laskemiseksi on kehitetty useita kaavoja. TF, eli termifrekvenssi, on yksinkertaisesti tietyn termin ilmenemismäärä tietyssä dokumentissa jaettuna kyseisen dokumentin kokonaissanamäärällä. /17/

$$tf_i = \frac{n_i}{\sum_k n_k} \tag{4}$$

Kaavassa 4 n_i on tietyn käsitteen esiintymismäärä ja nimittäjässä on kyseisen dokumentin kokonaistermimäärä. IDF, eli käänteinen dokumenttifrekvenssi, määrittää tietyn termin harvinaisuutta koko dokumenttikokoelmassa. /17/

$$idf_i = \log \frac{|D|}{|\{d : t_i \in d\}|} \tag{5}$$

Kaavassa 5 $|D|$ on kokoelman dokumenttien yhteislukumäärä ja $|\{d:t_i \in d\}|$ on niiden dokumenttien määrä, joissa termi t_i esiintyy. Lopullinen TF-IDF-paino on TF:n ja IDF:n tulo, joka normalisoidaan välille yhdestä nollaan. /17/

4.2 Kyselyn laajentaminen

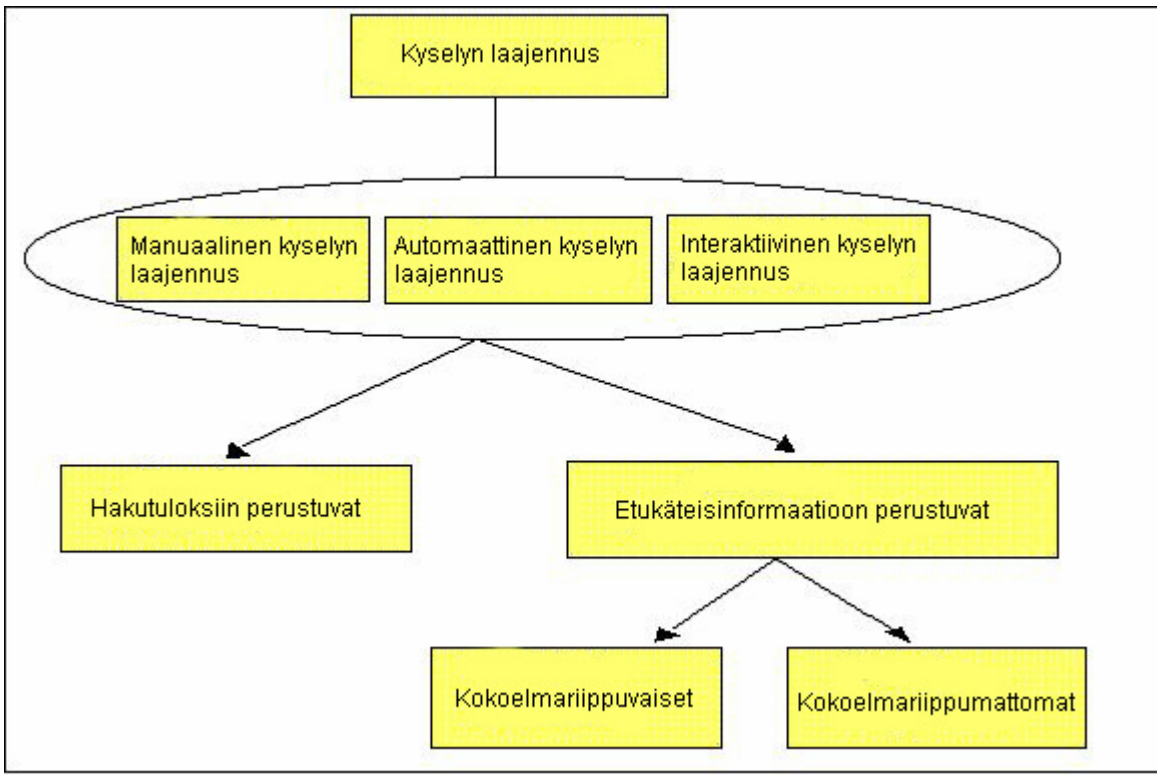
4.2.1 Yleistä

Haut ovat tyypillisesti epätäydellisiä kahdella eri tavalla: ensinnäkin, osa palautetuista dokumenteista on epäoleellisia etsityn informaation kannalta. Toiseksi osa oleellisista dokumenteista jää palauttamatta. Ensimmäinen piirre mitataan tarkkuutena, joka määrää kuinka suuri osa haussa palautetuista dokumenteista oli oleellisia. Toinen piirre on saanti, joka määrittää kuinka suuri osa tietokannan oleellisista dokumenteista palautettiin. Tyypillisesti nämä kaksi ominaisuutta ovat kääntäen verrannollisia siten, että tarkkuuden parantuessa saanti pienenee ja toisin päin. Optimaalista tasapainoa ja molempien kasvattamista voidaan edesauttaa erilaisilla informaation hakumalleilla (engl. information retrieval model), joista kyselyn laajentaminen on yksi. /14/

Kyselyn laajentaminen (engl. query expansion), josta käytetään myös nimitystä hakujen käsitteellinen laajentaminen, tarkoittaa haussa käytettävän termistön laajentamista hakukäsitteiden mukaan jonkin periaatteen mukaisesti. Tavallisesti haussa ei löydy dokumentteja, jos niissä ei esiinny haettavia termejä niiden täsmällisessä muodossa. Kyselyn laajentamisen tarkoituksena on sisällyttää hakuun hakutermien lisäksi näiden synonyymeja tai muita sanoja, joilla on jokin tärkeäksi katsottu suhde alkuperäisiin hakutermeihin. Näin saadaan tulosjoukon piiriin haettua nekin aihetta käsittelevät dokumentit, jotka eivät suoraan sisällä hakutermejä siinä muodossa, missä ne hakukoneelle annettiin. /13/

Ilmeinen ongelma kyselyn laajentamisessa on kyseisen haun kannalta epärelevanttien termien lisääminen hakuun, joka voi tuottaa tulosjoukkoon vääriä dokumentteja. Näin siis haun saanti kasvaa tarkkuuden kustannuksella. /13/

Kyselyn laajentaminen jaetaan tyypillisesti kolmeen luokkaan toimintatavan mukaan: manuaaliseen-, automaattiseen- ja interaktiiviseen kyselyn laajentamiseen. Nämä jaetaan edelleen hakutuloksiin perustuviin ja etukäteisinformaatioon perustuviin menetelmiin, joista jälkimmäinen jakaantuu vielä kokoelmariippuvaisiin ja -riippumattomiin. Tämä jako on esitetty kuvassa 13. /13/



Kuva 13. *Kyselyn laajentamisen jako erilaisiin menetelmiin /13/*

Kuvasta 13 nähdään, että toteutuksen automaattisuudesta riippumatta kyselyn laajennuksen implementaatiot jakaantuvat laajennukseen käytettävien termien lähteen sekä laajennustermien lajittelualgoritmin (engl. ranking algorithm) mukaan. Termien lähde voi joko iteratiivisesti perustua edellisiin hakutuloksiin tai sitten voidaan käyttää joko korpusperustaista tai kokoelmasta riippumatonta käsitelaajennusta. /13/

4.2.2 Relevanssin arvioinnista

Relevanssi on keskeinen käsite tiedonhaussa, jonka tarkoituksena on löytää nimenomaan relevantteja dokumentteja. Relevanssin käsite on intuitiivisesti ymmärrettävissä, esimerkiksi sanakirjamääritelmän ”liittyy käsiteltävään aiheeseen” kautta, mutta formaali määrittely on vaikeampaa. Saracevic määrittelee relevanssin viideltä eri kannalta: /19/

- Systeeminen tai algoritminen relevanssi (engl. system or algorithmic relevance) tarkoittaa haun ja palautettujen tai palauttamattomien dokumenttien relevanssialgoritmiin. Toisin sanottuna algoritmisella relevanssilla tarkoitetaan tietyn algoritmin tuottamaa relevanssia.
- Aihekohtainen relevanssi (engl. topical or subject relevance) tarkoittaa haun ja palautettujen tai kokoelmasta löytyvien dokumenttien aiheen suhdetta toisiinsa. Tässä oletetaan, että aihe voidaan jotenkin määrittää.
- Kognitiivinen relevanssi (engl. cognitive relevance or pertinence) tarkoittaa käyttäjän tarvitseman tiedon ja palautettujen tai palauttamattomien dokumenttien tietosisällön suhdetta toisiinsa.
- Tilannekohtainen relevanssi (engl. situational relevance or utility) tarkoittaa tilanteen tai tehtävän ja palautettujen tai palauttamattomien dokumenttien suhdetta

toisiinsa. Ratkaisevaa on dokumenttien hyödyllisyys ongelman ratkaisemisessa tai päätöksen teossa.

- Motivaatiorelevanssi (engl. motivational or affective relevance) tarkoittaa käyttäjän aikeiden tai tavoitteiden ja palautettujen tai palauttamattomien dokumenttien suhdetta toisiinsa.

Tässä työssä ei relevanssin syvempään olemukseen kuitenkaan puututa tarkemmin, vaan oletetaan yksinkertaisesti, että tietokannasta on mahdollista erottaa joukko dokumentteja, jotka ovat muita relevantimpia johonkin hakuun liittyen. Käytännössä tämä siis vastaa jotakuinkin edellä esiteltyä aihekohtaista relevanssia.

Palautettujen dokumenttien relevanssin arviointi on ensiarvoisen tärkeää onnistuneen kyselyn laajentamisen kannalta. Relevanssin arviointi voidaan sälyttää kokonaan käyttäjän tehtäväksi, kuten manuaalisessa kyselyn laajentamisessa (ks.4.2.3), mutta toimenpidettä voidaan myös automatisoida eri tavoin. Käyttäjän kannalta yksinkertaisimmillaan järjestelmä voi kysyä joukon kyllä-ei-kysymyksiä ja automatisoida itse laajennukseen käytettävien termien valinnan, mitä kutsutaan relevanssipalautteeksi (engl. relevance feedback). Tarkoituksena on tuottaa lisää haluttuja dokumentteja ja karsia pois ei-toivottuja ilman että käyttäjän täytyy tietoisesti tehdä valintoja hakustrategiaansa liittyen. /13/

Tyypillinen automaattinen relevanssipalautetoiminto käsittelee käyttäjän alkuperäisen kyselyn, joka tuottaa joukon dokumentteja. Tästä joukosta käyttäjä valitsee yhden tai useamman relevantin dokumentin, joiden pohjalta painotetaan alkuperäisiä hakutermejä ja lisätään uusia. Lopuksi järjestelmä luo uuden kyselyn ja tuottaa uuden joukon dokumentteja. Toinen tapa on käyttää hyväksi koko alkuperäiseen kyselyyn vastaukseksi saatua tulosjoukkoa, josta pyritään löytämään hakutermien synonyymit tämän nimenomaisen haun mittakaavassa. /21/

Relevanssia arvioiva järjestelmä pyrkii mallintamaan relevanttien ja epärelevanttien dokumenttien todennäköisyysjakaumat. Tätä varten pitää käyttäjän valita vähintään yksi dokumentti relevantiksi, joskin suurempi näytekoko voi parantaa tuloksia. Useissa järjestelmissä käyttäjä arvioi ensimmäisten kymmenen tai kahdenkymmenen dokumentin joukkoa ja valitsee sieltä relevanteiksi kokemansa dokumentit. Periaatteessa mitä useampia dokumentteja valitaan, sitä tarkempi seuraava haku on, mutta käytännössä useimmat tutkimukset suosittelevat valittavaksi noin viittä dokumenttia. /13/

Toinen kysymys on käyttäjälle näytettävä informaatio, jonka perusteella relevanssia arvioidaan. Tutkimuksissa on ilmennyt, että tietyn dokumentin arvioitu relevanssi muuttuu merkittävästi riippuen siitä, minkä kentän mukaan käyttäjä arvioi dokumentteja. Pelkät otsikot eivät tyypillisesti riitä, vaan parhaimpaan tulokseen päästään näyttämällä otsikon lisäksi esimerkiksi tiivistelmä tai osia tekstistä. /13/

Relevanssia arvioidaan tyypillisesti binäärisesti, eli dokumentti on joko relevantti tai epärelevantti. Tämä on merkittävä yksinkertaistus, mutta liukuva relevanssiasteikko on hankala käyttäjän kannalta. Sinällään mahdollisuus määritellä esimerkiksi vain osa dokumentista relevantiksi mahdollistaisi tarkemmat haut. /13/

Huomattavaa on, että relevanssi ei aina suoraan määritä tietyn dokumentin hyödyllisyyttä käyttäjän kannalta. On hyvinkin mahdollista, että sinällään relevantti dokumentti on esi-

merkiksi vanhentunut tai kirjoitettu väärällä kielellä. Tämä voi aiheuttaa ongelmia, koska dokumentin määrittäminen epärelevantiksi voi johtaa järjestelmää harhaan. /13/

4.2.3 Manuaalinen kyselyn laajentaminen

Manuaalinen kyselyn laajentaminen tarkoittaa prosessia, jossa ihmiskäyttäjä laajentaa kyselyä täysin ilman automaatiota edellisen kyselyn tuloksiin nojaten. Tälle prosessille on kehitetty useita malleja, joista useimmat nojaavat Boolen hakumalliin. /13/ Erilaisia hakustrategioita, -taktiikoita ja heuristiikkoja on kehitetty suuri joukko, mutta tässä niitä esitellään vain pääpiirteissään.

Manuaalisessa kyselyn laajentamisessa käyttäjä suorittaa ensin yksinkertaisen haun tietokantaan ja palautettujen dokumenttien perusteella joko laajentaa tai kaventaa hakua unionilla tai leikkauksella. Komplementilla voidaan sulkea epäoleellisia dokumenttityyppejä haun ulkopuolelle. /14/ Erilaiset hakustrategiat muuttavat tätä tyypillistä lähestymistapaa. /13/

Hakustrategia tarkoittaa suunnitelmaa, jolla jotakin tiedonhakutapahtumaa toteutetaan kokonaisuudessaan. Hyvä hakustrategia edellyttää yksityiskohtaista ymmärrystä hakujärjestelmästä, käytetystä indeksoinnista ja tietokannan luonnissa käytetyistä konventioista, sekä täyttä ymmärrystä tiedontarpeesta ja hakutavoitteista, eli halutaanko korkea saanti vai tarkkuus. Esimerkkinä hakustrategiasta on rakennuspalikkastrategia (engl. building blocks strategy), jossa haettava aihe jaetaan käsitteisiin tai fasetteihin ja nämä jaetaan edelleen termijoukoiksi. Termijoukon sisällä termit liitetään yhteen unioniksi ja eri fasetit yhdistetään leikkauksella. /13/

Verkkopohjaisissa arkistoissa on perinteisiin paperiarkistoihin verrattuna enemmän sisäänpääsypisteitä, mutta nämä ovat usein piilotettuja eli eivät ole suoraan lähestyttävissä käyttöliittymässä. Tästä johtuen haku jää pinnalliseksi, jos käyttäjä ei tunne järjestelmää, johon kyselyä suorittaa. Ongelman ratkaisemiseksi on kehitetty sääntöjä ja olettamuksia, jotka tunnetaan hakutaktiikoina ja heuristiikkoina. Hakustrategian käsittelee koko hakua kokonaisuutena, kun taas hakutaktiikat ja heuristiikat keskittyvät yksittäisiin toimenpiteisiin, joita suoritetaan vain tietyssä vaiheessa hakua. /13/

Tutkimuksissa on ilmennyt, että tyypillisimmin hakutermejä käytetään varsin niukasti. Lisäksi useasta hausta iteratiivisesti koostuva tiedonhakuprosessi on todettu tehokkaammaksi kuin useisiin hakutermeihin ja laajaan valmistautumiseen perustuvat toimintatavat. /13/

4.2.4 Automaattinen kyselyn laajentaminen

Automaattisessa kyselyn laajentamisessa hakusysteemi itsessään on vastuussa alkuperäisten kyselytermien laajentamisesta. Tämä asettaa korkean vastuun järjestelmän kehittäjälle ja vaatii tehokasta toteutusta relevanssin arvioinnille, lisätermien valinnalle ja tulosten järjestämiselle. Tämän vuoksi puhdasta Boolen hakumallia ei tyypillisesti käytetä osana automaattista kyselyn laajentamista. /13/

Automaattinen kyselyn laajentaminen voidaan jakaa kahteen luokkaan: relevanssipalautetta (ks. 4.2.2) hyödyntäviin ja sokeaa palautetta (engl. blind feedback) hyödyntäviin mene-

telmiin. Ensin mainittu nojaa käyttäjältä saatuun palautteeseen ensimmäisen kyselyn tuloksista, kun taas jälkimmäinen on täysin automaattinen järjestelmä, joka pyrkii laajentamaan kyselyä tyystin ilman käyttäjän vaikutusta. /13/

Relevanssipalautemenetelmissä käyttäjä valitsee alkuperäisen haun tuloksista relevantit dokumentit ja hakua jatketaan kyselyn laajentamisella näiden pohjalta. Menetelmän vahvuuksia ovat seuraavat: /22/

- Käyttäjältä ei vaadita syvempää osaamista kyselyn uudelleenmuokkausprosessista tai hakuympäristöstä.
- Hakuoperaatio jaetaan pienempiin osasiin, joiden avulla lopullista tulosta lähestytään askel kerrallaan.
- Hakutermit mukautuvat etsittävään dokumenttikokoelmaan luonnollisella tavalla painottaen käyttäjälle aidosti relevantteja termejä

Usein relevanssipalautteeseen perustuva kyselyn laajentaminen koetaan käyttäjien keskuudessa hankalaksi. Tyypillisin hakutilanne esimerkiksi Internetin hakukoneisiin on hyvin lyhyt koostuen vain yhdestä tai kahdesta hakutermistä, joiden valintaan käyttäjä ei käytä paljoa aikaa. Jos käyttäjää pyydetään tarkentamaan hakua, saattaa hän jopa kokea hakukoneen kokonaisuudessaan liian hankalaksi ja siirtyä käyttämään toisia hakujärjestelmiä. Sokean palautteen automaattinen kyselyn laajentaminen pyrkii vastaamaan tähän haasteeseen /73/

Sokean palautteen kyselyn laajennuksessa haetaan alkuperäisen kyselyn perusteella pieni joukko dokumentteja, joiden oletetaan olevan relevantteja kyseiselle haulle täysin ilman käyttäjäpalautetta. Tätä dokumenttijoukkoa käytetään laajentamaan kysely samalla tavoin kuin perinteisenkin relevanssipalautteen menetelmissä. Ilmeisenä ongelmana on se, ettei järjestelmällä ole mitään takeita siitä, että ensimmäinen dokumenttijoukko todella edustaisi relevantteja dokumentteja. Jos joukossa on aihetta yleisesti käsitteleviä, mutta todellisen haun kannalta epärelevantteja dokumentteja, voi lopullinen tulosjoukko lähestyä epärelevanttia dokumenttijoukkoa. Toisaalta jos ensimmäisen haun tuloksena saadaan pääosin relevantti dokumenttijoukko, paranee haun tulos selkeästi. /21/

Sokean palautteen kyselyn laajentaminen voi siis parantaa tai haitata hakua. Testeissä on selvinnyt, että keskimääräisesti joka kolmas haku huononee, mutta muut kaksi kolmasosaa hauista paranevat merkittävästi. Lopullinen vaikutuksen keskiarvo suuria hakumääriä arvioitaessa on tilastollisesti merkittävä ja positiivinen. Tästä päätellen sokean palautteen kyselyn laajentaminen on hyödyllinen tekniikka ja yleisesti tutkimuksessa pyritään löytämään keinoja, joilla aiheen harhailua voidaan välttää. /21/

Aiheen harhailu aiheutuu alkuperäisen kyselyn tulosjoukon epärelevanttiudesta, joten luonnollinen tapa lähteä hakemaan ratkaisua sokean palautteen kyselyn laajentamisen ongelmaan on pyrkiä parantamaan alkuperäistä dokumenttijoukkoa. Yksi tapa tehdä tämä on seuraava: /21/

1. Haetaan ensimmäisellä kyselyllä suurempi dokumenttijoukko kuin mitä lopulta aiotaan käyttää kyselyn laajennukseen.

2. Jokaiselle löydetylle dokumentille lasketaan uusi painoarvo perustuen pelkän hakusanan lisäksi muihin relevanssi-indikaattoreihin, joita kyseisestä dokumentista löydetään.
3. Järjestetään dokumentit uuden painoarvon mukaan.
4. Valitaan laajennukseen haluttu määrä dokumentteja painoarvon mukaisessa järjestyksessä
5. Suoritetaan kyselyn laajennus.

Vaiheiden 2 ja 3 dokumenttien uudelleenjärjestys on algoritmin kriittisin vaihe. Sen pitää parantaa dokumenttien relevanssia, mutta välttää suosimasta tietyntyyppisiä dokumentteja yli muiden, jotta tulosjoukko ei painottuisi haitallisesti. /21/

Mahdollisia relevanssi-indikaattoreita hakutermien lisäksi ovat esimerkiksi läheisyysrajoitteet (engl. proximity constraints) ja sumeat Boolen operaattorit. Läheisyysrajoitteilla tarkoitetaan sitä, että dokumentissa hakutermien ilmentymisetäisyydelle toisistaan asetetaan jokin kynnysarvo, jonka ylittävät tapaukset hylätään. Jos hakutermejä on esimerkiksi kaksi, voidaan vaatia, että relevantissa dokumentissa pitää molempien termien esiintyä alle sadan sanan päässä toisistaan. Läheisyysrajoitteet perustuvat siihen, että hakutermit tyypillisesti liittyvät hyvin läheisesti toisiinsa hakijan toivomissa dokumenteissa ja näin voidaan karsia sellaiset dokumentit, joissa kaikki hakutermit esiintyvät, mutta toisistaan irrallaan eri kokonaisuuksien osina. /74/

Sumeat Boolen operaattorit, joissa dokumenteille annetaan painoa hakuehtojen osittaisestakin täyttämisestä (ks. 4.1.2) ovat myös yksi mahdollinen relevanssi-indikaattori, tavalla saadaan eri dokumenttien relevanssierot tarkemmin eriteltyä. Erityisen hyödyllinen metodi on tapauksissa, joissa hyvin harvat dokumentit täyttävät kaikki hakukriteerit, jolloin tavallisella, binaarisella Boolen logiikalla löydettäisiin vain pieni joukko tulosdokumentteja. Sumeilla Boolen operaattoreilla saadaan kuitenkin palautettua relevanssijärjestyksessä mielivaltaisen kokoinen joukko dokumentteja, joita voidaan käyttää kyselyn laajentamiseen halutulla tavalla. /74/

Termikorrelaatiot (engl. term correlation) ovat hyödyllinen tekniikka etenkin tyypillisimmissä hauissa, joissa on vain joukko leikkauksen muodostavia sanoja. Kaavassa 6 on tämä esitettynä matemaattisessa muodossa.

$$Sim(D) = \sum_{t_i \in Q \wedge t_i \in D} idf(t_i) \qquad (6)$$

Sim(D) on dokumentin D saama painoarvo, t_i on hakutermi i ja Q on itse kysely. IDF on käänteinen dokumenttifrekvenssi (ks. 4.1.3). Tällaisessa kyselyssä ei tehdä eroa kahden voimakkaasti toisiinsa sidoksissa olevan termin ja kahden täysin toisistaan riippumattoman termin välillä. Kuitenkin dokumentti, joka saa korkean painoarvon toisistaan riippumattomien termien avulla on tyypillisesti hyödyllisempi. /21/

Termikorrelaatio lasketaan tutkimalla kuinka usein tietyt kaksi sanaa esiintyvät samassa dokumentissa alkuperäiseen hakuun vastauksena saadussa, suuressa dokumenttijoukossa. Jos kaksi sanaa korreloivat keskenään (tai muodostavat fraasin), niiden oletetaan esiintyvän yhdessä monissa näistä dokumenteista. Jos taas sanat ovat toisistaan riippumattomia, ei

toisen sanan dokumentissa esiintymisen pitäisi nostaa toisen sanan esiintymistodennäköisyyttä merkittävästi. Tätä ajatusta hyödyntävä, uusi painoarvo voidaan laskea kaavan 7 avulla. /21/

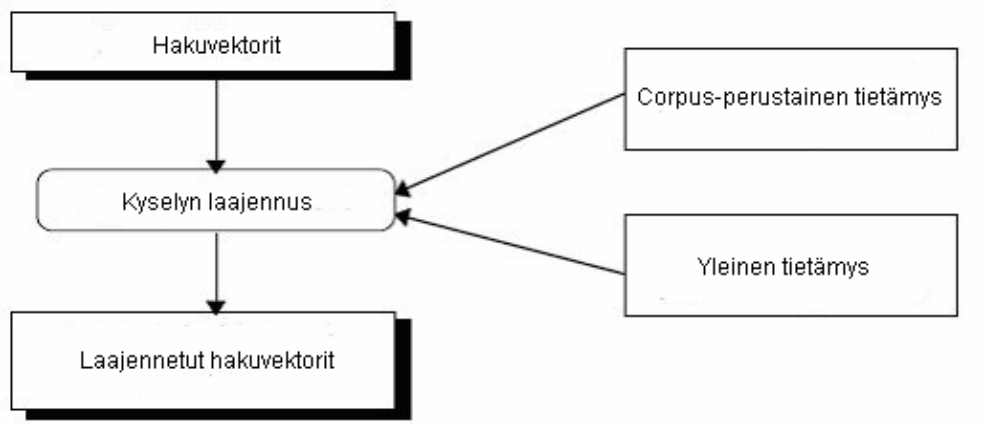
$$Sim_{new}(D) = idf(t_1) + \sum_{i=2}^{m} idf(t_i) \times \min_{j=1}^{i-1}(1 - P(t_i \mid t_j)) \quad (7)$$

Olkoon $df_s(t)$ dokumenttijoukossa S olevien sellaisten dokumenttien määrä, joissa termi t esiintyy. Kyselyn termit järjestetään nousevan $df_s(t)$:n mukaan joista ensimmäinen, eli kaikkein harvinaisin, tuottaa täyden IDF-painonsa lopulliseen painoon. Seuraavat termit tuottavat painon, joka riippuu siitä, kuinka hyvin jokin aiemmin esiintynyt termi ennusti tämän termin esiintymisen. Toisin sanottuna voimakkaasti edellisten termien kanssa korreloivien termien paino pienenee. $P(t_i|t_j)$, eli todennäköisyys sille, että t_i esiintyy, kun t_j on löytynyt, arvioidaan dokumenttijoukon S sanojen ilmenemisen mukaan jakamalla niiden dokumenttien määrä, joissa molemmat termit esiintyvät niiden dokumenttien määrällä, joissa vain t_j esiintyy. /21/

Lopputuloksena saadaan hausta dynaamisesti laajeneva kyselynlaajennus, joka automaattisesti sopeutuu kulloiseenkin corpukseen.

Toinen lähestymistapa automaattiseen kyselyn laajentamiseen on tietämyspohjainen (engl. knowledge-based), ei-esihaun tuloksiin nojaava toteutus. Tämä voidaan edelleen jakaa kahtia aihealuekohtaiseen ja yleiseen tietoon nojaaviin järjestelmiin. Yleensä käytännön sovelluksissa käytetään kuitenkin molempia tapoja. /52/

Tietämyspohjaisen kyselyn laajentaminen perustuu johonkin tietopohjaan, joka esitetään semanttisena verkkona. Käyttäjän suorittaessa haun, etsitään tästä semanttisesta verkosta lisätermejä, joilla on semanttisesti läheinen suhde alkuperäisiin hakusanoihin, ja lisätään ne automaattisesti suoritettavaan hakuun. /52/



Kuva 14. *Tietopohjainen kyselyn laajentamisprosessi /52/*

Kuvassa 14 on esitetty tietopohjainen kyselyn laajentamisprosessi, jossa alkuperäiseen kyselyvektoriin lisätään termejä sekä aihealuekohtaisesta että yleisestä semanttisesta tietämysverkosta. /52/

Keskeisin kysymys tietämyspohjaisessa kyselyn laajennuksessa on käytettävä tietämys /52/ ja yksi mahdollinen vastaus on käyttää ontologioita (ks. 2.2). Aihealuekohtainen tietopohja voidaan myös muodostaa korpusperustaisesti tilastollisilla menetelmillä hakujärjestelmän kohdetietokannasta, jolloin menetelmästä tulee tavallaan hybridi korpusperustaisen ja korpusriippumattoman menetelmän välillä. Tilastollinen hierarkia muodostuu termien yleisyydestä, jolloin kaikkein yleisimmät termit muodostavat verkon juuret ja niiden kanssa samoissa dokumenteissa esiintyvät, harvinaisemmat termit toimivat lapsina. Menetelmä perustuu olettamukseen, että yleisesti esiintyvät termit määrittävät laajempia käsitteitä ja jos kahden termin tiheysfunktiot vastaavat toisiaan muodoltaan, ovat termit hierarkkisessa suhteessa toisiinsa. /52/

Itse laajennus tapahtuu yksinkertaisesti semanttisen verkon kaaria seuraamalla. Laajentumista voidaan rajoittaa kahdella tekijällä, eli kuljettujen kaarien määrällä ja yksittäisestä solmusta lähtevien kaarien määrällä. Mitä syvemmälle kaaria seurataan, sen vähemmän painoa annetaan löydetyille termeille ja mitä enemmän kaaria tietystä termistä lähtee, sen vähemmän sille annetaan painoa. Ensimmäinen rajoite johtuu siitä, että semanttiset suhteet heikkenevät niiden välisen etäisyyden kasvaessa ja jälkimmäinen siksi, että laajasti kytkeytyneet termit ovat tyypillisesti juuritermejä ja näin ollen vähemmän oleellisia. /52/

Tarkemmin hakustrategiat voidaan luokitella eristettyihin- ja korreloiviin hakuihin. Ensin mainitut käsittelevät jokaisen hakutermin erillisenä toisistaan ja laajentavat ne yksittäin. Tämä voidaan tehdä laajentamalla hakua semanttisessa verkossa pelkästään laajempiin (eli yleisempiin) termeihin, jolloin saadaan parempi saanti tai pelkästään suppeampiin termeihin (eli harvinaisempiin), jolloin saadaan parempi tarkkuus. Periaatteessa laajennus voidaan suorittaa myös molempiin suuntiin, mutta yleisyyden muodostamassa hierarkiassa se ei ole tavallisesti mielekästä. Korreloivassa haussa puolestaan haetaan tietämyskannasta kyselyssä käytettyjä termejä yhdistävä reitti ja tätä käytetään laajentamaan kysely. /52/

4.2.5 Interaktiivinen kyselyn laajentaminen

Kolmas menetelmä kyselyn laajentamiseen on kahden ensiksi mainitun metodin yhdistelmä, eli interaktiivinen kyselyn laajentaminen. Siinä suoritetaan automaattinen kyselyn laajentaminen, jonka tuloksena saatu lista potentiaalisista lisähakutermeistä esitetään käyttäjälle. Tästä joukosta käyttäjä valitsee relevanteiksi katsomansa lisätermit ja ne sisällytetään uuteen hakuun. Tyypillisesti termit painotetaan automaattisesti ja esitetään käyttäjälle painojärjestyksessä. /13/

Interaktiivisessa kyselyn laajentamisessa vastuu laajennuksen onnistumisesta on siis jaettu järjestelmän ja käyttäjän välille ensiksi mainitun tarjotessa rajatun listan ja jälkimmäisen tehdessä lopullisen valinnan laajennustermien suhteen. Tämä aiheuttaa sen, että haun lopullisen onnistumisen tai epäonnistumisen syy-seuraussuhteiden selvittäminen voi olla hankalaa. /13/

Valintavaiheen termien lähde voi, kuten automaattisen kyselyn laajentamisenkin tapauksessa, olla joko esihaun tuloksiin tai vaihtoehtoisesti johonkin ennalta määrättyyn tietämysrakenteeseen perustuva. Jälkimmäisessä tapauksessa tietämysrakenne voi olla yleinen tai yksilöllinen haettavalle aineistolle. /13/

Teoriassa interaktiivinen kyselyn laajentaminen vaikuttaa erittäin lupaavalta, koska käyttäjä tuntee tarpeensa aina paremmin ja voi teoriassa ohjata hakua konetta paremmin. Käytännössä tämä voi kuitenkin olla hankalaa, koska pelkät termit eivät anna riittävästi informaatiota käyttäjälle, jotta tämä voisi tehdä hyviä laajennuspäätöksiä. Tämä puolestaan esittää haasteita käyttöliittymäsuunnittelulle, jonka pitäisi kyetä esittämään käyttäjälle tämän tarvitsema tieto hakutermien lisäämisen vaikutuksista kuitenkaan hämmentämättä käyttäjää liialla informaatiolla. /75/

4.2.6 Dokumentin laajentaminen

Dokumentin laajentamien (engl. document expansion) tarkoittaa dokumentin laajentamista sen termistön mukaisesti. Se siis vastaa kyselyn laajentamista toimintaperiaatteiltaan, mutta sen sijaan, että laajennus tehtäisiin haun yhteydessä, tehdään se erikseen etukäteen koko aineistolle indeksointivaiheessa. Tämä vastaa kyselyn laajentamisen keskeiseen heikkouteen, eli kyselyn uudelleenmuodostamisen luontaiseen tehottomuuteen, mutta samalla luonnollisesti menetetään dynaamisuuden tuomat edut. /51/

Kuten kyselyn laajennuskin, voidaan dokumentin laajennus toteuttaa joko korpusperustaisesti tai aineistoriippumattomasti. Ensin mainitussa tapauksessa dokumentteihin lisätään mahdollisia hakutermejä, jotka esiintyvät vastaavanlaisissa dokumenteissa. Tällainen käsittely on raskas operaatio, kun se tehdään koko aineistolle, mutta koska se suoritetaan indeksointivaiheessa, tehovaatimukset eivät koske järjestelmän loppukäyttäjää millään tavalla. Itse operaatiossa on mahdollista hyödyntää esimerkiksi käytettävää hakujärjestelmää. Esimerkkejä tästä ovat dokumenttikeskeinen- ja termikeskeinen dokumentin laajennus, joista ensin mainitussa aineiston jokainen kokonainen dokumentti ajetaan hakujärjestelmän lävitse kyselynä ja tulosdokumenttien suurimman relevanssin saaneet termit lisätään kyselyssä käytettyyn dokumenttiin. /51/

Termikeskeinen dokumentin laajennus vastaa läheisesti kyselyn laajennusta, mutta toimenpide on käänteinen. Kyselyn laajennuksessa haetaan dokumentteja, jotka ovat jonkin termin kannalta relevantteja, mutta joissa kyseinen termi ei kuitenkaan välttämättä esiinny. Termikeskeisessä dokumentin laajennuksessa termi lisätään sille relevantteihin dokumentteihin, joissa se ei alun perin esiintynyt. Tämä metodi on merkittävästi dokumenttikeskeistä dokumentin laajennusta nopeampi, mutta laajentuva aineisto aiheuttaa ongelmia, koska muuttuva aineisto tuottaa muuttuvat termit, joka tarkoittaa, että aiemmin lisätyt termit eivät ehkä olekaan optimaaliset. Eräs ratkaisu, joskin raskas, on koko prosessin suorittaminen alusta asti uudestaan aina kun kokoelma on kasvanut tietyn dokumenttimäärän verran. /51/ Tätä ongelmaa ei kuitenkaan esiinny aineistoriippumattomilla menetelmillä.

Korpusperustainen dokumentin laajennus on todettu kyselyn laajennusta tehottomammaksi toimenpiteeksi, jolla ei usein ole tilastollisesti merkitsevää vaikutusta hakujen tuloksiin. /51/ Asiasanastoperustainen, termien välisiin suhteisiin nojaava dokumentin laajennus on kuitenkin vaikuttanut lupaavammalta. /40/

4.3 Ontologiat ja kyselyn laajentaminen

4.3.1 Hahmonvalinta ontologisin perustein

Piirteenvalinta (engl. feature selection) tarkoittaa sitä, että dokumentista etsitään oleellisia piirteitä ja valitaan nämä edustamaan kyseistä dokumenttia sen koko sanasisällön sijaan. Valitun piirrejoukon pitäisi kuvailla dokumentin sisältö paremmin tai ainakin yhtä hyvin kuin kaikkien dokumentin piirteiden (termien ja näiden järjestyksen). Yksinkertaisimmillaan tämä tarkoittaa yleisten ja dokumentin relevanssin kannalta turhien sanojen kuten partikkelien ja pronominien jättämistä pois laskuista, mutta periaatetta voidaan soveltaa laajemminkin. /41/

Aiemmin piirteenvalintaa on suoritettu lähinnä tilastollisilla menetelmillä ja ne toimivatkin verrattain hyvin klusteroinnissa ja luokittelussa. Korkeamman tason merkitykset jäävät kuitenkin piiloon puhtaasti tilastollisia menetelmiä käytettäessä. Hyödyntämällä olemassa olevaa tietämystä dokumentin käyttämästä käsitteistöstä voidaan näitä korkeamman tason käsitteitä havaita ja hyödyntää. Tämä olemassa oleva tietämys voidaan esittää ontologiana, joka kuvaa käsitteistön hierarkian ja määrittelee käsitteiden väliset suhteet, joita voidaan sitten hyödyntää hahmonvalinnassa. /41/

Ontologian käyttäminen kuitenkin ohjaa hahmonvalintaa voimakkaasti. Jos käytetään puhtaasti ontologista lähestymistapaa, ei ontologian ulkopuolisia käsitteitä voida sisällyttää hahmonvalinnan löytämiin piirteisiin. Poikkeuksena tähän olisi ontologian populointi käsitellyn aineiston mukaisesti, mutta tämä ei tyypillisesti ole automaattinen prosessi, vaan vaatii ihmisohjausta tilastollisten menetelmien tueksi. /41/

Yleisesti ottaen ontologian luojan tapa hahmottaa ontologian kuvaamaa käsitteistöä vaikuttaa merkittävästi ontologian rakenteeseen ja täten myös sitä hyödyntävään hahmonvalintaan. Ontologiaan valitut sanat ja käsitteet eivät myöskään välttämättä tehokkaasti kuvaa jonkin dokumentin kannalta oleellisia termejä ja konsepteja. Ongelma korostuu, jos ontologia ja käsiteltävä datajoukko ovat erilaisten ihmisten tai yhteisöjen tuottamia, jolloin painotukset saattavat olla hyvinkin erilaiset. Parhaat tulokset ontologiapohjaisessa hahmonvalinnassa saavutetaankin silloin, kun sekä ontologia että datajoukko edustavat samaa katsantokantaa. Tämän vuoksi hahmonvalinnan tehokkuuden kannalta yleiset ontologiat eivät välttämättä tuota yhtä hyviä tuloksia kuin varta vasten tietylle aihealueelle suunnitellut, korpusperustaiset ontologiat. /41/

Termivektoriesityksen korvaaminen ontologisilla käsitevektoreilla tuottaa kaksi merkittävää etua. Ensinnäkin, synonyymien ongelma ratkeaa, koska samaa merkitsevät eri termit osoittavat samaan käsitteeseen. Toiseksi käsitteet ovat tyypillisesti termejä eksaktimpia ja tarkemmin määriteltyjä, mutta usein samalla laajempia ontologian hierarkian kontekstissa. Sinällään dokumentin toimiva käsitteistäminen saavuttaa hahmonvalinnan päämärän edustamalla dokumenttia paremmin kuin pelkkä termivektori. /40/

4.3.2 Disambiguointi

Ontologioiden keskeinen piirre on sanojen merkityksen selvittämien, joka vaatii termien ja käsitteiden erottamista toistaan. Yksi termi voi viitata useampaan käsitteeseen ja tämä sa-

nan merkityksen disambiguoiminen (engl. word sense disambiguation, WSD) on eräs keskeisimmistä haasteista automaattiselle ontologioiden hyödyntämiselle. Kyselyn laajentamisessa tämä problematiikka on ehdottoman keskeinen. /25/

Tutkimukset WordNetillä ovat osoittaneet, että jos disambiguoinnin tarkkuus on alle 90 %, ei saavuteta merkittäviä hyötyjä tietyissä hakutekniikoissa. Toisaalta jos tarkkuus on yli 40 %, eivät tulokset ole huonompia kuin ilman WSD:tä suoritetut haut. /27/ WordNetistä ei kuitenkaan hyödynnetty sen ontologisia ominaisuuksia, kuten käsitteiden monimutkaisempia suhteita, vaan vain synonyymi- ja hyperonyymiominaisuuksia, eli tuloksia kannattaa pitää vain suuntaa antavina.

On tärkeää huomata, että automaattisella disambiguaatiolla ei koskaan pyritä täydelliseen tulokseen. Ihmisetkin kykenevät disambiguoimaan vain n. 97 % tarkkuudella /30/, koska tekstistä on joskus yksinkertaisesti mahdotonta päätellä, mitä termiä vastaavaa käsitettä tarkoitetaan. Joidenkin tutkimusten mukaan ontologisia suhteita käytettäessä riittää huomattavasti 90 % alhaisempi tarkkuus parantamaan hakutuloksia tilastollisesti merkittävästi. /29/

Disambiguoinnin tarkkuuden arviointi on myös hankalaa ja työlästä. Koska disambiguointialgoritmien toiminta perustuu disambiguoitavaa termiä ympäröivään tekstiin, on aidosti puolueettoman testiaineiston kehittäminen mahdotonta. Tyypillisin tapa arvioida jonkin algoritmin toimintaa on verrata sitä ihmisen käsittelemään aineistoon, mutta tällöinkin ongelmana on yleistettävyys. Koska toiset termit ovat hankalampia disambiguoida kuin toiset lauseyhteyksistä ja sanaluokista riippuen, ei suorituksesta voida yleistää muutamien termien perusteella mitään yleisempää tulosarvoa. /30/

Jos pyrkimyksenä on dokumenttien klusterointi esimerkiksi aihealueittain ontologisten suhteiden mukaan, on disambiguaatio vähemmän tärkeää. Yksittäisten termien muuttaminen käsitteiksi mahdollistaa niiden liittämisen osaksi laajempaa kontekstia, jolloin on helpompaa tunnistaa dokumentille läheisiä muita aiheita. Käytännössä tämä toimii vastaavalla tavalla kuin dokumentin laajennus, eli useiden termien viitatessa johonkin käsitteeseen ontologisten suhteiden läheisyyden mukaan, voidaan tuo käsite lisätä dokumentin käsitevektoriin. /40/

Dokumenttien laajennusta ontologisella klusteroinnilla on suoritettu onnistuneesti hyvinkin alkeellisia disambiguaatiostrategioita noudatellen. Yksinkertaisin ratkaisu on olla tekemättä disambiguaatiota ollenkaan ja lisätä yksinkertaisesti kaikki termiä vastaavat käsitteet dokumenttiin. Toinen yksinkertainen ratkaisu on järjestää ontologian käsitteet niiden yleisyyden mukaan ja termin viitatessa useampaan käsitteeseen valita niistä yleisin. Tällöin virheet ovat tilastollisesti harvinaisempia kuin liittämällä termi satunnaiseen konseptiin, mutta kaikkia dokumentin termejä vastaa kuitenkin vain yksi ainoa käsite. Ilmeiseksi ongelmaksi luonnollisesti muodostuu se, että harvinaisempia käsitteitä ei tällä menetelmällä disambiguoidussa aineistossa esiinny laisinkaan. /40/

Huomattavasti hienostuneempi ja enemmän lisäarvoa tuottava tapa on pyrkiä disambiguoimaan termit käsitteiksi niitä ympäröivää kontekstia hyödyntäen. Tämä voidaan tehdä esimerkiksi määrittelemällä käsitteen semanttiseksi naapurustoksi sen kaikki suorat ala- ja yläkäsitteet. Termiä disambiguoitaessa valitaan se käsite, jonka semanttinen naapurusto

esiintyy voimakkaimmin dokumentissa. Jo näillä äärimmäisen yksinkertaisillakin disambiguaatiomenetelmillä saatiin aikaan positiivisia tuloksia hakujen tehokkuuteen. Viimeiseksi kuvatulla semanttiseen naapurustoon perustuvalla menetelmällä saatiin 8,4 %:n suhteellinen kasvu hakutulosten laatuun, kun ontologiana käytettiin WordNetiä ja aineistona Reutersin artikkeliaineistoa. /40/

4.3.3 Kyselyn laajentaminen ontologisilla suhteilla

Ei-korpusperustainen kyselyn laajennus on erittäin houkutteleva menetelmä, koska se mahdollistaisi samojen tekniikoiden soveltamisen mihin tahansa aineistoon. Tilastollisten laajennusmenetelmien suhteen tulokset ovat olleet tällä alueella heikkoja lukuun ottamatta relevanssin määrittämistä. Saanti itsessään ei ole tyypillisesti parantunut. /28/

Tilastollisten menetelmien sijaan voidaan käyttää leksikaalisia menetelmiä, eli sanastoja ja ontologioita. Nämä ovat vähemmän houkuttelevia sanastojen ja ontologioiden kehittämisen vaatimien resurssien ja ylläpitokustannuksien vuoksi, mutta sallisivat kuitenkin aineistosta riippumattoman lähestymistavan. Sanastoilla laajentaminen rajoittuu taksonomisiin suhteisiin (synonyymit ja hyperonyymit), minkä hyödyt ovat verrattain pienet ja, kuten edellä mainittiin, vaativat hyvinkin tarkkaa disambiguaatiota. /28/

Pelkkien taksonomisten suhteiden lisäksi ontologiasta voidaan löytää myös monimutkaisempia sanojen merkitysten välisiä suhteita. Hierarkkinen rakenne tarjoaa luontevan tavan näiden löytämiseksi ja tutkittavan ympäristön laajuus on helposti säädettävissä tarpeen mukaan. Tyypillisesti käsitteen yhteydessä esiintyvien toisten käsitteiden lisääminen kyselyyn parantaa tulosta merkittävästi enemmän kuin pelkkien vaihtoehtoisten kirjoitusmuotojen sisällyttäminen hakuun. /25/

Ontologisten suhteiden hyödyntäminen kyselyn laajennuksessa tuottaa parhaita tuloksia silloin kun alkuperäinen kysely on ollut lyhyt. Koska relevanssin määrittämisessä tilastolliset menetelmät ovat parhaita, on ontologisen kyselyn laajennuksen pääasiallinen tarkoitus parantaa haun saantia. Jos haku on jo alun perin ollut riittävän laajasti ja toisaalta tarkasti määritelty, ei ontologinen laajennus suo kovinkaan merkittäviä hyötyjä. Toisaalta jos käyttäjä kirjoittaa lyhyen kyselyn, on sen laajentaminen ontologisesti hyödyllistä. /28/

4.3.4 Ontologioiden hyödyntäminen hakukoneissa

Useimmat nykyiset Internet-hakukoneet ovat suoria sanahakuja, jotka saattavat sisällyttää sanan eri taivutusmuodot hakuun, mutteivät tyypillisesti muuten eroa standardista Boolen sanahausta. Semanttinen tieto ilmoitetaan yleensä sellaisessa muodossa, jossa tavanomaiset hakukoneet eivät sitä voi hyödyntää, vaan sivuuttavat tämän informaation web-sivujen rakenteellisten määrittelyjen yhteydessä. Jotta semanttisen webin visio koneymmärrettävällä metadatalla rikastetusta Internetistä voi toteutua, pitää hakukoneiden pystyä hyödyntämään tätä metadataa ja tuottamaan sen avulla aitoa lisäarvoa käyttäjille parempien hakutulosten muodossa. /42/

Nykyinen hakuprosessi on tyypillisesti seuraavanlainen: /42/

1. Käyttäjä muodostaa mielessään semanttisen kyselyn.

2. Käyttäjä muuttaa kyselyn hakutermien yhdistelmäksi, jonka uskoo esiintyvän haettua aihetta käsittelevissä dokumenteissa.
3. Hakukone palauttaa joukon dokumentteja, jotka täyttävät hakuehdot, järjestettynä jollakin tavalla termien esiintymistiheyden mukaan.
4. Käyttäjä selailee joitakin korkeimmalle relevanssille sijoitettuja dokumentteja ja pyrkii päättelemään niiden todellisen relevanssin alkuperäisen, semanttisen kyselynsä suhteen.
5. Jos käyttäjä löytää vastauksen kysymykseensä, haku oli onnistunut. Muussa tapauksessa käyttäjä palaa kohtaan kaksi ja muokkaa kyselyään.

Semanttisen webin ominaisuuksien toivotaan yksinkertaistavan tätä prosessia siten, että jo alkuperäinen, semanttinen kysely olisi validi haku, johon hakukone voisi vastata joukolla dokumentteja. Ideaalissa tapauksessa yllä esitetystä sekvenssistä jäisivät siis pois kohdat kaksi ja neljä. /42/

Nykyiset hakukoneet päättelevät dokumentin relevanssin tyypillisesti yksinkertaisten termimäärien avulla palauttaen esimerkiksi dokumentit, joissa termi esiintyy eniten, mahdollisesti painottaen myös jollakin tavalla suhteellista esiintymistiheyttä. Kyselyn laajennuksen tekniikoita, kuten sokean palautteen kyselyn laajennusta, voidaan pitää alkeellisena päättelynä, mutta aidosti käsitteiden merkityksiin nojaavat päättelyketjut voivat saavuttaa huomattavasti enemmän. Semanttisen informaation sisällyttäminen itse kyselyyn mahdollistaa laajemman päättelyketjun muodostamisen hakutulosten rajaamiseksi ja niiden relevanssin arvioimiseksi. /42/

Ontologiat ja semantiikka mahdollistavat myös päättelyn käytön hakukoneissa. Perinteiset lauseanalyysin keinot voivat selvittää termin merkityksen lauseen sisällä ja käsitteistys mahdollistaa monimutkaisen konepohjaisen päättelyn. Sanasta voidaan esimerkiksi selvittää sen rooli objektina tai subjektina tai vaihtoehtoisesti löytää myös astetta syvempiä merkityksiä, kuten onko kyse esimerkiksi paikasta vai organisaatiosta tai henkilöstä. Tätä informaatiota voidaan hyödyntää disambiguoinnissa, sekä suoraan hakujen relevanssin arvioinnissa. Vastaavia tekniikoita voidaan soveltaa myös itse hakuihin, joissa käyttäjä voisi esimerkiksi määrittää roolin, jossa toivoo hakusanan löydetyissä dokumenteissa esiintyvän. /42/

5 SEMANTIIKAN SUOMAT EDUT UUTISPALVELUILLE

5.1 Yleistä

Metadata on uutisalalla erittäin tärkeässä asemassa. Internetin suosion valtava kasvu vähensi oleellisesti uutisten välityksen monimutkaisuutta mahdollistaen kuluttajalle käytännössä rajattoman määrän uutisia päivittäin, jolloin pääasialliseksi ongelmaksi muodostui uutisten määrän sijaan niiden löydettävyys ja käytettävyys. Näihin viimeksi mainittuihin ongelmiin pyrkii metadata ja semanttinen web vastaamaan. /32/

Metadata auttaa myös toiseen uutisalalla ehdottoman tärkeään vaatimukseen eli nopeuteen. Uutiset pyritään julkaisemaan mahdollisimman nopeasti ja toisaalta kuluttaja pyrkii mahdollisimman nopeasti löytämään ja valitsemaan haluamansa uutiset suuresta tarjonnasta.

Kansallisten ja kansainvälisten uutistoimistojen tuottamat uutiset sisältävät metadataa niiden jatkokäsittelyn ja julkaisemisen helpottamiseksi. Julkaisussa uutisiin lisätään enemmän metadataa, jota kuluttaja hyödyntää etsiessään haluamiaan artikkeleja. /32/

Metadatan lisääminen vie kuitenkin aikaa uutisten tuottajan päässä ja yhtenäisen metadataskeeman opetteleminen ja noudattaminen vaatii toimittajan työpanosta. Näitä ongelmia helpottamaan on kehitetty monia automaattisia työkaluja, joilla pyritään generoimaan metadataa joko täysin automaattisesti tai interaktiivisesti toimittajien valitessa automaattisesti generoiduista ehdotuksista oikeat. /32/

5.2 Tehokkaammat haut

Ontologioihin perustuvat, käsitepohjaiset haut sisältävät automaattisesti hakutermien vaihtoehtoiset kirjoitusmuodot, vanhentuneet nimet ja synonyymit, jos ne vain ontologiaan kirjataan. Tällöin tulosdokumenttien joukkoon saadaan automaattisesti sisällytettyä nekin dokumentit, joissa itse hakusana ei esiinny, mutta joista sen synonyymi löytyy.

Ontologiat tarjoavat myös helposti laajennettavan tavan tehostaa hakuja halutun osa-alueen suhteen. Lisäämällä hakujärjestelmän indeksointiin ontologiakohtaisia indeksejä, voidaan palveluja personoida tiettyjä käyttäjäryhmiä varten esittelemällä spesifeihin sanastoihin perustuvia ontologioita, jotka mahdollistavat tehokkaammat haut haluttujen aihealueiden sisällä.

Esimerkkinä tällaisesta voitaisiin ajatella vaikkapa biologiaontologian avulla tehty indeksiä, jota voitaisiin hyödyntää esimerkiksi etsimällä artikkeleita merikotkista. Haku suoritetaan biologiaontologiaindeksiin, jossa merikotka ja *Haliaeetus albicilla* (merikotkan tieteellinen nimi) viittaavat molemmat samaan käsitteeseen ja molempia muotoja sisältävät dokumentit sisältyvät hakuun automaattisesti. Dokumenttien painotus perustuu TF-IDF:ään (ks. 4.1.3) ja koska molemmat termit nostavat käsitteen painoa, lasketaan tieteellisten nimien esiintyminen relevanssipainotukseen, jolloin nämä dokumentit nousevat listan kärkeen. Kyseinen ontologia voi myös sisältää tiedon, että merikotka on petolintu, jolloin tämänkin käsitteen esiintyminen tekstissä voi vaikuttaa dokumentin painoon.

Koska painotusskeemoista on mahdollista (ja kannattaakin) tehdä ontologiakohtaisia, voidaan painotus räätälöidä hyvinkin tehokkaasti aina kutakin aihealuetta parhaiten tukevalla tavalla. Esimerkiksi riippuen ontologian yleisestä syvyydestä voidaan alaluokkien painotusta säätää käsittämään alaluokat syvemmältä tai vähemmän syvältä kullekin ontologialle sopivalla tavalla.

Tekstin käsitteistyksellä voidaan suorittaa myös disambiguaatiota, joka tehostaa hakuja sellaisten termien osalta, jotka vastaavat kahteen käsitteeseen. Hyvä esimerkki tapauksesta, jossa käyttäjä voi haluta eron haun aikana tehdä, on ”Nokia”, joka voi viitata yritykseen tai paikkakuntaan. Käyttäjää voidaan pyytää tarkentamaan hakuterminsä avulla luotua hakukäsitettä tarjoamalla yksinkertainen lista käsitteistä, jotka vastaavat hakutermiä. Luonnollisesti haku voidaan suorittaa myös ilman disambiguointia, jolloin kaikki hakutermiä vastaavat käsitteet sisällytetään hakuun.

Aiheen lisäksi uutisartikkelit määritellään myös ajan ja paikan suhteen, jotka luovat näin kaksi luonnollista akselia, joiden mukaan uutisartikkeleita kannattaa intuitiivisesti jäsentää. Aika-akselille ei tarvita ontologiaa, vaan riittää, että hakija pystyy rajaamaan haun käsittämään tietyn ajanjakson. Lisäksi uutisaineiston ollessa kyseessä tulisi hakutulokset olla mahdollista järjestää niin relevanssin kuin myös ajan suhteen, koska käyttäjä on usein kiinnostunein kaikkein tuoreimmista artikkeleista.

Paikka-akselin hyödyntämiseen ontologiat käyvät mitä parhaimmin. Paikkaontologia on mahdollista integroida järjestelmään ja suorittaa indeksointia sen mukaan aivan vastaavasti kuin aihekohtaisilla ontologioilla. Paikkaontologiassa paikkojen hierarkia on helppo esittää todellisuutta vastaavassa muodossa, jossa suuremmat paikat muodostuvat pienemmistä, jotka muodostuvat edelleen pienemmistä (vrt. valtiot - läänit - kunnat).

Jos paikoilla on ontologiassa myös koordinaatit, voidaan ne sijoittaa helposti kartalle, joka tuo aivan uusia mahdollisuuksia käyttöliittymän kannalta. Hakutuloksia esitettäessä ne voidaan sijoittaa eri puolille karttaa tapahtumapaikkansa mukaisiin koordinaatteihin. Tämä voisi olla käyttäjän kannalta hyvinkin mielenkiintoista ja sallii nopean tavan esittää paikat, joissa on tapahtunut kyseiseen hakuun liittyviä uutisia.

Käyttäjälle voidaan myös tarjota mahdollisuus rajata kartalta jokin alue haun yhteydessä, jolloin haettaisiin vain artikkeleita, jotka koskevat alueen sisällä olevia paikkoja. Tällä tavalla toimittaessa voidaan helposti sisällyttää hakuun useita saman mittakaavan alueita sekä näiden sisällä olevat alueet.

Disambiguointi paikkojen välillä voi olla erittäinkin tärkeää. Esimerkiksi Amsterdam-nimisiä kaupunkeja löytyy Alankomaiden lisäksi Kanadasta ja USA:n Missourista, Ohiosta ja New Yorkista. Etenkin haluttaessa etsiä muihin kuin Alankomaiden Amsterdamiin liittyviä artikkeleja voi hakutuloksiin tulla valtava määrä ei-haluttuja dokumentteja, jos kaupunkeja ei ole mahdollista disambiguoida ja yksilöidä. Koska artikkeleissa tyypillisesti mainitaan kaupungin yhteydessä lisäksi sen sijaintivaltio, on disambiguointi verrattain yksinkertaista ontologian kertoessa, missä valtiossa mikin disambiguoitavista kaupungeista sijaitsee.

Paikkaontologiaan liittyy myös joitakin haasteita. Ensinnäkin paikkojen tarkkojen maantieteellisten rajojen määrittäminen eksaktisti on erittäin hankalaa. Miljoonien paikkojen paikkaontologia, jossa jokaisen paikan rajat on määritetty tarkasti, on nykyisellään käytännössä mahdoton. Tämän vuoksi pitää paikkojen koordinaattiesitykseen ottaa käyttöön jonkinlainen kompromissi. Yksinkertaisin ratkaisu on vain merkitä alueen keskipiste, mutta tällöin esimerkiksi kartalta rajaaminen muuttuu hankalammaksi. Ongelmaksi muodostuu laajojen alueiden keskipisteen valinta, eli jos paikkaontologian käsite `Suomi` on koordinaateiltaan yksittäinen piste keskellä Suomea, sisältyy käsite kaikkiin rajauksiin, jotka sisältävät kyseisen pisteen, olivat ne sitten kuinka pieniä tahansa. Tätä voitaisiin pyrkiä ratkaisemaan esimerkiksi määrittelemällä alue yhden sijasta kolmella koordinaatilla, jotka sijaitsevat eri puolilla aluetta, jolloin kaikkien pitäisi sisältyä rajatulle alueelle, jotta paikka valittaisiin. Toinen mahdollisuus olisi tehdä alueesta ympyrä, jonka keskipiste on alueen keskipiste ja jonka säde on minimimatka keskipisteestä alueen lähimmälle reunalle. Näissäkin molem-

missa tavoissa on toki omat ongelmansa ja lisäksi toteutukseen vaadittavan datan saaminen voi olla hankalaa joidenkin alueiden osalta.

Toinen pohdittava asia on artikkelit, jotka eivät varsinaisesti sijoitu minnekään tai joiden sijainti on epämääräinen. Esimerkiksi yritysostosta kertova artikkeli ei välttämättä sijoitu minnekään, mutta toisaalta sen voidaan ajatella sijoittuvan esimerkiksi paikkakunnalle, jossa yrityksen pääkonttori sijaitsee. Toisaalta suomalaisten yritysten voidaan myös ajatella sijoittuvan Suomeen, vaikka tätä ei eksplisiittisesti artikkelissa määriteltäisikään. Usein voidaan olettaa, että käyttäjä valitsee kartalta alueen vain siinä tapauksessa, että on kiinnostunut paikallisista uutisista, mutta alueen ollessa riittävän suuri, esimerkiksi kun liikutaan valtiotasolla, ei tämä olettamus enää välttämättä pidä paikkaansa. Käyttäjää pitää siis jollain tapaa ohjeistaa järjestelmän tästä osasta, koska yleispätevää ratkaisua tuskin voidaan kehittää.

5.3 Muiden relevanttien artikkelien linkitys

Ontologinen indeksointi mahdollistaa helpon linkityksen artikkelien välillä perustuen samojen käsitteiden ilmenemiseen kulloinkin sekä käyttäjän lukemassa dokumentissa, että siihen linkitetyissä artikkeleissa. Eri ontologiaindeksit mahdollistavat erilaisten näkökulmien tarjoamisen aineistoon linkittämällä artikkelit eri ontologioiden käsitteiden perusteella. Tällä perusteella voitaisiin tarjota käyttäjälle esimerkiksi samaan paikkaan liittyviä uutisia (paikkaontologia), samoja henkilöitä koskevia uutisia (toimijaontologia) tai aiheeltaan vastaavia muita uutisia (tapahtumaontologia).

Suositteluja voidaan tehdä useallakin tavalla. Yksi mahdollisuus olisi hakea termejä, joiden IDF-arvo on korkea ja käyttää niitä luokittelukäsitteinä tietylle artikkelille. Näitä luokittelukäsitteitä hyödyntämällä käyttäjä voi suorittaa hakuja muihin samoja luokittelukäsitteitä sisältäviin artikkeleihin. Tätä tarkoitusta varten luokittelukäsitteet voitaisiin esittää artikkelin yhteydessä yksinkertaisina linkkeinä hakuun kyseisellä käsitteellä. Linkit voisivat olla artikkelitekstin osana, jolloin käsitettä vastaavat sanat muutettaisiin linkeiksi käsitettä vastaaviin, dynaamisiin hakuihin, tai erillisenä listana artikkelin yhteydessä.

Moninäkymähaku on myös eräs ontologioiden tehokkaasti avaama vaihtoehto. Moninäkymähaussa aineistosta muodostetaan joukko semanttisia, ortogonaalisia hierarkioita, kuten paikka ja aika, ja aineisto esitetään näiden näkymien tai fasettien (engl. facet) avulla järjestettynä. Käyttäjän valitessa jonkin tietyn fasetin, päivitetään näkymä vastaamaan valittua fasettia. Jos valitaan useita fasetteja, näkymä muodostetaan näiden leikkauksena ja jokaisen valinnan jälkeen näkymä muodostetaan uudelleen. Näin voidaan tehokkaasti rajata askel kerrallaan näkymää tarkemmin käsittämään haluttu osa tarkasteltavana olevasta järjestelmästä. Jos jokin valinta johtaisi tyhjään joukkoon, sitä ei sallita valittavan. /24/

Moninäkymähaku on otollinen tiedonhakutapa etenkin silloin, kun käyttäjä ei tiedä tarkalleen hakemaansa, vaan selailee dokumenttitietokantaa etsien jotakin mielenkiintoista. Tämä on sinällään erittäin yleinen käyttötapaus uutisaineistoon ja usein uutiset jaotellaankin aihepiirien mukaan moninäkymähaun periaatteiden mukaisesti esimerkiksi kotimaan uutisiin, urheiluun ja kulttuuriin. Ontologisella lähestymistavalla saavutetaan kuitenkin joustavuutta ja automaatiomahdollisuuksia.

5.4 Ilmoitusten aihekohtainen linkitys

Artikkelien ontologisen jäsennyksen tuottaman semanttisen informaation avulla olisi mahdollista liittää artikkeleihin niiden aiheisiin liittyviä ilmoituksia. Voidaan olettaa, että ihminen, joka lukee artikkelia jostakin aiheesta, on siitä kiinnostunut ja edustaa näin ollen kyseiseen aiheeseen liittyvien tuotteiden ja palvelujen kohderyhmää. Tällainen kohdentamismahdollisuus on luonnollisesti erittäin mielenkiintoinen mainostajien näkökannalta.

Merkittävin ongelma ilmoitusten automaattisessa linkittämisessä on negatiivisten uutisten aiheuttamat hankaluudet. Tässä negatiivisiksi uutisiksi mielletään kaikki uutistapahtumat, joihin liittyy jonkinlaista inhimillistä kärsimystä tai materiaalisia vahinkoja ja menetyksiä. Esimerkiksi tuhopolttouutinen voisi löydettyjen käsitteiden puolesta helposti linkittyä tulitikkumainokseen, mikä ei varmasti ole mainostajan tai uutisten tarjoajan intressien mukaista. Yleistäen voidaan ajatella, että mitkään negatiiviset uutiset eivät tyypillisesti sovellu aiheeseen liittyvän mainonnan yhteyteen.

Helpoin tapa ongelman ratkaisemiseksi olisi jollain tavalla selvittää uutisen sävy ja linkittää ilmoituksia vain positiivisiin uutisiin. Tämä voitaisiin tehdä yksinkertaisella listalla negatiivisista käsitteistä (katastrofit, rikokset, onnettomuudet jne.) tai käyttämällä jonkinlaista tapahtumaontologiaa aineiston indeksoinnissa, jossa tapahtumatyyppien sävy on eksplisiittisesti määritelty (ks. 6.6). Periaatteessa positiivisten uutisten merkitseminen negatiivisiksi ja näin ollen aiheriippuvaisista mainoksista vapaiksi ei aiheuttaisi suurtakaan haittaa, kunhan se ei olisi liian yleistä. Järjestelmän tarkkuustason vaatimukset eivät siis olisi erityisen tiukkoja.

Negatiivisten uutisten lisäksi ongelmia automaattiselle linkitykselle tarjoavat tuotteisiin ja palveluihin itseensä liittyvät uutiset. Esimerkiksi tuotevertailuartikkelin vieressä oleva mainos vertailussa huonosti menestyneestä tuotteesta voi olla mainoksen ostaneen tahon mielestä huonosti sijoitettu. Toinen esimerkki vastaavasta ongelmasta olisi yritystä koskevan artikkelin yhteydessä esiintyvä kilpailijan mainos. Nämä ongelmat eivät ole yhtä suoraviivaisia kuin negatiivisten uutisten yhteydessä esitettävien ilmoitusten problemaattisuus, mutta saattavat väärissä yhteyksissä aiheuttaa epätoivottavia reaktioita niin kuluttajien kuin myös mainostavien yritysten keskuudessa.

5.5 Ulkopuolisten aineistojen linkitys

Automaattinen annotointi ja käsitteiden tunnistaminen tekstistä mahdollistavat tehokkaan automaattisen linkityksen myös arkiston ulkopuolisiin aineistoihin. Valittaessa tietty artikkeli, voidaan tutkia sen sisältämät käsitteet ontologiakohtaisesti ja suorittaa haku näillä ontologioilla annotoituihin toisiin aineistoihin.

Erityisen mielenkiintoinen linkityskohde ovat kansalliset portaalit, kuten TerveSuomi.fi /65/ tai KulttuuriSampo.fi /66/. Näihin portaaleihin on kerätty puolueetonta tietoa laajasti jonkin tietyn aihealueen tai teeman ympäriltä ja niihin olisi mahdollista suorittaa valikoitua linkittämistä tiettyjen osa-alueiden artikkeleista. Esimerkiksi Henkilökohtaista-teeman artikkelien linkittäminen Tervesuomeen palvelisi tehokkaasti lukijaa, jolle vaikkapa painonhallintaa käsittelevää artikkelia lukiessa herää toive lisäinformaatioon. Vastaavasti kulttuu-

riosan artikkelit voisi hyvinkin linkittää Kulttuurisampoon, jolloin esimerkiksi taidenäyttelystä kertova artikkeli linkittyisi automaattisesti muuhun näyttelyn taiteilijoista tai teoksista olemassa olevaan tietoon.

Wikipediaa voitaisiin myös pitää mielekkäänä linkittämisen kohteena lähes kaikkiin artikkeleihin. Artikkelista voidaan tunnistaa paikkoja ja nimiä, sekä koko aineiston suhteen harvinaisia termejä, joiden perusteella voidaan suorittaa hakuja Wikipediaan ja tulokset linkittää automaattisesti artikkelisivulle. Linkityksessä voitaisiin myös painottaa tiettyjä artikkelin osia, kuten otsikkoa ja kuvatekstejä, jotta voitaisiin varmistaa linkityksien oleellisuus.

Ongelmalliseksi Wikipedian tekee sen luonne vapaasti editoitavana tietosanakirjana, jolloin sen sisältämä tieto ei aina ole täysin luotettavaa. Internetin peruskäyttäjän ei voida olettaa olevan riittävän perehtynyt, että hän voisi varmuudella tehdä perusteltuja oletamuksia Wikipedia-artikkelin oikeellisuudesta. Tulevaisuudessa voi kuitenkin hyvinkin olla, että Wikipedian ja vastaavien verkkotietosanakirjaprojektien suosion kasvaessa ihmiset ymmärtävät ja tiedostavat konseptin heikkoudet ja vahvuudet ja kyseiseen aineistoon linkittäminen käy yksinkertaisemmaksi.

On myös ehdotettu, että Wikipedian yksittäisten artikkelien luotettavuutta voitaisiin automaattisesti arvioida. Kirjoittajia voitaisiin luokitella luotettaviksi joidenkin kriteerien mukaan, kuten yleisen aktiivisuuden. /20/ Näin olisi mahdollista tarkistaa linkitettävän artikkelin todennäköinen luotettavuus ja joko informoida käyttäjää artikkelin luotettavuudesta tai yksinkertaisesti linkittää ainoastaan jonkin ennalta määritellyn luottamuskynnysarvon ylittäviin artikkeleihin.

Ontologioita peilaamalla (ks. 2.2.2) mahdollistuisi myös linkitys esimerkiksi vieraskielisiin aineistoihin, jos nämäkin on annotoitu ontologiapohjaisesti. Tapahtumaontologialla (ks. 6.6) linkitys saataisiin kaikkein tehokkaimmaksi, mutta käytännössä tällaisen järjestelmän toimiva peilaaminen vaatisi laajoja yhteistyösopimuksia muiden mediayhtiöiden kanssa. Käsiteontologiat peilaamalla linkitys olisi helpompaa ja voisi kuitenkin toimia verrattain hyvin. Kun samat käsitteet ja paikat esiintyvät samaan aikaan julkaistuissa artikkeleissa eri uutislähteiden tietokannoissa, voidaan varsin suurella todennäköisyydellä olettaa, että kyse on samasta tapauksesta tai ainakin alkuperäiseen artikkeliin kiinnostavasti liittyvästä uutisesta.

Keskeisin vaatimus ulkopuolisiin aineistoihin linkitettäessä lienee ontologioiden väliset peilaukset. Yleistä suomalaista ontologiaa käytettäessä tämä ongelma ratkeaa itsestään, jos linkitettävät tahot ovat myös ottaneet kyseisen ontologian käyttöönsä tai ainakin jonkin siihen liittyvän alaontologian, jonka hierarkkiset suhteet YSO:on on eksplisiittisesti määritelty. Toinen helppo vaihtoehto on luoda artikkeleista uusi indeksi sillä ontologialla, jota linkitettävä sivusto käyttää. Viimeinen tapa on ontologioiden peilaus semanttisen webin vision (ks. 3.2.2) mukaisesti, jolloin kaksi ontologiaa saadaan yhteensopiviksi.

5.6 Tapahtumaontologia

Tapahtumat ovat uutisaineiston ytimessä ja artikkeleista suurin osa käsitteleekin jonkinlaista tapahtumaa tai sellaisen jälkiseuraamuksia. Tämän vuoksi luonnollinen tapa organisoida

uutisaineistoa sisällöllisistä lähtökohdista olisi juurikin tapahtumiin perustuva luokittelu ja lähestyttäessä ongelmaa semanttisen webin lähtökohdista on tapahtumaontologia luonnollinen vastaus.

Ontologisessa lähestymistavassa tapahtumat muodostaisivat luokkia, jotka järjestettäisiin hierarkiaksi ala- ja yläluokkasuhteilla. Näistä luokista voidaan muodostaa instansseja, jotka vastaavat tiettyjä spesifejä tapahtumia, joille on lisäksi määritelty aika ja paikka. Esimerkiksi urheilutapahtumat-luokalla voisi olla alaluokka olympialaiset, jolla voi olla instanssina vuoden 1972 Münchenin olympialaiset.

Tapahtumat instantioimalla on mahdollista yksilöidä ne ja viitata niihin uudestaan, kun samaa tapahtumaa käsitellään toisessa uutisessa. Eri organisaatioiden tapahtumaontologioiden instanssien peilaaminen toisiinsa mahdollistaisi erittäin tehokkaan linkityksen eri tahojen materiaalien välillä.

Merkittävin ongelma tapahtumaontologian toteuttamisessa on sen luomiseen ja ylläpitoon vaadittu työpanos. Lisäksi täyden hyödyn irtisaamiseksi vaadittaisiin useiden uutistuottajatahojen yhteistyötä, jonka järjestäminen on merkittävä haaste. Toisaalta toteutuessaan sellainen järjestelmä vähentäisi yksittäisen toimituksen kuormaa, mutta järjestelmän tehokas organisoiminen olisi valtava haaste. Esimerkiksi se, että samasta tapahtumasta ei synny useita instansseja on varmistettava, mikä vaatii paljon hakujärjestelmältä ja sen käyttäjiltä.

5.7 IPTC

5.7.1 Yleistä

International Press Telecommunications Council, eli IPTC, on uutisten vaihtoon käytettävien teknisten standardien kehitykseen ja ylläpitoon erikoistunut konsortio, johon kuuluu 55 uutismaailman toimijaa, kuten uutistoimistoja ja uutisten julkaisukanavia. IPTC perustettiin vuonna 1965 ja siihen kuuluu jäseniä Euroopasta, Aasiasta, Australiasta ja Pohjois-Amerikasta. /31/ IPTC:n luomat standardit ovat hyvä esimerkki hyvin määritellyn metatiedon mahdollisuuksista toimituksellisten ongelmien ratkaisemisessa. Niitä voidaan pitää edellä esitettyjen semanttisen webin tekniikoihin nojaavien ratkaisujen esiversiona ja ne esittelevät kansainvälisen yhteistyön luomia mahdollisuuksia.

IPTC:n pyrkimyksenä on toimia poliittisesti vapaana, kansainvälisenä foorumina, joka tukee ja mahdollistaa uutisten tehokasta, korkealaatuista vaihtoa hyödyntäen telekommunikaatiossa ja tietotekniikassa tapahtunutta kehitystä. Tietokoneiden ja tietoliikenteen noustua yhä keskeisempään osaan uutisvälityksessä on IPTC keskittynyt kehittämään sovelluksia ja standardeja uutisten nopeaa, digitaalista siirtoa varten. Viime aikoina suurin mielenkiinto on kohdistunut WWW:hen ja multimedia- sekä verkkouutisten julkaisuun liittyvään problematiikkaan. Muun muassa näitä ongelmia ratkaisemaan IPTC on kehitellyt joukon luokittelujärjestelmiä, joilla voidaan konsistentisti määrittää uutisaineistoon liittyviä ominaisuuksia alkuperäiskielestä riippumatta. /31/

Tällä hetkellä IPTC suosittelee viittä eri standardia: /31/

- NewsML 1 on julkaisutavasta riippumaton XML-pohjainen standardi multimediauutisten paketoimiseen, välittämiseen ja hallinnointiin niiden koko elinkaaren aikana.
- NITF on XML-pohjainen, tekstuaalisten uutisaineistojen merkitsemiskieli.
- SportsML on urheiluun liittyvän aineiston, kuten tulosten, aikataulujen ja urheilusijoitusten käsittelyyn ja välittämiseen keskittynyt XML-pohjainen standardi.
- IPTC Core on pääasiallisesti valokuville tarkoitettu joukko metadatamäärittelyitä, jotka pyrkivät olemaan yhteensopivia Adoben Extensible Metadata Platformin (XMP) kanssa.
- NewsCodes on uutisalalla laajalti käytössä oleva joukko sanastoja, joihin kuuluu laaja aihetaksonomia, tuki uutisalueiden ja tapahtumapaikkojen määrittämiselle, sekä luokitukset kiireellisyydelle, tärkeydelle ja relevanssille.

Näiden valmiiden standardien lisäksi IPTC kehittelee G2-standardiperhettä, joka tarjoaa yhtenäisellä tyylillä toteutetun joukon yleisiä spesifikaatioita ja komponentteja: /31/

- NewsML-G2 on tarkoitettu kääreeksi uutisille, jotka koostuvat tekstistä, valokuvista, grafiikasta, videosta tai muusta mediasta. Voidaan käyttää näiden vapaavalintaisten yhdistelmien paketoimiseen.
- EventsML-G2 on tiedonvaihtostandardi uutistoimitukselle oleellisen tapahtumainformaation välittämiseksi, joka sisältää standardit tapahtumien julkaisulle, suunnittelulle ja uutisoimiselle.
- SportsML-G2 on uusi versio SportsML:stä, joka on suunniteltu yhteensopivaksi muiden G2-standardien kanssa.
- ProgramGuideML on erikoistunut formaatti TV- ja radio-ohjelmatietojen esittämistä varten.

5.7.2 NewsML

NewsML on tarkoitettu mediariippumattomaksi, rakenteelliseksi kehykseksi multimediauutisille. Tämänhetkinen versio on järjestysnumeroltaan 1.2, mutta NewsML 2 on jo kehitteillä osaksi G2-standardiperhettä. NewsML on tarkoitettu käytettäväksi koko uutisen elinkaaren ajan selittämään uutisen osien keskinäisiä suhteita, kun uutista siirretään käsiteltäväksi eri järjestelmille. Tyypillisiä vaiheita, joissa uutista käsittelevä järjestelmä vaihtuu, ovat esimerkiksi toimitusjärjestelmästä toiseen siirtyminen, uutisen siirtyminen uutistoimistolta sen asiakkaille tai uutisten julkaisijan ja uutisia keräävän web-palvelun välinen kanssakäynti. /34/

NewsML perustuu uutisobjektiin, joka voi sisältää tekstiä, valokuvia, videota tms. ja näihin liittyvän metadatan, joka kertoo komponenttien keskinäiset suhteet ja jokaisen osan roolin uutisobjektissa. Ideana on, että kaikki mitä vastaanottajan pitää tietää uutisobjektista, sisältyy NewsML-rakenteeseen. NewsML tarjoaa esimerkiksi mahdollisuuden julkaista sama teksti eri kielillä, sama video eri formaateissa tai sama valokuva eri tarkkuuksilla – kaikki saman uutisobjektin osana. Metadataa tulkiten voi vastaanottaja, joka siis voi olla ihminen tai sovellus, hyödyntää helposti vain tarvitsemansa osat. Rakenteeseen on myös mahdollista sisällyttää versiointi-informaatiota, tilatietoa (esimerkiksi siitä onko uutinen julkaistu) sekä hallinnollista informaatiota esimerkiksi tekijänoikeuksista. /34/

NewsML sisältää metadatasanastoja, mutta ei sinällään pakota mihinkään tiettyyn formaattiin, vaan sallii eri sanastot jopa saman uutisobjektin sisällä. Uutisobjektin tekstille IPTC suosittelee omaa NITF-sanastoaan. NewsML on helposti laajennettavissa ja käyttää standardeja nimeämiskonventioita, joten uutisobjektin sisältö voi koostua linkeistä itse tiedostoihin. Tällöin järjestelmän mahdollistamia vaihtoehtoisia sisältöjä hyödyntävien tahojen ei tarvitse käsitellä näitä kaikkia sisältöjä, vaan vain tarvitsemiaan. /34/

5.7.3 NITF

News Industry Text Format (NITF) on itsenäinen, tekstipohjaiseen uutistenvaihtoon tarkoitettu metadataformaatti. Se on maailman eniten käytetty uutisalan XML-sanasto ja ensimmäinen IPTC:n kehittämä XML-pohjainen standardi. /31/ NITF:n kehittäminen aloitettiin 1990-luvun alkupuolella korvaamaan ANPA 1312 ja IPTC 7901-standardit vuodelta 1979. Alun perin NITF perustui XML:n edeltäjään, SGML:ään (Standard Generalized Markup Language), mutta vuonna 1998 NITF:stä tehtiin XML-pohjainen. Uusin versio on järjestysnumeroltaan 3.4 ja kehitystyö jatkuu edelleen. /35/

NITF-formaatilla toteutettu artikkeli on jaettu kahteen osaan, otsakkeeseen (<head>) ja runkoon (<body>). Otsake sisältää uutisen otsikon lisäksi metadataa esimerkiksi julkaisuajankohdasta, kirjoittajasta, artikkelin aiheesta ja kiireellisyydestä. Rungossa sijaitsee itse artikkelin teksti, josta jotkut itsellisesti merkittävät käsitteet on eroteltu ohjainkoodeilla (engl. tag). Esimerkkejä tällaisista ovat henkilöt, paikat ja hyperlinkit. Vastaavasti ohjainkoodeilla on tekstin lomaan merkitty taitolle oleellinen informaatio, kuten väliotsikot, taulukot ja kuvat. /31/

Suuri osa otsakkeessa sijaitsevasta metatiedosta sisältyy jo suunniteltuun NewsML-G2-standardiin ja jos NITF:ää käytetään edellisen sisällä tekstiaineistona, voi kahteen kertaan määritelty metatieto aiheuttaa hämmennystä tai ristiriitoja. Toisaalta joidenkin yksinkertaisten sovellusten ollessa kyseessä ei laaja metadata ole välttämättä tarpeellista. Näitä ongelmia ratkaisemaan on NITF:stä suunnitteilla kaksi versiota, ”power” ja ”core”. Näistä ”power” tarjoaisi saman funktionaalisuuden kuin nykyinenkin versio, mutta ”core” olisi riisutumpi versio. Koko dokumenttia koskevaan metadataa olisi karsittu toisteisuuden eliminoimiseksi, mutta rikastettu tekstiformaatti olisi jätetty entiselleen. /31/

5.7.4 NewsCodes

IPTC NewsCodes on joukko metadatamäärittelyjä, joilla erilaisiin uutisaineistoihin kuten tekstiin, kuviin ja videoon voidaan lisätä aihemäärittelyjä. Tämä mahdollistaa konsistentin luokittelukäytännön uutisaineistoihin liittyen, mikä helpottaa niiden jatkokäsittelyä ja myöhempää hyödyntämistä. /33/

NewsCodes on jaettu kolmeenkymmeneen erilliseen joukkoon niiden käyttötarkoituksen mukaan. Nämä luokat on puolestaan kerätty neljään ryhmään, jotka antavat hyvän yleiskuvan NewsCodesin käyttömahdollisuuksista: /33/

- Descriptive NewsCodes käsittää neljä taksonomiaa, joilla voidaan kuvailla uutisaineiston aihetta ja sisältöä. Genre määrittää uutisaineiston luonnetta ja journalistisia piirteitä, Scene kuvailee uutisobjektin (tässä tapauksessa tyypillisimmin ku-

va) kuvauksen esimerkiksi rajauksen ja etäisyyden suhteen, Subject Code on n. 1300 termiä sisältävä kolmiasteinen luokittelu uutisaineiston sisällölle ja Subject Qualifier sisältää kapeita määrittelyjä uutisobjektin kohteesta. Subject NewsCodes on tällä hetkellä kehityksessä ja siitä ollaan tekemässä uutta versiota /31/.

- Administrative NewsCodes sisältää yhdeksän taksonomiaa, joista kaksi vanhentuneita, uutisobjektien hallinnointiin. Esimerkkeinä mainittakoon väriavaruuden määrittämiselle ja uutisobjektin kiireellisyydelle tarkoitetut sanastot.
- Transmission NewsCodes on vain yksi taksonomia, Priority, joka määrittää uutisobjektin suhteellisen tärkeyden jakelua varten yksinkertaisella yhdeksänportaisella asteikolla.
- Exchange Format NewsCodes käsittää 16 taksonomiaa, jotka sisältävät metadatamäärittelyjä, joiden on tarkoitus tukea muita IPTC uutistenvaihtostandardeja.

IPTC NewsCodes-ryhmiä päivitetään säännöllisesti ja IPTC:n jäsenet voivat ehdottaa uusia kenttiä tai määrittelyjä. Osa taksonomioista on käännetty useille kielille. /33/

NewsCodes-taksonomiat on suunniteltu käytettäviksi muiden IPTC-standardien kanssa, mutta niitä voi soveltaa muissakin XML-skeemoissa. Käytännössä nämä koodit muodostuvat formaalista nimestä (FormalName), aihetyypistä (TopicType) ja nimestä (Name). Esimerkiksi Descriptive NewsCodesin Subject NewsCodes sisältää koodin aiheelle tanssi, jonka formaali nimi on 01006000, ihmisluettava nimi on dance ja aihetyyppi on SubjectMatter /43/. Lisäksi suurimmasta osasta koodeja on olemassa lyhyt kuvaus. Kun koodit käännetään toisille kielille, jätetään formaali nimi ennalleen, joka toimii näin tunnisteena käännösten välillä. /37/

5.7.5 EventsML

EventsML, tai EventsML-G2 on IPTC:n G2-perheen standardi uutisarvoisten tapahtumien julkaisemiseen, juttujen suunnitteluun ja uutisointioikeuksien järjestämiseen /36/. Se ei sinällään ole tapahtumaontologia samassa mielessä, kuin edellä käsitelty (ks. 6.6). EventsML on merkintäkieli, joka keskittyy tapahtumien uutisointiprosessiin, ei niinkään tapahtumien luokitteluun ja niiden ontologiseen instantiointiin.

EventsML:ää ei ole vielä julkaistu, mutta sitä on kehitetty vuodesta 2003 ja sen toiminta ja tarkoitus on dokumentoitu verrattain perusteellisesti. EventsML on lähtökohtaisesti yhteensopiva muiden IPTC-standardien kanssa ja siitä pyritään tekemään helposti laajennettava ja päivitettävä. Lisäksi sen varsinainen ydin pyritään pitämään yksinkertaisena, johon on mahdollista lisätä tarpeen mukaan monimutkaisempia ominaisuuksia ja määrittelyjä. Lisäksi standardin on määrä olla käytettävissä ja hyödyllinen myös IPTC:n ulkopuolisille tahoille. /36/

Suunnitellut osa-alueet, joilla EventsML toimii, ovat tapahtumainformaatio (Event information), uutisointi-informaatio (Coverage information) ja yleinen metainformaatio (Meta information). Tapahtumainformaatio vastaa kysymyksiin kuka, mitä, missä ja milloin ja mahdollistaa tapahtuman liittämisen muihin artikkeleihin. Tapahtumaan liittyy aina yksi tai useampia tunnisteita sekä mahdollisesti aika- ja muutostietoja. Liittäminen toisiin tapahtumiin voi tapahtua kronologisesti ajan suhteen, rinnakkaistapahtumiin jossain tapahtumako-

koelmassa tai pää- ja alitapahtumiin jonkin laajemman kokonaisuuden ollessa kyseessä. Tapahtuman voi liittää myös valmiisiin tai suunniteltuihin uutisobjekteihin. Tapahtumainformaatioon kuuluu myös metadataa aiheesta, avainsanoista, tapahtumaan ilmoittautumisesta jne. /36/

Uutisointi-informaatio käsittelee tapahtuman uutisointia ja sitä voidaan käyttää organisaation sisällä jakamaan tehtäviä tai eri uutisorganisaatioiden välillä ilmoittamaan suunnitellusta uutisoinnista. Tapahtuma voidaan jakaa yksittäisiin tehtäviin ja näistä voidaan kirjata ylös tyypillinen metadata koskien tehtävän aikataulua, suorittajaa jne. Lopuksi nämä tehtävät voidaan sitoa tapahtumainformaation lailla olemassa oleviin tai suunniteltuihin uutisobjekteihin. /36/

Metadatainformaatio puolestaan käsittelee tapahtumaan liittyvien tahojen metadataa, jota on mahdollista käyttää uudelleen tarpeen vaatiessa. Tapahtumaan liittyvät osapuolet, joiden tulee olla henkilöitä tai organisaatioita, voidaan kuvailla tarpeellisilta osin metadatainformaatio-osuudessa. Samoin tapahtuman paikka ja osapuolten yhteystiedot kuuluvat tähän osaan EventsML:ää. /36/

6 DOKUMENTIN LAAJENNUS KÄSITEKLUSTEROINNILLA

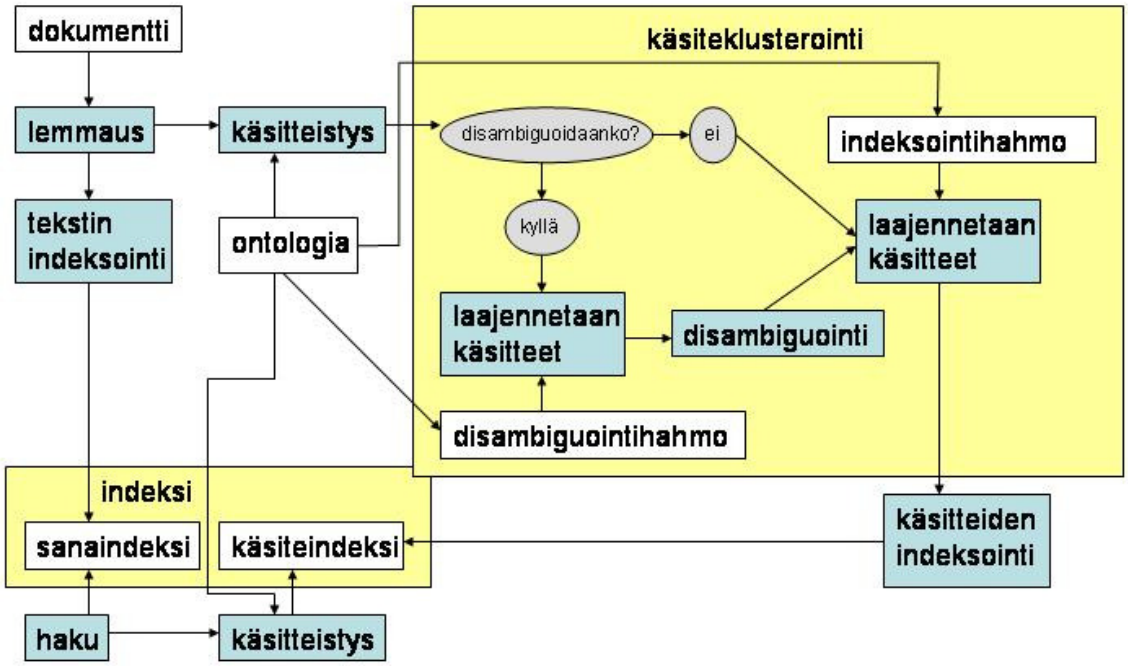
6.1 Yleistä

Tutkimusongelman mukaisesti projektin tarkoituksena oli kehittää uuden tyyppinen hakujärjestelmä tekstimuotoiseen aineistoon semantiikkaa ja dokumentin laajennusta hyödyntäen. Käytännössä tämä tarkoittaa tekstissä esiintyvien termien muuttamista käsitteiksi ja näiden lisäämistä käsiteltäviin dokumentteihin tarkoituksena parantaa dokumenttien vastaamista suoritettaviin hakuihin.

6.2 Käsiteklusterointi ja dokumentin laajennus

6.2.1 Menetelmä

Kuvassa 15 on esitetty käsiteklusteroinnin avulla toimivan dokumentin laajennuksen ja indeksoinnin kokonaisprosessi dokumentista indeksiksi, jossa menetelmän ominaiset osat on eroteltu keltaisiin laatikoihin..



Kuva 15. *Dokumentin laajennus ja indeksointi käsiteklusteroinnilla*

Kuvan 15 prosessi alkaa dokumentista, joka lemmataan. Tämän perusteella suoritetaan tekstin indeksointi sanaindeksiin perinteisen indeksoinnin mukaisesti. Tämän lisäksi tekstistä saatuja lemmoja verrataan käytettävään ontologiaan ja etsitään näin tekstissä esiintyvät käsitteet. Jos käsiteklusteroinnissa suoritetaan disambiguaatiota, laajennetaan käsitteet ensin disambiguaatiohahmon mukaisesti ontologisten suhteiden kautta, jonka jälkeen valitaan todennäköisin vaihtoehto disambiguoiduksi käsitteeksi. Indeksiä varten käsitteet laajennetaan indeksointihahmolla, jonka jälkeen käsitteet sijoitetaan erilliseen indeksiinsä. Seuraavassa käydään prosessin eri vaiheet tarkemmin läpi.

6.2.2 Käsitteistys

Tässä projektissa käsitteistämisellä ymmärretään dokumentin termien tunnistamista ja laajentamista ontologisiksi käsitteiksi. Käytännössä tämä tarkoittaa sitä, että jokaista dokumentin termiä kohti etsitään ontologiasta käsite, jonka suositeltava tai vaihtoehtoinen nimi vastaa kyseistä termiä, ja kyseisen käsitteen URI lisätään dokumenttiin omaan URI-kenttäänsä. Jotta dokumentin merkkijonot voidaan täsmätä ontologian käsitteiden nimiin, pitää merkkijonot ensin lemmata eli palauttaa eri sanamuotoja yhdistävään perusmuotoon, johon myös ontologiassa esiintyvät nimet muunnetaan. Lemmauksen jälkeen termiä vastaava ontologinen käsite on helppo löytää.

6.2.3 Käsiteklusterointi

Kun dokumentin termejä vastaavat käsitteet on löydetty, suoritetaan käsiteklusterointi, jossa termiin liittyvä käsite laajennetaan käsiteklusteriksi. Se sisältää myös muita, alkuperäi-

selle käsitteelle semanttisesti läheisiä käsitteitä. Toisin sanottuna löydetyn käsitteen semanttiset suhteet tutkitaan ja muodostetaan käsiteklusteri näiden suhteiden kohteista.

Koska sovellus on ontologiariippumaton, pitää käsiteklustereita varten haettavat semanttiset suhteet määritellä jokaiselle ontologialle erikseen. Tämä suoritetaan yksinkertaisella polkuperiaatteella, jossa määritetään joukko kyseisen ontologian sisältämiä ominaisuuksia, joiden kautta käsiteklusteri laajennetaan. Jokaiseen polun ominaisuuteen liittyy myös syvyysluku, joka kertoo kuinka pitkälle kyseistä ominaisuutta pitkin kuljetaan.

Kulkeminen polulla tapahtuu etsimällä valitusta ontologiasta kaikki ne kolmikot, joiden predikaattina on polun määrittämä ominaisuus ja subjektina laajennettava- tai laajennuksen aikaisemmassa vaiheessa löydetty käsite. Löydettyjen kolmikkojen objektit muodostavat sen käsitejoukon, joka toimii polun seuraavan vaiheen haun subjekteina.

Polkuun määritellään polkukohtainen painokerroin, joka määrää löydetyille käsitteille annettavan painon. Painokerroin asetetaan välille nollasta yhteen olettaen, että dokumentista löytyviä merkkijonoja vastaavien käsitteiden paino on normeerattu arvoon yksi. Käytännössä painokertoimet kannattanee pitää pieninä, jotta dokumenttien varsinainen, niihin eksplisiittisesti merkitty sisältö pysyy indeksissä tärkeimmässä asemassa.

Käsite laajennetaan iteratiivisesti polun mukaan, eli polun ensimmäisen ominaisuuden mukaiset käsitteet muodostavat uudet laajennettavat käsitteet polun seuraavalle ominaisuudelle. Yksinkertaisella totuusarvolla voidaan määrittää halutaanko polulle määritetty painoarvo lisätä jokaiseen polulta löydettyyn käsitteeseen vai vain polun päästä löytyvään, viimeiseen käsiteryppääseen. Näin voidaan valita helposti esimerkiksi yksinkertaisimmillaan kaikki dokumentista löydettyä termiä vastaavan käsitteen yläkäsitteiden alakäsitteet. Monimutkaisemmat polkurakenteet ovat myös helppoja implementoida käyttämällä polkujen syvyysarvoja.

Lopullinen klusteri on siis alkuperäinen käsite painolla yksi ja joukko semanttisesti läheisiä käsitteitä vaihtelevilla painoilla. Indeksoitavan klusterin lopulliset painoarvot lasketaan alkuperäisen käsitteen esiintymiskertojen neliöjuuren ja polkujen määrittämien painojen tulolla. Tämä tarkoittaa sitä, että painon lisääminen on voimakkaampaa, kun sitä lisätään useamman käsitteen kautta, eikä tekstissä useasti esiintyvä, yksittäinen käsite vaikuta liian merkittävästi. Neliöjuuri varmistaa, että jonkin yksittäisen käsitteen täytyy esiintyä tekstissä todella monta kertaa, että se nostaisi indeksointivaiheessa yksin polkunsa osoittaman klusterin kaikki käsitteet artikkelin käsitteiksi, ks. 5.3.1. Neliöjuuren sijaan voidaan käyttää muunkinlaista tasoittavaa funktiota, joka estää yleisen käsitteen liiallisen dominoinnin.

6.2.4 Hahmokieli

Käsiteklusteroinnin keskeinen elementti on ontologinen ohje, jonka mukaan käsitteet laajennetaan. Tämän ontologisen ohjeen kokonaisuutta kutsutaan tässä hahmoksi ja sen yksittäisiä osia poluiksi. Kuten edellä mainittiin, ovat hahmot aina ontologiakohtaisia ja tyypillisesti ainutlaatuisia, aina tiettyä ontologiaa varten suunniteltuja kokonaisuuksia. Tästä johtuen hahmoja pitää voida tuottaa näppärästi uusia sovelluskohteita varten. Hahmojen esittämiseen tarvitaan siis mahdollisimman yksinkertainen kieli, jossa voidaan ottaa huomioon erilaisten ontologioiden asettamat vaatimukset hahmojen ja polkujen esitykselle.

Jotta hahmokieli voidaan määritellä, tarvitaan määritelmä hahmosta:

- Hahmo koostuu yhdestä tai useammasta polusta
- Polku koostuu yhdestä tai useammasta ominaisuudesta, joiden mukaan ontologinen käsite rikastetaan luomalla käsitteen ympärille käsiteklusteri ontologisten ominaisuuksien yhdistämistä ontologian muista käsitteistä
- Jokaiseen polkuun liittyy painokerroin, jolla rikastettava käsite painottaa käsiteklusterin jäseniä

Polun ominaisuuksille, joita voidaan kutsua myös askeleiksi, pitää myös määritellä se, kumpaan suuntaan ominaisuutta seurataan, eli käsitelläänkö rikastettavaa käsitettä tietyn predikaatin kohdalla subjektina vai objektina. Tämä täytyy tehdä siksi, että kolmikot ovat suunnattuja, eikä tietylle ominaisuudelle suoraa, käänteistä ominaisuutta ole välttämättä määritelty. Esimerkki tästä on `http://www.w3.org/2000/01/rdf-schema#subClassOf`, jolle käänteistä, `superClassOf`-ominaisuutta ei tyypillisesti ontologiaan määritellä. Jos halutaan löytää tietyn käsitteen alaluokka, pitää etsiä sellaiset kolmikot, joissa kyseinen käsite on objektina ja joiden predikaattina on tuo edellä mainittu `subClassOf`-ominaisuus.

Näiden välttämättömien määrittelyjen lisäksi kielen käyttöä helpottamaan luotiin muutama turhaa toistoa välttävä määrittely. Askelille voidaan asettaa syvyysarvo, joka kertoo kuinka monta kertaa askel toistetaan ennen seuraavaan ominaisuuteen siirtymistä. Tällä tavalla voidaan siis helposti määritellä käsiteklusteroinnin piiriin esimerkiksi kymmenen aliluokka-suhteen päässä olevat käsitteet ilman, että tarvittava askel pitäisi kirjoittaa auki kymmentä kertaa.

Vastaava, toistoa vähentävä ominaisuus on polun inklusiivisuus, joka on yksinkertainen totuusarvoinen määrittely. Inklusiivisen polku painottaa kaikkien askelien löytämiä käsitteitä, kun taas epäinklusiivinen polku painottaa vain viimeisen askeleen tuottamaa käsitejoukkoa. Inklusiivisen polun tuottama lopputulos voidaan saavuttaa epäinklusiivisilla poluilla, joita tarvitaan yhtä monta kuin ensimmäisessä polussa on askelia ja niiden syvyyttä, mutta tällä notaatiolla vältetään turhaa toistoa polkujen luonnissa.

Hahmokieli päätettiin toteuttaa XML-pohjaisena, jotta formaatti olisi mahdollisimman helposti hyödynnettävissä ja omaksuttavissa. Esimerkiksi Java-sovelluksen osana XML-pohjaista hahmokieltä voidaan käyttää helposti DOM-parserin avulla (Document Object Model /59/).

Kielen XML-Schema on seuraavanlainen:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <xs:element name="hahmo">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="polku" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
```

```
    <xs:element name="ominaisuus">
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="xs:string">
                    <xs:attribute name="koskeeSubjektia" type="xs:boolean" use="required"/>
                    <xs:attribute name="syvyys" type="xs:integer" use="required"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="polku">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="ominaisuus" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="paino" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:decimal">
                        <xs:minInclusive value="0"/>
                        <xs:maxInclusive value="1"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inklusiivinen" type="xs:boolean" use="required"/>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Hahmokielessä juurielementtinä on `hahmo`-elementti, joka koostuu yhdestä tai useammasta `polku`-elementistä. Poluilla on attribuuttina totuusarvoinen inklusiivisuus, joka siis määrittää painotetaanko rikastuksessa jokainen polun varrella sijaitseva käsite, vai vain polun päässä olevat käsitteet. Polulla on myös paino, joka voi vaihdella välillä nollasta yhteen ja joka määrittää yksinkertaisesti rikastuksessa käsitteille annettavan painon.

Polku muodostuu `ominaisuus`-elementeistä, joiden arvona on seurattava RDF-Property. Ominaisuuksilla on attribuutteina kokonaislukuarvoinen syvyys, joka määrää kuinka monta kertaa ominaisuuden mukainen eteneminen suoritetaan, ja totuusarvoinen koskeeSubjektia, joka määrittää edetäänkö ontologiassa subjektin vai objektin mukaan. Elementin sisään sijoitetaan itse käytettävä ominaisuus.

Esimerkki hahmokielellä toteutetusta hahmosta löytyy liitteestä 2.

6.2.5 Disambiguointi

Käsiteklusterointia voidaan käyttää myös yksinkertaiseen disambiguointiin. Merkkijonon täsmätessä useampaan termiin (sanan ”puumassa” perusmuoto voi olla ”puuma” tai yhdyssana ”puumassa”) tai kun yksiselitteisesti lemmattu termi vastaa useampaa käsitettä (”lap-

set” voi viitata lapsiin perhesuhteena tai ikäryhmänä), voidaan oikeaa käsitettä lähteä arvaamaan nojaten yksiselitteisten termien perusteella dokumentista löytyneisiin käsitteisiin.

Kun käsiteklusteroinnilla halutaan disambiguoida, kannattaa klusterointi suorittaa kahteen kertaan. Ensimmäisellä kerralla suoritetaan disambiguoinnin kannalta edullinen, laaja käsiteklusterointi, jossa käsite antaa painoa hyvinkin pitkien polkujen päässä sijaitseville käsitteille. Kun ontologiaa tutkitaan laajalti pienillä painoilla, saadaan suuria klustereita, jotka leikkaavat toisiaan laajemmin kuin pienet klusterit, mutta toisaalta yksittäisen klusterin vaikutus ei nouse liian korkeaksi. Näin mahdollisimman monet dokumentin käsitteet vaikuttavat disambiguoinnin lopputulokseen. Lopuksi termi valitaan vastaamaan käsitettä, joka sai suurimman painon. Kun löydetyt käsitteet on disambiguoitu, hylätään ensimmäisen käsiteklusteroinnin tuottamat painot ja suoritetaan uusi klusterointi ottaen alkuperäisten varmojen käsitteiden rinnalle disambiguoinnissa saadut käsitteet. Tällä kertaa käytetään lyhyempiä polkuja, mutta painoina käytetään painoja, joiden mukaan lopullinen indeksointi suoritetaan.

Jos disambiguointiklusteroinnissa mikään vaihtoehtoisista käsitteistä ei nouse painoltaan muiden yli, valitaan ne kaikki, jotta taataan dokumentin löydettävyys etsittäessä spesifiä käsitettä. Tällöin lopullisessa klusteroinnissa nousevat kaikkien näiden käsitteiden naapurustojen painot tasaisesti. Haluttaessa disambiguointi voidaan jättää kokonaan tekemättä ja valita yksinkertaisesti kaikki dokumentin merkkijonoja vastanneet käsitteet, joka aineistosta riippuen voi olla paras vaihtoehto.

6.3 Hakutoiminnot

6.3.1 Indeksointi

Järjestelmän selkeyden ja toiminnan kannalta on edullisinta sijoittaa eri ontologioiden mukaiset käsitteistykset omiin indekseihinsä hakutietokannan sisällä. Tämä helpottaa laajentamista ja mahdollistaa modulaarisemman lähestymistavan. Erityistä hyötyä tiedon jakamisesta useaan indeksiin on ontologioiden muuttuessa, jolloin tarvitsee muokata vain yhden indeksin sisältöä uutta versiota vastaavaksi.

Käsiteklusteroinnin jälkeen indeksoitavasta dokumentista on olemassa suuri joukko ontologisia käsitteitä painoarvoineen. Nämä painoarvot pyöristetään yksinkertaisesti lähimpään kokonaislukuun, jonka jälkeen käsiteltävän dokumentin kohdalle ontologiakohtaiseen indeksiin lisätään tuloksen osoittama määrä vastaavia URI:eita. Tf-idf-tyyppinen painotus haun yhteydessä varmistaa sen, että yleiset käsitteet eivät dominoi hakutuloksia (ks. 4.1.3).

Ontologiakohtaisten käsiteindeksien lisäksi luodaan tavanomaiset indeksit otsikoille, alkuperäiselle tekstille, lemmatulle tekstille jne., jotta voidaan toteuttaa myös perinteinen sanahaku, joka on edelleen relevantti ja hyödyllinen niin kauan kun käsitteistykseen käytetyt ontologiat eivät ole täydellisiä.

6.3.2 Haku

Indeksien määrästä johtuen hakuja on yksinkertaista räätälöidä kattamaan haluttujen ontologioiden avulla käsitteistetyt osa-alueet materiaalista. Yksinkertaisin tapa hakea on kirjoit-

taa yksi hakusana, joka lemmataan. Saatua lemmaa verrataan indeksointiin käytettyihin ontologioihin ja etsitään sitä vastaavat käsitteet. Tässä vaiheessa voidaan antaa käyttäjälle mahdollisuus manuaaliseen disambiguointiin, jos käytetyissä hakusanoissa esiintyi monimerkityksisiä merkkijonoja. Löydetyillä käsitteillä suoritetaan haku niiden ontologioita vastaaviin indekseihin ja alkuperäisellä, lemmatulla sanalla etsitään perinteisen sanahaun mukaisesti lemmatusta artikkelimassasta.

Hakusanojen yhdistäminen Boolen loogisten operaattorien avulla (AND, OR, NOT) mahdollistaa hyvinkin tehokkaat ja monipuoliset haut. Myös eri indekseihin suuntautuvat haut voidaan yhdistää. Esimerkiksi sanahakuun voidaan liittää AND-operaattorilla eri ontologioihin kohdistuvien käsitehakujen unioni.

Dokumentteihin liittyvä aikatieto (esimerkiksi uutisartikkelien julkaisupäivämäärät) voidaan myös tallettaa omaan artikkeliinsa, jolloin tätä voidaan käyttää helposti suodattamaan tulosjoukosta vain halutulle aikavälille sijoittuvat dokumentit. Dokumentit voidaan haluttaessa myös järjestää ajan mukaan (yleistä esimerkiksi uutisaineistoon kohdistuvissa hauissa, joissa käyttäjä on kiinnostunut nimenomaan tuoreimmista uutisista), jolloin toissijainen järjestys saman päivämäärän omaavilla uutisilla määräytyy relevanssin mukaan.

Jos hakuja eri indekseihin ei yhdistetä, saadaan tulokseksi joukko dokumentteja, jotka vastaavat hakuun eri indekseistä. Nämä tulokset voidaan esittää erillään tai ne voidaan yhdistää painottaen eri indeksejä eri tavoilla. Painotuksissa voidaan esimerkiksi antaa suurempi painoarvo otsikon perusteella hakuun vastanneille dokumenteille kuin artikkelitekstin perusteella löytyneille dokumenteille. Älykkäästi painottamalla voidaan myös ottaa huomioon ontologioiden heikkouksia ja vahvuuksia. Esimerkiksi yleisen ontologian painotus voi olla spesifimpää ontologiaa alhaisempi, jolloin tiettyyn osa-alueeseen keskittyneen ontologian vahvuudet pääsevät paremmin esiin, kunhan hakukäsitteet löytyvät kyseisestä ontologiasta.

6.3.3 Suosittelu

Konservatiivisimmillaan käsitehakua voi käyttää yksinkertaisesti tukemaan perinteistä sanahakua tuottamalla suosituksia artikkeleista, jotka eivät sisältyneet hakusanat tulosjoukkoon, mutta jotka sisältävät jonkinlaisia yhtäläisyyksiä sen sisältämiin dokumentteihin. Koska dokumentit järjestyvät relevanssin mukaan, voidaan käyttäjälle näyttää lyhyt otos parhaista tuloksista ja jättää muut, heikommat tulokset näyttämättä. Tällä menetelmällä voidaan tarjota käyttäjälle automaattisesti laajempi näkymä haettuja aiheita ympäröiviin ilmiöihin.

Yksi tapa toteuttaa suosittelu käytännössä, on valita sanahaun tuloksista korkeimman relevanssin saaneet dokumentit, joiden määrä voidaan rajata jollain vakiolla joko relevanssin tai absoluuttisen määrän suhteen. Toisin sanottuna voidaan ottaa mukaan kaikki dokumentit joiden relevanssi on suurempi kuin jokin vakio tai vaihtoehtoisesti ottaa tietty määrä korkeimman relevanssin saaneita dokumentteja.

Oleellisimpien dokumenttien joukosta etsitään niiden indeksointiin käytetyt käsitteet ja tehdään leikkaus dokumenttien välillä näiden käsitteiden suhteen. Jälleen voidaan valita jokin vakio, joka määrää kuinka monessa dokumentissa tietyn käsitteen on esiinnyttävä,

että se valittaisiin suositteluun. Vakion sijaan voidaan käyttää myös jotakin prosenttiosuutta. Leikkauksen tuloksena saaduilla käsitteillä suoritetaan uusi haku aineistoon. Käytännössä tämä vaihe vastaa siis korpusperustaista kyselyn laajennusta.

Suosittelun tulosjoukosta pudotetaan pois ne dokumentit, jotka sisältyivät jo alkuperäisen sanahaun tuloksiin, koska näiden suosittelu ei tuota käyttäjälle lisäarvoa. Jäljelle jääneistä dokumenteista korkeimman relevanssin saaneet esitetään käyttäjälle. Jälleen kannattaa valita jokin vakio, joka rajoittaa joko relevanssia tai absoluuttista määrää, koska kuvatulla tavalla saadaan tyypillisesti hyvinkin laaja tulosjoukko, jonka esittäminen käyttäjälle kokonaisuudessaan ei välttämättä palvele tämän perimmäistä tarkoitusta eli tiedonhakua yhtä tehokkaasti kuin esimerkiksi uuden, tarkennetun haun suorittaminen.

Jos hakujärjestelmässä on käytössä aikatietoon perustuva suodatus esimerkiksi artikkelien julkaisuajankohdan mukaan, voidaan suosittelulle määritellä myös aikaikkuna, jonka sisälle suosittelu rajoitetaan. Tämä voidaan myös tehdä tulosjoukon mukaan alkuperäisen rajauksen sijaan, tai sisällyttää vain, jos alkuperäistä rajausta ei ole tehty. Tällöin etsitään tulosjoukon ensimmäinen ja viimeinen aikaleima ja ulotetaan haku näistä vain säädetyn aikaikkunan sisäpuolelle sijoittuviin dokumentteihin.

Suosittelun tuloksia voidaan myös rajoittaa esimerkiksi siten, että vaaditaan, että alkuperäisestä sanahaun hakusanoista löytyneiden käsitteiden on esiinnyttävä tulosdokumenteissa. Jos halutaan rajoittamisen sijaan laajentaa suosittelua, voidaan ottaa suosittelun lähtöjoukoksi pelkän sanahaun tulosjoukon lisäksi myös käsitehaun tulosjoukko.

6.3.4 Selite

Sanahaku on menetelmänä intuitiivinen ja käyttäjille tuttu. Käsiteklusteroinnissa dokumentissa esiintyvät termit nostavat ontologisten lähikäsitteiden painoja, mikä aiheuttaa sen, että dokumentti saatetaan indeksoida käsitteellä, jota suoraan vastaavia termejä ei dokumentissa esiinny. Tällöin on mahdollista, että käsitehaun tulosjoukossa esiintyy dokumentteja, joissa alkuperäisiin hakusanoihin suoraan liittyvät käsitteet eivät esiinny laisinkaan, mikä saattaa hyvinkin aiheuttaa hämmennystä käyttäjässä. Yksi mahdollinen tapa lieventää tätä hämmennystä on esittää käyttäjälle jonkinlainen selitys siitä, miksi jokin tietty artikkeli vastasi kyseessä olevaan hakuun.

Käytännössä tällainen selite tehdään käänteisellä käsiteklusteroinnilla, eli haetaan dokumentista ne käsitteet, jotka nostivat selitettävänä olevan käsitteen painoa. Toisin sanottuna tulosdokumentti rikastetaan alkuperäisillä hahmoilla ja kirjataan ylös ne käsitteet ja polut, jotka nostivat hakukäsitteen painoa. Lopputuloksena saadaan joukko käsitteitä, näiden vaikutusreitti polun muodossa ja vaikutuspaino. Selitteeseen voidaan kirjata myös hakukäsitteistä ne, jotka esiintyivät tulosdokumentissa suoraan sellaisenaan.

Kun selite on saatu muodostettua, pitää se esittää käyttäjälle ymmärrettävässä muodossa. Riippuen käyttäjän asiantuntemuksen tasosta, voidaan selitteeseen sisällyttää enemmän informaatiota. Yksinkertaisimmillaan voidaan vain listata kaikki hakukäsitteiden painoon vaikuttaneet, dokumentissa esiintyneet käsitteet ja esittää tämä vuorovaikutus käyttäjälle. Tarkempaa informaatiota haluavalle voidaan esittää vaikutus painoon ja polku, jota myöten

lopputulokseen päädyttiin. Peruskäyttäjän näkökulmasta tällainen tarkkuus saattaa kuitenkin aiheuttaa pikemminkin lisää hämmennystä kuin hälventää sitä.

7 AIRO-HAKUSOVELLUS

7.1 Johdanto

Projektissa toteutettiin Airo-hakusovellus, jossa sovelletaan dokumentin laajennusta vapaavalintaisen ontologian mukaan.

Airo ohjelmoitiin Java-kielellä hyödyntäen Jena-luokkakirjastoja /49/ (ks. 7.3.1) ontologioiden käsittelyyn, Lucene 2.0.0-pohjaista /26/ (ks. 7.3.2) hakukonetta indeksointiin ja hakuun sekä Poka-annotoijaa /61/ (ks.7.3.3) ontologisten käsitteiden etsimiseen dokumenteista. Aineistona käytettiin osaa Helsingin Sanomien digitalisoitujen artikkelien arkistosta.

Sovellusta testaavana ontologiana käytettiin Yleistä suomalaista ontologiaa (ks. 2.2.6), jossa on tällä hetkellä n. 20 000 käsitettä. /23/

Testikokoonpanolla 8000 artikkelin indeksointiin meni aikaa runsas tunti, jota voidaan pitää kelvollisena tuloksena. Tällöin miljoona artikkelia voitaisiin indeksoida yhdellä vastaavalla tietokoneella n. 125 tunnissa eli alle viikossa. Käytännössä indeksointi suoritetaan kuitenkin huomattavasti tehokkaammalla laitteistolla, jolloin aika lyhenee merkittävästi. Dokumentin laajennuksessa tämä prosessi on kertaluonteinen ja se pitää uusia vain, kun ontologioita lisätään tai muokataan. Mahdollisessa päivittäisessä käytössä kuitenkin yksittäisten, uusien artikkelien lisäys valmiiseen indeksiin on hyvinkin nopeaa.

7.2 Tavoitteet

Airo-sovelluksen tavoitteena oli luoda pohja laajan, tekstimuotoisen artikkeliaineiston automaattiselle annotoinnille ja myöhemmälle semanttisten menetelmien hyödyntämiselle ja niiden suomien mahdollisuuksien tutkimiselle. Lisäksi haluttiin toteuttaa tarkkuuden kärsimättä tavalliseen sanahakuun verrattuna saannin kannalta tehokkaampi hakujärjestelmä dokumentin laajennuksen tekniikoita hyödyntäen. Automaattinen annotointi ei koskaan tuota täydellistä tulosta, mutta se on käytännössä ainoa tapa lähestyä miljoonista artikkeleista muodostuvaa aineistoa.

Sovellukselta vaaditaan suoritusnopeutta ja -varmuutta ja sen täytyy kyetä sopeutumaan useita kertoja päivässä lisättävään materiaaliin. Lisäksi sen täytyy olla konfiguroitavissa monia erilaisia ontologioita varten, jotta sitä voidaan käyttää erilaisten sisältöjen vaatimien erilaisten jäsentävien ontologioiden kanssa.

Airo luotiin Sanoma Data-casen aineistoa ja sen vaatimuksia varten, mutta sovelluksen ydinosat pidettiin mahdollisimman aineistoriippumattomina. Airon muuttaminen toiselle aineistolle sopivaksi on yksinkertaista ja evaluaatio-osassa näin tehdäänkin (ks. 8.2).

7.3 Käytetyt ohjelmistokehykset

7.3.1 Jena

Jena /49/ on Javalle luotu, vapaan lähdekoodin ohjelmointirajapinta, joka mahdollistaa RDF-graafien käsittelyn ja sisältää tuen OWL:ille. Jenalla voidaan lukea ja kirjoittaa RDF-tiedostoja, sekä luoda, lukea ja muokata RDF-malleja. /49/

Käytännössä Jena-malleja navigoidaan listaamalla kaikki kolmikot, joissa tietty resurssi esiintyy. Kyselyissä voidaan rajata resurssin paikka kolmikon sisällä tai vaadittavia resursseja voidaan määrittää useampia. Kolmikkoja voidaan myös lisätä tai poistaa käsiteltävänä olevasta mallista. /49/

7.3.2 Lucene

Lucene /26/ on tehokas ja monipuolisesti skaalautuva joukko Java-kirjastoja tiedonhakusovelluksia varten. Käytännössä se siis tarjoaa helpon tavan lisätä indeksointi- ja hakuominaisuudet johonkin Java-sovellukseen. Doug Cutting kirjoitti Lucenen alun perin 1990-luvun lopulla, mutta siirsi sen vuosituhannen vaihteessa vapaaseen levitykseen ja –kehitykseen Open Source-periaatteen mukaisesti. Vuonna 2002 Lucene siirtyi osaksi Apache Software Foundationin Jakarta-tuoteperhettä. /26/

Lucene mahdollistaa suuren dokumenttimäärän indeksoinnin tietokannoista, tiedostojärjestelmistä tai muista datakonstruktioista. Etsittävän materiaalin formaatti ei sinällään ole rajoite, kunhan sen voi muuttaa tekstimuotoon. Hakupuolella Lucene tukee tiedonhaun tavallisimpia menetelmiä mukaan lukien Boolen kyselyt, fraasit, villit kortit ja tulosten järjestämisen TF-IDF-algoritmilla. /26/

Lucenea käytetään kymmenissä sovelluksissa ja kymmenillä verkkosivuilla. Yksi merkittävimmistä on Wikipedia, joka antaa hyvän kuvan Lucenen skaalautuvuudesta ja nopeudesta materiaalin määrästä riippumatta. /60/

7.3.3 Poka

Poka /67/ on FinnONTO-projektissa kehitetty ontologiaperustainen tiedon eristämisjärjestelmä, jota Airossa käytetään automaattiseen annotointiin. Poka sisältää tuen suomenkieliselle jäsennykselle, mutta sen arkkitehtuuri on kieliriippumaton ja siihen voidaan helposti asentaa erilaisia luonnollisen kielen eristimiä. /61/

Pokan arkkitehtuuri koostuu neljästä pääkomponentista: /61/

- Dokumentin käsittelijä muuttaa erilaiset tekstiformaatit Pokan ymmärtämään muotoon, suorittaa kielellisen esiprosessoinnin ja tallentaa sanojen sijaintitiedon myöhempää käyttöä varten
- Ontologiarajapinta hallinnoi käsite-eristyksessä käytettävät merkkijonoja ja käsittelee eristyksessä käytettävää ontologiaa, sekä mahdollistaa ontologisten suhteiden hyödyntämisen eristysprosessissa.
- Käsite-eristin etsii ontologiassa määritellyt käsitteet annotoitavasta tekstistä joko ontologian mukaisesti tai käsiteriippumattomilla eristimillä. Viimeksi mainituista

Pokaan on toteutettu sääntöperustainen henkilöiden tunnistin sekä säännöllisten lausekkeiden tunnistin.

- Indeksoija luo sana- ja käsitekohtaiset indeksit, joista ensin mainittu kertoo, onko tietystä tekstin sanasta löydetty vastaavuus johonkin käsitteeseen ja jälkimmäinen kertoo, missä dokumentissa jokin tietty käsite on esiintynyt.

7.4 Sovelluksen toiminta

7.4.1 Aineiston käsittely

8000 artikkelin testiaineisto toimitettiin epästandardina tekstitiedostona, jonka kaikkea informaatiota ei hyödynnetty Airon toiminnassa. Tälle tekstitiedostolle kirjoitettiin parseri, joka tuottaa artikkeli-olioita, joihin on tallennettu kaikki Airon kannalta oleellinen informaatio kustakin artikkelista. Koska alkuperäisen tekstitiedoston parsiminen on verrattain pitkä operaatio, tehtiin Artikkeleja varten myös yksinkertainen tiedostoformaatti ja luokka, joka pystyy kirjoittamaan ja lukemaan näitä tiedostoja.

Artikkeliolio itsessään on joukko kenttiä ja metodit näiden lukemiseen ja muuttamiseen. Kentät sisältävät mm. artikkelin otsikon, jokaiselle artikkelille yksilöllisen koodin sekä julkaisupäivämäärän. Näiden lisäksi artikkelin teksti on talletettu sellaisenaan sekä lemmattuna versiona. Artikkelista löytyneiden käsitteiden URI:t on myös talletettu omaan kenttäänsä yksinkertaisesti niin monta kertaa, kuin käsite artikkelissa käsiteklusteroinnin perusteella esiintyy.

Airo sisältää myös edellä mainitun artikkelitiedostoformaatin mukaisten tiedostojen lukemiseen ja kirjoittamiseen tarkoitetut luokat. Testausta varten luotiin myös luokka, joka kirjoittaa artikkeleista yksinkertaisen HTML-tiedoston, johon hakutulokset linkitetään tulosten arvioinnin helpottamiseksi.

Kaiken kaikkiaan aineiston käsittely jätettiin varsin alkeelliseksi, koska tämä osa-alue muuttuu erittäin voimakkaasti riippuen siitä, mihin sovellusta käytetään. Esimerkiksi Sanomadata-casessa, jos Airo otettaisiin varsinaiseen käyttöön, siirrettäisiin artikkelit sovellukseen jollain muulla tavalla ja hakujen tulosartikkelit esitettäisiin olemassa olevan artikkelien selailujärjestelmän avulla.

7.4.2 Rikastus

Rikastus on Airon monimutkaisin osuus ja se sisältää suurimman määrän luokkia. Prosessin tarkoituksena on löytää tekstistä jotakin ontologiaa vastaavat käsitteet ja suorittaa näiden avulla dokumentin laajennus ja käsiteklusterointi. Tämän lisäksi hakutulosten selitykseen käytettävät luokat löytyvät rikastuksen alta.

Käsitteet itsessään esitetään yksinkertaisina olioina, jotka sisältävät käsitteen URI:n merkkijonokentässä ja käsitteen esiintymiskerrat tai käsiteklusteroinnilla saadun lähikäsitteiden muokkaaman painoarvon desimaalilukukentässä. Käsiteklusteroinnin ja disambiguoinnin vaatimaa hahmokieltä varten luotiin hahmo-luokka, jolla on sisäinen polku-luokka, jolla puolestaan on sisäinen askel-luokka. Hahmo sisältää polku-vektorin, sekä metodit polkujen ja askelten lukemiseen kyseisen vektorin sisältä.

Keskeisin osa käsitteistämisessä on Pokaa hyödyntävä käsitteiden eristin, joka tuottaa artikkelin tekstistä sieltä löytyneiden käsitteiden URI:t. Nämä voidaan tuottaa joko merkkijonoina, tai monimutkaisemmassa muodossa hajautusjoukkona, joka ryhmittelee käsitteiden URI:t tekstistä tunnistettujen termien mukaan helpottaen näiden jatkokäsittelyä.

Yksiselitteiset termit lähetetään rikastusoliolle, joka etenee annettua hahmoa yksi polku kerrallaan. Tämä tapahtuu hakemalla Jenalla valitusta ontologiasta kaikki ne kolmikot, joissa esiintyy haluttu predikaatti ja rikastettava URI subjektina (tai objektina, jos rikastus suoritetaan objektin mukaan). Tulokseksi saadaan joukko kolmikkoja, joiden ainoana erona keskenään on niissä esiintyvä objekti. Nämä objektit toimivat seuraavassa vaiheessa rikastettavina URI:eina ja toimenpide toistetaan syvyyden määräämä määrä kertoja jokaiselle polun ominaisuudelle. Jos rikastettava polku on inklusiivinen, eli rikastuksessa painotetaan jokaista polun varrella esiintyvää käsitettä, otetaan jokaisessa vaiheessa tulosjoukko talteen hajautusjoukkoon. Ei-inklusiivisen polun ollessa kyseessä, on tulosjoukko vain viimeisen ominaisuuden jälkeinen objektijoukko. Molemmissa tapauksissa joukosta poistetaan alkuperäinen, rikastettava URI. Tulosjoukon käsitteet saavat lisää painoa polun painon verran kerrottuna alkuperäisen URI:n esiintymiskertojen neliöjuurella.

Yksiselitteisistä termeistä saatujen, rikastettujen käsitteiden avulla disambiguoijaolio arvaa moniselitteisiä termejä vastaavat käsitteet. Käytännössä tämä tapahtuu siten, että käsitekandidaattien yksiselitteisestä rikastuksesta saamia painoja verrataan keskenään ja joukosta valitaan suurimman painon saanut käsite. Jos useammalla käsitteellä on sama paino, valitaan ne kaikki.

Lopuksi yhdistetään yksiselitteiset ja disambiguoidut käsitteet alkuperäisine, rikastamattomine esiintymiskertapainoineen ja suoritetaan uusi rikastus uudella hahmolla. Saadut painot pyöristetään lähimpään kokonaislukuun ja URI:t tulostetaan merkkijonoon tämä määrä kertoja. Tämä merkkijono palautetaan ja talletetaan artikkeliolioon.

7.4.3 Indeksointi ja haku

Airon indeksointi- ja hakuominaisuudet tarjoavat Lucene-arkkitehtuuria hyödyntävän hakukonetoteutuksen artikkeliolioille. Rikastetut artikkelit muutetaan Lucene-Document-muotoon ja eri osat talletetaan omiin kenttiinsä.

Hakuprosessissa muodostetaan kaksi hakua, toinen käsitteiden ja toinen sanojen mukaan. Hakuun voidaan myös sisällyttää alku- ja loppupäivämäärät, jolloin luodaan suodatin, joka rajoittaa Lucenen tuottamat hakutulokset näiden päivämäärien välille. Hakusanat lemmataan ja sanahaku muodostetaan suoraan saaduilla lemmoilla oikeaan sanaindeksiin. Käsitehakua varten etsitään käytetystä ontologiasta hakusanoja vastaavat käsitteet ja saaduilla URI:eilla suoritetaan haku käsiteindeksiin. Jos hakusana vastaa useampaa käsitettä, sisällytetään nämä kaikki hakuun tai-operaattorilla. Tulokset voidaan tulostaa joko aika- tai relevanssijärjestyksessä.

7.4.4 Suosittelu

Suosittelu yhdistää Airossa dokumentin laajennuksen ja kyselyn laajennuksen ja toimii sekä sanahaun, että käsitehaun pohjalta. Algoritmi valitsee korkeintaan kymmenen kor-

keimman relevanssin saanutta dokumenttia sanahaun tuloksista. Näistä dokumenteista etsitään useammassa kuin yhdessä dokumentissa esiintyvät käsitteet ja näiden leikkaus lisätään hakuun. Alkuperäisistä hakutermeistä saadut käsitteet lisätään hakuun unionilla ja lopputuloksella suoritetaan uusi haku tietokantaan. Lisäksi voidaan laskea aikaikkuna, eli suosittelun alku- ja loppupäivämäärät, joiden väliin lopulliset tulokset suodatetaan. Tämä tapahtuu lisäämällä haluttu kokonaisluku alkuperäisten hakutulosten minimi- ja maksimiaikoihin, jolloin saadaan uudet päivämäärät.

Lopuksi Suosittelija poistaa saatujen dokumenttien joukosta alkuperäisessä tulosjoukossa esiintyneet artikkelit, jotta näitä ei näytettäisi käyttäjälle kahteen kertaan. Tämä tapahtuu yksinkertaisesti vertailemalla kahden joukon artikkelien yksilöllisiä koodeja ja säilyttämällä unionin ulkopuolelle jäävät dokumentit. Lopuksi tulosjoukko typistetään haluttuun määrään dokumentteja (esimerkiksi kymmeneen), jotta käyttäjä ei hämmentyisi suuren joukon äärellä.

7.4.5 Selite

Selite muodostetaan Airossa kahdella luokalla, joista toinen muodostaa selitteen ja toinen kirjoittaa siitä HTML-tiedoston. Selitettävällä haulla, artikkelilla ja käsiteklusterointiin käytetyillä hahmoilla muodostetaan selite siten, että artikkelille suoritetaan tavanomainen käsiteklusterointi, mutta rikastuksesta kirjataan ylös ne tapaukset, joissa hakutermeistä tunnistetut käsitteet saavat lisää painoa.

HTML-tiedostoon kirjoitetaan taulukko suoraan artikkelista löytyneistä hakukäsitteistä, sekä siitä, mitkä artikkelin käsitteet vaikuttivat hakukäsitteiden painoihin ja kuinka paljon.

7.5 Toimintaesimerkki

Oletetaan seuraavanlainen skenaario: halutaan indeksoida YSO:a käyttäen artikkeli, jonka otsikko on ”Missi Maija Meikäläinen jatkoi mallin töitä”. Käytännössä Airo hyödyntäisi artikkelin tekstin, jolloin käsitteistys suoritettaisiin huomattavasti useammalle sanalle yhtä aikaa, mutta nyt käsitellään vain otsikko erillisenä entiteettinään esimerkin havainnollisuuden vuoksi. Lause lemmautuu muotoon ”missi maija meikäläinen jatkaa malli työ” ja Poka löytää kyseisestä lauseesta kolme YSO-termiä, jotka vastaavat kuutta eri käsitettä. ”Missi” ja ”työ” vastaavat käsitteisiin `missit` ja `työ`, eivätkä vaadi disambiguaatiota. Sen sijaan termi malli vastaa neljää YSO:ssa esiintyvää käsitettä, jotka tarkoittavat mallia fyysisenä jäljitelmänä (esimerkiksi pienoismallit), mallia jonkin ilmiön selitteenä (esimerkiksi ilmastomallit), mallia ammattina ja viimeiseksi mallia ihmisen toiminnan motivoijana (esimerkiksi roolimallit).

Kaksi varmaa tapausta sijoitetaan käsitteiden joukkoon ja luodaan disambiguointia varten polut, jotka on esitetty taulukossa 1.

Taulukko 1. *Disambiguointiin käytetty hahmo*

Polun ominaisuudet	Syvyys	Paino	Inklusiivisuus
subClassOf (s), subClassOf (o)	1,1	0,001	Epätosi

associativeRelation (s)	2	0,003	Tosi
subClassOf (s)	1	0,001	Epätosi
subClassOf (o)	5	0,002	Tosi

Taulukon 1 ensimmäinen sarake, polun ominaisuudet, määrittää polun muodostavat ontologian ominaisuudet (Property), joita pitkin käsite laajennetaan klusteriksi. Ominaisuutta seuraava suluissa olevan kirjain kertoo koskeeko kyseinen ominaisuus subjektia (s) vai objektia (o), eli kumpaan suuntaan ominaisuutta pitkin edetään rikastusta suoritettaessa. Esimerkiksi ensimmäisessä tapauksessa kuljetaan ensin subjektina toimivan, klusteroitavan käsitteen yläluokkaan subClassOf-suhdetta pitkin ja valitaan sen jälkeen kaikki löydetyn yläluokan aliluokat. Koska superClassOf-suhdetta ei OWL-kielellä toteutetuissa ontologioissa tyypillisesti käytetä, tehdään haku käänteisesti etsien kolmikkoja, joiden predikaattina on subClassOf ja objektina lähtökäsite.

Taulukon toinen sarake määrittää syvyydet, eli sen, kuinka monta kertaa klusterointi iteroidaan kullekin polun Propertylle. Kolmannen sarakkeen paino on kerroin, jolla klusteroitavan käsitteen esiintymiskertojen neliöjuuri kerrotaan, jotta saadaan klusteroinnin tulosjoukon käsitteiden painot. Viimeinen sarake sisältää totuusarvon siitä lisätäänkö painoarvo kaikkiin klusteroinnissa ilmenneisiin käsitteisiin vai vain viimeisellä iteroinnilla saatuun tulosjoukkoon. Jos kentän arvo on tosi, painotetaan kaikkia, jos taas epätosi, lisätään vain viimeisen tulosjoukon painoa.

Projektia varten luodulla XML-pohjaisella hahmokielellä taulukko esitettäisiin seuraavasti:

```
<hahmo>
    <polku paino=”0,001” inklusiivinen=”false”>
        <ominaisuus koskeeSubjektia=”true” syvyys=”1”>
             http://www.w3.org/2000/01/rdf-schema#subClassOf
        </ominaisuus>
        <ominaisuus koskeeSubjektia=”false” syvyys=”1”>
             http://www.w3.org/2000/01/rdf-schema#subClassOf
        </ominaisuus>
    </polku>
    <polku paino=”0,003” inklusiivinen=”true”>
        <ominaisuus koskeeSubjektia=”true” syvyys=”2”>
             http://www.yso.fi/onto/yso-meta/2007-03-02# associativeRelation
    </polku>
        ...
</hahmo>
```

YSO:ssa on käsitteelle `missit` määritelty yläluokka `toimintaan liittyvä rooli`, jonka alaluokkana esiintyy käsite `mallit_3`, joka viittaa malleihin ammattina. Näin ollen kyseinen käsite saa painon 0,001 ensimmäisen polun mukaan. Kumpikaan varmoista termeistä, `missit` tai `työ`, ei esimerkin poluilla tuota yhtään lisäpainoa muille mallin käsitteille. Näin ollen disambiguoinnissa valitaan oikeaksi käsitteeksi `mallit_3`.

Klusteroitaviksi käsitteiksi siis saatiin `missit`, `työ` ja `mallit_3`, joilla kaikilla on yksi esiintymiskerta, eli painona yksi (disambiguointivaiheessa tuotetut rikastuspainot hylätään). Saadut kolme käsitettä rikastetaan taulukossa 2 esitetyillä poluilla.

Taulukko 2. *Lopulliseen klusterointiin käytetyt polut*

Polun ominaisuudet	Syvyys	Paino	Inklusiivisuus
subClassOf (s), subClassOf (o)	1,1	0,05	Epätosi
associativeRelation (s)	1	0,2	Tosi
subClassOf (s)	1	0,05	Epätosi
subClassOf (o)	1	0,1	Tosi

Kuten taulukosta 2 huomataan, ovat lopullisessa klusteroinnissa käytetyt painoarvot korkeampia kuin disambiguoinnin vastaavat, kun taas klusteri itsessään on kapeampi. Painot ja esiintymiskertojen neliöjuuret kerrotaan keskenään ja näin saadaan suuri joukko käsitteitä. `mallit_3` saa painon 1,05 ensimmäisen polun mukaan ja vastaavasti myös `missit`–käsitteen paino nousee samaan lukuun. `työ`-käsitteen paino pysyy ennallaan.

Klusteroinnin jälkeen saadut käsitekohtaiset painoarvot pyöristetään lähimpään kokonaislukuun ja indeksoidaan ontologiakohtaiseen indeksiin. Käsitteen URI sijoitetaan indeksiin pyöristetyn painoarvon osoittama määrä kertoja ja muut indeksit luodaan otsikolle, artikkelin päivämäärälle ja leipätekstille. Koska klusteroitavia käsitteitä oli alun perin vain kolme ja jokaisella vain yksi esiintymiskerta, ei yhdenkään uuden käsitteen paino nouse yli 0,5:n (jolloin se indeksoitaisiin), eli indeksiin päätyvät vain löydetyt kolme käsitettä. Jos jonkin käsitteen paino olisi noussut yli 1,5:n rikastuksen seurauksena, sijoitettaisiin se dokumentin käsitteet-kenttään useaan kertaan, jotta Lucenen sisäänrakennettu TF-IDF-painotus laskisi käsitteeseen liittyvän relevanssin oikein ja osaisi näin järjestää hauissa saadut tulokset.

Hakua suoritettaessa hakusanat lemmataan ja käsitteistetään edellä kuvatulla tavalla ontologiakohtaisesti ja saaduilla käsitteillä suoritetaan haku niitä vastaaviin ontologiaindekseihin. Hakutermejä ei disambiguoida, vaan hakuun sisällytetään kaikki termejä vastaavat käsitteet. Tämän lisäksi suoritetaan tavanomainen sanahaku otsikko- ja leipäteksti-indekseihin. Hakuun voidaan myös liittää tuloksia rajaavat alku- ja loppupäivämäärät, jotka suodattavat tulosjoukon kattamaan vain halutun aikajakson. Viimeiseksi määritellään tulosten esitysjärjestys, eli halutaanko tulokset esitettäviksi aika- vai relevanssijärjestyksessä. Relevanssi lasketaan yksinkertaisesti Lucenen sisäänrakennetulla, indeksikohtaisella TF-IDF-painotuksella.

Tarkka sovelluksen toimintakuvaus luokkineen ja metodeineen on sisällytetty tämän työn loppuun liitteeksi.

7.6 Ongelmat ja heikkoudet

7.6.1 Polkujen muodostaminen klusterointia varten

Tällä hetkellä polut, joiden mukaan klusterointi suoritetaan, on valittu ilman käytännön testejä. Ne edustavat puhtaasti teoreettista hypoteesia ja ovat validit vain Yleisen suomalai-

sen ontologian suhteen. Jotta järjestelmästä saataisiin tällaisenaan kaikki hyöty irti, pitäisi suorittaa laajoja käyttäjätestejä erilaisilla poluilla ja etsiä sekä disambiguoinnin, että painotusklusteroinnin kannalta toimivimmat polkukonfiguraatiot.

Disambiguoinnin suhteen optimaalisinta polkukonfiguraatiota voitaisiin etsiä yksinkertaisesti vertaamalla järjestelmän tuottamia tuloksia manuaalisesti käsitteistettyihin dokumentteihin. Kuitenkin, kuten luvussa 4.3.2 mainittiin, on yksiselitteinen disambiguointi usein hankalaa tai jopa mahdotonta, minkä vuoksi parhaimpiin tuloksiin päästäisiin käyttämällä useamman asiantuntijan disambiguoimaa aineistoa verrokkina Airon tuottamiin tuloksiin.

Polut pitää myös tällä hetkellä konfiguroida jokaista ontologiaa varten erikseen, mikä vaatisi jälleen uudet tutkimukset polun toimivuuden suhteen. Lisäksi käytännössä yksi polku-ontologia-pari ei voi olla ideaali kaikille datajoukoille, vaan parhaisiin tuloksiin epäilemättä päästäisiin sovittamalla polut kohdedokumenttien mukaan. Näitä toimintoja varten polkujen luomista, muokkaamista ja testaamista varten tarvittaisiin jonkinlainen helppokäyttöinen työkalu.

Yhden datajoukon sisälläkin voi olla tarvetta usealle erilaiselle polkujoukolle, jos aineiston eri dokumentit poikkeavat toisistaan merkittävästi. Esimerkiksi klusteroinnin kannalta käsitteiden esiintymismäärä dokumentissa on ensiarvoisen tärkeää. Algoritmissa käytetty neliöjuuri tarkoittaa, että yksittäisen käsitteen esiintyminen useasti ei nosta kyseisen klusterin painoa kovinkaan nopeasti, mutta pitkän dokumentin ollessa kyseessä, voi yksittäinen termi esiintyä riittävän monta kertaa, että se nostaa yksin koko klusterinsa artikkelin käsittelemien käsitteiden joukkoon. Tämä saattaa, ontologiasta riippuen, aiheuttaa virheellisiä käsitteistyksiä (ks. 5.2.1). Dokumentin pituuden pitäisi siis mahdollisesti vaikuttaa polkuihin lyhentäen niitä ja laskemalla eri suhteisiin liittyviä painoja virheiden välttämiseksi. Toisin sanottuna painon pitäisi olla funktio, jossa dokumentin pituus esiintyy muuttujana.

7.6.2 Disambiguoinnin toimivuus

Airoon toteutettu yksinkertainen disambgiuointi ei toimi täydellisesti. Ensinnäkin, käytetty menetelmä estää mahdollisuuden, että samassa artikkelissa käsiteltäisiin samaa termiä eri merkityksissä. Tämä muodostuu ongelmaksi etenkin silloin, kun haetaan artikkeleita yksittäisellä käsitteellä eikä kaikilla termiä vastaavilla käsitteillä. Jos artikkeli käyttää yhtä termiä useammassa kuin yhdessä eri merkityksessä, indeksoidaan artikkeli useiden käsitteiden mukaan vain, jos termiä vastaavat käsitteet saavat saman painoarvon disambiguoinnin aikana, mikä on käytännössä täysin sattumanvaraista.

Sanaklustereihin perustuva disambiguaatio toimii periaatteessa kelvollisesti, kun kyse on yleisestä käsiteontologiasta, mutta esimerkiksi paikkaontologian ollessa kyseessä tilanne mutkistuu. Kuten edellä esitettiin, voitaisiin paikkoja pyrkiä disambiguoimaan niiden maantieteellisen sijainnin perusteella (ks. 6.2). Tämän menetelmän ongelmana kuitenkin on, että sama uutinen voi hyvinkin käsitellä useita maantieteellisiä alueita samaan aikaan. Esimerkiksi artikkeli, jossa puhutaan Turusta ja Helsingistä saattaisi disambiguoida Helsingin Suomen pääkaupungin sijasta Turun Helsinki-kaupunginosaksi, vaikka todennäköisesti vain pieni murto-osa artikkeleista käsittelisi kyseistä kaupunginosaa. Samoin Alan-

komaiden ja USA:n välistä kauppaa koskevassa artikkelissa saattaisi Amsterdam disambiguoitua USA:n Amsterdamiksi, jos Yhdysvallat mainittaisiin Alankomaita useammin.

Ongelma voidaan osittain ratkaista suosimalla disambiguoinnissa ontologisessa hierarkiassa korkeammalla olevia paikkoja (esimerkiksi kaupunkeja kaupunginosien sijaan), mutta tällä tavalla ongelmaa ei saada kuitenkaan kokonaan poistettua.

7.6.3 Ontologioista aiheutuvat ongelmat

Vaikka käsitteet disambiguoitaisiinkin oikein, saattaa YSO:n rakenne aiheuttaa ongelmia klusteroitaessa tiettyjen käsitteiden suhteen. Esimerkiksi YSO:n käsite `lapset_2`, joka viittaa lapsiin sukulaisuussuhteena omaa assosiatiivisen suhteen `insesti`-käsitteeseen ja toisin päin. Tämä on kuitenkin suurimmassa osassa käytännön tapauksia yksisuuntainen assosiatiivinen suhde, eli insestiä käsittelevät artikkelit liittyvät sukulaisuussuhteisiin, mutta toisin päin tämä ei yleensä ole totta. Suurimmassa osassa käsitteitä assosiatiiviset suhteet ovat kuitenkin kaksisuuntaisia klusteroinninkin kannalta, eli sinällään niiden säilyttäminen poluissa on perusteltua.

Toinen, hieman vastaavanlainen, ongelma on aliluokkasuhteen merkityksen vaihtelevuus. Esimerkiksi kuvitteellisessa biologiaontologiassa käsitteen `linnut` aliluokkaketju voi kulkea seuraavanlaisen käsiteketjun: `eläimet` – `linnut` – `tiaiset` – `talitiaiset`. Huomataan, että hierarkian alaosassa aliluokkasuhde on käsitteellisesti läheisempi tai ainakin se edustaa yleisempää yhteyttä, joka kiinnostanee loppukäyttäjää enemmän. Toisin sanottuna, talitiaisista kertova artikkeli kiinnostaa todennäköisesti enemmän tiaisista kiinnostunutta käyttäjää kuin linnuista kertova artikkeli eläimistä kiinnostunutta käyttäjää.

Molemmissa ylläkuvatuissa tapauksissa voidaan kuitenkin laskea ongelman yhdeksi aiheuttajaksi ongelmallisten termien yleisyys ja voidaan olettaa, että ne eivät edusta tyypillisimpiä hakukäsitteitä.

Toimiakseen hyvin Airo vaatii ontologioilta paljon. YSO on jatkuvassa kehityksen tilassa, mutta sen juuret asiasanastona ovat vielä selkeästi näkyvissä. Esimerkiksi käsitteiden `jääkiekko` ja `jääkiekkoilijat` välillä ei ole minkäänlaista suoraa yhteyttä ja ne löytyvät hyvin eri puolilta varsinaista luokkahierarkiaa. Tämä aiheuttaa sen, että vaikka käsitteillä todellisuudessa onkin hyvin voimakas keskinäinen korrelaatio ja suurin osa yhtä käsittelevistä dokumenteista sivuaa myös toista, ei Airo voi tätä suhdetta mitenkään hyödyntää. Ongelma voidaan ratkaista kehittämällä ontologioista yhä parempia, mutta se vaatii merkittävää ajallista ja rahallista panostusta.

7.7 Jatkokehitysmahdollisuudet

7.7.1 Tuki tyypillisille hakuoperaattoreille ja suoralle käsitehaulle

Periaatteessa Airo sisältää jo tuen suoralle käsitehaulle, jossa käyttäjä valitsee ontologiasta haluamansa hakukäsitteet sisällytettäviksi hakuun. Käytännössä tämän toiminnallisuuden implementointi järkeväksi käyttöliittymäksi, jonka hyödyntämiseen ei vaadittaisi tiedonkäsittelyn ammattilaista, ei ole triviaalia. Erityisesti käytettäessä useita ontologioita aineiston indeksointiin, on erilaisten käsitteistövaihtoehtojen seulominen peruskäyttäjän näkökul-

masta hankalaa. Eräs mahdollisuus olisi tuottaa käytetyistä ontologioista visualisointiontologiat, jotka on tarkoitettu maallikon käyttöön aineiston selailussa, eivätkä sisällä kaikkea informaatiota, vaan vain selailun kannalta oleellisen mahdollisimman helposti ymmärrettävässä muodossa.

Tuki tyypillisille hakuoperaattoreille sisältyy jo sinällään Lucene-hakumoottoriin, mutta näiden sisällyttäminen Airoon vaatisi kysymysparserin implementointia. Jonkinlainen periaatepäätös hakusanojen välillä esiintyvien hakuoperaattorien vaikutuksessa käsitehakuun pitäisi myös tehdä.

7.7.2 Paikkaontologian implementointi

Airoon on tällä hetkellä mahdollista implementoida lisää ontologioita verrattain vaivattomasti, mutta paikkaontologia tarjoaa muutamia erityisiä haasteita ja mahdollisuuksia. Dokumenttien käsitteistäminen paikkaontologian avulla tapahtuu periaatteessa samalla tavalla kuin muidenkin ontologioiden ollessa kyseessä, mutta käsiteklusterointi suoritettaisiin vain maantieteellisessä hierarkiassa ylöspäin.

Disambiguointi eri käsitteiden välillä tarjoaa paikkaontologian tapauksessa omat haasteensa. Uutisten tapauksissa saatetaan mainita esimerkiksi ulkomaisen kaupungin yhteydessä valtio, jossa se sijaitsee, jolloin olisi hyödyllistä ottaa disambiguoinnissa huomioon kuinka monen sanan päässä toisistaan eri paikat tekstissä esiintyvät, jolloin disambiguoinnissa voitaisiin painottaa enemmän lähempänä esiintyviä paikkoja. Ongelmia tulee kuitenkin esimerkiksi artikkelien kanssa, jossa mainitaan Ranskan Pariisi ja tämän lisäksi Yhdysvallat useaan kertaan, jolloin nykyinen, varmojen käsitteiden esiintymistiheyteen nojaava disambiguointijärjestelmä tulkitsee Pariisin tarkoittavan Yhdysvalloissa sijaitsevaa, samannimistä kaupunkia. Tätä ongelmaa on hyvin hankala ratkaista siten, että Yhdysvaltojen Pariisiin on kuitenkin mahdollista viitata.

Paikkaontologian sisällyttämien Airoon mahdollistaisi myös uusia käyttöliittymäratkaisuja. Kuten kappaleessa 6.2 on esitetty, voitaisiin käyttöliittymään toteuttaa artikkelien rajaaminen kartalta valitulle alueelle, jolloin koordinaatteja voitaisiin käyttää nykyisen päivämääräsuotimen kaltaisesti.

8 JÄRJESTELMÄN TESTAUS

8.1 CLEF Test Suite

Airo-järjestelmän testaamiseen käytettiin Cross Language Evaluation Forumin (CLEF) luomaa testijärjestelmää, jolla voidaan arvioida tiedonhakujärjestelmien ominaisuuksia /53/. CLEF on EU:n Information Society Technologies -ohjelman tukema, kansainvälinen järjestö, jonka tarkoituksena on luoda infrastruktuuria ja testiympäristöjä Euroopan kielillä toimiville tiedonhakujärjestelmille. Näihin liittyen CLEF järjestää vuosittain evaluaatiokampanjan eurooppalaisille järjestelmille. /68/

Airon testaamisessa käytetty versio CLEF Test Suitesta oli ELRA-E00008 The CLEF Test Suite for the CLEF 2000-2003 Campaigns, joka nimensä mukaisesti sisältää vuosien 2000-

2003 evaluaatiokampanjoihin liittyvät tietokannat, sekä sovelluksen saatujen tulosten vertailuun.

Suomenkielinen osa tietokantaa koostuu dokumenttikokoelmasta, joka perustuu Aamulehden artikkeleihin vuosilta 1994-1996 SGML-muodossa, sekä näihin liittyvistä hakutehtävistä. Testaukseen käytettiin vuoden 2003 hakutehtäviä, joita oli 60 kappaletta. Hakutehtävät on esitetty XML-muodossa ja ne koostuvat yksilöllisestä tunnusluvusta, otsikosta, kuvauksesta ja narratiivista. Esimerkiksi ensimmäisen suomenkielisen hakutehtävän (koodi C141) otsikko on ”Kirjepommi Kiesbauerille”, kuvaus ”Etsi tietoja juontaja Arabella Kiesbauerille osoitetun kirjepommin räjähdyksestä TV-kanava PRO7:n studiossa,” ja narratiivi ”Oikeistoradikaalien mustalle TV-persoonalle Arabella Kiesbauerille lähettämä kirjepommi räjähti TV-kanava PRO7:n studiossa 9. kesäkuuta 1995. Avustaja loukkaantui. Kaikki räjähdystä ja poliisikuulusteluja käsittelevät dokumentit ovat relevantteja. Muut pommikirjehyökkäykset eivät ole mielenkiinnon kohteena.” Otsikko siis määrittelee aiheen muutamalla sanalla, kuvaus tarkentaa sitä ja narratiivi esittelee aiheen taustaa ja rajauksia. Itse haut muodostettiin pelkän hakutehtävän otsikon avulla. /53/

Hakutehtäviin liittyy relevanssitiedosto, jossa on binäärisesti määritelty jokaisen kokoelman dokumentin relevanssi kyseiselle hakutehtävälle. Tämä relevanssitiedosto edustaa siis oikeita vastauksia hakutehtäviin ja se on tehty manuaalisesti tiedonhaun ammattilaisten toimesta. Evaluoitavan järjestelmän tuottamia tuloksia voidaan verrata kyseistä relevanssitiedostoa vastaan ja toimenpiteen helpottamiseksi CLEF Test Suite sisältää C-ohjelman, joka tuottaa vertailusta joukon tunnuslukuja, kunhan tulokset esitetään tietyssä formaatissa. Huomattavaa on, että tietokanta ei sisällä joillekin hakutehtäville yhtään relevanttia dokumenttia. /53/

8.2 Testit

8.2.1 Alkuvalmistelut

Testejä varten luotiin joukko Java-luokkia, joilla CLEF-evaluaatioaineisto muutettiin indeksoitavaan muotoon ja jolla Airon tuottamat tulokset voitiin muuttaa CLEF Test Suiten vaatimaan muotoon. Indeksoinnissa jokaisesta tietokannan osasta, jolla on yksilöivä koodi, luotiin artikkeli-olio Airoon ja nämä indeksoitiin lemmattuna ja alkuperäisenä tekstinä. Lisäksi teksti käsitteistettiin ja saadut käsitteet indeksoitiin omaan indeksiinsä. Käsiteklusteroinnissa hahmoina käytettiin toimintaesimerkissä esiteltyjä hahmoja (ks. 7.5).

8.2.2 Suoritetut haut

Airon eri ominaisuuksia testattiin viidellä eri haulla:

- Sanahaku tarkoittaa perinteistä sanahakua, jossa hakusanat lemmataan ja haku suoritetaan lemmattujen dokumenttien indeksiin.
- Käsitehaussa etsitään hakusanoja vastaavat käsitteet YSO:sta ja näillä haetaan dokumentteja käsiteindeksiin perustuen.
- Sana- ja käsitehaku yhdistää kaksi edellistä hakua Lucenen Boolen haun Should-määreellä, joka vastaa unionia.

- Suosittelu on Airon suosittelutoiminnon tuottamat ensimmäiset 11 hakutulosta. Käytännössä suoritetaan sanahaku, etsitään tulosdokumenttien joukosta yleisimmin esiintyvät käsitteet ja suoritetaan näillä käsitehaku aineistoon, jossa kuitenkin jätetään huomiotta alkuperäisen sanahaun tuloksena saadut dokumentit.
- Älykkäästi yhdistetty sanahaku ja suosittelu tarkoittaa sitä, että tietokantaa suoritetaan sanahaku ja suosittelu edellä kuvatunlaisesti, jonka jälkeen lopulliset hakutulokset luodaan ottamalla sanahaun ensimmäiset 15 tulosta, joiden perään lisätään suosittelusta saadut kymmenen relevanteinta dokumenttia ja loppuun lisätään loput sanahaun tulosdokumentit.

Itse haku tehtiin yksinkertaisesti hakutehtävien otsikoista jättämällä kuvaukset ja narratiivit huomiotta. Tuloksia evaluoitaessa otettiin huomioon maksimissaan 1000 ensimmäistä palautettua dokumenttia muiden jäädessä huomiotta.

8.2.3 Saanti ja tarkkuus

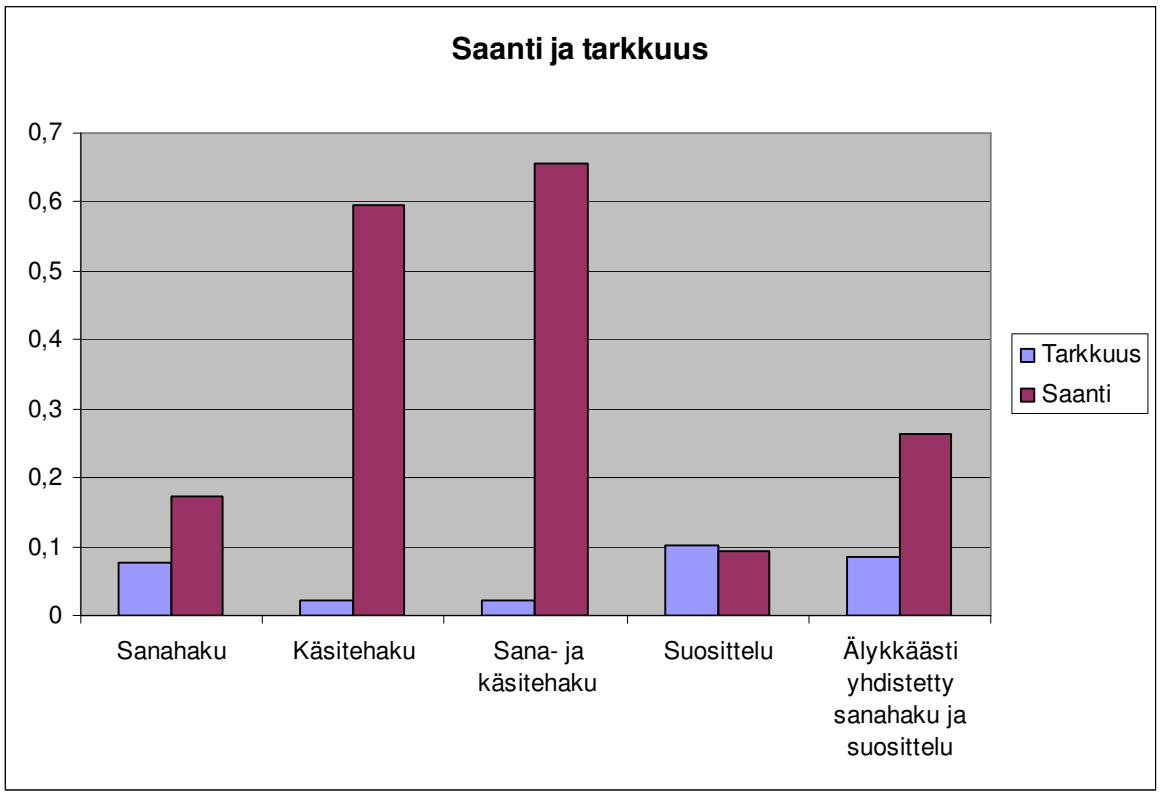
Tärkeimmät tiedonhakujärjestelmien arviointikriteerit ovat saanti ja tarkkuus (ks. 4.2.1).

$$Tarkkuus = \frac{N_{rel} \cap N_{ret}}{N_{ret}} \tag{8}$$

$$Saanti = \frac{N_{rel} \cap N_{ret}}{N_{rel}} \tag{9}$$

Kaavoissa 8 ja 9 on esitetty tarkkuuden ja saannin matemaattiset esitykset. N_{rel} viittaa haun kannalta relevanttien dokumenttien lukumäärään ja N_{ret} tarkoittaa hakuun vastauksena palautettujen dokumenttien määrää. Kuten aiemmin todettiin, voidaan toista yleensä kasvattaa toisen kustannuksella. /54/

Evaluoinnissa laskettiin eri hakujen tuottamat tarkkuuden ja saannin keskiarvot ja nämä on esitetty kuvassa 16.



Kuva 16. *CLEF Test Suiten evaluoinnissa saadut saannin ja tarkkuuden keskiarvot eri hakumenetelmille*

Kuvan 16 saannin ja tarkkuuden arvot sijoittuvat välille nollasta yhteen ja vertailussa sanahaun tuottamaa tulosta voidaan pitää perustasona, johon muita tuloksia verrataan. Kuvasta nähdään, että käsitehaun ja yhdistetyn sana- ja käsitehaun saannit ovat erittäin korkeat, mutta tarkkuus heikompi pelkkään sanahakuun verrattuna. Tämä on odotettu tulos, koska käsitehaku palauttaa merkittävästi enemmän dokumentteja kuin sanahaku, jolloin näiden joukossa on myös suuri osa relevanteista dokumenteista. Kaikki dokumentit palauttamalla päästäisiin luonnollisesti täydelliseen saantiin, mutta tarkkuus olisi erittäin pieni ja hausta ei olisi juuri mitään hyötyä, paitsi ehkä dokumenttikokoelman järjestämisen kannalta.

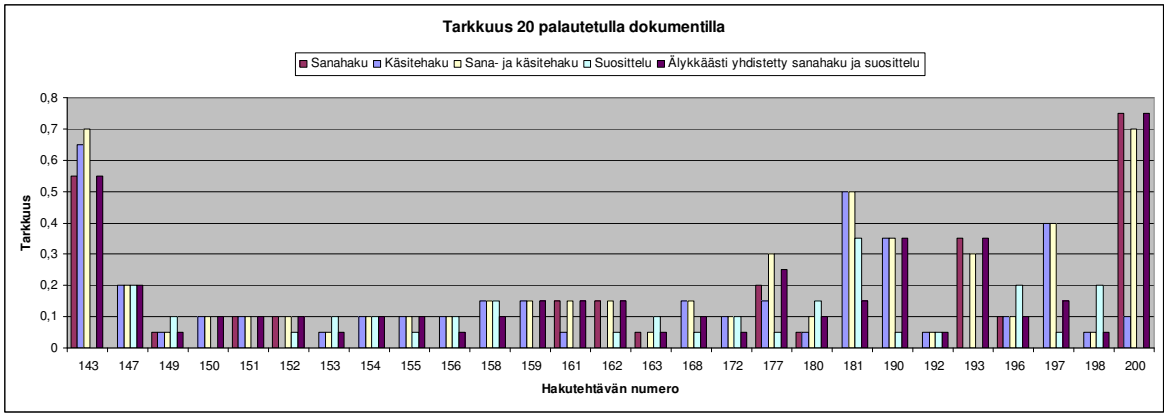
Suosittelussa tarkkuus on hieman sanahakua korkeampi ja saanti alhaisempi, mistä jälkimmäinen on luonnollista, koska tulosten määrä rajoitettiin yhteentoista, joka on vähemmän kuin joidenkin hakutehtävien sisältämien relevanttien dokumenttien lukumäärä. Tämä rajoitus vaikuttaa myös tarkkuuteen, joka täytyy ottaa huomioon suosittelun tuottamia tunnuslukuja arvioitaessa. Huomattavaa kuitenkin on, että menetelmästä johtuen kaikki suosittelun tuottamat dokumentit eroavat sanahaun tuottamasta tulosjoukosta.

Viimeinen hakumenetelmä, älykkäästi yhdistetty sanahaku ja suosittelu, yhdistää nimensä mukaisesti sanahaun ja suosittelun, joten odotetusti saanti on näiden hakujen saantien summa, koska kyseisten menetelmien tulosjoukot eivät leikkaa. Tarkkuus puolestaan on kahden komponenttimenetelmän tarkkuuksien välimuoto, joka sekin on odotettua.

8.2.4 Tarkkuus rajallisella dokumenttimäärällä

Pelkkä yhteissaanti ja -tarkkuus perustuen kaikkiin palautettuihin dokumentteihin eivät anna täydellistä kuvaa menetelmien ominaisuuksista todellisessa käyttötilanteessa. Hakua tietokantaan suorittava käyttäjä ei ole tyypillisesti kiinnostunut sadoista dokumenteista, vaan korkeintaan muutamasta kymmenestä ensimmäisestä tuloksesta. Tämän vuoksi tarkkuus rajallisella dokumenttimäärällä on mielekäs tunnusluku ja CLEF Test Suite tuottaakin tämän automaattisesti.

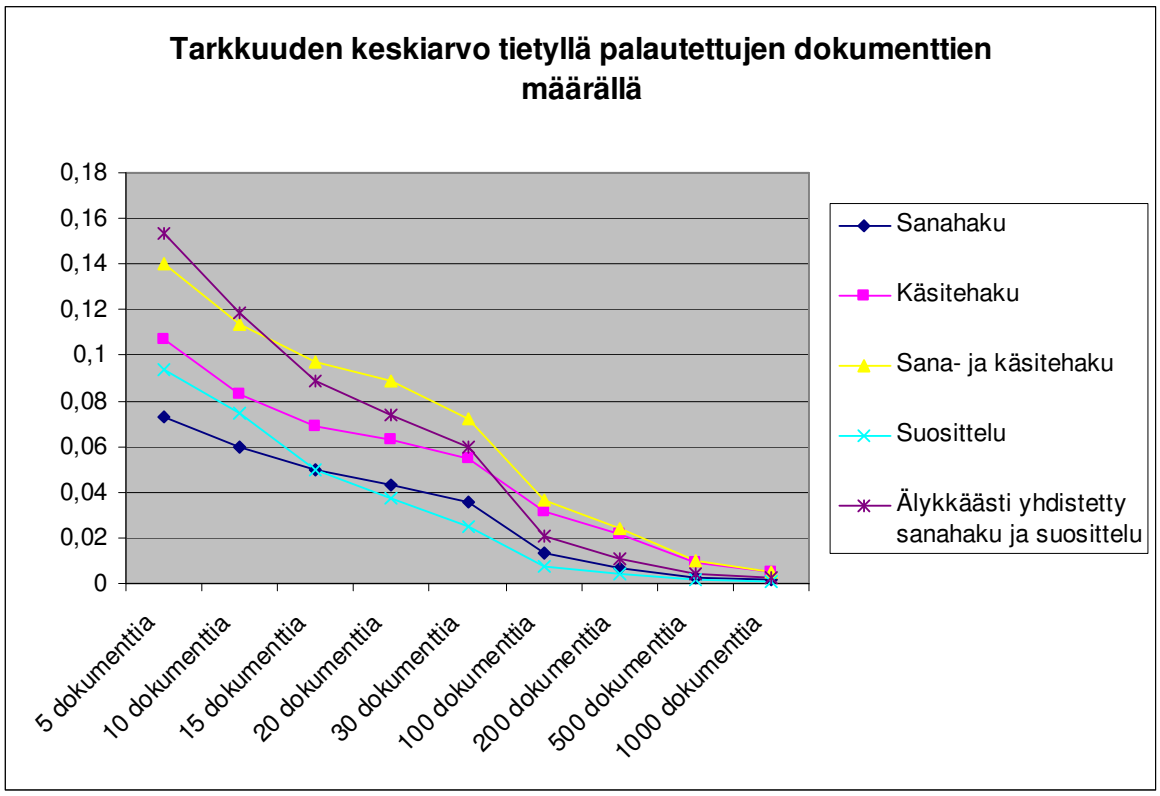
Tarkkuus rajallisella dokumenttimäärällä lasketaan kuten tarkkuus yleisestikin, mutta huomioon otetaan vain ensimmäiset N palautettua dokumenttia. Jos palautettuja dokumentteja on vähemmän kuin N kappaletta, oletetaan puuttuvat dokumentit vääriksi. Tällöin täydelliseen tarkkuuteen ei voida päästä, jos N on suurempi kuin hakutehtävään liittyvien relevanttien dokumenttien kokonaismäärä tietokannassa. Tässä tutkimuksessa käytettyihin hakutehtäviin liittyvien relevanttien dokumenttien lukumäärien keskiarvo oli 10,73 niissä hakutehtävissä, joille tietokannassa esiintyi vähintään yksi relevantti dokumentti.



Kuva 17. *Tarkkuus 20 palautetulla dokumentilla hakutehtävillä joihin palautettiin vähintään yksi relevantti dokumentti*

Kuvassa 17 on esitetty eri menetelmillä saatu tarkkuus 20 palautetun dokumentin kohdalla. X-akselilla on esitetty hakutehtävän numero, kun kuvasta on jätetty pois ne hakutehtävät, joihin mikään menetelmä ei palauttanut yhtäkään relevanttia dokumenttia. Jos jonkin hakutehtävän kohdalta puuttuu jokin viidestä palkista, tarkoittaa tämä, että kyseinen metodi ei palauttanut kyseiseen hakutehtävään yhtään oikeaa dokumenttia ensimmäisten 20 tuloksen joukossa. Palkeista nähdään, että kahdenkymmenen dokumentin kohdalla käsitehaun ja yhdistetyn käsite- ja sanahaun tarkkuus ei vielä poikkea merkittävästi puhtaan sanahaun vastaavasta. Useassa tapauksessa tulos on itse asiassa päinvastainen sanahaun saadessa huonomman tarkkuusarvon.

Jos tarkkuusarvoista lasketaan keskiarvot kaikkien hakutehtävien yli useilla eri palautettujen dokumenttien määrillä, voidaan helpommin arvioida eri menetelmien suorituskykyä. Tämä on esitetty seuraavassa kuvaajassa:



Kuva 18. *Keskiarvoinen tarkkuus tietyllä palautettujen dokumenttien määrällä*

Kuvassa 18 on esitetty tarkkuuden keskiarvo palautettujen dokumenttien määrän funktiona eri menetelmillä. Kuvaajassa sanahaku ja suosittelu suoriutuvat heikoimmin. Pienemmillä dokumenttimäärillä sana ja käsitehaun yhdistelmä, sekä sanahaun ja suosittelun yhdistelmä omaavat korkeimman tarkkuuden, mutta sadan dokumentin kohdalla ja siitä eteenpäin käsitehaku nousee sanahaun ja suosittelun ohi.

Tuloksiin pitää kuitenkin suhtautua lähinnä suuntaa-antavina, koska edellä mainittu laskutapa, jossa puuttuvat dokumentit oletetaan vääriksi, vääristää tarkkuusarvoa etenkin suurilla dokumenttimäärillä. Kuitenkin pienillä dokumenttimäärillä, jotka vastaavat todellista käyttäjätilannetta, jossa käyttäjä silmäilee ensimmäiset 10-30 tulosta, voidaan tuloksia pitää merkittävinä.

8.2.5 Pohdintaa tuloksista

Ensimmäinen silmiinpistävä seikka tuloksissa on tarkkuusarvon pienuus kaikilla käytetyillä menetelmillä. Tämä selittyy sillä, että hakutehtävät olivat vaikeita suorittaa pelkän otsikon avulla. Koska käytettävissä ei ollut sovellusta, joka muokkaisi hakutehtävien luonnollisen kielen kuvauksista sopivia hakusyötteitä, käytettiin otsikkoja sellaisinaan ja useissa tapauksissa ne eivät yksinään rajaa hakua kovinkaan tehokkaasti. Sinällään tilanne kuitenkin vastaa todellisuutta, jossa käyttäjä usein muodostaa ensimmäisen kyselynsä varsin nopeasti ja mahdollisesti epätäydellisenä ja muokkaa sitä vasta saatujen hakutulosten perusteella.

Ehkäpä keskeisin kysymys yllä olevia tuloksia pohdittaessa on kuinka suurena ongelmana nähdään se, että menetelmä palauttaa dokumentteja silloinkin, kun yksikään niistä ei ole

relevantti. Jos järjestelmä ei tällaisessa tapauksessa palauta yhtäkään dokumenttia, ei käyttäjältä mene aikaa turhien tulosten tarkastelemiseen, vaan tämä voi välittömästi suorittaa uuden haun. Toisaalta käyttäjä ei yleensä ole tiedonhaun ammattilainen ja saattaa muodostaa haun, joka ei suoraan vastaa käyttäjän tarpeita, jolloin tietokannan lähimmin hakua vastaavat dokumentitkin saattavat olla kiinnostavia. Suosittelu periaatteessa ohittaa tämän ongelman, jos suosittelun tulokset esitetään erillään itse haun tuottamista tuloksista. Tällöin käyttäjä voi itse päättää onko kiinnostunut tarkkaa hakua mahdollisesti sivuavista suositteluista vai ei.

Riippumatta vastauksesta em. kysymykseen, pärjäsi yhdistetty sanahaku ja suosittelu hyvin kaikilla käytetyillä mittareilla. Jos sanahakua pidetään vertailukohtana muille, tämän projektin puitteissa kehitetyille menetelmille, voidaan suosittelun lisäämistä siihen pitää selvänä parannuksena. Saanti paranee merkittävästi tarkkuuden kuitenkaan kärsimättä.

Käsitehaku ei yksinään sovellu korvaamaan sanahakua, mutta tämän rinnalla se voi joissain tapauksissa tuottaa lisäarvoa. Tarkkuuden voimakkaan laskemisen voidaan kuitenkin katsoa jättävän tämän menetelmän hyödyt varsin epävarmoiksi. Toisaalta on huomattava, että tutkimus suoritettiin vain käyttäen Yleistä suomalaista ontologiaa ja yleiseen sanomalehtiaineistoon. Spesifimmällä ontologialla ja aineistolla, esimerkiksi urheiluontologialla ja pelkkiin urheilu-uutisiin testaamalla olisi tarkkuutta todennäköisesti mahdollista nostaa merkittävästikin.

Ontologian lisäksi toinen muuttuja esitellyssä järjestelmässä on rikastamiseen käytetyt polut, joita lyhentämällä voidaan käsitehaun tuloksia rajoittaa ja näin oletettavasti nostaa tarkkuutta saannin kustannuksella. Erilaisia polkukonfiguraatioita ei kuitenkaan testattu.

9 YHTEENVETO

9.1 Yhteenveto työstä

Informaation lisääntyessä yhteiskunnassa vaaditaan tehokkaampia tapoja jäsentää ja hakea tietomassasta oleellisin tieto kulloiseenkin tarpeeseen. Koska tiedonhaku on siirtynyt ammattilaisilta koskettamaan oleellisesti myös tavallisia käyttäjiä, on luonnollinen ratkaisu lähteä kehittämään automaattisia tiedonhaun keinoja, jolloin vaaditaan koneymmärrettävää dataa. Semanttisen webin tekniikat pyrkivät vastaamaan tähän haasteeseen.

Semantiikka mahdollistaa myös tehokkaamman resurssien linkittämisen ja yhteistyön eri organisaatioiden välillä. Sanomalehtien levikit ovat olleet laskussa jo jonkin aikaa, mutta tarvetta ammattilaisten jäsentämälle informaatiolle tulee aina olemaan ja semanttisen webin tekniikoilla tätä prosessia voidaan kehittää ja automatisoida ja sen hyödyntämistä voidaan yksinkertaistaa loppukäyttäjän näkökulmasta. Semanttisen webin teknologioiden yhdistäminen perinteisiin tiedonhakualan kehityssuuntiin kuten kyselyn- ja dokumentin laajennukseen vaikuttaa lupaavalta yhdistelmältä näiden tavoitteiden saavuttamisessa.

Tämän työn puitteissa tehtiin sanomalehtiarkistoille automaattinen, ontologiaperustainen annotointi- ja hakusovellus Airo, joka suorittaa dokumentin laajennusta ontologisilla käsitteillä. Käsitteet rikastetaan käsiteklusteroinnilla, jossa käsitteen esiintyminen dokumentissa

nostaa ontologisessa hierarkiassa läheisten käsitteiden painoja. Hakutilanteessa hakusanat käsitteistetään, jolloin voidaan suorittaa sanahakua tukeva käsitehaku samanaikaisesti.

Järjestelmän evaluaatio suoritettiin CLEF Test Suitella ja tulokset osoittivat, että käsitehaku yhdistettynä perinteiseen sanahakuun laskee haun tarkkuutta, mutta nostaa saantia. Sen sijaan tarkkuus ensimmäisten 20 tuloksen suhteen paranee, kun käsitehaku lisätään sanahakuun. Mielenkiintoisin tulos oli kuitenkin suosittelu, joka on hybridimenetelmä dokumentin laajennuksesta ja kyselyn laajennuksesta. Suosittelussa sanahaun relevanteimmista tulosdokumenteista eristetään niille yhteiset käsitteet ja näillä suoritetaan uusi haku aineistoon. Saadusta tulosjoukosta pudotetaan alkuperäiseen hakuun vastanneet dokumentit ja jäljelle jääneistä korkeimman relevanssin saaneet kymmenen dokumenttia esitetään käyttäjälle. Tämän menetelmän yhdistäminen sanahakuun paransi sekä saantia, että tarkkuutta, verrattuna pelkkään sanahakuun.

Luotu järjestelmä on ontologiariippumaton ja jokaisen ontologian tuottamat käsitteistykset talletetaan omaan indeksiinsä. Testattavana ontologiana käytettiin Yleistä suomalaista ontologiaa ja spesifimpien ontologioiden, kuten esimerkiksi paikkaontologian testaus jää tulevaisuuden haasteeksi. Myös käsiteklusteroinnin laajuutta on mahdollista säätää hyvinkin monipuolisesti, mutta erilaisten konfiguraatioiden merkitystä ei tässä työssä tutkittu.

9.2 Muuta tutkimusta

Kaksi espanjalaista yliopistoa, Universidad Autónoma de Madrid ja Universitat de Lleida, kehittävät Neptuno-järjestelmää, joka soveltaa semanttisen webin teknologioita digitaalisiin sanomalehtiarkistoihin /46/. Järjestelmää varten kehitettiin oma uutisaineisto-ontologiansa, johon on integroitu IPTC:n NewsCodesin (ks. 6.7.4) aiempi versio, Subject Reference System. Ontologiaa käytetään artikkelien luokitteluun, jonka arkistoijat suorittavat manuaalisesti, ja yksi sen keskeisimmistä hyödyistä on toimia yhdistävänä tekijänä arkistohenkilökunnan ja reportterien erilaisessa ajattelutavassa artikkelien indeksointiin ja hakuun liittyen. /46/

Neptunoon on myös kehitetty semanttiset haku- ja navigaatiotoiminnot. Semanttinen haku on järjestelmässä toteutettu siten, että käyttäjä voi hakusanojen sijaan etsiä ontologisilla käsitteillä, joilla arkiston artikkelit on annotoitu, ja tuloksena voi olla kokonaisten dokumenttien sijaan myös niiden osia, jotka on merkitty käsitteen kannalta oleellisiksi. Haku on myös mahdollista suorittaa suoraan IPTC-taksonomian mukaisiin kenttiin, jolloin tuloksena on IPTC:n suositusten mukaisesti jäsennetty selailu arkistoon. Navigointia varten järjestelmään on myös kehitetty erillinen visualisointiontologia, johon on sisällytetty luokitteluontologian sisältämä, visualisoinnin kannalta oleellinen osa. Siitä on siis jätetty pois suuri määrä sellaista informaatiota, joka peruskäyttäjän näkökulmasta hankaloittaa ontologiapohjaista aineiston selailua. /46/

Airo-järjestelmään verrattuna Neptunon keskeisin ero on sen manuaalisuus, joka mahdollistaa aivan erilaisen lähestymistavan moniin keskeisiin ongelmiin, mutta toisaalta vaatii merkittävää työpanosta toimituksen arkistoijilta. Neptunossa käytetty ontologia on myös kehitetty varta vasten sanomalehtitoimituksen tarpeisiin ja uutisten kategorisointiin, jolloin

varsinaista käsitteistystä ei tehdä, vaan ontologiaa käytetään ennemminkin laajempaan luokitteluun.

EU:n IST (Information Society Technologies) -ohjelman puitteissa kehitetty NEWS-projekti /50/ sen sijaan sisältää ontologiaa hyödyntävän hakujärjestelmän lisäksi myös automaattisen annotaatiokomponentin. Järjestelmä tuottaa automaattisesti IPTC:n NewsCodes-luokitukset ja tunnistaa artikkelista henkilöt, organisaatiot ja paikat käyttäen sekä lingvistisiä, että tilasto-ominaisuuksia. Luonnollisen kielen prosessoinnin järjestelmässä hoitaa Ontology Ltd., ja tarkemmin toiminta perustuu syntaktisten roolien tunnistukselle (jäsennyspuu), stemmaukselle, monisanaisten käsitteiden löytämiselle tekstistä sekä käsiteluokille, joissa tietyssä kontekstissa esiintyvät käsitteet tulkitaan yksittäisten termien lisäksi myös laajempana kokonaisuutena. Vastaavia tekniikoita sovelletaan myös multimediaaineistolle hyödyntäen olemassa olevan metadatan rikastamista sekä videoiden ja nauhoitusten tapauksessa puheentunnistusta. /50/

Tietokannassaan NEWS käyttää hakukoneena Lucenea, minkä lisäksi hakujärjestelmään sisältyy päättelykone, joka toimii NEWS-ontologian sisällä. Kyseinen ontologia sisältää kolme moduulia, joista ensimmäinen perustuu IPTC:n NewsCodesiin, toinen sisältää toimituksellista metadataa ja kolmas sisältää uutisaineiston annotointiin käytettäviä käsitteitä. Tämä kolmas moduuli koostuu 200 luokasta, kuten kaupungit, maat, yhtiöt, henkilöt jne., joista on luotu yli 11000 yksittäistä instanssia. Annotointiongelmaa on siis lähestytty eri tavalla kuin Airossa ja käsitteet on rajoitettu pieneen määrään luokkia, joista luodaan instansseja. Varsinainen tarkka artikkelien käsitteistys on jätetty tekemättä. /50/

Instanssien välinen disambiguointi on NEWS-järjestelmässä toteutettu nojaten kahteen pääperiaatteeseen: semanttiseen koherenssiin, joka vastaa pääpiirteissään Airon käsiteklusterointia ja uutistrendeihin, eli verrataan instanssia muihin, ajallisesti läheisiin uutisiin. NEWS-järjestelmän semanttinen koherenssi eroaa Airon käsiteklusteroinnista siinä, että se nojaa voimakkaasti aiempiin artikkeleihin ja näiden annotaatioihin, eikä niinkään ontologiseen informaatioon. Menetelmällä päästiin 80 %:n tarkkuuteen mitä tuli Georgian sijoittamiseen joko Aasiaan tai Yhdysvaltoihin. /50/

Bulgariassa kehitetty KIM on semanttinen indeksointi-, annotointi- ja hakujärjestelmä /57/, jonka keskeinen toiminnallisuus on nimettyjen entiteettien tunnistaminen. KIM pyrkii tunnistamaan ontologiaan määriteltyjä käsitteiden instansseja tekstistä ja jopa täydentämään ontologiaa uusilla instansseilla. Tämän se tekee sääntöpohjaisesti eri instanssityypeille (esim. paikat, organisaatiot tai henkilöt) luoduilla tunnistimilla. /57/

Disambiguoinnin KIM suorittaa eri instanssityyppien välillä sanalistavihjeiden perusteella, jolloin esimerkiksi Jack London tunnistetaan henkilöksi paikan sijasta, koska Jack on tunnettu henkilön nimi. Sen sijaan samannimisten instanssien välillä tehtävästä disambiguoinnista ei mainita, eli ei esimerkiksi selvitetä Amsterdam Alankomaalaiseen vai Yhdysvaltalaiseen kaupunkiin. /57/

Airoa varten kehitettyä hahmokieltä vastaa etäisesti SPARQLeR (Simple Protocol and RDF Query Language extended with Regular paths), joka on RDF-kyselykieli SPARQL:n /47/ laajennus ja mahdollistaa kahden resurssin välisten polkujen etsimisen ontologiasta /48/. Verrattuna Airo-järjestelmää varten kehitettyyn hahmokieleen, on SPARQLeR taval-

laan käänteinen. Hahmokieli hakee tiettyä polkua ja alkupistettä vastaavat resurssit kun taas SPARQLeR hakee polut, jotka muodostavat kahden tietyn resurssin väliset yhteydet.

9.3 Jatkotutkimus

Airon kehitystä aiotaan jatkaa osana Semantic Web 2.0-hanketta /76/. Käsiteklusteroinnin ideaa aiotaan kehittää ja siihen liittyviä ongelmia korjata. Etenkin hahmojen mielivaltaisuuden ongelmaa aiotaan tutkia ja kehittää CLEF-aineiston pohjalta automaattinen hahmooptimoija, joka testaa erilaisia hahmomahdollisuuksia ja valitsee niistä parhaan. Tämä antanee suuntaa hahmojen luomiseen myös muihin aineistoihin ja ontologioihin.

LÄHDELUETTELO

/1/ **Hyvönen, E.** *Miksi Asiasanastot eivät riitä vaan tarvitaan ontologioita?* TKK Viestintätekniikka ja Helsingin Yliopisto, tietojenkäsittelytieteen laitos, 10.10.2005 [viitattu 26.6.2006]. Saatavilla WWW-muodossa <URL: http://www.seco.tkk.fi/publications/2005/hyvonen-miksi-asiasanastot-eivat-riita-2005.pdf>

/2/ **Manola, F., Miller, E. & McBride, B.** *RDF Primer.* W3C Recommendation, 10.2.2004 [viitattu 27.6.2006]. Saatavilla WWW-muodossa <URL: http://www.w3.org/TR/rdf-primer/>

/3/ **Antoniu, G. & van Harmelen, F.** *A Semantic Web Primer.* Massachusetts Institute of Technology, The MIT Press, Cambridge, Massachusetts, USA, 2004. 238s.

/4/ **Jacobs, I.** *About the World Wide Web Consortium (W3C).* W3C 2004 (online), päivitetty 14.6.2006 [viitattu 27.6.2006]. Saatavilla WWW-muodossa <URL: http://www.w3.org/Consortium/>

/5/ **Gruber, T.** *A Translation Approach to Portable Ontology Specifications.* Knowledge Systems Laboratory, Computer Science Department, Stanford University, Syyskuu 1992, päivitetty huhtikuussa 1993 [viitattu 28.6.2006]. Saatavilla WWW-muodossa <URL: http://ksl-web.stanford.edu/KSL_Abstracts/KSL-93-04.html>

/6/ **Klyne, G., Carroll, J. & McBride, B.** *Resource Description Framework (RDF): Concepts and Abstract Syntax.* W3C 2004 (online) [viitattu 28.6.2006]. Saatavilla WWW-muodossa <URL: http://www.w3.org/TR/rdf-concepts/>

/7/ **McGuinness, D. & van Harmelen F.,** *OWL Web Ontology Language Overview.* W3C Recommendation 10.2.2004 [viitattu 28.6.2006. Saatavilla WWW-muodossa <URL: http://www.w3.org/TR/owl-features/>

/8/ **Connolly D., van Harmelen F., Horrocks, I., McGuinness, D., Patel-Schneider, P. & Stein, L.** *DAML + OIL (March 2001) Reference Description.* W3C Note 18.12.2001 (online) [viitattu 29.6.2006]. Saatavilla WWW-muodossa: <URL: http://www.w3.org/TR/daml+oil-reference>

/9/ **Smith, M., Welty, C. & McGuinness, D.** *OWL Web Ontology Language Guide.* W3C Recommendation 10.2.2004 [viitattu 5.7.2006]. Saatavilla WWW-muodossa: <URL: http://www.w3.org/TR/owl-guide/>

/10/ **Antoniu, G. & van Harmelen, F.** *Web Ontology Language: OWL.* (online) [viitattu 7.6.2006]. Saatavilla WWW-muodossa: <URL: http://jeogoodjob.250free.com/data/OntoHandbook03OWL.pdf>

/11/ **Kauppinen, T., Ruotsalo, T., Salminen, M.** *Tietämyksen mallintaminen semanttisessa webissä.* TKK Viestintätekniikka ja Tietojenkäsittelytieteen laitos, Helsingin Yliopisto, Systeemityö-lehti 4/2005. Saatavilla WWW-muodossa: <URL:

http://www.seco.tkk.fi/publications/2005/kauppinen-ruotsalo-salminen-tiedon-mallintaminen-semanttisessa-webissa-2005.pdf>

/12/ *Protege*. Stanford Center for Biomedical Informatics Research, 2007. <URL: http://protege.stanford.edu/overview/>

/13/ **Efthimiadis, E.** *Query Expansion.* Annual Review of Information Systems and Technolgy (ARIST), v31, pp 121-187, 1996. Saatavilla WWW-muodossa: <URL: http://faculty.washington.edu/efthimis/pubs/Pubs/qe-arist/QE-arist.html>

/14/ **Spoerri, A.** *InfoCrystal – A Visual Tool for Information Retrieval.* Ph.D. Research and thesis, MIT, 1995. Saatavilla WWW-muodossa: <URL: http://www.scils.rutgers.edu/~aspoerri/InfoCrystal/InfoCrystal.htm>

/15/ **Marcus, R.** *The RIAO 94 Conference and the Status of Information Retrieval.* Laboratory for Information and Decisions Systems, MIT, 1994. Saatavilla WWW-muodossa: <URL: http://delivery.acm.org/10.1145/200000/195500/p7-marcus.pdf?key1=195500&key2=6913783711&coll=GUIDE&dl=GUIDE&CFID=16900669&CFTOKEN=23220893>

/16/ **Fox, E., Sharan, S.** *A Comparison of Two Methods for Soft Boolean Operator Interpretation in Information Retrieval.* Technical Report TR-86-01, Computer Science, Virginia Polytechnic Institute and State University, 1986. Saatavilla WWW-muodossa: <URL: http://eprints.cs.vt.edu/archive/00000008/01/TR-86-01.pdf>

/17/ **Salton, G., McGill, M.** *Introduction to modern information retrieval.* McGraw-Hill, New York, USA, 1986. 400s.

/18/ **Baeza-Yates, R., Ribeiro-Neto, B.** *Modern Information Retrieval.* Addison Wesley Longman Publishing Co. Inc., USA, helmikuu 1999. 513s.

/19/ **Saracevic, T.** *Relevance reconsidered.* School of Communication, Information and Library Science, Rutgers University, 1996. Saatavilla WWW-muodossa: <URL: http://www.scils.rutgers.edu/~tefko/CoLIS2_1996.doc>

/20/ **Copestake, A.** *Errors in wikis: new challenges and opportunities – a discussion document.* Computer Laboratory, University of Cambridge. Communications of the ACM, volume 48, issue 12, joulukuu 2005. Saatavilla WWW-muodossa: <URL: http://acl.ldc.upenn.edu/W/W06/W06-2802.pdf>

/21/ **Mitra, M., Singhal, A., Buckley, C.** *Improving Automatic Query Expansion.* Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, 1998.

/22/ **Salton, G., Buckley, C.** *Improving Retrieval Performance by Relevance Feedback.* Department of Computer Science, Cornell University, Ithaca. Journal of the American Society for Information Science, kesäkuu 1990.

/23/ **Hyvönen, E., Seppälä, K., Viljanen, K., Frosterus, M.** *Yleinen suomalainen ontologia YSO – kohti suomalaista semanttista webiä.* Semanttisen laskennan tutkimusryhmä (SeCo) TKK Viestintätekniikan laboratorio ja Helsingin yliopiston tietojenkä-

sittelytieteen laitos. Tietolinja, tammikuu, 2007. Saatavilla WWW-muodossa: <URL: http://urn.fi/URN:NBN:fi-fe20071264>

/24/ **Hyvönen, E., Saarela, S., Viljanen, K.** *Ontogator: Combining View- and Ontology-Based Search with Semantic Browsing.* Proceedings of XML Finland 2003, Open Standards, XML, and the Public Sector. Kuopio, lokakuu 2003. Saatavilla WWW-muodossa: <URL: http://www.seco.tkk.fi/publications/2003/hyvonen-saarela-viljanen-ontogator-combining-view-2003.pdf>

/25/ **Navigli, R., Velardi, P.** *An Analysis of Ontology-based Query Expansion Strategies.* Workshop on Adaptive Text Extraction and Mining, 14th European Conference on Machine Learning, Cavtat-Dubrovnik, Kroatia, syykuu 2003. Saatavilla WWW-muodossa: <URL: http://www.dcs.shef.ac.uk/~fabio/ATEM03/navigli-ecml03-atem.pdf>

/26/ **Gospodnetic, O.**, **Hatcher, E.** *Lucene in Action – A guide to the Java search engine.* Manning Publications Co., Greenwich, 2005. 421s.

/27/ **Gonzalo, J., Verdejo, F., Chugur, I., Cigarrán J.** *Indexing with WordNet synsets can improve text retrieval.* ARXIV, 1998. Saatavilla WWW-muodossa: <URL: http://acl.ldc.upenn.edu/W/W98/W98-0705.pdf >

/28/ **Voorhees, E.**, *Query Expansion using Lexical-Semantic Relations.* Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, 1994. Saatavilla WWW-muodossa: <URL: http://portal.acm.org/citation.cfm?id=188490.188508&dl>

/29/ **Stokoe, C., Oakes, M., Tait, J.** *Word Sense Disambiguation in Information Retrieval Revisited.* Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, 2003. Saatavilla WWW-muodossa <URL: http://portal.acm.org/citation.cfm?id=860466&dl>

/30/ **Gale, W., Church, K., Yarowsky, D.** *Estimating Upper and Lower Bounds on the Performance of Word-Sense Disambiguation Programs.* Proceedings of the 30th annual meeting on Association for Computational Linguistics, 1992. Saatavilla WWW-muodossa: <URL: http://portal.acm.org/citation.cfm?id=981999&dl>

/31/ **Johnstone, H.** *IPTC Spectrum 21.* Helmikuu, 2007. Saatavilla WWW-muodossa: <URL: http://www.iptc.org/download/mirror/IPTCSpectrum2006.pdf>

/32/ **Bacan, H., Pandric, I., Gulija, D.** *Automated News Item Categorization.* 19th Annual Conference of The Japanese Society for Artificial Intelligence JSAI, 2005. Saatavilla WWW-muodossa: <URL: http://www.ii.ist.i.kyoto-u.ac.jp/jsai2005ws/proceedings/bacan.pdf>

/33/ **IPTC.** *The IPTC NewsCodes – Metadata Taxonomies for the News Industry.* 2007. <URL: http://www.iptc.org/NewsCodes/>

/34/ **IPTC.** *News Markup Language.* 2005. <URL: http://www.newsml.org/>

/35/ **IPTC.** *News Industry Text Format – Introduction.* 2007. <URL: http://www.nitf.org/intro.php>

/36/ **IPTC.** *EventML 1.0 Business Requirements Draft 4.* 2004. <URL: http://www.iptc.org/std-dev/EventsML/1.0/specification/DRAFT-EventsML_1.0_spec_BusinessRequirements_4.pdf>

/37/ **Johnstone, H.** *IPTC Spectrum 19.* Tammikuu, 2005. Saatavilla WWW-muodossa: <URL: http://www.iptc.org/download/mirror/IPTCSpectrum2004.pdf>

/38/ **Brickley, D., Guha, R.** *RDF Vocabulary Description Language 1.0: RDF Schema.* W3C helmikuu 2004 (Online). Saatavilla WWW-muodossa: <URL: http://www.w3.org/TR/rdf-schema/>

/39/ **Holi, M., Hyvönen, E.** *A Method for Modeling Uncertainty in Semantic Web Taxonomies.* Proceedings of WWW2004, New York, Alternate Track Papers and Posters, toukokuu, 2004. Saatavilla WWW-muodossa: <URL: http://www.seco.tkk.fi/publications/2004/holi-hyvonen-a-method-for-modeling-2004.pdf>

/40/ **Hotho, A., Staab, S., Stumme, G.** *Wordnet improves Text Document Clustering.* ACM Sigir workshop on Semantic Web, SWIR 2003. ACM Press, New York, USA, 2003. Saatavilla WWW-muodossa: <URL: http://www.aifb.uni-karlsruhe.de/WBS/aho/pub/hothoetal_sigir_ws_sem_web.pdf>

/41/ **Ankolekar A., Seo, Y., Sycara, K.** *Investigating Semantic Knowledge for Text Learning.* ACM Sigir workshop on Semantic Web, SWIR 2003. ACM Press, New York, USA, 2003. Saatavilla WWW-muodossa: <URL: http://www.cs.cmu.edu/~anupriya/papers/sigir_03.pdf>

/42/ **Mayfield, J., Finin, T.** *Information Retrieval on the Semantic Web: Integrating Inference and Retrieval.* ACM Sigir workshop on Semantic Web, SWIR 2003. ACM Press, New York, USA, 2003. Saatavilla WWW-muodossa: <URL: http://www.cs.umbc.edu/pub/gopher/ftp/finin/papers/pox.pdf>

/43/ **IPTC.** *NewsCodes: SubjectCodes.* Saatavilla WWW-muodossa: <URL: http://www.iptc.org/NewsCodes/nc_ts-table01.php>

/44/ **OPA.** *OPA Research: Generational Media Study.* Syyskuu, 2004. Saatavilla WWW-muodossa <URL: http://www.online-publishers.org/media/136_W_opa_generational_study_sep04.pdf>

/45/ **Hölscher, C., Strube, G.** *Web Search Behaviour of Internet Experts And Newbies.* Computer networks 2000 vol: 33, s. 337.

/46/ **Castells, P., Perdrix, F., Pulido, E., Rico, M., Benjamins, R., Contreras, J., Lorés, J.** *Neptuno: Semantic Web Technologies for a Digital Newspaper Archive.* European Semantic Web Symposium 2004, s. 445-458, toukokuu, 2004. Saatavilla WWW-muodossa: <URL: http://www.springerlink.com/index/1DT41P13XGLMW00B.pdf>

/47/ **Prud'hommeaux, E., Seaborne, A.** *SPARQL Query Language for RDF.* W3C Candidate Recommendation, W3C, kesäkuu 2007. Saatavilla WWW-muodossa: <URL: http://www.w3.org/TR/rdf-sparql-query/>

/48/ **Kochut, K., Janik, M.** *SPARQLeR: Extended Sparql for Semantic Association Discovery.* The Semantic Web: Research and Applications, Springer Berlin/Heidelberg, 2007.

/49/ *Jena – A Semantic Web Framework for Java.* <URL: http://jena.sourceforge.net/>

/50/ **Fernández, N., Blázquez, J., Fisteus, J., Sánchez, L., Sintek, M., Bernardi, A., Fuentes, M., Marrara, A., Ben-Asher, Z.** *NEWS: Bringing Semantic Web Technologies into News Agencies.* The Semantic Web – ISWC 2006, s. 778-791, Springer Berlin/Heidelberg, 2006.

/51/ **Billerbeck, B., Zobel, J.** *Document Expansion versus Query Expansion for Ad-hoc Retrieval.* School of Computer Science and Information Technology, RMIT University, Melbourne, Australia, lokakuu 2005. Saatavilla WWW-muodossa: <URL: http://www.cs.rmit.edu.au/~jz/fulltext/adcs05.pdf>

/52/ **Bodner, R., Song, F.** *Knowledge-Based Approaches to Query Expansion in Information Retrieval.* Advances in Artificial Intelligence, 1996, Springer, New York. Saatavilla WWW-muodossa: <URL: http://www.imedia.mie.utoronto.ca/~rbodner/papers/bodner_cai96.ps>

/53/ *CLEF Test Suite for the CLEF 2000-2003 Campaigns -DVD*

/54/ **Buckland, M., Gey, F.** *The Relationship between Recall and Precision.* Journal of the American Society for Information Science, tammikuu 1994.

/55/ **Kalfoglou, Y., Schorlemmer, M.** *Ontology Mapping: State of the Art.* The Knowledge Engineering Review, 2003. Saatavilla WWW-muodossa: <URL: http://drops.dagstuhl.de/opus/volltexte/2005/40/pdf/04391.KalfoglouYannis.Paper.40.pdf>

/56/ **Wielinga, B., Schreiber, A., Wielemaker, J., Sandberg, J.** *From Thesaurus to Ontology.* University of Amsterdam, Social Science Informatics. International Conference On Knowledge Capture, 2001. Saatavilla WWW-muodossa: <URL: http://www.few.vu.nl/~guus/papers/Wielinga01a.pdf>

/57/ **Popov, B., Kiryakov, A., Manov, D., Kirilov, A., Ognyanoff, D., Goranov, M.** *Towards Semantic Web Information Extraction.* Proceedings of ISWC, USA, lokakuu 2003. Saatavilla WWW-muodossa: <URL: http://gate.ac.uk/conferences/iswc2003/proceedings/iswc2003-hlt4sw-proceedings.pdf#page=5>

/58/ **Miles, A., Brickley, D.** *SKOS Core Guide.* W3C Working Draft 2, maaliskuu 2005. <URL: http://www.w3.org/TR/swbp-skos-core-guide>

/59/ *Document Object Model (DOM) Level 1 Specification.* W3C Recommendation, lokakuu 1998. <URL: http://www.w3.org/TR/REC-DOM-Level-1/>

/60/ *PoweredBy.* Lucene-java wiki. <URL: http://wiki.apache.org/lucene-java/PoweredBy>

/61/ **Alm, O.** *Tekstidokumenttien automaattinen ontologiaperustainen annotointi.* Pro Gradu-tutkielma. Helsingin Yliopisto, syyskuu 2007. Saatavilla WWW-muodossa: <URL: http://www.seco.tkk.fi/publications/2007/alm-gradu-2007.pdf>

/62/ **Gilchrist, A.** *Thesauri, taxonomies and ontologies – an etymological note.* Journal of Documentation, nro 59, 2003.

/63/ **Sanchez, E.** *Fuzzy Logic and the Semantic Web.* Elsevier 2006.

/64/ **Kang, S., Lee, J.** *Semi-automatic practical ontology construction by using a thesaurus, computational dictionaries, and large corpora.* Proceedings of the workshop on Human Language Technology and Knowledge Management, Ranska, Toulouse, 2001. Saatavilla WWW-muodossa: <URL: http://portal.acm.org/citation.cfm?id=1118220.1118226>

/65/ **Semantic Computing Research Group.** *HealthFinland – a National Semantic Health Information Portal.* <URL: http://www.seco.tkk.fi/applications/tervesuomi/>

/66/ **Semantic Computing Research Group.** *CultureSampo – Finnish Culture on the Semantic Web.* <URL: http://www.seco.tkk.fi/applications/kulttuurisampo/>

/67/ **Semantic Computing Research Group.** *Poka – A Framework for automatic annotation.* <URL: http://www.seco.tkk.fi/tools/poka/>

/68/ *Cross Language Evaluation Forum.* <URL: http://www.clef-campaign.org/>

/69/ *Suomalaiset semanttisen webin ontologiat (FinnONTO).* <URL: http://akseli.tekes.fi/opencms/opencms/OhjelmaPortaali/ohjelmat/FENIX/fi/system/projekti.html?id=9112835&nav=Projekti>

/70/ *Sanoma Data.* <URL: http://www.sanomadata.fi/>

/71/ *The Finnish General Upper Ontology YSO.* <URL: http://www.seco.tkk.fi/ontologies/yso/>

/72/ *YSA – Yleinen suomalainen asiasanasto.* <URL: http://vesa.lib.helsinki.fi/ysa/>

/73/ **Jansen, B., Spink, A., Saracevic, T.** *Real life, real users and real needs: a study and analysis of user queries on the web.* Information Processing & Management, vol 36, issue 2, maaliskuu 2000.

/74/ **Pedrycz, W.** *Fuzzy clustering with a knowledge-based guidance.* Pattern Recognition Letters, vol 25, issue 4, maaliskuu 2004.

/75/ **Ruthven, I.** *Re-examining the potential effectiveness of interactive query expansion.* Annual ACM Conference on Research and Development in Information Retrieval, Toronto, Kanada, 2003. Saatavilla WWW-muodossa: <URL: http://portal.acm.org/citation.cfm?id=860475>

/76/ *Semantic Web 2.0.* <URL: http://www.seco.tkk.fi/projects/sw20/>

LIITTEET

1. Sovelluksen toiminta

Aineiston käsittely

8000 artikkelin testiaineisto toimitettiin epästandardina tekstitiedostona, jonka kaikkea informaatiota ei hyödynnetty Airon toiminnassa. Tälle tekstitiedostolle kirjoitettiin parseri, joka tuottaa Artikkeli-olioita, joihin on tallennettu kaikki Airon kannalta oleellinen informaatio kustakin artikkelista. Koska alkuperäisen tekstitiedoston parsiminen on verrattain pitkä operaatio, tehtiin Artikkeleja varten myös yksinkertainen tiedostoformaatti ja luokka, joka pystyy kirjoittamaan ja lukemaan näitä tiedostoja.

Artikkeli-olio koostuu joukosta kenttiä ja näiden get-, set- ja has-metodeista, joilla kenttien arvoja voidaan lukea ja muuttaa. Artikkeli sisältää seuraavat tiedot:

- otsikko, joka on yksinkertaisesti artikkelin pääotsikko
- koodi, joka on jokaiselle artikkelille yksilöllinen kirjain- ja numerosarja
- päivämäärä eli artikkelin julkaisupäivämäärä
- teksti, joka pitää sisällään artikkelin koko tekstin merkkijonomuodossa
- uritettuTeksti, joka sisältää artikkelista löytyneiden käsitteiden URI:t
- lemmattuTeksti, joka sisältää artikkelin tekstin lemmatussa muodossa ja näin mahdollistaa taivutettujen muotojen täsmäämisen eri muodossa esitettyihin hakuihin
- nimet, joka on artikkelista löytyneet, nimiksi tunnistetut merkkijonot sisältävä merkkijonovektori. Tällä hetkellä nimiksi tunnistetaan vain alkuperäisessä tekstissä jo valmiiksi nimiksi annotoidut merkkijonot.
- paikat, joka vastaa nimet-vektoria, mutta koskee tekstissä esiintyviä paikkoja

Edellä mainittu yksinkertainen tiedostoformaatti artikkelien tallentamiseksi käyttää vastaavaa rakennetta tiedon tallentamiseen. Artikkelit on talletettu tekstitiedostoon, jossa joka toinen rivi pitää sisällään määrittelyn seuraavan rivin sisällöstä. Nämä määrittelyt vastaavat Artikkeli-olion kenttien nimiä, mutta alkavat kolmella viivalla, esim. ”---otsikko”. Koska jokaisella artikkelilla pitää olla yksilöivä koodi, ilmoittaa ”---koodi” aina uuden artikkelin alun. Artikkeliformaatti olisi voitu toteuttaa myös esimerkiksi XML:llä, mutta tämä muoto valittiin sen yksinkertaisuuden vuoksi.

Airon sisältämä ArtikkeliLukijaKirjoittaja-luokka on tarkoitettu luodun tiedostoformaatin käsittelemiseen. Parserille on määritelty konfiguraatiokenttä, jossa ilmoitetaan yhteen tiedostoon talletettavien artikkelien määrä. Tämä siksi, että kaikkia artikkeleja ei välttämättä voida pitää yhtä aikaa keskusmuistissa, kun on kyse suuresta tietokannasta. ArtikkeliLukijaKirjoittaja pitää sisällään lähinnä kirjoitaArtikkelitTiedostoon- ja lueArtikkelitTiedostosta-metodit, jotka käsittelevät Artikkeli-vektoreita ja -tiedostoja.

Viimeinen aineiston käsittelyyn liittyvä Airon luokka on ArtikkeliHtmlKirjoittaja, joka nimensä mukaisesti luo artikkeleista html-tiedoston, jonka avulla näytetään hakujen tulos käyttäjälle. Jokaiselle päivälle luodaan oma html-tiedostonsa, joka sisältää sisäiset linkit yksittäisiin artikkeleihin niiden yksilöllisten koodien mukaan. Hakutulokset linkitetään näihin html-tiedostoihin, jotta hakuun vastanneita artikkeleita voidaan helposti tarkastella.

Kaiken kaikkiaan aineiston käsittely jätettiin varsin alkeelliseksi, koska tämä osa-alue muuttuu erittäin voimakkaasti riippuen siitä, mihin sovellusta käytetään. Esimerkiksi Sanomadata-casessa, jos Airo otettaisiin varsinaiseen käyttöön, siirrettäisiin artikkelit sovellukseen jollain muulla tavalla ja hakujen tulosartikkelit esitettäisiin olemassa olevan artikkelien selailujärjestelmän avulla.

Rikastus

Rikastus on Airon monimutkaisin osuus ja se sisältää suurimman määrän luokkia. Prosessin tarkoituksena on löytää tekstistä jotakin ontologiaa vastaavat käsitteet ja suorittaa näiden avulla dokumentin laajennus ja käsiteklusterointi. Tämän lisäksi hakutulosten selitykseen käytettävät luokat löytyvät rikastuksen alta.

Käsitteitä esittämään luotiin yksinkertainen UriAndStr-luokka, joka sisältää merkkijonomuotoisen uri-kentän, joka on tarkoitettu käsitteen URI:lle. Tämän lisäksi luokassa on double-muotoinen str-kenttä, johon voidaan tallettaa käsitteeseen liittyvä esiintymiskertojen määrä tai käsiteklusteroinnin jälkeinen lähikäsitteiden muokkaama painoarvo.

Käsiteklusteroinnin ja disambiguoinnin vaatimaa hahmokieltä varten luotiin Hahmo-luokka, jolla on sisäinen Polku-luokka, jolla puolestaan on sisäinen Askel-luokka. Hahmo sisältää Polku-vektorin, sekä metodit polkujen ja askelten lukemiseen kyseisen vektorin sisältä. Hahmon konstruktori ottaa argumentikseen merkkijonon hahmon sisältävään XML-tiedostoon, josta Hahmo-luokan sisäinen parseri luo vaadittavat Polku-oliot ja näiden sisältämät Askel-oliot. Näiden polkujen ja askelten ominaisuuksia käytetään kutsumalla Hahmo-luokan metodeja, joille annetaan argumentiksi halutun Polku-olion ja tämän sisältämän Askel-olion järjestysluvut. Esimerkiksi haluttaessa polun X askeleen Y ominaisuus, kutsutaan Hahmo-olion getPolunAskeleenOminaisuus-metodia argumenteilla X ja Y, joka sitten tuottaa merkkijonomuodossa kyseisen ominaisuuden.

Keskeisin osa käsitteistämisessä on Pokaa hyödyntävä SanomaExtractor, joka perii Pokan GeneralExtractor-luokan. Kyseessä on siis varsinainen eristin, joka ottaa vastaan merkkijono-muotoista dataa ja tuottaa siitä löydettyjä termejä vastaavat käsitteet. Tähän tarkoitukseen SanomaExtractor sisältää kolme metodia: tuotaUritus, tuotaUrit sekä viimeisenä tuotaUritusJaKirjaaUritusHtmlksi. Ensimmäistä metodia käytetään, kun halutaan tuottaa joukko URI:eita indeksointitarkoitukseen sijoitettavaksi edellä mainittuun Artikkeli-luokan uritettuTeksti-kenttään. TuotaUrit-metodi on vastaavanlainen, mutta tuottaa tulokseksi merkkijonon sijaan HashSet-olion, jonka sisällä on HashSet-olioita, joiden sisällä merkkijonot sijaitsevat. Alimmalla tasolla sijaitsevat käsitteiden URI:t merkkijonoina, joissa yhden HashSetin kaikki merkkijonot viittaavat aina samaan termiin. Toisin sanottuna, tämä metodi ei siis käytä disambiguaatiota, vaan listaa kaikki tiettyä termiä vastaavat mahdolliset käsitteet. Yhteenvetona voidaan sanoa, että ylin taso on joukko termejä, joista jokaista vastaa yksi tai useampi käsite, joilla jokaisella on yksiselitteinen URI. Viimeinen eristys-

metodi, tuotaUritusJaKirjaaUritusHtmlksi vastaa ensimmäistä, tuotaUritus-metodia, mutta tuottaa lisäksi html-muotoisen selitetiedoston, josta tuonnempana.

Tarkemmin eriteltynä tuotaUritus-metodi ottaa argumenteikseen eristettävän tekstin merkkijonomuodossa, sekä Hahmo-oliot disambiguointia ja lopullista rikastusta varten. Metodia kutsuttaessa luodaan myös uudet Disambiguoija- ja Rikastaja-oliot, joka tyhjentää edelliseen erityksen liittyvät tiedot SanomaExtractorin kentistä. Itse eristys noudattaa Pokan toimintaa, mutta termit, jotka vastaavat useampaa käsitettä ja näin ollen tuottavat enemmän kuin yhden URI:n, heitetään Disambiguoijalle ja vain varmat annotoinnit lisätään suoraan SanomaExtractorin loydetytUrit-kenttään, joka on UriAndStr-vektori.

Yksiselitteiset termit rikastetaan Rikastaja-luokan oliolla, jonka konstruktorille on olion alustusvaiheessa annettu rikastuksessa käytettävän ontologian Jena-Model. Itse rikastukseen käytetään rikastaUriHahmolla-metodia, mikä ottaa argumenteikseen rikastettavan UriAndStr-olion ja Hahmon, jolla rikastus suoritetaan. Rikastus etenee yksi hahmon polku kerrallaan, josta saadaan kulloiseenkin askeleeseen käytettävä predikaatti, rikastuksen syvyys, sekä rikastuksen suunta (objektin tai subjektin mukaan).

Rikastuksessa haetaan Jenalla valitusta ontologiasta kaikki ne kolmikot, joissa esiintyy haluttu predikaatti ja rikastettava URI subjektina (tai objektina, jos rikastus suoritetaan objektin mukaan). Tulokseksi saadaan joukko kolmikkoja, joiden ainoana erona keskenään on niissä esiintyvä objekti. Nämä objektit toimivat seuraavassa vaiheessa rikastettavina URI:eina ja toimenpide toistetaan syvyyden määräämä määrä kertoja jokaiselle polun ominaisuudelle. Jos rikastettava polku on inklusiivinen, eli rikastuksessa painotetaan jokaista polun varrella esiintyvää käsitettä, otetaan jokaisessa vaiheessa tulosjoukko talteen HashSettiin. Ei-inklusiivisen polun ollessä kyseessä, on tulosjoukko vain viimeisen ominaisuuden jälkeinen objektijoukko. Molemmissa tapauksissa joukosta poistetaan alkuperäinen, rikastettava URI. Tulosjoukon käsitteet saavat lisää painoa polun painon verran kerrottuna alkuperäisen URI:n esiintymiskertojen neliöjuurella. Kun tämä toimenpide on suoritettu kaikille hahmon poluille, palauttaa metodi UriAndStr-vektorin, jossa on lopulliset, yhden käsitteen rikastusta vastanneet URI:t painoineen.

Rikastaja-luokalla on myös yhdistaUrit-metodi, jolla voidaan yhdistää kaksi UriAndStr-vektoria. Käytännössä tämä tapahtuu siten, että ensimmäinen vektori otetaan tulosvektoriksi ja toisen sisältämät URI:t käydään läpi yksitellen ja niitä verrataan toisen vektorin URI:eihin. Jos URI löytyy molemmista joukoista, muutetaan tulosvektoriin sen paino vastaamaan yhdistettävien URI:en yhteenlaskettua painoa. Jos URI löytyy vain toisesta joukosta, lisätään se tulosvektoriin sellaisenaan. Lopuksi metodi palauttaa yhdistetyn UriAndStr-vektorin.

Yksiselitteisistä termeistä saatujen, rikastettujen käsitteiden avulla Disambiguoija arvaa moniselitteisiä termejä vastaavat käsitteet. Käytännössä tämä tapahtuu siten, että SanomaExtractor kutsuu Disambiguoijan lisaaDisambiguoitava-metodia aina kohdatessaan moniselitteisen termin, jolloin argumenttina annettava merkkijono-HashSet lisätään Disambiguoijan disambiguoitavat-kenttään. Tämä kenttä on Dis-vektori, joka on Disambiguoijan sisäinen luokka. Dis-olioilla on merkkijonovektorissa URI:t, joiden välillä disambiguointi suoritetaan, sekä näitä vastaavan termin esiintymiskerrat kokonaislukukentässä. Luokka

sisältää vertailumetodin, jonka avulla voidaan disambiguoitavat-kentän sisällä olevien Dis-olioiden esiintymiskertoja lisätä, silloin kun SanomaExtractor löytää uuden disambiguoitavan termin, joka vertailun perusteella vastaa jo olemassa olevaa Dis-oliota.

Disambiguoijan disambiguoi-metodi ottaa argumentikseen UriAndStr-vektorin, johon on talletettu yksiselitteisten käsitteiden perusteella rikastettu joukko käsitteitä, joiden perusteella disambiguointi suoritetaan. Käytännössä disambiguointi tapahtuu yksinkertaisesti siten, että disambiguoitavat-kentän Dis-oliot käydään yksitellen läpi ja näiden URI:ista valitaan rikastuksessa suurimman painon saanut. Jos kaikki tai osa URI:ista on saanut saman painon (tyypillisimmin tällöin nolla), valitaan kaikki saman painon saaneet käsitteet. Tuloksena saadaan UriAndStr-vektori, jossa URI:ina ovat disambiguoinnin tuloksena saadut käsitteet ja jonka painoihin sijoitetaan Dis-olioilta saadut esiintymiskerrat.

SanomaExtractorissa yhdistetään disambiguoidut URI:t ja rikastamattomat, artikkelista löytyneet, yksiselitteiset URI:t ja tälle joukolle suoritetaan lopullinen rikastus rikastushahmolla. Koska disambiguointirikastuksessa saadut painot eivät vaikuta tähän toiseen rikastukseen millään tavalla, ei hahmojen tarvitse muistuttaa toisiaan. Välissä voitaisiin jopa vaihtaa ontologiaa, jos niin haluttaisiin. Jälkimmäisen, lopullisen rikastuksen tuloksena saadut painot pyöristetään lähimpään kokonaislukuun ja URI:t tulostetaan merkkijonoon tämä määrä kertoja. Tämä merkkijono palautetaan ja se toimii Artikkeli-olion uritettuTekstinä.

Indeksointi ja haku

Airon indeksointi- ja hakuominaisuudet keskittyvät Hakulaite-luokan ympärille, joka tarjoaa Lucene-arkkitehtuuria hyödyntävän hakukonetoteutuksen Artikkeli-olioille. Hakulaitteen konstruktori ottaa argumenteikseen indeksin tiedostopolun ja totuusarvon siitä tehdäänkö kyseiseen kansioon uusi indeksi vai hyödynnetäänkö siellä jo olevaa indeksiä.

Airon pääluokka, SemSearch, sisältää teeIndeksi-metodin, joka käyttää SemSearchin konstruktorissa määriteltyä Hakulaite-oliota. Argumentteina metodille annetaan tiedostopolut indeksoitavaan aineistoon sekä hakemistoon, johon ArtikkeliHtmlKirjoittajan (ks. 7.4.1) halutaan tallettavan tuottamansa HTML-tiedostot. Itse metodi lukee ensin edellä kuvatulla parserilla artikkelit tiedostoihin, joista ne indeksointivaiheessa luetaan ja lisätään hakuindeksiin yksitellen kutsumalla Hakulaitteen lisaaArtikkeli-metodia, joka muuttaa Artikkelin Lucene-Document-muotoon ja indeksoi sen. Pokan tarjoamia lemmatisointiominaisuuksia hyödyntämällä indeksiin talletetaan myös artikkelin teksti lemmatussa muodossa.

Hakulaitteen hae-metodi ottaa argumentikseen SanomaExtractorin tuotaUrit-metodilla tuotetut hakukäsitteet HashSet-HashSet-merkkijonomuodossa (ks. 7.4.2), merkkijonovektorin jossa ovat alkuperäiset hakusanat, hakutuloksia rajoittavat alku- ja loppupäivämäärät, jolla välillä tulokset halutaan sekä viimeisenä totuusarvon siitä halutaanko tulokset järjestettävän ajan vai relevanssin mukaan. Tuloksenaan metodi palauttaa DocumentAndScore-vektorin, johon on talletettu tulokset halutussa järjestyksessä. DocumentAndScore on yksinkertainen luokka, jonka olioilla on Document-kenttä ja double-muotoinen relevanssikenttä.

Metodi muodostaa kaksi Lucene-hakua, toisen käsitteiden ja toisen sanojen mukaan. Hauille tehdään RangeFilter alku- ja loppupäivämääristä, joka rajoittaa Lucenen tuottamat tulok-

set näiden päivämäärien välille. Käsitehaku muodostetaan indeksin uritettuteksti-kenttiin siten, että yhtä hakutermiä vastaavat käsitteet, eli yhden HashSetin sisältämät merkkijonot, liitetään yhteen Lucenen Should-määreillä, eli leikkauksena. Lopullinen käsitekysely saadaan, kun muodostetuista leikkauksista tehdään unioni Lucenen Must-määreillä. Sanahaun kysely muodostetaan yksinkertaisemmin vain lemmaamalla hakusanat ja liittämällä ne yhteen unioniksi, joka kohdistetaan indeksoitujen dokumenttien lemmatun tekstin kenttään. Lopuksi metodi palauttaa yhdistetyn haun tulokset vektorimuodossa.

Hakulaite tallettaa myös edellisen haun tuottamat tulokset pelkälle sanahaulle ja pelkälle käsitehaulle. Nämä voidaan hakea pääasiallisen haun lisäksi ja näin ollen Hakulaitetta voidaan käyttää esimerkiksi puhtaaseen sanahakuun.

Hakulaite tarjoaa myös staattisen metodin muutaHitsDocumentAndScoreMuotoon, jolla voidaan siirtyä Lucenen hakutulosolioista Airon yleisemmin käyttämään DocumentAndScore-muotoon. Metodi palauttaa sille annettua Hits-oliota vastaavan DocumentAndScore-vektorin.

Suosittelu

Suosittelua varten Airossa on Suosittelija-luokka, joka sisältää suosittele- ja suositteleRajoitetusti-metodit. Näiden metodien erona on, että jälkimmäinen rajoittaa suosittelua vaatimalla, että alkuperäisiä hakutermejä vastaavien käsitteiden on esiinnyttävä tuotetuissa suositteluissa. Molemmat metodit tuottavat tuloksekseen DocumentAndScore-vektorin ja ottavat argumenteikseen alkuperäisen haun tuloksena saadun DocumentAndScore-vektorin, suosittelun aikaikkunan kokonaislukuna sekä Hakulaite-olion. Jälkimmäinen metodi ottaa lisäksi HashSetin, joka sisältää termejä vastaavat käsitteet merkkijonomuodossa omissa HashSeteissään.

Kumpikin metodi valitsee tuloksista maksimissaan kymmenen alkuperäisessä haussa suurimman relevanssin saanutta artikkelia ja etsii näille yhteiset käsitteet. Koska yksi dokumentti voi sisältää saman käsitteen monta kertaa, käydään yhden dokumentin kaikki käsitteet yksitellen läpi ja sijoitetaan ne merkkijono-HashSettiin. Tämä HashSet käydään läpi ja sen sisältämät käsitteet sijoitetaan lopulliseen, kaikkien suositteluun osallistuvien dokumenttien käsitteille tarkoitettuun HashSettiin. Ennen lisäämistä kuitenkin tarkistetaan esiintyikö käsite jo kyseisessä setissä ja jos esiintyi, lisätään käsite suositteluun menevään merkkijono-HashSettiin.

Suosittelukäsitejoukon selvittämisen lisäksi metodit laskevat aikaikkunan, eli suosittelun alku- ja loppupäivämäärät, joilla lopulliset tulokset suodatetaan. Dokumenttien käsitteiden käsittelyn yhteydessä kirjataan muistiin myös varhaisimman ja myöhäisimmän artikkelin päivämäärätieto. Argumenttina saadulla ajallisella maksimietäisyydellä lasketaan taaksepäin varhaisimmasta ja eteenpäin myöhäisimmästä artikkelipäivämäärästä ja lopputuloksena saadaan alku- ja loppupäivämäärät suodatusta varten.

Itse suosittelun suorittaa Hakulaite-olio haeSuositteluKasitesarjalla- tai haeRajoitetuSuositteluKasitesarjalla-metodilla. Näiden toiminta vastaa edellä esitellyn hae-metodin toimintaa sillä erotuksella, että suosittelukäsitteet yhdistetään yhdeksi kyselyksi OR-operaattoreilla. Jälkimmäisen metodin ollessa kyseessä rajoittavat käsitteet lisätään AND-operaattoreilla. Tuloksena saadaan DocumentAndScore-vektori.

Lopuksi Suosittelija poistaa saatujen dokumenttien joukosta alkuperäisessä tulosjoukossa esiintyneet artikkelit, jotta näitä ei näytettäisi käyttäjälle kahteen kertaan. Tämä tapahtuu yksinkertaisesti vertailemalla kahden joukon artikkelien koodeja ja säilyttämällä leikkauksen ulkopuolelle jäävät käsitteet. Lopuksi tulosjoukko typistetään kymmeneen dokumenttiin, jotta käyttäjä ei hämmentyisi suuren joukon äärellä.

Selite

Selitteen luomiseksi Airo sisältää kaksi luokkaa: RikastuksenSelittaja- ja RikastuskirjaajaHtmlksi-luokat. Nimiensä mukaisesti ensin mainittu luo selityksen ja jälkimmäinen tuottaa selitteestä HTML-tiedoston. Kolmas osapuoli selitteen luonnissa on SanomaExtractorin metodi tuotaUritusJaKirjaaUritusHtmlksi, joka tekee käytännössä samaa kuin tuotaUritus-metodi, mutta käyttää lisäksi RikastuskirjaajaHtmlksi-oliota selitteen luontiin.

Rikastuksenselittaja sisältää staattisen selita-metodin, joka ottaa argumenteikseen alkuperäisen, selitettävän haun merkkijonomuodossa, selitettävän tulosartikkelin Artikkeli-olion, rikastuksessa käytetyt disambiguaatio- ja rikastushahmot, sekä selitetiedoston osoitteen. Artikkeli-olion sijaan voidaan myös antaa halutun artikkelin koodi, ja Hakulaite-olio, jota halutaan käyttää artikkelin löytämiseksi. Tällöin metodi kutsuu annetun Hakulaite-olion haeKoodinPerusteella-metodia, joka muodostaa Lucene-kyselyn koodi-indeksiin ja palauttaa löytämänsä artikkelin.

SanomaExtractorin tuotaUritusJaKirjaaUritusHtmlksi-metodi suorittaa aiemmin kuvatun käsitteistyksen, mutta rikastusta suorittaessaan kutsuu Rikastajan rikastaUriHahmollaJaKirjaaRikastus-metodia. Tämä ottaa argumenteikseen rikastettavan UriAndStr-olion ja rikastukseen käytettävän Hahmo-olion lisäksi RikastuskirjaajaHtmlksi-olion sekä selitettävät hakukäsitteet. Metodi suorittaa rikastuksen tyypilliseen tapaan, mutta jonkin käsitteen painoa kasvattaessaan se tarkistaa sisältyykö käsite selitettäviin hakukäsitteisiin ja jos sisältyy, kutsuu kirjaajan setRikastuksessaSyntynytVaikutus-metodia, joka luo selitteeseen merkinnän vaikutuksesta.

RikastuskirjaajaHtmlksi-luokan konstruktori ottaa argumentikseen tiedostopolun ja poistaa tässä mahdollisesti aiemmin olleen tiedoston. Itse HTML-sivua se säilyttää merkkijonovektorikentässä, johon konstruktori aloittaa <html>- ja <body>-elementit. Luokka pitää sisällään kolme metodia, jotka lisäävät kenttään uusia merkkijonoja, joista lopullinen HTML-tiedosto muodostetaan. setVarmatUrit-metodi kirjaa ylös selitettävästä artikkelista sellaisenaan löytyneet hakukäsitteet, kun taas setRikastuksessaSyntynytVaikutus-metodi lisää rikastuksessa painoihin vaikuttaneiden käsitteiden synnyttämät muutokset. Käytännössä jälkimmäinen metodi ottaa argumenteikseen merkkijonomuodossa osuneen hakukäsitteen, polun jolla osuttiin vektorina, rikastuksessa olleen käsitteen sekä vaikutuksenvahvuuden double-lukuna. Näistä se tekee taulukkoon soluja HTML-elementteineen. Viimeinen metodi, kirjoitaSeliteHtmlksi, sulkee avoinna olevat elementit ja tuottaa valmiin HTML-tiedoston.

2. Rikastukseen käytetyt hahmot

YSO:lla disambiguointiin käytetty hahmo

```
<?xml version="1.0" encoding="UTF-8"?>
<hahmo>
	<polku paino="0.001" inklusiivinen="false">
		<ominaisuus koskeeSubjektia="true" syvyys="1">
			http://www.w3.org/2000/01/rdf-schema#subClassOf
		</ominaisuus>
		<ominaisuus koskeeSubjektia="false" syvyys="1">
			http://www.w3.org/2000/01/rdf-schema#subClassOf
		</ominaisuus>
	</polku>
	<polku paino="0.003" inklusiivinen="true">
		<ominaisuus koskeeSubjektia="true" syvyys="2">
			http://yso.fi/YSO#actuality
		</ominaisuus>
	</polku>
	<polku paino="0.001" inklusiivinen="false">
		<ominaisuus koskeeSubjektia="true" syvyys="2">
			http://www.w3.org/2000/01/rdf-schema#subClassOf
		</ominaisuus>
	</polku>
	<polku paino="0.002" inklusiivinen="true">
		<ominaisuus koskeeSubjektia="false" syvyys="5">
			http://www.w3.org/2000/01/rdf-schema#subClassOf
		</ominaisuus>
	</polku>
</hahmo>
```

YSO:lla indeksointiin käytetty hahmo

```
<?xml version="1.0" encoding="UTF-8"?>
<hahmo>
	<polku paino="0.05" inklusiivinen="false">
		<ominaisuus koskeeSubjektia="true" syvyys="1">
			http://www.w3.org/2000/01/rdf-schema#subClassOf
		</ominaisuus>
		<ominaisuus koskeeSubjektia="false" syvyys="1">
			http://www.w3.org/2000/01/rdf-schema#subClassOf
		</ominaisuus>
	</polku>
	<polku paino="0.2" inklusiivinen="false">
		<ominaisuus koskeeSubjektia="true" syvyys="1">
			http://yso.fi/YSO#actuality
		</ominaisuus>
	</polku>
	<polku paino="0.05" inklusiivinen="false">
		<ominaisuus koskeeSubjektia="true" syvyys="1">
			http://www.w3.org/2000/01/rdf-schema#subClassOf
		</ominaisuus>
	</polku>
	<polku paino="0.1" inklusiivinen="false">
		<ominaisuus koskeeSubjektia="false" syvyys="1">
			http://www.w3.org/2000/01/rdf-schema#subClassOf
		</ominaisuus>
	</polku>
</hahmo>
```