hyväksymispäivä arvosana

arvostelija

Perinnetiedon semanttinen annotointi

Juha Hautakangas

Helsinki 14.8.2013
Pro gradu -tutkielma
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta/Osasto – Fakultet/Sektion – Faculty/Section	Laitos – Institution – Department
Matemaattis-luonnontieteellinen tiedekunta	Tietojenkäsittelytieteen laitos

Tekijä – Författare – Author
Juha Hautakangas

Työn nimi – Arbetets titel – Title
Perinnetiedon semanttinen annotointi

Oppiaine – Läroämne – Subject
Tietojenkäsittelytiede

Työn laji – Arbetets art – Level	Aika – Datum – Month and year	Sivumäärä – Sidoantal – Number of pages
Pro gradu -tutkielma	14.8.2013	115 sivua

Tiivistelmä – Referat – Abstract

Organisaatioiden perinnejärjestelmät sisältävät valtavan määrän tietoa tallennettuna relaatiotietokantoihin. Tämä perinnetieto on organisaation liiketoiminnan kannalta usein erittäin tärkeää. Perinnejärjestelmiä tai perinnetietoa ei tyypillisesti ole kuvailtu riittävästi. Tällöin tiedon hakijan on osattava etsiä tietoa oikeasta paikasta, oikeilla menetelmillä, tunnettava ennalta haun kohteena olevaa aineistoa ja osattava tulkita hakutuloksia. Tilanne muuttuu, jos perinnetieto annotoidaan semanttisesti eli kuvaillaan semanttisen ja koneymmärrettävän metatiedon avulla. Tämä mahdollistaa nykyistä ilmaisuvoimaisempien kyselykielten käytön. Semanttisen metatiedon avulla koneet voivat ymmärtää tietosisältöjen merkityksiä. Näin ollen semanttisen annotoinnin avulla koneet voidaan valjastaa auttamaan ihmisiä relaatiotietokantoihin kohdistuvassa tiedonhaussa merkittävästi nykyistä paremmilla tavoilla.

Semanttinen web on näkemys webin laajennuksesta, jossa tietosisällöt kuvataan semanttisesti ja koneymmärrettävästi. Tämän toteuttamiseksi semanttisen webin tutkimuksessa on kehitetty joukko menetelmiä ja normeja. Tutkielmassa kuvataan, miten näitä menetelmiä ja normeja voidaan hyödyntää perinnetiedon semanttiseen annotointiin. Tutkielmassa vertaillaan eri lähestymistapoja ja esitetään, miten tilanteeseen nähden tarkoituksenmukaisin tapa voidaan valita.

Semanttinen annotointi kannattaa toteuttaa muuntamalla relaatiomallin mukainen tieto semanttisen webin tietomallin mukaiseksi tiedon aihealuetta kuvaavan ontologian rakenteiden ilmentymiksi. Tämä voidaan tehdä arkkitehtonisesti kahdella erilaisella lähestymistavalla. Ensimmäisellä tavalla ontologia luodaan automaattisesti relaatiotietokannan rakenteiden perusteella. Toisella tavalla jollakin ulkoisella muunnoskielellä määritellään, mihin ulkoisesti määriteltyjen ontologioiden rakenteisiin relaatiotietokannan rakenteet liittyvät. Muunnettua tietomallia ei tarvitse ylläpitää erillään, vaan muunnos voidaan tehdä ajonaikaisesti. Tällöin perinnejärjestelmät voivat edelleen ylläpitää tietoa alkuperäisissä relaatiotietokannoissa samalla, kun semanttista metatietoa hyödyntävät käyttäjät voivat suorittaa tietoon monipuolisia ja erittäin ilmaisuvoimaisia hakuja.

ACM Computing Classification System 2012 (CCS 2012):
Computing methodologies → Knowledge representation and reasoning; Information systems → Relational database model; Semantic web description languages; Ontologies;

Avainsanat – Nyckelord – Keywords
annotointi, Direct Mapping, kuvauslogiikat, metatieto, ontologiat, OWL, perinnejärjestelmät, R2RML, RDF, relaatiotietokannat, semanttinen web, SPARQL

Säilytyspaikka – Förvaringställe – Where deposited
Kumpulan tiedekirjasto, sarjanumero C-

Muita tietoja – Övriga uppgifter – Additional information

Sisältö

1 Johdanto

Perinnejärjestelmät (engl. legacy systems) ovat yleensä pitkällä aikavälillä ja useiden projektien aikana kehitettyjä laajoja tietojärjestelmiä. Niiden suunnittelusta tai toteutuksesta ei tiedetä juuri mitään eikä niitä voida ylläpitää tehokkaasti, mutta ne ovat edelleen arvokkaita niitä käyttäville organisaatioille [GOu05]. Yleinen syy perinnejärjestelmien käyttöön on, että ne ovat usein merkittävässä osassa organisaatioiden ydintoiminnoissa. Perinnejärjestelmät sisältävät monesti organisaation liiketoiminnan kannalta elintärkeää tietoa ja niiden toimintahäiriöt voivat vaikuttaa vakavasti organisaation liiketoimintaan [Ben95]. Tietojärjestelmät ikääntyvät väistämättä. Ikääntymistä nopeuttavat muun muassa ohjelmoijien ammattitaidon riittämättömyys, suunnittelemattomuus, puutteelliset asiakirjat ja niistä osittain johtuva vaillinainen katselmointi [Par94]. Toisin kuin ohjelmistotuotannossa, esimerkiksi laivan-, lentokoneen- tai talonrakennuksen arkkitehtonisten ratkaisuiden kuvaamiseen liittyy itsestäänselvyytenä joukko tarkasti määriteltyjä asiakirjoja, joita muut asiantuntijat katselmoivat huolellisesti.

Perinnetiedolla (engl. legacy data) tarkoitetaan perinnejärjestelmien varastoimaa tietoa, jota voi olla tallennettuna erilaisiin tietomalleihin. Tietomalleja voivat olla esimerkiksi tiedostomuotoinen tieto, relaatio- tai oliotietomallin mukaisesti tallennettu tieto. Useimmiten tieto on tallennettu relaatiotietokantoihin [Sel07], minkä johdosta tässä tutkielmassa perinnetiedolla tarkoitetaan organisaatioiden relaatiotietokantoihin tallennettua tietoa. Perinnetiedon hallittavuuden kannalta ongelmana on usein tietojärjestelmiin pitkällä aikavälillä tallennetun tiedon valtava määrä ja riittämätön kuvailu. Tiedon hakijalta vaaditaan tietämystä siitä, mitä tietoa on haettavissa ja mistä eri tietojärjestelmistä tietoja kannattaa hakea. Voidakseen tehdä tehokkaita hakuja hakijan on osattava käyttää sopivia avainsanoja, tulkita hakutuloksia ja usein ennalta tunnettava aineistoa. Tässä koneellinen käsittely voi tulla avuksi. Kun tieto kuvataan koneymmärrettävässä muodossa, koneet voivat erottaa

tiedon merkityksiä ja osaavat kertoa hakijalle täsmälleen, mitä tietoa on saatavilla ja mistä se löytyy.

Metatieto (engl. metadata) on kuvailevaa ja määrittelevää tietoa jostakin tietosisällöstä. Tiedon löydettävyyden kannalta laadukas metatieto on ratkaisevan tärkeässä asemassa [Gil08b]. Perinteisesti metatiedon kuvaaminen eli annotointi on parhaassa tapauksessa perustunut yhteisesti sovittuihin sanastoihin, joiden määrittelemien termien avulla tietoa kuvaillaan. Esimerkiksi kirjastoissa käytetty, nykyään usein sähköinen kortisto on metatietovarasto, joka sisältää hakemista helpottavaa kuvailutietoa kirjoista. Kortistossa kirjoista kuvaillaan muun muassa tekijä, nimi, osasto ja hyllypaikka, joiden perusteella hakija voi etsiä teoksen tiedot, kävellä suoraan oikealle osastolle ja poimia kirjan määritellystä hyllypaikasta. Tämä vaatii kortiston käyttäjältä ymmärrystä kirjoja kuvaavien käsitteiden merkityksistä. Ihmiselle käsitteiden ymmärtäminen on helppoa, mutta koneen näkökulmasta kortiston sisältö on merkityksetön joukko merkkijonoja. Esimerkiksi sisällössä esiintyvä merkkijono ”*kallio*” ei sisällä koneen näkökulmasta enempää tietoa kuin merkkijono ”*zxffcg*”. Tällöin kone ei erota käsitelläänkö sisällössä esimerkiksi umpikivimuodostumaa, kylää, kaupunginosaa, televisiosarjaa, ratamoottoripyöräilijää, valtionpäämiestä tai jotakin muuta kalliota. Vastaavasti käyttäjän etsiessä tietoa hakusanalla *kallio* hakukone ei tiedä minkälaista kalliota käyttäjä etsii. Tämän johdosta koneen on ymmärrettävä tiedon merkityksiä voidakseen auttaa hakijaa löytämään tietoa.

Semanttinen web [BHL01, AHa04] on näkemys internetin World Wide Web (WWW) -palvelun laajennuksesta, jossa tieto esitetään ihmiskäyttäjien lisäksi myös koneiden ymmärtämässä muodossa. Semanttisessa webissä tieto esitetään sen merkitystä vastaavien käsitteiden mukaisesti. Tämä on tarkoitus toteuttaa niin, että nykyisen WWW-palvelun tietosisältöihin liitetään koneymmärrettäviä semanttisia metatietokuvauksia eli annotaatioita [Hor02, Pan09]. Semanttisen webin toteuttamiseksi WWW-palvelun kehitykseen liittyviä normeja säätävä

World Wide Web Consortium (W3C) -yhteenliittymä [W3C11] on määritellyt joukon menetelmiä [W3C10], joilla semanttisia annotaatioita voidaan tuottaa ja käsitellä.

Semanttisen webin menetelmiä voidaan hyödyntää WWW-palvelun lisäksi muillakin sovellusalueilla. Tutkielman tavoitteena on selvittää, miten perinnetietoa voidaan kuvata semanttisten annotaatioiden avulla. Semanttisten annotaatioiden tuottaminen ja ylläpito on perinteisiä annotointimenetelmiä työläämpää. Tiedonhaun parantamisen kannalta semanttiset annotaatiot ovat kuitenkin tarpeellisia, sillä ilmaisuvoimaisemmat annotointimenetelmät mahdollistavat vastaavasti ilmaisuvoimaisempien kyselykielten käytön parantaen olennaisesti tiedon löydettävyyttä. Tutkielman tavoitteena on selvittää, miten voidaan löytää sovellusalueen kannalta tarkoituksenmukainen annotointimenetelmä, joka mahdollistaa riittävän ilmaisuvoimaisten kyselyjen suorittamisen aiheuttaen mahdollisimman vähän ylimääräistä työtä.

Tutkielma etenee seuraavasti. Luvussa 2 selvitetään kirjallisuuteen pohjautuen metatiedon merkitystä tiedonhaun näkökulmasta. Luvussa 3 läpikäydään semanttisen webin periaatteita ja menetelmiä. Luvussa 4 esitellään tutkielman kiinnostuksen kohteena olevan perinnetiedon ympäristö ja vertaillaan vaihtoehtoisia toteutustapoja perinnetiedon semanttiseen annotointiin. Luvussa 5 kuvataan, miten yhtä toteutusvaihtoehtoa voidaan hyödyntää käytännössä. Luvussa 6 arvioidaan tutkielman tuloksia.

2 Metatieto

Metatieto on nimensä mukaisesti tietoa tiedosta [LSw99, Gil08b]. Gillin [Gil08a] mukaan tämä määritelmä ei kuitenkaan riitä, sillä se tarkoittaa, että

esimerkiksi kirjaston kortistotietuetta voitaisiin kutsua metatiedoksi, mikäli se kuvaa sähköistä kohdetta, muttei esimerkiksi silloin, kun se kuvaa fyysistä kohdetta kuten kirjaa. Määritelmä ei niin ikään ilmaise riittävän tarkasti, että metatieto muotoillaan yleensä määrittämään vain kuvattavan kohteen tärkeimpiä ominaisuuksia. Esimerkiksi kirjaa kuvaavassa kortistotietueessa kuvaillaan kirjan otsikkoa, aihetta ja tekijää. Tämän johdosta on tarkempaa määritellä metatieto kuvattavan kohteen keskeisten ominaisuuksien rakenteiseksi kuvaukseksi [Gil08a]. Garsholin [Gar04] mukaan metatiedolla tarkoitetaan sekä tietoa tiedosta että yleisesti mitä tahansa lausuntoa jostakin tiedosta tai kuvattavasta kohteesta, kuten asiakirjasta, kuvasta tai tietosisällöstä. Edellä mainitut määritelmät eivät ota kantaa metatiedon muotoon. Metatietoa voi olla tallennettuna sähköisessä muodossa yhtä hyvin kuin paperille. Paperille tallennettu metatieto ei ole koneellisesti käsiteltävässä muodossa, mikä estää metatiedon tehokkaan hyödyntämisen. Tämän johdosta tietojärjestelmien yhteydessä on täsmällisempää määritellä, että metatieto on koneluettavassa muodossa olevaa kuvailutietoa [HKl10].

Guenther ja Radebaugh [GRa04] jakavat metatiedon kolmeen luokkaan: *kuvaavaan*, *rakenteelliseen* ja *hallinnolliseen* metatietoon. Kuvaava metatieto (engl. descriptive metadata) sisältää tiedon tunnistamisen ja löytämisen kannalta oleellista tietoa, esimerkiksi asiakirjan otsikon, tiivistelmän, tekijän ja avainsanat. Rakenteellinen metatieto (engl. structural metadata) määrittelee, miten tieto kootaan ja esitetään, esimerkiksi miten sivut järjestetään lukujen muodostamiseksi. Hallinnollinen metatieto (engl. administrative metadata) sisältää tiedon hallintaan liittyvää tietoa, kuten miten ja miksi tieto luotiin, kuka siihen pääsee käsiksi sekä muuta teknistä tietoa kuvattavasta kohteesta. Gilliland [Gil08b] luokittelee kuvaavan ja hallinnollisen metatiedon lisäksi *säilytys-*, *käyttö-* ja *teknisen metatiedon*. Säilytysmetatieto (engl. preservation metadata) kuvaa kohteiden säilytykseen liittyviä tietoja, kuten niiden fyysistä kuntoa tai säilytysmuodon muunnoksen aikana tapahtuneita muutoksia. Käyttömetatieto (engl. use metadata) kuvaa kohteiden käyttötapoja, kuten tietoja käyttäjistä, käyttöoikeuksista ja käytetyistä hakumenetelmistä. Tekninen

metatieto (engl. technical metadata) kuvailee järjestelmän tai metatiedon toimintaa ja käyttäytymistä. Teknistä metatietoa ovat esimerkiksi laitteisto- ja ohjelmistodokumentaatiot, käyttäjätunnukset, salasanat ja tiedostomuodot.

Metatiedon käytön pääasiallisena vaikuttimena on tiedonhaun helpottaminen ja parantaminen [Cat97]. Metatieto on hyödyllistä sekä sisällönhallinnan että tiedonhaun näkökulmasta, joten sitä käytetään molempien parantamiseen [Gar04]. Tiedonhaulla tarkoitetaan automaattisia menetelmiä, joilla suuresta määrästä tietoa voidaan löytää haluttuja tietosisältöjä niissä olevien sanojen ja ilmausten perusteella [Kos04]. Tiedonhaun yhteydessä tuloksellisuuden mittareina käytetään *saantia* (engl. recall) ja *tarkkuutta* (engl. precision) [Cat97]. Saanti mittaa, miten suuri osa kaikesta löydettävissä olevasta olennaisesta tiedosta löytyy [Kos04]. Tarkkuus mittaa, kuinka suuri osa kaikista haun tuottamista tuloksista on olennaista [Kos04]. Saannin ja tarkkuuden hyväksyttävä mittaustulos on useimmiten 10-20% [Gar04]. Tiedon löydettävyyden, eli hyväksyttävän saannin ja tarkkuuden kannalta laadukas metatieto on ratkaisevan tärkeässä asemassa [Gil08b]. Ollakseen laadukasta metatiedon on oltava täsmällistä, selkeää ja virheetöntä sekä ilmaistava tietoa, jota ei muuten voida päätellä metatiedon kuvaamasta kohteesta [HZo07]. Gillilandin [Gil08b] mukaan metatieto luo pohjan tehokkaiden ja yhteentoimivien tietämyspohjaisten tietojärjestelmien kehitykselle, ja useiden erilaisten metatietoluokkien olemassaolo tulee tulevaisuudessa osoittautumaan tärkeäksi tiedon esillesaannin kannalta.

2.1 Annotointi

Metatiedon tuottamista eli kohteen kuvaamista kutsutaan annotoinniksi. Annotaatioita ovat mitkä tahansa ilmaisut jostakin kuvattavasta kohteesta, esimerkiksi tekstin yhteyteen tehdyt merkinnät, asiakirjaan viittaavat hyperlinkit, sisältöä kuvaavat avainsanat ja kirjan tekijän tiedot. Marshall

[Mar98] määrittelee annotaatioille niiden ominaisuuksien perusteella erilaisia ulottuvuuksia. *Muodolliset annotaatiot* (engl. formal annotations) noudattavat rakenteellisia normeja ja sisältävät totunnaisia arvoja, jolloin eri kyselymenetelmät tulkitsevat niiden merkityksiä yhdenmukaisesti. *Epämuodollisia annotaatioita* (engl. informal annotations) ovat esimerkiksi kirjan tai lehden sivun reunaan käsin kirjoitetut muistiinpanot. Monet henkilökohtaisiksi huomautuksiksi tarkoitetut annotaatiot ovat *hiljaisia annotaatioita* (engl. tacit annotations). Tällaisia ovat esimerkiksi korostettu lause, selittämätön kirjanmerkki ja sivun reunassa oleva huutomerkki. Muiden kuin annotaatioiden tekijän on vaikea ymmärtää hiljaisia annotaatioita. Ajan kuluessa merkitykset saattavat kuitenkin unohtua annotoijalta, mikä johtaa annotoijan ymmärtämisedun menetykseen suhteessa muihin lukijoihin. *Selkeät annotaatiot* (engl. explicit annotations) on tarkoitettu muiden luettavaksi ja kuvaavat kohdetta ymmärrettävästi. Hypertekstidokumenttien yhteydessä esiintyvät linkit, tiedon pirstoutuminen ja toistuvat merkinnät ovat *laajalle levinneitä annotaatioita* (engl. hyperextensive annotations). *Avaria annotaatioita* (engl. extensive annotations) luodaan päivittäisiin lukemisiin liittyviin pohdintoihin perustuen. *Voimakkaat annotaatiot* (engl. intensive annotations) muodostuvat syvästä sitoutumisesta yksittäiseen tekstiin.

Annotaatioilla voi olla arvoa vain annotoijan kyseisellä tarkastelukerralla tehtyjen havaintojen huomioimiseksi. Tällöin kyseessä ovat *tilapäiset annotaatiot* (engl. transient annotations). Annotaatiot voivat olla annotoijan lisäksi hyödyllisiä myös muille tarkastelijoille, jolloin ne ovat *pysyviä annotaatioita* (engl. permanent annotations). *Yksityiset annotaatiot* (engl. private annotations) ovat esimerkiksi ilkeitä huomautuksia, joita ei ole tarkoitettu toisten nähtäviksi. *Julkisia annotaatioita* (engl. published annotations) ovat muun muassa ohjaajan merkinnät tutkielmassa, jotka on tarkoitettu muiden luettavaksi. Annotaatiot voivat olla myös *yleismaailmallisia* (engl. global), *jonkin tapajärjestelmän mukaisia* (engl. institutional), *työryhmäkohtaisia* (engl. workgroup) tai *henkilökohtaisia* (engl. personal) annotaatioita. Annotaatio on *kirjallista* (engl. annotation as writing), jos lukijat tekevät kirjallisia merkintöjä

havainnoistaan. Annotaatio on *ymmärrettyä* (engl. annotation as reading), jos lukijat pohtivat, keräävät, järjestävät ja tulkitsevat tekstin sisältöä, mutta eivät merkitse havaintojaan kirjallisesti. Annotaatioiden ulottuvuudet eivät poissulje toisiaan, vaan samalla annotaatiolla voi olla useita eri ulottuvuuksia. Esimerkiksi kalalajeja käsittelevän kirjan väliin jätetty kuva ahvenesta on epämuodollinen ja hiljainen annotaatio, mutta se voi myös olla esimerkiksi tilapäinen tai pysyvä, julkinen tai yksityiseksi tarkoitettu tai työryhmä- tai henkilökohtainen annotaatio.

Marshallin [Mar98] määrittelemiä ulottuvuuksia voidaan täydentää Gillilandin [Gil08b] esittelemillä annotaatioiden ominaisuuksilla. Annotaatiot ovat *merkityksellisiä*, jos ne noudattavat tiettyjä asiasanastoja tai muita sovittuja käytäntöjä, tai *merkityksettömiä*, jos ne eivät noudata mitään yhteisiä käytäntöjä. Annotaatiot voivat olla *asiantuntija-* tai *maallikkoannotaatioita* sen perusteella, ovatko ne aihealueen asiantuntijan vai asiaa tuntemattoman henkilön luomia. *Sisäisillä annotaatioilla* tarkoitetaan kuvatun kohteen luonnin yhteydessä kohteen luojan tekemiä kuvauksia. *Ulkoisilla annotaatioilla* tarkoitetaan kohteen elinkaaren myöhemmässä vaiheessa jonkun muun kuin kohteen luojan tekemiä kuvauksia. Annotaatiot voivat olla *kokoelmaa kuvaavia* tai *yksittäistä kohdetta kuvaavia*. *Vakiomuotoisten annotaatioiden* ei ole tarkoitus muuttua luonnin jälkeen. *Muuttuvien annotaatioiden* on tarkoitus muuttua esimerkiksi kohteen käytön tai muutosten yhteydessä. Annotaatiot voivat olla joko käsin tai automaattisesti jonkin tietokoneohjelman toimesta luotuja.

2.1.1 Annotointikielet

Annotaatioita ilmaistaan jollakin annotointikielellä. Karlssonin [Kar02, sivut 2-4] mukaan kielet luokitellaan *luonnollisiin* ja *keinotekoisiin kieliin*. Luonnollinen kieli on vuosituhansien aikana jossakin ihmisyhteisössä kehittynyt yhteydenpitoväline, joka toimii yhden tai useamman yksilön äidinkielenä.

Luonnollisten kielten tarkkaa määrää ei tiedetä, mutta niitä arvioidaan olevan tuhansia. Keinotekoinen kieli voi olla luonnollisen kielen kaltainen ihmisten välisen viestinnän apuväline tai tiettyä, lähinnä tieteellistä tai teknistä tarkoitusta varten kehitetty *formaali kieli*.

Formaali kieli on yleisnimitys kielille, joiden ilmausten merkitykset ovat täsmällisiä, käyttöalueet ahdasrajaisia ja perussymbolien lukumäärä pieni verrattuna luonnollisiin kieliin [Kar02, sivu 3]. Tämän johdosta formaaleilla kielillä voidaan ilmaista asioita täsmällisesti, mikä helpottaa niiden koneellista tulkintaa. Toisaalta ilmaisuja voidaan tehdä vain rajatuilla tavoilla. Esimerkiksi matematiikan ja logiikan kielet sekä tietokoneiden ohjelmointikielet ovat formaaleja kieliä. Niillä on helppo ilmaista matemaattisia ja loogisia formalismeja, mutta vaikea ilmaista tunteita tai solmia sosiaalisia suhteita [Kar02, sivu 3].

Kielen ilmaisuvoima tarkoittaa, miten hyvin asioita voidaan ilmaista kyseisellä kielellä. Luonnolliset kielet joustavat ilmaisutarpeen mukaan. Ne ovat formaaleja kieliä ilmaisuvoimaisempia, mutta epätäsmällisiä ja koneellisesti vaikeasti tulkittavia. Toisaalta formaali kieli voi olla liian täsmällinen, jolloin sen avulla ei voida ilmaista kaikkea tarpeellista. Tämän johdosta annotointikielen valinnassa on valittava kielen ilmaisuvoiman ja koneellisen ymmärrettävyyden välillä. Aina valinta ei ole mustavalkoista. Puoliformaalit (engl. semi-formal) kielet voidaan käsittää sovitteluratkaisuiksi luonnollisten ja formaalien kielten välillä. Puoliformaalien kielten avulla voidaan ottaa käyttöön tarkoituksenmukainen määrä luonnollisten kielten ilmaisuvoimaa ja formaalien kielten täsmällisyyttä siten, että tietty näkökulma on kuvattu formaalilla kielellä ja toinen luonnollisella kielellä [Lam09, sivu 144].

Tiedonhaun näkökulmasta on olennaista, että annotointikielen ilmaisuvoiman kasvaminen mahdollistaa vastaavasti ilmaisuvoimaisempien kyselykielten

käytön [HBS09] ja parantaa tietojärjestelmien semanttista yhteentoimivuutta [HKl10]. Semanttisesti yhteentoimivien tietojärjestelmien on pystyttävä vaihtamaan ja yhdistelemään tietoja sekä käyttämään toistensa palveluita niin, että merkitykset säilyvät yhtenäisinä tietojärjestelmien välillä [Hei95]. Annotointi- ja kyselykielten ilmaisuvoiman kasvaminen saattaa hidastaa kyselyitä. Sopivia kieliä valittaessa joudutaan usein tasapainoilemaan ilmaisuvoiman ja laskennallisen tehokkuuden välillä [GOS09].

2.2 Metatieto perinteisessä tiedonhaussa

Suurin osa tiedosta on tallennettu siten, että sitä on vaikea ymmärtää koneellisesti [AHa04, sivu 4]. Tämä aiheuttaa ongelmia tiedonhaussa. Antoniou ja van Harmelen [AHa04, sivu 2] luettelevat neljä perinteistä tiedonhakuun liittyvää ongelmaa, jotka esiintyvät esimerkiksi organisaatioiden hakupalveluiden tai WWW-hakukoneilla tehtävien avainsanaperustaisten hakujen yhteydessä: *korkea saanti ja matala tarkkuus*, *matala tai olematon saanti*, *tulosten riippuvaisuus käytetystä sanastosta* ja *yksittäisten asiakirjojen löytyminen*. Korkea saanti ja matala tarkkuus merkitsee, että vaikka hakutulosten joukossa olisivatkin etsityt asiakirjat, niillä ei ole juuri merkitystä, jos hakutulosten joukossa on lisäksi valtava määrä hieman tai ei ollenkaan haettuun asiaan liittyviä asiakirjoja. Matala tai olematon saanti tarkoittaa sitä, että hakutulosten joukko ei sisällä riittävän suurta määrää hakuun liittyviä tärkeimpiä asiakirjoja. Tulosten riippuvaisuus sanastosta kuvaa ongelmaa, jossa tiedonhaussa käytetyt hakusanat poikkeavat aineistossa esiintyvistä ilmaisuista. Tällöin merkityksellisesti toisiaan vastaavat erilaisia ilmaisuja sisältävät kyselyt tuottavat erilaisia hakutuloksia. Yksittäisten asiakirjojen löytymisen ongelma esiintyy, kun etsitään tietoa, joka on koostettava useiden asiakirjojen sisällöistä. Tällöin joudutaan suorittamaan useita kyselyitä eri asiakirjojen löytämiseksi sekä yhdistelemään käsityönä eri asiakirjojen sisältämää tietoa.

Tiedonhaussa hakutulosten laatu riippuu siitä, miten hyvin käyttäjän on onnistunut ilmaista toiveensa kyselyssä [Voo94]. Mikäli tieto on kuvattu eri sanastolla kuin kysely, saanti ja tarkkuus eivät yleensä ole hyviä. WWW-palvelun tietosisältöjen määrän valtavan kasvun myötä perinteisillä hakukoneilla saatavat hakutulokset eivät ole enää tyydyttäviä kaikille käyttäjille [Ren10]. Hakutulosten tarkkuus ei ole aina riittävällä tasolla, vaan monet käyttäjät haluavat jalostetumpia hakutuloksia tekemiensä kyselyiden merkitysten perusteella.

Tiedonhaun parantamiseksi useissa tietojärjestelmissä on tuotettu aihealuetta kuvaavaa metatietoa [HZo07]. Tällaisen metatiedon hyödyllisyys tekstipohjaisten hakujen yhteydessä on kuitenkin kyseenalaista. Hawking ja Zobel [HZo07] ovat selvittäneet tietosisällön aihealuetta kuvaavan metatiedon (engl. topic metadata) hyödyllisyyttä yleisten WWW-sivustoilla tehtävien tekstipohjaisten hakujen yhteydessä. Heidän mielestään aihealuetta kuvaavasta metatiedosta on vain vähän hyötyä. Syy tähän on pääosin se, että metatietomerkintöjen avulla on vaikea ilmaista yksittäisen tietosisällön tärkeyttä suhteessa muihin saman aihealueen tietosisältöihin. Mikäli tietosisällön aihealuetta kuvaava metatieto perustuu rajattuun sanastoon, useita tietosisältöjä on väistämättä kuvattu samalla tavalla, mikä vaikeuttaa tietyn tietosisällön löytämistä muiden joukosta [HZo07]. Tietosisältöjen arviointi on edelleen tehtävä ihmisten toimesta, sillä sovellukset eivät vielä pysty ymmärtämään sisältöjä riittävän hyvin [AHa04, sivu 4].

Hawking ja Zobel [HZo07] näkevät aihealuetta kuvaavan metatiedon ongelmaksi myös sen, että käyttäjät saattavat ymmärtää metatietokenttiä eri tavoin. Esimerkiksi parhaiten tunnettu sanasto metatiedon kuvaamiseen, Dublin Core (DC) -sanasto, määrittelee joukon kuvattavan kohteen ominaisuuksia määritteleviä kenttiä [Gar04]. Kenttien joukkoon kuuluvat esimerkiksi *otsikko*, *aihe*, *kuvaus*, *tekijä*, *julkaisija*, *päivämäärä* ja *kieli* [Gar04]. DC-sanasto määrittelee kenttien merkityksen, muttei miten kentät ja niiden arvot pitäisi

esittää [Gar04]. Tämän johdosta voi olla epäselvää, onko esimerkiksi tekijä-kentässä organisaation, henkilön vai roolin nimi [HZo07]. Käyttäjillä ei myöskään ole selkeää mahdollisuutta tunnistaa sopivia hakusanoja etukäteen [HZo07]. Hawking ja Zobel [HZo07] päättelevät, että aihealuetta kuvaava metatieto on hyödyllistä vain, jos tiedetään etukäteen, millaisia hakuja käyttäjät tulevat tekemään. Heidän mielestään näyttää kuitenkin mahdolliselta, että kaikki ongelmat, kuten eri ihmisten eri tavoilla ymmärtämät sanojen merkitykset, eivät ole koneellisesti ratkaistavissa. Ongelmien vähentämiseksi metatieto voidaan määritellä semanttisesti.

2.3 Semanttinen metatieto

Gillilandin [Gil08b] mukaan metatieto on semanttista, jos se noudattaa tietynlaisia sääntöjä. Säännöt voivat liittyä tiedon rakenteeseen, jolloin metatieto kuvataan esimerkiksi sovittua kenttämääritystä noudattaen. Säännöt voivat määrätä kentissä käytettävät arvojoukot, esimerkiksi tietyssä kentässä voidaan käyttää vain tietyn asiasanaston määrittelemiä termejä. Säännöt voivat ohjeistaa kentissä käytettävän kirjoitusasun ja -muodon, mahdollisen koodiston sekä tallennusmuodon. Säännöt voivat määrätä metatiedolle tallennusmuodon. Mikäli metatiedon tiedetään olevan semanttista eli noudattavan tunnettuja sääntöjä, on huomattavasti helpompaa ohjelmoida kone ymmärtämään tiedon merkityksiä. Laadukkaan metatiedon avulla voidaan vähentää tiedonhaun ongelmia [Cat97, Gar04, Gil08b, HZo07].

Gillilandin [Gil08b] määrittely ei ota kantaa, missä määrin metatietokuvauksen on noudatettava tietynlaisia sääntöjä ollakseen semanttista. Tämän johdosta voidaan ajatella, että metatieto ei ole joko semanttista tai ei-semanttista, vaan metatiedolla on vaihteleva määrä semanttisuutta sen mukaan, miten kattavasti metatietokuvaus noudattaa määrätynlaisia sääntöjä. Esimerkiksi joitakin yksinkertaisia sääntöjä noudattava metatietokuvaus on heikosti semanttista.

Toisaalta kattavasti erilaisia sääntöjä noudattava koneellista ymmärrettävyyttä merkittävästi parantava metatietokuvaus on vahvasti semanttista. Tarkoituksenmukainen määrä semanttisuutta riippuu sovellusalueesta.

Handschuh ja Staab [HSt02] ovat määritelleet tarkoituksenmukaisten semanttisten annotaatioiden vaatimuksia. Tietojärjestelmien välisen yhteentoimivuuden mahdollistamiseksi annotaatioiden on perustuttava yhteisiin sanastoihin ja noudatettava yhdenmukaisesti niiden rakenteita. Olennaisen tiedon esillesaannin mahdollistamiseksi annotoitavien kohteiden tunnisteiden on oltava yksilöiviä. Ylimäärän välttämiseksi yhtä asiaa ei saa kuvata useammalla kuin yhdellä annotaatiolla. Tämän lisäksi annotointiympäristön on oltava sellainen, että annotaatiot ovat ylläpidettävissä ja tuotettavissa helposti ja tehokkaasti.

Antonioun ja van Harmelenin [AHa04, sivu 3] mukaan tiedonhakua voidaan parantaa pitämällä tietosisällöt nykymuotoisina, mutta käyttäen merkitysten ymmärtämiseen älykkäämpiä kieliteknologian ja tekoälyn tutkimukseen perustuvia menetelmiä. Kieliteknologia tutkii "*luonnollisen kielen mallintamista tietokonetta varten, erityisesti kielen jäsentämistä eri tasoilla tai kielen generoimista ja näihin liittyviä menetelmiä*" [Kos04]. Tekoäly pyrkii ymmärtämään ihmiselle tyypillistä älykkyyttä ja jäljittelemään sitä koneellisesti [Hof00, sivu 560, RNo03, sivu 1].

Tämä lähestymistapa näyttää kuitenkin edelleen liian kunnianhimoiselta toteutuakseen, sillä menetelmät eivät toimi vielä riittävällä tasolla [AHa04, sivu 3]. Esimerkiksi tekstiaineiston ymmärtämiseen ei riitä tekstissä käytetyn kielen sanaston ja kieliopin tuntemus, vaan ymmärtämiseen tarvitaan lisäksi tietämystä aihealueesta ja sitä ympäröivästä maailmasta, joita koneella ei lähtökohtaisesti ole käytössään [Hof00, sivu 603].

Toinen, helpommin toteutettavissa oleva lähestymistapa tiedonhaun parantamiseen on, että tietosisällöt kuvataan koneymmärrettävässä muodossa ja niiden käsittelyssä hyödynnetään älykkäitä menetelmiä [AHa04, sivu 3]. Semanttinen web määrittelee menetelmiä tietosisältöjen koneymmärrettävään merkityksiä ilmaisevaan annotointiin. Sisällyttämällä tällaisia *semanttisia annotaatioita* osaksi tietoa saadaan tiedon merkitykset kaikkien tietoa käsittelevien sovellusten käyttöön ilman, että merkitykset ja päättelylogiikka täytyy ohjelmoida jokaiseen sovellukseen erikseen.

3 Semanttinen web

WWW-palvelun tietosisällöt ovat usein tuotettu vain ihmiskäyttäjiä varten, mikä on aiheuttanut tarpeen kehittää WWW-palvelua sekä ihmisille että koneille ymmärrettäväksi [BHS03]. Semanttisen webin [BHL01, AHa04] tavoitteena on WWW-palvelun tietosisältöjen kuvaaminen sekä ihmisten että koneiden ymmärtämällä tavalla. Tämä mahdollistaa esimerkiksi sähköisen kaupankäynnin ja web-palveluiden käytön automatisoinnin sekä WWW-hakukoneiden yhteydessä tiedonhakuun liittyvien ongelmien vähentämisen eli saannin ja tarkkuuden merkittävän parantamisen [Hor02].

Tavoitteen toteutuminen vaatii, että nykyisiä tietosisältöjä kuvataan *semanttisilla annotaatioilla,* jotka ilmaisevat tiedon merkityksiä koneymmärrettävällä tavalla [BHS03, Hor07, Kir04, Pan09]. Semanttiset annotaatiot ovat koneellisen käsittelyn näkökulmasta merkittävästi perinteisiä metatietokuvauksia ilmaisuvoimaisempia. Tämän johdosta ne mahdollistavat vastaavasti ilmaisuvoimaisempien kyselykielten käytön [HBS09], mikä parantaa merkittävästi tiedon löydettävyyttä. Semanttisilla annotaatioilla voidaan kuvata erilaisia tietosisältöjä, kuten WWW-sivuja, kuvia, videoita, asiakirjoja ja tietokantoja [HHa08]. Semanttinen web mahdollistaa esimerkiksi *semanttisten*

hakukoneiden toteuttamisen [Ren10]. Semanttiset hakukoneet ymmärtävät kyselyiden ja tietosisältöjen merkityksiä, joten ne pystyvät vastaamaan käyttäjien kyselyihin merkittävästi perinteisiä hakukoneita tarkemmin. Semanttisen webin menetelmät mahdollistavat esimerkiksi monipuolisten haku- ja selailukäyttöliittymien toteuttamisen, semanttisen suosittelun käyttäjää kiinnostavan aihepiirin asioista ja hakukenttään kirjoitetun tekstin semanttisen täydentämisen [HMä06, Hyv08, MHS05].

Semanttisen webin käyttäjiä kutsutaan *toimijoiksi* (engl. agents). Toimijoilla voidaan tarkoittaa ihmiskäyttäjiä [BHS03], mutta useimmiten ne ovat ohjelmistokomponentteja [AHA04, sivu 14], jotka suorittavat käyttäjiltä saamiaan tehtäviä. Tehtävä voi koskea yksinkertaista tiedonhakua tai laajempaa tehtäväkokonaisuutta. Esimerkiksi lomamatkan varaamisen tehtäväkseen saanut toimija voi selvittää muiden toimijoiden avulla lentolippujen ja hotellien hinnat, vertailla niitä käyttäjän toiveisiin ja ehdottaa yhtä tai useampaa sopivaa pakettia. Käyttäjä tekee päätöksen ja toimija voi saada uudeksi tehtäväkseen varausten tekemisen valituilla vaihtoehdoilla.

Koska tieto on koneymmärrettävässä muodossa, automaattiset työkalut voivat tukea ylläpitoa etsimällä annotaatioiden joukosta epäjohdonmukaisuuksia ja tunnistamalla tiedossa esiintyviä ilmaisuja [AHa04, sivu 4]. Semanttisessa webissä tiedonhakua voidaan parantaa korvaamalla hakusanaperustainen haku kyselyihin vastaamisella, jolloin pyydetty tietämys haetaan, tunnistetaan ja esitetään ihmisystävällisellä tavalla [AHa04, sivu 4]. Kyselyiden vastaukset voivat kattaa useita asiakirjoja. Lisäksi näkymäoikeuksien määritteleminen asiakirjoihin tai niiden osiin on mahdollista.

Semanttisen webin menetelmien käyttö ei rajoitu WWW-palvelun sisältöjen kuvailuun, vaan menetelmiä voidaan käyttää muillakin sovellusalueilla [DOS03, sivut 239-248]. Soveltuvissa tilanteissa käytettyinä menetelmät mahdollistavat

älykkäämpien, tietosisältöjen merkityksiä ymmärtävien sovellusten toteuttamisen.

3.1 Semanttinen web ja tekoäly

Semanttisen webin sovellusten älykkyydestä puhuttaessa on syytä varoa ylimainontaa. Ylimainonnalla luodaan ylisuurilla lupauksilla odotuksia, joihin ei pystytä vastaamaan. Tämä aiheuttaa pettymyksiä, mikä voi estää minkä tahansa uuden menetelmän käyttöönoton. Näin tapahtui [AHA04, sivu 16] tekoälyn tutkimuksessa 1980-luvulla, kun alkuperäisten näkemysten mukaisen ihmisen älykkyyttä vastaavan tekoälyn kehittäminen kahdessa vuosikymmenessä [McC55, Sim65, sivu 96] osoittautui liian kovaksi haasteeksi.

Tekoälyn tutkimuksessa alun perin vallinnut päämäärä ihmisälyn (engl. common-sense) koneellisesta, formaaleilla kielillä kuvattuihin tosiasioihin ja niistä tehtyihin päätelmiin perustuvasta jäljittelystä on kyseenalaistettu [Hal04]. Esimerkiksi käyttökelpoisen tietämyksen on esitetty liittyvän toimijan kulloinkin suorittamaan tehtävään. Tämän johdosta tarvitaan tehtäväkohtaisesti sopivia näkökulmia, joiden löytäminen perinteisen tietämyksenhallinnan menetelmin on vaikuttanut epätodennäköiseltä. On niin ikään epävarmaa, voidaanko ihmisaivojen toimintaa täysin jäljitellä koneellisesti, sillä kaikkia aivojen toimintaperiaatteita ei tunneta [Hof00].

Bechara ja kumppanit ovat tutkimuksillaan horjuttaneet käsitystä, että ihmisen päätöksentekoa hallitsee tunteeton, havaintoihin perustuva looginen tapahtumasarja [BDD94, BDD00, Bec04a, Mos11]. Tutkijat ovat selvittäneet tunteiden merkitystä päätöksenteossa tutkimalla sekä normaaleita että aivovaurioista kärsineitä potilaita. He laativat uhkapeliin pohjautuvan kokeen [BDD94], jonka tarkoitus on jäljitellä ihmisen jokapäiväiseen elämään liittyvää tasapainottelua riskin ja hyödyn välillä. Kokeen perusteella on havaittu, että

tunnepohjainen valinta näyttää edeltävän tietoista valintaa. Vaikka ihmiset uskovat tekevänsä päätöksiä järkipohjalta, he tekevät niitä usein tunnepohjalta.

Tekoälyn tutkimuksen alkuperäinen tavoite [McC55, Sim65, sivu 96] ihmisälyn keinotekoisesta vastineesta saattaa olla lähtökohtaisesti mahdoton toteuttaa ainoastaan formalisoinnin ja loogisen päättelyn keinoin. Tiettyjen ihmisaivojen piirteiden, kuten oppimisen, luovuuden, tunteidenkäsittelyn ja ymmärryksen omasta olemassaolostaan jäljittely on vaikeaa [Hof00, sivu 573]. Tekoälyn alkuperäisen tavoitteen saavuttamisen osaratkaisuita on mahdollisesti haettava tietojenkäsittelytieteen ja matematiikan lisäksi myös muilta tieteenaloilta, kuten biologiasta eli elävän maailman tutkimuksesta ja fysiikasta eli fyysisten ilmiöiden tutkimuksesta [AHa04, sivu 16]. On mahdollista, että aivojen toimintaa voidaan jäljitellä keinotekoisesti vasta, kun on onnistuttu rakentamaan toimiva jäljennös elävistä aivoista [Hof00, sivu 573]. Ihmisaivojen toiminnasta opitaan kuitenkin tutkimuksen edistyessä yhä enemmän [BDD94, BDD00, Fie04]. Esimerkiksi elävien aivosolujen yhdistäminen koneisiin on ajankohtainen tutkimuskohde [War10].

Tekoälyn ja semanttisen webin tavoitteet ovat samansuuntaisia, mutta eroavat ratkaisevasti toisistaan [AHa04, sivu 16, Hal04]. Päinvastoin kuin tekoälyn alkuperäisten tavoitteiden toteutuminen, semanttisen webin tavoitteena ei ole saavuttaa ihmisälyä vastaavaa tietoisuuden tasoa. Sen sijaan semanttisen webin tavoitteena on ihmiskäyttäjien päivittäisten tehtävien suorittamisen koneellinen avustaminen. Semanttisen webin menetelmillä voidaan auttaa ihmiskäyttäjiä hallitsemaan tietämystä olennaisesti perinteisiä menetelmiä paremmin. Esimerkiksi entistä älykkäämpien tiedonhakusovellusten toteuttaminen on mahdollista. Niiden käyttöönottoon ei vaadita valtavaa tieteellistä läpimurtoa, vaan toteutusmenetelmät ovat jo suurelta osin olemassa [AHa04, sivu 7]. Semanttisen webin näkemykseen kuuluu ihmisen älykkyyden hyödyntäminen tiedon kuvailussa: koneita ei vaadita saavuttamaan ihmistä vastaavaa älykkyyttä eikä ymmärtämään luonnollisia kieliä, vaan ihmisten on tarkoitus

oppia kuvaamaan tietoa koneellisesti ymmärrettävällä tavalla [Ber98b]. Tämä tapahtuu tuottamalla semanttisia, koneymmärrettäviä annotaatioita, joiden avulla koneille voidaan ilmaista tietosisältöjen merkityksiä.

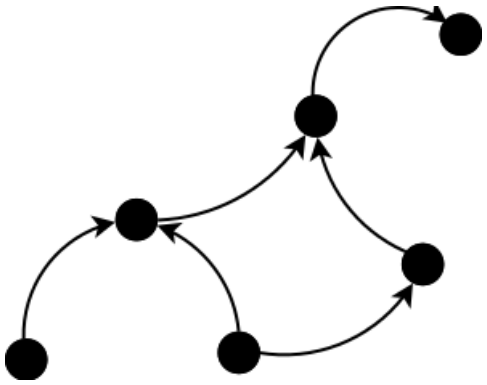
Alkuperäisten lupausten täyttämättä jättämisestä huolimatta tekoälyn tutkimus on tuottanut käyttökelpoisia menetelmiä, joihin semanttisen webin kehitys pohjautuu [AHa04, sivu 16]. Merkittäviin menetelmiin kuuluvat *kuvauslogiikat* ja *päättely*. Kuvauslogiikat ovat predikaattilogiikan osa-alueita [BHS09], jotka määrittelevät joukon formaaleja ilmaisuja, joiden avulla tietämystä voidaan esittää [NBr03]. Päättelyn avulla ulkoisesti kuvattua tietämystä voidaan laajentaa koneellisesti [NBr03]. Kuvauslogiikoiden ja päättelyn lisäksi semanttinen web pohjautuu vahvasti verkkojen ominaisuuksia tutkivaan *verkkoteoriaan* [Die05].

3.2 Verkkoteoria ja kuvauslogiikat

Verkkoteoria [Die05] on matematiikan osa-alue, joka tutkii verkkojen ominaisuuksia. Verkko voi esittää mitä tahansa verkoksi hahmotettavaa kohdetta ottamatta kantaa sen tulkintaan. Verkko esittää solmuja ja niiden välisiä suhteita eli kaaria. Verkkoteoriaa on kehitetty kuluneiden vuosisatojen aikana ja nykyään sitä sovelletaan useilla tieteenaloilla [Hay00, PDh07]. Se näkyy esimerkiksi kuljetusverkostojen suunnittelussa ja WWW-palvelun rakenteessa.

Suunnattu verkko on nimeltään suhteikko [SVä02, sivu 10]. Suhteikossa kaarilla on suunta, joka määrää miten verkossa voidaan edetä. Kuva 1 havainnollistaa suhteikkoa. Semanttisen webin tietomalli pohjautuu suhteikkoon, jossa solmut ja kaaret on nimetty [Ber98d]. Tekoälyn ja kielitieteen tutkimuksessa nimetystä suhteikosta käytetään nimitystä *semanttinen verkosto* [AHa04, sivu 65, Kar02, sivut 233-235]. Semanttisilla verkostoilla voidaan esittää kieliä kytkemällä

käsitteiden perusmerkityksiä toisiinsa. Semanttisissa verkostoissa käsitteet kuvataan verkoston solmuina ja sanojen väliset merkityssuhteet kuvataan kaksi verkoston solmua toisiinsa yhdistävinä kaarina. Semanttisen verkoston voidaan ajatella olevan laaja sanasto, joka kuvaa käsitteiden keskinäisiä yhteyksiä. Loogisten symboleiden käyttöön verrattuna semanttiset verkostot kuvaavat tietoa ihmisille selkeällä tavalla. Kaaviomuotoisen esityksen selkeysetu loogisiin lauseisiin verrattuna häviää kuitenkin nopeasti verkoston monimutkaistuessa. Koneita varten semanttinen verkosto on kuvattava kaaviomuotoisen esityksen sijaan jollakin formaalilla kielellä.



Kuva 1: Suunnattu verkko eli suhteikko [DOS03, sivu 196].

Semanttiset verkostot kuvaavat asioiden välisiä yhteyksiä, mutta eivät niiden merkityksiä. Toisin sanoen niistä puuttuu *formaali semantiikka* [BHS09]. Formaali semantiikka mahdollistaa tiedon rakenteiden merkitysten esittämisen täsmällisesti ja yksiselitteisesti. Formaalin semantiikan avulla voidaan varmistaa, että eri toimijoilla eli tietämyksen käyttäjillä on yhdenmukainen ymmärrys asioiden ja yhteyksien merkityksistä. Formaalia semantiikkaa voidaan lisätä koneellisesti tulkittavissa olevien käsitemallien avulla [HBa97]. Tällöin kuvauskielen rakenteiden merkityksiä ilmaistaan täsmällisesti yhdistämällä niitä formaalisti kuvattujen käsitemallien käsitteisiin [AHa04, sivu 110]. Semanttisessa webissä on kyse formaalin semantiikan sisältävien metatietokuvausten lisäämisestä WWW-palvelun tietosisältöjen yhteyteen [Kir04].

Formaalin semantiikan sisältävän tietämyksen esittämiseen voidaan käyttää logiikkaa [AHa04, sivu 151]. Logiikka tarjoaa täsmällisen ja yksiselitteisen formaalin semantiikan sisältävän kuvauskielen, jolla on suuri ilmaisuvoima. Logiikka tutkii varsinaisesti päättelyä [SVä02, sivu 46]. Logiikka sisältää todistuksia ja päättelysääntöjä, joiden perusteella kuvattua tietämystä voidaan laajentaa ja kyselyiden vastauksia selittää. Logiikan tärkein symbolikieli on ensimmäisen kertaluokan logiikka eli *predikaattilogiikka* [SVä02, sivu 67]. Predikaattilogiikan avulla voidaan esittää lähestulkoon kaikki todellisuutta tai sen mahdollisia vaihtoehtoja kuvaavat mallit.

Formaalin semantiikan sisältävän tietämyksen esittämiseksi semanttisten verkostojen pohjalta on kehitetty kuvauslogiikoita (engl. description logics). Kuvauslogiikat ovat predikaattilogiikan osa-alueita eli joukko formaaleja kieliä, joiden avulla tietämystä voidaan esittää [BHS09]. Logiikkaan perustuvan formaalin semantiikan lisäksi useissa kuvauslogiikoissa on sisäänrakennettu päättelyä tukeva järjestelmä [Hor07]. Tämän johdosta ne sopivat erittäin hyvin semanttisen webin tietämyksen esittämisen pohjaksi. Kuvauslogiikoilla voidaan ilmaista käsitteitä ja niiden välisiä yhteyksiä monipuolisesti ja merkityksellisesti erilaisten suhdetyyppien avulla. Kuvauslogiikoiden pääasiallisena tarkoituksena on taata kielelle laskennallinen ratkeavuus [BHS09].

Kuvauskielen valinta on usein tasapainottelua ratkeavuuden ja ilmaisuvoiman välillä [AHa04, sivu 114]. Kieli on *ratkeava* (engl. decidable), jos siitä voidaan jollakin algoritmilla äärellisessä ajassa määrittää, onko annettu lause kyseiseen kieleen kuuluva lause vai ei. Toisin sanoen on olemassa Turingin kone, joka pysähtyy aina saadessaan syötteeksi kielen aakkoston merkeistä koostuvan äärellisen merkkijonon, ja hyväksyy kieleen kuuluvat merkkijonot, mutta hylkää muut. Ratkeavan kielen käyttö tiedon kuvaamisessa mahdollistaa päättelyyn liittyvien ongelmien tehokkaan ratkaisemisen. Ratkeavuus on tärkeä ominaisuus, sillä nykyaikaiset kuvauslogiikoita hyödyntävät järjestelmät sisältävät päättelykoneita, joiden avulla ulkoisesti määritellystä tietämyksestä

voidaan päätellä koneellisesti uutta tietämystä äärellisessä ajassa [BHS09]. Päättelykoneet huomioivat laskennassa sekä tietosisällön keskinäiset yhteydet että niihin käsitemallien avulla liitetyt merkitykset. Esimerkiksi kuvauslogiikat sisältävät sääntöjä, joiden mukaan päättelykoneet voivat johtaa ulkoisesti kuvatusta tietämyksestä sen loogisia seurauksia [Krö10]. *Ratkeamattomien* kielten käytössä ongelma on, että kaikkia loogisia seurauksia ei voida selvittää luonnollisella päättelyllä [BHS09, SVä02, sivu 138]. Tällöin päättelykoneen laskennan ei voida taata päättyvän. Useimmat kuvauslogiikat ovat ratkeavia [BHS09].

Baader ja kumppanit [BHS09] kuvaavat esimerkin loogisesta ilmaisusta. Heidän mukaansa lause "*lääkärin kanssa naimisissa oleva mies, jonka kaikki lapset ovat joko lääkäreitä tai professoreita*" voitaisiin esittää logiikassa seuraavasti: "$Ihminen \wedge \neg Nainen \wedge (\exists naimisissa.Lääkäri) \wedge (\forall omistaaLapsen.(Lääkäri \vee Professori))$". Lauseen perusteella yksilö kuuluu joukkoon $\exists naimisissa.Lääkäri$, mikäli jokin toinen yksilö sekä kuuluu Lääkäri-käsitteeseen että on tähän yhteydessä *naimisissa*-roolin kautta. Yksilö kuuluu vastaavasti joukkoon $\forall omistaaLapsen.(Lääkäri \vee Professori)$, jos kaikki tähän *omistaaLapsen*-roolin kautta yhteydessä olevat yksilöt kuuluvat joko Lääkäri- tai Professori -käsitteeseen. Mikäli lisätään sääntö *"aviopari koostuu aviomiehestä ja aviovaimosta"*, voidaan loogisena seurauksena päätellä kaikkien *naimisissa*-roolin kautta keskenään yhteydessä olevien henkilöiden muodostavan avioparin.

Tekoälyn ja matematiikan tieteenaloilla pitkällä aikavälillä tutkittuja ja todistettuja verkkoteoriaan ja kuvauslogiikoihin pohjautuvia menetelmiä käytetään semanttisen webin rakenteiden pohjana. Verkkoteoriaan perustuvassa tietomallissa tietoa voidaan liittää yhteen. Kuvauslogiikoiden avulla voidaan ilmaista yhteenliitettyyn tietoon liittyvien rakenteiden merkityksiä sekä päätellä niiden perusteella uutta tietoa. Tämän johdosta semanttisen webin tietämyksen esittämiseen käytettävillä menetelmillä on vahva teoreettinen pohja.

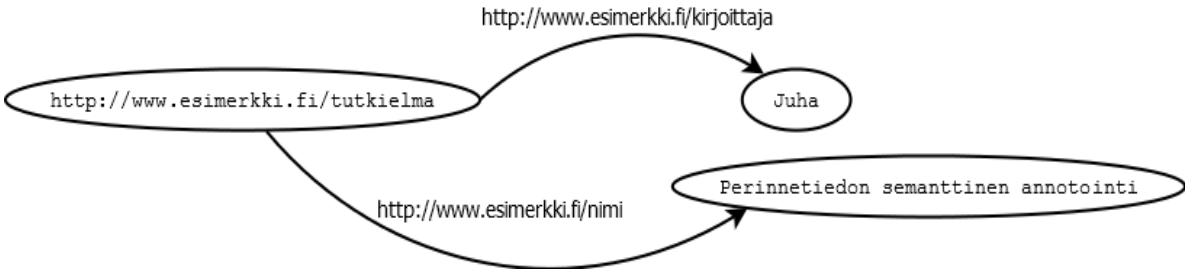
3.3 Semanttisen webin tietomalli

Yleisin tapa semanttisesti annotoidun tiedon esittämiseen [HBS09] [AHa04, sivu 69] on Resource Description Framework (RDF) -tietomalli [MMi04, AHa04, sivut 63-66]. RDF-tietomalli on W3C-yhteenliittymän suositus [MMi04, KCa04] semanttisen, koneymmärrettävän tiedon esittämiseksi. RDF-tietomallilla voidaan esittää semanttinen verkosto koneymmärrettävällä tavalla.

RDF-tietomallin peruskäsitteitä ovat *resurssit* (engl. resources), *ominaisuudet* (engl. properties) ja *lauseet* (engl. statements) [BKD01, AHa04, sivut 63-66]. Resurssit ovat mitä tahansa käsitteellisiä tai aineellisia kohteita, joita halutaan kuvata. Ne voivat olla esimerkiksi WWW-sivuja, niiden osia, ihmisiä, eläimiä, esineitä tai asioita. Jokaisella resurssilla on yksilöivä Universal Resource Identifier (URI) -tunniste [BFM05], joka erottaa resurssin muista. Ominaisuudet ovat resursseja, jotka kuvaavat resurssien välisiä suhteita. Esimerkiksi *omistaja, ikä, kirjoittaja* ja *otsikko* voivat olla jonkin resurssin ominaisuuksia. Kuten muut resurssit, myös ominaisuudet yksilöidään URI-tunnisteilla. Lauseet ilmaisevat resursseilla olevia ominaisuuksia. Lauseet ovat kolmikoita, jotka koostuvat resurssista, sen ominaisuudesta ja ominaisuuden arvosta.

Ominaisuuksien arvot voivat olla joko resursseja tai literaaleja eli atomisia merkkijonoja. Esimerkiksi luonnollisen kielen lause ”*Tutkielman kirjoittaja on Juha*” voidaan esittää RDF-tietomallissa kolmikkolauseella, jossa kuvattava resurssi on *tutkielma*, resurssin ominaisuus on *kirjoittaja* ja ominaisuuden arvo on *”Juha”*. Käyttämällä asianmukaisia URI-tunnisteita lause voisi olla kolmikko: (http://www.esimerkki.fi/tutkielma, http://www.esimerkki.fi/kirjoittaja, ”Juha”), jossa *http://www.esimerkki.fi/tutkielma* on tutkielman URI-tunniste, *http://www.esimerkki.fi/kirjoittaja* on kirjoittaja-ominaisuuden URI-tunniste ja *”Juha”* kirjoittaja-ominaisuuden arvo merkkijonoliteraalina. RDF-lauseita havainnollistaa kuva 2. RDF-lauseen resurssia, ominaisuutta ja arvoa kutsutaan

myös luonnollisten kielten lauseenjäsenten nimeämiskäytännön mukaisesti *subjektiksi*, *predikaatiksi* ja *objektiksi* [HHa08].



Kuva 2: Semanttisen verkoston muodostavia tutkielmaa kuvaavia RDF-lauseita.

Ominaisuuksien arvoina eli objekteina käytettävät literaalit voidaan määritellä ohjelmointikielten ja tietokantajärjestelmien tapaan tietyn tyyppisiksi, jolloin RDF-tietomallia tulkitseva sovellus näkee, miten kyseinen arvo pitää tulkita [AHa04, sivu 67]. RDF-tietomallissa tällaisia arvoja kutsutaan *tyypitetyiksi literaaleiksi*. Niitä käytetään ilmaisemaan, onko luettava arvo esimerkiksi kokonaisluku, liukuluku, merkkijono, totuusarvo vai päivämäärä.

RDF-tietomallissa on *konkretisointikoneisto* (engl. reification mechanism), jonka avulla on mahdollista kuvata kolmikkolauseilla toisia lauseita [AHa04, sivu 67]. Tällaisilla lauseilla voidaan kuvata luottamusta, mistä on hyötyä tietyillä sovellusalueilla. Esimerkiksi lause ”*Hanna uskoo, että tutkielman kirjoittaja on Juha*” lisää muiden käyttäjien silmissä luottamusta lauseelle ”*Tutkielman kirjoittaja on Juha*”. Lauseisiin viittaaminen on mahdollista, kun lauseelle määritellään yksilöivä URI-tunniste. RDF-tietomallissa on vain kolmikkolauseita, joten lauseen URI-tunnistetta ei voida lisätä suoraan lauseen osaksi. Edellä mainitun esimerkin mukainen kolmikkolauseen kuvaaminen on mahdollista luomalla RDF-tietomalliin uusi luottamusta kuvaava resurssi. Luottamusta kuvaava resurssi yhdistetään kuvattavaan lauseeseen kolmella ominaisuudella: *subjekti*, *predikaatti* ja *objekti*. Ominaisuuksien arvoiksi määritellään kuvattavan lauseen subjekti, predikaatti ja objekti.

Semanttisen verkoston ilmaisuvoiman rajallisuus merkitsee, että RDF-tietomallissa on heikkouksia [AHa04, sivu 68]. RDF-tietomallissa kaikki ominaisuudet eli yhteydet resurssien välillä esitetään kahden resurssin välisinä eli kaksiasteisina. Yhteyksiä on toisinaan kuitenkin luontevampaa ja yksinkertaisempaa esittää useampiasteisina. Tämä ei rajoita yhteyksien ilmaisemista, sillä jokainen useampiasteinen yhteys voidaan esittää joukkona kaksiasteisia yhteyksiä. Rajoituksen takia loogisesti moniasteisia yhteyksiä joudutaan kuitenkin esittämään monimutkaisemmin, useana kaksiasteisena yhteytenä.

Antonioun ja van Harmelen [AHa04, sivu 69] mainitsevat myös kaksi muuta heikkoutta, joista ensimmäinen liittyy ominaisuuksien käsittelyyn. Koska ominaisuudet ovat tietynlaisia resursseja, niitä voidaan käyttää kolmikkolauseen resursseina, joita voidaan kuvata ominaisuuksilla. Tällainen mahdollisuus lisää tietomallin joustavuutta, mutta saattaa epätavallisena ratkaisuna hämmentää mallintajia. Toinen ongelma liittyy RDF-tietomallin konkretisointikoneistoon. Lauseiden luominen lauseista lisää tietomallin ilmaisuvoimaa, mutta myös monimutkaisuutta. Tämän johdosta konkretisointikoneisto sopii paremmin ilmaisuvoimaisemmille esityskerroksille.

Semanttisena verkkona RDF-tietomallin ilmaisuvoima rajoittuu käytännössä resurssien välisten kaksiasteisten yhteyksien kuvaamiseen [AHa04, sivu 109]. RDF-tietomallissa ei voida kuvata resurssien tai ominaisuuksien merkityksiä. Tämän johdosta tarvitaan lauseiden rakenteiden merkityksiä kuvaavia asiasanastoja.

3.4 Asiasanastot

Asiasanasto (engl. vocabulary) koostuu aihealueita kuvaavista *termeistä*, joita voidaan käyttää tiedon luokittelussa [Gar04]. Asiasanaston jokainen termi

vastaa tiettyä *käsitettä*. Toisaalta samaa käsitettä voidaan kuvata useammalla termillä.

Valvottu asiasanasto (engl. controlled vocabulary) on täsmennetty asiasanasto, jossa jokainen termi viittaa täsmälleen yhteen käsitteeseen [Gar04]. Valvotun asiasanaston tarkoitus on estää tekijöitä määrittelemästä liian laajoja, suppeita tai merkityksettömiä termejä sekä estää eri tekijöitä käyttämästä eri termejä samasta käsitteestä. Yksinkertaisimmillaan valvottu sanasto on joukosta termejä koostuva lista.

Taksonomia (engl. taxonomy) on valvottu asiasanasto, jossa termeille on kuvattu hierarkia [Gar04]. Taksonomiassa termit voidaan ryhmitellä ja luokitella tavoilla, jotka helpottavat oikean termin löytämistä. Taksonomiasta nähdään helposti, mitkä termit liittyvät läheisesti toisiinsa. *Tesaurus* (engl. thesaurus) laajentaa taksonomiaa [Gar04]. Se mahdollistaa erilaisten ominaisuuksien määrittelyn termeille. Tesauruksessa hierarkiasuhde kuvataan määrittelemällä termin olevan laajempi tai suppeampi kuin jokin toinen termi. Termeihin voidaan liittää lyhyt kuvaus termin merkityksen ja käyttötarkoituksen selventämiseksi. Termin voidaan määritellä olevan toisen termin synonyymi ilmaisemalla, että jotakin toista termiä tulisi käyttää mieluummin. Termin voidaan määritellä liittyvän assosiatiivisesti toiseen termiin olematta sen synonyymi, laajempi termi tai suppeampi termi. Hierarkiasuhteessa olevalle termille voidaan kuvata *laajin mahdollinen termi*. Hierarkiasuhde voidaan esittää puurakenteena. Sanastoa kuvaava puu on verkon erikoistapaus, jossa solmut kuvaavat termejä ja solmuja yhdistävät kaaret termien välisiä suhteita. Puun solmut ja kaaret eivät saa muodostaa piirejä eli *syklejä*. Puussa esiintyvän termin laajin mahdollinen termi on käytännössä puun juurisolmu.

Valvottujen asiasanastojen, taksonomioiden ja tesaurusten ilmaisuvoima kasvaa edellä mainitussa järjestyksessä. Vaikka tesaurukset ovat ilmaisuvoimaisimpia

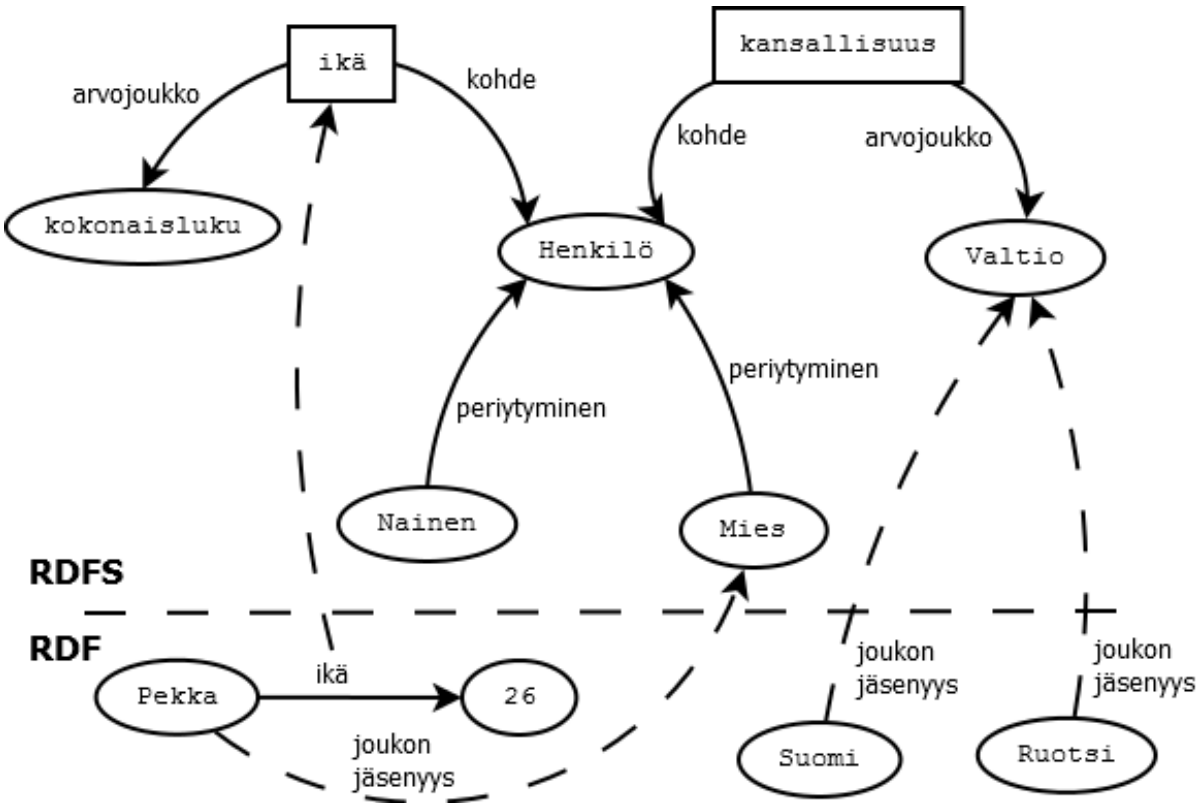
asiasanastoja, niiden ilmaisuvoima on rajallinen. Tämä johtuu siitä, että termien välisiä suhteita voidaan kuvata vain kolmella erilaisella suhdetyypillä: *laajempi tai suppeampi termi*, *suositeltu termi* ja *yhteys toiseen termiin*. Tämän lisäksi termejä voidaan kuvata vain yhdellä ominaisuudella: *kuvauksella*. Tämän johdosta tesaurusten kuvaamiseen käytettävä *sanastonkuvauskieli* on suljettu [Gar04]. Semanttisessa webissä sanastot on esitettävä koneymmärrettävällä tavalla. Näin ollen tarvitaan *formaaleja sanastonkuvauskieliä*.

Resource Description Framework Schema (RDFS) -kieli on W3C-yhteenliittymän suosittelema RDF-tietomallia laajentava formaali sanastonkuvauskieli [BGu04, HHa08]. RDFS-kiellä määritellyn sanaston avulla voidaan kuvata RDF-lauseiden ominaisuuksien merkityksiä sekä määritellä resursseja ryhmitteleviä luokkia. RDFS-kielellä luokat ja ominaisuudet voidaan järjestää hierarkkisesti. Ominaisuuksia voidaan tyypittää määrittelemällä, mihin luokkiin ominaisuuteen viittaavan lauseen subjektin ja objekti tulee kuulua. Tällainen sanasto mahdollistaa yksinkertaisten päättelyiden tekemisen.

RDFS-kielellä voidaan määritellä tietoa luokittelevia *luokkia*, *luokkien hierarkkioita*, *luokkien välisiä ominaisuuksia*, *luokkakohtaisia ominaisuuksia* ja *luokkien ilmentymiä* [HPH03]. Luokiksi voidaan määritellä esimerkiksi *Henkilö, Nainen, Mies* ja *Valtio*. Tällöin *Nainen*-luokan voidaan määritellä olevan *Henkilö*-luokan aliluokka. *Kansallisuus* voidaan määritellä luokat *Henkilö* ja *Valtio* yhdistäväksi ominaisuudeksi. *Ikä* voidaan määritellä *Henkilö*-luokan ominaisuudeksi, jonka arvoksi kelpaa kokonaisluku. *Suomi* ja *Ruotsi* voidaan määritellä *Valtio*-luokan ilmentymiksi. *Pekka* voidaan määritellä *Mies*-luokan ilmentymäksi, jonka *ikä*-ominaisuuden arvo on 26. Tätä esimerkkiä havainnollistaa kuva 3.

Simple Knowledge Organization System (SKOS) -malli [MBe09] on W3C-yhteenliittymän suositus valvottujen, rakenteisten asiasanastojen, kuten

taksonomioiden ja tesaurusten julkaisemiseen semanttisessa webissä. SKOS-malli koostuu joukosta RDF-tietomalliin ja RDFS-kieleen perustuvia formaaleja kieliä. SKOS-mallin käyttö on suositeltavaa, jos tarkoituksena on olemassaolevien sanastojen julkaiseminen koneymmärrettävässä muodossa tarvitsematta merkittävästi lisätä sanaston ilmaisuvoimaa.



Kuva 3: RDFS-sanaston avulla kuvattuja RDF-lauseita.

RDFS-kielellä laajennetun RDF-tietomallin avulla voidaan ilmaista verkkomuotoinen tietomalli, jossa käytettyä sanastoa rakenteita ja rajoitteita on ulkoisesti määritelty yksinkertaisilla ilmaisuilla. RDFS-kieli sisältää formaalin semantiikan [Hay04], mutta sen ilmaisuvoima ei riitä kattavaan käsitemallinnukseen ja päättelyyn [BKD01, HPH03, HHa08].

Antoniou ja van Harmelen [AHa04, sivu 111] kuvaavat RDFS-kielen rajoitteita. RDFS-kielellä ei ole mahdollista ilmaista joukko-opin ja Boolen operaatioita.

Tällöin ei voida ilmaista kahden luokan olevan *erillisiä joukkoja*. Lisäksi on mahdotonta määritellä uusia luokkia yhdistämällä muita luokkia operaatioiden *yhdiste, leikkaus* ja *komplementti* avulla. RDFS-kielellä ei esimerkiksi voida ilmaista, että luokkaan *Mies* kuuluva ilmentymä ei voi kuulua luokkaan *Nainen*. RDFS-kielellä ei voida määritellä ominaisuuksien arvoja koskevia lukumäärään perustuvia rajoitteita eli *kardinaalisuusrajoitteita*. Ominaisuuksien arvoja ei voida myöskään rajoittaa koskemaan vain tiettyjä luokkia. Esimerkiksi ei ole mahdollista ilmaista lauseita "*kirahvilla on täsmälleen neljä jalkaa*", "*puussa on vähintään yksi oksa*" tai "*kirahvi on kasvissyöjä, mutta muut eläimet ovat lihansyöjiä*". RDFS-kielellä ominaisuuksille ei voida määritellä päättelymahdollisuuksia lisääviä erityispiirteitä, kuten *symmetrisyys, transitiivisuus* ja *käänteisyys*. Symmetrisyys kuvaa ominaisuuden pätevyyttä molempiin suuntiin. Lauseen "*Matin veli on Teppo*" symmetriseksi määritellyn *veli*-ominaisuuden perusteella on pääteltävissä, että "*Tepon veli on Matti*". Transitiivisuus kuvaa ominaisuuden siirtyvyyttä. Mikäli edellisen esimerkin *veli* -ominaisuus määritellään transitiiviseksi ja kuvataan lisäksi lause "*Sepon veli on Matti*", voidaan päätellä, että Matin lisäksi myös Teppo on Sepon veli. Symmetrisyyden ja transitiivisuuden perusteella on mahdollista päätellä, että Matti, Teppo ja Seppo ovat veljeksiä. Käänteisyydellä voidaan kuvata ominaisuuden muuttumista käänteiseksi lukusuunnan vaihtuessa. Esimerkiksi lauseessa "*kirahvi syö lehtiä*" käänteiseksi määritellyn *syö* -ominaisuuden perusteella voidaan päätellä käänteisesti, että "*lehdet ovat kirahvin ruokaa*".

Berners-Lee [Ber98c] kuvaa semanttiselle webille kerrosarkkitehtuurin, jossa ilmaisuvoimaisemmat sanastonkuvauskielet pohjautuvat heikompiin edeltäjiinsä. Arkkitehtuurin alimmalla tasolla on RDF-tietomalli, joka tarjoaa kolmikkolauseilla tehtäviin määrittelyihin perustuvan menetelmän koneymmärrettävän metatiedon kuvaamiseksi [BKD01]. Seuraavalla tasolla on RDFS-kieli, jolla kolmikkolauseiden rakenteiden merkityksiä kuvataan yksinkertaisesti [BKD01]. Kolmannella eli *loogisella* tasolla RDFS-kielen ilmaisuvoimaa lisätään formaalin semantiikan sisältävien logiikkaan pohjautuvien ilmaisujen avulla [BKD01]. Semanttisessa webissä formaaleja

sanastonkuvauskieliä kutsutaan *ontologiakieliksi* ja niiden kuvaamia rakenteellisia sanastoja *ontologioiksi*. Tesaurukset voidaan nähdä yksinkertaisina (engl. lightweight) ontologioina [AHa04, sivu 180]. Perinteiseen tesaurukseen verrattuna ontologia on formaalisti kuvattu, koneymmärrettävä käsitemalli. Tesaurus voidaan muuntaa suoraan formaaliin, esimerkiksi SKOS-mallin mukaiseen muotoon. Käytännössä suoran muuntamisen yhteydessä tesaurukseen kannattaa lisätä koneellisen käsittelyn kannalta merkityksellisempiä yhteyksiä [HVT08]. Perinteiset tesaurukset ovat yleensä suunniteltu ihmiskäyttäjiä varten, jolloin niiden käyttöön vaaditaan ihmisen, usein aihealueen asiantuntijan ymmärrystä tesauruksen kuvaamasta maailmasta [HVT08]. Tämän johdosta tesaurukset eivät kuvaa käsitteiden välisiä yhteyksiä koneelle riittävällä tarkkuudella. RDFS-kielellä kuvattu ontologia on tesaurusta ilmaisuvoimaisempi, mutta ei riittävästi [HHa08]. Tämän johdosta RDFS-kielen ilmaisuvoimaa on kasvatettava ilmaisuvoimaisempien ontologiakielten avulla [AHa04, sivu 112]. Formaalin semantiikan sisältäviä loogisen tason ontologiakieliä käsitellään luvussa 3.5.

3.5 Käsitemallit eli ontologiat

Ontologiat mahdollistavat sovellusalueen kannalta merkityksellisen maailman esittämisen koneille, joten niitä voidaan pitää semanttisen webin selkärankana [DMN09]. *Ontologia*-sanaa on käytetty alun perin filosofiassa kuvaamaan ja luokittelemaan maailmassa esiintyviä asioita [SBF98, AHa04, sivu 10]. Tekoälyn tutkimukseen liittyy päättelyiden tekeminen mallinnetusta maailmasta. Tämän johdosta on luonnollista, että tekoälyn tutkijat ovat ottaneet ontologia-sanan käyttöönsä [SBF98]. Tietojenkäsittelytieteessä ontologia kuvaa, mitä maailmasta voidaan esittää koneymmärrettävästi [SBF98]. Semanttisessa webissä ontologioilla on merkittävä rooli [HPH03]. Ne kuvaavat rakenteellisia sanastoja, joissa termien keskinäiset suhteet on määritelty formaalisti. Tämä mahdollistaa kuvatun termistön yksiselitteisen ymmärtämisen eri toimijoiden kesken. Päinvastoin kuin asiasanastot, taksonomiat ja tesaurukset suljetun

kuvauskielen sanastoina, ontologiat ovat avoimen kuvauskielen sanastoja [Gar04]. Tämä tarkoittaa, että ontologioissa kuvaaja voi määritellä ominaisuuksia ja suhdetyyppejä tarpeen mukaan.

Gruberin [Gru93] mukaan ontologia on "*täsmällinen ja formaalisti määritelty käsitemalli tietyn, kiinnostuksen kohteena olevan aihealueen käsitteistä*". Gruber viittaa Geneserethin ja Nilssonin [GNi87] määritelmään, jonka mukaan käsitemalli on "*käsitteellinen, yksinkertaistettu näkemys tiettyä tarkoitusta varten esitettävästä maailmasta*". Borstin [Bor97] mukaan ontologia on "*jaetun käsitteistön formaali määrittely*". Tämä laajentaa Gruberin [Gru93] määritelmää siten, että ontologia on useiden osapuolien välillä jaettu yhteinen näkemys aihealueen käsitteistä ja niiden välisistä yhteyksistä. Studer ja kumppanit [SBF98] yhdistävät Gruberin [Gru93] ja Borstin [Bor97] määritelmät muotoilemalla, että "*ontologia on täsmällinen ja formaalisti määritelty jaettu käsitemalli tietyn aihealueen käsitteistä*". Ontologioiden hyödyllisyydestä tiedonhaussa vallitsee laaja yhteisymmärrys [HBS09].

Antoniou ja van Harmelen [AHa04, sivu 110] määrittelevät ontologiakielten vaatimuksia. Niihin kuuluvat koneellisen käsittelyn mahdollistava *hyvin määritelty lauseoppi*, merkitykset täsmällisesti ja yksiselitteisesti kuvaavat *tarkoituksenmukaiset ilmaisut ja formaali semantiikka*, *sovellusalueen kannalta riittävä ilmaisuvoima* sekä *tuki tehokkaalle päättelylle*. Tehokkaan päättelyn tukeminen mahdollistaa esimerkiksi uuden tietämyksen päättelyn ulkoisesti kuvatun ontologian perusteella, ontologian ja tietämyksen yhdenmukaisuuden koneellisen tarkistamisen sekä tiedon rakenteiden automaattisen luokittelun ontologian käsitteiden perusteella.

Kokonaisvaltaisten loogisten ilmaisujen lisääminen RDFS-kieleen muodostaa erittäin ilmaisuvoimaisen ontologiakielen semanttisen webin kerrosarkkitehtuurin [Ber98c] kolmannelle eli loogiselle tasolle [AHa04, sivu

113]. Laajentamalla puhtaasti RDFS-kieltä tällainen ontologiakieli noudattaa myös johdonmukaisesti semanttisen webin kerrosarkkitehtuuria. Ratkaisu ei kuitenkaan mahdollista tehokasta päättelyä [AHa04, sivu 112]. Mitä ilmaisuvoimaisempi ontologiakieli, sitä monimutkaisempaa ja tehottomampaa on päättelyiden tekeminen. Ilmaisuvoiman lisääminen johtaa jossakin vaiheessa laskennallisesti ratkeamattomaan kieleen. Päättelyiden päättymisen varmistamiseksi tarvitaan ontologiakieli, joka on sekä riittävän ilmaisuvoimainen sovellusalueen kannalta tarpeellisen tietämyksen kuvaamiseen että tehokkaiden päättelykoneiden tuettavissa. Tämän johdosta loogisen tason ontologiakielien suunnittelussa joudutaan tekemään tasapainoisia ratkaisuja, jotka mahdollistavat sekä riittävän ilmaisuvoiman että tehokkaan päättelyn.

3.5.1 Ontologiakielet

Defense Advanced Research Projects Agency (DARPA) -laitos on Yhdysvaltain puolustusministeriön tutkimuslaitos, jonka kehityshankkeilla on ollut merkittävä vaikutus muun muassa internetin ja Global Positioning System (GPS) -satelliittipaikannusjärjestelmän syntyyn [DAR08]. Yksi DARPA-laitoksen 2000-luvun alussa rahoittamista tutkimushankkeista oli DARPA Agent Markup Language (DAML) -hanke, jossa kehitettiin RDFS-kielen laajennokseksi ilmaisuvoimaisempaa DAML-ONT-ontologiakieltä seuraavan sukupolven WWW-palvelun perustaksi [HMc00, Hor02, Hor07]. Samaan aikaan pääosin eurooppalaisista tutkijoista koostuneella ryhmällä oli käynnissä On-To-Knowledge -hanke [Fen00], jossa kehitettiin vastaavaa Ontology Inference Layer (OIL) -ontologiakieltä [FHH01]. Hankkeiden tavoitteet olivat niin lähellä toisiaan, että kielten kehitys päätettiin yhdistää. Tämä johti DAML+OIL-ontologiakielen syntymään [Hor02, Hor07]. DAML+OIL-ontologiakieli on semanttisen webin sisältöjen kuvaamiseen suunniteltu erittäin ilmaisuvoimainen kuvauslogiikka. DAML+OIL-ontologiakielen ongelmia olivat virallisen aseman ja RDF-tietomallista sekä RDFS-kielestä tuolloin puuttunut

formaali semantiikka [Hor07], joka määriteltiin vuonna 2004 [Hay04]. Tämä johti siihen, että W3C-yhteenliittymän asettama työryhmä kehitti DAML+OIL-kielen pohjalta formaalin semantiikan sisältävien RDF-tietomallin ja RDFS-kielen kanssa yhteensopivan Web Ontology Language (OWL) -ontologiakielen [DSc04, Hor07]. OWL-ontologiakielestä tuli W3C-yhteenliittymän suositus [DSc04], joka vahvisti sen aseman semanttisen webin ontologiakielenä.

OWL-ontologiakieli [DSc04] pohjautuu kuvauslogiikoihin [Hor07, HPH03] ja sijoittuu semanttisen webin kerrosarkkitehtuurin [Ber98c] loogiselle tasolle. OWL-ontologiakieli hyödyntää kuvauslogiikoiden hyvin määriteltyä formaalia semantiikkaa ja käytännöllisiä päättelymenetelmiä [Hor07]. Formaali semantiikka mahdollistaa ontologioiden ja niiden käsitteisiin yhdistetyn tiedon jakamisen täsmälliset merkitykset säilyttäen [HPH03]. Päättelymenetelmät mahdollistavat ontologian kuvaaman tietämyksen laajentamisen koneellisesti [NBr03]. Ilmaisuvoiman ja ratkeavuuden välillä tasapainottelun vuoksi OWL-ontologiakieli ei ole suora RDF-tietomallin tai RDFS-kielen laajennos, mutta suurelta osin yhteensopiva. OWL-ontologiakielellä voidaan määritellä luokkia ja ominaisuuksia RDFS-kieltä vastaavasti, mutta OWL-ontologiakieli tarjoaa paljon monipuolisempia ja ilmaisuvoimaisempia rakenteita käsitteiden kuvaamiseen [HHa08, HPH03]. Esimerkiksi luokkia voidaan määritellä toisten pohjalta joukko-opin operaatioiden *yhdiste*, *leikkaus* ja *komplementti* avulla ja ominaisuuksille voidaan määritellä *arvoalue*- ja *kardinaalisuusrajoitteita.*

OWL-ontologiakielellä voidaan RDFS-kielen ilmaisujen lisäksi määritellä joukko-opin ja Boolen operaatioita, kardinaalisuusrajoitteita ja tiettyjä luokkia koskevia rajoituksia ominaisuuksien arvoille sekä muita erityispiirteitä, kuten symmetrisyys, transitiivisuus, käänteisyys ja funktionaalisuus [DSc04, HPH03]. Tarkastellaan luvun 3.4 esimerkin mukaisesti luokkia *Henkilö, Nainen, Mies* ja *Valtio*. OWL-ontologiakielellä voidaan määritellä luokat *Nainen* ja *Mies* erillisiksi joukoiksi. Ruotsi ja Suomi voidaan määritellä *Valtio*-luokan eri ilmentymiksi. *Kansalainen*-ominaisuuden voidaan määritellä olevan

kansallisuus-ominaisuuden käänteisominaisuus. *Valtioton*-luokkaan voidaan määritellä kuuluvan kaikki *Henkilö*-luokan ilmentymät, joiden *kansallisuus*-ominaisuuden arvoa ei ole määritelty. *Monikansallinen*-luokkaan kuuluvan täsmälleen sellaiset *Henkilö*-luokan ilmentymät, joiden *kansallisuus*-ominaisuudelle on vähintään kaksi arvoa. *Suomalainen*-luokkaan voidaan määritellä kuuluviksi kaikki *Henkilö*-luokan ilmentymät, joiden *kansallisuus*-ominaisuuden arvona on *Suomi*. *Ikä*-ominaisuuden voidaan määritellä olevan funktionaalinen eli kullakin ominaisuutta käyttävällä ilmentymällä voi olla ominaisuudelle vain yksi arvo.

OWL-ontologiakieli vastaa ilmaisuvoimaltaan useimpien kuvauslogiikoiden formalismeja, perustuu WWW-palvelun arkkitehtuuriin ja on laajasti käytetty [HHa08]. Se jakautuu kolmeen osajoukkoon: *OWL Lite*-, *OWL DL*- ja *OWL Full* -kieleen. Kielet rajoittavat käytettävissä olevia loogisia ilmaisuja eri tavoin, jotka soveltuvat eri sovellusalueille ja käyttäjäryhmille.

OWL Full -kieli sisältää kaikki OWL-suosituksen [DSc04] rakenteet. Se on ilmaisuvoimaltaan vahvin, mutta monimutkaisin ja vaikein oppia. OWL Full -kieli mahdollistaa RDF-tietomallin ja RDFS -kielen rakenteiden yhdistelemisen mielivaltaisilla tavoilla [AHa04, sivu 113]. Tämä mahdollistaa jopa näiden semanttisen webin kerrosarkkitehtuurin [Ber98c] näkökulmasta alempien tasojen kielten rakenteiden alkuperäisten merkitysten muuttamisen. OWL Full -kielellä kuvataan predikaattilogiikan ilmaisuja koneluettavassa muodossa [HHa08]. Se on ainoa OWL-ontologiakielen muoto, joka ei vastaa mitään kuvauslogiikkaa lauseopillisesti tai semanttisesti [Hor07]. OWL Full -kieli on lauseopillisesti ja semanttisesti täysin yhteensopiva RDF-tietomallin kanssa [AHa04, sivu 113, Hor07]. Jokainen hyvin muodostettu RDF-tietomallin mukainen esitys on myös OWL-ontologiakielen mukainen esitys. Monimutkaisuuden lisäksi OWL Full -kielen huono puoli on, että se on ratkeamaton. Sen ilmaisuvoima ylittää rajan, jonka jälkeen täydellisen ja tehokkaan päättelyn takaaminen ei ole mahdollista.

OWL DL -kieli on OWL-suosituksen [DSc04] kolmesta osakielestä eniten käytetty [HHa08]. Kielen nimi viittaa erääseen kuvauslogiikkaan (engl. description logic), predikaattilogiikan ratkeavaan osajoukkoon. OWL DL -kieli rajoittaa RDF-tietomallin ja OWL-ontologiakielen rakenteiden käyttöä. Tämä varmistaa, että jokainen OWL DL -kielellä kuvattu ontologia vastaa hyvin tunnettua kuvauslogiikkaa, joka on predikaattilogiikan ja siten OWL-ontologiakielen mahdollisimman ilmaisuvoimainen todistettavasti ratkeava osajoukko. Ratkeavuuden perusteella voidaan varmistaa, että kuvauslogiikoiden päättelysääntöjä noudattavat päättelykoneet pystyvät tehokkaasti laskemaan kuvatun käsitteistön määritelmien perusteella kaikki mahdolliset loogiset seuraukset. Tällöin päättelykoneet voivat päätellä ulkoisesti kuvatun ontologian pohjalta uutta tietoa, etsiä ontologiasta epäjohdonmukaisuuksia sekä luokitella tietoa [HHa08, AHa04, sivu 111]. OWL DL -kielen huono puoli on, että rajoituksista johtuen se ei ole täysin yhteensopiva RDF-tietomallin kanssa. RDF-tietomallin mukaista esitystä on yleensä laajennettava ja rajoitettava eri tavoin, että RDF-esityksestä saadaan OWL DL -kielen mukainen esitys [AHa04, sivu 113].

OWL Lite -kieli rajoittaa OWL-ontologiakielen rakenteiden käyttöä OWL DL -kieltä enemmän. OWL Lite -kieli on ilmaisuvoimaltaan heikoin, mutta samalla helppokäyttöisin [AHa04, sivu 114]. OWL Lite- ja OWL DL -kielet voidaan nähdä erittäin ilmaisuvoimaisina kuvauslogiikoina [HPH03].

Ontologiakielen valinnassa tulee harkita, mikä kieli sopii parhaiten kyseiseen käyttötarkoitukseen [AHa04, sivu 114]. OWL Lite- ja OWL DL -kielten välillä tehtävä valinta riippuu siitä, miten ilmaisuvoimaisia OWL DL- ja OWL Full -kielten rakenteita tarvitaan. OWL DL- ja OWL Full -kielten välillä tehtävä valinta riippuu siitä, miten laajasti RDFS-kielen metamallinnusrakenteita tarvitaan. Metamallinnusrakenteita ovat esimerkiksi luokkien määrittely toisille luokille ja ominaisuuksien liittäminen luokkiin. OWL Full -kieli mahdollistaa näiden käytön, mutta tekee päättelystä vähemmän ennustettavaa.

OWL-ontologiakieli on erittäin ilmaisuvoimainen ja soveltuu moneen käyttöön. Suurimmat tietokantojen toimittajat tukevat OWL-ontologiakieltä ja sekä sitä että heikompaa RDFS-kieltä käytetään laajasti yritys- ja WWW-sovelluksissa [Bre10, HHa08]. OWL-ontologiakielen käyttö laajenee jatkuvasti ja laajentuneen käytön myötä siitä on löydetty uusia kehityskohteita [CGr08, HHa08].

Käytännön havaintojen ja kuvauslogiikoiden kehittymisen myötä OWL-ontologiakieltä on voitu kehittää edelleen [CGr08]. Lokakuussa 2009 W3C-yhteenliittymä julkaisi suosituksen OWL 2 -ontologiakielestä [Hit09, MPP09a]. OWL 2 -ontologiakieli on rakenteiltaan OWL- eli nykyään OWL 1 -ontologiakielen laajennos [CGr08], jonka on tarkoitus korvata OWL 1 -ontologiakieli [DSc04]. OWL 2 -ontologiakielessä on huomioitu OWL 1 -ontologiakielen [CGr08] käytössä havaittuja puutteita ja kehitystarpeita [Mot09].

OWL 2 -ontologiakielessä on merkittäviä eroja OWL 1 -ontologiakieleen verrattuna [CGr08, Hit09, Krö12]. Kielen ilmaisuvoimaa on lisätty uusilla rakenteilla, kuten uusilla yhteys- ja tietotyypeillä. Uusien rakenteiden lisääminen jatkossa on tehty helpommaksi. Kielen rakenteita on yksinkertaistettu eli tiettyjen ilmaisujen tekeminen on yksinkertaisempaa. Esimerkiksi metamallinnusrakenteita on yksinkertaistettu. Tämä helpottaa käyttökelpoisten työkalujen kehittämistä ja päättelykoneiden toteutusta mahdollistamalla yksinkertaisempien päättelyalgoritmien käytön. Ontologian rakenteiden annotointimahdollisuudet ovat entistä laajempia. Rakenteet yksilöidään Internationalized Resource Identifiers (IRI) -tunnisteilla [DSu05]. IRI-tunnisteet täydentävät URI-tunnisteita, joiden korvaamiseksi ne on suunniteltu. URI-tunnisteet ovat tietynlaisia IRI-tunnisteita. IRI-tunnisteet sopivat URI-tunnisteita paremmin kansainväliseen käyttöön tukemalla Universal Character Set (UCS) -merkistön merkkejä.

OWL 2 -ontologiakieli jakautuu OWL 1 -kieltä mukaillen seuraaviin osakieliin: *OWL 2 Full*- ja *OWL 2 DL* -kieleen [Hit09]. OWL 2 Full -kieli on RDFS-kielen suoraviivaisena laajennoksena ratkeamaton ja OWL 2 DL -kieli OWL 2 Full -kielen lauseopillisesti rajoitettuna versiona ratkeava. Tämä mahdollistaa täydellisen päättelykoneen toteuttamisen OWL 2 DL -kielelle.

OWL 2 -ontologiakielessä on kaksi tapaa määritellä ontologioiden merkityksiä: *suora semantiikka* (engl. direct semantics) [MPC09] ja *RDF-pohjainen semantiikka* (engl. RDF-based semantics) [CHP09]. Suorassa semantiikassa merkitykset määritellään kuvauslogiikan menetelmillä. RDF-pohjaisessa semantiikassa laajennetaan RDFS-kielelle määriteltyä formaalia semantiikkaa [Hay04], ja merkitykset määritellään RDF-tietomallin mukaisesti. Molemmissa on hyvät ja huonot puolensa [Krö12]. Suora semantiikka on ratkeava, muttei yhteensopiva kaikkien OWL 2 -ontologiakielen ominaisuuksia käyttävien RDF-tietokantojen kanssa. RDF-pohjainen semantiikka on ratkeamaton. OWL 2 DL -kielessä käytetään suoraa semantiikkaa ja OWL 2 Full -kielessä RDF-pohjaista semantiikkaa.

OWL 2 Full- ja OWL 2 DL -kielten lisäksi OWL 2 -suositus esittelee kolme yksinkertaistettua kieltä (engl. profiles) [Krö12, Mot09]: *OWL 2 EL*-, *OWL 2 QL*- ja *OWL 2 RL* -kielen. Näiden jako ei perustu pelkkään ilmaisuvoiman ja ratkeavuuden väliseen tasapainotteluun. Sen sijaan OWL 2 EL-, OWL 2 QL- ja OWL 2 RL -kielet ovat ratkeavia, mutta ratkeavuuden ja helppokäyttöisyyden saavuttamiseksi niiden ilmaisuvoimaa on rajoitettu eri sovellusalueiden tarpeiden mukaan. Niiden tarkoitus on tarjota sovellusalueen kannalta tarkoituksenmukainen ilmaisuvoima ja mahdollistaa tehokas ja helposti toteutettava päättely [Krö12], joten ne ovat erittäin kiinnostavia vaihtoehtoja käytännön toteutuksiin.

OWL 2 EL -kieli vastaa laajojen ja monimutkaisten ontologioiden kehittäjien tarpeisiin rajoittamalla hieman OWL 2 -kielen ilmaisuvoimaa, mutta takaamalla laskennallisen ratkeavuuden [Mot09]. OWL 2 EL -kieli soveltuu laajojen, tehokkaan päättelykoneen vaativien ontologioiden kuvaamiseen.

OWL 2 QL -kieli on suunniteltu yhteensopivaksi perinteisten relaatiotietokantojen rakenteiden ja työkalujen kanssa [Mot09]. Laajoihin ja monimutkaisiin, esimerkiksi ihmisen anatomiaa kuvaaviin, ontologioihin verrattuna relaatiotietokantoja käyttävien sovellusten sovellusalueella ontologiat ovat suhteellisen yksinkertaisia. Niihin kuitenkin liittyy tarve kysellä erittäin suuria määriä ontologian käsitteiden ilmentymiä. OWL 2 QL -kieli vastaa näihin tarpeisiin mahdollistamalla yhdistävien kyselyiden tekemisen relaatiotietokantoihin niiden omilla menetelmillä, kuten Structured Query Language (SQL) -kyselykielellä. Tämä on mahdollista, koska OWL 2 QL -kieli mahdollistaa ontologioiden toteuttamisen suoraan relaatiotietokantojen päälle. Tällöin tieto voidaan varastoida relaatiotietokantaan ja semanttiset kyselyt voidaan uudelleenkirjoittaa ontologioiden perusteella useiksi vastaaviksi SQL-kyselyiksi. OWL 2 QL -kielen ilmaisuvoima riittää yksinkertaisten tesaurusten kaltaisten ontologioiden esittämiseen. Sen ilmaisuvoima vastaa tietokantakaavioiden, yksilö-yhteys (engl. entity-relationship) (ER) -tietomallin ja Unified Modeling Language (UML) -tietomallin ilmaisuvoimaa. OWL 2 QL -kieli soveltuu tilanteisiin, joissa käytetään suhteellisen yksinkertaista ontologiaa, jolla on suuri määrä ilmentymiä, ja joissa tehdään kyselyitä relaatiotietokannan tietueisiin.

OWL 2 RL -kieli on tarkoitettu sovellusalueille, joissa tarvitaan sekä mahdollisimman paljon ilmaisuvoimaa että päättelysääntöjä sisältävän ontologian ja olemassaolevien sääntökoneiden (engl. rule engine) välistä yhteentoimivuutta [Mot09]. Sääntökoneet ovat ohjelmistoja, jotka toimeenpanevat sääntöjä liiketoimintaympäristöissä [OCo05]. OWL 2 RL -kielen ilmaisuvoima riittää laajojen ja monimutkaisten kehittäjien tarpeisiin [Mot09].

OWL 2 RL -kieli tukee relaatiotietokantoja. Säännöillä laajennettujen tietokannanhallintajärjestelmien päälle voidaan toteuttaa päättelykoneita. Sääntöpohjainen toteutus voi käsitellä suoraan RDF-kolmikkolauseita, jolloin sitä voidaan soveltaa mihin tahansa RDF-tietomalliin eli myös OWL 2 -ontologiaan. OWL 2 RL -kieli soveltuu sovellusalueille, joilla voidaan uhrata hieman ilmaisuvoimaa päättelyiden ratkeavuuden takaamiseksi. Se soveltuu niin ikään RDF-tietomallia tai RDFS-kieltä käyttäville sovelluksille, jotka tarvitsevat lisäksi hieman OWL 2 -ontologiakielen sisältämää ilmaisuvoimaa.

OWL 2 -ontologiakielen osakielten väliset suhteet voidaan nähdä seuraavasti [Hit09]: OWL 2 DL -kieli on OWL 2 Full -kielen lauseopillinen osajoukko, OWL 2 EL-, OWL 2 QL- ja OWL 2 RL -kielet ovat OWL 2 DL -kielen lauseopillisia osajoukkoja ja OWL 2 EL-, OWL 2 RL- tai OWL 2 QL -kielet eivät ole toistensa osajoukkoja.

Tähän mennessä on nähty, että yksinkertaisimmillaan ontologia voi olla tesauruksen kaltainen yksinkertaisen käsitehierarkian muodostava sanasto. Toisaalta ontologia voi olla erittäin ilmaisuvoimainen kuvauslogiikka, joka kuvaa kattavan käsitemallin kiinnostuksen kohteena olevasta maailmasta. Näin ollen ontologiat voivat olla yksinkertaisia tai laajoja, ja niitä kuvaavat ontologiakielet ilmaisuvoimaltaan heikkoja tai vahvoja. Heikoimman ja vahvimman ontologiakielen väliin mahtuu joukko ilmaisuvoimaltaan vaihtelevia ontologiakieliä. Fluit ja kumppanit [FSH03] odottavat useimpien semanttisen webin ontologioiden olevan kuvattavan tiedon luonteen vuoksi yksinkertaisia. Yksinkertaisilla ontologioilla he tarkoittavat pääosin luokkien hierarkkisia suhteita kuvaavia ontologioita, joissa on suhteellisen vähän loogisia tai muita yhteyksiä. Tämän johdosta OWL 2 EL-, OWL 2 QL ja OWL 2 RL -kielet ovat semanttisen webin näkökulmasta yksinkertaistettuina, tehokkaan päättelyn takaavina ontologiakielinä erittäin mielenkiintoisia. Kaikkiin käyttötapauksiin yksinkertaiset ontologiat eivät kuitenkaan riitä, jolloin ontologioiden

kuvaamiseen kannattaa valita laajemmasta valikoimasta sovellusalueen vaatimuksiin nähden tarkoituksenmukaisin ontologiakieli.

3.6 Sarjallistaminen

Kaaviona esitetty tieto on selkeälukuista ihmisille, mutta semanttisessa webissä tieto on tallennettava koneymmärrettävästi. Tämän johdosta tieto on esitettävä joidenkin koneelliseen käsittelyyn, kuten lukemiseen, tallentamiseen ja tiedonsiirtoon soveltuvien rakennesääntöjen mukaisesti eli *sarjallistettuna* [Vra09]. RDF-tietomalli ja sen RDFS-kielellä tehty laajennos voidaan sarjallistaa eri muotoihin eri käyttötapauksia varten.

Ihmiselle helppolukuisimpia sarjallistamismenetelmiä ovat *N-Triples*- [BBa01], *Notation 3* (N3)- [BCo11] ja *Turtle* -kieli [BBe11]. N3-kieli on tiivis sarjallistamismenetelmä, jonka tavoitteena on olla mahdollisimman luonnollinen, luettava ja miellyttävä käyttää. Sarjallistamiseen käytettävän lauseopin määrittelyn lisäksi Turtle-kieli laajentaa RDF-tietomallia mahdollistaen esimerkiksi loogisten ilmaisujen ja päättelysääntöjen ilmaisemisen tiedon yhteydessä. Turtle-kieli on N3-kielen osakieli, joka tarjoaa ihmiselle helppolukuisen sarjallistamismuodon laajentamatta RDF-tietomallia. N-Triples-kieli on suunniteltu N3- ja Turtle -kieliä yksinkertaisemmaksi. Se on Turtle-kielen osakieli. N3-, Turtle- ja N-Triples -kielillä sarjallistettu tietomalli on yksinkertaisessa tekstimuodossa.

Enimmäkseen koneille tarkoitettuun ilmaisuun käytetään XML-merkkauskieltä [BPS98]. XML-merkkauskielen käyttö on W3C-yhteenliittymän suosittelema tapa RDF-tietomallin sarjallistamiseen [Bec04b]. RDF-tietomallin XML-kieleen pohjautuvaa sarjallistamismuotoa nimitetään RDF/XML-muodoksi. RDF/XML-muotoon sarjallistettu tieto on koneluettavaa, muttei helppolukuista ihmiskäyttäjille. Tämä on kuitenkin pieni ongelma, sillä ihmiskäyttäjien ei ole

tarkoitus käyttää semanttisen webin RDF-muotoista tietoa suoraan, vaan heille suunniteltujen helppokäyttöisten työkalujen avulla [AHa04, sivu 69].

Perinteisille WWW-sivuille voidaan sisällyttää RDF-tietomallin mukaisia annotaatioita Resource Description Framework in attributes (RDFa) -määritteillä [ABi08]. RDFa-määritteet lisätään WWW-sivujen lähdekoodiin. Niillä voidaan kuvata WWW-sivujen sisältöjä hakukoneiden tai muiden WWW-sivustoilta tietoa etsivien sovellusten käyttöön. Esimerkiksi *Google*-, *Bing*- ja *Yahoo!* -hakukoneet tukevat RDFa-määritteitä [Goe08, Mar12, STH10]. RDFa-määritteiden käyttö WWW-sivuilla on toistaiseksi vähäistä, mutta niiden avulla on mahdollista saavuttaa liiketoiminnan kannalta merkittäviä etuja [STH10]. Hakukoneelle ymmärrettävän sisällön sisältävä sivu sijoittuu usein korkeammalle sivun aihepiiriin liittyvien hakujen hakutuloksissa.

OWL 1 -ontologiakielellä kuvattu ontologia sarjallistetaan RDF/XML-muodossa [Bec04b, HPH03]. OWL 2 -ontologiakieli voidaan sarjallistaa RDF/XML-muodossa, mutta se ei ole pakollista [CGr08]. Vaihtoehtoina ovat esimerkiksi helpommin käsiteltävä OWL 2 XML -muoto [MPP09b] ja Turtle-muoto [BBe11]. Kaikkien OWL 2 -ontologiatyökalujen on tuettava RDF/XML-muotoa [Bao09].

3.7 Kyselyt

Semanttisesti annotoidun eli ontologioiden avulla kuvatun tiedon hyötyjen esillesaamiseksi tietomalliin on voitava tehdä sitä ilmaisuvoimaltaan vastaavia kyselyitä. Kyselyiden tekemiseen tarvitaan soveltuvia eli tiedon kuvaamiseen käytetyn annotointikielen ilmaisuvoimaa vastaavia kyselykieliä. Perinteisesti annotoituun tai annotoimattomaan tietoon verrattuna kyselyitä voidaan tehdä merkittävästi ilmaisuvoimaisemmilla kyselyillä. Semanttisen webin menetelmät mahdollistavat esimerkiksi *kyselyiden muokkaamisen* (engl. query modification) ja moniselitteisten termien merkitysten *yksiselitteistämisen* (engl.

disambiguation) [Man07]. Kyselyjen muokkaaminen parantaa hakutulosten saantia ja yksiselitteistämisen tarkkuutta. Tämä on hyödyllistä, sillä hakukoneiden tehtävänä on tuottaa mahdollisimman suuri saanti ja tarkkuus [Man07]. Näin ollen riittävän ilmaisuvoimaiset kyselykielet ovat käytännön sovellusten kannalta erittäin tärkeitä [CGr08].

RDF-tietomallia voidaan tarkastella kolmella tasolla: *lauseopillisella* (engl. syntactic), *rakenteellisella* (engl. structure) ja *semanttisella* (engl. semantic) [HBS09]. Lauseopillisella tasolla RDF-asiakirjat ovat esimerkiksi XML-normin mukaisia. Rakenteellisella tasolla RDF-asiakirjat koostuvat joukosta RDF-kolmikkolauseita. Semanttisella tasolla RDF-asiakirjat muodostuvat yhdestä tai useammasta verkosta, jotka sisältävät osittain ennalta määrättyjä merkityksiä. RDF-asiakirjoja voidaan kysellä jokaisella kolmella tasolla.

Lauseopillisella tasolla kyselyitä voidaan tehdä jotakin XML-asiakirjoja varten suunniteltua kyselykieltä käyttäen, mikäli RDF-tietomalli on tallennettu XML-muotoisena [HBS09]. Tällöin ei kuitenkaan huomioida, että RDF verkkomuotoisena tietomallina eroaa merkittävästi XML-normin mukaisesta puurakenteesta. RDF-tietomallin kuvaamia XML-puurakenteessa näkymättömiä yhteyksiä on erittäin vaikea kysellä XML-kyselykielillä.

Rakenteellisella tasolla RDF-tietomalli koostuu kolmikkolauseista, joiden kyselyyn on olemassa erilaisia kyselykieliä [HBS09]. Tällaisten kyselykielten käytössä on huomioitava, että RDF-tietomalli nähdään kokonaisuudessaan kolmikkolauseiden kokoelmana, RDFS-kuvauksessa erityisiä merkityksiä saaneet osatekijät mukaan lukien. Tämän johdosta joitakin olemassa olevia yhteyksiä jätetään rakenteellisen tason kyselyissä huomioimatta.

Semanttisella tasolla kysellään tietämystä, jonka RDF-tietomalli kokonaisuudessaan esittää. Semanttisen tason kyselyitä voidaan tehdä ainakin

kahdella tavalla [HBS09]. Ensimmäinen tapa on laskea ja tallentaa kyselyiden pohjaksi kaikki lauseet, jotka voidaan päätellä alkuperäisten lauseiden perusteella. Toinen tapa on antaa kyselynkäsittelijän tehdä päättelyä kyselyiden yhteydessä. Kyselykielen valinnassa on syytä huomioida, että suurin osa RDF-kyselykielistä on suunniteltu tekemään kyselyitä yksinkertaiseen kolmikkomuotoisen tietoon. Tällaiset kyselykielet eivät osaa erottaa tietoon ja käsitemalliin liittyviä yhteyksiä toisistaan.

SPARQL Query Language for RDF (SPARQL) -kieli [PSe08] on W3C-yhteenliittymän suosittelema semanttisen webin kyselykieli. Sillä voidaan kysellä RDF-tietomallin mukaisessa muodossa olevaa tietoa semanttisella tasolla. Kyselykielen mahdollistamien kyselyiden ymmärtämiseksi sen ilmaisuvoiman määrittäminen on erittäin tärkeää [AGu08]. Kyselyiden käsittelyn monimutkaisuuden ymmärtäminen on niin ikään tärkeää [PAG09]. Ilmaisuvoiman ja monimutkaisuuden välisen tasapainon ansiosta *relaatioalgebra* on yksi tutkituimmista ja suosituimmista kyselykielistä [AGu08]. Relaatioalgebra on ”relaatiotietokantoihin liittyvät operaatiot määrittelevä matemaattinen malli, jossa tietokantakysely esitetään näihin operaatioihin perustuvina lausekkeina” [Lai97]. *Relaatiokalkyyli* on ”predikaattilogiikkaan perustuva tapa esittää tietokantakysely loogisena lausekkeena, joka kyselyn tuloksen on täytettävä” [Lai97]. Coddin [Cod72**]** mukaan relaatioalgebra vastaa ilmaisuvoimaltaan relaatiokalkyyliä ja siten predikaattilogiikkaa. SPARQL-kieli vastaa ilmaisuvoimaltaan relaatioalgebraa [AGu08], joten se on ilmaisuvoimaltaan predikaattilogiikkaa eli OWL-ontologiakielten ilmaisuvoimaa vastaava kyselykieli.

Suurin osa SPARQL-kielellä tehtävien kyselyiden muodoista sisältää joukon *kolmikkomalleja* (engl. triple patterns) [PSe08]. Kolmikkomallit ovat RDF-tietomallin kolmikkolauseita vastaavia subjektista, predikaatista ja objektista koostuvia rakenteita. SPARQL-kielessä jokainen lauseenjäsen voidaan kuitenkin korvata ennalta tuntemattomalla osalla eli *muuttujalla*. Kolmikkomallien joukko

muodostaa *kyselyhahmon* (engl. basic graph pattern), jonka vastaavuuksia RDF-tietomallista etsitään. Kyselyhahmo täsmää kyselyn kohteena olevan RDF-tietomallin verkon osan eli *aliverkon* kanssa, kun kyselyhahmon muuttujat voidaan korvata aliverkon RDF-termeillä ja tuloksena saadaan aliverkkoa vastaava RDF-tietomalli. RDF-termejä ovat RDF-tietomallin sisältämät IRI-tunnisteet, literaalit ja tyhjät solmut. RDF-tietomallissa tyhjä solmu on RDF-lauseen erikoistapaus, jossa resurssia ei ole nimetty yleiskäyttöisellä URI-tunnisteella. Sarjallistamisen yhteydessä tyhjä solmu voidaan nimetä, mutta siihen ei voida viitata kyseisen RDF-asiakirjan ulkopuolelta.

Lauseopillisesti SPARQL-kyselykielellä tehtävä kysely esitetään lohkona, joka sisältää *kyselytavan* (engl. query form), mahdollisia *RDF-tietolähdettä rajoittavia ehtoja* (engl. dataset clauses), *kyselyjä rajoittavia avainsanoja* ja mahdollisia *muuntimia* (engl. solution modifiers) [AGu08]. SPARQL-kielessä on neljä erilaista kyselytapaa [PSe08]: *SELECT*-, *CONSTRUCT*-, *DESCRIBE*- ja *ASK* -muoto. SELECT-tapa on verrattavissa SQL-kielen SELECT-lauseeseen. Se etsii RDF-tietolähteestä kyselyhahmoa vastaavan aliverkon ja palauttaa kyselyn tulokset RDF-termeihin sidottujen muuttujien mukaisen taulukon muodossa. CONSTRUCT-tapa toimii SELECT-tavan tavoin, mutta palauttaa kyselyn tulokset muuttujista rakennettuna RDF-tietomallina. RDF-tietomalli on kyselijän toimesta vapaasti määriteltävissä muuttujien avulla, mutta sen on muodostuttava hyväksyttävistä RDF-lauseista. DESCRIBE-tavalla tehty kysely palauttaa kyselyssä määriteltyjä resursseja kuvaavista RDF-lauseista koostuvan RDF-tietomallin. Se antaa kyselyn vastaanottajalle vapauden päättää, mikä osa tiedosta on syytä liittää kyselyn tulokseen. Tällöin kyselijän ei tarvitse tuntea täysin kyselyn kohteena olevan RDF-tietomallin rakennetta. ASK-tapa palauttaa Boolen binäärisen totuusarvon sen perusteella, löytyykö kyselyä vastaavaa tietosisältöä. Kyselyn tulos on *tosi*, mikäli RDF-tietolähteestä löytyy kyselyhahmon kanssa täsmäävä aliverkko. Muuten *epätosi*. SELECT- ja ASK -muodossa tehtyjen kyselyjen tulokset voidaan sarjallistaa XML-muotoon [BBr08]. CONSTRUCT- ja DESCRIBE -muotoisten kyselyjen tulokset ovat RDF-

tietomallin mukaisessa muodossa [PSe08], joten ne voidaan sarjallistaa jotakin RDF-tietomallin sarjallistamismuotoa käyttäen.

RDF-tietolähdettä rajoittavia ehtoja voidaan määritellä *FROM*- ja *FROM NAMED* -rakenteiden avulla [PSe08]. FROM-rakenteella määritellään verkko (engl. graph), josta tietoa haetaan. FROM NAMED -rakenteella määritellään tietolähteeksi *nimetty verkko* (engl. named graph). Verkkoihin viitataan niiden IRI-tunnisteilla. FROM NAMED -rakennetta käytettäessä verkon muodostavat RDF-lauseet yhdistetään SPARQL-käsittelijän toimesta verkon IRI-tunnisteeseen. Tämän perusteella on esimerkiksi mahdollista selvittää, mistä verkosta tietty RDF-lause on peräisin. Mikäli FROM- tai FROM NAMED -rakenteita ei käytetä, haku kohdistuu hakupalvelun tarjoajan RDF-tietovarastoon.

Kyselyhahmo ilmaistaan *WHERE*-rakenteella, jonka perusteella RDF-tietolähteestä etsitään täsmäävää aliverkkoa [PSe08]. Kyselyhahmo kuvataan joukkona allekkain sijoitettuja kolmikkolauseita. Kyselyn tuloksia on mahdollista muuntaa SQL-kielen tapaan esimerkiksi *DISTINCT*-, *ORDER BY*-, *PROJECTION*-, *LIMIT*- ja *OFFSET* -muuntimilla [PAG09].

SPARQL-kielessä määritellään lauseenjäseniä seuraavasti [PSe08]: pienemmyys- ja suuremmuusmerkin sisään voidaan määritellä IRI-tunniste, lainausmerkkien sisään voidaan määritellä literaali ja muuttujan nimi voidaan ilmaista liittämällä muuttujaa kuvaavan termin eteen kysymysmerkki. Esimerkiksi: *<http://www.esimerkki.fi/esim1>*, *"100"* ja *?muuttuja*. IRI-tunnistetta tai muuttujaa voidaan käyttää jokaisen lauseenjäsenen paikalla. Literaaliarvo voi olla vain objektin paikalla. SPARQL-kieli sisältää XML-nimiavaruuksien kaltaisen *PREFIX*-mekanismin, jolla IRI-tunnisteet voidaan ilmaista helppolukuisemmassa muodossa.

SPARQL-kielen ilmaisuvoimaa voidaan havainnollistaa DBpedia-palvelun [Aue07, Biz09] avulla. DBpedia-palvelu on hanke, jonka tavoitteena on muuntaa Wikipedia-palvelun [Wik12] tietosisältöjä rakenteiseen, semanttisen webin menetelmillä kuvattuun ja keskinäisiä yhteyksiä sisältävään muotoon. Tämä mahdollistaa SPARQL-kyselyiden tekemisen DBpedia-palvelun tietosisältöihin. Kuva 4 esittää yhden tavan luonnollisen kielen kyselyn ”*näytä kaikki vuoden 1969 jälkeen Suomessa syntyneet National Hockey League (NHL) -jääkiekkoliigassa pelanneet jääkiekkoilijat, joiden kätisyys on vasen, jotka ovat pituudeltaan yli 180cm, ja jotka ovat uransa aikana pelanneet Ruotsin Elitserien-sarjassa*” esittämiseksi DBpedia-palvelun SPARQL-hakukäyttöliittymässä [SPA12].

SPARQL-kieli mahdollistaa tiedon hakemisen. Sillä ei voida päivittää tietoa. SPARQL-kieli sallii kuitenkin päivityksiä tukevien laajennusten toteuttamisen [Sea08]. Tiedon päivittämisen mahdollistava laajennus on esimerkiksi SPARQL 1.1 Update -kieli [GPP12], joka on ehdolla W3C-yhteenliittymän suositukseksi.

Kuten edellä on esitetty, vahvan teoreettisen pohjan lisäksi W3C-yhteenliittymä on määritellyt joukon menetelmiä, joilla teorioita voidaan soveltaa käytännössä. Tämän lisäksi semanttinen web hyödyntää internetissä toimiviksi todettuja menetelmiä ja normeja, joihin kuuluvat Hypertext Transfer Protocol (HTTP) -yhteyskäytäntö, IRI-tunnisteet sekä XML-merkkauskieli [HHa08]. Tämä mahdollistaa semanttisen webin tietämyksen jakamisen ja järjestelmien välisen yhteentoimivuuden. Kun yhteisesti käytettävät menetelmät ja normit yhdistetään vahvaan teoreettiseen pohjaan, voidaan todeta semanttisen webin menetelmillä olevan luotettava ja laajassa käytössä oleva eri toimijoiden välistä semanttista yhteentoimivuutta vahvistava perusta. Kuva 5 esittää yhden tavan havainnollistaa tätä perustaa. Semanttisen webin kerrosarkkitehtuurissa ylemmät tasot pohjautuvat alempiin. Kerrosarkkitehtuuri päivittyy tutkimuksen edistyessä ja kehittyvien normien vakiintuessa.

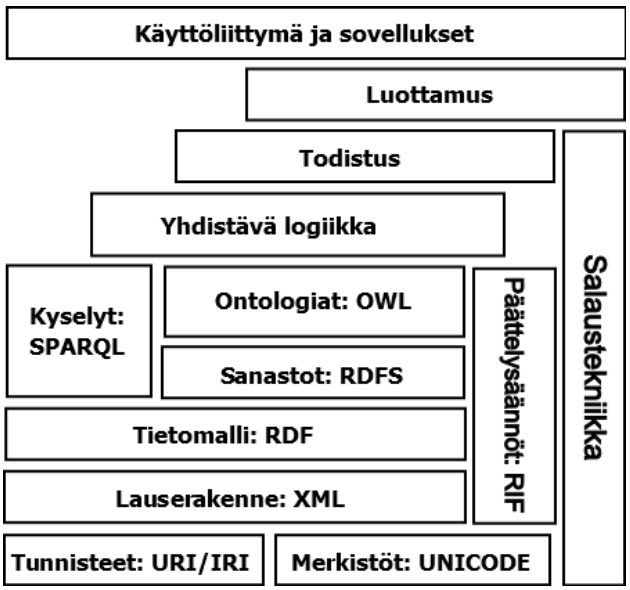
SELECT–muotoinen kysely SPARQL–kielellä:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbpedia2: <http://dbpedia.org/property/>
SELECT DISTINCT ?Nimi ?Syntymaeaika ?Pituus ?Kaetisyys ?Kuvaus
WHERE {
    ?person foaf:name ?Nimi .
    ?person dbo:birthPlace ?Syntymaepaikka .
    ?person dbo:birthDate ?Syntymaeaika .
    ?person dbo:shoots ?Kaetisyys .
    ?person dbo:height ?Pituus .
    ?person dbo:abstract ?Kuvaus .
    ?person dbo:league :National_Hockey_League .
    ?person dbo:formerTeam :Elitserien .
    ?person dbo:birthPlace :Finland .
    FILTER (?Syntymaeaika >= "1970-01-01"^^xsd:date) .
    FILTER (?Kaetisyys = "Left"@en) .
    FILTER (?Pituus > 1.80) .
    FILTER (LANG(?Kuvaus) = 'fi') .
}
ORDER BY ?Nimi
```

Kyselyn tulokset DBpedia-palvelun SPARQL-hakukäyttöliittymässä:

Nimi	Syntymaeaika	Pituus	Kaetisyys	Kuvaus
"Niklas Backstrom"@en	"1978-02-13"^^xsd:date	1.8542	"Left"@en	"Niklas Oskar Bäckström on suomalainen jääkiekkomaalivahti, joka edustaa tällä hetkellä NHL-seura Minnesota Wildia. [...] Ruotsin Elitserienissä hän o n-pelannut kaudella 2000-01 AIK IF:n joukkueessa [..]"@fi
"Olli Jokinen"@en	"1978-12-05"^^xsd:date	1.905	"Left"@en	"Olli Jokinen suomalainen jääkiekkoilija, joka edustaa tällä hetkellä NHL-liigassa pelaavan Calgary Flamesin joukkuetta [...]"@fi

Kuva 4: SELECT-muotoinen kysely SPARQL-kielellä ja kyselyn tulokset DBPedia-palvelun SPARQL-hakukäyttöliittymässä.



Kuva 5: Semanttisen webin kerrosarkkitehtuuri [Bra07].

4 Perinnetieto ja semanttisen webin menetelmät

Perinnetietoa sisältäviin relaatiotietokantoihin voidaan tehdä kyselyitä SQL-kielen kaltaisilla kyselykielillä. SQL-kieleen pohjautuvat kyselyt sisältävät täsmällisesti määritellyn formaalin semantiikan ja vastaavat ilmaisuvoimaltaan SPARQL-kielen kyselyitä [AGu08]. Kyselyiden mahdollistamiseksi tiedon on oltava tiukkojen tietokantakohtaisten ja kyselykielen käyttäjän ymmärtämien kaaviomäärittelyjen mukaisesti järjestettynä [Hul01]. Kaavioiden pohjalta ennalta määritellyt kyselyt eivät sovellu kaikkiin käyttötapauksiin [MVo00]. Tämän johdosta ongelmaan on etsitty ratkaisuita, joiden avulla relaatiotietokantoihin voidaan tehdä SQL-kieleen pohjautuvien kyselyiden lisäksi avainsanaperustaisia hakuja [MVo00]. Avainsanaperustaiset haut mahdollistavat tiedon etsimisen helposti ja haun kohteena olevien relaatiotietokantojen kaaviomäärittelyjä tuntematta, mutta ne kärsivät luvussa 2.2 kuvatuista perinteisen tiedonhaun ongelmista. Kuten mitä tahansa kohdetta, relaatiotietokantojen sisältämää tietoa voidaan annotoida semanttisen webin menetelmillä. Tämä mahdollistaa SPARQL-kieleen pohjautuvien kyselyiden

tekemisen annotoituun tietoon. Tällöin on mahdollista hallita tietämystä merkittävästi aiempaa tehokkaammin.

Relaatiotietokannat ovat yksi yleisimmistä tavoista säilöä tietoa [Sel07, Yin04], joten niihin on pitkällä aikavälillä tallennettu valtava määrä perinnetietoa. He ja kumppanit [He07] ovat arvioineet relaatiomalliin [Cod70] perustuvien relaatiotietokantojen käyttöä internetissä. Heidän tutkimustensa mukaan suurin piirtein 70% WWW-sivustoista käyttää tietovarastoinaan relaatiotietokantoja, ja internetin kautta löydettävissä olevat relaatiotietokannat sisältävät jopa 500 kertaa enemmän tietoa WWW-palvelun muuttumattomaan (engl. static) tietosisältöön verrattuna. Tämän johdosta relaatiotietokantojen tietosisältöjen kuvaamisen semanttisen webin tietomalliin yhteensopivalla tavalla on todettu olevan avainasemassa semanttisen webin toteutumisen kannalta [Seq11, TSM08]. Tämä on kannustanut kehittämään menetelmiä, joilla relaatiotietokantojen sisältämää tietoa voidaan esittää semanttisesti annotoituna. Menetelmät soveltuvat tämän tutkielman kiinnostuksen kohteena olevan perinnetiedon semanttiseen annotointiin, joten niiden tarkastelu on erittäin mielenkiintoisia. Luvun 3 perusteella koneymmärrettävien semanttisten annotaatioiden mahdollistamiseksi tarvitaan ontologioita.

4.1 Ontologioiden kehittäminen

Ontologioiden kehittäminen on yksi tärkeimmistä semanttisen webin menetelmiin liittyvistä tutkimusalueista [GPr09]. Ontologioilla on muiden ohjelmistotuotannon tuotteiden tavoin elinkaari: niitä suunnitellaan, toteutetaan, arvioidaan, korjataan, hyödynnetään ja uudelleenkäytetään. Ontologioiden ymmärtämisen monimutkaisuus ja niiden kaikkia elinkaaren hallintaa helpottavien työkalujen pieni määrä hidastavat osaltaan semanttisen webin menetelmien laajaa käyttöönottoa. Tämän johdosta semanttisen webin rakennuspalikoiksi tarvitaan ontologioiden kehittäjien saatavilla olevia

yksinkertaisia, ymmärrettäviä ja helposti muokattavissa olevia ontologioita. Ontologioiden kehitystyön helpottamiseksi on alettu tutkia ontologioiden suunnittelumalleja ja kehittää ontologioiden automaattisen tai puoliautomaattisen luomisen mahdollistavia työkaluja. Näin ollen on odotettavissa, että ontologiakehitys helpottuu tulevaisuudessa. Perinnetiedon semanttisen annotoinnin näkökulmasta työkalujen ja käytettävissä olevien ontologioiden lisääntyminen on hyödyllistä, muttei ratkaisevan tärkeää. Tämä johtuu kapeammasta sovellusalueesta, jossa ontologioiden elinkaaren hallinta on vastaavasti helpompaa.

Ontologioiden kehittämiseen on eri vaihtoehtoja [AHa04, sivu 205]. Niitä voidaan kehittää tarpeen mukaan käsin tai puoliautomaattisten menetelmien avulla. Tämän lisäksi olemassa olevia ontologioita voidaan uudelleenkäyttää. Sovelluskehittäjien lisäksi ontologioiden kehitystyössä tarvitaan usein aihealueen asiantuntijoita määrittelemään käsitteitä ja niiden välisiä suhteita. Koska ontologian on tarkoitus olla sen käyttäjien eli toimijoiden jakama ymmärrys aihealueen käsitteiden välisistä suhteista [SBF98], sen kehitystyöhön on syytä ottaa mukaan tulevat toimijat. Ontologioita voidaan kehittää hajautetusti useiden asiantuntijoiden ja järjestöjen yhteistyönä [Dev02]. Etenkin suuria ontologioita kehitettäessä työ kannattaa jakaa helpommin ymmärrettäviin osakokonaisuuksiin [Hor11]. Tällöin suuret ontologiakokonaisuudet ovat ihmiselle helpommin ymmärrettävissä ja kehitystyön jakaminen rinnakkain usean suunnittelijan kesken on yksinkertaisempaa. Eri ontologioita voidaan yhdistää suuremmiksi kokonaisuuksiksi luomalla niiden käsitteiden välille yhteyksiä [HHa08].

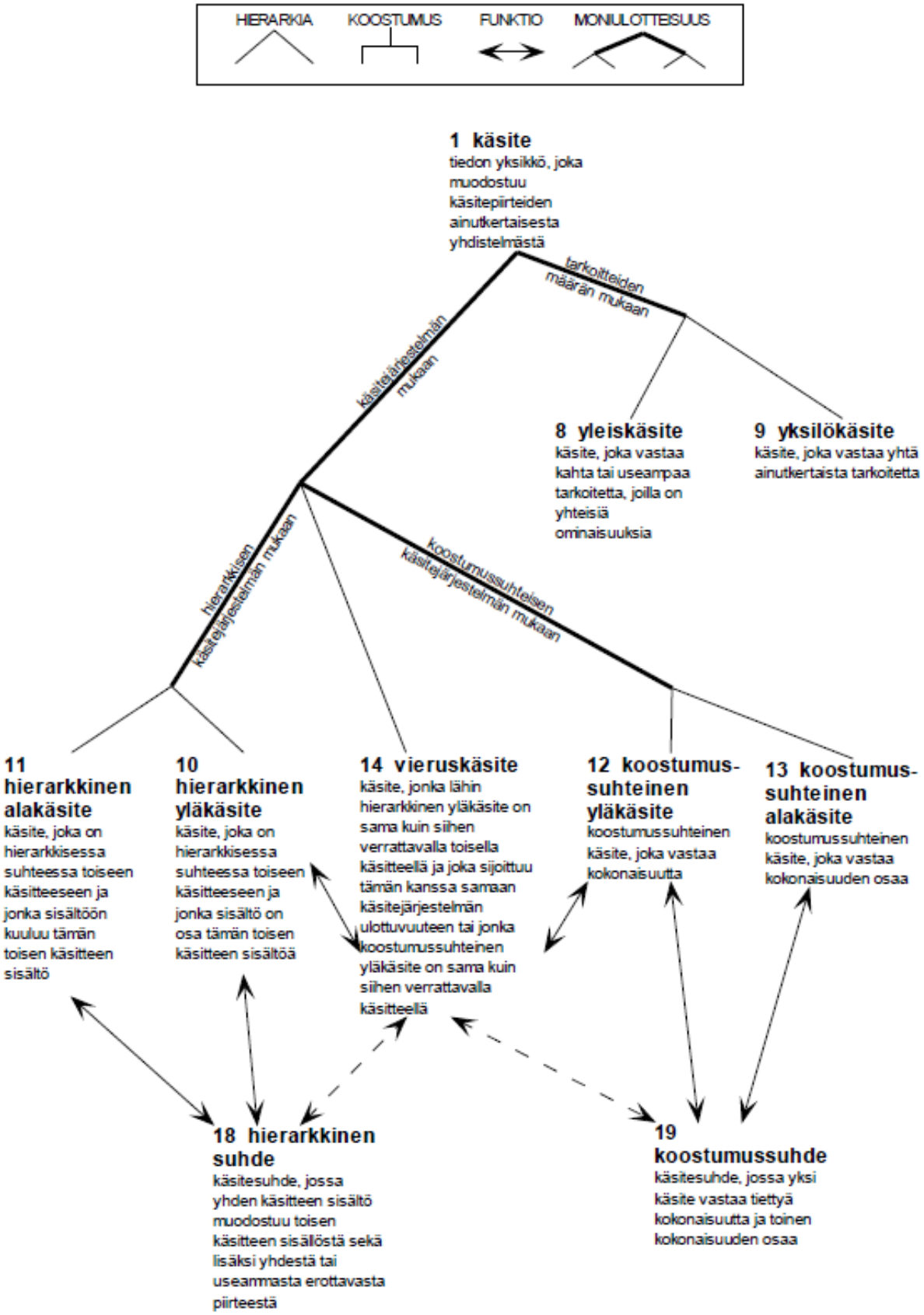
Ontologian kehitystyössä määritellään kiinnostuksen kohteena olevan aihealueen käsitteitä ja niiden välisiä suhteita. Sanastokeskuksen [TSK06] mukaan käsitteet ovat ympäröivän maailman kohteista eli *tarkoitteista* muodostettuja ajatusmalleja. Tarkoitteilla on erilaisia sisäisiä ja toisiinsa liittyviä ominaisuuksia, joista muodostettuja ajatusmalleja kutsutaan

käsitepiirteiksi. Käsite muodostuu erilaisista käsitepiirteistä, joista olennaiset kuvataan kielellisesti määritelmän avulla. Termit ovat käsitteiden nimityksiä, joilla voidaan viitata lyhyesti käsitteen koko sisältöön.

Sanastokeskuksen [TSK06] mukaan terminologisen sanastotyön tärkein työvaihe on *käsiteanalyysi*, jossa ”selvitetään kunkin käsitteen olennainen sisältö, käsitteiden väliset suhteet ja näiden suhteiden perusteella muodostuvat käsitejärjestelmät”. Käsiteanalyysin pohjalta kirjoitetaan määritelmiä ja valitaan sopivia termejä. Käsiteanalyysissä käytetään yleensä kolmenlaisia käsitesuhteita: *hierarkkisia, koostumus- ja funktiosuhteita.* Hierarkkisella suhteella kuvataan laajemman yläkäsitteen ja suppeamman alakäsitteen välistä yhteyttä. Koostumussuhteella kuvataan alakäsitteiden olevan osia yläkäsitteiden kuvaamasta kokonaisuudesta. Suhteet, joita ei voida luokitella hierarkkisiksi tai koostumussuhteiksi, kuvataan usein funktiosuhteina.

Ontologiat ovat usein *sekakoosteisia* ja *moniulotteisia* [TSK06]. Ontologia on sekakoosteinen, kun siinä esiintyy useita erilaisia käsitesuhdetyyppejä. Ontologia on moniulotteinen, kun yläkäsitteestä voidaan päästä eri alakäsitevalikoimiin eri jaotteluperusteita käyttäen. Kuva 6 havainnollistaa erilaisia käsitesuhdetyyppejä.

Ontologiakehityksessä voidaan erottaa seuraavat työvaiheet [AHa04, sivut 206-209]: *laajuuden määrittäminen*, *uudelleenkäytön harkinta*, *termien luetteloiminen*, *taksonomian määrittely*, *ominaisuuksien määrittely*, *näkökulmien määrittely*, *ilmentymien määrittely* ja *poikkeuksien tarkistus*. Laajuuden määrittämisessä päätetään, miten ontologian avulla kuvattava aihealue rajataan. Aihealueen perusteella ei voida määritellä oikeanlaista ontologiaa, sillä aihealue voidaan mallintaa eri näkökulmista. Tällöin samaa aihealuetta kuvaavat ontologiat ovat vaihtoehtoja toisilleen. Tarkoituksenmukaisen ontologian määrittävät aihealueen lisäksi sovellusalueen vaatimukset eli



Kuva 6: Käsitesuhdetyypit [TSK06, sivu 15].

ontologian tuleva käyttötarkoitus. Laajuuden määrittämisessä määritellään ontologian käyttötarkoitus ja odotettavissa olevat jatkokehitykseen liittyvät asiat.

Uudelleenkäytön harkinnassa selvitetään, voidaanko olemassa olevia ontologioita käyttää joko sellaisenaan tai kehitystyön pohjana [AHa04, sivu 206]. Esimerkiksi Suomessa Aalto-yliopiston johdolla on kehitetty kymmeniä eri aihealueita kuvaavia ontologioita, joita on julkaistu avoimena palveluna muiden toimijoiden käyttöön [Ont11]. Tämän lisäksi olemassa olevat asiasanastot, taksonomiat ja tesaurukset voivat olla hyödyllisiä lähteitä, joiden pohjalta ontologian kehitys voidaan aloittaa.

Termien luetteloinnissa kirjoitetaan ylös olennaisimmat termit, joiden odotetaan löytyvän ontologiasta [AHa04, sivu 206-207]. Tässä vaiheessa luettelointi ei sisällä rakenteellisuutta. Työvaiheen tuloksena saatavan luettelon substantiivit vastaavat yleensä ontologiaan tulevien luokkien nimiä ja verbit ominaisuuksien nimiä.

Taksonomian määrittelyssä luetteloidut termit järjestetään taksonomiaksi [AHa04, sivu 207]. Työvaiheessa on olennaisen tärkeää varmistua, että määritelty hierarkia on puhtaasti taksonomisia yläkäsite-alakäsite-suhteita sisältävä. Ominaisuuksien määrittely lomittuu usein taksonomian määrittelyyn [AHa04, sivu 207]. Tässä työvaiheessa määritellään käsitteitä kuvaavia ja yhdistäviä ominaisuuksia, jotka lisätään taksonomiaan. Toisin sanoen kuvaukseen lisätään puhtaasti taksonomisten suhteiden lisäksi muita käsitesuhdetyyppejä. Ominaisuuksia liitetään helposti mahdollisimman ylhäällä taksonomiassa esiintyviin käsitteisiin, jolloin ne pätevät kaikkiin alemman tason käsitteisiin. Tällöin kannattaa huomioida, että alhaisimmille tasoille tehty mahdollisimman kapea ominaisuuksien määrittely on usein hyödyllisempää. Tämä johtuu siitä, että tällöin on helpompi havaita ontologiaan rakennettuja

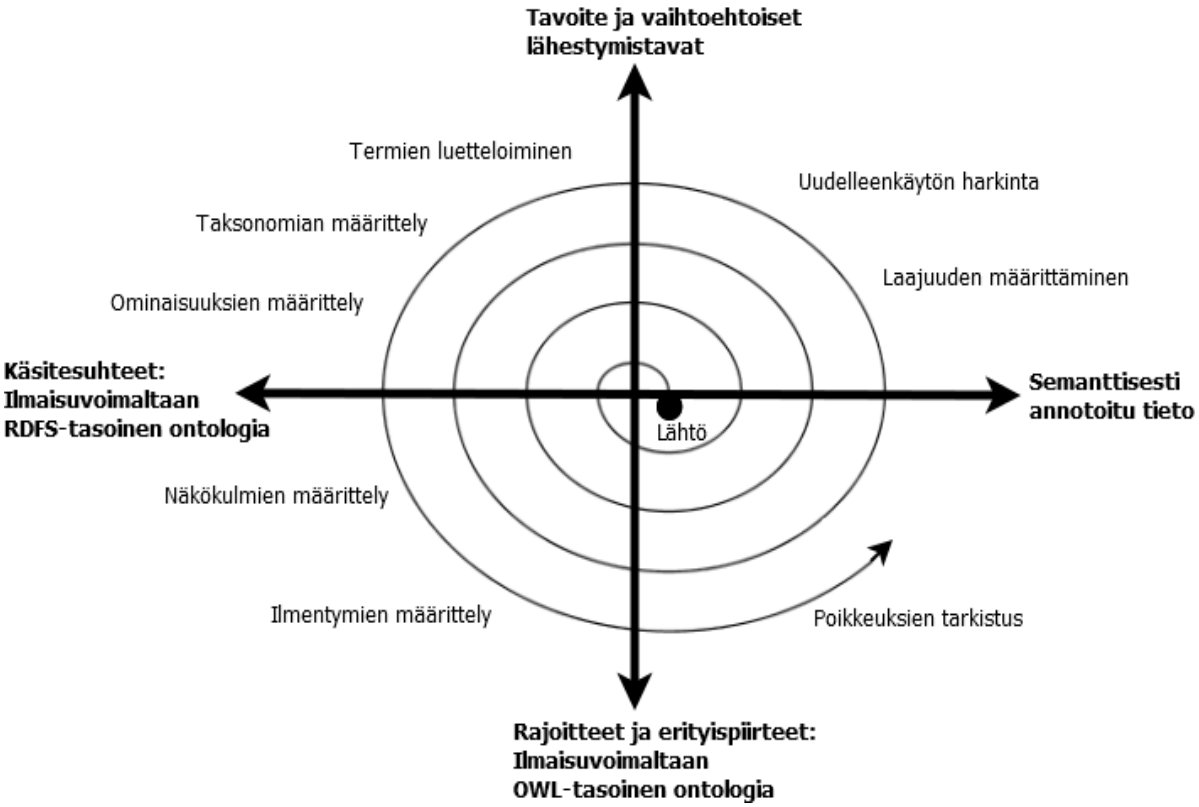
epäjohdonmukaisuuksia ja väärinymmärrettyjä asioita. Ontologian arviointi ei kuitenkaan ole mahdollista vielä tässä vaiheessa. Tämä johtuu siitä, että tässä vaiheessa on määritelty ontologia, jonka sisältämät ilmaisut voidaan esittää RDFS-kielellä. RDFS-kielen ilmaisuvoima ei riitä ilmaisemaan epäjohdonmukaisuuksia, jotka johtuvat usein ominaisuuksien rajoitteiden virheellisestä määrittelystä.

Näkökulmien määrittelyssä lisätään aiemmin kuvatuille ominaisuuksille OWL-ontologiakielten ilmaisuvoimaa vaativia arvojoukko- ja kardinaalisuusrajoitteita sekä päättelymahdollisuuksia lisääviä erityispiirteitä, kuten symmetrisyys, transitiivisuus, käänteisyys ja funktionaalisuus [AHa04, sivut 207-208]. Ilmentymien määrittelyssä tuotetaan ontologian rakenteiden avulla annotoitavaa tietoa [AHa04, sivu 208]. Tämän tutkielman tapauksessa tieto haetaan relaatiotietokannoista ja muunnetaan ontologian ilmentymiksi. Ilmentymät esitetään RDF-tietomallin mukaisesti. Työvaiheen lopputuloksena tieto on semanttisen ja koneymmärrettävän metatiedon avulla annotoituna ja semanttisen webin tietomallin mukaisesti esitettynä.

Viimeisessä työvaiheessa eli poikkeuksien tarkistuksessa arvioidaan tuotettu ontologia ja sen ilmentymät [AHa04, sivu 209]. Työvaiheessa pyritään löytämään ontologiaan tai sen ilmentymiin liittyviä epäjohdonmukaisuuksia. Tässä voidaan hyödyntää päättelykoneita [AHa04, sivu 219]. Ne voivat ontologian pohjalta tekemänsä loogisen päättelyn avulla havaita ontologiassa esiintyviä epäjohdonmukaisuuksia. Epäjohdonmukaisuuksien etsinnän lisäksi poikkeuksien tarkistuksessa varmistetaan, että ontologia vastaa sovellusalueen vaatimuksia [AHa04, sivu 219].

Käytännössä ontologiakehityksen työvaiheet eivät etene suoraviivaisesti, vaan tavallisten kehitysmenetelmien tapaan ne ovat luonteeltaan toistuvia [AHa04, sivu 206]. Kaikkea ei ratkaista kerralla, vaan tuotokset tarkentuvat vaiheittain,

ja edellisiin vaiheisiin palataan tarpeen mukaan. Tämän johdosta ontologiakehityksen työvaiheet voidaan kuvata spiraalina, jota havainnollistaa kuva 7. Näin ollen ontologiakehityksessä voidaan hyödyntää esimerkiksi ohjelmistotuotannon ketteriä menetelmiä (engl. agile software processes) [Mil01]. Ketteriä menetelmiä käytettäessä ohjelmistoa kehitetään lyhyissä jaksoissa rakentamalla siitä pieni osa kerrallaan. Jakson alussa sovitaan ominaisuudet, jotka jakson aikana toteutetaan. Ontologiakehityksen näkökulmasta yksi spiraalin kierros vastaa yhtä ketterien menetelmien jaksoa.



Kuva 7: Ontologiakehityksen työvaiheet.

Ontologioiden kehitystyökalut kehittyvät jatkuvasti, mutta ontologioiden kehittäminen käsin on aikaavievää sekä vaatii resursseja ja asiantuntemusta [AHa04, sivu 211]. Ontologioiden kehittämisessä voidaan käyttää puoliautomaattisia koneoppimisen (engl. machine learning) menetelmiä [AHa04, sivu 212]. Koneoppiminen on tekoälyn osa-alue, jossa tutkitaan olemassa olevaa tietämystä parantavia tai sen pohjalta kokonaan uutta tietämystä rakentavia

järjestelmiä [MKo83]. Täysin automaattiset menetelmät ovat virhealttiita [Jim08a]. Tämän johdosta hyödyntämällä soveltuvia työkaluja yhdistettynä soveltuvien alojen asiantuntijoiden näkemyksiin voidaan päästä laadukkaaseen lopputulokseen kohtuullisella vaivannäöllä.

Ontologioita kehittävät eri alojen asiantuntijat, joilla ei aina ole syvällistä tuntemusta kuvauslogiikoista tai ontologioiden sarjallistamismuodoista. Tämän johdosta asiantuntijoiden käyttöön tarvitaan helppokäyttöisiä ja laadukkaita kehitysympäristöjä [Hor11], jotka piilottavat käyttäjiltään mahdollisimman suuren osan monimutkaisista menetelmistä. Khondoker ja Mueller [KMu10] ovat tutkineet useiden saatavilla olevien ontologiamuokkainten käyttöä. Ontologiamuokkaimia ovat esimerkiksi OilEd [Bec01], WebODE [Arp01], Ontolingua [FFR97], OntoTrack [LNo03], Protégé [Pro02], SWOOP [KPH05] ja Top-Braid-Composer [Top13]. Ontologiamuokkainten välillä on eroja esimerkiksi havainnollistamiseen, hakutoimintoihin ja rajoitteiden tarkistamiseen liittyvissä ominaisuuksissa. Tarkoituksenmukaisimman työkalun valintaan vaikuttavat ensisijaisesti sen käyttäjät, heidän kokemuksensa ja tarpeensa. Tämän johdosta Khondoker ja Mueller perustavat tutkimuksensa käyttäjäkyselyyn. Kyselyyn vastanneista ylivoimaisesti suurin osa, 75% ontologioiden kehittäjistä käyttää eniten Protégé-muokkainta [Pro02]. Sitä käytetään laajasti eri sovellusalueilla ja käyttäjät ovat pääosin tyytyväisiä sen toimintaan.

Khondokerin ja Muellerin lisäksi Cardoso [Car07] on tutkinut ontologiakehitykseen tarkoitettujen työkalujen käyttöä. Cardoson käyttäjäkyselyn perusteella noin 70% ontologioita kehittävistä organisaatioista käyttää Protégé-muokkainta. Näiden tutkimusten perusteella Protégé on selvästi muiden vastaavien muokkainten edellä. Laadukkaiden muokkainten lisäksi ontologiakehitykseen tarvitaan työkaluja, jotka tukevat ainakin ontologioiden yhdistämistä, osakokonaisuuksiin jakamista, vertailua ja versionhallintaa [Hor11]. Tällaisia työkaluja ovat esimerkiksi ContentMap [Jim08a], ProSE

[Jim08b], OWLDiff [KSK11], Pellet 2 [Pel11] ja ContentCVS [Jim11]. Nämä työkalut voidaan liittää Protégé-muokkaimeen sen laajennusosina.

ContentMap [Jim08a] on ontologioiden puoliautomaattisen yhdistämisen mahdollistava työkalu. Se tutkii kahden ontologian yhdistämisessä syntyviä loogisia seurauksia, kun syötteenä annetaan tietynlaiset yhdistämismäärittelyt. Yhdistämismäärittelyillä ilmaistaan, minkä käsitteiden ja millaisten käsitteiden välisten suhteiden kautta ontologiat voidaan yhdistää. Kun tämän perusteella lasketut seuraukset on selvitetty, käyttäjä voi ilmaista, mitkä niistä ovat haluttuja ja mitkä eivät. Käyttäjän valintojen perusteella yhdistettyyn ontologiaan jätetään tietty määrä automaattisesti lisättyjä muutoksia.

ProSE-työkalu [Jim08b] mahdollistaa ontologian kehittämisen jaetuissa osakokonaisuuksissa. Sen käyttäjä voi valita ontologioista osia, jotka irrotetaan ontologioista kehitystyötä varten. Tämän lisäksi ProSE-työkalu helpottaa ontologioiden uudelleenkäyttöä. Sen avulla käyttäjä voi valita olemassa olevista ontologioista vain sovellusalueella tarvittavia osakokonaisuuksia.

OWLDiff-työkalu [KSK11] mahdollistaa ontologioiden keskinäisen vertailun ja yhdistämisen. Sen tarkoituksena on helpottaa ontologioiden ylläpitoa ja päivittämistä.

Pellet 2 [Pel11] on päättelykone, joka mahdollistaa loogisten johtopäätösten tekemisen kehitetyn ontologian pohjalta. Sen avulla kehitetyn ontologian pohjalta voidaan johtaa uutta tietämystä ja etsiä kehitetystä ontologiasta epäjohdonmukaisuuksia.

ContentCVS [Jim11] on versionhallintatyökalu, joka mahdollistaa ontologian kehittämisen rinnakkain useiden kehittäjien toimesta. ContentCVS hyödyntää

perinteisten syntaktisten menetelmien lisäksi semanttisia menetelmiä versioristiriitojen havaitsemiseen ja selvittämiseen.

Kun ontologia on kehitetty ja toimijoiden käytettävissä, sitä voidaan käyttää semanttiseen annotointiin [Kir04]. Perinnetiedon semanttisessa annotoinnissa olennainen ongelma on, miten annotaatioiden eli perinnetietoa ontologian käsitteisiin yhdistävien viittauksien luominen suoritetaan. Annotaatioiden tuottamista on pidetty yhtenä semanttisen webin pullonkaulana [HVT08]. Aihealueen asiantuntijan toimesta yksitellen luodut annotaatiot muodostavat laadukkaita kuvauksia, mutta niiden tuottaminen on hidasta. Tämän johdosta laajojen tietosisältöjen annotoinnissa on syytä harkita puoliautomaattisten ja automaattisten annotointityökalujen käyttöä [HVT08, Kir04]. Merkityksellisen tietämyksen kerääminen perinnejärjestelmistä täysin automaattisilla työkaluilla on monimutkaista [Vys10]. Tämä saattaa aiheuttaa virheitä tai puutteita annotoinnin lopputulokseen. Automaattisia ja puoliautomaattisia menetelmiä hyödyntämällä voidaan kuitenkin päästä kohtuulliseen tasapainoon annotoinnin tehokkuuden ja laadukkuuden välillä.

4.2 Relaatiomalli ja semanttisen webin tietomalli

Kuten mitä tahansa kohdetta, relaatiotietokantojen sisältämää tietoa voidaan kuvata määrittelemällä sitä kuvaavia ontologiapohjaisia semanttisia annotaatioita. Relaatiotietokantojen tapauksessa tällainen lähestymistapa on kuitenkin aikaavievä ja virhealtis [Ast04], joten relaatiomallin mukainen tieto kannattaa yleensä muuntaa semanttisen webin tietomallin mukaiseksi [Sel07]. Relaatiomallin vertaaminen RDF-tietomalliin on suoraviivaista [Ber98d]: relaatiotietokannan tietue vastaa RDF-lausetta, kentän nimi vastaa RDF-lauseen ominaisuutta ja kentän sisältö ominaisuuden arvoa. Muunnoksen tuloksena saatava RDF-muotoinen tieto voidaan tallentaa ennalta kehitetyn ontologian ilmentymiksi. Lopputuloksena on tietomalli, joka sisältää

relaatiotietokannan sisältämän tiedon ja sitä kuvaavan formaalin semantiikan sisältävän metatiedon. Tällaista muunnosta varten tarvittavan ontologian kehittämisessä voidaan hyödyntää alkuperäisen tiedon relaatiotietokantakaaviota [Ast04].

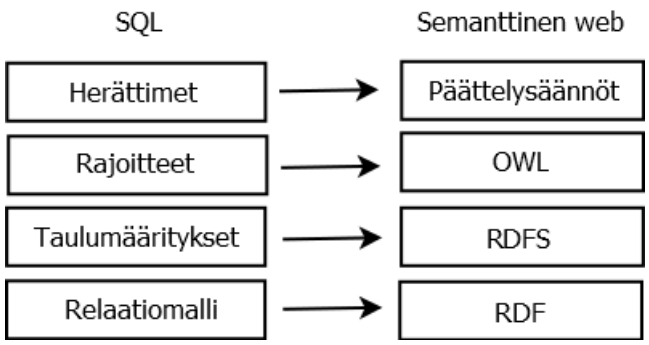
Monet relaatiotietokantoja tietovarastoinaan käyttävät perinnejärjestelmät on kuvattu Structured Analysis and Structured Design (SASD) -menetelmien avulla [Fri06]. SASD-menetelmiin kuuluu joukko mallinnusmenetelmiä, joilla tietojärjestelmiä voidaan kuvata. Sen yleisimmin käytettyihin osiin [Fri06] kuuluvat ER-malli [Che76] ja sen laajennukset [TYF86], jotka ovat pääasiassa tietokantojen mallinnukseen suunniteltuja tapoja kuvata tiedon rakenteita käsitekaavioina. ER-malli sisältää yksilöitä, yhteyksiä ja attribuutteja, joiden avulla käsitekaavioita muodostetaan. ER-malli ja ontologiat liittyvät toisiinsa, sillä molemmat kuvaavat käsitteitä ja niiden välisiä yhteyksiä. Tämän johdosta ER-mallin muuntaminen semanttisen webin ontologiaksi on suoraviivaista [Fah08]: ER-mallin yksilötyypit ja attribuutit vastaavat ontologian luokkia ja niiden sisäisiä ominaisuuksia, ja yksilötyyppien väliset yhteydet vastaavat luokkien välisiä ominaisuuksia.

ER-mallia ja sen laajennuksia vastaavia esityksiä tiedon yhteyksistä voidaan tuottaa nykyaikaisemmilla, olioympäristöön paremmin sopivilla Unified Modeling Language (UML) -menetelmillä [Fri06]. Relaatiotietokantaa kuvaavien UML- ja ER -mallien keskinäisen vastaavuuden johdosta ontologia voidaan kuvata yhtä hyvin UML- kuin ER -mallin pohjalta. Mikäli relaatiotietokannasta ei ole ER-kaaviota, se voidaan tuottaa takaisinmallinnuksella (engl. reverse engineering) [And94]. Takaisinmallinnuksessa perinnejärjestelmää tutkitaan tarkasti ja sen komponentit ja niiden väliset yhteydet kuvataan [CBS96]. Relaatiotietokantakaavioiden suoraviivainen takaisinmallinnus ER-mallin mukaiseksi esitykseksi ei säilytä kaikkea relaatiotietokantaan liittyvää semantiikkaa [Ast04]. Tämän johdosta relaatiotietokantakaavio kannattaa takaisinmallintaa ontologiaksi ER-kaavion sijaan. Relaatiotietokanta voidaan

esittää formaalisti predikaattilogiikan avulla, joten sen muuntaminen ontologiaksi on mahdollista [Zho10].

Relaatiotietokantojen hallintaan käytettävän SQL-kielen määrittely on jaettu osakieliin, joihin kuuluvat tiedonhaun mahdollistava kyselykieli, tietueiden muutokset, lisäykset ja poistot mahdollistava kannankäsittelykieli sekä tietokannan loogisen ja fyysisen tason rakenteiden määrittelyt mahdollistava kannanmäärityskieli [STM07]. SQL-kannanmäärityskielen ominaisuudet voidaan jakaa niiden ilmaisuvoiman mukaan semanttisen webin kerrosarkkitehtuuria vastaavalla tavalla [STM07].

Edellä verrattiin relaatiomallia RDF-tietomalliin ja ER-mallia ontologioihin. Tarkemmin ilmaistuna SQL-kannanmäärityskielen taulumäärittelyjen voidaan ajatella sisältävän RDFS-kieltä vastaavaa semantiikkaa [STM07]. SQL-kannanmäärityskielen avulla tietosisällöille voidaan määritellä relaatiomallin mukaisia relaatiokaavioita tauluina, joiden jokainen sarake sisältää tietotyyppimäärittelyn. RDFS-kieli mahdollistaa vastaavan määrittelyn RDF-tietomallille. Avain-, viite-eheys- ja arvojoukkorajoitteiden lisäämisen taulumäärittelyihin voidaan ajatella vastaavan OWL-ontologiakieliä semantiikaltaan [STM07]. Kaikki mahdolliset SQL-kannanmäärityskielen rajoitteiden ilmaisut voidaan esittää täsmälleen vastaavasti OWL-kielillä. SQL-kannankäsittelykielen avulla tehtyjen muutosten yhteydessä tietosisällön eheyden takaavien liiketoimintasääntöjen esittämisen mahdollistavien herättimien voidaan ajatella vastaavan semanttisen webin päättelysääntöjä [STM07]. Koska SQL-kannanmäärityskielellä voidaan kuvata aihealueen merkityksiä kattavasti, perinnetietoa sisältävät relaatiotietokannat voivat sisältää riittävästi metatietoa monipuolisten semanttisten annotaatioiden mahdollistamiseksi. Tällainen metatieto ei kuitenkaan ole koneellisesti saatavilla [STM07], joten se on esitettävä eri tavalla, kuten semanttisen webin menetelmien avulla. Kuva 8 havainnollistaa SQL-kannanmäärityskielen ominaisuuksien ja semanttisen webin menetelmien vastaavuutta.



Kuva 8: SQL-kannanmäärityskielen ominaisuuksien ja semanttinen webin menetelmien vastaavuudet [STM07].

Relaatiotietokantojen sisältämän tiedon muuntamista semanttisen webin tietomallin mukaiseksi voidaan lähestyä kolmella tavalla [PVi12]. *Ensimmäisessä lähestymistavassa* relaatiotietokantakaavio muunnetaan ontologiaksi ja relaatiotietokannan sisältö muunnetun ontologian ilmentymiksi. *Toisessa lähestymistavassa* relaatiotietokannan sisältö muunnetaan ontologiaksi. *Kolmannessa lähestymistavassa* relaatiotietokannan sisältö muunnetaan jonkin olemassa olevan ontologian ilmentymiksi. Eri lähestymistavoissa käytettävät muunnosmenetelmät voidaan jakaa arkkitehtonisesti kahteen luokkaan: *suoraan muuntamiseen* ja *muunnoskieliin perustuvaan muuntamiseen* [PVi12, Sel07, Seq11]. Suora muuntaminen (engl. direct mapping) on menetelmä, jolla voidaan muuntaa relaatiotietokantojen sisältöjä RDF-tietomallin mukaiseen muotoon täysin automaattisesti. Muunnoskieliin perustuvassa muuntamisessa käytetään jollakin muunnoskielellä (engl. mapping language) ulkoisesti kuvattua määrittelyä, jonka avulla relaatiotietokantakaavioiden ja aihealuetta kuvaavien ontologioiden välille voidaan kuvata yhteyksiä. Tällöin muunnoksen lopputuloksena saatava RDF-tietomalli luodaan muunnoskielellä määriteltyjen yhteyksien pohjalta.

Sahoon ja kumppaneiden [Sah09] tekemän tutkimuksen perusteella suurin osa olemassa olevista menetelmistä ja työkaluista perustuu ensimmäiseen tai kolmanteen lähestymistapaan. Tämä johtuu siitä, että suorassa muuntamisessa käytetään tavallisesti ensimmäistä ja muunnoskieliin perustuvassa

muuntamisessa kolmatta lähestymistapaa. Tietyissä tapauksissa on kuitenkin tarkoituksenmukaisinta käyttää toista lähestymistapaa. Tällaisia tapauksia ovat esimerkiksi relaatiotietokantaan tallennetut asiasanastot, taksonomiat ja tesaurukset, jotka halutaan muuntaa ontologioiksi [PVi12]. Toisen lähestymistavan toteuttaminen pelkästään olemassa olevien suoraa muuntamista tai muunnoskieliin perustuvaa muuntamista tukevien työkalujen avulla ei ole toistaiseksi mahdollista.

Priyatna ja Villazón-Terrazas [PVi12] ehdottavat, että relaatiotietokannan tietosisältö voidaan muuntaa ontologiaksi yhdistämällä ontologioiden luomiseen tarkoitettuja uudelleensuunnittelumalleja [Vil12] ja muunnoskielellä kuvattuja muunnosmäärittelyjä. Ohjelmiston uudelleensuunnittelumallit (engl. reengineering patterns) kuvaavat, miten perinnejärjestelmä voidaan uudistaa nykyisiin vaatimuksiin ja ympäristöön nähden sopivaksi [PSt98]. Uudelleensuunnittelumallien pääasiallinen tarkoitus on tarjota ratkaisuja tunnettuihin uudelleensuunnittelun ongelmiin. Edellä kuvatut muunnosmenetelmät voidaan nähdä eräänlaisina ohjelmiston uudelleensuunnittelumalleina, koska niiden avulla voidaan muuntaa relaatiotietokantojen sisältämää perinnetietoa uuteen tietomalliin [SPV12].

4.3 Suora muuntaminen

Suoran muuntamisen ensimmäisessä vaiheessa luodaan *oletettu* (engl. putative) ontologia [Seq11]. Oletettu ontologia luodaan muunnostyökalulle syötteenä annettavan relaatiotietokantakaavion pohjalta automaattisella takaisinmallinnuksella. Takaisinmallinnus perustuu relaatiotietokantakaavion rakenteiden tunnistamiseen [DSC12]. Suoran muuntamisen toisessa vaiheessa relaatiotietokannan tietosisältö muunnetaan RDF-tietomallin mukaiseen muotoon oletetun ontologian kuvaamien käsitteiden ilmentymiksi [Seq11]. Suoran muuntamisen lopputuloksena saadaan semanttisesti yhteenlinkitetty ja

oletetun ontologian käsitteistön avulla annotoitu tietomalli. Lopputuloksen laatu riippuu osittain siitä, miten hyvin syötteenä saatava relaatiotietokantakaavio esittää tietoon liittyviä merkityksiä.

Suora muuntaminen on muunnoksen toteuttajan näkökulmasta yksinkertaista ja tehokasta, koska se tapahtuu täysin automaattisesti [Seq11]. Suoran muuntamisen huono puoli on, että relaatiotietokannan alkuperäiseen mallinnukseen ei voida vaikuttaa [DSC12]. Suoran muuntamisen tuloksena saatu RDF-tietomalli vastaa aina täsmälleen alkuperäisen relaatiotietokannan rakennetta. Oletettu ontologia koostuu täsmälleen relaatiotietokantakaaviossa käytettyjen rakenteiden nimistä. Oletetun ontologian sisältämän semantiikan kattavuus riippuu siitä, miten hyvin relaatiotietokanta on mallinnettu ja miten sitä on voitu muuntamisen yhteydessä koneellisin menetelmin tulkita.

Hyvien ohjelmistotuotannon käytäntöjen mukaisesti relaatiotietokanta esitetään loogisen tason tietokantakaaviona, joka muunnetaan todelliseksi tietokantakaavioksi. Tällaisessa tapauksessa sopivassa normaalimuodossa oleva relaatiotietokantakaavio sisältää todennäköisesti tarpeeksi semantiikkaa monipuolisen ja alkuperäistä loogisen tason kaaviota ilmaisuvoimaltaan vastaavan ontologian luomiseksi [Seq11]. On kuitenkin mahdollista, että hyviä käytäntöjä ei ole noudatettu. Kaikki käytössä olevat relaatiotietokannat eivät noudata yleisesti hyväksyttyjä käytäntöjä ja suunnitteluperiaatteita [BCy07], joten niiden laatu vaihtelee merkittävästi eri järjestelmien välillä [Seq11]. Tällainen tilanne on sitä todennäköisempi, mitä vanhempi järjestelmä on kyseessä. Toisaalta on mahdollista, että vaatimusten muuttuessa todellista tietokantakaaviota on muutettu suoraan tekemättä muutoksia loogisen tason kaavioon. Näissä tapauksissa todellinen tietokantakaavio on luotettavin ja ajantasaisin lähde kuvaamaan tietokannan rakennetta. Tämän johdosta suorassa muunnoksessa käytetty takaisinmallinnus kannattaa perustaa todelliseen relaatiotietokantakaavioon [Seq11].

Sequedan ja kumppaneiden [Seq11] tutkimusten perusteella edes hyvin merkityksiä kuvaavista relaatiotietokannoista luotujen oletettujen ontologioiden ei voida taata vastaavan relaatiotietokantakaaviota semantiikaltaan. Tämä johtuu siitä, että monipuolisten merkitysten koneellinen tunnistaminen relaatiotietokantakaaviosta on monimutkaista. Esimerkiksi periytyvyys voidaan ilmaista relaatiotietokantakaaviossa eri tavoin, jolloin sen tunnistaminen yksittäisellä säännöllä ei ole mahdollista. Oletettua ontologiaa laajentamalla voidaan parantaa sen laatua ja yhteentoimivuutta. Luvussa 4.1 kuvatuilla ontologiakehityksen menetelmillä oletettuun ontologiaan voidaan lisätä uusia käsitteitä ja yhteyksiä tai yhdistää oletettu ontologia toiseen aihealuetta kuvaavaan ontologiaan.

Suoraa muuntamista käytettäessä relaatiotietokannan oletetaan yleensä olevan vähintään kolmannessa normaalimuodossa [Seq11]. Muunnoksen tekeminen muissa normaalimuodoissa oleville relaatiotietokannoille on kuitenkin mahdollista. Relaatiotietokanta voidaan normalisoida eri tavoin. Vastaavasti ontologia voidaan mallintaa eri tavoin. Oikeaa tai väärää tapaa ei välttämättä ole, vaan sopivin mallinnustapa riippuu sovellusalueesta ja aihealueen asiantuntijoiden mielipiteistä. Relaatiotietokannan normalisointia on esimerkiksi voitu vähentää tietynlaisten kyselyiden suorituskyvyn parantamiseksi. Muunnettavan relaatiotietokannan normaalimuoto ratkaisee, minkälainen oletetusta ontologiasta tulee.

Suoran muuntamisen lopputuloksen laatu riippuu siitä, miten hyvin relaatiotietokantakaavio ilmaisee esittämänsä aihealueen merkityksiä [Seq11]. Relaatiotietokantoja on jo kauan määritelty laadukkailla mallinnustyökaluilla, jotka tukevat esimerkiksi viiteavaimien ja eheysrajoitteiden tarkistusta. Tällaisilla työkaluilla ja menetelmillä hyvien käytäntöjen mukaisesti kehitetty ja normalisoitu relaatiotietokantakaavio sisältää riittävästi tietoa laadukkaan ontologian luomiseksi [Seq11]. Hyvin suunnitellun relaatiotietokannan tapauksessa täysin automaattisesti tehtävä suora muuntaminen on helpoin tapa

muuntaa relaatiotietokannan sisältö RDF-tietomallin mukaiseen muotoon. Suora muuntaminen voi toimia hyvänä lähtökohtana monipuolisemman kuvauksen toteuttamista varten, vaikka se ei aina johda aihealueen monipuolisen semantiikan kuvaamiseen [Sah09]. Kuvausta voidaan monipuolistaa laajentamalla oletettua ontologiaa. Suora muuntaminen soveltuu muuntamismenetelmäksi, kun relaatiotietokantakaavion suora esittäminen osana ontologiapohjaista annotointia on hyväksyttävää ja muuntamisen yksinkertaisempaa toteutusta pidetään laadukkaampaa lopputulosta tärkeämpänä [HRG11].

Suoran muuntamisen toteuttamista on tutkittu viimeisen vuosikymmenen aikana kattavasti [Sah09, Seq11]. Kehitystyö on johtanut siihen, että W3C-yhteenliittymä on julkaissut syyskuussa 2012 Direct Mapping -normin suoran muuntamisen toteuttamiseksi [Are12]. Direct Mapping -normi määrittelee menetelmän relaatiotietokannan tietosisältöä vastaavan RDF-tietomallin eli RDF-verkon luomiseksi. Tuloksena saatavaa RDF-verkkoa kutsutaan *suoraksi verkoksi*. Jatkossa suoran muuntamisen toteuttavien työkalujen on tarkoitus noudattaa Direct Mapping -normia. Tällä hetkellä tunnettuja Direct Mapping -normia noudattavia järjestelmiä ovat D2RQ- RDF-RDB2RDF-, SWObjects dm-materialize-, XSPARQL-, Ultrawrap-, db2triples- ja r2rml4net -järjestelmä [VHa12].

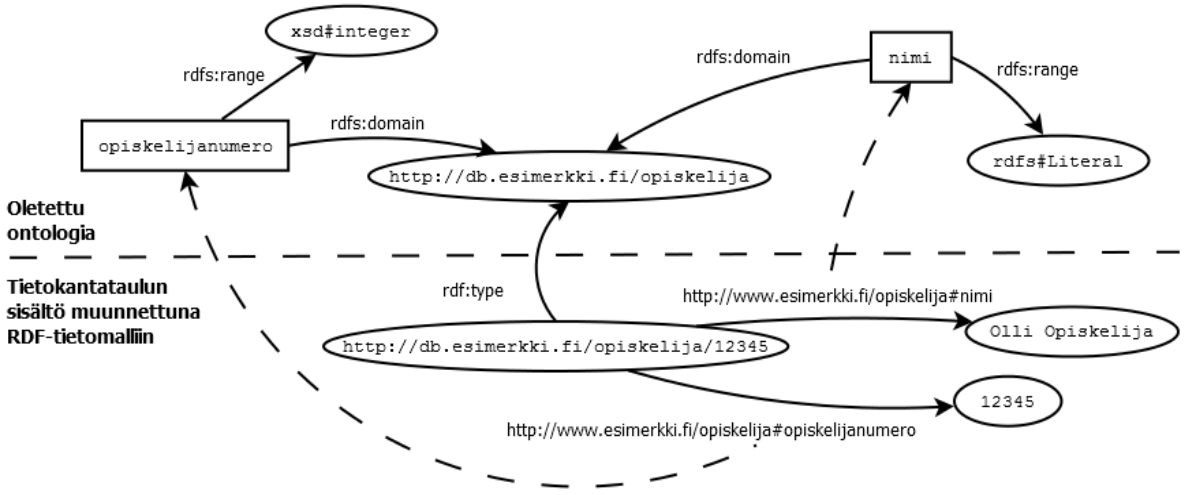
Tarkastellaan kuvitteellista relaatiotietokantaa, joka sisältää *opiskelija*-taulun. Opiskelija-taulun sarakkeita ovat *opiskelijanumero* ja *nimi*. Direct Mapping -suosituksen mukaan luodaan oletettu ontologia. Siinä opiskelija-taululle luodaan vastineeksi opiskelija-käsite, jonka ominaisuuksiksi asetetaan opiskelijanumero ja nimi. Tämän jälkeen taulun rivit muunnetaan ontologian rakenteisiin liitetyiksi RDF-lauseiksi. Jokaisesta rivistä luodaan opiskelija-käsitteen ilmentymä, jolle asetetaan kahdella RDF-lauseella opiskelijanumero ja nimi. Kuva 9 havainnollistaa suoran muunnoksen tuloksena saatavaa suoraa verkkoa.

Tarkastellaan kuvitteellisen relaatiotietokannan *opiskelija*-taulua:

```
CREATE TABLE opiskelija (
      opiskelijanumero int PRIMARY KEY,
      nimi VARCHAR(50)
)
INSERT INTO opiskelija (opiskelijanumero, nimi) VALUES (12345, 'Olli Opiskelija');
```

opiskelijanumero	nimi
12345	Olli Opiskelija

Suoralla muuntamisella opiskelija-taulusta voidaan saada esimerkiksi seuraavanlaiset RDF-kolmikkolauseet:



Etuliite rdf, nimiavaruustunniste http://www.w3.org/1999/02/22-rdf-syntax-ns#

Etuliite rdfs, nimiavaruustunniste http://www.w3.org/2000/01/rdf-schema#

Etuliite xsd, nimiavaruustunniste http://www.w3.org/2001/XMLSchema#

Kuva 9: Kuvitteellisen relaatiotietokannan opiskelija-taulun suora muuntaminen RDF-muotoon.

Suora muuntaminen ei aina ole tarkoituksenmukaisin ratkaisu. Muunnoksen yhteydessä voi yhtäältä olla tarve muuttaa tiedon rakenteita tai annotoida tietoa jonkin olemassa olevan ontologian avulla. Toisaalta voi olla tarve määritellä, että relaatiotietokannan sisältämää tietoa ei muunneta ontologian ilmentymiksi, vaan ontologian käsitteiksi. Tällaisissa tapauksissa voidaan käyttää muunnoskieliin perustuvaa muuntamista.

4.4 Muunnoskieliin perustuva muuntaminen

Muunnoskieliin perustuvassa muuntamisessa muunnoksen tekijä voi määritellä ulkoisten muunnoskielten avulla yhteyksiä relaatiotietokantakaavioiden ja ontologioiden välille [BCy07, DSC12, Sel07]. Näin voidaan vaikuttaa joustavasti muunnoksen lopputulokseen. Muunnoskielillä määriteltyjen muunnoskuvausten avulla relaatiotietokantojen sisältämästä tiedosta voidaan muodostaa RDF-tietomallin mukaisia esityksiä, jotka on kuvattu yhden tai useamman ontologian käsitteiden avulla. Muunnoskuvaukset ovat usein jollakin ihmiselle helppolukuisella sarjallistamiskielellä esitettyjä RDF-kuvauksia [BCy07, DSC12].

Kaikki käytössä olevat relaatiotietokannat eivät noudata yleisesti hyväksyttyjä tietokannan suunnitteluperiaatteita [BCy07], joten niiden laatu vaihtelee merkittävästi eri järjestelmien välillä [Seq11]. Tällaisten relaatiotietokantojen huonojen ominaisuuksien siirtyminen muunnoksen tuloksena saatavaan RDF-tietomalliin voidaan välttää riittävän monipuolisen ja ilmaisuvoimaisen muunnoskielen avulla. Tällaisen muunnoskielen on voitava käsitellä sekä korkealla tasolla normalisoituja että sellaisia tietokantarakenteita, jotka eivät ole edes ensimmäisessä normaalimuodossa. Tämän lisäksi monipuolisen muunnoskielen avulla on voitava korjata huonosti määriteltyjä tietokantarakenteita sekä tukea tiedon ehdollisia yhdistelyjä ja erilaisia arvojen muunnoksia.

Muunnoskielen ilmaisuvoiman kasvattaminen lisää sen monimutkaisuutta [HRG11]. Monimutkaisuuden lisääntyessä muunnoskielen käyttö ja sitä tukevien työkalujen toteuttaminen vaatii syvällisempää perehtymistä. Tämän lisäksi ilmaisuvoimaisempien muunnoskuvausten toteuttaminen kaksisuuntaisesti on vaikeampaa. Kaksisuuntaisesti toimivat muunnoskuvaukset mahdollistavat tiedon kaksisuuntaisen päivittämisen. Tällöin tietoa voidaan päivittää joko alkuperäiseen tai muunnoksen tuloksena

saatuun tietomalliin, ja muutokset toteutuvat molempiin tietomalleihin. Hyvin ilmaisuvoimaisten muunnoskielten tapauksessa kaksisuuntaisesti toimivien muunnoskuvausten toteuttamisesta tulee epäkäytännöllistä. Tämän johdosta hyvin ilmaisuvoimaiset muunnoskielet ovat vain lukemisen mahdollistavia. Niillä määriteltyjen yhteyksien avulla RDF-tietomallin mukaiseksi muunnettua relaatiotietokannan sisältämää tietoa voidaan hakea, muttei päivittää niin, että päivitykset näkyvät myös relaatiotietokannassa.

Hert ja kumppanit [HRG11] ovat vertailleet tunnettuja muunnoskieliä. Muunnoskielet voidaan jakaa kolmeen joukkoon: *yleisiin lukemisen mahdollistaviin (engl. read-only) muunnoskieliin*, *yleisiin lukemisen ja kirjoittamisen mahdollistaviin (engl. read/write) muunnoskieliin* ja *erityisiin käyttötarkoituksiin laadittuihin muunnoskieliin*. Muunnoskielet eroavat toisistaan ominaisuuksiltaan ja ilmaisuvoimaltaan.

Yleisillä lukemisen mahdollistavilla muunnoskielillä määriteltyjen muunnoskuvausten avulla relaatiotietokannasta RDF-tietomallin mukaiseksi muunnettua tietoa voidaan hakea, muttei päivittää niin, että päivitykset näkyvät myös relaatiotietokannassa. Yleiset lukemisen mahdollistavat muunnoskielet mahdollistavat hyvin ilmaisuvoimaisten muunnoskuvausten määrittelyn. Hyvän ilmaisuvoimansa ansiosta ne soveltuvat monipuolisesti erilaisille sovellusalueille, joilla RDF-tietomalliin muunnettua tietoa ei tarvitse päivittää. Yleisiä lukemisen mahdollistavia muunnoskieliä ovat Virtuoso- [EMi07], D2RQ- [BCy07, BSe04] ja R2RML -kieli [DSC12]. Niiden ominaisuuksissa on pieniä eroja, mutta ne vastaavat ilmaisuvoimaltaan toisiaan. Erot ominaisuuksissa ovat niin pieniä, että tarkoituksenmukaisimman muunnoskielen valintaan voivat ominaisuuksia enemmän vaikuttaa esimerkiksi normiyhteensopivuus, käyttöehdot ja määrittelyn kypsyys [HRG11]. Esimerkiksi muunnoskuvausten uudelleenkäytön mahdollistamiseksi niiden tekeminen yhteisten normien mukaisesti on olennaisen tärkeää [Sah09]. Tällöin varteenotettavimmaksi vaihtoehdoksi nousee R2RML, joka on syyskuussa 2012

saavuttanut W3C-yhteenliittymän suosituksen aseman. Se on vapaasti käytettävissä, valmistajariippumaton sekä todennäköisesti tulevaisuudessa laajimman käyttöönoton ja työkalutuen saavuttava muunnoskieli. Tämän lisäksi se tukee kaikkia Virtuoso- ja D2RQ -muunnoskielten ominaisuuksia [HRG11]. R2RML-kielen muunnoskuvaukset ilmaistaan ihmiselle helppolukuisella Turtle-kielellä [BBe11] sarjallistettuina RDF-verkkoina.

Yleiset lukemisen ja kirjoittamisen mahdollistavat muunnoskielet mahdollistavat sekä tiedon hakemisen että päivittämisen kaksisuuntaisesti [HRG11]. Tällöin RDF-tietomalliin tehdyt päivitykset voidaan siirtää relaatiotietokantaan muunnoskielellä määriteltyjen muunnoskuvausten perusteella. Ilman tukea kaksisuuntaisille päivityksille yleiset lukemisen ja kirjoittamisen mahdollistavat muunnoskielet voitaisiin luokitella yleisiksi lukemisen mahdollistaviksi muunnoskieliksi. Suurin ero näiden muunnoskielien välillä on, että näkymien päivitysongelmasta (engl. view update problem) [BSp81] johtuen kaksisuuntaisten päivitysten toteutuksessa ei voida tukea mielivaltaisia muunnosmäärittelyjä. Esimerkiksi *suurempi kuin* tai *pienempi kuin* - operaattoreiden käyttö kyselyn ehtomäärittelyissä voi aiheuttaa ongelmia, kun niiden perusteella yritetään tunnistaa päivitettävä tietue. Päivitysten mahdollistamiseksi muunnoskielen ilmaisuvoimaa on rajoitettava siten, että päivitettävää tietuetta ei voida esittää moniselitteisesti [HRG11]. Tämän johdosta yleiset lukemisen ja kirjoittamisen mahdollistavat muunnoskielet ovat ilmaisuvoimaltaan heikompia, kuin yleiset lukemisen mahdollistavat muunnoskielet. Kaikki muunnoskielet tarjoavat kuitenkin suoraa muuntamista enemmän ilmaisuvoimaa. Suoraan muuntamiseen verrattuna muunnoskieliin perustuvan muuntamisen huono puoli on työläämpi toteutus, sillä muunnoskielten opettelu ja käyttö vie enemmän aikaa [Sel07]. R3M [HRG10] on yleinen lukemisen ja kirjoittamisen mahdollistava muunnoskieli. Sitä kannattaa käyttää, kun sovellusalueen vaatimuksiin sisältyy tiedon kaksisuuntainen päivitysmahdollisuus.

Erityisiin käyttötarkoituksiin laaditut muunnoskielet on suunniteltu sisältämään ominaisuuksia, jota tarvitaan erityisissä käyttötapauksissa [HRG11]. Tämä ei välttämättä tarkoita, että muunnoskielten ilmaisuvoimaa on jouduttu heikentämään. Niiden käyttö on suositeltavaa, kun sovelluksen käyttötarkoitus vastaa läheisesti muunnoskielen suunnittelussa ajateltua käyttötarkoitusta, ja yleiset lukemisen ja kirjoittamisen mahdollistavat muunnoskielet eivät sovellu käyttöön tai ovat turhan monimutkaisia [HRG11]. Erityiseen tarkoitukseen laadittuja muunnoskieliä ovat eD2R- [BCG03], R2O- [BCG04], Triplify- [Aue09] ja Relational.OWL -muunnoskieli [LCo05]. D2RQ-muunnoskielen edeltäjän D2R MAP -muunnoskielen [Biz03] laajennus eD2R on tarkoitettu käytettäväksi sovellusalueilla, joissa muunnettava relaatiotietokanta on kevyesti rakenteinen tai ei täytä ensimmäisen normaalimuodon ehtoja. R2O-muunnoskielen käyttö soveltuu tilanteisiin, joissa relaatiotietomallin ja käytettävän ontologian välillä on vain vähän yhtäläisyyksiä. Triplify on suunniteltu käytettäväksi sovellusalueilla, joissa relaatiotietokannan sisältämä tieto halutaan esittää WWW-palvelussa yhdistetyn tiedon (engl. linked data) periaatteiden [Ber06] mukaisesti, JavaScript Object Notation (JSON)- [Cro06] tai Comma-Separated Values (CSV) -muodossa [Sha05]. Relational.OWL on poikkeuksellisen erikoistettu muunnoskieli. Sen käyttötarkoitus liittyy tilanteisiin, joissa tarvitaan vertaisverkossa olevien tietokantojen välistä tiedonsiirtoa.

Edellä kuvattujen erojen lisäksi yksittäisten muunnoskielten käytettävyydessä on eroja. D2RQ- [BSe04], R2RML- [DSC12], Triplify- [Aue09] ja Virtuoso [EMi07] -muunnoskielet hyödyntävät SQL-kieltä muunnoksen toteutuksessa. Tästä on sekä hyötyä että haittaa [HRG11]. SQL-kieli on erittäin ilmaisuvoimainen ja hyvin tunnettu relaatiotietokantojen kehittäjien ja ylläpitäjien keskuudessa. Tämä helpottaa muunnoskuvausten tuottamista ja mahdollistaa muuntamisprosessin siirtämisen pääosin relaatiotietokannan hallintajärjestelmän tehtäväksi. SQL-kieltä hyödyntävien muunnoskuvausten merkitykset eivät kuitenkaan ole helposti saatavilla, sillä ne sisältyvät SQL-lauseisiin. Tällöin merkitysten esillesaanti vaatii ylimääräistä SQL-

merkkijonojen jäsentelyä. Lisäksi SQL-näkymiin perustuvat muunnosmäärittelyt kärsivät samasta näkymien päivitysongelmasta [BSp81] kuin tavalliset SQL-näkymät. Muunnokseen tai näkymään sisällytettävä päivitysmahdollisuus on epäkäytännöllistä. Sen mahdollistamiseksi käytettävä kuvauskieli on rajoitettava alkuperäisen kuvauskielen osakieleksi, jolloin kuvauskielen ilmaisuvoima heikkenee ja tietomallien välinen yhteentoimivuus kärsii.

Tämän tutkielman näkökulmasta kaksisuuntaisen päivitysmahdollisuuden saavuttaminen ei ole välttämätöntä. Perinnetiedon semanttisella annotoinnilla on ensisijaisesti tarkoitus parantaa tiedon löydettävyyttä. Tämän saavuttamiseksi riittää, että tieto muunnetaan RDF-tietomallin mukaiseen muotoon lukemisen mahdollistavalla tavalla. Relaatiotietokantaan varastoitua perinnetietoa voidaan edelleen ylläpitää ja käyttää perinnejärjestelmien avulla. Näin ollen sopivimman muunnoskielen valinta voidaan rajata yleisiin lukemisen mahdollistaviin ja erityisiin käyttötarkoituksiin laadittuihin muunnoskieliin. Koska yksisuuntaiset muunnoskuvaukset riittävät, SQL-kieltä hyödyntävien muunnoskielten opittavuus ja helppokäyttöisyys puoltavat niiden valintaa. Tämän ja aiemmin tässä luvussa muunnoskielistä kuvatun perusteella sopivimman muunnoskielen valinta kannattaa kohdistaa joko soveltuvaan erityiseen käyttötarkoitukseen laadittuun muunnoskieleen tai R2RML-kieleen. Mikään edellä kuvatuista erityiseen käyttötarkoitukseen laadittujen muunnoskielten käyttötarkoituksista ei vastaa suoraan tämän tutkielman aihepiiriä. Tämän johdosta tarkoituksenmukaisimman muunnoskielen valinta on helppo kohdistaa W3C-yhteenliittymän suosittelemaan R2RML-kieleen.

W3C-yhteenliittymän suosittelemaa R2RML-muunnoskieltä tukevia muunnostyökaluja ovat Virtuoso-, RDF-RDB2RDF-, XSPARQL-, Morph-, Ultrawrap-, db2triples- ja r2rml4net -järjestelmä [VHa12]. Tämän lisäksi D2RQ-järjestelmään ollaan lisäämässä R2RML-tuki [Hau12].

R2RML-muunnoskielellä tehdyt muunnoskuvaukset perustuvat relaatiotietokannan loogisiin tauluihin [DSC12, PVi12]. Loogisia tauluja voivat olla relaatiotietokannan taulut, näkymät tai SQL-kyselyt. R2RML-kielellä loogiset taulut yhdistetään RDF-tietomalliin. Tämä tapahtuu kolmikkolausemäärityksen (engl. triples map) avulla. Kolmikkolausemääritys kuvaa sääntöjä, joita noudattaen loogisten taulujen rivit muunnetaan RDF-tietomallin mukaisiksi RDF-lauseiksi. Säännöt koostuvat pääosin subjektimäärityksestä ja useista predikaatti-objektimäärityksistä. Subjektimäärityksen perusteella luodaan kaikkien yhdestä loogisen taulun rivistä luotavien RDF-lauseiden subjekti. Predikaatti-objektimääritykset koostuvat useista erillisistä predikaatti- ja objektimäärityksistä. Sääntö yksittäisen RDF-lauseen luonnille saadaan yhdistämällä subjektimääritys predikaatti- ja objektimääritykseen jokaisen loogisen taulun rivin kohdalla.

Tarkastellaan uudelleen kuvan 9 havainnollistamaa suoran muuntamisen esimerkkiä. R2RML-muunnoskielellä voidaan määritellä esimerkiksi muunnoskuvaus, jonka perusteella opiskelija-taulun annotoinnissa käytetään kuvitteellista tietojenkäsittelytieteen laitoksen ylläpitämää opiskelija-ontologiaa sekä henkilöitä ja henkilösuhteita kuvaavaa Friend Of A Friend (FOAF) -ontologiaa [FOA00]. Tällöin määritellään opiskelija-taulua koskeva kolmikkolausemääritys, joka koostuu kahdesta predikaatti-objektimäärityksestä. Ensimmäisessä määritellään, että opiskelija-taulun opiskelijanumero-sarakkeen arvo yhdistetään opiskelija-ontologian opiskelijanumero-ominaisuuteen. Toisessa määritellään, että opiskelija-taulun nimi-sarakkeen arvo yhdistetään FOAF-ontologian name-ominaisuuteen. Kolmikkolausemäärityksen subjektimäärityksen perusteella kaikista opiskelija-taulun riveistä luotavien RDF-lauseiden subjektiksi määritetään IRI-tunniste, jolla opiskelijoihin viitataan. Tunnisteessa opiskelijanumero muuttuu rivikohtaisesti. Kaikkien opiskelijoiden määritellään olevan opiskelija-ontologian opiskelija-käsitteen ilmentymiä. Tätä esimerkkiä havainnollistaa kuva 10.

Suoran muuntamisen ja muunnoskieliin perustuvan muuntamisen avulla perinnetiedon semanttinen annotointi voidaan hoitaa automaattisesti tai puoliautomaattisesti. Monipuolisten ontologioiden ja mahdollisten muunnoskuvausten tuottamiseen tarvitaan ihmisiä, mutta ontologioihin perustuvat annotaatiot sisältävä tietomallin muunnos voidaan toteuttaa automaattisten työkalujen avulla. Perinnetiedon semanttinen annotointi voidaan toteuttaa täysin automaattisesti, jos käytetään suoraa muuntamista ja tyydytään sen tuloksena saatuun oletettuun ontologiaan. Täysin automaattinen annotointi toteutetaan tällöin lopputuloksen laadun kustannuksella. Tarkoituksenmukaisimman menetelmän valinnassa on etsittävä sopiva tasapaino annotoinnin kustannustehokkuuden ja lopputuloksen laadun välille. Tämän lisäksi menetelmän valintaan vaikuttavat sovellusalue, miten hyvin relaatiotietokannan rakenne ja sisältö tunnetaan, tarvitaanko kaksisuuntaista muunnosta ja onko aiemmin käytetty ratkaisua, joka rajoittaa menetelmän valintaa. Tarkoituksenmukaisin menetelmä voidaan löytää vertaamalla tässä luvussa kuvattujen menetelmien ominaisuuksia sovellusalueen tarpeisiin. Kuva 11 havainnollistaa, mitkä kysymykset vaikuttavat tarkoituksenmukaisimman annotointimenetelmän valintaan.

4.5 Relaatiotietokannan peittäminen

Perinnejärjestelmien muuttaminen tai uudelleentoteutus on usein erittäin työlästä [GOu05]. Perinnejärjestelmistä luopuminenkaan ei aina ole mahdollinen vaihtoehto, sillä niiden merkitys voi olla erittäin tärkeä [Ben95]. Tämän johdosta

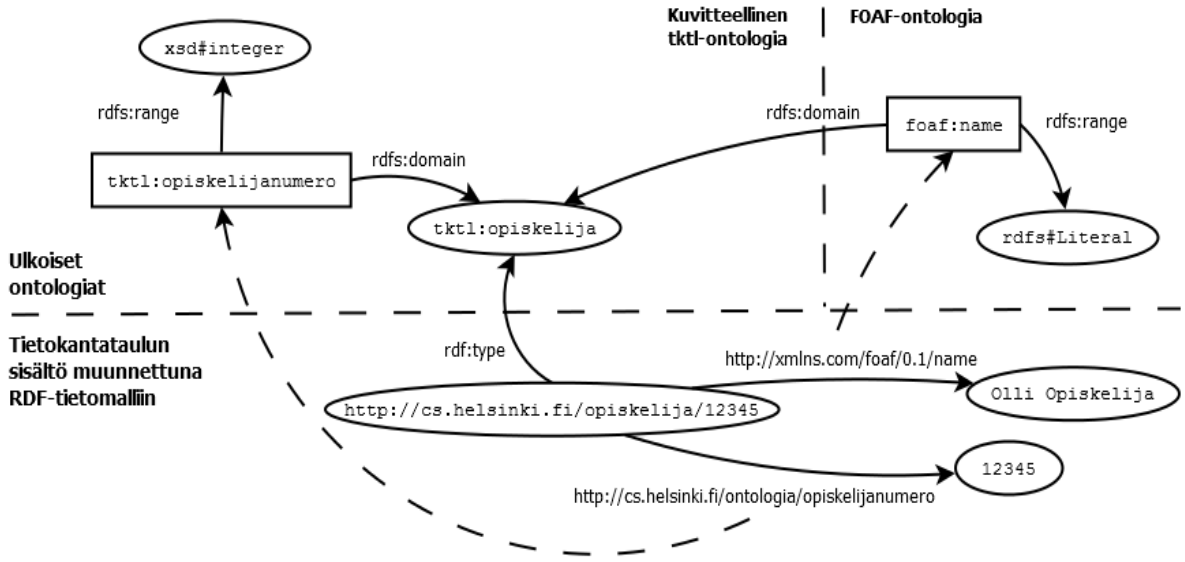
Tarkastellaan uudelleen kuvitteellisen relaatiotietokannan *opiskelija*-taulua:

opiskelijanumero	nimi
12345	Olli Opiskelija

R2RML-muunnoskielellä voidaan määritellä esimerkiksi seuraava muunnoskuvaus, jonka avulla voidaan määritellä ulkoiset ontologiat, joita muunnoksessa käytetään:

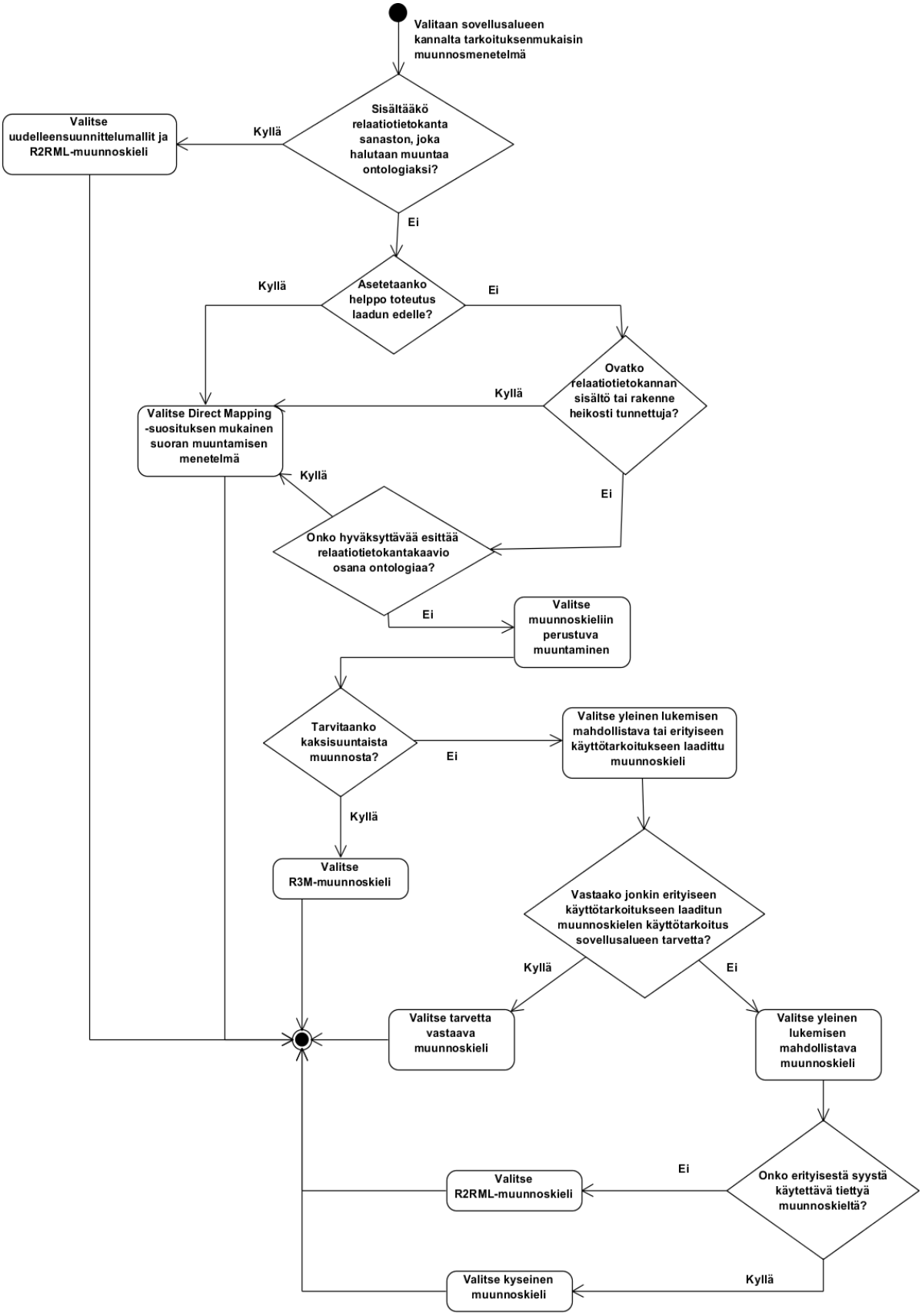
```
@prefix foaf: <http://xmlns.com/foaf/0.1/>
@prefix r2rml: <http://www.w3.org/ns/r2rml#>
@prefix tktl: <http://cs.helsinki.fi/ontologia/>

<#TriplesMap1>
    r2rml:logicalTable [ r2rml:tableName "opiskelija" ];
    r2rml:subjectMap [
        r2rml:template "http://cs.helsinki.fi/opiskelija/{opiskelijanumero}";
        r2rml:class tktl:opiskelija;
    ];
    r2rml:predicateObjectMap [
        r2rml:predicateMap [ r2rml:constant tktl:opiskelijanumero ];
        r2rml:objectMap [ r2rml:column "opiskelijanumero" ]
    ];
    r2rml:predicateObjectMap [
        r2rml:predicateMap [ r2rml:constant foaf:name ];
        r2rml:objectMap [ r2rml:column "nimi" ]
    ].
```

Muunnoskuvauksen perusteella saadaan luotua seuraavanlaiset RDF-kolmikkolauseet:



Etuliite foaf, nimiavaruustunniste http://xmlns.com/foaf/0.1/#
Etuliite rdf, nimiavaruustunniste http://www.w3.org/1999/02/22-rdf-syntax-ns#
Etuliite rdfs, nimiavaruustunniste http://www.w3.org/2000/01/rdf-schema#
Etuliite tktl, nimiavaruustunniste: http://cs.helsinki.fi/ontologia/#
Etuliite xsd, nimiavaruustunniste http://www.w3.org/2001/XMLSchema#

KUVA 10: Kuvitteellisen relaatiotietokannan opiskelija-taulun muuntaminen RDF-muotoon R2RML-muunnoskielen avulla ja tiedon annotointi ulkoisia ontologioita hyödyntäen.



Kuva 11: Muunnosmenetelmän valinta.

perinnejärjestelmien käyttämistä relaatiotietokannoista ei aina ole tarkoituksenmukaista luopua tiedon muuntamisen yhteydessä. Tällöin muunnoksen jälkeen ylläpidettäväksi jää kaksi tietomallia: relaatiomalli ja siitä toisinnettu RDF-tietomalli. Tiedon tallentaminen kahteen kertaan vaatii vastaavasti kertaalleen tehtävää tallennusta enemmän pysyvää tallennustilaa. Tämän lisäksi tiedon päivittäminen on monimutkaisempaa, sillä päivitykset on tehtävä molempiin tietomalleihin. RDF-tietomallin mukaisesti tallennetun tiedon hallintaan tarvitaan erillinen hallintajärjestelmä. Näin varastoitu RDF-muotoinen tieto muodostaa RDF-tietovaraston (engl. RDF store, triple store).

Perinnejärjestelmien relaatiotietokantoihin tekemät päivitykset aiheuttavat ongelmia, jos tieto on muunnettu RDF-muotoon suoran muuntamisen tai muunnoskieliin perustuvan muuntamisen yhteydessä [Seq11]. Päivityksen jälkeen aiemmin RDF-tietomallin mukaiseksi muunnettu tieto ei vastaa relaatiotietokannan nykytilaa. Tilanne korjaantuu vasta, kun muunnos relaatiotietokannasta RDF-muotoon on suoritettu uudelleen. Tämän johdosta päivitykset siirtyvät RDF-tietomalliin aina viipeellä. Sovellusalueesta ja sovelluksen toteutuksesta ja riippuu, miten suuri viive on tai millainen sovelluksen laatuun vaikuttava merkitys sillä on.

W3C-yhteenliittymän Direct Mapping -normin [Are12] ja R2RML-muunnoskielen [DSC12] avulla kuvataan, miten relaatiotietokannan sisältämä tieto muunnetaan ontologioiden avulla semanttisesti annotoidun RDF-tietomallin mukaiseksi. Suositusten mukaisen muuntamisen toteuttamisesta vastaavat normeja tukevat työkalut, jotka mahdollistavat suoran muuntamisen, muunnoskieliin perustuvan muuntamisen tai molemmat. Muunnostyökalujen ei tarvitse välttämättä toisintaa relaatiotietokannan sisältämää tietoa erilliseen RDF-tietovarastoon.

Tiedon toisintamisen sijaan relaatiotietokanta voidaan *peittää* (engl. wrap) eli esittää semanttisen webin sovelluksille semanttisen webin kyselyt mahdollistavalla tavalla [Seq11, Sel07]. Tällöin relaatiotietokannan sisältämää tietoa ei tallenneta erikseen semanttisen webin tietomallin mukaisesti, vaan tieto esitetään ulkopuolisille käyttäjille näennäisessä (engl. virtual) RDF-muodossa. Näennäiseen RDF-tietomalliin kohdistetut kyselyt muunnetaan peitettyyn relaatiotietokantaan kohdistettaviksi SQL-kyselyiksi. Kyselyn muunnos perustuu joko automaattisesti tai käsin jollakin muunnoskielellä tehtyyn määrittelyyn. SQL-kyselyn tuloksena saatu tieto muunnetaan RDF-muotoon, joka palautetaan alkuperäisen kyselyn tuloksena. Muunnos näennäiseen RDF-muotoon voi tapahtua joko suoran muuntamisen tai muunnoskieliin perustuvan muuntamisen periaatteella. Olennaista on, että peittämisessä tieto säilytetään vain relaatiotietokannassa ja tietomallin muunnos tapahtuu ajonaikaisesti. Tämän johdosta tiedon ylläpito on helpompaa, sillä päivitykset on tehtävä vain yhteen tietomalliin. Yhden tietomallin käyttö johtaa myös siihen, että päivitykset näkyvät kaikille käyttäjille välittömästi. Ne näkyvät heti sekä suoraan relaatiotietokantaan tehdyissä SQL-kyselyissä että peittämisen toteuttavalle järjestelmälle tehtävissä SPARQL-kyselyissä. Peittämiseen perustuvan ratkaisun etuna on helpomman ylläpidon lisäksi myös pysyvän tallennustilan säästyminen, koska relaatiotietokannan sisältämää tietoa ei tarvitse toisintaa RDF-muotoon.

Relaatiotietokannan peittämisen toteuttavien järjestelmien huono puoli on, että ajonaikainen SPARQL-kyselyiden muuntaminen SQL-kyselyiksi voi olla hidasta. Kahden vuonna 2009 tehdyn tutkimuksen [BSc09, GGO09] mukaan mikään tuolloin käytössä olleista peittämisratkaisun toteuttaneista järjestelmistä ei pystynyt muuntamaan SPARQL-kyselyitä SQL-kyselyiksi niin tehokkaasti, että ratkaisu olisi ollut nopeudeltaan vertailukelpoinen suoraan relaatiotietokantaan tehtyjen SQL-kyselyiden kanssa. Laajoihin tietokantoihin tehdyt kyselyt olivat pahimmassa tapauksessa jopa 1000 kertaa hitaampia suoriin SQL-kyselyihin verrattuina. Tutkitut D2RQ- [BSe04], Virtuoso- [EMi07] ja Squirrel RDF -järjestelmät [Squ11] ovat edelleen eniten käytettyjä järjestelmiä

relaatiotietokannan peittämiseen [SMi12]. Näistä vain D2RQ ja Virtuoso tukevat joko Direct Mapping- [Are12] tai R2RML -suositusta [DSC12]. Virtuoso tukee R2RML-suositusta [VHa12]. D2RQ tukee Direct Mapping -suositusta [VHa12], mutta siihen on suunnitteilla tuki R2RML-suositukselle [Hau12]. Molemmat järjestelmät mahdollistavat W3C-yhteenliittymän suositteleman SPARQL-kyselykielen [PSe08] käytön.

Sequeda ja kumppanit [SMi12] ovat löytäneet kyselyiden muuntamisen tehokkuusongelmaan merkittävän parannuksen. Heidän kehittämässään *Ultrawrap*-järjestelmässä hyödynnetään oletusta, jonka mukaan relaatiotietokannan hallintajärjestelmien kyselyiden optimoinnista vastaavat algoritmit riittävät SPARQL-kyselyiden tehokkaaseen uudelleenkirjoittamiseen SQL-kyselyiksi. Tämän johdosta kyselyn muunnosta ei kannata tehdä peittämisen toteuttavassa järjestelmässä kokonaan, vaan kyselyiden optimointi kannattaa jättää mahdollisimman pitkälti relaatiotietokannan hallintajärjestelmien kyselyn optimoijien tehtäväksi. Ultrawrap-järjestelmän kyselyn muuntamisen toimintaperiaate on yksinkertainen: se korvaa SPARQL-kyselyn sisältämiä merkkijonoja SQL-kielen vastineilla ja antaa relaatiotietokannan hallintajärjestelmän huolehtia kyselyn optimoinnista. Menetelmä toimii kahdesta syystä. Yhtäältä SPARQL- ja SQL-kielet ovat ilmaisuvoimaltaan toisiaan vastaavia [AGu08]. Toisaalta SPARQL-kielen määrittelyssä [PSe08] monet operaattorit määritellään relaatioalgebran operaattoreita vastaaviksi, joten SPARQL-kielen operaattoreita voidaan korvata suoraan SQL-kielen operaattoreilla. Ultrawrap-järjestelmä on toistaiseksi ainoa relaatiotietokannan peittämisen mahdollistava järjestelmä, jonka kyselyiden muuntamisen on todettu vastaavan suorituskyvyltään lähestulkoon suoria SQL-kyselyitä [SMi12]. Tämän lisäksi huomion arvoista on, että Ultrawrap-järjestelmä näyttää olevan kaikkia tunnettuja erillisiä RDF-tietovarastoja ylläpitäviä järjestelmiä nopeampi [SMi12]. Koska kyselyjen suorituskyvystä vastaaminen siirretään relaatiotietokannan hallintajärjestelmän kyselyn optimoijalle, tehokkaiden kyselyiden mahdollistamisen ehtona on tehokas optimoija [SMi12].

Tehokkaan kyselyiden muuntamisen lisäksi Ultrawrap-järjestelmä on selvinnyt kiitettävästi W3C-yhteenliittymän muunnosjärjestelmien toimivuutta tutkivista testitapauksista [VHa12], joten se toteuttaa laadukkaasti W3C-yhteenliittymän R2RML- [DSC12] ja Direct Mapping -suositukset [Are12]. Tämän johdosta Ultrawrap-järjestelmä erottuu edukseen tunnettujen peittämisen mahdollistavien järjestelmien joukosta. Ultrawrap mahdollistaa SPARQL-kyselykielen [PSe08] käytön.

4.6 Tunnisteet

Luvussa 3 nähtiin, että semanttisen webin menetelmiä käytettäessä tiedon rakenteet yksilöidään IRI-tunnisteilla. Ne mahdollistavat internet-yhteensopivan tavan viitata täsmällisesti asioihin ja luovat pohjan verkkomuotoiselle RDF-tietomallille. Perinnejärjestelmissä tiedon yksilöimiseen ei yleensä käytetä IRI-tunnisteita, joten perinnetiedon semanttisen annotoinnin yhteydessä on määritettävä tarvittavat IRI-tunnisteet. Niiden luominen kannattaa suunnitella huolellisesti.

Hyvien IRI-tunnisteiden ominaisuuksia ovat *ainutlaatuisuus*, *yksinkertaisuus, muuttumattomuus*, *ylläpidettävyys ja selvitettävyys* [Ber98a, SCy08]. Ainutlaatuisuudella tarkoitetaan, että yksi IRI-tunniste ei voi viitata kahteen eri asiaan. IRI-tunnisteiden ainutlaatuisuus voidaan varmistaa luomalla niitä ainutlaatuista nimiavaruutta käyttäen. Yksinkertaisuudella tarkoitetaan rakenteeltaan yksinkertaisia ja helposti muistettavia IRI-tunnisteita. Muuttumattomuudella tarkoitetaan, että IRI-tunnisteet eivät muutu olemassaolonsa aikana. Ylläpidettävyydellä tarkoitetaan, että IRI-tunnisteita luodaan ylläpidon mahdollistavalla tavalla. Tämä tarkoittaa esimerkiksi päivämäärän liittämistä osaksi IRI-tunnistetta. Tällöin tunnistepolkua voidaan muokata rikkomatta vanhempia IRI-tunnisteita. Selvitettävyydellä tarkoitetaan kuvauksen saatavuutta IRI-tunnisteen viittaamasta kohteesta. Selvitettävillä

IRI-tunnisteilla yksilöidyt tiedon rakenteet ovat koneellisesti tulkittavissa. Tämä mahdollistaa, että käyttäjät ja sovellukset voivat halutessaan lukea yksilöityä tiedon rakennetta kuvaavan asiakirjan WWW-palvelun menetelmillä. Tämän johdosta IRI-tunnisteet kannattaa muodostaa niin, että ne ovat käytännössä HTTP URI -tunnisteita [Ber05]. Selvitettävyyden toteuttaminen tuo lisähaasteita. Sen mahdollistamiseksi esimerkiksi ontologiat on sijoitettava palvelimille, joilla niiden rakenteille on mahdollista luoda ainutlaatuisia ja muuttumattomia HTTP URI -tunnisteita.

Selvitettävyyden mahdollistamiseksi HTTP URI -tunnuksia voidaan käyttää kahdella tavalla [SCy08]: ristikkomerkkiin (engl. hashtag) tai sisältöneuvotteluun (engl. content negotiation) perustuvina. Nämä menetelmät mahdollistavat, että käyttäjät voivat erottaa reaalimaailman kohteita ja niitä kuvaavia WWW-palvelun asiakirjoja osoittavat tunnisteet toisistaan.

Ristikkomerkkiin perustuvassa menetelmässä reaalimaailman kohteita kuvaavien tunnisteiden loppuun lisätään ristikkomerkillä alkava osa. Tunnisteen selvittävää palvelupyyntöä tehtäessä tämä osa poistetaan HTTP-yhteyskäytännön mukaisesti. Tällöin jäljelle jää kuvaavan asiakirjan tunniste.

Sisältöneuvotteluun perustuva menetelmä perustuu HTTP-sisältöneuvotteluun. Sen avulla palvelupyynnössä voidaan ilmoittaa sisältötyyppi, jonka palvelimen toivotaan tarjoavan. Tämän perusteella palvelupyyntö voidaan ohjata esimerkiksi RDF- tai HTML -muotoiseen kohdetta kuvaavaan asiakirjaan. Osoitetuilla kohteilla ja niitä kuvaavilla WWW-palvelun asiakirjoilla on erilliset HTTP URI -tunnisteet. Menetelmiä voidaan käyttää yhdessä. Tällöin käytetään ristikkomerkkiin perustuvaa menetelmää, mutta kohdetta kuvaavan asiakirjan sisältö tarjotaan sopivassa muodossa HTTP-sisältöneuvottelun perusteella. Ristikkomerkkiin perustuvaa menetelmää kannattaa käyttää, kun tunnistettavien kohteiden määrä on suhteellisen pieni ja selvitysmuodot on

helppo jakaa yhden asiakirjan kautta. Tällaisia tapauksia ovat esimerkiksi ontologiat, joissa käsitteiden määrä pysyy kohtuullisena. Puhtaasti sisältöneuvotteluun perustuvalla menetelmällä on helpompi jakaa eri muotoinen tieto eri asiakirjoihin suurten tietovarastojen tapauksessa. Sisältöneuvottelun avulla saadaan luettavampia HTTP URI -tunnisteita, mutta sen käyttö kuormittaa palvelinta ristikkomerkkiin perustuvaa menetelmää enemmän.

Relaatiotietokantojen tietosisältöä muunnettaessa on tärkeää huomioida, että muunnoksen yhteydessä luotavat IRI-tunnisteet säilyvät samoina, vaikka muunnos tehdään toistuvasti rakenteeltaan ja sisällöltään muuttumattomaan tietokantaan. Tämä onnistuu vain noudattamalla tiettyä menetelmää. Muunnosmenetelmiä käytettäessä IRI-tunnisteet voidaan luoda muunnostyökalujen toimesta relaatiotietokannan rakenteiden pohjalta [Are12, DCS12]. Tämä tarkoittaa IRI-tunnisteen luontia esimerkiksi relaatiotietokannan HTTP URI -tunnuksen sekä taulun ja sarakkeen nimen perusteella. Tällöin samalla työkalulla useamman kerran tehtävä muunnos voi tuottaa relaatiotietokannan rakenteista aina samat IRI-tunnisteet. Eri työkalut saattavat käyttää toisistaan poikkeavia menetelmiä IRI-tunnisteiden luontiin. Tämän johdosta yhteisten käytäntöjen eli W3C-yhteenliittymän suosituksia noudattavien muunnostyökalujen noudattaminen on hyödyllistä. Kun työkalujen tiedetään noudattavan samoja muunnosmenetelmiä, voidaan luottaa niiden tuottavan samanlaisia IRI-tunnisteita samasta relaatiotietokannasta. Näin ollen perinnetiedon semanttisen annotoinnin yhteydessä hyvien IRI-tunnisteiden ominaisuuksiin on perusteltua lisätä *johdonmukaisuus*. Johdonmukaisuudella tarkoitetaan, että muuttumattoman relaatiotietokannan tietosisällön toistuva muuntaminen RDF-tietomallin mukaiseksi tuottaa samoille tiedon rakenteille samat IRI-tunnisteet muunnoskerrasta toiseen.

Muunnostyökaluja käytettäessä riittää, että muunnoksen perustana käytettävä IRI-tunnus mahdollistaa edellä kuvattujen hyvien ominaisuuksien toteutumisen. Muunnoksen perustana voidaan käyttää esimerkiksi relaatiotietokannan HTTP

URI -tunnusta, jonka perusteella muunnostyökalu luo tiedon rakenteille yksilöivät tunnukset. Näin ollen perinnetiedon semanttisen annotoinnin näkökulmasta suurin tunnisteiden suunnittelua koskeva työ kohdistuu ontologioiden rakenteisiin viittaavien tunnisteiden suunnitteluun.

5 Perinnetiedon semanttinen annotointi käytännössä

Tässä luvussa kuvataan esimerkin avulla, miten perinnetiedon semanttista annotointia voidaan soveltaa käytännössä. Aluksi tarkastellaan kuvitteellista lähtötilannetta ja tavoitteita, jotka semanttisella annotoinnilla voidaan saavuttaa. Tämän jälkeen valitaan esimerkkiin soveltuva annotointimenetelmä ja sen käyttö kuvataan pääpiirteittäin.

5.1 Lähtötilanne ja tavoitteet

Tarkastellaan esimerkkinä tilannetta, jossa Helsingin yliopiston tietojenkäsittelytieteen laitoksen opiskelijoista, kursseista ja niille osallistumisista ylläpidetään tietoa relaatiotietokannassa. Opiskelijoista ylläpidetään henkilökohtaista tietoa, kuten etunimi, sukunimi, opiskelijanumero, osoite, puhelinnumero, sähköpostiosoite ja opintojen aloitusvuosi. Kursseista ylläpidettäviä tietoja ovat esimerkiksi nimi, laajuus, alkamis- ja päättymisajankohta, arvostelija, kieli ja kotisivun osoite. Tämän lisäksi ylläpidetään tietoa siitä, mille kursseille eri opiskelijat osallistuvat. Samaan aikaan matematiikan laitoksen opiskelijoista ylläpidetään vastaavia tietoa erillisessä relaatiotietokannassa.

Esimerkissä kuvattujen erillisten tietokantojen välillä on rakenteellisia ja terminologisia eroja, vaikka niissä ylläpidetään saman aihepiirin tietoa. Tämä johtuu siitä, että tietokannat on suunniteltu eri kehittäjien toimesta. Tavoitteena on, että molempien tietokantojen sisällöt annotoidaan semanttisesti ja nähdään jatkossa yhtenä yhdistettynä tietomallina, johon voidaan suorittaa kyselyitä SPARQL-kielellä.

5.2 Annotointimenetelmän valinta

Soveltuva annotointimenetelmä voidaan valita kuvassa 11 esitettyä kaaviota seuraamalla. Tässä tapauksessa relaatiotietokanta ei sisällä sanastoa, joka halutaan muuntaa ontologiaksi. Tämän lisäksi ei ole tarpeellista käyttää erityiseen käyttötarkoitukseen soveltuvaa muunnoskieltä tai toteuttaa kaksisuuntaista muunnosta, joten valinta tehdään suoran muuntamisen ja yleisiin lukemisen mahdollistaviin muunnoskieliin perustuvan muuntamisen välillä.

Mikäli helppo toteutus asetetaan annotaatioiden laadun edelle, relaatiotietokantojen rakenne tai sisältö tunnetaan heikosti tai relaatiotietokanta voidaan esittää suoraan osana annotoinnissa käytettyä ontologiaa, valitaan suora muuntaminen. Tässä tapauksessa päätetään, että käytettävässä esimerkkitapauksessa viimeinen ehdoista täyttyy, joten menetelmäksi valitaan suora muuntaminen.

Suoran muuntamisen toteuttamiseen valitaan käytettäväksi D2RQ-järjestelmä [BCy07, BSe04], joka sisältää työkaluja muunnoksen tekemiseen. Valintaperusteita ovat W3C-yhteenliittymän Direct Mapping -normin tuki, sallivat käyttöehdot, riittävän hyvä ohjeistus järjestelmän käyttämiseen, peittämisratkaisun mahdollistaminen sekä järjestelmän kuuluminen yleisimmin käytettyjen muunnostyökalujen joukkoon. D2RQ-järjestelmän heikkouksia ovat

vielä tällä hetkellä puuttuva tuki R2RML-muunnoskielen käytölle ja peittämisratkaisussa ajonaikaisesti muunnettavien kyselyjen hitaus. Mainituilla heikkouksilla ei kuitenkaan ole ratkaisevaa merkitystä esimerkin tapauksessa.

5.3 Valitun menetelmän soveltaminen

D2RQ-järjestelmässä suoran muuntamisen tapauksessa käytetään *generate-mapping*-sovellusta. Sovelluksen syötteenä annetaan tapauskohtaisesti parametreja, joiden pohjalta muunnos tehdään. Parametreja voivat olla esimerkiksi muunnettavan tietokannan yhteysasetukset, tulosteen tiedostonimi, luotavan oletetun ontologian URI-tunnisteen perusosa, käytetäänkö muunnoksessa W3C-yhteenliittymän Direct Mapping -suositusta, mitä tietokannan rakenteita käytetään ja luodaanko ontologia vai muunnoskuvaus. Sovellus tuottaa joko oletetun ontologian tai D2RQ-muunnoskielellä esitetyn muunnoskuvauksen, jonka perusteella relaatiotietokannan tietosisältö muunnetaan RDF-tietomallin mukaiseen muotoon.

D2RQ-järjestelmän *generate-mapping*-sovelluksen avulla esimerkin tilanteessa kuvatuista relaatiotietokannoista luodaan oletetut ontologiat ja muunnoskuvaukset. Tuloksena on yksi kutakin relaatiotietokantaa kuvaava ontologia sekä relaatiotietokannan ja sitä kuvaavan ontologian väliset muunnoskuvaukset. Toisin sanoen tuloksena on kaksi erillistä RDF-tietomallia, joihin voidaan tehdä erillisiä SPARQL-kyselyitä.

Esimerkin tapauksessa tietojenkäsittelytieteen laitoksen opiskelijatietokannan opiskelijoista ja kursseista luodaan ontologian käsitteet *opiskelija* ja *kurssi*. Käsitteiden ominaisuuksiksi määritellään tietokannan sisältämät ominaisuudet. Esimerkiksi *opiskelija*-käsitteen ominaisuuksiksi määräytyvät etunimi, sukunimi, opiskelijanumero, osoite, puhelinnumero, sähköposti ja opintojen aloitusvuosi. *Kurssi*-käsitteen ominaisuuksiksi määräytyvät esimerkiksi nimi,

laajuus, alkamis- ja päättymisajankohta, arvostelija ja opetuskieli. *Osallistuminen* voidaan määritellä *opiskelija-* ja *kurssi-* käsitteiden väliseksi ominaisuudeksi. Suoran muuntamisen tapauksessa lopullinen muoto riippuu todellisesta relaatiomallista. Matematiikan laitoksen opiskelijatietokannasta luotava oletettu ontologia luodaan vastaavalla tavalla sen todellisten rakenteiden perusteella. Tällöin tästä tietokannasta luodun oletetun ontologian opiskelijoita ja kursseja kuvaaviksi käsitteiksi voisi muodostua esimerkiksi *student* ja *course*.

Koska lähtötilanteessa tavoitteena oli muodostaa yksi yhdistetty tietomalli, tietokantojen rakenteet on kuvattava jonkin yhteisen ontologian avulla. Yhteinen ontologia saadaan yhdistämällä aiemmin luodut oletetut ontologiat toisiinsa tai yhteiseen kolmanteen ontologiaan. Ontologioiden yhdistäminen tapahtuu luomalla yhteyksiä niiden käsitteiden välille [HHa08]. Tällöin esimerkin tapauksessa voidaan määritellä, että oletettujen ontologioiden käsitteet *opiskelija* ja *student* ovat toisiaan vastaavia käsitteitä. Vastaava määrittely voidaan tehdä käsitteiden *kurssi* ja *course* välille. Tämän lisäksi voidaan määritellä muita soveltuvia yhteyksiä oletettujen ontologioiden rakenteiden välille. Tämä onnistuu esimerkiksi luvussa 4.1 kuvattujen ontologioiden kehitysmenetelmien ja -työkalujen avulla.

Muunnoskieliin perustuvaa muuntamista hyödyntämällä voitaisiin käyttää yhteistä ontologiaa ilman oletettuja ontologioita. Tällöin relaatiotietokantojen rakenteet voitaisiin kuvata yhteisen ontologian pohjalta suoraan *generate-mapping*-sovelluksella luotuja muunnoskuvauksia muokkaamalla.

D2RQ-järjestelmä sisältää peittämisratkaisun toteuttavan palvelinsovelluksen, jonka kautta RDF-muotoon muunnettua tietosisältöä voidaan selata ja siihen voidaan esittää SPARQL-kyselyitä. Palvelinsovellus yhdistää annotoinnissa käytetyt ontologiat ja relaatiotietokantojen tietosisällöt aiemmin luodun

muunnoskuvauksen perusteella. Sekä suoran että muunnoskieliin perustuvan muuntamisen lopputuloksena saadaan esimerkkitapauksessa virtuaalinen RDF-tietomalli, jossa kahden eri relaatiotietokannan tietosisällöt on yhdistetty ja annotoitu semanttisesti ontologioiden avulla. D2RQ-järjestelmä vastaa kyselyiden ja tietomallin muuntamisesta sekä sisältää käyttöliittymän RDF-tietomallin selaamiseen ja SPARQL-kyselyjen esittämiseen.

SPARQL-kyselyiden avulla virtuaaliseen RDF-tietomalliin voidaan esittää täsmällisiä ja koneymmärrettäviä kyselyitä. Tämän johdosta koneelta voidaan alkuperäisten tietokantojen rakenteita tuntematta kysyä esimerkiksi mitä asioita tietystä opiskelijasta tiedetään tai kenellä opiskelijoista on osaamista logiikasta ja semanttisen webin menetelmistä. Tietokantojen yhdistämisen näkökulmasta etuna on, että tietomalliin voidaan tehdä hyödyllisiä kyselyitä, jotka suoritetaan käytännössä molempien tietokantojen tietosisältöihin.

6 Tulosten arviointi

Semanttisen webin menetelmät tuovat perinteisistä ohjelmistoarkkitehtuureista poikkeavia ajatusmalleja tietämyksen hallintaan. Perinteisesti on totuttu suunnittelemaan yksi sovelluskohtainen tietomalli, johon tarjotaan tarvittava näkymä. Semanttisessa webissä tieto ja sitä kuvaava koneymmärrettävä metatieto on kaiken ydin. Koska tiedon merkitykset ovat tietoa käsittelevien koneiden luettavissa, merkityksiä ei tarvitse sisällyttää sovelluskohtaiseen ohjelmakoodiin. Tämän johdosta semanttisesti annotoitu perinnetieto mahdollistaa useiden erilaisten näkymien luonnin useista eri näkökulmista sovellusriippumattomasti.

Osana perinnetiedon hallittavuuden parantamista semanttisen webin menetelmät helpottavat merkittävästi tiedon yhdistämistä eri tietolähteistä. Tämä johtuu suurelta osin RDF-tietomallin joustavuudesta. Kun eri tietokantojen sisältämä tieto on muunnettu useaksi verkkomuotoiseksi RDF-tietomalliksi, ne voidaan helposti yhdistää yhdeksi suuremmaksi kokonaisuudeksi. Yhdistäminen tapahtuu luomalla yhteyksiä verkon solmujen välille tai yhdistämällä solmuja keskenään. Lopputuloksena eri tietokannoissa varastoitua perinnetieto voidaan esittää yhtenä yhdistettynä tietomallina, johon voidaan tehdä erittäin ilmaisuvoimaisia SPARQL-kyselyitä.

Semanttisen webin menetelmiin perustuva perinnetiedon semanttinen annotointi perustuu koneymmärrettäviin käsitemalleihin eli ontologioihin. Kaikkien semanttisen webin menetelmien hyödyntäjien ei kuitenkaan voida olettaa jakavan samanlaista käsitystä tietyn aihealueen käsitteistä. Esimerkiksi eri organisaatioissa ja eri sovellusalueilla voidaan nähdä samat käsitteet eri tavoin. Tämän johdosta ontologioiden kehittämisessä ei yleisesti kannata pyrkiä yhteen täydelliseen käsitemalliin. Samasta syystä olemassa olevien ontologioiden käyttö ei välttämättä ole sopivin vaihtoehto.

Semanttisen webin menetelmät tukevat hajautettua arkkitehtuuria, johon kuuluu mahdollisesti useita eri näkökulmia esiintuovia tietolähteitä. Näin ollen ei ole välttämätöntä pyrkiä täydellisen käsitemallin kehittämiseen, vaan ontologioiden kuvaamat käsitesuhteet ja näkökulmat voidaan rajata sovellusalueen kannalta tarkoituksenmukaisella tavalla. Näin voidaan välttää esimerkiksi käsitteen tai käsitesuhteen liian syvällisestä pohtimisesta johtuva ontologian kehitystyön hidastuminen. Perinnetiedon semanttisessa annotoinnissa voidaan keskittyä esittämään tiettyä näkökulmaa rajatussa aihepiirissä, mikä auttaa pitämään kehitystyön kustannukset kohtuullisina.

Eri aihealueita kuvaavien ontologioiden kehittämiseen tarvitaan eri alojen asiantuntijoita, jotka eivät välttämättä tunne semanttisen webin menetelmien toimintaperiaatteita. Tällöin esimerkiksi ontologioiden kehityksessä ratkaisevan tärkeässä asemassa ovat helppokäyttöiset työkalut, joilla toimintaperiaatteet voidaan piilottaa ontologioiden kehittäjiltä.

Kun formaalin annotointikielen ilmaisuvoimaa lisätään, se mahdollistaa vastaavasti ilmaisuvoimaisempien kyselyjen esittämisen. RDFS-kielen ilmaisuvoima on rajallinen: vasta OWL-kielten mahdollistamat rajoitteet ja erityispiirteet mahdollistavat ontologioiden koneellisen arvioinnin. Tämä helpottaa ontologioiden laadunvarmistusta. Ilmaisuvoimaisempien kyselyjen ja ontologioiden koneellisen arvioinnin mahdollistamiseksi ontologioita kehitettäessä kannattaa käyttää mahdollisimman ilmaisuvoimaisia OWL-kielten ominaisuuksia. Samalla on kuitenkin tiedettävä, missä laskennallisen ratkeavuuden raja kulkee. Tällöin voidaan varmistaa, että määriteltyjen ilmaisujen loogiset seuraukset ovat päättelykoneiden pääteltävissä polynomisessa ajassa. Ratkeamattomia ilmaisuja voidaan perustellusta syystä käyttää, mutta tämä voidaan tehdä suunnitellusti, kun ratkeavuuden rajat ovat tiedossa.

Relaatiotietokannoissa varastoitua tietoa ei kannata kuvata määrittelemällä tiedon ja ontologioiden välille ulkoisia RDF-lauseita, koska tämä on hidasta. Sen sijaan relaatiotietokannan sisältämä tieto kannattaa muuntaa RDF-tietomallin mukaiseen muotoon aihealuetta kuvaavan ontologian käsitteiden ilmentymiksi. Tämä voidaan tehdä automaattisten tai puoliautomaattisten työkalujen avulla. Tiedon toisintamisen välttämiseksi muuntaminen voidaan toteuttaa näennäisesti eli peittämällä relaatiotietokanta erillisen sovelluksen avulla. Tällöin tieto säilytetään relaatiotietokannassa, jolloin perinnejärjestelmät voivat edelleen käyttää ja ylläpitää sitä. Peittävälle sovellukselle SPARQL-kyselyitä esittävän toimijan, kuten semanttisen webin sovelluksen näkökulmasta tieto näyttää semanttisen webin tietomallin mukaiselta. SPARQL-kyselykieli

mahdollistaa käytännöllisten kyselyiden teon. Sen hyödyt tulevat selkeimmin esiin, kun yhden kyselyn vastaukseen tarvittavaa tietoa täytyy yhdistää eri tietolähteistä.

Peittämisratkaisun heikkous on kyselyjen muuntamisesta johtuva viive. Vastikään on kuitenkin löydetty ratkaisu, miten peittämisessä SPARQL-kyselyjen muunnos SQL-kyselyiksi toteutetaan tehokkaasti. Ratkaisu luottaa relaatiotietokannan hallintajärjestelmän kyselyn optimoinnista vastaavien algoritmien tehokkuuteen, joten se on riippuvainen tehokkaista kyselynoptimoijista. Tällaisen lähestymistavan keksimisen myötä tehokkaat kyselyt mahdollistavien peittämistyökalujen voidaan olettaa lisääntyvän lähitulevaisuudessa.

Tietomallin muuntamisessa W3C-yhteenliittymän Direct Mapping- ja R2RML -suositukset ovat erittäin tärkeitä. Niiden ansiosta käytettävissä on kaksi käytännöllistä, SQL-kieltä hyödyntävää tapaa perinnetiedon semanttiseen annotointiin. Suositukset mahdollistavat yhteisen näkemyksen, miten relaatiotietokantojen tietosisältöjä voidaan esittää semanttisen webin tietomallin mukaisessa muodossa semanttisesti annotoituna. Vaikka Direct Mapping- ja R2RML -suositukset ovat uusia, niitä tukevia työkaluja löytyy jo. Yhteiset menetelmät mahdollistavat niitä tukevien työkalujen kehittämisen useiden eri tahojen toimesta. Tämän voidaan olettaa lisäävän jatkossa saatavilla olevien työkalujen määrää. Laadukkaat työkalut helpottavat semanttisen annotoinnin toteutusta. Näin ollen ne edesauttavat semanttisen webin menetelmien käyttöönottoa.

Direct Mapping -suositukseen perustuva suora muuntaminen on muunnoskieliin perustuvaa muuntamista yksinkertaisempi ratkaisu, koska se voidaan tehdä täysin automaattisesti. Tällöin muunnoksen tekijän ei tarvitse tuntea relaatiotietokannan rakennetta tai sisältöä eikä määritellä, miten muunnos

tehdään. Tämä on samalla Direct Mapping -normin huono puoli, sillä muunnoksen tekijä ei voi määritellä annotaatioita ulkoisten ontologioiden avulla. Muunnoksen tekijän on kuitenkin mahdollista luoda relaatiotietokannan sisältämästä tiedosta erilaisia SQL-näkymiä ja määritellä, mitkä taulut ja näkymät muunnetaan. Tämän lisäksi on mahdollista, että muunnoksen tekijä laajentaa suoran muunnoksen tuloksena saatua oletettua ontologiaa tai yhdistää sen johonkin toiseen ontologiaan, jolloin kuvaus monipuolistuu.

Muunnoskieliin perustuvan muuntamisen toteuttaminen on työläämpää, mutta sillä voidaan paremmin vaikuttaa lopputulokseen laatuun. Muunnoskielillä voidaan määritellä yhteyksiä relaatiotietokannan ja ulkoisen ontologian rakenteiden välille. Muunnos tehdään määriteltyjen yhteyksien perusteella. Laadukkaan lopputuloksen edellytyksenä on, että tietokannan rakenne ja sisältö tunnetaan. Laadukkaan ja W3C-yhteenliittymän suosituksia noudattavan esityksen aikaansaamiseksi relaatiotietokanta kannattaa yhdistää aihealuetta kuvaavaan ontologiaan R2RML-muunnoskielellä. Ylläpidon näkökulmasta kannattaa tällöin huomioida, että relaatiotietokantakaavioon tai ontologiaan tehtävien muutosten yhteydessä myös muunnosta varten tehtyä muunnoskuvausta voidaan joutua muuttamaan.

Muihin muunnoskieliin verrattuna R2RML-muunnoskieltä käytettäessä voidaan varautua paremmin tuleviin muutoksiin. Koska R2RML-muunnoskieli on W3C-yhteenliittymän suositus, sitä tukevien järjestelmien määrän voidaan olettaa lisääntyvän jatkossa. Näin ollen R2RML-muunnoskielellä kertaalleen määriteltyä muunnoskuvausta voidaan mahdollisesti uudelleenkäyttää, vaikka tietomallin muunnoksessa tai peittämisratkaisussa käytettävä järjestelmä vaihdettaisiin myöhemmin toiseen.

Paitsi erikseen suorassa ja muunnoskieliin perustuvassa muuntamisessa, Direct Mapping- ja R2RML -suosituksia voidaan käyttää myös yhdessä. Ensin voidaan

esimerkiksi luoda muunnoskuvaus automaattisesti Direct Mapping -suosituksen mukaisesti ja tutustua muunnoksen tulokseen. Tämän jälkeen haluttuja muutoksia voidaan määritellä R2RML-muunnoskielellä. Tällöin koko muunnoskuvausta ei tarvitse välttämättä kirjoittaa alusta, vaan lopullinen muunnoskuvaus voidaan toteuttaa muokkaamalla suoran muuntamisen periaatteella luotua pohjaa. Vastaavaa lähestymistapaa voidaan hyödyntää R2RML-kielen lisäksi muidenkin muunnoskielten tapauksessa.

Tutkielman tavoitteena oli selvittää, miten perinnetietoa voidaan annotoida semanttisesti. Tämän lisäksi tavoitteena oli vertailla eri menetelmiä ja selvittää, miten voidaan löytää sovellusalueen kannalta tarkoituksenmukainen annotointimenetelmä, joka mahdollistaa riittävän ilmaisuvoimaisten kyselyjen suorittamisen aiheuttaen mahdollisimman vähän ylimääräistä työtä.

Tutkielmassa esiteltiin ja vertailtiin eri menetelmiä, joilla perinnetieto voidaan annotoida ja esittää semanttisen webin tietomallin mukaisesti. Menetelmien vertailussa havaittiin, että yhtä tarkoituksenmukaisinta menetelmää ei ole olemassa. Sen sijaan on olemassa eri tilanteisiin sopivia menetelmiä, joista kannattaa valita tilanteeseen sopivin. Tutkielmassa esitettiin, miten tarkoituksenmukaisin menetelmä voidaan valita.

Tutkielmassa kuvattiin esimerkkinä tilanne, johon semanttista annotointia voidaan hyödyntää. Tilanteeseen soveltuva menetelmä valittiin tutkielmassa esitettyjen valintakriteerien mukaisesti. Tämän jälkeen kuvattiin, miten perinnetiedon semanttinen annotointi voidaan esimerkkinä kuvatussa tilanteessa toteuttaa olemassa olevilla työkaluilla. Tutkielman perusteella semanttisen webin menetelmien hyödyntäminen perinnetiedon semanttiseen annotointiin onnistuu jo tällä hetkellä kohtuullisilla kustannuksilla. Tämän lisäksi odotettavissa on, että vakiintuvien menetelmien ja työkalujen kehityksen myötä kustannustehokkuus paranee jatkossa.

Semanttisen webin menetelmien käyttöönottoa edesauttavat niiden hyödyllisyys ja helppokäyttöisyys. Koska semanttisen webin menetelmien hyödyntäminen mahdollistaa olennaisesti ilmaisuvoimaisempien kyselykielten käytön, vaatimus hyödyllisyydestä täyttyy. Helppokäyttöisyyteen liittyvät ongelmat ovat suurelta osin ratkaistavissa työkalujen kehityksen myötä. Hyviä työkaluja on jo olemassa esimerkiksi ontologioiden kehitykseen ja relaatiotietokantojen tietosisällön muuntamiseen ontologioiden ilmentymiksi. Direct Mapping- ja R2RML -suositusten myötä näyttää siltä, että jatkossa useat muunnostyökalut tulevat noudattamaan yhteisiä periaatteita ja käyttämään yhteistä muunnoskieltä.

Tutkielman perusteella perinnetiedon semanttisella annotoinnilla voidaan kohtuullisilla resursseilla luoda pohja monipuolisille semanttisen webin sovelluksille, jotka auttavat ihmiskäyttäjiä hallitsemaan ja löytämään perinnejärjestelmien varastoimaa tietoa merkittävästi nykyistä paremmin. Eri ohjelmointikielet ja -ympäristöt eivät kuitenkaan tue semanttisen webin menetelmiä vielä kattavasti, mikä hidastaa semanttista metatietoa hyödyntävien sovellusten kehittämistä. Estettä sovellusten toteuttamiselle ei kuitenkaan ole. Semanttisen webin menetelmien käyttöönoton lisääntyessä ohjelmointikielten ja -ympäristöjen tuen voidaan odottaa paranevan nykyisestä.

7 Yhteenveto

Organisaatioiden perinnejärjestelmät sisältävät valtavan määrän tietoa tallennettuna relaatiotietokantoihin. Tämä tieto on usein erittäin tärkeää, mutta vaikeasti hallittavissa ja sitä ylläpitävien perinnejärjestelmien varassa. Tämä johtuu siitä, että tietoa on usein valtava määrä, mutta tietoa ja sitä ylläpitävää perinnejärjestelmää ei ole kuvailtu riittävästi. Tällöin esimerkiksi tiedon hakijan

on osattava etsiä tietoa oikeasta paikasta, oikeilla menetelmillä, tunnettava ennalta aineistoa ja osattava tulkita hakutuloksia.

Koneet voivat auttaa ihmisiä tietämyksen hallinnassa oleellisesti nykyistä paremmin, jos ne ymmärtävät tietoa, siihen liittyviä asiayhteyksiä ja ympäröivää maailmaa. Tämä vaatii tietämyksen esittämistä koneiden ymmärtämällä tavalla. Koneet vaativat asioiden määrittelyä täsmällisillä ja johdonmukaisilla ilmaisuilla. Tämä puolestaan ei ole ihmisille luonteenomaista, joten ongelmana on ollut, miten tietämys voidaan esittää koneille ymmärrettävällä tavalla.

Semanttinen web on näkemys WWW-palvelun laajennuksesta, jossa tieto on kuvailtu koneymmärrettävällä tavalla. Semanttisen webin toteuttamiseksi on kehitetty joukko menetelmiä, joilla tietoa voidaan kuvata koneymmärrettävästi. Tämä tapahtuu esittämällä tietoa kuvailevaa ja määrittelevää metatietoa formaalilla, koneymmärrettävällä kielellä. Semanttisen webin menetelmät pohjautuvat pitkällä aikavälillä tehtyyn tutkimukseen ja käytännön tasolla toimiviksi todistettuihin menetelmiin.

Semanttisen webin avainmenetelmiä ovat verkkomuotoinen RDF-tietomalli ja aihealueita kuvaavat formaalit käsitemallit eli ontologiat. RDF-tietomallin avulla tieto voidaan esittää yhteenlinkitettynä. Ontologioiden avulla aihealueen käsitteiden välille voidaan kuvata yhteyksiä, jotka selittävät tiedon merkityksiä. Ontologioiden kehittämiseen on olemassa monipuolisia työkaluja, jotka mahdollistavat esimerkiksi ontologioiden muokkaamisen, yhdistämisen, versionhallinnan ja arvioinnin. Ontologioiden kehitystyö sopii hyvin yhteen ketterien prosessimallien kanssa.

Menetelmiä hyödyntävät sovellukset ovat käytännössä verkkoteorian ja kuvauslogiikoiden ilmentymiä, minkä johdosta niillä on vahva teoreettinen pohja. Menetelmien käytännön toteuttamiseen käytetään useita internetin

toteutuksessa toimiviksi todettuja menetelmiä ja W3C-yhteenliittymän määrittelemiä normeja. Näitä ovat esimerkiksi IRI-tunnisteet, HTTP-yhteyskäytäntö ja XML-merkkauskieli.

Semanttisen webin menetelmiä voidaan käyttää perinnetiedon semanttiseen, koneymmärrettävään kuvailuun. Tämä aiheuttaa lisätyötä, mutta mahdollistaa perinteistä ilmaisuvoimaisempien kyselykielten käytön. W3C-yhteenliittymän suositus semanttisen webin kyselykieleksi on SPARQL. SPARQL-kyselykieli vastaa ilmaisuvoimaltaan relaatioalgebraa. Sillä voidaan kysellä RDF-tietomallin mukaisessa muodossa olevaa tietoa semanttisella tasolla. Tämä tarkoittaa, että kyselyissä voidaan huomioida sekä tieto että ontologiapohjaisesti esitetty metatieto. Tämän johdosta tietoa voidaan kysellä monipuolisesti ja siitä voidaan luoda erilaisia näkymiä sovellusriippumattomasti.

Relaatiotietokannan tietosisältö kannattaa annotoida muuntamalla se semanttisen webin tietomallin mukaiseksi soveltuvien ontologioiden ilmentymiksi. Tähän voidaan käyttää tietokannan rakenteiden pohjalta automaattisesti luotua ontologiaa tai ulkoisia ontologioita. Suora muuntaminen on menetelmä, jossa relaatiotietokannan rakenteiden pohjalta luodaan automaattisesti rakenteita kuvaava ontologia. Tämän jälkeen relaatiotietokannan tietosisältö muunnetaan ontologian ilmentymiksi.

Muunnoskieliin perustuvassa muuntamisessa voidaan käyttää ulkoisia ontologioita. Tällöin relaatiotietokannan ja ontologioiden rakenteiden välille määritellään yhteyksiä jollakin muunnoskielellä. Tämän jälkeen relaatiotietokannan tietosisällön muunnos ulkoisten ontologioiden ilmentymiksi tehdään määriteltyjen yhteyksien perusteella.

Muunnosten tekemiseen on olemassa uudet W3C:n suositukset Direct Mapping ja R2RML. Direct Mapping -suositus määrittelee, miten suora muuntaminen

tapahtuu. R2RML-suositus kuvaa muunnoskielen, jolla relaatiotietokantojen ja ontologioiden rakenteiden välisiä yhteyksiä voidaan määritellä. Vaikka suositukset ovat uusia, niitä noudattavia työkaluja on jo saatavilla.

Sekä suorassa että muunnoskieliin perustuvassa muuntamisessa lopputuloksena on semanttisen webin tietomallin mukainen esitys, jota semanttisen webin sovellukset voivat käyttää. Semanttisesti annotoitu tieto voidaan tallentaa erikseen, mutta ylläpidettävyyden kannalta parempi vaihtoehto on peittäminen. Peittämisratkaisussa relaatiotietokanta peitetään sovelluksella, joka toteuttaa muunnoksen ajonaikaisesti, kun sille esitetään semanttisen webin kyselyitä. Tällöin perinnejärjestelmiä ei tarvitse muuttaa, vaan ne voivat edelleen käsitellä ja ylläpitää tietoa. Semanttinen annotoinnin näkyvin hyöty on tällöin, että se mahdollistaa merkittävästi ilmaisuvoimaisempien kyselyjen esittämisen ja helpottaa tiedon yhdistämistä eri tietolähteistä.

Kiitokset

Ensimmäiseksi haluan kiittää ohjaajiani professori Juha Puustjärveä ja lehtori Harri Lainetta viisaista sanoista, asiantuntevista näkemyksistä, rakentavista kommenteista, joustavuudesta sekä avusta tutkielman aiheen määrittelyssä. Kiitokset Eero Mikkolalle ja Metsäntutkimuslaitokselle antoisista keskusteluista, tuotteliaisuutta edistävästä ilmapiiristä, tuesta ja kannustuksesta. Kimmo Sarevalle kiitos kannustuksesta ja huumorilla höystetystä ilmapiiristä toimistossamme. Hannalle ja Pullalle kiitokset kannustuksesta, tuesta, oikoluvusta, kärsivällisyydestä ja jaksamisesta. Lopuksi kiitokset kaikille niille, jotka ovat tukeneet ja kannustaneet työn teossa.

Lähteet

ABi08 Adida, B. ja Birbeck, M., toimittajat, *RDFa Primer. W3C Working Group Note 14-October-2008.* [Myös http://www.w3.org/TR/2008/NOTE-xhtml-rdfa-primer-20081014/, 20.4.2012].

AGu08 Angles, R. ja Gutierrez, C., The expressive power of SPARQL. *Proceedings of the 7 International Conference on the Semantic Web (ISWC 2008)*, Karlsruhe, Saksa, lokakuu 2008, sivut 114–129.

AHa04 Antoniou, G. ja van Harmelen, F., *A semantic web primer*. The MIT Press, Cambridge, Massachusetts, Yhdysvallat, 2004.

And94 Extracting an entity–relationship schema from a relational database through reverse engineering. *Proceedings of the 13th International Conference on the Entity–Relationship Approach (ER'94)*, Manchester, Britannia, joulukuu 1994, sivut 403–419.

Are12 Arenas, M. ja kumppanit, toimittajat, *A Direct Mapping of Relational Data to RDF. W3C Recommendation 27–September–2012.* [Myös http://www.w3.org/TR/2012/REC-rdb-direct-mapping-20120927/, 3.10.2012].

Arp01 Arpírez, J.C. ja kumppanit, WebODE: a scalable worbench for ontological engineering. *Proceedings of the 1st International Conference on Knowledge Capture (K–CAP 2001)*, lokakuu 2001, Victoria, British Columbia, Kanada, sivut 6–13.

Ast04 Astrova, I., Reverse engineering relational databases to ontologies. *Proceedings of the 1st European Semantic Web Symposium (ESWS 2004)*, Heraklion, Kreikka, toukokuu 2004, sivut 327–341.

Aue07 Auer, S. ja kumppanit. DBpedia: a nucleus for a web of open data. *Proceedings of the 6th International Semantic Web Conference*

(ISWC 2007) and 2nd Asian Semantic Web Conference (ASWC 2007), Busan, Korea, marraskuu 2007, sivut 722–735.

Aue09 Auer, S. ja kumppanit, Triplify – light–weight linked data publication from relational databases. *Proceedings of the 18th World Wide Web Conference (WWW2009)*, Madrid, Espanja, huhtikuu 2009, sivut 621–630.

Bao09 Bao, J. ja kumppanit, toimittajat, *OWL 2 Web Ontology Language. Document Overview. W3C Recommendation 27–October–2009.* [Myös http://www.w3.org/TR/2009/REC-owl2-overview-20091027/, 29.4.2012].

BBa01 Beckett, D. ja Barstow, A., *N–Triples. W3C RDF Core WG Internal Working Draft.* [Myös http://www.w3.org/2001/sw/RDFCore/ntriples/, 20.4.2012].

BBe11 Beckett, D. ja Berners–Lee, T., *Turtle – Terse RDF Triple Language. W3C Team Submission 28 March 2011.* [Myös http://www.w3.org/TeamSubmission/2011/SUBM-n3-20110328/, 20.4.2012].

BBr08 Beckett, D. ja Broekstra, J., toimittajat, *SPARQL query results XML format. W3C Recommendation 15–January–2008.* [Myös http://www.w3.org/TR/2008/REC-rdf-sparql-XMLres-20080115/, 15.5.2012].

BCG03 Barrasa, J., Corcho, O. ja Gomez–Perez A., FundFinder – a case study of database to ontology mapping. *Proceedings of the Semantic Integration Workshop (ISWC 2003)*, Sanibel Island, Florida, Yhdysvallat, lokakuu 2003.

BCG04 Barrasa, J., Corcho, O. ja Gomez–Perez, A., R2O, an extensible and semantically based database to ontology mapping language. *Proceedings of the Second Workshop on Semantic Web and Databases (SWDB2004)*, Toronto, Kanada, elokuu 2004, sivut 1069–1070.

BCo11 Berners–Lee, T. ja Connolly, D., *Turtle – Notation 3 (N3): A readable RDF syntax. W3C Team Submission 28 March 2011.* [Myös http://www.w3.org/TeamSubmission/2011/SUBM-turtle-20110328/, 20.4.2012].

BCy07 Bizer, C. ja Cyganiak, R., D2RQ – lessons learned. *Proceedings of the W3C Workshop on RDF Access to Relational Databases*, Boston, Yhdysvallat, lokakuu 2007.

BDD94 Bechara, A., Damasio, A. R., Damasio, H. ja Anderson, S. W., Insensitivity to future consequences following damage to human prefrontal cortex. *Cognition*, 50,1–3(1994), sivut 7–15.

BDD00 Bechara, A., Damasio, H. ja Damasio, A. R., Emotion, decision–making and the orbitofrontal cortex. *Cerebral Cortex*, 10,3(2000), sivut 295–307.

Bec01 Bechhofer, S. ja kumppanit, OILEd: a reasonable ontology editor for the semantic web. *Proceedings of the Joint German/Austrian Conference on AI: Advances in Artificial Intelligence (KI 2001)*, syyskuu 2001, Wien, Itävalta, sivut 396–408.

Bec04a Bechara, A., The role of emotion in decision–making: evidence from neurological patients with orbitofrontal damage. *Brain and Cognition*, 55(2004), sivut 30–40.

Bec04b Beckett, D., toimittaja, *RDF/XML Syntax Specification (Revised). W3C Recommendation 10 February 2004.* [Myös http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/, 1.10.2011].

Ben95 Bennett, K., Legacy systems: coping with success. *IEEE Software*, 12,1(1995), sivut 19–23.

Ber06 Berners–Lee, T., Design Issues: Linked Data, 2006. http://www.w3.org/DesignIssues/LinkedData.html. [17.2.2013]

Ber98a Berners–Lee, T., Cool URIs don't change, 1998. http://www.w3.org/Provider/Style/URI. [22.9.2011]

Ber98b Berners–Lee, T., What the semantic web can represent, 1998. http://www.w3.org/DesignIssues/RDFnot.html. [22.9.2011]

Ber98c Berners–Lee, T., Semantic web road map, 1998. http://www.w3.org/DesignIssues/Semantic.html. [2.5.2012]

Ber98d Berners–Lee, T., Relational databases and the semantic web, 1998. http://www.w3.org/DesignIssues/RDB-RDF.html. [17.5.2012]

Ber05 Berners–Lee, T. What HTTP URIs identify, 2005. http://www.w3.org/DesignIssues/HTTP-URI2.html. [27.2.2013]

BFM05 Berners–Lee, T., Fielding, R. ja Masinter, L., Uniform resource identifier (URI): Generic syntax, 2005. http://www.ietf.org/rfc/rfc3986.txt. [29.4.2012]

BGu04 Brickley, D. ja Guha, R. V., toimittajat, *RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation 10–February–2004.* [Myös http://www.w3.org/TR/2004/REC-rdf-schema-20040210, 2.10.2011].

BHL01 Berners–Lee, T., Hendler, J. ja Lassila, O., The semantic web. *Scientific American,* 284,5(2001), sivut 34–43.

BHS03 Baader, F., Horrocks, I. ja Sattler, U., Description logics as ontology languages for the semantic web. Teoksessa *Festschrift in honor of Jorg Siekmann, Lecture Notes in Artificial Intelligence*, Hutter, D. ja Stephan, W., toimittajat, Springer–Verlag, Saksa, 2003, sivut 228–248.

BHS09 Baader, F., Horrocks, I. ja Sattler, U., Description logics. Teoksessa *Handbook on Ontologies. Second Edition,* Staab, S. ja Studer, R., toimittajat, Springer–Verlag, Saksa, 2009, sivut 21–43.

Biz03 Bizer, C., D2R MAP – a DB to RDF mapping language. *Proceedings of the 12th International World Wide Web Conference*, Budapest, Unkari, toukokuu 2003.

Biz09 Bizer, C. ja kumppanit. DBpedia – a crystallization point for the web of data. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 7,3(2009), sivut 154–165.

BKD01 Broekstra, J., Klein, M., Decker, S., Fensel, D, van Harmelen, F. ja Horrocks, I., Enabling knowledge represenation on the web by extending RDF schema. *Proceedings of the tenth World Wide Web conference WWW'10*, Hong Kong, Kiina, 2001, sivut 467–478.

Bor97 Borst, W., Construction of engineering ontologies for knowledge sharing and reuse. Väitöskirja, Institute for Telematica and Information Technology, Twenten yliopisto, Hollanti, 1997.

BPS98 Bray, T., Paoli, J. ja Sperberg–McQueen, C. M., toimittajat, *Extensible Markup Language (XML) 1.0. W3C Recommendation 10–February–1998.* [Myös http://www.w3.org/TR/1998/REC-xml-19980210, 2.10.2011].

Bra07 Bratt, S., Semantic web, and other technologies to watch, 2007. http://www.w3.org/2007/Talks/0130-sb-W3CTechSemWeb/#%2824%29. [20.2.2013]

Bre10 Breslin, J. G. ja kumppanit, Semantic web computing in industry, *Computers in Industry*, 61,8(2010), sivut 729–741.

BSc09 Bizer, C. ja Schultz, A. The Berlin SPARQL benchmark. *International Journal on Semantic Web and Information Systems*, 5,2(2009), sivut 1–24.

BSe04 Bizer, C. ja Seaborne, A., D2RQ – treating non–RDF databases as virtual RDF graphs. *Poster at the 3rd International Semantic Web Conference (ISWC2004)*, Hiroshima, Japani, marraskuu 2004.

BSp81 Bancilhon, F. ja Spyratos, N., Update semantics of relational views. *ACM Transactions on Database Systems*, 6,4(1981), sivut 557–575.

Car07 Cardoso, J., The semantic web vision: where are we? *IEEE Intelligent systems*, 22,5(2007), sivut 84–88.

Cat97 Cathro, W., Metadata: an overview. *National Library of Australia Staff Papers*, 1997. http://www.nla.gov.au/openpublish/index.php /nlasp/article/view/1019/1289. [4.4.2013]

CBS96 Chiang, R. H. L., Barron, T. M. ja Storey, V. C., A framework for the design and evaluation of reverse engineering methods for relational databases. *Data & Knowledge Engineering*, 21,1(1996), sivut 57–77.

CGr08 Cuenca Grau, B. ja kumppanit, OWL 2: the next step for OWL. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6,4(2008), sivut 309–322.

Che76 Chen, P. P., The entity–relationship model – toward a unified view of data. *ACM Transactions on Database Systems*, 1,1(1976), sivut 9–36.

CHP09 Carroll, J., Herman, I. ja Patel–Schneider, P. F., toimittajat, *OWL 2 web ontology language RDF–based semantics. W3C Recommendation 27–October–2009.* [Myös http://www.w3.org/TR/2009/REC-owl2-rdf-based-semantics-20091027/, 10.5.2012].

Cod70 Codd, E. F., A relational model of data for large shared data banks. *Communications of the ACM*. 13,6(1970), sivut 377–387.

Cod72 Codd, E. F., Relational completeness of data base sublanguages. *Data Base Systems*, (1972), sivut 65–98.

Cro06 Crockford, D., The application/json media type for javascript object notation (JSON), 2006. http://www.ietf.org/rfc/rfc4627.txt. [17.2.2013]

DAR08 DARPA, First 50 years, 2008. http://www.darpa.mil/About/History/First_50_Years.aspx. [5.5.2012]

Dev02 Devedzic, V., Understanding ontological engineering. *Communications of the ACM.* 45,4(2002), sivut 136–144.

Die05 Diestel, R., *Graph theory. Third edition*. Springer–Verlag, Heidelberg, Saksa, 2005.

DMN09 De Nicola, A., Missikoff, M. ja Navigli, R., A software engineering approach to ontology building. *Information Systems*, 34,2(2009), sivut 258–275.

DOS03 Daconta, M., Obrst, L. ja Smith, K., *The semantic web: a guide to the future of XML, web services, and knowledge management*. Wiley Publishing, Inc., Indianapolis, Indiana, Yhdysvallat, 2003.

DSc04 Dean, M. ja Schreiber, G., toimittajat, *OWL Web Ontology Language Reference. W3C Recommendation 10 February 2004.* [Myös http://www.w3.org/TR/2004/REC-owl-ref-20040210/, 1.10.2011].

DSC12 Das, S., Sundara, S. ja Cyganiak, R., toimittajat, *R2RML: RDB to RDF Mapping Language. W3C Recommendation 27–September–2012.* [Myös http://www.w3.org/TR/2012/REC-r2rml-20120927/, 3.10.2012].

DSu05 Duerst, M. ja Suignard, M., Internationalized resource identifiers (IRIs), 2005. http://www.ietf.org/rfc/rfc3987.txt. [9.5.2012]

EMi07 Erling, O. ja Mikhailov, I., RDF support in the Virtuoso DBMS. *Proceedings of the 1st Conference on Social Semantic Web*, Leipzig. Saksa, syyskuu 2007, sivut 59–68.

Fah08 Fahad, M., ER2OWL: Generating OWL ontology from ER diagram. *Proceedings of the 5th IFIP International Conference on Intelligent Information Processing*, Peking, Kiina, lokakuu 2008, sivut 28–37.

Fen00 Fensel, D. ja kumppanit, On–To Knowledge: ontology–based tools for knowledge management. *Proceedings of the eBusiness and eWork 2000 (EMMSEC 2000) Conference*, Madrid, Espanja, lokakuu, 2000.

FFR97 Farquhar, A., Fikes, R. ja Rice, J., The Ontolingua server: a tool for collaborative ontology construction. *International Journal of Human–Computer Studies – Special issue: innovative applications of the World Wide Web,* 46, 6(1997), sivut 707–727.

FHH01 Fensel, D., van Harmelen, F., Horrocks, I., McGuinness, D. L. ja Patel–Schneider, P. F., OIL: an ontology infrastructure for the semantic web. *IEEE Intelligent Systems*, 16,2(2001), sivut 38–45.

Fie04 Fields, R. D., The other half of the brain. *Scientific American*, 290,4(2004), sivut 54–61.

FOA00 FOAF Project, 2000. http://www.foaf-project.org/original-intro. [26.2.2013]

FSH03 Fluit, C., Sabou, M. ja van Harmelen, F., Supporting user tasks throught visualization of lightweight ontologies. Teoksessa *Handbook on Ontologies*. Springer–Verlag, Karlsruhe, Saksa, 2003, sivut 415–434.

Gar04 Garshol, L. M., Metadata? thesauri? taxonomies? topic maps! making sense of it all. *Journal of Information Science*, 30,4(2004), sivut 378–391.

Gil08a Gill, T., Metadata and the web., Teoksessa *Introduction to Metadata. Second Edition,* Baca, M., toimittaja, The Getty Research Institute, Los Angeles, California, Yhdysvallat, 2008, sivut 20–37.

Gil08b Gilliland, A. J., Setting the stage., Teoksessa *Introduction to Metadata. Second Edition,* Baca, M., toimittaja, The Getty Research Institute, Los Angeles, California, Yhdysvallat, 2008, sivut 1–19.

GNi87 M. R. Genesereth, M. R. ja Nilsson, N. J., *Logical foundations of artificial intelligence*. Morgan Kaufmann, Los Altos, California, Yhdysvallat, 1987.

Goe08 Goer, E., SearchMonkey Support for RDFa enabled, 2008. http://developer.yahoo.com/blogs/ydn/posts/2008/09/searchmonkey_support_for_rdfa_enabled/. [29.4.2012]

GOS09 Guarino, N., Oberle, D. ja Staab, S., What is an ontology. Teoksessa *Handbook on Ontologies. Second Edition,* Staab, S. ja Studer, R., toimittajat, Springer–Verlag, Saksa, 2009, sivut 1–17.

GOu05 Graham, M. ja Oudshoorn, M. J., Reengineering software: a case study. *Proceedings of the Fifth SoMeT 2005*, Tokio, Japani, syyskuu 2005, sivut 18–32.

Fri06 Fries, T. P., A framework for transforming structured analysis and design artifacts to UML. *Proceedings of the 24th annual ACM international conference on design of communication (SIGDOC'06)*, Myrtle Beach, Etelä–Carolina, Yhdysvallat, lokakuu 2006, sivut 105–112.

GGO09 Gray, A. J., Gray, N. ja Ounis, I. Can RDB2RDF Tools Feasibily Expose Large Science Archives for Data Integration? *Proceedings of the 6th European Semantic Web Conference on The Semantic Web: Research and Applications (ESWC 2009)*, Heraklion, Kreikka, touko–kesäkuu 2009, sivut 491–505.

GPP12 Gearson, P., Passant, A. ja Polleres, A., toimittajat, SPARQL 1.1 update. *W3C Proposed Recommendation 08–November–2012.* [Myös http://www.w3.org/TR/2012/PR-sparql11-update-20121108/, 21.2.2013]

GPr09 Gangemi, A. ja Presutti, V., Ontology design patterns. Teoksessa *Handbook on Ontologies. Second Edition*, Staab, S. ja Studer, R., toimittajat, Springer–Verlag, Saksa, 2009, sivut 221–243.

GRa04 Guenther, R. ja Radebaugh, J., *Understanding metadata*. NISO Press, Bethesda, Maryland, Yhdysvallat, 2004.

Gru93 Gruber, T. R., A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5,2(1993), sivut 199–220.

Hal04 Halpin, H. The semantic web: the origins of artificial intelligence redux. *Proceedings of Third International Workshop on the History and Philosophy of Logic, Mathematics, and Computation (HPLMC 2004)*, Donostia San Sebastian, Espanja, 2004.

Hau12 Hausenblas, M., toimittaja, Implementations – RDB2RDF, 2012. http://www.w3.org/2001/sw/rdb2rdf/wiki/Implementations. [17.2.2012].

Hay00 Hayes, B., Graph theory in practice. *American Scientist*, 88,1(2000), sivu 9.

Hay04 Hayes, P., *RDF semantics. W3C Recommendation 10–February–2007.* [Myös http://www.w3.org/TR/rdf-mt/, 8.5.2012].

HBa97 Hartley, R. T. ja Barnden, J. A., Semantic networks: visualizations of knowledge. *Trends in Cognitive Sciences,* 1,5(1997), sivut 169–175.

HBS09 Hertel, A., Broekstra, J. ja Stuckenschmidt, H., RDF storage and retrieval systems. Teoksessa *Handbook on Ontologies. Second Edition,* Staab, S. ja Studer, R., toimittajat, Springer–Verlag, Saksa, 2009, sivut 489–508.

He07 He, B. ja kumppanit, Accessing the deep web. *Communications of the ACM*, 50,5(2007), sivut 94–101.

Hei95 Heiler, S., Semantic interoperability. *ACM Computing Surveys,* 27,2(1995), sivut 271–273.

HHa08 Hendler, J. ja van Harmelen, F., The semantic web: webisizing knowledge presentation. Teoksessa *Handbook of Knowledge*

Representation, van Harmelen, F., Lifschitz, V. ja Porter, B., toimittajat, Elsevier, Britannia, 2008, sivut 821–839.

Hit09 Hitzler, P. ja kumppanit, toimittajat, *OWL 2 Web Ontology Language Primer. W3C Recommendation 27–October–2009.* [Myös http://www.w3.org/TR/2009/REC-owl2-primer-20091027, 5.5.2012].

HKl10 Haslhofer B. ja Klas W., A survey of techniques for achieving metadata interoperability. *ACM Computing Surveys*, 42,2(2010), sivut 1–37.

HMc00 Hendler, J. ja McGuinnes, D. L., The DARPA Agent Markup Language. *IEEE Intelligent Systems.* 15,6(2000), sivut 67–73.

HMä06 Hyvönen, E. ja Mäkelä, E., Semantic autocompletion. *Proceedings of the 1st Asia Semantic Web Conference (ASWC 2006)*, Peking, Kiina, syyskuu 2006, sivut 4–9.

Hof00 Hofstadter, D. R., *Gödel, Escher, Bach: an eternal golden braid*. Penguin Books Ltd, Lontoo, Englanti, 2000.

Hor02 Horrocks, I., DAML+OIL: a description logic for the semantic web., *IEEE Bulletin of the Technical Committee on Data Engineering.* 25,1(2002), sivut 4–9.

Hor07 Horrocks, I. ja kumppanit, OWL: a description logic based ontology language for the semantic web. Teoksessa *The Description Logic Handbook. Second edition*, Baader, F. ja kumppanit, toimittajat, Cambridge University Press, Yhdysvallat, 2007, sivut 458–486.

HPH03 Horrocks, I., Patel–Schneider, P. F. ja van Harmelen, F. From SHIQ and RDF to OWL: the making of a web ontology language. *Journal of Web Semantics*, 1,1(2003), sivut 7–26.

HRG10 Hert, M., Reif, G. ja Gall, H. C., Updating relational data via SPARQL/Update. *Proceedings of the 2010 EDBT/ICDT Workshops*, Lausanne, Sveitsi, maaliskuu 2010, sivut 24:1–24:8.

HRG11 Hert, M., Reif, G. ja Gall, H. C., A comparison of rdb–to–rdf mapping languages. *Proceedings of the 7th International Conference on Semantic Systems*, Graz, Itävalta, syyskuu 2011, sivut 25–32.

HSt02 Handschuh, S. ja Staab, S., Authoring and annotation of web pages in CREAM. Proceedings of the 11th International World Wide Web Conference (WWW2002), toukokuu 2002, Honolulu, Hawaiji, Yhdysvallat, sivut 462–473.

Hul01 Hulgeri, A. ja kumppanit, Keyword search in databases. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 24,3(2001), sivut 22–32.

HVT08 Hyvönen, E., Viljanen, K., Tuominen, J. ja Seppälä, K., Building a national semantic web ontology and ontology service infrastructure – the FinnONTO approach. *Proceedings of the 5th European Semantic Web Conference (ESWC'08)*, Teneriffa, Espanja, kesäkuu 2008, sivut 95–109.

Hyv08 Hyvönen, E. ja kumppanit, CultureSampo – a national publication system of cultural heritage on the semantic web 2.0. *Proceedings of the 6th European Semantic Web Conference (ESWC 2009)*, Heraklion, Kreikka, touko–kesäkuu 2009, sivut 851–856.

HZo07 Hawking, D. ja Zobel, J., Does topic metadata help with web search? *Journal of the American Society for Information Science and Technology,* 58,5(2007), sivut 613–628.

Jim08a Jiménez–Ruiz, E ja kumppanit, Ontology integration using mappings: towards getting the right logical consequences. *Proceedings of the 6th European Semantic Web Conference (ESWC 2009)*, touko–kesäkuu 2009, Heraklion, Kreikka, sivut 173–187.

Jim08b Jiménez–Ruiz, E ja kumppanit, Safe and economic re–use of ontologies: a logic–based methodology and tool support. *Proceedings*

of the 5th European Semantic Web Conference (ESWC 2008), kesäkuu 2008, Teneriffa, Espanja, sivut 185–199.

Jim11 Jiménez–Ruiz, E ja kumppanit, Supporting concurrent ontology development: framework algorithms and tool. *Data & Knowledge Engineering*, 70,1(2011), sivut 146–164.

Kar02 Karlsson, F., *Yleinen kielitiede.* Yliopistopaino, Helsinki, 2002.

KCa04 Klyne, G. ja Carroll, J. J., toimittajat, *Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation 10 February 2004.* [Myös http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/, 1.10.2011].

Kir04 Kiryakov, A. ja kumppanit, Semantic annotation, indexing and retrieval. *Journal of Web Semantics.* 2,1(2004), sivut 49–79.

KMu10 Khondoker, R. M. ja Mueller, P., Comparing ontology development tools based on an online survey. *Proceedings of the World Congress on Engineering (WCE 2010)*, Lontoo, Britannia, kesä–heinäkuu 2010.

Kos04 Koskenniemi, K., Terms and concepts of language technology, 2004. http://www.ling.helsinki.fi/kit/2004s/terms-en.shtml. [1.10.2011]

KPH05 Kalyanpur, A., Parsia, B. ja Hendler, J., A tool for working with web ontologies. *International Journal on Semantic Web and Information Systems*, 1,1(2005).

Krö10 Krötzsch, M., Description logics rules. Väitöskirja, Karlsruhe Institute of Technology, Saksa, 2010.

Krö12 Krötzsch, M., OWL 2 profiles: an introduction to lightweight ontology languages. *Proceedings of the 8th Reasoning Web Summer School (RW 2012)*, syyskuu 2012, Wien, Itävalta, sivut 112–183.

KSK11 Kremen, P., Smid, M. ja Kouba, Z., OWLDiff: a practical tool for comparison and merge of OWL ontologies. *Proceedings of the 22nd International Workshop on Database and Expert System*

Applications (DEXA 2011), elo–syyskuu 2011, Toulouse, Ranska, sivut 229–233.

Lai97 Laine, H., toimittaja, Relaatiotietokantasanasto, 1997. http://www.cs.helsinki.fi/relaatiosanasto/. [17.5.2012]

Lam09 van Lamsweerde, A., *Requirements engineering*. Wiley Publishing, Inc., England, 2009.

LCo05 de Laborda, C. P. ja Conrad, S., Relational.OWL – a data and schema representation format based on OWL. *Proceedings of the 2nd Asia–Pacific Conference on Conceptual Modelling*, Newcastle, New South Wales, Australia, tammi–helmikuu 2005, sivut 89–96.

LNo03 Liebig, T. ja Noppens, O., OntoTrack: fast browsing and easy editing of large ontologies. *Proceedings of the 2nd International Workshop on Evaluation of Ontology based Tools (EON 2003)*, lokakuu 2003, Sanibel Island, Florida, Yhdysvallat, sivut 47–56.

LSw99 Lassila, O. ja Swick, R. R., toimittajat, *Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation 22 February 1999.* [Myös http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/, 9.10.2011].

Man07 Mangold, C., A survey and classification of semantic search approaches. *International Journal of Metadata, Semantics and Ontology*, 2,1(2007), sivut 23–34.

Mar98 Marshall, C. C., Toward an ecology of hypertext annotation. *Proceedings of the ninth ACM conference on Hypertext and hypermedia*, Pittsburgh, Pennsylvania, Yhdysvallat, 1998, sivut 40–49.

Mar12 Marking up your site: Overview, 2012. http://onlinehelp.microsoft.com/en-us/bing/hh207238.aspx. [29.4.2012]

MBe09 Miles, A. ja Bechhofer, S., toimittajat, *SKOS Simple Knowledge Organization System, W3C Recommendation 18–August–2009.* [Myös http://www.w3.org/TR/2009/REC-skos-reference-20090818/, 7.5.2012]

McC55 McCarthy, J. ja kumppanit, A proposal for the Dartmouth summer research project on artificial intelligence, August 31, 1955. *AI Magazine*, 27,4(2006), sivut 12–14.

MHS05 Mäkelä, E., Hyvönen, E. ja Sidoroff, T., View–based user interfaces for information retrieval on the semantic web. *Proceedings of the ISWC–2005 Workshop End User Semantic Web Interaction*, Galway, Irlanti, marraskuu 2005.

Mil01 Miller, G. G., The characteristics of agile software processes. *Proceedings of the 39th International Conference and Exhibition on Technology of Object–Oriented Languages and Systems (TOOLS39)*, heinä–elokuu 2001, Santa Barbara, Kalifornia, Yhdysvallat, sivut 385–387.

MKo83 Michalski, R. S. ja Kodratoff, Y., Machine learning: recent progress, classification of methods, and future directions. Teoksessa *Machine Learning. An Artificial Intelligence Approach. Volume III*, Kodratoff, Y. ja Michalski, R., Morgan Kaufmann Publishers, Inc., Yhdysvallat, 1983, sivut 3–30.

MMi04 Manola, F. ja Miller, E., toimittajat, *RDF Primer. W3C Recommendation 10 February 2004.* [Myös http://www.w3.org/TR/2004/REC-rdf-primer-20040210/, 20.4.2012].

Mos11 Mosley, M., toimittaja, Into the mind: emotions. Tiededokumentti, British Broadcasting Corporation (BBC), Britannia, 2011.

Mot09 Motik, B. ja kumppanit, toimittajat, *OWL 2 web ontology language profiles. W3C Recommendation 27–October–2009.* [Myös http://www.w3.org/TR/2009/REC-owl2-profiles-20091027, 5.5.2012].

MPC09 Motik, P., Patel–Schneider, P. F. ja Cuenca Grau, B., toimittajat, *OWL 2 web ontology language direct semantics. W3C Recommendation 27–October–2009.* [Myös http://www.w3.org/TR/2009/REC-owl2-direct-semantics-20091027/, 10.5.2012].

MPP09a Motik, B., Patel–Schneider, P. F. ja Parsia, B., toimittajat, *OWL 2 web ontology language structural specification and functional–style syntax. W3C Recommendation 27–October–2009.* [Myös http://www.w3.org/TR/2009/REC-owl2-syntax-20091027, 5.5.2012].

MPP09b Motik, B., Parsia, B. ja Patel–Schneider, P. F., toimittajat, *OWL 2 Web Ontology Language XML serialization. W3C Recommendation 22–September–2009.* [Myös http://www.w3.org/TR/2009/PR-owl2-xml-serialization-20090922/, 8.5.2012]

MVo00 Masermann, U. ja Vossen, G., Design and implementation of a novel approach to keyword searching in relational databases. *Proceedings of the East–European Conference on Advances in Databases and Information Systems*, Praha, Tšekki, syyskuu 2000, sivut 171–184.

NBr03 Nardi, D. ja Brachman, R. J., An introduction to description logics. Teoksessa *The Description Logic Handbook,* Baader, F. ja kumppanit, toimittajat, Cambridge University Press, Yhdysvallat, 2003, sivut 1–40.

OCo05 O'Connor, M. ja kumppanit, Supporting rule system interoperability on the semantic web with SWRL. *Proceedings of the 4th International Semantic Web Conference, ISWC 2005*, Galway, Irlanti, marraskuu 2005, sivut 974–986.

Ont11 Ontology service ONKI, 2011. http://onki.fi. [24.2.2013]

PAG09 Perez, J., Arenas, M. ja Gutierrez, C. Semantics and complexity of SPARQL. *ACM Transactions on Database Systems*, 34,3(2009), sivut 1–45.

Pan09 Pan, J. Z., Resource description framework. Teoksessa *Handbook on Ontologies. Second Edition,* Staab, S. ja Studer, R., toimittajat, Springer–Verlag, Saksa, 2009, sivut 71–90.

Par94 Parnas, D. L., Software aging. *Proceedings of the 16th International Conference on Software Engineering,* Sorrento, Italia, toukokuu 1994, sivut 279–287.

PDh07 Pirzada, S. ja Dharwadker, A., Applications of graph theory. *Journal of the Korean Society for Industrial and Applied Mathematics (KSIAM)*, 11,4(2007), sivut 19–38.

Pel11 Pellet: OWL 2 reasoner for Java, 2011. http://clarkparsia.com/pellet. [26.2.2013]

Pro02 The Protégé project, 2002. http://protege.stanford.edu. [19.2.2013]

PSe08 Prud'hommeaux, E. ja Seaborne, A., toimittajat, *SPARQL Query Language for RDF. W3C Recommendation 15–January–2008*. [Myös http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/, 1.10.2011].

PSt98 Pooley, R. ja Stevens, P., Systems reengineering patterns. *Proceedings of the ACM Sigsoft 6th International Symposium on the Foundations of Software Engineering*, Lake Buena Vista, Yhdysvallat, marraskuu 1998, sivut 17–23.

PVi12 Priyatna, F. ja Villazón–Terrazas, B. Building ontologies by using re–engineering patterns and R2RML mappings. *Proceedings of the 3rd Workshop on Ontology Patterns (WOP 2012) (tulossa)*, Boston, Yhdysvallat, marraskuu 2012.

Ren10 Renteria–Agualimpia, W. ja kumppanit, Exploring the advances in semantic search engines. *Proceedings of the International Symposium on Distributed Computing and Artificial Intelligence (DCAI–2010)*, Valencia, Espanja, syyskuu 2010, sivut 613–620.

RNo03 Russell, S. J. ja Norvig, P., *Artificial intelligence. A modern approach. Second edition.* Pearson Education, Yhdysvallat, 2003.

Sah09 Sahoo, S. ja kumppanit, *A Survey of Current Approaches for Mapping of Relational Databases to RDF. W3C RDB2RDF Incubator Group January–08–2009.* [Myös http://www.w3.org/2005/Incubator/rdb2rdf/RDB2RDF_SurveyReport.pdf 15.9.2012].

SBF98 Studer, R., Benjamins, V. R. ja Fensel, D. Knowledge engineering: principles and methods. *Data & Knowledge Engineering, 25,1–2(1998), sivut 161–197.*

SCy08 Sauermann, L. ja Cyganiak, R., toimittajat, *Cool URIs for the Semantic Web. W3C Interest Group Note 03 December 2008.* [Myös http://www.w3.org/TR/2008/NOTE-cooluris-20081203/, 1.10.2011].

Sea08 Seaborne, A. ja kumppanit, *SPARQL update. A language for updating RDF graphs. W3C Member Submission 15–July–2008.* [Myös http://www.w3.org/Submission/2008/SUBM-SPARQL-Update-20080715/, 17.5.2012].

Sel07 Seleng, M. ja kumppanit, RDB2Onto: approach for creating semantic metadata from relational database data. *Proceedings of the Ninth International Conference on Informatics*, Bratislava, Slovakia, kesäkuu 2007, sivut 113–116.

Seq11 Sequeda, J. F. ja kumppanit, Survey of directly mapping SQL databases to the semantic web. *The Knowledge Engineering Review*, 26,4(2011), sivut 445–486.

Sha05 Shafranovich, Y., Common Format and MIME Type for Comma–Separated Values (CSV) Files, 2005. http://tools.ietf.org/rfc/rfc4180.txt. [17.2.2013]

Sim96 Simon, H. A., *The shape of automation for men and management.* Harper & Row, New York, Yhdysvallat, 1965.

SMi12 Sequeda, J. F. ja Miranker, D. P., *Ultrawrap: SPARQL execution on relational data. Technical Report*. [Myös http://apps.cs.utexas.edu/tech_reports/reports/tr/TR-2078.pdf, 9 10.2012].

SPA12 SPARQL explorer for http://dbpedia.org/sparql, 2012. http://dbpedia.org/snorql/. [1.10.2012]

SPV12 Sequeda, J. F., Priyatna, F. ja Villazón–Terrazas, B., Relational database to RDF mapping patterns. *Proceedings of the 3rd Workshop on Ontology Patterns (WOP 2012) (tulossa)*, Boston, Yhdysvallat, marraskuu 2012.

Squ11 SquirrelRDF, 2011. http://jena.sourceforge.net/SquirrelRDF/. [17.2.2013]

STH10 Steiner, T., Troncy, R. ja Hausenblas, M., How Google is using linked data today and vision for tomorrow. *Proceedings of Linked Data in the Future Internet at the Future Internet Assembly,* Gent, Belgia, joulukuu 2010.

STM07 Sequeda, J.F., Tirmizi, S. H. ja Miranker, D. P., SQL databases are a moving target. *Position Paper for W3C Workshop on RDF Access to Relational Databases*, Cambridge, Massachusetts, Yhdysvallat, lokakuu 2007.

SVä02 Salminen, H. ja Väänänen, J., *Johdatus logiikkaan*. Gummerus Kirjapaino Oy, Saarijärvi, Suomi, 2002.

Top13 Top Braid Composer, 2013. http://www.topquadrant.com/products/TB_Composer.html. [26.2.2013]

TSK06 Terminologian sanasto. Sanastokeskus TSK ry, Helsinki, 2006.

TSM08 Tirmizi, S. H., Sequeda, J. F. ja Miranker, D., Translating SQL applications to the semantic web. *Proceedings of the 19th International Conference on Database and Expert Systems*

Applications (DEXA 2008), Torino, Italia, syyskuu 2008, sivut 450–464.

TYF86 Teorey, T. J., Yang, D. ja Fry, J. P., A logical design methodology for relational databases using the extended entity–relationship model. *Computing Surveys*, 18,2(1986), sivut 197–222.

VHa12 Villazón–Terrazas, B. ja Hausenblas, M., toimittajat, *RDB2RDF Implementation Report. W3C Editor's Draft 14–August–2012*. [Myös http://www.w3.org/2001/sw/rdb2rdf/implementation-report/, 9.10.2012].

Vil12 Villazón–Terrazas, B., A method for reusing and re–engineering non–ontological resources for building ontologies. Väitöskirja, Department of Artificial Intelligence, Technical University of Madrid, Espanja, 2011.

Voo94 Voorhees, E. M., Query expansion using lexical–semantic relations. *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '94)*, Dublin, Irlanti, heinäkuu 1994, sivut 61–69.

Vra09 Vrandecic, D., Ontology evaluation. Teoksessa *Handbook on Ontologies. Second Edition,* Staab, S. ja Studer, R., toimittajat, Springer–Verlag, Saksa, 2009, sivut 293–313.

Vys10 Vysniauskas, E. ja kumppanit, Enhancing connection between ontologies and databases with OWL 2 concepts and SPARQL. *Proceedings of the 16th International Conference on Information and Software Technologies (IT 2010).* Kaunas, Liettua, huhtikuu 2010, sivut 350–357.

W3C10 W3C semantic web standards, 2010. http://www.w3.org/standards/semanticweb/. [14.9.2011]

W3C11 About W3C, 2011. http://www.w3.org/Consortium/. [14.9.2011]

War10 Warwick, K. ja kumppanit, Controlling a mobile robot with a biological brain. *Defence Science Journal*, 60, 1(2010), sivut 5–14.

Wik12 Wikipedia, 2012. http://www.wikipedia.org/. [1.10.2012]

Yin04 Yin, X. ja kumppanit, CrossMine: efficient classification across multiple database relations. *Proceedings of the 20th International Conference on Data Engineering (ICDE 2004)*, Boston, Massachusetts, Yhdysvallat, maalis–huhtikuu 2004, sivut 399–410.

Zho10 Zhou, S. ja kumppanit, Ontology generator from relational database based on Jena. *Computer and Information Science*, 3,2(2010), sivut 263–267.