

Semanttinen web ja RDF-kuvauskieli

Joni Airaksinen

Sähkötekniikan korkeakoulu

Kandidaatintyö

Espoo 24.4.2018

Vastuuopettaja

TkT Pekka Forsman

Työn ohjaaja

TkL Pekka Aarnio



Copyright © 2018 Joni Airaksinen



Tekijä Joni Airaksinen

Työn nimi Semanttinen web ja RDF-kuvauskieli

Koulutusohjelma Sähkötekniikan kandidaattiohjelma

Pääaine Automaatio- ja systeemiteknikka

Vastuuopettaja TkT Pekka Forsman

Työn ohjaaja TkL Pekka Aarnio

Päivämäärä 24.4.2018

Sivumäärä 34+6

Kieli Suomi

Tiivistelmä

Semanttinen web on internetiä tehokkaampi tiedon hyödyntämisympäristö. Informaatio pitää kuitenkin ensin saattaa tietokoneymärrettävään muotoon, jotta se kelpaa semanttiseen webiin. Tutkielmassa tutustutaan semanttisen webin standardeihin, joiden avulla voidaan luoda tietokoneluettavaa ja -päätteltyä tie-toa. Keskeisiä standardeja ovat muun muassa: RDF, RDFS ja OWL. Erikoisten standardien tarpeellisuus pyritään perustelemaan niiden käyttökohteiden avulla. Standardien lisäksi tutkielmassa pohditaan semanttisen webin sovelluskohteita, haasteita ja nykytilaa. Työ pyrkii tarjoamaan kattavan yleiskuvan semanttisen webin pääkohdista. Tutkielma sisältää paljon esimerkkejä, jotta lukijan on helpo omaksua uutta tietoa. Lukijan ei siis oleteta tuntevan aihetta entuudestaan. Ohjelointitaidoista voi olla apua esimerkkejä seuratessa.

Avainsanat Semanttinen web, semantiikka, RDF, OWL, SPARQL

Sisällysluettelo

Tiivistelmä	3
Sisällysluettelo	4
Lyhenteet	5
1 Johdanto	6
2 Semanttinen web yleisesti	7
3 Semanttisen webin rakenne	8
3.1 Resursseihin viitataan URI:eilla	8
3.2 XML-merkintäkieli	9
3.3 Nimiavaruudet	9
3.4 RDF-tietomalli	10
3.4.1 RDF-toteamuksit	10
3.4.2 RDF/XML-syntaksi	11
3.4.3 Turtle-syntaksi	12
3.5 Semanttinen tieto	13
3.6 RDF Schema	14
3.7 Web Ontology Language	16
3.8 Ontologiat ja sanastot	17
3.9 RDF-muotoisen tiedon tallentaminen	18
3.9.1 Triplestore	19
3.9.2 SPARQL-kyselykieli	20
3.10 Linkitetty tieto	20
4 Käytännön sovelluksia	22
4.1 XMP-metadatiformaatti	22
4.2 Semanttinen haku	22
4.3 Semanttinen Finlex	23
5 Semanttisen webin nykytila ja tulevaisuus	25
5.1 Linkitetyn tiedon laadukkuus	25
5.2 Haasteita	25
5.2.1 Ihmiskielen monimuotoisuus	26
5.2.2 Semanttisen tiedon ja ihmisten luottavuus	26
5.2.3 Alan heikko tietämys	26
5.3 Semanttisen tiedon yleistyminen	26
5.4 Semanttisen webin arkkitehtuuri	28
6 Yhteenveto	29
Liite A Esimerkki tietokone ontologiasta	35
Liite B Dигиталіківн XMP-metadatа	39

Lyhenteet

CPU	Central Processing Unit
ELI	European Legislation Identifier
GPU	Graphics Processing Unit
HDD	Hard Disk Drive
HTTP	Hypertext Transfer Protocol
OWL	Web Ontology Language
PSU	Power Supply Unit
RAM	Random Access Memory
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SFL	Semantic Finlex Legislation ontology
SPARQL	SPARQL Protocol and RDF Query Language
SSD	Solid State Drive
Turtle	Terse RDF Triple Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WWW	World Wide Web
W3C	World Wide Web Consortium
XML	Extensible Markup Language
XMP	Extensible Metadata Platform

1 Johdanto

Internet on luotu tehokasta tiedon jakamista varten. Kehityksen alkuvaiheilla interneitin avulla jaettiin tutkimustuloksia ja muita tärkeitä tekstdokumentteja. Dokumentit luotiin ihmisten luettaviksi, mutta tietokoneet pystyivät seulomaan sanoja niistä. Nämä tietokoneet kykenivät hylkäämään merkinnältään virheelliset dokumentit, mutta dokumenteissa esitettyä tietoa ne eivät ymmärtäneet [1]. Tim Berners-Lee, yksi internetin keskeisimmistä kehittäjistä, halusi hyödyntää tietokoneita tiedon jakamisessa aiempaa enemmän. Hän jatkoi internetin kehitystä luoden *semanttisen webin* (engl. Semantic Web) ideologian. Semanttista webiä luonnehditaan internetin jatkeeksi [2].

Semanttisella webillä viitataan internettiin, jonka sisältöä kykenevät lukemaan ja ymmärtämään myös tietokoneet [2]. Tiedon ymmärtämisen ansiosta tietokoneet pystyvät tekemään itsenäisiä päätelmiä esitetyn tiedon pohjalta ja näin ne kykenevät auttamaan ihmisiä. Tiedon hyödyntäminen tehostuu, koska osa ihmisten tekemästä työstä voidaan siirtää tietokoneiden tehtäväksi. Tieto pitää kuitenkin esittää tietokoneille sopivassa muodossa standardoitujen semanttisen webin tekniikkoiden avulla, jotta tietokoneet voivat ymmärtää esittämäänsä tietoa.

Semanttisen webin yhteydessä tiedon esittämiseen käytetään RDF-tietomallia (Resource Description Framework). Tietomallin ansiosta tietokoneet voivat lukea ja ymmärtää tiedonvälisiä suhteita [3]. Tiedon merkityksen ilmaisemiseen käytetään RDF Schemaa (Resource Description Framework Schema). Tiedon suhteiden ja merkityksen kuvailu mahdollistaa tiedon ymmärtämisen ja päättelykyvyn. RDFS:n lisäksi tietoa voidaan kuvalla Web Ontology Languagen (OWL) avulla [4]. Tietoa halutaan useimmiten myös tallentaa ja hakea tehokkaasti. RDF:n avulla esitettyä tietoa tallennetaan usein niin sanottuihin triplestore tietokantoihin ja haetaan SPARQL-kyselykielen (SPARQL Protocol and RDF Query Language) avulla [5].

Tutkielmana tarkastellaan edellä mainittuja semanttisen webin keskeisimpia teknikoita. Tavoitteena on löytää syitä edellisten teknikoiden tarpeellisuudelle. Tekniikat käydään yksitellen lävitse ja niiden pääkohdat pyritään tiivistämään helposti omaksuttavaan muotoon. Työ ei sisällä kaikkea semanttiseen webiin tai sen tekniikkoihin liittyvää tietoa, vaan ainoastaan keskeisimman tiedon. Tutkielman lukemisen jälkeen lukijan tulisi pystyä hahmottamaan semanttisen webin rakenne, merkitys ja laajuus. Työ on toteutettu kirjallisesti tutkimuksena.

Tutkielman toisessa luvussa käsitellään semanttista webiä yleisellä tasolla. Kolmannessa luvussa syvennytään semanttisen webin toteutukseen erilaisten tekniikkoiden johdattelemana. Neljännessä luvussa esitellään joitakin semanttisen webin mahdollisia sovelluksia. Viidennessä luvussa pohditaan lyhyesti semanttisen webin nykytilaa, haasteita ja suosiota. Lopuksi tutkielman esitelty semanttista webiä koskeva keskeisin tieto kootaan yhteen.

2 Semanttinen web yleisesti

Semanttinen web ei vastaa miinkään yksittäiseen ongelmaan, vaan se mahdollistaa uudenlaisten sovellusten kehittämisen ja tiedon entistä tehokkaamman levittämisen ja hyödyntämisen. Tim Berners-Lee visioi tietokoneiden soveltuvan muuhunkin kuin pelkän tiedon esittämiseen. Hän halusi tehdä tietokoneista ihmisten apureita sen sijaan, että ne olisivat ainoastaan työkaluja tiedon esittämiseen [2].

Internet on valtava tietolähde, mutta tietokoneet eivät voi hyödyntää kaikkea internetissä esiintyvää tietoa. Tietokoneet eivät ymmärrä tiedon välisiä suhteita suoraan tekstistä kuten ihmiset. Sanat ja lauseet eivät itsessään tarkoita tietokoneelle mitään, sillä ne koostuvat ainoastaan biteistä ilman minkäänlaista selitystä merkityksestä [6]. Siten tietokone ei voi hahmottaa tiedon välisiä asiayhteyksiä ja lauseiden merkityksiä suoraan tekstistä. Tietokone ei esimerkiksi ymmärrä synonyymien terveyskeskus ja terveysasema tarkoittavan samaa asiaa ilman, että kyseinen tieto ilmoitetaan tietokoneelle jollain erillisellä tavalla. Tarvitaan tietoa kuvailevaa tietoa eli *metatietoa*. Tietokoneen ymmärtämää metatietoa voidaan luoda hyödyntämällä semanttisen webin tekniikoita.

Antoniou et al. tiivistää semanttisen webin menetelmien suunnitteluperiaatteeen kolmeen pääkohtaan [7].

1. Informaation pitää olla saatavilla standardoidussa muodossa.
2. Tietoelementtien ja niiden keskinäisten kuvausten täytyy olla tietokoneiden luettavissa.
3. Tiedon keskinäiset suhteet täytyy esittää tavalla, joka sallii tietokoneen prosessoida suhteita.

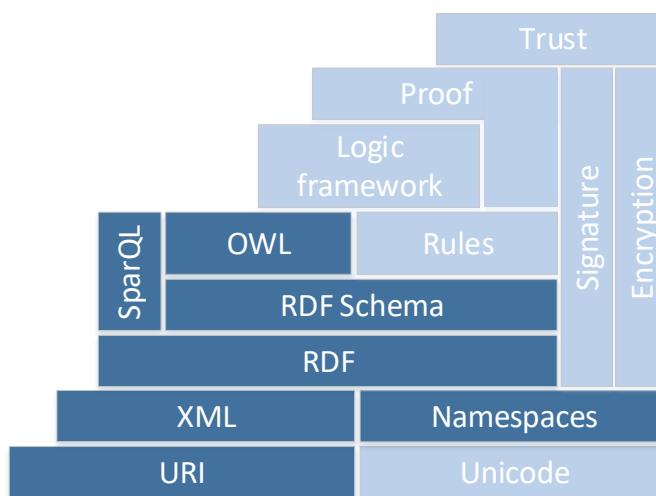
Edellä mainittujen periaatteiden täyttyä tietokoneet voivat hyödyntää niille esitettyä tietoa tehokkaasti. Semanttisen webin teknikat toteuttavat edelliset suunnitteluperiaatteet, joten niiden käyttö mahdollistaa tiedon tehokkaan hyödyntämisen.

Kansainvälinen World Wide Web Consortium (W3C) on ottanut tehtäväkseen kehittää ja ylläpitää yhteisiä internettiin liittyviä standardeja. Järjestö vastaa semanttisen webin standardeista ja standardeihin voi halutessaan tutustua heidän verkkosivuiltaan. Järjestön johtajana toimii Tim Berners-Lee, joka luonnehtii semanttista webiä internettiin jatkeeksi [8] [2]. Berners-Lee on yksi internettiin ja semanttisen webin pääkehittäjistä.

Semanttisen webin tehokkuuden kannalta on tärkeää, että standardimuodossa olevaa tietoa on paljon tarjolla. Siksi on oleellista, että jokin järjestö ylläpitää yhteisiä standardeja, joiden avulla tietoa voidaan esittää yhteisesti ymmärrettävässä muodossa. Standardoidun esitysmuodon ansioista oman tiedon lisäksi voidaan hyödyntää myös muiden esittämää tietoa. Tiedonkäyttö tehostuu, kun samaa tietoa ei tarvitse kirjoittaa useita kertoja, sillä yksittäistä avointa tietoresurssia voi hyödyntää kaikki tietoa tarvitsevat. Tietokoneluettavan tiedon yhdistämisen ideaan viitataan termillä *linkitettyn tieto* (engl. linked data) [9] [10].

3 Semanttisen webin rakenne

Semanttinen web rakentuu useista osista. Jokainen osa vastaa tiettyihin haasteisiin, jolloin osasia yhdistämällä voidaan luoda laadukkaita sovelluksia. Modulaarinen jaotteluun perustuva rakenne on tehokas ja sitä hyödynnetään myös internetin kehityksessä (HTML, CSS ja JavaScript). Vanhentuneiden ratkaisujen korvaaminen ja päivittäminen on helpompaa, koska kehittäjältä ei vaadita koko laajan alan tarkkaa tuntemusta. Riittää, että kehittäjät hallitsevat pienempiä kokonaisuuksia hyvin. Modulaarisuus edistää myös yksilöllisten sovellusten kehitystä, sillä sovelluksissa tarvitsee hyödyntää ainostaan tarpeellisia komponentteja kaikkien sijaan. Seuraava pinomalli (kuva 1) auttaa laadukkaan sovelluksen suunnittelussa [11].



Kuva 1: Vuonna 2005 esitetty arkkitehtuuri. Lähde: [11].

Mallista on himmennetty osa-alueet, joita ei käsitellä tässä työssä. Unicoden oletetaan siis olevan lukijalle jokseenkin tuttu käsite, mutta muuten kaikki perusteet käydään lävitse. Pinon pohjalla olevien perusteiden ymmärtäminen on tärkeää, jotta voi löytää syitä korkeampitason teknikoiden tarpeellisuudelle. Seuraavissa alalvuissa tutustutaan siis pinossa esitelttyihin teknikoihin edeten pinossa alhaalta ylöspäin.

3.1 Resursseihin viitataan URI:eilla

Semanttisessa webissä tietoon viitataan aina Uniform Resource Identifierin eli URI:n avulla [3]. Tämän ansiosta kaikki viittaukset tiedonkohteisiin eli resursseihin ovat yksiselitteisiä, jolloin väärinymärrysten vaaraa ei ole. URI:eja ovat esimerkiksi URL (Uniform Resource Locator), ISBN, sähköpostiosoite ja puhelinnumero. Usein kuitenkin suositaan HTTP-protokollan mukaisia URI:eja (eli käytännössä URL:eja),

jotta käyttäjä voi helposti tarkistaa mihin tunnisteella viitataan [12]. Resurssit voivat olla mitä tahansa semanttisen webin standardeilla määritettyä tietoa, kuten esimerkiksi tekstidokumentteja tai fyysisiä laitteita [3].

3.2 XML-merkintäkieli

XML (eXtensible Markup Language) on W3C:n standardoima merkintäkieli, jonka avulla voidaan tallentaa- ja siirtää tietoa. XML-merkintäkieli on suunniteltu ihmisen ja tietokoneen luettavaksi. Merkinnältään XML muistuttaa hyvin paljon HTML-merkintäkieltä (Hypertext Markup Language). XML kuitenkin eroaa HTML:stä käyttötarkoituksestaan. XML on suunniteltu mielivaltaisen tiedon tallennuskieleksi, kun taas HTML:llä pystytään esittämään ainoastaan ennalta määriteltyjä websivuja [13]. XML mahdollistaa tiedon välittämisen yksittäisessä standardoidussa muodossa. Tämän ansiosta sovellukset voivat ymmärtää toisiaan, mikäli ne tukevat XML-standardia. Alla esimerkki XML-syntaksista:

```

1 <computer>
2   <cpu>i5-4200m</cpu>
3 </computer>
```

Koodifragmentti 1: XML-syntaksiesimerkki.

Semanttisen webin näkökulmasta XML-merkintäkieli on ongelmallinen. Edellisestä esimerkistä voidaan havaita, että prosessori on sijoitettu XML-puussa tietokone-elementin sisälle, mutta muuten merkintä on melko epämääräinen. Pelkän merkinnän pohjalta ei ole mahdollista tietää mihin tietokoneeseen viitataan, sekä prosessorin ja tietokoneen suhdetta ei voida tuntea. Ihminen tekee intuitiivisen päätelmän, että tietokone varmaankin käyttää prosessoria toimiakseen, mutta näin ei välittämättä ole. XML-merkintä ei siis sisällä riittävästi tietoa, jotta se voisi toimia semanttisen webin merkintäkielenä [13]. XML on kuitenkin ollut lähtökohta sopivan syntaksin kehityksessä, mikä tekee siitä merkittävän merkintäkielen semanttisen webin näkökulmasta.

3.3 Nimiavaruudet

Etuliite eli prefiksi (engl. prefix) määräää käytettävän nimiavaruuden (engl. namespace). Prefiksien käyttämisestä seuraa useita hyötyjä. Dokumenteista tulee siistimpää ja helpompia lukea, koska tekstiriveistä tulee lyhyempiä. Pitkiä URI:eja ei tarvitse toistaa jokaisen RDF-lauseen alussa, koska tarpeelliseen URI:iin viittaaminen tehdään lyhyellä prefiksillä. Prefiksien käyttö nopeuttaa dokumenttien kirjoitusprosessia, koska lyhyen etuliitteen kirjoittaminen on huomattavasti nopeampaa kuin kokonaisen URI:n. Nimiavaruudet myös sallivat samannimisten termien käytön useissa eri konteksteissa, mikä on hyvin tärkeää ominaisuus [7].

Sama termi voi esiintyä dokumentissa useita kertoja, mutta termillä ei välttämättä aina viitata samaan asiaan. Jotta kaksi samannimistä, mutta eri asiaa tarkoittavaa termiä voidaan erottaa toisistaan, täytyy hyödyntää nimiavaruuksia. Esimerkiksi nimellä Punainen viiva voidaan viitata sekä kirjaan, että elokuvaan. Mikäli elokuvia ja kirjoja sisältävät dokumentit halutaan yhdistää, niin niissä käytetyt nimet tulisi pystyä erottamaan toisistaan, jotta tieto ei mene sekaisin. Nimien erottamiseen voidaan hyödyntää nimiavaruuksia. Elokuvat voidaan määritellä *elokuva*-etuliitteen avulla ja kirjat vastaavasti *kirja*-prefiksillä. Siten lopullisessa dokumentissa esiintyisi: *elokuva:Punainen_viiva* ja *kirja:Punainen_viiva*.

Etuliitteet määritellään yleensä dokumenttien alussa, minkä jälkeen niitä voidaan käyttää määritelmissä. Prefiksin saa valita vapaasti, kunhan kyseinen etuliite ei ole jo valmiiksi käytössä. Etuliitteet täytyy siis kyettä erottamaan toisistaan. Prefiksin URI:n täytyy osoittaa olemassa olevaan resurssiin, jotta nimiavaruuden määritykset on mahdollista löytää.

3.4 RDF-tietomalli

Resource Description Framework (RDF) on semanttisen webin keskeinen tietomalli. RDF on suunniteltu kuvaamaan internetissä esiintyvää tietoa niin, että metatieto ei menetä koskaan merkitystään [3]. Lisäksi RDF pyrkii sovellus riippumattomuuteen, jolloin sen avulla voidaan kuvata lähes kaikentyyppistä tietoa [3]. Tämän ansiosta RDF-tietomalli soveltuu kaikille aloille käytettäväksi, jolloin merkityksellistä metatietoa syntyy laaja-alaisesti ja paljon. Metatiedosta on hyötyä niille, jotka pystyvät tulkitsemaan sitä. Vapaasti käytettävä ja standardoitu RDF-tietomuoto on kaikkien ihmisten ja tietokoneiden hyödynnettävissä, joten RDF-metatieto edistää kaikkia [14]. RDF-tietomuotoa hyödynnetään esimerkiksi digitaalisten valokuvien metatiedon tallennukseen [15] [16].

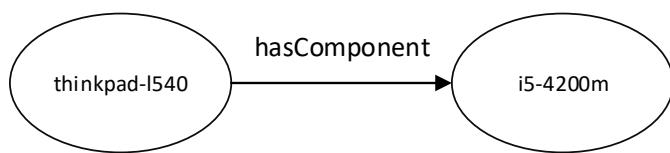
RDF-tietomalli sisältää kuvausia resurssien välisistä suhteista. Suhteita kutsutaan *predikaateiksi* tai vaihtoehtoisesti *ominaisuuksiksi* [7]. Predikaatit täytyy määritellä tietokoneelle erikseen, koska tietokone ei kykene luomaan niitä itsestään. Predikaatilla voidaan liittää kaksi resurssia yhteen ilmaisten resurssien välisestä suhteesta. Esimerkiksi predikaatilla *omistaa* voidaan kuvata resurssien liitos: *pankinjohtaja omistaa talon*. (Huomaa, että oikeaoppisesti pankinjohtajaan ja taloon pitäisi viitata URI:en avulla). Edellinen kuvaus siis määrittää tietyn pankinjohtajan ja tietyn talon välille omistusliitoksen.

3.4.1 RDF-toteamukset

Toteamukset (engl. statements) ovat RDF-tietomallin pääasiallinen tapa esittää tietoa. Toteamukset voidaan kuvata resurssien, predikaattien ja määriteltyjen arvojen avulla [7]. Esimerkiksi *auto ajaa 100 km/h* on toteamus, joka voidaan esittää RDF-tietomallilla. Esitetyn arvon ei tarvitse olla lukuarvo, vaan se voi olla myös toinen

resurssi, kuten edellisen kappaleen omistusliitos-esimerkissä on tehty [7][13]. Toteamuksset sisältävät paljon tietoa ja esitysmuoto on hyvin joustava, minkä vuoksi RDF-tietomallilla voidaan kuvata monenlaista tietoa.

Toteamus rakentuu *subjektista*, *predikaatista* ja *objektista* [17]. Rakenteensa vuoksi toteamuksia voidaan myös nimittää *kolmikoiksi* (engl. triple). Kolmikkoja voidaan esittää graafimuodossa kuvan 2 tapaan. Kaaviot saattavat auttaa hahmottamisessa, minkä vuoksi niitä käytetään havainnollistamisen tukena. On kuitenkin hyvä tiedostaa, että graafien esittämiseen ei ole määritelty standardia, joten tässä työssä käytetyt graafit ovat vain yksi monista mahdollisista esitysmuodoista. Lisäksi on hyvä ymmärtää, että resursseihin täytyy aina viitata URI:en avulla, mutta selkeyden vuoksi graafeissa saattaa esiintyä ainoastaan yksittäisiä sanoja.



Kuva 2: Esimerkki kolmikosta.

Kaaviossa esitetään toteamus: *thinkpad-l540 hasComponent i5-4200m*. *thinkpad-l540* on toteamuksen subjekti ja *i5-4200m* on objekti. Predikaatin asemassa on *hasComponent*, jonka avulla ilmaistaan, että tietokone sisältää komponentin *i5-4200m*. Yhdessä edelliset toteamuksenjäsenet muodostavat kolmikon.

3.4.2 RDF/XML-syntaksi

RDF/XML on syntaksi, jonka avulla tietokoneelle voidaan esittää RDF-tietomallin mukaista tietoa. RDF/XML-syntaksi määräää tiedon esitysmuodon, jolloin tietokone ymmärtää ihmisen esittämän tiedon halutulla tavalla. Syntaksi pohjautuu XML:ään [18]. Alla on esitetty edellisen graafin (kuva 2) toteamus RDF/XML-merkintäkielellä:

```

1 <?xml version="1.0"?>
2 <rdf:RDF
3   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
4   xmlns:rel="http://www.cmspecs.com/relationships">
5
6   <rdf:Description rdf:about="http://www.cmspecs.com/computers/thinkpad-l540">
7     <rel:hasComponent rdf:resource="http://www.cmspecs.com/processors/i5-4200m"/>
8   </rdf:Description>
9 </rdf:RDF>
  
```

Koodifragmentti 2: RDF/XML syntaksiesimerkki.

Taulukko 1: Toteamuksen osat taulukoituna. Lähde: [19].

Number	Subject	Predicate	Object
1	http://www.cmspecs.com/computers/thinkpad-l540	http://www.cmspecs.com/relationshipshasComponent	http://www.cmspecs.com/processors/i5-4200m

Yllä oleva taulukko (taulukko 1) on generoitu automaattisesti RDF/XML-merkinnän pohjalta käyttämällä W3C:n validointipalvelua. Voimme havaita, että ulkopuolin palvelu on ymmärtänyt esimerkin rakenteen täysin oikein. Validaattori on löytänyt subjektiksi tietokoneen (*thinkpad-l540*), objektiksi prosessorin (*i5-4200m*) ja predikaatiksi *hasComponent*-suhteen. Esimerkin RDF/XML-syntaksi on siis mahdollistanut tietokoneen lukea ihmisen esittämää tietoa halutulla tavalla. Nyt tietokoneet kykenevät havaitsemaan *thinkpad-l540*:n ja *i5-4200m*:n välillä vallitsevan yhteyden.

3.4.3 Turtle-syntaksi

Nykyisin RDF:ää kirjoitettaessa suositaan paljon muitakin syntakseja kuin pelkkää RDF/XML:ää. Esimerkiksi Turtle (Terse RDF Triple Language) on hyvin suosittu syntaksi sen helpon luettavuuden ja kirjoitettavuuden vuoksi [20]. Turtlessa toteamukset päättyvät pisteesseen. Alla on esitetty edellinen tietokone–prosessori esimerkki Turtle-syntaksilla:

```

1 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2 @prefix rel: <http://www.computerspecs.com/semantics/relationships> .
3 @prefix computer: <http://www.computerspecs.com/semantics/computers> .
4 @prefix cpu: <http://www.computerspecs.com/semantics/processors> .
5
6 computer:thinkpad-l540 rel:hasComponent cpu:i5-4200m .

```

Koodifragmentti 3: Turtle syntaksiesimerkki.



Kuva 3: Turtle-syntaksista automaattisesti generoitu graafi. Lähde: [21].

Vaikka edelliset syntaksiin liittyvät esimerkit ovat minimaalisia, voidaan niistä silti havaita Turtlen selkeys ja tiiveys verrattuna RDF/XML-merkintään. Turtlella RDF-toteamus voidaan kirjoittaa yhdelle riville. Subjektista, objektista ja predikaatista koostuva kolmikko päätetään aina pisteesseen tai puolipisteeseen.

3.5 Semanttinen tieto

Filosofia määrittelee termin *semantiikka* yleiseksi teoriaksi sanojen, ilmaisujen ja lauseiden merkityksestä [22]. Tietotekniikan tutkimusala on omaksunut termin käytöönsä. Semantiikka ei määrittele tiedon esitysmuotoa, joten suomenkielisen lauseiden sijaan voidaan myös tutkia RDF-tietomallin mukaisia toteamuksia. Semantiikan näkökulmasta on mielenkiintoista, kuinka tietoa voidaan esittää niin, että tiedon merkitys ymmärretään yksiselitteisesti. Semanttinen web pyrkii erityisesti esittämään ihmiseltä peräisin olevaa tietoa niin, että tietokone ymmärtää tiedon samalla tavalla kuin ihminen [2].

Luonnollinen kieli (ihmiskieli) sisältää paljon tietoa, jota on vaikea esittää tietokoneelle [23]. Esimerkiksi lauseesta “*En voi ajaa tuota autoa yhdellä jalalla*” ihminen voi tehdä useita päätelmiä. Kokonaisesta lauseesta voidaan esimerkiksi päättää:

- Lause kertoo jostain tietystä autosta, joten ehkä lauseenkertoja osoittaa autoa.
- Lauseesta käy ilmi puhujan kyvyttömyys tietyn auton ajamiseen. Ehkä auto on vääränmallinen (esim. auto on manuaalivaihteinen automaatin sijaan).
- Lauseessa esiintyvä jalka on mitä luultavimmin puhujan.
- Lauseessa puhutaan yhdestä jalasta, joten ehkä puhujan toiselle jalalle on käynyt jotain.

Lauseiden sanajärjestysellä on myös oma merkityksensä. Esimerkiksi lauseet *lapset söivät keksit ja keksit söivät lapset* eivät tarkoita samaa, vaikka ne koostuvatkin täysin samoista sanoista [23]. Kokonaiset lauseet ovat siten informatiivisempia kuin sanoista erikseen päättelävissä oleva informaatio. Tästä syystä RDF:n lauserakenne on tehokas kuvaamaan tietoa. Käytännössä tietokoneelle esitetyt toteamuksset eivät voi kuitenkaan olla yhtä tietorikkaita kuin edellinen esimerkkilause, koska tietoa täytyy pystyä käsittelemään logiikan avulla ja RDF-toteamukset koostuvat ainoastaan subjektista, predikaatista ja objektista.

Semantisella tiedolla viitataan tietoon, josta selviää itse tiedon lisäksi myös sen merkitys. Merkityksen ymmärtäminen vaatii ymmärrystä tiedon välisistä suhteista. Tiedon suhteisuutta voidaan kuvata semanttisen webin yhteydessä muun muassa luokkamäärittelyjen ja luokkahierakioiden avulla. Siten semanttisen webin tekniikoiden avulla esitetty tieto on semanttista, koska tekniikat mahdollistavat tietokoneymärrettävien suhteiden ja merkityksien kuvaamisen. Tämän ansiosta tietokone kyllätkin tekee itseenäisiä päätelmiä esitetyn tiedon pohjalta, koska se ymmärtää loogisten kuvausten ansiosta tiedon merkityksen.

3.6 RDF Schema

RDF Schema (Resource Description Framework Schema) mahdollistaa tietojoukkojen ja toisiinsa liittyvien resurssien kuvailun. RDFS tarjoaa RDF:n jatkeena lisämahdolisuksia tiedon kuvailuun. Esimerkiksi luokkamäärittelyt ovat yksi erittäin tärkeä RDFS:n työkalu, joka mahdollistaa tiedon kuvailun yleisellä tasolla [24]. RDF Schema avulla voidaan kuvata esimerkiksi talojen yleisiä ominaisuuksia luokkakuvausten avulla, kun taas RDF:llä pystytään kuvaamaan ainoastaan yksittäisten tunnettujen talojen ominaisuuksia [7]. RDF Schema on kirjoitettu RDF-tietomallin avulla, minkä vuoksi RDFS:ää hyödyntävät sovellukset ovat myös valideja RDF-sovelluksia [25].

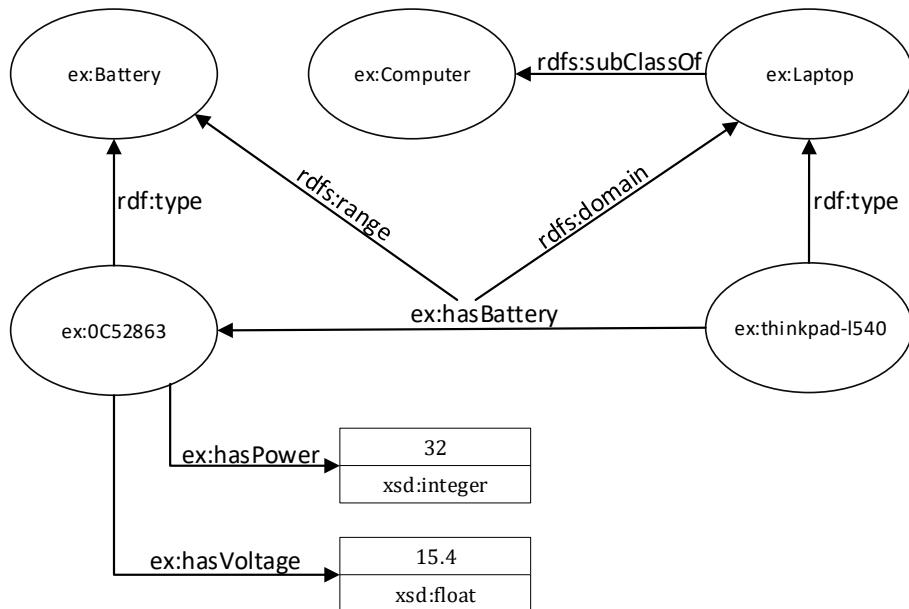
Luokkien määrittäminen auttaa järjellisen tiedon esittämisessä. Luokkien avulla voidaan luoda rajoitteita liittyen RDF-väitteisiin, mikä estää mielettömien väitteiden esittämisen. Voimme esimerkiksi määritellä *kirjoittaja*-ominaisuuden, joka kuuluu ainoastaan *kirja*-luokalle. Tämän jälkeen voidaan todeta, että kirjan *Sinuhe* kirjoittaja on henkilöluokkaan kuuluva *Mika Waltarin*. Määrittely kuitenkin estää esittämästä järjettömiä toteamuksia: *Helsingin kirjoittaja on Mika Waltarin. Helsinki* on kaupunki ja kaupungilla ei ole kirjoittajaa, joten sen kirjoittaja ei voi olla *Mika Waltarin*. Semanttinen tieto siis mahdollistaa päätelyiden tekemisen ja ristiriitaisuuksien havaitsemisen [7].

RDFS:n luokat muistuttavat luokkapainotteisten olio-ohjelointikielten luokkia. Selvästi erona on kuitenkin tapa, jolla luokan ominaisuudet määritetään. Tavallinen ohjelointikieli voi määrittää esimerkiksi luokan *auto* koostuvan *renkaista*, *ovista* ja *moottorista*. Kun luokasta luodaan autoinstanssi, niin sillä ilmenee aina edelliset ominaisuudet. RDF Schemassa luokkien ominaisuudet kuvataan globaalissa nimiavaruudessa ja ominaisuudet sidotaan sopivin resursseihin *domain* ja *range* mekanismien avulla. Siten käyttäjät voivat jatkaa toistensa tekemiä luokkia, vaikka heillä ei olisi mahdollisuutta muokata alkuperäistä luokkamääritelmää [7] [24].

Domain, *range* ja *subClassOf* ovat tärkeitä luokkiin liittyviä RDFS:n ominaisuuksia. *Domainin* avulla voidaan määrittää tietyn ominaisuuden kuuluvan tietylle luokalle. Voidaan esimerkiksi määrittää *auto*-luokka ja predikaatti *hasMotor*. *Domainin* avulla voidaan kertoa, että *hasMotor*-predikaatti kuuluu *auto*-luokalle, joten predikaattia on mieletöntä käyttää kuvailemaan esimerkiksi auton *rengasta*. *Rangen* avulla määritetään mihiin predikaatilla voidaan viitata. Edellisen esimerkin tapauksessa olisi järkevä määrittää *hasMotor*-predikaatin *rangeksi moottori*-luokka. Siten auton *moottori* voi olla ainoastaan *moottori*-luokan ilmentymä (engl. instance). RDFS:n *subClassOf* ominaisuuden avulla voidaan esitellä alaluokkia, mikä mahdollistaa luokkahierarkioiden kuvaamisen. *Auto*-luokan alaluokka voisi olla esimerkiksi *sedan* [24].

Seuraavassa esimerkissä (graafi 4 ja koodifragmentti 4) käytetään edellisessä kappaaleessa esiteltyjä RDFS-ominaisuksia. Esimerkissä määritetään luokat: *akku*, *tietokone* ja *kannettava tietokone* ja ominaisuudet: *hasBattery*, *hasVoltage* ja *hasPower*. Tämän jälkeen luodaan *Akku* ja *kannettava tietokone* ilmentymät, joita kuvataan

edellisillä ominaisuuksilla. Alla oleva graafi selventää tilannetta. Koodifragmentti on kirjoitettu Turtle-syntaksilla. Esimerkissä hyödynnetään Turtlen tarjoamaa mahdollisuutta lyhentää toteamuksia määrittelemällä niiden subjekti ainoastaan kerran. Puolipisteen jälkeiset toteamukset viittaavat edelleen samaan subjektiin, kun taas pisteen jälkeen täytyy määritellä aina uusi subjekti.



Kuva 4: RDFS esimerkin graafi.

```

1 @prefix ex: <http://www.computerspecs.com/semantics/example> .
2 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
3 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
4 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
5
6 # Class definitions
7 ex:Battery rdf:type rdfs:Class .
8 ex:Computer rdf:type rdfs:Class .
9 ex:Laptop rdf:type rdfs:Class ;
10    rdfs:subClassOf ex:Computer .
11
12 # Data properties
13 ex:hasBattery a rdf:Property ;
14    rdfs:domain ex:Laptop ;
15    rdfs:range ex:Battery .
16
17 # Instances
18 ex:0c52863 a ex:Battery ;
19    ex:hasPower "32"^^xsd:integer ;
20    ex:hasVoltage "15.4"^^xsd:float .
21
22 ex:thinkpad-l540 a ex:Laptop ;
23    ex:hasBattery ex:0C52863 .

```

Koodifragmentti 4: RDFS esimerkki.

3.7 Web Ontology Language

OWL (Web Ontology Language) on semanttisen webin yhteydessä käytetty tietämyksenesisittämiskieli. Tietämyksenesisittämiskielille tärkeitä ominaisuuksia ovat selkeästi määritelty syntaksi, formaali semantiikka, riittävä ilmaisuvoima ja päättelytuki [7]. Käyttötarkoituksestaan OWL on rinnastettavissa RDF Schemaan, mutta ilmaisuvimaltaan OWL on huomattavasti tehokkaampi kuin RDFS. RDF Schema tarjoaa ainoastaan vähimmäisedellytykset semanttisen tiedon kuvaamiseen [26]. OWL:sta on tarjolla kolme erilaajuista versiota: OWL Full, OWL DL ja OWL Lite. OWL Full on kaikista laajin versio ja monet sovellukset eivät tue sitä koko laajuudessaan [4].

Formaalilla semantiikalla tarkoitetaan kykyä ilmaista esitetyn tiedon merkitys tarkasti ja yksiselitteisesti [7]. Tämä mahdollistaa päättelyn esitetyn tiedon pohjalta. Esimerkiksi *joutsen*-alaluokkaan kuuluva *laulujoutsen* kuuluu selvästi myös *lintuluokkaan*, ja siten omaa linnulle tyypillisiä ominaisuuksia. Päättelmä voi olla tarpeellinen esimerkiksi sovelluksessa, joka kuvilee eläintyyppin tavanomaisia ominaisuuksia, kun tarkastellaan eläinlajia. OWL tarjoaa monia erilaisia keinoja päätteltävän tiedon määrittelyyn, mikä tekee siitä ilmaisuvoimaisen.

OWL määrittelee muun muassa funktionaalisen, transitiivisen ja symmetrisen luokkaominaisuuden. Edelliset ominaisuudet eivät ole ainoita OWL:n ominaisuuksia, mutta niiden tarkoitus on antaa lukijalle riittävä mielikuva OWL:n tarjoamista mahdollisuksista määritellä tietoa. Funktionaalilla ominaisuudella tarkoitetaan instanssien välistä suhdetta, jossa instanssia kohden voi ilmetä ainoastaan yksi uniikki arvo. Esimerkiksi auton ja rekisterinumeron välistä suhdetta voidaan kuvata funktionaalilla ominaisuudella. Transitiivisella ominaisuudella voidaan kuvata päätteltäviä osakokonaisuussuhdeita. Esimerkiksi *pöydänjalan* ollessa *pöydän* osa ja *ruuvin* ollessa *pöydänjalan* osa, voidaan päättellä, että *ruuvi* on myös *pöydän* osa. Symmetrisellä ominaisuudella viitataan ystävyyssuhdetta vastaavaan ilmentymään. Yleensä kaverukset kokevat toisensa ystävikseen, joten suhde on molemminpuolinen eli symmetrinen [4].

Seuraavassa esimerkissä (koodifragmentti 5) on esitetty RDF Schema luvun kannettava tietokone -esimerkki täydennettynä OWL:lla. Esimerkistä voidaan huomata, että se näyttää edelleen hyvin samalta kuin aiemmin. OWL-määritelmät ovat lisätietoa.

```

1 @prefix : <http://www.Computer-Specs.com/ontologies/example#> .
2 @prefix owl: <http://www.w3.org/2002/07/owl#> .
3 @prefix xml: <http://www.w3.org/XML/1998/namespace> .
4 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
5 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
6 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7 @base <http://www.Computer-Specs.com/ontologies/example> .
8
9 <http://www.Computer-Specs.com/ontologies/example> rdf:type owl:Ontology .
10

```

```

11 # Object Properties
12 :hasBattery rdf:type owl:ObjectProperty ;
13     rdfs:domain :Laptop ;
14     rdfs:range :Battery .
15
16 # Data properties
17 :hasPower rdf:type owl:DatatypeProperty ;
18     rdfs:domain :Battery .
19
20 :hasVoltage rdf:type owl:DatatypeProperty ;
21     rdfs:subPropertyOf owl:topDataProperty ;
22     rdfs:domain :Battery .
23
24 # Classes
25 :Battery rdf:type owl:Class .
26 :Computer rdf:type owl:Class .
27 :Laptop rdf:type owl:Class ;
28     rdfs:subClassOf :Computer .
29
30 # Individuals
31 :C52863 rdf:type owl:NamedIndividual ;
32     :hasPower "57"^^xsd:int ;
33     :hasVoltage "10.8"^^xsd:float .
34
35 :thinkpad-l540 rdf:type owl:NamedIndividual ,
36     :Laptop ;
37     :hasBattery :0C52863 .

```

Koodifragmentti 5: OWL esimerkki.

Esimerkin Turtle-merkintä on generoitu Protégé-nimisellä ohjelmistolla, jonka avulla voidaan tuottaa ontologiaita tehokkaasti [27]. Tutkielman liitteistä löytyy myös toinen laajempi ontologiaesimerkki, missä kuvataan tietokoneen rakennetta. Myös tietokone-ontologia on luotu käyttämällä Protégé-ohjelmistoa.

3.8 Ontologiat ja sanastot

Vuonna 1998 Studer et al. määritteli ontologian seuraavasti: ontologia on formaali, eksplisiittinen määritelmä sovellusalueen yhteisistä käsitteistä [28]. Edellinen määritelmä tiivistää ontologia-käsitteen merkityksen hyvin, mutta määritelmä voi olla vaikeaselkoinen, mikäli ontologia käsitettä ei tunne entuudestaan. Ontologiat voidaankin ymmärtää kokoelmina toteamuksista, jotka määrittelevät käsitteiden välisiä suhteita ja määrittelevät loogisia sääntöjä, joiden avulla tiedosta voidaan tehdä päätelmiä [2].

Ontologiat voivat olla rakenteeltaan monimutkaisia, formaaleja ja laajoja. Jos ontologian rakenne on kuitenkin melko yksinkertainen, niin ontologiaan viitataan usein termillä *sanasto* (engl. vocabulary) [29]. Toisinaan sanaston ja ontologian toisistaan

erottaminen voi olla kuitenkin hyvin haastavaa, minkä vuoksi termejä voidaankin käyttää myös toistensa synonyymeinä [29].

Ontologoiden ja sanastojen toimivuuden kannalta on tärkeää, että määrittelyt tehdään tietokoneluettavassa muodossa ja yksiselitteisesti. Yksiselitteisen eksplisiittisen määrittelyn seurauksena tieto täytyy kuvata sovellusaluekohtaisesti, jotta jokaisella termillä on vain ja ainoastaan yksi merkitys [3]. Luonnollisessa kielessä yhdellä sanalla voi olla monia merkityksiä, mutta ylimääräisistä merkityksistä päästää eroon, kun termi määritellään sovellusaluekohtaisesti. Tietokoneluettava ja eksplisiittinen käsitteiden määrittely tehostaa ihmisen ja tietokoneen välistä kommunikointia [30]. Lisäksi tietokoneen ymmärtämä termistö ja semantiikka mahdollistavat päättelyn tiedon pohjalta.

Ontologiaita ja sanastoja voidaan kirjoittaa OWL:n ja RDF Scheman avulla. Ontologioilla voidaan määritellä sovellusalueelle ominaisia tiedon rakenteita ja rajoitteita. [29]. Esimerkiksi tietokoneavusteista kokoonpanosuunnittelusovellusta varten voitaisiin määritellä *yhteensopiva*-predikaatti, sekä erilaisia *mutteri* ja *pultti* resursseja. Tämän jälkeen voitaisiin määritellä, että saman kierteen omaavat *pultit* ja *mutterit* ovat yhteensopivia, minkä seurauksena sovelluksen käyttöliittymä voisi tarjota käyttäjälle ainoastaan yhteensopivia rakennustarvikkeaihetoja. Siten M8-kierteisen pultin valinnan seurauksena ohjelma ehdottaisi käyttäjälle seuraavaksi M8-kierteistä mutteria. Hyvin määriteltyjä sanastoja ja ontologiaita voidaan uudelleenkäyttää ja jakaa muiden kesken. Esimerkiksi Dublin Core ja Friend of a Friend (FOAF) ovat todella tunnettuja ja maailmanlaajuisesti RDF:n yhteydessä usein käytettyjä ontologiaita [31].

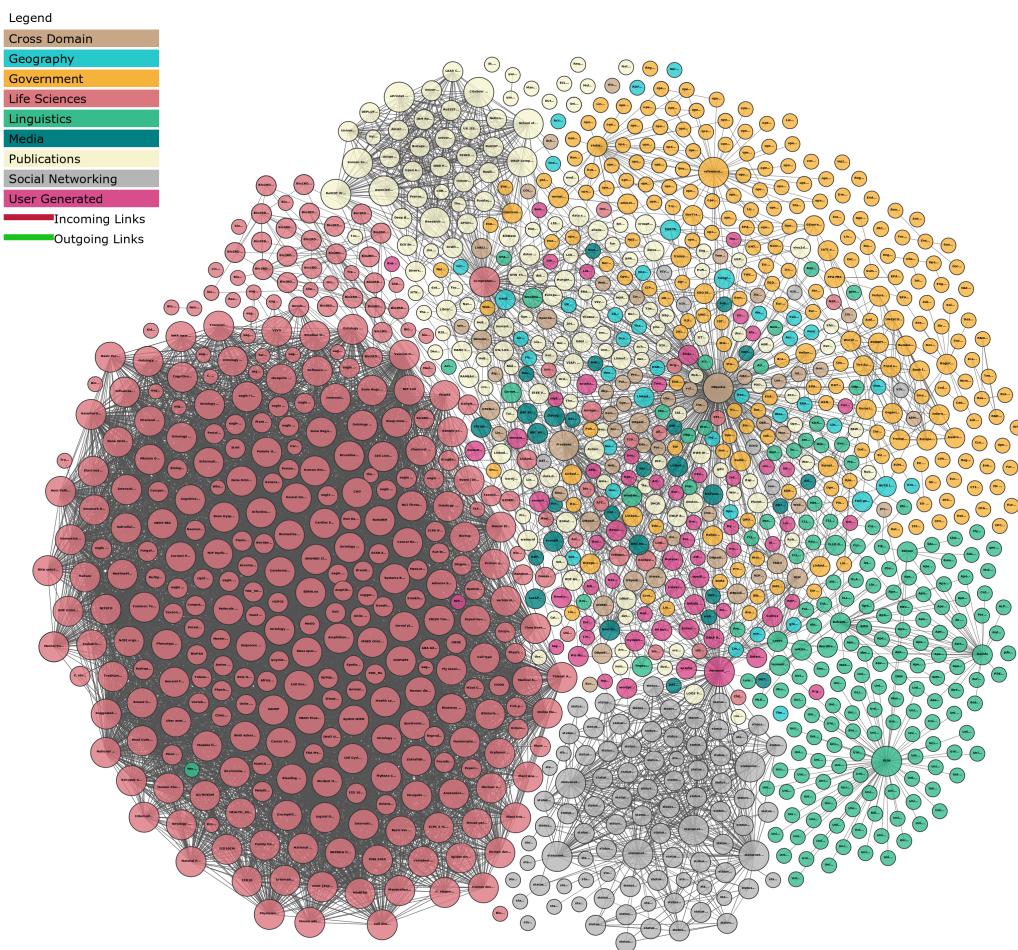
Sanastoja ja ontologiaita hyödynnetään myös tiedon integroinnissa ja organisoinnissa [29]. Esimerkiksi tuotteen maahantuojan tuotetietojen ja kauppiaan myynti tietojen yhdistäminen voi olla hyödyllistä. Kauppiaan ei tarvitse kuvalla myymiensä tuotteiden ominaisuuksia, koska hän voi hyödyntää maahantuojan valmiita kuvauksia. Kauppias ja myyjä eivät kuitenkaan todennäköisesti kuvile tuotteita täysin samalla RDF-termistöllä. Siksi tarvitaan sanasto, jonka avulla voidaan esittää tarpeellinen termien yhdistämiseen vaadittava lisätieto. Ilman termejä yhdistävää sanastoa tietokone ei kykene yhdistämään maahantuojan tuotekuvausta ja myyjän vastaavaa tuotetta toisiinsa [29].

3.9 RDF-muotoisen tiedon tallentaminen

RDF-tietomuodossa olevaa tietoa voidaan tallentaa muistiin ja tallennettua tietoa voidaan hakea muistista tiedon käyttöä varten. Yksinkertaisimillaan voidaan tallentaa suoraan RDF/XML-, Turtle- tai vastaavia tiedostoja, mutta tämä ei ole kovin tehokasta. Yksittäisen tarpeellisen asian etsiminen on tekstitiedostosta hidasta, kun tiedostokoot ovat suuria. Tästä syystä on kehitetty edellistä tehokkaampia tiedon tallennustapoja, jotka pohjautuvat tietokantoihin.

3.9.1 Triplestore

Triplestore on tietokantatyyppi, joka on suunniteltu erityisesti RDF-graafien tallentamiseen. RDF-tieto on todella linkittynyttä, minkä vuoksi graafi-tietokannat soveltuu sen tallentamiseen paremmin kuin perinteiset relatiotietokannat. Relatiotietokannoissa useiden linkittyneiden alkioiden välille on huomattavasti vaikeampaa luoda suhderakenteita ja kyselyiden tekeminen on myös haastavaa [32]. Triplestore -tietokanta tallentaa RDF-lauseet suoraan kolmikkoina, jotka sovelluskohdaisia relatiotauluja ei tarvitse luoda [33]. Triplestoren yksinkertainen rakenne ja tarkka standardointi sallivat tietokantojen liittämisen toisiinsa vaivattomasti. Tämän ansiosta tietokantoja linkittämällä voidaan luoda suuria tietoverkkoja.



Kuva 5: Graafi vapaasti hyödynnettävistä linkittyneistä tietokannoista vuonna 2017. (LOD cloud diagram). Lähde: [34]

3.9.2 SPARQL-kyselykieli

SPARQL-kyselykielen (SPARQL Query Language) avulla voidaan hakea RDF-muotoista tietoa triplestore-tietokannoista. Kyselykieli vastaa relatiotietokantojen SQL-kyselykieltä. Jokaisella triplestorella on SPARQL-päätepiste (engl. endpoint), johon SPARQL-kyselyt voidaan lähettilä HTTP (Hypertext Transfer Protocol) protokollan avulla. Päätepisteen luominen internettiin on helppoa, joten avoimia päätepisteitä on paljon tarjolla. Esimerkiksi DBpedia tarjoaa avoimen päätepisteen (<http://dbpedia.org/sparql>), josta voi hakea tietoa RDF-tietomuotoisesta Wikidiasta [7].

RDF:ssa toteamukset koostuvat subjektista, predikaatista ja objektista. SPARQL-kyselylauseessa, jokin edellisistä RDF-toteamuksen osista voidaan korvata muuttujalla, mitä merkitään kysymysmerkillä muuttujanimen alussa [7]. SPARQL etsii kyselylausesta vastaavat toteamukset ja palauttaa niistä muuttujan paikalle sopivat arvot. Kyselylause voi olla esimerkiksi seuraavanlainen:

```

1 PREFIX ex: <http://www.Computer-Specs.com/ontologies/example#> .
2
3 SELECT ?battery
4 WHERE {
5   ex:thinkpad-l540 ex:hasBattery ?battery.
6 }
```

Koodifragmentti 6: SPARQL esimerkki.

Edellinen SPARQL-kysely palauttaisi käyttäjälle tunnisteen *0C52863*, koska aiemmissa esimerkeissä *thinkpad-l540*:nakuksi on määritelty *0C52863*. Kyselyn tulos voi koostua myös useammasta kuin yhdestä arvosta, mikäli useampi toteamus toteuttaa kyselyn. SPARQL hakee tietoa prefiksien avulla määrätyistä resursseista.

SPARQL tarjoaa monia keinoja hakujen tekemiseen. Kyselyssä voi esiintyä useita muuttujia. Haut voivat koostua yhdisteistä (engl. union) ja negaatioista. Lukuarvoista koostuva tulos voidaan summata yhteen *sum*-funktion avulla. Tulokset voidaan myös järjestellä haluttuun järjestykseen *ORDER BY* -lauseen avulla. Haku voi olla *SELECT*-kyselyn sijaan myös *ASK*-tyyppinen, joka testaa, mikäli haulla on olemassa olevia ratkaisuja ja palauttaa tuloksen mukaisen boolean-arvon [35]. Erilaisia mahdollisuksia kyselylauseiden muodostamiseen on monia. SPARQL tarjoaa useita erilaisia syntakseja kyselytuloksen esittämiseen. Kyselyn tulos voidaan esimerkiksi esittää Turtle- tai RDF/XML- syntaksin mukaisessa muodossa. [36].

3.10 Linkitettu tieto

Linkitettu tieto (engl. Linked Data) sallii yhteisiin tietoresursseihin viittaamisen. Tietoa ei tarvitse monistaa, vaan eri tietolähheet voivat käyttää samoja tietoresursseja

hyödykseen [10]. Tämän ansiosta tiedon tallennukseen tarvitaan vähemmän muistitila ja tiedon muuttaminen on nopeampaa, koska muutos täytyy tehdä ainoastaan yhteen paikkaan. Lisäksi tiedon katsotaan rikastuvan, kun siihen liitetään muuta merkityksellistä tietoa [10]. Saumaton ja tehokas yhdistetyn tiedon hyödyntäminen vaatii, että sovelluksen suunnittelussa noudatetaan seuraavia sääntöjä [12] [37].

1. Resurssien nimeämiseen täytyy käyttää URI:eja.
2. URI:en tulee tukea HTTP-protokollaa, jotta ihmiset ja sovellukset voivat selvittää URI:en merkityksen.
3. URI:n osoittaman merkityksellisen tiedon täytyy olla jossakin standardiformaatissa (esimerkiksi RDF).
4. Käyttäjälle täytyy tarjota linkkejä muihin resursseihin, jotta hän voi löytää uusia mielenkiintoisia resursseja.

Graafimaiset tietokannat ovat merkittäviä linkitetyn tiedon kannalta. Tavallisten relaatiotietokantojen rakenne riippuu pitkälti sovelluksesta. Koska tietokannat ovat erilaisia eri sovellusten välillä, niin tiedon hakeminen on haastavaa, mikäli tietokantaa ei tunne entuudestaan. Tästä syystä graafimaiset tietokannat soveltuват yhdistetyn tiedon käyttöön paremmin, koska niiden rakenne ei vaihtelee sovellusten kesken. Kun viittaukset resursseihin tehdään URI:en avulla, niin tietokannan käyttäjä tietää tarkalleen mihin resurssit viittaavat [12].

4 Käytännön sovelluksia

Semanttista tietoa voidaan käyttää monella tavalla hyödyksi. Tässä kappaleessa esitellään eräitä hyödyntämismahdollisuuksia esimerkkien avulla. Esimerkit on laadittu käytössä olevista ja toimivista ratkaisuista. Jokaisessa esimerkissä yritetään painottaa eri semantisen webin osa-alueutta. Ensimmäisessä esimerkissä käsitellään RDF-tietomuotoa, toisessa semantiikkaa ja kolmannessa ontologiaita, jotka liittyvät vahvasti semantiikkaan.

4.1 XMP-metadataformaatti

XMP (Extensible Metadata Platform) on Adoben kehittämä metadataformaatti, jota käytetään muun muassa kuvien metatietojen tallennukseen. XMP on ISO:n (International Organization for Standardization) hyväksymä standardi (ISO 16684-1:2012) [38]. XMP-merkintätekniikka pohjautuu RDF-tietomallin osajoukkoon ja se hyödyntää toteutuksessaan Dublin Core sanastoa [39]. Merkintäteknikan avulla tallennetaan hyödyllistä metatietoa itse tiedostoihin, jolloin tietoa kuvaileva tieto on aina saatavilla tiedostojen yhteydessä [15].

XMP-metadataformaatista on useita hyötyjä. Sen avulla tiedostot säilyttävät niiden kontekstin, kun tiedostoja siirrellään ohjelmistolta, laitteelta tai tietokannasta toiseen. Yhteinen metadataformaatti mahdollistaa tiedostojen tehokkaan etsimisen eri tiedostotyyppien ja tietokantojen välillä. XMP-metatietojen avulla voidaan merkitä yhteyksiä eri tiedostojen välille, mikä auttaa tiedostojen järjestelyssä. Avoimen standardin ansiosta XMP:n hyödyt ovat kaikkien ihmisten ja sovellusten saatavilla [40]. Konkreettinen esimerkki XMP:n hyödyistä voidaan havaita, kun digitaalikameralla otettu valokuva siirretään tietokoneelle ja kuvanottohetki ja kameran asetukset ovat edelleen selvitettyvissä kuvan metatietoista. Liitteessä on esimerkki valokuvan XMP-metatietoista.

4.2 Semanttinen haku

Internetistä tietoa hakissaan käyttäjä pyrkii ennakoimaan sopivia avainsanoja, jotka identifioivat merkitykselliset tietolähteet [41]. Avainsanahaussa merkityksellisen tiedon löytymisen edellytyksenä on, että käyttäjä hakee tietoa tietolähteestä löytyvillä sanoilla ja termillä. Sanojen eri taivutusmuodot ja synonyymit voivat aiheuttaa haun epäonnistumisen, koska käyttäjälle merkitykselliset tulokset voivat jäädä löytymättä. Haun lopputulos voidaan kokea epäonnistuneeksi, jos käyttäjä ei kykene ennakoimaan sopivia hakusanoja [41]. Tällöin hakusanat johtavat vääränlaiseen tietoaineistoon.

Semanttinen haku pyrkii parantamaan haun onnistumismahdollisuuksia. Semanttinen haku on erityisen hyödyllinen sosiaalisen median sovelluksissa, joissa tavallinen

avainsanahaku on hankala suorittaa käyttäjän tarpeita tyydyttävällä tavalla. Esimerkiksi oikean henkilön etsiminen on haastavaa, koska usein parhaiten identifioiva avainsana on henkilön nimi, joka sekä tuottaa useita tuloksia. Semanttinen haku huomioi haun kontekstin, minkä ansiosta se tuottaa parempia hakutuloksia kuin pelkiin avainsanoihin perustuva haku [42]. Sovelluksessa konteksti voi olla esimerkiksi automaattisesti hankittu tieto hakijan sijainnista tai tieto palvelussa määritetyistä ystävyysuheteista. On yleisempää, että käyttäjä etsii lähellä asuvan ystävän ystävää kuin maapallon toisella puolella asuvaa samannimistä henkilöä.

Semanttisessa haussa yritetään ymmärtää hakijan haun merkitys, jotta käyttäjälle osataan tarjota osuvia hakutuloksia [42]. Siten käyttäjä ei ole yksin vastuussa haun onnistumisesta. Semanttinen haku sallii tavallista laajemmasta tietojoukosta etsimisen, koska haun ei tarvitse rajoittua pelkästään hakusanoilla löydettäviin tuloksiin. Yhdistetyn tiedon ansiosta voidaan esittää vahvoja asiaayhteyksiä, joita epäillään haun kannalta merkityksellisiksi [43]. Muun muassa Google hyödyntää semanttista hakua [44] [10]. Esimerkiksi haku hakusanalla “Leonardo da Vinci” tuottaa hakutulosten lisäksi tietoikkunan, jossa esitetään yleistietoa Leonardo da Vincista. Google arvaa haun tarkoitukseen olleen ehkä löytää perustietoa kyseisestä henkilöstä.

Haku voi tapahtua myös puhumalla. Puheentunnistuksen ansiosta laite pystyy poimimaan ihmisen esittämät hakusanat ja tekemään haun niiden avulla. Haun ei tarvitse koostua yksittäisistä sanoista, vaan se voidaan tehdä luonnollisen lauseen tai kysymyksen pohjalta. Kun sovellus ymmärtää kysymyksen tarkoituksen ja sovelluksella on pääsy vastauksen sisältämään tietokantaan, niin sovellus pystyy tajoamaan käyttäjän kysymykseen oikean vastauksen. Tästä syystä tietokantojen linkittäminen ja laajojen tietoverkkojen rakentaminen on antoisaa, sillä muun muassa hauissa hyödynnettävän tiedon määrä kasvaa. Edellistä tekniikkaa hyödynnetään esimerkiksi Applen Sirissä, Googlen Assistantissa ja Microsoftin Cortanassa [45] [46] [47].

4.3 Semanttinen Finlex

Semanttinen Finlex on kehitetty tarjoamaan Suomen lainsäädännöllisiä sisältöjä tietokonehyödynnettävässä muodossa. Tämä sallii sovellusten ja verkkopalveluiden hyödyntää lainsäädännöllistä materiaalia, joko lataamalla tai vaihtoehtoisesti avoimien rajapintojen avulla [48]. Semanttinen Finlex tarjoaa muun muassa SPARQL-palvelupisteen tiedon hakemiseen [49].

Semanttilaisen Finlexin esittämä tieto ja palvelut perustuvat semanttilaisen webin standardiin ja parhaisiin käytäntöihin [49]. Tieto esitetään RDF-tietomallin ja ELI-standardin (European Legislation Identifier) mukaisessa muodossa. ELI-standardi määrittelee metadatamallin (ontologian) säädöksille sekä mallin URI:n luomiseen. Tarkoituksena on edistää lainsäädännön yhteen toimivuutta Euroopan ja sen jäsen maiden kesken. ELI-ontologiaa on jatkettu projektia varten kehitetyllä SFL-ontologialla (Semantic Finlex Legislation ontology), mikä mahdollistaa muun muassa

lain väliaikaisten versioiden ja suomenkielisen lain termistön määrittämisen [50]. Ontologiat ovat hankkeen kannalta merkittäviä, koska ne määrittävät tietokoneen ja ihmisen yhteisesti ymmärtämän termistön. Ennen hanketta lait ovat olleet ainoastaan ihmisen ymmärrettävissä, mutta nyt tietokoneet kykenevät myös ymmärtämään lain termistöä [48]. Tämä mahdollistaa uusien lakiin perustuvien älykkäiden sovellusten kehityksen.

Finlexin SFL-ontologiassa on useita kymmeniä erilaisia määritelmiä. Alla (koodifragmentti 7) on kaksi esimerkkimääritelmää ontologiasta. Määritelmien on tarkoitus havainnollistaa SFL-ontologian rakennetta.

```

1 :Statute
2   rdf:type owl:Class ;
3   rdfs:subClassOf eli:LegalResource ;
4   rdfs:label "Statute"@en, "Säädös"@fi, "Författning"@sv;
5 .
6 :StatuteVersion
7   rdf:type owl:Class ;
8   rdfs:subClassOf eli:LegalResource ;
9   rdfs:label "Statute version"@en, "Säädöksen versio"@fi ;
10 .

```

Koodifragmentti 7: SFL-ontologia esimerkki.

5 Semanttisen webin nykytila ja tulevaisuus

Tässä luvussa tutustutaan semanttisen webin nykyiseen ja tulevaisuuden näkymään. Aluksi käsitellään linkitetyn tiedon laadukkuutta. Tämän jälkeen tutustutaan kolmeen erilaiseen semanttisen webin pitkääikaiseen haasteeseen. Haasteiden jälkeen käsitellään semanttisen webin kasvua ja lopuksi arkkitehtuuria.

5.1 Linkitetyn tiedon laadukkuus

Linkitetyn tiedon laadukkuudessa on eroja, sillä jotkin tietolähteet määrittelevät tietoa paremmin kuin toiset. Tästä syystä Tim Berners-Lee on luonut arvosteluasteikon, jonka avulla tarjolla olevaa tietoa voidaan luokitella. Arvosteluastekon mukaan tietolähde arvostellaan 1-5 tähdellä. Tiedon täytyy aina täyttää tähtimäärään mukainen ja sitä aiempien vaatimusten edellytykset. Arvosteluasteikko on ehdotettu laajennettavaksi välille 1-7 tähteä SeCo:n (Semantic Computing Research Group) toimesta, sillä 1-5 tähdien asteikko ei takaa tiedon laadukkuutta jatkokäytöä ajatellen [51]. Laajennettu arvosteluasteikko on esiteltyn alla [51] [52] [37].

- * Tieto on vapaassa jaossa internetissä.
- ** Tieto on tietokoneen luettavissa.
- *** Tarjolla oleva tieto esitetään avoimen standardin mukaisesti.
- **** Tiedon esittämiseen käytetään URI:eja, jotta tietoa voidaan linkittää.
- ***** Esitetty tieto on linkitetty muuhun tietoon, jolloin tiedon konteksti on saatavissa.
- ***** Tietoa kuvavat sanastot on kuvailtu eksplisiittisesti ja ne on mahdollista hankkia tiedon yhteydessä.
- ***** Tiedon tulee olla todennäköistä ja sen alkuperän täytyy olla selville, jotta tiedon laatuun voidaan luottaa.

Viiden tähdien tieto ei takaa tietoa koskevien määritysien laadukkuutta, vaikka tieto olisikin tietokoneluettavaa. Seitseman tähdien tiedon mukana täytyy tarjota dokumentaatio ja käytettyt sanastot, jolloin käyttäjä voi itse halutessaan tarkistaa sanastojen laadukkuuden ja soveltuvuuden jatkokäytöä varten. Laajennetun arvosteluasteikon tarkoitus on helpottaa ja nopeuttaa tiedon uudelleenkäytöä [51].

5.2 Haasteita

Semanttisen webin kehitykseen liittyy haasteita. Ideana semanttinien web ei ole uusi, joten useisiin teknisiin ongelmuihin on ehditty jo kehittää ratkaisuja. Osa haasteista on kuitenkin luonteeltaan sellaisia, että niihin ei voida kehittää suoraa ratkaisua. Tässä luvussa esitellään kolme semanttisen webin pitkääikaista haastetta.

5.2.1 Ihmiskielien monimuotoisuus

Ihmiskielessä termien merkitys voi riippua asiayhteydestä, minkä vuoksi sanoilla ei ole välttämättä yhtä ainoaa merkitystä. Siksi tietoa linkittääessä tätyy olla tarkkana, jotta termejä vastaavat merkitykset eivät sekottuisi keskenään. Tämän vuoksi termit määritellään yleensä sovellusalakohtaisesti [10]. Tästä kuitenkin seuraa tarve ontologioiden, semanttisen internetin tekniikkoiden ja alan termistön ymmärtävien ihmisten palkkaamiselle. Ontologiaita on mahdollista luoda automaattisesti, mutta usein viimeistään ontologian validointivaiheessa vaaditaan ihmisen tulkintaa. Validointi on myös mahdollista jättää kokonaan tekemättä, mutta tällöin ei voida varmistua ontologian oikeellisuudesta [53]. Lisäksi kaikkia luonnollisen kielen lauseita ei voida esittää RDF:n avulla. RDF soveltuu ainoastaan toteamusten esittämiseen.

5.2.2 Semanttisen tiedon ja ihmisten luotettavuus

Tiedon luotettavuuden varmentaminen on haastavaa, mikäli tiedon esittäjä ei perustele väitteitään itse tai muiden luotettavien lähteiden avulla. Jos tietoon ei voi luottaa, niin sitä on hankala hyödyntää. Ongelmaan on tarjottu ratkaisuksi tiedostojen sähköistä allekirjoittamista, minkä avulla voidaan varmentaa tietolähteen aitous. Tiedostojen sähköisen allekirjoituksen avulla ei kuitenkaan kyötä varmentamaan ihmisten luotettavuutta [54]. Ihmisten välinen luottamus pystytäisiin ratkaisemaan niin sanottujen henkilökohtaisten luotettavuusontologioiden avulla. Käyttäjät voisivat luokitella ystäviään ja automaattisesti jakaa ontologiaansa muille. Ontologiaita yhdistelemällä voidaan luoda verkkomainen graafi, minkä pohjalta voidaan laskea kaikkien käyttäjien luotettavuus. Luotettavuus riippuisi verkko-graafissa esiintyvien henkilöiden etäisyydestä ja luotettavuusarvosteluista [54]. Tämän seurauksena läheiset käyttäjät olisivat usein luotettavampia kuin ventovieraat. Ratkaisu ei kuitenkaan takaa käyttäjältä peräisin olevan tiedon todeneräisyyttä, koska luotettavatkin henkilöt voivat valehdella.

5.2.3 Alan heikko tietämys

Semanttisen webin tekniikoita ei osata hyödyntää. Ihmiset eivät tunne semantiikan tuomia hyötyjä, joten semanttisen tiedon tuottamiseen tarvittavaa työtä ei nähdä vaivan arvoisenä. Tämän seurauksena ala kehittyy ja kasvaa hitaanmin kuin se kasvaisi suuremmalla käyttäjämäärellä. Suuret yritykset, kuten esimerkiksi Facebook, Google ja Microsoft kehittävät linkitetyn tiedon verkkajaan, mikä kuitenkin lisää kiinnostusta myös pienemmissä yrityksissä [55] [44] [47].

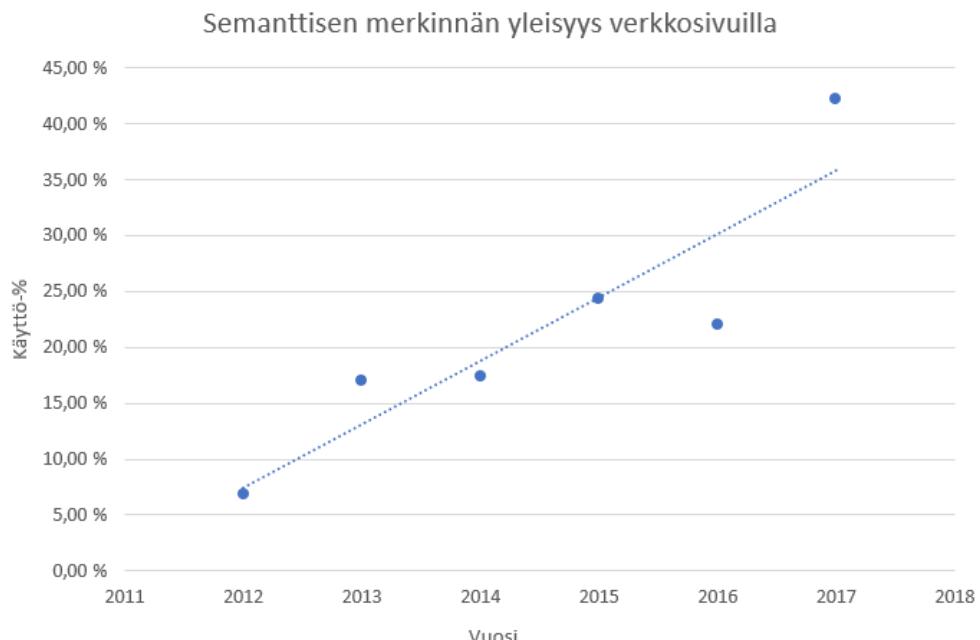
5.3 Semanttisen tiedon yleistyminen

Internetissä esiintyvä semanttisen tiedon määrä on jatkuvassa kasvussa. Kasvava kiinnostus semantiikkaa kohtaan voidaan havaita tutkimalla verkkosivuja. Verkkosi-

vuilla käytettyjä yleisiä semanttisia merkintäkieliä ovat muun muassa: Microdata, Microformat, RDFa ja JSON-LD. Bizer et al. on tilastoinut satunnaisesti valituilta verkkosivustoilta semanttista merkintää käyttävien verkkosivustojen määrän [56]. Esimerkiksi vuonna 2013 näytetako oli: 12 831 509 satunnaista verkkosivustoa, josta semanttista merkintää hyödynsi 2 189 256 sivustoa eli 17,06 %. Alla on kooste Bizerin tuloksista ja lisäksi taulukon viimeiseen sarakkeeseen on laskettu semanttisen merkinnän käyttöprosentti tulosten pohjalta.

Taulukko 2: Bizerin tilastojojen pohalta laadittu kooste.

Vuosi	Näytetako (kpl)	Semanttinen merkintätyyppi (kpl)									Esiintyy yht.	Käyttää %		
		RDFa		Microdata		Microformat								
		RDFa	Microdata	hcard	xfn	hcalendar	hreview	JSON-LD	geo	others				
2012	40600000	519379	140312	1511855	490286	37620	20781	-	48415	8659	2777307	6,84 %		
2013	12831509	471406	463539	995258	195663	20981	12880	-	23044	6485	2189256	17,06 %		
2014	15668667	571581	819990	1095517	170202	24208	13772	-	20261	6894	2722425	17,37 %		
2015	14409425	521806	1100783	1095517	139426	25721	11469	596229	-	21812	3512763	24,38 %		
2016	34076469	938830	2537539	1668039	195595	22313	16984	2116755	-	9311	7505366	22,03 %		
2017	26271491	1209430	3743822	2758884	392035	40257	27181	2685738	-	244865	11102212	42,26 %		



Kuva 6: Edellisen taulukon pohjalta laadittu kuvaaja.

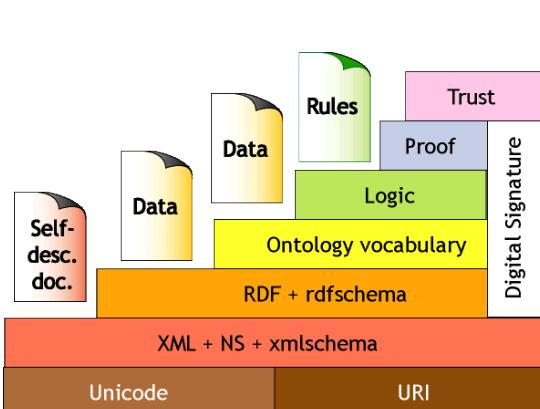
Kuvaajan trendikäyrä on johdettu yhteensä yli 140 miljoonan verkkosivustonäytteen pohjalta. Semanttisen tiedon yleistyminen on siis selvästi kasvava trendi. Bizerin tilastoja tukee vuonna 2010 tehty vastaavanlainen tiedonkeräys. Goel ja Gupta (2010) keräsivät miljoona näytettä, joista 42 605 sivua (eli 4,3 %) käytti joko Microformat tai RDFa merkintää [57]. Tulos sijoittuu Bizerin tilastojojen pohjalta johdetun lineaarisen trendikäyrän läheisyyteen.

Vuonna 2014 julkaistu HTML5-merkintäkieli sisältää semanttisia merkintöjä. Esimerkiksi elementit *header*, *main* ja *footer* voitaisiin kaikki korvata rakenteensa puolesta

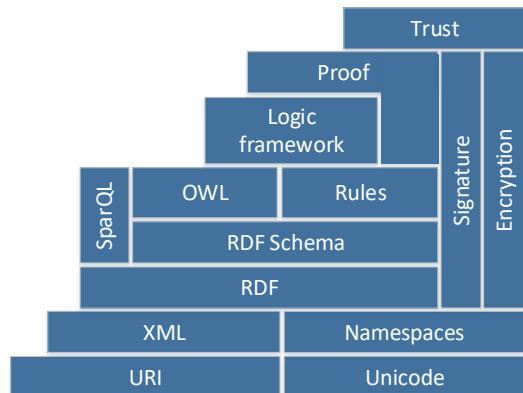
div-elementillä, mutta tällöin elementtien sisällöstä ei voida päätellä mitään. Uudet elementit, kuten *article*, *form* ja *figure* vihjaavat tietokoneelle tiedon sisällöstä ja merkityksestä eli semantiikasta [58]. Näin esimerkiksi Googlen indeksoijat (engl. crawlers) kykenevät tunnistamaan verkkosivulla esiintyvän informaation tyypin [58]. HTML5:n tarjoamat ominaisuudet on otettu laajalti käyttöön, joten nykyisin semantiikkaa hyödyntävien verkkosivustojen määärän voidaan nähdä olevan edellisiä tilastoja suurempi. Semanttisia elementtejä on kuitenkin rajallinen määärä, joten HTML5:n avulla ei voida määritellä kaikkea tiedon semantiikkaa.

5.4 Semanttisen webin arkkitehtuuri

Tietotekniikan yhteydessä arkkitehtuurilla viitataan ratkaisujen jäsentelyyn eli rakenteeseen. Semanttinen web koostuu selkeistä komponenteista, joten voidaan puhua, että semanttisella internetillä on arkkitehtuuri. Tim-Berners Lee esitti vuonna 2000 semanttisen webin arkkitehtuurin kuvan 7 mukaisena. Pinomallissa ajatuksena oli jatkaa kehitystä aina edellisen tason päälle, jolloin tekniikoihin nähdyt investoinnit säilyisivät semanttisen webin kehittyessä. Ajatus kuitenkin vaatii, että tekniikat olisivat ikuisia ja aina yhteensopivia uusien kanssa, mikä ei ole käytännössä mahdollista [11]. Tästä syystä vuoden 2000 pinomalli ei ole realistinen ja sen tilalle on esitetty kuvan 8 tarkennettua mallia.



Kuva 7: 2000-luvun malli [59]



Kuva 8: 2005-luvun malli [11]

Merkittävin ominaisuus uudessa pinomallissa on sen jakautuminen useampiin pinoihin. Tämä sallii teknikoiden muuttua ja vaihtua vapaammin ilman oletusta, että ne säilyisivät ikuisesti [11]. Vaikka malli on esitetty vuonna 2005, niin se vastaa edelleen nykypäivän (vuoden 2018) arkkitehtuuria melko hyvin. W3C:n ylläpitämästä standardeja on päivitetty, joten teknikat eivät ole päässeet vanhenemaan ja niitä hyödynnetään edelleen. Jälkimmäistä pinomallia voidaan siis edelleen käyttää lähtökohtana sovellusten rakenteen suunnittelussa.

6 Yhteenveton tietotekniikka

Semanttisessa webissä sekä ihmiset että tietokoneet voivat hyödyntää esitettyä tietoa. Tavallisessa internetissä informaatio on lähtökohtaisesti suunnattu ihmisiille, jolloin tietokoneet eivät kykene ymmärtämään esittämäänsä informaatiota. Semanttisessa webissä tietoa esitetään sellaisessa muodossa, jota myös tietokoneet kykenevät lukemaan ja ymmärtämään. Tämän seurauksena tiedon hyödyntämismahdollisuudet paranevat, koska tietokoneet kykenevät toimimaan itsenäisesti ymmärtämänsä tiedon pohjalta. Näin ihmisten ei tarvitse tehdä kaikkea työtä itse, vaan osa työstä voidaan siirtää tietokoneille.

Informaation muuntaminen tietokoneiden ymmärtämään eli semanttiseen muotoon voidaan tehdä W3C:n ylläpitämien standardien avulla. Keskeisiä tiedon esittämiseen ja määritykseen liittyviä standardeja ovat RDF, RDFS ja OWL. RDF-tietomallilla voidaan esittää tietokoneiden ymmärettävissä olevaa tietoa toteamuksien avulla. Tiedon merkityksen kuvaamiseen käytetään RDF Schemaa, mikä mahdolistaan tietokoneiden itsenäisen päättelykyvyn. RDFS määrittelee ainoastaan tarpeellisimmät tiedonkuvausmekanismit, joten tarkkaan tiedon kuvaamiseen täytyy käyttää ilmaisuvaimoista OWL:ia. Mitä tarkemmin tietoa kuvataan, niin sen paremmin tietokoneet pystyvät hyödyntämään esitettyä tietoa.

Tiedon esittämisen ja kuvaamisen lisäksi tietoa halutaan usein myös tallentaa. RDF-tietomuodossa olevaa tietoa voidaan tallentaa tehokkaasti graafimaisiin triplestore tietokantoihin. Triplestore tietokantojen tehokkuus johtuu RDF-tiedon linkityneestä rakenteesta. Triplestore tietokannoista tietoa voidaan hakea RDF:lle suunnatun SPARQL-kyselykielen avulla. Standardin mukaisia graafi-tietokantoja on helppo linjittää yhteen, jonka vuoksi niitä suositaan linkitetyn tiedon yhteydessä. Tietokantoja linkittämällä voidaan rakentaa laajoja tietoverkkoja, joita voidaan hyödyntää soveltuksissa. Tietoverkoista voidaan esimerkiksi hakea vastauksia ihmisten esittämiin kysymyksiin.

Semanttisen webin kehittäminen ei ole ongelmatoista. Semanttisessa webissä on vaikea varmistua tiedon todennäköisyydestä. Lisäksi tietokoneluettavan tiedon luomiseen vaaditaan osaamista. Alalla työskentelevien ihmisten täytyy tuntea semanttisen internetin tekniikat, jotta he osaavat hyödyntää niitä tehokkaasti. Tekniikkoiden lisäksi vaaditaan hyvää sovellusalan tuntemusta, koska muuten ontologiaista voi tulla virheellisiä. Luonnollisessa kielessä termeillä voi olla useita merkityksiä, mutta ontologioissa termien merkitykset täytyy määritellä tietokoneelle yksiselitteisesti.

Tietokoneiden ymmärtämä semanttinen tieto on arvokasta, koska sen avulla voidaan tehdä aiempaa kehittyneempiä sovelluksia. Sovellukset voivat olla toiminnaltaan itsenäisempiä ja siten älykkäämmiltä vaikuttavia. Sovellusten tehokkuus voi myös parantua, kun semanttista tietoa osataan hyödyntää järkevästi. Esimerkiksi hyvin optimoitu semanttinen haku on tavallista avainsanahakua tehokkaampi. Tehokkaamat sovellukset ovat herättäneet mielenkiintoa yrityksissä, minkä vuoksi semanttista tietoa ja semanttisia sovelluksia syntyy jatkuvasti enemmän. Semanttinen webin kasvun vuoksi alan tutkiminen ja kehittäminen on järkevää.

Viitteet

- [1] T. Berners-Lee, “Www: Past, present and future,” *IEEE Internet computing*, vol. 2, no. 4, pp. 30–37, 1998, [Artikkeli]. Saatavissa: DOI: 10.1109/2.539724. ISSN: 0018-9162.
- [2] T. Berners-Lee, J. Hendler, ja O. Lassila, “The semantic web,” *Scientific American*, vol. 284, no. 5, pp. 34–43, 1996, saatavissa: www.jstor.org/stable/26059207.
- [3] F. Manola, E. Miller, ja B. McBride, “RDF Primer,” 2014, [Internet][Viitattu 23.2.2018]. Saatavissa: <https://www.w3.org/TR/rdf11-primer/>.
- [4] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, ja L. A. Stein, “Owl web ontology language reference,” 2009, [Internet][Viitattu 23.2.2018]. Saatavissa: <https://www.w3.org/TR/owl-ref/>.
- [5] W3C SPARQL Working Group, “SPARQL 1.1 Overview,” 2013, [Internet][Viitattu 14.4.2018]. Saatavissa: <https://www.w3.org/TR/sparql11-overview/>.
- [6] American National Standards Institute, “7-bit american national standard co-defor information interchange (7-bit ascii),” new York: American National Standards Institute.
- [7] G. Antoniou, P. Groth, F. van Harmelen, ja R. Hoekstra, *A Semantic Web Primer*. The MIT Press, massachusetts Institute of Technology, USA: The MIT Press, 2012. Saatavissa: ISBN: 978-0-262-01828-9.
- [8] World Wide Web Consortium, “About W3C,” 2018, [Internet][Viitattu 23.2.2018]. Saatavissa: <https://www.w3.org/Consortium/>.
- [9] “Linked Data,” [Internet][Viitattu 15.4.2018]. Saatavissa: <https://www.w3.org/standards/semanticweb/data>.
- [10] E. Hyvönen, “Linked Data Finland,” [Internet][Viitattu 18.4.2018]. Saatavissa: <http://www.terminfo.fi/sisalto/linked-data-finland-50.html>.
- [11] M. Kifer, J. D. Bruijn, H. Boley, ja D. Fensel, “A realistic architecture for the semantic web,” 2005, saatavissa: DOI: https://doi.org/10.1007/11580072_3.
- [12] Cambridge semantics, “Introduction to linked data,” [Internet][Viitattu 15.3.2018]. Saatavissa: <https://www.cambridgesemantics.com/blog/semantic-university/intro-semantic-web/intro-linked-data/>.
- [13] S. Decker, S. Melnik, F. V. Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, ja I. Horrocks, “The semantic web: The roles of xml and rdf,” *Internet computing*, vol. 4, no. 5, pp. 63–74, 2000, [Artikkeli]. Saatavissa: DOI: 10.1109/4236.877487.

- [14] O. Lassila, “Web metadata: A matter of semantics,” *IEEE Internet computing*, vol. 4, no. 5, pp. 63–74, 2000, [Artikkeli]. Saatavissa: DOI: <http://dx.doi.org/10.1109/4236.707688>.
- [15] Adobe, “XMP specification part 1,” 2012, [Viitattu 16.4.2018]. Saatavissa: <https://www.adobe.com/devnet/xmp.html>.
- [16] Profium, “Metadata,” [Internet][Viitattu 29.3.2018]. Saatavissa: <https://www.profium.com/fi/teknologiat/metadata>.
- [17] O. Lassila, “Programming Semantic Web Applications: A Synthesis of Knowledge Representation and Semi-Structured Data,” 2007, väitöskirja, Department of Computer Science and Engineering, Helsinki. Saatavissa: ISBN: 978-951-22-8985-1.
- [18] World Wide Web Consortium, “XML RDF,” [Internet][Viitattu 3.3.2018]. Saatavissa: https://www.w3schools.com/xml/xml_rdf.asp.
- [19] “Validation Service,” 2018, [Internet][Viitattu 7.3.2018]. Saatavissa: <https://www.w3.org/RDF/Validator/>.
- [20] Cambridge semantics, “RDF vs. XML,” [Internet][Viitattu 27.3.2018]. Saatavissa: <https://www.cambridgesemantics.com/blog/semantic-university/learn-rdf/rdf-vs-xml/>.
- [21] Semantic Computing Research Group, University of Helsinki and a large consortium of Finnish public organizations and companies, “RDF Grapher,” 2018, [Internet][Viitattu 7.3.2018]. Saatavissa: <http://www.ldf.fi/service/rdf-grapher/>.
- [22] Tieteen termipankki, “Filosofia: semantiikka,” 2018, [Internet][Viitattu 8.4.2018]. Saatavissa: <http://tieteentermipankki.fi/wiki/Filosofia:semantiikka>.
- [23] R. Thomason, “What is Semantics?” 2012, [Internet][Viitattu 8.4.2018]. Saatavissa: <https://web.eecs.umich.edu/~rthomaso/documents/general/what-is-semantics.html>.
- [24] D. Brickley, R. Guha, ja B. McBride, “RDF Schema,” 2014, [Internet][Viitattu 28.3.2018]. Saatavissa: <https://www.w3.org/TR/rdf-schema/>.
- [25] F. Manola ja E. Miller, “RDF Primer,” 2004, [Internet][Viitattu 9.4.2018]. Saatavissa: <https://www.w3.org/TR/rdf-primer/>.
- [26] N. Shadbolt, W. Hall, ja T. Berners-Lee, “The semantic web revisited,” *IEEE Intelligent Systems*, vol. 21, no. 3, pp. 96–101, 2006, [Artikkeli]. Saatavissa: DOI: 10.1109/MIS.2006.62.
- [27] M.A., Musen, “The Protégé project: A look back and a look forward,” *aI Matters. Association of Computing Machinery Specific Interest Group in Artificial Intelligence*, 1(4), June 2015. Saatavissa: DOI: 10.1145/2557001.25757003.

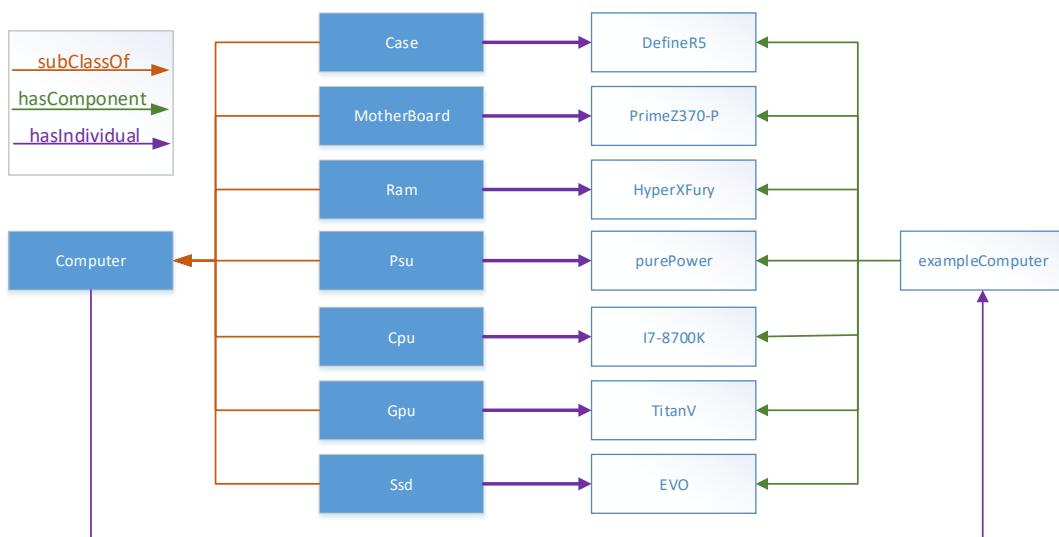
- [28] R. Studer, V. R. Benjamins, ja D. Fensel, “Knowledge engineering: principles and methods,” *Data & knowledge engineering*, vol. 25, no. 1-2, pp. 161–197, 1998.
- [29] World Wide Web Consortium, “Vocabularies,” [Internet][Viitattu 8.4.2018]. Saatavissa: <https://www.w3.org/standards/semanticweb/ontology>.
- [30] A. Maedche ja S. Staab, “Ontology learning for the semantic web,” *IEEE Intelligent Systems*, vol. 16, no. 2, pp. 72–79, 2001, [Artikkeli]. Saatavissa: DOI: 10.1109/5254.920602. ISSN: 1541-1672.
- [31] R. Cyganiak, “Top 100 most popular RDF namespace prefixes,” [Viitattu 28.3.2018]. Saatavissa: <http://richard.cyganiak.de/blog/2011/02/top-100-most-popular-rdf-namespace-prefixes/>.
- [32] K. Alexander, “The Difference Between a Triplestore and a Relational Database,” [Internet][Viitattu 14.3.2018]. Saatavissa: <http://www.krisalexander.com/uncategorized/2013/07/16/the-difference-between-a-triplestore-and-a-relational-database/>.
- [33] Ontotext, “What is RDF Triplestore?” 2018, [Internet][Viitattu 14.3.2018]. Saatavissa: <https://ontotext.com/knowledgehub/fundamentals/what-is-rdf-triplestore/>.
- [34] A. Abele ja J. McCrae, “The Linking Open Data cloud diagram,” 2017, [Internet][Viitattu 14.3.2018]. Saatavissa: <http://lod-cloud.net/versions/2017-08-22/lod.svg>.
- [35] S. Harrus, A. Seaborne, ja E. Prud’hommeaux, “SPARQL 1.1 Query Language,” 2013, [Internet][Viitattu 14.4.2018]. Saatavissa: <https://www.w3.org/TR/2013/REC-sparql11-query-20130321/>.
- [36] D. Beckett ja T. Berners-Lee, “Turtle - Terse RDF Triple Language,” [Internet][Viitattu 15.3.2018]. Saatavissa: <https://www.w3.org/TeamSubmission/turtle/>.
- [37] T. Berners-Lee, “Linked Data,” 2009, [Internet][Viitattu 14.3.2018]. Saatavissa: <https://www.w3.org/DesignIssues/LinkedData.html>.
- [38] International Organization for Standardization, “ISO 16684-1:2012,” [Internet][Viitattu 16.4.2018]. Saatavissa: <https://www.iso.org/standard/57421.html>.
- [39] Adobe, “XMP specification part 2,” 2016, [Viitattu 16.4.2018]. Saatavissa: <https://www.adobe.com/devnet/xmp.html>.
- [40] “Extensible Metadata Platform (XMP),” [Viitattu 19.4.2018]. Saatavissa: <https://www.adobe.com/products/xmp.html>.
- [41] W. Webber, “Evaluating the effectiveness of keyword search,” *IEEE Data Engineering*, vol. 33, no. 1, p. 54, 2010.

- [42] Profium, “Semantic search,” [Internet][Viitattu 19.4.2018]. Saatavissa: <https://www.profium.com/en/technologies/semantic-search>.
- [43] R. Guha, R. McCool, ja E. Miller, “Semantic search,” in *Proceedings of the 12th international conference on World Wide Web*. ACM, 2003, pp. 700–709.
- [44] Google, “The Knowledge Graph,” [Internet][Viitattu 19.4.2018]. Saatavissa: <https://www.google.com/intl/bn/insidesearch/features/search/knowledge.html>.
- [45] M. Delaforce, “Semantic Search, Siri and What it Means for SEO,” 2012, [Internet][Viitattu 19.4.2018]. Saatavissa: <https://www.4psmarketing.com/blog/search-siri-semantics/>.
- [46] A. Sanders, “What Is Semantic Search and What Should You Do About It?” 2016, [Internet][Viitattu 19.4.2018]. Saatavissa: <https://moz.com/blog/what-is-semantic-search>.
- [47] L. Heck, “Anticipating More from Cortana,” 2014, [Internet][Viitattu 19.4.2018]. Saatavissa: <https://www.microsoft.com/en-us/research/blog/anticipating-more-from-cortana/>.
- [48] Aalto-yliopisto, Semanttisen laskennan tutkimusryhmä (SeCo), oikeusministeriö ja Edita Publishing Oy, “Semantinen Finlex: Laki ja oikeus semantisessa webissä,” [Internet][Viitattu 22.4.2018]. Saatavissa: <http://data.finlex.fi/fi/main>.
- [49] “Datapalvelu ja tietomalli,” [Internet][Viitattu 22.4.2018]. Saatavissa: <http://data.finlex.fi/fi/tietomallinnus>.
- [50] “Legislation,” [Internet][Viitattu 22.4.2018]. Saatavissa: <http://data.finlex.fi/fi/lainsaadanto>.
- [51] E. Hyvönen, J. Tuominen, M. Alonen, ja E. Mäkelä, “Linked Data Finland: A 7-star Model and Platform for Publishing and Re-using Linked Datasets,” 2014, [konferenssipaperi][Viitattu 14.3.2018]. Saatavissa: DOI: https://doi.org/10.1007/978-3-319-11955-7_24 ISBN: 978-3-319-11954-0.
- [52] Linked Data Finland, “Linked Data Finland,” [Viitattu 18.4.2018]. <http://www.ldf.fi/>.
- [53] I. Bedini ja B. Nguyen, “Automatic ontology generation: State of the art,” *PRISM Laboratory Technical Report. University of Versailles*, 2007.
- [54] J. Golbeck, B. Parsia, ja J. Hendler, “Trust networks on the semantic web,” in *International Workshop on Cooperative Information Agents*. Springer, 2003, pp. 238–249.
- [55] H. Tsukayama, “Facebook introduces social search feature,” 2013, [Verkkolehti] [Viitattu 23.02.2018]. Saatavissa: https://www.washingtonpost.com/business/technology/facebook-introduces-social-search-feature/2013/01/15/599c6f7e-5f3d-11e2-9940-6fc488f3fec_story.html.

- [56] C. Bizer, R. Meusel, ja A. Primpeli, “Web Data Commons - RDFa, Microdata, and Microformat Data Sets,” 2018, [Internet][Viitattu 26.4.2018]. Saatavissa: <http://webdatacommons.org/structureddata/index.html#results-2017-1>.
- [57] T. Steiner, R. Troncy, ja M. Hausenblas, “How Google is using Linked Data Today and Vision For Tomorrow,” [Viitattu 15.3.2018]. Saatavissa: <https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/37430.pdf>.
- [58] World Wide Web Consortium, “HTML5 Semantic Elements,” [Internet][Viitattu 26.4.2018]. Saatavissa: https://www.w3schools.com/html/html5_semantic_elements.asp.
- [59] T. Berners-Lee, “Semantic web - xml2000,” 2000, [internet] Saatavissa: <https://www.w3.org/2000/Talks/1206-xml2k-tbl/Overview.html>.

Liite A Esimerkki tietokone ontologista

Tässä liitteessä on yhtenäinen Protégé-ohjelmistolla luotu ja Turtle-syntaksin mukainen ontologia esimerkki [27]. Esimerkissä kuvataan kuvitteellisen pöytätietokoneen rakennetta. Esimerkin tarkoituksena on auttaa hahmottamaan työssä esiteltyjen teknikoiden yhteiskäyttöä ja ontologioiden rakennetta, sillä tekstissä esiintyvät esimerkit ovat hyvin pelkistettyjä. Myös tämän esimerkin ontologia on yksinkertaistettu, jotta se ei veisi liikaa paperitilaa. Alla on esitelty esimerkin karkeaa rakenne graafimaisessa muodossa. Graafissa ei ole esimerkin kaikkia määritelmiä.



Kuva 9: Tietokone ontologian yksinkertaistettu rakenne.

```

1 @prefix : <http://www.semanticweb.org/otteri/ontologies/2018/2/computer#> .
2 @prefix owl: <http://www.w3.org/2002/07/owl#> .
3 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
4 @prefix xml: <http://www.w3.org/XML/1998/namespace> .
5 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
6 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7 @base <http://www.semanticweb.org/otteri/ontologies/2018/2/computer> .
8 <http://www.semanticweb.org/otteri/ontologies/2018/2/computer> rdf:type owl:Ontology .
9
10 ###### Object Properties #####
11 :hasComponent rdf:type owl:ObjectProperty ;
12     rdfs:domain :Computer ;
13     rdfs:range :Case ,
14             :Cpu ,
15             :Gpu ,
16             :MotherBoard ,
17             :Psu ,
18             :Ram ,
19             :Ssd .
20
  
```

```

21
22
23 ##### Data properties #####
24 :capacityGigaByte rdf:type owl:DatatypeProperty ;
25         rdfs:domain :Ram ,
26             :Ssd ;
27         rdfs:range xsd:integer .
28
29 :hasClockRateGhz rdf:type owl:DatatypeProperty ;
30         rdfs:domain :Cpu ,
31             :Gpu ;
32         rdfs:range xsd:integer .
33
34 :hasClockRateMhz rdf:type owl:DatatypeProperty ;
35         rdfs:domain :Ram ;
36         rdfs:range xsd:integer .
37
38 :hasCores rdf:type owl:DatatypeProperty ;
39         rdfs:domain :Cpu ;
40         rdfs:range xsd:integer .
41
42 :hasDepth rdf:type owl:DatatypeProperty ;
43         rdfs:domain :Case ,
44             :Cpu ,
45             :Gpu ,
46             :MotherBoard ,
47             :Psu ,
48             :Ram ,
49             :Ssd ;
50         rdfs:range xsd:float .
51
52 :hasHeight rdf:type owl:DatatypeProperty ;
53         rdfs:domain :Case ,
54             :Computer ,
55             :Cpu ,
56             :Gpu ,
57             :MotherBoard ,
58             :Psu ,
59             :Ram ,
60             :Ssd ;
61         rdfs:range xsd:float .
62
63 :hasMemoryType rdf:type owl:DatatypeProperty ;
64         rdfs:domain :Ram ;
65         rdfs:range rdfs:Literal .
66
67 :hasSizeInch rdf:type owl:DatatypeProperty ;
68         rdfs:domain :Ssd ;
69         rdfs:range xsd:decimal .
70
71 :hasSocket rdf:type owl:DatatypeProperty ;
72         rdfs:domain :Cpu ;
73         rdfs:range rdfs:Literal .
74

```

```

75 :hasTdp rdf:type owl:DatatypeProperty ;
76     rdfs:domain :Cpu ;
77     rdfs:range xsd:integer .
78
79 :hasWeightKg rdf:type owl:DatatypeProperty ;
80     rdfs:domain :Case ,
81             :Computer ,
82             :Cpu ,
83             :Gpu ,
84             :MotherBoard ,
85             :Psu ,
86             :Ram ,
87             :Ssd ;
88     rdfs:range xsd:decimal .
89
90 :hasWidth rdf:type owl:DatatypeProperty ;
91     rdfs:domain :Case ,
92             :Cpu ,
93             :Gpu ,
94             :MotherBoard ,
95             :Psu ,
96             :Ram ,
97             :Ssd ;
98     rdfs:range xsd:float .
99
100 :isModular rdf:type owl:DatatypeProperty ;
101     rdfs:domain :Psu ;
102     rdfs:range xsd:boolean .
103
104 #####
105 ##### Classes #####
106 :Case rdf:type owl:Class ;
107     rdfs:subClassOf :Computer .
108
109 :Computer rdf:type owl:Class .
110
111 :Cpu rdf:type owl:Class ;
112     rdfs:subClassOf :Computer .
113
114 :Gpu rdf:type owl:Class ;
115     rdfs:subClassOf :Computer .
116
117 :MotherBoard rdf:type owl:Class ;
118     rdfs:subClassOf :Computer .
119
120 :Psu rdf:type owl:Class ;
121     rdfs:subClassOf :Computer .
122
123 :Ram rdf:type owl:Class ;
124     rdfs:subClassOf :Computer .
125
126 :Ssd rdf:type owl:Class ;
127     rdfs:subClassOf :Computer .
128

```

```

129 | ###### Individuals #####
130 | :DefineR5 rdf:type owl:NamedIndividual ,
131 |   :Case ;
132 |   :hasDepth 52.1 ;
133 |   :hasHeight 45.1 ;
134 |   :hasWeightKg 11.8 ;
135 |   :hasWidth 23.2 .
136 |
137 |
138 | :EVO rdf:type owl:NamedIndividual ,
139 |   :Ssd ;
140 |   :capacityGigaByte 250 ;
141 |   :hasSizeInch 2.5 .
142 |
143 | :HyperXFury rdf:type owl:NamedIndividual ,
144 |   :Ram ;
145 |   :capacityGigaByte 16 ;
146 |   :hasClockRateMhz 2666 ;
147 |   :hasMemoryType "DDR4" .
148 |
149 | :PrimeZ370-P rdf:type owl:NamedIndividual ,
150 |   :MotherBoard ;
151 |   :hasSocket 1151 .
152 |
153 | :TitanV rdf:type owl:NamedIndividual ,
154 |   :Gpu ;
155 |   :capacityGigaByte 12 ;
156 |   :hasClockRateMhz 1455 .
157 |
158 | :exampleComputer rdf:type owl:NamedIndividual ,
159 |   :Computer ;
160 |   :hasComponent :DefineR5 ,
161 |     :EVO ,
162 |     :HyperXFury ,
163 |     :PrimeZ370-P ,
164 |     :TitanV ,
165 |     :i7-8700K ,
166 |     :purePower .
167 |
168 | :i7-8700K rdf:type owl:NamedIndividual ,
169 |   :Cpu ;
170 |   :hasClockRateGhz 3.7 ;
171 |   :hasCores 6 ;
172 |   :hasSocket "LGA1151" ;
173 |   :hasTdp 95 .
174 |
175 | :purePower rdf:type owl:NamedIndividual ,
176 |   :Psu ;
177 |   :hasDepth 20.0 ;
178 |   :hasHeight 10.0 ;
179 |   :hasWidth 15.0 ;
180 |   :isModular "false"^^xsd:boolean .

```

Koodifragmentti 8: Tietokone ontologia esimerkki.

Liite B Digitaalikuvan XMP-metadata

RDF:ään pohjautuvaa XMP-metadataformaattia käytetään muun muassa digitaalikuvien metatietojen tallentamiseen [39]. Metadata tallennetaan kuvatiedostoon, jolloin kuvatiedostoa kuvaava tieto on aina kuvan yhteydessä saatavilla [15]. Seuraavan järjestelmäkameralla otetun valokuvan (kuva 10) XMP-metadata on esitetty kuvan alla (koodifragmentti 9).



Kuva 10: Digitaalinen valokuva kissasta.

```

1  <?xpacket begin="ï»¿" id="W5M0MpCehiHzreSzNTczkc9d"?>
2  <x:xmpmeta xmlns:x="adobe:ns:meta/" x:xmptk="Adobe_XMP_Core_5.6-c067_79
   .157747,_2015/03/30-23:40:42">
3  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
4    <rdf:Description rdf:about=""
5      xmlns:aux="http://ns.adobe.com/exif/1.0/aux/"
6      xmlns:xmp="http://ns.adobe.com/xap/1.0/"
7      xmlns:photoshop="http://ns.adobe.com/photoshop/1.0/"
8      xmlns:xmpMM="http://ns.adobe.com/xap/1.0/mm/"
9      xmlns:dc="http://purl.org/dc/elements/1.1/"
10     xmlns:tiff="http://ns.adobe.com/tiff/1.0/"
11     xmlns:exif="http://ns.adobe.com/exif/1.0/">
12     <aux:SerialNumber>2050758336</aux:SerialNumber>
13     <aux:LensInfo>18/1 55/1 0/0 0/0</aux:LensInfo>
14     <aux:Lens>EF-S18-55mm f/3.5-5.6 IS</aux:Lens>
15     <aux:LensID>48</aux:LensID>
16     <aux:ApproximateFocusDistance>93/100</aux:ApproximateFocusDistance>
17     <aux:FlashCompensation>0/1</aux:FlashCompensation>
18     <aux:Firmware>1.1.0</aux:Firmware>
19     <xmp:ModifyDate>2012-04-06T16:12:47.01</xmp:ModifyDate>
20     <xmp:CreateDate>2012-04-06T16:12:47</xmp:CreateDate>
21     <xmp:MetadataDate>2012-04-06T16:12:47.01</xmp:MetadataDate>
22     <xmpMM:DocumentID>C988F5B67AD0AF364502326AC4517C8B</xmpMM:
          DocumentID>
```

```

23 <xmpMM:InstanceID>C988F5B67AD0AF364502326AC4517C8B</xmpMM:
24   InstanceID>
25   <dc:format>image/jpeg</dc:format>
26   <tiff:ImageWidth>4752</tiff:ImageWidth>
27   <tiff:ImageLength>3168</tiff:ImageLength>
28   <tiff:BitsPerSample>
29     <rdf:Seq>
30       <rdf:li>8</rdf:li>
31       <rdf:li>8</rdf:li>
32       <rdf:li>8</rdf:li>
33     </rdf:Seq>
34   </tiff:BitsPerSample>
35   <tiff:SamplesPerPixel>3</tiff:SamplesPerPixel>
36   <tiff:XResolution>72/1</tiff:XResolution>
37   <tiff:YResolution>72/1</tiff:YResolution>
38   <tiff:ResolutionUnit>2</tiff:ResolutionUnit>
39   <tiff:Make>Canon</tiff:Make>
40   <tiff:Model>Canon EOS 500D</tiff:Model>
41   <exif:ExifVersion>0221</exif:ExifVersion>
42   <exif:FlashpixVersion>0100</exif:FlashpixVersion>
43   <exif:ColorSpace>1</exif:ColorSpace>
44   <exif:PixelXDimension>4752</exif:PixelXDimension>
45   <exif:PixelYDimension>3168</exif:PixelYDimension>
46   <exif:DateTimeOriginal>2012-04-06T16:12:47</exif:DateTimeOriginal>
47   <exif:ExposureTime>1/200</exif:ExposureTime>
48   <exif:FNumber>63/10</exif:FNumber>
49   <exif:ExposureProgram>1</exif:ExposureProgram>
50   <exif:ISOSpeedRatings>
51     <rdf:Seq>
52       <rdf:li>200</rdf:li>
53     </rdf:Seq>
54   </exif:ISOSpeedRatings>
55   <exif:ShutterSpeedValue>499712/65536</exif:ShutterSpeedValue>
56   <exif:ApertureValue>5310704/1000000</exif:ApertureValue>
57   <exif:ExposureBiasValue>0/1</exif:ExposureBiasValue>
58   <exif:MaxApertureValue>5/1</exif:MaxApertureValue>
59   <exif:MeteringMode>5</exif:MeteringMode>
60   <exif:Flash rdf:type="Resource">
61     <exif:Fired>False</exif:Fired>
62     <exif:Return>0</exif:Return>
63     <exif:Mode>2</exif:Mode>
64     <exif:Function>False</exif:Function>
65     <exif:RedEyeMode>False</exif:RedEyeMode>
66   </exif:Flash>
67   <exif:FocalLength>55/1</exif:FocalLength>
68   <exif:FocalPlaneXResolution>4752000/894</exif:FocalPlaneXResolution>
69   <exif:FocalPlaneYResolution>3168000/593</exif:FocalPlaneYResolution>
70   <exif:FocalPlaneResolutionUnit>2</exif:FocalPlaneResolutionUnit>
71 </rdf:Description>
72 </rdf:RDF>
73 </x:xmpmeta>
74 <?xpacket end="w"?>
```

Koodifragmentti 9: Valokuvan XMP-metadata.