DIPLOMAMUNKA

Szakácsi János

Debrecen

2008

Debreceni Egyetem
Informatikai Kar

BÖNGÉSZÉS A SZEMANTIKUS WEBEN

Témavezető:
Jeszenszky Péter
egyetemi tanársegéd

Készítette:
Szakácsi János
programtervező matematikus

Debrecen
2008

Tartalomjegyzék

1. fejezet

Bevezetés

Az Internetet használata egyre inkább a mindennapi élet szerves részét képezi. Felhasználók milliói elektronikus levelezéssel tarthatják a kapcsolatot. A vállalatok számára ma már elengedhetetlen a webes jelenlét. A felhasználók e-boltban vásárolhatnak, a termékekről előre letölthetnek részletes technikai útmutatókat, vagy akár saját igényeik szerint állíthatják össze a kívánt terméket.

Az Interneten keresztül elérhető információ mennyisége olyan hatalmas, hogy a hagyományos kulcsszavas keresőrendszerek kezdik elveszíteni a használhatóságukat, mert egyre nehezebb úgy megadni a keresési feltételeket, hogy az eredmények száma ne százezres, vagy milliós nagyságrendű legyen, ugyanakkor nulla se. Ezen felül a felhasználó általában nem is tudja konkrétan megfogalmazni, hogy mit is keres, vagy olyan összetett információra van szüksége, amihez többféle keresési találat összekapcsolása szükséges.

A problémákat az okozza, hogy a webes tartalom nagy része emberi fogyasztásra készül szöveg, képek, stb. formájában, emiatt a gépi feldolgozásuk meglehetősen nehéz. Az ilyen jellegű problémák megoldására a Szemantikus Web adhat választ. Ennek célja, hogy metaadatokkal lássa el a webes információkat, így gépek számára is értelmezhetővé téve azokat. Ezt a gondolatot hat éve fogalmazta meg a világháló feltalálója, Tim Berners-Lee.

A metaadatok egységesen tárolt információk az információkról és a közöt-

tük fennálló kapcsolatokról, asszociációkról. Mivel a tárolás egységes, így a különböző automatizált rendszerek képesek őket egymás között megosztani és feldolgozni. Bár a Szemantikus Web technológia széleskörű elterjedése még messze van, már napjainkban is sok hatékony alkalmazás létezik.

A jelen dolgozatnak a célja, hogy áttekintést adjon azokról az eszközökről és technológiákról, melyeket azon célból fejlesztenek, hogy a tudásunkat ne csak természetes nyelven, hanem számítógép által feldolgozható formában is megoszthassuk a weben, illetve a már létező hagyományos oldalakból ki is nyerhessük ezt a tudást.

1.1. RDF (Resource Description Framework)

A Szemantikus Web lényege tehát, hogy a fellelhető erőforrásokról leírást tudjunk adni. Már a Szemantikus Web koncepciójának létrejötte előtt bevezették a webes erőforrások azonosítására az *URI* (Uniform Resource Identifier) fogalmát. Az URI tetszőleges fizikai objektumok és elvont fogalmak azonosítását is lehetővé teszi. Kétféle típusú URI létezik:

- *URL* (Uniform Resource Locator): az erőforrásokat az elérés módjával azonosítja;
- *URN* (Uniform Resource Name): az erőforrásokat egy helytől független névvel azonosítja.

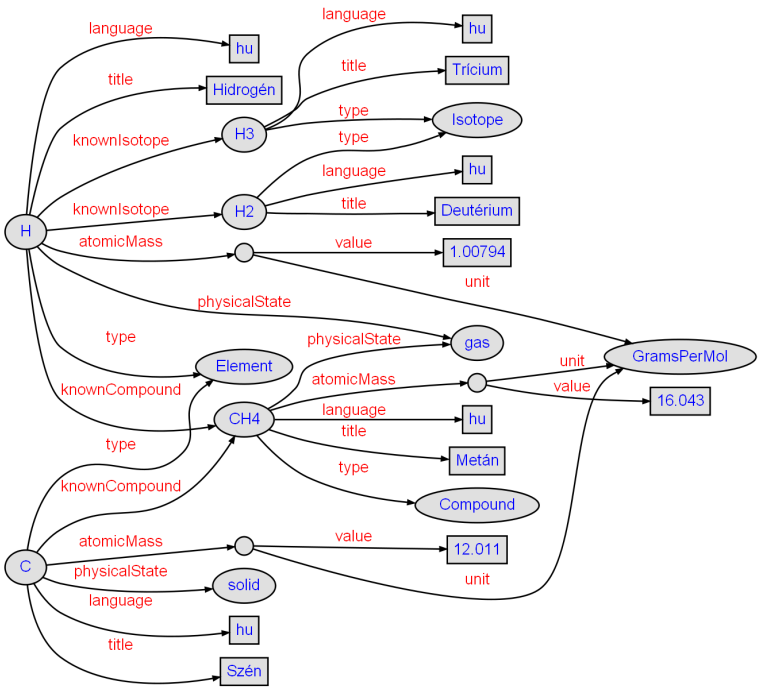
Emellett beszélünk abszolút URI-król, melyek a környezetüktől függetlenek, illetve relatív URI-król, melyek esetén szükséges egy bázis-URI jelenléte a feloldásukhoz.

A leírást – a tudás reprezentálását – az *RDF* szabvány valósítja meg. Az RDF-et úgy tervezték, hogy egyszerűen tudjon megfogalmazni kijelentéseket webes erőforrásokról. A leírás elég egyszerű, de ez egy kompromisszum az automatikus feldolgozhatóság érdekében. Az RDF-ben történő leírás olyan kijelentések sorozata, melyek három részből állnak:

- alany (subject): az erőforrás, melyről az állítást megfogalmazzuk;

- állítmány (predicate): az alany valamely tulajdonsága, ami szintén egy erőforrás;
- tárgy (object): az alanyt leíró tulajdonság értéke, mely lehet erőforrás, vagy literál.

Egy kijelentés három URI, vagy két URI és egy literál rendezett hármasa. A literálokat karakterláncokkal ábrázoljuk, és csak tárgyai lehetnek a kijelentéseknek. Az RDF *URI hivatkozás*ai[1] mellett léteznek *IRI*-k (Internationalized Resource Identifier) és *IRI hivatkozás*ok is, melyek UNICODE karaktereket használnak annak érdekében, hogy több nyelven is lehessen hivatkozásokat kreálni. Mivel gyakorlatilag bármi azonosítható URI segítségével, ezért az RDF bármit meg tud nevezni, és le tud írni bármilyen fennálló viszonyokat.



1.1. ábra. RDF gráf rövidített URI-kkal (ld. az A. függelékben kiegészítve)

[1] opcionális erőforrásrész-azonosítóval ellátott URI

Az RDF kijelentések ábrázolhatók olyan irányított gráffal is, melyben az erőforrásokat és literálokat a csúcsok, a tulajdonságokat pedig az élek reprezentálják. Egy ilyen gráfban egy kijelentést egy adott él és az általa összekötött két csúcs ír le. (Megjegyzés: RDF gráfokban csak abszolút URI-k alkalmazhatók.)

Az RDF a kijelentések géppel való feldolgozhatóságát az XML (Extensible Markup Language) technológiával valósítja meg. Az XML-t arra tervezték, hogy bárki kialakíthassa vele a saját dokumentumformátumát, majd utána meg is írhassa ebben a formátumban a dokumentumait. Az RDF egy specifikus XML szintaxist definiál, melyet *RDF/XML*-nek nevezünk. Ezt használjuk az információk reprezentálására, illetve gépek közötti cseréjére.

1.2. RDF séma

Az RDF-nek létezik egy szemantikus kiterjesztése a szókészletek leírására: az *RDF séma*. Az RDF tulajdonságok amellett, hogy attribútumként jellemezhetnek egy erőforrást, leírhatnak erőforrások közötti viszonyokat is. Ám az RDF nem bocsát a rendelkezésünkre módszereket sem az attribútumok, sem a viszonyok ábrázolására. Az RDF séma ezt a célt szolgálja, mert definiál bizonyos beépített osztályokat és tulajdonságokat, melyek segítségével aztán további osztályok, tulajdonságok és erőforrások írhatók le. Az RDF sémát az RDF szókészlet-leíró nyelvének is nevezik. A szókészletek RDF-ben íródnak megfelelő kifejezések használatával. Ezek a kifejezések tetszőleges erőforrás tetszőleges jellemzőit határozhatják meg, pl. egy tulajdonság értéktartományát.

Az RDF séma tulajdonság-, és osztályfogalma hasonló az objektum-orientált programozási nyelvek típusrendszeréhez. A köztük lévő különbséget az adja, hogy az RDF séma nem az osztályokat definiálja a tulajdonságokkal, hanem a tulajdonságokat definiálja azon erőforrásosztályokkal, amelyek rendelkeznek vele. Ilyen módon, ha már definiáltunk egy osztályt néhány tulajdonságával együtt, és később egy újabb tulajdonságot szeretnénk definiálni

az osztályhoz, nem szükséges az osztály eredeti definícióját módosítani. Ez a tulajdonság-centrikus megközelítés könnyen lehetővé teszi bárki számára egy erőforrás leírásának bővítését, ami pont a Web egyik alapelve.

Ám az RDF séma igazából alkalmatlan arra, hogy a Szemantikus Web alapja legyen, mert hiányosságokkal bír az osztályok közötti kapcsolatok leírásában, csak korlátozott módon képes következtetések elvégzésére, stb. Ezért bevezették az ontológiákat, melyek leírására az *OWL* nyelv (Web Ontology Language) szolgál. Az OWL célja már nem az, hogy pusztán reprezentálja az információt az ember számára, hanem az információ gépi feldolgozása, következtető gépek nagymértékű támogatása. Ám ez már kívül esik a böngészés témakörén.

1.3. Megvalósítások

Napjainkban különféle módszerek léteznek arra, hogy miként szolgáltathatunk RDF adatokat egy weboldal mellé. Az egyik legkézenfekvőbb módszer az, ha a metaadatok egy különálló RDF fájlban szerepelnek. Ám ez meglehetősen kényelmetlen megoldás, kevesen alkalmazzák. Egy másik törekvés az RDFa szabvány, mely az RDF adatok közvetlenül XHTML-be történő beágyazását tűzte ki célul. Létezik még egy harmadik lehetőség is, mely során egyszerűen a „nyers” HTML fájlokból nyerik ki az adatokat. Erre szolgálnak a screen scraperek.

A további fejezetekben ez a három módszer kerül bemutatásra részletesen, illetve konkrét megvalósítási technikáik.

2. fejezet

RDFa

2.1. A szintaxis kialakulása, elemei, lehetőségei

Az RDF-hez sokáig nagy remények fűződtek, mert rugalmasan képes metaadatokat tárolni, csoportosítani, és használni. A leggyakrabban használt szintaxisa az RDF/XML, ám sok embert elriasztott a bonyolultsága miatt. Ezért a W3C[1] belekezdett egy új, egyszerűbb szintaxis kidolgozásába, mely az RDFa (eredetileg „RDF/a") nevet kapta, és használata sokkal könnyebb az alkalmazásokban. Azt remélik, hogy így sikerül visszahódítani azokat a felhasználókat, akik elidegenedtek az RDF/XML-től annak kellemetlenségei miatt.

Az RDF/XML másik problémája az, hogy még egy egyszerű, RDF/XML-ben adott metaadatleírás sem illeszthető be jól XHTML-be, mert a HTML köré tervezett alkalmazások nem tudják megemészteni. Tehát a célkitűzésben az RDFa tetszőleges XML-be való ágyazhatósága mellett az XHTML-be ágyazhatósága is benne foglaltatott. Ez utóbbi pedig igen fontos tulajdonság a Szemantikus Weben. Hiszen a Szemantikus Web filozófiája szerint egy weboldal az emberi szemmel való olvashatóság mellett automatizált folyama-

[1] World Wide Web Consortium

tokkal is feldolgozható kell legyen, melyek összesítik az adatokat, a hozzájuk kapcsolt metaadatokat, majd olyan bonyolult feladatokat végeznek el rajtuk, amire a mai hagyományos screen scraperek nem képesek.

Már manapság is számos szoftver áll rendelkezésre, melyek lehetővé teszik RDF hármasok XHTML-ből történő kinyerését és feldolgozását, pedig a specifikáció még nem készült el teljes egészében.

2.1.1. Mi is az „a" az „RDFa"-ban?

Az RDF az alany, állítmány, tárgy kombinációkat használja, hogy név-érték pár formájában leírást adjon egy konkrét erőforrás valamely attribútumáról. Ahhoz, hogy metaadatokat rendelhessünk egy weboldalhoz anélkül, hogy a böngészőbeli megjelenése megváltozna, az RDFa már létező XHTML 1 és néhány XHTML 2 attribútumot használ az alany, állítmány, tárgy hármasok tárolására. Ezek XHTML 1-ben a *href*, *content*, *rel*, *rev* és *datatype*, illetve XHTML 2-ben az *about*, *role* és *property* attribútumok. A két legalapvetőbb eset látható a következő táblázatban. Az elsőben a hármas tárgya egy sztring literál, a másodikban egy URI.

alany	állítmány	tárgy
about	*property*	*content* attribútum vagy PCDATA
about	*rel*	*href*

2.1. táblázat. XHTML-beli metaadatok leírására használható attribútumok

2.1.2. RDFa elemek

Tekintsük az alábbi két példát:

```
<span about="http://gumicsizma.hu/index.php"
      property="dc:title" content="Gcs 2007"/>
```

```
<span about="http://gumicsizma.hu/index.php"
      property="dc:title">Gcs 2007</span>
```

A szintaxis különböző, de mindkét hármas ugyanazt az állítást fogalmazza meg, miszerint a *http://gumicsizma.hu/index.php* erőforrásnak van egy cím tulajdonsága, amely a „Gcs 2007" értékkel rendelkezik. Emellett mind a két példa a *span* elemet alkalmazza, amely közkedvelt az RDFa-ban, mert tetszőleges helyre beilleszthető egy HTML dokumentum törzsében. A *link* és *meta* elemek hasonlóan jól használhatóak a fejrészben. Ez is része az RDFa szépségének, hiszen ez a két elem már évek óta metaadatok hozzárendelésére szolgál (például a weboldal CSS stíluslap URL-jének megadása), és most mindössze apró módosításokkal az RDFa felismerő szoftverek hasznos információkat tudnak kinyerni belőlük. (A módosítások azért szükségesek, mert egy XHTML 1 *link* eleméből kinyert hármas, amely egy CSS stíluslapra mutat, nem lenne teljes mértékben megengedett RDF, hiszen a *rel* „stylesheet" értéke nem egy URI, és ennek megfelelően nem valódi RDF állítmány.)

```
<html>
  <head>
    <link href="special.css" rel="stylesheet" type="text/css">
    ...
```

A *link* elem további hasznos tulajdonsága, hogy alkalmas RDFa metaadatok tárolására, mert minden esetben egy viszonyt ír le egy erőforrás (a dokumentum, amelyben szerepel), és egy másik között (ahová a hivatkozás mutat). Az *a* elem *rel* attribútuma a viszonyról szolgáltat információt, mely ezáltal az *a* elem által leírt hármas állítmányaként szolgál.

2.1.3. Több hármas, kevesebb alany

Ha egy dokumentum 100 hármasnyi metaadattal rendelkezik, azoknak valószínűleg nem 100 különböző alanya van. Ugyanakkor az az alany, melyről sok állítás szerepel, alighanem maga a dokumentum, mivel az állítások megadják

a címét, szerzőjét, stb. A hármasok egy másik csoportja leírást adhat egy képről, hogy ki fényképezte, mikor, milyen újrafelhasználási jogokkal rendelkezik.

Meglévő XHTML szintaxisra építve az RDFa lehetővé teszi ugyanarról az alanyról több hármas megfogalmazását anélkül, hogy túlságosan rendezetlenné, áttekinthetetlenné válna a dokumentum. Ha egy RDFa feldolgozó az *about* attribútum hiányát észleli, akkor feltételezi, hogy a magában foglaló elem az alany. A következő példa három metaadatot rendel a helyen lévő erőforráshoz, mivel a három *span* elem bár nem rendelkezik *about* attribútummal, de a szülő *img* elem igen:

```
<img src="http://egyhost.hu/szaki/aranybika.jpg"
     about="http://egyhost.hu/szaki/aranybika.jpg">
  <span property="dc:subject" content="Aranybika"/>
  <span property="dc:creator" content="Szakácsi János"/>
  <span property="dc:format" content="img/jpeg"/>
</img>
```

Ha az RDFa feldolgozó egy elem esetén nem talál *about* attribútumot, megpróbál alanyt keresni a szülőelemeken fölfelé haladva az adott elem állítmányával és tárgyával. Amennyiben a keresés sikertelen, az alany az üres sztring, amely az RDFa szintaxis specifikációja szerint az aktuális dokumentumot jelzi.

Ez praktikus, hiszen egy dokumentum metaadatai közül számos, magáról a dokumentumról szól. Például egy dokumentum főcímében szerepelhet az alábbi *span* elem, mely a tartalmának a Dublin Core cím tulajdonságára utal (feltéve, hogy a *h1* elem egyetlen felmenője sem rendelkezik *about* attribútummal).

```
<h1><span property="dc:title">Mest. int. 2</span></h1>
```

De a fejrészben megjelenítés nélkül is lehet metaadatot tárolni a dokumentumról.

```
<html>
  <head>
    <meta description="dc:date" content="2007-10-02T11:09:36"/>
    ...
```

2.1.4. Sorokba beágyazott metaadatok

Az RDFa használatának ezen kategóriája valóra váltja az eredeti elképzelést, amely a megalkotásához vezetett, vagyis hogy miként tegyünk egy ember által olvasható weboldalt gép által is olvashatóvá. Az RDFa sorokba ágyazott használatát szemlélteti a következő példa:

```
<p>Ezt a fényképet <span class="author" about="dcim0173.jpg"
  property="dc:creator">Kiss Éva</span> készítette.</p>
```

A *span* elem és attribútumai az alábbi (RDF-ben leírt) hármas kinyerését teszik lehetővé egy RDFa felismerő szoftver számára:

```
<rdf:Description rdf:about="file:///C|/web/kepek/dcim0173.jpg">
  <dc:creator>Kiss Éva</dc:creator>
</rdf:Description>
```

A fenti esetben a „Kiss Éva” értékű sztring szolgáltatja a hármas tárgyát. Szükség lehet azonban az adatok megjelenítésének egy alternatív megoldására, például egy dátum esetén. Ilyenor a *content* attribútum segítségével felülírható az érték:

```
<p>
  Utoljára módosítva:
  <span about="http://egyhost.hu/szaki/docs/doc1.html"
    property="dc:date" content="20071002T13:57:09">
    2007. október 2. 12:57:09
  </span>
</p>
```

A kinyert hármas a *content* értékét használja fel, mely RDF-ben a következőképpen néz ki:

```
<rdf:Description rdf:about="http://egyhost.hu/szaki/doc1.html">
  <dc:date>20071002T12:57:09</dc:date>
</rdf:Description>
```

Ezek után sokkal könnyebben meghatározható egy dátum szerinti keresés eredménye – például mely dokumentumokat módosították 2007. október 1-je és 10-e között legutoljára.

2.1.5. Metaadatok a tartalmazó dokumentumról

Az RDFa eredeti célja az volt, hogy a dokumentum összetevőiről annak soraiba ágyazva lehessen metaadatokat leírni. Ám az egyszerű konstrukciójának köszönhetően könnyen használható másfajta – pl. maga a dokumentum metaadatainak definiálására is. Néhány metaadat – mint a dokumentum címe és szerzője – gyakran redundásan szerepel a dokumentumban már meglévő adatokkal, és a 2.1.4. szakaszban leírt módon soron belül megjelölhető. Más – például munkafolyamattal kapcsolatos – metaadatok megadására viszont a fejléc is megfelel. Ha nincs megadva a hármasok alanya, akkor az RDF feldolgozó üres sztringet tekint a hármas alanyának, amely magát a dokumentumot reprezentálja:

```
<html xmlns:fm="http://www.foomagazine.com/ns/prod/">
  <head>
    <title>Is Black the New Black?</title>
    <meta property="fm:newsstandDate" content="2006-04-03"/>
    <meta property="fm:copyEditor" content="RSelavy"/>
    <meta property="fm:copyEdited"
          content="2006-03-28T10:33:00"/>
  </head>
  <body>
  ...
```

Ebből az alábbi RDF/XML nyerhető ki:

```
<rdf:Description rdf:about="">
  <fm:newsstandDate>2006-04-03</fm:newsstandDate>
  <fm:copyEditor>RSelavy</fm:copyEditor>
  <fm:copyEdited>2006-03-28T10:33:00</fm:copyEdited>
</rdf:Description>
```

2.1.6. „Soron kívüli" metaadatok

Az XHTML 2 újdonsága, hogy a *meta* elemek egymásba ágyazhatók, mely megengedi, hogy egy alanyról tetszőleges sok hármast írhassunk le. Abban az esetben, ha ezt a fejrészben tesszük meg, specifikálhatjuk a dokumentum konkrét elemeit alanyként. Így már a fejlécben lehetőség nyílik metaadatok megadására a weboldal különálló részeiről. Például egy recepteket leíró dokumentum definiálhat metaadatokat a konkrét receptekről a fejben. (A következő minta a *section* és a *h* új, XHTML 2-beli elemeket is alkalmazza.)

```
<html xmlns:fm="http://www.foomagazine.com/ns/prod/">
  <head>
    <meta about="#recipe13941">
      <meta property="fm:ComponentID">XZ3214</meta>
      <meta property="fm:ComponentType">Recipe</meta>
      <meta property="fm:RecipeID">r003423</meta>
    </meta>
  </head>
  <body>
    <h>Add Some Tex Mex Sizzle to Your Kid's Lunch</h>
    ...
    <section id="recipe13941">
      <h>EZ Bean Tacos</h>
      ...
    </section>
```

```
    ...
  </body>
</html>
```

A kinyert hármasok metaadatai csak a dokumentum *recipe13941* azonosítóval rendelkező elemére vonatkoznak.

```
<rdf:Description
  rdf:about="file:///C|/honlap/receptek.html#recipe13941">
  <fm:ComponentType>Recipe</fm:ComponentType>
  <fm:RecipeID>r003423</fm:RecipeID>
  <fm:ComponentID>XZ3214</fm:ComponentID>
</rdf:Description>
```

Mivel az RDFa lehetővé teszi teljes hármasok eltárolását egy HTML dokumentumban, ennek erőforrásairól még egy különálló HTML dokumentumban is megadhatunk metaadatokat (feltéve, hogy rendelkeznek azonosítóval, mint a fenti recept).

Az RDFa egyik alkalmazási módja a mikroformátumok specifikálása és használata. Ezekről a 2.2. szakaszban lesz szó.

2.2. Mikroformátumok

A mikroformátumokat sokféleképpen definiálják. A hivatalos definíciója szerint mikroformátumoknak nevezzük a már létező, széleskörben alkalmazott szabványokra épített, egyszerű és nyílt adatformátumok halmazát, melyeket elsősorban emberek, másodsorban gépek számára terveztek. Egy szintén népszerű, ám részletesebb megfogalmazás alapján a mikroformátumok:

- Egyszerű konvenciók szemantikus jelentés beágyazására egy specifikus problémakörhöz kialakítva.
- Szabványosított, de ember által is olvasható formában írnak le adatokat (X)HTML, XML dokumentumokban és Atom/RSS feedekben, miközben tömör osztályneveket alkalmaznak.

– Gyakran már létező és funkcionáló szabványokra alapoznak.

– Lehetővé teszik az erőforrások, eszközök és szolgáltatások decentralizált fejlesztését.

Manapság már számos mikroformátum létezik. Ha például egy vállalat szeretné az elérhetőségével kapcsolatos információit megosztani és könnyen kereshetővé tenni, megteheti a *hCard* segítségével. Ha egy rendezvényeket szervező cég jobban fel akarja magára hívni az emberek figyelmét, *hCalendar* formátumban leírhatja és közzéteheti az eseményeinek adatait. Az *hReview* lehetővé teszi termékekről, szolgáltatásokról, vállalatokról, rendezvényekről, stb. szóló vélemények szabványos megfogalmazását. Lássunk egy konkrét példát, a *hCard* formátumot!

2.2.1. hCard

Az hCard mikroformátum alapjául a *vCard* szabvány (RFC2426) szolgál. A célja alapvetően a vCard objektum- és attribútumneveinek kisbetűs használata osztálynevek gyanánt, illetve a vCard objektumok közvetlen hozzárendelése egymásba ágyazott XHTML elemekhez.

A gyökér osztály neve „vcard", így a „vcard" osztálynévvel definiált elemeket *hCard*oknak nevezzük. A hCard attribútumait a hCardon belül elhelyezkedő elemek reprezentálják. Az attribútumok értékeit az alább felsorolt osztálynevekkel rendelkező elemek adják meg. Némelyik rendelkezdik kiegészítő attribútumokkal is, melyeket a beléjük ágyazott elemek írnak le. (A kiegészítő attribútumok kettőspont után felsorolva szerepelnek.)

Kötelező osztálynevek:

- fn (formális név)
- n (egyéb nevek): family-name, given-name, additional-name, honorific-prefix, honorific-suffix

  - opcionális, ha érvényben van valamely implicit optimalizációs szabály

Opcionális osztálynevek:

- nickname, sort-string (rendezést elősegítő sztring)
- url
- email: type, value
- tel: type, value
- adr (cím): post-office-box, extended-address, street-address, locality, region, postal-code, country-name, type, value
- label (adatmezők szerint nem különválasztott – egybeírt – cím)
- geo (földrajzi hely): latitude, longitude
- tz (időzóna)
- photo, logo, sound, bday (születésnap)
- title, role
- org (szervezet): organization-name, organization-unit
- category, note
- class, key, mailer, uid, rev

Az *fn*, *n*, *bday*, *tz*, *geo*, *sort-string*, *uid* és *class* osztálynevek egyediek, azaz több deklaráció esetén csak az első elemnek van hatása, a többi figyelmen kívül marad. A többi osztálynév szerepelhet többszörösen – ekkor az adott attribútumnak ugyanannyi példánya is lesz.

Bizonyos esetekben egy attribútum értékét csak az elemének egy része képezi. Általában akkor fordul elő, ha az attribútumnak van egy altípusa, pl. *tel*. Ilyenkor a speciális *value* osztálynév segítségével választható ki az elem azon része, mely az attribútum értékét adja. Az alábbi példa ezt szemlélteti:

vCard:

```
TEL;TYPE=HOME:+1.415.555.1212
```

hCard:

```
<span class="tel">
 <span class="type">home</span>:
 <span class="value">+1.415.555.1212</span>
</span>
```

A továbbiakban a hCard mikroformátum használata közben rendelkezésre álló számos optimalizációs lehetőség és egyéb kiegészítések lesznek ismertetve.

Attribútum kivételek

A vCard néhány attribútumának vagy nincs értelme, vagy már eleve kiderül az értékük egy weboldal környezetében. Ide tartozik a *name*, *profile*, *source*, *prodid* és a *version*. A tartalomszolgáltatóknak nem szabad ezeket használniuk a hCard-jaikban, illetve a fogyasztóknak figyelmen kívül kell hagyniuk. Viszont a hCard-vCard konvertereknek például célszerű felhasználniuk a weboldal címét (vagyis a HTML dokumentum `<title>` elemét) a *name* attribútum létrehozásakor, az oldal URL-jét a *source* attribútumhoz, stb.

Szervezetek elérhetőségi információja

Ha az *fn* és az *org* attribútumok értéke megegyezik, akkor a hCard a cég, szervezet, vagy hely(szín) elérhetőségi információit írja le, és így is kell kezelni. Ilyen esetben a hCard szerkesztőjének nem szabad beállítania az n attribútumot, vagy explicit módon az üres sztringet kell megadnia. Ám a szintaktikai elemzés során figyelembe kell venni az üres n attribútumot és minden kiegészítő attribútumát is üres sztringként kell értelmezni.

Implicit *n* optimalizáció

Bár a vCard megköveteli az *n* attribútum jelenlétét, bizonyos speciális esetekben mégis elhagyható – ilyenkor az *fn* attribútumból lesznek származtatva az *n* értékei.

Ha az *fn* és az *org* attribútumok értékei különböznek, és az *fn* pontosan két szóból áll (whitespace-szel elválasztva), illetve nincs explicit módon megadva az *n* attribútum, akkor az *n* az *fn*-ből származtatható. Amennyiben az *fn* egy szóból áll, ld. lejjebb, illetve ha három, vagy annál több szóból tevődik össze, akkor a szerkesztőnek kötelező értéket adnia az *n* attribútumnak is (kivéve az előző bekezdésben szereplő szervezetek elérhetőségi információi esetén).

A származtatás során az *fn* két különálló szóvá tagolódik, melyek közül az első az *n* *given-name* kiegészítő attribútumaként lesz értelmezve, a második pedig a *family-name* értéke lesz. Ám abban az esetben, ha az első szó vessző karakterrel végződik, a sorrend felcserélődik.

Implicit *nickname* optimalizáció

Mivel a weboldalakon jelentős mértékben használnak becenéveket, felhasználói neveket, stb., ezért erre is kialakítottak egy optimalizálási lehetőséget. Ha az *fn* attribútom értéke különbözik az *org* értékétől, és az *fn* csupán egyetlen szóból áll, akkor egyrészt ez a szó jelenti a *nickname* attribútum értékét, másrészt a hiányzó *n* minden kiegészítő attribútumának értéke implicit módon az üres sztring lesz.

Implicit *organization-name* optimalizáció

Az *org* attribútumnak két kiegészítő attribútuma van: az *organization-name* és az *organization-unit*. Nagyon gyakran előfordul, hogy a szerkesztők csak az *organization-name*-et teszik közzé. Ezért ha egy *org* attribútumon belül nincs definiálva *organization-name*, akkor az *org* teljes tartalmát kell az *organization-name* értékének tekinteni.

Lássunk végül egy teljes hCard példa a WikiMedia Foundation vállalatról!

```
<div class="vcard">
  <div class="fn org">Wikimedia Foundation Inc.</div>
  <div class="adr">
    <span class="type">work</span>
    <div class="street-address">200 2nd Ave. South #358</div>
    <div>
      <span class="locality">St. Petersburg</span>
      <abbr class="region" title="Florida">FL</abbr>
      <span class="postal-code">33701-4313</span>
    </div>
    <div class="country-name">USA</div>
  </div>
  <div>Phone: <span class="tel">\\*
    <span class="type">work</span>\\*
    +1-727-231-0101</span>\\*
  </div>
  <div>Email: <span class="email">\\*
    info@wikimedia.org\\*
  </span></div>
  <div>
    <span class="tel"><span class="type">Fax</span>:
    <span class="value">+1-727-258-0207</span></span>
  </div>
</div>
```

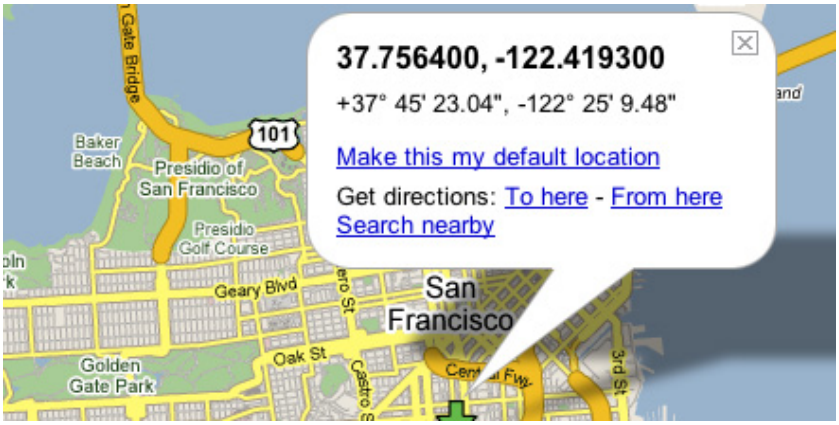
2.2.2. Operator

2006. decemberében elkészült a Mozilla Labs egyik mikroformátum felismerő kiegészítőjének első verziója, az Operator. A bővítmény már akkor számos lehetőséget nyújtott, melyek folyamatosan bővülnek. Ahogy egyre több web-

oldal kezd mikroformátumokat beágyazni, egyre sokrétűbb lesz a használhatósága.

Az Operator képes

- Például egy kedvenc étterem telefonszámát közvetlenül a saját címjegyzékbe másolni a Yahoo! Local oldaláról egyetlen kattintással, anélkül, hogy bármit is gépelni kellene.
- Az Upcoming.org webhelyről egy elkövetkezendő eseményt és adatait a naptárunkba illeszteni, hogy megnézhessük ráérünk-e épp, vagy egy megfelelő térképen bejelölni a helyszínét, hogy vizuálisan is megtekinthető legyen.
- A Flickr képgalériába feltöltött képet szintén térképre helyezni, mivel ún. *geotag* jelöléssel vannak ellátva a származási helyük szerint.
- És még számos egyéb funkcióra, melyek köre folyamatosan bővül.



2.1. ábra. Geotag-gel ellátott objektumok térképen való megjelenítése

A kiegészítő a mikroformátumok felderítése mellett tartalmaz fejlesztők számára tervezett funkciókat is, ám ez túlmutat a disszertáció keretein.

3. fejezet

Csatolt RDF fájlok

Weboldalakhoz úgy is lehet szemantikus tartalmat hozzárendelni, hogy egy szeparált RDF fájlban írjuk le a metaadatokat. Különféle szókészletek léteznek specifikus témakörök leírására. Ilyen a FOAF, a DOAP és a SIOC is.

3.1. FOAF (Friend of a Friend)

A FOAF project célja, hogy a weben gép számára is olvasható módon leírhatók legyenek az emberek, a közöttük fennálló kapcsolatok, és olyan dolgok, amiket létrehoznak vagy tesznek. A technológia igen egyszerű, és lehetővé teszi ezen információk megosztását, webhelyek közötti átvitelét, automatikus kiterjesztését vagy összefésülését.

A FOAF nem más, mint egy összekapcsolt információs rendszer – mint maga a web. Decentralizált Szemantikus Web technológiára épül, és úgy tervezték, hogy lehetővé tegye az adatok integrációját különféle alkalmazások, webhelyek, webes szolgáltatások és szoftverrendszerek között. Ennek érdekében a FOAF liberális felfogással közelíti meg az adatcserét. Nem szükséges, hogy bármit is mondjunk magunkról, vagy másokról, de felülről sem korlátoz-

za a leírható információ mennyiségét, vagy azt, hogy milyen szókészleteket használhatunk fel közben.

A FOAF olyan definíciók összessége, melyek segítségével a világról szóló állításokat fejezhetünk ki. A szókészlet megalkotásakor főképp az emberek leírására összpontosítottak, hiszen az emberek kapcsolnak össze szinte minden más dolgot, amit a web leír: dokumentumokat készítenek, találkozókon vesznek részt, fényképeken szerepelnek, stb.

Ám a FOAF szókészlet szabvány nem az ISO szabványosítás értelmében vett szabvány, és nem áll kapcsolatban a W3C tevékenységével. Nagyban épül a W3C szabványaira, kiváltképp az XML-re, XML névterekre, RDF-re és OWL-re. Minden FOAF dokumentumnak jól-formázott RDF/XML dokumentumoknak kell lennie. A FOAF projekt menedzselése inkább hasonlít egy nyílt forrású szoftverprojektére, mint egy ipari szabványosítási törekvésre.

A szókészlet tartalmaz

- osztályokat
  - `foaf:Person`, `foaf:Document`, `foaf:Image`, `foaf:Project`, `foaf:Group`, `foaf:Organization`, ...
- osztályokat leíró tulajdonságokat
  - `foaf:name`, `foaf:mbox`, `foaf:homepage`, ...
- osztályok között fennálló kapcsolatokat leíró tulajdonságokat
  - `foaf:interest`, `foaf:knows`, `foaf:member`, ...

A osztályok és tulajdonságok között fennállnak értelmezési tartománybeli, illetve értékkészletbeli viszonyok. Például a `foaf:Person` osztály értelmezési tartománya a `foaf:knows` tulajdonságnak, ugyanakkor értékkészlete a `foaf:firstName`, `foaf:surname`, `foaf:interest`, `foaf:knows` és még számos más tulajdonságnak.

Példa egy személy leírására:

```
<foaf:Person rdf:about="#me"
             xmlns:foaf="http://xmlns.com/foaf/0.1/">
  <foaf:nick>Szaki</foaf:nick>
  <foaf:givenname>János</foaf:givenname>
  <foaf:family_name>Szakácsi</foaf:family_name>
  <foaf:gender>male</foaf:gender>
  <foaf:mbox>szaxxi@freemail.hu</foaf:mbox>
  <foaf:homepage rdf:resource="http://example.com/~me" />
  <foaf:img rdf:resource="/images/szaki.jpg" />
</foaf:Person>
```

Egy HTML dokumentumhoz a fejrészbe illesztett link segítségével csatolható FOAF dokumentum, ahogyan az alábbi minta mutatja:

```
<head>
  ...
  <link rel="meta" type="application/rdf+xml" title="FOAF"
    href="http://example.com/~me/foaf.rdf"/>
  ...
</head>
```

3.2. DOAP (Description of a Project)

A DOAP arra igyekszik megoldást adni, hogy minél pontosabban, részletesebben legyen képes információkat szolgáltatni szoftverprojektekről. A megalkotásakor a legfontosabb kitűzött követelmények az alábbiak voltak:

- A lehető legáltalánosabb módon leírható legyen egy szoftverprojekt és minden hozzákapcsolódó erőforrás, beleértve a projekt résztvevőit és webes erőforrásait is.
- Alapvető eszközök kifejlesztése, melyek könnyűvé teszik a leírások létrehozását és felhasználását

- Együttműködő-képesség más népszerű webes metaadat projektekkel (pl. RSS, FOAF, Dublin Core).
- Lehetőség a szókészlet bővítésére speciális szakterületek céljai esetén.

3.2.1. Egyedi projekt azonosítók

Ahhoz, hogy a szókészlettel leírt adatok hatékonyan kezelhetők legyenek, szükség van legalább egy kitüntetett tulajdonságra, amely egyedien azonosítja a projekteket. Ez analóg az adatbázisbeli elsődleges kulcs fogalmával. Ám a weben való megoszthatóság miatt a kulcsnak nem csak lokálisan, hanem globálisan is egyedinek kell lennie. Tekintve, hogy a DOAP egyik alapelve a decentralizáltság, nem oldható meg a probléma egy kitüntetett webhelyen való regisztrációval, sem hasonlóval.

A weben a szokásos módja a globális azonosításnak az URI-k használata. Mivel minden szoftverprojekt rendelkezik honlappal, ésszerű megoldás lehetne a honlap URI-ját választani azonosítóként. Az egyetlen másik lehetőség a projektnévvel történő azonosítás volna, de nem ritkán fordul elő, hogy több projektet ugyanazzal a névvel látnak el. Ezzel szemben a honlap URI-k (vagyis URL-ek) esetén a globális hatóságok biztosítják, hogy ne legyenek névütközések.

Ugyanakkor a honlap URL-eknek van egy nyilvánvaló hátránya is. Egy ideális világban az URI-k változatlanok, de a valóságban sajnos nem, hiszen egyik Internet szolgáltató jön a másik után, a hostnevek változnak, a honlapok átszerveződnek.

Erre a problémára bevezették a „régi honlap” tulajdonságot, mely többszörösen is szerepelhet egy projekt leírásban és bármikor hozzá lehet adni egy új „régi honlap” tulajdonságot. Az egyedüli megszorítás az, hogy a régi honlap címeket soha többé nem használhatja más projekt.

3.2.2. Tulajdonságértékek korlátozása

Az egyedi projekt azonosítókon kívül más területen is vetődik fel probléma a leírások készítésekor. A globális feldolgozhatóság érdekében az tulajdonságok értékkészletét valamilyen módon előre meg kell határozni. Erre a célra egy megoldás az, ha definiálunk egy W3C XML Séma felsorolást, amelyben szereplő sztring literálok szolgáltatják megengedett értékek halmazát. Ennek a megoldásnak az a hátránya, hogy többletterhet ró ki a DOAP fájlok feldolgozásánál, hiszen szükség van egy W3C XML Séma validációra, és még utána is csak egy sztring a végeredmény, amit extra információval kell kiegészíteni az emberi olvashatóság végett. Emellett a DOAP szókészlek karbantartójára is több feladat hárul, mert a sémára is külön figyelmet kell fordítani.

Flexibilisebbé tehető a leírás, ha a literálok helyett erőforrásokat használunk, amit úgy érhetünk el, hogy saját URI-kat allokálunk a bázis-URI által meghatározott névtérben. Az URI hivatkozások mutathatnak egy-egy weboldalra is, melyek további információkat írnak le az adott erőforrásról. Emellett a bázis-URI helyén közzétehető az erőforrások teljes listája – akár egy mellékelt RDF fájllal együtt, mely a gépi feldolgozást segíti címkékkel, leírásokkal és egyéb hasznos adatokkal.

Ezzel a technikával egyszerűen megoldódik a bővíthetőség problémája is, hiszen új erőforrások felvételénél mindössze az erőforráshoz tartozó URI érvényességét kell biztosítani.

Van ugyanakkor két hátránya is. Az egyik az, hogy sokkal egyszerűbb egy rövid sztring literált begépelni, mint a teljes URI-t. Ám ez nem olyan nagy probléma és valamelyest enyhíthető szintaxisbeli rövidítésekkel, vagy RDF label-ök használatával.

A másik gond teljesen jogosan az, hogy megszűnhet az adott URI által hivatkozott névtér fölötti felügyelet. Rövid távon az URI-k még használhatók maradhatnak, mielőtt érvénytelenné válnak, az idő múlásával viszont valószínűleg az URI-k által hivatkozott tartalom megváltozik, vagy törlődik. Ennek megoldására manapság két bevett szokás létezik:

- egy szolgáltatás igénybevétele, mint a purl.org[1], mely valamelyest garantálja a regisztrált URI-k hosszú élettartamát;
- URL-ek helyett URN-ek alkalmazása.

Az URN-ek egy felügyelt névteret biztosítanak az Internet Assigned Numbers Authority (IANA) által. Az URN névtér DOAP számára kiosztott részének igazgatása az Internet Engineering Task Force (IETF) számára benyújtott dokumentumokkal lehetséges. Ám ez a folyamat meglehetősen lomha, emiatt általában nem használható projektleírások esetén.

Tehát nem mindig a legjobb megoldás a konstansok erőforrásokként való reprezentálása. Erőforrásokat akkor érdemes alkalmazni, ha a későbbi bővítés lehetőségén van a hangsúly. Más esetekben viszont kényelmesebb egyszerű sztring literálok használata.

Végül egy példa, mely magáról a DOAP projektről ad minimális leírást:

```
<Project xmlns="http://usefulinc.com/ns/doap#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:foaf="http://xmlns.com/foaf/0.1/">
  <name>DOAP</name>
  <homepage rdf:resource="http://usefulinc.com/doap" />
  <created>2004-05-04</created>
  <shortdesc xml:lang="hu">
    Közösségalapú szoftverprojektek leírására szolgáló
    szókészlet és eszközök együttese.
  </shortdesc>
  <description xml:lang="hu">
    A DOAP (Description of a Project) egy olyan RDF szókészlet
    és különbözö eszközök együttese, mely segítségével
    közösségalapú szoftverprojektek írhatók le. A célja, hogy
    lehetövé tegye szoftvertárházak közötti adatok cseréjét és
```

[1]Persistent URL: a domain nevekhez hasonlóan biztosít egy URL-t, mely nem változik, ám egy rá történtő hivatkozás esetén átirányítja az éppen aktuális URL-re

```
    a projektekröl szóló információk decentralizált tárolását.
  </description>
  <maintainer>
    <foaf:Person>
      <foaf:name>Edd Dumbill</foaf:name>
      <foaf:homepage rdf:resource="http://usefulinc.com/edd" />
    </foaf:Person>
  </maintainer>
</Project>
```

3.3. SIOC (Semantically-Interlinked Online Communities)

Az online közösségi webhelyek lehetővé tették, hogy a közösség tagjai sokkal hatékonyabban kommunikálhassanak egymással, akár interaktív módon is. A tagok kereshetnek és közzétehetnek mások számára a tárgyhoz kapcsolódó információkat, ötleteket. A közösségi oldalak egyaránt használhatók profit és nonprofit célok esetében is.

A Szemantikus Web technológiával tovább gazdagíthatók a közösségi webhelyek által nyújtott szolgáltatások, és lehetőség nyílik metaadatok hozzárendelésére emberek és szoftverek számára egyaránt. A legtöbb közösségi oldal egy személyre szabott felhasználói felületet alkalmaz az információk feldolgozására és megosztására a tagok között, emellett általában kulcsszavas keresőrendszerrel rendelkeznek. Így jelentősen limitálták az információk elérhetővé tételének hatékonyságát.

A korábbi technológiák korlátai miatt az online közösségek csak szigeteket alkotnak, mert nincsenek összekapcsolva. Például adott egy fórum hozzászólás az A közösségi webhelyen, amely a B levelezőlista egy emailjére utal, de egy fórumbeli keresés ezt nem képes megjeleníteni. Ahogy gyarapodnak a közösségi oldalak és az általuk nyújtott lehetőségek, ezek a különböző webhelyek összekapcsolhatók (a példában az A oldalon található hozzászólásban elhelyezhető egy valódi hivatkozás a B levelezőlista emailjére). Ugyanakkor ha egy felhasználó az A helyen már rendelkezik egy felhasználói fiókkal, a B webhely lekérdezheti az A-ról, hogy elfogadja-e az adott bejelentkezési adatokat, így nem szükséges saját felhasználói fiók adatbázist karbantartania.

Ha a közösségekkel összefüggő adatok elérhetők egy egységes, struktúrált formátumban (mint az RDF), az olyan egyéb hasznos lehetőségeket rejt magában, mint

- következtetések végzése;
- egyének, vagy emberek csoportjainak együttes időbeosztásának vizualizációja;
- több közösségi hely keresési lehetőségeinek integrációja;
- egy adott témához kapcsolódó személyek földrajzi helyzetének egyidejű megjelenítése.

Szerepüket tekintve az alábbi három nagy csoportba sorolhatók az online közösségek:

- az információszolgáltatók, akik kibocsátják az információkat;
- az információ fogyaszók, akik fogyasztják az információkat;
- és az infrastruktúra szolgáltatók, melyek az információk közzétételéhez és cseréjéhez biztosítanak intrastruktúrát.

A Szemantikus Weben az információk RDF-ben való elérhetőségéhez arra van szükség, hogy az infrastruktúra szolgáltatók – mint a webes rendszergazdák, levelezőlista adminisztrátorok – elérhetővé tegyék az adatbázisaikat

a szemantikus webes alkalmazások számára. Az alapvető technológiák már adottak, mindössze az embereket kell meggyőzni, hogy átvegyék őket.

A napjaink online közösségei által használt eszközök közé tartoznak:

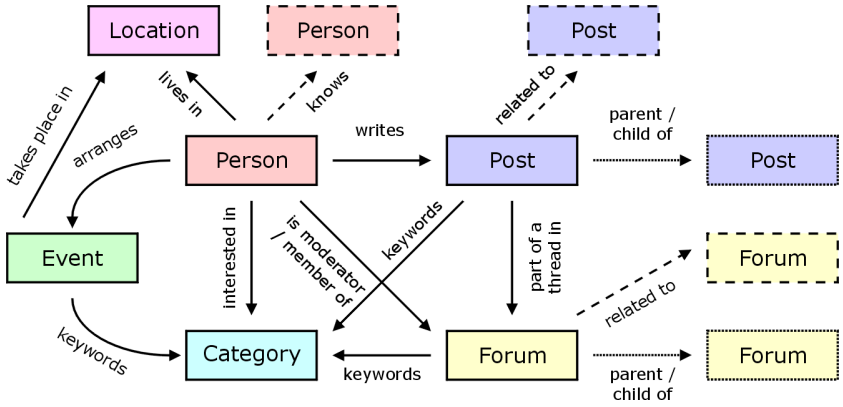
- A levelezőlisták, melyek technológiája az Internet mai napig is legelterjedtebb aszinkron kommunikációs formája.
- A Usenet hírcsoportok, amelyek a közösségszervező módszerek közül a legrégebbiek.
- Az elektronikus hírdetőtáblák – népszerűségben a levelezőlisták és fórumok után következő megoldás, mely általában kategorizált fórumok halmazából áll.
- A csevegés, mely szinkron kommunikáció tesz lehetővé.
- A weblogok, melyeket a létrehozóik bővítenek időről időre új információkkal.
- A wikik, amelyek közösen szerkesztett weboldalak, vagyis lehetővé teszik egy közösségnek a weboldalak egy szabadon kialaklított rendszerének létrehozását.

3.3.1. A SIOC ontológia

A fent említett eszközök már eleve részeik a HTML webnek. A weblog szoftverek már rendelkeznek RDF-be való exportálási funkcióval, főleg abból a célból, hogy a blogok és blog elemek leírhatók legyenek olyan egyszerű szókészletek használatával, mint az RSS és a Dublin Core. Így a weblog elemek már képesek hivatkozni egymásra a HTML szintjén.

A SIOC egy olyan ontológia, amely képes adatokat befogadni a korábban említett közösségek által használt kommunikációs eszközökből. A legtöbb közülük valamilyen részben struktúrált formátumban tárolja az adatokat, a bonyolultabbak – mint hírdetőtáblák, weblogok, wikik – pedig relációs adatbázist használnak. A SIOC célja, hogy olyan széles körben legyen képes

lefedni az információk halmazát, amennyire csak lehet, ám eközben mégis egyszerű maradjon az ontológia és a felhasználók könnyedén eligazodjanak benne. A 3.3.1. ábra egy áttekintést ad az ontológiáról.



3.1. ábra. A SIOC ontológia főbb osztályai között fennálló kapcsolatok

Felmerül egy kérdés, hogy célszerűbb-e meglévő ontológiákat felhasználni, avagy leképezéseket alkalmazni egy teljesen új ontológiára, mely maga után vonja intelligensebb szoftverek szükségességét. Ha adott egy leképezés, nagyobb a flexibilitás, de a leképezés végrehajtásához algoritmusok kellenek és az adatokat egyik formátumból a másikba kell konvertálni. Az ontológiák között az `owl:equivalentProperty` és `owl:equivalentClass` attribútumok segítségével lehet megvalósítani a leképezéseket. A probléma a különböző leképezések végrehajtási hatékonyságának kérdésében rejlik.

Ahol csak lehet, a SIOC a már létező ontológiák kijelentéseit alkalmazza, mert igen költségesek a leképezések és a meglévő kijelentések használata közelebb visz a webhez. A SIOC ontológia a FOAF, DC és RSS 1.0 kijelentéseivel dolgozik, illetve néhány újjal, melyek lehetővé teszik a korábban említett közösségi eszközök által exportált adatokhoz való hozzáférést. A legfőbb fogalmak közé a személy, a fórum, a hozzászólás, a kategória, a helyszín és az esemény tartozik. Lássuk ezeket kicsit részletesebben!

A sioc:Person osztály

Egy elektronikus hírdetőtábla alapú közösségben egy felhasználó profiladatainak begyűjtése elsősorban már a regisztrációkor megtörténik, mielőtt a felhasználó aktiválja a fiókját. A kötelező mezők között szerepel általában a név, email cím, érdeklődési kör, foglalkozás, stb., melyek egy FOAF fájl alapját képezhetik egy-egy felhasználó számára, amennyiben hozzájárultak az adataik nyilvánossá tételéhez. A felhasználók kialakíthatnak maguknak egy listát a barátaikról, mely segítségével nyomon követhetik, hogy mikor online-ak, vagy mindegyiküknek egyszerre küldhetnek üzenetet. Ezek a listák általában privátak, de egy opció hozzávételével a lista egy része is publikussá tehető, mely szintén exportálható a FOAF fájl részeként.

A személyek adatai származhatnak többféle helyről és formában, de a SIOC főként a FOAF szókészletet használja a leírásukra.

A sioc:Post osztály

A Post osztályba sorolható egy cikk, egy email, egy audio-, vagy videorészlet, egy online elérhető dokumentum, és így tovább. Habár manapság az RSS a legkisebb és leggyakrabban használt eszköz ezek közzétételére, szükség van bonyolultabb leírásokra a hozzászólások egymás közötti cseréjéhez. A `sioc:Post` több más szókészlet hasonló koncepcióját foglalja össze.

Egy `sioc:Post` azonosítható egy URI vagy egy üzenet ID segítségével, de alapvetően az URI-t ajánlják.

A sioc:Forum osztály

A SIOC fórum fogalmába tartoznak az elektronikus hírdetőtáblák, Usenet hírcsoportok, levelezőlisták, IRC botok, weblogok és minden más olyan eszköz, melyekben a hozzászólások generálódnak. Ezeket összefoglalóan fórum konténereknek nevezzük.

A fórum osztály célja, hogy kiegészítő információkat szolgáltasson a hozzászólásokról. Kapcsolódhatnak hozzájuk személyek, mint adminisztrátorok,

moderátorok, vagy különböző jogokkal rendelkező tagok. A felhasználók hivatkozhatnak egy fórumra, de fordított hivatkozási lehetőség is létezhet – például egy privát fórum tagjainak hitelesítéséhez.

A fórumok általában egy bizonyos témáról szólnak, mely a `dc:subject` tulajdonsággal fejezhető ki.

A sioc:Event osztály

Bár egy hozzászólás alkalmas egy esemény kihírdetésére, alapesetben nem tartalmaz olyan gép által feldolgozható formájú struktúrált információkat, melyekből felismerhető lenne például az eseménye időpontja, időtartama, vagy helyszíne.

Számos közösség középpontjában állnak találkozók, konferenciák és más események. A meghívókat, bejelentéseket általában email formájában szórják szét vagy egy webhelyen teszik közzé. Az egyének napirend szervező eszközök segítségével alakítják ki az időbeosztásukat.

A `sioc:Event` osztállyal lehetőség nyílik az események mindenféle információinak egybekötésére. Az ütemező alkalmazások felhasználhatják az események leírásait, hogy beosztást készítsenek találkozókról, vagy közzétegyék egy konferencia résztvevőinek listáját.

A sioc:Location osztály

A SIOC a `geo`[2] és a `wail`[3] egy kombinációját használja helyzetmeghatározáshoz. Az ontológiában a helyszínekhez név, illetve földrajzi hosszúság és szélesség tulajdonságokat rendelnek. A helyszín fontos osztály, hiszen az online közösségek általában fölrajzi területek alapján csoportosulnak. A helyszín osztály példányaira a helyszín URI-jának segítségével lehet hivatkozni.

[2]Basic Geo – egy alapvető RDF szókészlet, mely földrajzi elhelyezkedésbeli információkat képes leírni (pl. szélesség, hosszúság, tengerszint feletti magasság, stb.)

[3]Where Am I Language – egy igen egyszerű módja a helyszínek leírásának, mely elsősorban az egymáshoz viszonyított helyzeten alapul

A sioc:Category osztály

A kategóriákat leíró információk többnyire természetes nyelvű kulcsszavak. Az alkalmazásuk hatékony a felhasználók azon zártabb körű csoportjain, melyekben azonos nyelvet beszélnek és a fogalmak meghatározására ugyanazokat a szavakat használják, ám globális kiterjedésű közösségek esetén már gondok lehetnek.

A természetes nyelv feldolgozása nehéz feladat. A tárgyak azonosítására használt kifejezések között meg kell szüntetni a kétértelműséget, illetve le kell fordítani a nevüket más nyelvekre. Bár a SIOC az URI-kat javasolja a kategória információk leírására, de a kulcsszavak is megengedettek. Tetszés szerinti URI-k használhatók az azonosításra, így a feldolgozás is könnyebbé válik, mintha sztringek illesztését kell végezni.

3.3.2. Az adatok integrációja

Az ontológia felvázolása után lássuk azt az architektúrát, mely lehetővé teszi az információk elérhetőségét az egymáshoz kapcsolt közösségi helyek között!

Hivatkozások

A SIOC nagy előnye, hogy gyakorlatilag bármire képes hivatkozni egy közösségi oldalon. A hivatkozások manuális létrehozásával lehetővé válik gép által feldolgozható módon is ezen oldalak összekötése, és kialakulnak a közösségi helyek hálózatai.

Webhelyek

A SIOC kibővíti a webhely fogalmát. Webhely alatt ért minden fajta korábban említett közösségi eszközt. Ha a webhelyek implementálnak egy közös, általános interfészt, mely megengedi az általuk szolgáltatott információ megosztását, egy hálózatba integrálhatók. Az egyébként lényegtelen, hogy az adatok honnan származnak – a forrásuk lehet egy ilyen hálózatbeli webhely, egy internetes webszolgáltatás, stb.

Adattárházak

Vannak olyan webhelyek, amelyek „nyers" adatokat tesznek elérhetővé. Tegyük fel, hogy ezek az adatok vagy már eleve SIOC formátumban vannak leírva, vagy létezik olyan SIOC leképezés, mellyel átkonvertálhatóak. Egy lehetséges megoldás az ilyen webhelyek adatainak feldolgozására az, ha egy helyi adatbázisba időről-időre lekérdezzük őket. Természetesen importálás közben el kell végezni az esetleges sémakonverziókat.

Az adattárházak hátulütője, hogy az adatok lemásolva, két helyen vannak eltárolva. Az efféle másolatkészítés gyakori napjaink web intrastruktúrájában – ezt teszik például a levél archiváló rendszerek, keresőmotorok, stb. is. Ugyanakkor az időszakos lekérdezések miatt az adatok valószínűleg nem a legfrissebbek. Ha explicit módon nem elérhető az utolsó frissítés dátuma, a változások nyomon követése csak egy költséges fájlösszehasonlítással oldható meg.

A harmadik hátrány az, hogy bizonyos esetekben óriási tárterület és számítási kapacitás szükséges az összes begyűjtött adat számára.

Virtuális integráció

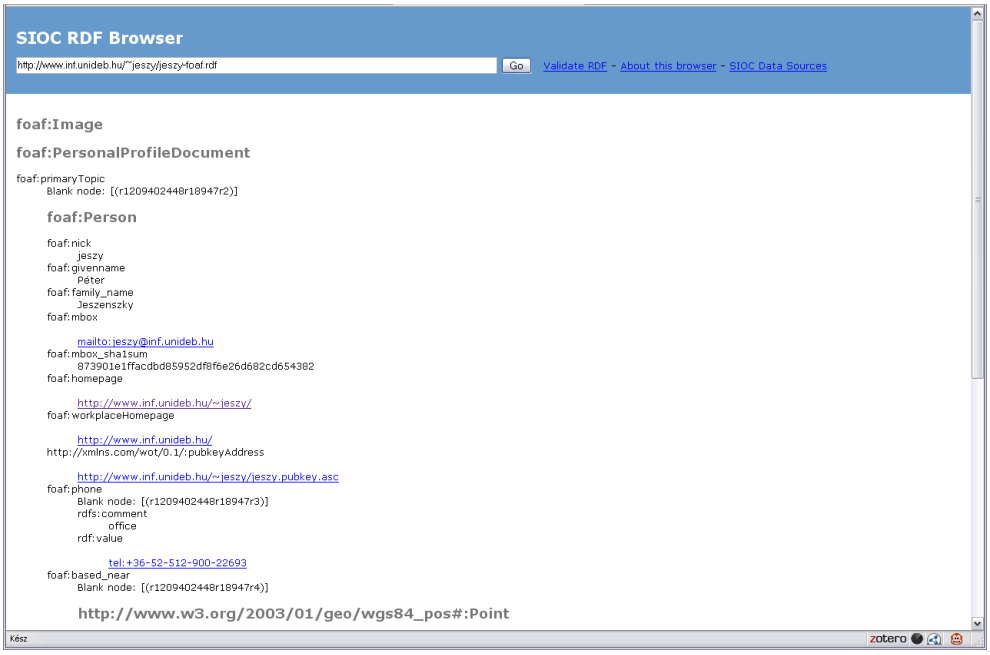
Az adattárházak problémái valamelyest kiküszöbölhetők ún. virtuális integráció alkalmazásával, melynek működése a hagyományos adatbázisok nézeteivel hasonlatos. Az adatbázis csak szükség esetén – az igények érkezésekor – kérdezi le az adatokat a megfelelő helyekről. Minden kellő transzformációt (sémaleképezés, stb.) e folyamat közben végez el és csak ezután küldi vissza az eredményt.

Ám ennek a megoldásnak is vannak hátrányai. A hatékonyság egyrészt függ a kívánt adatok eloszlásától (hány ugrásra van, hány lépésben kérdezhető le), másrészt a különböző szókészletek közötti sémaleképező algoritmusok hatékonyságától, hiszen ez döntően befolyásolja a rendszer gyorsaságát. Elosztott csomópontok és egy peer-to-peer SIOC rendszer használata esetén útvonalválasztásra is szükség van. Ezzel felvetődik a kérdés, hogy milyen algoritmus alapján történjen az útvonalak kiválasztása és mikor álljon le

(hurkok elkerülése végett).

3.4. Semantic Radar

A Semantic Radar egy Firefox kiegészítő, mely automatikusan detektálja az aktuális weboldalahoz csatolt RDF metaadatfájlokat. A támogatott formátumai közé tartozik a FOAF, a DOAP és a SIOC. Ha ezek közül észleli bármelyiket, megjeleníti az állapotsoron egy megfelelő ikonnal, melyre rákattintva megtekinthetőek az adatok.



3.2. ábra. Kattintás után formázva megjelenik a FOAF fájl tartalma

4. fejezet

„Nyers” HTML fájlok feldolgozása

Ebben a fejezetben olyan eszközök kerülnek bemutatásra, melyek tetszőleges, ún. „nyers” HTML oldalakból nyerik ki az információkat.

4.1. Piggy Bank

Bár a Szemantikus Web sokkal hatékonyabbnak ígérkezik az információelérés terén, felmerül a tyúk vagy a tojás problémája. Mivel még nem áll rendelkezésre tekintélyes mennyiségű adat szemantikus formában, a felhasználók nem sok hasznot húznak az olyan eszközökből, melyek közvetlenül metaadatokkal dolgoznak, nem pedig weboldalakkal. Ugyanakkor a Szemantikus Web alapú szoftverek kevésbé tudják bizonyítani hatékonyságukat az adatok hiányában. Ilyen eszközök nélkül pedig az emberek továbbra is a már létező böngészőket használják információkeresésre. Ezáltal a tartalomszolgáltatók sem igyekeznek azon, hogy az információkat közzétegyék szemantikus formában.

A *Piggy Bank* egy olyan kiegészítője a Firefox böngészőnek, mely lehetővé teszi a felhasználók számára információk kiemelését a weboldalakból és RDF formátumban való elmentését metaadatokkal együtt, majd az adatok használatát ugyanazon böngészőn belül. A különböző oldalakról összegyűjtött adatok böngészhetők, kereshetők, osztályokba rendszerezhetők és egymáshoz rendelhetők attól függetlenül, hogy milyen típusúak, illetve honnan származnak. A Piggy Bank olyan új, hasznos lehetőségeket nyújt a Web felhasználóinak mindennapos munkájához, amelyek használata nem kíván komoly erőfeszítést.

Az információk begyűjtését kétféleképpen végzi a Piggy Bank. Egyrészt a weboldalakhoz csatolt RDF adatfájlokból, másrészt – ha ilyet nem talál – *screen scraper*eket alkalmaz. Emellett lehetőség van *szemantikus bank*ok létrehozására, melyek olyan közös RDF adattárak, ahol Piggy Bank felhasználók csoportjai megoszthatják egymással az általuk összegyűjtött információkat. A Piggy Bank és Semantic Bank együttesen segít meggyorsítani a Szemantikus Web felhasználói körének gyarapodását, hiszen a hagyományos Web felhasználói úgy térhetnek rá át, hogy közben nem szükséges lecserélniük egy jól megszokott böngészőt.

4.1.1. Funkcionalitás

Adatgyűjtés

A Piggy Bank alapvető ötlete az volt, hogy információt gyűjthessen különböző weboldalakról struktúrált formában, majd ezeket tetszőlegesen felhasználhatóvá tegye a kliens oldalon. Az adatgyűjtésben kétféle stratégiát különböztet meg: a tartalomszolgáltatók hozzájárulásával és a tartalomszolgáltatók segítsége nélkül. Ha a szolgáltató egy HTML oldal tartalmát RDF-ben is hozzácsatolja az oldalhoz, akkor a Piggy Bank egyszerűen csak azzal foglalkozik. Ellenkező esetben megpróbálja screen scraperek használatával kinyerni és újrastruktúrálni a HTML-ben található információkat.

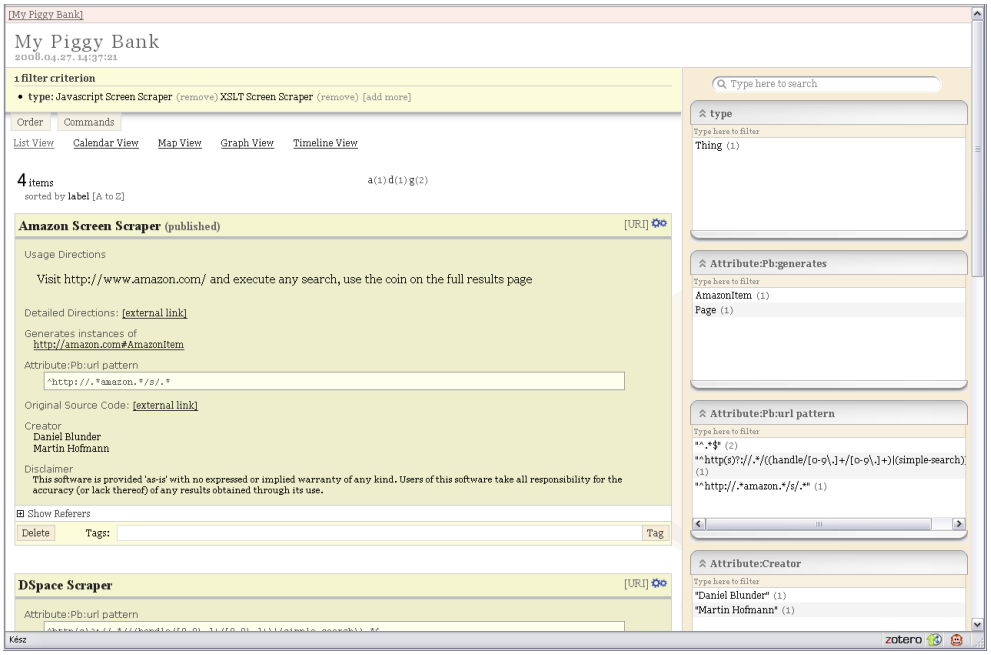
A két módszer együttes alkalmazása egyrészt kedvező a tartalomszolgál-

tatók számára, amennyiben RDF-ben teszik közzé az információikat, másrészt nem okoz problémát, ha mégsem teszik. Ugyanakkor ösztönzőleg hat a szolgáltatókra, hiszen RDF használata esetén maguk befolyásolhatják azt a formát, ahogyan megjelennek a közzétett adatok a Piggy Bank felhasználók előtt, illetve versenybeli előnyhöz juthatnak más szolgáltatókkal szemben.

Mentés

A Piggy Bank az összes forrásból származó információelemet egy ideiglenes adatbázisban tárolja el, amelyet egy társzemétgyűjtő tart karban. Ha a felhasználó elment egy letöltött elemet, átmásolásra kerül a permanens „My Piggy Bank" adatbázisba.

Egy lehetséges alternatív implementációban a letöltött elemek már eleve bekerülhetnek a permanens adatbázisba, és egy flag jelezné, ha a felhasználó explicit módon is elmentet egy-egy adott elemet. Ennek a megoldásnak jelentős tárigénye van. Egy másik alternatíva az, ha csak a hivatkozások



4.1. ábra. Screen scraperek a My Piggy Bank adatbázisban

lesznek eltárolva, és csak szükség esetén töltődnek le az általuk hivatkozott információk. Az utóbbi megoldás viszont időigényes, és bár az „elmentett" elemek mindig naprakészek, el is veszhetnek. A Piggy Bank ezért a közbülső utat választotta.

Rendszerezés

Minden egyes információelemet elláthat a felhasználó *címkék*kel (kulcsszavakkal), ezáltal egyidejűleg többféle rendszerezési sémába is besorolhatja őket. A Piggy Bank fejlesztésekor a címkéző módszer mellett felmerült a könyvtár-hierarchiák alkalmazása is, ám ez többletmunkát jelentene létrehozáskor és karbantartáskor, ráadásul ugyanazokon az adatokon nem tenné lehetővé különböző rendszerezési sémák egyidejű használatát.

A kulcsszavak beírását legördülő listákban megjelenített javaslatokkal segíti elő a Piggy Bank, így a használata viszonylag egyszerű. A címkék modellezése RDF erőforrásokkal történik – erről a későbbiekben lesz szó.

Megjelenítés

Az elmentett információk megjelenítéséhez a Piggy Bank-nek a nyers adatokat át kell transzformálni olvasható – vagy scraping esetén eredeti – formába. Mivel a felhasználók tetszőleges információkat gyűjthetnek be, nem lehet előre tudni, hogy ezeket milyen névterekben és ontológiákban kell értelmezni. Emiatt jelenleg minden információelem általánosan, táblázatos formában tekinthető meg tulajdonság-érték párokként. Habár vannak olyan fejlesztési tervek, melyek megengedik a felhasználók számára, hogy saját sablonok alapján jeleníthessék meg az adatokat.

Böngészés, keresés

A begyűjtött információk névtereit jellemző tulajdonságok hiányában nehéz a böngészést megvalósítani. Különösképpen akkor, ha az adatok heterogének és nem azonos ontológiában vannak leírva. Ezek az információk természetükből fakadóan több *facet*-tel rendelkeznek (többféle tulajdonság szerint

csoportosíthatók), ezért a Piggy Bank ún. *faceted* böngészői felületet biztosít, mely segítségével a felhasználók részhalmazaira szűkíthetik a teljes információ halmazt a facet-ek alapján.

Mindemellett a Piggy Bank megtartja a Web navigációs paradigmáját, vagyis az információkat URL-ekkel ellátott oldalakon jeleníti meg, melyeket a felhasználó elmenthet könyvjelzőként, és a megszokott módon használhatja a böngésző előre és vissza gombját a navigáláshoz.

Megosztás

Az webes információk begyűjtése mellett lehetőség van arra is, hogy a felhasználók megoszthassák egymás között ezeket az RDF-ben tárolt adatokat. A Piggy Bank ismét az egyszerű kezelhetőségre törekedett: a felhasználó egyetlen kattintással közzétehet egy információelemet. A közzététel a már említett szemantikus bankok segítségével történik, vagyis a kattintás után az elem és annak tulajdonságai feltöltődnek azon szemantikus bankokra, melyekre a felhasználó feliratkozott. Igaz, hogy ezzel a módszerrel nem lehet szabályozni, pontosan mely RDF állítások lesznek elküldve, mert a rendszer implicit módon az elemből induló teljes RDF részgráfot fogja továbbítani. Ez az ára az egyszerűségnek.

Egy szemantikus bankba feltöltött elemek ugyan összevegyülnek, ám minden egyes elem meg van jelölve, hogy kitől származik. Ez egyrészt egy facette lesz az információknak, másrészt visszakereshető a forrásuk, ha például több/bővebb információkra van szükség tőlük. Ugyanakkor olyan célt is szolgálhat, hogy ki lehessen jelölni megbízható forrásokat.

Együttműködés

Amikor egy információelem bekerül egy szemantikus bankba, a forrás által hozzárendelt címkéket is eltárolja a rendszer vele együtt. Tehát a bank felhasználói nem csak információkat gyűjtenek össze, hanem a rájuk alkalmazott rendezési sémákat is.

Manapság a kulcsszavak használata számos szolgáltatásban vált közkedveltté, mert a segítségükkel egy közösség könnyedén felépíthet együttműködve egy közös taxonómiát (osztályozási rendszert) az általuk megosztott adatokra. Ez a stratégia megspórolja egy előre kialakított taxonómia tervezési költségeit, ráadásul kezdetben valószínűleg nem ismert minden ehhez szükséges információ. Emellett a kulcsszavas taxonómia természeténél fogva dinamikusan fejlődik, követve a folyamatos bővüléssel járó változásokat. Az ilyen kulcsszavas stratégiát alkalmazó rendszereket együttesen *folkszonómiák*nak[1] nevezzük.

Egy másik igen hasznos tulajdonsága az említett stratégiának az, hogy a felhasználók együttműködése valószínűleg nem szándékos, hanem inkább véletlenszerű. Egy felhasználó valószínűleg olyan kulcsszavakat használ, mely a saját rendszerezési céljait szolgálja, vagy esetleg azt próbálja elérni velük, hogy egy ismerőse könnyen rátaláljon az általa megosztott információra. Így vagy úgy, de a kulcsszavai automatikusan elősegítik a teljes információkészlet mintázatainak kialakulását. A Piggy Bank az egy-kattintásos megosztási módszerével lehetővé teszi a folkszonómiák alkalmazását, akár szándékos, akár véletlenszerű az együttműködés.

Amíg a taxonómiák a dolgok neveivel dolgoznak, az ontológiák fogalmakkal és kapcsolatokkal. A Piggy Bank fejlesztői úgy gondolták, hogy tovább bővítik az RDF adta lehetőségeket, és a folksz*onómia* mellett bevezették az ún. folksz*ológia* (folk ontology) fogalmát. E cél érdekében a címkéket nem szöveges kulcsszavakként modellezik, hanem URI-kkal megnevezett RDF erőforrásokként, melyekhez RDF címkékkel rendelik hozzá a kulcsszavakat. Ezáltal lehetővé válik az annotálásuk.

Lássunk egy példát! Valaki az egyik receptjét megjelöli a „mangó” címkével, majd magát a „mangó”-t a „gyümölcs” címkével. Hasonlóképpen hozzárendelheti valaki egy utazási iroda ajánlatához a „Jamaica” kulcsszót, majd ehhez a „hely” kulcsszót. Ezek után kapcsolatba hozható a „gyümölcs” és a „hely” címke, vagyis kifejezhetők lesznek az olyan állítások, melyek szerint

[1] a szó eredete az angol *folk taxonomy* kifejezés

valamely „gyümölcs”-csel megjelölt dolog „megtalálható” egy bizonyos „hely”-en. (Itt a „megtalálható” címke már egy kapcsolat kifejezésére szolgál két másik címke között.)

A címkék RDF erőforrásokkal való reprezentálása segíti a folkszonómiák terjedését és fejlődését. A már létező szöveges kulcsszó alapú implementációkban problémát jelent az, ha ketten ugyanazt a kulcsszót használják különböző jelentéssel, mert így a megjelölt elemek azonos csoportba esnek. Szintén nem várt eredmény áll elő, ha két különböző kulcsszóval írják le ugyanazt a dolgot. Az előző két példa jól illusztrálja a szintaktikából adódó ütközéseket, melyek megszabják a címkékkel történő csoportosítás korlátait. Ám az RDF erőforrásokkal és címkékkel való modellezés megszűnteti ezt a korlátot. Két azonos kulcsszóval ellátott címke megkülönböztetése az `OWL:differentFrom`, illetve fordított esetben az `OWL:sameAs` tulajdonsággal oldható meg.

A Piggy Bank és Semantic Bank két különböző, ám megegyező címkével jelölt tag elemeit azonos csoportba sorolja. Bár a jelenlegi felhasználói interfész úgy működik, mint egy szöveges kulcsszó alapú implementáció esetén, de az adatmodell már RDF erőforrásokkal dolgozik, és a fejlesztők dolgoznak egy olyan felhasználóbarát felületen, mellyel ezt ki is lehet használni.

Bővítés

A Piggy Bank igen egyszerűen bővíthető új screen scraperekkel. A screen scraperek leírása szintén RDF-ben történik, mint bármilyen más információé. A telepítéshez mindössze a scraper metaadatait kell elmenteni – ugyanúgy, mint bármely más információelemet –, majd aktiválni. Az aktiválással fejezhető ki a Piggy Bank számára, hogy a screen scraper megbízható és a továbbiakban használhatja a rendszer. (Az elmentést követően alapértelmezésben egyetlen adatelem sem „megbízható”.)

4.1.2. Implementáció

Piggy Bank

Tekintve, hogy a Piggy Bank-hez szervesen kapcsolódik a böngészés, a fejlesztők úgy döntöttek, hogy nem egy teljesen új böngészőt írnak, hanem egy meglévő kiegészítőjeként implementálják a Piggy Bank-et. Így egyrészt igénybe vehetik a kiválasztott böngésző gazdag felhasználói felületét, másrészt a felhasználóknak sem kell átterniük és megszokniuk egy újat.

Ezek után egy olyan böngészőre volt szükség, amelybe integrálni lehet a meglévő Java alapú RDF hozzáférést és tárolást biztosító könyvtárakat, így a Firefox böngésző mellett döntöttek. Mivel a Java-t választották a Piggy Bank alapjául, sok egyéb Java-ban implementált funkció is elérhetővé vált, mint például az RSS feed-ek elemzése és az információelemek szöveges részeinek indexelése.

A Piggy Bank a minél könnyebb információgyűjtés érdekében egy állapotsor ikont használ, mellyel jelzi, ha talált kinyerhető információkat az aktuális weboldalon, de egyben egyetlen kattintással el is indítható vele a kinyerés. Az információk begyűjtése az alábbi három eljárás tetszőleges kombinációjával zajlik:

- Az aktuális oldalon található RDF/XML, N3 és RSS formátumú Webes erőforrásokra mutató linkek kinyerése, és az általuk hivatkozott elemek letöltése, majd RDF-be konvertálása.
- Az elérhető és az aktuális oldal DOM[2]-jára alkalmazható XSL transzformációk elvégzése.
- A szintén elérhető és az aktuális oldal DOM-jára alkalmazható Javascriptek lefuttatása, illetve az esetleges egyéb feldolgozandó weboldalak letöltése.

Miután a felhasználó rákattint az érme ikonra, a begyűjtött adatok megjelennek egy új oldalon. Mint már korábban szó volt róla, a Piggy Bank

[2]Document Object Model

igyekszik megtartani a web navigációs paradigmáját, és ennek érdekében URL-ekkel azonosított weboldalakon keresztül böngészhetők a tárolt információk. Ehhez a Piggy Bank-nek le kell generálnia a saját felhasználói interfészét DHTML-ben. Ráadásul ezt „röptében" kell biztosítania a dinamikusan begyűjtött és elmentett adatokra, ezért beépítettek egy szervletet, amely képest ezt megvalósítani.

Így tehát a Piggy Bank már nem más, mint egy három rétegű Java alapú webszerver alkalmazás, mely tartalmaz egy RDF adatbázist, egy modellező motort, egy DHTML felhasználói felületet, és mindezt beágyazva a Firefox böngészőbe.

Semantic Bank

A Semantic Bank architektúrája nagyon hasonló a Piggy Bank-éhez a Java-s rész tekintetében. Mindkettő DHTML-es böngészőfelületét ugyanaz a szervlet szolgálja ki. A Semantic Bank profilokat használ az adatmodellek elkülönítéséhez – minden feliratkozott felhasználó rendelkezik eggyel, ahol az általa feltöltött adatok találhatók meg, illetve van egy „közös" profil, ahová a felhasználók által publikussá tett adatok kerülnek vegyesen.

A Semantic Bank folyamatosan figyeli a felhasználók Piggy Bank-e által küldött HTTP POST kéréseket, melyekkel az adataikat töltik fel. Feltöltés után az információelemeket elmenti a felhasználó profiljába, illetve a publikusként megjelölt elemeket a közös profilba is. Az összes közös profilban található elemhez hozzá van rendelve a bank egy vagy több felhasználója, aki(k)től származik.

4.2. ClearForest Gnosis

Egy becslés szerint a weben szöveges formában megjelenő információk 80%-ára igaz, hogy természetes nyelven íródtak. Nincsenek struktúrálva, nincsenek szabványosítva, és a helyesírási hibák gyakorisága sem elhanyagolható – vagyis mindennapi „rendetlen" szöveg. A ClearForest az elmúlt évtizedben

kifejlesztett egy meglehetősen hatékony eszközt a természetes nyelvű szövegek feldolgozására. Eleinte csak nagy pénzügyi szervezetek, bankok, gyártók alkalmazták a technológiát, hogy jelentést nyerjenek ki a szövegekből. Ám a közelmúltban közzétettek egy Gnosis nevet viselő Firefox bővítményt, mely weboldalakkal is hasonlóképpen tud dolgozni.

4.2.1. SWS-1

Első lépésben az SWS-1 (Semantic Web Service) nevű webes szolgáltatást hozták létre, mely ráépül a természetes nyelvű szövegfeldolgozó eszköztárra. Bár a feldolgozási folyamat igen bonyolult, a funkciója egyszerű: szöveget információvá alakít. A szolgáltatás jelenleg képes felismerni a szövegben megemlített személyek, szervezetek, vállalatok és földrajzi helyek neveit, és mindezt jól struktúrált XML formátumban visszaadni. A lehetőségeket folyamatosan bővítik – a következő lépésben termék- és eseménydetektáló funkciók beépítését tervezik.

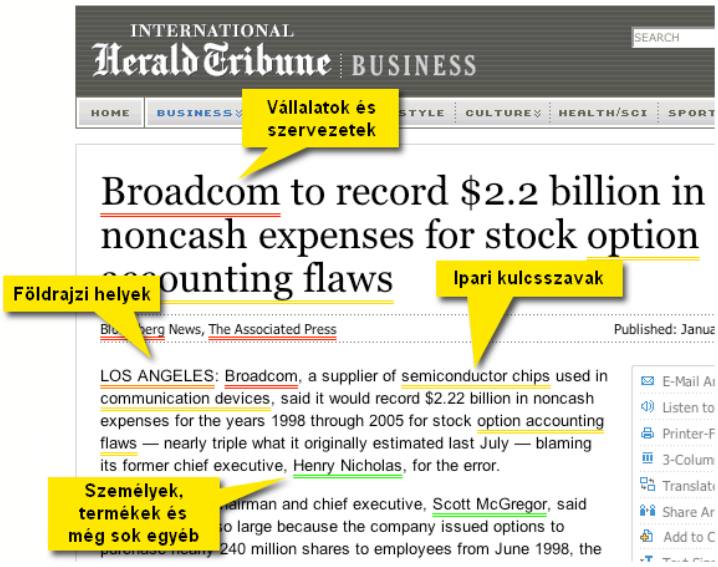
Az SWS-1 szolgáltatást ingyenessé tették, hogy bárki írhasson a saját ötlete alapján olyan programot vagy bővítményt, amely kihasználja a lehetőségeit.

4.2.2. Gnosis

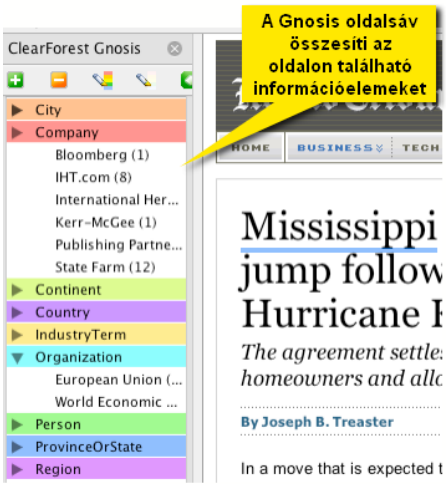
Időközben a cég is elkezdett egy szoftvert is fejleszteni, amely az SWS-1 segítségével könnyebbé és izgalmasabbá teheti a böngészést. A szoftvert egy Firefox bővítmény formájában valósítják meg, mely a Gnosis nevet kapta, és egy kísérleti verziója nemrégiben a nyilvánosság számára is elérhetővé vált.

Miután a bővítmény feltelepült és aktív, minden weboldal megnyitásakor valós időben feldolgozza azokat és listába rendszerezi a talált információelemeket. Ezután az elemeket egyrészt csoportosítva felsorolja egy oldalsávon, másrészt a típusoknak megfelelően különféle aláhúzási stílusokkal kiemeli a weboldalon. Ha egy aláhúzott szövegrészlet fölé visszük az egeret, megjelenik egy gyorsmenü, melyben keresési és egyéb funkciók közül lehet választani.

A Gnosis jelenleg a CNN, Wikipedia, Google Finance és hasonló tudástár, illetve hírközlő oldalakat támogat, de tetszőleges weboldalon is ki lehet próbálni, mire képes.



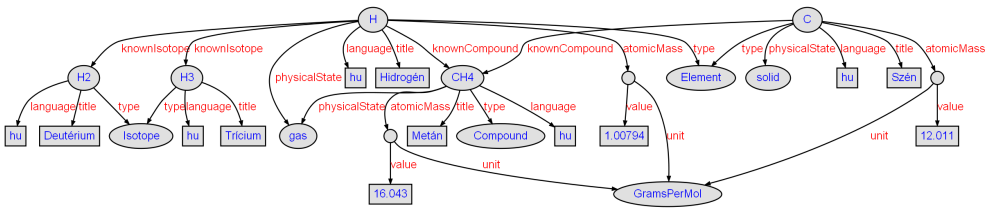
4.2. ábra. Egy Gnosis által feldolgozott weboldal



4.3. ábra. A Gnosis oldalsávja

A. függelék

Az 1.1. ábra rövidítéseihez tartozó teljes URI nevek



A.1. ábra. Kicsit más elrendezésben az 1.1. ábra

A csúcsok és élek címkéinek teljes neve:

- http://example/persys.owl#H
- http://example/persys.owl#H2
- http://example/persys.owl#H3
- http://example/persys.owl#C
- http://example/persys.owl#CH4
- http://example/persys.owl#Element
- http://example/persys.owl#Isotope

- http://example/persys.owl#Compound
- http://example/persys.owl#GramsPerMol
- http://example/persys.owl#knownIsotope
- http://example/persys.owl#knownCompound
- http://example/persys.owl#physicalState
- http://example/persys.owl#atomicMass
- http://example/persys.owl#unit
- http://www.w3.org/1999/02/22-rdf-syntax-ns#type
- http://www.w3.org/1999/02/22-rdf-syntax-ns#value
- http://purl.org/dc/elements/1.1/language
- http://purl.org/dc/elements/1.1/title

Irodalomjegyzék

[1] Resource Description Framework (RDF). http://www.w3.org/RDF/.

[2] Frank Manola, Eric Miller, editors. RDF Primer. http://www.w3.org/TR/rdf-primer/, 2004. W3C ajánlás.

[3] Dan Brickley, R.V. Guha, editors. RDF Vocabulary Description Language: RDF Schema. http://www.w3.org/TR/rdf-schema/, 2004. W3C ajánlás.

[4] RDFa. http://rdfa.info/.

[5] Ben Adida, Mark Birbeck, editors. RDFa Primer. http://www.w3.org/TR/xhtml-rdfa-primer/, 2008. W3C ajánlás.

[6] hCard. http://microformats.org/wiki/hcard.

[7] Operator. http://www.kaply.com/weblog/operator.

[8] Friend of a Friend (FOAF). http://www.foaf-project.org/.

[9] Dan Brickley, Libby Miller, editors. FOAF Vocabulary Specification, 2007. http://xmlns.com/foaf/spec/.

[10] Description of a Project (DOAP). http://usefulinc.com/doap.

[11] Semantically-Interlinked Online Communities (SIOC). http://sioc-project.org/.

[12] SIOC Core Ontology Specification. http://www.w3.org/Submission/sioc-spec/, 2007. W3C előterjesztés.

[13] Semantic Radar. http://sioc-project.org/firefox/.

[14] Screen scraping. http://en.wikipedia.org/wiki/Screen_scraping.

[15] PiggyBank. http://simile.mit.edu/wiki/Piggy_Bank.

[16] ClearForest Gnosis. http://gnosis.clearforest.com/.