Proceedings of the Joint International Workshop:

- **2013 Linked Data in Practice Workshop (LDPW2013)**
- **The First Workshop on Practical Application of Ontology for Semantic Data Engineering (PAOS2013)**

Co-located with 3rd Joint International Semantic Technology Conference (JIST 2013)
Seoul, Korea, November 30, 2013

Edited by

- Hideaki Takeda (LDPW) *
- Jason J. Jung (LDPW) **
- Kouji Kozaki (PAOS) ***
- Marut Buranarach (PAOS) ****
- Thepchai Supnithi (PAOS) ****

* *National Institute of Informatics (NII), Japan*
** *Yeungnam University, Korea*
*** *Institute of Scientific and Industrial Research, Osaka University, Japan*
**** *National Electronics and Computer Technology Center (NECTEC), Thailand*

2013 Linked Data in Practice Workshop (LDPW2013)

Organizing Committee

- Hideaki Takeda (National Institute of Informatics, Japan)
- Jason J. Jung (Yeungnam University, Korea)

Program Committee

- Christian Bizer (University of Mannheim, Germany)
- Naoki Fukuta (Shizuoka University, Japan)
- Sebastian Hellmann (University of Leipzig, Germany)
- Ryutaro Ichise (National Institute of Informatics, Japan)
- Kouji Kozaki (Osaka University, Japan)
- Shinichi Nagano (Toshiba Cooperation, Japan)
- Claus Stadler (University of Leipzig, Germany)
- Takahira Yamaguchi (Keio University, Japan)

The First Workshop on Practical Application of Ontology for Semantic Data Engineering (PAOS2013)

Organizing Committee

- Thepchai Supnithi (NECTEC, Thailand)
- Marut Buranarach (NECTEC, Thailand)
- Kouji Kozaki (Osaka University, Japan)

Program Committee

- Chutiporn Anutariya (Shinawatra University, Thailand)
- Key-Sun Choi (KAIST, Korea)
- Ronald Denaux(iSOCO, Spain)
- Dimoklis Despotakis (University of Leeds & Blueclaw Ltd., UK)
- Vania Dimitrova (University of Leeds, UK)
- Seiji Koide (National Institute of Informatics, Japan)
- Takeshi Morita (Aoyama Gakuin University, Japan)
- Ekawit Nantajeewarawat (SIIT, Thammasat University, Thailand)
- Ponrudee Netisopakul (KMITL, Thailand)
- Munehiko Sasajima (YMP-Mundus Corporation, Japan)
- Boontawee Suntisrivaraporn (SIIT, Thammasat University, Thailand)
- Dhaval Thakker (University of Leeds, UK)

Preface

The Joint International Workshop: 2013 Linked Data in Practice Workshop (LDPW2013) and the First Workshop on Practical Application of Ontology for Semantic Data Engineering (PAOS2013) took place in Seoul, Korea on November 30, 2013. The event was affiliated with the 3rd Joint International Semantic Technology Conference (JIST2013). Seven submissions (four for LDPW and three for PAOS) were selected based on a rigorous reviewing process. Each submission was reviewed by at least two program committee members. The organizers would like to thank the program committee members for their contribution in carefully reviewing the submissions.

June 2014

Hideaki Takeda
Jason J. Jung
Kouji Kozaki
Marut Buranarach
Thepchai Supnithi

Table of Contents

Building DBpedia Japanese and Linked Data Cloud in Japanese

Fumihiro Kato[1,2], Hideaki Takeda[1,3], Seiji Koide[1,2], and Ikki Ohmukai[1,3]

[1] National Institute of Informatics, 2-1-2, Chiyoda-ku, Tokyo, Japan
[2] Research Organization of Information and Systems, Tokyo, Japan
[3] The Graduate University for Advanced Studies, Kanagawa, Japan
email: {fumi, takeda, koide, i2k}@nii.ac.jp

Abstract. Wikipedia is one of the most valuable language and ontological resources covering wide domains so that DBpedia, LOD based on Wikipedia, plays the important role in the LOD cloud by connecting various resources. DBpedia Japanese is the LOD created from Wikipedia Japanese just as DBpedia data sources in other languages like English and German. We here describe how the conversion could be carried out with the efforts to fit DBpedia software and show the results of the dataset. We also describe how the created DBpedia Japanese is used by other Linked Data and show the Linked Data Cloud in Japanese.

1 Introduction

Today, various datasets are interconnected to each other under the concept of Linked Data[1]. In particular cross-media data such as encyclopedia plays a hub to connect data in various fields. In order to promote Linked Data in Japan, we have started to offer DBpedia Japanese which is generated from Wikipedia Japanese since 2012. It is expected to be a hub for Japanese resources. It has links to DBpedia English [2] via cross-language links so that it is also exected to connect Japanese resources with other international resources.

We also created RDF version of Japanese WordNet[3][4] and connected it to DBpedia Japanese[5]. Although we have an electoric dictionary called EDR Electronic Dictionary[4] originally developed from the scratch [6]. It is still continued updating but not open-free. So the Japanese WordNet is the first free online dictionary and therefore its RDF version is also the first LOD resource for Japanese. Both DBpedia Japanese and the RDF version of WordNet Japanese are important as primary resources for Japanese.

In this paper, we focus on DBpedia Japanese. In Section 2, we describe how DBpedia Japanese has been created, in particular show language-related issues in using the software. Then we show the Linked Data Cloud in Japanese in Section 3. We describe what we have learnt during the process in section 4, and conclude the paper at Section 5.

[4] See `http://www2.nict.go.jp/out-promotion/techtransfer/EDR/J_index.html`

2 DBpedia Japanese

DBpedia Japanese is the DBpedia generated from the Japanese Wikipedia. It is an internationalization of DBpedia where all parts fo the software used to build it from Wikipedia are those developed for English DBpedia. Therefore most of the building process were done except some language-specific treatments like the configuration for information extraction, and ontology mapping.

DBpedia Japanese is built and maintained in the activity of LODAC Project[5] where various data resources such as museums and biology are published as Linked Data.

Currently it contains 79,423,068 triples including links to the Japanese WordNet and the Japanese Wikipedia Ontology[7][8], and the statistics for ontology mapping is shown in Table 2. Roughly speaking, the rate for ontology mapping is a half of English DBpedia. There is still room to improve. In the following sections, we describe what we have done to build DBpedia Japanese and the results we achieved.

2.1 Customization of DBpedia Information Extraction Framework

DBpedia Information Extraction Framework (DIEF) [6] is the package of the software to extract information from Wikipedia. It is basically applicable to Wikipedia sources in any languages but is needed to customize in order to adapt information written in the language[9]. In building DBpedia Japanese, we set up the following extraction modules for Japanese;

1. DisambiguationExtractor
2. HomepageExtractor
3. ImageExtractor
4. PersondataExtractor

Some modules are needed to customize for Japanese Wikipedia as follows.

DisambiguationExtractor We added the following line in the configuration file for DisambiguationExtractor for processing disambiguation pages.

```
"ja" -> "(曖昧さ回避)"
```

HomepageExtractor This module is used to extract official web pages. We added three patterns to process for hyperlinks with the following strings.

```
propertyNamesMap
"ja" -> Set("homepage", "website", "ホームページ", "ウェブサイト", "Web
サイト", "Ｗｅｂサイト")
externalLinkSectionsMap
```

[5] http://lod.ac
[6] http://wiki.dbpedia.org/Documenation

```
"ja" -> "外部リンク"
officialMap
"ja" -> "公式"
```

They mean Web page, Outgoing page from Wikipedia, and Official respectively.

ImageExtractor This module is used to extract links to images. To avoid non re-usable images, we added the following line;

```
"ja" -> """(?i)\{\{\s?(Non free|Non-free pubart)\s?\}\}""".r
```

PersondataExtractor This module is used to extract information on persons. We added "ja" in the supported languages and defined the following patterns.

1. Names of templates for personal information
2. "名前"(name)
3. "別名"(alias)
4. "概要"(abstract)
5. dates and places for birth and death

DBpedia Japanese uses IRIs to identify resources now so that there are no prolems to include Japanese characters. But there was the problem for coding since URI instead of IRI was used in Virtuoso before version 6.1.4.

2.2 Statistics of the conversion

The results of the conversion is shown in Table 1. It is the statistics by the first build of DBpedia Japanese on January, 2012. Wikipedia Japanese contains 1,558,754 fragments including article, templates, image descriptions, and primary meta-pages. We can conclude ca. 90% of the original data is converted into DBpedia. The bold figures indicate the improvement by the configuration for Japanese Wikipedia.

2.3 Ontology Mapping

In DBpedia, an ontology is created and shared among different languages. A class in the ontology typically corresponds to templates which are used to generate Infobox in Wikipedia. Properties in a class corresponds to parameters in templates. Ontology mapping in DBpedia is to create classes and their properties and to create mapping from templates in Wikipedia. Ontology mapping is not easy task indeed since it is required to understand both background knowledge in domains and structures in templates. In order to stimulate the ontology mapping activity, we hold two so-called "mapping party" where people gathered and created mapping together on August 2012 and March 2013 in which 10 and 25 people participated. The current results of mapping is shown in Table 2. The

Table 1. Statistics of the conversion

Type	No. w/o configuration	No. w configuration
label	1,409,191	1,409,191
geo	34,368	34,368
infobox_properties	7,664,573	**7,664,575**
infobox_properties_definitions	33,214	33,214
infobox_test	7,192,853	**7,192,855**
page_links	44,421,598	44,421,598
wikipedia_links	4,227,573	4,227,573
article_categories		**2,153**
disambiguation		**106,386**
homepages		**49,355**
personadata		**1,811**
images		**843,170**

numbers of mapped templates are still low in comparison with English DBpedia. One of the reasons is that we did not create new classes yet, rather added mapping from the existing classes to templates in Wikipedia Japanese. There are some original classes like "武士" or Samurai but we have not created such classes yet. We need more improvement on ontology mapping.

Table 2. Statistics for Ontology Mapping in DBpedia (October 10, 2013)

	Japanese	English
rate of all templates in Wikipedia are mapped	4.67% (81 of 1,733)	6.33% (369 of 5826)
rate of all properties Wikipedia are mapped	2.47% (1581 of 62,679)	3.47% (6,169 of 177,599)
rate of all template occurrences Wikipedia are mapped	47.99% (286,858 of 597,696)	82.24% (2,2435,773) of 2,728,357
rate of all property occurrences Wikipedia are mapped	38.75% (3,128,208 of 8,071,982)	54.95% (27,283,343 of 49,654,072)

3 Linked Data Cloud in Japanese

In this section, we describe Linked Data resources in Japanese. Here Linked Data resources in Japanese means Linked Data resources where their important or significant part is data presented in Japanese or data is mainly about Japan. An example for the former is DBpedia Japanese and the latter is the statistics data about Japanese Industry. The distinction between them is sometimes important but they mostly overlapped in our case since Japanese is mostly used in Japan.

Linked Data resources in Japanese have been limited but have increased rapidly in a couple of years. As mentioned in Section 1, LODAC project initiated

to create datasets Linked Data in Japanese. We create and maintain LODAC Location, LODAC Museum, LODAC SPECIES, and WorldNet-J, and connect them to each other.

Semantic Web Community in Japan started the contest to promote LOD, namely the LOD Challenge. LOD Challenge was hold twice in 2012 and 2013, and collected over 70 and 200 applications respectively. It should be noted that the category of LOD challenge includes Dataset section as well as Idea and Application sections. Through the LOD challenge, several Linked Data resources are published.

We collect major Linked Data resources in Japan and map them in a figure (see Figure 3 and 1). The criteria to include datasets in it is as follows;

1. providing more than 10,000 triples,
2. providing either derefernece, data dump or SPARQL Endpoint,
3. providing labels in Japanese, and
4. open license is recommended but not mandatory.

The last criteria may seem odd but we think currently that forcing clear open licenses is not suitable at the beginning of the emergent community of LOD in Japan since there are a lot of datasets which are widely used just like with open licenses but do not have licenses yet.

3.1 Overview of the Linked Data Cloud in Japanese

The Linked Data Cloud in Japanese is still small, i.e., only 21 datasets are included which is just one tenth of Global LOD cloud. The proportion of the cloud is similar. Major category is publication-related datasets. Then some datasets are from geographic, life science and government domains. The clear difference is that there are many datasets without open licenses. It is mainly because the original datasets from which Linked Data datasets are generated often lack licenses. Some look very open but there are no clear mentions for licenses yet.

We pick up some of the datasets that are important resources connected to DBpedia tightly.

3.2 Japanese WordNet and its Linking to DBpedia Japanese

The efforts for multilingual WordNet has been made worldwide based on the Princeton's English WorldNet so far. In 2008, the Japanese WordNet (WN-ja) was developed and released by the National Institute of Information and Communications Technology (NICT) in Japan [3][4]. Currently WN-ja is built using the structure of the English WorldNet so that the synsets in WN-ja is equivalent to those in English WordNet and only words in Japanese are added as translation of words in English. We used version (1.1) of the Japanese WordNet with 187,000 senses (word-synset pairs), 57,000 concepts (synsets) and 94,000 unique Japanese words. For up-to-date information on the Japanese WordNet see `nlpwww.nict.go.jp/wn-ja`.

There are some attempts to convert WorldNet into RDFs. In 2006, W3C published the Working Draft for the representation in RDF of WorldNet 2.0 [10], in which the OWL representation of WorldNet and an OWL schema for WorldNet were introduced. Then, we applied the proposal to WorldNet 2.1 with the two extended pointer properties for the new version, i.e., `instanceHypernymOf` and `instanceHyponymOf` in 2006 [11][12], and subsequently for WorldNet 3.0. Up to now, several attempts to represent Princeton's WorldNet in OWL were made along with updating the WorldNet. Whereas the team members of the W3C Working Draft had actually converted WorldNet 2.0 to OWL representation [13], and then from the viewpoint of Linked Open Data (LOD), de Melo and Weikum has made the word search web pages [14]. The latest WN-ja is built on Princeton's English WorldNet 3.0. Appropriate Japanese words are added and linked to synsets via wordsenses as usual in the WorldNet manner. Thus, according to the W3C proposal of RDF representation of WorldNet, we have made the conversion of WN-ja to OWL [15].

Since both WorldNet and Wikipedia are the most famous comprehensive languages resources, there are many studies how combining them can contribute to build better languages resources (e.g., Yago [16]). Currently we here link entities in both datasets literally, i.e., link entities which share the same strings since we want to provide the basic dataset for Japanese language resources. We pick up nouns from WorldNet-ja and map names of resources and properties.

Table 3 shows the statistics of linking data of WN-ja to DBpedia Japanese, and Table 4 shows the statistics of linking data of DBpedia Japanese to WN-ja. In this attempt of linking by words in WN-ja and resource and property names in DBpedia Japanese, we made the connection by literally exact matching. Therefore, the mapping is exactly one by one and inversely equivalent in this case.

Table 3. Number of Linked Data from WN-ja to DBpedia Japanese

DBpedia	# of linked	# of WN nouns	rate
resources	33,017	65,788	50.1%
properties	1,245	65,788	1.9%

Table 4. Number of Linked Data from DBpedia Japanese to WN-ja

DBpedia	# of linked	# of IRIs	rate
resources	33,017	1,456,158	2.3%
properties	1,245	16,020	7.8%

3.3 Japanese Wikipedia Ontology

Takahira Yamaguchi and his group.[7][8] created Ontology for Wikipedia, i.e., it categorizes articles in Wikipedia. It provides the class hierarchy with properties, and is built by combining different techniques to analyze texts of articles, infobox, and categories. Currently Japanese Wikipedia Ontology and DBpedia Japanese are connected via `owl:sameAs` but we are planning to integrate them just as DBpedia and Yago.

4 Lessons Learnt

We have worked to create DBpedia Japanese as LOD resources in Japanese which can be used by other LOD datasets. What we have learnt through the process are as follows;

1. The language-dependent software problems are been solved mostly, but some still exist. They are not serious but it takes time and techniques to solve them. In particular, some software and tools are not completely compatible with IRI or UTF-8, and some lack language tags when describing literals. Anyway it is important to share demands from local communities with the global community.
2. People are willing to link to our resources. Within a year and so, we have 13 datasets which link their resources to those in DBpedia Japanese. The problem is that it is difficult to find datasets linking to us. We collect the datasets by personal communication or by local academic meetings. We need some techniques to find them automatically.
3. Having the hub for linking is not just good to make LOD cloud but also good to connect people. We had several academic meetings and tutorials for LOD. DBpedia Japanese is always a good example to demonstrate LOD since it covers various topics including daily issues and they are of course written in Japanese. People were motivated to join the LOD community in order to connect their data to DBpedia Japanese.

5 Conclusion

Wikipedia isthe precious ontology resources covering wide domains so that DBpedia play an important role in the LOD cloud by connecting various LOD resources. Building DBpedia for an lanugage is not so difficult task since all parts of the software are ready to use except some minor issues. We strongly recommend people in other languages to build own DBpedia. Building DBpedia is not just adding an useful resource but stimulating people to connect their data. It is the first step to make the society to be connected by data.

References

1. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space (1st edition). Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool (2011)
2. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: Dbpedia-a crystallization point for the web of data. Web Semantics: Science, Services and Agents on the World Wide Web **7**(3) (2009) 154–165
3. Isahara, H., Bond, F., Uchimoto, K., Utiyama, M., Kanzaki, K.: Development of japanese wordnet. In: Sixth international conference on Language Resources and Evaluation (LREC 2008), Marrakech (2008)
4. Bond, F., Isahara, H., Fujita, S., Uchimoto, K., Kuribayashi, T., Kanzaki, K.: Enhancing the japanese wordnet. In: Proceedings of the 7th Workshop on Asian Language Resources. ALR7, Stroudsburg, PA, USA, Association for Computational Linguistics (2009) 1–8
5. Koide, S., Takeda, H.: Rdfization of japanese electronic dictionaries and lod. In: 2nd Workshop on Linked Data in Linguistics: Representing and linking lexicons, terminologies and other language data Pisa, Italy, 23rd September 2013. Collocated with GL2013. (2013)
6. Yokoi, T.: The edr electronic dictionary. **38**(11) (1995) 42–44
7. Morita, T., Sekimoto, Y., Tamagawa, S., Yamaguchi, T.: Building up a class hierarchy with properties from japanese wikipedia. In: Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01, IEEE Computer Society (2012) 514–521
8. Tamagawa, S., Sakurai, S., Tejima, T., Morita, T., Izumi, N., Yamaguchi, T.: Building up a class hierarchy with properties from japanese wikipedia. In: Proceedings of the The 2010 IEEE/WIC/ACM International Joint Conferences on Web Intelligence, IEEE Computer Society (2010) 279–286
9. Kontokostas, D., Bratsas, C., Auer, S., Hellmann, S., Antoniou, I., Metakides, G.: Internationalization of linked data: The case of the greek {DBpedia} edition. Web Semantics: Science, Services and Agents on the World Wide Web **15**(0) (2012) 51 – 61
10. van Assem, M., Gangemi, A., Schreiber, G.: Rdf/owl representation of wordnet. W3C Working Draft 19 June 2006 (2006) http://www.w3.org/TR/2006/WD-wordnet-rdf-20060619/.
11. Koide, S., Morita, T., Yamaguchi, T., Muljadi, H., Takeda, H.: Owl expressions on wordnet and edr. In: SIG for Semantic Web and Ontology, SIG-SWO-A601-03, The Japanese AI Society (2006) (In Japanese).
12. Koide, S., Morita, T., Yamaguchi, T., Muljadi, H., Takeda, H.: Rdf/owl representation of wordnet 2.1 and japanese edr electronic dictionary. In: ISWC2006, Poster. (2006)
13. Van Assem, M., Gangemi, A., Schreiber, G.: Conversion of wordnet to a standard rdf/owl representation. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC '06), Genoa, Italy. (2006)
14. De Melo, G., Weikum, G.: Language as a foundation of the semantic web. In: Proceedings of the 7th International Semantic Web Conference (ISWC 2008). Volume 401. (2008)
15. Koide, S., Takeda, H., Ohmukai, I.: An lod approach toward wordnet japanization. In: SIG for Semantic Web and Ontology, SIG-SWO-A1103-05, The Japanese AI Society (2011) (In Japanese).

16. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A large ontology from wikipedia and wordnet. Web Semantics: Science, Services and Agents on the World Wide Web **6**(3) (2008) 203–217

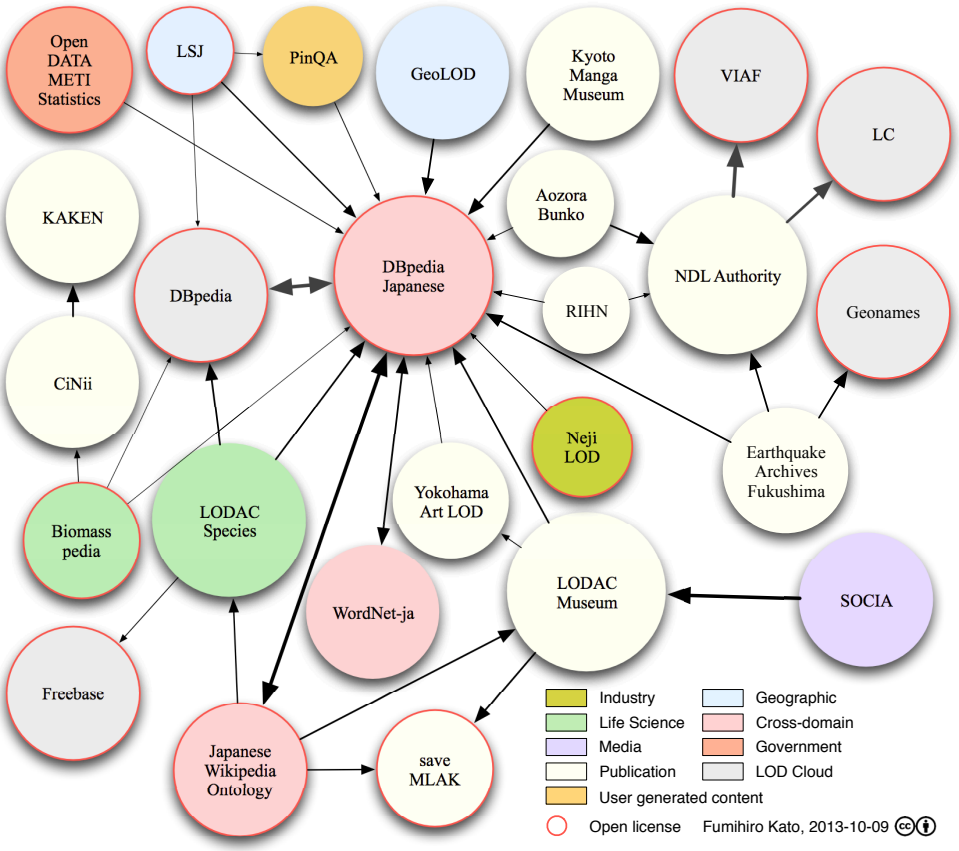


Fig. 1. Linked Data Cloud in Japan

Dataset name (en)	Explanation	Homepage	De-reference-able	Dump	SPARQL Endpoint	License	Triples	Property to DBpedia Japanese	Links to DBpedia Japanese
DBpedia Japanese	Dataset from Wikipedia	http://ja.dbpedia.org	○	○	○	CC-BY-SA	69833731	N/A	0
LODAC Museum	Museum Collection Data in Japan	http://lod.ac	○	×	○	?	42566090	dcterms:references	1150
Japanese Wikipedia Ontology	Ontology for Wikipedia	http://www.wikipediaontology.or	○	○	○	CC-BY-SA	22359069	owl:sameAs	956640
Web NDL Authorities	Authority Files of National Diet Library	http://id.ndl.go.jp/auth/ndla	○	△ (only	○	http://iss.ndl.go.jp/ndla/use/	16777215	N/A	0
LODAC Species	Dataset on Species	http://lod.ac/species/	○	×	○	?	13375045	owl:sameAs	7367
Kyoto Kokusai Manga Museum Authority LOD	Bibliography of Manga	http://mdlab.slis.tsukuba.ac.jp/lodc2012/kmm/	×	○	○	CC-BY-NC-SA	8517803	rdfs:seeAlso	4187
GeoLOD	Dataset for geographical names in Japan	http://geolod.ex.nii.ac.jp/	○	×	○	?	6383185	rdfs:seeAlso	14870
SOCIA	Social Opinions and Concerns for Ideal Argumentation	http://www.open-opinion.org	×	×	○	?	6139197	N/A	0
WordNet-ja	Dataset from WordNet	http://lod.ac/wiki/WordNet-ja	×	○	○	http://nlpwww.nict.go.jp/wn-	4074535	skos:closeMatch	34262
Open DATA METI statistics	Statistics Data by Ministory of ETI, Japan	http://datameti.go.jp	×	○	○	CC-BY	2827071	rdfs:seeAlso	2020
The Great East Japan Earthquake Archives Fukushima	Archive for the Earthquake-related contents	http://fukushima.archive-disaster	○	×	○	.archive-disasters.jp/doc	2152941	ma.archive-disasters.jp/	24689
Yokohama Art LOD	Art Inforamation in Yokohama City	http://fp.yafjp.org/yokohama_art_lod	○	×	○	CC-BY-ND	638846	owl:sameAs	42
Biomasspedia	Information about Biomass	http://biomasspedia.net	×	○	○	CC-BY	608806		
saveMLAK	Information about Musuem, Library, Archive and Kominkan in Tohoku Area	http://savemlak.jp	○	△ (not RDF)	N/A	CC0, CC-BY-SA	514002	N/A	0
Aozora Bunko Linked Open Data	Bibliography of Open Books	http://mdlab.slis.tsukuba.ac.jp/lodc2012/aozoralod/	×	○	○	CC-BY-NC	490532	rdfs:seeAlso	612
Screw LOD	Dataset for Screw product	http://monodzukurilod.org/neji/	○	○	○	CC-BY-SA	209807	owl:sameAs	11
PinQA	Q & A on local information	http://pinqa.com	○	×	○	?	154136	owl:sameAs	1672
LSJ: Location Site of Japanimation	DB for locations used in Anime	http://cheese-factory.info/	×	○	○	CC-BY	15444	cfo:link_to_dbpedia_jp	978
Envrionment Repository Prototype System	Repository for Environment-related research data	http://rihnexers.chikyu.ac.jp	○	×	○	?	12603	dcterms:subject,	1032 (dcterms:
CiNii	Database of papers published in Japan	http://ci.nii.ac.jp	○	×	N/A	?		N/A	0
KAKEN	Report DB on Mext-funded project (Kaken)	http://kaken.nii.ac.jp	○	×	N/A	?		N/A	0

Fig. 2. Linked Data resources in Japan

An LOD Practice
Lessons Learned from Open Data METI

Seiji Koide[1,4], Fumihiro Kato[1,4], Iwao Kobayashi[1],
Tadashi Mima[2], Takayuki Yamada[2],
Yu Asano[3], Makoto Iwayama[3],
Hideaki Takeda[4,1], and Ikki Ohmukai[4,1]

[1] Linked Open Data Initiative, Inc., Tokyo, Japan,
koide@linkedopendata.jp,
WWW home page: http://linkedopendata.jp
[2] Hitachi Consulting Co., Ltd., 2-4-1 Kojimachi, Chiyoda-ku, Tokyo 102-0083, Japan
[3] Hitachi, Ltd., Central Research Laboratory, 1-280, Higashi-koigakubo,
Kokubunji-shi, Tokyo 185-8601, Japan
[4] National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430,
Japan

Abstract. The Ministry of Economy, Trade and Industry of Japan has launched the catalog site of their own open data at January 2013, starting with 70 data sets using customized CKAN. All of data sets are published under the Creative Commons Attribution license. After the initial publication, new data sets are added into the catalog, and the download number is also increasing smoothly. In this paper, we report the process of building this successful catalog site, and discuss the lessons learned from the project.

Keywords: open data, catalog site, CKAN, METI, LODI

1 Introduction

Pressed by the worldwide movement of the Open Government, e.g., Data.gov.uk[5] and Data.gov in USA[6], etc., and in recognition of the potential of open data after the Great Earthquake Disaster at East Japan, the Japanese IT Strategic Headquarter under the Cabinet Secretariat issued the Policy of Open Government Data Strategy in July 2012, and then, the Ministry of Economy, Trade and Industry (METI) organized the Public Data Working Group of IT Fusion Forum in order to promote a project of the Open Government Data Strategy. As an actual activity for the promotion, METI planned the launch of a portal site for METI's own data. Thus, in October 2012, the operation group composed of Hitachi Consulting Co., Ltd., Central Research Laboratory of Hitachi, Ltd., and

[5] http://data.gov.uk/
[6] http://www.data.gov/

Linked Open Data Initiative, Inc. (LODI), which is a specific non-profit corporation in Japan, was formed in order to apply to the public call for the project fulfillment.

This first project in Japanese Government on open data was successfully finished at March 2013 in spite of the short period in a half year, and the result had an impact on not only over all Japanese Government but also the Japanese Society along with the encouraged LOD momentum in Japan.

This paper is the first report of the Open Data METI portal site[7] to the Semantic Web and LOD community by who engaged in it.[8]

2 Semantic Web and LOD Background in Japan

Up to 2012, the time of the public calling of the Open Data METI project, the effort of promoting Semantic Web and LOD had been made in long time in Japan. Especially, it is remarkable that the recent activities by the Executive Committee of LOD Challenge under the Semantic Web Conference supported by Prof. Hagino's Lab. in Keio University has made a great contribution to LOD in Japan. The momentum is shown as the increasing number of applicants for LOD Challenge. In the LOD Challenge 2011, we had 21 works of data sets course, 34 works of idea course, and 18 works of application course. In 2012, we had 87, 50, 44, and 24 works of data sets, idea, application, and visualization course, respectively.

Furthermore, in advance to the movement in the Japanese Central Government, several local governments had tackled open data. Most of board members of LODI had deeply concerned themselves to two local governments, Yokohama and Sabae, in order to promote the LOD movement in these cities.

LODI was officially born in August 2012 as specific non-profit corporation for the purpose of spreading LOD in Japan, but the first intention was projected at a pub in Yokohama in 2010 by two key persons, a local volunteer in Yokohama, Mr. Kobayashi, and Prof. Takeda at National Institute of Informatics. So, everything had it got up to work well at the beginning of the Open Data METI project.

3 Formation of Operation Group and Roles of Members

Hitachi Consulting in charge of secretariat of the IT Fusion Forum had the advantage of seeing and understanding the trend of Japanese IT and the movements of the Central Government, and then intended to acquire the Open Data METI project. They approached to LODI, who was established as specific non-profit corporation just one month before the time. Hitachi Consulting described in the proposal, "As LOD Initiative retains advanced experiences on the Open Data

[7] The current web pages are shown at http://datameti.go.jp/.

[8] The public announcement and a private report to the United Nations have already done by METI.

Project of Sabae city, Japanization of CKAN, etc., it allows us to enable re-use of data by its competence."

On the other hand, it was obvious for the board members of LODI that this project would be a critical point whether LOD would populate in Japan or not. Thus, the operation group of Open Data METI was formed with Hitachi Consulting, Central Research Laboratory of Hitachi, and LODI. The roles in the project of these three bodies are shown in Table 1.

Table 1. Roles of Project Group Members

Organization	Role
Hitachi Consulting	Project management Secretariat of the working group Market evaluation, investigation on needs Research of rules and promoting organizations Investigation and decision of target data
Central Research Center of Hitachi	Data preparation and registration Development of tools
LODI	Data arrangement and translation Preparation of meta-tags Research of WebAPI

4 Problems and Solving Process

4.1 Predicted Problems and Solving Process

Short Term Project and Different Characteristics between a Profit Organization and a Non-Profit Organization. LODI voluntarily charged themselves with the responsibility of building an open site using CKAN, because there was no software engineering body responsible for it in the group, whereas LODI had the potential on using CKAN. Fortunately, one of the board members of LODI had started the investigation of CKAN and its Japanization. He had already experiences of CKAN installation on top of his VPS servers. Then, he quickly developed the initial evaluation version on top of the personal VPS servers. Thus, it was able to show the prototype system of open data catalog site to METI persons in charge of this project at November. We used to call it the 0th version, while we called an official developing system on EC2 the Alpha version, and the system intended to run on the proper machine for Open Data METI the Beta version.

Before the notification of operation from Hitachi Consulting to LODI, we had a problem on the contract documents between Hitachi Consulting and LODI.

It arose from the incompatibility of basic characteristics between a profit organization and an NPO. As a default condition on contract, Hitachi Consulting addressed the contract condition that all copyrights were reserved to Hitachi Consulting in usual custom of their own. However, it was never acceptable for LODI, who aims the promotion of LOD knowledge and technology in Japan. At the end, as Hitachi Consulting recognized the purpose of open data and this project, this problem was solved by adding a special term into the contract document.

Contents Management System. It was well known that a content management system Drupal was often used in combination with CKAN. We also regarded the appearance of sites as important. It was obvious that an attractive and flexible top page of Open Data METI was needed apart from the appearance of CKAN. We discussed in the inside of LODI about the choice of a CMS, Drupal or WordPress. Drupal by PHP is equipped with highly convenient but complex functionalities. On the other hand, WordPress is the most popular one as CMS and a member of LODI had rich experiences on WordPress. In this project, the period of engagement was very short, and it was estimated that no intimate linkage between a CMS and CKAN was required. Therefore, we decided to adopt WordPress as CMS for Open Data METI.

Water Fall Model v.s. Agile Development. In the water fall model of software development, specifications of software requirements are presented as ordering conditions by a vendee, and software vendors reply to the requirements by submitting external software specifications to the vendee, and then, both agree on the detail specifications or the internal software specifications before or after the formation of contracts. In this classical methodology of software development, a vendee is required to evaluate software specifications written on the documents. The requirements for human interface must be also made clear before the contract. Furthermore, it is usually impossible for vendors to accept drastic changes of software functionalities and interface after contract.

On the other hand, in the agile development method, frequent meetings and discussions for decision making are held in the operation between vendors and a vendee. In the case of Open Data METI, the project had to be proceeded without any precise software specification documents. Thus, we had to inevitably be in the agile development methodology. There were no software requirement specification documents and no external specification documents. So, the details of software are decided on the fly, and the software was changed step by step, seeing revised versions, starting with simple systems composed of native WordPress plus CKAN, to special systems tailored so as to meet the exposed requirements of Open Data METI.

Although many details were modified from original ones of CKAN, the most remarkable change was enabling the selection of CC licenses, including CC-BY-ND, for CKAN data sets and the representation of CC icons upon the web pages.

4.2 Unpredicted Problems and Solving Process

Operating System. Since ubuntu is the standard operating system of CKAN, we were anxious about the OS on Open Data METI. The Open Data METI systems were a part of larger METI web sites. At the beginning of the project, we confirmed that the OS was CentOS, and then we ported the CKAN system to CentOS 6.3, which was the latest version of CentOS at the time. However, it was CentOS 5.8 in fact. There were many differences of module compositions between CentOS 6.3 and 5.8. We solved this problem with hard work in a very short period, but this effort was useless at the end because of the memory shortage.

Memory Shortage. Another big unexpected problem was allowable memory size. It is common sense for LOD researchers that the more memory the better performance. We usually use 8 GB at least and more for LOD research. However, at first it was announced to us that available memory size allocated for Open Data METI was 512MB. The reason was that they aimed the effective management of METI sites as a whole, and the software vendor who was responsible for the larger METI site did not know the common sense of LOD and no interaction between them and us. In the end, this problem for Open Data METI was hidden by the evidence of memory shortage for the whole system of METI sites, and the initial plan that the Beta version (the final version in this project) runs on the same machine as the larger METI sites was abandoned. As a result, the Beta version did not leave at EC2, which was prepared for the Alpha version (official development version in this project), and it was provided to the public on EC2 up to the end of the term of this project.[9]

Strong and Urgent Requirement for Beta Version from METI. The date of meeting of the Public Data Working Group strongly swayed the schedule of Open Data METI development. In the middle stage of development, METI planned to complete the functions of the Beta version for the purpose of the demonstration to the members of the Public Data Working Group. We made big efforts to meet this requirement. METI thanked us for making efforts after the demonstration. Thus, the Alpha version was finished in advance to the initial schedule plan.

Requirement for Page Views and Download Numbers. At the earlier stage of development, METI required to know the number of page views and data download. Then, we enabled to know the number of access to this sight using Google Analyticator plugin for WordPress. For CKAN, we utilized CKAN Google Analytics Extension to enable the access counting. The times of pressing the down load push button is also counted.

[9] The current version runs on the proper machine as a part of the whole METI site systems.

5 RDFization of Statistic Data and LOD

5.1 Selection of Statistic Data and a Question.

CKAN is an open software system for building catalog sites of open data, but not an RDF store for linked data. Seizing this opportune moment, LODI planned not only to build a catalog site for open data but also to realize several examples of linked data and show the potential of linked data along with providing a SPARQL endpoint. During this project in 2012, cataloged data sets were typed into two kinds of documents, white papers and statistic data. We supposed that statistic data should be appropriate to LOD. The industrial census data are published in a large number of tables. Each table shows a summary of data according to a specific view of specific aspects. These tables are categorized into 7 volumes depending on 7 standpoints.

At the first contact to experts of statistics in METI, we noticed that statistic data in the form of tables contain specific views for raw data of census. We also noticed that experts are very afraid of wrong interpretation of census data. On the other hand, we, LOD researchers, expect to show the potential of LOD by producing new knowledge in combination of distinct data in different domains. This is a sort of basic difference among professional and cultural mentalities in distinct science disciplines. It is difficult to obtain immediate consensus of opinion.

In expectation of admission from statistics experts, we made inquiries about the appropriate combinations of tables on industrial census data in 2010 that are cataloged in Open Data METI. Thus, we selected 4 tables, Industrial Fine Data by fine industrial product codes and Tokyo/Hokkaido/Osaka/Kyoto/prefectures, Industrial Semi-Fine Data by industrial semi-fine codes and Tokyo/Hokkaido/Osaka/Kyoto/prefectures, Land and Water Data by industrial semi-fine codes and industrial districts, and Industrial Semi-Fine Data by industrial semi-fine codes and municipality. Whereby many names of properties in tables are shared, and there is no incompatibility among coarse industrial codes in 4 digits and fine codes in 6 digits of commercial products, and there is no incompatibility among municipality areas and the area components of industrial area districts.

5.2 Brief Note on RDFization of Statistic Data

We learned the W3C proposal of RDFization for data in table forms, RDF Data Cube Vocabulary [1], and then applied it to statistic data in Open Data METI data sets. The result was reported in Takeda [2] and Asano [3]. Through this operation, LODI found the correct method of RDFizing table data, and informed it to Hitachi Central Research Laboratory with several examples, then Hitachi Central Research Laboratory practiced the transformation and registration of targeted statistic data according to the demonstrated examples by developing templates and tools.

Why RDFize table data Generally, the advantage of RDFization of table forms is pointed out as follows.

- A datum is defined by an IRI, which provides a globally unique name.
- A datum can unambiguously denotes a meaningful object by the help of the strict model theory and semantics of RDF. Whereby it is enabled to process data by machine.
- Relations among data in distinct domains can be expressed. Whereby linked data are available over the barrier of domains and organizations.

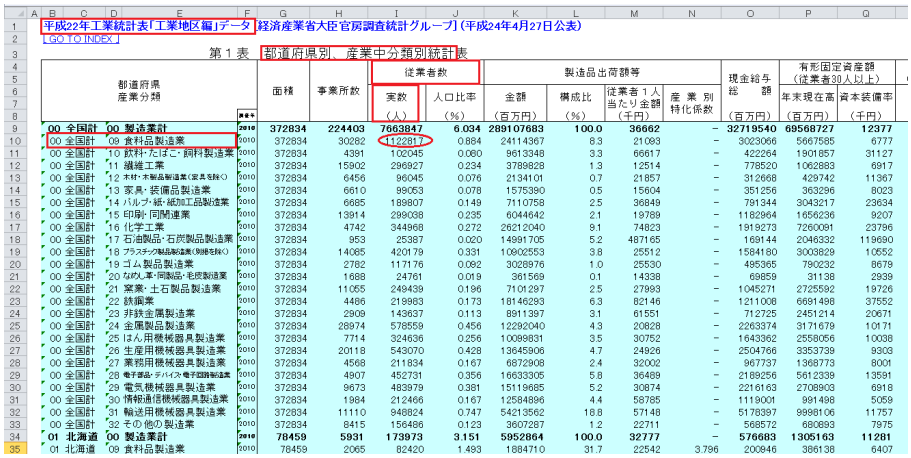
RDFization by RDF Data Cube Vocabulary The RDF Data Cube Vocabulary by W3C [1] is an RDF extension of the Statistical Data and Metadata Exchange (SDMX) ISO Standard. In the Data Cube Vocabulary, the benefit of RDFization of statistic data is captured as follows.

- The individual observations, and groups of observations, become (web) addressable. This allows publishers and third parties to annotate and link to this data; for example a report can refer to the specific figures it is based on allowing for fine grained provenance trace-back.
- Data can be flexibly combined across datasets sets. The statistical data becomes an integral part of the broader web of linked data.
- For publishers who currently only offer static files then publishing as linked-data offers a flexible, non-proprietary, machine readable means of publication that supports an out-of-the-box web API for programmatic access.
- It enables reuse of standardized tools and components.

Especially, on statistic data, the documentation of the RDF Data Cube states;

> "A statistical data set comprises a collection of observations made at some points across some logical space. The collection can be characterized by a set of dimensions that define what the observation applies to (e.g. time, area, gender) along with metadata describing what has been measured (e.g. economic activity, population), how it was measured and how the observations are expressed (e.g. units, multipliers, status). We can think of the statistical data set as a multi-dimensional space, or hyper-cube, indexed by those dimensions. This space is commonly referred to as a cube for short; though the name shouldn't be taken literally, it is not meant to imply that there are exactly three dimensions (there can be more or fewer) nor that all the dimensions are somehow similar in size."

Data Centric RDF Annotation A datum in a cell of a table can be regarded as an meaningful object with the annotations of row names at heads of rows, column names at the top of columns, titles of tables, and so on. For example, Fig. 1 shows a number of employees on food production in a industrial district area, which is annotated the row and the column names and table names. In such case, we create a cell node of RDF graph with linked annotations of a relevant area code, industrial code, and a table name, as shown in Fig.2.



平成22年工業統計表「工業地区編」データ　経済産業省大臣官房調査統計グループ（平成24年4月27日公表）

GO TO INDEX

第１表　都道府県別　産業中分類別統計表

都道府県	産業分類		面積	事業所数	従業者数 実数（人）	従業者数 人口比率（%）	製造品出荷額等 金額（百万円）	製造品出荷額等 構成比（%）	製造品出荷額等 従業者１人当たり金額（千円）	製造品出荷額等 産業別特化係数	現金給与総額（百万円）	有形固定資産額（従業者30人以上） 年末現在高（百万円）	有形固定資産額（従業者30人以上） 資本装備率（千円）
00 全国計	00 製造業計	2010	372834	224403	7663847	6.034	289107683	100.0	36662	－	32719540	69568727	12377
00 全国計	09 食料品製造業	2010	372834	30282	1122817	0.884	24114367	8.3	21093	－	3023066	5667585	6777
00 全国計	10 飲料・たばこ・飼料製造業	2010	372834	4391	102045	0.080	9613348	3.3	66617	－	422264	1901857	31127
00 全国計	11 繊維工業	2010	372834	15902	296927	0.234	3789828	1.3	12514	－	778520	1062883	6917
00 全国計	12 木材・木製品製造業（家具を除く）	2010	372834	6456	96045	0.076	2134101	0.7	21857	－	312668	429742	11367
00 全国計	13 家具・装備品製造業	2010	372834	6610	99053	0.078	1575390	0.5	15604	－	351256	363296	8023
00 全国計	14 パルプ・紙・紙加工品製造業	2010	372834	6685	189807	0.149	7110758	2.5	36849	－	791344	3043217	23634
00 全国計	15 印刷・同関連業	2010	372834	13914	299038	0.235	6044642	2.1	19789	－	1182964	1656236	9207
00 全国計	16 化学工業	2010	372834	4742	344968	0.272	26212040	9.1	74823	－	1919273	7260091	23796
00 全国計	17 石油製品・石炭製品製造業	2010	372834	953	25387	0.020	14991705	5.2	487165	－	169144	2046332	119690
00 全国計	18 プラスチック製品製造業（別掲を除く）	2010	372834	14085	420179	0.331	10902553	3.8	25512	－	1584180	3003829	10552
00 全国計	19 ゴム製品製造業	2010	372834	2782	117176	0.092	3028976	1.0	25530	－	495365	790232	8679
00 全国計	20 なめし革・同製品・毛皮製造業	2010	372834	1688	24761	0.019	361569	0.1	14338	－	69859	31138	2939
00 全国計	21 窯業・土石製品製造業	2010	372834	11055	249439	0.196	7101297	2.5	27993	－	1045271	2725592	19726
00 全国計	22 鉄鋼業	2010	372834	4486	219983	0.173	18146293	6.3	82146	－	1211008	6691498	37552
00 全国計	23 非鉄金属製造業	2010	372834	2909	143637	0.113	8911397	3.1	61551	－	712725	2451214	20671
00 全国計	24 金属製品製造業	2010	372834	28974	578559	0.456	12292040	4.3	20828	－	2263374	3171679	10171
00 全国計	25 はん用機械器具製造業	2010	372834	7714	324636	0.256	10099831	3.5	30752	－	1643362	2558056	10038
00 全国計	26 生産用機械器具製造業	2010	372834	20118	543070	0.428	13645906	4.7	24926	－	2504766	3353739	9303
00 全国計	27 業務用機械器具製造業	2010	372834	4568	211834	0.167	6872908	2.4	32002	－	967737	1368773	8001
00 全国計	28 電子部品・デバイス・電子回路製造業	2010	372834	4907	452731	0.356	16633305	5.8	36489	－	2189256	5612339	13591
00 全国計	29 電気機械器具製造業	2010	372834	9673	483979	0.381	15119685	5.2	30874	－	2216163	2708903	6918
00 全国計	30 情報通信機械器具製造業	2010	372834	1984	212466	0.167	12584896	4.4	58785	－	1119001	991498	5059
00 全国計	31 輸送用機械器具製造業	2010	372834	11110	948824	0.747	54213562	18.8	57148	－	5178397	8998106	11757
00 全国計	32 その他の製造業	2010	372834	8415	156486	0.123	3607267	1.2	22711	－	568572	680693	7975
01 北海道	00 製造業計	2010	78459	5931	173973	3.151	5952864	100.0	32777	－	576683	1305163	11281
01 北海道	09 食料品製造業	2010	78459	2065	82420	1.493	1884710	31.7	22542	3.796	200846	386138	6407

Fig. 1. An Example of Table Data

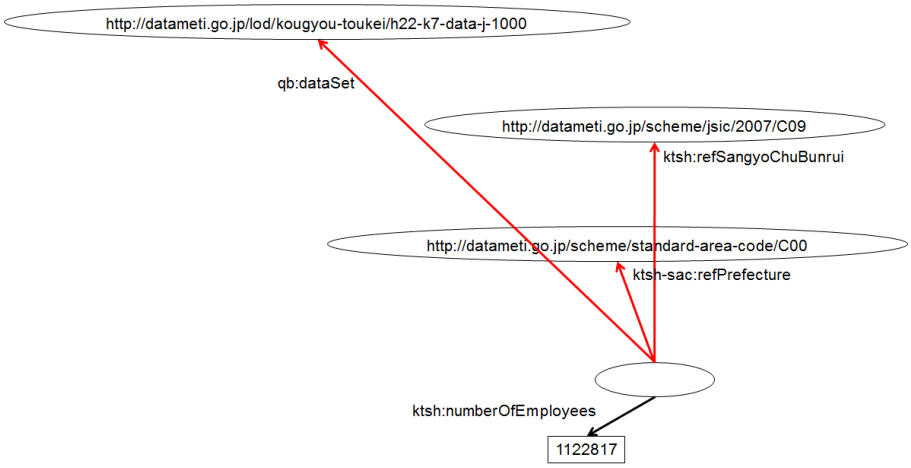


Fig. 2. Data Centric RDF Annotation of Table Data

6 System Configuration and Modification

6.1 System Modules

The main modules of Open Data METI at March 2013 are shown in Table 2 and Figure 3. The OS was CentOS 6.3 and Apache 2.2.15 was utilized for http daemon.

Table 2. Modules of Open Data METI Beta Version

Module	Functionality
CKAN 1.8	Data catalog software
WordPress 3.5.0	Content Management
PostgreSQL 8.4.13	Data base for CKAN and WordPress
SPARQL	SPARQL endpoint sever for Virtuoso
Virtuoso	RDF store and linked data server
Solr	Infomation extraction engine for CKAN
Postfix	Mail tranfer system
Tomcat 6.0.24	Java Servlet

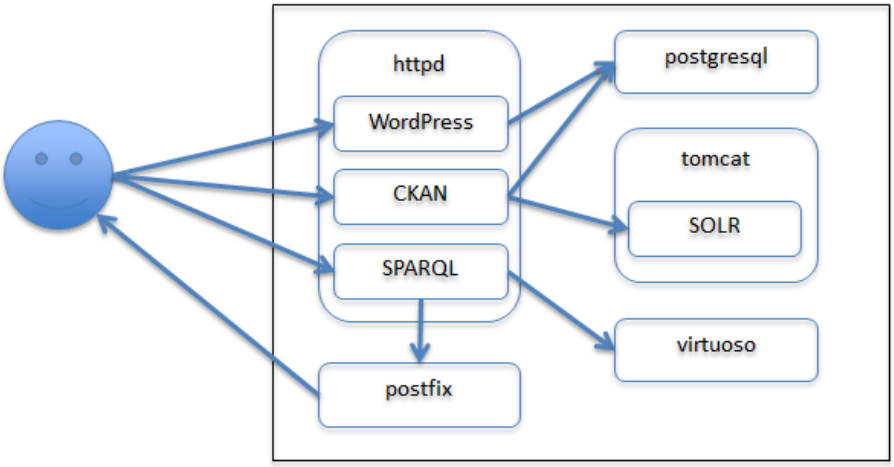


Fig. 3. Relation among Modules of Open Data METI

6.2 Versions of System

As mentioned earlier, we had three versions for Open Data METI after all. The 0th version was prepared on the responsibility of LODI. The machine for the

Alpha version was prepared on EC2 by Hitachi Consulting. The Beta version on the proper METI machines was intended in the initial plan to be the outcomes of this project. However, the Beta version was actually implemented on EC2 and provided for public use. The actual time table is shown in Figure 4 with a number of landmark points.

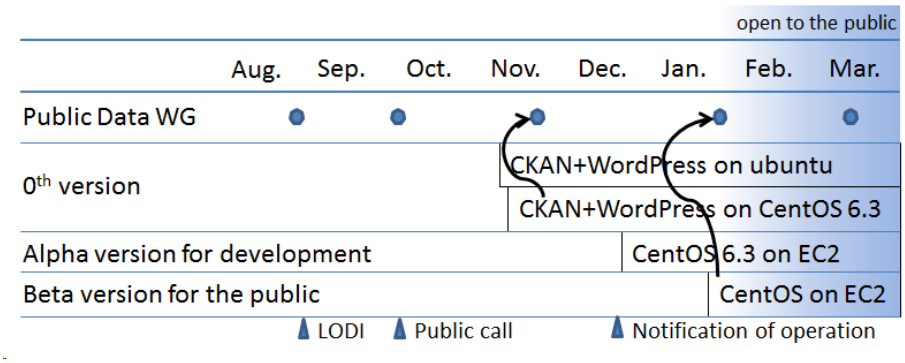


Fig. 4. Time Table of Open Data METI

The 0th version was used the reference model for discussions in operation, and also opened for the evaluation of systems at the third meeting of the Working Group. The Beta version was used for the final confirmation in the fourth meeting of the Working Group before opening to the public.

In Figure 4, the reason of gradation at "open to the public" part is the publication was gradually proceeded from the inside of the Working Group, to the limited number of Partners of Open Data METI, and an unlimited number of public people.

6.3 Step by Step Modification

Unified Page Design. The unified impression of web pages is important in web design, even if two different modules are utilized. So, we created common web materials, e.g., a logo of Open Data METI, a guiding menu for navigation, which are shared by WordPress module and CKAN module. The color of web pages is also unified so as to give the same feel between WordPress and CKAN. The original page of CKAN was drastically customized. See Figure 5 as WordPress top page and Figure 6 as CKAN top page.

Page Views and Download Number. Figure 7 shows an example of ranking page for downloaded data.

Steering by Working Group and Create Commons Attributes License. METI organized the Public Data Working Group of IT Fusion Forum in order to discuss

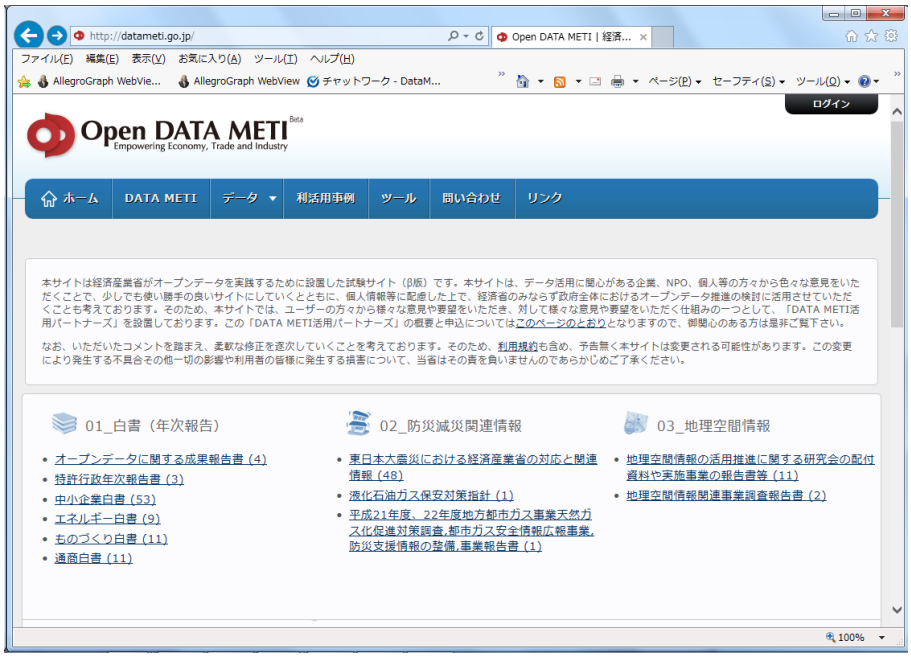


Fig. 5. Web Top Page of Open Data METI

Fig. 6. CKAN Top Page in Open Data METI

Fig. 7. A Part of Downloaded Data Ranking Page

rules and incentives needed for promoting public open data through actually publishing METI's own data. The open license that allows reuse of data resources was deeply and earnestly discussed in the meeting of the Public Data Working Group. Two board members of LODI participated in the Working Group, and they also made a contribution to the discussion. As a result of the discussion, METI decided all data in Open Data METI should be published by CC-BY in principle except white paper documents by CC-BY-ND. Therefore, the original CKAN was customized to allow the selection of any CC licenses and enabled the CC icons including CC-BY-ND to indicate the licenses on web pages. Figure 8 shows the examples of web pages with representation of some CC license icons.

7 Lessons Learned

As summary of this report, we can conclude that there are nothing remarkable things on building an open data catalog site and a linked data site except specific knowledge and skills how to build Linked Data and CKAN. We operated the work *i) with predicting problems and making decisions on the fly by agile development methodology*, *ii) solved occasional contingencies through efforts by relevant persons at work*, and *iii) met requirements from customers and the change of computational environments*. It is in the ordinary way for any software developments in inexperienced fields. The success of a project is beyond one's efforts. It was lucky for us that *i) the momentum for LOD was gained in the society* and *ii) LODI existed at the moment*. It was also happy that *iii) two board*

Fig. 8. Various CC icons on Open Data METI

members of LODI participated in the Public Data Working Group of IT Fusion Forum. Each one of three members from LODI in the operational team made an individual contribution to the success of the project upon their own attributes. One was the best one on programming CKAN in Japan. He had experiences of CKAN installation and Japanization. Another one was a web designer who had rich experience on WordPress, and the last one who was a leader of this project in LODI once directed two national projects when he worked for a company. The construction of operational team members for the project may also be a sort of luck. Hitachi Consulting and Hitachi Central Research Laboratory also perfectly performed their part in the project.

On the other hand, we found that it is difficult for ones to understand why linked data are prominent in technologies around Webs. We often experienced that it was hard to persuade not only people in general but also ones in charge of the project without any actually working examples. For example, there is no difference in appearance between one application instance built by linked data and another application instance built by Web APIs. It is difficult to understand the potential of Linked Data from the appearance. An extreme power is required to imagine the world in which all data are linked each other and people enjoy to traverse Webs from data to data over the globe. It is also usual that a few visionaries suffers lack of understanding in the society.

8 Today's Features and Future Work

The new term of Open Data METI II was already started, obtained a new member of Hitachi Systems, Ltd, who are playing a role of software development vendor. We were anxious about what went next to this project at the end of the project. We were afraid that this would become one of evaluation tests and the site would be closed. However, after the speech of METI at the end of the period of the project and METI's private report to the United Nations, we were certain of the next stage of the project. The Open Data Charter at the G8 Summit held at Lough Erne in 2013 also pushed the Government attitude to open data.

The Open Data METI site is changed step by step at this moment. The primary mission of the new project is establishing a lasting and genuine open data site in METI. It includes new data sets, new functionalities, e.g., a bulletin board, previewing data, what's new, and suggestion to users. Figure 9 shows the increasing number of data sets at the moment of writing this paper.

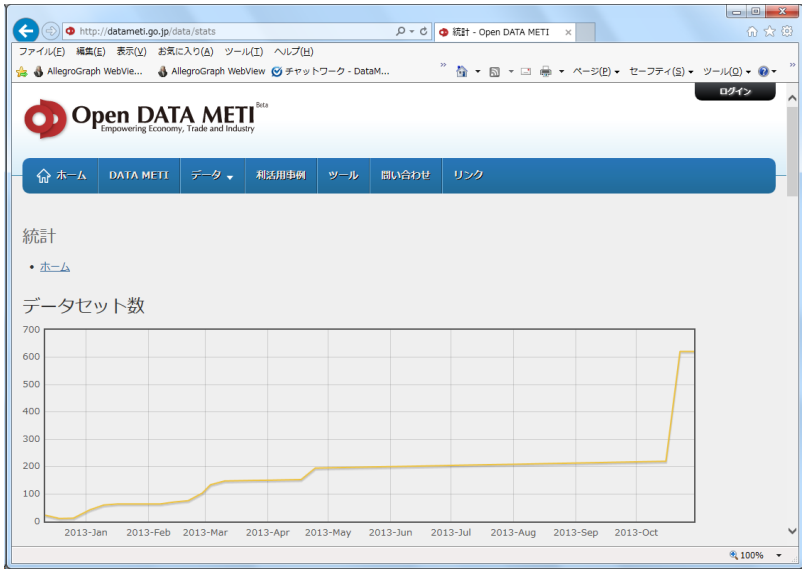


Fig. 9. Increase of Data Sets of Open Data METI

However, the most remarkable item of the development from the viewpoint of LOD is setting up terminologies of meta-tags, which are utilized in CKAN to search data sets. METI and Information-technology Promotion Agency (IPA) has started the Promotion Committee for Infrastructure for Multi-layer Interoperability (IMI) at this September, aiming the introduction of common vocabulary used in the Government beyond the boundary of Ministries and Agencies. This IMI vocabulary is going to be reflected to meta-tags in Open Data METI. In

addition, a requirement of putting down in both Japanese and English expressions on the web pages is now coming up as a next requirement. LODI is again a member of the Open Data METI II operation team. We will contribute the population of LOD in Japan by making efforts to build Open Data METI and other activities.

References

1. Cyganiak, R., Reynolds, D. (Eds.): The RDF Data Cube Vocabulary. W3C Working Draft 12 March 2013, http://www.w3.org/TR/2013/WD-vocab-data-cube-20130312/ W3C (2013)
2. Takeda, H., Kato, F., Koide, S., Matsumura, F., Ohmukai. I., Kobayashi, I., Iwayama, M., Asano, Y., Hamasaki, M.: Presentation of Statistical Data and their Relationship as LOD. 27th JSAI, N4-OS-10b-6, (2013)
3. Asano, Y., Iwayama, M., Takeda, H., Koide, S., Kato, F., Kobayashi, I.: Template for Converting Statistical Data to RDF, 12th FIT, F-034 (2013)

EDI support with LOD

Akihiro FUJII, Shusaku EGAMI, Hiroyasu SHIMIZU

Faculty of Science and Engineering, Hosei University 3-7-2Kajino-cho, Koganei-shi, Tokyo, 184-8485 Japan

E-mail: fujii@hosei.ac.jp

Abstract A wide variety of mechanical parts circulate in the field of manufacturing. It is often the case that a set of codes for products is necessary for EDI (Electronic Data Interchange) that supports distributions of them. Generally such product codes are maintained by a certain organization consists of related companies, business communities, so forth. While *Linked Open Data* is getting popularity it is quite meaningful to add semantically rich information to such data resources in order to enhance business transaction usability. As a case study, this paper explains LOD of so called "N-ken Code". This set of numerical data is based on a well-specified product identification code that has been utilized over 10 years in a certain business community in Osaka, Japan. We have produced LOD from them as well as implemented a mashup application based on the data set. The application is for a new CAD service in the field of mechanical engineering. We discuss perspective in utilizing LOD for variety of EDI requirements.

Keywords: LOD EDI CAD SVG mashup Screw

1 Introduction

LOD (Linked Open Data) has a potential in supporting existing EDI (Electronic Data Interchange) procedures of business transactions in various aspects. EDI is an acronym for Electronic Data Interchange,[1] and is defined as the "means to represent a variety of documents—management, commercial, and transfer—using a set of structured alpha-numeric linguistic strings based on standardized rules.[2] The use of EDI enables enterprises to exchange transaction information with other companies in a more effective way. Introducing LOD to EDI functions, we can expect enhancement of the scheme.

In this paper our interest is focused on the distribution of mechanical parts and related industry sector. A wide variety of mechanical parts circulate in the field of manufacturing. A set of codes for identifying products is necessary for EDI that supports distributions of these products. Commonly such product codes are maintained by a certain organization of business community. EDI has been considered an indispensable element of business-to-business transactions in the field.

There is so called “N-ken Code” of *screw* identification code that is EDI supporting numerals of a business community for in Osaka, Japan[6]. We have produced LOD from them. A mashup application based on the LOD is provided for a new CAD service in the field of mechanical engineering. In this paper we discuss perspective in utilizing LOD for variety of EDI requirements with this example.

This paper is organized in the followings. In the next section, overview of research activities in terms of business data linkage is introduced. In section 3, typical problem occurred in applying EDI is explained. In 4, the configuration of a LOD and related application software is explained. The system’s architecture style is based on so-called ROA(Resource Oriented Architecture) and RESTful Web API. We will see the detail of the structure of the application. Section 5 is the concluding remarks.

2 Related Research

There have been not many LOD activities applying to EDI so far, for the best of our knowledge. The following is rather general perspective about high-level data linkage in business transactions. Combining input and output from a Web API, it is necessary to describe the API and input and output as a series flow in order such as EDI. One of the research example is the European FAST Project, which has been funded from the ICT Policy of the European Commission to research enterprise information system methodologies. This project is the successor to the “SecSE (Service Centric Software Engineering) Research Project. The goal of the projects was to construct a platform that would allow multiple enterprises to coordinate their services aiming to develop service-centered applications and also effective methods and tools for use. In [3], it analyzed of a number of data-link patterns between enterprises using Web APIs, and used the results of this analysis to create models.

3 Problem to Solve

3.1 Perspective over EDI in Manufacturing related industry in Japan

Generally among the enterprises in Japan that are affiliated with supply chains of large distributors and large manufactures, 80 to 90 percent of them have already introduced EDI. However among small- and medium-sized companies (SMC), EDI usage in transactions is much smaller, and the introduction ratio is estimated to be 10 percent or less. So that the introduction of LOD for EDI associated features has high potential to improve business transactions in the area of manufacturing and products circulations.

3.2 Multiple Screen Phenomenon

What should we cope with in terms of EDI, when we are able to introduce LOD related technology for solving such issue? A typical problem that should be solved is

that the occurrence of the multi-terminal phenomenon has been one of the impediments, which causes problems including the following:

1. Because of multiple independent EDIs for contacting each business customer, the EDI operator has to switch from screen to screen (many Web browser windows).
2. Data is not always compatible with that of the existing in-house information system. The operator may have to re-enter the data manually.
3. The user is billed for each EDI and for each user ID.

In this paper, a pilot implementation to cope with above issue by introducing LOD features based on the database.

4 Mashup Application

The most important and labor-intensive process in EDI-system construction is to put a common standard in place, and to resolve the associated management problems in line with it. To the former aspect of the challenge, LOD is highly promising. Even so, we have to cope with the later problem and put efforts on it. In the following use case scenario, we would like to describe an example usage of LOD in EDI.

4.1 N-ken Code

N-ken is a business community of screw dealers. They have been collaborating about EDI introduction and knowledge sharing among member companies. It consists of many SMCs which deal with screws in Osaka, and their code is defined for the group's business activities and has been used over 10 years. This code covers 250 thousand items in 3400 categories.

While Linked Open Data is getting popularity it is quite meaningful to construct semantically rich information over such data resources in order to enhance business transaction usability. As a case study, we will explains so called "N-ken Code" of screw identification code that is EDI supporting numerals of a business community for in Osaka, Japan. We have produced LOD from them. Once LOD is created, at lease for transactions among companies which use same N-ken Code, EDI function becomes more flexible and possesses potentials for new feature in business transactions. We are going to show a pilot implementation in the followings.

4.2 System Configuration

N-ken Code LOD is provided as RESTful Web API based on ROA design paradigm [4]. The resource oriented architecture of the design can provide a suitable interface between the client and the server in this representation. In the architecture, the resource is divided into each URI and names are given to them. These specified resources are accessed through Web service schema. The data is managed by a general Java framework.

4.3 Mashup Application

Here CAD data associated EDI transaction is introduced. In the application, we focused on SVG (Scalable Vector Format) uses XML (Extensible Markup Language). SVG is a file extension for a vector graphics image file format created by the W3C to describe such images by mathematical equations. With the addition of technologies like the canvas element, CAD data processing is possible over normal Web browser.
For our mashup service, we have utilized Japanese DBpedia. It is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. There is Japanese sub-version of DBpedia that we used for our mashup application. N-ken LOD may linkage with other LOD by referring key words related manufacturing such as “neji(screw)”. The LOD allows you to ask sophisticated queries against Wikipedia, and to link the different data sets on the Web to Wikipedia data (Fig.1). Suppose you have your CAD design sheet of planed product. For the estimation of cost of assembling parts, it is quite useful if you’ll be able to refer inventory database of screw which may be provided by several companies. We take mashup example for our N-ken LOD with SVG data produced common CAD application (Fig.2).

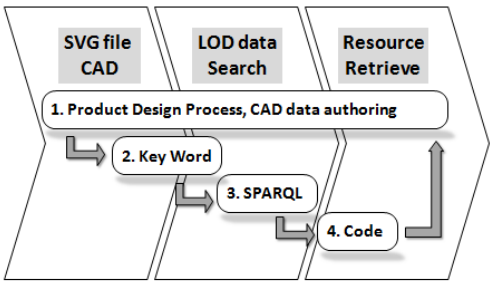


Fig. 1. Process of Mashup

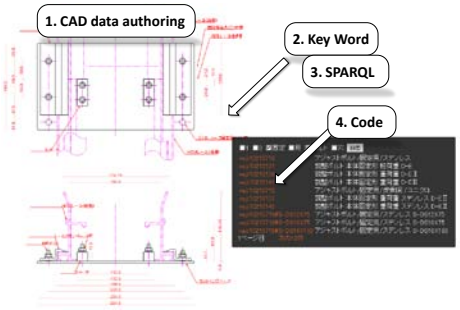


Fig. 2. CAD-Code Mashup linkage Application

SVG over LOD.

Although SVG (Scalable Vector Graphics) is not yet fully matured technology, there is an important aspect in this example application. SVG format has high potential to produce new services especially CAD data related business practices in the context of LOD applications. When we design LOD platform, RESTful Web API is standard architectural style. When we provide a CAD data, URI can be uniquely associated with some DOM (document object modeling) representation. Ongoing pilot project touches upon mashup editor for DOM managements based on the concept that chunk of data could be provided through Web API from LOD database.

HR/MR.

The above mentioned scenario may help the designer to choose most favorable item for the design. And such function could be provided either HR(human readable) inter-face or MR(machine readable) one. The mashup process can be realized in both format and even those interface schema could be provided from different source through Web API . We prepare parsers for both HR and MR interface for the LOD.

Web Site

We have opened a web site for this LOD (http://monodzukurilod.org/), named after "monodukuri" which is the Japanese word for manufacturing. It is officially authorized LOD practice in Japan. Not only for just publishing N-ken code LOD, the site is also aiming to be a center of information of utilization of this type of LOD in manufacturing industry sector. We are planning to enhance the collections of service applications starting with above explained CAD applications with having help through industry-academy collaborations. (Fig. 3)

Fig. 3. http://monodzukurilod.org/

5 Conclusion

As the cloud computing becoming more and more popular in enterprise IT-system, the cost of implementation of EDI may become lower. For the direction, a large amount of standardization effort is currently being taken on the premise of wide-spread use of the cloud, for example in [8]. In the mean time the standardization of EDI in utilizing LOD over cloud environment is one of the key objectives in this trend[5][7][9].

Author of this article consider that introduction of LOD has several positive effects over EDI practices in general.

A) Lowering of (initial) cost for EDI introduction

B) Flexibility of EDI and the software running on it, capable of agile response to changes in the business environment of the enterprise

C) Ease of participation in the EDI promoted by an industry group

Introduced Mashup application in this paper is one of academic-industrial collaboration example. Between a university and an individual industry group, could prove productive, whereby the university could provide a platform for experiments. It may be difficult to reach agreements for standardization effort when it comes to break down to realistic business practice. However, enterprise IT system over the cloud environment, especially in utilizing semantic feature, is an ongoing effort of many companies and business communities. We hope that the lesson learned in our approach should be reviewed and modified for other trials.

References

1. http://www.w3.org/TR/XPROC/, An XML Pipeline Language
2. Tom Heath, Christian Bizer, (Translation in Japanese, Hideaki Takeda, et al),"Linked Data: Evolving the Web into a Global Data Space", Kindai-Kagakusya Publishing, 2013
3. Elisabetta Di Nitto, et al,"At Your Service: Service-Oriented Computing from an EU perspective", MIT Press. 2009.
4. Jerome Louvel, Thierry Templier, Thierry Boileau, "REST in Action", Manning Publications, 2013
5. Next Generation Electronic Commerce Promotion Council of Japan (ECOM): http://www.ecom.or.jp/
6. Mostafa Hashem Sherif, "Standardization of Business-To-Business Electronic Exchanges," IEEE, Standardization and Innovation in Information Technology: SIIT 2007 Proceedings, 2007
7. Japan Information Processing Development Corporation (JIPDEC), "An investigation of actual conditions of EDI/electronic tag application in Japanese industries," March 2010
8. "GCommerce," A case study of Microsoft Corporation, Dec. 2010 http://www.microsoft.com/casestudies/
9. Akihiro Fujii, "Standardization of electronic commerce in the cloud environment and its future evolution" Technology Management for Emerging Technologies (PICMET), 2012 Proceedings of PICMET '12, pp. 2207- 2213

On Implementing a SPARQLoid Query Coding Support – Vocabulary Discovery for Queries with Weighted Ontology Mappings

Hiroki Noguchi, Takahisa Fujino, and Naoki Fukuta

Graduate School of Informatics
Shizuoka University
Hamamatsu, Japan
{gs13025@s,gs12033@s,fukuta@cs}.inf.shizuoka.ac.jp

Abstract. When writing SPARQL queries that will access to multiple data sources (i.e., a federated querying), one has to know the corresponding vocabulary where one can name the property URIs to be used in the query, and to know the exact URIs for such vocabulary is not easy task since those vocabularies are often separately defined and managed. To avoid such a situation, the techniques to use some ontology mappings in a SPARQL query have been actively developed. In this paper, we present our preliminary implementation of SPARQLoid query coding support system that can utilize familiar vocabularies that may be contained in some ontology mappings by using weighted ontology mappings associated to specified keywords in the query.

Keywords: SPARQL, ontology mapping, Linked Open Data

1 Introduction

The use of ontology mapping, alignment, and matching techniques [5, 6, 14] could be useful for writing queries when it allows us to use any ontologies in the queries unless those ontologies are not directly used in the endpoint [7]. There are some ontology alignment methods to produce precise and crisp mappings among their entities. However, in some cases, producing a precise mapping is very difficult[5, 14], especially when the two ontologies have semantic gaps. Alignment methods generally require much time to produce an alignment when the two ontologies are huge. Hence, an efficient way to get an alignment has also been proposed (e.g.,[18]). To utilize these ontology mappings that have a variety of confidence values, it is important to explicitly control their effective use in a query, since without such control a query can produce so much unwanted data in the process of its execution so that they cannot be processed. Although there are some approaches that can utilize mappings on querying with SPARQL[13, 16], they will not provide a functionality to explicitly control how to process queries with such 'somewhat unreliable' weighted ontology mappings. To solve this issue, SPARQLoid, a small extension to SPARQL that allows us to utilize such weighted ontology mappings effectively, has been proposed[8, 9].

However, on SPARQLoid, finding a base ontology and understanding their vocabulary defined in the ontology is still remained as one of the user's tasks, and it would be still difficult for various novice users. In this paper, we present our preliminary implementation of SPARQLoid query coding support system that can help users to utilize vocabularies that users are familiar with in the query, by providing a recommendation mechanism for sufficient ontologies and endpoints in combination with weighted ontology mapping-based querying functionalities implemented on SPARQLoid engine.

2 Background

2.1 Linked Open Data and SPARQL

The datatype of Linked Open Data (LOD) is basically written by RDF like Semantic Web. As an example, there is DBpedia that has been converted from the original Wikipedia content to the form of LOD. Furthermore, to realize "Open Government", various countries have been spent much efforts to realize their open data access site, such as "data.gov". In 2013, there are at least 447 registered public SPARQL endpoints with over 31 billion RDF triples, which are interlinked by around 504 million RDF links, and the amount of data are increasing[1].

There are many SPARQL endpoints (e.g., DBpedia[3]) that allow us to retrieve and see the data by queries that are written in SPARQL query language. We can retrieve data not only from one endpoint but also from several endpoints using federated queries [11, 12, 15]. However, it is not always easy to understand the ontology used at an endpoint well enough to write a query based on it. The main reason could be that, often an ontology is constructed with some specific terms that were not used in others. Furthermore, it often lacks detailed documentations.

2.2 Ontology Mapping

In open systems such as Semantic Web, how to deal with heterogeneity of data or ontology is becoming an important issue to be solved. To solve the issue, lots of efforts have been done about developing and evaluating approaches for ontology mapping[6]. Also, some practical software that implemented such ontology mapping techniques are widely used (e.g. LogMap[10], Alignment API[2], etc).

LogMap uses HermiT[3] as a reasoner to allow the method to generate higher quality mappings based on the reasoning results among the ontologies. Therefore, to fully utilize it, we need to prepare target ontologies that are ready for reasoning on HermiT. In this way, sometimes we need to utilize various approaches to generate mappings and their quality would be varied. For example,

[1] http://sparqles.okfn.org/availability
[2] http://alignapi.gforge.inria.fr/
[3] http://hermit-reasoner.com/

AlignmentAPI allows us to generate mappings based on string similarity or other mapping methods that users have chosen. It generates mappings in short time but it may produce mappings with lower precision.

In OAEI2012[1], 23 types of mapping generation methods were presented. We prepared a set of mapping generation and updating mechanism to obtain results by querying to heterogeneous LODs.

2.3 SPARQLoid

SPARQLoid enables users to control their utilization of different ontology mappings as well as confidence values in a query[7–9]. It allows users to choose and utilize ontology mappings to obtain the data in the target endpoint and specify the criterion for sorting the results and cutting output data that are far from accurate.

Here, we show a typical SPARQLoid query in listing 1.1. Notice that the SPARQLoid query in listing 1.1 obtains the matched instances as ?a, ?b, and ?c using the vocabularies in the conference namespace. However, each target endpoint to be accessed by a URI (*targetEndpoint* or *federatedQueryEndpoint* in listing 1.1) is expected to be constructed using each different ontologies.

```
SELECT ?a ?b ?c
WHERE
{
        ?a rdf:type conference:Regular_author
        ?a conference:has_the_last_name ?b {
                SERVICE < federatedQueryEndpoint > {
                        ?a conference:has_the_first_name ?c
                }
        } RANKING < targetEndpoint > {
                conference:Regular_author *0.5 +
                conference:has_the_last_name *0.2
        } THRESHOLD < targetEndpoint > {
                conference:Regular_author =0.3 ,
                conference:has_the_last_name =0.3
        } RANKING < federatedQueryEndpoint > {
                conference:has_the_first_name *0.3
        } THRESHOLD < federatedQueryEndpoint > {
        conference:has_the_first_name =0.2
        }
}
```

Listing 1.1. Example Query in SPARQLoid

For example, the class *conference:Regular_author* and the property *conference:has_the_first_name* require appropriate mappings to run the queries in listing 1.1 on the endpoints. In listing 1.1, *conference:Regular_author* was specified as more important than *conference:has_the_first_name* by the ranking clause. In case of a federated SPARQL querying, the weights of mappings are merged in the specified ways. In listing 1.1, some threshold values are specified to cut the

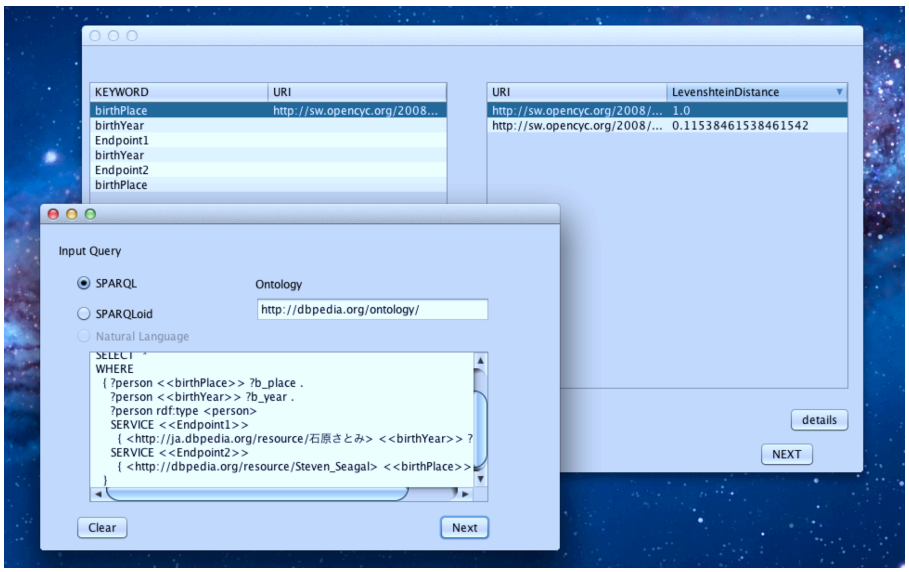


Fig. 1. Overview of System

irrelevant results that come from lower-weight mappings. For example, the mappings for *conference:Regular_author* whose confidence values are higher than 0.3 are used to execute the query on the target endpoint.

3 Proposed System

Our system, SuPARQooL(SUPport system for spARql Query cOding with Ontology mapping on sparqLoid) realizes a WOM(weighted ontology mapping)-based SPARQL query coding support with vocabulary discovery support technique. The users can write a query in SPARQL or SPARQLoid[7–9] that includes some natural language texts as the concepts (i.e., types) or properties to be used in them, and the system recommends sufficient URIs for them. Although other systems[4, 13, 16, 17] are not aware of weighted ontology mappings, SPARQLoid allows us to utilize weighted ontology mappings effectively. The system semi-automatically finds candidates of vocabularies or the entire ontologies that can be mapped to the users' familiar ontologies. Then, after clicking the "NEXT" button on the main window (see Figure1), our system calculates the matching degree to the endpoints that have been registered to the system. The endpoints in the list are scored and ranked based on the matching degrees. By pressing the button "more", the user can see the endpoints to be accessed and test-run some queries and investigate the endpoint itself to confirm it is sufficient to be used based on the investigated details of the endpoint. When the user has successfully coded the queries to be used, the system will prompt the user to select the endpoints to be accessed from the list. Then, based on the ontologies used in the endpoints, the SPARQLoid query is translated to a standard SPARQL

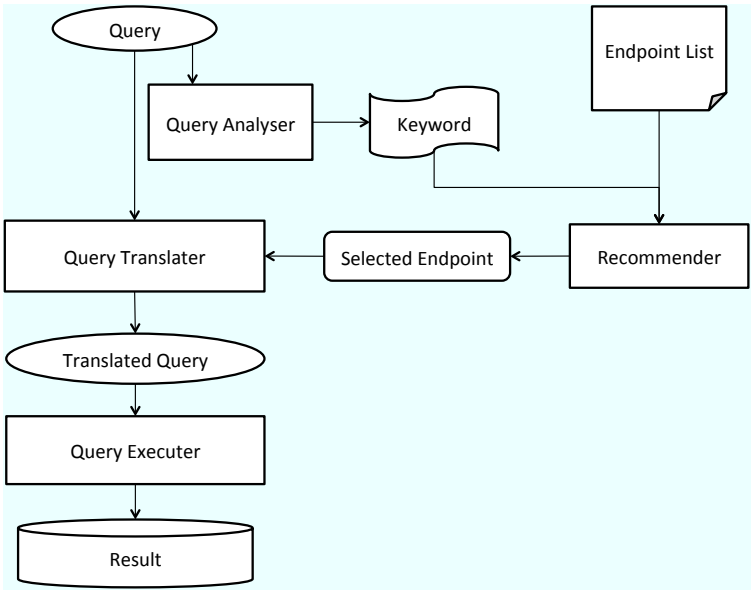


Fig. 2. Processing model of our system

query to be run on various SPARQL endpoints. The user can manually rewrite the generated SPARQL query for fine tunings of performance or adjust detailed behaviors of the query. The results of the query are also shown in the system, as well as converting the query that can embed some external applications to access external SPARQL endpoints.

Figure2 shows the structure of our preliminary implemented system. Query Analyser analyzes and extracts keywords. Recommender recommends endpoints by the specified keywords or related endpoint list. Query Translater translates SPARQLoid queries to standard SPARQL queries that can be run on various endpoints.

4 Conclusion

In this paper, we presented a SPARQL query coding support system with vocabulary discovery function for queries with weighted ontology mappings, which can be utilized in SPARQLoid queries. Our system allows users to query various open endpoints with weighted ontology mappings, as well as finding such endpoints and user's familiar base vocabularies to be used in the query.

References

1. Aguirre, J. L., Eckert, K., Euzenat, J., Ferrara, A., Van Hage, W. R., Hollink, L., Meilicke, C., Nikolov, A., Ritze, D., Scharffe, F., Shvaiko, P., Zamazal, O. Šváb, Trojahn, C., Jiménez-Ruiz, E., Grau, B. C. and Zapilko, B. Preliminary results of the

Ontology Alignment Evaluation Initiative 2012. In: Proc. of 7th Ontology Matching Workshop (OM2012), at International Semantic Web Conference (ISWC2012), 2012.
2. Atencia, M., Borgida, A., Euzenat, J., Ghidini, C., Serafini, L.: A formal semantics for weighted ontology mappings. In: Proc. of the 11th International semantic web conference (ISWC2012). (2012) 17-33
3. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: DBpedia: A Nucleus for a Web of Open Data. In: Proc. of the 6th International Semantic Web Conference (ISWC2007). (2007) 722-735
4. Bizer, C., Schultz, A.: The R2R Framework: Publishing and Discovering Mappings on the Web. In: Proc. of the 1st International Workshop on Consuming Linked Data (COLD 2010). (2010)
5. Duan, S., Fokoue, A., Srinivas, K., Byrne, B.: A clustering-based approach to ontology alignment. In: Proc. of the 10th International Semantic Web Conference (ISWC2011). (2011) 146-161
6. Euzenat, J., Shvaiko, P.: Classifications of ontology matching techniques. In: Ontology matching. (2007) 61-72
7. Fujino, T., Fukuta, N.: A SPARQL Query Rewriting Approach on Heterogeneous Ontologies with Mapping Reliability. In: Proc. of the IIAI International Conference on Advanced Applied Informatics (IIAI-AAI2012). (2012) 230-235
8. Fujino, T., Fukuta, N.: SPARQLoid - a querying system using own ontology and ontology mappings with reliability. In: Proc. of the International Semantic Web Conference (Posters & Demos) (ISWC2012). (2012)
9. Fujino, T., Fukuta, N.: Utilizing Weighted Ontology Mappings on Federated SPARQL Querying. In: Proc. of the 3rd Joint International Semantic Technology Conference (JIST2013). (2013) (to appear)
10. Jiménez-Ruiz, E., Grau, B, C.: LogMap: Logic-based and Scalable Ontology Matching. In: Proc. of the 11th International Semantic Web Conference (ISWC 2011), Part I, LNCS 7031, pp.273-288, 2011.
11. Ladwig, G., Tran, T.: Linked data query processing strategies. In: Proc. of the 9th International semantic web conference (ISWC2010). (2010) 453-469
12. Ladwig, G., Tran, T.: SIHJoin: querying remote and local linked data. In: Proc. of the 8th European Semantic Web Conference (ESWC2011). (2011) 139-153
13. Makris, K., Bikakis, N., Gioldasis, N., and Christodoulakis, S.: SPARQL-RW: Transparent Query Access over Mapped RDF Data Sources, In: Proc. of the 15th International Conference on Extending Database Technology (EDBT2012), (2012).
14. Noy, N. F.: In: Proc. of the Ontology mapping. In Steffen, S., Rudi, S., eds.: Handbook on Ontologies. (2009) 573-590
15. Quilitz, B., Leser, U.: Querying distributed RDF data sources with SPARQL. In: Proc. of the 5th European SemanticWeb Conference (ESWC2008). (2008) 524-538
16. Rivero, C., Hernandez,I., Ruiz,D., and Corchuelo, R.: Mosto: Generating SPARQL Executable Mappings Between Ontologies. In: Proc. of the 30th International Conference on Conceptual Modeling Demos and Posters. (2011).
17. Rivero, C., Hernandez,I., Ruiz,D., and Corchuelo, R.: Generating SPARQL Executable Mappings to Integrate Ontologies. In: Proc. of the 30th International Conference on Conceptual Modeling. (2011) 118-131
18. Seddiqui, M.H., Aono, M.: An efficient and scalable algorithm for segmented alignment of ontologies of arbitrary size. In: Web Semantics. 7(4). (December 2009) 344-356

Translation of Various Bioinformatics Source Formats for High-Level Querying

John McCloud and Subhasish Mazumdar

New Mexico Institute of Mining and Technology
Socorro, New Mexico, U.S.A
{amccloud, mazumdar}@cs.nmt.edu

Abstract. Representations of genomic data is currently captured for use in the field of bioinformatics. Each of these representations have their own format, caveats, and even specific "sub-languages" used to represent the information. This makes it necessary to use different specialized programs to make sense of — and then query — the data. Unfortunately, this means a great deal of lost time necessary for learning both the tool and the format itself in hopes of extracting details that can then later be used in answering queries. There is, therefore, a great need for a general purpose tool that can address all of these different formats at once: a tool that will abstract such low-level, format-specific representations of raw data to a higher, operating level with terms that are immediately recognizable to experts within the bioinformatics and biology domains. In addition, it will help biologists engaged in research pose questions that were not anticipated by the authors of the special purpose software built for the data files.

1 Introduction

Ontologies are a way to formalize conceptual representations of a domain: conceptual entities, their constituents, and/or how they relate to other entities. Such formalizations are necessary to bring a coherent, shared view of items and processes within domains, removing ambiguity in both definition and relationship. There are many issues, however, when it comes to automatically classifying such information, especially when one is attempting to merge ontologies of both related and unrelated domains. Such difficulties are compounded when the data within such heterogeneous sources are obfuscated in some manner — unable to simply be decoded by language processing or recognition through keywords. Instead, an expert in the domain is generally needed to make sense of such data.

In this paper, we explore the issue of both merging different low-level data sources and making them accessible to queries at a higher level of abstraction, addressing a current need in the field of bioinformatics and biology. We present a way to build robust and correct ontologies that allow querying across various, low-level data formats. We show that this system is very beneficial to many researchers in the bioinformatics domain, giving them the ability to combine data from all their various data formats and also pose queries that are hard to formulate within the dedicated software systems that come with the data files.

If we only examine DNA sequencing data, there are already many different formats with their own definitions. SAM, FASTA, BEM, FASTQ, PIR, MSF, CLUSTAL, and so on, along with different annotation formats BED, GTF, GFF3, GVF, and VCF. Right now, biologists have a lot of dedicated software that has been built to deal with one or a few data formats as these. The problem is that there is a real need for a general purpose tool that allows biologists to explore all the data that

they have in any format, but there is no tool that allows them to look at anything they wish in the data. Instead, they are forced to use whatever definitions and terms those specialized pieces of software define.

We have developed a solution to this issue by manually mapping terms in the ontology for these various, low-level data sources. Alongside an effective, simple target at an operational level for querying, contextual explanation for the ontology-level classifications are presented, giving a more cohesive understanding for adding new information to an ever-growing map of relationships in data.

The approach does not have only one, completely unambiguous ontology from which all knowledge representation stems. Instead, it allows for multiple domain-level ontologies to coexist at once in a shared, operational level (from which queries are made).

This paper is structured as follows. In the next two sections, we present a review of related work and a short primer on biology. Subsequently, in section 4, we delve into design issues. A case study dealing with files using the SAM format is presented in section 5, followed by respective sections of short analysis and concluding remarks.

2 Previous Work

Much of the work that has been done in linking data sources usually starts out as a way to combine low-level ontologies. While such research is extremely valuable in the work presented here, we remind the reader that we are not merging well-defined ontologies, but instead linking the raw or semi-raw data that a researcher in the field may collect — and linking *data formats* — with larger ontologies. Our research attempts to implement ideas from the previous work below.

2.1 C-OWL

Contextual OWL took the original language of OWL and made a system that assumed a single, global ontology for all term interpretation would not provide suitable representation for particular *local* cases. By this extension, the research realized that context within a local ontology must be maintained. In order to represent ideas from knowledge at the local and higher levels, contextual mappings are created with relationships to the various levels of abstractions (from local to higher levels) linked by *bridges* [1].

These bridges are similar to what is represented here in this research, but they are much more abstract, often times explicitly mentioning the levels of abstraction (where a given term within the ontology resides), and the relationship to another term in the bridge. (This allows for such mappings as $Term_A$ is identical to $Term_B$ from low-level ontology A and high-level ontology B.) The bridges and their mappings allow for denoting *contextual levels of specificity* of terms, but do not describe how to actually derive terms from low-level sources.

This is because both source-level and target-level information they deal with are assumed to be OWL ontologies themselves. In our research, we assume all data from the low-level source is *not* within any such formalization to begin with, but must instead be built from RAW data and linked to a higher-level ontology. Much as in OWL, though, the source-level definitions maintain a useful context, and terms within that context must be dealt with differently than terms with other contexts (from other sources).

2.2 KRAFT

Perhaps the most inspiring work for this research comes from the KRAFT project, which recognized the necessity of a shared ontology amongst many low-level ontologies and manual linking from the low-level definition up to the single, shared one [2]. Their agent-model design also includes a distributed solution, whereby one user agent can receive knowledge from another user agent (and his definitions) across a network.

By creating a new language that acts as a link between the shared ontology and the low-level sources, this Constraint Interchange Format (CIF) made it possible for user agents to make queries against data not defined in the shared ontology. This turned the query into a Constraint Satisfaction Problem. A user could then ask a question, and these constraints would be developed for the various low-level sources, using their particular, individual languages. This kept a user from needing to know the format and definitions of anything other than the shared ontology.

The issue with the KRAFT system is that it allows mappings to be arbitrary and possibly completely wrong. This, however, is an acceptable drawback of the system, since we must trust those knowledge experts who deal with their data regularly to define it and expose caveats, fringe cases, and easy-to-miss constraints within the defined resources.

Additionally, it is not clear if the mappings allow for any links larger than one step. That is, multiple transformations from low-level heterogeneous source and the shared ontology. Certain multiple steps may be necessary to produce an effective, overall mapping. Furthermore, combining these intermediate definitions may be an important task that should be available to provide richer explanation from low-level details.

3 Primer on Biology and Sequencing

For readers unfamiliar with the concepts and terms we will use shortly, this section provides a quick primer on the central dogma of biology. This begins with *DNA*, an extremely long sequence of nucleotide bases, which makes up almost every living thing (*genome* is another term used to refer to the total genetic data of an organism, and is more general); and all living organisms are a collection of cells containing a certain number of recognizable *chromosomes*, wound up in tight bundles around proteins; each chromosome contains one or two DNA molecule(s).

DNA is a code that contains all the genetic information necessary for every type of *cell* in our bodies; from cells that make up our eyes to cells that form our skin. Each cell is specially tuned to perform a certain way, depending on which portions of the DNA are manifested in it and how the organism should respond to the environment it lives in.

Portions of the DNA have regions called *genes*, and when these genes are *expressed*, it can result in the creation of tiny, biological actors within the cell, called *proteins* (and if expressing the region does result in proteins, we say that it is a *coding region*), which perform all types of functions. Each cell usually only contains a certain subset of proteins, and this is dependent on the type of cell in which the proteins are built in. When an organism exhibits a certain type of protein, we say it expresses the gene that codes for the protein; this is referred to as *gene-expression*.

Usually, cells produce a reasonably normal amount of proteins in each cell (as per the cell's type). However, when the pressure from the environment results in some shift in the need of the organism, a cell's DNA can adapt to create many of a particular type of protein. This is usually to protect the organism or help it cope with whatever environmental stresses it needs to respond to. Such shifts in gene expression are interesting to examine in biology, because it can assist in

discovering new regions of the genome that were not previously known. Often, such discoveries are made by comparing an observed DNA coding region against a *reference genome*; an agreed upon reference for an organism's total genetic material.

For the purposes of this paper, it also helps to understand sequencing and sequence alignment. Sequencing is the process of discovering the actual order of the nucleotide bases that constitute the organism's DNA. Once sequenced, the various sections of the DNA are matched against a reference to create a standard alignment of the data.

As a result of sequencing, one may find a series of *reads* (portions/fragments of genetic data) that are very similar, appear in many locations of the genome, or sometimes just shifted slightly by location. When the locations and lengths are overlaid one on top of one another, the ones with the greatest overlap – these overlapping *pileups* — leads to stronger confidence in accurate representations of that fragment of the genetic code; they are referred to as *consensus sequences*. These consensus sequences are important, because one might conclude that a gene exists there, with gene expression strength depending on how many reads were overlapping along this same area [8].

4 Design

In building the ontology design, these core goals have been identified:

- An expert in the domain should require only the knowledge of her domain and not the additional definitions in the file formats for querying and query resolution. We must separate the knowledge of the low-level, specific data formats from the domain expert's "standard" knowledge of the domain.

- We must create intuitive methods for domain experts to implement additional mappings as new data formats present themselves. A tool that can deal with multiple data formats should be easily extensible.

- Queries at the topmost level must be independent of data formats. Seamless execution on different files (and therefore, different formats), will allow the data to be cross-referenced, shared, and pooled together.

4.1 The Ontology's Structure

In describing the architecture of this ontology design, it helps to think of a staircase. At the top of the staircase are the top-level definitions, while at the bottom are the source-file definitions. The source-file definitions are transformed at each ascending step, becoming a little closer to the shared-level definitions at the top. Definitions at the highest, top step of the staircase are what experts in their domain might use to describe a question.

The definitions at the lowest, source-level step are what each particular file format defines. These are terms that are designed by individuals crafting file formats. Such definitions can be arbitrary, and do not generally share the language of the high-level. Instead, they define terms that are terse, complex, or otherwise lacking in sufficient identification as to be used by a general expert of the domain. For instance, DNA Polymerase, in such a low-level, could be defined as DNA_P1 or DP1,

but this is not how biologists refer to it; this is how someone who built the file format has chosen to represent it.

The goal is to allow queries using the concepts of the high-level to pull information from the lower levels, even though their definitions may not immediately align. Their definitions are mapped from the source-level to the shared-level (and vice-versa), transforming a definition in one format, to intermediate definitions, and then finally to the agreed upon definitions of terms used by an entire field or domain.

Take this staircase analogy and imagine that the steps are now layers. There are several layers, and at each layer L a given definition D exists. In order to go from one layer to another, some transformation defined by a function is performed on that definition. Sets of such functions are contained within bridges B.

Users, such as biologists, may ask queries such as: ***Which alignment is repeated the most in the HFD1.sam data, and does it represent a new gene?*** This query aims at finding possibly unknown genes that may have been exposed by environmental stress in the experimental, sequenced organism.

Next, let us break down the query into the high-level terms (and their attributes) that need to exist at the shared-level.

- **Chromosome** (*Number*)
- **Gene** (*Name*, *Start Location*, *End Location*, *Expression*)
- **Sequence** (*Start Location*, *End Location*, *Length*, *Bases*)
- **Alignment** (*Unique*, *Total Matches*, *Mapped*, *Mismatching Positions*)

Since these terms (and attributes) are used in queries, a domain expert must build bridges for them from the source-level up to the shared-level (Fig. 1).

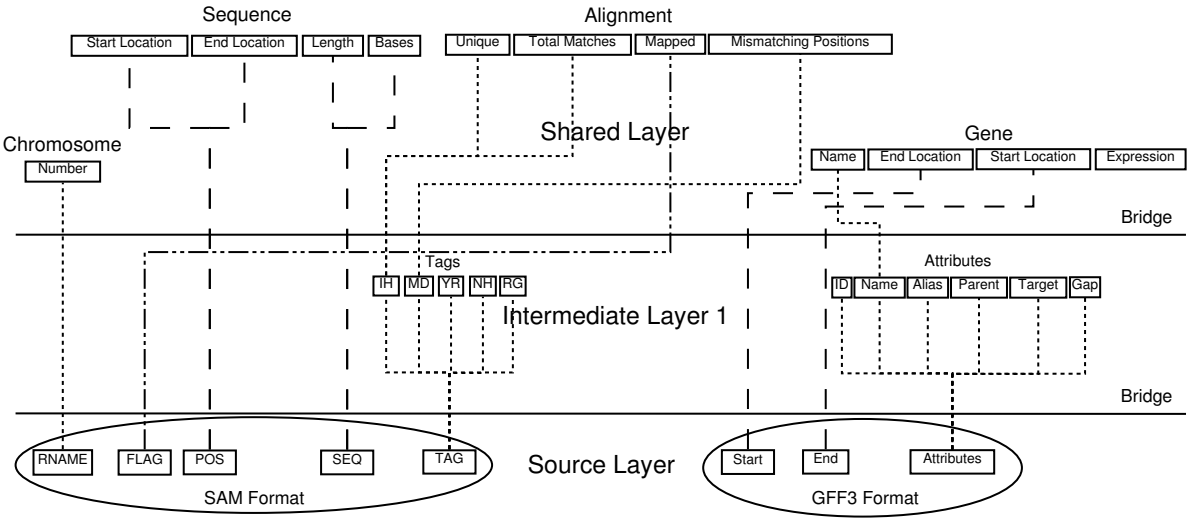


Fig. 1: *Depiction of layers and bridges within the architecture. Each layer contains its own particular set of definitions, while bridges contain the sets of functions for transforming definitions at a given level to adjacent ones. The example here has two source formats: SAM and GFF3.*

Our ontology design requires at least two layers and one bridge: the *shared* L_S and *low-level* ("ground") L_G layers, and the single bridge $B_{L_S L_G}$ containing two sets of functions: one set of functions for the transformation of data defined in the low-level layer *to* the shared layer f_{ij} and another for the inverse functions (mapping shared-layer definitions back down to the low-level) f_{ij}^{-1}. An arbitrary number of additional, intermediate layers and their associated bridges, however, is possible (and necessary in some cases).

Generalized representations for these concepts are given here:

- A function f that lives within a bridge, transforming some definition i at a lower layer into definition j at a higher layer.
$$f_{ij} : D_i \rightarrow D_j \ . \tag{1}$$
- The inverse of the transformation in Equation (1), showing the transformation of some definition j into definition i (where j exists at a higher layer and i exists at a lower layer).
$$f_{ji} : D_j \rightarrow D_i \ . \tag{2}$$
- Each pair of layers only needs one bridge, and both functions in Equations (1) and (2) are included in the same bridge, but in two different sets (one set for transformations from the higher-to-lower layer with the other set dealing with lower-to-higher layer transformations). These sets are denoted in (3) and (4) (with generic layers A and B).
$$B_{L_A L_B} \equiv B_{L_B L_A} \ . \tag{3}$$
$$B_{L_A L_B} = \{f_{ij}\}, \{f_{ji}\} \ . \tag{4}$$

These definitions, however, do not elucidate the full definition of bridges and the functions within; *bridges can be non-surjective and non-injective*. From the source-level, a mapping to the domain level *may* exist, and *if* it exists, the inverse function(s) defined by the bridge can be performed to return to the source-level representation.

That is, given f_{nm}, a function to transform definition n to m it's inverse would be f_{nm}^{-1} if it exists.

Such transformative functions can also be combinations of other, already defined functions. This means that a transformation from one layer to another can be dependent on terms defined at the same level as well as those below it. This allows one to create intermediate definitions and functions at various levels that can be called on at any time to be used in adding further transformation.

4.2 Foundational Definitions

Upper-level ontologies define abstract definitions (which we refer to as *primitives*) for the elements within domain-specific ontologies, establish relationships amongst these abstract definitions, and so on. Perhaps it is easiest to think of the upper ontology — indeed, the entire ontology — from an object-oriented point-of-view, where the top level is as generic as possible, and with the definitions becoming more concrete as one descends the hierarchy.

These primitives have been identified in our design:

- **Component**: The pieces an entity is composed of; what makes up an entity. Components "belong to" an entity.

- **Constraint**: Conditions that allow/do not allow a certain function or process to occur.
- **Entity**: Any physical building block or standalone actor that performs actions or functions. Entities "have one-to-many" functions. Entities "have one-to-many" components.
- **Function**: Actions which are performed by entities. Acts "with an" entity. May be "part of" a process. Can "depend on one-to-many" constraints.
- **Process**: Possibly non-linear sequence of interacting functions. Processes "has one-to-many" functions. Processes may "depend on one-to-many" constraints.
- **Property**: Descriptive parts of a primitive type (any of component, entity, etc). It is possible for everything to "have one-to-many" properties.
- **Rule**: Axioms for determination of validity in interaction or outcome thereof. It is possible for everything to "have one-to-many" rules that govern them.

Using such terms, we can take concepts from a given domain, set its interactions, rules, and so on in order to relate definitions within the ontology to one another.

Upper ontologies contain basic, atomic definitions for concepts in the world. The shared ontology (here, the collections of domain-specific ontologies) contains various implementations of these atomic, abstract concepts. For example, an abstract definition may be an entity. A particular implementation in the financial domain may be a bond, stock, or future. This concrete definition may differ depending on the domain-specific ontology in which it is used, since a "future" in physics is not the same concept as in finance.

If "future" in both physics and finance is defined as an *entity*, it means that both definitions have the same primitive type that possesses rules and context for interaction with one another. The terms have both been given meaning from that base definition, but can also extend their abilities by defining something more specific to their domain-related purpose.

Such relationships similar to "has a", "is a", and others are defined for understanding connections amongst definitions at this level and other levels. For example, consider that an entity is an object that is distinctly different from a function; that functions are performed by entities. Therefore, an entity would "have a" function.

For example, *RNA Polymerase* is a term that biologists know; it is an entity, with rules and functions associated with it. A process might be DNA Replication, with a series of functions in a particular order, and rules for how they must interact. All would be defined at the shared-layer, using the actual biological terms.

To clarify, take the example of RNA Polymerase, while again considering the structure in Fig. 1. At the shared-level, we develop a definition for this entity and define its various relationships as given in Fig. 2. Even from such a small example, one is able to see how these various primitive definitions for the model are linked together.

4.3 The Shared-Level Ontologies

The shared-level ontologies are a series of domain-specific views. At this level, multiple domain ontologies coexist as a union of all the domains that are relevant (and would be loaded into the system). Within each specific domain, basic terms are defined, and the shared-level deals with overlaps, such as identical terms and possible definitions, that may arise as a result of this union.

In our design, the shared-level ontology is also the operational level, meaning that queries are formed from here, potentially involving every domain in the union. This is to allow for the same terms across domains to be used in a single query. However, this also means that a system of

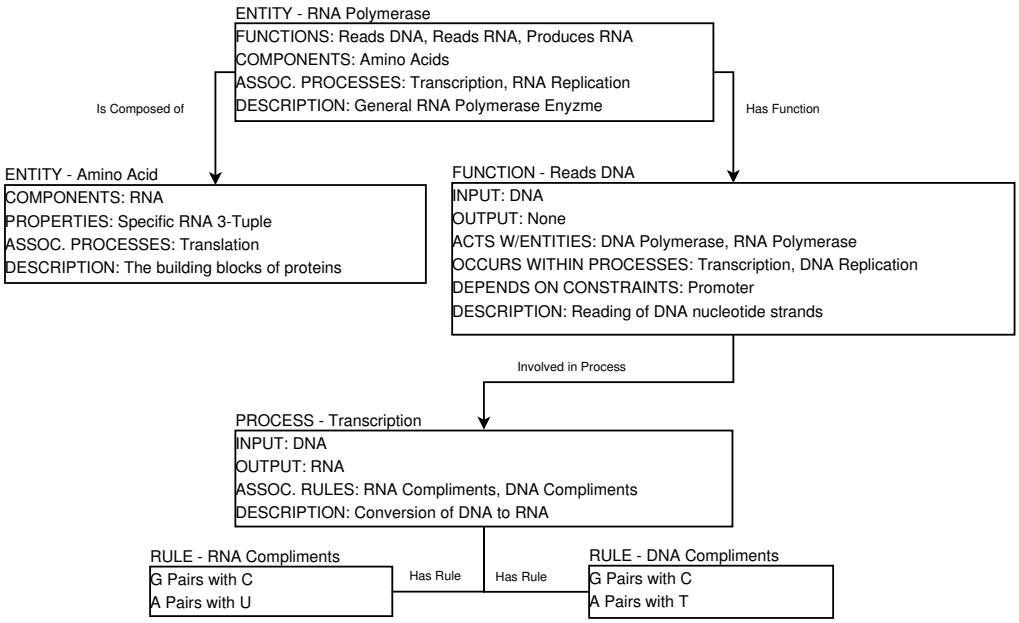


Fig. 2: *Example representation of RNA Polymerase Entity and its relationship to other types of example primitives.*

metadata tagging may be necessary for terms to be differentiated when there are repeated terms across domains [3]. This allows a user to make queries using terms that would overlap on the same domain.

4.4 The Low-Level Source Materials

The low-level sources are the raw sources of data that one may capture from the field.

Here, low-level sources represent the final word in implementation specifics for given definitions. One format may define a term or even *hide* a necessary definition within a larger, format-specific definition, while another format defines it completely differently. However, an individual performing queries should not — indeed, does not — need to concern oneself with such implementation specifics. This is the familiar concept of encapsulation.

Consider multiple source-level mappings to the shared-level have been made. An individual with great familiarity of a given format has taken the raw data, with its specific representations of knowledge, and created a hierarchy of definitions and mappings for us. Once this mapping is built, the most precise definition exists at the source-level, but we can still call on the specific implementation from the top of the hierarchy, hiding the source-specifics transformation details.

Consequently, a user interfaces only with the shared-level queries using shared-level terms such as *Packets* and *Timestamp* instead of source-level terms like *RPACK77* and timestamps as *FIELD3*. Even when the correspondences between terms are simple, they are hidden from the user and it is the system that translates shared-level terms into their format-specific implementations at the low level data source.

4.5 Bridging and Linking

Data transformation from the shared (or *operational*) level of the ontology is taken care of through a series of manually mapped *bridges*. These bridges exist as points between levels or *layers* of the overall ontology; each contains two sets of functions: transforming definitions from a lower layer to a higher layer, and vice-versa.

5 Case Study: Mappings with SAM Files

To demonstrate the usefulness of this ontology design, we consider the example of building some mappings with a low-level source format. Many different tools have been developed in the field of bioinformatics for organizing and representing the data, all with their own specific formats; there may even be versions of formats *in response* to the evolving needs of scientists. One such format is the *Sequence Alignment Mapping* (SAM) [4], [5].

The SAM format was originally built to be generic enough to respond to various different alignment formats that were developed. In so doing, it became very flexible, capable of holding a great deal of information in such a small size. However, in order to accommodate translation from various other formats, additional information had to be developed (by way of tagging and other pieces of the format).

The following outlines a short case of mapping between the SAM file low-level format, and the shared-level ontology for querying.

5.1 Data

Research is currently being conducted at the New Mexico Institute of Mining and Technology that involves the examination of fat transport within *rattus norvegicus* (common, brown rat) on two different and distinct diets. The raw data has been sequenced, and now exists in a SAM format [6]. (For the rest of this example, we will be using data from the rats fed the high-fat diet.)

5.2 The SAM Format

The SAM format is similar to that of a CSV file. That is, there are a series of rows (called *alignment lines*) with different, delimited fields. The two exceptions to this rule are found in optional header section (which we do not fully detail in our study here) and a variable number of elements for the "TAG" field defined for each alignment line. (Note that columns in SAM files are not given a named mapping as is done in CSV. The field/column names are implicitly derived by the SAM format.)

The header section (which is optional) defines information that corresponds to the entirety of reads in the file. Such general information as the SAM format version number, the specifics of the alignment lines at various locations, the order in which data will appear, and so on are defined within this section.

Immediately following the header section (should it exist at all) are the rows of alignment lines (Fig. 3). Each of these lines are broken up into eleven distinct fields — all with their own caveats — and an additional, variable-length twelfth field (which contains various tags).

```
SNPSTER3_0511:3:21:542:508#0/1 0 gi|62750345|ref|NC_005100.2|NC_005100 289632 0 36M *
0 0 CTCCTCCTCCCCTTCTTTATGGGTATTCCCCTCCCC HHHHHHHHHHHHGGHHHHHHHHHHHHHHHHHHHHHGGG MD:Z:36
RG:Z:s_3_sequence.txt.gz IH:i:2 NH:i:2 YR:i:1

SNPSTER3_0511:3:37:1137:468#0/1 0 gi|62750345|ref|NC_005100.2|NC_005100 305783 0 36M *
0 0 CTCTCATGGCTTAAAGTCCTGCCCCGGGACCACCCTG HHHHHHHHHHHHHHGHHGHHHHHHHEHGHHHHHHIHH@ MD:Z:36
RG:Z:s_3_sequence.txt.gz IH:i:4 NH:i:4 YR:i:1

SNPSTER3_0511:3:23:1266:1799#0/1 0 gi|62750345|ref|NC_005100.2|NC_005100 329635 0 36M *
0 0 ATTNAATATAGTTCTCGAACTCCTAGCCAGAGCAAT GGG&GHHHFHFGGHHHHHHIHHIHHFIIHHGIGHIG MD:Z:3X32
RG:Z:s_3_sequence.txt.gz IH:i:2 NH:i:2 YR:i:1

SNPSTER3_0511:3:108:1205:762#0/1 0 gi|62750345|ref|NC_005100.2|NC_005100 368226 0 20M2D16M
* 0 0 GTTTTGGGGATTTAGCTCAGGTAGAGCGCTTGCCTA HGHHHHHHHHHHHHHHHHHHHHGHIHHHHHIFHHHHH MD:Z:38
RG:Z:s_3_sequence.txt.gz IH:i:5 NH:i:5 YR:i:1

SNPSTER3_0511:3:31:1390:225#0/1 16 gi|62750345|ref|NC_005100.2|NC_005100 389280 0 36M *
0 0 GGGGTCAACCTTCAGTCAGCCCCTCGTAGCGAGGGA EGEGEGGEEGGGGGGDEEAFGGCEAGGGGGGGAGGG MD:Z:36
RG:Z:s_3_sequence.txt.gz IH:i:1 NH:i:1

SNPSTER3_0511:3:103:43:1576#0/1 0 gi|62750345|ref|NC_005100.2|NC_005100 394139 0 36M *
0 0 TACGAATGTGATCAATGTGGTAAAGCCTTTGCATCT HHHHHHHHHHHHHHHHHHHHHGHHHHGHHHHHHHHHH MD:Z:36
RG:Z:s_3_sequence.txt.gz IH:i:4 NH:i:4 YR:i:1
```

Fig. 3: *Several alignment lines from the HFD1.sam file. Fields are delimitted by whitespace. The final TAG field itself has several subfields of tags.*

5.3 The Alignment Section and Lines

The bulk of the SAM file is taken up by the alignment section, filled with various alignment lines. Each alignment line represents a particular portion of sequence that was read when sequencing was done on the observational data. These particular sequences are referred to as *reads*. In an alignment line, these *reads* correspond to some type of match that has been aligned to a reference genome.

There are at least eleven distinct fields, delimited by whitespace, in each alignment line. The twelfth field, TAG, is optional and slightly different, since there can be multiple tags within that field or even none at all (whose multiple tags are also delimited by whitespace). An example alignment line's fields (Fig. 4) are detailed below.

- **QNAME:** *SNPSTER3_0511:3:85:824:196#0/1*
- **FLAG:** *0*
- **RNAME:** *gi|62750345|ref|NC_005100.2|NC_005100*
- **POS:** *1350933*
- **MAPQ:** *0*
- **CIGAR:** *36M*
- **RNEXT:** ***
- **PNEXT:** *0*
- **TLEN** *0*
- **SEQ:** *CACACATACGAGGTGTACTTTGTGTGCAGAATGTGT*
- **QUAL:** *E?9BBCC8?C=?;C844555FFF?F:>==6>CCFCE*
- **TAG:** *MD:Z:36 RG:Z:s_3_sequence.txt.gz IH:i:3 NH:i:3 YR:i:1*

These identifiers correspond to fields (1-indexed, i.e., starting with index 1) 1 through 12 respectively. These are the low-level source representations of data in the SAM format.

5.4 Mapping from the Low-Level Source

This is where one defines the mappings for those shared, operational level terms such as **chromosome** and others. We first examine the alignment lines in the SAM file (many lines are given in Fig. 3, while a single sample line to be used in examples is given in Fig. 4) to better understand their structure.

While building the bridges, it is important to keep in mind that we are trying to match a common, shared-level definition. Let us try this with *Chromosome*.

At first glance, it may be difficult to identify chromosome information within the alignment lines. Fortunately, a domain expert is aware of such caveats and can identify that the chromosomal numbering is given within the RNAME field (3rd element in Fig. 4).

```
SNPSTER3_0511:3:85:824:196#0/1   0   gi|62750345|ref|NC_005100.2|NC_005100   1350933   0
36M   *   0   0 CACACATACGAGGTGTACTTTGTGTGCAGAATGTGT    E?9BBCC8?C=?;C844555FFF?F:>==6>CCFCE
MD:Z:36  RG:Z:s_3_sequence.txt.gz IH:i:3  NH:i:3  YR:i:1
```

Fig. 4: *A random alignment line from the HFD1.sam file; broken up by field.*

The last 2 digits within RNAME (**00** in Fig. 4) denote the chromosome number (0-indexed; meaning 00 is chromosome 1) for this alignment line. Slicing off the last two digits of the RNAME field is then defined as the function for this level's bridge for Chromosome.*Number*.

This transformation provides a mapping from the low-level source to the shared-level.

$$f_{RNAME-Chromosome.Number} \in B_{L_G L_S} : RNAME \rightarrow Chromosome.Number\,. \tag{5}$$

If one were to code the function by-hand, one might define it as (in pseudo-code):

```
FUNC RNAME-TO-CHROMNUM(RNAME): return RNAME[-2:]
```

This is one of the functions that the bridge would contain. Recall that bridges hold the functions necessary for transformations. As a high-level definition is transformed as part of a query (or a low-level definition is transformed), the bridges know which function to call for which definition. As a query descends the layers, each definition is transformed accordingly via these functions.

This satisfies a complete mapping for the chromosome number in the SAM file format, version 1.0. This mapping goes into a collection for the SAM format with this specific version, which is kept within a file and pointed to by the system's framework. This helps to maintain a proper organization for the bridges and the formats they are linked to.

We examine alignment uniqueness next, since this is a particularly interesting "free-form" case of the SAM fields. In order to find uniqueness information, we have to consult the twelfth field, TAG. The information from the TAGs given in our example alignment line are:

```
MD:Z:36   RG:Z:s_3_sequence.txt.gz   IH:i:3   NH:i:3   YR:i:1
```

Each tag is given a name, a type, and a value. Anyone familiar with the SAM format can look at these tags and tell right away whether or not an alignment line is unique. Such information is located within tag "IH."

The "IH" flag denotes the number of alignments in the entire SAM file that have the same template name. The type for our "IH" flag is "i" meaning that the value will be an (i)nteger. In this case, it has value 3. If this template (QNAME) were unique, the value for IH would be 1.

To map this value, the low-level definition D_G is set as whatever the file defines. However, it cannot simply suffice to set the definition as TAG, since the TAG field defines many tags. Instead, one must set a more appropriate definition on a per-tag basis.

To do this, we create an intermediate level up from the low-level source. (This is to illustrate that an arbitrary amount of intermediate levels can be built to suit the needs of the low-level source, combining definitions from various levels if necessary.) The intermediate level splits up the TAG field into its component parts (the different tags).

– **Tags:** *IH*, *MD*, *RG*, *NH*, *YR*

From the low-level source to this intermediate layer, the functions defined in the bridge are to separate the Tag field up by cutting it into pieces, and setting those pieces (the individual tags), into a new set of definitions:

This transformation provides a mapping from the low-level source definition to the intermediate definition of "Tags."

$$f_{TAG-Tags} \in B_{L_G L_1} : \quad TAG \rightarrow Tags.IH,\ Tags.MD,\ Tags.RG,\ Tags.NH,\ Tags.YR\ . \tag{6}$$

The internals of this function might look like the following (in pseudo-code) :

```
FUNC TAG-EXPANSION(TAG):
       local dictionary tag-map
       for Tag in TAG:
              tag-map[Tag[:2]] = Tag[Tag.findLast(':'):]
       return tag-map
```

To go the other direction, the inverse function suffices:

$$f^{-1}_{TAG-Tags} \in B_{L_1 L_G} : Tags.IH,\ Tags.MD,\ Tags.RG,\ Tags.NH,\ Tags.YR \rightarrow TAG\ . \tag{7}$$

Each tag in the example line (IH, MD, RG, NH, and YR) is mapped to the "Tags" definition in this intermediate layer. This provides a quick way for one to reference and use it in other intermediate layers as needs be, having singled-out the information from the larger sample line itself. Now the tags are split up, and from each of them, one can answer questions such as "uniqueness".

Since the IH tag denotes uniqueness amongst all alignments in a SAM file, we can instead set the IH tag from our intermediate-level "Tags" definition to be "unique", mapping the value from the intermediate layer to the shared layer. Only a boolean value of True or False is necessary. Also note that this particular tag can be used to set the Total Matches for the alignment, but in that case, the precise integer value becomes important (as opposed to being 1 or otherwise).

$$f_{Tags.IH-Alignment.Unique} \in B_{L_1 L_2} : Tags.IH \rightarrow Alignment.Unique\ . \tag{8}$$

The function for this transformation might be defined (in pseudo-code) as:

```
FUNC IH-IS-UNIQUE(Tags.IH):
       if Tags.IH == 1: return True
       else: return False
```

Note, however, that it may not be possible to reconstruct the IH tag merely from the "Unique" definition. Instead, the only way to get back the full, original value of this IH tag is from the "Total Alignment Matches" portion of the Alignment Definition. That is, there may not exist a suitable inverse function for uniqueness in the case where uniqueness is false, and instead, the function to go back to the full IH tag must be:

$$f^{-1}_{Tags.IH-Alignment.Unique} \in B_{L_2L_1} : Alignment.TotalAlignmentMatches \rightarrow Tags.IH \quad (9)$$

It should be clear that other mappings for the format — and indeed all different formats — can be constructed similarly. Some may be more simple, while others are far more complex.

5.5 Some Example Queries

Now that the low-level source information has bridges set up to allow for definition transformation, we can make queries on the data. The overall query is broken up into sub-queries, answering questions in parallel on the source files used with the aid of the bridges, transforming the SAM format into something more recognizable by all domain experts familiar with bioinformatics, but not necessarily the SAM format itself. Some examples are given below.[1]

– Which sequence is repeated the most in the HFD1.sam data, and does it represent a new gene?

```
USES FORMAT SAM, GFF3

(SELECT MOST REPEATED Sequence FROM FILE HFD1.sam) AS RepeatedSeq
(SELECT Gene FROM FILE RATGENOMEv3_4.sam) AS KnownGenes

SELECT Gene.Expression
FROM FILE HFD1.sam
WHERE NOT KnownGenes
AND WHERE RepeatedSeq
```

This will return all the possible pileups from the most repeated sequence in the HFD1.sam data that sufficiently overlap to denote a "new" gene. By forming sub-queries to count the pileup start locations, examining their overlaps, and finding runs of sufficiently piled up sequences, the system can have certain definitions for what constitutes — and then matches to — a possible gene region such as this.

– Which non-coding regions of the genome, near the coding region for the leptin gene, are expressive?

```
USES FORMAT SAM, GFF3

(SELECT Chromosome.Number, Sequence.StartBase.Number, Sequence.EndBase.Number
FROM FILE RATGENOMEv3_4.gff
```

[1] Queries are presented as small snippets of SQL-like code, but we want to stress that biologists will not have to learn SQL to make use of the system. Their high-level queries (in a language that will involve biological concepts — still a work in progress) will be translated into similar code.

```
WHERE Gene.Name=lep) AS LeptinGene

SELECT Gene.Expression
FROM FILE HFD1.sam
WHERE Chromosome.Number=LeptinGene.Chromosome.Number
AND Sequence.Start NEAR LeptinGene.Base.Start
```

Here, one is examining non-coding regions of rats that have been given a high-fat diet, exploring any up-regulated portions of the genome around leptin, a protein known to be involved in fat transport. This is to aid a researcher in a hunch regarding non-protein-coding regions of the genome being relevant in dietary aberrations beyond the simple analysis of coding regions alone.

Combining a sample with a particular reference is common in bioinformatics and is necessary for asking certain types of questions (here, the reference genomic information comes from a file in the GFF3 format [7]). With this model, a biologist can — in a single query — take two heterogeneous source formats, combine them together, and make complex queries that result in meaningful answers for questions. Questions that may prove useful in understanding diseases. Practically, any question whose information exists within the source formats is answerable. And this is only an examination of that particular field. This model can be applied to any field with source formats of any type and number.

Such an ability to make queries as this across many different, possibly overlapping source formats at once is impossible with current tools available. There is no single language or system by which such queries can be made with the richness of object-oriented design and the flexibility of any source format the mind can dream up, alongside coexisting, multiple domain-level ontologies, each with their own definitions.

6 Analysis

The example of SAM and GFF files is a fairly complex one, and while this paper is not extensive in its exhibition of how all transformations of that format are to be performed, they can be built as prescribed by this proposed design methodology

This proposed design still "suffers" from the same concern as existed in the KRAFT system, mainly that arbitrary — even wrong — mappings can be defined. This is a fundamental risk of crowd-sourced solutions. Since we imagine bridge definitions are just files that can be sent to others, we believe that a type of self-checking will go on amongst the community of researchers that make use of this design. More accurate, perhaps even faster, bridging definitions for the different formats will be accepted as the best, used most often, and signed off on, while those mappings that map data incorrectly will wither and die.

Of course, the question of how to ensure that bridge definitions do not produce poor results before using them is another task that we are currently looking into. The ability to verify that a mapping produces good, accurate results is something that we would certainly like to have. This would both cut down on conscious attempts to distort results as well as notify users about honest mistakes in transformation of definitions.

Facilitating the creation of bridging will require a great deal of work, since not everyone knows how to make a function to transform data into various definitions for different link levels. It will, however, be critical in producing something that experts want to use. For those that are comfortable with building functions, the ability to just program them in should also be available.

7 Conclusion

Ontologies offer a way to organize and transfer knowledge, but they suffer when overlap across domains occur or formats differ. In such cases as arise in merging/linking ontologies, some form of automation would be helpful in resolving issues of clarity and yield a single, well-formed and unambiguous ontology. Unfortunately, there are many problems that arise in pursuing such an automated approach.

Instead, we have presented a solution that relies on using the work of experts who build an ontology of their field of expertise, e.g., biology, with that of experts who can build transformations on data representations on a per-format basis. The result benefits practitioners of the field, who have working knowledge of terms within their fields, but neither possess specialized knowledge of a given data format, nor are technically trained to exploit it.

By defining shared-level terms for a given domain and building bridges from low-level sources up to those domains, queries can be made on all variety of source formats. These bridges can then be modified and shared by the community of researchers. Further, it is possible to query across an ensemble of data in different formats allowing comparison, contrast, and union of the diverse information content that no format-specific software system would enable.

8 Acknowledgements

We would like to thank Dr. Rebecca Reiss for her assistance in helping us formulate queries and realizing how those queries could be understood in a piece-wise manner from SAM files. We also thank anonymous reviewers whose comments have helped improve this paper.

References

1. Bouquet, Paolo, et al. *C-OWL: Contextualizing Ontologies*. The Semantic Web-ISWC 2003. Springer Berlin Heidelberg, 2003. pp. 164-179.
2. Preece et al. *The kraft architecture for knowledge fusion and transformation*. In Proceedings of the 19th SGES International Conference on Knowledge-Based Systems and Applied Artificial Intelligence (ES99). Springer, 1999.
3. Xu, Zhichen, et al. *Towards the Semantic Web: Collaborative Tag Suggestions*. Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland. 2006.
4. Li, Heng, et al. *The Sequence Alignment/Map Format and SAMtools*. Bioinformatics 25.16 (2009): 2078-2079.
5. *Sequence Alignment/Map Format Specification*. SAMTools. The SAM/BAM Format Specification Working Group, 29 May 2013. Web. 17 July 2013. http://samtools.sourceforge.net/SAMv1.pdf
6. *Fat-Rat SAM Data*. NMT Biology Bioinformatics Portal. New Mexico Institute of Mining and Technology. Web. 03 Aug. 2013. http://bioinformatics.nmt.edu/
7. *The Sequence Ontology - Resources - GFF3*. The Sequence Ontology. Web. 18 Sept. 2013, http://www.sequenceontology.org/gff3.shtml
8. Mortazavi, Ali, et al. *Mapping and Quantifying Mammalian Transcriptomes by RNA-Seq*. Nature methods 5.7 (2008): 621-628.

Design and Implementation of a Rule-based Recommender Application Framework for the Semantic Web Data

Thanyalak Rattanasawad[1], Marut Buranarach[2], Ye Myat Thein[2], Thepchai Supnithi[2], and Kanda Runapongsa Saikaew[1]

[1] Department of Computer Engineering, Faculty of Engineering, Khon Kaen University, Khon Kaen, Thailand
thanyalak.rattanasawad@gmail.com, krunapon@kku.ac.th
[2] Language and Semantic Technology Laboratory
National Electronics and Computer Technology Center (NECTEC), Pathumthani, Thailand
{marut.bur, thepchai.sup}@nectec.or.th, yemyatthein@gmail.com

Abstract. This paper describes the design and implementation of a recommender application framework which aims to simplify development of ontology-based recommender applications over the Semantic Web data. Recommender system is a type of system that generates meaningful recommendations to support user's decision. Development of recommender system for the Semantic Web data typically requires ontology, rules and rule-based inference engine to be applied over the RDF data. To facilitate development of recommender applications, our application framework introduces a recommendation template that is a specific form of rule language that provides high-level abstraction in generating recommendations. Recommendation rules can be created based on the template using recommendation editor to hide complexity of rule language syntax. The framework proposes two implementation approaches for generating recommendation results based on the recommendation rules: rule-based reasoner and SPARQL-based implementation. The preliminary evaluation results over a data set in public health domain showed more efficient execution time using the SPARQL-based implementation approach. The results suggested that, by limiting expressiveness of recommendation rules, a specialized form of recommendation processor can be developed for more efficient system performance.

Keywords: Semantic Web application framework, ontology application framework, rule-based recommender system framework

1 Introduction

Although creation of the Semantic Web data rapidly grows, e.g. the Linked data initiatives [1], applications and uses of the data are relatively limited. This is partly due to high learning curve and efforts demanded in building Semantic Web and ontology-based applications. Recommender system is a type of system that generates meaningful recommendations to support user's decision. Recommender system development for the Semantic Web data typically requires ontology, rules and rule-based inference engine to be applied over the RDF data. To facilitate development of

recommender applications, development tools should allow application developers to focus more on domain problems and knowledge rather than implementation details.

Our work proposes an application framework aimed to simplify development of recommendation systems that apply the knowledge-based analysis technique [2] over the Semantic Web data. A typical application of the technique includes modeling a user's profile information and resource properties based on ontology. Then, a recommendation of list of resources recommended for the user can be generated based on recommendation rules externally defined by domain expert users. Our framework introduces a recommendation template that is a specific form of rule language that provides high-level abstraction in generating such a recommendation. Recommendation rules can be created based on the template using recommendation editor to hide complexity of rule language syntax. Subsequently, the template can be processed by a recommendation processor to produce recommendation results.

This paper describes the design and implementation of the recommender application framework developed as a part of the Ontology Application Management (OAM) Framework [3], which is an application framework aimed to simplify development of ontology-based applications over the Semantic Web data. Design of the framework focuses on three main principles: abstraction, extensibility, and interoperability. It proposes a generalized recommendation template for ontology-based recommender applications. We also present two implementation approaches of recommendation processor to execute the recommendation rules and evaluate the performance over a data set in public health domain. The evaluation results showed that, by limiting expressiveness of recommendation rules, a specialized form of recommendation processor can be developed for more efficient system performance.

2 OAM Framework

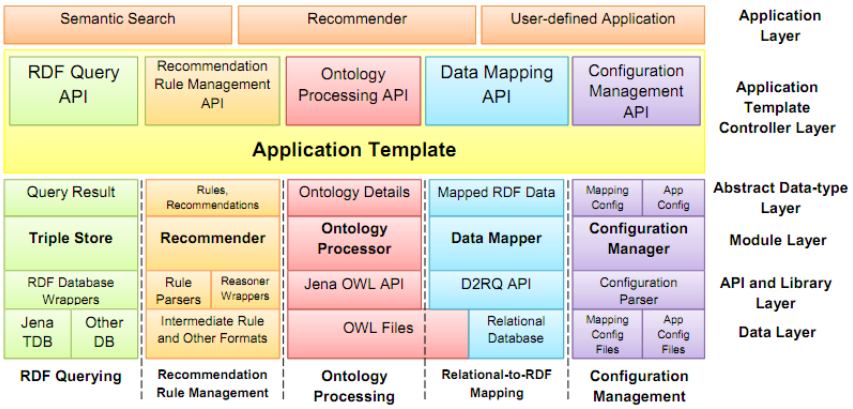


Fig. 1 Architecture of OAM framework

Ontology-based Application Management (OAM) framework [3] is a framework to simplify development of semantic web applications focusing on semantic search and

recommender systems. It enables the user to create customized applications based on provided application templates without high learning curve. The framework was developed in Java programming language on top of existing tools: Jena framework[1] for processing RDF-based data and D2RQ[2] for mapping data from relational databases to RDF database. The layered architecture of the OAM framework is shown in Fig. 1. The framework provides Application Template, the base template for creating semantic web applications. It is a collaboration of application modules for processing each type of data: Triple Store for storing and querying RDF data, Recommender for creating recommendations and managing recommendation rules, Ontology Processor for processing details of ontology, Data Mapper for mapping data from relational database to RDF database, and Configuration Manager for managing configurations of the data mapping and other applications. This paper focuses on the Recommender module of the OAM framework.

3 Recommendation Template

The recommender application framework focuses on simplifying creation and management of recommendation rules. It provides a recommendation rule management tool that supports two processes: create recommendation and link recommendation. Creating recommendation will create an instance of a recommendation container class, e.g. “Promotion” where the user can define conditions of class instances to be the recommendations, e.g. “Product”. Linking recommendation allows the user to define conditions of class instances to recommendation receivers, e.g. “Customer”. The framework facilitates the user to create such business logics using a form-based user interface and hides complexity of the rule syntax to be processed by reasoning engine.

In this framework, recommendation rule can be created based on a recommendation template, whose structure can be summarized as follows.

- **Recommendation rule:** A recommendation rule consists of rule name, two condition sets for matching each part of recommendation rules: recommendations and recommendation receivers, and a property of the receiver class for receiving the recommendations.
- **Condition set:** A condition set consists of condition set name, matching class, a class of the individuals to be matched, and conditions.
- **Condition:** A condition for matching individuals. It consists of a property chain, an operator, and a value object.
- **Property Chain:** A property chain is an ordered series of one or more object-type properties. Each property links to an individual of the object of a triple recursively like joining many tables in relational database.
- **Operator:** An operator for matching or comparing between the condition’s value object and a triple's object. The operators supported are mathematics comparison operator (=, >, <, >=, <=), string operator (=, contains). They can be used in the forms: <property> <operator> <value> and <property> <oper-

[1] http://jena.apache.org/
[2] http://d2rq.org/

ator> <property>. RDF comparison operator (type) can be used in the form <property> <operator> <class>.

- **Object:** Object of a condition. It is in three types: literal value, URI, and property chain node value. A literal value is an RDF literal with a specified data-type. A property chain node is a node that refers to value of another property chain, allowing it to be operated with the literal value of the current condition.

Recommendation rules are stored in an intermediate format using JSON and can also be exported to a number of interchange formats.

To exemplify the format of the rule template, we make an example of a recommendation "Recommend the products which have discount rate more than 10 percent, and have price more than 10 dollars but less than 15 dollars, to the customers which have bought products from the store more than 20 times".

Fig. 2 illustrates elements of the recommendation rule and linked condition sets in JSON format. Condition sets are illustrated in Fig. 3. The user can use a provided rule editor to create recommendation rules based on the template that hides the complexity of the created rule syntax.

```
"ClearancePromotion": {
  recOfRule: "DiscountedProducts",
  recToRule: "FrequentlyVisitingCustomers",
  recProperty: "http://ex.org/suggestedProducts" }
```

Fig. 2 Elements of recommendation rule and linked condition sets

```
"DiscountedProducts": {
matchingClass: "http://ex.org/Product", conditions: [
{ propertyChain: ["http://ex.org/discountRate"], operator: ">", object: ["float","10.00"]},
{ propertyChain: ["http://ex.org/price"], operator: ">", object: ["float", "10.00"]},
{ propertyChain: ["http://ex.org/price"], operator: "<", object: ["float", "15.00"]]}

"FrequentlyVistingCustomers": {
matchingClass: " http://ex.org/Customer",
conditions: [{ propertyChain: ["http://ex.org/boughtRecord", “http://ex.org/boughtTimes”],
operator: ">", object: ["int", "20"]]}
```

Fig. 3 Matching Condition Sets

4 Design and Implementation of the Recommender Application Framework

The framework is designed based on three main principles: abstraction, extensibility, and interoperability. Abstraction is achieved by providing higher level of constructs than those provided by the RDF data model for building Semantic Web-based recommender applications. Extensibility is achieved by designing each module that is independent of the underlying implemented systems, i.e. inference engines and data-

bases. Interoperability is achieved by allowing exports to the standard rule formats to enable interchanging and integration with other rule-based systems.

4.1 System Architecture

Fig. 4 illustrates the system architecture of the recommender module. The follows describe details of the related modules of recommendation data management.

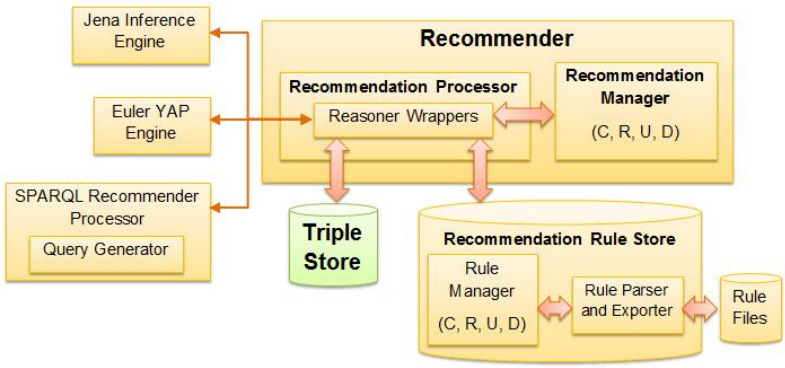


Fig. 4 Architecture of the OAM recommender module

- **Triple Store:** The module which provides database-independent interfaces of common functions for querying RDF database. This module allows the user to perform arbitrary queries and also provides query generator with some query templates.
- **Recommender:** The module for recommendation processing which contains sub-modules for managing and processing recommendation data: recommendation rules and recommendation instances.
- **Recommendation Rule Manager:** The module for recommendation rule management, which enables the user to create, view, edit, and delete recommendation rules, import other rules in format of the rule template, and export rules to interchange formats, e.g., RDF, JSON, and RIF [4], and a format of supported reasoner, such as Jena rule [5] and Notation3 [6].
- **Recommendation Processor:** The sub-module which provides common interfaces and encapsulates algorithms of recommendation creation, and recommendation results generation.

4.2 Implementations of Recommendation Processor

Recommendations are created and linked with the related resources by passing recommendation rules and data to the recommendation processor. The implementations are categorized into two approaches: rule-based inference engine approach, and

SPARQL-based approach, which are described as follows.

- **Rule-based inference engine approach:** We created implementations using two different rule-based reasoners: Jena inference engine [5] and Euler YAP Engine [7]. The recommendation processor generated recommendation rules in form of if-then rule in the supported format of each reasoner. Recommendation results are created by means of rule-based reasoning. Fig. 5 shows an example of the recommendation rule created in Jena rule syntax.

```
@prefix : <http://ex.org/>. @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
(?recOf rdf:type :Product) (?recOf :discountRate ?v1) greaterThan(?v1, 10.00)
(?recOf :price ?v2) greaterThan(?v2, 10.00) lessThan(?v2, 15.00) ->
(ProductRec_1 :hasInstances ?recOf)

(?recTo rdf:type :Customer) (?recOf :boughtRecord ?v3)
(?v3 :boughtTimes ?v4) greaterThan(?v4, 20) ->
(?recTo :hasSuggestedProducts :ProductRec_1)
```

Fig. 5 Jena rule syntax of the recommendation rule

- **SPARQL-based approach:** This approach adopts a SPARQL engine as the recommendation processor [8][9], and generates recommendation rules by means of SPARQL queries and updates. Fig. 6 shows an example of SPARQL update syntax for generating recommendations and results.

```
PREFIX : <http://ex.org/> PREFIX rdf: < http://www.w3.org/1999/02/22-rdf-syntax-ns#>
INSERT {
 ?recTo :hasSuggestedProducts :ProductRec_1.
 ?ProductRec_1 rdf:type :Product_Rec.
 ?ProductRec_1 :hasInstances ?recOf.
 ?ProductRec_1 :hasRecName “ProductSetA”.
 ?ProductRec_1 :hasRecId “1”. }
WHERE {
 ?recOf rdf:type :Product.      ?recOf :hasDiscountRate ?v1.      ?recOf :price ?v2.
 ?recTo rdf:type :Customer.   ?recTo :boughtRecord ?v3.         ?v3 :boughtTimes ?v4.
 FILTER(?v1 > 10.00)  FILTER(?v2 > 10.00)  FILTER(?v2 < 15.00)  FILTER(?v4 > 20)}
```

Fig. 6 SPARQL update syntax of the recommendation rule

5. Performance Evaluation

The performance of the framework implementations were evaluated in supporting the Thalassemia clinical decision support system (CDSS) research [10]. The total of 34 rules was created based on the template to enable diagnosis of 17 Thalassemia disease and carrier types. The test data were extracted from the Siriraj hospital database entries of Thai patients who were examined for Thalassemia. Each patient data consists of personal health data, lab and DNA test results. Five data sets containing different sizes of patient data were prepared: 100, 500, 1000, 1500 and 2000 patients. Data processing time was measured from when the RDF/XML file was read and processed

by the recommendation processor and stored in the RDF data storage. Wrappers for three different implementations of the recommendation processor, Jena inference engine, Euler YAP Engine (EYE) and the SPARQL-based implementation were deployed and compared in terms of system response time. The tests were performed on a computer with the following specifications: Intel core i5 – 430M CPU with 8 GB DDR3 memories, using JDK version 1.6 with minimum memory size 1 GB and maximum memory size 4GB.

The detailed comparison of total processing time for different implementations is shown in Fig. 7. The total response time for data processing and analysis for 2,000 patient data, which consist of 0.12 million triples and 1.14 million triples before and after recommendation respectively, is approximately under 14 minutes using Jena's, under ten minutes using EYE and under two minutes using the SPARQL-based implementation. Based on the results, the Thalassemia CDSS can achieve a reasonable system performance by using the SPARQL-based implementation.

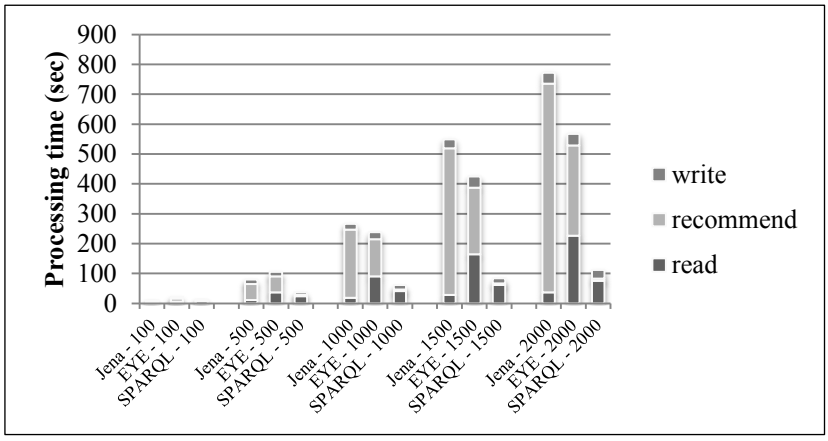


Fig. 7 Results of data processing time in Thalassemia CDSS using different implementations

The good performance of the SPARQL-based implementation is largely achieved by limiting rule expressiveness. Currently, it can only support rules that generate results which do not fire another rule. By limiting rule expressiveness of the recommendation template, the SPARQL-based implementation is more efficient than rule-based inference engine implementation since less reasoning operations are performed.

6. Conclusion

In this paper, we presented design and implementation of a rule-based recommender application framework for the Semantic Web data. The framework is different from existing Semantic Web application framework in that it simplifies the development of recommender applications by providing a generalized recommendation template that can be used in ontology-based recommender applications. The template can be managed and processed by specialized recommendation editor and processor. The design of the framework also focuses on extensibility and interoperability to allow it to be

independent of underlying implemented systems. An evaluation study was conducted by comparing performance of different implementations of the recommendation processor using a data set in public health domain.

Our planned future work includes improving design of the recommendation template to support more expressiveness and functionality. We also plan to improve the SPARQL-based implementation to support more expressive recommendation rules.

Acknowledgement

The financial support from Young Scientist and Technologist Programme, NSTDA (YSTP: SP-56-NT03) is gratefully acknowledged.

References

1. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. Int. J. Semant. Web Inf. Syst. 5, 1–22 (2009).

2. Burke, R.: Knowledge-Based Recommender Systems. Encycl. Libr. Inf. Syst. 69, (2000).

3. Buranarach, M., Thein, Y.M., Supnithi, T.: A Community-Driven Approach to Development of an Ontology-Based Application Management Framework. Joint International Semantic Technology Conference (2012).

4. Kifer, M., Boley, H.: RIF Overview (Second Edition), http://www.w3.org/TR/rif-overview/.

5. The Apache Software Foundation: Apache Jena - Reasoners and rule engines Jena inference support, http://jena.apache.org/documentation/inference.

6. Berners-Lee, T., Connolly, D.: Notation3 (N3): A readable RDF syntax, http://www.w3.org/TeamSubmission/n3/.

7. Roo, J. De: Euler Yet Another Proof Engine, http://eulersharp.sourceforge.net/.

8. Hawke, S., Herman, I., Parsia, B., Seaborne, A.: SPARQL 1.1 Entailment Regimes, http://www.w3.org/TR/sparql11-entailment/.

9. Knublauch, H., Hendler, J.A., Idehen, K.: SPIN - Overview and Motivation, http://www.w3.org/Submission/spin-overview/.

10. Assawamakin, A., Chalortham, N., Ruangrajitpakorn, T., Limwongse, C., Supnithi, T., Tongsima, S.: A development of knowledge representation for thalassemia prevention and control program. Natural Language Processing and Knowledge Engineering (NLP-KE), 2011 7th International Conference on. pp. 190–193 (2011).

A Basic Consideration on Ontology Refine Method using Similarity among *Is-a* Hierarchies

Takeshi Masuda

Kouji Kozaki

Graduate School of Engineering Osaka University#1

The Institute of Scientific and Industrial Research (ISIR)，Osaka University#2

Abstract. Quality of ontology is important because it is connected directly with the performance of an application system using the ontology. However ontology refinement to improve its quality needs knowledge and experiments in ontology development. Therefore, ontology refinement task is too difficult especially for beginners in ontology building. In order to solve this problem this article proposes an ontology refinement support system based on similarity among is-a hierarchies and an evaluation of it. The system can support content refinements for ontologies.

1 Introduction

Nowadays, ontologies are constructed in various fields such as medical information, mechanical design, and etc. These ontologies are used as knowledge bases and knowledge models for application systems. Quality of ontologies is an important factor for the application system which uses them because it directly affects to its performance. Therefore a construction of better quality ontology is a considerable issue.

However, in order to build well organized ontologies we need knowledge and experience about ontology construction and also expertise in their target domain. For this reason, it is not easy for beginners to construct good ontologies. Because of these backgrounds, ontology construction and refinement support system are expected.

There are some approaches to support ontology building. One is an approach to give some guidelines for the whole process of ontology building [1]. Some researchers propose semi-automatic method to support ontology building process. For example, proposes a semi-automatic method to construct a large scale ontology using semi- structured information such as Wikipedia [2]. On the other hand, ontology refinement is also one of important process to improve quality of was ontologies. This paper proposes an ontology refinement method and an ontology refinement support system based on it.

The rest of this paper is organized as follows. Section 2 outlines existing approaches on supporting method for ontology refinement. In section 3, we propose ontology refinement method based on similarity among *is-a* hierarchies. Section 4 discusses evaluation of the proposed method. Finally, Section 5 concludes this paper.

2 Ontology Refinement Supporting Method

There are two kinds of supporting methods for ontology refinements to improve quality of ontologies. One is a method to detect and correct formal errors in ontologies. And the other is a method to refine contents of ontologies. The former includes many methods for consistency checking and debugging function for ontologies [3]. Poveda et al. collect common errors found in OWL ontologies, which are called ontology pitfalls, and develop a system to detect these pitfalls and correct some of them [4]. On the other hand, when it comes to the latter, there are a few methods to refine contents of ontologies. Although some methods such as voting system [5] and visualization to support understanding of ontologies are proposed, they require human intervention in order to judge right and wrong of contents of ontologies. That is, in these approaches, knowledge of their target domains and experiences on ontology building are necessary to evaluate their contents. Therefore they are not enough to support content refinement of ontologies. To overcome this problem, this paper propose an ontology contents refinement support system which could be help for users who do not have enough experiences on ontology building and domain knowledge.

3 Ontology Refinement Based on Similarity among *Is-a* Hierarchies

3.1 Similarity among *Is-a* Hierarchies

There is a guideline for building well-organized ontologies that "Each subclass of a super class is distinguished by the values of exactly one attribute of the super class. [6]" When ontologies are built under the guideline, we can find many *is-a* hierarchies whose conceptual structures are similar to other *is-a* hierarchies in the same ontologies. For example, in Fig.1, "vehicle" is classified into two lower concepts such as "ground vehicle" and "aircraft" according to their "movement space". These characteristics are represented by referring to other concepts as a class restriction for "movement space". In "vehicle" class, the class constraint of "movement space" slot is "natural space". And it is specialized in slots of its lower concepts "ground vehicle" and "aircraft" to "ground" and "air" respectively. As the result, *is-a* hierarchies of "vehicle", "movement space" slot and "natural space" have partly similar structures.

When a concept is specialized into its lower concepts, some of its slots are also specialized according to *is-a* hierarchies of concepts which are referred as their class constraints. In the same way, we can find similar structures of *is-a* hierarchies of "bicycle", "handle" slot and "handle" in Fig.1.

In this paper, we call "an *is-a* hierarchy of slots whose class constraints are specialized" *"slot hierarchy"*, and "an *is-a* hierarchy of concepts which are referred as class constraints of these slots" *"referred concept hierarchy"*. "*Slot hierarchy*" and its "*referred concepts hierarchy*" have a similar structure. In addition, because a *"slot hierarchy"* is built according to a basic concept hierarchy[1] in which these slots are defined, the basic concept hierarchy also has similar structure with the slot hierarchy. Therefore, we can find partly similar structure among *"basic concept hierarchy"*, *"slot hierarchy"* and *"referred concept hierarchy"* each other (see Fig.1).

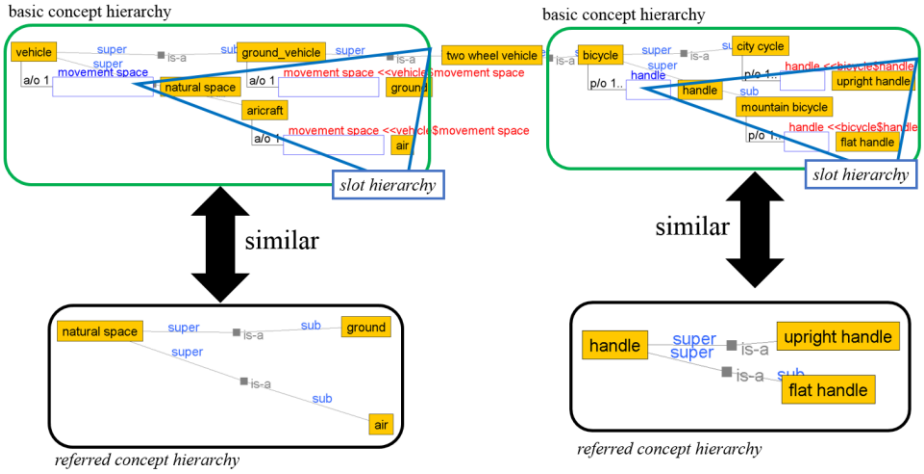


Fig. 1. Similarity among *Is-a* Hierarchies

3.2 Ontology Refinement Using Similarity among *Is-a* Hierarchies

In this paper, we consider ontology refinement system using similarity among three kinds of *is-a* hierarchies introduced in the previous section. There are two directions to use the similarity among them for the ontology refinement.

(1) The first direction is to propose refinement of a basic concept hierarchy according to structure of its referred concept hierarchy. The refinement system compares "referred concept hierarchy" to "slot hierarchy", then it proposes some modification of the "slot hierarchy" to have similar structure with the referred concept hierarchy. As the result, the user also refines "basic concepts hierarchy" according to the proposal.

(2) The other direction is to propose refinement of a referred concept hierarchy according to structure of a basic concept hierarchy which refers to it. The refinement system compare "slot hierarchy" to "basic concept hierarchy", then it make proposal to modify

[1] It corresponds to a normal class hierarchy. We call it basic concept hierarchy to distinguish it with slot hierarchy and referred concept hierarchy.

so that they have a same structure. The proposal involves addition of some concepts to "referred concepts hierarchy". According to the proposal, the user decides what concept should be added and modifies the ontologies.

In this paper, we call the former approaches "Forward refinement" and the later ones "Backward refinement".

3.3 Forward Refinement

Outline of Forward Refinement

For forward refinement, at first the system compare a referred concept hierarchy to the slot hierarchy which refers to it. For comparing them, the system focuses on concepts which are not refereed by any concepts in the slot hierarchy. When the referred concept hierarchy contains such un-referred concepts, it means that these two hierarchies are not similar structure. Therefore, such un-referred concepts are considered as *support target concepts* for forward refinement.

Here, we consider an example for proposals for modifications of an ontology based on forward refinement. In *is-a* hierarchy shown in Fig.2, "gear change" is a support target concept for forward refinement because it is not referred by any other concepts. Then the system consider its upper and lower concepts, "driving operation" and "shift to a lower gear". These two concept are referred by "driving action" and "slowdown" as class constraints of the same slot "part of action". Based on this observation, the refinement system can propose *to add "part of action" slot on "speed control"(fig.2 ①), which is a middle concept between "driving action" and "slowdown", and refer "gear change" (the support target concept) as its class constraint(fig.2 ②).* By following this proposal, "basic concepts hierarchy", "slot hierarchy" and "referred concepts hierarchy" become a similar structure.

In this way, the system makes proposals for modification of an ontology so that these three kinds of hierarchies have similar structures by focusing on un-referred concepts and their upper and lower concepts. There are patterns how the system propose modification of an ontology based on forward refinement. In order to detect which patterns are applicable by the system, we have to make them explicit as conditions which computer are able to distinguish. Using these condition, the system can find out proposals for ontology refinement automatically.

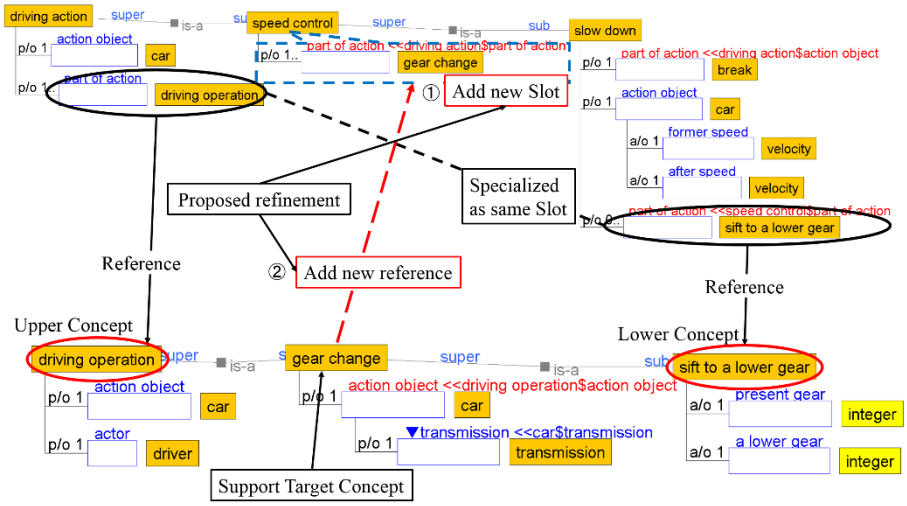


Fig. 2. Refinement Suggestion Example

Patterns for Forward Refinement

Patterns for forward refinement are classified into the following three types by the structure of focusing referred concept hierarchy.

(i) Focusing on the support target concept and both of its upper and lower concepts as the referred concept hierarchy.
(ii) Focusing on the support target concept and its upper concept as the referred concept hierarchy[2].
(iii) Focusing on the support target concept and one of its lower concept as the referred concept hierarchy[3].

On the other hand, which kinds of modifications are proposed for ontology refinement depends on the basic concept hierarchy to which the proposed modification are apply. Therefore patterns for forward refinement are classified into two more types as followings,

A) There is a concept which can be added a new slot referring to the support target concept as its class constraint in the basic concept hierarchy.
B) There is no concept which can be added a new slot referring to the support target concept as its class constraint in the basic concept hierarchy.

---

[2] We does not care whether the support target concept has lower concept or not. We assume that the ontology does not have multiple inheritances.

[3] As the support target concept which has multiple lower concepts, this system propose each suggestions for each lower concepts. And we distinguish these proposals as different one. When we discuss the result in section 4, we also count these proposals as different suggestions.

In the case of type B), the system can propose two kinds of modification methods; (1) to add new slot on an existing concept and refer target concept as its class constraint and (2) to add new concept which has a slot referring to the support target concept as its class constraint. The users can choose the method (2) when they consider the method (1) is not appropriate, that is, the existing concept could not have the proposed slot. In the case of type B, the system can propose only the method (2) because there is no concept which can be added the new slot.
Based on the above classifications, forward refinement is classified into six patterns by combinations of the above (i) - (iii) and A) - B).

3.4 Backward Refinement

Outlines of Backward Refinement

For backward refinement, the system firstly compares a "slot hierarchy" to its "basic concept hierarchy". Then, the system proposes to add a new concept to its "referred concept hierarchy" so that these three hierarchies have similar structure. For example, a "basic concepts hierarchy" shown in Fig.3 consists of "car race", "formula car race", "F1 race", "F3 race" and "Indy 500 race". At the same time, their "machine type" slots are specialized according to its "referred concept hierarchy" shown in Fig.4 and compose its "slot hierarchy". However in this example, these two hierarchies are not similar. In this case, the system can propose *to add a new middle concept which is referred to as a class constraint concept of the "machine type" slot at "formula car race" concept in the referred concept hierarchy* in Fig. 4 so that it has the similar structure with its slot hierarchy.

Backward refinement has a significant different from forward refinement. It is that the system always proposes to add new concept on the "referred concept hierarchy" for backward refinement.

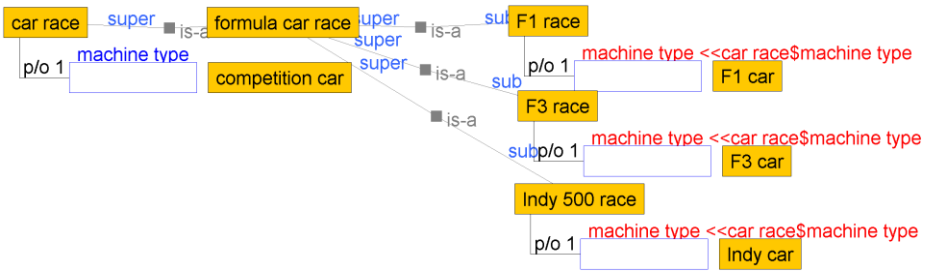


Fig. 3. Base Concepts Hierarchy and Slots Hierarchy about "car race"

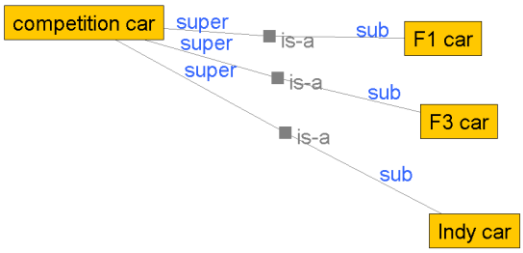


Fig. 4. Referred Concept Hierarchy

Patterns for Backward Refinement

Patterns for backward refinement are classified into the following three types by the structure of focusing basic concept hierarchy.

(i) Focusing on a pair of upper and lower slots which compose a *"slot hierarchy"* and the *"basic concepts hierarchy"* which consists of concepts which have these slots.
(ii) Focusing on a slot, the *"basic concept"* which has the slot and the *"basic concepts hierarchy"* which consists of the concept and its upper concept.
(iii) Focusing on a slot, the *"basic concept"* which has the slot and the *"basic concepts hierarchy"* which consists of the concept and its lower concept.

On the other hand, which kinds of modifications are proposed for ontology refinement In addition, the structure of its referred concept hierarchy affects to which kinds of modifications are proposed for ontology refinement. Therefore patterns for backward refinement are classified into two more types by considering its referred concepts hierarchy as follows;

A) Focusing referred concept hierarchy and basic concept hierarchy have similar structure and there is a concept which can be referred as class constraint for its slot hierarchy in the referred concept hierarchy.
B) There is no concept in the focusing *"referred concepts hierarchy"* which can be referred as class constraint for its slot hierarchy.

In the case of type A), the system can propose two kinds of modification methods; (1) to add new slot on an existing concept in the basic hierarchy and refer to a concept in the referred hierarchy as its class constraint and (2) to add new concept in the referred hierarchy and add new slot which refers to the concept as its class constraint on an existing concept in the basic hierarchy. The users can choose the method (2) when they consider the method (1) is not appropriate, that is, there is no concept which is appropriate as a class constraint of the added slot. In the case of type B), the system can propose only the method (2) because there is no concept which can be referred as class constraint for the added slot.

Based on the above classifications, backward refinement is classified into six patterns by combinations of the above (i) - (iii) and A) - B).

3.5 The Prototype of the Ontology Refinement System.

Based on ontology refinement methods discussed the above, we designed and developed a prototype of ontology contents refinement support system. This system consists of the following three modules;

- *The candidates estimating module* estimates and proposes possible modifications for ontology refinement based on pattern matching discussed in section 3.3 and 3.4.
- *The candidates display and select module* shows the results proposed by the candidates estimating module. The users can select proposed modification they want to apply.
- *The modification apply module* applies the selected modification on the target ontology.

Currently, this prototype system supports for only forward refinement. We are developing supporting function for backward refinement. The system is implemented using Java with some APIS called HozoCore and Hozo OAT (Ontology Application Toolkit) which are libraries deal with ontologies developed by Hozo.

4 Evaluation

4.1 Evaluation Methods

Target ontologies

To evaluate the proposed refinement system, we applied the prototype system to refine several ontologies. Its targets are 9 ontologies built by beginners and 3 ontologies which are developed by ontology experts and open to the public. 4 of them are ontologies which were revised by beginners after meeting ontology experts about given advices for them. They are shown as ontologies whose name have suffix "2" (e.g. race2) in Table.1.

Evaluation criteria

In this paper, we used the following evaluation criteria.

(1) Applicable scope

In the case of forward refinement, the system proposes some modifications focusing on un-referred concepts as support target concept. Therefore, in order to evaluate applicable scope of the method, we calculated the following rate in each target ontology.

(a) The rate of *support target concepts* to *the all concepts* in the target ontology
(b) The rate of *concepts which the system could propose any modifications for refinement* to *support target concepts* in the target ontology

(2) Validity of proposed modifications for refinement

The author evaluated whether proposed modifications are appropriate for ontology refinements. This evaluation was conducted for one of target ontologies built by beginners ("race1" in Table.1).

4.2 Result and Consideration

Applicable Scope

Table.1 shows the evaluation result for applicable scope of forward refinement. The rates of support target concepts to the all concepts in each target ontology are 49% for ontologies built by beginners 61% for ontologies built by experts on average (Tabel.1-(a)). This result is reasonable because of two reasons. First, all concepts do not have to be similar completely. Second, ontologies built by experts have some concept as concepts at middle layer of the ontology. The rates of concepts which the system could propose any modifications for refinement to them are 49% for the beginner's ontologies and 34% for the experts' ontologies on average (Table.1-(b)). We suppose this result shows that the proposed method could cover enough number of concepts as its target for refinement support.

	Ontology made by beginner										Ontology made by expert			
	race1	race2	drums	drums2	traffic	traffic2	rail	rails2	hero_1	average	soccer_ver.12	vehicle	YAMATO101215	average
Group 1((i)A)	1	2	1	1	6	7	2	5	0		6	1	10	
Group1/ All refinement proposal	1%	1%	1%	1%	4%	4%	3%	4%	0%	2%	3%	2%	2%	2%
Group 2((i)B)	2	9	2	2	0	10	3	5	0		0	3	24	
Group1/ All refinement proposal	3%	6%	2%	2%	0%	6%	4%	4%	0%	3%	0%	5%	6%	4%
Group 3((ii)A)	22	34	40	30	20	23	14	19	68		39	6	120	
Group3/ All refinement proposal	29%	24%	43%	36%	14%	13%	20%	14%	49%	27%	23%	9%	29%	20%
Group 4((ii)B)	32	44	23	23	81	80	27	62	67		68	32	148	
Group4/ All refinement proposal	42%	32%	24%	27%	55%	46%	38%	47%	48%	40%	39%	50%	36%	42%
Group 5((iii)A)	3	1	0	0	1	1	0	0	1		0	0	4	
Group5/ All refinement proposal	4%	1%	0%	0%	1%	1%	0%	0%	1%	1%	0%	0%	1%	0%
Group 6((iii)B)	17	49	28	28	39	53	25	42	4		60	22	104	
Group6/ All refinement proposal	22%	35%	30%	33%	27%	30%	35%	32%	3%	27%	35%	34%	25%	31%
Total														
Number of the whole concepts	133	212	135	135	193	251	112	175	50		261	122	561	
Number of all non-reference concepts	79	88	58	58	87	69	77	97	29		148	75	363	
Number of non-reference concepts be found refinement proposal	34	52	34	34	40	41	29	36	11		55	28	101	
Number of all refinement proposal	77	139	94	84	147	174	71	133	140		173	64	410	
(a) Support target concepts/all concepts	59%	42%	43%	43%	45%	27%	69%	55%	58%	49%	57%	61%	65%	61%
(b) Refinement proposal findable rate	43%	59%	59%	59%	46%	59%	38%	37%	38%	49%	37%	37%	28%	34%

(a)The rates of support target concepts to the all concepts in each target ontology.
(b)The rate of each patterns for refinement to concepts which the system could propose any modifications for refinement.

Table 1. Evaluation result for applicable scope of forward refinement.

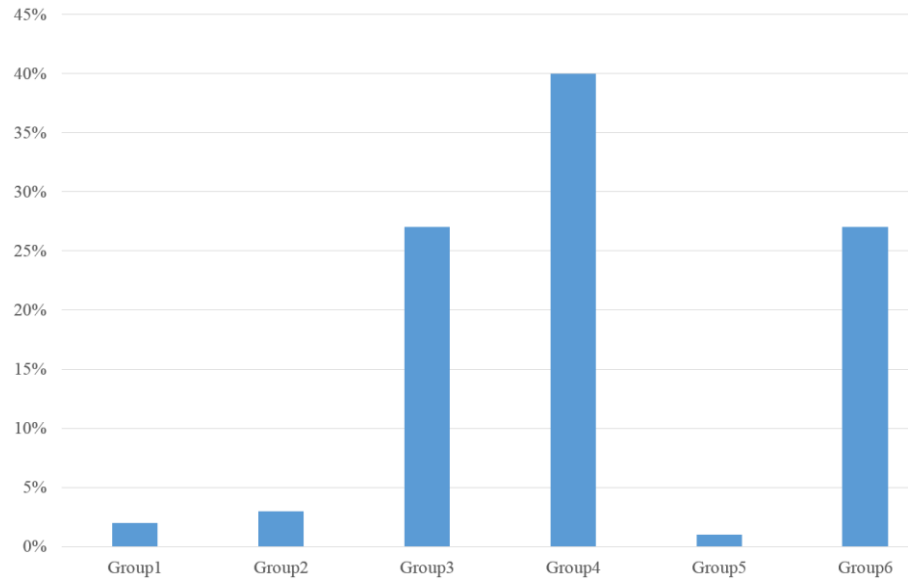


Fig. 5. Distribution of Proposed Refinement Candidates Patterns

Fig.5 said that the number of pattern (i)-A) and (i)-B) are fewer than other patterns. It is because they have stricter conditions to be applied, which consider both of upper and lower concepts of support target concepts, than others. The pattern (i) is also expected higher validity than others because of its condition. On the other hand, the number of pattern (iii)-A) is also fewer than others. It is because the pattern is applied when the lower concept of the support target concept refer a concept which has no upper concept while there is only few such concept.

Validity of Proposal Modifications for Refinement

The author evaluated whether proposed modifications are appropriate for "race2" which was built by beginner. As a result, 20% of the suggestions by this system is evaluated as appropriate modifications (see Table.2).

	Group(i)		Group(ii)		Group(iii)		
	A	B	A	B	A	B	Total
Number of suggested refinement	2	9	34	44	1	49	139
Number of the reasonable proposal	2	5	8	2	0	11	28
Reasonable proposal rate	100%	56%	24%	5%	0%	22%	20%

Table 2. Validity of proposed Modifications for Refinement of race2 ontology

Here, we consider an example of proposed modification which is evaluated as inappropriate. When the system regards "artifact" as a support target concept because it is not referred and its upper concept "entity" is referred by "target" slot at "action" concept, the system makes a proposal that *to add a new lower concept of "action" and add "target" slot whose class constraint is "artifact" on the new concept* [fig.6]. This suggestion means to add a new concept like "artifact target action", but such concept is unnatural to define. Such unnatural examples were appeared at around top level concept in the ontology. Note here that these examples are just unnatural while they are not ontological errors.

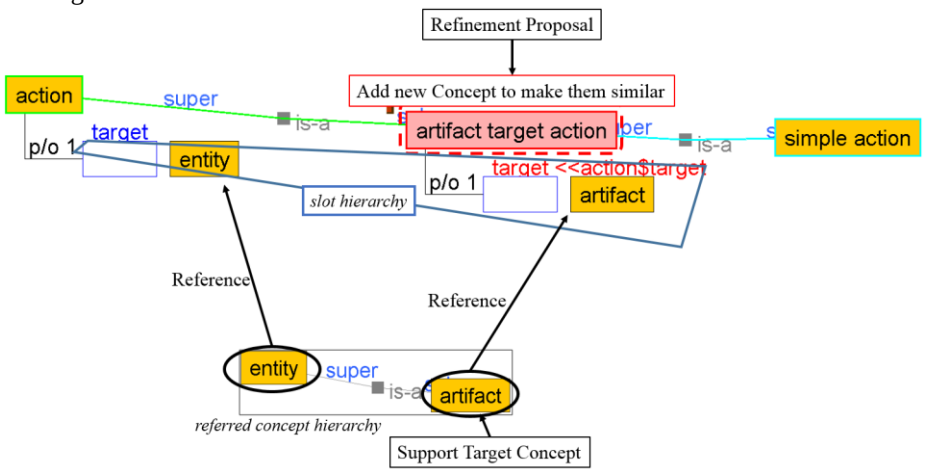


Fig. 6. Inappropriate Example of Refinement Candidate

On the basis of this observation, we removed top level concepts from the support target concepts and evaluated validity of proposed modifications again. As the result, the validity rate promoted 20% to 39% (see Table.3). This suggest us it is effective to remove top level concept from supporting target concepts for improving validity of the propose method.

	Group(i)		Group(ii)		Group(iii)		
	A	B	A	B	A	B	Total
Number of suggested refinement	2	5	18	25	0	21	71
Number of the reasonable proposal	2	5	8	2	0	11	28
Reasonable proposal rate	100%	100%	44%	8%	0%	52%	39%

Table 3. Validity of proposed modifications for refinement of race2 ontology after top level concepts are removed from supporting target concepts.

4.3 Discussion

In the proposed ontology refinement method, the users decide whether they apply suggested modifications or not. So, we believe that 39% of validity is enough high to use it. In addition, inappropriate suggestions are just unnatural while they are ontological correct. Therefore, we think they are not so inconvenient for ontology refinement. However, it is difficult for beginners to decide whether they apply suggested modifications. Therefore, it is an important future work to consider how to compare priority of refinement proposal and how to support finding the most appropriate modification for refinement.

Another important future work is to evaluate the proposed method for more ontologies. At the present, the author evaluated validity of forward refinement for only one ontology made by beginner. We plan to evaluate for more other ontologies and for backward refinement in the future.

5 Conclusion

This article proposed an ontology refinement support system based on similarity among *is-a* hierarchies and evaluated the proposed method partly. The system can support content refinements for ontologies. The feature of the proposed refinement system is to refine ontology semi-automatically. In the future, we will evaluate and consider the backward refinement, and develop the refinement system using forward and opposite refinement and improve the precision of this system's proposal.

References

1. Natalya F. Noy and Deborah L. McGuinness. ``Ontology Development 101: A Guide to Creating Your First Ontology". Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001.
2. Takeshi Morita, Yuka Sekimoto, Susumu Tamagawa, Takahira Yamaguchi. : Building Up a Class Hierarchy with Properties from Japanese Wikipedia. WI-IAT '12 Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Volume 01, Pages 514-521
3. Oscar Corcho [OEG], Catherine Roussey [LIRIS], Luis Manuel Vilches blázquez [OEG], Ivan Pérez [IMDEA]. : Pattern-based OWL ontology debugging guidelines Dans Workshop on Ontology Patterns (WOP 2009), collocated with the 8th International Semantic Web Conference (ISWC-2009)., Eva Blomqvist, Kurt Sandkuhl, Francois Scharffe, Vojtech Svatek. ed. Washington D.C., USA. pp. 68-82. CEUR Workshop proceedings Vol 5. ISSN 1613-0073.
4. María Poveda-Villalón, Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez. : Validating Ontologies with OOPS! Knowledge Engineering and Knowledge Management Lecture Notes in Computer Science Volume 7603, 2012, pp 267-281
5. Benjamin M Mark D. Wilkinson. : Ontology engineering using volunteer labor. WWW '07 Proceedings of the 16th international conference on World Wide Web, Pages 1243-1244
6. Tutorial on ontological engineering - Part 2: Ontology development, tools and languages New Generation Computing, OhmSha&Springer, Vol.22, No.1, pp.61-96, 2004