Semantic Search: An Implementation, Deployments, and Lessons Learned

Seiji Koide[1,3], Masanori Kawamura[1], Takeshi Morita[2], Takahira Yamaguchi[2], and Hideaki Takeda[3]

[1] Galaxy Express Corporation, 1-18-16 Hamamatsu-cho, Minato-ku, Tokyo, Japan, koide@galaxy-express.co.jp
[2] Keio University, 4-1-1 Hiyoshi, Kohoku-ku, Yokohama, Japan, {t_morita,yamaguti}@ae.keio.ac.jp
[3] National Institute of Infomatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan, takeda@nii.ac.jp

Abstract. In this paper, we report an implementation, deployments, and lessons leaned from an early prototype of semantic search, which is equipped with on an Enterprise Search Platform called GXFinder. Each GXFinder share the search indices with other systems, then the users can find documents through enterprise intranet in such environments as documents are scattered at local and distant offices and firms. The objective of the semantic search based on ontologies is to find out potentially valuable documents with user's vague notion and keywords. Several systems were deployed in our company and connected with each other among the head quarter and branches. Tens of systems has been installed in a related company, and the deployment of hundreds systems are still ongoing at the present of writing this paper. Finally, we address open questions on the semantic search from our early experience.

1 Introduction

There exist serious demands for information extraction, sharing, and reuse, against huge and explosive volume of documents in an office. The Enterprise Search Platform (ESP) aims to solve this problem by utilizing search engines emerged from the search over WWW pages. An ESP called GXFinder has been developed in order to enable discovering and sharing diverse documents among distributed offices and firms. It provides means to share search indices among hosts on distributed sites via intranet. Moreover, we have added the semantic search capability onto GXFinder, in which a user can use vague keywords that do not exactly match terms in documents by means of abstraction and specialization of search concepts based on ontological taxonomy. Several systems were deployed at the head quarter and branches in our company. Recently, tens of systems has been installed in a related company, and the installation of hundreds systems are still ongoing at the present of writing this paper. In this paper, we describe the system architecture and implementation of GXFinder and the machinery of semantic search on GXFinder. Then, we report lessons learned from

our early experience. Finally, we address open questions and future works on the semantic search based on ontology.

2 Design of Enterprise Search Platform

2.1 System Architecture

The whole architecture of distributed systems is shown in Figure 1. In the figure, every host has same software configurations and functionalities but each host is dedicated to site-specific documents at different divisions in distance. The host is charged with crawling and indexing division-specific documents on its own site. They are connected each other by peer-to-peer basis through intranet, and communicate other hosts so as to share indices among hosts. A user can search and access any document in the connected environments over the intranet via shared indices under the restriction of the security and the authorized accessibility for documents and users.

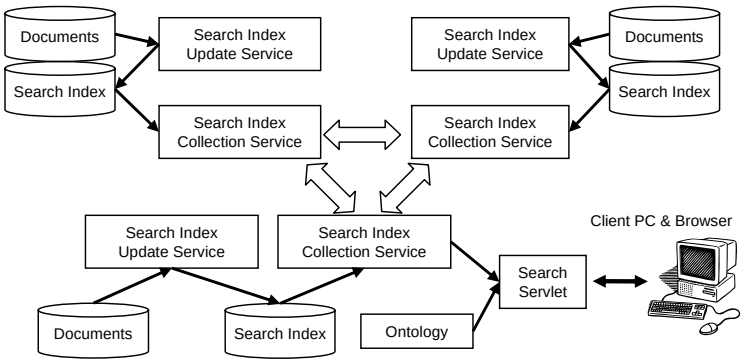

Fig. 1. Connection and Cooperation among Multiple Sites

The system architecture of each host is shown in Figure 2. For the purpose of the rapid prototyping, we utilized several open sources as follows.

- Apache Lucene: Lucene[4] is an information retrieval API set originally implemented in Java by Doug Cutting around 1997. In 2001, the management and control of development is shifted to Apache Software Foundation from SourceForge.net.
- Java.net Sen: Sen[5] is the Java port of Mecab, a Japanese morphological analyzer that originally developed by Graduate School of Informatics of Kyoto University and NTT Communication Science Laboratories.

[4] http://lucene.apache.org/
[5] https://sen.dev.java.net/

- Lucene-Ja: Lucene-Ja is a localized Lucene module to Japanese that is delivered by Sen Project.
- Apache TOMCAT: Apache TOMCAT is used to build a server host for search over intranet.

Note that Lucene has no crawlers. Then, we made a crawler that crawls all documents under specified folders in a site with filter programs for text extraction from various document formats. The crawler retrieves terms contained in documents using Japanese morphological analyzer Sen, and produces an index for search on the site using Lucene APIs.

Apache TOMCAT and Java servlet accept an access to a specific URL and serves as web host by making Web pages designed for search.

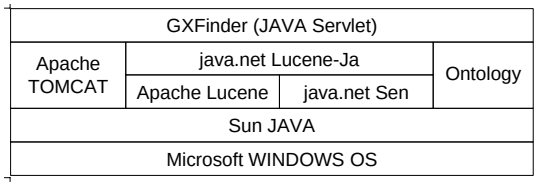

Fig. 2. Software Architecture of GXFinder

2.2 Ontology for Semantic Search

Lucene provides the functionality of search by keywords. Additionally, we have developed a semantic search functionality based on ontological taxonomy of terms. In particular, we have implemented *abstraction concept search* and *specialization concept search*. All terms composes IS-A hierarchy in ontology. Suppose that a user want to search Disney & Pixar's animation DVD in which a little clownfish is a hero. Most senior persons do not remember the exact name of clownfish but know the hero is a tropical fish or a fish. In such case, a user can search it by concept specialization with word "tropical fish" or "fish". By concept abstraction, a user can search the object with word "angelfish", supposing "angelfish" is only one term that a user knows about tropical fishes. In search of concept specialization, special concepts of a keyword in taxonomy are used as keywords (then "clownfish" and others are added into a keyword list instead of keyword "tropical fish" or simply "fish"), and in search of abstraction concept, siblings or cousins of a keyword concept are added into a keyword list (if "clownfish" is a sibling or a cousin of "angelfish" in taxonomy). A user can control the magnitude of abstraction and specialization with the selection of the ambiguity degree, which is described in the next section.

3 Implementation of Web Platform

3.1 Search Service as Web Pages

As mentioned above, GXFinder is a Web application by Apache TOMCAT, so there are no needs of client programs. A user can search documents only if he/she accesses a specified URL address through web browser as usual. The web server controls and maintains a session for search when a user accesses. Figure 3 shows an example of search condition setting page after user log-in. On this page, a user sets the following items up as search conditions.

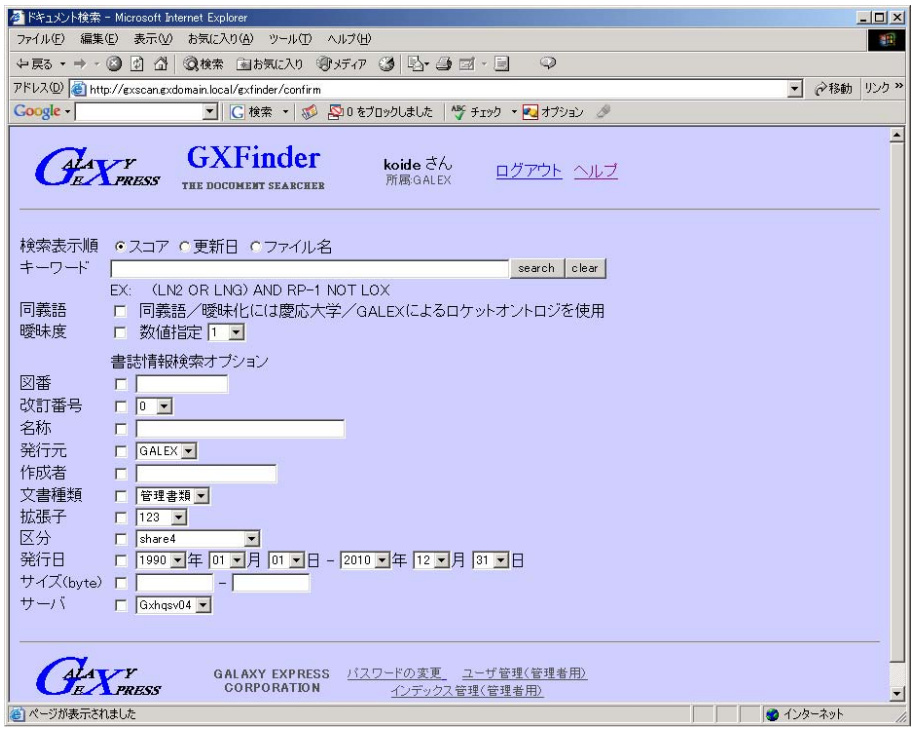

Fig. 3. An Example of Search Conditions Setting Page

- Sort algorithm: select the sorting algorithm that is used in listing search results, i.e., by rating score, updated date, and file name alphabetical order.
- Keywords: AND/OR combination of keywords. If no symbol, the default is AND.
- Synonyms: switch for synonymous search of keywords
- Ambiguity degree: concept search selection. Plus digits mean abstraction search, i.e., 1 for parents (expand to siblings), 2 for grandparents (exapnd to

cousins) concept in taxonomy. Minus digits mean specialization search, −1 for children, −2 for grandchildren. Note that all concepts up to designated super concepts and down to the same level of a keyword concept are used in abstraction search, and all concepts under a keyword concept down to designated sub concept level are used in specialization search.
- File name: these words are used to search by partially matching with file names.
- File extension: designate file extensions for search documents if desirable, e.g., txt, csv, pdf, doc, xls, vsd, jpg, gif, dwg, mpeg, mpg, asf, wm, wma, wmv, and wav[6].
- Publisher: select publisher organizations of search documents. If none, all publishers are targets.
- Document kind: select the kind of search documents. If none all kinds are targets.
- Site: select a target host name for search documents. If none, all documents in all territories are targets.
- Folder: select a target folder name for search documents. If none all folders are targets.
- Date: select a target date published. If none all documents are targets.
- Size: select a target size of documents. If none all documents are targets.

The system shows search results with the list of filenames and a relevant textual portion of the contents via Web pages. See Figure 4, an example of search result listing. When a user points and clicks one of items in the list, the selected document is downloaded and displayed.

3.2 Ontology Construction for Rocket Launch Operation using DODDLE

We have developed a domain specific ontology on rocket launch operation for the in-house usage of semantic search in our company. Figure 5 shows the process of ontology construction we adopted. We collected 13,000 documents that are relevant to rocket launching in our company in two years, then we extracted 15,000 words (only Japanese nouns) contained in the documents with the morphological analyzer and composed them into ontology by using DODDLE-J, which is a Japanese version of DODDLE [Morita2006]. DODDLE is a document centric ontology construction toolkit. DODDLE is capable of semi-automatic construction for a domain-specific ontology with general upper ontologies and materials that describe domain knowledge. We utilized Japanese EDR electronic dictionary [Yokoi1995] as Japanese general upper ontology.

The 11,000 words out of the extracted 15,000 words were general noun words that are included in EDR, and 4,000 words were domain-specific words unregistered in EDR. Using DODDLE, we registered those words manually under the compactly reorganized upper ontology that is automatically extracted and reduced from EDR ontology.

[6] Such multi-media data files are annotated through GXFinder user-interface and the terms in annotation are used for search.

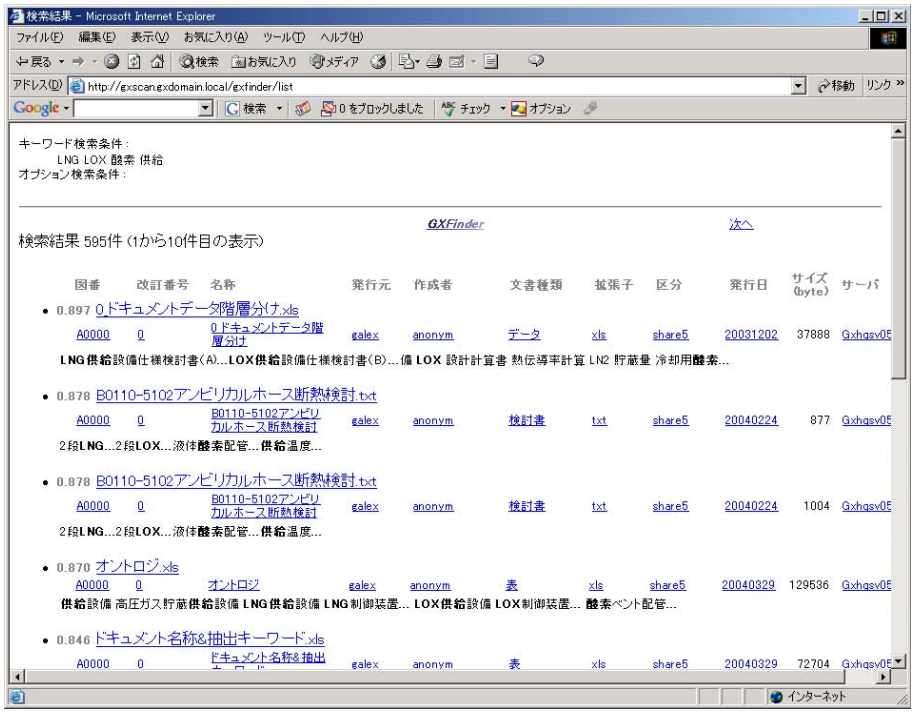

Fig. 4. An example of search result listings

4 Deployments and Lessons Learned

4.1 In-House Usage for Document Search

In 2004, the stand alone local host systems are developed and installed at the head quarter and one of branches in our company. The design engineers welcomed the system and used it in daily work as their convenient search engine in hand. Their requirements for system improvement were effectively reflected to the system. In 2005, the index sharing functionality was developed and the distributed systems were installed at the head quarter and all branches, then they were connected with index-sharing. We are now upgrading the systems as document management and control systems for our formal business process.

4.2 Test and Evaluation on Ontology based Information Retrieval

As another work in 2005, we examined the performance of semantic search based on ontology and evaluated the effect of ontology refinement with the cooperation of design engineers [Ono2006]. In this investigation, we recognized the problem of collocation in science and engineering terms, and invented several ontology

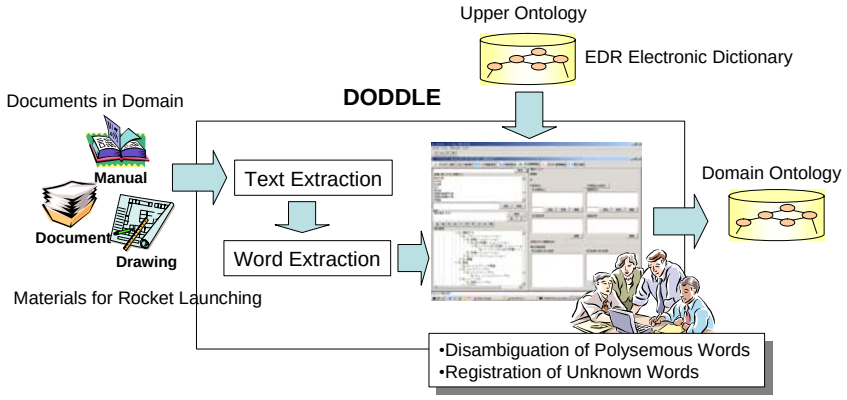

Fig. 5. Process of Document Centric Ontology Construction

mapping algorithms for collocations. The research is still continued but we confirmed that the ontology refinement and specialization search was effective in search. The result will be reported in English soon.

4.3 Application for Operation Support System for Rocket Launching

In 2005, GXFinder was embedded into the Operation Support System for Rocket Launching [Koide2005]. Figure 6 shows a display image of search result listing in the support system. A GXFinder was wrapped up with Web Service and interfaced to other systems in the decision support system. When a contingency event happens, an operation support agent automatically invokes GXFinder with some keywords that are extracted from the situation without user intervention in order to search contingency reports in the past and other relevant documents under similar operational conditions. The search results are displayed to operators.

4.4 On-going Deployment as General Purpose ESPs

In 2006, a related company decided to adopt hundreds of GXFinder systems as general purpose ESPs. The installation is still on-going at the present of writing this paper. At the beginning, we attempted to promote the index-sharing functionality but they would not use it. The reason is that they do not have the perception of needs for document-sharing so far. It may take time for ESP users in an enterprise to perceive the need of connection and cooperation in document search. On the other hand, when we advertised the functionality of conceptual search based on ontology, they showed curiosity for ontology, whereas there are no ontologies for their own engineering domains. They seemed to imagine the potential of ontology in their domains. This fact that ordinary researchers and

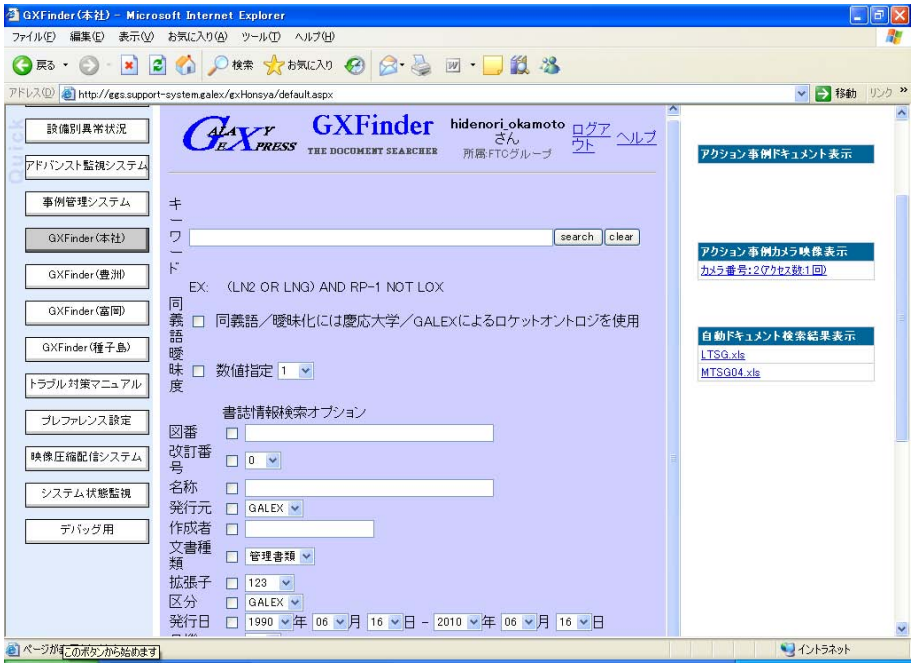

Fig. 6. A Display Image in Search Result Listing of GXFinder Embedded in the Rocket Launch Operation Support System

engineers are interested in their domain ontologies suggests the rationale of the ontology based search. It would be possible that engineers themselves construct and refine their own ontologies, if useful tools and core domain ontologies were available for them.

In an R&D center of the related company, a couple of systems were installed against tera-bytes size of document storage. The index-sharing functionality was utilized for scaling-up in a local site instead of document sharing among distant sites.

5 Open Questions and Future Works

5.1 Rating of Search Documents

As well known, the secret of Google success is due to the page rank algorithm based on the hypotheses that important pages are frequently referred by other pages, which originally comes from rating algorithm for academic papers. It is likely to be true in the light of the reality of the Internet. However, the Google-like rating algorithm is not applicable for ESPs in intranets, because it is rare that technical documents refer other technically related important documents. In this early prototype of ESP, we do not invent any dedicated algorithm for

rating. Lucene computes the document score by a weighted *tf-idf* formula for search precision. We adopted the Lucene original rating points to sort results by document score, whereas we do not investigate the effect of Lucene original formulation for our application. It may be required to invent a new rating formula in future, particularly in conceptual search. It is a basic question for us to have two distant spaces for document search, namely Lucene's *tf-idf* vector space and taxonomic distant space in ontology.

5.2 Richer Ontologies

In order to perform conceptual search in our application, we constructed the domain ontology for rocket launch operation. The ontology for document search is large but rather simple in the sense that the ontology includes only one property, i.e., rdfs:subClassOf in OWL[7] representation. An ontology description may be richer normally in the sense that the ontology may include numbers of properties, from generic ones to domain-specific ones. Suppose that a user makes a query to retrieve articles that describe tanker collisions or strikings on a reef in the Atlantic Ocean. The user can use the IS-A hierarchy for abstract search to obtain siblings of "collision" or "striking", but cannot use the IS-A hierarchy in specialization search for "Atlantic Ocean" to obtain "Caribbean Sea", "Mexican Gulf", "North Sea", "Baltic Sea", etc., because they should be axiomatized with transitive property "locatedIn" instead of rdfs:subClassOf as demonstrated in Wine Ontology[8]. We need richer ontologies that contain various types of properties in OWL representation for more sophisticated semantic search. OWL processor like SWCLOS [Koide2006], which can perform RDFS entailments and OWL entailments, will be available to reason and find out precise documents with document-contained terms and richer ontologies.

5.3 Requirement of Open Source Ontology

We claim that the most problematic obstacle for semantic search in Japan is that we have no free ontologies in Japanese. We have utilized EDR Electronic Dictionary as the upper ontology in construction of the rocket launching ontology. However, it was impossible to bundle EDR to GXFinder in delivery by the reason of cost. In English, WordNet [Fellbaum1998] is available as free resource of upper ontology. It is obvious that we need a good and free ontology in Japanese. We are now going to construct a free ontology like WordNet. We hopefully expect that an open source of upper ontology propels the construction of domain ontologies in various science and engineering fields in the future.

6 Related Works

FindUR by AT&T [McGuinness1997] is the first search engine that used conceptual superclass-subclass relation and class-instance relation of terms. It expanded

[7] http://www.w3.org/TR/owl-features/
[8] http://www.w3.org/TR/2003/CR-owl-guide-20030818/wine.rdf

given keywords to a larger set of terms called a topic set, taking synonyms and hyponyms of keywords. The functionality of the concept search in FindUR is same as synonymous and specialization search in GXFinder. FindUR was applied to several search engines on web sites but it seems the web content categories were fairy limited and the size of pages was rather small. We argue that the semantic search will require more diverse properties than conceptual subsumption and synonym relationships. In practice, we manually constructed a task ontology for rocket launching operation, which is composed of 600 concepts, rather small but richer with complex relations. There we have an explicit requirement to use the task ontology for full-fledged information retrieval.

SemanticOrganizer by NASA [Keller2004] is a collaborative knowledge management system for 500 users over 25 groups (ranging from 2 persons to 100 people) with 45,000 information nodes and 150,000 links. It has been used in NASA at diverse domains from the Space Shuttle accident investigation, Mars mission simulation, the search for life on other planets, to the study of malarial disease in Kenya. The system includes a master ontology over 350 classes and 1,000 relationships that are shared among users. As SemanticOrganizer is a not search system but a semantic repository system, in the repository users can create and interlink items using Web interface that navigates through the semantic network repository, the Semantic Annotator employs WordNet to select relevant items for linking. The user interface restricts the types of links between items based on their item types and the domain/range specifications defined in the ontology. In short, SemanticOrganizer is a semantically linked virtual directory. From the lessons learned in fielding SematnicOrganizaer, it is stated that the attitudes of users to ontologies have wide spectrums. At one end, they appreciated the descriptive power of semantics and did not mind taking the time required to annotate and link the new material appropriately. At the other end, users simply wanted to add their documents to the repository without the additional work to weave them in the semantic network. It is also stated that SemanticOrganizer quickly evolved to incorporate automated knowledge acquisition functionalities to meet the requirement from the reality. The task of adding information to the repository and linking to existing content is time consuming and error prone when the volume of information is large.

We admit that we are still challenged to store the huge volume of information, either way, organizing information pieces in semantically well-formed network, or retrieving useful information from messy information bulk in semantically smart way. We expect that the semantic search reduces the workload of information organization and information retrieval from huge volume of documents and materials.

7 Conclusions

In this paper, we reported the development and the deployment of an enterprise search platform and an early prototype of semantic search. We addressed several open questions and one critical issue upon semantic search based on ontology in

order to grow it into the full-fledged sophisticated semantic search engine. We are going to tackle those problems, i.e., open source Japanese upper ontology, domain ontology construction in science and engineering fields, and the development of full-fledged semantic search engines that is augmented by an OWL reasoner and rich ontologies.

8 Acknowledgements

The main part of this work was carried out as one of the Japanese IT projects entitled "Building a Support System for the Large-Scale Operation System using Information Technology", which was supported by Ministry of Education, Culture, Sports, and Technology (MEXT) in Japan.

References

[Morita2006] Morita, T., N. Fukuta, N. Izumi, T. Yamaguchi: DODDLE-OWL: A Domain Ontology Construction Tool with OWL. 1st Asian Semantic Web Conference (ASWC2006), Beijing, (to appear) 2006

[Yokoi1995] Yokoi, T.: The EDR Electronic Dictionary. Communications of ACM, Vol.38, No.11 (1995-11) 42–44

[Ono2006] Ono, Y., et al.: Constructing a Rocket Operation Ontology and it's Application of an Information Retrieval System based on Japanese DODDLE (in Japanese). The 20th Annual Conf. JSAI (2006) 3B4-4

[Koide2005] Koide, S., et al.: Decision Support System for Rocket Launch Using Semantic Web Services. WWW2005 Workshop Activities on Semantic Web Technologies in Japan (2005)

[Koide2006] Koide, S., H. Takeda: OWL-Full Reasoning from an Object Oriented Perspective. 1st Asian Semantic Web Conference (ASWC2006), Beijing, (to appear) (2006)

[Fellbaum1998] Fellbaum, C. (ed.): WORDNET An Electronic Lexical Database. MIT Press (1998)

[McGuinness1997] McGuinness, D. L., H. Manning, and T. Beattie: Knowledge Augmented Intranet Search. Proc. 6th World Wide Web Conf. CDROM version, Santa Clara (1997)

[Keller2004] Keller, R. M., et al.: SemanticOrganizer: A Customizable Semantic Repository for Distributed NASA Project Team. Proc. 3rd Int. Semantic Web Conf., Hiroshima (2004) 767–781