PAPER Special Section on Data Mining and Statistical Science

Extracting Know-Who/Know-How Using Development Project-Related Taxonomies

Makoto NAKATSUJI[†a)], **Akimichi TANAKA**[†], ***Members***, **Takahiro MADOKORO**[††], **Kenichiro OKAMOTO**[††], **Sumio MIYAZAKI**[††], ***Nonmembers***, ***and*** **Tadasu UCHIYAMA**[†], ***Member***

SUMMARY Product developers frequently discuss topics related to their development project with others, but often use technical terms whose meanings are not clear to non-specialists. To provide non-experts with precise and comprehensive understanding of the know-who/know-how being discussed, the method proposed herein categorizes the messages using a taxonomy of the products being developed and a taxonomy of tasks relevant to those products. The instances in the taxonomy are products and/or tasks manually selected as relevant to system development. The concepts are defined by the taxonomy of instances. That proposed method first extracts phrases from discussion logs as data-driven instances relevant to system development. It then classifies those phrases to the concepts defined by taxonomy experts. The innovative feature of our method is that in classifying a phrase to a concept, say C, the method considers the associations of the phrase with not only the instances of C, but also with the instances of the neighbor concepts of C (neighbor is defined by the taxonomy). This approach is quite accurate in classifying phrases to concepts; the phrase is classified to C, not the neighbors of C, even though they are quite similar to C. Next, we attach a data-driven concept to C; the data-driven concept includes instances in C and a classified phrase as a data-driven instance. We analyze know-who and know-how by using not only human-defined concepts but also those data-driven concepts. We evaluate our method using the mailing-list of an actual project. It could classify phrases with twice the accuracy possible with the TF/iDF method, which does not consider the neighboring concepts. The taxonomy with data-driven concepts provides more detailed know-who/know-how than can be obtained from just the human-defined concepts themselves or from the data-driven concepts as determined by the TF/iDF method.

key words: *taxonomy, knowledge management, know-who/know-how*

1. Introduction

Many system development projects, such as those for banking systems and network management systems, require the collaboration of developers in various development divisions within the company. This collaboration may include creating basic or detailed system designs, implementing module prototypes according to the designs, and adding or testing modules. Developers responsible for handling modules or development procedures often collaborate with each other in the course of their work. Given the long development schedules common for complex projects, some turn over of personnel must be accepted. It is essential that the new people be able to utilize the know-who and know-how information created by the original experts as contained in the logs of message systems.

Semantic web techniques allow important concepts to be analyzed and extracted as classes within an objective domain, with the goal being to create a domain ontology comprising classes, relationships among classes, and instances that are members of each class [4], [14]. By referring to this ontology, users can acquire detailed knowledge about the objective domain so described. Several researchers have also adopted the idea of using a task ontology to describe the relationships among objective classes in a domain ontology and tasks related to those classes [18]. Referring to these created ontologies provides opportunities for non-expert users to acquire detailed and precise expert knowledge.

It is difficult, however, for product developers to develop the detailed, comprehensive relationships of concepts required to create a regular ontology. Our idea is to semi-automatically create taxonomies of developed products and of tasks related to those products by analyzing the mailing-lists of the product developers generated over the course of the project, and then extract the know-who/know-how information of that project. A taxonomy is defined as a concept hierarchy. Each concept in the taxonomy has manually defined developed products or tasks relevant to those products as instances. The concepts are defined by the taxonomy of instances.

To build a taxonomy automatically, several ontology learning studies [7], [24] first analyze key phrases that represent the topic discussed in a document, and create the hierarchical relationships between those phrases extracted from a document set. [7] extracts the key-phrases as tags from blog entries using the tf/iDF method, and attach those to the blog entries. They then create the tag hierarchy by analyzing the subsumption relationships between tags, to visualize the knowledge discussed in blogs. Their approach is similar to ours from the viewpoint that they first extract key-phrases from the document set, however their aim is to build tag hierarchy out of nothing. We consider that when applying ontology learning techniques to extract knowledge in a domain such as system development domain (in which the expert knowledge is needed), an ontology generated out of nothing is too ambiguous to permit its reuse by the system developers in further system development tasks. We also consider that phrases attached to a class in the taxonomy should be specific phrases against that class if we are to express such

Manuscript received November 30, 2009.
Manuscript revised April 2, 2010.

[†]The authors are with the NTT Cyber Solutions Laboratories, NTT Corporation, Yokosuka-shi, 239–0847 Japan.
[††]The authors are with the Research and Development Center, NTT West Corporation, Yokosuka-shi, 239–0847 Japan.
a) E-mail: nakatsuji.makoto@lab.ntt.co.jp

DOI: 10.1587/transinf.E93.D.2717

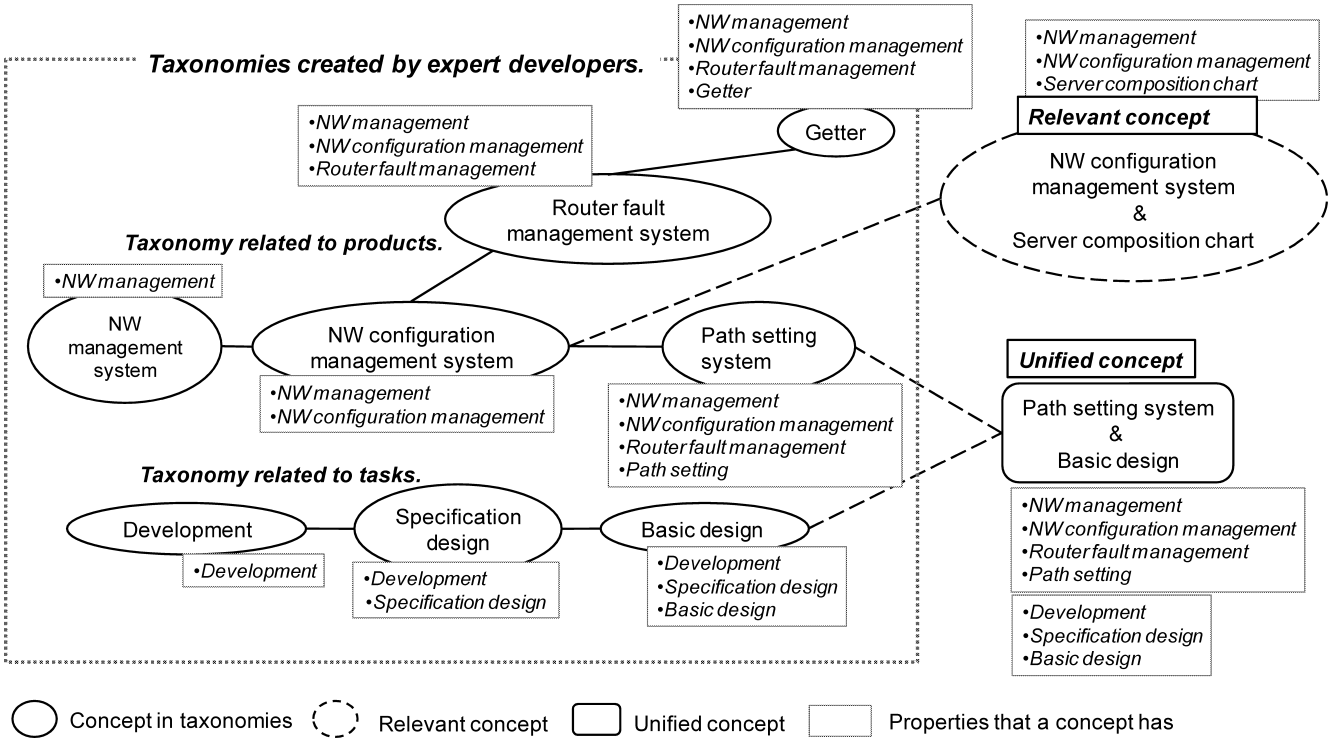


Fig. 1 Image of taxonomy with relevant and unified concepts.

detailed expert knowledge.

Our solution starts with preparing small taxonomies built by taxonomy experts who participated in a long-running development project. These taxonomies are small enough that they could be built by developers even while pursuing their current activities. Next, we extract phrases used for product development by analyzing the mailing-lists of messages about the project and classify those phrases to the concepts defined by taxonomy experts. We calculate the association between phrase p and the instances in the concepts, and classify p to one or more concepts if the association level is high. The method proposed herein allows for the creation of new data-driven concepts: relevant concepts C_r and unified concepts C_u. Relevant concept is defined by the instances in a certain concept C_i and the extracted phrase as a data-driven instance that is relevant to the concept C_i. Unified concept is defined by the instances in a concept in product taxonomy and a different concept in task taxonomy. We call the human-defined taxonomy with data-driven concepts the expanded taxonomy in this paper. The former has strong association with one of the concepts in the original taxonomy while the latter are associated with both product and task concepts as shown in Fig. 1. The goal is to allow a non-expert to easily grasp the accumulated know-who/know-how by referring to the expert knowledge identified by the expanded taxonomy.

The main contributions of this paper are described below.

- We classify phrase p to concept C in the taxonomy by considering not only the association between the instances in C and p, but also the associations between the instances in the concepts near C (as defined by experts who created the taxonomies) and p. In this way, we can accurately discriminate which concept the phrase should be assigned to.
- We can extract know-who/know-how information in more detail by using a taxonomy with data-driven concepts than we can by using only small static taxonomies. We can also analyze know-who/know-how in more detail by referring to the actual contents of the e-mails classified under C_r and C_u. This has the effect of taxonomy enrichment as well as yielding concrete application examples that can drive forward the taxonomy (or semantics) based knowledge management.

We evaluated our method using small taxonomies created by a few taxonomy experts and product developers who participated in an actual project and the actual e-mails used to discuss and advance the project. As a result, our method classified phrases as data-driven instances to the concept twice as accurately as the TF/iDF method. Furthermore, our method provided more detailed know-who/know-how information than could be acquired by using only the expert-created concepts themselves or the data-driven concepts as extracted by the TF/iDF method.

The rest of this paper is organized as follows. We describe related works in the next section. We then propose our method of classifying phrases to concepts to create the expanded taxonomy and show how to apply the expanded taxonomy to analyze know-who/know-how in Sect. 3. Section 4 evaluates the effectiveness of our method using actual

mailing-lists created in a project; Section 5 concludes the paper.

2. Related Works

Taxonomies have been created in various service domains on the Web. The DBPedia project [3] extracts structured information from Wikipedia and makes this information available on the Web. The authors in [27] extract a taxonomy from English Wikipedia, by inducing *isa* and *notisa* labels for the edges of the category network. [22] automatically distinguishes between instances and classes in large scale taxonomies derived from Wikipedia.

As for practical studies that try to express business processes as formal knowledge, [17] extracts taxonomy concepts of business activities. We also find eTOM (eBusiness and expanded Telecom Operations Map)[†], which defines taxonomies of main business processes related to telecom business activities. However, the concepts in these taxonomies are defined in a rather abstract manner because they make use of knowledge common to a wide variety of business domains or a wide variety of sub-domains within those domains. Thus, we cannot simply use them for expressing knowledge specific to actual system development projects.

Several studies have been conducted to reuse knowledge distributed in social networks by formalizing knowledge in approaches, such as ontologies, which must be carefully created by humans [4], [14], [25]. The authors in [14] developed a system that automatically classifies browsed web pages against an ontology, and allows users to share comments made on these pages. As the user browses web pages, recommendations of relevant documents which have already been shared are produced, based upon both the user's social network as well as the semantic content of the page currently in view. However, creating the formal knowledge needed to create an ontology by hand in a specific domain is still too hard for non-specialists.

There are several studies on ontology learning that aim to generate ontologies automatically from text document sets [5], [7], [9], [11], [24]. Several researchers use Hearst-patterns such as "X, Ys and other Zs" or "Ws such as X, Y and Z" [11] to extract *is-a* relationships from the documents. Here, X, Y, Z, and W mean the words in the sentence. Those methods can acquire quite accurate word to word relationships, though they fail to acquire a lot of relationships. [1] used a large amount of web pages using Google search API to acquire a lot of relationships. We consider Hearst-patterns-based methods may acquire quite accurate relationships of concepts or those of instances, but they will often fail to analyze the relationships of key-phrases in a certain domain because they only focus on the linguistic lexical patterns, and do not analyze the key-phrases used specifically with regard to the topic discussed in that domain.

On the other hand, [7] extracts the key-phrases from blog entries using the TF/iDF method, and then creates the tag hierarchy by analyzing The subsumption relationships between tags. The TF/iDF method is often used when measuring the associations between a term and document set S. Expressing TF as the number of occurrences of a term in document set S, where $|D|$ is the total number of documents and DF is the number of documents that include the term, TF/iDF determines the associations between the term and a document set S as $TF/IDF = TF \times \log \frac{|D|}{DF}$. We can consider that the term and document set S are related if the association value is high. However, as explained in the Introduction, their method aims to generate key-phrase hierarchies out of nothing; the generated hierarchy is likely to be too ambiguous for the developers to acquire the detailed experts knowledge needed in domains such as the system development domain.

There are studies that target the enrichment of existing ontologies or taxonomies by extracting data-driven topics from a document set. In [8], each concept is located in a simple ontology and has a manually defined set of words. On the other hand, topics can be learned automatically from a text corpus by applying a statistical topic model, such as the latent Dirichlet Allocation (LDA) model [6]. The authors in [8] train a probabilistic model based on LDA over the words in a document and concepts. They then map a document to concepts in a given ontology. Ontology enrichment is addressed by the authors of [10]; they extract phrases relevant to instances in an ontology by using Latent Semantic Analysis (LSA) [15]. They then create classes that have extracted phrases as their instances. In addition, [12] uses concepts defined within WordNet[††] to preprocess the texts when clustering text documents. In [13], the authors map words in text documents onto concepts defined within WordNet, and use them in their proposed learning process for document classification.

For taxonomy or ontology enrichment, [2] classifies the document or sentence to the concept in a given taxonomy by checking the co-occurrence of the concept name and the words in the document or the sentence in the document. Those works are related to ours because both approach adopt document or sentence classification for taxonomy/ontology enrichment. However, [2] classifies documents from the root concept in the taxonomy to their child concepts by checking the co-occurrence frequency of the concept name and the words in the sentence accumulated in each concept. As a result, the classification of the sentence to end concepts in the taxonomy often fails, when the co-occurrence frequency between the name of concepts in an upper hierarchy level and the sentence is lower though that between concepts in a lower hierarchy level and the sentence is high.

Our approach differs from the above approaches and methods because we classify phrase p used in discussion message among developers to concept C in a taxonomy by considering not only the associations between p and instances in C, but also the associations between p and instances in concepts near C (as defined by experts who cre-

[†]http://www.tmforum.org/browse.aspx?catid=1647
[††]http://wordnet.princeton.edu/

ated the taxonomies). Our method does not classify the phrase from root concept to its child concepts. It directly checks a certain target concept in any hierarchy level in the taxonomy to calculate the association of that concept name and the phrase also considering whether any concepts are more strongly associated with the target phrase among the concepts that are near to the target concept. As a result, we can extract phrases whose characteristics approach those of C but are different from those of the sibling concepts of C.

In [26], the authors try to automatically identify communities of practice within an organization. Their method uses e-mail data to construct a network of correspondence, and then discovers the communities by partitioning this network. The only pieces of information used from each e-mail are the names of the sender and receiver. We, on the other hand, analyze topics about a development project within mailing-lists using taxonomies related to the project. Thus, we can extract know-who/know-how comprehensively by referring to the concepts in the taxonomies.

In [19], [20], we proposed user interest extraction from users' blog entries following the taxonomy of content items that is created by content providers, and similarity measurement between those extracted user interests. The evaluation in [19], [20] shows that the proposed method can accurately extract user interests from blogs and measure the similarity of users. In this paper, we apply the above previously proposed method to knowledge extraction from the mail messages accumulated in mailing lists. The problem is that there is no taxonomy related to system development, thus the main technical contribution in this paper is taxonomy enrichment from the mails accumulated in the mailing list for system development. Using this expanded taxonomy, we can analyze know-who/know-how accurately as we show in our evaluation.

3. Method

We first describe the design of the small taxonomies and then explain our method of classifying phrases as data-driven instances to concepts in the taxonomy. We then explain creating data-driven concepts (relevant concepts and unified concepts) using classified results and creating expanded taxonomy by attaching those data-driven concepts to the human-created concepts in the taxonomy. Next, we explain how to use the expanded taxonomy in analyzing know-who/know-how information.

3.1 The Design of Taxonomies

We explain the procedure used to design the taxonomies using the example in Fig. 1.

(1) First, a designer chooses the project for which to knowledge management is to be realized. We believe the best project granularity is that found in the mailing-lists, because mailing-list participants share their in-depth knowledge of the project by exchanging messages.

(2) The designer creates a taxonomy of products and another of the tasks relevant to those products. An example is shown in Fig. 1. Each concept has instances tagged with manually-defined name attributes that express the product or task itself. Name attribute of an instance means the name of the instance as assigned by humans. This taxonomy follows the notion of the prototype-based ontology in [5]. Concepts in the prototype-based ontology are distinguished by typical instances rather than by axioms and definitions in logic. Concepts are formed by collecting instances extensionally rather than describing the set of all possible instances in an intensional way, and selecting the most typical members for description. This idea is suitable in creating our taxonomy because developers do not have enough time to describe the set of all possible instances in an intensional way. Here, we give the concept name using one of the name attributes of the instances in the concept, for convenience following the notion of the prototype-based ontology. In the prototype-based ontology, concepts are formed by collecting instances extensionally. Thus, in fact, it does not set concept names in the taxonomy.

The taxonomies express only a hierarchical relationship among the concepts and restrict the succession conditions of the hierarchy. For example, the concept "router fault management system" in Fig. 1 has the property of "NW management" via the concept "NW management system", the parent of the "router fault management system". The property of the concept is used for restricting the characteristics of the concept. We can assign several instances to concepts. For example, there are the instances "NW management system", "NW operation system", "Network operation system" and "Network operation system" in the concept "NW management system".

3.2 Method

We now explain our method. First, we assign e-mails into the most relevant concepts in the small taxonomies.

3.2.1 Assigning E-Mails to Concepts

We can apply our previous method of classifying blog entries into taxonomies [19], [20], to classify e-mails into the concepts of the small taxonomies. However, we should consider that e-mails differ from blog entries in at least one characteristic; i.e., we discuss various topics in an e-mail thread exchanged over a period of time. We divide each thread in the mail log into mails exchanged on a week-by-week basis, because we think message topics of a project thread are apt to change every week. We call these weekly blocks of threaded e-mails a "mail set" hereafter.

We first remove citations (lines beginning with >) or signatures (lines that include physical or mail addresses) from the mails in each mail set. An example of the procedure for classifying mail sets into the concepts of the small taxonomies is shown in Fig. 2. In Fig. 2, we give the concept name using one of the name attributes of the instances in the concept, for convenience.

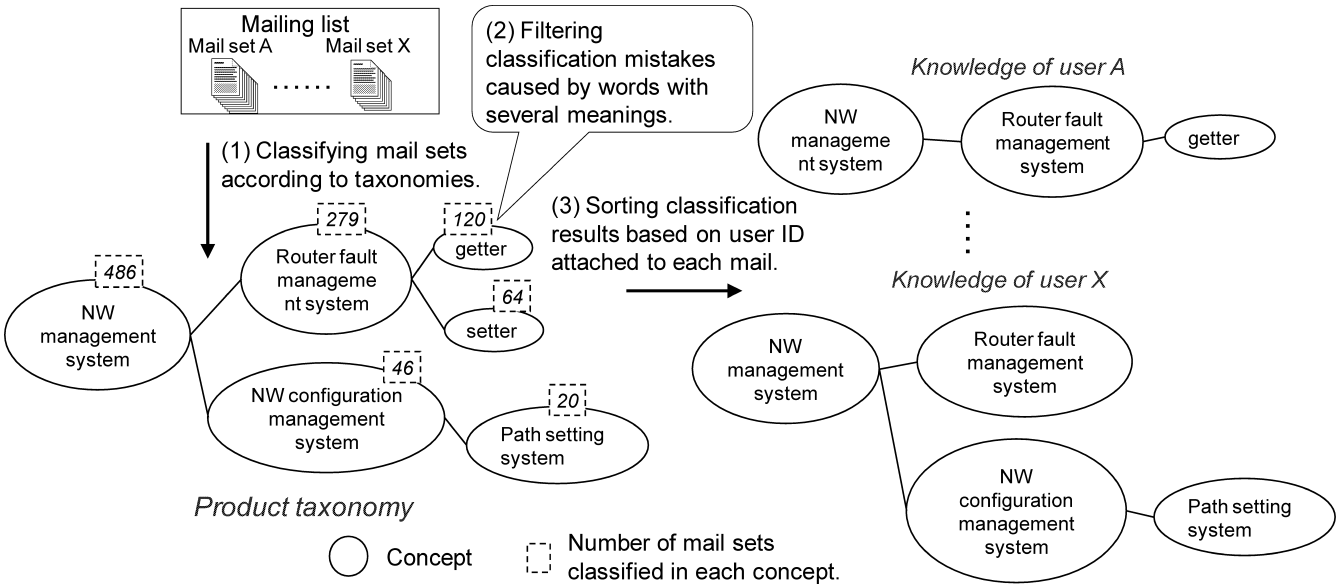


Fig. 2 Procedure of assigning e-mails.

(1) If there is more than one name attribute of instances in C_i in the descriptions of mails in a mail set, we classify the mail set into C_i. A mail set can be classified into several different concepts.

(2) We then eliminate classification mistakes caused by words with several meanings by filtering. For example, the name attribute of instance "getter" has several different meanings; e.g., a product that collects NW management information from NW nodes such as a router or server, or a JAVA language method that returns the value of a field. In this paper, we explain the filtering algorithm below. It uses the characteristics present in the taxonomies; neighboring concepts of concept C have properties that are close to those of C.

If there are name attributes of instances in C_i in mail m of mail set S, the filtering algorithm checks whether more than one name attribute assigned to instances exists in the concepts of neighbors of C_i in the descriptions of mails of S. If such name attributes exist in the descriptions, we classify m and S into C_i, considering they include topics for instances related to C_i. If no such name attributes exist in the descriptions, we filter m of S by not classifying them into C_i. In Fig. 2, mail m of mail set S is classified into the concept "getter" under the concept "NW management system" if there are descriptions of both "getter" and "NW management system" in the mails in S.

We can adjust the range of neighboring concepts of C_i by using hop counts, which are defined as the number of concepts between the concept being considered as a neighbor and C_i. For example, neighboring concepts are only parent concepts or sub-concepts of C_i if the hop count limit equals one. The concept "router fault management system" in Fig. 2 is a neighbor of "getter" if the hop count is set to one. If the hop count equals two, neighboring concepts also include grandfather concepts or sibling concepts of concept C_i. The "setter" and "NW management system" concepts are neighbors of "getter" when the hop count is set to two in Fig. 2.

3.2.2 Attaching Data-Driven Concept to Human-Created Concepts

Next, we explain how our method classifies phrases as data-driven instances into concepts in expert-prepared taxonomies.

(1) We first extract phrases relevant to the development project from the mailing list by using a corpus created by termEX (http://www.r.dl.itc.u-tokyo.ac.jp/~nakagawa/resource/termext/atr-e.html). TermEX is a tool for automatically extracting technical terms from the corpus of technical domains. Developers in the project then manually removed the phrases that were not relevant to the project, and extracted phrase set P that is relevant to the topics of the project.

(2) Next, we classify phrases to concepts in taxonomies. Concepts in taxonomies precede the properties of their parent concepts, and concepts in a deeper hierarchy of taxonomies are closer only to their parent concepts. Considering this characteristic, we assign the topic of phrase p_l to C_i if p_l is closer to the instances in C_i than to those in the neighbor concepts of C_i. Neighbor concepts of C_i are determined by the taxonomy.

Now, we explain the procedure of assigning a new relevant concept C_r to concept C_i.

(a) We first select concept C_j that is a neighboring concept of C_i but is not an ancestor concept of C_i, because C_i precedes the properties of ancestor concepts.

(b) Next, we measure the occurrence rate $|p_l \in S(C_i)|$ of phrase p_l in mail sets $S(C_i)$ that are classified to C_i. If we classify p_l to concept C_i or to neighboring concepts of C_i only based on occurrence rate, almost the same phrases

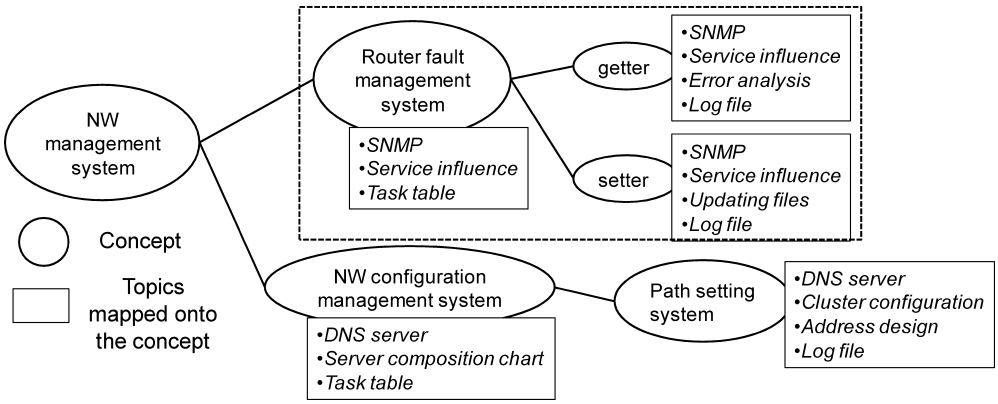


Fig. 3 Explanatory image of classifying phrases to concepts.

will be classified to C_i and its neighboring concepts, because the properties of the neighboring concepts of C_i are close to those of C_i.

To solve this problem, we propose a method that measures the association score $U(p_l)$ between instances in C_i and p_l as expressed by Eq. (1).

$$\frac{\frac{|p_l \in S(C_i)|}{\sum_{p_o \in P} |p_o \in S(C_i)|}}{\frac{|p_l \in S(C_i)|}{\sum_{p_o \in P} |p_o \in S(C_i)|} + \frac{|p_l \in S(C_j)|}{\sum_{p_o \in P} |p_o \in S(C_j)|}} \quad (1)$$

For example, if the value of $U(p_l)$ is greater than 0.5, phrase p_l occurs more often in $S(C_i)$ than $S(C_j)$. Thus, p_l has a higher association with C_i than C_j.

(c) We repeat procedures (a) and (b) to examine X number of neighboring concepts C_N of concept C_i, and measure the associations between C_i and C_j in C_N. Next, we acquire $\bar{U}(p_l)$, the average score of $U(p_l)$ among the X neighboring concepts of C_i. If $\bar{U}(p_l)$ is greater than heuristic parameter α, we consider p_l to be a phrase specific to C_i.

Here, phrases specific to C_i are fewer if C_j is closer to C_i in terms of hop counts as defined in Sect. 3.2.1. This is because the semantic similarity defined by the taxonomy between C_i and C_j is too close. Thus, we adjust the semantic similarity between C_i and C_j by changing the hop counts. The main advantage in using taxonomies is that we can measure the association in detail by considering not only the similarities between instances in concepts and phrases in p but also those among instances in neighboring concepts of C (which are selected by using hop counts) and phrases in p. Here, we don't use all neighboring concepts of C. We select concepts in which have almost the same amount of mail sets are classified as those classified in C, and the less similar concepts (which are defined by taxonomies) with C among all neighboring concepts.

For example, we can acquire the specific phrase "SNMP" and remove the buzz phrase "log file" for the concept "getter" if C_i is set to "getter" and the hop count is set to more than four as shown in Fig. 3. We can also remove "SNMP" and obtain the more specific phrase "error analysis" if we set the hop count to a small value such as two.

(d) Finally, we assign the relevant concept C_r, which is defined by the instance set of C_i and phrase p_l (as a data-driven instance), to C_i.

3.2.3 Analyzing the Relationships between Products and Tasks

We analyze the mail messages that include instances in concept C_i in the product taxonomy and those in concept C_j in the task taxonomy, because we consider that both sets of instances may include know-how. Thus, we create the unified concept C_u, which shares the instances of C_i and those of C_j.

We create unified concept C_{ij} that comprises the topics for instances in C_{ij} in mail sets $|S(C_i) \cap S(C_j)|$ if the association between concept C_i and C_j, defined as $\frac{|S(C_i) \cap S(C_j)|}{min(S(C_i), S(C_j))}$ using the Simpson coefficient [23], is Y highest among all candidate concepts C_j. Here, $S(C_i)$ and $S(C_j)$ are mail sets classified into C_i and C_j, respectively.

3.2.4 Classifying Mails to Concepts in Expanded Taxonomies

We can extract user knowledge as explained in Sect. 3.2.1 by classifying mail sets to taxonomy concepts combined with data-driven concepts. We classify a mail set to C_r if the set shares the name attributes of instances (both human-created instances and data-driven instances) in both C_i and C_r. We also classify a mail set into C_u that unifies C_i (product taxonomy) and C_j (task taxonomy) if the set shares the name attributes of instances in both C_i and C_j.

3.2.5 User Knowledge Extraction from the Classified Results

Finally, we can obtain the system development knowledge of each user by sorting classification results based on the user ID attached to each mail.

3.3 Analyzing Know-Who Using Expanded Taxonomy

We analyze know-who by introducing a knowledge score

that defines the degree of knowledge of each user for each concept in the taxonomies. The following definitions are taken in our previous work [19], [20].

(1) The weight of knowledge of each mail is one. (2) If mail m_i has $N(m_i)$ kinds of name attributes of instances that are located in different concepts, the knowledge score of a concept in m_i becomes $1/N(m_i)$. (3) If we define the set of mails published by a user as M, the score of knowledge $N(C_i)$ of each concept C_i is $N(C_i) = \sum_{(m_i \in C_i)}^{|M|}(1/N(m_i))$. $|M|$ means the number of mails in M. (4) The score of knowledge of the concepts is reflected in that of the parent concept. If we have concept set C that includes subconcepts, relevant concepts, and unified concepts under concept C_i, the score of knowledge under concept C_i is given by $N(C_i) + \sum_{C_k}^{|C|}(N(C_k))$. $|C|$ means the number of concepts in C.

We can analyze user knowledge under concept C_i from the user knowledge to extract relevant concepts or unified concepts under concept C_i. Thus, we can identify a user as having a high degree of knowledge if the user frequently discusses not only instances in concept C_i but also those in relevant concepts or unified concepts. Thus, our approach, combining concepts with data-driven concepts, is very useful in analyzing know-who because in discussing the instances of concept C_i, developers often talk about related topic of C_i such as topics for data-driven instances, and frequently omit direct references to C_i and its instances themselves.

3.4 Analyzing Semantic Relationships between Similar Users

We can measure the similarity of knowledge among developers by using extracted user knowledge. Collaborative filtering techniques [21] usually express the degree of knowledge of user u_a about item i as element v_{ai} of vector $\mathbf{V_a}$. They then calculate the similarity between users based on the covariance of vector $\mathbf{V}$ of users. In our previous work [19], [20], we proposed a method of extracting user interests according to a taxonomy of content items by analyzing the descriptions of users in their blog entries, and measured the similarities between the interests of users.

Combining the extracted user knowledge based on concepts with data-driven concepts yields more accurate similarity measurements. Developers engaged in actual system development projects often discuss details of modules that are not listed in our small taxonomies. Similarity measuring methods can judge the similarity by using not only the concepts in our small taxonomies but also the relevant concepts or unified concepts assigned to C.

One of the important advantages of using taxonomies is that we can assign taxonomy-based semantic tags to the relationships between users as shown in Fig. 4. When using conventional methods that do not use taxonomies in expressing user knowledge, we can only set keyword-based tags to the relationships between users with similar knowledge. In this case, we should check the similarities and differences in their knowledge by checking all keywords, such as similarities in postgres or oracle and differences in zone or address design assigned to the relationships. However, it is a heavy burden to check all keywords assigned to the relationships between users, and the user may not initially understand such keywords. On the other hand, users can semantically understand the similarities between user A and B as defined in "Address Design" under the concept "Path Design", and differences in "Oracle" under the concept "Database (DB)" by referring to the semantic tags assigned to their relationships.

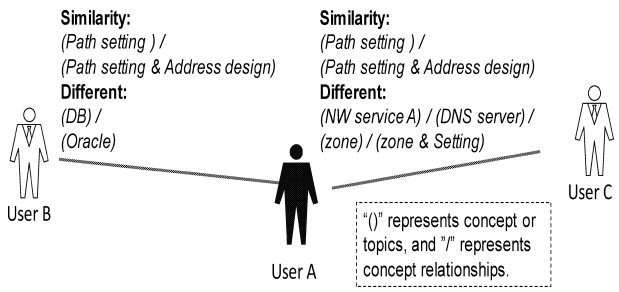


Fig. 4 Assigning taxonomy-based semantic tags to the relationships between users.

3.5 Analyzing Know-How Using Expanded Taxonomies

Users can acquire know-how of each concept by referring to the mail sets as classified into the concepts or relevant concepts of expanded taxonomies. By following the expert knowledge contained in the taxonomies, they can acquire know-how of a project even if they are new to the project. In particular, they can acquire a lot of know-how about the development activity of each module by referring to the unified concept which includes topics of each module and each task activity. When a user wants to browse information about a concept, we can provide mail sets in the order of those that contain the highest number of name attributes of instances in the concept.

4. Evaluation

We evaluated our method using the simple taxonomies created by expert developers who participated in a NW management system development project, and the mailing list created during the project.

4.1 Dataset and Methodology

The mailing lists used in this evaluation were created by 111 developers and included 23,833 e-mails sent from April 2006 to June 2007; there were 13,342 mail sets and 8,913 mail threads. The product taxonomy contained 49 concepts; the task taxonomy contained 57 concepts. These taxonomies were created by three development engineers over a period of six hours and, on average, had three hierarchy levels. Furthermore, they contained 156 instances, thus the engineers defined each concept by giving it at most two instances. We took three hours to extract a phrase set, which

contained 6132 phrases, from the mailing lists by using termEX. Phrases that consisted of only a single written character, those consisting of only "to be" verb forms, etc. were eliminated. The proposed method created 1,469 relevant concepts.

In evaluating our method, we focused on the following three points: (1) How well each relevant concept mirrored the characteristics of its original concept in the taxonomy. (2) How effective our method was in extracting know-who. (3) The effectiveness of our method in extracting know-how.

We evaluated these three points by carefully checking the results obtained from three expert development engineers in this project. In this way, we were able to avoid mistakes and changes in their answers. In our evaluation, we set parameter X, which is the number of selected neighboring concepts of the evaluation target concept as explained in Sect. 3.2.1, to 3, 5, and 10. We assumed that few relevant concepts would be created when X was three, and many relevant concepts would be created when X was ten. α, which is a parameter used in creating relevant concepts as explained in Sect. 3.2.1, was varied from 0.8 to 0.95; we found that result accuracy increased with α, but fewer relevant concepts were created. Thus, we set α to 0.85. Furthermore, we set hop counts to five in Sect. 3.2.1, and set Y, which is a heuristic parameter used to judge the association strength of concepts in different taxonomies (product taxonomy and task taxonomy) as explained in Sect. 3.2.3, to 5. We compared our method to a method that does not add relevant concepts to taxonomies and a method that used the conventional TF/iDF method to add relevant concepts to taxonomies.

In a previous experiment, we evaluated the accuracy obtained when classifying user descriptions into the concepts in taxonomies, as explained in 3.2.1 in [19], [20]. This prior experiment confirmed that high classification accuracy was possible even though it focused mainly on classifying blog entries.

Accordingly, in this paper we do not describe the accuracy of evaluation in classifying the results of mail sets in much detail. Briefly, we were able to achieve highly accurate results by checking several messages that were manually classified because we used mailing lists closely related to the taxonomies used for classifying the mail sets, and because we used taxonomies that consisted of concepts whose name attributes of instances were technical phrases that were used rather specifically in the project. (The accuracy of classifying mail messages is also reasonable given the high accuracy of creating relevant concepts.) However, several concepts had instances whose name attributes consisted rather general words such as "problem" and "verify". Thus, for the task taxonomy in particular, the classification results we obtained were not wholly accurate.

4.2 Evaluating the Characteristics of Relevant Concepts

First, we evaluated the adequacy of relevant concepts. We prepared two types of correct answers created by development experts. In the first type, we call the corresponding answer set answer set (a), the topics were close to the characteristics of C and/or those described in e-mails that may include know-how information. Second, we call this answer set answer set (b), the topics were not close to the characteristics of C but the e-mails describing them may include know-how information. We then evaluated 10 concepts randomly selected from among the 116 concepts in our taxonomies. Table 1 shows the results obtained when checking the top five or ten topics related to each concept. These results suggest that the proposed method can extract correct answers with twice the accuracy possible with TF/iDF when we use correct answer set (a). On the other hand, both methods achieved almost the same accuracy against correct answer set (b). The reason for the superiority of the proposed method is that it produced fewer inaccurate relevant concepts.

Table 1 Accuracy of extracting relevant concepts.

		Answer set (a)	Answer set (b)
TF/iDF	Top 5	0.24	0.66
	Top 10	0.23	0.53
Proposed method	Top 5	0.60	0.68
	Top 10	0.47	0.55

4.3 Evaluating the Effectiveness of Know-Who Analysis

The effectiveness of the method's know-who analysis was checked, because know-who information is a critical component of product development. In the evaluation, three development experts assessed the "experts" identified by the method in 10 randomly selected concepts out of the 49 product concepts. Each "expert" was assigned a knowledge score, and we evaluated the results obtained by changing the number of "experts" so extracted to 5, 10, 15, or 20.

Figure 5-(a) shows the average results obtained for the ten concepts. These results confirm that the expanded taxonomies created by our method could extract 3% to 10% more "experts", those with know-who information, than was possible by using just the original taxonomies, and 3% to 8% more than was possible by using the expanded taxonomies created by the TF/iDF method. A key finding is that our method becomes relatively more accurate as "experts" set increases. This is because of the more extensive relevant and precise knowledge contained in the expanded taxonomies created by the proposed method.

We then measured the effectiveness of the method in determining developers with similar ranges of knowledge using the similarity measurement method proposed in [19], [20]. From the 111 developers in the mailing list, we manually selected five with wide ranging knowledge and five who had more restricted knowledge. The proposed method, the original taxonomy and TF/iDF methods were then used to identify sets of the 101 remaining developers who most closely matched the knowledge levels of the reference set. Three different expert developers then graded the actual similarity of the knowledge levels.

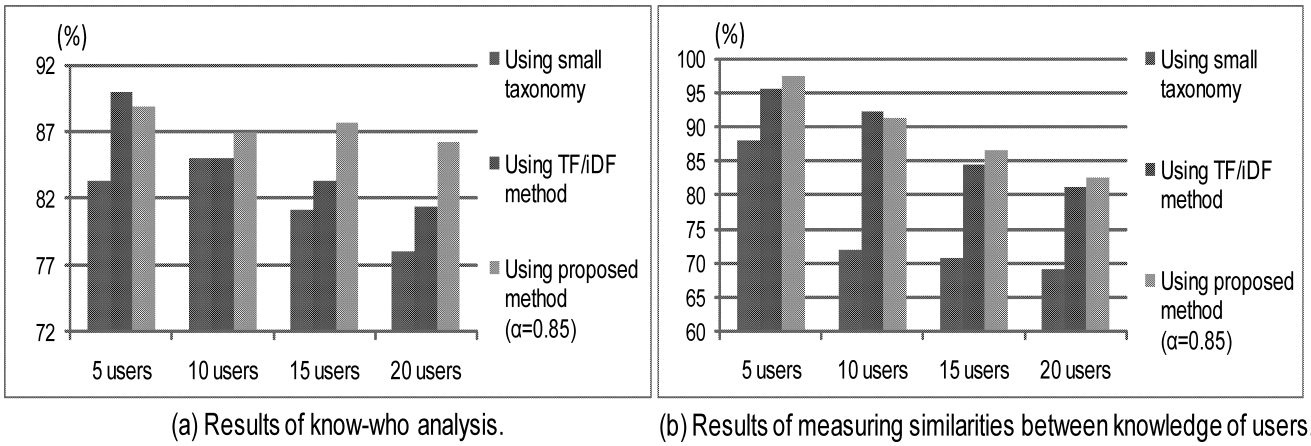


Fig. 5 Results of know-who analysis (X axis indicates the number of users and Y axis indicates accuracy of the results).

Table 2 Results of know-how analysis (MS means mail set).

	Top 3 MSs	Top 6 MSs
Concepts	0.571	0.571
Related concepts	0.750	0.646
Unified concepts 1	0.510	0.549
Unified concepts 2	0.722	0.639

The methods were challenged to identify groups of 5, 10, 15, and 20 matching developers.

The results in Fig. 5-(b) confirm that the expanded taxonomies created by our method extracted 10% to 20% more developers with similar knowledge to those in the target set than was possible with the original simple taxonomies. For three of the four cases examined, the proposed method was more accurate the expanded taxonomies created by TF/iDF method. Thus, we found that the relevant and precise knowledge contained in the expanded taxonomies created by the proposed method helps us to find more suitable know-who developers. Furthermore, it is important that we can assign detailed semantic tags between similar developers using expanded taxonomies as explained in Sect. 3.4.

4.4 Evaluating the Effect of Know-How Analysis

We then evaluated the effectiveness of our method in finding know-how by checking the classification of mail sets. We evaluated the results we obtained by having the three expert developers check three and six mail sets in order of their scores as classified in each concept. Five randomly selected concepts from the 49 product concepts were targeted. Here, the score of the mail set in a concept is computed by checking the frequency of description of instances in the concept.

Table 2 shows the results obtained. They confirm that the relevant concept mail sets contained more know-how information. This is convincing because detailed knowledge is stored in mail sets assigned to deeper concepts. The unified-concept mail sets contained very little know-how information, see "unified concepts 1" in Table 2. Checking the results in detail, we found that the low accuracy of the unified concept assignments originated from the concept "problem", the concept "test" yielded especially low values. This is because we were not able to remove classification mistakes caused by name attributes of topics that had several meanings, such as "problem" and "test" even though the filtering technique explained in Sect. 3.2.1 was applied. Manually removing the unified concepts of "problem" and "test" improved the accuracy as indicated in "unified concepts 2" in Table 2. To improve filtering performance, we may need to calculate the association between message topics and those in the taxonomies by checking whether both sets of name attributes occur in each e-mail, i.e. processing individual e-mails rather than sets. Individual e-mails have relatively narrow and/or stable topics, and so evaluating the co-occurrence of the instances in such topics achieves higher precision.

5. Conclusion

We combined expert-created concepts with data-driven concepts created by analyzing e-mail messages exchanged during a long-term project. Our method classifies phrases as data-driven instances to concepts in a taxonomy by considering associations between not only the phrases and the instances in taxonomy concept C, but also the instances in concepts near C (as defined by experts who created the taxonomy). Then, we attach the data-driven concept that has an extracted data-driven instance and instances of C to expert-created concept C in the taxonomy. Our proposed method was found to offer twice the accuracy of the TF/iDF method, which does not consider the neighbors of C. It also enables us to acquire more detailed know-who/know-how information than could be acquired by using only the expert-defined concepts themselves or using the expert-defined concepts with data-driven concepts created by the TF/iDF method.

We now apply our method to a knowledge management system in present system development projects [16]. In the future, we will analyze how know-who/know-how information varies among different development projects,

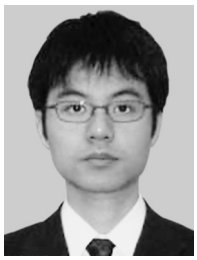
with the goal of creating common-use taxonomies relevant to tasks for several objective concepts. Through the use of these taxonomies, we will comprehensively analyze know-who/know-how information pertaining to the development of various products.

References

[1] E. Agirre, O. Ansa, E. Hovy, and D. Martinez, "Enriching very large ontologies using the WWW," Proc. Ontology Learning Workshop, 2000.
[2] E. Alfonseca and S. Manandhar, "Extending a lexical ontology by a combination of distributional semantics signatures," Proc. 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, EKAW '02, pp.1–7, Springer-Verlag, 2002.
[3] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "DBpedia: A nucleus for a Web of open data," International Semantic Web Conference/Asian Semantic Web Conference, ISWC/ASWC'07, pp.722–735, 2007.
[4] R. Bhagdev, A. Chakravarthy, S. Chapman, F. Ciravegna, and V. Lanfranchi, "Creating and using organisational semantic Webs in large networked organisations," International Semantic Web Conference, ISWC'08, pp.723–736, 2008.
[5] C. Biemann, "Ontology learning from text: A survey of methods," LDV Forum, vol.20, no.2, pp.75–93, 2005.
[6] D.M. Blei, A.Y. Ng, and M.I. Jordan, "Latent dirichlet allocation," J. Machine Learning Research, vol.3, pp.993–1022, 2003.
[7] C.H. Brooks and N. Montanez, "Improved annotation of the blogosphere via autotagging and hierarchical clustering," Proc. 15th International Conference on World Wide Web, WWW '06, ACM, pp.625–632, 2006.
[8] C. Chemudugunta, A. Holloway, P. Smyth, and M. Steyvers, "Modeling documents by combining semantic concepts with unsupervised statistical learning," International Semantic Web Conference, ISWC'08, pp.229–244, 2008.
[9] P. Cimiano, A. Hotho, and S. Staab, "Learning concept hierarchies from text corpora using formal concept analysis," J. Artif. Int. Res., vol.24, no.1, pp.305–339, 2005.
[10] A.M. Gliozzo, A. Gangemi, V. Presutti, E. Cardillo, E. Daga, A. Salvati, and G. Troiani, "A collaborative semantic Web layer to enhance legacy systems," International Semantic Web Conference/Asian Semantic Web Conference, ISWC/ASWC'07, pp.764–777, 2007.
[11] M.A. Hearst, "Automatic acquisition of hyponyms from large text corpora," Proc. 14th International Conference on Computational Linguistics, COLING'92, pp.539–545, 1992.
[12] A. Hotho, S. Staab, and G. Stumme, "Text clustering based on background knowledge," Technical Report 425, University of Karlsruhe, Institute AIFB, 76128 Karlsruhe, Germany, 2003.
[13] G. Ifrim, M. Theobald, and G. Weikum, "Learning word-to-concept mappings for automatic text classification," Proc. 22nd International Conference on Machine Learning - Learning in Web Search (LWS 2005), pp.18–26, 2005.
[14] N.J. Kings, C. Gale, and J. Davies, "Knowledge sharing on the semantic Web," European Semantic Web Conference, ESWC'07, pp.281–295, 2007.
[15] T.K. Landauer and S.T. Dumais, "A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge," Psychological Review, vol.104, pp.211–240, 1997.
[16] T. Madokoro, M. Nakatsuji, K. Okamoto, S. Miyazaki, and T. Harada, "Know-who/know-how navigation using development project-related taxonomies," International Conference on Semantic Computing, ICSC'09, pp.559–560, 2009.
[17] T.W. Malone, K. Crowston, and G.A. Herman, Organizing Business Knowledge, The MIT Process Handbook, MIT Press, Cambridge, MA, USA, 2003.
[18] R. Mizoguchi, J. Vanwelkenhuysen, and M. Ikeda, "Task ontology for reuse of problem solving knowledge," Knowledge Building & Knowledge Sharing 1995 (KB&KS'95), pp.46–59, 1995.
[19] M. Nakatsuji, Y. Miyoshi, and Y. Otsuka, "Innovation detection based on user-interest ontology of blog community," International Semantic Web Conference, ISWC'06, pp.515–528, 2006.
[20] M. Nakatsuji, M. Yoshida, and T. Ishida, "Detecting innovative topics based on user-interest ontology," J. Web Sem., vol.7, no.2, pp.107–120, 2009.
[21] J. O'Donovan and J. Dunnion, "Evaluating information filtering techniques in an adaptive recommender system," Adaptive Hypermedia and Adaptive Web-Based Systems, pp.312–315, 2004.
[22] S. Ponzetto and M. Strube, "Deriving a large scale taxonomy from Wikipedia," Proc. 22nd National Conference on Artificial Intelligence (AAAI-07), pp.1440–1447, 2007.
[23] E.M. Rasmussen, "Clustering algorithms," Information Retrieval: Data Structures & Algorithms, pp.419–442, 1992.
[24] M. Sanderson and B. Croft, "Deriving concept hierarchies from text," Proc. 22nd annual International ACM SIGIR Conference on Research and development in information retrieval, SIGIR '99, pp.206–213, ACM, 1999.
[25] S. Scerri, S. Handschuh, and S. Decker, "Semantic email as a communication medium for the social semantic desktop," European Semantic Web Conference, ESWC'08, vol.5021, pp.124–138, Springer, 2008.
[26] J.R. Tyler, D.M. Wilkinson, and B.A. Huberman, "E-mail as spectroscopy: Automated discovery of community structure within organizations," The Information Society, pp.143–153, 2005.
[27] C. Zirn, V. Nastase, and M. Strube, "Distinguishing between instances and classes in the Wikipedia taxonomy," European Semantic Web Conference, ESWC'08, pp.376–387, 2008.

Makoto Nakatsuji graduated in 2001 in Applied Mathematics at Kyoto University Faculty of Engineering. Completed Master's Degree in 2003 in Systems Science at Kyoto University Graduate School of Informatics. Currently working at the NTT Cyber Solution Laboratories. Interested in Web mining, semantic-based search systems, and context-aware ubiquitous networks research. Received the JSAI SIG Research Award in 2007. Member of The Japanese Society for Artificial Intelligence (JSAI) and The Database Society of Japan (DBSJ).

Akimichi Tanaka received the B.E. and M.E. degrees in precision machinery engineering from the University of Tokyo in 1985 and 1987, respectively. Currently working at the NTT Cyber Solutions Laboratories. Interested in artificial intelligence, pattern recognition, and educational system. Member of the Information Processing Society of Japan (IPSJ).



Takahiro Madokoro received the B.E./M.E. degree in electrical engineering and computer science from Nagoya University, Japan, in 2005 and 2007, respectively. Currently working at the NTT West R&D Center. Interested in semantic-based search systems.

Kenichiro Okamoto received the B.E./M.E. degree in electronic engineering from Ritsumeikan University, Japan, in 2003 and 2005, respectively. Currently working at the NTT West R&D Center. Interested in semantic-based search systems, and IPv6-based services.

Sumio Miyazaki received the B.E. and M.E. degrees in electronic engineering from Saga University, Japan, in 1998 and 2000 respectively. Currently working at the NTT West R&D Center. Interested in semantic-based search systems, and IPv6-based services. Member of Information Processing Society of Japan (IPSJ).

Tadasu Uchiyama received the B.S. and M.S. degrees in physics from Nagoya University in 1985 and 1987, respectively. Currently working at the NTT Cyber Solutions Laboratories. Interested in Web service and technology. Member of the Information Processing Society of Japan (IPSJ) and Japan Society for Industrial and Applied Mathematics (JSIAM).