2010 Asia Pacific Software Engineering Conference

Enhancing Domain Knowledge for Requirements Elicitation with Web Mining

Haruhiko Kaiya, Yuutarou Shimizu, Hirotaka Yasui, Kenji Kaijiri
Dept. of Computer Science
Shinshu University
Nagano, Japan
kaiya@acm.org, kaijiri@cs.shinshu-u.ac.jp

Motoshi Saeki
Dept. of Computer Science
Tokyo Institute of Technology
Tokyo, Japan
saeki@cs.titech.ac.jp

***Abstract*—To elicit software requirements, we have to have knowledge about a problem domain, e.g., healthcare, shopping or banking where the software is applied. A description of domain knowledge such as a domain ontology helps requirements analysts to elicit requirements completely and correctly to some extent even if they do not have such knowledge sufficiently. Several requirements elicitation methods and tools using domain knowledge description have been thus proposed, but how to develop and to enhance such description is rarely discussed. Summarizing existing documents related to the domain is one of the typical ways to develop such description, and an interview to domain experts is another typical way. However, requirements cannot be elicited completely only with such domain-specific knowledge because a user of such knowledge, i.e., a requirements analyst is not a domain expert in general. Requirements could be also elicited more correctly with both specific and general knowledge because general knowledge sometimes improves understandings of analysts about domain-specific knowledge. In this paper, we propose a method and a tool to enhance an ontology of domain knowledge for requirements elicitation by using Web mining. In our method and our tool, a domain ontology consists of concepts and their relationships. Our method and tool helps an analyst with a domain ontology to mine general concepts necessary for his requirements elicitation from documents on Web and to add such concepts to the ontology. We confirmed enhanced ontologies contribute to improving the completeness and correctness of elicited requirements through a comparative experiment.**

***Keywords*-Requirements Elicitation; Domain Knowledge; Ontology; Web Mining**

I. Introduction

Software systems are never used alone, but they used together with things in the real world, e.g., hardware, human, existing other systems and so on for resolving problems in the real world. Suppose requirements for a conference management system like EasyChair. A requirements analyst has to understand things such as submissions, review processes, notifications and so on if he has to develop the system. We call such a set of things related to similar problems as *a domain* in this paper. Software requirements analysts thus take such a domain into account when they define requirements for a software system that solves problems in the domain. To take them into account, the most effective way is to work with domain experts. However, they are so busy in general and it is hard to work together so frequently. In addition, such domain experts never work together earnestly if analysts have little knowledge about the domain. Therefore, software requirements analysts have to acquire domain knowledge as much as possible when they elicit and define requirements for a system that will resolve problems in the domain.

Referring explicit descriptions about domain knowledge is one of the ways for analysts to have domain knowledge, and a domain ontology is a typical description about such domain knowledge. Most domain ontologies are represented in a graph, which nodes are units of knowledge (called concepts) and which edges show the relationships among the units. Such a graph enables analysts to browse units of knowledge according to their relationships. Some ontologies provide inference rules on such a graph for analysts to have knowledge more than directly described on an ontology. Requirements elicitation methods and tools using a domain ontology are thus studied [5], [6], [7], [8] actively. However, how to acquire or to develop such a domain ontology is rarely discussed even though such methods and tools largely depend on the quality of a domain ontology.

There are typical ways to develop a domain ontology useful for requirements elicitation. A way is asking domain experts to develop a domain ontology. Another way is summarizing existing documents related to a domain. In both cases, a developed domain ontology is too specific to the domain, thus requirements analysts sometimes hardly understand the contents of the domain ontology. In addition, requirements analysts need such domain specific knowledge as well as knowledge that is used in a lot of domains, i.e., general knowledge when they elicit requirements for a system to solve some problems in the domain. For example about a conference management system, typical mechanism for collaboration on Web will help the analyst to understand a part of review processes. However, such general knowledge is rarely described in a domain ontology about conference management. In addition, such general knowledge is updated

1530-1362/10 $26.00 © 2010 IEEE
DOI 10.1109/APSEC.2010.11

rapidly independent of a specific domain, e.g., new collaboration ways such as Twitter. We thus have to enhance the quality especially availability of such a domain ontology based on domain independent resources with respect to the convenience of requirements analysts.

In this paper, we thus propose a method to enhance a domain ontology by adding general concepts to the ontology. In the example of a conference management system, our ontology enhancement method helps analysts to have knowledge about mechanisms that can be used or related to the system as well as a domain knowledge about the system. Such general concepts can be mined from Web pages, and the pages are usually up to date everyday. Because simple natural language techniques such as co-occurrences of terms and metrics of term occurrences can be used for this Web mining, most steps in the method can be performed automatically. We also developed a supporting tool called OREW to perform this method, and evaluate this method using OREW through a comparative experiment.

The rest of this paper is organized as follows. In the next section, we review related works to explain why ontology enhancement is important. In section III, we will briefly introduce a requirements elicitation method using a domain ontology. We then explain our ontology enhancement method in detail in section IV and its supporting tool called OREW in section V. We made a comparative experiment to evaluate the usefulness of our ontology enhancement method. The experiment is reported in section VI. Finally, we summarize our current results and show our future issues.

II. RELATED WORK

In this paper, the term "domain" is used as a set of things related to similar problems as mentioned in introduction. In the field of software product line development, identifying domain and reusing it is useful [1]. On the other hand, one of the most famous definitions of the term "ontology" is "formal explicit specification of shared conceptualization" [2]. Therefore, a domain ontology in this paper is a formal explicit specification of things related to similar problems.

In the field of software engineering, ontologies are widely used [3]. For example, a domain ontology is used for program comprehension [4]. Especially in requirements engineering, requirements elicitation methods and tools using a domain ontology are studied [5], [6], [7], [8]. Domain ontologies make up domain knowledge for requirements analysts. They sometimes help analysts to detect missing requirements and inconsistencies among requirements. Most of such methods and tools assume there is a domain ontology in high quality. However, it is not so easy to develop or acquire such a good domain ontology in practice. Therefore, we have to focus on how to develop or acquire such a good domain ontology.

There already exist several methods for developing ontologies in general [9], [10], [11]. Most of them use natural language processing (NLP) techniques. Some methods use existing dictionaries [12], and another uses Web Crowling [13], [14], [15]. There are also several tools to develop ontologies, such as Protege, OntoEdit, KAON, WebODE, TEXT-TO-ONTO. Some of them are compared in some literatures [16], [17]. An application specific ontology is developed based on an analysis of an application itself [18].

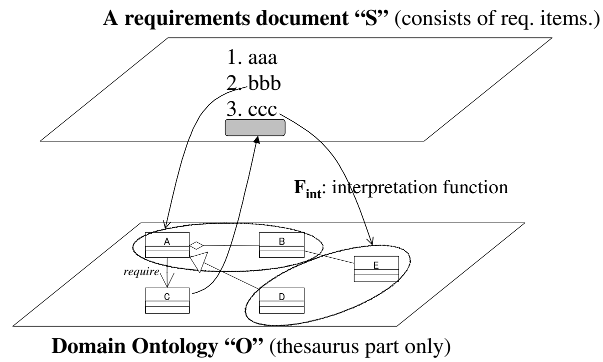

Figure 1. How to elicit requirements using a domain ontology in ORE

As mentioned in the former part in this section, requirements analysis and elicitation using a domain ontology is researched actively, but how to develop such a domain ontology is usually out of scope of such researches. A method and its tool called TCORE for developing a domain ontology for requirements elicitation is proposed [19]. In the method and its tool, an ontology is developed based on technical documents such as manuals and specifications about several similar systems. However, an analyst could not elicit around 40% of requirements with the help of such a domain ontology [20]. The analyst reported he elicited such 40% of requirements by using public resources mainly Web pages. Therefore, we have to make good use of such public resources for requirements elicitation as well as domain specific resources such as a domain ontology.

III. ORE: ONTOLOGY BASED REQUIREMENTS ELICITATION

In this section, we will explain one of requirements elicitation methods using a domain ontology called ORE (Ontology based Requirements Elicitation) [7] because of the following reasons. First, explaining a concrete elicitation method makes readers to easily understand and evaluate the ontology enhancement method proposed in this paper even though the enhancement method does not fully depend on the elicitation method ORE. Second, a supporting tool of the enhancement method in section V depends on the data structure of a ontology in ORE. Third, there is a CASE tool of ORE to support an analyst to elicit requirements.

Figure 1 shows how to elicit requirements using a domain ontology in ORE. In this figure, a graph like a class

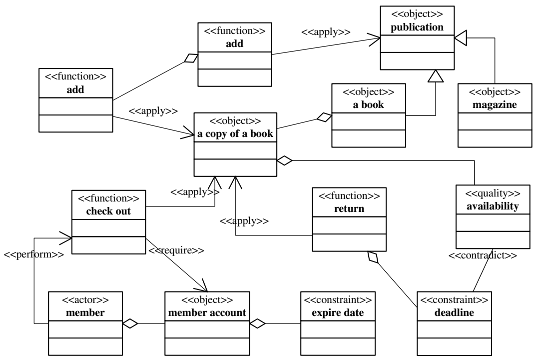

Figure 2. A part of an example of an ORE domain ontology

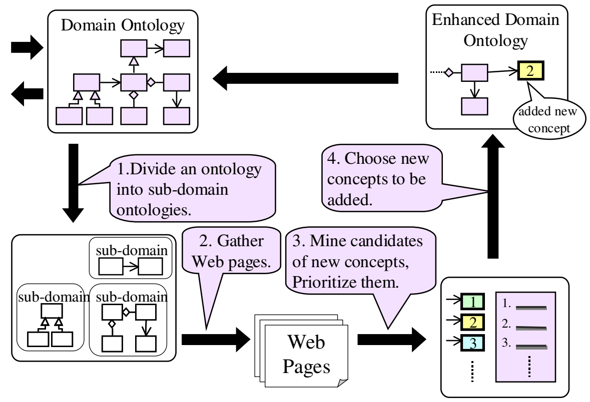

Figure 3. An Overview of Our Ontology Enhancement Method

diagram corresponds to a domain ontology, and rectangles like classes correspond to concepts in the ontology. A requirements analyst prepares an initial list of requirements ("aaa", "bbb" and "ccc" in the figure) from stakeholders for example. The analyst then makes relationship between each requirement and concepts in a domain ontology based on the terms in a requirement and the name or the label of a concept. In this figure, a requirement "bbb" is related to two concepts "A" and "B". The analyst finds a concept "C" that are related to "A" or "B" but that is not related to requirements. The analyst adds new a requirement based on a concept "C" if he decides the requirement is necessary. To iterate this cycle, requirements become more complete. Requirements could be more correct if a domain ontology contains some relationships such as potential contradiction among the concepts.

Figure 2 shows a concrete example of a domain ontology for a book management system in a library. As shown in the example, each concept and each relationship has its type. For example, two concepts "add" and "a copy of a book" and a relationship "apply" between them can correspond to a requirement "A copy of a book shall be added". Types in ORE are defined suitable for requirements definition, and used to define inference rules. Example of an inference rule is as follows.

- Conditions:
  - If an ≪object≫-type concept has several ≪apply≫-relations to ≪function≫-type concepts
  - and the ≪object≫-type concept and one of ≪function≫-type concepts correspond to an existing requirement
- Actions:
  - We may consider adding new requirement which corresponds to the pair of the ≪object≫-type concept and one of other ≪function≫-type concepts.

An example of applying this rule is as follows. When a requirement "A copy of a book shall be added" already exists and we have a domain ontology in Figure 2, requirements "A copy of a book shall be checked out" and "A copy of a book shall be returned" are candidates of requirements to be added according to the above rule.

Although an ontology in ORE does not follow formats of RDF and OWL, the ontology is based on components in a sentence such as subject, object and predicate in the same way as RDF. We thus regard that discussion about an ORE ontology can be applied to general ontology formats such as RDF and OWL.

IV. An Ontology Enhancement Method

A. Requirements for Ontology Enhancement

As discussed in introduction, some kinds of domain independent concepts, i.e., general concepts are also useful for requirements elicitation as well as domain-specific concepts. In the example of a conference management system, concepts about online-voting mechanism are useful as well as concepts about review processes. However, too general concepts are of course useless, thus we have to collect general concepts that are related to domain-specific concepts. In the example of a conference management system, most analysts do not need concepts about TCP (transmission control protocol) even if most on-line mechanisms are based on the protocol. It is not easy to collect such general concepts only from domain-specific documents or domain experts. We thus collect such general concepts from some other resources.

B. Overview of our Enhancement Method

To satisfy requirements in section IV-A, we propose an ontology enhancement method using Web mining because documents on Web have diverse kinds of concepts. Its overview is shown in Figure 3. The input of this method is a domain ontology that contains domain specific concepts. The ontology should not be empty, and should be high quality. The domain ontology is enhanced through four steps that are explained in the following sections in detail. The enhanced ontology may be an input of this method again. Basically, a requirements analyst performs this method before he elicits requirements. However, another analyst may perform this method for his colleague, or an analyst may use a domain ontology that has been already enhanced according to this method.

We only use occurrences of words or terms in a sentence, “a part of speech” of each word, i.e., the type of each word and the dependency structure among a sentence. In other words, we do not use complex techniques about natural language processing. This method can be thus applied to most natural languages. At least, this method can be applied to English and Japanese.

C. Step 1: Divide an ontology into Sub-domain Ontologies

Some concepts in an input ontology are not so domain-specific even if the ontology is generated based on domain-specific documents or domain experts. For example, some concepts about functions, qualities and constraints are related to several different domains. If such general concepts are used for Web mining, concepts that are not related to a domain can be also mined. We thus exclude such general concepts before Web mining. If the input ontology is based on ORE mentioned in section III, we exclude concepts which types are function, quality or constraint. We of course exclude concepts that are added by this method if we iteratively apply this method to an ontology.

Even after excluding general concepts from an input ontology, the ontology has a lot of concepts in general. Few candidates of added concepts are mined from Web if all the concepts are used together. In addition, several concepts in a domain ontology are strongly related with each other. For example, a domain about banking system can contain concepts about savings, concepts about loan and so on. We thus divide an ontology into so called sub-domain ontologies.

We use a metric called *betweenness centrality* [21] to do that because the metric can divide an ontology to sub-domain ontologies automatically. An ontology should be represented in graph structure when we use betweenness centrality. Each node in a graph has a value of betweenness centrality, and the value is large if a lot of routes on the graph contain the node. A route is a set of nodes and edges that are connected sequentially. A formal definition of betweenness centrality of a node $v \in V$ on a graph $G(V, E)$ is as follows.

$$BC(v) = \sum_{s,t \in V \ where \ s \neq t} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$$

Where $\sigma_{s,t}$ is the number of the minimal routes between s and t, and $\sigma_{s,t}(v)$ is the number of the minimal routes on a node v between s and t.

We choose concepts that have a larger value of betweenness centrality as central concepts for each sub-domain ontology. We may decide how many concepts can be central concepts at will. For each central concept, we choose other concepts that are contained in the sub-domain ontology based on the distance between the central concept and a concept. Some concepts can be contained in more than one sub-domain ontologies as the distance between a concept and a central concept is the same as the distance between the concept and another central concept.

D. Step 2: Gather Web Pages

For each sub-domain ontology, we mine Web pages using search engines such as Google. We basically use conjunction of all concepts as a query for mining Web pages. Because each concept usually has its name or caption, we use such a name or a caption for constructing a query. We set a target number of Web pages X and, gather more than X pages. Even if we divide an ontology into several sub-domain ontologies in step 1, we cannot sometimes mine enough number of documents because of too many concepts that are not related with each other in a sub-domain ontology. Even if we exclude useless concepts in step 1, some unnecessary documents can be mined. We overcome these problems using Simpson coefficient and NTF (Normalized Term Frequency).

To mine more than X pages, we exclude some concepts from the query for mining Web pages. Simpson coefficient [22] shows similarity between two terms. We thus calculate Simpson coefficient between a central concept and another concept respectively. We exclude a concept that has the smallest Simpson coefficient in turn. The formal definition of Simpson coefficient between a central concept cc and a concept c $S(cc, c)$ is as follows.

$$S(cc, c) = \frac{Hit(cc\&c)}{Min(Hit(cc), Hit(c))}$$

Where $Hit(q)$ returns the number of pages mined by a query q, and $Min(x, y)$ returns x if $x < y$, otherwise returns y.

To exclude unnecessary pages, we calculate NTF of each page and exclude pages which NTF is less than certain value t. We define NTF of a page as follows so that the value reflects how much the page is related to the sub-domain ontology.

$$NTF(sd, d) = \frac{\sum_{i=1}^{N} tf(c_i, d)}{Count(d)}$$

Where $Count(d)$ is the number of words in a page d, N is a number of concepts in the sub-domain ontology sd, c_i is ith concept in sd and $tf(c_i, d)$ is the frequency of c_i in d.

E. Step 3: Mine Candidates of New Concepts and Prioritize Them

Basically, terms in Web pages mined in the step 2 are candidates of new concepts to be added. However, too many unsuitable terms can be such candidates without some filtering. We thus focus on sentences in the pages. If a term and one of concepts in a sub-domain ontology do not occur together in any sentence, the term is excluded from the candidates of new concepts. If a term and a concept occur together in a sentence, we also analyze "a part of speech" of the term and the dependency structure between the term and the concept. "A part of speech" of the term is used for specifying the type of a concept if a concept is added based on the term. The dependency structure is used for specifying the type of a relationship between a concept and the concept to be added.

We then prioritize the candidates according to the following two metrics: *probability difference* (PD) and *term frequency* $\times$ *inverse document frequency* $(TF \times IDF)$. PD is based on co-occurrence between two terms in a document. We regard a set of sentences in all Web pages after filtering above as one document, and prioritize candidates of concepts with respect to each concept in a sub-domain ontology. By using $TF \times IDF$, we can exclude too general terms such as "the", "http" and so on. $TF \times IDF$ focuses on both a frequency of a term and distribution of the term over several documents. If a term frequently occurs but the term occurs in most of all documents, $TF \times IDF$ gets smaller value.

Suppose we want to decide whether a term t can be a candidate of a new concept added to a concept c using a document d that contains several sentences. PD can be formally defined as follows.

$$PD(c,t,d) = \frac{lines(d, \{c,t\})}{lines(d, \{c\})} - \frac{tf(d,t)}{line(d)}$$

Where $line(d)$ is the number of sentences in d, $lines(d,S)$ is the number of sentences in d that contains all terms in S, and $tf(d,t)$ is a term frequency of t in d.

Suppose we want to decide whether a term t frequently occurs but does not have wide distribution over a set of documents DS. $TF \times IDF$ can be formally defined as follows.

$$TF \times IDF(t, DS) = tf(DS,t) \times (log(\frac{|DS|}{df(DS,t)}) + 1)$$

Where $tf(DS,t)$ is term frequency of t in all documents in DS, $|DS|$ is the number of documents in DS and $df(DS,t)$ is the number of documents in DS that contains t.

F. Step 4: Choose new Concepts to be Added

Up to this step, candidates of new concepts added to each concept in a sub-domain ontology are gathered and prioritized almost automatically. We then actually add new concepts to an existing concept manually. For each existing concept, we can get the list of candidates prioritized by PD in step 3. We also know $TF \times IDF$ of each candidate. Based on these metrics and the contents of each candidate, we can add new concept to an existing concept. In the example of a conference management system, some analyst may need concepts about TCP as well as concepts about on-line discussion mechanism. We thus have to choose such candidates subjectively because which candidates are useful depends on the expertise of a requirements analyst. As mentioned in step 3, the relationship between the existing concept and the new concept is assumed based on the dependency structure in a sentence that contained both these concepts. A type of the new concept is assumed based on "a part of speech" of the concept. The concrete example will be shown in the next section.

V. OREW: An Ontology Enhancement Tool

To perform our ontology enhancement method in section IV, we have developed a supporting tool OREW (domain Ontology Reconstruction Environment by Web search). Because the method in section IV requires a domain ontology that is represented in a graph structure of concepts, OREW uses the data model of ORE requirements elicitation method in section III.

OREW performs most of all tasks in steps 1, 2, 3 in section IV almost automatically. In step 1, a user of OREW only has to give the number of sub-domain ontologies to OREW. In step 2, OREW uses "Yahoo! Web API"[1] for Web mining. In step 3, OREW automatically identifies sentences in Web pages based on punctuation, but the identification may sometimes be incorrect in general. The user thus may modify identified sentences manually.

In step 4, OREW provides GUI as shown in Figure 4 to make the user to choose new concepts to be added. In an example in Figure 4, an exiting concept "Conference Paper" is focused, and a new concept "Technical Paper" has been added. The user can browse the structure of an ontology as shown at the left side in the figure. At the right side in this figure, candidates of new concepts for the concept "Conference Paper" are listed as a tabular format, and three metrics, TF, TF×IDF and PD are shown in each line of the table. In this figure, the candidates are sorted based on the value of PD. The user may put the checks on the first column in the table, then the checked candidates are appearing at the left side of the figure like "Technical Paper". OREW automatically puts the type of added concept based on "a part of speech" of the candidate term, and the type of a relationship between an existing concept and an added concept based on the dependency structure in a sentence. The user of course may change such types manually.

VI. Evaluation

To evaluate the usefulness of our method and its tool OREW, we made an experiment in the following way.

A. Hypotheses

Concepts added to the enhanced ontology are expected to be general in the sense that they can be used to a lot of different domains. In addition, we expect an enhanced ontology is more helpful than an ontology based only on

[1] http://developer.yahoo.com/

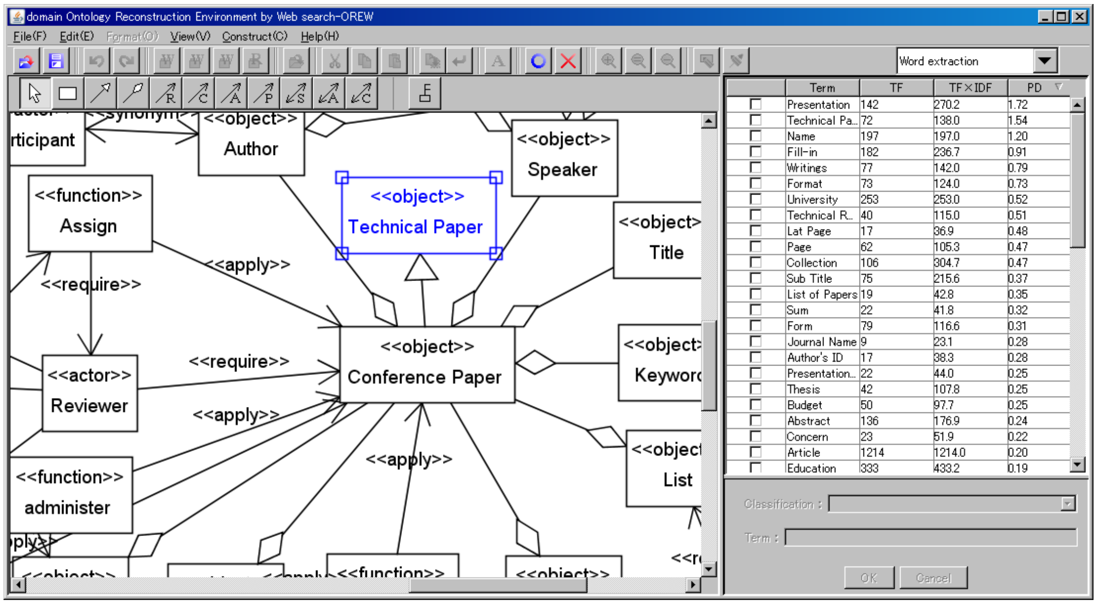

Figure 4. A Snapshot of OREW: Prioritizing Candidates of Concepts to be Added

domain specific documents or only on domain experts with respect to the quality of requirements elicitation.

According to the mechanism in our method and its tool, general concepts will be gathered and added to an existing ontology. We especially assume more concepts related to solutions [23] will be added to an existing ontology because solutions such as information technologies (IT) can be used in several different domains. We also assume added concepts could have some different characteristics from those of concepts in an initial ontology. We thus set up the following hypotheses.

> **H1**: Our method and its tool will add concepts to an ontology and more of those concepts will be general rather than specific.
>
> **H2**: Our method and its tool will add concepts to an ontology and more of those concepts will be about solutions rather than problems. If a concept is about solutions such as cryptography or IC tag, we regard the concept is solution. Otherwise, it is about problems.
>
> **H3**: According to problem frames [23], concepts can be categorized into causal, biddable or lexical. Our method and its tool will add concepts to an ontology and more of those concepts will be one of categories than others.

Completeness and correctness of a requirements specification are important quality factors [24]. We thus set up the following hypotheses.

> **H4**: A requirements specification becomes more complete when the specification is elicited with an enhanced ontology.
>
> **H5**: A requirements specification becomes more correct when the specification is elicited with an enhanced ontology.

B. Objects and Subjects

We have the following two initial ontologies as inputs of our tool respectively.

- $\mathbf{IO_{pos}}$: An ontology of POS (point of sales) systems. The ontology was generated by a tool [25] based on 14 documents. The lengths of the documents that we used were from 3 to 23 pages of A4 paper size.
- $\mathbf{IO_{conf}}$: An ontology of conference management system like cyberchair. The ontology was written by domain experts. They investigated actual systems when they wrote them.

For the sake of comparison between requirements elicitation using either $\mathrm{IO_{pos}}$ or $\mathrm{IO_{conf}}$, the size of each ontology was unified. Because it is hard for subjects to concentrate on requirements elicitation for a long time, we reduced the size of each ontology. The sizes of these ontologies are shown in section VI-E.

One of authors enhanced $\mathrm{IO_{pos}}$ and $\mathrm{IO_{conf}}$ respectively with our tool OREW. He took about four hours to enhance each ontology. We call an enhanced ontology based on $\mathrm{IO_{pos}}$ as $\mathbf{EO_{pos}}$, and an enhanced ontology based on $\mathrm{IO_{conf}}$ as $\mathbf{EO_{conf}}$. Note that an ontology $\mathrm{EO_d}$ ($\mathrm{EO_d}$ is a generic form of $\mathrm{EO_{pos}}$ and $\mathrm{EO_{conf}}$) contains all concepts and relationships in $\mathrm{IO_d}$.

We asked two subjects called $\mathbf{Subject_a}$ and $\mathbf{Subject_b}$ to elicit requirements as a requirements analyst using each ontology according to the method and its tool mentioned in

Table I
THE ORDER OF REQUIREMENTS ELICITATION FOR EACH SUBJECT

	$\mathrm{Subject_a}$	$\mathrm{Subject_a}$
First	IRLstock and $\mathrm{O_{stock}}$	
Second	IRLconf and $\mathrm{IO_{conf}}$	IRLconf and $\mathrm{EO_{conf}}$
Third	IRLpos and $\mathrm{EO_{pos}}$	IRLpos and $\mathrm{IO_{pos}}$

section III. Because both initial and enhanced ontologies were developed by not subjects but other people, there would be no learning effect on their requirements elicitation. According to the method and its tool, requirements are elicited based on the initial requirements and a domain ontology. Therefore, we prepared lists of initial requirements as follows.

- $\mathbf{IRL_{pos}}$: A list of initial requirements for the POS domain.
- $\mathbf{IRL_{conf}}$: A list of initial requirements for the conference management systems domain.

Each initial requirements list plays a role of stakeholders such as customers and users. The sizes of initial requirements lists will be shown in section VI-E. Most initial requirements were requirements not for the machine but for the real world in the sense that problem frames [23] because the method and the tool in section III mainly help an analyst to refine and decompose requirements into specifications.

Subjects were bachelor students in the last grade in software engineering course, thus they had already learned programming languages, software engineering fundamentals. There were no differences about the expertise about the domains, POS and conference management between the subjects.

C. Experimental Design

One of authors developed ontologies $\mathrm{EO_{pos}}$ and $\mathrm{EO_{conf}}$ based on $\mathrm{IO_{pos}}$ and $\mathrm{IO_{conf}}$ as mentioned in section VI-B. We also prepared another ontology $\mathrm{O_{stock}}$, which contained domain knowledge about stock trading systems, and an initial requirements list for the systems $\mathrm{IRL_{stock}}$. The ontology $\mathrm{O_{stock}}$ and $\mathrm{IRL_{stock}}$ was used as a learning material for the requirements elicitation method and the tool mentioned in section III.

For the hypotheses H4 and H5, we asked each subject to actually elicit requirements based on an initial requirements list and an ontology. We also allowed each subject to use ordinary Web mining such as Google or Yahoo search. To minimize the learning effects about requirements elicitation method, we asked subjects to perform requirements elicitation in the order shown in Table I. Because each subject becomes familiar with the elicitation method step by step, the third result will be better than the second result. We thus asked $\mathrm{Subject_a}$ to use IO second and EO third. On the other hand, we asked $\mathrm{Subject_b}$ to use EO second and IO third. Note that each subject uses $\mathrm{O_{stock}}$ first for learning the elicitation method. Because it is hard for subjects to concentrate on requirements elicitation for a long time, we asked each subject to elicit requirements with an ontology up to two hours.

D. Data Gathering and Measurement

For the hypotheses H1, H2 and H3, we focus on concepts contained in each ontology. We especially focus on an increasing rate of concepts categorized into a type, e.g., general type, specific type, problem related type or solution related type. To measure concepts and their types, we introduce the following functions.

- $\mathrm{Concept(X)}$: this function returns concepts contained in an ontology X.
- $\mathrm{Type(C, d, t)}$: this function returns a maximal set of concepts each of which type is t and the maximal set is a subset of C in a domain d. This function is used to categorize concepts into several types. We have to calculate the return values of this function subjectively because a type of a concept cannot be decided objectively in general.
- We of course use general set operators such as $|S|$ (the number of elements), $\bar{S}$ (complement set of S), $S - T$ $(= S \cap \bar{T})$ and so on.

We can then derive the following values for each domain d.

- $\mathrm{Concept(IO_d)}$: set of concepts contained in an initial ontology of a domain d.
- $\mathrm{Concept(EO_d)} - \mathrm{Concept(IO_d)}$: a set of concepts added by our method.
- $\frac{|\mathrm{Type(Concept(EO_d)-Concept(IO_d),d,t)}|}{|\mathrm{Type(Concept(IO_d),d,t)}|}$: an increasing rate of t-type concepts. We simply abbreviate this expression to $\mathbf{Gain(d, t)}$ because it is similar to the gain of electric amplifiers.

In this experiment, we focus on the following four different categorizations for concepts. Types in each categorization are exclusive with each other.

- general or specific:
If a concept can be used several different domains, we regard the concept is general. Otherwise, we regard the concept is specific. For example,
$$\mathrm{Type(\{"paper\ submission", "password"\}, "conference\ management", "general")} = \{\mathrm{"password"}\}$$
because "password" can be used in many different domains.
- solution or problem:
If a concept is about solutions such as cryptography or IC tag, we regard the concept is solution. Otherwise, it is about problems.
- causal, biddable or lexical:
These types come from problem frames. If a concept is about something that follows causalities, e.g., mechanical or electric parts, it is causal. If a concept is

unpredictable and autonomous such as human, it is biddable. If a concept is about data or information, it is lexical.

- any:
  We regard any concept has a type "any", thus this is a kind of special type. We use this type for identifying total gain of an ontology enhancement.

For hypotheses H4 and H5, we need both complete and correct list of requirements, i.e., a right requirements list. We prepared a right requirements list for each initial requirements list before this experiment. However, subjects can find correct requirements that we could not find before the experiment. Therefore, we define a right requirements list of a domain $\mathbf{RRL_d}$ as follows.

- $\mathrm{RRL_d}$ is the union of the following three lists of requirements: a right requirements list we prepared before this experiment, a list of requirements each of which is contained in requirements elicited by $\mathrm{Subject_a}$ and each of which seems to be correct, a list of requirements each of which is contained in requirements elicited by $\mathrm{Subject_b}$ and each of which seems to be correct. We exclude initial requirements from $\mathrm{RRL_d}$, i.e., $\mathrm{IRL_d} \cap \mathrm{RRL_d} = \phi$ so that we can easily define completeness and correctness below.

Although $\mathrm{RRL_d}$ depends on the initial requirements list, we do not focus on this factor in this experiment.

We have already defined $\mathrm{IRL_d}$ as an initial requirements list of a domain d, thus we define the following.

- $\mathrm{ERL_d(O_d)}$: an extended requirements list of the initial requirements list of a domain d during a requirements elicitation with an ontology $\mathrm{O_d}$ by a subject. We exclude initial requirements from $\mathrm{ERL_d(O_d)}$, i.e., $\mathrm{IRL_d} \cap \mathrm{ERL_d(O_d)} = \phi$ so that we can easily define completeness and correctness below.

Although $\mathrm{ERL_d(O_d)}$ depends on a subject performing requirements elicitation, we do not focus on this factor in this experiment. The reason is there is no difference between two subjects with respect to the expertise in domains and requirements elicitation.

We define the metrics for completeness and correctness as follows.

- Completeness: $\mathrm{Comp(O_d)} = \frac{|\mathrm{RRD_d} \cap \mathrm{ERL_d(O_d)}|}{|\mathrm{RRD_d}|}$
- Correctness: $\mathrm{Corr(O_d)} = \frac{|\mathrm{RRD_d} \cap \mathrm{ERL_d(O_d)}|}{|\mathrm{ERL_d(O_d)}|}$

These metrics almost correspond to recall and precision in information retrieval area.

Finally, we inquired of each subject whether two hours was enough to elicit requirements using each domain ontology. We observed whether each subject used ordinary Web search.

E. Results

Table II shows the sizes of initial and enhanced ontologies. As we mentioned in previous section, one of authors enhanced these initial ontologies according to our tool OREW. It took about 4 hours to enhance them. Enhanced ontologies were about twice as large as initial ontologies, thus each total gain is almost 100 %. For example, an initial ontology of POS systems ($\mathrm{IO_{pos}}$) had 43 concepts, and 51 concepts were added due to an enhanced ontology ($\mathrm{EO_{pos}}$) by our tool OREW. The gain of this ontology enhancement $\mathrm{Gain(pos, "any")}$ is 118.6 %.

Tables III, IV and V show the results of increasing rates of concepts caused by ontology enhancement, i.e., gains of an ontology enhancement. For example in Table III, $\mathrm{Gain(d, "general")}$ is 307.6%. This value means our ontology enhancement method and its tool added general concepts threefold as many as general concepts in the initial ontology of POS systems ($\mathrm{IO_{pos}}$).

Finally, Table VI shows the sizes of initial and extended requirements lists. Their completeness and correctness are also shown in the table. In the case of POS domain, an

Table II
THE SIZES OF ONTOLOGIES (NOTE: $|\mathrm{Concept(O_{stock})}|$ IS 69.)

d	pos	conf
$\lvert\mathrm{Concept(IO_d)}\rvert$	43	42
$\lvert\mathrm{Concept(EO_d)}\rvert$	94	98
$\lvert\mathrm{Concept(EO_d) - Concept(IO_d)}\rvert$	51	56
$\mathrm{Gain(d, "any")}$	118.6 %	133.3 %

Table III
INCREASING RATE OF GENERAL OR SPECIFIC CONCEPTS

d	pos		conf	
$\mathrm{Gain(d, "general")}$	307.6 %	(40/13)	163.1 %	(31/19)
$\mathrm{Gain(d, "specific")}$	36.6 %	(11/30)	108.6 %	(25/23)

Table IV
INCREASING RATE OF PROBLEM OR SOLUTION CONCEPTS

d	pos		conf	
$\mathrm{Gain(d, "solution")}$	133.3 %	(20/15)	212.5 %	(17/8)
$\mathrm{Gain(d, "problem")}$	110.7 %	(31/28)	114.7 %	(39/34)

Table V
INCREASING RATE OF CAUSAL, BIDDABLE OR LEXICAL CONCEPTS

d	pos		conf	
$\mathrm{Gain(d, "causal")}$	150.0 %	(12/8)	350.0 %	(21/6)
$\mathrm{Gain(d, "biddable")}$	215.3 %	(28/13)	100.0 %	(17/17)
$\mathrm{Gain(d, "lexical")}$	50.0 %	(11/22)	94.7 %	(18/19)

Table VI
REQUIREMENTS LISTS AND THEIR COMPLETENESS AND CORRECTNESS

d	pos		conf	
$\mathrm{O_d}$	$\mathrm{IO_{pos}}$	$\mathrm{EO_{pos}}$	$\mathrm{IO_{conf}}$	$\mathrm{EO_{conf}}$
$\lvert\mathrm{IRL_d}\rvert$	11		10	
$\lvert\mathrm{ERL_d(O_d)}\rvert$	26	58	33	65
$\lvert\mathrm{RRL_d}\rvert$	71		91	
$\lvert\mathrm{ERL_d(O_d) \cap RRL_d}\rvert$	18	50	29	59
$\mathrm{Comp(O_d)}$	24.4 %	70.4 %	31.9 %	64.8 %
$\mathrm{Corr(O_d)}$	69.2 %	86.2 %	87.9 %	90.8 %
2 hours enough?	Yes	No	Yes	No

initial requirements list ($\mathrm{IRL_{pos}}$) contained 11 requirements. When a subject performed requirements elicitation based on $\mathrm{IRL_{pos}}$ and an ontology ($\mathrm{IO_{pos}}$) that was not enhanced by OREW, he elicited 26 requirements as a result. We regarded 71 requirements were correct each of which was contained in $\mathrm{IRL_{pos}}$, $\mathrm{ERL_{pos}(IO_{pos})}$ or $\mathrm{ERL_{pos}(EO_{pos})}$. Therefore, completeness of $\mathrm{ERL_{pos}}$ was 24.4% and its correctness was 69.2%. Because we assume this $\mathrm{ERL_{pos}}$ depends on $\mathrm{IO_{pos}}$, we regard completeness caused by $\mathrm{IO_{pos}}$ was also 24.4%.

The last line in Table VI shows the results whether two hours was enough for each requirements elicitation. When a subject used an ontology $\mathrm{IO_d}$, two hours was enough for his elicitation. However, each subject used an enhanced ontology $\mathrm{EO_d}$ said he needed more time to elicit requirements but he tried to finish elicitation within about two hours. We observed subjects rarely used ordinary Web search during each elicitation.

F. Discussion

In hypotheses H1, H2 and H3, we are interested in the increasing rate of some type of concepts caused by ontology enhancement, i.e., gain of ontology enhancement. We thus focus on the gains in Tables II, III, IV and V. Although the total gains in Table II are almost 100 %, most gains in other tables are not 100 %. As shown in Table III, the gains of general concepts are larger than those of specific concepts and total concepts in Table II. We thus regard H1 is true. In the same way, H2 seems to be true due to the results in Table IV. Results in Table V show gains of causal concepts are relatively larger than a gain of lexical concepts. Our ontology enhancement method and the tool seem to mainly add concepts such as mechanical or electric parts or their functions. Because H3 is an exploratory question rather than a hypothesis, the result that the gain of causal concepts is relatively large is the answer to H3. Causal concepts are usually not specific to a domain but general, thus this result agrees on the goal of our ontology enhancement method and the tool.

In hypotheses H4 and H5, we are interested in the effects by ontology enhancement on the quality of elicited requirements. As shown in Table VI, the completeness by an enhanced ontology is almost twice as large as the completeness by an initial ontology in both domains. Therefore, H4 seems to be true. The results are not surprising because the number of concepts in an enhanced ontology is larger than those in an initial ontology. The correctness by an enhanced ontology is also larger than the correctness by an initial ontology as shown in Table VI. H5 also seems to be true. In general, correctness (precision) declines when the given information increases. The reason is incorrect or inadequate information could be usually contained in added information. However, both completeness and correctness rise in the results. We assume added concepts helped subjects to understand existing domain-specific concepts thus correctness also rises. Therefore, our ontology enhancement method and the tool seem to effective for requirements elicitation.

G. Threats to Validity

1) Internal Validity: Independent variables in this experiment are ontologies and requirements lists. For the requirements lists, we eliminated learning effects as shown in Table I. Although the requirements in both lists seem to be enough realistic and not to lean toward general concepts, there is no formal proofs to validate the fact. We did not eliminate leaning effects for enhancing ontologies because only one of authors enhanced them with our tool. However, this is not a threat to internal validity because we do not focus on comparison between two enhanced ontologies. We focus on comparison between an initial ontology and its enhanced ontology. Note that original initial ontologies were developed by different people, but one of authors reduced them for our experiment as mentioned in VI-B. Because one of authors enhanced ontologies, the enhanced ontologies could have bias.

2) External Validity: Because one of authors enhanced ontologies and the other people (not authors) elicited requirements using ontologies, our results are general in the sense that we eliminate learning effect by enhancing ontologies. Because our method for enhancing ontologies cannot be achieved fully automatically and only one author enhanced both ontologies with our tool, the results of ontology enhancement by another people can be different from our results. Because we use students as subjects for requirements elicitation, this might be a large threat to external validity.

3) Construct Validity: We directly measure variables for our hypothesis. However, we do not focus on relationships among concepts for H1, H2 and H3. For H4 and H5, we need a right requirements list $\mathrm{RRL_d}$ for each domain, thus we developed it as mentioned in section VI-D. Because both $\mathrm{RRL_d}$ and enhanced ontologies were developed by us, there could be some relationships between $\mathrm{RRL_d}$ and enhanced ontologies. As shown in the last line in Table VI, each subject using an enhanced ontology required more than two hours for his requirements elicitation. If we did not set the two hours limit for our experiment, more requirements can be elicited with an enhanced ontology. Therefore, completeness can increase but we cannot assume whether correctness decreases or not. We find these threats to construct validity.

4) Conclusion Validity: We do not achieve statistical test because of a few number of results. Therefore, threat to this validity remains.

VII. Conclusion

In this paper, we propose a method for enhancing a domain ontology for requirements elicitation. Requirements could be missing during requirements elicitation using a domain ontology if such a domain ontology is based only on domain-specific documents and/or domain experts. The goal of this method is thus adding new general and useful concepts to a domain ontology that contains mostly domain-specific knowledge. To gather such general and useful concepts, we use Web mining and lightweight natural language processing techniques in the method. We have also developed a supporting tool called OREW to perform this method. Through a comparative experiment, we confirmed our method enhances a domain ontology according to the goal of the method. We asked subjects to elicit requirements using either a domain ontology before enhancement and or a domain ontology after enhancement. As a result, the enhanced ontology was more useful than another with respect to both completeness and correctness of elicited requirements.

As mentioned in section IV, most steps can be performed automatically. However, the last step for choosing new concepts still requires some human efforts. Especially, we have to improve prediction about the type of new added concept and the type of a relationship between the new concept and an existing concept. Currently, we do not use general dictionaries nor ontologies such as WordNet [26] at all, thus we cannot identify synonyms in Web Pages automatically. We want to make use of such dictionaries or ontologies to improve the quality of Web mining.

REFERENCES

[1] K. Pohl, G. Bockle, and F. V. D. Linden, *Software Product Line Engineering: Foundations, Principles And Techniques*. Springer-Verlag New York Inc, 2005.

[2] T. R. Gruber, "A translation approach to portable ontologies," *Knowledge Acquisition*, vol. 5, no. 2, pp. 199–220, 1993.

[3] Y. Zhao, J. Dong, and T. Peng, "Ontology classification for semantic-web-based software engineering," *IEEE Transactions on Services Computing*, vol. 2, pp. 303–317, 2009.

[4] H. Zhou, et al, "Developing application specific ontology for program comprehension by combining domain ontology with code ontology," in *QSIC*, 2008, pp. 225–234.

[5] K. K. Breitman and J. C. S. do Prado Leite, "Ontology as a Requirements Engineering Product," in *(RE'03)*, Sep. 2003, pp. 309–319.

[6] S. W. Lee, et al, "Ontology-based active requirements engineering framework," in *APSEC*, 2005, pp. 481–490.

[7] H. Kaiya and M. Saeki, "Using Domain Ontology as Domain Knowledge for Requirements Elicitation," in *RE'06*. Sep. 2006, pp. 189–198.

[8] D. V. Dzung and A. Ohnishi, "Improvement of quality of software requirements with requirements ontology," *Quality Software, International Conference on*, pp. 284–289, 2009.

[9] J. S. Dong, Y. Feng, Y.-F. Li, and J. S. 0001, "A tools environment for developing and reasoning about ontologies," in *APSEC*, 2005, pp. 465–472.

[10] A. Zouaq, et al, "Evaluating the generation of domain ontologies in the knowledge puzzle project," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 11, pp. 1559–1572, 2009.

[11] M. Li and F. Zang, "A self-feedback methodology of domain ontology modeling," *Software Engineering, World Congress on*, vol. 2, pp. 218–223, 2009.

[12] X. Wang, et al., "Research on chinese domain ontology modeling based on automatic knowledge acquirement from multiple dictionaries," *Knowledge Acquisition and Modeling, International Symposium on*, pp. 360–366, 2009.

[13] H. P. Luong, et al., "Ontology learning through focused crawling and information extraction," *Knowledge and Systems Engineering, International Conference on*, pp. 106–112, 2009.

[14] V. C. Storey, R. H. L. Chiang, and G. L. Chen, "Ontology creation: Extraction of domain knowledge from web documents," in *ER*, 2005, pp. 256–269.

[15] K. Nakayama, et al., "Wikipedia mining for an association web thesaurus construction," in *WISE*, 2007, pp. 322–334.

[16] J. Durham, et al., "Enabling a common and consistent enterprise-wide terminology: An initial assessment of available tools," in *Web Intelligence*, 2008, pp. 544–548.

[17] A. Mikroyannidis and B. Theodoulidis, "Heraclitus ii: A framework for ontology management and evolution," in *Web Intelligence*, 2006, pp. 514–521.

[18] I. Hsi, C. Potts, and M. M. Moore, "Ontological excavation: Unearthing the core concepts of the application," in *WCRE*, 2003, pp. 345–352.

[19] M. Kitamura, R. Hasegawa, H. Kaiya, and M. Saeki, "An Integrated Tool For Supporting Ontology Driven Requirements Elicitation," in *ICSOFT 2007*, Jul. 2007, pp. 73–80.

[20] ——, "A Supporting Tool for Requirements Elicitation Using a Domain Ontology," in *Software and Data Technologies*, vol. 22. Springer, Nov. 2008, pp. 128–140, communications in Computer and Information Science (CCIS).

[21] K.-I. Goh, et al., "Betweenness centrality correlation in social networks," *Phys. Rev. E*, vol. 67, no. 1, p. 017101, Jan 2003.

[22] W. C. Fallaw, "A Test of the Simpson Coefficient and Other Binary Coefficients of Faunal Similarity," *Journal of Paleontology*, vol. 53, no. 4, pp. 1029–1034, 1979.

[23] M. Jackson, *Problem Frames, Analyzing and structuring software development problems*. Addison-Wesley, 2000.

[24] "IEEE Recommended Practice for Software Requirements Specifications," 1998, IEEE Std. 830-1998.

[25] R. Hasegawa, et al., "Extracting Conceptual Graphs from Japanese Documents for Software Requirements Modeling," in *(APCCM 2009)*, Jan. 2009, pp. 87–96.

[26] G. A. Miller, "Wordnet: A lexical database for english," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.