Computing and Informatics, Vol. 30, 2011, 857–879

A SEMANTIC APPROACH TO INTEGRATING XML SCHEMAS USING DOMAIN ONTOLOGIES

Haeran KANG, Kyong-Ho LEE*

Department of Computer Science
Yonsei University
134 Shinchon-Dong, Sudaemoon-ku
Seoul, 120-749 Korea
e-mail: khlee@cs.yonsei.ac.kr

Communicated by Jacek Kitowski

Abstract. XML documents might often conform to different schemas even in the same application domain. To support the interoperability among different IT systems, this paper proposes a sophisticated method for integrating XML schemas. The proposed method determines the synonym, hypernym, and holonym relationships among XML elements and attributes by using domain ontologies as well as general dictionaries. Specifically, the proposed method takes the structural information of elements and attributes into account. The conciseness of the schema integrated is also considered. Experimental results with a variety of schemas show that the utilization of a domain ontology and the structural information improved the performance of schema integration.

Keywords: XML schema, schema integration, semantics, domain ontology, databases

Mathematics Subject Classification 2000: 68N01, 68U35, 68T35

1 INTRODUCTION

Since XML [1] represents the logical structure of data and documents, it is widely used as a standard technology for exchanging and sharing information among busi-

* corresponding author

ness-to-business applications on the Internet. When users query an XML document, its structure would first be interpreted through its XML schema. However, since there are a number of XML schemas even in the same application domain, the extraction of information from these polymorphic XML schemas entails repeated queries. This process not only increases search space and time complexity but also degrades the quality of search results. Thus, to satisfactorily manage the schema heterogeneity, an efficient integration method of XML schemas is necessary.

To integrate schemas particularly in the same application domain, it would be effective to utilize domain-specific knowledge such as a domain ontology [2]. Most of the previous methods concerning schema integration, except the method of Huma et al. [3] use general dictionaries without considering such domain-specific information.

This paper presents a sophisticated method for integrating XML schemas in the same or similar application domain based on a domain ontology. The proposed method uses a domain ontology as well as a domain-independent general dictionary. Moreover, to extract relationships between lexical terms such as synonym, hypernym (the relation of class to subclass), and holonym (the relation of whole to part), the proposed method proposes the concept of the structural weight of elements and attributes in an XML schema. The proposed method also devises more sophisticated rules to integrate and optimize compositors.[1]

Generally, XML schema integration can be classified into two types. The first type converts XML schemas to canonical models, which abstract the structural heterogeneity of XML schemas, and integrates the models. The second type integrates XML schemas without the model conversion. While the first type requires a conversion process and mapping between models, the complexity of integration after the conversion is reduced. The second one does not require the model conversion, but requires handling the complexity inherent to the integration of XML schemas [4]. The first type may have the drawback that a canonical model does not include all the specifics of XML schemas, resulting in the loss of information from original sources. To preserve the meaning of XML source schemas, the proposed method follows the second type and does not convert XML schemas to models.

If human intervention was needed, the time complexity of schema integration would increase sharply. However, most of the previous methods need users' intervention to extract a global schema, resulting in a bottleneck during the integration. To minimize human intervention and maximize accuracy, the proposed method utilizes as much domain knowledge such as domain ontologies as possible. Experimental results show that a domain ontology and the proposed method of identifying the synonym, hypernym, and holonym relationships improve the precision and recall of schema integration, and the conciseness of the schema integrated. The rules devised to optimize compositors also decrease the size of the global schema integrated.

The organization of this paper is as follows: Section 2 briefly discusses related work. Section 3 describes the proposed integration method in detail. Section 4

[1] In this paper, compositors include operators of sequence, choice, and all.

shows the experimental results. Finally, Section 5 summarizes conclusions and future works.

2 RELATED WORK

Previous methods for integrating XML schemas can be classified by whether the methods adopt a conceptual model, use a domain ontology, or involve human intervention. Particularly, in this section, our discussion focuses on the works that utilize a domain ontology. Table 1 presents a brief survey about the characteristics and constraints of previous methods.

Huma et al. [3] substitute element labels with terms in a global ontology, resulting in the construction of local ontologies. To match two concepts, they calculate the similarities of labels and structures. Mello and Heuser [4] convert XML schemas into the Object with Roles Model/Natural language Information Analysis Method (ORM/NIAM) models and integrate the models. They do not consider all the features of XML schemas such as a sequence compositor.

Islam et al. [5] propose a name-based element-level schema matching method. While the method uses a single property, i.e., element name, for schema matching, it achieves high performance that is comparable to the conventional methods that use multiple properties such as element name, text description, data instance, and context description.

Do and Rahm [6] propose a context-dependent matching method, which handles schemas with shared elements and also scales to very large schemas. To support user interaction and to improve the performance of matching large schemas, the method is based on a fragment-based approach. Following the divide-and-conquer idea, it decomposes a large matching problem into smaller sub-problems at the level of schema fragments.

Meo et al. [7] try to determine the meaning of elements or attributes by examining their neighborhoods, whose concepts are semantically similar. They integrate synonymous elements and change the labels of homonymous elements. A global schema is constructed by removing redundancy and ambiguity. Users can select a severity level, against which the integration task is performed. The examination of all the neighbors of elements and attributes may result in a significant overhead.

Cruz et al. [8] convert XML schemas into Resource Description Framework Schema (RDFS) ontologies, resulting in the information loss of sources. They have to manually construct mappings between a global ontology and local ontologies. Jeong and Hsu [9] extract an integrated schema automatically. They utilize a renaming module, to which users can give additional guidelines. Yang et al. [10] resolve structural conflicts and remove redundant object classes and transitive relationships based on data semantics.

To remove human intervention and increase accuracy, we propose an XML schema integration algorithm based on domain knowledge. For the concise and precise integration of XML schemas, the proposed method utilizes the structural

Authors	Year	Features	Domain ontology
Islam et al. [5]	2008	Calculate similarity between two target words by two corpus-based methods and map schemas by name-based element-level matching	X*
Do and Rahm [6]	2007	Match schemas using a flexible infrastructure to combine a library of matchers	X
Meo et al. [7]	2006	Integrate two XML schemas using inter-schema properties and examine neighborhoods to determine the meaning of two concepts	X
Huma et al. [3]	2005	Use different schema integration approaches (rule-based, learner-based, ontology-based, and wrapper-based)	O
Mello and Heuser [4]	2005	Transform XML schemas to conceptual schema ORM/NIAMs and use XML Path Language (XPath) for mapping between input schemas and an integrated schema	X
Cruz et al. [8]	2004	Model XML schemas to RDFS ontologies and obtain a global ontology based on the similarity of local ontologies semi-automatically	X
Jeong and Hsu [9]	2003	Model XML schemas to DTD trees, extract grammars from them, and integrate similar states and relationships	X
Yang et al. [10]	2003	Model XML schemas to the Object-Relationship-Attribute Model for Semi-Structured Data (ORA-SS) schema diagrams	X

* Hereafter in this paper, X indicates "No" and O means "Yes".

Table 1. A brief survey on schema integration methods

information of schemas, and devises sophisticated rules to optimize the schemas integrated.

3 THE PROPOSED SCHEMA INTEGRATION METHOD

As shown in Figure 1, our method consists of three steps: determination of the synonym, hypernym, and holonym relationships, integration, and optimization. The proposed method does not consider XML document instances but targets XML schemas.

The proposed method adopts a schema tree [11] for the efficient representation of XML schemas. The nodes in a schema tree are classified into general nodes

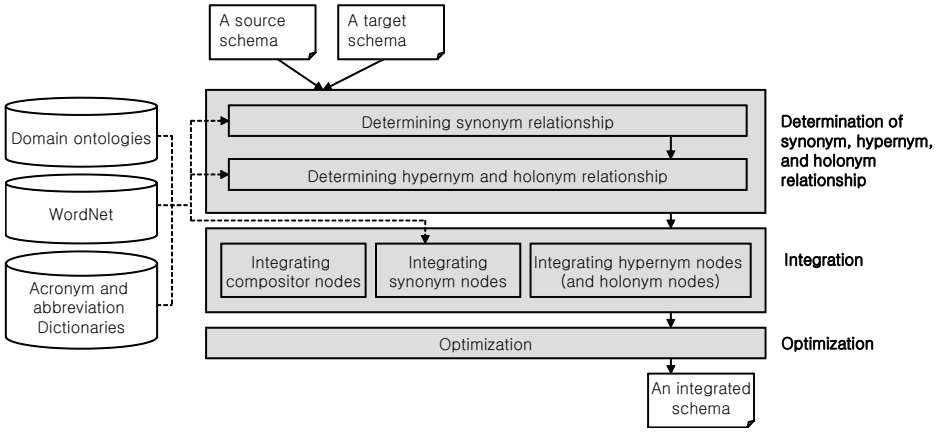


Fig. 1. XML schema integration process

(element and attribute nodes) and compositor nodes. General nodes have labels. Leaf elements and attributes in a schema correspond to leaf nodes, which have data types as their properties. If one general node has the other general node as its direct child node, the child node is an attribute node.

Elements with sub-structures are defined as internal nodes in a schema tree. Compositor nodes are classified as three types: choice, sequence, or all, and are labelled with '|', ',', or '&', respectively. An occurrence constraint as a node property has a '?', '*', '+', or a (minimum value, maximum value) label. If an occurrence range is $(0, 1)$, its label can be omitted.

The proposed method integrates input schemas starting from two synonym nodes. If there is no synonym relation between root nodes, the synonym nodes to be integrated first are found by the method of Deen and Ponnamperuma [12].

3.1 Determination of the Synonym, Hypernym and Holonym Relationships

The proposed method searches the synonym, hypernym, and holonym relationships among elements and attributes using a domain ontology, WordNet [13], and an acronym/abbreviation dictionary. Domain ontologies are written in the Web Ontology Language (OWL) [14, 15]. A synonym, hypernym, or holonym relationship may exist among the labels of general nodes. The proposed method searches for a synonym, hypernym, or holonym relationship in input schemas starting from root nodes by breadth-first search. The proposed method first searches for a synonym relationship between labels. If there is no synonym relationship, a hypernym or holonym relationship is considered.

3.1.1 Determining Synonym Relationship

The similarity between labels L_s and L_t is calculated by LabelSimilarity(L_s, L_t) as defined in Equation (1). If the similarity rate is greater than threshold TH_{sim}, L_s and L_t are considered to have a synonym relationship. The proposed method first tokenizes the labels based on capital letters or special symbols, and finds the tokens with non-zero structural weights.

$$\text{LabelSimilarity}(L_s, L_t) = \sum_{i=1}^{n} \left(\sum_{j=1}^{m} (\text{TokenSimilarity}(\text{T}_{\text{si}}, \text{T}_{\text{tj}})) \right) / n \times m \qquad (1)$$

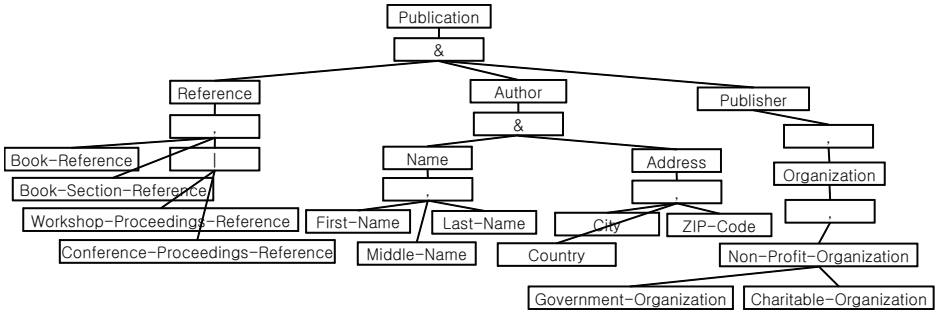
where n is the number of tokens with a non-zero structural weight in L_s, m is the number of tokens with a non-zero structural weight in L_t, T_{si} is the i^{th} token in L_s, $1 \leq i \leq n$, and T_{tj} is the j^{th} token in L_t, $1 \leq j \leq m$.

Figure 2 shows two schema trees, WordNet, and a domain ontology. In particular, Figure 2 c) and Figure 2 d) illustrate synonym, hypernym, and holonym relationships between concepts. Figure 2 d) illustrates only part of a domain ontology, which shows information necessary to integrate schemas in Figure 2 a) and Figure 2 b).

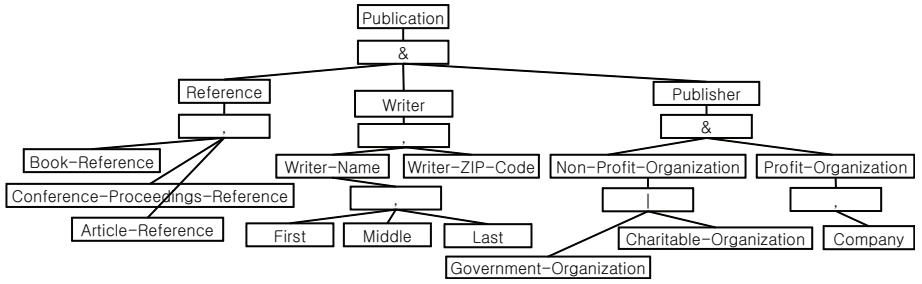
A schema includes hierarchical structure as well as the lexical names of elements (or attributes). The implication of the hierarchical structure of a schema should be considered. Due to the background knowledge of schemas, the names of elements (or attributes) may have a removable term unlike a domain ontology or dictionary. For example, if node First-Name has its closest ancestor node Writer-Name, the term Name is removable in node First-Name as shown in Figure 2 b) as its closest ancestor node Writer-Name provides background knowledge. In contrast, term name is not removable in word firstname in a domain ontology or dictionary. The proposed method uses this removable term to integrate schemas.

In the case where a term in a node is identical or similar to a term in its closest ancestor element node, the term in a node is generally removable. Based on this feature of schema structure, the proposed method defines the concept of a structural weight to calculate the structural importance of a token in a label. The structural weight of a token is determined through its relationship with tokens in the closest ancestor element node.

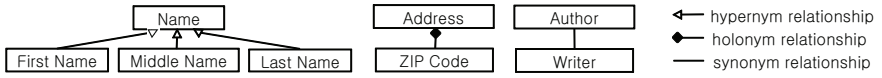
A node and its closest ancestor element node may have a hypernym ("is-a") relationship or holonym ("a part of" or "a member of") relationship. In this case, the meaning of a node is generally delimited by the meaning of its closest ancestor element node. Therefore, if token T_{des} in node N_{des} has a synonym relation with token T_{anc} in the closest ancestor element node N_{anc}, the meaning of N_{des} does not change even if T_{dec} is omitted. Thus, if a token is a synonym of a token in the closest ancestor element node, the proposed method considers its structural weight as having the value of zero. Otherwise, the structural weight of the token comes to 1. This paper assumes that the determination of the relationship between nodes should not be affected by tokens with the structural weight of zero.



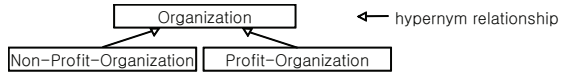
a) A source schema tree



b) A target schema tree



c) WordNet



d) A domain ontology

Fig. 2. An example of source and target schema trees and a domain ontology

In the case where a node has an identical or similar term in its closest descendant element node (or closest descendant attribute node), the term in a node is not removable because the meaning of a node is not delimited by the meaning of its closest descendant element node (or closest descendant attribute node). Likewise, if a node has an identical or similar term with its sibling element node (or sibling attribute node), the term in a node is not removable. If a node has a hypernym or holonym term in its closest ancestor element, closest descendant element node (or closest descendant attribute node), or sibling element node (or sibling attribute node) in a schema, the term in a node is not removable.

Unlike the conventional string similarity methods [7, 16], the proposed method calculates label similarity considering the domain-specific knowledge and structural information of XML schemas while not increasing the time complexity sharply. In other words, the proposed string similarity method utilizes domain ontologies and the proposed structural weight.

After finding tokens with non-zero structural weights, the method determines the similarity between tokens, TokenSimilarity. If TokenSimilarity between the two tokens is greater than threshold TH_{tokensim}, they are considered to have a synonym relationship.

To calculate TokenSimilarity between two tokens, the proposed method first examines whether each token is an acronym or abbreviation referencing an acronym and abbreviation dictionary. If each token is an acronym or abbreviation in the dictionary, the proposed method substitutes the token with the corresponding whole word. Then, if two tokens are identical, the TokenSimilarity of the tokens comes to 1. If there is an equivalentClass (or sameAs) relationship between tokens in a domain ontology, the two tokens have TokenSimilarity of 1. Otherwise, if one token is a synonym of the other token in WordNet, their TokenSimilarity comes a number between 0 and 1.

The process to determine a synonym relationship between label First-Name in Figure 2 a) and label First in Figure 2 b) is explained as follows. In Figure 2 a), label First-Name's closest ancestor has token Name. So, the two tokens First and Name in First-Name have 1 and zero as structural weights, respectively. Likewise, token First in label First is not a synonym of the Name token in the closest ancestor Name. Thus, token First in label First has the structural weight of 1. Because label First-Name and label First have the tokens with non-zero structural weights and they have a synonym relationship, the two nodes have a synonym relationship.

3.1.2 Determining Hypernym and Holonym Relationship

This section explains how to determine a hypernym and holonym relationship. First, the proposed method examines if a hypernym or holonym relationship exists between labels through WordNet or a domain ontology. If not, it examines whether the two labels satisfy the proposed conditions of a hypernym or holonym relationship as shown in Figure 3.

The proposed method checks whether a hypernym relationship between two tokens exists through a domain ontology as follows. If the ontological concept that token T_t belongs to is subsumed by the concept that contains token T_s in a domain ontology, T_s is identified as a hypernym of T_t. Generally, a holonym relationship between two tokens is not directly described in an OWL domain ontology. Thus, the proposed method extracts holonym relationships only through WordNet.

For instance, the proposed method determines a holonym relationship between label Address in Figure 2 a) and label Writer-ZIP-Code in Figure 2 b) as the following process. Token Address is a holonym of tokens ZIP-Code in Writer-ZIP-Code as

Given two labels L_s and L_t, after deleting articles in the labels, if they satisfy all the conditions $1 \sim 3$, L_s is assumed to be a hypernym (or holonym) of L_t.

1. A token in L_s is a hypernym (or holonym) of a token in L_t.
2. Every token with the structural weight of 1 in L_s is a hypernym (or holonym) or synonym of a token in L_t.
3. Every token with the structural weight of 1 in L_t is a hyponym (or meronym) or synonym of a token in L_s.

Fig. 3. The conditions of a hypernym or holonym relationship

described in Figure 2 c). Token Writer has the structural weight of zero because its closest ancestor element's label is Writer. So, label Address comes to a holonym of label Writer-ZIP-Code.

3.2 Integration

The proposed method integrates nodes in two schemas based on the relationships determined in Section 3.1 by breadth-first search. Specifically, the proposed integration method consists of integrating compositor nodes, synonym nodes, hypernym nodes, and holonym nodes.

3.2.1 Integrating Compositor Nodes

One or more compositor nodes may exist between the current element node and its closest descendant element node. They are defined as the lower compositor set (LCS) of the current element node and as the upper compositor set (UCS) of the closest descendant element node (or the closest descendant attribute node). In Figure 2 a), for instance, the LCS of node Publication consists of an all node and the UCS of node Reference also consists of the all node.

The proposed method integrates the following compositor sets: the LCSs of synonym nodes, and the LCS of a hypernym (or holonym) node and the UCS of a hyponym (or meronym) node. This paper defines compositor integration rules in the following three cases where each compositor set consists of one compositor; where one compositor set is empty and the other compositor set consists of at least one compositor; and where both compositor sets consist of at least one compositor and at least one compositor set consists of more than one compositor.

Case 1: The case where each compositor set consists of one compositor

This case is handled by the compositor integration rules of Table 2. The proposed method also modifies occurrence constraints of the closest descendant

nodes of compositor nodes adequately to preserve the meaning of compositors in input schemas.

In the case of integration of an all node and a choice node, the compositor set integrated might result in consisting of a choice (or sequence) node and an all node as its child node. This indicates that a choice (or sequence) node does not directly include an all node, but includes a group, which includes an all node.

For example, the LCS (sequence) of Publisher in Figure 2 a) and the LCS (all) of Publisher in Figure 2 b) are integrated as follows. Only child node of sequence is an element and its maximum occurrence constraint has the value of 1 as they are omitted in a schema tree. Thus, according to the rules in Table 2, the integrated LCS of Publisher consists of an all as shown in Figure 4. The LCS (sequence) of Organization in Figure 2 a) and the UCS (all) of Non-Profit-Organization in Figure 2 b) are integrated in the same way. Thus, the integrated LCS of Organization consists of an all.

Case 2: The case where one compositor set is empty and the other compositor set consists of at least one compositor

In this case, the two compositor sets are integrated by the compositor integration rules in Table 3. In this case, if a compositor set consists of more than one compositor, the highest depth compositor becomes only one compositor to be integrated by the rules in Table 3, and the sub-structure of the highest depth compositor becomes the sub-structure of the integrated compositor. The highest depth compositor in a compositor set indicates the compositor which is the highest among all compositors of the compositor set in a schema tree. For instance, sequence is the highest depth compositor of the LCS of Reference in Figure 2 a).

For instance, the LCS of node Non-Profit-Organization in Figure 2 a) and the LCS of node Non-Profit-Organization in Figure 2 b) are integrated as shown in Figure 5 as follows. Because the compositor set in Figure 2 a) is empty and the compositor set in Figure 2 b) consists of a choice node, the two compositor sets are integrated by the rules in Table 3. Two child nodes of the compositor set in Figure 2 a) are the synonym nodes of two child nodes of the compositor set in Figure 2 b). In addition, all the child nodes in the compositor set in Figure 2 b) are element nodes, and the maximum occurrence constraints come to 1. Therefore, the integrated compositor set consists of an all. In this case, minimum occurrence constraints of the child nodes of the all are set to zero to preserve the meaning of choice in an input schema.

Case 3: The case where both compositor sets consist of at least one compositor and at least one compositor set consists of more than one compositor

In this case, the two compositor sets are integrated from inside to outside. In other words, the two compositor sets are integrated from the lowest depth compositors to the highest depth compositors by the rules in Table 2.

Source node	Target node	Integration condition	Integrated compositor set
all	all		all
choice	choice		choice
sequence	sequence	If no child node of the source compositor node has a synonym, hypernym, or holonym relationship with any child node of the target compositor node	sequence and its child nodes, a source sequence and target sequence (The child nodes of the source and target compositor node become the child nodes of the integrated sequences which were the source and target compositor, respectively.)
		If the above condition is not satisfied	sequence (the longest common subsequence approach ([17]))
all	choice	If all child nodes of the target compositor node are elements and have maximum occurrence constraints smaller than 2	all
		If the above condition is not satisfied	choice and its child node, all (The child nodes of the source and target compositor node, which have synonym, hypernym, or holonym relationships between them, become the child nodes of the integrated choice node. Also, the child nodes of the source and target compositor node, which do not have a synonym, hypernym, or holonym relationship between them, become the child nodes of the integrated all and choice node, respectively.)
all	sequence	If all child nodes of the target compositor node are elements and have maximum occurrence constraints smaller than 2	all
		If the above condition is not satisfied	choice and its child nodes, all and sequence (The child nodes of the source and target compositor node become the child nodes of the integrated all and sequence node, respectively.)
sequence	choice		sequence and its child node, choice (The child nodes of the source and target compositor node, which have synonym, hypernym, or holonym relationships between them, become the child nodes of the integrated sequence node. Also, the child nodes of the source and target compositor node, which do not have a synonym, hypernym, or holonym relationship between them, become the child nodes of the integrated sequence and choice node, respectively.)

Table 2. The compositor integration rules in the case where each compositor set consists of one compositor

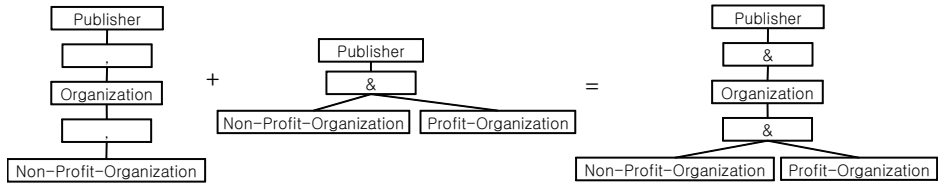


Fig. 4. An example of compositor integration in the case where each compositor set consists of one compositor

The highest depth source compositor	Integration condition	Integrated compositor
all	If all the child nodes of the empty set have occurrence constraints smaller than 2	all
	Otherwise	not integrated
choice	If there is at most one synonym, hypernym, or holonym relationship between the child nodes of the source and target compositor set	choice
	If there are more than one synonym, hypernym, or holonym relationship between the child nodes of the source and target compositor set, and all the child nodes of the source compositor set are element nodes and have occurrence constraints smaller than 2	all
	Otherwise	not integrated
sequence	If there is at most one synonym, hypernym, or holonym relationship between the child nodes of the source and target compositor set	sequence
	If there are more than one synonym, hypernym, or holonym relationship between the child nodes of the source and target compositor set, and all the child nodes of the source compositor set are element nodes and have occurrence constraints smaller than 2	all
	Otherwise	not integrated

Table 3. The compositor integration rules in the case where one compositor set is empty and the other compositor set consists of at least one compositor

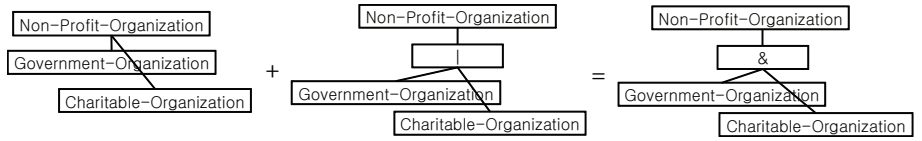


Fig. 5. An example of compositor integration in the case where one compositor set is empty and the other compositor set consists of at least one compositor

For example, the LCS of Reference in Figure 2 a) and the LCS of Reference in Figure 2 b) are integrated by the following process as shown in Figure 6. The LCS of Reference in Figure 2 a) consists of a sequence and a choice and the LCS of Reference in Figure 2 b) consists of a sequence. First, the choice in Figure 2 a) and the sequence in Figure 2 b) are integrated by the rule in Table 2. The integrated compositor set consists of a sequence and a choice. In addition, the child nodes of this sequence are Book-Reference, a choice, Conference-Proceedings-Reference, and Article-Reference. This choice has only one child node Workshop-Proceedings-Reference.

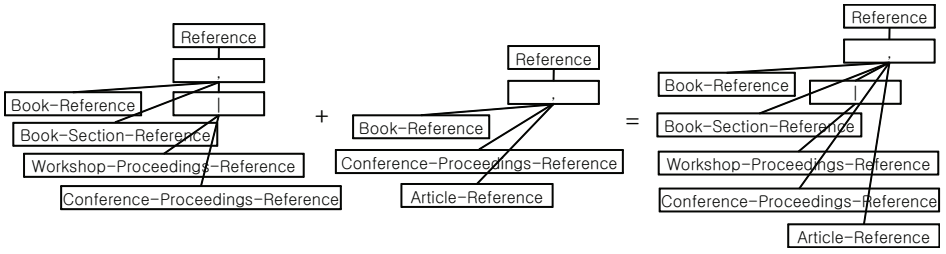


Fig. 6. An example of compositor integration in the case where both compositor sets consist of at least one compositor and at least one compositor set consists of more than one compositor

Next, the sequence in this integrated compositor set and the sequence in Figure 2 a) are integrated by the rule in Table 2. The sequence in this integrated compositor set has a choice as a child node but the sequence in Figure 2 a) does not have a child compositor because it is already integrated. Thus, the final integrated compositor set consists of a sequence and a choice. The child nodes of this sequence are Book-Reference, Book-Section-Reference, a choice, Conference-Proceedings-Reference, and Article-Reference. This choice has only one child node Workshop-Proceedings-Reference.

3.2.2 Integrating Synonym Nodes

The method of integrating synonym nodes is divided into following three types. First, if both nodes are leaf nodes, the integrated node is a leaf node extracted after resolving collisions of labels, data types, occurrence constraints, and enumerations of the nodes.

If the two nodes have different labels, the label with broader meaning is chosen as the label of the integrated node. The broader label between label L_s and label L_t is calculated by BroaderLabelChoice(L_s, L_t) as defined in formula (2). If the two nodes have different data types, the type which can include the other type without conversion is selected. If any type between the two types does not include the other type, the two types are converted into a type which includes the two types. If the conversion is impossible, the data type of the integrated node becomes string. If

the two nodes have different occurrence constraint values, the integrated node takes the smaller value between the two minimum values as the minimum value and the larger value between the two maximum values as the maximum value.

$$\begin{gathered}\text{BroaderLabelChoice}(L_s, L_t) = \\ \begin{cases} L_s, & \text{where TotalLabelOccurence}(L_s) > \text{TotalLabelOccurence}(L_t) \\ L_t, & \text{otherwise} \end{cases} \\ \text{TotalLabelOccurence(a label)} = \\ W_{sch} * \text{LabelOccurenceInFiles(a label, a source and target schema)} \\ + W_d * \text{LabelOccurenceInFiles(a label, domain ontologies)} \end{gathered} \tag{2}$$

- LabelOccurenceInFiles(a label, files): the total occurrence number of tokens in a label in files
- L_s: a label of a source schema
- L_t: a label of a target schema
- W_{sch}: the weight of a source and target schema
- W_d: the weight of domain ontologies

Next, if a leaf node and an internal node are to be integrated, the integrated node will be an internal node with a mixed content model. The integrated node includes the sub-structure of the input internal node, and the collisions of labels and occurrence constraints of the two nodes are resolved by the same method as the leaf node integration.

Finally, if both nodes are internal nodes, the integrated node is an internal node which includes a conjunction of their sub-structures. In addition, the integrated node resolves the labels, occurrence constraints, and enumeration collisions by the same method as the leaf node integration.

On the other hand, schema integration methods might need to integrate schemas where the similar labels are further apart than nearest neighbors. For instance, in the case of two schemas 'XSD1: filmFestival-CannesFestival' and 'XSD2: filmFestival-internationalFilmFestival-CannesFestival', the two CannesFestival nodes, which have filmFestival as the same ancestor node but different closest ancestor element nodes, can be integrated. However, in the case of 'XSD3: school-name' and 'XSD4: school-department-manager-name', the two name nodes do not have the same meaning. It will be useful to distinguish the above two cases for the sophisticated integration of schemas, which is one of the future extensions of the proposed method.

3.2.3 Integrating Hypernym Nodes (and Holonym Nodes)

If two nodes with a hypernym (or holonym) relationship are integrated, the hyponym (or meronym) node is integrated as the closest descendant general node of the hypernym (or holonym) node. If the hyponym (or meronym) node is a synonym

node of the closest descendant general node of the hypernym (or holonym) node, the hyponym (or meronym) node and the closest descendant general node of the hypernym (or holonym) node are integrated by the method of integrating synonym nodes.

An example of integrating hypernym nodes in two schemas is described in the following: The domain ontology in Figure 2 d) defines that label Organization in Figure 2 a) is a hypernym of label Non-Profit-Organization in Figure 2 b). Therefore, the hyponym node Non-Profit-Organization is integrated as the closest descendant element node of the hypernym node Organization. Additionally, because the hyponym node Non-Profit-Organization is a synonym node of the closest descendant element node Non-Profit-Organization of the hypernym node Organization, the two synonym nodes are integrated by the method of integrating synonym nodes.

3.3 Optimization

In this step, the XML schema extracted in Section 3.2 is optimized using the rules for optimization in Figure 7. First, the proposed method applies Rules $1 \sim 3$ to all compositors in the XML schema by Breadth-First Search. Rules $1 \sim 3$ are proposed for compositor optimization and they are applied in order. After that, the method applies Rule 4 to the schema by Breadth-First Search.

Rule 1. If the current compositor node is the same kind as the child compositor node, delete the child compositor node. The sub-structure of the child compositor node becomes the sub-structure of the current compositor node.

Rule 2. If the current compositor node is not the same kind as the child compositor node and the child compositor node does not have any sibling node, delete the current compositor node. The sub-structure of the current compositor node becomes the sub-structure of the upper node.

Rule 3. If the child compositor node has an only child node, delete the child compositor node. The sub-structure of the child compositor node becomes the sub-structure of the current compositor node.

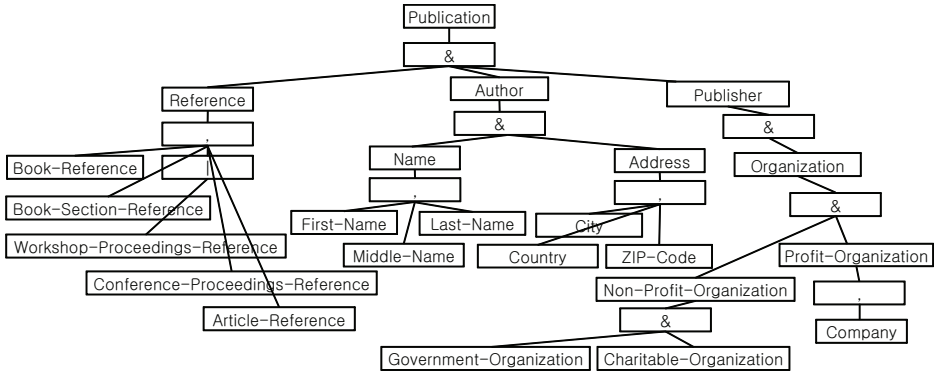
Rule 4. Delete cycles, transitive edges, and duplicated occurrence constraints.

Fig. 7. The rules for optimization

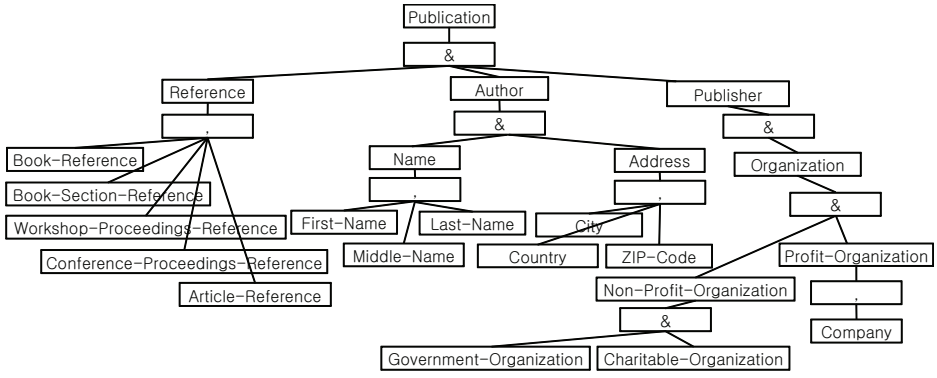
Rule 1 is reasonable because if the current compositor node is the same kind as the child compositor node, the child compositor node does not add any meaning to the schema. Moreover, Rules 2 and 3 are assumed to be correct for the following

reasons. If a sequence node has only one child node, the meaning of a sequence that the child nodes should be arranged in order in an XML document is lost. Also, if a choice node has only one child node, meaning of a choice that only one child node can appear in an XML document is lost. Therefore, if a compositor node has only one child node, the compositor kind specific meaning is lost. Also, if the compositor node with only one child node has an upper or child compositor node, the compositor node loses general compositor meaning. Therefore, although the compositor node with only one child node is deleted, meaning of the schema does not change.

The integrated schemas before and after application of optimization rules to the two schemas in Figure 2 is shown in Figure 8. An optimization process of the LCS of node Reference in Figure 8 a) is described in the following: the LCS of node Reference consists of a sequence and a choice. The choice has only one child node Workshop-Proceedings-Reference. Therefore, the choice is deleted and the only one child node becomes a child node of the sequence as shown in Figure 8 b).



a) Before the application of the optimization rules



a) After the application of the optimization rules

Fig. 8. An example of integrated schema trees

4 EXPERIMENTAL RESULTS

This section shows the performance of the proposed method. The proposed method was experimented with 15 XML schemas and 3 domain ontologies as shown in Table 4. This paper analyzed the performance of the proposed method in terms of utilizing domain ontologies and the proposed structural weight.

Domain	No. of XML schemas	Avg. no. of element and attribute nodes	Avg. no. of leaf nodes	Sources of domain ontology
Publication	5	23	16	`http://www.aktors.org/ontology/portal`
Movie	5	35	29	`http://139.91.183.30:9090/RDF/VRP/Examples/moviedatabase.rdf`
Science	5	33	22	`http://www.astro.umd.edu/~eshaya/astro-onto/owl/IVOAO.owl`

Table 4. XML schemas and domain ontologies

4.1 Assessment Measure

This paper uses the assessment measures of Meo et al. [7] such as precision, recall, F-measure, overall, and Relative Schema Size (RSS). Additionally we define Relative Compositor Number (RCN) shown in Table 5.

The higher precision, recall, F-measure, or overall indicates the more accuracy of an integration method. RSS and RCN measure the complexity of the schema integrated. RSS falls within $[0.5, 1]$ and RCN ranges between 0 and ∞. The lower value of RSS or RCN represents the less complexity of the schema integrated.

4.2 Performance Evaluation

This paper categorized the experimental results based on an application domain. The system performed integration on every combination of input schemas in each domain. Tables 6 through 9 show the mean value of the integration results on the every combination in each domain. Particularly, Tables 6 through 8 shows how much the utilization of a domain ontology and the structural weight affect the schema integration. In the version of the method with no use of a domain ontology, the synonym, hypernym, and holonym relationships were determined from WordNet. In addition, Table 9 compares the RCNs of the two versions of the proposed method depending on whether the compositor optimization rules are used or not.

The experimental results show that the utilization of a domain ontology by the proposed method improved the precision, recall, F-measure, overall, and RSS. This

Measure	Definition
Precision	$\frac{\lvert A \cap C \rvert}{\lvert C \rvert}$, where A is the set of matchings between an integrated schema and input schemas by a domain expert, C is the set of matchings between an integrated schema and input schemas returned by the proposed method. Here $A \cap C$ is the common matchings between A and C.
Recall	$\frac{\lvert A \cap C \rvert}{\lvert A \rvert}$
F-measure	$2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$
Overall	$\text{Recall} * \left(2 - \frac{1}{\text{Precision}}\right)$
RSS	$\frac{\lvert \text{ConstructSet(IntegratedSchema)} \rvert}{\lvert \text{ConstructSet(SourceSchema)} \rvert + \lvert \text{ConstructSet(TargetSchema)} \rvert}$, where ConstructSet($S$) is the set of all elements and attributes in schema S.
RCN	$\frac{\lvert \text{CompositorSet(IntegratedSchema)} \rvert}{\lvert \text{CompositorSet(SourceSchema)} \rvert + \lvert \text{CompositorSet(TargetSchema)} \rvert}$, where CompositorSet($S$) is the set of all compositors in schema S.

Table 5. Assessment measures

	With a domain ontology and the structural weight	Without a domain ontology	Without the structural weight
Precision	0.952	0.804	0.811
Recall	0.808	0.679	0.626
F-measure	0.853	0.696	0.674
Overall	0.766	0.525	0.512
RSS	0.8228	0.8605	0.8801

Table 6. Experimental results on publication schemas

is due to the fact that domain ontologies represent the synonym, hypernym, and holonym relationships. For example, the proposed method could detect the hypernym relation between labels Serial-Publication and Journal based on the publication domain ontology. In addition, the proposed method integrated the closest descen-

	With a domain ontology and the structural weight	Without a domain ontology	Without the structural weight
Precision	0.984	0.984	0.913
Recall	0.812	0.553	0.548
F-Measure	0.815	0.597	0.577
Overall	0.794	0.535	0.514
RSS	0.8548	0.9084	0.9121

Table 7. Experimental results on movie schemas

	With a domain ontology and the structural weight	Without a domain ontology	Without the structural weight
Precision	0.991	0.991	0.874
Recall	0.870	0.709	0.732
F-Measure	0.912	0.794	0.784
Overall	0.860	0.699	0.620
RSS	0.8418	0.8647	0.8664

Table 8. Experimental results on science schemas

	With the compositor optimization rules	Without the compositor optimization rules
Publication	0.7750	0.8102
Movie	0.7517	0.7979
Science	0.8176	0.8455
Average	0.78143	0.81787

Table 9. RCNs on various domain schemas

dant element node Journal of Serial-Publication in the first schema and node Journal in the second schema by integrating synonym nodes. However, in the movie and science domains shown in Table 7 and Table 8, the utilization of a domain ontology has no effect on precision. This was because the domain ontologies did not have enough information to correct wrong matchings.

The utilization of the structural weight by the proposed method also improved the precision, recall, F-measure, overall, and RSS. The structural weight contributes a lot to the more accurate determination of the synonym, hypernym, and holonym relationships. For instance, the proposed method extracted the synonym relation between node Genre in the first movie schema and the child node MovieGenre of Movie in the second movie schema, and integrated the two nodes by integrating synonym nodes. With the version of not using the structural weight, this synonym relation could not be determined, resulting in a lower recall, F-measure, and overall, and a higher RSS.

In addition, the proposed method deleted semantically redundant compositor nodes by the compositor optimization rules, resulting in a lower RCN while keeping the meaning of the input compositor nodes. In the integrated schema of the first and third science schemas by the version of not using the compositor optimization rules, the LCS of node Scientist consists of a sequence node and its child node, a choice node. Also, the choice node has only one child node Geophysicist. However, in the integrated schema by the proposed method with the application of Rule 3 of Figure 7, the LCS of node Scientist consists of a sequence node which contains Geophysicist and the other element nodes.

4.3 Comparison with Previous Works

The qualitative comparison of our system with previous works is shown in Table 10. The feature of using the structural information of tokens in labels indicates whether a method utilizes structural importance information of each token in schema integration.

Features	Huma et al. [3]	Mello and Heuser [4]	Meo et al. [7]	Cruz et al. [8]	The proposed method
Conceptual model	XML Ontology	ORM/NIAM	XML schema	RDFS ontology	XML schema
Using a domain ontology	O	X	X	X	O
Label integration	O	O	O	O	O
Structure integration	O	O	O	O	O
Type integration	O	X	O	X	O
Using structural information of tokens in labels	X	X	X	X	O
Considering an order information among nodes	X	X	O	X	O
Optimization	O	O	O	X	O
Compositor optimization	X	X	X	X	O

Table 10. Qualitative comparison of our system with previous works

Although the proposed method tries to minimize human intervention, some cases need to be handled by human beings to integrate XML schemas correctly. Particularly, most of the cases occur in the process of determining a synonym, hypernym, and holonym relationship as shown in Figure 9.

5 CONCLUSIONS AND FUTURE WORKS

XML is widely used as a standard format for data expression and exchange in diverse computing environments. To extract information efficiently from XML documents, it is necessary to integrate XML schemas. This paper proposed a sophisticated method of integrating XML schemas in the same application domain. The proposed method determines the synonym, hypernym, and holonym relationships among XML elements and attributes by using a domain ontology. Specifically, it takes the structural information of elements and attributes into account. The conciseness of the schema integrated is also considered. It deletes semantically redundant nodes through the proposed compositor optimization rules.

Experimental results show that using a domain ontology or the structural weight improved the accuracy of schema integration as well as the complexity of the integrated schema. In addition, the proposed compositor optimization decreased the

1. If a synonym, hypernym, or holonym relationship information between two labels in input schemas is absent from general dictionaries and domain ontologies
   (a) If a label is absent from general dictionaries and domain ontologies because the dictionaries and domain ontologies do not have enough information
   (b) If a label is absent from a domain ontology because a complete standard to determine the range of a domain is impossible to exist
   (c) If a label is an archaic or coined word
   (d) If input schemas are written in different languages
   (e) If a label is misspelled in an input schema, general dictionary, or domain ontology
2. If a label has more than one meaning and it is impossible to determine automatically in which meaning the label was used
3. If a label consists of more than one token, and a relationship among those tokens cannot be automatically determined
4. If synonym, hypernym, or holonym relationship information between two labels is latent in a path in an input schema, and the information cannot be determined automatically

Fig. 9. The cases when human intervention is needed to determine a synonym, hypernym, or holonym relationship

ratio of compositor nodes in the integrated schema while preserving the meaning of compositor nodes in input schemas.

Recently, as the automated and intelligent processing of information is needed, there is a growing interest in adopting semantics on the Web. In dealing with numerous and heterogeneous schemas, ontology technologies are required to map between them. In reality, an ontology is considered as one of the main components of semantic Web services [18], which allow the semantic annotation of XML schemas by using domain ontologies. Therefore, we have a plan to expand the proposed method for integrating domain ontologies.

Acknowledgements

This research was supported by the Seoul R & BD Program (10705), Korea.

REFERENCES

[1] World Wide Web Consortium: Extensible Markup Language (XML) 1.0 (Fourth Edition). W3C Recommendation, available on: http://www.w3.org/TR/2006/REC-xml-20060816/, 2006.

[2] Ray, P.—Paramesh, N.—Ying, W.—Sujanani, A.—Lee, D.—Bhar, R.: Design and Development of Financial Applications Using Ontology-Based Multi-Agent Systems. Computing and Informatics, Vol. 28, 2009, No. 5, pp. 635–654.

[3] Huma, Z.—Rehman, J.—Iftikhar, N.: An Ontology-Based Framework for Semi-Automatic Schema Integration. Journal of Computer Science and Technology, Vol. 20, 2005, No. 6, pp. 788–796.

[4] Mello, R. D. S.—Heuser, C. A.: BInXS: A Process for Integration of XML Schemata. Proceedings of 17th International Conference on Advanced Information Systems Engineering (CAiSE), Porto, Portugal, June 2005, 3520, pp. 151–166.

[5] Islam, A.—Inkpen, D.—Kiringar, I.: Applications of Corpus-Based Semantic Similarity and Word Segmentation to Database Schema Matching. International Journal on Very Large Data Bases (The VLDB Journal), Vol. 17, 2008, No. 5, pp. 1293–1320.

[6] Do, H. H.—Rahm, E.: Matching Large Schemas: Approaches and Evaluation. Information Systems, Vol. 32, 2007, No. 6, pp. 857–885.

[7] Meo, P.—Quattrone, D.—Terracina, G.—Ursino, D.: Integration of XML Schemas at Various "Severity" Levels. Information Systems, Vol. 31, 2006, No. 6, pp. 397–434.

[8] Cruz, I. F.—Xiao, H.—Hsu, F.: An Ontology-Based Framework for XML Semantic Integration. Proceedings of International Database Engineering and Applications Symposium (IDEAS), Coimbra, Portugal, July 2004, pp. 217–226.

[9] Jeong, E.—Hsu, C.-N.: View Inference for Heterogeneous XML Information Integration. Journal of Intelligent Information Systems, Vol. 20, 2003, No. 1, pp. 81–99.

[10] Yang, X.—Lee, M. L.—Ling, T. W.: Resolving Structural Conflicts in the Integration of XML Schemas: A Semantic Approach. Proceedings of 22nd International Conference on Conceptual Modeling, Chicago, Illinois, October 2003, 2813, pp. 520–533.

[11] Rhim, T.-W.—Lee, K.-H.: Clustering of XML Schemas for Information Integration. Journal of Computer Information Systems (JCIS), Vol. 46, 2006, No. 2, pp. 3–13.

[12] Deen, S. M.—Ponnamperuma, K.: Dynamic Ontology Integration in a Multi-Agent Environment. Proceedings of 20th International Conference on Advanced Information Networking and Applications (AINA '06), Vienna, Austria, April 2006, pp. 373–378.

[13] Miller, G. A.: WordNet: A Lexical Database for English. Communications of the ACM, Vol. 38, 1995, No. 11, pp. 39–41.

[14] World Wide Web Consortium: OWL Web Ontology Language Overview. W3C Recommendation, Available on: http://www.w3.org/TR/owl-features/, 2004.

[15] Kim, J.-M.—Kwon, S.-H.–Park, Y.-T.: Enhanced Search Method for Ontology Classification. Computing and Informatics, Vol. 28, 2009, No. 6, pp. 795–809.

[16] Hadjieleftheriou, M.—Srivastava, D.: Weighted Set-Based String Similarity. IEEE Data Engineering Bulletin, Vol. 33, 2010, No. 1, pp. 25–36.

[17] Moh, C.-H.—Lim, E.-P.—Ng, W.-K.: Re-Engineering Structures from Web Documents. Proceedings of 5th ACM Conference on Digital Libraries, San Antonio, Texas, June 2000, pp. 67–76.

[18] Du, H.-J.—Shin, D.-H.—Lee, K.-H.: A Sophisticated Approach to Semantic Web Services Discovery. Journal of Computer Information Systems, Vol. 48, 2008, No. 3, pp. 44–60.

Haeran Kang received her B. Sc. degree in Computer Science and Engineering from Sangmyung University in 2003 and her M. Sc. degree in Computer Science from Yonsei University, Seoul, Korea in 2007. She is currently a Ph. D. candidate in the Department of Computer Science at Yonsei University. Her research interests include ontology, XML and semantic Web.

Kyong-Ho Lee received his B. Sc., M. Sc., and Ph. D. degrees in Computer Science from Yonsei University, Seoul, Korea, in 1995, 1997, and 2001, respectively. Currently, he is an Associate Professor in the Department of Computer Science at Yonsei University. Previously, he worked as a Guest Researcher in the IT Laboratories at NIST (National Institute of Standards and Technology), Maryland. He was a Visiting Professor at the School of Information and Computer Sciences of University of California, Irvine. His research interests include service-oriented computing and semantic Web. He is a member of the editorial boards of Journal of Web Science, Journal of Information Processing Systems, and Journal of Korea Multimedia Society.