Information Engineering Express
International Institute of Applied Informatics
2016, Vol.2, No.2, 37 - 46

Class-Based Generalization and Class-Based Specialization

Teruhisa Hochin [*], Hiroki Nomiya [†]

Abstract

This paper extends class-based generalization and class-based specialization. In the class-based generalization (specialization, respectively), the names of the attributes of the superclass (subclass) are changed according to the name of the superclass (subclass). Revised class-based generalization (specialization, respectively) permits subclasses to have the attributes whose names are the same as those of the attributes of the superclass (superclasses). This extension improves the representation capability through the class-based generalization and the class-based specialization.

Keywords: class-oriented, data model, generalization, semantic, specialization.

1 Introduction

In recent years, various kinds of knowledge have been represented, gathered, and used around us according to the advances of computers and computer networks. Wikipedia is an encyclopedia collaboratively created over the Internet [1]. It gathers the knowledge of many people, and many people use it. Question and answer sites receive questions, and many users answer them [2]. The knowledge has been gathered all over the world. We could find some answers to our questions on these web sites. The conceptual descriptions of web resources have been represented in the Resource Description Framework (RDF), which is a kind of semantic network [3]. By using these descriptions, web resources could effectively be manipulated. These descriptions represent the knowledge of web resources.

Generalization is an important mechanism in conceptualizing the real world [4][5][6][7]. Generalization is an abstraction in which a set of similar classes is regarded as a generic class [4]. In making such an abstraction, many individual differences between classes may be ignored. For example, students and teachers are required to be treated just as persons. In this case, the class ”Student” and the class ”Teacher” are generalized to the class ”Person.” The attributes which are of both of the classes ”Student” and ”Teacher” are those of the generalized class ”Person.” The instances of ”Student” and ”Teacher” are treated as those of ”Person.” Attributes are inherited upward in the case of generalization. On the other hand, specialization can be used to define possible roles for members of a given class [6].

[*] Kyoto Institute of Technology
[†] Kyoto Institute of Technology

For example, the class ”Dog” is defined as a specialized class of the class ”Animal.” All of the attributes of ”Animal” are inherited to ”Dog.” That is, the class ”Dog” has all of the attributes of the class ”Animal.” Moreover, the class ”Dog” could have its own attributes. Attributes are inherited downward in the case of specialization.

Generalization and specialization are introduced in many data models. The entity-relationship (ER) model supports ISA relationships for specialization and generalization [5]. The IFO model introduces two kinds of ISA relationships: specialization and generalization relationships [6]. Generalization and specialization are introduced into a graph-based data model [7]. Generalization and specialization of edges as well as nodes are considered. These are examples of the support of generalization and specialization. The representation power becomes high through them.

In the database design, there are some cases where generalization or specialization does not work well. Let us consider the situation that we treat two classes ”Student” and ”Teacher.” The class ”Student” has an attribute ”student_no,” while the class ”Teacher” has an attribute ”teacher_no.” When we try to generalize these two classes to create a generalized class ”Person,” we encounter the problem. As these attributes ”student_no” and ”teacher_no” are not the same, the attribute corresponding to them, e.g., ”person_no,” is not included in the class ”Person.”

The class-based generalization and the class-based specialization have been proposed to address to this kind of problem [8]. When the attributes having their class names in the attribute names, are inherited upward to the superclass, the name of the attribute generalized has the superclass name in its name. In the case described above, the attribute name becomes ”person_no” when the name of the superclass is ”Person.” For the class-based specialization, the functionality is the same as the class-based generalization. Database designers can put appropriate names to the generalized attributes. These mechanisms result in the improvement of understandability and maintainability of databases.

The original class-based generalization, however, does not consider that subclasses have the attributes whose names are the same as that of the generalized attribute of the superclass when the names include class names in them. The class-based specialization does not permit subclasses to have the attributes whose names are the same as that of the attribute of the superclass, when the names include class names in them. The class-based specialization is also too strict in the case of multiple inheritance.

This paper extends the class-based generalization and the class-based specialization to permit subclasses to have the attributes whose names are the same as that of the attributes of the superclass when the names include class names. This extension improves the representation capability through the class-based generalization and specialization.

The remainder of the paper is organized as follows: Section 2 describes traditional generalization and specialization. Section 3 explains the class-based generalization and specialization. Section 4 extends the class-based generalization and the class-based specialization. Section 5 gives some considerations. Lastly, Section 6 concludes the paper.

2 Generalization and Specialization

2.1 Fundamental Structure

The structure of a class is represented with a pair of its name and a set of attributes. An attribute is a pair of its name and its data type. For example, a class named ”Student” is represented with (”Student”, { (”student_no”, Int), (”name”, String), (”address”, String) }).

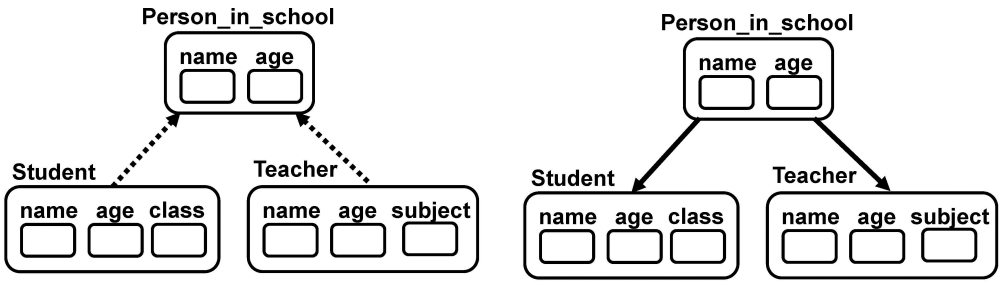


Figure 1: Examples of generalization relationships (a) and specialized ones (b).

2.2 Generalization and Specialization

2.2.1 Generalization

In the generalization, the common attributes of original classes are the attributes of a generalized class. When a class ”Student” has attributes named ”name,” ”age,” and ”class,” and a class ”Teacher” has attributes named ”name,” ”age,” and ”subject,” the class obtained by the generalization from these two classes has the attributes named ”name” and ”age.”

This generalization is shown in Figure 1(a). A generalization relationship is represented with a broken arrow. The classes ”Student” and ”Teacher” are generalized to the class ”Person_in_school.” The attributes of ”Student” and ”Teacher” are inherited upward. In the generalization, the classes ”Student” and ”Teacher” are usually called *subclasses*, and the class ”Person_in_school” is called a *superclass*. The attribute of a subclass (superclass, respectively) is called an original (generalized) attribute in this paper.

Classes form a layer structure through generalization relationships. This structure is called the *class lattice*. The direction of edges of this lattice is opposite to that of generalization relationships.

Although the *data type tree* was also introduced in order to generalize data types [7], it is omitted because of the space limitation.

2.2.2 Specialization

In the specialization, the attributes of an original class are the attributes of a specialized class. When a class ”Person” has attributes named ”name” and ”age,” the class obtained by the specialization from the class ”Person” also has the attributes named ”name” and ”age.” These attributes are said to be inherited downward. The specialized class also could have attributes of its own. The attribute ”class” of the class ”Student” is an example of this kind of attribute.

This specialization is shown in Figure 1(b). A specialization relationship is represented with an arrow. The classes ”Student” and ”Teacher” are specialized from the class ”Person_in_school.” The attributes of ”Person_in_school” are inherited downward. In the specialization, the classes ”Student” and ”Teacher” are usually called *subclasses*, and the class ”Person_in_school” is called a *superclass*.

Classes form a class lattice through specialization relationships as through generalization ones. The direction of edges of this lattice is the same as that of specialization relationships.

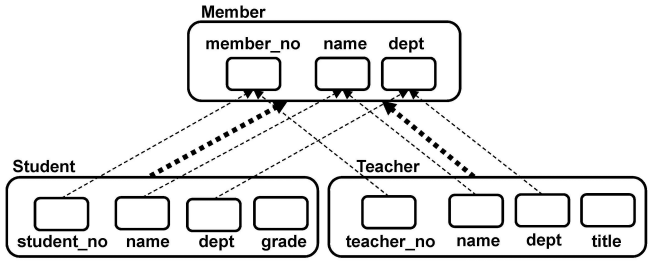


Figure 2: An example of class-based generalization.

3 Original Class-Based Generalization and Specialization

3.1 Class-Based Generalization

Class-based generalization enables a superclass to have attributes whose names are different from those of subclasses [8]. This capability is similar to that of the semantic generalization [9][10][11][12][13][14]. The names of the original and the generalized attributes are, however, based on the names of classes. These are not free from the class names.

Definition 1 Let C_{sup} be a superclass based on n subclasses C_{sub_i} $(1 \leq i \leq n)$. It is said that C_{sup} is created through the class-based generalization from the subclasses C_{sub_i} $(1 \leq i \leq n)$ when an attribute of C_{sup}, say $attr_{sup}$, satisfies one of the following conditions:
(1) For all C_{sub_i}, there is the attribute in C_{sub_i} whose name is the same as that of $attr_{sup}$.
(2) The name of $attr_{sup}$ is the concatenation of the name of C_{sup} and a character string str. For all C_{sub_i}, there is the attribute in C_{sub_i} whose name is the concatenation of the name of C_{sub_i} and the character string str.

Example 1 An example of the class-based generalization is shown in Figure 2. The attributes common to all of subclasses, which are ”name” and ”dept,” are the attributes of the class ”Member.” These are the attributes satisfying the condition (1) of **Definition 1**. The attribute ”student_no” of the class ”Student,” the attribute ”teacher_no” of the class ”Teacher,” and the attribute ”member_no” of the class ”Member” are an example of the attributes satisfying the condition (2) of **Definition 1**, although lower and upper letters of names are ignored. In this case, the character string str is ”_no.” The attribute ”member_no” is treated as the attribute generalized from ”student_no” and ”teacher_no.”

As shown in Example 1, the generalized attribute, whose name is not the same as those of the subclasses, can be introduced through the class-based generalization. This makes it possible to give more appropriate names to the attributes of subclasses than the ordinary generalization. It is preferred from the viewpoint of the database design, where it is recommended to put an appropriate name to an attribute [5]. For example, the name ”person_name” is said to be better than the name ”name” for the class ”Person.”

3.2 Class-Based Specialization

The class-based specialization enables a subclass to have attributes whose names are different from those of superclasses [8]. The names of the original and the specialized attributes

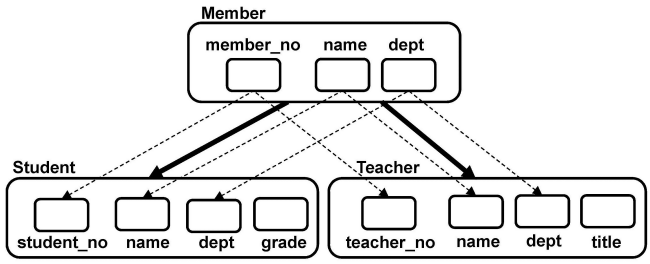


Figure 3: An example of class-based specialization.

are based on the names of classes.

Definition 2 Let C_{sub} be a subclass based on n superclasses C_{sup_i} $(1 \leq i \leq n)$. It is said that C_{sub} is created through the class-based specialization from the superclasses C_{sup_i} $(1 \leq i \leq n)$ when an attribute of C_{sub}, say $attr_{sub}$, satisfies one of the following conditions:
(1) For all C_{sup_i}, there is the attribute in C_{sup_i} whose name is the same as that of $attr_{sub}$.
(2) The name of $attr_{sub}$ is the concatenation of the name of C_{sub} and a character string str. For all C_{sup_i}, there is the attribute in C_{sup_i} whose name is the concatenation of the name of C_{sup_i} and the character string str.

Example 2 An example of the class-based specialization is shown in Figure 3. The attribute "member_no" of the class "Member" is an example of the attribute satisfying the condition (2) of **Definition 2**, although lower and upper letters are ignored. This attribute is inherited downward to the attribute "student_no" of the class "Student," and the attribute "teacher_no" of the class "Teacher." The other attributes of the class "Member," which are "name" and "dept," are inherited downward to the classes "Student" and "Teacher" as they are because these are the attributes satisfying the condition (1) of **Definition 2**. The attribute "grade" of the class "Student" is a specific attribute of this class.

4 Class-Based Generalization and Specialization Revised

4.1 Class-Based Generalization Revised

The original class-based generalization does not consider the situation that subclasses have the attributes whose names are the same as that of the attribute generalized to the superclass when the names include class names in them. The class lattice shown in Figure 4 (a) is of this situation. The attributes "member_no" are included in the subclasses "Student" and "Teacher." These attributes are generalized to the attribute "member_no" of the superclass "Member" owing to the condition (1) of **Definition 1**. As the attributes "student_no" and "teacher_no" are included in the classes "Student" and "Teacher," respectively, these attributes are also generalized to the attribute "member_no" of the superclass "Member" due to the condition (2) of **Definition 1**. This situation does not violate the conditions of **Definition 1**.

Let us consider another situation shown in Figure 4 (b). As all of the subclasses do not have the attribute "member_no," the condition (1) of **Definition 1** is not satisfied. The condition (2) of **Definition 1** is neither satisfied because the class "Teacher" does not have

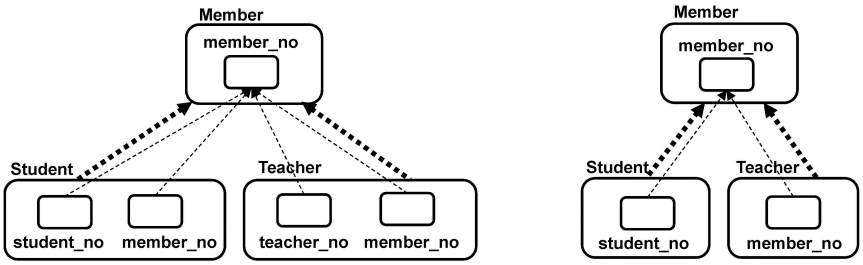


Figure 4: Class-based generalization revised.

the attribute "teacher_no." In this situation, the superclass "Member" cannot include the attribute "member_no" as the generalized one under **Definition 1**. However, the attribute "member_no" had better be included in the superclass as the generalized one because the attribute "member_no" of the class "Teacher" is the same name as that of the superclass "Member," and all of the subclasses, "Student" and "Teacher" have the attributes related to the attribute "member_no." **Definition 1** is revised in order to permit this situation.

Definition 3 Let C_{sup} be a superclass based on n subclasses C_{sub_i} $(1 \leq i \leq n)$. It is said that C_{sup} is created through the class-based generalization from the subclasses C_{sub_i} $(1 \leq i \leq n)$ when an attribute of C_{sup}, say $attr_{sup}$, satisfies one of the following conditions:
(1) For all C_{sub_i}, there is the attribute in C_{sub_i} whose name is the same as that of $attr_{sup}$.
(2) The name of $attr_{sup}$ is the concatenation of the name of C_{sup} and a character string str for all C_{sub_i}, where there are one or two attributes in C_{sub_i} whose names are the concatenation of the name of C_{sub_i} or C_{sup}, and the character string str.

From here on, the class-based generalization means the one based on **Definition 3**, and the original one means the one based on **Definition 1**.

The procedure of deciding the attributes of a superclass is shown in Figure 5. This procedure takes the superclass name and a set of subclasses as the inputs. At Line 6, a class `c1` is selected from the set of subclasses. It is considered to be good to select the subclass having the least attributes because the number of loops becomes the minimum. All of the attributes are examined whether they can be generalized or not at Lines 8–13. Lines 8 and 9 are for the attributes satisfying the condition (1) of **Definition 3**, while Lines 10 to 13 are for those satisfying the condition (2). The expression `a - b` (`a + b`, respectively) means that the character string `b` is removed from (appended to) the character string `a`.

4.2 Class-Based Specialization Revised

According to **Definition 2**, the subclass "TA," which is a subclass of classes "Student" and "Teacher," has only the attributes "ta_no" and "name" as shown in Figure 6 (a) because all of the superclasses must have the attributes whose names are the same one. This is, however, too restrictive to use. All of the attributes of superclasses are inherited downward to a subclass in the traditional specialization. In the class-based specialization, this inheritance must be supported as shown in Figure 6 (b).

Moreover, the attribute, whose name is the same as that of the attribute of the superclass, as well as the one, whose name includes the class name in it, may be required to be included

```
 1: PROCEDURE: Decide_attributes_of_superclass
 2: INPUT: sup: superclass name
 3:        Csub: a set of subclasses
 4: OUTPUT: Asup: attributes of superclass
 5:     Asup = {}
 6:     c1 = one class in Csub
 7:     foreach attribute ai of subclass c1
 8:         if(all cj in Csub have ai) then
 9:             Asup = Asup U {ai}
10:         else if(for all cj in Csub, cj has (cj + (ai - c1)) or (sup + (ai - c1))) then
11:             sa = sup + (ai - c1)
12:             Asup = Asup U {sa}
13:         endif
14:     end_foreach
15: END
```

Figure 5: Procedure of deciding attributes of a superclass for class-based generalization.

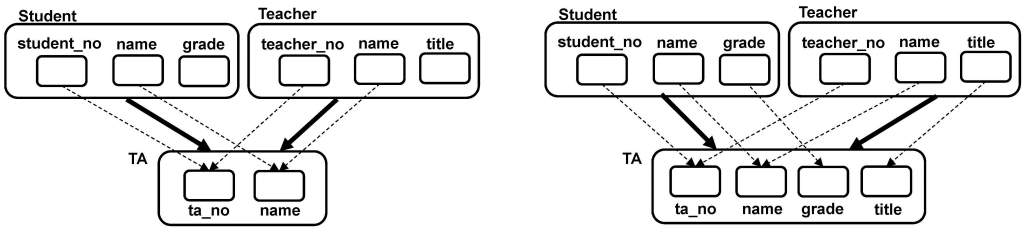


Figure 6: Class-based specialization with multiple inheritance.

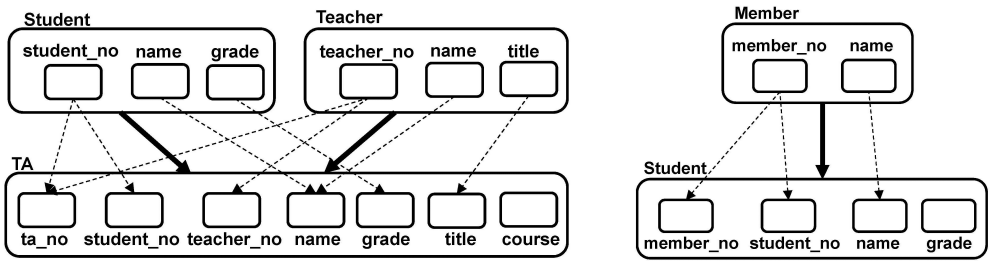
in a subclass. Figure 7 shows examples of this situation. In Figure 7 (a), the attribute ”ta_no” is the one inherited from the attributes ”student_no” and ”teacher_no” of the classes ”Student” and ”Teacher,” respectively, based on the class-based specialization. The attributes ”student_no” and ”teacher_no” are also included in the subclass ”TA.” Figure 7 (b) shows the situation in the single inheritance. The attribute ”member_no” of the class ”Member” is inherited to the class ”Student” as the attributes ”member_no” and ”student_no.”

Definition 2 is revised in order to permit this situation. For this purpose, the following definition only defines the attributes inherited from superclasses. A subclass could have its specific attributes as in the conventional specialization.

Definition 4 Let C_{sub} be a subclass based on n superclasses C_{sup_i} $(1 \leq i \leq n)$. It is said that C_{sub} is created through the class-based specialization from the superclasses C_{sup_i} $(1 \leq i \leq n)$ when an attribute of C_{sub} inherited from the superclasses, say $attr_{sub}$, satisfies one of the following conditions:
(1) There exists the attribute in C_{sup_i} whose name is the same as that of $attr_{sub}$.
(2) The name of $attr_{sub}$ is the concatenation of the name of C_{sub} and a character string str, where there is the attribute in C_{sup_i} whose name is the concatenation of the name of C_{sup_i} and the character string str.

From here on, the class-based specialization based on **Definition 4** is called the class-based specialization, and the one based on **Definition 2** is called the original one.

The procedure of deciding the attributes of a subclass inherited from the superclasses



(a) (b)

Figure 7: Class-based specialization revised.

```
 1: PROCEDURE: Decide_attributes_of_subclass_inherited
 2: INPUT: sub: subclass name
 3:        Csup: a set of superclasses
 4: OUTPUT: Asub: attributes of subclass
 5:     Asub = {}
 6:     foreach superclass cj in Csup
 7:         foreach attribute ai of superclass cj
 8:             Asub = Asub U {ai}
 9:             if(ai contains cj) then
10:                 sa = sub + (ai - cj)
11:                 Asub = Asub U {sa}
12:             endif
13:         end_foreach
14:     end_foreach
15: END
```

Figure 8: Procedure of deciding attributes of a subclass inherited from superclasses for class-based specialization.

is shown in Figure 8. All of the attributes of the superclasses are inherited to the subclass (Line 8). Additionally, the attribute whose name contains the name of the superclass in its name is inherited to the subclass. The attribute name, however, contains the subclass name instead of the superclass name.

5 Considerations

As the attributes which do not exist in a class cannot be inherited upward (downward, respectively) in the conventional generalization (specialization), the attribute having a name different from that of a class cannot be included in the conventional generalization (specialization). In the class-based generalization (specialization), the attribute having a name different from that of a subclass (superclass) can be included.

Semantic generalization and specialization have been proposed [9][10][11][12][13][14]. In the semantic generalization (specialization, respectively), the superclass (subclass) could have the attribute having the name different from those of subclasses (superclasses) as with the class-based one. The semantic generalization and the semantic specialization, however, require additional information. This information includes correspondences and viewpoints. The correspondence represents the mapping between the attributes of subclasses and those of superclasses. The viewpoint is a lattice of concepts. It guarantees the validity of the

attribute names. Although the attributes having different names can be included in a superclass or subclass owing to the correspondences and the viewpoints, specifying them is cumbersome. The class-based generalization and specialization do not require them. The attribute name is automatically decided according to the name of the superclass or subclass. It is considered that it is easier to use the class-based generalization and specialization than to use the semantic ones.

The condition (1) of **Definition 3** and that of **Definition 4** are of the conventional generalization and the conventional specialization, respectively. The condition (2) of **Definition 3** and that of **Definition 4** are specific to the class-based ones. The attribute name is changed according to the name of a superclass (subclass, respectively) in the class-based generalization (specialization).

6 Conclusion

This paper revised the class-based generalization and the class-based specialization. The revised class-based generalization (specialization, respectively) permits a subclass to have the attribute whose name is the same as that of the attribute of the superclass (those of the attributes of the superclasses) as well as the name including the class name in it. Database designers can generalize (specialize) attributes to the one having an appropriate name more flexibly than the original class-based generalization (specialization).

Semantic generalization and specialization are also flexible ones as with the class-based ones. The integration of them is included in future work. Implementation of the class-based generalization and the class-based specialization for showing the feasibility and the effectivity is also in future work.

References

[1] Wikimedia Foundation, ”Wikipedia,” https://en.wikipedia.org/wiki/ Wikipedia

[2] Yahoo!, ”Yahoo! Answers,” https://answers.yahoo.com/

[3] World Wide Web Consortium (W3C), ”Resource Description Framework (RDF),” http://www.w3.org/TR/PR-rdf-syntax/

[4] J. M. Smith and D. C. P. Smith, ”Database abstractions: Aggregation and generalization,” ACM Transactions on Database Systems, vol. 2, no. 2, 1977, pp. 105–133.

[5] A. Silberschatz, H. Korth, and S. Sudarshan, Database System Concepts (4th ed.), McGraw Hill, 2002.

[6] S. Abiteboul and R. Hull, ”IFO: A Formal Semantic Database Model,” ACM Transactions on Database Systems, vol. 12, no. 4, 1987, pp. 525–565.

[7] Y. Ohira, T. Hochin, and H. Nomiya, ”Introducing Specialization and Generalization to a Graph-Based Data Model,” Lecture Notes in Computer Science, Springer, 6884 (Proc. of 15th Int’l Conf. on Knowledge-Based Intelligent Information and Eng. Systems (KES2011)), 2011, pp. 1–13.

[8] T. Hochin and H. Nomiya, ”Class-based Generalization and Specialization,” Proc. of 2015 International Conference on Computer Application Technologies (CCATS2015), 2015, pp. 42–47.

[9] T. Hochin and H. Nomiya, ”Semantic Generalization in Graph-Based Data Model and Its Easy Usage,” ACIS International Journal of Computer & Information Science, vol. 13, no. 1, 2012, pp. 8–18.

[10] T. Hochin, ”Extension for Explicit Specification of Semantic Generalization,” Proc. of 14th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD 2013), 2013, pp. 432–437.

[11] T. Hochin and H. Nomiya, ”Inner Specialization and Generalization in Semantic Specialization and Generalization,” Proc. of 1st ACIS International Symposium on Applied Computers & Information Technology (ACIT 2013), 2013, pp. 349–354.

[12] A. Hayashi, A. Terakawa, and T. Hochin, ”Semantic Generalization and Semantic Specialization in Archaeology,” Proc. of 2nd ACIS International Symposium on Applied Computers & Information Technology (ACIT 2014), 2014, pp. 807–812.

[13] T. Hochin and H. Nomiya, ”Semantic Specialization: Specialization Using Explicit Specification and Semantic Information,” ACIS International Journal of Computer & Information Science, vol. 14, no. 1, 2013, pp. 21–30.

[14] T. Hochin and H. Nomiya, ”Explicit Generalization as Generalization of Semantic Generalization,” Proc. of 3rd ACIS International Conference on Applied Computers & Information Technology (ACIT 2015), 2015, pp. 511–516.