A General Metadata Schema Operation using Formula Expression

Toshio Kodama
School of Engineering, University of Tokyo
and Maeda Corp.
Tokyo, Japan
kodama@ken-mgt.t.u-tokyo.ac.jp
kodama.ts@jcity.maeda.co.jp

Yoichi Seki
Software Consultant
Tokyo, Japan
gamataki51@hotmail.com

***Abstract*—In data management, there is a situation where equivalent objects are managed in different management spaces. This often brings about a lack of data consistency, which can often decrease the efficiency of management work. We call it the data overlapping problem. We consider the attaching function by an equivalent relation in the Incrementally Modular Abstraction Hierarchy to be quite effective to solve the problem. In this paper, we propose a metadata centralized space, a data centralized space, and their interconversion maps using Formula Expression. We then apply them to parts ledger management, where part data oftentimes becomes unexpectedly overlapped in metadata schema-centered management. These help users to arrange dynamic worlds from a data-centric viewpoint and prevent data overlap. In other words, if you utilize these functions in data management, you can reconstruct data spaces from different viewpoints.**

Keywords-metadata schema; topological space; formula expression; attaching function.

I. INTRODUCTION

In recent data management, situations where data and their dependencies change dynamically and constantly have been increasing in business environments. When data are managed after designing metadata schemas, data overlap occurs, which brings about a lack of data inconsistency. For example, when customer ledgers are designed and managed in different departments within a company, data on the same customer may not be recognized as the same in the system. As a result, the more the number of customer ledgers increases, the more complexity of the system increases.

To avoid this, certain functions are needed: 1. As with data, metadata schemas should also work flexibly; and 2. A data model should support the mechanism which guarantees an equivalence relation. But, in data management using conventional data models [2][3][5], unlike data, metadata schemas are not generally dealt with. Instead, they have to be defined in advance in the system design, and an equivalent relation is not modeled. A more powerful mathematical and fundamental background and a finite automaton to implement it are needed to model dynamic worlds accurately. Then, we propose the Incrementally Modular Abstraction Hierarchy (IMAH) [1] as the most appropriate model. The IMAH consists of the following seven mathematical space levels:

1. A homotopy level
2. A set level
3. A topology level, and a graph theoretical level as a special case
4. An adjunction space level
5. A cellular structured space level
6. A representation model level
7. A projection level

In modeling cyberworlds in cyberspaces, we define general properties of cyberworlds at the higher level and add more specific properties step by step, while moving down IMAH. The properties defined at the homotopy level are invariants of continuous changes of functions. The properties that do not change by continuous modifications in time and space are expressed at this level. At the set theoretical level, the elements of a cyberspace are defined, and a collection of elements constitutes a set with logical calculations. When we define a function in a cyberspace, we need domains that guarantee continuity such that the neighbors are mapped to a nearby place. Therefore, a topology is introduced into a cyberspace through the concept of neighborhood. Cyberworlds are dynamic. Sometimes cyberspaces are attached together, an exclusive union of two cyberspaces where attached areas of two cyberspaces are equivalent. It may happen that an attached space is obtained. These attached spaces can be regarded as a set of equivalent spaces called a quotient space that is another invariant. At the cellular structured level, an inductive dimension is introduced into each cyberspace. At the presentation level, each space is represented in a form which may be imagined before designing cyberworlds. At the view level, the cyberworlds are projected onto view screens.

In IMAH, elements as data are defined at the set level while information corresponding to a metadata schema is defined at the topological space level for the first time.

Next, we propose Formula Expression [9][11] as a finite automaton, which is explained in Section II. Since it expresses symmetry and recursiveness of information with minimum restrictions, it can be considered that general versatility in modeling is higher than with any other data model. In this paper, we focus on a generalization of metadata schema operation to prevent data overlap. In Section III, we first design a metadata centralized space, a data centralized space with Formula Expression, and their interconversion maps using the quotient map and the attaching map [9]. Next, we implement them in Section IV.

We demonstrate them in a simple example of parts ledger management to show their effectiveness in Section V. We reference related work in Section VI, and we conclude in Section VII.

II. The definition of Formula Expression

Formula Expression is a finite automaton defined as follows:

Formula Expression in the alphabet is the result of finite times application of the following (1)-(7).

(1) a ($\in\Sigma$) is Formula Expression
(2) unit element ε is Formula Expression
(3) zero element φ is Formula Expression
(4) when r and s are Formula Expression, addition of $r+s$ is also Formula Expression
(5) when r and s are Formula Expression, multiplication of $r\times s$ is also Formula Expression
(6) when r is Formula Expression, (r) is also Formula Expression
(7) when r is Formula Expression, $\{r\}$ is also Formula Expression

Combination is stronger in (5) than in (4). If there is no confusion, ×, (), {} can be abbreviated. + means disjoint union and is expressed as Σ specifically and × is also expressed as Π.

III. The design of topological spaces and interconversion maps

A. The space design

We design a formula for two topological spaces with a metadata schema by Formula Expression as follows:

1. metadata centralized spaces:

$$\Sigma\ \textit{metadata id}\times(\Sigma\ \textit{data id})$$

where each metadata id is uniquely identified.

2. data centralized spaces:

$$\Sigma\ (\Sigma\ \textit{metadata id})\times \textit{data id}$$

where each data id is uniquely identified.

B. The design of interconversion maps

Next, we design the two interconversion maps f and g between the above spaces using the quotient map and the attaching map [6].

f: Σ *metadata schema id*$\times$(Σ *data id*)
$\rightarrow \Sigma$ (Σ *metadata schema id*)$\times$*data id*
g: Σ (Σ *metadata schema id*)$\times$*data id*
$\rightarrow \Sigma$ *metadata schema id*$\times$(Σ *data id*)

f is onto mapping from a disjoint union of metadata centralized spaces to disjoint union of data centralized spaces attaching equivalent data identifiers, and g is also onto mapping from a disjoint union of data centralized spaces to disjoint union of metadata centralized spaces attaching equivalent metadata identifiers. These designs make the general operation of a metadata schema with data possible. The simple example of map f is shown below.

f (metadata 1×(data 1+data 2+data 3)+metadata 2×(data 1+data 3+data 4)+metadata 3×(data 1+data 2+data 4))
=(metadata 1+metadata 3)×data 1+(metadata 1)×data 2+(metadata 1+metadata 3)×data 3+(metadata 2+metadata 3)×data 4

IV. Implementation

This system is a JAVA application using JDK6. Below is the coding for the interconversion map *f*. Pseudo-code is used for simplicity. The focus is the recursive process (line 7) that is done if a coming numerical calculation is of the type ().

```
Function f (the argument p)
1   term = null; factor = p;
2   while (factor is not null){
3   term = getTerm(factor);
4   while (term is not null & term includes p){
5           factor = getFactor(term)
6           if(factor is of the type ()){
7                   factor = Function f (the contents);
            }
8           newFactor = newFactor×factor;
    }
9   newTerm = newTerm + term;
10  newFormula = newFormula + newTerm;
    }
11  return newFormula;
```

V. A Case Study: parts ledger management

A. Outline

In this section, we take up an example of *parts ledger* management, which is done in most manufacturing companies.

Parts ledgers management with consistency is generally considered to be difficult due to its complexity. The major reasons are: 1. Parts ledgers are managed in different places with different metadata schemas within a company; 2. Parts ledgers often change dynamically during mergers in companies or departmental integration within a company; and 3. Parts codes, which identify each part, are oftentimes different for the same part, because the codes are named differently by suppliers and there are also many inconsistencies in the way data is entered, since parts information is managed in different departments. For these reasons, important information for management, such as information about changes in the total price of a product due to changes in the unit price of a part cannot be outputted promptly by the management system. To avoid this, we arrange parts ledger data using the above design with Formula Expression.

In this case study, we assume that company A and company B have merged, and that their parts ledgers data need to be managed in an integrated way. To do so, we first create a formula for metadata centralized spaces of parts ledgers, and then convert it to a formula for data centralized spaces by the interconversion map *f*. Example data are shown in Figure 1, which is simplified as much as possible without losing generality.

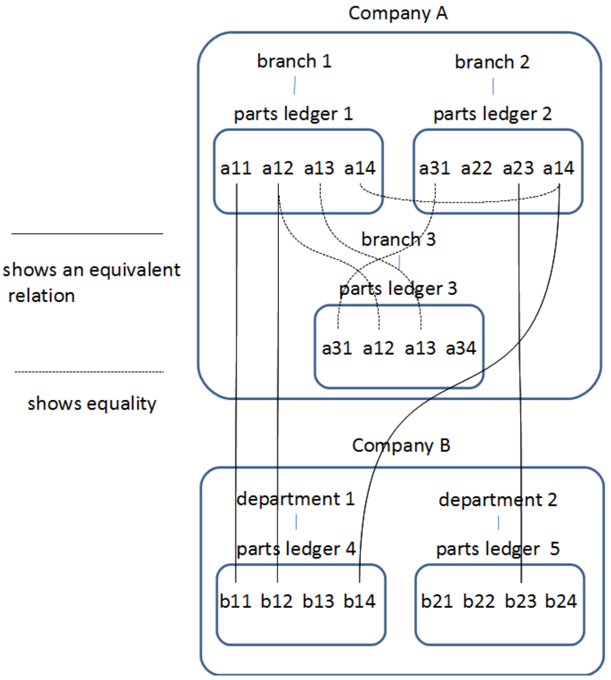

Figure 1. An example of metadata centralized spaces in parts ledger management

B. Input according to the space design

We first create a formula for parts ledgers in company A and B according to the metadata centralized space (III.A) as follows:

Formua1:

company A×(branch 1×parts ledger 1×(a11+a12+a13+a14)+branch 2×parts ledger 2×(a31+a22+a23+a14)+branch 3×parts ledger 3×(a31+a12+a13+a34))+company B×(department 1×parts ledger 4×(b11+b12+b13+b14)+department 2×parts ledger 5×(b21+b22+b23+b24))+company B×(branch 1×parts ledger4×(b11+b12+b13+b14)+branch 2×parts ledger 5×(b21+b22+b23+b24))

Here, identifiers of *company A and B, branch 1~3, department 1~2 and parts ledger 1~5* express *metadata id*, and *a11~a34* and *b11~b24* express *parts id*.

C. Data conversion by the interconversion maps

Next, you convert *Formula 1* to data centralized spaces thorough map *f* and also you attach the image recognizing equivalent relations of *a11 ~ b11, a12 ~ b12, a14 ~ b14* and *a23 ~ b23 as seen in Figure 1*. The result is the formula below:

Formula 2:

{company A×branch 1×parts ledger 1+company B×department 1×parts ledger 4}×{a11+b11}
+{(company A×branch 1×parts ledger 1+company A×branch 3×parts ledger 3)+company B×department 1×parts ledger 4}×{a12+b12}
+(company A×branch 1×parts ledger 1+company A×branch 3×parts ledger 3)×a13
+{(company A×branch 1×parts ledger 1+company A×branch 2×parts ledger 2)+company B×department 1×parts ledger 4}×{a14+b14}
+(company A×branch 2×parts ledger 2)×a22
+{company A×branch 2×parts ledger 2+company B×department 2×parts ledger 5}×{a23+b23}
+(company A×branch 2×parts ledger 2+company A×branch 3×parts ledger 3)×a31
+(company A×branch 3×parts ledger 3)×a34
+(company B×department 1×parts ledger 4)×b13
+(company B×department 2×parts ledger 5)×b21
+(company B×department 2×parts ledger 5)×b22
+(company B×department 2×parts ledger 5)×b24

In the outputted formula, you can know that there is no overlap of parts data, consequently, which ledgers a specified part belongs to accurately. See Figure 2.

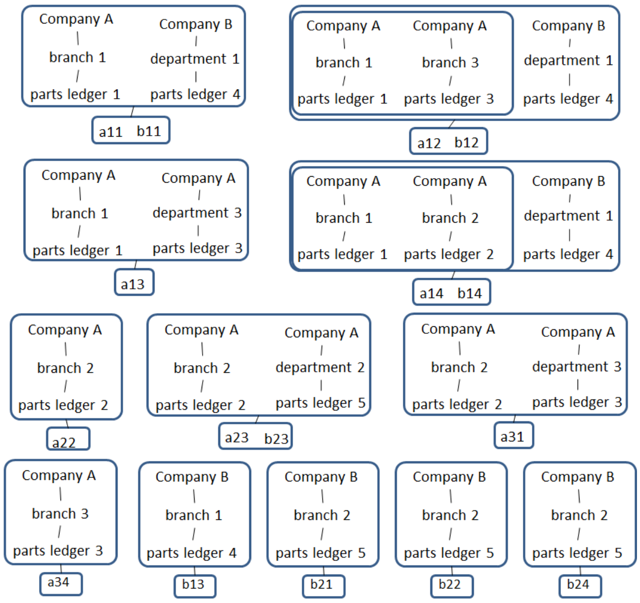

Figure 2. An example of data centralized spaces after the interconversion map in parts ledger management

D. Considerations

As we see in this example, using the above design with Formula Expression, we can say that (1) in data input, you only have to create a formula of spaces, instead of a metadata schema design or data input programs in advance, (2) in data output, you can see metadata schemas from a specified part's data, instead of developing output programs of metadata schemas, and (3) you only have to attach equivalent factors, instead of the design of unified notation. This means that the parts- centered spaces, which include no overlap of parts data, are constructed from the parts ledgers spaces, which include some overlap. In other words, the parts ledger data are arranged from a parts-centric view. The novelty of this function in the system is that data spaces can be reconstructed generally from other points which differ from the initial metadata schema design. Consequently, data overlap can be prevented using the function.

VI. Related work

One of the distinctive features of our research is the attaching function by equivalent relations, which can eliminate data overlap and return it back to the previous state [9]. Such a function, based on the adjunction space level which extends the topological space level, has never before been seen in other research [1]. Another feature is the application of the concept of topological process, which deals with a subset as an element, and that the cellular space extends the topological space, as seen in Section 2. Relational OWL as a method of data and schema representation is useful when representing the schema and data of a database [2][5], but it is limited to representation of an object that has attributes. Our method can represent both objects: one that has attributes as a cellular space and one that does not have them as a set or a topological space. Many works applying other models to XML schema have been done. The motives of most of them are similar to ours. The approach in [8] aims at minimizing document revalidation in an XML schema evolution, based in part on the graph theory. The X-Entity model [9] is an extension of the Entity Relationship (ER) model and converts XML schema to a schema of the ER model. In the approach of [6], the conceptual and logical levels are represented using a standard UML class and the XML represents the physical level. XUML [10] is a conceptual model for XML schema, based on the UML2 standard. This application research concerning XML schema is needed because there are differences in the expression capability of the data model between XML and other models. On the other hand, objects and their relations in XML schema and the above models can be expressed consistently by CDS, which is based on the cellular model. That is because the tree structure, on which the XML model is based, and the graph structure [3][4][7], on which the UML and ER models are based, are special cases of a topological structure mathematically. Entity in the models can be expressed as the formula for a cellular space in CDS. Moreover, the relation between subsets cannot in general be expressed by XML.

VII. Conclusions

In this paper, we designed the metadata schema centralized spaces, the data centralized spaces, and their interconversion maps. And we successfully applied them to parts ledger management, preventing data overlap. We conclude that the attaching function using Formula Expression is effective to model dynamic changing information worlds.

References

[1] T. L. Kunii and H. Kunii, “A Cellular Model for Information Systems on the Web - Integrating Local and Global Information”, In Proc. of DANTE'99, IEEE Computer Society Press, 1999, pp. 19-24.

[2] C. Laborda and S. Conrad, “Bringing Relational Data into the Semantic Web using SPARQL and Relational OWL”, In Proc. of 22nd International Conference On Data Engineering workshop 2006, IEEE Computer Society Press, 2006, pp. 55.

[3] Z. H. Liu, H. J. Chang, and B. Sthanikam, “Efficient Support of XQuery Update Facility in XML Enabled RDBMS”, In Proc. of 2012 IEEE 28th International Conference on Data Engineering (ICDE), IEEE Computer Society Press, 2012, pp. 1394-1404.

[4] J. Zhang, B. Lang and Y. Duan, “An XML Data Placement Strategy for Distributed XML Storage and Parallel Query”, In Proc. Of 12th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT), IEEE Computer Society Press, 2011, pp. 433-439.

[5] H. Zhang, Z. Wang, Z. Gao and W. Li, “Design and Implementation of Mapping Rules from OWL to Relational Database”, In Proc. of 2009 WRI World Congress on Computer Science and Information Engineering, IEEE Computer Society Press, 2009, pp. 71-75.

[6] V. Mascardi, A. Locoro, and P. Rosso, “Automatic Ontology Matching via Upper Ontologies: A Systematic Evaluation”, IEEE Transactions on knowledge and data engineering, IEEE Computer Society Press, no.5, 2010, pp. 609-623.

[7] F. A. Currim, S. A. Currim, C. E. Dyreson, R. T. Snodgrass, S. W. Thomas, and R. Zhang, “Adding Temporal Constraints to XML Schema”, IEEE Transactions on knowledge and data engineering, IEEE Computer Society Press, vol. 24, no. 8, 2012, pp. 1361-1377.

[8] P. Kilpeläinen and R. Tuhkanen, “Towards Efficient Implementation of XML Schema Content Models,”, In Proc. of 2004 ACM Symposium on Document Engineering, ACM Press, 2004, pp. 239-241.

[9] T. Kodama, T. L. Kunii and Y. Seki, “A New Method for Developing Business Applications: The Cellular Data System”, In Proc of CW'06, IEEE Computer Society Press, 2006, pp. 65-74.

[10] K. Ohmori and T. L. Kunii, “Designing and modeling cyberworlds using the incrementally modular abstraction hierarchy based on homotopy theory”, The Visual Computer: International Journal of Computer Graphics, vol. 26, no.5, Springer-Verlag, 2010, pp. 297-309.

[11] T. Kodama, T. L. Kunii, and Y. Seki, “An Example of a Charge Calculation System using the Numerical Value and Exponential Calculation of the Cellular Data System”, Proceedings of the 2011 International Conference on Cyberworlds 2011 (CW2011, Oct.4-6, 2011, Banff, Alberta, Canada), IEEE Computer Society Press, 2011, pp. 31-37.