Reprint: Toshio Kodama, Tosiyasu L. Kunii, and Yoichi Seki, “A Product Control System using the Cellular Data System ”, Proceedings of The 8th WSEAS International Conference on E-ACTIVITIES (E-Learning, E-Communities, E-Commerce, E-Management, E-Marketing, E-Governance, Tele-Working / E-ACTIVITIES '09), Puerto De La Cruz, Canary Islands, Spain, December 14-16, 2009, pp. 57-64.

A Product Control System using the Cellular Data System

Toshio Kodama[1], Tosiyasu L. Kunii[2], Yoichi Seki[3]
[1] CDS Business Dept., Advanced Computer Systems, Inc., Tokyo, Japan
kodama@lab.acs-jp.com, https://www.cellulardatasystem.com/e/index.html
[2] Morpho, Inc., The University of Tokyo, Tokyo, Japan
kunii@ieee.org, kunii@acm.org, http://www.kunii.net/
[3]Software Consultant, Tokyo, Japan
ysekicbl@mail.hinocatv.ne.jp

Abstract: In the era of cloud computing, data is processed within "the cloud", and data and its dependencies between systems or functions progress and change constantly within "the cloud", as user requirements change. Such information worlds are called cyberworlds. Now we need a more powerful mathematical background which can model the cyberworlds in "cloud" as they are. We consider the Incrementally Modular Abstraction Hierarchy (IMAH), with its ability to descend from the most abstract homotopy level to the most specific view level while preserving invariants, to be appropriate to model dynamically changing cyberworlds. We have developed a data processing system called the Cellular Data System (CDS) based on IMAH. In this paper, we introduce a numerical value identifier and processing maps as a function on the presentation level of IMAH. This function is very effective in a business application, in which, in most cases, numerical values defined in information spaces are calculated while data is managed. We have shown its effectiveness through examples of core processing of a product control system in the manufacturing industry using a numerical value identifier.

Key-Words: incrementally modular abstraction hierarchy, formula expression, cellular data system, topological space, product control system, numerical value identifier

1 Introduction

Cyberworlds are information worlds formed in clouds, either intentionally or spontaneously, with or without design. As information worlds, they are either virtual or real, and can be both. In terms of information modeling, the theoretical ground for the cyberworlds is far above the level of integrating spatial database models and temporal database models. They are more complicated and fluid than any other previous worlds in human history, and are constantly evolving. The number of companies that conduct business in cyberspace, such as Google and Microsoft, is increasing and the market is growing remarkably. On the other hand, in general business application systems, as the scale of systems becomes larger and system specifications change more frequently, development and maintenance become more difficult, increasing costs and delays. In some cases, a huge system as the mainstay system in a large company, where the number of program steps is hundreds of millions, needs several years to develop, while increases in development and maintenance costs squeeze management. Such situations occur because combinatorial explosions arise. The era of cloud computing requires a more powerful mathematical background to model the cyberworlds and to prevent combinatorial explosions. In the cyberworlds, every business object and also business logic should be expressed in a unified form to prevent discontinuity between systems or functions and to meet changes in user requirements. The needed mathematical mechanisms are considered to be as follows: 1.disjoint union of spaces by an equivalence relation; 2. changes in spaces to guarantee preservation of invariants; 3. attachment of different spaces by an equivalence relation; 4.a space with dimensions as a special case. We consider the Incrementally Modular Abstraction Hierarchy (IMAH) that one of authors (T. L. Kunii) proposes to be able to satisfy the above requirements, as it models the architecture and the dynamic changes of cyberworlds from a general level (the homotopy level) to a specific one (the view level), preserving invariants while preventing combinatorial explosion [1]. It also benefits the reuse of information, guaranteeing modularity of information based on the mechanism of disjoint union. Unlike IMAH, other leading data models do not support the disjoint union or the attaching function by equivalence relation.

In this research, one of the authors (Y. Seki) proposed a finite automaton called Formula Expression as a development tool to realize IMAH. Another of the authors (T. Kodama) has actually designed spaces and implemented a data processing system called the Cellular Data System (CDS) using Formula Expression. In this paper, we put emphasis on practical use by taking up some examples. First, we have designed a useful numerical value identifier to put a numerical value in Formula Expression as a function on the presentation level and implemented it. We have demonstrated the effectiveness of CDS by developing a general business application system of a product control system and abbreviating the process of implementing most application programs.

2 IMAH and Formula Expression

2.1 Incrementally Modular Abstraction Hierarchy

The following list constitutes the Incrementally Modular Abstraction Hierarchy to be used for defining the architecture of cyberworlds and their modeling:

1. the homotopy (including fiber bundles) level
2. the set theoretical level
3. the topological space level
4. the adjunction space level
5. the cellular space level
6. the presentation (including geometry) level
7. the view (also called 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 climbing down the incrementally modular abstraction hierarchy. 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 operations. 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.

2.2 The definition of Formula Expression

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

(1) a ($a \in \Sigma$) 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×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

Strength of combination is the order of (4) and (5). If there is no confusion, ×, (), { } can be abbreviated. + means disjoint union and is expressed as $\uplus$ specifically and × is also expressed as Π. In short, you can say " a formula consists of an addition of terms, a term consists of a multiplication of factors, and if the () or { } bracket is added to a formula, it becomes recursively the factor". In Formula Expression, five maps (the expansion map, the bind map, the division map, the attachment map, the homotopy preservation map) are defined [9].

3 A numerical value identifier on the presentation level

3.1 The properties of a numerical value identifier

If we assume that *p, q, r* are arbitrary numerical factors, and that *s, t, u* are arbitrary letter factors, the numerical value identifier has the following properties:

(1) 1 =ε
(2) s×1 = s
(3) s×0 = ε
(4) s×p+s×q = u×$(p+q)$

(5) $s \times p \times t \times q = s \times t \times (p*q)$
(6) $s \times p = p \times s$
(7) $s \times p\ (t \times q + u \times r) = s \times t \times (p*q) + s \times u \times (p*r)$
(8) $(s \times p + t \times q) u \times r = s \times u \times (p*r) + t \times u \times (q*r)$

An example of the processing of the numerical value identifier is shown below.
cat+dog+rabbit+dog+cat+rabbit+dog+rabbit+mouse = cat×*2*+dog×*2*+rabbit×*3*+mouse×*1*

3.2 A numerical value identifier calculation map *f*

A numerical value identifier calculation map *f* is defined based on the above-mentioned properties. If you assume the entire set of a formula, including the numerical value identifiers, to be A, f: A → A and f is the following:

f: $\varepsilon \rightarrow 1$
f: $u \rightarrow u \times 1$
f: $u \times p + u \times q \rightarrow u \times (p+q)$
f: $u \times p \times v \times q \rightarrow u \times v \times (p*q)$
f: $s \times p\ (t \times q + u \times r) \rightarrow s \times t \times (p*q) + s \times u \times (p*r)$
f: $(s \times p + t \times q) u \times r \rightarrow s \times u \times (p*r) + t \times u \times (q*r)$
f: $u \times p + v \times q \rightarrow u \times p + v \times q$

And if we assume that T is an arbitrary term, and that E is an arbitrary formula, f is:

$f(T*T) = f(T)*f(T)$
$f((E)) = (f(E))$

Next let *s, t, u* be arbitrary terms. A graph decomposition map *g*, which decomposes the term of a directed graph, is defined as follows:

g: $s(t+u+\ldots) \rightarrow t+u+\ldots$
g: $s \rightarrow s$ (except the above case)

3.3 Implementation

This system is a web application developed using JSP and Tomcat 5.0 as a Web server. The client and the server are the same machine. (OS: Windows XP; CPU: Intel Pentium 3, 1.2GHz; RAM: 1.1Gbyte; HD: 20GB) The following is the coding for the calculation of a numerical value identifier. The focus is the recursive process (line 7, in bold) that is done if a coming numerical value identifier is of the type (). The explanation is abbreviated due to space limitations.

```
1  term = null; factor = null;
2  while(factor is not null){
3    term = getTerm(factor);
4    while(term is not null){
5          factor = getFactor(term)
6                if(factor is of the type ()){
7                factor = calculate(the contents);
           }
8          factor = getNumericalFactor(factor);
9          LetteFactor = getLetteFactor(factor);
10         newNF = newNF×NumericalFactor;
11         newLF = newLF×LetteFactor;
        }
12    newTerm = newNF + newLF;
13    newFormula = newFormula + newTerm;
    }
14 return newFormula;
  }
```

4 Development of a Product Control System

4.1 Outline

We take up an example of a production control system to secure generality because they are generally developed in most industries. The most important thing in production control is to make a plan for production and next, to make a plan for procurement of materials from the production plan. A product is assembled from many kinds of materials and semi-products, which form the whole-part hierarchy. If there are too many kinds of products or semi-products, or the architecture of assembly becomes complicated or changes frequently, the development of the production control system and its maintenance cost a lot because of its complexity, and use of the system is likely confusing. To solve the difficulties, we apply CDS to the development of core processing of the product control system. An object for a material in stock is designed as a topological space, where terms expressing materials form disjoint union, and an object for a product which is assembly of materials or semi-products is designed as a directed graph (a special case of a topological space), where each node which expresses the name of a material and each edge which expresses a quantity of a material are merged to create another node recursively. If you make each object according to the design and use the functions of CDS, you can make a plan for procurement based on the plan for production quite easily, even if the kinds of materials and semi-products increase significantly or the architecture of assembly of materials becomes more complicated or changes. In these designs, a numerical value identifier and the calculation map are used to express quantities of materials, semi-products and products and the calculation between them. Here, actual data and functions are simplified to focus on verifying development of core processing without losing generality.

4.2 The design of topological space

We design a formula for topological spaces for (1) the stock of products, semi-products and materials, (2) the estimated demand for products, and (3) products and semi-products as assembly of semi-products or materials. Numerical value identifiers are used to express each quantity. The formulas for (1), (2) are designed as follows (Figure 4.2-1, Figure 4.2-2):

(1) $\mathrm{Stock}(\Sigma(material_i \times price_i \times m))$
(2) $\Sigma(\mathrm{Estimate}(\Sigma e_i \times \text{-}p))$
Here, each $material_i$ is a factor which expresses the name of a material and $price_i$ is a factor which expresses the unit price of $material_i$ and e_i are terms which are designed in (3).

The formula for (3) is designed as follows (Figure 4.2-3):

(3) $e_{m,1}(e_{n,1}(e_{o,1}(\ldots e_{i,j}(\Sigma material_i \times price_i \times j) \times o) \times p + e_{o,2}(\ldots e_{j,2}(\Sigma material_i \times price_i \times j) \times s) \times t) \times u + e_{n,2}(\ldots) \times v + \ldots + e_{n,i}(\ldots) \times w) \times x + e_{m,2}(\ldots) \times y$

Here, $e_{i,j}$ is a factor which expresses the name of an estimate for a semi-product$_{i,j}$ or a product$_{i,j}$. *l, m, n, o,…* are factors which express zero or positive integers.

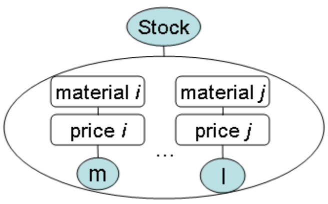

Figure 4.2-1 The topological spaces for a stock

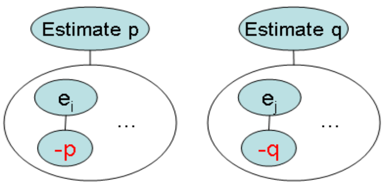

Figure 4.2-2 The topological spaces for demand estimate

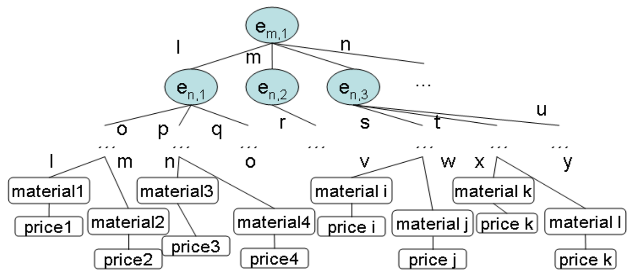

Figure 4.2-3 The topological space for a estimate

4.3 Data input according to the design -Determining the structures of semi-products and products-

First, the structures of the estimates of semi-products and products are determined. Assume that there are four kinds of materials: a material$_1$ whose unit price is \$100, a material$_2$ whose unit price is \$200, a material$_3$ whose unit price is \$300, and a material$_4$ whose unit price is \$400; there are three kinds of semi-products: a semi-product$_1$ which consists of six material$_1$ and four material$_2$; a semi-product$_2$ which consists of seven material$_1$, two material$_3$ and ten material$_4$; and a semi-product$_3$ which consists of three material$_2$ and five material$_3$. First you create the formula for the topological space for semi-product$_1$, semi-product$_2$, and semi-product$_3$ according to the above design (3) as follows:

$e_{2,1}(material_1 \times \$100 \times 6 + material_2 \times \$200 \times 4)$
$e_{2,2}(material_1 \times \$100 \times 7 + material_3 \times \$300 \times 2 + material_4 \times \$400 \times 10)$
$e_{2,3}(material_2 \times \$200 \times 3 + material_3 \times \$300 \times 5)$

In the same way, assume that there are two kinds of products: a product$_1$ which consists of three semi-product$_1$, one semi-product$_2$, and five material$_2$;

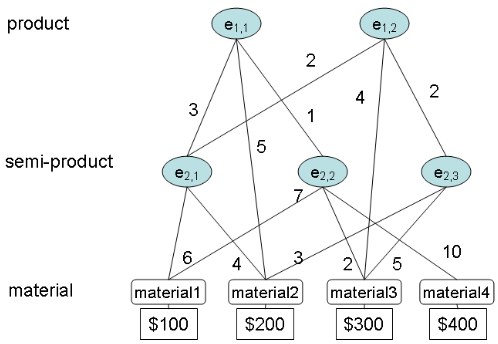

Figure 4.3-1 Assigning the compositions of the estimates of semi-products and products

a product$_2$ which consists of two semi-product$_1$, two semi-product$_3$, and four material$_3$. You create the formula for the topological space for the product$_1$ and the product$_2$ according to the design as follows (Figure 4.3-1):

$e_{1,1}(e_{2,1}(material_1\times\$100\times6+material_2\times\$200\times4)\times3+e_{2,2}(material_1\times\$100\times7+material_3\times\$300\times2+material_4\times\$400\times10)\times1+material_2\times\$200\times5)$
$e_{1,2}(e_{2,1}(material_1\times\$100\times6+material_2\times\$200\times4)\times2+e_{2,3}(material_2\times\$200\times3+material_3\times\$300\times5)\times2+material_3\times\$300\times4)$

-Determining stock-
Next, the kind and the quantity of stock are determined. We assume that there are 10 material$_1$ whose unit price is \$100, 20 material$_2$ whose unit price is \$200, 30 material$_3$ whose unit price is \$300, and 40 material$_4$ whose unit price is \$400 in stock. You create the following formula for the topological space according to the design (1) in 4.2 and input it into data storage (Figure 4.3-2).

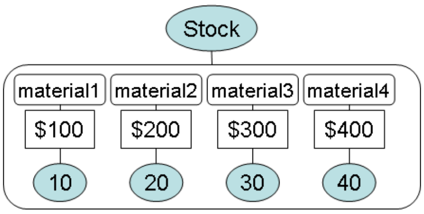

Figure 4.3-2 The topological space of stock

formula 4.3-1:
$Stock(material_1\times\$100\times10+material_2\times\$200\times20+material_3\times\$300\times30+material_4\times\$400\times40)$

-Data input for estimates-
Next, assume that demand is estimated to be 5 product$_1$ and 10 product$_2$ in estimate1; and 5 product$_1$ and 5 product$_2$ in estimate2. You create the following formula for the topological space according to the design (2) in 4.2 and add it into data storage (Figure 4.3-3)

formula 4.3-2:
(formula4.3-1)$+Estimate1(e_{1,1}(e_{2,1}(material_1\times\$100\times6+material_2\times\$200\times4)\times3+e_{2,2}(material_1\times\$100\times7+material_3\times\$300\times2+material_4\times\$400\times10)\times1+material_2\times\$200\times5)\times\mathbf{-5}+e_{1,2}(e_{2,1}(material_1\times\$100\times6+material_2\times\$200\times4)\times2+e_{2,3}(material_2\times\$200\times3+material_3\times\$300\times5)\times2+material_3\times\$300\times4)\times\mathbf{-10})+Estimate2(e_{1,1}(e_{2,1}$

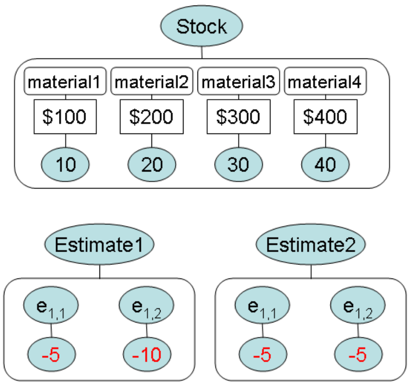

Figure 4.3-3 Disjoint union of spaces of the stock and the demand estimate

$(material_1\times\$100\times6+material_2\times\$200\times4)\times3+e_{2,2}(material_1\times\$100\times7+material_3\times\$300\times2+material_4\times\$400\times10)\times1+material_2\times\$200\times5)\times\mathbf{-5}+e_{1,2}(e_{2,1}(material_1\times\$100\times6+material_2\times\$200\times4)\times2+e_{2,3}(material_2\times\$200\times3+material_3\times\$300\times5)\times2+material_3\times\$300\times4)\times\mathbf{-5})$

Here, each numerical value identifier of $e_{1,1}$ and $e_{1,2}$ takes a *minus* value because the formula for "Estimate" is added to the formula for "Stock (<->Estimate)".

4.4 Data output

-Making a plan for production and procurement-
To make a plan for production and procurement, first you attach the three topological spaces in formula 4.3-3 by the factors "Stock", "Estimate1" and "Estimate2" through the adjunction map *h* [10], and you calculate it using the numerical value identifier calculation map *f* (calculation 1). The outputted formula is the following (Figure 4.4-1):

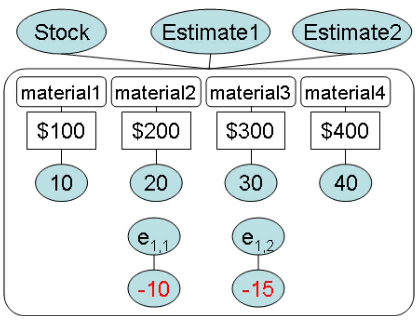

Figure 4.4-1 The adjunction space after numerical value identifier calculation 1

formula 4.4-1:

$$(Stock+Estimate1+Estimate2)(material_1\times\$100\times10+material_2\times\$200\times20+material_3\times\$300\times30+material_4\times\$400\times40+e_{1,1}(e_{2,1}(material_1\times\$100\times6+material_2\times\$200\times4)\times3+e_{2,2}(material_1\times\$100\times7+material_3\times\$300\times2+material_4\times\$400\times10)\times1+material_2\times\$200\times5)\times\mathbf{-10}+e_{1,2}(e_{2,1}(material_1\times\$100\times6+material_2\times\$200\times4)\times2+e_{2,3}(material_2\times\$200\times3+material_3\times\$300\times5)\times2+material_3\times\$300\times4)\times\mathbf{-15})$$

From the result, you can knou that there is a shortage of 10 product1 and 15 product2. Next, you break down the shortage of the products. You use the graph decomposition map g on $product_1$ and $product_2$, which are short in formula 4.4-1, and calculate the result using the numerical value identifier calculation map f (calculation2). The outputted formula is the following (Figure 4.4-2):

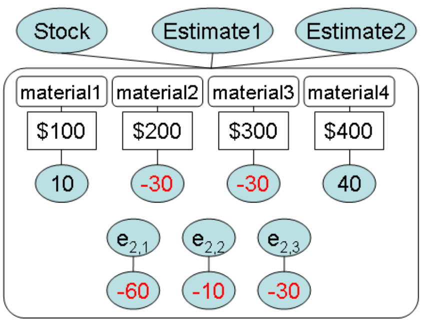

Figure 4.4-2 The adjunction space after numerical value identifier calculation 2

formula 4.4-2:

$$(Stock+Estimate1+Estimate2)(material_1\times\$100\times10+material_2\times\$200\times\boldsymbol{-30}+material_3\times\$300\times\boldsymbol{-30}+material_4\times\$400\times40+e_{2,1}(material_1\times\$100\times6+material_2\times\$200\times4)\times\boldsymbol{-60}+e_{2,2}(material_1\times\$100\times7+material_3\times\$300\times2+material_4\times\$400\times10)\times\boldsymbol{-10}+e_{2,3}(material_2\times\$200\times3+material_3\times\$300\times5)\times\boldsymbol{-30})$$

You can know that you are short 30 $material_2$, 30 $material_3$, 60 $semi\text{-}product_1$, 10 $semi\text{-}product_2$, and 30 $semi\text{-}product_3$. In the same way, you break down the shortage of semi-products. You use the map g about the semi-products and calculate the result by the map f (calculation3). The outputted formula is the following (Figure 4.4-3):

formula 4.4-2:

$$(Stock+Estimate1+Estimate2)(material_1\times\$100\times\boldsymbol{-420}+material_2\times\$200\times\boldsymbol{-360}+material_3\times\$300\times\boldsymbol{-140}+material_4\times\$400\times\boldsymbol{-60})$$

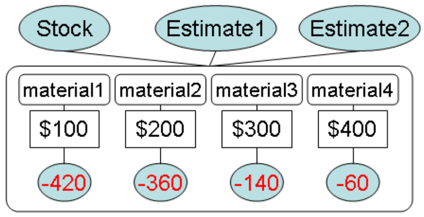

Figure 4.4-3 The adjunction space after numerical value identifier calculation 3

You can know that there is shortage of 420 $material_1$, 360 $material_2$, 140 $material_3$ and 60 $material_4$.

Next, when you want to calculate the procurement cost of materials, you can calculate it from the above formula if you use the maps of CDS [9] as follows:

$$(Stock+Estimate1+Estimate2)(material_1\times\$100\times-420+material_2\times\$200\times-360+material_3\times\$300\times-140+material_4\times\$400\times-60)$$
$$\to (Stock+Estimate1+Estimate2)\{material_1+material_2+material_3+material_4\}\{\$100\times-420+\$200\times-360+\$300\times-140+\$400\times-60\}$$
$$\to (Stock+Estimate1+Estimate2)(material_1+material_2+material_3+material_4)\times\boldsymbol{-\$192,000}$$

You can know that the procurement cost of all of the materials is “\$192,000”. In the same way, for example, when you want to know the cost for $material_1$ in Estimate1 from formula 4.3-2, it is possible if you use the maps of CDS as follows:

$$j(i(h(formula4.3\text{-}2, \text{“}Estimate1\times material_1\text{”}), material_1))$$
$$= j(i(Estimate1(e_{1,1}(e_{2,1}(material_1\times\$100\times6)\times3+e_{2,2}(material_1\times\$100\times7)\times1)\times-5+e_{1,2}(e_{2,1}(material_1\times\$100\times6)\times2)\times-10)))$$
$$=j(Estimate1\{e_{1,1}\{e_{2,1}+e_{2,2}\}+e_{1,2}\times e_{2,1}\}material_1\{(\$100\times6\times3+\$100\times7\times1)\times-5+(\$100\times6\times2)\times-10\})$$
$$=Estimate1\{e_{1,1}\{e_{2,1}+e_{2,2}\}+e_{1,2}\times e_{2,1}\}material_1\times\boldsymbol{-\$24,500}$$

h: the condition formula processing map
i: the attaching map
j: the unit formula processing map

You can know that the cost of the materials in Estimate1 is “\$24,500”.

Similarly, from a series of outputted results, you can make a plan to produce semi-products and products as well as a plan to procure materials to meet the estimated demand for products.

4.5 Considerations

In general, it costs a lot to develop and maintain application programs for the product control system because of the complexity of and frequent changes in the data structure. This example shows that you only have to design formulas for products, stock, and estimated demand such as (1), (2), (3) in 4.2. and use the maps, thereby reducing the amount of application development and maintenance. This is mainly because:

1. The data structures of the stock, the estimated demand and the products can be expressed as they are coherently by formulas.

Therefore, even if the structure of a product changes, you only have to modify the changed part and not the whole system; and

2. The quantities of materials, semi-products and the products can be modeled on formulas using the numerical value identifier and they can be calculated by the maps.

Therefore even if kinds of materials or semi-products increase significantly, you won't have to modify application programs.

5 Related works

The distinctive features of our research are the application of the concept of topological processing, which deals with a subset as an element, and that the cellular space extends the topological space, as seen in Section 2. The conceptual model in [2] is based on an ER model and is a model where tree structure is applied. The approach in [3] aims at grouping data in a graph structure where each node has attributes. The ER model, graph structure and tree structure are expressed as special cases of topological space, and a node with attributes is expressed as one case of the cellular space, so these models are included in the function of CDS. Many works dealing with XML schema have been done. The approach in [4][5] aims at introducing simple formalism into XML schema definition for its complexity. An equivalence relation of elements is supported in CDS, so that complexity and redundancy in schema definition are avoided if CDS is employed, and a homotopy preservation function is introduced into CDS in the model for preserving information. As a result, the problems described in [4][5] do not need to be considered in CDS.

Some research using inductive data processing has been done recently. CDS also can be considered as one of those inductive systems. The goal of research on the inductive database system in [7] is to develop a methodology for integrating a wide range of knowledge generation operators with a relational database and a knowledge base. The main achievement in [8] is a new inductive query language extending SQL, with the goal of supporting the whole knowledge discovery process, from pre-processing via data mining to post-processing. If you use the methods in [7], [8], attributes according to users' interests have to be designed in advance. Therefore it is difficult to cope with changes in users' interests. If you use CDS, a formula for a topological space without an attribute as a dimension in database design can be created so that the attributes of objects don't need to be designed in advance.

Plenty of CASE tools are currently available, but they are effective for data structure that is already defined. The differences from CDS are mainly that we apply a novel model, IMAH, for building CDS, and that CDS not only visualizes objects, but can also model business logic using Formula Expression, so that, if spaces are designed, they function immediately.

6 Conclusions

We have developed a data processing system called the Cellular Data System (CDS) based on IMAH. In this paper, we added a numerical value identifier and processing maps on the presentation level to the functions of CDS. Using this function, you can insert numerical values as they are in designing formulas for business applications, and in so doing you can make system development simpler. In other words, if you take advantage of this function, the method of development changes to visually designing spaces that express business objects. It means that the quality of a system depends on the quality of the designed spaces. In business application development, use of CDS will allow developers to be creative and enable them to reuse the application functions modularly, thereby preventing troubles between the customer side and the supplier side and combinatorial explosions.

This research is still in its infancy, but it is progressing every day, and we are sure that CDS has possibilities to bring great social impact in the era of cloud computing.

References:

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

[2] Miguel I. Aguirre-Urreta, George M. Marakas, “Comparing conceptual modeling techniques: a critical review of the EER vs. OO empirical literature”, ACM SIG Database, Vol.32, Issue2, pp.9-32, ACM press, New York, U.S.A., 2008.

[3] Lionello Pogliani, “Modeling molecular polarizabilities with graph-theoretical concepts”, Journal of Computational Methods in Sciences and Engineering, pp.737-751, IOS Press, Amsterdam, The Netherlands, 2004.

[4] Wim Martens, Frank Neven, Thomas Schwentick, Geert Jan Bex, “Simple off the shelf abstractions for XML schema”, ACM SIGMOD Record, pp.15-22, ACM press, New York, U.S.A., 2007.

[5] Wim Martens, Frank Neven, Thomas Schwentick , “Inferring XML schema definitions from XML data”, Proceedings of the 33rd international conference on Very large data bases 2007, Vienna, Austria, pp.998-1009, VLDB Endowment, 2007.

[6] Toshio Kodama, Tosiyasu L. Kunii, Yoichi Seki, “A New Method for Developing Business Applications: The Cellular Data System”, In Proceedings of Cyberworlds 2006, Lausanne, Switzerland, pp.64-74, IEEE Computer Society Press, Los Alamitos, California U.S.A., 2006.

[7] Hong-Cheu Liu, Jeffrey Xu Yu, John Zeleznikow, Ying Guan, “A Logic-Based Approach to Mining Inductive Databases”, Proceedings of the 7th international conference on Computational Science, Part I: ICCS 2007, Lecture Notes In Computer Science; Vol.4487, pp.270-277, Springer-Verlag, Berlin, Heidelberg, 2007.

[8] Lothar Richter, Jörg Wicker, Kristina Kessler, Stefan Kramer, “An inductive database and query language in the relational model”, Proceedings of the 11th international conference on Extending database technology, pp.740-744, ACM press, New York, USA, 2008.

[9] Toshio Kodama, Tosiyasu L. Kunii and Yoichi Seki, “WWW Busines Application Development based on the Cellular Model”, Journal of Computer Science and Technology, pp.176-187, Springer-Verlag, Berlin, Heidelberg, 2007.

[10] Tosiyasu L. Kunii, “Discovering Cyberworlds”, in Special Issue on Vision 2000, IEEE Computer Graphics and Applications, pp. 64-65, January/February 2000, IEEE Computer Society Press, Los Alamitos, California U.S.A., 2000.