The Development of Core Logic for an Estimate System using the Cellular Data System

Toshio Kodama  
CDS Project  
Maeda Corporation  
Tokyo, Japan  
kodama.ts@jcity.maeda.co.jp  
kodama@lab.acs-jp.com

Tosiyasu L. Kunii  
Morpho Inc  
Tokyo, Japan  
kunii@ieee.org

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

Abstract— In the era of cloud computing, data is processed in a “Cloud”, and data and its dependencies between systems or functions progress and change constantly within the cloud, as a user’s requirements change. Such information worlds are called cyberworlds. We need a more powerful mathematical background which can model the cyberworlds in the “cloud” as they are. We consider the Incrementally Modular Abstraction Hierarchy (IMAH) to be appropriate for modeling the dynamically changing cyberworlds by descending from the most abstract homotopy level to the most specific view level, while preserving invariants. We have developed a data processing system called the Cellular Data System (CDS) based on IMAH. In this paper, we introduce into CDS the applied functions for calculating numerical values, and apply CDS to develop the core logic of an estimate system, used in most industries but costly due to its complexity, to verify its effectiveness.

Keywords-component; incrementally modular abstraction hierarchy, formula expression, estimate system, numerical value calculation, exponential calculation

I. INTRODUCTION

Cyberworlds are information worlds formed in a cloud 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 eBay is increasing and the market is growing remarkably. On the other hand, in general business application system, as the scale of systems becomes larger and the specifications of systems changes more frequently, development and maintenance becomes more difficult, leading to higher costs and delays. In some cases, a huge system as the mainstay system in a large company, where the number of program steps is the hundreds of millions, needs several years to develop. Increases in development and maintenance cost squeeze management. Such situations arise because of combinatorial explosions. The era of cloud computing requires a more powerful mathematical background to model the cyberworlds and to prevent combinatorial explosions. In the cloud, every business object and business logic should be expressed in a unified form to eliminate discontinuity between systems or functions and to meet changes in user requirements. The needed mathematical mechanisms are: 1. disjoint union of spaces by an equivalence relation; 2. change in spaces to preserve invariants; 3. attachment of different spaces by an equivalence relation; 4. space with dimensions as a special case. We consider the Incrementally Modular Abstraction Hierarchy (IMAH) that one of authors (T. L. Kunii) proposes 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. One of authors (T. Kodama) has actually designed spaces and implemented the 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 design numerical value and exponential calculations to put numerical values and exponential identifiers in Formula Expression as a function on the presentation level and implemented it. We demonstrate the effectiveness of CDS by developing a general business application system of an estimate system and abbreviate the process of implementing application programs.

II. IMAH AND FORMULA EXPRESSION

A. The Incrementally Modular Abstraction Hierarchy

The following list is 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 (or attaching 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 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 may happen that an attached space is obtained. These attached spaces can be regarded as a set of equivalent spaces expressed by equivalence relations of the user’s request and the relation of the status of the seats (Phase 2), a quotient space by an equivalence relation of the status of the seats (Phase 1), a quotient space by an equivalence relation of neighborhoods. The attachment map) are defined [9].

A simple example of topological modeling of train seat reservations according to IMAH is shown in Figure 1. The steps include the creation of a topological space for a user and train seats (Phase 1), a quotient space by an equivalence relation of the status of the seats (Phase 2), a quotient space that satisfies a condition formula from a formula. When there is no confusion, ×, ( ), { } can be abbreviated. + means disjoint union and is expressed as Σ 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^w. In Formula Expression, four maps (the expansion map, the bind map, the division map, the attachment map) are defined [9].

**B. 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∈Σ) 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 rs 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 Σ 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^w. In Formula Expression, four maps (the expansion map, the bind map, the division map, the attachment map) are defined [9].

**C. The Grammar that generates Formula Expression**

The grammar that generates Formula Expression is defined as follows:

We assume that Σ₀ is a set of ideograms and its element is w (w∈Σ₀).

G = (|Σ₀|, Σ₁, Σ₂, Σ₃), P = {E|T,F}, E = |{E|T,F, id, ε, +, ×, (, ), {, }}, P, E),
P = {E|T,F, id, ε, +, ×, (, ), {, }}, P, E),

Here, E is called a formula, and T is called a term, and F is called a factor, and id is called an identifier; + is called an addition operation, × is called a multiplication operation, ( ) is called a 1st bracket, and { } is called a 2nd bracket. If you paraphrase the above-mentioned grammar, you can say “a formula consists of an addition of terms, a term consists of a multiplication of factors, and if the 1st or 2nd bracket is added to a formula, it becomes recursively the factor^w. And when a term is a component of a formula, we say that the formula has the term. And when a factor of the bracket that includes a term is a component of a formula, we say that the formula includes the term. It is the same with a term and a factor. An example is shown below.

The term "a(b+c)(d+e)" has factors "a", "(b+c)", "(d+e)", and includes factors "b", "c", "d", "e".

**D. A Conditional Formula Search**

A function for specifying conditions defining a condition formula by Formula Expression is supported in CDS. This is one of the main functions, and the map is called a condition formula processing map. A formula created from these is called a condition formula. "!" is a special factor which means negation. Recursivity by ( ) in Formula Expression is supported, so that the recursive search condition of a user is expressed by a condition formula. The condition formula processing map f is a map that gets a disjoint union of terms that satisfies a condition formula from a formula. When
condition formula processing is considered, the concept of a remainder of spaces is inevitable. A remainder acquisition condition formula processing is considered, the concept of a

A condition formula: “horse+zebra”

A simple example is shown below.

III. NUMERICAL VALUE CALCULATION AND EXPONENTIAL CALCULATION AND APPLIED MAPS

A. The properties of numerical value calculation and exponential calculation

A. The properties of numerical value calculation and exponential calculation

And the exponential calculation in Formula Expression has the following properties:

(1) \( s^0 = 1 \)
(2) \( s^1 = s \)
(3) \( s^s = s^q \)
(4) \( s^{sx^s} = s^{0^{+q}} \)
(5) \( s^{s^y} = s^{x^q} \)

B. The numerical value and exponential calculation map

The numerical value and exponential calculation map \( f \) is defined based on the above mentioned properties. If you assume the entire set of formulas, including the numerical factors, to be \( A, f: A \rightarrow A \) and \( f \) is the followings:

\[ f: s \rightarrow sxI \]
\[ f: ps \rightarrow sxp \]
\[ f: sxp+sxq \rightarrow sx(x(p+q)) \]
\[ f: sxp×sxq \rightarrow sx(x(p×q)) \]
\[ f: sxp + sxq \rightarrow sx(x(p+q)) + sx(x(q+q)) \]
\[ f: sxp×sxq \rightarrow sx(x(p×q)) + sx(x(q×q)) \]
\[ f: uxp+v×q \rightarrow uxp+v×q \]
\[ f: s^0 \rightarrow \epsilon \]
\[ f: s^1 \rightarrow s \]
\[ f: (\text{expression}) \rightarrow y^{q+q} \]

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

\[ f(T+T) = f(T)+f(T) \]
\[ f(T+T) = f(T)+f(T) \]
\[ f((E)) = (f(E)) \]

C. Other applied maps

-A mask calculation map-

You can extract the specified factors from a formula through a mask calculation map. A simple example is shown below.

\[ f_m \left( \text{"y100(30pieces+20m)"}, \text{"[0...9]"} \right) \]
\[ = 100(30+20) \]

where \([0...9] \) stands for the factor of every numerical value.

-A replacement map-

A simple example is shown below.

A simple example is shown below.
You can replace the specified factors with others in the formula through a replacement map. A simple example is shown below.

\[ f(\text{"a(b+c)d"}, \text{"a=A"}) = a(b+c,d) \]

D. 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, 2.1GHz; RAM: 2.1Gbyte; HD: 20GB) The following is the coding for the calculation of numerical value and exponential calculation. The focus is the recursive following is the coding for the calculation of numerical value

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

IV. DEVELOPMENT OF CORE LOGIC OF AN ESTIMATE SYSTEM

A. Outline

We take up the example of the core logic development of an estimate system to secure generality, because the systems are needed in most industries. The data architecture is a recursive directed graph, where each node has attributes and estimate spaces are accumulated in each level recursively. Here, actual data and functions are simplified to focus on verifying development of core processing without losing generality.

Firstly, a formula for an estimate space is designed as a topological space and basic processing for data output using the maps is designed. In these designs, numerical value calculation, exponential calculation, and the applied maps are used to express unit costs, the amount of resources, and the calculations among them. Secondly, a formula is created for data input according to the space design and added to previous formulas, forming a disjoint union of topological spaces. Thirdly, the cost of an estimate and resources can be calculated using the maps of CDS.

B. The design of topological spaces

Assume that \( m \) and \( n \) are positive integers, that the topological space of an estimate is \( est_{m,n} \) and that the topological space of the estimated cost is \( cost_{m,n} \). \( m \) counts a kind and \( n \) counts a layer of an estimate. \( est_{m,n} \) has properties of an estimate name \( name_{m,n} \) and an estimate amount \( a_{m,n} \). If you assume that arbitrary subsets of \( E \) and their amounts are \( \sum est_{m,n} \), \( \sum a_{m,n} \) respectively, we design the topological spaces of \( est_{m,n} \) as follows.

\[
est_{m+1,n} = name_{m+1,n} ( \sum est_{m,n} \times \sum a_{m,n} ) \times unit_{m+1,n}
\]

\[
cost_{m+1,n} = f(g(est_{m+1,n})^2 + \sum unit_{m+1,n})
\]

where \( est_{m,n} \), \( cost_{m,n} \)

\[
= (\sum (\sum est_{m,n} \times \sum a_{m,n} )) \times unit_{m+1,n}
\]

\[
g: the mask calculation map (III.B)
\]

They use to express unit costs, the amount of resources, and the maps is designed. In these designs, numerical value calculation, exponential calculation, and the applied maps are used.

```
A(b+c)d
```

```
1. newLF = newLF + LetteFactor;
2. newNF = newNF + NumericalFactor;
3. newFormula = newFormula + newNF + newLF;
```

Figure 3. The image of the topological space of an estimate

C. Data input according to the design

Assume that unit data of material and labor are as in TABLE I. First you create the formulas for the topological spaces for material and labor as initial conditions according to the above design and make the disjoint union \( \sum est_{m,n} \) of them.

```
<table>
<thead>
<tr>
<th>Item</th>
<th>Unit</th>
<th>Unit Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>material_1</td>
<td>m^2</td>
<td>¥1,000</td>
</tr>
<tr>
<td>material_2</td>
<td>100kg</td>
<td>¥200</td>
</tr>
<tr>
<td>material_3</td>
<td>piece</td>
<td>¥2,000</td>
</tr>
<tr>
<td>labor_1</td>
<td>day</td>
<td>¥120</td>
</tr>
<tr>
<td>labor_2</td>
<td>hour</td>
<td>¥2,000</td>
</tr>
</tbody>
</table>
```

C-1:

```
material_1 \times ¥1000/m^2 \times material_2 \times ¥2000/100kg \times material_3 \times ¥500/m \times labor_1 \times ¥120/day \times labor_2 \times ¥2000/hour
```

Next the estimate part_1 (the unit is “piece”) is made from 10m^2 of material_1, 900kg of material_2 and five days of labor_1. According to the space design, you create \( est_{2,1} \) and add it to the formula.

```
formula C-2:
```

```
material_1 \times ¥1000/m^2 \times material_2 \times ¥2000/100kg \times material_3 \times ¥500/m \times labor_1 \times ¥120/day \times labor_2 \times ¥2000/hour
```

Next the estimate part_1 (the unit is “piece”) is made from 10m^2 of material_1, 900kg of material_2 and five days of labor_1. According to the space design, you create \( est_{2,1} \) and add it to the formula.
The image of the topological space of an estimate

D. Data output through the maps

-Calculation of the estimated cost-

To calculate the estimated cost of $product_1$, you first get the term for “product,” from formula C-4 by the condition formula processing map (II-C), and you use the replacement map on the assumption that $1 is ¥90, and use the maps according to the above design for the cost space (III-C) as follows:

\[
\begin{align*}
&\text{f}(\text{the term of “product”}, “[\ldots]+m^2+kg+
\text{piece}\times\text{day}\times\text{hour}\times$ ¥90$)) \\
&= \text{f}(¥1000/m^2\times10^3+¥90\times200/100kg\times900kg+¥90\times12/day\times5/day) \\
&\times2/\text{piece}\times¥500/m\times10m+¥2000/hour \times4/hour/\text{piece}\times10/\text{piece}+\text{labor} \\
&= ¥1,381,600/piece
\end{align*}
\]

From the result, you can know that the cost of “product” is ¥1,381,600.

-Estimation of resources-

To calculate the cost of $material_1$ in $product_1$, first you get the image of formula C-4 by “product:\{material,;” through the condition formula processing map as follows:

\[
\begin{align*}
&\text{D-1:} \\
&\text{product},(\text{parts}\times¥200/100kg\times900kg)/\text{piece}\times5 \\
&\text{piece}+\text{parts}\times¥200/100kg\times1000kg)/\text{piece}\times10 \\
&\text{piece}
\end{align*}
\]

Next, in the same way, you use the maps as follows:

\[
\begin{align*}
&\text{f}(\text{f}(\text{formula D-1}, “$=¥90”), “[\ldots]+kg+\text{piece}+$$+ ¥”)) \\
&= ((¥90\times200\times100^2\times900)\times5+(¥90\times200\times100^2)) \times1000 \\
&= ¥990,000
\end{align*}
\]

From the result, you can know that the cost of $material_1$ in $product_1$ is ¥990,000.

In this way, you can flexibly calculate the estimated cost of a product or the cost of a resource that you want from the formula created using the maps of CDS.

E. Considerations

In general, it is difficult to model the estimate space because the unit costs, quantities and unit names, etc. of resources are combined in multiple layers complicatedly in an estimate. As a result, in many cases, the development of core logic of the estimate system is costly and the function or space design is actually limited. This example shows that if you only have to design formulas for the estimate spaces such as in IV.B and input the estimate data according to the design, you can output the required data using the maps of CDS, thereby reducing the number of application programs. This is mainly because:

1. The formula for a disjoint union of topological spaces of estimates can be designed simply using CDS.

If you utilize CDS, you can design the formula for estimates quite simply, as you would do in your head, because CDS supports recursive expression and the expression of a mixture of numerical values and characters in an identifier.

2. The formulas created for the estimate spaces can be calculated flexibly according to user requirements through the maps of CDS

It is possible to calculate estimated costs or material data with units to meet user requirements using the maps in IMAH, such as the conditional formula processing map on the topological space level, and the numerical and exponential calculation and the mask map on the presentation level.

V. Related Works

The distinctive features of our research are 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.

VI. CONCLUSIONS

We have developed a data processing system called the Cellular Data System (CDS) based on IMAH. In this paper, we introduced into CDS the function of numerical and exponential calculation and the applied maps, and successfully applied them to the core logic development of an estimate system. Using those, the core logic development of the system becomes much simpler, reducing the number of application programs. As a result, use of CDS can make developers more creative, preventing frequent problems between the customer side and the supplier side, while preventing combinatorial explosions.

REFERENCES