지능정보연구 제17권 제3호 2011년 9월(pp.169~185)

Association-Based Conceptual Modeling for Smart Database Design

Sangwon Lee
Division of Information and Electronic Commerce, Wonkwang University
(sangwonlee@kaist.edu)

Data redundancy is problematic in that it not only induces heavy storage management cost but also could bring critical degradation of information systems. Unfortunately, to our knowledge, only few enterprises willingly afford time and efforts for the faithful conceptual design to prevent the degree of inappropriate data as much as they could, while most of enterprises pay rare attention to the notion of that sort of data quality. Wondering if there would be any other way to design the enterprise-wide data design without prior knowledge about business works is our major motivation for this study. In this paper, we present our data modeling methodology in which associations among objects in each sentences of a business job descriptions are treated as the focal point in database design. A proposed agent for automated design tool simply takes a business job description written in natural language as an input, and then designs an entity relationship diagram with some smart rules. We introduce the scope of the proposed agent and its detailed logics with several examples. And then, we verify the appropriateness of the resulted associations among objects. Lastly, we perform case studies to evaluate the devised agent's applicability to a business field.

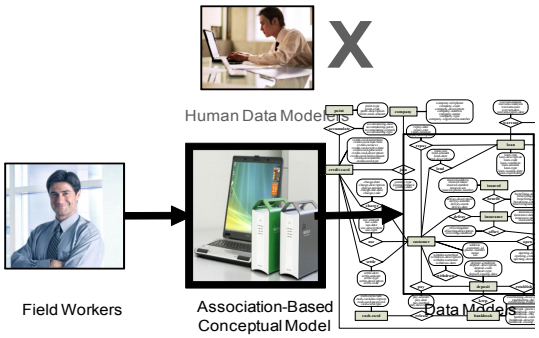
Received : July 05, 2011 Revision : July 15, 2011 Accepted : August 01, 2011
Type of Submission : Excellent Paper of Spring Conference Corresponding author : Sangwon Lee

1. Introduction

To obtain an appropriate Entity Relationship Diagram (ERD) (Batini et al., 1992; Chen, 1976) for a specific application, we should attach great importance to field workers' requirements that reflect real business. And the requirements (Sommerville, 2011) should be obtained by the business descriptions that portraits field works for an appropriate enterprise-wide data

* This paper was supported by Wonkwang University in 2011.

map (Moon, 2004). But, many automation tools (Elmasri et al., 2007; Yang et al., 1999; Teorey et al., 1986; Choobineh et al., 2005; Dogac et al., 1989; Kaula, 2000; Mills et al., 2002; Storey et al., 1997; Gomez et al., 1999; Gangopadhyay et al., 2001) for conceptual data modeling have been researched without caring about business requirements. Some of those tools such as ERWin and Rational Rose make it possible to design databases without any help of a data modeler. But they still lay a burden on a user by demanding knowledge on database of or put too many questions to the user.



<Figure 1> Automated database design

To design databases effectively, a domain user should represent his/her own business works without intervening of other person like data modeler as well as without any knowledge on databases. It is cardinal as a trial to graft technology on management how to map domain tasks to data for systemizing business works. The entire design processes tends to be complicated, which means that data are not complicated but the processes of describing or analyzing data are complicated. It is no better than Herbert Simon's (Simon, 1981), Ant on the Beach. To quote Simon, "Viewed as a geometric figure, the ant's path is irregular, complex, and hard to describe. But its complexity is really a complexity in the surface of the beach, not a complexity in the ant." It would certainly increase the correctness and usability of data modeling to make design processes simple by laying down rules for business descriptions and by automating its transformation with these.

This paper proposes a new methodology in <Figure 1> for extracting entity or attribute or relationship so that a field worker, not an expert modeler, can perform the conceptual data modeling with several rules. In getting an appropriate ERD, an agent with this methodology uses only the business descriptions without any other information on databases and without any interactions between the automated tool and a user. Before creating an ERD, a graphical automated agent called Association-Based Conceptual Model (ABC Model or ABC) finds out associations between two or more objects described in the business descriptions. This ABC Model covers the following algorithms; refining rules for business descriptions, detecting associations among objects extracted from the business descriptions, and extracting entities, attributes, and relationships. After reviewing the previous works which are related to the automation of conceptual data modeling, we introduce the scope of the proposed agent with conceptual modeling rules. We show some examples and perform a case study to evaluate

the devised agent's applicability to practical business fields. And also we perform experimental verification of associations extracted from the case study with a popular tool.

2. Related Works

There are many Computer Aided Software Engineering (CASE) tools to draw data model for information systems. It is true that it is easy for a data modeler to draw various diagrams for data model with them. Especially automation tools such as ERW in and Rational Rose save much time and endeavors for a data modeler to draw ERDs and generate logical schemata from them. However, since those tools adopt too complicated development steps in data modeling, they could give much help for a data modeler but no help for a field worker. As a most widely used tool in business fields, even ERW in offers only a drawing function For ERD. The tool does not play a role to analyze users' requirements, which could bring about passing over the original information or distorting it (Turban et al., 2011).

Or course, all traditional tools for data modeling do not have only a drawing function. Many previous studies on logical and physical phases have accomplished much contribution. But, few studies on the formal methodology for conceptual data modeling have been found in the literature. An approach (Yang et al., 1999) tried to formulate an ERD by use of data-intensive source codes. Meanwhile, one of the most famous approach (Teorey et al., 1986) tried to extract entity according to the principle of Entity Relationship Methodology (ERM) (Chen, 1976) theory and provided its guideline with two assertions that most human database designers extract entities are as follows; One is that an object can be classified as an entity if it has descriptive information for itself. And the other is that, if a descriptor is multi-valued, the descriptor can be classified as an entity although it does not have any other descriptors. And then Teorey (Teorey et al., 1986) created another guideline for extracting entities with a statistical mechanism called Attribute Synthesis Method (ASM). ASM statistically manipulates the objects of given tasks and then finds out entities with defined thresholds for the indices. But, ASM does not clearly determine whether a value of an index is high or low. The users' arbitrary selection of the threshold could make the quality of ERD uneven.

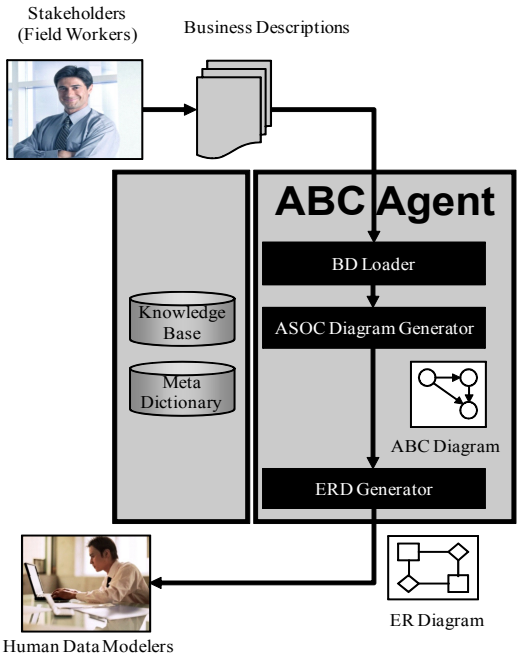
Sometimes, knowledge base was applied to automate conceptual modeling (Choobineh et al., 2005; Dogac et al., 1989; Kaula, 2000; Mills et al., 2002; Storey, 1997). On the basis of knowledge base, a field worker can get lists of entities and attributes as initial inputs and then validate or refine them. More developed approaches on the basis of ontology (Story et al., 2002; Sugumaran et al., 2002; Sugumaran et al., 2003; Wand et al., 1999; Weber, 2003) have also been researched. Especially, Weber (Weber, 2003) used the ontology theory in addressing four limitations in grammar of ERM. Another methodology (Kim et al.,

2008) tried to extract entities and attributes from business description in natural language, but its complex algorithm and inaccuracy would not extract entity and attribute soundly and completely. Moreover, the methodology does not accompany any objective and positive experiment and requires too many questions to a user.

3. Association-Based Conceptual Agent for Entity Relationship Modeling

3.1 Association-Based Conceptual Agent

Our proposed system in <Figure 2>, Association-Based Conceptual (ABC) Agent is an automated tool for conceptual data modeling based on ABC Model. The agent draws an ERD by use of several inner rules of ABC Model and business descriptions that a field worker writes. ABC Agent is made up of three major modules and two repositories. Three modules are Business Description (BD) Loader, Association-Based Conceptual Diagram (ABCD) Generator, and ERD Generator. And two repositories are Knowledge Base (KB) and Meta Dictionary (MD). After a field worker writes his or her own BD in natural language, BD Loader loads it and parses its sentences into objects with referring to KB or MD. ABCD Generator transforms extracted refinery objects into an initial ABCD with several ABCD rules. Consulting KB and MD, ABCD Generator refines the initial ABCD to get a revised ABCD. Lastly, with ERD-related rules, ERD Generator draws a final ERD with translating the revised ABCD. This ABC Agent is an upgraded version of Association-Based Conceptual Systems (Lee et al., 2010) with more detail algorithm to make data model sound and complete.



<Figure 2> ABC Agent Architecture

3.2 Assumptions of Association-Based Conceptual Agent

To delimit environments of the devised system, we need to propose several assumptions so as to define the environmental scope of ABC Agent.

(Assumption 1) All potential users must be well acquainted with their own works required by to-be data model.

(Assumption 2) The business descriptions a field worker makes should include all business behaviors related to business objects.

(Assumption 3) For uniqueness and consistency of object's name, ABC operates MD that has all objects list. A field worker could describe objects of his/her own fields' works with referring to MD.

3.3 Definitions of Association-Based Conceptual Agent

All the procedure from BD to ERD of ABC Model for ABC Agent is wholly based on the business descriptions. And they have has several rules in order to extract objects from DB, detect associations among objects, and extract entities, attributes, and relationships for a final ERD.

(Definition 1) Each sentence of BD is S_1, S_2, ⋯, or Sn.

(Definition 2) Each object of every S_i is O_1, O_2, ⋯, or On. A situation that a sentence S_i has two objects, O_1 and O_2, is expressed as $\exists$ O_1, $O_2 \in S_i$ for $S_i \in$ BD.

(Definition 3) A subordination between objects is expressed by flowing links between objects as →, ←, or ↔. A subordination $O_1 \rightarrow O_2$ implies that an object O_1 does not appear without an object O_2. If O_1 is subordinated to O_2, where O_1 is the subordinate and O_2 is the subordinator, O_1 necessitates O_2; for $\forall$ $S_i \in S$, $\urcorner \exists$ S_i such that $O_1 \in S_i \wedge O_2 \notin S_i$. $O_2 \rightarrow O_1$ is generally represented for functional dependency (Ramakrishnan et al., 2011) $O_1 \rightarrow O_2$.

(Definition 4) A subordination $O_1 \leftrightarrow O_2$ is the same as two subordinations, $O_1 \leftarrow O_2$ and $O_1 \rightarrow O_2$.

(Definition 5) A subordination $O_2 \leftarrow O_1 \rightarrow O_3$ stands for both two subordinations, $O_1 \rightarrow O_2$ and $O_1 \rightarrow O_3$.

3.4 Rules of Association-Based Conceptual Agent

1) Rules to Extract Objects and Associations

An ABC Diagram represents objects and associations by use of not only five definitions but also some articulated rules and theorems. The following five ABC Diagram Rules are fundamental to transform BD into an ABC Diagram, because they create objects and associations among objects in given BD.

(ABC Diagram Rule 1) If O_1 always appears with O_2 in a sentence of BD, its subordination is represented as $O_1 \rightarrow O_2$. If there is an object O_3 for an object O_2, a subordination $O_2 \rightarrow O_3$ should be added. If there is initially a subordination $O_1 \leftrightarrow O_2$ between an object O_1 and an object O_2, $O_1 \leftrightarrow O_2$ can be separated into $O_1 \rightarrow O_2$ and $O_2 \rightarrow O_1$. And if there is no mention of O2 in a sentence Si with O_1, it means that there is not a subordination of O_2 for O_1. So, the subordination $O_1 \rightarrow O_2$ should be deleted from $O_1 \leftrightarrow O_2$.

(ABC Diagram Rule 2) A node that has one or more outgoing link is defined as a leaving node (LN). A node that has only incoming links and does have any outgoing link is also defined as a reaching node (RN). Of course, if any LN follows its links, it should meet RN (case 1) or be located in a cycle (case 2). RN cannot be located in a cycle.

(ABC Diagram Rule 3) If there is a cycle with three outgoing node, O_1, O_2, O_3, a pseudo object (PO) reflecting three subordinations $O_1 \rightarrow$ PO, $O_2 \rightarrow$ PO, $O_3 \rightarrow$ PO should be added. If PO is not created, the nodes O_1, O_2, O_3 could be orphaned without necessitating any object. PO should be a reaching node. PO is a reaching node and O_1, O_2, O_3 are leaving nodes. Cycled objects are all LNs. If a PO is added, this cycle is deleted. All O_1, O_2, O_3 as LN necessitates PO as RN. After all, each leaving node surely meets a reaching node if following its links.

(ABC Diagram Rule 4) $LN_1 \rightarrow LN_2 \rightarrow$ RN is transformed into $LN_1 \rightarrow$ RN and $LN_2 \rightarrow$ RN. Each LN has an RN as its host. Also, every link that remains in ABC Diagram is a relation, LN $\rightarrow$ RN.

2) Rules to Extract Entities and Relationships

An ABC Diagram is transformed into a final ER Diagram by use of following ER Diagram Rules. The rules are used to simplify all associations among objects in the shape of A(attribute) $\rightarrow$ E(entity).

(ER Diagram Rule 1) Every RN is transformed into an entity and every LN is transformed into an attribute.

(ER Diagram Rule 2) If an attribute A has only one outgoing link and its indicated end is an entity E, A is an attribute of E.

(ER Diagram Rule 3) If an attribute A has two or more outgoing links and its indicated ends are E_1, E_2, ⋯, En, a relationship R that participates in E_1, E_2, ⋯, En is created and A is an descriptive attribute of R.

(ER Diagram Rule 4) For entity E_1, E_2, ⋯, En that appear together in a sentence, if there is no relationship that is composed of E_1, E_2, ⋯, Em, a relationship R is created by use of a verb in the sentence.

(ER Diagram Rule 5) If attributes A_1, A_2, ⋯, An have two or more outgoing links and their indicated ends are E_1, E_2, ⋯, En, a relationship R that participates in E_1, E_2, ⋯, En is created and A_1, A_2, ⋯, An are descriptive attributes of R.

4. Practical Applications of Association -Based Conceptual Modeling

4.1 Practical Examples of ABC Modeling

Now, we show four practical examples with the above-mentioned three assumptions, five definitions, and nine rules with four ones to extract objects and associations and five ones to extract entities and relationships.

(Example 1) Let us suppose that a field worker makes the following BD <Figure 3>(a). The revised version of BD <Figure 3>(b) is the one after refining to <Figure 3>(a).

Descriptive ambiguities should be removed during the process from its original version to revised one. It is absolutely necessary for the business descriptions that all the statement should be described clearly. For example, when we call an exit number of a subway station, we must call

both the exit number and its station number. If we call only the exit number, we can't know where the exit is, which is vague. RS_5 in <Figure 3> is refined with removing the ambiguity of S_5. The 'credit' should be described as 'credit of a course.' The only 'credit' would be misunderstood as 'credit of a student' of 'credit of a professor.' In the same manner, 'salary step' of S_7 must be written as 'salary step of a professor.' Writing without exception of necessary information assures the completeness. Even though optionally required, a simple sentence is better than a complex or compound one in describing business works. As far as there is no information loss, a simple sentence makes the business descriptions clear and accurate. And then it gives a help to understand the business works. For more clear-cut understanding, a compound statement S_2 should be separated into simple sentences RS_2 and RS_6. Especially, over-issuing complex or compound sentences could bring about over-valuating objects and so could produce unnecessary relations among objects. Writing only simple sentences for necessary information assures the compactness. Though making ambiguous sentences clear, there is no use in writing the same objects several times in a sentence. To assure the completeness of the sentence S_7, RS_8 would be written as "The salary step of a professor is calculated by the salary class of the professor." In this case, only one 'professor' in RS_8 is enough. So, 'professor' of 'the salary class of the professor' may be removed. Whether the count of 'professor' is one or two does not matter and causes the same result. Hence, there is no restriction about appearance frequency for an object in a sentence.

S_1. A student has student id and name.
S_2. A student should belong to a department, and then takes a course and gets a grade.
S_3. A student is identified by student id.
S_4. A course is identified by course code, and has its own credit.
S_5. A credit is determined from 1 to 4.
S_6. A professor is identified by professor id.
S_7. The salary step is calculated by the salary class.
S_8. A professor can teach many courses.

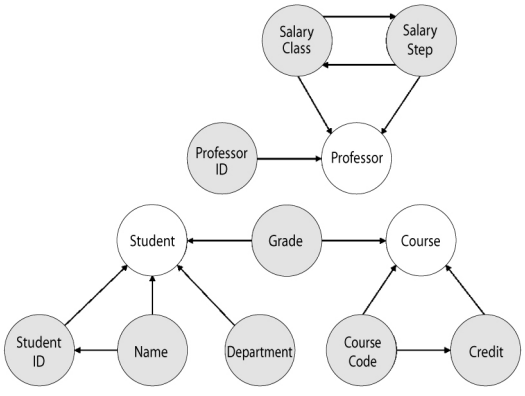
(a) Original version

RS_1. A student has student id and name.
RS_2. A student should belong to a department.
RS_3. A student is identified by student id.
RS_4. A course is identified by course code, and has its own credit.
RS_5. A credit of a course is determined from 1 to 4.
RS_6. A student takes a course and gets a grade.
RS_7. A professor is identified by professor id.
RS_8. The salary step of a professor is calculated by the salary class.
RS_9. A professor can teach many courses.

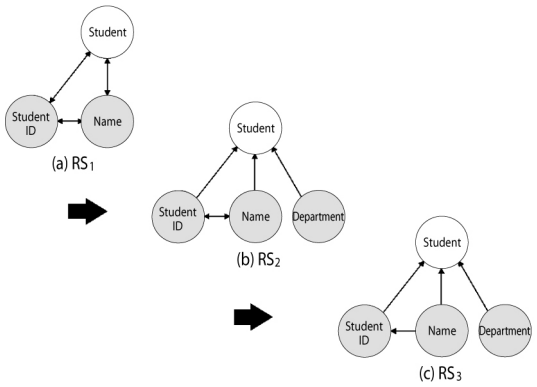
(b) Revised version

<Figure 3> BD

(Example 2) The RS_1~RS_9 in the revised business descriptions in <Figure 3>(b) is transformed into an ABC Diagram in <Figure 4> by Definitions 1 to 5 and ABC Diagram Rules 1 and 2.



<Figure 4> Initial ABC Diagram

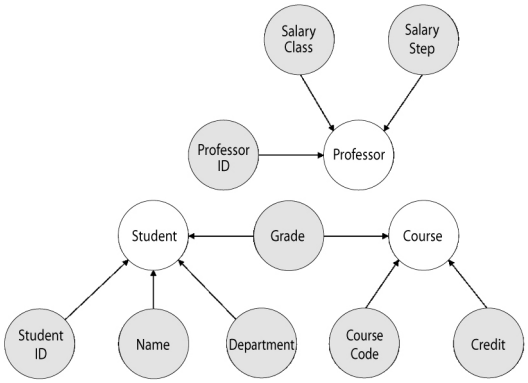


<Figure 5> Object Subordinations for RS_1, RS_2, and RS_3

Let us examine the process for <Figure 4>. (a) RS_1 in <Figure 3>(b) is a sentence with three objects, Student, Student ID, and Name. The subordinations among these objects is expressed by Definition 2 and the <Figure 5>(a) is drew by Definition 3 and ABC Diagram Rule 1-1. The subordinations among objects, Student ↔ Student ID, Student ↔ Name, and Student ID ↔ Name is all shown. (b) Let us add RS_2 of <Figure 3>(b) to <Figure 5>(a) and redraw <Figure 5>(b). Definitions 1 and 2 create two objects, Student, Department. And then Definitions 3 and 4 and ABC Diagram Rule 1-1 could create a subordination Student ↔ Department. But, although there is a mention of Student in RS_1 and there is not a mention of Department, Student does not always accompany Department. Hence, only the subordination Student → Department is added to <Figure 5>(a).

Meanwhile, since there is no mention of Student ID and Name while mentioning Student, we can see that Student always does not escort Student ID and Name. Hence, ABC Diagram Rule 1-2 deletes the subordinations Student → Student ID and Student → Name from <Figure 5>(a). (c) Applying RS_3 to <Figure 5>(b) results in <Figure 5>(c). Since Definition 2 creates objects Student and Student ID and Since RS_3 does not mention Name even with Student, ABC Diagram Rule 1-2 deletes the subordination Student ID → Name. After this manner, the ABC Diagram in <Figure 4> can be obtained. In the end, ABC Diagram Rule 2 is applied to <Figure 4> by color-code classifying coding into reaching or leaving nodes.

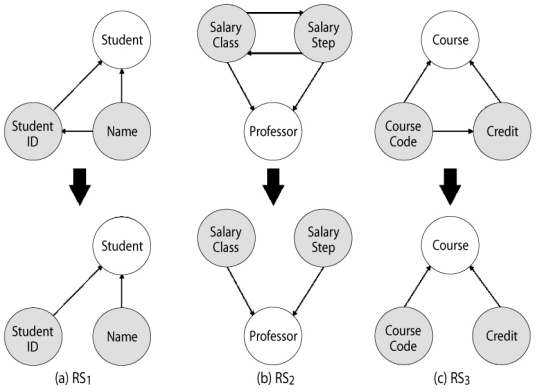
(Example 3) As ABC Diagram Rules 1 and 2 draw the Initial ABC Diagram in <Figure 4> of Example 2, ABC Diagram Rules 3 and 4 draw the Revised ABC Diagram (<Figure 6>). This diagram will be a basis for our ultimate goal, ERD.



<Figure 6> Revised ABC Diagram

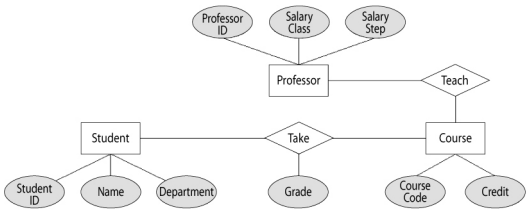
Let us examine the process to draw <Figure 6> in detail. <Figure 6> is completed after

deleting subordinations among leaving nodes from Initial ABC Diagram in <Figure 4>. In <Figure 7>(a), ABC Diagram Rule 4 is applied to Name (SRC) → Student ID (SRC) → Student (SNK) in <Figure 4>. By transitivity, Name (SRC) → Student ID (SRC) → Student (SNK) is simplified into Name (SRC) → Student (SNK) and Student ID (SRC) → Student (SNK). In <Figure 7>(b), Salary Class → Salary Step → Professor is transformed into Salary Class → Professor and Salary Step → Professor, and Salary Step → Salary Class → Professor is transformed into Salary Step → Professor and Salary Class → Professor. These are summarized as Salary Class → Professor and Salary Step → Professor. In the same manner, in <Figure 7>(c), Course Code → Credit → Course is changed into Course Code → Course and Credit → Course.

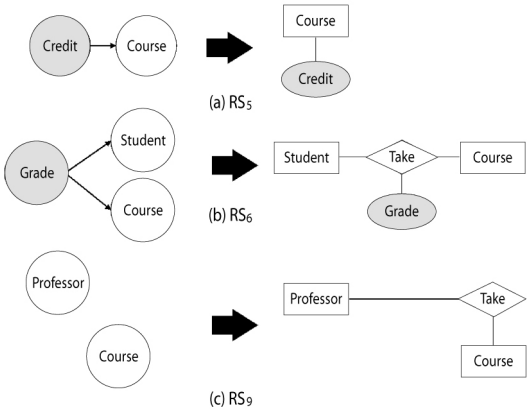


<Figure 7> Transitivity for RS_1, RS_4, and RS_8

(Example 4) By ERD Rules 1 to 5, ABC Diagram in <Figure 6> is transformed into an ERD in <Figure 8>.



<Figure 8> Final ERD



<Figure 9> Drawing an ERD for RS_5, RS_6, and RS_9

Let us examine the process to Draw <Figure 8> in detail. (a) RS_5 in <Figure 3>(b) is represented as <Figure 9>(a) by ERD Rule 2. Since Course is an entity in a subordination Credit → Course, the object Credit becomes an attribute of Course. (b) RS_6 in <Figure 3>(b) is represented as <Figure 9>(b) by ERD Rule 3. Since Student and Course are entities in a subordination Grade → Student and a subordination Grade → Course, Grade becomes a descriptive attribute of a relationship between two entities, Student and Course. This relationship is endowed with the name, Take, by MD that has verb information. (c) RS_9 in <Figure 3>(b) is represented as <Figure 9>(c) by ERD Rule 4.

A customer opens an account with opening-date, opening-condition, and opening-description. Each customer has customer-id, ssn, name, address, and phone-number. An account has account-code, account-starting-date, and account-description. A employee has employee-code, employee-name, position-class, and entering-date. Each employee administers an account with administering-date, administering-condition, and administering-remark. A branch manages many employees. A branch has branch-code, branch-name, branch-location, branch-telephone, and branch-address. The head-quarter controls several branches. The head-quarter publishes many security-cards with publishing-code, publishing-date, publishing-condition, and publishing-description. A security-card has security-code, security-id, pin-code, pin-password, and security-recognition-number. A security-card is assigned to an account. An account establishes many deposits with establishing-date,

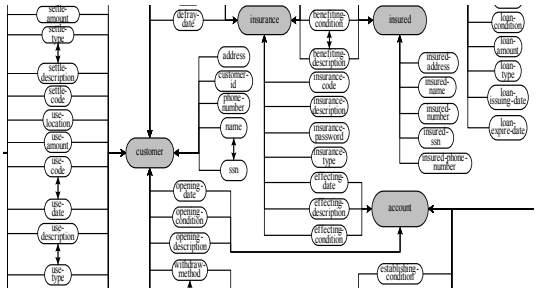
<Figure 10> A Part of Loaded BD of S Bank

Since there is no relationship between entities, Professor and Course, in RS_9, a relationship named Teach is created by verb information of MD.

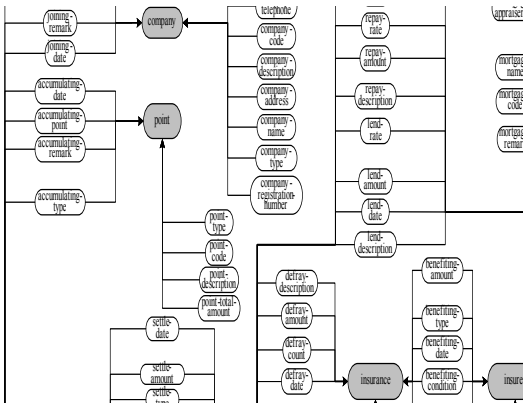
4.2 Case Studies of ABC Modeling

Although we showed rules-applied examples, it is however impossible to discover an absolutely answer that could correctly analyze or evaluate the methodological practicability. The reason is that a database schema is artistic and does not propose a correct answer to a business work. Then it is meaningless to verify the practicability theoretically. Now we conduct a case study to analyze the practicability of our methodology with chosen real business works, which will generate a to-be ERD we want.

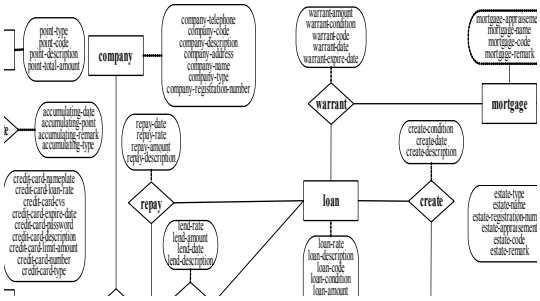
The target work of the experiment is personal finance of S Bank, in the Republic of Korea, where a field worker wrote his/her own BD in <Figure 10>. BD Loader firstly extracts 374 objects from all sentences of BD. And, The Initial ABC Diagram <Figure 11> is drawn by 288 associations among objects by applying ABC Diagram Rules 1 and 2. Eliminating transitivity on ABC Diagram Rules 3 and 4 simplifies the Initial ABC Diagram into a Revised ABC Diagram in <Figure 12>. By removing all 46 two-way edges, the number of subordinations of the Initial ABC Diagram is reduced from 288 to 242. ERD Rules 1 to 5 are applied for a final ERD in <Figure 13>. Finally, from 374 objects in original 80 sentences of BD, an ERD is finally completed, which is composed of 17 entities, 162 attributes, and 21 relationships.



<Figure 11> A Part of Initial ABC Diagram of S Bank



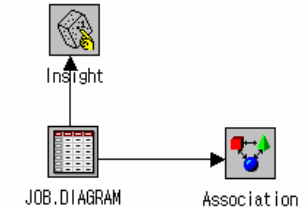
<Figure 12> A Part of Revised ABC Diagram of S Bank



<Figure 13> A Part of ER Diagram of S Bank

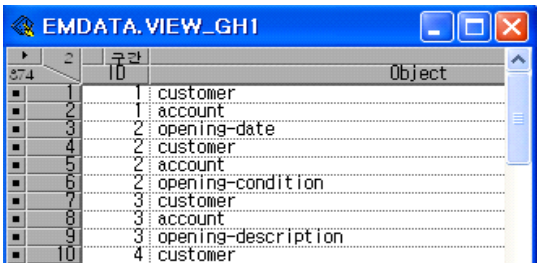
5. Experimental Verification of Associations

Now we perform experimental verification of associations extracted from the case study of S Bank. The most important phase from business descriptions to a final ERD is to discover subordinations between objects in each sentence of given business descriptions. ABC Diagram Rules 1 and 2 are bases for completing ABC Diagram that can represent basic subordinations among objects. As an experimental analysis of associative relations among objects, we verify the appropriateness of the resulted relations among objects by use of a package SAS (originally Statistical Analysis System) System version 9.1.3 made by SAS Institute. To show that dependencies among objects in the results of SAS E* Miner are identical to subordinations from ABC Diagram Rule 1, we use Association (Han et al., 2007) technology in Association Rule Modeling (ARM) in Enterprise Mine (E* Miner) Module of the system to discover basic subordinations among objects instead of using ABC Diagram Rule 1.

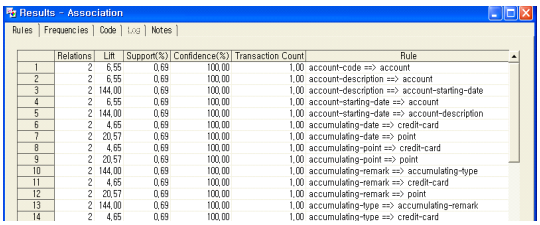


<Figure 14> Association Tools of E* Miner

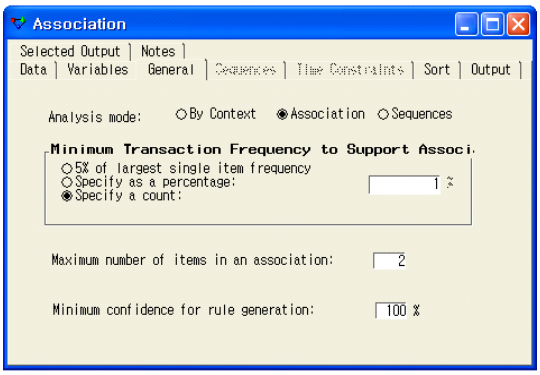
<Figure 14> shows three modules provided by Association module of E* Miner, which is composed of Input Data Source Module, Insight Module, and Association Module. Node "JOB. DIAGRAM" as Input Data Source Module is a tool to load data source that stores objects in each sentence of BD. Node "Insight," a tool for Insight Module, shows the loaded data itself or the distribution of variables in source graphically. The loaded data are shown through Insight Node like <Figure 15> and the objects in each sentence are input in the record unit. Each sentence has its own ID (identification number) and has several objects in it. Let us take a sentence, "1. A customer opens an account." The ID of the sentence is '1' and there are two objects, 'customer', and 'account.' The pair of an ID and an object is input into one record. In this way, we can get two pairs from the sentence; (1, customer), and (1, account). After this manner, we have 374 records from the loaded BD of <Figure 10>. Node "Association" finds out associations, namely subordinations, among objects. This node results in associations among objects in <Figure 16>. There are 288 associations which are exactly identical with subordinations acquired by Initial ABC Diagram of <Figure 11>.



<Figure 15> Loaded Data in Insight Node



<Figure 16> Results in Association Nodes



<Figure 17> Setting for Associations

To find out associations among objects, ARM requires parameter to be set in <Figure 17>. (1) In the upper portion of the screen, there are three radio buttons for 'Analysis mode.' We must check 'Association' to use Association tool. (2) 'Minimum Transaction Frequency to Support Associ' is used for setting the value of 'support.' For two objects A and B, the 'support (on A → B)' is defined as the number of (A ∩ B) over the total count of transactions. The fact that there exists A → B means that A does not appear at all without B. If an object appears just one time, it participates in the ERD as an entity or attribute. Our experiment set '1' for the minimum threshold of support count so that the restriction of support could prevent association rules from being leached. (3) 'Maximum number of items in an association' sets the number of objects appeared in each association. We set 2 for the value because we manage only the relations between two objects. That is, the '→' is located between two objects. (d) 'Minimum confidence for rule generation' is the conditional probability that an object will appear when another object appears. For two objects A and B, the 'confidence (on A → B)' equals the number of (A ∩ B) over the number of A. The fact that B appears when A appears is identical with that (A ∩ B) appears when A appears. We set '100' percent for the confidence value because we manage only perfect associations between A and B. This fact is n(B | A) = 100 (%), or n(A ∩ B)/n(A) = 100 (%) in conditional probability. <Figure 16> shows only subordinations where n(B | A) = 100 (%).

5. Conclusions

Our automated conceptual modeling agent is a requirements-translucent modeling tool that its whole processes from BD to ERD are transparent. While this model draws an initial or revised ABC Diagram and a final ERD, it eval-

uates the importance of objects on the basis of associations of objects in contexts of each sentence and tries to eliminate transitivity redundancies of data. And ABC Agent reflects given business descriptions from the viewpoint that the conceptual design process depends not on other subjective interventions but on only given business descriptions. Since this modeling demands only business descriptions ready in advance as initial inputs, it does not necessitate additional interactions that would bother a user and produces a requirements-faithful conceptual schema.

Although our ABC Agent could have elaborate and perfect rules to formulate an appropriate ERD, insincere inputs of a user could make the ERD a failure. In getting an ERD that reflects real business works well, to induce a user to write business description sincerely is as important as to develop complete rules. It is absolutely necessary to develop a module for inputting business works conveniently rather than writing business description in a natural language, which would certainly increase expediency for a user in using a tool and make a final ERD more correct. Further studies should concentrate its focus on develop a method to input business works more effectively and efficiently rather than in a natural language. For instance, to input by use of graphic notations or by use of template sentences would be a good alternative to input sentences in a natural language. An alternative method would certainly encourage field-working users to input their up-to-date business works more correctly and thoroughly, which will benefit an appropriate ERD and then be to the good of business-faithful enterprise information systems.

Our thesis not only included the algorithm for extracting subordinations of the proposed modeling methodology, but also verified it by use of Association of E* Miner of SAS. We examined that the result of association analysis performed when confidence is 100 percent is identical with final result acquired by the staged algorithm to extract subordinations. However, this sameness of results complies with the assumption of completeness of information introduced in this thesis, so this would be difficult while modeling in the real fields. Since an additional trial to alleviate this strict assumption is needed, we are performing a further study on easing the strictness with confidence level varied. The classification of entity or attribute would be certainly different according to levels of confidence, which leads to modification of ERD. If business descriptions have large number of sentences, an ERD that is resilient against changes of the confidence level would be obtained. That is to say that, even when information is imperfect, we would certainly obtain an ERD almost identical with that of perfect information. In further studies on this, we will perform an experiment with business descriptions that have a larger number of sentences and also examine the modification of ERD according to confidence level.

References

Batini, C., S. Ceri and S. B. Navathe, *Conceptual*

Database Design, The Benjamin/Cummings Publinshing, 1992.

Chen, P. P., "The Entity-Relationship Model-Toward a Unified View of Data", *ACM Transactions on Database Systems*, Vol.1, No.1(1976), 9~36.

Choobineh, J. and A. W. Lo, "Should Rule-based Reasoning Be Enhanced by Case-based Reasoning for Conceptual Database Design? A Theory and an Experiment", *Journal of Computer Information Systems*, Vol.46, No.2 (2005), 69~77.

Dogac, A., B. Uruten and S. Spaccapietra, "A Generalized Expert System for Database Design", *IEEE Transactions on Software Engineering*, Vol.15, No.4(1989), 479~491.

Elmasri, R. and S. B. Navathe, *Fundamentals of Database Systems*, Benjamin/Cummings, California, 2007.

Gangopadhyay, A., "Conceptual Modeling from Natural Language Functional Specifications", *Artificial Intelligence in Engineering*, Vol. 15, No.2(2001), 207~218.

Gomeza, F., C. Segami and C. Delaune, "A System for the Semiautomatic Generation of E-R Models from Natural Language Specifications", *Data and Knowledge Engineering*, Vol.29, No.1(1999), 57~81.

Han, J. and M. Kamber, *Data Mining* : *Concepts and Techniques*, Morgan Kaufmann Publishers, 2007.

Kaula, R., "Integration of Rule-based Systems and Database", *Journal of Computer Information Systems*, Vol.40, No.3(2000), 38~43.

Kim, N., S. Lee and S. Moon, "Formalized Entity Extraction Methodology for Changeable Business Requirements", *Journal of Information Science and Engineering*, Vol.24(2008), 649 ~671.

Lee, S., N. Kim and S. Moon, "Context-Adaptive Approach for Automated Entity Relationship Modeling", *Journal of Information Science and Engineering*, Vol.26, No.6(2010), 2229~2247.

Mills, K. L. and H. Gomaa, "Knowledge-based Automation of a Design Method for Concurrent Systems", *IEEE Transactions on Software Engineering*, Vol.28, No.3(2002), 228 ~255.

Moon, S., Data Architecture, *Hyung-Seoul Publishing Company*, 2004.

Ramakrishnan, R. and J. Gehrke, *Database Management Systems*, McGraw-Hill, 2011.

Simon, H. A., *The Sciences of the Artificial*, The MIT Press, 1992.

Sommerville, I., *Software Engineering*, Addison Wesley, 2011.

Storey, V. C. and R. H. L. Chiang, Debabrata Dey, Robert C. Goldstein, and Shankar Sundaresan, "Database Design with Common Sense Business Reasoning and Learning", *ACM Transactions on Database Systems*, Vol.22, No.4 (1997), 471~512.

Storey, V. C., R. C. Goldstein and H. Ullrich, "Naive Semantics to Support Automated Database Design", *IEEE Transactions on Knowledge and Data Engineering*, Vol.14, No.1(2002), 1~12.

Sugumaran, V. and V. C. Storey, "Ontologies for Conceptual Modeling : Their Creation, Use, and Management", *Data and Knowledge*

Engineering, Vol.42, No.3(2002), 251~271.

Sugumaran, V. and V. C. Storey, "Supporting Database Designers in Entity-Relationship Modeling : An Ontology-based Approach", *Proceedings of 24th International Conference on Information Systems*, (2003), 59~71.

Teorey, T. J., D. Yang and J. P. Fry, "A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model", *ACM Computing Surveys*, Vol.18, No.2(1986), 197~222.

Turban, E., D. Leidner, E. McLean and J. Wetherbe, *Information Technology for Management*, John Wiley and Sons Inc, 2011.

Wand, Y., V. C. Storey and R. Weber, "An Ontological Analysis of the Relationship Construct in Conceptual Modeling", *ACM Transactions on Database Systems*, Vol. 24, No.4(1999), 494~528.

Weber, R., "Conceptual Modelling and Ontology : Possibilities and Pitfalls", *Journal of Database Management*, Vol.14, No.3(2003), 1~20.

Yang, H. and W. C. Chu, "Acquisition of Entity Relationship Models for Maintenance- Dealing with Data Intensive Programs in a Transformation System", *Journal of Information Science and Engineering*, Vol.15, No.2(1999), 173~198.

Abstract

스마트 데이터베이스 설계를 위한 연관성 기반 개념적 모형화

이상원*

데이터 중복은 과중한 저장관리비용을 유발할 뿐만 아니라, 정보시스템의 질적 저하를 초래한다. 불행히도, 업무규정에 적합하지 않은 데이터를 방지하기 위해서, 업무에 충실한 개념적 설계를 위한 시간과 노력을 투자하는 기업이 극소수에 불과하다. 즉, 대부분의 기업들은 데이터 품질에 대한 인식에 관심을 갖지 않는다. 우리 연구의 주요 동기는, 업무에 대한 사전지식이 없이 업무에 충실한 기업전사 데이터를 설계하는 방식을 연구하는 것이다. 본 논문에서는 업무기술서에 나온 각 문장 내의 객체들 간의 연관성을 이용해서, 데이터설계를 자동화하는 데이터설계방법론을 제안한다. 자동화 설계도구를 위해 제안된 에이전트는 자연어 상태의 업무설계서만을 입력 받아, 특정한 규칙을 이용하여 개체관계도를 설계한다. 우리는 본 에이전트의 전체 범위와 자세한 절차를 약간의 예를 가지고 설명을 할 것이고, 그 다음에 객체 간에 추출된 연관성의 적법성을 검증할 것이다. 또한, 고안된 에이전트의 실무 적용성을 평가하기 위한 사례연구를 진행할 것이다.

Keywords : 에이전트, 규칙, 연관성, 개체관계도

* 원광대학교 정보전자상거래학부

저 자 소 개

Sangwon Lee

Received his Ph.D. degree in Management Engineering from Korea Advanced Institute of Science and Technology in 2009. He worked as a system analyst for Daewoo Information Systems Co., Ltd., as an adjunct professor for Hansung University, and as a research professor for Ewha Womans University. Since 2011, he has been working as an assistant professor for Wonkwang University. His research interest is management engineering such as data engineering, knowledge engineering, and so on.