Automatic Formal Verification of Conceptual Model Documentation by Means of Self-organizing Map

Algirdas Laukaitis and Olegas Vasilecas

Vilnius Gediminas Technical University , Sauletekio al. 11,
LT-10223 Vilnius-40, Lithuania
{algirdas.laukaitis,olegas}@fm.vgtu.lt

Abstract. By using background knowledge of the general and specific domains and by processing new natural language corpus experts are able to produce a conceptual model for some specific domain. In this paper we present a model that tries to capture some aspects of this conceptual modeling process. This model is functionally organized into two information processing streams: one reflects the process of formal concept lattice generation from domain conceptual model, and the another one reflects the process of formal concept lattice generation from the domain documentation. It is expected that similarity between those concept lattices reflects similarity between documentation and conceptual model.In addition to this process of documentation formal verification the set of natural language processing artifacts are created. Those artifacts then can be used for the development of information systems natural language interfaces. To demonstrate it, an experiment for the concepts identification form natural language queries is provided at the end of this paper.

Key words: Information systems engineering, formal concept analysis, IS documents self-organization, natural language processing.

1 Introduction

Software engineers spend hours in defining information systems (IS) requirements and finding common ground of understanding. The overwhelming majority of IS requirements are written in natural language supplemented with conceptual model and other semi-formal UML diagrams. The bridge in the form of semantic indexes between documents and conceptual model can be useful for more effective communication and model management. Then, an integration of the natural language processing (NLP) into information system documentation process is an important factor in meeting challenges for methods of modern software engineering.

Reusing natural language IS requirement specifications and compiling them into formal statements has been an old challenge [1], [14]. Kevin Ryan claimed that NLP is not mature enough to be used in requirements engineering [13] and

Advances in Computational Linguistics.
Research in Computing Science 41, 2009, pp. 143-154

Received 10/11/08
Accepted 26/01/09
Final version 06/02/09

our research express that as well. Nevertheless, we hope that the current paper will suggest some promising findings towards this challenging task.

The idea that we want to investigate in this paper consist from comparison of two concept lattices received from different information processing streams: 1) the first one is information processed by the human expert in the process of conceptual modeling, 2) and the second one is NLP of the domain documents. We suggest the formal concept analysis (FCA) [3] as a framework to compare results from those two information processing streams. We assume that an expert can define the domain object-attribute matrix. From that matrix FCA produces concept lattice which can be interpreted as the domain's conceptual model. For the documents set the formal concept analysis, if applied directly to words-document matrix, produces lattice that is far too big for any comprehensible analysis. Then, we suggest the self-organizing map (SOM) [10] as a tool for preprocessing and the problem dimensionality reduction. Finally, all presented ideas and methodological inference is tested with the IBM Information FrameWork (IFW) [7], which is a comprehensive set of banking specific business models from IBM corporation. For our research we have chosen the set of models under the name *Banking Data Warehouse*.

Then, solution for the stated problem organize the rest of the paper as follows. First, we present the general framework of the automated model generation system from the documents set. Next, we present IBM's IFW solution and the model which we used in our experiments. At this section we present FCA as the formal technique to analyze the *object:attribute* sets. In the Section 4, we present the architectural solution of the natural language processing system that has been used in this paper. SOM of the conceptual model is introduced in chapter 5. Finally to prove the soundness of the proposed method we provide a numerical experiment in which the ability of the system to identify concepts from users utterance is tested. The IBM Voice Toolkit for WebSphere [8] (approach based on statistical machine learning) solution is compared with the system suggested in this paper.

2 General Framework of the Solution

Conceptual models offer an abstracted view on certain characteristics of the domain. They are used for different purposes, such as a communication instrument between users and developers, for managing and understanding the complexity within the application domain, etc. The presence of tools and methodology that supports integration of the documents and communication utterance into conceptual model development is crucial for the successful IS development. In this paper for such tool we suggest a framework that is presented in the Figure 1. First, the corpus is created from the model's concept descriptions and in the figure it is named as the domain descriptions. The goal of the framework is to derive two concept lattices shown in the right side of the Figure 1. One concept lattice is generated from conceptual model which was created by the domain analyst. In the next section the process is described in details. The second concept

lattice is generated automatically from domain descriptions and we say that we have *"good"* descriptions if those lattices resembles each other.

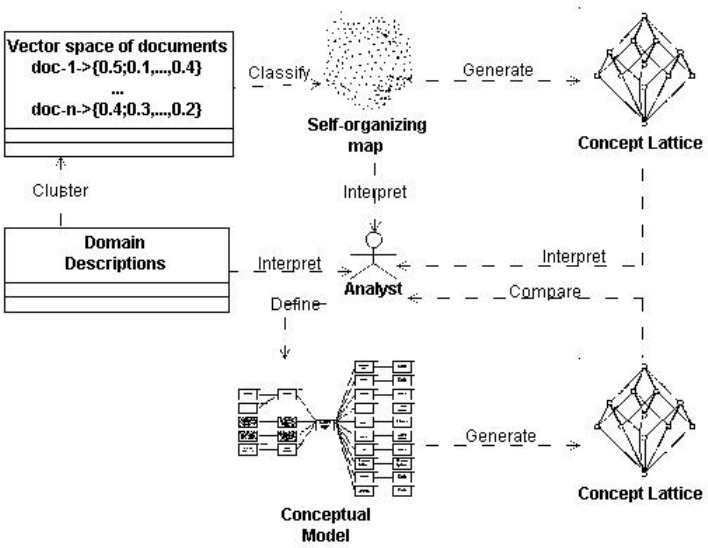

Fig. 1. Process of integration: Conceptual modeling, textual descriptions clusters detection and interpretation by use of FCA.

At the core of the second lattice generation is self-organizing map which is used for cluster analysis. In this paper we suggest the use of SOM to classify IS documentation and IS utterance on a supervised and an unsupervised basis. SOM has been extensively studied in the field of textual analysis. Such projects like WEBSOM [9], [11] have shown that the SOM algorithm can organize very large text collections and that it is suitable for visualization and intuitive exploration of the documents collection. The experiments with the Reuters corpus (a popular benchmark for text classification) have been investigated in the paper [6] and there was presented evidence that SOM can outperform other alternatives.

Because our goal is to derive formal lattice from the domain documents, which then can be compared with formal lattice from conceptual model, we suggest transformation schema that use FCA and transforms SOM to concepts lattice. It is important to notice that when directly applied to the big data set of textual information, FCA gives overwhelming lattice. This argument motivates integration of the FCA and other text clustering techniques. In that sense our work bears some resemblance with the work of Hotho et.al. [5]. They used BiSeckk-Means algorithm for text clustering and then FCA was applied to explain relationships between clusters. Authors of the paper have shown the usability of such approach in explaining the relationships between clusters of the Reuters-21578 text collection.

3 Concept Lattice Representation of the Conceptual Model

The problem with data centric enterprise wide models is that it is difficult to achieve their use by all employees in the company. Their abstract and generic concepts are unfamiliar to both business users and IS professionals, and remote from their local organizational contexts. Natural language processing can be used to solve mentioned problems. But, before applying the NLP techniques for the model and its documentation management, we must have some formal method to deal with the set of {*classes, object and attributes*}. In this section we introduce the formal concept analysis as the method for automatically building the hierarchical structure of concepts (or classes) from the {*object:attribute*} set.

As an example, consider Figure 2 (left side) we can see an excerpt of the IBM IFW financial services data model (FSDM) [7]. The IBM financial services data model is shown to consist of a high level strategic classification of domain classes integrated with particular business solutions (e.g. Credit Risk Analysis) and logical and physical data entity-relationship (ER) models.

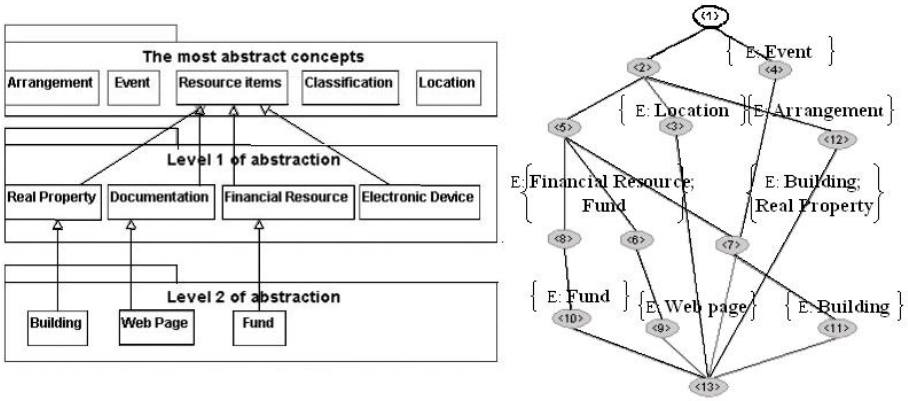

Fig. 2. Left side: A small extract from the financial services conceptual model. Right side: CL from this conceptual model. (We see that FCA depicts the structure from the conceptual model.)

Concept lattice of this model have been produced by FCA with Galicia software [16] and is shown in the right side of the Figure 2. As we can see it is consistent with the original model. It replicates underlying structure of conceptual model originally produced by an expert team and in addition suggests one formal concept that aggregates *Arrangement* and *Resource Item*: the two top concepts from the original model. As we can see from this simple example, FCA is used to represent underling data in the hierarchical form of the concepts. Due

to its comprehensive form in visualising underlaying hierarchical structure of the data and rigorous mathematical formalism FCA grown up to mature theory for data analysis from its introduction in the 1980s [3].

In order to better understand the process of concept lattice generation from conceptual model, we can return to the Figure 2. As we can see from the figure, there are 12 concepts. As the first step, we instantiate the set of those concepts. The set of instantiated objects we name as G. Let M be the set of all attributes that characterise those objects i.e. an attribute is includes into the set M if it is an attribute for at least one object from the set G. In our example we have 137 attributes (the whole model has more than 1000 objects and more than 4000 attributes). We identify the index I as a binary relationship between two sets G and M i.e. $I \subseteq G \times M$. In our example the index I will mark that, eg., an attribute "interest rate" belongs to an object "Arrangement" and that it does not belong to an object "Event".

The FCA algorithms starts with the definition of the triple $\mathbb{K} := (G, M, I)$ which is called a formal context. Next, the subsets $A \subseteq G$ and $B \subseteq M$ are defined as follows:

$$A^{'} := \{m \in M | (g, m) \in I \texttt{ for all } g \in G\},$$

$$B^{'} := \{g \in G | (g, m) \in I \texttt{ for all } m \in B\}.$$

Then a formal concept of a formal context (G, M, I) is defined as a pair (A, B) with $A \subseteq G$, $B \subseteq M$, $A^{'} = B$ and $B^{'} = A$. The sets A and B are called extend and intend of the formal concept (A, B) . The set of all formal concepts $\mathfrak{B}(\mathbb{K})$ of a context (G, M, I) together with the partial order $(A_1, B_1) \leqslant (A_2, B_2) :\Leftrightarrow A_1 \subseteq A_2$ is called the concept lattice of context (G, M, I) .

In the Figure 2 the FCA algorithm *Incremental Lattice Builder* generated 11 formal concepts. In the lattice diagram, the name of an object g is attached to the circle and represents the smallest concept with g in its extent. The name of an attribute m is always attached to the circle representing the largest concept with m in its intent. In the lattice diagram an object g has an attribute m if and only if there is an ascending path from the circle labeled by g to the circle labeled by m. The extent of the formal concept includes all objects whose labels are below in the hierarchy, and the intent includes all attributes attached to the concepts above. For example the concept 7 has {*Building; Real Property*} as extend (the label *E:* in the diagram), and {*Postal Address; Environmental Problem Type;Owner;... etc.*} as intent (due to the huge number of attributes they are not shown in the figure).

4 Vector Space Representation of the Conceptual Model

The vector space model (VSM) for documents transformation to the vectors is a well-known representation approach that transforms a document to a weight vector. The most naive way to construct vector is based on the bag-of-words

approach, which ignores the ordering of words within the sentence, ignores their semantic relationships and uses basic occurrence information [15]. On the other hand, if we take the bag-of-words approach, the dimension of the vector space is based on the total number of words in the data set and we are faced with problems of big dimensionality reduction.In this paper, the process of dimensionality reduction and noise filtering is depicted in Figure 4. All presented processes are described in details below.

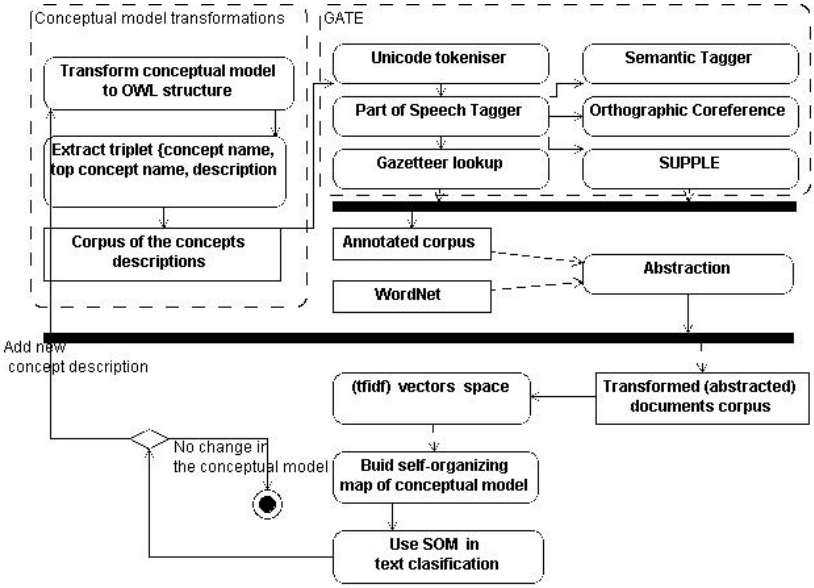

Fig. 3. The processes of dimensionality reduction and the conceptual model SOM design.

1. *Transform conceptual model.* As the first in this process, we transform conceptual model to the Web Ontology Language (OWL) structure. The motivation behind this step is that OWL is a standard for expressing ontologies in the Semantic Web (W3C recommendation).

2. *Extract triplet* is a process of data preparation. The triplet: concept name, the most abstract parent-concept name, and textual description of the concept are extracted. We received 1256 documents in the corpus where each document describes one concept. For example, the concept *Employee* has the following entry in the corpus: {***Concept**-Employee;* ***Top parent concept*** *- Involved Party ;* ***Description*** *- An Employee is an Individual who is currently, potentially or previously employed by an Organization, commonly the Financial Institution itself...* }. It is important to mention that there was only 9 *top parent* concepts:

(*involved party, products, arrangement, event, location, resource items, condition, classification, business*).

3. *GATE - Natural Language Processing Engine* is a well-established infrastructure for customization and development of NLP components [2]. It is a robust and scalable infrastructure for NLP and allows users to use various modules of NLP as the plugging. We briefly describe the components used for the concepts vector space construction. The *Unicode tokeniser* splits the text into simple tokens. The *tagger* produces a part-of-speech tag as an annotation on each word or symbol. The *gazetteer* further reduces dimensionality of the document corpus. *Semantic tagger* - provides finite state transduction over annotations based on regular expressions. *Orthographic Coreference* produces identity relations between named entities found by the semantic tagger. *SUPPLE* is a bottom-up parser that constructs syntax trees and logical forms for English sentences.

4. *Abstraction.* The basic idea of the abstraction process is to replace the terms by more abstract concepts as defined in a given thesaurus, in order to capture similarities at various levels of generalization. For this purpose we used WordNet [12] and annotated GATE corpus as the background knowledge base. WordNet consists of so-called synsets, together with a hypernym/hyponym hierarchy [4]. To modify the word vector representations, all nouns have been replaced by WordNet corresponding concept('synset'). WordNet '*most common*' synset was used for a disambiguation.

5. *Vectors space.* In our experiments we used vector space of the term vectors weighted by *tfidf* (term frequency inverse document frequency)[15], which is defined as follows:

$$tfidf(c,t) = tf(c,t) \times \log \frac{|C|}{|C_t|},$$

where $tf(c,t)$ is the frequency of the term t in concept description c, and C is total number of terms and C_t is the number of concept descriptions. $tfidf(c,t)$ weighs the frequency of a term in a concept description with a factor that discounts its importance when it appears in almost all concept descriptions.

5 Self-organizing Map of the IS Conceptual Model

Neurally inspired systems also known as connectionist approach replace the use of symbols in problem solving by using simple arithmetic units through the process of adaptation. The winner-take-all algorithms also known as self-organizing network selects the single node in a layer of nodes that responds most strongly to the input pattern. In the past decade, SOM have been extensively studied in the area of text clustering. It consists of a regular grid of map units. Each output unit i is represented by prototype vector $m_i = [m_{i1}...m_{id}]$, where d is input vector dimension. Input units take the input in terms of a feature vector and propagate the input onto the output units. The number of neurons and topological structure of the grid determines the accuracy and generalization capabilities of the SOM.

During learning the unit with the highest activation, i.e. the best matching unit, with respect to a randomly selected input vector is adapted in a way that it will exhibit even higher activation with respect to this input in future. Additionally, the units in the neighborhood of the best matching unit are also adapted to exhibit higher activation with respect to the given input.

As a result of training with our financial conceptual model corpora we obtain a map which is shown in the Figure 4. This map has been trained for 100,000 learning iterations with learning rate set to 0.5 initially. The learning rate decreased gradually to 0 during the learning iterations.

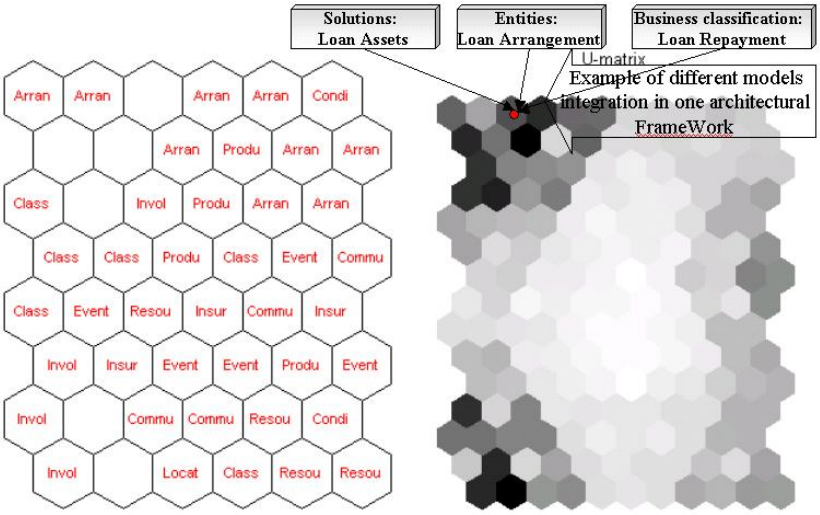

Fig. 4. SOM for the conceptual model. Labels: invol, accou, locat, arran, event, produ, resou, condi represents concepts: involved party, accounting, location, event, product, resource, condition.

We have expected that if the conceptual model vector space has some clusters that resembles conceptual model itself, then we can expect that the model will be easier understood compared with the model of more random structure. On a closer look at the map we can find regions containing semantically related concepts. For example, the right side top of the final map represents a cluster of concepts "Arrangement" and bottom right side "Resource items". Such map can be used as scarificator for any textual input. It always will assign a name of some concept from the conceptual model. Nevertheless, such map, as in the Figure 4, is difficult to use as an engineering tool. A hierarchical structure is more convenient for representation of the underlying structure of the concept vector space.

Figure 5 shows the concepts lattice computed from SOM shown in the figure 4. We obtain a list of 23 formal concepts. Each of them groups several neurons from SOM. We can find the grouping similarity of the neurons that are locate

in the neighborhood of each other. On the other hand some concepts makes a group of neurons that are at some distance form each other. The basic idea of this step is that we received a closed loop in the business knowledge engineering by artificial intelligent agent. The agent classifies all textual information with the SOM technique and then using FCA it builds hierarchical knowledge bases. For the details on how to apply FCA to the cluster analysis (SOM in our case) we refer to the paper [5]. The paper describes an algorithm which has been used in our research.

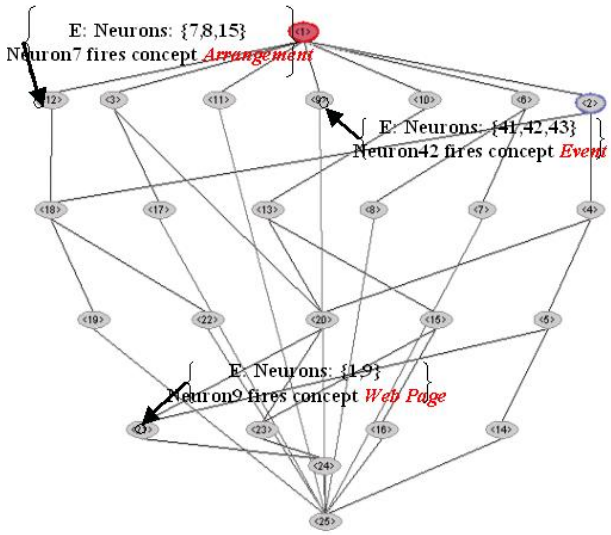

Fig. 5. Concepts lattice that has been received from the SOM presented in the Figure 4.

6 Experiment

Automatically generated concept lattice of document corpora is a useful tool for visual inspection of underlying vector space. Nevertheless, we would like to have a more rigorous evaluation of the lattice capability to depict conceptual model structure. For that purpose a classification accuracy (CA) measure has been used. CA simply counts the minority of concepts at any grid point and presents the count as classification error. For example, after the training, each map unit (and lattice node) has a label assigned by highest number of concepts mapped to this unit (Figure 4). As we can see in the Figure 4, the top left neuron mapped 4 concepts with the label *arrangement* and 2 with label *event*. Thus, classification accuracy for this neuron will be 66 %. For the whole concept map

we received $CA = 39.27\%$. It is not very high classification accuracy, but it can be used as a benchmark to compare other methods.

The framework that we presented in the previous sections generates document corpora lattice which can be visually compared with lattice generated from conceptual model. But as mentioned in the introduction, one of the objectives in this research was to find the techniques and tools of modeling that, in addition to the visualization, generate some artifacts for natural language interfaces. In this paper we suggest to reuse SOM as classification component. Each time the sentence is presented to the SOM component we have one activated neuron which is associated with one concept from the conceptual model. Additionally, we have the set of formal concepts associated with the activated neuron. Both, the label from activated neuron and the set of formal concepts can be used by some formal language generation engine (i.e. structured query language (SQL) sentence generator for querying databases).

The following experiment has been conducted to test this approuch. IBM WebSphere Voice Server NLU toolbox [8], which is a part of the IBM WebSphere software platform have been chosen as the competitive solution to the one suggested in this paper. SOM of the conceptual model and CL have been used as an alternative to the IBM WebSphere Voice Server NLU solution. We have taken the black box approach for both solutions: put the training data, compile and test the system response for the new data set. The data set of 1058 pairs *textual description:concept name* mentioned above were constructed to train the IBM NLU model. The same set has been used to get SOM.

Then a group consisting of 9 students has been instructed about the database model. They have the task to present for the system 20 questions about information related to the concept "Involved Party". For example one of the questions was: "*How many customers we have in our system*?" We scored the answers from the system as correct if it identified the correct concept "Involved Party".

Table 1. Concept identification comparison between IBM NLU toolbox and SOM of database conceptual model.

	CN=9	CN=50	CN=200	CN=400	CN=500
IBM NLU	36.82	17.26	14.82	11.15	8.22
SOM	46.73	30.70	27.11	20.53	18.83

At the beginning only 9 top concepts were considered i.e. all 1058 documents have been labeled with the most abstract concept names from the conceptual model. For example documents that described concepts "Loan" and "Deposit" are labeled with the concept name "Arrangement" because concepts "Loan" and "Deposit" are subtypes of the concept "Arrangement".

Next we increased the number of concept names that we put into the model up to 50. For example, documents that described concepts "Loan" and "Deposit" have been labeled with "Loan" and "Deposit" names. Then, number of concept

names has been increased up to 200, 400 and finally 500. Table 1 shows the results of the experiment. Column names show the number of concepts. The row named *IBM NLU* represents results for the IBM WebSphere Voice Server NLU toolbox. The row named *SOM* represents results for the SOM of the conceptual model that has been constructed with the method described in this paper. For the classification error, the proportion of the correctly identified concepts has been used.

As we can see, the performance of the IBM system was similar to the SOM response. For all cases i.e. IBM and SOM the performance decreased when the number of concepts increased.

7 Conclusion

Conceptual models and other forms of knowledge bases can be viewed as the products emerged from human natural language processing. Self-organization is the key property of human mental activity and the present research investigated what self-organization properties can be found in the knowledge base documentation. It has been suggested to build conceptual model vector space and its SOM by comparing concept lattice received from manually constructed conceptual model and concept lattice received from SOM of the conceptual model. We argued that if both concept lattices resemble each other then we can say that IS documentation quality is acceptable.

Presented architectural solution for the software developers can be labor intensive. The payoff of such approach is an ability to generate formal language statements directly from IS documentation and IS user utterance. We have shown that with the SOM and FCA we can indicate inadequateness of the concept descriptions and improve the process of knowledge base development. Presented methodology can serve as the tool for maintaining and improving Enterprise-wide knowledge bases.

There were many research projects concerning questions of semantic parsing i.e. the automatic generation of the formal language from the natural language. But those projects were concerned only with semantic parsing as separate stage not integrated into the process of software development. Solution presented in this paper allows us to integrate IS design and analysis stages with the stage of semantic parsing. In this paper we demonstrated that we can label documents and user questions with the conceptual model concept name. In the future we hope to extend those results by generating SQL sentences and then querying databases. The present research has shown that if we want to build comprehensible model then, we must take more attention in describing concepts by the natural language.

Acknowledgment. The work is supported by Lithuanian State Science and Studies Foundation according to High Technology Development Program Project "Business Rules Solutions for Information Systems Development (VeTIS)" Reg. No. B-07042.

References

1. Burg, J.F.M., Riet, R.P.: Enhancing CASE Environments by Using Linguistics. International Journal of Software Engineering and Knowledge Engineering 8(4), (1998) 435–448.
2. Cunningham, H.: GATE, a General Architecture for Text Engineering. Computers and the Humanities, 36, (2002) 223–254.
3. Ganter B., Wille. R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin-Heidelberg, (1999).
4. Hofmann, T.: Probabilistic latent semantic indexing. In Research and Development in Information Retrieval, (1999) 50–57.
5. Hotho, A., Staab, S., Stumme, G.: Explaining text clustering results using semantic structures. In Principles of Data Mining and Knowledge Discovery, 7th European Conference, PKDD 2003, Croatia. LNCS. Springer (2003) 22–26.
6. Hung, C., Wermter, S., Smith, P.: Hybrid Neural Document Clustering Using Guided Self-organisation and WordNet. Issue of IEEE Intelligent Systems, (2004) 68–77.
7. IBM. IBM Banking Data Warehouse General Information Manual. Available from on the IBM corporate site http://www.ibm.com (accessed July 2006).
8. IBM Voice Toolkit V5.1 for WebSphere Studio. http://www-306.ibm.com/software/ (accessed July 2006).
9. Kaski, S., Honkela, T., Lagus, K., Kohonen, T.: WEBSOM self-organizing maps of document collections. Neurocomputing, 21, (1998) 101–117.
10. Kohonen, T.: Self-Organizing Maps, Springer-Verlag, (2001).
11. Lagus, K., Honkela, T., Kaski, S., Kohonen, T.: WEBSOM for textual datamining. Articial Intelligence Review, 13 (5/6) (1999) 345-364.
12. Miller, G.A.: WordNet: A Dictionary Browser, Proc. 1st Int'l Conf. Information in Data, (1985) 25-28.
13. Ryan, K.: The role of natural language in requirements engineering. Proceedings of IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, (1993) 240–-242.
14. Rolland, C., Proix, C.: A Natural Language Approach to Requirements Engineering. 4th International CAiSE Conference, Manchester UK, (1992) 257-277.
15. Salton. G.: Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer. Addison-Wesley, (1989).
16. Valtchev, P., Grosser, D., Roume, C., Rouane H. M.: GALICIA: an open platform for lattices. In A. de Moor B. Ganter, editor, Using Conceptual Structures: Contributions to 11th Intl. Conference on Conceptual Structures (2003) 241--254.