

Semantic Structure Transition with Elapsed Time ^{*}

Katsuaki Tanaka
University of Tokyo
4-6-1 Komaba, Meguro-ku
Tokyo, Japan
katsuaki@ai.rcast.u-
tokyo.ac.jp

Mina Akaishi
University of Tokyo
4-6-1 Komaba, Meguro-ku
Tokyo, Japan
akaishi@ai.rcast.u-
tokyo.ac.jp

Koichi Hori
University of Tokyo
4-6-1 Komaba, Meguro-ku
Tokyo, Japan
hori@ai.rcast.u-
tokyo.ac.jp

ABSTRACT

To enable cooperative problem solving by people and computers it is necessary to share the transition of objects, because objects at a given time are the accumulation of their transitions. With this perspective, we defined semantic structure as the organization of problem-solving systems. Semantic structure transition with elapsed time reflects how and what knowledge is used to solve a problem. Based on this argument, we constructed two systems for a semantic structure. The first system was designed to construct and manage the semantic structure, and the second was designed to derive the semantic structure transitions.

Keywords

information sharing system, semantic structure transition, topic detection, document stream processing

1. INTRODUCTION

Semantic computing is information technology based on semantics shared by people and computers so that they can cooperate to solve problems. To share semantics a common representation is required. This is often represented as a tree or a graph, called a semantic structure in this paper. Semantic structure is manipulated dynamically by people and computers during the problem-solving process, resulting in changes in form as time passes.

In this paper, we show that this transition of semantic structure is important in semantic computing because its structure is the accumulation of transitions. Transitions are caused by operators applying their knowledge. Therefore, people and computers need to share the semantic structure at any given time, as well as its transitions.

Based on this description, we propose two systems, the first is a system to animate the semantic structure and the second is a system to derive the semantic structure transitions from the problem-solving records.

2. RELATED WORK

2.1 Semantic web service

^{*}(Produces the WWW2005-specific release, location and copyright information). For use with www2005-submission.cls V1.4. Supported by ACM.

The semantic web is a project to store semantics that are useful for problem solving in the web. Semantic web service [5] is a project to compose a problem-solving system with semantics prepared by the semantic web. For these purposes, OWL [4] and OWL-S [2] are being developed to represent the structure of objects and services.

The semantic web service creates a system of a user's requirements automatically by combining elemental semantic web services. It may change its behavior using parameters in the process, but the system once created is not changed during the problem-solving process. This means that semantics to solve a problem are not shared between a user and the semantic web system. It is not suitable for a problem where the user's intention is not clear in the initial stage of problem solving.

2.2 Chance discovery

Chance discovery is a method that forms the structure from data and interacts with the structured data and people to solve a problem, concretely to discover a chance. Structured data is not a semantic structure described in this paper, it is a structured expression of explicit data at a given time.

In chance discovery[7][6], people are involved in a problem-solving process and the data structure is reconstructed at any point during this process. Therefore, it can be said that semantics is shared in the problem-solving process. By this means, interaction between people and the structured data is an important part of chance discovery, but transition of these interaction results is not a focus. In addition, structured data is constructed from data at a given time, it does not manage the transition of data and interactions.

Structured data is generally represented as a graph that includes only explicitly expressed nodes. In the chance discovery process, however, people can extract the background structure as groups of nodes.

3. SEMANTIC STRUCTURE

To enable cooperation of people and computers in the problem-solving process they must share a common representation, with a structure that reflects the problem structure. People and computers operate using this to solve a problem. Such a structured object for cooperation we call a semantic structure.

3.1 Example of a problem-solving process

Generally, a tree or graph structure is used to represent the semantic structure. In this section, we define an object

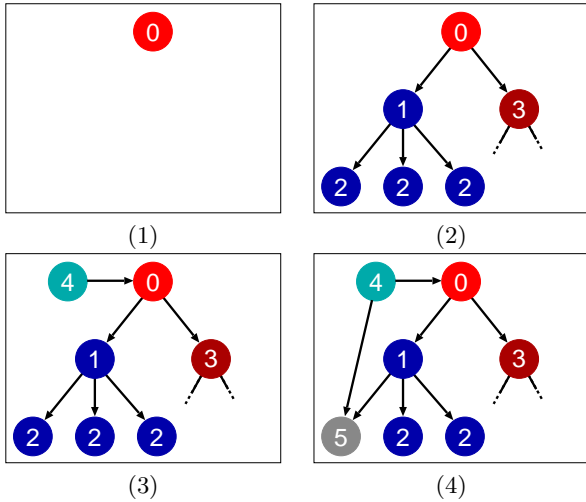


Figure 1: Example of semantic structure transition

of a problem-solving process as a semantic structure and give an example of how semantic structure is manipulated. In the next section, we extend the concept of semantic structure from an object to an organization of problem-solving systems.

In the section that follows, we use the writing of this paper as an example of a problem-solving process, that is a semantic structure operating process.

First, we conceived the semantic structure and its transition is important in problem solving. Node 0 in figure 1 denotes this intention. We developed a prototype system where people and computers cooperate to solve problems (node 1). Next, we wrote a description of the system (node 2). We also developed another system that derives topic transitions from the problem-solving records (node 3). Then we planned to write this paper (node 4). To describe these systems in detail the limit of space is insufficient so some of node 2 is deleted (as node 5). It follows that nodes 1 and 3 are refined from the perspective of nodes 0 and 4. In these steps the semantic structure is transformed as time passes.

3.2 Cause of semantic structure transition

In example above, knowledge (ideas, intentions, etc.) is used to transform the semantic structure. Knowledge is not a semantic structure itself, but knowledge is something that transforms semantic structure in line with the problem solver's intention. Semantic structure exists as a result of knowledge application. Therefore, things required to be shared between people and computers are not only the semantic structure but also the knowledge to construct it.

3.3 Semantic structure as an organization of problem-solving systems

In section 3.1, we described semantic structure as a single object. It is not a static object, but dynamically transformed. We manipulate it as a subject. In this case, one problem-solving system was formed as the entire semantic structure and its manipulator.

In a large-scale problem, manipulation is not performed on the whole structure but on a part of the structure, especially at each node. For example, in writing a paper, we

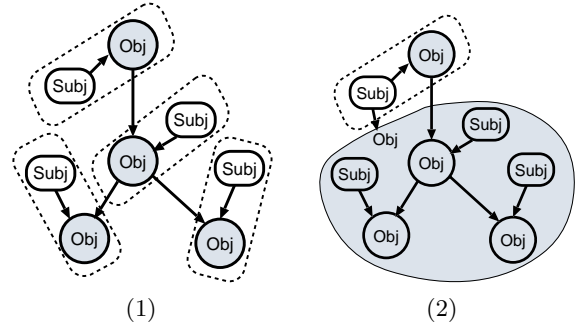


Figure 2: Internal object and external object

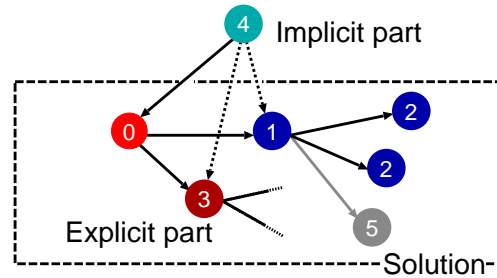


Figure 3: Nodes not included in a solution

must focus many points from various positions. Therefore, it is natural that there exists a subject for each node of a semantic structure. This means that each node contains a subject and an object, and so each node is a problem-solving system.

Therefore, the semantic structure is an organization of problem-solving systems. Each problem-solving system has a subject and an object, and a subject manipulates its object based on its knowledge.

In a problem-solving process, many standards are applied to the semantic structure. For example, in writing a paper, we not only perform text writing (as in node 2 above) but also thinking opinion, structuring paragraphs (as for node 4), and proofreading text, etc. Therefore, in this case, objects of each subject are not only one node of the semantic structure (figure 2(1)) but also group of nodes (figure 2(2)).

3.4 Semantic structure and a solution of a problem

In general, the entire semantic structure is not required as solution of a problem. For example, writing this paper, semantic structure includes a node that represents the structuring of paragraphs (node 4 above), the limit on the number of pages, the date of the deadline, etc. However, these nodes are not expressed explicitly as this paper in the solution, but included implicitly as part of the semantic structure (figure 3).

Semantic structure includes all problem-solving processes, therefore the user is required to decide how to derive the solution from the semantic structure. In most situations, lower nodes linked to each other are suitable as a solution.



Figure 4: Example of problem structure represented as problem-solving units

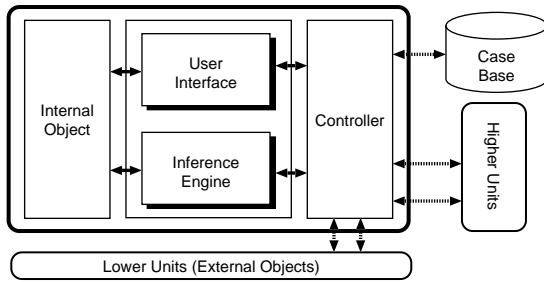


Figure 5: Components of a problem unit

4. SEMANTIC STRUCTURE MANAGEMENT SYSTEM

We constructed a semantic structure management system based on the above description [11][8]. The system consists of problem units and the case base. A problem unit represents a node of the semantic structure. The case base contains structures of problem units constructed previously.

Figure 4 shows an example of a problem structure display-user interface. Problem structure is represented as the organization of problem units. The displayed problem structure corresponds to the semantic structure.

This example problem structure represents a communication subsystem simulator design problem for a small space satellite, "XI-IV". "XI-IV" is a development project of the Intelligent Space Systems Laboratory in the University of Tokyo.

4.1 Problem unit

One problem unit corresponds to one node of the semantic structure. Figure 5 shows the components of a problem unit. Each problem unit has subjects, users assigned to the problem, and a built-in inference engine. Each unit also has an internal object represented by couple, attribute and value, and a controller to communicate with any other part of the system. Figure 6 shows the main user interface of

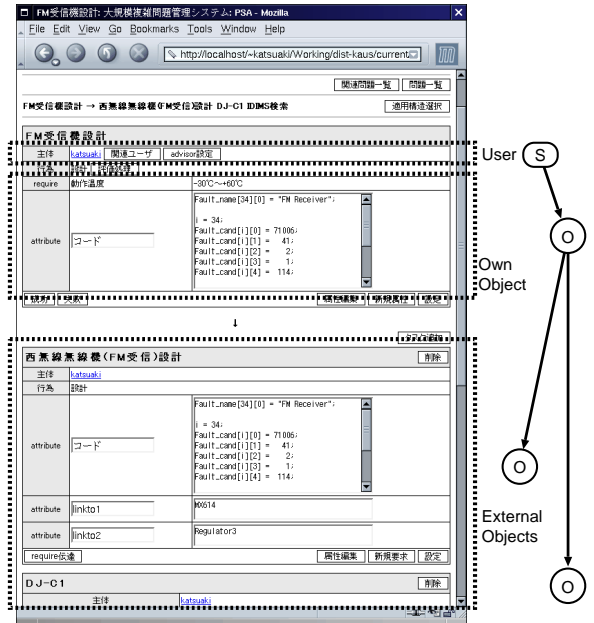


Figure 6: Main user interface of a problem unit

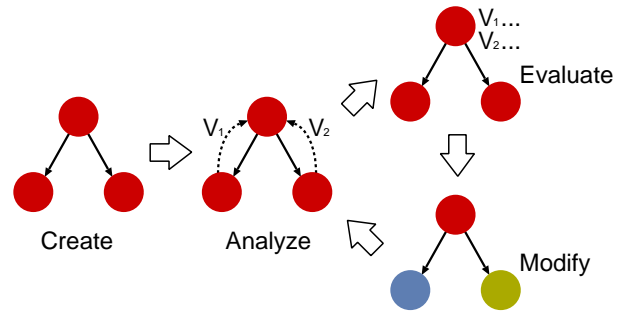


Figure 7: Problem-solving process

a problem unit. With this interface, a user controls the behavior of a unit.

Each problem unit has SOAP[1] interface, so a unit can handle web services like as lower problem units.

4.2 Problem management process

Figure 7 shows the tasks of a problem unit. In this flowchart, a problem unit is created by another unit, the whole problem structure being constructed top-down. In addition, the problem structure is managed by each part of the problem units.

The tasks of a problem unit shown in figure 7 are divided into two parts. The first is to create the lower structure, and the second is to analyze, evaluate and modify its own object and lower units, as external objects.

To create the lower units, either the subject (the user or the built-in inference engine) selects an existing structure from the case base, or the user creates new units. The user creates lower units based on his or her own intention and knowledge. On the lower structure selection screen (figure 8(a)), the user can read the intention of the existing struc-

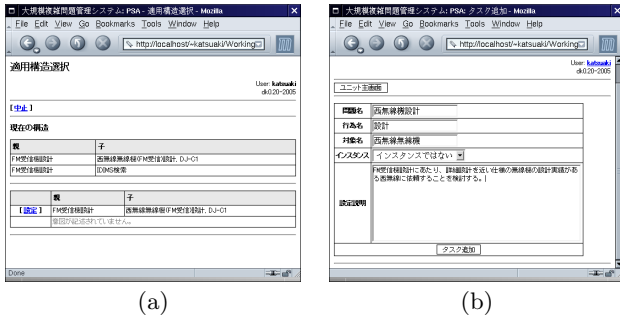


Figure 8: (a) Selecting the lower structure, (b) Creating a lower unit

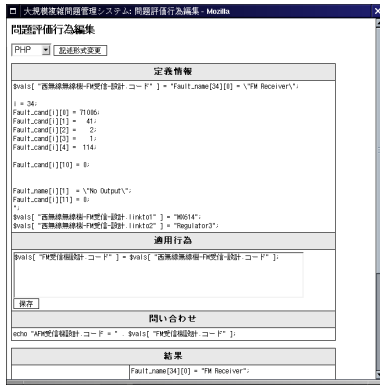


Figure 9: Object evaluation code writing interface

ture. On the lower unit creation screen (figure 8(b)), users are required to write their intentions. The inference engine selects a structure based on the numerical priority of the current system. In the future, we plan to redesign a problem unit to enable the inference engine to select a lower structure based on other grounds, such as a logical description that computers and people can share.

The second task of a problem unit is evaluating and modifying objects to solve the assigned problem. This task consists of three parts. First is analyzing objects, which is performed by retrieving the attributes and values of the objects. The second is evaluating these values. The third is modifying these values and adding or deleting attributes. A user performs these tasks using the user interface, shown in figure 6. This interface provides functions that add and delete attributes, modify values, and copy attributes and values from other objects, with the user’s knowledge.

The built-in inference engine performs this based on knowledge prepared beforehand, represented as declarative rules or procedural codes. Figure 9 shows the evaluation code writing screen. The attributes and values of each object are provided as input data, and the evaluation results are applied to each object.

Significantly describing the user’s intention and knowledge in advance enables an increase in the computer’s component of the tasks. On the other hand, a user could perform all the tasks of a problem-solving process with own knowledge. A problem unit that provides the user interface records knowledge, the context of it applied, and the result as structure of

lower problem units. In this way, the users provide knowledge as cause of semantic structure transition, and the computers records semantic structure as composition of knowledge. In addition, the computers (problem units) can indicate candidates of semantic structure transitions based on recorded context. As thus described, problem-solving tasks and semantic structure transforming procedures are shared in problem-solving units between the users and the computers.

4.3 Solution of a problem

The organization of problem units represents a problem structure that is a reflection of the semantic structure. As described in section 3.4, some part of the object structure in a semantic structure is not used in a solution. Therefore, the solution of a problem could be derived from the objects in problem units. This derivation process could be done after each problem unit found a solution; or, if it is clear in advance whether or not a problem unit is included in the solution, the process could be performed by adding an attribute to the problem unit’s object beforehand.

Otherwise, there exists another approach, to make each evaluation procedure of problem units return the pair, attribute and value, that comprise a solution, and to collect the entire object of a solution at the top problem unit.

5. ACQUISITION OF SEMANTIC STRUCTURE TRANSITION

We constructed a system that derives topic structure transitions from the problem-solving records. Topics discussed in the problem-solving process are the problem solver’s focal points which reflect his or her view of the problem structure. Therefore, topic structure reflects semantic structure and topic structure transitions reflect semantic structure transitions.

With derived transitions, the user can consider what causes these transitions. Structure transition is useful as a candidate for the lower structure of a problem unit in the problem management system.

5.1 Topic structure transition extraction procedure

We detect the transition of topics in the following three steps[12]:

1. Grouping documents depending on their creation time.
2. Forming clusters within each group.
3. Detecting the relationships between clusters over the group.

For a document \mathbf{d} , let $c(\mathbf{d})$ denote its creation time. Suppose a set D of documents is given and let $E(D)$ denote the creation time of the earliest document in D , i.e., $E(D) \equiv \min_{\mathbf{d} \in D} c(\mathbf{d})$. Similarly, let $L(D)$ denote the creation time of the latest document in D . Then, we set a document group at every time interval T such that

$$T \equiv \frac{L(D) - E(D)}{N}$$

Based on the time interval T , we form N document groups D_1, D_2, \dots, D_N , where

$$D_i \equiv \{\mathbf{d} \mid c(\mathbf{d}) \leq E(D) + i \cdot T\}$$

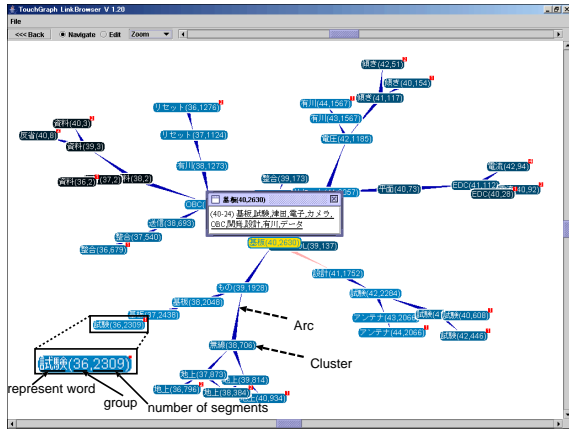


Figure 10: Screen of the transition acquisition system

Note that $D_1 \subseteq D_2 \subseteq \dots \subseteq D_N = D$.

Because each document often includes more than one topic, we must segment each document.

First, Japanese language morphological analysis is applied each document with Chasen[3]. Technical terms are often used in the documents, so we set to analyze a series of nouns are treated as one noun. Then windowing is performed with a window size of 200 characters, and the overlap is 67 characters to make text segments. If boundary between segments exists on a word, each length of segment is extended to include the word.

Next, to make clusters of these segments for each document groups, we make document vector of each text segment by couples of words and its frequency, and perform clustering these document vectors by Ward's method[9] with GETA[10]. We divided the text segments into 50 clusters for each document group. If the upper bounds to the number of clusters are fixed and a new cluster is added composed of new elements far from existing clusters, it is necessary to reorganize the existing clusters to reduce their number. As a result, clusters that are close to each other are collected into one cluster. Therefore, with the goal that similar items be organized into the same clusters as time passes, and newly added topics form new clusters, we fixed the number of clusters.

Then, to measure the similarity between clusters, function $sim(C_{n,i}, C_{m,j})$ is defined as follows:

$$sim(C_{n,i}, C_{m,j}) = \frac{|C_{n,i} \cap C_{m,j}|}{|C_{n,i}|} \quad (1)$$

where n and m are group numbers such that $1 \leq n \leq m \leq 50$ and $C_{n,i}$ denotes the i th cluster at group n .

The following Jaccard similarity measure (2) is usually used to quantify the similarity of the clusters:

$$Jaccard(C_{n,i}, C_{m,j}) = \frac{|C_{n,i} \cap C_{m,j}|}{|C_{n,i} \cup C_{m,j}|} \quad (2)$$

However, when $C_{n,i} \subseteq C_{n+k,j}$ ($k \geq 1$), the topic corresponding to $C_{n,i}$ has been merged with other topics to form a broader topic, $C_{n+k,j}$. Therefore, we used (1) as our similarity measure.

Based on this similarity function, the relation between clusters belonging to adjoin document groups is made vis-

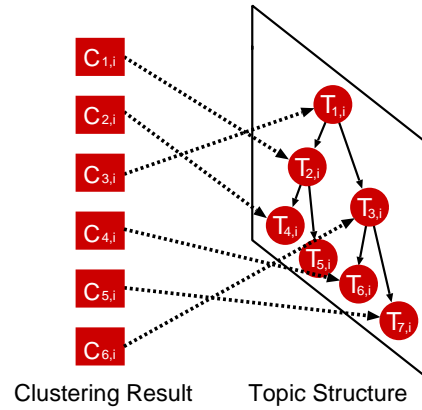


Figure 11: Mapping clusters to topics

able. We use the TouchGraph LinkBrowser¹ for visualization (figure 10).

The similarities between clusters of adjoining groups are calculated using the expression (1), and clusters with similarities of 0.3 or more are linked.

The distance of the cluster $C_{n,i}$ and $C_{n+1,j}$ is calculated by the following expressions:

$$\begin{aligned} distance(C_{n,i}, C_{n+1,j}) \\ \equiv -A \cdot sim(C_{n,i}, C_{n+1,j}) + B \end{aligned} \quad (3)$$

When visualizing relationships between clusters, the distance between nodes without links is taken as zero.

One of the feature words of the cluster was used as the label for each node. The numerical values of a label represent the number of the document group that the cluster belongs to and the number of text segments that the cluster contains. The direction of the arrow of a link shows the chronological direction. When a user selects a cluster the feature words of the cluster are displayed in a pop-up window, then the user can refer to documents that belong to the cluster.

5.2 Extraction of semantic structure transition

To extract topic structure from clusters, it is necessary to map clusters to topics. Therefore, mapping function F

$$T_{k,i} = F(C_{n,i})$$

is needed (figure 11) where the i th document group is D_i , k th node of topic structure of D_i is $T_{k,i}$, n th cluster of D_i is $C_{n,i}$.

F is not defined in the system, users are required to define F by reading the detail of the topics and clusters, or by using text similarity, etc. Each label of a cluster graph (figure 10) shows a representative word of the cluster. It is selected based on TFIDF measurement function used by [13]. However, it is not easy to understand the meaning of a cluster from only one representative word. To understand the meaning of a cluster precisely the user needs to read text segments that belong to it. Other representative words and a HTTP link to text segments are shown as a pop-up window similar to figure 10, the user can read the details

¹<http://www.touchgraph.com/>

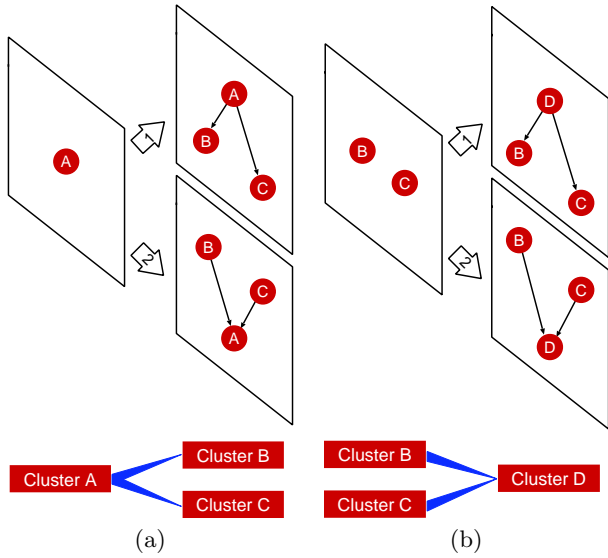


Figure 12: Relation between topic transition graph and topic structure

of a cluster following the link. When users decide that a cluster reasonably represents a node of the semantic structure, they can set it to be a problem unit in the semantic structure management system.

Figure 12 shows the elemental relations between the cluster graph and topic structure transitions. Clusters B and C belong to the same document group.

As described above, document group sets correspond to the creation time of the document, figure 12(a) shows cluster A changed into cluster B and C as time elapsed. This means text segments that belong to cluster A are divided into B or C. In this situation two topic structures could be considered. The first case is where A is an upper concept of B and C, because A includes text segments of B and C. The second case is that A is a lower concept of B and C, because a part of B and a part of C make A. The decision as to which case represents the real topic structure transition depends on the user's judgment.

Figure 12(b) shows clusters B and C merged into cluster D as time elapsed. This means the text segments that belong to cluster B and C are collected into cluster D. In this situation it is also possible to consider two topic structures. The first is that D is an upper concept of B and C, because D includes B and C. The second case is that D is a lower concept of B and C, because a part of B and a part of C make D. The judgment as to which case represents the real topic structure transition depends on the mapping function F and on the user's decision.

6. EXAMPLE OF SYSTEM UTILITY

In this section, we describe the construction process of the "XI-IV" communication subsystem simulator as an example of the utility.

To simulate the communication subsystem, there exists a method that constructs a semantic structure of the communication subsystem and animates it as the simulator using a semantic structure management system. However, because speed of simulation is required, we defined the requirement

of the problem as the generation of a C-source code to simulate the communication subsystem.

First, construction of the structure of the communication subsystem is required. This purpose is aided by the application of the topic structure transition acquisition system for documents, which were created in the XI-IV design process. Topic structure transition indicates a knowledge of XI-IV design.

The XI-IV project has 398 minutes from January, 2000 to December, 2002. All were written in Japanese. The average length of the minutes is 2703 bytes. Every minute includes the creation date and the contents of the discussions. As a result, a graph with 492 intersections was obtained.

We could obtain an overview of topic transition with the graph. Moreover, we added a function to the visualization section of the system, to limit displaying nodes that include text segments that contain designated keywords. With this function, a topic transition that related to the communication subsystem was obtained and applied to the construction of semantic structure, with the management system.

In the semantic structure management system, we assigned a problem unit to each functional part of the communication subsystem, and describe the C-source code fragment that simulates its function. Each code fragment was collected to an upper problem unit and the upper unit merged these fragments. Finally, the entire simulation source code was collected in the top problem unit and this was the solution to this problem.

7. CONCLUSION

To enable cooperation of people and computers for problem solving, it is necessary to share information concerning the target objects; and to share the transition of objects, because objects at a given time are the accumulation of transitions. Transition is caused by the problem solver's knowledge. Knowledge is not a representation of a static structure at a given time but the cause of dynamic transition of the structure.

With this perspective, we defined the semantic structure as an organization of problem-solving systems. In the semantic structure people and computers cooperate to solve a problem by sharing information about objects and tasks. Semantic structure transition with time reflects what and how knowledge is used to solve a problem.

We constructed two systems for the semantic structure. The first system was designed to construct and manage the semantic structure, and the second was designed to derive the semantic structure from problem-solving records. The latter system provides partial structure transition knowledge to construct the semantic structure. The former system constructs the semantic structure with this knowledge, and operates the semantic structure with the knowledge.

While applying topic transition to construct the communication subsystem simulator, it was little difficult to find nodes that have similar granularity in the topic transition graph and the problem structure. We planned to resolve this difficulty by improvement of clustering method in topic transition acquisition system. Current system employs Ward's method for clustering. In this method, to reduce number of clusters leads to make large clusters that include small clusters. It has not always been upper concept of smaller cluster, but only group of them. Therefore, we are trying to another method of clustering to make clusters of text

segments based on structure of a document.

Currently, users are required to perform tasks other than operating the semantic structure, e.g. searching existing knowledge, and mapping topic and semantic structure. In the future, we plan to reduce these tasks and refine the concepts and systems so that the problem-solving process will work better.

8. REFERENCES

- [1] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte, and D. Winer. Simple object access protocol (soap) 1.1. <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>, 2000.
- [2] D. Martin. Owl-s: Semantic markup for web services. <http://www.w3.org/Submission/OWL-S/>, 2004.
- [3] Y. Matsumoto, A. Kitauchi, T. Yamashita, Y. Hirano, H. Matsuda, K. Takaoka, and M. Asahara. *Japanese Morphological Analysis System ChaSen version 2.2.1*, 2000.
- [4] D. L. McGuinness and F. van Harmelen. Owl web ontology language overview. <http://www.w3.org/TR/owl-features/>, 2004.
- [5] S. McIlraith, T. Son, and H. Zeng. Semantic web services. *IEEE Intelligent Systems (Special Issue on the Semantic Web)*, pages 46–53, March/April 2001.
- [6] Y. Nara and Y. Ohsawa. Exploring collaboration topics from documented foresight of experts. In *Proc. of 8th International Conference, Knowledge-Based Intelligent Information and Engineering Systems*, volume 2, pages 823–830, 2004.
- [7] Y. Ohsawa and Y. Nara. Modeling the process of chance discovery by chance discovery on double helix. In *Proc. of AAAI Fall Symposium on Chance Discovery*, pages 33–40, 2002.
- [8] S. Ohsuga. Significance of ai in solving problems that can not be foreseen beforehand. *Journal of the Japanese Society of Artificial Intelligence*, 19:478–498, July 2004.
- [9] S. Sharma. *Applied Multivariate Techniques*. John Wiley & Sons, 1996.
- [10] A. Takano, S. Nishioka, and O. Imaichi et al. Development of the generic association engine for processing large corpora. In *19th IPA Technical Forum*, 2000.
- [11] K. Tanaka and S. Ohsuga. Model-based creation of agents and distribution of problem solving. In *Proc. of the 2001 International Conference on Intelligent Agent Technology*, October 2001.
- [12] K. Tanaka and A. Takasu. Topic change extraction from problem solving records. In *Proc. of the 8th World Multi-Conference on Systemics, Cybernetics and Informatics*, July 2004.
- [13] Y. Yang, T. Ault, T. Pierce, and C. W. Lattimer. Improving text categorization methods for event tracking. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 65–72, 2000.