Knowledge Sharing and Organization by Multiple Ontologies

Motoyuki Takaai, Hideaki Takeda, and Toyoaki Nishida

Graduate School of Information Science,

Nara Institute of Science and Technology

Address: 8916-5,Takayama,Ikoma,Nara 630-01 JAPAN

Phone: +81-7437-2-5265 Fax: +81-7437-2-5269

E-mail: motoyu-t@is.aist-nara.ac.jp

Abstract: In this paper, we introduce ontology-centric knowledge organization approach to realize knowledge base system for sharing and reuse of knowledge. Since ontology is an intermediate level of information representation between the model and media level of information, it can work to bridge multiple models and multiple users. To share personal information among users, they must share their manners of conceptualization which form personal ontologies. Here we introduce two systems. First, we show knowledge base system for engineering called Designers Amplifier that organizes poorly-structured documents. Next, we show the multiple ontology management system called Donden that enables users to edit multiple ontologies and helps users to integrate ontologies.

Keywords: **knowledge organization, multiple ontologies, ontology development environment**

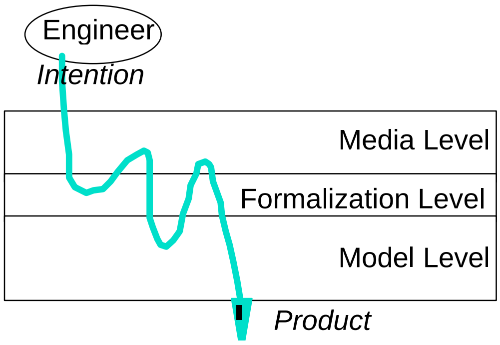

Figure 1: Levels of Information

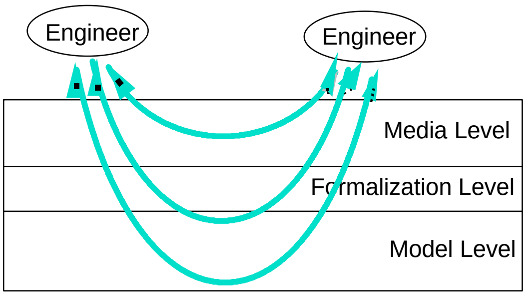

Figure 2: Levels of Information

1. Introduction

We get much knowledge from various sources in productive activities. Especially in the large and complex activities, knowledge from others and past experience is very important. But knowledge is often poorly-organized or unknown to get. Poorly-organized knowledge is not useful to share among users and to reuse.

Here we discuss the ways to reuse and to share knowledge among people. Ontology, an explicit specification of conceptualization, organizes knowledge and makes us deep-understanding. Knowledge of different people is represented by different manner which is based on their ontologies.

In this paper, we discuss ontology-centric knowledge organization, the way of knowledge sharing and reuse based by ontology. And we discuss the ways to manage multiple ontologies and to integrate them.

In the section 2, we show ontology-centric knowledge organization. In the section 3, we show our system called Designers Amplifier based on ontology-centric knowledge organization. In the section 4, we show the system to manage multiple ontologies and to help to integrate them. In the last section, we summarize this paper.

2. Ontology-Centric Knowledge Organization

In this section, we discuss how human organize heterogeneous information especially in engineering, and show an architecture for knowledge organization.

2. 1. Information Levels

Most crucial part of information organization is not in well-structured information but in unstructured one. Engineers use various kind of information from physical models to natural languages. We have usually paid attention to so-called “model”, e.g., 2/3D model, some mathematical model like thermodynamics and kinematics. Since models are well investigated and understood, there are many books and studies to handle them. But models are not sources but results of knowledge organizing formed after much engineering and scientific effort. Individual engineers try to capture unstructured information, and some excellent and common organization of information can become models. Much of their work is done with unstructured information.

Figure 1 illustrates levels of information to deal with. The lowest level is “model” level, i.e., information represented as models. The middle level is “formalization” level, where information is represented by some formal language. The difference between the model and formalization levels is that the former has syntax and semantics, and the latter has only syntax. The top level is “media” level where information representation is only

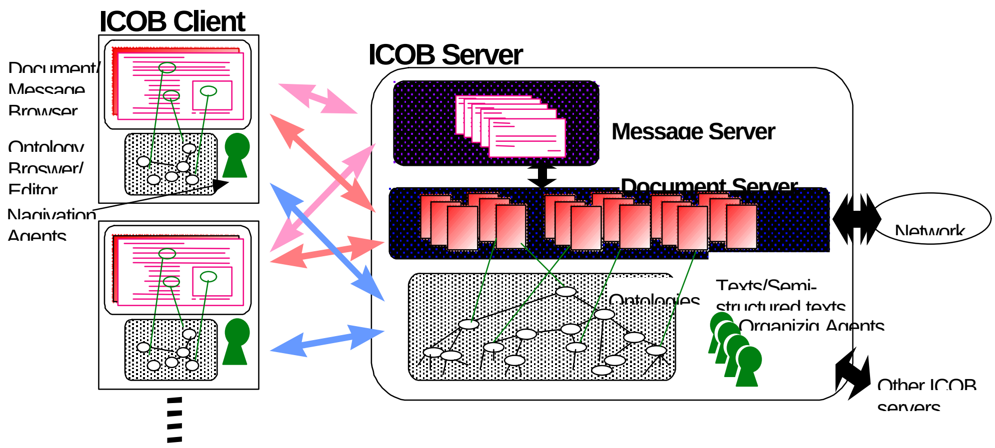

Figure 3: Architecture for Intelligent Corporate Base

restricted by property of medium which is represented in.

All of these three levels are needed for engineering work, because real engineering problems are not closed in a single model but should be solved in multiple models. In general, the lower levels can deal with only specific domains and can offer specific solutions in those domains. On the other hand, the upper level can deal with wider domains but can offer less specific solutions. The middle level can offer interface between both levels.

Design process is performed by penetrating the media, formalization, and model levels in order (see Figure 3). When domain of design is quite new, much effort of design is done in the media level before going the lower levels. On the other hand, when domain is well investigated, the media level is relatively less important and much work is done in the model level.

The same situation happens when engineers communicate to each other (See Figure 3). They are sometimes devoted to communication in the model level, and sometimes wandering from the model level to the media level.

2. 2. Role of Ontology in Knowledge Representation

We discuss the formalization level as "ontology" in this paper. Although ontology cannot capture all functions of the formalization level, it can offer one way to connect the media and model levels.

Ontology is a term in philosophy and defined as "the branch of metaphysical enquiry concerned with the study of existence itself" [1]. In AI community, an ontology is defined as "an explicit specification of conceptualization" [2] which is intended to use as base for knowledge representation. Ontology is represented as systems of symbols in computers, i.e., symbols and their relations represents concepts and their relations. Ontology commits only declarative nature of systems of concepts because its purpose is to provide consent for knowledge representation, e.g., identification of concepts. Therefore ontology is adequate as representation for the formalization level.

2.3. An Architecture for Ontology-Centric Knowledge Organization

The above discussion leads us an approach for knowledge based systems called *ontology-centric knowledge organization*. In this approach, roles of ontologies are bridges between multiple models, between the model and the media level representation, and multiple users.

Figure 3 shows an architecture for engineering knowledge and communication bases called ICoB (Intelligent Corporate Base) which is based on ontology-centric knowledge organization approach.

There are servers which contains shared documents and communication messages, and clients each of which an engineer uses. Users can retrieve or submit documents or communication messages by using shared and private ontologies. The ICoB servers organize documents and messages by using ontologies which consist of shared and users' ontologies. At the same time, they can extend and their ontologies by referring and comparing shared and other private ontologies. The latter process corresponds organization of information we discussed in the previous section. ICoB clients and servers can have some facilities to assist users' information organization.

There are some projects and studies related to this approach. For example, PACT Project[3] is a good example for integration of engineering process with multiple engineers. Ontolingua Server Project at Stanford[4] is studying an testing collaborative ontology construction. Enterprise modeling (e.g., Ref. [5]) is partially related to it. While these systems expect knowledge engineers who are different from engineers, our aim is to provide environments for engineers which can be evolved by activities of engineers themselves.

3. Designers Amplifier

Designers Amplifier is the knowledge base system which supports engineers. The systems to supports engineers to access design knowledge should have the following properties;

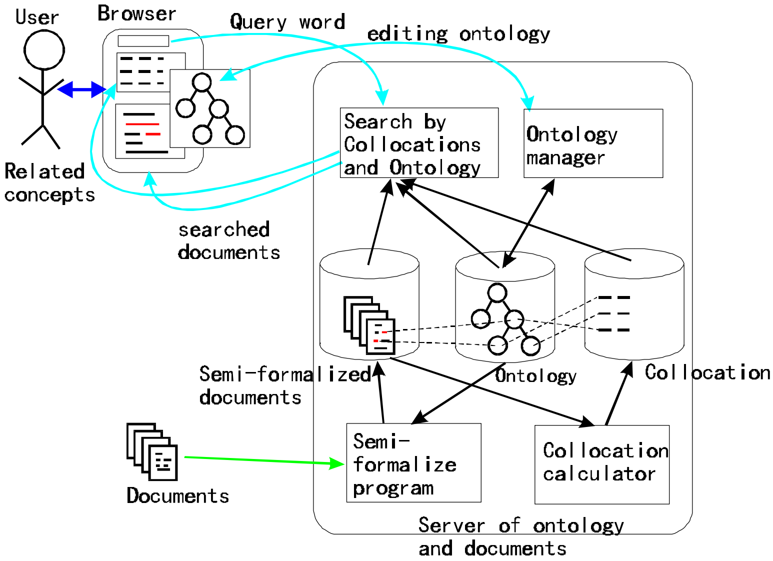

Figure 5: The architecture of Designers Amplifier (Ver. 1.0)

1. adapt *wide* situations,
2. assist designers *continuously*, and
3. be *intimate* to designers.

The first requirement means that the systems should cover as wide domains as possible in design processes. As mentioned in the previous section, model-centered systems provide deep information in narrow domains, while the systems should adapt various situations that designers would encounter. The second requirement is needed to realize knowledge-level support for designers. The systems should always track what designers are doing in order to estimate their intention. The third one is somewhat additional requirement to the second one. To realize continuous support to designers, designers should be willing to use the systems.

Designers Amplifier is our current prototype system to support designers from the media level to the model level totally. It is designed to evaluate how the current information technologies can contribute to support designers in ontology-centric

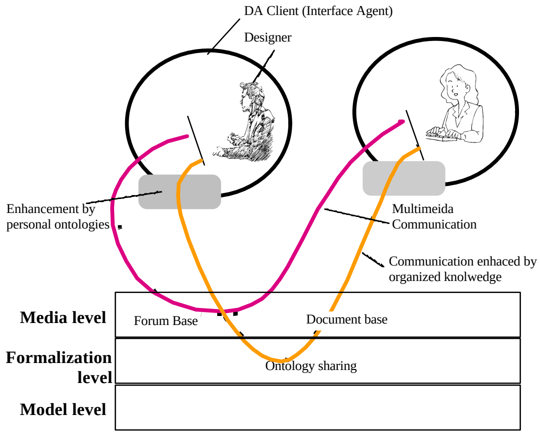

Figure 4: Concept of Designers Amplifier

knowledge organization approach. The basic idea of this system is agent that enhances designers' access to information sources and other designers (see Figure 4). Designers can access information sources with support of Designers Amplifier, e.g., by adding related information, at the same time Designers Amplifier can collect designers' behavior by monitoring their information access. Information access and communication is sometimes purely in media level, sometimes in both media and formalization levels, and sometimes in media, formalization, and model levels. Designers Amplifier mainly supports media and formalization levels, and can connect model-centered systems.

3.1. Designers Amplifier (Ver. 1.0)

The current implementation (Designers Amplifier Ver. 1.0) is a simple system in which only accumulated textual documents are handled. The main function is to organize documents by ontologies. Each document is associated to ontologies, i.e., it is modified to hyper texts which link concepts in the documents and concepts in ontologies.

Designers Amplifier also estimates relations among concepts in ontologies by calculating co-occurrence of concepts in the whole set of documents. Each user can retrieve documents by using concepts in ontologies and know what concepts are included in each document. Furthermore she/he can modify ontologies by referring documents. Modification by users are then reflected to organization of documents.

The architecture of the current implementation (Designers Amplifier Ver. 1.0) is shown in Figure 5. It consists of a Designers Amplifier server and clients.

The server has mainly two functions, i.e., to maintain ontologies and to organize and maintain documents. In the current implementation, ontologies are frame ontologies which can be easily translated into Ontolingua[6]. Each concept has its name, expression, and slots. Expression specifies how the concept appears in documents. There are super-sub class relations and other relations between concepts. The server maintains common and personal ontologies. The former is provided by system developers and therefore cannot be edited by users. The latter is a set of user ontologies each of which is edited by a user. A user can view and use concepts in ontologies of other users, and can add relations to concepts of others.

The server accepts documents and organizes and modifies them by these ontologies. The system analyzes the given documents to find concepts in the ontologies. If an expression of concepts matches texts in documents, the system adds a hyper-link to the concept. Then, the system calculates co-occurrence of concepts by counting set of concepts which appear in a document together.

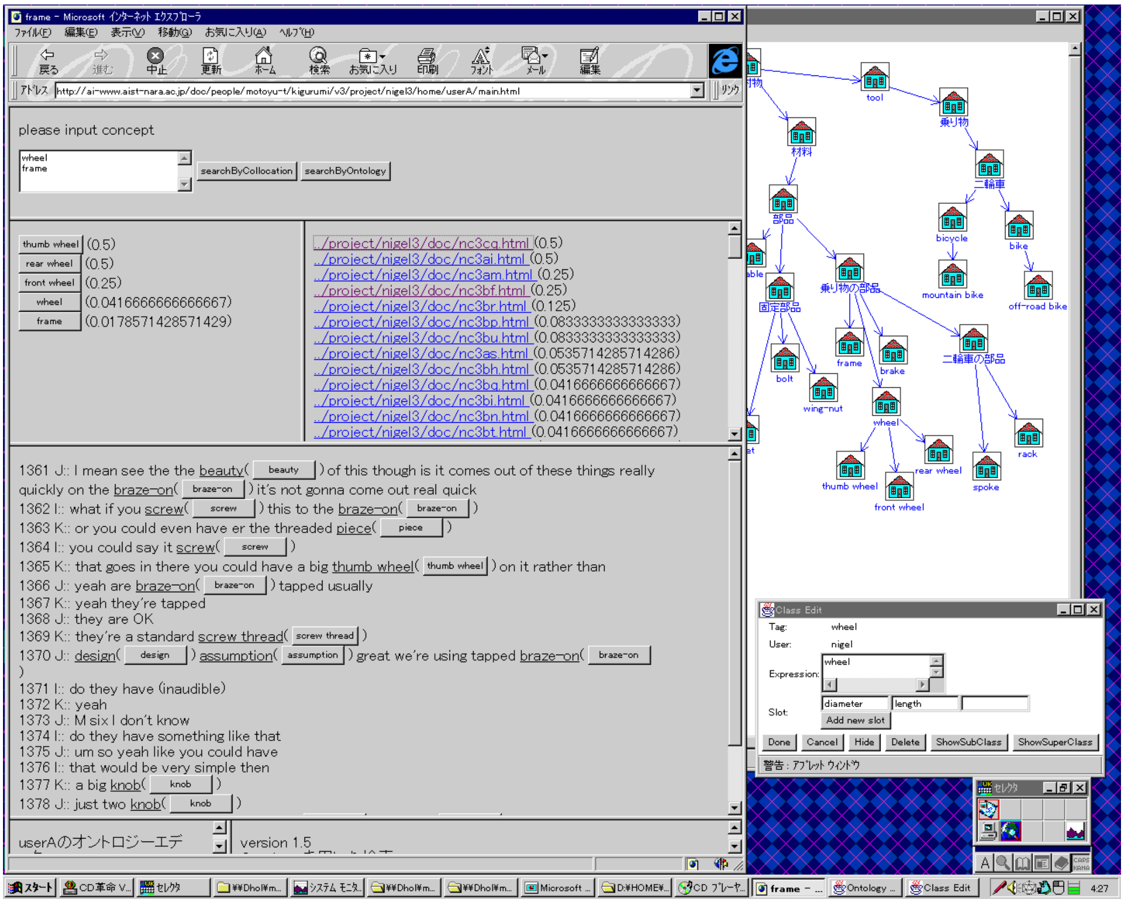

Figure 6: Outlook of Designers Amplifier (Ver. 1.0)

The client system is provided for each user. The outlook of the client is shown in Figure 6. It consists of three components, i.e., document retrieval, document browsing, and ontology browsing. The user can retrieve documents by ontologies or co-occurrence of concepts. The top window of Figure 6 is the input window and the middle-right window is results of retrieval.

The bottom window is a window for document browsing. Buttons in texts are added to original documents by the system. Each button links to an icon in the ontology browsing window. The ontology browsing window (shown bellow the retrieval and document browsing window) shows concepts in ontologies and their relations graphically, and allows users to edit ontologies. The small popped-up window is an editing window for concepts.

3.2. Experiments

We applied this system to two domains. One is protocol data for design which was used in design studies[7]. The other is documents of development of information systems for a factory. We describe how this system works by using the former case.

In the protocol data we used, a rack of bicycle is designed by three designers for two hours. We regard this protocol data as a set of documents by cutting it by twenty lines. We extracted more than 700 concepts which are used in this design, and organized them as ontology.

Then the protocol data is transformed into texts tagged by concepts in the ontology. Designer Amplifier holds this tagged protocol data as organized documents and the ontology as common ontology. The user can retrieve the protocol data either by ontological relations or by co-occurrence relations. For example, Figure 6 shows the user retrieves the protocol data by the words 'wheel' and 'frame' and browses the ontology. In the top window, user put the words 'wheel' and 'frame' and pushed ‘searchByOntology’ button. Then in the middle window the system returns the result. The left window shows the sub class concepts of wheel and frame and their importance values which are computed from the occurrence frequency. The right window shows a document list as the search result. The bottom window shows one document in the list. The some words in the bottom window are click-able. When user clicks them, the concept appears in the ontology window. The large bellow window shows the example of this ontology. It shows the front wheel, thumb wheel, and rear wheel is subclass concept of wheel. In the small bellow window, user is editing the concept wheel.

4. Donden: a distributed ontology development environment

Donden is a ontology development tool which enables users to build multiple ontologies and helps users to integrate them.

Multiple ontologies allows multiple descriptions per concept. In ontology building by Donden, users can make concepts which have links to concepts in other's ontologies and users can see similarity of concepts for integration.

4.1. Multiple ontology of Donden

Figure 7 is the architecture of Donden. Donden consists of ontology browser which is provided for each user and ontology server which is provided for a group of users who want to share ontologies. Ontology browser helps user to edit ontology and to make links to other user's ontology by graphical user interface. Ontology server manages multiple ontologies at the same time by using an ontology description language ASPECTOL[8]. Ontology server helps users to synthesize ontologies by computing similarity between concepts of ontologies. Ontology server and ontology browser are realized as the KQML[9] agents.

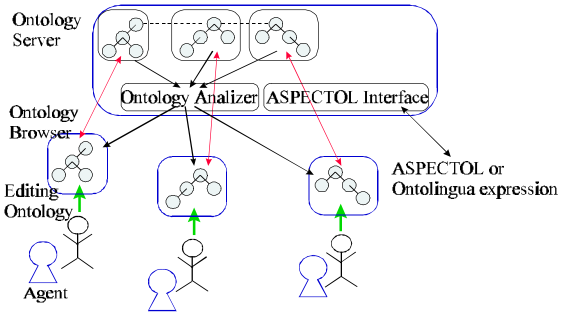

Figure 7: the architecture of Donden

```
(define-class rail:fare (?fare)
:def
(and
  (has-one ?fare rail:adult-fare)
  (value-type ?fare rail:adult-fare Basic:money)
  (has-one ?fare rail:child-fare)
  (value-type ?fare rail:child-fare Basic:money)))
```

Figure 9: the fare class of rail way

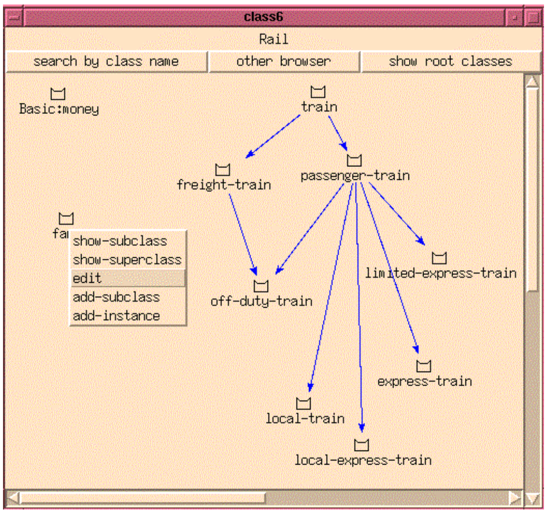

Figure 8: Ontology Browser

```
(define-class taxi:fare (?fare)
 :def
(and
  (has-one ?fare taxi:fare)
  (value-type ?fare taxi:fare Basic:money)))
```

Figure 10: the fare class of taxi

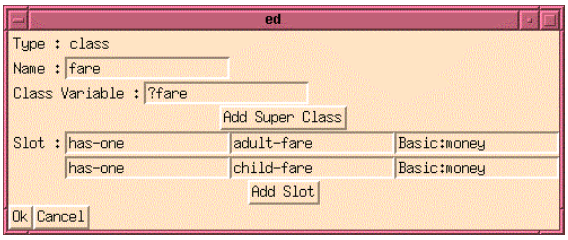

Figure 12: Editing class by Ontology Browser

4.1.1. Frame ontology of Ontolingua

Ontolingua is an ontology description language which was made by KSE (Knowledge Sharing Effort) of DARPA. With Ontolingua, we can write declarative frame expression. We can define classes, relations, functions, and instances by using Ontolingua primitives as define-class, define-relation, define-function, and define-instance.

4.1.2. Aspect

(1)	timepoint:universal-time-spec timepoint:long-time-spec timepoint:calendar-date timepoint:calendar-year
(2)	hotelguide:hotel guesthouse:guesthouse hotel-with-building-information:hotel business-hotel:hotel generic-hotel:hotel
(3)	overnight-with-two-meals:hotel-charge overnight-with-breakfast:hotel-charge overnight-without-meals:hotel-charge lowest-highest-room:hotel-charge
(4)	move-with-traffic:transfer move-with-place:destination

Figure 13: Cluster of a concept for travel

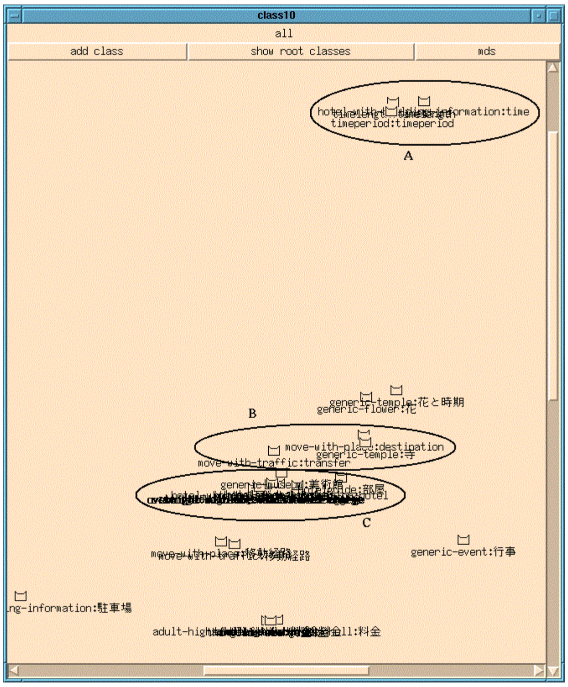

Figure 14:Classes arranged by multi-dimensional scaling

Generally, a concept has different descriptions. To share the concept among systems, we must use a common description for the concept. But different systems usually have different descriptions for a concept because of differences of their purpose and a point of view. It may disturb users to understand each other. Aspect is a framework which can manage such different descriptions of concepts. Ontology is a set of the concept units called aspect. Different aspects for a concept are different expressions for it. And translation rules between aspects enable users to share knowledge.

Figure 9 and Figure 10 are the example expressions of the rail and taxi fare systems. The description after *:def* is a necessary condition for the instance (shown by *?fare*) into the class. In Figure 9, the instance is defined as a thing with one adult-fare slot (the type is *Basic:money*) and one child-fare slot (the type is also *Basic:money*).

Figure 11 shows an example of the translation rule from taxi fare to rail fare. The *taxi:fare* slot of *?taxi-fare* is translated to the *rail:adult-fare* slot of *?rail-fare* and the *rail:child-fare* slot of *?rail-fare*.

4.2. Editing ontology by Donden

Donden provides Ontology Browser for each user and Ontology Server for a group of users who want to share ontologies.

Ontology Server has following functions.

1. Management of users
2. Management of ontology for each ontology builder
3. Support of integration of concepts for ontology builders
4. Translation between expression inside Ontology Server and Ontolingua

Ontology Browser exchanges information of ontologies with Ontology Server and realizes the following functions.

1. To indicate class and class-instance relations by using graph and to edit ontology visually
2. To browse other user's ontologies to use

We implemented the prototype of Ontology Browser and Ontology Server by scheme interpreter, STk.

Figure 8 shows that a rail road user makes ontology by using Ontology Browser. In this figure, there are a class hierarchy about train, a class about train and a class about the money which is brought in from Basic ontology builder. Edit is chosen in the pop-up menu of fare class in this screen.

Figure 8 shows the scene editing the fare actually. Here, we can edit class name, slot and so on.

Ontology Server translates the class of fare into Ontolingua in Figure 9.

4.3. Support of ontology integration

Donden supports ontology integration by collecting similar expression of concepts and showing them to ontology builders. We employ hierarchical cluster analyses as the method of collecting similar classes and multidimensional scaling (TypeIV quantification method) as the method of arranging the classes on a plane.

4.3.1 Method of calculating similarity of concepts

Mainly there are following similarity relations between classes.

1. same concepts of different expressions
2. similar expressions of different concepts (e.g. super - subclass relation and so on)

The examples of Figure 9 and Figure 10 have the same name fare. Since they are not the same concepts, their relation belongs to (2). To count the similarity of classes we use the following properties;

1. name of class
2. names of super class and sub class

3. name of slot and type of slot
4. instance that belongs to the class
5. relations and the functions which use the class
6. document of the class

Donden calculates the similarity of classes by using this information and shows the possibility of concept integration to ontology builder.

We performed an experiment for integration of class expression. In this case the calculation of the similarity is a sum of the name similarity of each slot combination. The other types of information are omitted. The name similarity is

- 1.0 if they are same name.
- 0.5 if one name includes the other.
- 0.0 else case.

4.3.2. Collecting similar concepts by hierarchical cluster analysis

Figure 13 is the result of the experience of hierarchical cluster analysis with the method of calculating similarity. The ontology forigure this experiment is a set of aspects for travel concepts e.g. time, hotel and sightseeing place. It is written in Ontolingua [2,8]. This table shows four highest clusters. The cluster (1) includes all about time, and the cluster (2) all about hotel in the given ontology. Thus we can collect similar concepts from the similarity of slot expression.

4.3.3. Arranging concepts by multidimensional scaling

Figure 14 is the result of the experiment by multidimensional scaling from the same ontology and the same method of calculating similarity with hierarchical cluster analysis.

We can also find some groups from it. The groups are classified three types as follows:

1. The group of same or similar concepts. For example, the four classes enclosed by oval A on Figure 14 are all about time.
2. The group including different concepts which have same property concept. For example, the three classes enclosed by oval B on Figure 14 are "transfer", "destination" and "temple" which have the transportation slot.
3. The others. Sometimes, we can find some groups in the group by zooming in. For example, the twelve classes enclosed by oval C on Figure 14 further consists three groups (five classes about hotel, six classes about hotel-charge and one class about museum).

5. Summery

We discussed two systems for knowledge sharing and reuse. One is the document browsing and organizing system. The other is the ontology browsing and integration supporting system. Both of systems aim to share personal knowledge among users. We show the Designers Amplifier can organize the poorly-structured engineering information since mixture of media and formalized representation can facilitate deep-understanding among users. We show Donden can manage multiple ontologies among users and help to integrate them by calculating similarity of concepts.

We are going to integrate the two system and apply to real and larger engineering domains to know how it can contribute knowledge organization.

References

[1] Flew, A.(editor) A Dictionary of Philosophy, Pan Books, 1979.

[2] Gruber, T.R. Toward principles for the design of ontologies used for knowledge sharing. Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford University, August 1993.

[3] Cutkosky, M.R. Engelmore, R.S. Fikes, R.E. Genesereth, M.R. Gruber, T.R. Mark, W.S. Tenenbaum, J.M. and Weber, J.C. PACT: An experiment in integrating concurrent engineering systems. IEEE Computer, 1993; January: 28—38.

[4] Farquhar, A. Fikes, R.E. Pratt, W. and Rice, J. Collarborative ontology construction for information integration. Technical Report KSL 95-63, Knowledge Systems Laboratory, Stanford University, 1995.

[5] Fraser, J. and Tate, A. The enterprise tool set—an open enterprise architecture. In Proc. IJCAI'96 Workshop on Intelligent Manufacturi ng Systems, 1995.

[6] Gruber, T.R. Ontolingua: A mechanism to support portable ontologies. Technical Report KSL 91-66, Stanford University, Knowledge Systems Laboratory, 1992.

[7] Takeda, H. Yoshioka, M. Tomiyama, T. and Simomura, Y. Analysis of design processes by functional evaluation process. Analysing Design Activity (Nigel Cross,Henri Christaans,and Kees Dorst,editors), Chichester, John Wiley & Sons, 1995, 187—209.

[8] Takeda, H. Iino, K. and Nishida, T. Agent organization and communication with multiple ontologies. the International Journal of Cooperative Information Systems, 1995; 4(4): 321—337.

[9] Finin, T. Weber, J. Wiederhold, G. Genesereth, M. Fritzson, R. McKay, D. McGuire, J. Pelavin, P. Shapiro, S. and Beck, C. Specification of the KQML agent-communication language. Technical Report EIT TR 92-04, Enterprise Integration Technologies, 1992 (Updated July 1993).