Human Aspects in Computer Integrated Manufacturing
G.J. Olling and F. Kimura (Editors)
Elsevier Science Publishers B.V. (North-Holland)

Systematizing Design Knowledge for Intelligent CAD Systems

Tetsuo Tomiyama, Deyi Xue, Yasushi Umeda, Hideaki Takeda,
Takashi Kiriyama, and Hiroyuki Yoshikawa

Department of Precision Machinery Engineering, Faculty of Engineering,
The University of Tokyo, Hongo 7-3-1, Bunkyo-ku, Tokyo 113, Japan

Abstract

To build design knowledge bases for intelligent CAD systems, systematization of design knowledge itself is needed. Five processes of systematization were identified, i.e., setting up a view, articulation, codification, crystallization, and reusing and sharing. This paper illustrates our research effort towards design knowledge systematization, including building physical feature database, function modeling, design process formalization, and design knowledge analysis.

Keyword Codes: J.6; I.2.4; I.2.1
Keywords: Computer-Aided Engineering; Knowledge Representation, Formalisms and Methods; Artificial Intelligence, Applications and Expert Systems

1. INTRODUCTION

Developing *intelligent computer aided design systems* (ICAD) is crucial within a future advanced computer integrated manufacturing environment [1-7]. There is an emphasis on the use of advanced knowledge processing techniques to incorporate more domain knowledge and intelligence that are missing from conventional CAD. Despite these efforts, there seems to exist no such truly intelligent CAD developed yet, perhaps because design knowledge is too huge and complex to be organized and dealt with by existing knowledge representation techniques [8].

A considerable body of current AI research centers on the approach of very large-scale knowledge bases (VLKB). An example is the Cyc Project conducted at MCC [9]. The KIF Project advocates the idea of *knowledge standard* for exchanging knowledge [10]. These projects aim at building powerful AI systems, first by collecting a large number of knowledge chunks from ontological, fundamental common knowledge to domain dependent specific knowledge, and second by providing mechanisms for reusing and sharing knowledge [11].

ICAD as an intelligent system needs to be equipped with VLKB containing design knowledge that is considered huge and complex. At the University of Tokyo, we have been working on research to develop ICAD [12] and as its part we have been conducting various projects to collect design knowledge. The most significant result of these knowledge collection projects is that, if knowledge is *systematized*, the task boils down to a matter of actual collection. If not, however, it is extremely difficult even to collect knowledge.

Conventional knowledge acquisition techniques mention only about acquiring techniques

and do not discuss systematizing techniques. Consequently, expert systems constructed with conventional techniques are not provided with systematically organized knowledge. This suggests that their knowledge bases are *scruffy*, i.e., unorganized or unstructured, thus difficult to maintain, and that generality of knowledge is lost. This certainly causes hard-fails of the system and prevents it from reusing and sharing.

This paper discusses the concept of *systematized* design knowledge and illustrates its examples in the context of developing ICAD. To begin with, Chapter 2 discusses a general systematization process. Chapter 3 illustrates our effort to obtain systematized design knowledge toward ICAD and demonstrates the usefulness of the systematization process described in Chapter 2. Chapter 4 concludes the paper.

2. WHAT IS SYSTEMATIZED KNOWLEDGE?

We consider that systematization of knowledge has the following processes, *viz.*, setting up a view, articulation, codification, crystallization, and reusing and sharing of knowledge. Science is an effort to achieve such systematized knowledge.

2.1. Classification of Knowledge

Generally speaking, knowledge has two forms, i.e., *recognized knowledge* and *unrecognized knowledge*. Another dimension of knowledge is *tacit knowledge* and *codified knowledge*. Tacit knowledge is a form of knowledge that is explicitly or implicitly recognized by human and used for reasoning but very difficult to describe. Codified knowledge is a form of knowledge that is described with symbols, figures, and so on.

Figure 1 depicts our viewpoint about these two dimensions. (Unrecognized and codified knowledge is meaningless.) Textbook knowledge and information stored in a database are examples of recognized and codified knowledge. Scientific knowledge largely falls into this category. So-called commonsense is a typical example of recognized but tacit knowledge. Expertise and skill are mostly composed of unrecognized knowledge.

The primary goal of systematization of knowledge is to convert recognized and tacit knowledge to recognized and codified knowledge. However, there can be various degrees of codification. For instance, information stored in a database is structured and computable. Textbooks can be machine-readable and has some structure, but might not be computable. Since our ultimate goal is the development of ICAD, computable knowledge is, perhaps, the best form of systematized knowledge. In this paper, as a working definition of systematization we discuss conversions from recognized and tacit knowledge to (recognized and) codified, and preferably, further to computable knowledge.

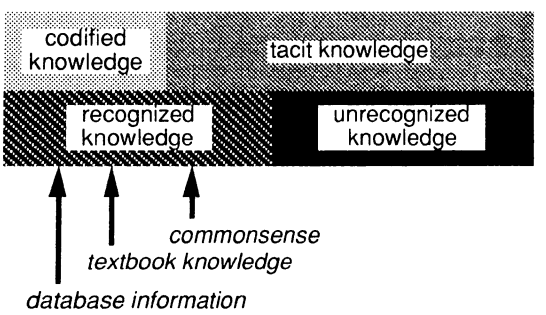

Figure 1. Classification of knowledge.

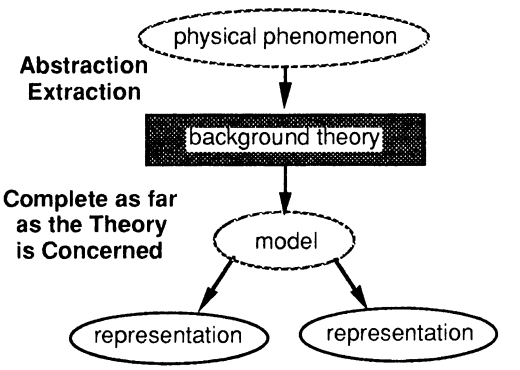

Figure 2. Modeling.

2.2. Setting up a View

To convert recognized and tacit knowledge to codified, first we need to set up a view. This can be best explained by how modeling works [13]. In science and engineering, a model is created by observing a phenomenon based on a background theory (Figure 2). A kinematic model is generated based on kinematics, using only the vocabulary and concepts of kinematics. In this regard, modeling is a kind of filtering through observation, such that a generated model becomes complete as far as the theory is concerned.

There can be many different background theories applicable to a single phenomenon, each of which resulting in a different model. For instance, a geometric model is created with only the vocabulary and concepts of algebraic geometry, such as points, lines, vectors, and so on, while a distortion model is generated from the concepts of strength of materials that has only a single law (i.e., Hooke's Law).

A background theory dictates general, conceptual relationships among relevant parameters and entities. This suggests that a background theory can have different interpretations for an identical phenomenon and that even a single background theory can generate different models depending on interpretations.

Therefore, when converting tacit knowledge into codified knowledge, we need to have a predefined set of concepts. This set is called a *view* and often associated with a well-defined theory that describes relationships among those concepts and has well-defined interpretation.

2.3. Articulation

Once we set a view together with a background theory and its interpretation, we can identify instances of concepts that belong to the view in the observed phenomenon. This process is *articulation* in which unorganized instances of concepts are given *representations* chosen from the vocabulary. Articulation is thus a process for recognizing the world and naming; knowledge after articulation is recognized knowledge.

2.4. Codification

After articulation, we have a set of instances that were given representations. Since the process of setting up a view introduces a background theory that defines relationships among these instances, this set of instances has *structure.* In artificial intelligence, structure is often given by entities and relationships among entities, and attributes of entities and relationships. The background theory defines the whole system of these entities, relationships, and attributes and gives *expressions* to the structure.

We now have codified knowledge that is a collection of facts observed in the phenomenon. A *fact* is is a (partial) account of the phenomenon within the theory. Rules obtained in a knowledge acquisition process are examples of such codified knowledge.

2.5. Crystallization

Codification process generates only pieces of factual knowledge. This type of knowledge is extremely useful when massively stored in a database system, though such knowledge may result in *hard-fail* [14]. A knowledge based system can hard-fail, when it encounters a situation that is not described in the knowledge base. An approach to removing this limitation is either to broaden the range that the knowledge base covers or to generalize the knowledge. Projects to construct VLKB are examples of the former approach. The latter is achieved by two ways: one is to use deeper knowledge [14] and the other is to use a general, abstract knowledge.

Crystallization is a process to generate general and abstract knowledge from purely factual knowledge in an organized manner. Such general and abstract knowledge is called a *theory.* This theory is different from the background theory initially used for setting up the view, articulation, and codification in that it contains more domain-specific, object level descriptions. Having such a theory, however, we can give better organization to the factual knowledge obtained from the codification process, further improve the initial background theory, or even make knowledge collection much easier. Figure 3 depicts the setting-up-a-view, articulation, codification, and crystallization processes.

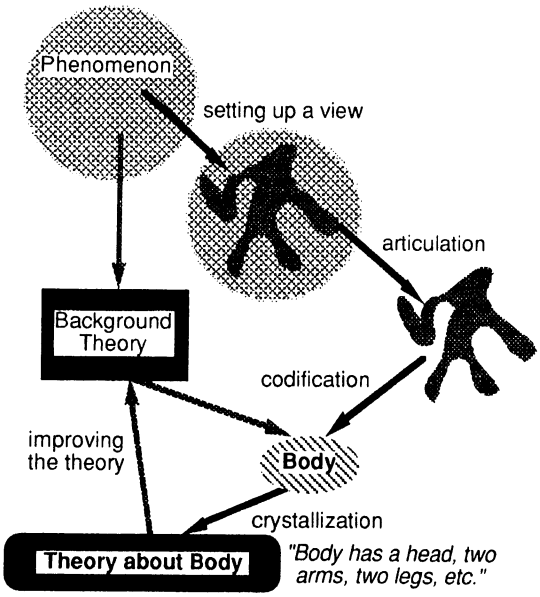

Figure 3. Setting up a view, articulation, codification, and crystallization.

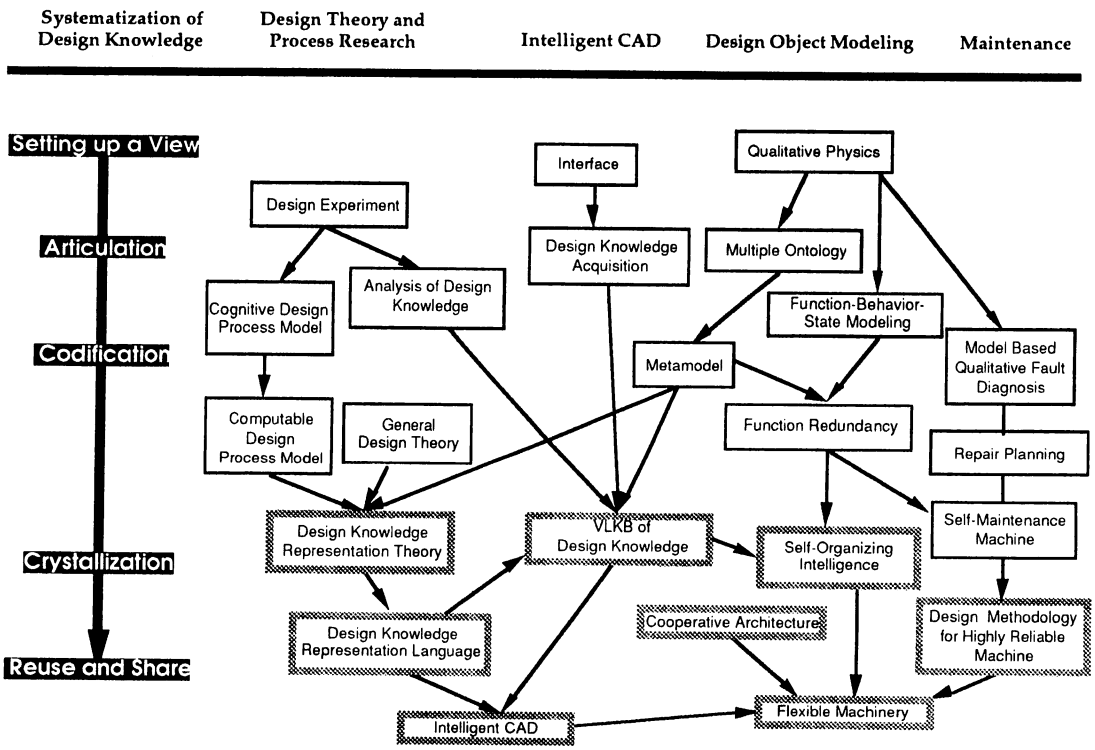

Figure 4. Systematizing design knowledge: A research agenda.

An important lesson that we learnt from a project to build VLKB is that crystallization is indispensable for organizing design knowledge that is huge and complex and for improving the background theory. The background theory is often too general and tells almost nothing about the object level. This can be improved by having an object level theory obtained from crystallization. In other words, a brute force approach to collect knowledge eventually fails, because collected knowledge chunks have almost no relationships among them and do not yield any meaningful, interesting results besides initially expected. Even if the task is just to build a factual database, knowledge collection must be done in a systematic and organized way. To do so, a theory obtained as a result of crystallization is helpful.

2.6. Reusing and Sharing of Knowledge

To reuse and share knowledge, we need not only knowledge representation techniques but also a common knowledge description format (knowledge standard) such as KIF. However, KIF is almost all concerned about interchanging knowledge among different knowledge based systems represented in logic. In this context, it does not deal with more semantical part of knowledge, called ontology. On the other hand, the Cyc Projects pays more attention to standardization at terminological, taxonomical, and ontological levels.

Although here we do not elaborate this issue any further, we just point out that systematization of knowledge cannot be accomplished without having such knowledge standard and that any standard can serve as a useful basis in every process of systematization.

3. EXAMPLES OF SYSTEMATIZATION OF DESIGN KNOWLEDGE

3.1. Design Knowledge

In this chapter, we illustrate examples of systematized design knowledge. Figure 4 illustrates the research agenda of our group towards systematization of design knowledge.

Design knowledge has two types, i.e., design process knowledge and design object knowledge. There is a difference between them that design process knowledge is *process knowledge* which describes how, whereas design object knowledge is largely *fact knowledge* which describes what. In design, we need to build a goal from imprecise, incomplete, and sometimes not consistent specifications. A design process begins with ambiguous or rough descriptions of the design object and they will be gradually detailed. This indicates that a design process is a typical ill-defined and ill-structured problem that needs a cognitive approach to observe and describe it. Perhaps primarily due to this fact, process knowledge has been long ignored in CAD research and no serious scientific methods were applied.

3.2. Design Object Modeling

The triumph of geometric modeling also contributed to the ignorance of studying design process knowledge. However, there still remains issues to be studied even for design object modeling. First, the evolutionary nature of design processes influences much on the representation and design objects cannot be given a fixed or rigid representation scheme. Second, in engineering design we need to deal with the physical world anyway. Knowledge about the physical world must be incorporated into the system symbolically. Qualitative physics [15] is an approach to handling this type of knowledge. The symbolic (i.e., qualitative) nature of reasoning in qualitative physics is useful to reason about behaviors from rough descriptions of the design object that will be gradually detailed.

Qualitative physics has two roles, *viz.*, to reason about dynamic behaviors of physical systems and to symbolically model their structure and behaviors. Given a set of environmental conditions, one can set up an appropriate design object model and reason about what might happen, that is, envision physical phenomena.

Knowledge of qualitative physics can also be used to manage multiple models in ICAD, which is an integration issue. In mechanical design, a design object can be modeled from different aspects, such as FEM analysis, kinematic analysis, and dynamic analysis. Differences

among these models originate from the differences in their background theories (see Figure 2). However, obviously the models are not independent from each other and models have relationships accordingly. In a multiple modeling environment, these relationships must be maintained. Suppose, for example, a kinematics model of a mechanism and its distortion model. Changing motion in the kinematic model affects forces in the distortion model. This can be inferred in such a way that, since acceleration of a solid object depends on force applied to it, changes in acceleration in the kinematic model should lead to changes in force in the distortion model and, as a result, bend must be recalculated. To find such relationships among models, we use Qualitative Process Theory [17] and developed a new modeling framework called *metamodel* [13, 18-20].

The metamodel mechanism has symbolic representation of concepts (i.e., physical parameters and phenomena) used to represent the design object. It also has a knowledge base about relationships among these physical concepts (see Figure 5). The designer describes the design object as a combination of *physical features* that represent relationships between unit physical phenomena and entities. A qualitative reasoner tries to figure out, first, if the given combination of physical features performs the desired behaviors given as the specifications. Second, it generates necessary models for evaluation. The metamodel mechanism then prepares necessary data for each of these modelers that might be, e.g., an FEM analysis system. When the model is modified due to results of analysis, the metamodel mechanism propagates the change to other models accordingly through the network about concepts (see Figure 6).

Developing the metamodel mechanism and its qualitative reasoning system corresponds to the setting-up-a-view, articulation, and codification processes toward systematized design knowledge.

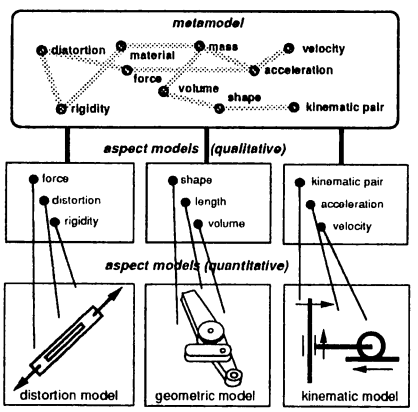

Figure 5. Metamodel mechanism.

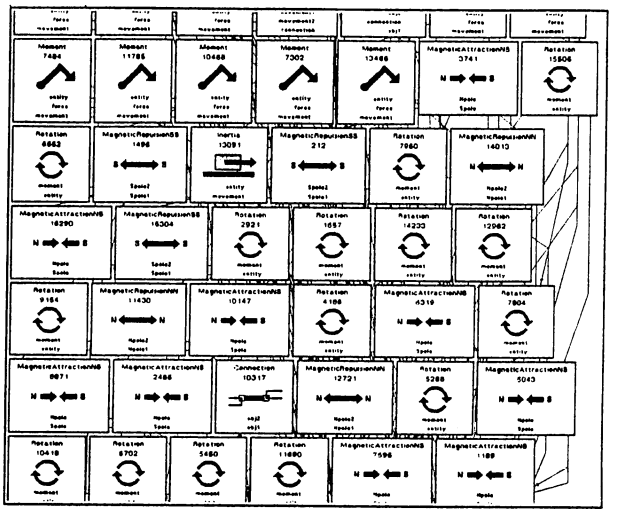

Figure 6. A conceptual network of metamodel.

3.3. Physical Feature Database Project

We started a project to build a large-scale physical feature database about engineering design knowledge. This project of collecting physical features was aiming at intensifying codified design knowledge. We chose three engineering domains as the source of knowledge, namely kinematics, robotics, and classical physics, because (i) knowledge about these domains is fundamental for mechanical engineers, (ii) domain theories as background theories are already well established, and therefore (iii) we thought we could systematically obtain descriptions of do-

main knowledge from textbooks.

Figure 7 shows a screen copy of the interface of the knowledge base. The physical feature shown in the figure is a planetary gear mechanism. We have collected some thousand of physical features. The physical features stored in the database are used for qualitative reasoning about behaviors, suggesting alternative behaviors, and correlating aspect models in the metamodel mechanism. In order to use the physical features for reasoning, they must be described on the basis of the common vocabulary such as time, space, shape, and material, i.e., common ontology.

However, our initial estimation was too naive. For instance, a collected feature could not be used in other situations than originally described one, not because of discrepancy in the viewpoints of the knowledge collectors but because of the lack of the more unified, ontological basis than qualitative physics. In this respect, collecting naive knowledge about physical phenomena and entities [21] is indispensable.

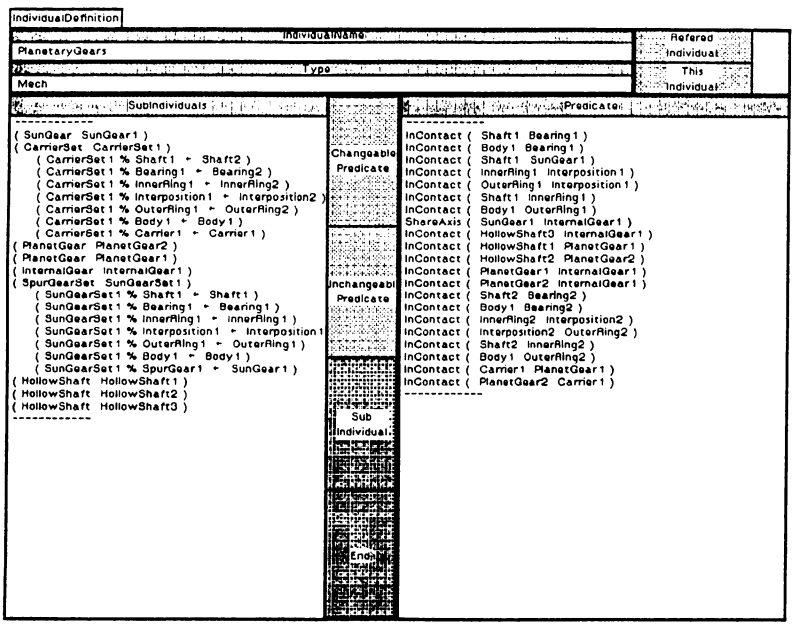

Figure 7. Physical feature database.

3.4. Function Modeling

For modeling machines, our research includes development of representational scheme of *functions* of a machine. Figure 8 shows our representational scheme named FBS (Function-Behavior-State) diagram [22]. In an FBS diagram, a design object is represented with a functional hierarchy constructed subjectively based on the designer's intention and each subjective function is represented with respect to objective behavior constrained by physical laws.

Figure 9 is a screen hardcopy of the FBS modeler that implements the FBS diagram. The upper half of the network represents a functional hierarchy and the lower half represents structure and physical phenomena that embody the intended function. Using qualitative reasoning, the system examines if the intended function can be achieved by the structure. In Figure 9, functions represented with white nodes are justified by behaviors, whereas functions represented with black nodes are not justified by appropriate behaviors. Hatched nodes represent physical phenomena the designer did not anticipate.

Function modeling is an attempt to set up a view and to articulate yet-to-be-clarified functional knowledge. Codifying knowledge about functions within the framework of the FBS diagram helps to build intelligent CAD systems that can be used in the conceptual design stage.

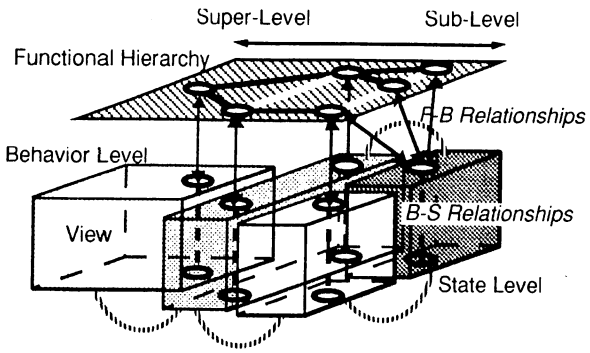

Figure 8. FBS diagram.

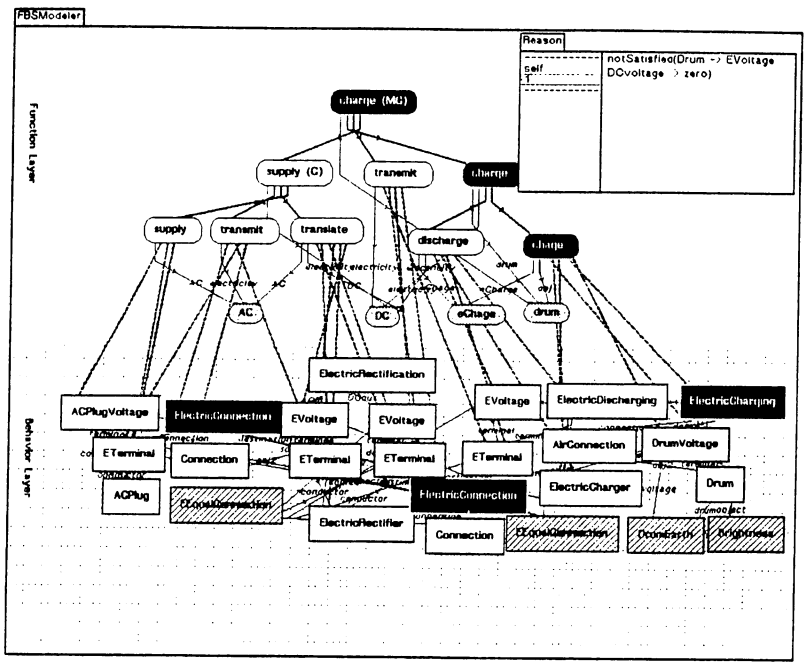

Figure 9. FBS modeler.

3.5. Design Process Formalization

Since design studies are still in prescientific stage [23] and among other things design processes are least studied, we had to take an empirical approach [24, 25]. We conducted *design experiments* to collect experimental data about design. A design experiment is a psychological experiment in which the entire design session is recorded on video to obtain protocol data from analysis of the records. We derived a cognitive model of design process from the results of design experiments (see Figure 10).

We could observe that a design process was decomposed into small design cycles. A design cycle has five processes, *viz.*, awareness of the problem, suggestion of solution candidates, development of each of candidates, evaluation of the result of development, and conclusion. Protocol data obtained from design experiments could be translated into logical formulae. Figure 11 shows logical formalization of design knowledge extracted from protocols in designing a weighing machine.

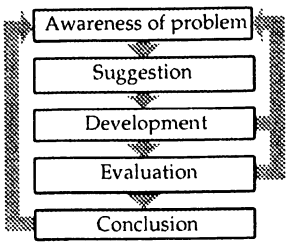

Figure 10. Cognitive design process model.

scale(*s)← can_measure(*s *w) ∧ support(*x *w)
support(*s *w)← has(*s spring)
can_measure(*s *w)← translate(*s *w *d) ∧ has(*s indicator1)
translate(*s *w *d)← is_in_proportion(*s *w *d)
is_in_proportion(*s *w *d)← has(*s rack) ∧ has(*s pinion)
translate(*s 100kg 5mm)← has(*s indicator1) ∧ has(indicator1 many_gears)
not(scale(*s))← has(*s indicator1) ∧ not(is_easy_mechanism(indicator1))
not(is_easy_mechanism(indicator1))←has(indicator1 many_gears)

Figure 11. Logical representation of design protocols.

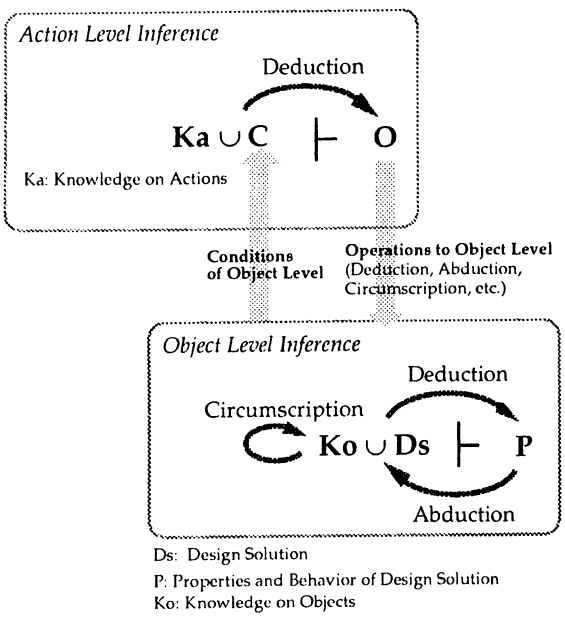

Figure 12. Computable design process model.

Figure 12 illustrates a computable design process model that formalizes design processes as combinations of three types of reasoning, i.e., deduction, abduction, and circumscription [25]. Deduction is a process to derive facts from known facts. Abduction is a process to find possible ways to obtain the desired result. Circumscription is a process to reform knowledge to resolve contradiction within the knowledge base. Object level inference deals with facts about the design object, whereas the action level inference guides the object level inferences. Based on this design process model, we implemented Design Simulator (Figure 13) that traces back design processes using design knowledge extracted from protocol analyses.

The cognitive and computable design process models are examples of codified and crystallized design knowledge. Having such models, collecting and systematizing design knowledge becomes easier, because they serve as a template for describing design process knowledge.

3.6. Analysis of Design Process Knowledge

We analyzed protocol data obtained from design experiments and found six categories of design knowledge; i.e., knowledge about entities, functions, attributes, topologies, relation-

ships, and manufacturing methods. Between these categories, there are eight primitive transitions, which comprise design processes. The transitions are;

- from functions to entities
- from entities to functions
- from attributes to entities
- from entities to attributes
- from attributes to attributes
- from topologies to relationships
- from entities to manufacturing methods, and
- from manufacturing methods to attributes.

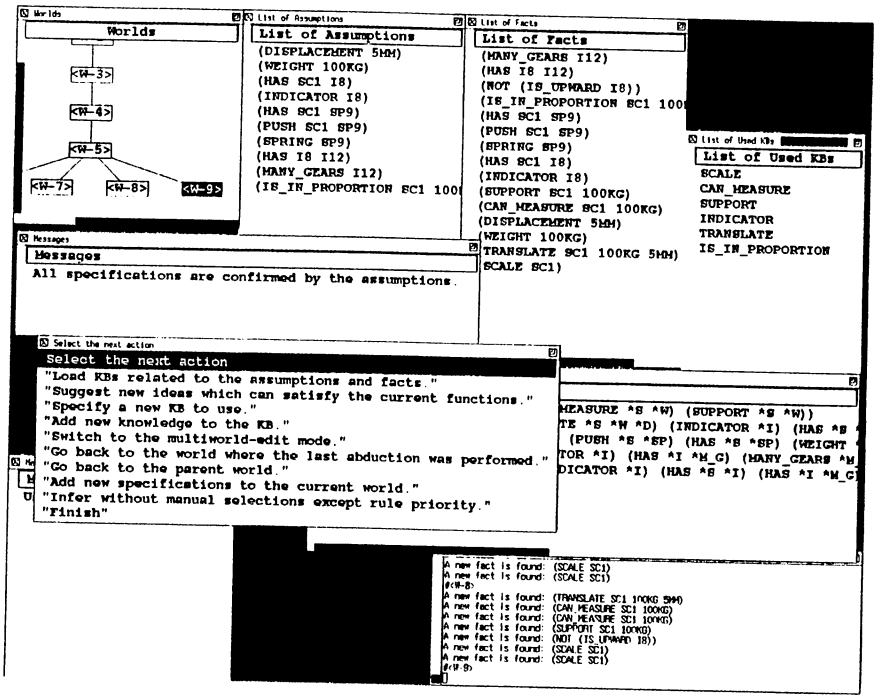

Figure 13. Design Simulator.

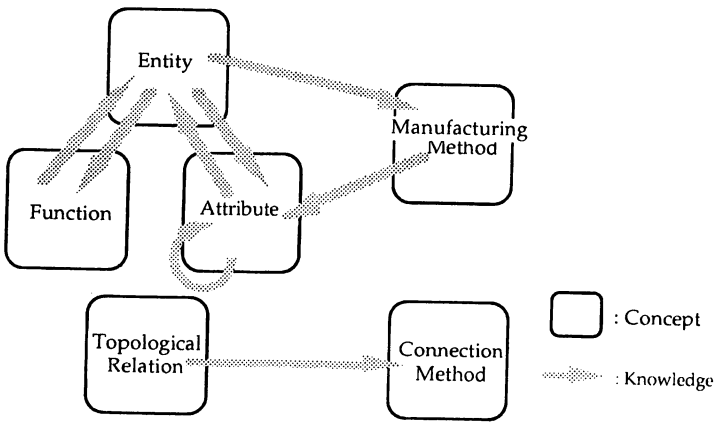

Figure 14. Design process knowledge as relationships among concepts at the object level.

Figure 14 depicts the six categories of design knowledge and the eight transitions among them. This figure is a typical example of crystallized design knowledge, because these transitions can be used as a template of design process knowledge when collecting design process knowledge at the design object level. Obviously, such categorization makes easier to collect design knowledge.

At this moment, however, we have not yet succeeded in building an integrated design process knowledge theory clarifying the use of design process knowledge shown in Figure 14 and the computable design process model. This is definitely a future research issue.

4. CONCLUSIONS

Building an intelligent system such as intelligent CAD is not achieved only by collecting massive amount of knowledge; we need systematization of design knowledge even before we start collecting knowledge. In this paper, we discussed systematization methods of design knowledge that include such processes as setting up a view, articulation, codification, crystallization, and reusing and sharing. We also illustrated our research effort corresponding to these processes of systematization, such as developing the metamodel mechanism, building a physical feature database, function modeling, design process modeling, and analysis of design process knowledge.

As a result of crystallization, we have theories that can give better organization and structure to collected knowledge and can make the knowledge collection task much easier. Its good examples are the computable design process model and the framework of design process knowledge at the object level shown in Figure 14. These can further simplify and make easier the knowledge collection tasks; in this sense, systematization of design knowledge is a recursive process in which better knowledge representation and acquisition techniques are pursued.

Currently, we are developing IIICAD (Intelligent, Integrated, Interactive CAD) [12]. Design knowledge collected through the research on systematization is represented and integrated in its prototype.

5. REFERENCES

1. P. J. W. ten Hagen and T. Tomiyama (eds.): *Intelligent CAD Systems I: Theoretical and Methodological Aspects*, Springer-Verlag, Berlin, (1987).
2. V. Akman, P. J. W. ten Hagen, and P. J. Veerkamp (eds.): *Intelligent CAD Systems II: Implementational Issues*, Springer-Verlag, Berlin, (1989).
3. P. J. W. ten Hagen, and P. J. Veerkamp (eds.): *Intelligent CAD Systems III: Practical Experience and Evaluation*, Springer-Verlag, Berlin, (1990).
4. H. Yoshikawa, D.C. Gossard (eds.): *Intelligent CAD, I*, North-Holland, Amsterdam, (1989).
5. H. Yoshikawa, T. Holden (eds.): *Intelligent CAD, II*, North-Holland, Amsterdam, (1990).
6. H. Yoshikawa and F. Arbab (eds.), T. Tomiyama (Managing Editor): *Intelligent CAD, III*, North-Holland, Amsterdam, (1991).
7. T. Tomiyama: "Intelligent CAD Systems," in G. Garcia and I. Herman (eds.), *Advances in Computer Graphics VI, Images: Synthesis, Analysis, and Interaction*, Springer-Verlag, Berlin, (1991), pp. 343-388.
8. B. Veth: An Integrated Data Description Language for Coding Design Knowledge, in P. J. W. ten Hagen, T. Tomiyama (eds.), *Intelligent CAD Systems I: Theoretical and Methodological Aspects*, Springer-Verlag, Berlin, (1987), pp.295-313.
9. D. B. Lenat and R. V. Guha: *Building Large Knowledge-Based Systems*, Addison-Wesley, Reading, MA, (1989).
10. M. R. Genesereth and R. Fikes: *Knowledge Interchange Format Version 2.2 Reference Manual*, Technical Report Logic-90-4, Computer Science Department, Stanford University,

(1990).
11. R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W.R. Swartout: "Enabling Technology for Knowledge Sharing," *AI Magazine*, Vol. 12, No. 3, (1991), pp. 36-56.
12. D. Xue, H. Takeda, T. Kiriyama, T. Tomiyama, and H. Yoshikawa: "An Intelligent Integrated Interactive CAD – A Preliminary Report," in D. C. Brown, M. B. Waldron, and H. Yoshikawa (eds.), *Preprints of the IFIP WG 5.2 Working Conference on Intelligent CAD*, The Ohio State University, Columbus, OH, (1991), pp. 173-200.
13. T. Tomiyama, T. Kiriyama, H. Takeda, D. Xue, and H. Yoshikawa: "Metamodel: A Key to Intelligent CAD Systems," *Research in Engineering Design*, Vol. 1, No. 1, (1989), pp.19-34.
14. C. Price and M. Lee: "Applications of Deep Knowledge," *Artificial Intelligence in Engineering*, Vol. 3, No. 1, (1988), pp. 12-17.
15. D.G. Bobrow (ed.): *Qualitative Reasoning about Physical Systems*, North-Holland, Amsterdam, (1985).
16. Y. Umeda, T. Tomiyama, and H. Yoshikawa: "A Design Methodology for a Self-Maintenance Machine," in L.A. Stauffer (ed.), *Design Theory and Methodology – DTM '91 –*, DE-Vol. 31, ASME, NY, (1991), pp. 143-150.
17. K. Forbus: "Qualitative Process Theory," *Artificial Intelligence*, Vol. 24, No. 3, (1984), pp. 85-168.
18. T. Kiriyama, F. Yamamoto, T. Tomiyama, and H. Yoshikawa: "Metamodel: An Integrated Modeling Framework for Intelligent CAD," in J.S. Gero (ed.), *Artificial Intelligence in Design*, Computational Mechanics Publications, Southampton, Boston, (1989), pp.429-449.
19. T. Kiriyama, T. Tomiyama, and H. Yoshikawa: "A Model Integration Framework for Cooperative Design," in D. Sriram, R. Logcher, and S. Fukuda (eds.), *Computer-Aided Cooperative Product Development*, Lecture Notes in Computer Science 492, Springer-Verlag, Berlin, (1991), pp. 126-139.
20. T. Kiriyama, T. Tomiyama, and H. Yoshikawa: "The Use of Qualitative Physics for Integrated Design Object Modeling," in L.A. Stauffer (ed.), *Design Theory and Methodology – DTM '91 –*, DE-Vol. 31, ASME, New York, USA, (1991), pp. 53-60.
21. P. Hayes: "Naive Physics Manifesto I: Ontology for Liquids," in J. Hobbs and R. C. Moore (eds.), *Formal Theories of the Commonsense World*, Ablex, Norwood, NJ, (1985), pp. 71-107.
22. Y. Umeda, T. Tomiyama, and H. Yoshikawa: "Function, Behaviour, and Structure," in J.S. Gero (ed.), *Applications of Artificial Intelligence in Engineering V, Vol. 1: Design*, Computational Mechanics Publications, Southampton, Boston, Springer-Verlag, Berlin, (1990), pp. 177-193.
23. J. R. Dixon: "On Research Methodology Toward a Scientific Theory of Engineering Design," *AIEDAM (Artificial Intelligence for Engineering Design, Analysis and Manufacturing)*, Vol. 1, No. 3, (1987), pp. 145-157.
24. H. Takeda, T. Tomiyama, and H. Yoshikawa: "Logical Formalization of Design Processes for Intelligent CAD Systems," in H. Yoshikawa and T. Holden (eds.), *Intelligent CAD, II*, North-Holland, Amsterdam, (1990), pp. 325-336.
25. H. Takeda, P. Veerkamp, T. Tomiyama, and H. Yoshikawa: "Modeling Design Processes," *AI Magazine*, Vol. 11, No. 4, (Winter 1990), pp. 37-48.