18

Design Knowledge Organization and Acquisition Through Visual Representation

Yasuyuki Maeda, Yoshiyuki Koseki, Yuichi Koike and Midori Tanaka
C&C Research Laboratories, NEC Corporation
4-1-1 Miyazaki, Miyamae-ku, Kawasaki, Kanagawa 216, JAPAN.
e-mail: {maeda,koseki,koike,midori}@mmp.cl.nec.co.jp

Abstract

To realize knowledge intensive CAD systems, it is necessary to provide a facility with which various types of design knowledge in different forms are organized so as to work together. Visualization is one of the most effective methods to achieve this capability. This paper proposes a framework of intelligent CAD based on design knowledge visualization. The proposed framework is composed of three steps: (1) visualizing knowledge utilization, (2) visualizing knowledge organization, and (3) acquiring knowledge from visual information. To integrate the above steps, we propose putting focus on navigational knowledge and develop the framework. To show the validity of the proposed framework, three visual tools are investigated and employed to manage navigational design knowledge. Applications of a visual knowledge organization tool (DT) and a visual programming tool (VIP) show that knowledge visualization greatly improves the productivity and maintenability of navigational knowledge. Application of a spatial parsing tool shows the possibility of automatic knowledge extraction from existing design data, which may lead to the elimination of knowledge acquisition bottleneck in future.

Keywords

knowledge organization, knowledge acquisition, visual programming, visual design knowledge, visual language, spatial parsing

1 INTRODUCTION

The role of CAD systems in product design has evolved dramatically over the last decade, moving from drawing tools to integrated engineering environments. Recent advances in expert system technology have made it possible to provide CAD systems with intelligent facilities which save designers' labor and to some extent supplement their skill and expertise. However, the benefits of intelligent CAD technology are still limited to small domains.

In order to achieve the expansion of intelligent CAD facilities and further improvement

T. Tomiyama et al. (eds.), *Knowledge Intensive CAD*

in productivity over the entire manufacturing environment, important design information should be shared among the different tools in the design and manufacturing processes.

As for design object, product model has been aimed to enable the exchange of product data and information among different tools. As for design knowledge, however, existing intelligent CAD systems convey only a limited amount of domain knowledge in restricted knowledge representation forms. Such design knowledge is useful only where the design process is simple and established, since existing knowledge representation models can cover a very limited range of design knowledge compared with the wide variety of knowledge designers are required to use.

In this paper, we investigate a methodology for organizing piecemeal knowledge and constructing an integrated knowledge base for intelligent CAD without forcing excessive labor on either its builders or its users. We tackle this subject by utilizing "knowledge visualization" techniques, which overcome the deficit of existing expert systems.

In section 2, we analyze types of design knowledge and specify which types of knowledge would be particularly effective when incorporated into CAD systems. In section 3, a general methodology to organize and utilize design knowledge through visual knowledge representation is proposed. Section 4 illustrates some visual tools which are used to build intelligent CAD systems in accordance with the proposed methodology, and discusses the effectiveness of visual representation for design knowledge.

2 DESIGN KNOWLEDGE FOR ICAD

CAD systems usually incorporate facilities to help designers such as:

- creating and editing design objects using 2D and 3D representation
- analysis and simulation

From this point of view, these systems can be considered as a combination of DB systems which store various types of data regarding products, and collections of computer programs which use the data in the DBs.

Over and above these ordinary CAD facilities, intelligent CAD systems help the user by providing intelligent facilities, such as:

- checking constraints and solving them if needed
- evaluating conditions of transforming rules and applying them to design data
- locating related information appropriately and showing it in comprehensible forms

To realize such intelligent facilities, however, a great quantity of design knowledge is required, from general knowledge to the specific knowledge required for a particular company, product, team or individual. Such knowledge usually amounts to a large quantity of data, making it difficult to achieve a suitable balance between the benefits and the input cost of knowledge bases. This problem, which is commonly shared among many application fields related to building intelligent systems, is known as "*knowledge acquisition bottleneck*".

One of the most promising approaches to solving this problem is knowledge visualization. In conventional expert systems, knowledge is described in such textual forms as rules and logic programming languages. By supplementing a system's facility for knowledge construction and conservation with visual representation, users can build and maintain their own intelligent systems without the aid of programmers or knowledge engineers.

During product design, various kinds of design knowledge are required. To realize an environment in which designers can make full use of the stored knowledge available to them, the knowledge should be organized so that it can be looked up easily and applied appropriately.

From this point of view, our primary focus is on a kind of knowledge which links relatively fragmentary knowledge in various forms. We call such knowledge *navigational knowledge*, while the fragmentary knowledge is called *elementary knowledge*. Navigational knowledge plays an essential role in design, since it provides designers with navigational information which helps them decide the next course of action or give them hints for novel ideas. It should be noted that the distinction between elementary knowledge and navigational knowledge is relative.

3 VISUAL KNOWLEDGE BASED INTELLIGENT CAD

In this section, we propose a general methodology to organize and utilize design knowledge through visual knowledge representation. The proposed methodology is composed of three steps:

1. Visualization of Knowledge Utilization

   This is concerned with how design knowledge is presented to the user. Visual navigation capability greatly improves the "*user friendliness*" of the system.

2. Visualization of Knowledge Organization

   In order to generate the above navigational knowledge with a minimum of effort, the task of knowledge construction and maintenance should also incorporate knowledge visualization techniques.

3. Knowledge Capture from Visual Information

   A good way to more dramatically reduce knowledge construction effort is to acquire design knowledge by mining existing design data. Knowledge extraction from visual information is particularly important for acquiring design knowledge.

These three steps, as shown in Figure 1, form the evolving process toward the development of visual knowledge based intelligent CAD. In the following sections we will describe how each of the above steps is accomplished.

3.1 Visualizing Knowledge Utilization

As the use of computers continues to expand and modern designers become more reliant on computer tools, design knowledge essential to maintaining design quality and efficiency tends to be computerized all the more. In order for a designer to utilize such design knowledge appropriately, however, this knowledge should be organized for easy access and use. The visual representation technique serves this purpose in several ways as follows.

Hyper Linking of Design Information

As typically shown in the recent diffusion of World-Wide Web (WWW) information services, hyper linking mechanisms provide users with the means to look for useful information in a search space with the help of spatially arranged clues. Since most design data and

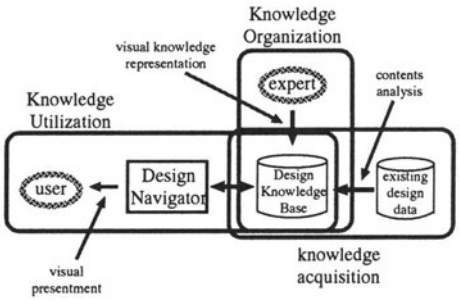


Figure 1 Visual Knowledge Based Intelligent CAD Architecture.

knowhow requires visual representation by nature, visual navigation among such information should be effective, especially when users take the initiative in using navigational knowledge.

Contextual Linking of Design Information

While hyper linking mechanisms support only one type of navigation which is restricted to direct mapping from one data to another, areas of design knowledge are generally related to each other in more complicated ways, thus making hyper linking facilities unsuitable for representing real world design knowledge. Since there usually exist complicated constraints and correspondence among many design factors and relevant decisions should be made according to design situations, navigation facility is also required to be able to describe context-dependent conditional branches concisely. Conventional relational DB systems are sufficient for relationships between formal and uniform data, such as dimensional relationships regarding standards and the relationship between a type of material and its strength. However, most design knowhow includes more informal and diversified design knowledge, which requires a more sophisticated representation.

The above contextual linking facility can be achieved by an expert system, composed of an inference engine for navigation and a knowledge base that stores knowledge on context dependency. The navigation engine in the system gathers information regarding the current situation by referring to data, calculating data and directing queries to users. After that, the system either presents the most appropriate information or provides advice or warnings.

Visualization of Design Knowledge Execution

While the above two linking facilities merely play a supporting role when a designer searches for related information, some of the retrieved knowledge is directly applicable to current design. Such applicable design knowledge includes the knowledge for the following elementary tasks:

- constraint check
- constraint satisfaction
- engineering analysis, simulation and evaluation
- automatic calculation, expansion and development

and combinations of the above.

In addition to visualizing the results of each design tool, visualizing the combination of design tools and the relationship between them is also important to give current design status feedback to designers.

3.2 Visualizing Knowledge Organization

Many types of visual representation have played important roles in design over a long time reusability[McKim(1972), Lohse, et al.(1994)]. From the viewpoint of representing design knowledge, two types of visual representation; table and diagram, are particularly promising because of their simplicity and expressiveness.

Tabular Knowledge Representation

Tabular representation is widely used to show the relationship between data. This representation is especially effective when there is a large amount of data with a uniform structure and it is necessary to display them simultaneously for comparison.

The types of design knowledge suitable for description in tabular representation include:

1. design object's attributes and their values for surveys and comparisons
2. relationships between parameters (ex. dimensional constraints)
3. correspondence between a conditional branch and its results (ex. next action to take, correspondence regarding values)

By using these types of design knowledge in a tabular representation, knowledge creation and revision by designers themselves becomes possible and its increases due to its comprehensibility.

Diagrammatic Knowledge Representation

Diagrammatic representation seems to be one of the most natural forms of knowledge in the field of design. Various types of diagrams are extensively used through the design process and they often convey essential design information.

Some of them can also be described using tabular representation. Which representation has greater advantages is dependent on the problem's size and characteristics.

For other types of knowledge, such as sequential operations regarding design, diagrammatic representation is favorable since it gives a general view of the structure.

The types of design knowledge suitable for description in diagrammatic representation include:

1. design object's structural information (ex. relationship between its elements)
2. conditions for decisions or selection and resulting branch flow
3. geometrical or spatial relationships (ex. constraints in layout)

Productivity and maintainability of knowledge would greatly be improved by providing the facility to directly manipulate diagrams to generate and revise this knowledge.

3.3 Acquiring Implicit Knowledge from Visual Representation

The above visual knowledge representations help experts to organize design knowledge and they certainly reduce construction and maintenance cost of design knowledge bases.

This effect is especially significant when applied to organize information that is used repetitively.

However, a great deal of knowledge exists that is implicitly stored in computerized forms. As this knowledge is not frequently referred to, it is difficult to counterbalance its input cost and its utility. However, some knowledge is often essential to design. If this implicit design knowledge can be extracted from already computerized data, it will greatly contribute to overcoming the "*knowledge acquisition bottleneck*". This kind of knowledge acquisition requires the following facilities:

1. Schema Extraction from Texts
   extracting keywords and structure from (natural language) texts
2. Schema Extraction from Diagrams
   extracting structures and keywords from diagrammatic representations
3. Schema Extraction from Images
   extracting structures from image representations or sketches

The labor cost in the knowledge acquisition process can be greatly reduced if the above facilities are provided and the extracted results are transferred to relevant programs including those for retrieval, classification and analysis. As for design knowledge acquisition, the second category is particularly important, since various types of diagrams are extensively used throughout the design process.

4 VISUAL TOOLS AND THEIR APPLICATIONS

In this section, we illustrate three tools that are capable of handling visual design knowledge. These tools are used to build intelligent CAD systems, based on the methodology in the previous section, to measure the relevance of visual representation to organize and acquire design knowledge.

4.1 Visual Knowledge Organization Tool

To achieve visual design knowledge organization, some tool allowing ease of use is necessary to construct and maintain design knowledge.

With only a direct link between design information, the system is unable to contain sufficient design knowledge to handle real-world design problems. Adding a capability to represent the contextual link between design information is necessary to build an intelligent system.

Many expert systems have been developed to realize such intelligence. In most conventional expert systems, however, those relationships are represented by if-then rules. In these rule-based systems, the inference mechanism evaluates the conditional part of the rules and chooses an applicable rule. Though rule representation is claimed to be easy to describe and maintain, the truth is that coordination and adjustment between rules is so delicate that it requires a specialist called a *knowledge engineer* to tune this knowledge carefully when constructing and maintaining knowledge bases.

If the target domain requires only a fixed knowledge base which is not frequently updated, the above rule-based approach would suffice. However, in domains that require

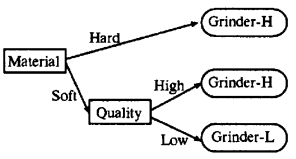
frequent additions and changes to knowledge bases, a design expert system is not practical without a capability by which expert designers can maintain the knowledge bases by themselves.

DT

To overcome the "*knowledge acquisition bottleneck*", DT [Koseki, et al.(1991)] was developed to help build intelligent systems, especially those for classification tasks. The classification problem involves finding appropriate classes for unknown concepts, given a set of attribute value conditions for the concept. Consider the simple example of a grinder selection problem in a factory. The problem is to select an appropriate grinder for a product, according to product requirements, such as material features and required quality. An example of grinder selection knowledge, described in decision table form is shown in Figure 2(a). Figure 2(b) shows another description of the same knowledge in a decision tree form.

	Material		Quality	
	Hard	Soft	High	Low
Grinder-H	○	×	○	○
Grinder-H	×	○	○	×
Grinder-L	×	○	×	○

(a) Decision Table Form



(b) Decision Tree Form

Figure 2 Example Decision Table and Decision Tree in DT.

By adopting visual knowledge representation and an intelligent query control mechanism, DT separates control knowledge from logical knowledge in the knowledge base. In DT, tabular representation and tree representation are used to express concept functions in DNF (Disjunctive Normal Form), making it possible to separate control knowledge from classification knowledge. Control knowledge is used by the query control mechanism in the problem solving cycle, called the *candidate elimination cycle*, to determine the most appropriate questions to ask in given situations.

DT enables users to create, add and edit knowledge in knowledge bases without the help of knowledge engineers. DT has been successfully applied to many practical applications, including hardware diagnosis, machining tool selection, financial consultation and tourist information guidance.

Design Knowhow Base System

The grinder selection example above shows that DT can describe some types of expert knowledge in the domain of mechanical design. In addition to representing such elementary design knowledge, DT is also expected to represent navigational knowledge. To measure the relevance of the visual knowledge organization tool to design tasks, we applied DT to build an intelligent CAD system for mechanical design.

As shown in Figure 3, the system is composed of three subsystems: a conventional

CAD system, an engineering DB (Product Model DB), and an expert system (Design Navigation System) built by DT.

Product Model DB stores various data and information regarding products, from drawings and manufacturing data to standards and design knowhow. For structured design data that can be retrieved via the normal DB retrieval interface, the Product Model DB's own facilities are used. Conversely, less structured design information, such as piecemeal knowledge on design principles, rationale and design knowhow, is retrieved through the DT-built Design Navigation system.

The contexts in which each piece of design knowhow should be applied, for example, are defined using DT's decision table editor or decision tree editor. When users want to obtain information during design, they specify keywords and provide answers to questions the system asks (if necessary). The Design Navigation system searches for relevant information (including drafts, tables, text and documents) according to the current context, and presents it to the user.

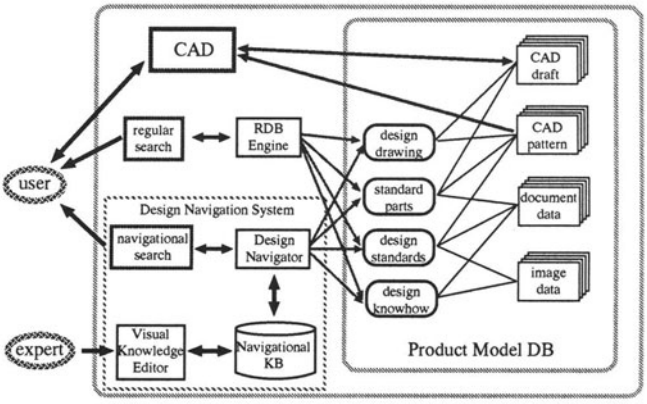


Figure 3 Intelligent CAD with Design Knowhow Base.

4.2 Visual Programming Tool

Although CAD systems in the early years were no more than drawing tools, modern CAD systems provide various kinds of facilities, including facilities allowing cooperation with external tools (programs). Users should know how to make use of these facilities. With increases in the number of available tools and facilities, increasing the number of possible combinations, present-day CAD users are required to have detailed operational knowledge of those tools, most of which have little relationship to design itself from the designer's viewpoint.

To lighten these burdens imposed on designers, many companies have organized staffs who administrate CAD systems and support their users. Though it is possible for the supporting staffs to develop special purpose CAD systems based on designer's demands, such customization usually takes time and the resulting system lacks flexibility.

For dramatic solutions to the above problem, "*end-user programming*" capabilities should be sought. End-user programming by designers can be achieved if:

- primitive functions of CAD tools and programs are standardized and implemented as components.
- components can easily be combined by designers in the way they want.

Emerging object-oriented programming techniques are applicable for tool standardization and decomposition into components, As the means of combining these components, conventional systems usually provide textual programming languages to describe macros and scripts, which requires designers to learn a programming language. The visual programming [Shu(1988), Ambler and Burnett(1989)] capability is expected to overcome this shortcoming, since users can generate new application programs by just assembling program components visually.

VIP

VIP [Koike, et al.(1994), Koike, et al.(1995)] is an iconic visual programming tool in which program components are represented as box icons and connections between components are represented as wires (Figure 4).

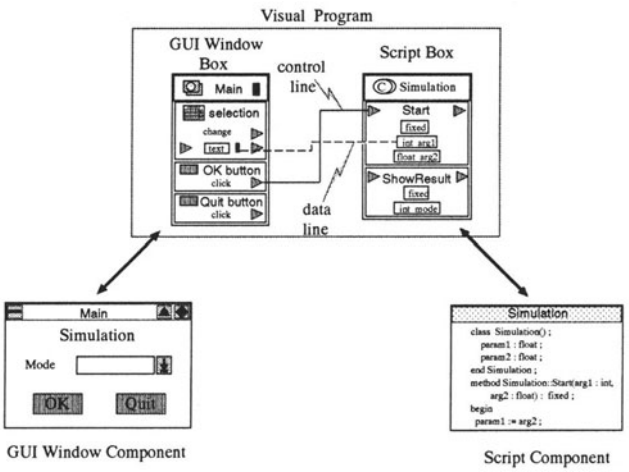


Figure 4 Visual Programming Tool VIP.

A box corresponds to a program component (an object), and there are two types of components: basic components and functional script components. *Basic components* include GUI components, DB access components, table components and form components. These components are visually developed with their respective building tools. *Functional script components* are composed of standard libraries and application-specific components defined with a script editor. Two types of connections between components, control lines and data lines, are provided to define control flow and data flow respectively. Since each component corresponds to an object, the composed application programs have high modularity and reusability. To supplement the iconic programming's deficit in scalability, VIP

allows users to define textual scripts as components and use them in the same way as others defined with visual component building tools.

Design Flow Control System for VLSI CAD

Current electronic circuit design requires various types of CAD tool programs applied to circuit design. These CAD tool programs include logical verification, timing simulation, layout rule checks and path analysis.

As design targets become larger and more complex, each CAD tool adds new features and the number of tool applications increases. To achieve better design quality in a limited time, efficient coordination between tools is essential. To achieve this coordination, we built a visual design flow control system based on VIP. Figure 5 shows an outline of the system.

The system incorporates an iconic programming editor, in which a box represents either a CAD tool program, a GUI object or a visual programming element (such as a conditional branch and a variable). When designers want to define new design flow, they select several boxes representing CAD tool programs, GUI objects or visual programming elements. By connecting lines between these boxes, a design flow is defined. The resulting design flow can be executed on the spot by a visual program interpreter. When a design flow is saved as a component, a shell program is generated, which can be executed in a batch job afterwards (at night or holidays, for example). These design flow programming facilities have made it possible for designers to represent their knowledge of design flow and share such knowledge with other designers.

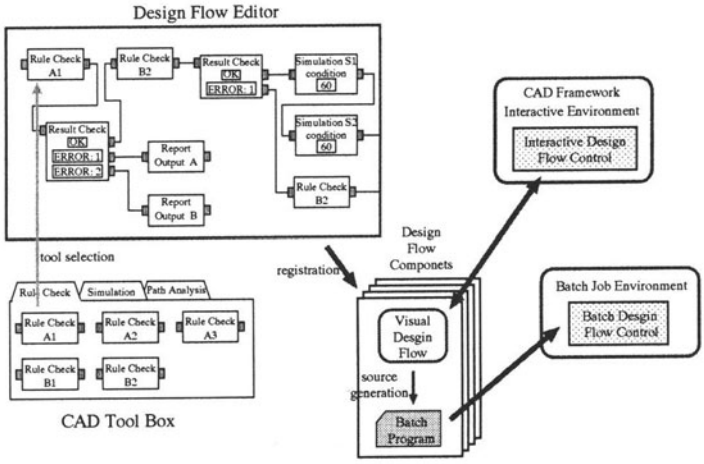


Figure 5 Visual Design Flow Control System.

4.3 Electronic Design Notebook and Spatial Parsing Tool

Though visual knowledge programming tools greatly reduce the difficulties designers have in constructing and organizing design knowledge by themselves, they still require some sort of programming.

One important goal is to realize a tool which effectively helps designers to extract implicit knowledge from existing design information. This implicit knowledge cannot be extracted without the semantic analysis of information content. In general, investigation of textual contents, in which use is made of natural language processing techniques, accounts for the major part of such knowledge extraction. In the design process, however, the capability for handling design information stored in a graphical or visual form is equally important. Spatial parsers for visual languages help such content analysis by transforming less formal information into a more usable manner.

Though many parsers have been developed for specific purposes [Lakin(1987), Wittenburg and Weitzman(1990), Yu and Chang(1990), Helm, et al.(1991)], they require the user to work in accordance with fixed grammars defined in advance, thus reducing their use especially in the earlier design stages. We proposed a framework to adapt a generic grammar to fit a given diagram [Maeda(1993)]. Figure 6 illustrates the overview of "adaptive spatial parsing" method.

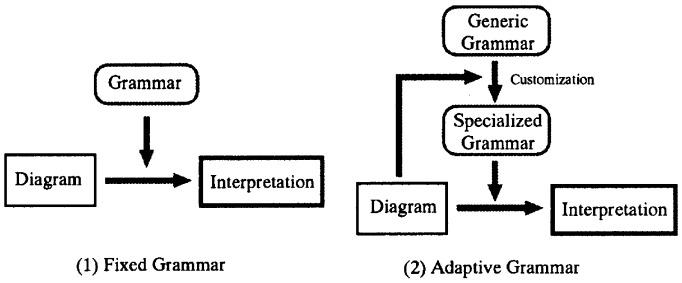


Figure 6 Adaptive Spatial Parsing.

EDN

The Electronic Design Notebook (EDN) [Lakin, et al.(1989)], developed by Stanford University, is a prototype of an integrated design environment, in which designers can perform with as much agility as a notebook without losing the benefits of processing power. The design stages which EDN is supposed to cover includes the upper level such as conceptual design, and the system's main goal is to balance the notebook's characteristics as a performing medium and processing medium.

EDN provides various facilities for designers, including drawing, documenting, communicating, and connecting to external programs. It also provides such facilities as recording and reproduction of operation history, and design information navigation using tags.

EDN is implemented on a general-purpose text-and-graphics editor called *vmacs**,

**vmacs is a trademark of the Performing Graphics Company*

which supports both the user's free performance and the computer's data processing capabilities. In *vmacs*, diagrams are composed of three types of visual objects:

textline text
drawline one or more continuous segments of lines
pattern arbitrary combination of textlines and drawlines

Visual objects are organized in a tree structure, and their spatial properties are easily accessible [Lakin(1986)].

Adaptive Spatial Parser for Diagram Schema Extraction

Based on the adaptive parsing framework, a prototype system has been developed, using *vmacs*'s visual object processing facilities, to parse a variety of node-link diagrams,

The system is composed of three parts:

generic grammar composed of spatial taxonomy and generic rules.
grammar customization process generates a specific grammar by selecting rules from the generic grammar and refining them.
grammar application process applies the refined rules to extract structural information from the diagram.

The spatial taxonomy in generic grammar is a class hierarchy of graphical object types and a generic rule defines the relationship between graphical objects and their role in a diagram. The system analyzes a given diagram drawn on EDN and customizes its grammar rules to fit them to the given diagram by repeating the "rule refinement cycle" shown in Figure 7. The system then parses the diagram and extracts schematic information. The system is aimed at extracting the schema for a diagram which becomes the input to an appropriate design expert system.

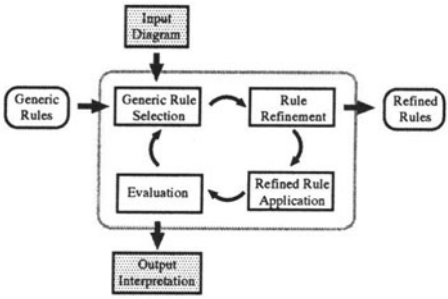


Figure 7 Rule Refinement Cycle in Grammar Specialization.

The PERT (Program Evaluation and Review Technique) diagram shown in Figure 8 is a typical example of the node-link diagrams used in design. Though it is mainly used in project management and task scheduling, similar diagrams are widely used in design.

As a result, three specialized rules have been established to describe the given diagram, and they yield an interpretation that is composed of nodes, directed links, labels of nodes, and labels of links, as follows:

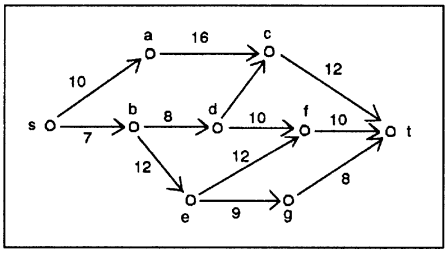


Figure 8 PERT Diagram.

```
($node obj2 obj8 obj12 obj17 ......)
($directed-link (obj3 obj2 obj12) (obj4 obj2 obj8)
                (obj9 obj8 obj17) ......)
($label-node (obj1 obj2) (obj7 obj8) (obj13 obj12) ......)
($label-link (obj5 obj3) (obj6 obj4) (obj10 obj9) ......)
```

The bond graph shown in Figure 9 is another typical example of node-link diagrams frequently used during conceptual design [Thoma(1990)].

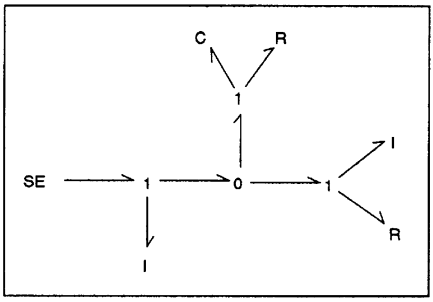


Figure 9 Bond Graph.

In Figure 9, three types of elements are identified: single arrows, integer text and symbol text. As in the PERT diagram, the rule selection process selects the "link-directed" rule, and the rule refinement process modifies its class specifications as follows:

```
subject $0 : single-arrow  ->  single-arrow
 object $1 : text or closed-graphics  ->  integer-text
 object $2 : text or closed-graphics  ->  text
```

The rule activation process then applies this modified rule to eight arrows in the diagram out of nine, leaving only one arrow (starting from node "SE" unresolved). In the next rule refinement cycle, the rule selection process selects the link-directed rule again, whose class specifications are then modified as follows:

```
subject $0 : single-arrow  ->  single-arrow
 object $1 : text or closed-graphics  ->  symbol-text
 object $2 : text or closed-graphics  ->  integer-text
```

While the above two rules acquired from the given diagram do not capture sufficient grammatical characters to regulate all valid bond graphs, these rules, to a certain extent, can be used to check the validity of given graphs.

Since the ability to adapt generic grammar to given diagrams improves the precision of parsing, the proposed technique should contribute to expanding the ability of computer systems to handle diagrams and other visual information.

4.4 Effects of Visual Representation

As stated above, we have implemented three intelligent CAD systems based on the proposed framework featuring knowledge visualization techniques. We discuss the effectiveness of visual knowledge representation in these systems, from the following viewpoints: visual knowledge utilization, visual knowledge organization, and visual knowledge acquisition.

Visual Knowledge Utilization

In the design knowhow base system, two types of retrieval are available: attributive retrieval by ordinary RDB facilities and contextual retrieval by DT. The DT's contextual retrieval facility provides users with design data and knowhow by directing necessary queries to them. In this case, DT works as a "navigator" to save the user's labor in the search.

Visual Knowledge Organization

In the knowhow base system, DT was used to define contextual links between elementary knowledge. DT's decision table editor and decision tree editor were shown to be much more productive and efficient than rule-based expert systems.

In the VIP-built design flow control system, designers can visually define design flow without programming and store it in such a way as to make it easily reusable.

As for more complex flow controls, VIP's visual programming capability makes it possible to define the user's own "navigator" interface visually, that is, by generating GUI components and defining the relationships between them.

Knowledge Acquisition from Visual Information

In the DT and the VIP, composed design knowledge and programs can be test-run on the spot. Since this allows users to obtain visual feedback immediately, labor and cost for the user have been considerably reduced.

From the viewpoint of realizing automatic knowledge acquisition, it has been shown that the node-link diagram parser on EDN has the capability of extracting schema information from different types of node-link diagrams.

Automatic knowledge acquisition could be attained by combining such a design content extraction parser and programs to analyze extracted schema. So far, we have only developed the diagram parser, and need to further investigate potentialities for automatic knowledge extraction.

5 CONCLUSION

As artifacts have increasingly become more complex and are composed of a greater number of elements, computer tools have become indispensable to designers. With increasing demands to achieve still better design solutions within a shorter time, designers have been required to make the most of the design knowledge at their disposal. However, most existing design knowledge is either stored in an "intelligent program", in such a way that designer interpretation is required to gain access to it, or it is left unrecorded from the beginning.

To handle such a wide variety of design knowledge and to organize it into a usable manner, we proposed a framework to organize and acquire this knowledge featuring knowledge visualization techniques. To show the validity of the proposed framework, three visual tools were investigated and employed to manage navigational design knowledge. Applications for a visual knowledge organization tool (DT) and a visual programming tool (VIP) show that knowledge visualization greatly improves the productivity and maintenability of navigational knowledge. Application for spatial parsing tool shows the possibility of automatic knowledge extraction from existing design data, which may lead to the elimination of the knowledge acquisition bottleneck. In addition to strengthening and enlarging the spatial parsing approach, we also anticipate user operations to provide the key to extract implicit knowledge in the future.

Acknowledgements

The authors are grateful to Larry Leifer, David M. Cannon, Vinod Baya and Fred Lakin of Stanford University for their insightful discussions and helpful suggestions in developing the adaptive spatial parser on EDN. The authors also express their appreciation to Masahiro Yamamoto and Masanobu Watanabe of NEC Corporation for giving them the opportunity to pursue this research.

REFERENCES

Ambler, A.L. and Burnett, M.M. (1989) Influence of Visual Technology on the Evolution of Language Environments. *IEEE Computer*, **22(10)**, 9–22.

Helm, R., Marriott, K. and Odersky, M. (1991) Building Visual Language Parsers. *Proceedings of CHI'91*, 105–112.

Koike, Y., Maeda, Y. and Koseki, Y. (1994) Mixing GUI Layout with Visual Programming. *Proceedings of 1994 IEEE Youth Forum in Computer Science and Engineering.*

Koike, Y., Maeda, Y. and Koseki, Y. (1995) Improving Readability of Iconic Programs with Multiple View Object Representation. *Proceedings of 1995 IEEE Symposium on Visual Languages*, 37–44.

Koseki, Y., Nakakuki, Y. and Tanaka, M. (1991) DT: A Classification Problem Solver with Tabular Knowledge Acquisition. *Proceedings of Third International Conference on Tools for AI*, 156–163.

Lakin, F. (1986) Spatial Parsing for Visual Languages, in *Visual Languages* (eds. S. Chang, T. Ichikawa and P. A. Ligomenides), 35–85, Plenum.

Lakin, F. (1987) Visual Grammars for Visual Languages. *Proceedings of AAAI-87*, 683–688.

Lakin, F., Wambaugh, J., Leifer, L., Cannon, D.M. and Sivard, C. (1989) The Electronic Design Notebook: Performing Medium and Processing Medium. *Visual Computer*, **5**, 214–226.

Lohse, G. L., Violsi, K., Walker, N. and Reueler, H.H. (1994) A Classification of Visual Representations. *Communications of the ACM*, **37(12)**, 36–49.

Maeda, Y. (1993) Grammar Customization for Spatial Parsing. *Proceedings of WISS'93*, 257–264.

McKim, R. H. (1972) *Experiences in Visual Thinking.* Wadsworth.

Shu, N.C. (1988) *Visual Programming.* Van Nostrand Reinhold.

Thoma, J.U. (1990) *Simulation with Bond Graphs: An Introduction to a Graphical Method.* Springer-Verlag.

Wittenburg, K. and Weitzman, L.. (1990) Visual Grammars and Incremental Parsing for Interface Languages. *Proceedings of 1990 IEEE Workshop on Visual Languages*, 111–118.

Yu, B. and Chang, S-K. (1990) A Fuzzy Visual Language Compiler. *Proceedings of 1990 IEEE Workshop on Visual Languages*, 162–167.

Yasuyuki MAEDA received his B.E. and M.E. degrees in precision machinery engineering from University of Tokyo in 1982 and 1984, respectively. He joined NEC Corporation in 1984, and he has been engaged in the research of 2D/3D CAD systems, expert systems for CAD/CAM, and visual programming systems. He is now Assistant Manager of C&C Research Laboratories.

Mr. Maeda is a member of the Japan Society for Precision Engineering and the Information Processing Society of Japan.

Yoshiyuki KOSEKI received his B.E. degree in computer science from Tokyo Institute of Technology in 1979 and M.S. degree in computer science from University of California at Los Angeles in 1981. He received his PhD from Tokyo Institute of Technology in 1993. He joined NEC Corporation in 1981, and he has been engaged in the research of artificial intelligence, expert systems and visual programming systems. He is now Research Manager of C&C Research Laboratories.

Dr. Koseki is a member of the IEEE, the AAAI, the Japanese Society for Artificial Intelligence and the Information Processing Society of Japan.

Yuichi KOIKE received his B.E. and M.E. degrees in precision machinery engineering from University of Tokyo in 1990 and 1992, respectively. He joined NEC Corporation in 1992, and he is now a member of C&C Research Laboratories. He has been engaged in the research of visual programming systems.

Mr. Koike is a member of the Information Processing Society of Japan.

Midori TANAKA received her B.L.A. degree in physics from International Christian University in 1986 and M.E. degree in computer science from Tokyo Institute of Technology in 1988. She joined NEC Corporation in 1988, and she has been engaged in the research of knowledge representation, expert systems and AI tools. She is now Assistant Manager of C&C Research Laboratories.

Ms. Tanaka is a member of the Information Processing Society of Japan.