INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED'07
28 - 31 AUGUST 2007, CITE DES SCIENCES ET DE L'INDUSTRIE, PARIS, FRANCE

ONTOLOGY BUILDING FOR DESIGN KNOWLEDGE MANAGEMENT SYSTEMS BASED ON PATTERNS EMBEDDED IN DESIGN-FOR-X METHODOLOGIES

Yutaka Nomaguchi and Kikuo Fujita
Department of Mechanical Engineering, Osaka University, Japan

ABSTRACT

This paper discusses a viewpoint of ontology building as a means toward a context-rich computer-supported knowlede-based design environment from the patterns embedded in design-for-X methodologies. Ontology-based knowledge systems have become an efficient way for supporting intelligent human activity, but the contents of the ontology dominates their performance. On the other hand, design methodologies, typically design-for-X methodologies, are prescriptions that advocate how design should be done in particular circumstances. That is, taxonomy of design and a pattern of design operation can be seen in a design methodology. This is what an ontology is. Therefore, a design methodology can be a basis of defining ontology for a knowledge management oriented design support system, although a design methodology is originally made for practical use without taking care of its ontology. This paper discusses the contrast between design-for-X methodologies and knowledge-based design support systems, and then explores an ontology implementation from a QFD-based cost-worth analysis method as an example of design methodology. The validity and promise of the viewpoint is demonstrated through its application to a case study over a prototype system.

Keywords: Design Knowledge, Ontology, Design for X Methodology, Knowledge Management Systems

1 INTRODUCTION

The increasing demand for rationalizing design activities launches a wide branch of engineering design research. While knowledge-based approaches initiated in 1980s under the paradigm of expert systems, a current challenge is to handle complicated design knowledge and related information more sophisticatedly and flexibly. It was a remarkable step that knowledge processing has expanded a computational support to upper streams of design process. Today, ontology-based knowledge processing is getting much attention for enhancing capability of product data management systems, etc. The concept of ontology is to represent knowledge over shared meta-data and to manipulate its contextual structure under the standpoint of computer engineering and apart from design engineering. On the other hand, design-for-X (DFX) methodologies, such as quality function deployment (QFD), design for assembly (DFA), etc., had been highlighted in the late 1980s as a part of concurrent engineering movement and other DFX methodologies has succeeded in 1990s and later. They are proceduralization of appropriate or superior design operations in product definition or conceptual design stages. While their development has been mainly achieved in empirical ways, their contents can be novel prescription of rational design operations as a result. When contrasting the meaning of meta-data in ontology and the prescriptiveness of DFX methodologies, both are related to contexts of the design activities. Thus, there is an expectation that more useful ontology can be deduced from the patterns hidden in DFX methodologies and that such deduction expands the performance and potential of knowledge-based design support systems. Under this standpoint, this paper discusses ontology building from DFX methodologies.

The objective of this research is to establish a vision for enhancing knowledge-based design support systems by using more useful and sophisticated ontology that is deduced from the patterns hidden in DFX methodologies. Such deduction is expected to essentially expand the performance and potential of knowledge-based design support systems under the above discussion. This research extracts the system of ontology from a QFD-based cost-worth analysis method [1] and

experimentally implements a prototype system of knowledge-based design support system based on it and in an object-oriented language, Java. The prototype system is integrated with our design rationale acquisition mechanism, which hierarchically integrates designer's actions, operations over product models and structure of associated argumentation [2,3]. The system is applied to a simplified design problem of cellular phones for validating the vision of building ontology from the patterns embedded in design-for-X methodologies.

2 CONTRAST OF DESIGN-FOR-X METHODOLOGIES AND DESIGN KNOWLEDGE MANAGEMENT SYSTEMS

2.1 Prescription and Management of Contexts in the Design Process

A purpose of a design methodology is to have prescriptions that advocate how design should be done in particular circumstances [4]. Many researchers of engineering design have proposed various design-for-X (DFX) methodologies. Among various DFX methodologies, QFD [5] is a typical and comprehensive one. It is effective for exploring and defining design requirements by a sequential procedure, for instance, across customer's requirements, functional realization, manufacturing modules, production process, etc. By means of its reflective refinement, a designer gets overall image of a product. While such an instruction is easy to understand for designers, it consists of some general and abstract concepts by excluding its case-dependent aspects. Therefore, it is suggested that QFD promotes more efficient and effective information sharing and discussion among designers in practical use [6]. Although a DFX methodology is not a theory established by a scientific law such as physics, it is often used in actual design cases as a rational prescription of design contexts.

The concept of ontology was developed apart from design engineering. It is to represent knowledge over shared meta-data and to manipulate its contextual structure under the standpoint of computer engineering. The usage of ontology in information systems enables to manage a mount of knowledge across different domains, tasks, disciplines, etc. more interoperably and more comprehensively. Ontology is originally a term of philosophy that means theory of existence. From artificial intelligence (AI) point of view, an ontology is defined as "explicit specification of conceptualization" [7]. This is widely accepted in the fields of AI and informatics. For instance, Kitamura has been developed a meta-data schema for systematically representing functionality of a product based on Semantic Web technology for the management of the information content of engineering design documents [8]. This research also adopts this definition.

Figure 1 shows how DFX methodologies and ontology-based knowledge management system are beneficial for more complicated design problems. As shown in the figure, both techniques deal with different aspects of design contexts as discussed in the above. An approach based on a DFX methodology can support a designer by means of prescription of superior design. An information

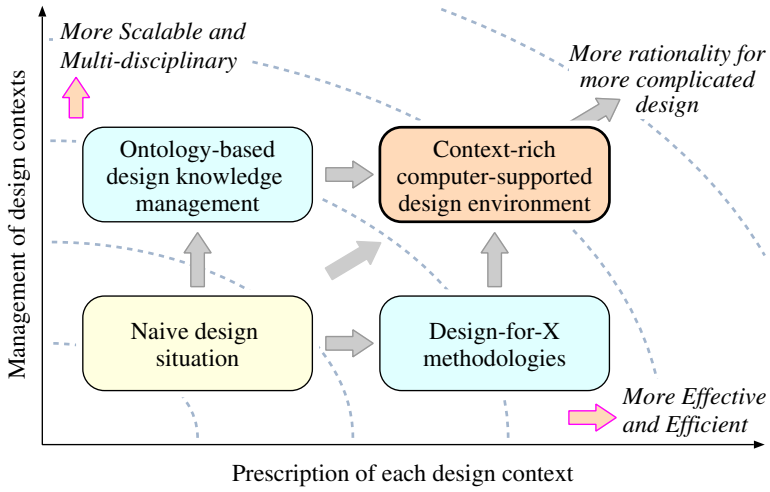

Figure 1. More rational design through context-rich design environment based on DFX methodologies and ontology based computer supports

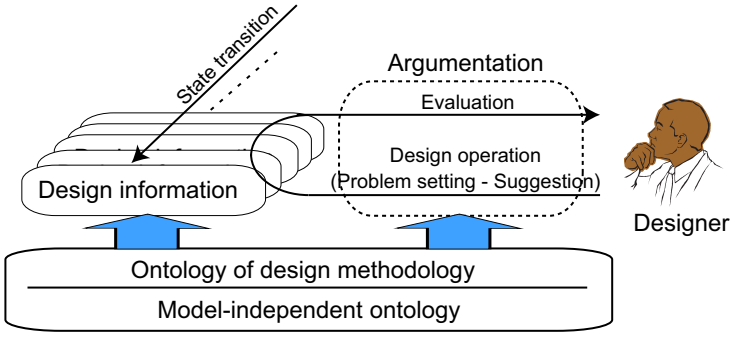

Figure 2. Outline of reflective design process supported by a knowledge-based design support system

system based on ontology can manipulate large-scaled and multi-disciplinary knowledge. Therefore, the integration of both techniques is a probable framework toward context-rich computer-supported design environments and it is expected to be a base for implementing a design support system that has high intensiveness of various prescriptive design knowledge.

2.2 Framework of Knowledge-based Design Support Systems

While a DFX methodology gives a prescription of rational design, design process is inevitably iterative and reflective process due to the open-ended nature of design problems. For supporting designers under such complicated difficult process, we have been researching a knowledge management oriented design support system that dynamically acquires such reflective design process as a by-product of design [2, 3]. By using this system, a designer can compare multiple alternatives concurrently during design process, and can also review a rejected design alternative. Figure 2 shows the outline of the situation that a designer is involved under the knowledge management with the support system. This system consists of two subsystems; a subsystem that captures state transition of design information and a subsystem that captures an argumentation structure. The former subsystem adopts a simple mechanism based on a truth maintenance system (TMS) [9] to capture all design states and to track each of them anytime without redundancy and incorrectness. The latter subsystem adopts an IBIS-based model [10] to represent argumentation structure. This framework integrating TMS and IBIS is called DRIFT (Design Rationale Integrating Framework of Three-layers).

Under the mechanism of DRIFT, each design operation is defined as a pattern of problem setting and alternative solution in order to capture both design state transition and an argumentation structure. For example, a design operation to detail a customer need of a product is defined as follows; a problem set by the operation is ‘what are sublevel customer needs of it?’ and its alternative solution is a set of its sublevel customer needs. Since the system records all alternatives suggested as solutions of the problem, a designer can compare them and review one at any time.

An ontology is the meta-data structure that enables capturing design state transition and argumentation structures almost automatically through designer’s inherent design actions. While the system includes a set of fundamental ontologies for implementing basic functionality, another set of subsidiary ontologies for capturing and managing practical and complicated design operations is needed. Thus it is necessary and essential that an set of specific ontologies must be formulated for supporting a particular type of design operations in order to apply a design methodology under the corresponding context.

2.3 Principle for Extracting Ontology from DFX Methodologies

As a foundation of a knowledge-based design support system, an ontology should be formulated for defining two concepts; taxonomy of design to manage complicated design information through design process, and patterns of problem setting to provide design operations. This research states that both of them are embedded in DFX methodologies. A methodology has taxonomy of design information by necessity, and has a pattern of design operation.

For example, QFD is a series of two-dimensional tables between system decompositions in different aspect spaces for mapping and arranging their consistency and soundness toward product integrity. This viewpoint implies the following concepts for product representation and design

operations. A product is represented in different views. It has a top node and is hierarchically and recursively decomposed into sub-nodes in each view. Associated nodes are linked each other across difference views. Each node or linkage has attributes characterized it. Each attribute has a value.

QFD operations are syntactically composed of definition of the top node, its decomposition into subsidiary nodes, assigning level of importance to each subsidiary node, assigning level of contribution of a node in a view to another node in another associated view, evaluating mapping of levels of importance across a series of views, and overall procedure and its outcomes are reflectively refined for accommodating their mutual compromise. Contextual structure of these all can be a base of building ontology for supporting a specific design activity under the knowledge-based design support system.

3 BUILDING DESIGN ONTOLOGY

3.1 Example of DFX methodologies and Types of Design Ontology

In this research, a QFD-based cost-worth analysis method [1], which was originally configured for exploring the conceptual transition of product definition over product life cycle, is adopted as an example of DFX methodologies. An ontology corresponding to it is configured as a base of knowledge-based design support system. When the perspective is shifted to value graph, for instance, a modified concept is a set of operations for exploring nodes and constructing their categorization into respective views and structuring their hierarchy in each view. That is, they can be formulated over shared ontology with small modification. This kind of expansions is expected to be feasible for other methodologies such as cost-worth graph that is a tool to evaluate a balance of a product's cost and its worth [11].

An ontology proposed in this research consists of two layers:

Model-independent layer, which consists of concepts independent from any specific design methodology.

Model-dependent layer, which consists of concepts to represent specific prescriptive design methodologies.

A concept of the latter layer is defined as a subclass of model-independent layer concepts. The combination of the above two layers makes expansibility higher. When a new design methodology is integrated to the design support system, arrangement of new concepts is required only in the model-dependent layer.

3.2 Model-Independent Concepts

The following five concepts is defined as model-independent concepts. Figure 3 shows the definition in UML (Unified Modeling Language) format.

Element is the concept that constructs a product. This is further categorized into three concepts; customer need, function and entity. For example, an element of customer need of an cellular phone is 'good quality of call,' 'fascinating appearance,' etc., an element of function is 'to make a call,' etc., and an element of entity is 'battery,' 'radio transmission circuit,' etc.

Hierarchy is the concept that represents a hierarchical relationship between elements. A hierarchy node has an association to an element, which is a super level node of the hierarchy, and elements,

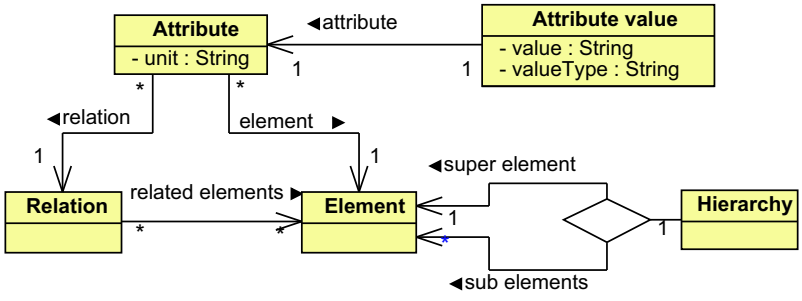

Figure 3. Model-independent concepts

which are sub level nodes of the hierarchy. For example, a hierarchy node represents that a customer need 'good cellular phone' has four sub elements; 'good quality of call,' 'fascinating appearance,' 'many value-adds' and 'high reliability.'

Relation is the concept that represents a relation between elements. For example, 'Function-Entity relation' is defined among a function 'to make a call' and entities 'radio transmission circuit,' 'baseband LSI,' 'antenna' and so on.

Attribute is the concept that represents a character of an element or a relation, such as 'cost.' An attribute node has an association to an element or a relation.

Attribute value is the concept that represents a value of an attribute. An attribute value node has a value and a value type as property.

3.3 Model-Dependent Concepts

A concept for representing specific methodology is defined as a subclass of a model-independent concept. There is a possibility that other concepts are defined in addition to concepts explained in this subsection when the other methodology is integrated.

3.3.1 Value graph, function-structure mapping

Value graph describes development of a top customer need 'good product' into sublevel customer needs. Function-structure mapping describes development of functions and entities of a product, and relationships between them. Three concepts; customer need, function and entity, are defined as sublevel concepts of element.

3.3.2 QFD two-dimensional tables

QFD two-dimensional table describes correlation factors between different aspects such as ones between customer needs and functions, ones between functions and entities, etc. These correlation factors are used to deploy weights of customer needs to weights of entities by simple matrix calculation. The following additional concepts are introduced to integrate QFD in the framework.

C-F relation and F-E relation are both sub class concept of relation. They are used to represent the existence of correlation between a customer need and a function, and one between a function and an entity.

Weight is a sub class concept of attribute. It is used to represent a weight of a customer need, a function and an entity.

Relation factor is a sub class concept of attribute. It is used to represent the correlation factor of a C-F relation or an F-E relation.

3.3.3 Cost-worth graph

Cost-worth graph describes a balance between relative worth and relative cost of an entity. The following additional concepts are defined to integrate cost-worth graph in the framework.

Relative worth is a sub class of attribute. This is calculated by regularization of weights of entities which are calculated by QFD two-dimensional tables.

Cost is a sub class concept of attribute. It is used to represent cost of an entity.

Relative cost is a sub class concept of attribute. It is used to represent relative cost of an entity. Its value is calculated by regularization of cost of entities.

3.4 Patterns of Problem Setting

A design process under the implemented methodologies is executed by the following steps; (i) making QFD two-dimensional tables of a product, (ii) estimating cost of entities, (iii) evaluating a balance of cost and worth, and go back to a former step if necessary. These steps are not proceeded straight forward but in iterative way. Therefore, a designer is required to reflectively consider elements, their relations, weight of customer needs and cost of entities represented in QFD two-dimensional tables. Ten patterns of problem setting are extracted for covering a whole design process in this methodology. The followings are their explanation by using a case study of cellular phone design as an example demonstrated in the next section.

3.4.1 What is a top element?

To make QFD two-dimensional tables of a product, a designer should enumerate elements of each view. Value graph and function-structure mapping help this process. In these tools, a designer firstly sets a top element of each view and then he/she details it to sublevel elements. When three views are considered in QFD, the following three problems should be considered.

1. *What is a top customer need?*

   For example, 'good cellular phone.' is a possible alternative of a top customer need.

2. *What is a top function?*

   'To access information when going out' is a possible alternative of a top function.

3. *What is a top entity?*

   'Cellular phone' is a possible alternative of a top entity.

3.4.2 What are sublevel elements?

When a designer uses value graph and function-structure mapping, he/she enumerates elements by detailing an abstract element to concrete sublevel elements. Detailed sublevel elements would be often detailed to more concrete sublevel elements. When three views are considered in QFD, the following three problems should be considered.

1. *What are sublevel customer needs?*

   Four elements; 'good quality of call,' 'fascinating appearance,' 'many value-adds' and 'high reliability' can be sublevel customer needs of 'good cellular phone.'

2. *What are sublevel functions?*

   As sublevel functions of 'to access information when going out,' functions such as 'to make/receive a call,' 'to send/receive an e-mail,' 'to browse a web page,' 'to take a picture' can be an alternative.

3. *What are sublevel entities?*

   As sublevel entities of 'cellular phone,' entities such as 'case,' 'speaker,' 'microphone,' 'antenna,' 'numeric keypad,' 'LCD display,' 'battery' can be an alternative.

3.4.3 How much is a customer need important?

A designer sets weights of customer needs by considering what kind of customers would be a target of a product. The weight is usually marked by three grades, such as 1, 3, 9. When customers who care value-adds of cellular phone are target, for example, the following mark can be an alternative solution; 'good quality of call: 3, fascinating appearance: 3, many value-adds: 9, high reliability: 1.'

3.4.4 Which elements are related?

By QFD two-dimensional tables, a designer considers relationships between elements and relation degrees. Since we consider three views, there are the following two problem settings.

1. *Which functions are related to a customer need?*

   A possible alternative solution of functions related to a customer need 'good quality of call' and their relation degrees is 'to make a call: 9, to receive a call:9.'

2. *Which entities are related to a function?*

   A possible alternative solution of entities related to a function 'to make a call' and their relation degrees is 'mic: 3, antenna: 3, numeric keypad: 3, radio transmission circuit: 9, baseband LSI: 9.'

3.4.5 How much is cost of entity?

A designer estimates cost of entity. For example, 'camera lens: 150, CPU: 100, LCD display: 50, ...' is a possible alternative solution for this problem setting.

3.5 Defining Design Operation

A design operation takes an important role in generating an argumentation structure for design knowledge management. This research defines ten design operations each of which corresponds to a pattern of problem setting defined in Subsection 3.4. For example, a design operation 'function development' is defined as follows.

Operation name: Function development
Refereed nodes: Function *f*
Added nodes: Function [] *subF*, Hierarchy *h*
Problem setting: What are sublevel functions of *f* ?
Solution: *subF*
Operation primitives: 1. adding *subF*, 2. adding *h*, 3. adding justification from *f* and *subF* to *h*

A design operation defines *operation primitives*, which are operations to TMS. When a design operation is performed, operation primitives of the list are performed. A design operation defines *referred nodes*, which are requisites of an operation, and *added nodes*, which are added as a result of an operation. In the definition of 'function development,' a function node is referred, and multiple function nodes and a hierarchy node are added as a result of this operation.

Under the above definition of design operations, the following procedure generates IBIS description after a design operation O_n is performed. The detail of this algorithm is shown in our preceeding articles [2, 3].

1. Creating a new issue node I_n, which has referred nodes of O_n in its focused node list, and has operation name of O_n in its operation type.
2. Searching an issue node I_s which is the same as I_n. If I_s is found, $I_n := I_s$.
3. Creating a new position node P_n, which has added nodes of O_n in its focused node list, and has operation name of O_n in its operation type.
4. Searching a position node P_s which is same as P_n. If P_s is found, $P_n := P_s$.
5. Searching a position node P_{n-1}, whose focused node list contains at least one of focused nodes of I_n.
6. Connecting *raised* relation from P_{n-1} to I_n.
7. Connecting *respond-to* relation from I_n to P_n.

Here, the *same* IBIS node is defined as a node which has the same operation type and the same focused node list.

4 DEMONSTRATING DESIGN EXAMPLE ON KNOWLEDGE-BASED DESIGN SUPPORT SYSTEM

4.1 Integration of Extracted Ontology to Design Support System

Based on the discussion above, the ontology that is extracted from QFD-based cost-worth analysis method [1] is integrated to the knowledge based design support system [2, 3]. The system was developed in Java programming language (jdk 1.4.1) on Windows XP. Figure 4 shows its architecture.

Under the integrated system, a designer carries out design by using tools of design methodologies; value graph, function-structure mapping, QFD two-dimensional tables and cost-worth graph. The system calls design operations to TMS when a designer input design information on the tools. Design process performed by a designer is automatically recorded in three levels; action level, model operation level and argumentation level along designer's actions and operations over the embedded tools. A designer can edit description of an argumentation structure. A recorded design process is stored in database in XML format.

4.2 Design Example

This subsection illustrates an application to designing a cellular phone for demonstrating the capabilities of the implemented system. In this example, it is assumed that a product is aimed to Japanese market and that it is equipped with many value-adds such as camera, music player, electric money, bar-code scanner, etc.

A designer enumerates customer needs, functions and entities of a cellular phone by using value graph and function-structure mapping. Then, he/she uses QFD table to set weights of customer needs

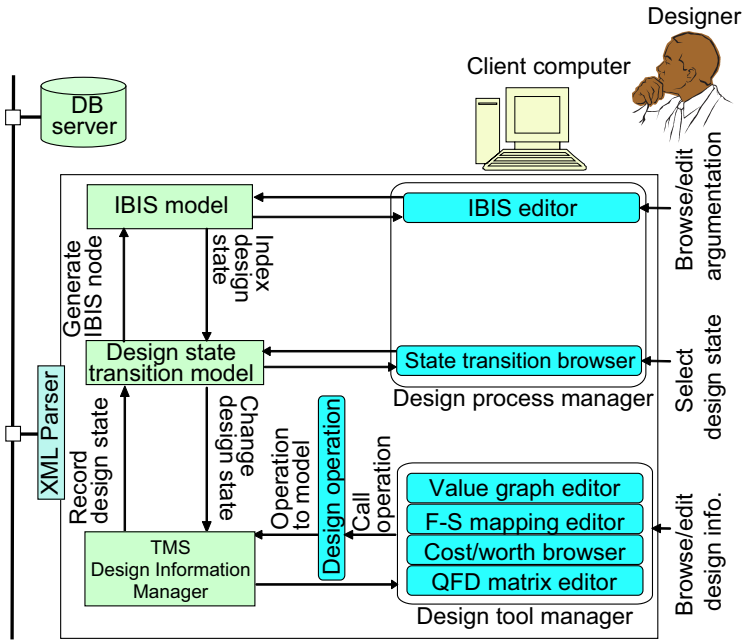

Figure 4. System architecture

(see Figure 5-①). The weights are deployed to weights of entities by automatic calculation of QFD two-dimensional tables. A designer also sets cost of each entity, and evaluates balance of relative cost and relative worth of each entities that cost-worth graph shows (see Figure 6-①). This graph reveals that cost of camera lens would be higher than its worth.

In order to dissolve this unbalance of cost and worth, a designer has to choose either from among two; (A) reducing relative cost or (B) increasing relative worth as shown in Figure 6. A possible solution can be enumerated according to the patterns of problem setting. According to problem setting 'how much is cost of entity?,' reducing cost of camera lens can be proposed as an alternative solution of (A). Otherwise, according to problem setting 'how much is a customer need important?,' targeting customers who care good quality of camera can be proposed as an alternative solution of (B). Figure 5- ② shows an alternative of detailing customer needs that are proposed by problem setting 'what are sublevel customer needs?.' A new customer need, 'barcode reader,' is added. A relative worth of a camera lens is increased by this alternative (see Figure 6-②).

This design process is automatically captured as a byproduct of a designer's operations on the system. Figure 7 shows a part of the captured design process in argumentation level. Issue nodes (blue node with question mark) and position nodes (red node with exclamation mark) are automatically generated. A gray node is a discarded alternative position. An argument node (green node) is added by a designer to explain the branch of position nodes. Figure 7 shows that two positions are suggested for an issue of detailing a customer need, and that one position that adopts bar-code reader is adopted. Since all operations and all design states are recorded by TMS, a designer can review discarded position at any time if he/she wants to review it again, and he/she can go back to any former design state.

4.3 Discussion

This example assumed that a product is aimed to Japanese market and that it should equip with many value-adds. As a result, it is required that the integrity of wide variety of value-adds is compromised with total cost. Through several experiments on the case studies with this example problem, it is ascertained that the system can automatically and smoothly record designer's design process that includes operations, alternatives and argumentation. Such functionality facilitates designer's reflective refinement of alternatives over the prototype system through making full use of related methodologies.

This example also shows that the contextual structure of QFD is appropriate as a base of building ontology for the knowledge-based design support system. An ontology of value graph, function-

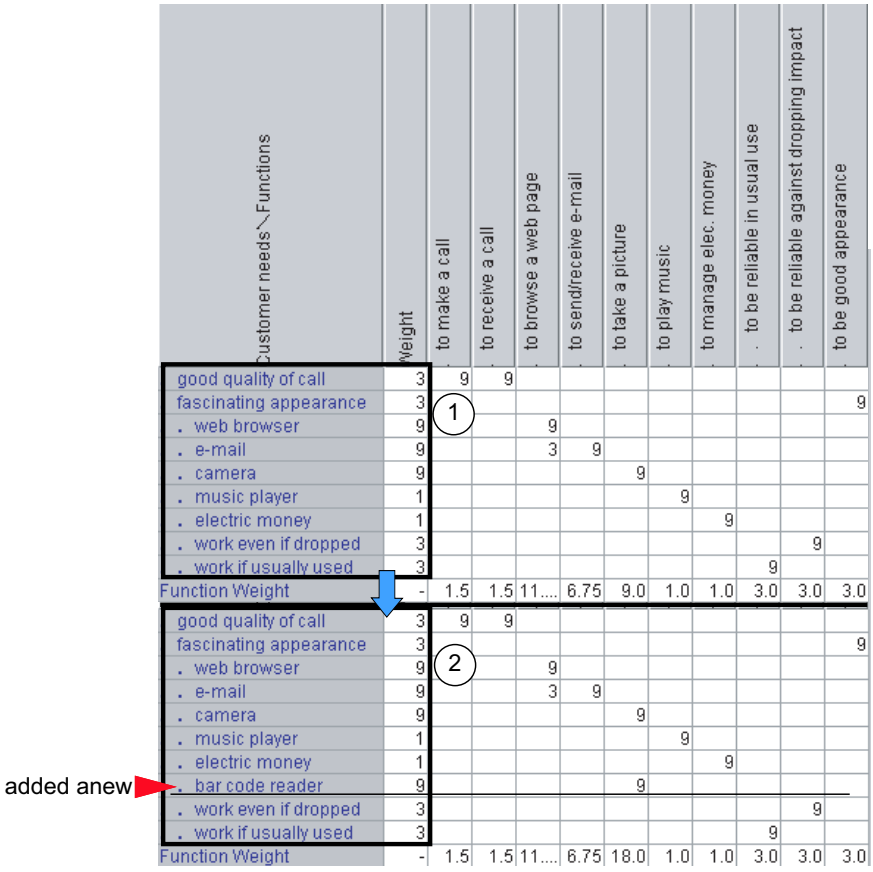

Figure 5. Alternatives of detailing customer needs

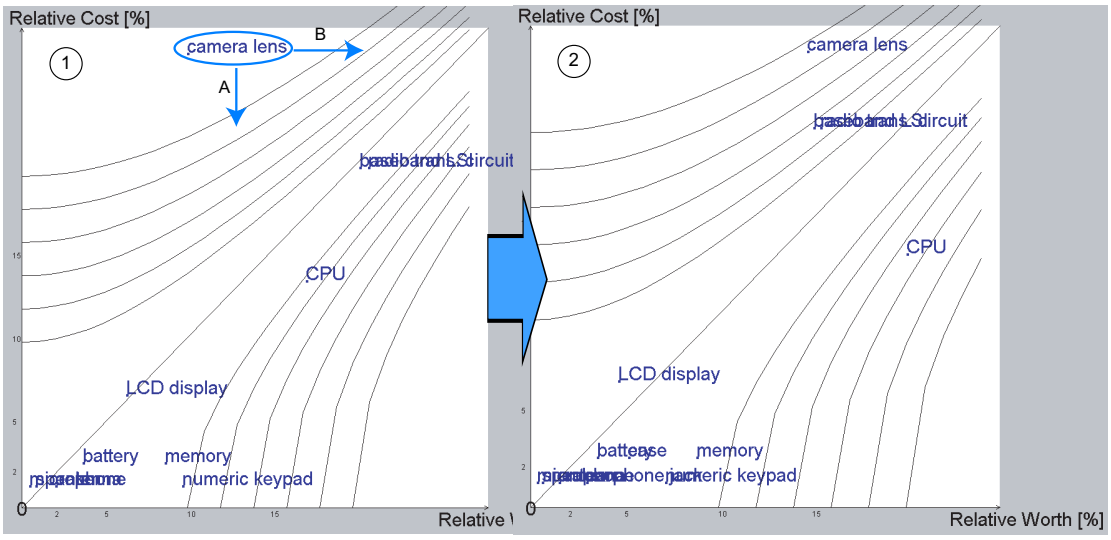

Figure 6. Revision of cost-worth balance

structure mapping and cost-worth graph is defined with small modification. This kind of expansions can be expected to be feasible for other DFX methodologies. This expansion is our future work.

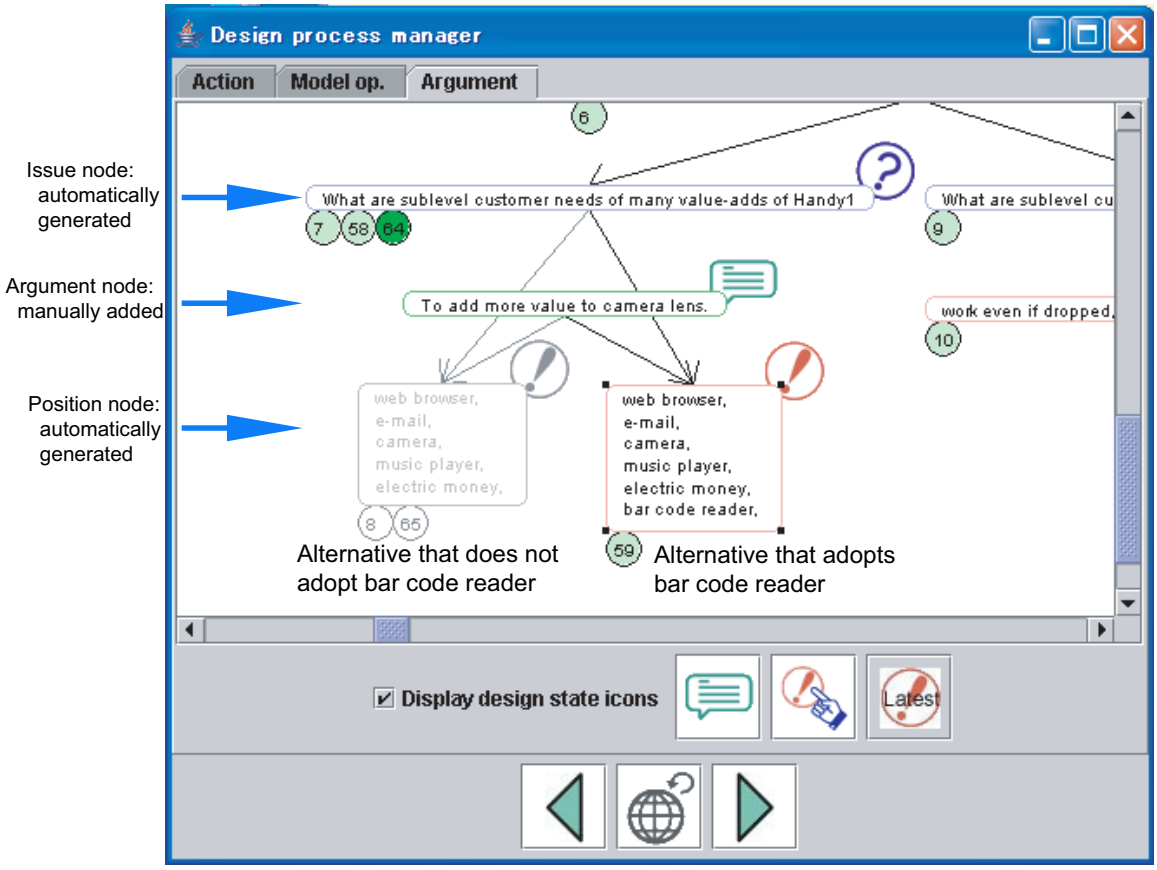

Figure 7. A part of design process

5 CONCLUSIONS

This research stated that more useful ontology can be deduced from the patterns embedded in DFX methodologies and that such deduction expands the performance and potential of knowledge-based design support systems. This paper demonstrated an experimental integration of the system of ontology extracted from QFD and related design methodologies to a prototype of a knowledge-based design support system. Since discussion and implementation are still limited within a partial range of various design activities and application is still kept in a small-scale problem as compared with practice, further research and development remains as future work.

REFERENCES

[1] Fujita, K. and Nishikawa, T. Value-Adds Assessment Method for Product Deployment across Life Stages through Quality Function Deployment, *Proceedings of 13th International Conference on Engineering Design — ICED 01, Design Methods for Performance and Sustainability*, 2001, pp. 405-412

[2] Nomaguchi, Y., Ohnuma, A. and Fujita, K. Design Rationale Acquisition in Conceptual Design by Hierarchical Integration of Action, Model and Argumentation," *In Proceedings of the 2004 ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2004, CIE-57681.

[3] Nomaguchi, Y., Taguchi, T. and Fujita, K. Knowledge Model for Managing Product Variety and its Reflective Design Process, *Proceedings of ASME 2006 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, 2006, DETC2006-99360.

[4] Dixon, J. R. On Research Methodology Towards a Scientific Theory of Engineering Design, *AI EDAM*, 1(3), 1987, pp. 145.157.

[5] Clausing, D. *Total Quality Development. A Step-By-Step Guide to World-Class Concurrent Engineering*, 1994, (ASME Press).

[6] Ohfuji, T., Ono, M. and Akao, Y. *Quality Function Deployment*, 1990, (Union of Japanese Scientists and Engineers). (in Japanese)
[7] Gruber, T. A Translation Approach to Portable Ontology Specifications, *Proceedings of JKAW'92*, 1992, pp. 89-108.
[8] Kitamura, Y., Washio, N., Koji, Y., Sasajima, M., Takafuji, S. and Mizoguchi, R. An Ontology-based Annotation framework for representing the functionality of engineering devices, *Proceedings of ASME 2006 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, 2006, DETC2006-99131.
[9] Doyle, J. A Truth Maintenance System, *Artificial Intelligence*, 12(3), 1979, pp. 231-272.
[10] Conklin, J., and Begeman, M. L. gIBIS: A Hypertext Tool for Exploratory Policy Discussion, *ACM Transactions on Office Information Systems*, 6(4), 1988, pp. 303-331.
[11] Tanaka, M. Cost Planning and Control Systems in the Design Phase of a New Product, *Japanese Management Accounting: A World Class Approach to Profit Management*, 1989, (Productivity Press).

Contact: Y. Nomaguchi
Osaka University
Department of Mechanical Engineering
2-1 Yamadaoka,
Suita, Osaka, 565-0871
Japan
Phone: +81-6-6879-7324
Fax: +81-6-6879-7325
e-mail: noma@mech.eng.osaka-u.ac.jp
URL: http://syd.mech.eng.osaka-u.ac.jp/˜noma/