Proposal of Knowledge Model for Designing Product Architecture and Product Family

Yutaka Nomaguchi
Dept. of Mechanical Engineering,
Osaka University.
2-1 Yamadaoka, Suita, Osaka
Japan, 565-0871
+81-6-6879-7324
noma@mech.eng.osaka-u.ac.jp

Tomohiro Taguchi
Dept. of Mechanical Engineering,
Osaka University.
taguchi@syd.mech.eng.osaka-u.ac.jp

Kikuo Fujita
Dept. of Mechanical Engineering,
Osaka University.
fujita@mech.eng.osaka-u.ac.jp

Abstract
In order to keep global competences, recent manufacturers have been utilizing product families to diversify and enhance the product performance by simultaneously designing multiple products under commonalization and standardization. Design information of product architecture and family is inevitably more complicated and numerous than that of a single product. Thus, more sophisticated computer-based support system is required for product architecture and family design. This paper proposes a knowledge model for product architecture and family design support system. This research aims to tackle three problems which should be overcome when product family are modeled in the computer system; design repository without data redundancy and incorrectness, knowledge acquisition without forcing the additional effort on the designer, and integration of descriptive models to support early stage of design. The ontology of the knowledge model is defined as a foundation of resolving there problems. This paper is concluded with discussion of future works.

Keywords: Product architecture, product family, knowledge model, knowledge acquisition, design support, ontology

1. Introduction
As manufactures adapt to a recent highly competitive global marketplace, the need to integrity of varying high-end performances and cost reduction of product will increase. They are utilizing product families to diversify and enhance the product performance by simultaneously designing multiple products under commonalization and standardization [6]. The key to rationale and successful product family design is product architecture [15]. This research perceives product architecture as the schema which connotes each product variant by defining possible structures and possible attribute values. Product architecture is written by three aspects; customer needs, functions and entities such as components and parts. Good product architecture yields a successful product family. Platform design of automobiles, i.e., Volkswagen's A-Platform [17], is a representative example, which shows the power of established product architecture.

Design information of product architecture and family is inevitably more complicated and numerous than that of a single product. Thus, more sophisticated computer-based support system is required for product architecture and family design. One of major problems to be overcome when product family are modeled in the computer system, is the product description of multiple aspects on a product family without data redundancy and incorrectness even in large-scaled design repository [8]. The second problem is to acquire design knowledge, which is common problem for knowledge management systems [2]. The third problem is to integrate descriptive models in order to support design. Even though many research groups have been tackling the computational searching techniques to optimize product architecture and family design [14], the stage before the optimization should be supported by descriptive methodology. From the different points of view, different descriptive models have been developed. A designer has to integrate various models and switch them in order to reflectively evaluate the design.

This paper proposes the knowledge model of product architecture and family which solves the above mentioned three problems. For design repository without data redundancy and incorrectness, this research defines ontology of product architecture and family design. For knowledge acquisition, the ontology is defined to represent not only concepts of a result of design but also concepts of any state of design. For integration of descriptive models to support early stage of design process, the concepts of the descriptive models are defined as an extended part of the ontology. This knowledge model serves as a core of a knowledge management system which we have been developing to support complicated and iterative design process as product architecture and family design [12].

2. Requirements of Knowledge Model

2.1 Knowledge Acquisition by Iteration Design Process
In general, human's problem solving including design is done through iteration process [13]. The process of product architecture and family design also includes iteration process to reflectively consider problems such as '*which product should be in family?*,' '*what product architecture should be?*,' '*which component should be commonalized?*' and so on. To support product architecture and family design, some descriptive methods have been developed, e.g. the market segmentation grid [9] which is to plan family deployment strategy, GVI/CI [7] which is the QFD based indices to evaluate product architecture from the viewpoint of robustness against market changes. Although a designer cannot expect these methods to automatically search solutions like optimization

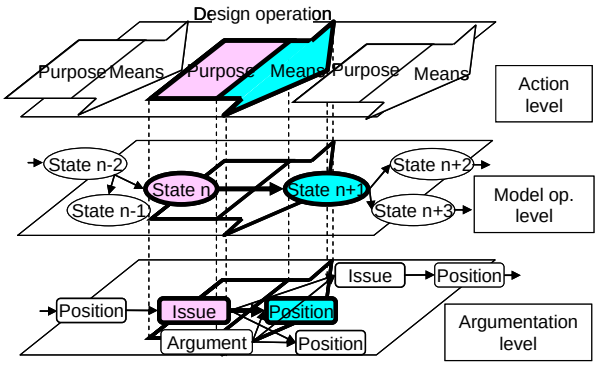

Figure 1. Three levels of design process

methodology, it is valuable for a designer by using the descriptive models to document his/her idea and its problem, then to consider various ideas reflectively.

Knowledge takes a crucial role to do effective trials in the iteration process as much as possible and to aim at the goal. Some of the design knowledge is based on the already-systematized theory such as a physical phenomenon. Besides, a designer acquires many of knowledge through experiences of iterations as know-how. Acquisition of latter design knowledge without forcing additional effort on a designer is a key to success in knowledge management.

2.2 Toward Knowledge Management Approach

We have been researching a computer-aided design system to acquire a designer's knowledge, which was used in iterative design process, as a by-product of design without forcing additional effort on the designer. This system captures design process at three levels as shown in Figure 1; log of a designer's action, state transition of design information and argumentation structure of a designer's thinking process. The second and the third level represent the iteration process explicitly, and they are composed automatically based on the first level. In this framework, a knowledge model of design information takes important role. The log of a designer's action is captured as the sequence of operations to the knowledge model. The state transition of design information is captured as the state transition of the knowledge model. The argumentation structure of a designer's thinking process is captured as a structure of purpose and mean of the operation. The detailed algorithm can be seen in [12].

From the view point of this framework, the knowledge model should meet the following requirements. Firstly, the knowledge model should record not only the result of design but also the state of design at a certain moment of design process. Secondly, the knowledge model should integrate various design models to represent design information from various aspects such as customer needs, functionality, structure and cost. Lastly, the operations to the knowledge model should be defined by considering its purpose and means, which corresponds to an issue and a position of argumentation, respectively. This paper mainly discusses the first and the second requirements. The last requirement is not discussed in this paper.

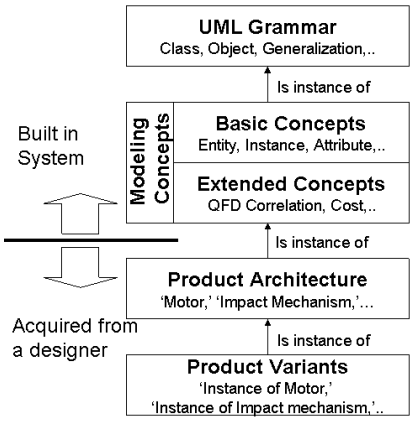

Figure 2. Ontology layers for product architecture and family

2.3 Modeling Ontology for Product Architecture and family Design

Ontology is a basis of knowledge modeling. Note that we interpret ontology in the sense; a theory about the sorts of concepts, properties of concepts, and relations between concepts that are possible in a specified domain of knowledge. Figure 2 shows the ontology layers of the knowledge model defined in this research.

The UML grammar is the first layer because all concepts of the knowledge model are defined in the UML format. The second layer is the modeling concepts. This layer is categorized into two types; basic concepts and extended concepts. The basic concept is the concept to represent generic model which is independent from any specific descriptive design model. The extended concept is the concept to represent specific descriptive design models.

The above mentioned layers are built in the computer. Product architecture and product variants are modeled as an instance of the built-in concepts. By using the modeling concepts, a designer can represent his/her knowledge of product architecture and family while using descriptive models.

3. Modeling Product Architecture and Product Family

This section explains the modeling concepts of the knowledge model, which is proposed for product architecture and family design. Firstly, the illustrative example of screw driver design is introduced. The example product architecture and deployed product variants are represented by the knowledge model, which is described in the UML format. Secondly, concepts of modeling ontology are introduced. Finally, the extended concepts to integrate descriptive models are introduced.

3.1 Illustrative Example of Product Architecture and Family Deployment

This subsection introduces the example used throughout this paper, in advance. The design of the electric screw driver product family, and the attributes which are used to specify a particular variant, are shown in Figure 3 by UML. Rectangle nodes are objects of classes which are defined in Figure 5. Figure 3 shows a part of entity aspect structure, but it would be enough to understand the overview of proposed knowledge model.

It can be seen from Figure 3 that an electric screw driver has a motor, a gear box, a battery and an impact mechanism. There are two possible hierarchical structures of an electric screw driver. One is a *normal type* hierarchy, which has a motor, a gear box and a battery as a subcomponent of an electric screwdriver. The other is an *impact type* hierarchy, which has an impact mechanism in addition to the normal type structure.

The attributes which are used by a designer to specify a screw driver are *screwing torque (T)* which is a functional attribute, *impact factor (F)* of the impact mechanism, *reduction ratio (i)* of the gear box, *rating torque (Tm)* of the motor, *rating round per minutes* of the motor and *capacity* of the battery. A value of screwing torque, *T*, is calculated by three variables; *F*, *i* and *Tm*. Module alternatives are designated to each component. A value of the attribute of each component can be selected from the possible values designated by module alternatives. The combination of hierarchy types and attribute values results in 8 possible variants of the screw driver family. However constraints reduce the potential variety. In this example, there is a constraint to a value of screwing torque; $T > 50$. Because of this constraint, 2 impact type variants are possible. Figure 4 shows one of the possible variants.

3.2 Basic Concepts

The following concepts are the basic parts of the ontology employed for the knowledge model of product architecture and family. The concepts in the ontology form a taxonomical hierarchy as shown in Figure 5. The concept taxonomy, properties of concepts and associations between concepts are represented in the UML format. The first level of the taxonomy consists of four concepts; *product architecture concept*, *product variants concept*, *commonalization* and *attribute value concept*.

3.2.1 Product architecture

Product architecture is a concept representing the schema which connotes the all possible structure and the possible value of the attribute of each product variant. This concept has the following sub concepts; *qualitative architecture*, *quantitative architecture* and *module*.

Qualitative architecture is a concept representing qualitative view point of product architecture. This has the following sub concepts.

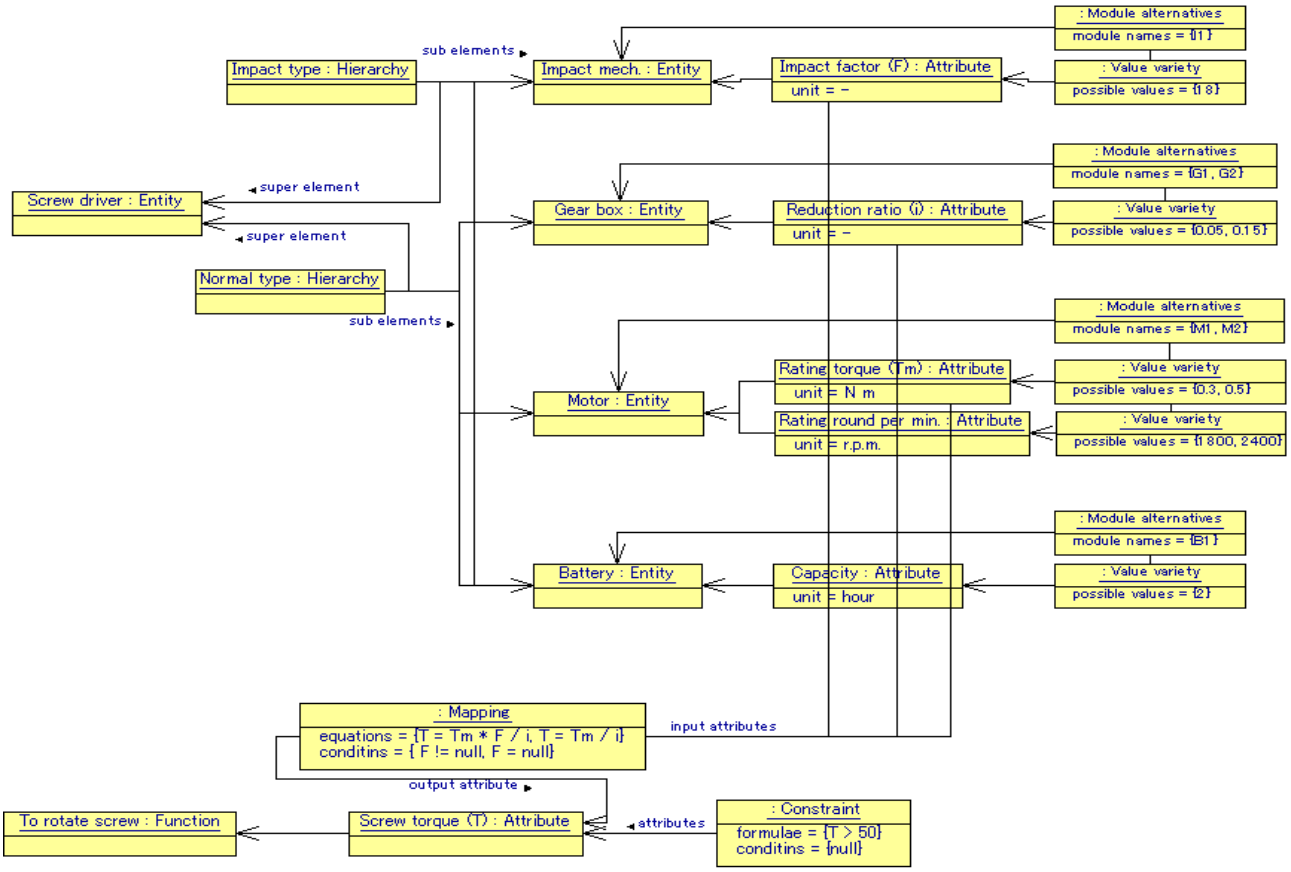

Figure 3. Example of product architecture of electric screwdriver

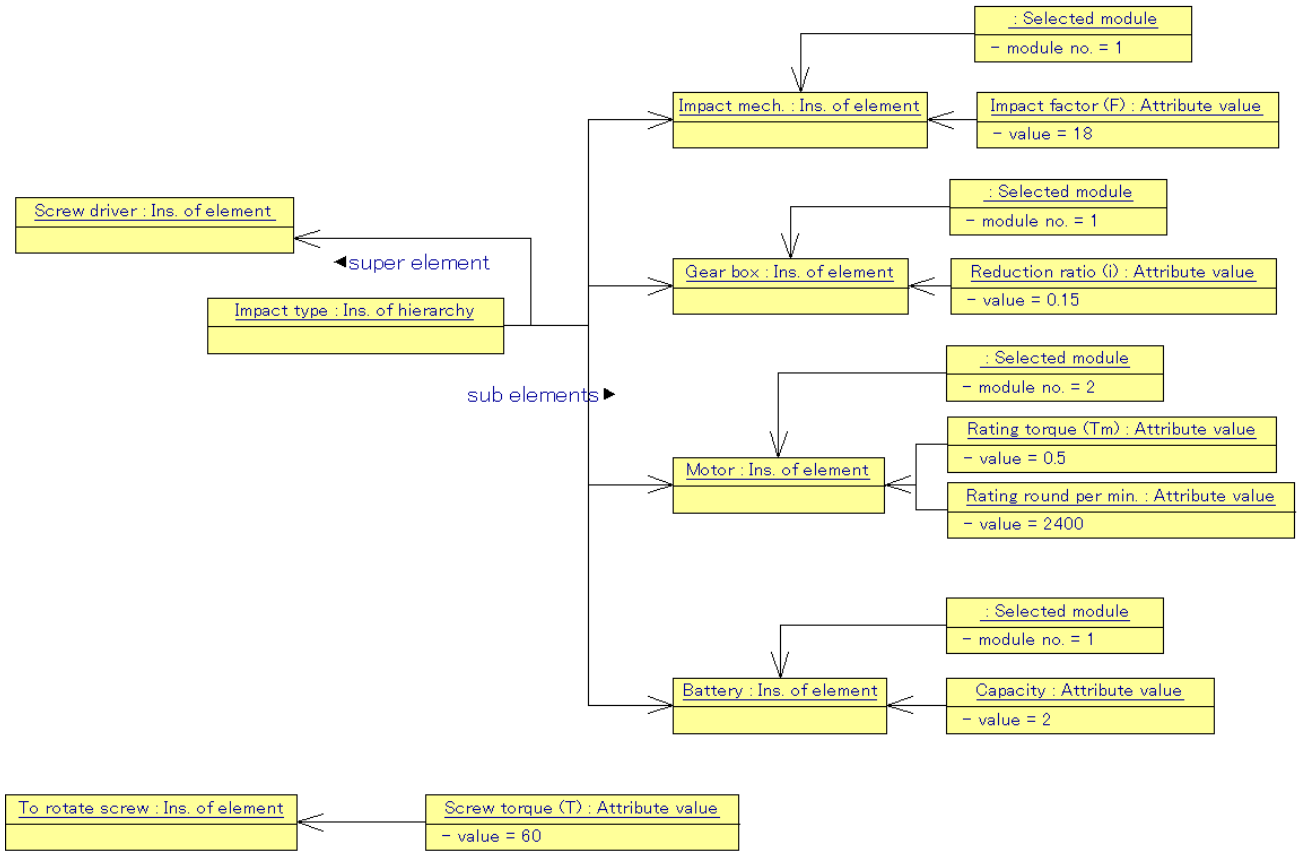

Figure 4. Example of screw driver variant

Figure 6 shows the associations defined among the qualitative architecture concepts.

Element is a concept that constructs a product architecture. This is further categorized into three concepts; *customer need*, *function* and *entity*. For example, an element of customer need of screw driver is 'easy to operate,' 'powerful screwing,' etc., an element of function is 'to generate screwing torque,' etc., and an element of entity is 'motor,' 'battery,' etc.

Hierarchy is a concept that represents a possible hierarchical relationship between elements. A hierarchy node has an association to an element, which is super level of the hierarchy, and elements, which are sub level of the hierarchy. In the screw driver example, a hierarchy node 'impact type' represents that 'electric screw driver' has four sub entities; 'impact mechanism,' 'battery,' 'gear box' and 'motor.'

Relation is a concept that represents a relation between elements. For example, 'Function-Structure relation' is defined among a function 'to generate screwing torque' and entities 'motor,' 'gear box,' 'impact mechanism.'

Quantitative architecture is a concept representing a quantitative constraint which deduces the possible values of each product variant. This has following three sub concepts. Figure 7 shows the associations defined among quantitative architecture concepts.

Attribute is a concept that represents a character of an element or a relation. An attribute node has an association to an element or a relation. In the example of Figure 3, three attributes are defined as the attribute of entity 'motor'; 'rating torque,' 'rating round per minutes' and 'weight.' An attribute node has a unique *attribute value*.

Mapping is a concept that represents existence of a numerical function which determines a value of an attribute. A mapping node has an association to one output attribute and plural input attributes. A list of pairs of an equation and its condition is defined as a property of a mapping node. In Figure 3, a value of attribute 'screwing torque (*T*)' is determined by a mapping equation '$T = Tm * F / i$' and values of input attributes; 'rating torque (*Tm*),' 'impact factor (*F*)' and 'reduction ratio (*i*)' when the attribute 'impact factor' exists. In case of normal type screw driver, a value of *T* is determined by an equation '$T = Tm / i$' because the attribute 'impact factor' does not exist.

Constraint is a concept that represents existence of a constraint among attribute values. A constraint node has an association to plural attributes. A list of pairs of a formula and its condition are defined as a property of a constraint node. In Figure 3, a formula '$T>50$' is defined as a constraint of attribute 'screwing torque (*T*)'.

Module is a concept representing available modules of the entity. This has the following two sub concepts. Figure 8 shows associations defined among module concepts.

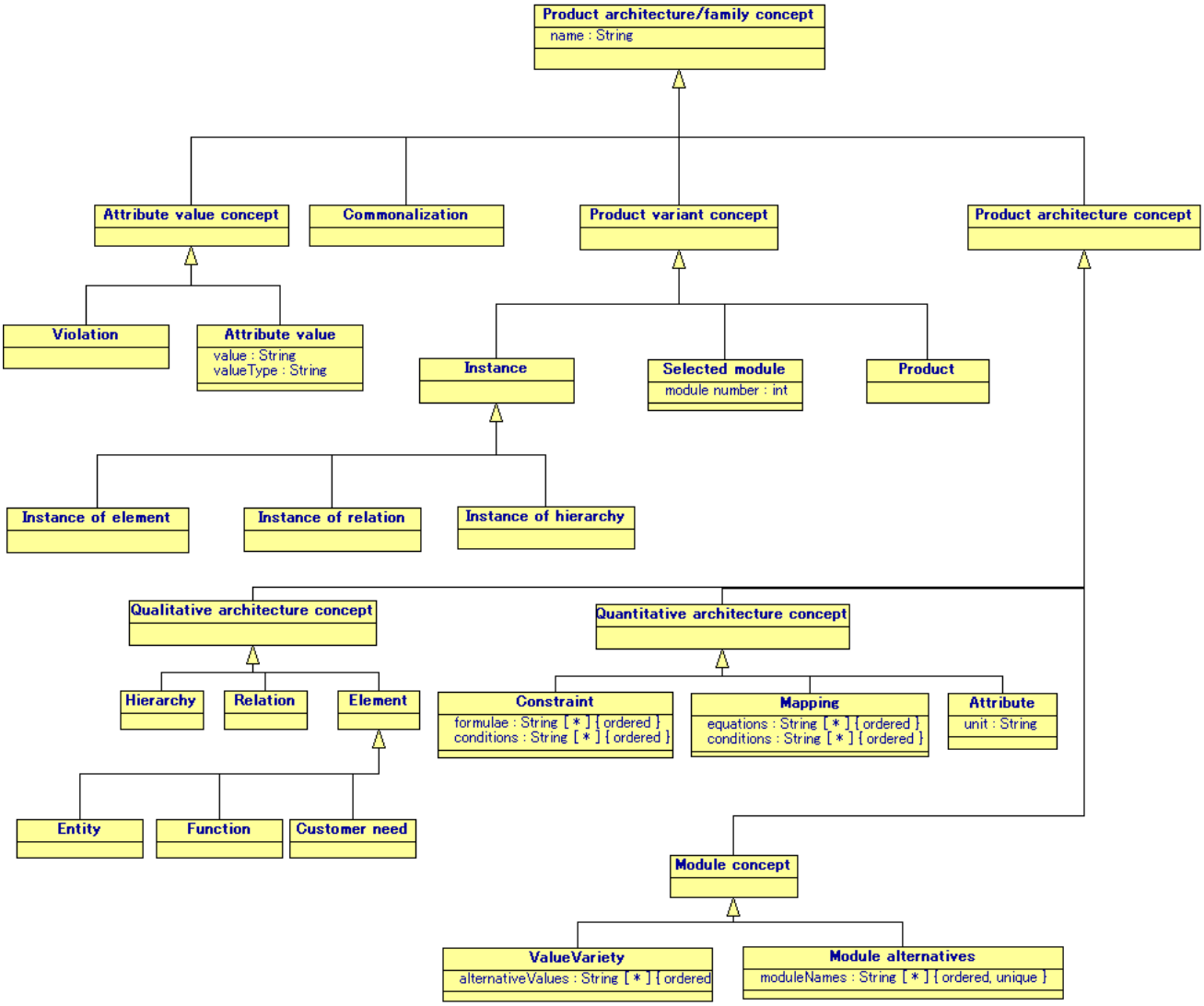

Figure 5. Taxonomy of modeling concepts

Module alternative is a concept which represents existence of available modules for the entity. A module alternative node has an association to one entity node. A list of module names is defined as property of the module alternative node.

Value variety is a concept which represents attribute values of each module which is defined by a module alternative node. A value variety node has an association to one attribute node and one module alternative node. A list of possible values is defined as property of the value variety node.

3.2.2 Product variant concept

Product variant concept represents a qualitative structure of a product in the product family. Figure 9 shows associations defined among module concepts. This concept has the following three sub concepts.

Product is a concept which represents a product variant in the product family.

Because the product architecture connotes the all possible structure, a qualitative structure of a product variant is represented as an *instance* of concepts of product architecture. Note that this research uses instance as the similar sense of object-oriented technology; the substance of information of the class that defines the properties and methods of the object. In this sense, qualitative architecture concepts correspond to the class of the instances.

Instance is a concept which represents that a product has an instance of product architecture concept. Three sub concepts corresponding to qualitative architecture concepts are defined; *instance of element*, *instance of relation* and *instance of hierarchy*. Each of an instance node has an association to a product node and a 'class' concept of product architecture.

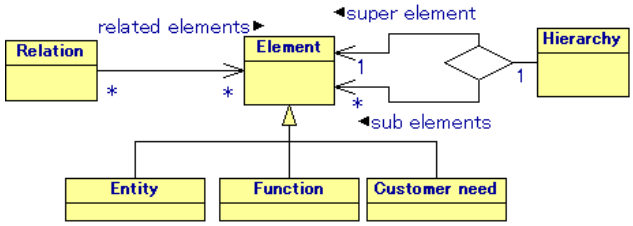

Figure 6. Associations of qualitative architecture concepts

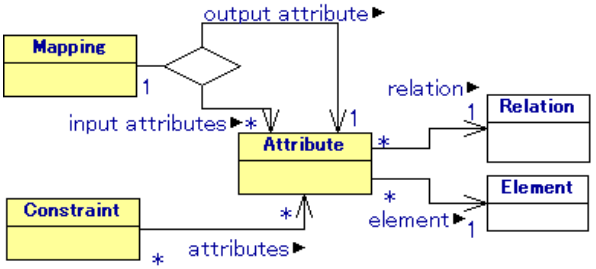

Figure 7. Associations of quantitative architecture concepts

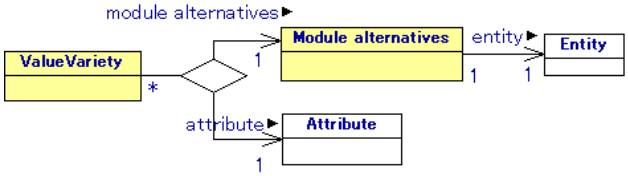

Figure 8. Associations of module concepts

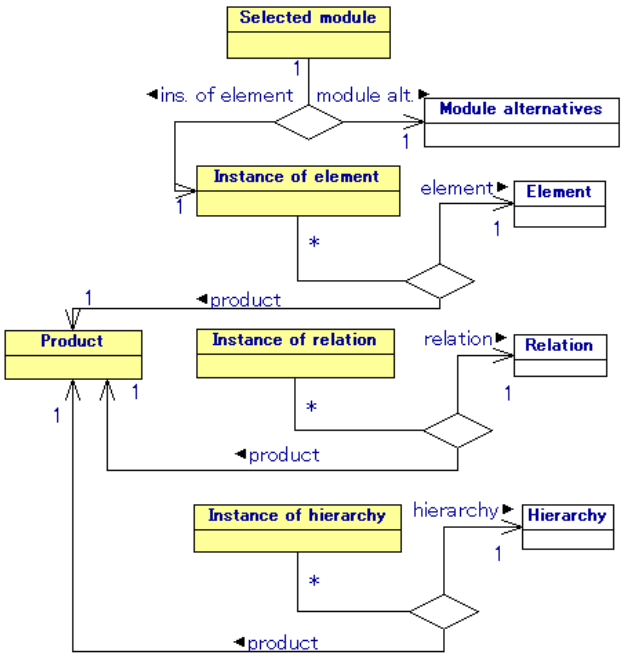

Figure 9. Associations of product variant concepts

The existence of instance node means that the product inherits the concept of product architecture, and that the product also inherits the attributes, the mappings, the constraints, the module alternatives and the value varieties which are associated to the class concept. This mechanism serves to reduce the amount of information required to represent product family.

Selected module is a concept which represents the instance module selected for the instance of entity. A selected module node has a module number as property. A selected module node has an association to a module alternative and an instance of element.

3.2.3 Commonalization

Commonalization of components or parts is defined for instances of product variants. Figure 10 shows the association related to commonalization concept.

Commonalization is a concept which represents the intention of a designer to commonalize components or parts of plural product variants. Commonalization node has an association to instance nodes of element.

3.2.4 Attribute value concept

Attribute value concept has two sub concepts; attribute value and violation. Figure 11 shows the associations related to attribute value concepts.

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

There are four association types for an attribute value node as shown in Figure 10. The different association means the different rationale of determining the attribute value.

Association to an attribute and an instance is a usual type of association. This represents that the attribute value node is determined as an input of design.

Association to an attribute, an instance and a commonalization is an association which presents that the attribute value node is determined as a result of commonalization.

Association to an attribute, an instance, a mapping and attribute values is an association which represents that the attribute value node is determined as a result of calculating the mapping equation and input attribute values.

Association to an attribute, an instance and a value variety is an association which represents that the attribute value node is determined as a result of selecting a module.

'An instance' in above definition can be omitted when the attribute value is common to all product variants.

After the attribute value is determined anyway, it should be checked whether it satisfies the constraint or not. If not, violation

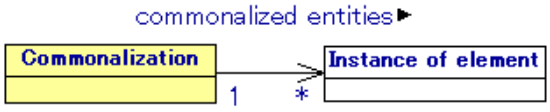

Figure 10. Association of commonalization

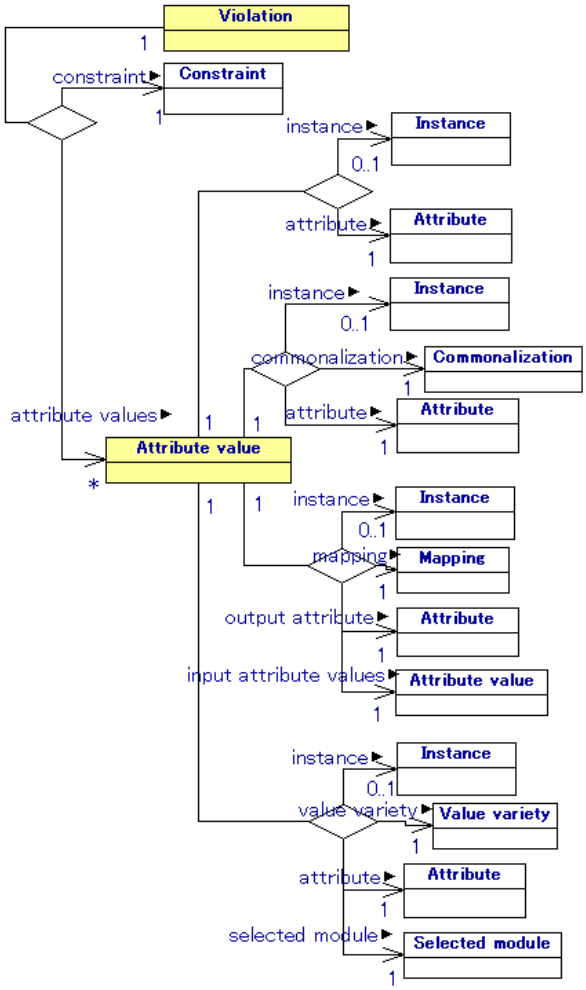

Figure 11. Association of attribute value concepts

node is created to indicate a designer.

Violation is a concept which represents the attribute value violates the constraint. A violation node has an association to attribute values and a constraint.

The violation should be resolved by altering input attribute values or by relaxing the constraint. Note that the knowledge model does not take charge of resolving the constraint violation. The knowledge model does just describe the state of design which includes the constraint violation.

3.3 Integrating Models

This research employs the following descriptive models; value graphs, function-structure mapping, QFD and cost/worth graph. The standard design process by using these models is as follows. A designer firstly uses the value graph and the function-structure mapping in order to deploy customer needs, functions and components of product architecture. Then, varied weights of customer needs are set for each product variant, and they are deployed to worth of components. Finally, a designer can check the balance of relative worth and relative cost of each component. Note that this design process includes iterations in order to reflectively consider plural ideas.

A concepts necessary to express these models is defined as a subclass of the basic concepts. There is a possibility that other concepts in addition to the extended concepts, which is explained in this subsection, are defined if necessary when the other model is integrated to the knowledge model.

3.3.1 Value graph, Function-Structure mapping

The value graph describes the deployment of customer need 'good product' into the detail customer needs. The function-structure mapping describes the deployment of function and structure of the product, and relationships between functions and components. The concept of *customer need*, *function*, *entity* and *hierarchy* is used to represent the deployment hierarchy. The additional concept, *F-S relation*, is defined as sub class of *relation* concept in order to represent the relationship between a function and a component.

3.3.2 QFD

QFD describes correlation numbers between customer needs and functions, and ones between functions and components. These correlations numbers are used to deploy the weights of customer needs to the weights of components by simple matrix calculation. The following additional concepts are introduced to represent QFD.

C-F relation and *F-S 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 a component.

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

QFD correlation is a sub class concept of attribute. It is used to represent the correlation number of a C-F relation or an F-S relation.

3.3.3 Cost/Worth Graph

The cost/worth graph describes the balance of relative worth and relative cost for each component. The following additional concepts are defined to represent cost/worth graph.

Relative worth is a sub class of attribute. This is calculated for each product variant by regularization of weights of components which are calculated by QFD.

Cost is a sub class concept of attribute. It is used to represent cost of each component of each variant.

Relative cost is a sub class concept of attribute. It is used to represent relative cost of each component. Its value is calculated for each product variant by regularization of cost of components.

3.4 Plan of Implementation

Now we are going to implement the design support system based on the proposed knowledge model. This system is developed in JAVA programming language (jdk 1.4.1) on Windows XP. Figure 12 shows the architecture of the system. The rectangles with bold line are the parts explained in this article.

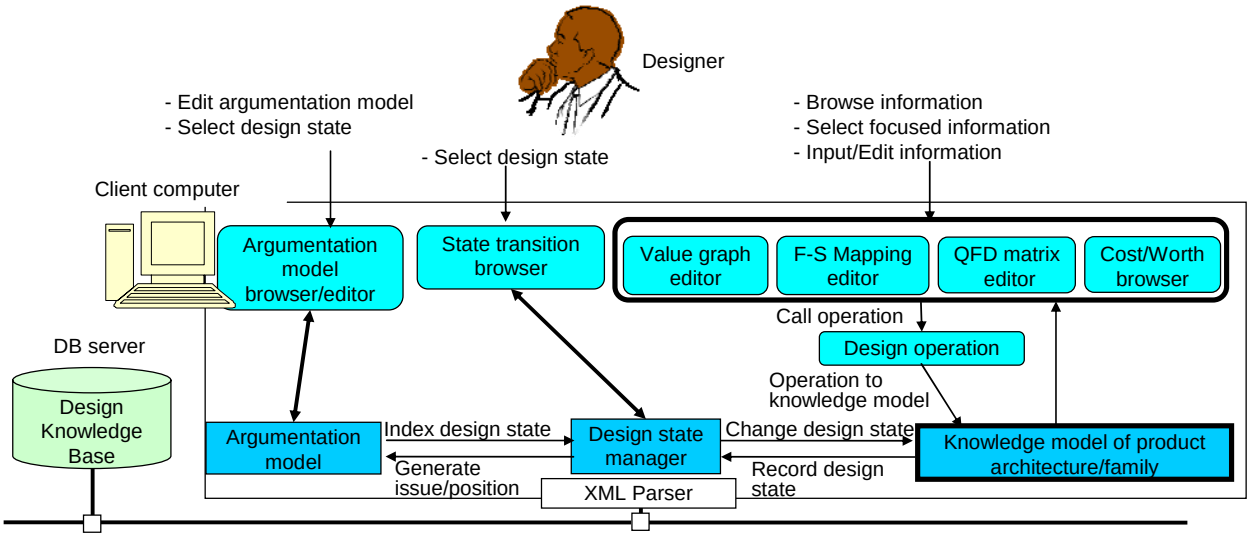

Figure 12. System architecture

4. Related Works

Knowledge modeling methods provide the necessary framework for the representation of design information in all design domains. Many research groups have conducted this topic. In the domain of product architecture and family design, knowledge modeling of design is becoming hot topic although its ability to model is still very limited [14]. Table 1 summarizes the six works related to this research and compares their research aims and modeling ontology with ours.

The research group of NIST (National Institute of Standard and Technology, USA.) has been developing the knowledge model called Core Product Model (CPM) toward large-scaled repository system of design rationale [5]. Wang extended CPM to represent the evolution of product families and of the rationale of the changes involved [16]. They focus on design repository of rationale of product family deployment so that the modeling ontology includes concepts, such as *version* and *series* of product variants. However, they don't care about knowledge acquisition so much. The concepts about available module alternatives and commonalization of components are not considered, which are mainly used at the process of designing rather than the result of design.

A configuration framework for mass customization of products that employs the UML is introduced by Felfernig [4]. They focus on knowledge acquisition and maintaining knowledge bases. The knowledge model written in the UML format is automatically translated into an executable logic representation in order to employ model-based diagnosis techniques for debugging faulty configuration knowledge bases, detecting infeasible requirements and for reconfiguring old configurations. Their knowledge model represents qualitative architecture and module alternatives, but it

Table 1. Comparison of related works

		Nomaguchi et al. 2006	Wang et al. 2003	Felfering et al. 2001	McKay et al. 1996	Claesson et al. 2001	Mortensen et al. 2005	Nanda et al. 2005
Research aim	Design repository	X	XX	X	-	–	-	x
	Knowledge acquisition	X	-	X	X	x	x	x
	Design support by descriptive model	X	-	-	x	x	x	X
Modeling ontology	Qualitative architecture	X	X	X	X	X	X	X
	Quantitative architecture	X	x	-	X	-	-	-
	Module alternative	X	-	X	-	x	X	-
	Product variant	X	X	X	X	X	X	X
	Commonalization	X	-	-	-	-	-	-
	Attribute value	X	X	-	x	-	-	-
	Descriptive model	X	-	-	-	-	-	x
	Others		Product evolution (version, series, ..)					

XX; more considered than this research, X; considered as well as this research, x; considered but a little, -; not considered.

does not care of qualitative architecture.

To reduce data redundancy when modeling families of products, the Generic Bill-of-Material (GBOM) concept developed at the Eindhoven University of Technology allows all variants of a product family to be specified only once [3]. McKay combined the GBOM concept with product modeling concepts and software to reduce data redundancy when considering multiple views, e.g., sales, manufacturing and assembly [8]. McKay's knowledge model is useful for design repository and knowledge acquisition because it can represent qualitative architecture, quantitative architecture and module alternatives. His knowledge model does not cares about integrating descriptive models in order to support design. .

Claesson uses function-means-trees to create configurable components that represent a parameterized set of design solutions [1]. This knowledge model was deployed at Saab automobile to help control product variety. Mortensen proposed PFH (Product Family Hierarchy) diagram which visualizes the qualitative architecture and the key component which could be a key of commonalization among product family [10]. PFH is deployed at the Danish company Martin Professional A/S. Both knowledge models employ simple ontology of qualitative architecture and module alternatives in order to actually be used in companies. However, a designer has to use another tools to check the limitation of designed architecture, because the knowledge model represents neither descriptive models nor quantitative architecture.

Nanda proposed a knowledge model based on OWL (Web Ontology Language) in order to capture, share, and organize product design contents concepts and contexts across different phases of the product design process [11]. This knowledge model represents qualitative architecture of three aspects; customer needs functions and components. It also represents relationships among them by integrating QFD into the knowledge mode.

5. Conclusion

This paper reports a knowledge model for a knowledge management oriented support system for product architecture and family design. The ontology of the knowledge model is designed in order to capture any design state of product architecture and family design, and to integrate descriptive design models. This paper also shows our plan of implementing knowledge management system based on the knowledge model. The ability of the knowledge model will be evaluated by performing the illustrative example design on the system.

Acknowledgements

This research has been carried out partially at the Strategic Research Base, Handai (Osaka University) Frontier Research Center supported by the Japanese Government's Special Coordination Fund for Promoting Science and Technology.

References

[1] Claesson, A., Johannesson, H. and Gedell, S. Platform Product Development: Product Model a System Structure Composed of Configurable Components, In *Proceedings of the DETC'01 ASME 2001 Design Engineering Technical Conferences and Computer and Information in Engineering Conference*, DTM-21714, 2001.

[2] Dieng, R. Knowledge Management and the Internet, *IEEE Intelligent Systems*, Vol. 15, No. 3, pp. 14-17, 2000.

[3] Erens, F. J. and Hegge, H. M. H. Manufacturing and Sales Co-ordination for Product Variety, *International Journal of Production Economics*, Vol. 37, No. 1, pp. 83-99, 1994.

[4] Felfernig, A., Friedrich, G. and Jannach, D. Conceptual Modeling for Configuration of Mass-Customizable Products, *Artificial Intelligence in Engineering*, Vol. 15, pp. 165-176, 2001.

[5] Fenves, S. *A Core Product Model for Representing Design Information*, NISTIR 6736, NIST, Gaithersburg, MD, 2001.

[6] Fujita, K. Product Variety Optimization under Modular Architecture, *Computer-Aided Design*, Vol. 34, No. 12, pp. 953-965, 2002.

[7] Martin, M. V. and Ishii, K. Design for Variety: Developing Standardized and Modularized Product Platform Architectures, *Research in Engineering Design*, Vol. 13, No. 4, pp. 213-235, 2002.

[8] McKay, A., Erens, F. and Bloor, M. S. Relating Product Definition and Product Variety, *Research in Engineering Design*, Vol. 2, pp. 63-80, 1996.

[9] Meyer, M. H. Revitalize Your Product Lines Through Continuous Platform Renewal, *Research Technology Management*, Vol. 40, No. 2, pp. 17-28, 1997.

[10] Mortensen, N. H., Munk, L. and Fiil-Nielsen, O. Preparing for a Product Platform – Product Family Hierarchy Procedure -, In *Proceedings of International Conference on Engineering Design (ICED 05)*, 296.45, 2005.

[11] Nanda, J., Simpson, T. W., Shooter, S. B. and Stone, R. B, A Unified Information Model for Product Family Design Management, In *Proceedings of the DETC'05 ASME 2005 Design Engineering Technical Conferences and Computer and Information in Engineering Conference*, 84869, 2005.

[12] 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 & Computers and Information in Engineering Conference*, DETC2004/CIE-57681, 2004.

[13] Simon, H. A. *The Sciences of the Artificial*, The MIT Press, 1969.

[14] Simpson, T. W. Product Platform Design and Customization: Status and Promise, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 18, No. 1, pp. 3-20, 2004.

[15] Ulrich, K. The role of product architecture in the manufacturing firm, *Research Policy*, Vol. 24, No. 3, pp. 419-440, 1995.

[16] Wang, F., Fenves, S. J., Sudarsan, R. and Sriram, Ram. D. Towards Modeling the Evolution of Product Families, In *Proceedings of the DETC'03 ASME 2003 Design Engineering Technical Conferences and Computer and Information in Engineering Conference*, CIE-48216, 2003.

[17] Wilhelm, B. Platform and Modular Concepts at Volkswagen-Their Effect on the Assembly Process, *Transforming Automobile Assembly: Experience in Automation and Work Organization*, K. Shimokawa, U. J¨urgens and T. Fujimoto, eds., Springer-Verlag, New York, pp. 146-156, 1997.