INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 05 MELBOURNE, AUGUST 15-18, 2005

PRODUCT ARCHITECTURE DESIGN PROCESS FOR MODEL-BASED KNOWLEDGE MANAGEMENT

Yutaka Nomaguchi and Kikuo Fujita

Keywords: *Design knowledge, Design theory, Product family, Product architecture*

1 Introduction

In a recent highly competitive global marketplace, many manufacturing companies have been tackling integrity of varying high-end performances and cost reduction of product. They are utilizing product families to diversify and enhance the product performance by simultaneously designing multiple products under commonalization and standardization [3]. The key to rationale and successful product family design is product architecture [10]. Product architecture is the scheme by which the functions and the customer's requirements are allocated to components. Good product architecture yields a successful product family. Platform design of automobiles, i.e., Volkswagen's *A-Platform* [11], is a representative example, which shows the power of established product architecture. In order to establish superior product architecture, a designer should grasp mappings between customer's requirements, functions and components, and commonalize the components while diversifying the target customer's requirement.

We have been stated that knowledge management approach is required to tackle such complicated design problem as product architecture design [6]. The aim of this research is to establish the knowledge management methodology to support product architecture design by explicitly representing mappings between customer's requirements, functions and components, and to implement the methodology as a knowledge-management-based design support system. This paper formalizes product architecture design process to establish a knowledge model and operations, which are the basis of the knowledge management methodology. As a knowledge model, the research introduces PAQS (Product Attribute Quantity Space). PAQS is a quantity space of product attributes in the aspect of customer's requirement, function and substance. Mappings among the attributes are also defined on PAQS. A mapping from functions to customer's requirements is represented by a membership function of fuzzy theory, and a mapping from substances to functions is represented by a numerical equation. A product is represented as a set of attributes and their values on PAQS.

The above features of PAQS facilitate to grasp product architecture through a qualitative relationship among customer's requirements, functions and substances, and deploy products under the product architecture. This is the point because goodness of product architecture design must be measured by performance variation and costs of products deployed under the product architecture.

2 Product Architecture Design and Knowledge Management Approach

2.1 Requisites to support product family deployment

Ulrich defines product architecture as; (1) the arrangement of functional elements; (2) the mapping from functional elements to physical components; (3) the specification of the interfaces among interacting physical components [10]. Although Ulrich mainly focuses on functional elements, this research also focuses on customer's requirements and mappings from customer's requirements to components via functional elements because it is the point of product family deployment to consider a variety of customer's requirements.

(I) Simultaneously designing multiple products under commonalization and standardization: A product family project aims both a variety of products and cost reduction, which would be yielded by volume effect of commonalization and standardization through the product family. It is impossible unless multiple products are simultaneously designed in a product family project.

(II) Well-refining mapping of customer's requirement and function to component: As seen in many design methods, one of the most common understanding of product design is that a specification of a product should be determined in terms of satisfying target customer's requirements and maximizing the product integrity concerning of performance and cost. In QFD (Quality Function Development), for example, customer's requirements are mapped to functional elements, which are finally mapped to components of the product. In a product family project, this nature of product design is emphasized much more than design project of a single product, because multiple products of product family should be deployed to satisfy various customer's requirements under commonalization and standardization. In order to perform such deployment of multiple products, a project has to refine the mappings of customer's requirements and functions to components. Golf series of Volkswagen [11] is a successful case. A-platform of Golf series is shared across several models and brands, i.e., New Beatle, Bora, Skoda Fabia, Seat Altea and Audi A3. These automobiles differ in appearance and brand image, which are the target customer's requirements of the series. A-platform consists of the floor group, drive system, running gear along with unseen parts of the cockpit. These components scarcely affect the target customer's requirements although they can be expected to yield cost reduction by commonalization. This is the reason why A-platform could be a good platform of Golf series. A project should found such successful platform by refining mappings of customer's requirements, functions and components.

(III) Grasping commonality and differences of multiple products: A product family project has to grasp commonality and differences of multiple products under the mappings between customer's requirements, functions and components. The point is not only what the differences among the products are, but also how much different they are. The market segmentation grid [5] is one of good methods to grasp commonality and deference of multiple products. Figure 1 shows an example of the segmentation grid of screwdriver market. The horizontal axis of this grid represents the user of screwdriver, and the vertical axis represents torque of screwdriver. When products deployed are too much distributed in the segmentation, they might be not able to be commonalized. When distribution of products deployed is too narrow, it means that too much commonalization might be employed. A project hast to determine appropriate deployment of multiple products while grasping commonality and differences of them.

(IV) Controlling design space to exploring principal attribute: Based on above discussion, Fig.2 illustrates differences between product family project and design project of

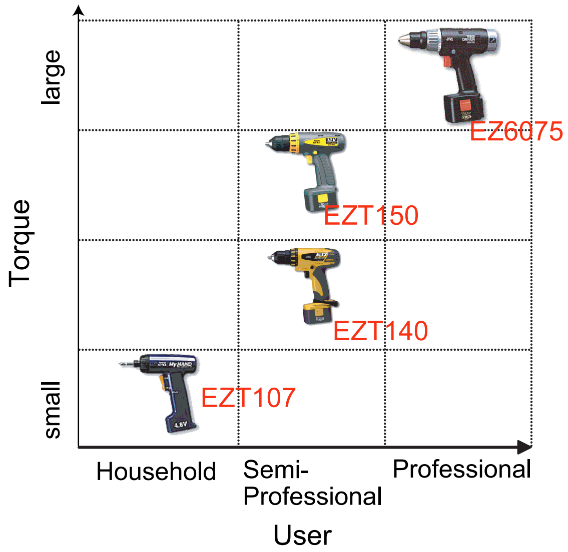


Figure 1 Market segmentation grid of screwdriver family

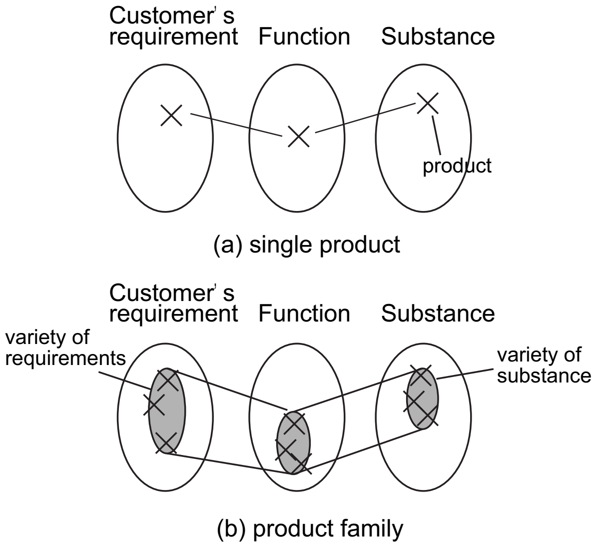


Figure 2 Defferences between design of single product(a) and product family design (b)

a single product. Design of a single product can be understood as determining a mapping one target in customer's requirement space to one member of substance space via function space. A product family design can be understood as determining mappings plural members of customer's requirement space to plural members of substance space via function. Besides, in a product family project, the more various customers' requirement covered by product family and the more commonalized components, the better design. It is obvious that a product family design is more complicated than design of a single product. Therefore, a product family design requires a designer to grasp principal attributes, which contribute to diversify customer's requirements and to commonalize substances of products, by reducing the design space and making the problem more tractable [9]. However, this must be carefully executed because designers wish to "explore varying levels of platform commonality to help to identify variables to make common and unique within the family" [8]. This is why controlling design space is more critical for a product family project than design project of a single product.

2.2 Knowledge management approach

This research considers that the knowledge management approach can be effective for supporting such complicated design problem as product architecture design. For the direction knowledge model and operations must be defined so as to capture designer's knowledge and any means of explicitly representing design rationale is indespensable to help him/her[6]. First of all, this research introduces a product attribute quantity space called PAQS as a knowledge model to meet the requisites of product family deployment stated above. PAQS has the following four features; (i) describing a product by a set of attributes and their values to meet the requisite **(III)**, (ii) defining quantity space of product attributes to meet the requisite **(I)**, (iii) defining mapping between attributes in different aspects to meet the requisite **(II)**, and (iv) defining three levels of PAQS in terms of the number of product attributes to meet the requisite **(IV)**. Then, this research defines the operations which facilitates a designer to explore and establish a superior product architecture.

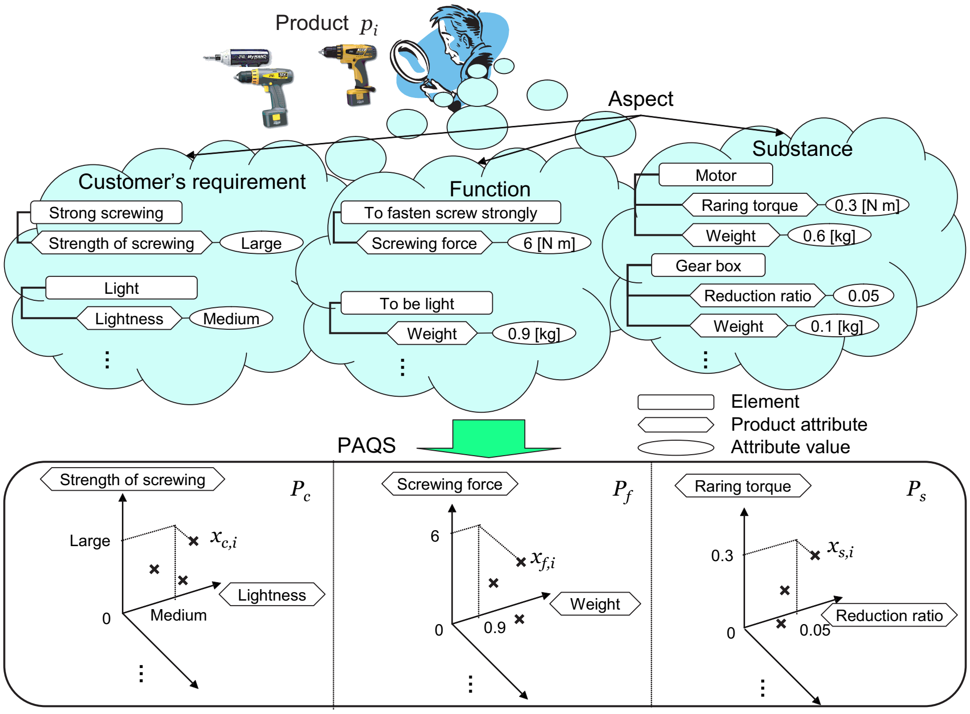


Figure 3 Overview of PAQS

2.3 Glossary

Before the beginning of the detail explanation of PAQS, the following definitions should be stated. Figure 3 illustrates the overview of terminology of PAQS.

Product attribute: A product attribute is a characteristic feature of a product. An **attribute quantity** is defined for each product attribute. A product has a specific **value** of each product attribute. In this paper, a symbol a corresponds to a product attribute. A symbol q is used to represent an attribute quantity. A symbol v is used to represent a specific value of an attribute quantity. A subscript j of a, q and v means an identity serial number of product attribute, that is, a_j, q_j means jth attribute and its quantity. J means the number of product attributes.

Product attribute quantity space: An arbitrary set of product attributes, $\{a_j\}$, which are appropriate to describe a product, constitutes a space called PAQS. A symbol P corresponds to a PAQS. A set of product attributes, $T = \{a_j\}$, constitutes the basis of PAQS.

Product description A product can be represented as a set of values of product attributes. A different product has a different value for the same product attribute. Supposed that a symbol p_i represents an ith product, a jth attribute's value of an p_i is described as v_{ij}. A product p_i can be identified by a set of attribute values, $x_i = \{v_{ij}\}$, which is a member of a PAQS. v_{ij} equals to 0, when a product p_i has nothing to do with a product attribute a_j. This definition enables to handle various products in a single PAQS.

Aspect: An aspect is a view point to represent a product. All product attributes are categorized into one of the three aspects; **customer's requirement**, **function** and **substance**. An initial letter of each aspect is subscribed for symbols; i.e., $a_{c,j}$ means jth product attribute of customer's requirement. A symbol y is used to describe one of aspects; $y \in \{c, f, s\}$. A set of product attributes categorized into each aspect constitutes a subspace of a PAQS. A symbol P_c, P_f, P_s means a subspace of customer's requirement, a subspace of function and a subspace of substance, respectively.

Element: An element represents a partial character of product. A product attribute is clearly defined by defining an element which describes the detail of the product. A symbol e represents an element. An element is defined in each aspect as follows; an element of customers requirement aspect, e_c, is a customer's requirement, i.e., 'be strong screwing.' An element of function aspect, e_f, is a function, i.e., 'to screw a thick screw.' An element of substance aspect, e_s, is a **component**, i.e., 'bite' and 'motor.' The kth element of p_i is described as e_{ik}. K_i is the number of elements of p_i. A product attribute corresponds to just one element of p_i. A set of product attributes corresponds to e_{ik} is described as follows: $T_{ik} = \{a_{j_{ik1}}, a_{j_{ik2}}, ..., a_{j_{ikJ_{ik}}}\}$, here $j_{ikn} \in \{1, 2, ..., J\}$, and J_{ik} is the number of product attributes defined to e_{ik}. By definition, $\bigcup_k T_{ik} = \{a_j | a_j \in T\}$.

3 Knowledge Model of Product Architecture Design

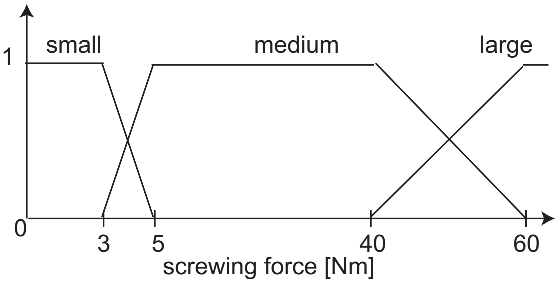
Based on the above discussion of product family deployment, this research introduces Product Attribute's Quantity Space; PAQS, as a knowledge model of product architecture design. PAQS is a quantity space of product attributes in the aspect of customer's requirement, function and substance. In this section, the example of electrical screwdriver design is used to explain the definition of PAQS.

3.1 Product attribute

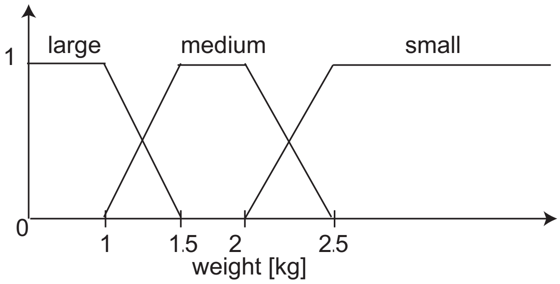
(1) Product attribute of customer's requirement A product attribute of customer's requirement (PAC) represents a criterion, by which a customer finds value and effect in a functional attribute of a product. A single PAC corresponds to a value of a certain function $h(x_f)$, which is function of a set of plural PAF (product attribute of function), x_f, stated bellow. In an electrical screwdriver design, 'strength of screwing' and 'lightness' are instances of attribute names of PAC. Because a value of PAC is fuzzy and linguistic, i.e., 'large' and 'small', it is represented by a fuzzy set $s_{j,m}$ and its the membership grade $\mu_{j,m}$ as eq.(1).

$$q_{c,j} = \sum_m \mu_m(h_j(x_f)) \qquad (1)$$

where $\sum_m$ means an union of membership grades. For example, fuzzy sets $\{s_{1,m}\} = \{small, medium, large\}$ are defined for 'strength of screwing ($a_{c,1}$) as illustrated in Fig.4(a). The membership function of the fuzzy set is defined as a function of h_j. In this case, h_j is an identity mapping from a single PAF named 'screwing force'. The value of 'strength of screwing' can be described by a union of the membership grades of $s_{1,m}$, i.e., $\{small : 0.8,\ medium : 0.2\}$.



(a) membership function of strength of screwing



(b) membership function of lightness

Figure 4 Example of membership function

A set of PAC in P is described as $T_c = \{a_{c,1}, a_{c,2}, ..., a_{c,J_c}\}$, where $a_{c,j}$ is jth PAC, and J_c is the number of PAC defined in P. T_c constitues a subspace of P; P_c. A set of values of PAC of p_i can be represented by a member of P_c; $x_{c,i} = (v_{c,i1}, v_{c,i2}, ..., v_{c,iJ_c})$.

(2) Product attribute of function A product attribute of function (PAF) represents an engineering metrics of a function of a product, which can be recognized as an output of a product. In an electrical screwdriver design, 'screwing force,' 'bite rotation per minutes,' 'battery duration' and 'weight' are instances of attribute names of PAF. Quantity of PAF is continuous and positive real number.

A set of PAF in P is described as $T_f = \{a_{f,1}, a_{f,2}, ..., a_{f,J_f}\}$, where $a_{f,j}$ is jth PAF, and J_f is the number of PAF defined in P. T_f constitues a subspace of P; P_f. A set of values of PAF of p_i can be represented as a member of P_f; $x_{f,i} = (v_{f,i1}, v_{f,i2}, ..., v_{f,iJ_f})$.

(3) Product attribute of substance A product attribute of substance (PAS) represents a characteristic feature of a component of a product. A PAS is a controllable factor of a product, while a PAF is an output of a product. In an electrical screwdriver design, 'motor rating torque,' 'motor rating round per minute,' 'reduction gear ratio' and 'battery weight' are instances of attribute names of PAS. Quantity of PAS is continuous and positive real number.

A set of PAS in P, is described as $T_s = \{a_{s,1}, a_{s,2}, ..., a_{s,J_s}\}$, where $a_{s,j}$ is jth PAS, and J_s is the number of PAS defined in P. T_s constitutes a subspace of P; P_s. A set of values of PAS of p_i can be represented as a member of P_s; $x_{s,i} = (v_{s,i1}, v_{s,i2}, ..., v_{s,iJ_s})$.

3.2 Mappings

According to the above definition of a product attribute, product attributes of three aspects are related each other, i.e., PAC is related to PAF, and PAF is related to PAS. These relationships between product attributes are defined as mappings in a PAQS. A symbol f is used to describe a mapping.

(1) Mapping from PAS to PAF A mapping from PAS to PAF (MSF) corresponds to a relationship between controllable factors and an output behavior of a product. An MSF maps plural PAS to a single PAF. Just one MSF maps to a single PAF. Therefore, the number of MSF in P equals to J_f.

An MSF consists of plural numerical equations among the plural PAS and the single PAF. A numerical equation of an MSF is generally based on physical phenomena and engineering theories, i.e., dynamics, kinematics, strength of materials and so on. Generally speaking, the equation depends on a physical structure of a product, so that each equation of an MSF has the exclusive condition corresponding to the physical structure.

In an electrical screwdriver design, for example, an MSF ($f_{sf,1}$) to 'screwing force ($a_{f,1}$)' from 'motor rating torque ($a_{s,1}$),' 'reduction gear ratio ($a_{s,2}$)' and 'reduction ratio of impact mechanism ($a_{s,3}$)' is defined as eq.(2).

$$q_{f,1} = f_{sf,1}(q_{s,1}, q_{s,2}, ..., q_{s,J_s}) \tag{2}$$

where, $f_{sf,1}$ is defined in eq.(3).

$$f_{sf,1} = \begin{cases} \frac{q_{s,1}}{q_{s,2}} & : (q_{s,3} = 0) \\ \frac{q_{s,1}}{q_{s,2} \cdot q_{s,3}} & : (q_{s,3} \neq 0) \end{cases} \tag{3}$$

In this case, all PAS quantities except for $q_{s,1}$, $q_{s,2}$ and $q_{s,3}$ have nothing to do with determining $q_{f,1}$. $q_{s,1}$, $q_{s,2}$ and $q_{s,3}$ are called input attribute quantities of $f_{sf,1}$. A set of input attribute quantities of f is represented by $L^{in}(f)$.

A set of all MSF in P is described as $F_{sf} = \{f_{sf,1}, f_{sf,2}, ..., f_{sf,J_f}\}$. $x_{f,i}$ is determined by eq.(4).

$$x_{f,i} = F_{sf} \cdot x_{s,i} \tag{4}$$

(2) Mapping from PAF to PAC A mapping from PAF to PAC (MFC) corresponds to a relationship between a functional output of a product and a criterion, by which a customer finds value and effect in a function of a product. Plural PAF is mapped to a single PAC by a MFC via a certain function $h(x_f)$. Just one Mfs corresponds to a single PAC. Therefore, the number of MFC equals to the number of PAC. An MFC is represented by membership functions of fuzzy theory, which map a value of $h(x_f)$ to membership grades of each of fuzzy sets. The membership function can be generally defined through market analysis. In an electrical screwdriver design, for example, an MFC ($f_{fc,1}$) from 'screwing force ($a_{f,1}$)' to 'strength of screwing ($a_{c,1}$)' is defined as eq.(5).

$$q_{c,1} = f_{fc,1}(q_{f,1}, q_{f,2}, ..., q_{f,J_f}) \tag{5}$$

where, $f_{fc,1}$ is defined in eq.(6).

$$f_{fc,1} = \sum_m \mu_{1m} \cdot h_1 \tag{6}$$

The membership functions μ_{1m} of $f_{fc,1}$ are shown in Fig.4 (a). In this case, h_1 is identity mapping from $\{q_{f,1}\}$, and all PAF quantities except for $q_{f,1}$ have nothing to do with determining a value of $q_{c,1}$, that is, $L^{in}(f_{fc,1}) = \{q_{f,1}\}$.

A set of MFC P is described as $F_{fc} = \{f_{fc,1}, f_{fc,2}, ..., f_{fc,J_c}\}$. $x_{c,i}$ is determined by eq.(7).

$$x_{c,i} = F_{fc} \cdot x_{f,i} \tag{7}$$

(3) Synthetic mapping The design process includes synthesis-oriented phase, where a designer determines function from customer's requirement, and determines substance from function; and analysis-oriented phase, where a designer analyses function from substance, and customer's requirement from function. Both of MSF and MFC are analytical mapping, which supports analysis-oriented phase. It is relatively easy to determine general analytical mappings. However, it is pointed out that defining synthetic mappings is case-specific. In this research, it is supposed that the synthesis-oriented phase is performed by referring the analytical mapping. As shown in Fig 5, when a value of the PAC, $a_{c,1}$, is determined as $v_{c,11}$, a designer can determine a value of the PAF, $a_{f,1}$, within the range limited by $v_{c,11} = f_{fc,1}(x_{f,1})$. In this way, a designer can determine a value of the PAS $a_{s,1}$ within the range limited by $v_{f,11} = f_{sf,1}(x_{s,1})$.

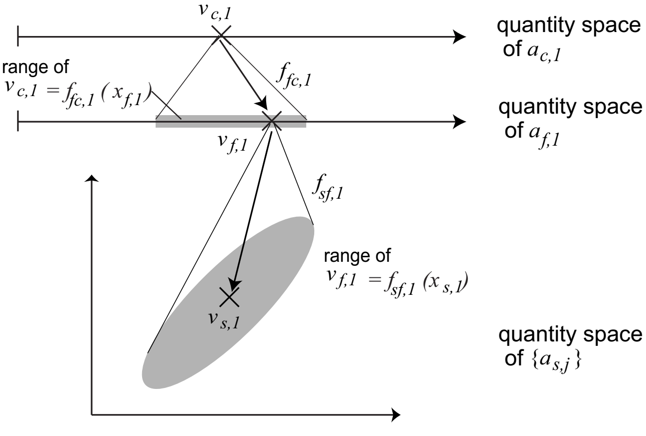


Figure 5 Synthetic mapping

3.3 Level of PAQS

In order to deploy a product family, a designer has to control design space to exploring principal attribute as stated in subsection 2.1. A design space on a PAQS can be controlled by increasing or decreasing the number of product attributes, J. In order to formalize the process of controlling design space, this research defines the following three levels of PAQS;

Design space (PAQS-1): PAQS of level 1 (PAQS-1) means design space of product family. A designer recognizes a product by using of the necessary and sufficient set of attributes, i.e., ’strength of screwing,’ ’weight’ and ’motor rating torque’ of a screwdriver, while any other attributes, i.e., ’thermal conductivity’, ’smell’ and ’feeling of touch’ are omitted. PAQS-1 is such actual design space. A symbol P, which has no accent mark, means PAQS-1. A member of P is a product, which is represented by x without accent mark. A set of products is represented by S.

Principal attribute space (PAQS-2): PAQS of level 2 is a principal attribute space, which consists of a few principal attributes in order to deploy a product family. PAQS-2 is a subspace of PAQS-1. A symbol $\tilde{P}$ represents PAQS-2. A member of $\tilde{P}$ is description of a product on PAQS-2, which is represented by $\tilde{x}$. A set of $\tilde{x}$ is represented by $\tilde{S}$.

Utmost attribute space (PAQS-0): The set of attributes on PAQS-1 is arbitrary, because *the necessary and sufficient set of attributes* can enlarge without limit in order to describe a product more and more precisely. If necessary, ’smell’ and ’feeling of touch’ would be considered as attributes of a screwdriver. PAQS of level 0 is an ideal PAQS, which includes the utmost set of attributes in order to describe the product. By defining PAQS-0, PAQS-1 can be defined as a subspace of PAQS-0. A symbol $\hat{P}$ means PAQS-0. A member of $\hat{P}$ is represented by $\hat{x}$. Note that it is not feasible to actually represent $\hat{x}$ because nobody can recognize *utmost* of an attribute set. A set of $\hat{x}$ is represented by $\hat{S}$.

4 Formalization of Product Architecture Design Process on PAQS

In this section, product architecture design process is formalized based on PAQS.

4.1 Assumption

Before the formalization of product architecture design process on PAQS, we should clarify our assumption about product family deployment. The aim of product family deployment is not to create a totally new product, but to modify existing products or existing idea of products to deploy. That is, a designer can refer the existing products/ideas at the beginning of product family project. Note that this assumption does not restrict any design case of product family from PAQS. PAQS can represent the totally new design and its deployment to product family, although PAQS cannot support the process of totally new design.

4.2 Formation of PAQS

As stated in subsection 3.3, PAQS has three levels in order to control design space. Because PAQS-0 is an ideal design space, PAQS-1 and PAQS-2 is considered as an actual design space of product family. In each of two levels, we can define a quantity space, whose basis is a set of product attributes, and a product, which is a member of the quantity space. Therefore, the following four formations of PAQS can be defined;

(I) **Product/products x, S:** a member/members of PAQS-1.
(II) **Design space P:** PAQS of Level 1 called PAQS-1.
(III) **Principal attribute space $\tilde{P}$:** PAQS of Level 2 called PAQS-2.
(IV) **Derivative product/products/ $\tilde{x}, \tilde{S}$:** a member/members of PAQS-2

4.3 Operations

Among the four formations stated above, the following five steps can be defined.

Step (0): Describing products: attributes and values of products are defined.
Step (1): Mapping between aspects: inter-aspect mappings among the attributes are defined.

Step (2): Selecting principal attributes: a few principal attributes, which is dominant for the product family, are selected.

Step (3): Deploying derivative products: values of each derivative product are determined while focusing on the principal attributes.

Step (4): Determining subsidiary attributes: values of subsidiary attributes, which is not selected at step(2), are determined.

The five steps consists of cyclic process as illustrated in Fig6. However, note that this cyclic process includes back-and-forth process, which is the nature of design.

Each step consists of some operations to perform the step. The operations fall into the following three categories; (a) an operation to control the number of attributes, (b) an operation to set up/cancel mappings between aspects, and (c) an operation to control values of attributes of each product. Figure 7 depicts operations of type (a) and (b) on Venn diagram of product attributes on PAQS of each level. The arrow in the Fig.7 means the operation, and the number attached to the arrow, i.e., (2)-(i), means ID of operation, whose detail is explained in section 5.

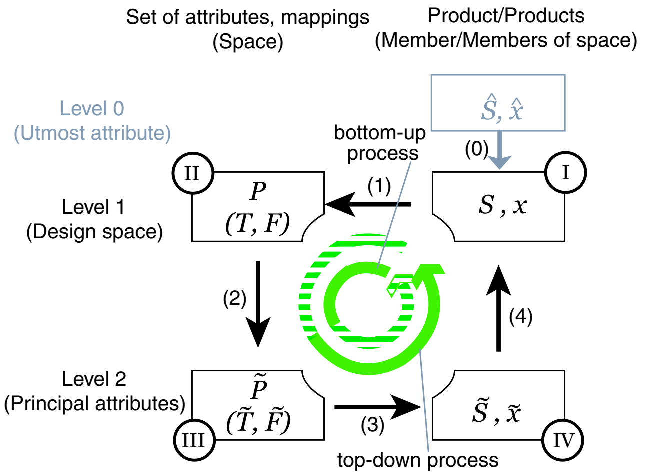


Figure 6 Cyclic steps of product family design

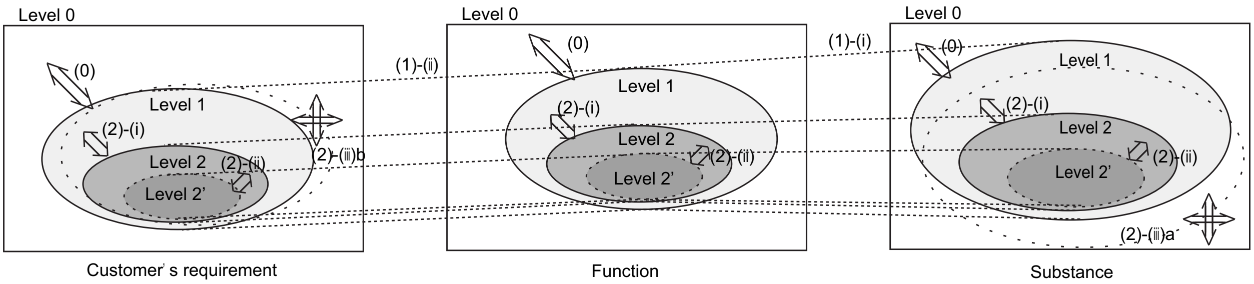


Figure 7 Venn diagram of product attributes in PAQS

5 Design Operation on PAQS with Illustrative Example

This section introduces design operation defined on PAQS with an illustrative example of product family design of an electrical screwdriver and an impact driver. An impact driver has an impact mechanism to generate thrust force by impact frequency. By thrust force, the screwing force of an impact driver is larger than that of a usual screwdriver.

The followings subsections explain the operations of each step stated in subsection 4.3.

5.1 Describing products [Step (0)]

In this step, a designer defines attributes and values of each existing product or existing idea of product. Table 1 shows the line up of elements and product attributes of p_1 and p_2 defined in this step. This step consists of the following three operations.

(0)-(i) List up products: This is the operation to list up each existing product. A set of listed-up products is represented as follows; $\{p_1, p_2, ..., p_i, ...p_I\}$. In the case of a screwdriver and an impact driver, for example, $p_1 = \mathit{screwdriver}, p_2 = \mathit{impactdriver}$

(0)-(ii) List up elements: This is the operation to list up elements of the listed-up products in order to list up product attributes. So far as a product p_i, the listed-up elements of each aspect is represented as follows; $E_{y,i} = \{e_{y,i1}, e_{y,i2}, ..., e_{y,ik}, ..., e_{y,iK_{y,i}}\}$, where, $K_{y,i}$ is the number of elements in aspect y.

(0)-(iii) List up product attributes: This is the operation to list up product attributes of the listed-up elements. Usually, this operation would be performed together with the operation (0)-(ii). A designer determines the set of attributes $T_{y,ik}$ and their values $V_{y,ik}$ for each product p_i, where, $T_{y,ik}$ is a set of product attributes assigned to $e_{y,ik}$; $T_{y,ik} = \{a_{y,j_{ik1}}, a_{y,j_{ik2}}, ..., a_{y,j_{ikJ_{ik}}}\}$, and $V_{y,ik}$ is a set of values; $V_{y,ik} = \{v_{y,j_{ik1}}, v_{y,j_{ik2}}, ..., v_{y,j_{ikJ_{ik}}}\}$. $J_{y,ik}$ is the number of product attributes assigned to $e_{y,ik}$.

Table 1 Description of products

Element	Attribute	p_1	p_2	Unit
Customer's requirement				
strong screwing ($e_{c,i1}$)	*strength of screwing* ($a_{c,1}$)	M:1.0	L:1.0	-
quick screwing ($e_{c,i2}$)	*quickness of screwing*($a_{c,2}$)	M:1.0	M:1.0	-
screwing thick screw ($e_{c,i3}$)	*thickness of screw* ($a_{c,3}$)	M:1.0	L:1.0	-
working for a long time ($e_{c,i4}$)	*working duration* ($a_{c,4}$)	M:1.0	M:1.0	-
light ($e_{c,i4}$)	*lightness* ($a_{c,5}$)	L:1.0	M:1.0	-
Function				
to fasten screw strongly ($e_{f,i1}$)	*screwing force* ($a_{f,1}$)	6	60	[N m]
to fasten screw quickly ($e_{f,i2}$)	*bite rpm* ($a_{f,2}$)	90	360	[rpm]
to fasten thick screw ($e_{f,i3}$)	*capable screw thickness* ($a_{f,3}$)	5	10	[cm]
to supply power for a long time ($e_{f,i4}$)	*battery duration* ($a_{f,4}$)	2	2	[hour]
to be light	*weight* ($a_{f,5}$)	0.9	1.8	[kg]
Substance				
motor ($e_{s,i1}$)	*rating torque* ($a_{s,1}$)	0.3	0.5	[N m]
	rating rpm ($a_{s,2}$)	1800	2400	[rpm]
	weight ($a_{s,3}$)	0.6	0.8	[kg]
gear box ($e_{s,i2}$)	*reduction ratio* ($a_{s,4}$)	0.05	0.15	-
	weight ($a_{s,5}$)	0.1	0.4	[kg]
battery ($e_{s,i3}$)	*battery capacity* ($a_{s,6}$)	2	2	[hour]
	weight ($a_{s,7}$)	0.2	0.2	[kg]
impact mech. ($e_{s,i4}$)	*impact ratio* ($a_{s,8}$)	0	18	-
	weight ($a_{s,9}$)	0	0.4	[kg]

In Table 1, the values of all attributes of *battery* ($e_{s,i3}$) of p_1 is same as those of p_2. This means that *battery* of p_1 ($e_{s,13}$) and *battery* of p_2 ($e_{s,23}$) is commonalized. On the other hand, the value of each attribute of *impact mechanism ($e_{s,i4}$)* of p_1 is 0. This means that *impact mechanism* does not exist in p_1.

After describing products, a designer can estimate production cost of components based on the cost model of product family [3]. Table 2 shows the production costs of each component of screwdriver and impact driver. The production number of screwdriver and impact driver is 15,000 and 10,000, respectively. In this case, total cost is about 8,700,000 yen.

5.2 Mapping between aspects [Step (1)]

In this step, mappings among the attributes are defined on PAQS-1 so as to grasp relationships between different aspects.

This step consists of the following two operations.

(1)-(i) Defining MSF: This is the operation to define $f_{sf,j}$, which is an MSF from T_s to $a_{f,j}$.

(1)-(ii) Defining MFC: This is the operation to define $f_{fc,j}$, which is an MFC from T_f to $a_{c,j}$.

After this step, a set of MSF, F_{sf}, and a set of MFC, F_{fc}, are defined. Figure 8 illustrates $F_{sf} = \{f_{sf,j} | j \in \{1,2,3,4,5\}\}$ and $F_{fc} = \{f_{fc,j} | j \in \{1,2,3,4,5\}\}$ defined on PAQS-1 of screwdriver and impact driver. For example, $f_{sf,1}, f_{sf,2}$ and $f_{sf,5}$ are defined in eq.(8), (9) and (10), respectively.

$$f_{sf,1} = \begin{cases} \frac{q_{s,1}}{q_{s,4}} & : (q_{s,8} = 0) \\ \frac{q_{s,1} \cdot q_{s,8}}{q_{s,4}} & : (q_{s,8} \neq 0) \end{cases} \tag{8}$$

$$f_{sf,2} = \frac{q_{s,2}}{q_{s,4}} \tag{9}$$

$$f_{sf,5} = q_{s,3} + q_{s,5} + q_{s,7} + q_{s,9}. \tag{10}$$

The definition of $f_{fc,1}$ and $f_{fc,5}$ are illustrated in Fig.4(a) and Fig.4(b), respectively.

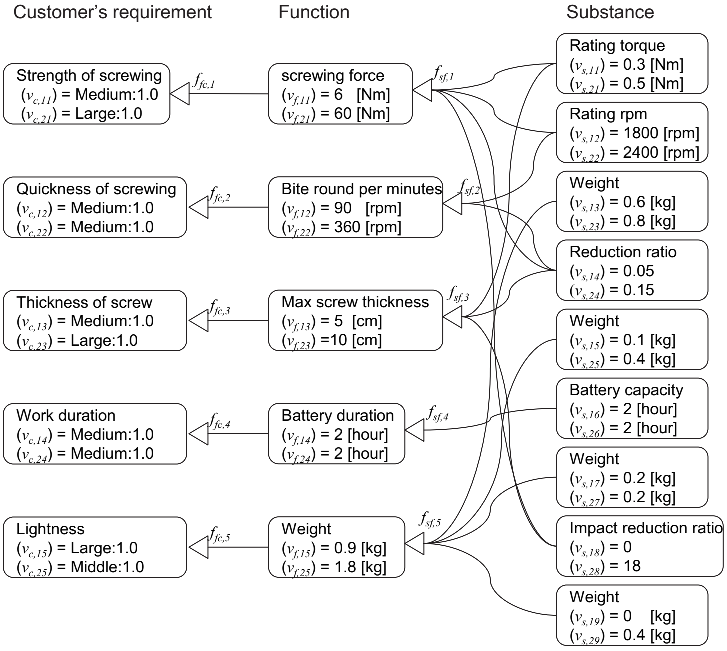


Figure 8 Mappings in PAQS-1

5.3 Defining principal attributes [Step (2)]

In this step, a few principal attributes $\tilde{T}_y$, which is dominant for diversity or commonality of products, are defined. $\tilde{T}_y$ constitutes the basis of PAQS-2.

This step consists of the following four operations.

(2)-(i) Select principal attribute: This is the operation to select principal attributes $\tilde{T}_y$, where $\tilde{T}_y \subset T_y$. At the beginning of this operation, a designer select a set of target customer's requirement, $\tilde{T}_c = \{a_{c,j_c}\}$. And then, a set of PAF $\tilde{T}_{f0} = \{a_{f,j_f} | a_{f,j_f} \in L^{in}(f_{c,j_c}),\ a_{c,j_c} \in \tilde{T}_c\}$ and a set of PAS $\tilde{T}_{s0} = \{a_{s,j_s} | a_{s,j_s} \in L^{in}(f_{f,j_f}),\ a_{f,j_f} \in \tilde{T}_{f0}\}$ is selected. After all, a designer selects a set of a few principal attributes $\tilde{T}_f \subset \tilde{T}_{f0}$, $\tilde{T}_s \subset \tilde{T}_{s0}$. In the design case of screwdriver and impact driver, *strength of screwing ($a_{c,1}$)* and *lightness ($a_{c,5}$)* are selected as a target customer's requirement. In this case, all PAS is related to both of $a_{c,1}$ and $a_{c,5}$. However, a designer excludes $a_{c,6}$ and $a_{c,7}$, because $e_{s,i3}$ (*battery*) is already shared by both products. Therefore, 11 of 19 attributes are selected as principal attributes $\tilde{T}_y$. Figure 9 illustrates an example of PAQS-2 as a result of the above operation. A black-highlighted node means a product attribute of $\tilde{T}_y$.

(2)-(ii) Defining coupling of attributes: This is the operation to define a representative attribute, which controls another attribute although they are theoretically independent. The operation (2)-(ii) reduces the number of the variations by coupling each other so as to grasp the principal attribute. By this operation, a certain coupling function g is defined as follows; $q_{y,j_{dep}} = g(q_{y,j_{rep}})$, where $a_{y,j_{rep}}$ is the representative attribute, and $a_{y,j_{dep}}$ is the attribute depending on $a_{y,j_{rep}}$. PAQS constituted by a set of $\{a_{y,j_{rep}}\}$ is called Level-2'.

(2)-(iii) Modifying product architecture: This is the operation to modify product architecture by adding/canceling attributes, components and mappings in terms of defining principal attributes. The following two types according to modified aspect is defined; (A)

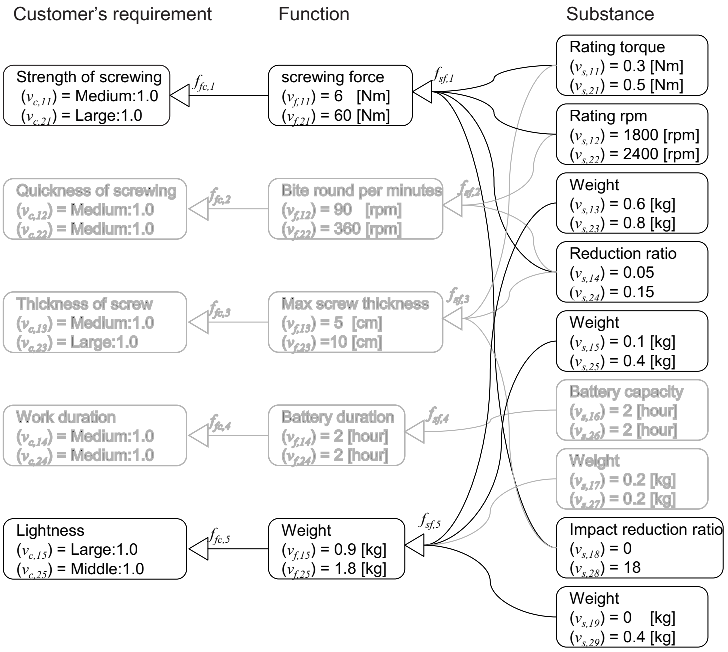


Figure 9 PAQS-2

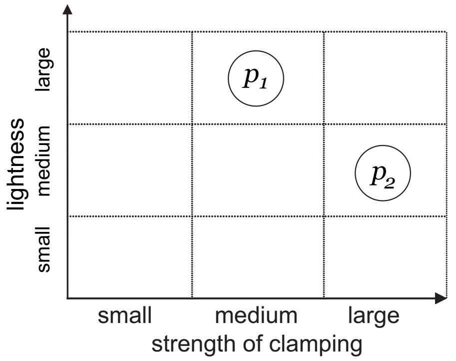


Figure 10 Target customer's requirement in market segment grid

Modifying substance then modifying function and customer's requirement, and (B) Modifying customer's requirement then modifying function and substance. The detail of each type is same as operation(0)-(ii), (0)-(iii), respectively.

(2)-(iv) Determining target customer's requirement: This is the operation to determine the target of customer's requirement in the market segmentation grid. By this operation, a set of values of $\tilde{T}_c$, $\tilde{V}^t_c$, are determined. The values are represented by $\tilde{V}^t_c = \{\tilde{v}^t_{c,1}, \tilde{v}^t_{c,2}, ..., \tilde{v}^t_{c,\tilde{J}_c}\}$. In the example, $\tilde{V}^t_c$ is determined as shown in Fig.10.

5.4 Deploying derivative products [Step-(3)]

In this step, values of PAS of each derivative product are determined under commonalization and diversification so as to meet the target customer's requirements. In this step, the values of PAC and PAF listed-up in PAQS-1 are determined by MSF and MFC, after determined values of PAS. Note that operations of this step would also change a value of a subsidiary attribute. Its change should be calculated but neglected by a designer so as to reduce complication. This is the benefit of introducing PAQS-2.

This step consists of the following three operations.

(3)-(i) Commonalization: This is the operation to equalize values of a certain component among plural products. When a designer determines to commonalize e_{s,i_1k_1} and e_{s,i_2k_2}, the following equation exists; $v_{s,i_1j} = v_{s,i_2j}$, where $j \in \{1, 2, ..., J_{i_1k_1} = J_{i_2k_2}\}$. There are two types of commonalization, (A)Partial commonalization, whereby a certain component of a few of all products are commonalized and (B)Total commonalization, whereby a certain component of all products are commonalized.

(3)-(ii) Diversification: This is the inverse operation of (3)-(i), that is, the operation to diversify values of commonalized component. As a result of this operation, the commonality of the components is canceled.

(3)-(iii) Individual adjustment: This is the operation to determine the value of a principal PAS of individual product.

For example, supposed that a designer decided to commonalize motors of screwdriver

Table 2 Production cost before commonalization

(1000 yen)

component	production	total
motor ($e_{s,11}$)	15,000	2,547
gear box ($e_{s,12}$)	15,000	848
motor ($e_{s,21}$)	10,000	2,037
gear box ($e_{s,22}$)	10,000	738
impact mech. ($e_{s,24}$)	10,000	2,537
total		8,707

Table 3 Production cost after commonalization

(1000 yen)

component	production	total
motor($e_{s,11}$, $e_{s,21}$)	25,000	3,563
gear box ($e_{s,12}$)	15,000	848
gear box ($e_{s,22}$)	10,000	738
impact mech.($e_{s,24}$)	10,000	2,537
total		7,686

and impact driver by operation (3)-(i). The values are determined as follows; 'rating torque' $\tilde{v}_{s,11} = \tilde{v}_{s,21} = 0.3$, 'rating rpm' $\tilde{v}_{s,12} = \tilde{v}_{s,22} = 1800$, 'weight' $\tilde{v}_{s,13} = \tilde{v}_{s,23} = 0.6$. This commonalization changes the quantity of 'screwing force ($\tilde{v}_{f,21}$),' 'weight ($\tilde{v}_{f,22}$),' 'bite round per minute ($v_{f,22}$)' and 'strength of screwing ($\tilde{v}_{c,21}$)' of impact driver as follows; $\tilde{v}_{f,21} = 36, \tilde{v}_{f,22} = 1.5, v_{f,22} = 270, \tilde{v}_{c,21} = \{medium : 1.0\}$. By these changes, the value of 'strength of screwing' does not meet the target. A designer has to modify $\tilde{v}_{f,21}$ to meet the range of $\{large : 1.0\}$ of $\tilde{v}_{c,21}$.

The designer has four alternatives to solve the problem; (a) to change gear box or impact mechanism, which is related to determining 'screwing force,' by operation (3)-(iii), (b) to give up commonalization of motor by operation (3)-(ii), (c) to give up the target customer's requirement by operation (2)-(iv), and (d) to change components of product by operation (2)-(iii,A) so as to use any other mechanism to gain 'screwing force.' Figure 11 illustrates the result of applying (a) so as to change gear box. By changing 'reduction ratio ($\tilde{v}_{s,24} = 0.09$)' and 'weight ($\tilde{v}_{s,25} = 0.6$),' related PAF and PAC are changed as follows; $\tilde{v}_{f,21} = 60, \tilde{v}_{f,22} = 1.8, \tilde{v}_{c,21} = \{large : 1.0\}, \tilde{v}_{c,22} = \{middle : 1.0\}$. Table 3 shows the evaluation for this commonalization. The production cost can be reduced by about 1,000,000 yen, while the accomplishment of customer's requirement is kept. The designer can decide if this commonalization is good design, by explicitly evaluating plural alternatives of product family design with the evaluation indices of production cost and the accomplishment of the target customer's requirement.

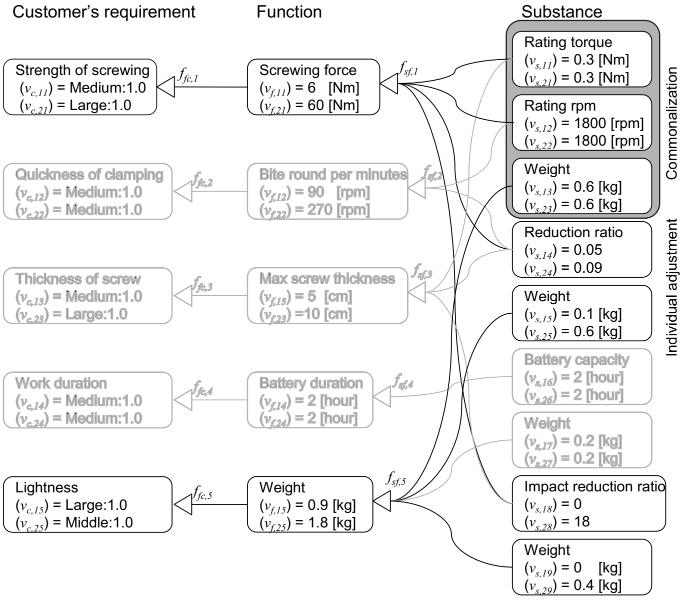


Figure 11 Example of commonalization to meet the target

5.5 Determining subsidiary attributes [Step-(4)]

In this step, values of subsidiary attributes, which is not selected at step(2), are determined.

6 Disscussion

Many research groups have been researching on product family and product architecture design in these days. They can be classified into two approaches based on Dixon's study [2]; prescriptive approach and computational approach.

The researches based on prescriptive approach mainly focus on conceptual design stage of product architecture. Martin and Ishii [4] introduces two indices of sensitivity of variation of component; GVI (Generational Variation Index), which estimates the variation caused by variation of customer's requirement, and CI (Coupling Index), which estimates the variation caused by varying related components. A designer can explore good product architecture by modifying high-GVI or high-CI components. Baldwin and Clark [1] propose six operations for controlling modularity of product architecture; *spliting* a system into two or more modules, *substituting* one module design for another, *augmenting* a new module to a system, *excluding* a module from the system, *inverting* to create new design rules and *porting* a module to another system. On the other hand, the researches based on computational approach mainly focus on optimization of derivative products, which are based on *a priori* defined architecture. There are so many instances of this approach (see [9]).

The former approach provides a design method to explore product architecture. However, it just supports conceptual and qualitative understandings of product family, and it has not yet succeeded in comprehensive implementation including modeling of product family and its optimization. The latter approach has been established as a feasible tool to solve the formalized design problem. However, a designer has to define product architecture, which is a prerequisite of optimization. To utilize the optimization of product architecture design, the process of exploring and defining product architecture should be supported. PAQS integrates benefits of both approaches; quantity space provides a basis of a numeric model, whereby optimization formalization can be employed, and formalized operations on PAQS and evaluation indices can support exploring and defining product architecture suitable for deploying product family.

Although some researchers have conducted to enumerate operations of product family design as stated above, little attention has been given to its formalization. The formalized operations in this research facilitate to define design operations of product architecture design on knowledge management system, which we have been developing [6].

According to the test case of product family of screwdriver and impact driver shown in section 5, it is not too far from the truth to say that this formalization could provide valid operations of product architecture design. To further justify this argument, Simpson's empirical study [7] should be stated as a related work. Simpson categorizes product family projects into two approaches; (i) top-down approach, whereby derivative products are strategically deployed from product architecture and (ii) bottom-up approach, whereby product architecture is defined by integrating existing products. Although Simpson introduces this categorization in order to perform case study, two processes, corresponding to the two approaches, can be considered to iteratively occur in a single design. As shown in Fig.6, it is considered that top-down process mainly consists of step (3) and (4) stated in 4.3 and bottom-up process mainly consists of step (1) and (2). This is why the formalization of this research can be considered to represent actual design cases.

7 Conclusions

This paper formalizes product architecture design process to establish a knowledge model and operations, which are the basis of the knowledge management methodology. The research introduced PAQS as a knowledge model. The illustrative example of product family deployment stated in section 5 shows the power of the PAQS to explore and define a good product architecture.

Our future works includes integrating PAQS with knowledge management system, which we have been developing, by defining design operation based on the formalized operations, and incorporating modular design methodology.

Acknowledgements

The authors gratefully acknowledge helpful discussions with Kenichi Kawakami, Nobuyuki Matsuda and Yoji Wakai on several points in the paper. This research was supported in part by 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] Baldwin, C. Y., and Clark, K. B., *Design Rules, Volume 1. The Power of Modularity,* The MIT Press, 2002.

[2] Dixon, J. R., "On Research Methodology Towards a Scientific Theory of Engineering Design," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AI EDAM),* Vol. 1, No. 3, pp.145-157, 1987.

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

[4] 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.

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

[6] Nomaguchi, Y., Ohnuma, A. and Fujita, K., "Design Rationale Acquisition in Conceptual Design by Hierarchical Integration of Action, Model and Argumentation," *Proceedings of the 2004 ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference*, Paper No. DETC2004/CIE-57681, 2004.

[7] Simpson, T. W., Maier, J. R. A. and Mistree, F., "Product Platform Design: Method and Application," *Research in Engineering Design*, Vol. 13, No. 1, pp. 2-22, 2001.

[8] Simpson, T. W. and D'Souza, B., "Assessing Variable Levels of Platform Commonality within a Product Family Using a Multiobjective Genetic Algorithm," *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization,* AIAA-2002-5427, 2002.

[9] Simpson, T. W., "Product Platform Design and Optimization: Status and Promise," *Proceedings of the 2004 ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference,* Paper No. DETC2004/DAC-48717, 2004.

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

[11] 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. Jürgens and T. Fujimoto, eds., Springer-Verlag, New York, pp. 146-156, 1997.

Corresponding author:
Yutaka Nomaguchi, Assistant Professor
Department of Mechanical Engineering
Graduate School of Engineering, Osaka University
2-1 Yamadaoka, Suita, Osaka 565-0871, JAPAN.

Telephone: +81-6-6879-7324
Facsimile: +81-6-6879-7325
E-mail: noma@mech.eng.osaka-u.ac.jp
URL: http://syd.mech.eng.osaka-u.ac.jp/˜noma