Journal of Civil Engineering and Construction Technology Vol. 2(9), pp. 198-211, 2 September, 2011
Available online at http://www.academicjournals.org/jcect
ISSN 2141-2634 ©2011 Academic Journals

Full Length Research Paper

A framework of customer requirement classification and application to product configuration

Yanhong QIN* and Guangxing WEI

School of Management of Chongqing Jiaotong University, Chongqing 400074, China.

Accepted 29 June, 2011

Modular structure of product family is established and the product family is decomposed into generic modules. Then, the module model denoted by attribute variable is established for each generic module. Based on the classification of customer requirement and erection of product decision tree, the description and explanation type of customer requirement can be used to restrict the sub branch of decision tree so to find out the certain product family satisfying customer. By reference to QFD, the mapping relation of customer requirements to module attributes is formed, which can be the determinate value and weight of module attribute variable. After retrieving the candidate set of modules which are nearest to the customer requirement on the module accordingly, the candidate modules are combined efficiently under the constraints relative to modules and attributes, In this way, many invalidate module combinations can be avoided and hence the efficiency of product configuration is promoted.

Key words: Modular product family, generic module, attribute variable, customer requirement classification.

INTRODUCTION

In an age when consumers demand high-quality, low-priced and customized products, the competition among firms have been aiming at product variety and speed to market. Mass customization aiming at delivering an increasing product variety that best serves customer requirements while keeping mass product efficiency, has recently received numerous attention and popularity in industry and academia alike (Pine, 1993). For the product configuration is aiming at satisfying customer requirement, but various customer will adopt different method to express their own requirement, so it is important to classify customer requirements and translate it to attribute information of product module. Dan and Qin (2008) classified the customer requirement into three types: binary, optional, parameter, and furthermore, Dan and Wang (2008) set a taxonomy model of customer requirements information, and he classified the customer requirement into five types: binary, optional, parameter, description and explanation. Based on the taxonomy model of customer requirement information, the corresponding formal semantic description approaches to these five types of the information about customer requirement were discussed. Yan et al. (2010) analyzed the current research situation of customer requirement in mass customization and built the matter-element model. Su and Dong (2010) pointed out the limitation of mass customization, such as small optional scope of customized user and long delivery time and so on, and then he proposed the model of prototype of dynamic customization system.

A methodology of product family architecture (PFA) was developed to rationalize product development for mass customization and the diverse customers requirements are matched with the capabilities of a firm through systematic planning of modularity in three views, that is, functional, technical and physical views. Besides, mapping of functional requirements to specific modules is considered (Jiao and Tseng, 1999). The product variety optimization includes three degree of optimization problems, that is attribute assignment, module combination, and simultaneous design of both, and when the possibilities of computational optimization for product variety design under fixed product architecture are explored, optimization is demanded to determine the contents of modules and their combination under fixed product architecture (Fujita, 2002). Besides, Fruto and Borenstein (2008) proposed an optimization method of module combination for products in a family, and the Unified

*Corresponding author. E-mail: qinyanhong24@163.com

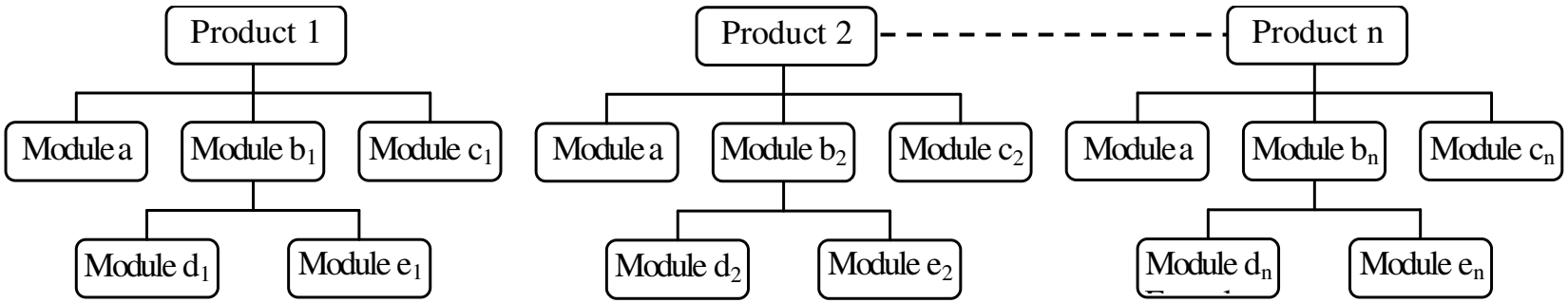

Figure 1. Modular structure of each product in the same product family.

Modelling Language (UML) was used to describe a product family (Zheng and Liu, 2006), and the method also focuses on how the customer's functional requirements can be translated into a selection of specific modules in the product family.

The functional variety is used broadly to mean any differentiation in the attributes related to a product's functionality from which the customer derives a benefit. On the other hand, the technical variety is referred to as diverse technologies, design methods, manufacturing process, components and assemblies, etc., which are necessary to achieve some functionality of a product required by the customer (Liu et al., 2004). While the functional variety is often related to the customer satisfaction, the technical variety usually involves the manufacturability and costs. An important target of product configuration in mass customization is to increase functional varieties, meanwhile reduce technical varieties.

Design for modular product family is the efficient method to realize mass customization. Wang and Lin (2004) established modular structure of product family, and he explained the constraints between module types, between module type and attributes, between attribute in different module types and internal constraints between different attributes in the same module type. A module type is a model of the set of modules, which are interchangeable, perhaps with some restriction. But the problem on how to choose the exact modules for the specific customer requirements or configure product based on decomposing customer requirements is not solved. has developed an optimization method based on simulated annealing technique for the assemble-to-order manufacture paradigm. But they did not consider the constraints defined in the module and attribute and they did not formulate how to match modules efficiently when there are a large number of various modules in the module case database.

Modular product family

Module model and generic module

Based on analyzing function of products that share different functionalities, common functionality but different performance or different specifications, design for modular product decomposes the products into functional modules, and various products can be configured by choosing and combining the various functional modules to satisfy the various customer requirements in the market. The concepts of modules and modularity are the core in the description of a product family architecture and design for mass customization. While a module is a physical or conceptual grouping of components that share some characteristics, modular technology tries to separate a system into independent modules that can be treated as logic units (Newcomb et al., 1998). In module identification, interactions between modules should be minimized while the interactions of components within a module may be high (Felfernig, 2000).

Figure 1 illustrates the modular structure of a single product in a product family. Each tree has a root node denoting a certain product having some child nodes and child nodes denote modules. Some child nodes involve their child nodes further, but some other child nodes have none, which are called leaf nodes. On the design principle of product family and modularity, the product family is decomposed into a series of Generic Module (GM), based on analyzing the functionality of various products in the same product family. Figures 1 and 2 illustrate the product family which is represented as the root node of the tree. Decomposing the product family

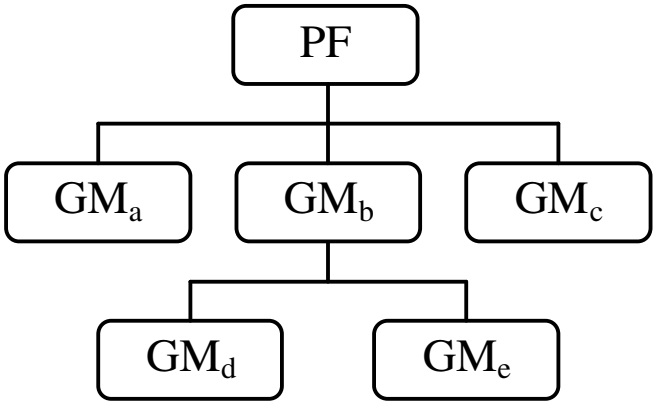

Figure 2. Generic module of product family.

into various GMs, in Figure 2, the root node has some child nodes which denote the GMs (the fist-class GMs), e.g. the GM_a, GM_b and GM_c.

Some GMs can be decomposed into lower-level of GMs, e.g. GM_b can be divided into GM_d and GM_e further. But some GMs cannot be divided which are denoted as leaf node, e.g. GM_a and GM_c. The grades of GMs should be reasonable, because too meticulous granular of GMs will cause problems of management and difficult combination of modules. But if the granular of the GMs is too rough, it will be unfavorable to match the modules with the customer requirements exactly. In Figure 2, the set of GMs in the product family can be denoted as $\{GM_a, GM_d, GM_e, GM_c\}$.

Each module case belonging to same generic module has similar attributes, similar structure, same interface but different attribute value. To keep consistency, the common module in product family can also be regarded as GM, but the GM involves only one module case, e.g. the GM_a only involves module a. The analysis of the functional requirement of the products is the reason to realize modular product family. Module is the unit of modular design and modular manufacture, so the various modules must satisfy the variety of customer requirements, convenient to manufacture and management. Different modules in the same GM should have unattached functionality, in order to achieve more agility, adaptability and no redundancy functionality while combining the modules.

If a module model (shorten as M) is established for each GM, each M can be regarded as a configurable unit. Modules derived from the same M belong to a certain kind of GM. The M can be described by multi-attributes, when some attribute can be evaluated by many values, the attribute can be treated as variable, i.e. attribute variable in module model, shorten as module attribute variable. The differentiations of the different modules in the same GM are derived from different values of partial or all module attribute variable.

Module attribute variable

For the specific modular product, value of each module attribute variable is certain. But the value of all module attribute variable of modular product family is not decided, only a domain corresponding to each attribute variable (Jørgensen, 2005). For some GMs, they involve only one module, e.g. GM_a, so the value of module attribute variable is certain. To keep consistency, however, we denote the corresponding module model with the module attribute variables, e.g. M_a is denoted by A_{a1} and A_{a2}, but there is only one element in the domain of each attribute variable.

In Figure 3, the module attribute variable of module model in the middle level can be denoted by the module attribute variable belonging to module in the lower level, but the attribute variable in the same combination is irrelevant, which means the random variable in the set cannot be reasoned by another variable in the same set. For Figure 4 as an instance, there are three module models, and M_1 is decomposed into M_2 and M_3. M_2 is denoted by attribute variable A_1 and A_2, M_3 is denoted by attribute variable A_3 and A_4, and the value of A_1 can be reasoned by the value of A_3, so the variable A_1, A_2, A_4 or A_2, A_3, A_4 is irrelevant. M_2 and M_3 constitute M_1, so the module attribute variable of M_1 can be denoted by the attribute variable of M_2 and M_3, but the variable A_1 and A_3 are relevant, so the attribute variable of M_1 is A_1, A_2, A_4 or A_2, A_3, A_4.

Sometimes, some attributes of certain module may be relevant to attributes of another certain module, so attributes of module are not independent with others, some attribute variables of certain module are the attribute variables of other modules at the same time. Because the attribute variable is pre-defined in product configuration, maybe they are available material parts or nonfigurative object or concept, e.g. attribute, character, etc. to express the performance and functionality of module or product, so some module or part can be substituted by some attributes. For example, whether a PC has video camera (here, video camera is an available attribute) or a PC can be set as a video camera (here, video camera is a module), they have the same meaning. So the course of product configuration is the mixed course of attribute definition and module selection to satisfy customer requirement.

Some attribute variables can express the module model, and all independent attribute variables can express the product family. For example, we can denote the product family in Figure 3 as $\mathbf{A}(MPF) = \{A_{a1}, A_{a2}, A_{d1}, A_{e1}, A_{e2}, A_{c1}, A_{c2}, A_{c3}\}$. When each attribute variable in the product family is assigned with a value subjected to constraints, then the specific product can be decided and configured. When each attribute variable of certain module model is assigned with a value subjected to constraints, then a specific module can be decided. For example, engine of automobile (that is, a module model of the automobile) can be expressed by

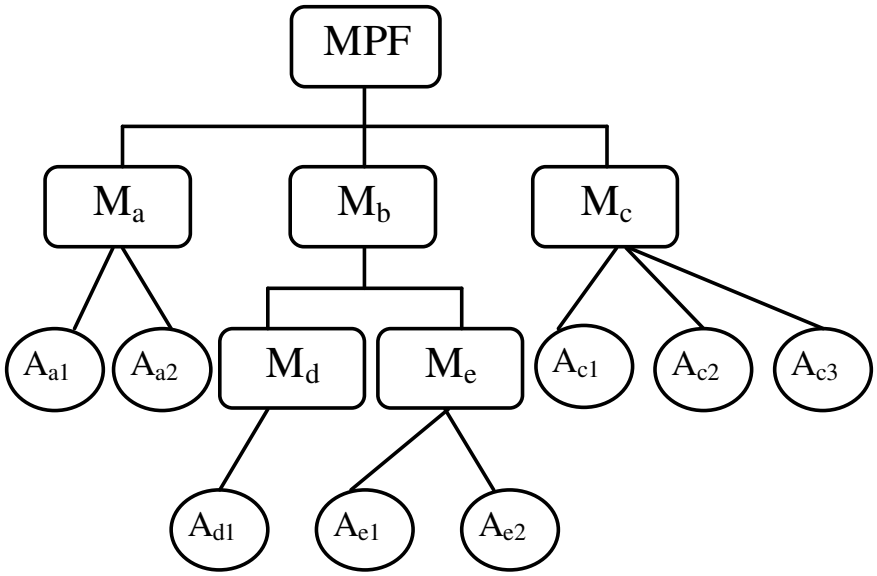

Figure 3. Module model of modular product family.

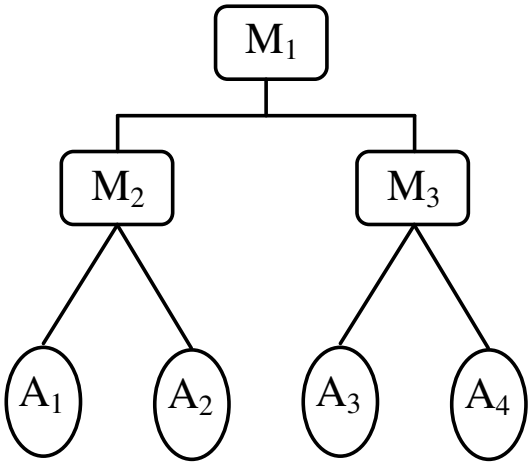

Figure 4. Reasoning of module attribute variable in the middle level.

attribute variables such as cylinder number of engine, array form of cylinder, delivery capacity, the number of valve, the maximum power output, etc. While each of them are assigned with an appropriate value subjected to some constraints and customer, then the specific engine of automobile can be decided and configured to satisfy customer requirements. Definition of the attribute variable should be simplified at all under the precondition of design, manufacture and assemble constraints. The domain of the attribute variable should be fixed according to customer requirement of the attribute variable and design specification of product structure.

Sometimes, as a result of the complex structure and functionality of product and module, we can set more attributes for the module in order to search the special module case matching customer requirements more exactly. According to the domain form of module attribute variable and customer requirement classification in the next section, that is, binary, optional, parameter (including continuous and discrete parameter), description and explanation type, where the description and explanation type is applied to search the product family matching customer requirement, we will classify the attribute variable as binary, optional, parameter (including continuous and discrete parameter). For example, the domain of the cylinder number of engine is $\{3,4,6,8,\cdots\}$ and the domain of color of automobile body is {white, black, blue, red,…}, so the cylinder number of engine is discrete parameter variable, and color of automobile body is optional variable, but the maximum power output is a continuous parameter variable.

In general, on the assumption that M denotes module model, so $\{M_1, M_2, \cdots, M_m\}$ can denote the product family. $M_i \ (i = 1,2,\cdots m)$ is a random module model in $\{M_1, M_2, \cdots, M_m\}$, and the set of attribute variable $\mathbf{A}(M_i) = \{A_j(M_i) | j = 1,2,\cdots, n_i\}$ express M_i. The number of attribute variables in each module model may not be equal, so n_i denotes the number of attribute variable in the i th module model. Analogously, the set of attribute variable $\mathbf{A}(MPF) = \{A_j(M_i) | i = 1,2,\cdots m; j = 1,2,\cdots, n_i\}$ can express the product family. If $a_j(M_i)$ implies value of the j th variable $A_j(M_i)$ of M_i, then $D_j(M_i)$ denotes the domain of $A_j(M_i)$. If $A_j(M_i)$ is a continuous parameter variable and $l_j(M_i), h_j(M_i)$ imply the minimum value and maximum

value of $D_j(M_i)$, respectively, then $D_j(M_i)=[l_j(M_i),h_j(M_i)]$, of course will restrict $l_j(M_i),h_j(M_i)$ to $l_j(M_i)<h_j(M_i)$.

Customer requirement translation

Customer requirement analysis

Customer requirement of product involves functionality, performance, appearance, price and dimension of the product, etc, that is to say, the customer requirement is denotation of unsolved precept in customer domain and it includes the essential characters of the product in need. In the course of product customization practically, for the difference in the customer themselves or the knowledge relative to product, different customers may choose different expression mode to represent their own requirement expediently.

For those customers of more product knowledge, they can represent their own requirement more exactly, because they can communicate with sale staff in technology traits and engineering characteristics, and those requirement information can be translated to value of module attribute variable directly, which is easy for mass customization enterprise to deal with, e.g. when the customer is planning to purchase a PC, they can speak out some engineering characteristic, such as type of CPU, capacity of memory, rotate speed of HD, and so on. But for some customers short of product knowledge, maybe it is hard for them to represent their requirement by choosing given options about technology traits or engineering characteristics, e.g. for they do not know the function and performance of characteristic parameter about CPU, memory, hard disk (HD), etc., so it is difficult for them to decide purchase correctly.

In the condition, the mass customization enterprise will allow them to represent requirements by some fuzzy or illustrative language, e.g. they can explain the main purpose of PC and the endured price to achieve the PC product customization, but these requirements information must be dealt with firstly and then translated to value of module attribute variable indirectly. In a word, we must classify the customer requirement in order to obtain and deliver various requirements effectively and exactly, i.e. the customer requirement can be classified as binary, optional, parameter, description and explanation type:

(1) Binary type: Binary type represents the additional function or attribute of configurable object, and formally there are two elements or values in the domain, that is, $\{1,0\}$,"1" imply the object in need and "0" has the opposite meaning, and the elements are mutually exclusive. That is to say, the function or attribute is dispensable in the view of customer. In the customization system, some available additional functions are provided to customer to finish their customization, e.g. the optional hardware: video camera, the customization system can set the option as {Yes, No} or {select, unselect}.

(2) Optional type: Optional type has defined available finite options, each of them can substitute for each other but mean different character or performance of product or parts, e.g. motor color may take on four states: red, yellow, green and blue. Of course, the option can be set as default value. The difference between optional and binary type is that optional type represents the replaceable function or characteristics but not additional function, e.g. the color of PC display appearance is {white, black, red}, but not {1,0} or {yes, no}.

(3) Parameter type: Parameter type is numeric, which can describe the quantity variety of component and numerical attribute. On the variety character, parameter type can be further classified into continuous and discrete type. The potential element of continuous type usually is a range, e.g. bigger than some value, smaller than a certain value, or between two different values, which can be decided by customer inputting the maximal and minimum value on the internet. For the elevator as an instance, the customer may require the operation speed to be between1.5-2.0 m/s. On the contrary, discrete type always indicates that the value is discrete real number, and the customer can input the value directly to deliver their requirement.

The binary, optional and parameter type are structured requirement information. It is difficult to represent requirement clearly only by the three type, so we need some other requirement type to help represent requirement, that is, description and explanation type, which allow customer represent requirement in nature language, and they are useful to represent expediently, they are semi-structured and unstructured requirement information respectively.

(4) Description type: The description type is semi-structured, and the customer is allowed to choose description language handily to describe attributes or function, that is, the customer can adopt fuzzy description semantic manner to illuminate the product elemental characteristics, which may be product function, part function, price, appearance, structure and technique parameter, etc. So this type can further be divided as functional description and technology description. Functional description not only includes requirement of application function, but also many functional characteristics, such as price, appearance, life span, color and so on. Technology description is to represent the structure and technology performance. The main fuzzy semantic language includes: size (bigger, smaller, as much as, higher, broader), color (partial red, darker, extreme dark, extreme light), and price (too cheap, about some value, too costly).

(5) Explanation type: The customer can represent the requirement by informal and natural language according to their own thinking. The type of requirement can be obtained by inputting requirement sentence in communicative interface on internet.

Product decision tree

Product decision tree illustration

When customer input requirements in online shopping malls and comparison sites on the Internet are shown, by referring to classified customer cluster, the requirements will be classified into similar customer cluster in order to search for a certain product family that can satisfy the customer requirements.

As well known to us, there are many attributes in product, such as function, performance, appearance, price, size and so on, and the character set representing a certain product family is different from another. For each attributes, there exist some levels of classification, and for each level, the classification is complete and mutually exclusive, e.g. Figure 5 displays the classification of price. So, it is reasonable to denote certain product kind as character set, where some of the characters are in common, and some others can be selected to differentiate each product family. Generally, the expert system in mass customization enterprise defines some characters that can distinguish product type distinctly, and establish a decision tree for product classification. For an enterprise of many product families, it is effective to differentiate each product family by decision tree. The decision tree is similar to a tree structure of flow figure, and each non-leaf node denote a test on the product attribute; besides, each branch denote a test result, i.e. the lower level node restricted to attribute value, and each leaf node denote a certain product family (shorten as PF). For the motor as an example, if the attribute of motor type and motor price (unit price: Yuan, in Figure 5 "(0-3)" denotes the motor price is between 0 Yuan and 3000Yuan, and "[3-4]" denotes the motor price is between 3000Yuan and 4000Yuan, etc.) can distinguish product type distinctly, and motor type is classified as three type, i.e. ride type, footplate type and crook type. Besides, the price can be divided into low grade, medium, and top grade. Based on the knowledge on customer psychoanalysis, the product can be classified from top to down, which is a clustering of distance decreasing. Here, on the assumption that the price hierarchy precedes motor type, that is to say, the price grade is on the first level of the decision tree, motor type is on the second level, and the third level is a certain PF, as illustrated in Figure 6. In decision tree, the classification on each level is complete and mutually exclusive.

Attribute ranking

For the difference of each attribute contributing to product sale, that is, when customer is purchasing the products, different customers may prefer different attributes, even for the same customer, he or she may prefer different attributes in different purchasing time. If the salesmen think the customer prefer the motor use and motor structure, then the grade of motor type precede price. Of course, the method of order ranking is subjective to some extent, so it is necessary to find out certain order ranking method fit for most customers.

Here, the information gain (IG) of each attribute is calculated and ranked in degrading order, and the attribute of maximal IG can discriminate the products to maximal extent. The bigger the IG of the attribute is, the more obvious the discrimination degree is, so the level of the attribute in decision tree is higher, i.e. it precedes more attributes.

If there are K attributes of $C_1, C_2, \cdots, C_K$, and $K < \sum_{i=1}^{m} n_i$, which can divide the products into K product families, i.e. $PF_1, PF_2, \cdots, PF_K$. If $PF_k (k = 1,2,\cdots,K)$ occurred s_k in history transaction, then the total number of this kind of products occurred is $S = \sum_{k=1}^{K} s_k$, for given sample classification, it needs expectation information is $I(s_1, s_2, \cdots, s_K) = -\sum_{k=1}^{K} p_k \log_2(p_k)$, here, p_k is the probability of random product k belonging to PF_k, which can be estimated by s_k / S.

Suppose the random attribute $C_k (j = 1,2,\cdots,K)$ has V_k different available values $\{C_k^1, C_k^2, \cdots, C_k^{V_k}\}$, and these V_k values are discrete otherwise the continuous value must be divided into exclusive range and become discrete. The attribute C_k can divide S into V_k subset $\{S_1, S_2, \cdots, S_{V_k}\}$, and in S_v $(v = 1,2,\cdots,V_k)$, there are some products which have C_k^v in attribute C_k. If C_k is a test attribute in decision tree, then its subset is corresponding to a sub branch of a node in decision tree and if s_{kv} equals the number of PF_k occurring in S_v, then the entropy of the subset divided by attribute C_j is $E(C_k) = \sum_{v=1}^{V_k} \frac{s_{1v} + s_{2v} + \cdots + s_{Kv}}{S} I(s_{1v}, s_{2v}, \cdots, s_{Kv})$, and $\frac{s_{1v} + s_{2v} + \cdots + s_{Kv}}{S}$ denote the weight. For each given subset S_v, $I(s_{1v}, s_{2v}, \cdots, s_{Kv}) = -\sum_{k=1}^{K} p_{kv} \log_2(p_{kv})$, here, $p_{kv} = s_{kv} / S_v$ means the probability of S_v belonged to product family PF_k. So the IG of the sub branch corresponding to attribute C_k is $Gain(C_k) = I(s_{1v}, s_{2v}, \cdots, s_{Kv}) - E(C_k)$. The attribute of maximal IG will be selected as the first level of test attribute in decision tree to classify the given products. With the IG decreasing, the attribute of less IG will hold the higher level in decision tree. So we can decide the product family subjected to each constraints of decision tree.

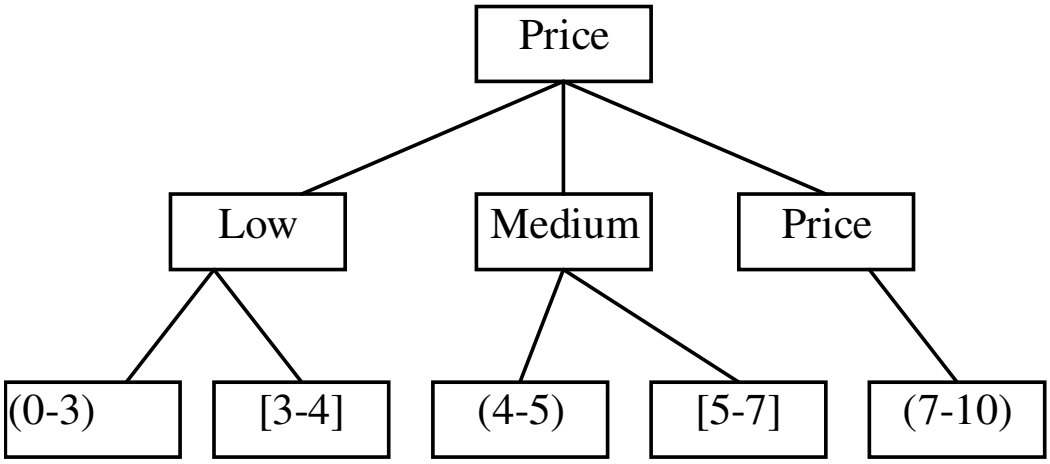

Figure 5. Price classification of motor.

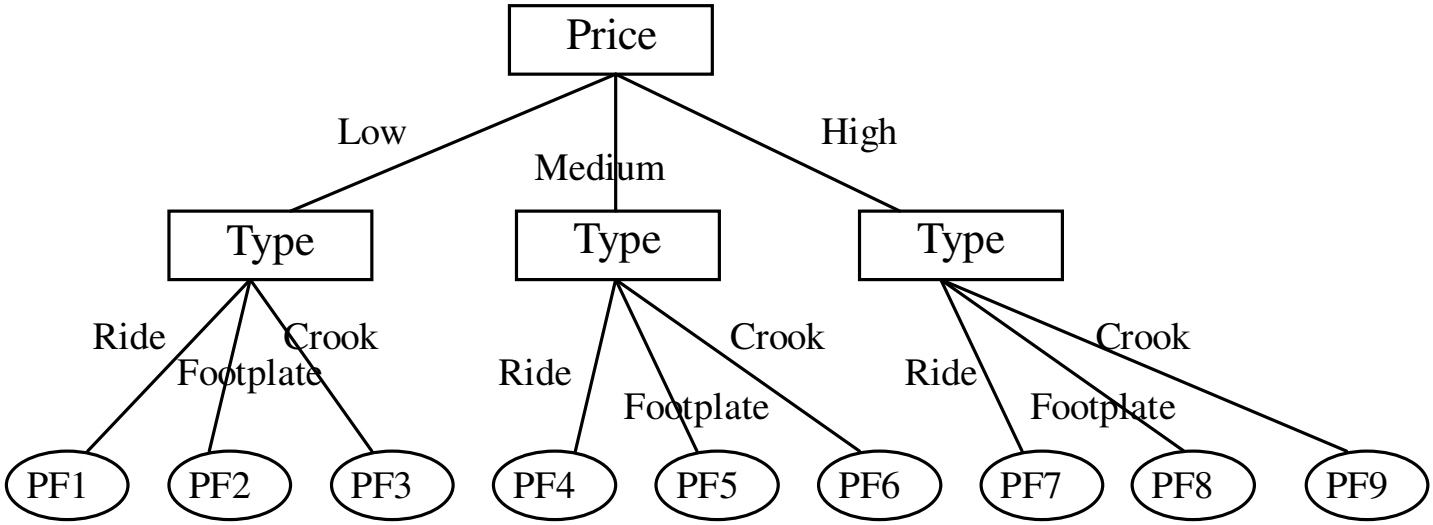

Figure 6. Product design tree of motor.

According to description and explanation type of customer requirement, we can cut the sub branch of decision tree to find out the certain product family satisfying customer. For example, if customer requires the motor of high-price and footplate type, then the motor in need can be limited in PF_8 according to product decision tree of motor in Figure 6, the scope of retrieving product shrinking. The binary, optional, and parameter type of customer requirement can be translated into the value of attribute variable according to the following product planning matrix and module deployment matrix.

For easy illustration, here, we assume that the final product family PF_k is selected according to description and explanation type of customer requirement, and PF_k is denoted as PF .

Translation of customer requirement to module attribute variable

As a matter of fact, the individual customer requirements of the product are the special request on the value of each attribute variable in all module models in a certain family. The system in mass customization must set up platform based on internet for collecting information on customer requirements (Lee et al., 2004). The customer requirements should be understood exactly at all, and they should be inducted appropriately.

After the customer requirements are restricted to certain available product family according to the description and explanation type of customer requirement, the binary, optional and parameter type of requirement should be translated into technique requirements of the

Table 1. Product planning matrix.

Customer requirements	Weight	Character matrix
		The target value of each technique requirement
		Importance of each technique requirement
		Each technique requirement

product, that is, the information about general technique character of the product, the certain technique requirements should be translated into value information about the attribute value of each module model. Here, the product in need for customer is called target product and analogously, the module that can be combined with other modules to satisfy customer requirements are called target module.

For modular configuration of product, the customer requirements should be analyzed in the first place. QFD (Quality Function Deployment) is a method to transform customer demands into design quality and it can be applied to deploy the functions forming quality and to deploy methods for achieving the design quality into subsystems and component parts, and ultimately to specific elements of the manufacturing process. So QFD can be seen as an important method for enterprise in mass customization to analyze customer requirements. When QFD is used to translate the customer requirements, there are some assumptions as follows:

1. There are no unreasonable or opposite requests in the customized customer requirements, but some relativity in the customer requirements is allowed.
2. The enterprise has implemented strategy of mass customization and has established complete modules case base or components base.
3. The cost for enterprise to analyze individual customer requirements by QFD is very low, to some extent, i.e. it can be regarded as zero.

By reference to idea of QFD, the mapping relation of customer requirements to technique requirements, and of technique requirements to module attributes can be established, and ultimately, the value and weights of attribute variable in each module model can be decided according to customer requirements. Now, we can illustrate product planning matrix and module deployment matrix, as shown in Tables 1 and 2.

In Table 1, the first row implies customer requirements listed out. $crw_i \quad (i = 1,2,\cdots, p)$ denotes weight of ith customer requirement for she/he deciding to buy the product, then the vector of weight of customer requirements is $(crw_1, crw_2, \cdots, crw_p)$, restricting crw_i to $\sum_{i=1}^{p} crw_i = 1$. Product planning matrix includes engineering character matrix (denoted by T) and relativity matrix (denoted by B).

$$\mathbf{T} = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1q} \\ t_{21} & t_{22} & \cdots & t_{2q} \\ \vdots & \vdots & & \vdots \\ t_{p1} & t_{p2} & \cdots & t_{pq} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1q} \\ b_{21} & b_{22} & \cdots & b_{2q} \\ \vdots & \vdots & & \vdots \\ b_{p1} & b_{p2} & \cdots & b_{pq} \end{bmatrix}$$

In Matrix T, p is the total number of items of customer requirements after analysis by experts, and q is the total number of general technique characters in the target product. t_{ij} denotes the value of j th technique character that can satisfy the i th customer requirement. The experts can analyze customer requirement by QFD and make sure the matrix T after all. In matrix B, b_{ij} implies the relative degree of i th customer requirement and j th general technique character. Let 9 denote “highly relative”, 3 denote “relative” and 1denote “lowly relative”. If there is no relativity between i th requirement and j th technique character, then $b_{ij} = 0$. The customer requirement can be translated into technique requirements in product planning matrix.

Matrix T indicates that individual customer requirements may cause different values for some general technique characters. So a general technique character may correspond to a scale but not a certain value. The paper adopts the weighted average to deal with the range in order to get a certain value for each technique character, and then the certain value replaces the range. By doing that, if t_j^* denotes the target value (that is, weighted average value) of the j th technique requirement, then $t_j^* = \sum_{i=1}^{p} crw_i \times t_{ij}$. As a result, vector of target value of technique requirements equal to $(t_1^*, t_2^*, \cdots, t_q^*)$. Besides, if tw_j denotes the important degree of the j th technique character, then $tw_j = \sum_{i=1}^{p} crw_i \times b_{ij}$ and vector of importance of technique character is $(tw_1, tw_2, \cdots, tw_q)$.

Table 2 displays a module deployment matrix, $(t_1^*, t_2^*, \cdots, t_q^*)$ and $(tw_1, tw_2, \cdots, tw_q)$ in the row which denote the vector of target value of technique requirement and

Table 2. Module deployment matrix.

Module model			M_1				…	M_i				…
Attribute variable of module			$A_1(M_1)$	$A_2(M_1)$	…	$A_{n_1}(M_1)$	…	$A_1(M_i)$	$A_2(M_i)$	…	$A_{n_i}(M_i)$	…
Technique requirement	t_1^*	tw_1	$\theta_{11}(M_1)$	$\theta_{12}(M_1)$	…	$\theta_{1n_1}(M_1)$	…	$\theta_{11}(M_i)$	$\theta_{12}(M_i)$	…	$\theta_{1n_i}(M_i)$	…
	t_2^*	tw_2	$\theta_{21}(M_1)$	$\theta_{22}(M_1)$	…	$\theta_{2n_1}(M_1)$	…	$\theta_{21}(M_i)$	$\theta_{22}(M_i)$	…	$\theta_{2n_i}(M_i)$	…
	⋮	⋮	⋮	⋮	…	⋮	…	⋮	⋮	…	⋮	…
	t_q^*	tw_q	$\theta_{q1}(M_1)$	$\theta_{q2}(M_1)$	…	$\theta_{qn_1}(M_1)$	…	$\theta_{q1}(M_i)$	$\theta_{q2}(M_i)$	…	$\theta_{qn_i}(M_i)$	…
The important degree of attribute variable of module			$mw_1(M_1)$	$mw_2(M_1)$	…	$mw_{n_1}(M_i)$	…	$mw_1(M_i)$	$mw_2(M_i)$	…	$mw_{n_i}(M_i)$	…
Weight of module attribute variable			$w_1(M_1)$	$w_2(M_1)$	…	$w_{n_1}(M_i)$	…	$w_1(M_i)$	$w_2(M_i)$	…	$w_{n_i}(M_i)$	…
Value of module attribute variable			$a_1(M_1)$	$a_2(M_1)$	…	$a_{n_1}(M_i)$	…	$a_1(M_i)$	$a_2(M_i)$	…	$a_{n_i}(M_i)$	…

important degree, respectively; on the assumption that M_i is a random module model in the modular product family and $\mathbf{A}(M_i)=\{A_1(M_i),A_2(M_i),\cdots A_{n_i}(M_i)\}$ denotes the set of attribute variables in M_i. Similarly to product planning matrix, module deployment matrix includes the Relativity Matrix of technique requirements and attribute variable in module model.

If $\theta_{xy}(M_i)$ is a random element in the relativity matrix of technique requirements and attribute variable, and $\theta_{xy}(M_i)(x=1,2,\cdots,q;y=1,2,\cdots,n_i)$ denotes the relative degree of the x th technique requirement and the y th attribute variable in M_i $(i=1,2,\cdots m)$. Similarly to Matrix B, let 9 denote “highly relative”, 3 denote “relative” and 1denote “lowly relative”. If there is no relativity between x th technique character and y th attribute variable in M_i, then $\theta_{xy}(M_i)=0$. When the important degree of the y th attribute variable in M_i is $mw_y(M_i)=\sum_{x=1}^{q} tw_x\times\theta_{xy}(M_i)$ $(y=1,2,\cdots,n_i)$, then $(mw_1(M_i),mw_2(M_i),\cdots,mw_{n_i}(M_i))$ can be normalized, and then $w_y(M_i)=\dfrac{mw_y(M_i)}{\sum_{t=1}^{n_i} mw_t(M_i)}$. So we can get the weights vector of each attribute variable of M_i, that si, $\mathbf{w}(M_i)=(w_1(M_i),w_2(M_i),\cdots,w_{n_i}(M_i))$. By reference to the corresponding relationship between technique requirements and module attributes which is prescribed in product design specification, the experts can make sure of the value of each attribute variable of M_i. The value vector of attribute variable is $\mathbf{a}(M_i)$, i.e. $\mathbf{a}(M_i)=(a_1(M_i),a_2(M_i),\cdots a_{n_i}(M_i))$. The value of some attribute variables can be computed out by the logic relation between different module models, between module model and attributes or between attributes in different module models. For example, there may be logic relation between cylinder number of engine (N) and delivery capacity (V), i.e. V=0.4N, and when customer requirement of cylinder number of engine (N) is 8, then delivery capacity (V) is 3.2. Of course, the logic relation between cylinder number of engine (N) and delivery capacity (V) represents the constraint in the same module model, so it is a local constraint. Using the above principle, value vector and weight vector of each module model can be calculated.

Module search

After we compute out the value and weight of all attribute variable in each module models, the mapping relation of target module (the module that can satisfy customer requirements) and the module case in the module base will established, that is, the similar module cases to target module will be searched in the module case base and these similar module cases belong to a GM. The similarity degree of module case and target module on the same module model can be computed quantitatively. The module case and target module belonging to same GM ensure that module case and target module have same attribute variables, so it is reasonable to compute the similarity degree of them. For the search space is restricted in generic modules, so it is helpful to reduce the search time and promote search efficiency. For example, the search algorithm on the module model M_i is as follows:

S**tep 1:** According to the requirement translation on the above, the customer requirement of the attribute variable of each module model can be gained. For random module model M_i, the set of its attribute variable is $\mathbf{A}(M_i)=\{A_1(M_i),A_2(M_i),\cdots A_{n_i}(M_i)\}$. Previously, customer requirement on the M_i can be translated into the value requirement of attribute variable, that is, $\mathbf{a}^r(M_i)=(a_1^r(M_i),a_2^r(M_i),\cdots a_{n_i}^r(M_i))$ and the weight vector $\mathbf{w}^r(M_i)=(w_1^r(M_i),w_2^r(M_i),\cdots,w_{n_i}^r(M_i))$, which denote the target module.

Step 2: If a module case u is a random case of GM_i and value vector of attribute variable is $\mathbf{a}^u(M_i)=(a_1^u(M_i),a_2^u(M_i),\cdots,a_{n_i}^u(M_i))$, $u=1,2,\cdots,v_i$, v_i denotes the number of module cases in GM_i,we search all the module cases of GM_i, i.e. compare the target module and each module case of GM_i;

Step 3: The similarity of the target module and each module case in GM_i can be computed out, i.e. $s^u(M_i)=1-\sum_{j=1}^{n_i} w_j^r(M_i)\delta_j^u(M_i)$, and $\delta_j^u(M_i)$ denotes some distance value relative to the type of attribute variable, i.e. binary, optional and parameter. $\delta_j^u(M_i)$ can be computed out by the following formulas:

1. If the attribute variable is binary type and $a_j^r(M_i)=a_j^u(M_i)$, then $\delta_j^u(M_i)=0$, else, $\delta_j^u(M_i)=1$.

2. If the attribute variable is optional type, then the distance between random two options will be provided by expert system. Here, the matrix $(\delta_{pq})_{Q\times Q}$ can denote the distance between them. $(\delta_{pq})_{Q\times Q}$ is a symmetrical matrix, i.e. $\delta_{pq}=\delta_{qp}$, Q is the number of option available to customer. δ_{pq} denotes the distance between the option p and option q, if $p\neq q$, then $\delta_{pq}\in(0,1]$; but if $p=q$, then $\delta_{pq}=0$.

3. If attribute variable is parameter type, then the distance is $\delta[a_j^r(M_i),a_j^u(M_i))=1-\frac{\min[a_j^r(M_i),a_j^u(M_i)]}{\max[a_j^r(M_i),a_j^u(M_i)]}$. We can estimate the similar degree of the target module and module case in GM_i according to $s^u(M_i)$ quantitatively. $s^u(M_i)=0$ means the module case of GM_i is different to target module absolutely, i.e. the module case cannot satisfy the customer requirement on M_i at all. $s^u(M_i)=1$ denote that the target module and module case in GM_i are the same, that is, the module case can fully satisfy customer requirements on M_i. If $s^u(M_i)$ is closer to 1, then it means that the module case is more similar to the target module, that is, the module case can better satisfy the customer requirement. When $s^u(M_i)$ is compared with ε, which denotes some degree of satisfaction, the candidate modules subjected to $s^u(M_i)\geq\varepsilon$ can be searched out. If the number of the candidate modules is z_i, then the set of candidate modules on M_i is $\{M_i^1,M_i^2,\cdots,M_i^{z_i}\}$.

For various and customized customer requirement, sometimes, the module cases cannot satisfy the individual customer requirement, i.e. there is no module case meeting $s^u(M_i)\geq\varepsilon$. So it is necessary to modify the module with maximum $s^u(M_i)$ or set up new module case to satisfy customer requirement on M_i and then store the modified or new case into module case base. The value combination of attribute variables is not at will, because there are some constraints on the different attribute variables and they must be subject to these constraints.

Step 4: According to above 3 steps, after computing the similarity degree of each module case in different GM and target modules and then filtering the set of candidate modules of each GM is $\{M_1^1,M_1^2,\cdots,M_1^{z_1}\}$, $\{M_2^1,M_2^2,\cdots,M_2^{z_2}\}$, …, $\{M_m^1,M_m^2,\cdots,M_m^{z_m}\}$.

Module combination is not only the simple integration of different module, which is subjected to some constraint. When the customer requirement is decomposed, the module which is most near or similar to corresponding customer requirement will be retrieved and combined to achieve product configuration. The simplest way to solve module combination is enumeration, i.e try all possible module combinations, and then find out the combination meeting all constraints and compute the similar degree of each combination and customer requirement, finally range the similar degree from big to small, the satisfactory configuration will be obtained based on the price requirement. But the method is only suit for the few number modules, when the number of candidate module is bigger, and the computation will become difficult and error-prone. Some scholars have introduced the genetic algorithm to solve the problem of module combination. The merit of genetic algorithm is the relatively small search space, because the algorithm can search more feasible solutions and approximate optimal solutions, thus the search time is reduced and the efficiency of the module combination is improved. But the disadvantage of the algorithm is that genetic algorithm cannot search the feasible solutions and approximate optimal solutions under the constraints of module and attributes.

As a matter of fact, product configuration for modular product is the course that customer evaluates the value of each attribute variable so as to choose exact modules to achieve combination. The module combination can be

regarded as a constraint satisfaction problem (Yao, 2004), that is, a generic module denotes a variable and the corresponding set of candidate modules denote the domain of variable.

Each element in the domain denotes a module case, the constraints on module models and attribute variable restrict the legal module combination. For the modular structure of product family, we can find out the module combination subjected to above constraints from down to top in the modular structure of product family. By doing that, we can simplify the problem, i.e. reduce time and promote the efficiency of configuration. The deposition of modular product family is from top to down, in contrary, the combination of modules in model of modular product family is from down to top, as illustrated in Figure 2. If the candidate module of M_c, M_d and M_e are $\{M_c^1, M_c^2, \cdots, M_c^{z_c}\}$, $\{M_d^1, M_d^2, \cdots, M_d^{z_d}\}$ and $\{M_e^1, M_e^2, \cdots, M_e^{z_e}\}$ accordingly, then we can combine the set $\{M_d^1, M_d^2, \cdots, M_d^{z_d}\}$ and $\{M_e^1, M_e^2, \cdots, M_e^{z_e}\}$ and check whether the combination is subjected to relative constraints. If the set $\{M_b^1, M_b^2, \cdots, M_b^{z_b}\}$ is the combination of modules derived from the set $\{M_d^1, M_d^2, \cdots, M_d^{z_d}\}$ and $\{M_e^1, M_e^2, \cdots, M_e^{z_e}\}$, then the modules $\{M_a^1, M_a^2, \cdots, M_a^{z_a}\}$ will be combined with $\{M_b^1, M_b^2, \cdots, M_b^{z_b}\}$ and $\{M_c^1, M_c^2, \cdots, M_c^{z_c}\}$. By the same reason, we will obtain the final modular product satisfying customer requirement. The candidate modules are subjected to some constraints relative to modules and attributes, in this way, many invalidated module combinations can be avoided and hence the efficiency of product configuration is promoted.

Applications to configuration of PC

This section will apply the above approach of product configuration based on modular product family to PC configuration. The main modules of a PC are display, hard drive, motherboard-assembly, and other accessories (such as mouse, keyboard, video camera and so on, where the video camera is an additional module). The motherboard-assembly is made up of motherboard, CPU, memory. The attribute variable of each module model is as following: (1) display: {size, color, type}; (2) hard drive: {HD capacity, HD interface, HD RPM}; (3) CPU: {CPU type, CPU frequency}; (4) motherboard: {motherboard type, CPU type}; (5) memory: {memory type, memory capacity, operating frequency}; (6) video camera: {lens material, CCD size, CCD pixels}. Because the motherboard-assembly is made up of motherboard, CPU, memory and there is a common attribute variable shared between motherboard and CPU, i.e. PU type, the attribute variable set of motherboard-assembly is {motherboard type, CPU type, CPU frequency, memory type, memory capacity, operating frequency}. The video camera in accessories is additional module, so a variable of binary type should be set for it. A simplified model of modular product family of PC is showed in Figure 7. The domain of the attribute variable in Figure 7 is illustrated in Table 3, while the constraints relevant to the attribute variable and module are listed in Table 4 respectively.

Now, a customer wants to purchase a PC with the price less than ￥4000 and no video camera, which is mainly used in mechanical drawing. He has non-mobile office need but he requires that the PC operates stably, besides, he wants to choose the CPU as series product of Pentium and he chooses the display of LCD and 17 inches, weight of these requirements are 0.5, 0.3 and 0.2. Here, the requirement “a PC with the price less than￥4000” is description type of customer requirement, “the PC operates stably” is explanation type of customer requirement, “no video camera” is binary type of customer requirement, and “he chooses the display of LCD and 17 inches” is the optional and continuous parameter type. In the case, the customer restricts his requirement in the products with prices less than￥4000 and CPU with Pentium, and then, it is easy for us to search the product family satisfying customer requirement.

In the structure of modular product family, the module models in the lowest level of tree are: CPU, HD, memory, display, mouse and accessory. We adopt the CPU, HD and memory to illustrate problem. The attribute variables of hard drive are HD capacity, HD interface and HD RPM; the attribute variables of CPU are CPU type and CPU frequency; the attribute variables of memory are memory type, memory capacity and operating frequency.

There are some constraints in PC configuration: {CPU frequency, RPM}={((2.8 GHz, 3 GHz) (5400,7200))};{CPU frequency, capacity of HD}={((2.8 GHz) (60 GB, 80 GB)) ((3 GHz) (80 GB))}; {CPU frequency, operating frequency} = {((2.8 GHz) (400 MHz)) ((3 GHz) (533 MHz, 667 MHz))}. So in the above 18 combination, there are 7 combination subjected to the constraints, that is to say, what we do is to find out the optimal configuration in the 7 configuration project. If we configure the product for customer directly after the requirement is obtained, then we must retrieve the configuration project from the 10×10×10=1000 module combination, but now, we can find out the configuration project from only 7 combinations. So, the efficiency of product configuration is improved obviously.

CONCLUSION

For various customer will adopt different method to express their own requirement, so it is important to classify customer requirements and translate the various requirements to attribute information of product module. This paper has classified the customer requirement as five types: binary, optional, parameter, description and

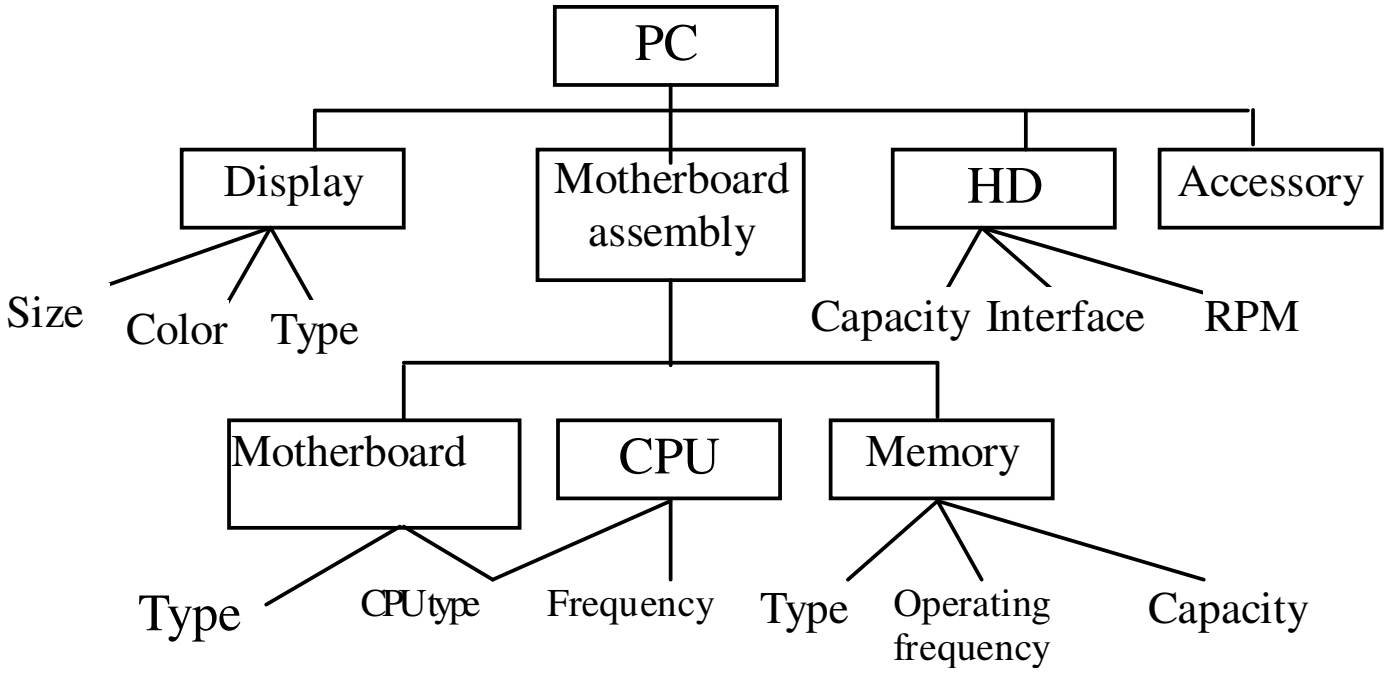

Figure 7. Modular structure of product family of computer.

explanation. There are many attributes in product, such as function, performance, appearance, price, size and so on, and the character set representing a certain product family is different from another. Generally, the expert system in mass customization enterprise defines some characters that can distinguish product type distinctly, and establish a decision tree, which is effective to differentiate each product family by decision tree. According to description and explanation type of customer requirement, we can cut the sub branch of decision tree to find out the certain product family satisfying customer, so the scope of retrieving product is shrinking.

Based on analyzing function of products that share different functionalities or common functionality but different performance, modular structure of product family is established and the product family is decomposed into generic modules. Then, the module model represented by attribute variable is established for each generic module. By reference to QFD, the binary, optional, and parameter type of customer requirement can be translated into the value of attribute variable, that is, the mapping relation of customer requirement to technique requirement, and continuously, the technique requirement to module attribute can be achieved based on the product planning matrix and module deployment matrix, and ultimately, the value and weight of each attribute variable in all module models can be decided according to customer requirements. After searching the candidate modules set which are nearest to the customer requirement on the module model accordingly, the similarity degree of module case and target module on the same module model can be calculated quantitatively. The module case and target module belonging to same GM ensure that module case and target module have same attribute variables, so it is reasonable to compute the similarity degree between them. For the search space is restricted within generic modules, so it is helpful to reduce the search time and promote efficiency. The candidate modules are combined efficiently under the constraints relative to module models and attributes. In this way, many invalidate module combinations can be avoided and hence the efficiency of product configuration is promoted.

Although we classified the various customer requirement, but how to process natural customer language online and how to apply virtual reality technology and communication tool to obtain customer requirement will be our future research direction. In our work, when we translate customer requirement to technique requirement or value of module attribute variable, which is set as a real number, but in practical product family layout, technique requirement or value of module attribute variable is always a range, rather than a single value. So it is difficult to apply module deployment matrix to compute the value range of variable directly. It is worth to study the optimal decision model about product planning matrix and module deployment matrix.

ACKNOWLEDGEMENT

The research is supported by Chongqing Education Commission under Grant Number KJ100412.

Table 3. Module attributes and value domain.

Attribute variable		Domain
Display	Size	14 inche, 15 inche, 17 inche, 18 inche, 19 inche, 20 inches
	Color	silver, black, white, green
	Type	CRT, LCD
Hard drive	Capacity	60 GB, 80 GB, 160 GB, 250 GB
	Interface	ATA, ATAPI-IDE
	RPM	5400 RPM, 7200 RPM
Motherboard type		M2N-E, P5PL-E, P5PB, P5LD2-SE
CPU	Type	Pentium4, Pentium D, Intel Core2
	Frequency	2.8 GHz, 3.0 GHz, 1.5 GHz
Memory	Type	DDR, DDR Ⅱ
	Operating frequency	266 MHz, 333 MHz, 400 MHz, 533 MHz, 667 MHz, 800 MHz, 1066 MHz
	Capacity	128 MB, 256 MB, 512 MB, 1024 MB

Display type, display color, HD type, memory type are optional variable; display size, HD capacity, memory capacity are continuous parameter variable; HD RPM, CPU frequency, operating frequency are discrete parameter variable.

Table 4. The relevant constraint to attribute variable.

Attribute variable	Constraints
CPU frequency, HD RPM	((2.8 GHz, 3 GHz)(5400, 7200))
CPU frequency, HD capacity	((2.8 GHz) (60 GB, 80 GB)) ((3 GHz)(80 GB))
CPU frequency, operating frequency	((2.8 GHz)(400 MHz)) ((3 GHz)(533 MHz, 667 MHz))
CPU type, CPU frequency	((Pentium D915)(2.8GHz))((Pentium D925, Intel Core2)(3GHz))
Motherboard type, CPU type	((P5PL-E,P5PLD-SE))((P5PGZ-MX)(Pentium D, Pentium 4,Intel Core2))
Memory type, capacity	((DDR)(512MB)(400MHz)) ((DDR Ⅱ)(512MB)(533MHz))((DDR Ⅱ)(1024)(667MHz,80MHz))
HD capacity, interface, RPM	((60GB,160GB,)(5400)(AIAPI-IDE)) ((80GB)(7200)(AIAPI-IDE))((80GB)(5400)(ATA))
Display size, type	((CRT)(15 inche,17 inche,19 inche,20 inches))((LCD)(15 inche,17 inche,18 inches))

Constraint 1, 2 and 5 denote the relation between attribute of different modules, and constraint 3, 4, 6, 7 and 8 are the relation between different attribute of same module.

REFERENCES

Felfernig A, Friedrich G, Jannach D (2000). Exploiting structural Abstractions for Consistency Based Diagnosis of Large Configuration Knowledge bases [J]. Proceedings of the Applications of prolog 14th international conference on Web knowledge management and decision support.

Pine BJ (1993). Mass Customization: The New Frontier in Business Competition. Harvard Business School Press, Boston, USA.

Dan B, Qin YH, Wang JP (2008). Product configuration method based on an authentic real case and a constraint satisfaction problem. J. Chongqing Univ. (Natural Science Edition), 31(5): 511-514.

Dan B, Wang JP (2008). Taxonomy model and representation approaches of customer needs information for mass customization. Comput. Integrated Manuf. Syst., 14(8): 1504-1511.

Liu F, Li SX, Liu S (2004). Customer requirement collaborative optimization between customer and enterprises for mass customization and the application. Chinese J. Mech. Eng., 40(1): 113-117.

Li GX, Ci YZ (2004). Study of Module Solution for Product Configuration. J. National Univ. Def. Technol., 26(5): 85-89.

Zheng HL, Liu F (2006). Process optimization model Oriented to integrated control of CAPP/PPC. China Mech. Eng., 12(8):1188-1191.

Fruto JD, Borenstein D (2008). A framework to support customer-company interaction in mass customization environments. Comput. Ind., 54: 2.

Jiao J, Tseng MM (1999). A methodology of developing product family architecture for mass customization. J. Intell. Manuf., (10): 3-20.

Fujita K (2002). Product variety optimization under modular architecture. Computer - Aided Design, 34(12): 953 -965.

Jørgensen KA (2005). Product configuration and product family modeling. Working Paper, 2005.

Su LY, Dong M (2010). Service product configuration of mass customization based on ontology. Appl. Res. Comput., 2: 483-487

Newcomb PJ, Bras B, Rosen DW (1998). Implication of modularity on product design for the life cycle. J. Mech. Des., Trans. ASME, 120(3): 483-490.

Yan ST, Wu WJ, Huang HX (2010). The recognition method of customer's demand based on matter-element theory. Sci. Technol. Eng., 10(2): 582-585.
Lee SK, Lee JK, Lee KJ (2004). Online customization with configurable standard models. Proceedings of the 6th international conference on Electronic commerce. New York, pp. 419 – 428
Yao XH (2004). Constraints programming and application in product configurators. Comput. Appl. Software, 21(3): 36-37.
Wang Y, Lin J (2010). Research on mass dynamic customization system. Comput. Eng. Appl., 4: 193-196.