

Q. L. Xu · S. K. Ong · A. Y. C. Nee

Function-based design synthesis approach to design reuse

Received: 25 September 2005 / Revised: 21 December 2005 / Accepted: 22 January 2006 / Published online: 21 April 2006
© Springer-Verlag London Limited 2006

Abstract In this paper, a design reuse framework with a function-based design synthesis approach is proposed in the context of conceptual product development. Previous researches in design reuse have lacked a comprehensive functional base for knowledge representation and reasoning. The method presented in this paper uses a function-based product information model and a multiple objective optimization model to achieve design reuse. The information model is dependent on a functional core which is the key element vector. It is capable of modeling product information with sufficient abstraction, which in turn facilitates intelligent construction of product platforms. The multiple objective optimization method carries out automated design synthesis and evaluation subject to various design constraints. The approach has been applied in the design of the fan filter unit, a key clean room device. It has achieved intelligent design reuse in product conceptual design with significant rapidity and solution variety.

Keywords Design reuse · Design synthesis · Function · Multi-objective optimization

1 Introduction

Design reuse plays a vital role in product development. It is an experience-oriented approach and has long been adopted by designers, consciously or sub-consciously. According to the statistics in industries, only about 20% of OEM's investment is on new design while about 80% is on the reuse of existing products, with or without modification (Rezayat 2000). Although direct reuse of previous design components is not applicable in most

design projects, it is a common practice for a designer to resort to similar past designs as a starting point, even in the original design. On the other hand, conceptual design has been considered as the most important stage in the whole product development cycle. Using design reuse methodologies to facilitate product conceptual design is a notable investigation topic. However, considerable effort is needed to overcome the difficulties in the process. It is recognized that the following factors are vital to the success of design reuse in conceptual design, namely, (1) methods to handle the complexity and volatility of the conceptual design knowledge, (2) fast and robust methods for design synthesis and evaluation, and (3) the adaptation to diverse design objectives and constraints.

This paper proposes a function-based design synthesis approach to support conceptual design. The approach is implemented in an integrated design reuse framework, which helps to achieve intelligent building of product structures. The paper is organized into eight sections. After an overview of the related work in Sect. 1, a design reuse framework is presented in Sect. 2. Section 3 focuses on the function-based product modeling scheme. Section 4 presents the function analysis method using neural networks. It also discusses the mapping from the key characteristics (KCs) to the design parameters. A design synthesis method using multi-objective optimization is proposed in Sect. 5. Section 6 illustrates the method with a case study. Sections 7 and 8 present the discussions and conclusions, respectively.

Product information takes various forms and is subject to changes. To achieve effective design reuse, the product model must capture the multiple aspects of the product information. Various generic modeling languages have been employed in engineering design, such as the Unified Modeling Language (UML) (Pulm and Lindemann 2001), the Standard for the Exchange of Product Model Data (STEP) (Pratt and Anderson 2001; Szykman et al. 1998), etc. Bobrow et al. (1996) developed the Compositional Modeling Language (CML) to support model sharing and interchange between differ-

Q. L. Xu · S. K. Ong (✉) · A. Y. C. Nee
Mechanical Engineering Department, Faculty of Engineering,
National University of Singapore, 9 Engineering Drive 1,
Singapore, Singapore 117576
E-mail: mpeongsk@nus.edu.sg
Tel.: +65-6516-2222
Fax: +65-6779-1459

ent research groups. CML provides common syntax with well-defined semantics so that it can model a broad variety of physical processes and objects. However, information processing on basis of CML is not straightforward due to its generic structure. In product design, it is preferable to focus on the products instead of the ‘entities’ or ‘physical phenomena’ that are highly abstract.

Stone and Wood (2000) proposed a function basis method for product design. They developed a formalized function representation using functions and flows, as well as taxonomies. The function basis has been motivated by several factors, such as the product architecture design, the storage and transmission of information, creativity in concept generation, etc. This function basis provides a comprehensive representation scheme. However, a clear scenario of reusing the function information has not been presented, i.e., the use of the function basis with respect to the proposed motivations has not been illustrated.

A function basis is helpful to establish product architectures from which the product configurations are generated. Quantitative and heuristic methods have been proposed to identify product architectures based on the product function (McAdams et al. 1999; Zamirowski and Otto 1999; Stone et al. 2000a, b). However, the process still relies heavily on human designers. It is desirable to achieve a more intelligent analysis of product functions to build the product architecture.

The NIST design repository project is aimed at developing a computational framework for creating design repositories (Szykman et al. 1998, 1999, 2001). This research presents a generic product information model that is characterized by a formalized function-flow representation schema. Although the interoperability of the information model and the role of knowledge are emphasized, the research does not provide a comprehensive solution to design synthesis. Knowledge reuse is largely restricted to conventional case-based reasoning (CBR).

In this research, design synthesis is limited to configuration design, which refers to the generation of product structures from the combination of design components. Mittal and Frayman (1989) defined the general configuration task and proposed methods of problem solving. The definition of configuration task can help a designer to formulate the design problem in a simplified form. However, problem solving may need to be supported by more efficient computational methods. Yu and MacCallum (1995) proposed a knowledge-based system for product configuration design and management. Efforts have been made to (1) capture product structure knowledge, (2) create product structure using the reason maintenance system, and (3) maintain configuration consistency using the inference engine. However, it is difficult to establish the knowledge base that is required in this system. Moreover, solutions are created through the interactions between the designer and the decision support system. The effectiveness of the method in solving large combinatorial problems was not validated.

A-Design is an agent-based system that combines multi-objective optimization with a multi-agent system for automated design synthesis (Campbell et al. 1999, 2000). It is also capable of accepting changing design inputs and decision-making based on previous experience. Such capacities make the system intelligent and adaptive to the dynamic environment. However, one limitation of this system is its dependence on the function parameters (FPs), which relies heavily on the input–output relations of design variables, usually in the form of equations. This limits the flexibility to build the product architectures. Moreover, the design components in A-Design were hard-coded in a number of Lisp files making it difficult to update the relevant information.

Chakrabarti and Bligh (1996) proposed a component-based catalog design method that emphasizes the representation and abstraction of structural elements to generate concepts. The authors attempted to abstract the mechanical components as input–output elements. Machine design is carried out through composing the elements based on the input–output specifications using an exhaustive search. The method is similar to the agent-based approach. However, it lacks the optimization scheme that is present in A-Design.

Studying the work presented earlier, there is a common deficiency in establishing a unified approach to combine a formalized function representation scheme with an intelligent design synthesis method that accommodates various design objectives and design constraints. This paper proposes an integrated approach to address the above deficiency.

2 A design reuse framework for conceptual design

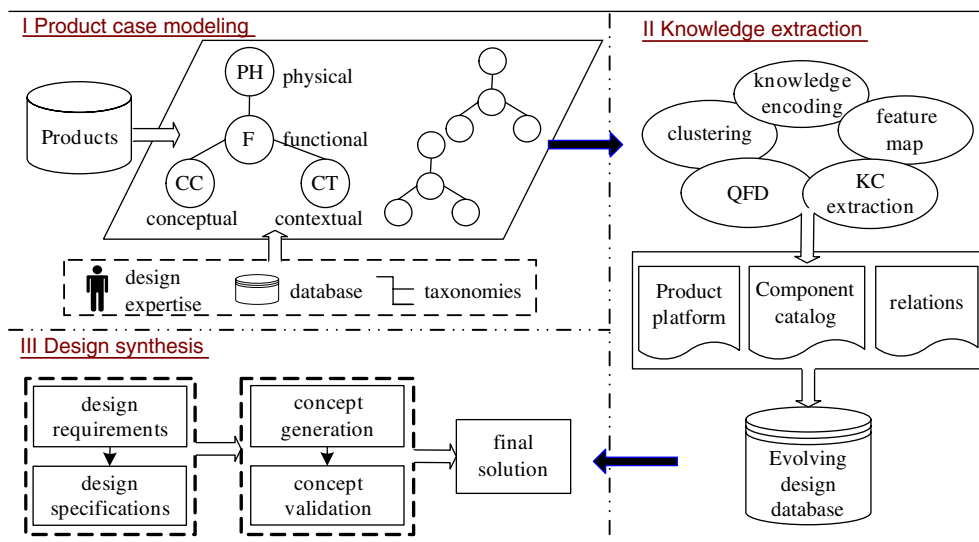
This section presents a design reuse process model for conceptual design. It involves three major processes, namely, product information modeling, knowledge extraction, and design synthesis. Figure 1 shows the system architecture of this model.

In the first process, product information is accumulated by means of product modeling. Product information is collected within a virtual space, i.e., a space that contains a particular category of products. The central part of the process is a function-based product information model, which will be elaborated in Sect. 3.

The next process is knowledge extraction. Here, information collected in the first process is analyzed using predefined algorithms. In particular, a product platform is constructed using neural networks; correlation between the KCs and the platform components are established using the quality function deployment (QFD) method.

The design synthesis deals with the design-by-reuse process. Initially, a design task is put forth as a list of design requirements, which are in turn interpreted as design specifications. These specifications serve as objectives and constraints for solution synthesis. Final

Fig. 1 System architecture of the design reuse process model



solutions can thus be put forth by combining and reusing existing product components.

Following this process model, Sects. 3–5 discuss the steps in detail. To demonstrate the processes, a fan filter unit (FFU) is used as the design case throughout this paper. The FFU is the key device in the clean room products. It draws in air from outdoor space, removes the unwanted particles, and supplies clean and laminar airflow continuously into a clean room. Figure 2 illustrates the structure of a horizontally mounted FFU. The performance requirements of the FFU are rigorous and diverse with respect to different applications. In addition, a customer may have individual demands, such as a special size or a specific material. Therefore, the FFU manufacturers have to be able to provide customizable, low-cost products, while conforming to various industry standards. Traditionally, the FFU design is largely dependent on a designer’s expertise, with the support of computer-based or paper-based design catalog. In comparison with the method presented in this research, this traditional method lacks a formalized design reuse foundation. Automated development of solution alternatives at the early design stage is rare; let alone a systematic estimation of the design superiority.

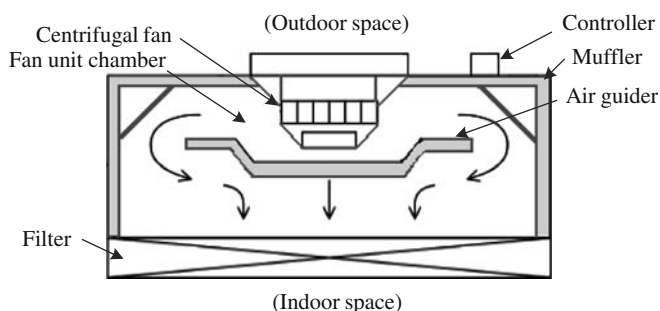


Fig. 2 Fan filter unit (FFU) structure and major components

3 Function-based product modeling

3.1 Function-based product information model

In this research, the product information model is defined as a triple $p(p_{at}, p_{kc}, p_{ref})$, where p_{at} contains a few general attributes of a product, such as a product model, manufacturer, etc., p_{kc} defines a set of KCs, and p_{ref} contains the reference to four predefined information facets. The structure of the information model is shown in Fig. 3.

The KCs are qualitative or quantitative customer requirements that are translated to support engineering specifications for product design (Rezayat 2000). They are used to limit the scope of the problem by focusing on a few crucial business or customer factors. In this

Product- p

General attributes- p_{at}

Id: integer;

Name: String;

Description: String;

ref_FLSP: TFlatSpace //space a product belonging to

...

KCs- p_{kc} //defined according to particular product type

{kc_name: String;

kc_value: String;}*

Reference- p_{ref}

ref_Physical: TPhysical

ref_Functional: TFunction

ref_Conceptual: TConcept

ref_Contextual: TContext

*: ‘{ }’ denotes a set of data

Fig. 3 Definition of product information model

research, the KCs of a product refer to the engineering metrics that signify the major performance features of the product. They are defined for a category of similar products. For example, the KCs for the FFU include: the power type (AC/DC), static air pressure (Pa), air quantity (m^3/h), air velocity (m/s), noise level (dBA), etc. A detailed list of the KCs of the FFU is presented in Sect. 4.

The four information facets are defined as the functional, physical, conceptual, and contextual facets. Each facet has a pre-defined data structure. The purpose and content of each facet are listed in Table 1. Among them, the functional facet serves as the core facet.

3.2 Function representation with KEV

A formalized representation of function is required for computation. This research adopts a flow-oriented, hierarchical function structure, similar to the model presented by Pahl and Beitz (1996). It involves function description on a micro- and macro-level. The micro-level description is a function block (FB), while the macro-level structure deals with the overall organization of the function hierarchy. An FB defines a function in terms of the input/output flows and the function actions (FAs), i.e., the input flow is transformed into the output flow as the result of the FAs (Pahl and Beitz 1996). A flow can be one of the three forms, namely, energy (E), material (M) and signal (S), while an FA is usually in the form of a transitive verb. Based on these definitions, an FB is described as follows, where [] denotes optional object(s):

transitive verb(s)
 + [source object(s) (i.e., the input flow)]
 + [destination objects(s) (i.e., the output flow)].

For example, the FFU involves a function of producing air flow with a certain flow rate. An ‘air flow generation’ function may involve ‘converting’ the input energy ‘electric voltage’ to an output energy ‘velocity’ of the air (Fig. 4). As a result, the function can be represented as *convert electric voltage to air velocity*.

A complex product function is usually decomposed into several sub-functions. A hierarchical function structure is obtained as the result of the decomposition.

The decomposition stops when each of the leaf functions can be fulfilled by a single, basic solution principle. Such a leaf function is called an atomic function. Furthermore, it is assumed that the characteristics of non-atomic functions can be defined by the complete set of its descendant atomic function blocks. Figure 5 is a typical function structure of FFU.

The atomic functions are formally represented by three main attributes, namely, the input-flow (I), output-flow (O) and FAs (A). Each FA, in combination with the corresponding input/output flows (E, M, or S), constitutes a key element f_i . Thus, the key element is represented as a vector:

$$f_i = [A_i \quad I_i \quad O_i]$$

where A_i is the FA, I_i is the input flow(s) and $I_i \in E \cup M \cup S$, and O_i is the output flow(s) and $O_i \in E \cup M \cup S$.

If a function involves multiple FAs and multiple key elements, the FB is represented as a key element vector (KEV), which is a combination of these elements:

$$F = [f_1 \quad \cdots \quad f_n]^T = \begin{bmatrix} A_1 & I_1 & O_1 \\ \cdots & \cdots & \cdots \\ A_n & I_n & O_n \end{bmatrix}$$

where n is the total number of FAs. Usually n should not be a large value because in such a case, the function can be further decomposed into sub-functions. As an example, the ‘air flow generation’ function can be represented as a one-element KEV $F = [f_1]^T = [\text{convert voltage:air velocity:air}]$.

The elements in a KEV are non-numeric, and as such they are not suitable for computational analysis. The solution to this problem involves two steps: firstly, developing taxonomies with proper coding schemes; and secondly, mapping the categorical FAs and flows to quantitative values using these coding schemes. In this research, taxonomies are extracted from the researches reported by Szykman et al. (1998), Pahl and Beitz (1996), Kirschman and Fadel (1998), and Otto and Wood (2001). They are further coded such that an FA is represented by a unique three-digit code and a flow is represented by a unique four-digit code. For example, the coded form of the ‘air flow generation’ is $F = [302 \quad 1202 : 2203 \quad 1163 : 2203]$.

Table 1 A summary of the four facets of the product information model

	Purpose	Content
Functional	Describes the function of a product; an abstraction of design concepts	<i>Structure</i> : the hierarchical structure from function decomposition <i>Function block</i> : elements of the function structure
Physical Conceptual	Presents a visual and editable information entity Depicts how a function is achieved	<i>Form</i> : a geometric model (preferably CAD model) <i>Solution principles</i> : embodiment of the working principles of various components with respect to the functions
Contextual	Set the products in the evolving context	<i>Design history</i> : conceptual component reuse and evolution records

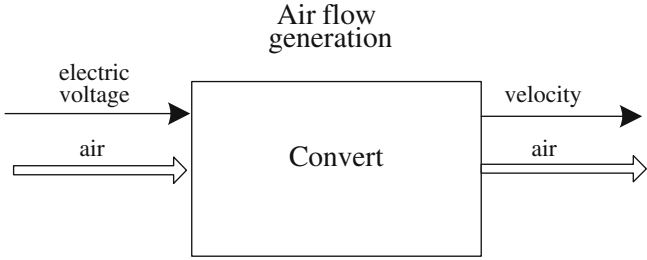


Fig. 4 A function block: air flow generation

The taxonomy scheme used in this research can support a broad variety of product functions. However, the taxonomy is not necessarily complete as some functions may not be properly defined. Two methods are used to solve this problem. Firstly, the system accepts user-defined functions to allow for certain level of ambiguity. Secondly, the taxonomy can be updated to include new vocabularies that are constantly used. Another problem is that the taxonomy scheme is not strictly orthogonal, i.e., the function models are not free of inconsistencies because different designers may use different vocabularies to describe the same function. To alleviate this problem, the function modeling is supported by interactive graphical interface with rich guidance.

This section presents a function-based product information model. It is developed to provide a formal and comprehensive representation scheme to capture the essential product information. This model is generic in that it is not restricted to specific types of products and it is not limited to specific aspects of product information. However, the function base used in this research may be restrictive in dealing with the information that is not functional in nature. The solution to this problem is to include such information in the other information facets, e.g., physical and contextual facets. In comparison,

some general modeling languages, such as CML, provided more generic structure to represent various types of entities and processes. This research did not adopt CML because its generic structure is not directly relevant to product design. Various extensions to the base language have to be made to accommodate the requirements of the domain-specific representation (Bobrow et al. 1996).

4 Knowledge extraction approaches

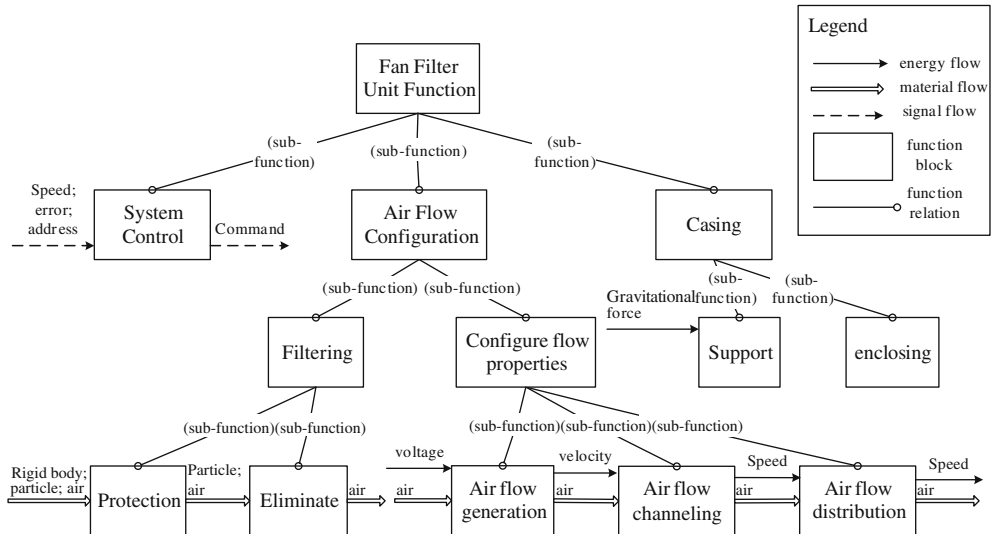
4.1 Mapping from design requirements to design components

Typically, design is the process of finding the proper design parameters to fulfill the design requirements. Therefore, the relationships between the design requirements and the design parameters are essential to any design system. Such relationships can be broadly considered as a mapping route from the design requirements to the design components. This section presents methods to identify the mapping route.

The process is achieved using the domain mapping method similar to the zig-zag decomposition in axiomatic design (Suh 2001). Three transformations are involved to correlate the design requirements with the final conceptual components. The transformation matrix is similar to the one developed in axiomatic design (Suh 2001), but without the constraint of uncoupled/decoupled design. The process is illustrated in Fig. 6.

Let R, K, F, C represent the vectors of customer requirements, KCs, function groups (FGs), and conceptual components, respectively. Among them, the FG is a mechanism to organize the product functions. It can be obtained using the function analysis method to be presented in Sect. 4.2. The conceptual components are defined as the conceptual facet in the product

Fig. 5 Function structure of FFU



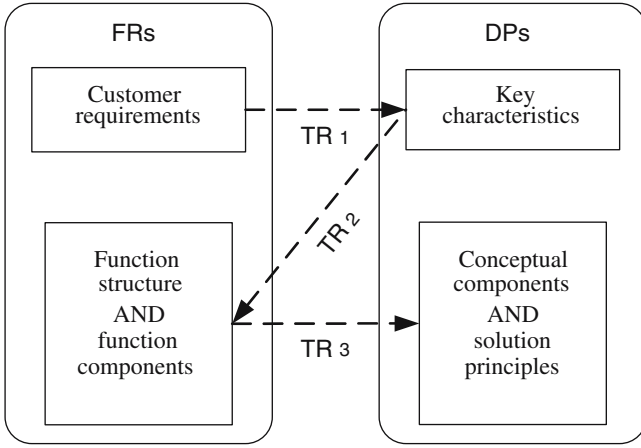


Fig. 6 Mapping from design requirements to solution principles

information model discussed earlier. For each product case, it is assumed that a key conceptual component can be identified corresponding to a functional component (Mittal and Frayman 1989). Thus, a component catalog can be obtained through the decomposition of the product cases. Each function can be instantiated by a set of components, for which the attributes are different. In the design synthesis processes, the outcome is a sensible combination of conceptual components that are selected from the component catalog. TR_1 , TR_2 and TR_3 are the mapping matrices. Equations 1–3 present a mapping route from the customer requirements to the design conceptual components:

$$R = TR_1 \times K, \quad (1)$$

$$K = TR_2 \times F, \quad (2)$$

$$F = TR_3 \times C. \quad (3)$$

The first transformation is needed when the customers have a set of general requirements to be measured quantitatively with design specifications. The transformation can be performed manually using the QFD method (Martin and Ishii 2002). Using the FFU as an example, the customer may define the design requirements to be ‘low energy usage’, ‘low noise’, and ‘light weight’. These requirements can be addressed by KCs such as ‘power consumption’, ‘efficiency’, ‘noise level’, and ‘product weight’. Once the KCs have been obtained, they will be used to specify the design objectives and set the constraints. Next, the KCs are mapped onto the function domain using TR_2 , which is the correlation of KCs and FGs. TR_2 is established using a second QFD phase. As will be discussed shortly, seven FGs are formed for the FFU using the function analysis method. The correlation between the KCs and the FGs is shown in Table 2. Lastly, the FGs are correlated to conceptual components through a third transformation. As will be discussed in Sect. 4.2, TR_3 is a natural outcome of the function analysis.

4.2 Product platform building using function analysis

4.2.1 Introduction of feature map and the self-organizing map (SOM) method

The pre-requisite of function analysis is a set of products whose function structures have been built according to the functional model. Next, all the FBs within the function structures belonging to different products are retrieved. The FBs are represented using the coded KEV. The argument is that the attributes contained in the key elements can be considered as coordinates that collectively constitute a multi-dimensional discrete space. Hence, the FBs can be viewed as data points distributed in this space. If N attributes are used to represent the FBs, the FBs can be considered as data points in an N -dimensional space. This concept is illustrated with a 2D plane in Fig. 7. The FBs belonging to a set of related products are scattered in this space (the shaded circles). It is expected that similar FBs from different products are close to each other and intrinsically fall into a specific group (the ellipse). However, these groups are ‘invisible’ to a designer before any computational analysis. In this research, the self-organizing map (SOM) is used to perform an unsupervised clustering to find the grouping patterns. These patterns stand for a preliminary functional product platform.

A typical SOM network structure consists of three layers, namely, the input layer, the competitive layer, and the output layer (Fig. 8). The input layer accepts a multi-dimensional data pattern, which is usually represented as a vector. In the function analysis problem, the input data vector is derived from the KEV. Seven nodes are used to represent seven elements, namely, function action, input energy flow, input material flow, input signal flow, output energy flow, output material flow, and output signal flow. Table 3 shows an example of the input vector for the function ‘air flow generation’.

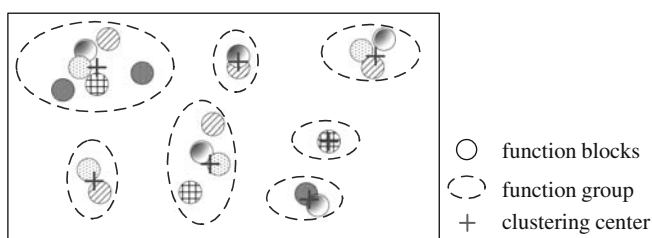
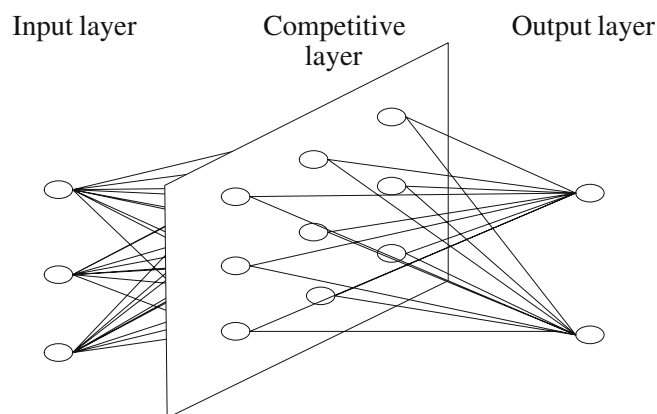
The competitive layer can be organized in one- or two-dimensions. Each neuron receives a summation of the weighted inputs from the input layer, and is associated with a collection of adjacent neurons, which form its ‘neighborhood’. In the function analysis problem, neurons in the competitive layer are organized in a two-dimensional lattice. Once the network has been initialized, three processes are involved in the formation of the feature map (Haykin 1999), described as follows:

1. Competition: For each input vector, the neurons in the competitive layer compute their responsive values according to a distant function. The neuron with the largest responsive value is declared as the winner.
2. Cooperation: The topological neighbors of the winner are determined to provide the basis for cooperation among them. The winner and its neighbors are collectively called the excited neurons.
3. Synaptic activation: The excited neurons increase their individual responsive values of the distant function in relation to the input vector. This is

Table 2 Correlation between key characteristics (KCs) and function groups (FGs) (TR₂)

KCs	FGs							
	Flow generation	Control	Air filtering	Casing	Flow distribution	Inlet	Insulation	
Power supply	1	1	0	0	0	0	0	0
Power consumption	1	1	1	1	0	0	0	0
Controller type	0	1	0	0	0	1	0	0
Motor type	1	1	0	0	1	0	0	0
Filter type	0	0	1	0	0	0	0	0
Air quantity	1	0	1	1	1	0	0	0
Total static pressure	1	0	1	1	1	0	0	0
Air velocity	1	0	1	1	1	0	0	0
Air uniformity	0	0	1	1	1	0	0	0
Service cleanliness	0	0	1	0	0	0	0	0
Noise level	1	0	0	0	1	0	1	0
Casing size	0	0	1	1	0	0	0	0
Casing material	0	0	0	1	0	0	0	0
Vibration	1	0	0	1	0	0	1	0
Mounting type	1	0	0	1	1	1	0	0
Cost	1	1	1	1	1	1	1	1
Weight	1	1	1	1	1	1	1	1
Manufacturing time	1	1	1	1	1	1	1	1

1 KC is dependent on a function; 0 KC is not dependent on a function

**Fig. 7** Graphical interpretation of function clustering**Fig. 8** The network architecture of a self-organizing map (SOM) (Haykin 1999)

achieved through adjusting the weight vectors of the excited neurons such that they move towards the input vector.

At the initial stage of building the network, no specific order is present. However, after the training processes, the neurons in the competitive layer are organized into some meaningful patterns, i.e., the feature maps.

The purpose of the output layer is to visualize the interconnections between the nodes in the competitive layer. In this research, the output layer is organized as a tree structure, with the root nodes representing groups of functions and the leaf nodes representing the function components belonging to actual product cases (Fig. 10a).

4.2.2 Functional platform building using SOM

A SOM network is built to analyze the FBs from a set of representative FFU products. Seven nodes are used in the input layer, and a two-dimensional, 5×5 hexagonal grid is used in the competitive layer (Fig. 9). In a hexagonal grid, each node has six 'neighbors', except those on the borders. When the input data (the FBs) are initially imported, all these data are located at the origin. Next, a training process is performed according to the three procedures discussed earlier. Upon repeated presentation of the training data, the synaptic weight

Table 3 Input vector of a function block (FB)—air flow generation

Elements	Function action	Input energy	Input material	Input signal	Output energy	Output material	Output signal
Value	Convert	Electric current	Air	–	Velocity	Air	–
Code	302	1202	2203	0	1163	2203	0

Fig. 9 The feature pattern in the competitive layer after training

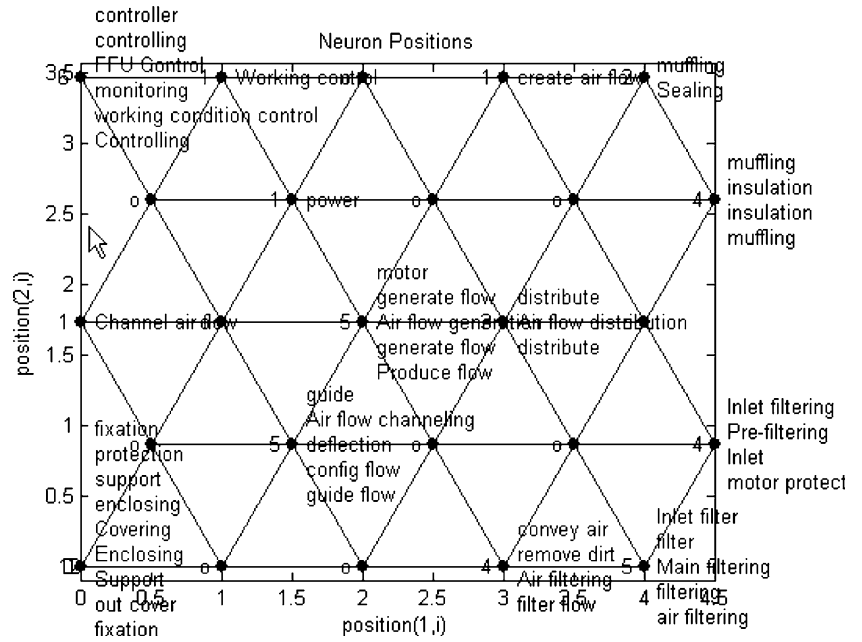
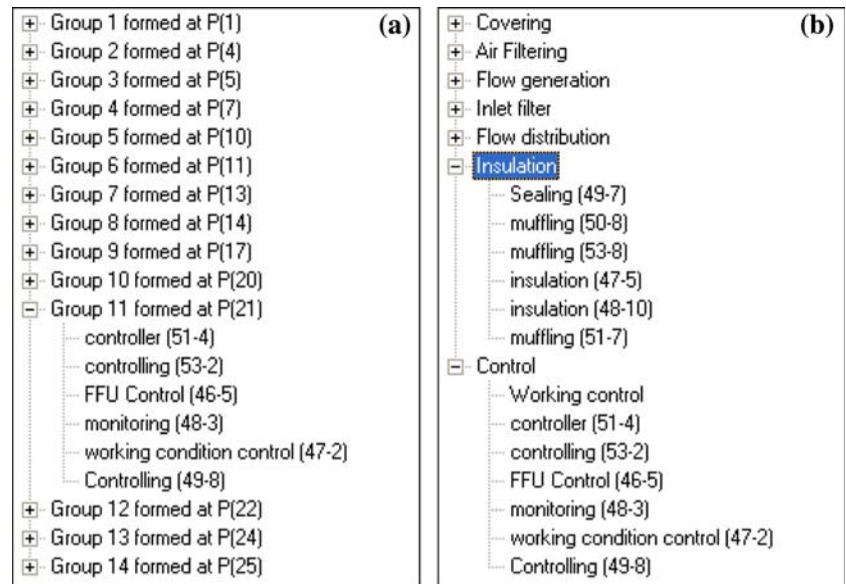


Fig. 10 The functional product platform



vectors tend to follow the distribution of the input vector due to the synaptic updating. A typical pattern after the training process is shown in Fig. 9.

As can be seen from the feature map, the functions have been grouped according to their affinity to each other. The number on the left of each node represents the number of functions that have been clustered at that node. For example, 12 functions are located at $N(1,1)$, and they are reasonably considered as a group denoting an ‘enclosing’ or ‘casing’ function. Based on this pattern, a tree structure is generated as the output layer according to the node that has at least one FB assigned (Fig. 10a). This tree structure will be further refined by a human designer. The refinement process allows the empirical knowledge of human designers to be incorporated, and

the identification of possible errors in the SOM that are caused either by insufficient training or by noise data. The refinement includes operations such as merging similar functions, deleting non-representative functions, and changing the FG name. For example, the function ‘Working control’ located at $N(2, 5)$ is merged with its neighbor $N(1,5)$, which is the center of the ‘control’ function. Finally, seven FGs are identified for the FFU products (Fig. 10b).

4.2.3 Evaluation of the feature map

The formation of the feature map relies on the function basis of the product information model. Function sim-

ilarity is the underlying rule to group the functions, where function similarity is estimated based on the coded functions and flows. In essence, these are analogous to the quantitative and heuristic methods used in modular design (McAdams et al. 1999; Zamirowski and Otto 1999; Stone et al. 2000a, b). The resulting function groups are similar to the product platforms obtained in these approaches.

Self-organizing map (SOM) is a necessary step although a refinement process is carried out subsequently by a human designer. The reasons are, firstly, the SOM method analyses the data from the KEV while a designer relies more on empirical information; secondly, a manual analysis of the data from the KEV will involve significant effort especially when a large number of functions are included; and thirdly, the designer may not be aware of the appropriate numbers or patterns of FGs at the beginning of the analysis. Therefore, the advantages of using the SOM method are (1) an expedition of the process, (2) alleviation of human labor, (3) determination of useful initial knowledge and pattern of the platform structure, and (4) analysis of the data from a perspective other than empirical observation. Therefore, the SOM process and human refinement are complementary to each other.

To summarize, the SOM networks can generate a feature map from a set of functions without supervision. It can identify the preliminary patterns of the functions and facilitate product platform building. This method is different from product platform design using the top-down approaches which require extensive product analysis (Simpson et al. 2001). In comparison with the product architecture building methods in modular design (McAdams et al. 1999; Zamirowski and Otto 1999; Stone et al. 2000a, b), the SOM method is a computational technique using unsupervised learning algorithms. Therefore, it has less reliance on human expertise and is very fast. Although, the final formation of a product platform still requires refinement by human, it is carried out after a feature map has been formed such that a rudimentary architecture is already present. Therefore, the refinement process is not a heavy load.

It should be noted that the elements in each FGs are the atomic functions belonging to the existing products. According to the initial product information model, the design conceptual components can be referenced by these atomic functions. Thus, a matrix TR_3 can be automatically formed to correlate the FGs with the conceptual components.

5 Solution synthesis using multi-objective optimization

5.1 Problem definition

The design specifications with respect to KCs are categorized into two types, namely the design objectives and the design constraints. In a typical product design problem, there are usually multiple non-commensurable

objectives to be achieved simultaneously, such as minimizing cost, minimizing manufacturing time, etc. The design constraints refer to a number of restrictions imposed on the design artifact, such as the product dimension, material, power consumption, etc. At the same time, the conceptual components are considered as the design parameters to fulfill the objectives, subject to the design constraints.

Let $C = [c_1, c_2, \dots, c_n]^T$ be a vector of the design parameters, where c_i is the i th conceptual component. The possible values of C constitute a parameter space. The values of the KCs, considered as design objectives, are computable from C . Let $T(C) = [t_1(C), t_2(C), \dots, t_r(C)]^T$, where $t_1(C), t_2(C), \dots, t_r(C)$ are the r objective functions. The values of $T(C)$ constitute an attribute space. The values of the KCs, considered as design constraints, are computable from C , i.e., $G(C) = [g_1(C), g_2(C), \dots, g_s(C)]^T$, where $g_1(C), g_2(C), \dots, g_s(C)$ are the s constraint functions. The constraints will divide the parameter space into two regions, namely, feasible and infeasible regions. The optimal solutions must also be feasible. Assuming that the design requirements for a particular design task are initiated as design constraints $g_1^0, g_2^0, \dots, g_s^0$, it is required the suitable C be set, from which the corresponding $G(C)$ is computed to fulfill the design constraints. Thus, the problem can be formulated as a multiple objective optimization problem (MOOP).

$$\text{Min } T(C) = [t_1(C), t_2(C), \dots, t_r(C)]^T \quad (4)$$

$$\text{s.t. } C = [c_1, c_2, \dots, c_n]^T \in S \quad \text{and} \quad (5)$$

$$[g_1(C), g_2(C), \dots, g_s(C)]^T \leq [g_1^0, g_2^0, \dots, g_s^0]^T$$

In Eq. 5, the inequality applies component-wise, i.e., $g_i(C) \leq g_i^0$ ($i = 1, \dots, s$). It should be noted that $g_i(C) \leq g_i^0$ is a generalized representation of the constraint, such that when ‘ \geq ’ or ‘ $=$ ’ are involved, the equation need to be converted slightly.

The design problem is reduced to the selection and combination of the conceptual components from the existing products to minimize the design objectives subject to the constraints. In a fully developed design database, sufficient number of conceptual components will be retrieved as building elements. Thus, there will be many possible combinations. Moreover, for a typical engineering design problem, little is known about the shape and modality of the attribute space a priori. For example, the objective functions can be linear or non-linear; the attribute space can be convex or non-convex, discrete or continuous. This is especially true considering a generic method applicable to the design of various types of products. A robust and efficient search and optimization method is needed to find the optimal solutions.

5.2 Multi-objective optimization algorithm

A number of multi-objective optimization algorithms have been reported. Some of the algorithms have

converted a MOOP into a single objective optimization problem using a few user-defined procedures, such as the weighted sum method, the ϵ -constraint method, the weighted metric methods, etc. (Deb 2001). Deb (2001) pointed out that a major limitation of these methods is that they involve a number of user-defined parameters which are difficult to set in an arbitrary problem.

The multiple objective optimization methods based on genetic algorithm (GA) have been extensively studied and reported to address the above deficiency (Deb 2001). These algorithms are usually divided into the non-Pareto and Pareto-based approaches (Fonseca and Fleming 1998). Andersson and Wallace (2002) proposed a Pareto-based approach, namely, the multi-objective struggle genetic algorithm (MOSGA), and compared it with a few typical multi-objective GA, such as vector evaluating GA (VEGA), non-dominated sorting GA (NSGA), multi-objective GA (MOGA), etc. It is claimed that the MOSGA method can handle the multi-modal attribute spaces with improved robustness. Moreover, it requires less tuning of the GA parameters in comparison with the other methods. These are desirable for the design synthesis problem. Therefore this research adopts the MOSGA method.

In a MOOP, the search results usually constitute a Pareto-optimal solution set instead of just one optimal solution. A Pareto-optimal solution is also called a non-dominated solution. A solution is said to dominate another if it is superior in most, if not all objectives. Considering the minimization problem $T(C)$ with r objectives, and two solution vectors C_1 and C_2 , C_1 is said to dominate C_2 , if

$$\begin{aligned} \forall i \in \{1, 2, \dots, r\} : t_i(C_1) \leq t_i(C_2) \quad \text{and} \\ \exists j \in \{1, 2, \dots, r\} : t_j(C_1) < t_j(C_2) \end{aligned} \quad (6)$$

In a Pareto-based optimization GA, the fitness of an individual solution is represented as its rank. Each individual is assigned a rank, which is the number of population members that dominate over it plus one (Fonseca and Fleming 1998). Therefore, a lower ranking indicates a better fitness, and the solutions with the rank '1' are the non-dominated solutions. Based on the above discussion, the MOSGA algorithm for the design synthesis problem is outlined next (Fig. 11).

In Step 1, a design task is initiated by selecting a few KCs and setting them either as design objectives or constraints. For various types of products, the KCs are different, and as a result, the design objectives and constraints are different. In Step 2, a product platform is chosen and conceptual components are retrieved according to the FGs. Retrieval of conceptual components is subjected to the design constraints, which is discussed in Sect. 5.5. In Step 3, the configuration of the algorithm is set up by setting the population size and the number of generations. The initial population is generated according to the population size. In Step 4, two individuals are randomly selected from the population, and crossover and mutation is performed to generate the

children. Next, the r objective functions of each child are calculated. The child is compared with every individual in the population with respect to the r objective functions, through which the rank of the child is obtained. Next, the algorithm searches for an individual that is most similar to this child and replaces it with the child if the child has a lower rank, or if the child dominates it. In Step 5, the ranking of the population is updated if the child has been inserted. The population energy and population plot are updated accordingly. The Pareto-optimal front is identified in Step 6. It should be noted that the purpose of this method is to find adequate design variants, instead of identifying each and every global optimum. Therefore, once the design space has been explored sufficiently such that the population converges to a specific curve/surface, presumably the Pareto front, the search is stopped.

Two simple criteria are presented next. The sufficient exploration of the design space is confirmed when both conditions in the criteria are satisfied:

1. The average population energy is stabilized at a reasonably low level. The average population energy refers to the average values of the objective functions according to the current population. The energy is calculated for each generation. It will decrease gradually, and converge to a relatively stable state after a certain number of generations. It is observed that few new Pareto optima are discovered after the search reaches the stable state, and hence this is an indication that the design space has been effectively explored. A typical population energy curve is shown in Fig. 12a.
2. The Pareto-optimal front can be identified in the population plot. A population plot illustrates the individuals as points in the design space, with their objective functions as the coordinates. During the GA-based search, new individuals will be added into the plot. It is observed that at the early stage of the search, a number of new individuals are produced. As the search proceeds, fewer new individuals are produced, and the non-dominated optimal solutions gradually form a specific curve/surface. When such a curve/surface becomes apparent, the search is stopped. A typical population plot is shown in Fig. 12b. In this figure, the circles represent the initial population and the crosses represent the individuals generated during the search. The search converges to a Pareto-optimal set, indicated by the triangles.

A prototype design synthesis system has been developed. It is capable of visualizing the population energy and population plot. The user can observe the search process, stop the search, or fine-tune the parameters with a new iteration. For the parameters fine-tuning, two parameters are essential, namely, the population size and the number of generations. A user can carry out a few tentative design synthesis sessions to determine these parameters. Generally, the design space

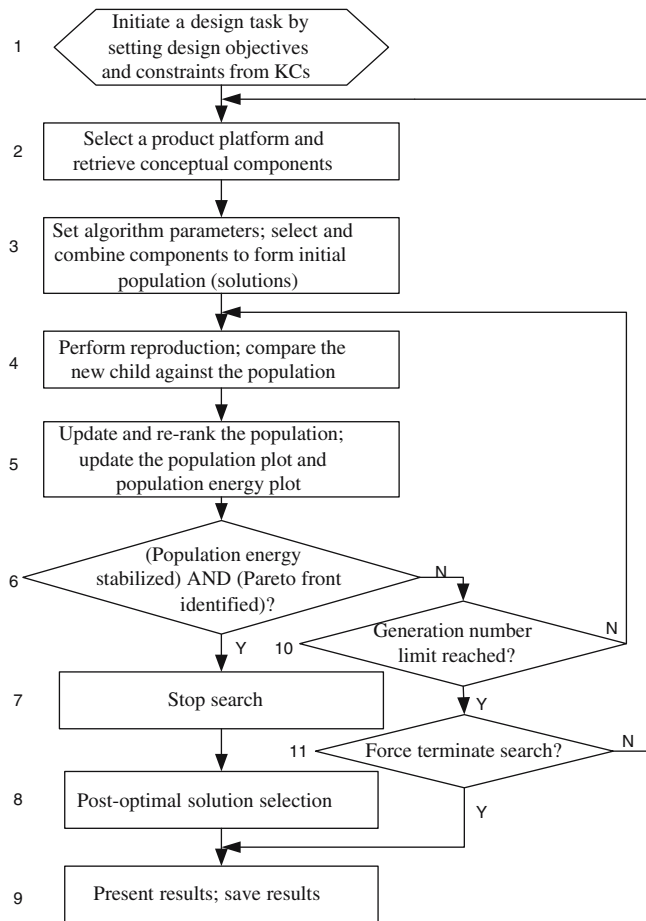


Fig. 11 A flowchart of the multi-objective struggle genetic algorithm (MOSGA) algorithm for the design synthesis problem

can be sufficiently explored with a large population size and a large number of generations. However, this significantly increases the computational load. A practical method is to start from a relatively small population size and small number of generations, and increase them gradually. The parameters are finalized when the increase of the two parameters does not result in an obvious improvement of the Pareto front.

In the subsequent steps, the algorithm controls the search by checking whether the generation reaches a limit, or whether the user force stops the search. The Pareto-optimal set, if it is obtained, undergoes a post-optimal selection process to arrive at the final candidate solutions. Design synthesis results are saved into the database if desired.

5.3 The structure of a chromosome

In this research, a chromosome, i.e., a solution is a combination of the available conceptual components. The length of a chromosome is the total number of candidate conceptual components, i.e., all the components that have been retrieved for the design synthesis.

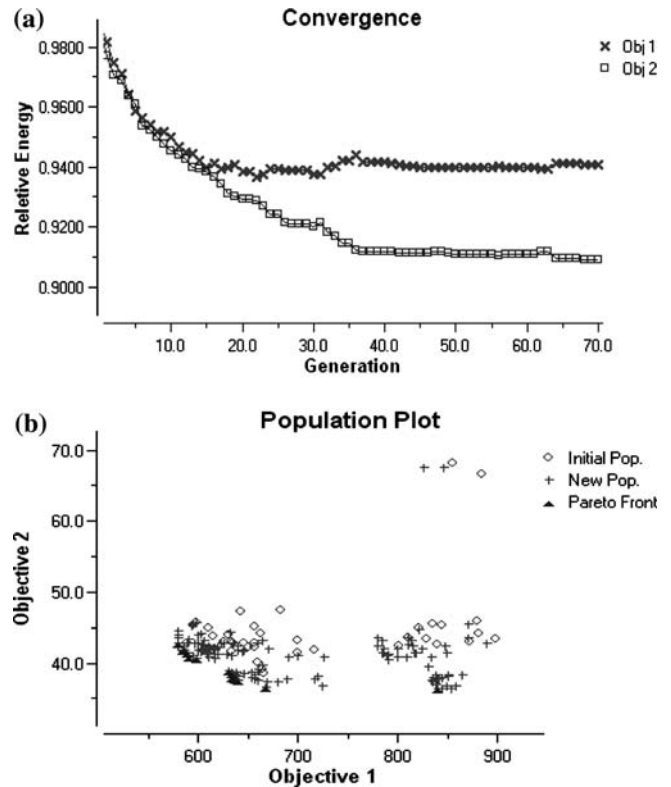


Fig. 12 Visualization of population energy and population plot

Each conceptual component is represented as one bit on the chromosome string and this one bit indicates whether that component has or has not been selected to form a solution. If a component has been selected to form a solution, the bit has a value of '1'; otherwise it has a value of '0'. This results in a string of bits indicating the selection of the conceptual components. An additional constraint is that only one conceptual component can be selected from within any one FG. For example, each column in Table 4 denotes one chromosome. Besides these, the stochastic universal selection, the single-point crossover, and the bit-wise mutation, which are common to genetic algorithms, are used to generate the offspring.

5.4 The objective functions

As has been discussed earlier, the individual solutions have to be ranked to carry out the GA-based search. To obtain the rank of a solution, the individual objective functions, namely, $t_i(C)$, ($i = 1, \dots, r$) have to be calculated. The information of every relevant gene is used to calculate the individual objective functions. In the FFU design, three objectives are considered, namely, cost, product weight, and manufacturing time. These objectives can be calculated based on the attributes of the conceptual components. In particular, each component has three corresponding attributes, namely, cost, weight, and manufacturing time. These attribute values can be

Table 4 A chromosome structure of the fan filter unit (FFU) product

Components	Solutions									
	1	2	3	4	5	6	7	8	...	N
R4E 355-AK05-05	0	1	0	0	0	0	0	0		1
S-1 LS controller	1	0	0	0	0	0	0	1		0
KLD Rock wool	0	0	0	1	0	0	0	0		0
G-type casing	0	0	1	0	0	1	0	0		0
Casing enclosed	0	0	0	0	0	0	1	0		0
Integrated controller	0	0	0	0	1	0	0	0		0
R3G 310-AJ38-61	1	0	0	1	0	0	0	0		0
ULPA filter	1	0	0	0	0	1	0	0		0
Al shaft casing	0	0	0	0	0	0	0	1		0
HEPA(99.99%@0.3 μ m)	0	1	0	0	0	0	0	0		0
R4E 310-AF-12-05	0	0	0	0	0	1	0	0		0
⋮										
D-S-S air guider	0	1	0	0	0	0	1	0		1

obtained from the existing product components. The final value of each objective for a particular solution is the summation of the corresponding component attributes. Once the individual objective functions are obtained, the rank of the solutions can be obtained. It should be noted that the three design objectives adopted in this case study are only for the FFU product. Other objective functions can be designed according to the characteristics of the designed product. For example, the objective functions for household/office products (e.g., power tools, printers, air conditioners, etc.) usually include cost, weight, reliability (measured by MTBF), efficiency, etc. The objective functions for industrial equipments (e.g., measurement devices, machine tools, etc.) include cost, precision, respond speed, reliability, etc. These objective functions are adopted to address the major concerns of the customers and the manufacturers. To compute the objective functions, the relationships between the objective functions and the attributes of the conceptual components must be established. The relationships can be in the form of equations (Chidambaram and Agogino 1999; Nelson et al. 2001) or meta-models. A few widely adopted meta-modeling techniques include the response surface method (Chen et al. 1996), regression analysis (Fujita and Yoshioka 2003), and kriging (Simpson 1998).

5.5 Applying design constraints

The design constraints are used to filter the design components so that only those components that can satisfy the constraints are used in the actual design synthesis. Firstly, it is noted that the constraints are extracted from the KCs, which can have four types of values, namely, categorical (C), Boolean (B), ordinal (O) and real number (R). Among them, the real number is continuous while the other three are discrete. The categorical type of data represents a set of mutually exclusive values. For example, the casing material of an FFU can be chosen from stainless steel, zinc–aluminum-coated steel, Aluminum, etc. A Boolean type refers to

the presence or absence of a characteristic. It is actually a special case of the categorical type. An ordinal type is similar to the categorical type but it has an ordering feature, i.e., a higher order value can satisfy a lower order requirement. For example, the service cleanliness can have the following grades $\{100,000, 10,000, 1,000, 100, 10, 1\}$. The succeeding ones can satisfy the preceding ones but not vice versa. A real type is usually given as a real number and a tolerance. For example, the air quantity can be specified as $1,800 \text{ m}^3/\text{h} \pm 18\%$.

The product case from which a component is extracted from is called its host product. A filtering process is carried out through a comparison of the KCs of the host products with the specifications of a new design task. It is assumed that the design specifications of a new design task constitute a design range vector $R_d = [d_1, d_2, \dots, d_q]^T$. R_d is defined with respect to a set of KCs (k_i , $i = 1, \dots, q$). The subscript here is q instead of p as not all the KCs are used as the design constraints. A host product is characterized by the same set of KCs, which determines a system range vector $R_s = [s_1, s_2, \dots, s_q]^T$. Thus, a pair-wise comparison can be made between R_s and R_d , based on the type of KCs. This leads to a set of results showing whether a system range can satisfy a design range.

At the same time, each conceptual component c_j ($j = 1, \dots, m$) is related to a few KCs through the transformation matrices TR_2 and TR_3 , i.e., there exists a subset of $\{k_i | i = 1, \dots, q\}$, whose elements are the KCs related to c_j ($j = 1, \dots, m$). If, for every element in this subset, the corresponding s_i satisfies d_i , the candidate component c_j will be selected for the design synthesis. Otherwise, the component is discarded. This process is shown in Fig. 13.

5.6 Post-optimal solution selection

The MOSGA method will result in a Pareto-optimal set. Usually, a designer needs to select one or more candidate solutions to be further developed in the subsequent design processes. To do so, some meta-knowledge is

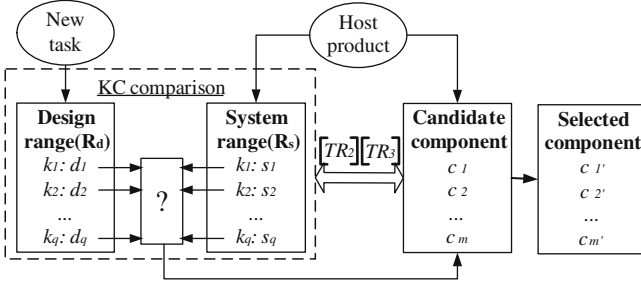


Fig. 13 Retrieving components through key characteristics (KCs) comparison

needed, e.g., a designer may need to show his/her preference of the design objectives through assigning different weights to them. In this research, a relative weight is assigned to each objective and the weighted fitness $T_w(C)$ is obtained for every Pareto-optimal solution.

$$T_w(C) = \sum_{i=1}^r \frac{w_i (t_i(C) - t_i^{\min})}{t_i^{\max} - t_i^{\min}} \quad (7)$$

where r is the number of objectives; w_i is the weight assigned to the i th objective and $\sum_{i=1}^r w_i = 1$; t_i^{\max} and t_i^{\min} are the maximum and minimum objective functions for the i th objective, respectively. $(t_1^{\min}, t_2^{\min}, \dots, t_r^{\min})$ constitutes the coordinates of a utopian ideal solution. $T_w(C)$ is actually a normalized, weighted distance that a solution is to the ideal solution. The final solution is chosen as the one(s) with the smallest weighted fitness. If the weighted fitness of several solutions is very close, additional meta-knowledge is required to assess the merits of the candidate solutions.

6 A case study

A case study is carried out to design a new FFU product using the design reuse approach. To study the effectiveness of this approach, parallel design sessions were carried out by two groups of designers, with respect to the same set of design requirements (Table 5). Group A carried out the design based on experience. Group B performed the task using the design reuse system. Both groups consisted of two designers, namely, a product planner who proposes the configuration of the product, and a technical engineer who is responsible for estimating the technical feasibilities. Designers in the two groups have the same level of experience, i.e., the average years of profession are equivalent (3.5 years). The product planner in Group B was given a short training course to use the design reuse system. In addition, the subjective weights assigned to the design objectives would influence the results. Group B adopted quantitative weights, namely, $w_1 = 0.5$ for cost, $w_2 = 0.4$ for manufacturing time, and $w_3 = 0.1$ for product weight. A similar strategy was adopted by Group A. However, estimated values of the objective functions were used where the exact values were not available. The proce-

Table 5 Design objectives and design constraints

Objectives	
Minimize cost	
Minimize manufacturing time	
Minimize weight	
Constraints	
Mounting type	Horizontal
Air quantity	1,375 m ³ /h \pm 10%
Total static pressure	200 Pa \pm 4%
Air velocity	0.42 m/s \pm 10%
Air uniformity	< 15%
Service cleanliness	Class1000
Noise level	< 50 dBA
Casing material	Aluminum
Casing height	< 330 mm
Vibration	< 0.025 G

dures using different approaches are presented next, and the outcome is compared according to solution variety and solution superiority.

6.1 Experience-based design

6.1.1 The design process

The experience-based method involves procedural selection and combination of the design components. Each component is selected or designed to achieve a set of performance factors, derived from the design requirements. The designer usually relies heavily on individual/team experience, documentation, and simulation results. Therefore, individual/team preferences significantly influence the efficiency and validity of the design. Since the components are determined by a number of factors, it is not easy for a designer to manage them simultaneously. Modifications and rework are often inevitable.

6.1.2 Results

The configuration of the product is generated by Group A. It is noted that the division of the design components is different for the experience-based method and the design reuse method. For the convenience of comparison, this paper has purposely rearranged the components into an identical format. The product configuration is shown in Table 6.

6.2 Product design using the design reuse approach

6.2.1 The design process

In the preparation stage for carrying out this case study, a total of 22 FFU product cases have been collected, modeled and stored in the database. A set of 18 KCs is

Table 6 Configuration of the candidate solution generated by Group A

Technical data		
Blower	Type	R4E 310-AP20-01
	Motor	M6E-097
	Voltage (V)	230
	Power (W)	60
	Cost (\$\$)	130
	Air flow (m ³ /h)	1,380
	Speed (rpm)	1,160
	Noise (dBA)	54
	Capacitor (μ F/VDB)	1.5/450
	Weight (kg)	4
Control	Type	LSC-S1
	Cost (\$\$)	10
	Communication	RS485
	Weight (kg)	0
Filter	Type	HEPA filter
	Thickness (mm)	70
	Pressure loss (Pa)	80
	Cost (\$\$)	200
	Width (mm)	865
	Length (mm)	1,165
	Weight (kg)	10
Casing	Material	Aluminum
	Height (mm)	330
	Thickness (mm)	1.5
	Cost (\$\$)	220
	Width (mm)	870
	Length (mm)	1,170
	Weight (kg)	25
Air guider	Type	Set-piece
	Mounting	Rivet
	Cost (\$\$)	15
	Set number	2
	Weight (kg)	2
Inlet cone	Shape	Square
	Type	Punched hole
	Cost (\$\$)	10
	Size (mm)	330
	Weight (kg)	2
Insulation	Type	Corner blocks
	Mounting	Sticking
	Cost (\$\$)	5
	Set number	2
	Material	Rock wool
	Weight (kg)	2

Summary Cost: \$\$590; Weight: 45 kg; Manufacturing time: approx. 30 min

formalized and the correlation of KCs and FGs is established as TR₂ (Table 2). Product analysis has been carried out to establish a functional product platform, which is composed of seven FGs. In each product case, a set of conceptual components were identified for the functions. If a conceptual component is implemented in more than one product cases, the duplicated ones were eliminated. Thus, a component catalog was built to include all the unique conceptual components according to product functions. Accordingly, the transformation matrix TR₃ to correlated FGs with conceptual components is available. TR₃ is not shown here due to its huge size.

Following the procedures presented in Sect. 5, design synthesis is carried out using the design reuse prototype system. The major steps are:

Step 1. Input the design objectives and design constraints as are shown in Table 5.

Step 2. Select a product platform and retrieve the conceptual components (Fig. 14). In this step, only the components that satisfy the design constraints are selected.

Step 3. Set the parameters of the algorithm, namely, the population size and the number of generations. To determine the suitable algorithm parameters, a few tentative synthesis sessions are performed. A population size of 60 and number of generations of 100 are adopted for this problem.

Step 4. Identify the Pareto front by observing the population plot and population energy.

Step 5. Set a relative weight to each objective. Calculate the weighted fitness and present the solution candidates (Fig. 15).

6.2.2 Results

Based on the above procedures, the design synthesis has arrived at a Pareto-optimal set of 33 solution variants. Each solution presents a possible configuration of the product. An excerpt of the solution configuration is shown in Fig. 15. In this figure, each cell shows the name of the component. The numbers in the brackets are the indices of the product and the component. A list of the objective functions of the solutions is shown in Fig. 16.

Finally, to choose the candidate solutions from the Pareto-optimal set, the weights to the objectives are assigned as: cost ($w_1 = 0.5$), manufacturing time ($w_2 = 0.4$) weight ($w_3 = 0.1$). These weights are assigned according to the designer's knowledge of the objectives' importance. It can be observed that the weighted fitness of three solutions are very close, namely, Solution 3 (0.09), Solution 13 (0.09), and Solution 14 (0.08). A closer study of the components in the three solutions is required. It is observed that the *Al-shaft* (51-4) has been shared by the three solutions corresponding to component 1 (casing) (Fig. 15). Similarly, identical components have been used for components 3 (blower) and 7 (controller), respectively. For the other components, namely, 2, 4, 5, and 6, there are only minor differences among the attributes of the actual components. For example, Solution 14 has used 2-4 *HEPA T70* (71-4) as the filter, while solutions 3 and 13 have used *HEPA T75* (77-7). Differences exist concerning the filter thickness, cost, and weight. However, these differences are insignificant. Therefore, the three solutions are almost identical to each other. The designer can choose any one without making much difference in the final results. Next, the designer retrieves the components, and

Fig. 14 Retrieve product platform and conceptual components

Index	Available Platforms	Create time
14	Fan Filter Unit	2/16/2005

ID	Component Name	Cost	Weight	Production Time	Pro...	Product Name	Funcio...	Detail
6	SUS Casing	284	18.3999996	25	46	FFU-0501-SG	3	standard SUS Casing
7	N.A.	0	0	0	46	FFU-0501-SG	5	3PH power supply. No speed c
5	Punch square array	18	3.5	2	46	FFU-0501-SG	8	Inlet cone
4	ULPA	240	10	0	46	FFU-0501-SG	9	Ultra Low penetration air filter
3	R4D-310-APA0-09	130	3.79999995	0	46	FFU-0501-SG	10	OEM German ebm papst
1	Set air guiders	18	3.5	2	46	FFU-0501-SG	11	standard 3 planer air guider
2	slanted air guider	18	3.5	2	46	FFU-0501-SG	12	
7	Integrated controller	10	0.5	0	47	FFU-0502-SG	2	
4	Rockwool Insulation	10.5	4.80000019	3.5	47	FFU-0502-SG	5	
1	4-4HEPA	320	20	0	47	FFU-0502-SG	7	Thickness 70mm HEPA filter
6	Motor guard	27	4	2.5	47	FFU-0502-SG	8	
5	4-4 Galvalume casing	424	40.4000015	41.5	47	FFU-0502-SG	9	
3	R3G 400-AD36-61	390	5	0	47	FFU-0502-SG	10	OEM size 400 motor by ebm pe
2	Deflectors	21	6.69999980	9	47	FFU-0502-SG	11	standard 3 set air deflector
7	R4D 310-AD26-09	110	3.79999995	0	48	FFU-0401-TW	4	OEM 310 AC type motor , ebm
6	2-4 HEPA	180	10	0	48	FFU-0401-TW	5	High efficiency particulate air fil
4	Galvalume casing	300	21.2000007	28.5	48	FFU-0401-TW	6	
1	standard Inlet cone	28	3.5	1.79999995	48	FFU-0401-TW	8	
2	Air flow guider	10	4.40000009	8	48	FFU-0401-TW	9	
5	Rockwool insulation	8	3.79999995	4.19999980	48	FFU-0401-TW	10	
3	AL shaft fixation	300	21.2000007	28.5	48	FFU-0401-TW	11	

Fig. 15 Configurations of the Pareto-optimal solutions and evaluation results

	Candidate Solution A	Candidate Solution B	Candidate Solution C
Component 1	Al shaft (51-4)	Al shaft (51-4)	Al shaft (51-4)
Component 2	2-4 HEPA T70(71-4)	hepa t75(77-7)	hepa t75(77-7)
Component 3	R4E 310-AF-12-05(53-3)	R4E 310-AF-12-05(53-3)	R4E 310-AF-12-05(53-3)
Component 4	Set air guiders(46-1)	slanted air guider(46-2)	Set air guiders(46-1)
Component 5	RockWool(67-3)	RW STD Muffler(66-3)	RW STD Muffler(66-3)
Component 6	Coil Al(74-2)	STD-310(75-6)	STD-310(75-6)
Component 7	N.A.(46-7)	N.A.(46-7)	N.A.(46-7)
Objecives	Cost Weight Production time;	Cost Weight Production time;	Cost Weight Production time;
Obj. Values	524.00 40.90 28.20	526.00 40.80 28.40	526.00 40.80 28.40
Weighted fitness	0.08	0.09	0.09

Fig. 16 The objective functions of the Pareto-optimal set

	Cost	Weight	Production time	Weighted fitness
Number of Pareto optima: 33				
* Solution 1	*520.00	40.80	33.20	0.35
* Solution 2	*724.00	40.50	28.20	0.37
* Solution 3	*526.00	40.80	28.40	0.09
* Solution 4	*644.00	36.90	30.20	0.32
* Solution 5	*646.00	36.80	30.40	0.33
* Solution 6	*522.00	40.70	33.40	0.36
* Solution 7	*640.00	39.20	30.50	0.36
* Solution 8	*635.10	37.80	34.00	0.53
* Solution 9	*840.00	36.40	35.20	0.88
* Solution 10	*642.00	36.70	35.40	0.60
* Solution 11	*844.00	36.50	30.20	0.61
* Solution 12	*646.00	36.80	30.40	0.33
* Solution 13	*526.00	40.80	28.40	0.09
* Solution 14	*524.00	40.90	28.20	0.08
* Solution 15	*644.00	36.90	30.20	0.32
* Solution 16	*522.00	40.70	33.40	0.36
* Solution 17	*727.00	40.20	29.70	0.46
* Solution 18	*726.00	40.40	28.40	0.39
* Solution 19	*527.00	40.60	29.70	0.16
* Solution 20	*644.00	36.90	30.20	0.32
* Solution 21	*632.00	40.10	35.00	0.61
* Solution 22	*637.10	37.70	34.20	0.54
* Solution 23	*517.10	41.70	32.20	0.30
* Solution 24	*726.00	40.40	28.40	0.39
* Solution 25	*517.10	41.70	32.20	0.30
* Solution 26	*646.00	36.80	30.40	0.33
* Solution 27	*511.10	44.10	32.30	0.33
* Solution 28	*637.10	37.70	34.20	0.54
* Solution 29	*646.00	36.80	30.40	0.33
* Solution 30	*515.10	41.80	32.00	0.29
* Solution 31	*647.00	36.60	31.70	0.40
* Solution 32	*844.00	36.50	30.20	0.61
* Solution 33	*640.00	36.80	35.20	0.59

adapts them to the design requirements. Finally, a detailed product configuration is generated (Table 7).

6.3 Comparison of the two methods

The effectiveness of the experience-based method and the design reuse method are compared with respect to solution variety and solution superiority.

The design reuse method has generated more design variants. The design synthesis based on the multiple objective optimization has generated a set of 33 Pareto-optimal solutions. The post-optimal selection leads to three solutions from the Pareto set. Although only one solution is chosen for further development, it is advantageous to have a few backups, in case demerits are discovered in the chosen solution in the subsequent design process.

Table 7 Configuration of the candidate solution generated by Group B

Technical data		
Blower	Type	R4E 310-AF12-05
	Motor	M6E-068
	Voltage (V)	230
	Power (W)	55
	Cost (\$\$)	110
	Manufacture time (min)	0
	Air flow (m ³ /h)	1,320
	Speed (rpm)	1,120
	Noise (dBA)	53
	Capacitor (μ F/VDB)	1.5/450
Weight (kg)	3.8	
Control	Type	Motor integrated
	Cost (\$\$)	0
	Manufacture time (min)	0
	Weight (kg)	0
Filter	Type	(2*4)HEPA T70
	Thickness (mm)	70
	Pressure loss (Pa)	80
	Cost (\$\$)	180
	Width (mm)	865
	Length (mm)	1,165
	Manufacture time (min)	–
	Weight (kg)	10
Casing	Material	Aluminum
	Height (mm)	325
	Thickness (mm)	1.5
	Cost (\$\$)	184
	Width (mm)	870
	Length (mm)	1,170
	Manufacture time (min)	23
	Weight (kg)	18.4
Air guider	Type	Set air guiders
	Mounting	Rivet
	Cost (\$\$)	18
	Set number	3
	Manufacture time (min)	2
	Weight (kg)	3.5
Inlet cone	Shape	Circular
	Type	Aluminum coil set
	Cost (\$\$)	24
	Manufacture time (min)	0.2
	Size (mm)	310
	Weight (kg)	1.4
Insulation	Type	Corner block
	Mounting	Sticking
	Cost (\$\$)	8
	Manufacture time (min)	3
	Set number	3
	Material	Rock wool
	Weight (kg)	3.8

Summary Cost: \$\$524; Weight: 40.9 kg; Manufacturing time: 28.2 min

The design reuse method can generate solutions that are superior in the design objectives, namely, cost, weight, and manufacturing time. The final adopted solution has the objective functions of cost 524 (\$\$), weight 40.9 (kg), and manufacturing time 28.2 (min). In comparison, the solution generated by Group A is estimated as cost 590 (\$\$), weight 45 (kg), and

manufacturing time 30 (min). There is a notable improvement in the design objectives. This can be attributed to the advantage of the design reuse method, namely, the GA-based search has explored the design space more effectively to reuse the best-fitted components.

7 Discussions

The method proposed in this research is aimed at facilitating conceptual design through design reuse. The benefits of this method are discussed next.

Firstly, design is less dependent on expertise using the proposed method. The design reuse system is not only a database for recording product information, but also can extract knowledge from the raw data. The SOM method can effectively analyze product functions and facilitate the building of a functional product platform. Moreover, using this system, each new product case will be recorded and possibly contribute to future design. Hence, the system has established a method to record the knowledge and reuse it in a systematic way.

Secondly, the design time using the proposed method is shorter as compared with the manual approach. The design synthesis process itself can be carried out quickly. Moreover, the subsequent detailed design can be carried out more efficiently. This is because the design reuse system is developed with the support of database technology. The product information has been collected and well-indexed in the database, time saving can be achieved through more efficient information retrieval. It should be noted that to achieve this, an initial stage of information collection and analysis must be performed, which causes overhead to the design-by-reuse process. However, the product information can be utilized repeatedly, and the database can be constantly updated when new design cases are presented.

The proposed method and prototype system provide assistance in design variants generation. The design synthesis based on multiple objective optimization can generate a set of Pareto-optimal solutions. The post-optimal selection process will present the designer with one or a few solutions from the Pareto set, depending on the weighted fitness. This can lead to better design efficiency in comparison to manual or interactive decision support methods (Yu and MacCallum 1995), especially in large combinatorial problems. Moreover, with the support of a large engineering database, the solutions are generated based on more design cases than could be accessed by a designer based solely on his/her experiences. Thus, a designer is better supported to access the vast amount of product information.

A few limitations of the proposed method are discussed next. Firstly, the function-based product information model reduces the representation flexibility. Although the information model strives to capture product information in a comprehensive way, the use of function structure and taxonomies is restrictive and

inevitably reduces the flexibility. However, such a model is necessary in order to perform data analysis using computational methods.

Secondly, the formation of the product platform requires human interaction. The SOM method is an unsupervised learning method, but the building of the product platform still requires human intelligence. Given the same set of data, different designers can arrive at different product platform structures, leading to different effectiveness of product information management. It is proposed that future work be carried out to enhance the building of product platforms.

Thirdly, the design synthesis method using the MOSGA algorithm does not guarantee that the best solution is found. The method is robust in the sense that it is able to explore the design space effectively and arrive at near global optimum. However, being a heuristic algorithm, the method does not guarantee that all the global optimums are found, which is especially true for a product design problem that involves significant complexity and nonlinearity.

8 Conclusions

This paper presents a function-based design synthesis approach to design reuse in conceptual design. Efforts have been made to combine the function-based information management methodologies and the computational design synthesis approaches. The function-based product information modeling provides a coherent scheme for knowledge representation and subsequent knowledge extraction. These results, in turn, are critical to the design synthesis procedures, which constitute the design-by-reuse process. The research adopted an unsupervised learning technique, namely, the SOM neural networks, to facilitate the building of functional product platforms. The SOM method can achieve intelligent product architecture building. The design-by-reuse process is mainly carried out using the multi-objective optimization algorithms, which is capable of dealing with diverse design objectives and constraints. A design reuse system was developed to enable automated solution generation for product conceptual design. The effectiveness of the methods has been illustrated using a case study of an industrial product.

References

- Andersson J, Wallace D (2002) Pareto optimization using the struggle genetic crowding algorithm. *Eng Opt* 34:623–644
- Bobrow D, Falkenhainer B, Farquhar A, Fikes R, Forbus KD, Gruber TR, Iwasaki Y, Kuipers BJ (1996) A compositional modeling language. In: *Proceedings of the 10th international workshop on qualitative reasoning*. Menlo Park, AAAI Press, pp 12–21
- Campbell M, Cagan J, Kotovsky K (1999) A-Design: an agent-based approach to conceptual design in a dynamic environment. *Res Eng Des* 11:172–192
- Chakrabarti A, Bligh T (1996) An approach to functional synthesis of mechanical design concepts: theory, applications, and emerging research issues. *Artifi Intell Eng Des Anal Manuf* 10:313–331
- Campbell M, Cagan J, Kotovsky K (2000) Agent-based synthesis of electromechanical design configurations. *J Mech Des* 122:61–69
- Chen W, Allen JK, Mavris DN, Mistree F (1996) A concept exploration method for determining robust top-level specifications. *Eng Opt* 26:137–158
- Chidambaram B, Agogino AM (1999) Catalog-based customization. In: *Proceedings of the 1999 ASME design engineering technical conferences*, Paper No. DETC99/DAC-8675, Las Vegas, Nevada
- Deb K (2001) *Multi-objective optimization using evolutionary algorithms*. Wiley, New York
- Fonseca C, Fleming P (1998) Multiobjective optimization and multiple constraint handling with evolutionary algorithms—part I: a unified formulation. *IEEE Trans Syst Man Cybern Part A: Syst Humans* 28:26–37
- Fujita K, Yoshioka S (2003) Optimal design methodology of common components for a class of products: its foundations and promise. In: *Proceedings of the 2003 ASME design engineering technical conferences and computers and information in engineering conferences*, Paper No. DETC2003/DAC-48718, Chicago
- Haykin S (1999) *Neural networks: a comprehensive foundation*. 2nd edn. Prentice Hall, Upper Saddle River
- Kirschman CF, Fadel GM (1998) Classifying functions for mechanical design. *J Mech Des* 120:475–482
- Martin M, Ishii K (2002) Design for variety: developing standardized and modularized product platform architectures. *Res Eng Des* 13:213–235
- McAdams DA, Stone RB, Wood KL (1999) Function interdependence and product similarity based on customer needs. *Res Eng Des* 11:1–19
- Mittal S, Frayman F (1989) Towards a generic model of configuration tasks. In: *Proceedings of the 11th international joint conference on artificial intelligence*. Detroit, pp 1395–1401
- Nelson SA, Parkinson MB, Papalambros PY (2001) Multicriteria optimization in product platform design. *J Mech Des* 123:199–204
- Otto K, Wood KL (2001) *Product design techniques in reverse engineering and new product development*. Prentice Hall, New Jersey
- Pahl G, Beitz W (1996) *Engineering design: a systematic approach*. 2nd edn. Springer, London
- Pratt MJ, Anderson BD (2001) A shape modeling applications programming interface for the STEP standard. *Comput Aided Des* 33:531–543
- Pulm U, Lindemann U (2001) Enhanced systematics for functional product structuring. In: Culley S et al. (eds) *International conference on engineering design (ICED 01) design research—theories, methodologies and product modelling*. Professional Engineering Publishing, pp 477–484
- Rezayat M (2000) Knowledge-based product development using XML and KCs. *Comput Aided Des* 32:299–309
- Simpson TW (1998) A concept exploration method for product family design. Ph.D. Dissertation, Georgia Institute of Technology, Atlanta
- Simpson T, Maier J, Mistree F (2001) Product platform design: method and application. *Res Eng Des* 13:2–22
- Stone RB, Wood K (2000) Development of a functional basis for design. *J Mech Des* 122:359–370
- Stone RB, Wood KL, Crawford RH (2000a) Using quantitative functional models to develop product architectures. *Des Stud* 21:239–260
- Stone RB, Wood KL, Crawford RH (2000b) A heuristic method for identifying modules for product architectures. *Des Stud* 21:5–31
- Suh NP (2001) *Axiomatic design—advances and applications*. Oxford University Press, New York

- Szykman S, Sriram RD, Bochenek C, Racz J (1998) The NIST design repository project. In: Roy R, Furuhashi T, Chawdhry PK (eds) *Advances in soft computing—engineering design and manufacturing*. Springer, London, pp 5–19
- Szykman S, Racz JW, Sriram RD (1999) The representation of function in computer-based design. In: *Proceedings of the 1999 ASME design engineering technical conferences*, Paper No. DETC99/DFM-8742, Las Vegas
- Szykman S, Fenves SJ, Keirouz W, Shooter SB (2001) A foundation for interoperability in next generation product development systems. *Comput Aided Des* 33:545–559
- Yu B, MacCallum K (1995) Modelling of product configuration design and management by using product structure knowledge. In: *Proceedings of the 1st international federation for information processing working group 5.2 workshop on knowledge intensive CAD (KIC-1)*, Helsinki University of Technology, Helsinki
- Zamirowski EJ, Otto KN (1999) Identifying product family architecture modularity using function and variety heuristics. In: *Proceedings of the 1999 ASME design engineering technical conferences*, Paper No. DETC99/DTM-8760, Las Vegas