SCIENCE CHINA
Technological Sciences

• Article •

February 2017 Vol.60 No.2: 209–218
doi: 10.1007/s11431-016-0236-8

Functional knowledge integration of the design process

CHEN Bin[*] & XIE YouBai

State Key Laboratory of Mechanical System and Vibration, Shanghai Jiao Tong University, Shanghai 200240, China

Received March 29, 2016; accepted October 9, 2016; published online December 20, 2016

Functional knowledge integration is the initial and core phase of a design process. It is the key phase to ensure that the functional requirement of the design product can be appropriately complied with, and its result is also the rudiment of the subsequent detailed design work. If this important phase can be supported by an increasingly distributed resource environment, and be automated such that its completion requires less manual work, the efficiency of the design process would be largely improved and its ability to promote innovation would be enhanced. Therefore, this study involved a detailed analysis of the functional knowledge integration of the design process, as well as the proposal of a corresponding running model. Based on the model, a computational algorithm and an evaluating method were established to automate functional knowledge integration. A corresponding computer program was developed to prove the feasibility of this approach, and it was used to design a solar-powered wiper blade.

functional knowledge integration, design process, running model, computational algorithm, evaluating method

Citation: Chen B, Xie Y B. Functional knowledge integration of the design process. Sci China Tech Sci, 2017, 60: 209–218, doi: 10.1007/s11431-016-0236-8

1 Introduction

A design process can be described by three kinds of variables, i.e., functional variables relating to the uses of the design product, behavioral variables about what the design product does, and structural variables determining the nature of the design product [1,2]. These three kinds of variables actually correspond to the integration of three kinds of knowledge from abstract to concrete, in that order, i.e., the integration of the functional, behavioral, and structural knowledge [3]. Among them, functional knowledge integration is the starting and core phase of the entire design process. In this phase, the design resources should be combined to construct a functional design scheme based on the functional requirement. This requirement is obtained by the designers during the demand investigation, and the functional design scheme is the rudiment of the following detailed design work. Thus, functional knowledge integration bridges the demand investigation and the detailed design work.

Much previous research about the analysis and modeling of the overall design process has been conducted and constitutes the basic foundation of the study. Pahl et al. [4] decomposed the design process into four stages, i.e., clarification of the task, conceptual design, embodiment design, and detail design. In their research, conceptual design corresponds to functional knowledge integration. In the Axiomatic design approach proposed by Suh [5–7], the world of design can be divided into four domains, i.e., the customer, functional, physical, and process domains. Functional knowledge integration completes the work in the functional domain.

Based on this fundamental research, some studies focusing on functional knowledge integration have been carried out. Chen and Xie [8] introduced control theory into the field of design study. They used a transfer function as the model of the functional unit, and a control block diagram as a method of integrating functional units. Additionally, they

*Corresponding author (email: chenbinsun@outlook.com) also proposed a definition of the stability of the functional design scheme. Vermaas and Dorst [9] used a philosophical method to improve the design model proposed by Gero, and also developed a stable definition of function; however, deciding how to represent the function remained a problem. Chakrabarti and Bligh [10,11] used the input-output method to describe the function, and they used flows to represent an input or output. In their research, the flows are recorded by using only their names. Welch and Dixon [12] promoted the precision by improving the representation of flows with the 7-tuple method. The flow method requires all the inputs and outputs to be classified into three kinds of flows, i.e., material, energy, and signal flows. This classification needs to be refined into a complex classifying system for all the flows of the existing functional units. It is difficult for designers to grasp this complex system. Furthermore, an endless number of new functional units emerge; thus, the system has to be updated constantly. Therefore, the flow method continues to be insufficiently flexible and developable, requiring more advanced analysis and models to be proposed for functional knowledge integration.

With the progress in Internet technology, numerous design resources distributed within different regions and disciplines can be consulted by the designers and used in the design process. These resources actually constitute a huge distributed resource environment. Introduction of this environment into functional knowledge integration would largely promote design innovation. For example, Chen et al. [13,14] introduced multiple design resources from different disciplinary domains into the traditional mechanism design field and successfully designed an offshore wave energy converter array. However, the introduction of this large number of design resources would increase the workload of data processing considerably. The data processing speed would therefore need to be increased by developing the automation of functional knowledge integration. Umeda et al. [15] proposed a computer tool to support functional design in the analytical and synthetic phases. Lin [16] proposed an automatic approach to complete the function identification in the conceptual design of mechanistic systems. Kota and Chiou [17,18] also proposed the automatic conceptual design of mechanisms. Although this research can only partly achieve the design automation of some special disciplinary domains, they continue to remain very inspiring.

Therefore, we analyzed the functional knowledge integration of the design process in detail, and also proposed a corresponding running model. Based on this model, functional knowledge integration can be completed by the proposed computational algorithm and evaluating method. To prove the feasibility of this proposed approach, a corresponding computer program named the functional knowledge integrating system (FKIS) was established, and was then used to design a solar-powered wiper blade. The entire research project consists of five parts, i.e., analysis of functional knowledge integration of the design process, the running model, the computational algorithm, the evaluating method, and application and implementation. Our work is presented in the remainder of this paper. Finally, we provide some discussions and an analysis. The paper is concluded with a plan for future work.

2 Analysis of functional knowledge integration of the design process

2.1 Three inferences about the essence of design

During the design process, all the required design knowledge can be classified into two categories, i.e., existing knowledge and new knowledge yet to be discovered. Existing knowledge largely influences the obtainment of new knowledge in many aspects, such as the obtaining method, the species of the new knowledge, and the disciplinary domains of the new knowledge. Therefore, based on existing knowledge, designers can obtain new knowledge and integrate these two kinds of knowledge into a design scheme for a product that has never existed before. This is the core task of the design process, and here, all the knowledge mentioned above should be selected by the designers, such that the designers can discover suitable knowledge for the functional requirements of the product. That means, if the design knowledge does not meet the functional requirement, it cannot be used in the design process. However, this design knowledge can be decomposed into some smaller pieces of knowledge, and, if one of these pieces meets the functional requirement, this design knowledge can also be used in the design process, but only partly.

All the discussions above can be concluded into the following three inferences.

1) Design is a process of knowledge integration.

2) Meeting the functional requirement is the first priority during knowledge integration.

3) Design knowledge should be as little as possible to insure the efficiency of the knowledge integration.

2.2 Three levels of knowledge integration

Knowledge integration can be divided into three levels based on a descending order of abstraction, i.e., the conceptual, systematic, and assembly level. The knowledge integrated in these three levels can be classified into three categories, i.e., functional knowledge at the conceptual level, behavioral knowledge at the systematic level, and structural knowledge at the assembly level. Through these three levels of knowledge integration, the final design scheme emerges from the design knowledge. Here, functional knowledge integration is the starting point of the design process, and it is also the key phase to ensure that the functional requirement can firstly be met. That means during this phase, a functional design scheme should be generated as the rudiment of the final design scheme, and it should meet the functional requirement. If no appropriate functional design scheme is generated, the subsequent processes involving the integration of behavioral and structural knowledge would lose direction.

2.3 Functional design scheme

As mentioned before, the key of functional knowledge integration is to generate a functional design scheme that meets the functional requirement. As shown in Figure 1(a), a functional requirement is represented by its inputs and outputs. That means it requires the design product to be able to complete the transformation from the inputs to the outputs.

A functional design scheme can be determined to meet the functional requirement based on its inputs and outputs. If they satisfy the following two conditions, the determination can be made.

1) The inputs of the functional requirement contain all the inputs of the functional design scheme.

2) The outputs of the functional design scheme contain all the outputs of the functional requirement.

As shown in Figure 1(b), the functional requirement is to design a cake-producing system that can transform its five inputs into its single output. Moreover, the four inputs of an appropriate functional design scheme are all contained by the inputs of the functional requirement, and its three outputs also contain the single output of the functional requirement. This example shows that for an appropriate functional design scheme, not all the inputs of the functional requirement are needed, such as Input 5 shown in the box with a dotted outline, and not all the outputs of the functional design scheme are useful, such as Output 2 and Output 3 shown in boxes with dotted outlines.

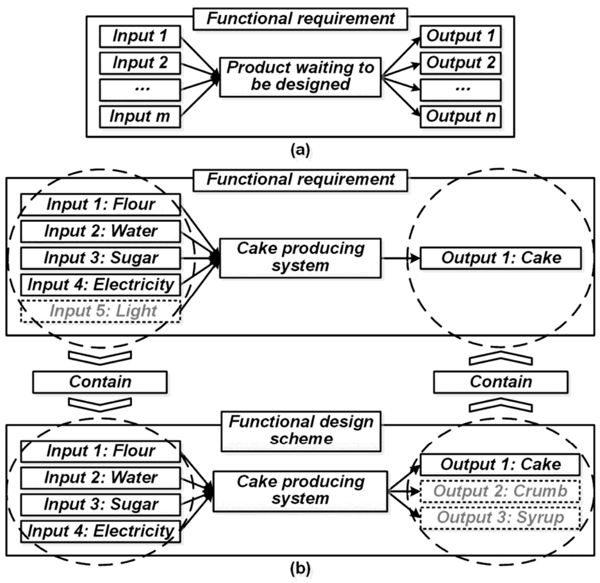


Figure 1 (a) Representation of a functional requirement; (b) the functional design scheme should meet the functional requirement.

In the next part of this paper, a running model for the functional knowledge integration is proposed. In this model, all the functional knowledge is modeled as functional units. They are as small as possible, and we treated them as elements of functional knowledge integration. Every functional unit may have several practical entities. These functional units and practical entities are distributed throughout different regions and disciplines. They form a distributed resource environment that we introduced into functional knowledge integration.

3 Running model

3.1 Functional unit

A functional unit is used to model functional knowledge, and it is also the operating element during the integration of functional knowledge. A functional unit is represented by its inputs and outputs, which means its essence is the transformation from its inputs to its outputs. Based on the relationships of the inputs and outputs, several functional units can be connected to each other in series to construct a functional unit chain which is also a functional design scheme. Here, these functional units may originate from an existing functional design scheme. As shown in Figure 2, a cake-producing system needs to be designed. In this case, the functional requirement comprises four inputs, i.e., flour, water, sugar, and electricity, and it has only one output, the cake. To achieve this transformation, a functional unit capable of transforming a velocity into several possible velocities is needed. We can analyze the existing functional design schemes to determine the alternative functional units. Based on the analysis, an automobile can be decomposed into many functional units, such as the chassis, carriage body,

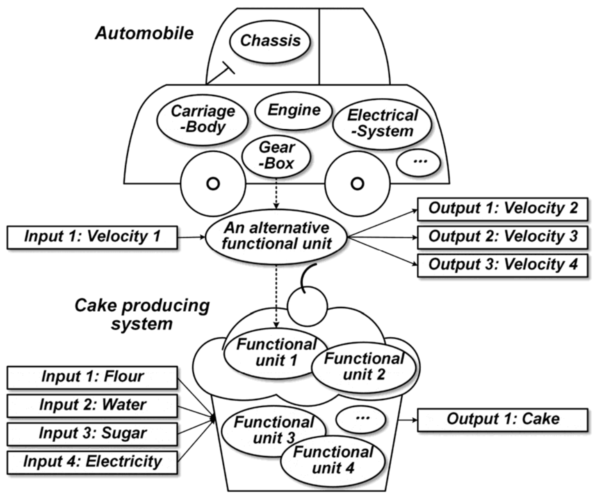


Figure 2 An alternative functional unit can be found in some existing functional design schemes.

engine, gearbox, and electrical system. Among them, the gearbox is an alternative functional unit, such that, it can be used in the design process.

A functional unit may have several practical entities in the physical world. They may have different behavior and structures, but they can all achieve the same transformation from the inputs to the outputs of their corresponding functional unit. That means designers can have a wider range of choices and the practical entities would have to compete with each other for the preference of the designers. A practical entity has many features, all of which should be considered during the design process, such as its height, length, width, weight, date of delivery, cost, and emission. The designers should select the practical entities based on these features, to enable them to select the optimal alternative.

3.1.1 Representation

A functional unit is represented by its inputs and outputs, which are described by the functional unit provider. Only when this representation is satisfactorily understood by the functional unit users, can this functional unit be fully evaluated and considered for the design. Thus, the more precise and detailed the representation is, the higher the efficiency of the functional knowledge integration will be. However, if the representation is too precise and detailed, it is no longer suitable to be applied to a computationally achieving algorithm for functional knowledge integration, because the algorithm needs the representation to be sufficiently concise.

Some researchers adopted the concept of flow to describe the inputs and outputs [12,19]. In this method, the inputs and outputs can be treated as three categories of flows, i.e. material, energy, and signal flows. Furthermore, the researchers proposed a complex classifying system for these three kinds of flows. However, this system is very complicated. For example, there are even four classifying levels for a single electromechanical product. Similar to the AC-motor, it is classified in a category represented by four levels, i.e., e.m.k.r (energy.mechanical-energy.kinetic-energy.rotation) [20]. If the range of the functional units extends to mental and social functional units, the inputs and outputs may be plants, animals, human beings, human-made systems, human-constructed systems, etc. In this situation, the classifying system needs to be constantly maintained to adapt the increasingly complicated flow range, which means an enormous input in terms of human effort and cost. Furthermore, new functional units emerge continually, thus the classifying system needs to be updated at any time. This is another significant task for the maintainers. In addition, the users also need to update their understanding of the classifying system due to its increasing complicatedness and the rapid changes it undergoes.

Solving these problems first requires the concept of flow to be removed. This concept assumes that the inputs proceed to the functional unit, and that the outputs emerge from the functional unit, but sometimes this assumption is wrong. Most of the time, the inputs and outputs simply interact with their functional unit, and they neither enter it nor emerge from it. Therefore, we used keywords to represent the input and output. Here, keywords are represented by the natural language, so based on them, the functional unit providers and users can all accurately describe and search for the functional units without the above flow classifying system. That means all the problems caused by the flow concept and the corresponding classifying system can be solved.

However, for any one input or output, if there is merely one keyword, the functional unit representation is not sufficiently precise. As shown in Table 1, this is a representation with only one keyword for each input and output. This representation has a large number of different practical entities corresponding to the same functional unit, which means this representation has no practical meaning for classification. In this case, the objectives of the actions and reactions may be a solid, liquid, gas, microorganism, plant, animal, human being, human-made system, human-constructed system, etc. The actions and reactions may be a force action, thermal action, optical action, acoustical action, electrical action, etc. Therefore, the representation should be more detailed to improve discrimination among the functional units.

Therefore, we increased the number of the keywords such that one input or output can have 1 or 2 keywords. This improves the precision of the representation. For example, Table 1 can be improved to Table 2. The new representation largely decreases the number of practical entities corresponding to the same functional unit.

A functional unit may have many corresponding practical entities, where every entity can achieve the same objective function, but they may have different detailed features. These features can be classified into three categories, i.e., input features, output features, and entity features, as shown in Table 3.

Table 1 Representation with only one keyword for each input and output

Functional unit	Input	Output	Practical entity
FU_1 [a)]	Action	Reaction (preventing the changing trend caused by the action)	Beam Roadway Dyke Blanket Curtain

a) In this paper, "FU" is the abbreviation of "functional unit".

Table 2 Representation with 1 or 2 keywords for each input and output

Functional unit	Input	Output	Practical entity
FU_1	Gravity, Tiles	Equilibrant	Beam
FU_2	Gravity, Vehicles	Equilibrant	Roadway
FU_3	Water, Pressure	Equilibrant	Dyke
FU_5	Human, Heat	Thermal, Resistance	Blanket
FU_6	Sunshine	Photo, Resistance	Curtain

Table 3 Detailed features of a practical entity

Input features			
Input keywords	Feature name	Feature value	Feature unit
AC, Electricity	Voltage	220	V
	Power	[1000, 3000]	W
	Frequency	50	Hz
Output features			
Output keywords	Feature name	Feature value	Feature unit
Light	Luminous flux	600	lm
	Color temperature	6500	K
Entity features			
Feature name	Feature value	Feature unit	
Price	30	CYN	
Operating cost	1.34	CYN/h	
Power waste	7	W	
Weight	0.1	kg	

The information about the detailed features of the practical entities also forms part of the representation of the corresponding functional unit, but it should be considered after the final functional units being selected. That means the functional knowledge integration can be divided into two steps as follows.

Step 1. Search and combine appropriate functional units.

Step 2. Select the optimal practical entities for these functional units.

Here, the first step can be completed by the proposed computational algorithm, and the second step can be completed by the proposed evaluating method. These two parts of the work are introduced later.

3.1.2 Classification

During the design process, there may be numerous functional units the designers would have to consider. However, they may not all be useful because the functional requirement only requires the functional units in some specific domains. In order to decrease the searching redundancy and promote the efficiency, the functional units should be classified.

As mentioned before, the essence of a functional unit is the transformation from its inputs to its outputs. Based on this, functional units can be classified into four functional types, i.e., transforming, supporting, storing, and stimulating functions. This classification closely approximates the natural language description for the function, such that functional unit providers and users can all accurately describe and search for the functional units. The detailed rule of this classification is as follows.

1) Transforming function. The inputs enter their functional unit, and the outputs emerge from the functional unit. This functional unit achieves the function of transforming one unit into another.

2) Supporting function. The inputs and outputs are both actions. The output actions prevent the change or the changing trend caused by the input actions.

3) Storing function. The inputs enter the functional unit, after which the functional unit stores them and outputs them in their initial form. The output quantity has no need to be equal to the input quantity but may be less than it.

4) Stimulating function. The inputs and outputs are both actions. The input actions stimulate the change or the changing trend caused by the output actions.

Functional units can also be classified into three requirement types according to the domain of their functional requirements. They are physical, mental, and social requirements. This classification leads the functional unit providers to send their functional units into domains of the corresponding type, and it also leads the functional unit users to search for the functional units in the correct type of domains.

3.2 Distributed resource environment

3.2.1 Constitution

As the materials of functional knowledge integration, the numerous functional units are distributed in different regions and disciplines, as are their practical entities. All these design resources can actually be used to construct a distributed resource environment. During functional knowledge integration, the designers need to search for and combine the appropriate functional units, and then choose suitable practical entities. During this selecting process, all the functional units and practical entities should compete with each other for the preference of the designers, which is the fundamental power of the evolution of functional units and practical entities as shown in Figure 3.

As shown in the dotted circles in Figure 3, a functional unit may not have any practical entity. However, for these functional units, their providers have the ability to create the corresponding practical entities immediately, if the users ask for them. This is a kind of knowledge service, it is the representative of the ability of the functional unit providers, and it transforms the tacit design resources into explicit ones, such that the range of the functional units can be extended and the functional unit users would have a greater number of choices for their designs.

3.2.2 Data structure—Function unit graph

(1) Connection between two function units

As mentioned before, functional knowledge integration can be divided into two steps, i.e., search for and combine the appropriate functional units, and select the optimal practical entities for these functional units. During the first step, these functional units should be connected in a series as a functional unit chain. This connection is completed in the distributed resource environment, which means this environment should have a data structure to store the relationships among the functional units. In this structure, two functional units, such as FU_1 and FU_2, should be connected if their

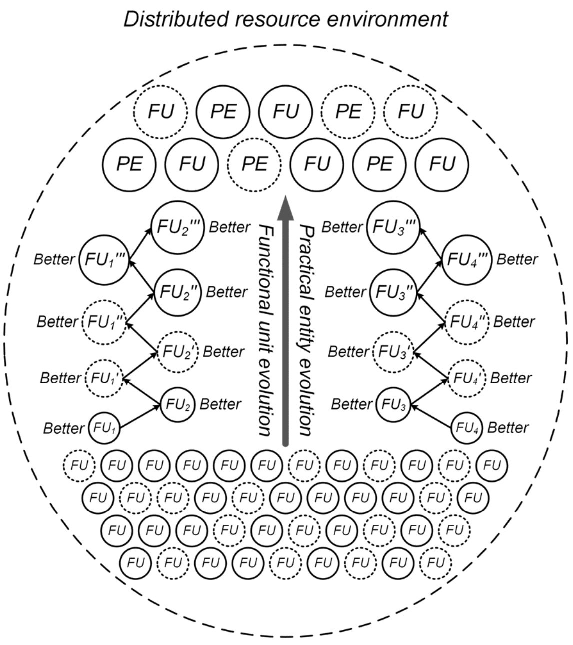

Distributed resource environment
Functional unit evolution
Practical entity evolution
Better

Figure 3 Evolution of the functional units and practical entities in the distributed resource environment.

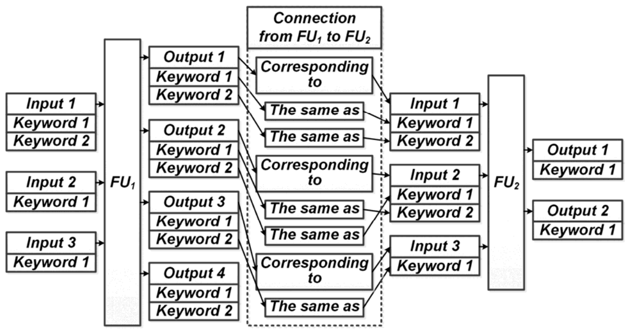


Figure 4 Connection between two functional units.

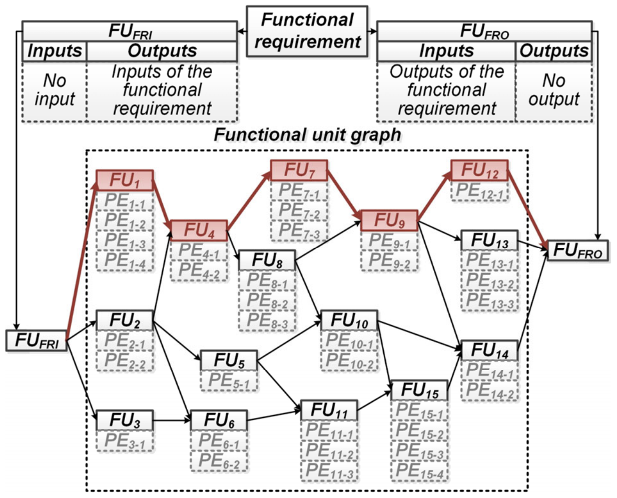


Figure 5 (Color online) Functional unit graph and the aim of the computational algorithm.

inputs and outputs meet the following two conditions.

1) For every input of FU$_2$, there is a corresponding output of FU$_1$.

2) For each keyword of this input of FU$_2$, the corresponding output of FU$_1$ always shares this same keyword.

If the above two conditions can be met simultaneously, we can determine that FU$_1$ is connected to FU$_2$, as shown in Figure 4.

(2) Function unit graph

Based on the concept of the connection between the two functional units mentioned above, we can connect all the functional units in the distributed resource environment to construct the functional unit graph as shown in Figure 5. As the data structure of the distributed resource environment, this functional unit graph stores all the connecting relationships of the component functional units. These connections are represented as edges from one functional unit to another. For example, the edge from FU$_8$ to FU$_{10}$ represents that FU$_8$ is connected to FU$_{10}$. For the convenience of the following discussions, we designated FU$_8$ as the precursor of FU$_{10}$, and considered FU$_{10}$ as the successor of FU$_8$, and simultaneously, we designated the edge from FU$_8$ to FU$_{10}$ as the successor edge of FU$_8$, and also considered it as the precursor edge of FU$_{10}$.

4 Computational algorithm

4.1 Function and aim

The first step of functional knowledge integration is searching for and combining the appropriate functional units. These functional units should be connected in series as a functional unit chain, and this chain should meet the functional requirement.

As shown in Figure 5, we divided the functional requirement into two special functional units, FU$_{\mathrm{FRI}}$ and FU$_{\mathrm{FRO}}$. FU$_{\mathrm{FRI}}$ has no input and it processes the inputs of the functional requirement as its outputs. Thus, it can only be connected to other functional units, and no functional unit can be connected to it. As for FU$_{\mathrm{FRO}}$, the situation is the contrary. Based on this, the relationship between the functional requirement and the functional unit graph can be established. A functional unit chain can be determined to meet the functional requirement only if it can satisfy the following two conditions.

(1) FU$_{\mathrm{FRI}}$ is connected to the first functional unit of this chain.

(2) The last functional unit of this chain is connected to FU$_{\mathrm{FRO}}$.

As shown in Figure 5, FU$_{\mathrm{FRI}}$ is connected to FU$_1$, and FU$_{12}$ is connected to FU$_{\mathrm{FRO}}$; hence, this functional unit chain, FU$_1$→FU$_4$→FU$_7$→FU$_9$→FU$_{12}$, meets the functional requirement.

Now, we can conclude the mission of the first step of functional knowledge integration into generating functional unit chains from the functional unit graph. The aim of the computational algorithm is to complete this mission. During this step, there is no need to consider the practical entities of the functional units; thus, in Figure 5, they are all represented in dotted boxes.

4.2 Principle

This computational algorithm is intended to generate the appropriate functional unit chain (the functional unit chain meeting the functional requirement) from the functional unit graph. For the convenience of the following introduction of the basic principle of this algorithm, we took the functional unit graph shown in Figure 5 as an example.

This computational algorithm has two steps, i.e., an operation to determine all the successors of the current functional units, and the retrospect to generate the functional unit chains.

Step 1. As shown in Figure 6, for the first step, the operations always begins from $\mathrm{FU}_{\mathrm{FRI}}$, and ends with $\mathrm{FU}_{\mathrm{FRO}}$. As for this case, it needs the following six operations.

Operation 1. There is only one current functional unit, i.e., $\mathrm{FU}_{\mathrm{FRI}}$. Thus, find all its successors, i.e., FU_1, FU_2, and FU_3. However, none of them is connected to $\mathrm{FU}_{\mathrm{FRO}}$, so continue with Operation 2.

Operation 2. There are three current functional units, i.e., FU_1, FU_2, and FU_3. Thus, find all their successors, i.e., FU_4, FU_5, and FU_6. However, none of them is connected to $\mathrm{FU}_{\mathrm{FRO}}$, so continue with Operation 3.

Operation 3. There are three current functional units, i.e., FU_4, FU_5, and FU_6. Thus, find all their successors, i.e., FU_7, FU_8, FU_{10}, and FU_{11}. However, none of them is connected to $\mathrm{FU}_{\mathrm{FRO}}$, so continue with Operation 4.

Operation 4. There are four current functional units, i.e., FU_7, FU_8, FU_{10}, and FU_{11}. Thus, find all their successors, i.e., FU_9, FU_{10}, FU_{14}, and FU_{15}. Not all of them are connected to $\mathrm{FU}_{\mathrm{FRO}}$, so continue with Operation 5. Here, FU_{14} is connected to $\mathrm{FU}_{\mathrm{FRO}}$, so stop finding its successors in Operation 5.

Operation 5. There are three current functional units (FU_{14} is excluded), i.e., FU_9, FU_{10}, and FU_{15}. Thus, find all their successors, i.e., FU_{12}, FU_{13}, FU_{14}, and FU_{15}. Not all of them are connected to $\mathrm{FU}_{\mathrm{FRO}}$, so continue with Operation 6. Here, FU_{12}, FU_{13}, and FU_{14} are connected to $\mathrm{FU}_{\mathrm{FRO}}$, so stop finding their successors in Operation 6.

Operation 6. There is only one current functional unit (FU_{12}, FU_{13}, and FU_{14} are excluded), i.e., FU_{15}. Thus, find all its successors, i.e., FU_{14}. Now, FU_{14} is connected to $\mathrm{FU}_{\mathrm{FRO}}$, which means all the current functional units are connected to $\mathrm{FU}_{\mathrm{FRO}}$. Thus, terminate this operation.

Step 2. This step is the retrospect to generate the functional unit chains. A retrospect begins from $\mathrm{FU}_{\mathrm{FRO}}$, and ends with $\mathrm{FU}_{\mathrm{FRI}}$. It should run only in one operation, and it should traverse all the precursors of the current functional

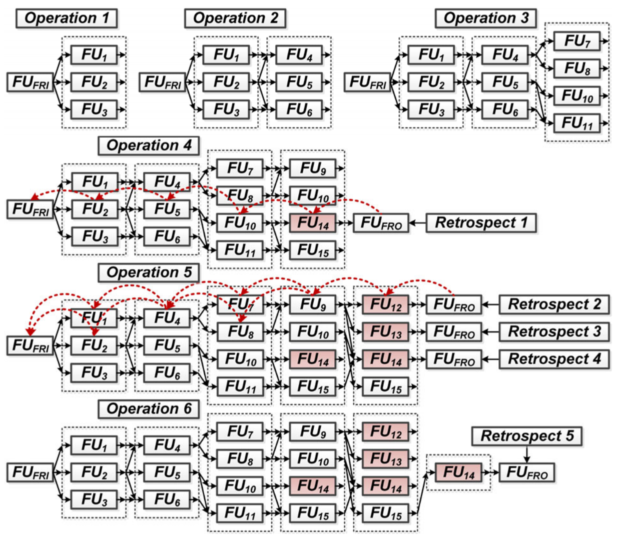


Figure 6 (Color online) Principle of the computational algorithm.

unit, and every precursor corresponds to a functional unit chain. That means if there are n functional units which have been tested, i.e., FU_1, FU_2, …, FU_n, and their precursor numbers are P_{FU_1}, P_{FU_2}, …, P_{FU_n}, the number of the generated functional unit chain, N_{FUC}, can be calculated based on the following equation.

$$N_{\mathrm{FUC}} = \prod_{i=1}^{n} P_{\mathrm{FU}_i}. \tag{1}$$

As shown in Figure 6, these two groups of dotted arrows represent Retrospect 1 and 2. Retrospect 1 runs only in Operation 4, and starts from $\mathrm{FU}_{\mathrm{FRO}}$. $\mathrm{FU}_{\mathrm{FRO}}$ has only one precursor, FU_{14}. FU_{14} only has one precursor, FU_{10}. FU_{10} has only one precursor, FU_5. FU_5 has only one precursor, FU_2. The precursor of FU_2 is $\mathrm{FU}_{\mathrm{FRI}}$, which is also the end of this retrospect. The number of generated functional unit chains in this retrospect can be calculated using the following equation:

$$N_{\mathrm{FUC}} = P_{\mathrm{FU}_{14}} P_{\mathrm{FU}_{10}} P_{\mathrm{FU}_5} P_{\mathrm{FU}_2} = 1\times1\times1\times1 = 1. \tag{2}$$

This functional unit chain comprises $\mathrm{FU}_2 \rightarrow \mathrm{FU}_5 \rightarrow \mathrm{FU}_{10} \rightarrow \mathrm{FU}_{14}$.

As for Retrospect 2, it runs only in Operation 5, and it starts from $\mathrm{FU}_{\mathrm{FRO}}$, which has only one precursor, FU_{12}. FU_{12} has only one precursor, FU_9, which has two precursors, FU_7 and FU_8. FU_7 has only one precursor, FU_4. FU_8 has only one precursor, FU_4, which has two precursors, FU_1 and FU_2. FU_1 and FU_2 have the same precursor, $\mathrm{FU}_{\mathrm{FRI}}$, which is the end of this retrospect. The number of generated functional unit chains in this retrospect can be calculated using the following equation:

$$\begin{aligned} N_{\mathrm{FUC}} &= P_{\mathrm{FU}_{12}} P_{\mathrm{FU}_9} P_{\mathrm{FU}_7} P_{\mathrm{FU}_8} P_{\mathrm{FU}_4} P_{\mathrm{FU}_1} P_{\mathrm{FU}_2} \\ &= 1\times2\times1\times1\times2\times1\times1 = 4. \end{aligned} \tag{3}$$

These four functional unit chains are $FU_1 \rightarrow FU_4 \rightarrow FU_7 \rightarrow FU_9 \rightarrow FU_{12}$, $FU_1 \rightarrow FU_4 \rightarrow FU_8 \rightarrow FU_9 \rightarrow FU_{12}$, $FU_2 \rightarrow FU_4 \rightarrow FU_7 \rightarrow FU_9 \rightarrow FU_{12}$, and $FU_2 \rightarrow FU_4 \rightarrow FU_8 \rightarrow FU_9 \rightarrow FU_{12}$.

5 Evaluating method

This evaluating method is intended to complete the second step of functional knowledge integration, i.e., select the optimal practical entities for the component functional units of the functional unit chain. This method can be divided into two steps, i.e., feasibility analysis for testing the feasibility of the practical entity chains, and optimization for selecting the optimal feasible practical entity chain as the final functional design scheme.

5.1 Feasibility analysis

As shown in Figure 7, every component functional unit of a functional unit chain may have several practical entities. Not every combination of entities is feasible because the features of the input and output of two neighboring practical entities may not be suitable to each other.

As shown in Figure 7, if we choose PE_{1-2} for FU_1, PE_{2-1} for FU_2, and PE_{3-2} for FU_3, we can obtain a practical entity chain, $PE_{1-2} \rightarrow PE_{2-1} \rightarrow PE_{3-2}$. This chain can be determined to be feasible, only if every pair of corresponding inputs and outputs satisfies the feasibility conditions. For the convenience of expression, we took Output 3 of PE_{1-2} and Input 1 of PE_{2-1} as an example of an input-output pair. They need to satisfy the following conditions.

1) For every feature of Input 1 of PE_{2-1}, there is a corresponding feature of Output 3 of PE_{1-2}.

2) For every pair of these corresponding features, the two features should have the same name and unit, and the value of the feature of Output 3 of PE_{1-2} should be contained by the value of the feature of Input 1 of PE_{2-1}.

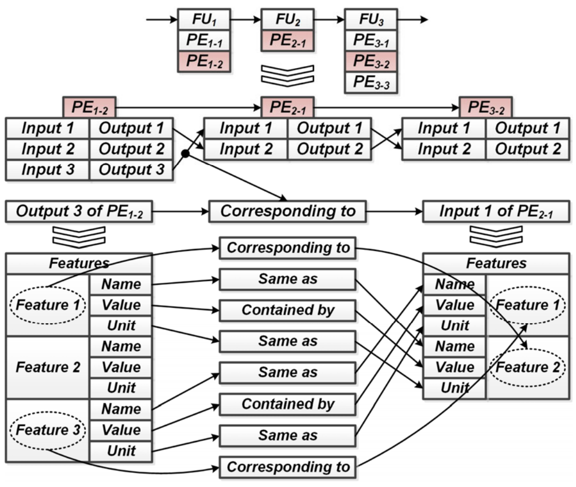


Figure 7 (Color online) Feasibility analysis.

5.2 Optimization

A functional unit chain may have several feasible practical entity chains. Selecting the optimal chain is the objective of this part of the work.

As shown in Figure 8, this feasible practical entity chain consists of three practical entities, i.e., PE_{1-2}, PE_{2-1}, and PE_{3-2}. Every entity has several entity features, i.e., price, weight, volume, emission, and lifetime. These features enable us to calculate the total features of this practical entity chain. For example, the total price can be calculated by summing up the prices of all the entities. The total weight, volume, and emission can be determined similarly. In addition, there are also other calculating operations besides summation, such as the total lifetime, which should be calculated by minimizing the lifetimes of all the entities.

The designers can select the optimal feasible practical entity chain based on the evaluating feature. For example, if we take price as the evaluating feature, we can select the optimal entity chain based on Table 4. Here, the optimal feasible chain should be Chain 6 whose total price is the smallest.

6 Application and implementation

We tested the feasibility of this proposed functional knowledge integration by establishing a corresponding computer program named functional knowledge integrating system (FKIS). Furthermore, a solar-powered wiper blade was designed using this system.

As shown in Figure 9, we should first input the information of the functional requirement. As for this design

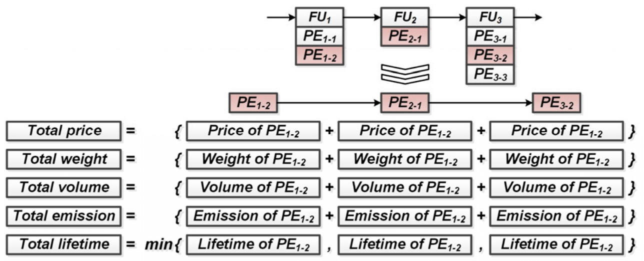
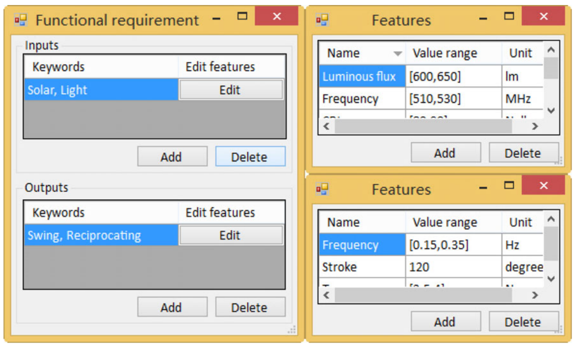

FU1
PE1-1
PE1-2
FU2
PE2-1
FU3
PE3-1
PE3-2
PE3-3
PE1-2
PE2-1
PE3-2
Total price = { Price of PE1-2 + Price of PE1-2 + Price of PE1-2 }
Total weight = { Weight of PE1-2 + Weight of PE1-2 + Weight of PE1-2 }
Total volume = { Volume of PE1-2 + Volume of PE1-2 + Volume of PE1-2 }
Total emission = { Emission of PE1-2 + Emission of PE1-2 + Emission of PE1-2 }
Total lifetime = min{ Lifetime of PE1-2 , Lifetime of PE1-2 , Lifetime of PE1-2 }

Figure 8 (Color online) Calculate the total features of the practical entity chain.

Table 4 Total prices of the alternative practical entity chains

Practical entity chain	Practical entities			Feasibility	Total price
	FU1	FU2	FU3		
Chain 1	PE_{1-1}	PE_{2-1}	PE_{3-1}	YES	2331 CYN
Chain 2	PE_{1-1}	PE_{2-1}	PE_{3-2}	YES	3432 CYN
Chain 3	PE_{1-1}	PE_{2-1}	PE_{3-3}	NO	–
Chain 4	PE_{1-2}	PE_{2-1}	PE_{3-1}	YES	2321 CYN
Chain 5	PE_{1-2}	PE_{2-1}	PE_{3-2}	NO	–
Chain 6	PE_{1-2}	PE_{2-1}	PE_{3-3}	YES	1234 CYN


Functional requirement

Inputs

Keywords	Edit features
Solar, Light	Edit

Add Delete

Outputs

Keywords	Edit features
Swing, Reciprocating	Edit

Add Delete

Features

Name	Value range	Unit
Luminous flux	[600,650]	lm
Frequency	[510,530]	MHz

Add Delete

Features

Name	Value range	Unit
Frequency	[0.15,0.35]	Hz
Stroke	120	degree

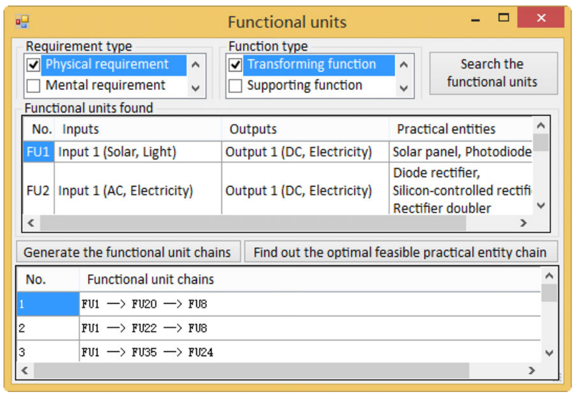
Add Delete

Figure 9 (Color online) Input the information of the functional requirement and its inputs and outputs.

mission, we should design a product with the ability to transform solar light into reciprocating swing. Thus, there is only one input and one output for this functional requirement. The input keywords can be "Solar" and "Light," and the output keywords can be "Swing" and "Reciprocating." Input these keywords, and click the button "Edit" to edit the information of the input and output features. Subsequently, click the button "Next" to continue to the next step to search for functional units from the distributed resource environment.

As shown in Figure 10, based on the design mission, we can select the functional units from "Physical requirement" and "Transforming function" to narrow down the functional unit scope. This is helpful for promoting the running efficiency of the computational algorithm that is used next. After clicking the button "Search the functional units", all the functional units found are listed in the table. Click the button "Generate the functional unit chain", such that the functional unit chains are generated and listed in the table. We can select the first one, $FU_1 \rightarrow FU_{20} \rightarrow FU_8$, and then, click the button "Find out the optimal feasible practical entity chain" to continue.

As shown in Figure 11, we are first required to select an evaluating feature. In this case, we select "Price" as the evaluating feature. Clicking the button "Analyze" lists the


Functional units

Requirement type: [✓ Physical requirement] Mental requirement

Function type: [✓ Transforming function] Supporting function

Search the functional units

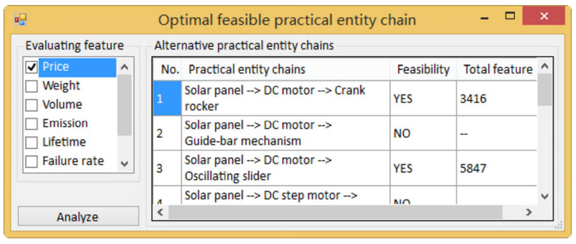
Functional units found

No.	Inputs	Outputs	Practical entities
FU1	Input 1 (Solar, Light)	Output 1 (DC, Electricity)	Solar panel, Photodiode
FU2	Input 1 (AC, Electricity)	Output 1 (DC, Electricity)	Diode rectifier, Silicon-controlled rectifi Rectifier doubler

Generate the functional unit chains Find out the optimal feasible practical entity chain

No.	Functional unit chains
1	FU1 —> FU20 —> FU8
2	FU1 —> FU22 —> FU8
3	FU1 —> FU35 —> FU24

Figure 10 (Color online) Search functional units from the distributed resource environment and generate the functional unit chains.


Optimal feasible practical entity chain

Evaluating feature: [✓ Price] Weight Volume Emission Lifetime Failure rate

Analyze

Alternative practical entity chains

No.	Practical entity chains	Feasibility	Total feature
1	Solar panel --> DC motor --> Crank rocker	YES	3416
2	Solar panel --> DC motor --> Guide-bar mechanism	NO	--
3	Solar panel --> DC motor --> Oscillating slider	YES	5847
4	Solar panel --> DC step motor -->	NO	

Figure 11 (Color online) Select the optimal feasible practical entity chain.

corresponding information of the alternative practical entity chains in the table.

Scanning the information of these practical entity chains enables us to select the optimal and feasible chain, "(Solar panel)→(DC motor)→(Crank rocker)", with the lowest price, "3416 CYN" .

7 Discussion

Compared with the other design approaches and tools, this proposed approach has some special advantages.

1) This approach focuses on the starting and core phase of the design process, functional knowledge integration. Its result is the rudiment of the final design scheme, which means this approach can directly affect the source of the flow of design ideas, and readily introduce innovative thoughts and lucrative design resources.

2) This approach introduces the distributed resource environment into the design process. This improvement not only increases the number of innovative design ideas, but also overcomes the limitations imposed by a designer's cognitive domain.

3) This approach can smoothly bridge the investigation of customer demands and detailed design work. Designers should first investigate customer demands and abstract them into functional requirements. These functional requirements will be completely met by the optimal feasible practical entity chain generated by this approach.

4) This approach can achieve the automation of functional knowledge integration, during which the main workload is distributed across two processes. The first is to search for appropriate functional units from the distributed resource environment and combine them into functional unit chains. The second is to traverse all the possible practical entity chains for a selected functional unit chain, test their feasibilities, and determine their overall features. These two tasks can be completed by the computational algorithm and the evaluating method, both of which have been programed into FKIS. This system can be operated on a PC as shown in the above application and implementation.

5) The running model used in this approach supports multiple inputs and outputs in a functional unit. In many previous research studies, a functional unit was only al-

lowed to have one input and one output. This precondition is proposed for the convenience of establishing the concept of the connection between two functional units, but it is not suitable for actual product design. Therefore, this approach proposes a new concept of functional unit connection to enable a functional unit to contain multiple inputs and outputs.

6) The running model used in this approach provides a flexible representation of functional unit inputs and outputs. This representation provides a large space for the imagination, which helps the functional unit providers to free their thoughts when they describe their functional units.

7) The running model used in this approach provides a concise classification of the functional units which is helpful to narrow down the scope of the functional unit for the computational algorithm.

8 Conclusions

This paper presents a detailed analysis of the functional knowledge integration of the design process, and proposes a corresponding running model. Based on this model, functional knowledge integration can be completed in two steps, i.e., search for and combine the appropriate functional units from the distributed resource environment, and select the optimal practical entities for these functional units. These two steps can be completed by the proposed computational algorithm and evaluating method, respectively. To prove the feasibility of this proposed approach, a corresponding computer program named functional knowledge integrating system (FKIS) was established, and then used to design a solar-powered wiper blade.

Although this approach can finally complete the functional knowledge integration of the design process, there remains room for improvement. For example, further study could be devoted to the competition and evolution of the functional units and the practical entities during functional knowledge integration. Moreover, the efficiency of the computational algorithm can also be promoted by conducting a more in-depth analysis of the functional unit connections. Our future work aims to take these aspects into consideration.

This work was supported by the National Natural Science Foundation of China (Grant No. 51575342).

1 Gero J S. Design prototypes: A knowledge representation schema for design. AI Mag, 1990, 11: 26–36

2 Gero J S, Kannengiesser U. The situated Function-Behaviour-Structure framework. Design Stud, 2004, 25: 373–391

3 Chen B, Xie Y B. A computer-assisted automatic conceptual design system for the distributed multi-disciplinary resource environment. P I Mech Eng C-J Mec Eng Sci, 2016, doi: 10.1177/0954406216638886

4 Pahl G, Beitz W, Feldhusen J, et al. Engineering design: A systematic approach. Nasa Sti/Recon Tech Rep A, 2007, 89: 63–64

5 Suh N P. Designing-in of quality through axiomatic design. IEEE T Reliab, 1995, 44: 256–264

6 Suh N P. Axiomatic design theory for systems. Res Eng Des, 1998, 10: 189–209

7 Suh N P. Axiomatic Design: Advances and Applications. New York: Oxford University Press, 2001

8 Chen B, Xie Y B. A complex-number-domain-based conceptual design synthesis for multidisciplinary products. P I Mech Eng C-J Mec, 2016, doi: 0954406216668207

9 Vermaas P E, Dorst K. On the conceptual framework of John Gero's FBS-model and the prescriptive aims of design methodology. Design Stud, 2007, 28: 133–157

10 Chakrabarti A, Bligh T P. An approach to functional synthesis of solutions in mechanical conceptual design, part I: Introduction and knowledge representation. Res Eng Des, 1994, 6: 127–141

11 Chakrabarti A, Bligh T P. An approach to functional synthesis of solutions in mechanical conceptual design, part II: Kind synthesis. Res Eng Des, 1996, 8: 52–62

12 Welch R V, Dixon J R. Guiding conceptual design through behavioral reasoning. Res Eng Des, 1994, 6: 169–188

13 Chen W, Gao F, Meng X, et al. An offshore hydraulic wind turbine generator with variable-diameter rotor: Design, modeling and experiment. P I Mech Eng M-J Eng Maritime Environ, 2016, doi: 1475090216661449

14 Chen W, Gao F, Meng X, et al. Design of the wave energy converter array to achieve constructive effects. Ocean Eng, 2016, 124: 13–20

15 Umeda Y, Ishii M, Yoshioka M. Supporting conceptual design based on the function-behavior-state modeler. AI EDAM, 1996, 10: 275–288

16 Lin T. An approach to function identification in automated conceptual design of mechanism systems. Res Eng Des, 2008, 19: 151–159

17 Kota S, Chiou S J. Conceptual design of mechanisms based on computational synthesis and simulation of kinematic building blocks. Res Eng Des, 1992, 4: 75–87

18 Chiou S J, Kota S. Automated conceptual design of mechanisms. Mech Mach Theory, 1999, 34: 467–495

19 Chakrabarti A. A new approach to structure sharing. J Comput Inform Sci Eng, 2004, 4: 165–170

20 Li X, Zhang Z N, Liu Z L, et al. A novel semi-heuristic planning approach for automated conceptual design synthesis. P I Mech Eng C-J Mec Eng Sci, 2013, 227: 2291–2305