1 INTRODUCTION

Mechatronic system design is complex since it involves knowledge of different disciplines. As the core of model-based systems engineering (MBSE) (Estefan, 2008), system architecture is essential in system design. It influences the whole system quality, the subsequent detail design and the design cost. In resource-limited society, how to generate a feasible and optimal system architecture has attracted many scholars' attention. As stated by Ulrich (1995), the architecture contains three parts:

- The arrangement of functional elements;
- The mapping from functional elements to physical components;
- The specification of the interfaces among interacting physical components.

Although there are a lot of researches on system architecture generation, some deficiencies still exist.

- Most of the mapping from function elements to physical components is heavily based on designers' experience (the existing products). The innovation is insufficient. The matching degree between function and its mapping component is not considered.
- There should be different evaluation criterion for different kinds of components. According to detailed requirements, different criteria have different weights in distinct design process of the same product. How to evaluate the suitability of a component in realizing the corresponding function is not yet studied.
- Compatible information between components is not considered in most system architecture generation processes. The infeasible system architectures are always excluded after the generation process. This kind of method may cause combination explosion and it is not efficient.

In this paper, a new system architecture generation method is proposed to solve the above problems. Therefore, three necessary procedures are studied in this study.

- Two kinds of function-component mapping method are proposed to ensure finding all available components that can realize the function;
- Components that can realize the same function are evaluated using AHP (Saaty, 1990) in order to evaluate components under specific requirement and various criteria;
- A new component combination method based on dynamic programming is proposed to solve component combination and inefficient problem.

The generation method is applied to the automobile design to verify its feasibility. In system design, the automobile is abstracted with its main functions, then the feasible physical architectures are obtained to realize these functions through the function-component mapping, the component selection and evaluation, and the physical architecture generation processes.

2 RELATED WORK

There are some researches about system architecture generation and evaluation over the past years. According to Wyatt (2012), methods to support product architecture design can be divided into informal methods and formal methods. The informal methods like branstorming (Osborn, 1957), which rely on human creativity, always lead to 'Fixation' effect (Purcell, 1996). The discontinuous and qualitative nature of system architecture, together with 'Fixation' effect may hinder designers from thinking beyond known architectures. Compared with informal methods, formal methods can be made systematic (Pahl et al. 2007), it can identify high-quality architecture more reliably and reduce the effect of fixation (Kurtoglu, 2009). Bryant et al. (2005) proposed rule-based repository definition and explored function/component allocation. They proposed to use the repository with a set of matrices that define a number of function/component allocation rules and compatibility constraints. Potential component configurations can be resulted from this concept generation process. However, whether the components are compatible is judged after concept generation, which may lead to combination explosion. Kurtoglu (2009) analysed the existing products, generated a series of generation rules, and proposed an automatic configuration flow graph generation method. The deficiency of this method is that it depends on the existing products, a new component which is not used before may not be found, the innovation of design is limited. Wyatt (2012) researched on the computational method to assist product architecture design, a formal representation of design space and existing product architecture is proposed, four kinds of network structure constraints is defined to identify the rationality of system architecture. The method is proved to be usable, applicable and useful in practice. However, the constraints and design alternatives in this method are formally and

graphically specified, they are not represented in multidisciplinary engineering platforms such as SysML, and the information between function and component is omitted in this method. Moullec (2012) proposed a product architecture generation method based on Bayesian network. The components, the feature and constraints are all represented as node in Bayesian network. If all constraints are satisfied, the final global confidence value is computed to judge the feasibility of system architecture. However, this method generates a huge number of feasible architecture, which is difficult for designers to evaluate them. This generation method did not consider the specific requirement in each design process. Fixson (2005) developed a multi-dimensional framework that enables comprehensive product architecture assessments. The framework supports assessments from function-component allocation scheme and interface characteristics. It can be used to focus advice for product architecture design, to assess advantages and limitations of operational strategies in given product architecture etc. However, this framework is focused on analysis, which is not applicable to the automatic evaluation of huge product architectures in generation process. Okudan (2009) classified and analysed concept selection methods (CSM) provided between 1980 and 2008, he emphasized that a fast and simple method which gives importance to customer requirements and allows for coupled decisions under uncertainty is needed.

3 METHOD OVERVIEW

Generally, system architecture includes logical system architecture (the function layer) and physical system architecture (the component layer), system architecting is an indispensable part of system design and represents the transformation from an abstract system function to detailed physical components. The focus of this paper is physical system architecture generation, which is corresponding to the second and third parts of system architecture defined by Ulrich (1995). As shown in Figure 1, for each function in logical architecture (which is assumed to be already determined), the following three steps are conducted:

- Find the components which can be used to realize the function and evaluate their matching degree with the corresponding function;
- Exclude some components that do not meet specific requirement quota and evaluate the satisfied components according to the specific criterion of each kind of components together with the specific requirement;
- Combine different components that can realize different functions when all functions in logical architecture are considered. The compatible information and the pros and cons of components are also considered in this combination process.

After that, all feasible physical architectures are generated. The whole method is supported by an ontology knowledge base, which contains knowledge of components, functions, functional effects, criterion, flows etc.

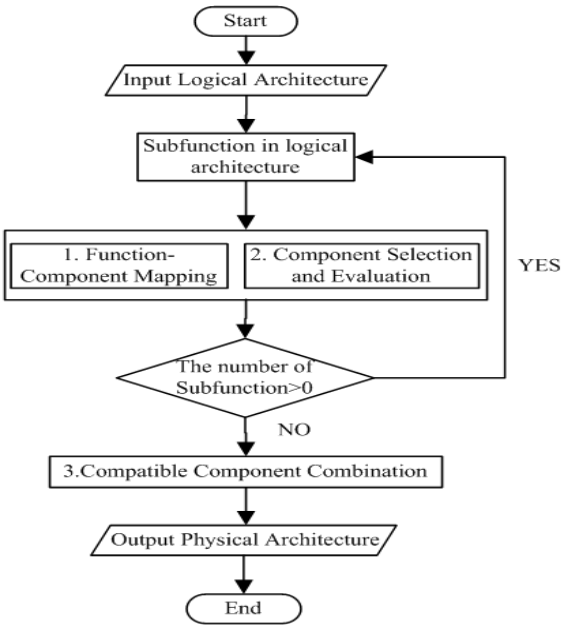

Figure 1. The flowchart of the physical system architecture generation method

4 FUNCTION-COMPONENT MAPPING

Function-component mapping is the first step in physical system architecture generation process. In order to realize function-component mapping, the model of function and component should be established consistently.

4.1 The unified model of function and component

The unified model of function and component is established based on Systems Modelling Language (SysML) (Weilkiens, 2007). There are two extension mechanisms for SysML: the heavy-weight method that defines new meta-classes and the light-weight method that creates stereotypes by extending existing constructs (Cao, 2013). Here, the latter is chosen since it is well supported by the existing SysML modelling tool. Several stereotypes and their instances are established in this study, some typical ones are shown in Figure 2. In this study, the function is modelled with four tags: 'input flow', 'output flow', 'functional effect needed' and 'realizedBy'. The flow tags are established according to functional basis (Hirtz, 2002), whose type is «FlowDefined». 'Functional effect needed' means the functional effect needed to realize the function, the value of the tag 'realizedBy' are the components which have been used to realize the function in existing products. The component model contains 'input flow', 'output flow', 'functional effect provided' and a series of criterion tags. Here, the flow tags and the tag named "functional effect provided" are built in the same way as function. Based on «ComponentDefined», several specific component stereotypes are defined, e.g. «Engine». Different kinds of components have different criteria. The common criteria such as mass and cost are modelled in the stereotype «ComponentDefined», while the specific criteria of each component are modelled in its own stereotype.

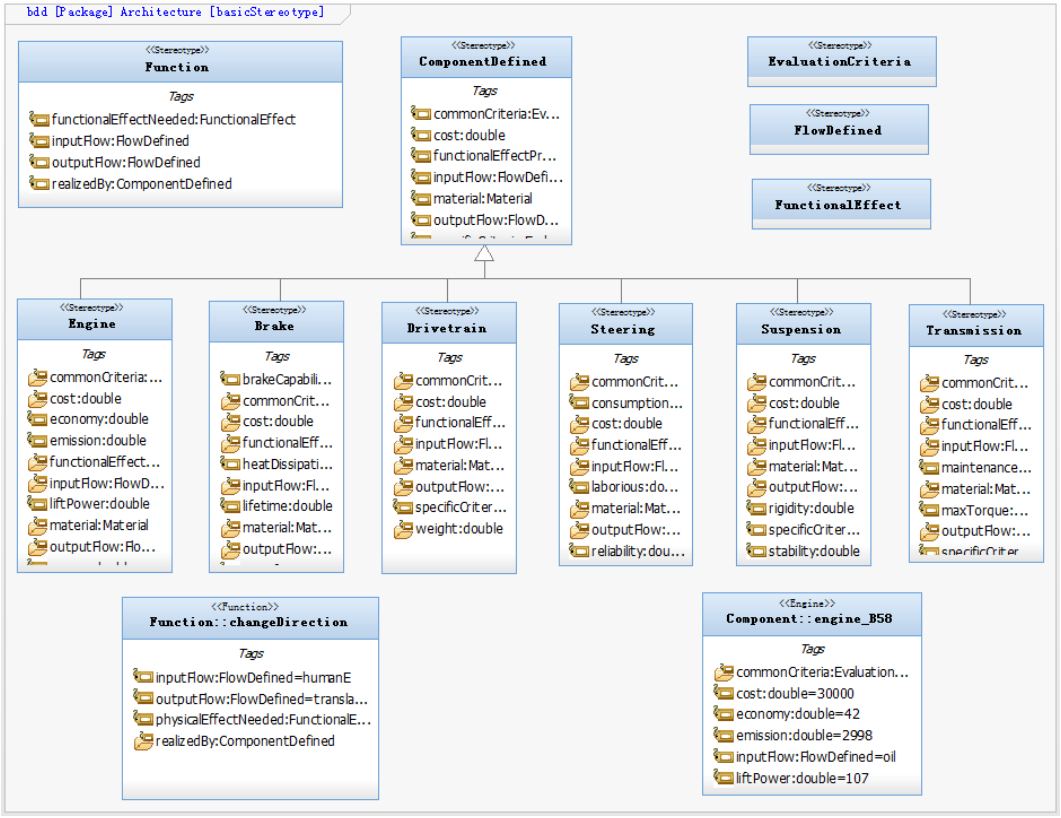

Figure 2. The unified modelling of function and component

4.2 Mapping method

Two mapping method are provided here to ensure all components which can realize a function are found. In order to support two kinds of product architectures (modular product architecture and integral product architecture) stated by Ulrich (1995), the definition of function and component in this paper is generalised, which may refer to the combination of functions and components respectively.

4.2.1 Case-based mapping

Case-based mapping is the mapping method based on empiric. The existing products are analysed and the frequently used function-component mapping lists are stored in knowledge base. In a new product design process, the logical architecture can be analysed to find whether there are some functions which have some corresponding components in knowledge base. If there are some components corresponding to a function in the mapping list, it means that these components are options to realize the function, set them as the function's 'realizedBy' tag value.

4.2.2 Rule-based mapping

The case-based mapping may limit the innovation of design, perhaps there are some components which can realize a function have never been used in previous design. Therefore, rule-based mapping is proposed according to the two classical definitions of function stated in Rodenacker (1997) and Miles (1972). There are mainly three rules based on the unified model:

- For a function in logical architecture whose input is coming from environment and whose input flows are different with output flows, if the output flow of a component is the same as or the superclass of the output flow of the function, it is deemed that the function can be realized by the component.
- For other functions in logical architecture, if a function's input flows and output flows are the same as or the subclass of a component's input flows and output flows respectively, it is deemed that the function can be realized by the component.
- For all functions in logical architecture, if the value of its 'functional effect needed' tag is matching with a component's 'functional effect provided' tag value, it is deemed that the function can be realized by the component.

In innovative design, perhaps there are some functions which have no mapping component in the knowledge base. In this situation, the above two mapping methods are invalid. Designers should discuss with each other or search online to find the components which can realize these functions, and extend the knowledge base.

4.3 The matching degree between function and component

A function may be realized by many components, different components may have different performance in realize the same function. As shown in Section 4.2.2, if a component's input flow and output flow is totally matched with a function's input flow and output flow, while another component's input flow and output flow are superclass of the same function's input flow and output flow, it is obviously that the former component is more suitable to be used to realize the function. Therefore, evaluate the matching degree between function and component from qualitative aspect is necessary. In this paper, the components which can realize a function are divided into three categories: totally match, mostly match and half match. Totally match means that the input flow and output flow of a component is the same as the input flow and output flow of a function, the matching degree is set to 1; mostly match indicates that a component's input flow or output flow is the same as its mapping function's input flow or output flow, and another one has the superclass relationship, the matching degree is set to 0.8; half match implies that both the input flow and output flow of a component are superclass of its corresponding function's input flow and output flow, the matching degree is set to 0.5. This matching degree is used as an index to evaluate the component's suitability to realize the function in the following section.

5 COMPONENT SELECTION AND EVALUATION

Using the method stated in Section 4, all components which can realize the function are found and evaluated from qualitative aspect. However, as the class of components which can realize the same function have some common criterion, e.g. the power and torque of engine, and different criterion of the same component class may have different weights according to the specific requirement in different design processes. Therefore, evaluate the suitability of each component in realizing the same function from quantitative aspect according to specific requirement is needed.

5.1 The evaluation criterion collection

In this study, the evaluation criteria are divided into two kinds: common criterion and specific criterion. The common criteria are referred to the criterion all kinds of components have, e.g. price, mass. The specific criteria are referred to the specific criterion belonging to specific component, e.g. power of

engine. For different kinds of components, collect their commonly used criterion respectively and establish them in knowledge base.

5.2 Component selection

In specific product design, there may be some hard requirements for some criterion, e.g. the power of engine should not lower that 200PS in premium car design. Therefore, the components found in Section 4 should be checked, and excluded some components which are not satisfied in specific requirement.

5.3 Component evaluation

After the exclusion process, the satisfied components for each function should be evaluated according to specific criterion and specific requirement. Different kinds of components have different kinds of criterion, there may be some relationships between these criterion. For example, power, torque, price etc. are commonly used criterion in evaluating engines, generally the higher the power, the higher the price, however, the higher the power and the lower the price is what we want. In different design processes, different criteria have different weights. For example, the power has higher weight than price in premium car design, and they may have the same weight in ordinary family car design. In this paper, Analytic Hierarchy Process (AHP) (Saaty, 1990) is used to evaluate different components with different criterion according to various requirements. Take the function 'generate power' as an example, the suitability degree of different engines in realizing the function are evaluated using AHP. As shown in Figure 3, the first layer is the function 'generate power', which is the overall goal of the problem; the criterion layer contains qualitative criteria (the matching degree stated in Section 4.3) and quantitative criterion (power, torque, emission, price etc.); different engines form the component alternatives layer.

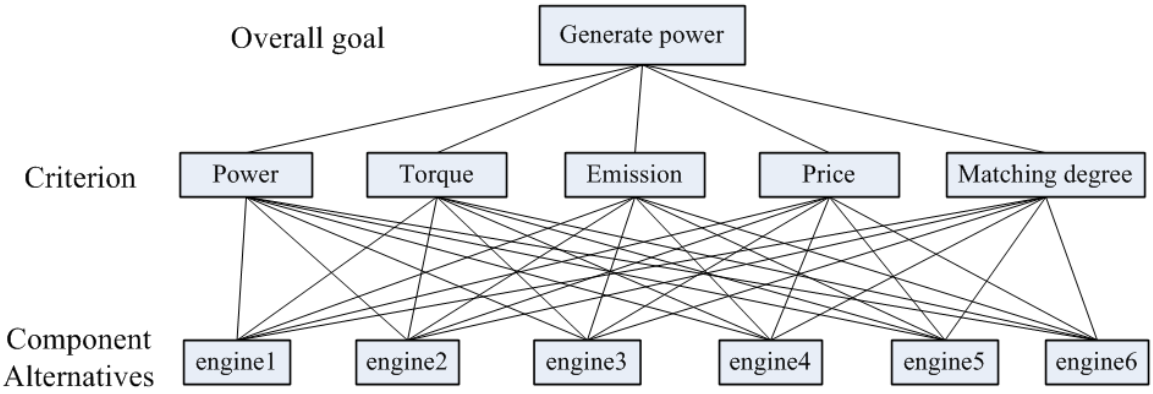

Figure 3. A hierarchy for choice of function 'generate power'

After evaluate the satisfied components for each function in logical architecture, different kinds of components should combine together to fulfil the overall function of the product. In the combination process, the compatibility between components should be checked. How to generate all feasible physical architecture efficiently and avoid the problem of combination explosion is the task of next section.

6 PHYSICAL ARCHITECTURE GENERATION

An objective function that combines criterion is constructed in Kurtoglu (2010). According to the objective function, compute the transition cost for each node in Configuration Flow Graph (CFG), find the component which has the minimum transition cost until the whole CFG is instantiated and the optimum solution is reached. However, a criterion in system design may refer to an attribute, an objective, a performance requirement, a goal or a point of view, it is not always considered as a mathematical function (Moullec, 2016). Therefore, the solution reached in Kurtoglu (2010) is the optimum one from transition cost aspect, but it maybe not the one designer want. In this paper, all feasible physical architectures are listed and sorted by weight.

6.1 Component combination method based on dynamic programming

In this paper, the physical architecture generation process is divided into several component combination steps, which is based on dynamic programming. The number of combination steps is different according to the granularity of the function in logical architecture, whose maximum value is the number of functions. The satisfied components for each function are added successively, whether the new added component is compatible with the existing components in the list is checked. If compatible, the new component is added and a new component combination is reached; if not, try other existing component list. The whole set of feasible physical architecture is reached when all functions in logical architecture are considered.

As shown in Figure 4, assume that there are three functions in logical architecture(function1, function2, function3) and there are three kinds of components(A, B, C) correspondingly. Assume that {A1, A2, A3}, {B1, B2}, {C1, C2, C3} are the satisfied components of function1, function2, function3 respectively. Assume that the suitability of A1 to realize function1 is 0.6, the suitability of A2 to realize function1 is 0.3, the suitability of A3 to realize function1 is 0.1. Similarly, the suitability of B1 and B2 to realize function2 is 0.4 and 0.6 respectively. The suitability of C1, C2, C3 to realize function3 is 0.4, 0.3, 0.3 respectively. The physical architecture generation process is divided into three steps in this case. In the first step, there are three options {A1, A2, A3}; in the second step, there are only four choices {A1, B1; A1, B2; A2, B1; A3, B2} since {A2, B2}, {A3, B1} are not compatible; similarly, in the third step, there are only six options {A1, B1, C2; A1, B2, C1; A2, B1, C2; A2, B1, C3; A3, B2, C1; A3, B2, C3}. The red dotted line in Figure 4 means that the components linked are not compatible. In this way, all feasible physical architecture and their information (suitability in this design, rough price etc.) are generated, and they can be evaluated by these information. Designers can choose one or more physical architecture from this set to do further design.

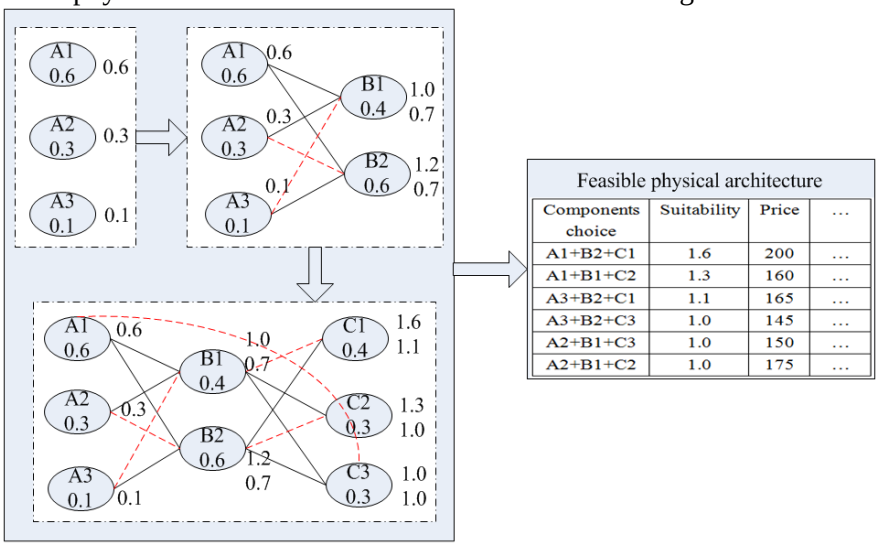

Figure 4. Component combination method based on dynamic programming

7 IMPLEMENTATION AND CASE STUDY

The whole physical architecture generation method is implemented as a plugin in IBM Rational Rhapsody. It is supported by an ontology knowledge base. The automobile design is used here to verify the effectiveness of the method. The logical architecture shown in Figure 5 is the input of this method, here the logical architecture is simplified and only the main functions are considered.

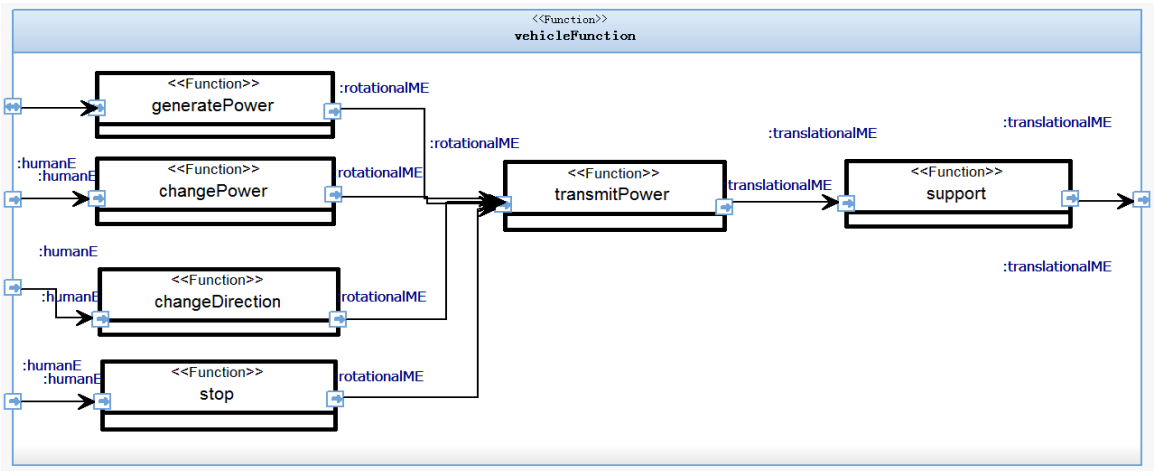

Figure 5. The logical architecture of automobile

For each function in Figure 5, find its matching components (stated in section 4), exclude some components which are not satisfied under specific requirements and evaluate the other satisfied ones (stated in section 5). The corresponding components to realize function 'generate power', the corresponding mapping ways and their matching degree with the function are shown in Figure 6. The component selection procedure of function 'generate power' is shown in Figure 7. If there are some specific requirements about this function, designers can input them in the selection table and the components which are satisfied are shown as the right part of Figure 7. The blank space means that there is no limit about the criteria. When the whole satisfied components are obtained, the component evaluation procedure is carried out as shown in Figure 8. The relative weight of each criterion is input in the table (the blank space means that the criteria is not considered in this procedure) and the suitability of each engine is shown as the right part of Figure 8.

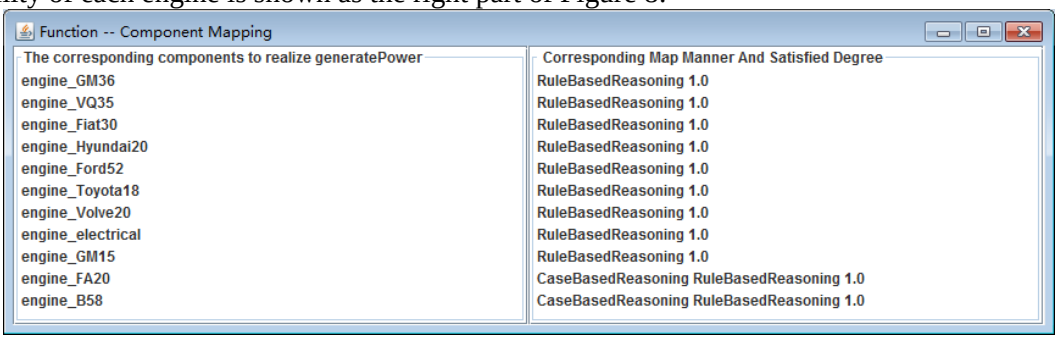

Figure 6. The components corresponding to function 'generate power'

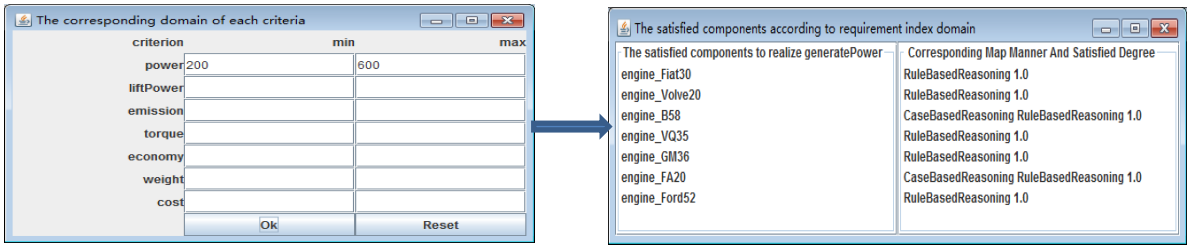

Figure 7. The remaining components after component selection

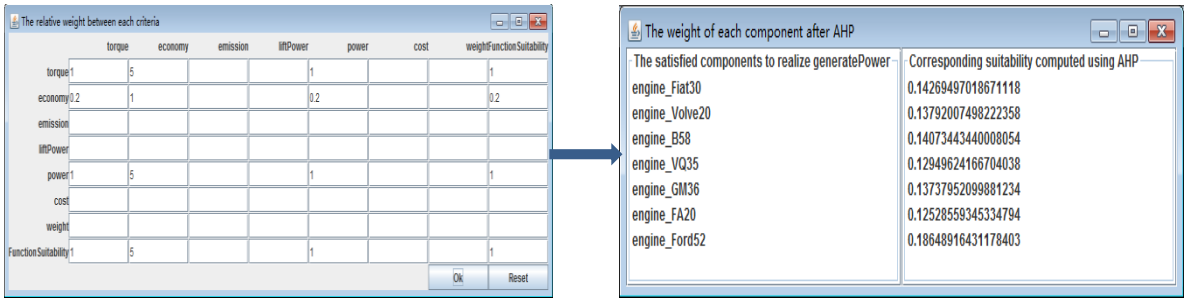

Figure 8. The suitability of each component to realize function 'generate power'

When the satisfied components and their corresponding suitability degree of each function in Figure 5 are obtained as shown in the right part of Figure 8, the component combination procedure can be executed. All feasible physical architecture together with their information, e.g. weight, price, are listed in Figure 9. Here, weight is the sum of suitability degree of each component in the list item. The higher the weight, the better the components.

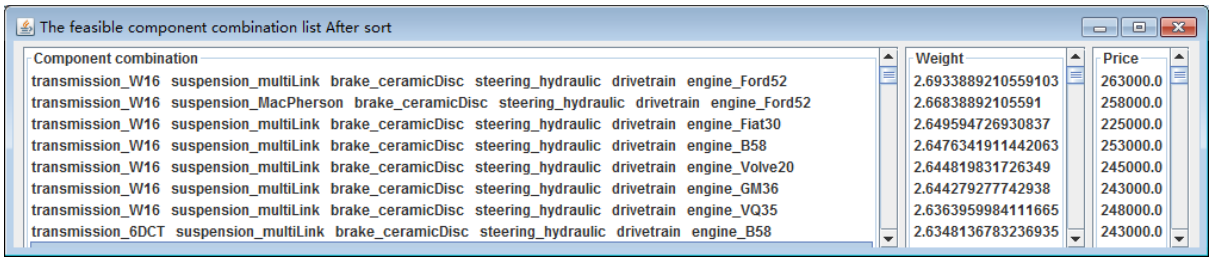

Figure 9. The feasible physical architecture list

8 CONCLUSION

System architecture is the core of system design. It reflects the results of early design decisions and is also the basis of other system-level work(system optimization, system simulation, etc.) and subsequent detail design. Although several system architecture generation method are proposed, there is still no method which support the automatic transformation from logical system architecture to physical system architecture and consider the suitability of components according to specific requirement. In this paper, an automatic physical system architecture generation method is proposed. The main contribution is as follows:

- Two kinds of mapping method between function and component are proposed. Under the premise that the library is perfect, all components corresponding to the function can be found and their matching degree with the function from qualitative aspect is considered.
- The common criterion of each kind of components are collected and built in ontology knowledge base. The satisfied components of each function are evaluated relatively with these criteria from quantitative aspect according to specific requirement.
- An efficient component combination method based on dynamic programming is proposed. The evaluation information of the physical architecture can be collected in each combination step, and the combination explosion is avoided. As all feasible physical architectures are listed and sorted by suitability weight, there is no risk that some preferable physical architecture options are lost.

The physical architecture generation method can be used hierarchically in product design. This means that the automobile can use this method to find the optimum component combination, the engine, which is a part of automobile, can use this method to find the optimum component combination either. The method is efficient and generic for any product design. However, this study is still in its infancy. The first physical architecture item listed in Figure 9 is component optimum. How to layout these components to obtain the optimum product performance will be addressed in our future research.

ACKNOWLEDGMENTS

The authors appreciate the support from the National Key technology R&D Program (No. 2016YFD0400301), the National Science Foundation of China (No. 61572427 and No. 61672247), the Open Project Program of the State Key Lab of CAD&CG (No. A1720), Zhejiang University and the Founding Program for the Cultivation of Young University Teachers of Shanghai (No. ZZGCD15084).

REFERENCES

Bryant, C. R., Mcadams, D. A., Stone, R. B., Kurtoglu, T., and Campbell, M. I. (2005). "A Computational Technique for Concept Generation". *ASME 2005 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (pp.267-276). American Society of Mechanical Engineers. https://doi.org/10.1115/detc2005-85323

Cao, Y., Liu, Y., Fan, H., andFan, B. (2013). "SysML-based uniform behavior modeling and automated mapping of design and simulation model for complex mechatronics". *Computer-Aided Design*, 45(3), 764-776. http://dx.doi.org/10.1016/j.cad.2012.05.001

Estefan, J. A. (2007). “Survey of model-based systems engineering (MBSE) methodologies”. *Incose MBSE Focus Group*, 25(8).
Fixson, S. K. (2005). “Product architecture assessment: a tool to link product, process, and supply chain design decisions”. *Journal of Operations Management*, 23(3–4), 345-369. http://dx.doi.org/10.1016/j.jom.2004.08.006
Hirtz, J., Stone, R. B., McAdams, D. A., Szykman, S., and Wood, K. L. (2002). “A functional basis for engineering design: reconciling and evolving previous efforts”. Research in engineering Design, 13(2), 65-82. https://doi.org/10.1007/s00163-001-0008-3
Kurtoglu, T., and Campbell, M. I. (2009). “Automated synthesis of electromechanical design configurations from empirical analysis of function to form mapping”. *Journal of Engineering Design*, 20(1), 83-104. https://doi.org/10.1080/09544820701546165
Kurtoglu, T., Swantner, A., and Campbell, M. I. (2010). “Automating the conceptual design process: from black-box to component selection”. *Artificial Intelligence for Engineering Design Analysis & Manufacturing*,24(1), 49-62. https://doi.org/10.1007/978-1-4020-8728-8_29
Miles, L. D. (1972). “Techniques of value analysis and engineering”. Value Analysis.
Moullec, M. L., Bouissou, M., Jankovic, M., & Bocquet, J. C. (2012). “Product architecture generation and exploration using Bayesian networks”. *In DS 70: Proceedings of DESIGN 2012, the 12th International Design Conference*, Dubrovnik, Croatia.
Moullec, M. L., and Eckert, M. J. C. (2016). “Selecting system architecture: what a single industrial experiment can tell us about the traps to avoid when choosing selection criteria”. *Artificial Intelligence for Engineering Design Analysis & Manufacturing*, 30(3), 250-262. https://doi.org/10.1017/s0890060416000238
Okudan, G. E., and Tauhid, S. (2009). “Concept selection methods – a literature review from 1980 to 2008”. *International Journal of Design Engineering*(3), 243-277. https://doi.org/10.1504/ijde.2008.023764
Osborn A.(1957) “Applied imagination”. *Scribner*, New York.
Pahl G, Beitz W, Feldhusen J, Grote K-H (2007) “Engineering design: a systematic approach”, 3rd edn. *Springer*, London. https://doi.org/10.1016/0261-3069(96)84970-3
Purcell, A. T., and Gero, J. S. (1996). “Design and other types of fixation.Design Studies”, 17(4), 363-383. https://doi.org/10.1016/s0142-694x(96)00023-3
Rodenacker, I. W. G. (1984). Methodisches konstruieren. In Methodisches Konstruieren (pp. 37-227). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-87484-0_2
Saaty, T. L.(1990). “How to make a decision: the analytic hierarchy process. European Journal of Operational Research”, 48(1), 9-26. https://doi.org/10.1287/inte.24.6.19
Ulrich K.(1995) “The role of product architecture in the manufacturing firm”. Research policy, 24(3): 419-440. https://doi.org/10.1016/0048-7333(94)00775-3
Wyatt, D. F., Wynn, D. C., Jarrett, J. P., and Clarkson, P. J. (2012). “Supporting product architecture design using computational design synthesis with network structure constraints”. *Research in Engineering Design*, 23(1), 17-52. https://doi.org/10.1007/s00163-011-0112-y
Weilkiens, T. (2007). “Systems Engineering with SysML/UML”. *Morgan Kaufmann OMG Press/Elsevier*. https://doi.org/10.1016/b978-0-12-374274-2.x0001-6