Transdisciplinary Lifecycle Analysis of Systems
R. Curran et al. (Eds.)

Generic Functional Decomposition of an Integrated Jet Engine Mechanical Sub System Using a Configurable Component Approach

Visakha RAJA[a,b,1] and Ola ISAKSSON[a,b]
[a] *GKN Aerospace Sweden AB, SE 461 81, Trollhättan, Sweden*
[b] *Chalmers University of Technology, Department of Product and Production Development, SE 412 96, Göteborg, Sweden*

Abstract. A procedure is proposed to functionally decompose an already existing integrated mechanical jet engine subsystem. An integrated sub system is a system where the same design object satisfies multiple functions: which is typically the case in aircraft engine sub systems and components. A generic decomposition method will allow implementation and use in automated design systems and will function as a means to build experiences into platforms. Using the procedure, an enhanced function-means tree (E F-M tree) consisting of functional requirements, means to satisfy the requirements and constraints was created for the integrated jet engine component. The E F-M tree is then used to generate a hierarchy of configurable components (CCs). A configurable component (CC) is a stand-alone conceptual object that contains the functional requirement, means to satisfy the requirement (or design solution) and constraints at a certain level of the E F-M tree. A specific CC hierarchy configuration results in the description of the product concerned. The usage of the CC hierarchy as design documentation as well as a template to derive other designs from is demonstrated. Finally limitations of describing product functional requirements using CC method and recommendations for further development of the method are discussed.

Keywords. Integrated design, functional decomposition, enhanced function-means tree, configurable components

1. Introduction

To be competitive in the market, companies need to prepare their products for upcoming, novel developments. For a tier one aircraft sub systems supplier like GKN Aerospace, this means to quickly integrate the sub systems into alternative system architectures introduced by their customers. One such situation is engine - sub system integration. Different engine architectures demand different designs of sub systems. Due to market pressure, the sub system designs must be defined and evaluated to a minimal cost and within a short timeframe. For delivering solutions quickly, in depth knowledge about products that the company designs and manufacture is necessary. Such knowledge can be captured in a product platform.

[1] Corresponding author; E mail: visakha.raja@gknaerospace.com

In order to create a platform, it is necessary to understand the functions and interactions of important product features. This necessitates a systematic way of documenting the design. What is required of the product, what features satisfy the requirements and what are the limitations of the features. The design documentation should be such that addition, change or deletion is possible making it a continually expanding database. Option should exist to link such a database to a CAD system that can generate various input dependant configurations. Thus such a system will result in a platform being made for the products concerned.

The first step in making such a database is to carefully identify and relate the functions the structure is intended to satisfy, the means to satisfy those functions and constraints if any. Once the identification is done, the function-means [1] and constraints should be represented in an easily understandable form. For a modular product or an assembled product, different modules or assemblies satisfy different functions. The identification and representation of function-means is thus straightforward for such products. However, for an integrated product system, where the same part satisfies a multitude of functions, identification of function-means is difficult. This paper examines how the said identification and representation can be done for an integrated architecture product system. The structure considered in this paper is known as a "cold structure".

1.1. A cold jet engine structure

The "Cold structure" used as an example in this paper is a static component in a turbo-fan engine, sufficiently complex to view as a sub-system by itself. The major function of this cold structure is to connect compressors and guide the flow between them. It is referred to as cold since the temperatures it is exposed to (~250^{0}C) is considerably less than other parts of the engine such as combustor or turbines (~1200^{0}C). In a two shaft turbo fan engine, such cold structure connects the low pressure compressor (booster) with the high pressure compressor. In a three shaft engine the cold structure connects the intermediate pressure compressor with the high pressure compressor. Depending on the engine architecture (two/three shaft, geared) and engine manufacturer OEM the cold structure may be referred to using different names (Intermediate Casing or IMC for example) but in this report a generic structure with the most general functionalities is considered for modeling. **Figure 1** shows the position of a cold structure in a two-shaft turbo-fan engine.

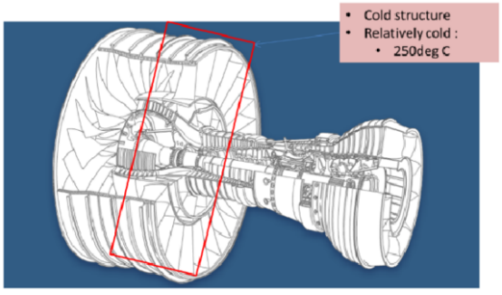

Figure 1. Position of cold structure in a two spool turbofan engine (generic figure).

2. Current practice

For mechanical component and sub-system design, engineering design activities are driven by "statements of work" documents, as agreed with the system integrator, in this case the engine OEM. Such document state design requirements and criteria, as well as how information is shared between the integrating team and the component design team. The requirements stem from a Requirements Engineering work where practices differ, yet have in common that they should be expressed to replicate function and performance, irrespective of a design solution. In practice, it is quite difficult to separate out the dependency to the choice of product solution, since requirements are being derived from a system solution with a certain component solution in mind.

From a component and sub system developers point of view, it is highly desired to 1) enable re-use of best practices and experiences, 2) to customize a product solution based on the desired behavior "profile" , i.e. whether to optimize on low weight or against manufacturing robustness as an example. Such decisions are made in any design project, and typically require trading alternative concepts against each other.

Hence, it is desired to understand a components performance and behavior in a systematic way. At present, such trades are captured in "Product Platforms" [2][3][4] which has been developed together with Chalmers over the last years.

It is also noted that the value of how to trade the performance and behavior of the targeted design solution, is an integral part of the dialogue within the design team and together with the integrator. At present the decision support for this work is limited to "structural analysis" such as using PUGH matrices, QFD analysis or FMECA work [5] where the function is separated out from the design solution.

There is – however – not an established support for how to systematically express a component performance and behavior from a functional perspective. F-M studies are being made, yet not a part of a standard work, but rather as an analysis tool for training or post design engineering work.

2.1. Literature review

There exist a variety of approaches to functional description and decomposition. Aurisicchio et al. [6] classifies methods of representing product functional breakdown into form-dependant methods and form-independent methods. Form dependant methods are those that depend on the shape of the product (components in the assembly). Form-independent methods do not depend on the shape of the component and proposes generic solutions. In order to generate the functions, methods such as functional analysis system technique or subtract and operate procedure can be used. For an already existing product, form dependant methods are more intuitive to use. In order to represent the functional breakdown, Aurisicchio et al. [6] proposes a functional analysis diagram (FAD). The FAD aims to combine a CAD model and functional descriptions of different parts in the model using day-to-day language.

Levandowski et al. [7] did an early effort to represent an integrated component using configurable component approach. However the study was aimed at exhibiting the capabilities of the configurable component method as a means of product platform creation and did not focus on how to extract functions and means from an integrated component like a cold structure and represent them as a configurable component. In contrast to Levandowski et al. [7], we focused on functional decomposition itself and

subsequent representation of the identified function-means-constrains using configurable component approach.

3. The E F-M tree and Configurable Component (CC) concept

3.1. The enhanced function means tree

A function-means tree (F-M tree) is a graphical representation of a need and the solution that satisfies the need. For example the need to cut vegetables satisfied by a chef's knife. Andersson et al. [8] propose the enhanced function means tree as function-means tree enhanced to include constraints associated with the solutions as well. This enables visualizing a complete picture of the product functional breakdown. The elements of a function means tree according to Andersson et al. [8], considered in this paper are:

3.1.1. Functional requirement

Functional requirements are what a product or element of a product does in order to contribute to a certain purpose by creating an internal or external effect [8].

3.1.2. Means

Means are physical or abstract entities chosen during design process to fulfill the functional requirements. Means are referred to as Design Parameters (DP) in [8] though in this paper, following [7], they are referred to as Design solutions (DS).

3.1.3. Constraints

Constraints are non functional requirements that do not have specific solution rather bound and add value to solution space such as weight, cost, reliability, safety and ergonomics.

Therefore it is possible to represent product information as an E F-M tree hierarchy. **Figure 2** show an E F-M tree for a kitchen knife.

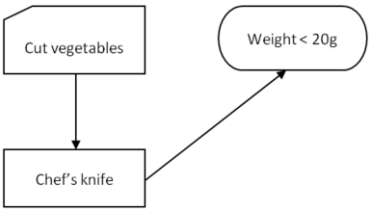

Figure 2. E F-M tree for at chef's knife. FR, DS and C representations adapted from [7].

3.2. The configurable component (CC) method

The CC method is described [9]. The method was developed as a concept model for developing computer based product platforms. In its simplest form, when functional requirements, means and constraints in the E F-M tree at a certain level are taken

together, the resulting construct is a CC. In the CC model, means (as defined in section 3.1.2) is termed as design solution (DS). Therefore in the most basic form a CC contains Functional Requirements (FR), Design Solutions (DS) and Constraints (C). The E F-M tree hierarchy thus turns into a number of interconnected CCs.

A CC is a standalone object that can call other CC objects. A particular CC can be made such that it satisfies the constraints in a certain manner. Eg. choose the 3rd dimensional value from an available list of 5. Thus a CC has been configured according to a requirement and hence the name 'configurable' component. It can also be that a design solution is assigned a certain dimension. When a CC is configured (assigned values or made selection or such), it is called an instantiation. A product will then be a collection of all configured (or instantiated) CCs.

Multiple CCs can communicate to each other and can have interfaces. CC Interfaces are not considered in this paper. Details about CC interfaces can be found in [9].

According to definition in [9], inside a certain CC, an FR is satisfied by one and only one DS. Multiple FRs and DSs inside a single CC will make the configuration of the component difficult and decisions as to which solution needs to be applied difficult. In other words, the CC does not exist as a stand-alone entity and configuration is difficult.

When a DS satisfies a FR, the relation (indicated by arrows) between them is termed '*isb* - is solved by'. Similarly the relation between a DS and C is termed '*icb* - is constrained by'. When a DS refers to secondary FRs, the relation is termed '*rf* – requires function'. When a CC refers to other CCs, the relation is termed '*icu* - is composed using'.The terms, *isb*, *icb*, *rf* and *icu* can be noted in **Figure 3**. Other relational terms also exist in CC definitions but are not considered in this paper.

In contrast to the definition of means in section 3.1.2, the DS in a configurable component are generic in nature. Multiple DS can be solved by a component (CO) [7]. This is further exemplified in section 4.

Detailed application of the configurable component method can be found in [3] and [4].

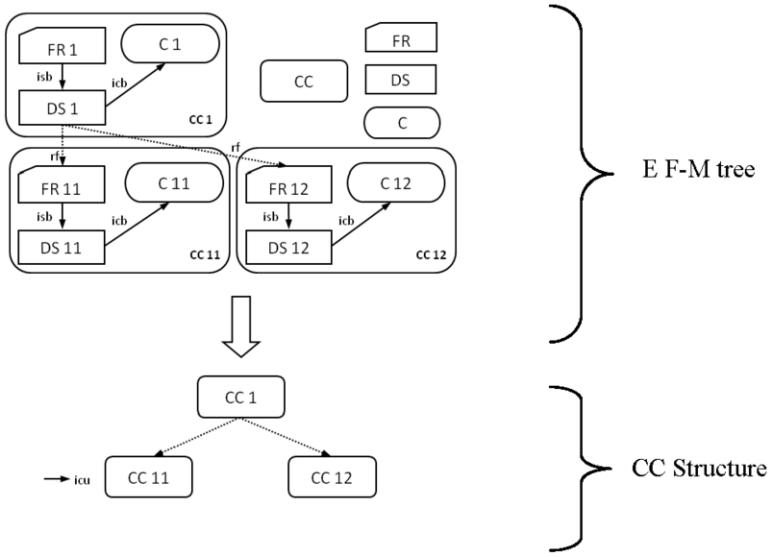

Figure 3. Creation of CCs from E F-M tree. Adapted from [9].

4. Application of configurable component method to an integrated system

For the configurable component method to be applied to an already existing product, a decomposition of the functions of the product must be performed. The method used for functional decomposition is according to that suggested in Ullman [10]. For a certain product it involves disassembling each constituent component and listing its functions. For an integrated component, the same component satisfies a number of functions. Different sections of the component can be thought of as satisfying different functions. Similar to noting functions for each constituent component, functions can be noted for each section. **Figure 4** shows the cold structure designed by GKN Aerospace in connection with the EU project Environmentally Friendly Aero-Engine, VITAL [11]. The most important sections, their functions, constraints and interfaces are noted down in a table. Each row of the table will form the basis for a configurable component. The functions of the section are the functional requirements, the section itself is the design solution and constraints are constraints. The collection of each row is the listing of function-means-constraints for the integrated mechanical component in its entirety. Not all functions and design solutions concerning a cold structure are shown in this paper. Only a selected number of functions are shown.

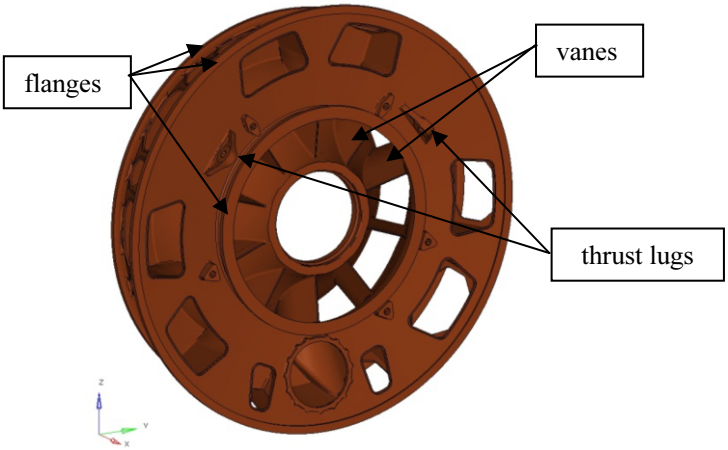

Figure 4. Cold structure designed for the VITAL program [11]. Figure does not correspond to part marked in **Figure 1**.

The method used can be summarized into the following steps.

1. Prepare an exhaustive list of functions that the integrated component is required to satisfy
2. Separate the component into identifiable sections
3. Identify constraints associated with each section
4. Create a table in which each identified product section is assigned functions that it satisfies and associated constraints
5. Create an E F-M tree for each section connecting functional requirement, design solutions and constraints
6. Prepare configurable components from the E F-M tree table

While performing step-4, identification of further functional requirements might result. These additional functional requirements should be added to the list created in step-1.

Table 1. Functions, solutions and constraints.

No	Sections (Design Solutions)	Functions that the sections satisfy (Functional Requirements)	Constraints
1	Thrust lugs	transfer thrust loads to aircraft	Length of thrust lug arm diameter of thrust lug thickness of thrust lug angle of inclination of thrust lug distance between the arms of the thrust lug
2	Flanges	act as component interfaces	Inner diameter Outer diameter flange thickness
3	Vanes	connect flow annulus walls transfer rotor loads to engine outer frame induce changes in flow properties	vane thickness vane forming methods (cast/sheet metal) vane height vane length (actual chord, axial chord)

The configurable components that are derived from the functions-means-constraints for thrust lug is (first row of Table 1) shown in **Figure 5**.

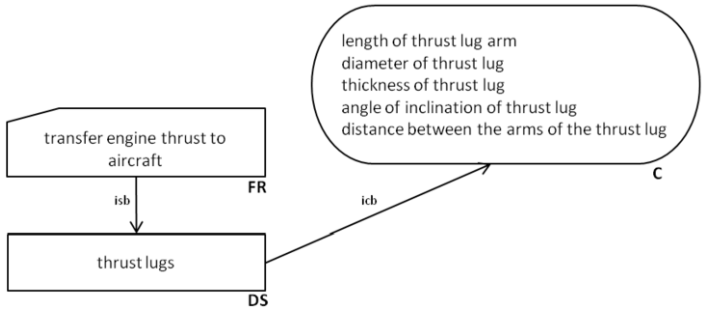

Figure 5. CC corresponding to thrust lug section as design solution.

There are a number of sections that satisfy function 'to act as component interfaces'. Thus a CC is made for flanges with applicable constraints. The instantiation of the flange CC results in different flanges for the components.

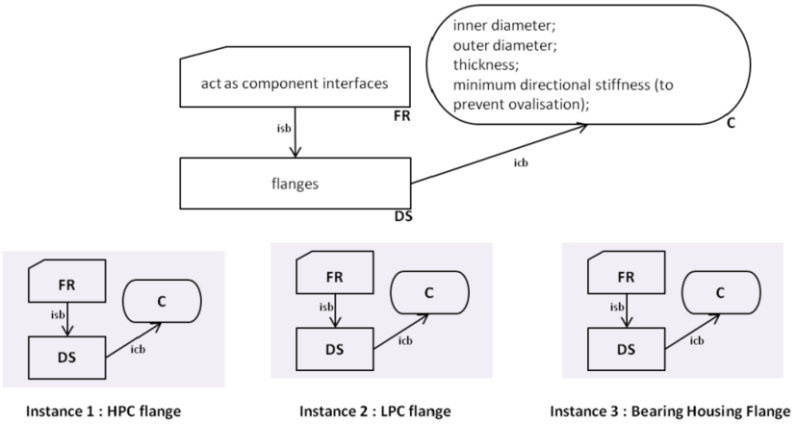

Figure 6. Flange CC and its instantiation.

It can be noted from Table 1 that the same sections satisfy multiple functions. As stated in section 3.2, in a CC, a functional requirement is satisfied by one and only one design solution. Therefore for a certain functional requirement, the closest concept name that indicates a solution is noted with its constraints which in turn are satisfied by the section (section becomes a CO as stated in section 3.2) mentioned in the table. Thus, the vane section has three functions to satisfy which are met by three design solutions expressed in general terms which in turn are provided by the section vane (similar to what has been done for flanges, instantiation is also possible for vanes or thrust lugs though they are not shown here).

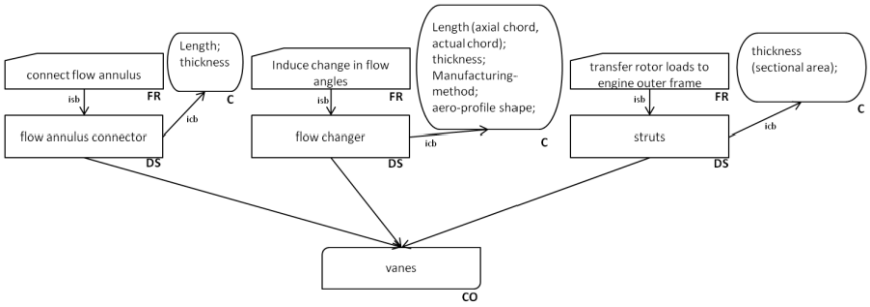

Figure 7. CC for Vane.

The collection of all CCs forms the integrated component. Following sub-sections detail two of the application cases for such a functional decomposition.

4.1. Application Case-1: Design documentation

Along with the table that lists the functional requirements and design solutions, the collection of CCs form a basis for documenting the design. It is possible to get an overview of the component design- which functions required the inclusion of what sections and the constraints associated with the sections. Once the E F-M tree/CC structure is made it is possible to visually identify the contribution of different product sections towards satisfying different functional requirements.

4.2. Application Case-2: generation of new designs

With reference for **Figure 7**, the section 'vane' satisfies three different functional requirements. If the vanes are manufactured using sheet metal forming method, they may not have enough strength to carry rotor loads towards engine outer frame. Therefore the load transfer function needs to be satisfied using another section. As an example, strut rods can satisfy the function of load transfer and they can be located inside the vanes. If such is the case, the CC listing can be changed as shown in **Figure 8**.

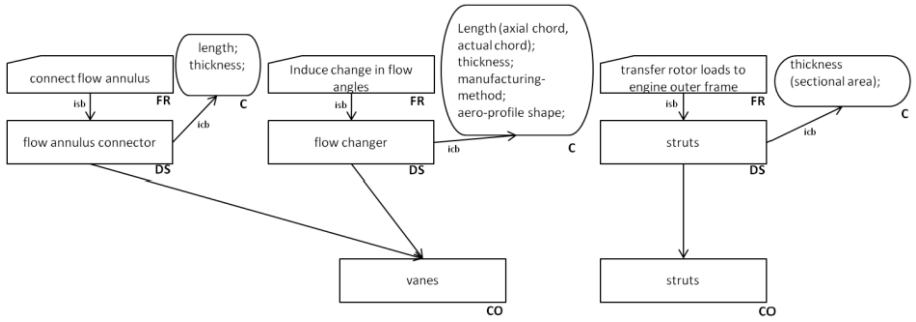

Figure 8. Separation of function from a section and formation of new design.

Only two CCs are satisfied by the vane section, the other CC is satisfied by the struts sections. Thus a function is separated from a section and assigned to a new section in order to create a new solution. In this case the functional breakdown structure forms the basis of a product platform.

5. Concluding discussion and further work

A procedure has been proposed to represent the function-means-constraints of an integrated jet engine component. The function-means-constraints representation was then used to create configurable components, which are stand-alone objects that contain a certain functional requirement, design solution that satisfies the requirement and constraints associated with the design solution. The method was applied onto a representative complex component, a cold structure. The generated CC structure was demonstrated as design documentation as well as a template to generate additional designs.

It is possible to perform a functional decomposition using the procedure though hard to perform it without contextual knowledge. The decomposition was not possible to uniquely derive, rather there exist alternate ways of decomposition still following the suggested procedure. From a system viewpoint this is a limitation. Also, some lack of clarity as to up to which level the decomposition should be done exist. These limitations are discussed in detail below.

5.1. Way of identifying sections

Separation of the product into sections does not follow any uniquely defined rule. This was done according to already existing knowledge about the system. It may be possible

to decompose the product into sections in more than one way. Consequently different CCs get generated. It is difficult to discern if CCs generated following a certain section identification scheme is correct or not. Objective criteria must exist to evaluate which way of section identification is most appropriate. This can be related to self sufficiency of resulting CC in that all relevant constraints are identified and quantified.

5.2. Granularity for CCs

By granularity, the number of levels at which configurable components exist are meant [9]. In this paper, only one level of configurable components (CCs) was identified though CCs can work in multiple levels. The single level of CC comes from the single level of the E F-M tree. A guideline is desired in terms of number of levels of CC structure for efficient operation. In general, it can be stated that CCs should exist at that level at which the design solutions (DSs) and constraints (Cs) are sufficiently simple, resulting CCs are self contained and easily re-usable.

The CC method is a powerful way of representing the function means of a structure. It captures the function, means and constraints in an effective and re-usable manner. If made sufficiently generic (a suitable level of granularity), it is possible to generate solutions automatically corresponding to requirements arising from higher levels. As an example, architecture for a sub-system can then be automatically derived from a CC model of an engine.

Acknowledgement

This work has financially been supported by NFFP, the national aeronautical research programme, jointly funded by the Swedish Armed Forces, Swedish Defense Materiel Administration (FMV) and Swedish Governmental Agency for Innovation Systems (VINNOVA).

References

[1] M.M. Andreasen, *Machine Design Methods Based on a Systematic Approach—Contribution to a Design Theory*, Lund University, Sweden, 1980.
[2] A. Claesson, *A Configurable Component Framework Supporting Platform-Based Product Development*, Chalmers University of Technology, Göteborg, 2006.
[3] M.T. Michaelis, *Co-Development of Products and Manufacturing Systems Using Integrated Platform Models*, Chalmers University of Technology, Göteborg, 2013.
[4] C.E. Levandowski, *Platform Lifecycle Support Using Set-Based Concurrent Engineering*, Chalmers University of Technology, 2014.
[5] K.T. Ulrich, and S.D. Eppinger, Product Design and Development, McGraw-Hill, New York, 1995.
[6] M. Aurisicchio, R. Bracewell, and G. Armstrong, The Function Analysis Diagram: Intended Benefits and Coexistence with Other Functional Models, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 27, no. 03 (2013): 249-57.
[7] C. Levandowski, M.T. Michaelis, and H. Johannesson, Set-Based Development Using an Integrated Product and Manufacturing System Platform, *Concurrent Engineering* 22, no. 3 (2014): 234-52.
[8] J. Malmqvist, Improved Function-Means Trees by Inclusion of Design History Information, *Journal of Engineering Design* 8, no. 2 (1997): 107-17.
[9] H. Johannesson and A. Claesson, Systematic Product Platform Design: A Combined Function-Means and Parametric Modeling Approach, *Journal of Engineering Design* 16, no. 1 (2005): 25-43.
[10] D.G. Ullman, *The Mechanical Design Process*, McGraw-Hill, Boston, 2010.
[11] Transport - Research & Innovation , European Commission. "VITAL Environmentally Friendly Aero-Engine. Last update 20/02/2012. http://ec.europa.eu/research/transport/projects/items/vital_en.htm