Describing and evaluating functionally integrated and manufacturing restricted product architectures

Research in Engineering Design (2018) 29:367–391
https://doi.org/10.1007/s00163-018-0286-7

ORIGINAL PAPER

Describing and evaluating functionally integrated and manufacturing restricted product architectures

Visakha Raja[1,2] · **Hans Johannesson**[2] · **Ola Isaksson**[2]

Received: 14 June 2016 / Revised: 14 November 2017 / Accepted: 21 March 2018 / Published online: 5 April 2018

Abstract
Within manufacturing companies, the architectural description of how a product is built is typically well defined while the architecture of the product from a functional view describing how the functional requirements are met in the product is often less articulated. For products composed of many components (modular products) such descriptions are clear, whereas few representation schemes are available that treat highly functionally integrated components, where all the functions are satisfied by one integral, homogenous physical structure. In this paper, an approach to describe the architecture for integrated components in the aerospace industry is described. Different regions of the component, termed as sections, are assumed to satisfy the functions required of the structure which are often manufactured by joining (welding) different segments. By assigning sectional and functional information to different manufactured segments of the structure, graphs are created that link the functional requirements and sections. Two different methods, one based on set compositions and other on creating an enhanced function-means (EF-M) tree are used to link the functions to the sections of the component, resulting in different graphs for different types of manufacturing splits for the same component. Comparison of the methods is then carried out using properties of the graphs produced. The method that utilises set compositions performs well for entire component descriptions while the method that uses an EF-M tree to create a graph describes sections (regions) of the component well so that functional relationships can be better described (integration of already existing design knowledge). The product descriptions created can help designers to identify how alternative manufacturing splits impact the functionally defined product architecture which in turn enables both improved manufacturing and improved design decisions.

Keywords Product architecture · Integrated product architecture · Graph theory · Enhanced function-means (EF-M) tree

1 Introduction

The success of products is determined by how well they satisfy the needs and expectations of the stakeholders, expressed generally in terms of functional requirements on the products' behaviour and performance. There is a trade-off between satisfying customers' expectations on functionality and performance with competitive, cost efficient development and production. Depending on the specific requirements on the products, industries adopt different strategies to maintain their competitive advantage. For example, in context of the aerospace industry, the next generation products need to provide seemingly ever increased performance and extended functionalities to satisfy both certification conditions and operative excellence. The refinement and optimisation of aero engines and their constituent parts drive designs towards a higher degree of integration, meaning that the fragmentation of design may need to be minimised (Ghisu et al. 2011).

The integrated nature of design necessitates tighter design iterations among the component designers, the system integrators and the final customers to deal with evolving and refined functional requirements. To achieve this, aerospace manufacturers adopt strategies such as platform-based developments (Suh et al. 2007), set-based concurrent engineering (Sobek et al. 1999), design automation and knowledge-based engineering (Rocca 2012). The effectiveness of such strategies relies on a clear understanding of the functions that the product is expected to satisfy and what features of the

✉ Visakha Raja
visakha@chalmers.se

[1] GKN Aerospace Sweden AB, 461 81 Trollhättan, Sweden

[2] Department of Industrial and Materials Science, Chalmers University of Technology, 41296 Gothenburg, Sweden

product satisfy the functions. In other words, the physical organisations of parts in a component to satisfy the functions or more specifically, the architecture of the product (Ulrich 1995) must be clearly defined. Ulrich and Eppinger (1995) define two types of architectures, modular and integrated. A product is modular when one particular function is satisfied by one particular component or a group of components. When all functions required of the product are satisfied by one or few components, the architecture is integrated.

The concept of modular architecture designs, where parts can be combined, re-placed and re-used, is well known to be cost effective (Holtta-Otto and de Weck 2007). Not only parts but also technology, resources and even knowledge can be defined such that they are reusable across a number of products. This is particularly attractive in the aerospace domain where solutions/designs that have been proven in flight conditions reduces risk and simplifies certification work. However, many of the aerospace component and sub-systems belong to the integrated product architecture category. Multiple functions are satisfied by one single component. The component, when used by a system integrator is often designated by one part number even though at the manufacturing firm, the component might be fabricated from a number of segments which have their own part numbers. Also, the segments might be manufactured at different geographic locations using different manufacturing technologies.

The architecture of a product can be represented using design structure matrices (DSMs) (Eppinger and Browning 2012) or node link diagrams (Keller et al. 2006). Both represent the interaction among parts in a product. A DSM represents part interactions in a matrix while node link diagrams represent the interactions as a graph with nodes as parts of the product and links (edges) connecting different parts. Keller et al. (2006) also suggest using the graphical representation for products with less number of parts and DSM for several parts. When it comes to identifying how functions are satisfied, modular products fare better because for ideally modularised products, it is straight forward to link functions to the product's constituent parts. Also, the product's DSM (or node link diagram) will be composed of many rows (nodes in case of a node link diagram) and columns. For an integrated product, the same structure satisfies the functions required of it thereby making it difficult to associate functions to constituent parts as there is only one part which is the product itself. One way of identifying and representing how functions are satisfied by the product is to identify sections (Raja and Isaksson 2015) in the product. Sections are regions or areas in the integrated product which can be thought of as satisfying a limited number of functions. This makes it possible to create a function to section mapping that can be represented as a graph (a node link diagram) which forms the basis of architecture evaluations.

In this paper, we investigate ways to represent highly integrated products in a systematic way to create a better understanding of the products in terms of how the products satisfy their expected functions. The study further seeks to incorporate different manufacturing options available for the product while creating the representation such that differences in manufacturing options results in differences in representation too. To do this, we concentrate on the architecture characterisation of aero-engine structural components which are very integrated products in that a single component (as designed and produced) satisfies multiple functions. How the component will be manufactured is typically decided by the component developer. The decision will involve if the component should be cast as a whole or if a number of cast segments should be welded together [fabrication-driven manufacturing (Whitney 1993)]. It is noted here that when we use the term manufacturing options, we mean ways of splitting an integrated component for fabrication. How to split the component has a fundamental impact on several aspects during manufacturing and production, but in practice the decision on how to split the component is also important for its functional design. Therefore, manufacturing options and manufacturing splits are used synonymously in this paper. The manufacturing decisions affect the functional requirements by having the sections that satisfy the functions to be located in different segments. We describe a way to link the functional requirements with different sections of the component by taking into account various manufacturing options (cast or welding different segments together) such that the functions to section relations result in a graph. Thus, the work describes efforts to include more nodes in a graph so that the architecture of an integrated component is better described. In essence, modularity is attributed to an integrated architecture product to better describe how functions are satisfied by different sections of the component.

With this effort, we highlight two complementary views to look at a product, an engineering design view and a manufacturing view. In engineering design the emphasis is on what the product will do whilst manufacturing focus on how the product is realised. From a manufacturing point of view, the product has a manufacturing bill of materials that represents all sub-components (or segments) that constitute a product. Prior to manufacturing, the engineering bill of materials is used by the development team to represent the functions and solutions. The higher the integration and optimisation of the products, the higher the influence of manufacturing bill of materials over engineering bill of materials. For decision making during early phase design, these two views of the product are combined into one single representation.

The load carrying, non-rotating (static) structural components of an aero engine are good examples where both functional and manufacturing integration are common.

Fig. 1 Example of a complex jet engine component embodying many functions and can be fully casted as one single piece (GKN Aerospace 2014)

Such components are typically found at the intersection of two modules of the engine, for example, between the low pressure compressor module and high pressure compressor module. Figure 1 shows such a static component, generically termed a turbine rear structure, located just after the low pressure turbine.

Even though the components satisfy a large number of functions, they can eventually be physically produced as a single-piece casting, or alternatively fabricated (welded) from a number of segments. From a design point of view, engineers need to ensure that a range of functional requirements are met. During early design phases, where alternative arrangements are studied with the objective to meet the functional requirements, the geometrical definition emerges, while the manufacturing split (for fabricated designs), as being considered to be less dependent on the functional requirements, are not included. As an example, a circumferential ring, covering the gas path, represent the geometry so that structural and aero-thermal analyses can be made, whereas the fact that the ring can be split into sections and can be welded together while manufacturing is not included in early design representations of the product. How the product is split so that it can be manufactured is important information as it affects the load paths and flow paths of the product. This paper takes into account how the product is split for manufacturing and how the splits affect the functional distribution and thereby architecture of the product. Two representations are created for the product and the representations are compared using already established measures, offering insight into the architecture of the integrated product.

2 Related work

2.1 Product architecture

Theoretical discussions and definition of product architecture took steam after the work by Ulrich (1995) which was later expanded by Ulrich and Eppinger (1995). Ulrich and Eppinger (1995) define product architecture in terms of functional elements and physical elements. Functional elements are the individual operations and transformations that contribute to the overall performance of a product. For example, for a static aero engine structure, located between the low pressure (LP) and high pressure (HP) compressors of a two-shaft engine, one of the functional elements will be to 'transfer flow between LP compressor and HP compressor. Physical elements of a product are the parts, components and sub-assemblies that ultimately implement the product's functions. The physical elements are organised into several major building blocks called 'chunks'. Chunks are formed by the components of the product. Product architecture then, according to Ulrich and Eppinger, is the scheme with which the functional elements are arranged into physical chunks and by which the chunks interact. It is possible to define two types of architectures, modular and integral. In general, when one or a few of the functional elements are satisfied by a single chunk, the architecture is modular. In case of an integrated architecture, multiple chunks act together to satisfy one or a few of the functional elements. An integrated architecture product is, therefore, a case where a very high degree of 'function sharing' (Ulrich and Seering 1990) occurs.

Defining the architecture of a product is important as it influences many decisions downstream of the product development cycle such as selection of suppliers, manufacturing operations and delivery and service plans. For instance, Yassine and Wissmann (2007) hypothesise that a company's ability to efficiently and effectively manage product portfolio is directly dependent on the product architecture. They attribute it partly due to the flexibility offered by modular architecture so that increased product variety is on offer. Besides, the product architecture plays a key role in how design organisations are structured as described by Sosa et al. (2004). The work goes on to propose a method to compare different architectures for the same product so that managers are aware of the changes in interaction among different departments due to the changes in the architecture.

Having its origins in graph theory (Christopher 1964), representation of product architecture is often carried out using a design structure matrix (DSM) (Eppinger and Browning 2012). Individual 'chunks' are represented as the rows and columns of a symmetric matrix and the

interactions marked. This is similar to an undirected graph with the nodes of the graph being different 'chunks' and the edges, indicating the interaction among the chunks. The adjacency matrix of the graph that captures the nodal interactions (links or edges connecting the nodes) is the same as the design structure matrix (DSM). Once the architecture is formed in a DSM, analyses could be performed on the matrix. This could, for example, be a clustering analysis where members of the DSM are grouped according to a common objective, such as suppliers for the components. For a modular architecture product, the clustering analysis is useful as it notes the interaction among several components in the product. However, an integrated architecture product will have only a few rows (or even one single row) in the DSM if the as-built architecture is used, making such analyses difficult. For example, consider that the product that needs to be analysed is the turbine rear structure shown in Fig. 1 which is a monolithic structure. The DSM for this structure will include only one part in it which is the structure itself. Since a DSM is the underlying structure of a graph, if more nodes are included in the graph, the integrated component could be subject to analysis. Addition of more nodes is possible if functional and sectional information is included in the graph. Our method of architecture representation involves adding both functional and sectional (solution) information in a graph.

A set of guidelines to represent the product architecture is proposed by Wie et al. (2003). The work starts from the premise that 'architecture representation offers designers benefits if it presents a well-defined domain in which designs can be documented, observed and manipulated'. Consequently, they develop a vocabulary for describing the architecture. Then, a number of diagrams are developed to represent the product architecture during each stages of design. The work considered in this paper also sets out to describe products of the same kind, using a common vocabulary making use of different manufacturing options. We do not consider all stages of product development; rather our work is an exercise to document previously finalised designs. Wyatt et al. (2011) put forward automatic synthesis and evaluation of product architectures. The synthesis is based on existing knowledge as well creation of a 'schema' or formal declarative descriptions for the 'design space' for architecture description. Thus both Wie et al. (2003) and Wyatt et al. (2011) bases their work on the definition of a vocabulary that can be used to describe the architecture.

Experienced design engineers have their own vocabulary (tacit understandings) to describe the products during development. For example, some of the structures might include a circumferentially hollow region to have increased stiffness. This region is termed as a "torsion box" although this is not immediately identifiable or visible for those unfamiliar with the structure. This is an example of how engineering designers use their own vocabulary to articulate the functional structure of a product. The definition of functional requirements in this paper is based on the authors' own knowledge of working with the structures and to a lesser extent, based on semi-structured interviews with lead engineers, responsible for different products. From experience, usage of already existing terminology has resulted in quicker response during the discussions which corroborates to the view that vocabulary for describing structures is better done in accordance with existing practices.

Theoretical concepts, even though consistently defined, are often not readily applicable in practice largely because experienced design engineers were not part of creating them. Practitioners rarely find the time or interest to develop a rigorous framework to describe their products in functional terms while scientists might have limited knowledge of the applied context. Our approach here is to bridge practice and research as close as possible. It is in our advantage that the research is carried out in the industry and the components are already in service.

2.2 Product representations using graphs

The approach to characterise the architecture in this paper is based on graph theory. Several researchers have utilised graphs to represent product models towards varying ends at different stages of design although not many methods are discussed to represent the architecture of an integrated product. Lyu and Saitou (2003) propose partition of a structure by making use of graphs so that the optimum split-up for a certain assembly stiffness can be found out. A structure such as the side door frame of a passenger car is partitioned into a number of segments. An optimum number of segments are then found out for a certain load condition using FE analyses and optimisation using genetic algorithm. Compared to the method discussed in this work, we associate the partitioning with functions and features for aero engine structures. The design for the structure is not only influenced by the loads but also by considerations such as manufacturability and availability of suppliers. The partitioning in case of Lyu and Saitou (2003) is random while the partitioning considered in this work is determined by experience and existing practices at the company. A work by Iwankowicz (2016) concentrates on assembly sequence planning for ship structures using directed graphs. The planning is later optimised using genetic algorithms. According to Iwankowicz (2016), in the ship-building industry, welded assembly units are treated as one homogenous structure at later stages of development. Structures that are considered in our work are also homogenous in the same sense though they are not physically as big as a ship's hull. Besides, when the part is delivered to the customer, customers note them with only one part number although with the manufacturing firm, the

structure is composed of several segments. Each segment might be produced using a different method (cast, forged, sheet-metal) and potentially of different materials. Jenab and Liu (2009) proposes a model for manufacturing complexity based on graph theory. The method measures the relative complexity of already existing products. The example of a jet engine is discussed in the paper. A complexity graph is made after analysing data such as processing times for different products which makes it possible to visually compare the complexity or similarity for the products. The work is similar to that discussed in this paper in the sense that already existing components are analysed.

Use of the graph theoretical approach is not only limited to physically realised components but also to CAD models. Joshi and Chang (1988) proposes an un-directed graph-based algorithm for recognising machined features in a CAD model. To assist in the search for machined features, the 3D CAD solid model is represented as an un-directed graph based on its boundary representation. You and Tsai (2009) use the same approach to identify common local features among several CAD models. Their work is based on identifying common sub-graphs among all the parts considered after representing the parts as graph based on boundary representations. Mathieson et al. (2013) created a model for predicting the assembly time using connective complexity matrices. The connective complexity matrix is a representation of the product architecture based on the type of connections it uses such as bolted connections or surface contacts. A bipartite graph is made that associates each type of connection with the parts in the assembly. This graph later forms the basis for the calculations. This is very similar to our approach in that the graph between functions and sections that we propose (described in Sect. 3) is also bi-partite. Bin et al. (2002) suggest a web-based approach for creation of product platforms that makes use of graphs which represent structural configuration of a product.

Once the product architecture description based on graph theory is available, calculations can be performed on the graph. For example, Luo (2015) proposes a method to evaluate the 'evolvability' of a product based in its architecture. The 'evolvability' measure is based on the cyclic degree (components in the product that affect each other cyclically) and interaction density (number of components that a certain component in the product affects). The work also notes that the method could be used in selection of architecture for products such as cell phones that need to evolve fast. In case of aerospace components, the 'evolvability' will not be high because the product architecture is very integrated in nature which makes the interaction density very high and component cycles low. Sosa et al. (2007) define measures for modularity of a component, based on the network structure available for the product. The measures are applied to an aero engine and the likelihood of re-design of components in the engine is predicted based on the modularity measures proposed.

The underlying idea of all works discussed so far is to represent the product under consideration as a graph and apply existing mathematical relationships pertaining to graphs so that relevant information about the product can be retrieved. Our approach is also based on representing the structure or part of the structure, along with the functions they satisfy, using un-directed graphs.

2.3 Platform-based product descriptions

Manufacturing companies often refer to a "platform strategy" as a means to (1) satisfy customer-driven variation of their system with a minimal amount of components, and (2) as a means to systematically build and re-use knowledge and technologies to reduce risk in new development. Minimising the number of and management of interfaces also further reduce the cost of realisation. Effective re-use of component technology reduces cost in both development and production through risk reduction and increased automation. Often it is a problem of trading an optimum product against an optimum platform. How this trade is achieved in industrial practice is described by Isaksson et al. (2014) based on their interviews with practicing engineers in a truck company.

There are several ways to represent platforms. Researchers have different views on product platforms and how they relate to adjacent concepts such as product families, modules and brands. An extensive review of platform-related research was performed by Jiao et al. (2007). It is reported that the product architecture is one of the fundamental issues associated with platform research. The review identifies modular product architecture as having received the most attention and lists many contributions related to modular architecture. The products that we deal with in this paper are of the integrated type where few contributions are identified.

A wide scope when studying platforms is exemplified by Robertson and Ulrich (1998). They describe a platform as a collection of assets, components, processes, knowledge people and relationships that are shared by a set of products. An approach along these lines, using configurable system family models as reusable platform elements, configurable components (CC), was proposed by Claesson (2006). It is based on systems theory principles (Hitchins 2003) and design theory (Hubka 2013, Andreasen, 1991). A CC is a system description that functions as a model of the whole system family. It contains information about both the system itself, and the underlying requirements and motivations for the system, i.e. its design rationale. As CC is a model of a multi-functional system family, it can fulfil an arbitrary number of main functional requirements (FRs) of which each one is the root to an Enhanced Function-Means (EF-M) tree (Johannesson and Claesson 2005) branch in the system´s Design Rationale

(DR). In this work, the entire concept of CC's are not made use of but the EF-M tree is used to describe the integrated product in question. The changes in EF-M tree upon adopting different manufacturing splits results in different associations among the design solutions which in turn produces a number of underlying graphs. These graphs are used to analyse the products. A detailed description of the EF-M tree is provided in Sect. 3.3 when the product characterisation of an integrated component is exemplified.

It is clear that an approach based on graph theory combined with platform-based product descriptions is useful not only in terms of representing the product architecture but also in performing analyses on it such as those mentioned in Sect. 2.2 so that relevant information from the architecture could be extracted. This paper discusses usage of un-directed graphs for connecting functions and common sections in a structure, starting from different segments in the assembly.

3 Methods to represent product architectures

Two methods for representing product architectures have been used. The graph theory, that has been primarily used to analyse products as built, and the EF-M method that has been used to support engineering design from a functional perspective. A general description of graphs is given in this section followed by creating the graph model for a simple integrated component, a pump casing. The same casing is then represented using an EF-M tree after introducing necessary concepts related to the EF-M tree. It should be noted that the architecture descriptions are created for already existing products that are in service and is not intended for introducing new products or finding an alternative solution to an existing requirement. The objective is to create a product architecture description of existing integrated components whose architecture is needed to be better understood.

3.1 Graphs

A graph G is a finite non-empty set of objects called *vertices* together with a (possibly empty) set of unordered pairs of distinct vertices of G called *edges* (Chartrand and Lesniak 1996). Figure 2 shows a graph for which set of vertices is,

$$V = \{v1,\ v2,\ v3,\ v4,\ v5\}$$

and the set of edges is

$$E = \{(v5,\ v1),\ (v1,\ v3)\} = \{e1,\ e2\}$$

For the method in this paper, one set of vertices will be formed by the set of functions that are required of a class of products and another set of vertices by the sections (regions or features of the product that satisfies functions)

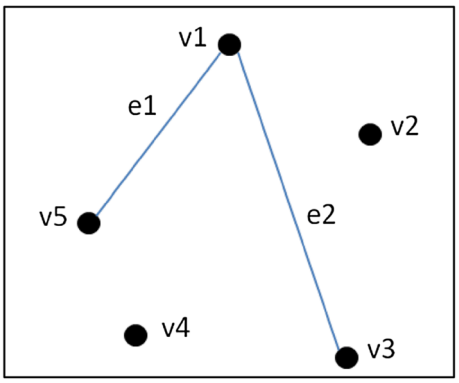

Fig. 2 Example of a graph with five vertices and two edges

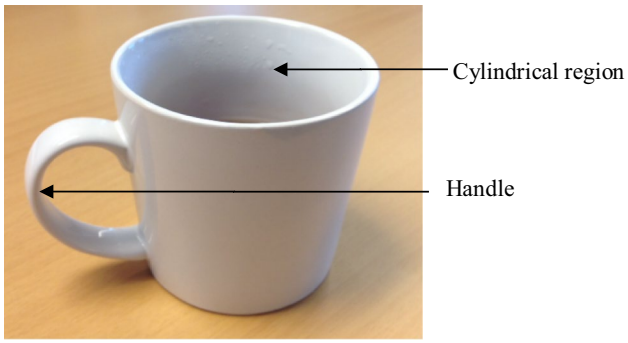

Fig. 3 Sections in a coffee cup

of the products. Consider the example of a coffee cup in Fig. 3. The functions that the coffee cup is expected to satisfy are (a) carry coffee (b) be able to be held in hand. The sections (Raja and Isaksson 2015) that the cup has are (a) cylindrical region where coffee is held and (b) the handle. Sections are distinct regions of the product. For an integrated product, sections are similar to 'chunks'. While 'chunks' are groups of physical elements that satisfy functions, sections are the regions (areas) of the product that can be thought to satisfy a function. Ulrich and Eppinger (1995) also refer to regions for a component which act similar to software subroutine definitions inside the definition for the component. In this paper, we mean sections as identifiable region or area that can be thought to satisfy a function for an integrated product. Sections do not have to be similar to a software subroutine that does only one or a few tasks. They can be associated with more than one task or in this case, function.

Extending this logic of sections satisfying functions in an integrated product, each structure that belongs to a class can be represented as a graph where the vertices are the functions and sections of the structures and the edges are the relationships between them; that is which sections contribute to satisfying a function.

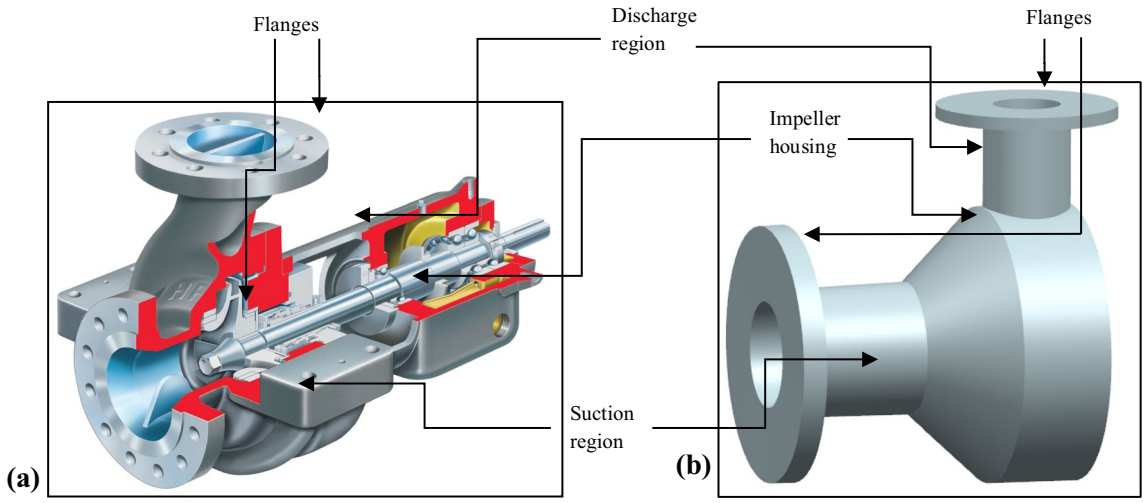

Fig. 4 **a** Sections for centrifugal pump casing (Flowserve Inc) and **b** simplified model of the casing

The relationship between functions and sections for a certain structure that belongs to the class can be identified at first sight only by a skilled design engineer. Besides, different design engineers can have varying opinions about function–section connections. Also, identifying something as a single structure is difficult. For instance, Chakrabarti and Singh (2007) cite examples of designers characterising a pen cap. Some identified the whole cap piece as a single structure and some separated the clip for holding it on to a surface as one and the cylindrical cover as another structure. Therefore, to facilitate some rigour in associating functions to sections, the manufactured segments of the structure were used. Each manufactured segment is associated with a certain section, creating a manufactured segment to section graph. Similarly each manufactured segment is associated with a function, creating a manufactured segment to function graph. It is then possible to associate the functions to sections by eliminating the manufactured segments. This is similar to doing a composition (Goldrei 1996) operation of two functions among three sets with one set being common for both the functions. This is illustrated next.

Let GF be the set of functions, M be the set of manufactured segments and GS be the set of sections. Let G1 be the graph that connects functions (of the product) to manufactured segments and G2 be the graph that connects manufactured segments to sections. Graphs G1 and G2 are essentially mathematical functions f1 and f2 that associate the functions (of the product) to manufactured segments and manufactured segments to sections. The composition (Goldrei 1996) of f1 and f2 can be written as,

$$f1 : \mathrm{GF} \rightarrow \mathrm{M},$$

$$f2 : \mathrm{M} \rightarrow \mathrm{GS},$$

$$f2 \,\mathrm{o}\, f1 : \mathrm{GF} \rightarrow \mathrm{GS}.$$

3.2 Product description of a pump casing using graphs

To illustrate the procedure of associating functions to sections, consider the example of a centrifugal pump casing. Figure 4 shows the different sections of a centrifugal pump (Flowserve Inc) in cross-sectional and simplified cross-sectional views. Three hypothetical manufacturing alternatives for the pump casing are considered. Design-1 has the casing manufactured as one piece, design-2 has four segments, each manufactured separately from one another and design-3 has three cast segments. The three design alternatives and their different manufactured segments are shown in Fig. 5a, b and c.

It is possible to associate each manufactured segment in a pump casing to the generic functions that the casing is intended to satisfy. Similarly, each manufactured segment can also be associated to generic section names (Raja and Isaksson 2015) that is common to all pump casings of a certain type, regardless of how it is manufactured. Thus, the generic functions that each manufactured segment of a pump casing contributes to and the generic sections that the manufactured segment forms part of can be identified. This information is represented in a graph. The nodes in the graph are the generic functions (GF), generic sections (GS) and manufactured segments (M). The relations between the manufactured segments and functions (and the manufactured segments and sections) form the edges of this graph. Sub-figures (a) and (b) of Figs. 6, 7 and 8 show the generic functions to manufactured segments mapping (GF–M) and manufactured segments to sections mappings (M–GS). Once the GF–M and M–GS mappings are made, it is possible to eliminate the manufactured segments from the mappings and generate GF–GS mapping. The

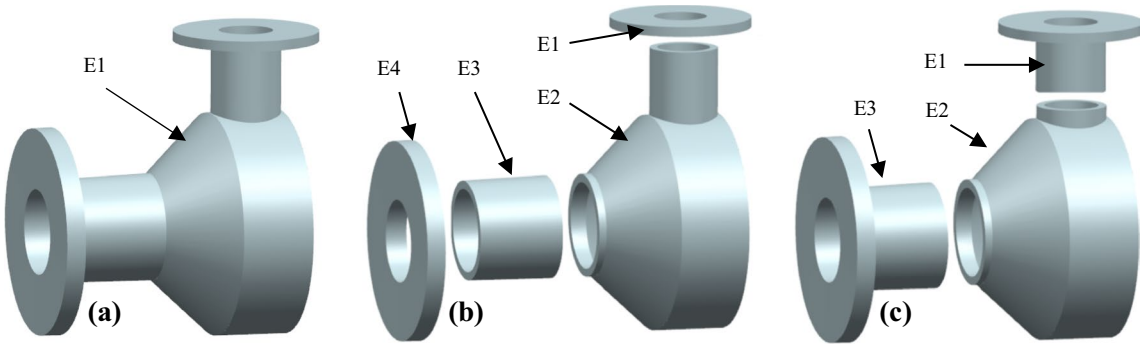

Fig. 5 Manufacturing alternatives of a centrifugal pump casing divided into segments. **a** Single piece casting, **b** 4 manufactured segments: E1, E2, E3 and E4 as castings, **c** 3 manufactured segments: E1, E2 and E3 as casting

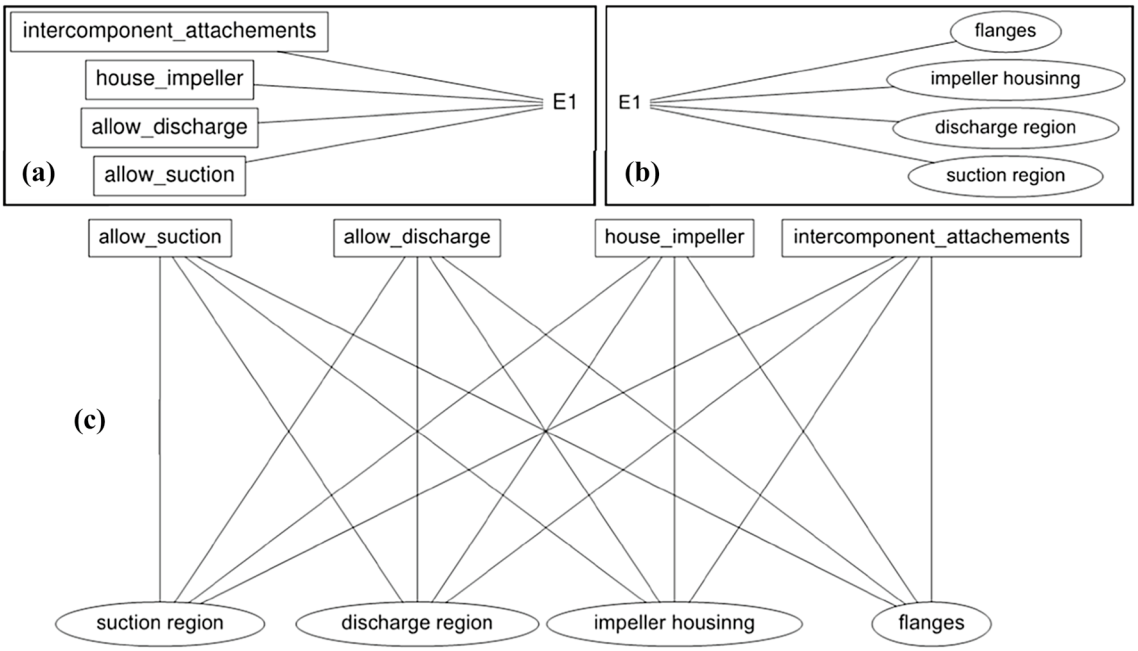

Fig. 6 **a** Generic functions (GF) to manufactured segments (M) mapping (GF–M) for design-1 **b** manufactured segments (M) to generic sections (GS) mapping (M–GS) for design-1 **c** generic sections (GS) to generic functions (GF) mapping (GF–GS) for design-1

GF–GS mapping is akin to composition of functions that represent GF–M mapping and M–GS mapping. Thus,

$$\mathrm{GF} \rightarrow \mathrm{GS} \ : \ \mathrm{GF} \rightarrow \mathrm{M} \ \mathrm{o} \ \mathrm{M} \rightarrow \mathrm{GF}$$

where

$$\mathrm{GF} = \left\{ \begin{matrix} \text{intercomponent\_attachements, house\_impeller,} \\ \text{allow\_discharge, allow\_suction} \end{matrix} \right\},$$

$$M = \{E1,\ E2,\ E3,\ E4\},$$

$$\mathrm{GS} = \left\{ \begin{matrix} \text{suction region, discharge region,} \\ \text{impeller housing, flanges} \end{matrix} \right\}.$$

For simplicity, pump casing has four main functions. Descriptions of the functions and sections are given in Table 1.

The GF–GS mappings are shown in sub-figure c of Figs. 6, 7 and 8.

With reference to Fig. 6, when the product is cast, there are number of lines that connect functions and sections of the component. From the figure, the large number of connecting lines indicates that the design and manufacturing complexity of the product is high. However, the design complexity is high as it may, the manufacturing

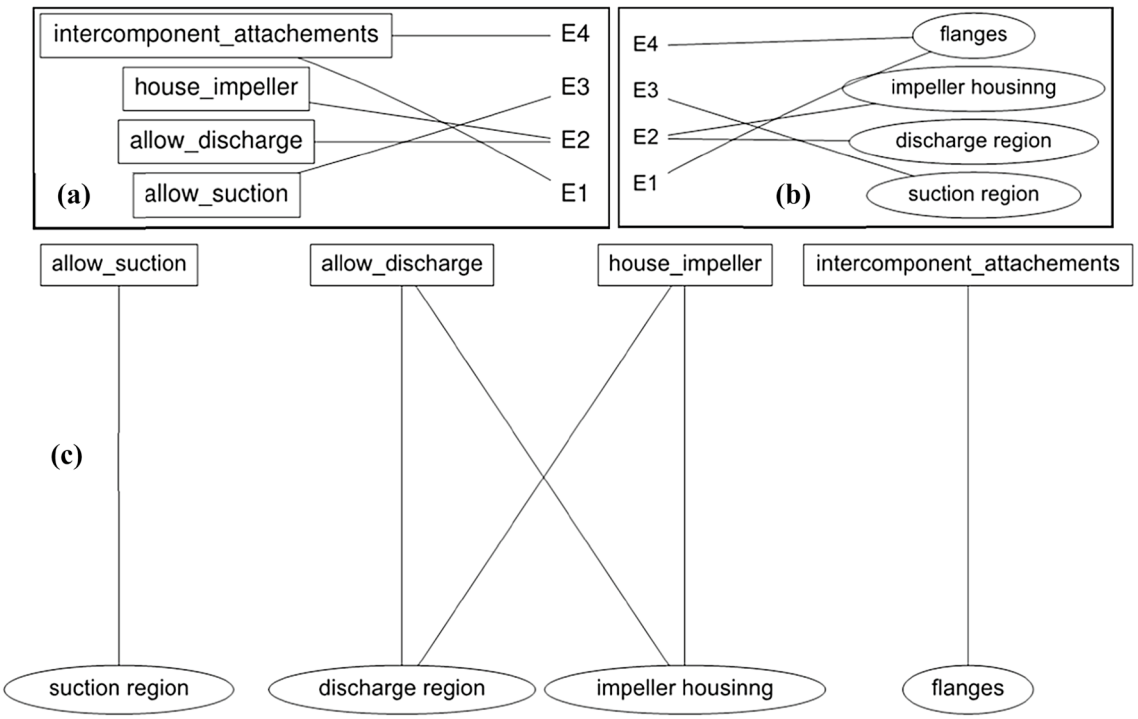

Fig. 7 **a** Generic functions (GF) to manufactured segments (M) mapping (GF–M) for design-2, **b** manufactured segments (M) to generic sections (GS) mapping (M–GS) for design-2, **c** generic sections (GS) to generic functions (GF) mapping (GF–GS) for design-2

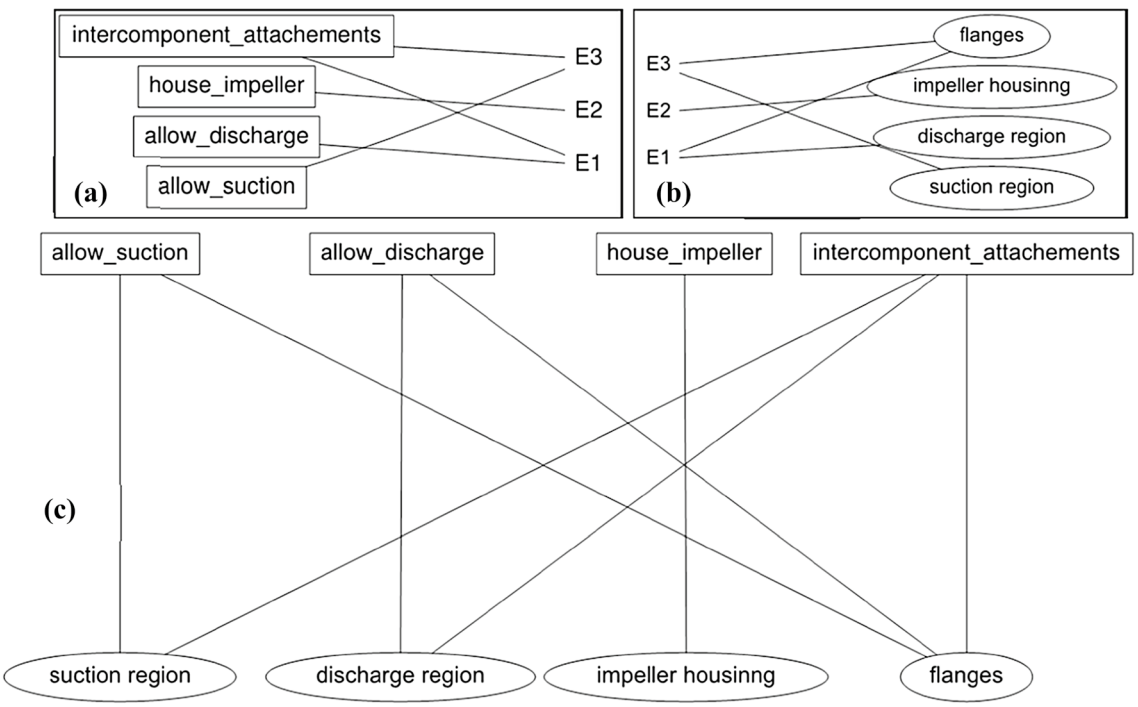

Fig. 8 **a** Generic functions (GF) to manufactured segments (M) mapping (GF–M) for design-3, **b** manufactured segments (M) to generic sections (GS) mapping (M–GS) for design-3, **c** generic sections (GS) to generic functions (GF) mapping (GF–GS) for design-3

Table 1 Description of the functions and sections labels

Function	Function label	Section	Section label
Attach interconnecting components	Intercomponent_attachment	Area where suction takes place	Suction region
Contain the impeller	House_impeller	Area where discharge takes place	Discharge region
Enable discharge of fluid	Allow_discharge	Region where the impeller will be located	Impeller housing
Enable suction of fluid	Allow_suction	Area where other components will be attached	Flanges

complexity is not high[1] since the primary means of production is casting. After casting, the component may undergo machining though complicated operations such as welding are not needed on the structure. Therefore, the said way of associating functions and section is ad hoc and needs detailed definition that takes into account the experience of the designers who decide on the geometry of the component as well as the manufacturing splits of the component. The enhanced function-means modelling (Johannesson and Claesson 2005) can be utilised for adding more details to the definitions.

3.3 The enhanced function-means (EF-M) tree

As indicated in Sect. 2.3, the EF-M tree forms the basis of a configurable component (CC) description of a product platform, by proving its design rationale. As CC is a model of a multi-functional system family, it can fulfil an arbitrary number of main functional requirements (FRs). Each FR in a CC is the root to an Enhanced Function-Means Tree branch in the system´s Design Rationale (DR). A DR with its Enhanced Function-Means trees (EF-M tree) can be seen as a formalised description of a specification of a technical system. In axiomatic design terminology this description exists in the functional and physical domains.

Functional requirements (FRs) are here defined as what a product, or an element of a product, actively or passively shall do to contribute to a certain purpose by creating internal or external effects. In this sense, they motivate the downright existence of a specific solution. The means, organs or design solutions[2] (DSs), are the to-be physical (for example, components or features) or non-physical (for example, service or software) entities that can possibly fulfil a specific FR. The role of the non-functional requirements (referred to here as constraints) is to delimit the allowed design space for the FR-driven DSs. In contrast to FRs, constraints (Cs) do not have specifically allocated DSs. In this paper, we try to describe our products based on functions (FRs) and solutions (DSs) only so that the architecture is better understood, similar to the approach we used for the description using set compositions, described in Sects. 3.1 and 3.2. Including constraints in the description does not contribute to understanding the product architecture and at the same time, is inconsistent with the set composition approach in the generated descriptions (a new category of nodes that represents the constraints will be present in the graphs that results from the EF-M tree-based product descriptions, in addition to the functions and sections category of nodes). Therefore, we did not model the constraints in this paper.

An EF-M-tree (see Fig. 9) represent how higher system level functions are met by means, that in turn require refined functions and so on. The highest level overall functional requirement FR_1 can be solved by a number of alternative design solutions $DS_{1a...k}$ which are linked to FR_1 with isb-relations (the expansions of various relations used in this paragraph, such as 'isb' is available in Fig. 9). rf-relations (which are not used in this paper) are then used to link a DS_1 solution to new FR_{11} and FR_{12} sub-ordinate functional requirements. Design sub-solution alternatives, $DS_{11a...x}$ and $DS_{12a...y}$, are derived to fulfil FR_{11} and FR_{12}, respectively. Next, overall system family constraints ($C_{1a...z}$) are described, modelled as objects, and linked to DS_1 with icb-relations. These overall constraints are also linked to $DS_{11a...x}$ and $DS_{12a...y}$ by means of ipmb-relations indicating that appropriate portions of $C_{1a...z}$ are allocated to these design sub-solutions. These portions plus additional new constraints, emerging as a result of the finalised design decisions, ($C_{11a...m}$ and $C_{12a...n}$) are linked to $DS_{11a...x}$ and $DS_{12a...y}$ with icb-relations. Finally, the abilities of $DS_{11a...x}$ and $DS_{12a...y}$ to fulfil their governing FRs and Cs are evaluated, and feasible DSs, chosen to contribute to the system family bandwidth, are kept. Identified dependencies in terms of interactions between design solutions and identified functional couplings are modelled with iw-relations and iib-relations[3], respectively. For each chosen alternative of the

[1] Castings are complicated to design and manufacture though the cast designs are outsourced which makes the manufacturing complexity at the sourcing firm simple. A more appropriate measure for complexity might be cost though this aspect is not considered in this paper.

[2] Means are renamed from "design parameter" to "design solution". One reason is to make the word "parameter" free, usable in e.g. parameterized designs.

[3] The iib-relations are the relations from FRs to DSs which is the axiomatic design matrix itself. In the initial graph creation method, the iib matrix will be the matrix of relations between GFs and GSs.

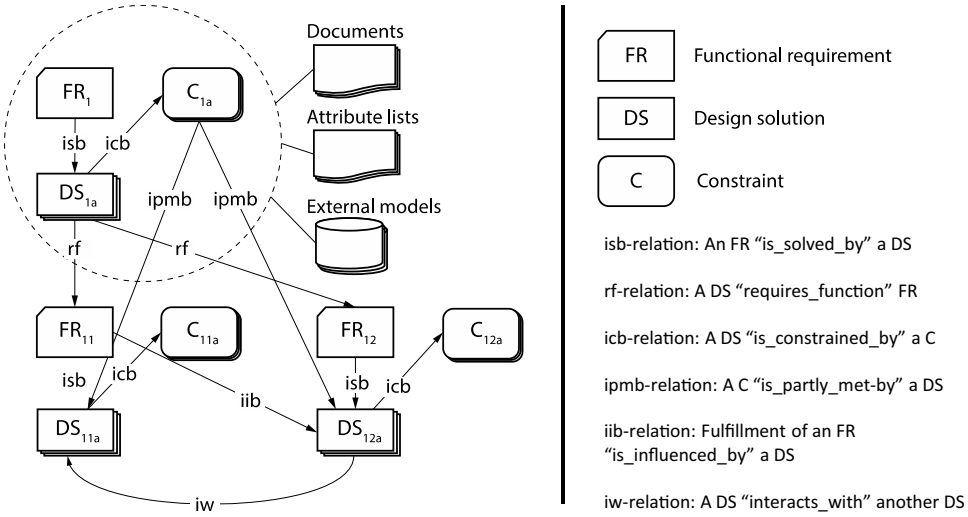

Fig. 9 Enhanced function-means tree adapted from Johannesson and Claesson (2005)

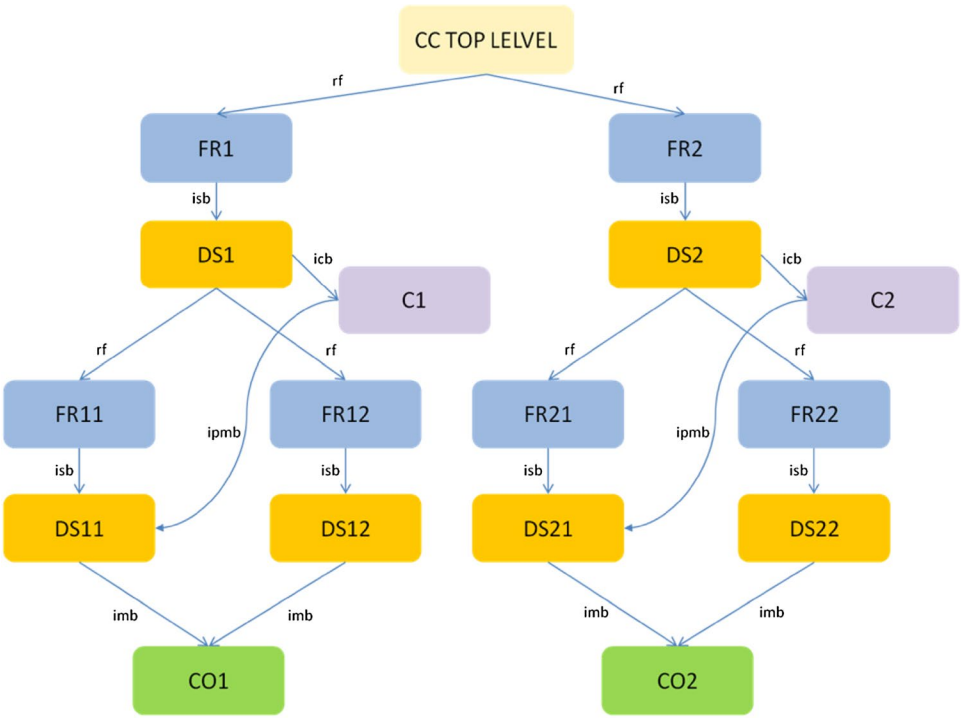

Fig. 10 DR for a system family with two main functions

design solutions $DS_{11a\ldots x}$ and $DS_{12a\ldots y,}$ new functional sub-requirements are formulated and the described procedure is repeated on the next lower hierarchical level and so on.

A design solution (DS) shall be seen as functional feature chosen to fulfil one, and only one, specific functional requirement. It is thus important to make a distinction between physical components/articles/parts (CO) on the one hand and design solutions (DS) on the other. When DSs are to be materialised into COs the following three possibilities exist:

- DS/CO = 1/1 (Single-function component)
- DS/CO = n/1 (multi-functional component, "function integration")
- DS/CO = 1/n (functionality realisation by means of component interaction)

Figure 10 shows a DR for a system with two main functional requirements (FR1 and FR2), decomposed into four sub-functions, which are realised by four bottom line design solutions (DS11, DS12, DS21 and DS22), in turn materialised by three physical components (CO1, CO2 and CO3). Note that a imb-relation (is_materialized_by) connects a DS with a CO.

Fig. 11 EF-M tree for the cast pump casing design (as generated in the CCM tool)

The Design Rationale (DR) can be modelled in a software, called Configurable Component Modeler or CCM (COPE Sweden AB 2014). For the rest of the DR figures in this paper, images of DRs constructed in the CCM software are used.

In this work, the functional requirements (FRs) and design solutions (DSs) are equated to the generic functions (GF) and generic sections (GS) mentioned in Sect. 3.1.

3.4 EF-M tree for the pump casing

To overcome the limitations of the set composition-based, graph theoretical representation (discussed in Sect.3.2) and to make use of already existing design knowledge with the designers, an EF-M tree was created for the pump model, extending on the model created by the set composition. In this case, the function 'intercomponent attachment' was divided into two ('enable suction attachment' and enable 'discharge attachment'), following axiomatic design principles. While creating the initial graph, this was identified as only one function. The 'intercomponent attachment' is a function that needs to be satisfied at multiple locations in the casing. That is, the function is satisfied by different regions of the casing. Axiomatically, one function is satisfied by one and only one design parameter; in our case a design solution, which is a section of the casing. Therefore, the function was split according to the location at which it is demanded, to be satisfied by a section at that region. These sections are the 'suction flange' and the 'discharge flange'. This enables us to use the EF-M tree, the definition of which is based on axiomatic principles. An FR is solved by one and only one DS with an 'isb' relation. Interactions that exist between an FR and DS that do not directly solve it, is represented by the 'iib' relations. Thus the elements in the EF-M tree with the 'isb' and 'iib' relations become the axiomatic design equation.

The cast design for the pump casing can be expressed using an EF-M tree as shown in Fig. 11. In addition to splitting 'intercomponent attachments' into two, all sections (design solutions in this case) are provided by a single-component CO. Since a CO is a single physical structure, the interactions for the sections within the structure are merely physical proximities. This is judged to be uncomplicated and, therefore, the interactions among sections shown in

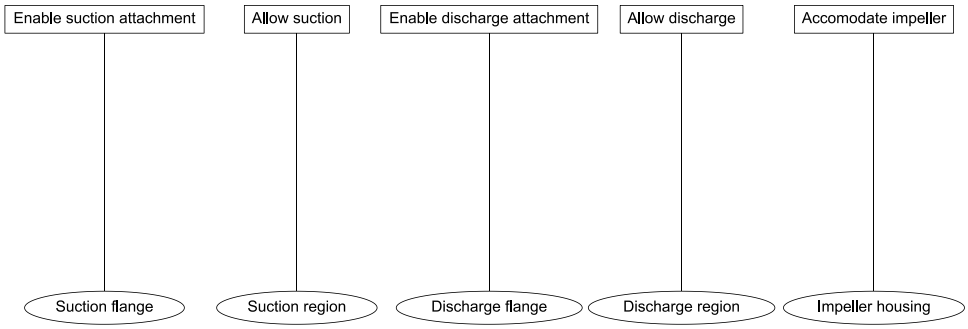

Fig. 12 Graph created from the *iib* matrix for the cast pump casing design

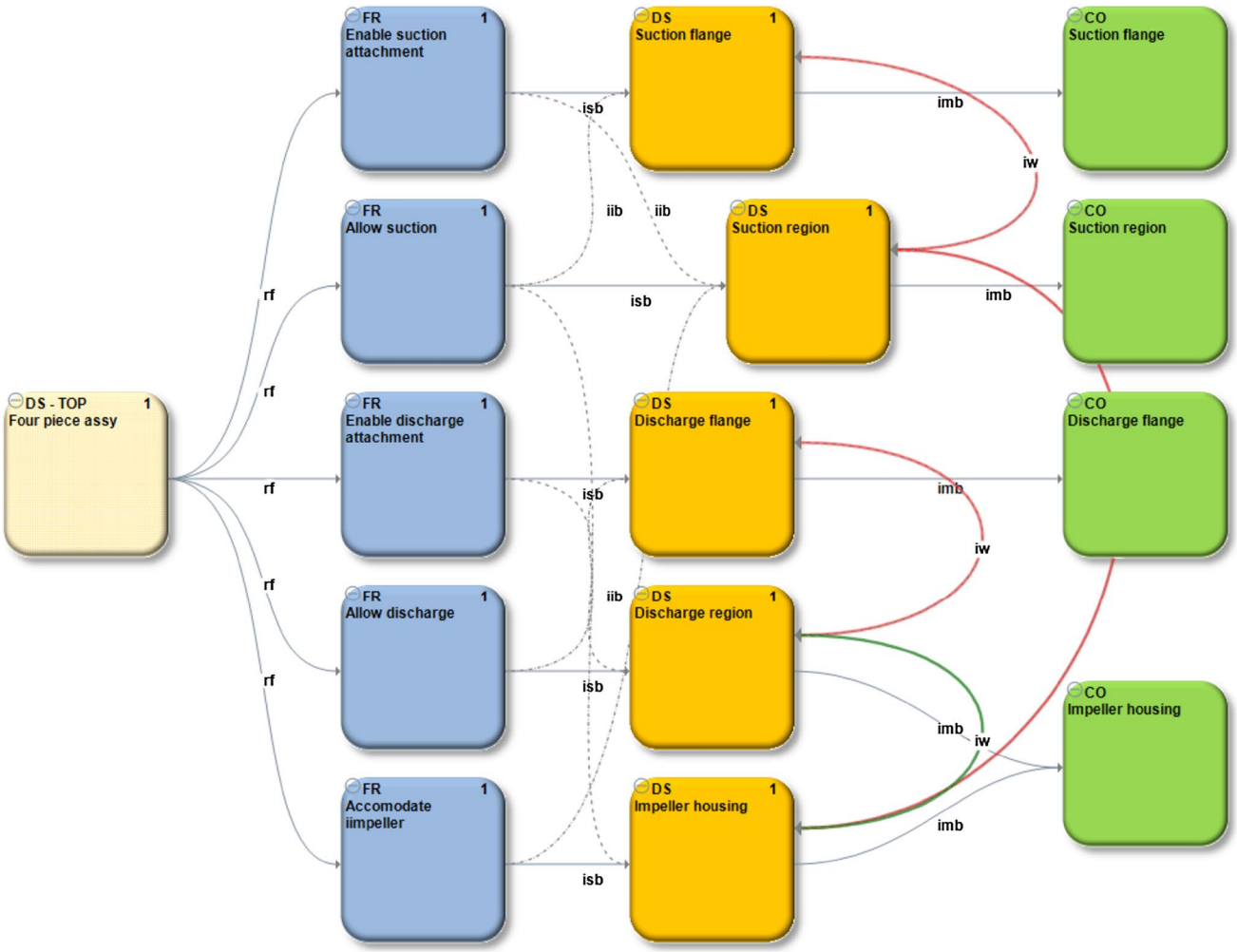

Fig. 13 EF-M tree for the four-piece pump casing design. The four pieces are welded together to produce the casing

Fig. 12 are not as intense as noted in Fig. 6c for the initial graph model.

The EF-M tree generated for the four-piece pump casing design is shown in Fig. 13. The four-piece casing will have four CO's in the EF-M tree. In contrast to the fully cast design shown in Fig. 11, the interaction among different design solutions in the EF-M tree is more complicated since welding is involved while joining the different segments. For instance both 'discharge region' and 'impeller housing' are realised in the same component (CO), termed 'impeller housing' in Fig. 13. The interaction between 'discharge region' and 'impeller housing' is judged to be uncomplicated as these two sections are close to each other in a single, cast segment. However, the interaction of these

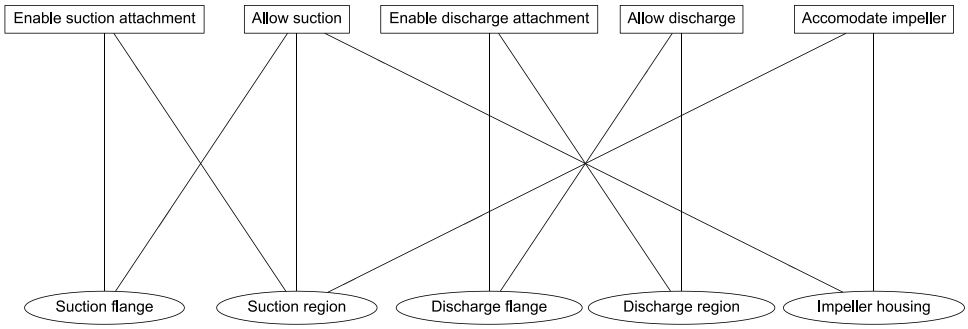

Fig. 14 Graph created from the *iib* matrix for the four-piece pump casing design

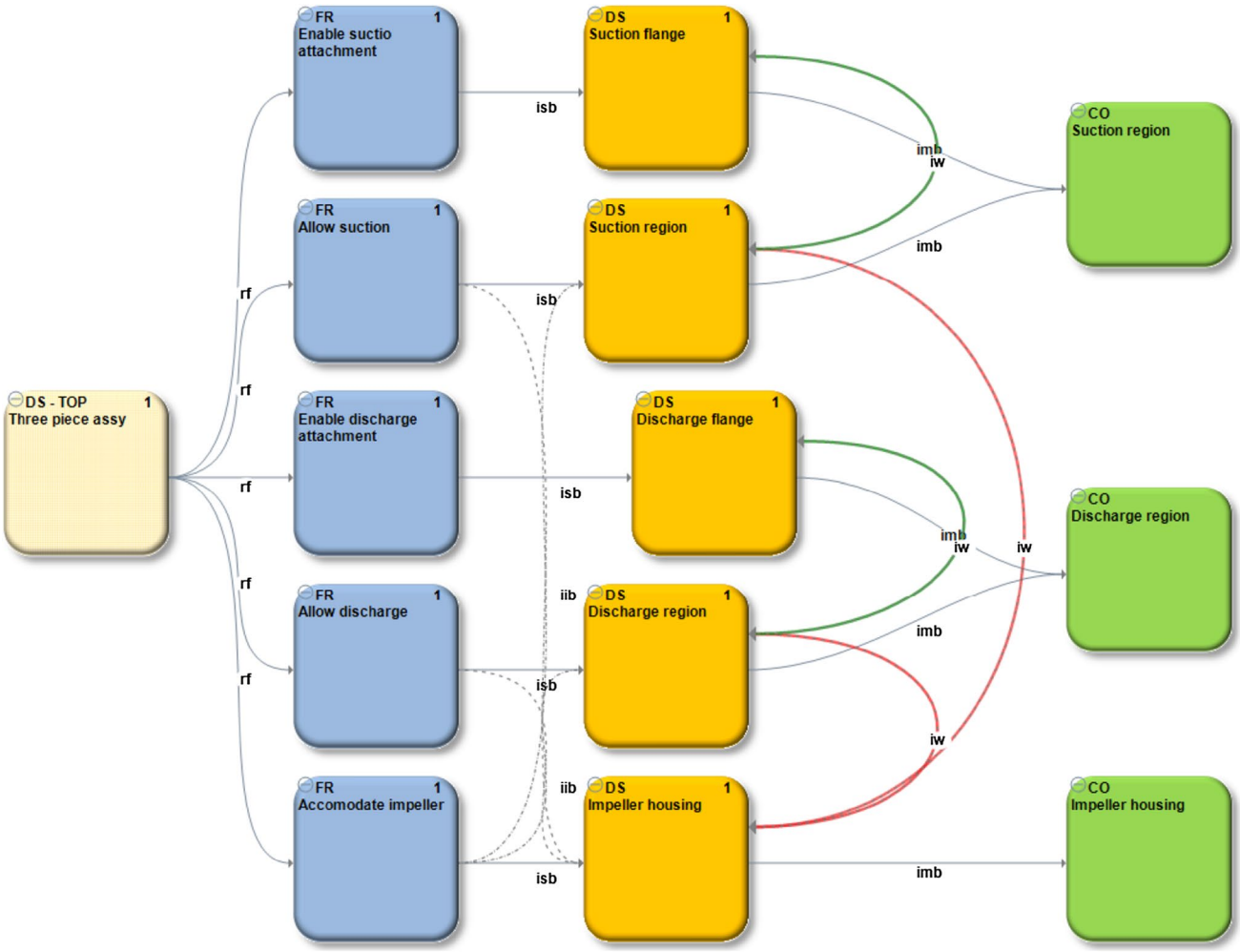

Fig. 15 EF-M tree for the three-piece pump casing design. The three pieces are welded together to produce the casing

sections with other sections (interaction between 'impeller housing' and 'suction region' for instance as they are located on two different COs) might create problems while welding them together. The resulting graph from the interactions that are represented in an *iib* matrix (which is the same as the axiomatic design matrix), for the four-piece pump casing design is shown in Fig. 14.

Finally, the EF-M tree generated for the three-piece pump casing design is shown in Fig. 15. Judging the functional influences on different sections in the same way for the four-piece casting design, the resulting graph from the influence matrix (iib matrix or the axiomatic design matrix) is shown in Fig. 16.

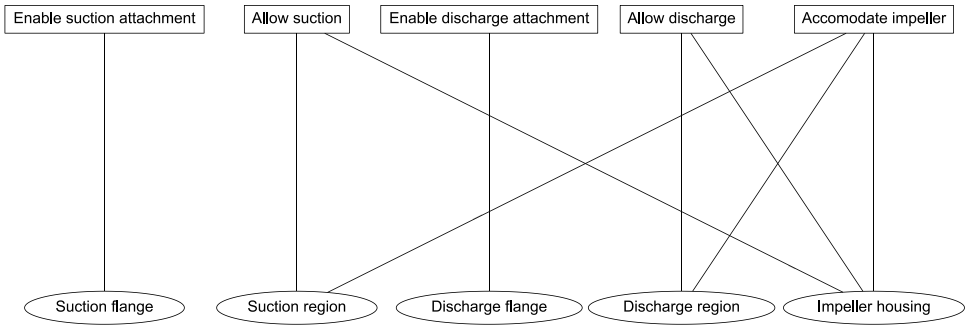

Fig. 16 Graph created from the *iib* matrix for the four-piece pump casing design

3.5 Comparing the graph and the EF-M approach

The approach of dividing the structure into sections is essentially attributing modularity for an integrated component. The modularity derived in this manner is not physical in that mixing and matching different components is not possible. In this paper, we concentrate on product descriptions in terms of functions and sections (solutions). It is possible to apply some of the methods that have been used to evaluate modularity to the matrices that we define from the product descriptions. This will enable comparison of the product descriptions and drawing inferences. For analysing modularity, Holtta-Otto and de Weck (2007) demonstrate fully integrated products, bus modular products and fully modular products using singular value decomposition (SVD)[4] of the respective design structure matrices. When the singular values are sorted in descending order, integrated architecture products show a quick decay pattern while fully modular product shows a more gradual decay pattern. In contrast to the work by Holtta-Otto and de Weck (2007) that uses DSMs as their product architecture descriptions, this paper uses the adjacency matrices of the graphs that connects functions and solutions. In other words, DSMs represent uni-partite graphs (graphs that have only one category of nodes; e.g. parts in an assembly) while our descriptions are bi-partite graphs (graphs that have two categories of nodes with relations only between different categories of nodes and no relations among the same category of nodes). Thus, we use SVD decay pattern in the context of bipartite graphs. Despite the presence of two different categories of nodes, the decay pattern can quickly represent if there are function–section groupings present in the structure. From the work by Holtta-Otto and de Weck (2007), bus modularity would indicate a stepped decay pattern. In the context of bipartite graphs, this means clustering of one category of nodes around a single, other category of node. Since the nodes in our graphs are functions and sections, a stepped decay pattern would indicate presence of either (a) many functions being satisfied by one section or (b) many sections contributing to satisfy one function. Thus a stepped decay pattern will indicate the presence of function sharing or structure sharing (Chakrabarti 2001), within the integrated structure. A drawback of using the patterns will be that it is not possible to identify from the pattern if it is a case of function sharing or structure sharing as the pattern does not distinguish between the categories of nodes present in the graph.

To compare the product representations generated, the decay pattern for the adjacency matrix for a revised initial graph (with five functions and five sections instead of four functions and four sections) was compared to the decay pattern for the adjacency matrix of the graph from the iib matrix. The SVD decay patterns for all three designs of the pump casing are shown in Figs. 17a, 18a and 19a.

For the fully cast design, the SVD decay pattern is a sharp fall as shown in Fig. 17a, indicating the integrated nature of the product in terms of functions and sections. EF-M iib matrix decay pattern is just a single value and shows no decay. The single value also indicates that the function–section groupings present in the description are all of the same kind; equal number of functions is associated with equal number of sections. This is because the influence of sections that do not directly satisfy a function (cross relations) was removed while creating the EF-M tree and thus, the graph for EF-M tree is not as dense as the initial graph. The non-influencing section to function relations was removed because the manufacturing method (casting) is not judged to create problems provided the quality of casting is appropriately managed. For the four-segment and three-segment manufacturing option for the pump casing, the SVD decay pattern is more gradual, indicating presence of function–section groupings as well as function–section integration (similar to Fig. 17) within the structure. The SVD decay pattern for a four-segment casting design using EF-M tree iib matrices shows a much more gradual pattern when compared to the pattern generated from the initial graph description. This is a result

[4] The singular value decomposition (SVD) for any matrix M can be expressed as $M = U \Sigma V^T$ where U and V are orthogonal matrices and Σ is a diagonal matrix containing the Eigen values of M. SVD is often used for finding the structure of matrices and compressing information contained in matrices.

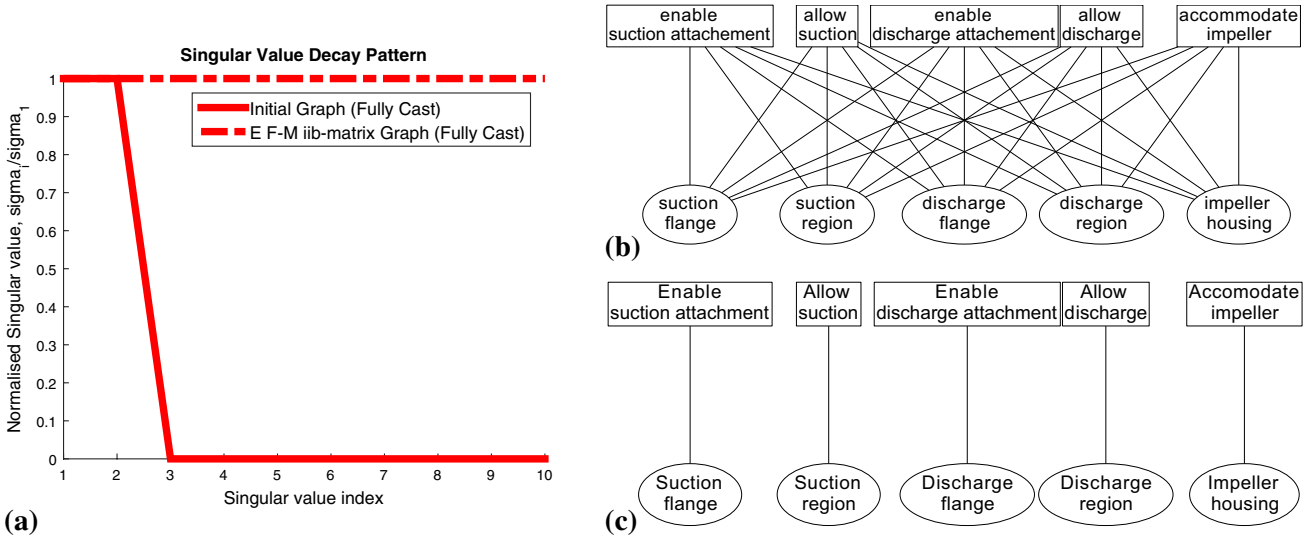

Fig. 17 **a** SVD decay pattern for the graph descriptions for fully cast (manufacturing option-1) pump casing, using initial graph generation approach and EF-M iib matrix approach, **b** Initial graph for fully cast pump casing option, **c** graph from EF-M tree for fully cast option

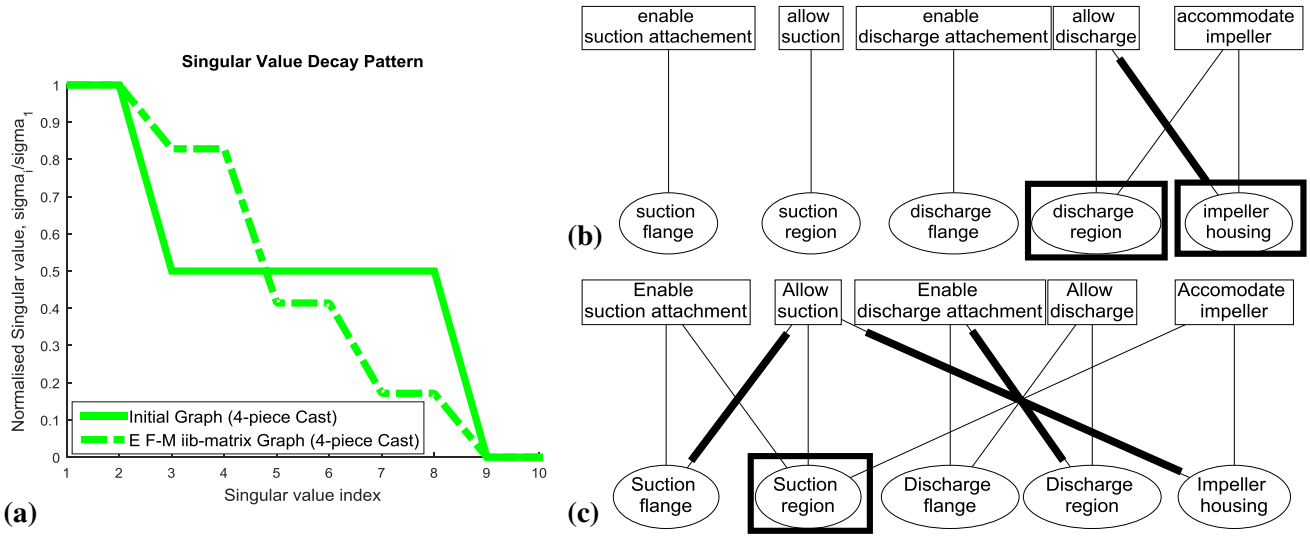

Fig. 18 **a** SVD decay pattern for the graph descriptions for four-piece cast (manufacturing option-2) pump casing, using initial graph generation approach and EF-M iib matrix approach, **b** Initial graph for four-piece cast pump casing option with highest edge-betweenness centrality cross-relationships highlighted. Sections with highest degree centrality are also highlighted, **c** graph from EF-M tree created for four-piece cast option with highest edge-betweenness centrality cross-relationships highlighted. Section with highest degree centrality are also highlighted

of manually removing a number of non-influencing sections to function relations. Thus, despite the initial graph description being a quick method, it results in grouping complexities in the product. This can be rectified if looked at by a designer using the EF-M tree, a fact demonstrated by the difference in decay patterns for the initial graph and EF-M generated graph, shown Fig. 18.

The most important nodes in a graph are often determined by the centrality of the graph. Different centrality measures, both for the nodes and edges in a graph, are available in the graph theory literature. In this paper, nodes of the graph are the sections and functions of a structure. Therefore, the most important section of a structure can be determined from the centrality of the sections in the graphs that describe the structure. One such centrality measure that concerns the nodes in a graph is the *degree centrality. Degree centrality* just notes the number of incident edges on to a node. The node with the greatest number of incident edges is the most central node in the graph. Since the graphs considered here are undirected (edges do not indicate directions) and

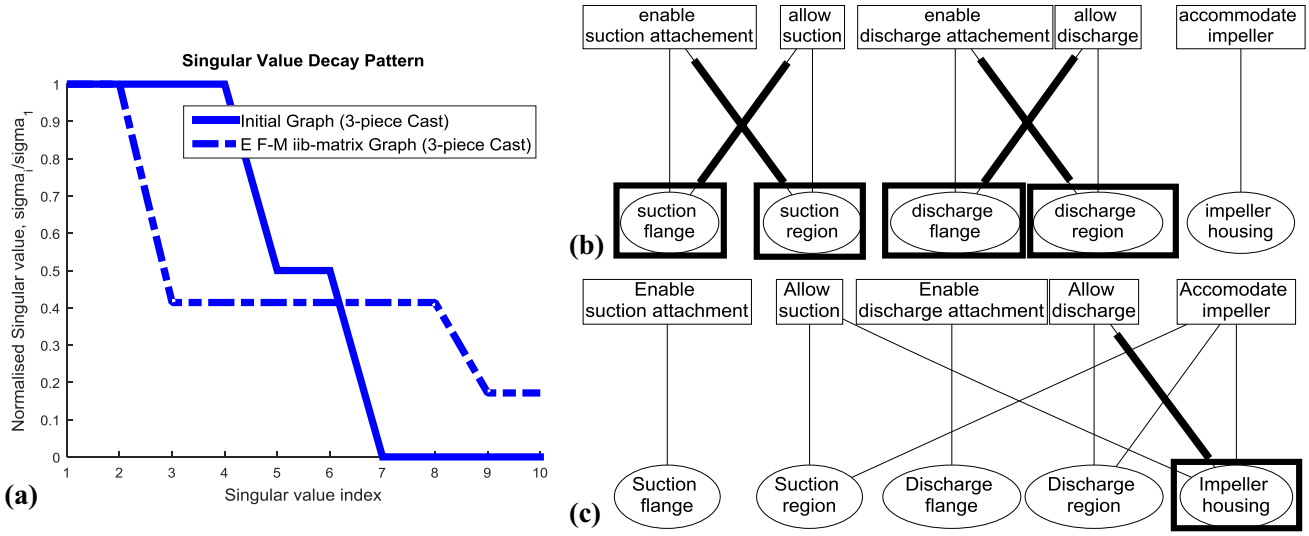

Fig. 19 **a** SVD decay pattern for the graph descriptions for three-piece cast (manufacturing option-3) pump casing, using initial graph generation approach and EF-M iib matrix approach, **b** Initial graph for three-piece cast pump casing option with highest edge-betweenness centrality cross-relationships highlighted, **c** graph from EF-M tree created for three-piece cast option with highest edge-betweenness centrality cross-relationships highlighted

bipartite (meaning that relations exist only from the set of general functions, GF, to the set of general sections, GS and no relations exist within the GF and GS set), there is no hierarchy that can be observed in the graph. A function is satisfied by one or several sections and the sections in turn do not give rise to sub-functions or sub-sections. The most important section can be identified by looking at the section that has the highest degree centrality. This just means that the section with the highest degree centrality contributes to satisfying a number of functions.

In Fig. 17b, c, Fig. 18b, c and Fig. 19b, c, the sections in each graph with the highest degree centrality are marked. For example, in the initial graph for four-segment pump (Fig. 18b, c), sections 'discharge region' and 'impeller housing' both have a degree of 2 (2 incident edges). In the EF-M tree generated graph, where more relationships among sections were manually identified, the section with the highest degree is 'suction region' which was not identified in the initial graph.

To ascertain the importance of GF–GS relations, a centrality measure for the edges, the edge-betweenness centrality, can be used. The edge-betweenness centrality[5] for an edge is defined as the ratio of the number of shortest path passages (between all pairs of nodes in the graph) through the concerned edge to the number of total number of shortest paths (between all pairs of nodes in the graph) in the graph. With respect to the GF–GS relationships, a particular GF–GS edge with a high-edge-betweenness centrality indicates that this edge (relationship) needs to be traversed a number of times when tracing any function to a section in the graph which indicates that the edge (relationship) is an important one in terms of the integrity of the structure. When a particular function is satisfied by sections other than those intended for satisfying the function (cross-relationships), important cross relationships can be identified by looking at the cross edge with high edge-betweenness centrality.

In Fig. 17b,c, Fig. 18b, c and Fig. 19b, c, the cross-edges in each graph with the highest degree centrality are highlighted. For example in the three-segment pump (Fig. 19b, c), betweenness centrality for the initial graph predicts high importance for four relationships which are, enable suction attachment—suction region, allow suction—suction flange, enable discharge attachment—discharge region and allow discharge—discharge flange. However, with the EF-M tree generated graph, where relationships have been manually modified, only one high betweenness centrality cross-relationship is identified which is the allow discharge—impeller housing relationship.

GF to GS associations created in the initial graph might not identify important GF–GS relations which are manually identified while creating the EF-M tree. The initial GF to GS relationships were also revised during the creation of the EF-M tree. A combined look at the all predicted important cross-relations will give the designer information about which sections of the design should be paid attention to while deciding on a manufacturing split. It will also function

[5] For a graph G, the betweenness centrality of an edge, e, is defined as $\mathrm{EB}(e) = \sum_{v_i} \sum_{v_j} \frac{\sigma_{v_i v_j}(e)}{\sigma_{v_i v_j}}$ where $\sigma_{v_i v_j}$ is the number of shortest paths between edges V_i and V_j, $\sigma_{v_i v_j}(e)$ is the number of shortest path that passes thorough edge e.

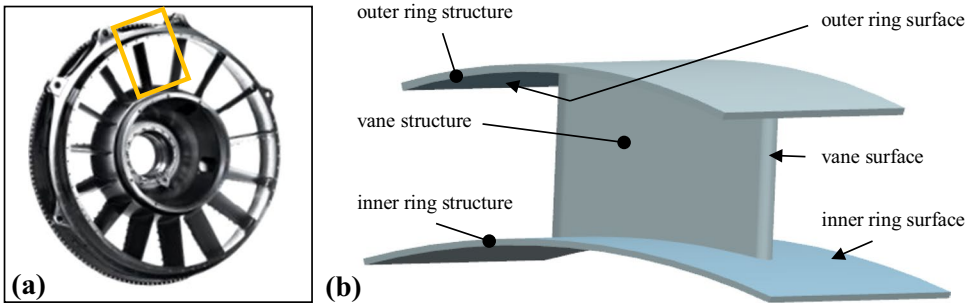

Fig. 20 **a** A turbine rear structures with a vane section highlighted. **b** A simplified view of the vane with different sections

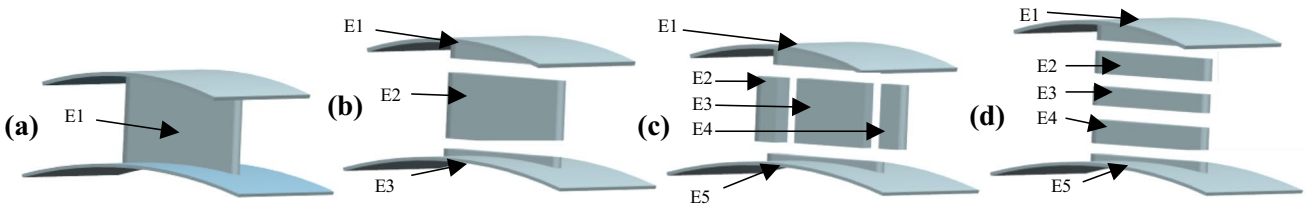

Fig. 21 Different possibilities for manufacturing segments for the vane **(a)** fully cast **(b)** vane, outer ring and inner ring structures as separate casts **(c)** outer ring and inner ring structure as separate casts, while vane is split along radial planes into different casts **(d)** outer ring and inner ring structure as separate casts, while vane is split circularly into different casts

as a tool to display project chief engineers (who are already aware of different sections in the structure) about the potential implications of a manufacturing split without displaying complicated CAD images. It should be kept in mind that all of the functions and sections are associated with only one structure. When the different manufacturing segments are joined, all sections exist in one single segment, identified by one part number by the customer. However, to facilitate analysis of the structure, it has been partitioned; different regions in the structure are designated as sections.

4 Application in an aero engine structural component

The method described for characterising architecture of an integrated component was applied to a section of an aerospace component with several alternative manufacturing options. The section and the characterisation are described in the following sections.

4.1 Description of the structure

A guide vane in the core gas flow of a jet engine is a typical design solution, occurring in multiple cross sections of a jet engine. The primary function of a guide vane is to guide the flow from one module to the other (for instance, from the HP compressor to the LP compressor). In many instances, the vane also carries loads generated from the engine shafts towards engine outer case. Vanes also host devices such as oil supply tubes and sensors so that they do not hinder the flow. The location and various sections of the vane in a turbine rear structure are shown in Fig. 20.

The vanes are integral in the structure and are generally formed as an integral part of the casting. From an engineering design view, the sections displayed in Fig. 20 are present in all product variants available, yet can be manufactured in many different ways. In this example, the vane itself is considered as one integral structure and four different manufacturing segmenting options are considered for the vane. The different manufacturing options are shown in Fig. 21.

For each option, a graph was created for the vane structure as explained in Sect. 3.2 as well as from EF-M tree, as explained in Sect. 3.4. "Appendix A" includes the EF-M trees for all the manufacturing options. The singular value decay pattern as well as the centrality measures from both styles of modelling is presented in Sect. 4.2.

4.2 Application

The SVD decay pattern and the graphs generated using the two methods for each manufacturing option detailed in Fig. 21, are shown in Figs. 22, 23, 24 and 25. The graphs from the set composition graph creation method for all four manufacturing options (sub figures (b) of Fig. 22 through Fig. 25) have the same degree centrality. Thus the initial graph creation method predicts equal importance for all sections. However, the graphs generated using EF-M tree iib matrix predicts different importance for different sections

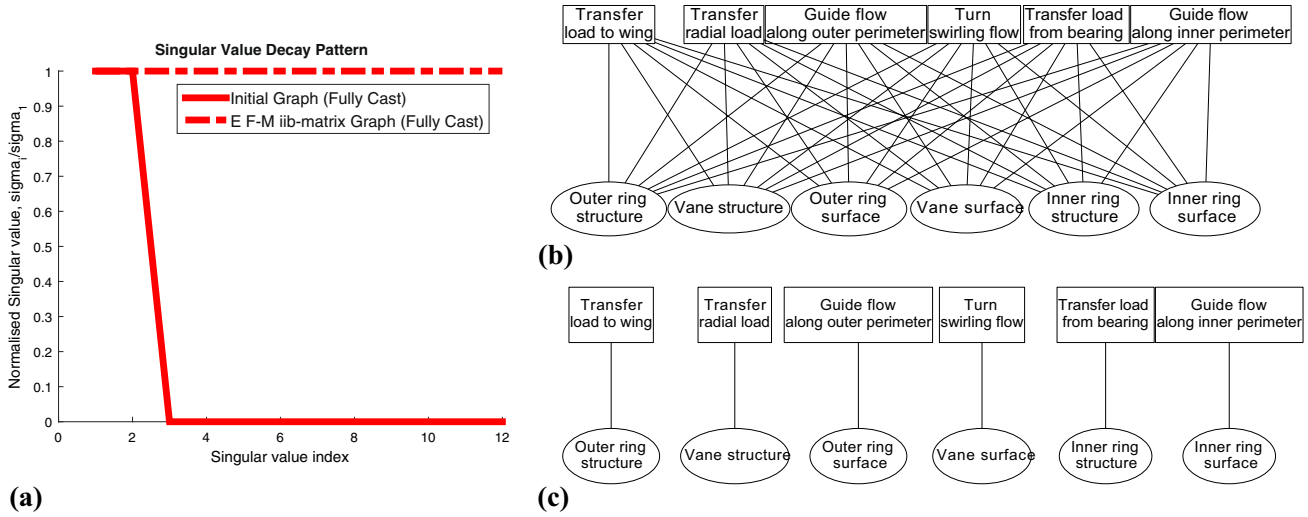

Fig. 22 **a** SVD decay pattern for fully cast vane shown in Fig. 21a. **b** Initial graph; every section has the same degree centrality and each edge has the same betweenness centrality and thus, sections and edges are of equal importance. **c** Graph generated from the EF-M tree iib matrix. Similar to initial graph, each section has the same degree centrality and each edge has the same betweenness centrality but the graph is much less denser compared to the initial graph

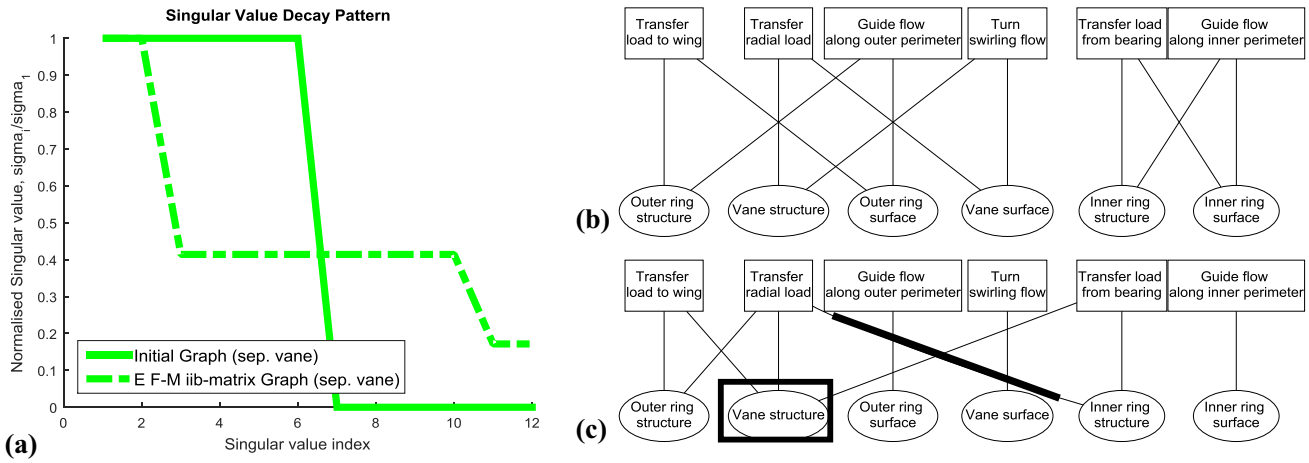

Fig. 23 **a** SVD decay pattern for the separate vane shown in Fig. 21b. **b** Initial graph; every section has the same degree centrality and each cross-edge has the same betweenness centrality and thus, section and edges are of equal importance. **c** Graph generated from the EF-M tree iib matrix. The section with the highest degree centrality and the edge with the highest betweenness centrality are highlighted

as can be observed from sub figures (c) of Fig. 22 through Fig. 25.

With the initial approach for generating a graph, the graph produced exhibits a SVD decay pattern very similar to that of an integral component, irrespective of how the vane structure is split for manufacturing. This can be observed in Fig. 22a to Fig. 25a. However, the patterns for graphs generated from the EF-M tree *iib* matrices indicate a different behaviour compared to that of the initial graphs. For the fully cast vane structure option, the SVD decay pattern is similar to an integrated component, as shown in Sect. 3.5. The pattern for the split vane option (Fig. 21b), shown in Fig. 23a is more gradual and very similar to the patterns for the radial and circular split options (shown in Fig. 24a and Fig. 25b, respectively). The EF-M trees for each manufacturing option for the vane was created such that interactions among different design solutions were considered depending on which splits do the design solutions (sections) belong to. Therefore in case of the vane, the function–section groupings (or an apparent modularity in the integrated structure) inherent in the splits were better described by the EF-M tree modelling. The EF-M trees for the four manufacturing options are included in "Appendix A".

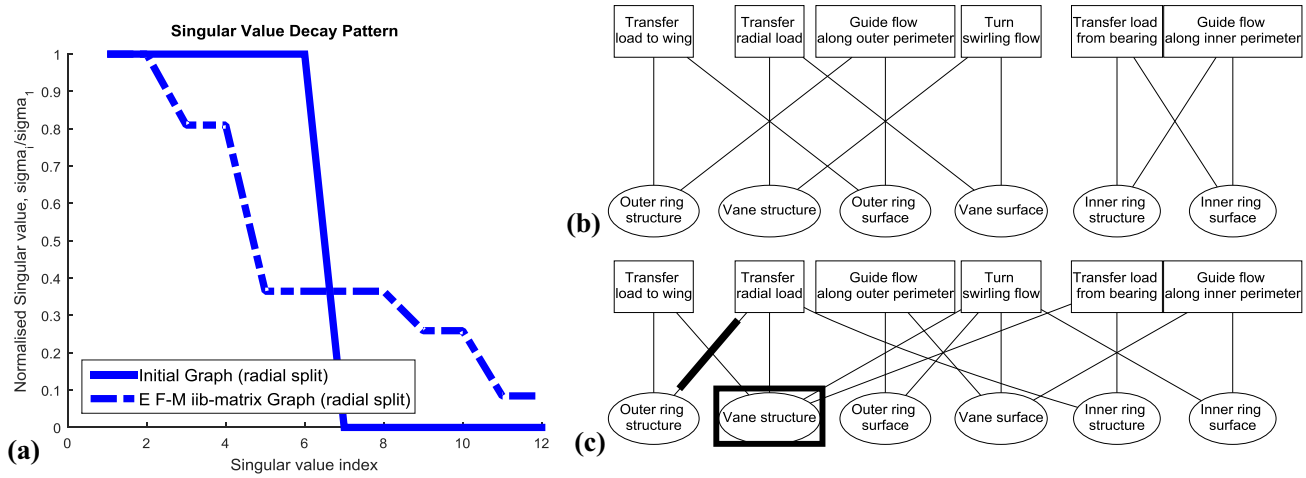

Fig. 24 **a** SVD decay pattern for the radial split vane shown in Fig. 21c. **b** Initial graph; every section has the same degree centrality and each cross-edge has the same betweenness centrality and thus, section and edges are of equal importance. **c** Graph generated from the EF-M tree iib matrix. The section with the highest degree centrality and the edge with the highest betweenness centrality are highlighted

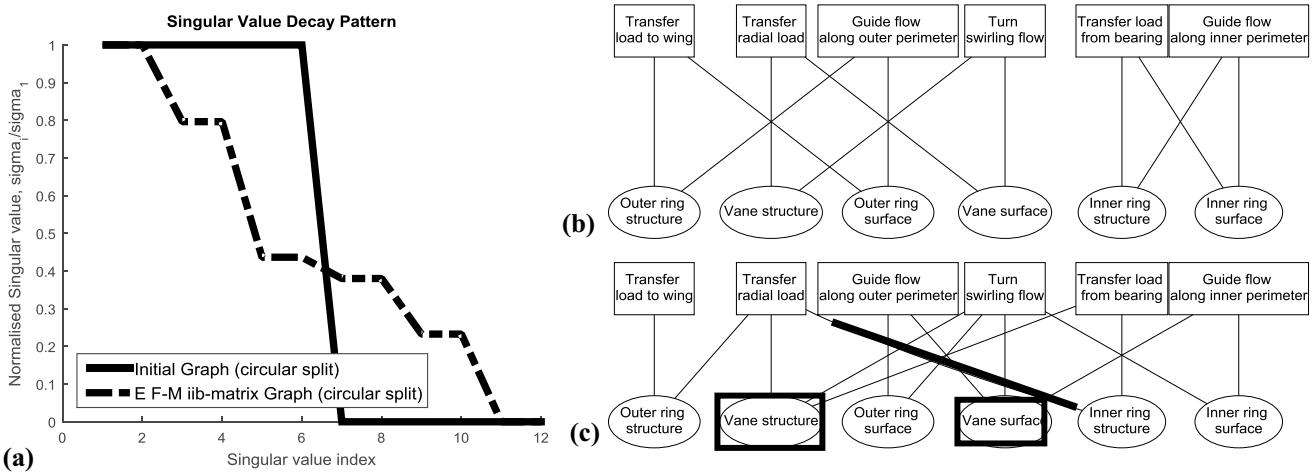

Fig. 25 **a** SVD decay pattern for the circular split vane in Fig. 21d. **b** Initial graph; every section has the same degree centrality and each cross-edge has the same betweenness centrality and thus, section and edges are of equal importance. **c** Graph generated from the EF-M tree iib matrix. The section with the highest degree centrality and the edge with the highest betweenness centrality are highlighted

The degree centrality of the vane changes depending on which manufacturing split option is chosen. For instance, both for the separate vane and radial split vane option, 'vane structure' is the most important section while for the circular split option, the section 'vane surface' also becomes important. Thus, sectional importance is directly affected due to which splits each section belongs to and this can be isolated from the centrality measures in the graphs. Similarly, the edge-betweenness centrality for different cross-relationships also changes depending on how the structure is split for manufacturing. The separate vane option (Fig. 23b) as well as the circular split vane option (Fig. 25b) have the same GF to GS relationship as the most important, which is the transfer radial load–inner ring structure relation. Therefore, depending on the functional dependence of different sections on the structure alone, critical sections can be identified and subjected to further studies or detailed analyses.

5 Discussion and conclusions

This study was concerned with representing the functional product architecture (physical organisations of parts in a component to satisfy the functions) for an integrated product considering alternative manufacturing options. For integrated products, conventional architecture representations such as DSMs (Design Structure Matrices) or node link diagrams are insufficient in indicating how the product satisfies the functions required of it since the matrices have only one element in it, as is the case when using a fully integrated component such as a complex casting. To solve the representation problem, it was proposed that each manufactured segment of the product (such as a cast segment) be associated with a generalised function and a generalised section. Generalised functions (GFs) are the collection of all functions that a class of products are typically required to satisfy. Generalised sections (GSs) are the collection of all regions present in different products that contribute to satisfying the functions. Since the same manufactured segment of the product is associated with generalised functions and generalised sections, a GF–GS mapping can be created from a set composition operation and represented in a graph. The GFs and GSs form the nodes and relations between GSs and GFs form the edges of the graph. In addition to the approach that associates manufactured segments to GFs and GSs, an enhanced function-means (EF-M) tree for the components was created so that already existing design knowledge can be included in the architecture representation. Graphs can then be generated from the axiomatic design matrix (iib-matrix) for the EF-M tree which is similar to the initially created graph but with more refined functions to sections to relationships in it.

Graphs were generated for three separate manufacturing options for a pump casing and on a section of an aeroengine structural component. It can be seen that the set composition-based graph creation method, using the manufactured segments to associate generic functions and generic sections, results in too many relationships among functions and sections. When the generic functions were associated with generic sections manually, taking help from the manufactured segments so that an enhanced function means (EF-M) tree is created, the relationships tended to be less dense. Since the underlying graph structure essentially represents the architecture of the component considered, graphs generated from the two methods (initial method of associating GFs to GSs through manufactured segments and the more-refined method, where a graph is created from an EF-M tree) were evaluated using measures applicable to the underlying matrices for the graphs. Considerable difference in evaluated measures can be observed for entire products (pump casing) and for sections of products (vane structure in an aero engine static structure) when generating graphs based on the two methods. The set composition-based graph creation method, while producing different graphs for different manufacturing options, results in indistinguishable graphs for sections of components (the vane structure). This is due to the limited number of functions and sections associated with the vane structure. The EF-M tree-based graph generation on the other hand involves manual decisions while setting the interactions among different design solutions and generates distinctive graphs for a complete component (the pump casing) or a section of a component (the vane structure). Thus the methods considered can be used as a sequence of measures to generate and evaluate information about integrated products. To begin with, the initial GF to GS graph creation method can be used to create an overall idea of the product with different function to section associations. Then, each section can be looked in detail with the EF-M tree and analysed separately. This will help designers realise previously hidden information or look in more detail at a particular section or function to section relation (as demonstrated by the centrality measures discussed in Sect. 3.5). For example, the graphs for the two manufacturing method for the vane structure (Figs. 24, 25) appear very similar. Thus the architecture representations based on the two manufacturing options are identical. In such cases, designers can direct their attention at other distinguishing characteristics so that a certain manufacturing split is favoured over the other. In the vane structure case, the radial split causes weld lines to be present in the flow path of the structure while the circumferential split causes discontinuities to be present in load path. Decision about choosing a manufacturing option then becomes weighing the impact on load paths and flow paths due to the selection of a certain manufacturing split. Furthermore, the EF-M trees might also enable the designers to build hierarchies into the design such that a certain SVD decay pattern is observed which may indicate the possibility of particular manufacturing split. If the initial function–section association is combined with the creation of detailed EF-M trees at early stages of development, it might help designers realise the implications of different manufacturing options on the product's functional realisation. Therefore, product architecture evaluation based on manufacturability considerations is facilitated.

It should be noted that the definition of generalised functions (GF) and generalised solutions (GS) is the basis on which the representation is constructed. A change in the description of GF or GS such as splitting a function or section into two and denoting with different names will markedly change the make-up of the representation. However, provided the GFs and GSs are rigorously established for a given class of products, such issues can be avoided. In this paper, the method was shown to be applied to products

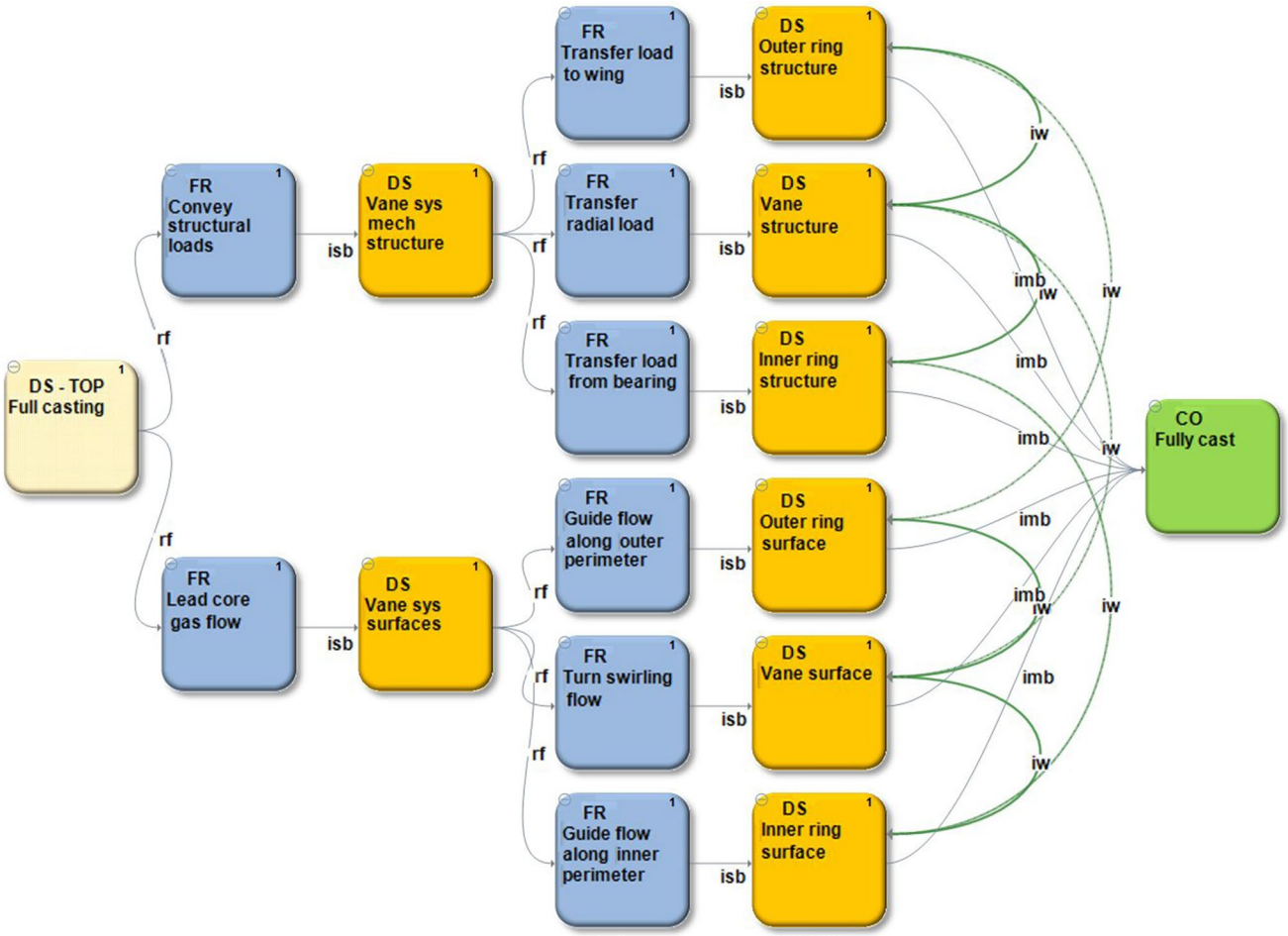

Fig. 26 EF-M tree for the fully cast vane. Sect. 4.2 shows the corresponding graphs (Fig. 22) with discussions

that are made to a more or less exhaustive set of customer specifications. The freedom to choose the shape or contours of the product is limited. However, what sets the product competitive is how it is manufactured. It is important that the manufacturability of the product is also considered when deciding the functional allocation to different sections of the product.

For further work, it could be possible to associate analysis results from detail design stages to the GF–GS representations. For instance, the areas of a certain structure where the fatigue life is too low could be noted and associated to a section in the structure. Since there is a GF–GS representation available for the structure, the functions that cause a low fatigue life can be identified by looking at which functions the sections satisfy. Thus, when there is a change in functional requirements, based on the GF–GS representations, it might be possible to predict the kind of analyses that might be needed (life analyses for instance) at what sections. This could help in analysing only the area that is of interest (a sub-model analysis for example) rather than considering the entire structure. The analysis of sections affected by changes in functional requirements also enables examining the implications of the requirement changes on the manufacturing splits. This could enable a systematic comparison of alternative product architectures based on the available manufacturing options and the likelihood of change in functional requirements. A further direction is the inclusion of constraints while evaluating the implications of a manufacturing split using the 'C' objects in the EF-M tree. When less-established manufacturing methods such as additive manufacturing is employed, the same split might be feasible with different manufacturing methods and the inclusion of 'C' objects in the EF-M tree will then enable differentiating among different feasible alternatives. Finally, a wider definition of functions can be utilised while characterising the architecture so that additional lifecycle aspects such as

maintainability and recyclability could be included in the architecture definition.

Acknowledgements 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). The authors further acknowledge support from GKN Aerospace Engine Systems, Sweden.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Appendix A: EF-M trees for the vane structure

See Appendix Figs. 26, 27, 28 and 29.

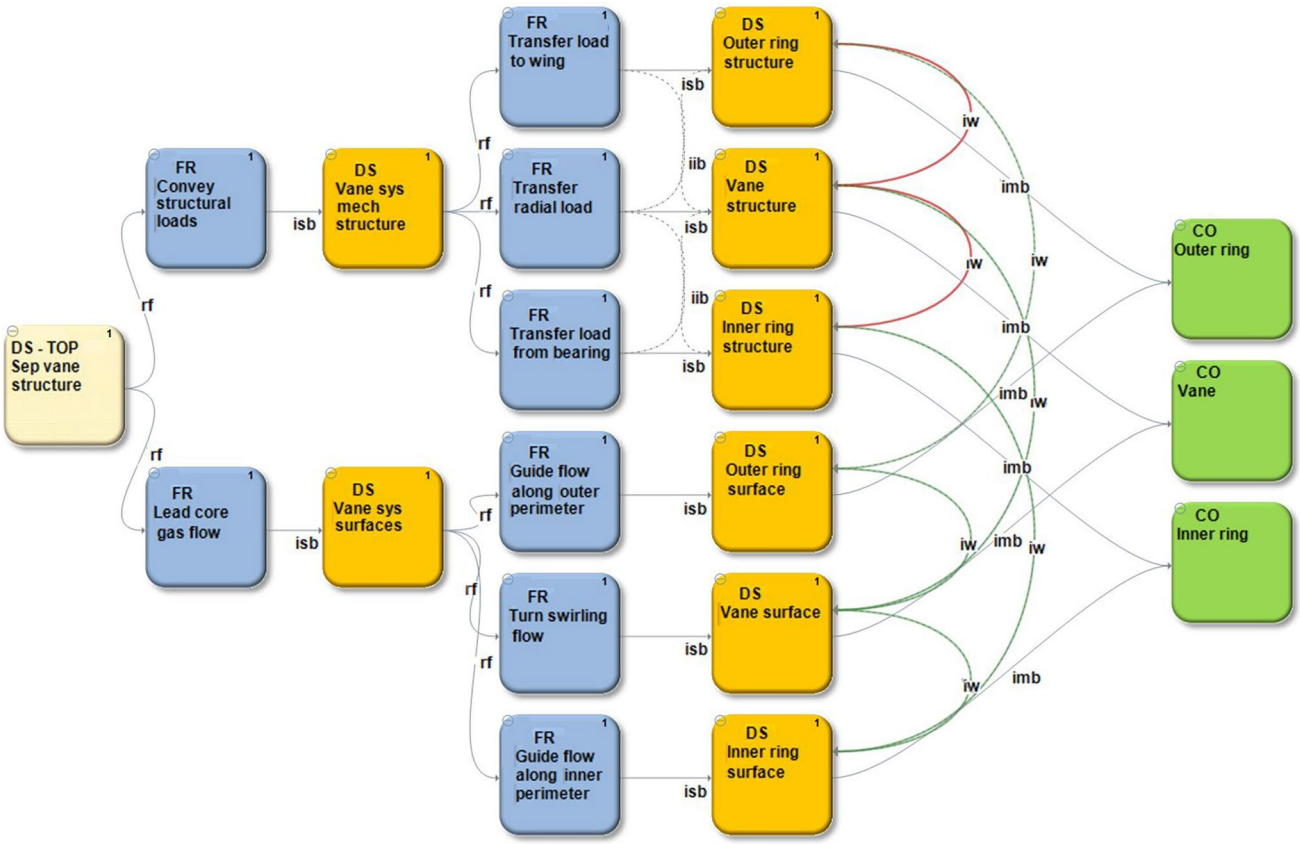

Fig. 27 EF-M tree for the separate vane option. Sect. 4.2 shows the corresponding graphs (Fig. 23) with discussions

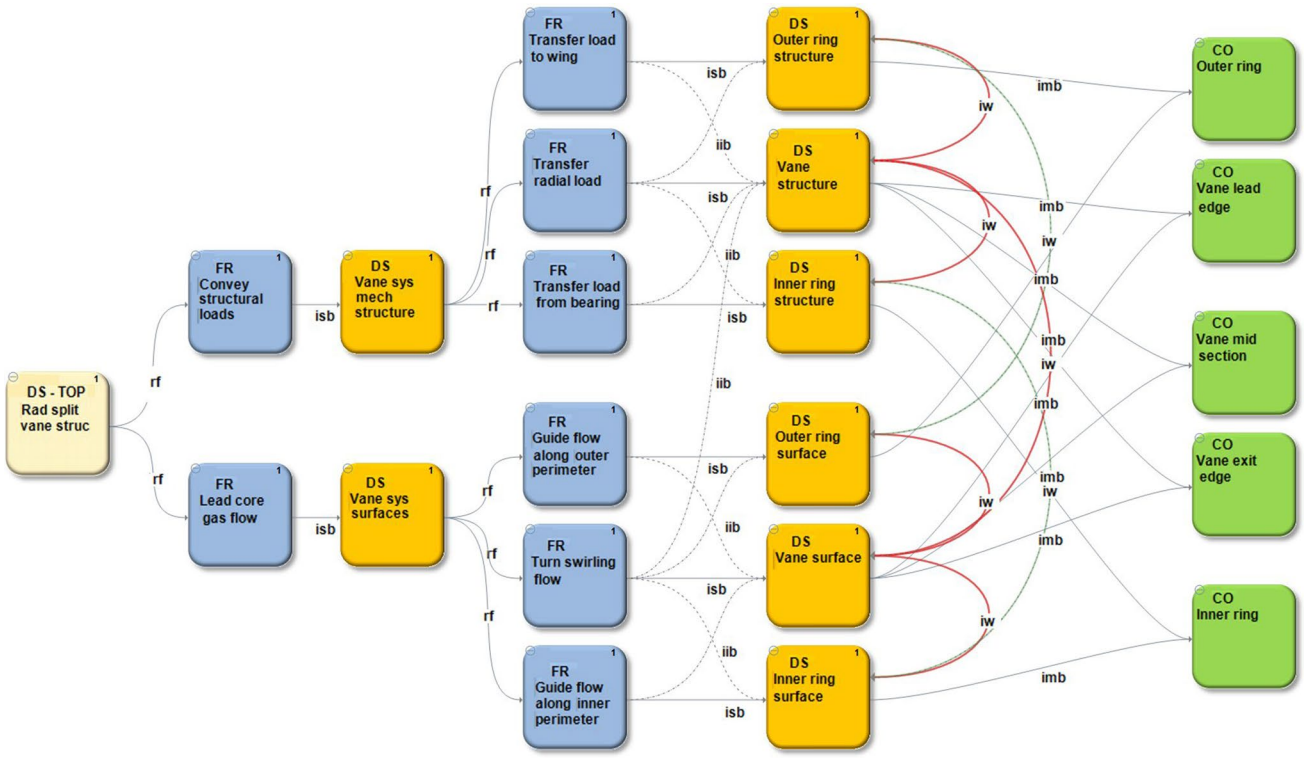

Fig. 28 EF-M tree for the radially split vane option. Sect. 4.2 shows the corresponding graphs (Fig. 24) with discussions

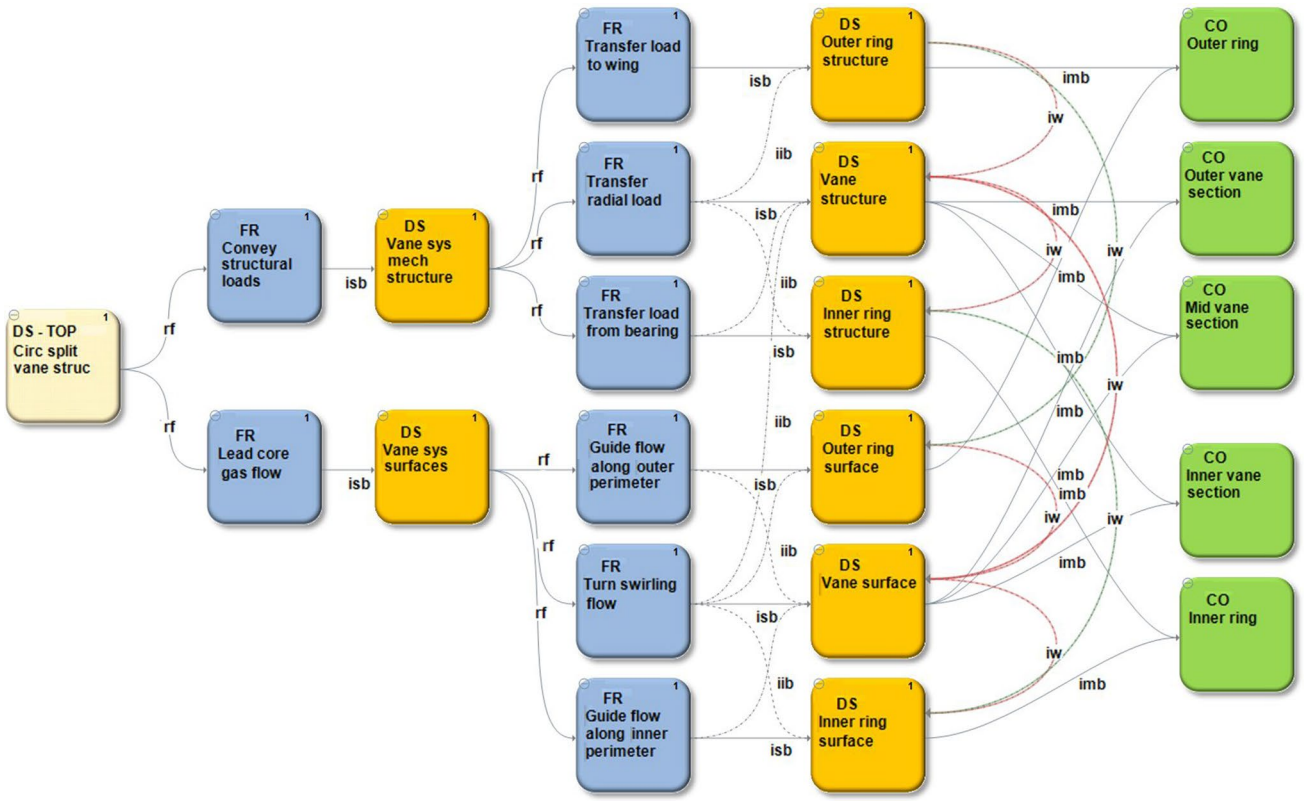

Fig. 29 EF-M tree for the circular split vane option. Sect. 4.2 shows the corresponding graphs (Fig. 25) with discussions

References

Andreasen M (1991) The theory of domains. In: Workshop on understanding function and function to form evolution, Cambridge University, UK

Bin S, Huang GQ, Mak KL (2002) Web-based product platform development using graph theory. In: Engineering Management Conference, 2002. IEMC '02. 2002 IEEE International, 2002, vol.1. 107–112

Chakrabarti A (2001) Sharing in design—categories, importance and issues. In: Proceedings of the international conference on engineering design (ICED01), design methods for performance and sustainability, 2001 Glasgow. pp 563–570

Chakrabarti A, Singh V (2007) A method for structure sharing to enhance resource effectiveness. J Eng Des 18:73–91

Chartrand G, Lesniak L (1996) Graphs & digraphs. 3rd edn. Chapman & Hall, Boca Raton

Christopher A (1964) Note on the synthesis of form

Claesson A (2006) A configurable component framework supporting platform-based product development. Dissertation/Thesis, Chalmers University of Technology, Göteborg

Cope Sweden AB (2014) Object oriented platform development: about (online). http://www.copesweden.se/About.html. Accessed 22 Feb 2017

Eppinger SD, Browning TR (2012) Engineering systems: design structure matrix methods and applications. MIT Press, Cambridge

Flowserve Inc HPX API 610 (OH2) Centerline mounted pump

Ghisu T, Parks GT, Jarrett JP, Clarkson PJ (2011) An integrated system for the aerodynamic design of compression systems—part I: development. J Turbomach 133:011011

GKN Aerospace (2014) Presentations (online). http://www.gkn.com/aerospace/media/resources/Presentations/GKN-Aerospace-Fan-and-Turbine-Engine-Structures.pdf. Accessed 19 April 2016

Goldrei D (1996) Classic set theory: a guided independent study. Chapman & Hall, London

Hitchins DK (2003) Advanced systems thinking, engineering, and management. Artech House, Norwood

Holtta-Otto K, De Weck O (2007) Degree of modularity in engineering systems and products with technical and business constraints. Concurr Eng 15:113–126

Hubka V (2013) Principles of engineering design. Elsevier, Burlington

Isaksson O, Lindroth P, Eckert C (2014) Optimisation of products versus optimisation of product platforms: an engineering change margin perspective. In: DS 77: proceedings of the DESIGN 2014, 13th international design conference

Iwankowicz RR (2016) An efficient evolutionary method of assembly sequence planning for shipbuilding industry. Assembly Autom 36:60–71

Jenab K, Liu D (2009) A graph-based model for manufacturing complexity. Int J Prod Res 48:3383–3392

Jiao J, Simpson TW, Siddique Z (2007) Product family design and platform-based product development: a state-of-the-art review. J Intell Manuf 18:5–29

Johannesson H, Claesson A (2005) Systematic product platform design: a combined function-means and parametric modeling approach. J Eng Des 16:25–43

Joshi S, Chang TC (1988) Graph-based heuristics for recognition of machined features from a 3D solid model. Comput Aided Des 20:58–66

Keller R, Eckert CM, Clarkson PJ (2006) Matrices or node-link diagrams: which visual representation is better for visualising connectivity models? Inform Vis 5:62–76

Luo J (2015) A simulation-based method to evaluate the impact of product architecture on product evolvability. Res Eng Design 26:355–371

Lyu N, Saitou K (2003) Decomposition-based assembly synthesis for structural stiffness. J Mech Des 125:452

Mathieson JL, Wallace BA, Summers JD (2013) Assembly time modelling through connective complexity metrics. Int J Comput Integr Manuf 26:955–967

Raja V, Isaksson O (2015) Generic functional decomposition of an integrated jet engine mechanical sub system using a configurable component approach. In: Richard Curran NW, Borsato M, Stjepandić J, Verhagen WJC, (eds), ISPE CE. IOS Press, Delft, pp 337–346

Robertson D, Ulrich K (1998) Planning for product platforms. Sloan Manag Rev 39,:19–31

Rocca GL (2012) Knowledge based engineering: between AI and CAD. Review of a language based technology to support engineering design. Adv Eng Inform 26:159–179

Sobek DK II, Ward AC, Liker JK (1999) Toyota's principles of set-based concurrent engineering. Sloan Manag Rev 40:67

Sosa ME, Eppinger SD, Rowles CM (2004) The misalignment of product architecture and organizational structure in complex product development. Manage Sci 50:1674–1689

Sosa ME, Eppinger SD, Rowles CM (2007) A network approach to define modularity of components in complex products. J Mech Des. 129: 1118–1129

Suh ES, De Weck OL, Chang D (2007) Flexible product platforms: framework and case study. Res Eng Design 18,:67–89

Ulrich K (1995) The role of product architecture in the manufacturing firm. Res Policy 24:419–440

Ulrich KT, Eppinger SD (1995) Product design and development (international editions 1995), McGraw-Hill, Inc, Singapore

Ulrich KT, Seering WP (1990) Function sharing in mechanical design. Des Stud 11:223–234

Whitney DE (1993) Nippondenso Co. Ltd: a case study of strategic product design. Res Eng Design 5:1–20

WIE MJV, Rajan P, Campbell MI, Wood KL (2003) Representing product architecture. In: International design engineering technical conferences and computers and information in engineering conference, September 2–6 2003 Chicago, Illinois, USA

Wyatt DF, Wynn DC, Jarrett JP, Clarkson PJ (2011) Supporting product architecture design using computational design synthesis with network structure constraints. Res Eng Design 23:17–52

Yassine AA, Wissmann LA (2007) The implications of product architecture on the firm. Syst Eng 10:118–137

You C-F, Tsai YL (2009) 3D solid model retrieval for engineering reuse based on local feature correspondence. Int J Adv Manuf Technol 46:649–661