Multi-attribute optimization-based system decomposition considering several value chain stakeholder perspectives

Eun Suk Suh¹ · Kaushik Sinha² · Jaemyung Ahn³

Received: 12 October 2019 / Revised: 10 June 2020 / Accepted: 22 June 2020 / Published online: 16 July 2020
© The Author(s) 2020

Abstract

The final architecture of a complex system reflect preferences of several value chain stakeholders on system attributes, also called “ilities”. Owing to differences in their individual roles and responsibilities, different stakeholders prefer different approaches to architect and decompose a system to optimize their attributes of interest. However, owing to increasing complexity of modern engineering systems, optimizing multiple attributes of complex systems has become challenging; moreover, very few researches have been published in this regard. Thus, to address this gap in available literature, this paper presents a multi-attribute optimization framework for complex system decomposition. The proposed framework primarily optimizes two attributes—system robustness (to the perspective of the stakeholder), and modularity—while system maintainability is considered an optimization constraint. Feasibility of the proposed framework has been demonstrated through a case study, wherein system attributes of three different mechanical clock models having different architectures were optimized.

Keywords Multi-attribute optimization · Robustness · Modularity · Maintainability

1 Introduction

System architecture selection and decomposition is one of the most important processes that determine the success or failure of a system. It is the key responsibility of the system architect to generate the most optimal system architecture and decompose the chosen architecture into a subsystem configuration that optimizes various system attributes. Additionally, this aspect considerably influences downstream design processes. Given the escalating demand for superior system performance, implementation of latest technologies, and increasing pressure to reduce development time as

well as lifecycle costs, the front-end system architecting has assumed critical importance.

Designing complex systems requires the participation of several value chain stakeholders who attempt to optimize particular system attributes, which are critical to the success of the system being designed. These attributes include system complexity, performance, modularity, and maintainability. Based on their responsibilities, the stakeholders involved seek to optimize different system attributes. For example, system engineers who are responsible for the overall system design, attempt to reduce the overall complexity of the system to reduce the resources required for the system development. Design engineers, who are responsible for the detailed design, strive to achieve the maximum system performance for their assigned subsystems. Production engineers seek to modularize the system to optimize the assembly sequence and overall productivity. Field engineers responsible for system maintenance, strive to optimize the maintainability as the prime objective for the final deliverable.

However, improving a particular system attribute often affects other system attributes, in a positive or negative manner. For example, a design engineer who seeks to generate a design that achieves the highest system performance might propose an integral system design. The choice of the integral system design might affect the assembly sequence developed

✉ Eun Suk Suh
essuh@snu.ac.kr

¹ Graduate School of Engineering Practice, Institute of Engineering Research, Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul 08826, Republic of Korea

² Sociotechnical Systems Research Center, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA

³ Department of Aerospace Engineering, Korea Advanced Institute of Science and Technology (KAIST), 291 Daehak-ro, Yuseong-gu, Daejeon 34141, Republic of Korea

by the production engineer, making it longer and more complex, resulting in a loss of productivity. Additionally, the integral system design choice influences field engineers because the maintenance time defined by them may increase owing to the system's integral nature. It is clear that attempts by individual stakeholders to optimize a particular system attributes may affect attributes relevant to other stakeholders. Consequently, a balance among system attributes must be realized to ensure that the system can be designed, launched, operated, and disposed in an optimum manner. If system attributes can be optimized upfront during the architecting and decomposition stages, the overall development effort and associated lifecycle costs can be reduced.

In this study, we propose a multiple system attribute optimization framework that can be implemented during the system architecting and decomposition phases. Feasibility of the proposed framework has been demonstrated by selecting three system attributes—robustness, system modularity, and maintainability. From the traditional engineering design viewpoint, robustness refers to the ability of a system to function normally in the event of uncertainties within the operating environment and system conditions (Phadke 1989). However, in this study, robustness pertains to that of system architecture decomposition considering different stakeholder perspectives (Ahn et al. 2018). Modularity pertains to the ability of a system to be decomposed into distinct subsystems with each subsystem being tightly integrated within itself but loosely connected to other subsystems (Baldwin and Clark 2000). Lastly, maintainability corresponds to a system's ability to be maintained throughout its lifecycle (Blanchard and Blyler 2016).

The objective of this study is to determine a Pareto optimal set of modular system architecture decompositions that are both robust with regard to different value chain stakeholder perspectives and easy to maintain throughout the system lifecycle. The optimization framework has been demonstrated through a case study, wherein system attributes pertaining to three clock architectures have been optimized. This work contributes to complex system design research by introducing an enabler to optimize and balance different system attributes early during the system architecting phase through various decompositions. Additionally, this research offers several optimized system architecture configurations to support subsequent system design, development, operation, and/or retirement.

Subsequent sections provide a review of existing works and describe the formulation of the proposed multiple system attribute optimization framework. Next, described is the case study performed as part of this research along with a discussion of results obtained. Finally, conclusions drawn and scope for future work have been elucidated.

2 Previous work

2.1 System architecture and attributes

The system design process is a set of activities that translate ambiguous customer needs into quantifiable functions and the final architecture that imbeds the identified functions. A typical system design process is a linear process with critical reviews scheduled at major milestones, such as the voice of customer analysis, system concept selection, detailed design, and system ramp-up (Otto and Wood 2001; Ulrich and Eppinger 2016). However, owing to the wide variety of system types, many system design processes are modified to suit the type of system being designed. Examples of such processes include the “Vee” process (Shortell 2015) used in aerospace/defense industries, and the spiral process (Boehm 1988) and Scrum process (Schwaber 2015) used in software industries.

The front-end design phase, also known as the system architecting phase, is one of the most important phases in the system design process. In this phase, different architectural concepts for the final system are generated and the key system functions are mapped to the appropriate segments of the generated system architectures. The final deliverable of the system architecting phase is the final definition of the system architecture involving a clear mapping of the key system functions to the final architecture subsystems. It is well known that the choice of the final system architecture determines the success or failure of a system. However, because many key information points pertaining to the system are unknown during this phase, the system architects need to model the system in an abstract yet informative manner to make informed decisions. To fulfill this requirement, system architecture modeling methods such as the system modeling language (Friedenthal et al. 2015), object process methodology (Dori 2002), and integration definition for function modeling (IDEF 1993) have been developed and used in the industry and academia.

The main purpose of the system architecting process is to conceive and deliver an architecture that performs required functions as well as optimize system attributes related to aforementioned functions. System attributes refer to inherent system properties that determine particular aspects of system lifecycle performance, and they are closely related to system functionality. Some of the key system attributes are the complexity, modularity, flexibility, robustness, reliability and maintainability (Allen et al. 2001). The system complexity is related to the number of system elements, and interconnecting relationships (Braha and Maimon 1998; Dolan and Lewis 2008; Maimon and Braha 1996; Malik 1984). There exist different types of

system complexities, such as the structural, dynamic, and organizational complexities (Sheard and Mostashari 2009). Modularity can be associated with systems wherein a particular function or specific amount of complexity can be allocated to a chunk or module of concerned systems. Such modules typically comprise several intra-connections along with a small number of inter-connections to other modules (Baldwin and Clark 2000). Systems can be decomposed into different types of modules such as function-based, assembly-based, or maintenance-based modules (Suh et al. 2015).

Flexibility refers to a system's ability to allow changes to its configuration and adapt to different operating scenarios (Engel et al. 2017; Engel and Reich 2015; Suh et al. 2007). It is often recognized as an active approach to respond to uncertainty. Robustness refers to the ability of a system to perform a required function under changing operating conditions (Phadke 1989). A robust system is designed to perform consistently under uncertain operating conditions without making changes to the system and it is a passive way to address the uncertainty. This is a system attribute central to popular design methodologies, such as six sigma and Taguchi methods. Reliability refers to the degree of system operating consistency (Carmines and Zeller 1979). It is an especially important attribute for a mission critical system, such as production equipment or power grid equipment, whose downtime can result in a significant loss of productivity and shutdown. Maintainability is measure of the degree of field maintenance of a system. Ease of maintenance results in a reduced overall operation cost owing to the reduction in the maintenance time (Blanchard and Blyler 2016). In addition to the abovementioned system attributes, others, such as safety, usability, sustainability, availability, commonality, and affordability exist, and these attributes must be carefully considered and balanced during the system architecting stage.

2.2 System design and optimization

Designing complex system architectures is an arduous task that requires significant resources and participation of several stakeholders. Depending on the system's nature, there exist several design methodologies, customized to particular system types. Despite several methodologies being available for designing different complex system types, a common task explicitly or implicitly included in most methodologies is system decomposition.

System decomposition is an activity that divides the system into several coherent subsystems. Crawley et al. (2016) reported system decomposition as one of the most important tasks assigned to system architects. System decomposition significantly affects the overall lifecycle performance of a system. Typically, systems are decomposed to improve

certain attributes. For example, a system can be decomposed for balanced complexity allocation among subsystems (Sinha and Suh 2018). Additionally, systems can be decomposed to have each subsystem perform a specific function. Alternately, systems can also be decomposed as being assembly friendly (Baldwin and Clark 2000). Another means to decompose a system involves divide it into several subsystems such that it can be flexible to accommodate future modifications in system requirements via addition, replacement, or removal of relevant subsystems (Engel et al. 2017; Engel and Reich 2015; Suh et al. 2007).

Owing to system decomposition activity taking place during the early system architecting stage, system modeling must be performed in an abstract manner with limited information availability. Additionally, the system model must be easy to facilitate system decomposition. A popular system model widely used in academia and industry alike for this purpose is the design structure matrix (DSM), which is a matrix representation of a system (Eppinger and Browning 2012). Since its formal introduction by Steward (1981), DSM has been used to model systems for decomposition and integration problem solving purposes (Browning 2001). It has been widely used to model products (Kasperek et al. 2015; Suh et al. 2015; Tilstra et al. 2012) such as a jet engine (Sosa et al. 2003), an industrial printing system (Suh et al. 2010), a robotic spacecraft (Brady 2002), a helicopter (Eckert et al. 2004), a train bogie (Sinha and Suh 2018), and an electric train static inverter system (Sinha et al. 2020). It is also used to model organizations (Coates et al. 2000; Moon et al. 2015; Sage and Rouse 2009) and processes (Browning 2014; Sacks et al. 2004; Sharon et al. 2011).

A key objective of system decomposition is to divide systems into several subsystems or “modules” to improve system attributes. For example, system decomposition to improve product manufacturing and assembly is focused on improving system modularity (Whitney 2004). Decomposing a system into several modules with redundant features aims to improve system availability (Suh and Kott 2010). Product platform design methodologies focus on decomposing systems into common modules used for all products in the product family as well as variant modules to be used for individual products. This, in turn, enhances system commonality (Meyer and Lehnerd 1997). Additionally, decomposition and embedment of real options in modules to respond to future uncertainties are performed to enhance system flexibility (Suh et al. 2007; Trigeorgis 1996). Several studies in the field of modular system design have been published to this end. Researches concerning modular system design can be classified into modularity metric development and modular system clustering algorithms.

The modularity metric measures the extent to which a particular system decomposition can be modularized. Several metrics have been proposed to this end, and reviews

of past metrics have been reported in several extant studies (Guo and Gershenson 2003; Guo and Gershenson 2004; Holtta-Otto et al. 2012; Van Eikema Hommes 2008). According to Holtta-Otto et al. (2012), there exist two modularity-metric types. The first type measures the degree of coupling between modules. These metrics account for connections between elements as well as identify strong and loose couplings. Metrics categorized in the first type include those developed by Martin and Ishii (2002), Holtta-Otto and de Weck (2007), Guo and Gershenson (2003), and Jung and Simpson (2017) to mention just a few. Metrics of the second type help identify similarities between individual modules by considering commonalities in manufacturing processes, materials, and ease of reuse. Metrics following this approach include those proposed by Newcomb et al. (1998), Gershenson et al. (1999), Siddique et al. (1998), and Mattson and Magleby (2001).

Decomposition of a system into modules requires following a particular set of rules incorporated within clustering algorithms. Several clustering algorithms have been proposed to modularize complex system. Newman and Girvan (2004) proposed a clustering algorithm based on edge removal, whereas the algorithm proposed by Blondel et al. (2008) follows a two-stage approach to divide systems into several clusters. Li et al. (2017) proposed an algorithm utilizing spectral decomposition. The clustering algorithm proposed by Borjesson and Holtta-Otto (2014) was based on modular-function deployment. Another widely used algorithm, called the Idicula–Gutierrez–Thebeau algorithm (IGTA) (Thebeau 2001) generates modular system using cluster coordination costs. For additional references on clustering algorithms, please refer to Herrmann et al. (2018).

Using above-mentioned modularity metrics and clustering algorithms, several studies concerning modular-system design and optimization have been published. Recent works in this regard include modularization of magnetic resonance imaging scanners (Sanaei et al. 2016) using the IGTA+ algorithm, modularization of an electric train static inverter system considering several design constraints, including the maintenance constraint (Sinha et al. 2020), and modularization of a train bogie system (Sinha and Suh 2018) using the metric proposed by Newman (2018).

2.3 Research opportunity

Although several studies have been published concerning system design methodologies to optimize system attributes, there remain unexplored research topics that can potentially contribute towards the advancement of complex system design. Recently, researchers investigating system architecture design and optimization started to explore techniques to optimize and balance multiple system attributes. The driving factor in such researches is that although there exist important system

attributes that affect system lifecycle performance, they need to be prioritized and balanced depending on the system's nature and use context. For instance, systems that require future function updates must be designed with function-based modularity as the top priority system attribute. Meanwhile systems that require highly reliable lifecycle performances must to be designed with reliability as the top system attribute. Similarly, systems that need to be available and operational when required by customers and/or operators must be designed with the availability as the key system attribute. To reduce the amount of effort involved in system design and development, reduction in system complexity is a key priority owing to its correlation with the effort required for system design (Kim et al. 2017).

Baylis et al. (2018) proposed a Pareto-optimization approach to optimize the commonality and modularity of product family platforms. In addition, Sinha and Suh (2018) introduced an optimization framework to optimize the system modularity and allocation of system complexity to system modules, which was demonstrated by performing a case study of a train bogie system. Sinha et al. (2020) optimized the modularity of an electric train's static inverter system using two different clustering algorithms, whilst enforcing several design constraints, such as system component's maintenance interval, system component's layout, and allocation of heat generating components in different modules. Such frameworks represent an important advancement in system design research as multiple system attribute optimization can be used to obtain a set of system architecture configurations that can fulfill the needs of several value chain stakeholders.

As stated previously, the value chain stakeholders participating in system design have different priorities and preferences regarding various system attributes, owing to their varied roles and responsibilities in the value chain, and it has been demonstrated that the preferred decomposition configuration of a system can be vary depending on the priorities of individual stakeholders (Suh et al. 2015). To address this aspect and optimize the system design, the research presented in this paper introduces a multiple system attributes optimization framework that can accommodate the preferences of different value chain stakeholders. In the subsequent sections, the proposed multiple system attribute optimization framework is described in detail. Definitions of target system attributes, such as robustness (to the perspective of the stakeholder), modularity, and maintainability, are presented, and appropriate metrics for robustness and modularity have been introduced.

3 Formulation of optimization framework

3.1 Definition of system attributes used in proposed optimization framework

The objective of this study is to establish an optimization framework that can generate several system architecture configurations that are optimized for robustness and modularity and subject to maintenance constraints. However, it is important to define each system attribute in terms of the optimization framework context considered in this work.

Robustness, in this study, pertains to that of system architecture decomposition based on different stakeholder perspectives. During the early stages of system design, once the system architecture concept has been decided, the system needs to be decomposed into several subsystems and allocated to different engineering teams for detailed design. Typically, the system is decomposed into function-based subsystems, reflecting the perspective of the design engineers who are responsible for particular system functions. However, this decomposition approach may not be suitable for other value chain stakeholders, such as production engineers, and field service engineers. Production engineers prefer a system decomposition based on the physical interconnections between the components to realize a reduced assembly time, whereas field service engineers aim to group the frequently serviced components into same subsystems to realize ease of maintenance. Such regrouping or re-decomposition may lead to a prolonged design and development period. This situation of a system being decomposed differently to suit the needs of individual stakeholders creates inefficiencies in the system design process.

Consequently, the system design process can be enhanced if a set of system decomposition configurations that change minimally from stakeholder to stakeholder can be generated. For example, if a particular function-based system decomposition configuration involving minimal alterations to the composition of the components in the subsystems can be accepted by the production engineers and field service engineers, the time to rearrange and re-decompose the system will be reduced. In this context, the system decomposition configuration is considered robust to the perspective of different stakeholders. This definition of the system robustness is used throughout this study.

Modularity is defined as the property of a system whereby the system is decomposed into several modules with each module being easily assembled or removed from the system owing to its minimal interface with the rest of the system. This is a desirable attribute for various stakeholders in the value chain, albeit in different contexts. For instance, design engineers prefer function-based modules,

in which a specific module performs a particular function for the entire system. Production engineers prefer assembly-based modules for ease of assembly, resulting in an increase in the productivity. Field service engineers prefer maintenance-based modules with components that require frequent service and replacement grouped together for easy disassembly and replacement.

Maintainability pertains to the degree of system maintenance. If the system can be maintained through easy repair and replacement of the consumable and spare components, the degree of maintainability is high. However, if the system is difficult to maintain, requiring a high level of labor and other resources, the system maintainability is low. The maintainability is an important attribute for systems with long lifecycles, in which the post-launch operating and maintenance costs take up the majority of the total system lifecycle cost.

In other words, in this study, the robustness is the degree of changes in the system decomposition configuration from function-based decomposition to other decompositions such as assembly-based and maintenance-based decompositions. The function-based decomposition was chosen as the reference decomposition configuration as function-based system design usually occurs early in the system design process. The modularity of a system is set based on the perspective of production engineers. Here, each module comprises complex intra-connections but minimal inter-connections to other modules, thereby making the system easy to assemble. The system maintainability is built into the optimization framework as a constraint by forcing a set of serviceable components into the same maintenance-specific modules.

3.2 Metrics of system attributes

Quantitatively assessing the system attributes for a system decomposition configuration requires measurable metrics for the corresponding system attributes. In this study, two metrics, one for robustness and the other for modularity were used to optimize the system architecture decomposition. These metrics are referred to from previously published literature.

The metric for robustness is used to measure the difference between the original system decomposition configuration and the configuration changed to suit the need of another stakeholder. A system decomposition configuration is more “robust” to the perspective of different stakeholders if the difference between these two configurations is small. This metric is known as the module diffusion index (MDI). A condensed explanation of the metric is presented in this study. For the detailed derivation, readers are referred to the paper by Ahn et al. (2018). The MDI can be defined as

$$\text{MDI}_i^{AB} = e^{S_i^{AB}}. \quad (1)$$

In particular, when the individual elements of the i th module configured for perspective A are diffused to a number of modules for perspective B , the MDI value is the exponential function of the module diffusion entropy S_i^{AB} . The module diffusion entropy S is based on statistical mechanics, in which the entropy is often interpreted as the degree of uncertainty of the microscopic state of a system (Schwabl 2006). Consequently, S_i^{AB} can be stated as

$$S_i^{AB} = \sum_{k \in T_i} (-p_{i,k} * \ln(p_{i,k})) \quad (2)$$

where T_i represents the index set of modules included in perspective B that include at least one component from module i in perspective A , and $p_{i,k}$ represents the fraction of components in the i th module for perspective A that are contained in the k th module for perspective B . Figure 1 shows the examples of module diffusion from perspective A to B with the corresponding S and MDI_{*i*} values.

As shown in the figure, the original module in perspective A has ten components. When the module is re-decomposed to satisfy the needs from the perspective of stakeholder B , the components in the original module can be diffused differently. Example (a) shows that the modular configuration of perspective A is identical to that of perspective B , thereby resulting in an S_i value of zero and MDI_{*i*} value of one. Examples (b) through (d) demonstrate the variation in S_i and MDI_{*i*} values as the components in the original module

for perspective A are diffused into different modules in perspective B .

The examples shown in Fig. 1 correspond to a single module; however, most complex systems consist of several modules, and the total MDI value for the system must be calculated. For a system composed of m modules, the total MDI value can be calculated as follows:

$$\text{MDI}_{\text{tot}}^{AB} = \sum_{i=1}^m \left(\frac{c_i}{N} \cdot \text{MDI}_i^{AB} \right). \quad (3)$$

In the equation, c_i represents the number of components in the i th module of perspective B that were present in the original module in perspective A , and $N (= \sum_{i=1}^m c_i)$ represents the total number of components in the original module for perspective A . Using the MDI, system architects can measure the amount of diffusion of the original modular configuration from one perspective into the modular configurations of other perspectives. A low total MDI value for one perspective with respect to another indicates small resulting difference in system architecture decomposition between two perspectives, thereby indicating the particular decomposition to be robust to stakeholder perspectives. The opposite is true when a high total MDI value is observed.

In terms of the modularity, several modularity metrics have been developed owing to the industrial and academic interest in system modularity. These metrics can be categorized into metrics that measure the degree of coupling between modules and those that measure the similarities between modules (Holtta-Otto et al. 2012). For this study,

Fig. 1 Examples of module diffusion from perspective A to B with corresponding S and MDI values (reproduced with permission from Ahn et al. (2018))

the modularity metric developed by Newman (2018), which is based on the classical graph theory and extensively used in network science is used to measure the degree of system modularity. The formal equation for the modularity metric Q is as follows:

$$Q = \sum_{i=1}^k (e_{ii} - a_i^2) = \text{Tr}(e) - ||ee^T|| \quad (4)$$

where

$$e = \begin{bmatrix} y_{11} & y_{12}/2 & \dots & \dots & y_{1k}/2 \\ y_{21}/2 & y_{22} & \dots & \dots & y_{2k}/2 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ y_{k1}/2 & \dots & \dots & \dots & y_{kk} \end{bmatrix}$$

and

$$a_i = \sum_{j=1}^k e_{ij} = y_{ii} + \left(\sum_{j=1}^k y_{ij} \right) / 2. \quad (5)$$

To compute Q , the system needs to be modeled in a matrix format, as shown above in the form of matrix e . Q essentially represents the sum of the diagonal entries of matrix e or the trace of matrix e , minus the Frobenius norm of the product term between e and its transpose (Horn and Johnson 1991).

The last system attribute considered in this study is maintainability. For the optimization, the maintainability is treated as a constraint. Typically, during the system design process, the components requiring maintenance during the system's lifecycle are determined by the design and field service engineers. The system is then designed to facilitate the easy disassembly and replacement of selected components.

Vehicle oil filters, batteries, and printing system toners are some examples of regular-maintenance components. The constraint imposed in this optimization framework is to force these maintenance components into modules by themselves, and then optimize the entire system architecture decomposition configuration for the robustness and modularity.

3.3 System model

Once the definitions and metrics of the system attributes have been defined, the next step is to generate a model of the system that can be used for the system attribute optimization. In the system design and development process, the system architecture decomposition and system attribute optimization need to be completed in the early stage, at a time when the detailed design for the system is not yet available or is uncertain at best. Such a situation necessitates a system model that can be constructed with minimal information regarding the system. For such cases, connection-based abstract system models, such as network models or matrix-based models, are widely used. These system models typically require only two information points: a list of the system components and their connective relationships. Because this information can be obtained with relative ease during the early stages of the system design process, the system architects can construct the system model with relative ease and high accuracy. Figure 2 shows a hypothetical system with ten components modeled as both a network and DSM.

In the DSM, the connective relationship shown in the network diagram is represented by the numerical entry “1” in the off-diagonal cells, which represents the existence of the connections between components. For bi-directional connections such as physical connections, the entry is symmetric with respect to the diagonal cells. The empty cells are typically filled with entry “0” for quantitative matrix-based

Fig. 2 Hypothetical system with ten components, represented as a network and DSM

analyses. Because the proposed optimization framework uses system attributes metrics that can be easily calculated using the DSM system models, it was employed as the system model for this study.

3.4 Formulation of optimization statement

Based on established metrics and constraints described in preceding sections, an optimization statement can be formulated as follows. Optimization aims at determining system configuration (decision variable) that is robust to different stakeholder perspectives (minimum MDI) as well as modular (maximum Q), whilst being subject to engineering constraints. Examples of typical constraints include grouping of system elements into specific modules for maintenance convenience specifying a total number of modules (N_F) in the final optimized configuration. The multi-objective optimization problem involves determining an optimum system decomposition $Gr(.)$ that minimizes MDI, whilst maximizing system modularity Q . Thus, a typical multi-objective optimization problem can be expressed as:

$$\text{Minimize: } f_M(Gr(.)) \equiv [MDI, -Q] \Bigg\} \\ \text{s.t. } g_i(Gr(.)) \leq 0$$

The constraint set defined by $g_i(Gr(.)) \leq 0$ in the equation may include a target number of components along with other manufacturing constraints. This grouping constraint is considered a hard constraint using a penalty-function-based methodology. This method follows a modification introduced by Deb (2000) to circumvent the need for a penalty parameter by defining the following objective function (minimization):

$$\hat{f}_j(gr) = \begin{cases} f_j(gr), j \in \{1, 2\}, & \text{if } A \text{ is feasible} \\ (f_j)_{\max} + \sum_{i=1}^M |h_i(gr)|, j \in \{1, 2\}, & \text{otherwise} \end{cases}$$

where $(f_j)_{\max}$ represents the worst *feasible* solution for the j th objective and $h_i(.)$ denotes the constraint function value, usually measured in terms of the normalized number of constraint violations. This normalization can be performed by either counting a bound on the maximum number of violations possible or using a large number as the denominator. The second approach is much simpler and has been deemed adequate for the purpose of this research. During multi-objective optimization, objective is augmented using the constraint handling approach. The proposition is that any feasible solution is preferred over a non-feasible one. Non-feasible solutions were compared based on their degree of constraint violation. The method of enforcing hard constraints is always based on a bias for feasible over

non-feasible solutions through use of a certain penalty function strategy. There may exist circumstances wherein additional context-specific constraints are imposed to arrive at feasible system decompositions. An equality constraint can be applied to the target number of modules (N_F).

The proposed implementation of multi-objective optimization (i.e., minimization) is based on the concept of domination. A solution Gr_i is said to dominate Gr_j if $\forall M \in 1, 2, \dots, k, f_M(Gr_i) \leq f_M(Gr_j)$, such that $f_M(Gr_i) < f_M(Gr_j)$. Now, from the solution set $\mathbf{P}$, a Pareto optimal solution set $\mathbf{P}_0$ contains elements that are not dominated. This Pareto optimal set refers to a non-dominated set in the entire search space S . A multi-objective optimization algorithm yields a set of solutions not dominated by any solution encountered by it. In this study, we have developed a combinatorial variant of the AMOSA algorithm (Bandyopadhyay et al. 2008) with adaptive cooling schedule to improve its efficiency (Ingber et al. 2012). The next section describes the case study demonstrating the utility of the proposed optimization framework.

4 Case study for clock architecture decomposition

4.1 Overview

Feasibility of the proposed optimization framework was demonstrated through its implementation to actual clock models. The main purpose of a clock is accurate time measurement, and several system architectures have been developed to achieve this purpose. In this study, three clock models, consisting of 40–50 individual components, with different system architectures were selected. The first clock is the verge and foliot clock (VFC), which uses the verge escapement mechanism to power the foliot to measure the time. The second clock is the flying pendulum clock (FPC), which utilizes a flying pendulum escapement mechanism to track the time. The third clock is the Congreve style clock (CSC), which uses a ball rolling down the oscillating path to power the timekeeping mechanism. Figures 3, 4 and 5 show pictures of VFC, FPC, and CSC models, respectively, and the complete function-based decomposition DSM for each clock. These clock models were selected for numerous reasons. First, these clock models have different escapement architectures for the time keeping function, which allows a comparative analysis of the optimization results for different system architectures. Second, the chosen clock models can be decomposed to equal levels of granularity with a comparable number of individual components, thereby allowing a reasonable comparison of the optimization results. Typically, system architects decompose a system down to two levels, dividing the system into 7 ± 2 elements at each

Fig. 3 Verge and foliot clock (VFC) and corresponding function-based decomposition DSM

Verge and Foliot Clock (VFC)

Fig. 4 Flying pendulum clock (FPC) and corresponding function-based decomposition DSM

Flying Pendulum Clock (FPC)

Fig. 5 Congreve style clock (CSC) and corresponding function-based decomposition DSM

Congreve Style Clock (CSC)

level, resulting in 25–81 elements (Crawley et al. 2016). The decomposed system elements and their interconnecting relationships can be transformed into a DSM model, which in turn, can be optimized for the system attributes. If the said

optimization can be performed for proposed clock models, complex systems with equivalent granularity and elements count can be optimized as well, demonstrating the generalization possibility of the framework.

Clock DSMs were constructed using assembly instructions and inspecting final physical assemblies. DSM size corresponds to the number of components used in each clock; 49×49 for VFC, 45×45 for FPC, and 41×41 for CSC. Post construction, all DSMs were manually clustered into six functional modules. This was performed to reflect the perspective of the system design team, responsible for delivering modules that perform specific clock functions. These function-based modular DSMs were used as reference perspectives for calculating MDI values during the optimization process.

Six functional modules were defined, namely, the platform module, structure module, potential energy supply module, power transfer module, power control module, and user interface module. The platform module is a set of components that support the clock assembly. The structure module is a set of components that hold the moving components in place. The potential energy supply module is a set of components that provide energy for the clock motion by providing weight at a high elevation. The power transfer module is a set of gear components that transform the potential energy provided by the potential energy supply module into rotational motion while distributing power throughout the clock. The power control module is a set of components that regulate the timekeeping motion. Finally, the user interface module is a set of components that display the time information to the users.

4.2 System attributes in context of optimization

Once the DSMs for the clock models have been achieved and their function-based modular decomposition has been completed, the next task is to define the context of the optimization. The context of the system attributes (robustness, modularity, and maintainability) for this particular optimization can be explained as follows:

Robustness The function-based modular decomposition for the clock models is shown in Figs. 3, 4 and 5 and represents the perspective of the system design team, who typically divides and designs the system based on the required functions. When the system components are re-clustered to improve the physical modularity of the system, the resulting modular configuration represents the perspective of the production engineering team, whose primary objective is to improve the productivity of the assembly line by reducing the production time through system modularization. The MDI indicates the degree of diffusion from the original function-based modular configuration to the re-clustered configuration. A low MDI value indicates that the re-clustered modular configuration has a small number of components diffused from the original modular configuration and is more robust than the re-clustered modular configuration with a higher MDI value.

Modularity The system modularity, quantified by the modularity metric Q , is of considerable interest to the production engineering team. If the system can be clustered into a set of physical modules that are easy to integrate in the final assembly line, the overall productivity can be significantly improved. However, in many instances, the physical modules sought by the production engineering team and the functional modules designed by the system design team are different, requiring an extensive rearrangement of the system components for the final assembly line. This presents a challenge for the production engineering team, who must perform the testing for the individual functions embedded in the function-based modules. Consequently, an optimal assembly-based modular configuration (Max Q) that is similar to the original function-based modular configuration designed by the system design team (Min MDI) must be determined throughout the design space by the optimization. In the case study, the original function-based modular configurations of the three clock models were re-clustered to determine the modular configuration with the optimum MDI and Q values.

Maintainability The clock models comprise several components. For this case study, these clock components were categorized into components requiring maintenance and those that do not need maintenance throughout the lifecycle of the clock. Moving components subject to frequent service and replacement, such as gears and shafts, were designated as maintenance-requiring components. Other components, such as structural components that may not need any maintenance or replacement, were designated as non-maintenance components. When the maintenance constraint was imposed during the optimization, the maintenance-requiring components clustered into the same module.

4.3 Optimization scenarios

After establishing the optimization problem, the next task is to generate several optimization scenarios for analyzing the trend of the optimization results. For the case study, two factors were used to generate the relevant optimization scenarios: enforcement of the maintenance constraint and number of modules in the final modular configuration.

Maintenance constraint enforcement Several components in the clock may undergo wear and may need to be serviced and replaced. Figure 6 shows the clock models with the main gears and shaft components highlighted as the maintenance components. The corresponding entries in the clock DSMs are clustered and grouped into a single maintenance module that can be serviced or replaced as a unit to incorporate the maintenance-based perspective of field service engineers, needing to repair and/or replace wearable components promptly and efficiently.

Number of modules in the final modular configuration (N_F) For this case study, one additional constraint was

Fig. 6 Clock models with clustered maintenance components shown in corresponding DSMs

imposed to generate the optimization scenarios. During the optimization run, the number of modules allowed in the final clock configuration (N_F) was set as an equality constraint. For each clock model, N_F was initially set as five to yield the optimal solution for clocks with a five module configuration. N_F was then increased to six to yield the optimal solution for the given constraint. The optimization run was repeated with N_F increasing up to 15 to analyze the trend of the optimized MDI and Q values with the variation in N_F . For each clock model, 22 optimization scenarios were generated and analyzed.

The actual optimization algorithm, as mentioned in a previous section, is based on a simulated annealing algorithm with an adoptive cooling schedule. A heuristic algorithm was used because the DSM-based clustering optimization is discrete in nature. During each optimization run, once a feasible modular configuration is obtained, the MDI is calculated by comparing the newly generated DSM to the original DSM shown in Figs. 3, 4 and 5. The Q value for the new DSM is calculated using Eq. (4). The iteration continues until the best modular configuration is obtained.

4.4 Optimization results and discussion

In this study, optimizations were performed for generated scenarios. Figures 7, 8 and 9 depict optimization solutions for VFC, FPC and CSC decomposition under corresponding scenarios without maintenance constraints, both non-dominated and dominated. In all figures, solutions are

categorized based on the number of modules contained in final decomposition configurations. Also, the direction of the utopia point ($MDI = 1.0$, $Q = 1.0$) and Pareto fronts for all three clock architectures have been indicated. Figure 10 shows an example DSM representation of the VFC in the function-based decomposition and Pareto optimal decomposition, marked as “* Pareto-Optimal Decomposition” in Fig. 7. For the Pareto optimal decomposition, number of modules, Q value, and MDI value are listed.

Several interesting trend can be observed in optimization results obtained for the three clock architectures. First, MDI increases with increase in N_F owing to spreading out of elements in the original module to a large number of modules. In all three cases, solutions obtained for optimization for a specific N_F are clustered together, and solutions with small and large N_F values are characterized by low and high MDI values, respectively. This is expected, owing to the nature of the MDI metric. However, the Q trend exhibit a more interesting behavior. As the figure depict, the Q value initially increases, maintains a certain level of modularity, and subsequently decreases with further increase in N_F value and commencement of the number of inter-modular connections outweighing intra-modular connections. Plots depicted in figures indicate existence of a certain region wherein the degree of modularity is maximized, thereby resulting in reduced N_F sensitivity.

Additionally, there exist notable differences between different clock architectures. The VFC exhibits gradual increase in modularity, followed by a “plateau” region, and

Fig. 7 Optimization results for VFC decomposition (without maintenance constraint)

Fig. 8 Optimization results for FPC decomposition (without maintenance constraint)

subsequent gradual decline in modularity with increase in N_F . For the FPC, modularity increases to a certain maximum value prior to a subsequent rapid decline. The CSC demonstrates a trend similar to that of FPC, albeit with a more gradual modularity decline. This implies that plots depicted in above figures serve as guidelines for establishing the threshold point for Q vs. MDI tradeoff before the overall system modularity rapidly declines.

Solutions that compose the Pareto front for each clock architecture comprise 5–9 modules. However, solutions

corresponding to large N_F values, albeit not part of the Pareto front, are also useful. In certain system design and development scenarios, instances requiring the system to be decomposed into a certain number of modules may be encountered owing to constraints imposed by the functional organization, production lines, field service organization, and/or module suppliers. Optimization solutions obtained by imposing a hard constraint on the number of modules can be examined to suit such situations.

Fig. 9 Optimization results for CSC decomposition (without maintenance constraint)

Original Function-Based Decomposition

Number of Modules: 6

Q Value: 0.36

*** Pareto-Optimal Decomposition**

Number of Modules: 7

Q Value: 0.59

MDI Value: 2.84

Fig. 10 DSM representation of the VFC in the original function-based decomposition and Pareto optimal decomposition (marked “* Pareto-Optimal Decomposition” in Fig. 7)

Figures 11, 12 and 13 depict optimization solutions for the three above-mentioned clock architectures with maintenance constraint imposed. Additionally, the direction of the utopia point ($MDI=1.0$, $Q=1.0$) as well as Pareto fronts and Pareto front plots for the said clock architectures without the maintenance constraint, have been illustrated for comparison.

The results for the optimization with the imposed maintenance constraint demonstrated a trend similar to the optimization results without the maintenance constraint. The MDI values for the equivalent scenarios with and without the maintenance constraint did not show significant differences. However, notable differences existed between the Q values of the optimization scenarios with and without constraints. In the former case, these values were reduced by 40–50%

Fig. 11 Optimization results for VFC decomposition (with maintenance constraint)

Fig. 12 Optimization results for FPC decomposition (with maintenance constraint)

because of the forced grouping of the maintainable components into a single module, thereby resulting in feasible design space reduction, which in turn, eliminated possible solutions that may have had better modular configurations compared to selected configurations. Furthermore, for all the three clock models, the number of modules comprising optimum configurations for both system attributes nearly equaled that in the original decomposition.

A key issue for system architects involves selecting a suitable system decomposition configuration from acquired Pareto optimal solutions. By observing the Pareto front profile, system architects can perform a trade-off between configurations having a low degree of diffusion (low MDI) combined with a relatively low degree of modularity (low Q) and those having an increased level of diffusion (high MDI) with increased level of modularity (high Q). This aspect requires performing additional supplementary analyses to determine

Fig. 13 Optimization results for CSC decomposition (with maintenance constraint)

the degree of influence of each configuration on the remaining value chain stakeholders. System decomposition with a high degree of modularity (high Q) may benefit production engineers, field service engineers, and supply chain partners to facilitate their managing of assembly, maintenance, and inventory managing processes with greater ease. However, system decomposition with a low degree of diffusion (low MDI) may benefit system architects and the design engineering team to streamline module design with downstream value chain partners.

An additional investigation of Pareto optimal solutions placed in the “plateau” region of the Pareto front may prove beneficial. Solutions in this region demonstrate a high degree of modularity and are relatively insensitive to the number of modules to which a system can be decomposed. System architects can decide on the degree of module diffusion based on other value chain analyses to select the best system decomposition configuration for their respective needs.

4.5 Summary of case study

The proposed optimization framework was successfully demonstrated using three clock models with different architectures. The modular decompositions that optimized the system robustness and modularity were explored using a simulated-annealing-based optimization algorithm. For each clock model, 22 optimization scenarios were generated, using the number of modules in the final configuration and the imposition of the maintenance constraint as hard constraints. The results yielded sets of Pareto optimal

modular configurations that were robust and modular to the limitations. The implications and preliminary guidelines for proper system decomposition configuration selection were presented to aid system architects and decision makers.

5 Conclusion and future scope

To design complex engineered systems, generating, assessing, and selecting the final system architecture is a significant challenge. It is well known that the system decomposition can have significant impact on the several system level attributes. Because there exist several stakeholders with different perspectives on what system attributes to optimize and how the system needs to be decomposed to optimize their preferred attributes, determining a system architecture decomposition configuration that can be accepted by these different stakeholders requires a more rigorous approach.

This paper proposes a multiple system attribute optimization framework for complex systems, with emphasis on optimizing the system robustness and modularity with a maintainability constraint. The objective was to find a modular decomposition that exhibits minimum variation when different stakeholders re-cluster and decompose the system to optimize their preferred system attributes (robustness), while simultaneously achieving the maximum modularity for efficient system integration. The system maintainability was treated as a constraint, and the system elements with similar maintenance properties were grouped into the same module.

The proposed optimization framework was demonstrated by optimizing the modular decomposition of three clock models

having different architectures. Using the simulated-annealing-based optimization algorithm, the optimized modular configuration for each optimization scenario was generated. The results indicated that there exist regions or “sweet spots” in which both the system attributes were optimized. It was also noted that although imposing the maintenance constraint did not impact the overall robustness of the final modular configuration, it had a severe negative impact on the modularity.

Implementation of the proposed optimization framework was successfully demonstrated through a case study. The proposed framework can complement several system architecture analysis types and selection methods already in place. Results obtained in this study present opportunities for several future endeavors. In this study, the three system attributes were either optimized or set as constraints, and results obtained can be used as a basis for expanding the optimization framework, thereby encompassing more system attributes. In addition, the impact of assigning different weights to different system attributes can provide valuable information to system architects. Another important research topic may involve comparison of optimization results obtained using different module clustering algorithms. Although this requires significant work and is beyond the scope of this study, it will provide valuable information regarding use of appropriate module clustering algorithms under different situations. Finally, linking the current optimization framework with cost models, such as development, unit (for mass-produced products), and maintenance cost models are expected to add significant value to systems architects and decision makers alike. This can help realize the overall lifecycle cost analysis and optimization, which is of considerable interest to various decision makers within system design, development, sales, and maintenance organizations. These aspects can serve as excellent bases for research in the domain of system architecture and design, thereby contributing to its advancement.

Acknowledgements This research was supported by the Republic of Korea Science, Technology, and Research (RoKST&R) program funded by the Lockheed Martin Corporation (S17-005), as well as the Basic Science Research Program of the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2016R1D1A1A09916273).

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Ahn J, Choi M, Suh ES (2018) Entropy-based system assessment metric for determining architecture's robustness to different stakeholder perspectives. *Syst Eng* 21:476–489. <https://doi.org/10.1002/sys.21448>
- Allen T, McGowan D, Moses J, Magee C, Hastings D, Moavenzadeh F, Lloyd S, Nightingale D, Little J, Roos D, Whitney D (2002) ESD terms and definitions. <https://esd.mit.edu/WPS/esd-wp-2002-01.pdf>
- Baldwin CY, Clark KB (2000) *Design rules*. MIT Press, Cambridge
- Bandyopadhyay S, Saha S, Maulik U, Deb K (2008) A simulated annealing-based multiobjective optimization algorithm: AMOSA. *IEEE Trans Evol Comput* 12:269–283. <https://doi.org/10.1109/TEVC.2007.900837>
- Baylis K, Zhang GL, McAdams DA (2018) Product family platform selection using a Pareto front of maximum commonality and strategic modularity. *Res Eng Des* 29:547–563. <https://doi.org/10.1007/s00163-018-0288-5>
- Blanchard BS, Blyler J (2016) *System engineering management*, 5th edn. Wiley, Hoboken
- Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. *J Stat Mech Theory E*. <https://doi.org/10.1088/1742-5468/2008/10/P10008> (Art ID P10008)
- Boehm BW (1988) A spiral model of software development and enhancement. *Computer* 21(5):61–72
- Borjesson F, Holta-Otto K (2014) A module generation algorithm for product architecture based on component interactions and strategic drivers. *Res Eng Des* 25:31–51. <https://doi.org/10.1007/s00163-013-0164-2>
- Brady TK (2002) Utilization of dependency structure matrix analysis to assess complex project designs. In: *ASME 2002 international design engineering technical conferences and computers and information in engineering conference*, 2002. American Society of Mechanical Engineers Digital Collection, New York, pp 231–240
- Braha D, Maimon O (1998) The measurement of a design structural and functional complexity. *IEEE Trans Syst Man Cybern A* 28:527–535. <https://doi.org/10.1109/3468.686715>
- Browning TR (2001) Applying the design structure matrix to system decomposition and integration problems: a review and new directions. *IEEE Trans Eng Manag* 48:292–306. <https://doi.org/10.1109/17.946528>
- Browning TR (2014) Managing complex project process models with a process architecture framework. *Int J Proj Manag* 32:229–241. <https://doi.org/10.1016/j.ijproman.2013.05.008>
- Carmines EG, Zeller RA (1979) *Reliability and validity assessment*. A Sage University Paper: quantitative applications in the social sciences, vol ser no 07-017. Sage Publications, Beverly Hills
- Coates G, Ritchey I, Duffy AHB, Hills W, Whitfield RI (2000) Integrated engineering environments for large complex products. *Concurr Eng Res A* 8:171–182. <https://doi.org/10.1106/Xpab-Ex5p-61lu-F924>
- Crawley E, Cameron B, Selva D (2016) *System architecture: strategy and product development for complex systems*. Pearson, Boston
- Deb K (2000) An efficient constraint handling method for genetic algorithms. *Comput Methods Appl Mech* 186:311–338. [https://doi.org/10.1016/S0045-7825\(99\)00389-8](https://doi.org/10.1016/S0045-7825(99)00389-8)
- Dolan B, Lewis K (2008) Robust product family consolidation and selection. *J Eng Des* 19:553–569
- Dori D (2002) *Object-process methodology: a holistics systems paradigm*. Springer, Berlin

- Eckert C, Clarkson PJ, Zanker W (2004) Change and customisation in complex engineering domains. *Res Eng Des* 15:1–21. <https://doi.org/10.1007/s00163-003-0031-7>
- Engel A, Reich Y (2015) Advancing architecture options theory: six industrial case studies. *Syst Eng* 18:396–414. <https://doi.org/10.1002/sys.21312>
- Engel A, Browning TR, Reich Y (2017) Designing products for adaptability: insights from four industrial cases. *Decis Sci* 48:875–917. <https://doi.org/10.1111/deci.12254>
- Eppinger SD, Browning TR (2012) Design structure matrix methods and applications. Engineering systems. MIT Press, Cambridge
- Friedenthal S, Moore A, Steiner R (2015) A practical guide to SysML: the systems modeling language, 3rd edn. Elsevier, MK, Morgan Kaufmann is an Imprint of Elsevier, Amsterdam
- Gershenson JK, Prasad GJ, Allamneni S (1999) Modular product design: a life-cycle view. *J Integr Des Process Sci* 3:13–26
- Guo F, Gershenson JK (2003) Comparison of modular measurement methods based on consistency analysis and sensitivity analysis. In: ASME 2003 international design engineering technical conferences and computers and information in engineering conference, 2003. American Society of Mechanical Engineers Digital Collection, New York, pp 393–401
- Guo F, Gershenson JK (2004) A comparison of modular product design methods based on improvement and iteration. In: ASME 2004 international design engineering technical conferences and computers and information in engineering conference, 2004. American Society of Mechanical Engineers Digital Collection, New York, pp 261–269
- Herrmann JW, Morency M, Anparasan A, Gralla EL (2018) Evaluating clustering algorithms for identifying design subproblems. *J Mech Des*. <https://doi.org/10.1115/1.4040176> (ArtN 081401)
- Holttä-Otto K, de Weck O (2007) Degree of modularity in engineering systems and products with technical and business constraints. *Concurr Eng Res A* 15:113–126. <https://doi.org/10.1177/1063293x07078931>
- Holttä-Otto K, Chiriac NA, Lysy D, Suh ES (2012) Comparative analysis of coupling modularity metrics. *J Eng Des* 23:787–803. <https://doi.org/10.1080/09544828.2012.701728>
- Horn RA, Johnson CR (1991) Topics in matrix analysis. Cambridge University Press, Cambridge
- IDEF D (1993) Integration definition for function modeling. ideo. Technical report. Federal Information Processing Standards Publications
- Ingber L, Petraglia A, Petraglia MR, Machado MAS (2012) Adaptive simulated annealing. In: Stochastic global optimization and its applications with fuzzy adaptive simulated annealing. Springer, Berlin, Heidelberg, pp 33–62
- Jung SJ, Simpson TW (2017) New modularity indices for modularity assessment and clustering of product architecture. *J Eng Des* 28:1–22. <https://doi.org/10.1080/09544828.2016.1252835>
- Kasperek D, Maisenbacher S, Kohn A, Lindemann U, Maurer M (2015) Increasing the reproducibility of structural modelling. *J Eng Des* 26:259–281. <https://doi.org/10.1080/09544828.2015.1026883>
- Kim G, Kwon Y, Suh ES, Ahn J (2017) Correlation between architectural complexity of engineering systems and actual system design effort. *ASME J Mech Des* 139(3):034501. <https://doi.org/10.1115/1.4035319>
- Li YP, Wang ZT, Zhang L, Chu XN, Xue DY (2017) Function module partition for complex products and systems based on weighted and directed complex networks. *J Mech Des*. <https://doi.org/10.1115/1.4035054> (ArtN 021101)
- Maimon O, Braha D (1996) On the complexity of the design synthesis problem. *IEEE Trans Syst Man Cybern A* 26:142–151. <https://doi.org/10.1109/3468.477869>
- Malik FF (1984) Strategie des Managements komplexer Systeme : ein Beitrag zur Management-Kybernetik evolutionärer Systeme. Schriftenreihe Unternehmung und Unternehmensführung, vol Bd 12. P. Haupt, Bern
- Martin MV, Ishii K (2002) Design for variety: developing standardized and modularized product platform architectures. *Res Eng Des* 13:213–235. <https://doi.org/10.1007/S00163-002-0020-2>
- Mattson C, Magleby S (2001) The influence of product modularity during concept selection of consumer products. In: ASME design engineering technical conferences-design theory and methodology conference, Paper No. DETC2001/DTM-21712, Pittsburgh, PA, September
- Meyer MH, Lehnerd AP (1997) The power of product platforms: building value and cost leadership. Free Press, New York
- Moon J, Lee D, Lee T, Ahn J, Shin J, Yoon K, Choi D (2015) Group decision procedure to model the dependency structure of complex systems: framework and case study for critical infrastructures. *Syst Eng* 18:323–338. <https://doi.org/10.1002/sys.21306>
- Newcomb PJ, Bras B, Rosen DW (1998) Implications of modularity on product design for the life cycle. *J Mech Des* 120:483–490. <https://doi.org/10.1115/1.2829177>
- Newman MEJ (2018) Networks, 2nd edn. Oxford University Press, Oxford
- Newman MEJ, Girvan M (2004) Finding and evaluating community structure in networks. *Phys Rev E*. <https://doi.org/10.1103/physreve.69.026113> (ARTN 026113)
- Otto KN, Wood KL (2001) Product design: techniques in reverse engineering and new product development. Prentice Hall, Upper Saddle River
- Phadke MS (1989) Quality engineering using robust design. Prentice Hall, Englewood Cliffs
- Sacks R, Eastman CM, Lee G (2004) Process model perspectives on management and engineering procedures in the precast/prestressed concrete industry. *J Constr Eng Manag ASCE* 130:206–215. [https://doi.org/10.1061/\(ASCE\)0733-9364\(2004\)130:2\(206\)](https://doi.org/10.1061/(ASCE)0733-9364(2004)130:2(206))
- Sage AP, Rouse WB (2009) Handbook of systems engineering and management, 2nd edn. Wiley series in systems engineering and management. Wiley, Hoboken
- Sanaei R, Otto KN, Hölttä-Otto K, Wood KL (2016) Incorporating constraints in system modularization by interactive clustering of design structure matrices. In: ASME 2016 international design engineering technical conferences and computers and information in engineering conference, 2016. American Society of Mechanical Engineers Digital Collection, New York
- Schwaber K (2015) Agile project management with scrum, 2nd edn. Microsoft Press, Redmond
- Schwabl F (2006) Statistical mechanics, 2nd edn. Springer, Berlin
- Sharon A, de Weck OL, Dori D (2011) Project management vs. systems engineering management: a practitioners' view on integrating the project and product domains. *Syst Eng* 14:427–440. <https://doi.org/10.1002/sys.20187>
- Sheard SA, Mostashari A (2009) Principles of complex systems for systems engineering. *Syst Eng* 12:295–311. <https://doi.org/10.1002/sys.20124>
- Shortell TM (2015) INCOSE systems engineering handbook: a guide for system life cycle processes and activities. Wiley, Hoboken
- Siddique Z, Rosen DW, Wang N (1998) On the applicability of product variety design concepts to automotive platform commonality. In: ASME design engineering technical conferences, Atlanta, GA, ASME. DETC/DTM-5661
- Sinha K, Suh ES (2018) Pareto-optimization of complex system architecture for structural complexity and modularity. *Res Eng Des* 29:123–141. <https://doi.org/10.1007/s00163-017-0260-9>
- Sinha K, Han SY, Suh ES (2020) Design structure matrix-based modularization approach for complex systems with multiple design constraints. *Syst Eng* 23:211–220. <https://doi.org/10.1002/sys.21518>

- Sosa ME, Eppinger SD, Rowles CM (2003) Identifying modular and integrative systems and their impact on design team interactions. *J Mech Des* 125:240–252. <https://doi.org/10.1115/1.1564074>
- Steward DV (1981) The design structure-system—a method for managing the design of complex-systems. *IEEE Trans Eng Manag* 28:71–74. <https://doi.org/10.1109/Tem.1981.6448589>
- Suh ES, Kott G (2010) Reconfigurable parallel printing system design for field performance and service improvement. *ASME J Mech Des* 132(3):034505. <https://doi.org/10.1115/1.4000961>
- Suh ES, de Weck OL, Chang D (2007) Flexible product platforms: framework and case study. *Res Eng Des* 18:67–89. <https://doi.org/10.1007/s00163-007-0032-z>
- Suh ES, Furst MR, Mihalyov KJ, de Weck O (2010) Technology infusion for complex systems: a framework and case study. *Syst Eng* 13:186–203. <https://doi.org/10.1002/sys.20142>
- Suh ES, Chiriac N, Holtta-Otto K (2015) Seeing complex system through different lenses: impact of decomposition perspective on system architecture analysis. *Syst Eng* 18:229–240. <https://doi.org/10.1002/sys.21294>
- Thebeau RE (2001) Knowledge management of system interfaces and interactions from product development processes. Massachusetts Institute of Technology, Cambridge
- Tilstra AH, Seepersad CC, Wood KL (2012) A high-definition design structure matrix (HDDSM) for the quantitative assessment of product architecture. *J Eng Des* 23:764–786. <https://doi.org/10.1080/09544828.2012.706748>
- Trigeorgis L (1996) Real options: managerial flexibility and strategy in resource allocation. MIT Press, Cambridge
- Ulrich KT, Eppinger SD (2016) Product design and development, 6th edn. McGraw-Hill Education, New York
- Van Eikema Hommes QD (2008) Comparison and application of metrics that define the components modularity in complex products. In: ASME 2008 international design engineering technical conferences and computers and information in engineering conference, 2008. American Society of Mechanical Engineers Digital Collection, New York, pp 287–296
- Whitney DE (2004) Mechanical assemblies: their design, manufacture, and role in product development. Oxford series on advanced manufacturing. Oxford University Press, New York

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.