CrossMark

ORIGINAL PAPER

Pareto-optimization of complex system architecture for structural complexity and modularity

Kaushik Sinha[1] · Eun Suk Suh[2]

Abstract Due to ever-increasing complexity of cutting-edge engineering systems, the need for managing structural complexity and modularity of such systems is becoming important. The complexity of the overall system architecture is mostly decided during the initial concept generation stage, when configurations of major modules within the system are determined. In this paper, we present a multi-objective optimization framework for (1) minimizing the variation in complexity allocation to individual modules, while (2) maximizing for the degree of modularity. The optimization framework was applied to a case study, where a trailing bogie system for railroad train was optimized for structural complexity allocation among individual modules and overall system modularity. The modularity maximizing decomposition is shown to induce a large variation in module-level complexity distribution with a small fraction of modules sharing a disproportionately large chunk of overall system complexity, while equitable distribution of module-level complexity leads to erosion in the degree of modularity achieved for the resulting system decomposition.

Keywords System architecture · Structural complexity · Modularity · Pareto-optimality

✉ Eun Suk Suh
essuh@snu.ac.kr

[1] Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA

[2] Graduate School of Engineering Practice, Institute of Engineering Research, Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul 08826, South Korea

1 Introduction

One of the most fundamental heuristic guidelines in complex system design is to keep the system architecture as simple as possible. However, contrary to basic design rules, architectures of latest cutting-edge engineering systems are becoming more complex due to ever-increasing complexity of new technologies and infrastructures to support them, as well as demand for better lifecycle performances (Frey et al. 2007). This overall trend necessitates an important need for proper system architecture complexity management process. Without the smart management of system complexity, the system's overall complexity may become unwieldly, leading to undesirable results, such as longer research and development (R&D) period, higher R&D and lifecycle costs, and possible increase in system's post-launch maintenance cost.

One of the widely used design strategies for managing complex system architecture is the modular design strategy (Baldwin and Clark 2000). Using modular design methods, a system is divided into several subsystems called modules. A module in a typical modular system has many intra-connections among internal module elements, but has relatively few external connections to other modules. Ideally, each module in the system performs a specific function required by the system to achieve overall system performance objective. This allows system architects to replace individual modules to improve specific functionality carried out by that specific module, with minimum impact to the rest of the system. A good example is a desktop computer, consisting of the main computer module that performs actual computation, monitor module responsible for information display, and keyboard and mouse modules for information input. While each module has its own internal structural arrangement, its connection to other modules is

Springer

minimal, usually by a simple cable or wireless signal. Additionally, each desktop computer module can easily be replaced without any significant engineering design change to the rest of the system.

As discussed, module-based system has significant advantages which can influence ways that complex systems can be developed and updated throughout the entire life-cycle. However, many works published in academia proposed module-based design approaches without explicitly taking into account the overall complexity of the system or complexity of its modules. Also, previous research efforts either focused on measuring and managing system complexity, or only on measuring and managing system modularity. This existing research gap needs to be addressed to advance the state-of-art in complex system design methodologies.

In this paper, we propose a Pareto-optimization framework to design complex engineering system architectures for modularity and system's structural complexity allocation. Using the framework, degree of system modularity and structural complexity allocation among system modules can be explored simultaneously within the architectural design space. This allows system architects to solve the system architecture design problem as a multi-objective optimization problem, where the system is designed for maximum degree of modularity while variation of structural complexity among system modules is minimized. Such module-level structural complexity allocation strategies enable more stable and efficient planning of development schedule for module-level design and development, since design and development efforts invested for each module would be approximately the same with similar level of complexity for all modules. As result, a Pareto-optimal frontier for system modularity and complexity allocation is created, which then can be used to guide decision for final system architecture selection. The framework introduced in this paper contributes to areas of complex system design by proposing a new way to optimize for system modularity and system complexity allocation simultaneously.

2 Previous research

2.1 System complexity

Theory of complexity has received a lot of attention in the field of engineering system architecting and design (Asikoglu and Simpson 2010; Braha and Maimon 1998; Dolan and Lewis 2008; Malik 1992; Tamaskar et al. 2014). There are multiple aspects of complexity aspects in the context of a complex engineered system design and development effort. Complexity can be categorized into (1) internal and

(2) external complexity. There are three main dimensions of internal complexity that emerge from the existing literature in the context of system design and development— (1) structural complexity; (2) dynamic complexity and (3) organizational complexity (Braha and Bar-Yam 2007; Lindemann et al. 2009; Malik 1992; Riedl 2013; Weber 2005).

Structural complexity arises when a system is composed of many components and whose interaction is difficult to describe or understand. Structural complexity relates to the notion of the architecture of a system and, therefore, represents complexity determined by the form. Given a set of basic functions, there are multiple forms that can perform specified set of functions. An example is the function of slowing food spoilage. Over millennia humans have developed numerous concepts for achieving this particular function—cooling, freezing, irradiating, salting, canning and vacuum-packing the food. Achieving the desired functionality and system behavior with minimal structural complexity becomes an important criterion for down-selection of design/architectural concept. The system architecture impacts the complexity of the system during its initial design phase, during the implementation phase and during the changes that will occur in its lifetime.

Dynamic complexity is driven by what system does (i.e., its functions as well as the chosen concept). A system is deemed dynamically complex if its external behavior/dynamics is difficult to describe and predict effectively. Since the system behavior is described over an operational envelope of the system, dynamic complexity is an implicit function of this envelope. There are two primary sources of dynamic complexity: (1) interactivity among functional attributes and (2) uncertainties in their interactions. Note that the system behavior is often bounded by the underlying system architecture and, therefore, dynamic complexity possesses a strong positive correlation with structural complexity for well-engineered systems (Conway 1968; Sinha 2014) and they relate to the interplay between form and function/system behavior.

Finally, organizational complexity relates to the system development organization's structure and the system development process. Organizational structure often mirrors the system's architecture and is thereby closely related to structural complexity. Figure 1 describes a complexity typology for engineered complex systems from a system design and development perspective. In general, it has been observed that structural complexity strongly correlates with organizational complexity (Conway 1968; MacCormack et al. 2012).

The external complexities relate to facets that are usually not under control of the system development organization. They typically include complexities associated with funding mechanisms, market dynamics, political and

Fig. 1 Complexity typology for engineered systems Adapted from Sheard and Mostashari (2009)
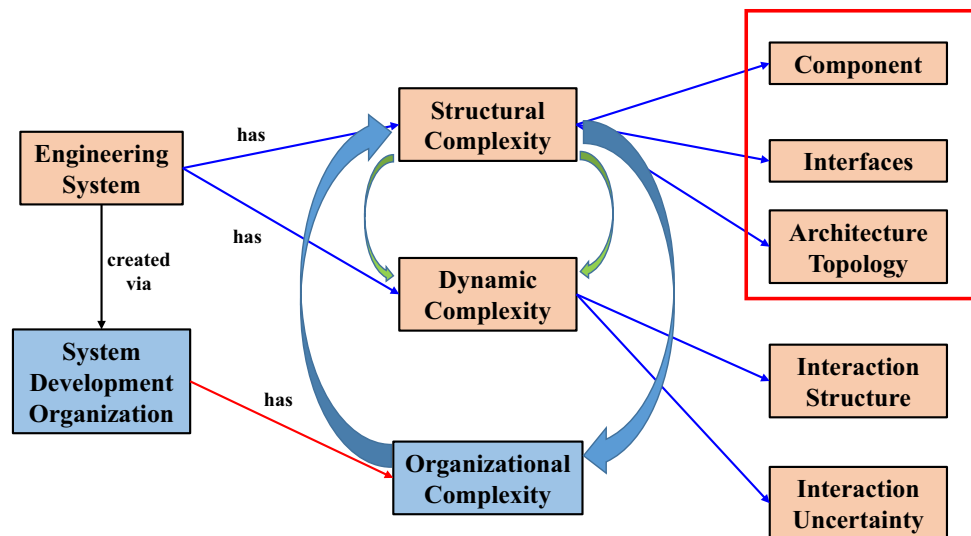
institutional complexities, etc. They are much larger in scope and usually tend to encompass very large-scale projects (Sussman 2000).

From different complexity categories mentioned, structural complexity of the system architecture is the focus of this paper. The structural complexity is a characteristic that can be measured, and is influenced by the number of components, interfaces and the connectivity structure of the system. More components and intricate connectivity structure increase the design, development and operation effort (Maier and Rechtin 2009) and it also has been observed that complexity is strongly correlated with the development effort and cost (Barton et al. 2001; Braha and Bar-Yam 2007; Sinha 2014). Increasing complexity also leads to the phenomena of cost-complexity spiral (Tamaskar et al. 2014). This issue was explicitly addressed in the defense sector, where numerous scholars observed the growing cost of system complexity (Arena et al. 2008; Augustine 1997). There is consensus among academia and industry that there is a need for framework to manage system complexity. For the rest of this article, we will focus our attention to structural complexity, which is related to the architecture of system. In this context, the term structural complexity and architectural complexity are used interchangeably throughout the paper.

To address the management of system complexity, the complexity of the system needs to be measured in a quantitative way. Previous literature on system complexity has often stopped short of prescribing an objective and quantifiable metric, which is an essential element in enabling complexity-inclusive system architecting and active complexity management. Various system complexity metrics are developed and proposed since 1970s, with early works originating from the software engineering sector, such as cyclomatic metric (McCabe 1976) and

interconnectivity-based metric (Kafura and Henry 1981). As time progressed, numerous scholars introduced metrics based on different theories. According to literature survey conducted by Tamaskar et al. (2014), complexity metrics developed are based on information theory, network theory and empirical data. Information theory-based metrics determine system complexity using system information content, and metrics developed by Maimon and Braha (Braha and Maimon 1998; Maimon and Braha 1996), Hornby (2007) and Allaire et al. (2012) are in this category. Network theory-based metrics determine system complexity using overall topology of the system represented by the network. Metrics proposed by Mathieson and Summers (2010), Sinha and de Weck (2013) and Chen and Li (2005) are based on network theory. Another notable complexity metric based on empirical data is proposed by Bearden (2003). Application of proposed complexity metrics can be found in numerous literature. Tamaskar et al. (2014) demonstrated the use of their metric on space satellite application. Complexity metrics developed in Sinha and de Weck (2013) were utilized for system architectural configuration complexity assessment (Min et al. 2016), technology infusion impact on system complexity (Min et al. 2015), and product platform complexity assessment (Kim et al. 2016).

2.2 System modularity

Modularity is an important property of a system where it represents the degree to which a system can be divided into several strategic chunks called modules. A module can be defined as "a relatively independent chunk of a system that is loosely coupled to the rest of the system", (Holtta-Otto et al. 2012) where a chunk may be a subsystem or a single element. In many engineering systems, a module consists

of a form where a specific function of the system is embedded during the system synthesis process, enabling system architects to add, remove or enhance a specific system function with ease.

A modular architecture enables relatively independent development of modules due to minimal systemic interaction among them. This aids localized development and technology infusion within the system while limiting the perception of underlying system complexity (Baldwin and Clark 2000). This helps to manage the complexity and the development process better while minimizing the "design churn effect" (Yassine et al. 2003). A highly modular system decomposition strategy might ease the development process by enabling efficient application of divide and conquer principles and by impacting the perception of underlying complexity. This helps manage the actual, underlying system complexity more efficiently. A modular system is deemed to favor flexibility for future modifications and novel technology infusion into the system over a period of time by limiting the degree of interaction between system modules (Baldwin and Clark 2000). Modularity benefit depends on the system decomposition strategy and is a measure of relative cohesion within and across the modules (Newman 2010).

Modular system has several advantages such as ease of assembly (Holtta-Otto and de Weck 2007), ease of maintenance (Suh and Kott 2010) and ease of architecture adaptability (Engel and Browning 2008; Engel et al. 2016). Due to such advantages, modular design strategy is widely used in engineering community to design complex system architectures using manageable and independently upgradable modules, such as fiber placement system and truck powertrain (Engel and Reich 2015). There is a whole field of research concerning modular system design and analysis methodologies. Early works by Alexander (1964) and Simon (1996) set the tone for complex system design by hierarchical decomposition, followed by the seminal work of Baldwin and Clark (2000), which established design rules for modular systems. Numerous methods to determine modularity are proposed in the literature, ranging from more engineering and functionality focused (Guo and Gershenson 2004; Martin and Ishii 2002; Mikkola and Gassmann 2003; Sosa et al. 2003) to more network science centric methods (Blondel et al. 2008; Newman 2010). These works are supplemented by development of other methods, such as modular function deployment (Ericsson and Erixon 1999), flexible platform development (Suh et al. 2007), modular product family concept development (Otto et al. 2016), and modular system design based on architecture options theory (Engel and Reich 2015) to just name a few.

To support modular system design, various modularity metrics have been proposed in time. According to Holtta-

Otto et al. (2012), there are two types of modularity metrics. First type of metric measures the degree of coupling in order to determine degree of independence of a module. Several modularity metrics of first kind are introduced over time. Examples include singular value-based metric proposed by Holtta-Otto and de Weck (2007), coupling-based metric proposed by Martin and Ishii (2002), design interface-based metric proposed by Sosa et al. (2003) and the minimum description metric (MDL) proposed by Yu et al. (2007), which measures the sum of model description length and mismatched data description of the architecture. Second category of metric measures the similarity of modules in the system. Primary consideration for modular metrics of this type is connective relationship between elements within the system, being sensitive to tight and loose coupling among them. This metric has been proposed by the network science community and is context-agnostic and solely based on iterative graph partitioning techniques (Newman 2010). This methodology is based on splitting a graph into sub-graphs such that interfaces within sub-graphs are maximized while minimizing those between the sub-graphs. In this paper, we use this approach for finding optimal decompositions from modularity maximization point of view.

2.3 Research gap analysis and proposed research approach

Based on comprehensive literature survey in system complexity management, modular system design methodology, complexity metrics and modularity metrics, there are many works published in their respective fields. However, there is virtually no work that approached complex system design problem from the quantitative management of modularity and structural complexity concurrently. Only work that authors are aware of is by Tamaskar et al. (2014), who applied their complexity and modularity measuring framework to the space satellite application. This paper addresses this research gap in complex system design and complexity management through the introduction of Pareto-optimization framework for system architecture from the viewpoint of system modularity and module-level structural complexity allocation. The optimization is based on the assumption that the actual complexity of the system is a property of the system and is independent of the system decomposition strategy employed. However, system decomposition does influence the allocation of the total complexity to individual modules and those tied to the system integration aspect. Using the proposed framework, complex systems can be optimized for degree of modularity, while variation of structural complexity among system modules is minimized. This leads to more equitable allocation of module-level complexity and such

allocation strategies enable more stable and efficient planning of development schedule for module-level design and development.

3 System architecture analysis and optimization framework

3.1 Overview

The objective of this research is to introduce a system architecture optimization framework that simultaneously optimizes the allocation of structural complexity to its modules, while maximizing the overall system modularity. To accomplish this, multiple system representation and analysis steps were performed. First, appropriate metrics for measuring structural complexity and system modularity are selected. Second, using selected complexity and modularity metrics, a multi-objective optimization problem for system architecture design is formulated. The formulated optimization framework is then applied to a system, where the number of different modularized concepts of the system is generated and evaluated for its optimality in modularity and structural complexity. Using the optimization result, Pareto-optimal frontier of system's structural complexity versus modularity is generated for decision maker's evaluation and final concept selection. In this section, complexity and modularity metrics selection and optimization framework formulation are presented in detail.

3.2 Complexity and modularity metrics

3.2.1 Complexity metric

Structural complexity metric created for engineering systems is composed of (1) sum of individual component's internal complexity; (2) the complexities of each connection between components; and (3) a quantitative measure that describes structural topology of component arrangements. What is missing from the proposed complexity measures in the literature is adherence to rigorous construct validity (Weyuker 1988; Yassine et al. 2003). A set of desirable characteristics that any proposed complexity metric should have has also been discussed elsewhere (Bashir and Thomson 1999) that essentially reiterates the requirements of construct and face validity.

We believe that any proposed complexity metric should be investigated for construct and face validity. One rigorous criteria set for construct validity of complexity measure was proposed by Weyuker (1988) and the complexity metric used in this paper was validated against the same (Sinha 2014; Sinha and de Weck 2013). In addition, any complexity metric proposed should also correlate strongly

with design/development cost or effort (i.e., face validity) (Bellmann et al. 2011). Table 1 shows a shortlist of prominent complexity metrics and their comparison with respect to satisfaction of construct validity (i.e., Weyuker's criteria) and their computability (for any medium to large size graphs). The chosen spectrum of complexity metrics covers a variety of structural characteristics they emphasize. We can observe that the graph energy satisfies Weyuker's criteria while being computable for large graphs (Sinha 2014; Sinha and de Weck 2013).

The functional form that is adopted (Sinha 2014; Sinha and de Weck 2013) for structural complexity estimation is

$$\text{Structural complexity, } C = C_1 + C_2 C_3. \tag{1}$$

The first term C_1 is the summation of components complexities within the system, which can vary from component to component. The second combined term, $C_2 C_3$, represents complexities arising due to component-to-component interfaces and the overall connective arrangements (topology) of system components for the integrated system. This term is closely related to the system realization effort to integrate the whole system using system components, with the assumption that as the system interface and topology complexity increase, so does the system integration effort. For C_1, if the individual component's complexity is designated by α_i, then can be written as:

$$C_1 = \sum_{i=1}^{n} \alpha_i.$$

It is important to note that C_1 is about individual component, which utilizes information local to that specific component.

The second term C_2 is the summative term that describes interaction complexity β_{ij} between components, which can be rewritten as:

$$C_2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \beta_{ij} A_{ij},$$

where $A \in M_{nxn}$ is the representation of system's connectivity structure in the form of binary adjacency matrix:

$$A_{ij} = \begin{cases} 1 & \forall [(i,j)|(i \neq j) \text{ and } (i,j) \in \Lambda \\ 0 & \text{otherwise} \end{cases}.$$

In the equation above, the connected node set is represented by Λ and the total number of components in the system is represented by n. In the matrix A, the diagonal elements are set to zero. For the interface complexity term C_2, information regarding categorization or quantification of pair-wise interfaces and partial knowledge of the system architecture are required to estimate β_{ij}. One way of estimating complexity of interface β is proposed in the original

Table 1 List of existing complexity metrics showing their computability and construct validity (Weyuker 1988). Graph energy stands out as both computable and satisfies Weyuker's criteria and establishes itself as a theoretically valid measure (i.e., construct validity) of complexity

Complexity measure	Characteristic/feature	Computability	Construct validity
Number of components (Bralla 1986)	Count-based measure	Yes	No
Number of interactions (Pahl and Beitz 1996)	Count-based measure	Yes	No
Structural design complexity (Braha and Maimon 1998)	Count-based embedded information content	Yes	No
Whitney index (Whitney et al. 1999)	Count-based measure	Yes	No
CoBRA complexity index (Bearden 2003)	Empirical measure for similar systems	Yes	No
Decomposability based measures (Ameri et al. 2008)	Decomposition of entity-relationship graphs	Yes	No
Graph planarity (Kortler et al. 2009)	Information transfer efficiency	Yes	No
Nesting depth (Lindemann et al. 2009)	Extent of hierarchy	Yes	No
Automorphism based entropic measures (Dehmer and Emmert-Streib 2009)	Heterogeneity of network structure	No	Yes
Graph energy based measure (Sinha and de Weck 2013)	Composite measure on connectivity structure	Yes	Yes

research by Sinha (2014). In the work, it is suggested that the interface complexity β can be expressed as a fraction of component complexity. As an example, if the interface is about 20% as complex as the interfacing components, the value for β would be 0.2. In the case study presented later in this paper, we have assumed the pair-wise interface complexity β_{ij} between component i and j to be 10% of component complexity of the component with greater individual complexity value. For an additional reference, recent work by Yassine and Naoum-Sawaya (2017) also explores the topic of system interface complexity.

The final term of the complexity equation, C_3, is the metric for topological complexity of the system, which measures the complexity arising due to the system interface's topological arrangement. The expanded equation for C_3 is:

$$C_3 = \frac{E(A)}{n}, \quad \text{where } E(A) = \sum_{i=1}^{n} \sigma_i(A).$$

In the equation the $\sigma_i(A)$ is the matrix A's ith singular value and $E(A)$ is the graph energy of the matrix A, which is the sum of the singular values of matrix A. The sum of singular values of a matrix has been given different names in various disciplines. In the linear algebra literature, it is called the nuclear norm (Bernstein 2009; Horn and Johnson 1991) or matrix energy (Nikiforov 2007) or a modified version of graph energy (Gutman 2001). Important point to note is that singular values of a generic adjacency matrix are real and non-negative and are well defined for symmetric and asymmetric matrices. This property allows for uniform handling of both undirected and directed nature of

system interfaces. To calculate C_3, the system level binary adjacency matrix needs to be available and, therefore, the connectivity structure of the architecture must be known. This term represents complexity due to system connectivity structure or the topology and helps distinguish architectures with different connectivity structures, even if they have same number of components and connections. This is demonstrated through simple architecture examples shown in Fig. 2. In the figure, architectures $A1$ and $A2$ have same number of nodes and connections, but different connectivity structures. To calculate $E(A)$, a binary adjacency matrix for each architecture is constructed, which is also shown in the figure. Using these matrices, singular values for each architecture were calculated, and $E(A)$ values were obtained. Results show that two architectures shown in Fig. 2 have different $E(A)$ values due to different connective structural arrangements. The term $C_2 C_3$ in Eq. (1) reflects the system realization effort from set of components (it could be custom designed or from a catalogue of off-the-shelf components). On the other hand, term C_1 in Eq. (1) relates to the component development efforts.

Finally, the structural complexity metric introduced in Eq. (1) in detailed form is shown below:

$$C = \sum_{i=1}^{n} \alpha_i + \left(\sum_{i=1}^{n} \sum_{j=1}^{n} \beta_{ij} A_{ij} \right) \frac{E(A)}{n}. \tag{2}$$

Implication of different terms of the structural complexity metric in relation to systems development activities is described in Fig. 3.

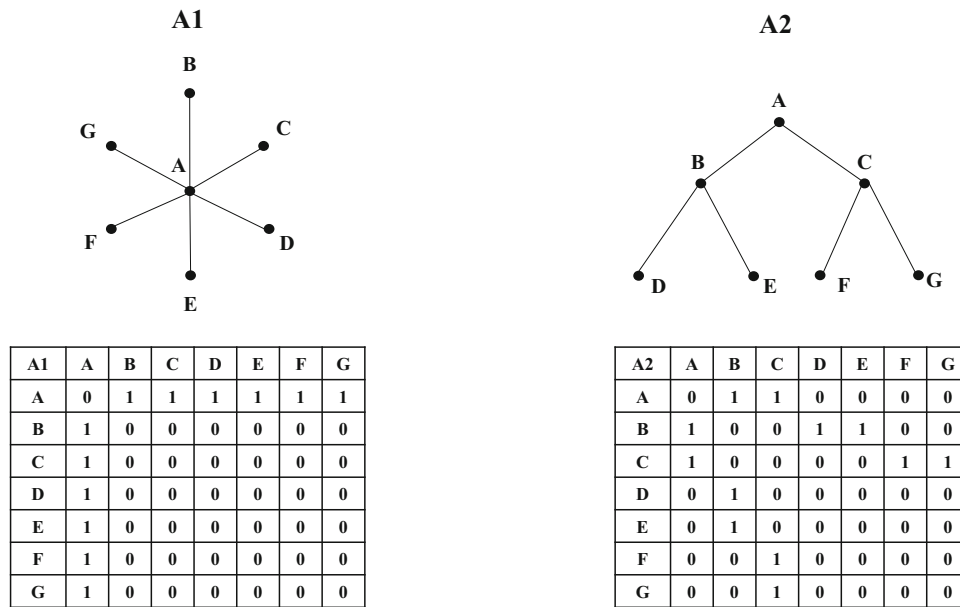The structural complexity metric in Eq. (1) is demonstrated to be fully compliant with conceptual validity

Fig. 2 Two architectures with different connective structures, but with same number of nodes and connections and their corresponding binary matrix representations: $E(A1) = 2.45 + 2.45 = 4.90$ and $E(A2) = 2.00 + 2.00 + 1.415 + 1.415 = 6.83$

Fig. 3 Detailed description of the complexity metric and what each term actually represents in the context of system design and development

$$C(n,m,A) = \sum_{i=1}^{n} \alpha_i + \left(\sum_{i=1}^{n} \sum_{j=1}^{n} \beta_{ij} A_{ij} \right) \frac{E(A)}{n}$$
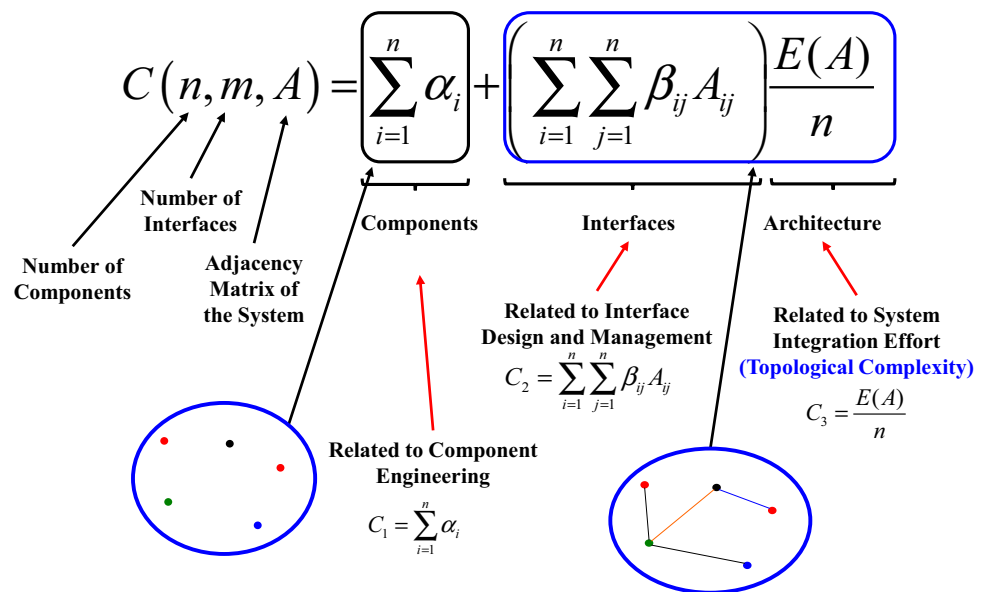
properties proposed by Weyuker (1988). For more detailed explanation of the metric introduced in Eq. (1), interested readers can refer other literature by Sinha (2014). The important characteristics of the complexity metric adopted in this paper are as follows:

1. The topological complexity factor is well-defined, non-negative real number for any general matrix and, therefore, applies to both directed and undirected adjacency matrices.

2. The structural complexity metric has been validated against Weyuker's criteria and shown to have sound theoretical grounding as a complexity measure.

3. This metric considers heterogeneities in (1) components, (2) pair-wise interfaces and (3) system connectivity structure in estimating system complexity.

4. Segregates distributed architectures from hierarchical and centralized structures based on regimes of values of topological complexity, C_3.

5. This metric enables quantitative system-level trade-off analysis between different architectural patterns (distributed vs. hierarchical vs. centralized connectivity structures) and their effectiveness.
6. This metric provides domain-specific customizability in estimation of component and interface complexities and accommodates for systematic uncertainty propagation in complexity estimation process.

3.2.2 Complexity attribution

System decomposition strategy refers to the decomposition of any system into smaller sub-systems/modules that are easier to manage (Frey et al. 2007; Simon 1996; Sosa et al. 2003). There are other related definitions or point of view on system decomposition (Baldwin and Clark 2000) that often uses functional view of the system. Once system decomposition is made available, the complexity attribution process performs accounting of complexity of different modules and integrative complexity required due to integration of modules.

Complexity attribution is a method for consistent accounting of complexity assigned to different sub-systems/modules and contribution of complexity from system integration. In essence, the complexity attribution method describes how overall structural complexity is distributed within the system, given a system decomposition strategy. System decomposition strategy refers to the decomposition of any system into smaller subsystems/modules that are easier to manage (Hirschi and Frey 2002; Ulrich and Eppinger 2012). There are other related definitions or point of view on system decomposition (Baldwin and Clark 2000) that often uses functional view of the system. Once system decomposition is made available, the complexity attribution process performs accounting of complexity of different modules and complexity attributed to integration of modules.

Attribution of system complexity to its modules and to integration has important consequence to system design. Empirical results published by Kim et al. (2017) indicate that as the complexity of system increases, actual design time increases in exponential fashion. If the system complexity is evenly attributed to modules, meaning that the complexity difference between the most complex module and the least complex module is small, then the time it takes to design individual module is roughly in equivalent. However, if the complexity is attributed unevenly, meaning that the complexity difference between the most complex module and the least complex module is large, then the time it takes to design the most complex module can increase, impacting the overall system development schedule.

Let us define the system decomposition by a map Gr(.), which is a component to module map. Here, each component is assigned to a module and this map is unique (i.e., a component can be a member of a unique module). It is probably best to use a motivating illustration in Fig. 4 as we move through the steps. In the figure, we have ten components which are assigned to two modules. The binary adjacency matrix A (which is symmetric since all interfaces are bi-directional) for this synthetic system representation can be written in terms of sub-matrices ($A^{(1)}$, $A^{(2)}$, K). Notice that sub-matrix K represents the inter-module connectivity structure and is different from the number of modules, with $k = 2$ in this case. Here, $A^{(1)}$ and $A^{(2)}$ represent the binary adjacency matrices of module 1 and 2 respectively.

Expanding to the general case with k modules and given system decomposition map Gr(.), we can express the individual module complexity for ith module as follows:

$$C^{(i)} = C_1^{(i)} + C_2^{(i)} C_3^{(i)}, \tag{3}$$

where the components of the complexity metric are defined as:

$$C_1^{(i)} = \sum_{p=1}^{n_i} \alpha_p^{(i)}; \quad C_2^{(i)} = \sum_{p=1}^{n_i} \sum_{q=1}^{n_i} \beta_{pq}^{(i)} A_{pq}^{(i)}; \quad C_3^{(i)} = \frac{E(A^{(i)})}{n_i}.$$

The method described above is same as that of computing structural complexity metric for a module in isolation.

Given the system decomposition, the integrative complexity is defined as:

$$\text{Integrative complexity, IC} = C - \sum_{i=1}^{k} C^{(i)}. \tag{4}$$

Since the components are divided into modules, we have $C_1 = \sum_{i=1}^{k} C_1^{(i)}$ and, therefore, we can write integrative complexity (IC) as:

$$\text{IC} = C_2 C_3 - \sum_{i=1}^{k} C_2^{(i)} C_3^{(i)}. \tag{5}$$

While comparing across a range of engineering system architectures with given decompositions, it is prudent to use the normalized version of integrative complexity defined as:

$$\text{IC}_{\text{normalized}} = 1 - \left(\frac{\sum_{i=1}^{k} C_2^{(i)} C_3^{(i)}}{C_2 C_3} \right). \tag{6}$$

It should be noted that the normalized version of the integrative complexity is bounded in [0, 1]. It is also noted that the integrative complexity is independent of

Fig. 4 Simplified representation of a synthetic two-module system with ten components and ten bi-directional interfaces in network and binary adjacency matrix form
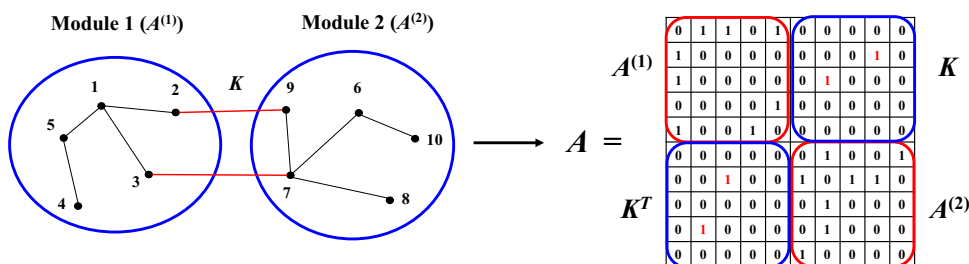
components and what matters are the interfaces and how they are topologically arranged.

The integrative complexity captures the part of the complexity associated with integration of the modules. The integrative complexity (IC) can be treated as the complexity associated with the integration effort at the system level. The relative magnitude of integrative complexity indicates how much of the total structural complexity arises from integration of modules/subsystems. We can think of integrative complexity as an indicator of system realization effort, given the module description. For a constant level of total structural complexity, lower integrative complexity implies higher in-module complexity (i.e., complexity embedded within the module) for the system under consideration.

3.2.3 Modularity metric

Now let us focus our attention on modularity. The degree of modularity (Q), is dependent on the system decomposition adopted and distribution of intra- and inter-module interfaces. In this paper, the modularity metric developed by Newman (2010) is used, since it has been used extensively in network science literature and has direct links to classical graph theory, and also to graph splitting algorithms used in graph theory. Let us define y_{ii} as fraction of intra-module interfaces while y_{ij} represents fraction of inter-module interfaces. For computation of the Newman modularity metric Q, we created the module matrix e (also known as community matrix), shown below:

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

For the module matrix e, the row sum is written as $a_i = \sum_{j=1}^{k} e_{ij} = y_{ii} + \left(\sum_{j=1}^{k} y_{ij} \right) \bigg/ 2$ and the modularity metric Q (Newman 2010) is defined as:

$$Q = \sum_{i=1}^{k} (e_{ii} - a_i^2) = \text{Tr}(e) - ||ee^T||, \tag{7}$$

here e_{ii} stands for the fraction of edges with both end vertices in the same module i, while a_i represents the fraction of edges with at least one end vertex inside module i (i.e., the inter-module interfaces). To account for double counting of inter-module interfaces, a factor of ½ has been applied to all off-diagonal terms in the module matrix above. Newman modularity metric (Q) can also be written as the function of trace of the matrix e, which is the sum of its diagonal entries, and the Frobenius norm of the product of module matrix e and its transpose (Horn and Johnson 1991).

System complexity C and modularity Q can be traded and that this trade-space can be defined by a family of non-dimensional ratios. We can write structural complexity as the sum of in-module structural complexities and integrative complexity (as shown in Sect. 3.2.2), which is the complexity introduced due to inter-module connectivity. Note that modularity does not depend on how many components belong to each module and detailed topological features within each module. Once the modules are 'defined', there is no role for topological features within the modules in the modularity index expression. This is not the case for topological complexity, since it inherently captures the global effect of relative module size (i.e., number of nodes within each module). It has been demonstrated analytically that complexity and modularity trade-space can be traversed using a set of seven ratios that are constrained by the set of three equations for a system with two modules (Sinha 2014). These ratios were shown to determine how the overall complexity is distributed within and across the modules and are related to integrative complexity of the system (Sinha et al. 2017).

3.3 Formulation of multi-objective optimization problem

Now we are in position to formulate the multi-objective optimization problem to find the optimum system decomposition Gr(.) that minimizes the structural complexity variation among system modules while maximizing the overall system modularity: Let us consider the distribution of module structural complexity, $C^{(\text{module})} = [C^{(1)}, C^{(2)},...,$ $C^{(k)}]$ where k stands for the number of modules. An

approximate measure of standard deviation of module-level structural complexity is given by

$$S_{C^{(\text{module})}} = \left[\max\left(C^{(\text{module})}\right) - \min\left(C^{(\text{module})}\right) \right] \Big/ 4.$$

The widely used approximation of standard deviation (Mood et al. 1973) above was used in lieu of the sample standard deviation formula to account for sample size effects, since the number of modules in the system being analyzed may be relatively small (Hozo et al. 2005). As stated earlier, an equitable distribution of module structural complexity (i.e., with minimal standard deviation in module-level structural complexity) is desirable from system architecting and design perspective. This will ensure minimal asymmetry in the module-level complexity that needs to be handled by the development groups responsible and there will be no module-level complexity bottleneck that could jeopardize the system development effort and subsequent operation and continuous availability of the system. Such an equitable distribution of module-level structural complexity tends to have a negative impact on the overall modularity of the system, and reduction in variation of module complexity is accompanied with reduction in the degree of modularity of the system. This requires us to make a trade-off between the two aspects of system architecture with the intention of achieving the optimal system architecture. Once the complexity and modularity metrics are available, our aim is to find system decompositions (i.e., association of components to modules/groups) Gr(.) with minimal variation in module structural complexity while maximizing the degree of modularity (i.e., maximization of the modularity index). For the modularity maximization strategy in this research, a two-phase community detection strategy used by Blondel et al. (2008) is implemented. In the first phase of the mentioned strategy, each node (or a component) is assigned a community (or a module) based on modularity-delta maximization using local search. During the second phase, it creates a hierarchical module structure. In the next iteration, the current community structure is the output from the second phase of the previous iteration. Iteration continues until there is no further improvement in overall modularity score, measured by the Q metric. Finally, the associated multi-objective optimization problem can be stated as:

$$\left. \begin{array}{l} \underset{\text{Gr}(.)}{\text{Minimize:}}\ f_M(\text{Gr}(.)) \equiv \left[S_{C^{(\text{module})}}, -Q \right] \\ \text{s.t.}\quad g_i(\text{Gr}(.)) \le 0 \end{array} \right\}. \tag{8}$$

In this case, there is no other constraint to be satisfied for the system decompositions to be feasible. There may be circumstances where additional context-specific constraints are imposed for system decompositions to be feasible. Notice that this is a combinatorial optimization problem

and, therefore, outside the realm of applicability of gradient-based optimization methods and we, therefore, adopt the heuristics-based, evolutionary computational approach. Our 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 $\forall\ M\ \varepsilon\ 1, 2,\ldots, k$, $f_M(\text{Gr}_i) \le f_M(\text{Gr}_j)$, such that $f_M(\text{Gr}_i) < f_M(\text{Gr}_j)$. Now, from the solution set $\mathbf{P}$, Pareto-optimal solution set $\mathbf{P_0}$ are those that are not dominated. The Pareto-optimal set in the entire search space S is the non-dominated set. A multi-objective optimization algorithm yields a set of solutions that are not dominated by any solution that it encounters. In this study, we have developed and used a combinatorial variant of AMOSA algorithm (Bandyopadhyay et al. 2008) with adaptive cooling schedule to improve the efficiency of the algorithm (Ingber 1996). The details of this combinatorial, adaptive multi-objective simulated annealing approach will be part of a future manuscript.

3.4 Abstract system model creation and modular system concept generation

To extract necessary information required for the optimization, an abstract model of the system needs to be created. The system model should be easy enough to create with minimal information available about the system. This is due to the timing of design and development phase that the optimization must take a place. The ideal timing for applying proposed optimization is during the preliminary system concept development stage, when there is very little information available for the new system in consideration. Given the constraints, it was decided to use the design structure matrix (DSM) (Eppinger and Browning 2012) as the abstract modeling methodology of choice. Modeling system elements as simple row and column entries of the matrix, and specifying connections between system elements using a simple binary entry allows system model to be created without much of detailed system information, making it ideal for the preliminary concept design phase. Additionally, information required for optimization can be extracted and the optimization results can also be displayed using the DSM.

Once the system DSM is created, several different modular configurations of the system can be generated using several clustering algorithms available, such as the ones proposed by Newman (2010) and Blondel et al. (2008). Additionally, using clustering algorithms mentioned, one can generate population of system configurations, and evaluate them for optimality through the use of heuristic-based optimization algorithm, such as genetic algorithm (Holland 1992) or simulated annealing (Kirkpatrick et al. 1983). For the research presented, we have used various clustering algorithms to generate population

of modularized system architecture configurations and applied multi-objective optimization approach presented in the previous section.

In this section, the selection of complexity and modularity metric, formulation of multi-objective optimization problem, and description of system model creation and modular system concept generation are introduced. The proposed framework is demonstrated using a real-life complex system example, which is optimized for structural complexity and modularity.

4 Case study: train bogie

4.1 Overview

The train bogie system is the chassis in the railroad locomotive. It is responsible for supporting the train body, keeping the train on railroad track, and provides ride comfort through suspension's noise, vibration and harshness control. The bogie system used in the case study was specifically designed for the tilting train conceived by Korean Railroad Research Institute (KRRI) Consortium. When the train corners a curve, the train body as well as passengers inside it shift towards the outer side of the curve due to inertia. To prevent the train from going off the track, the train is forced to travel at a slower speed, increasing the overall travel time. To alleviate this problem, a tilting mechanism is added to the bogie to counter the inertia force. By counteracting the force, the train can travel through curves faster, while the inertia force felt by the passengers is minimized. Figure 5 shows the train, the computer-aided design (CAD) representation of a bogie system, and the constructed DSM model of the bogie system shown. In this case study, the train bogie system shown will be used to demonstrate the optimization framework introduced.

4.2 Bogie system model (DSM) creation

To construct the DSM model of the train bogie, relevant engineering information was collected. The engineering firm that designed and developed the particular bogie system provided the engineering data. Complete bill of material (BOM), bogie installation drawing and module drawings were provided to aid the DSM construction. Table 2 shows the list of modules and number of components for all modules.

There are 24 modules that make up the complete train bogie with 153 components in total. It should be noted that some modules are symmetric copies of the other about the center of the train, with left side and right side being symmetric. There are modules with just one component

listed. This is due to the engineering firm delegating the actual designs to another supplier and receives specified subsystems in one piece. In this case study, these modules are treated as modules with a single component.

The next task is to construct the actual DSM for the entire bogie system, comprising current modules and components listed in Table 2. This was accomplished using the bogie system installation drawing and individual module engineering drawings, which show connective relationship between components. Resulting DSM of the bogie system is shown in Fig. 5 with 153 rows and columns, each row and corresponding column representing 153 components in the bogie. In the figure, a bold box marks each module. Connections between components are represented by black marks. To turn this into analysis-ready system model cells with connections (marked black) are filled with number "1" while other cells are filled with number "0".

In addition to information obtained from engineering drawings, the engineering firm provided individual complexity rating for all components shown in the DSM. A rating scale between 1 and 5 was used to rate complexity of a component, 1 being the least difficult to design, and 5 being the most difficult. Due to symmetry in design, nearly identical components were used in front and rear of the bogie (or on the right and left sides of the bogie). For these components with multiple instances, individual component complexity value was divided by the number of instances of that component used in the bogie system. If an axle bearing assembly has complexity of 5, and if 4 of them are used to complete the system, the complexity of each assembly is assigned a component complexity of 1.25. This way, the total component complexity is conserved and estimation of interface complexity is not adversely impacted. Table 3 shows examples of component complexity allocation.

Having estimated the component complexities ($\alpha_i \in [1,5]$), the next step is to estimate the interface complexities (β_{ij}) for each interface in the bogie system model. The interface complexities depend on the associated component complexities and are estimated using the following relationship:

$$\beta_{ij} = 0.1 * \max(\alpha_i, \alpha_j).$$

The proposed interface complexity estimation process is depicted in Table 4 below.

It can be noted from the above description that the range of interface complexity lies in [ε, 0.5], where $\varepsilon = 0.1*\min(\text{MaxList})$. Once the component and interface complexities are estimated, we have the overall estimates for the component and interface complexity components (C_1, C_2) to be plugged into Eq. (1) for estimating the
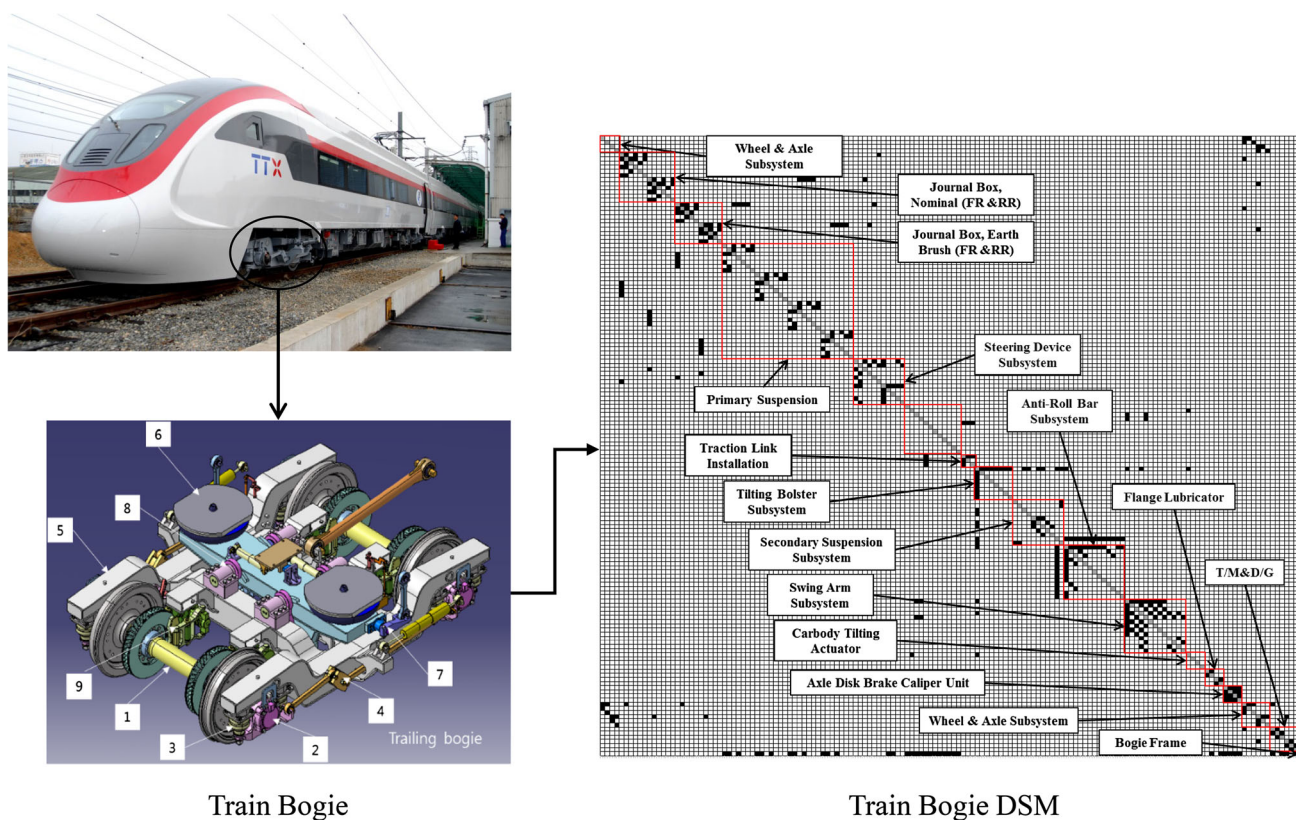
Train Bogie   Train Bogie DSM

Fig. 5 Train, CAD representation of a train bogie system and corresponding DSM model [train picture and CAD representation provided from Korean Railroad Research Institute (KRRI) with permission]

structural complexity of the bogie system and related bookkeeping.

4.3 System optimization and results

Table 5 shows module-level structural complexity distribution for each module, along with the bogie system level modularity and overall structural complexity values. The values of integrative complexity (IC) and its normalized counterpart are also provided at the bottom of the table. The integrative complexity is reflective of the amount of integration effort that one might expect at the time of system integration/assembly.

The current system decomposition strategy divides the bogie system into 24 modules. Based on this decomposition, the resulting degree of modularity, Q, is 0.56, indicating a system that is not very modular. We applied modularity maximization strategy used by Blondel et al. (2008) to find out the system decomposition that leads to maximization of Q. This resulted in decomposition strategy which divides the bogie system into 11 modules. This increased the degree of modularity by more than 30%, from 0.56 to 0.74. A DSM view of the modularity-optimized decomposition for the bogie system is shown in Fig. 6, and the corresponding

module complexity values and other system metrics are shown in Table 6. Observing the results, the integrative complexity (IC) for the bogie system has dropped from a value of 22.93 for the original decomposition to 9.16 for the modularity maximizing decomposition.

It is interesting to note the changes in distribution of module complexity in this case as compared to the current system decomposition. The modularity maximizing decomposition led to creation of modules with a larger dispersion of module complexity, with three most complex modules taking up a large share of the total module complexity. The standard deviation of module-level complexity distribution has gone up from 4.86 (in case of original decomposition) to 8.32 for the modularity maximizing system decomposition. Modular decomposition structure shown in Fig. 6 and module-level complexity distribution is shown in Table 6 for the modularity maximizing decomposition on the Pareto front, which has 11 modules, as labeled in Fig. 7. One can notice from the table that modularity maximizing decomposition has led to smaller number of highly complex modules (i.e., module 1 and 2 in this case). This asymmetric distribution of module-level complexity can have serious impact on the system development planning and effort where modules with larger

Table 2 Current Bogie modules and number of components (based on BOM)

Number	Module name	Number of components
1	Wheel and axle module—front	5
2	Wheel and axle module—rear	5
3	Journal box module—nominal	6
4	Journal box module (anti-skid)—front	10
5	Journal box module (anti-skid)—rear	10
6	Journal box module—earth brush	5
7	Primary suspension module—front left	7
8	Primary suspension module—front right	7
9	Primary suspension module—rear left	7
10	Primary suspension module—rear right	7
11	Steering device module	12
12	Bogie frame module	12
13	Traction link module	3
14	Tilting bolster module	8
15	Secondary suspension module	11
16	Anti-roll bar module	13
17	Swing arm module	13
18	Carbody tilting actuator module	4
19	Flange lubricator module—left	2
20	Flange lubricator module—right	2
21	Axle disk brake caliper module—front left	1
22	Axle disk brake caliper module—front right	1
23	Axle disk brake caliper module—rear left	1
24	Axle disk brake caliper module—rear right	1
	Total	153

Table 3 Individual component complexity rating and allocation among same components used throughout the bogie system

Component name	Component complexity rating	Quantity used	Resulting component complexity in DSM (α_i)
Axle bearing assembly	5	4	1.25
…	…	…	…
Traction link bracket	2	1	2

Table 4 Component complexity

Component 1 (α_i)	Component 2 (α_j)	MaxList [max(α_i, α_j)]
2	1	2
…	…	…
1	3	3

complexity require increased attention and possibly lead to schedule slippage in the system development process.

To address this problem, module-level complexity distribution should be balanced (i.e., relative uniformity in module-level complexity). What this means is that the standard deviation of complexity values for all modules (Sc) must be minimized, while the overall modularity (Q) of the bogie system is maximized.

This is formulated as a multi-objective optimization problem stated in Eq. (8), and solved using a simulated annealing-based multi-objective optimization algorithm. The generated Pareto front and a subset of the generated modular decompositions are shown in Fig. 7. It shows the optimal Pareto front with a subset of dominated points obtained by minimizing the spread of the module complexity and maximizing the modularity index. Non-dominated points are traced to show a piece-wise Pareto front, with associated annotations, for clarity. The current decomposition structure of the bogie system, which is

Table 5 Module complexity values for each module and overall system level metric values

Number	Module	Module complexity
1	Wheel and axle module—front	6.98
2	Wheel and axle module—rear	6.98
3	Journal box module—nominal	16.04
4	Journal box module (anti-skid)—front	23.06
5	Journal box module (anti-skid)—rear	22.37
6	Journal box module—earth brush	6.69
7	Primary suspension module—front left	6.69
8	Primary suspension module—front right	7.19
9	Primary suspension module—rear left	7.19
10	Primary suspension module—rear right	16.98
11	Steering device module	21.04
12	Bogie frame module	5.75
13	Traction link module	15.32
14	Tilting bolster module	23.13
15	Secondary suspension module	18.42
16	Anti-roll bar module	15.51
17	Swing arm module	19.98
18	Carbody tilting actuator module	7.00
19	Flange lubricator module—left	3.20
20	Flange lubricator module—right	3.20
21	Axle disk brake caliper module—front left	1.25
22	Axle disk brake caliper module—front right	1.25
23	Axle disk brake caliper module—rear left	1.25
24	Axle disk brake caliper module—rear right	1.25
Bogie modularity index (Q)		0.56
Standard deviation of module complexity (Sc)		5.47
Bogie structural complexity (C)		293.62
Integrative complexity (IC) ($IC_{normalized}$)		22.93 (0.52)

located very close to the Pareto front, is also plotted on the same graph.

It is observed that increasing the balance in module-level complexity distribution leads to partial destruction of modularity and vice versa. These results show that there is no free lunch and one has to look for an acceptable compromise between the two aspects. This observation also alludes to an interesting system architecting decision that one needs to take while trying to balance these two attributes. For example, if an original equipment manufacturer (OEM) has great competency in handling higher degree of integrative complexity (i.e., lower modularity), they can opt for any decomposition with simpler and balanced modules (i.e., lower and equitable module-level complexity). On the other hand, having very capable module suppliers might influence OEM's to focus on decompositions with low interface complexity (i.e., high degree of modularity). This

decision is a critical one for complexity management strategies that might be put in place during the system development process.

Another interesting observation is the negative correlation between modularity (Q) and the number of modules (k) in the decomposition shown in Fig. 8. Corresponding to a given number of modules (k), there are points with varying degree of modularity (Q). For a fixed number of modules (k), the average modularity value is computed and plotted in the figure. This negative correlation between the number of modules and modularity indicates that higher degree of complexity leads to a smaller number of highly separable modules.

Combining the observations from the multi-objective optimization study, we see that modularity maximizing decompositions tend to have smaller number of modules with a higher degree of asymmetry in module-level complexity distribution. This can lead to a likely negative
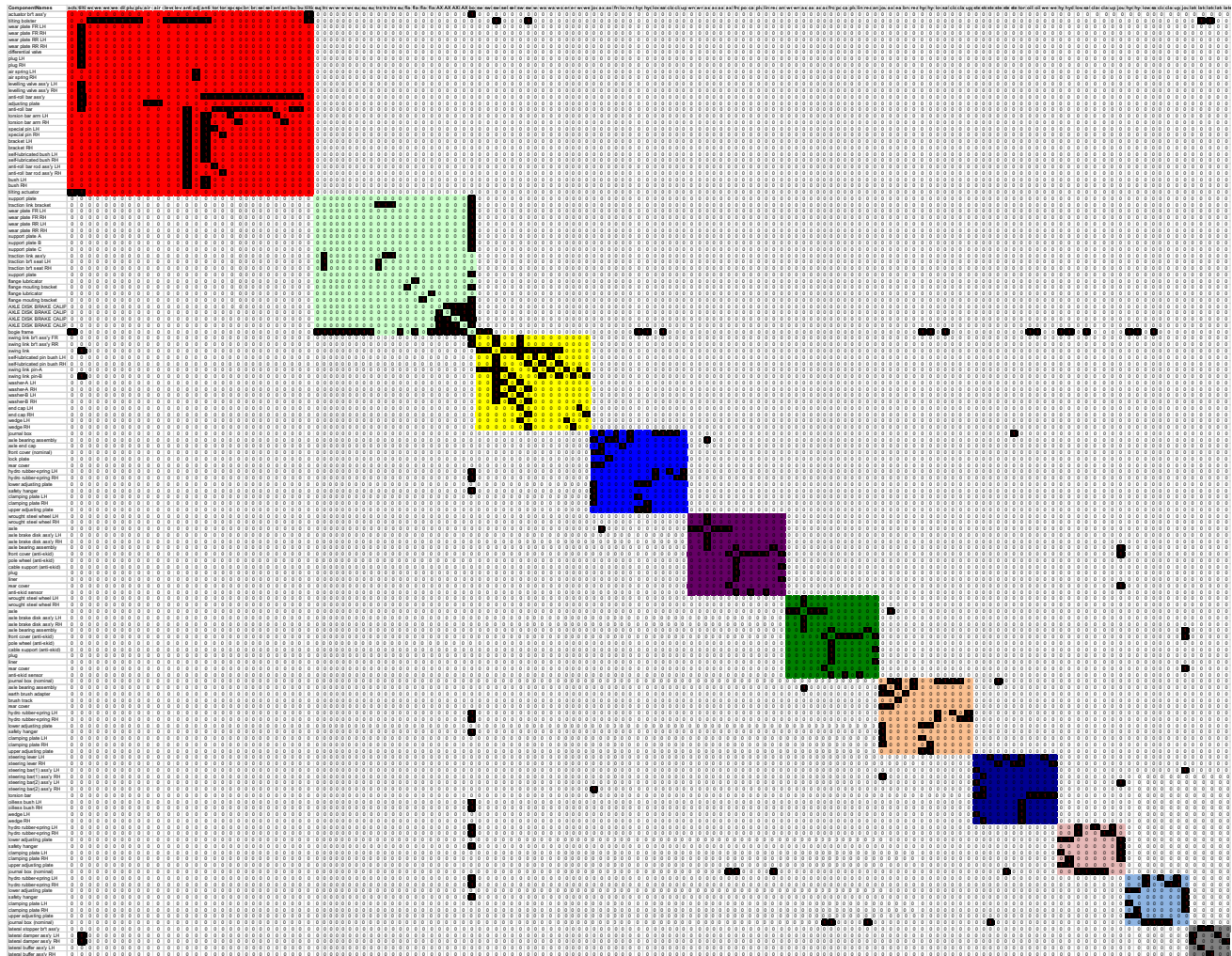
Fig. 6 Modularity maximizing decomposition ($Q = 0.74$) of the bogie system with 11 modules, resulting in more than 30% increase in degree of modularity compared to the original decomposition

impact where a few number of highly complex modules can act as bottlenecks and results in system development delay. On the other extreme, highly equitable and balanced distribution of module-level complexity can lead to increased integrative complexity (i.e., lowering of modularity).

4.4 Discussion and case summary

Pareto-optimization results shown in Fig. 7 provide valuable information regarding the modularity-complexity trade-space for complex system design. As the number of modules (k) in the system decreases, the standard deviation for module complexity (Sc) tends to increases in a superlinear manner. It means that with fewer numbers of modules, there are likely some very complex modules with large portion of overall system complexity allocated, while some modules are relatively simple, as shown on the right

side of the graphs in Fig. 7. This would lead to severe imbalance in individual module development time, which will be impeded by the more complex module. On the other hand, decomposing the system into many smaller complexity modules will balance the allocation of complexity among modules and minimize Sc, but at the cost of reduced system modularity (i.e., results in higher integrative complexity).

Assessing system decompositions that lies on the Pareto front requires evaluation from several viewpoints. First viewpoint is that of the system design and development team, whose main task is the initial system design development, mapping key system functions to forms. One of the key issues from this viewpoint is that each module in the system is responsible to perform a specific function, which enables coherent initial design and ease of system upgrade when a new technology is developed to enhance that specific function. This will influence the choice of

Table 6 Module complexity values for each module and other system level metric values with a new decomposition strategy

Number	Subsystems	Module complexity
1	Module 1	60.43
2	Module 2	43.96
3	Module 3	24.77
4	Module 4	25.43
5	Module 5	26.28
6	Module 6	22.36
7	Module 7	20.54
8	Module 8	14.16
9	Module 9	10.68
10	Module 10	14.93
11	Module 11	10.1
Bogie modularity index (Q)		0.74
Standard deviation of module complexity (Sc)		12.58
Bogie structural complexity (C)		293.62
Integrative complexity (IC) (IC$_{normalized}$)		9.16 (0.26)

design from the Pareto front. The second viewpoint is that of the system production team, responsible for the initial system assembly. In Figs. 7 and 8, system decompositions with fewer modules were shown to be more modular compared to designs with larger number of modules. This results in smaller integrative complexity, which can have a positive impact on system assembly time, influencing the variable production cost of the system.

The final viewpoint is related to the system operation and maintenance team, responsible for operating and maintaining the system after it is deployed in the field. In some instances, modules consist of collection of components that have roughly equivalent maintenance interval. This allows ease of system inspection and maintenance, minimizing the number of modules to inspect and maintain per scheduled maintenance. When selecting candidate modular design, these factors, along with associated cost data, should be taken into consideration. All these viewpoints should be taken into consideration when the choice of final design is made.

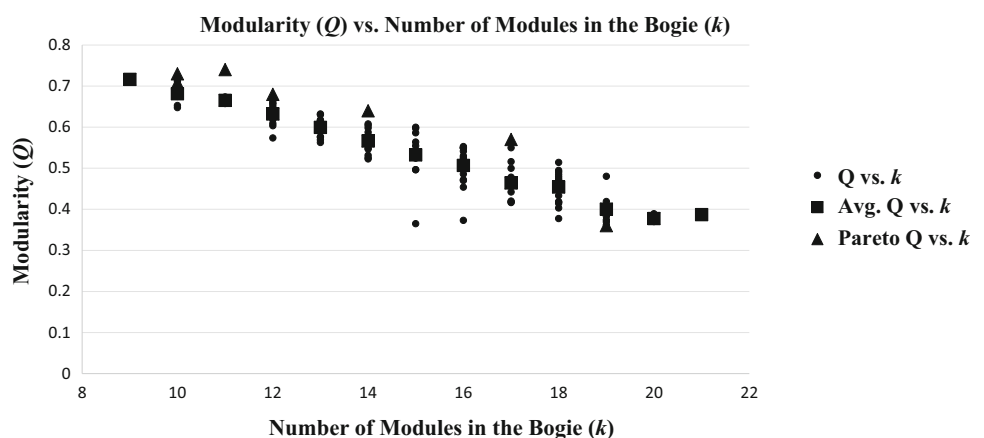**Fig. 7** Pareto front of generated bogie modularization configurations

Moduarity (Q) vs. Standard Deviation of Module to Module Complexity (Sc)
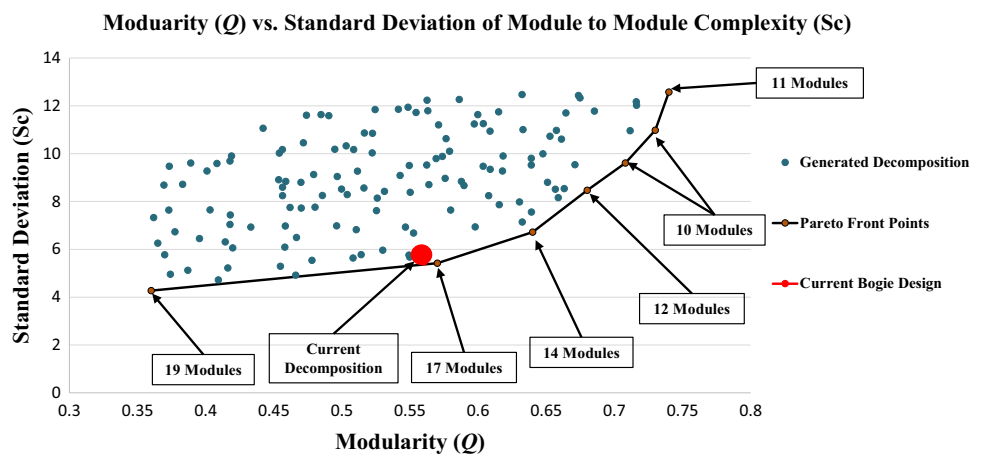
Fig. 8 Relationship between the number of modules in the bogie and corresponding modularity metric

Modularity (Q) vs. Number of Modules in the Bogie (k)

In this section, the proposed multi-objective optimization framework is demonstrated through an actual complex system case study, where a train bogie was used to create optimum modular designs, while attempting to minimize the standard deviation between individual modules. Using the optimization algorithm, a Pareto front was generated, which showed the trade-off between the degree of modularity and standard deviation of module to module complexity. From these modular designs generated, the system architect can assess designs on Pareto front from different perspectives to select top candidates for further concept evaluation.

5 Conclusion and future work

In this paper, we presented a multi-objective optimization framework that minimizes the variation in complexity allocation among individual modules, while maximizing the overall degree of modularity of the system. The optimization framework was applied to a case study, where a train bogie system was optimized for structural complexity allocation among individual modules while maximizing for overall system modularity. It was shown that the modularity maximizing decomposition induced a large variation in module-level complexity variation, while more balanced complexity distribution among modules leads to erosion of modularity.

The optimization framework developed and demonstrated in this paper can be used by system architects to assess their initial system design concepts in terms of system complexity allocation and overall modularity, allowing evaluation of pros and cons of trading one system property for another. Additionally, these assessments can be evaluated in connection with associated costs, which are influenced by system complexity allocation and the overall system modularity. The multi-objective optimization framework linking modularity and module-level complexity distribution are demonstrated through a real-world complex system case study, where a train bogie was used to create system decompositions that maximize modularity, while attempting to minimize the standard deviation of module-level complexity distribution. The resulting Pareto front showed the trade-off between the degree of modularity and standard deviation of module-level complexity. From these modular decompositions generated, the system architect can assess decompositions on Pareto front, representing optimal trade-offs, from multiple system design perspectives to select top candidates for further concept evaluation.

There are numerous possibilities for future work which can lead to very interesting research in the area of architectural complexity management and modular system design. One topic that was not explored in this paper was the implementation of constraints. Due to constraints in engineering design, such as separation of components due to heat generation, packaging, or other customer and system imposed restrictions, the resulting constrained optimal Pareto front might be different. This is an important topic to explore and bring the proposed framework closer to realities of real-world engineering design. Another topic is the establishment of relationship between module complexities and its corresponding design time or effort. If this relationship can be established, individual module design time/effort can be estimated as function of its structural complexity, thus linking the abstract complexity metric to the actual engineering design time/effort. Lastly, how does integrative complexity relate to the overall system modularity? Are there some meaningful relationships between these two aspects of system architecting? All these aspects are research topics for future work, which can advance the state-of-the art in complex system design in terms of complexity and modularity management.

References

Alexander C (1964) Notes on the synthesis of form. Harvard University Press, Cambridge

Allaire D, He QX, Deyst J, Willcox K (2012) An information-theoretic metric of system complexity with application to engineering system design. J Mech Des. doi:10.1115/1.4007587

Ameri F, Summers J, Mocko G, Porter M (2008) Engineering design complexity: an investigation of methods and measures. Res Eng Des 19:161–179. doi:10.1007/s00163-008-0053-2

Arena MV, Younossi O, Brancato K, Blickstein I, Grammich CA (2008) Why has the cost of fixed-wing aircraft risen? RAND Corporation, Santa Monica, CA. http://www.dtic.mil/get-tr-doc/pdf?AD=ADA485503. Accessed 27 June 2017

Asikoglu O, Simpson TW (2010) A new approach for evaluating design dependencies in product architectures. In: 13th AIAA/ISSMO multidisciplinary analysis optimization conference, Fort Worth, TX, p 9028

Augustine NR (1997) Augustine's laws, 6th edn. American Institute of Aeronautics and Astronautics, Reston

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. doi:10.1109/Tevc.2007.900837

Barton JA, Love DM, Taylor GD (2001) Design determines 70% of cost? A review of implications for design evaluation. J Eng Des 12:47–58

Bashir HA, Thomson V (1999) Estimating design complexity. J Eng Des 10:247–257

Bearden DA (2003) A complexity-based risk assessment of low-cost planetary missions: when is a mission too fast and too cheap? Acta Astronaut 52:371–379. doi:10.1016/S0094-5765(02)00177-7

Bellmann J, Knorr A, Schomaker R (2011) Factors influencing time and cost overruns in aircraft projects. Aerlines Mag Online J Air Transp Aviat Bus Stud Prof 18:1–4

Bernstein DS (2009) Matrix mathematics : theory, facts, and formulas, 2nd edn. Princeton University Press, Princeton

Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J StatMech Theory Exp 2008(10):P10008

Braha D, Bar-Yam Y (2007) The statistical mechanics of complex product development: empirical and analytical results. Manag Sci 53:1127–1145. doi:10.1287/mnsc.1060.0617

Braha D, Maimon O (1998) The measurement of a design structural and functional complexity. Ieee Trans Syst Man Cyber A 28:527–535. doi:10.1109/3468.686715

Bralla JG (1986) Handbook of product design for manufacturing : a practical guide to low-cost production. McGraw-Hill, New York

Chen L, Li S (2005) Analysis of decomposability and complexity for design problems in the context of decomposition. J Mech Des 127:545–557. doi:10.1115/1.1897405

Conway ME (1968) How do committees invent? Datamation 14:28–31

Dehmer M, Emmert-Streib F (eds) (2009) Analysis of complex networks: from biology to linguistics. Wiley, Weinheim. ISBN: 978-3-527-32345-6

Dolan B, Lewis K (2008) Robust product family consolidation and selection. J Eng Des 19:553–569

Engel A, Browning TR (2008) Designing systems for adaptability by means of architecture options. Syst Eng 11:125–146. doi:10.1002/sys.20090

Engel A, Reich Y (2015) Advancing architecture options theory: six industrial case studies. Syst Eng 18:396–414. doi:10.1002/sys.21312

Engel A, Browning TR, Reich Y (2016) Designing products for adaptability: insights from four industrial cases. Decis Sci. doi:10.1111/deci.12254

Eppinger SD, Browning TR (2012) Design structure matrix methods and applications. MIT Press, Cambridge

Ericsson A, Erixon G (1999) Controlling design variants: modular product platforms. Society of Manufacturing Engineers, Dearborn

Frey D, Palladino J, Sullivan J, Atherton M (2007) Part count and design of robust systems. Syst Eng 10:203–221. doi:10.1002/sys.20071

Guo F, Gershenson JK (2004) A comparison of modular product design methods based on improvement and iteration. In: ASME international design engineering technical conferences, DETC2004-57396, Salt Lake City, Utah

Gutman I (2001) The energy of a graph: old and new results. In: Betten A, Kohnert A, Laue R, Wassermann A (eds) Algebraic combinatorics and applications. Springer, Berlin, Heidelberg, 196–211

Hirschi NW, Frey DD (2002) Cognition and complexity: an experiment on the effect of coupling in parameter design. Res Eng Des Theory Appl Concurr Eng 13:123–131. doi:10.1007/s00163-002-0011-3

Holland JH (1992) Genetic algorithms. Sci Am 267:66–72

Holtta-Otto K, de Weck O (2007) Degree of modularity in engineering systems and products with technical and business constraints. Concurr Eng Res Appl 15:113–126. doi:10.1177/1063293x07078931

Holtta-Otto K, Chiriac NA, Lysy D, Suh ES (2012) Comparative analysis of coupling modularity metrics. J Eng Des 23:787–803. doi:10.1080/09544828.2012.701728

Horn RA, Johnson CR (1991) Topics in matrix analysis. Cambridge University Press, Cambridge

Hornby GS (2007) Modularity, reuse, and hierarchy: measuring complexity by measuring structure and organization. Complexity 13:50–61. doi:10.1002/cplx.20202

Hozo SP, Djulbegovic B, Hozo I (2005) Estimating the mean and variance from the median, range, and the size of a sample. BMC Med Res Methodol 5:13. doi:10.1186/1471-2288-5-13

Ingber L (1996) Adaptive simulated annealing (ASA): lessons learned. Control Cybern 25:33–54

Kafura D, Henry S (1981) Software quality metrics based on interconnectivity. J Syst Softw 2:121–131. doi:10.1016/0164-1212(81)90032-7

Kim G, Kwon Y, Suh ES, Ahn J (2016) Analysis of architectural complexity for product family and platform. J Mech Des. doi:10.1115/1.4033504

Kim G, Kwon Y, Suh ES, Ahn J (2017). Correlation between architectural complexity of engineering systems and actual system design effort. J Mech Des. doi:10.1115/1.4035319

Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680. doi:10.1126/science.220.4598.671

Kortler S, Kreimeyer M, Lindemann U (2009) A planarity-based complexity metric. In: The 17th international conference on engineering design, Palo Alto, CA, USA

Lindemann U, Maurer M, Braun T (2009) Structural complexity management : an approach for the field of product design. Springer, Berlin

MacCormack A, Baldwin C, Rusnak J (2012) Exploring the duality between product and organizational architectures: a test of the "mirroring" hypothesis. Res Policy 41:1309–1324. doi:10.1016/j.respol.2012.04.011

Maier MW, Rechtin E (2009) The art of systems architecting, 3rd edn. CRC Press, Boca Raton

Maimon O, Braha D (1996) On the complexity of the design synthesis problem Ieee Trans Syst Man Cybern A 26:142–151. doi:10.1109/3468.477869

Malik F (1992) Strategie des managements komplexer système. Bern, Stuttgart, Wien: Haupt

Martin MV, Ishii K (2002) Design for variety: developing standardized and modularized product platform architectures. Res Eng Des Theory Appl Concurr Eng 13:213–235. doi:10.1007/S00163-002-0020-2

Mathieson JL, Summers JD (2010) Complexity metrics for directional node-link system representations: theory and applications. In: ASME international design engineering technical conferences, DETC2010-28561, Montreal, Canada

McCabe TJ (1976) A complexity measure. IEEE Trans Softw Eng 2(4):308–320

Mikkola JH, Gassmann O (2003) Managing modularity of product architectures: toward an integrated theory. Ieee Trans Eng Manag 50:204–218. doi:10.1109/Tem.2003.810826

Min G, Suh ES, Holtta-Otto K (2015) Impact of technology infusion on system complexity and modularity. In: ASME international design engineering technical conferences, DETC2015-46547, Boston, MA

Min G, Suh ES, Holtta-Otto K (2016) System architecture, level of decomposition, and structural complexity: analysis and observations. J Mech Des. doi:10.1115/1.4032091

Mood AM, Graybill FA, Boes DC (1973) Introduction to the theory of statistics. McGraw-Hill series in probability and statistics, 3rd edn. McGraw-Hill, New York

Newman MEJ (2010) Networks: an introduction. Oxford University Press, Oxford

Nikiforov V (2007) The energy of graphs and matrices. J Math Anal Appl 326:1472–1475. doi:10.1016/j.jmaa.2006.03.072

Otto K, Höltä-Otto K, Simpson TW, Krause D, Ripperda S, Ki Moon S (2016) Global views on modular design research: linking alternative methods to support modular product family concept development. J Mech Des 138:071101. doi:10.1115/1.4033654

Pahl G, Beitz W (1996) Engineering design: a systematic approach. Springer, London

Riedl R (2013) Strukturen der Komplexität: eine Morphologie des Erkennens und Erklärens. Springer, Berlin

Sheard SA, Mostashari A (2009) Principles of complex systems for systems engineering. Syst Eng 12:295–311. doi:10.1002/sys.20124

Simon HA (1996) The sciences of the artificial, 3rd edn. MIT Press, Cambridge

Sinha K (2014) Structural complexity and its implications for design of cyber physical systems. Dissertation, Massachusetts Institute of Technology, Cambridge

Sinha K, de Weck O (2013) Structural complexity quantification for engineered complex systems and implications on system architecture and design. In: ASME international design engineering technical conferences, DETC2013-12013, Portland, OR

Sinha K, Suh ES, de Weck O (2017) Correlating integrative complexity with system modularity. In: ASME international design engineering technical conferences, DETC2017-67254, Cleveland, OH

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. doi:10.1115/1.1564074

Suh ES, Kott G (2010) Reconfigurable parallel printing system design for field performance and service improvement. J Mech Des 132. doi:10.1115/1.4000961

Suh ES, de Weck O, Chang D (2007) Flexible product platforms: framework and case study. Res Eng Des 18:67–89. doi:10.1007/s00163-007-0032-z

Sussman J (2000) Ideas on complexity in systems—twenty views. Massachusetts Institute of Technology. http://web.mit.edu/esd.83/www/notebook/20ViewsComplexity.PDF. Accessed 24 June 2017

Tamaskar S, Neema K, DeLaurentis D (2014) Framework for measuring complexity of aerospace systems. Res Eng Des 25:125–137. doi:10.1007/s00163-014-0169-5

Ulrich KT, Eppinger SD (2012) Product design and development, 5th edn. McGraw-Hill/Irwin, New York

Weber C (2005) What is complexity? In: Proceedings of ICED 2005, the 15th international conference on engineering design, Melbourne, Australia

Weyuker EJ (1988) Evaluating software complexity measures. IEEE Trans Softw Eng 14:1357–1365. doi:10.1109/32.6178

Whitney DE, Dong Q, Judson J, Mascoli G (1999) Introducing knowledge-based engineering into an interconnected product development process. In: ASME international design engineering technical conferences, Las Vegas, NV

Yassine AA, Naoum-Sawaya J (2017) Architecture, performance, and investment in product development networks. J Mech Des. doi:10.1115/1.4034673

Yassine AA, Joglekar N, Braha D, Eppinger S, Whitney D (2003) Information hiding in product development: the design churn effect. Res Eng Des Theory Appl Concurr Eng 14:145–161. doi:10.1007/s00163-003-0036-2

Yu T-L, Yassine AA, Goldberg DE (2007) An information theoretic method for developing modular architectures using genetic algorithms. Res Eng Des 18:91–109