Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2014), *28*, 323–337.
© Cambridge University Press 2014 0890-0604/14 $25.00
© Massachusetts Institute of Technology 2014
doi:10.1017/S089006041400050X

Improving the systems engineering process with multilevel analysis of interactions

STEVEN D. EPPINGER,[1] NITIN R. JOGLEKAR,[2] ALISON OLECHOWSKI,[1] AND TERENCE TEO[1]
[1]Massachusetts Institute of Technology, Cambridge, Massachusetts, USA
[2]Boston University School of Management, Boston, Massachusetts, USA

(RECEIVED February 24, 2014; ACCEPTED May 5, 2014)

Abstract

The systems engineering V (SE-V) is an established process model to guide the development of complex engineering projects (INCOSE, 2011). The SE-V process involves decomposition and integration of system elements through a sequence of tasks that produce both a system design and its testing specifications, followed by successive levels of build, integration, and test activities. This paper presents a method to improve SE-V implementation by mapping multilevel data into design structure matrix (DSM) models. DSM is a representation methodology for identifying interactions between either components or tasks associated with a complex engineering project (Eppinger & Browning, 2012). Multilevel refers to SE-V data on complex interactions that are germane either at multiple levels of analysis (e.g., component versus subsystem) conducted either within a single phase or across multiple time phases (e.g., early or late in the SE-V process). This method extends conventional DSM representation schema by incorporating multilevel test coverage data as vectors into the off-diagonal cells. These vectors provide a richer description of potential interactions between product architecture and SE-V integration test tasks than conventional domain mapping matrices. We illustrate this method with data from a complex engineering project in the offshore oil industry. Data analysis identifies potential for unanticipated outcomes based on incomplete coverage of SE-V interactions during integration tests. In addition, assessment of multilevel features using maximum and minimum function queries isolates all the interfaces that are associated with either early or late revelations of integration risks based on the planned suite of SE-V integration tests.

Keywords: Complexity; Design for Testability; Design Structure Matrix; System Integration; Systems Engineering

1. INTRODUCTION

Complex engineering involves multiple types of data and contexts. Figure 1 presents a stylized map of phases in a systems engineering V (SE-V) process for managing such a project. The SE-V consists of conceptual development, preliminary system design, detailed design, construction, integration testing and validation, startup, operations, and expansion phases (Blanchard & Fabrycky, 1990). An investment decision after the preliminary system design phase results in the commitment of capital to execute detailed design, procurement, construction, testing, validation, and startup activities (Halman & Braks, 1999). Hence, decision makers explore ways in which cost, performance, and the impact of downstream integration tasks and risks (the right side of the SE-V) can be examined early, that is, during the decomposition stage (the left side of the SE-V).

Reprint requests to: Steven D. Eppinger, Massachusetts Institute of Technology, Sloan School of Management, 77 Massachusetts Avenue, Room E62-468, Cambridge, MA 02139, USA. E-mail: eppinger@mit.edu

The complexity in SE-V propagates through a sequence of changes (Giffin et al., 2009). These changes have been associated with the large number of interactions involved within the system, the need for learning, and the fact that these systems involve decisions at multiple levels. Allied literature used the term "multiscale" to capture robust design involving multiple level or time phases of decisions (Reich, 1998; Allen et al., 2006; Zha et al., 2008). We will use the term "multilevel" to identify decisions that involve two or more decomposition levels (e.g., components vs. subsystems) and/or differing time scales (e.g., time gaps and sequencing needed across preliminary system design, detailed design, and integration tests). Multilevel decisions result in poorly understood interactions. These interactions can lead to negative consequences such as cost overruns, poor startup or operational performance, and even propagation of failures (Lewis, 2012; Papakonstantinou et al., 2012).

The design structure matrix (DSM) methodology has made many contributions toward improving complex decisions involving choice of product, process, and organizational archi-

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

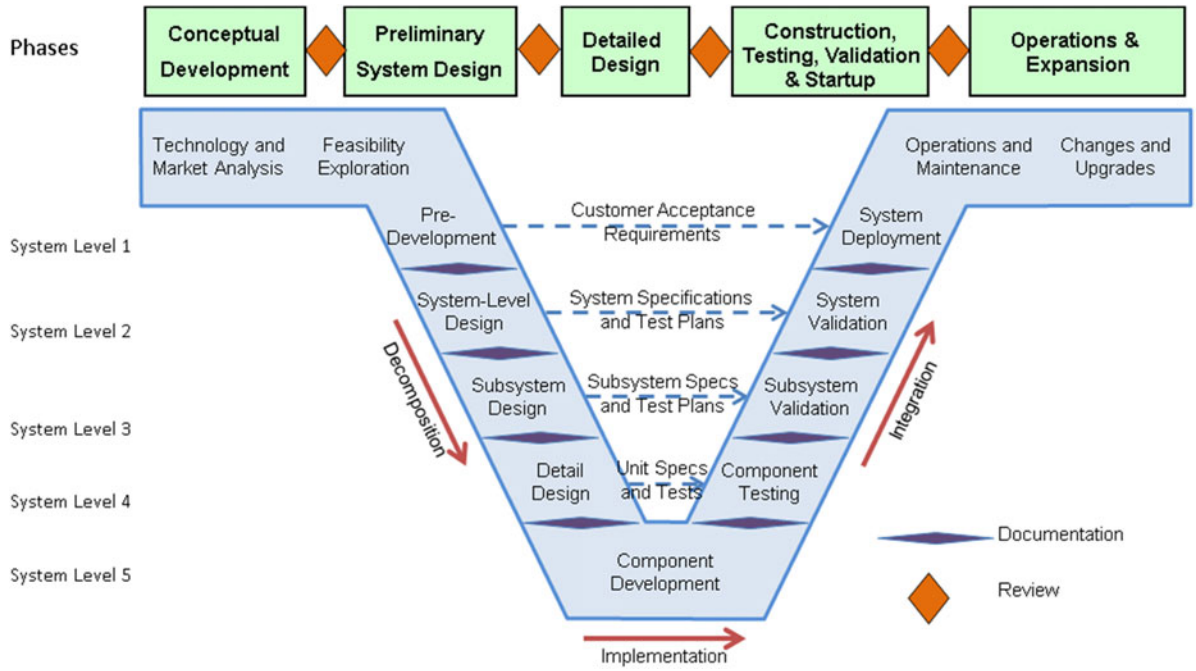


Fig. 1. Phases and levels within a systems engineering V process.

tectures during the decomposition of systems on the left-hand side of the SE-V (Eppinger & Browning, 2012). The SE-V diagram in Figure 1 indicates that a complex systems engineering process involves five levels of decomposition (concept development, system-level design, subsystem design, detailed design, and component development), specification, and integration testing. At each level of the system, the twin outcomes of a decomposition task are selection of the architecture for the next level of design and the specification for the corresponding integration tasks (shown by the horizontal dashed arrows in the SE-V). The goal of our research is to build relevant *multilevel maps* of DSMs involving integration tasks and corresponding component decomposition dependencies and to examine whether analysis of such maps can provide engineering managers with insights to improve the system integration process.

This paper presents a method to account for multilevel data in the analysis of dependencies using DSM models. This method contributes to the DSM literature (Browning, 2001) by extending representation schema that incorporate multilevel and multiple-timescale test coverage data as vectors into the off-diagonal DSM cells. These vectors provide a detailed mapping between the product architecture and the SE-V integration test tasks. This mapping is richer than conventional domain mapping matrices (DMMs; see Danilovic & Browning, 2007). We report on the collection of a preliminary data set and multilevel analysis of 374 interactions related to a complex offshore oil industry project. Results indicate potential for unanticipated outcomes in terms of incomplete coverage of SE-V integration tasks. We also show that accounting for multilevel features using maximum and minimum function queries, readily identifies all the design interfaces associated with early and late revelations of coverage risks based on a selected suite of integration test tasks. Finally, we discuss theoretical and applied implications of the findings.

2. FORMULATION

2.1. Integration and testing for system failures

Failures, sometimes of the most glaring and consequential nature, can and do occur at the boundaries or interfaces between elements. These failures have often been ascribed to uncontrolled, unanticipated, and unwanted interactions between elements: in many cases between elements thought to be entirely separate (Griffin, 2010). For instance, based on an in-depth case study of errors in the Italian air force, Leveson et al. (2009) have argued that "emergent safety properties are controlled or enforced by a set of safety constraints related to the behavior of the system components. Accidents result from interactions among system components that violate these constraints—in other words, from a lack of appropriate and effective constraints on component and system behavior." One goal of integration is to identify and resolve potential failures of the system. Among techniques commonly used to this end is failure mode and effect analysis (Stamatis, 2003; International Electrotechnical Commission, 2006*a*). The aim of this technique is to identify not only all potential failures of the system and its parts but also the effect and the mechanism of the failure. These failures are identified based on the analysis of drawings or flowcharts of the system, an understanding of the function of the system and its component, and details of the environment in which it operates. The process

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

involves generating solutions for how to avoid and/or mitigate the effects of these failures on the system.

Alternatively, the hazard and operability study is used to identify failure risks for a given system. The identification is directed by the use of guidewords (International Electrotechnical Commission, 2001). The process involves the generation of solutions and treatments to address the identified risks. Potential causes of failure can be identified and understood using a fault tree analysis, whereby various failure factors are hierarchically organized and depicted in a tree according to their causal relationship (International Electrotechnical Commission, 2006*b*). This method is best performed when the team has a deep understanding of the system and the causes of failure. It is recommended that the team use detailed diagrams of the system as an aid in analysis. Presented as a fundamentally different accident model with an emphasis on systems theory, the systems-theoretic accident modeling and processes model focuses on controller or enforcement failures, not traditional component failures (Leveson, 2011). This method requires the analyst to conceive of the system as a control problem, and it is facilitated through the generation of the process model and control structure for the system. Some methods address failure earlier in the SE-V process, for example, the function–failure design method (Stone et al., 2005), which can be used during the conceptual design phase.

2.2. Hierarchical decomposition and composition

The SE-V process incorporates potential failure modes as constraints on components and subsystem integration based on hierarchical decomposition. The study of constraints on component and system behavior is a nontrivial problem, especially as the complexity of a system rises. Braha and Maimon (1998) have modeled the underlying design process as an automaton and proved that the managing of such a planning problem is NP-Hard. Thus, both theorists and practicing engineers look for tools to visualize and understand the dependencies between components and subsystems within a system, especially when the complexity of the system design rises. Related work draws upon managing the decomposition based on hierarchy. For instance, Albers et al. (2011) explore a "contact and channel" principle arguing that function and form emerge together during design, and therefore should be considered together in a design representation. This principle is explored in a model of the system architecture of a humanoid robot arm considering the impact of a proposed design change. Tilstra et al. (2012) have introduced an extended DSM, illustrated in the context of a screwdriver design, to quantify the degree of nesting during the development of hierarchical product architecture.

The DSM is the representation for capturing complex networks of dependencies used in this work (Eppinger & Browning, 2012). Groups of tasks associated in the SE-V (Fig. 1) are mapped into a stylized task DSM in Figure 2. Several properties of this task DSM are noteworthy. Owing to the logic of SE-V, there is a regular precedence pattern between task groups as shown by "x" marks immediately below the diagonal, where each DSM mark represents information dependency. The dotted arrows depicting information flowing from the decomposition to the integration tasks in Figure 1 result in

Domain	Level ID	Group ID	Description	A&F	DT1	DT2	DT3	DT4	CD	IT4	IT3	IT2	IT1	O&M	C1	C2	C3	C4	...	Cn
Analysis Tasks	1	A&F	T& M Analysis & Feasibility	■										?	aDMM					
Decomposition Tasks	1	DT1	Concept Development	x	■								?		dDMM					
	2	DT2	System Level Design		x	■						?								
	3	DT3	Subsystem Design			x	■				?									
	4	DT4	Detailed Design				x	■		?										
Component Tasks	5	CD	Component Development					x	■	?					cDMM					
Integration Tasks	4	IT4	Component Testing					x	x	■					iDMM					
	3	IT3	Subsystem Validation				x			x	■									
	2	IT2	System Validation			x					x	■								
	1	IT1	System Deployment		x							x	■							
Operations Tasks	1,2,3, 4 & 5	O&M	Operations & Maintenance	x									x	■	oDMM					
Component Dependecies	1,2,3, 4 & 5	C1	Component 1	aDMM	dDMM				cDMM	iDMM				oDMM	■	z				z
		C2	Component 2												z	■				
		C3	Component 3													z	■	z		
		C4	Component 4													z		■		
		...	....													z	z		■	
		Cn	Component n																z	■

Fig. 2. A multilevel design structure matrix of systems engineering V tasks and components dependencies.

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

off-diagonal marks at each level. The "?" marks represent design iterations which may occur after integration tasks. Collectively, these marks form an X-shaped set of dependencies when tasks are grouped at each level of system decomposition. The "z" marks in the component DSM represent the component and subsystem dependencies. Mark "z" is distinct from mark "x" because interactions in the component DSM represent interfaces between the system elements (captured as spatial, energy, etc.).

We define DMMs aDMM, dDMM, cDMM, iDMM, and oDMM, corresponding to linkages between the components and each of the task groups: analysis, decomposition, detailed component design, integration, and operations, respectively. The focus of this research is on the dependencies between the component architecture and the integration tasks. Thus, iDMM and corresponding task and component DSMs are highlighted with chain dotted borders. Extending this representation schema, a single DSM can capture multiple types of interaction data if each off-diagonal cell contains a vector (Browning, 2001). For instance, these data types might be spatial, information, energy, and material dimensions of component interactions (Pimmler & Eppinger, 1994). In contrast, these vectors may capture different types of task interactions (Yassine, 2004). Within this context, two types of gaps are evident in the DSM literature:

1. Conventional DMMs map the elements in one domain to another. For example, component-task DMM maps a component DSM (that captures the complex interaction in product architecture) to a task DSM (that captures the complex interaction among system integration tasks, such as subsystem validation or a subsequent system verification test). However, such DMM mappings (Danilovic & Browning, 2007) have not accounted for the amount of coverage available at each interface within the product architecture based on a selected suite of integration tasks.
2. The importance of accounting for multilevel evolution of complexity has been recognized in the complex engineering literature. For instance, the law of requisite variety (Ashby, 1956; Beer, 1975) postulates that aggregation can absorb variety, where the term variety refers to the total number of possible states of a complex system. A simple example for the application of this law is a patient in a hospital with temperature fluctuation (i.e., uncertainty) associated with fever. Aggregation of some kind is needed if the doctor is not to sit all the month staring at the thermometer. Action must be taken immediately to isolate the patient, such that the root cause of the temperature fluctuation may be explored and understood based on different units of analysis (e.g., either fluctuation in food intake or exposure to environments with different types of germs). In complex engineered systems, analogous decisions may involve situations where subsystem tests during a software development suite fail to reveal a bug, even if a test engineer suspects that a bug exists based on failure history. The test team may have to resort to higher level integration tests, with a sufficient variety of stimuli, to replicate this failure.

Based on the requisite variety law, Bar-Yam (2003) argues that "Modularity and abstraction are generalized by various forms of hierarchical and layered specification . . . these two approaches either incorrectly portray performance or behavioral relationships between the system parts or assume details can be provided at a later stage." This builds the case for taking a multilevel view of potential integration problems. Multilevel methods, such as logarithmic transformation and filtering of data, enable system design teams to understand patterns of emergent behavior as the complexity of their system rises (Simoncelli et al., 1992). For example, data analysis on system-level tests may reveal unique insights about coverage on certain components that may be missing in subsystem-level test data. Conventional DSM models have typically not aggregated, or disaggregated, product architecture and process dependency data based on their levels of decomposition.

Our premise is that both of these gaps can be addressed by appropriate data mapping and analysis at each and every interface within the product architecture DSM based on multilevel views of the SE-V process. Hence, we develop a method for data collection, query, and aggregation that accounts for differing levels of testing to examine if different types of integration risks may be evident at different times during the integration process. Integration risk in this instance refers to the potential that any interface covered by a suite of tests during the SE-V integration process may reveal a failure mode within a system design. Data associated with this method grow quickly with increase in the rank of the system DSM, the number of measurement dimensions, and the size of the integration test suite.

We have developed a vector representation scheme to capture all interactions from a suite of integration tests that are relevant to a particular DSM cell. Further, in order to isolate the contributions of multilevel analysis, we assign the interactions associated with different interface dimensions (e.g., structural vs. information interactions) at relevant levels (e.g., component vs. subsystem) with unique codes. Thus, the relevant interaction at any level can be queried, analyzed, and displayed as a DSM map. A number of multilevel data aggregation and analysis techniques, ranging from renormalization using finite element analysis to optimal control, have been reported in the literature (Bar-Yam, 2006; Weinan et al., 2007; Hartmann et al., 2013). Many of these multilevel implementations have been limited to either stylized data or small-scale problems. In our case, we have implemented multilevel analysis in a complex DSM context using maximum and minimum value filters in Section 4.

3. RESEARCH CONTEXT AND DATA

We are working with a research sponsor in the offshore petroleum industry to study a deepwater development project,

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

with focus on the blowout preventer (BOP). The primary function of the BOP is to manage well pressure during drilling by completely sealing off the well bore and circulating out the influx in the event of high-pressure hydrocarbons entering the drill hole. Data collection was performed in three stages. First, we assembled data to create the system architecture DSM. Second, we collected data regarding integration testing. Third, we documented interactions in the system architecture DSM that were tested in each type of integration test. Data were collected over a period of 3 months based on review of engineering documentation and onsite interviews with subject matter experts. These onsite interviews were conducted during 2 weeklong visits. These interviews were followed by e-mail and phone conversations to clarify open issues. Each interface included in the data set was reviewed through this process. Our experience with this data collection process, and allied literature (Whitney et al., 1999), indicates that DSMs are sparsely populated and the size of the data collection scales linearly with the rank of the DSM. In order to manage the data collection effort, some of the subcomponents were grouped into a single component, based on inputs from this review.

3.1. System architecture

The BOP system architecture describes its decomposition into subsystems and components. We placed our focus on including those primary components that affect system functions and are critical for system reliability. Ancillary parts (e.g., shuttle valves, piping, and hoses) were grouped with their corresponding components. An initial list of 93 components was created based upon company and industry documentation. These 93 components were classified into eight subsystems. The component list and subsystem boundaries were reviewed with company subject matter experts. The list was refined to 67 components in the following six subsystems:

- lower marine riser package (LMRP)
- blowout preventer (BOP)
- auxiliary lines (Aux Lines)
- choke and kill system (C&K)
- hydraulic power unit
- surface control system

Because the surface control system has minimal interactions in the types of DSMs we will show, we omit this subsystem for clarity, resulting in five subsystems in our analysis here.

The next step in data collection was to identify interactions between pairs of components. We were interested in interactions in five dimensions critical to reliability and function, as advised by the subject matter experts. These dimensions are

- *spatial*, involving the physical connection or adjacency of two components;
- *structural*, involving a load or pressure-transferring interaction between two components;
- *energy*, involving the transfer of hydraulic or electrical energy between two components;
- *information*, involving the transfer of information between components by means of electrical signals or hydraulic pilot signals; and
- *materials*, involving the transfer of material (principally drilling mud, but also gas and other wellbore fluids) from one component to another.

All possible pairs of interacting components were identified using engineering documentation. These data were then reviewed with the subject matter experts. The presence of an interaction in any of the five dimensions was recorded. Interaction data are recorded on a binary scale, "0" (*no interaction*) or "1" (*required interaction*). These interaction data for each pair of components form a 67×67 system architecture DSM. We considered the five interaction dimensions separately and created a distinct DSM in each dimension. An entry of "1" indicates the presence of an interaction between the component pair, while a blank indicates a lack of interaction. Figure 3 shows the system architecture DSM for the structural dimension, including 56 of the 67 components and their interfaces. (For clarity, we omit the remaining 11 components having no interfaces in the structural dimension. DSM data showing the other four dimensions are also excluded here, for brevity.) The DSM is symmetric, because the interactions are nondirectional.

It is possible for interactions to occur within a subsystem or across subsystems. The five areas of possible subsystem interactions, occurring within blocks along the diagonal, have been shaded gray for visual clarity. For example, in Figure 3, we see that a within-subsystem interaction exists between components 1 and 2 (i.e., LMRP frame and junction box). Illustration of a cross-subsystem interaction is evident between components 6 and 24 (i.e., the pod hydraulic section within the LMRP and the pod hydraulic section receptacle within the BOP). Some subsystems have more interactions associated with them than others. In general, there are more interactions within a subsystem (in the gray areas along the diagonal) than across subsystems. In the five interaction dimensions combined, 279 interactions are within subsystems and 62 are across subsystems.

3.2. Integration tests

The second stage of data collection focused on integration test data. Company documentation was first consulted to assemble a list of 57 integration tests. Given that our DSM analysis focuses on integration issues, only tests of interactions between components and subsystems are illustrated. Tests of isolated components are excluded from the analysis in this paper. In addition, while it is possible to conduct tests using digital models, such tests are also excluded when they occur

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

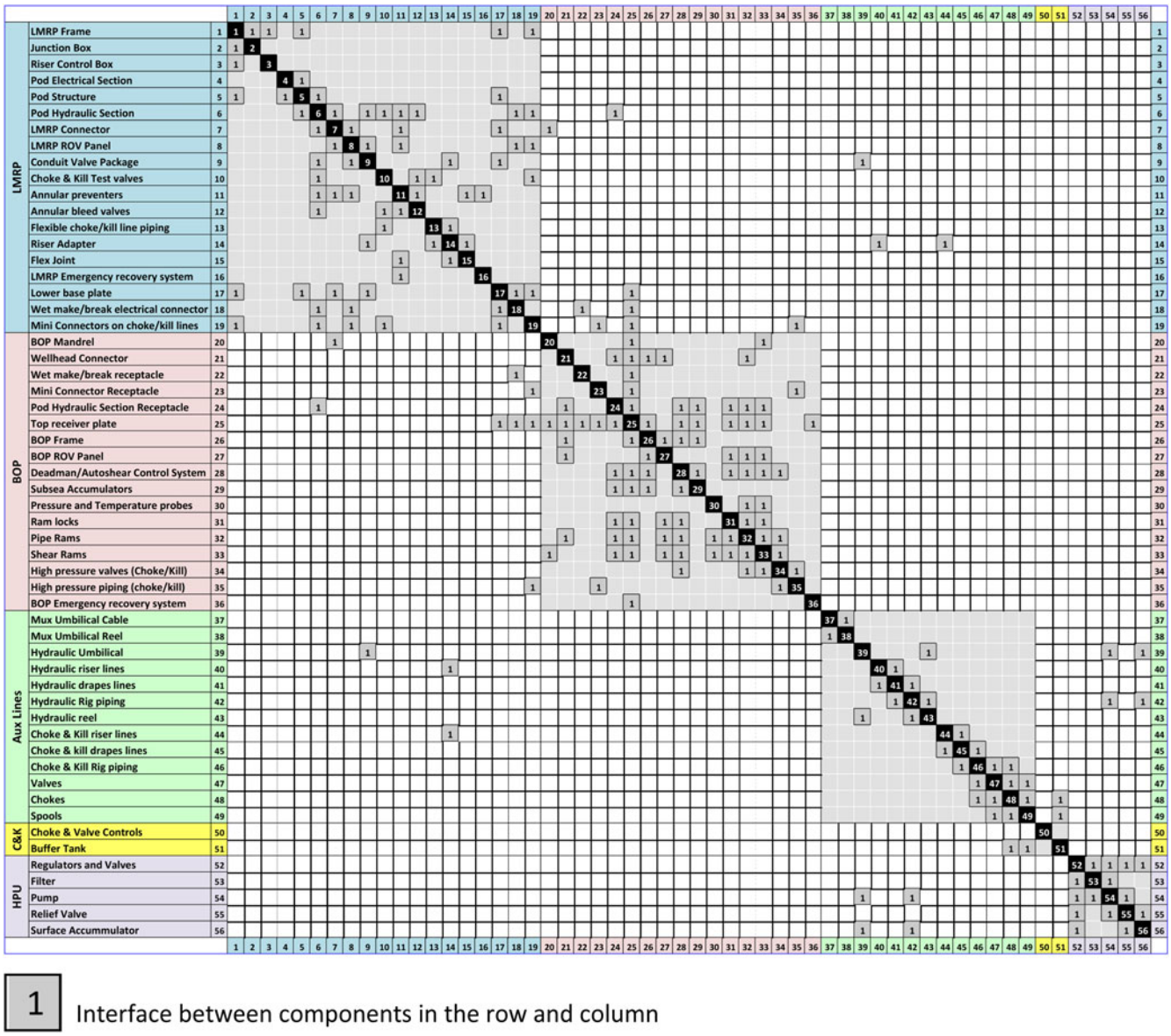


Fig. 3. System architecture design structure matrix representation of structural interactions between components. Marks in off-diagonal cells identify interfaces between components in their row and column. Five subsystems are highlighted with gray background: lower marine riser package (LMRP), blowout preventer (BOP), auxiliary lines (Aux Lines), choke and kill system (C&K), and hydraulic power unit (HPU).

during the decomposition tasks on the downside of SE-V, as opposed to the integration upside of the SE-V.

Upon consultation with the company subject matter experts, the list of integration tests was reduced to 25 tests important to system function and reliability, as presented in Table 1. It is worth noting that the data shown here are representative and not exhaustive. Thus, it is possible that a test suite in the current analysis might show that an interface is not tested, while it might be tested later through a test that is not discussed here. Each test included in this analysis falls into one of three test levels. Each subsystem is assembled and tested separately in "subsystem-level tests." Next, the full system is assembled and tested in "dock-level tests." Finally, tests of the complete system are conducted in the deployed environment in "subsea-level tests." These tests are sequenced within each level and are temporally separated. In some instances two subsystems are assembled together before the first level of testing, in which case interfaces across these two subsystems are reported to be tested at the subsystem level.

3.3. Interactions addressed by integration tests

The third stage of data collection sought to identify which interactions, and which dimensions, were tested in each of the integration tests. Each interaction–test combination was reviewed with the subject matter experts in order to identify these data.

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

Table 1. *Integration tests*

Index	Test	Level
T1	Mechanical FIT check	Subsystem
T2	Function verification check	Subsystem
T3	Actuator leak check	Subsystem
T4	Shear test	Subsystem
T5	Emergency systems test	Subsystem
T6	EDS test	Subsystem
T7	BOP pressure test	Subsystem
T8	Mechanical FIT check	Subsystem
T9	C&K control and pressure test	Subsystem
T10	HPU function an performance test	Subsystem
T11	Panel function test	Subsystem
T12	System setup and verification	Dock
T13	HPU function test	Dock
T14	Panel/BOP function test	Dock
T15	High pressure mud system test	Dock
T16	Signature test of operators (performance)	Dock
T17	Power and communications redundancy test	Dock
T18	Emergency systems test	Dock
T19	EDS test	Dock
T20	BOP pressure test	Dock
T21	BOP drift test	Dock
T22	Function test	Subsea
T23	BOP pressure test	Subsea
T24	Emergency systems test	Subsea
T25	EDS test	Subsea

Given that there are 25 tests and 374 total interactions across the five dimensions in the full data set, there was a challenging number of combinations to review. To facilitate the subject matter expert consultation process, we developed a data table where the integration tests could be mapped to the component–component interactions in an efficient manner. An example of this data input is shown in Table 2. Each test–component combination was assigned its own row in the table. There is an entry in the row corresponding to an interaction in each of the five dimensions. If no interaction exists in the corresponding dimension, the cell is shaded gray and the combination does not need to be reviewed. If the interaction does exist, the cell is white and the subject matter expert identifies whether that interaction is tested by the test under review. If it is included in the test, an "X" is marked, and if it is not, the cell is marked with an "O." For example, in Table 2, test T2, the function verification test, is a test of the valve and actuator functions within the LMRP and BOP subsystems. For this test, the spatial and structural interactions (shown by X's in Table 2) between the LMRP connector and the pod hydraulic section are tested and verified. However, T2 does not test the integrity of the connection between the LMRP and BOP; the BOP mandrel's spatial, structural, and material interactions with the LMRP connector are not tested in T2 (as shown by O's in Table 2).

3.4. Vector representation

An effective way to represent the data set is to envision a vector of 25 tests associated with each of the off-diagonal entries in the 67×67 DSM. Abstracting to a higher level, and given that each test is classified into one of three levels (subsystem, dock, or subsea), we set a three-dimensional vector behind each interaction in the DSM. We label each test level numerically; subsystem is Level 1, dock is Level 2, and subsea is Level 3. We further add details so that a vector in each DSM cell captures the integration test sequence coverage (i.e., for individual interaction, for each of 25 integration tests spread across three levels), for five types of dependency dimensions (spatial, structural, energy, information, and flow). This yields an augmented $67 \times 66 \times 25 \times 5$ (i.e., 552,750 potential interactions in the full data set, most of which are null because the matrices are sparse) vector data set that captures the multilevel complexity associated with the system development and integration test architecture.

4. RESULTS

We have explored several alternative data aggregation mechanisms to visualize these data vectors. In order to improve the ease of visualization during multilevel information comparison, we present these data by levels, and then use the maximum and minimum filters to construct maximum and minimum integration level DSMs for each of the five dimensions in our data set. For ease of exposition, we present the results on only one dimension (structural) out of the five dimensions of interactions. As explained in Section 3.1, these structural interactions are only presented for a subset of the data (that form a DSM with rank 56) instead of the full data set (with rank 67).

4.1. Interactions by levels

Figures 4 through 6 depict the results of queries by different levels in DSMs. For instance, "1" (and "0") in Figure 4 show

Table 2. *Test coverage data*

Index	Test	Component 1		Component 2		Material	Information	Energy	Structural	Spatial
T2	Function verification check	6	Pod hydraulic section	7	LMRP connector				X	X
T2	Function verification check	20	BOP mandrel	7	LMRP connector	O			O	O

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

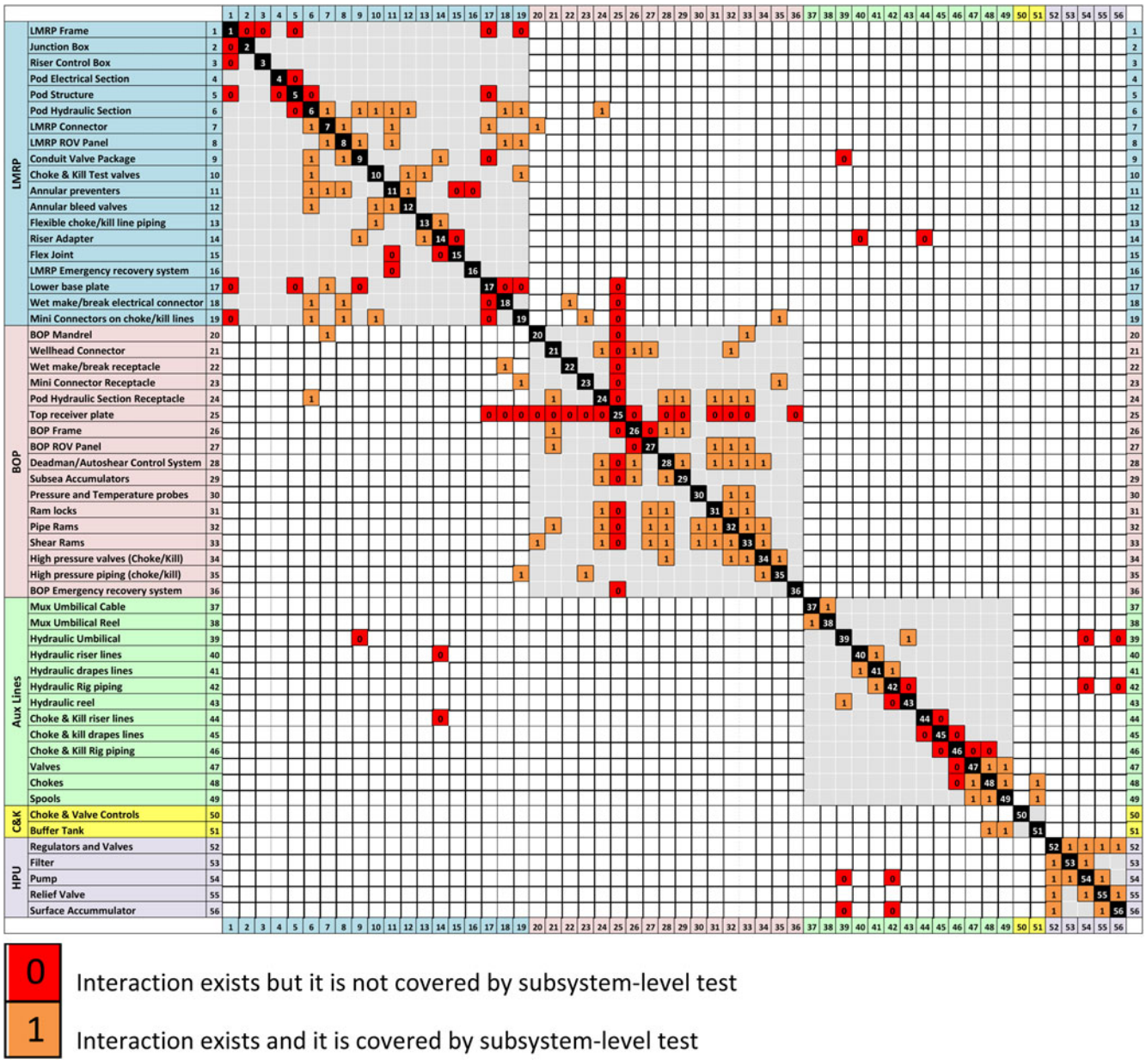


Fig. 4. Multilevel structural interaction design structure matrix showing the subsystem test level.

the structural interfaces that are (or are not) addressed by subsystem-level tests. There are a total of 190 structural interactions within our five subsystems. They are identified in the gray segments of Figure 4. A total of 126 of these interactions are addressed during subsystem level tests and they are marked "1," and the other 64 are marked "0." A small number of interfaces across subsystems are also tested at the subsystem level because those subsystems are tested together; there are 10 such interfaces between the LMRP and BOP, and 4 between the C&K system and Aux Lines. Similarly, the "2" (and "0") in Figure 5 show the interfaces that are (or are not) addressed by set of dock system-level tests. Finally, Figure 6 uses "3" and "0" to identify the interfaces tested in the subsea system-level tests. In principle, every interface can be tested at the subsea level because the full system is installed in its operational condition. It is clear that not all tests are relevant to each interface. It is also evident that the test suite we analyzed has very different distribution of coverage at the subsystem, dock, and subsea levels of tests.

4.2. Multilevel output: Maximum integration level

Figures 7 and 8 combine interaction marks from multiple test levels. The off-diagonal terms in the DSM can be filtered out of the data across multiple levels to reveal the highest test level at which each interaction is tested, per dimension. We map the largest index of a positive test level in the vector corresponding to each of the interactions onto the system archi-

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

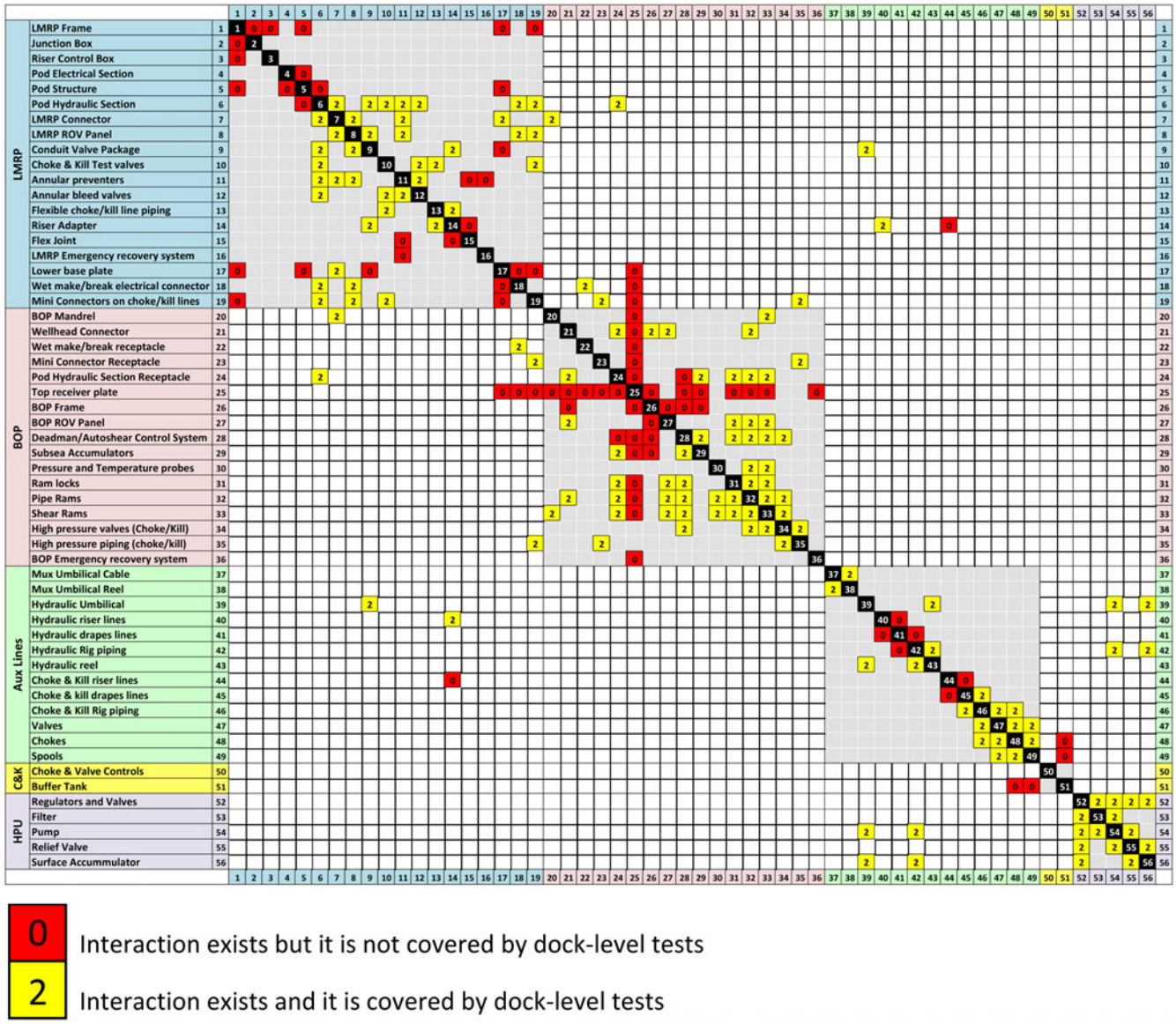


Fig. 5. Multilevel structural interaction design structure matrix showing the dock test level.

tecture DSM. The maximum integration level DSM for the structural dimension is presented in Figure 7. For example, a "1" in the maximum integration level DSM indicates that that particular interaction is last tested at the subsystem level and is not at all tested at the dock level or subsea level.

Given no constraints on resources, an ideal system validation procedure would have all interactions tested at the final test level in the sequence. In this way, all interactions are tested in the most completely assembled configuration and in the most realistic setting to actual operational conditions. A system that is fully tested at the subsea level would lead to a maximum integration level DSM in Figure 7 with every interaction entry a dark green "3." A red entry of "0" indicates that the interaction does exist but is not tested in any of the integration tests in this data set.

4.3. Multilevel output: Minimum integration level

A second useful way to present the integration test data is the minimum integration level DSM. Such a DSM shows the first level at which each interaction is tested in each dimension. The minimum integration level DSM for the structural dimension is presented in Figure 8. The data displayed in the DSM are the result of a minimum search of the test-level vector for each interaction. A red entry of "0" in the DSM indicates that the interaction is not tested in the integration test sequence in any assembled configuration.

From the minimum integration level DSM, we would ideally see that each interaction within a subsystem would be first tested at the subsystem level. This area is shaded gray for clarity of visualization. Therefore, all of the entries in the

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

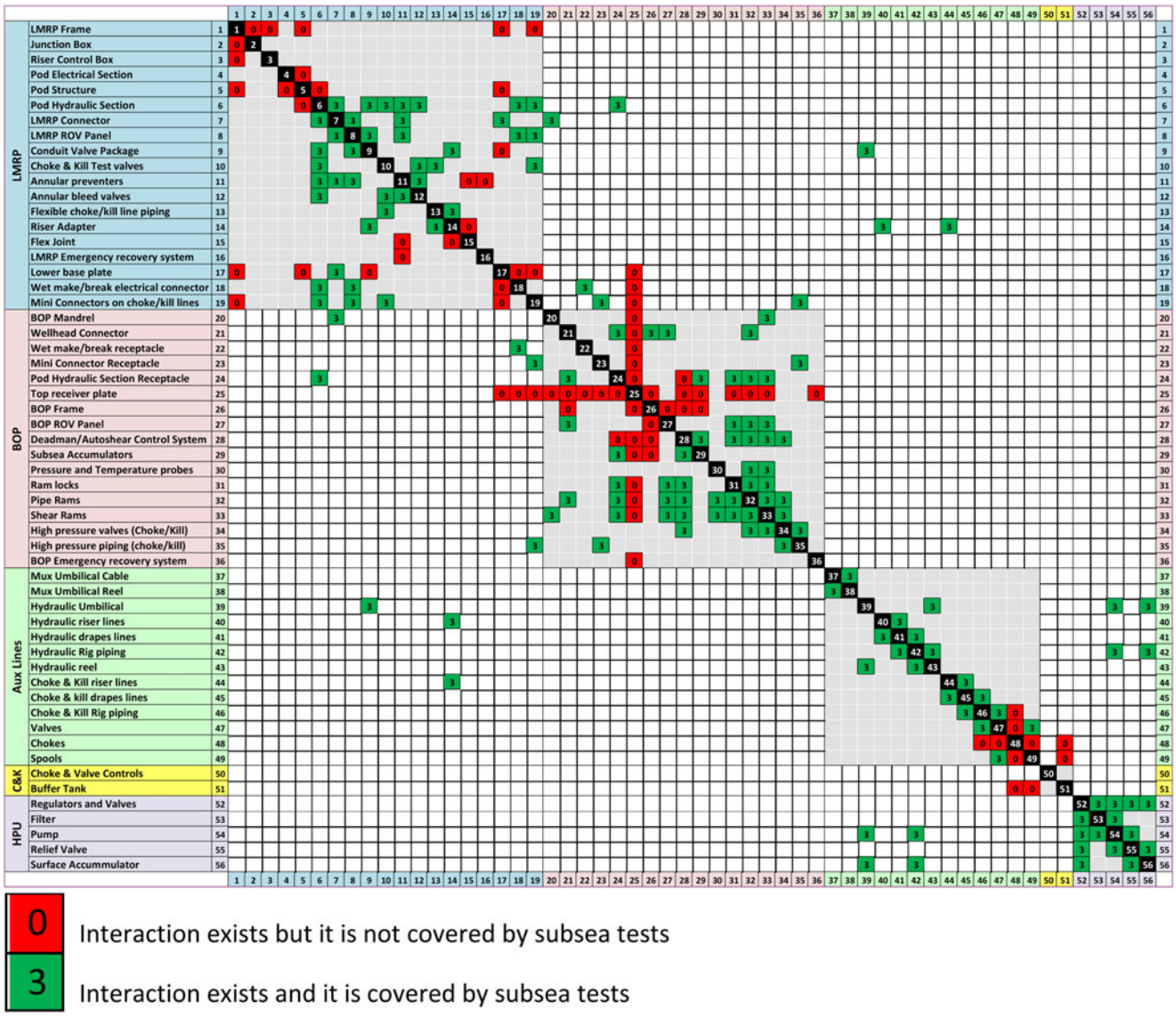


Fig. 6. Multilevel structural interaction design structure matrix showing the subsea test level.

gray shaded area along the diagonal should ideally be "1." In Figure 8 we see that, within the Aux Lines subsystem, 14 interfaces have the ideal "1" value, 8 are "2," and 2 are "3." For interactions between those subsystems that are assembled prior to subsystem level tests (BOP and LMRP, C&K and Aux Lines), we also see a "1."

We would also expect that any intersubsystem interaction could not be tested until the second or third (dock or subsea) levels, because those interactions do not exist for testing before the subsystems have been assembled. Therefore, the DSM entry for those interactions outside the gray shaded area would be a "2" or a "3." For example, an interaction between a component in the LMRP subsystem and the Aux Lines subsystem could only first be tested at the dock or subsea level.

This DSM is a map of when information regarding interaction performance is revealed within the SE-V process. An ideal testing protocol would reveal as much information about the performance of the interactions as soon as possible, revealing issues and risks early to allow time for mitigation, rework, or redesign. From this interpretation, the ideal minimum integration level DSM would show that all intrasubsystem entries are tested at subsystem level (all entries are "1") and the intersubsystem entries are all tested when assembled (all entries are "2").

5. DISCUSSION

In many industries, test procedures are based on regulatory requirements and industry standards. Such standards do not tend to specify tests from an interaction point of view. The DSM-based query of interactions is a different lens through which the completeness of the test set can be considered.

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

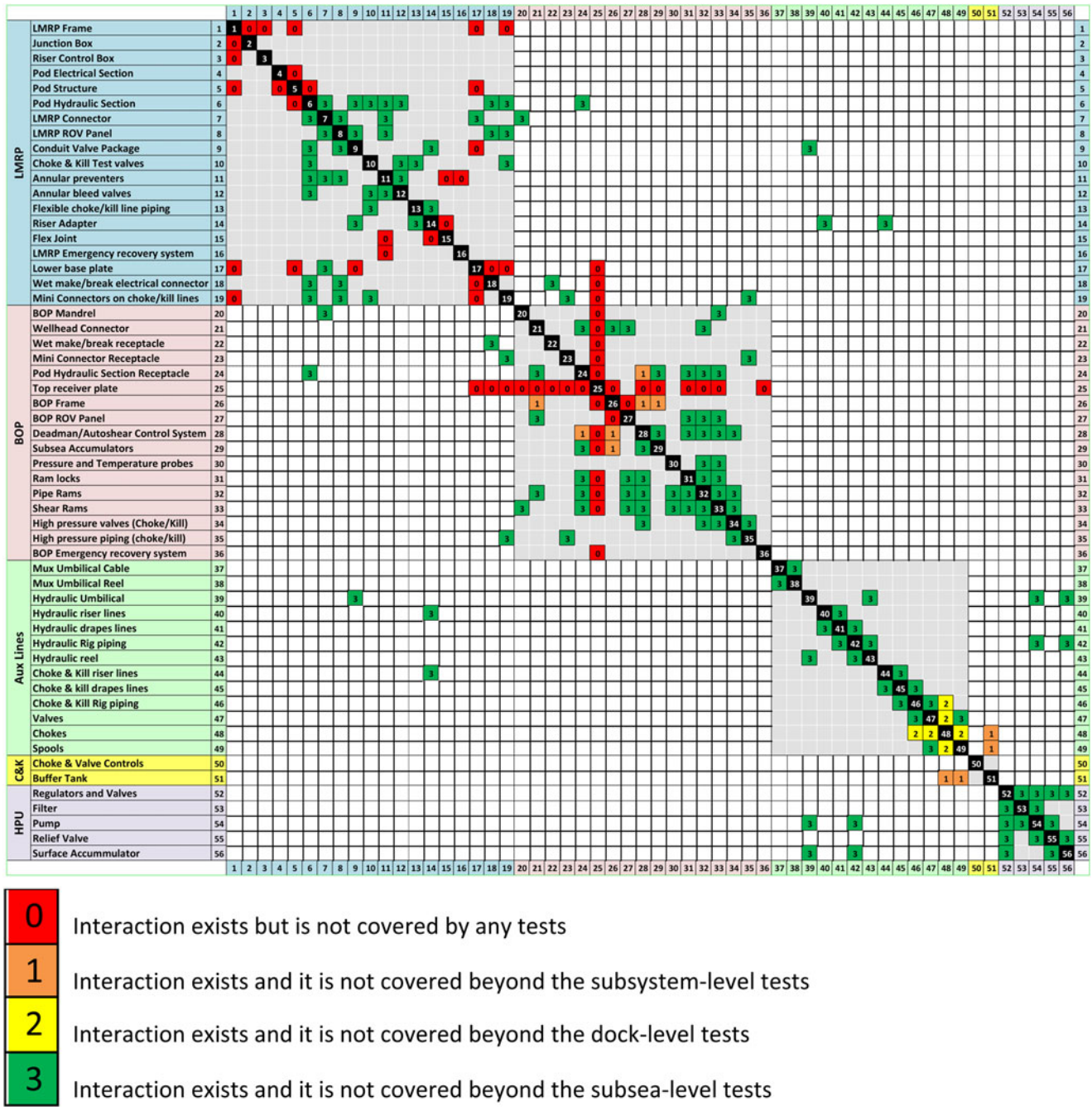


Fig. 7. Maximum structural integration level of the design structure matrix.

Thus, this analysis has the potential to reveal previously undiscovered information and insights to systems engineers.

5.1. Potential for unanticipated outcomes

Upon examination of the maximum integration level DSM, we see in Figure 7 that two-thirds (66%) of the interactions are tested to the highest test level (subsea) in the structural dimension; however, a quarter (26%) of the interactions are not being tested in the integration test set at all. For instance, we observe that all of the interactions involving the top receiver plate and all of the interactions involving the LMRP frame are not structurally tested during system integration. This is because these two components are not instrumented with strain gauges during these tests. We presume such instrumentation would require costly or time-consuming procedures in order to check these interfaces after assembly. Thus, it is possible for the multilevel analysis proposed in this paper to yield outcomes that can point to opportunities to improve the integration stage of the SE-V process.

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

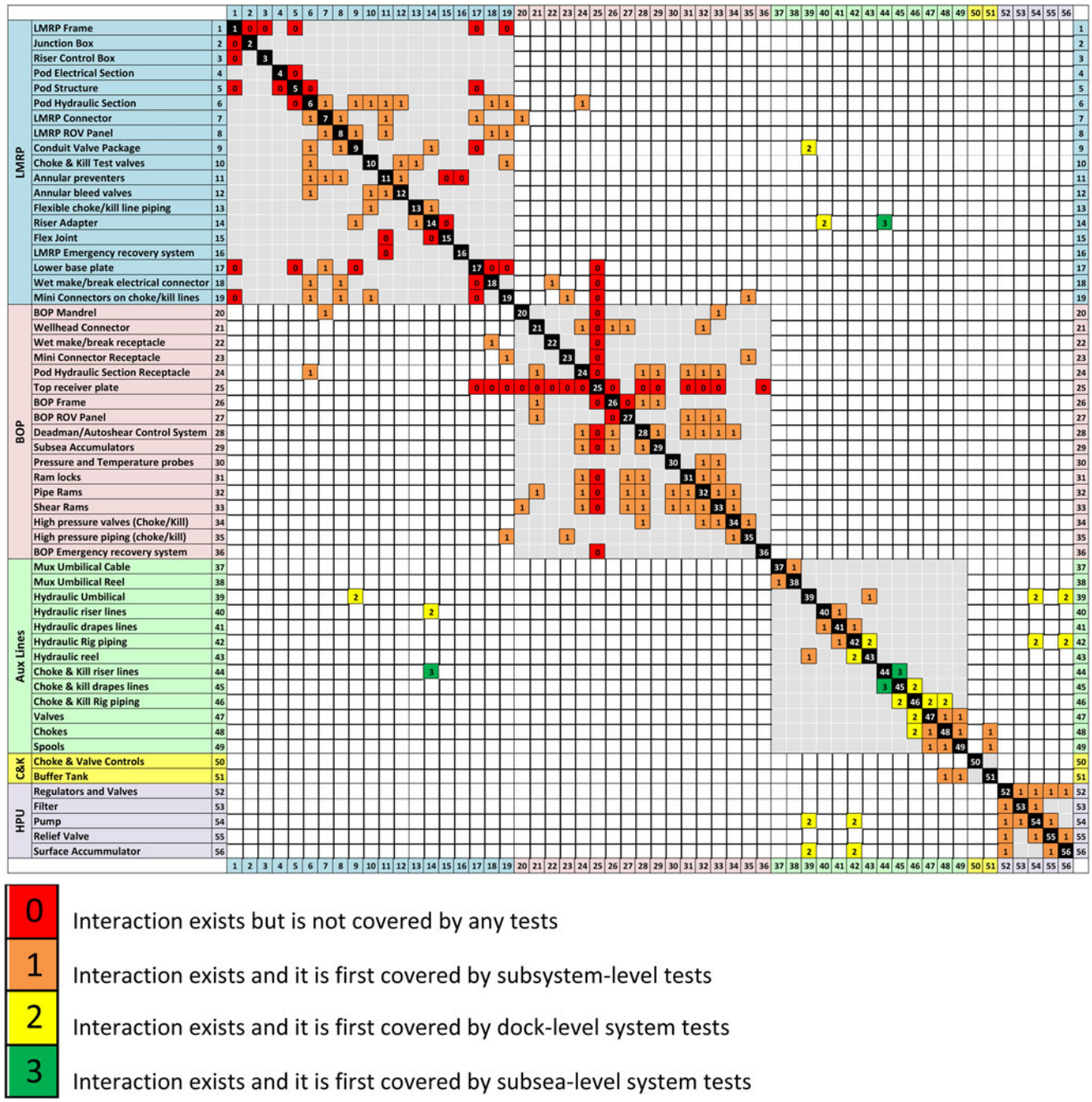


Fig. 8. Minimum structural integration level of the design structure matrix.

A deviation from the ideal test level discovered through the maximum and minimum integration level DSMs may either prompt a redesign of the interface or call for additional instrumentation on the existing interface so that it can be tested. Furthermore, it may induce the development team to introduce additional integration tests. One caveat to these findings is that the quality of output in terms of completeness of coverage is predicated upon the completeness of the chosen integration test suite. In many complex systems ranging from offshore oil operations to mission critical software development (Rosenblum & Weyuker, 1997), it is difficult to include all the test conditions and their combinations. It is therefore common to use a range of test cases (sometimes known as regression tests) to create adequate test coverage.

In any case, DSMs (shown in Figs. 4, 5, and 6) provide useful maps for designing test coverage and for debugging structural failure modes. Such findings are not limited to the structural dimension. We have studied the maximum and minimum integration level DSMs for the other four dimensions (not shown here). For instance, the information dimen-

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

sion DSMs show that, within the scope of the 25 tests we considered, the interface between the pod hydraulic section receptacle and the deadman/autoshear control system is not tested beyond the subsystem level. This analysis of integration-phase testing raises the possibility of potentially revealing unanticipated failure modes and when additional tests should be performed, either at the subsystem or system level.

5.2. Insights from multilevel analysis

A key contribution from this paper lies in the manner in which test and integration data are represented within the DSM. The use of maximum and minimum functions is merely one analytical approach for improving outcomes based on this representation. Other analytical formulations are also possible. The choice of query and formulation function depends on the question being asked. For instance, we have examined the data generated by alternative multilevel queries (one set for each dimension of the 25 tests, disaggregated by levels, listed in Table 1) to figure out either how early or how completely a particular test may address integration issues at a given level of analysis. We have also examined the failure modes associated with an aggregate (i.e., a single level) map of the product architecture by querying the DSM representation that yielded measures, such as "network centrality," and provided insights on whether the network position of a component contributed to system failure. Such results are not presented in the current manuscript for brevity.

The minimum integration level DSM reveals that in the structural dimension, some interactions are not tested until the subsea level, even though these interactions are present earlier in the test sequence (assuming that subsystems are assembled first). Many of the auxiliary lines interactions exhibit this behavior, likely because they are not yet assembled for dock tests because they are too physically large. Further, we see that some intersubsystem interactions are not tested until the subsea level despite the fact that the interacting components may be fully assembled, although not in the deployed environment, in the second (dock) level. There is only one example of such an interface, that between the choke and kill riser lines and the riser adapter. The maximum integration level DSM (see Fig. 7) reveals that in the structural dimension, some interactions are tested at the subsystem and then are not tested as the system progresses through integration. For example, the interactions within the BOP subsystem between the BOP frame and the wellhead connector are tested at the subsystem level but are not tested at the dock or subsea system-level configurations. Thus, the multilevel timing information revealed in the maximum and minimum integration level DSM analyses shows which of the interfaces are tested early and late in the integration process. Based on their coverage of interfaces, a design team can assign different levels of risks to the integration plan. This observation gives rise to questions of how the dock testing and subsea testing scope are decided. For instance, we found that in the material dimension minimum integration level DSM, that all of the intrasubsystem interactions are tested at the ideal time, as soon as possible, except for those involving the flex joint, which are not integration tested through the set of tests examined in this work.

The interaction information in the DSM representation is restricted to our review of engineering documentation, followed by inputs provided by subject matter experts. It is possible that other interactions exist, but they are neither reported in the documentation nor anticipated by an expert. It is also possible that some potential failure modes might precipitate through a combination of interactions. This heightens the need for careful design of the integration phase in the SE-V through a series of tests to uncover unanticipated interactions or combinations of interactions. The rigor of the method described in this paper is restricted by the representation schema and data that we have captured. It does not guarantee completeness of the test coverage. It also does not rule out the possibility of unanticipated failures during integration tests. The DSM representation can inform failure model and effect analysis (IEC 60812) in terms of interaction pattern identification and coverage while exploring the causes for unanticipated failures. INCOSE (2011) recommends an integration process that "verifies that all boundaries between system elements have been correctly identified and described." DSM representation and allied maximum and minimum integration level analyses can complement several useful alternatives for investigating system integration: the hazard and operability study, (IEC 61882), network reliability modeling (Michelena & Papalambros, 1995), and so forth.

Our initial field study has restricted the scope of the work to five dimensions of dependencies: spatial fit, structural load, energy flow, information flow, and material (fluid) flow across only two domains (component and testing) from a list of five domains shown in Figure 2. The current analysis is preliminary and limited to demonstrate a proof of the multilevel analysis concept. Thus, we have restricted the analysis of the interactions to a single dimension, in this case, structural, as shown in Figures 4 through 8. In reality, there can be significant interactions across the five dimensions. For instance, a structural load may cause deflections that could create spatial misalignment while making hydraulic line connections. It is possible to augment the analysis, by constructing combinations of interaction measures. We leave such an analysis as an extension for follow-on work.

6. CONCLUSION

The research underlying this project, and the method outlined in this paper, are at an early stage of development. Multilevel analysis of DSMs developed in this study contributes to the design of complex engineered systems by addressing two gaps: first, it develops a data collection and mapping methodology to account for the amount of coverage available at each interface within DSM representation of complex SE-V processes; and second, it offers a theoretical basis and a method for data aggregation and query that accounts for differing scales,

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

in terms of both level and timing, to explore if different types of integration risks may be evident at different time scales.

Design and analysis of complex engineered systems is a growing research area that calls for systematic and rigorous approaches based on advances in complexity and behavioral sciences (Anderson & Joglekar, 2012). Augmented vector DSM data and visualizations presented in this paper can lend themselves to further analysis. For instance, multilevel data can be used to inform the development of system architecture decomposition options and optimal sequencing of the integration tasks based on design for testability and design for reliability considerations. Developments based on detailed understanding of interactions at each interface, captured in the off-diagonal cells of a system architecture DSM, may yield novel integration risk metrics, algorithms, and behavioral research opportunities for improving complex system design early in the SE-V process.

REFERENCES

Albers, A., Braun, A., Sadowski, E., Wynn, D.C., Wyatt, D.F., & Clarkson, P.J. (2011). System architecture modeling in a software tool based on the contact and channel approach (C&C-A). *Journal of Mechanical Design 133(10)*, 101006.

Allen, J.K., Seepersad, C., Choi, H., & Mistree, F. (2006). Robust design for multiscale and multidisciplinary applications. *Journal of Mechanical Design 128(4)*, 832–843.

Anderson, E., & Joglekar, N. (2012). *The Innovation Butterfly: Managing Emergent Opportunities and Risks During Distributed Innovation*. New York: Springer.

Ashby, W.R. (1956). *An Introduction to Cybernetics*. New York: Chapman & Hall.

Bar-Yam, Y. (2003). When systems engineering fails—toward complex systems engineering. *Proc. IEEE Int. Conf. Systems, Man and Cybernetics*, Vol. 2, pp. 2021–2028.

Bar-Yam, Y. (2006). Engineering complex systems: multiscale analysis of evolutionary engineering. In *Complex Engineering Systems* (Braha, D., Minai, A., & Bar-Yam, Y., Eds.), pp. 22–39. Berlin: Springer.

Beer, S. (1975). *Designing Freedom*. New York: Wiley.

Blanchard, B.S., & Fabrycky, W.J. (1990). *Systems Engineering and Analysis*. Englewood Cliffs, NJ: Prentice Hall.

Braha, D., & Maimon, O.Z. (1998). *A Mathematical Theory of Design: Foundations, Algorithms and Applications*. Boston: Kluwer Academic.

Browning, T.R. (2001). Applying the design structure matrix to system decomposition and integration problems: a review and new directions. *IEEE Transactions on Engineering Management 48(3)*, 292–306.

Danilovic, M., & Browning, T.R. (2007). Managing complex product development projects with design structure matrices and domain mapping matrices. *International Journal of Project Management 25(3)*, 300–314.

Eppinger, S.D., & Browning, T.R. (2012). *Design Structure Matrix Methods and Applications*. Cambridge, MA: MIT Press.

Giffin, M., de Weck, O., Bounova, G., Keller, G.R., Eckert, C., & Clarkson, P.J. (2009). Change propagation analysis in complex technical systems. *Journal of Mechanical Design 131(8)*, 081001.

Griffin, M.D. (2010). How do we fix system engineering? *Proc. 61st Annual Int. Congr.*, Paper No. IAC-10.D1.5.4, Prague, September 27–October 1.

Halman, J., & Braks, B. (1999). Project alliancing in the offshore industry. *International Journal of Project Management 17(2)*, 71–76.

Hartmann, C., Zhang, W., Latorre, J., & Pavliotis, G., (2013). Optimal control of multiscale systems: an approach using logarithmic transformations. *Proc. Int. Conf. Scientific Computation and Differential Equations, SciCADE 2013*, Valladolid, Spain, September 16–20.

International Electrotechnical Commission. (2001). *IEC 61882: Hazard and Operability Studies (HAZOP Studies)—Application Guide*. Geneva: Author.

International Electrotechnical Commission. (2006*a*). *IEC 60812: Analysis Techniques for System Reliability—Procedures for Failure Mode and Effect Analysis (FMEA)*. Geneva: Author.

International Electrotechnical Commission. (2006*b*). *IEC 61025: Fault Tree Analysis (FTA)*. Geneva: Author.

INCOSE. (2011). *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities* (Haskins, C., Ed.). San Diego, CA: Author.

Leveson, N.G. (2011). *Engineering a Safer World*. Cambridge, MA: MIT Press.

Leveson, N., Dulac, N., Marais, K., & Carroll, J. (2009). Moving beyond normal accidents and high reliability organizations: a systems approach to safety in complex systems. *Organization Studies 30(2–3)*, 227–249.

Lewis, K. (2012). Making sense of elegant complexity in design. *Journal of Mechanical Design 134(12)*, 120801.

Michelena, N., & Papalambros, P. (1995). A network reliability approach to optimal decomposition of design problems. *Journal of Mechanical Design 117(3)*, 433–440.

Papakonstantinou, N., Sierla, S., Jensen, D., & Tumer, I.Y. (2012). Simulation of interactions and emergent failure behavior during complex system design. *Journal of Computing and Information Science in Engineering 12(3)*, 031007.

Pimmler, T.U., & Eppinger, S.D. (1994). Integration analysis of product decompositions. *Proc. ASME Design Theory and Methodology Conf.*, Minneapolis, MN.

Reich, Y. (1998). Learning in design: from characterizing dimensions to working systems. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 12(2)*, 161–172.

Rosenblum, D.S., & Weyuker, E.J. (1997). Lessons learned from a regression testing case study. *Empirical Software Engineering 2(2)*, 188–191.

Simoncelli, E.P., Freeman, W.T., Adelson, E.H., & Heeger, D.J. (1992). Shiftable multiscale transforms. *IEEE Transactions on Information Theory 38(2)*, 587–607.

Stamatis, D.H. (2003). *Failure Mode and Effect Analysis: FMEA From Theory to Execution*. Milwaukee, WI: ASQ Quality Press.

Stone, R.B., Tumer, I.Y., & Van Wie, M. (2005). The function–failure design method. *Journal of Mechanical Design, 127(3)*, 397.

Tilstra, A.H., Seepersad, C.C., & Wood, K.L. (2012). A high-definition design structure matrix (HDDSM) for the quantitative assessment of product architecture. *Journal of Engineering Design 23(10–11)*, 767–789.

Weinan, E., Engquist, B., Li, X., Ren, W., & Vanden-Eijnden, E. (2007). Heterogeneous multiscale methods: a review. *Communications in Computational Physics 2(3)*, 367–450.

Whitney, D.E., Dong, Q., Judson, J., & Mascoli, G. (1999). Introducing knowledge-based engineering into an interconnected product development process. *Proc. 1999 ASME Design Engineering Technical Conf.*

Yassine, A. (2004). An introduction to modeling and analyzing complex product development processes using the design structure matrix (DSM) method. *Urbana 51(9)*, 1–17.

Zha, X.F., Sriram, R.D., Fernandez, M.G., & Mistree, F. (2008). Knowledge-intensive collaborative decision support for design processes: a hybrid decision support model and agent. *Computers in Industry 59(9)*, 905–922.

Steven D. Eppinger holds the General Motors LGO Chaired Professorship at the MIT Sloan School of Management, where he teaches product development and technology management in graduate and executive programs. He received bachelor, master, and doctoral degrees in mechanical engineering from the Massachusetts Institute of Technology. Dr. Eppinger is the coauthor of the widely used textbook *Product Design and Development* (McGraw–Hill) and *Design Structure Matrix Methods and Applications* (MIT Press). His research interests span management of complex engineering processes, technical project management, and product design methods.

Nitin Joglekar is on the faculty at the Boston University's School of Management. He studied engineering at IIT Kharagpur, India, Memorial University, St. John's, Canada, and the Massachusetts Institute of Technology. He received a doctoral degree from the MIT Sloan School. His work expe-

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.

rience includes stints in the energy and IT industries. Dr. Joglekar's research interests include management of distributed innovation, technology readiness and commercialization, entrepreneurial operations, and allied public policy issues.

Alison Olechowski is a PhD candidate in the Department of Mechanical Engineering at the Massachusetts Institute of Technology. She earned her master's degree from the Massachusetts Institute of Technology studying risk management in product development, and a bachelor's degree from Queen's University in Kingston, Ontario. Her current work focuses on the development of system-level tools for technology maturity and risk analysis.

Terence Teo is a master's candidate at the Singapore University of Technology and Design. He has earned a master's degree in engineering and management from the Massachusetts Institute of Technology. His research interests are in product design and development and the role of crowd funding for equity and debt financing of small businesses.

Downloaded from https://www.cambridge.org/core. 16 Nov 2026 at 20:28:55, subject to the Cambridge Core terms of use.