2013

Model-Based System Integration (MBSI) Key Attributes of MBSE from the System Integrators Perspective

Montgomery, Paul R.

Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 16 (2013) 313 – 322

Procedia Computer Science

Conference on Systems Engineering Research (CSER'13)
Eds.: C.J.J. Paredis, C. Bishop, D. Bodner, Georgia Institute of Technology, Atlanta, GA, March 19-22, 2013.

Model-Based System Integration (MBSI) – Key Attributes of MBSE from the System Integrator's Perspective

Paul R. Montgomery, D.Sc., ESEP[a]*

Naval Postgraduate School, Monterey, CA, USA

Abstract

Many system developments fail at the integration and qualification (I&Q) phases in the development and acquisition cycle. How can we, as Systems Engineers (SE), navigate the uncertainty and risk of system development to ensure I&Q success? One solution is applying significant influence of the system integrator (SI) member of the SE team at the very beginning of system design. If this is a solution, then what processes, methods, and practices can the SI apply in the overall SE process? Can the emerging Model-Based Systems Engineering (MBSE) methods and tools be leveraged by the SI? This paper discusses a Model Based System Integration (MBSI) approach that applies essential MBSE methods and tools to the unique goals of the SI. While MBSE is supportive of the entire SE process, it tends to be optimized for the design-side of the SE process and not necessarily for I&Q goals and objectives. MBSI highlights how MBSE tools and methods can be extended to benefit the SI. Lessons-learned from several SE graduate school projects applying these MBSI methods are provided to demonstrate the efficacy of the MBSI approach.

Keywords: Model Based Systems Engineering; integration; qualification; system acquisition; system design;

1. Introduction - The SI role and mission

This paper focuses on the role of the system integrator (SI) within the system engineering (SE) process in the acquisition and design of systems. This paper examines the methods and practices of the SI and contrasts them to those of the SE, particularly the design system engineer. Although underlying theories of integration are emerging [1], this paper explores integration *practices* and *principles*. A Model-Based Systems Integration (MBSI) approach is introduced that is a derivative of model-based systems engineering (MBSE) methods and tools which proposes a key SI principle and practice:

- Principle - Integration begins early at system concept and directly influences system design
- Practice - Do not attempt to build / integrate the system until the system model is integrated first

It is commonly discussed in the system acquisition community that system failures often occur during the integration and qualification (I&Q) phase [2,3]. What are the root causes of these failures? The SE community will often relegate I&Q activities to the post-development phases of system acquisition. This approach may be attributed to the classic system engineering model diagram (Fig. 1(a)) that shows system I&Q activities on the right side of the SE "V". Although not originally intended to imply sequential and linear process flows, this ubiquitous model has

* Tel.: 703-568-1165 prmontgo@nps.edu

doi:10.1016/j.procs.2013.01.033

tended to create a very segmented view of system I&Q as separate from system design (i.e. the left side of the "V"). The INCOSE Handbook [4] implies the primary input to the system integration is the acquired and built system elements and their documentation. In other words, the SI may or may not have had influence on the design of the system components. The most recent Systems Engineering Book of Knowledge (SeBoK) [5] also reinforces the common understanding that integration is a post-development set of activities (Fig. 1(b)). This SE model, although useful at the top-level SE process understanding, can often be over-extended to the point of misuse [6].

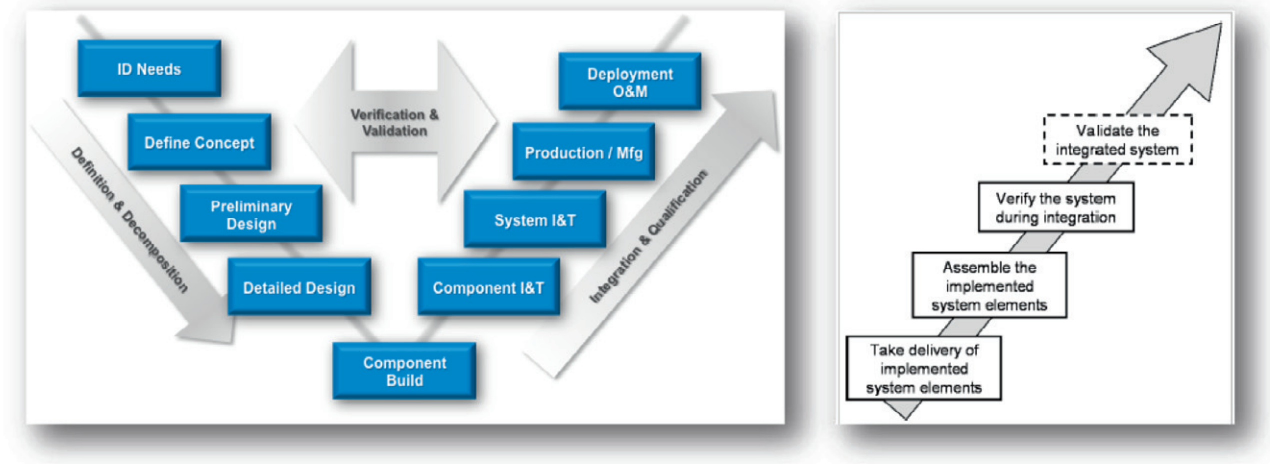

Fig. 1. (a) Systems Engineering "V" Model (derived from [4] and multiple sources); (b) SEBok Depiction of the Systems Integration Process ([5])

What are the essential roles and missions of the system integrator and why might the figures above be somewhat misleading to system engineers? The primary mission of the system integrator is to ensure that a system comes together by assembling the essential components and elements into a cohesive and fully functional system that satisfies the original user needs. For the purposes of this discussion, the SI's role and mission includes both integration and qualification, both of which are essential design activities as discussed below.

1.1. Integration Design

Integration design is the key activity of an integrator that can begin early that ensures multiple elements of the systems are designed such that their functionality, interfaces, interoperability, behaviors, and essential missions can be integrated. These attributes need to be considered early and tested often to increase likelihood of success of bringing the system together into a cohesive unit. The SI design emphasis is on system interfaces and interactions.

1.2. Qualification Design

In this paper, the term qualification includes verification, validation, and acceptance. MBSE principles and tools can augment the development of qualification methods and strategies [7]. Qualification activities are continuing activities that are not necessarily well depicted in the system engineering "V" (Fig. 1). Arrows between the two legs of the "V" often confuse many to think that validation and verification are only performed in the SE "endgame". The SI early focus is on whether or not the system can be verified successfully against parametric performance and can be proven to accomplish the user's mission performance demands.

2. Model-Based Systems Engineering (MBSE) Concepts

2.1. What is a system model?

The SE is focused on the architecture and physical design early in the system acquisition process. The design is derived from requirements, user needs, and operational analysis. Developing a system model can support all facets of the system acquisition process (Fig. 2(a)) and provides a well-defined system baseline. Many of the model-based system engineering tools in use today will produce and enable those design processes in a disciplined and complete manner [8,9]. For the purposes of this discussion, a system model will be defined as the data-driven collection of views and attributes that defines these various dimensions of a system as shown in Fig. 2(b). Unlike some discussions of building systems models with a diagrammatic (e.g., DoDAF) approach [10], the system model in this presentation is a data-driven, tool-based approach whereby output artifacts (diagrams) are the by-products of the system definition data.

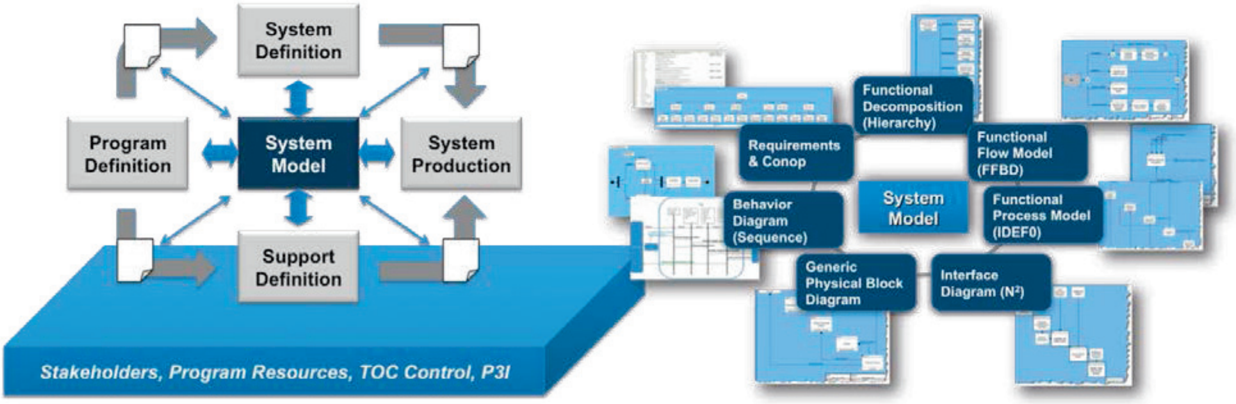

Fig. 2. (a) System model supports all aspects of system acquisition [original]; (b) A system model [Derived from [11]]

2.2. SE and MBSE Contrasts

System engineering processes (Fig. 3 a) are often document-driven and event-driven. These events are often system engineering technical reviews which establish key milestones in system development. A contrasting MBSE process is shown in Fig. 3b where a system definition model is created to support all of the system engineering activities. The system definition can produce any documentation desired as a byproduct of the model itself. The system model can be used throughout the lifecycle of a system. The model can be used to quantitatively analyze the performance of system-of-systems and provide a repeatable baseline of system design information that can be used in trade-off analyses, technical reviews, and future developments.

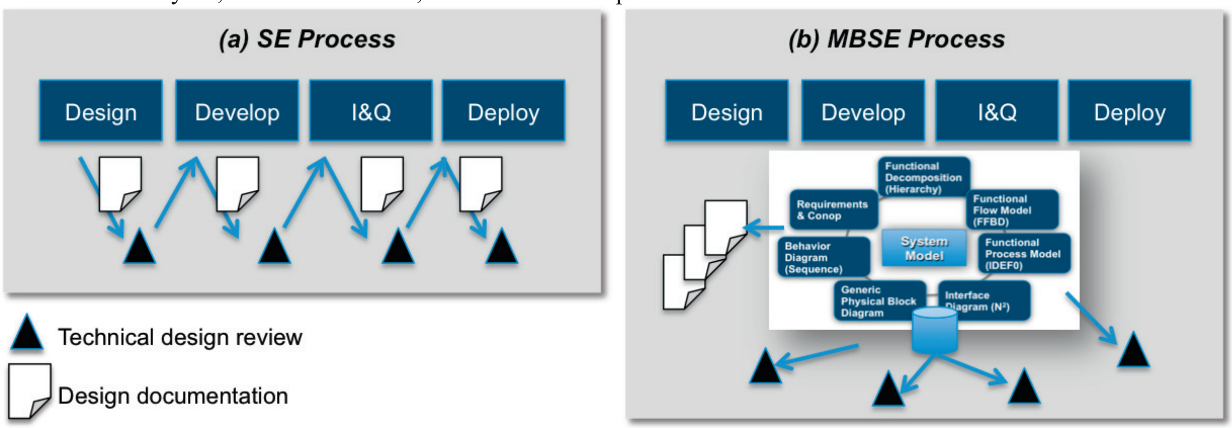

Fig. 3. An SE process (top) is often a document-driven process while an MBSE process (bottom) is a system model-driven process [original]

3. The MBSI Concept

The MBSI process extends the MBSE process in that it (1) increases the impact of the SI early in the design and (2) demands the subsystem models be integrated successfully prior to building and integrating the subsystems themselves. A simple example of the MBSI concept applied to the SE process is shown in (Fig. 4).

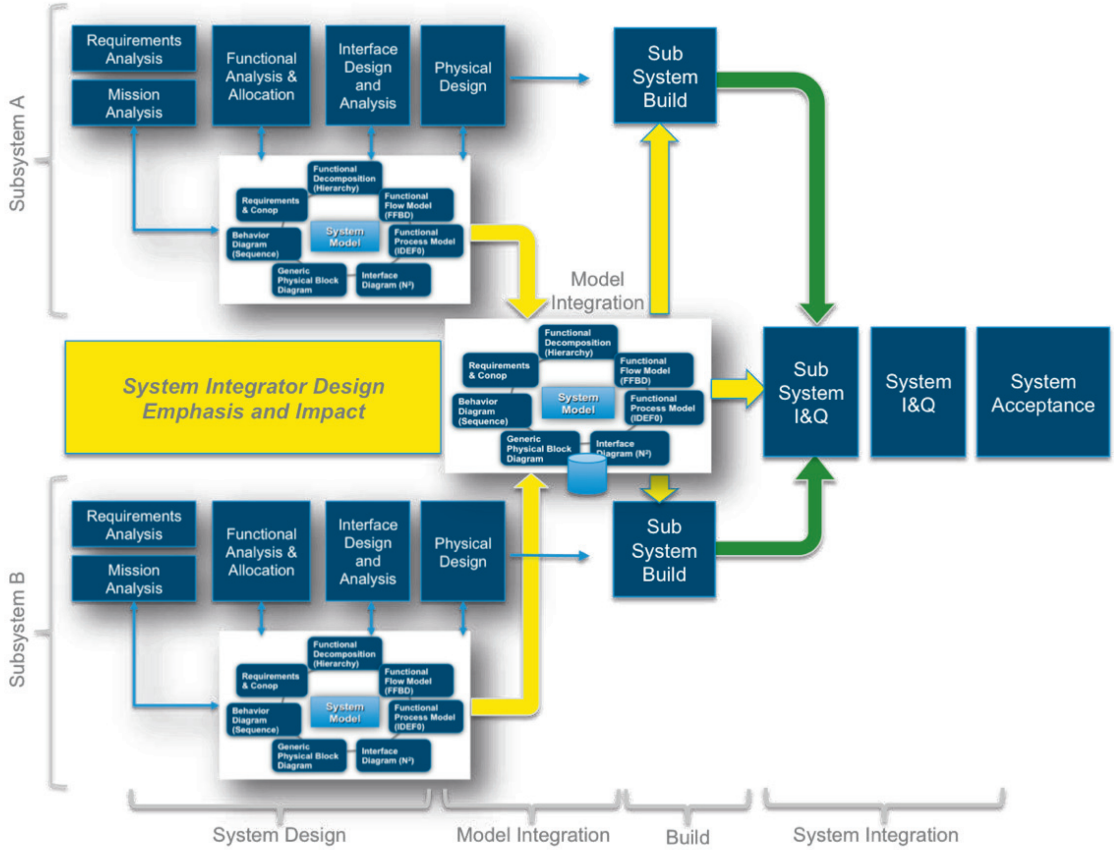

Fig. 4. The Model-Based System Integration (MBSI) concept increases emphasis on early SI design involvement and integrates the system model before system build and integration [original]

3.1. SI Design Emphasis

The SI needs to be involved early in the design process. The SI does not want to *inherit* a system design but, rather, needs to *influence* the design to reduce I&Q risks. The influence of the system integrator at these early stages can be either underestimated or even unwanted in some teams. The system integrator has a different perspective during design that can mitigate many latent design problems that could create significant integration challenges and risks. Early modeling of the system is just as imperative to the SI as to the SE. The SI, however, views the system model with an I&Q perspective. In the table below, some of these SI / MBSI emphasis areas during early design are highlighted and occur in the yellow emphasis block in Fig. 4. While not all-inclusive, they do show MBSI emphasis areas that could be incorporated into existing or emerging MBSE tools and/or practices.

Table 1. Early design emphasis of the SI

SE Process	SI/MBSI Emphasis
Requirements validation	Ensure that requirements are achievable in that they can be *qualified*
Requirements-to-functionality validation	Ensure that the functional architecture leads to sufficient modularity, completeness, and can ultimately be *integrated*
Requirements-to-testing validation and planning	Ensure the requirements can be proven during *validation/acceptance* and the acceptance strategy is explicitly agreed upon among all stakeholders
Conop-to-mission performance	Ensure that the mission and operational goals and

	objectives have been parameterized (e.g., measures of effectiveness – MOEs) with sufficient fidelity to form a basis of *validation / acceptance*.
Conop-to-system performance	Ensure that the system-level requirements are valid and have been parameterized (e.g., measures of performance – MOPs) with sufficient fidelity to form a basis of *verification* throughout I&Q.
System interface design	Ensure functions are connected and interfaced with minimum complexity, modularity, and simplicity to minimize *integration* risks
Functional interactions and interoperability	Ensure system functionality is *interoperable* with external systems with which the system will be required to interoperate.

3.2. Integrate the models before build and integration

The MBSE process and supporting tools help system designers develop a cohesive and coherent design that is quantifiable, easily shared and viewed by wide variety of stakeholders, and provides a stable baseline of design data which reduces risks throughout design. If the MBSE tools are robust enough, they will allow subsystem models to be integrated, validated, tested for compliance, and simulated for performance. The MBSE tools, however, should not be viewed to only support design. The risks associated with subsystem build and integration process can be greatly reduced if the development team challenges itself to *integrate the subsystem models into a system model before proceeding to subsystem build and integration*. The discoveries that will emerge from model integration will greatly benefit functional, interface, and potential emergent design challenges that will often need to be revisited and/or modified prior to committing to subsystem build. This idea is shown in the center of Fig. 4.

4. How can MBSI reduce I&Q risks?

If the SI is involved early in the desing process and system models are to be integrated first, how can MBSE tools provide the SI with indications and warnings of I&Q risks? This paper is not intended to describe all the potential attributes of an MBSI tool but there are some perspectives provided by existing tools than an experienced SI can view and discern as potential risk areas for integration. Examples follow that show MBSE tool outputs and how they can provide insight into I&Q risks. They are derived from graduate SE class projects as the system designs were being formulated using CORE® [8] as an MBSE tool (diagrams that follow are simplified cartoons of the CORE diagrams for presentation).

4.1. Functional flow

The SI is keenly focused on the modularity of the functions within a system and validates how these functions and their control structures contribute to the overall performance of the system. In particular, can these functions be integrated independently, incrementally, in parallel, in series, from the bottoms up, or from the top down? An attribute that is produced by MBSE tools is the functional flow control structure of the system model. This is often in the form of a functional-flow block diagram (FFBD). The control flow indicates the mutual dependencies, control, and functional flow of the system. The SI uses the FFBD to determine

- Analyze system complexity
- Prioritize integration efforts
- Determine interface complexity
- Determine system modularity and independence
- Discover redundancy or gaps in functionality
- Discover inadequate control description

All of the above shape the SI integration strategy for a system, particularly the order of integration that often follows control flow. Additionally, single-stage or incremental integration strategy selection [12] can be driven by functional control architecture. The SI may challenge the design if the control indicates deficiencies. Some common examples:

- All-linear FFBD (Fig. 5a) - Inadequate feedback and parallelism. Sequential structure is unlikely in that it does not account for functions that can operate independently. These 'independent' functions can provide greater flexibility for integration.
- All-parallel FFBD (Fig. 5b) - Inadequate recognition of control. Over-emphasized independence of functionality and unrecognized co-dependence and mutual coupling. Can lead to a "naive" integration strategy.
- Complex functionality (Fig. 5c) - A function with several flow and control entries/exits can indicate a higher risk I&Q element.

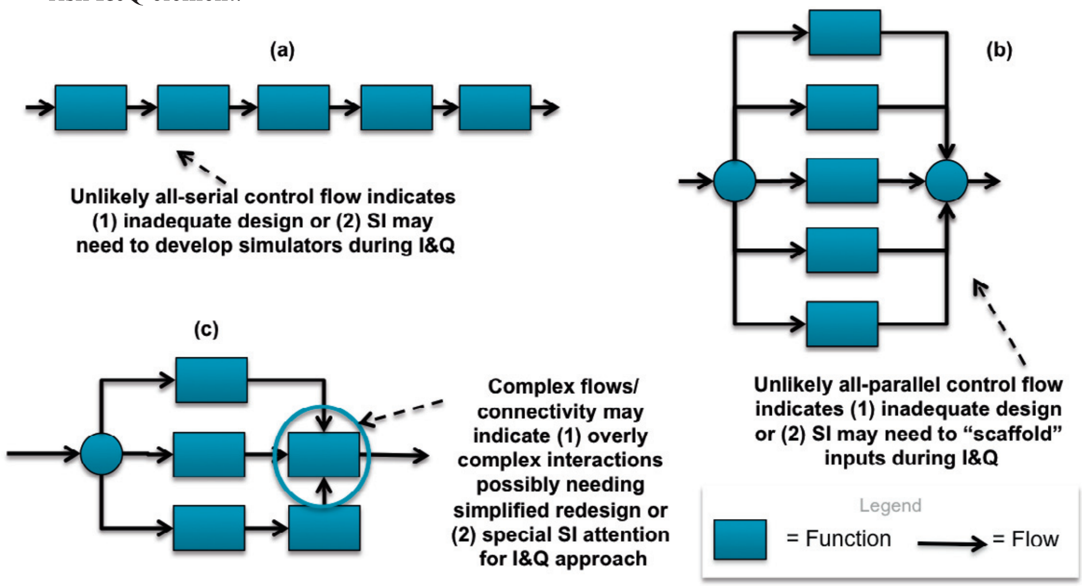

Fig. 5. Functional flow and control can determine integration strategy [original]

4.2. Complexity of interfaces

The SI minimum system interfaces and to keep those interfaces non-complex. The SI may recommend greater simplicity during functional design or be alerted to complex interfaces that could affect I&Q strategy. The N^2 diagram (Fig. 6) is a simple MBSE tool output that tallies the functional interfaces of a system. The SI uses the N^2 diagram to determine:

- Interface completeness
- Interface gaps
- Interface complexity
- System feedback robustness
- External system coupling and boundaries

All of the above alert the SI to external systems I&Q, internal system order of integration based upon interface complexity, qualification emphasis based on number or complexity of interfaces, or I&Q support systems based upon complex feedback interfacing. The SI may challenge the system interface design to minimize the number of interfaces or to simplify interfaces, where possible. Some common N^2 insights include:

- Minimum off-diagonal (upper): Indicates incomplete interface definition
- Minimum off-diagonal (lower): Indicates incomplete feedback design
- Minimum edges: Indicates incomplete external system interface definition
- Many interfaces in single block: Indicates potential complex interface
- Many blocks clustering a function: Indicates a potential I&Q challenge for that function

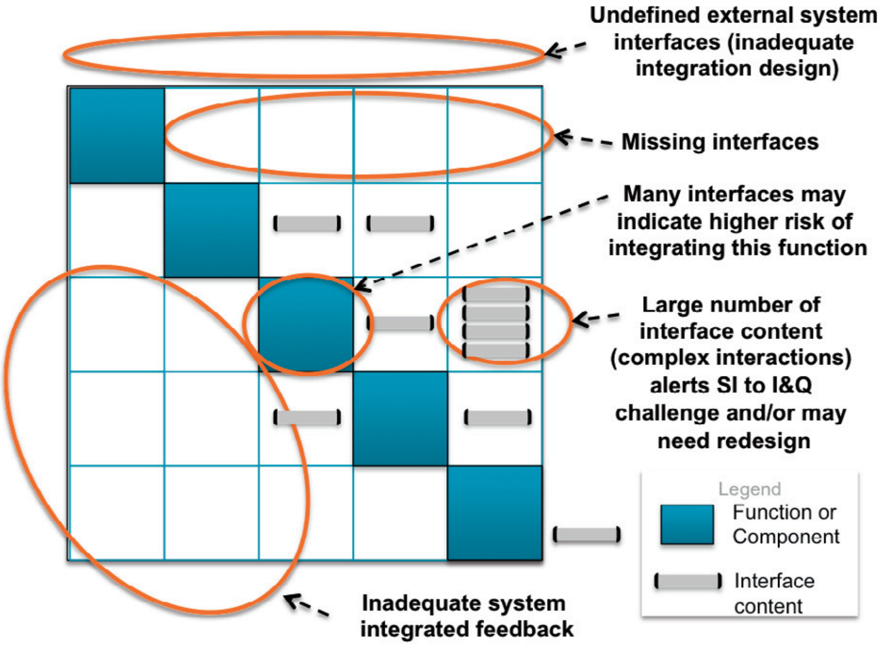

Fig. 6. Interface analysis can inform SI to integration challenges [original]

4.3. System process and data flow

As discussed above, the system integrator views a large number interfaces or complex control flows as presenting risk to I&Q. An IDEF0 diagram (Fig. 7) provides a data flow and functional relationship (often call process flow) depiction. These diagrams can be viewed as an amalgam of FFBD and N^2 diagrams. As such, they can become visually complex very quickly. Its connection-based depiction, however, can provide better visual cues to the SI related to functional design and interface complexity. Because each element follows a prescribed syntax (Input-Control-Output-Mechanism) model [13], the SI can better discern control interfaces from input/output interfaces (often masked in the N^2). Additionally, the "M" (mechanism or resource) dimension is an indication of the allocation of functionality-to-physical components. The SI can use IDEF0 to determine:

- Control interface completeness
- Input/output interface completeness
- Interface gaps
- Interface complexity
- System feedback robustness
- External system coupling and boundaries

All of the above alert the SI to establishing an I&Q strategy that considers the (1) control, (2) functional transformation of inputs-to-outputs, (3) physical resource availability, and (4) external system I&Q limits. Some common examples:

- Functions without control (although this is not uncommon, it alerts the SI to examine system control for integration strategies)
- Non-conservation of interfaces (broken links among functions - this is fundamental design issue but SI is often hyper-alert to interface design issues)
- Broken external interface/data (prevents complete qualification and acceptance testing)
- Undefined data flows (incomplete data definition prevents complete I&Q)

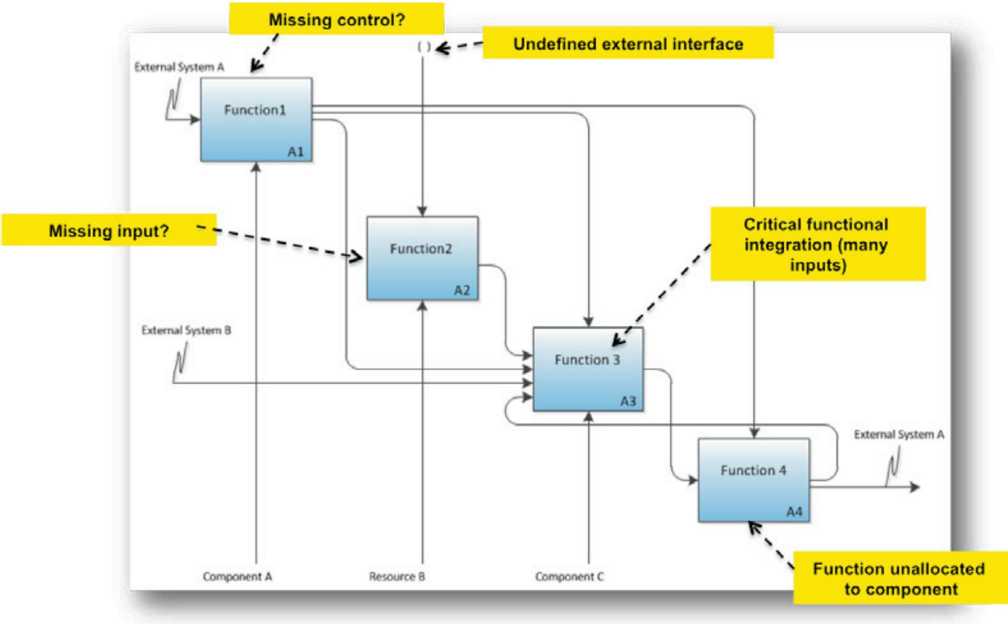

Fig. 7. Excessive interfaces can be identified early to warn of integration risks [original]

4.4. System behavior

System behavior models are derived from the analysis of user mission needs coupled with function-physical design. The SI uses the behavior modeling to plan for qualification activities, in particular, validation and acceptance. A sequence diagram (Fig. 8) is an MBSE output that adds temporal insight into functional flow. These diagrams give the SI insight into how the system behaves for given scenarios or use cases. For the SI, this is critical to developing a strategy for acceptance testing. The SI uses the sequence diagram to determine:

- Functional timing dependencies
- Critical timing
- Functional looping and iteration design
- Functional-operational complexity
- External system operational co-dependencies

All of the above alert the SI to establish qualification strategies, especially testing that represent actual system operations. Some common examples:

- Missing internal triggering (inadequate functional behavioral definition preventing I&Q metrics) (Fig. 8c)
- High functional interactions (a function responds to or outputs many triggers and/or results) (Fig. 8c)
- Recursive of iterative looping (loop completion criterion and/ or exit criterion set I&Q key events/metrics)
- Timing structure (identifies synch or synch behavior or centralized/decentralized control structures) (Fig. 8a,b)
- Missing external triggering or results (leads to incomplete I&Q)

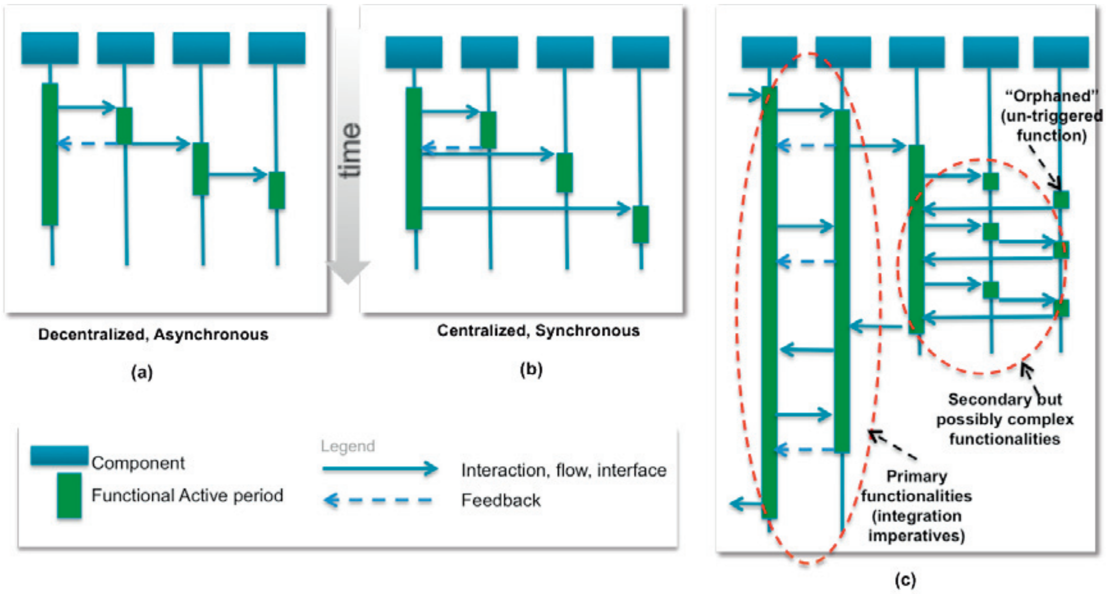

Fig. 8. Sequence diagrams provide SI insight into qualification (particularly acceptance testing) strategies [original]

5. Anecdotal class project experiences / lessons-learned

The author has been able to exercise the above MBSI concepts in a system engineering graduate school environment over 5 different class sections, each with 25 students. The students are DoD acquisition professionals who are pursuing a Master's Degree in Systems Engineering. The students' assignment was to develop a system from needs-to-design-to-acceptance in a very short time period. Teams are formed in diverse geographic areas around the country. Each subsystem team had to design, develop, and integrate their appropriate subsystems. A system integrator (SI) team was responsible for coordinating the system-level requirements, interfaces, architectures, taxonomy, structures, integration strategy, and acceptance test plans. The project was designed such that there was little chance of success if the students went from design-to-development-to-testing in a classic waterfall fashion (a common approach for many real-world system acquisitions). The use of a MBSE tool was prescribed in order to discipline the entire design process. Additionally, the system model was to be completed and integrated *before* starting the actual development of each of the subsystems.

The system was not complex, in and of itself. The challenge was to have multiple, geographically dispersed teams design, develop, and integrate / qualify the system successfully. This represents normal acquisition business models in DoD acquisition. The modeling formed the underpinning of design and integration planning and provided sufficient definition and boundaries for the development of the system. The class MBSE/I efforts included functional, physical, interface, behavioral, and qualification modeling that insured this simple design (but complex integration) could be successfully developed on schedule. Throughout, the students took an SI perspective (MBSI) as discussed in the sections above to inform their technical approach.

In several cases, the teams tried to "shortcut" the modeling process and started development (coding) early. In other cases, interface design or functional interaction considerations were not given high priority. This led to regrouping, returning to model completion, and rework. The rework reinforced the need for prerequisite, successful system modeling. In the end, the teams developed a successful system that passed acceptance on schedule. The lessons learned from the students (derived from several iterations of this course) include:

- SI involvement in design reduces I&Q risks
- Project would have failed without MBSE/MBSI methods
- Early requirements clarification is important (especially interface requirements)
- Early architecture design imperative (especially functional and interface)
- Rushing to development prior to model definition wastes time and effort
- Early model integration drives out:
  - Functional gaps and overlaps

- Interface inconsistencies and discontinuities
- System behavior misunderstandings
- Inter and intra-system interface problems

6. Summary

The SI's perspective is extremely valuable during the earliest phases of system design. The SI brings increased emphasis on critical functional interactions, system interfaces, along with a keen eye toward whether or not the system can be assembled, integrated, verified, and validated in a low risk manner. The SI does not want to *inherit* a system design but, rather, needs to *influence* the design. Although system integration is often portrayed in the SE community as a post-development set of activities, successful I&Q hinges upon the SI being involved early and throughout the entire development process. Additionally, MBSE methods and tools can contribute to alerting an experienced system integrator to early design decisions that may increase risks to successful I&Q. If all subsystems use MBSE tools for system design, then I&Q risks are greatly reduced if these models are first integrated into a complete system model *before* subsystem build / integration. There is, however, still much room for improvement of these tools to develop products, views, metrics, and perspectives that would directly aid any SI, experienced or not, to determine risks of integration qualification and help plan I&Q strategies. Several iterations of graduate SE class projects have tested the essential concepts of early system integrator involvement and MBSI and have anecdotally shown that: *Successful I&Q requires*:

- Strong LSI / SI
- Detailed system definition (particularly interfaces and functional interactions)
- Early taxonomy and structure definition
- Early SI influence with I&Q success perspective
- Modeling in order to discipline design efforts
- Model integration prior to system integration to reduce I&Q risks
- Diverse and integrated SE/SI support system (i.e. tool sets, etc.)
- MBSE tools can provide support to SI but are not yet MBSI tools

References

1. Langford, G. *Engineering Systems Integration*. CRC Press, (2012)
2. *Top 10 Corporate Information Technology Failures*, Retrieved from www.computerworld.com/computerworld/records/images/pdf/44NfailChart.pdf
3. Sage, A. P. and Lynch, C. L. (1998), Systems integration and architecting: An overview of principles, practices, and perspectives. Syst. Engin., 1: 214. doi: 10.1002/(SICI)1520-6858(1998)1:3<176::AID-SYS3>3.0.CO;2-L
4. Haskins, C., Forsberg, K., Krueger, M., Walden, M., and Hamelin, R (eds). *Systems Engineering Handbook*, Section 4.5, INCOSE-TP-2003-002-03.2, INCOSE, (January 2010).
5. Pyster, A., D. Olwell, N. Hutchison, S. Enck, J. Anthony, D. Henry, and A. Squires (eds). 2012. "System Integration", *Guide to the Systems Engineering Body of Knowledge (SEBoK) version 1.0*. Hoboken, NJ: The Trustees of the Stevens Institute of Technology ©2012. Available at: http://www.sebokwiki.org/index.php/article_title.
6. Welby, S. (2010, October 10). *DoD systems engineering update* [Presentation to INCOSE WMA Chapter]. Retrieved from http://www.incosewma.org/category/past-meeting-presentations
7. Bjorkman, E. A., Sarkani, S. and Mazzuchi, T. A. (2012), Using model-based systems engineering as a framework for improving test and evaluation activities. Syst. Engin.. doi: 10.1002/sys.21241
8. Vitech Corporation (2012). *CORE 8 (product documentation)*. Retrieved from http://www.vitechcorp.com/products/core.shtml
9. IBM. (2012). *Rational Rhapsody™ for systems engineering (product documentation)*. Retrieved from http://www-01.ibm.com/software/rational/products/rhapsody/sysarchitect/
10. Piaszczyk, C. (2011), Model Based Systems Engineering with Department of Defense Architectural Framework. Syst. Engin., 14: 305–326. doi: 10.1002/sys.20180
11. Vitech Corporation. (2011, October). *A primer for model-based systems engineering* (2nd ed.). Retrieved from http://www.vitechcorp.com/services/mbse_primer.shtml
12. Tahan, M. and Ben-Asher, J. Z. (2005), Modeling and analysis of integration processes for engineering systems. Syst. Engin., 8: 62–77. doi: 10.1002/sys.20021
13. IDEF0 Functional Modeling Method. Retrieved from http://www.idef.com/idef0.htm