A Framework for Evolving System of Systems Engineering

Dr. Ricardo Valerdi, Dr. Adam M. Ross, and Dr. Donna H. Rhodes
Massachusetts Institute of Technology

We provide a framework for examining the differences between systems engineering and system of systems engineering (SoSE). By taking normative, descriptive, and prescriptive views of these constructs, similarities and differences can be better identified. Moreover, we note that additional work is needed in the development of normative and prescriptive models in order to advance our understanding of both systems engineering and SoSE.

There is an ongoing debate that questions whether there are differences between a system and a system of systems (SoS). For purposes of this discussion, we define a system to be *a construct or collection of different elements that together produce results not obtainable by the elements alone*. The elements – or parts – can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce systems-level results [1]. On the other hand, an SoS is a condition where a majority of the following five characteristics are present: operational independence, managerial independence, geographic distribution, emergent behavior, and evolutionary development [2]. There are numerous definitions for both systems and SoS, but in essence there are three schools of thought that exist on the issue of similarities and differences.

The first school believes that there are fundamental differences between the systems and SoS and, as a result, they warrant different names, methodologies, and tools to bring them to realization. The *pro-difference* camp appears to represent a majority on the debate, as evidenced by the amount of its advocacy across government, industry, and academia. Examples within the last five years include the following:

- Inauguration of the Institute of Electrical and Electronics Engineers (IEEE) Conference on SoS.
- Inception of the International Journal of SoS Engineering.
- Definition of the SoS signature area at Purdue University.
- Creation of the National Center for Systems of Systems Engineering at Old Dominion University.
- Inclusion of SoS considerations in the Systems Engineering Chapter of the Defense Acquisition Guidebook [3].
- Procurement and development of systems uniquely labeled as System-of-Systems such as the Army's Future Combat Systems by Boeing, Science Applications International Corporation, and thousands of subcontractors.
- Creation of the SoS Engineering Center of Excellence by the Office of the Under Secretary of Defense for Acquisition, Technology and Logistics, specifically the Deputy Director of Joint Force Integration.

The second school believes that SoS is simply an unnecessary term for a system, and that society should not be influenced by this subtlety. To follow Rechtin's heuristic: One man's architecture is another man's detail [4]. Similarly, an SoS for one organization may be seen as a system to another. The advocates for this approach believe that traditional systems engineering practices along with effective program management will be sufficient for both systems and SoS.

The third school is agnostic on the issue, playing the *wait-and-see* game. Since there may not be enough evidence to prove that there is or is not a difference, then there is no sense in claiming loyalty to either of the first two schools of thought to avoid the risk of being wrong. This opinion represents the minority, since from a practical perspective there are few engineers who are unwilling to take a side on technical issues.

At the center of the debate about the semantic difference between systems and SoS are the engineering activities that are involved in each. For the definition of systems engineering, for example, we can turn to widely accepted standards [5, 6, 7] used to define the what and how of the activities needed to engineer a system. However, consensus has not been reached on the activities needed to engineer an SoS. Despite the ongoing debate, organizations need to take action on the systems they want to create. But how does one make sense of these concepts? The following section provides a framework to help make sense of this debate.

Normative, Descriptive, and Prescriptive Framework

To help structure the discussion, a framework is proposed that provides a categorization of concepts into normative, descriptive, and prescriptive models. For purposes of this discussion, the term model is used to include formal and informal representations of structure.

A normative model is one that represents norms or cultural standards. Similarly, a normative statement describes how the world *should* be; it provides a yardstick to measure whether something is good. One example is the IEEE 1220 standard entitled *Application and Management of the Systems Engineering Process* [5]. As a normative model, it describes how systems engineering ought to be implemented for the realization of successful programs. Because these standards result from reflection and analysis, normative models are representative of the type of models developed in philosophy, mathematics, and other theory-based disciplines [8]. In the domain of systems, the development of normative models relies on assumptions about how systems will be implemented, and through interpreting the implementation, suggests the best way to build systems. Normative models are analogous to first principles of nature since they are foundational propositions from which all other propositions can be deduced.

A descriptive model characterizes actual behavior of decision-makers, or how the world *actually is*. An example is the set of systems engineering case studies developed by the Center for Systems Engineering at the Air Force Institute of Technology [9]. These include detailed accounts of how systems engineering was actually done on large programs such as the B-2 stealth bomber, Theater Battle Management Core System, and the Joint Air-to-Surface Standoff Missile. As descriptive models, these case studies are a representation of our understanding of how systems engineering has been done in the past, whether good or bad. Deviations from a normative model (i.e., an IEEE standard) can be highlighted through a descriptive model. Descriptive models capture not only behavior, but also decision-making, insofar as the outcomes of the decisions are discernable, or elicited from decision-makers. A classical example of a descriptive model is a regression analysis on empirical data. The *best fit* curve to a set of data captures the essence of empirical reality and can be used to predict the future but is limited by the context and conditions of the original data underlying the model.

Descriptive models also provide insight into organizational practices of systems engineering. Some organizations, for example, may think that it is appropriate to allocate systems engineering functions (i.e., testing, documentation) to software engineers while others may decide to treat those as completely separate functions. A descriptive model would relate these differences without a justification in terms of a normative *right* or *wrong* basis, but rather just in terms of stating the specific process used in organizations.

A prescriptive model is one that aims at correcting biases with the intent to improve judgments and decisions according to normative standards. In other words, a prescriptive model is based on advice on how to best achieve the ideals suggested by the normative view, given the facts highlighted through the descriptive view. An example of a prescriptive model is one that provides practical directions on how to implement process improvement through Capability Maturity Model Integration (CMMI®) in organizations that develop systems or SoS. It contains the prescriptive representation of how it *can* be done and a strategy on how to get from the present state (descriptive) to ideal state (normative) as viewed by best practices in the software defense industry.

Equipped with these concepts, the differences between systems and SoS can be considered through a more analytical perspective. First, a descriptive view is provided, followed by a discussion on how normative and prescriptive models can be developed to further compare systems and systems of systems.

A Descriptive View of Systems

Clearly, there are similarities between systems and SoS. Some of these similarities are the following:

- Both can be very complex.
- Both involve people, hardware, software, facilities, policies, processes, etc.
- Both have purpose.

Even though both systems and SoS share many traits, it is the difference between them that provides the basis for debate. A helpful descriptive view of the issue is to compare what is involved with engineering a system to engineering an SoS. Academia [10] and government [11] have highlighted key differences between the two. We provide an adaptation of previous work [12] that highlights salient descriptive differences between traditional systems engineering practice and SoSE practice (see Table 1).

Note that not all systems will contain the attributes in the middle column and not all SoS will contain the attributes of the third column, but in general a SoS always requires at least some of the elements of SoSE.

Presently, there are a number of descriptive research efforts that are helping inform the debate about the differences between system and SoS. In the area of cost estimation, work is being done to define the unique attributes of SoS for purposes of developing a cost model to estimate the SoS integration effort [13, 14]. This work was motivated by the inability of existing system-level cost models to estimate the effort needed to integrate SoS.

	Traditional Systems Engineering	SoSE
Purpose	Development of a single system to meet stakeholder requirements and defined performance.	Evolving new system of systems capability by leveraging synergies of legacy systems and emerging capabilities.
Systems Architecture	Established early in the life cycle; expectation set remains relatively stable.	Dynamic adaptation as emergent needs change.
System Interoperability	Interface requirements are defined and implemented for the integration of components in the system.	Component systems can operate independently of SoS in a useful manner; protocols and standards are essential to enable interoperable systems.
System *ilities*	Reliability, maintainability, and availability are typical concerns.	Enhanced emphasis on *ilities* such as flexibility, adaptability, and composability.
Acquisition and Management	Centralized acquisition and management of the system.	Component systems separately acquired and continue to be managed and operated as independent systems.
Anticipation of Needs	Concept phase activity to determine system needs.	Intense concept phase analysis followed by continuous anticipation, aided by ongoing experimentation.
Cost	Single or homogenous stakeholder group with stable cost/funding profile and similar measures of success.	Multiple heterogeneous stakeholder groups with unstable cost/funding profile and measures of success.

Table 1: *Areas of Emphasis in Systems Engineering and SoSE [12]*

As a first step toward a prescriptive model, the Department of Defense (DoD) is developing a Guide to System of Systems Engineering [11], based on best present state knowledge. The guide provides 16 DoD technical and management processes to help sponsors, program managers, and chief engineers address the unique considerations for DoD SoS. Results from these efforts will serve to further characterize the state of the practice of systems engineering and may begin to identify enablers, barriers, and successful techniques for SoS engineering practice. Ultimately, the observation of real systems and SoS enable the development of prescriptive models that are grounded in reality and eventually generate useful prescriptive guidance. It is important to recognize that implicit in any prescriptive model is a normative basis for why a particular guidance is suggested. Users of such guidance should make an effort to recognize that basis to better understand the rationale for the guidance and under what conditions that guidance may be ill-advised.

Ideally, a normative model is developed prior to any prescriptive model. In a practical sense, these often evolve a bit differently. For example, the DoD Guide to System of Systems Engineering implicitly relies upon normative models for traditional systems engineering [15], due to a lack of SoS normative models [16]. In the future, normative models for SoS are needed in order to provide a sound and defensible basis for *good* SoSE practice. Researchers and practitioners alike should strive towards a better understanding of successful systems and SoS and the development of normative models to improve understanding of the fundamental differences between successful systems and SoS. Recent efforts to define a research agenda for SoS architecting [17] demonstrate the need for government, academia, and industry to work together to advance the state of affairs.

Practical Implications

The immaturity of normative models as discussed has several implications for acquirers and implementers of SoS. First, there are no industry best practices that can be used as normative models to compare how SoS acquisitions and implementations should be done. Such a model would describe the rationale for how *best* is defined and how it could be accomplished in an ideal scenario. A natural progression suggests that descriptive models will come first through the observation of successful and unsuccessful SoS. Subsequently, understanding how SoS should be acquired and implemented are derived into normative models. Once a descriptive model of the present and a normative model of the *ideal* future are developed, a thorough benchmarking process can begin. The development of practical strategies and processes, such as those expressed in a guide, require the codification of prescriptive models. Since prescriptive models require descriptive and normative models as their basis, the better the descriptive and normative models, the better the prescriptive advice.

A second implication is that there is a lack of reference costs and schedule estimates for SoS. This is a crucial void for predictive estimation models since they require historical data for their development and calibration. This also emphasizes the need to capture lessons learned from completed programs to develop descriptive models.

Returning to the debate on how the engineering of a system differs from that of a SoS requires a sound theoretical and practical foundation on what these two processes entail. Much of the research to date on SoS involves descriptive type approaches. Advancing prescriptive guidance on how to do SoSE requires a normative model as well. The absence of validated normative models presents an opportunity for industry, government, and academia to collaborate. It is only when that piece falls into place that a rigorous and defensible development of a standardized and successful SoSE process can take place.◆

References

1. Rechtin, E. The Art of Systems Architecting. New York: CRC Press, 2000.
2. Sage, A., and C. Cuppan. "On the Systems Engineering and Management of Systems of Systems and Federations of Systems." Information, Knowledge, and Systems Management 2 (2001): 325-345.
3. Defense Acquisition University (DAU). Defense Acquisition Guidebook. Vers. 1.06. DAU, 2006.
4. Rechtin, E. Systems Architecting: Creating and Building Complex Systems. New Jersey: Prentice Hall, 1990.
5. "Application and Management of the Systems Engineering Process." IEEE 1220, 1998.
6. American National Standards Institute /Electronic Industry Alliance (ANSI/EIA). "Processes for Engineering a System." ANSI/EIA-632-1988. ANSI/EIA, 1999.
7. International Standards Organization/International Electrotechnical Commission (ISO/IEC). "Systems Engineering – System Life Cycle Processes." ISO/IEC 15288:2002(E). ISO/IEC, 2002.
8. Baron, J. Thinking and Deciding. 3rd ed. Cambridge University Press, 2000.
9. Systems Engineering Case Studies. Air Force Center for Systems Engineering <www.afit.edu/cse/cases.cfm>.
10. Keating, C., et al. "System of Systems Engineering." Engineering Management Journal 15.3 (2003).
11. Office of the Under Secretary of Defense. System of Systems Systems Engineering Guide: Considerations for Systems Engineering in a System of Systems Environment. Vers. 0.9. DoD, 2006 <www.acq.osd.mil/se/publications.htm>.
12. Rhodes, D. "Evolving Systems Engineering for Innovative Product and Systems Development." Proc. From Massachusetts Institute of Technology (MIT) Systems Design and Management Alumni Conference, Oct. 2004.
13. Lane, J., and R. Valerdi. "Synthesizing SoS Concepts for Use in Cost Estimation." Systems Engineering 10.4 (2007).
14. Lane, J. "System of Systems Lead System Integrators: Where Do They Spend Their Time and What Makes Them More/Less Efficient?" USC-CSE-2005-508. Los Angeles: USC Center for Systems and Software Engineering, June 2005.
15. Sage, A.P. Systems Engineering. Wiley-Interscience, 1992.
16. U.S. Air Force. "Report on System of Systems Engineering for Air Force Capability Development." SAB-TR-05-04. July, 2005.
17. Axelband, E., et al. "A Research Agenda for Systems of Systems Architecting." Proc. of the 17th INCOSE Symposium, June 2007, San Diego, CA.

About the Authors

Ricardo Valerdi, Ph.D., is a research associate in the Systems Engineering Advancement Research Initiative (SEAri) and a lecturer in the Engineering Systems Division at MIT. His research interests are in systems engineering cost estimation and the synergies between systems engineering and software engineering. Valerdi has a doctorate in industrial and systems engineering from USC.

MIT
77 Massachusetts AVE
41-205
Cambridge, MA 02139
Phone: (617) 253-8583
Fax: (617) 258-7845
E-mail: rvalerdi@mit.edu

Adam M. Ross, Ph.D., is a research scientist in the SEAri at MIT. His research interests are in the exploration of system architecture tradespaces, and the advancement of dynamic value considerations through quantification and visualization during the system design process. Ross has a doctorate in engineering systems from MIT.

MIT
77 Massachusetts AVE
NE20-388
Cambridge, MA 02139
Phone: (617) 324-0473
Fax: (617) 258-7845
E-mail: adamross@mit.edu

Donna H. Rhodes, Ph.D., is the director of SEAri, Principal Researcher, and is a senior lecturer in the Engineering Systems Division at MIT. Her research interests are in advanced systems engineering methods and practices. Rhodes has a doctorate in systems science from the State University of New York, Binghamton.

MIT
77 Massachusetts AVE
NE20-388
Cambridge, MA 02139
Phone: (617) 324-0473
Fax: (617) 258-7845
E-mail: rhodes@mit.edu