Architecting Approaches for Systems of Systems

> The printable version is no longer supported and may have rendering errors. Please update your browser bookmarks and please use the default browser print function instead.

Lead Authors: *Judith Dahmann, Bud Lawson, Mike Henshaw*

A key part of systems engineering (SE) for system of systems (SoS) is the composition of systems to meet SoS needs. This may include simply interfacing systems and leveraging their existing functionality or it may require changing the systems functionality, performance or interfaces. These changes occur incrementally, as a SoS evolves over time to meet changing SoS objectives. System of Systems Engineering (SoSE) supports these changes by developing and evolving a technical framework that acts as an overlay to the systems of which the SoS is composed. This framework provides the *architecture* for the SoS. The SoS architecture defines how the systems work together to meet SoS objectives and considers the details of the individual systems and their impact the SoS performance or functionality.

☐

Contents

The Role of System of Systems Architecting

An architecture is the structure of components, their relationships, and the principles and guidelines governing their design evolution over time (IEEE 610.12-1990).

In a SoS, the architecture is the technical framework for the systems comprising the SoS which designates how the systems will be employed by the users in an operational setting (sometimes called the concept of operations (CONOPs or CONOPs), the internal and external relationships and dependencies among the constituent systems and their functions and, finally, the end-to-end functionality and data flow as well as communications among the systems.

Because SoS largely comprise extant independent systems, these place constraints on the SoS architecture and may require that migration to a SoS architecture be incremental. Developing a SoS architecture requires consideration of technical feasibility for the constituent systems as well as the needs of the SoS itself. Architecture data for the constituent systems can also be important data for architecting the SoS. There is some similarity here to the introduction of Commercial Off The Shelf (COTS) products into systems: the COTS product has been independently managed but sufficient data is required by the systems developer to ensure satisfactory integration. However, in this case the COTS product is not independently operated

Maier (1998) provides a conceptual discussion on the impact of SoS characteristics on SoS architecting. Additionally, the US DoD SE Guide for SoS (2008) describes practical considerations in developing and evolving a SoS architecture as a core element of SoSE.

Challenges in Architecting SoS

In the case of a new system development, the systems engineer can begin with a fresh, unencumbered approach to architecture. However, in a SoS, the systems contributing to the SoS objectives are typically in place when the SoS is established and the SoS engineer needs to consider the current state and plans of the individual systems as factors in developing an

architecture for the SoS. This, along with the fact that constituent systems may be complex systems in their own right, leads to a set of challenges to architecting SoS. The approach to architecting must be determined by the type of SoS under consideration. Whereas a directed SoS can be architected in much the same way as a monolithic system, the other types are less straightforward, because the SOI may be less clearly defined and because the SoS architects knowledge of the constituent systems may be partial. Furthermore, whereas in a directed SoS the owner may have authority and funding to require architectural changes of constituent systems, in acknowledged and collaborative SoS re-architecting is at the discretion of the owners of the constituent systems. Maier (Maier 1998) has focused architecting attention on communication for SoS, arguing that this is the common feature of all types, and he partitions the communication into layers that have a similarity to the layers of interoperability (NCOIC, 2008).

The independence of the constituent systems means that these systems are typically not designed to optimize SoS objectives. It may even be the case that a constituent system should operate sub-optimally at the system level in order to achieve overall SoS effectiveness. (Rebovich 2009) has articulated this difficulty as a fundamental problem of SoS:

> *From the single-system community's perspective, its part of the SoS capability represents additional obligations, constraints and complexities. Rarely is participation in an (sic) SoS seen as a net gain from the viewpoint of single-system stakeholders.*

The development and implementation of a SoS architecture may be significantly constrained by a reluctance to make changes or invest in the constituent systems, which could be very mature (e.g. in sustainment) or currently productively supporting other uses. In this case, approaches such as gateways and wrapping may be used to incorporate these systems into the SoS without making significant changes in the other systems.

Architecture Analysis

Large-scale systems integration has grown in importance and correspondingly, there has been a growing interest

in SoS concepts and strategies. The performance and functioning of groups of heterogeneous systems has become the focus of various applications including military, security, aerospace, distributed energy, healthcare, and disaster management systems (Lopez 2006; Wojcik and Hoffman 2006). There is an increasing interest in exploiting synergy between these independent systems to achieve the desired overall system performance (Azarnoush et al. 2006).

Modeling and simulation is conducted to analyze architecture effectiveness and to verify architectural features. In the literature, researchers have addressed the issues of coordination and interoperability in a SoS (Abel and Sukkarieh 2006; Sahin et al. 2007). In order to study SoS characteristics and parameters, one needs to have realistic simulation frameworks properly designed for system of systems architecture. There are some attempts to develop simulation frameworks for multi-agent systems using Discrete Event Simulation (DEVS) tools (Zeigler et al. 2000a). In these research efforts, the major focus is given to DEVS architecture with JAVA. In (Mittall 2000), DEVS state machine approach is introduced. Finally, DEVS Modeling Language (DEVSML) is developed by using XML based JAVA in order to simulate systems in a net-centric way with relative ease. Sahin et al. (2007) have recently introduced a discrete event XML based SoS simulation framework based on DEVS and JAVA.

The Open Approach to SoS Engineering

As noted above, one of the key challenges with SoS architecting is that the constituent systems of a SoS may not have been designed, developed and employed with regard to their role in the SoS, which constrains SoS architecture options. The degree to the architecture which *overlays* these constituent systems and supports the SoS end-to-end capabilities can be based on open standards; the SoS may be able to benefit from open architecture for future evolution.

The critical challenge of moving from SoS, as a concept to the engineering of SoS, is the significant technological, human, and organizational differences in consideration system of systems engineering and management approaches (Wells and Sage 2008). A potential approach to engineering a SoS can be the *open systems approach* to SoSE (Azani 2009). The following open systems principles are listed by Azani (2009):

- **Open interface principle** - Open systems have permeable boundaries that allow them to exchange mass, energy, and information with other systems;
- **Synergism principle** – The notion that designates that the co-operative interaction between constituent systems has a greater effect in their combined efforts than the sum of their individual parts. Essentially, this is what gives rise to emergence;
- **Self-government principle** - This implies that the SoS maintains and develops its internal order without interference from external sources. This could be through cybernetic control, homeostasis, or self-organization;
- **Emergence principle** - In this case, this refers to the occurrence of novel and coherent structures, patterns, and properties during the self-organization of the SoS;
- **Conservation principle** – This principle states that energy and mass (material) are conserved within the SoS;
- **Reconfiguration principle** – This refers to the SoS reconfiguring and adapting itself to sustain itself against changes in its environment;
- **Symbiosis principle** - The systems within the SoS have a symbiotic relationship to each other; more transparently, the successful development and sustainment of a SoS depends on symbiotic collaboration between the stakeholders of the systems of which it is comprised; and
- **Modularity principle** - This holds that each level and each system is to some extent independent of others. In SoS design, the development of independent modular systems that interoperate with each other through standardized interfaces enables greater flexibility to promote better evolution of the SoS.

Azani (2009) elaborates on the open systems development strategies and principles utilized by biotic SoS, discusses the implications of engineering of man-made SoS, and introduces an integrated SoS development methodology for the engineering and development of an adaptable, sustainable, and interoperable SoS based on open systems principles and strategies.

Hitchens (2003, 107), on the other hand, discusses the principles of open systems rather differently in terms of their systems life cycles, as the seven principles that he

addresses are system reactions, system cohesion, system adaptation, connected variety, limited variety, preferred patterns, and cyclic progression. This description takes a systems dynamics approach to show how open systems evolve; the description is applicable to natural and man-made systems.

The enablers of openness include open standards and open specifications, which draw from consensus amongst a community of interest, and are published by, and freely available within, that community. An open specification must ensure that its detail-level is allows for it to be implementable by independent parties. Compliance with open standards is intended to ensure consistent results (Henshaw, et. al., 2011). This parallels the notion of open systems architecture, which is an open specification of the architecture of a system or system of systems for the purpose of acquiring specified capabilities. As a general feature of good design (for a system or system of systems), an open system architecture should allow for the quick and easy improvement and updating of the system capabilities, by adding or changing components. However, Henshaw et. al. (2011) also denote that open architecture represents a commercial challenge (rather than a technical one) and that establishing open architecture approaches to acquisition can be challenging, due to issues involving protection of intellectual property (IP) and commercial advantage.

Networks and Network Analysis

Because networks are such a common component of SoS, they warrant specific attention. In SoS that are based on an underlying network, communications and information exchange typically constitute a SoS in its own right. This enabling SoS requires architecting like any other SoS, which will be addressed in this section. In the case of an enabling network SoS, the 'user', the end-to-end functionality of the larger SoS and enabling network SoS is driven by these user needs. The relationship between SoSE concepts and network enablement, as well as the concepts of networks and network analysis that extend beyond information sharing, have been explored extensively by the defense community (Dickerson and Mavris 2009). For instance, during the U.S. Navy's work on command, control, communications, computers, intelligence, surveillance, and reconnaissance (C4ISR) as part of a SoS (Owens 1996), the term network included organizational aspects of command and control (C2) structure as well as

communications.

Differences in the architecting of an enabling network SoS derive from the fact that these SoS are typically built on commercial technologies and architectures, which are changing rapidly in today's dynamic technological environment. In addition, these enabling networks are often shared among SoS and hence may further constrain the overall SoS architecture. For example, many SoS (for cost and convenience) expect to operate over the internet, and therefore must consider characteristics of the internet in the expectations for performance and security provided by use of a shared enabling infrastructure.

Enabling network SoS architecting is particularly well-served by the initial analysis that explores sensitivities through modeling, simulation, analysis, and/or laboratory experimentation and identifies scalability issues or divergent behavior (e.g., concerning requirements or usage assumptions, assumed network bandwidth, or others), beyond which performance starts to break down. This type of analysis provides a basis for network architecture decisions.

In directed SoS, because of the top-down control, there is the option for creating a specialized network for the particular SoS. In the other types of SoS, if the constituents are already supported by some type of a network then the overall SoS networking approach typically needs to accommodate these since the constituent systems are likely to need to continue to use their current approach to support their original users.

Interoperability

Interoperability within a SoS implies that each system can communicate and interact (control) with any other system regardless of their hardware and software characteristics or nature. This implies that each constituent member (and potential new members) of a SoS should be able to communicate with others without compatibility issues in the operating systems, communication hardware, and so on. For this purpose, a SoS needs a common language the SoS's systems can speak. Challenges here are to work towards a common language for exchange of information and data among systems of a SoS. Examples of such system are XML (eXtensible Markup Language), as one potential candidate (Jamshidi, 2009a).

However, interoperability must be achieved at many

levels and not just at the data/network level. There are a number of frameworks that describe the levels of interoperability. From military applications, the NCOIC (Network Centric Operations Industry Consortium) Interoperability Framework (NCOIC 2008) covers three broad levels of interoperability, subdivided into further layers as indicated below:

- Network Transport:
    - Physical Interoperability and
    - Connectivity and Network Interoperability;

- Information Services:
    - Data/Object Model Interoperability,
    - Semantic/Information Interoperability, and
    - Knowledge/Awareness of Actions Interoperability; and

- People, Processes and Applications:
    - Aligned Procedures,
    - Aligned Operations,
    - Harmonized Strategy/Doctrine, and
    - Political or Business Objectives.

This spectrum of interoperability layers requires appropriate coherence at each layer consistent with the SoS shared goals.

There exist interoperability frameworks in other fields of activity. An example is the European Interoperability Framework (European Commission 2004), which focuses on enabling business (particularly e-business) interoperability and has four levels within a political context:

- Legal Interoperability,
- Organizational Interoperability,
- Semantic Interoperability, and
- Technical Interoperability.

The interoperability between the component systems of a SoS is a fundamental design consideration for SoS that may be managed through the application of standards.

References

Works Cited

Abel, A., and S. Sukkarieh. 2006. "The Coordination of Multiple Autonomous Systems using Information Theoretic Political Science Voting Models." Proceedings of the IEEE International Conference on System of Systems Engineering, April 24-26, 2006, Los Angeles, CA, USA.

Azani, C. 2009. *A Multi-criteria Decision Model for Migrating Legacy System Architectures into Open Systems and Systems-of-Systems Architectures.* Washington, DC, USA: Defense Acquisition University.

Azarnoush, H., B. Horan, P. Sridhar, A.M. Madni, and M. Jamshidi. 2006. "Towards optimization of a real-world robotic-sensor system of systems". Proceedings of the World Automation Congress (WAC), July 24-26, 2006, Budapest, Hungary.

Cloutier, R.M., J. DiMario, and H.W. Polzer. 2009. "Net-Centricity and System of Systems," in *Systems of Systems Engineering - Principles and Applications*, edited by M. Jamshidi. Boca Raton, FL, USA: CRC Press.

Dagli, C.H., and N. Kilicay-Ergin. 2009. "System of Systems Architecting," in *Systems of Systems Engineering - Principles and Applications*, edited by M. Jamshidi. Boca Raton, FL, USA: CRC Press.

Dickerson, C.E., and D. Mavris. (2009) *Architecture and Principles of Systems Engineering.* New York, NY, USA: CRC Press, Auerbach Publications.

DoD. 1998. *Levels of Information System Interoperability.* Washington, DC, USA: C4IST Interoperability Working Group, US Department of Defense.

Hall, J. 2007. "Openness – An Important Principle For The Stewardship of DoD IT Standards." *DSPO Journal*, 4-7. Available: http://www.dsp.dla.mil/app_uil/content/newsletters/journal/DSPJ-01-07.pdf.

Henshaw, M. (ed.). 2011. *Assessment of open architectures within defence procurement issue 1: systems of systems approach community forum working group 1 - open systems and architectures.* London, UK: SoSA Community Forum Working Group 1 (Crown owned copyright). Available: https://dspace.lboro.ac.uk/2134/8828.

Hitchins, D.K. 2003. *Advanced Systems Thinking, Engineering and Management*. Norwood, MA, USA: Artech House, Inc.

IEEE. 1990. IEEE 610.12-1990, *Standard Glossary of Software Engineering Terminology*. Washington, DC, USA: Institute of Electrical & Electronics Engineers (IEEE).

Jamshidi, M. (ed.) 2009. *Systems of Systems Engineering - Principles and Applications*. Boca Raton, FL, USA: CRC Press.

Johnson M. 2009. "System of Systems Standards," in *System of Systems Engineering - Innovations for the 21st Century*. Hoboken, NJ, USA: Wiley.

Lopez D. 2006. "Lessons Learned From the Front Lines of the Aerospace." Proceedings of the IEEE International Conference on System of Systems Engineering, April 24-26, 2006, Los Angeles, CA, US.

Mittal, S. 2000. "DEVS Unified Process for Integrated Development and Testing of Service Oriented Architectures." PhD Dissertation. Tucson, AZ, USA: University of Arizona.

NCOIC. 2008. "NCOIC Interoperability Framework (NIF(R))." Available: http://www.ncoic.org/technology/technical-products/frameworks/10-technology/33-tech-prod-framework-nif.

Owens, W.A. 1996. *The Emerging U.S. System-of-Systems*. Washington, DC, USA: The National Defense University, Institute of National Security Studies.

Rebovich Jr., G. 2009. "Chapter 6: Enterprise System of Systems," in *Systems of Systems Engineering - Principles and Applications.* Boca Raton, FL, USA: CRC Press. p. 169.

Sahin, F., M. Jamshidi, and P. Sridhar. 2007. "A Discrete Event XML based Simulation Framework for System of Systems Architectures." Proceedings of the IEEE International Conference on System of Systems, April 16-18, 2007, San Antonio, TX, USA.

US Department of Defense. 2008. *Systems Engineering Guide for Systems of Systems,* version 1.0. Washington, DC, USA: US Department of Defense.

Wells, G.D., and A.P. Sage. 2008. "Engineering of a System of Systems," in *Systems of Systems Engineering -*

Principles and Applications. Boca Raton, FL, USA: CRC Press.

Wojcik, L.A., and K.C. Hoffman. 2006. "Systems of Systems Engineering in the Enterprise Context: A Unifying Framework for Dynamics." Proceedings of the IEEE International Conference on System of Systems Engineering, April 24-26, 2006, Los Angeles, CA, USA.

Zachmann, J. 1987. "A framework for information systems architecture." *IBM Systems Journal.* 26 (3).

Zeigler, B.P., T.G. Kim, and H. Praehofer. 2000. *Theory of Modeling and Simulation.* New York, NY, USA: Academic Press.

Primary References

Chen, D., G. Doumeingts, F. Vernadat. 2008. "Architectures for Enterprise Integration and Interoperability: Past, Present and Future." *Comput.Ind.* 59 (7):647-659.

Maier, M.W. 1998. "Architecting Principles for Systems-of-Systems." *Systems Engineering.* 1 (4): 267-284.

Additional References

Dickerson, C.E., S.M. Soules, M.R. Sabins, and P.H. Charles. 2004. *Using Architectures for Research, Development, and Acquisition.* Washington, DC, USA: Office of the Assistant Secretary of The Navy (Research Development And Acquisition). ADA427961. Available: http://handle.dtic.mil/100.2/ADA427961.

European Commission. 2010. "Annex to the Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of Region," in *Towards interoperability for European public services.* Available: http://ec.eupora.eu/isa/strategy/doc/annex_ii_eif_en.pdf

Giachetti, R.E. 2010. *Design of Enterprise Systems, Theory, Architecture, and Methods.* Boca Raton, FL, USA: CRC Press.

Rhodes, D.H., A.M. Ross, and D.J. Nightingale. 2010. "Architecting the System of Systems Enterprise: Enabling Constructs and Methods from the Field of Engineering Systems." IEEE International Systems

Conference, March 23-26, 2009, Vancouver, Canada.

MITRE. 2012. "Architectures Federation," in *Systems Engineering Guide.* Bedford, MA, USA: MITRE Corporation. Accessed September 11, 2012. Available: http://www.mitre.org/work/systems_engineering/guide/enterprise_engineering/engineering_info_intensive_enterprises/architectures_federation.html.

Mittal, S. 2000. "Extending DoDAF to Allow DEVS-Based Modeling and Simulation." *Journal of Defense Modeling and Simulation (JDMS).* 3 (2).

Valerdi R., E. Axelband, T. Baehren, B. Boehm, and D. Dorenbos. 2008. "A research agenda for systems of systems architecting." *International Journal of System of Systems Engineering.* 1 (1-2): 171-188.

Zeigler, B.P., D. Fulton, P. Hammonds, and J. Nutaro. 2000. "Framework for M&S–Based System Development and Testing in a Net-Centric Environment." *ITEA Journal of Test and Evaluation.* 26 (3): 21-34.

< Previous Article | Parent Article | Next Article >
SEBoK v. 2.11, released 25 November 2024