
Background for COSOSIMO

by

Jo Ann Lane

jolane@usc.edu

University of Southern California
Center for Software Engineering
June 2005
ABSTRACT
As organizations strive to expand system capabilities through the development of system-of-systems (SoS) architectures, they want to know "how much effort" and "how long". In order to answer these questions, it is important to first understand the types of activities performed in SoS architecture development and integration and how these vary across different SoS implementations. This paper provides preliminary results of research conducted to determine types of SoS Lead System Integrator (LSI) activities and how these differ from the more traditional system engineering activities described in EIA 632 (Processes for Engineering a System). It also looks at concepts in organizational theory, complex adaptive systems, and chaos theory and how these might be applied to SoS LSI activities to improve success rates and efficiency in the development of these “very large” complex systems.

Introduction
As organizations strive to expand system capabilities through the development of system-of-systems (SoS) architectures, they want to know "how much effort" and "how long". Efforts are currently underway at the University of Southern California (USC) Center for Software Engineering (CSE) to develop a cost model to estimate the effort associated with SoS Lead System Integrator (LSI) activities. The research described in this paper is in support of the development of this cost model, COSOSIMO. Research conducted to date in this area has focused more on technical characteristics of the SoS. Feedback from USC CSE industry affiliates [7] indicates that the extreme complexity typically associated with SoS architectures and political issues between participating organizations have a major impact on the LSI effort. This is also supported by surveys of system acquisition managers [1] and studies of failed programs [11].

The focus of this directed research project is to further investigate effort and schedule issues on “very large” SoS programs such as Future Combat Systems (FCS) and to determine organizational characteristics that significantly impact overall success and productivity of the program.

Scope
Because the USC CSE Industry affiliates and other surveys and studies have indicated that organizational and political issues can significantly impact the success of very large, complex system development efforts as well as the productivity and total effort required to accomplish the system development objectives, we continue to investigate related theories, studies, and surveys in this area. The specific goals of the research effort presented in this paper are to:

1. More clearly define the LSI activities for which the model will estimate effort and show how they differ from the more traditional system engineering activities described in EIA 632 [4].

2. Identify organizational factors from organizational, complexity, and chaos theory that may apply to LSI activities.
3. Analyze actual effort data from several LSI teams with different organizational characteristics, and through this analysis, identify sets of LSI activities that are impacted by each factor identified in (2) above.

As indicated above, initial plans for this research included the collection and analysis of effort data from sample SoS activities. However, due to contract delays, this data was not available for this research effort. Therefore, this paper only presents the results of 1 and 2 above.

**Background**

The first step in developing a cost model is to describe the effort you are trying to estimate as well as the characteristics of the activities that you are trying to estimate. Key to the COSOSIMO cost estimation model is understanding what an SoS is, how it is different from the more traditional system, and what are the major sources of effort.

**What is an SoS:** So to begin, we started with a basic definition of an SoS. An initial survey of the literature shows that there are a considerable number of definitions of SoSs. It also became clear during this survey that it would be extremely difficult, if not impossible to develop a single cost model that could handle all of the various types of SoSs. Therefore, we looked at defining a set of SoSs that would benefit from a cost model [10] and tailoring the development of COSOSIMO to that set of SoSs. After reviewing the various SoS concepts, the authors concluded that the SoSs of interest to the cost model development effort had the following characteristics:

- SoSs are very-large systems developed by creating a framework or architecture to integrate existing systems, systems currently under development, as well as new systems still to be developed. Many of the existing SoS system components were initially developed as a closed, stovepipe type of system that cannot easily share information with other systems.

- There are SoS sponsoring organizations and user communities that have a long-term vision at some level of what the SoS should be, are responsible for the development and maintenance of the SoS, and are interested in development cost information and tradeoffs based on SoS characteristics and development approaches. (This is in opposition to business systems that are integrated over time with no real long-term vision or strategy, but rather based on current fiscal funding and new technologies on the near-term horizon.)

- SoS development activities are planned and coordinated by an LSI organization.

- SoS system components are typically systems that can operate on their own outside of the SoS framework. They are also independently developed and managed throughout their full lifecycle by organizations other than the LSI.

As shown in Figure 1, most SoS architectures are described as net-centric architectures that allow the system components to exchange information and perform tasks within the framework that they are not capable of performing on their own outside of the framework. This is often referred to as an “emergent behavior”. Key issues in developing an SoS are the security of information shared between the various system components, how to get the right information to the right destinations efficiently without
overwhelming users with unnecessary or obsolete information, and how to maintain dynamic networks so that “nodes” can come and go.

An example that illustrates the types of enterprise capabilities that organizations attempt to integrate through an SoS might be a county sheriff’s system to provide more detailed information to deputies in the field to support their current investigations. The deputies connect to the SoS via wireless laptop computers in their cars. Through this connection, they can receive detailed information from various county agencies. This might include county maps showing property boundaries, utility easements and locations, floor plans for residences of interest, whether there are any known vicious animals at a given residence, wanted and warrants information on suspected criminals that include photographs of the suspect, and live feeds from helicopter cameras in an area of interest. In addition, once they apprehend a suspect, they can begin the booking process into the central jail information management system on their way back to the jail.

**SoS LSI Activities:** Next, we looked at better defining the typical LSI activities and comparing them to the more traditional system engineering activities. Based on information gleaned from LSI statements of work, most of the LSI effort focuses on defining the SoS concepts and overall architecture; conducting source-selection activities to acquire system components to be integrated into the SoS framework; managing changes to the SoS-level architecture, communications requirements, and design; tracking changes to the system components that are happening in parallel with the SoS development activities; and integrating and testing the SoS system components at the SoS level. Because of the complexity and high requirements volatility of these SoSs, they are often developed using an incremental or spiral development process that spans many years.

**COSOSIMO Cost Model Overview:** As stated earlier and shown in Figure 2, COSOSIMO estimates the LSI effort associated with SoS definition, system component supplier acquisition and oversight, change management, and SoS-level integration and test activities. This effort is calculated based on a set of user-specified size drivers and scale factors. The current proposed set of size drivers and scale factors are described in [8]. These proposed size drivers and scale factors are still being analyzed for completeness and sufficiency with respect to the desired cost model results.
The eventual goal is for the model to provide effort data by key LSI activity and phase of development, as done for some of the other COCOMO suite models. Figure 3 shows an example effort distribution associated with the system engineering cost model, COSYSIMO [14]. A detailed description of the EIA 632 activities [4] are provided in Appendix A.

Note that the effort associated with SoS component system engineering and software development is not considered as part of COSOSIMO since it is handled by other existing cost models [3].

**How Do SoS Activities Differ from Traditional System Engineering Activities**

The question often asked is how do the COSOSIMO LSI activities differ from the more traditional system engineering activities. To better explain these differences, the SoS LSI
activities have been analyzed [9] with respect to the types of issues they typically face and to the applicable process areas in the Software Engineering Institute (SEI) Capability Maturity Model Integration (CMMI\textsuperscript{SM}) project management and engineering process categories described in [13]. The results of this analysis are described in the following paragraphs.

**LSI Activity Overview [2,9]:** Once an LSI (or LSI team) is under contract to develop a system-of-systems, they quickly begin to concurrently define the scope of the SoS, plan the activities to be performed, analyze the requirements, and start developing the SoS architecture/framework. As the scope, requirements, and architecture start to firm up, the LSI begins source selection activities to identify the desired system component suppliers. As the suppliers start coming on board, the LSI organization must focus on teambuilding, re-architecting, and feasibility assurance with the selected suppliers. Teambuilding is critical since the LSI organization(s) and the selected suppliers have often been competitors in the past and now must work together as an efficient, integrated team. Re-architecting is necessary to make adjustments for the selected system components that may not be compatible with the initial SoS architecture or other selected components. And feasibility assurance is conducted to better evaluate technical options and their associated risks. Many of the technical risks in an SoS framework are due to incompatibilities between different system components or limitations of older system components with today’s technology.

As the SoS development teams begin to jell, the LSI focuses on incremental acquisition activities for the development and integration/test of the required system components for the SoS. During this process, there are often continuous changes and new risks that must be managed. In addition, the LSI is continuously looking for opportunities to simplify the SoS architecture and reduce effort, schedule, and risks. Key management issues for the LSI include:

- **Number of stakeholders** – The stakeholders in an SoS development effort are numerous. They come from sponsoring and funding organizations as well as the various user communities that have high expectations for the planned SoS.
- **Number of development organizations** – Because the system components are often “owned” by an organization other than the sponsoring or LSI organizations, there is often a separate development organization associated with each system component in the SoS. In addition, some of the system components can be systems of systems in their own right. This means that there may be lower level suppliers associated with each system component, adding to the number of development organizations.
- **Number of parallel, independent (or not so independent) developments** – The hope in an SoS development effort is that the overall SoS architecture can be defined well enough up front to allow for many concurrent development efforts in parallel. However, there can be compatibility and commonality issues between system components and what were initially thought to be independent developments become dependent developments, often affecting integration and

\textsuperscript{SM} CMMI is a service mark of Carnegie Mellon University.
test activities. As an example, a slip in schedule for one system component may mean that several other system components cannot perform their planned integration and test activities for a given increment/build.

- Number of decision “approvers” – Studies [1,11] have shown that as the number of people involved in the decision-making process increases, the probability of getting a timely (or even any decision) often decreases. In the SoS development arena, the stakeholders, the system component “owners”, as well as the LSI organizations are often all involved in making key decisions.

- Cross-cutting risks – These are risks that cut across organizational boundaries and/or system components (as opposed to system component risks that can be managed by the system component supplier).

Key to the success of this SoS development approach is the fact that the system components within an SoS are often “owned” by another organization. This means that SoS timelines are often controlled by other “outside” goals and timelines. SoS-enabling features are often incorporated into SoS components along with the other enhancements and features planned by the component “owner”. There may be long-lead enhancements that are not required by the SoS architecture/system, but are more important to the component owner or user organization and will delay implementation of the SoS features. Also, these other on-going changes (not required for the SoS) may impact the stability of the component (including its architecture). A current example of this is the limited resources and specialists available to develop new features needed to support today’s Iraq operations vs. features needed to support the Army’s Future Combat System (FCS) SoS in the future. While this may be perceived as more of a schedule issue, it can also impact effort since component delivery delays can result in inefficient integration activities and significant rework.

**Key LSI Activities in the CMMI<sup>SM</sup> Project Management Process Category:** The CMMI<sup>SM</sup> Project Management process areas key to LSI efforts are project planning, project monitoring and control, supplier agreement management, integrated project management, risk management, integrated teaming, and quantitative project management. As LSI organizations try to scale up their traditional system engineering processes, they find that there are often new and unexpected issues. Typical issues include:

- Traditional planning and scheduling may lead to unacceptably long schedules, requiring the LSI to be more creative in both their technical and implementation approaches.

- Planning and tracking activities must integrate inputs from a variety of different organizations, each with its own (and probably different) process.

- Traditional oversight and coordination can spread key LSI personnel too thin.

- More emphasis is required for contracting and supplier management. Incentives are often needed to better align priorities and focus of the system component supplier organizations. In addition, contracts must provide mechanisms to allow suppliers to participate more in the change management process to help assess impacts and to develop efficient approaches to proposed changes.
• Standardization of all processes may be overwhelming. The LSI needs to decide what to standardize and what to let the suppliers control.

• The decision making process involves considerably more organizations. As mentioned above, this can make the decision making process much more complex and time-consuming and it may have significant impacts on the overall schedule and effort.

• Risk management for cross-cutting risks needs to cross organizational boundaries. It is important that risk management activities for cross-cutting risks don’t select strategies that are optimal for one area of the SoS, but are to the detriment of other areas.

**Key LSI Activities in the CMMI<sup>SM</sup> Engineering Process Category:** The CMMI<sup>SM</sup> Engineering process areas key to LSI efforts are requirements development, requirements management, technical solution, product integration, verification, and validation. In the SoS LSI environment, there is a definite change in the focus of these engineering activities. For example,

• Requirements are primarily at the SoS level and only address the system components with respect to their integration into the SoS framework/architecture.

• The LSI needs to know when to system engineer and when not to system engineer. When SoS architecture conflicts or incompatibilities are identified, tradeoffs must be conducted to determine if it is better to accommodate the conflict or incompatibility at the SoS framework level or to require the supplier to make system component changes.

• SoS technical solution, product integration, verification, and validation focuses primarily on the communications between the system components.

• Other system component technical solutions, integration, verification, and validation activities are the responsibility of the system component “owners”.

• LSI may or may not be responsible for actual development of system components for the SoS. If the LSI is responsible for the development of one or more components, it may lead to perceived conflicts of interest and must be carefully managed. The goals at each level of development must be carefully balanced to ensure the overall SoS success.

**How Traditional Processes Are Adapting to the SoS Environment:** Since SoS development efforts usually span many years and include many incremental or evolutionary developments, there are opportunities for the LSI organizations to adapt and mature their processes to the LSI environment. The following are some observations on how these organizations are evolving their processes:

• Traditional planning and scheduling: LSIs try to plan activities as independent projects that can be performed concurrently. This requires that up-front SoS architecting be performed in sufficient detail to allow sub-projects to be somewhat independent of each other. It also requires that risk-driven processes be used to identify and manage risks early at SoS and sub-project levels. These
risk-driven processes often include Life Cycle Objectives (LCO) reviews, Life Cycle Architecture (LCA) reviews, and Feasibility Rationale (FR) studies.

- More agile processes: LSIs are attempting to blend traditional processes with more agile processes. They are more agile when dealing with risk, change, and opportunity management for future increments, but plan for stabilized evolutionary increments in the near term. Key to this approach is knowing when to plan, control, and stabilize and when to be more flexible, agile, and streamlined. The agile teams are responsible for performing acquisition intelligence, surveillance, and reconnaissance functions, and then rebaselining future increment solutions as necessary.

- Competing priorities: LSIs use stakeholders to negotiate priorities with other ongoing system component enhancements and maintenance.

- Project monitoring and control: LSIs attempt to manage key personnel resources in order to avoid burnout. One of the ways this is accomplished is to prioritize oversight areas.

- Integrated project management: LSIs must identify key cross-cutting processes for standardization and allow for flexibility in other areas. System component and supplier organizations are allowed to use their own proven processes—they have been selected for their technical expertise and ability to produce.

- Decision making process: LSIs need to reduce the number of required SoS-level decisions to the extent possible. Many organizations are involved at this level and decisions can be difficult to make because of competing priorities and the availability of the required decision makers. Decisions that are not timely can delay needed work or cause unnecessary rework. Studies have also shown that the probability of success of making any decision decreases as the number of required decision approvers increases [11].

- Risk management: Cross-cutting risks need to be managed and balanced across system and organizational boundaries. Each risk needs a responsible “owner”. Without “owners”, critical risks can move to the backburner and forgotten until it is almost too late. One approach is to create risk portfolios and “owners” for each portfolio to manage cross-cutting risks.

- Integrated product teams typically play a much larger role and have more responsibilities.

- The people processes are at least as important as the technical processes. Personal, organizational, and political motivations and priorities can impact the success of the project.

**Comparison to System Engineering Activities in EIA 632:** To better understand “how are LSI activities different that the traditional system engineering activities”, we have compared the list of LSI activities to those identified in EIA 632. Appendix A shows the results of this analysis. As one can see, the system engineering activities identified in EIA 632 are generally applicable to SoS LSI efforts. What is different is that these activities are primarily focused at the SoS architecture level.
The EIA 632 activities provide good coverage of the technical activities performed by the SoS LSI, but current observations indicate that they do not adequately reflect the organizational management and political negotiation activities that are a major source of effort for LSI organizations. Also, current observations indicate that the types of issues that can impact the overall effort are related more to the greater SoS complexity and the number of organizations, often with competing priorities, that are involved in the SoS development process. These issues are often not a significant factor in smaller system engineering projects. Further research is required in this area to better quantify the major sources of SoS LSI effort over the SoS life cycle as well as the range of impact of the more significant effort scale factors.

**What Can Be Learned From Organizational, Complex Adaptive Systems, and Chaos Theories**

One of the goals in the development of the COSOSIMO model is to allow users to look at the impact of various architecture decisions and development approaches on the overall LSI effort. As we have observed LSI organizations mature, we have seen how their management approach and focus seems to change. And as we note these changes, we are intrigued by the idea that there are many similarities with concepts from organizational, complexity, and chaos theories.

By investigating these theories, we hope to discern organizational characteristics that can help improve the chance of success of complex SoS development efforts and even improve the efficiency of the development effort. In this vein, research was conducted and the following concepts and related organizational characteristics analyzed as candidate factors for the COSOSIMO model.

Chaos theory explains and helps identify underlying order in apparently random data or events. Behaviors thought to be random may have natural boundaries or patterns. Once these boundaries or patterns are determined, the order becomes apparent. This order is sometimes referred to as “emergent order” [5,6,15].

So how does this apply to SoS development and LSI activities? According to Highsmith [5], complex systems are often nonlinear, leading us to look at chaos theory. These new, never been done before, complex systems require innovation and creativity, often leading to new technologies. Because there are so many unknowns, these tasks are difficult to predict and difficult to control.

One of many observations of interest is that we are now recognizing organizations as systems [15]. Highsmith [5] further states that it takes a complex adaptive system (organization) to develop a complex adaptive system (or SoS). He defines a complex adaptive system (CAS) as an ensemble of independent agents:

- Who interact to create an ecosystem
- Whose interaction is defined by the exchange of information
- Whose individual actions are based on some system of internal rules
- Who self-organize in nonlinear ways to produce emergent results
- Who exhibit characteristics of both order and chaos (the edge of chaos)
- Who evolve over time.
To encourage innovation and creativity requires the team to be mission or vision driven; have both technical and organizational flexibility, along with the ability to re-organize (self-organization) as new information becomes available; and work in a learning environment with a collaborative management style [5,15]. Sheard [12] further states that complex systems with learning capabilities can adapt until “they are poised on the edge of chaos”. It is at this point, the sweet spot between too much control and too little control, that they have “maximum fitness relative to their environment”, allowing them to take in information from the outside world and quickly change themselves to react to it, and in the process, “increasing their own fitness faster than competing systems”.

By creating a learning environment for the development team, teams have the flexibility to explore critical options, resulting in innovative “emergent” outcomes. Innovative learning focuses on “what we know”, “what do we need to know”, and “what do we not understand”. Speed and focus are important when pursuing innovative ideas, but it is also important to pause occasionally to allow the team to learn from increment to increment. Highsmith [5] recommends a “go fast, pause to reflect, then go fast again, while minimizing the risk of a major wreck” way. A spiral development process would also add a step to adjust plans after the reflection in order to better refocus efforts. This all ties in with Wheatley’s recommendation to “think globally, act locally”[15]. By this she means, that teams should focus more on the mission, be flexible in their approaches, and not to spend time on detailed, long-term planning—detailed, long term planning is often a waste of time in this type of environment since plans often change significantly as a result of on-going learning. This leads to Highsmith’s characteristics of adaptive development cycles [5]:

1. Mission driven
2. Component-based
3. Iterative
4. Time-boxed
5. Risk driven and change tolerant.

**SoS LSI Success Challenges**

Many of the challenges to the successful development of an SoS are not always technical in nature—they are more related to political stakeholder issues and management decisions, some of which eventually impact the technical solution. This section describes some of the key challenges and ways that are being used to mitigate them.

**Organizational and Management Impacts to Success:** As described in [11], there can be many issues and obstacles that can significantly affect the success of major government projects or programs. Reasons for lack of success on very large programs can include:

1. Changing leaders
2. New (unforeseen) organizations or players
3. Inflation
4. Different goals/focus between organizations/players
5. Changing priorities
6. Incompatible goals and desires
7. The ability or inability to shortcut.

There are not always solutions to some of these, but anticipating and managing them can decrease the impact to the system or program of interest.

**Candidate Ways to Improve Success Rates and Efficiency:** By viewing SoS LSI teams as complex adaptive systems/organizations and focusing on development approaches that support innovation and learning, one can see a match between CAS organizations and the processes that are evolving in current LSI organizations:

1. Plan for risk-driven spiral processes and organizations along with stabilized evolutionary builds (*mission driven, iterative, risk driven and change tolerant*).
2. Streamline SoS-level processes to take advantage of suppliers’ own processes—this will lead to fewer steps and fewer decisions (*allow suppliers to self-organize*).
3. Not too fast (beware of speed problems) – Need to do up-front architecting and engineering to get everyone moving in the right direction. Need to develop prototypes and conduct analyses up-front to minimize technical risks and potential performance problems (*flexibility to explore critical options*). If time is not taken to perform these activities adequately, it may cause excessive rework down the road.
4. Base program on performance, not promises. Tie key decisions to LCA simulations/models to reduce risk (*flexibility to explore critical options*).
5. Have the appropriate infrastructure in place (*support innovation and learning*). “Infrastructure” is the services and capabilities required to support development. In the case of a program such as FCS this could be labs, development processes, standards, right technical talents/staff. The right resources and the right people at the right time can have a big positive impact on the program.
6. New program must fit into arrangements that have been made with other purposes in mind, although this will increase the number of required clearance points. Need to be aware that system component “owners” may be evolving systems to meet potentially incompatible SoS requirements. (*mission driven, change tolerant*)

Sheard [12] also presents additional recommendations for moving system engineers to the edge of chaos in the areas of system architecting, design, and impact assessments to support the changing environment and requirements.

**Conclusions**

Back to the cost model, COSOSIMO: what does this mean with respect to trying to estimate the LSI effort to architect, manage the development, and field a functional SoS? LSI organizations are realizing that if more traditional processes are used to architect and integrate SoSs, it will take too long and too much effort to find optimal solutions and build them. Complexity theory and its implications for complex adaptive systems are presenting ways to potentially shorten schedule and effort and end up with a higher quality results. If these results are as significant as the literature indicates they might be,
they should be reflected in the COSOSIMO estimation model. By capturing the effects of these differences in organizational structure and system engineering processes in the COSOSIMO estimation model, management will have a tool that will better predict LSI effort and to conduct “what if” comparisons of different development strategies.

**Future Work**

The conclusions indicate that we are looking at some interesting relationships between SoS LSI effort, organizational management, political relationships, and complex adaptive system theories. However, little has been done to quantify the results achieved using different approaches in different organizational and political environments. Additional research is planned to collect data from SoS LSI organizations and developers and analyze it with respect management approaches, team organization and flexibility, the overall decision-making processes, and the evolution of processes and technologies from increment to increment. Additional investigations are also planned in the areas of value-based, risk driven development processes, activity theory, and engineering collaboration/negotiation theory. The results of these investigations will be used to evaluate the adequacy of the proposed COSOSIMO size drivers and scale factors and, for selected scale factors values, start determining factor weights.

**References**


Appendix A

Summary of EIA 632
System Engineering Processes [4]
<table>
<thead>
<tr>
<th>Process</th>
<th>EIA/ANSI 632 Task</th>
<th>Definition</th>
<th>SoS LSI Focus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acquisition and Supply</td>
<td>1. Product Supply</td>
<td>Assess acquisition request, offer or directive, negotiate agreement, deliver products</td>
<td>Similar to SE focus</td>
</tr>
<tr>
<td></td>
<td>2. Product Acquisition</td>
<td>Prepare acquisition requests, evaluate supplier response, make offer, negotiate agreement, accept delivered products</td>
<td>Similar to SE focus</td>
</tr>
<tr>
<td></td>
<td>3. Supplier Performance</td>
<td>Define supplier relationships, participate in product teams, monitor product metrics and assess products (invoke Reqs. 9-11 as applicable), flow down CONOPs requirement changes, control requirement changes, assess progress against requirements, validate products received (invoke Req. 33a)</td>
<td>Major activity for LSI</td>
</tr>
<tr>
<td></td>
<td>5. Technical Effort Definition</td>
<td>Identify project requirements, establish information database, define risk management strategy, define product and process metrics, establish trade/off cost goals, identify TPMs, identify applicable project tasks, identify methods and tools, establish technology insertion approaches</td>
<td>Major LSI responsibility</td>
</tr>
<tr>
<td></td>
<td>6. Schedule and Organization</td>
<td>Develop event-based and calendar-based schedules, Identify resource requirements, define staffing/discipline needs, define team and org. structure</td>
<td>Similar to SE focus at SoS level</td>
</tr>
<tr>
<td></td>
<td>7. Technical Plans</td>
<td>Develop Engineering Plan, Risk Plan, Technical Review Plan, Validation Plans, Verification Plans, Other Applicable Plans (e.g., Human Factors, Security Plans)</td>
<td>Similar to SE focus at the SoS level System component “owners” and suppliers have primary responsibility at the SoS component level</td>
</tr>
<tr>
<td>Process</td>
<td>EIA/ANSI 632 Task</td>
<td>Definition</td>
<td>SoS LSI Focus</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>8. Work Directives</td>
<td>Develop work packages and generate work authorizations</td>
<td>Similar to SE focus at the SoS level System component “owners” and suppliers have primary responsibility at the SoS component level</td>
<td></td>
</tr>
<tr>
<td>9. Progress Against Plans and Schedules</td>
<td>Identify events, tasks, and process metrics for monitoring, collect and analyze metrics data, compare process metrics against plans and schedules, implement required changes</td>
<td>Performed at the SoS level</td>
<td></td>
</tr>
<tr>
<td>10. Progress Against Requirements</td>
<td>Identify product metrics to be monitored, collect and analyze product metrics data, record rationale for decisions/assumptions, compare results against requirements, identification and implementation of required changes</td>
<td>Performed at the SoS level At the system component/supplier level, integrated with other on-going changes</td>
<td></td>
</tr>
<tr>
<td>11. Technical Reviews</td>
<td>Identify technical review objectives and requirements, determine progress against event-based plan, establish technical review board, agenda and speakers, prepare technical review package and presentation material, conduct technical review, close-out review</td>
<td>Performed at the SoS level Key reviews defined for suppliers, other supplier reviews managed at the supplier level</td>
<td></td>
</tr>
<tr>
<td>12. Outcomes Management</td>
<td>Capture process outcomes, perform configuration management, perform change management, perform interface management, perform risk management, perform data and document management, manage information database, manage and track requirements</td>
<td>Performed at the SoS level</td>
<td></td>
</tr>
<tr>
<td>13. Information Dissemination</td>
<td>Provide progress status, provide planning information, disseminate approved and controlled requirements, provide formation for and from reviews, make available design data and schema, make available lessons learned, report variances, disseminate data deliverables, disseminate approved changes, disseminate directives</td>
<td>Performed at the SoS level</td>
<td></td>
</tr>
<tr>
<td>System Design</td>
<td>14. Acquirer Requirements</td>
<td>Identify, collect, and prioritize acquirer's system requirements, ensure completeness and consistency of the set of collected acquirer requirements (Invoke req. 26), record set of acquirer requirements</td>
<td>Performed at the SoS level Responsibility of the supplier to integrate with other on-going changes</td>
</tr>
<tr>
<td>Process</td>
<td>EIA/ANSI 632 Task</td>
<td>Definition</td>
<td>SoS LSI Focus</td>
</tr>
<tr>
<td>---------</td>
<td>------------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>15. Other Stakeholder Requirements</td>
<td>Identify and collect other stakeholders' end product requirements, identify and collect other stakeholders' enabling product requirements, identify and collect other stakeholders' external constraints, ensure completeness and consistency of the set of other stakeholders' requirements (invoke req. 27), record set of other stakeholder requirements.</td>
<td>Performed at the SoS level Responsibility of the supplier to integrate with other requirements from other system component stakeholders</td>
<td></td>
</tr>
<tr>
<td>16. System Technical Requirements</td>
<td>Establish required transformation rules, priorities, inputs, outputs, states, modes, and configurations, define operational requirements, define performance requirements, analyze acquirer and other stakeholder requirements (e.g. human factor effects, capacities and timing, technology constraints, product design constraints), challenge questionable requirements, resolve identified conflict of requirements, prepare a set of acceptable system technical requirement statements, ensure completeness and consistency of the set of system technical requirements (invoke req. 28), reset the set of system technical requirements</td>
<td>Key activity at the SoS level Responsibility of the supplier to integrate with other system component requirements</td>
<td></td>
</tr>
<tr>
<td>17. Logical Solution Representations</td>
<td>Select and implement one or more these four approaches (Functional Analysis, Object Oriented Analysis, Structured Analysis, Information Modeling), or another approach designated by enterprise policies, guides, or standards; Establish a set of logical solution representations (see list); Assign system technical requirements — including performance requirements and constraints; Identify, define, and validate derived technical requirement statements (invoke Req. 25); Ensure completeness and consistency of the logical solution representations (Invoke Req. 29); Record logical solution representations and derived technical requirements</td>
<td>LSI responsible at the SoS framework level System component suppliers responsible at the system component level</td>
<td></td>
</tr>
<tr>
<td>18. Physical Solution Representations</td>
<td>Analyze logical solution representation sets, assigned system and derived technical requirements: Assign representations, derived technical requirements and unassigned system technical requirements to appropriate physical entities (see list)</td>
<td>LSI responsible at the SoS framework level System component suppliers responsible at the system component level</td>
<td></td>
</tr>
<tr>
<td>Process</td>
<td>EIA/ANSI 632 Task</td>
<td>Definition</td>
<td>SoS LSI Focus</td>
</tr>
<tr>
<td>---------</td>
<td>------------------</td>
<td>------------</td>
<td>---------------</td>
</tr>
<tr>
<td>19. Specified Requirements</td>
<td>Fully characterize design solution, Ensure design solution consistency (Invoke Req. 30), Specify requirements, Record design solution and related specified requirements, Establish projects for development of enabling products</td>
<td>LSI responsible at the SoS framework level System component suppliers responsible at the system component level</td>
<td></td>
</tr>
<tr>
<td>Product Realization</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20. Implementation</td>
<td>Acquire Products (Goods or Services), Validate acquired products (Invoke Req. 33), assemble/integrate validated end products, Verify integrated end products (Invoke Req. 31), Verify enabling products for each associated process (Invoke Req. 32), Validate the verified end product (Invoke Req 33b)</td>
<td>LSI responsible at the SoS framework level System component suppliers responsible at the system component level</td>
<td></td>
</tr>
<tr>
<td>21. Transition to Use</td>
<td>Acquire and put in place enabling products, Prepare end products for shipping or storage, Prepare the operational sites, Installation of products, Perform commissioning, provide ghosting, train users and maintenance personnel, provide in-service support</td>
<td>LSI responsible at the SoS framework level System component suppliers responsible at the system component level</td>
<td></td>
</tr>
<tr>
<td>Technical Evaluation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22. Effectiveness Analysis</td>
<td>Plan effectiveness analyses, Analyze system cost effectiveness, analyze total ownership cost, analyze environmental impacts, analyze system effectiveness, record outcomes of effectiveness analysis</td>
<td>Possible LSI responsibility</td>
<td></td>
</tr>
<tr>
<td>23. Tradeoff Analysis</td>
<td>Plan tradeoff analysis, perform tradeoff analysis, record outcomes of tradeoff analysis</td>
<td>Major LSI activity in development of SoS architecture and in system component/ supplier selection</td>
<td></td>
</tr>
<tr>
<td>24. Risk Analysis</td>
<td>Identify risks, characterize risks, prioritize risks, evaluate ways to avert risks, define and implement a plan or approach for averting each significant risk, capture and communicate risk analysis outcomes</td>
<td>Major LSI activity at SoS level</td>
<td></td>
</tr>
<tr>
<td>25. Requirements Statements Validation</td>
<td>Invoked by Req. No. 17, Analyze and ensure each technical requirement statement with (list of criteria), Analyze and ensure each technical requirement statements in pairs and as a set are stated with (list of criteria)</td>
<td>Similar to SE focus at SoS level</td>
<td></td>
</tr>
<tr>
<td>26. Acquirer Requirements Validation</td>
<td>Invoked by Req. No. 14, Select methods and define procedures, Establish downward traceability, Establish upward traceability, Identify and resolve variances, Record validation results</td>
<td>Similar to SE focus at SoS level</td>
<td></td>
</tr>
<tr>
<td>Process</td>
<td>EIA/ANSI 632 Task</td>
<td>Definition</td>
<td>SoS LSI Focus</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>27. Other Stakeholder Requirements Validation</td>
<td>Invoked by Req. No. 15, Select methods and define procedures, Establish downward traceability, Establish upward traceability, Identify and resolve variances, Record validation results</td>
<td>Similar to SE focus at SoS level</td>
<td>--------------</td>
</tr>
<tr>
<td>28. System Technical Requirements Validation</td>
<td>Invoked by Req. No. 16, Select methods and define procedures, Establish downward traceability, Establish upward traceability, Analyze assumptions, Analyze other system technical requirements, Identify and resolve variances, Perform Revalidation, Record validation results</td>
<td>Similar to SE focus at SoS level</td>
<td>--------------</td>
</tr>
<tr>
<td>29. Logical Solution Representations Validation</td>
<td>Invoked by Req. No. 17, Select methods and define procedures, Establish downward traceability, Establish upward traceability, Analyze assumptions, Identify and resolve variances, Perform Revalidation, Record validation results</td>
<td>Similar to SE focus at SoS level</td>
<td>--------------</td>
</tr>
<tr>
<td>30. Design Solution Verification</td>
<td>Invoked by Req. No. 19, Plan the design solution verification in accordance with the Verification Plan, the agreement, and the applicable enterprise-based life cycle phase, and level in the system structure, Perform the planned design solution verification using selected methods and procedures within the established verification environment, Perform reverification, Record verification results</td>
<td>Similar to SE focus at SoS level</td>
<td>--------------</td>
</tr>
<tr>
<td>31. End Product Verification</td>
<td>Invoked by Req. No. 20, Plan the end product verification in accordance with the Verification Plan, the agreement, and the applicable enterprise-based life cycle phase, and level in the system structure, Perform the planned end product verification using selected methods and procedures within the established verification environment, Perform reverification, Record verification results</td>
<td>Similar to SE focus at SoS level</td>
<td>--------------</td>
</tr>
<tr>
<td>32. Enabling Product Readiness</td>
<td>Invoked by Req. No. 20, Plan enabling product readiness determination in accordance with the agreement, and the applicable enterprise-based life cycle phase, and level in the system structure, Perform planned enabling product readiness determination using selected methods and procedures, Reaccomplish readiness determination, Record readiness determination results</td>
<td>Similar to SE focus at SoS level</td>
<td>--------------</td>
</tr>
<tr>
<td>Process</td>
<td>EIA/ANSI 632 Task</td>
<td>Definition</td>
<td>SoS LSI Focus</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>33. End Products Validation</td>
<td></td>
<td>Invoked by Req. 3. Confirmation by examination and provision of objective evidence that the specific intended use of an end product, or an aggregation of end products, is accomplished in an intended usage environment; Representative tasks include: Determine validation exit criteria, Acquire appropriate test article, Conduct validation, Perform revalidation, Record validation results.</td>
<td>Similar to SE focus at SoS level</td>
</tr>
</tbody>
</table>