A System-of-Systems Engineering GEOSS: Architectural Approach
Marion L. Butterfield, Jay S. Pearlman, and Stephen C. Vickroy

Abstract—There is an increasing need to perform systems-of-systems engineering (SoSE) in a global environment. A new SoSE Process has been developed which is a significant breakthrough in the development of large complex systems and net-centric systems-of-systems (SoS). The SoSE Process provides a complete, detailed, and systematic development approach for military and civil SoS. This architecture-centric, model-based Systems Engineering Process emphasizes concurrent development of the system architecture model and system specifications. It is applicable to all phases of a system’s lifecycle. The significant benefits of developing a system architecture model for GEOSS using the SoSE Process are described. An example of how the process would capture the architecture model of GEOSS is presented.

I. INTRODUCTION

A N ARCHITECT translates problem domain concepts of the customer into the solution domain concepts of the developer and user. For the system to be built as envisioned, the architect develops a system architecture model to communicate the vision and track construction against it. Architecture models are constructed to relate system functionality to other architectural attributes such as reliability, manufacturability, interoperability, or virtually any other quality attribute one requires. These attributes are modeled so we can have reliability, manufacturing, logistics supportability models, and level of systems interoperability capabilities model, etc.

Architecture models are abstractions of the unrealized system. Architecture model contents are dictated by the purpose of the model. Some models are for use in analysis while others are used only for system visualization. The results of these models are documented in a quality attribute tree whose results are included in the architecture. These models also provide a common system picture for the development team as shown in Fig. 1.

The application of system models, and for this paper, the system-of-systems (SoS) models comes with a number of constraints and challenges. With the evolution of technology and globalization, the machinery of mankind has become more complex. The development of these complex systems to meet a customer’s goals has also become more complex. To manage this new developmental complexity, engineers must have available more detailed and comprehensive systems engineering processes and tools. The use of these new processes and tools will enable engineers to more efficiently and effectively develop a balanced system of system’s solution.

Since the SoS is a construct of both legacy and new systems, the analysis will look at the systems definitions first. A recognized definition for the concept of systems is that a system is “an integrated composite of people, products, and processes that provide a capability to satisfy a stated need or objective” [1].

Several phrases in this definition are particularly important to the context of this discussion. First, a system is a composite of several pieces, which may include one or more components, comprised of people, products, and processes. Each of the components in a system brings its own capabilities and limitations, which include specific interfaces that exist between the individual components within the system as well as the components and the environment in which it operates.

Tools have been developed to assist in defining and optimizing the requirements and interfaces in which components are built and used, to satisfy some stated need or objective. Many additional considerations are introduced with the concept of joint operations of independent systems or SoS.

Consider a specific example for illustration of a system verses an SoS. An example of a large complex system is a commercial airplane, where an integrator is responsible for bringing together individual systems, such as galleys, lavatories, controls and displays, and engines, to develop a transportation machine that will safely and effectively serve its primary function for 25 or more years. None of these subsystems acts independent of one or more of the other subsystems in creating its services. An example of an SoS is our national air traffic management system. The very complex aircraft, while it can provide transportation in its own right, are parts of a SoS which is the national and international traffic control. In fact there are many systems within the airspace SoS—those for communications, weather observation and dissemination, etc—which operate interactively to produce an effect transportation capability.

II. WHAT IS A SYSTEM-OF-SYSTEMS?

The difference between an SoS and a large scale integrated system should be examined as a first step to addressing SoS implementations. Features of an SoS are that the component systems can operate independently to produce products or services satisfying their customer objectives. The SoS may connect systems through interoperability arrangements that do not require tight coupling or strong integrations. This, in turn, leads to the attribute that an SoS maintains its inherent operational character even as system components join or disengage from the SoS. The Global Earth Observation System of Systems (GEOSS) is a good example of a complex SoS.
It is comprised of operational and research systems that are created for their own purposes, usually to satisfy national or regional requirements. The prioritization of these systems remains with the originator, as well as the resource allocations for operations.

Under this environment, it is imperative that GEOSS draw upon the benefits of an SoSE Process similar to that in Fig. 2 to maximize its impacts. Architecture models minimize the risk that the system stakeholders are not aware of significant elements of the architecture. Each architecture model is intended system architecture definition at all levels, the associated specification definition, and the needed design comprehension and realization.

As a foundation for investigating options for the architecture of GEOSS, based on the previous discussion, a definition of SoS is: “A System-of-Systems (SoS) is a “super-system” comprised of elements that are themselves complex, independent systems which interact to achieve a common goal” [1]. In the context of this definition, attributes of an SoS are that the component systems achieve well substantiated purposes in their own right even if detached from the overall system. In fact, the components systems are managed in large part for their own purposes rather than the purposes of the whole. To then justify the creation of an SoS, the SoS must exhibit behavior, including emergent behavior, not achievable by the component systems acting independently, offering significantly new capabilities that justify the “overhead” associated with the SoS. Since the SoS does may not have absolute control over the component systems, the SoS must be a dynamic and open environment so that component systems, functions, and
behaviors may be added or removed during its use with significant impact to the SoS framework.

Over the last decade, advances in information technology have made possible processes that enhance the ability of individual hardware platforms or systems to interoperate with one another. Consider networked computers and the internet, where various diverse computing systems interact for data creation, processing, and display. Computer languages have been developed to facilitate their interoperability using techniques such as software and the internet, the potential exists to seamlessly share sensor and display information, coordinate information flow, or provide subsystem-like functionality as part of a greater "SoS." Clearly, as information technology has extended operational capability to include the utilization of disparate resources by sharing remotely located components, the need to understand how previously disparate, independent systems interoperate has become even more critical.

The SoSE process described in this paper uses a disciplined, enterprise wide effort to integrate systems, operations, and information-based tools and processes. Fig. 2 outlines how the integration of disparate technologies provides a unique product that creates system synergies across complementary interfaces.

III. WHY IS THIS IMPORTANT?

Complex SoS include characteristics that are not typical of heretofore assembled system designs. For example, systems that become components of SoS are typically designed without knowledge of other component members. The SoS legacy components were designed using the expectations and limitations that provide a capability to satisfy a stated need or objective, but without knowledge of the capabilities and limitations of other SoS components. To have the SoS goals realizable without putting unreasonable burdens on the systems from which it is formed, the systems engineering goal is to determine what few things must be the same so that everything else can be different.

The SoSE process must therefore stress increased awareness of interoperability and risk management because of the dispersed nature of SoS components and the constraints imposed by the legacy features of these components. At the SoS architecture level, it is necessary to focus on ensuring that components behave in a coordinated manner across system boundaries and thereby results in the desired operational capabilities at the SoS level.

When addressing a system with top down requirements it control by the system manager, the architect is able to prescribe component system requirements. This architecture is used to provide traceability between the goals of stakeholders by defining functional requirements and system objects in each cycle of systems development. In a SoS, the architect focuses on interoperability and system interfaces with the goal of balancing performance and risk. The relationship between the system architecture and reducing program risk is shown in Fig. 3.

IV. COMPONENTS OF THE SoSE PROCESS

The SoSE architect must manage a complex mix of system capabilities and interfaces with customer goals and constraints. An effective system architect must understand the relationships of all system components from the operational SoS level to the component level where the basic functions are performed by people, products, and processes. Requirements are assembled from multiple program stakeholders and integrated based on operational strategies and industry standards.
By applying the SoSE process, a Unified Modeling Language (UML) architecture model is developed, leading to requirements for analysis, modeling, and simulation. Aligning system and operational responsibilities assists in the identification of gaps in functionality and technology. Using this approach, an architectural model (description) that guides system design decisions, identifies the required SoS operational framework within which the system level specifications can be incorporated. The flow of data and functionality is shown in Fig. 4.

While the diagram may appear to be a linear progression from requirements to architecture descriptions and specifications, a substantial number of iterations or internal loops exist between the SoS architecture model and analysis, modeling, and simulation. Analysis and modeling may be used to effectively test the formulated Architecture and concepts. Only when there is a satisfactory model for the intended application are the system characteristics extracted to form system specifications and capability description.

V. SoSE REQUIREMENTS EVOLUTION

Mission objectives are the genesis of SoS architectures. These mission objectives are defined by operational stakeholders and provide the top-level requirements and constraints for the SoS. The objectives are abstracted into a requirements baseline using a variety of tools such as context views, use cases, activity diagrams, and state diagrams during SoS requirements analysis. System-level functionality has not been assigned at this level; this initial optimization and gap analyses is performed at the system of system level.

Similarly, designs are synthesized into a physical architecture from the functional architecture via development of physical view models and a behavioral model. From this high level model, specific system and component functional alignments can occur.

The process is an iterative one at each level, assuring that requirements are consistently and validly matched between the SoS level interoperability specifications and available component systems, to best address operational functionality for the stakeholder needs. Here again, we see a significant difference between a top down requirements to design flow of a system design and the synthesis of an SoS from legacy/existing components. Fig. 5 shows how operational functionality is incorporated into all levels of the SoS architecture and helps detail portions of the overall SoSE Process found in Fig. 4. The stakeholder requirements/constraints at the top of Fig. 5 are equivalent to the three input items at the top of Fig. 4. The three systems engineering subprocess tasks of requirements analysis, functional analysis, and design synthesis describe and comprise the activities found in the SoSE Process activity shown in Fig. 4. The accompanying products shown in Fig. 5 for each of the subprocesses will constitute the level-by-level design and that is equivalent to SoS/System Architecture Model shown in Fig. 4.

VI. SoS ENGINEERING APPROACH FOR DEVELOPMENT

An enterprise process for developing SoS architectures for the purpose of large scale systems development and implementation employs a tiered approach that starts by identifying or defining the stakeholder mission objectives that define the overall system purpose. For GEOSS, the high level objectives are to provide timely, quality, long-term, global information as a basis for sound decision making. Attributes were identified at the first GEO Summit [2] for the SoS as follows:

1) improved coordination of strategies and systems for observations of the Earth with a view to moving toward a comprehensive, coordinated, and sustained Earth observation system or systems;
2) coordinated effort to involve and assist developing countries in improving and sustaining their contributions to observing systems, as well as their access to and effective utilization of observations, data and products, and the related technologies;
3) exchange of observations recorded from *in situ*, aircraft, and satellite networks, in a full and open manner with minimum time delay and minimum cost.

To address the goals in terms of requirements, they should be restated so that only one goal is included in a statement and so that there is a minimum of overlap between statements. The three goals can be restated, for example, as follows:

1) achieve a comprehensive, coordinated, and sustained Earth observation SoS;
2) assist developing countries in improving and sustaining their contributions;
3) achieve universal and effective utilization of observations, data and products, and the related technologies;
4) full and open exchange of data with minimum time delay and minimum cost.

The goals are traceable as follows. New goal 1 is a restatement of Old goal 1. New goals 2 and 3 are a restatement of Old goal 2. New goal 4 is a restatement of Old goal 3. It is noted that New goal 4 could be considered a way of measuring New goal 3, which leaves three distinct goals to be satisfied by the SoS architecture.

For application of the SoSE, the previous objectives can be addressed from the perspective of a hierarchical framework. A flow of the SoSE Hierarchical Framework is shown in terms of the SoS development levels in Fig. 6. This Framework was chosen to provide the system developers a clear picture of both the problem and potential solution and is dependent on the requirements and objectives being addressed. The Framework begins at the SoS Goals level. The Goals level is the SoS enterprise level where the high level objectives are enumerated. From these mission objectives, defined and derived measurable SoS operational capabilities create the basis for providing an architecture that achieves mission goals. These are shown at the next level.

The next development level is the SoS Services (Operational Capabilities) level. The SoS Services are all of those operational capabilities required to achieve the collective SoS Goals. The operational capabilities are associated with a single SoS or a class of SoSs. Therefore, at the SoS operational capabilities level there is no suggested physical structure.

An example of how the GEOSS goals at this level might be restated in terms of the SoS’ Operational Capabilities follows.

To achieve a comprehensive, coordinated, and sustained Earth observation SoS, GEOSS will:
1) facilitate coordination;
2) provide for sustained operations;
3) provide a Full breadth of information;
4) possess the ability to evolve;
5) perform in a dynamic, open environment.

To assist developing countries in improving and sustaining their contributions, GEOSS will:
1) provide expertise outreach;
2) create and maintain communications infrastructure;
3) assure technology and data availability to developing countries.

To achieve universal and effective utilization of observations, data and products, and the related technologies, GEOSS will:

![Fig. 6. SoSE hierarchical framework [9].](image)
1) define and facilitate interoperability;
2) enable full and open exchange of data and products;
3) provide access to data with minimum time delay and minimum cost;
4) translate data to useful information products.

Operational capabilities provide a clearer notion of SoS functionality, based on clearly defined goals statements. Each operational capability is stated as an action statement, and will provide the basis for system level requirements that are necessary to be met to provide overall SoS functionality. Note that New goal 4 has now been restated as operational capability 3 b.

The Node/System Services level is the next level and the first level to suggest a structure by asserting an initial node structural concept and subsequently the associated SoS structure. The proposed SoS nodes are expressed as logical objects that may or may not be physically realizable. Operation nodes in a GEOSS may include individuals systems, such as tsunami warning systems, sensing devices such as weather satellites or coastal buoys, or infrastructure such as a hurricane data integration complex or emergency response centers. Each contributes services, data, or an integrated view of the earth to users, and can be subject to overall system integration.

The nodes may be mapped one-to-one or one-to-many systems. All of the systems at this level are considered to be physically realizable logical objects.

Physically realizable logical objects are systems or classes of systems that will have a physical implementation. Maintaining the systems as physically realizable logical objects helps the developers maintain the correct level of abstraction and minimizes the implementation of unnecessary physical constraints prior to when they are needed. SoS operational capabilities are now parsed to the proposed node/system structure. SoS operational capabilities have now been allocated to physically realizable logical objects that operate jointly to achieve the designated SoS Goals. This part of the process allows the attributes and services of the legacy system to be aligned with SoS objectives and the identification of gaps where additional capabilities are needed.

At the node/system services level, SoS operational capabilities are first assigned to operational nodes and then to the individual systems that comprise and SoS components. For GEOSS, as for most complex SoS, capabilities can become quite complex and numerous. Consequently, system level assignment will not be documented for this example. There are three large categories of system level assignments, corresponding to the three types of nodes expected to be part of the SoS, including the following:

- SoS Services contributor nodes (hard legacy);
- SoS Services contributor nodes (soft legacy or new);
- SoS Services GEO nodes (modifiable).

In the first case, nodes are currently designed and in use, with no further plans for modification. These systems are therefore defined as a constraint to the GEOSS, providing specific operational capabilities. The second category, soft legacy or new nodes, is considered similar to the first for the initial design spiral, with capabilities considered design constraints. However, plans for modification, providing improved capability, can be factored into subsequent design spirals. The final category, GEO nodes, provides for standard services and capabilities across all nodes, to be referenced functionally or operationally throughout the design process.

The next level, system behavior, is the first level where functional capabilities of a physical object or class of objects, not operational characteristics, are considered. Those operational capabilities assigned to each system at the node/system level are transformed into functional capabilities that a system must possess. Again, the current operational capabilities of the hard legacy systems constitute an architecture constraint. System behaviors are used to describe the system functional capabilities required to achieve the desired SoS performance.

At this level, the difference between different types of SoS is apparent. Attributes of a SoS from a single controlling source can include the following:

- uniformity of objectives;
- single “corporate” high level directions;
- cultural uniformity;
- common technology;
- data standards and quality established;
- process for encouraging or enforcing participation.

GEOSS contributing systems are made available by participating countries with the desire to benefit from SoS synergies. An example of this synergy is coordination of satellites from multiple countries to form a “virtual” constellation, which can provide similar or complementary data with frequent revisit times. An SoS with multiple controlling sources, can include the following attributes:

- disparate motivations;
- competing agendas;
- diverse backgrounds;
- varying technology levels;
- multiple views of data standards and quality.

GEOSS will necessarily be a collaborative effort, with contributions from organizations and countries whose requirements and needs are vastly different, and whose budgets may not support integration of services that are required for SoS interoperability. Systems are put in place to meet local or regional needs, whether meant to monitor river or estuary water height, ground saturation level, or highway traffic conditions. The technologies necessary to implement these nodes as independent systems is well established. In many locations, these systems are currently in place. For an SoS to assist in hurricane evacuations, the combination of these disparate, competing, and diverse technologies and data therefore need their objectives uniformly aligned with a single purpose of moving traffic smoothly through the available thoroughfares in a safe and orderly fashion. Local authorities responsible for public safety are therefore an SoS stakeholder and should be responsible for providing and implementing the integration requirements and resources. This is one example of how legacy systems can be modified to meet SoS needs.

The subsystem behavior level is the next level and the first level where we are not dealing with any operational capabilities considerations and only dealing with derived functional capabil-
ities. Again, at this level, we are only concerned with physical behavior expressed as functional capabilities. The functional capabilities assigned to the system are now decomposed, grouped by functionality, and allocated to proposed subsystem concepts.

The last level shown in Fig. 6 is repeated until the component level is reached and the remainder of the task is design oriented.

Each level of the process provides an essential portion of the complete description of the SoS. The views developed will provide the needed operational, informational, and systems descriptions of the SoS at all levels of use. Each level of the SoSE process requires a complete and, many times, iterative cycles of architecture definition to ensure systems sufficient coordination to optimize the SoS performance during the development process. Every iteration results in the traceability of objectives and operational functionality at all levels of the system architecture.

The complete decomposition defines and validates the details of the interoperability of the systems and components that comprise the SoS. When subsequent iterations occur during further SoSE development, consistent decomposition ensures that derived and other lower level attributes are fully qualified throughout the definition process. Multiple iterations of the decomposition process allow customers and other system stakeholders to have all of the required inputs for system operational completeness.

In the hurricane evacuation example, for the systems to work as a single entity, several conditions have to be met to create a public safety SoS. River level data comes in many forms, ranging from visual observation of gauges to automated reports of height and flow levels transmitted to a data center. The range of ground saturation data types can be similarly varied. Timely reporting for the purpose of evacuation requires that all necessary data be integrated with current terrain maps to predict expected flooding levels. Certainly this does not require all river, estuary, and ground saturation sensors to be integrated. However, subsystems required to provide the data necessary to make accurate predictions must be iteratively analyzed to simulate SoS performance and ensure proper traffic flows prior to evacuation needs.

As a global resource, GEOSS data will provide an improved coordination of strategic resources to monitor the health of the planet. A coordinated effort will assist developing countries to improve and sustain contributions to the observing system, and thereby increase their effective utilization of observations and related technologies. Exchanging observed data recorded in situ, aircraft, and satellite networks in an open manner with a minimum time delay and minimum cost.

There are many types of earth observing systems in place today, at all levels of technology. Well known are weather satellites that provide views of storms and cloud locations and movements. Weather data stations, installed at airports throughout the world, provide data to assist pilots in developing flight-plans. In the oceans, instrumented buoys monitor and report data for mariners on their voyages. These modern observation systems were designed with specific operational objectives, many of which require technology to communicate specific data to serve their specific purposes.

Other observations systems are designed to monitor river stage height and flow rate, ground temperature and water saturation, seismic activity, ice thickness, air component analysis, and cloud cover density. Monitors also measure non-weather related data, such as highway traffic flow and density, the price of petrol, electric grid conditions, and internet transmission rates.

It is important to consider disparate data types because all of them, in some way, contribute to the ability of Earth Observing Systems to perform their function. However, it is important to note that these systems were generally designed independently of one another, with specific operational objectives. It is unlikely that they will interoperate without an overarching SoS.

As the sophistication of data monitoring and collection capabilities has increased, it has become evident that selected combinations of data will provide key insights into monitoring the health of the planet. For example, monitoring the oxygen level in bodies of water may allow interpretation of causes for declines in the capacity of the region to sustain life. In the extreme cases, dead zones have been observed.

Over time, these dead zones may move, impacting other areas. These systems, initially designed for other purposes, can therefore impact the lives of many people by identifying problem source causes.

An example of how GEOSS may be described using the SoSE hierarchical framework levels is shown in Fig. 7. While multiple items would normally be identified and described at each level of an architecture model, only a single example is shown here for clarity.

This example mission objective is expressed as an operational capability when we say, “provide timely, high-quality, long-term global information as a basis for sound national or regional decision making” [3]. In other words, we are not only saying that GEOSS is intended to provide information to its stakeholders, but to do it in a manner that is readily useful for national or regional decision making.

An example of how a selected GEOSS goal would be decomposed into those SoS-level services (operational capabilities) required to meet this goal is shown for the SoS services level. Timely dissemination of information to reduce the loss of life and property from natural and human-induced disasters was chosen as an example of one of the desired SoS-level operational capabilities for GEOSS. At this level, it is expected that the GEOSS may have from eight to ten SoS-level operational capabilities. These services are logical objects comprised of functionality without any physically realizable structure imposed and are assigned, in this case, to the single entity; GEOSS.

An example of how these GEOSS operational capabilities are decomposed and then allocated to operation nodes that will be subsequently mapped into physically realizable logical objects is shown at the system services level. These activities establish the GEOSS node/system level.

In the case of GEOSS, the nodes will consist of functionality clusters that will each provide a service that when used in the joint GEOSS activities will result in the GEOSS stakeholders meeting their collective goal. The nodes are initially logical ob-
Fig. 7. Example GEOSS hierarchical framework [9].

Subjective that may or may not map into physically realizable logical objects. The resulting structural concept of physically realizable logical objects is the initial structural elements of the SoS architecture model. From these logical objects that are intended to be physically realizable will emerge the initial physical objects that will be subject to the constraints of technology and design. Timely, coordinated dissemination of observation information related to tsunamis is a single example of a set of nodes that would constitute the GEOSS.

Operational capabilities allocated to the physically realizable logical objects identified at the node/system services level will be translated into system behavior (functional capabilities) for each system at the system behavior level. The example shown in Fig. 7 is monitor ocean conditions and disseminate information (e.g., wave height, speed, direction). In other words, the timely dissemination of ocean conditions by those systems monitoring ocean conditions is required to meet the information dissemination goals at the SoS level. This is only one of the many examples of functional behavior assigned to the large number of GEOSS systems around the world.

Each of the many systems participating in GEOSS will be comprised of subsystems that must be identified and/or developed. The many functional capabilities of system must be decomposed and allocated to the system’s subsystems at the behavior level. The GEOSS subsystem functional capability example chosen to show in Fig. 7 is measure ocean height changes. The subsystem behavior level process is repeated for each subsystem down to the component level.

Each of these development levels result in a GEOSS architecture model that provides a clear view of how the GEOSS works from the SoS Goals level down to the lowest subsystem level. The resulting functional architecture provides a clear, unambiguous trace of the functional capabilities of the systems and subsystems up to the originating operation capability requirements.

VII. PERSPECTIVES ON A GEOSS

The three operational perspectives for GEOSS are defined by the interoperability of various systems and components based on the perceived need of the resulting system. The perspectives belong to communities whose interests vary, ranging from scientific to regional economic impacts, to sociopolitical need. Although the perspective on the need for such a system varies, the resulting outcome provides ready access to data and improves the quality of life for all concerned.

In evolving the SoS development, there are a number of different approaches with the final selection guided by the constraints at the customer and political levels. A first perspective is a large scale system that integrates various independent observation, modeling, and data management systems. Using this perspective, local control over observation systems continues to allow resource utilization to produce the data and information necessary to serve local needs efficiently and effectively by keeping decisions local. As long as the interfaces for integrating data transmission are maintained, and key parameters are reported for the collective good, the SoS is maintained so that integrated observation and modeling can proceed.

A second perspective is that the GEOSS is a system for converging worldwide monitoring systems. Under this perspective, the objective is to post all available data that is gathered to a centrally available resource like the internet, and permit access to use the data for the respective user’s purpose. The interoperability of these systems and the cross-purpose utilization of the data are dependent on the manners in which data are made available to end users. Data ownership rights and intellectual
property claims provide some of the motivation for providing data from observing systems. Conversely, liability as a result of data misapplication, timeliness, sensor error, or unavailability could restrict dispersal. Methods for addressing these concerns are important to this perspective.

The third perspective involves meeting social and political needs by using global observing data. As previously mentioned, multi-use of specific data, as in the water oxygen example, could provide social benefits in regions affected by those impacts. Disparate data areas such as land observation, ecosystem, biodiversity, weather, water, climate, energy, health, and disasters, can be made available to local, national and regional authorities to address the concerns of their respective constituencies. This information can drive local policy, planning, and project implementation that impacts the countries or regions industry, welfare, and general quality of life.

Recent research has shown that combining selected data can improve the accuracy of predicting storm path and intensity. Strategic and tactical uses of measurement systems have been shown to assist in preparing for these storms. Future goals include identifying gaps in knowledge and prediction capabilities that will enable more precise storm warnings and an integrated response.

VIII. MEASURES OF MERIT (MoM)

MoMs have been established for each of the system development levels, see Fig. 8. The MoMs at each level provide a measure by which the SoS developers can determine whether they meet the stated objectives at that level. For example, at the SoS level, the measures of policy effectiveness are quantities that measure how well the policy objectives that drive the SoS level mission are being met. These would be established by the GEO Plenary in conjunction with the National Authorities and the system operators. For GEOSS, the policy objectives for the SoS are to monitor continuously the state of the Earth, to increase understanding of dynamic Earth processes, to enhance prediction of the Earth system, and to further implement our international environmental treaty obligations. This flows down to the mission objectives, which includes the need for “timely, quality, long-term, global information as a basis for sound decision making” [3]. The value of such data for decision making introduces additional requirements that data from the “GEOSS,” should be comprehensive coordinated and sustained.

At the system services level, the National Program Managers that are participating in GEOSS establish their own compatible measures of effectiveness that measure attributes or properties of system behavior that support the system level mission. For GEOSS, the issues at the system level are related to interoperability, openness and sustainability. The measures are evaluated independently at each development level, but they are all related in that they must be traceable through the hierarchical framework to measure the effects of not meeting a lower measure on the higher measures and our SoS level objectives.

The legacy approach to system verification was primarily driven by industry standards that perform verification by means of documents and requirements tracing. The SoSE process extends the legacy method by adding the capability to validate requirements and design with a self validating architecture.
Fig. 9. SoSE “V” [9].

Fig. 9, shows the legacy verification approach (inner V) relative to the much more comprehensive SoSE integrated verification and validation approach.

The system architecture model is essential to the success of the development team on both sides of the development “V”. Stakeholder requirements are inputs to both the system architecture model and the associated system specifications. However, the system derived requirements originate through development of the system architecture model and are documented there. The derived system requirements are extracted from the system architecture model and documented in the system requirements specifications. After generating and peer-reviewing the SoS-level documentation, requirements validation is performed. In performing requirements validation, requirements are compared to customer expectations, enterprise rules, and external constraints. Variances and conflicts are identified architectural elements, and the baseline requirements are established.

The established SoS requirements baseline can be analyzed and compared with the mission/enterprise and business constraints based upon defined risk levels, project resources, technology limitations, and program objectives. The established requirements baseline can also be compared with the external constraints to ensure that the technical requirements meet applicable national and international laws such as applicable environmental standards, acquiring agency policies, and T&E facility requirements.

An example benefit of the system architecture model is that during selection of the best system functional architecture, the development products can be verified and system documentation can be generated for peer review. For example, in GEOSS the architecture model will help the developers readily identify all of the systems participants that may be grouped as a distinct class of systems that may be sensors or communication devices. The system objects in the model can be verified by ensuring that all of the SoS and system object descriptions are complete and consistent.

Then, the system model decomposition diagram, that shows the decomposition from the SoS object to the system objects, can be updated. Completeness and consistency of the SoS and system interfaces can also be readily verified using the system model to ensure that all of the SoS and system interface descriptions are complete and consistent. Then the SoS model interface diagram can be updated.

The architect translates problem domain concepts of the customer into the solution domain concepts of the developer. For the SoS to be built as envisioned, the architect develops a system of system architecture model to communicate the vision and
track construction against it. Architecture models are abstractions of the unrealized system.

Architecture model contents are dictated by the purpose of the model. Some architecture models are used for relatively abstract SoS concept development while others are used for more detailed SoS and system development and design. In each case the resolution and scope of the SoS architecture model should be scaled to meet the needs of the particular activity. Architecture models are constructed to relate system functionality to other architectural attributes such as reliability, manufacturability, interoperability, or virtually any other quality attribute that a particular type of development requires.

These attributes are modeled so we can have reliability, manufacturing, logistics supportability models, and level of systems interoperability capabilities model, etc. Most GEOSS operational nodes or systems will not need to address all of these quality attributes. However, most developers or operators of GEOSS operational nodes or systems should address those attributes that are required to meet their respective performance and maintenance levels.

For example, those GEOSS participants purchasing commercial systems may not be concerned about the manufacturability of the system, but they should be very concerned about the reliability. The results of these models may be documented in a quality attribute tree whose results are included in the architecture. Architecture models minimize the risk that the system stakeholders are not aware of significant elements of the architecture. Each architecture model is intended to provide insights into particular aspects of the system and related issues.

The system model and the use of common processes and tools result in all of the development team activities being based on a single, common product data set, Fig. 1. To support such attributes as interoperability, the models can be built upon a collection of recognized best practices. GEOSS is establishing a registry of “best practices” for this purpose.

It is imperative that GEOSS experience the benefits of an SoSE Process equivalent to the one shown in Fig. 2. The SoSE Process used must be architecture-centric, model-based, and user-driven. Architecture-centric means that the SoS/System architecture model serves as the primary artifact for conceptualizing, constructing, managing, and evolving the SoS. The architecture model facilitates dialog between stakeholders by having a common notation and an intuitive depiction of the system structure and behavior. This approach provides a continuous model stream from the SoS Goal level to the architecture subsystem base level in a notation that supports all of the associated engineering design disciplines.

Enterprise-driven commercial and military SoS normally have a relatively high operational control of the involved systems and can therefore expect that systems or classes of systems assigned to an SoS, for selected missions, will participate. An all-volunteer organization-driven SoS like GEOSS will not have high operational control of the systems that comprise the SoS. All of the contributing data systems will be voluntary participants in the GEOSS SoS. Therefore, the GEOSS SoS architecture must be developed with a robustness that allows the SoS to operate with a minimum loss of overall effectiveness as systems drop in and out of the SoS.

The SoS/System architecture model also significantly improves the traceability required across a product’s lifecycle by clearly defining functional requirements and system elements in each cycle of an incremental or spiral development (e.g., capability-based acquisition). In fact, the concept of spiral development, in which the full development occurs in a series of stages, each with more capability, is essential to the GEOSS development approach because of the continuing development of national systems which elect to participate in GEOSS. A stable architecture model is essential for spiral development and enables a system architecture definition at all system levels, the associated specification definition, and the needed design comprehension and realization.

Systems developed today are in the field for decades. The system architecture should be readily adaptable to future changes in a cost-effective manner. The primary characteristics of a robust architecture are evolvability and flexibility of the system.

Architecture “evolvability” is the ability of the architecture to handle future upgrades. This requires that the SoS architect identify, coordinate, and document system capabilities; functionalities; and technology maturation requirements for future program spirals, where they can be defined. There will be many cases over timeframes of decades where neither the evolution of technology nor the directions of national authorities in authorization of Earth observation and information systems can be adequately predicted. This then requires that the GEOSS architecture be designed as an open SoS, where the interfaces and interoperability specifications are simple and durable. For example, GEOSS architecture should minimize the complexity of the standard reporting communications structure that reporting agencies will need to use to be a part of the contributing participants.

Flexibility is the ability of the architecture to handle changes, new features, or new knowledge late in the development cycle or after operational implementation is mature without significant consequences for costs, schedule and technical achievability. A flexible architecture is needed to accommodate technology changes that include producing new alternatives and obsolescence affecting the continuation of the current systems. For example, a flexible SoS GEOSS architecture feature might be that the data services provided by a particular sensor have been modularized in such a way that a new sensor can be readily adapted to provide the same data without altering the GEOSS architecture.

Evolutionary systems require a process to plan, define, and prepare for program spirals. Key to this process is identifying and assessing technologies, processes, and system attributes required to support future capabilities. This process is critical to a successful evolutionary program.

The SoS architecture development process provides a structured process to identify and document SoS capabilities, functionalities, and maturation requirements needed for future program spirals. Documents that outline the evolution of SoS capabilities and functionality, and the program risk reduction for each spiral are needed. These documents outline the SoS evolution and technology maturation roadmap for progression from program spiral to spiral.
IX. CONCLUSION

A summary of the principal architecture and interoperability features have been collected, as noted earlier. The first is to build a flexible architecture and integration framework on a set of reusable components. This includes the ability to: leverage existing external and internal standards, architectures, and models; capture future capabilities through open architecture. Support wide range of business processes and environments; enable disconnected/connected modes for all implementations; and Integrate efforts for SoS development using distributed teams. The architecture must be configurable and scalable to customer needs and leverage robust systems and processes for global interoperability.

This paper addressed the evolution in approach as organizations have evolved from standalone platform providers to an architect and implementer of SoS. This has stimulated the development of new, innovative SoSE approaches that surpass current industry state-of-the-art systems engineering processes. The simple scaling of the former process was ineffective in dealing with global distributed large-scale integration efforts of network-centric systems of systems.

The SoSE Process in this paper represents a significant breakthrough for defining net-centric SoS and the associated systems such as the GEOSS. The SoSE Process is a complete, detailed, and systematic development approach. The SoSE Process is a unified approach for system architecture development that involves all of the engineering disciplines. The process provides a single model of the system architecture and the associated requirements to be used for hardware and software design without translation or restatement.

The new architecture-centric, model-based systems engineering process emphasizes concurrent development of the system architecture and system interface and interoperability specifications. Specifically, we have developed explicit mechanisms to more completely define the interoperability of functions, data, control, events, interfaces, operations, and timing that the new more complex architectures require.

REFERENCES