An Implementer's View of the Evolutionary Systems Engineering for Autonomous Unmanned Systems

Chris Scrapper and Ryan Halterman
SPAWAR Systems Center, Pacific
San Diego, CA

Judith Dahmann
Systems Engineering Technical Center
The MITRE Corporation
McLean, VA

***Abstract*—This paper presents the application of an Evolutionary Systems Engineering Model for Unmanned systems. It is a novel approach for research, development, test, and evaluation of unmanned systems. The Evolutionary Systems Engineering Model for Unmanned System is based on the System of Systems (SoS) Systems Engineering (SE) Wave Model [2] adopted by the DoD as best practices for Systems of Systems System Engineering [3], [5]. The SoS Wave Model was adopted by SPAWAR Systems Center (SSC) Pacific's Unmanned Systems Group to enable the agile and rapid evolution of autonomous unmanned systems. It provides the backplane for the Unmanned Systems Integration, Test, and Experimentation (UxSITE) capability at SSC Pacific. It establishes a systematic process for technology insertion and overarching strategies for managing risk and ensuring key capability objectives are met as the system evolves. Implemented as a continuous improvement process, the Evolutionary System Engineering approach provides flexibility and adaptability to address the inevitable changes in both the technical landscape and the larger operational environment. This paper provides a description of the Evolutionary Systems Engineering Model, its drivers, and how implementation of the SoS Wave Model addresses them.**

I. Introduction

The development of autonomous unmanned systems represents a set of challenges that share many of the same cross-cutting issues as traditional SoS. These challenges arise from the need to tightly integrate component technologies in a federated system where components may be owned and developed independently. Additionally, the technical landscape of unmanned systems is changing rapidly. Furthermore, the cost of building and maintaining development platforms often limits their number and availability—especially for larger unmanned systems. A disciplined system engineering approach addresses these dynamic, heterogeneous development challenges. In 2013, the SSC Pacific Unmanned Systems Group adopted the SoS Implementers' View (Wave Model) [2] approach to address issues associated with the distributed development, integration, test, and experimentation of autonomous unmanned systems.

The model for the Implementer's View of SE for SoS [2] is shown graphically in Fig. 1. This is a conceptual view of SoS evolution based on a set of logical steps: SoS analysis, SoS architecture evolution, planning and orchestrating updates to a SoS implemented as a set of overlapping 'waves' of activity with feedback both within and across waves. The model recognizes the need for disciplined iterations to systematically address impacts of inevitable change, supported by a backbone of ongoing analysis and architecture evolution.

The UxSITE capability at SSC Pacific tailors the original SoS Wave Model [2] to support the integration, test, and experimentation of autonomous unmanned systems. It applies to this situation where there has been an explicit decision to apply systems engineering across the SoS. The tailored model, shown in Fig. 2, decomposes a wave into four overlapping phases: Conduct Analysis, Evolve System Architecture, Integrate Capability Enhancements, and Validate System. This decouples the development and integration processes and expedites delivery of critical technologies.

Note that the Conduct Analysis phase is depicted as a set of simultaneous activities. This is to indicate that each system or technology proceeds largely independently through this phase. To give a sense of time scale, each Conduct Analysis phase notionally lasts six months; this serves as the drumbeat of the effort.

A. Why the Wave Model for Unmanned Systems?

Expediting delivery of autonomous capabilities to the Warfighter benefits from a continuous improvement process for assessing capabilities and limitations of the autonomous system, integrating maturing technologies based on key performance parameters, and reducing risk by understanding performance tradeoffs and associated cost as the system evolves. Central to this continuous improvement process is the ability to measure and accumulate evidentiary information to facilitate the agile and rapid response to uncertainty and unexpected issues arising during the development process. The ability to continuously measure and inter-compare results is paramount to the identification required to mitigate risk in a timely-manner. This ensures critical capability objectives and functional requirements are being met.

The rapid cycle and overlapping nature of the Wave Model necessitates the system and subsystems are under continuous formal assessment—even with research, development, and integration occurring across largely independent organizations. This philosophy of "test early, test often" facilitates the timely acquisition of evidentiary information needed to quickly iden-

IEEE Systems Conference, Orlando, FL, 18-21 April, 2016

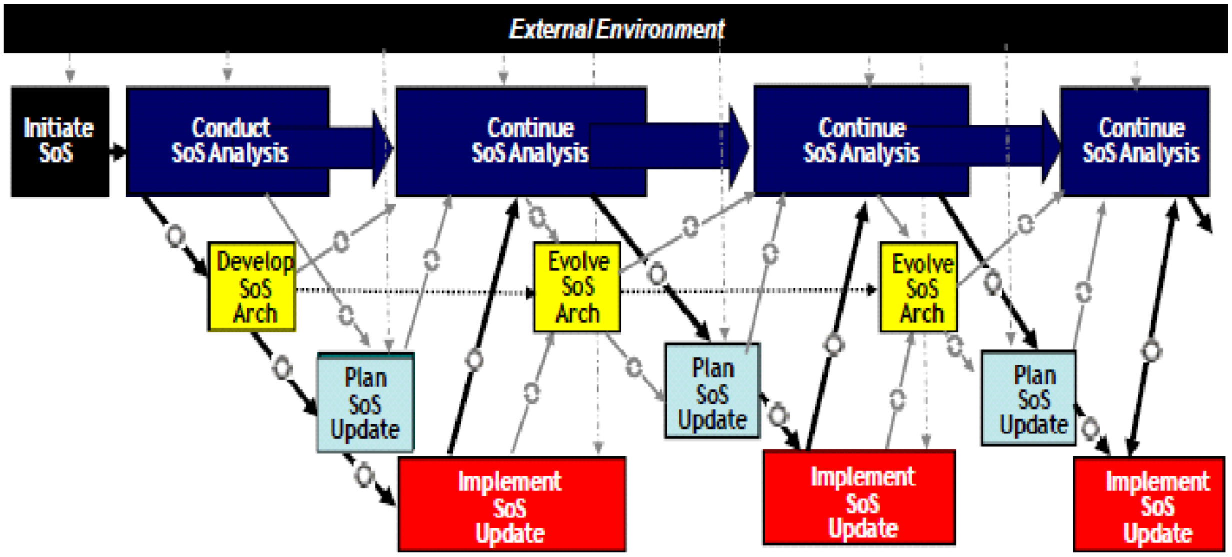

Figure 1. Depiction of the SoS Implementer's View (Aka 'Wave Model') [2]

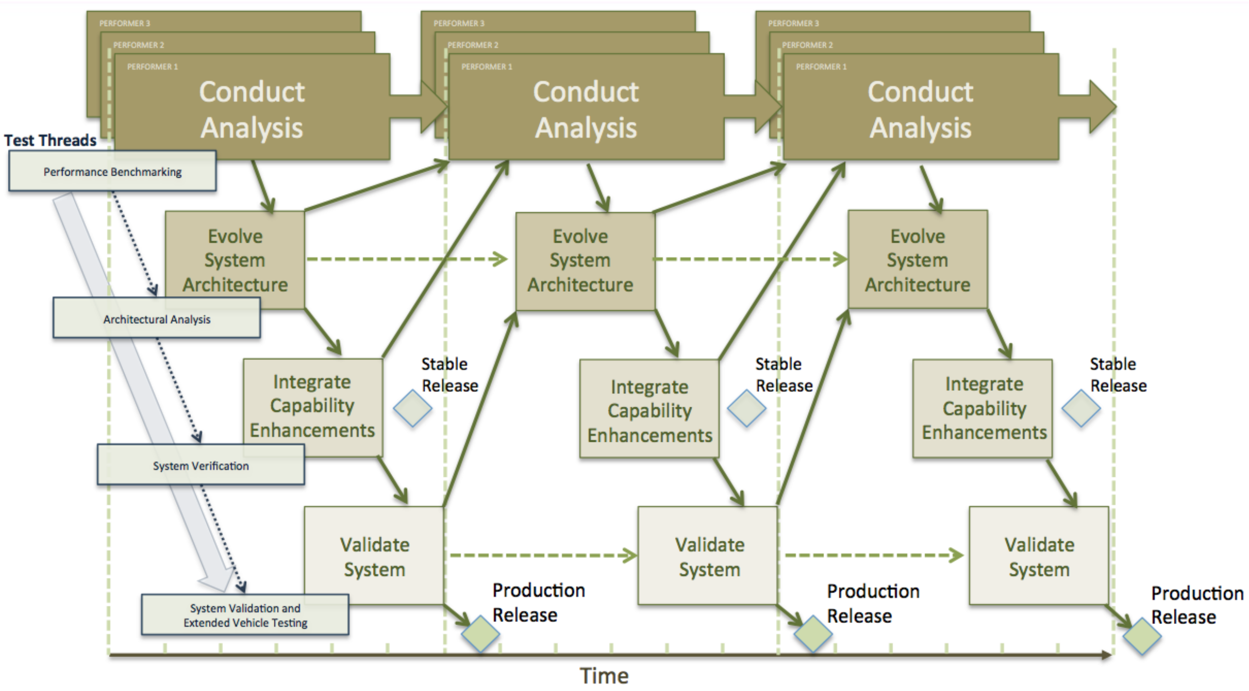

Figure 2. Evolutionary Systems Engineering Model for Unmanned System. This model tailors the original 'Wave Model' approach [2] for use in research and development of autonomous unmanned systems.

tify risks. Decoupling the development and integration processes fosters innovation by preventing a “fail fast” approach in the former from being overwhelmed by the latter.

II. Unmanned Systems Integration, Test, and Experimentation (UxSITE)

The Evolutionary Systems Engineering Model is used by the SSC Pacific Unmanned Systems Group within an overarching integration approach termed Unmanned Systems Integration, Test, and Experimentation (UxSITE). The UxSITE approach was developed in an effort to make the government more more technologically responsibility for investments made with taxpayer funds. It enables the government to benefit by leveraging all available technology to build, maintain, and extend a capability base in order to achieve higher levels of autonomy. In order to do so, the government takes increased technical responsibility for the overall architecture design and implementation. Functional decomposition of the project (through a modular, open architecture) establishes a common lexicon and foundations for interface expectations. Combined with data licenses granting government purpose rights at the architecture level and some visibility at the capability level, this provides a foundation that facilitates leveraging investments across myriad efforts to expedite the delivery of autonomous capability with increased efficiency and flexibility in terms of cost and schedule. The UxSITE approach is composed of the following elements: Vision, Modular Open Architecture, Continuous Integration Environment, Test and Experimentation Master Plan, and the System Engineering Plan (comprising the Evolutionary Systems Engineering Model). The following sections provide a brief overview of the elements of this larger strategy as context for the backdrop surrounding the model. The last of these elements, the Systems Engineering Plan, constitutes the Evolutionary Systems Engineering Model itself which is described in greater detail.

A. Vision

A clear vision provides accountability and empowerment to individuals across the effort. For one effort, the Multi-Role Autonomous Ground Vehicle (MAGV), this is distilled, at a high level, as the overarching technical objective of a low-cost autonomous tactical expeditionary navigation system. At a lower level it includes artifacts such as the concept of operations, requirements, and system definition.

B. Modular Open Architecture

A modular, open architecture benefits the government and supports the Office of the Secretary of Defense (OSD) Better Buying Power guidance [6] to “do more without more”. It provides flexibility and cost savings by establishing the structure for component interoperability through interface and functionality specifications. This enables evolutionary development by facilitating piecemeal component implementation or upgrade vital to building higher levels of autonomy.

The reference model architecture, shown in Fig. 3, represents, at a high level, a Modular Open Architecture. It serves as a guide to support the design, implementation, and organization of the intelligent unmanned systems [1], [2]. This model manages the complexity of the system through a hierarchical decomposition of functional nodes for sensor processing, world modeling, behavior generation, and mission payload interfaces. The hierarchical structure defines knowledge requirements and allocates functionality for achieving higher levels of autonomy. This also provides a lexicon and technical roadmap for assessing capability gaps, dependencies, and risk for prioritizing investments.

C. Continuous Integration Environment

The Continuous Integration Environment (CIE) is a repository of data and knowledge, with customized viewing templates for different needs. It is data driven rather than document driven. A concept analogous to Model-View-Controller (MVC) is employed for data exploitation. A view can be any output representation of information, such as a chart, a diagram, or tabular information, with multiple views of the same information customized to meet requirements of different stakeholders such as sponsors, management, technical integrators, performers, testers, or whomever. It is built using best-of-breed applications for knowledge management, issue and requirements tracking, source code repositories, continuous integration, and automated testing. Within it, functional capabilities are linked to technical data, risks, tasks, etc. The importance of the CIE as a means for cross-correlation of information is further discussed in [4].

D. Test and Experimentation Master Plan

The Test and Experimentation Master Plan (TEMP) defines the overarching assessment strategy and follows a crawl, walk, run approach. It includes a suite of test methods that exercise the system at a variety of levels from common software tests through full, custom system tests. The application of this test regime accumulates evidentiary information over the course of the program.

The test regime is composed of the following test levels: build, software compliance, unit, regression, experimental, and operational. Build tests comprise standard software build automation tools that verify that all subsystems build properly. Software compliance tests consist primarily of automated static code analysis tools and manual code reviews to ensure that coding standards are met. Automated unit tests are created by the development and integration teams to provide an assertion of expected output given known input for each functional element. Automated regression tests use simulated or recorded data to ensure that current development does not break existing functionality. They are distinguished from unit tests in that they are generally system-performance tests, while unit tests reside at a lower functional level. Experimental tests consist of elemental hardware-in-the-loop tests that exercise the system or subsystem in isolated tasks. Operational tests are system-level tests performed in operationally relevant environments and are designed to emphasize known system failure modes and uncover unknown failure conditions.

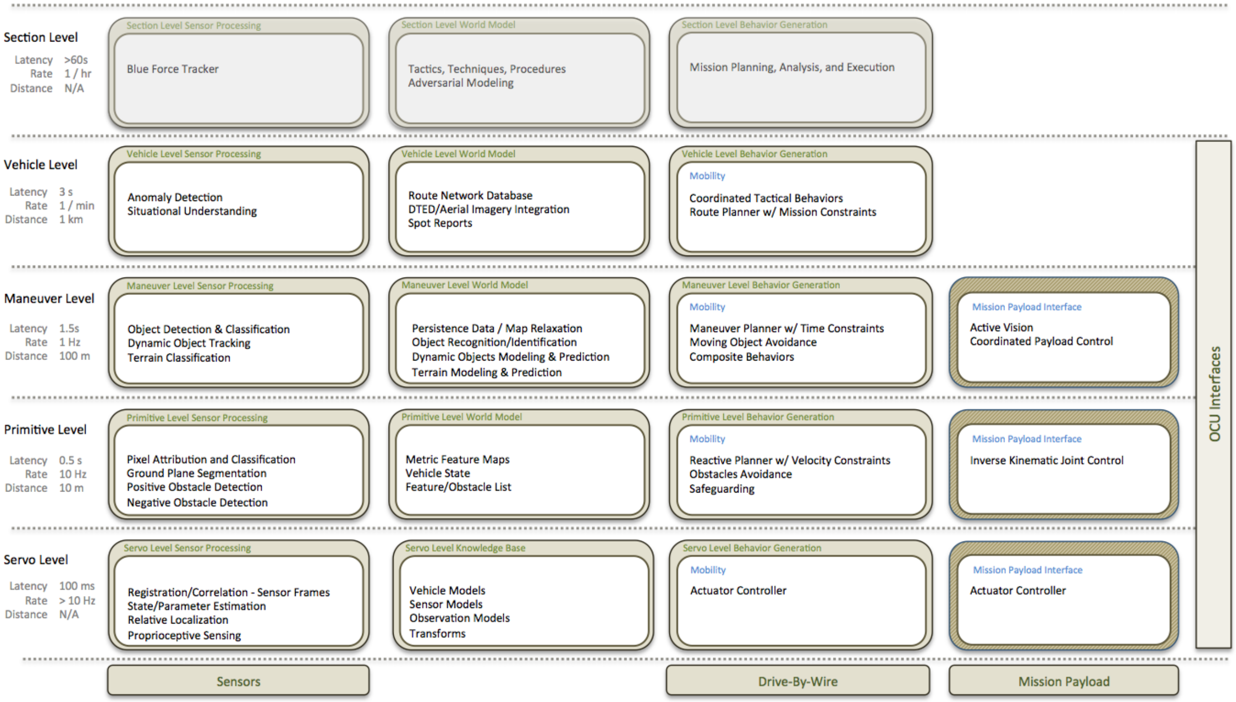
Section Level
Latency >60s
Rate 1 / hr
Distance N/A
Section Level Sensor Processing
Blue Force Tracker
Section Level World Model
Tactics, Techniques, Procedures
Adversarial Modeling
Section Level Behavior Generation
Mission Planning, Analysis, and Execution
Vehicle Level
Latency 3 s
Rate 1 / min
Distance 1 km
Vehicle Level Sensor Processing
Anomaly Detection
Situational Understanding
Vehicle Level World Model
Route Network Database
DTED/Aerial Imagery Integration
Spot Reports
Vehicle Level Behavior Generation
Mobility
Coordinated Tactical Behaviors
Route Planner w/ Mission Constraints
Maneuver Level
Latency 1.5s
Rate 1 Hz
Distance 100 m
Maneuver Level Sensor Processing
Object Detection & Classification
Dynamic Object Tracking
Terrain Classification
Maneuver Level World Model
Persistence Data / Map Relaxation
Object Recognition/Identification
Dynamic Objects Modeling & Prediction
Terrain Modeling & Prediction
Maneuver Level Behavior Generation
Mobility
Maneuver Planner w/ Time Constraints
Moving Object Avoidance
Composite Behaviors
Mission Payload Interface
Active Vision
Coordinated Payload Control
Primitive Level
Latency 0.5 s
Rate 10 Hz
Distance 10 m
Primitive Level Sensor Processing
Pixel Attribution and Classification
Ground Plane Segmentation
Positive Obstacle Detection
Negative Obstacle Detection
Primitive Level World Model
Metric Feature Maps
Vehicle State
Feature/Obstacle List
Primitive Level Behavior Generation
Mobility
Reactive Planner w/ Velocity Constraints
Obstacles Avoidance
Safeguarding
Mission Payload Interface
Inverse Kinematic Joint Control
Servo Level
Latency 100 ms
Rate > 10 Hz
Distance N/A
Servo Level Sensor Processing
Registration/Correlation - Sensor Frames
State/Parameter Estimation
Relative Localization
Proprioceptive Sensing
Servo Level Knowledge Base
Vehicle Models
Sensor Models
Observation Models
Transforms
Servo Level Behavior Generation
Mobility
Actuator Controller
Mission Payload Interface
Actuator Controller
OCU Interfaces
Sensors
Drive-By-Wire
Mission Payload

Figure 3. Reference Model Architecture for Autonomous Ground System describing functional decomposition of an autonomous system. This represents, at a high level, a modular, open architecture.

While high-quality automated software test and analysis tools (e.g. lint, gtest, unittest, selenium, soapui, etc.), are widely available, tools for automating system-level tests of autonomous unmanned systems are less so. Because manual unmanned system testing can be tedious, time consuming, and error prone, during design of new tests, we emphasize the development of automated data acquisition and analysis tools to support them. This includes a flexible web-based automated system test tool (see Fig. 4) used at milestone test events to easily collect, analyze, and format data of interest. The metrics and evidentiary information accumulated through continuous application of the test regime provide point estimates for building confidence and assessing technical and program risk. The systematic testing approach contributes to all stages of the development lifecycle [3], facilitating the agile and rapid evolution of unmanned systems.

E. System Engineering Plan

The UxSITE System Engineering Plan comprises the Evolutionary System Engineering Model. It is depicted in the strategy chart, shown in Fig. 5, which illustrates the major activities and artifacts for the development, assessment, and integration of new and maturing technologies into the baseline autonomous system. The figure shows one full wave (middle diagonal section) and the ending and beginning portions of the surrounding waves (shaded at lower left and upper right). Each technology introduced follows a systematic progression from conception in one wave to validation in the same or a subsequent wave. The following section describes the activities and artifacts associated with the different phases of the Evolutionary Systems Engineering model.

III. Evolutionary System Engineering Model

The Evolutionary System Engineering Model decomposes each wave into four overlapping phases: Conduct Analysis, Evolve System Architecture, Integrate Capability Enhancements, and Validate System (see Fig. 5). This facilitates decoupling the maturation (occurring primarily within Conduct Analysis) and integration (occurring primarily within the remaining three phases) processes.

A. Conduct Analysis

Conduct Analysis focuses on technology maturation and risk reduction for new capability enhancements. This requires close technical collaboration between capability developers and the integration team to provide better insight into the development challenges and risks associated with technical approaches. This phase consist of a series of technical reviews, milestones, and test events to compile evidentiary information for assessing the capabilities readiness for integration.

The System Requirements Review (SRR) and System Functional Review (SFR) represent a program refresh and draw from the Technology Readiness Assessment (TRA) of the previous wave to determine the best next step. In the process, they also draw from an updated the Risk Registry (Risk). The Risk Registry tracks all known system and subsystem risks and issues. New capabilities—or improvements to existing ones—are identified to address high priority risks.

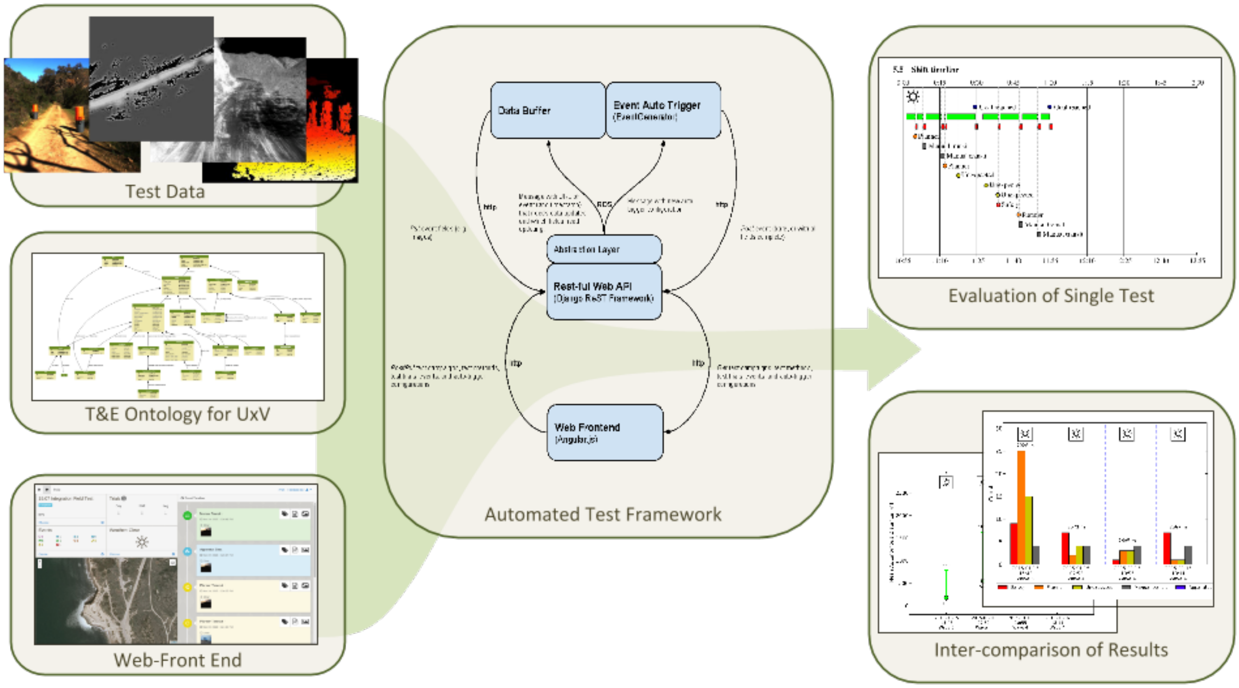

Figure 4. Automated system test tool facilitates inter-comparison of results, monitoring technical performance parameters over the duration of the campaign, and analysis of performance trade-offs.

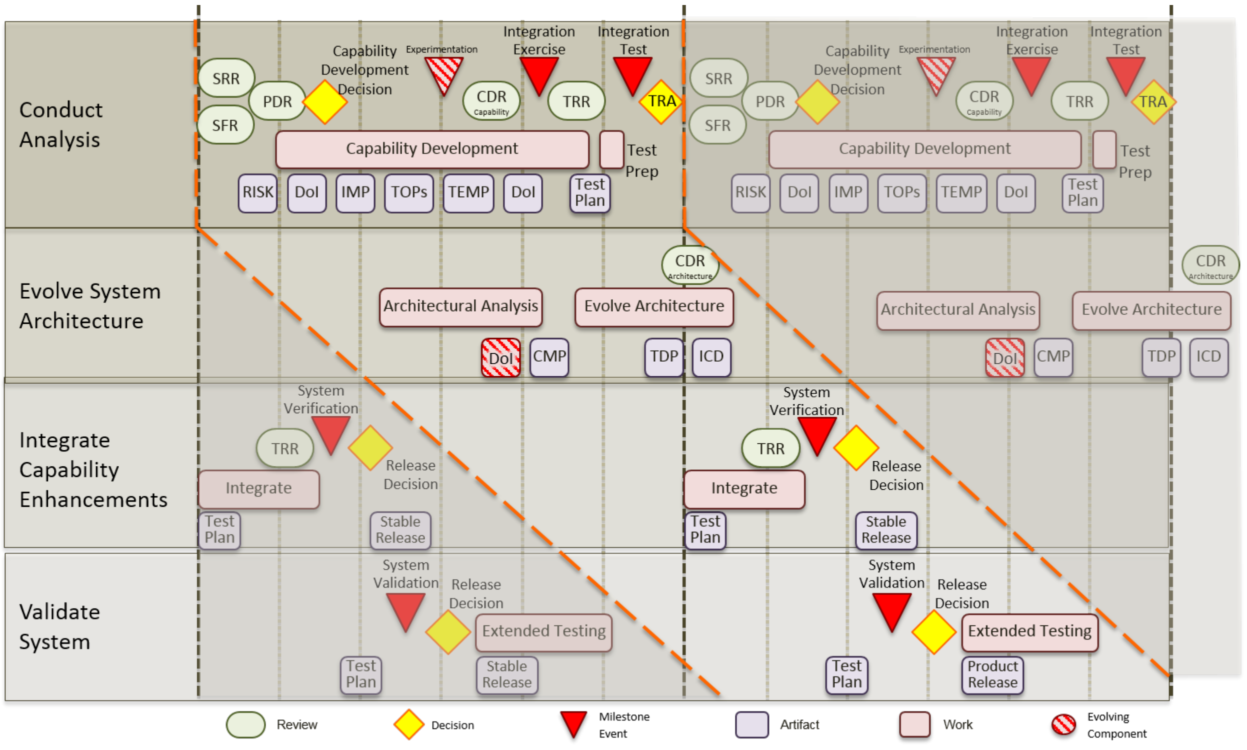

Figure 5. SSC Pacific Evolutionary Systems Engineering Strategy Chart illustrating the major activities and artifacts for the development, assessment, and integration of new and maturing technologies into the baseline autonomous system.

Development teams subsequently prepare initial designs for potential new capabilities to be presented in Preliminary Design Reviews (PDRs). The PDRs include an initial Declaration of Intent (DoI), which serves as a proposal of the development work to be performed over the course the coming wave by that particular development team. The DoI lists the capabilities the developer expects to be transitioned at the conclusion of the current wave. This gives the integration and assessment teams the ability to anticipate and develop necessary test methods. Based on the PDRs and Risk assessment, the program officers (with input from the integration team) prioritize capability development for the new wave (Capability Development Decision). At this point, Capability Development of the prioritized new capabilities begins in earnest; maturation of previously initiated capabilities that have not yet been integrated into the baseline system also continues.

Following the Capability Development Decision, the program officers update the Integrated Master Plan (IMP). The integration team updates the Test Operating Procedures (TOPs) to include test methods for newly planned capabilities or updates to existing test methods. The integration team also updates the TEMP to incorporate any revisions necessary for the coming wave.

Throughout the capability development process, the development platforms and test facilities are leveraged for dedicated Experimentation by the various development teams. Experimentation is differentiated from other milestone events in that they are less formal and their agenda is largely prescribed by the associated development team. Still, due to the limited number and availability of both development platforms and test facilities, enforcing well-structured experimentation is important to make efficient use of these limited resources.

Critical Design Reviews (CDRs) are presented shortly after the midpoint of a given wave, after capabilities are sufficiently developed that the implementation is relatively well understood and characterized. CDRs include an updated Declaration of Intent which then serves as an informal agreement of the development work to be performed over the course of the current wave.

Toward the end of the Conduct Analysis phase, the assessment team performs formal testing of the new and maturing capabilities intended for integration during the current wave. These tests are referred to as Integration Exercises. They comprise a compendium of elemental test methods that stress individual capabilities and the system as a whole. The elemental test methods are designed to be portable, repeatable, and reproducible in a fashion similar to those developed through the ASTM E54.08.01 Committee on Homeland Security Applications; Operational Equipment; Robots [7]. The characteristics of portability, reproducibility, and repeatability enable the elemental tests to be easily co-located with development or integration teams, thus facilitating continual access.

At the close of development work within a wave, performance metrics derived from Integration Exercises are presented by development teams at Test Readiness Reviews (TRRs). These serve as a gatekeeper to the larger-scale field test that culminates a wave's Conduct Analysis phase—the Integration Field Test. This test is a system-level test that assesses system and subsystem performance through operationally relevant test scenarios over larger areas—typically leveraging local military test ranges. Because of limited availability at preferred test ranges, the TRR aids in efficient allocation of testing resources by ensuring that valuable field test time is not allocated to capabilities that are unlikely to meet performance standards for integration into the baseline system.

The TRA, prepared by the assessment team, is a report that recommends for integration or continued maturation of capabilities brought to the Integration Field Test. It references metrics and analysis derived thereby and those from the Integration Exercises to assess the risk/benefit outlook for each potential new capability's effect on overall system performance. Based on this recommendation, program officers determine which capabilities will be integrated into the baseline system.

B. *Evolve System Architecture*

Evolve System Architecture focuses on the development and maintenance of an open, modular architecture to accommodate new capabilities under development and to ensure compliance with coding standards, interfaces, and architectural constraints. This includes the configuration and optimization of software and hardware to meet computational and networking requirements.

Following completion of the Capability Development Decision, the integration team begins Architectural Analysis. This considers the interfaces and architectural constraints of capabilities expected for integration in the current or a subsequent wave and recommends necessary changes via the architecture Declaration of Intent (DoI). It also considers upgrades to and improvements for overall system (hardware and software) maintainability and reliability. The integration team prepares a Configuration Management Plan (CMP) that describes the process for deploying the architecture updates to the unmanned systems and development workstations across the program. The architecture DoI and CMP are advertised across the program and input solicited for negative implications to secondary capabilities. The integration team then performs this work in Evolve Architecture, resulting in a new Technical Data Package (TDP) and Interface Control Document (ICD). The results are presented to program officers via the architecture CDR.

C. *Integrate Capability Enhancements*

Integrate Capability Enhancements focuses on the technical integration, optimization, and the verification of system and subsystem performance in order to assess trade-offs and ensure technical objectives have been met prior to the stable release of the integrated baseline system. This includes monitoring and tracking of technical performance parameters from system and regression testing and execution of corrective actions to mitigate technical risk.

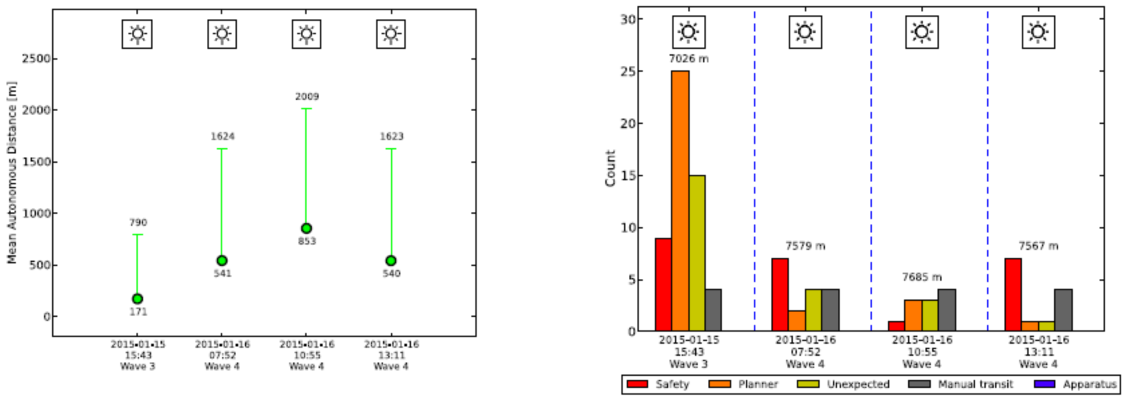

Figure 6. Experimental results over two consecutive waves showing 4x improvement in mean distance traveled and 85% reduction in the number of operator interventions.

Shortly after completion of the Technology Readiness Assessment in the Conduct Analysis phase, the integration team prepares a Test Plan for the soon-to-be integrated system. The integration team, working with development teams, performs Integration for capabilities approved in the TRA. The newly integrated system is then evaluated through elemental test methods according to the Test Plan prior to the verification Test Readiness Review (TRR) and larger scale System Verification test in operationally relevant scenarios. Once again, the TRR serves to ensure that the integrated system has performed sufficiently well in relatively inexpensive regression and elemental testing to justify incurring the larger logistical and material costs of full-scale scenario testing.

Following System Verification testing—and based on performance metrics derived therefrom—a Release Decision is made to distribute the new Stable Release for use program-wide as the updated development baseline. Capability Development work within the already-begun Conduct Analysis phase of the following wave is, from this point, based on this updated Stable Release.

D. Validate System

In general, Validate System focuses on effectiveness, suitability, and utility to ensure the system meets stakeholder requirements and operational objectives in relevant environments. However, for Science and Technology (S&T) programs, this phase focuses on the validation of the operational concept and evaluation of the system capabilities to ensure the S&T objectives are met. This includes system and software safety assessments, hazard tracking, and coordination of technical demonstrations.

This phase proceeds similarly to System Verification with a Test Plan, System Validation, Release Decision, and Product Release. Its focus is vetting the operational concept and feeding updated operational landscape and requirements analysis to System Requirements Reviews in subsequent Conduct Analysis phases. To do this, a variety of stakeholders are engaged via workshops that present the operational concept, demonstrate the system state, and solicit feedback on both via formal stakeholder surveys.

IV. Experimental Results

Experimental results shown in Fig. 6 highlight the integrity of the model by illustrating significant improvement in the performance of an autonomous ground system in a short-period of time. Over the course of one six-month wave, for a particular operationally relevant scenario, the system saw a 4x improvement in the mean distance traveled autonomously without operator intervention. This was accompanied by a decrease in the number of operator interventions by 85% over the same period. Over the course of another six-month wave and for another operationally relevant test scenario divided into contiguous segments for the purpose of metrics and analysis, the system saw a 2x increase in the rate of segment completion without operator intervention.

The analysis and metrics derived from this continuous testing approach allows for ongoing assessment of progress toward development objectives, enabling accelerated decisions based on quantitative results. The implementation of the Evolutionary System Engineering Model within the UxSITE strategy provides product lifecycle management that supports the DoD Better Buying Power initiative [6] by producing systems that are more maintainable and extensible.

V. Summary and Conclusion

This paper has presented the application of an Evolutionary Systems Engineering Model for Unmanned Systems, a novel approach for research, development, test, and evaluation of unmanned systems. As presented, The Evolutionary Systems Engineering Model for Unmanned System is based on the System of Systems (SoS) Systems Engineering (SE) Wave Model [2] adopted by the DoD as best practices for Systems of Systems System Engineering [3], [5]. The SPAWAR Systems Center (SSC) Pacific's Unmanned Systems Group adapted the SoS Wave Model as a framework for their Evolutionary Systems Engineering Model. The paper highlights the critical elements of the SPAWAR strategy—shared vision, modular open architecture, the continuous integration environment and the test and experimentation master plan—and how these have used the element of the wave model to implement disciplined iterations to systematically address impacts of inevitable change, supported by a backbone of ongoing analysis and architecture evolution based on modular design.

REFERENCES

[1] DoD, “Defense Acquisition Guidebook,” Washington, D.C.: Pentagon, May 2013.

[2] J. Dahmann, G. Rebovich, J.A. Lane, R. Lowry, K. Baldwin, “An Implementers’ View of Systems Engineering for Systems of Systems,” Proceedings of IEEE International Systems Conference 2011, April 4-7, 2011, Montreal, Quebec, Canada.

[3] J. Dahmann, R. Heilmann, et al. "SoS Systems Engineering and Test & Evaluation: Final Report of the NDIA SE Division SoS SE and T&E Committees," Results of a National Defense Industrial Association (NDIA) Systems Engineering Division task implemented by members of the Systems of Systems and Test and Evaluation Committees, March 2012.

[4] R. Dove, B. Shindel, and C. Scrapper, "Agile Aircraft Installation Architecture In a Quick Reaction Capability Environment.Agile Systems Engineering Process Features Collective Culture, Consciousness, and Conscience at SSC Pacific Unmanned Systems Group," Proceedings INCOSE International Symposium, Edinburg, July. 2016., in press.

[5] DoD, “Systems Engineering Guide for Systems of Systems, version 1.0,” Washington, DC, USA: U.S. Department of Defense (DoD). August 2008.

[6] DoD, “Implementation Directive for Better Buying Power 3.0 – Achieving Dominant Capabilities through Technical Excellence and Innovation,” Washington D.C.: Under Secretary of Defense, April 2015.

[7] ASTM, “Subcommittee E54 on Operational Equipment,” 2016, http://www.astm.org/COMMIT/SUBCOMMIT/E5408.htm.