A method of assessing the time-variant value of multi-domain architectures

Thomas Ford, David Meyer, John Colombi, Brian Scheller, and Cody Palmer

Abstract
Modeling and simulation continues to improve the ability to evaluate future complex concepts and conduct early systems analysis. While the systems engineering community now advocates model-based systems engineering, the actual integrated methods and tools need to be refined. This research synthesizes descriptive architectural depictions of multi-domain (air and space) concepts, simulates these concepts through a physics-based model, collects several performance metrics, and integrates them using value-focused thinking. The proposed method enables the assessment of alternatives whose value can increase or decrease over time, depending on the design properties and operational scenario. Three surveillance concepts are proposed, made up of a low-earth orbit satellite constellation, with varying payloads, and extended by an unmanned air system flying waypoints in the area of interest. Uniquely, this method shows promise of depicting time-variant value assessments of these concepts across the simulation.

Keywords
Multi-domain, architecture, operational concept, value modeling

1. Introduction
The Department of Defense (DoD) recognizes that future combat environments will be less permissive than those of today and is proactively investigating future capabilities that could enable operations in these environments. The systems that implement these future capabilities must not only be able to penetrate the denied environment, but must also be able to survive and persist in an environment while still performing their respective missions. To this end, the Air Force is investigating innovative operational concepts that will ensure the ability to conduct operations effectively in a denied environment. Among these future operational concepts are those that recognize the possible utility and benefit of placing future capabilities in more than one domain (e.g., space, air, ground, or sea).

This research defines a decision analysis and model-based systems engineering (MBSE) approach toward assessing the military value of future multi-domain operational concepts. Specifically, we introduce and demonstrate the concept of time-variant value. The method represents a synthesis of systems architecting, physics-based simulation, and value-focused thinking (VFT). The systems engineering community would consider such a synthesis within the evolving concept of MBSE. The method is designed to present a model-based approach for making architectural decisions early in the system acquisition process and assessing the trade space of system architectural options prior to, or during, an analysis of alternatives (AoA).

A review and critical analysis of important publications that form the foundation of this research is presented in Section 2. This review addresses the pedigree and validity of using a synthesis of decision analysis and MBSE to develop a value-focused architectural assessment method. Next, in Section 3, the method for describing and assessing the value of architectural instantiations of an operational concept is presented. This is followed by an example in Section 4 showing how a candidate space–air operational
concept was assessed and how its value varies with respect to time. Lastly, recommendations for further evaluation and improvement of the method are discussed in Section 5.

2. Background

Decisions involving multi-domain solutions to operational needs are similar to many problems in the public domain that are inherently complex. These types of problems often require some subjective evaluation, are constrained by competing objectives, and involve probabilistic uncertainty.1 Because of this, the VFT2 approach is an appropriate decision analysis method for use in this research as it uses the concept of value to not only motivate the development of alternatives, but also to evaluate the alternatives.3 Because the solution space of multi-domain implementations of any one specific operational concept is infinite and each competing alternative can be complex, a value hierarchy, representing the objectives of the stakeholders, can be used to (1) constrain the solution space of acceptable alternatives and (2) make better decisions amongst the sea of alternatives. The value hierarchy translates subjective objectives of the stakeholder into objective values that can be quantified and assessed.1

There are two categories of value hierarchy construction, top-down and bottom-up.4 Within these two categories, different methods can be applied. One such method, called the “Gold Standard,” uses strategic vision and objectives to develop the value hierarchy.5 Pruitt et al.6 noted there is a precedent for using the Gold Standard within military circles. They described instances where not only was a value hierarchy developed based upon organizational strategic vision and objectives, but that doctrine (e.g., JCS Pub 3-14, “Military Space Operations Doctrine”)7 and other military publications can also form the foundation of a value hierarchy.6 There is also precedence for using a value hierarchy to score architectural alternatives in terms of system effectiveness and architectural quality.8 The 10-Step VFT Process9–11 has been used to develop value hierarchies for this purpose.12 In this research, a combination of the Gold Standard and the 10-Step VFT Process was used to improve a value hierarchy initially supplied to the research team. The initial value hierarchy was already based upon the strategic vision and objectives of an Air Force Command, but the research team added rigor by re-organizing it using the Joint Capability Areas (JCA) commonly relied upon in Department of Defense Architecture Framework (DoDAF)13 architecting.

The goal of this research was to describe a synthesis of multi-domain architecture, modeling and simulation, and value assessment that could be used in an automated or semi-automated fashion to generate architectural alternatives and score their value over time. Miller14 proposed a systems engineering approach that partially achieved this goal. His method generated value-focused alternative solutions for the Joint Capability Command and Control Management (JC3M) system that could be simulated and have their implementation costs estimated.14 Although a well-defined process, Miller’s method did not completely meet our needs as it was not focused on multi-domain architectural alternatives and did not explicitly describe the time variance of value of each alternative; instead, his method objectively analyzed performance criteria and life cycle cost resulting from the simulation. Our research team used an operational concept and value hierarchy to generate candidate architectural solutions, which could be modeled and simulated in such a way that architectural performance values could be generated. These values were then fed back into the leaf-level elements of the value hierarchy to create a feedback loop, which would allow the architectural alternatives to be refined and the time-variant value of each alternative to be captured.

In order for candidate architectural concepts and solutions to be modeled, they were first described as capability architectures using DoDAF. They were then refined into executable architectures whose performance could be simulated. The idea of executable architecting15–18 has been heavily researched since the early 2000s and has been established as a useful branch of systems architecting. Early methods of executable architecting were focused on Colored Petri Nets.18 More recent research recognizes that modern system design techniques increasingly make use of modeling and simulation tools and techniques as a method of executable architecting.19 For example, some have applied discrete-event simulations19 while others have combined the ideas of Colored Petri Nets with discrete-event simulation20, or performed agent-based simulations.18 Helle and Levier21 published a method of converting a traditional integrated architecture into an executable architecture.

The emergence of MBSE as a systems engineering philosophy generated a new wave of discussion pertaining to executable architecting. Ge et al.25 surveyed methods of creating executable architectures based upon the Unified Modeling Language (UML) and Systems Modeling Language (SysML) as well as Colored Petri Nets (2012). Some of these MBSE-focused methods of executable architecting have a specific focus, such as executable architecting of systems-of-systems (SoSs)23 and data-centric executable architectures based upon the DoDAF metamodel.24 Colombi et al.19 demonstrated that an executable architecture can be modeled and simulated using the Systems Tool Kit (STK) software developed by Analytical Graphics, Incorporated (AGI). In this research, Python computer code was written and used to automate
the STK Engine to order to automate the generation and simulation of architectural alternatives based upon the operational concept and value hierarchy supplied to the research team. Python was then used to capture and analyze the STK output. Although there have been demonstrations of the integration of discrete-event simulation and value assessment and a value measurement of a simulated system design, automated coupling of a multi-domain, architecturally based simulation and a value measurement with a feedback loop to iterate the architectural design has not been done previously.

3. Method
3.1. Overview
The method developed by this research is used to assess the comparative value of different architectural implementations of an operational concept. The method relates to the Decision Support Model Development Framework, described by Shoviak, that involves identification of a problem, development of a value hierarchy with associated weights and value functions, and use of the value hierarchy to generate and assess alternatives. Shoviak’s framework is based upon Keeney’s and Kirkwood’s work on multi-objective VFT and decision-making.

The method (Figure 1) described in this research can be summarized as follows. Given a purpose and an operational concept, a value hierarchy can be created that assigns value and weights to the concept’s specific mission tasks. Since there are numerous systems (or sets of systems), which can perform the mission tasks and be operated in different ways, the value hierarchy can be used to generate any number of candidate architectural solutions. The value hierarchy is also used to evaluate the efficacy of one architectural implementation over another in implementing the operational concept. This assessment is done by first modeling the system architecture so it can be simulated (executed), then capturing the performance of the system (in the context of any threats to the system, if applicable). The system performance parameters (or “consequences”) generated by the architectural simulation can be used as feedback inputs into the value hierarchy. The new output of the value hierarchy is a “score,” or total value, of how well the architecture implemented the operational concept. If the architectural simulation is time variant, then the output of the value hierarchy is also a function of time. Hence, analysis of the architectural simulation results and the associated value hierarchy output can show how well the system architecture implements the operational concept across all time, or at “important” points in time. Such times could include target illumination (sunrise, sunset), target activation (on, off) or target movement. Although system cost is not discussed in this research, it is feasible that system performance, architectural value, cost, and time can all be analyzed using the method described here. Succeeding paragraphs describe the method in more detail. The reader should also note in Figure 1 that iteration and feedback is necessary in refining scope, level of abstraction, concepts, architectures, and value assessments.

Figure 1. Method for creating and assessing the value of multi-domain architectures.
3.2. Identify purpose

Although there are many reasons for comparing architectural implementations of an operational concept, one important purpose is to support an AoA. Once a problem or need has been identified, which indicates a materiel solution may be required, the next logical step is to analyze options for solving that problem and meeting the need. The option analysis should reveal which alternative performs the best, costs the least, or is most maintainable, for example. Another purpose for applying the method in this paper pertains to operational use of an existing set of systems. The method is appropriate for investigating alternatives using both existing and conceptual systems.

3.3. Define operational concept

The operational concept forms the foundation for the value assessment. The operational concept is not just a paper document that describes “operations, functions, and activities [performed] in response to a range of future challenges,”26 but it is part of the model itself. The operational concept, whether institutional, functional, operational, or enabling, contributes the time horizon, assumptions, capabilities, sequences of actions, command relationships, desired end state, and other important elements27 to the model. For example, if the system is architected using DoDAF, then the operational concept defines the capabilities to be included in the CV-2 Capability Hierarchy model. Similarly, the actions described by the operational concept are modeled in both the OV-5a Operational Activity Hierarchy and the OV-5b Operational Activities Model. The operational concept’s command relationships can be expressed in the OV-4 Organizational Relationships model. The Operational Resource Flow Diagram (OV-2) can be depicted with conceptual block diagrams, using a SysML block definition diagram (bdd; see Figure 2). Any restrictions regarding the system’s concept of operation are captured in the OV-6a Operational Rules model.13 Some elements of the model are indirectly described by the operational concept. For example, the time horizon given by the operational concept not only defines the architectural epoch of the model, but also indirectly influences which external systems are modeled and employed. Figure 2 also captures external systems that could include Command and Control (C2), communications, and data...
processing. Other indirect influences on the model include a wide range of assumptions and environmental factors (e.g., geography, such as location and terrain, weather, season, buildings) and operating environment (e.g., targets or areas of interest, radio frequency spectrum characteristics, etc.).

3.4. Create value hierarchy

The value hierarchy provides the eventual “score,” or total value, of a particular architectural alternative in implementing the operational concept. Ideally, the value hierarchy should be developed independent of and prior to the architecture, but not independent of the operational concept. The reasons for this are as follows. The value hierarchy should not be biased toward any one possible solution, but should accurately reflect the needs (values) of those who operate the system and implement the operational concept. System operators should be careful in writing the operational concept and value hierarchy to not presuppose a solution to the problem or a response to the purpose for which the analysis is taking place. A value hierarchy should not be overly complex, but should include both value functions and weights for all leaf elements in the hierarchy\(^3,28\) in order to be sufficient for directly comparing alternatives. In addition, the element set of a value hierarchy must be complete, operational, decomposable, non-redundant, and of minimum size.\(^1\)

Although an initial value hierarchy was provided to the research team, other alternatives to the VFT approach were considered. Specifically, the Analytical Hierarchy Approach (AHP)\(^29,30\) and Alternative Focused Thinking (AFT)\(^31\) methods were investigated. Although there are benefits to each, the Office of Aerospace Studies (OAS) AoA Handbook discourages these methods, instead recommending a focus on best value be pursued.\(^32\) Some might note a caution within the handbook to avoid “roll-up and weighting schemes that...mask important information or...provide misleading results.”\(^32\) While that approach seems to discredit the use of value hierarchies, it should be noted this comment is in reference to how the results of an AoA are communicated to senior leaders and is not a prohibition against using a weighted value hierarchy.

3.5. Develop, model, and simulate architectures

Because the research sponsor had begun writing the operational concepts and DoDAF architecture, the research team made several early decisions. An MBSE approach toward modeling both the operational concept and candidate systems was taken to capitalize upon the architecture modeling already performed. Any additional architecting was done using DoDAF to maintain compatibility with the research sponsor. Enterprise Architect by Sparx Systems, Inc.\(^33\) was chosen as the architecting tool due to the research team’s familiarity with the product. The research team explored the integration of the descriptive architectural models with the physics-based simulation, together with an interface to the value hierarchy. According to the operational concept, the SoS was allowed to operate in both the space and air domains. In order to accurately assess the performance of this SoS, the model had to include both abstract and parametric architectural elements. Building a parametrical model of the SoS that accounted for the physics of operating in space and in the air was better handled outside of the architectural tool, so STK\(^25\) by AGI was used to build the integrated model. Finally, since manually creating individual models in STK for simulation using the graphical user interface can be time consuming, Python\(^30\) was used to drive the STK Engine.

This arrangement of tools had numerous benefits, including the ability to quickly build a model (i.e., populate a large constellation of satellites or aircraft) and explore permutations of the model (i.e., change the composition of the constellation of satellites and aircraft). Other benefits included the ability to perform calculations required by the value hierarchy that were not produced directly in the STK simulation environment. Although other programming languages could have been used (e.g., MATLAB), Python was chosen due to its versatility, ubiquity on DoD networks, and low academic costs.

The integration of tools occurred as follows. Python was used to build text-based “Connect” commands that are sent to STK to set up the simulated scenario, the systems, their orbits, and their sensor characteristics. A Transmission Control Protocol/Internet Protocol (TCP/IP) connection was used to send Connect commands to STK from its own library whose purpose it is to enable (Python) scripting. The benefit of using Connect is that creating scenarios can now be automated and performed much faster. The value hierarchy is coded in Python and uses the results of the STK simulation, which computes the access between each satellite or aircraft and the target, as a text file. Python then reads this file containing the azimuth-elevation range between the sensor and target for the entire scenario in 1 s intervals.

3.6. Assess value and provide recommendation

Once the STK simulation has completed, performance measures generated by the simulation were fed back into the value hierarchy in order to produce a time-variant value graph that showed how the value of the architectural solution changed with respect to time. Once created, this graph was analyzed to determine whether a specific architectural solution provided the right value at the right time.
If the architecture did not perform as expected, then the value analysis was used to modify the current architectural solution or to generate a new solution.

4. Results and analysis

In order to demonstrate the method, three architectural solutions (i.e., Baseline, Alternative 1, and Alternative 2) were built and evaluated using the value hierarchy and operational concept provided by the research sponsor. Table 1 summarizes the three alternatives and Figure 3 provides the value hierarchy used. All three alternatives include eight planes of five satellites in low-earth orbit (LEO).

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Alternative 1</th>
<th>Alternative 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total # of Sats</td>
<td>40</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td># of Planes</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td># of Sats per Plane</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Orbit</td>
<td>LEO</td>
<td>LEO</td>
<td>LEO</td>
</tr>
<tr>
<td>Sat ID Sensor</td>
<td>Pointing / Day Only</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>UAV ID Sensor</td>
<td>N/A</td>
<td>N/A</td>
<td>Pointing / Day Only</td>
</tr>
</tbody>
</table>

UAV: Unmanned Air Vehicle; LEO: low-earth orbit.

Figure 3. The value hierarchy (which assumes all weights are equal within the Collect Against Target objective). Performance measures generated by the Systems Tool Kit, or calculated in Python, are represented by ovals. NIIRS: National Imagery Interpretation Rating Scale.

If the architecture did not perform as expected, then the value analysis was used to modify the current architectural solution or to generate a new solution.

Detect and ID Sensor capabilities vary by alternative and only Alternative 2 includes an Unmanned Air Vehicle (UAV). For all three alternatives two assumptions were made: (1) communications were perfect and instantaneous (except for a constant 20 min delay for sensor data to transmit from the sensor platform to the ground station); (2) ID Sensors always point toward the assigned ground target. Thus, assessments can be considered optimistic. The UAV flies an ingress route into the general target area, then egresses back to the launch site.

All three architectural alternatives were modeled in STK using Python. Performance data generated by the simulation, in 1-s time steps, was fed back into the value hierarchy.
hierarchy to obtain a near-continuous, time-variant value scoring of the alternative’s performance. The time-variant value was assessed because all performance measures in the value hierarchy were functions of time.

Note that the performance measures feeding the value hierarchy include National Imagery Interpretation Rating Scale (NIIRS) image quality, % Coverage, and System Response Time. NIIRS is a standard approach to quantify the interpretability and usefulness of imagery. The visible NIIRS rating scale consists of 10 levels of image quality from 0 to 9, with 9 being the most demanding and requiring the highest image quality. The unique advantage of the NIIRS is that it can be used as both a system specification and an operational requirement, bridging the gap between developmental and operational communities. We use NIIRS definitions for detection and identification as follows.

**Detection:** Detect is the capability to find or discover the presence or existence of an installation, object, activity, or item of interest, based on its general shape (configuration) and on other contextual information in the scene. Some level of identification is implied in detection, so that the feature detected can be properly named.

**Identification:** Identify is the capability to name an object by type or class, based primarily on its configuration and detailed components. Identification depends on observation of detail in the image and not on information from non-imagery sources.

So, to *detect* a target, the image must be clear enough to be properly named and to *identify* a target, a higher level of detail is required in which one can identify a specific configuration.

The value contributed by each performance measure was calculated using a Single Attribute Value Function (SAVF) with STK/Python performance values as inputs. Figure 4 shows the NIIRS Identification SAVF. The shape of the function indicates no value is assigned until a NIIRS level of 5 is achieved (i.e., no identification is possible unless an image quality of NIIRS 5 is obtained). The value increases linearly until the maximum defined level of NIIRS 9 is reached. Individual SAVFs are used for Coverage Detection and Coverage Identification. In order to simplify the simulation, System Response Time was assumed to be constant.

Some assumptions were made in calculating value. For example, when multiple sensors (Satellite or UAV) were within line-of-sight of the target, the highest value of the NIIRS was used.

Figure 5 show the Total Value versus Time plots for each of the three architectural alternatives. All three alternatives exhibit value periodicity caused by the satellite orbital period in which total value varies from zero to a peak maximum. The Baseline Alternative (Figure 5) shows a relatively consistent value over time, but with expected degradation during the nighttime hours when the pointing ID Sensor does not function. Alternative 1 (Figure 6) yields a consistent value, but with lower peaks because there was no ID Sensor included. Alternative 2 (Figure 7) resembles Alternative 1, but has a brief high-value spike when the UAV is within view of the ground target early in the scenario. While the three alternatives are rather simple, generally this overall alternative assessment, with multiple weighted factors, each changing over time, is challenging.

The application of various aggregated value assessments over time (minimal value, average value, peak maximum, etc.) is left for future research.

While decision makers are concerned with the total time-variant value of an alternative, it can also be useful
to investigate the contributions of individual performance measures on overall value (Figures 8–10). The largest value contribution is attributed to the satellites’ persistent Detection Coverage of the target, with the smallest contribution being the satellites % Coverage of the ID Sensor (due to the ID Sensor’s narrow field-of-view and the satellite’s short time over target).

Using Python, time-dependent plots were created for NIIRS and % Coverage, which capture moments of both collection and dead time. Figures 11–13 show plots of Maximum NIIRS versus Time for all three alternatives. Each instantaneous NIIRS value is attributable to either the Detect or ID Sensor, whichever yields a higher NIIRS value.

Figure 2 (Baseline Alternative) clearly shows a drop in maximum NIIRS due to nighttime operations. Although each satellite’s Detection Sensor is operational at night (maximum NIIRS of 3.5), ID Sensors (maximum NIIRS of 5.7) only operate during daytime. When the value attributable to the ID Sensor is calculated, for example, the value is only assigned when NIIRS > 5 (Figure 4). The dashed line in Figure 2 shows that, for the majority of the time,
the Detect and ID Sensors provide no value. Alternative 1 (Figure 12) never provides the minimum required NIIRS of 5 (because no ID Sensor is employed), but Alternative 2 (Figure 13) shows a very high value of NIIRS (nearly 9) during the short time the UAV is over the ground target (approximately 1100 UTC). The numerous tightly grouped peaks on all three plots show the periodicity of the satellite fly-bys in which the ground target is in view for only approximately 10 minutes. The UAV remains in sight of the ground target for approximately 18 minutes.

The shape of the NIIRS curves are not perfectly square because the NIIRS calculation relies upon altitude and distance to the target, which is a function of look angle, or angular offset from the target. Figure 14 graphically portrays the difference in the NIIRS curve from the satellite to the UAV. The UAV provides a higher NIIRS than the satellite (because it flies closer to the ground target) as well as a longer dwell time over the target (the satellite’s dwell time is limited by its orbital geometry). The non-zero NIIRS calculations for each UAV’s ID Sensor ranged from 6.9 to 9 as the UAV approached the ground target, stayed at 9 while directly overhead, and then fell back down to about 6.9 as the UAV departed out-of-view. Since aperture diameter and altitude were held constant, the differing NIIRS values were attributed to elevation angle changes and range adjustments to the ground target. However, the overall value of an alternative combines NIIRS with % Coverage, so the target must be found across the covered area, then identified.

Analysis of the Total Value versus Time plots (Figures 5–7) and the individual Performance Measure

---

**Figure 11.** National Imagery Interpretation Rating Scale (NIIRS) versus Time for the Baseline Alternative, showing degradation due to nighttime operations. SAVF: Single Attribute Value Function; UTC: coordinated universal time.

**Figure 12.** National Imagery Interpretation Rating Scale (NIIRS) versus Time for Alternative 1. SAVF: Single Attribute Value Function; UTC: coordinated universal time.

**Figure 13.** National Imagery Interpretation Rating Scale (NIIRS) versus Time for Alternative 2. SAVF: Single Attribute Value Function; UTC: coordinated universal time; UAV: Unmanned Air Vehicle.

**Figure 14.** Comparison of National Imagery Interpretation Rating Scale (NIIRS) from Unmanned Air Vehicle (UAV) versus Satellite. The plot shows that not only does the UAV provide a greater NIIRS than the satellite (heights of the curves), but also longer dwell time over the target (width of the plots). UTC: coordinated universal time.
Value versus Time plots (Figures 11–13), and/or the value contribution Time plots (Figures 8–10) all highlight the architectural trade space. For example, the multi-domain aspect of Alternative 2 highlights the fact that high-quality NIIRS imagery for target identification can be obtained at the exact time it is required while nearly constant target detection coverage is maintained. It would be an expensive and complex undertaking to obtain constant high value from an architectural alternative, but exploring the Value versus Time plots allows the operator and the architect to determine when the most value is needed (and when it is not) and design an alternative that optimally meets the value requirements.

5. Conclusions

A method was developed through a synthesis of decision analysis and MBSE with the goal of assessing the military value of future multi-domain conceptual architectures. The method was implemented by using a value hierarchy to developing architectural alternatives in support of an operational concept. Each alternative was modeled, then simulated to generate performance data. This data was fed back into the value hierarchy to generate value versus time plots. These plots allowed each architectural alternative to be analyzed, in the time domain, to determine if it provides the right amount of value at the right time. The plots of value with respect to time allow comparison of architectural alternatives from a common perspective (the operational concept and the value hierarchy) and form the basis for iteratively improving an architectural concept.

This research will be extended by investigating additional methods of analyzing the time-variant value data. For example, new information about an alternative, and how to improve it or new ways of visualizing the complexity of an alternative could be found by integrating or transforming the time-domain information. A spectral analysis, in particular, seems promising as a way of visualizing the commonality or rarity of value-producing events. Other methods of architecture and model development may be more efficient than that used in this research. Finally, software tools other than the ones used in this research may be more capable of integrating architecture models, physics-based models, and value hierarchies. Lastly, we hypothesize there are many problems within many domains that would benefit from an analysis of the time-variant value.

Acknowledgments

The authors would like to thank Don Miceli from the Air Force Life Cycle Management Center for his guidance and sponsorship of this research. The authors are indebted to the extended research team of Stuart Everson, Ralph Korthauer, Cameron Cunningham, Bola Sobande, and Ryan Samuelsen who contributed to this research. Finally, the authors extend their gratitude to the editor and reviewers who provided constructive feedback on the paper’s layout.

Declaration of conflicting interest

The authors declare that there is no conflict of interest.

Funding

This work was supported by the Air Force Life Cycle Management Center office of Development Planning (grant number AFLCMC/XZ).

Disclaimer

The views expressed in this paper are those of the authors and do not reflect the official policy or position of the United States Air Force, the DoD, or the US Government.

6. References


Author biographies

Thomas C Ford is a senior systems engineer with Integrity Applications Incorporated, Dayton, OH. He recently retired with 22 years of engineering and management experience from the US Air Force. He can be contacted via email at tcfordphd@gmail.com.

David W Meyer is a research analyst and member of the Center for Space Research and Assurance (CSRA) at the Air Force Institute of Technology (AFIT), Wright-Patterson AFB, Ohio, USA. His research interests include modeling and simulation, optimization, data analytics, systems engineering, and military strategy.

John M Colombi is an Associate Professor and Program Chair of systems engineering with the AFIT. He publishes extensively on systems architecture and engineering, modeling and simulation, acquisition, and human systems integration.

Brian K Scheller graduated with a Master of Science in the Systems Engineering from the AFIT in March 2016. He is a Captain in the US Air Force currently assigned to Space and Missile Center, Los Angeles AFB.

Cody G Palmer graduated with a Master of Science in Systems Engineering from the AFIT in March 2016. He is a Lieutenant in the US Air Force currently assigned to a Space and Missile Center Detachment at Kirtland AFB NM.