SYSTEMS ENGINEERING
Research Center

Engineered Resilient Systems: Tradespace Tools Research

Technical Report SERC-2016-TR-108

August 26, 2016

Principal Investigator: Dr. Tommer R. Ender, Georgia Tech Research Institute

Research Team:

Bryan Ake
Dr. Santiago Balestrini-Robinson (*author*)
Daniel Browne
Dr. Dane Freeman (*author*)
Thomas Mark
L. Drew Pihera
Victor Roetman
Dr. Valerie Sitterle (*author*)
James Arruda (*author*)
Erika Brimhall
Jordan Carroll
Robert Kempf (*author*)
Dr. Frank Patterson (*author*)
Jason Poovey
Melissa Rost

Sponsor: Engineer Research and Development Center, U.S. Army Corps of Engineers

Contract Number: HQ0034-13-D-004
Report No. SERC-2016-TR-108

Task order 0045, RT 145

The Systems Engineering Research Center (SERC) is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology.

This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Office of the Assistant Secretary of Defense for Research and Engineering (ASD(R&E)) under Contract HQ0034-13-D-0004 (Task Order 0045).

LIST OF TABLES

TABLE OF FIGURES

This page intentionally left blank

1 INTRODUCTION

1.1 EXECUTIVE SUMMARY

The Department of Defense's Community of Interest for Engineered Resilient Systems (ERS), led by the US Army Engineer Research and Development Center (ERDC), calls for systems that are effective over their life cycle, even when the mission context changes beyond its initial intention. Towards this end, tradespace analysis is of great importance, which enables adaptable designs using diverse systems models that can easily be modified and re-used, and the ability to iterate those designs quickly with a clear linkage to evolving mission needs. The Georgia Tech Research Institute (GTRI) is co-developing a web-based, collaborative tradespace environment along with ERDC for the ERS Community of Interest. This leverages GTRI's expertise in collaborative, executable Model-Based Systems Engineering (MBSE), and ERDC's leadership of the DoD High Performance Computer (HPC) infrastructure and operations research expertise.

Tradespace analysis facilitates and enables the vision and goals of the ERS program. It is a proven operations research method used to assess design trades. Coupled with data generated from engineering models, operational simulations, and other authoritative sources, tradespace analysis can be used throughout a system's lifecycle, particularly within the requirements generation phase, to expand the number of feasible alternatives analyzed. To facilitate this tradespace analysis, a collaborative environment is required to allow researchers to input multiple variables with linear and non-linear relationships to investigate viable trades and view second, third, and higher order effects of changing parameters. ERS therefore aims to create a comprehensive tradespace analytics capability that supports complex DoD systems under a wide range of operation scenarios. This effort produced a collaborative, modular open architecture software framework, which allows users to conduct trade studies leveraging executable MBSE integrated with HPC assets. This enables communication of complex results to stakeholders in order to support effective decision processes.

This Final Report will walk through the systems engineering processes applied, as well as demonstrate a notional "design-execute-explore" workflow. During the "design" step, a user would identify capability needs and/or gaps, and derive requirements; a decision maker may apply utility functions to those needs to support analyzing alternatives during the "explore" step. This "define" step includes the definition of the system itself, to include a physical decomposition (which increases in detail with further iteration). Finally, the design step includes the characterization of any modeling and/or analysis required to assess the system(s) of interest. This report will demonstrate how the Systems Modeling Language was used to achieve these goals via a collaborative web-authoring tool. The "execute" step includes the process of selecting the attributes to be varied, and their bounds. This includes the ability to identify sampling methods, to include various forms of Monte Carlo simulation and Designs of Experiments. Further, the user has the ability whether to select to locally or remotely hosted modeling and simulation assets, such as those run via HPC. Finally, the "explore" step enables a user to develop a custom dashboard by selecting visualizations of interest, to include classic descriptive analytics such as interactive histograms and 4-D bubble plots, as well as more advanced techniques to enable rapid Analyses of Alternatives, based on Multi-Objective Decision Analysis leveraging utility functions identified in an earlier step.

1.2 SUMMARY OF PRIOR WORK

Under SERC RT-120 (April 2014 through June 2015), GTRI development of decision support methods and a tradespace toolset framework architecture in support of ERS[1]. This included conducting research and development of methodologies such as scenario based Needs Context based Utility Functions, and risk mitigation of uncertain future events through option buy-ins. Next, this effort investigated various toolset usability upgrades, building on GTRI's experience in building web-based, collaborative systems engineering frameworks, with updated inputs from stakeholder elicited requirements. Specific goals achieved for this effort included:

- Developing utility functions to score alternatives that captured components of a resiliency metric for a system design. This effort covered elements of resilience across competing or changing requirements on its performance attributes
- Developing an initial method for system assessment to select components or alternatives given uncertainty in future needs
- Redesigning and codifying a tradespace tool which can integrate into the ERS architecture. This tool is usable by variety of systems, e.g., ships, aircraft, and ground vehicles, and released in April 2015 as **ERS TRADESPACE v1.0**
- Extending the work conducted under a previous effort to develop an API between related collaborative tradespace tools and the Army Research Laboratory's EASE framework to increase the number of attributes passed[2]
- Working with the Naval Surface Warfare Center – Carderock Division (NSWC-CD) to integrate models and data/information developed at NSWC-CD resulting in an updated ship model characterization, and updated modeling tools to study the ship example used by the Navy to help assess tens of thousands of ship variants.

Figure 1 shows a snapshot of the home screen of the ERS TRADESPACE v1.0 toolset developed under this prior SERC RT-120 work. Note that "Define, Execute, Analyze" are the major entry points in to the ERS TRADESPACE toolset suite and are immediately visible from the landing page. Intuitively, these capture the steps of defining a user's system of interest, executing a defined set of analyses for that system, and then analyzing the results. This provides a user interface for analysts to quickly and accurately assess and compare alternatives to execute materiel solutions analysis.

ERS TRADESPACE consists of a core, containing the data structure, a layer containing the analytical process blocks, and a layer through which analyses are ordered, managed, and executed. OpenMDAO, an open source tool developed by NASA, is a key element of this latter layer, and it is here that a tradespace is generated and interaction with externally hosted models is controlled. These models are reachable via tools external to ERS TRADESPACE and exist as some synthesis of a server, a library of various analytical models and simulations, and an interface through which external software may submit inquiries. The SERC RT-120 Final Report provides, in great detail, a description of theoretical underpinnings associated with capturing the "resiliency" concept, as well as an "under the hood" walkthrough of the ERS TRADESPACE v1.0 software. This RT-120 Final Report included a

[1] The Final Report for SERC RT-120 can be accessed at: http://www.sercuarc.org/wp-content/uploads/2014/05/RT-120_Technical-Report_ERS-Tradespace-Tools.pdf

[2] The Executable Architecting Systems Engineering (EASE) framework is an Army Research Laboratory framework that provides an integration point for multiple large scale operational simulations.

recommendations section with proposed improvements to the tradespace software toolset, many of which were realized under the RT-145 effort described in this document.

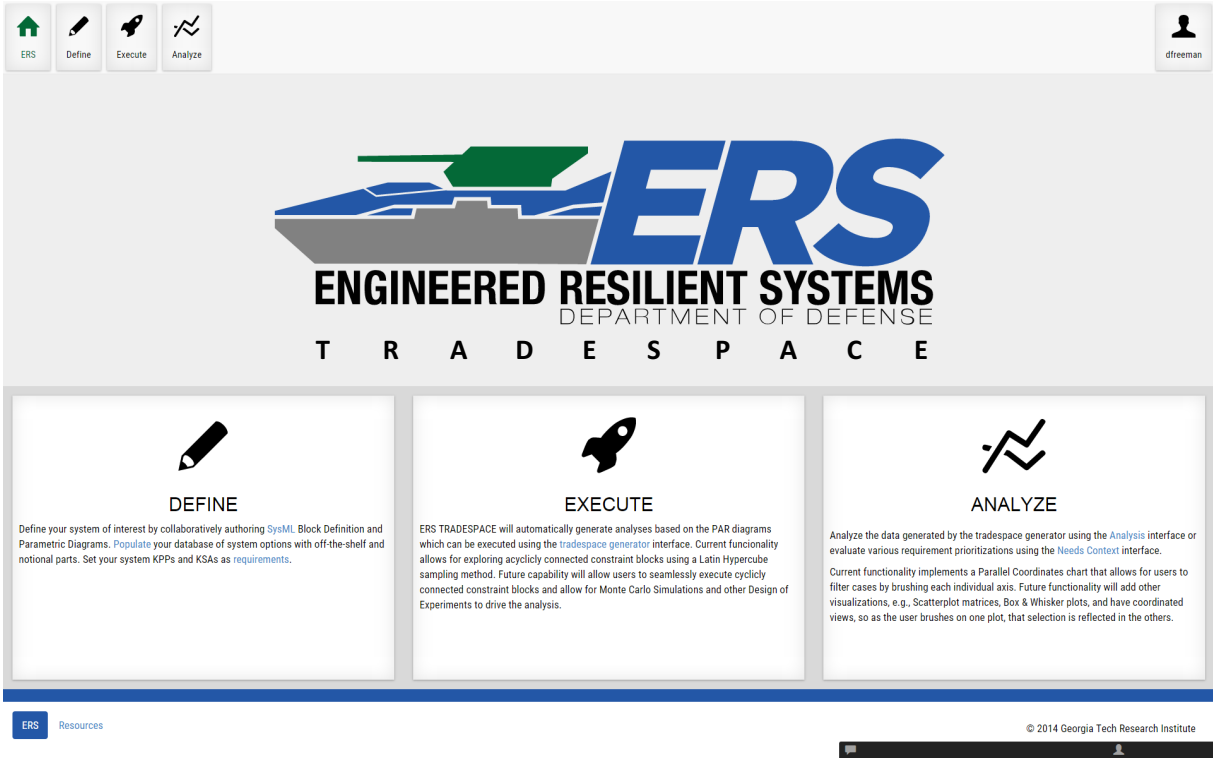


Figure 1. The ERS TRADESPACE v1.0 toolset developed under the prior SERC RT-120 effort

Note that the effort described in this report includes updates to the ERS TRADESPACE v1.0 software, released under RT-145 as version 1.4.1 and rebranded by ERDC as the ***ERS TradeBuilder***. A detailed walkthrough of TradeBuilder is provided later in this document in Section 3.

1.3 SUMMARY OF DOCUMENT

This document is divided into the following sections:

- **Section 2** provides a description of the analytical underpinnings of the enhanced resiliency framework developed as part of this research. This includes a thorough description of what "resiliency" means to the systems engineering process, and the related impacts and value of tradespace analysis. This breaks down to four primary areas researched for this effort, to include uncertainty, disparate Concepts of Operations, dealing with product families, and systems of systems/portfolio assessments.
- **Section 3** walks the reader through ERS TradeBuilder, the tradespace analysis software developed for this effort by GTRI. This includes a detailed description of the plug-in, modular architecture approach for software development, and the Application Protocol Interface (API) one would use to extend the capabilities of this toolset. A special focus is given to the model and simulation execution engine approach developed. Note that TradeBuilder is the version 1.4.1 release of what was referred to as ERS TRADESPACE under the prior SERC-RT-120 effort. Additionally, this section describes the modules which GTRI developed for ERS TradeAnalyzer, the large-data-set visualization software developed by ERDC.[3]

[3] ***TradeAnalyzer*** and ***TradeBuilder*** are the two primary components of the broader ERS ***TradeStudio*** Suite.

- **Section 4** describes GTRI's effort to align TradeBuilder with ERSTAT, a toolset developed by ERDC to deploy modeling and simulation on High Performance Computing assets. This includes showing how high fidelity modeling and simulation can be directly integrated within the systems engineering and tradespace analysis process.
- **Section 5** provides a summary of several analysis modules developed within TradeBuilder to support systems engineering, to include designs of experiments, optimization, and using the *R* programming language to support statistical analysis.
- **Section 6** describes how ERS TradeBuilder leverages the Systems Modeling Language (SysML) to support system and process definition, to include in-browser collaborative editing of SysML models.
- **Section 7** describes the activities to support Analyses of Alternatives, to include modules developed for both TradeBuilder and TradeAnalyzer.
- **Section 8** describes the interface developed between ERS TradeBuilder and Executable Architecting Systems Engineering (EASE), a simulation framework which links analytical, experimental, and training objectives with the technical complexity of modeling and simulation. EASE, developed by the US Army Research Laboratory, enables exploration of operational aspects of the analytical questions in simulation. The integration with ERS TradeBuilder further enables users to evaluation higher-level Measures of Effectiveness based on scenarios (i.e. the scenario based Needs Context methodology introduced in this work). Note that this interface was developed under a prior effort; this effort extended the interface to account for the updates to TradeBuilder.
- **Section 9** provides information on specific DoD programs supported during this effort, using the TradeBuilder tools. This includes developing a notional fighter aircraft sizing model, a rotorcraft conceptual design model, and a ground vehicle model.
- **Section 10** closes the formal portion of this document with a summary and series of recommendations
- **Section 11** provides a list of references cited throughout this document.
- **Appendix A – Publication & Outreach Outputs for RT-145** provides those publications and outreach activities associated with this effort.

2 ENHANCED RESILIENCY FRAMEWORKS AND METRICS

The analytical goals for the suite of ERS decision support tools are to begin to create the building blocks whereby ERS may guide and tailor resiliency exploration of a tradespace for any future system. Within the framework of ERS, the present analytical development adheres to a specific context of evaluation. Namely, this task seeks to expand and mature how we may evaluate early-stage, Pre-Milestone A designs in terms of their fielded system capabilities, performance, and operational context through tradespace exploration.

These efforts will preserve synergy with the analytical methods already developed for the broader ERS effort as well as the ERS TradeStudio software engineering effort described in Tasks 3 and 7. The supporting software and architecture engineering helps guide these analytical processes effectively and efficiently, orders and preserves analytical executions, and supports visualization and interactive exploration of the tradespace. All efforts under this task will consequently focus on maturing and extending analytical methods in ways that are:

- **Realizable** based on available data (i.e., recognize the potential of limited data availability outside of the tradespace itself)
- **Executable** in quantitative analyses
- **Scalable** to large data sets for complex engineered systems
- **Repeatable** across analyses without major changes to the analytical construct(s) (i.e., ease of use)
- **Directly applicable** to dimensions of resiliency as defined by ERS via building blocks that are consistent with existing ontological bases.

There were four primary foci of this task for this phase of the program:

i. Uncertainty
ii. Disparate operational environments and concepts of operations (CONOPS)
iii. Product family identification and evaluation
iv. Systems of systems and capability portfolio assessment

2.1 UNCERTAINTY

Objectives

There will be many sources of uncertainty inherent in a given tradespace data set, whether associated with data, models, SME rankings, etc. In some cases, we can identify where we can best and most meaningfully apply uncertainty to the analyses. In others, this remains more of a question.

Efforts under this subtask were focused along two different lines. First, we matured and expanded the Needs Context construct from previous work (to include work from GTRI's SERC RT-120 effort) to add more operational insight and filtering of potential designs. Second, we evaluated uncertainty directly, using the Needs Context structure developed previously. The goals were to better understand what types of uncertainty contributed to the final decision making and how they should be used within or across design variable inputs and outputs from disparate models and simulations while maintaining computational feasibility as the dimensionality of the problem increases. Finally, we considered uncertainty at a higher level as it related directly to how deeply uncertain we might be about the contexts in which we are evaluating our system and what approaches could help us move forward in the decision space.

This subtask is strongly related to the efforts surrounding the disparate operational environments and CONOPS (section 2.2), and overlaps will be pointed out and discussed in detail in that section.

Work Description and Accomplishments

Maturing and Expanding the Needs Context.

A significant concern during early phases of acquisition, or during the Pre-Milestone A analysis of the DoD Acquisition process, is the resiliency of a system design across simultaneously competing or sequentially changing requirements on its performance attributes. A Needs Context, defined in previous work (Sitterle, Curry, and Ender, 2014; Sitterle, Freeman, Goerger, and Ender, 2015), is a scalable, applied methodology to capture certain resiliency dimensions related to how well a system performs its functions in the face of requirements perturbations. It builds from robustness as defined by (Ryan, Jacques, and Colombi, 2013) and the concept of Broad Utility advocated by (Goerger, Madni, and Elsinger, 2014), creating a requirements-based evaluation of the non-cost value of system design alternatives. Needs Contexts may be more completely described as characterizing "Robustness of Fielded System Capabilities and Capacity with respect to Operational Requirements". Contexts are defined based on flexible subsets of performance attributes relevant to the stakeholder(s) and ranking of those attributes within each. Succinctly, an individual Needs Context specifies a subset of evaluation measures deemed critical to a stakeholder as the basis for analysis. The motivation is that choices must be made based on what is valued most by stakeholders, recognizing that some stakeholders may have a greater influence. Together, multiple Needs Contexts can be constructed to represent different viewpoints and can represent different or directly competing objectives for a system's performance:

- Different stakeholders, each with different or competing priorities in parallel
- Changes in requirements over time (future performance requirements differ in series)
- Different mission profiles with performance objectives, whether in parallel or in series

Requirements Basis for Value Functions.

Value of a given system attribute is scaled against objective and threshold requirement levels using a KPP concept to promote comparability across analyses. The attribute value functions limit all possible valuations to the range of 0 to 1 by assigning any levels below threshold or above the objective equal to 0 or 1 respectively. A tradespace may or may not cover the entire range. Value of a system design alternative is then assessed using an additive multi-attribute value (MAV) model, synergistic with the concept of evaluating Broad Utility via the robustness descriptor presented above. Since each Needs Context may be defined using different attributes, and/or different valuations and preference weightings, Needs Contexts can produce a different value for each system design alternative k (SD_k) within each individual Needs Context m, i.e.:

$$| \quad U_k = w_i * v_i(Y_{ik}) + w_j * v_j(Y_{jk}) + \ldots w_n * v_n(Y_{nk}) = {}^{n}\Sigma_{i=1}\, w_i * v_i(Y_{ik}) \quad | \; _{\text{Needs Context } m}$$

U_k denotes the overall value of system design alternative k (SD_k) for a given Needs Context, Y_{ik} represents a system attribute i for SD_k, each v_i is a value function expressing the relative value of the given system attribute level to a stakeholder, and w_i are weights derived from preference rankings or other means. In keeping with traditional utility theory, overall system value is limited to the range of 0 to 1. Value functions are typically linear or exponential expressions but may be any monotonic function. Cost is a function of system design alternative characteristics, though it depends on other influences and variables as well. Utility and cost are therefore expressed as related dimensions, linked by an underlying SD_k.

*Limitations of Additive Multi-Attribute Value Models***.**

In the previously cited work, the Needs Context served as the basis from which an analyst could construct an overall valuation for each system design alternative from the perspective of the individual stakeholders. Though using a unique, requirements-based valuation construct, the overall valuation still relied on the commonly used additive multi-attribute value (additive MAV) model, also called the sum additive weight (SAW) method. This

approach is scalable and intuitive, yet it does not adequately represent a more operationally focused perspective. For example, consider a set of five attributes each with equal weights (all $w_i = 0.2$). A system design that exhibits valuations of each attribute to a level of 0.8 (all $v_i = 0.8$) will produce the same measure of overall value, $U_k = 0.8$, as a design alternative where 4 of the 5 attributes meet the objective but one attribute fails to meet threshold ($v_i = [1, 1, 1, 1, 0]$). Similarly, as the number of attributes in the measure increases, the impact of attributes failing to meet threshold decreases. This same effect can be seen with the traditional attribute value scaling to the given tradespace. In an operational environment, a defense system failing to achieve a key requirement threshold is not equally acceptable.

Maturing to an Operational Needs Context with a Penalty Function.

The challenge is to mature the overall valuation measure from a traditional additive MAV to a construct more representative of the operational viewpoint. The Needs Context is already well suited to represent disparate operational scenarios that may exist, and by definition it captures those measures of performance deemed critical from a given operational perspective. However, failure to meet one or more critical requirement thresholds should be either readily apparent or carry some penalty that prevents the alternative from possessing a valuation on the same level as an alternative meeting all thresholds. Since there are other types of analyses that may need data points representing designs that do well in many measures but fail in one or two to persist, we will not force the valuation for these designs to zero.

Considering the operational perspective, we sought to modify the additive MAV model to include an "operational penalty" for alternatives with any one or more individual attribute value functions evaluating to zero. The additive MAV model value was taken as the maximum valuation an alternative could achieve (as it contains no penalty), while the overall valuation even with every attribute failing to meet threshold preserved the understood lower limit of zero. An exponential function of the value was used to generate a penalty effectively equal to the weight of any attribute with a value of zero and no penalty otherwise. A direct comparison of the traditional additive MAV model and the model with operational penalty for a given system design alternative are as follows:

$$U_{+MAV,k} = {}^{n}\Sigma_{i=1}\ w_i * v_i(Y_{ik}) \qquad U_{OpPenalty,k} = U_{+MAV,k} * [\ 1 - {}^{n}\Sigma_{i=1}\ w_i * exp(-\vartheta * v_i(Y_{ik}))\ ]$$

where n represents the number of attributes included in the value model, w_i are the weights of each attribute, and v_i are the values of the attributes for the given design alternative as obtained from the individual attribute value functions as before. The exponential penalty function produces $U_{OpPenalty} = U_{+MAV}$ when all requirements meet or exceed threshold levels, and a penalty effectively equal to the weight of the individual attribute if its level is below threshold such that $v_i = 0$. ϑ is chosen to be sufficiently large as to ensure this outcome for even the smallest feasible value. $\vartheta = 1000$, for example, reduces the exponential term to 4.54E-5 even if an individual valuation $v_i = 0.01$.

Weighting.

Methods used in ERS TradeBuilder and all methods described here are agnostic to how ranks, or even weights, are derived. Weights in an additive MAV model may originate from any number of methods including subject matter expert (SME) opinions, historical priorities, guided stakeholder discussions, and pairwise comparisons. Another approach that may be particularly useful for analysis of early-stage designs is to use surrogate weights based on the attribute rankings. If ranks are inconsequential or unknown across our subset of critical performance measures, equal weights are an appropriate starting point. If the ranks are known and the preference order holds, different weighting methods may better reflect how the ranks are valued, e.g., linearly, exponentially, etc. (Roszkowska, 2013). Among these, rank order centroid (ROC) surrogate weights are one of the most robust options when there is some uncertainty in weights but the rank preferences are clear. ROC weights are computed from the vertices of the simplex where $w_1 \geq w_2 \geq \ldots \geq w_n \geq 0$. Weights are then defined as the coordinates of the centroid for the simplex, found by averaging the coordinates of the defining vertices. This approach assumes that

the information set on the weights is completely specified by the ranks and that no point in the simplex is therefore more likely than another (i.e., weight density is uniformly distributed over the simplex). Consequently, ROC weights are the expected value weights for the respective probability density functions over the feasible weight space (Barron and Barrett, 1996). (This is readily demonstrated using a Monte Carlo simulation.)

Despite their advantages, ROC weights alone do not solve issues of range sensitivity in decision analysis. Weights are usually adjusted from one tradespace analysis to the next because MAV functions are traditionally normalized to the range of the local decision context, the current tradespace. Normalizing this way can produce very different decision outcomes when the tradespace range changes if the weights are not also changed. This is referred to as the "range dependence of weights" or "range sensitivity principle".

Swing weights are frequently used to preserve consistent decision outcomes in the face of changing tradespace ranges even though this was not the original intent. Swing weights were developed to adjust the weighting scheme in the face of restricting a set of alternatives under consideration to a narrow region of preference so that the final ranking of alternatives would trade off of the dimensions of analysis driving that particular set. A classic example of Dr. Greg Parnell is that of car safety. If car safety is highly important and the alternatives under consideration includes a wide range of safety values, then it is appropriate to weight the safety attribute highly to best reflect the stakeholder preferences. If, however, car safety is deemed so important that a later set of alternatives under consideration only includes those that are highly safe, then swing weights allow the alternatives to be best evaluated on the other prioritized attributes according to the stakeholder preferences (Johnson, Parnell, Tani, and Bresnick, 2013). This concept has been widely misinterpreted as a need to apply swing weights whenever the range of the tradespace changes, which is not the basis for this approach at all.

The methods advocated here, including using an external reference for valuation (and namely one compatible with the DoD understanding of requirements), are ***directly compatible with using swing weights*** *as they are intended to be used as a reflection of preference in the set.* ***However****, we strongly advocate against using any method (including a misused swing weight approach) where normalization or weighting shifts simply because the range of the current tradespace instantiation changes.*

Understanding that intuitive perceptions of attribute importance are often independent of the range of the outcomes, we focused instead on how to adjust the value functions that effectively grade the individual attributes within the MAV model. Developing value functions that normalize to a basis external to the local decision context offers two primary advantages. First, it preserves consistency in decision outcomes just as do the previously developed swing weight methods. Second, externally valuing attributes promotes direct comparability from one tradespace to the next while weight-adjusting methods with tradespace-dependent value functions do not. When using a value function basis external to the tradespace, weight-adjusting methods are not necessary. Our approach exploiting the KPP/KSA requirements structure to form value functions not dependent on the current tradespace range also offers a clear analytical link to requirements. The impact of competing or changing requirements can readily be compared through construction of new Operational Needs Contexts. ROC weights, as expected values over the feasible weight space, are now a solid starting point for analyses when more rigorously obtained weight data are not available. When using an additive MAV model and the individual valuations are also expected values of the given attributes, ROC weights produce the expected MAV Broad Utility given the preference order established by the weights.

Requirements Differentiation.

As discussed earlier, requirements may be expressed according to hierarchical type. While the Needs Context and Operational Needs Context valuation models presented above make no distinction between types of requirements, the models are easily amenable to do so. Weights in any MAV are scaling constants and, as such, may be "re-scaled" if necessary to differentiate between levels of priority or value. Returning to the concept of

"must", "should", and "could" have requirements corresponding to KPP, KSA, and OPP/Tier III requirements types respectively, the equations used to assess Broad Utility may be altered via a "requirement weight", β_i. For example, all "must have" requirements could be assigned $\beta_i = 1$, which reduces to the previous version of these value models. Measures of performance classified as "should have" and "could have" requirement types, might be assigned values of 0.8 and 0.6 for β_i respectively. The β_i values for these lower level requirement types are simply examples in accordance with but not directly based on DoD guidelines. The following valuation shows the function form when the requirement weight is only applied to the penalty term if a design fails to meet a requirement threshold:

$$U_{OpPenalty\text{-}\beta,k} = U_{+MAV,k} * [\, 1 - {}^{n}\Sigma_{i=1}\, \beta_i * w_i * exp(- \vartheta * v_i(Y_{ik})) \,]$$

If the β_i term is applied in the traditional $U_{+MAV,k}$ model as well (yielding $U_{+MAV\text{-}\beta,k} = {}^{n}\Sigma_{i=1}\ \beta_i * w_i * v_i(Y_{ik})$ and subsequently a $U_{OpPenalty\text{-}\beta,k}$ model also using this form) , analyses that choose to focus on OPP/Tier III measures will not yield valuations of Broad Utility on par with those based only on critical, KPP type measures even when meeting all requirements. That can be logical in the sense designs focused on meeting OPP/Tier III requirements should not be valued as highly as those meeting KPPs. The key to keeping such analyses meaningful, however, is consistency in application and documenting why that application is warranted.

Directly Evaluating Locations of Uncertainty in the Decision Analysis and their Impact.

We focused on better understanding what types of uncertainty contributed to the final decision making and how they should be used within or across design variable inputs and outputs from disparate models and simulations while maintaining computational feasibility. There is an enormous body of work on uncertainty and how it could apply to various modeling problems. In many of those studies, however, the methods used assume independence between the variables and relatively normal distributions for possible bounds on the variables for the results to be meaningfully interpretable. In engineering systems design, however, neither will be the normal case.

As an openly sharable example, a traditional additive MAV valuation model and the operational penalty valuation model were applied to the Iris dataset, a multivariate data set introduced by (Fisher, 1936) and is also included in the Seaborn Python visualization library (Waskom, 2015). We also used this data set as a shareable example from which to evaluate the impact of uncertainty on different parameters directly impacting the decision analysis. [4]

Approaches investigating uncertainty on attribute weights in additive value models are well defined with many extensions as discussed in (Sitterle, et. al, 2016). They apply well to the $U_{OpPenalty}$ model, but there are interesting ramifications when investigating uncertainty in the value functions comprising the model and the impact of this synthesis with the $U_{OpPenalty}$ construct. First, if normalizing attribute value to the tradespace range, uncertainty must be characterized or propagated prior to normalization. Otherwise, valuations at range extremes can produce values below 0 or above 1. In contrast, distributions associated with uncertainty can be incorporated at any stage in the analysis when using an external value reference; attribute values will always be bounded between 0 and 1.

Interesting dynamics appear for uncertain attributes with levels close to their objective and threshold. As shown in Figure 2, which investigated a uniform distribution of uncertainty on the iris attributes, data (shown as normalized histograms) can be highly skewed near the objective (*sepal_width*) and discontinuous near the threshold (*petal_length*). Pulling value function distributions with these characteristics into a higher-level model

[4] A mode defense-oriented example of the traditional MAV versus operational penalty MAV approach is stepped through in the "Analysis of Alternatives Roadmap" interim report for this effort. (Sitterle, Balestrini-Robinson, Freeman, and Ender, 2016)

such as $U_{OpPenalty}$ necessitates a sampling strategy since they are not readily mathematically convoluted with other distributions. Figure 3 extends these results to the evaluation of Broad Utility as characterized by the U_{+MAV} and $U_{OpPenalty}$ constructs. Figure 3 (a) and (b) show uncertainty only on the weights, evaluated through a Monte Carlo simulation as described by (Lahdelma, Miettinen, and Salminen, 2003). The distributions are understandably narrower when rank order is enforced as shown in (b). Figure 3 (c) and (d) then show the impact of uncertainty on the weights and iris "alternative" valuations when preference ranks are enforced. Both distributions are broader than the comparable case for weights-only uncertainty in (b). The $U_{OpPenalty}$ case in (d) magnifies the effect from (c), resulting in a heavier distribution toward the lower values due to uncertainty of an attribute near to its threshold (*petal_length*).

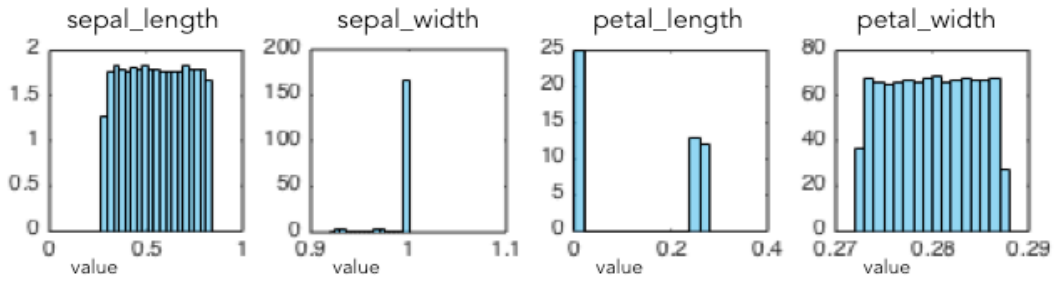


Figure 2. Illustration of Impact of Uncertainty on Value Function Results

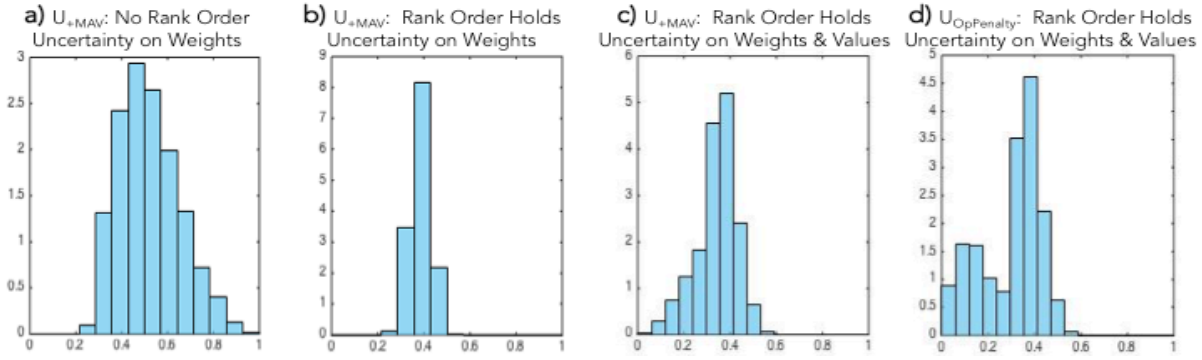


Figure 3. Illustration of Relation between Broad Utility and Uncertainty

This underscores the importance of rigorously investigating design alternatives with uncertain attribute levels near to thresholds and/or objectives, especially if they are classified as being in the "best set" of Pareto designs. Valuation uncertainty can result in highly skewed or discontinuous distributions, and uncertainty on weights and values can produce a bimodal aggregate distribution when an operational penalty is applied. Simply taking a mean, standard deviation, or quartile representation may not represent the uncertainty well. This highlights the utility of cumulative distribution functions in some evaluations when the PDFs exhibit these characteristics. As with any mathematical method, selection and effectiveness of synthesis with other methods are problem and process dependent. Uncertainty analysis with respect to identifying a "best set" of alternatives should focus on what aspects of uncertainty change our decision about which design alternatives are included in that set.

In traditional decision analysis of tradespace data, the context of an evaluation is usually based on the whole of a specific instantiation of a tradespace. Yet an ERS perspective requires the maturation of a more holistic approach toward integrating the whole tradespace analytical view with analyses targeted toward the "best" set of designs. As we will discuss in Section 2.2, focusing some aspects of the analysis on a more tailored, decision-oriented space, a "*Local*" data set, will enable deeper insights regarding sensitivity, uncertainty, and correlations across the

"best" set of designs. Toward this end, the Operational Needs Context can alter which alternatives are in that set via the penalty function. It may be used after a different analytical treatment narrows the alternatives or attributes thereof, or it may be used as a precursor, reducing the "input tradespace" to the next analytical treatment to a more rational set of designs. Which way to take a "best set" to propagate to the next analytical processes depends entirely on the needs of the next treatment and the overall process.

Deep Uncertainty about the Context of our Evaluation.
For our final focus of this subtask, we considered uncertainty at a higher level as it related directly to how deeply uncertain we might be about the contexts in which we are evaluating our system and what approaches could help us move forward in the decision space.

To address Defense contexts, systems engineers will find it necessary to test potential design alternatives against multiple diverse scenarios to understand the resilience of a design's capabilities against uncertain operational needs. This is a distinctly different problem from assigning a distribution or other probabilistic representation to some strongly understood parameter such as variation in machined parts or even environmental temperature variation in a given region. There are many cases where we cannot help being uncertain about our uncertainty. For example, an analyst cannot simply put bounds reflecting high, medium, or low levels of uncertainty on an asymmetric or even generic threat in the course of evaluating potential system performance. The very nature of a threat can and will change, often unpredictably. Similarly, the operational environment may radically alter the realized performance from certain types of sensor systems, and one SoS stakeholder may need to address a completely different type of adversary threat than another SoS stakeholder. In these cases, we cannot simply assume a distribution to carry forward. Not only would that bias our analysis with overt guesses, but it would not at all help systems engineers evaluate whether or not a system design could achieve the desired capabilities under such different conditions. Uncertainty bounds for a given design architecture or operational scenario do not equate to understanding system performance across different scenarios.

At present, we do not have a shared, empirically validated framework for understanding judgment within the AoA process in the face of deep uncertainty, especially when that uncertainty stems from the intersection between the nature of the missions, the nature of the threat environment, the physical environment itself, etc. Instead of selecting design alternatives that are "best" for one or across a small set of weakly defined futures, we need to identify designs that are most robust in the face of many possible but likely futures. We may achieve this robustness from a single design architecture, from a product family consisting of variations on a base design to achieve different capabilities, or from a capabilities portfolio.

Our traditional approaches to this problem follow a set of potential design alternatives across a few defined mission threads or CONOPS-associated stakeholder needs. Sometimes, combinatorial mission threads form various timeline profiles, and any multitude of decision or real options analysis methods help select alternatives with the "best" characteristics. These approaches ask questions such as "Which alternatives provide the best value to stakeholders in key scenarios or across a few possible scenario pathways?" and can offer valuable insights. But, to address this effectively, it helps to not be deeply uncertain about the factors defining those futures. What we need in the future of Defense acquisitions is to answer questions from a different perspective. "Under what circumstances would this design alternative do well?" "Under what circumstances would it fail?" We need to elucidate operational limits for design alternatives to provide the best insight to key decision makers.

High-level concepts of different approaches that can help address these challenges were discussed in two interim reports delivered as part of this effort: "*Analysis of Alternatives Roadmap*" and the "*Cross-scale resilience: Relating Systems of Systems to Individual System Analysis and Back Again*" (being completed as part of the SoS subtask of this effort, section 2.4).

Publications and Links to other Tasks

Concepts and methods in this subtask relate strongly to the subtasks of disparate operational environments (section 2.2) and product family identification (section 2.3) and will be further discussed in those sections.

Concepts and methods in this subtask also relate to the software engineering and architecture effort (Task 3) as well as the AoA tool effort (Task 7).

This work was submitted as a paper on these matured methods and their tie-in to the DoD problem to the INCOSE 2016 International Symposium conference, which was accepted for podium presentation and publication (Sitterle, Brimhall, Freeman, Balestrini-Robinson, Ender, and Georger, 2016).

In tandem with understanding the roadmap for modeling and analysis needs to support the vision of ERS in DoD acquisitions, we described an "*Analysis of Alternatives Roadmap*", which was codified as an interim report (Sitterle, Balestrini-Robinson, Freeman, and Ender, 2016). The document discusses how an AoA, both as an informal supporting tool and a formal mandate, can mature to more effectively address the right requirements questions needed by senior decision makers. Every AoA will be different in terms of nuances, context, available information, etc. for a given system being evaluated at a specific point in the Defense acquisition system process. Consequently, there will be no one-size-fits-all methodology applicable to all AoAs. The roadmap therefore discusses major concepts vital to maturing AoAs to successfully support DoD needs and the goals of ERS relating to resilient design and a resilient design process. It further explains why these aspects are necessary and how they will support materiel development decisions.

The report covering cross-scale resilience will be discussed and referenced in the SoS subtask, Section 2.4.

2.2 DISPARATE OPERATIONAL ENVIRONMENTS/CONOPS

Objectives

A key challenge is to develop a tradespace in a scalable and efficient manner while keeping track of the variations in CONOPS for each engineering design alternative. This task is concerned with developing an efficient means of generating and keeping track of the various partitions of the tradespace that may arise across different models and simulations for the engineering design alternatives. This will aid the identification and evaluation of product families of a given system as compared to a monolithic version of that engineered system. Additionally, this task will be concerned with utilizing the ERS TradeBuilder environment initiated under an earlier effort towards orchestrating and managing the execution.

Work Description and Accomplishments

The efforts in this subtask focused on the relationship between model development, tradespace generation, and decision analyses. Only by creating the appropriate decision space can we begin to explore the robustness, sensitivity, and other nuances associated with systems that must perform across multiple operational environments and mission profiles.

Developing a Model.
One of the first challenges of this task was to find an appropriate example model that could serve as the basis for understanding not only added dimensionality of the analyses but also how to define and propagate the relevant data structures under Task 3 in ways that preserve scalability, consistency, and ease of intuitive analyses across the metadata. Lacking such a model, we developed one for a fairly high-level, notional Joint Light Tactical Vehicle

(JLTV) like platform. Developing a model from scratch helped us understand degrees of granularity and patterns of reusability.

Higher-level M&S components used in early-stage AoA evaluations help identify which concept architectures or regions of design space are the most promising. Some design concepts are different enough that they will require completely different model architectures and computational representations. It is frequently not possible to analyze each distinct design type, as captured in distinct model architectures, in an equivalent fashion. For example, 'miles per gallon' is not a concept that would apply to a purely electric vehicle. A higher-level metric such as cruise range given specified operational characteristics, however, would be comparable. In addition to multiple architectures (design concepts), different operational viewpoints corresponding to different stakeholder needs are critical aspects that drive system evaluation and decision analysis. Different operational environment characteristics will very much impact fielded operational performance. In a computational evaluation, this means that the model(s) of a system alone is insufficient to produce all quantitative system attributes important to the decision making process.

Generating a tradespace from various disparate models and simulations – some representing the system under evaluation and others capturing different operational environment and use characteristics or SoS performance – is not a trivial task if the goals are to achieve flexibility, scalability (often via properly orchestrated modularity), and efficiency of the process. For example, if we delineate operational scenarios into the two parametric classes of operational environment and operational use, we may have an intuitive scaffolding for modelers to specify attributes necessary to generate an operationally specific tradespace. Parts may be reused and expanded upon for creation of new operational scenario blocks. Like most aspects of model development, the attributes or measures the tradespace generation is intended to produce will drive what parameters must be defined in an operational scenario. Any number of operational scenarios may be defined in this way depending on the scope and needs of the analysis.

Scenarios may be interested in the same output measures (e.g., cruise range) but impose different objective or threshold levels or require operation in vastly different environmental profiles that, in turn, alter the measured performance. (This is precisely what the Needs Context discussed in the previous section is designed to address.) Similarly, operational scenarios may be interested in entirely different output measures regardless of operational environment, imposing requirements not present in other scenarios. How to implement these scenario aspects then relates directly to modularity of the parameter specification, the modularity of the evaluation function components (i.e., the model constraints), and hence the degree of reusability of these specifications and model components for additional scenarios and/or tradespace generation.

In the course of using our JLTV development to explore some of these questions, we determined that a purely atomic level of granularity (i.e., each equation is captured as its own block) is not only a painful, time consuming experience, it creates too many opportunities for errors simply from human input. There is no magic answer for what constitutes the appropriate level of granularity; that is really an engineering modeling decision informed by experience and an understanding of the larger problem context. However, the delineation of parametric classes into 'system', 'operational environment', and 'operational use' is intuitive and highly flexible in terms of being reusable and changeable to future needs. We need to mature this understanding in the next phase of the work, specifically how the abstractions will be specified in the ERS TradeBuilder tool. This will directly impact not only how effectively the patterns may be implemented and executed in a computational environment but also how intuitively they are perceived, understood, and employed by an analyst in the development and exploration of a tradespace for an AoA.

Comparison of Model Types.

The notional JLTV models a class of military ground-transport vehicles that are meant to part-replace the Humvee. Inputs to the model include physical dimensions, engine parameters, transmission properties, operating

conditions, notional armament add-ons, crew capacity, and several other related inputs. There are a total of 24 system specific design variables with 3 additional variables pertaining to the operational environment as it will impact the notional JLTV performance in some areas and 7 variables related to how the platform will be used in operations. The model returns attributes such as velocity, acceleration, payload capacity, maneuverability, and transportability. Some attributes are combined into higher-level characterizations. For example, 'gross_maneuverability' is defined by several performance characteristics such as velocity around a curve, acceleration time, etc. In all, it is a simple model where inputs (including all system design variables) are mapped directly to output parameters through simple physics models.

We also had a notional Single Main Rotor (SMR) Helicopter model for comparison, which sizes an SMR vehicle for a given mission specification. An SMR helicopter is the most common kind, such as the UH-60 Blackhawk. Using inputs that define the rotor system shape and efficiencies, engine performance parameters, crew requirements, and mission ranges, payload, and hover times the model returns the vehicle weight breakdown, additional rotor parameters, engine power, and cost. The SMR model differs from the notional JLTV model in a key area, which is that design requirements are inputs to the model. This is due to the nature of the sizing process of aerospace vehicles, which requires a mission definition in order to define an appropriate vehicle for that mission. While it is possible to define a vehicle without a sizing mission and then analyze what missions it is capable of, using both model inputs and outputs as attributes that can be values supplies this effort with more varied examples of tradespace data. The SMR model produces a tradespace where data is highly correlated, making some analyses more difficult to interpret as we shall see in Section 2.3.

Generating a Tradespace.
Once models are defined and all inputs and their ranges are specified, we need to generate a tradespace. This occurs when we use some approach to sample the inputs across their ranges, creating unique combinations across the multidimensional descriptions that feed through the M&S components to produce associated output (attribute) vectors. The prevailing paradigm at present is to generate thousands or millions of design alternatives, enumerated by their unique design variable vectors and stored along with their output characteristics and performance attributes in a database (i.e., the tradespace), and then perform multi-objective or other decision analysis founded on expressions of stakeholder preferences to determine the "best" set of candidate designs.

The reason the tradespace generation is so paramount to this problem is that is forms the foundation on which decision analysis – typically multi-objective – is performed. Preferred criteria are specified, valued, and weighted according to stakeholder priorities. A typical output from this process is a multi-attribute additive value (MAV) measure which may be evaluated as a function of another MAV or, as is more common, cost associated with each design alternative. The entire tradespace is transformed to a functional view of MAV, and the "best" candidate designs are identified based on their Pareto efficiency across the space. When the notion of Pareto efficiency is applied to the selection of alternatives, each option is first assessed under multiple criteria and then a subset of options is identified with the property that no other option can categorically outperform any of its members. A design alternative is considered to be on the Pareto frontier if no other point is more preferred. Dominated alternatives are those for which another candidate exists that is equal or better in all attributes contributing to the measure and strictly better in at least one. A Pareto efficient criteria for selection of that set rests upon the foundational assumptions of monotonicity and independence in the valuation for each attribute that contributes to the MAV functional relationship. The Pareto Frontier is a set produced by these integrated tradespace-decision analysis processes that is seen to be the "best" set of alternatives. Further analyses are often performed to determine the effects of uncertainty on the concept of whether an alternative is dominated across the range of its uncertainty distribution, and therefore might affect the overall decision on its inclusion in a "best" set.

The Pareto Frontier is critical to the decision making process and its conclusions. How the tradespace is generated and, specifically how well points are generated from the design variables that will occupy the Pareto Frontier, can consequently create significant differences in any *a posteriori* analysis.

A very common means of sampling the input design variable levels to produce a tradespace is Latin Hypercube Sampling (LHS). LHS is a type of stratified sampling that controls the way that random samples are generated for a probability distribution. The method controls the sampling of each distribution separately to provide even coverage for each distribution individually, but does not control the sampling of combinations of distributions. When sampling a function of *N* variables, the range of each variable is divided into *M* equally probable intervals. *M* sample points are then placed to satisfy the Latin hypercube requirements, and the number of divisions (*M*) are equal for each variable.

Evolutionary algorithms (EAs) represent a completely different approach to sampling across design variables to generate a tradespace. The objective of these algorithms is to improve the adaptive fit of a population of candidate solutions to a Pareto frontier constrained by a set of objective functions thereby finding as many Pareto-optimal solutions as possible. Since EAs work with a population of solutions, a simple EA can be extended to maintain a diverse set of solutions and find multiple Pareto-optimal solutions in a single simulation run. An example is the Non-dominated Sorting Genetic Algorithm II (NSGA-II) algorithm. The algorithm uses an evolutionary process with surrogates for evolutionary operators including selection, genetic crossover, and genetic mutation. The population is sorted into a hierarchy of sub-populations based on the ordering of Pareto dominance. Similarity between members of each sub-group is evaluated on the Pareto front, and the resulting groups and similarity measures are used to promote a diverse front of non-dominated solutions.

A key difference is that Designs of Experiments (DoE), including LHS, require no a priori understanding of preferences to sample the design variables and thereby create a tradespace. In contrast, EAs, including the NSGA-II algorithm, must either assume or be based on a priori expression of preferences and constraints. The algorithm – and this applies to all multi-objective optimization algorithms – first requires a definition of a decision hierarchy and preferences. In this work, the multi-objective decision construct is captured as a Needs Context. Using a Needs Context to determine the MAV of an alternative along with the cost, multi-objective optimization searches for a Pareto Frontier that trades between the MAV and cost. The search results in a set of alternatives that are on or near the solved Pareto frontier. The alternatives do not span a hypercube like they do for traditional tradespace methods like design of experiments. Instead the candidate designs will populate inside a hypervolume that surrounds the Pareto Frontier and whatever other alternatives were stored by the algorithm. This hypervolume can be of any shape and size.

Algorithmic generation of Pareto Frontiers results in a smaller tradespace than is found using the DoE methods and, because it is based on the objective preferences, is suitable only for that Needs Context as defined. It will not be suitable for other Needs Contexts. For example, a LHS generated tradespace will typically have a surrogate model fit to it for explorations of design alternatives. An NSGA-II tradespace can still have a model fit to it, but since the tradespace spans a smaller hypervolume any exploration of design alternatives outside the hypervolume constitutes extrapolation and therefore retains no guarantees of goodness of fit or other standard measures. So, while a LHS approach creates an entire tradespace, including those designs that are not "good", the EA methods focus the sampling effort on creating a tradespace that is richer near to the Pareto Frontier.

When a tradespace is generated from a space-filling design, such as a Latin Hypercube, the data may change enough that the analysis of the Pareto Frontier changes significantly when compared to tradespace data that is algorithmically generated such as with NSGA-II. The number of points can change, resulting in fewer alternatives in the Pareto Frontier, and the location of those points in the decision space can change. We will demonstrate the significance of these dynamics in the following sections. The take away, however, is that analysts should not assume that a tradespace (and hence its set defined by the Pareto Frontier of some decision space) is (a) necessarily the true frontier possible from the design space, or (b) an unambiguous and universal representation of drivers across the decision space. This provides a strong motivation for combining algorithms with high performance computing such that high quality data is available to the analyst. In some instances, tradespace data

may be all that is available with no recourse to adapt and improve the sampling. In these cases, the analyst must take care about conclusions regarding candidate designs, understanding that the tradespace frontier may not be the "true" frontier.

Decision Analysis and a Soft Pareto Frontier.
Formal methods such as multi-objective decision analysis (MODA) provide the scaffolding for a traceable, justifiable basis for reducing a set of options for further analysis. There is a wealth of information on various decision analysis techniques, their application, and example problems in the literature. One resource that is particularly relevant to the DoD AoA process is the *Handbook of Decision Analysis* (Parnell, Bresnick, Tani, and Johnson, 2013). This reference includes technical and soft aspects of decision analysis, qualitative and quantitative techniques, and incorporates these approaches into case studies by way of example.

In traditional decision analysis of tradespace data, the context of an evaluation is usually based on the whole of a specific instantiation of a tradespace. Yet an ERS perspective requires the maturation of a more holistic approach toward integrating the whole tradespace analytical view with analyses targeted toward the "best" set of designs. For small sets of system design alternatives, it may be quite feasible to evaluate various measures of performance feeding resiliency measures (in turn a higher-level MOP or MOE) for all design alternatives. As the number of design alternatives increases, say to 10,000 or 1,000,000 or more, the global versus local treatment becomes more complicated. Some sensitivity analyses may need to be global, performed across the entire tradespace to reduce a set of parameters or attributes under evaluation. In other analytical treatments, a global approach can compromise insights that may be specific to the "better" design alternatives with bias from the "poor" alternatives as well as inhibit the ability to scale both computationally and visually. (This is especially true for component-based tradespaces.) Focusing some aspects of the analysis on a more tailored, decision-oriented space, a "*local*" data set, will enable deeper insights regarding sensitivity, uncertainty, and correlations across the "best" set of designs.

Specifically, instead of evaluating only a Pareto frontier of design alternatives or an entire tradespace, we identify a 'soft Pareto set' or 'soft Pareto frontier' (sPF) of alternatives. This set serves as the basis for additional explorations and analyses that can add great insight to the decision making process. There are some methods, e.g., variance-based measures that may be pointless when applied to a Latin hypercube generated tradespace. But, when the design alternatives are "filtered" according to a view of MAV, a set including and off of the Pareto Frontier will have certain characteristics (e.g., covariance) that are meaningful.

A tradespace of alternatives can come from many data sources of vary quality and quantity. These include test data, designs of experiments, and a wide variety of optimization and search algorithms. Therefore, any analysis aimed at understanding drivers within the decision space (i.e., the MAV context) should be flexible enough to accommodate the available data. Any statistical test, regression, or other mathematical exploration of a dataset benefits from having more data available to it. Additionally, no model or data is completely accurate, meaning that any points on a Pareto Frontier are subject to uncertainty. Therefore we want to include, as much data as possible, but the amount of data available on a strict Pareto Frontier may not provide enough information to determine Pareto Frontier tradeoffs.

We propose using an sPF, which is created by allowing a defined level of dominated points to enter the Pareto Frontier. This can be done by either specifying the number of successively non-dominated frontier by removing the non-dominated points and repeating the Pareto Sorting, or by altering the standard domination definition to include a region where points are still considered non-dominated. Recalling the definition of a Pareto Frontier, P, of a set of points, Y:

$$P(Y) = \{y \in Y \mid \{y' \in Y : y' \succ y, y' \neq y\} = \emptyset\}$$

The dominance relation over the p-dimensional Pareto Frontier is originally strict (assuming minimization):

$$y' \succ y \;\Leftrightarrow y_i' \leq y_i \; \forall i \in \{1, \ldots, p\}$$

The soft frontier dominance relation is relaxed by some margin in each dimension, ε:

$$y' \succsim y \;\Leftrightarrow y_i' \leq y_i - \epsilon_i \; \forall i \in \{1, \ldots, p\}$$

A point is then only said to softly dominate (symbolized as a weak preference) if it is at least better by ε than all other points in the data in each output dimension. An alternative to the soft dominance is to progressively solve for the strict frontier, remove it from the data, and repeat. Then one can select *n* of the first frontiers to gather enough data. In testing, *n = 5* was a good starting point. The advantage to using the soft dominance relation is that the degree of softness is defined in terms of the outputs and is constant for any MAV as a function of cost of a tradespace. Using a number of frontiers can result in larger or smaller distances from the strict frontier between different MAV functions. A comparison of a strict Pareto Frontier versus a sPF is shown in Figure 4 for a notional SMR Helicopter model tradespace that was generated by a Latin hypercube design and an NSGA-II algorithm. The sPFs retain the same shape as their strict frontier but with the benefit of more data.

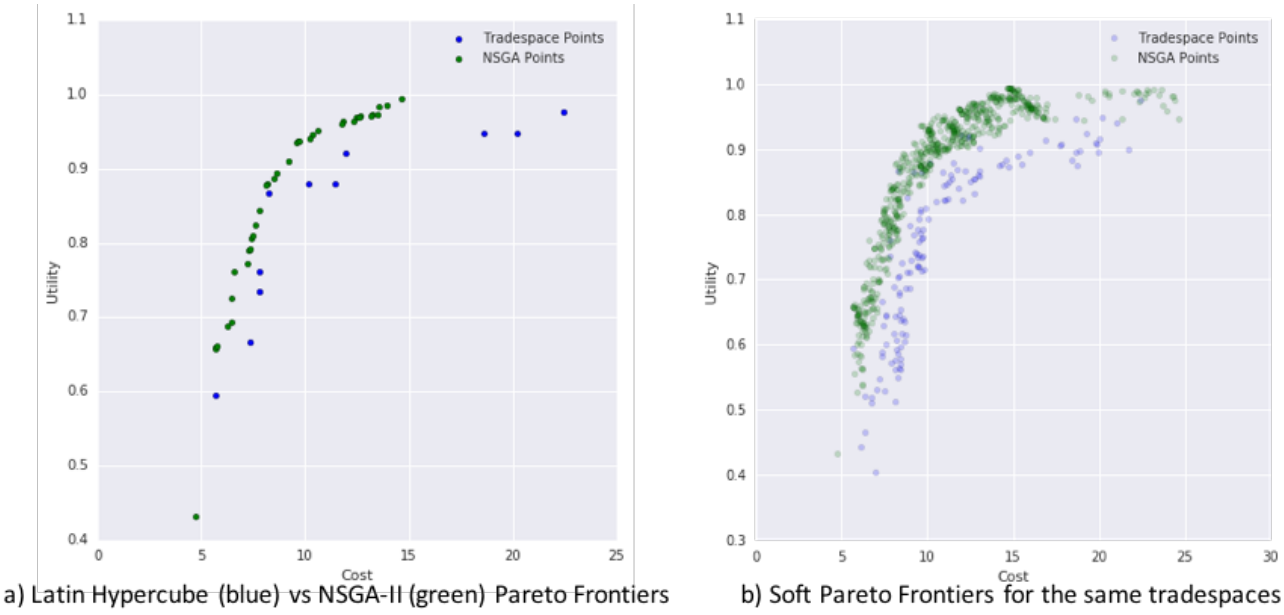


Figure 4. Illustration of a Pareto Frontier compared to a Soft Pareto Frontier for Latin hypercube and NSGA-II generated tradespace data

Importantly, note also that the algorithmically generated frontiers dominate the Latin hypercube generated frontiers. This underscores the discussion in the previous section regarding the need to mature how we generate tradespace data that will serve as the foundation for additional design-to-decision exploration, analyses, and creation of new insights.

The Operational Needs Context described under the uncertainty subtask directly relates to these concepts since its application can alter which alternatives are in that set via the penalty function. It may be used after a different analytical treatment narrows the alternatives or attributes thereof, or it may be used as a precursor, reducing the "input tradespace" to the next analytical treatment to a more rational set of designs. This may be done in any number of ways: taking a top set of designs including and off of the Pareto front as described earlier, taking a specified top percentage of alternatives with the highest values (irrespective of cost), etc. Which way to take a

"best set" to propagate to the next analytical processes, depends entirely on the needs of the next treatment and the overall process.

Contextual Insight.
The Needs Context is simply a way to structure and represent a multi-objective or multi-criteria decision analysis problem, where preferred attributes are valued according to some threshold and objective levels defined by a stakeholder (and independent of the tradespace range), prioritized, and used to create a decision space. Each needs context may represent:

i. Different stakeholders with different or even competing preferences for system attributes, how they are prioritized, and/or how they are valued
ii. Changes in requirements over time or preferences of a stakeholder or stakeholders across time
iii. Changing operational needs that necessitate different performance objectives, priorities, or valuations in parallel or in series

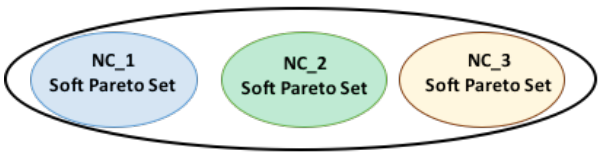
The decision space enabled by a Needs Context approach is multidimensional. Any number of Needs Contexts may be specified, and they may stem from the same or even different decision trees that capture the objective space and its hierarchy for that stakeholder or context.

It is the third bullet above that directly relates to the concept of different operational environments and CONOPS discussed at the beginning of this section. The performance characteristics in one environment may differ substantially from the same performance characteristics in another. A system may be envisioned for one use such as transport in one mission profile and its ability to offensively help control an area of responsibility via fires capability in another. Resilience in keeping with the ERS vision will be characterized in part by a system that can perform to its objectives across a wide range of operational contexts through either its initial design or flexibility to be modified in a way that adapts to the changing needs.

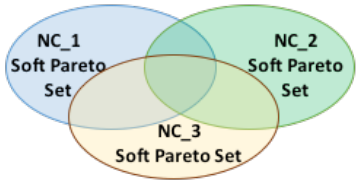
The efforts in this subtask described prior to this point focus on model structures, reusable patterns that can be carried through in appropriate data structures to support downstream analyses, generation of a tradespace and particularly a well-representative Pareto frontier as we transform from the design space to the decision space, and the value of using a soft Pareto frontier set to support analyses and further exploration. All of these concepts are vital if we are to effectively explore the impact of varying operational environments and CONOPS on our decision space for candidate system designs.

The next step is to articulate what an analyst might do with a collection of sPFs as defined by various Needs Contexts. Figure 5 illustrates three main actions that combine these sets in more or less restricted ways to provide a set from which to advance the analysis. These concepts will form the basis for evaluating a system across different operational environments and/or CONOPs, each of which may be captured in a different Needs Context. In other cases, the Needs Context may be the same but the variation of operational context will create additional performance parameters in the tradespace. For example, uphill speed carrying a payload will differ for a notional JLTV in dry sandy conditions compared to snowy conditions. The physics change. This can be represented by having different uphill speed performance parameters that correspond to each environment (i.e., two columns in a tradespace). The same combinatory approaches if we express these as distinct Needs Contexts apply.

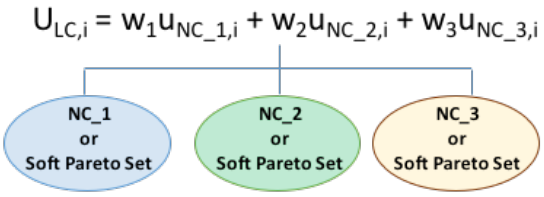
The goals are to expand and mature how we explore and analyze our decision space to help answer questions such as "which design alternatives are least sensitive to variation in the input parameters?" and "How sensitive are our valued parameters (defining a Needs Context) to (a) environment variation? or (b) operational use characteristic variation?"



1. **UNION** of the soft Pareto frontier sets across all or targeted NCs
   - *All design alternatives that may be preferred according to a set of NCs*
   - *Each NC may represent a stakeholder view or different operational need (environment or CONOPs) that compete simultaneously in the decision space (parallel) or vary over time (series).*



2. **INTERSECTION** of the soft Pareto frontier sets across all or targeted NCs
   - *Only those design alternatives that exist in each soft Pareto frontier for every NC under consideration*
   - *A more restricted set than the union, which could be a null set*



3. **LINEAR COMBINATION** of the values in each NCs across all or targeted NCs
   - *A higher-level MAV measure effectively evaluating robustness of an alternative across multiple NCs*
   - *To use with soft Pareto frontiers, the design alternative should exist in each soft Pareto frontier that will be linearly combined*
   - *Here, i in equation designates a given design alternative in the NCs*

Figure 5. How Needs Contexts may be combined to further analytical exploration of the decision space

Publications and Links to other Tasks

Concepts and methods in this subtask relate strongly to the subtasks of uncertainty (Section 2.1), product family identification (Section 2.2), and SoS evaluation (Section 2.4).

Concepts and methods in this subtask also relate to the software engineering and architecture effort (Task 3) as well as the AoA tool effort (Task 7).

2.3 PRODUCT FAMILY IDENTIFICATION AND EVALUATION

Objectives

Frequently, developing a product family for an engineered system is a more cost effective and/or better way to meet the disparate operational requirements of the end users. This task will investigate maturing the previous analytical concepts of Broad Utility (via the Needs Context) and Engineering Flexibility to help identify potential product families of systems. We will explore the potential identifying a single system that strives to meet multiple needs to a proposed product family and methods by which we may do so. The overriding question that will be answered is "*can we identify feasible sets of designs in terms of potential product families (or clusters of product family alternatives)*?"

Work Description and Accomplishments

Identifying Drivers.
The efforts in this subtask are really about identifying drivers. What key attributes, dynamics, etc. are driving which design alternatives are considered to be the "better" alternatives by decision analysis? What similar design variables are driving candidate designs in a "good" set of alternatives to be classified as similar systems that may comprise a product family? Which requirements are dominating the overall valuation in the decision space,

resulting in a clear bias toward system designs with specific architectures of component characteristics? There may be other drivers biasing the decision space as well from specific technology component selection to a prioritization of one stakeholder's needs.

The key to providing actionable insights to key decision makers is not simply to articulate performance capabilities as they meet requirements but to discover what characteristics are driving the decision outcomes, the very nature of the Pareto-efficient decision space.

Of all the questions that could be asked, this effort focused on two distinct but highly related questions:

i. Which requirements, expressed as objective preferences in a Needs Context, are driving the cost across the preferred set of alternatives?
ii. Can we identify groups of similar designs across a preferred set of alternatives that may be similar enough to form the basis of a product family, thereby meeting diverse needs more effectively?

Meeting disparate operational needs more cost effectively than a monolithic system solution is the primary reason product families emerge from an acquisitions process. We are striving to help identify that possibility earlier in the design phase. This work builds directly from the previous subtask, especially the context-dependent generation and identification of an sPF.

In the discussion that follows, we will first address these questions separately and then combine their commonalities into a holistic picture.

Identifying Drivers – Requirements-to-Cost.

A GAO report on challenges associated with AoAs found many programs begin without a sound match between requirements and the resources needed to achieve them (U.S. Government Accounting Office, 2009). That is, these historical programs entered the acquisition process with requirements that were not fully understood, cost and schedule estimates that were based on optimistic assumptions, and a lack of sufficient knowledge about technology, design, and manufacturing. Because the maturation of requirements was performed in a separate process that was not well connected to the evaluation of alternatives, a holistic evolution of design alternative selection and refinement alongside increasing technical knowledge and requirements compromises and refinement was not possible. In turn, this resulted in poor cost, schedule, and/or performance outcomes.

Motivated by this discussion and by any number of presentations given by University of Southern California Professor Dr. Barry Boehm in which he relates a stakeholder finally understanding how much a requirement is costing the program and coming to the conclusion "it's not a requirement if we can't afford it", we sought to identify which requirements (expressed a preference objectives defining a Needs Context) are most responsible for driving costs of the alternatives identified as "best" options. While there is a lot of attention paid to the generation of Pareto-optimal alternatives, less attention is paid to understanding what drives the value vs cost trade-off on the frontier.

Here we will re-specify a MAV, the weighted sum of n valued attributes, to clearly show the functional dependency of the value function portion of the equation:

$$MAV = \sum_{i=1}^{n} w_i v(y_i, t_i, v_{ti}, o_i)$$

Where:

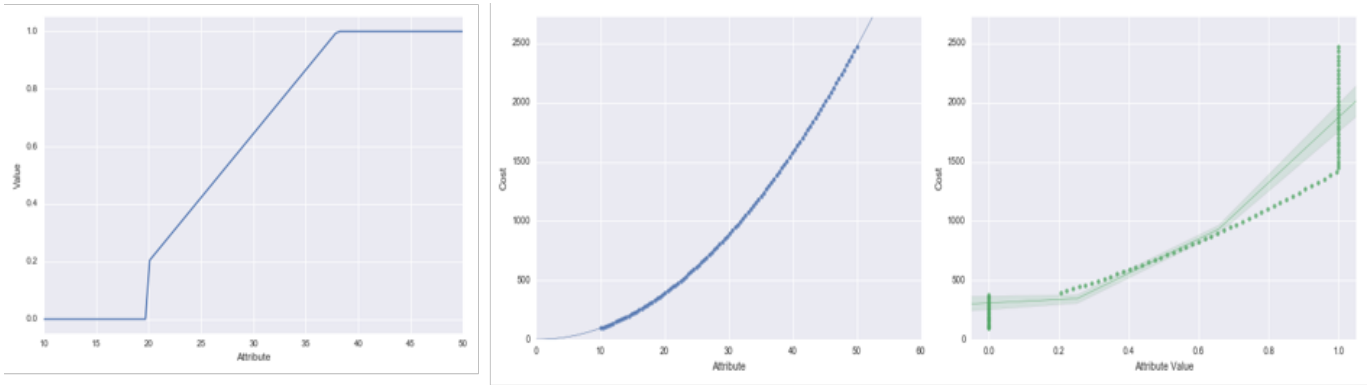
$$y = attribute$$
$$w = value\ weight$$
$$t = attribute\ threshold$$
$$v_t = value\ at\ threshold$$

$$o = attribute\ objective$$
$$v = value\ function$$

While the MAV is fully defined by the value functions and the feasibility of attribute combinations, the cost is driven by the attributes and the design properties of the alternatives. The relationship between cost and attributes is defined by the underlying variable relationships (i.e., the "physics") of the alternatives and not by pre-defined value functions. This relationship may not be straightforward, depending on the models or data used to generate Pareto-optimal points. The cost-attribute relationship is made more complicated by the down-selection to alternatives that meet certain MAV criteria.

A significant challenge when using data from an sPF is that the inputs to the MAV as a function of cost relationship do not follow a 1-to-1 equivalence. There may be many designs where the attributes defined in the decision analysis problem (i.e., a Needs Context) are at or above objective levels, just as there may be many designs at or below threshold levels. This creates a 1-to-many relationship from the inputs (raw levels of each attribute) to outputs (valuations of those attributes defined by the value functions) that is not due to any stochastic effects in the model. These regions of an sPF are therefore very difficult to analyze using any type of method such as regression that thrives on a more 1-to-1 relationship dynamic.

The MAV as a function of cost decision space obfuscates the attribute-to-cost relationship due to utility function aggregation and valuation thresholding. Figure 6 illustrates a notional attribute shown as being valued linearly before the objective level together with an assumed example relationship of cost as a quadratic function of the attribute. Also shown are two 2nd-order linear model fits of Cost = f(attribute) and f(attribute value). The large 'column' of higher cost points causes a regression to over-emphasize the relationship between the cost and attribute values. The same 'objective concentration' phenomenon occurs below the threshold, so valued attributes can exhibit both objective and threshold concentration. In a single dimension, this is easily remedied by ignoring the values below threshold and above the lowest-cost max-value point. Doing so forms a perfect fit since we have restricted the values to be only in the monotonically-transformed region of the value function. When moving to higher dimensions, though, points above the objective value cannot be discarded. This is because in a MAV it is possible to have highly valued points that are at their objective value in many attributes, but not yet at objective in others. Depending on the physics underlying the points, the realization of high value in one attribute may necessitate high cost, thus retaining many points on the 'columns' of some attribute values.



a) Linear value function

b) Quadratic cost as a function of attribute level

c) Linear regression fit of cost to attribute value

Figure 6. Example of challenges value functions pose to fitting data to discern driving factors

In light of these data characteristics in the decision space, we explored various methods that could help us identify drivers across the sPF, using both the notional JLTV model and the SMR helicopter model discussed under Section 2.2. A summary of these methods to elucidate variable importance is presented here:

- Linear Regression Models of 1st and 2nd Order.

  Linear models are a general class of models that fit a regression equation that is linear with respect to the coefficients. These include Ordinary Least Squares (OLS), as well as more advanced fitting methods such as LASSO and Ridge Regression. Linear models are good candidates for modeling due to their easily interpreted coefficients and because of their ability to fit a model quickly and in almost any software language. The primary limitation is that interaction terms make coefficients harder to interpret. The first and second order regression models worked well with the notional JLTV model, but not as well with the SMR helicopter model. The latter had several highly correlated attributes and some in which the attribute value exhibited small variations over a comparatively wide range of costs.

- Generalized Additive Models.

  Generalized Additive Models (GAMs) are a class of regression model that uses a summation of basis functions to model data. One class of GAMs that is considered here is the Multivariate Adaptive Regression Splines (MARS). MARS uses a 'hinge function' as its basis function. A constant determined by the model fitting algorithm, also called a 'knot' in the hinge function serves as an activation point above or below which a linear equation of the form $c_i(b - x)$ is active, and below or above which the hinge function is zero. The advantages of MARS is that it has the same interpretability as OLS with the added feature that the hinge functions allow for the input space to be segmented into different regions. This essentially creates several linear models throughout the data, which could lead to better fits and more interesting results, such as knowing that a variable is only impactful in different regions of its values. When interactions are included, interpretability suffers in the same way as for OLS. For the notional JLTV model, which used a Needs Context valuing 'payload mass' and 'cruise range', this approach showed payload mass as a large cost driver when it is above 0.38 (when the first basis function activates). Below that value, payload has a slightly lower impact on cost. The cost impact of cruise range remains constant according to the model. The MARS fit is able to add more nuance than the OLS regression to the interpretation of results while still being easily interpreted. The fit says that low values of payload utility do not impact cost as much as high values. This is most likely indicative of more flexibility in finding low cost ways to achieve lower-valued payloads, while high values for payloads require more cost. The model fit for the SMR helicopter, based on R^2, is better than that of its OLS model. We see the same behavior as in the notional JLTV example model where attributes become more impactful on cost the larger they are. Interpretability, however, is more of a problem and dependent on knot locations. The large coefficient values near the top of the attribute levels make interpretation difficult, which would hamper analyses that change thresholds and objectives to enable further requirements studies. The repeated basis functions and overlapping basis functions that are seen further complicate the issue.

- Neural Network Models.

  Neural Networks (NNs) are a large class of models that use a variety of 'activation functions' in a network that takes in the data as inputs and passes the data forward through the activation functions in successive 'hidden layers'. This kind of network is known as a feedforward neural network. For regression, a network with one hidden layer of sigmoid functions and a single linear output layer, all with bias nodes, is a good starting point. NNs have the advantage that they do not assume any structure to the data, rather that structure is what is trying to be learned. The disadvantage is the NNs are notoriously difficult to interpret. Since NNs use a series of coefficients and activation functions, the importance of an input is found by measuring the total of all the coefficients that the input travels along in the network. This will measure the aggregate amount of activation and deactivation that the network has selected for the input. The importance can be measured using the absolute value of the activation weights or by accounting for the sign of the weights. The former describes the gross amount of work being done by an input, while the latter describes the net amount. Of all the methods tried, the Absolute Neural Network Importance

measure is the only clear loser. This is due to its inability to tell the difference in activation or deactivation in the network.

- Local Covariance and Correlation

  Local Covariance (LC) is a weighted version of the standard covariance definition where weights are applied as a function of the distance of a data point from a given reference point. Like covariance, it gives the strength of a monotonic relationship between features of data, but prioritizes information that is closer to a particular point in the input space. The local covariance is calculated around every data point in the soft Pareto Frontier, and is used to produce plots of covariance or correlation values of the attribute as a function of cost. The local covariance alone was not very informative since it did not reside on any given interval, and in fact may be quite small. Converting covariances to correlations provides more discriminating information.

- Partial Correlation.

  Partial Correlation (PC) is a measure of correlation that controls for the effects of other variables. It does so by looking at the correlation between the residuals of two attributes after the two attributes have been fit by a linear model of the other attributes. Given a set of attributes, A, if X and Y are attributes in that set (such as cost and payload mass), then the partial correlation is found from:

$$X_r = X - f(\{a \in A : a \notin \{X, Y\}\})$$
$$Y_r = Y - f(\{a \in A : a \notin \{X, Y\}\})$$
$$partial\ correlation = corr(X_r, Y_r)$$

  Doing so attempts to remove the effect of correlated attributes, which we have already seen occur and cause issues with interpreting the models discussed so far. Visually for the Local Partial Correlation and numerically for the Global Partial Correlation we see agreement with several of the other methods. The partial correlation reduces the impact of moderately correlated variables by fitting a model to the data using highly correlated variables, leaving very little residual information for correlations on the attribute in question.

The low dimensional attribute space for the notional JLTV example made all the analysis methods simple to interpret, and no methods disagreed in this case. The high dimensionality and high correlation of the SMR example were expected to make some of the analyses more differentiable, which were seen to some extent. The difficulty here is discovering which variables are leading and which are following. When they are all highly correlated it may very well be that all of them are significant cost drivers. The notional JLTV example illustrated that non-cost drivers can be readily detected. Binning the sPF into regions helped to some extent by illustrating how cost drivers can change across the sPF, but still suffered from the same interpretability issues as to what precisely was driving when there were highly correlated attributes.

Whether the tradespace was generated using LHS or the algorithmic approach also made differences in the ensuing analyses. Soft frontiers were defined as relaxing the dominance relation within a user-defined band. The soft frontiers allow for more data to enter the analysis to improve them. Without the extra data, methods other than linear regression would not have worked due to the sparsity of data, especially for the local correlation methods. For the LHS, there were fewer data points along the frontier and so fewer in the sPF set for analysis. Using the same methods on each type of generated sPF found lower certainty and fit quality for the LHS generated tradespace. For the SMR example, a first order linear regression achieved an R^2 of 0.824 for the NSGA-II tradespace compared to 0.74 for the LHS tradespace. Interpretability issues between the model parameters became more challenging as well. In all, all methods produced poorer results when applied to a LHS frontier as compared to the NSGA-II derived frontier. This is because there are fewer data points within the soft frontier, as well as no real mechanism of enforcement except space-filling sampling and luck to achieve a good frontier.

For studying cost drivers, the actual attribute values themselves should be used with the appropriate normalization, as is standard practice for many statistical methods. We do not fit any methods to the value function levels of the attributes due to the non-uniqueness of data at and above objective and below threshold as explained previously. By using the attributes levels directly, we retain the information of the physics of the problem. The soft Pareto Frontier already implicitly contains the value statements that generated it, so we are no longer consider with what generated value, but what is changing in the valued attributes that changes cost the most.

Identifying Similarity and Relationship to Drivers.
Our initial goal was to see if we could use the concept of soft Pareto frontier sets as a foundation representing "better" design alternatives for a given Needs Context, combine the sPF sets from at least two differently prioritized Needs Contexts via their union, identify designs exhibiting some definition of similarity, and thereby identify a potential product family (or families) for which designs in that set may satisfy distinct Needs Contexts better than a single design. The intent was not to to address engineering feasibility of the designs at this point. Simply, we wish to identify these groupings as potential families of designs.

We used a combination of randomized data representing a tradespace as well as the JLTV model to evaluate different similarity measures that could apply to design data. Among these were the Sorensen similarity quotient (a variant of the Jaccard Index), where the similarity quotient will exist for each pair of candidate designs (i.e., *i* and *j*) and is given by:

$$QS = \frac{2|X \cap Y|}{|X| + |Y|}$$

where |X| and |Y| are the numbers of species in the two samples, and QS ranges between 0 and 1. In our case, |X| is the number of design variables in design alternative_i and |Y| is the number of design variables in design alternative_j (which for us is equal across all *i* and *j*). The intersection of the two is how many design variable levels they have in common, with common being +/- epsilon for that design variable. The Sorensen distance measure is semi-metric in that it does not satisfy the triangle inequality and given by:

$$d_S = 1 - \frac{2|X \cap Y|}{|X| + |Y|}$$

We also investigated an alternate similarity measures, the Jaccard distance and its modification given by Tamimoto. The Jaccard similarity measure is given by:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

where A and B are sets of data and, for our problem, their components correspond to the design variables for distinct design alternatives as for the Sorensen measure. The Jaccard distance, which measures dissimilarity between sample sets, is complementary to the Jaccard coefficient and is obtained by subtracting the Jaccard coefficient from 1, or, equivalently, by dividing the difference of the sizes of the union and the intersection of two sets by the size of the union:

$$d_J(A,B) = 1 - J(A,B)$$

In a modification of the Jaccard index, generally called the Tamimoto index, the similarity index is value is equal to the Jaccard coefficient of the two sets when each sample is modelled as a set of attributes. The distance metric is different, however, and is intentionally semi-metric:

$$d_{Tamimoto} = -log_2(J(A,B)$$

Of these measures, the Sorenson places greater weight on common elements than Jaccard, e.g., 2 sets with 8 items each and 4 in common would have: S_QS = 2*4/(8+8) = 8/16 = 0.5 and S_J = 4 / (16 - 4) = 4/12 = 0.33. Sorenson asserts that the co-occurrence or coincidence of variable states among objects is more informative or important than disagreements. This is based on the logic of the harmonic mean and is thus suitable for data sets with large-valued outliers. It may, however, increase the influence of small-valued outliers. For comparison, Euclidean distance measures will place greater emphasis on outliers by definition.

As an initial step in our investigations, we evaluated all of the candidate designs in a sPF on the basis of their design variables (and not performance attributes). Similarity between any two design alternatives was defined on a soft measure, for example, +/- 5% of the design variable value, and not on a binary equivalence. This produces a square (number of candidate designs by number of candidate designs), symmetric similarity matrix with values of 1 along the diagonal and all values between 0 and 1 otherwise. We arbitrarily determined a threshold to specify two designs as similar or not. For example, if the threshold was set to be 0.75, all design pairs with similarity measures at or above 0.75 were defined as similar while those with measures below 0.75 were not.

We then investigated various methods by which we might group the designs that were determined to be similar including several clustering methods that would allow for non-unique clustering (i.e., designs could reside in more than one cluster as in fuzzy k-means clustering) and not necessarily every design had to reside in a cluster.

We found that we could apply the similarity measures easily and with strong intuitive interpretability. Clustering, however, was more of a problem. There was so much variation across the sets of designs that were deemed to be similar, that the designs did not readily separate into well-formed groupings. There were several reasons for this difficulty, but chief among them were the relationships of the design variables with respect to membership in the sPF and with respect to each other. There were several ways a design alternative could reside in the sPF, and the design variables most responsible for helping it get there (i.e., the design variable drivers) were not necessarily those that ended up being the primary basis for similarity. We had made no weighting of design variable importance in the similarity measures. Also, in a tradespace for an engineered system, the design variables will not be independent. There will be constraints in terms of how the design variables relate to one another to produce a feasible engineered system, even if those constraints are hidden in the model transfer functions because the engineer who understood the system physics captured them but they are not stated explicitly otherwise. These two factors strongly affect what we really need to be evaluating on as the basis for system design similarity and relate to the findings in the cost driver section. We will synthesize these efforts and discuss next steps below.

Synthesizing the Findings into a Holistic View.

The challenges discovered during the course of this effort were similar and due to the nature of the relationships between variables in the design space and the relationship between the design space and the decision space. Namely, the mapping of any design space to a decision space may not at all carry through the main drivers of the problem. For example, if low fuel consumption technologies are a major cost driver but fuel usage is not a valued attribute, the Pareto-Frontier will contain only minimal variation in fuel consumption so that it can avoid high-cost regions. An analysis of the sPF alone would show that cost was not driven by fuel consumption even though that is not reflective of the reality. The similarity effort also found that design variables that were responsible for any two designs being classified as similar were not necessarily the important ones that drove membership in the sPF. Also there were strong correlations found in some problems between valued attributes in the decision space as well as strong correlations and non-independence across many design variables in the design space.

Based on the findings here, there are two primary recommendations to make these approaches robust across a wide variety of problem types:

1) Explore robust ways to identify drivers acting on the complete set of raw data, meaning design variable and performance attributes together, for designs in sPF sets. These drivers will always be related, and the more complete picture can help resolve the issue of major drivers not being captured in the decision space due to the preference architecture.
2) Mature our understanding of the most effective ways to adaptively generate and sample a tradespace to get a "good" Pareto Frontier and therefore a rich, well-represented set of designs that offer a solid foundation for analyses as per (1) above.

Publications and Links to other Tasks

Concepts and methods in this subtask relate strongly to the subtasks of disparate operational environments (Section 2.2).

Concepts and methods in this subtask will relate to the software engineering and architecture effort (Task 3) once they mature sufficiently to merit inclusion in the tool.

2.4 SYSTEMS OF SYSTEMS AND CAPABILITY PORTFOLIO ASSESSMENT

Objectives

For this subtask, we set out to develop at least a notional framework or guiding philosophy for how to bridge the concepts and measures of resilience at an individual system level with the system of systems (SoS) level and vice versa.

Through ERS, the DoD seeks a transformation in Defense acquisition with the contribution of systems engineering throughout a system's lifecycle that is needed to address a geopolitical environment marked by rapidly changing threats, tactics, missions, and technologies. As a critical part of this challenge, a growing number of military capabilities are achieved through a SoS or capabilities portfolio approach, even though requirements and design decisions are nearly always specified at the individual system level. Defense systems typically operate within an environment with many other systems, rarely performing operations in a strictly solitary sense. In tandem with the SoS engineering efforts, the DoD focus on ERS strives for effective and efficient design and development of complex engineered systems across their lifecycle and changing operational needs.

With the framework of ERS as a foundation, we seek to develop an understanding and description of cross-scale resiliency through an operational lens to help bridge SoS and constituent system evaluation. While multi-scale assessments seek to evaluate behavior or some other attribute at two or more distinct and discrete scales, cross-scale analyses take multi-scale assessments and purposefully look for interactions across the scales. We strive to build a more tangible perspective on resilience and how we can create more informed decision analysis at both the system and the SoS levels through a more integrated and operationally relevant process.

Our efforts to date have included extending web-based decision support frameworks to support methods, processes, and tools (MPTs) that enable highly flexible and scalable tradespace exploration and analysis. As these efforts mature, we must better understand how to leverage and integrate models and/or outputs at the SoS level to guide tradespace exploration and analysis at the constituent system level and vice versa. The penultimate goal for this work is to begin to generate the insights needed for truly synthesized SoS-to-Local System analyses,

thereby helping to evolve the overall DoD acquisitions process. Adding conceptual dimensionality at the SoS level can help focus constituent-level system design analysis, feeding back to and therefore more effectively supporting SoS evaluation.

Work Description and Accomplishments

We thoroughly investigated concepts of resilience across different disciplines, reviewed any metrics that had been suggested in the literature, and also reviewed the SoS work being conducted by Purdue University under the SERC as well as their associated publications. We also reviewed DoD guidance on the SoS perspective as well as how systems engineering should fit or evolve to address the future needs facing the acquisitions process.

As we began this study, our goals were to specify a relatively direct mapping between a resilience hierarchy at the individual and SoS levels of analysis. As we synthesized the understanding of resilience and fields of study that must come together to support the next-generation of materiel analysis, however, we realized that resilience is an emergent attribute and its evaluation would not be so simple. This work was summarized with specific recommendations for how to harmonize SoS-to-individual system analyses in the report "*Cross-scale resilience: Relating Systems of Systems to Individual System Analysis and Back Again*" (Sitterle, 2016). The discussion that follows below is the executive summary of this work.

Resilience is achieved through engineering design decisions that result in a system's or SoS's ability to be (i) effective in the face of many threats to its operational performance (preparation enabling the ability to be absorptive in the moment), (ii) robust in terms of its ability to deliver intended performance across a diverse range of operational contexts, (iii) amenable to modification that enables the system to recover from an adverse event or better address a new threat that presents itself in the future, and (iv) efficient in terms of the time, cost, and personnel resources required to do so.

Resilience is not a linear aggregate of other system qualities, but rather their contextual, nonlinear synthesis. We arrive at an understanding of whether or not our system is resilient through an ebb and flow of design alteration, requirements maturation, and M&S of different contexts and system architectures, all of which are linked, guided, and supported through decision analysis. To help evolve the overall DoD acquisitions process for new system development, we need a more representative and effective synthesis of SoS to local system analyses. We seek to contribute to the discussion by deliberately bringing in an operational lens, a "red team" view, to compliment the prevalent capability-based perspectives currently in practice.

The DoD seeks a transformation in Defense acquisition with the contribution of systems engineering throughout a system's lifecycle that is needed to address a geopolitical environment marked by rapidly changing threats, tactics, missions, and technologies. As a critical part of this challenge, a growing number of military capabilities are achieved through a SoS approach, even though requirements and design decisions are nearly always specified at the individual system level. Defense systems typically operate within an environment with many other systems, rarely performing operations in a strictly solitary sense. In tandem with the SoS engineering efforts, the DoD focus on Engineered Resilient Systems strives for effective and efficient design and development of complex engineered systems across their lifecycle and changing operational needs. With the framework of ERS as a foundation, we seek to develop an understanding and description of cross-scale resiliency through an operational lens to help bridge SoS and constituent system evaluation.

Whether focused on design of new systems as for the ERS program, systems of systems engineering, or lifecycle perspectives, "resilience" has become a key term across many aspects of the DoD's acquisitions ecosphere. In this session, we aim describe resilience in a way that is meaningfully helpful to system design and decision analysis. We offer the view that resilience is an inherent system quality created through design choices that enable a

system to maintain its performance objectives in the face of diverse operational challenges, in either a preparative or recovery sense, within acceptable time and cost parameters. This view, while abstract enough to be broadly relevant, is stated in a way that offers a clear linkage between design choices for a system, other system qualities those choices may produce, and the functional objectives for the system. It includes the operational perspective, preparative design choices that enable the system to perform across diverse contexts, and (related to those preparative choices) the ability to be modified later in order to preserve performance. What we explicitly added to the prior ERS view were the concepts of time and cost to do so.

This concept has critical ramifications with respect to the realities facing fielded systems and their analysis. Critical system functionality and performance characteristics must be able to withstand adversary tactics, diverse operational environments and, especially in today's joint defense environment, variation in how that system is used in the course of pursuing mission and/or tactical objectives (i.e., its CONOPS). To understand systems across these contexts, we must extend our lens beyond the current capability-based analyses – going beyond the principles of modular, open systems – to include a "red team" view during the design process itself and not limit these considerations to a verification and validation stage.

Most current SoS analyses investigate only system-specific, capability-based concepts such as reliability or mean time between failure (MTBF). Yet to effectively design an SoS to support operational goals, we must include the concept of threats to the realization of system-level critical functionality and performance that are needed at the SoS level. Whether due to environment, operational use, or adversary action, some threats will affect certain constituent systems and not others. Articulating classes of threats relevant to the SoS provides specific guidance for engineers and subject matter experts to construct appropriate analyses at the individual system level. A key aspect will be to identify and incorporate necessary M&S components to produce the relevant analytical data. Individual system analyses will then feed the SoS evaluations more comprehensively and specifically. SoS engineers will understand not just the reliability of a constituent system, but the likelihood it can withstand a threat of a given magnitude.

There are several related concepts that are especially vital to putting these ideas into practice. For one, we advocate embracing the notion of co-evolving requirements alongside system design from the earliest stages. Early stage design evaluations should include structures that help evaluate requirements in terms of their impact to system performance, overall affordability, and other system qualities as part of an integrated decision analysis process. How different requirements for a potential system may impact capabilities, performance characteristics, and affordability (along with other program concerns) should be transparent to key decision makers. Since this is the very basis of tradeoff decisions that must be made, bringing them into the decision analysis process offers an informed way to help mature them into "good" requirements.

Secondly, a 'threat' is anything that may compromise intended functional performance. Threats can result in direct damage to a system, whether from an adversary action or operation in an environment too extreme for the design. A threat is therefore a multidimensional concept encompassing how a system is used, under what conditions, and under what external influencing factors including adversary actions or even actions from other systems in the SoS. The system alone may not be resilient to a threat, but within the SoS, it is. The SoS view is vital to helping bound the scope of the analytical effort required at the individual level.

In addition, synthesizing capability and threat-based perspectives within an M&S environment requires a different mindset and initial level of effort from the mean-value approach where system qualities are evaluated in the abstract. To evaluate a system's resilience, M&S components must accept inputs representative of threat characteristics and produce outputs representative of the consequences to system capabilities. Importantly, these M&S components must be coupled. To support these concepts, we seek a unification of MBSE, requirements analysis, and decision theory as illustrated in Figure 7. Bringing these areas of expertise together

will help us to make appropriate choices regarding levels of abstraction and interaction between M&S representations for the current stage of the design process.

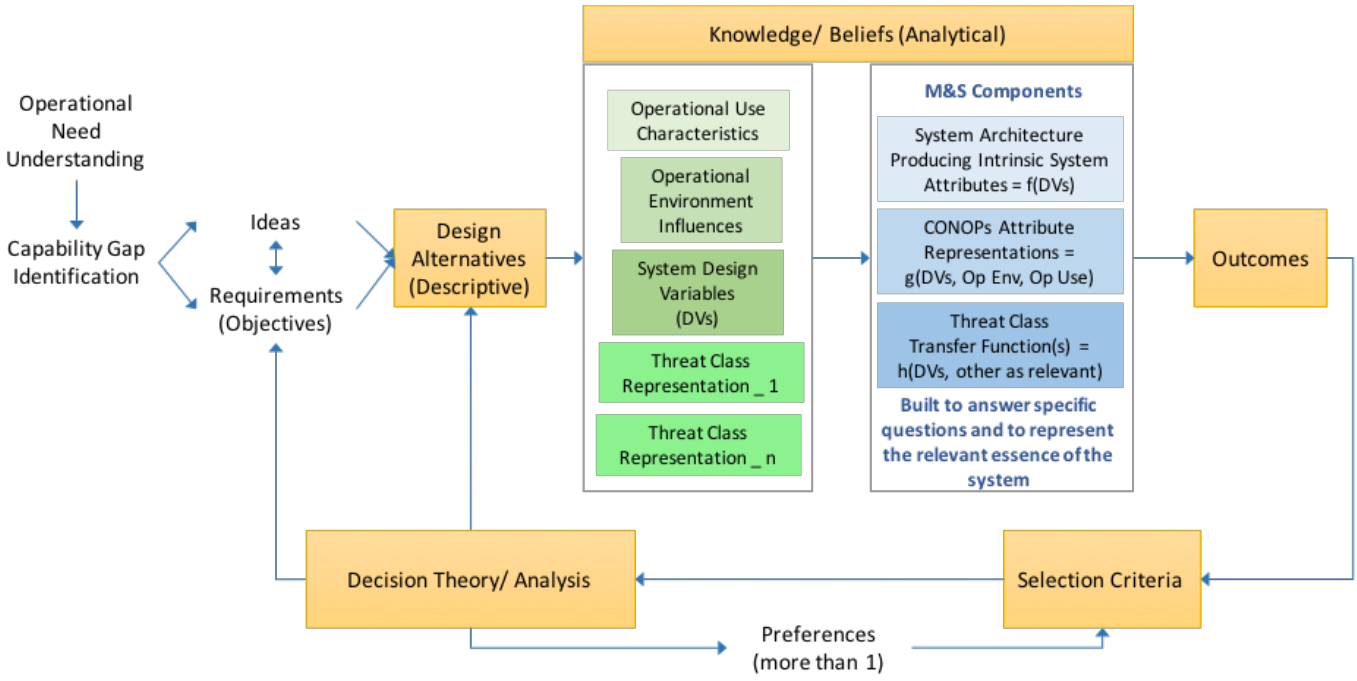


Figure 7. Illustration Unifying MBSE, requirements analysis, and decision theory within a synthesized framework

Throughout this process, we need to incorporate multi-objective decision analysis as a truly integrated tool. Decision analysis in this context should not be viewed as an activity that occurs only at the end of a design process to guide decision makers. Instead, decision analysis should help frame the problem (or each stage of the problem) from the beginning. In this paradigm, decision analysis will help frame what analytical data must be produced as part of a tradespace for evaluation. It will help guide the identification of the next M&S components required to support informed decision making. This continual refinement will help target the early stages of design evaluation, enabling not only a more thorough understanding of the problem but a more efficient process.

In contrast to the traditional waterfall approach, this paradigm constitutes a more iterative, cyclic process manifested as an ebb and flow between mission engineering needs at the SoS level, the individual system evaluation needs, and the generation of new knowledge via data-driven analyses. While adding a "red team" perspective integrating capabilities and threats does not immediately solve the problem, it guides and focuses our analyses and empowers more comprehensive decision analysis. In the end, we strive to create a more enhanced understanding of the complex relationships across the constituent systems, the SoS, operational environments and CONOPS, as well as adversary threats for systems engineers, mission engineers, and decision makers. Bidirectional concepts of operations will result in a more thorough, grounded approach to mission engineering for SoSs as well as enable a mission-relevant exploration of the tradeoffs between design margins and affordability at the system level.

Publications and Links to other Tasks

This work was summarized with specific recommendations for how to harmonize SoS-to-individual system analyses in the report "*Cross-scale resilience: Relating Systems of Systems to Individual System Analysis and Back Again*" (Sitterle, 2016).

Portions of this work and some of the concepts it embodies were presented to attendees of the Decision Analysis and Resilience Workshop hosted by the Institute of Systems Engineering Research (ISER), U.S. Army Engineer Research Development Center (ERDC). The workshop was held in May 2016 in Vicksburg, MS.

This work was also submitted for consideration as a presentation under the SoS section for the NDIA Systems Engineering conference in October 2016.

3 TRADESPACE SOFTWARE DEVELOPMENT (AND TRADESTUDIO INTEGRATION)

3.1 OBJECTIVES AND SUMMARY

This primary goal of this task was to extend the capabilities of the ERS TRADESPACE v1.0 software architecture developed under an earlier effort (SERC RT-120). Furthermore, this task sought to develop modules for the broader ERS TradeStudio software suite developed by ERDC. The ERS TRADESPACE toolset developed under RT-120 has been updated to version 1.4.1 and released as a rebranded "**ERS TradeBuilder**" under the TradeStudio suite.

Building off previous phases (RT-120), the workflow needed for a practicing Systems Engineer is a nontrivial set of interconnected steps that depend on the particular problem under consideration and the mental model of the engineer (Figure 8). For example, with the next generation military vehicles the Systems Engineer must need to consider all of the requirements that the vehicle must meet whether operational, fiscal, or technical. These requirements as well as the fundamental vehicle dynamics combine to generate a multitude of system alternatives that need to be explored and traded against one another. If no feasible alternative can satisfy the high level needs then the design process can begin again either by considering new architectures (with associated physics based models) or relaxing some stringent requirements. This is just an over simplified example but all these steps should be inside a common data model that pulls together the desperate analyses and conform to the Systems Engineers thought pattern facilitating their workflow rather than limiting their creativity.

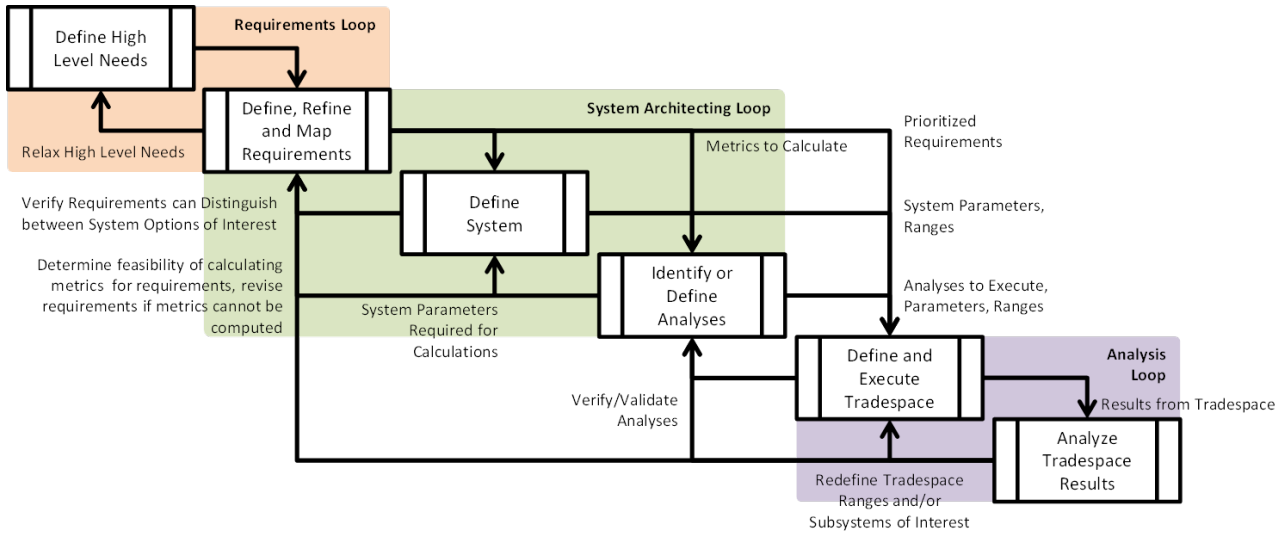


Figure 8. Systems Engineering Networked Workflow

This means the tool needs to allow the systems engineer to move easily between different tasks and views so they can understand more aspects of the problem and the implications of certain decisions. Furthermore, by tracking the actions taken by the Systems Engineer the evolution of the system and design analysis history can be replayed for reporting purposes and presented to higher level decision makers and stakeholders.

The Systems Modeling Language (SysML) is a standardized set of descriptions that has been adopted by the Systems Engineering community to help document the entities and their inter-relationships for systems. The ERS Tradespace Tool (TradeBuilder) core data model is inspired by SysML for the visual representation of the encoded system but currently only supports a subset of the full SysML specification.

3.2 ERS TRADEBUILDER WALKTHROUGH

The ERS TradeBuilder v1.4.1 revolves around three primary views (Define, Execute, and Explore) to try and satisfy the main analysis loops that must be performed by the Systems Engineer. When arriving at ERS TradeBuilder the user is presented with the landing page, Figure 9, giving a brief overview of the alternative views and providing a mechanism for login and authentication before entry into the tool. Once authenticated, the user can then proceed to the different views based on their particular needs and workflow.

Figure 9. ERS TradeBuilder Landing Page

3.2.1 DEFINE

In the Define views the system of interest can be collaboratively authored in SysML Block Definition, Parametric Diagrams, and Requirement Diagrams.

Block Definition Diagrams

Block Definition diagrams capture the structural information of the system. For example, if designing a helicopter the helicopter could own various value properties like weight or cost and contain additional parts (as separate blocks) like the airframe or rotor, Figure 10. In turn these blocks can have their own value properties and sub parts.

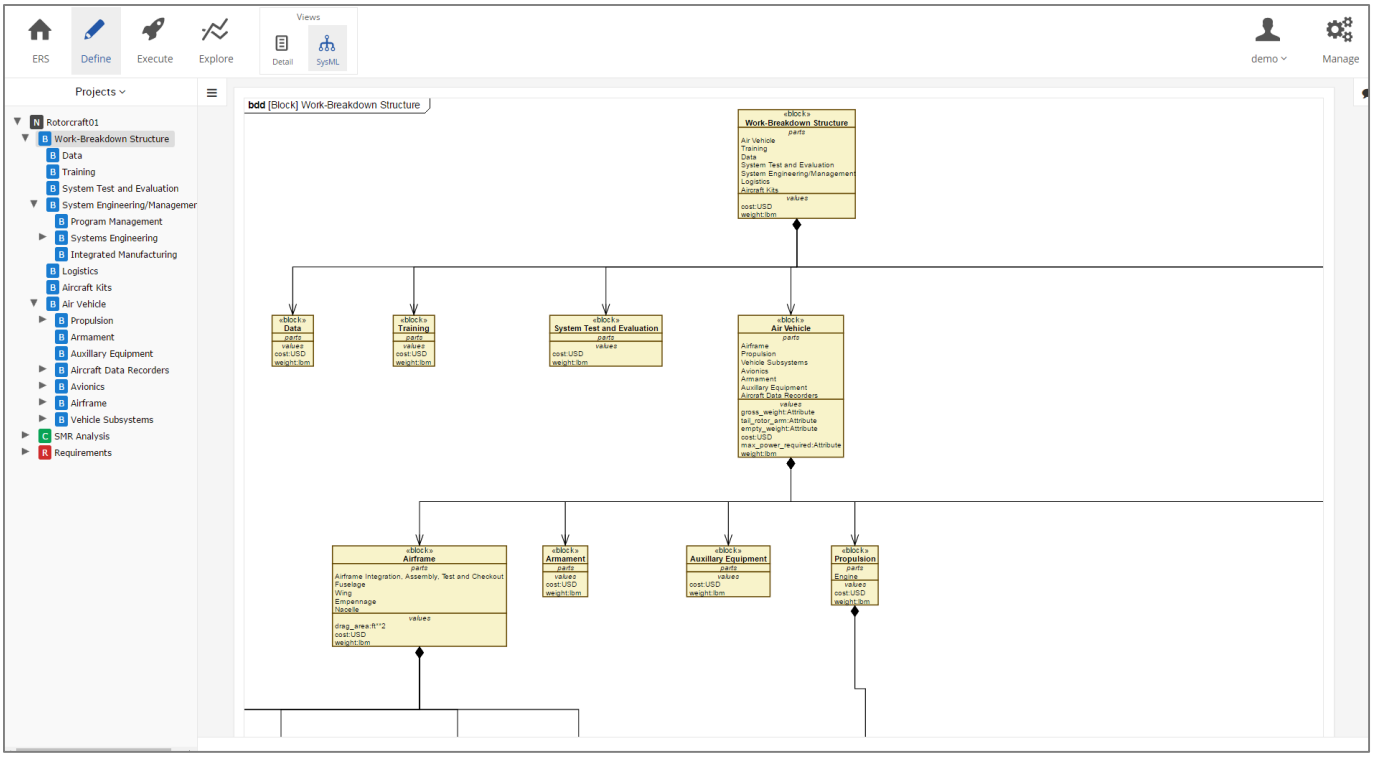


Figure 10. SysML Block Diagram Example

To interact with the diagram, the user can click on any block to open a popup showing the detail view with various properties for the user to edit, Figure 11. These properties include adding or removing block value properties or changing a particular value properties lower limit, upper limit or unit. Additionally the detail view for the block allows the user to add/remove different relationships and to change the kind of relationship between different entities. For example, below shows the detail view of the Work-Breakdown Structure block that lists the different value properties and all the part containment relationships. If the user were to edit this popup the diagram would reflect the new reality.

Parametric Diagrams

Parametric Diagrams focus on the connection of these value properties to the inputs and outputs of constituent engineering physics based analysis models, Figure 12. For example with the helicopter, there might be a physics based model to calculate the general vehicle performance based on the main rotor diameter and the air vehicle gross weight.

By adding these linking relationships between the different block value properties and the constraint input and outputs the overarching set of constraints and their execution order structure can be reasoned about when trying to get the full performance analysis for a particular helicopter. For example, when clicking of the SMR Analysis Constraint a diagram is rendered showing the execution order and information flow between the constituent (Sizing, Performance, and Cost) constraints, Figure 13.

Requirement Diagram

Requirement Diagrams show how the different requirements are linked to the block value properties and the hierarchy of requirements that can be used in downstream analyses to weight the different alternatives. Figure 14 shows the requirements diagram for the helicopter example as the high level requirements are broken into lower level requirements tied to particular value properties.

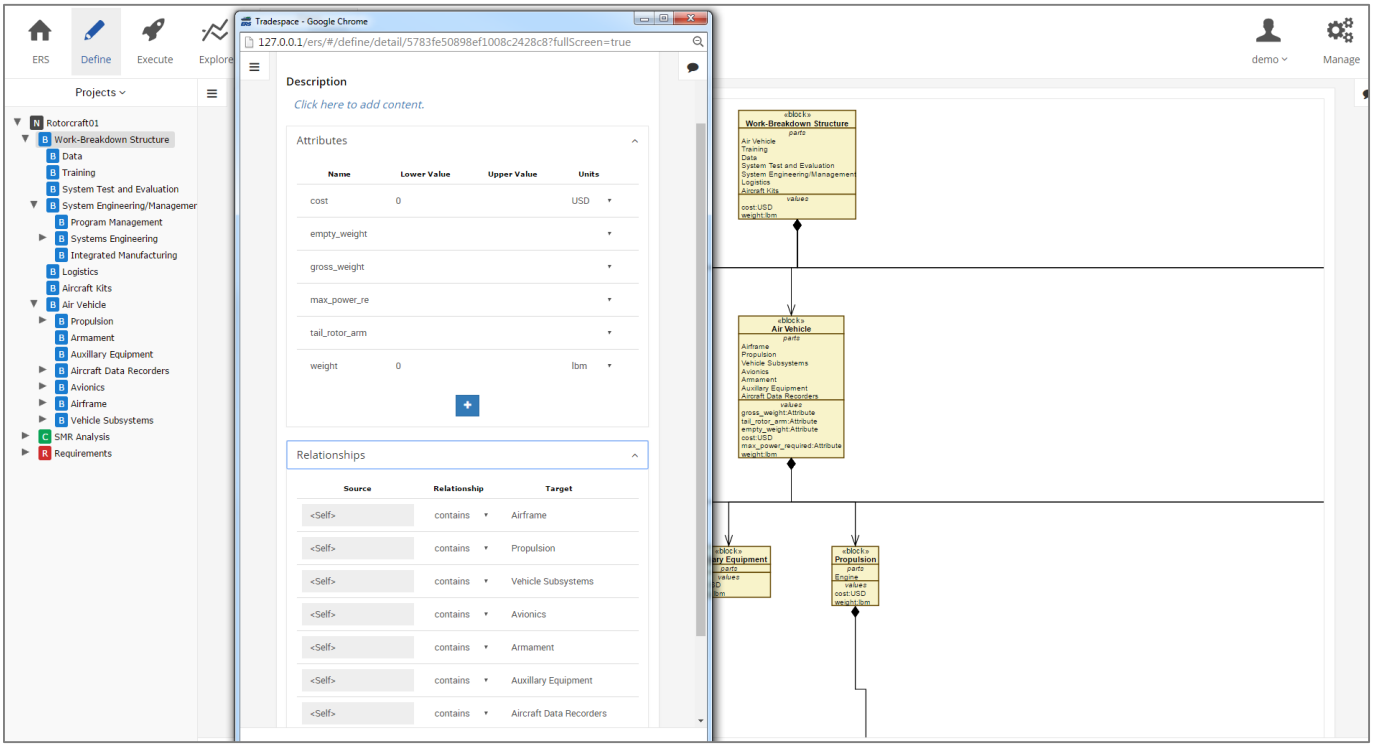


Figure 11. Detail SysML View Popup

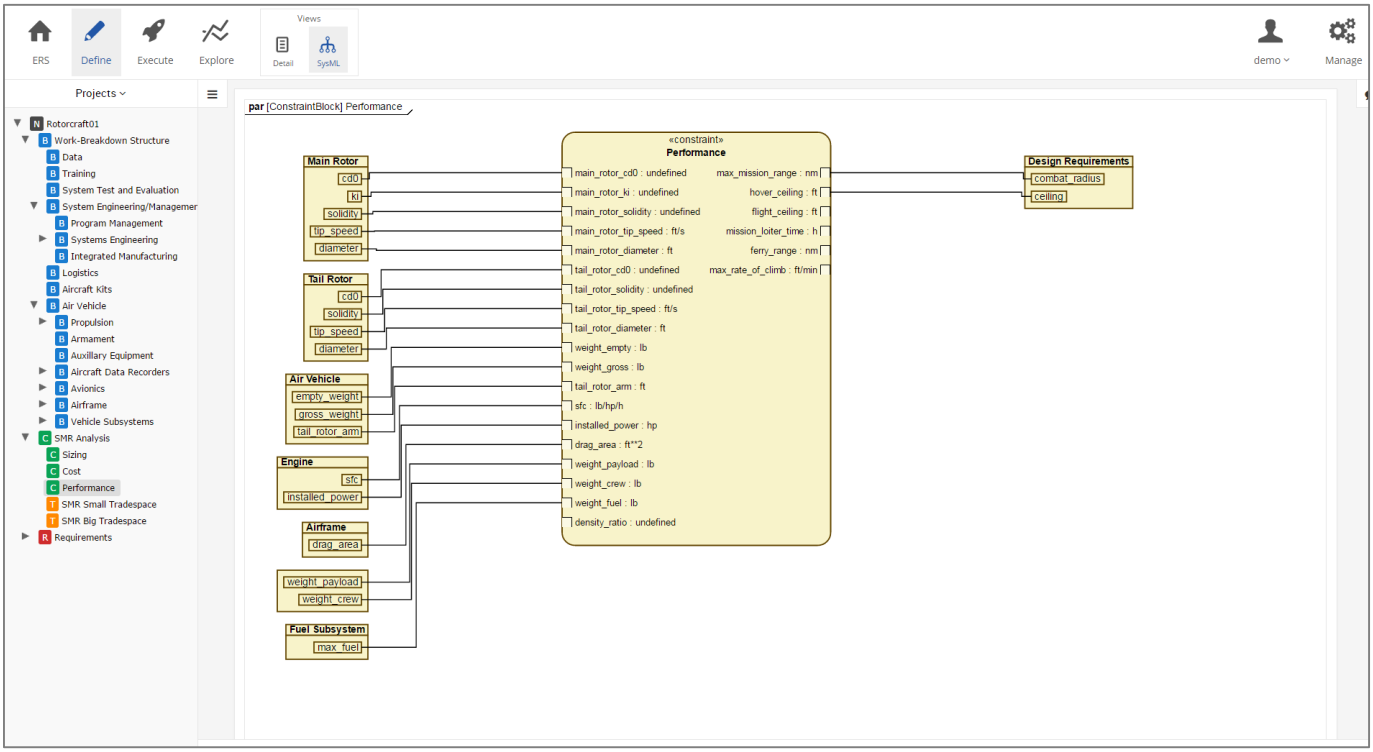


Figure 12. SysML Parametric Diagram

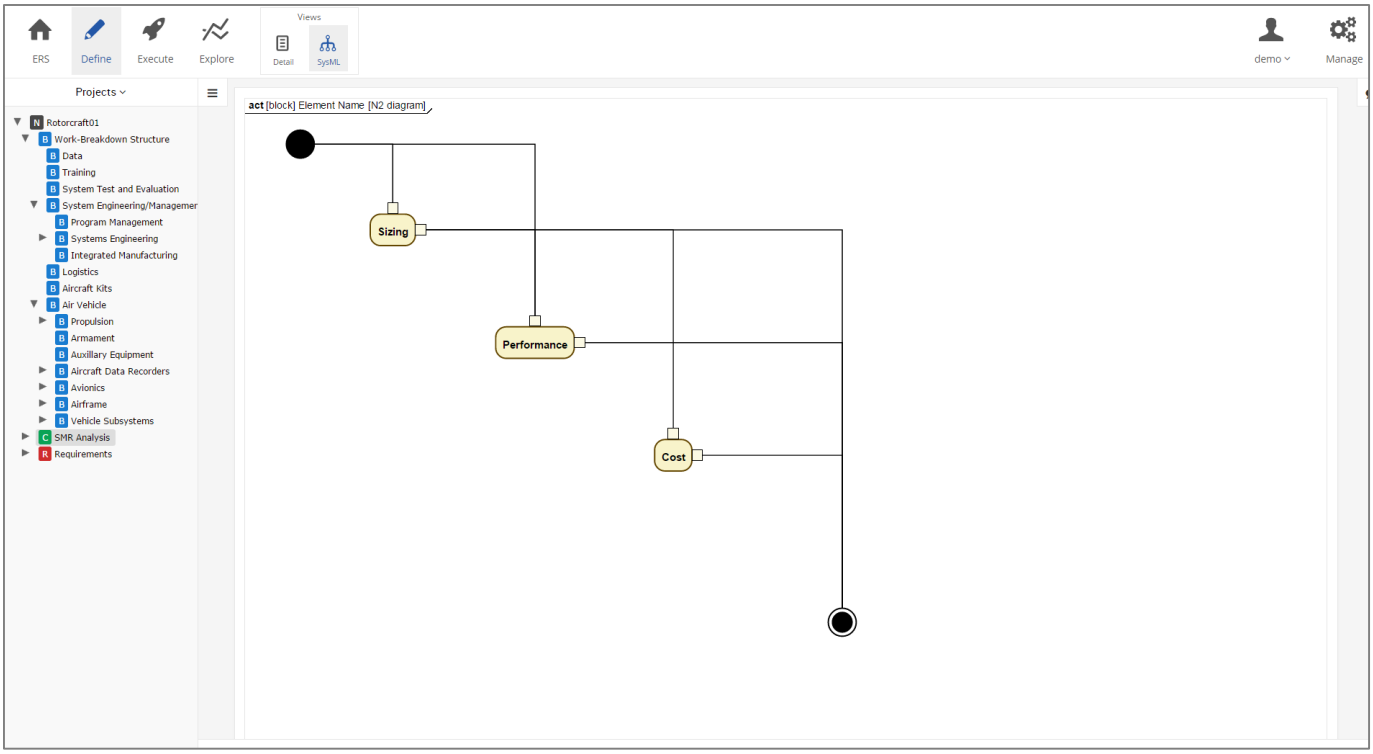


Figure 13. SysML Activity Diagram

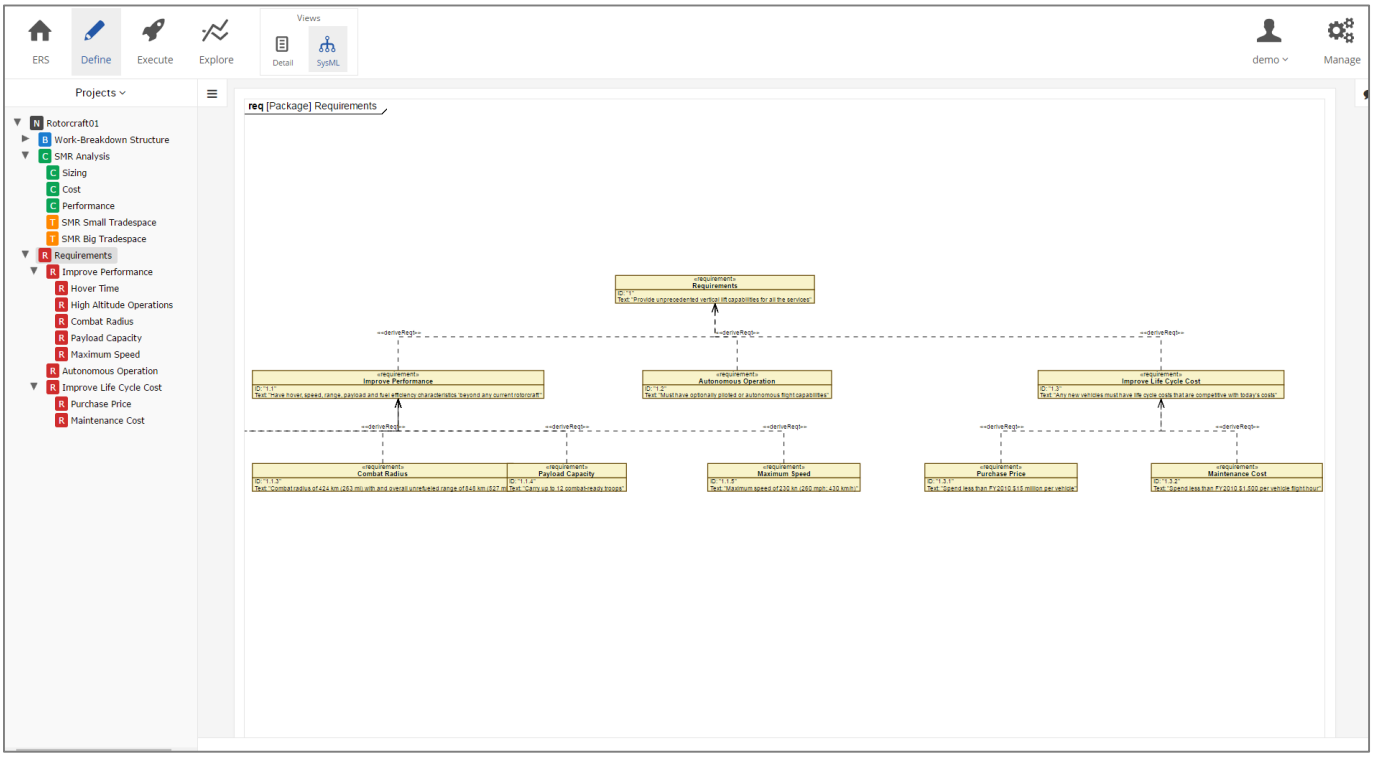


Figure 14 SysML Requirement Diagram

3.2.2 EXECUTE

In the Execute views, Figure 15, the System Engineer can curate previously run tradespace explorations or run a new study by using the declared constraints for the Parametric Diagrams in a larger orchestrated model based simulation. This larger model based simulation can be automatically generated based on the PAR diagrams which can be exercised using the Execute run interface. Current functionality allows for exploring connected constraint blocks using a Latin Hypercube Design of Experiments (DoE). Future capability will allow users to seamlessly execute Monte Carlo Simulations, other DoEs, and optimization techniques to drive the tradespace sampling.

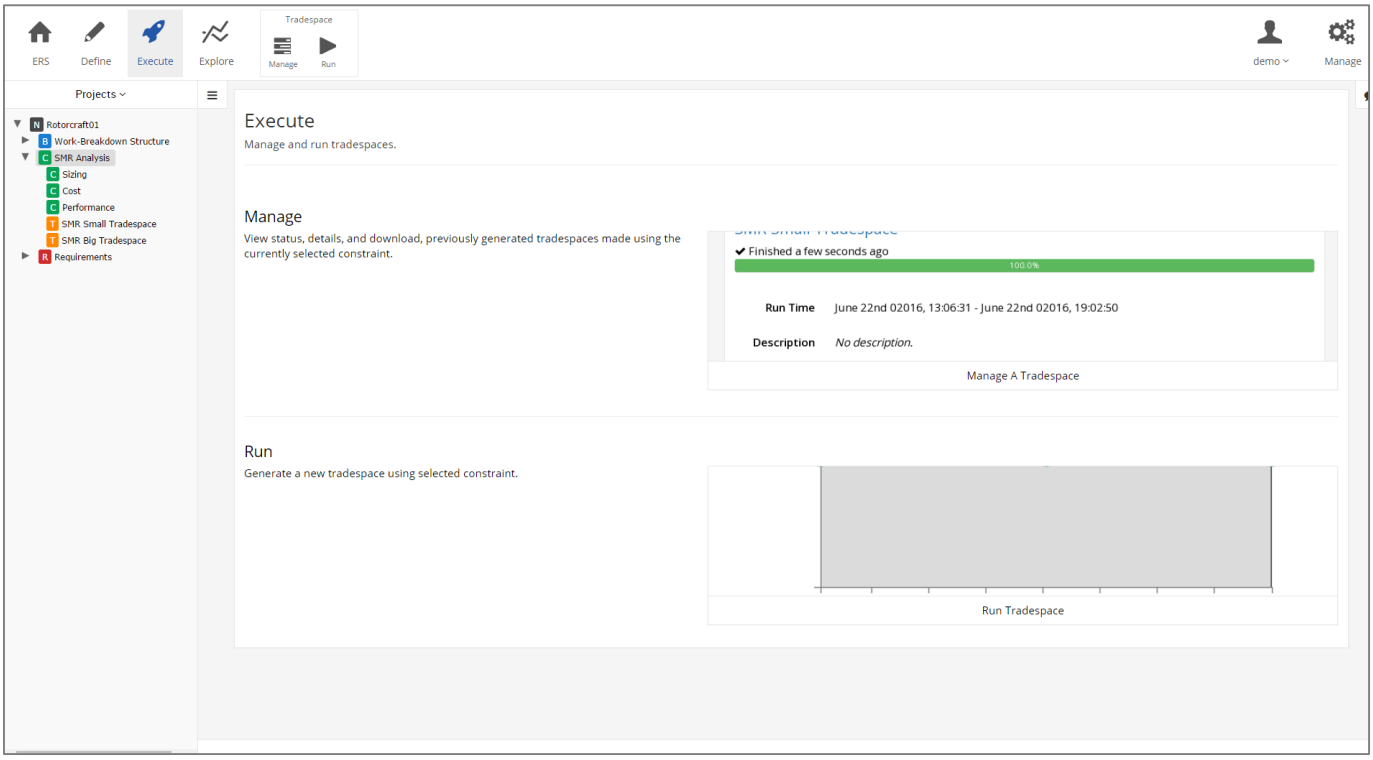


Figure 15. Execute View

Manage

For curating the different generated tradespaces, the user can get a management overview of the different studies previously completed or currently running on a per constraint basis. The user can then delve into each the tradespace's details including the variables and ranges used as well as the generation design method along with the number of cases completed or failed. Note failed cases can be due to violation of the constraints value properties lower and upper bounds e.g. cases that yield negative mass values. Additionally, the user can download a copy of the tradespace data as a csv file for use in other tools or delete unneeded tradespaces.

Run

When deciding to run a new exploration of a constraint the user can specify which subset of parameters from the constraints input value properties to vary and how. For example with the helicopter SMR Analysis, Figure 17, the Composite Cost Ratio and deciding to vary it uniformly between roughly 20,000 and 71,000 USB/lb. Constraint value properties not added to vary will to the be set to their default values.

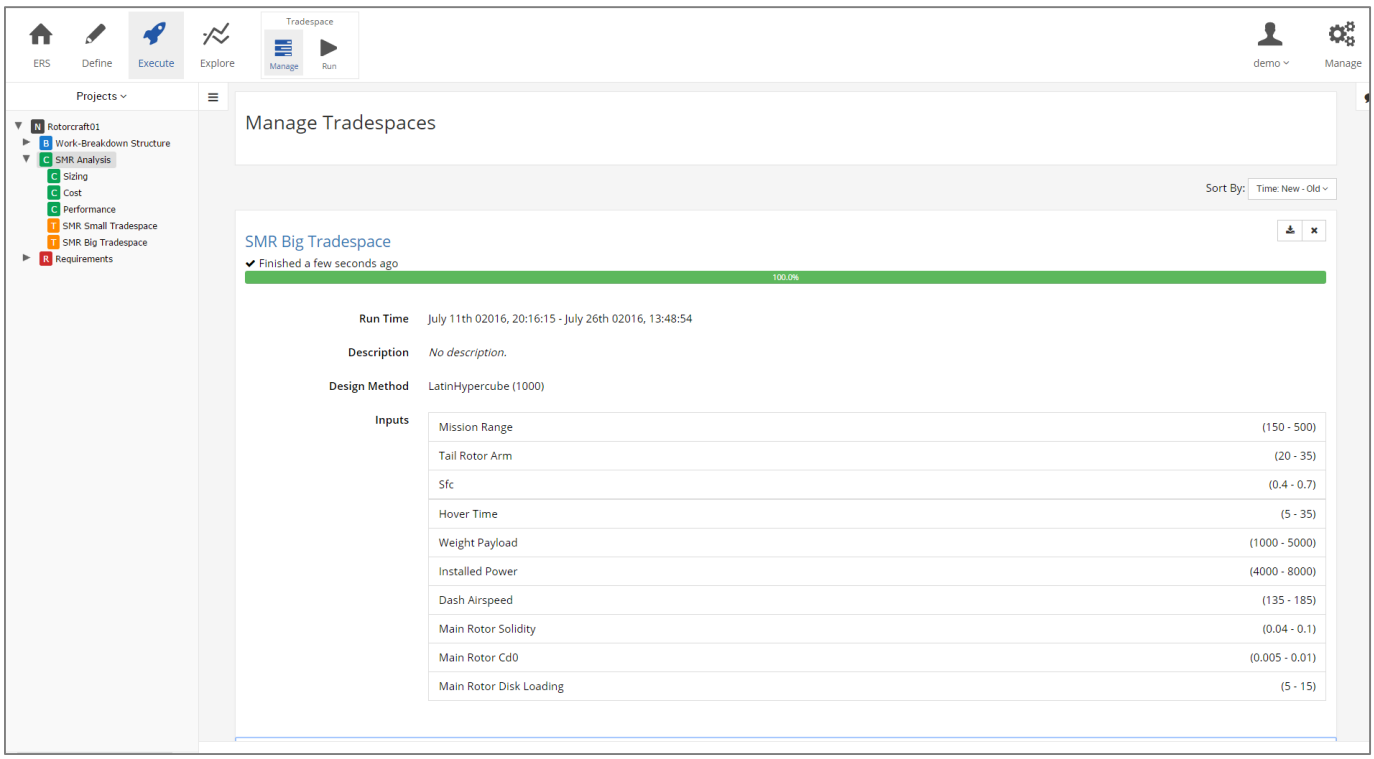


Figure 16. Manage View

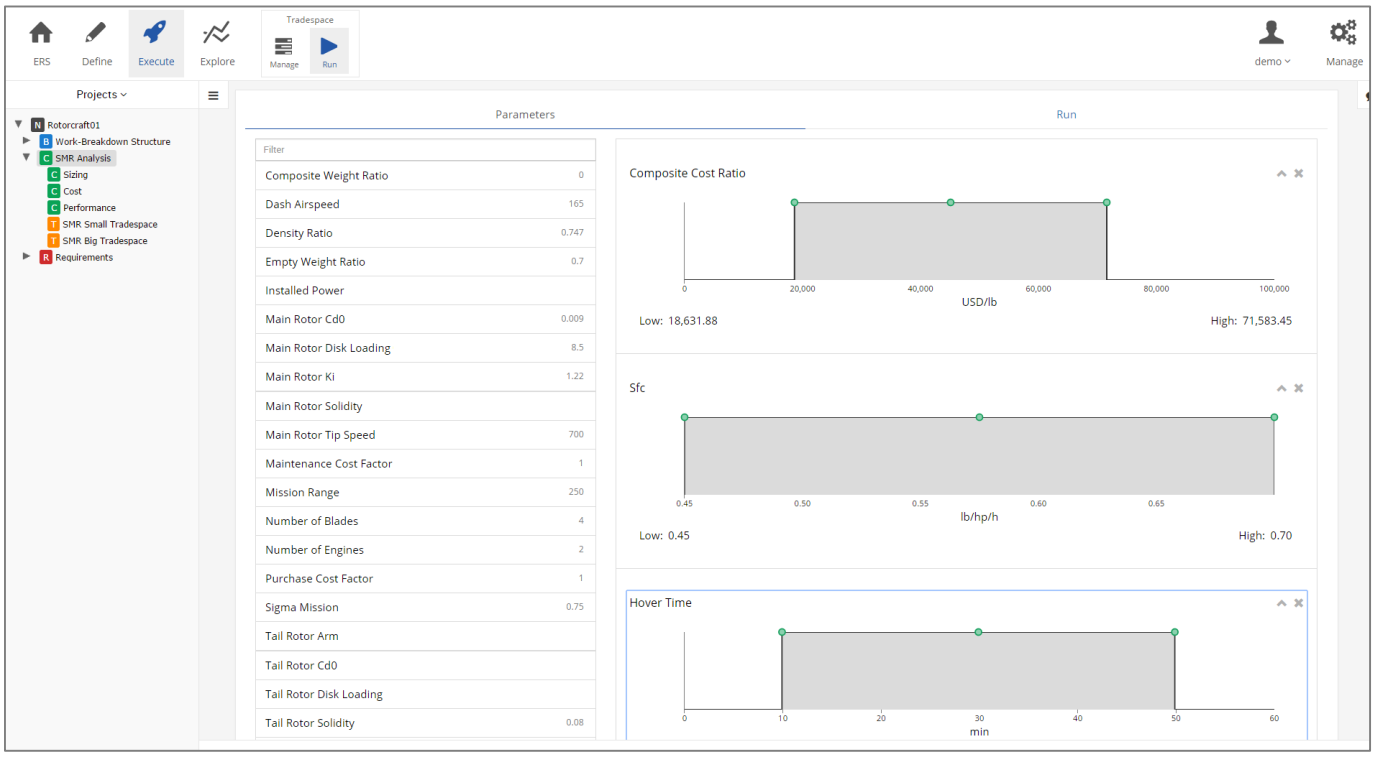


Figure 17. DoE Parameter Specification

After specifying which top level parameters to vary and their ranges the user can click on the `Run` tab at the top to specify the number of cases, design method, add description and change the name, Figure 18. Additionally the

worker can be changed between local and HPC. If the worker is left as local the job will be schedules on the local OpenMDAO job queue rather than the HPC. **The HPC worker will be discussed in Section 4** (ERSTAT Alignment)**.**

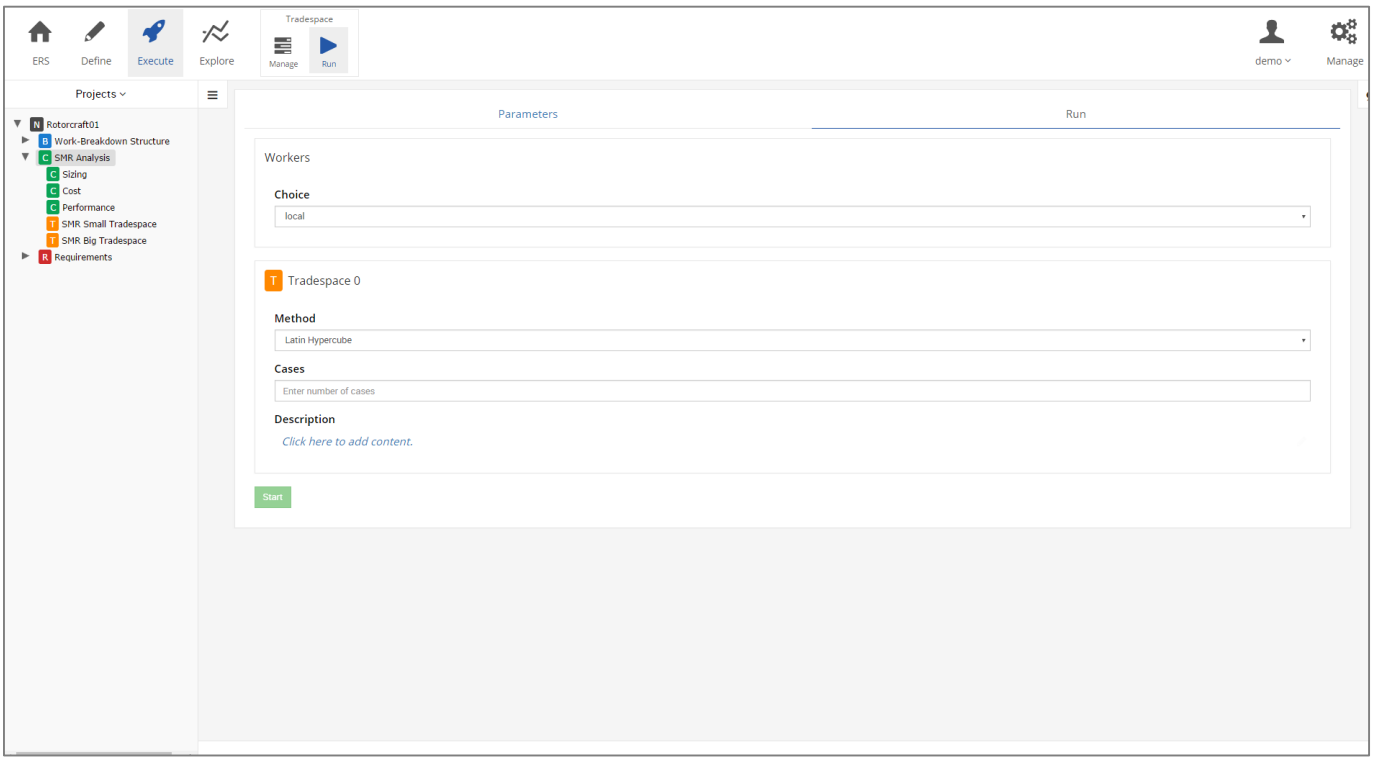


Figure 18. Execute Run View

Clicking on the start button will then queue the tradespace generation task with the current configuration on the specified worker. As the tradespace is running, status messages will be sent back to the Manage view for the current constraint. Once all the tradespace cases are finished being evaluated it can be selected in the Explore view to begin to filter and analyze the different alternatives.

3.2.3 EXPLORE

In the Explore views, the data generated by the tradespace generator using the Explore interface can be processed. Current functionality implements a customizable dashboard with coordinated views, Figure 19, so as the user brushes on one plot, that selection is reflected in the others. Additionally the value hierarchy from the Requirements specified from above provides a means to score the different system design alternatives. Future functionality will add other visualizations, e.g., Parallel Coordinates, Scatterplot matrices, and additional value transformations.

The core of the explore view is the same Analysis of Alternatives (AoA) module that was developed and integrated into ERS TradeStudio's *TradeAnalyze* and will be discussed in additional detail in Section 7 Analysis of Alternatives Module.

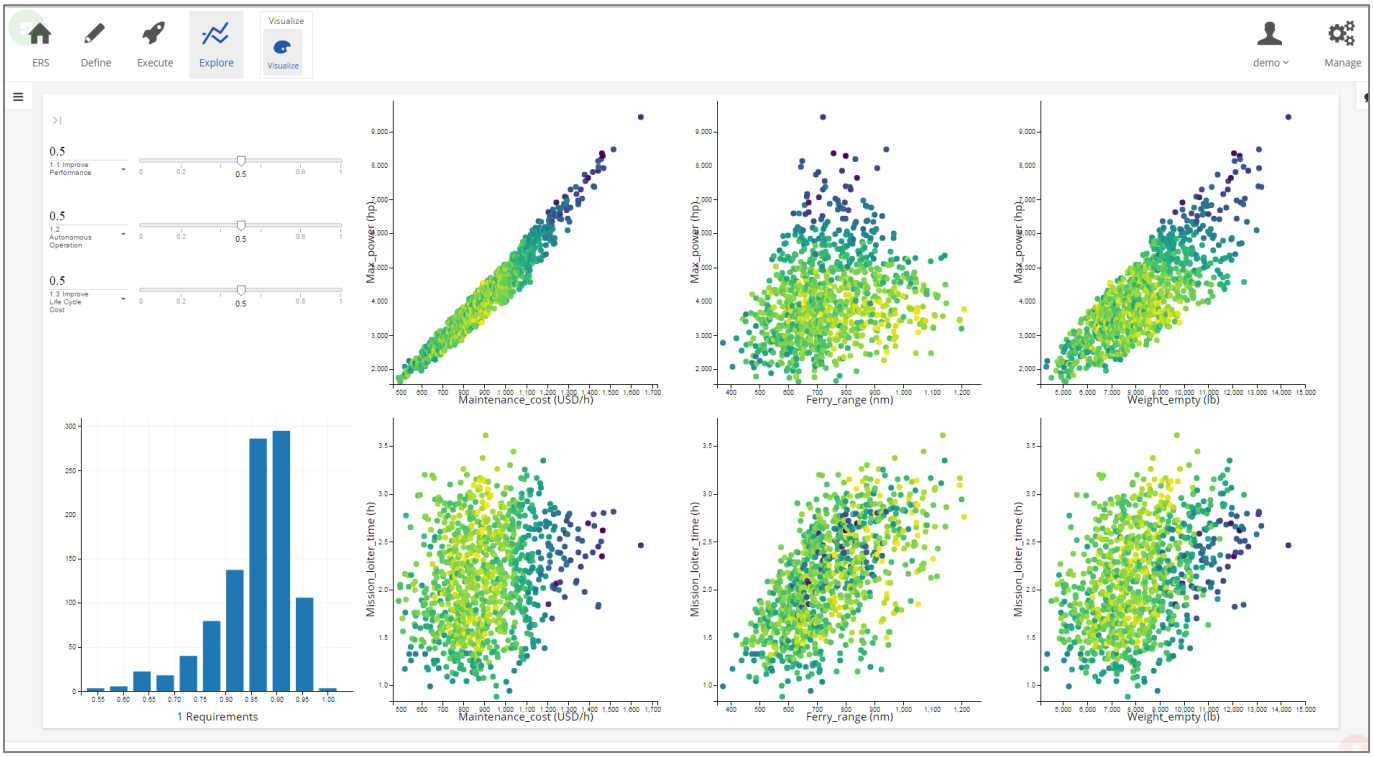


Figure 19. AoA Dashboard View

3.3 SOFTWARE ARCHITECTURE

To support the dynamic nature of the System Engineering workflow there need to be a variety of ways to interact with the core data model of the ERS TradeBuilder v1.4.1, Figure 20. One of largest change from the ERS TRADESPACE Tool v1.0 software architecture is the update of the IPython Notebook to the Jupyter Notebook and the associated JupyterHub project. JupyterHub is a multi-user server for Jupyter notebooks. When a user logs into JupyterHub a new docker container is spawned specified to that user. This helps to keep users sandboxed from the main server while allowing them the freedom of an open environment for performing functions essential for their work.

The other changes for TradeBuilder v1.4.1 (over v1.0) is better modularization of the different code libraries. The backend libraries have been packaged using the standard python packaging techniques so that they can be loaded with a `pip install` command. Frontend libraries have been updated to use the standard npm interfaces rather than the Bower method these JavaScript differences will be discussed more in Section 3.4 Components for TradeStudio Integration. By breaking apart the code into independently installable packages the implementations become more easily tested adding to the general robustness and maintainability of the codebase.

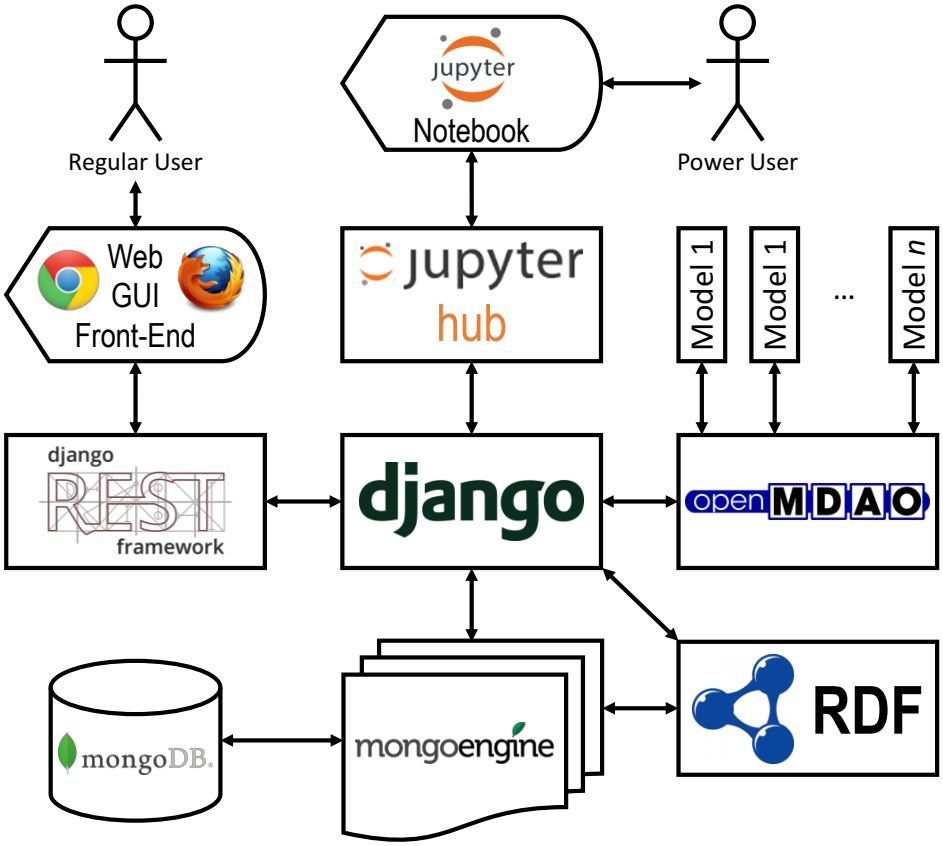


Figure 20. User Interactions and elements of the framework

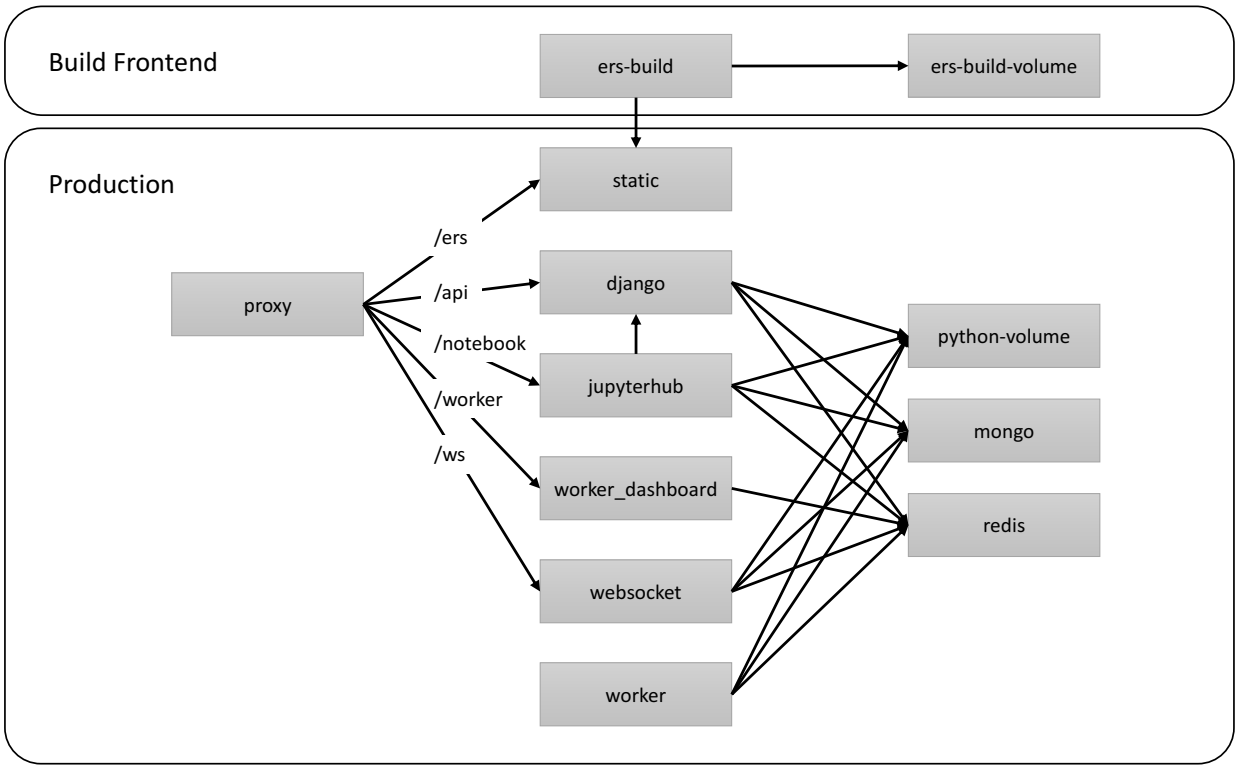


Figure 21. ERS Tradespace Tool Process and Docker Container Architecture

An overview of the containers in Figure 21 includes:

- **Proxy** – an apache proxy server to forward requests to the different services. One head proxy makes it easy to secure the multiple services behind a common host.
- **Static** – another apache server that serves the build ERS TradeBuilder tradespace tool HTML, JavaScript, CSS, and other assets.
- **ERS-build** – a temporary container that houses the build chain for processing the ERS Tradespace Tool frontend code into the static container
- **Django** – serves the REST API that the frontend application consumes. This REST API exposes the core data objects.
- **Jupyterhub** – to spin up and orchestrate user specific scripting environments (python, R…) for interacting directly with the python Systems Engineering data models.
- **Websocket** – pushed events and messages to ERS Tradespace Tool users to keep the model state in sync.
- **Worker Dashboard** – simple webpage to view the status of the workers and jobs that are processing the local job queue.
- **Worker** – processes OpenMDAO jobs that are locally queued.
- **Python-volume** – a volume that is mounted into the various containers so the same python code is used by the different services.
- **Mongo** – the MongoDB service for persisting the data
- **Redis** – used for message and event passing as well as queuing tradespace jobs.

3.4 COMPONENTS FOR TRADESTUDIO INTEGRATION

Paramount for TradeStudio Integration is the packaging of JavaScript modules into reusable component libraries. By keeping the salient component code modular it can more easily be tested and maintained. Additionally if the reusable component is packaged correctly the amount of additional work to use it in different frontend is lower. In the JavaScript Node environment there are a multitude of alternative package managers each with their own advantages and disadvantages.

JavaScript package management alternatives:

- **NPM** – is the native Node Package Manager (NPM) for Node based projects. The main drawback is its inability to specify component file dependencies beyond JavaScript without additional tooling like webpack or browserify.
- **Bower** – is a frontend asset package manager. The main benefit is bower offers a native way to specify all the file dependencies (css, html) and not just JavaScript. However the community is smaller than NPM and has some inherent tooling limitations like adding require statements to import JavaScript code.
- **Component** – another frontend package manager that still offers the ability to specify all types of file dependencies but has a much smaller user base.
- **JSPM** – is a package manager built for SystemJS and the Universal Module loader. It shows promise but has lower adoption and is newer than npm or bower.

In the previous phase (SERC RT-120), *Bower* was selected to package the different modules as all the different asset files could be specified. For this phase, the overarching TradeStudio suite moved to using only NPM to package the reusable JavaScript code to reduce the number of development dependencies i.e. not having to install Bower and its additional tooling. In addition to selected NPM, *browserify* was chosen to then crawl through all of the JavaScript libraries to create a single bundle for the browser to load and the other types of static assets were copied around using helper scripts. In the future, moving to a bundling solution that can process all of the asset types would help achieve more reusable frontend components.

4 ERSTAT ALIGNMENT

ERDC has been in development of the ERS Tadespace Analysis Tool (ERSTAT), led by Dr. Andrew Strelzoff. This toolset has been used in several rapid studies for programs across the Department of Defense, helping to drive the requirements for a more long-term solution. Therefore in addition to improving the capabilities of ERS, this task worked towards enhancing ERSTAT's capabilities by developing and integrating modules from ERS TradeBuilder.

This work initiated with two primary goals of the ERSTAT team, namely:

- Users need to be self-reliant when it comes to recurrent tasks, (e.g., defining and verifying parameters in a tradespace, define parameters ranges for a tradespace, running a tradespace).
- Users need to document their analysis process, obtain the results generated by ERSTAT, and be able to present the results in the necessary manner and context.

ERDC's ERSTAT team excels at providing unprecedented computational analysis capabilities to government teams. Their primary role consists in modernizing and parallelizing legacy analysis or design tools, installing them on HPC resources, exercising the tools, and supporting the teams as they evolve their projects. This task is intended to help the ERSTAT team focus on the initial portion of its efforts and be able to offload the more labor intensive portions associated with supporting the teams (primarily re-defining and re-running tradespaces).

The team recognized that there would be four use cases that the ERS TradeBuilder framework must support from a computation/collaboration perspective. These are illustrated in Figure 22 below. Use Case A concerns a local user, running the framework on a local VM and using the computational resources of its own personal computer. Use Case B involves a group of users accessing a webserver but limiting the execution of tradespaces to the CPUs on the webserver. Use Case C enhances Use Case B by allowing those users to reach out to an HPC resource and submitting tradespace execution jobs to it. Use Case D provides the same HPC access but for local independent users. The goal of Task 4 was to enhance the ERS TradeBuilder framework to support Use Cases C and D.

The first step in achieving this goal was ensuring the architecture selected for executing tradespace would scale when using High Performance Computing. The ERS TradeBuilder tool uses NASA's OpenMDAO as the orchestrator for coordinating the execution of analyses (e.g., performance models, cost models, campaign simulations). The team had to ensure that OpenMDAO would scale when distributing jobs to a cluster. For this purpose, the team tested the weak and strong scaling characteristics of an "embarrassingly parallel" problem that would distribute many individual runs of a computationally inexpensive model. The team used ERDC's ground vehicle model as the test case.

The strong scaling results proved that the problem scales linearly up to 100 computational cores (as evidenced by Figure 23), dropping the time required to analyze the problem from 8-10 minutes to under 10 seconds. The weak scaling analysis studied how many points could you analyze in a given amount of time. Again, the results proved satisfactory, as evidenced by the linear relationship (in log-log-scale) in Figure 24. The results proved that it is possible to run all possible combinations of the sample problem (approximately 1 billion) in under 2 minutes if 30,000 cores are used. However, it is important to note that this time did not account for the MPI setup time (which is approximately 10 to 15 minutes for 30,000 cores). The strong and weak scaling results proved that using OpenMDAO's MPI implementation would perform satisfactorily and gave the team the necessary confidence to develop the HPC extensions for the ERS TradeBuilder framework.

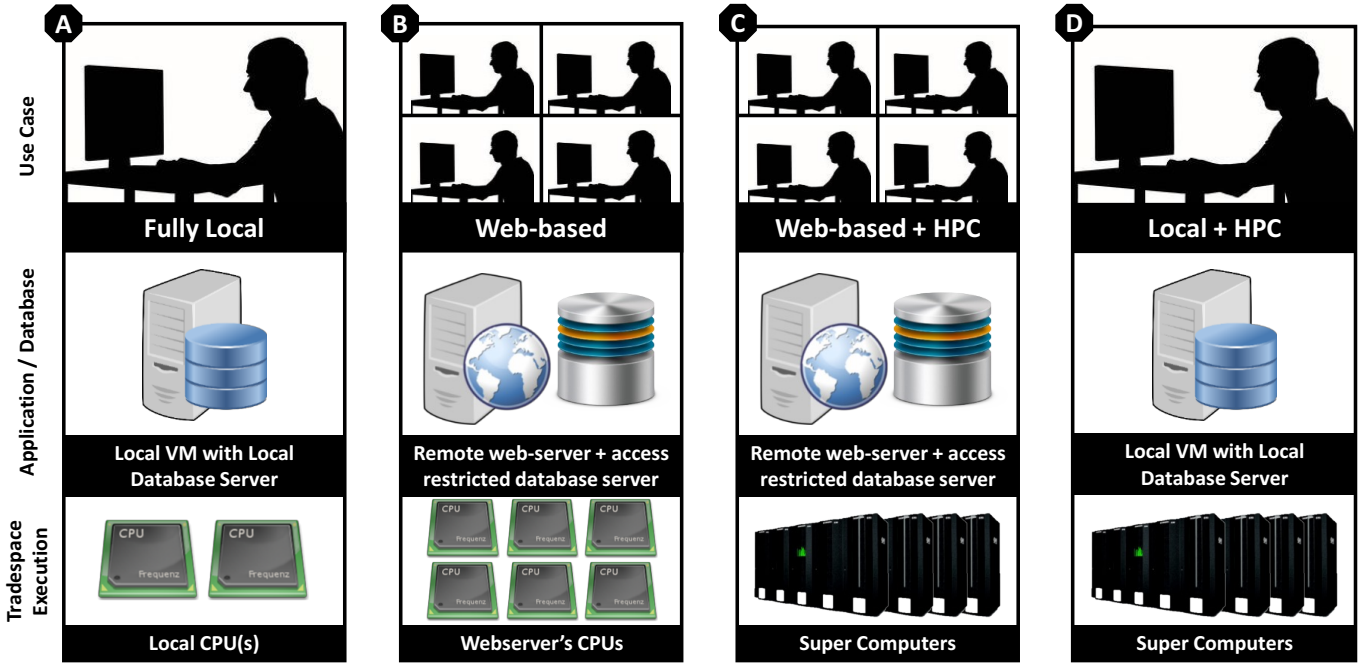


Figure 22. ERS TradeBuilder Framework Collaboration/Computation Use Cases

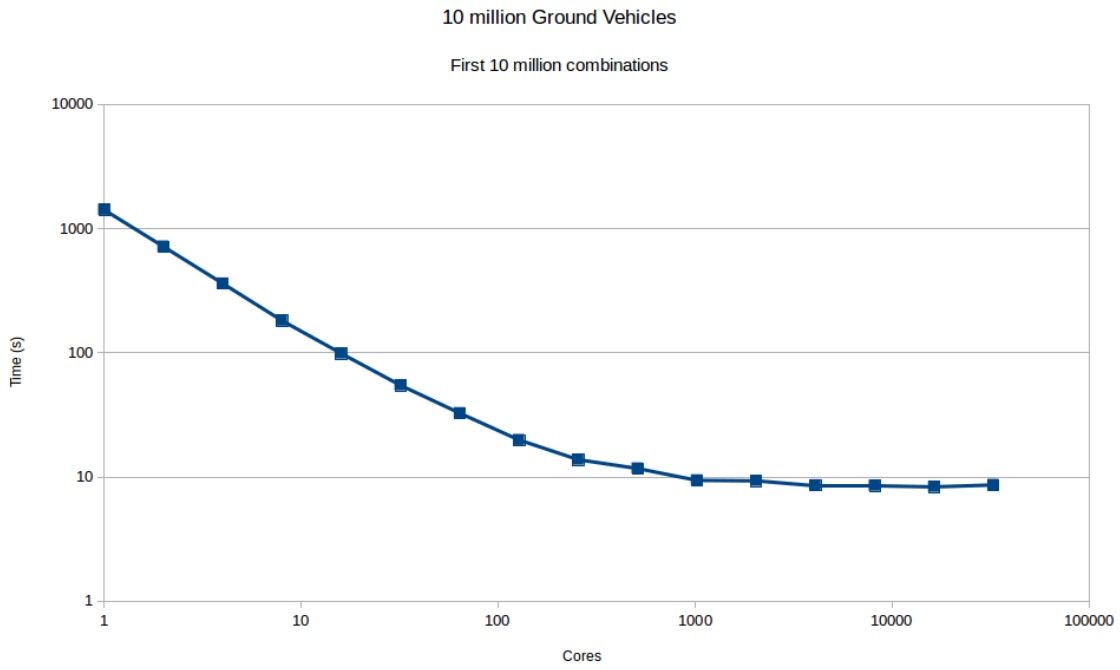


Figure 23. Run time required to analyze 10 million vehicles vs. number of computational cores (strong scaling analysis)

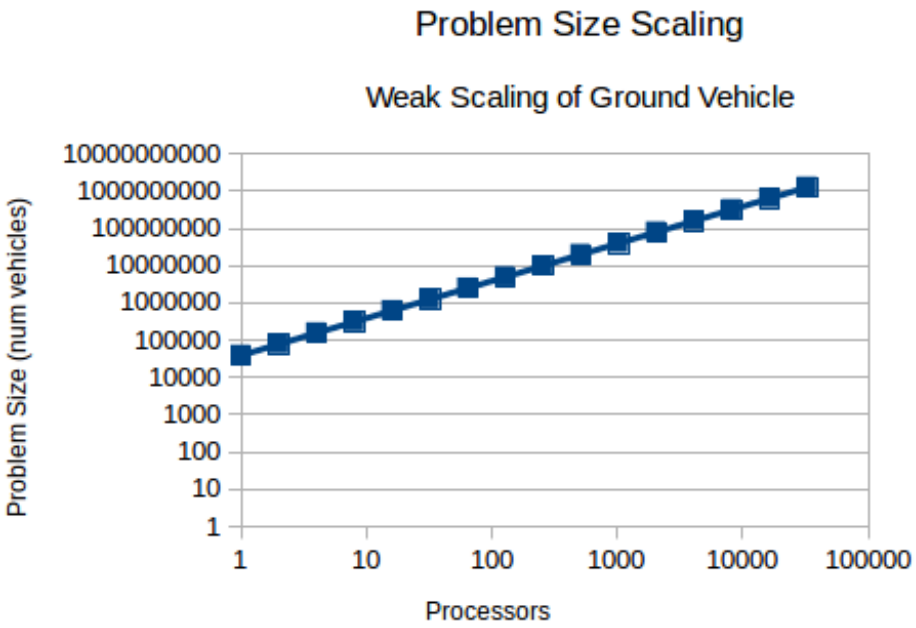


Figure 24. Number of vehicles analyzed as a function of processors (weak scaling analysis)

The ERS TradeBuilder HPC extension would enabled users to submit jobs to ERDC's High Performance Computing resources (namely Topaz[5]), monitor the execution of said jobs, and retrieve the results. The ERS TradeBuilder framework was extended (as shown in Figure 25) by creating a custom "worker" that connects to the sub submission queue (built on *Redis*). This worker interfaces with ERDC's HPC resources using ERDC's User Interface Toolkit (UIT).

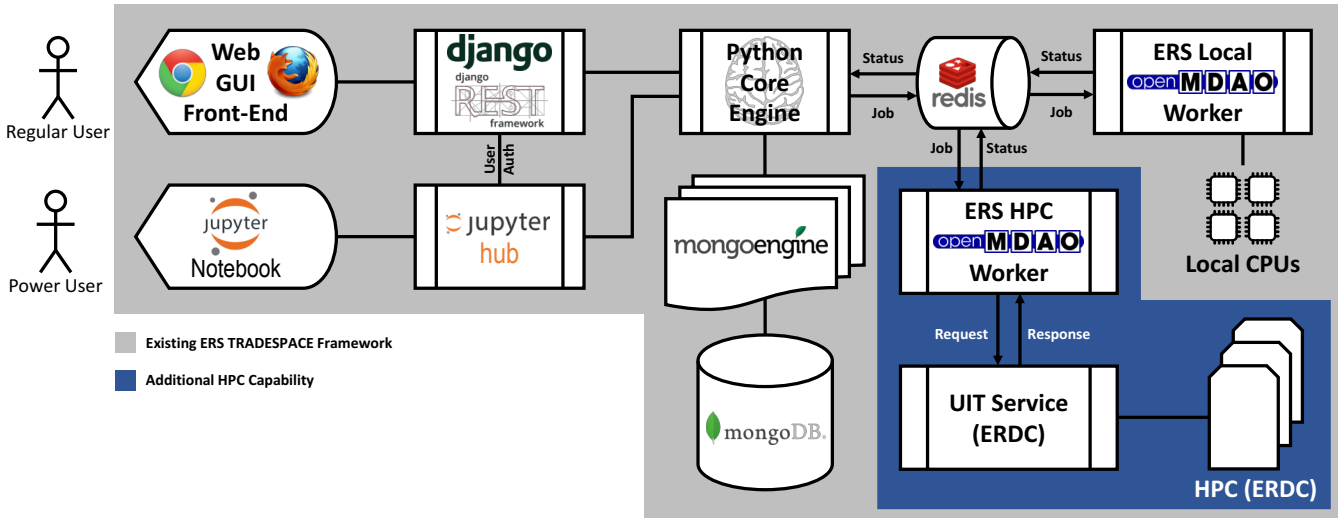


Figure 25. ERS TradeBuilder integration with ERDC's HPC resources

The ERS HPC worker follows the processes described in Figure 26. The process begins with a request from the user to utilize the HPC resources. The ERS TradeBuilder server then requests the user to provide the necessary information to login through UIT. At this time, the only method supported is the Kerberos DoD HPC OpenID with

[5] https://erdc.hpc.mil/hardware/index.html

Yubikey. In order to support CAC authentication, UIT would have to be updated. Once the user submits his/her username, password and the one-time password generated by the Yubikey, the ERS TradeBuilder server forwards the information to initiate the handshake with UIT. UIT replies with the session id, and a prompt. This prompt can either be a declined authentication or a request for the one-time password, in which case the ERS TradeBuilder forwards the one-time password to finalize the handshake. The UIT server then replies with a Kerberos token and a prompt that is forward to the client. The token is never stored nor held in memory on the server, only on the user's browser. With the token the user can submit requests to the HPC resources. The ERS TradeBuilder user interface allows for the submission of tradespace generation jobs, checking the status of a job, and retrieving the tradespace data to the ERS TradeBuilder server for further analysis. Finally, the user can log out of the service, which sends a "destroy token" command to UIT. As a precautionary measure, UIT expires tokens after 24 hrs.

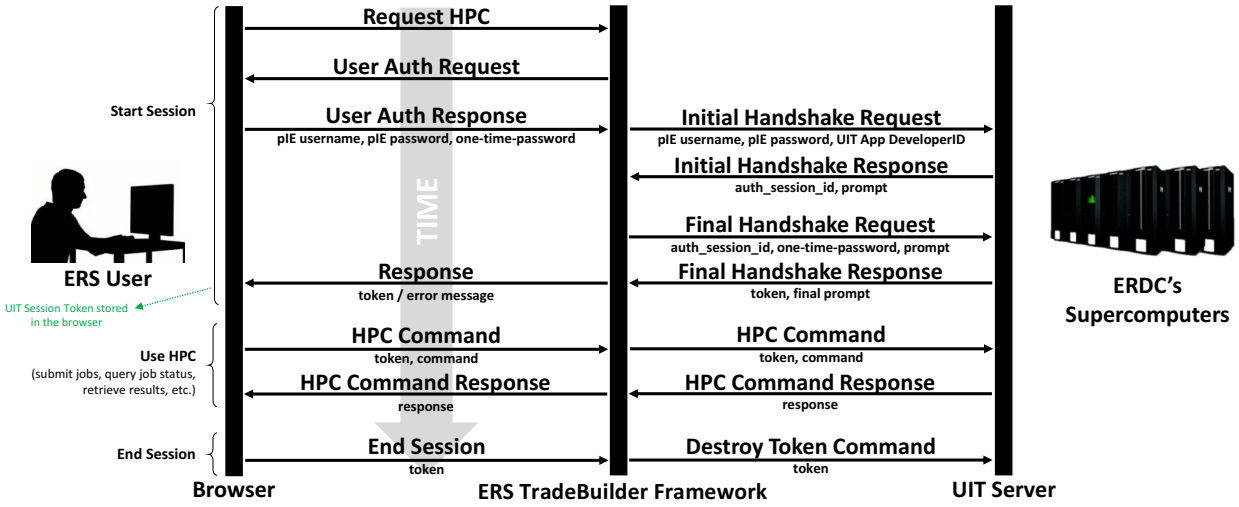


Figure 26. User interface for utilizing ERDC's supercomputers from ERS TradeBuilder

The work conducted under the ERSTAT alignment manifested itself on the ERS TradeBuilder tool in the Execute page, under the "Run" tab's "workers" choice (Figure 27). There are two options to run tradespace, either 'local' or 'hpc'. Local uses the CPUs on the ERS TradeBuilder server. When the user selects 'hpc', they are prompted to login using their DoD HPC OpenID. The information required is the principal account login (the field is labeled 'Principal' in order to keep it consistent with other DoD HPC OpenID systems), the Kerberos password and the one time token generated by the Yubikey.

If the authentication is successful, the user will be presented with the UIT Kerberos prompt that will indicate when his or her password will expire, a list of hosts to connect to (by default the system selects topaz01). At the time of this writing, only a few accounts on Topaz had been configured to receive jobs. The user can then select the account (which shows as blank on Figure 28 because the user had no access to an HPC account when the screen capture was taken). The user can also select the queue to submit the job to, the number of processors they'd like to reserve and the number of hours the reservation should be for after the job is started. It is important to note that if the user makes a reservation for 1 hour, but the job takes longer to run, the system will forcefully end the running processes after an hour has elapsed. It is critical that the reservation time is estimated correctly.

The current implementation requires that all executable code, e.g., OpenMDAO components, is also installed on the user's account in the system they are submitting the job to. It is not possible to upload an OpenMDAO component to the ERS TradeBuilder server and try to execute it on the HPC resources. The one exception to this is the "function" which uses OpenMDAO's *ExecComp* to dynamically generate an executable component. The team is currently evaluating other means to be able to submit the python executable code with the tradespace job submission to minimize the amount of effort on the users' part. Conversely, it is prudent to require users to go through a vetting process for submitting their models for review before executing them on the HPC systems.

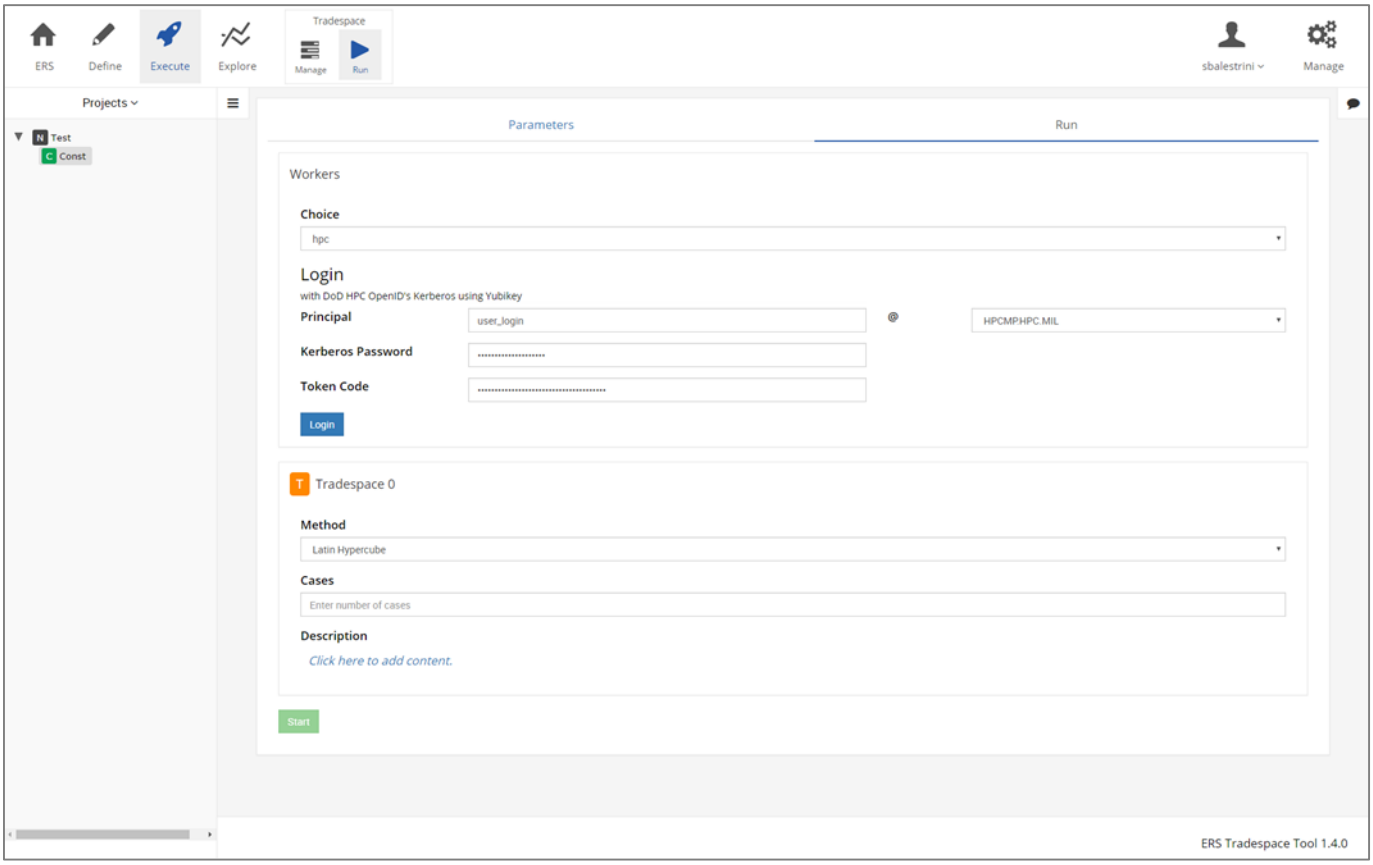


Figure 27. HPC Access Via the Execute Page

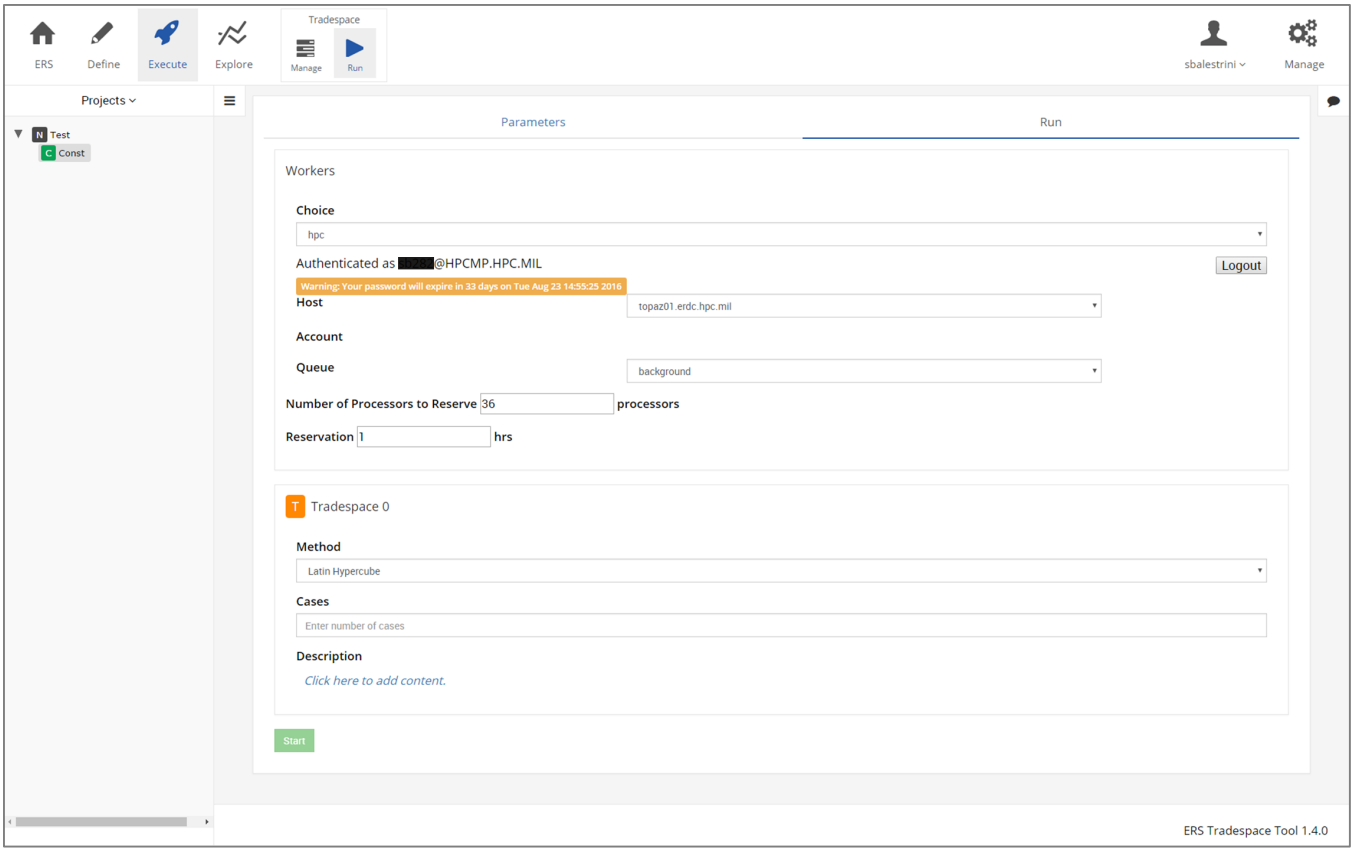


Figure 28. Screen capture of the HPC interface after successful authentication

Under this effort, the team also provided assistance to the ERSTAT team with the use of analysis tools, primarily NASA's OpenVSP[6]. The work consisted in developing a means to integrate OpenVSP and the Jupyter Notebook, e.g., an interface for OpenVSP, a visualizer for the 3D models in the notebook. The majority of these support efforts were focused on methods to compile the Python interface for OpenVSP on a Linux environment and writing an improved pythonic interface built atop the basic OpenVSP one to facilitate the interaction with the models. A JavaScript STL viewer was extended to allow users to visualize the 3D models generated by OpenVSP inside the notebook. Future work could further develop the Python interface to OpenVSP to facilitate manipulation and visualization of the models in real-time. The GTRI team has reached out to the OpenVSP development team who are interested in having someone extend their python interface.

In summary, the new HPC interface in the ERS TradeBuilder framework will enable users that have the analyses setup on the HPC to modify and execute their own tradespaces, enabling the ERSTAT team to focus on what they do best without having to provide day-to-day support to its users.

Future work could further extend this by allowing capable users to submit execution runs to the HPC that do not need to be pre-loaded into the HPC resources. The interface can also be extended to provide guidance to users as to the time it may take to execute a particular tradespace, minimizing the overestimation necessary to ensure the job is not cut off prematurely.

[6] http://www.openvsp.org/

A twofold approach for building systems engineering modules was utilized, with the research and development of both *R* (statistical programming language) and *Python* analytical tools. In pursuant of the original proposed concept for system analysis modules, the ability to create a notebook that accepts R language coding was included in the Jupyter Hub/Notebook tools. Methods were developed to allow the reading of the *HDF5* files created by TradeBuilder runs into R, allowing notebook users to perform R-based analysis on the tradespace data.

In order to better integrate the tradespace analysis with the front-end framework, follow on efforts were transitioned to supporting tradespace analysis using *Python*, as it would allow for front-end modules to interact with the analysis modules through the ReST API more seamlessly. Using Python for system engineering analysis keeps these modules consistent with the rest of the backend *Python* development, helping to ensure a smooth integration with existing Tradespace capabilities. *Python* syntax is also the basis for experienced users to develop projects in the Jupyter Hub/Notebook environment, and using *Python* to utilize systems engineering analysis modules would provide continuity for the ERS Tradespace user. Moreover, a number of existing *Python* modules, (e.g. Numpy, SciPy, Pandas, Statsmodels, etc.) provide an existing, proven framework of tools consistent with those available in *R*.

The following core systems engineering analysis modules and functionality was completed during this development phase:

Design of Experiments:

- The core design of experiments methods was augmented with new experiment types, including Full Factorial, a Plackett-Burman screening design, and a Monte Carlo Simulation (only currently accessible through the notebooks).
- The capability to link and specifically purpose (optimization, surrogate model fitting, validation, etc.) multiple experiments to a Tradespace was developed.
- All case sampling experiments were implemented as using an OpenMDAO driver capable of parallel runs in conjunction with the added HPC capability.

Optimization:

- The ability to drive constraint sampling with optimization based methods was integrated into the Tradespace backend.
- A Non-dominated Sorting Genetic Algorithm (NSGA-II) was integrated into TradeBuilder for multi-objective optimization of constraint(s).
- A module was implemented to allow a suite of single objective optimization methods, including many gradient-based (analytical and numerical) methods. This is based on the SciPy optimization module.
- Other optimization libraries could be integrated with reduced effort if they have existing OpenMDAO plugins (e.g., DAKOTA's pydakdriver).

Surrogate Modeling:

- A simple correlation tool was generated to indicate statistical correlation between generated data columns and provide a rough guide to surrogate model generation.
- A core module was developed to allow for flexible generation of surrogate models based on data generated by Tradespace. Three core surrogate modeling techniques were made available:
    - Ordinary Linear Regression – allows for regression and surrogate modeling based on Response Surface Methods. (based on a Statsmodels module)

    - Artificial Neural Networks – allows for training of simple neural networks to be used as surrogate models. (based on the PyBrain module)
    - Kriging – allows for Gaussian Process-based Kriging surrogate models to be generated and trained. (based on a Scikit-Learn module)
- Surrogate models were expanded to include the ability to perform a preliminary suite of numerical goodness of fit and validation tests. The implemented tests include R^2, Adjusted-R^2, Root Mean Squared Error (RMSE), and normalized RMSE. These can be associated with any single experiment dataset, or any combination of datasets.

A number of ongoing research and development efforts are also still underway on this task, with an eye to future integration into the formal toolset:

- The ability to generate and run new constraints based on fitted surrogate models (as OpenMDAO components) to quickly create new data has been developed and is undergoing integration into the larger toolset.
- A "batch mode" that allows surrogate model-based constraints (and other capable OpenMDAO components) to calculate large batches of experiment points at once has been developed and is undergoing integration into the larger toolset. This can serve to reduce overhead, and facilitate quicker calculation of large numbers of design points using only local/server resources.
- The ability to generate surrogate models with the tool frontend is under development.
- Automated surrogate modeling has been researched and explored with the goal of creating a multi-fidelity approach to surrogate generation for users with varying degrees of experience. A number of indicators (parameter type/count, runtime environment, response nature, etc.) could be gathered from the Tradespace to guide the development of surrogate modeling.
- Advanced topics for surrogate modeling are also being explored with the desire to integrate both active learning to more efficiently sample data for surrogate model generation, and multi-surrogates for combining multiple models to generate the most accurate representation of complex constraints.

6 SysML Maturation Module

The SysML module in ERS TradeBuilder acts as the primary interface for viewing and interacting with the system specification in the form of requirements diagrams, work breakdown structures, and constraints. The SysML capability of the ERS Tradespace Tools v1.0 (SERC RT-120) used front-end JavaScript libraries and technologies that have been deprecated since the v1.0 release. The primary, open source JointJS library used to render the SVG elements in the original SysML tool was commercialized and support was dropped for modern browsers. Additionally, the previous SysML code was tightly integrated with the front-end ERS Tradespace Toolset code in a way that made capabilities difficult to extend and made the SysML module unusable in application other than the tradespace toolset itself. Therefore, work on the SysML module for this effort, and released under the ERS TradeBuilder v1.4.1 banner, primarily focused on refactoring the code base to incorporate the latest JavaScript libraries for SVG rendering, as well as making the SysML tool modular and independent of the TradeBuilder front-end so that it may be used across other web-based applications.

The elements of the SysML specification that are useful for system modeling in ERS TradeBuilder were identified and prioritized according to the needs of ERDC. The SysML specification features currently included in the TradeBuilder consist of relevant elements that enable the use of Block Definition, Parametric, Requirement, and Activity diagrams. Table 1 lists the elements utilized in TradeBuilder v1.4.1 from the SysML v1.4[7] specification.

Table 1. SysML Specification Features Included in TradeBuilder v1.4.1

Diagram	Element	Description
Block Definition	Block Node	The Block is the fundamental modular unit for describing system structure in SysML
Block Definition	Composite Association Path	A composite association path relates a whole to its parts
Parametric	Constraint Parameter Node	A constraint parameter is a special kind of property that is used to bind parameters
Parametric	Value Binding Path	Binding connectors connect constraint parameters to each other and to value properties
Requirement	Requirement Node	A requirement specifies a capability or condition that must be satisfied, a function that a system must perform, or a performance condition a system must achieve
Requirement	Derivation Path	A derive relationship occurs between a source requirement and a derived requirement, based on analysis of the source requirement
Activity	Primitive Action Node	Primitive actions include object access/update/manipulation and value actions
Activity	Initial Node	When an activity starts executing a control token is placed on each initial node in the activity
Activity	Activity Final Node	When a control or object token reaches an activity final node during the execution of an activity, the execution terminates
Activity	Object Flow Path	Object flows connect inputs and outputs

The JavaScript library developed to render the SysML diagrams and elements for the TradeBuilder SysML Module has been designed to be extensible and to facilitate the addition of more elements from the SysML v1.4 specification in the future. As part of the process of prioritizing SysML specification features needed by ERDC for ERS, a roadmap was created to guide future development of the SysML module. The roadmap groups and prioritizes SysML specification features for future releases of TradeBuilder. The roadmap and required/desired specification features, outlined in Table 2, include two new diagram types that would be useful for system

[7] http://www.omg.org/spec/SysML/1.4/

modeling in TradeBuilder, the Internal Block and Use Case diagrams, as well as several features and elements in Block Definition, Requirement, and Activity diagrams that were desired, but not required, for the most recent release of the TradeBuilder v1.4.1.

Table 2. Roadmap for Required and Desired SysML Features

Diagram	Feature	Desired/Required	Release (Version Number corresponds to SysML Module)
Block Definition	Interface Compartments for Block Node	Required	v0.2
	Reference Association Path	Required	v0.2
	Multiplicity	Required	v0.2
	Actor Node	Desired	v0.2
	Association Block Path and Node	Desired	v0.2
	Generalization Path	Desired	v0.2
Internal Block	Part Node	Required	v0.2
	Connector Path	Required	v0.2
	Connector Property Path and Node	Desired	v0.3
Requirement	Containment Path	Required	v0.2
	Satisfaction Path	Required	v0.2
	Package Node	Desired	v0.2
	Refinement Path	Desired	v0.2
	Trace Path	Desired	v0.2
Activity	Merge Node	Required	v0.2
	Decision Node	Required	v0.2
	Join Node	Required	v0.2
	Fork Node	Required	v0.2
	Flow Final Node	Required	v0.2
	Control Flow Path	Required	v0.2
Use Case	Actor Node	Desired	v0.3
	Use Case Node	Desired	v0.3
	Subject Node	Desired	v0.3
	Association Path	Desired	v0.3
	Extension Path	Desired	v0.3
	Inclusion Path	Desired	v0.3
	Generalization Path	Desired	v0.3
Miscellaneous	Allocation Path	Desired	v0.3
	Comments and Notes	Desired	v0.3

In addition to a new diagram and new specification features, future development of the SysML module will also include an effort to improve the user experience by allowing the user to save multiple versions of diagrams, and enabling users to view multiple diagrams at the same time within the SysML module. Currently, diagrams are automatically generated using database calls and layout algorithms. Future work will allow users to not only automatically generate diagrams based on the system architecture, but to also modify the diagram layouts and have those changes saved. For example, a BDD could be collapsed to show only one branch of the system work breakdown structure, and then saved as a new diagram that could be accessed by other uses and further modified. One key feature that will enable this functionality is a tool bar, or menu, for the SysML module that will enhance the user interface to make editing or viewing diagrams an intuitive and seamless experience. Interlinking

the diagrams in such a way that users can switch diagram views by clicking elements (for example, clicking a Constraint Block in a Block Definition Diagram to access the corresponding Parametric Diagram) will also help to enhance the user experience in future versions of the tool.

7 ANALYSIS OF ALTERNATIVES MODULE

The Analysis of Alternative (AoA) module required developing a tool that could be integrated into TradeAnalyzer (formerly referred to as TradeStudio, which is now the term for the umbrella suite which includes TradeBuilder, developed by GTRI, and TradeAnalyzer, developed by ERDC). TradeAnalyzer should allow users to perform an analysis of alternatives, which as it became clear through the execution of this contract, can mean a variety of things. It can be argued that the entire ERS TradeBuilder tool is intended to perform (as one of its roles) analysis of alternatives. The first order of business was developing an AoA Roadmap, which was submitted as an interim report. This roadmap specified a series of capabilities required to perform an AoA, which were used to guide the development of the AoA module. The module was then integrated into both TradeAnalyzer and TradeBuilder tools.

7.1 ANALYSIS OF ALTERNATIVES ROADMAP

At its core, an AoA is an evaluation of potential alternative solutions according to their performance, system characteristics, or other criteria deemed highly important to key stakeholders alongside the estimated costs and resources required to develop and field such a solution. An AoA may lead to insights, high-level solution decisions, or refinements as part of an informal or formally documented evaluation. In the DoD an AoA is a formally required part of the Defense Acquisition System (DAS) process with significant levels of associated planning, study, and oversight. The 'Analysis of Alternatives Roadmap' discusses how an AoA, both as an informal supporting tool and a formal mandate, can mature to more effectively address the right requirements questions needed by senior decision makers. (Sitterle, Balestrini-Robinson, Freeman, Ender, 2016)

Every AoA will be different in terms of nuances, context, available information, etc. for a given system being evaluated at a specific point in the Defense acquisition system process. Consequently, there will be no one-size-fits-all methodology applicable to all AoAs. The roadmap report discusses major concepts vital to maturing AoAs to successfully support DoD needs and the goals of ERS relating to resilient design and a resilient design process. It explains why these aspects are necessary and how they will support materiel development decisions.

Through the roadmap, we sought to elucidate on future directions that are needed to meet the goals of AoAs in support of highly dynamic and complex systems engineering problems. The early stages of complex defense system design are characterized by immature knowledge. Design concepts may still be somewhat nebulous, new technologies underlying these designs may still be in development, requirements have yet to be solidified with respect to compromises that must be made across operational needs, stakeholders, and design tradeoffs, etc. Many AoAs of varying levels with respect breadth across design alternatives and operational System of System (SoS) considerations and depth with respect to level of detail and available data for specific potential design architectures will take place prior to defining the study plan for the Milestone A AoA of the DoD Acquisition process.

AoAs are often conducted in a large part through exploration of a tradespace. A tradespace is defined as the complete enumeration of the system alternative design variables together with the set of program and system parameters, attributes, and characteristics required to satisfy performance standards associated with each system alternative. A tradespace is, in effect, the complete solution space. Using a matrix analogy, each row represents a specific design alternative with its associated characteristics and performance measures. Each column then represents a specific design variable/ system characteristic or performance measure either specified (i.e., design variables) or evaluated (i.e., performance measures) for each alternative. Each entry in this matrix need not be singular, but could be an array or other more complicated representation of data.

So, it follows that to explore a tradespace as part of an AoA, one must first generate that tradespace. The roadmap discusses near and longer term priorities that address the end-to-end problem:

- synthesizing requirements,
- system and problem specification,
- transformation through a suite of various locally- and non-locally hosted M&S components,
- setting the conditions for and managing the execution of the integrated models in order to create a tradespace, and
- mapping the output measures on which tradeoffs will be assessed to stakeholder needs through a dynamic decision analysis well-tied to the metadata.

This perspective brings model-based systems engineering (MBSE), multidisciplinary design analysis and optimization (MDAO), and multi-criteria decision making/ analysis (MCDM /A) together to enable the exploration of a larger number of design concepts, variations within designs, operational variations, mission threads, etc. Together, this synthesis will enable decision makers to understand the relationships across projected capabilities, costs, risks, and compromises that may be necessary to best facilitate achievement of system (and operational) objectives. It is important to note that not all insights with respect to measures and metrics within an AoA will be directly quantifiable and sortable. Similarly, the dynamic narrative for data (tradespace) exploration and analysis becomes critical to understanding the insights gleaned. Any one exploration or analysis path may produce similar or completely different insights from another. To realize a decision-centric AoA, we therefore need to be able to capture and communicate the "Living Narrative" of the exploration or analysis process.

There are many concepts throughout the roadmap that inter-relate and overlap at various stages of the AoA process. While they are in some cases inseparable, despite being described in different sections, they cannot all be implemented at once. This roadmap therefore characterizes challenges facing AoAs supporting defense system development as near and longer-term priorities, even though the distinctions when synthesizing these concepts to provide implemented capabilities will not be as cleanly decomposable. A high-level view of these priorities is shown in Figure 29.

The guiding belief underlying the AoA roadmap is that design of complex systems for mission critical applications requires an integrated design and analysis process focused on identifying the set of alternatives most likely to meet requirements based on the information (data, design architectures, models, etc.) available. The concepts of multiple architectures, operational scenarios, stakeholder perspectives, SoS views, and a traceable tie-in to requirements (that promotes their evaluation and maturation) all seek to address the complexity of analysis needed while recognizing that being all-encompassing is not feasible. Throughout the roadmap, we adhere to the philosophy that the most important goal is insight, not numerical treatment or inference. In decision analyses, qualitative concepts and judgments are often required to be translated into quantitative measures to enable scalable, consistent, and traceable analyses. Methods, processes, and tools (MPTs) should be engineered together to promote transparent, intuitive, rational, and quantifiably traceable foundations for resiliency analyses.

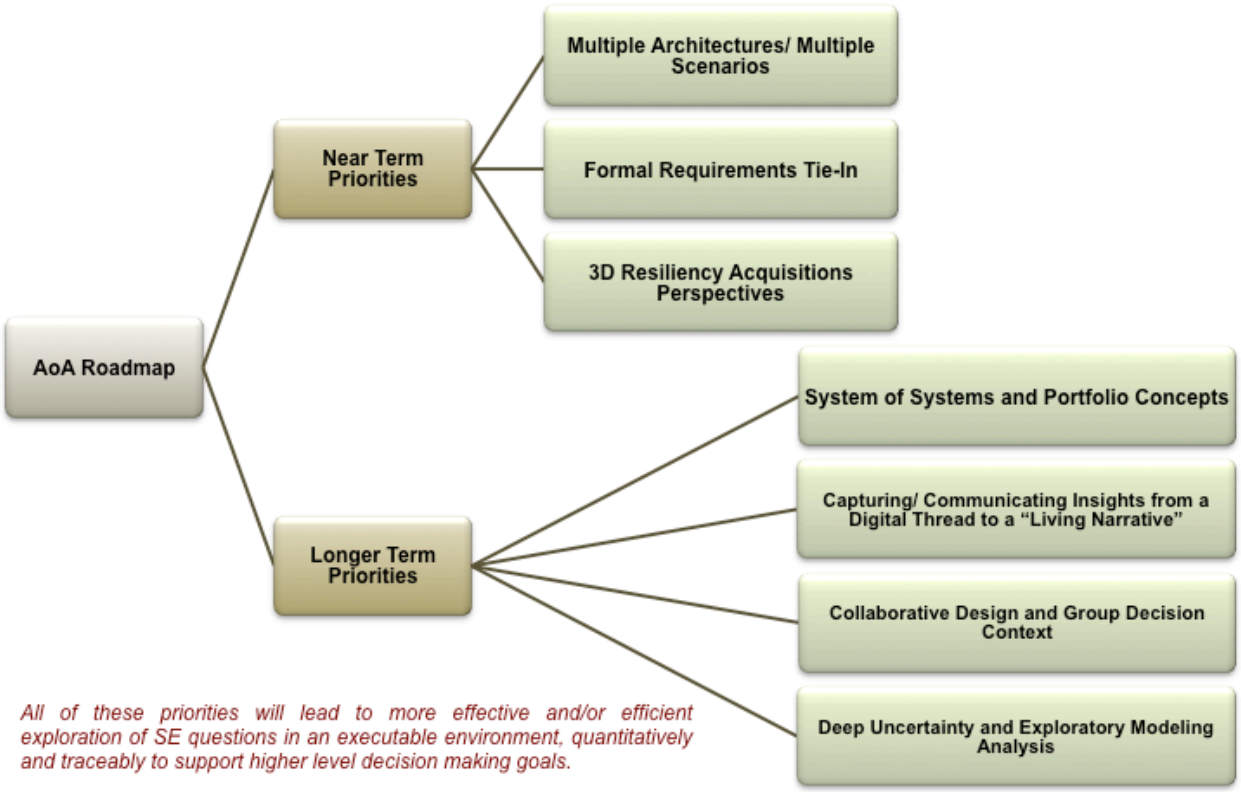


Figure 29. Near and longer term AoA Roadmap priorities

Just as there is no one monolithic AoA in terms of nature and scope that will address all of the insights required of the DoD's Materiel Solution Analysis process, there is no single answer for how to support and implement the process in an executable environment. Rather, implementation tools and techniques should evolve to support a diverse community of users with diverse backgrounds and needs as well as a diverse community of externally hosted M&S components that may be needed for any given analysis. In this way, we support the design of resilient systems through a resilient implementation amenable to a resilient analysis process. The overarching goal is to support DoD Leadership in keeping with the vision for ERS through more effective evaluation, definition, and maturation of systems architectures and requirements alongside diverse stakeholders needs and expectations (present and future) in a computational environment. This will enable operationally relevant analyses of resiliency that better support the acquisitions process across a system's lifecycle.

7.2 Relation of the Roadmap to the AoA Module

At this point, the AoA Roadmap begins to relate directly to the AoA Module and its development. To effectively support the concept of computationally based AoAs, we need to do several things:

- Design the framework that executes and manages analyses to provide efficient and scalable frontend to backend data retrieval, data manipulation, data updating, etc.
- Define, at least at a perhaps generalized but common level, what actions in the course of these analyses happen where in the computational environment.

- Develop an understood framework in support of common operations that will enable us to create truly dynamic, interactive analyses in line with an analyst's needs.

The overall effect of these processes is to not replace human intelligence by making the decisions automatically, but rather to enhance or augment the human analyst's intelligence through better implementation of information technology, effectively enhancing our human abilities to manipulate and understand information. In turn, this controls our ability to develop new, improved insights.

The AoA module is a direct implementation of these concepts, serving as an enhanced analytical user interface that handles the data retrieval, manipulation, and updating behind the scenes and allowing the analyst to focus on developing a better understanding to the problem.

7.3 THE AOA MODULE

The AoA module was developed so it could be integrated into both TradeAnalyzer for large data sets (1M-10M) and TradeBuilder for small to medium size data sets (1k-10k). The significant difference in the two integration efforts was that when integrated into the ERS TradeBuilder framework, the changes done to requirements could be persisted and explored using the other views in the framework (e.g., SysML viewer).

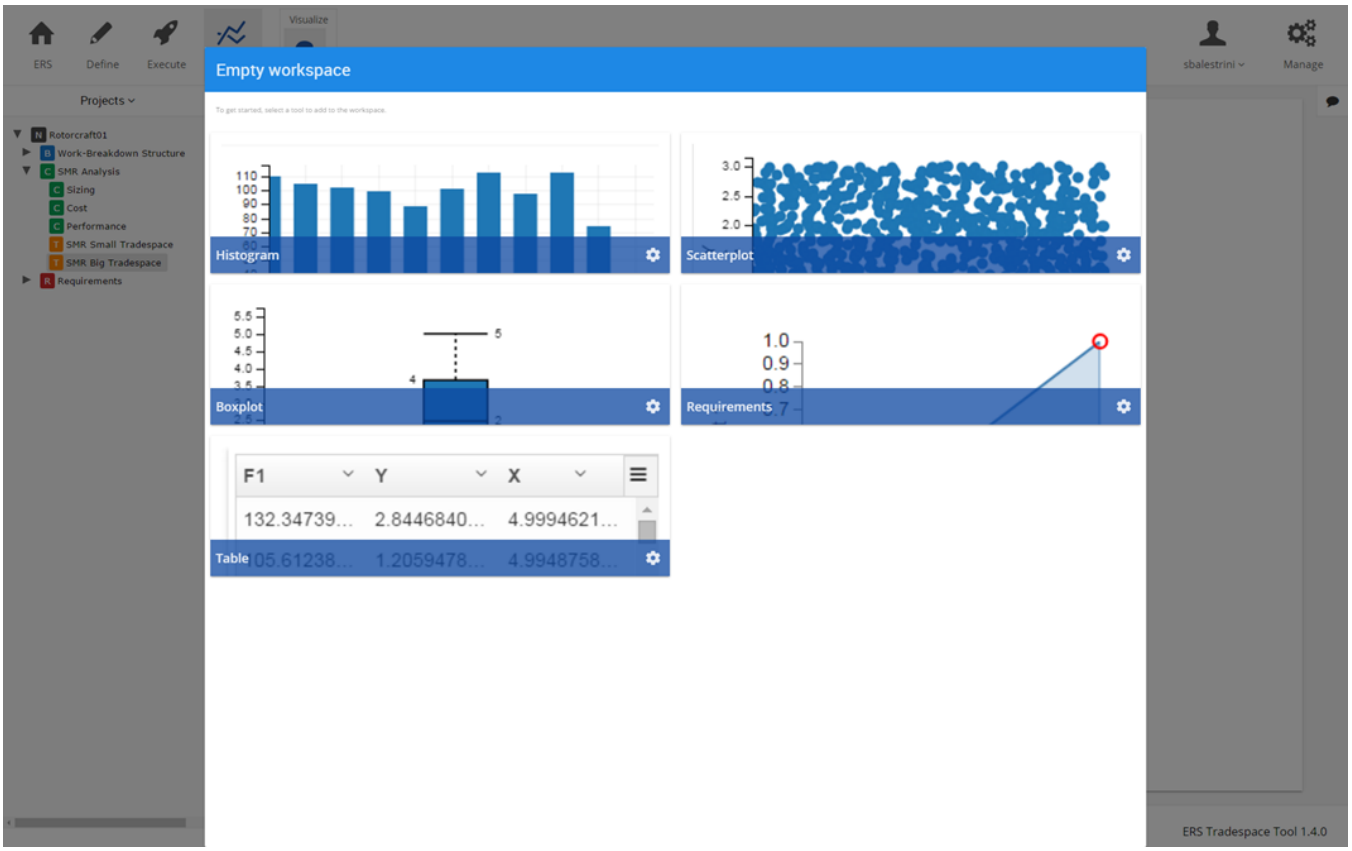


Figure 30. Initial AoA Module screen (after tradespace selection)

The user is required to select a tradespace before the AoA module can be initialized. The user is then required to specify an initial tool (or graphical widget) to include in the dynamic visualization environment, as depicted in

Figure 30. If the user selects the scatterplot tool, he/she will then be presented with a simple scatterplot of two variables in the tradespace, as depicted in Figure 31, where maintenance cost is plotted against max power.

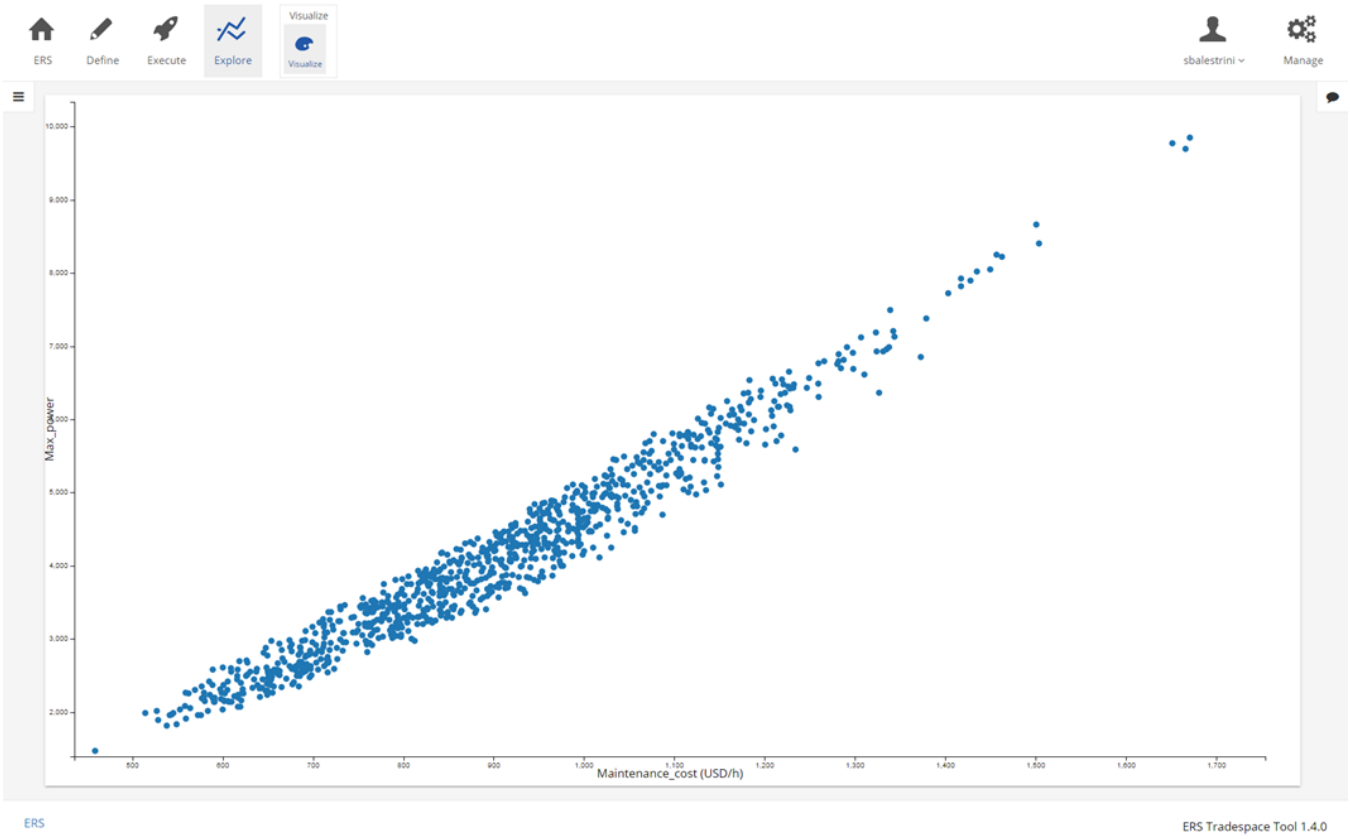


Figure 31. Initial Scatterplot

The user can then change the variables visualized or add more tools. When adding new tools, the user must specify where to add it. When selecting a new variable, the user must specify on which tool to include it, e.g., as an axis on the scatterplot tool.

The tools that can currently be included in a dynamic dashboard are:

- **Barchart**: Displays one variable.
- **Box and Whisker Plot**: Displays one variable.
- **Scatterplot**: Displays three variables (x-axis, y-axis, and color)
- **Requirements**: Allows users to modify the requirements tree (can add/delete requirements, modify weights and utility functions).
- **Table**: a table of the tradespace where columns are variables and rows are cases (or data points, or alternatives).

The data in the dashboard is dynamically updated, allowing users to brush (i.e., select) data on one view and see the impact on the other views.

Caveats for the AoA tool is that due to limitations in current browser capabilities, the number of data points that can be displayed is generally limited to a few thousand, with performance quickly decreasing when the datasets reach 10,000 cases or so.

Figure 32. A more complex dashboard comparing total satisfaction of requirements, KPPs, KSAs and cost

In Figure 32 we see a more complex dashboard that can be generated. In this case we are comparing a KPPs (ferry range and hover ceiling), a KSA (max power), a cost metric (maintenance cost), the total aggregate requirement (the color, yellow is better, blue is worse) while changing our preference for the purchase price of the system (which impacts the total satisfaction of our requirements. This tool allows users to dynamically rearrange the types of visuals they want to display, the variables they evaluate, and their requirements preferences (both in terms of the utility functions and the aggregation functions).

7.4 FUTURE SYNTHESIS WITH TRADESTUDIO

The ERDC developed TradeAnalyzer includes the capability to display much larger datasets than what is possible with the current instantiation of the AoA tool as used in TradeBuilder. Conversely, the AoA module allows users to interact with the data and cast it in multiple ways to try to understand the tradespace and interact with it by filtering and changing the decision makers' preferences (i.e., requirements). The two tools are therefore synergistic and there would be value in developing a tighter integration between the two. Steps that will be required to do so include: (1) expand the data model in TradeAnalyzer or connect it to TradeBuilder database, (2) coordinate the *HDF5* format between the two tools, (3) update the look and feel of the two to produce a more seamless experience, (4) and potentially allow the user to filter in the TradeAnalyzer widgets and update the views in the AoA module.

Other tools are external to TradeBuilder and may exist as some synthesis of a server, a library of various analytical models and simulations, and an interface through which external software may submit inquiries. Executable Architecting Systems Engineering (EASE), developed by the US Army Research Laboratory, is one such toolset. EASE focuses on creating a systems engineering data-driven infrastructure to facilitate interoperability. EASE allows System of Systems (SoS) design encapsulation and connects an interview system that allows a user to launch a distributed M&S execution based on functional and scenario choices relevant to the operational context. Prior efforts, to include those under SERC RT-120, developed an interface between EASE and the ERS tools.

This task (under RT-145) was concerned with creating an update to the interface to EASE given the release of TradeBuilder v1.4.1, perform some minor upgrades to the implementation, and support the delivery of an initial integrated solution to the Army's Edgewood Chemical and Biological Center (ECBC).

The interface to EASE had to be updated due to (1) changes by the EASE development team, (2) the upgrades to OpenMDAO v1.0 (which serves as TradeBuilder's primary execution engine), and (3) an improvement to the security of the protocol used for communicating with EASE. The updates required were not as extensive as originally planned so the majority of the resources associated with this task were used to support other efforts who had grown in scope. A demonstration Jupyter notebook was developed to illustrate the more advanced functionality for integrating with EASE through its API.

During a two-day trip to Edgewood, MD, the GTRI team supported the installation of the framework in the ECBC servers, introduced the ECBC attendees to the tool, and demonstrated the interface to EASE. In conjunction with the present EASE team members, the GTRI team developed sample problems on the fly and showed the attendees how a OneSAF scenario exposed by EASE could be integrated into the ERS TradeBuilder framework, executed some scenarios, and demonstrated how the data generate could then be analyzed in the framework.

9 TRADESPACE SUPPORT TO ERS DEMO

The ERS Demo task primarily supported the development of example models to illustrate how the framework can support representative projects. In addition, the funding was also used to provide assistance to the ERDC team exploring integration with various analysis tools, the most significant one being NASA's OpenVSP, as well as supporting the deployment of the tool to other entities (e.g., Naval Postgraduate School).

9.1 DEMO MODELS

The demonstration models task evolved to become an opportunity to highlight how the framework could be used to support representative applications. Three models were developed during this effort, (1) a fighter aircraft sizing model, (2) a rotorcraft conceptual design models, and (3) a ground vehicle performance analysis model.

9.1.1 FIGHTER AIRCRAFT SIZING MODEL

The fighter aircraft model was developed to illustrate how a simple aircraft sizing and synthesis routine could be integrated into the framework. Rather than integrating a complex tool (e.g., NASA's FLOPS), the team opted to implement a simpler routine that leverages open literature approaches to aircraft sizing and synthesis. Three primary sources provided the mathematical foundation for the model, namely: (Mattingly, 1987), (Raymer, 1999), and (Roskam, 2006). This model notionally sizes and costs aircraft based on the required missions the aircraft is expected to perform. The initial version of the model was demonstrated and delivered to ERDC, but shifting priorities led the team to stop development on the model and instead focus on new features for the framework.

9.1.2 ROTORCRAFT CONCEPTUAL DESIGN MODEL

A series of rotorcraft models were developed under this task to exemplify how a project may evolve from high-level requirements to tradespace analysis. Physics-based models were generated to size a single main rotor helicopter and to analyze its performance, and a simple cost model was also created. A Jupyter notebook is used to generate all rotorcraft artifacts and serves as an example of how a team may codify this process in a repeatable environment. In the notebook, a new project was created with a hierarchy of requirements. A Work Breakdown Structure (WBS) was created in accordance with MIL-STD-881C, representing the single main rotor helicopter. When run, the resulting WBS is generated in SysML in the frontend as well, as shown in Figure 33. Attributes are assigned to the WBS and the models are added to create a tradespace. Both a direct execution and asynchronous workers are used to demonstrate how to execute the models, creating a tradespace.

The team was also able to acquire a copy of NASA Design and Analysis of Rotorcraft (NDARC), a more complex rotorcraft sizing and synthesis tool. A wrapper for a tandem helicopter was used to demonstrate how such a model could be executed by OpenMDAO and by the tool itself.

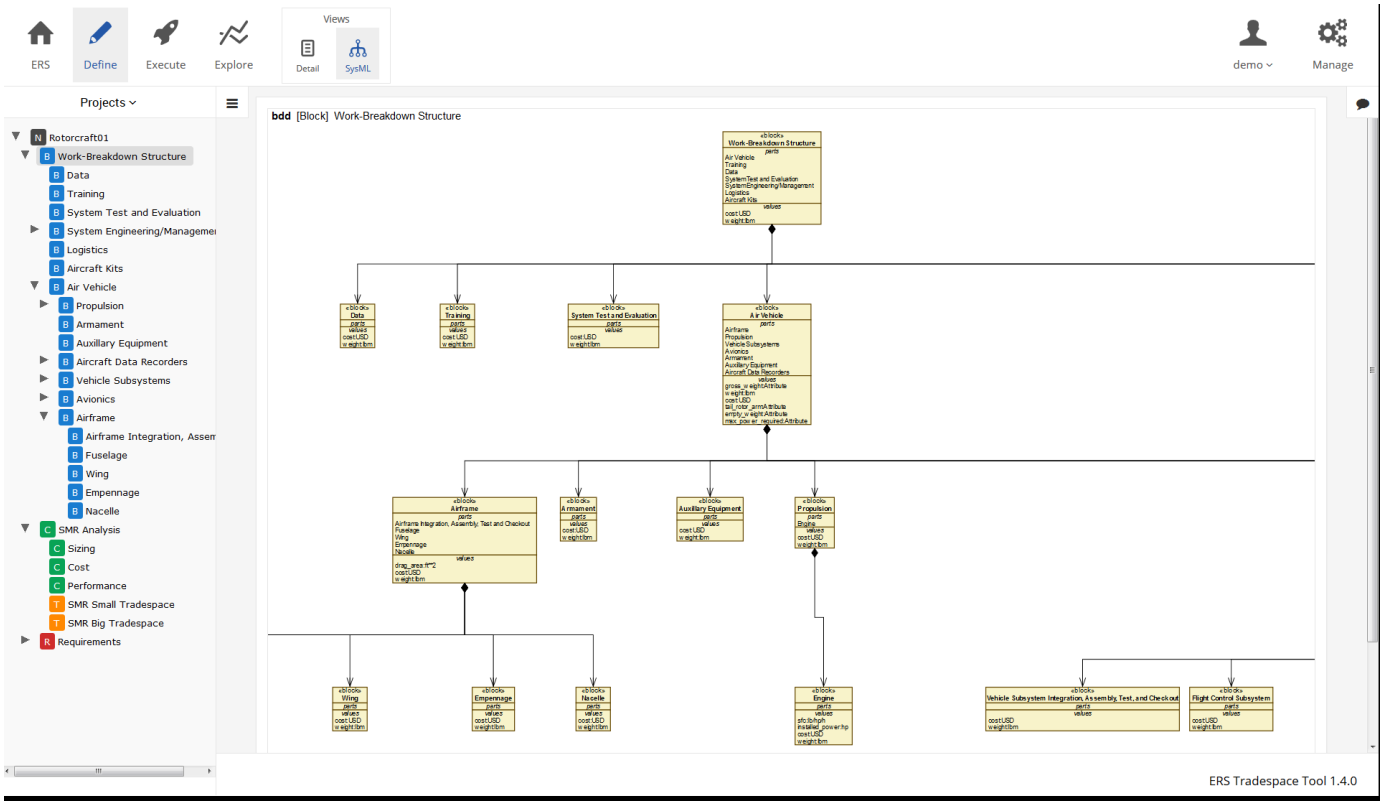


Figure 33: Rotorcraft Model Screenshot

9.1.3 GROUND VEHICLE MODEL

A notional model for a Joint Light Tactical Vehicle (JLTV) was developed to help illustrate the applicability of the framework to systems other than aircraft. The JLTV model was formulated as a performance analysis model, where values are given for the environment and the design variables, and the model calculates the performance of the vehicle (e.g., acceleration, turn radius). This exemplifies a different problem than the rotorcraft and aircraft model that focus on sizing and synthesis of a vehicle to achieve a mission or series of missions.

9.2 OPENVSP

This task included providing support to ERDC in using OpenVSP by helping them compile the tool, as well as the python interface. Used the SWIG python interface provided with OpenVSP to build a higher-level interface that is easier to employ. Using this higher-level interface, it was possible to execute OpenVSP from the Jupyter notebook. The team experimented with developing a visualizer in the browser for STL files which can be created by OpenVSP, as given in Figure 34 for simplified, notional low cost aircraft. Using this method it was possible to modify the model using the API, have OpenVSP generate an STL file, and then visualize the model in 3D inside the Jupyter notebook. The team also provided some limited support to the ERDC team in integrating OpenVSP with the Navy's Multidisciplinary Analysis Optimization Integrated Environment (MAOIE), as well as some subject matter expertise in the use of the tool.

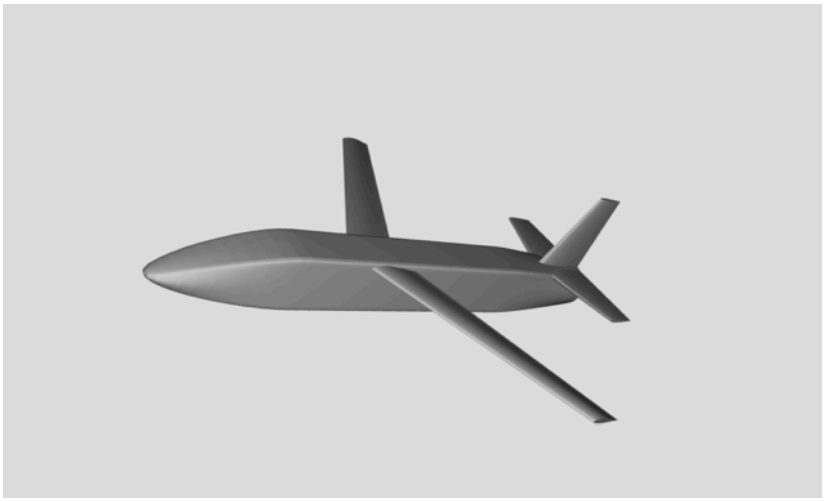


Figure 34. Simple Interactive Model of Notional Low Cost Aircraft

10 RECOMMENDATIONS AND NEXT STEPS

The focus for early-stage design is not identifying an optimal solution, but rather narrowing down the potential solution space for more rigorous data collection, generation, and evaluation. In decision analyses, qualitative concepts and judgments are often required to be translated into quantitative measures to enable scalable, consistent, and traceable analyses. Throughout this effort, we adhere to the philosophy that the most important goal is insight, not numerical treatment or inference.

Recommendations for the next stage of maturation for the **analytical methods** are as follows:

- Work in tandem with the software engineering effort to more formally define a multi-objective decision analysis workflow, albeit an expanded method that includes classical steps as well as some of the more nuanced approaches such as those described in this work. This should be graphically driven and easy for MODA practitioners to follow so that we may develop broader community use and insights that lead to best practices across these methods.
- Use decision theory, and explicitly the decision space, as a springboard and guidepost for further analytical explorations that will inform the decision process. This includes maturing methods to better sample and generate Pareto-relevant data as discussed in Section 2.3.
- Harmonize and mature methods to robustly identify drivers in the sPF that are equally informative across parameters that may be highly correlated and not independent. More effectively identifying drivers will be one of the most important ways to add insight to the decision making process. For example, how does changing the preference structure either entirely or simply by changing thresholds and objectives change the identification of "best" alternatives, and how does that change the cost drivers?
- Mature the way we bring different sensitivity analysis methods into the workflow and problem space so that we can evaluate a wider range of problems for "How sensitive is *P* with respect to *Q*?". For example, we need to know how sensitive our valued parameters (defining a Needs Context) are to (a) environment variation or (b) operational use characteristic variation. Conversely, we may need to ask which design alternatives in our soft Pareto set are least sensitive (in terms valued parameter levels or multi-additive value) to these dimensions.

Recommendations for the **software development** (to include TradeBuilder, and any other component of the ERS TradeStudio suite):

- More streamlined integration of the ERS TradeBuilder toolset with ERDC's broader TradeStudio suite of tools; this includes investigation of options for back end refactoring and/or front end "look and feel"
- Defining opportunities for enhanced software stability
- Enabling server deployment of the software on ERDC assets (to include ERDC's Common Computing Environment)
- User interface enhancements; this may include features such as AoA "stop light" charts, prediction and/or contour profilers of attributes, and others as prioritized by ERDC
- Development of an initial software user's manual
- Identification of operational simulations of interest to specific ERS applications (e.g. AFSIM for aircraft concepts, OneSAF for ground vehicles, etc.); extend this to include development of and/or enhancements to existing Application Protocol Interfaces (API) between the ERS tools and those operational simulations.

11 REFERENCES

Barron, F. H., & Barrett, B. E. (1996). “Decision quality using ranked attribute weights”. *Management Science 42*(11): 1515-1523.

Fisher, R. A. (1936). "The use of multiple measurements in taxonomic problems". *Annals of Eugenics* 7 (2): 179–188.

Goerger, S.R., Madni, A.M., and Eslinger, O.J. (2014). “Engineered Resilient Systems: A DoD Perspective”, *Conference on Systems Engineering Research (CSER 2014)*, *Procedia Computer Science* 28: 865-872.

Lahdelma, R., Miettinen, K., & Salminen, P. (2003). “Ordinal criteria in stochastic multicriteria acceptability analysis (SMAA)”. *European Journal of Operational Research* 147(1): 117-127.

Mattingly, J. D., Heiser, W. H., and Daley, D. H., "Aircraft Engine Design," AIAA Education Series, AIAA, New York, NY, 1987.

Johnson, E. R., Parnell, G. S., Tani, S. N., & Bresnick, T. A. (2013). “Perform Deterministic Analysis and Develop Insights”. In *Handbook of Decision Analysis*, 166-226.

Parnell, G. S., Terry Bresnick, M. B. A., Tani, S. N., & Johnson, E. R. (2013). *Handbook of decision analysis*, (Vol. 6). John Wiley & Sons.

Raymer, D. P., "Aircraft Design: A Conceptual Approach," 3rd ed., AIAA Education Series, AIAA, Reston, VA, 1999.

Roskam, J., "Airplane Design: Part I-VIII." DARcorporation, Lawrence, KS, 2006.

Roszkowska, E. (2013). “Rank Ordering Criteria Weighting Methods–a Comparative Overview”. *Optimum Studia Ekonomiczne* 5 (65): 14-33.

Ryan, E. T., Jacques, D. R., & Colombi, J. M. (2013). “An ontological framework for clarifying flexibility-related terminology via literature survey,” *Systems Engineering*, *16*(1), 99-110.

Sitterle, V. B., Curry, M. D., Freeman, D. F., & Ender, T. R. (2014). “Integrated Toolset and Workflow for Tradespace Analytics in Systems Engineering”. In *INCOSE International Symposium* 24(1): 347-361. Las Vegas, NV (US): INCOSE.

Sitterle, V. B., Freeman, D. F., Goerger, S. R., & Ender, T. R. (2015). “Systems Engineering Resiliency: Guiding Tradespace Exploration within an Engineered Resilient Systems Context”. *Procedia Computer Science 44*: 649-658.

Sitterle, V.B., Balestrini-Robinson, S., Freeman, D.F., and Ender, T.R. (2016). “Analysis of Alternatives Roadmap”, interim report prepared for U.S. Army Engineering Research and Development Center (Vicksburg, MS), under contract # HQ0034-13-D-0004 (SERC RT-145).

Sitterle, V.B. (2016). “*Cross-scale resilience: Relating Systems of Systems to Individual System Analysis and Back Again*“, interim report prepared for U.S. Army Engineering Research and Development Center (Vicksburg, MS), under contract # HQ0034-13-D-0004 (SERC RT-145).

Sitterle, V.B., Brimhall, E.K., Freeman, D.F., Balestrini-Robinson, S., Ender, T.R., and Goerger, S.R. (2016). "Bringing Operational Perspectives into the Analysis of Engineered Resilient Systems", in 26th Annual INCOSE International Symposium (IS 2016), Edinburgh, Scotland, UK, July 18-21, 2016.

U.S. Government Accounting Office. (September 2009). "Defense Acquisitions: Many Analyses of Alternatives Have Not Provided a Robust Assessment of Weapon System Options", Report GAO-09-665.

Waskom, M. (2015). http://stanford.edu/~mwaskom/software/seaborn/index.html

APPENDIX A – PUBLICATION & OUTREACH OUTPUTS FOR RT-145

CONFERENCE PAPERS (REFEREED)

- Sitterle, V.B., Brimhall, E.L., Freeman, D.F., Balestrini-Robinson, S., Ender, T.R., Goerger, S.R., "Bringing Operational Perspectives into the Analysis of Engineered Resilient Systems," Proceedings of the 26th INCOSE International Symposium, Edinburgh, Scotland, UK, 18-21 July 2016.
- Balestrini-Robinson, S., Freeman, D., Arruda, J., Ender, T.R., "ERS TRADESPACE in support of a Heavy Vertical Lift Capability," 2015 AHS Systems Engineering Technical Specialists Meeting, Huntsville, AL, 14-15 September, 2015.

CONFERENCE PAPERS (NON-REFEREED OR ABSTRACT ONLY)

- Ender, T.R., Goerger, S.R., "ERS Tradespace Toolset," Proceedings of the 19th Systems Engineering Conference, NDIA, Springfield, VA, October 2016. [ACCEPTED]
- Sitterle, V., Balestrini-Robinson, S., Ender, T., Goerger, S.R., "Cross-scale resilience: bridging system of systems and constituent system engineering and analysis," Proceedings of the 19th Systems Engineering Conference, NDIA, Springfield, VA, October 2016. [ACCEPTED]
- Sitterle, V., Balestrini-Robinson, S., Freeman, D., Arruda, J., Goerger, S., and Ender, T., "Tradespace Tools for Engineering Resilient Systems,", INFORMS Annual Meeting, Nashville, TN, November 2016. [ACCEPTED]
- Goerger, S.R., Ender, T.R., Chausse, D.C., Freeman, D.F., Balestrini-Robinson, S., Richards, J.E., Baylot, E.A., "Emergent DoD Tool for Tradespace Analysis" Proceedings of the 84th MORS Symposium, Quantico, VA, 20-23 June 2016.
- Goerger, S.R., Ender, T.R., Chausse, D.C., Freeman, D.F., Balestrini-Robinson, S., Richards, J.E., Baylot, E.A., "How to Use an Emergent DoD Tool for Tradespace Analysis" Proceedings of the 84th MORS Symposium, Quantico, VA, 20-23 June 2016.
- Ender, T.R., Goerger, S.R., "Engineered resilient systems: large scale tradespace capabilities," Proceedings of the 18th Systems Engineering Conference, NDIA, Springfield, VA, October 2015.

SERC INVITED TALKS

- Balestrini-Robinson, S., Freeman, D.F., Sitterle, V.B., Ender, T.R., Goerger, S.R., "Engineered Resilient Systems – Tradespace Tools, "Systems Engineering Research Center Sponsored Research Review, Washington, DC, 3 December 2015.