This ERS Tradespace Tools effort will build upon GTRI’s prior experience with web-based, collaborative Model Based Systems Engineering frameworks.

Several proofs of concept have been developed for ERS, and will be expanded on in this effort.

**Develop Decision Support Methods and Tradespace Toolset Framework for ERS**

Explores scenario based needs context to forecast resilient options
Builds the collaborative tradespace tools incorporated within the ERS Software Architecture
Analytical Objectives

- Investigate how to operationalize formalisms into measureable and executable constructs ... to support Pre-Milestone A Tradespace Analysis
- Create flexible, rationally guided workflow supported by modular and composable analytical constructs
- Enable the customer to quantify, compare, and visualize dimensions of resiliency in the AoA process

GTRI’s ERS Focus
- Linking Pre-Milestone A Design to Modeled Operational Capabilities

Early stage design to evaluate system designs in terms of their fielded characteristics and performance in operational environments

Systems engineering of resiliency through AoA and refinement of more complete and robust requirements definition.
Dimensions of Resiliency for Analytical Evaluation

**Initial Analytical Focus**

- Robustness to needs across wide range of contexts
- Flexibility to reconfiguration and replacement
- Value-centric (Broad Utility) basis

Needs Context Analysis

Evaluate Design Alternatives according to their capabilities and characteristics as ‘valued’ by Stakeholders

- Addresses ‘Robustness’ of system capabilities using a Broad Utility construct.
- Enables analysis across Stakeholders & Mission Profiles that necessitate different objectives
- Helps capture Resiliency of a system design across competing or changing requirements
- Scaled against Objective and Threshold requirement levels (KPP concept) to promote comparability across analyses

**Robustness**

“the measure of how effectively a system can maintain a given set of capabilities in response to external changes after it has been fielded”

(Ryan, Jacques, and Colombi 2013)

‘Robustness of Fielded System Capabilities and Capacity with respect to Operational Requirements’
Evaluate risk associated with initial engineering design choices or later changes of key system components

• Addresses ‘Flexibility’ of a system design through sensitivity to Engineering Change
• Evaluates the sensitivity of design alternatives to changes across critical Design Variables
• Helps capture resiliency to potential reconfiguration or replacement

Flexibility
“the measure of how easily a system’s capabilities can be modified in response to external change”
(Ryan, Jacques, and Colombi 2013)

‘Flexibility of a Designed System to Future Engineering Change’

Begin to paint a proxy picture of how an Engineering Change will impact the entire system design.
A use case has a specific path through the networked workflow. Driving the tool development with the generalized workflow helps ensure we can meet the requirements of future use cases.
Leveraging USMC Investment in Collaborative MBSE

• **Collaborative Development:** browser-based tool enables analysis as a web service

• **Performance:** “black box” approach to performance models integration through metadata interface

• **Cost:** acquisition cost estimating relationships and trusted Operations & Sustainment cost model

• **Model Based Sys Eng (MBSE):** conforms to SysML and accepted systems engineering standards

**Dynamic System Comparison**

<table>
<thead>
<tr>
<th></th>
<th>Alternative 1</th>
<th>Alternative 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Land Range at Cruise</td>
<td>300.00  400.00</td>
<td>409.58 mi</td>
</tr>
<tr>
<td>Max Swim Speed</td>
<td>7.89 mph</td>
<td>10.30 mph</td>
</tr>
<tr>
<td>Max Water Speed at 1' SWH</td>
<td>8.19  17.48</td>
<td>5.31 knot</td>
</tr>
<tr>
<td>Minimum HP Required to Maintain Speed</td>
<td>670.20 hp</td>
<td>603.21 hp</td>
</tr>
</tbody>
</table>
Software Re-architecting (1)

- **Database structure**
  
  **Previously**
  - Hierarchical format but across many collections
  - Queries were complex, redundant and data access was generally difficult

  **Now**
  - Refactored into a single tree structure
  - Data access eased
  - Allows future capability in things like linked data queries
  - Whenever possible the model resembles SysML data structures

- **Object Document Mapping (ODM)**

  **Previously**
  - No mapper at all
  - Queries scattered throughout code (often duplicated)

  **Now**
  - MongoEngine based mapping of objects in code to database documents
  - Data access performed through this layer instead of queries in code

- **REST API**

  **Previously**
  - Django views that queried the database and provided data in formats specific to the view (duplicating code)

  **Now**
  - First cut API is a 1-to-1 mapping of the objects to endpoints
  - Working on second cut that provides data in more useful sets
**Software Re-architecting (2)**

### Execution Engine

**Previously**
- Home-grown execution engine
- Could not handle cyclic references in execution graph

**Now**
- Replacing with OpenMDAO from NASA GRC
- Cyclic references can be solved for
- Comprehensive and extensible libraries for units, optimization and DoE
- Developed methods of wrapping existing parametric models as well as ERS packages

### Leveraging AngularJS

**Previously**
- Pages that required similar components (e.g., a tree layout of a WBS) did not reuse common components
- Control logic was often duplicated where it was needed rather than reused
- Plug-ins not supported

**Now**
- Creating AngularJS components (e.g., directives and controllers) from common front-end code
- Angular components are being packaged (using Bower) into plug-ins
- Intend to reuse these plug-ins in further refactoring of the front end UI
- New plug-ins to use the new REST API for data access
Define

Define your system of interest through authoring SysML Block Definition and Parametric Diagrams. Populate your database of system options with off-the-shelf and notional parts. Set your system KPPs and KSAs as requirements. Compare solutions and measure against requirements. This text is not correct. Science words all of them yes what am I typing.

Execute

Execute trade studies using set distributions, sampling available system options or defining Designs of Experiment. Visualize and explore the results of your trade study and DoE executions using box & whiskers, probability graphs, scatterplot matrices, and coordinated interactive views. Compare solutions and measure against requirements.

Analyze

Analyze a configuration using the Point Solution Sandbox or Sensitivity Analysis. Visualize and explore the results of your trade study and DoE executions using box & whiskers, probability graphs, scatterplot matrices, and coordinated interactive views. Compare solutions and measure against requirements. Adding more words that mean nothing.
Collaborative SysML Authoring

Model Tree
This tree allows you to traverse your model. Pick a node to see changes to the view screen you have chosen to the right.

- Car_Domain
  - Environment
    - Car
      - Car_mass
      - Car_acceleration
      - Car_torque_ratio
      - Car_engine_rpm
    - Car_engine_power_torque_fuelburn
  - Vienna

SysML View
This view allows you to manipulate your nodes using SysML.

- Car_Domain
  - Environment
    - Car
      - Car_mass
      - Car_acceleration
      - Car_torque_ratio
      - Car_engine_rpm
    - Car_engine_power_torque_fuelburn
  - Vienna

- Car
  - Car_mass
  - Car_acceleration
  - Car_torque_ratio
  - Car_engine_rpm
- Car_engine_power_torque_fuelburn

- Engine
  - Fuel
    - Piston_Stroke_Length
    - Number_of_Cylinders
    - Throttle
    - Air_to_Fuel_Ratio
    - Compression_Ratio
    - Exhaust_Gas_Pressure
    - Fuel_Burn
    - Compression_Wall_Temperature
    - Engine_Torque
    - Direct_Valve_Diameter

- Transmission
  - Engine
    - Final_Drive_Ratio
    - Current_Gear
    - Transmissions_Mass
    - Transmission_Ratio
  - Engine_Mass
  - Engine_Volume

- Tire
  - Circumference
  - C_D_dia
  - Area

- Chassis
  - Chassis_Mass
  - Chassis_Size

- Transmission_Mass
  - Transmission_Ratio
  - Transmission_Mass

- Environment
  - Atmosphere_Temperature
  - Specific_Car_CO2
  - Atmospheric_Pressure
  - Molar_Mass
  - Universal_Gas_Constant
  - Atmospheric_Density
  - Specific_Hill_Ratio
  - Car_Mass

- Tire_Mass
  - Tire_Ratio

- Transmission_Mass
  - Transmission_Ratio

- Engine
  - Fuel
  - Piston_Stroke_Length
  - Number_of_Cylinders
  - Throttle
  - Air_to_Fuel_Ratio
  - Compression_Ratio
  - Exhaust_Gas_Pressure
  - Fuel_Burn
  - Compression_Wall_Temperature
  - Engine_Torque
  - Direct_Valve_Diameter
• Executable Architecture Systems Engineering (EASE)
  — Links analytical, experimental and training objectives with the technical complexity of Modeling & Simulation
  — Explore operational aspects of the analytical questions in simulation

• ERS effort develops interface between MBSE/Tradestudies and Army Research Lab investment in executable, cloud-computing resources

• Enables evaluation of MoE’s based on scenarios
EASE-Tradespace Integration

Phase 1 - Software Interface Demonstration (RT-49)
- API developed between EASE and legacy FACT tools
- Executed EASE-facilitated tradespace study for limited prototype of a UAV traversing McKenna terrain

Phase 2 – Two Tracks: Tradestudies of Lifecycle Capability Resiliency and Simulation on Operational Scenarios using “ERS Tradespace” (RT-120)
- Demonstration of Needs Context/Risk Methodology
- Demonstration of Operational Scenario (Use Case: TBD)
EASE: Early Tradespace Results

- Tradestudy execution generates numerous design options for evaluation against various threats
- Pareto frontier illuminates the “non-dominated” design trades
- Heat map can be used to identify those threat characteristics against design options that meet given Probability of Detection thresholds and objectives
ERS Ships: Early Tradespace
Proof of Concept

SysML models of Expanded Ship WBS

Browser friendly 3-D based on NSWCCD CAD models

Robust Design Driven Evolution of Parameters
ERS-Ships: LX(R) “Firedrill” Study

LX(R) Program Objective
• Build 11 amphibious ships to replace aging fleet of Landing Ship Dock
• Must be more cost effective than the San Antonio (LPD-17) program

ERS LX(R) Study Objective
• Support NAVSEA in analyzing tradespace for LX(R)
• Process improvement: learn how to support tradespace analysis for current and relevant projects

LX(R) surrogate models
Obfuscated surrogate models
Integrated LX(R) Tradespace Analysis Tool
ERS Ships: Lessons Learned on Usability

- LX(R) effort enabled GTRI to “teach” others how to use the tool and evaluate it its usability
  - Other users learned the process end-to-end and apply their knowledge to a project without having GTRI assistance

- Dr. Andrew Strelzhoff (ERDC) was taught how to use FACT and model a system and integrate analyses

Lessons Learned:

- There is not ONE workflow that can be supported, but the networked workflow described previously
- Graphical User Interfaces are good for novice users, but as soon as users become proficient and the project scales, they are inefficient
- Modeling with SysML needs to be improved for large scale models, either importing table data, or create a command line interface (or ideally both)
- Mapping attributes to constraints is a painful process
- Must support generation and processing of orders of magnitude more data than it is currently possible
Next Steps: Full Integration with the ERS Hub
Recommended Next Steps

Systems Engineering Resiliency Analysis

Expand and mature analytical building blocks for resiliency dimensions in modular, scalable, and composable manner

- Mature the concept of context to include operational & environmental characteristics
  - Adds dimension to Tradespace and takes a step toward a profile or topography of Resiliency

- Integrate uncertainty into Needs Context analysis
  - Creates probabilistic distributions of broad utility

Mission Profile_1
Mission Profile_2

Attribute Tradespace:
[ Intrinsic | Extrinsic – Profile 1 | Extrinsic – Profile 2 ]
Acknowledgements

Simon R. Goerger, PhD
Director, Institute for Systems Engineering Research
Engineer Research & Development Center
US Army Corps of Engineers

Christopher McGroarty
Simulation and Training Technology Center
Human Research and Engineering Directorate
Army Research Laboratory

Eric Spero
Vehicle Applied Research Division
Vehicle Technology Directorate
Army Research Laboratory

Portions of this material are based upon work supported, in whole or in part, by the United States DoD through the Systems Engineering Research Center (SERC) under Contract HQ0034-13-D-0004. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology. The views and conclusions are those of the individual authors and participants, and should not be interpreted as necessarily representing official policies, either expressed or implied, of the DoD, any specific US Government agency, or the US Government in general.