System Re-tasking to Achieve Resilience in an SoS

Payuna Uday and Dr. Karen Marais
Aeronautics and Astronautics Engineering
Purdue University

24 October, 2012
(NDIA – 15th Annual Systems Engineering Conference)

This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract H98230-08-D-0171. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology.
Outline

• Motivation

• Representation of an SoS

• Analytic framework

• Illustrative example
  — Notional 5-node SoS
  — Results and discussion

• Summary and Future work
Motivation

• What is resilience?
  — “Resilience is the ability of a system or organization to react to and recover from disturbances at an early stage with minimal effect on its dynamic stability”
  
  *(Resilience Engineering: Concepts and Precepts (2010))*

• Historical approach:
  — Improve resilience through over-design
  — Traditional systems engineering practices anticipate and resist disruptions
  — Resilience incorporated through classical reliability methods:
    o Redundancy
    o Preventive maintenance
Stand-in redundancy in SoSs

- Not suitable for SoSs:
  - Heterogeneity, geographical distribution, interdependencies
  - Backup systems are costly and impractical

- Stand-in redundancy:
  - Compensate for loss of performance in one constituent system by re-tasking remaining systems
  - As one node experiences degradation, other nodes can alter their operations to compensate for this loss

- Raises interesting questions:
  - Given a system failure, what is the best configuration to compensate for the loss?
  - What level of performance can be recovered with new configuration?
  - What is upstream effect of stand-in redundancy on development costs and risks?
Stand-in redundancy in SoSs

- Impact of stand-in redundancy on resilience of SoS:
  - Reactive Resilience
  - Proactive Resilience
SoS Representation

Capability (Performance)
Requirements (Requirement Capability)
Systems (System Capability)

SoS
Capabilities
Functions
SoS Representation

- Consider metrics at capability level:
  - Level of Performance (LoP)
  - Level of Reliability (LoR)

- LoP depends on systems, functions, performance metrics, interdependencies

- LoR depends on reliability of individual systems
Analytic framework

Gradual degradation of systems with time

Drop in performance due to loss of system(s)

Initial SoS configuration

SoS after system failure
Analytic framework

Gradual degradation of systems with time

Drop in performance due to loss of system(s)

Initial SoS configuration

Improved performance by re-tasking systems

Value of stand-in redundancy

SoS after system failure
Analytic framework

Minimize $\text{SoS operations cost}$

Subject to:

$\text{LoP}_i \geq \text{LoP}^T_i$

$\text{LoR}_i \geq \text{LoR}^T_i$

**LoP:**
- systems, functions, performance metrics, interdependencies

**LoR:**
- reliability of individual systems

**Desired LoP, LoR:**
- Min. level of high performance and high reliability original SoS should satisfy

**Acceptable LoP, LoR:**
- Min. acceptable level re-tasked systems must provide

**SoS Operations Cost**

*Fully functional state:* Number of systems, operating cost of each system

*Failed system state:* Operating costs of remaining systems, cost to repair/replace system, downtime costs

*Re-tasked state:* Operating cost of remaining systems, acquisition costs of additional features
Features available on each system:
UAV1: High-definition camera
UAV2: Basic camera + weapons
UAV3: Basic camera + weapons

<table>
<thead>
<tr>
<th>Capability</th>
<th>Description</th>
<th>Systems Needed</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Surveillance</td>
<td>S1</td>
</tr>
<tr>
<td>C2</td>
<td>Target identification</td>
<td>S1, S2</td>
</tr>
<tr>
<td>C3</td>
<td>Target elimination</td>
<td>S3, S4</td>
</tr>
</tbody>
</table>
Illustrative example

- **Representation of SoS:**

  ![Diagram showing SoS with different elements and capabilities.]
## Systems and functions

<table>
<thead>
<tr>
<th>Systems</th>
<th>Individual system functions/features</th>
<th>Area imaged</th>
<th>Imaging resolution</th>
<th>Revisit rate</th>
<th>Target strike-rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Satellite</td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>-</td>
</tr>
<tr>
<td>UAV-1 “search”</td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>-</td>
</tr>
<tr>
<td>UAV-2 “seek and destroy”</td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>UAV-3 “seek and destroy”</td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

- Modifications/enhancements in SoS:
  - Features on satellite cannot be changed
  - Easier to retrofit UAVs with higher performance devices
  - (UAVs can also be reprogrammed for higher revisit rates)
Results: Surveillance (C1)

LoP for Capability 1 (Surveillance)

- **Satellite**: 100
- **UAV-1**: 60
- **UAV-1 & UAV-2**: 70

**Satellite fails**
- **Graceful degradation**
- **Re-tasked systems**
- **Stand-in redundancy**

Systems that contribute to Capability 1 (Surveillance)
Results: Target identification (C2)

Systems that contribute to Capability 2 (Target ID)

- Satellite & UAV-1
  - Satellite fails
  - Graceful degradation
- UAV-1
  - Re-tasked systems
  - Stand-in redundancy
- UAV-1 & UAV-2
  - Graceful degradation
  - Re-tasked systems
  - Stand-in redundancy

Systems that contribute to Capability 2 (Target ID)

- Satellite & UAV-1
- Satellite
- Satellite & UAV-2
Results: Target elimination (C3)

- Indirect impact of system failure on capabilities
- Need to consider immediate needs of mission during functional reconfiguration
Discussion

• Using stand-in redundancy, systems can:
  — Contribute to SoS-level capabilities in ideal case, and
  — “Stand-in” for failed functions during failures

• Limit to level of stand-in redundancy that can be incorporated
  — Appropriate resource allocation

• Need to consider balance between resilience, costs, and adaptability of the SoS
  — For example, multi-modal transportation networks are designed for long lifetimes with gradual modifications and/or upgrades
Summary and future work

• Large scale SoSs evolve with time along with changing environment

• This approach indicates incremental enhancements/modifications to existing systems can provide inherent resilience

• Approach can help decision-makers quantitatively assess resilience of different SoS architectures

• Future work:
  — Expand static model to dynamic model (resilience under uncertainty)
  — Track system degradation with time (proactive resilience)
  — Consider multi-system failures