SysML Strategies to Characterize and Analyze Systems of Systems

Jo Ann Lane (jolane@usc.edu)
Tim Bohn (tbohn@us.ibm.com)
Overview

- System of systems (SoS) engineering core elements
- SysML models that support SoS engineering
- Example SoS SySML models
- Conclusions
What is a “System of Systems”?  

- Very large systems using a framework or architecture to integrate constituent systems  
- Exhibits emergent behavior not otherwise achievable by constituent systems  
- SoS constituent systems (CS)  
  - Independently developed and managed  
  - New or existing systems in various stages  
  - May include multiple COTS products  
  - Have their own purpose  
  - Can dynamically come and go from SoS  

- Typical domains  
  - Business: Enterprise-wide and cross-enterprise integrations  
  - Military/Crisis Response: Dynamic communications infrastructure

Based on Mark Maier’s SoS definition [Maier, 1998]

Lane and Bohn NDIA 2009
SoS Engineering Core Elements

- Translating capability objectives
- Understanding systems & relationships
- Orchestrating upgrades to SoS
- Addressing requirements & solution options
- Monitoring & assessing changes
- Developing & evolving SoS architecture
- Assessing performance to capability objectives

External Environment

New SoS SE role
Persistent SoS overlay framework
SoS upgrade process
External influences
SoSE Core Element Description

- **Translating Capability Objectives**
  - Starts with an SoS need or new capability
  - Works to understand new capability and alternatives for providing it

- **Understanding Systems and Their Relationships**
  - Collects and maintains information about current state of the SoS and its CSs

- **Assessing Performance to Capability Objectives**
  - Evaluation of current performance and how performance meets current and future needs

- **Developing/Evolving SoS Architecture**
  - Evaluation of existing SoS architecture and identification of alternatives to mitigate limitations and improve performance

- **Monitoring and Assessing Changes**
  - Monitoring of CS non-SoS changes

- **Addressing Requirements and Solution Options**
  - Evaluation/prioritization of SoS requirements
  - Evaluation of solution options and selection of option

- **Orchestrating Upgrades**
  - Oversight activity to monitor progress of the CS SoS capability upgrades and mitigate obstacles
Desired SoS Engineering Modeling Support

- Understand CSs and their relationships
  - SoS architecture and capabilities
  - CS functional capabilities
  - Interfaces and protocols
  - Data elements, precision, and rates

- Develop and evolve an SoS architecture
  - Understand current architecture
  - Develop target architecture to guide SoS evolution
Desired SoS Engineering Modeling Support (continued)

- Assess CS changes
  - Impact to SoS architecture and capabilities
- Address new requirements and options
  - Implementation and transition strategies for desired capability
  - Impact to constituent systems
SysML Models that Support SoS Engineering Needs

- **Object classes**
  - Characterize each SoS CS and its capabilities

- **Interface classes**
  - Describe each CS interface

- **Input/output entity classes**
  - Express the associated data attributes of each data item transferred over that interface

- **Use cases**
  - Characterize both CS and SoS capabilities from the different user perspectives

- **Sequence diagrams**
  - Characterize and analyze the operational flow for an SoS capability
Example SoS: Regional Area Crisis Response SoS (RACRS)

Command Control Center (CCC) Context Diagram
Scenarios: CCC Use Cases
Evacuate Area Sequence Diagram
Evacuate Area Alternate Sequence for Intruder “Management”
CCC Interface Class

Command Control Center

- Evacuate Area (where: Area)
- Acknowledged (AckType: Acknowledgment)
- Report All Clear (who: Reporting Entity)
- Report Intruder (where: Area)
- Eliminate Intruder (where: Area)
- Report Intruder Eliminated (where: Area)
Evacuate Area I/O Entities
Evacuate Area I/O Entities by Actor

- **Fire Control**
  - Status Area Evacuated (where: Area)

- **Police Control**
  - Evacuate Area (where: Area)
  - Report Intruder (where: Area)
  - Report Intruder Eliminated (where: Area)

- **Reverse 911**
  - Send Calls (callType: Call Type)

- **UAV**
  - Patrol Evacuated Area (where: Area)
  - Eliminate Intruder (where: Area)
  - Target and Eliminate ()

- **Reporting Entity**
  - Police
  - Fire
  - UAV
  - UGV

- **Acknowledgment**
  - Evacuate

- **Area**
  - «send_receive»
Summary and Conclusions

- Recent SoSE research identified need for useful SoSE models
- Goal of presentation to show how SysML models can be used to support some of these needs
  - Context diagrams
  - Use cases
  - Object blocks
  - Interface classes
  - I/O classes
Summary and Conclusions (continued)

- Captures information distilled from multiple sources and integrates to provide a “bigger” picture and support
  - End-to-end performance of SoS mission scenarios
  - Evaluate new capability alternatives
  - Evaluate proposed architecture changes
  - Evaluate impacts of proposed CS changes not related to SoS capability changes

- Key to success in modeling SoSs
  - Model only the aspects that are important for the engineering activity
  - Consider using models in new ways, for example I/O classes to capture interface data attribute information
References


