



# Formal Arguments for Large-Scale Assurance (FALSA) Kick-off

February 6, 2023

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

Copyright 2023 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

DM23-0101

# JADC2/LSA Provides Motivation

**JADC2 vision:** Rapid deployment of new mission capability with confidence by composing capabilities across multiple domains from existing systems.

**Our focus:** rapid assurance with confidence through

- increased formality
- maximal reuse of prior assurance results

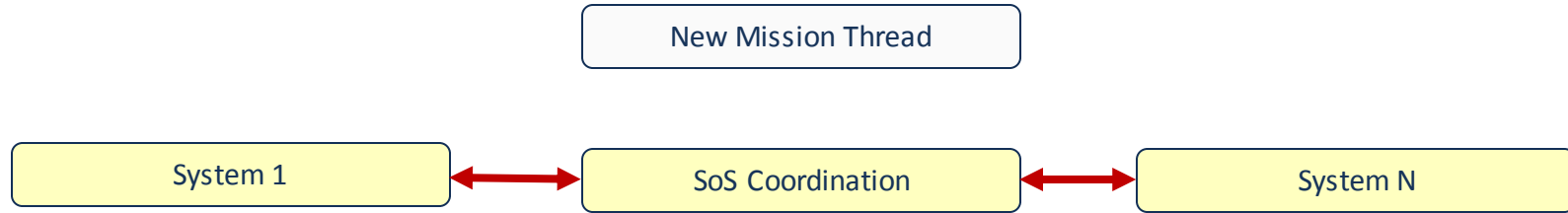
**Our approach:**

- Incrementally coalesce and structure prior assurance results of existing systems into assurance architectures and
- compose those assurance architectures to assure new capabilities provided by systems of systems (SoS).

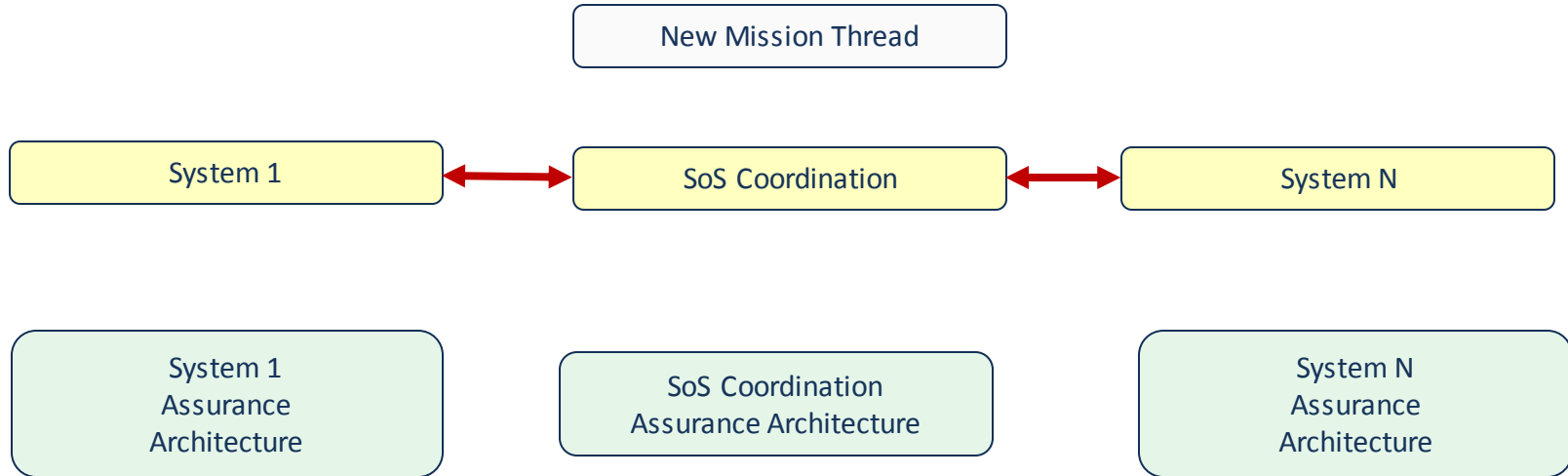
# LSA Approach



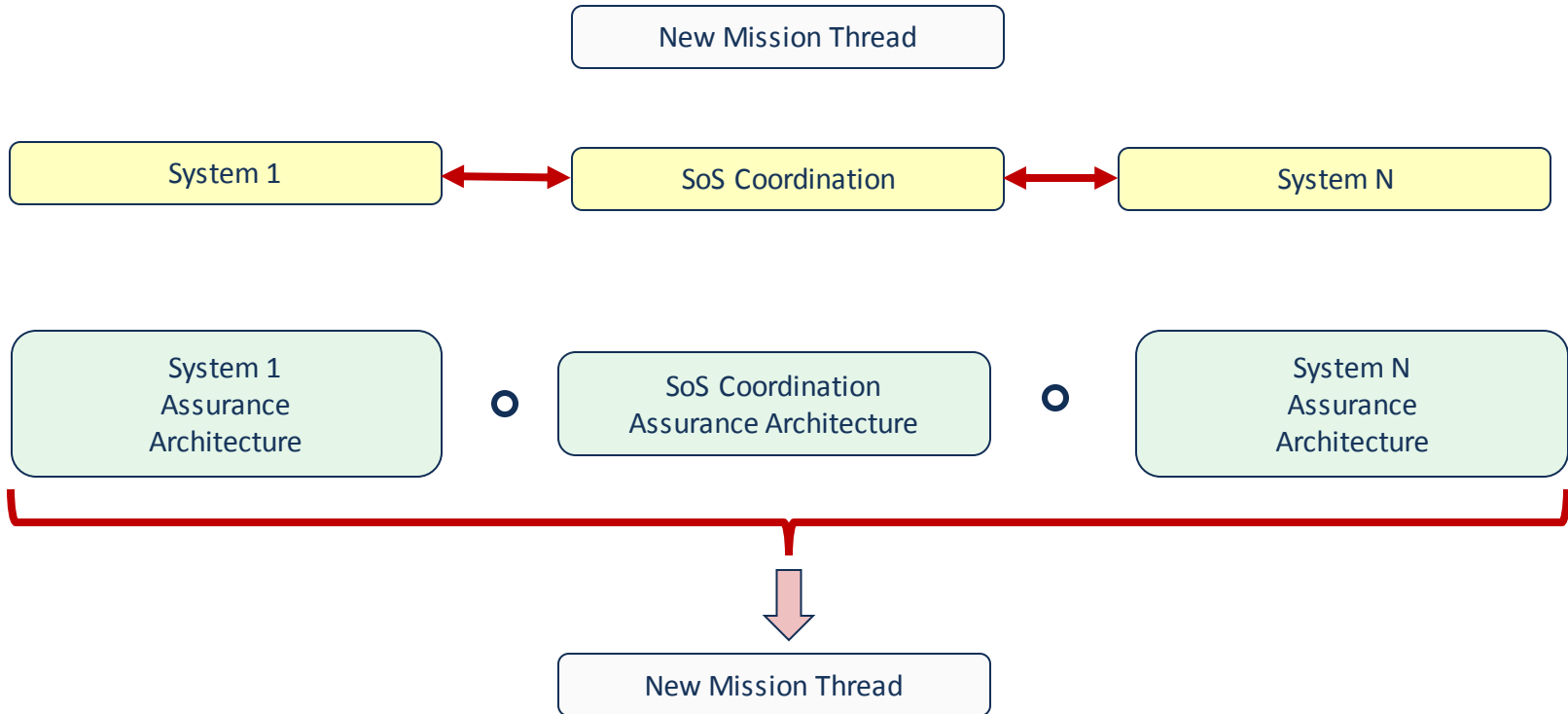
# LSA Approach



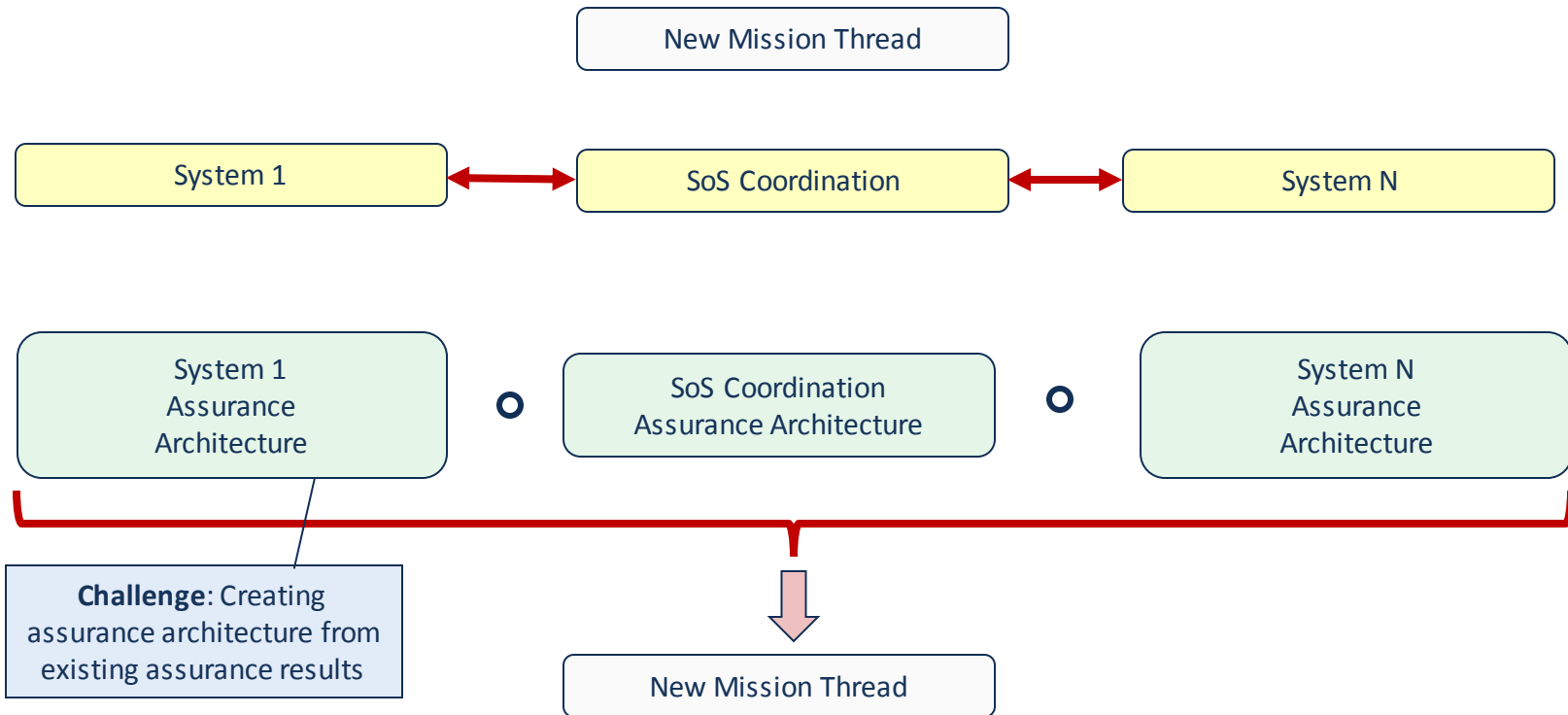
# LSA Approach



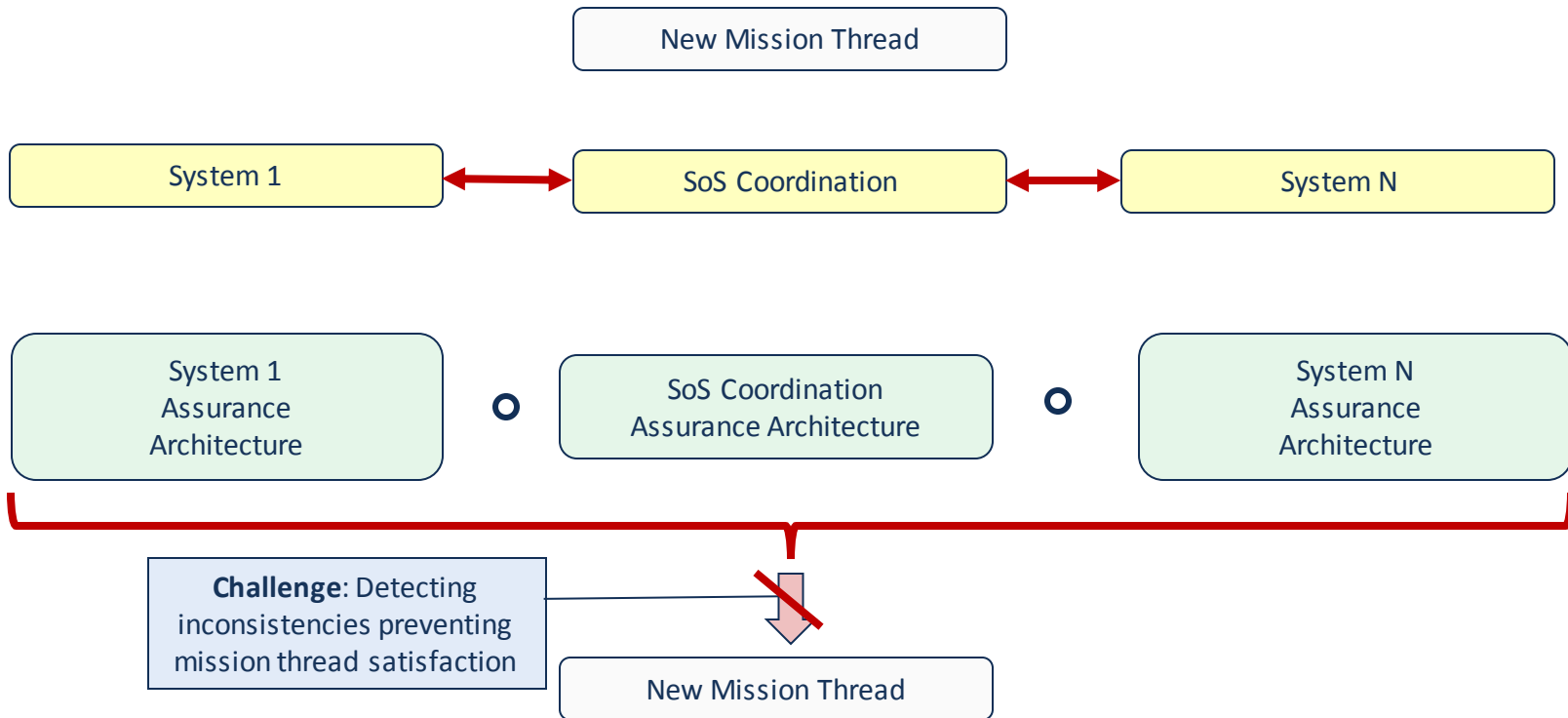
# LSA Approach



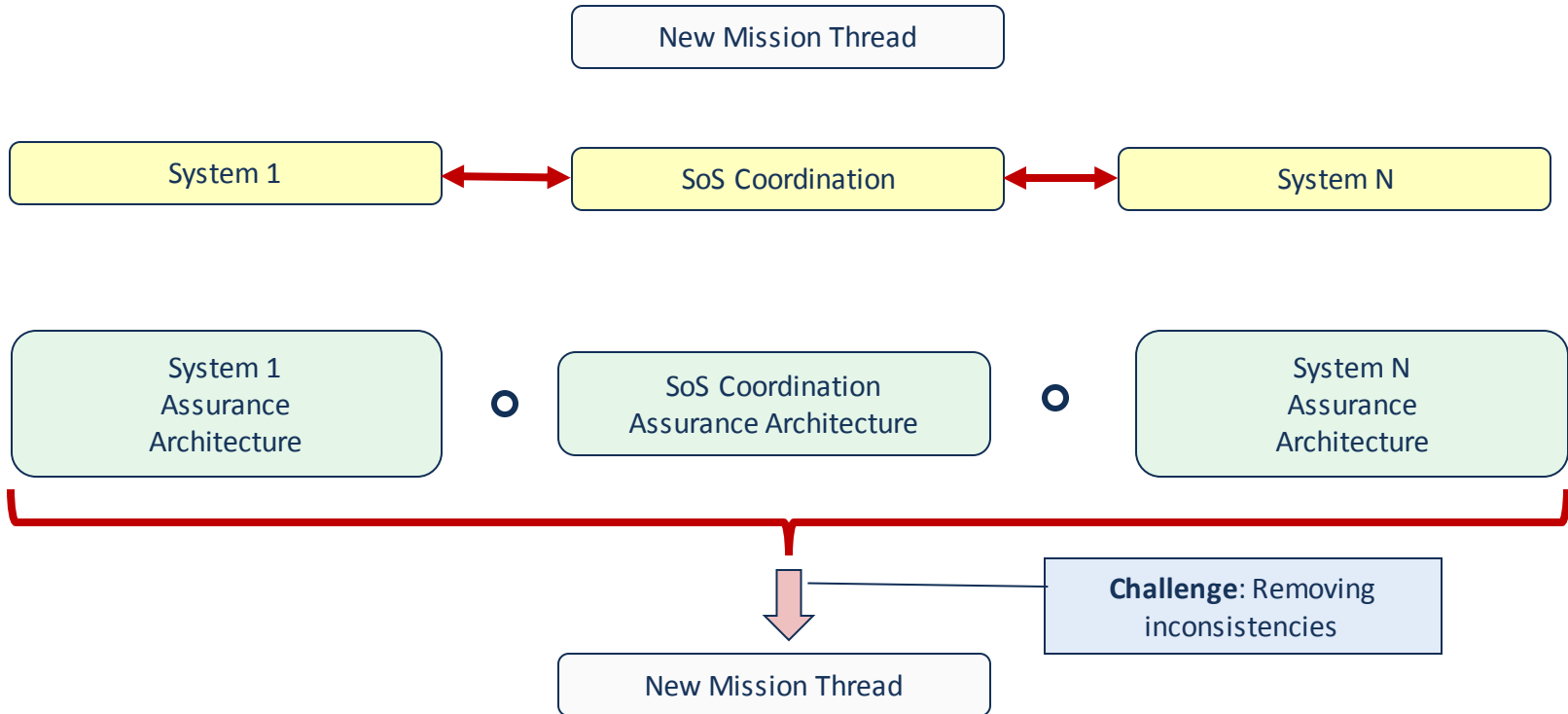
# Challenges



# Challenges



# Challenges



	FY23 ETP	FY24 MTP	Beyond FY24
Model problem	<ul style="list-style-type: none"> <li>• Start with existing in-house system</li> <li>• Control, timing, security</li> <li>• DoD Collaborator sandbox</li> </ul>	<ul style="list-style-type: none"> <li>• Representative of DoD Collaborator</li> <li>• Unclassified model problem related to classified parts of DoD Collaborator</li> </ul>	<ul style="list-style-type: none"> <li>• SoS</li> </ul>
Representing assurance architectures	<ul style="list-style-type: none"> <li>• Representing formal and informal</li> <li>• New definition of SAT</li> </ul>	<ul style="list-style-type: none"> <li>• Representative assurance architecture</li> <li>• Representing composed assurance architectures</li> </ul>	
Runtime conformance	<ul style="list-style-type: none"> <li>• Determining conformance</li> </ul>	<ul style="list-style-type: none"> <li>• Develop example of using assurance argument for ML</li> <li>• TBD</li> </ul>	<ul style="list-style-type: none"> <li>• Runtime enforcement</li> <li>• Runtime adaptation</li> </ul>
Refining assurance architectures	<ul style="list-style-type: none"> <li>• Using scientific ML</li> <li>• Thoughts on generalizing</li> </ul>		<ul style="list-style-type: none"> <li>• Incorporating runtime experience</li> </ul>
Composing assurance arguments		<ul style="list-style-type: none"> <li>• Detecting inconsistencies</li> <li>• Resolving inconsistencies</li> </ul>	<ul style="list-style-type: none"> <li>• Dynamic composition</li> <li>• System recommendations</li> </ul>

# FALSA Tasks and Milestones

Task 1 - Designing, developing, and representing an assurance argument. Milestones:

- Model problem created
- Requirements for representation of assurance architecture and technologies to satisfy them

Task 2 - Monitoring runtime behavior vis-à-vis the conformance to system's assurance argument. Milestone:

- Strategies for conformance checking

Task 3 - Learning from runtime data. Milestone

- Applying scientific ML to the control aspects of the model problem

# LSA Tasks

Task 1 - Creating assurance architectures with formal and informal parts

Task 2 - Compositional Assurance: Identifying inconsistencies

Task 3 - Compositional Assurance: Resolving inconsistencies

# Model Problem

Define experiments for ETP and MTP

Model problem – ETP

- Start with something we have on hand
- Probably should have control, timing and security aspects

Model problem – MTP

- Work with DoD collaborator

For milestone:

- Define experiments for ETP
- Model problem running
- Assurance architecture for subset of model problem with only formal parts

# Representing Assurance Architecture

Do some logic homework

Gain deeper insight into System M

Redefining SAT for formal / informal

Get Py-Z3, System M, etc tooling running

Develop an initial assurance architecture for the model problem

For milestone:

- Draft paper on representing formal and informal and new definition of SAT

# Conformance Checking

Question: Does runtime behavior conformance to assurance architecture?

- How to determine conformance?
- How to distinguish between erroneous behavior and imprecise assurance architecture?

For milestone:

- Results of conformance experiments

# Applying Scientific ML

Applying scientific ML to refine control models

Applying scientific ML to refine timing models

Using scientific ML for conformance checking

For milestone:

- Draw some conclusions about refining assurance architectures based on runtime behavior perhaps leading to an ETP

# Administrative

## Subteams

- Model problem: John, Anton, Gabe, Dio, Raffaele
- Representation: Amit, Limin, Gabe, Mark, Dio
- Conformance: Gabe, Mark, Raffaele, Anton
- Learning: Mark, Jasmine, Raffaele, Gabe

## Tools

- Jira Kanban board
- Zotero group library