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

February 6, 2023

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

JADC2/LSA Provides Motivation

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

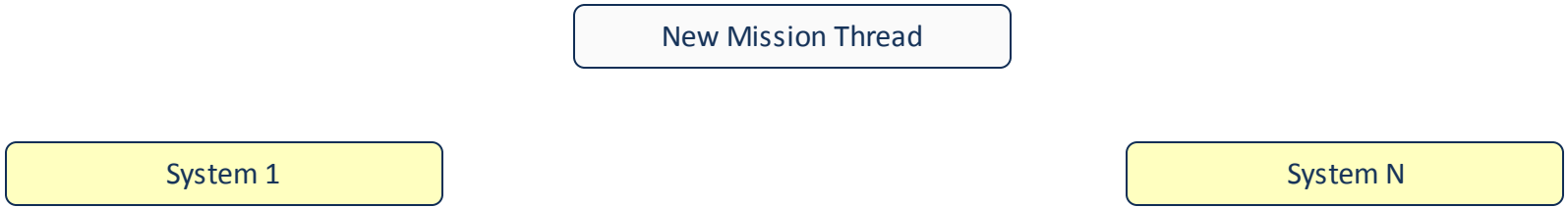
Our focus: rapid assurance with confidence through

- increased formality
- maximal reuse of prior assurance results

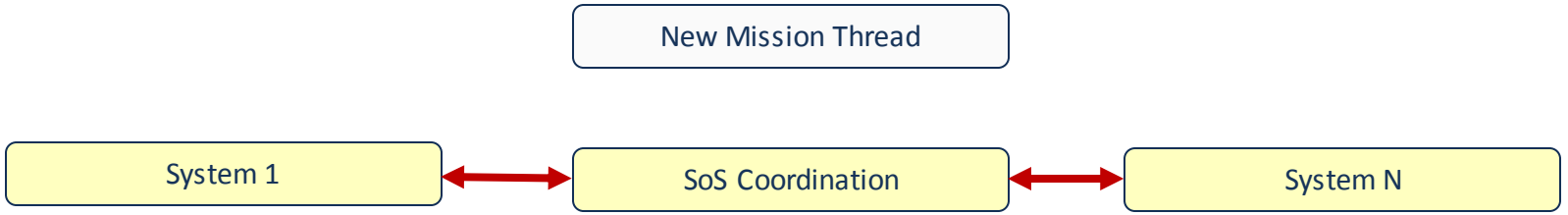
Our approach:

- 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).

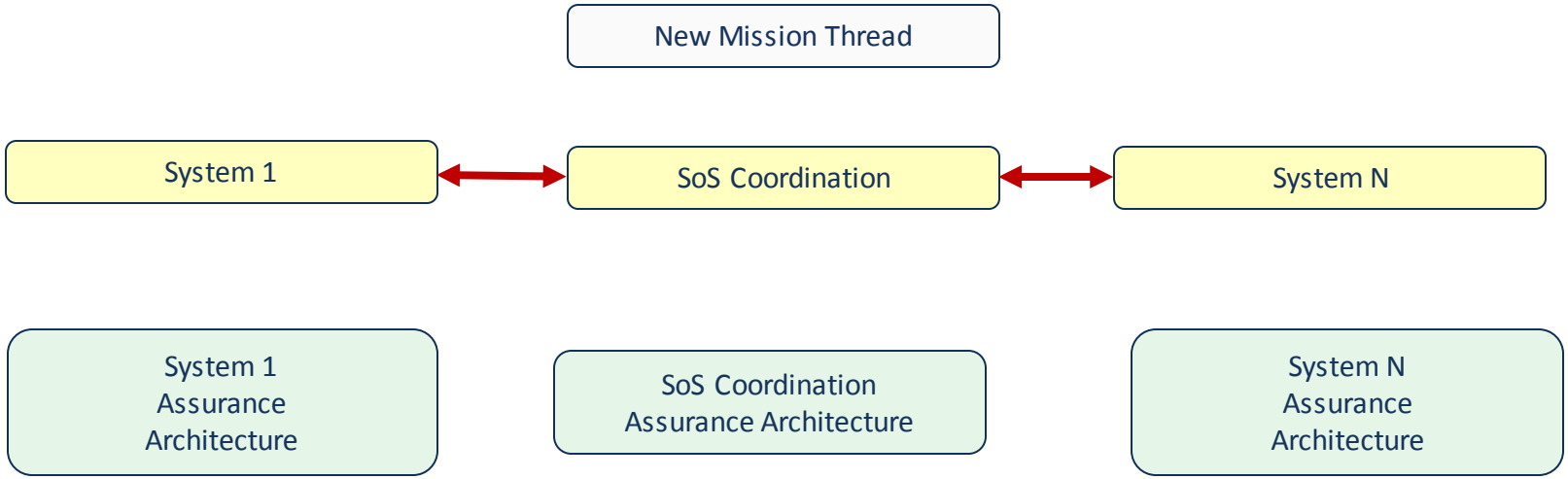
LSA Approach



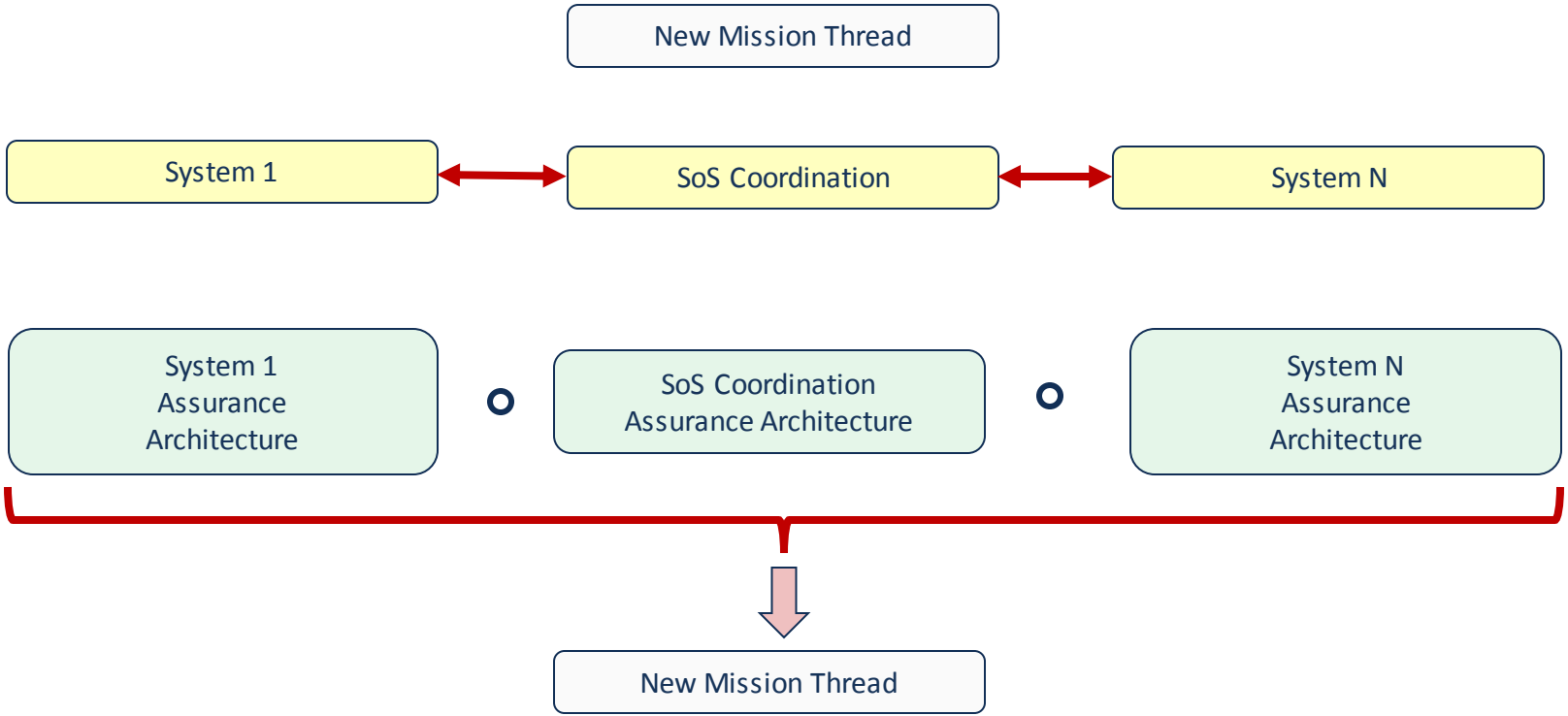
LSA Approach



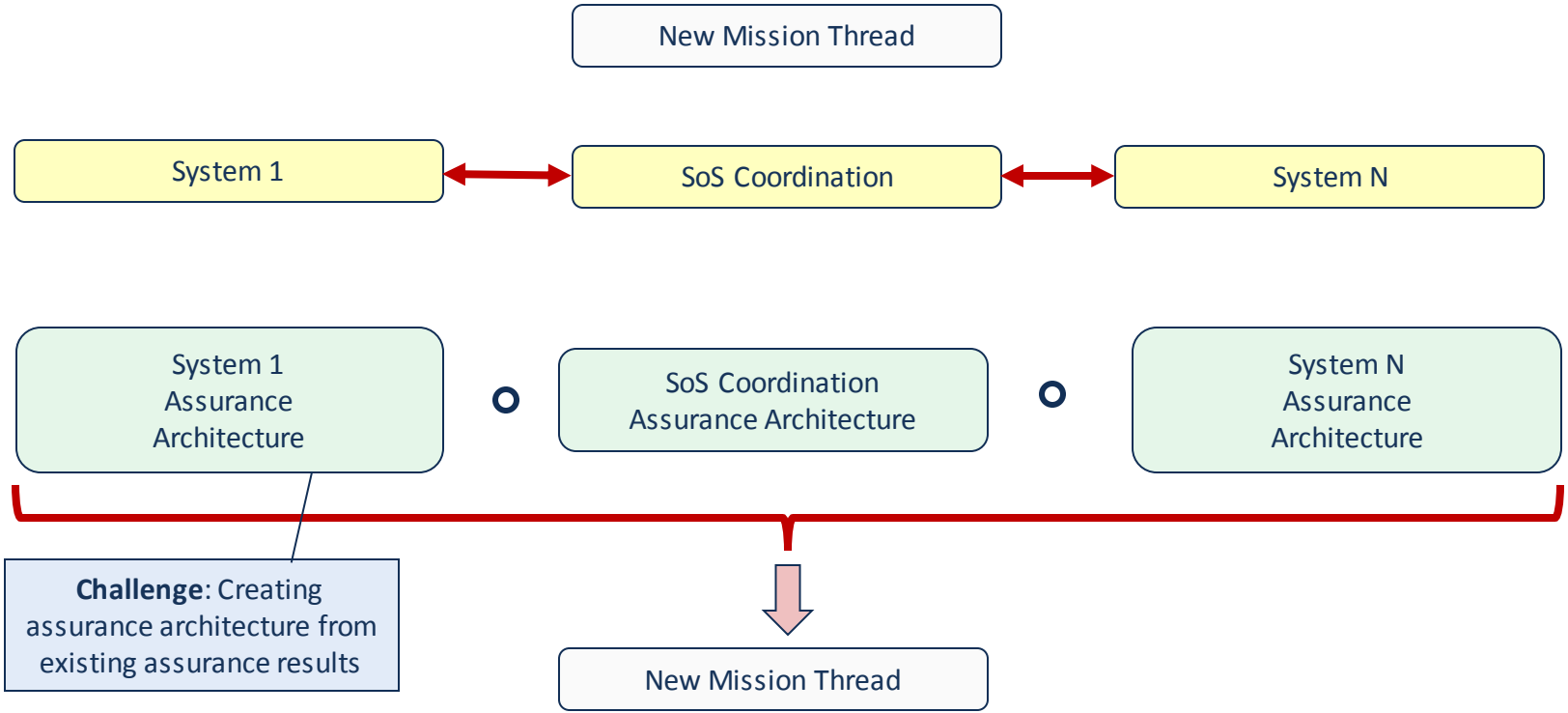
LSA Approach



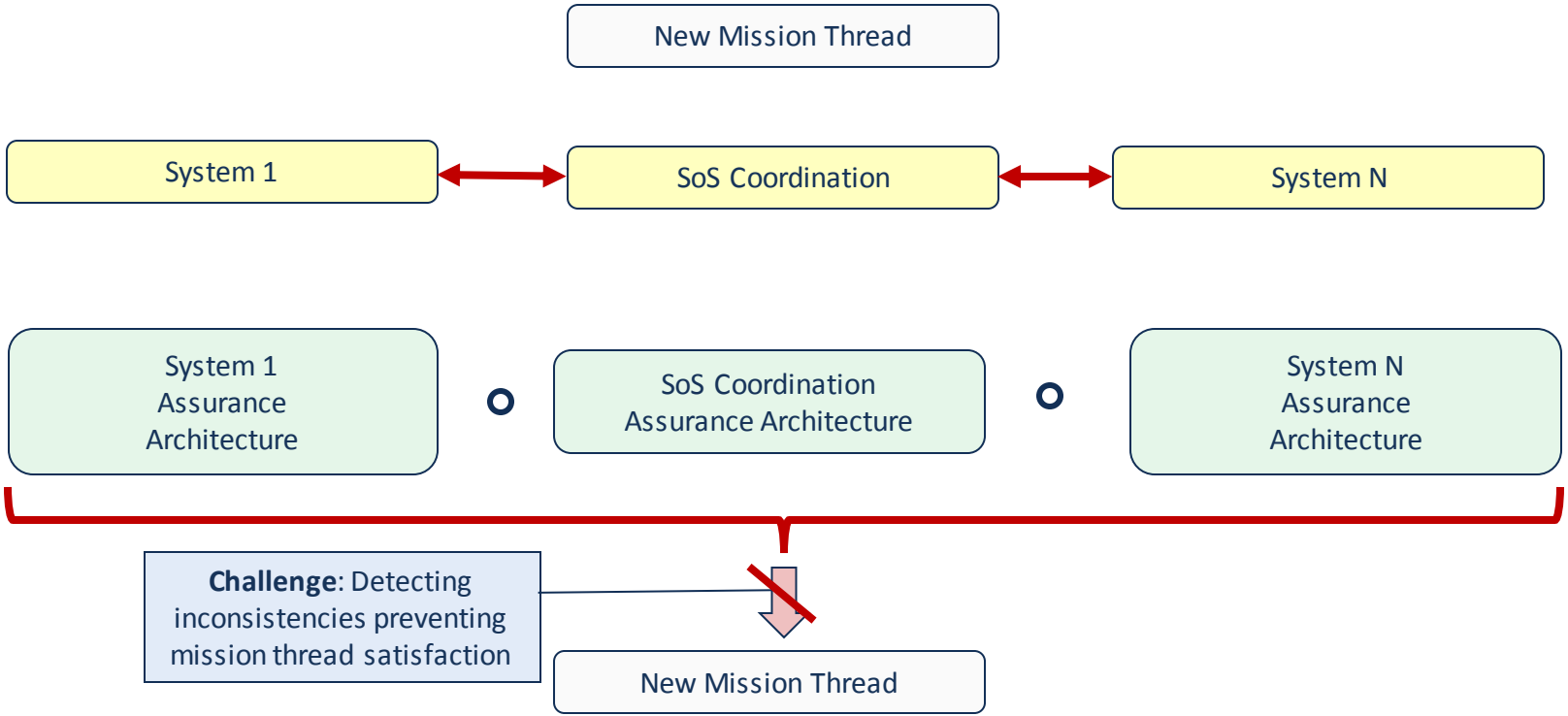
LSA Approach



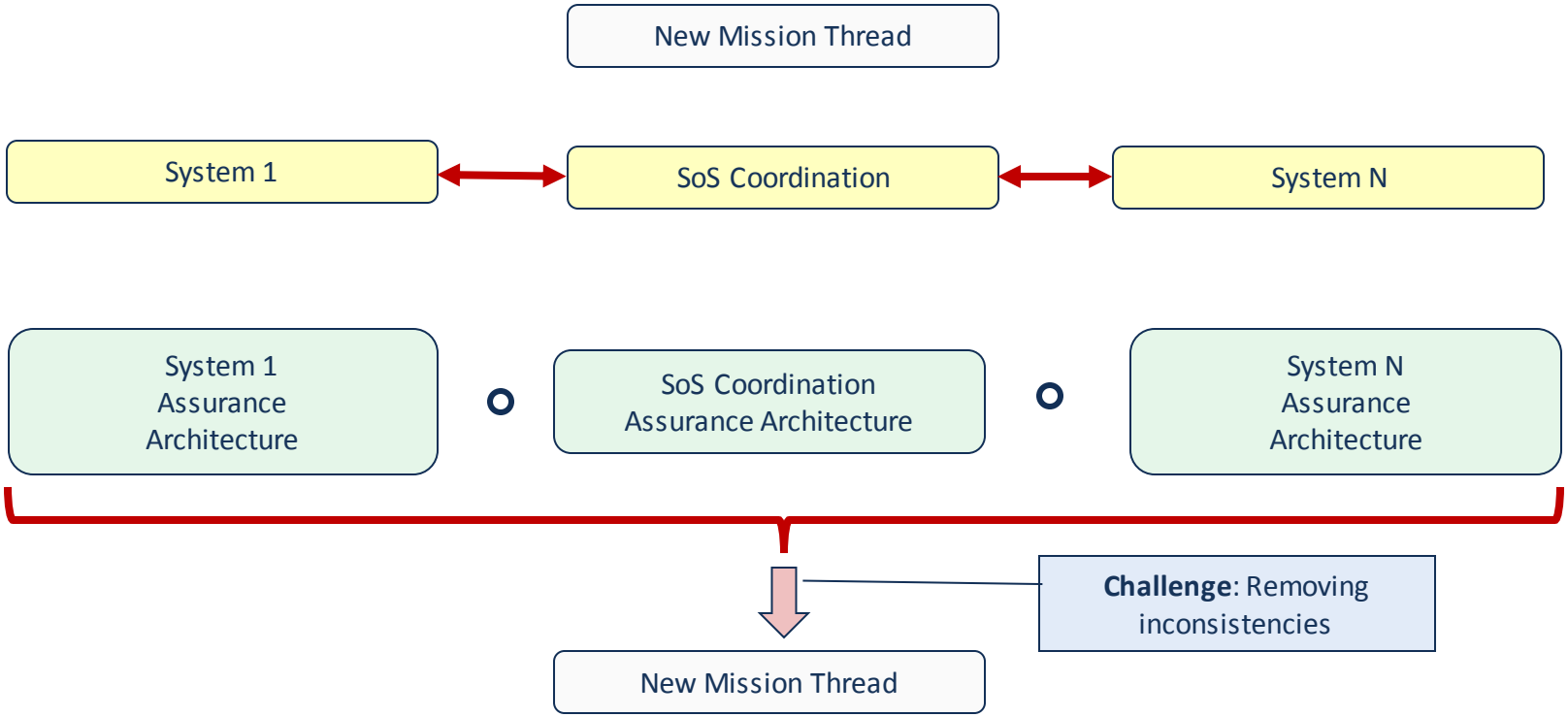
Challenges



Challenges



Challenges



	FY23 ETP	FY24 MTP	Beyond FY24
Model problem	• Start with existing in-house system • Control, timing, security • DoD Collaborator sandbox	• Representative of DoD Collaborator • Unclassified model problem related to classified parts of DoD Collaborator	• SoS
Representing assurance architectures	• Representing formal and informal • New definition of SAT	• Representative assurance architecture • Representing composed assurance architectures	
Runtime conformance	• Determining conformance	• Develop example of using assurance argument for ML • TBD	• Runtime enforcement • Runtime adaptation
Refining assurance architectures	• Using scientific ML • Thoughts on generalizing		• Incorporating runtime experience
Composing assurance arguments		• Detecting inconsistencies • Resolving inconsistencies	• Dynamic composition • System recommendations

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