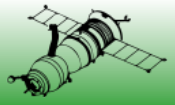
INCOSE International Symposium
Las Vegas, NV
June 30 - July 3, 2014

Synthesizing and Specifying Architectures for System of Systems

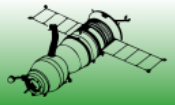
C. Robert Kenley, Timothy M. Dannenhoffer, Paul C. Wood, and Daniel A. DeLaurentis

Purdue University

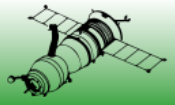

Acknowledgement

- This publication was developed under work supported by the US Missile Defense Agency (MDA) under contract No. HQ0147-10-C-6001. It has been reviewed by MDA and approved for public release (13-MDA-7638, 14 December 13). The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the US Missile Defense Agency. The US Missile Defense Agency does not endorse any products or commercial services mentioned in this publication.


A common question about SoS

- What is it that I should be doing for systems of systems that is different from what I always have done when engineering a system?

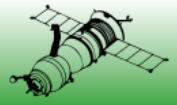




Our answer comes in two parts

- Part 1
  - Experience-based practices for generating and evaluating C2BMC architectures
- Part 2
  - Review of applicable model-based systems engineering methods
  - Showing how model-based methods apply to our C2BMC example

Experience-based practices for generating and evaluating C2BMC architectures

PART 1


A Missile Defense System of Systems

- US Ballistic Missile Defense System (BMDS)
  - Land-, sea-, air-, and space-based assets
  - "Acknowledged" system of systems (Dahmann and Baldwin 2008)
    - Objectives, management, funding, and authority are established for the system of systems
    - The participating systems retain their own management, funding, and authority in parallel

Reference Process for Synthesizing Architectures

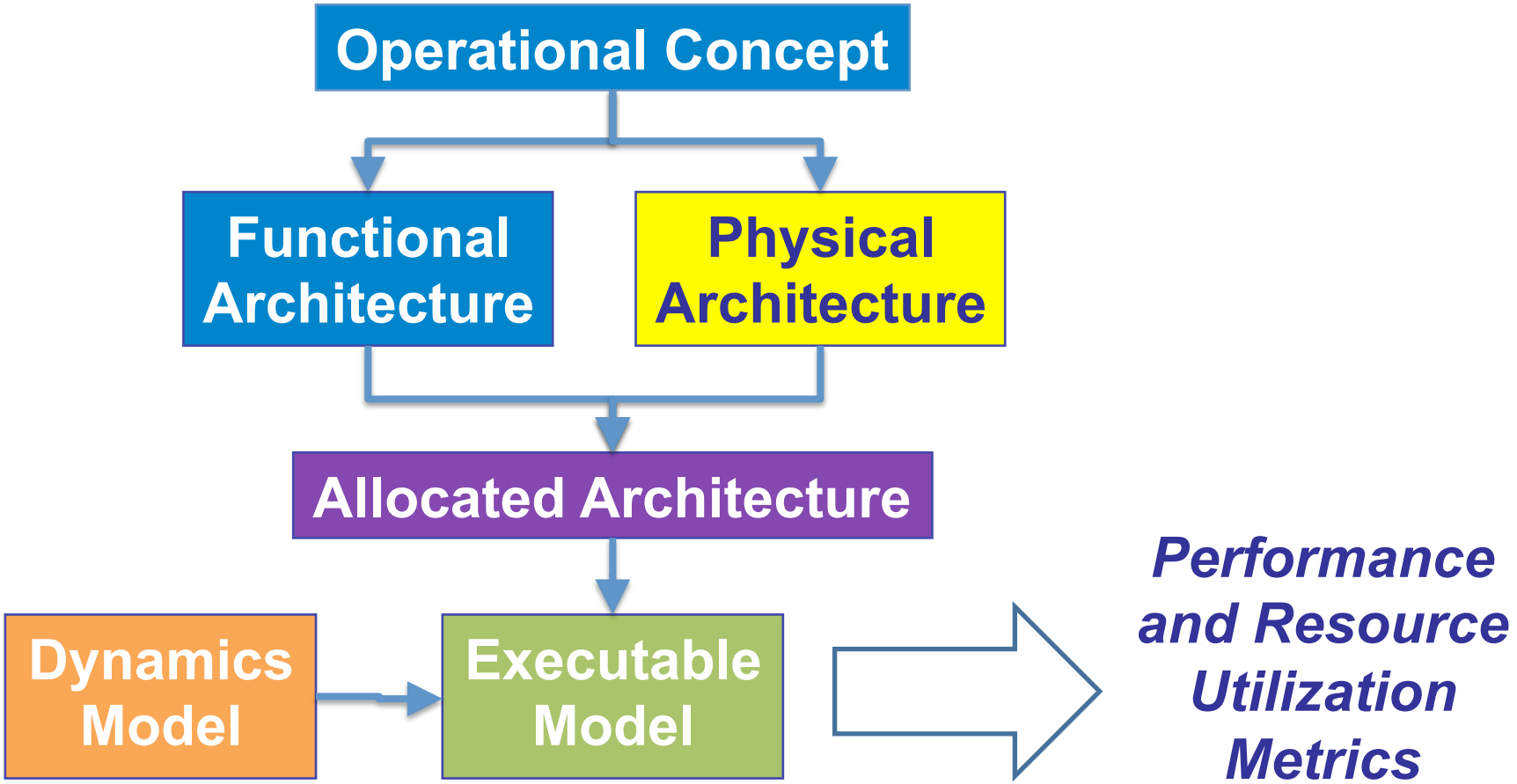
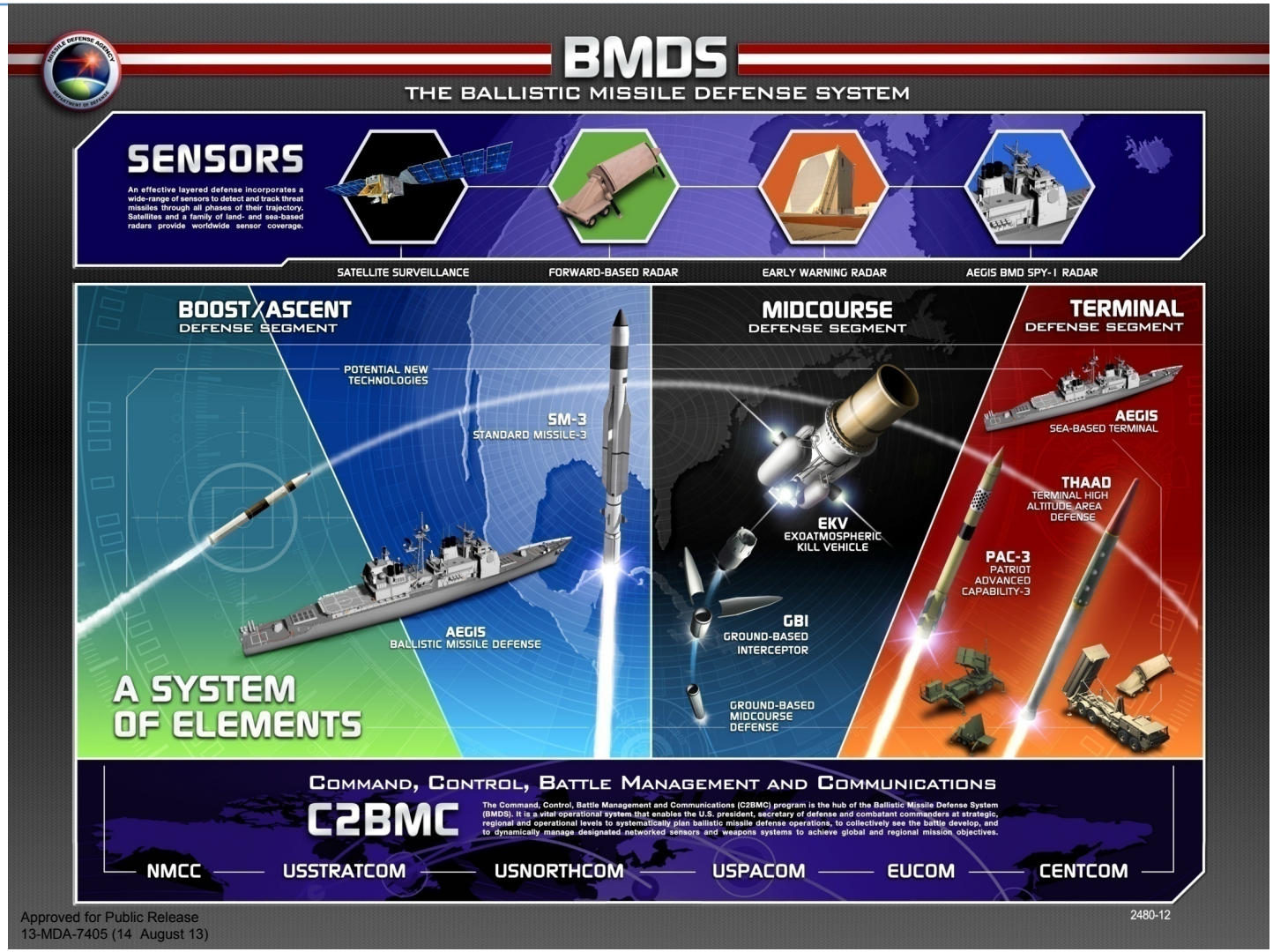
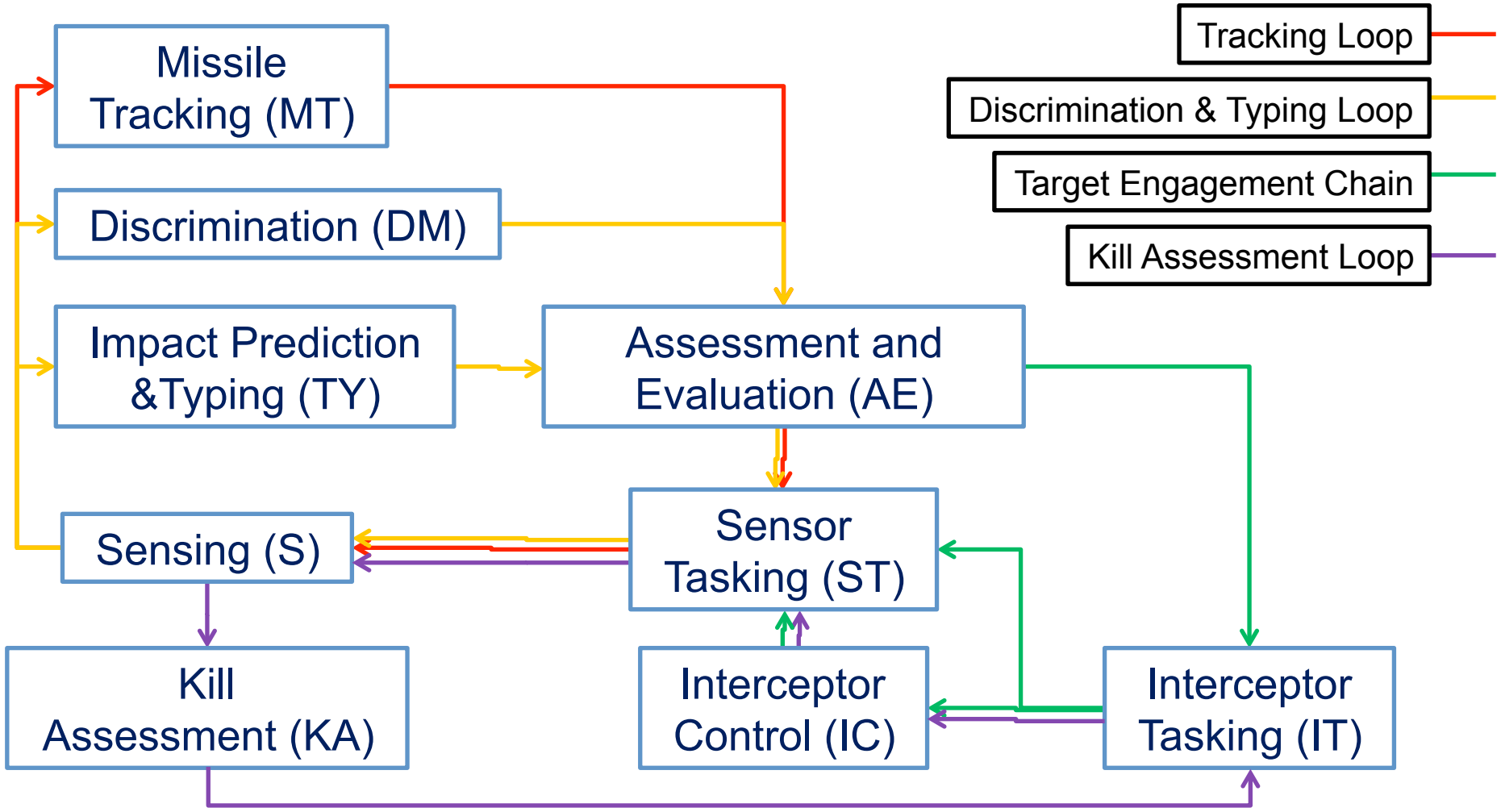


Figure adapted from Levis, Alexander H., and Lee W. Wagenhals. 2000. "C4ISR architectures: I. Developing a process for C4ISR architecture design." *Systems Engineering* no. 3 (4):225-247.

BMDS Operational Concept



Functional Architecture: Control and Information Flow

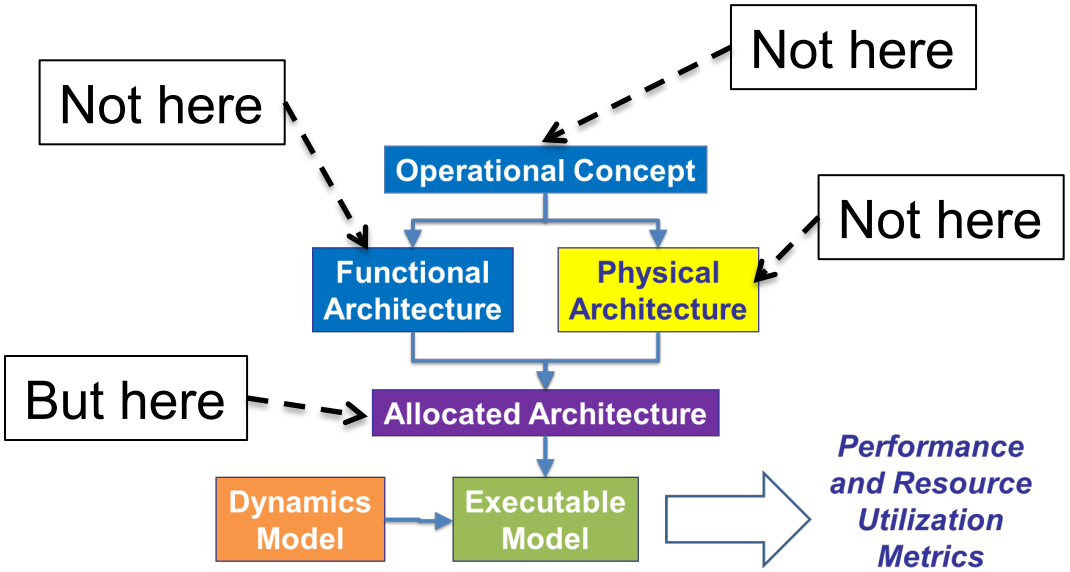


Physical Architecture: Platforms and Communications Links

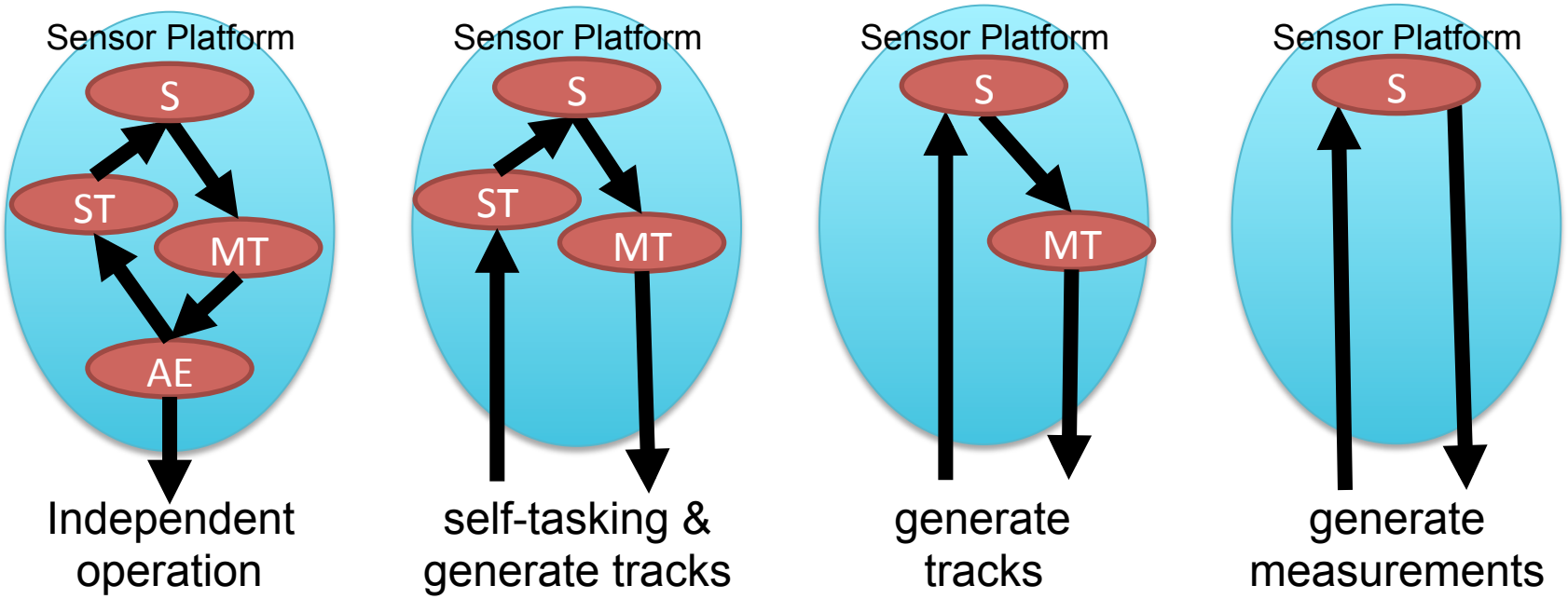
Class	Physical Entity	Relevant Attributes
Platform	• Aircraft • Satellite • Ground Station • C2 Node • Interceptor	• Location and Trajectory • Processing Resources • Interfaces to Communications Links
Communications Link	• Satellite • Wireless • Fiber	• Communication Protocols and Capacities

When is the SoS distinction manifest in the process?

- It is in defining the allocated architecture that the distinguishing trait of operational independence is exhibited.



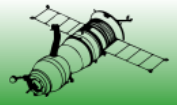
Allocated Architecture: Options for Allocating Functions to a Sensor Platform



Allocated Architecture: Example of Centralized vs. Decentralized Tracking

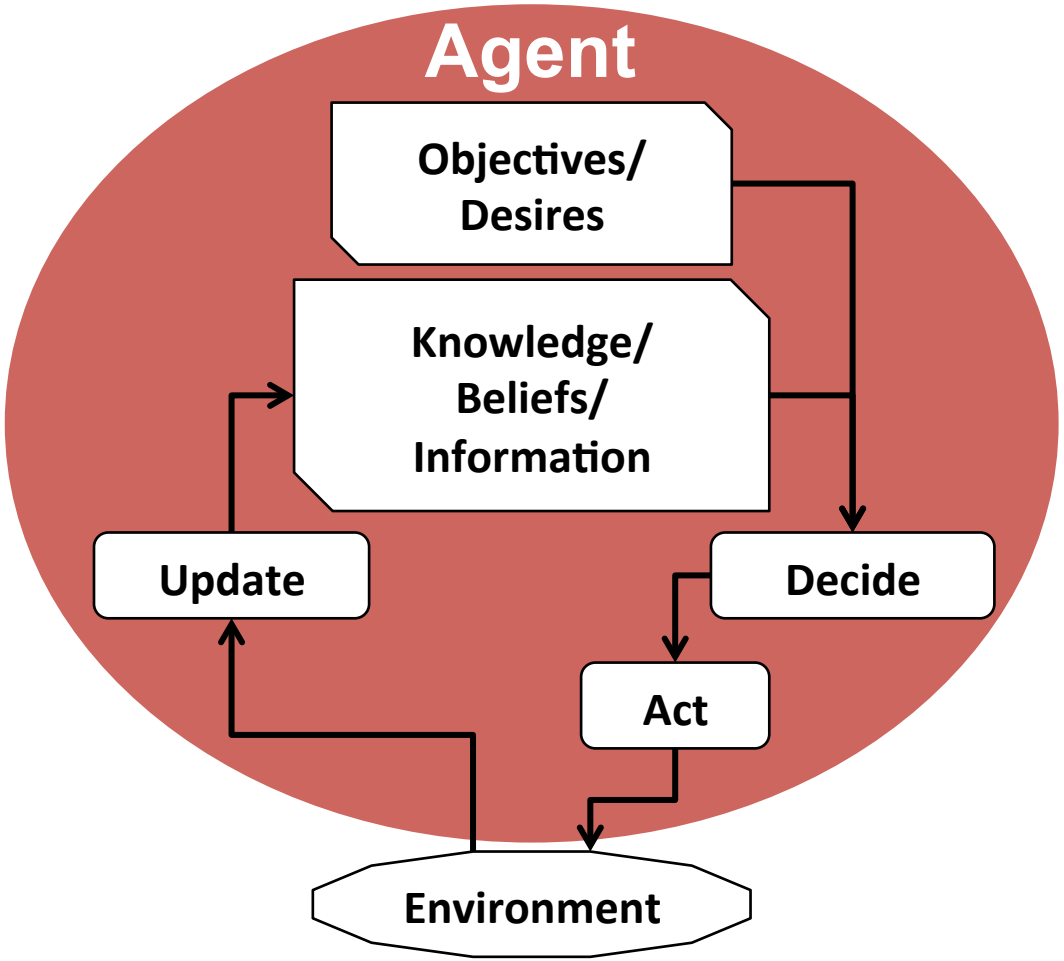
INCOSE International Symposium
Las Vegas, NV
June 30 - July 3, 2014

		Location of Functionality According to Architecture Centralization			
		Centralized	Centralized Tracking and Prioritization	Centralized Tracking	Decentralized
Functions	Missile Tracking (MT)	C2	C2	C2	Sensors
	Assessment and Evaluation (AE)	C2	C2	Sensors	Sensors
	Sensor Tasking (ST)	C2	Sensors	Sensors	Sensors
	Sensing (S)	Sensors	Sensors	Sensors	Sensors

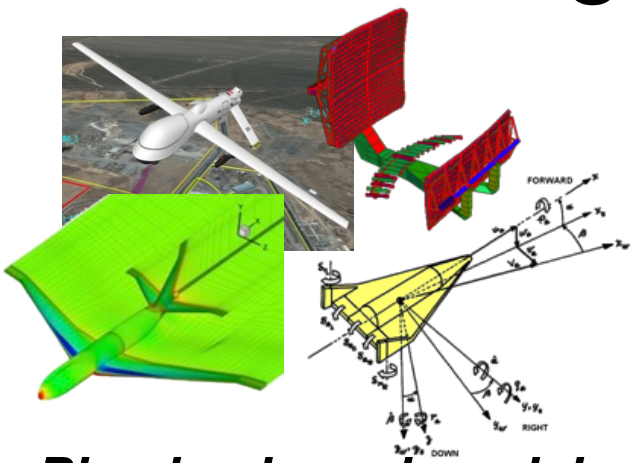

Agent-Based Dynamics Model

- Modeling functions as agents captures operational independence

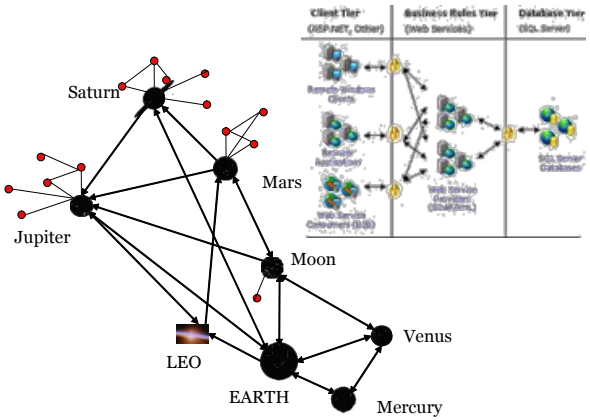


Executable Model: Discrete Agent Framework (DAF)

INCOSE International Symposium
Las Vegas, NV
June 30 - July 3, 2014



+



=

DAF

Physics-based models

Architecture-based models

- Individual system behavior
  - Physics-based and heuristic-based behavior models
- Architecture of systems or systems-of-systems
  - Modes and types of interactions across multiple system types (e.g. human, technological, etc.)
  - Interdependencies between systems (e.g., exchange of info, data, energy, etc.)
- *New knowledge via design of agents, their capabilities, and interaction rules*

Generating Communications Architectures

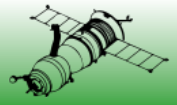
- Architecture for a system of systems is defined by interfaces [Maier (1998)]
- For C2BMC
  - Interfaces = Communications Network
  - Logical agent-to-agent connections prescribed by functional architecture
- SoS architect allocates agents to platforms to create architectures
- Physical network connections (communications architectures) must be defined for all logical connections

What Our Model Builder Does

- Architect specifies which agents are to be logically connected, ignoring complexities of physical network paths
- Architect specifies constraints and assumptions for physical network (e.g., each ground station is connected to only a single type of sensor)
- Model builder automatically creates physical communication paths between agents based on a shortest path algorithm
  - Distance can be defined in several ways (number of links, or total time to transmit, which favors fiber connections over lower speed links)
- Benefits
  - Reduces bookkeeping burden and errors
  - Increases productivity and coverage (large number of architectures can be created for evaluation)

- Review of applicable model-based systems engineering methods
- How the methods apply to our C2BMC example

PART 2


Desiderata for Specifying SoS Using MBSE

- MBSE methods that specify SoS dynamics models and executable models must support
  - Agent-based modeling of actions
  - Interactions of actors who perform concurrent, asynchronous activities

Using UML for Agent-Based Modeling [Park, Kim, and Lee (2000)]

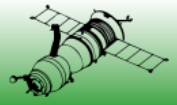
Intra-agent Models

Model	Approach
Goal	Object model of a goal hierarchy
Belief	Object model of beliefs and external message protocols
Plan	Update beliefs; and determine actions to take and messages to send
Capability	Logic for actions to be taken by the agent

Inter-agent Models

Model	Approach
Agent Mobile	Define how an agent coordinates its actions to perform a task with other agents (assumes a coordinator agent)
Agent Communication	Define how messages are exchanged between agents including sequence diagram of agent actions and messages

- Based on UML 1.1: does not assume complete autonomy among the agents nor does it assume concurrency

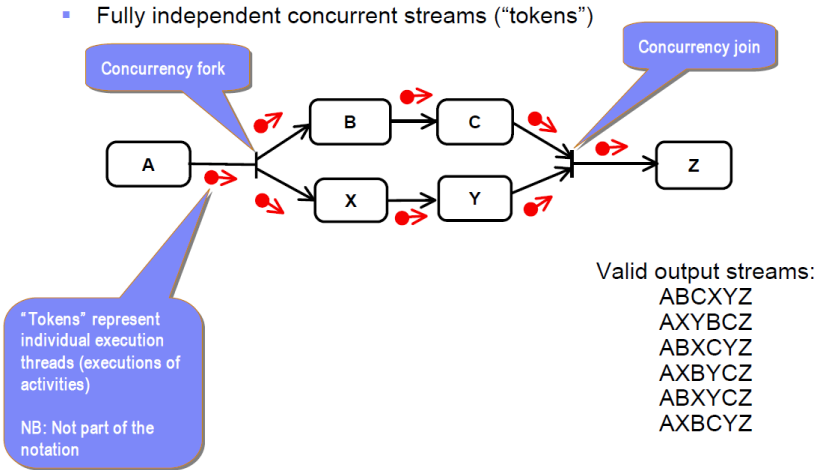




Mapping Dynamics Models to Executable Petri Net Models

- Petri nets
  - Executable models for simulating interactions of concurrent, asynchronous activities
- Pre-UML 2.0 Examples
  - Mapping a business-process workflow model of the dynamics of a biological system to a Petri net [Peleg, Yeh, and Altman (2002)]
  - Converting a UML 1.3 specification for the dynamics of a C4ISR system to a colored Petri net [Wagenhals, Haider, and Levis (2003)]

UML 2.0 to the Rescue

Figure from Quatrani's 2005 "Introduction to UML 2.0"



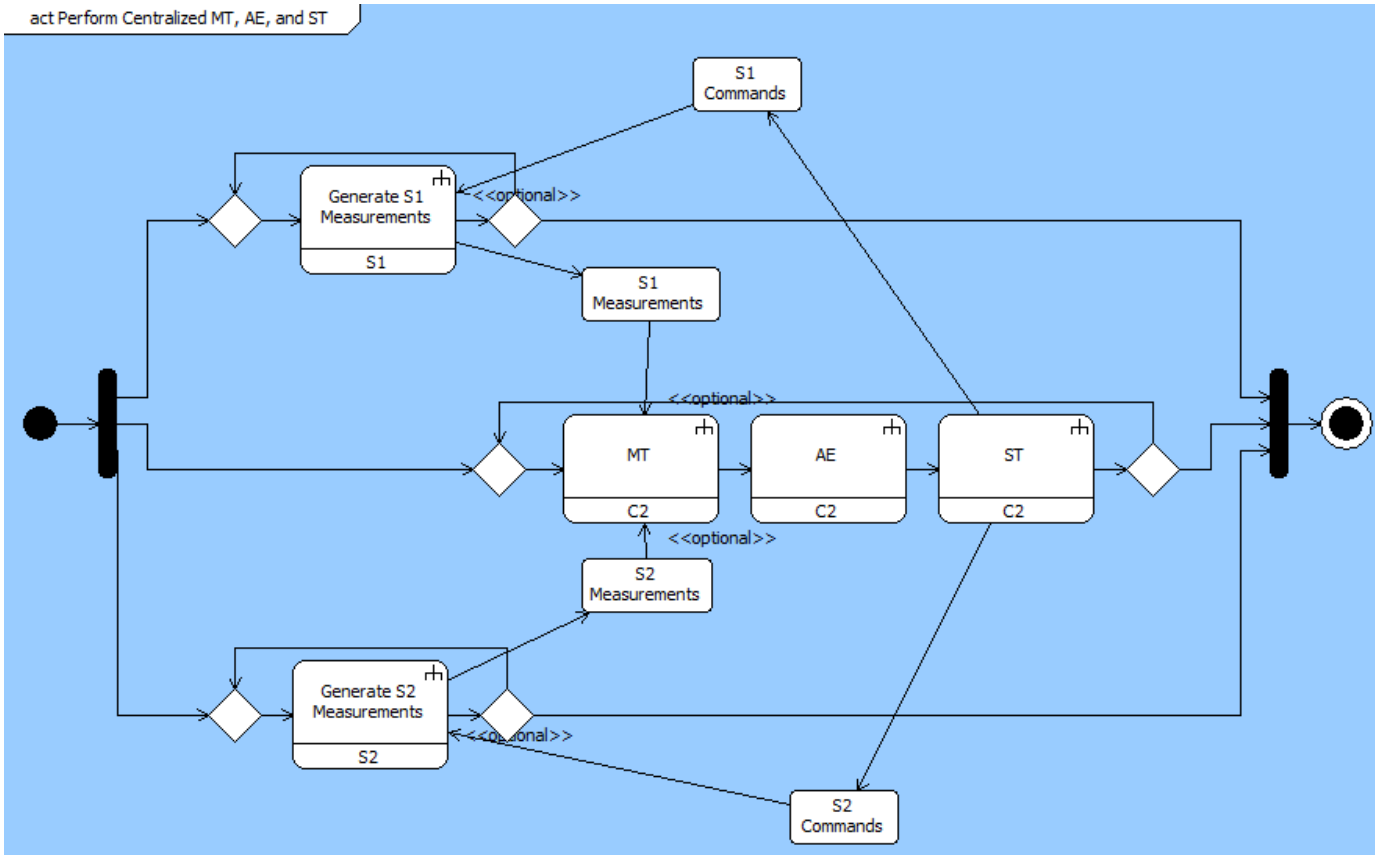
Claims in the UML 2.0 spec

- "Petri-like semantics instead of state machines" to allow for concurrency that includes tokens [OMG, OMG Unified Modeling Language: Superstructure (final adopted spec, version 2.0, 2003-08-02), Technical report, Object Management Group (2003)]

UML 2.0 and Petri Nets

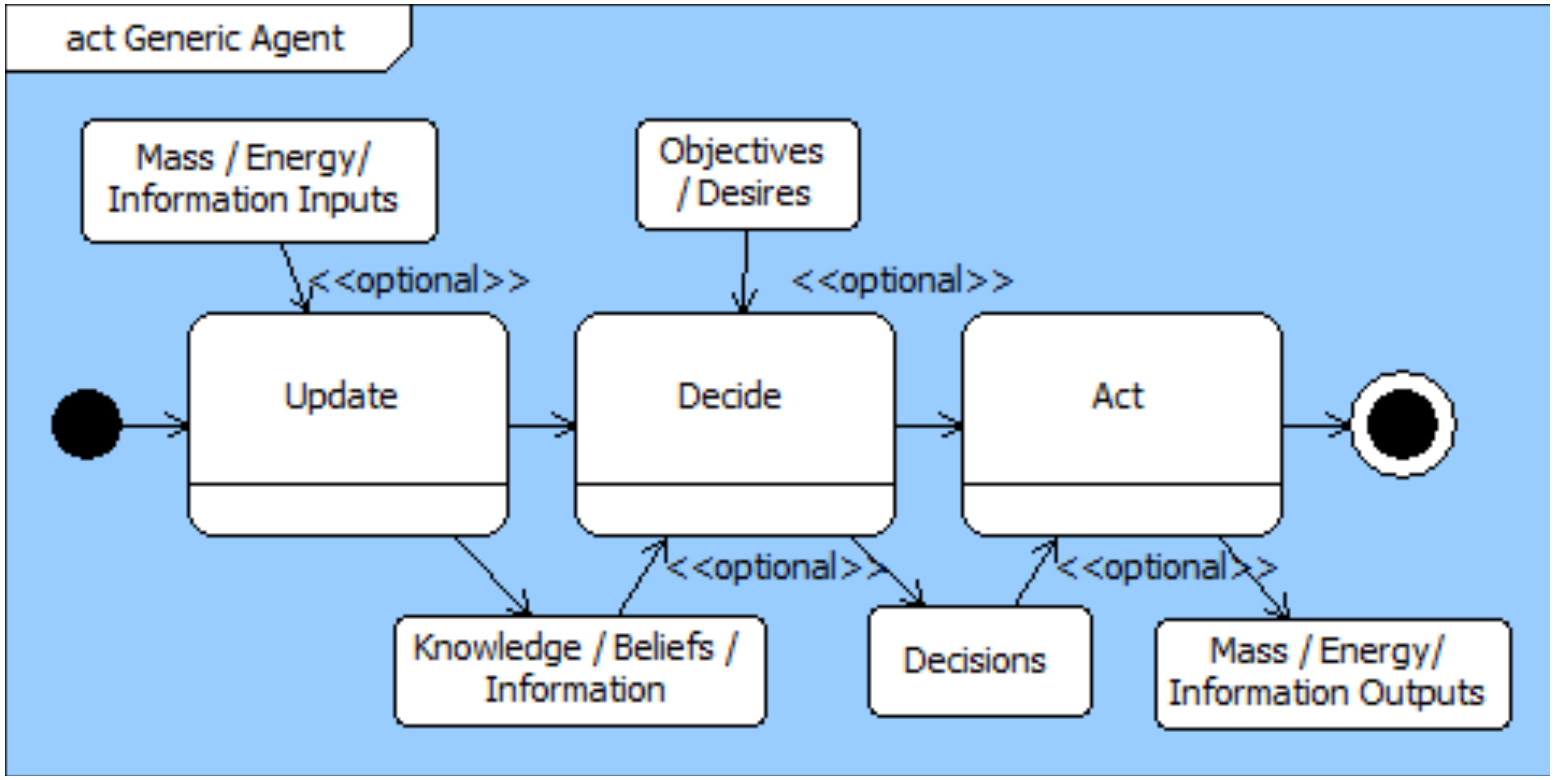
- Mapping UML 2.0 activity diagrams to
  - Colored Petri nets [Störrle (2005)]
  - Fundamental Modeling Concepts version of Petri net diagram [Staines (2008)]
- Proposal to extend UML [Sinclair (2009)]
  - Add explicit UML constructs for hierarchical and timed colored Petri nets
  - Purpose is to enable modeling and simulation of system of systems

UML Activity Diagram for Completely Centralized Tracking Architecture

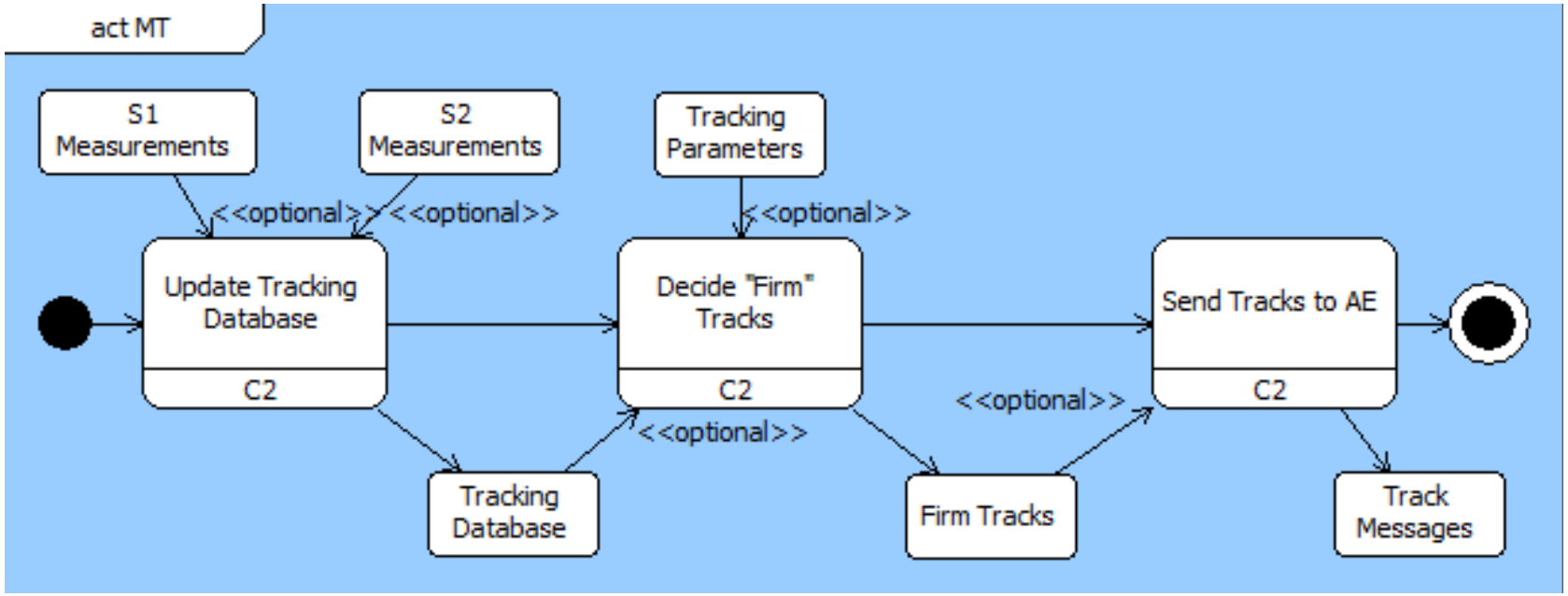


S1= Sensor 1, S2 = Sensor 2, MT = Missile Tracking, AE = Assessment and Evaluation, ST = Sensor Tasking, C2 = Command and Control

UML Activity Diagram for Generic Agent



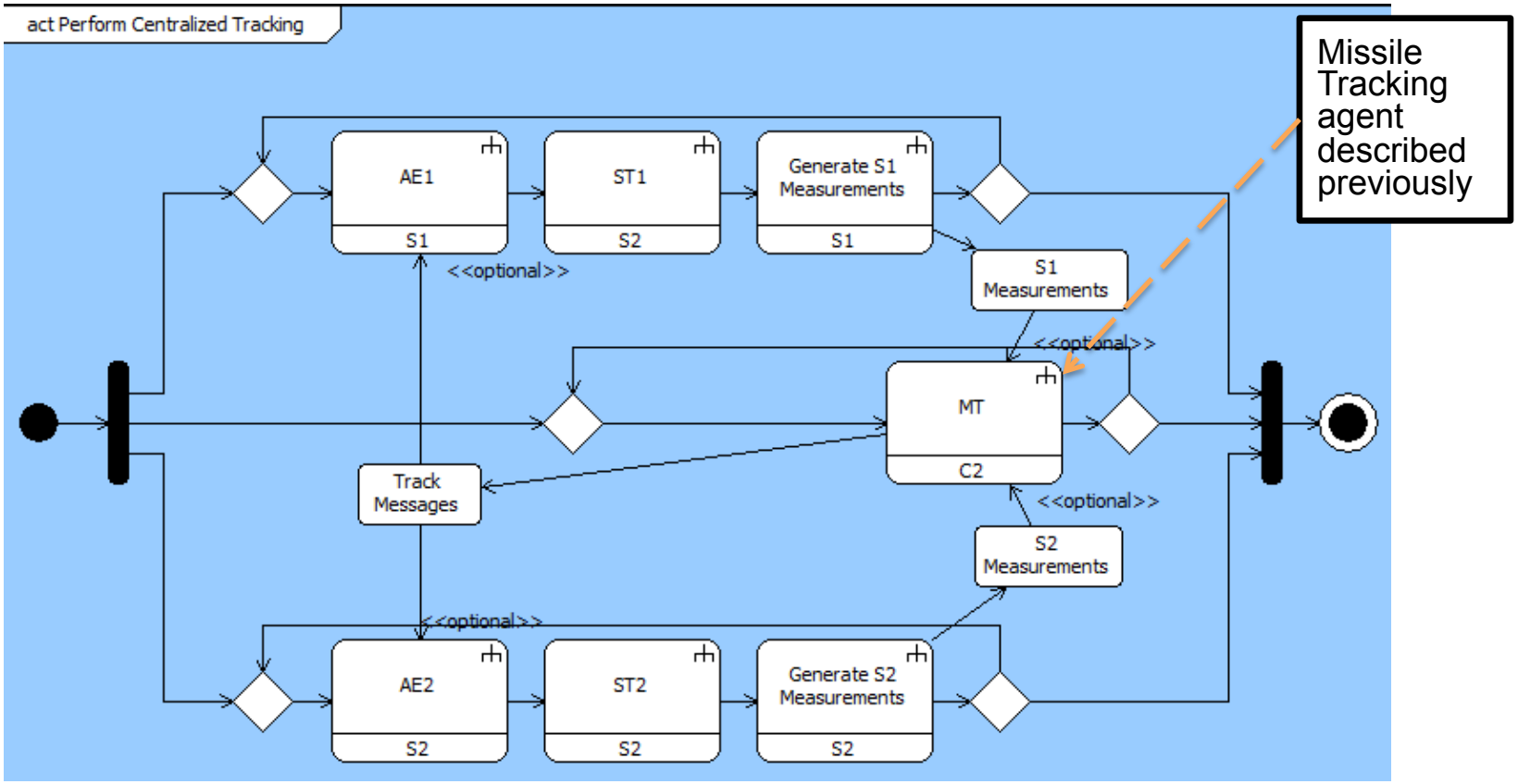
UML Activity Diagram for Missile Tracking Agent



Mapping of Generic Agent to Missile Tracking Agent

Generic Agent Item	Missile Tracking Agent Item
Mass / Energy / Information Inputs	S1 and S2 Measurements
Update	Update Tracking Database
Knowledge / Beliefs / Information	Tracking Database
Objectives / Desires	Tracking Parameters
Decide	Decide “Firm” Tracks
Decisions	Firm Tracks
Act	Send Tracks to AE
Mass / Energy / Information Outputs	Track Messages

UML Activity Diagram for Centralized MT with Distributed AE and ST



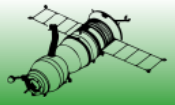
Sn= Sensor *n*, MT = Missile Tracking, AEn = Assessment and Evaluation *n*, STn = Sensor Tasking *n*, C2 = Command and Control

What We Have Done

- Applied “traditional” systems architecting process to SoS
- Discovered that the dynamic modeling of a SoS is key step in applying the process
  - Used agent-based modeling to capture emergent behavior that derives from complex interactions of systems of systems.
- Developed methods to ease burden of manually synthesizing network architectures
- Developed a “pattern” for agent-based models using UML activity diagrams to specify the independently operating constituent systems within SoS

What Next?

- Investigate the details of going from UML activity diagrams to executable models
  - Agent-based modeling tools such as Purdue's Discrete Agent Framework
  - Petri-net modeling tools
- Look at usefulness of other UML constructs
  - Executable models based on state machine diagrams


C. Robert Kenley, PhD, ESEP
Associate Professor of Engineering Practice
School of Industrial Engineering, Purdue University
315 N Grant St, West Lafayette, IN, 47907-2023
Phone: +1 765 494 5160 • Mobile Phone: +1 765 430 3774
E-mail: kenley@purdue.edu
Web: http://web.ics.purdue.edu/~ckenley/

THANK YOU