Lifecycle Modeling – Application to Architecture Development

Steven H. Dam, Ph.D., ESEP
President, SPEC Innovations
571-485-7805
Steven.dam@specinnovations.com

October 27, 2011

Overview

- Why a New Language?
- Lifecycle Modeling Language Overview
- Use of LML for Architecture to Systems Design Specification
- Use of LML in Test and Evaluation
- Use of LML in Operations and Support
- Summary

SPEC INNOVATIONS

WHY A NEW LANGUAGE?

We already have SysML … what else do you need!

State of Current "Languages"

- In the past decade, the Unified Modeling Language (UML) and now the profile Systems Modeling Language (SysML) have dominated the discussion
- Why?
  - Perception that software is "the problem"
  - Hence need for an "object" approach
- SysML was designed to relate systems thinking to software development, thus improving communication between systems engineers (SE) and software developers

SPEC INNOVATIONS

Why Objects Are Not the Answer

- Although SysML *may* improve the communication of design between SEs and the software developers it does not communicate well to anyone else
  - No other discipline in the lifecycle uses object oriented design and analysis extensively
  - Users in particular have little interest/acceptance of this technique
  - Software developers who have adopted Agile programming techniques want functional requirements (and resent SEs trying to write software)

SPEC INNOVATIONS

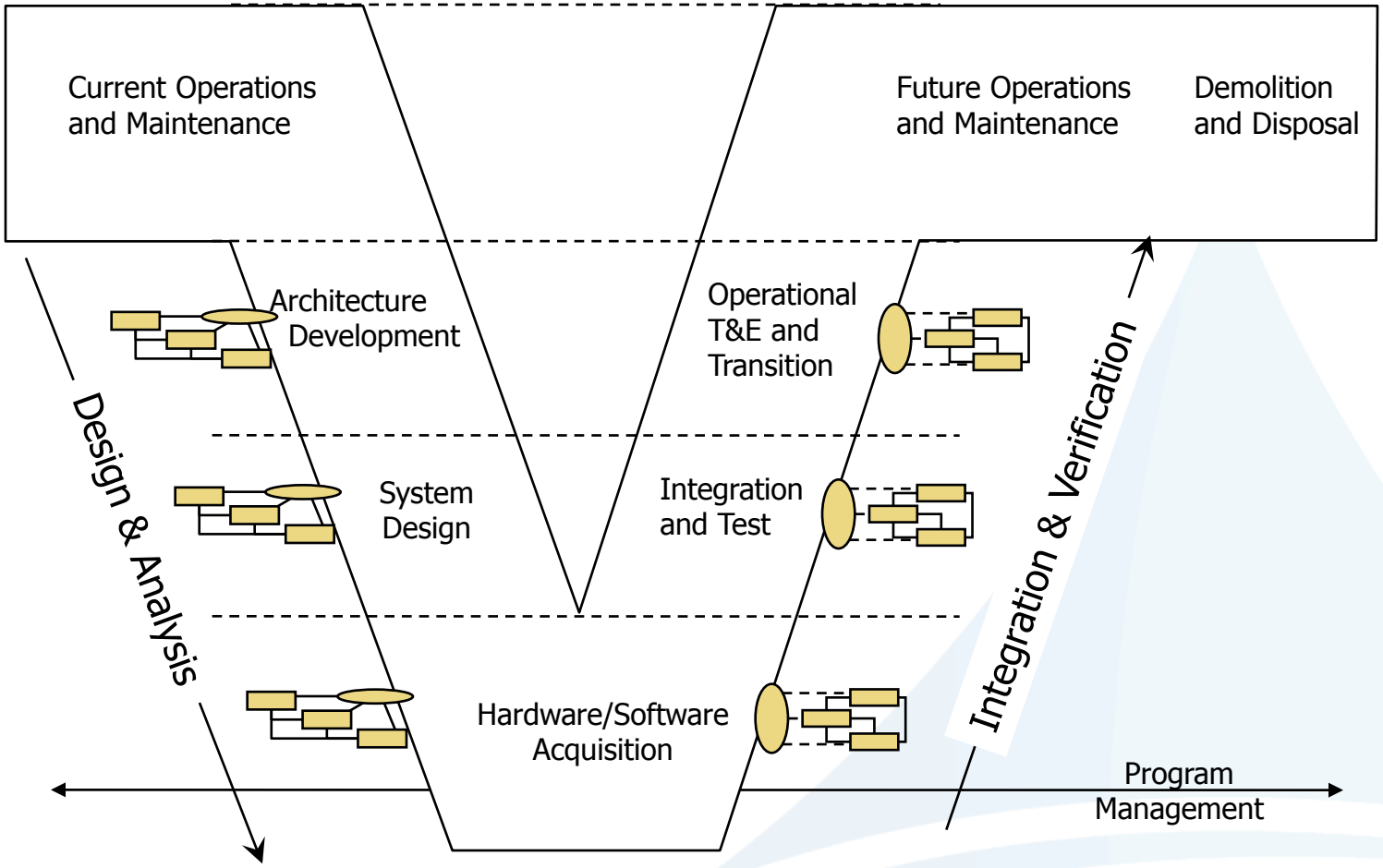
So What Do We Do?

- Recognize that our primary job as SEs is to communicate between all stakeholders in the lifecycle
- Be prepared to translate between all the disciplines
- Reduce complexity in our language to facilitate communication

SPEC INNOVATIONS

LIFECYCLE MODELING LANGUAGE (LML) OVERVIEW

The Lifecycle



Lifecycle Modeling Language (LML)

- LML combines the logical constructs with an ontology to capture information
  - SysML – mainly constructs – limited ontology
  - DoDAF Metamodel 2.0 (DM2) ontology only
- LML simplifies both the "constructs" and ontology to make them more complete, yet easier to use
- Goal: A language that works across the full lifecycle

SPEC INNOVATIONS

LML Ontology* Overview

*Ontology = Taxonomy + relationships among terms and concepts
** Taxonomy = Collection of standardized, defined terms or concepts

- Taxonomy**:
  - 12 primary element classes
  - Many types of each element class
    - Action (types = Function, Activity, Task, etc.)
- Relationships: almost all classes related to each other and themselves with consistent words
  - Asset performs Action/Action performed by Asset
  - Hierarchies: decomposed by/decomposes
  - Peer-to-Peer: related to/relates

SPEC INNOVATIONS

LML Taxonomy Simplifies Classes

- Technical
  - Action
  - Artifact
  - Asset
  - Characteristic
  - Input/Output
  - Link
  - Statement
- Programmatic/Technical
  - Cost
  - Issue
  - Location
    - Physical, Orbital, Virtual
  - Risk
  - Time
    - Duration, Timeframe, Point-in-Time

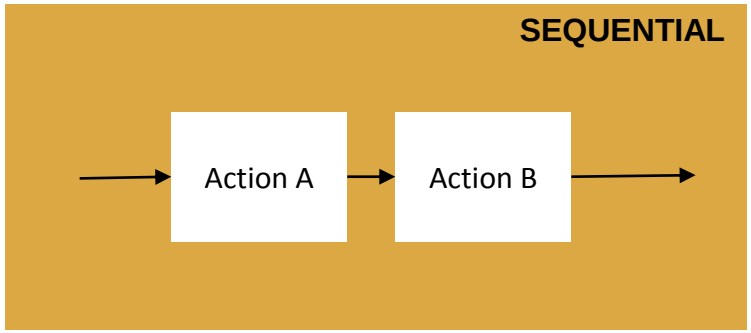
SPEC INNOVATIONS

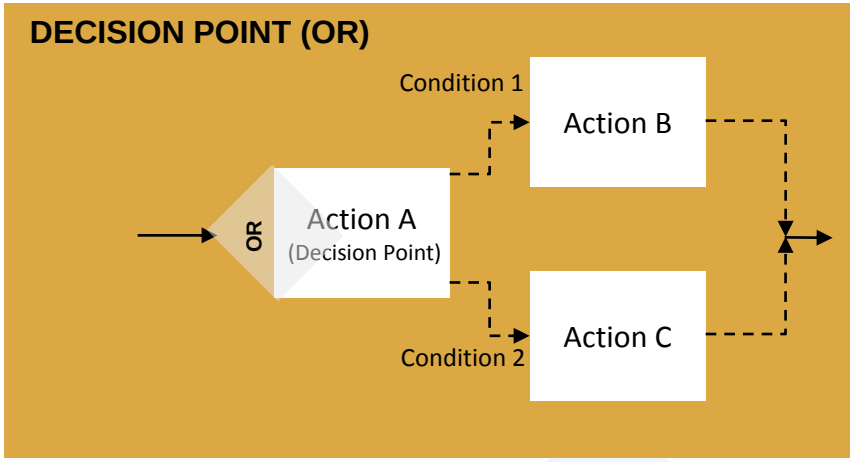
LML Relationships Provide Linkage Needed Between the Classes

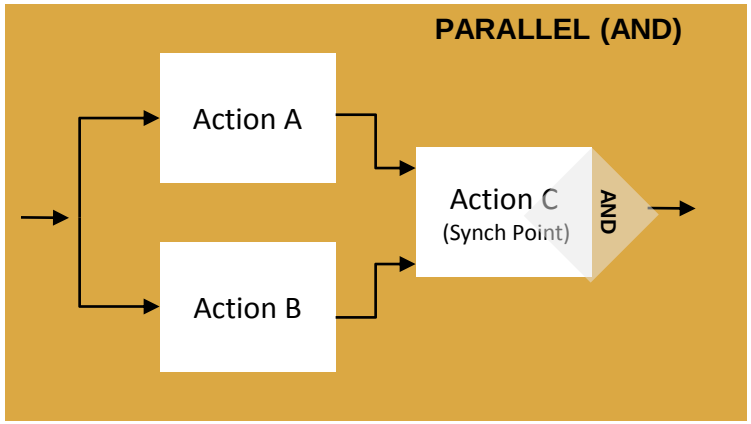
	ACTION	ARTIFACT	ASSET	CHARACTERISTIC	COST	INPUT/OUTPUT	ISSUE	LINK	LOCATION	RISK	STATEMENT	TIME	
ACTION	decomposed by related to	references	captures consumes preformed by produces	specified by	incurs	generates receives	causes resolves	-	located at	causes mitigates resolves	based on	takes occurs	ACTION
ARTIFACT	referenced by	decomposed by related to	referenced by	specified by referenced by	incurs referenced by	referenced by	causes referenced by	defines protocol for referenced by	located at	causes mitigates	based on referenced by	occurs	ARTIFACT
ASSET	captured by consumed by performs produced by	references	decomposed by orbited by related to	specified by	incurs	-	causes resolves responds to	connected by	located at	causes mitigates resolves	based on	occurs	ASSET
CHARACTERISTIC	specifies	references specifies	specifies	decomposed by related to	incurs specifies	specifies	causes resolves	specifies	located at	causes mitigates resolves	based on specifies	occurs	CHARACTERISTIC
COST	incurred by	incurred by references	incurred by	incurred by specified by	decomposed by related to	incurred by	causes incurred by resolves	incurred by	located at	causes incurred by resolves mitigates	cased on incurred by	occurs	COST
INPUT/OUTPUT	generated by received by	references	-	specified by	incurs	decomposed by related to	causes resolves	transferred by	located at	causes mitigates resolves	based on	occurs	INPUT/OUTPUT
ISSUE	caused by resolved by	caused by references resolved by	caused by resolved by responded by	caused by resolved by	caused by incurs resolved by	caused by resolved by	causes decomposed by related to resolved by	caused by resolved by	located at	caused by mitigates causes	caused by resolved by	date resolved by decision due occurs	ISSUE
LINK	-	defined protocol by references	connects to	specified by	incurs	transfers	causes resolves	decomposed by related to	located at	causes mitigates resolves	based on	delayed by occurs	LINK
LOCATION	locates	locates	locates	locates	locates	locates	locates	locates	decomposed by related to	locates mitigates	based on locates	occurs	LOCATION
RISK	caused by mitigated by resolved by	caused by mitigated by references resolved by	caused by mitigated by resolved by	caused by mitigated by resolved by	caused by incurs mitigated by resolved by	caused by mitigated by resolved by	caused by causes resolved by	caused by mitigated by resolved by	located at mitigated by	causes decomposed by related to resolved by	caused by mitigated by resolved by	occurs	RISK
STATEMENT	basis of	basis of references sourced by	basis of	basis of specified	basis of incurs	basis of	causes resolves	-	basis of located at	causes located at mitigates resolves	decomposed by related to	occurs	STATEMENT
TIME	taken by occurred by	occurred by	occurred by	occurred by	occurred by	occurred by	date resolves decided by occurred by	delays occurred by	occurred by	occurred by mitigates	occurred by	decomposed by related to	TIME
	ACTION	ARTIFACT	ASSET	CHARACTERISTIC	COST	INPUT/OUTPUT	ISSUE	LINK	LOCATION	RISK	STATEMENT	TIME	

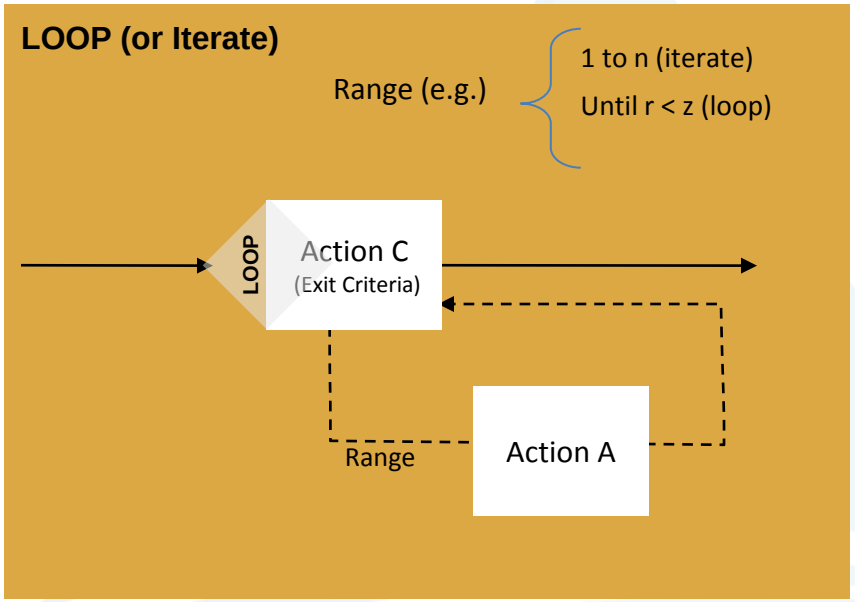
- decomposed by/decomposes
- orbited by/orbits
- related to/relates

LML Logic



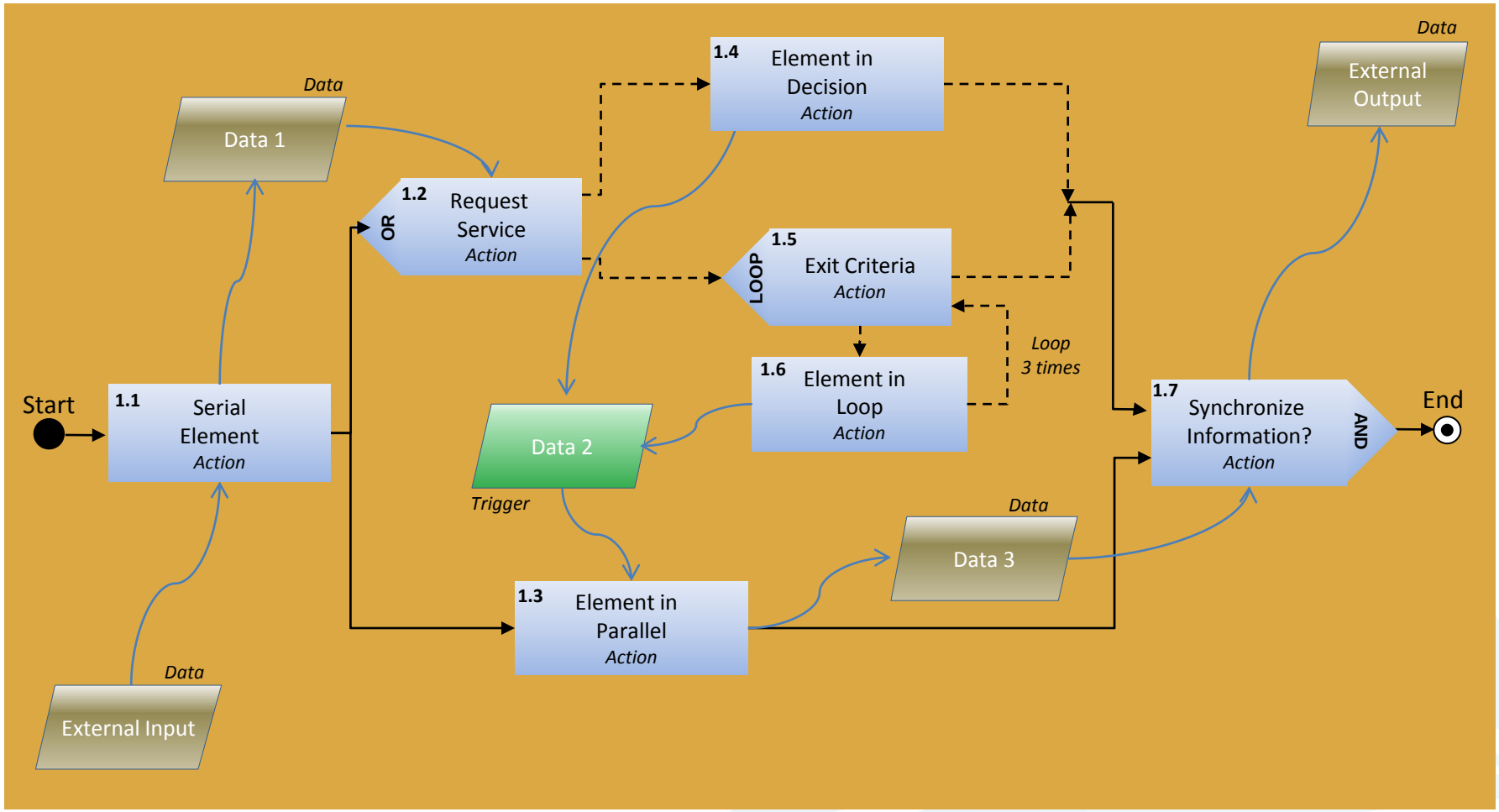




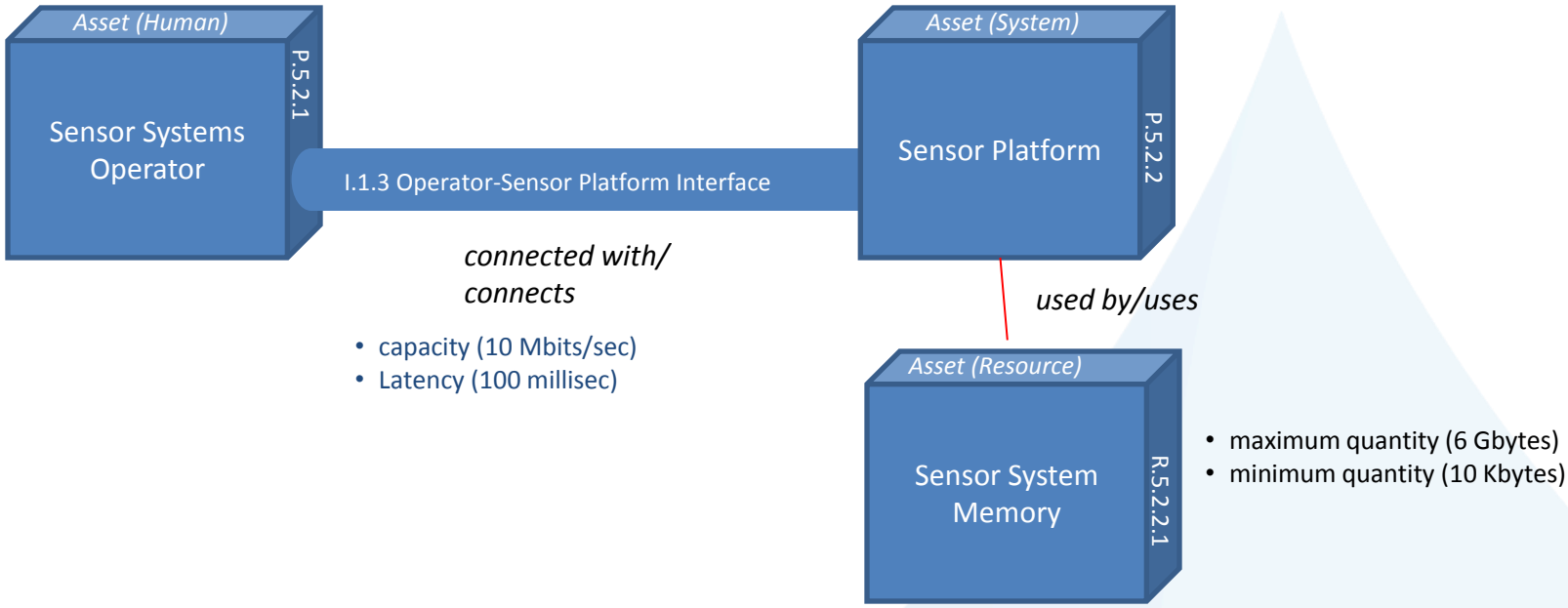


No constructs – only special types of Actions

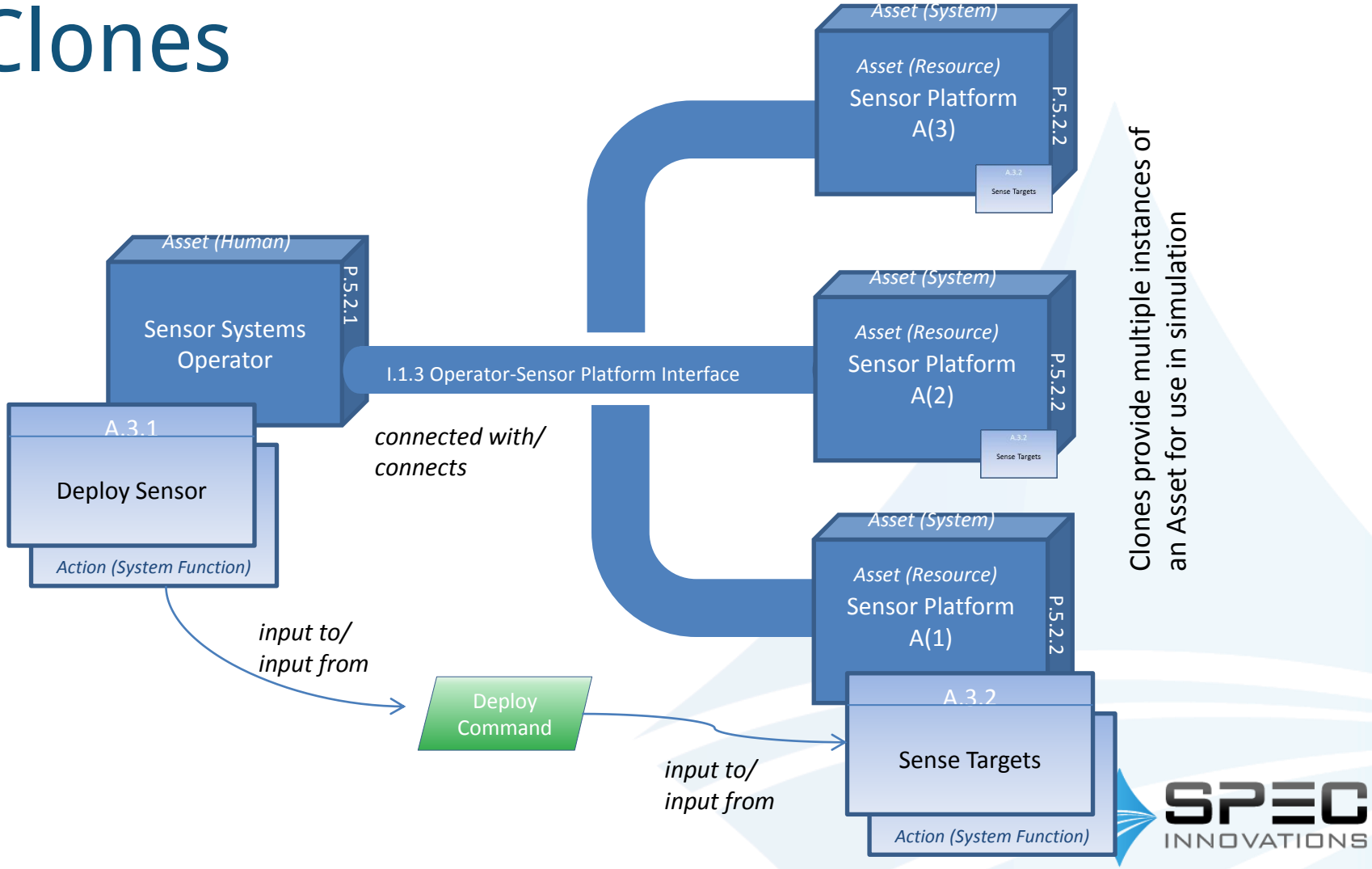
LML Action Diagram Captures Behavior



LML Physical Block Diagram



LML Combined Physical Behavior Diagram Enables Instances and Clones



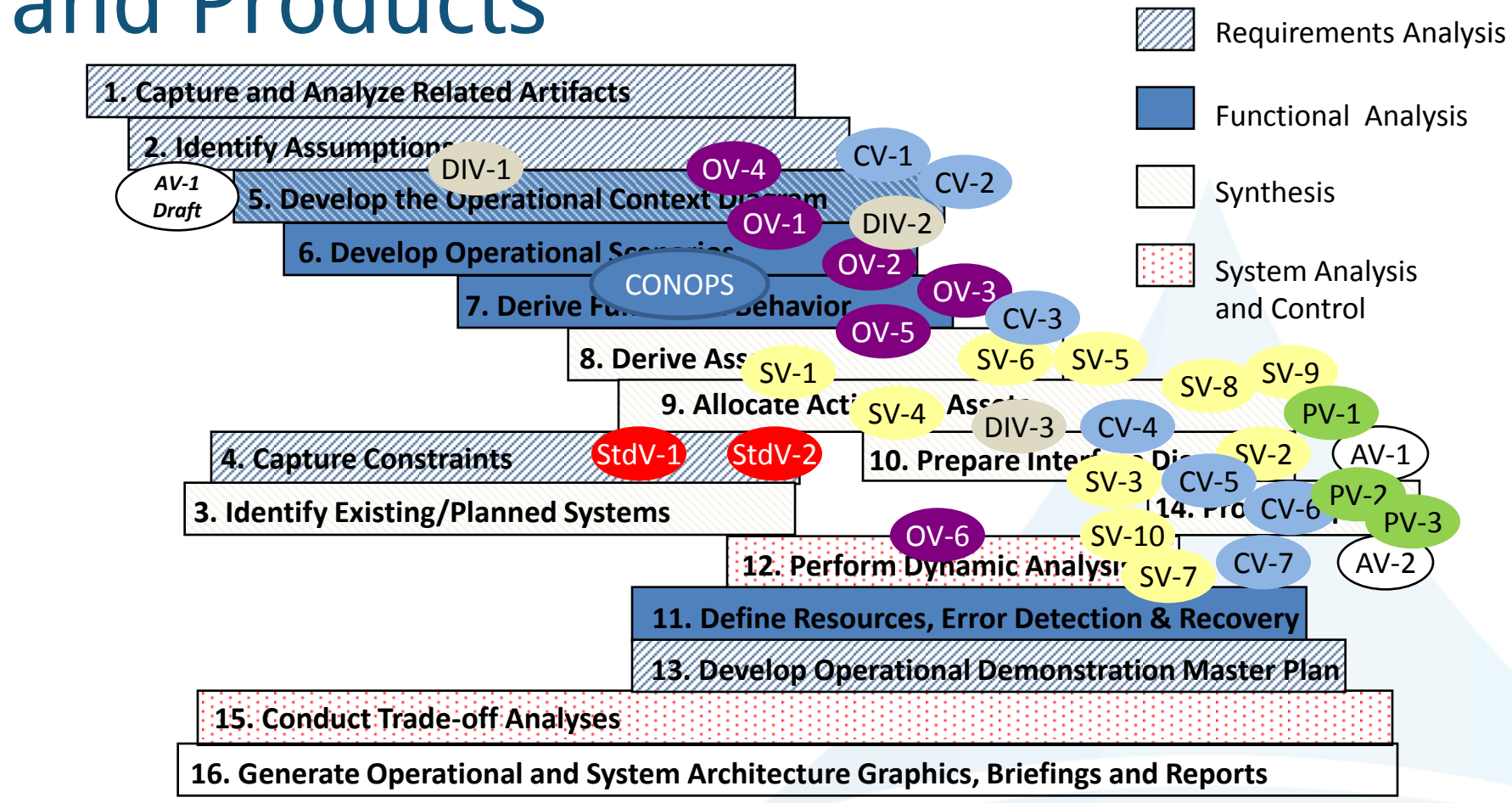
LML Summary

- LML contains the basic technical and programmatic classes needed for the lifecycle
- LML defines the Action Diagram to enable better definition of logic as functional requirements
- LML uses Physical Diagram to provide for abstraction, instances, and clones, thus simplifying physical models
- LML provides the “80% solution”
  - It can be extended to meet specific needs (e.g. adding Question and Answer classes for a survey tool that feeds information into the modeling)

SPEC INNOVATIONS

USE OF LML FOR ARCHITECTURE TO SYSTEMS DESIGN SPECIFICATION

Architecture Development Process and Products



This implementation of the middle-out approach has been proven on a variety of architecture projects

SPEC INNOVATIONS

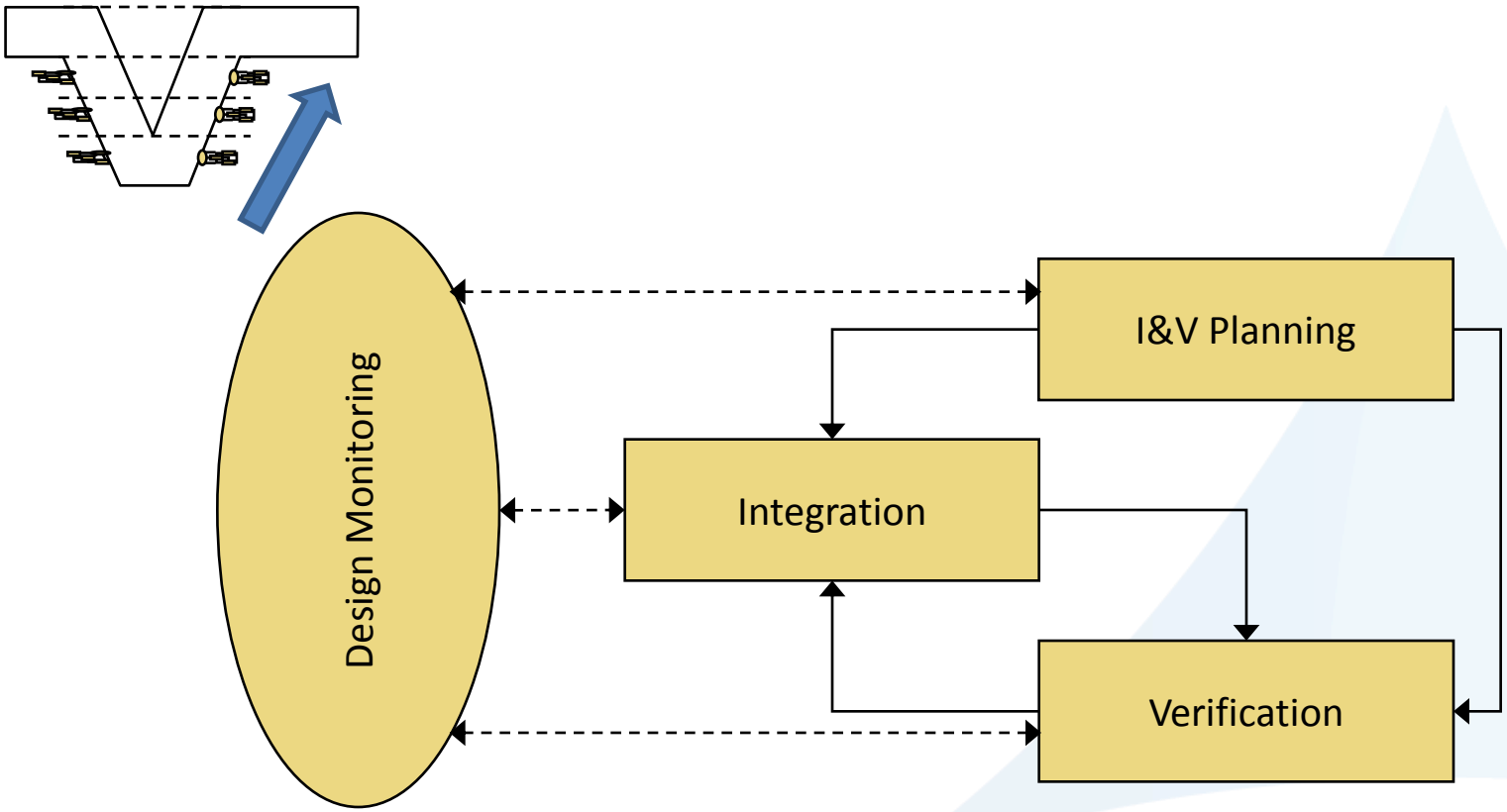
Key Architecture Products

- DoDAF Diagrams
- Concept of Operations (CONOPS)
- Functional Specifications of Hardware and Software
- Early Design Validation Through Modeling and Simulation
- Test and Evaluation Plans (for T&E)
- Processes and Procedures (for Operations and Support, as well as inputs to training plans)

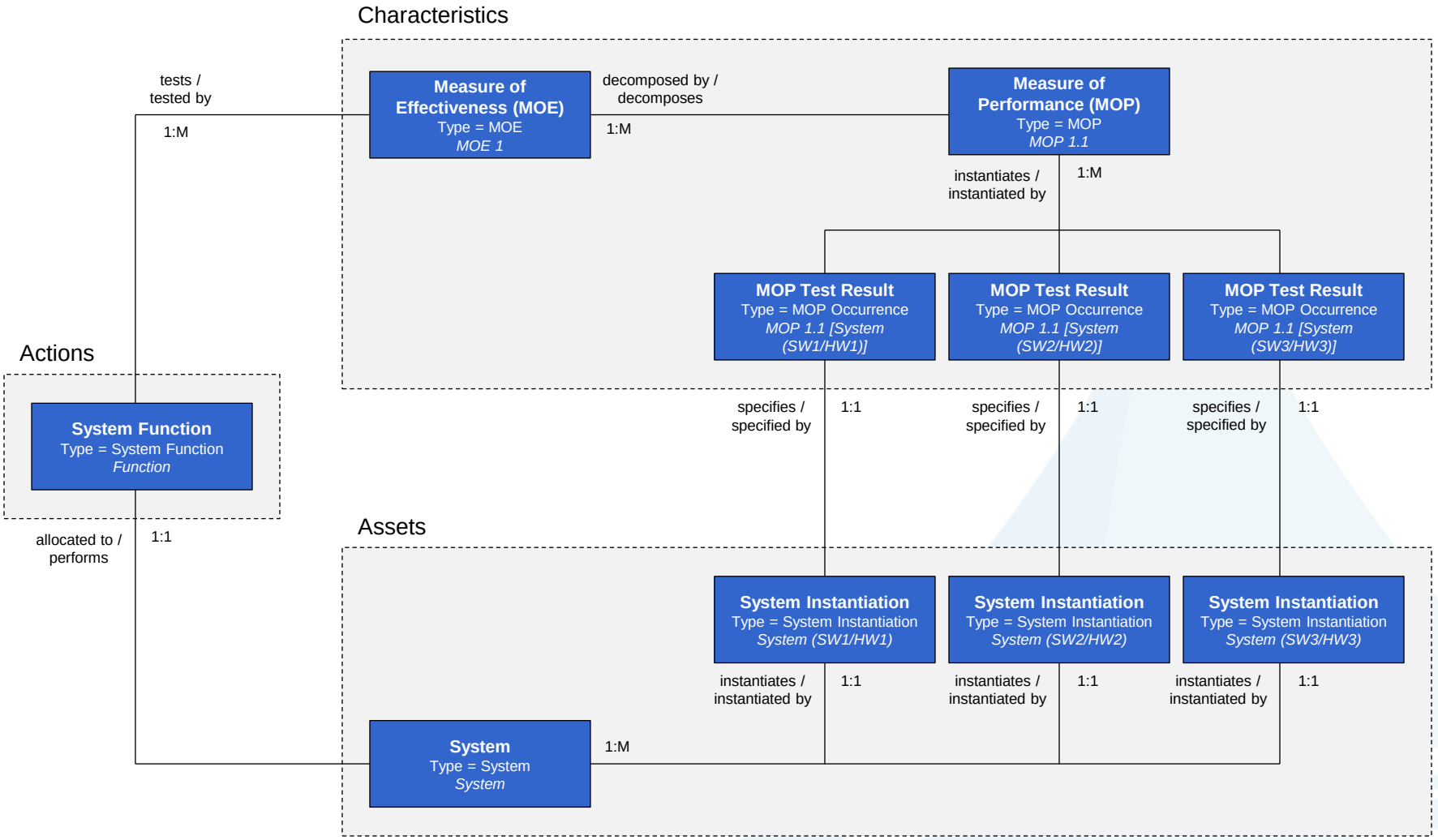
SPEC INNOVATIONS

USE OF LML IN TEST AND EVALUATION

Coming Up the Vee



Measure of Performance (MOP) View



USE OF LML IN OPERATIONS AND SUPPORT

LML Support Operations and Support Analyses

- Process Modeling
- Simulation of Operations
- Training Processes and Procedures
- Operations Manuals
- Logistics Analysis

SUMMARY

LML Bottom-Line

- LML provides a simple, complete language for all stakeholders, not just software developers
  - SysML/UML focus on software developers only
- Use of Actions instead of constructs to capture command and control functions explicitly
- Translation from LML to other languages now feasible
- Support for entire lifecycle

SPEC INNOVATIONS