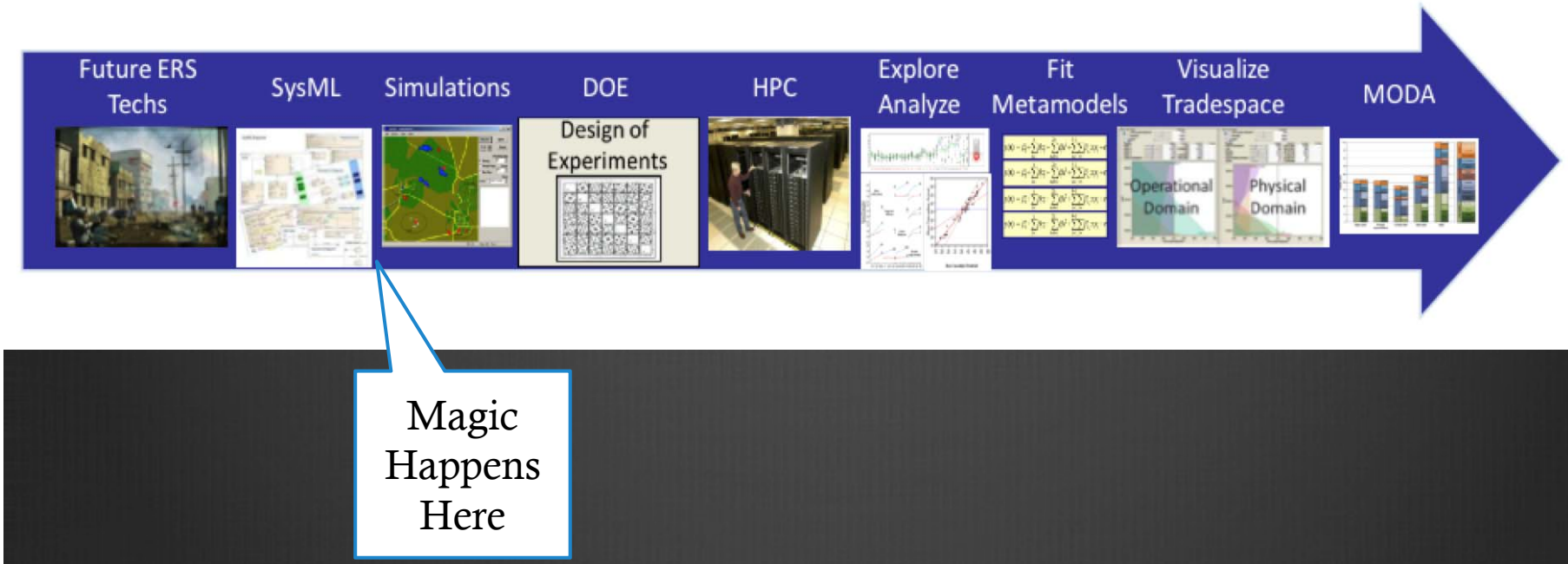
Cloud-Based Deployment and Distributed Execution of Models

Rob Kewley, USMA

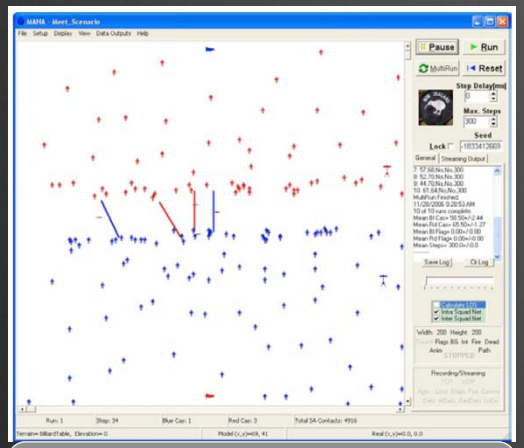
Motivation



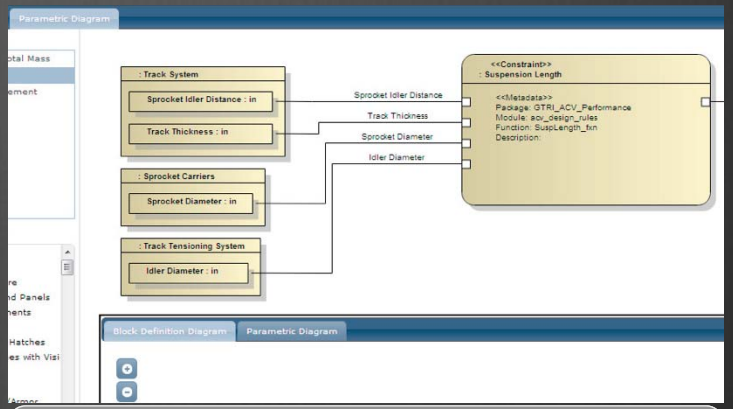
- Chain of tools for using M&S to develop system of systems architecture
- How to we connect the system design properties to the simulation inputs?

Connect Systems Architecture to Executable Model

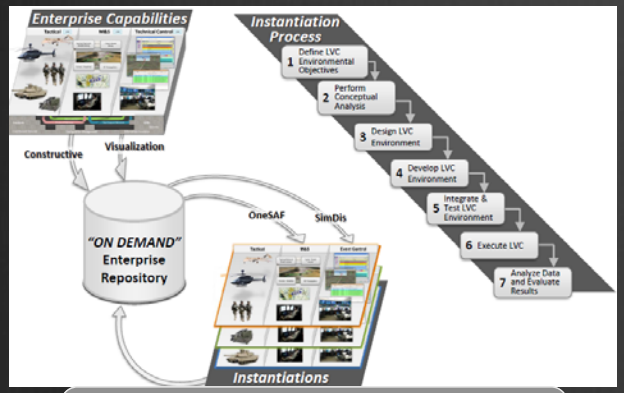
Current Approaches



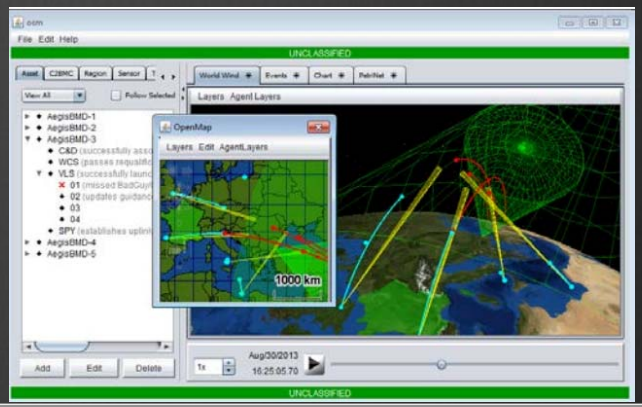
Agent Based Models
Engineering Resilient Systems



Link many independent models
Framework for Assessing Cost and Technology



HLA/DIS
Always On



Frameworks
Orchestrated Simulation through Modeling

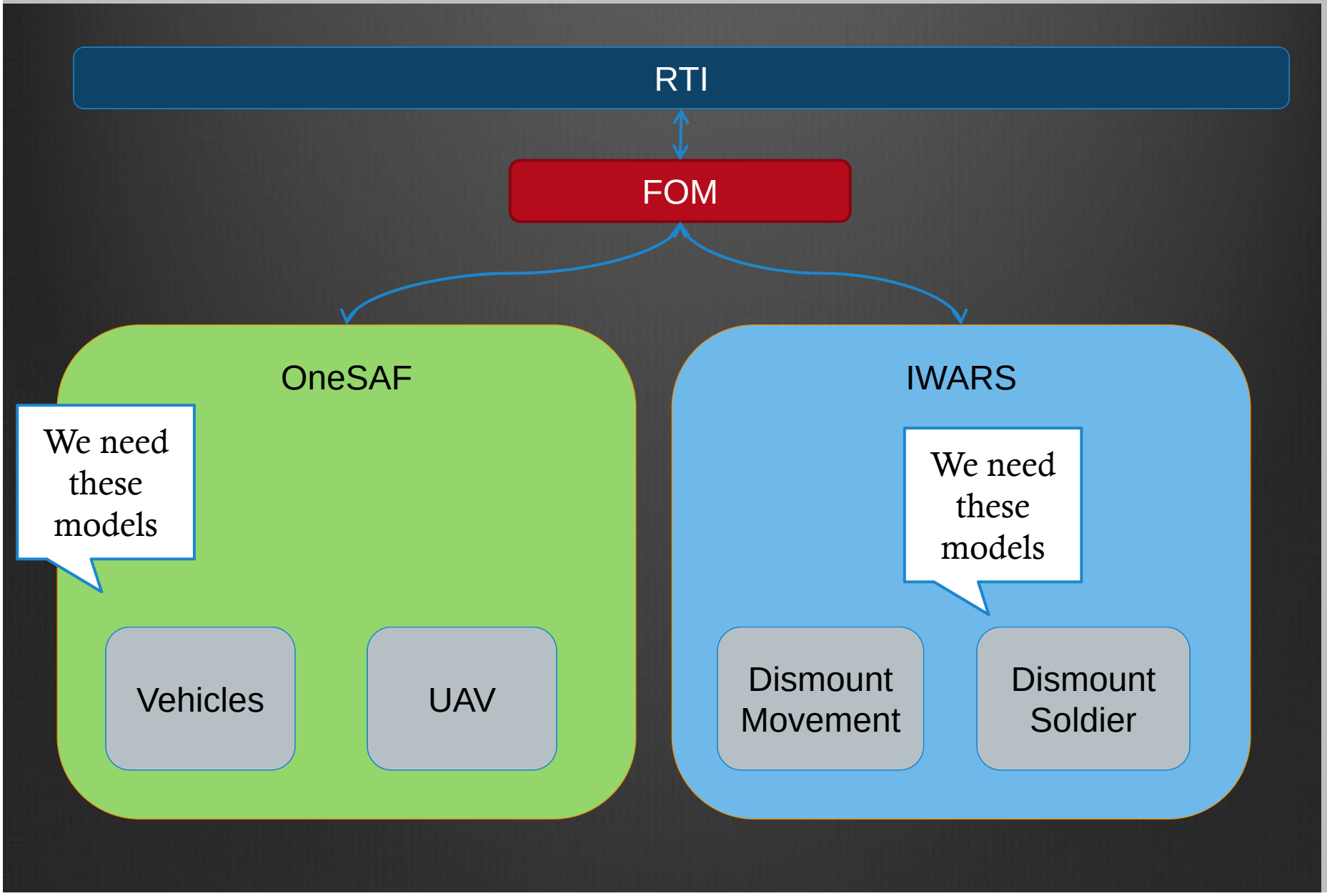
West Point Department of Systems Engineering

We have tried HLA/DIS before…

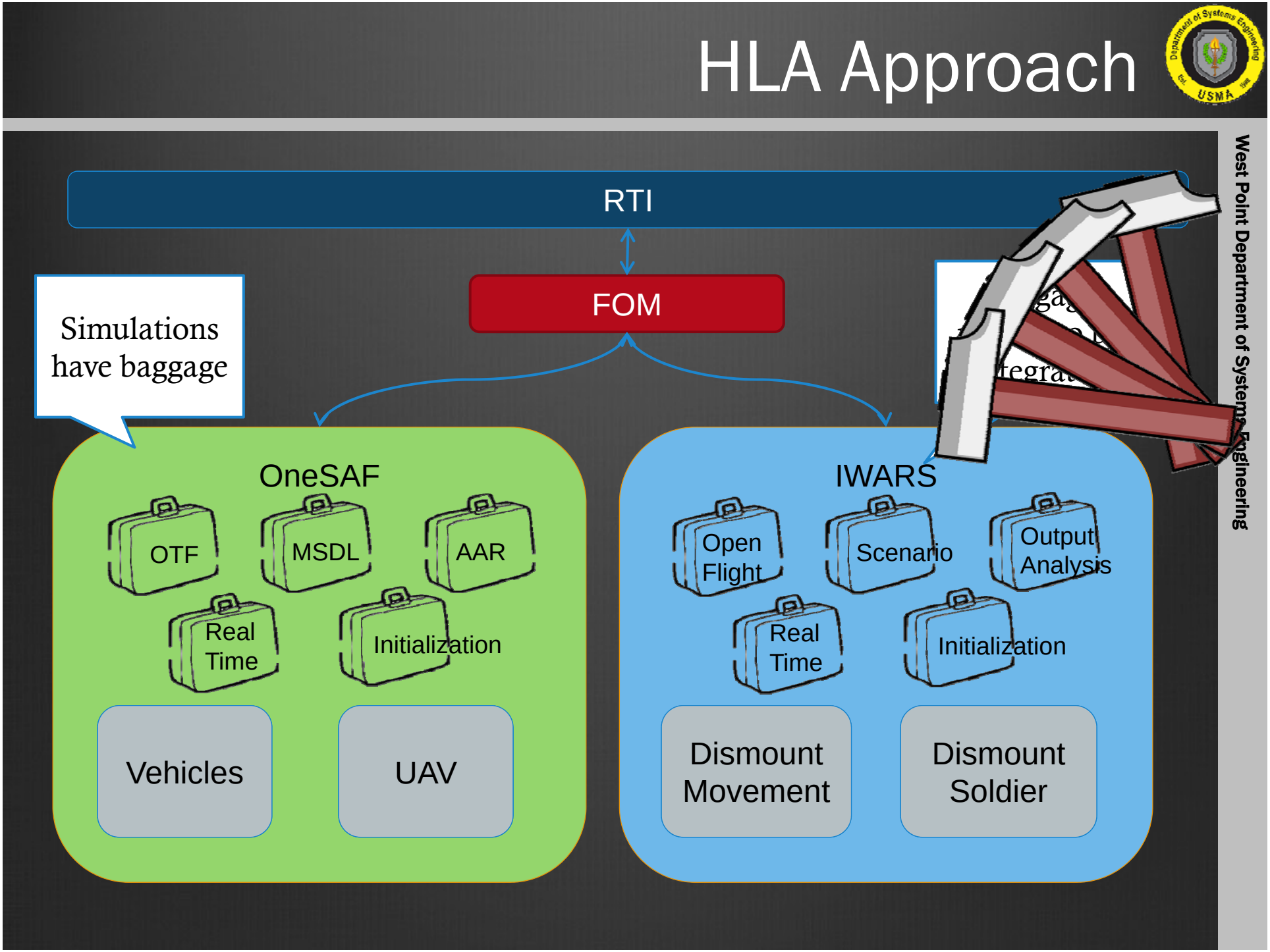
…HLA/DIS integration does not scale….

- 5 different information exchange protocols
- 4 different terrain databases
- Runs only in real time
- Complex scenario initialization must be manually coordinated across federates
- Scenario execution is manual and error-prone
- Data collection complex and causality can be impossible
- Changing the scenario is a months-long proposition

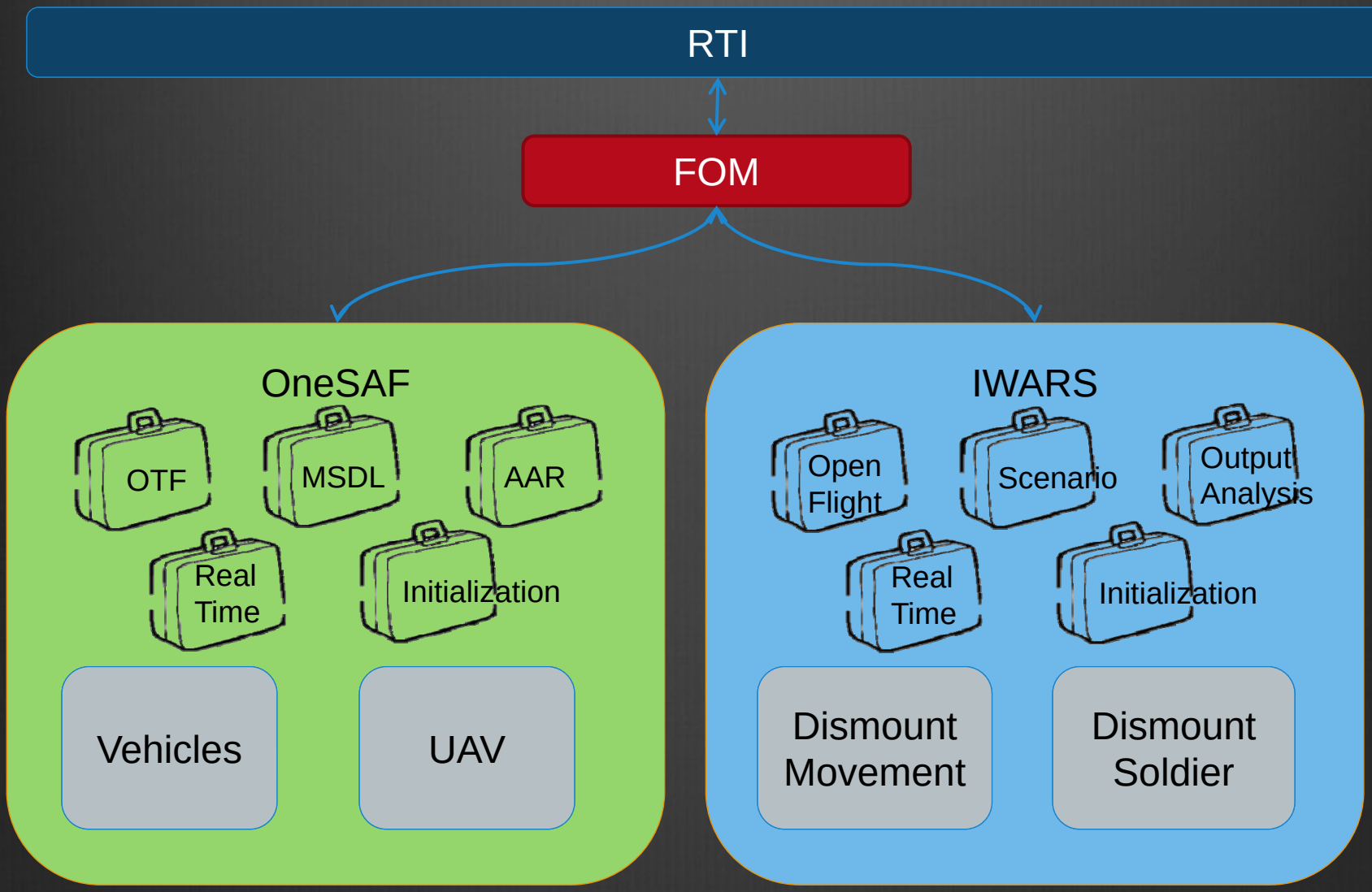
HLA Approach



HLA Approach



HLA Approach



Distributed Modeling Framework

Bring your models a "pure" state transition functions. Leave the baggage at home.

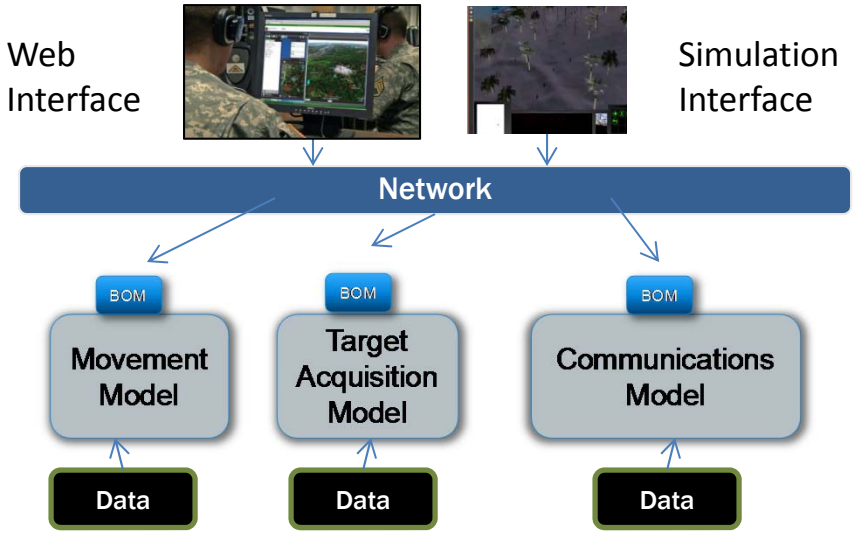
Things to Discuss

- Federate models, not simulations
  - Summer vacation at AMSAA
- Loosely managed distributed architecture
  - Models are services via an interface (BOM)
  - Communicate via messaging (Actor Model)
  - Compose models in an interface (…like ProModel)
  - Systems model (SysML) drives model parameters
  - Distributed and parallel execution engine (DEVS-Akka)
  - Support with design and analysis of experiments
- Take advantage of latest advances
  - Enterprise technologies
  - Discrete Event Systems Specification (DEVS) models
  - Actor model of computation
- Proof of principle implementation
- Target implementation for small UAS

Web Based Modeling and Simulation for Analysis

Develop capability to deploy Army models and simulations to distributed users via web a cloud technologies.

Technical Approach



Models as services – connected to data

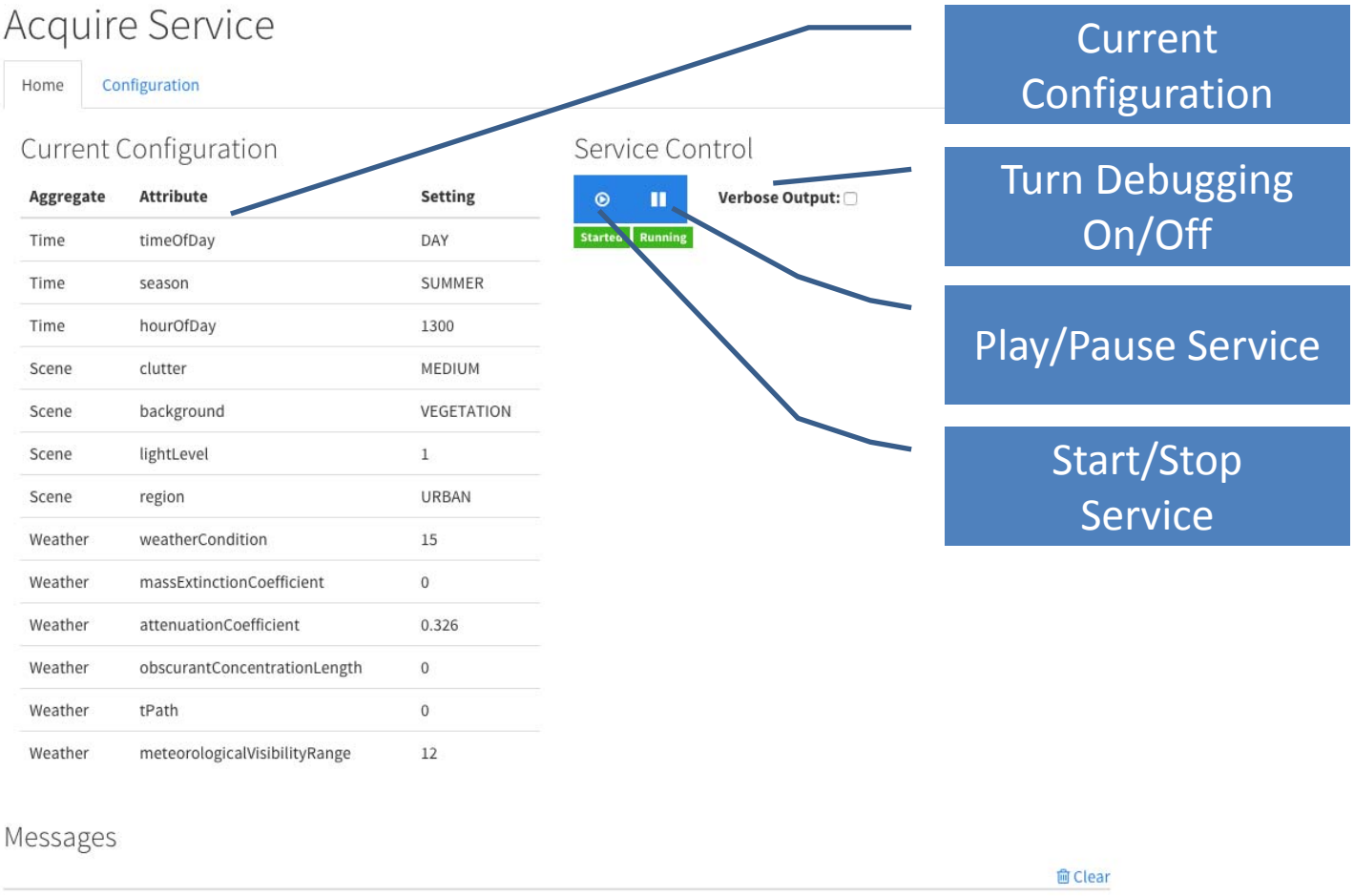
Objectives

Current Business Model	Proposed Business Model
Deploy as software	Deploy as web services
Accessed locally by a few users	Run distributed by many users
Data deployed separately	Data integrated and updated
Single programming language	Cross-platform integration of models in different languages

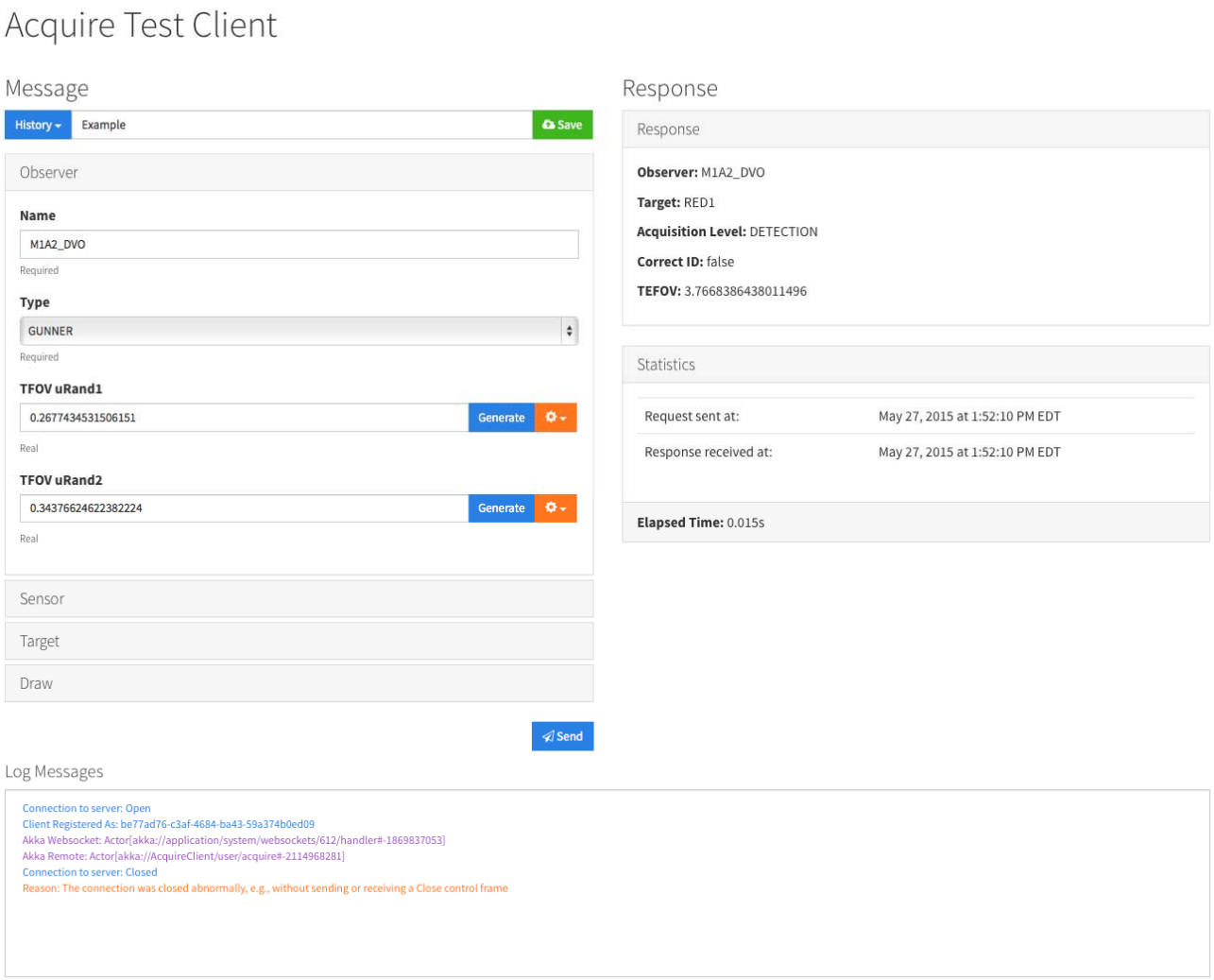
Deliverables

- AMSAA models deployed on classified network as services with web and simulation interfaces
  - ACQUIRE-TTPM-TAS
  - Direct Fire Accuracy
  - Dismounted Vulnerability
  - Dismounted Mobility
  - Hand Grenade Accuracy
- Proof of principle web-based simulation integrating these models

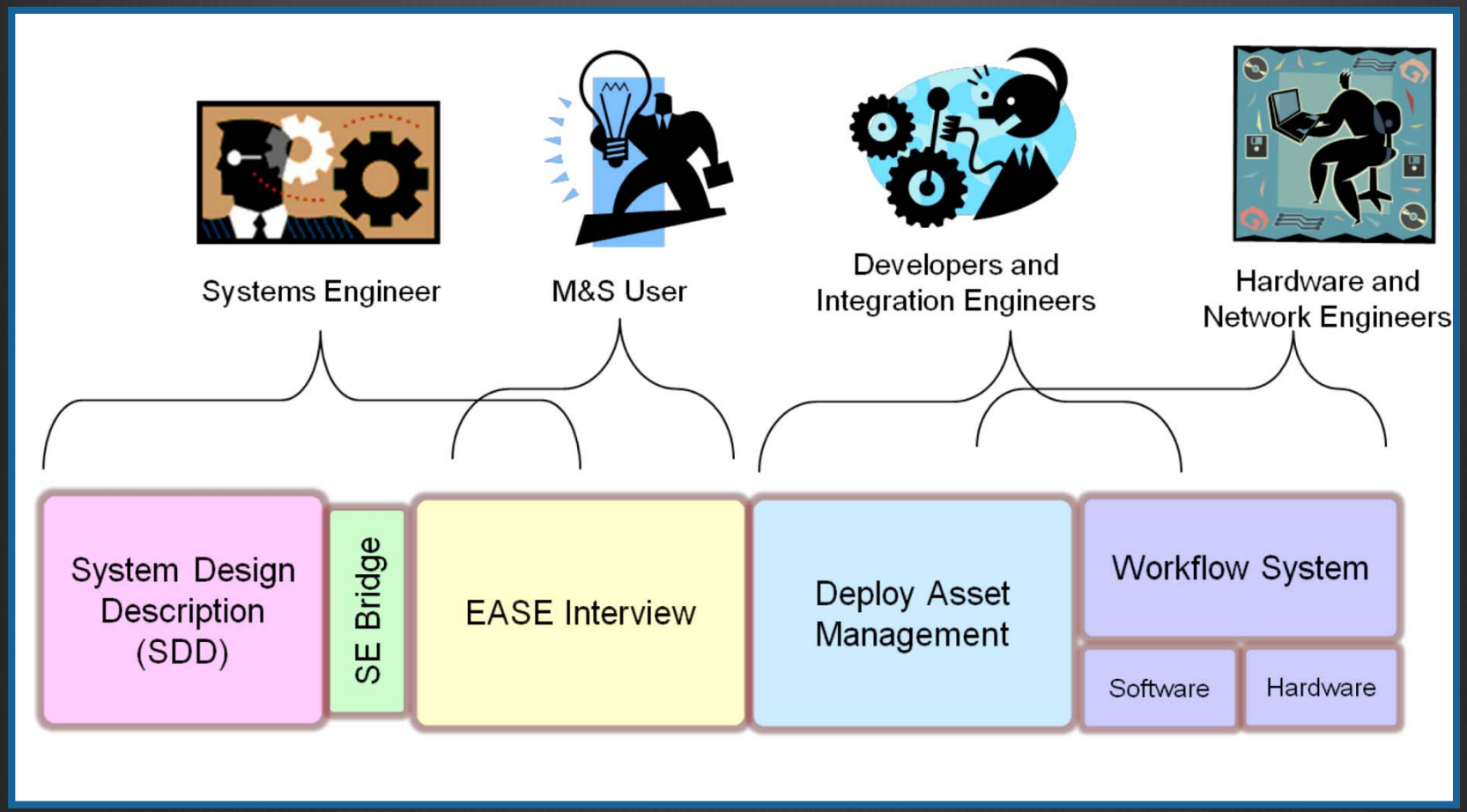
Acquire Service



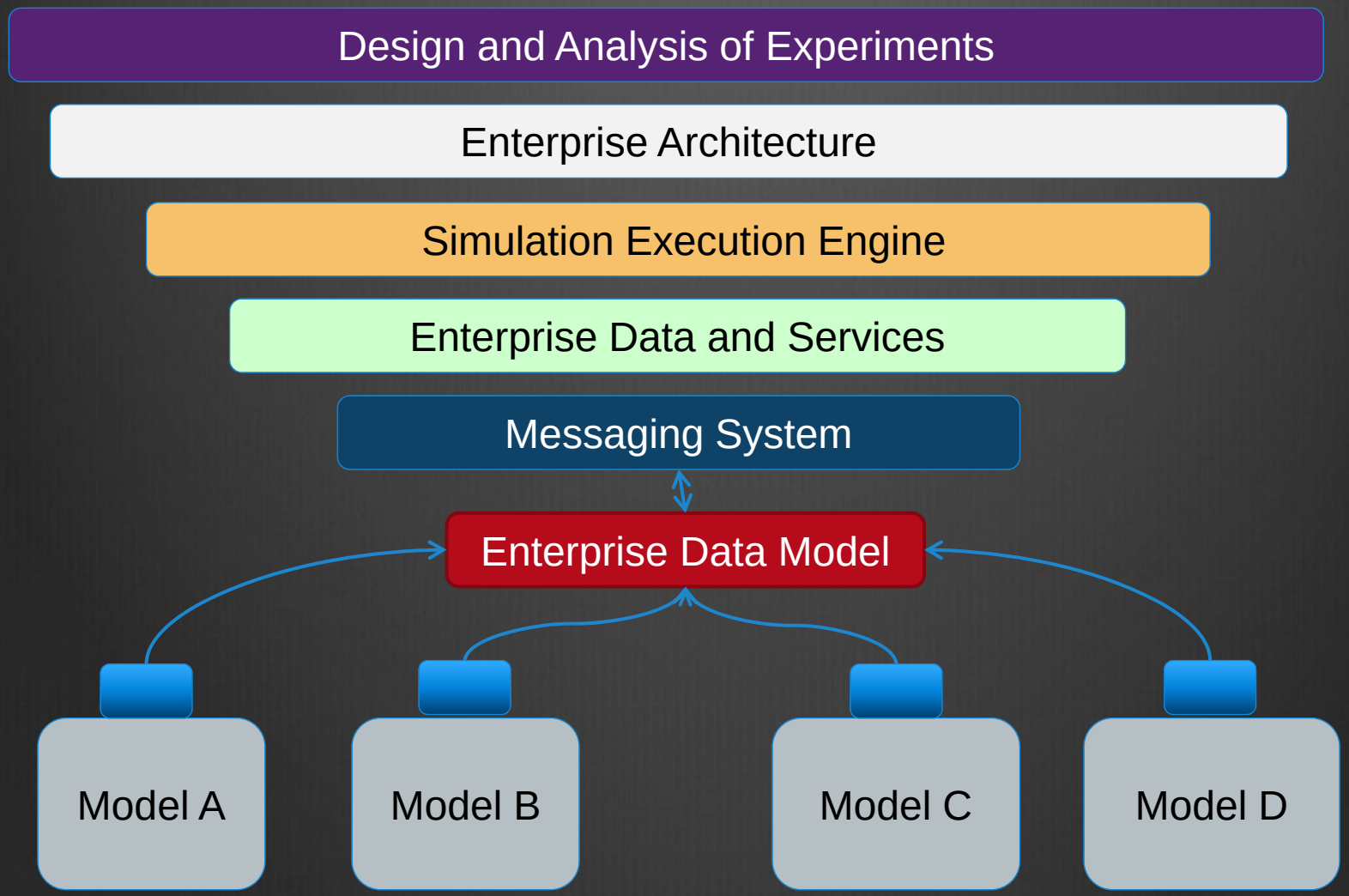
Acquire Test Client



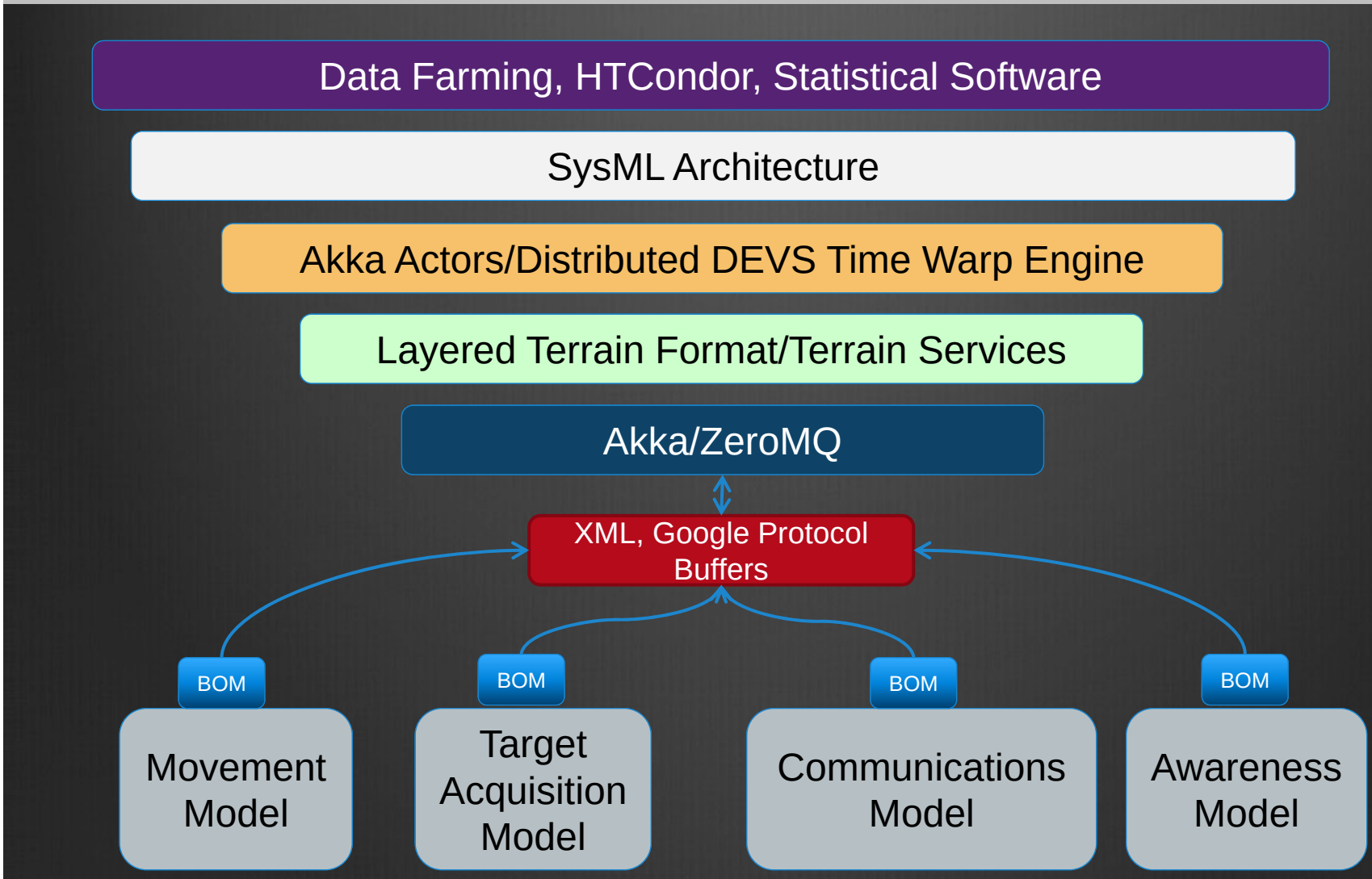
EASE Program
ARL Simulation and Training Technology Center



Enterprise Model Integration



Technology Stack



Executable Architecture for Systems Engineering Distributed Modeling Framework (EASE-DMF)

EASE-DMF provides the tools for the simulation analyst and engineer to build scenarios, combine models, set properties, and collect data in order to analyze system performance.

Sensor Properties

Soldier Properties

C2 System

Simulation Analyst Engineer

NSRDEC

Soldier Models
- Movement
- Weapon
- Target Acquisition

NVL

Sensor Models
- Scanning
- Target Acquisition

Executable Architecture

Systems-model mapping
Federation execution
Time management
Data management
State management
Data collection
Design of Experiments

Results

Scenario

Simulation State Events

Squad on patrol

Enemy activity

Video
Spot reports
Position reports

TOC

QRF

CERDEC

Network Models
- Propagation
- Bandwidth

Network Properties

TARDEC

Vehicle Models
- Movement
- Survivability

Vehicle Properties

CERDEC

Situation Awareness Models
- Sensor Fusion
- COP Management

C2 Properties

Operational Scenario

Legend
- Simulation Execution
- Simulation Initialization
- System-Model Mapping

System models are functional, discrete, and modular. They can be added as needed to the simulation and run in parallel. In contrast to current federations, the analyst combines many simple and functional models of individual systems, not entire simulations.

West Point Department of Systems Engineering

Useful Theories - Simulation

- Discrete Event Specification (DEVS)
  - DEVS models are modular
  - Composable hierarchies in coupled models
  - Strong track record
- Base Object Model (BOM)
  - Adds semantics to the models
  - Complete specification of data inputs and outputs
  - Situates models in a chain or interactions
- Parallel algorithms
  - Optimistic time advance – Time Warp
  - Supports distributed and cloud-based implementations

Useful Theories – Computer Science

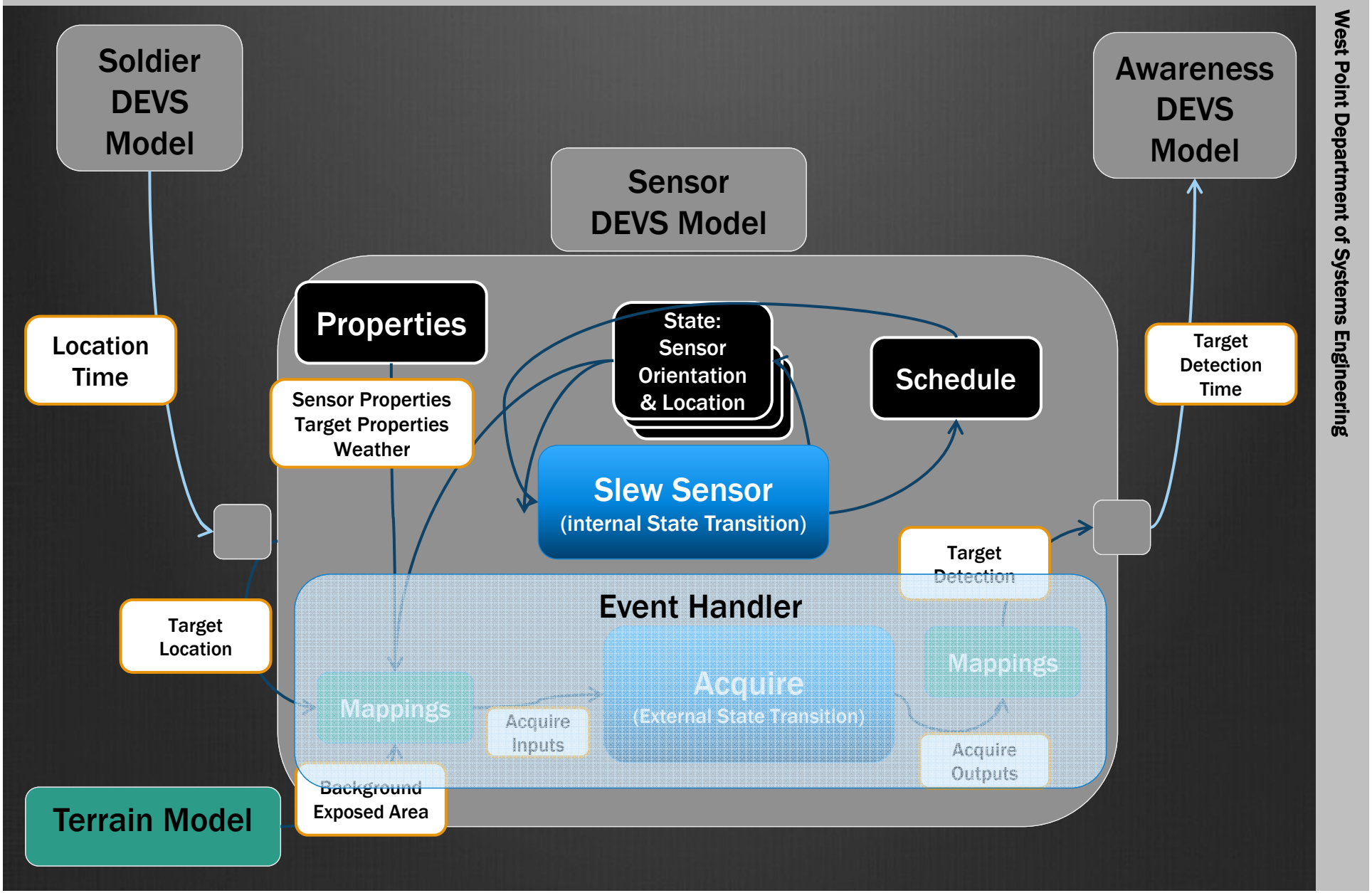
- Functional programming
  - Functions are composable
  - Functions have no side effects
  - Predictable behavior
- Actor model of computation
  - Encapsulation of state
  - Responds to messages by…
    - Sending messages to other actors
    - Changing state in a way that influences future messages
    - Creating new actors
  - Reactive programming
    - Event driven
    - Responsive
    - Asynchronous
    - Loosely coupled messaging
    - Fault tolerant

Akka

- Implementation of reactive actor model
- Scala and Java versions
- Each DEVS model is an Akka SimActor
- Each actor runs on its own thread
- Exchanges data only through messaging
- Support for serialization and distributed actors
- Ability to manage threads
- Open source

http://akka.io

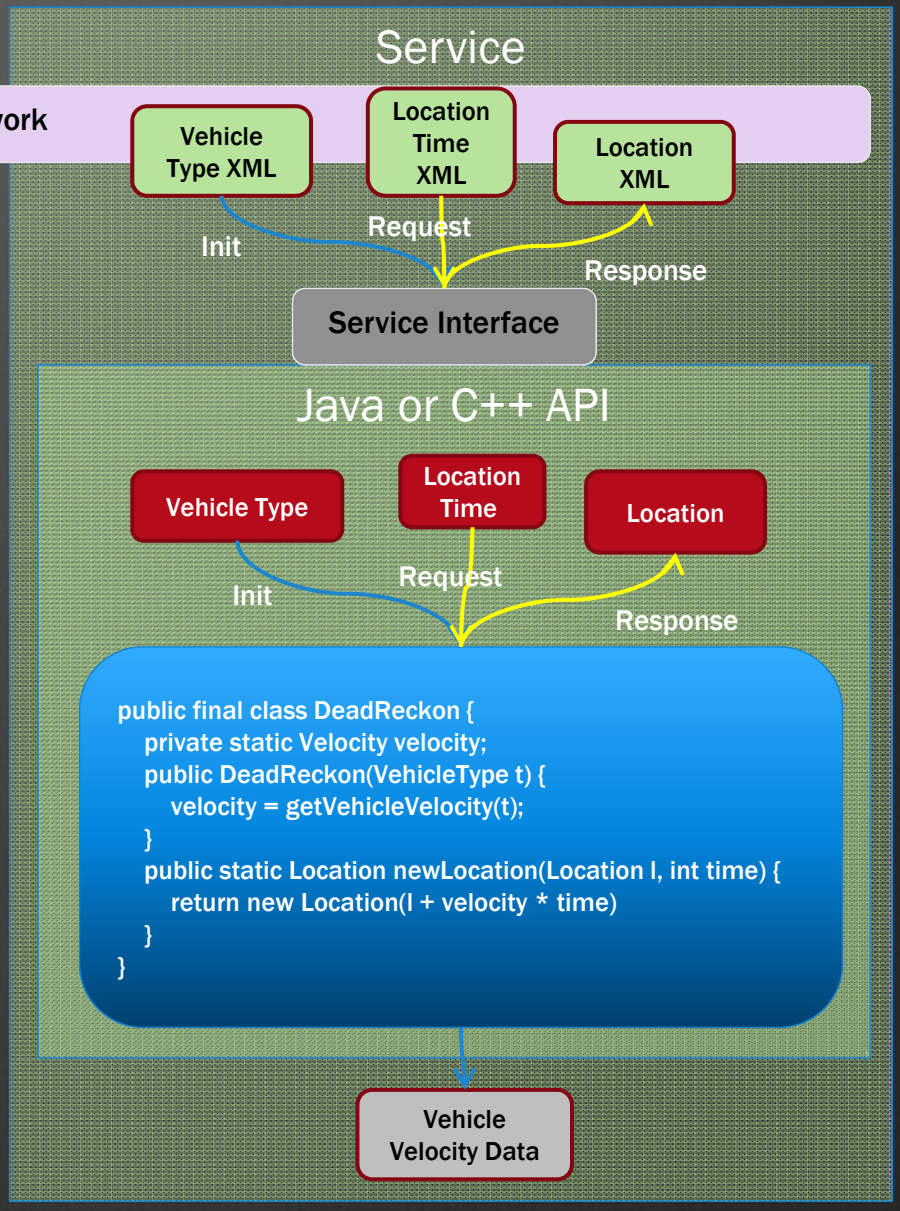
ACQURIE Sensor Model



West Point Department of Systems Engineering

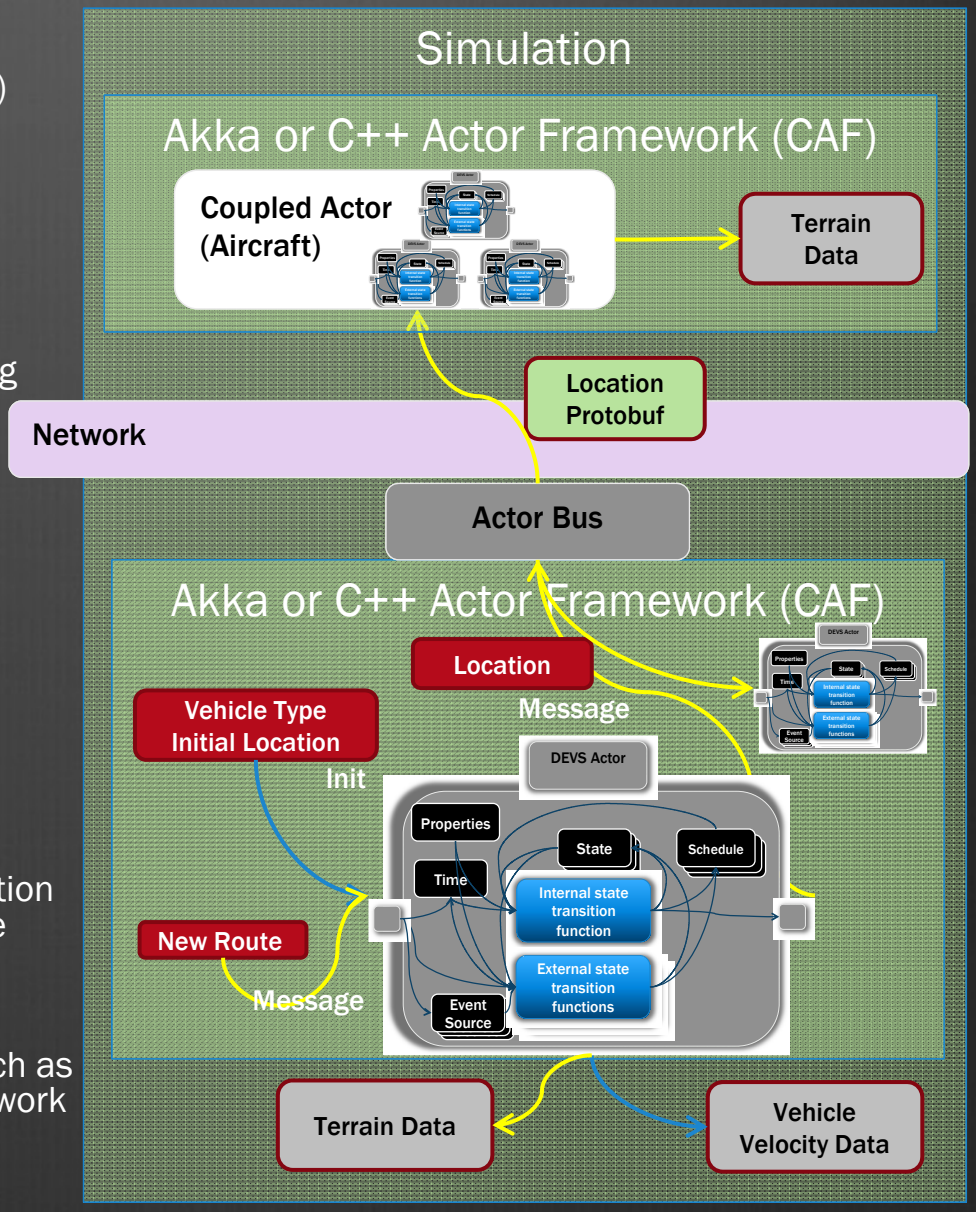
Exposing Function Interfaces

- As a class in API
  - Use static class
  - Pass in initialization data at instantiation
  - Use public static methods
- As a state transition function service
  - Define message classes for input/output
  - Map message classes to messaging protocol such as:
    - JSON
    - Protocol Buffers
    - XML
  - Expose through a service interface
  - Document everything thoroughly

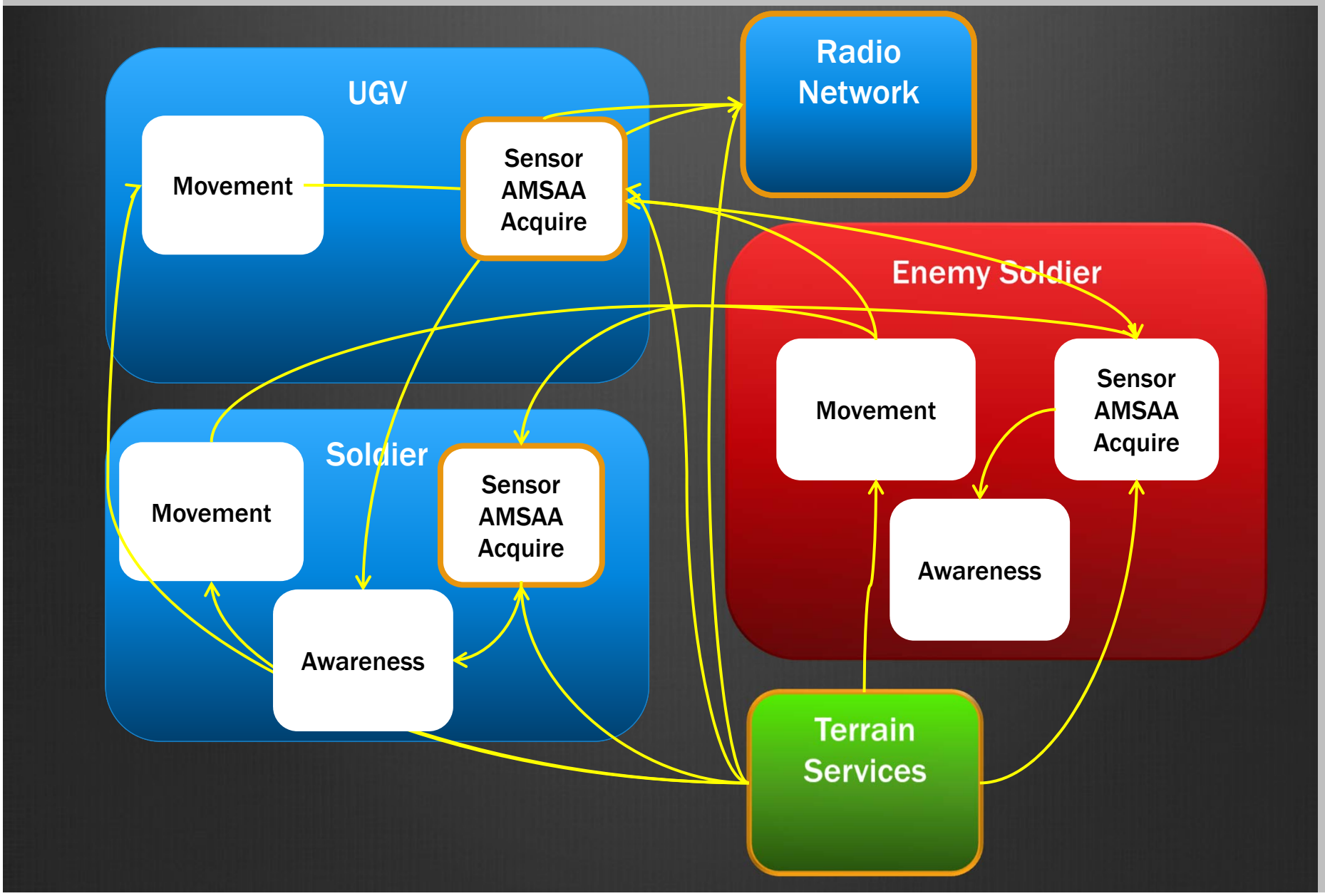


Composing DEVSActors

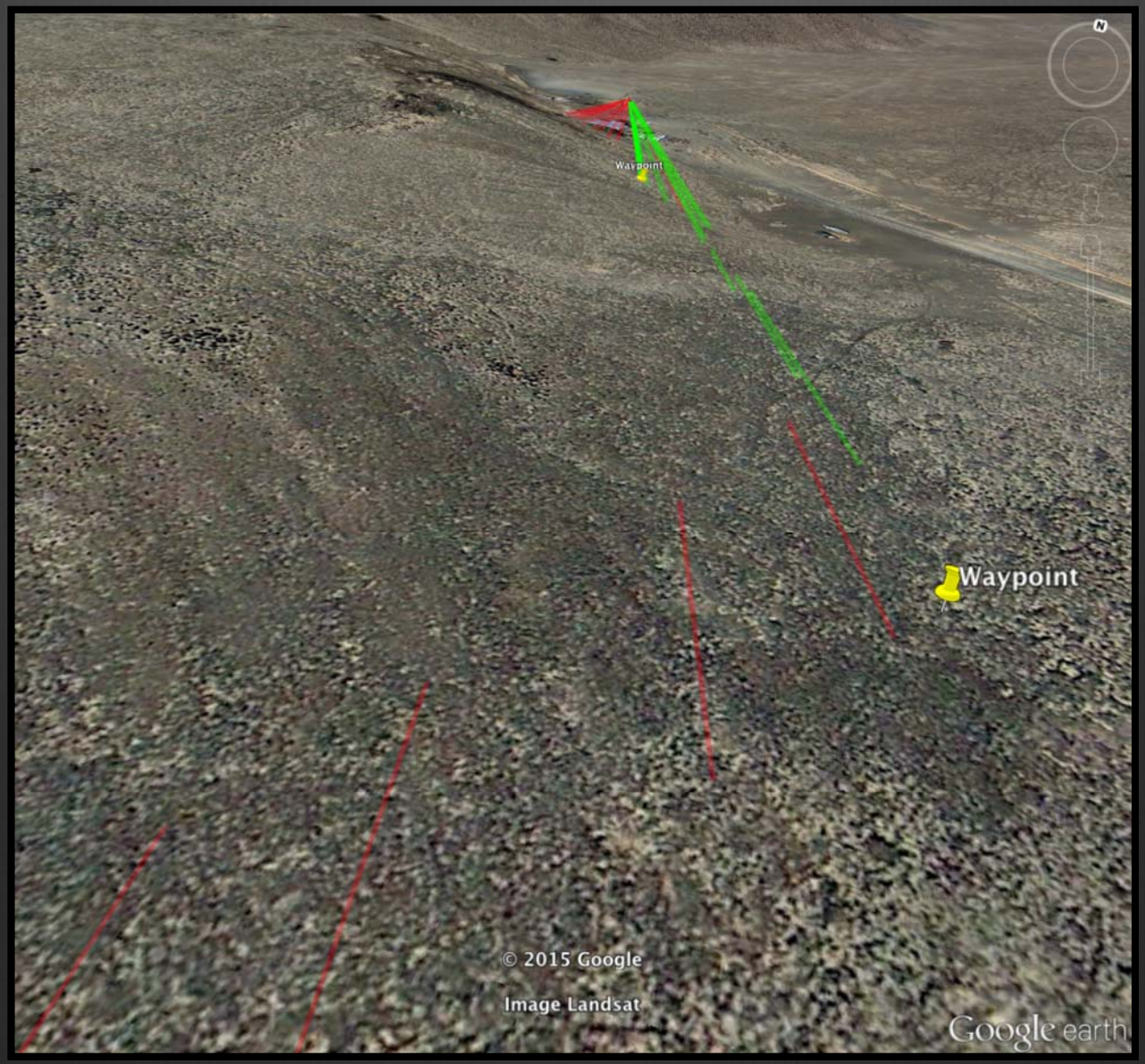
- As DEVSActor
  - Akka or C++ Actor Framework (CAF)
  - Pass in initialization data at instantiation
  - Define message interface
- As a distributed actor
  - Map message classes to messaging protocol such as:
    - JSON
    - Protocol Buffers
    - XML
  - Akka or CAF remote actors
  - Document everything thoroughly
- As simulation
  - Compile tightly linked models in a coupled actor
  - Compile models with heavy interaction traffic in same JVM (Akka) or native library (CAF)
  - Control with execution engine
  - Copy large interactive data sets such as terrain to each process to save network traffic



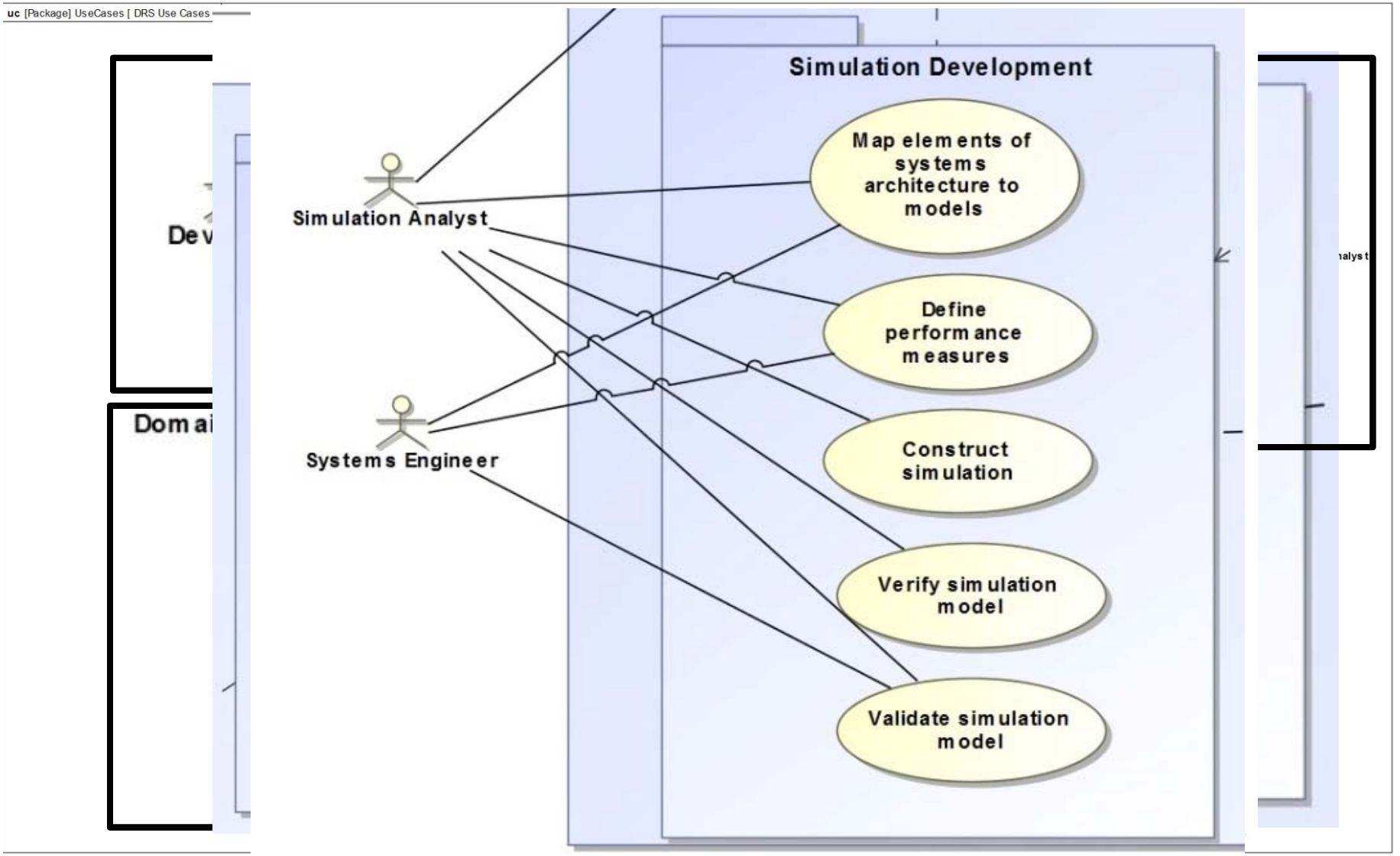
Reference Implementation



Simulation Execution



EASE-DMF Use Cases



Small Unmanned Aircraft Modeling Framework

PM UAS
USMA
AMRDEC

ARL STTC
AMSAA
NSRDEC

Technical Approach

- Develop a series of discrete analysis models to represent small UAS technical performance
  - Target acquisition
  - Communications propagation
  - Situation awareness
  - Power
  - Flight dynamics
  - Terrain
- Wrap in web-based interface
- Integrate in simulation framework using Discrete Event Systems Specification (DEVS) and SysML

Summary

- New SUAS acquisition strategy requires PM UAS to operate as lead systems integrator (LSI)
- LSI role requires assessment of engineering trades
- Develop a web-based integrated modeling and simulation framework to assess engineering trades for small unmanned aircraft systems

Deliverables

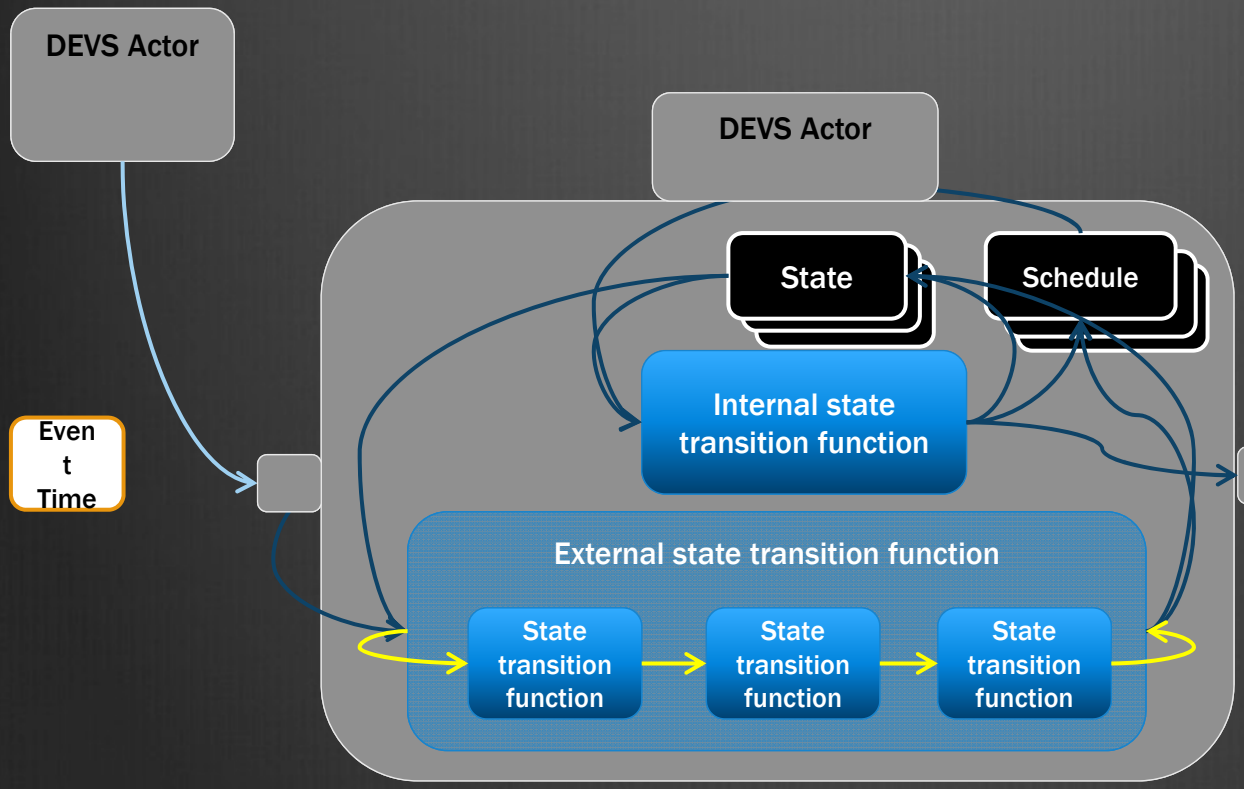
- SysML architecture of proposed micro UAS
- Scenario development and design of experiments
- Simulation analysis of micro key UAS performance parameters tradespace
  - Size, weight, and power
  - Fixed vs. Rotary wing
  - Noise
  - Sensor performance
  - Radio performance
- Squad or platoon level mission performance metrics

West Point Department of Systems Engineering

Functions of a DEVSActor

- Initialize with static properties
- Local and global virtual time
- An internal schedule of state transitions
- A set of internal and external state transition functions
- It may update its state, schedule and internal event, or generate a message to other DEVSActors
- Maintain record of internal state and drop as global time advances
- List of events that may be replayed in the event of roll back
- An ability to generate random variates needed by state transitions
- A publish and subscribe mechanism that asks for a notification message upon sepecific state transitions of other DEVSActors

M&S Composition
State Transition Functions



- Functions are mathematically composable
- State transitions that take place at same instant can be chained into one state transition
- Calls out to stateless utility functions such as coordinate transformation

- Rules for functions:
- Must be stateless
- May use initialization data under closure
- These are static objects with only static data and methods
- After initialization, calling with same data always gives same answer

West Point Department of Systems Engineering