

Domain Modeling

Roadmap to Convergence

Nathaniel Horner ↔ Steve Topper

24 April 2008

"You got to be careful if you don't know
where you're going, because you might not
get there."

- Yogi Berra

APL
The Johns Hopkins University
APPLIED PHYSICS LABORATORY

Introduction

Problem Domain:

The real-world things and concepts related to the problem that the system is being designed to solve.

Domain Modeling:

The task of discovering “objects” (classes) representing things and concepts, and the relationships between them.

Source: Rosenberg and Scott, (2001)
Architecture & Design. Driving Design: The Problem Domain

Problem Statement:

Develop efficient techniques to support complex system analysis

Given:

Complex systems, lots of components, subsystems, sophisticated behaviors, networks, information processing, collaboration

Organizations involved in design & development of these systems

Analysis, requirements, architecture, systems engineering, software engineering, testing, operations.

Approach:

Understand the problem Domain and progress from there...

Overview

- **Introduction**
- **Conceptual Modeling Process Overview and Walk-Through**
- **Why is domain modeling the foundation of this process?**
- **How does it relate to other project tasks?**
 - **Analysis**
 - **M&S, Software Engineering**
 - **Systems Engineering / Architecting**
 - **Business Processes**
- **“Goods and others” of Domain Modeling**

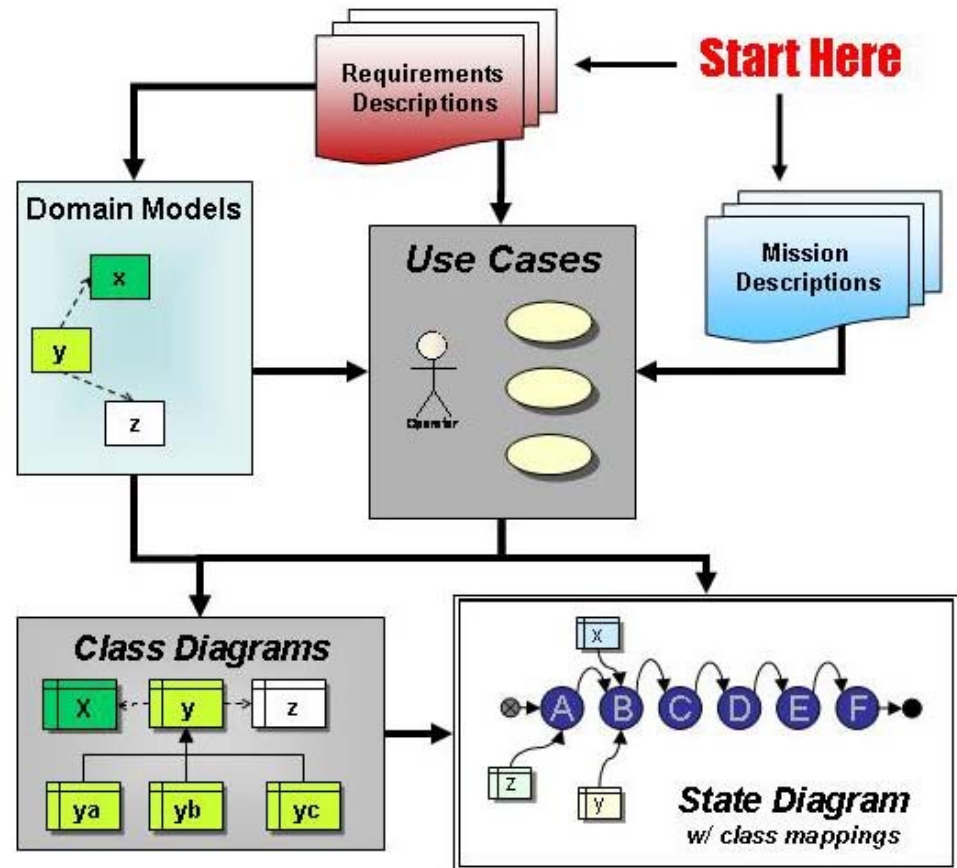
Why are we here?

- **Initial activities attempted**
 - **To develop generic DoDAF artifacts,**
 - **To link these artifacts more closely to developed models,**
 - **To provide a basis for new model and simulation (M&S) development across a wide community of stakeholders.**
- **Issues**
 - **Legacy tool challenges for complex systems-of-systems analysis (configuration/preparation time, fidelity and interoperability issues).**
 - **Lack of standardized foundation.**
 - **Traditional architectures often difficult to assess using M&S (lacked underlying referential structure).**
 - **Activities difficult to accurately plan and estimate.**

How can we fix it?

Conceptual Modeling Process

- Based on standard software and systems engineering processes.*
- Translates informal, generalized information from disparate sources into formal system models.
- Maintains focus on understanding and standardizing the problem space before moving on to the solution.
- Allows iteration and feedback until it's "right."
- Produces documentation allowing traceability throughout the process.



- * Though significantly changed, this conceptual modeling process is informed by ICONIX, a software engineering process falling between RUP and XP with respect to rigor and flexibility. ICONIX is documented in *Use Case Driven Object Modeling with UML* by Rosenberg and Stephens (Apress, 2007), on which some material in this briefing is based.

Input to the Domain Model

What it is: Known information about the system and its environment.

How to do it:

- Informal requirements descriptions and mission descriptions.
 - ***NOT*** detailed, formal system requirements.
 - Generalized statements about system and what it does.
- CONOPS.
- Existing documentation.
- Stakeholder brainstorming sessions.

What it is for: Nouns extracted from these documents form a list of candidate domain entities.

Domain Model

What it is: A “10,000-foot view,” a live “project glossary,” a simplified class diagram.

How to do it:

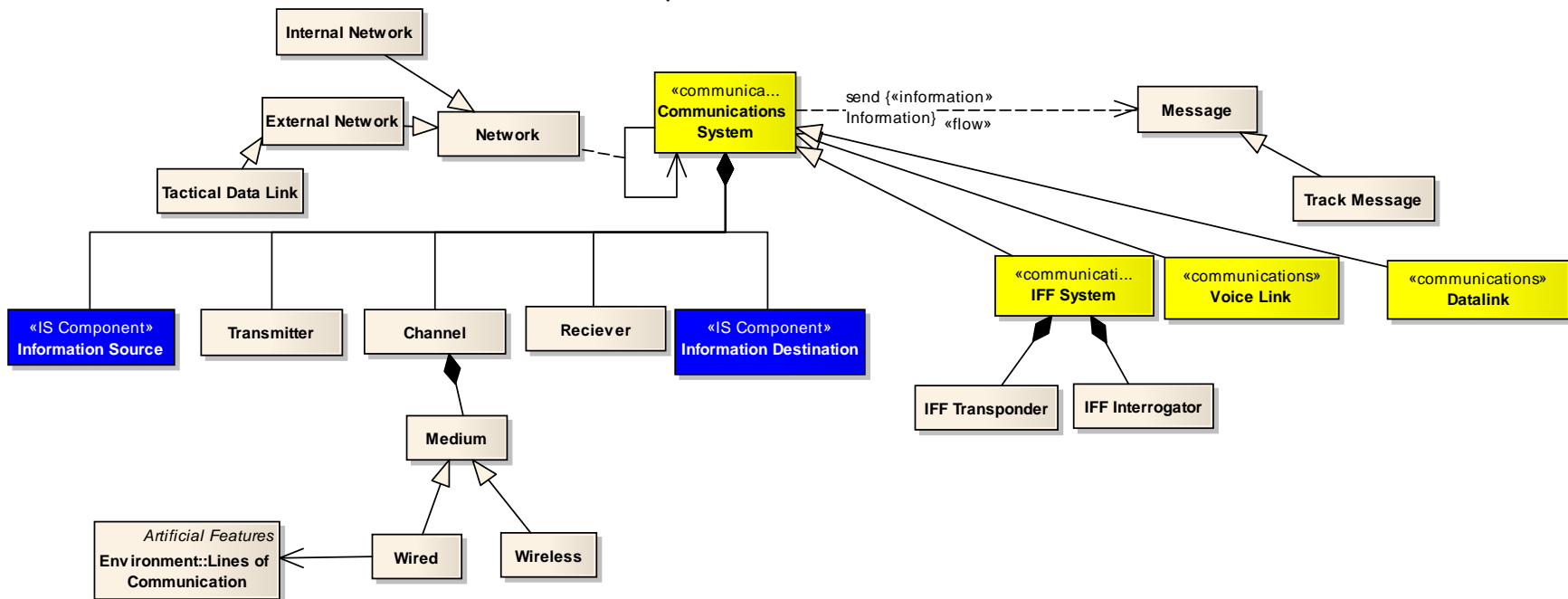
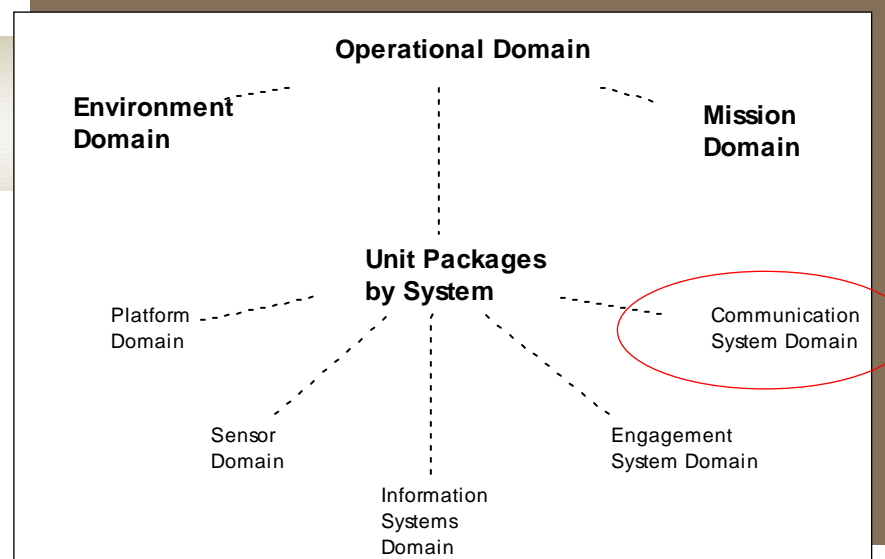
- Create list of candidate domain entities by extracting nouns from input documents.
- Review list, standardizing and defining terms.
- Deploy entities in a simplified class diagram (no attributes or operations) and draw important relationships (generalization, composition/aggregation).
- *Iterate as needed with all stakeholder groups and revisit throughout the project.*

What it is for:

- Answers the question, “What makes up the system and its environment?”
- Defines the scope of the project, standardizes terms.
- *Provides foundation for static structural model.*

Domain Model (Example)

- High Level Domain covers environment, mission and systems-of-systems representations →→→→
- Expands to increasingly detailed system representations



Use Cases

What it is: Descriptions of interactions between the system and its users.

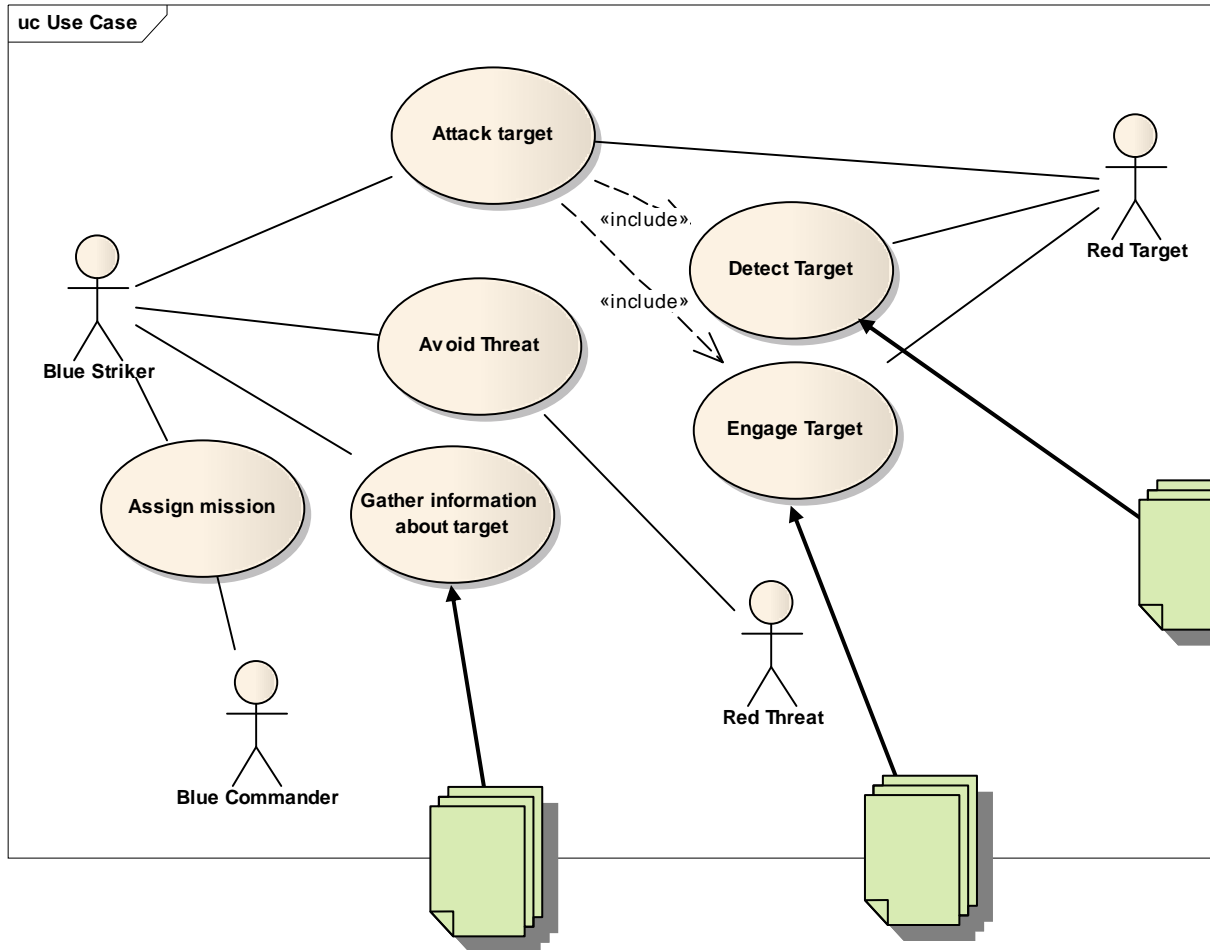
How to do it:

- **Identify**
 - The actors – users of the system, including other systems.
 - The tasks facilitated by the system.
 - The actors' participation in the tasks, including alternate courses of events.
- ***Use vocabulary previously defined in domain model.***
- ***Go back and alter the domain model as errors are uncovered through use case exploration.***

What it is for:

- Answers the question, “What are the user experiences with the system?”
- Helps define scope and provides general basis for more formal modeling.
- ***Provides foundation for the dynamic behavioral model.***

Use Case (Example)



- Use cases are listed in a diagram showing the participating actors.
- Each use case is expanded into a document describing the flow of events involved, including:
 - Actors involved
 - Preconditions
 - Event sequences
 - Exceptions
 - Participants
 - Alternatives
 - Unresolved issues

Class Model

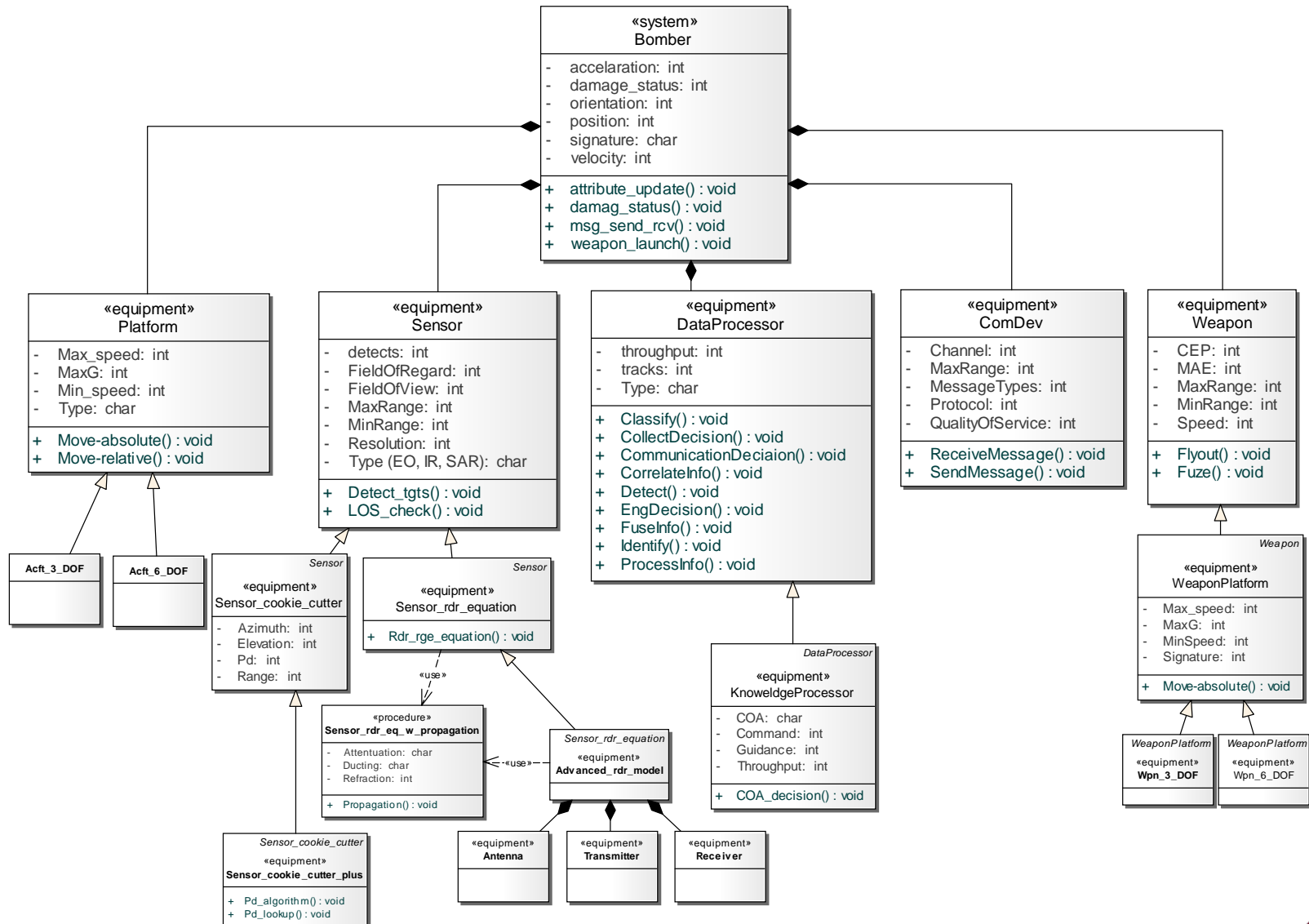
What it is: A more detailed static representation of the domain.

How to do it:

- ***Extend the domain model.***
- **Allocate behaviors to domain model entities based on use case descriptions.**
- **Add attributes and operations to domain model entities.**
- **Add classes to the solution space as necessary.**
- **Work iteratively, going back and forth between static model and behavioral model (e.g., activity, sequence diagrams).**

What it is for: Begins to translate general descriptions into more formal system design.

Class Model (Example)



Activity, State, other Behavioral Diagrams

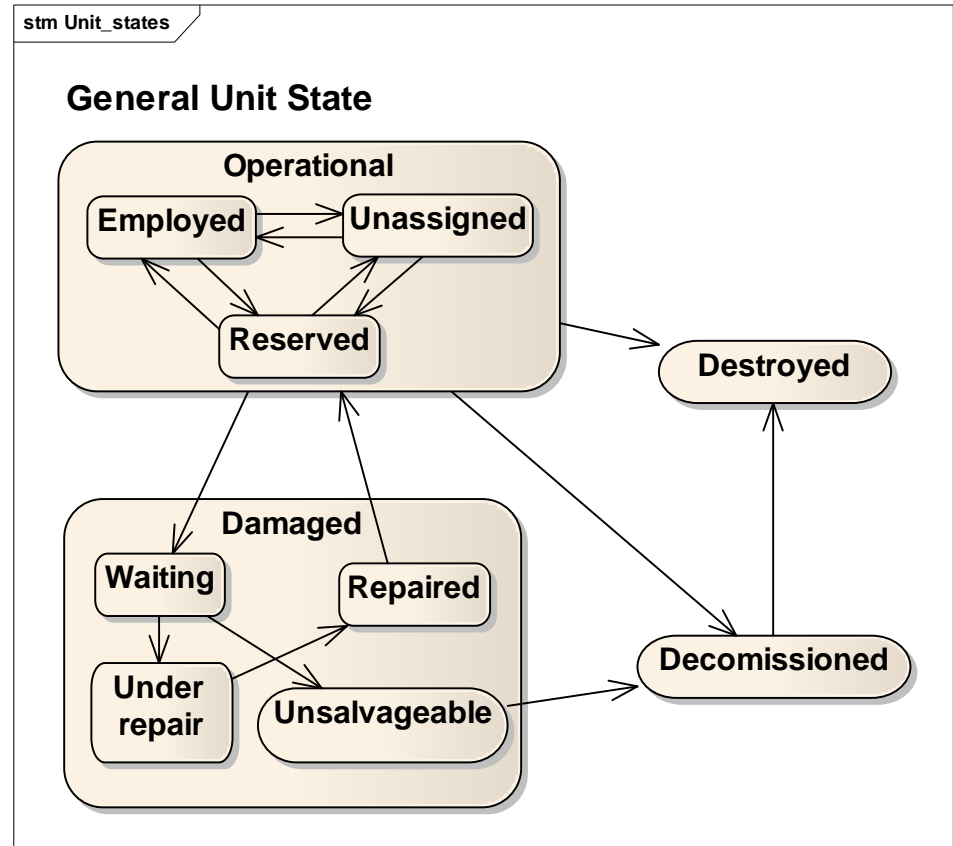
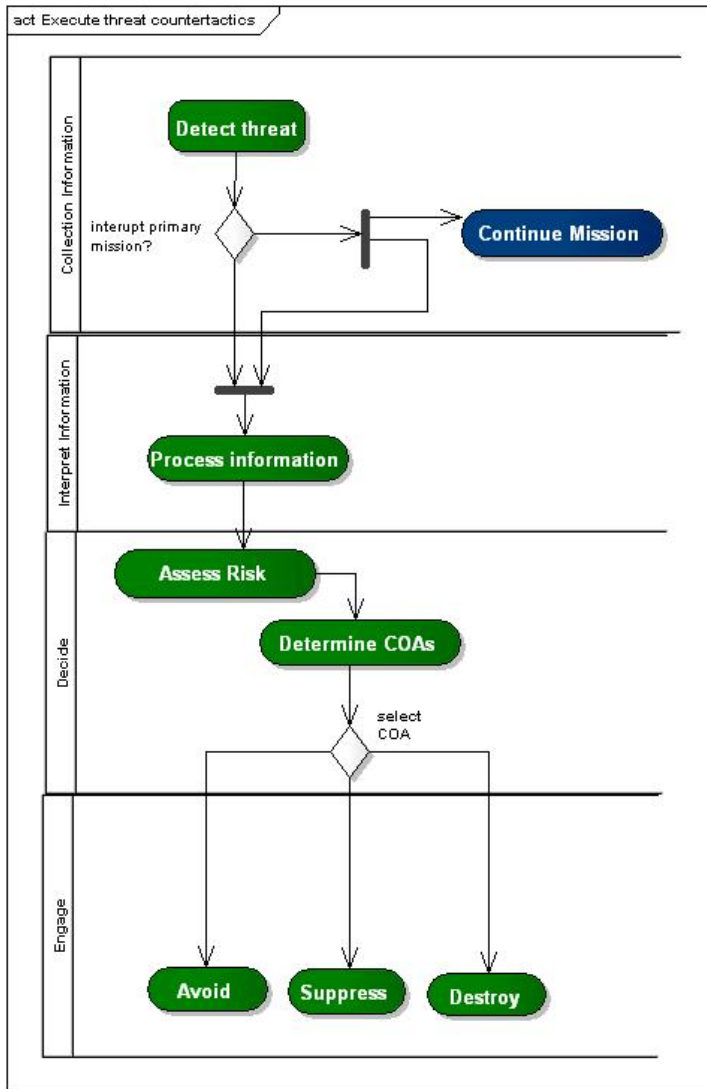
What it is: A more detailed dynamic representation of the system.

How to do it:

- **Create activity diagrams:**
 - **Break up use cases into component transactions or activities.**
 - **Sequence the activities.**
 - **Assign responsibility for each activity to a domain entity via swimlanes.**
- **Create state diagrams:**
 - **Define atomic states for each domain entity.**
 - **Sequence the states.**
 - **Define conditions and constraints governing state transitions.**
- **Use the use cases as a primary input.**
- **Work iteratively, going back and forth between the behavioral model and the static model (domain and class model), ensuring compatibility.**

What it is for: Begins to formalize use cases into more detailed system behaviors and activities.

Behavioral Model (Example)



- 📌 State Diagram
- 📌 Activity Diagram

Why Use Domain Modeling?

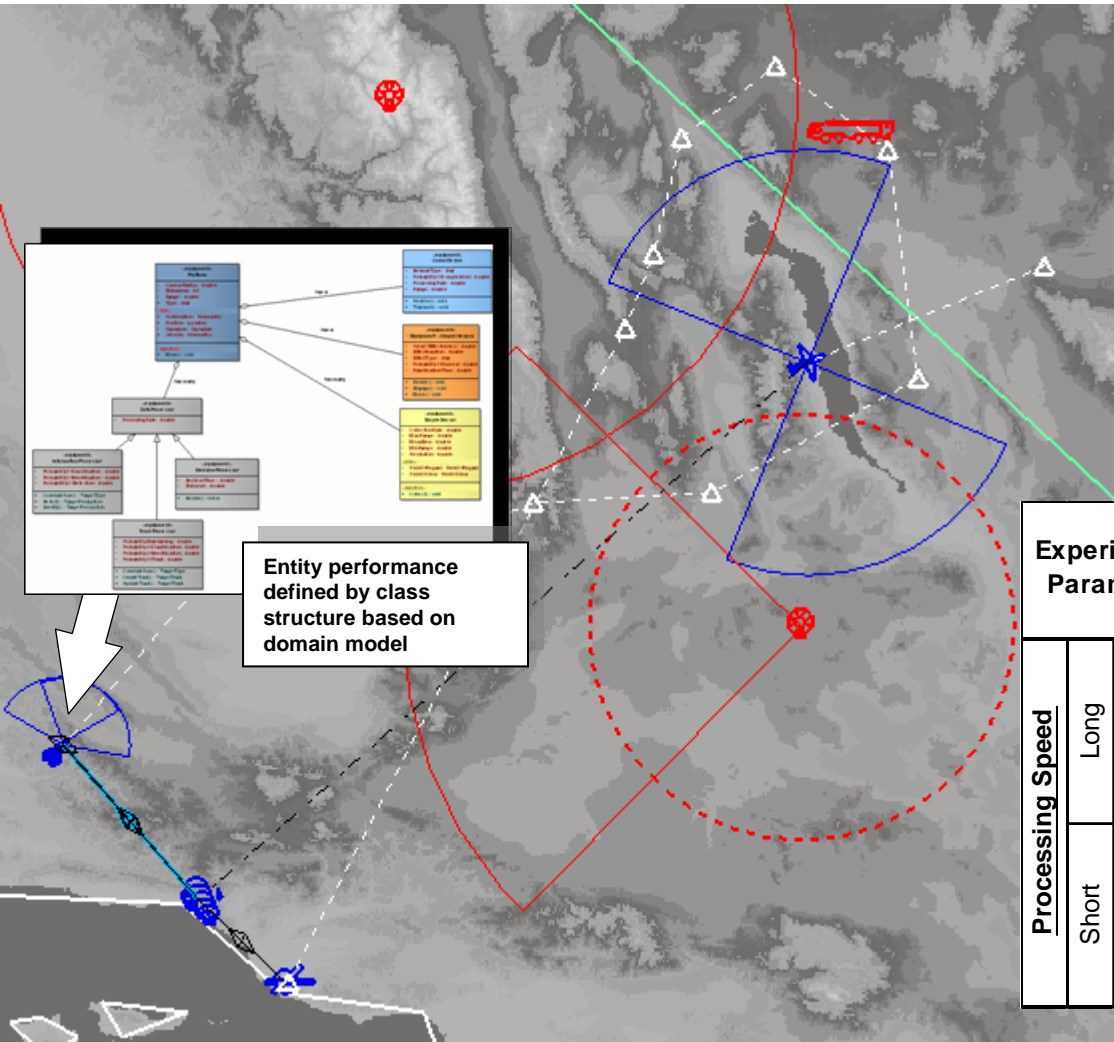
- **Standardize and define the problem space.**
 - **Use as a project glossary/naming convention.**
 - **Focus on real-world (problem domain) objects.**
- **Document domain structure.**
 - **Organize around key problem domain factors.**
 - **Encapsulate (sub) systems.**
 - **Simplify and/or standardize interfaces.**
 - **Identify systems and their interrelationships.**
 - **Enable analysis of the concepts.**
- **Provide foundation for follow-on artifacts (e.g., use cases, activity models, state diagrams, M&S software design, etc.).**

Complex systems-of-systems require a design approach that formalizes the mapping between behaviors and entities and remains flexible and resilient to change.

Why It Matters: Research, development & analysis

- **Establishes framework for factor identification and selection including:**
 - **Structure: defines systems and capabilities.**
 - **Behavior: defines functional processes.**
- **Defines the domain entities each group must focus on to achieve their objectives.**
 - **There will be overlap identified – requiring coordination.**
- **Provides the terminology and factors for development of:**
 - **Tests and experiments including specification of alternatives and trades, and scenario development requirements.**
 - **System functions which emerge from domain entities: methods, attributes, and interfaces.**
- **Supports analysis at different levels of abstraction/fidelity without changing the underlying model/architecture.**

Analysis Example



- Analysis factors selected using domain entities and derived artifacts
 - Independent of simulation tool

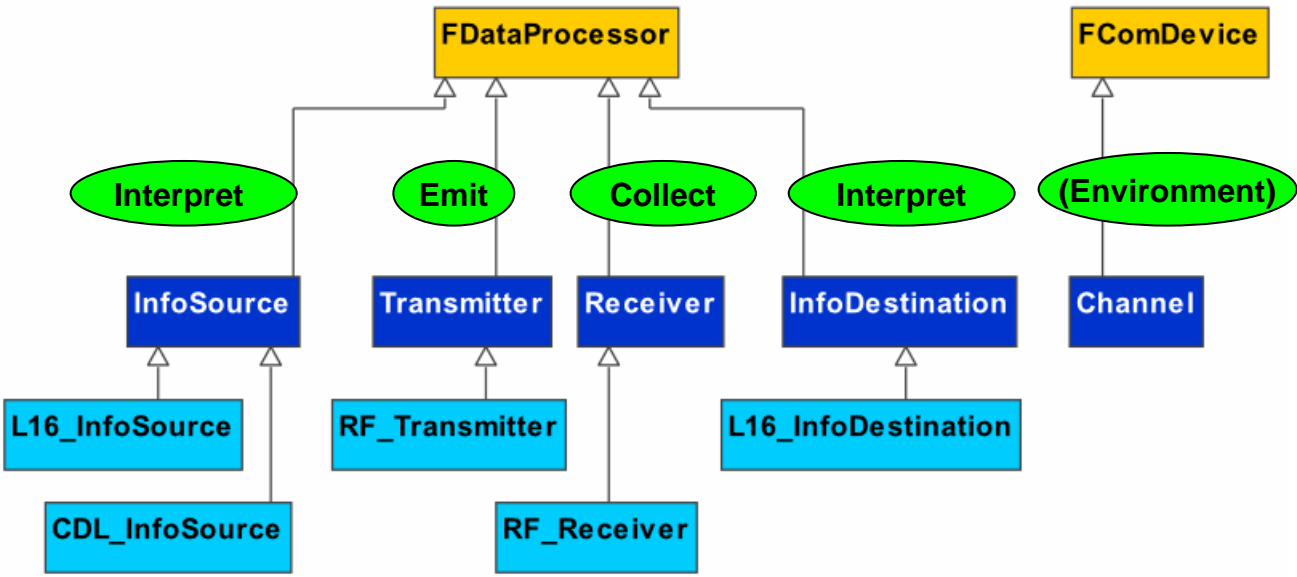
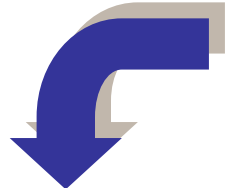
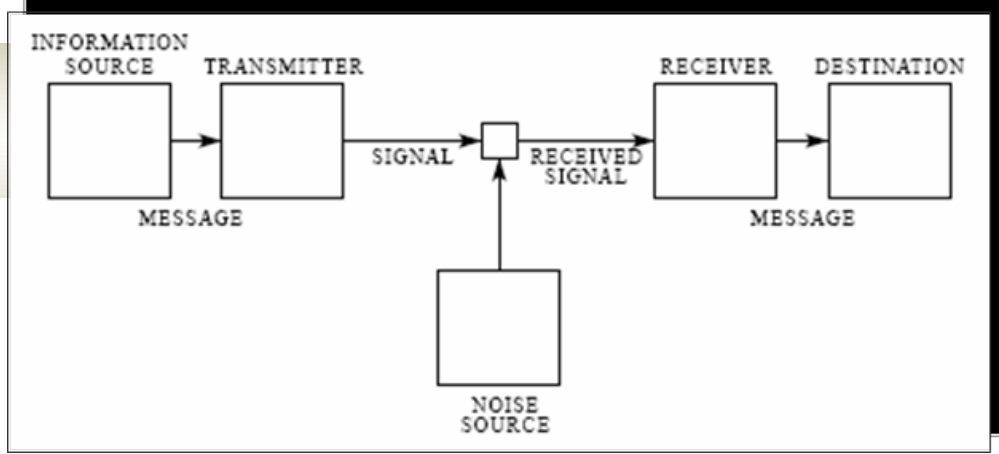
Experimental Parameters		Endurance					
		Short		Long			
		Speed					
		Slow	Fast	Slow	Fast		
Processing Speed	Long	Network Throughput	Low				
			High				
	Short	Network Throughput	Low				
			High				

Results provide assessment of the efficacy of the system alternatives and the sensitivity of the factors on one another.

Why It Matters: Model and Simulation, S/W Eng.

- **Helps identify where M&S software should be developed.**
- **Represents the top level classes and associations for M&S design.**
 - **Forms a foundation for software design model (UML).**
 - **Models are derived, developed, or specified from the domain-level superclasses.**
- **Reusability, extensibility, and re-configurability of models developed using these techniques enable assessment of complex network centric issues.**
- **Identifies M&S needs/requirements for potential assignment to available tools (including legacy simulations).**
 - **i.e., once a simulation need is identified, existing tools can be evaluated against it.**

M&S Example



Sim Framework Basic Types

Metamodel Activities

Generic Model Classes

Inherited Model Classes

- Domain entity becomes class for model implementation.
- Model parameters used to compose system representation.
- Domain artifacts provide basis for evaluation of existing simulations.

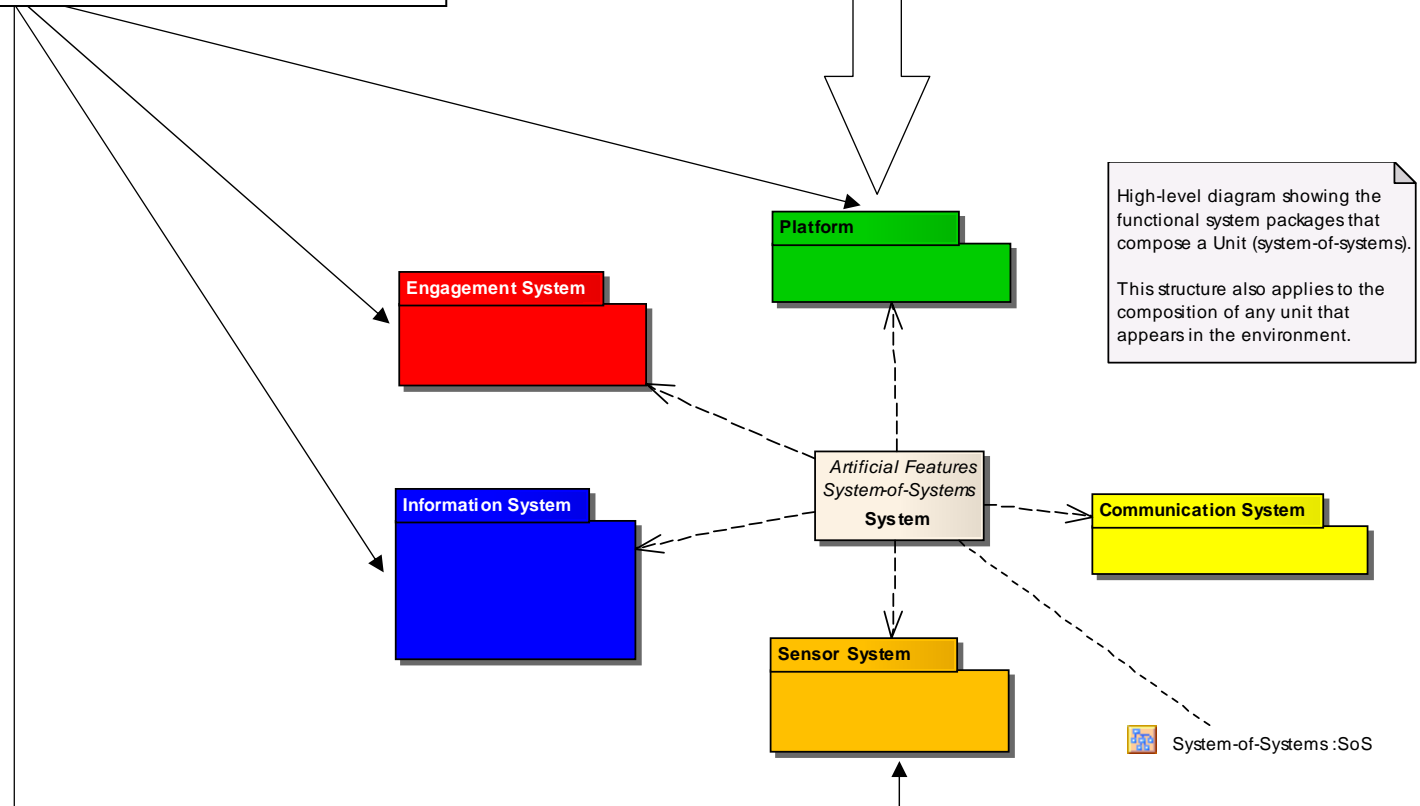
Why It Matters: Systems Engineering

- Tracks overall system-of-systems development and interactions.
- Provides insight into the system/subsystem alternatives.
- Useful as a foundation for system architectures.
- Supports requirements development/refinement.
- Identifies redundant or superfluous systems/processes.
- Simplifies design.
- Identifies capability shortfalls.
- Identifies program risks:
 - Technical readiness,
 - Interoperability challenges,
 - Critical technologies.
- Stored in a database, which can be linked to other SE products.

SE Example

Requirement: The system will engage advanced air-to-air and surface-to-air threats based on the rules of engagement.

Program Database:
Requirements, domain model and other artifacts, MS&A information, project management info, etc.



Why It Matters: Business Processes

- **Identifies**
 - **Areas of responsibility for different stakeholders.**
 - **Maps to project WBS.**
 - **Shortfalls in coverage/investments.**
 - **Return on investment and related tech maturity of individual systems.**
 - **Risks to the overall goals of the program.**
- **How is this done?**
 - **Each domain entity is related to activities supporting development of applications, data or products needed to accomplish objectives and goals.**
- **Represents a unified simulation-based acquisition process with all components interconnected via the UML-based architecture.**

Business Process Example

Task Name	Duration	Start	Finish
<input type="checkbox"/> Project X	56 days	Wed 2/27/08	Wed 5/14/08
<input type="checkbox"/> Scenario Development	56 days	Wed 2/27/08	Wed 5/14/08
<input checked="" type="checkbox"/> Identify Target Sets	42 days	Wed 2/27/08	Thu 4/24/08
<input checked="" type="checkbox"/> Assess TCT characteristics	4 days	Thu 4/24/08	Tue 4/29/08
<input checked="" type="checkbox"/> Scenario development	11 days	Wed 4/30/08	Wed 5/14/08
<input checked="" type="checkbox"/> Intelligence Collection Process Design	1 day	Thu 4/24/08	Thu 4/24/08
<input type="checkbox"/> Command and Control Process Design	1 day	Thu 4/24/08	Thu 4/24/08
Develop Baseline Processes/Metrics	1 day	Thu 4/24/08	Thu 4/24/08
Develop Advanced Processes	1 day	Thu 4/24/08	Thu 4/24/08
<input checked="" type="checkbox"/> Engagement System Development	1 day	Thu 4/24/08	Thu 4/24/08
<input checked="" type="checkbox"/> Develop assessment process	1 day	Thu 4/24/08	Thu 4/24/08
<input type="checkbox"/> Model and Simulation Development	1 day	Thu 4/24/08	Thu 4/24/08
Select Model Environment	1 day	Thu 4/24/08	Thu 4/24/08
Unit Development	1 day	Thu 4/24/08	Thu 4/24/08
System Development	1 day	Thu 4/24/08	Thu 4/24/08
Sensor Development	1 day	Thu 4/24/08	Thu 4/24/08
Communications Development	1 day	Thu 4/24/08	Thu 4/24/08
Information System Development	1 day	Thu 4/24/08	Thu 4/24/08
Engagement System Development	1 day	Thu 4/24/08	Thu 4/24/08
<input checked="" type="checkbox"/> Conduct Analysis	1 day	Wed 2/27/08	Wed 2/27/08
<input checked="" type="checkbox"/> Write Report	1 day	Wed 2/27/08	Wed 2/27/08

- Project's work breakdown structure(s) and activities based on domain entities and follow-on artifacts.
- Enables improved governance.
- Enhances task estimation and risk assessment.

Domain Modeling Assessment

Goods

- Replacement of legacy applications (incremental implementation)
- Gain understanding of current capabilities, analyze costs, compare w/ proposed replacement systems
- Make future programs more efficient
- Better risk management
- Potential for program-wide database or knowledge management system
- Reuse across portfolio
- Common foundation/linkages for program tasks (s/w and system engineering, analysis, business processes)
→ **CONVERGENCE**
- Standardization
- Greater accessibility to stakeholders
- Lasting documentation (domain longevity)
- Tool/simulation/code agnostic

Others

- Up-front costs
- Understanding new tools, language, processes
- Personnel and skillset availability
- Inertia of DoD acquisition practices
- Cultural resistance

Summary

- **Domain modeling is not a new idea.**
 - Under-utilized in the DoD community?
- **Useful for large and complex projects.**
- **Powerful for discovering relationships between entities within the domain and analyzing technical problems.**
- **Invariant Properties.**
- **Useful for linking diverse projects and processes into a unified portfolio.**
- **Flexibility and reusability increases efficiency of acquisition processes.**
- **Convergence:** the domain model provides a common foundation for M&S, architecture, analysis, and project management tasks.

Questions?

Contact Info:

Nathaniel Horner Nathaniel.Horner@jhuapl.edu
(240) 228-2908

Steve Topper Steve.Topper@jhuapl.edu
(240) 228-2701

“It is far better to grasp the Universe as it really is than to persist in delusion, however satisfying and reassuring.”

- *Carl Sagan*
(1934-1996)