



# OneSAF Behaviors

## OneSAF User Conference

Clark R. Karr

13 April 2003



# Outline

- **Introduction and Background**
- **Modeling Infrastructure**
- **Composite Behaviors**
- **Developing Primitive Behaviors**
  - **Behavior Composability**
  - **Guiding Principles**
  - **Strategies**
- **Knowledge and Actor/Agent Communications**
- **Sources of Information**
- **Primitive Behaviors**



# OOS Modeling Goals

## Requirements:

- Support a wide range of existing and future military operations.
- Support multiple levels of resolution.
- Support multiple user domains.
- Support both training and analysis.
- Interoperate with other simulations.

As a result, **composability** is a pervasive design goal.



# Fundamental Concepts: Atomic Components

- **Primitive behaviors (PB)** provide chunks of doctrinal functionality from which more complex behavior models are built. Primitive behaviors interact with behavior agents.
- **Agents:**
  - Behavior agents provide command and control capabilities such as planning, plan execution, and situation assessment.
  - Physical agents are the “middlemen” between behavior agents and physical models. They represent the effectors and perceptors of simulated entities.
- **Models**
  - Behavior Models provide doctrinally correct answers to behavior questions asked by behavior agents.
  - Physical Models provide physics of simulated entities for mobility, weapons, vulnerability, sensing, and communications.



# Fundamental Concepts: Model Compositions

- **Composite behaviors** are composed of primitive behaviors and other composite behaviors.
- **Entities** represent equipment platforms such as tanks, soldiers, and trucks. They are composed of physical models and behavior agents and are controlled by behavior models.
- **Units** represent the command and control of organizations such as platoons, companies, and battalions. They are composed of behavior agents and are controlled by behavior models.

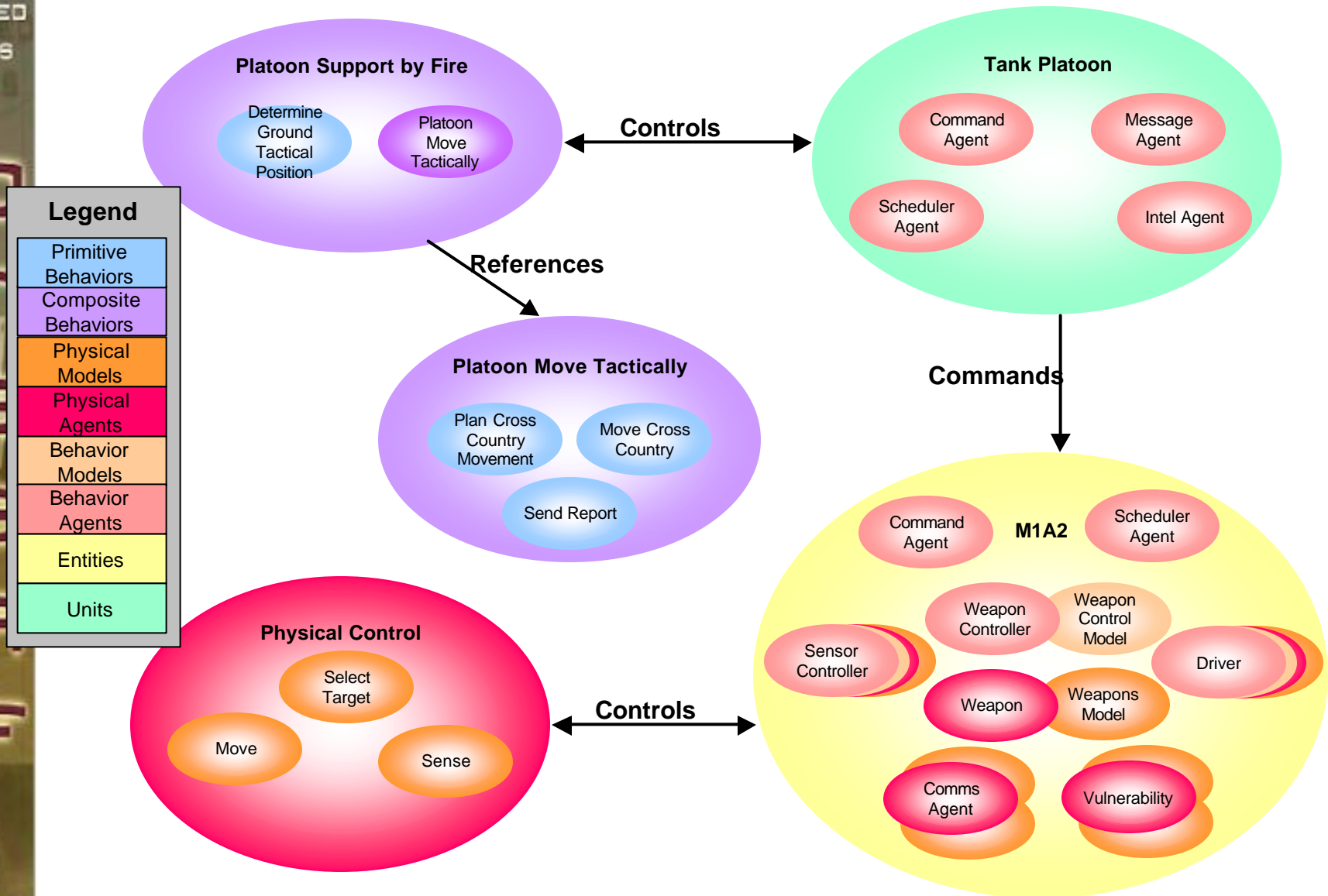


# OneSAF Command and Control (C2)

OneSAF C2 is based on three concepts:

- **Agents:**
  - Software artifacts exhibiting autonomy, reactivity, inferential capability, temporal persistence, and cooperative behavior.
  - Agent control within an actor is implemented as a subscription mechanism through the actor's blackboard. Agents receive only the triggers they're interested in, may send any trigger, and are unaware of other agents.
- **World Model:**
  - Store of what is known, suspected, inferred, and/or correlated.
  - One World Model per actor—an actor's agents share this information.
- **Hierarchical decomposition of behavior:**
  - Behaviors (missions and tasks) are hierarchically decomposed at each echelon into trees with primitive behaviors at the leaf nodes and composite behaviors at the root and interior nodes.
  - Execution control involves traversing the leaves of behavior trees in accordance with temporal, phase, and command constraints.

# OneSAF Model Component Types

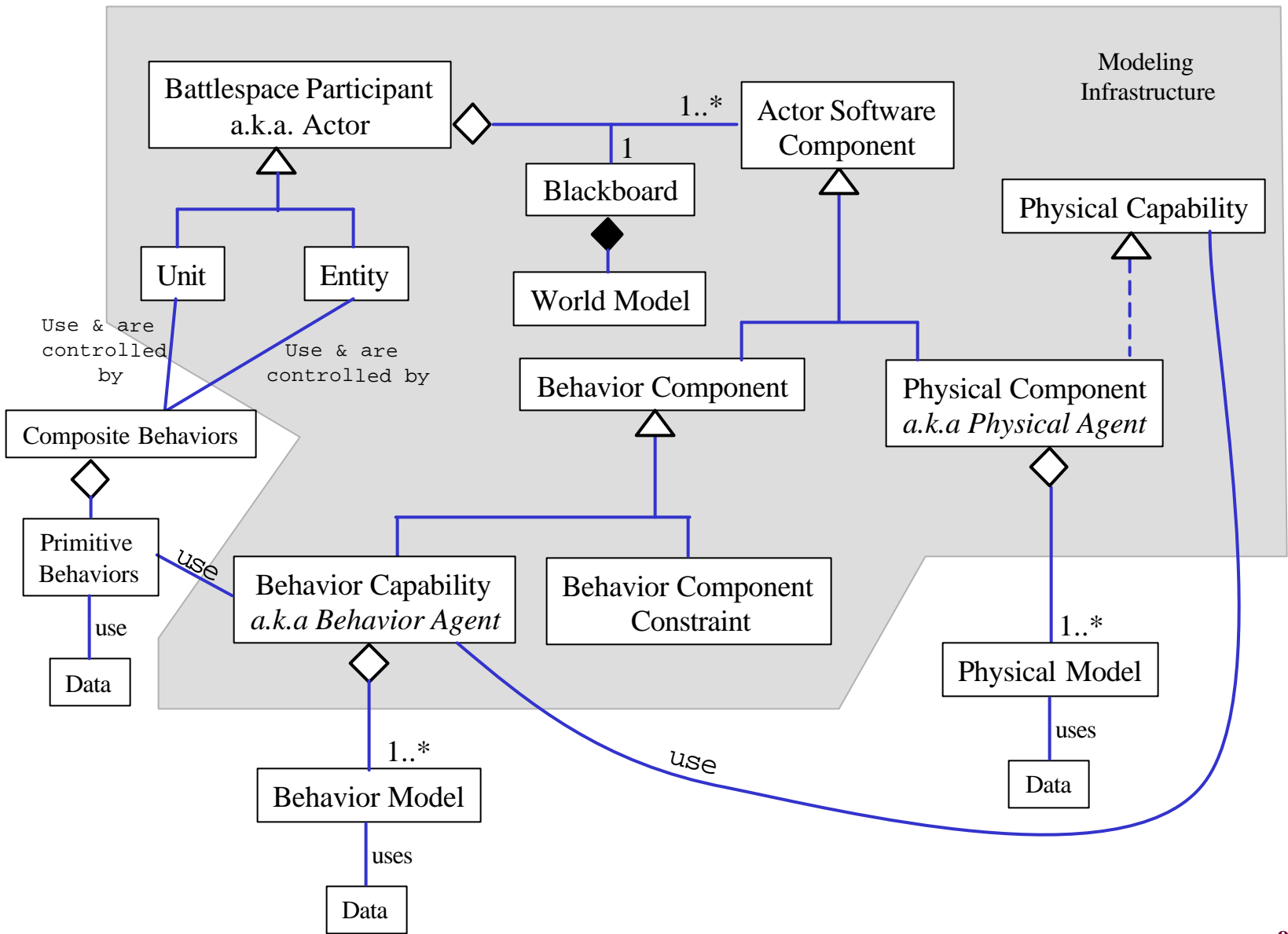




# Outline

- Introduction and Background
- **Modeling Infrastructure**
- Composite Behaviors
- Developing Primitive Behaviors
  - Behavior Composability
  - Guiding Principles
  - Strategies
- Knowledge and Actor/Agent Communications
- Sources of Information
- Primitive Behaviors

# Elements of the Modeling Infrastructure





# Example Agents

- **Behavior Agents:**
  - **Command**
  - **Scheduler**
  - **Intel**
  - **Message**
  - **Driver**
  - **Direct Fire Weapons Controller**
  - **Fire Direction Center**
- **Physical Agents:**
  - **Weapon**
  - **Radio**
  - **Sensor**
  - **Mobility**
  - **Transport**
  - **Vulnerability**



# Outline

- Introduction and Background
- Modeling Infrastructure
- **Composite Behaviors**
- Developing Primitive Behaviors
  - Behavior Composability
  - Guiding Principles
  - Strategies
- Knowledge and Actor/Agent Communications
- Sources of Information
- Primitive Behaviors



# Composite Behaviors

The OneSAF Company and below C2 strategy Blocks A, B, and C uses Hierarchical Decomposition. Ordered behaviors are hierarchically decomposed at each echelon into trees with primitive behaviors at the leaf nodes and composite behaviors at the root and interior nodes.

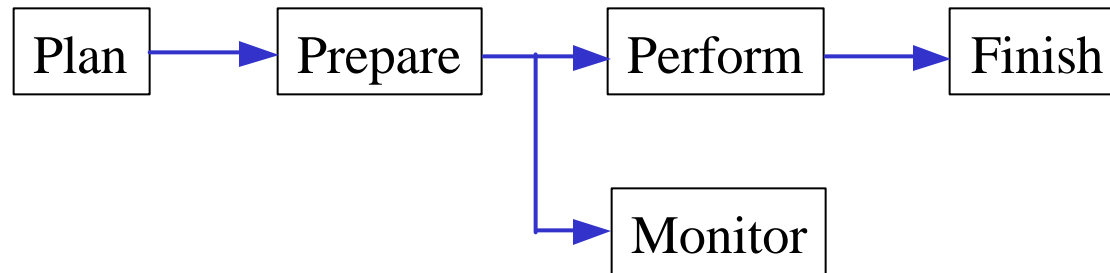
In general, to build behaviors:

1. Write code for primitive behaviors.
2. Use the Behavior Composer to build composite behaviors. The Behavior Composer allows placeholders for not yet written behaviors.

The Unit, Entity, and Behavior Composers use meta-data to tie behaviors to the appropriate entities and units.

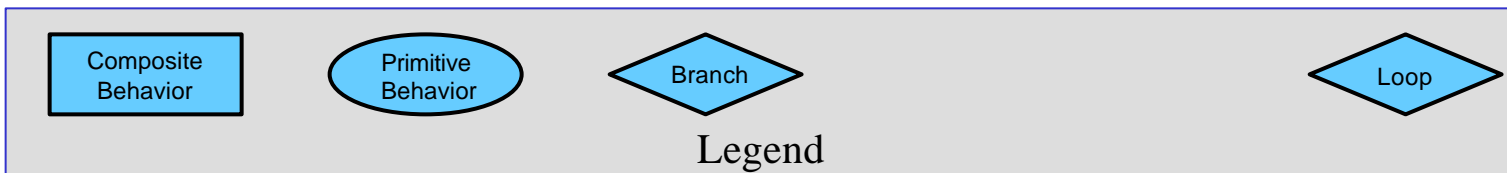
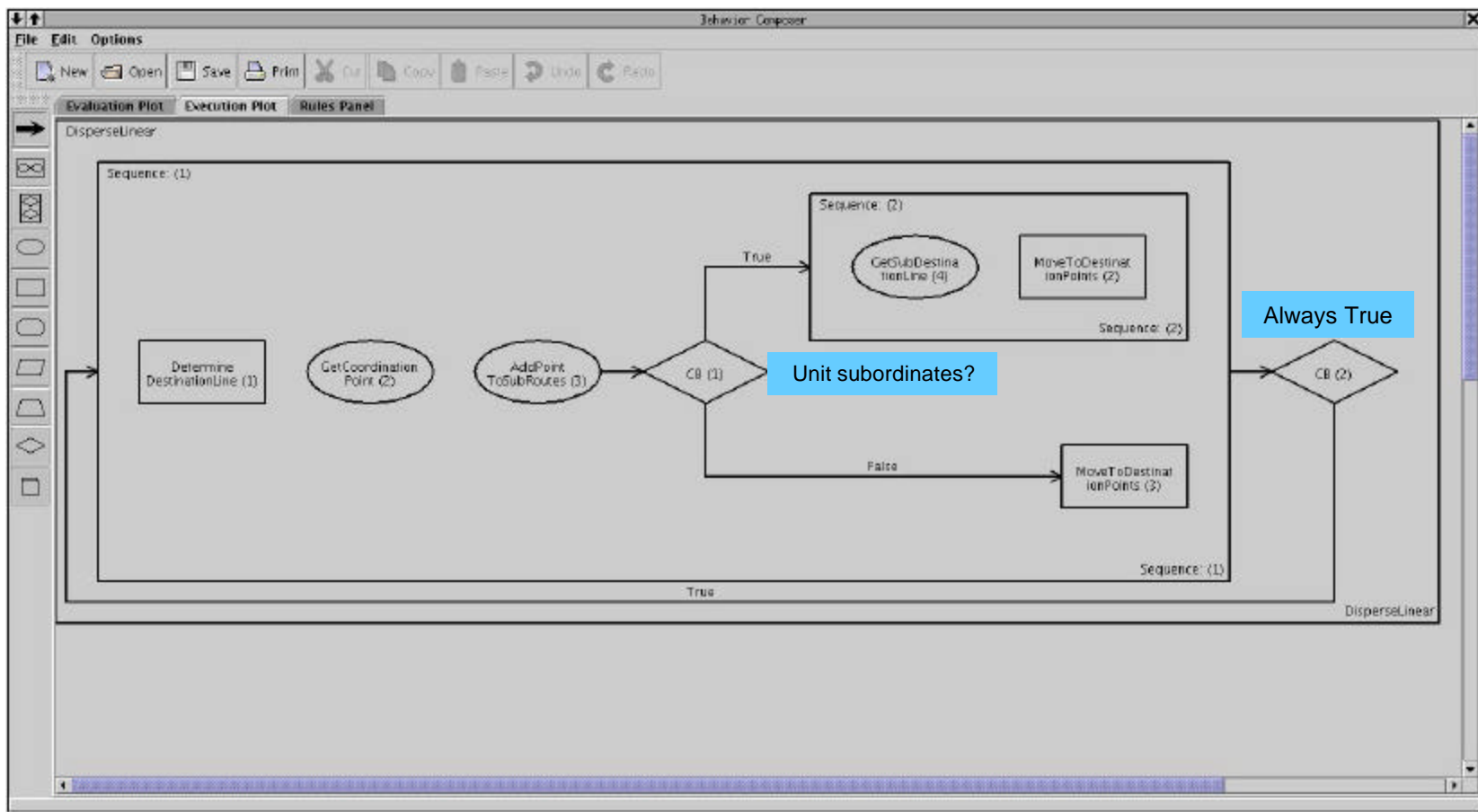


# Generic Composite Behavior



- **Each phase (plan, prepare, ...) is optional; e.g.,:**
  - Only the **Perform** phase is necessary for performing of a task.
  - Only the **Plan** phase is necessary for planning a task.
- **Within each phase:**
  - **Composite and primitive behaviors are organized into sequential and parallel temporal relations.**
  - **Branches and loops control the execution of behaviors.**

# Example Composite Behavior





# Outline

- Introduction and Background
- Modeling Infrastructure
- Composite Behaviors
- **Developing Primitive Behaviors**
  - **Behavior Composability**
  - **Guiding Principles**
  - **Strategies**
- Knowledge and Actor/Agent Communications
- Sources of Information
- Primitive Behaviors



# Behavior Composability

- An important element of OneSAF composability is the ability to compose behaviors.
- Our task is to develop a language for non-programmers to develop executable representations of behavior:
  - The set of Primitive Behaviors is the language dictionary.
  - Primitive Behavior inputs and outputs establish the language’s syntax (i.e., rules for arranging “words” to make valid sentences).
  - Composite behavior execution and evaluation plots are sentences.



# Guiding Principles

- **Principles for a useful, robust dictionary:**
  - **Because dictionary must make sense to non-programmers, our primitive behaviors' names and actions should make sense in the problem domain.**
    - We hide solution space decomposition through naming and software engineering (e.g., OO design, helper classes).
  - **Primitive behaviors must be semantically rich (i.e., usable in many contexts). Our dictionary should be:**
    - Concise.
    - Support a large set of sentences.
- **Behavior over-decomposition creates primitive behaviors each usable only in the context of a few other primitive behaviors. Such semantically weak primitive behaviors lead to:**
  - A verbose dictionary.
  - Limited set of sentences.



# Strategies for Coalescing AUTL tasks into Robust Primitive Behaviors

- Support Implicit Modeling
- Centralization
- Semantic Richness



# Strategy: Support Implicit Modeling

Support implicit modeling across BOSs (Battlefield Operating Systems) and categories; e.g., many AUTL tasks may simply require a sequence of three PBs:

1. **DetermineTimesToPerform(taskX, actor, location)**
  - Data lookup
  - Algorithm
2. **TimeDelay(timeToPerform)**
3. **AssertFactOrState(taskXDone(true))**

**With such an approach:**

- An arbitrary number of specialized Composite Behaviors can be developed and given doctrinally meaningful names.
- DetermineTimeToPerform could accept an optional data file name. If so, behavior developers could develop their own data files (according to OneSAF specifications).
- The assertion of facts will, in some cases, cause distributable state changes. Fact developers are responsible for making fact assertion cause state attribute distribution.



# Strategy: Centralization

## Centralization:

- Put conceptually similar things together.
- For example, MoveCrossCountry PB selects the appropriate movement algorithm given a “movement technique” discriminator.
  - New movement techniques require:
    - Code to implement movement.
    - Additions to movement technique enumeration.
  - This approach doesn't reduce the amount of code to write but it does:
    - Reduce the number of PBs
    - May illuminate areas for software reuse.

# Strategy: Semantic Richness

## Semantically rich PB:

- A semantically rich PB has an input set supporting multiple uses and internally selects the appropriate algorithm.
- For example, we avoid building a PB to find each type of tactical position. The terrain analysis involved may be done using a common approach; e.g., a cost function applied to cells of a tessellated map. Instead:
  - DetermineGroundTacticalPosition PB selects ground positions using algorithms suitable for terrain skin.
  - DetermineUrbanTacticalPosition PB selects interior, exterior, and near building positions using algorithms in a 2<sup>1/2</sup>d world.
  - DetermineAirTacticalPosition PB selects 3d positions.
- Variable Inputs – No support unless needed to reduce number of PBs.



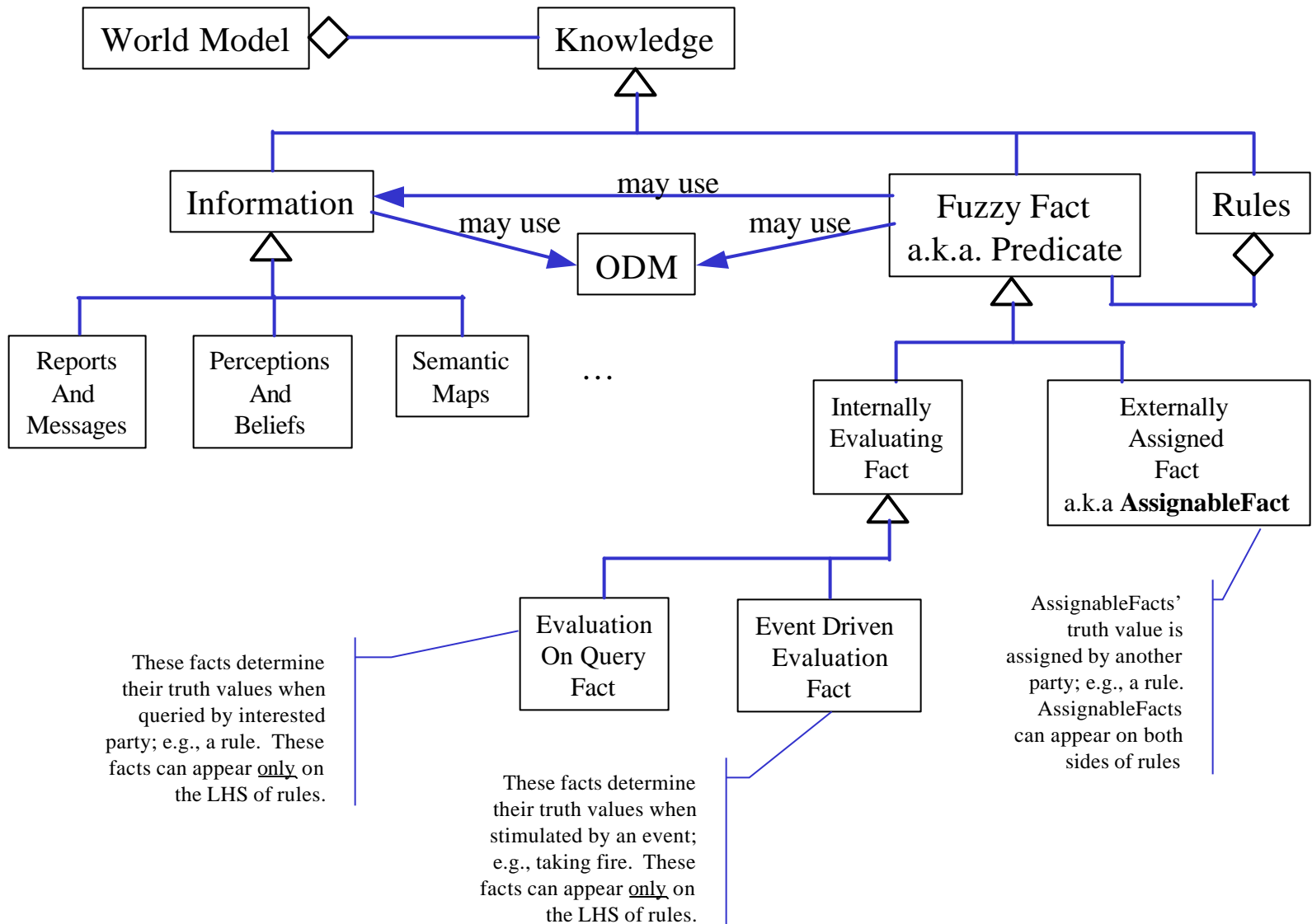


# Outline

- Introduction and Background
- Modeling Infrastructure
- Composite Behaviors
- Developing Primitive Behaviors
  - Behavior Composability
  - Guiding Principles
  - Strategies
- **Knowledge and Actor/Agent Communications**
- Sources of Information
- Primitive Behaviors



# Abstract taxonomy of Knowledge



# Communicating Among Agents and Behaviors

- **Events: inter-actor communication.**
  - **Order:** initiates behavior execution.
  - **Directives:** modifies executing behavior.
  - **Messages and Reports:** doctrinal messages.
  - **Element Coordination:** simulation specific, model coordination.
- **Triggers: inter-agent communication.**
- **Missions, phases, tasks, and overlays:**
  - **Mission:** collection of tasks for an actor.
  - **Phase:** collection of missions for a workstation (e.g., a unit).
  - **Task:** a composite behavior.
  - **Overlays:** used by composite behaviors.
  - **RDM has separate representations for each.**



# Outline

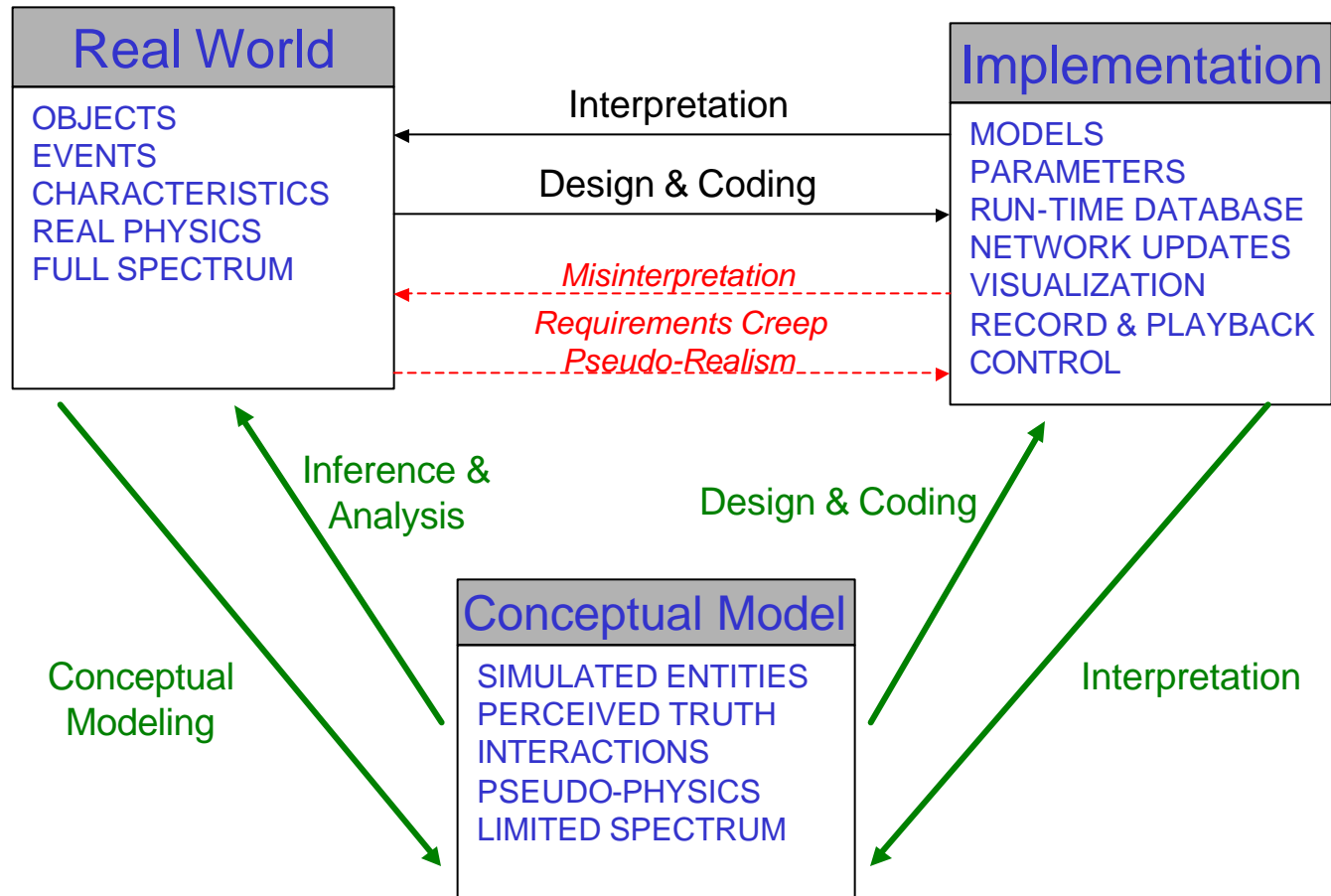
- Introduction and Background
- Modeling Infrastructure
- Composite Behaviors
- Developing Primitive Behaviors
  - Behavior Composability
  - Guiding Principles
  - Strategies
- Knowledge and Actor/Agent Communications
- **Sources of Information**
- Primitive Behaviors



# Sources of Information to Create Behaviors

- 1. Conceptual Model – In Block C, conceptual modeling precedes behavior development.**
- 2. Primitive Behavior Taxonomy**
- 3. KA/KE artifacts:**
  - **TDs: Task Descriptions**
  - **PSDs: Process Step Descriptions**
  - **Modeling notes: answers to questions created in conceptual modeling, requirements analysis, and design.**

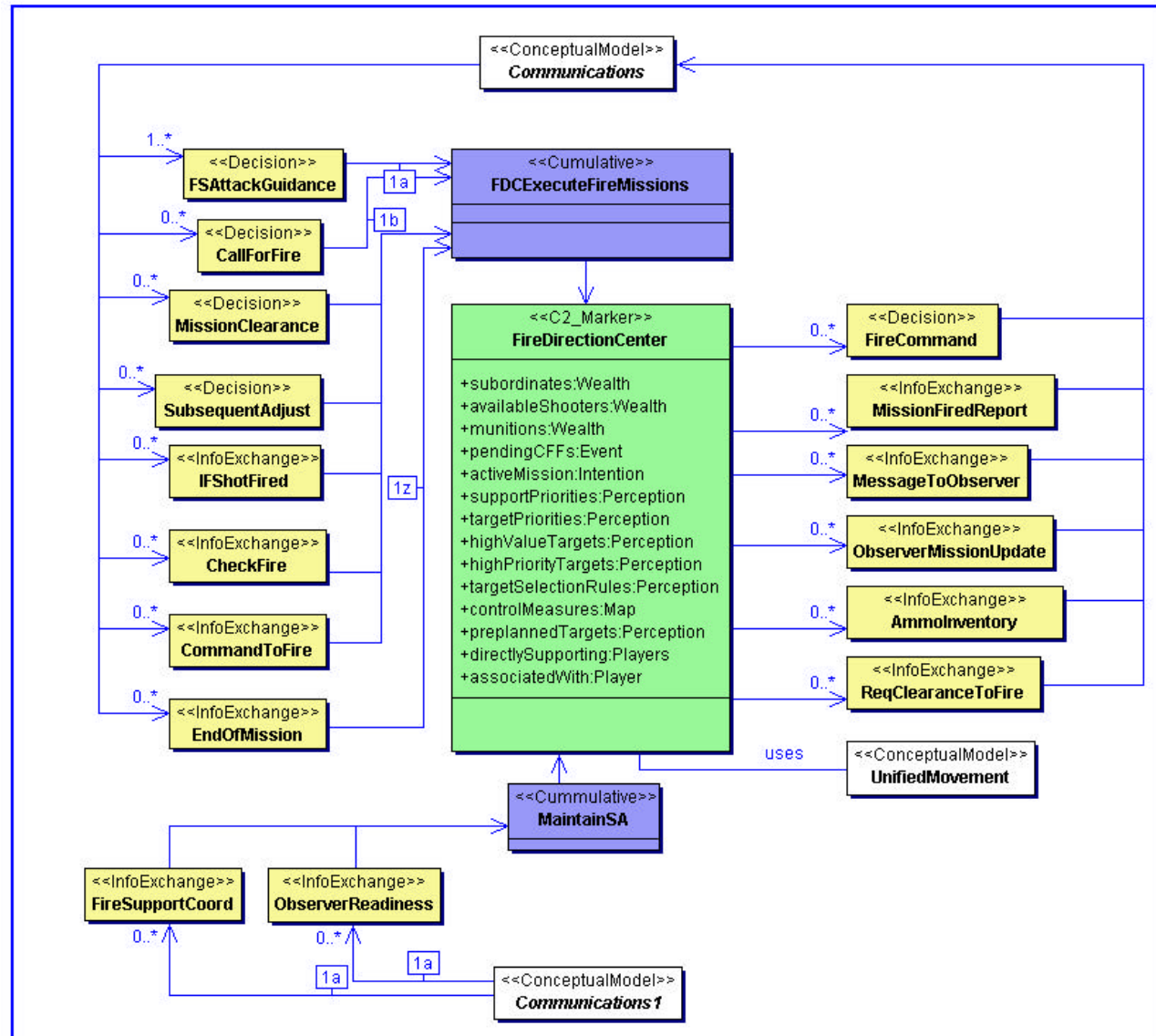
# Conceptual Model



**Simulating the real world is difficult and error prone.**

**A well-conceived, consistent intermediate model eliminates many problems by providing a model of the battlespace usable by all participants (customer, domain expert, developer, and user).**

# Execution Fire Direction Center Conceptual Model





# Outline

- Introduction and Background
- Modeling Infrastructure
- Composite Behaviors
- Developing Primitive Behaviors
  - Behavior Composability
  - Guiding Principles
  - Strategies
- Knowledge and Actor/Agent Communications
- Sources of Information
- **Primitive Behaviors**



# Common Primitive Behaviors

1. DetermineTimesToPerform
2. TimeDelay
3. AssertFactOrState
4. RetrieveFactOrState
5. DetermineSubordinates
6. RetrieveInteriorMap
7. TransferItems
8. EvaluateRules
9. CoordinateActorState
10. CreateLoadPlan
11. SendOrder
12. TaskOrganize



# Communication Primitive Behaviors

13. ReceiveMessage

14. SendMessage

Each message and report is prepared by a “PrepareXYZMessage” PB or “PrepareXYZReport PB”.



# Sensing Primitive Behaviors

14. ImplementEmissionSchedule



# Movement Primitive Behaviors

- 15. PlanCrossCountryMovement
- 16. MoveCrossCountry
- 17. PlanUrbanMovement
- 18. MoveInUrbanArea
- 19. PlanAirMovement
- 20. MoveAlongAirRoute
- 21. KeepSupportedUnitInRange
- 22. DetermineBypassOrBreachTechnique



# Combat Primitive Behaviors

23. DetermineGroundTacticalPosition
24. DetermineUrbanTacticalPosition
25. DetermineAirTacticalPosition
26. DetermineSubordinatesLocationsInTacticalPosition
27. DetermineEngagementCriteria
28. SetEngagementCriteria
29. UseWeapon
30. SelectCombatLoad
31. RetrieveControlMeasure
32. DetermineControlMeasure
33. DetermineFireDistribution
34. AllocateSubordinatesToProtectAssets
35. DesignateSubordinateToAttackTarget



# Maneuver Primitive Behaviors

- 36. PlanCounterAmbushActions (2.2.4)
- 37. PlanAssault
- 38. PlanDefense
- 39. PlanInteriorClearing



# Fire Support Primitive Behaviors

- 36. RequestAndCoordinateFireSupport (implemented as CallerForFire Behavior Agent instead of a PB)
- 37. CoordinateFireMissions (implemented as CoordinateFire Behavior Agent instead of a PB)
- 38. ExecuteFireMissions
- 39. ReceiveAndShootFireCommand
- 40. PlanAirToSurface Attack (3.3.1.2)
- 41. ConductAirToSurface Attack (3.3.1.2)
- 42. RequestAirToSurfaceAttack (3.3.1.2.1)
- 43. EmployCloseAirSupport (3.3.1.2.2 )



# Air Defense Artillery Primitive Behaviors

44. PrepareADAPlan

45. ExecuteADAPlan

46. CoordinateADADefense

# Mobility/Counter-mobility/Survivability Primitive Behaviors

- 47. PlanConstruction
- 48. PlanRiverCrossing (5.1.1.3 Conduct River Crossing Operations)
- 49. PlanSEAD
- 50. IdentifyFriendlyVulnerabilities (5.3.5.6.1 EEFI Essential Elements of Friendly Information)
- 51. CreateObscurantPlan





# Combat Service Support

52. Organize LOGPAC

53. Regulate Movement (6.3.1.2 )

54. Treat Or Repair



# Command and Control

55. Establish Target Priorities (7.4.4)