# Distributed Simulation / High Level Architecture Overview: *Engineering Principles of Combat Modeling and Distributed Simulation*

**Andreas Tolk, Ph.D.**
*Computer Science Principal, The MITRE Corporation**
*Adjunct Professor Old Dominion University*

** The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions or viewpoints expressed by the author.*

**Approved for Public Release; Distribution Unlimited. Case Number 15-3164**

# Objectives of this Session

1. The participant understands the general challenges for creating distributed simulation systems that comprise independently developed simulation systems that provide the required functionality.
2. The participant understand how the IEEE 1516-2010 High Level Architecture (HLA) standard addresses these various challenges in general.

# Structure of the Session

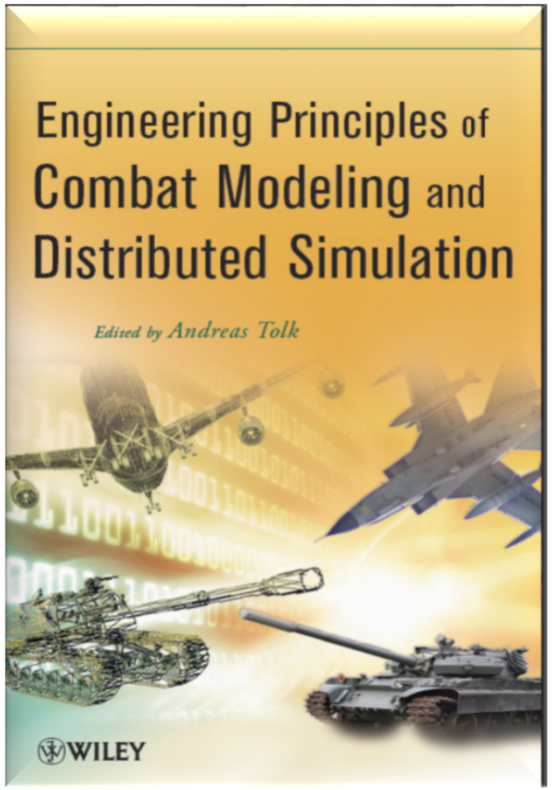
- Introduction to Constructive Simulation
  - Elements and entities of interest to constructive simulation
  - Shoot, look, move, and communicate in a situated synthetic environment
- Challenges of federating Simulation Systems
  - Aligning of entities
  - Synchronization of activities and events
  - Assumptions and constraints
  - Integrateability, Interoperability, and Composability
- High Level Architecture
  - Rules, Runtime Infrastructure, and the Object Model Template
  - DSEEP
- Some Alternatives
  - DIS, TENA, and Semantic Web

Part One

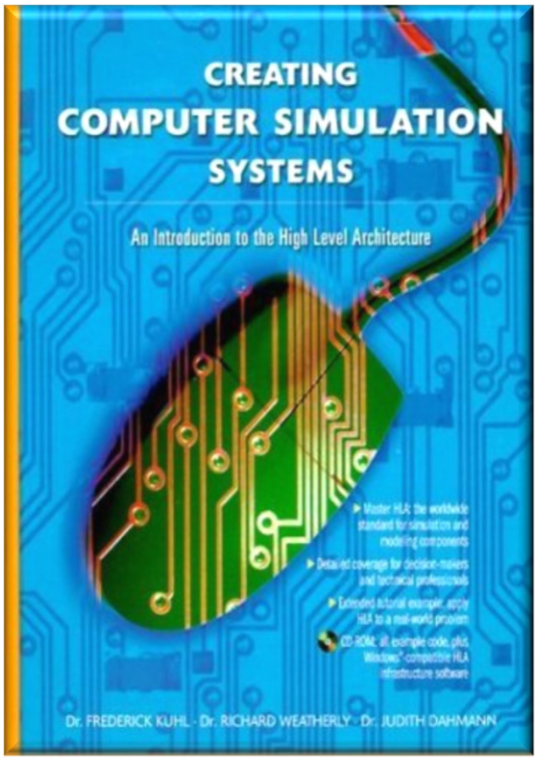
# GENERAL CHALLENGES

# Books on this Topic

Andreas Tolk:
"*Engineering Principles of Combat Modeling and Distributed Simulation*,"
Wiley, 2012




Fred Kuhl, Judith Dahmann
and Richard Weatherly:
"Creating Computer Simulation Systems,"
Prentice Hall, 1999



# Types of Simulation

| | Personnel | Weapon System | Environment |
|---|---|---|---|
| **Live Simulation** | Real Soldiers | Real Weapon Systems | Real Environment |
| **Virtual Simulation** | Real Soldiers | Simulated Weapon Systems | Synthetic Environment |
| **Constructive Simulation** | Simulated Soldiers | Simulated Weapon Systems | Simulated Environment |

# Board Games and Constructive Simulation

- Focus of the first session are the main activities of traditional combat
  - Influenced by strategic board games
    - Chess
    - Go
    - "*Kriegsspiel*" (war game)
  - Elements of such board games
    - Board with cells
    - Figures with capabilities
    - Rule sets
  - Main activities of traditional combat
    - Move
    - Shoot
    - Look
    - Communicate

© http://www.nixiepixel.com

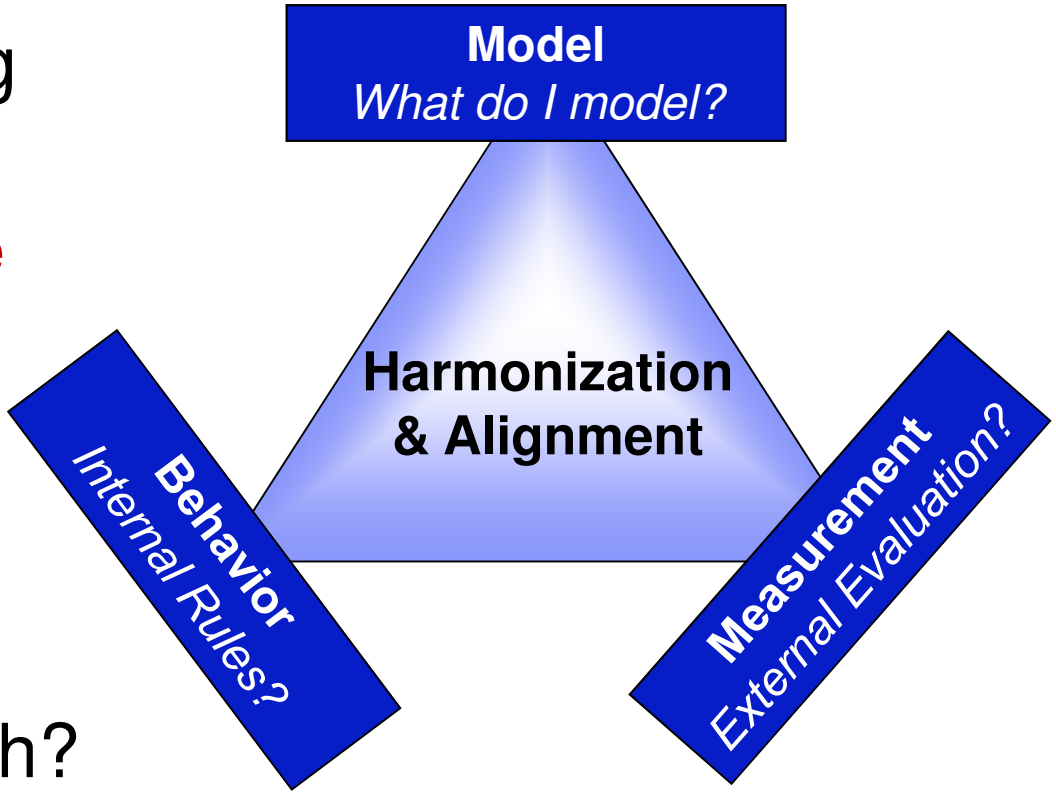
# Entities and Scenarios

- The Scenario Elements to be defined are highly dependent on the problem to be solved
  - What is the Mission
  - What Capabilities are important
  - What Relations are important
- A Scenario is a description of the
  - Area
  - Environment
  - Means
  - Objectives and
  - Events

  related to a conflict or a crisis during a specified time frame suited for satisfactory study objectives and the problem analysis directives

# Harmonization and Alignment Principle

- The Crux of Building the Right Model
  - We only can simulate and analyze what we model
  - Every piece of detail added increases the complexity
- How much is enough?



# Characteristics of High Resolution Models

- High resolution combat modeling is dealing with **Detailed Interactions** of **Individual Combatants**
  - Each modeled entity has its **Individual State Vector** describing the unique situation and own perception
  - Interactions are generally modeled **one-on-one**
  - Every **Single Process** is computed individually
  - Every single process can be modeled stochastically with **Individual Probability** and density functions

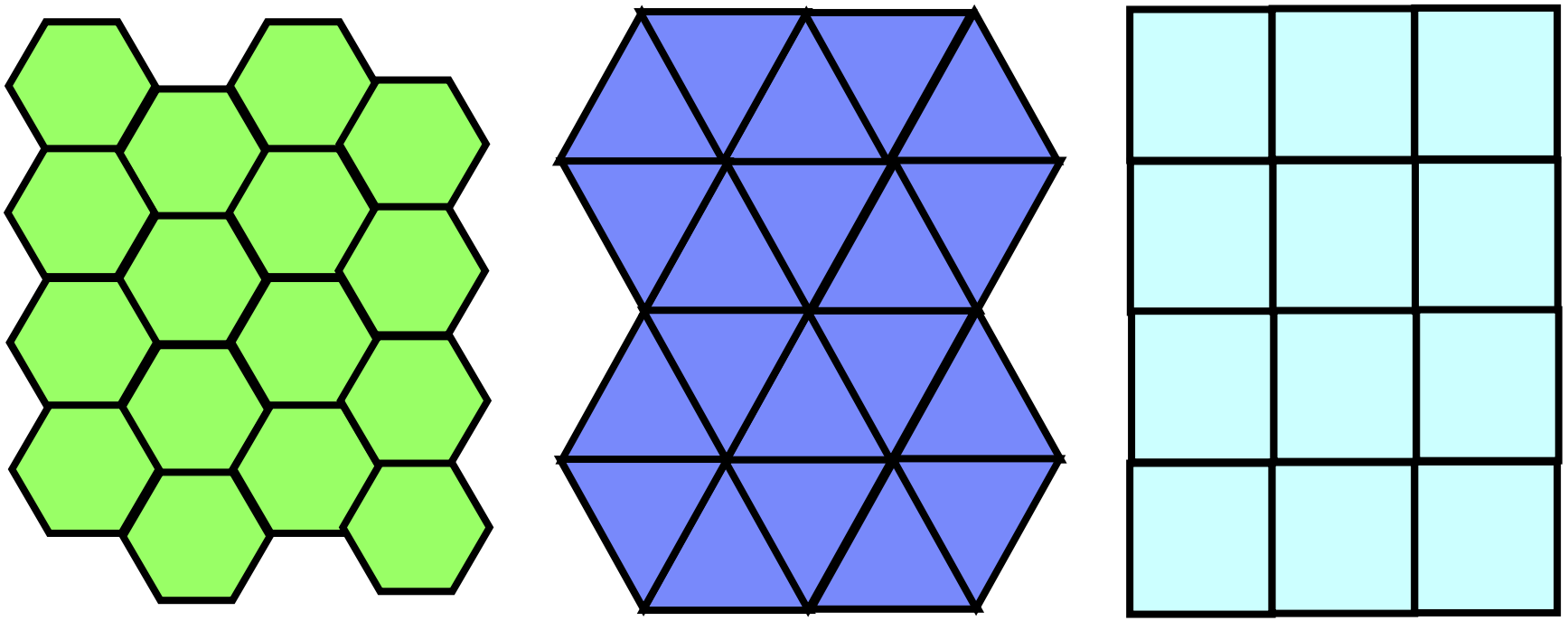
# Characteristics of Aggregated Models

- Aggregated combat modeling is dealing with **Aggregated Interactions** of **Groups of Combatants**
  - Each modeled entity has its **State Vector** describing the unique situation and perception, but the entity is describing a group a combatants sharing this state vector
  - Interactions are generally modeled **many-on-many**
  - Processes are computed based on **Group Assumptions**, e.g. movement of the center-of-mass of a company
  - Process can be modeled stochastically with probability functions derived from the **Underlying Individual Processes**' probability functions

# What is the Environment?

- Terrain
  - Elevation
  - Surface Type
  - Covering
- Sky/Space
  - Clouds
  - Winds
  - Temperature
- Underwater
  - Temperature
  - Salinity
  - Current
- Littoral
  - Elevation / Depth
  - Surface Type
  - Current

# “Chessboards” for Terrain Modeling



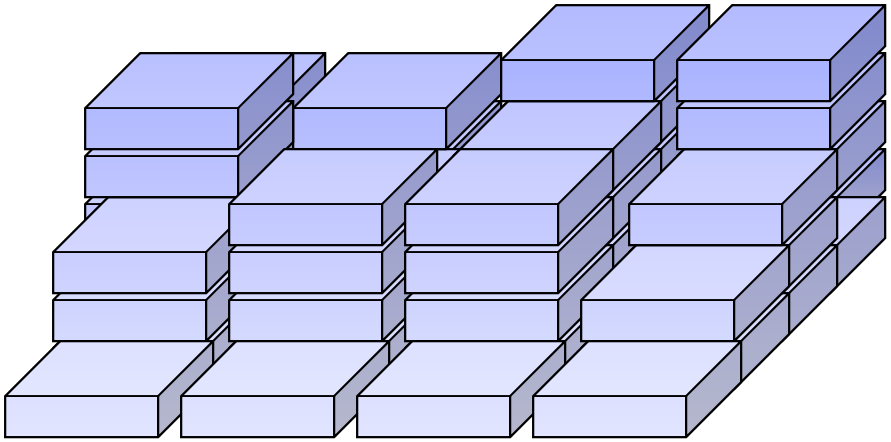
Hexagon

Triangle

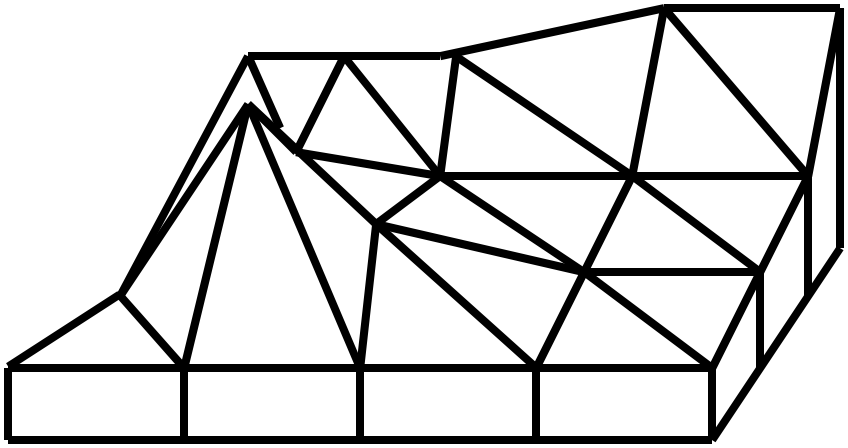
Rectangle / Rhombus

# Examples of Explicit Terrain Models

Explicit Grid Terrain Model

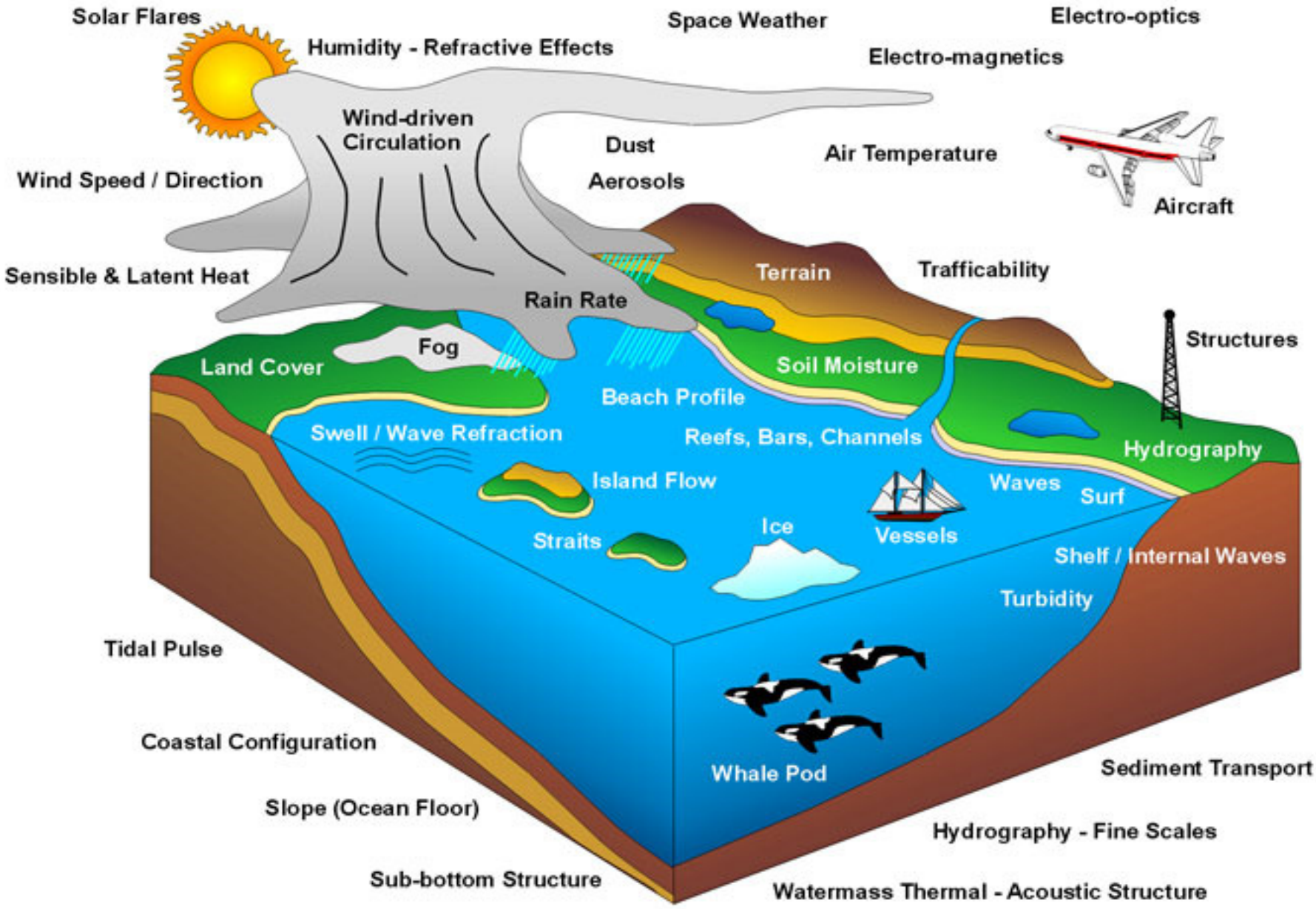


Explicit Surface Terrain Model



# Environmental Models

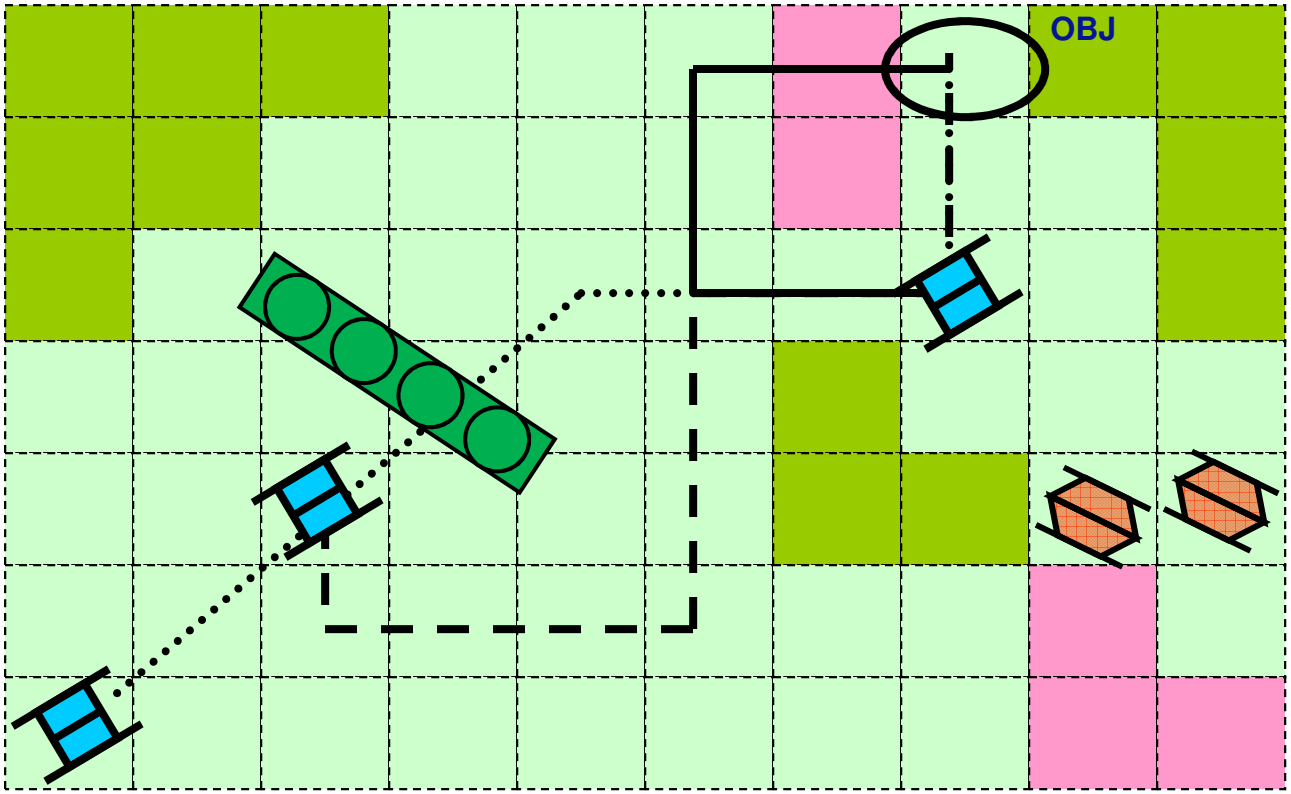
- Environment is more than terrain elevation, e.g.
  - Terrain roughness
  - Urbanization and/or Forestation
  - Vegetation and soil type
  - Rivers, Roads, and Bridges
  - Obstacles and barriers (look and move)
  - Sea layers of different salt density and temperature
  - Clouds, fog, smoke
  - “Dirty battlefield” effects
  - Precipitation
  - Weight bearing capacity
  - Passable to different types of objects
  - Season (Are trees leafy?)


Solar Flares
Space Weather
Electro-optics
Humidity - Refractive Effects
Electro-magnetics
Wind-driven Circulation
Dust
Air Temperature
Wind Speed / Direction
Aerosols
Aircraft
Sensible & Latent Heat
Terrain
Trafficability
Rain Rate
Structures
Fog
Land Cover
Soil Moisture
Beach Profile
Swell / Wave Refraction
Reefs, Bars, Channels
Hydrography
Island Flow
Waves
Surf
Straits
Ice
Vessels
Shelf / Internal Waves
Turbidity
Tidal Pulse
Coastal Configuration
Whale Pod
Sediment Transport
Slope (Ocean Floor)
Hydrography - Fine Scales
Sub-bottom Structure
Watermass Thermal - Acoustic Structure

# Movement and Terrain

- Explicit grid models
  - Use a grid to store the required terrain characteristics for the movement model
- Explicit patch models
  - Breaking up the battlefield into areas where the characteristics do not change
- Implicit mobility models
  - Computation of mobility factors or multipliers used to estimate the mobility based on a base mobility for the overall battlefield
- Network models
  - Using physical nodes and connecting paths to model mobility, often used for roads, bridges, etc.

# Movement and Optimization



# Ground Truth and Perception

- Sensing is pivotal to create a perception
- Ground Truth is the ***REALITY*** as modeled with the systems
  - Ground Truth "knows" all state vectors of entities and terrain in all detail
  - Ground Truth also knows all plans, doctrines, orders, etc.
- Sensors "observe" the Ground Truth and create observations
  - Often a subset of Ground Truth (cookie cutter principle)
  - Sometimes additional algorithms are applied to change some values (filters)
- Perception is unique to the perceiving entity
  - What does the individual know about the environment and the entities therein?
  - Based on own information (sensors) plus reports and messages

# Types of Sensors

- Acoustic sensors
  - Microphones and hydrophones
- Chemical sensors
  - Artificial noses/tongues
  - Biological and chemical anomalies
- Electromagnetic sensors
  - Radars
  - Observe electric and magnetic field
- Thermal sensor
  - Infrared sensors
  - Night goggles
- Optical sensors
  - Subset of electromagnetic
  - Binoculars, telescopes, etc.

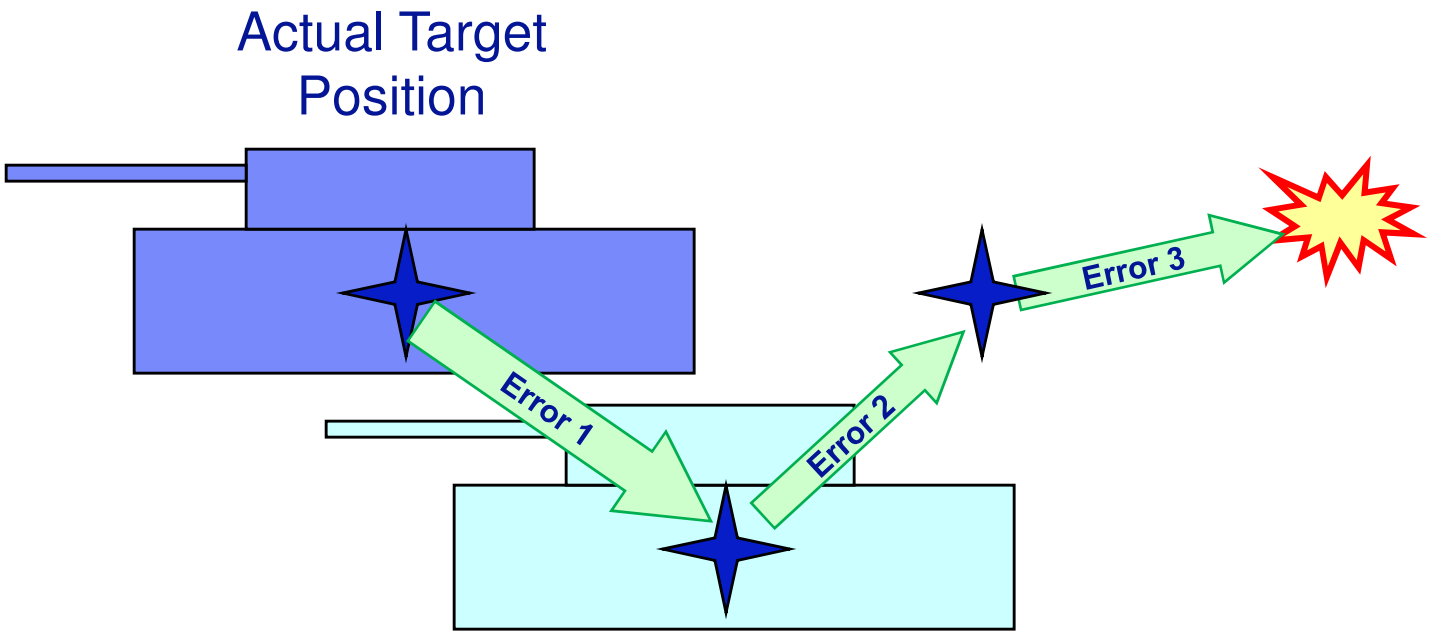
**Target – Background – Noise Ratio**

## Sensor and Perception

- Sensor is able to detect a certain property or combination thereof
- Target exposes at least one of the properties
- Background does not exposes the same characteristics regarding the property

- Searching and looking
- Detecting
- Tracking
- Classifying, recognizing, identifying
- Target acquisition
- Damage perception

# Some Influences on (Single) Shot Accuracy




Aim Point

Impact Point

Error 1: Wrong perception
Error 2: Poorly Aimed
Error 3: Ballistic Error

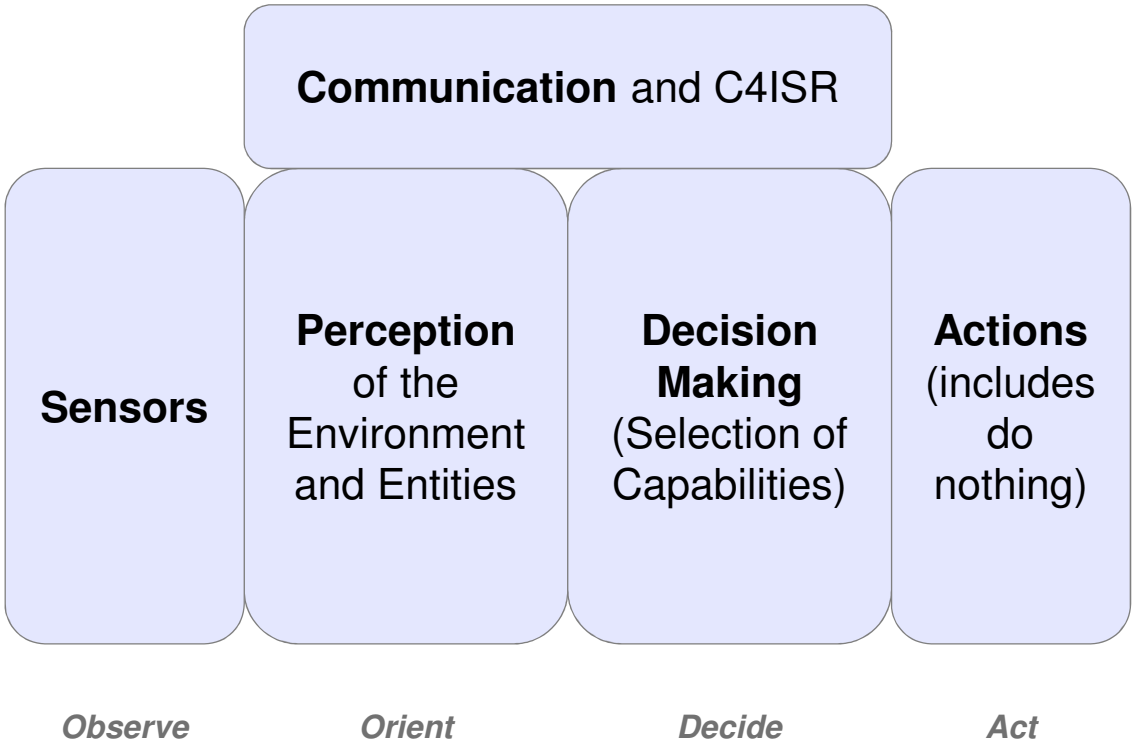
# Command and Control

- Explicit modeling of communications
  - Equipment, frequency, capacity, etc.
  - Modeling of jamming, malfunctions
- Explicit modeling of tactical messages
  - Integration of real C2 systems (sends and receives messages)
  - Modeling of intercepted or failed messages
- Explicit modeling of headquarters
  - Creating of a perception
  - All phases and contributions to the **Observe** – **Orient** – **Decide** – **Act** - Loop

# A Validation and Verification Template (S-P-C-D-A)

**Environment**

- Terrain
- Weather
- Objects
- Other Entities
  - *Own forces*
  - *Opposing and*
  - *Neutral forces*



# Tasks of the Simulation Engineer

- Selecting the best simulation systems
- Composing the selected system into a federation
- Exposing the information needed using the selected protocol
- Integrating provided information into the receiving simulation system
- Avoiding inconsistencies, anomalies, and unfair fight conditions
- Addressing multiple interoperability protocols (including live – virtual – constructive challenges)
- Ensuring consistent initialization of all simulation systems and other components
- Ensuring consistent and timely information exchange during execution

# Technical tasks

## Simulated Objects

- Export information of internally simulated objects (including mapping to protocol)
- Import information about externally simulated objects (including mapping from protocol to internal presentation)
- Representing all objects of interest within the simulation

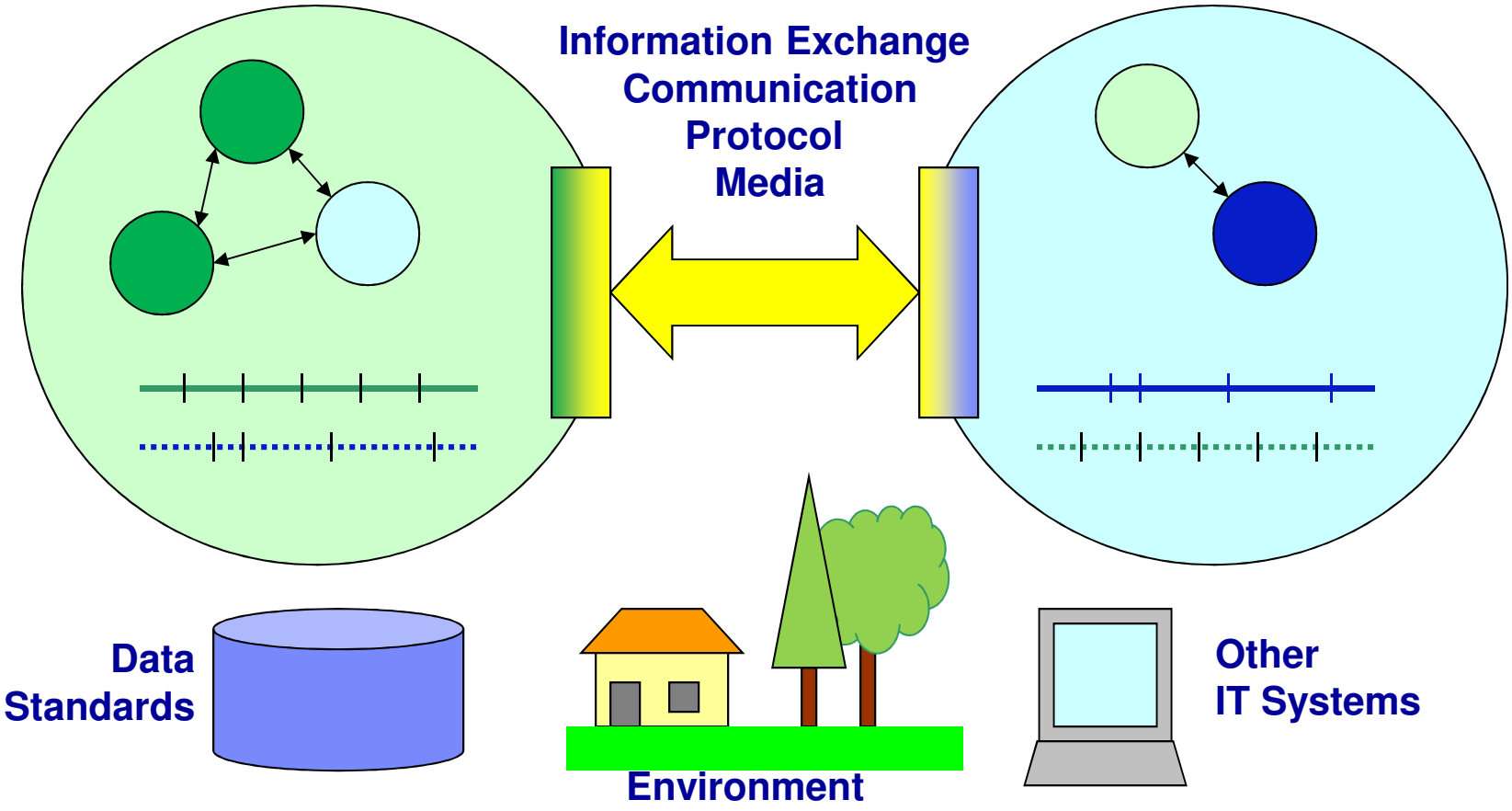
## Simulated Interactions

- Distinguish between own objects and objects owned by other simulations
- Representing all interactions (including cause-effects)
- Calculate and distribute interaction results concerning own objects
- Distribute interactions to other simulations

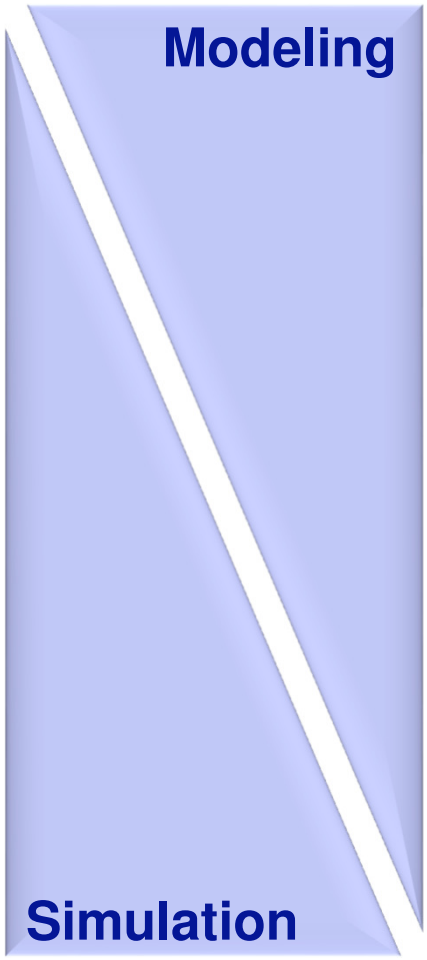
# Support by Infrastructure

- Infrastructure connects the Simulation Systems
  - Making sure that ***all needed information*** is exchanged between source and target
  - Making sure that ***only the needed information*** is exchanged between source and target
  - Making sure that information is delivered ***in time***

# Overview of Requirements



# The M&S Spectrum




Modeling is the purposeful simplification and abstraction of a perception of reality that is shaped by physical and cognitive constraints …

*… leading to a conceptualization of the relevant subset of the problem domain*

Computational and simulation modeling require many implementation choices:

*mathematical modeling, discretization and algorithm selection, computer programming, and numerical solutions*

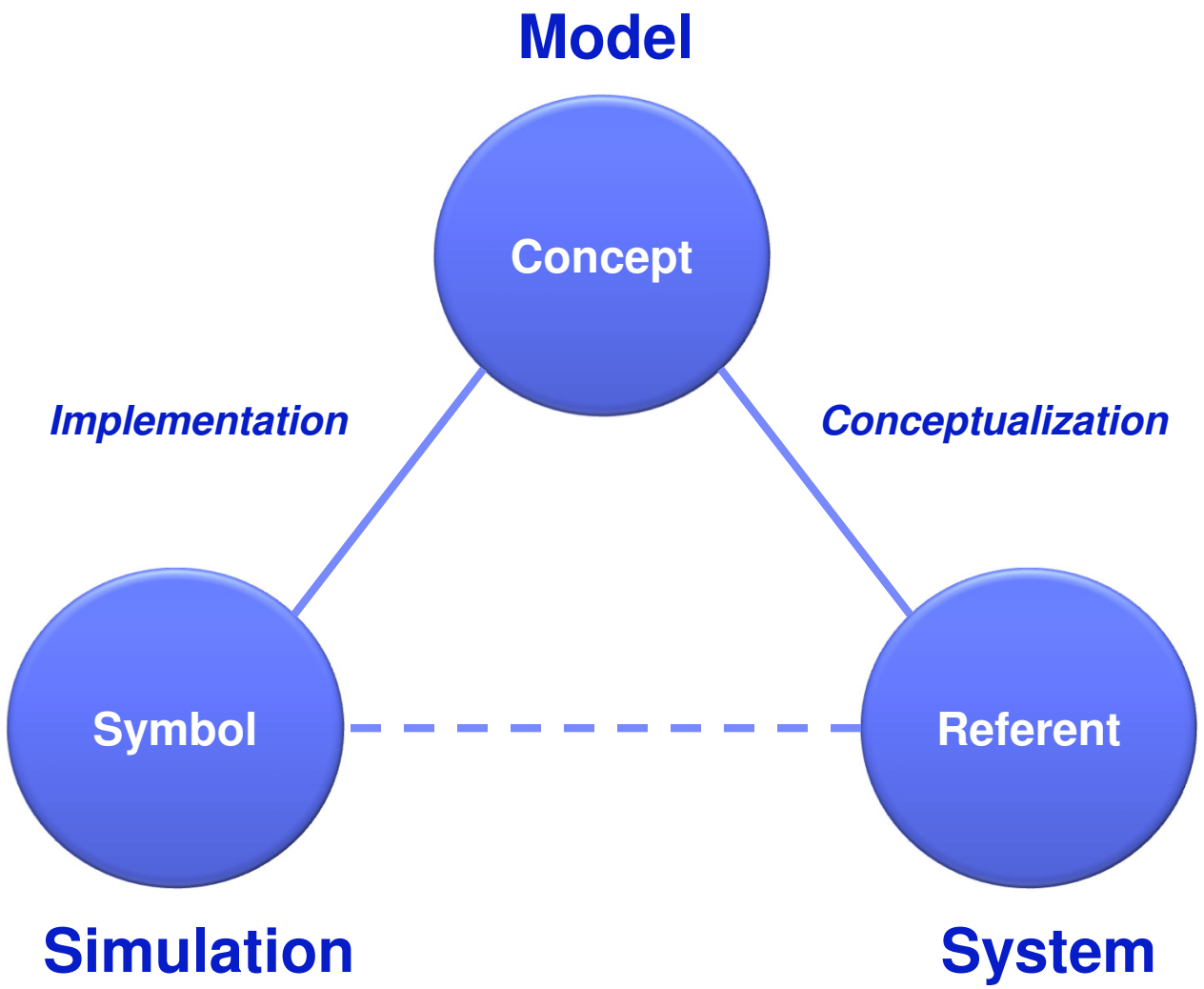
**Real World Reference**

**Abstract Model**

**Computational Model**

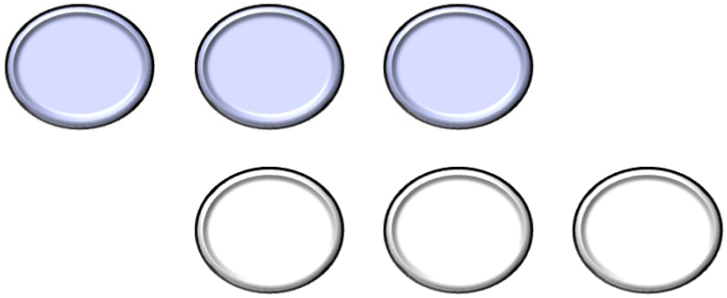
**Software Model**

# Modeling and Simulation



# Data/Information Misalignment

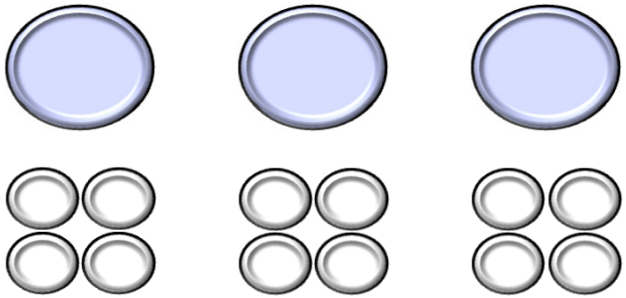
## Multi-Scope



System A

System B

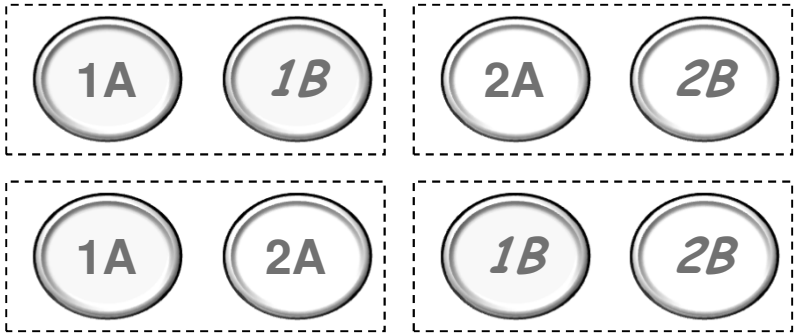
## Multi-Resolution



System A

System B

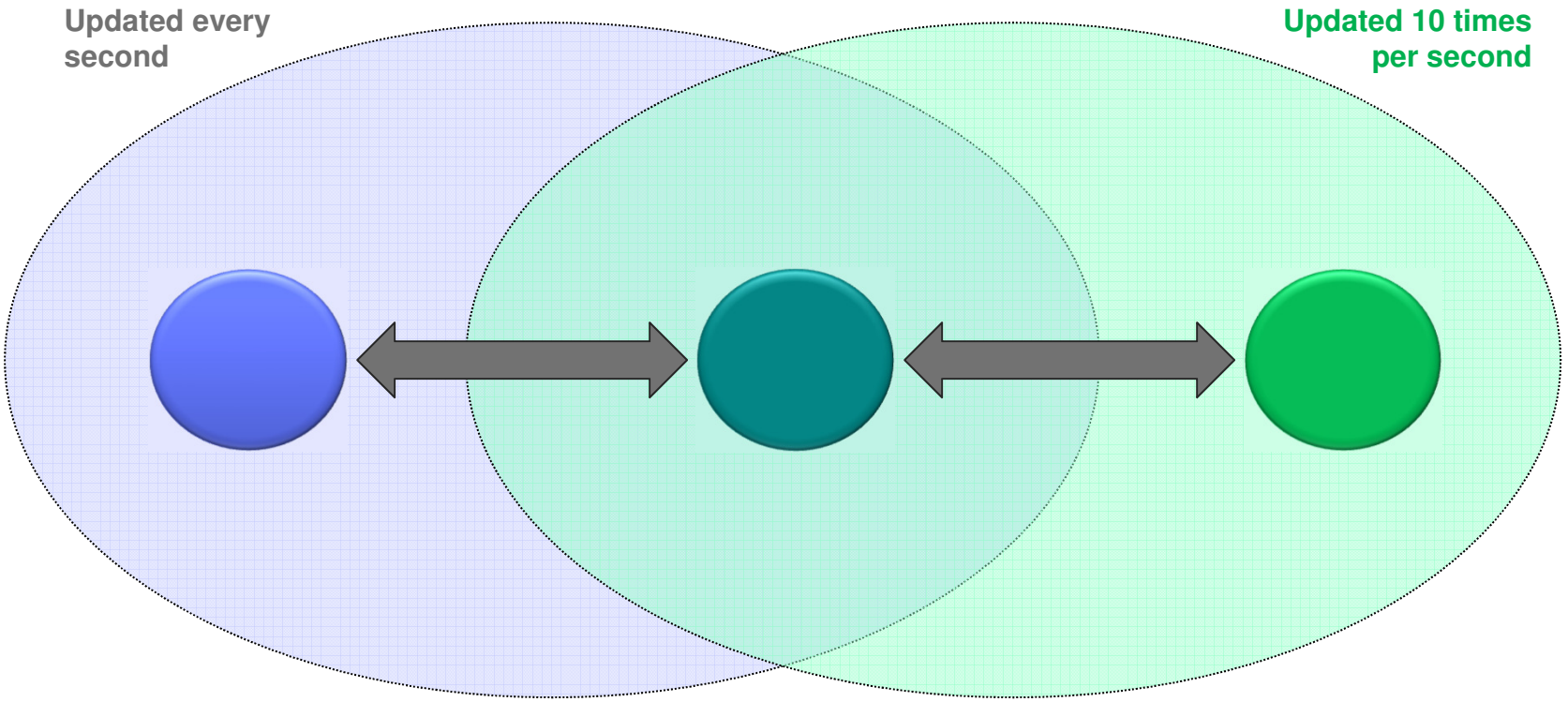
## Multi-Structure



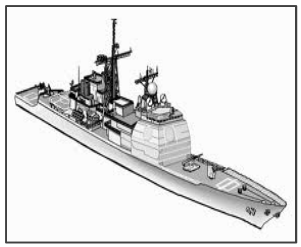

System A

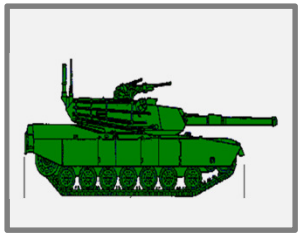
System B

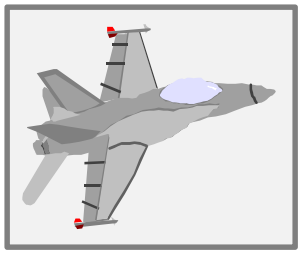
# Temporal Inconsistencies



# Time Anomaly







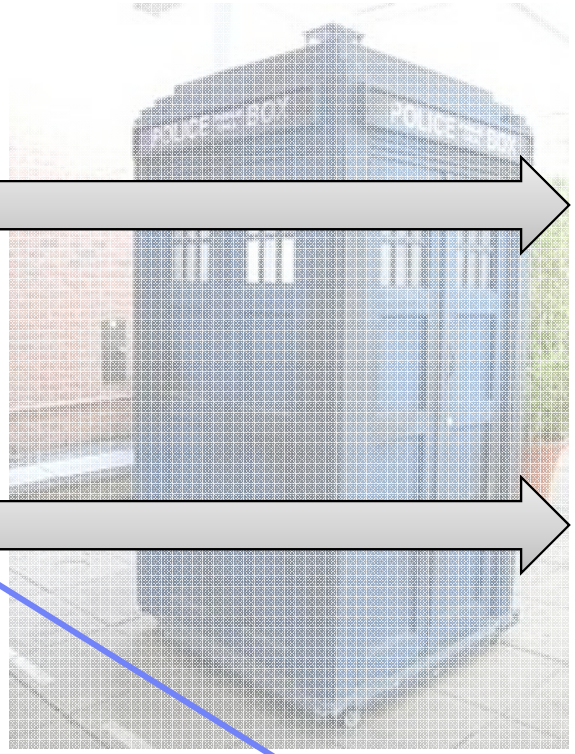
1 Navy Guns Fire

2b Tanks explode

Tanks get Hit 2a

UAV observes exploding Tanks 3a

UAV observes Navy firing guns 3b

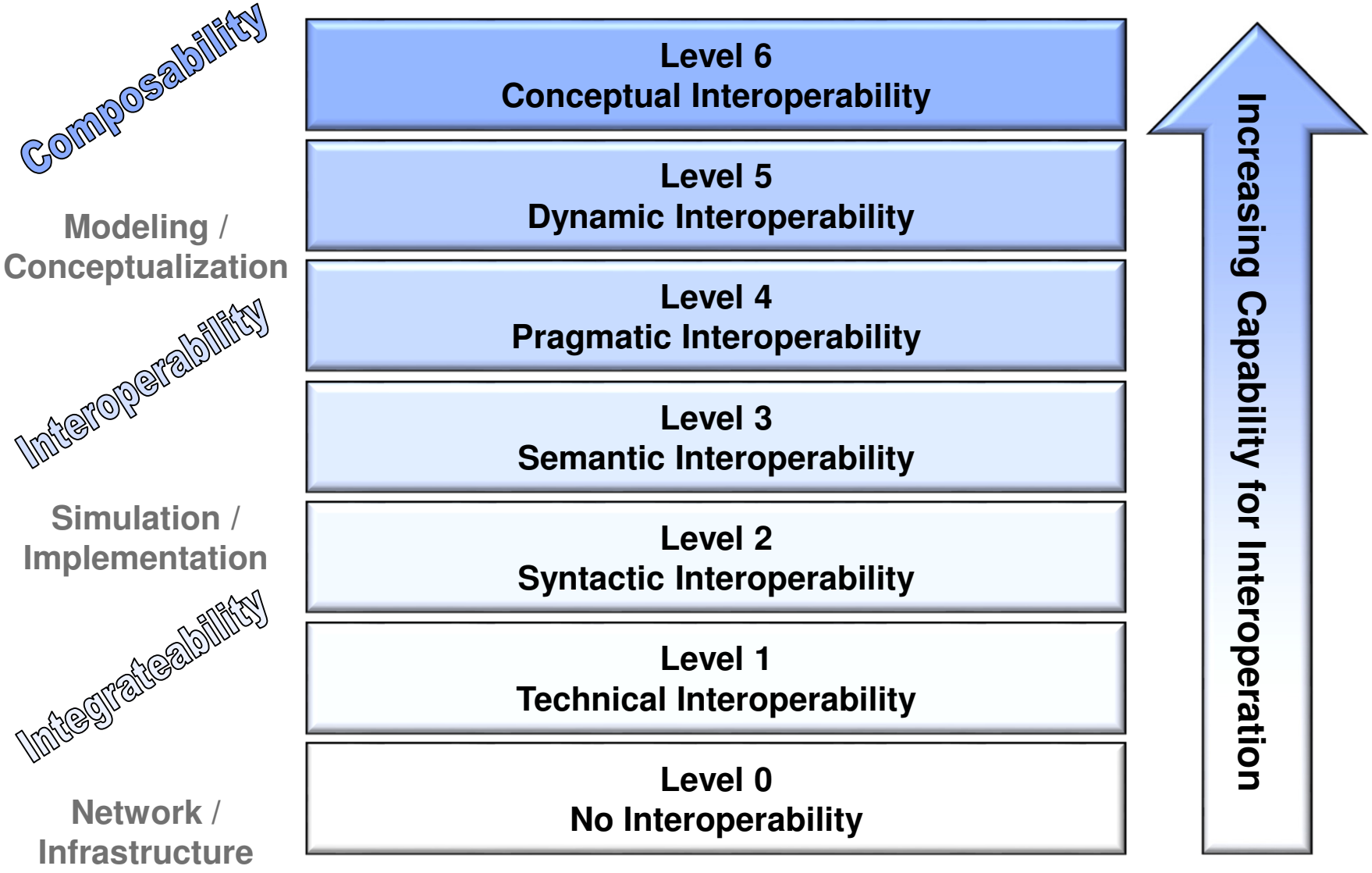


# Integrateability, Interoperability, & Composability

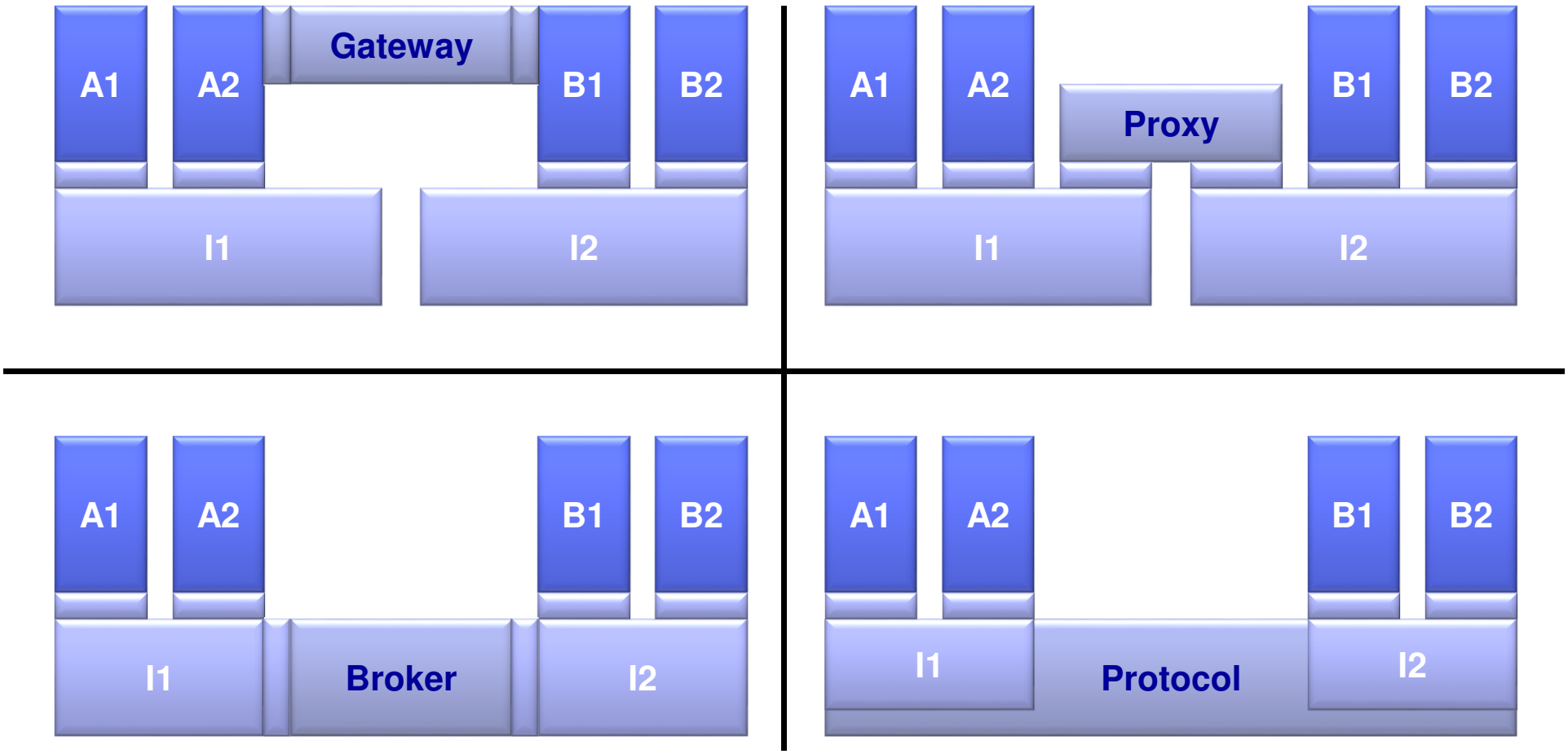
- *Integrateability* contends with the physical/technical realms of connections between systems, which include hardware and firmware, protocols, networks, etc.
- *Interoperability* contends with the software and implementation details of interoperations; this includes exchange of data elements via interfaces, the use of middleware, mapping to common information exchange models, etc.
- *Composability* contends with the alignment of issues on the modeling level. The underlying models are purposeful abstractions of reality used for the conceptualization being implemented by the resulting systems.

In summary, ***successful interoperation of solutions requires integratability of infrastructures, interoperability of systems, and composability of models*.** Successful standards for interoperable solutions must address all three categories.

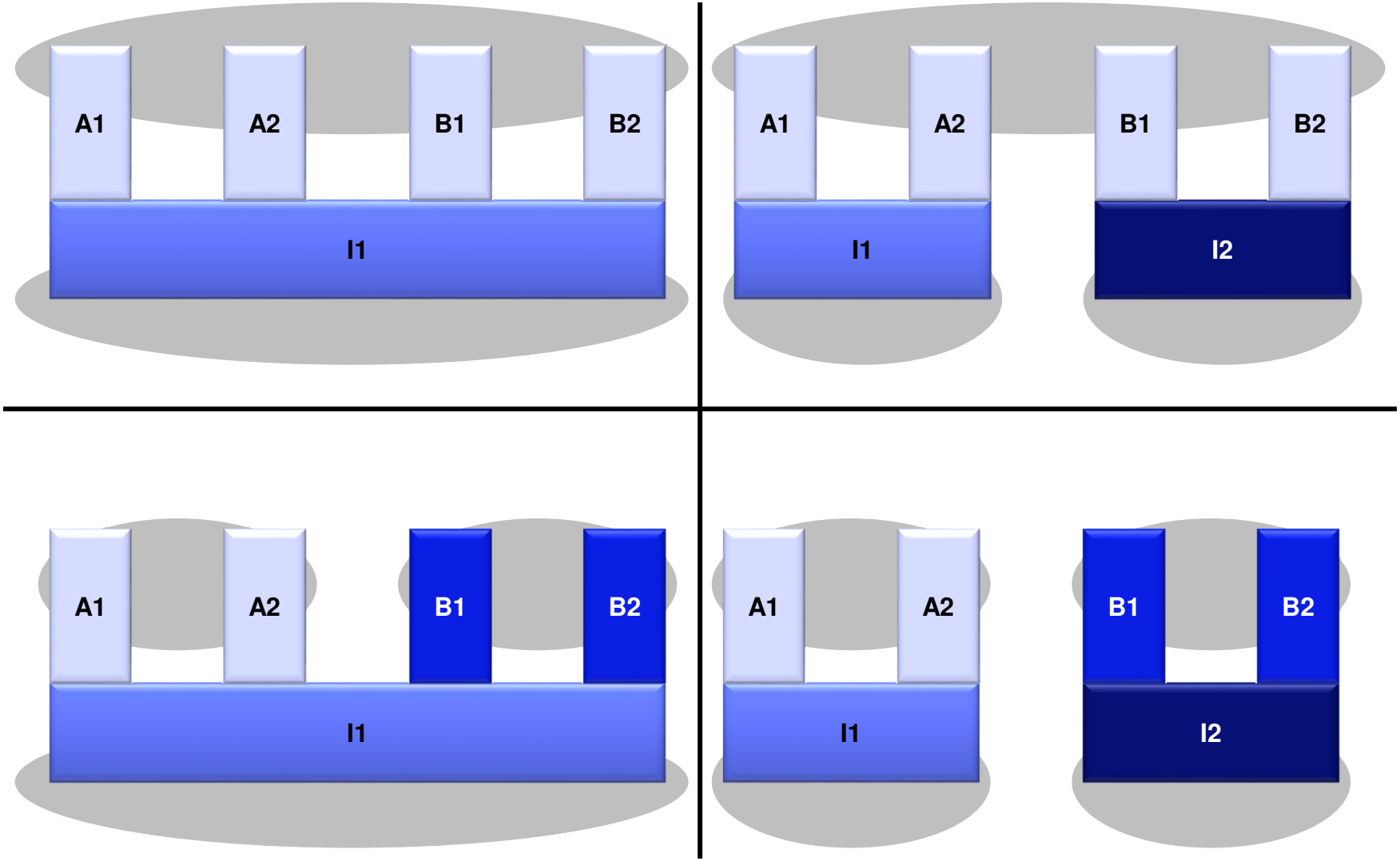
# Levels of Conceptual Interoperability Model



# Gateway, Proxy, Broker, and Protocol



# Information Exchange / Infrastructure



# Summary General Challenges

- Alignment of Entities
  - Multi-scope, multi-resolution, and multi-structure issues
  - Synonyms and homonyms, different namespaces
- Harmonization of Processes and Activities
  - Time inconsistencies and anomalies
  - Different event-queues
  - Latencies
- Supporting infrastructures
  - Gateways, proxies, brokers, and protocols

**Integratability** ensures the proper exchange of information.
**Interoperability** ensures the mediation of data into usable information.
**Composability** ensures assumptions and constraints are met.

Part Two

# HIGH LEVEL ARCHITECTURE

# Common Technical Framework

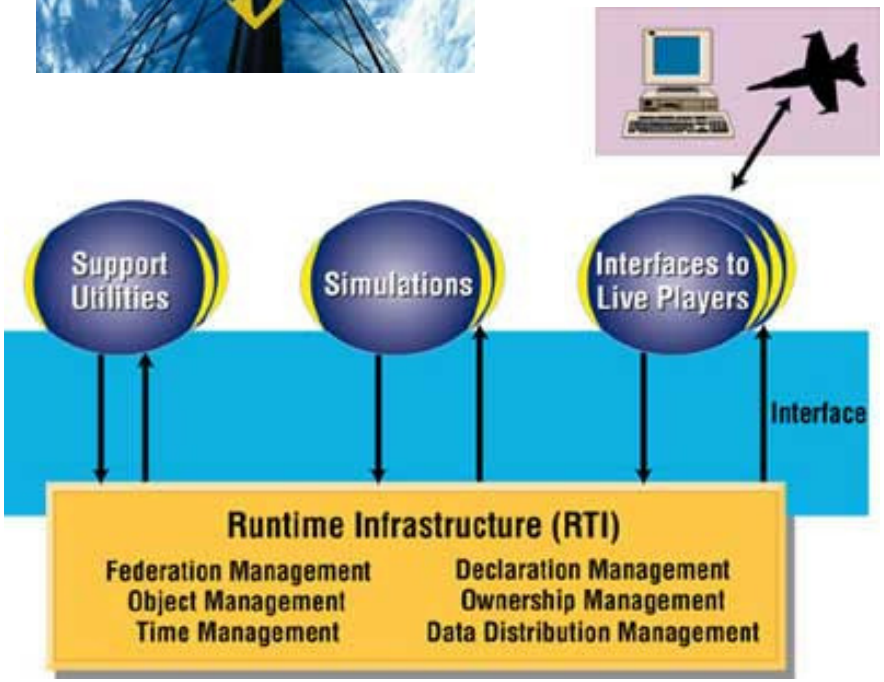
- Common Understanding of the Mission Spaces
  - Conceptual Model of the Mission Space (CMMS)
  - Functional Description of the Mission Space (FDMS)
- Common Data Standards
  - Information Exchange Requirements
  - Synthetic Environment Data Representation and Interchange Specification (SEDRIS)
  - Joint Consultation, Command and Control Information Exchange Data Model (JC3IEDM) / Command and Control (C2) Core / National Information Exchange Model (NIEM)
  - …
- High Level Architecture

# High Level Architecture (HLA)

- HLA Rules
  - Interaction of federates in a federation
  - Responsibilities of federates
- Interface Specification
  - RTI services and interfaces
  - Interfaces to be provided by the federate
- Object Model Template
  - Syntax for information exchange
  - Definition of the Key Models
- Engineering and Execution Process
  - Guideline/Best Practice on how to build a federation
- Validation, Verification & Accreditation
  - Guideline/Best Practice on how to show the federation is correct and applicable

# HLA Principles

- Simulations are federates within a federation
- All information is exchanged via the Runtime Infrastructure (RTI)
  - Information Exchange is specified in the Federation Object Model (FOM)
  - Interface between RTI and federate (application program interface) is standardized
- Each object in the FOM is only controlled by one federate at a time and updated by the others
- RTI provides services to orchestrate the overall execution and consistency

# HLA is not HLA

- There are three major High Level Architecture lines (and several sub-lines)
  - **HLA 1.3 NG**
    - This is the final HLA version provided by the US Department of Defense for free. This version was submitted to IEEE for standardization. Many US organizations are still using this version.
  - **HLA 1516-2000**
    - IEEE made several changes to the submitted proposal, such as replacing the BNF documents with XML and generalizing hard-coded solutions into configuration driven solutions. This version was accepted by NATO and has been implemented in many European systems.
  - **HLA 1516-2010**
    - IEEE updates their standards every ten years. The latest version supports more web technology as well as modular and reconfigurable federation object models. There are reference implementation and declared intention to upgrade from NG as well as 1516-2000 users.

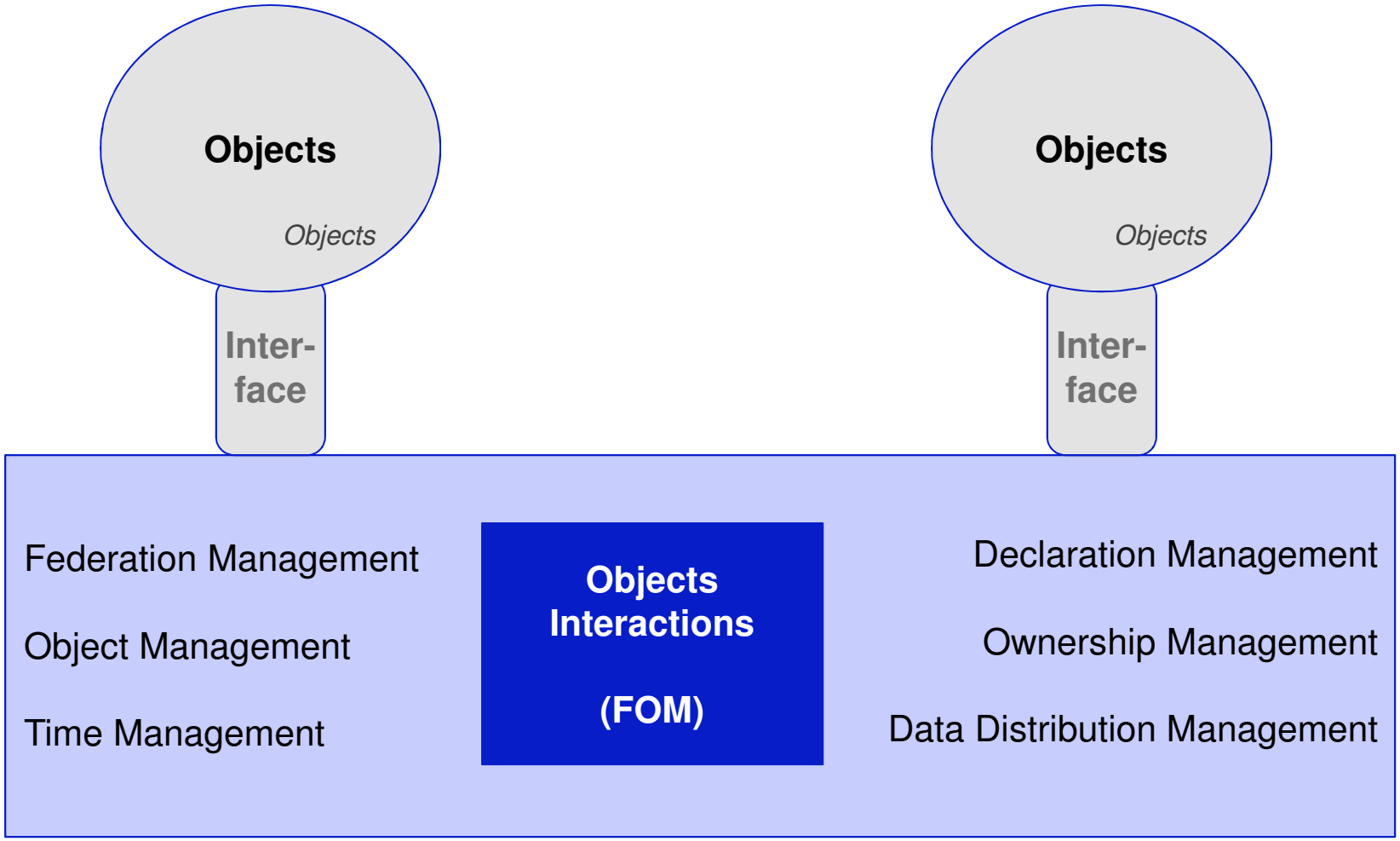
# IEEE 1516 HLA

- The IEEE 1516 standard consists of five parts that were released and updated in the following documents.
  - IEEE 1516-2010 - Standard for Modeling and Simulation High Level Architecture - **Framework and Rules**
  - IEEE 1516.1-2010 - Standard for Modeling and Simulation High Level Architecture - **Federate Interface Specification**
  - IEEE 1516.2-2010 - Standard for Modeling and Simulation High Level Architecture - **Object Model Template (OMT) Specification**
  - ~~IEEE 1516.3-2003 - Recommended Practice for High Level Architecture **Federation Development and Execution Process (FEDEP)**~~
  - IEEE 1516.4-2007 - Recommended Practice for **Verification, Validation, and Accreditation** of a Federation an Overlay to the High Level Architecture Federation Development and Execution Process
- In addition, IEEE 1516.3-2003 will not be renewed but replaced by
  - IEEE 1730-2010 - IEEE Recommended Practice for **Distributed Simulation Engineering and Execution Process (DSEEP)**
  - IEEE 1730.1-2013 – IEEE Recommended Practice for Distributed Simulation Engineering and Execution Process **Multi-Architecture Overlay (DMAO)**

# HLA Glossary

- **Federation:** Group of several simulation systems executed together to support a common objective
- **Federate:** Simulation system within a federation
- **Runtime Infrastructure (RTI):** Software bus providing the services required by the HLA infrastructure
- **Federation Object Model (FOM):** Information exchange specification in OMT for a given federation
- **Simulation Object Model (SOM):** General information exchange capability of a given simulation specified in OMT
- **Management Object Model (MOM):** Standardized information needed for the management of a federation specified in OMT

# Conceptual View



# Federation Rules

1. Federations shall have a FOM, documented in accordance with the OMT.
2. All representation of objects in the FOM shall be in the federates, not in the RTI.
3. During a federation execution, all exchange of FOM data among federates shall occur via the RTI.
4. During a federation execution, federates shall interact with the RTI in accordance with the HLA interface specification.
5. During a federation execution, an attribute of an instance of an object shall be owned by only one federate at any given time.

# Federate Rules

6. Federates shall have a SOM, documented in accordance with the OMT.
7. Federates shall be able to update and/or reflect any attributes of objects in their SOM, and send and/or receive SOM interactions externally, as specified in their SOM.
8. Federates shall be able to transfer and/or accept ownership of attributes dynamically during a federation execution, as specified in their SOM.
9. Federates shall be able to vary the conditions under which they provide updates of attributes of objects, as specified in their SOM.
10. Federates shall be able to manage local time in a way which will allow them to coordinate data exchange with other members of a federation.

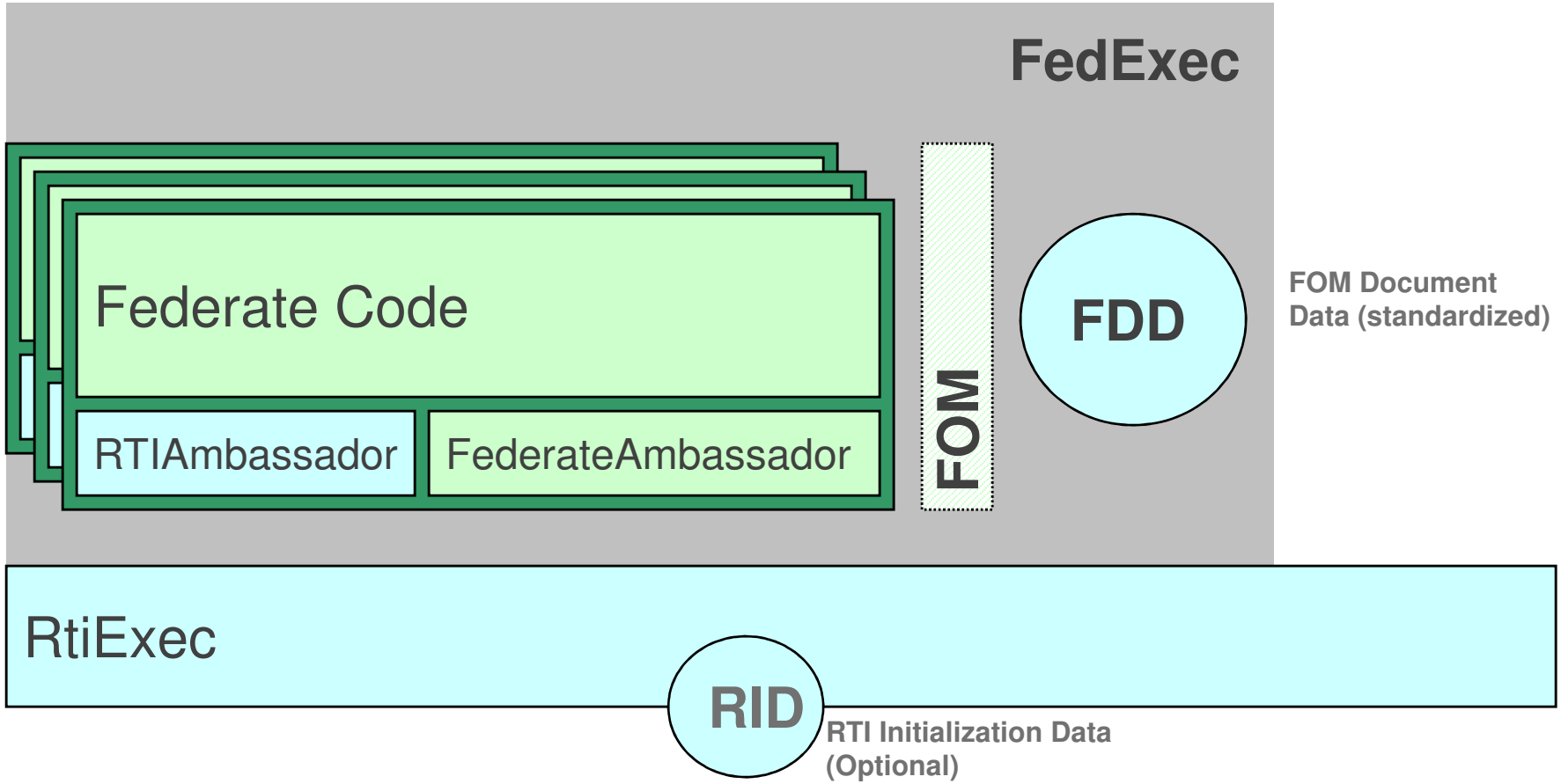
# Components of a Standard HLA Federation

- FedExec
  - Federation Executive
  - One FedExec per running federation
  - Manages the federation
- RtiExec
  - The Runtime Infrastructure Executive
  - One RtiExec can support several FedExec
- libRTI
  - Comprises the API of the services for access to the federates
- Federates
  - The simulation functionality

# The Ambassador Concept

- RTIambassador
  - Included in the libRTI
  - Provides the RTI functionality
  - Implemented by RtiExec
- RTI_FederateAmbassador
  - Included in the libRTI
  - Abstract class with virtual functions
  - Federate derives own class and implement the necessary functions

# Components of an HLA Federation



# What to know about exchanging data

- Objects
  - Persistent object
  - Have to be created
  - Have to be destroyed
- Interactions
  - Transient objects
  - Have to be created
- Publish
  - Create the objects
  - Populates the objects/attributes
  - Updates the objects/attributes
- Subscribe
  - Discovers the objects
  - Reflects the objects/attributes

# Creating the OMT Files for 1516

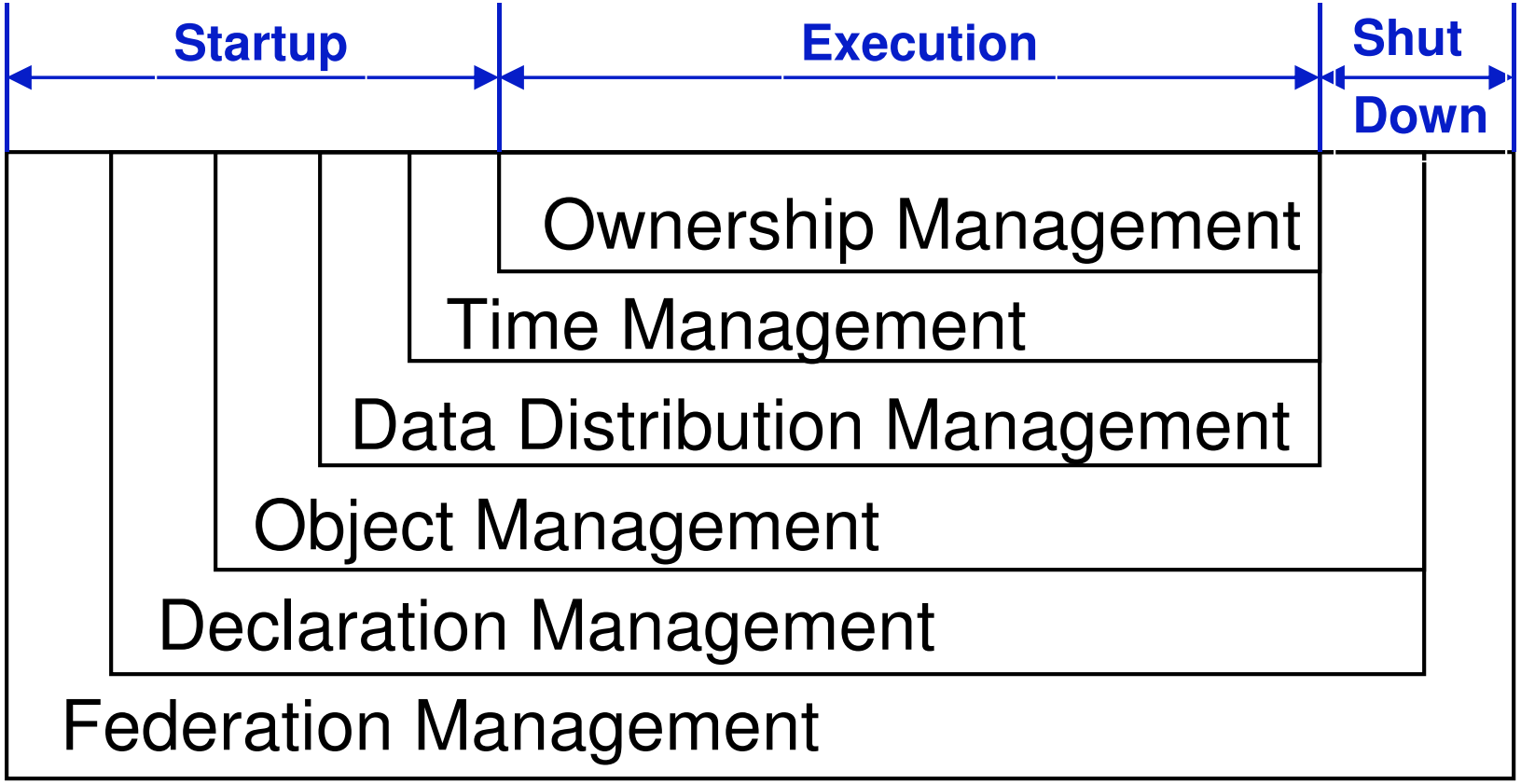
1516 OMT consists of 19 tables

1. **Object Model Identification Table**
2. **Object Class Structure Table**
3. **Interaction Class Structure Table**
4. **Attribute Table**
5. **Parameter Table**
6. Dimension Table
7. Time Representation Table
8. User-supplied Tag Table
9. Synchronization Table
10. Transportation Type Table
11. **Switches Table**
12. **Notes Table**
13. **Basic Data Representation Table**
14. **Simple Datatype Table**
15. **Enumerated Datatype Table**
16. **Fixed Record Datatype Table**
17. **Array Datatype Table**
18. **Variant Record Datatype Table**
19. **FOM/SOM Lexicon**

# The functions of the RTI

- The OMT describes how the data needs to be specified
- The RTI defines six categories of functions providing the services
  - Functions to be called and parameters are defined in the RTIAmbassador
  - Callback functions and parameters are defined in the FederateAmbassador
  - The interplay of functions and callbacks is defined in detail in the IEEE standard
- Typical lifecycle
  - Federate joins the federation
  - Federate declares what object types and interaction types he can provide
  - Federate declares what object types and interaction types he is interested in
  - Federate sets constraints (data distribution)
  - Federate publishes objects and interactions
  - Federate receives objects and interactions
  - Federate leaves the federation

# The RTI Services



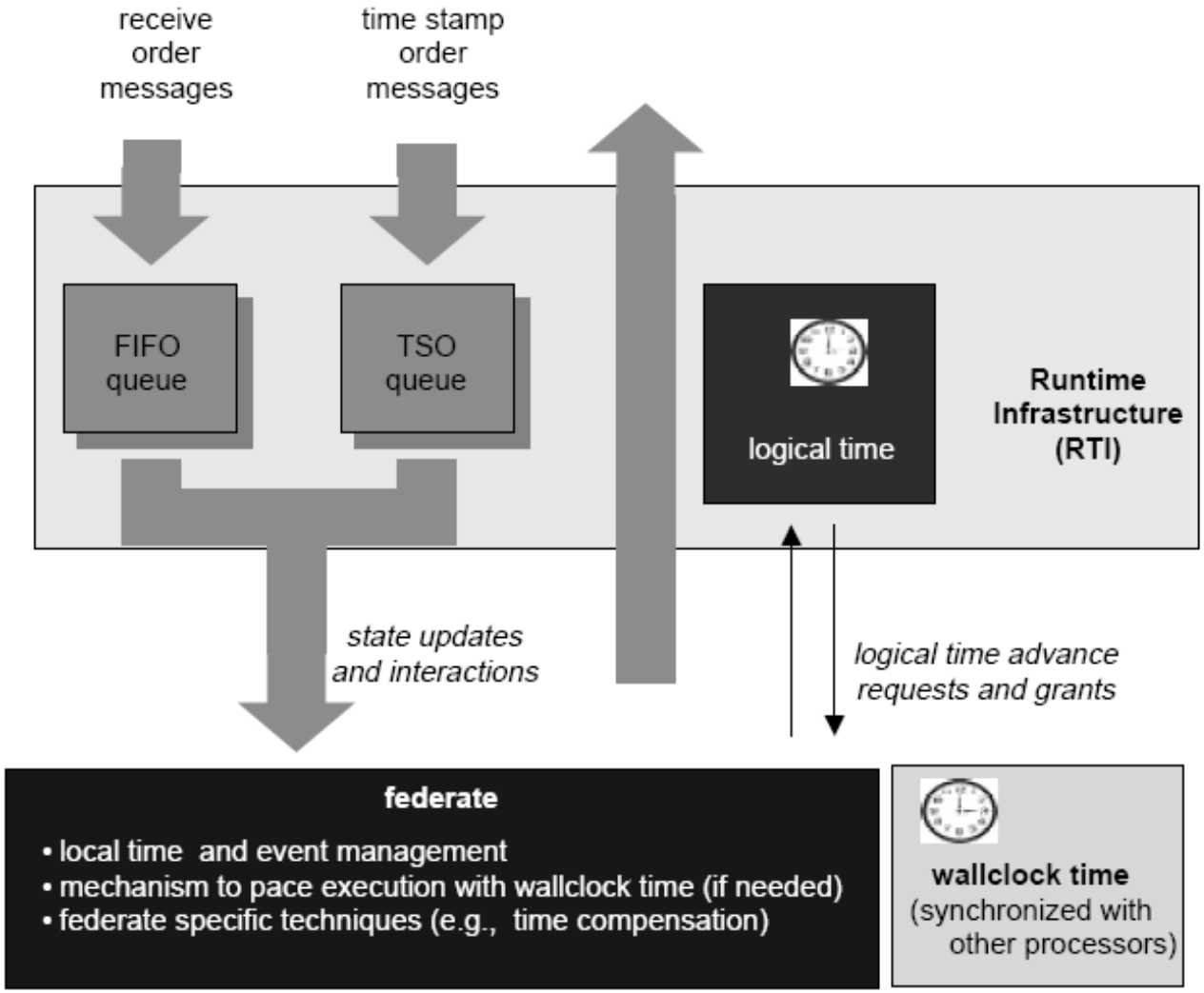
# Two Types of Message Ordering

- Time Stamped Order (TSO)
  - Each message receives a time stamp from the sending system
  - This is the internal simulated time of the sender
  - Messages are delivered to the receiver in order of this time stamp
  - RTI guarantees that no message will be received from the past
- Receive Order (RO)
  - Message will be delivered in the order received by the RTO
  - First in, first out (FIFO) principle
- This ordering types are not mutual exclusive

# Comparison of RO and TSO

| | Receive Order (RO) | Time-stamped Order (TSO) |
|---|---|---|
| **Latency** | Low | Higher |
| **Reproduce temporal order of messages** | No | Yes |
| **All federates see same order of events** | No | Yes |
| **Execution repeatable** | No | Yes |
| **Typical application domain** | Training, T&E, Real Time | Analysis, Experimentation |

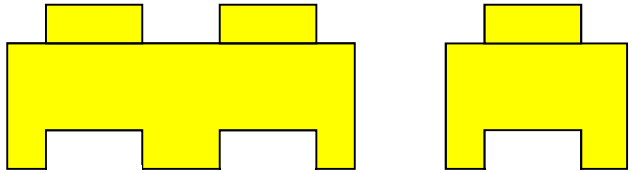
# Time Management



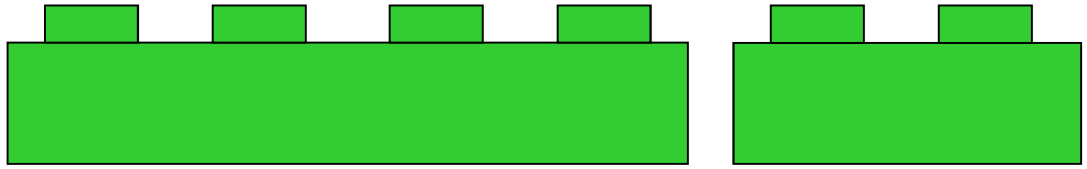
# Summary RTI Services

- The RTI functions and callbacks are standardized
  - There are many RTI implementations
  - Not all implementations support all functions
- The six functions categories provide all services needed
  - Federation management
  - Declaration management
  - Object management
  - Ownership management
  - Time management
  - Data distribution management
- Main improvements of IEEE 1516-2010
  - Modular FOM (can be reloaded and changed during runtime)
  - Fault tolerance (crash of federates)
  - Enhanced support of web-services and XML based information exchange

# Modular FOM Block Types



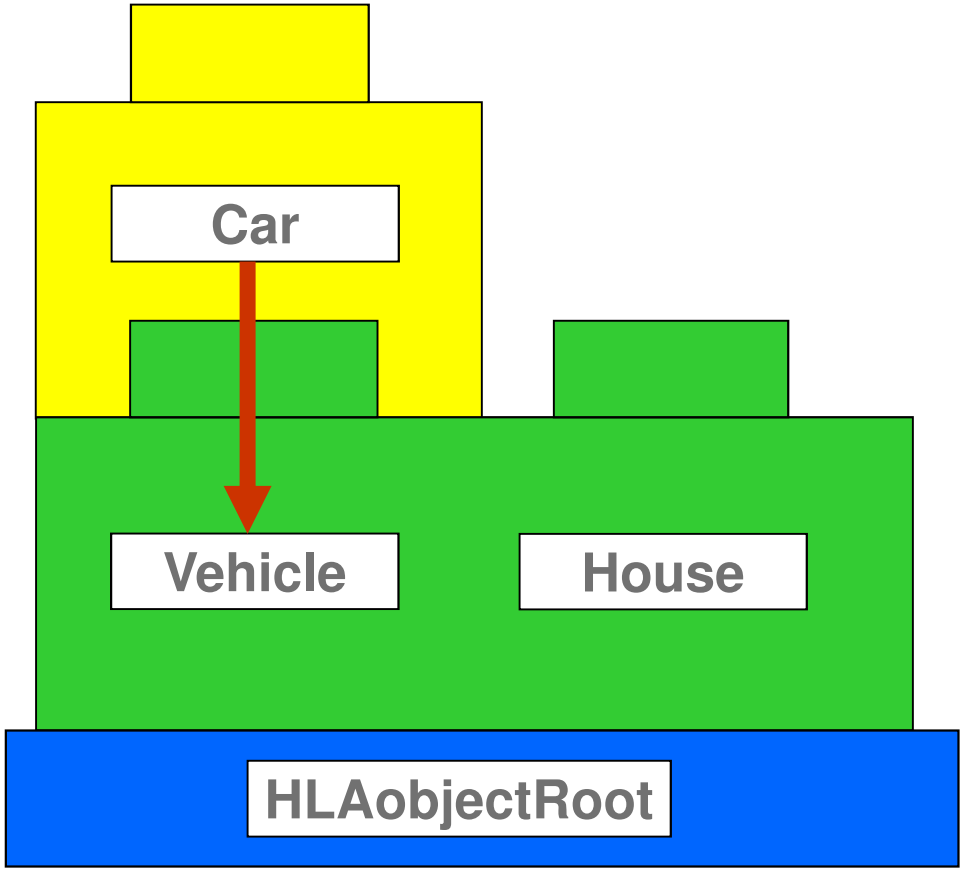
Dependent FOM Modules
These module reference concepts in other modules and are thus dependent.



Standalone FOM Modules
These modules can stand on their own and only depend on predefined HLA concepts.

The MOM Module
Contains MOM and predefined concepts. This is the platform that everything else is built upon.

# Whats In a Dependency?



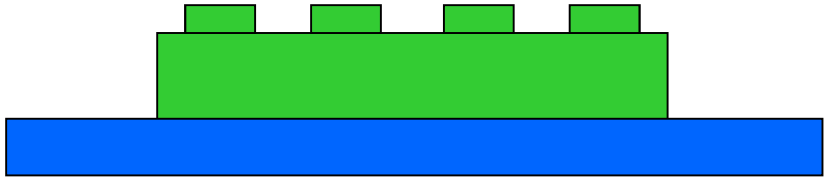

A Dependent FOM module contains references to FOM data in another module. The following FOM data can be referenced:

- Object class definition
- Interaction class definition
- Data type definition
- Dimension definition
- Update rate definition
- Transportation type definition
- Note

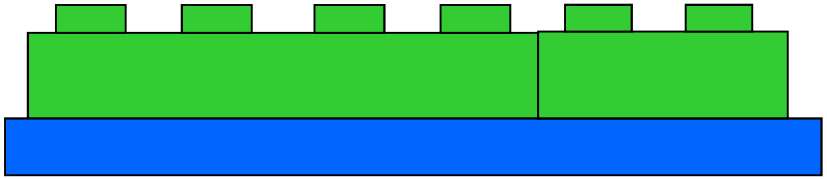
A Standalone FOM may only reference concepts in the MOM module.

# Allowed Combinations

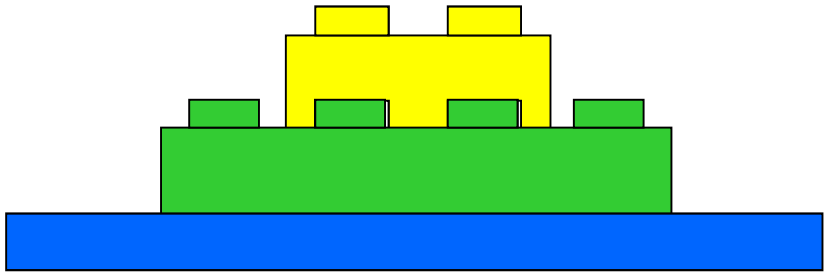
1. A MOM Module is always required.
2. A Standalone module and a MOM Module is sufficient for a valid FOM.



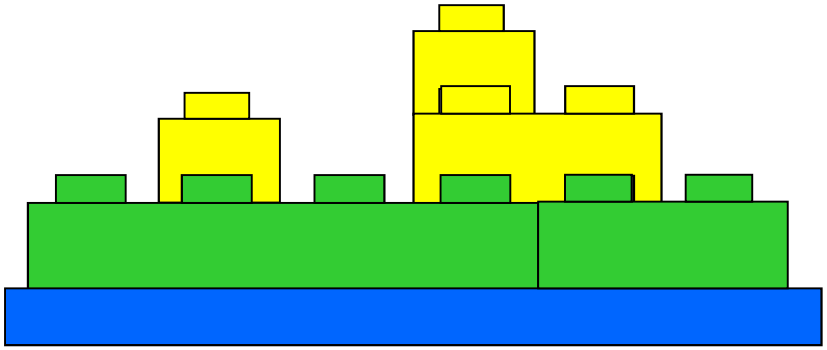
3. Several Standalone modules and a MOM Module are also allowed.



4. Dependent FOM Modules may be used on top of a standalone FOM.

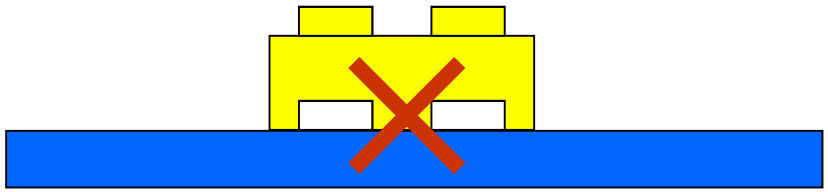


5.Several dependent FOMs may build upon a standalone FOM.
6. A dependent FOM may build upon several standalone FOMs.
7. A dependent FOM may build upon a dependent FOM (but there has to be a standalone FOM in the bottom)

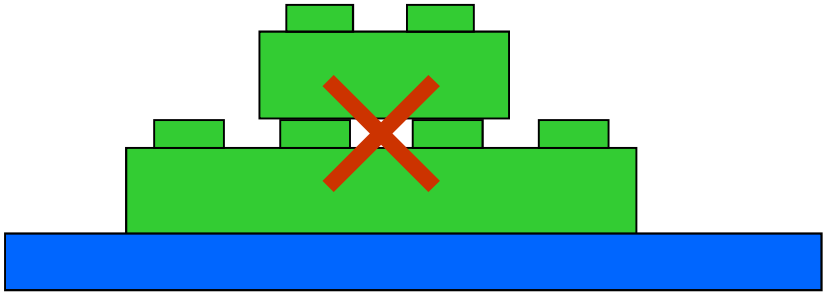


# Disallowed Combinations

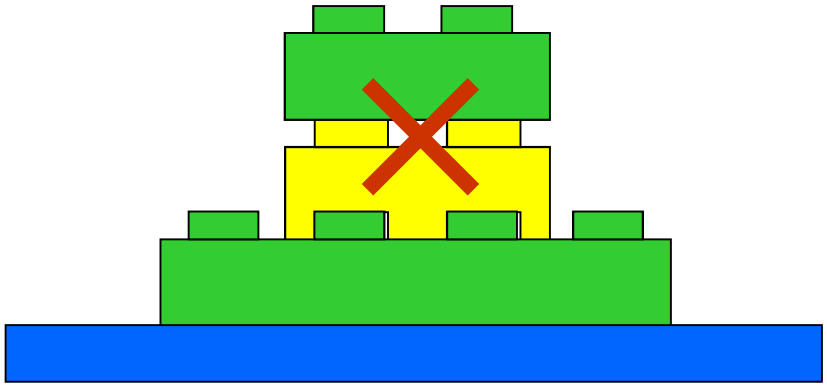
8. A Dependent FOM Module may not be used without a Standalone FOM module.



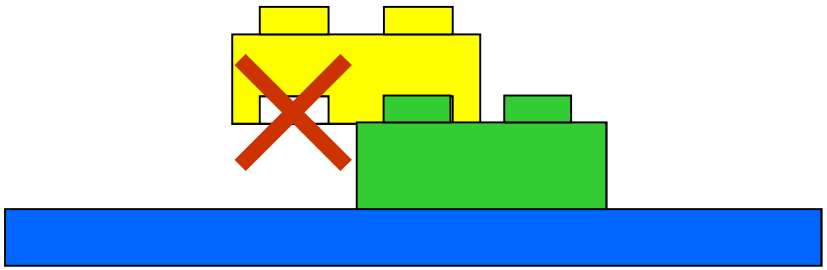
9. A Standalone FOM Module may not build upon another Standalone FOM module.

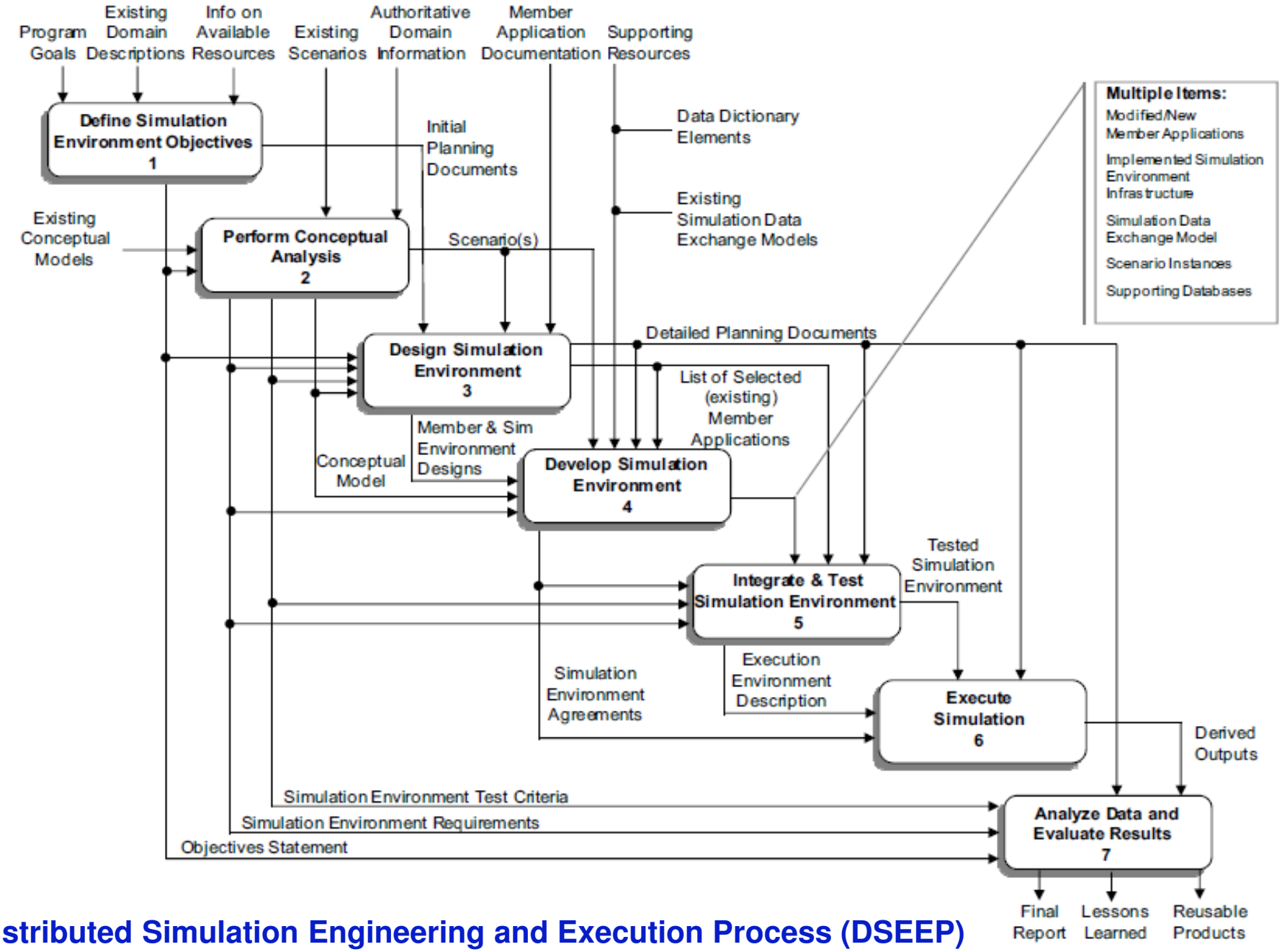


10. A Standalone FOM Module may not build upon a Dependent FOM module.



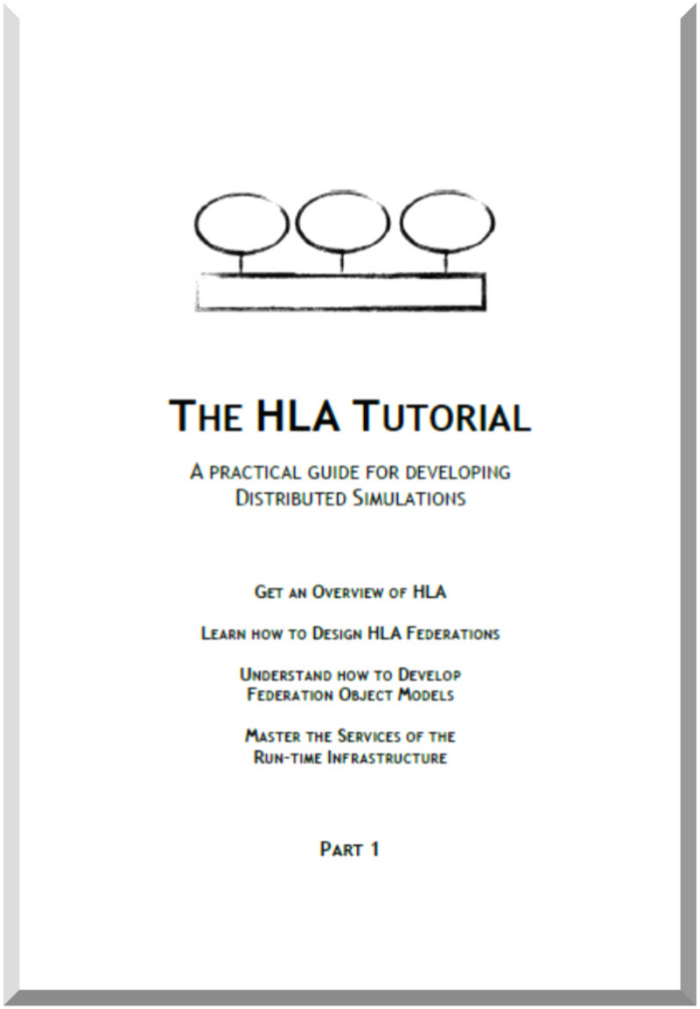
11. Every FOM concept that is referenced in a Dependent FOM Module must be provided by another (Dependent or Standalone) FOM Module.






**Distributed Simulation Engineering and Execution Process (DSEEP)**

# Free material




- The HLA Tutorial
  - http://www.pitch.se/hlatutorial

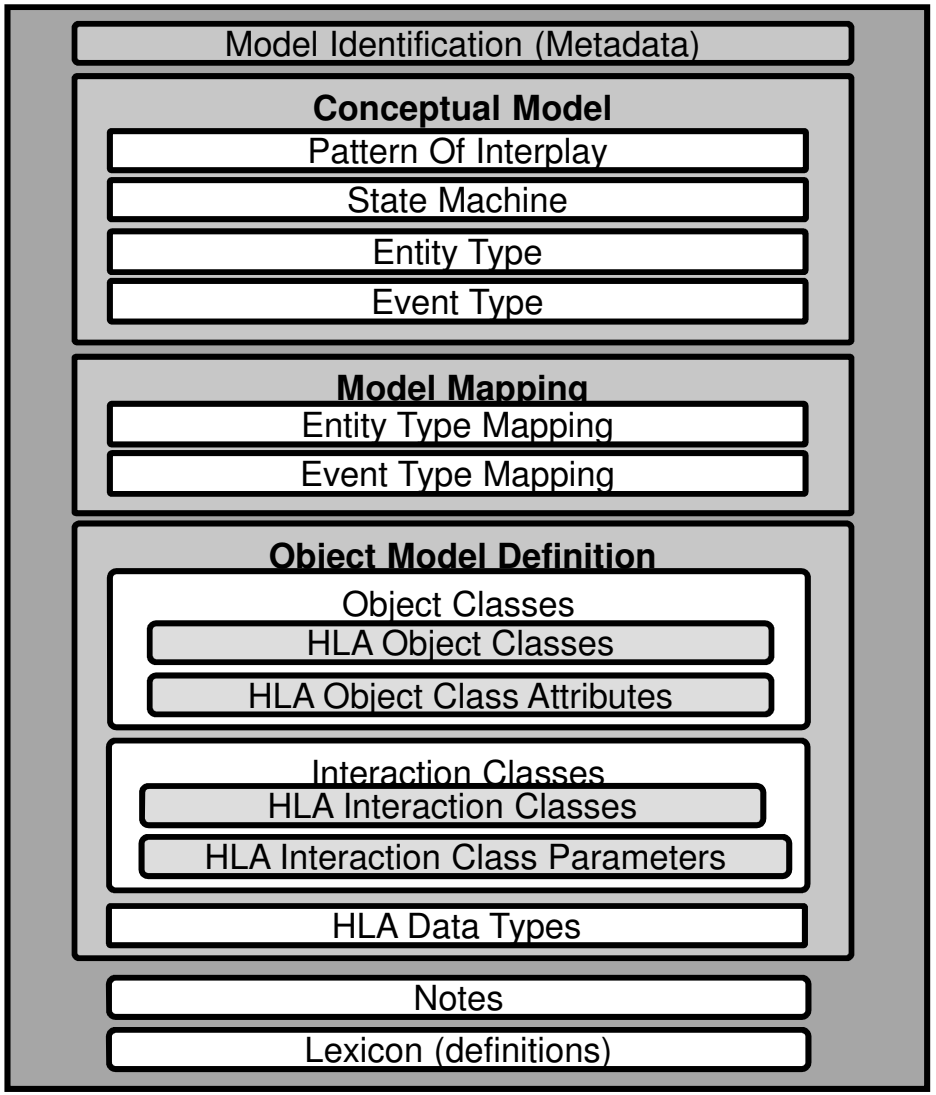
  - HLA Tutorial
  - HLA Evolved Starter Kit
  - Pitch pRTI
  - Pitch Visual OMT

Part Three

# SOME ALTERNATIVES AND EXTENSIONS

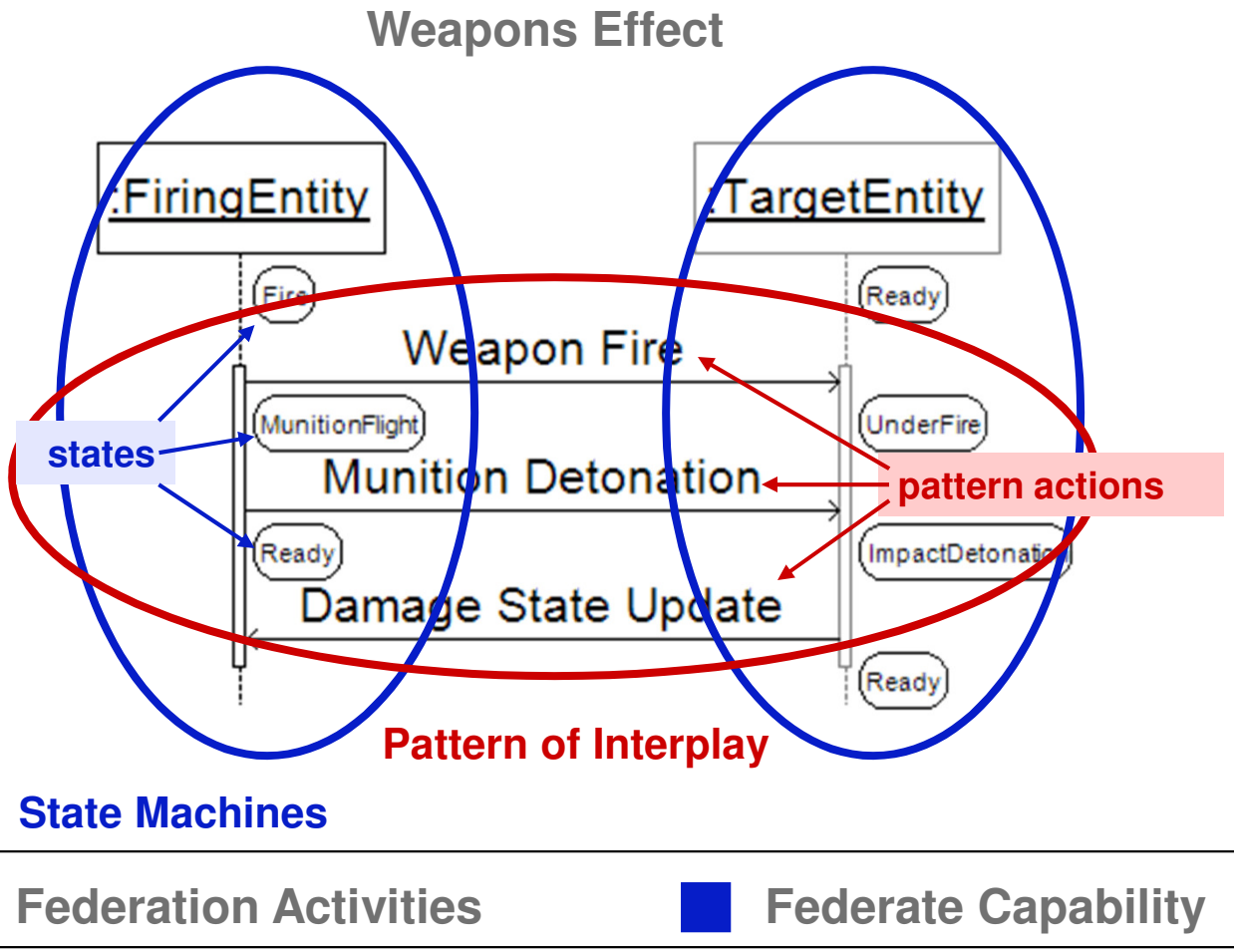
# Base Object Models

- Standardized
  - SISO-STD-003-2006: Base Object Model (BOM) Template Specification
  - SISO-STD-003.1-2006: Guide for BOM Use and Implementation
- Basic ideas
  - Utilize UML standard to better describe federates
    - Conceptual level support
    - Federation level support
    - Federate level support
  - Add behavior to the interface (replace black box with grey box)
  - Define patterns of interplay
  - Not HLA dependent, but defines a general approach



# BOM Illustration

*Derived from RPR FOM*



# IEEE 1278 Distributed Interactive Simulation

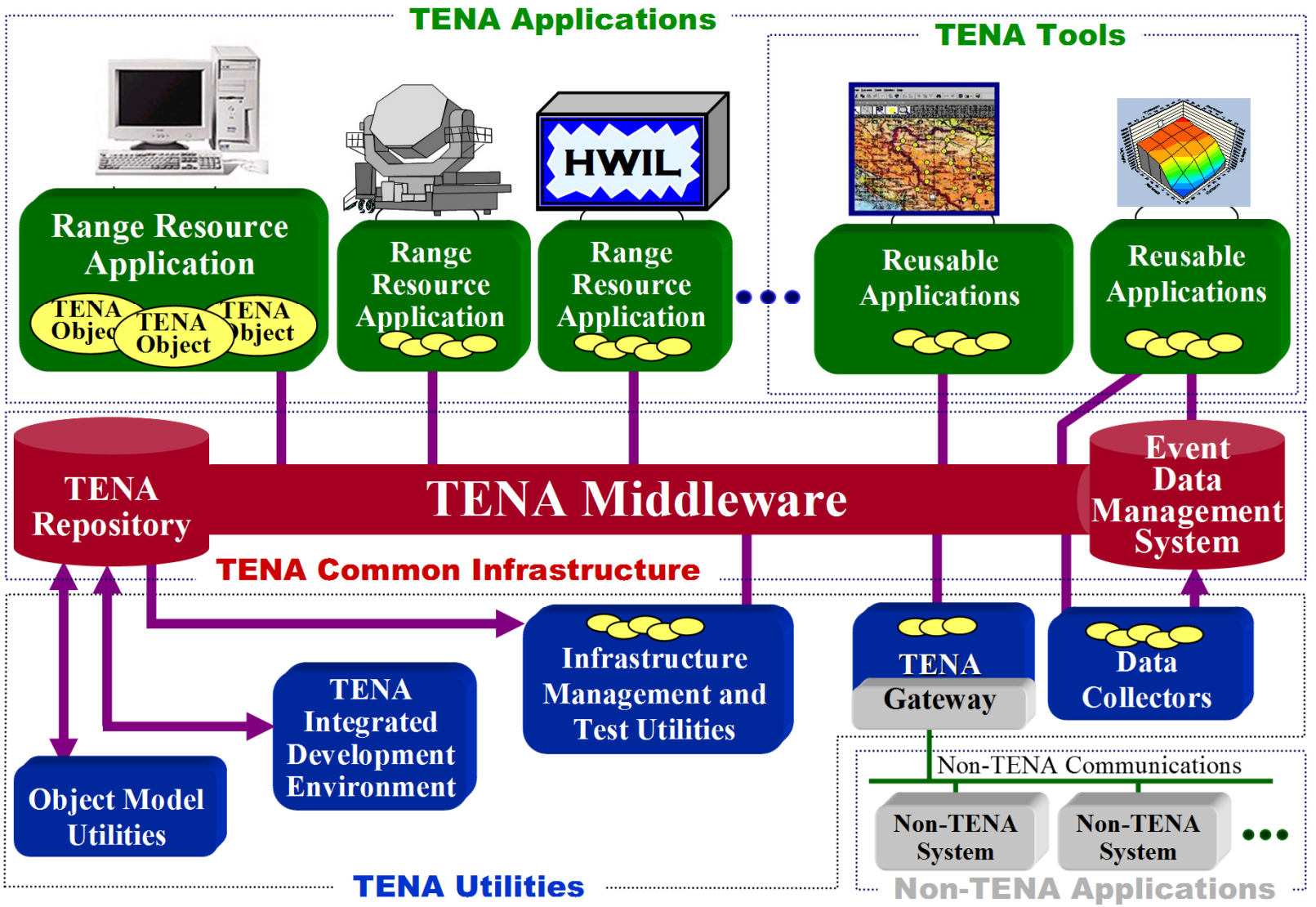
- Simulators exchange updates and events through standardized Protocol Data Units (PDU)
- All events are broadcast and available to all interested participants
- Receiving node is responsible for calculating and distributing the effect
- No central node for scheduling or conflict resolutions
- “ground truth” information is shared, perceptions have to be created by receiving nodes
- Dead reckoning computes expected positions

- Documentation
  - IEEE 1278.1 - Application Protocols
  - IEEE 1278.1A - Supplement to Application Protocols - Enumeration and Bit-encoded Values
  - IEEE 1278.2 - Communication Services and Profiles
  - IEEE 1278.3 - Exercise Management & Feedback (EMF) - Recommended Practice
  - IEEE 1278.4 - Verification Validation & Accreditation

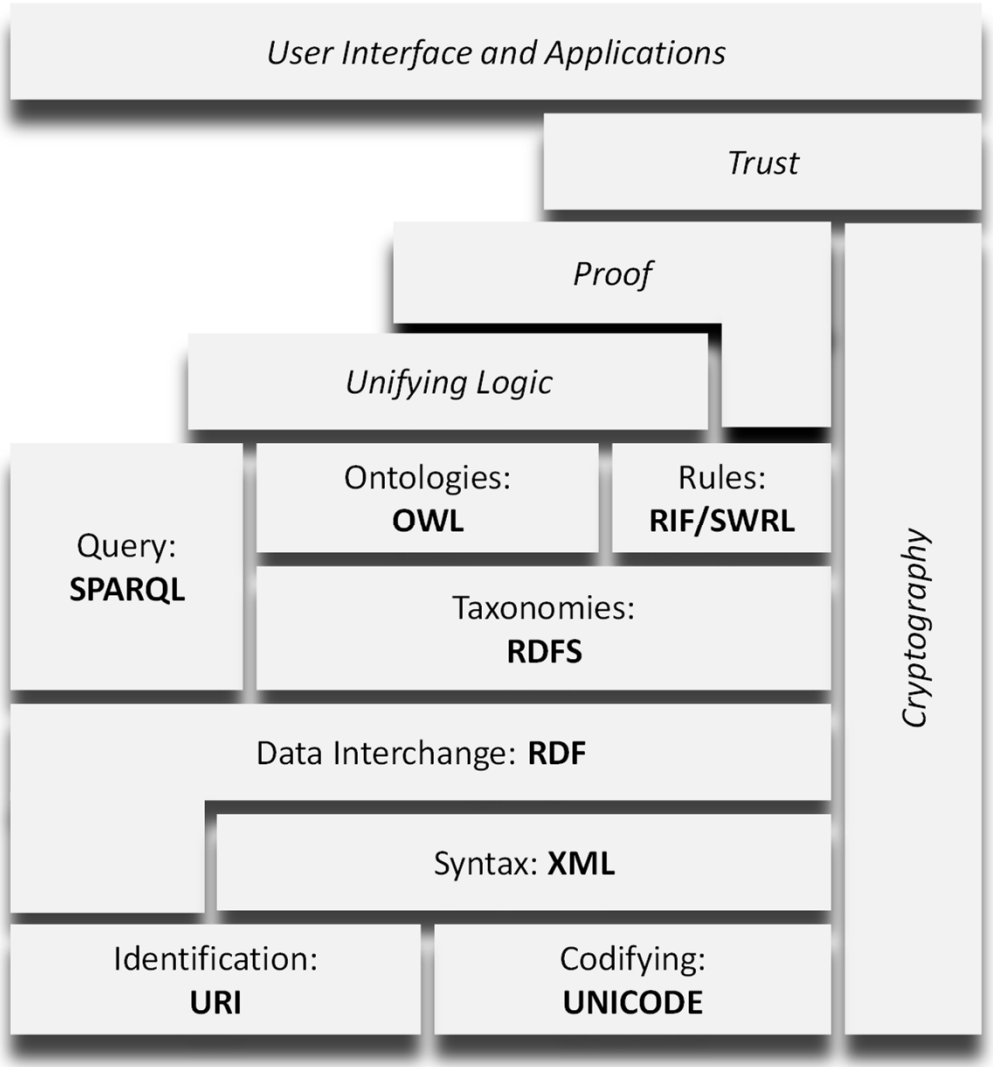
## Test and Training Enabling Architecture

- Test and Training Enabling Architecture (TENA)
  - Developed for the US Test and Training Ranges
  - Need to integrate Live, Virtual, and Constructive Solutions
  - Real-time LVC
  - Plug-and-play for systems after they were adapted by the TENA group
  - Developed as an alternative to HLA, but both are now part of the LVC-Architecture Roadmap

# TENA Overview



# Semantic Web Stack



# Interoperability and Composability Revisited

## Interoperability

- The ability of two systems to **exchange data** and the ability of the receiving system to use these data after reception.

## Composability

- The **consistence representation of truth** in all participating components of the simulation.

# Questions / Point of Contact

**Andreas Tolk, PhD**

andreas.tolk@gmail.com