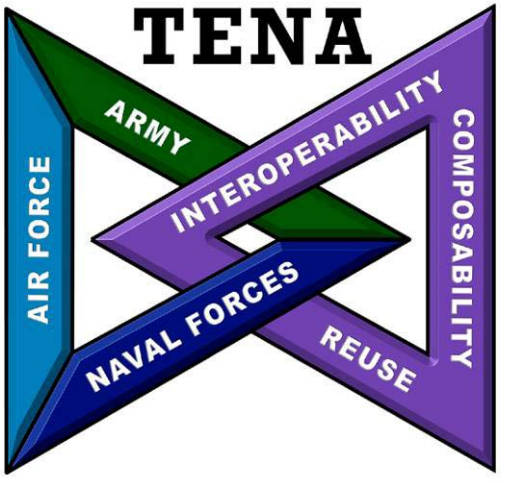


Test and Training Enabling Architecture (TENA)

Overview Course

Dr. Edward T. Powell

TENA Architect

Agenda

- **Building and Using TENA**
  - TENA Software Development Activity
  - Some Uses of TENA
  - Managing and Using TENA
- **The TENA Architecture**
  - Architecture Structure
  - Architecture Details
    - Meta-Model
    - Object Model
    - Middleware
    - Repository and Utilities
- **Summary and Conclusions**

Goals of the TENA Overview Course

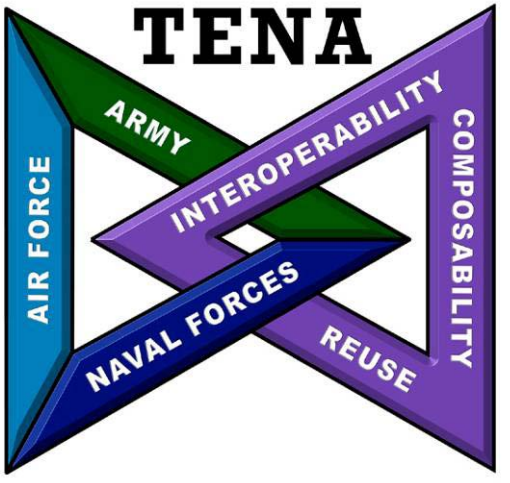
- **Attendees**
  - Anyone who wants to know more about TENA and its current capabilities
  - Anyone who will be developing TENA-compliant systems
- **Goals**
  - Provide an overview of TENA concepts and features (with rationale)
  - Provide insight to the significance of TENA capabilities
  - Provide insight to the current capabilities of TENA
- **This lecture will not cover:**

Topics are covered in:

• The functionality & operation of the TENA Middleware • TENA Definition Language (TDL) in detail • The TENA Middleware API	**Technical Introduction Course (TIC)**
• Design techniques for TENA-compliant applications • Hands-on experience using TENA Middleware • Sample applications and programming exercises	**Hands-On Training (HOT)**

What You Should Learn in This Course

- What is TENA?
- What are the components of TENA?
- What is a Logical Range?
- What is the TENA Meta-Model?
- What is a SDO?
- What is the TENA Object Model?
- What is a Logical Range Object Model (LROM)?
- What is TENA compliancy?
- How do you develop a TENA-compliant application?
- What is the TENA Definition Language (TDL)?
- On what computer platforms does the TENA middleware currently operate?
- How can you get the software and more training on TENA?



TENA Software Development Activity

Overview

TENA Mission

Currently, range systems tend to be non-interoperable, "stove-pipe" systems

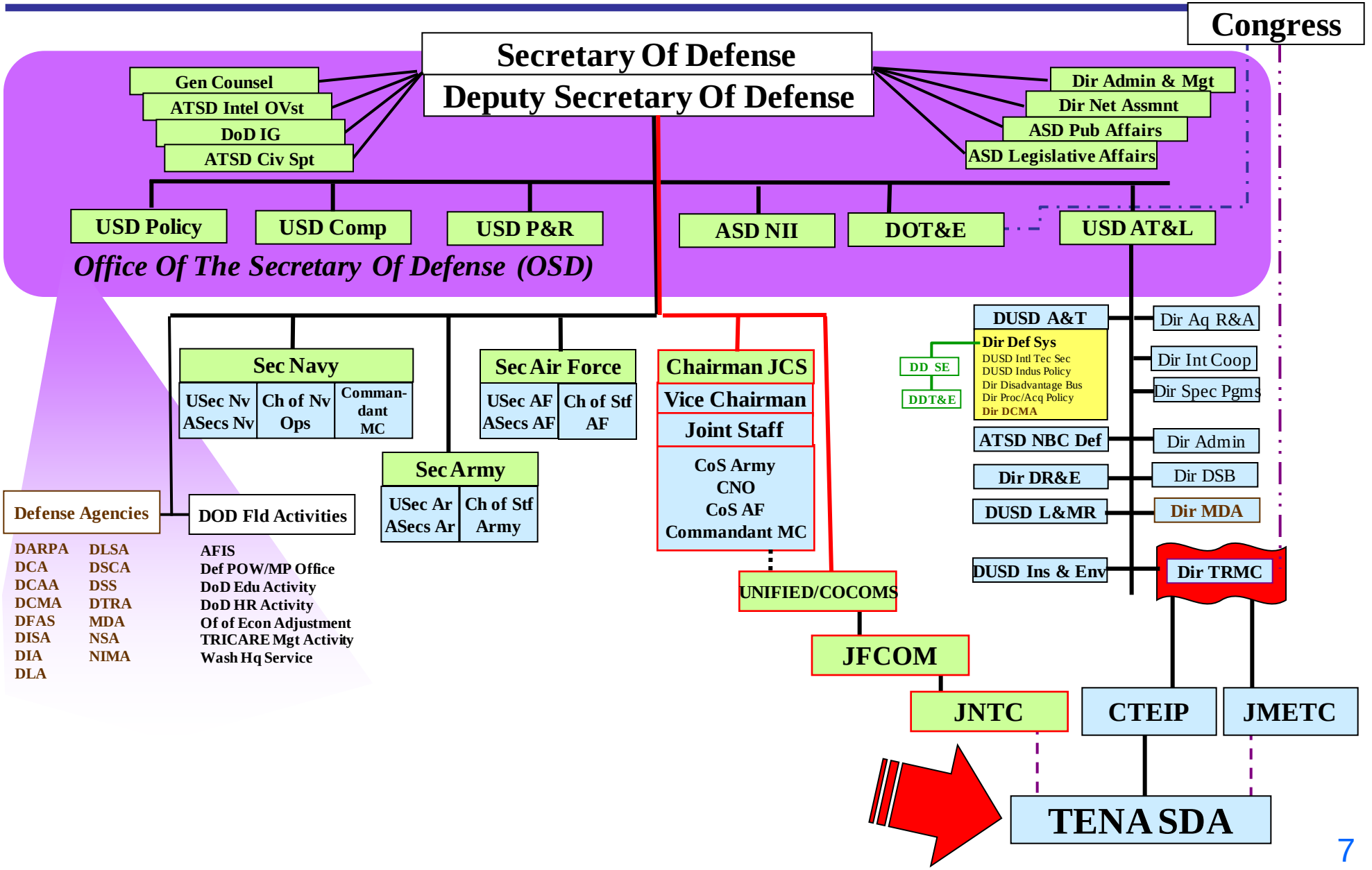
The purpose of TENA is to provide the architecture and the software implementation necessary to

- **Enable Interoperability among range systems, facilities, simulations, C4ISR systems in a quick, cost-efficient manner, and**
- **Foster Reuse for range assets and for future developments**

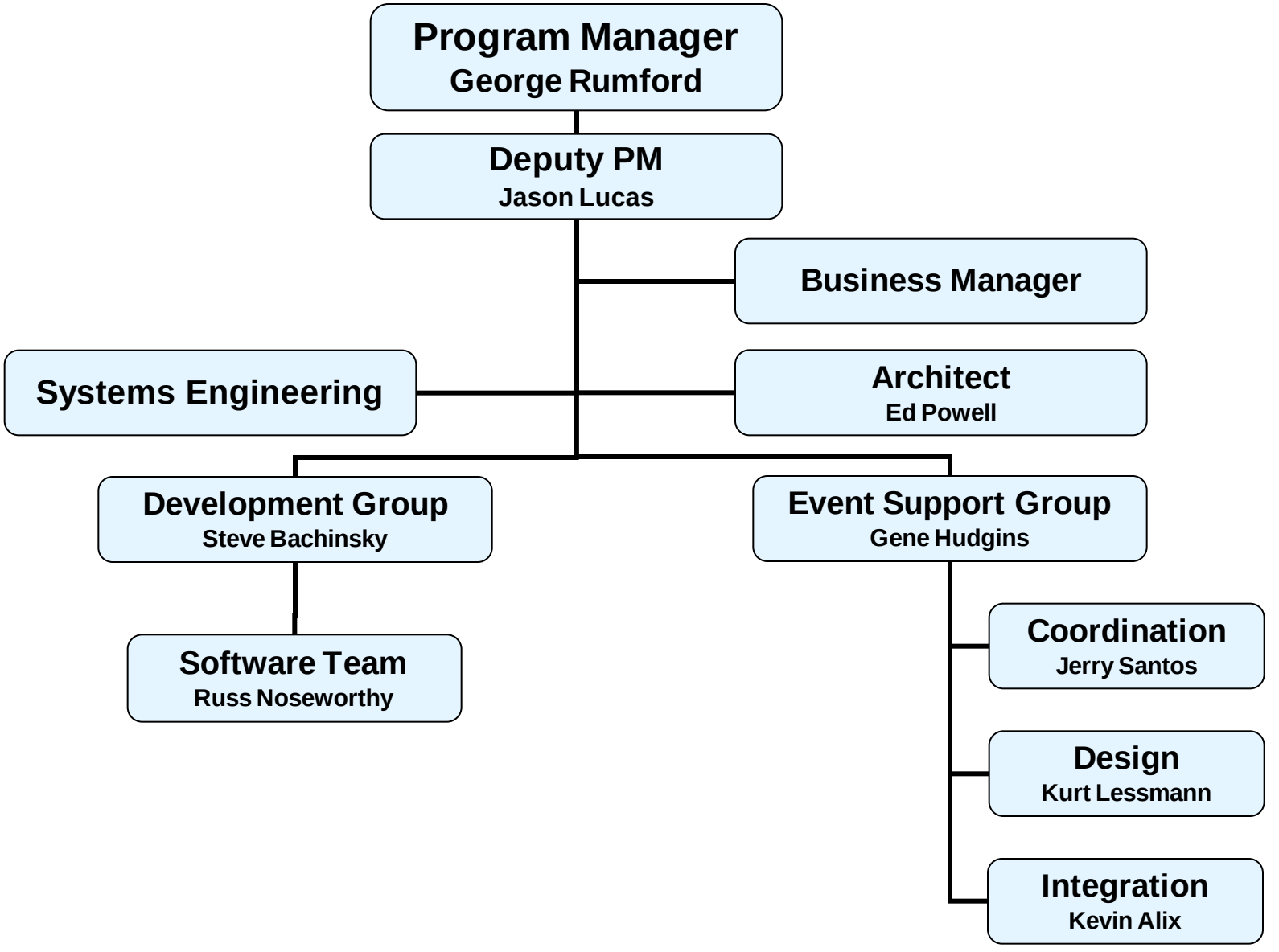
- **Support the warfighter (Joint Vision 2010/2020)**
- **Enable simulation-based acquisition**
- **Foster test and training integration**
- **In the long term: SAVE MONEY!**

Lay the Foundation for Future Test and Training Range Instrumentation

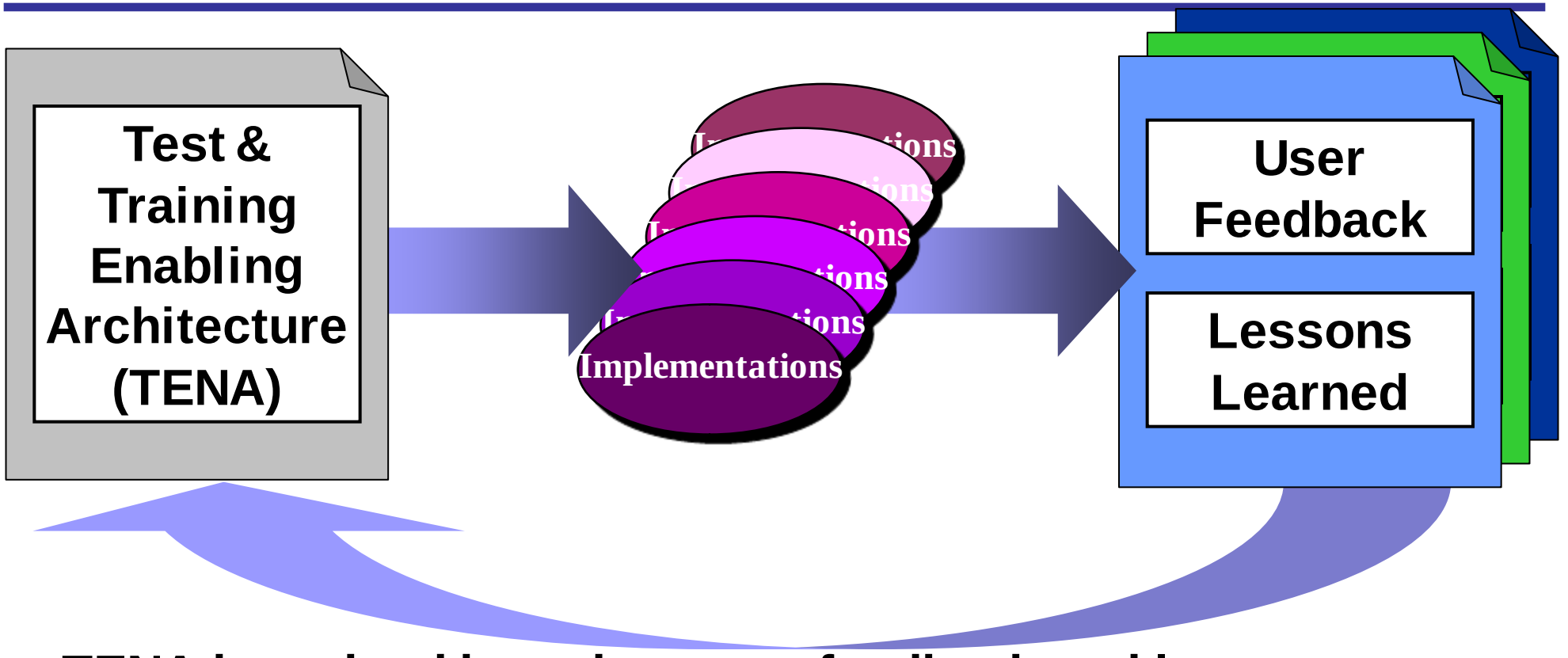
Where TENA SDA Fits in DoD



TENA SDA Organization



TENA Development Strategy

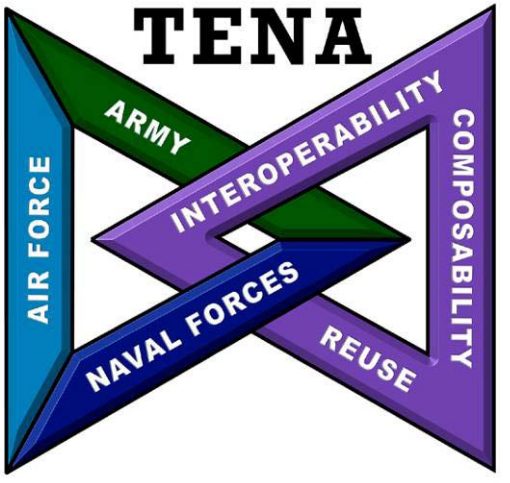


- TENA is revised based on user feedback and lessons learned from working software implementations
- TENA will be revised in the future based on future Implementations

TENA is based on real-world tests at real ranges

TENA is an Open Architecture

- **SEI defines an Open System as "a collection of interacting software, hardware, and human components designed to satisfy stated needs with interface specifications of its components that are fully defined, available to the public, maintained according to group consensus, in which the implementations of the components conform to the interface specifications."**
- **TENA is maintained according to a consensus of its users assembled as the TENA Architecture Management Team (AMT)**
  - TENA Architectural Specification is publicly defined and available on the web
  - TENA Middleware Specification (API) is publicly available on the web
  - TENA Object Model is publicly available and downloadable without restriction
    - An Event Designer can create or modify object models for a given event to satisfy their particular event requirements
- **TENA Middleware exists and is being used to support real events**
  - Built on open source software – CORBA ACE/TAO
  - Government owned, without proprietary software
  - Studying possible open source release



Some Uses of TENA

Range Integration in Millennium Challenge 2002 (MC02)

Blue Forces

- Ships
- Ground forces
- Aircraft

Opposing Forces

- Aircraft & air targets
- Ships
- Ground forces

Nellis AFB
TENA Gateway

Land Range/China Lake
TENA Gateway

National Training Center/Ft. Irwin
TENA Gateway

Electronic Counter-measures Range/China Lake
TENA Gateway

Sea Range/Point Mugu
TENA Gateway

US Marines/So. California Logistics Airfield

Joint Network

Joint Training, Analysis, and Simulation Center

TENA Gateway

Integrating Software

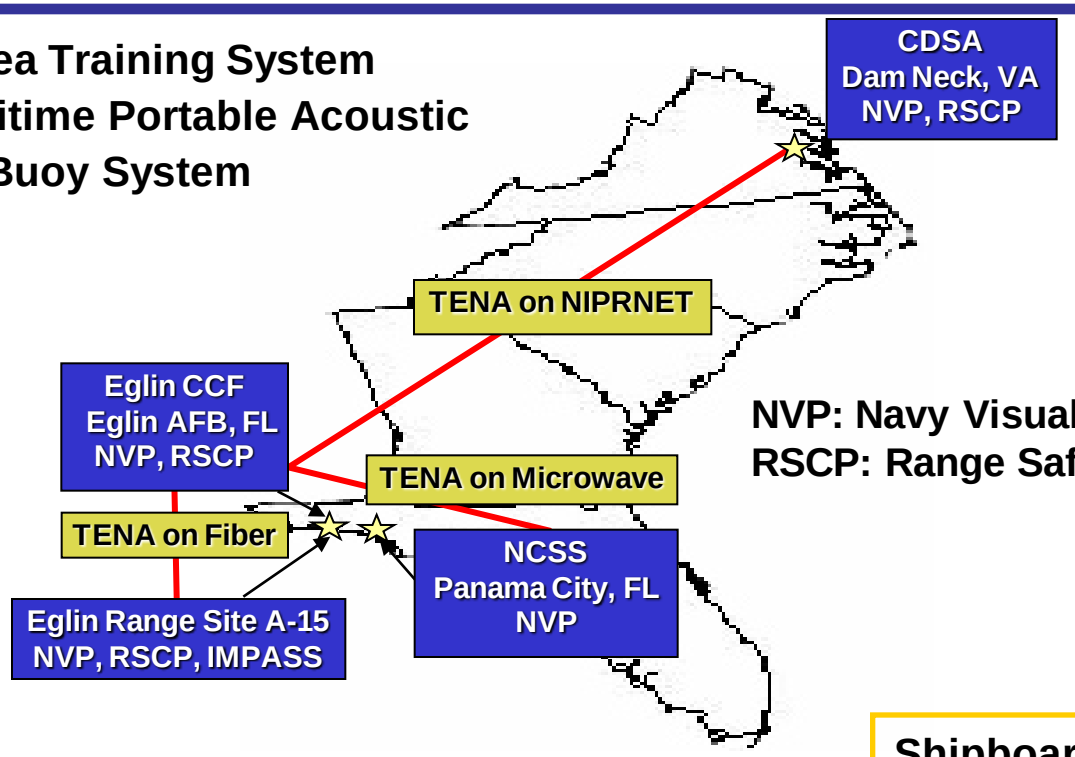
Global Command & Control System

Modeling & Simulation Feed

Command, Control, Communications, Computers, Intelligence Feed

VAST / IMPASS Over-the-Water Scoring

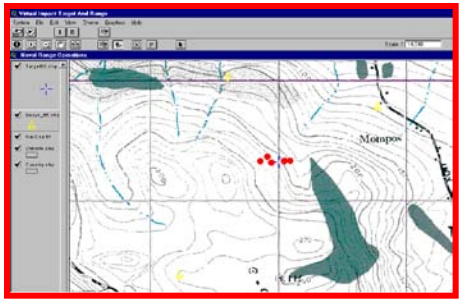
VAST: Navy Virtual At Sea Training System
IMPASS: Integrated Maritime Portable Acoustic Scoring and Simulator Buoy System



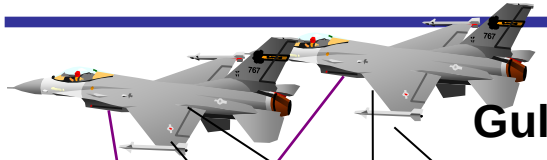
NVP: Navy Visualization Program
RSCP: Range Safety Control Program

GPS
Acoustic Processing
Communication Link

Shipboard Processing
Map Rendering
Virtual Target



JCIDEX 03 / TENA Activity



Live Infrastructure

Gulfport/Shelby/Camden MOA

ARDS GPS Pods

TACTS Pods

JTIDS Terminal

CRTC TACTS GND STN

CRTC LAN

ARDS GND STN

JTIDS TENA IF Gateway

TACTS TENA IF Gateway

ARDS TENA IF

Router

JCIET ADNET

Router

SA/AAR Display

- PCDS -

Camp Shelby MS

JECG Display

Rangeview

Ft. Rucker (opt)

SA/AAR Display

JECG Display

JECG Display

-Rangeview –

(Analysis

(AMO, TSPI, JTIDS,

Instrumentation)

Gulfport CRTC

Casualty Assessment Workstation

(A/G, G/G, A/A geo-pairing)

SA/AAR Display

- PCDS -

(TSPI)

Router

Router

TENA Display

Rangeview

Eglin AFB

SA/AAR Display

AUV Fest 2003
SIMDIS

Seahorse

Crawler

REMUS

CETUS

Acoustic

Range Craft

RF

Range Buoy

Range Control

RS-232

Range Data Gateway

TCP

Static Mine Locations

TENA

Range Information Display Center (Keyport)

AUV Fest Ops Center (Keyport)

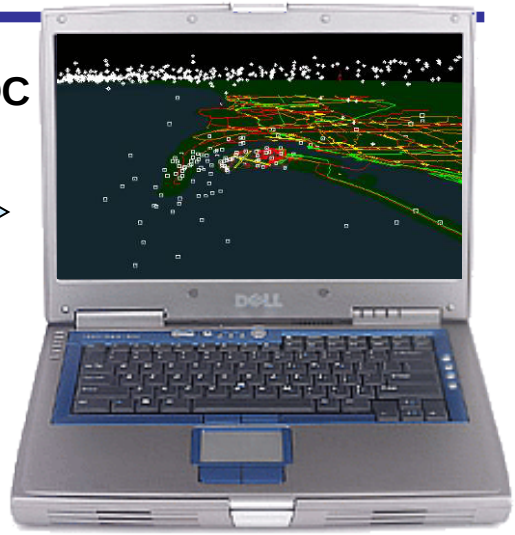
Newport

SIMDIS Use of TENA

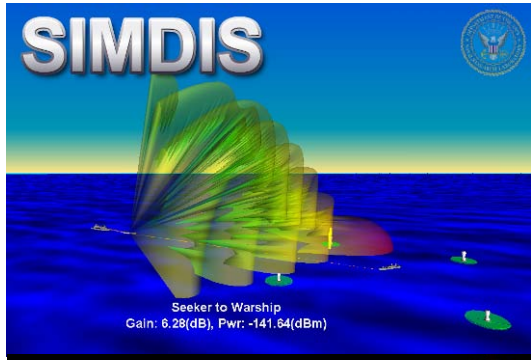
Southern California

TENA

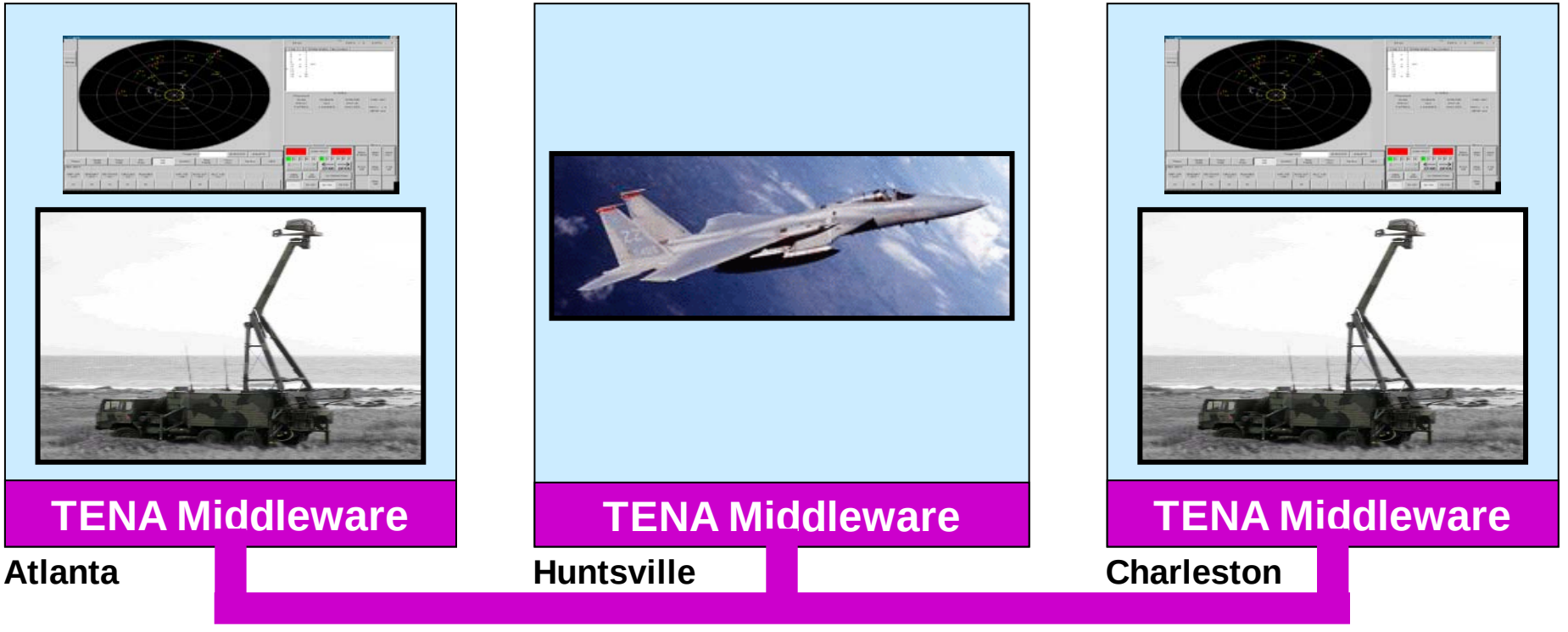
NRL Washington, DC



- **Duration testing using SCORE TSPI data feed**
  - Four consecutive days
    - Win XP, Red Hat 9, Solaris 5.8
    - Processed 180,000+ entities
  - Two consecutive days
    - Win XP, Red Hat 9
    - Processed 53,000+ entities
- **Results and observations**
  - No issues with discovery latency
  - No issues with update latency
  - No issues with CPU usage
  - No issues with memory usage



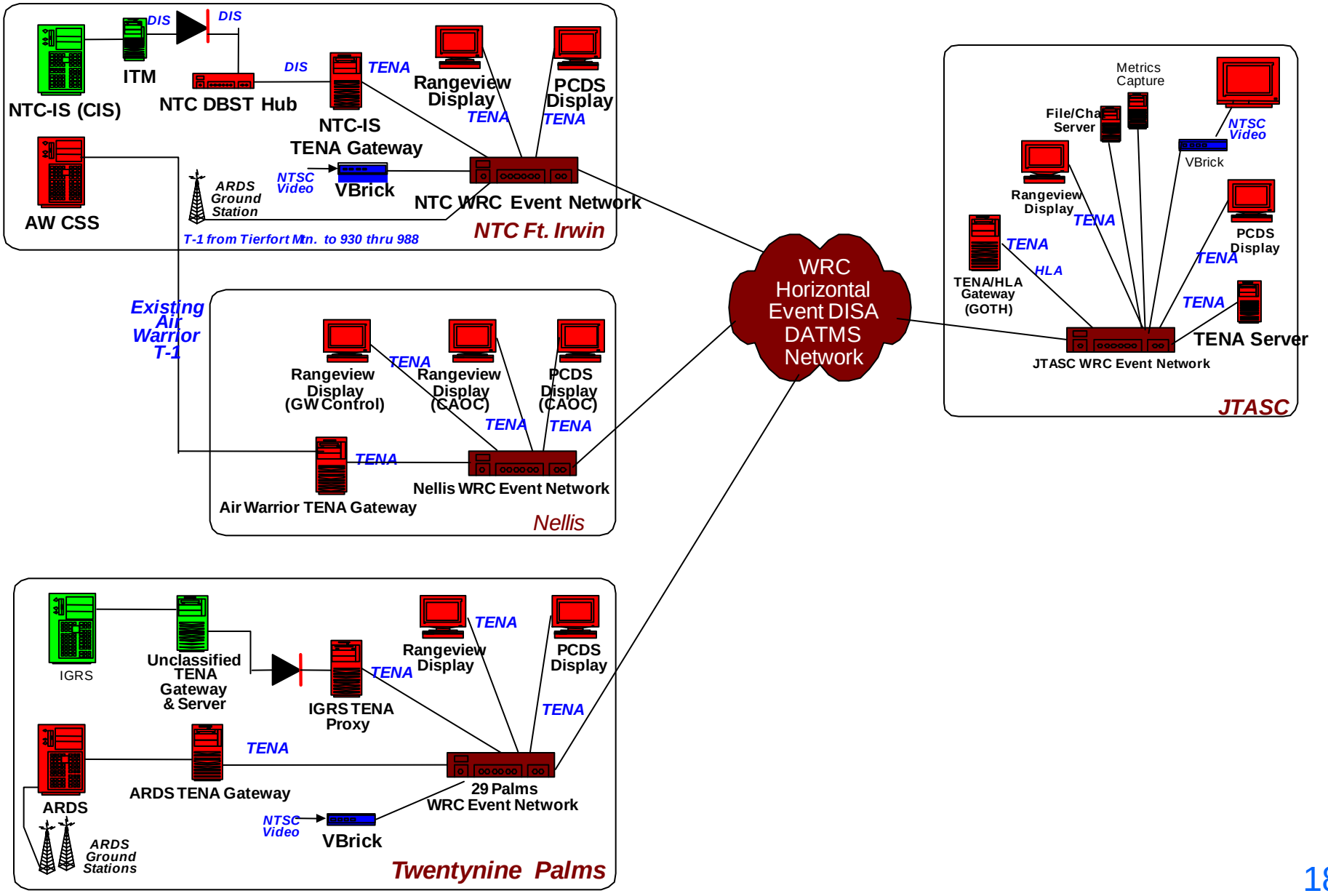
Threat Systems Test of TENA



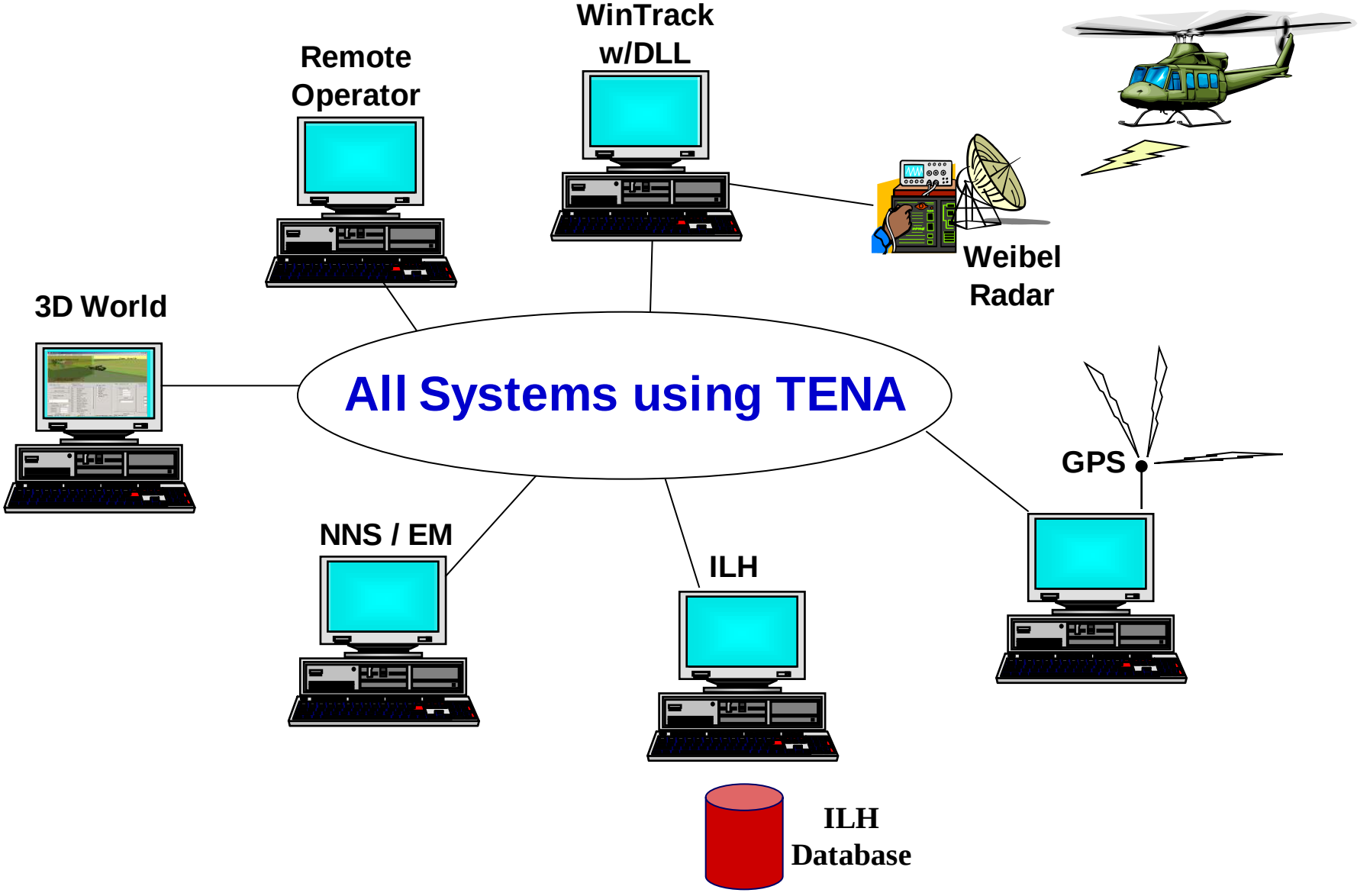
- **Testing and analysis by Scientific Research Corporation (SRC)**
- **Results and observations:**
  - TENA Middleware appears stable and predictable
  - TENA Object Model format is sufficient for representation of threat systems
  - TENA provides satisfactory functionality and performance to be utilized within a threat simulation scenario and for fielding threat simulations

JNTC Horizontal Thrust Event Jan 04

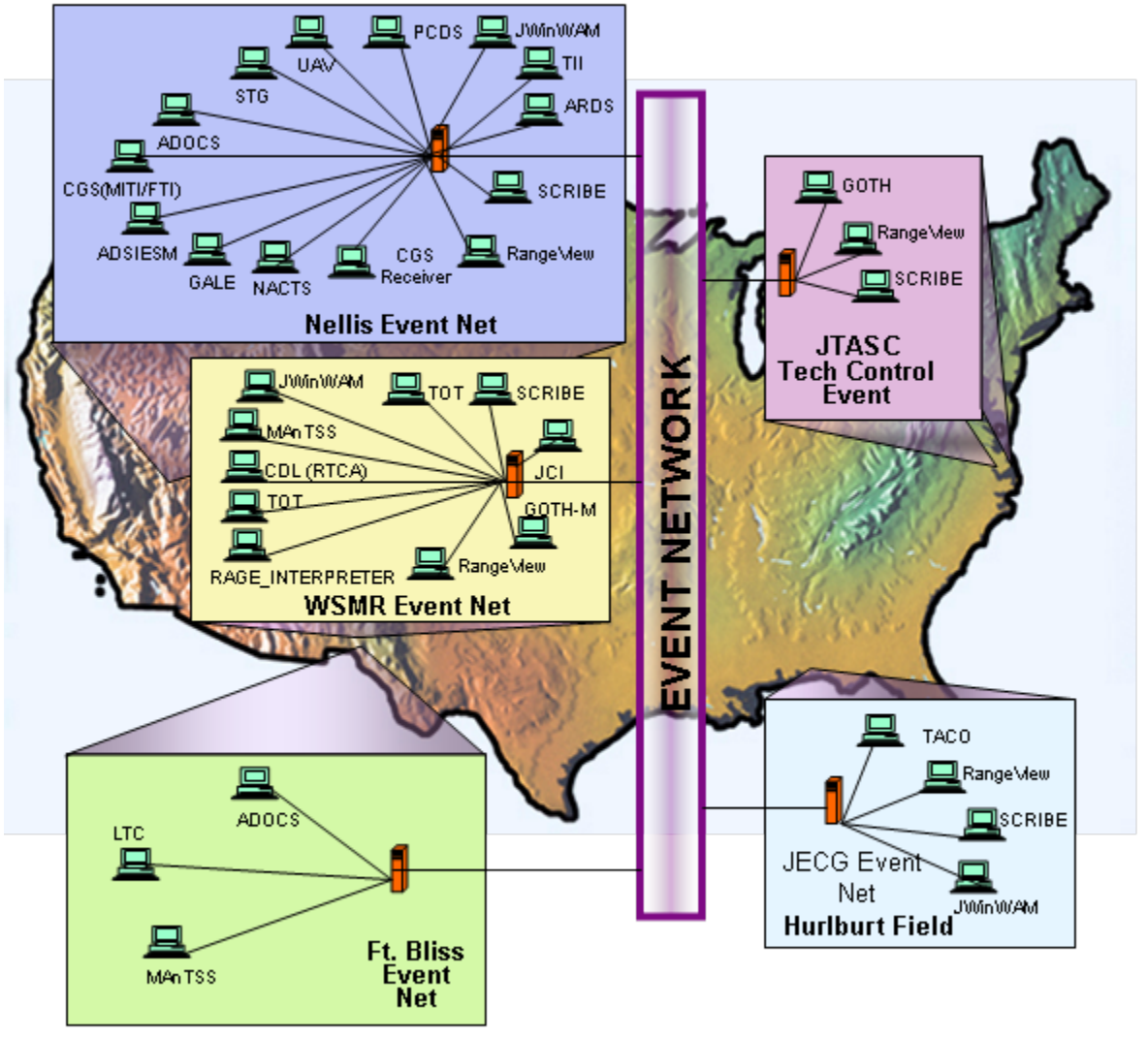
Range Integration & Instrumentation Solution



Weibel Radar Integration



Joint Red Flag 2005



- **ADOCS**: Automated Deep Operation Coordination System
- **ADSIESM**: Air Defense Systems Integrator
- **ARDS2TENA Gateway** (aka ARDS/ARDS-lite)
- **CDL**: Common Data Link (aka RTCA: Real-Time Casualty Assessment)
- **GALELITE**: Graphical Area Limitation Environment
- **GOTH**: Gateway of TENA/HLA
- **GOTH-M**: Gateway of TENA/HLA (Using Mak RTI)
- **JCI**: Joint CDL Interface (aka TENA ARDS/ARDS-Lite interface)
- **JLT**: JRF05 LROM Tool (TENA Interface Acceptance Test Tool)
- **JWinWAM**: Joint Windows-based Warfare Assessment Model (aka TENA2WAM)
- **LTC** (JFIIT): Link16 to TENA Converter
- **MAnTSS**: Modular Analysis and Test Support Software (White Cell) (JCMD)
- **CGS Receiver**: Common Ground Station (aka MTI/FTI: Moving Target Indicator/Fixed Target Indicator
- **NACTS**: Nellis Air Combat Training System Gateway
- **PCDS**: Personal Computer Debriefing System (aka Catch.TENA)
- **RAGE_INTERPRETER** (WSMR)
- **RangeView**: Display
- **SCRIBE**: TENA Logger
- **STG**: Static Target Generator
- **TACO**: TACtical Office
- **TTALC**: JFIIT TSPI Logger
- **TII**: TENA ICADS (Individual Combat Aircrew Display System)
- **TOSTADA** (aka DIS-TENA Gateway)
- **TOT** (aka OT-TES: Operational Test Tactical Engagement System)
- **UAV**: Unmanned Aerial Vehicle

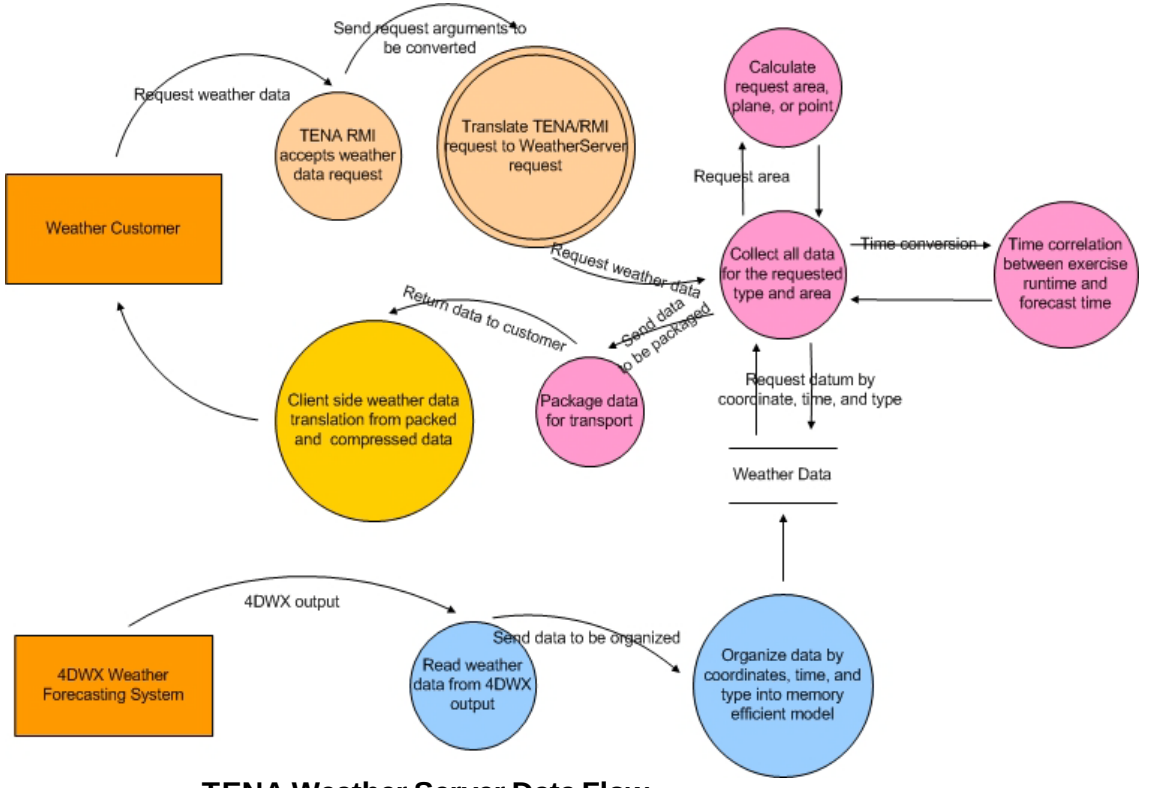
TENA in Real-Time Embedded Instrumentation by *NetAcquire*

- **Goal: demonstrate commercial-off-the-shelf (COTS) TENA operation in the following domains:**
  - **Real-time (strict constraints on data acquisition and response time)**
  - **Direct hardware interfaces not standard on COTS desktops**
    - **Aerospace serial I/O formats (synchronous, telemetry, special protocols, etc.)**
    - **GPS (time and position)**
    - **Analog input/output**
    - **Digital and pulse input/output**
    - **IRIG timing**
    - **Avionics buses (1553, ARINC, 1394)**
    - **GPIB (IEEE-488) instrumentation**
  - **Inexpensive, ruggedized, mobile form-factor**

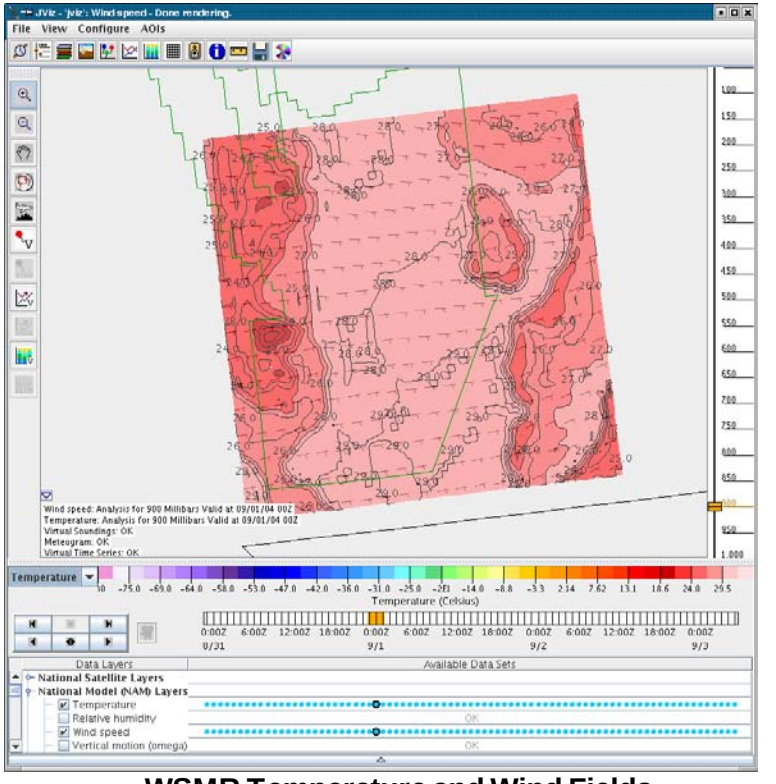
- **Accomplishments:**
  - **NetAcquire hardware/software product now successfully runs TENA**
  - **Direct synchronous serial hardware interface to FPS-16 radar system is supported**
  - **Radar system data auto-populated into TENA Radar SDO in real-time**
  - **Little or no programming required to support different radar data formats**
- **NetAcquire runs a true real-time operating system, device drivers, and application software**
  - **Provides TENA with deterministic and bounded response times**

TENA Used to Distribute 4-Dimensional Weather Data

- **Team from Dugway Proving Ground Meteorology Division, National Center for Atmospheric Research, and Keane Corporation developed a sophisticated weather server using TENA**
- **Weather information generated by real-time, 4D data acquisition is processed by the TENA Weather Server and made available to TENA-enabled test event clients**
- **Distributed Test Events need weather data:**
  - Wind, temperature, barometric pressure, precipitation, time (4th dimension)



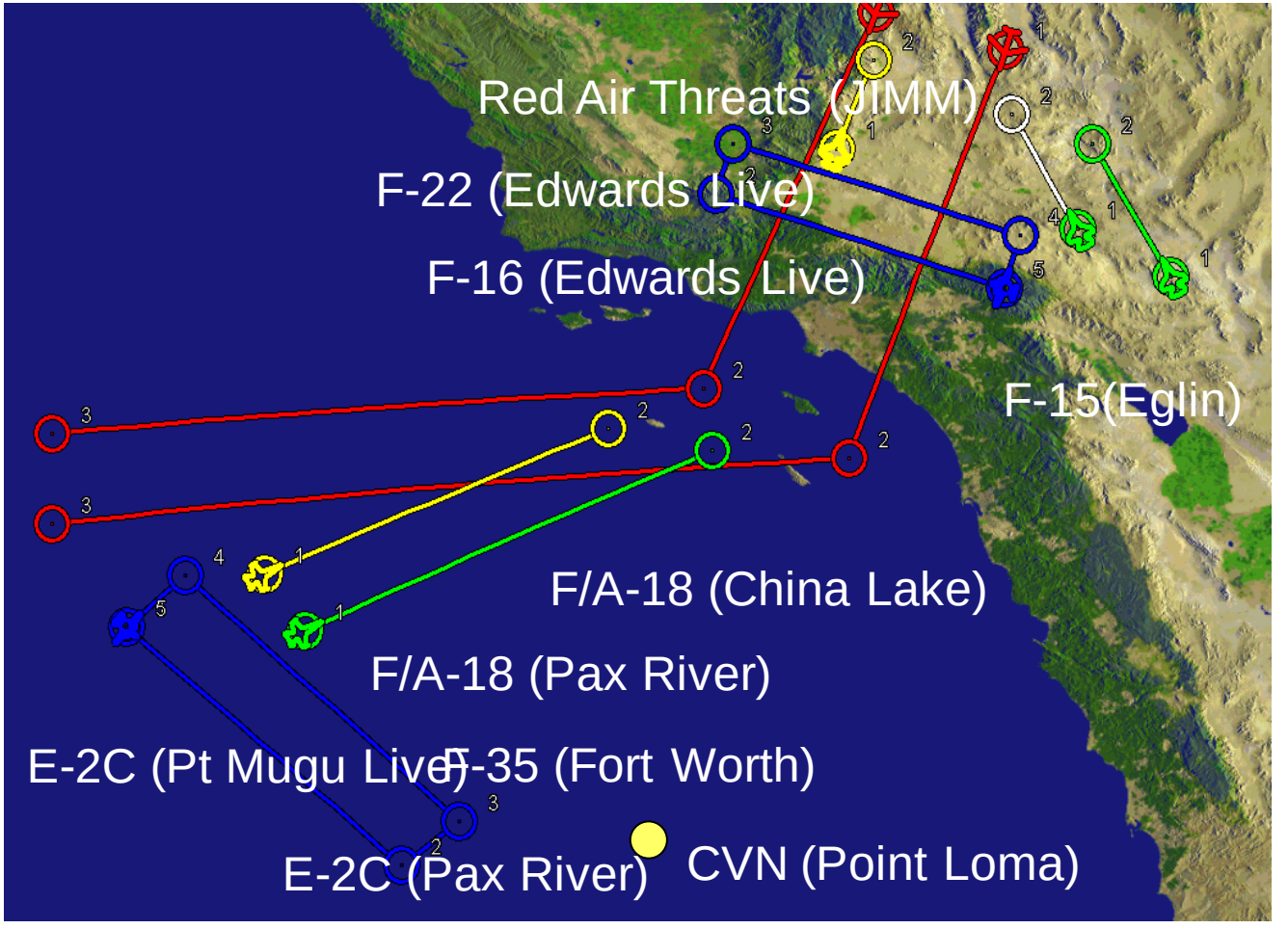
TENA Weather Server Data Flow



WSMR Temperature and Wind Fields

InterTEC Air Combat Mission JMETC Event

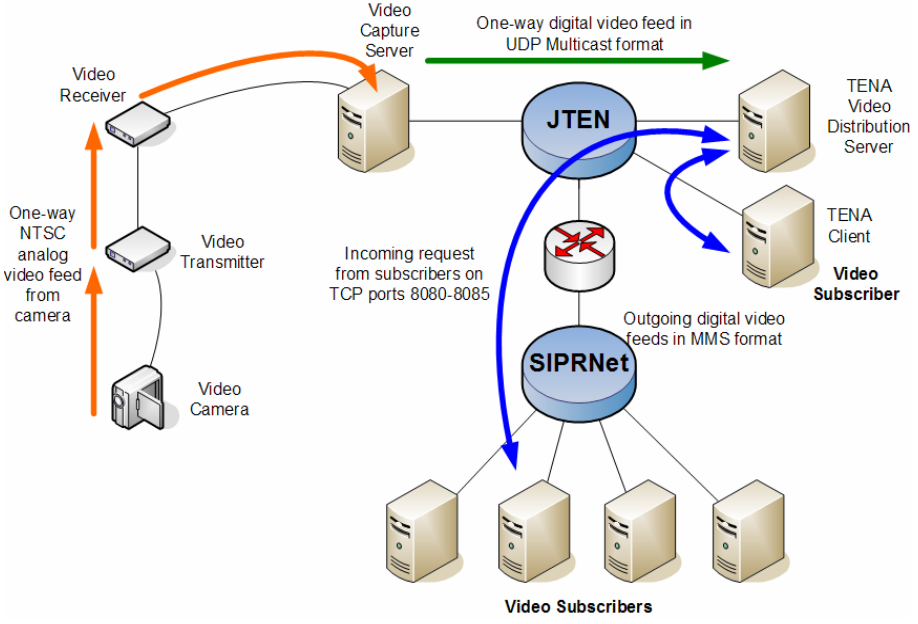
- TENA used in this large distributed LVC C4I Link-16 test event for data distribution of instrumentation, test control and distributed simulation between multiple sites



10 locations, 12 different applications, 56 instances of those apps linked together

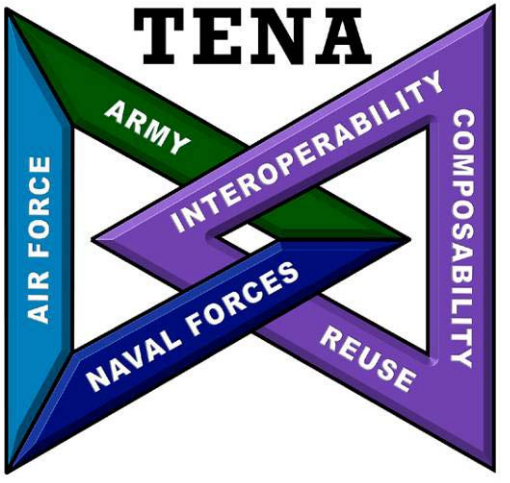
TENA Used to Control Video Distribution Services with IO Range

- **TENA used to implement video distribution system for Information Operations (IO) Range in Austere Challenge 06 exercise.**
  - CONUS and OCONUS client terminals (30+) received video streams over SIPRNet.



- **Video Distribution Server published availability of real-time and recorded data streams via Stateful Distributed Objects (SDO)**

- **TENA auto-code generation enabled rapid development and integration of software**
  - Reduced technical risk and resulted in zero software failures during live fire event periods.



Managing and Using TENA

Architecture Management Team (TENA AMT)

- **AMT Members:**
  - 329 Armament Systems Group (329 ARSG)
  - Aberdeen Test Center (ATC), Aberdeen Proving Ground, MD
  - Air Armament Center (AAC), Eglin AFB, FL
  - Air Force Flight Test Center (AFFTC), Edwards AFB, CA
  - Army Operational Test Command (OTC), Fort Hood, TX
  - Common Training Instrumentation Architecture (CTIA)
  - Dugway Proving Ground (DPG)
  - Electronic Proving Ground (EPG)
  - integrated Network Enhanced Telemetry (iNET)
  - Interoperability Test and Evaluation Capability (InterTEC)
  - Joint Fires Integration & Interoperability Team (JFIIT)
  - Joint National Training Capability (JNTC)
  - Naval Air Warfare Center – Aircraft Division
  - NAWC – Weapons Division
  - Naval Aviation Training Systems Program Office (PMA-205)
  - Naval Undersea Warfare Center (NUWC)
  - NAVSEA Warfare Center - Keyport
  - P5 Combat Training System (P5CTS)
  - Pacific Missile Range Facility (PMRF)
  - Redstone Technical Test Center (RTTC)
  - T&E/S&T Non-Intrusive Instrumentation
  - White Sands Missile Range (WSMR)

Meetings every 3 months

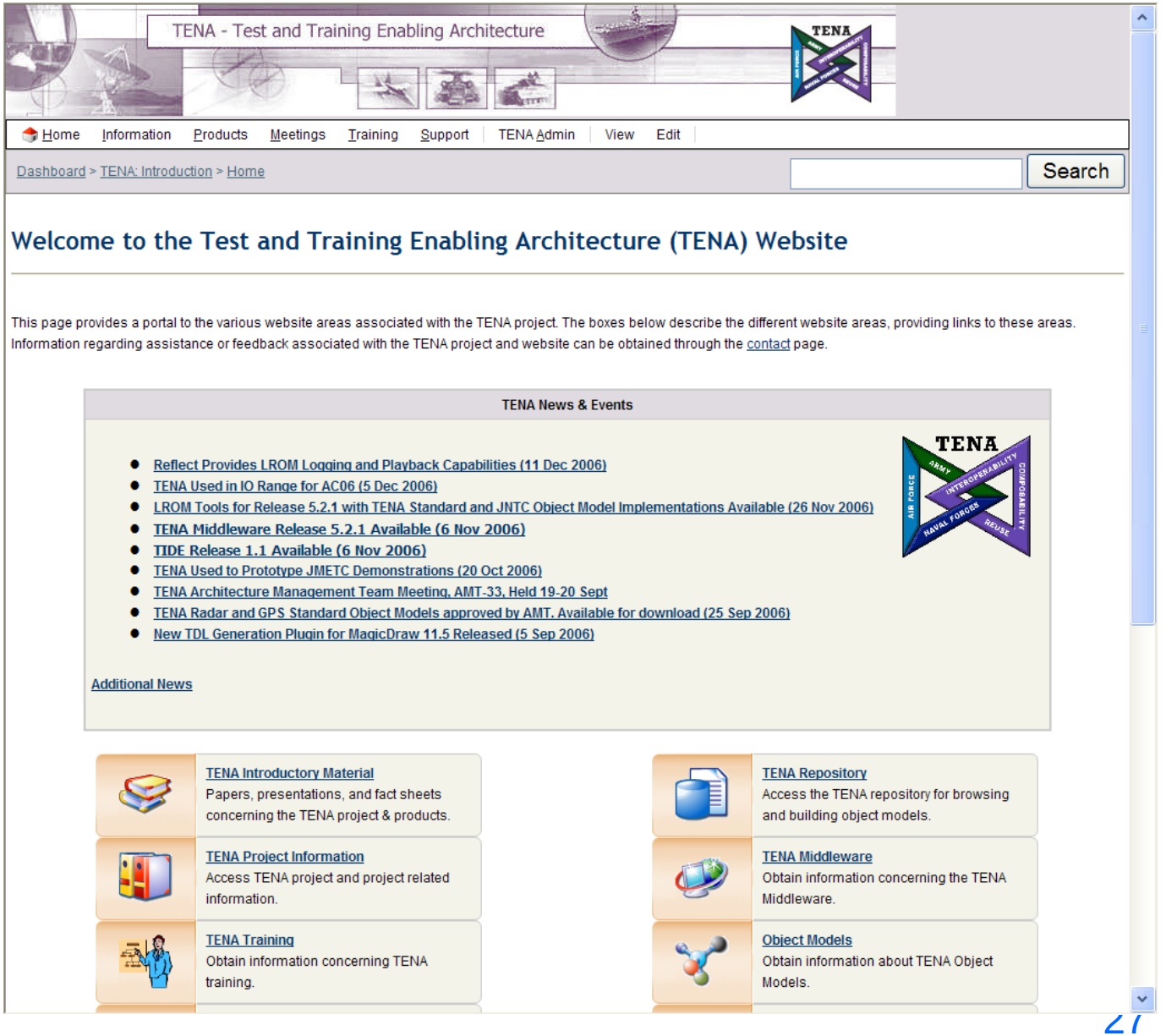
Advising Members:
- BMH Associates, Inc.
- Boeing
- Cubic Defense
- DRS
- Embedded Planet
- EMC
- Kenetics
- MAK Technologies
- NetAcquire
- Science Applications International Corporation (SAIC)
- Scientific Research Corporation (SRC)
- Scientific Solutions, Inc. (SSI)

- **Design Decisions / Trade-offs / Status / Technical Exchanges of Lessons Learned / Use Cases / Testing / Issues & Concerns Identification, Investigation & Resolution**

TENA Web Portal
http://www.tena-sda.org/

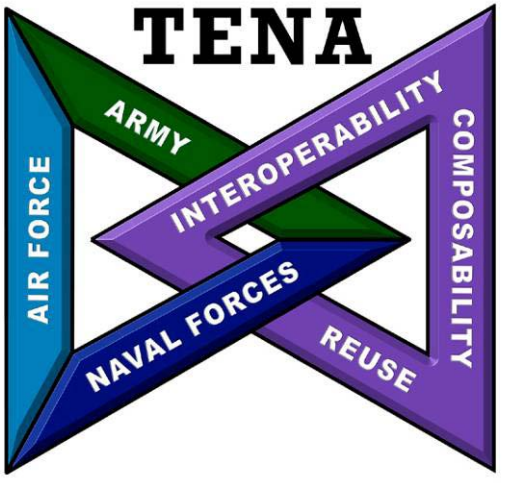
- **Registered user account required**

- **Contains**
  - News
  - Meeting Notices
  - Documentation
  - Middleware
  - Object Models
  - Training Materials

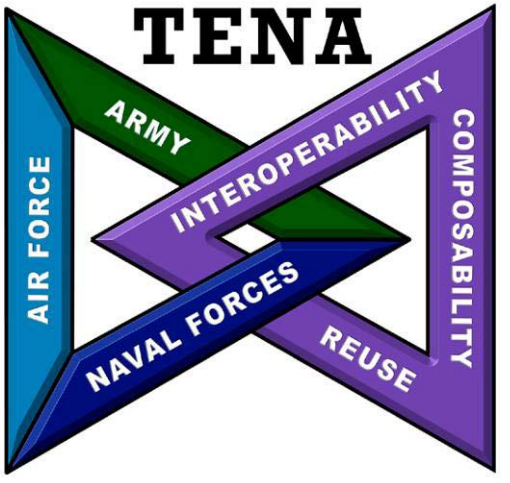


The Way Ahead for TENA

- **Continue ongoing partnership with the Joint National Training Capability (JNTC) and Joint Mission Environment Test Capability (JMETC)**
  - Use the JNTC and JNTC-like events to reduce risk and refine application of TENA to JNTC needs
- **Technically support and partner with PMs in their assessment and implementation of TENA for Test and Training applications**
- **Use the current TENA Requirements-Driven and Stakeholder-Prioritized process to spiral develop and prototype further TENA capabilities**


TENA
ARMY
AIR FORCE
INTEROPERABILITY
COMPOSABILITY
NAVAL FORCES
REUSE

Questions?

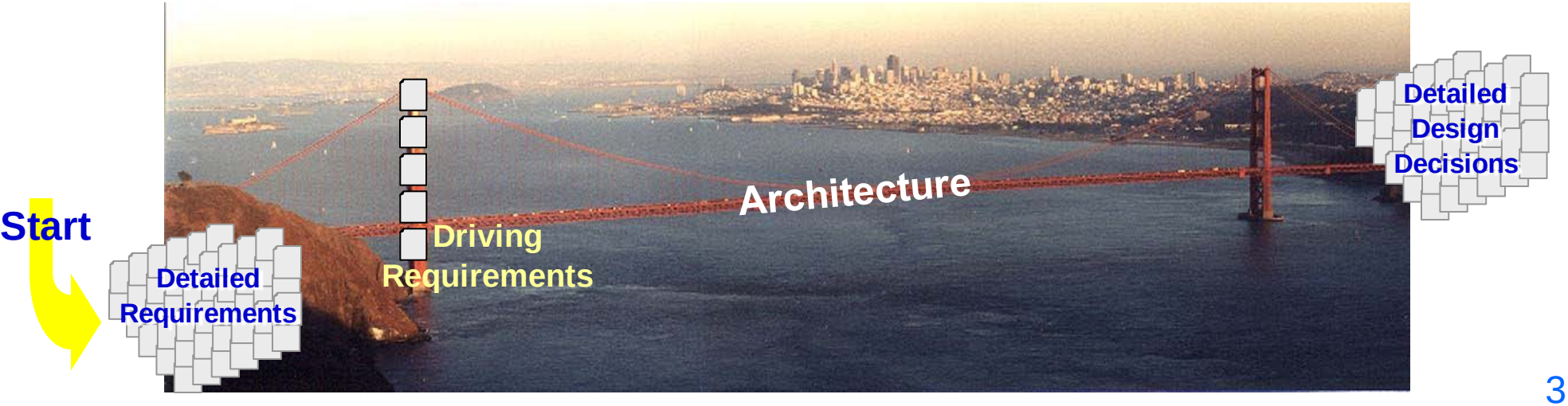


Test and Training Enabling Architecture

(TENA) 2005

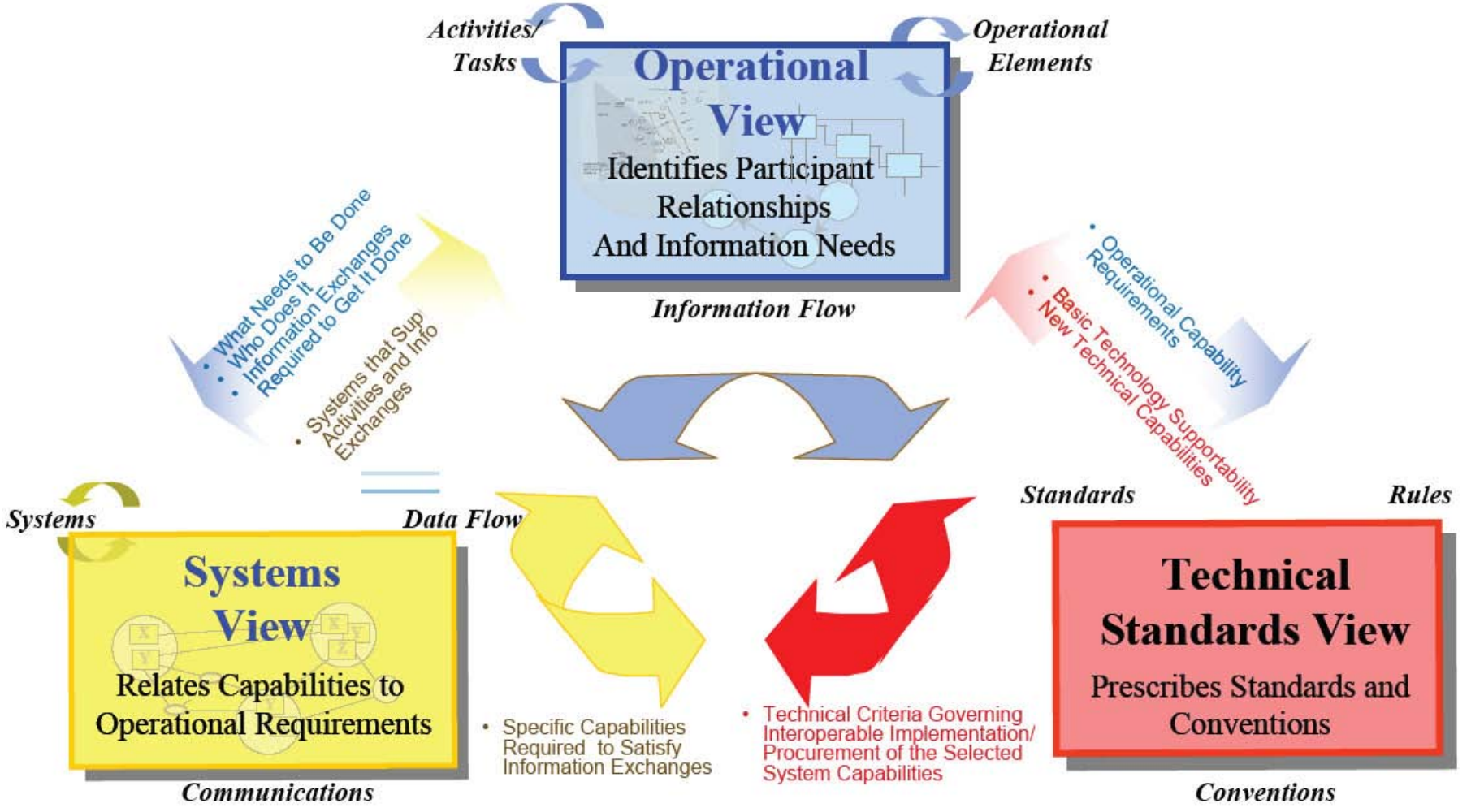
What is an Architecture?

- An architecture is a segmentation of a system (or system of systems) such that the primary pieces are identified, as well as their purpose, function, interfaces, and inter-relatedness, along with guidelines for their evolution over time
- Architectures put constraints on developers. These constraints make possible the achievement of higher level goals.
- These higher-level goals are called the system's driving requirements
- An architecture is a bridge from requirements to design



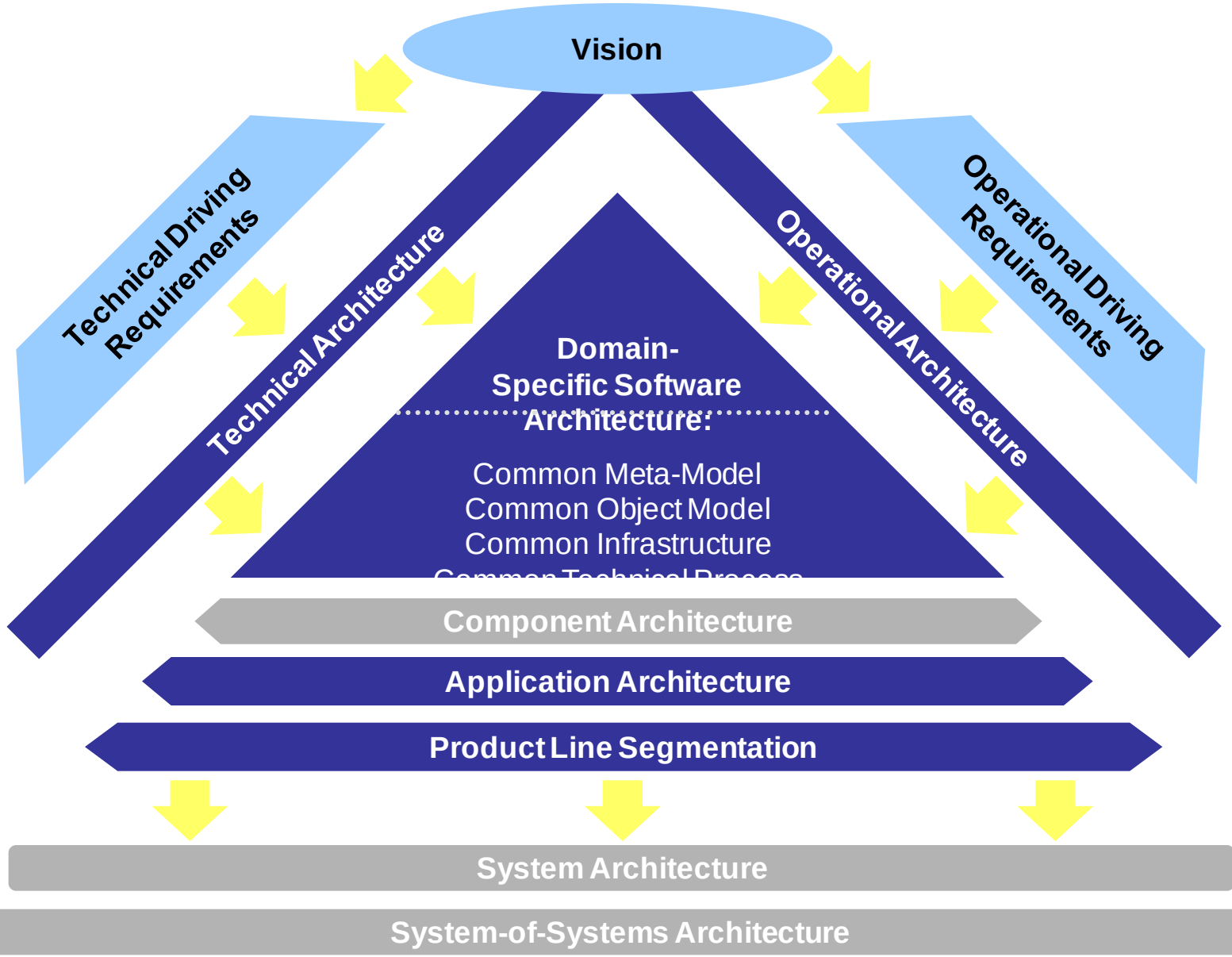
How is an Architecture Organized?

- **The C4ISR (DoD) Architecture Framework is the standard format for describing architectures for *systems***

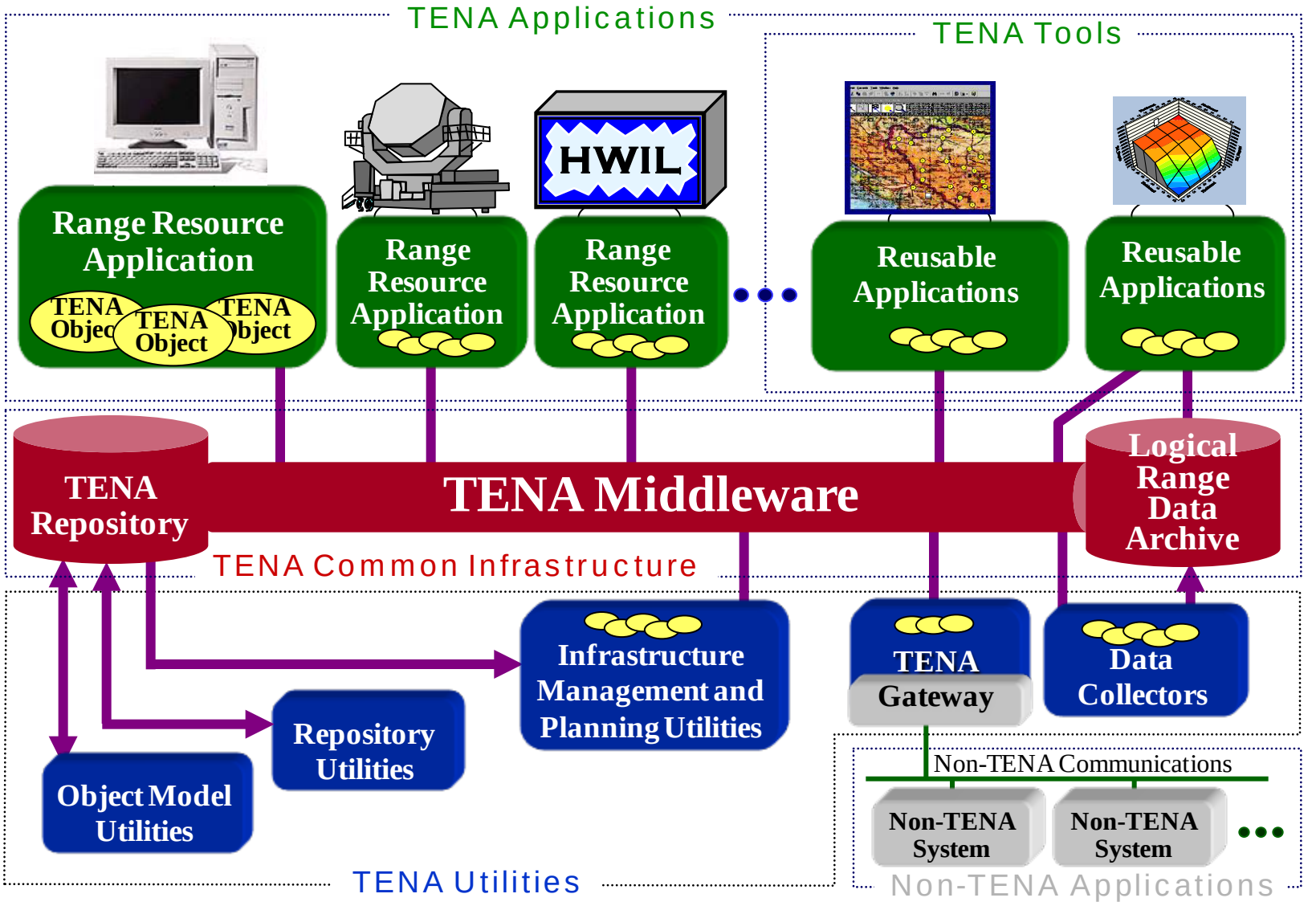


How is an Architecture Organized?

The Extended C4ISR Architecture Framework

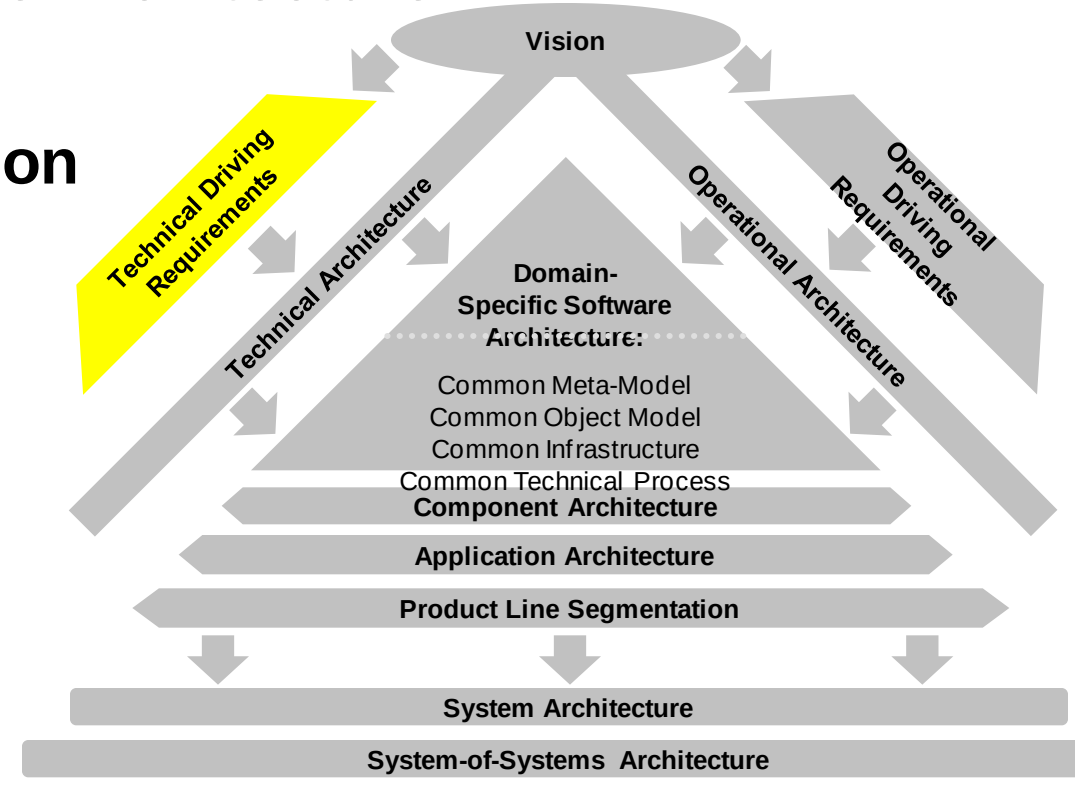


TENA Architecture Overview



TENA Organization

- **Technical Driving Requirements**
- **Operational Driving Requirements**
- **Technical Architecture View**
- **Operational Architecture View**
- **Domain-Specific Software Architecture**
- **Application Architecture**
- **Product Line Segmentation**



Technical Driving Requirements

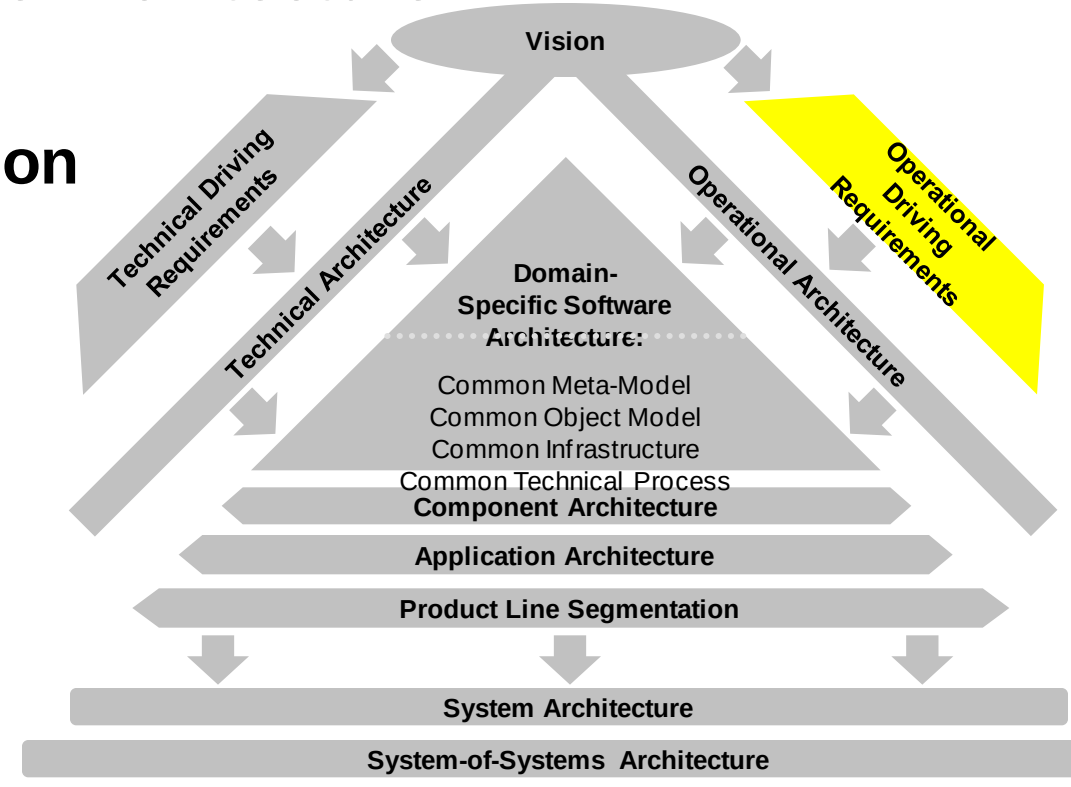
1. **Interoperability**
   - The characteristic of a suite of independently-developed components, applications, or systems that implies that they can work together, as part of some business process, to achieve the goals defined by a user or users
2. **Reusability**
   - The characteristic of a given component, application, or system that implies that it can be used in arrangements, configurations, or in enterprises beyond those for which it was originally designed
3. **Composability**
   - The ability to rapidly assemble, initialize, test, and execute a system from members of a pool of reusable, interoperable elements
   - Composability can occur at any scale—reusable components can be combined to create an application, reusable applications can be combined to create a system, and reusable systems can be combined to create an enterprise

Achieving Interoperability and Reuse

- **Interoperability requires**
  - A common architecture → **TENA**
  - An ability to meaningfully communicate
    - A common language → **TENA Object Model (OM)**
    - A common communication mechanism → **TENA Middleware, LRDA**
  - A common context
    - A common understanding of the environment → **SEDRIS (as part of the TENA OM)**
    - A common understanding of time → **TENA OM, Middleware**
    - A common technical process → **TENA Technical Process**
- **Reuse and Composability require the above, plus**
  - Well defined interfaces and functionality for the application to be reused → **Reusable Tools, Repository**
  - Place to store reusable components → **Repository**

TENA Organization

- Technical Driving Requirements
- Operational Driving Requirements
- Technical Architecture View
- Operational Architecture View
- Domain-Specific Software Architecture
- Application Architecture
- Product Line Segmentation



Operational Driving Requirements

A. **TENA must support the implementation of logical ranges,** including the management of both software and data throughout the entire event lifecycle.

B. **TENA must support the Joint Vision 2010/2020** by providing the foundation for testing and training in a net-work-centric warfare environment.

C. **TENA must support rapid application and logical range development,** testing, and deployment in a cost-effective manner.

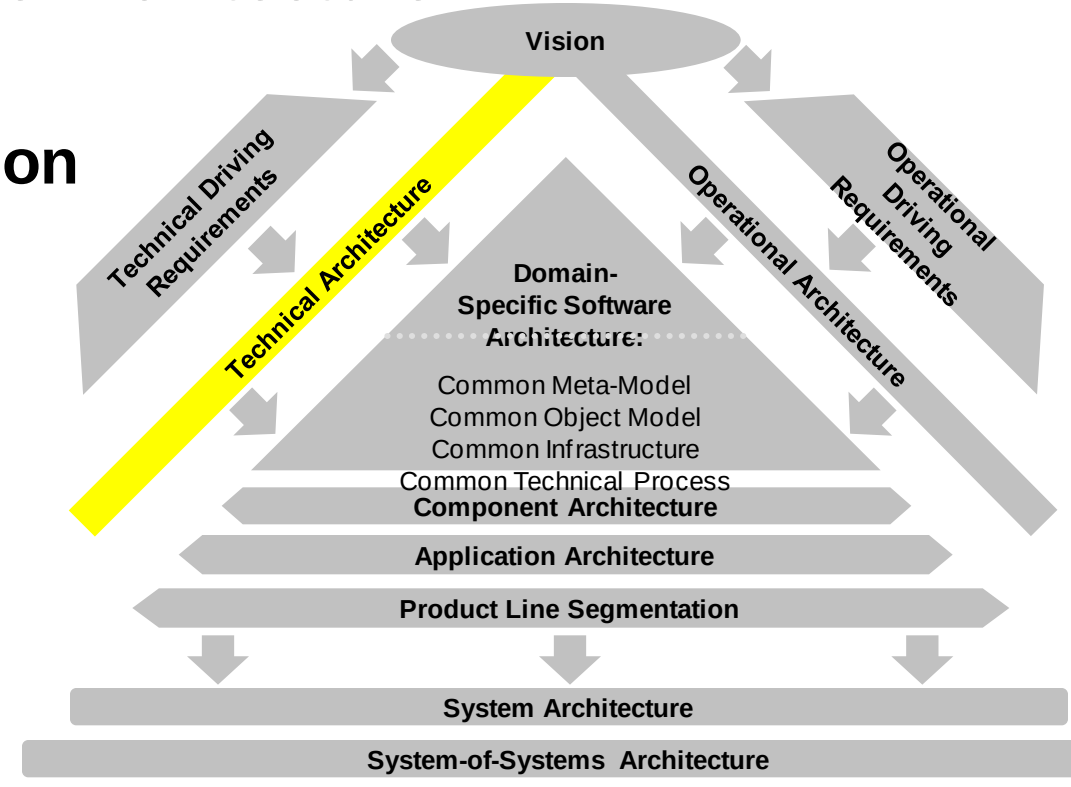
D. **TENA must support easy integration with modeling and simulation** to advance the DoD's simulation-based acquisition concepts.

E. **TENA must be gradually deployable** and interact with non-TENA systems without interrupting current range operations.

F. **TENA must support a wide variety of common range systems** by meeting their operational performance requirements, including sensors, displays, control systems, safety systems, environment representations, data processing systems, communication systems, telemetry systems, analysis tools, data archives, and others.

TENA Organization

- Technical Driving Requirements
- Operational Driving Requirements
- **Technical Architecture View**
- Operational Architecture View
- Domain-Specific Software Architecture
- Application Architecture
- Product Line Segmentation



Technical Architecture

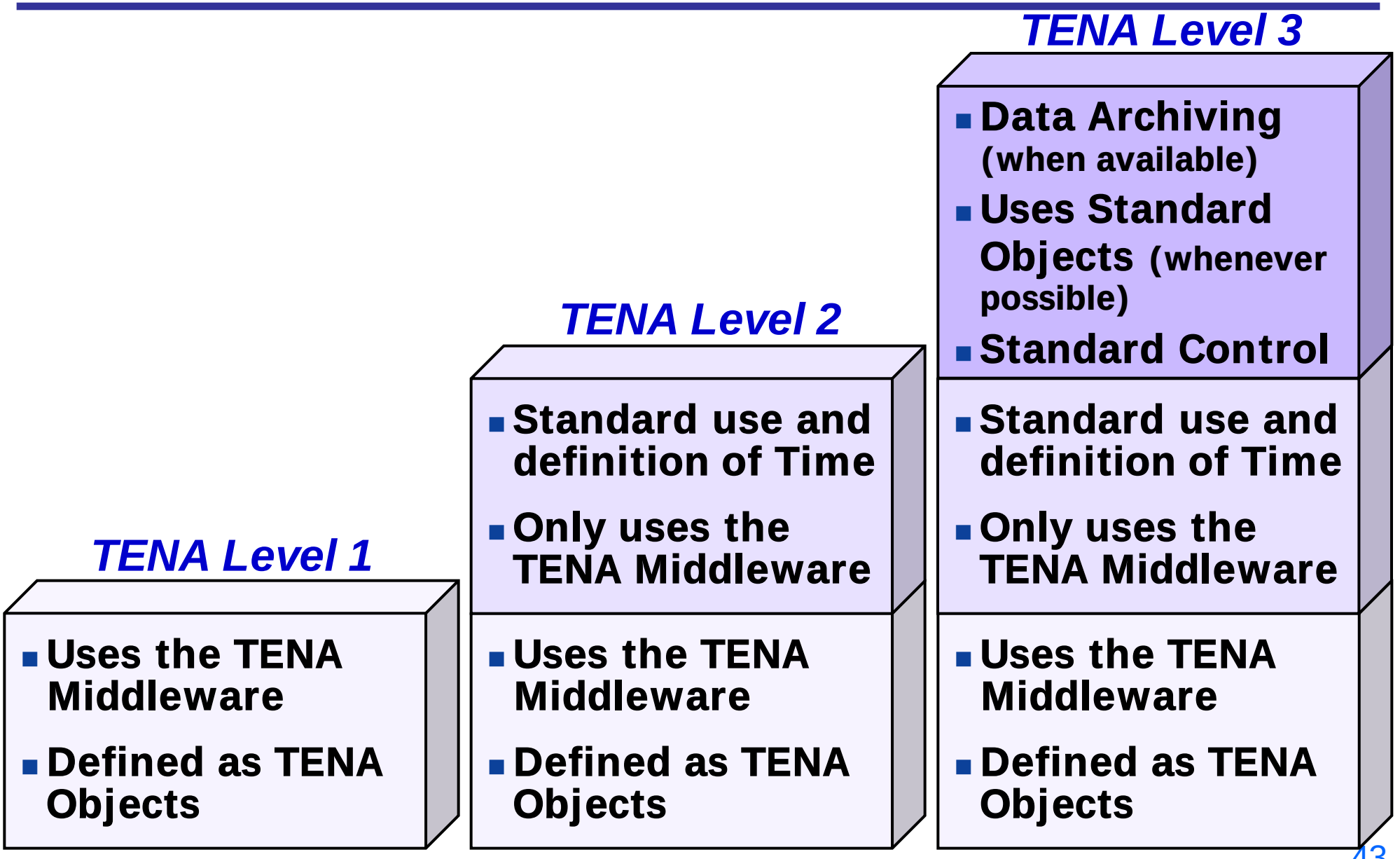
- **Rules**
  - Three sets of rules
  - Each set represents an increased level of compliancy
- **Standards**
  - Focus on those standards that TENA "incorporates" directly and indirectly
  - Including, especially, the Joint Technical Architecture

- **Compliance is based on how well a software system obeys the rules**

Technical Architecture
TENA Rules

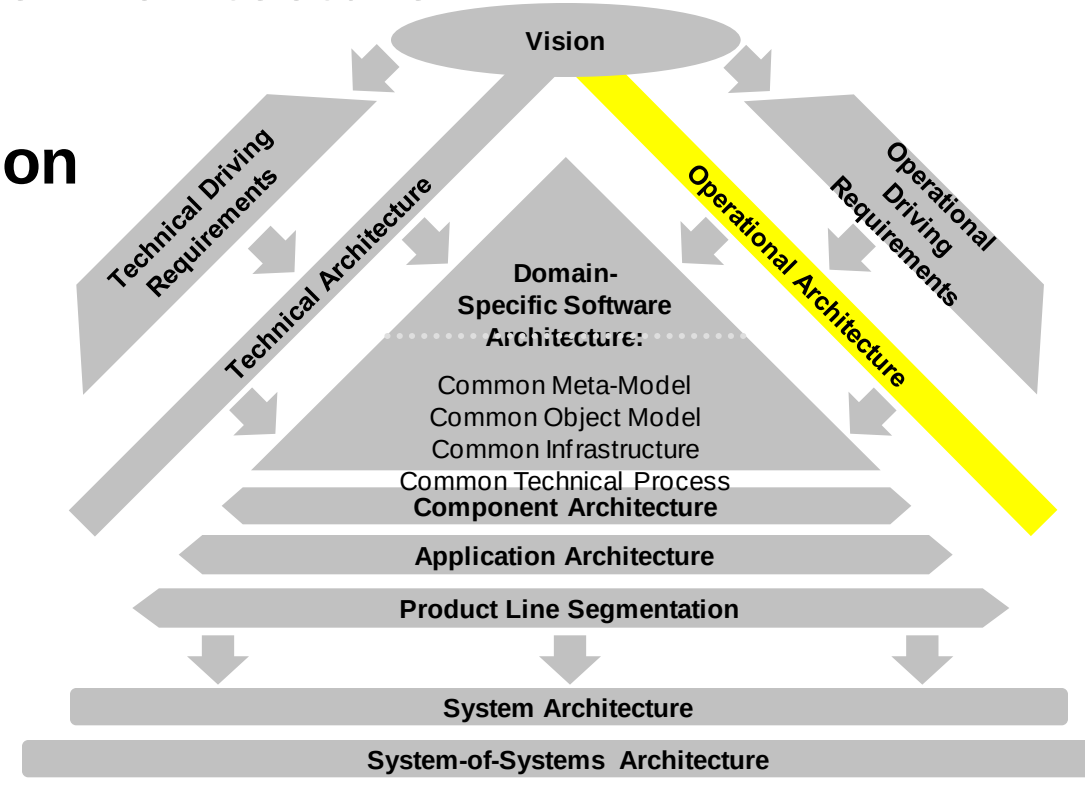
- **Rules for Minimal Compliance:**
  1. All range resource applications shall interact with each other via the TENA common infrastructure using the standard API.
  2. Each logical range shall have a logical range object model (LROM), specified in the standard manner, that contains all of the object definitions that may be produced and consumed by all range resource applications in that logical range execution.
  3. All objects in any LROM shall conform to the TENA meta-model.
- **Rules for Extended Compliance:**
  4. All execution-time information exchange among range resource applications in a logical range shall be done using the TENA Middleware as the communication mechanism with the information described in the LROM.
  5. Every range resource application owner shall provide documentation in the standard format of what information the application has been implemented to both produce and consume; and the object implementations must adhere to the contract contained in their definition.
  6. All range resource applications shall maintain accurate time. This can be done either by implementing the underlying functionality for measuring time based on a standard time-related interface provided by the TENA Middleware, or by ensuring that the computer on which the application runs has a system clock that is accurate to within tolerances required by the particular logical range. Each application developer must document how their application implements time, including a description of the accuracy of the measurements.
- **Rules for Full Compliance:**
  7. All range resource applications shall implement and publish a TENA Application Management Object
  8. Range resource applications shall not use an object definition that conflicts with a provisional or standard TENA object definition as part of a logical range object model.
  9. Range resource applications shall use the Logical Range Data Archive, when it is available for use, for all data storage and persistent communication.

TENA Compliancy Levels



TENA Organization

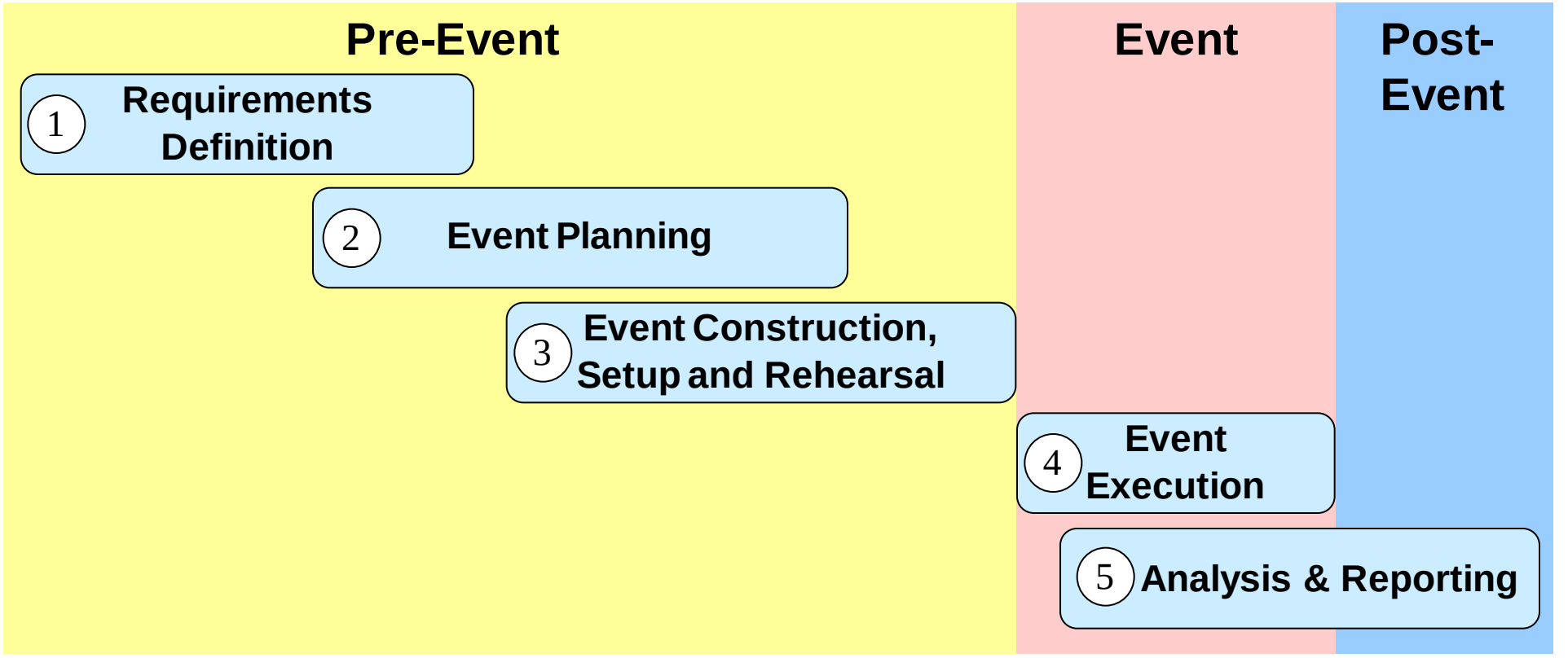
- Technical Driving Requirements
- Operational Driving Requirements
- Technical Architecture View
- Operational Architecture View
- Domain-Specific Software Architecture
- Application Architecture
- Product Line Segmentation



Operational Architecture Overview

- **Provides a "Concept of Operations" for how TENA-based range events work**
  - Three phases
  - Five activities
- **Explains concept of the "Logical Range"**
- **Defines a "Functional Decomposition" of the elements of the range domain**

Operational Architecture (including ConOps)



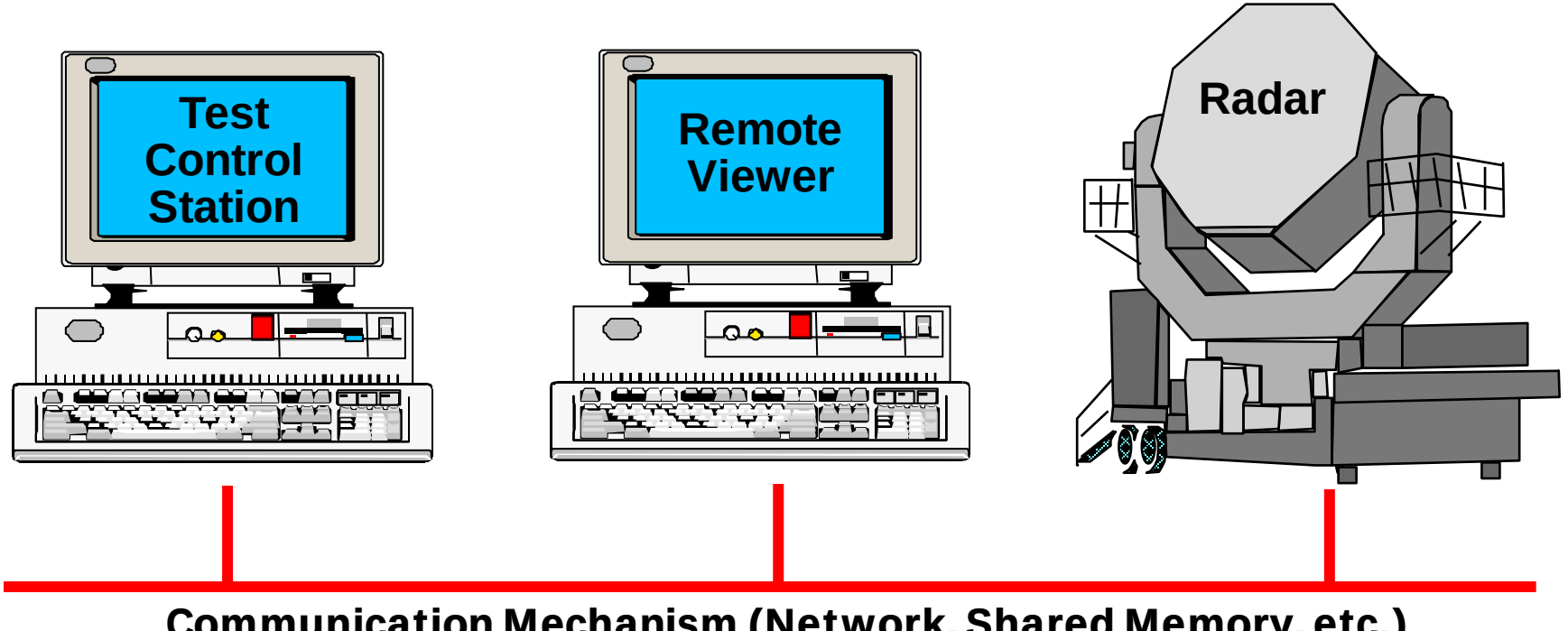
- **Three Phases**
  - Pre-Event / Event / Post-Event
- **Five Activities**
  - Requirements / Planning / Set-up / Execution / Analysis & Reporting

TENA Uses the Concept of a Logical Range

- **Logical Range – a suite of TENA Resources, sharing a common object model, that work together for a given range event**
- **TENA Resources are:**
  - Range Resource Applications - compiled to use the services provided by the TENA Middleware for interaction
  - Gateway Applications - to bridge TENA systems to legacy or other protocols or architectures
  - TENA Tools and Utilities - configured for a particular event
- **Common Object Model**
  - Logical Range Object Model (LROM) – the object definitions used in a particular event

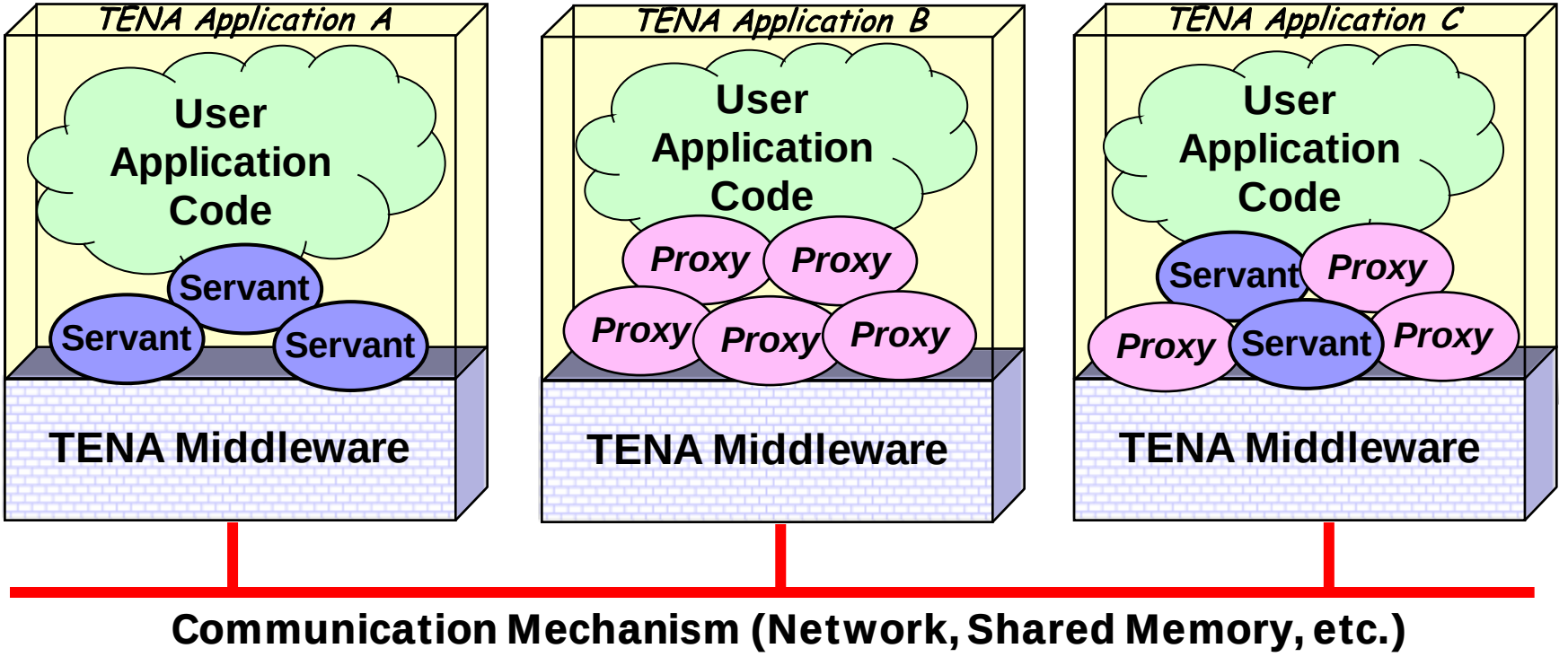
Logical Range Simple Example

TENA specifies an architecture for range resources participating in logical ranges



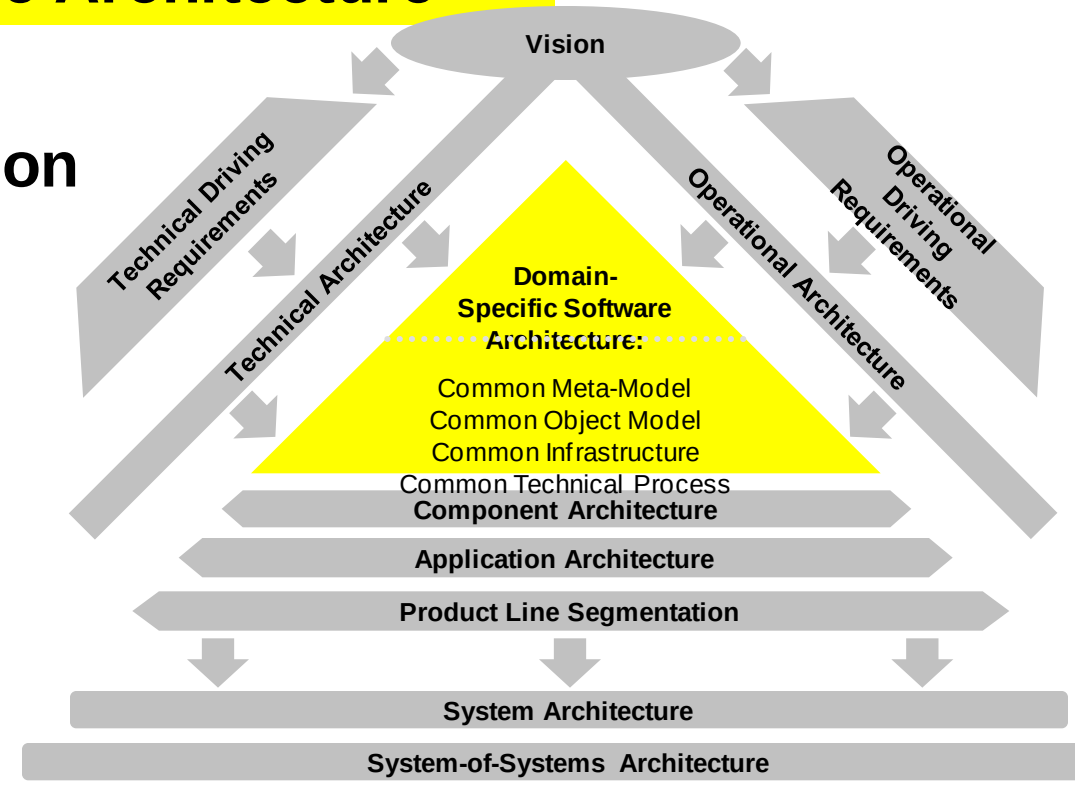
Logical Range Simple Example

- TENA specifies a peer-to-peer architecture for logical ranges:
  - Applications can be both clients and servers simultaneously
  - In their role as servers, applications serve TENA objects called "servants"
  - In their role as clients, applications obtain "proxies," representing other applications' servants. Only servers can write to their servant objects' publication state
- The TENA Middleware, the TENA objects, and the user's application code are compiled and linked together



TENA Organization

- Technical Driving Requirements
- Operational Driving Requirements
- Technical Architecture View
- Operational Architecture View
- **Domain-Specific Software Architecture**
- Application Architecture
- Product Line Segmentation



Domain-Specific Software Architecture – What is it?

- **Common Meta-Model**
  - What are "TENA Classes" and "TENA Objects?" (i.e., what are SDOs?)
  - What features do these objects have?
- **Common Object Model**
  - What are the standard TENA Classes?
  - It is a standard language for semantic interoperability
- **Common Infrastructure**
  - How are the TENA Objects managed and communicated?
  - Must support entire range event lifecycle
- **Common Technical Process**
  - What are the basic processes for initiating, conducting, and finishing communication about TENA objects?
  - Focused on the **technical** processes, not operational processes

What is a Meta-Model, and Why is it Important?

- **What is a Meta-Model?**
  - A meta-model is "a model that defines an abstract language for expressing other models," from Common Warehouse Metamodel specification by Dr. Daniel T. Chang.
  - All computer languages have meta-models
  - The TENA Meta-Model describes the features of objects defined in an LROM
- **Why is it important?**
  - The TENA Meta-Model is the architectural construct that specifies both the rules for defining an LROM and the requirements for the middleware

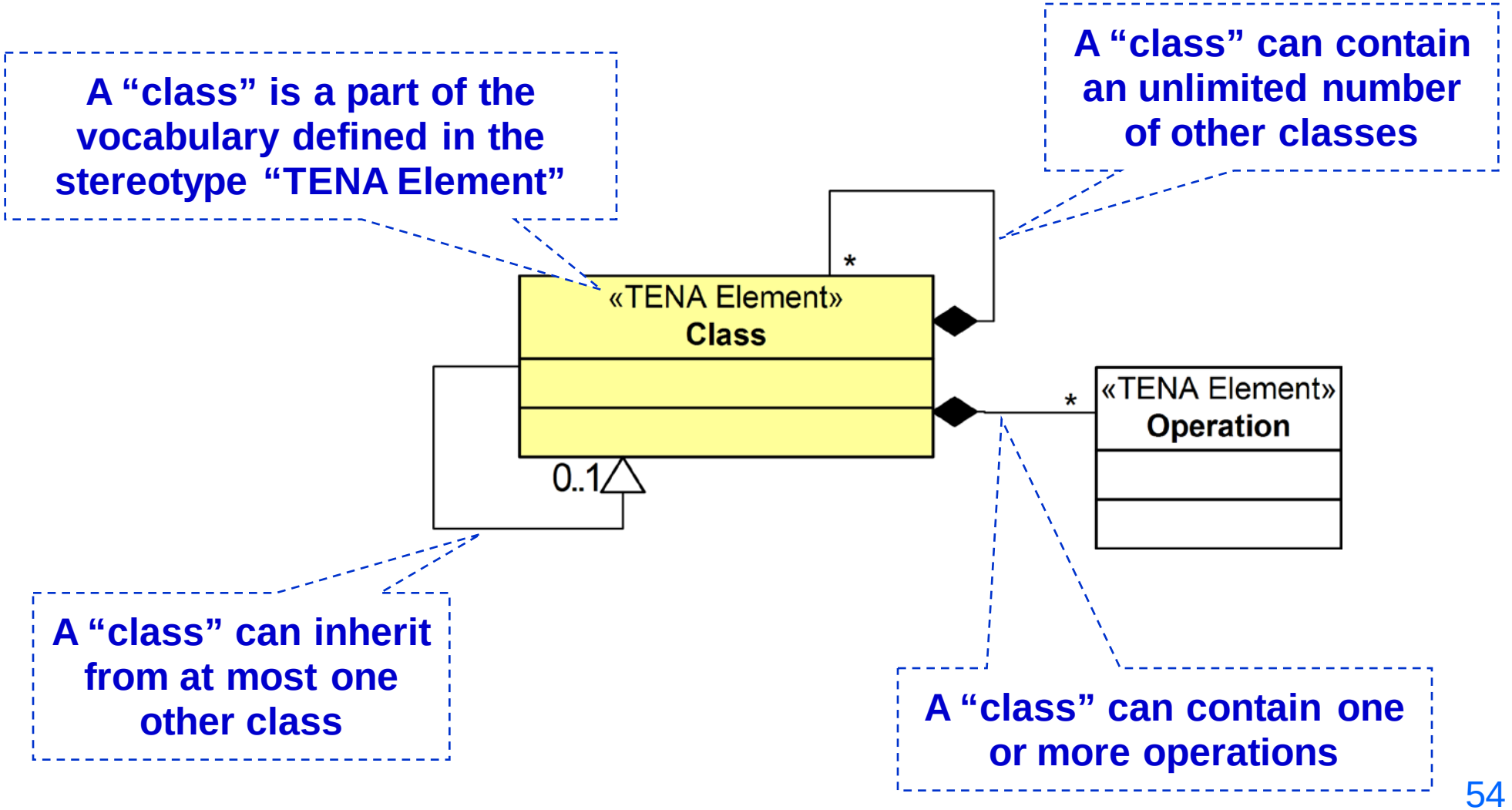
Every Computer Language Has A Meta-Model

(…and They're All Different)

- **C++**
  - Classes, structs == classes, abstract base classes, multiple inheritance, composition, generics, functions, methods, operators, fundamental types, exceptions, arrays, etc.
- **Java**
  - Classes, interfaces, exceptions
  - No structs, no functions, no generics, no multiple inheritance
- **CORBA IDL**
  - Interfaces, structs, valuetypes, sequences, enumerations, multiple inheritance of interfaces, unions
  - No classes
- **HLA**
  - Classes (objects), interactions, attributes, single inheritance
  - No interfaces, no composition, no functions/methods, no …

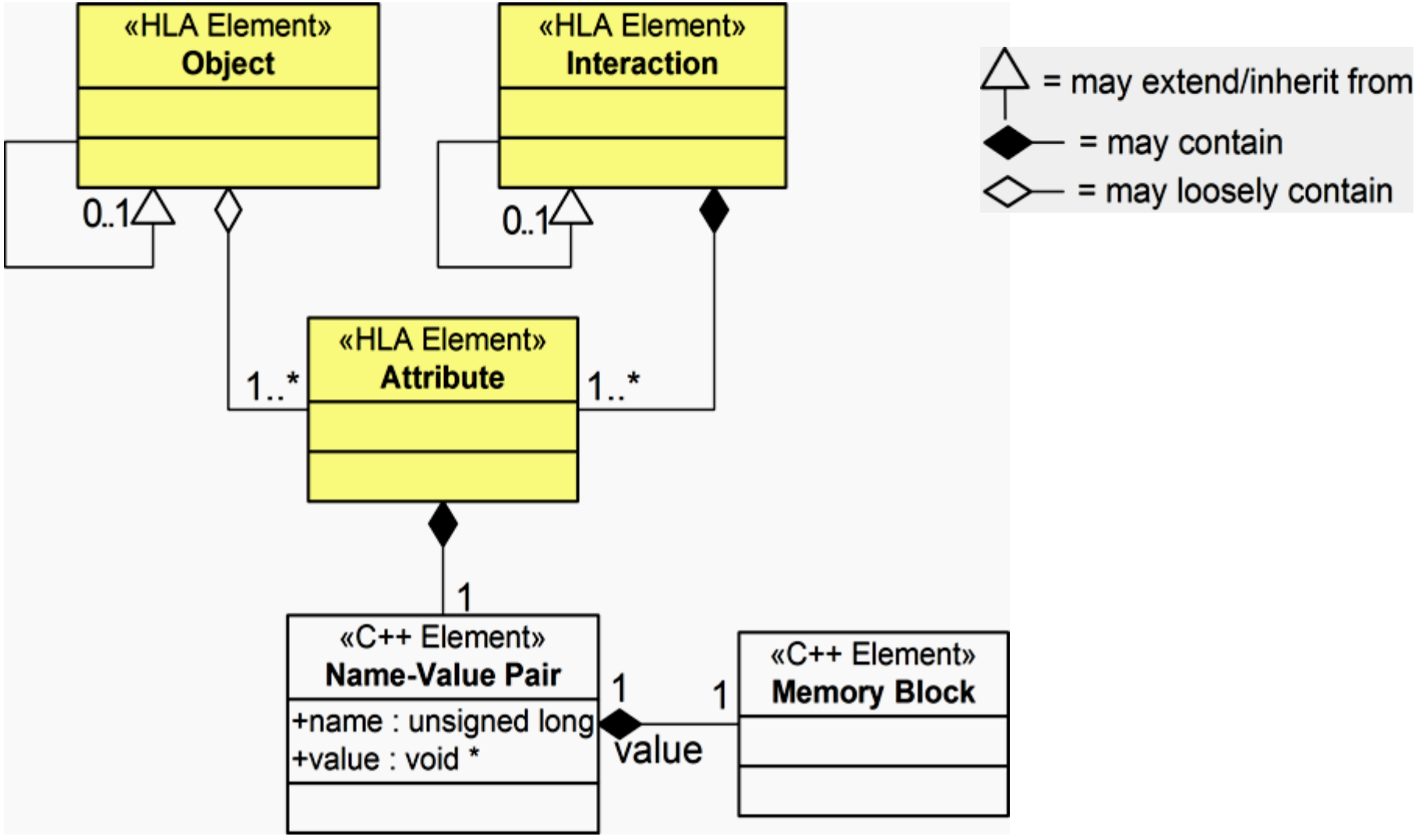
Representing a Meta-Model

- **"Pseudo-UML" is used, since formal UML is not as compact or communicative**



HLA Meta-Model
(with C++ additions)

- **Based on the HLA Object Model Template (OMT)**



Deficiencies of the HLA Meta-Model

- **Interpreted nature of attributes/parameters leads to serious engineering problems**
- **Structures are not marshaled/de-marshaled**
- **HLA does not support composition (objects containing other objects)**
- **HLA meta-model does not support:**
  - Remote-method invocation
  - Native support with tailored Quality-of-Service for data streams such as voice, video, or telemetry
  - Interfaces, user-defined exceptions, etc.

Requirements for Defining the TENA Meta-Model

- **Must support distributed computing**
- **Must be rich enough in features to support the object modeling needs of the entire test and training range community**
  - Objects and Messages
- **Must provide a semantic unification of information amenable to the creation of a simple, yet powerful, standard TENA Object Model**
- **Must be as easy to use and understand as possible given the above requirements**

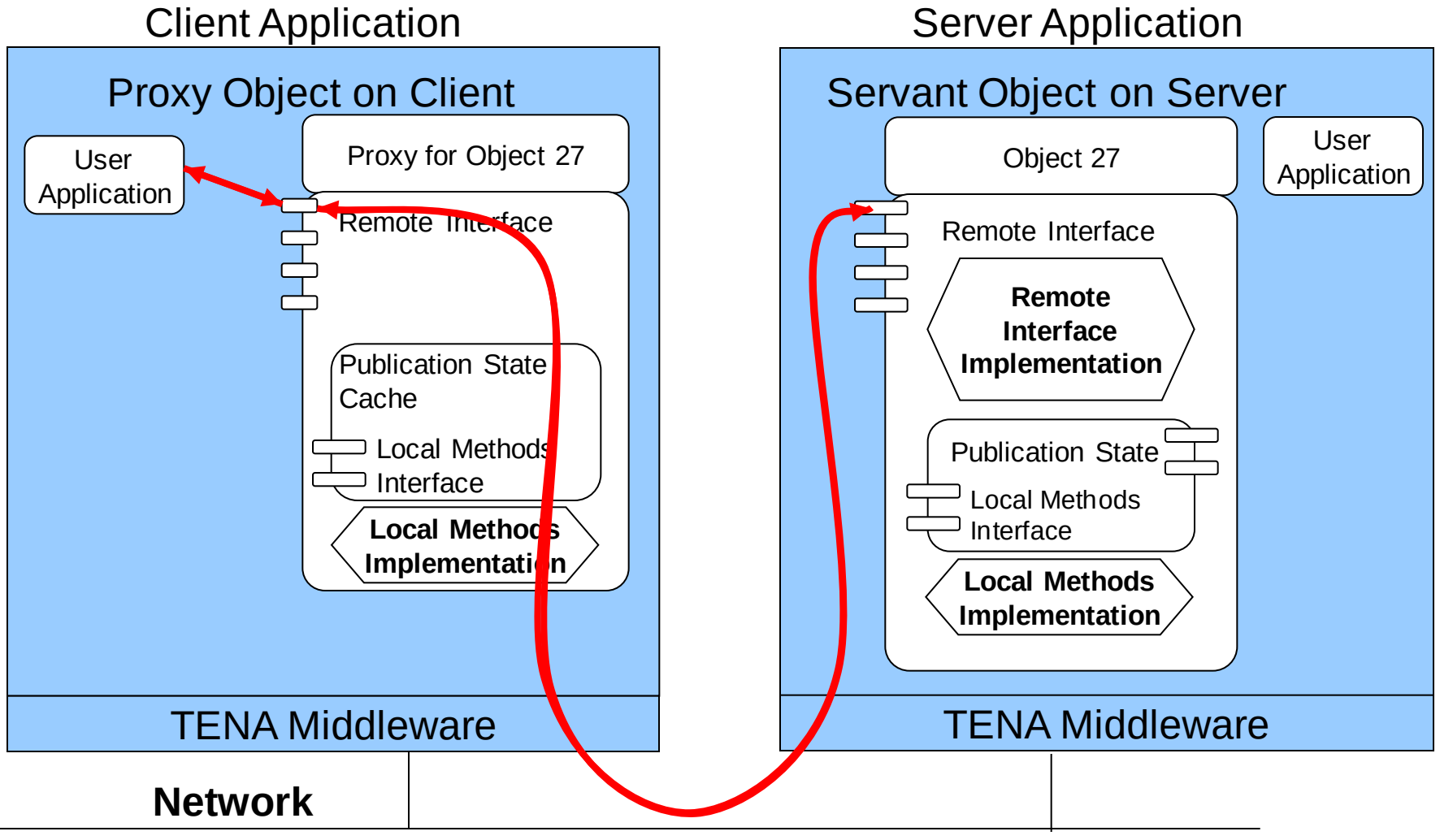
These requirements led to the invention of the Stateful Distributed Object, combining the best features of CORBA and the HLA in one easy-to-use concept

Stateful Distributed Objects

- **An SDO is a combination of two powerful concepts:**
  - a distributed object paradigm (like the one used in CORBA)
  - a distributed publish and subscribe paradigm
- **Benefits of this combination:**
  - A conventional distributed object-oriented system offers no direct support to the user for disseminating data from a single source to multiple destinations
  - A conventional publish-subscribe system does not provide the abstraction of objects with a set of methods in their interface
  - Interface to SDOs is a lot simpler and more usable than the combination of interfaces to their underlying technologies

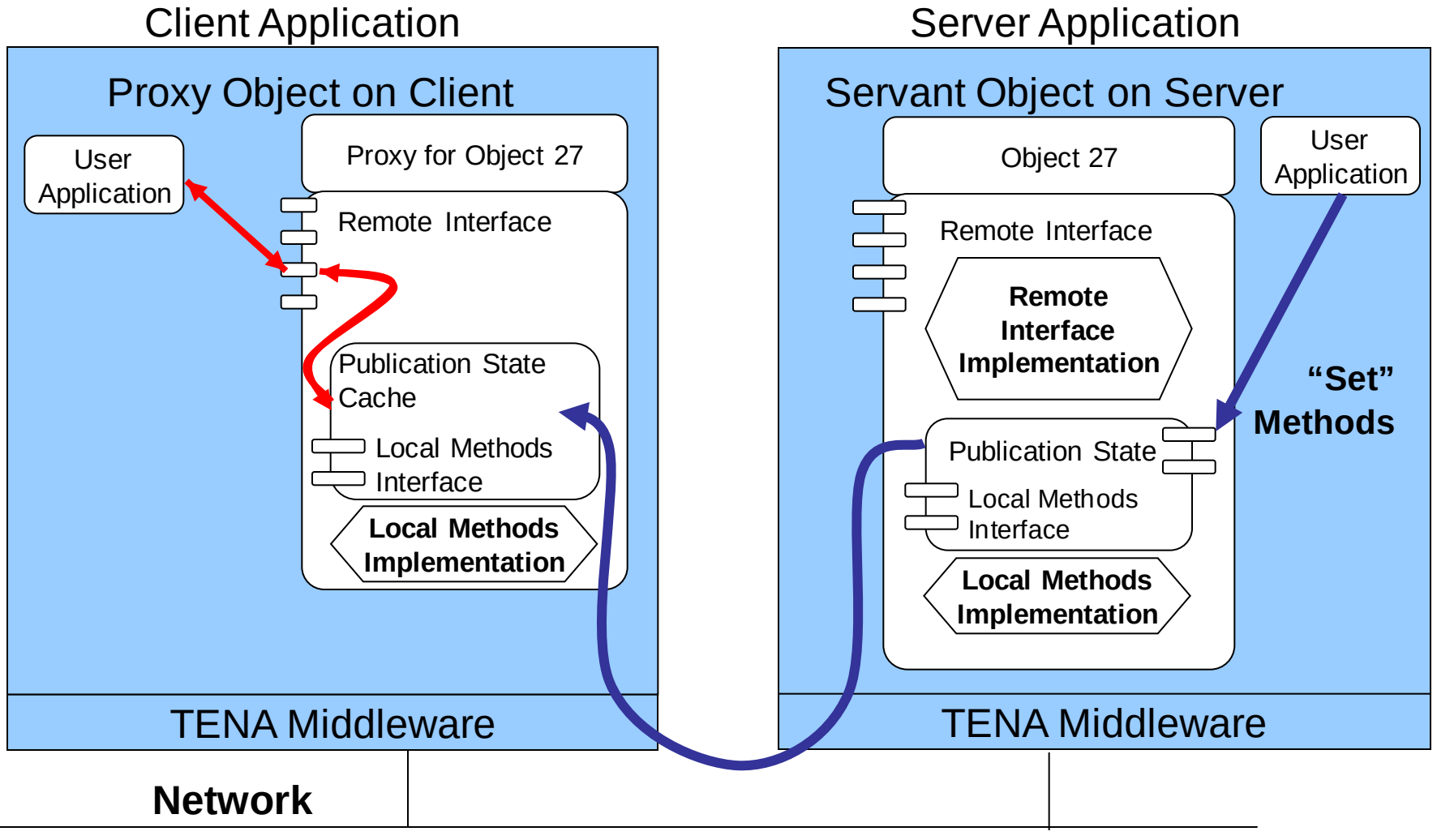
Clients and Proxies, Servers and Servants

- **Remote Method Invocation**



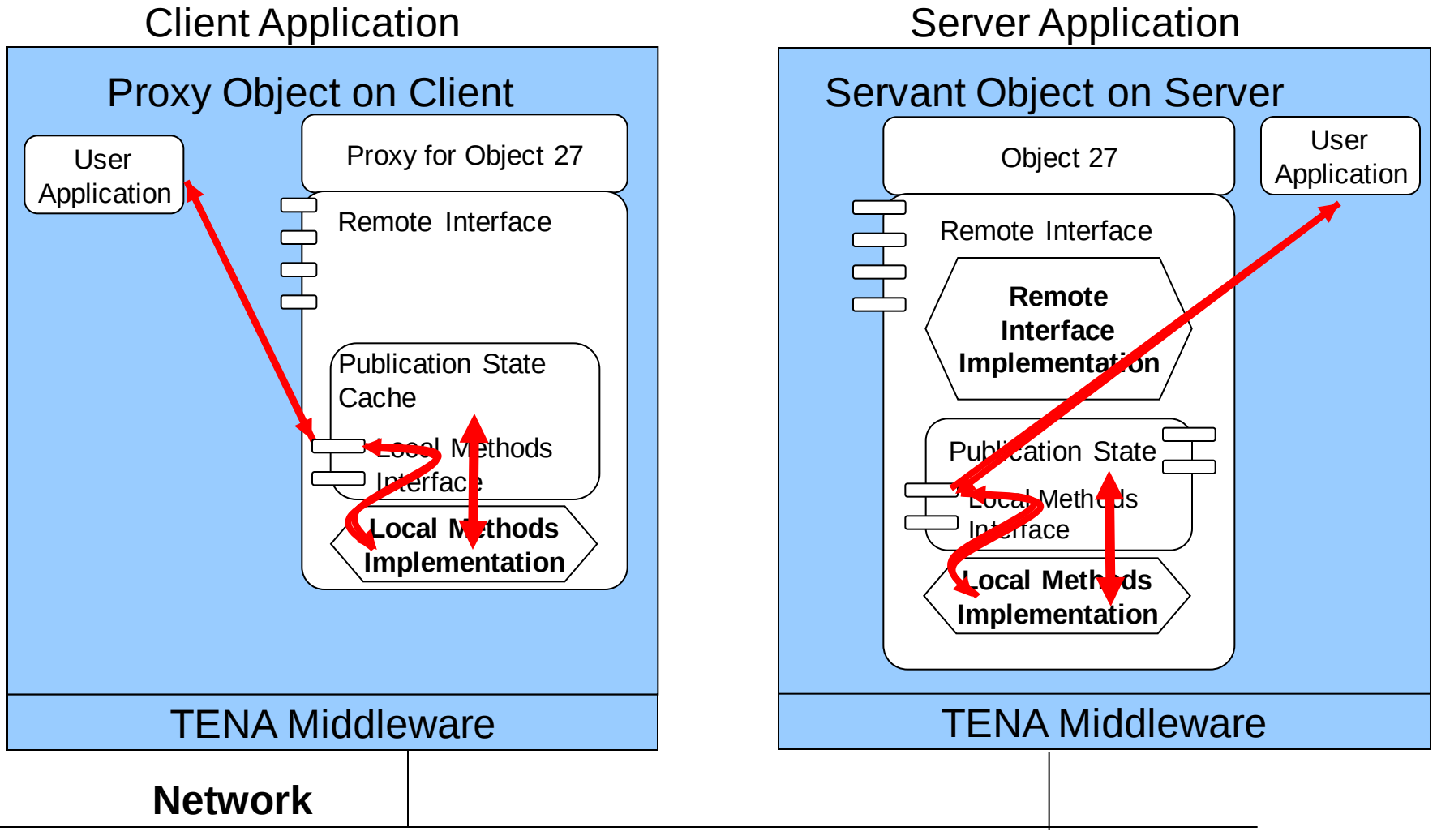
Clients and Proxies, Servers and Servants

- **Publication State Dissemination and Access**



Clients and Proxies, Servers and Servants

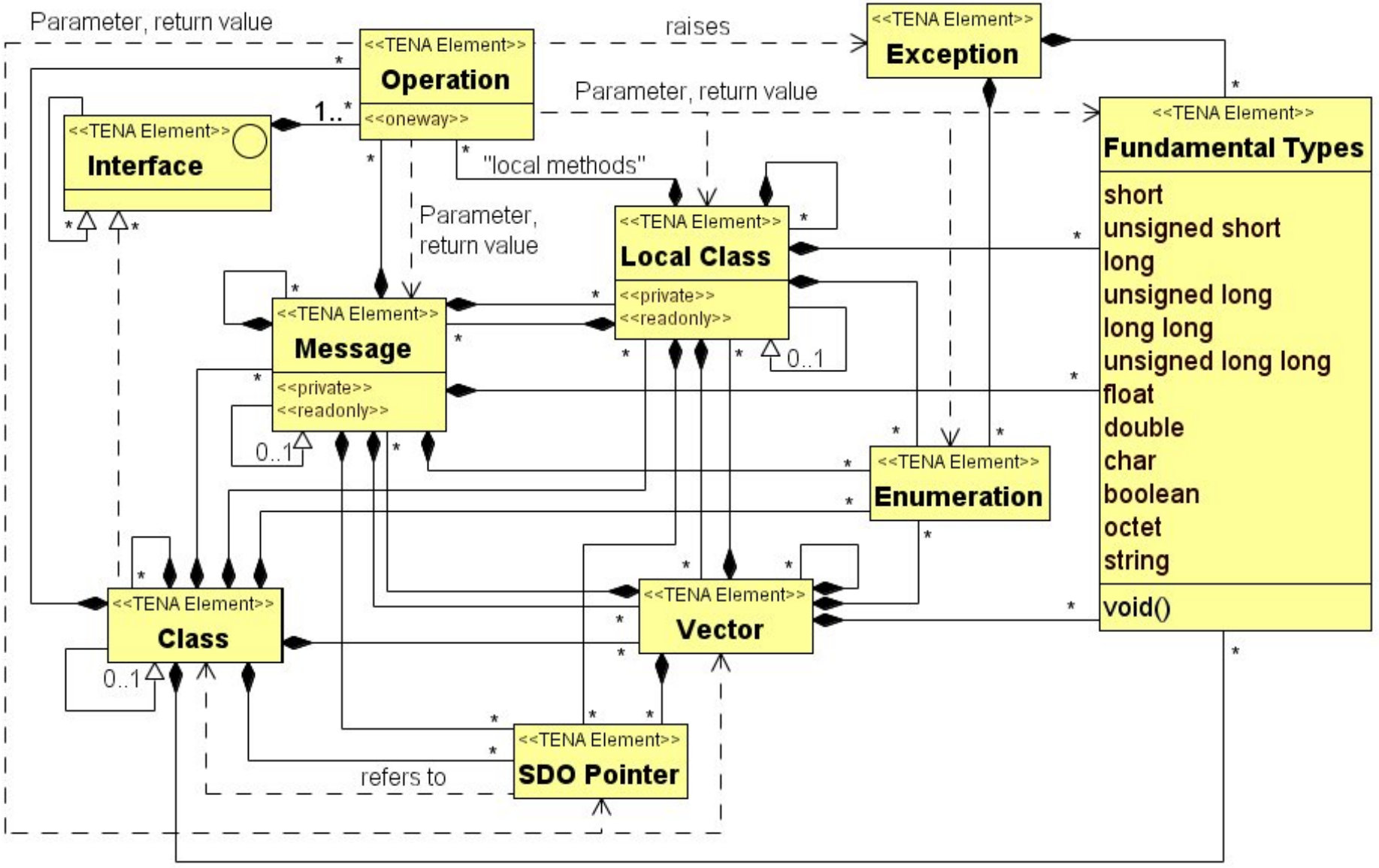
- **Local Methods used on both Client and Server**



Local Classes

- **The concept of local methods are implemented in what are called "local classes"**
- **Local classes are simply classes that get moved in their entirety (identity, state, and behavior) from servers to clients**
- **Local classes can be contained in SDOs**
- **A "message" is a special type of local class that can be sent from an application to any subscribing applications**
  - Messages can contain other messages as well as contain local classes

TENA Meta-Model Release 5.2.2



TENA Objects are Compiled In

- **Why use compiled-in object definitions?**
  - **Strong type-checking**
    - Don't wait until runtime to find errors that a compiler could detect
  - **Performance**
    - Interpretation of methods/attributes has significant impact
  - Ability to easily handle complex object relationships
  - Conforms to current best software engineering practices
- **How do you support compiled-in object definitions?**
  - Use a language like CORBA IDL to define object interface and object state structure
  - Use **code generation** to implement the required functionality
- **Thus the concept of the TENA Definition Language (TDL) was created**
  - Very similar to IDL and C++

Sample OM in TDL

```
package OMsample
{
  local class Time
  {
    unsigned long nanoseconds;
    long seconds;
  };

  local class Position
  {
    double x;
    double y;
    double z;
  };

  local class Identifier
  {
    string name;
    string type;
    unsigned long ID;
    string convertToString();
  };

  class Platform
  {
    Identifier ident;
    double fuel;
    Time time;
    Position position;
  };

  message LocationMessage
  {
    Identifier ident;
    Position location;
  };

};
```

Domain-Specific Software Architecture – What is it?

- **Common Meta-Model**
  - What are "TENA Classes" and "TENA Objects?" (i.e., what are SDOs?)
  - What features do these objects have?
- **Common Object Model**
  - What are the standard TENA Classes?
  - It is a standard language for semantic interoperability
- **Common Infrastructure**
  - How are the TENA Objects managed and communicated?
  - Must support entire range event lifecycle
- **Common Technical Process**
  - What are the basic processes for initiating, conducting, and finishing communication about TENA objects?
  - Focused on the **technical** processes, not operational processes

The Logical Range Object Model

- **A Logical Range Object Model (LROM) consists of those object definitions, derived from whatever source, that are used in a given logical range execution to meet the immediate needs and requirements of a specific user for a specific range event**
- **The LROM is the common object model shared by all TENA resource applications in a logical range**
- **The concept of an LROM is necessary because it will not be possible to create the entire standard TENA Object Model before the first logical range is created.**
  - As time progresses, each LROM will contain more standard elements and fewer elements that are chosen on an *ad hoc* basis
- **TENA must be deployable *gradually* – the LROM concept supports this requirement**

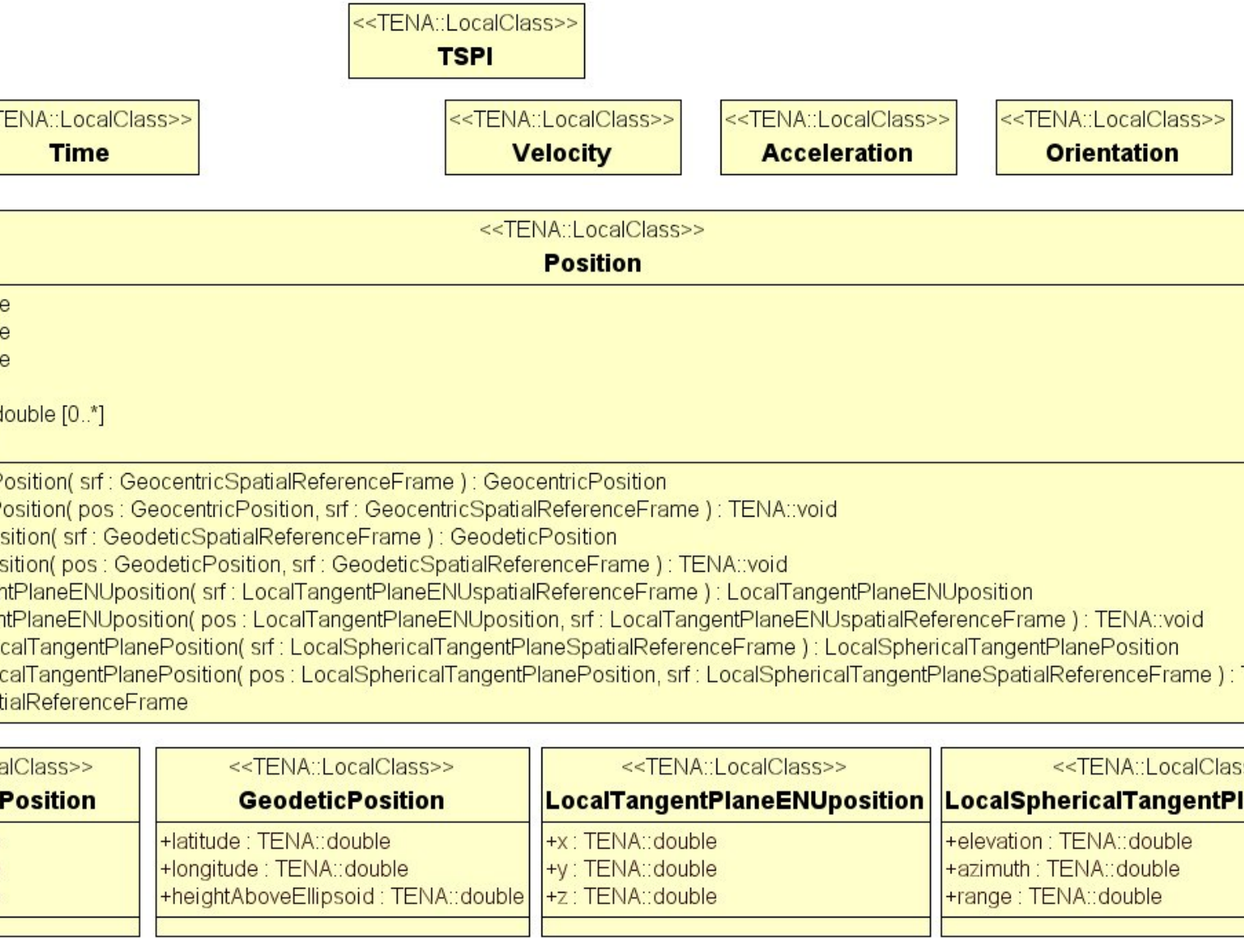
The Standard TENA Object Model

- **To enable semantic interoperability among range resource applications**
- **To provide the "common language" that all range resource applications use to communicate**
  - It will eventually encode almost all information communicated among range resource applications
- **Object Model Stages**
  - **User-Defined Objects** – objects defined solely for the purpose of a given logical range by TENA users
  - **Candidate Objects** – objects defined as potential standards, which are undergoing test and evaluation by the community prior to standardization
  - **TENA Standard Objects** – objects which have been approved for standardization by the AMT

TENA Standard Object Models

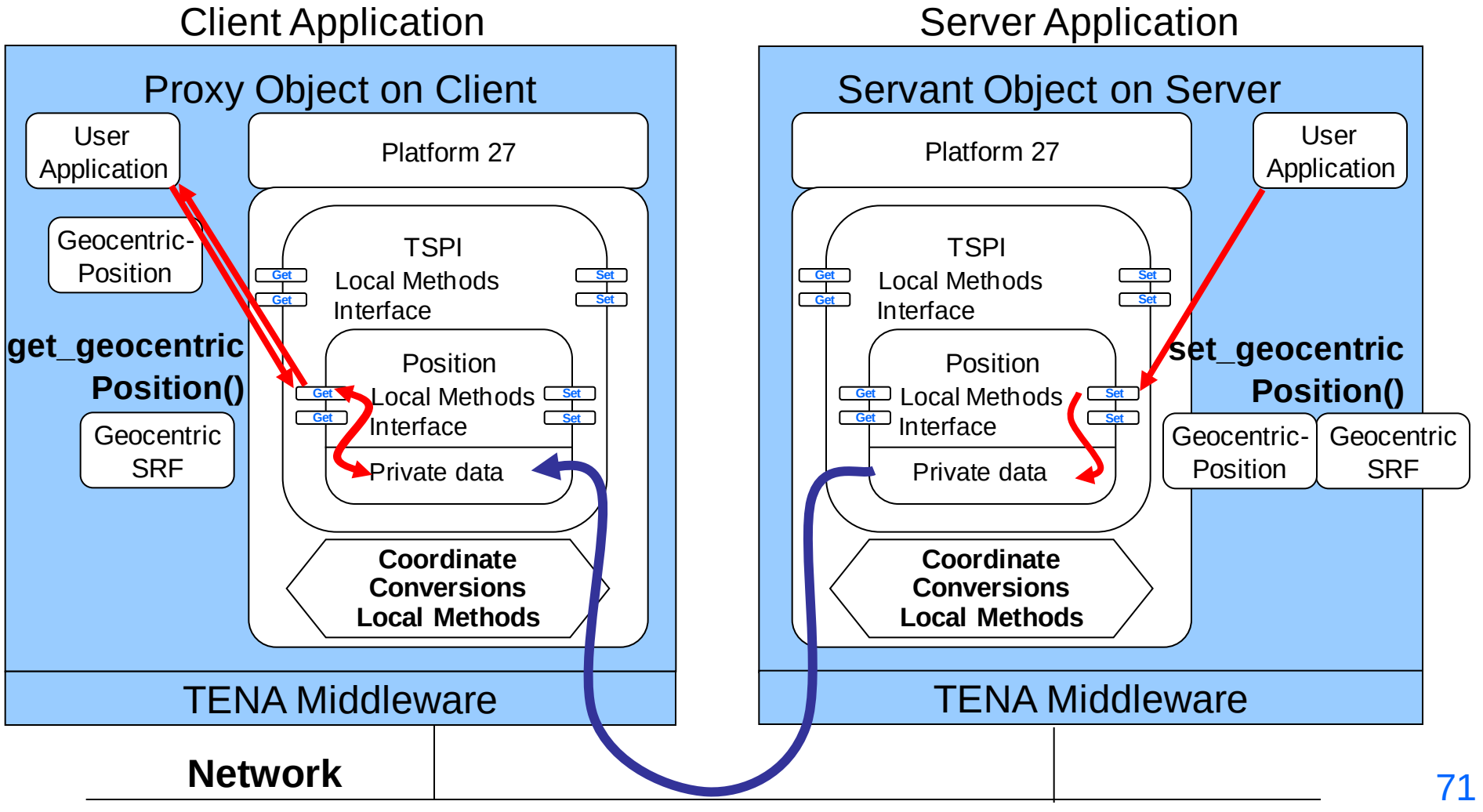
- **TENA-Platform:**
  - TENA-Platform-v3.1
  - TENA-PlatformDetails-v3
  - TENA-Affiliation-v1
  - TENA-UniqueID-v2
  - TENA-PlatformType-v1
  - DIS-EntityType-v2
  - TENA-Munition-v2.1
  - TENA-Engagement-v3.1
  - TENA-Organization-v1
  - TENA-EmbeddedSystem-v2
  - TENA-EmbeddedSensor-v2
  - TENA-EmbeddedWeapon-v2
- **TENA-AMO:**
  - TENA-AMO-v1
- **TENA-TSPI:**
  - TENA-TSPI-v4
  - TENA-Time-v1.1
  - TENA-Position-v1
  - TENA-Velocity-v1
  - TENA-Acceleration-v1
  - TENA-Orientation-v1
  - TENA-AngularVelocity-v1
  - TENA-AngularAcceleration-v1
  - TENA-ORM-v1
  - TENA-SRF-v1
  - TENA-SRFserver-v1
- **TENA-Radar-v2**
- **TENA-GPS-v2**

TENA-TSPI-v4
(TENA SDA Supported)



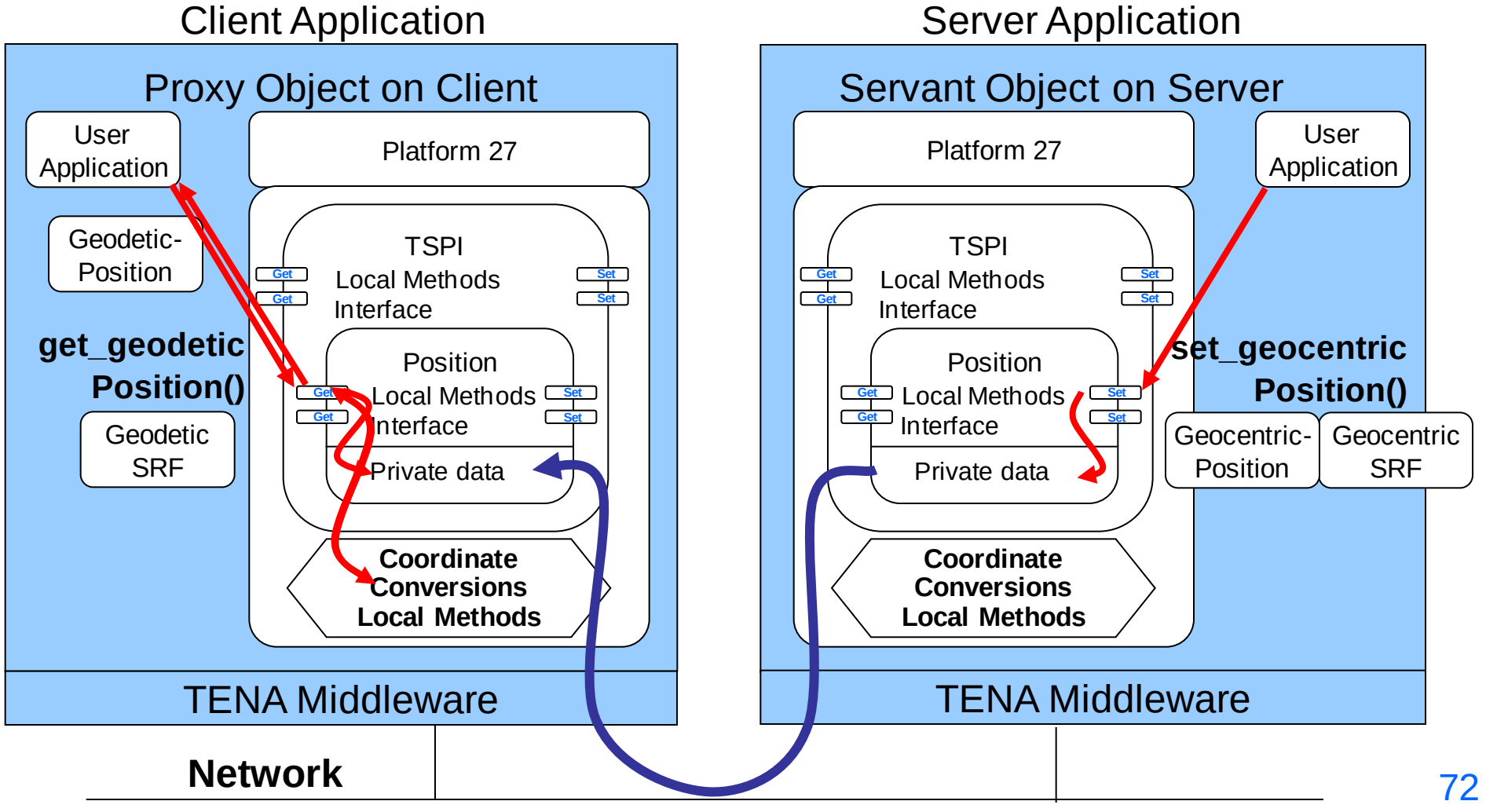
TSPI v4 with Coordinate Conversions

- **Case 1: Reading and writing in the same coordinate system**

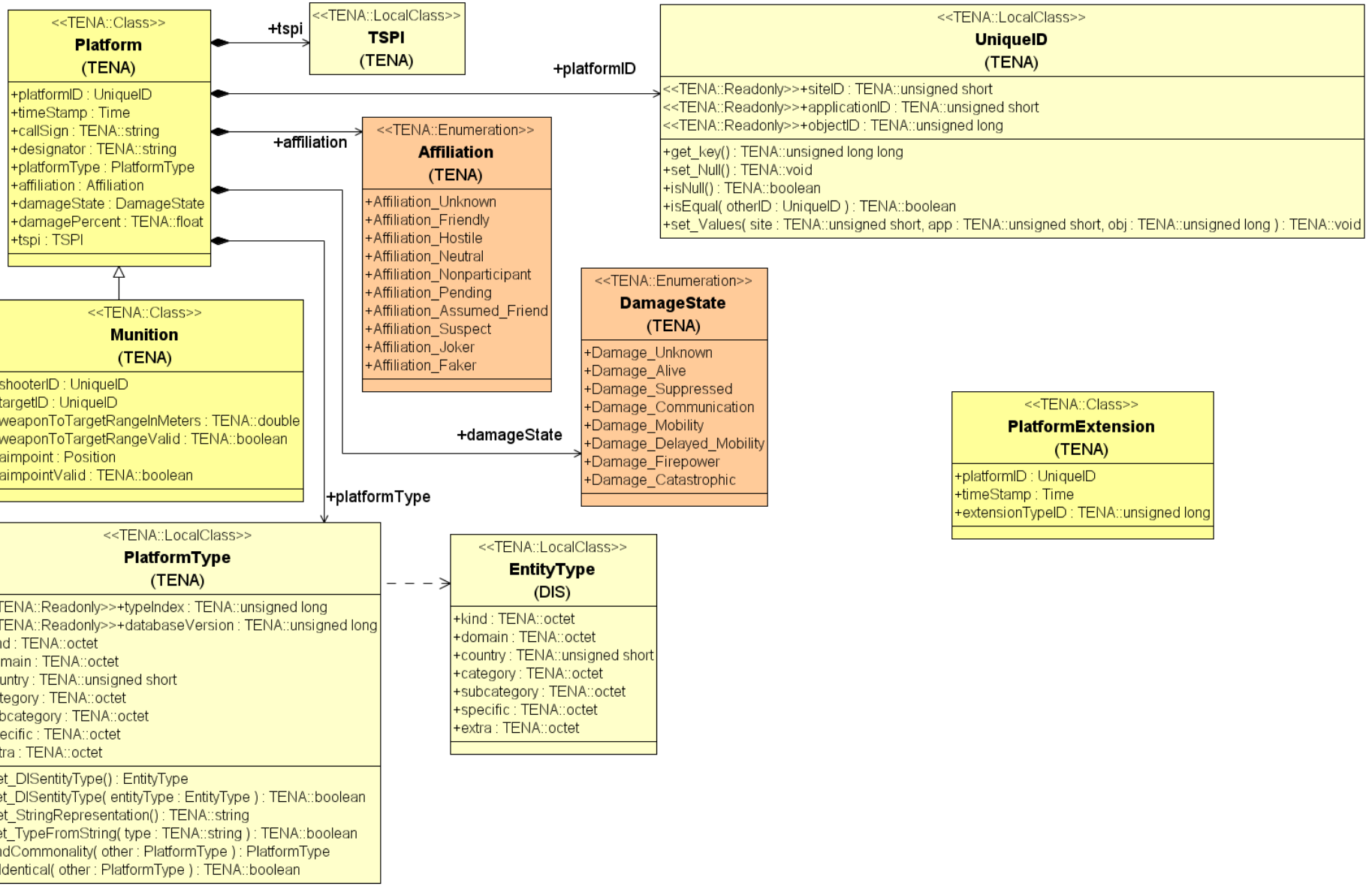


TSPI v4 with Coordinate Conversions

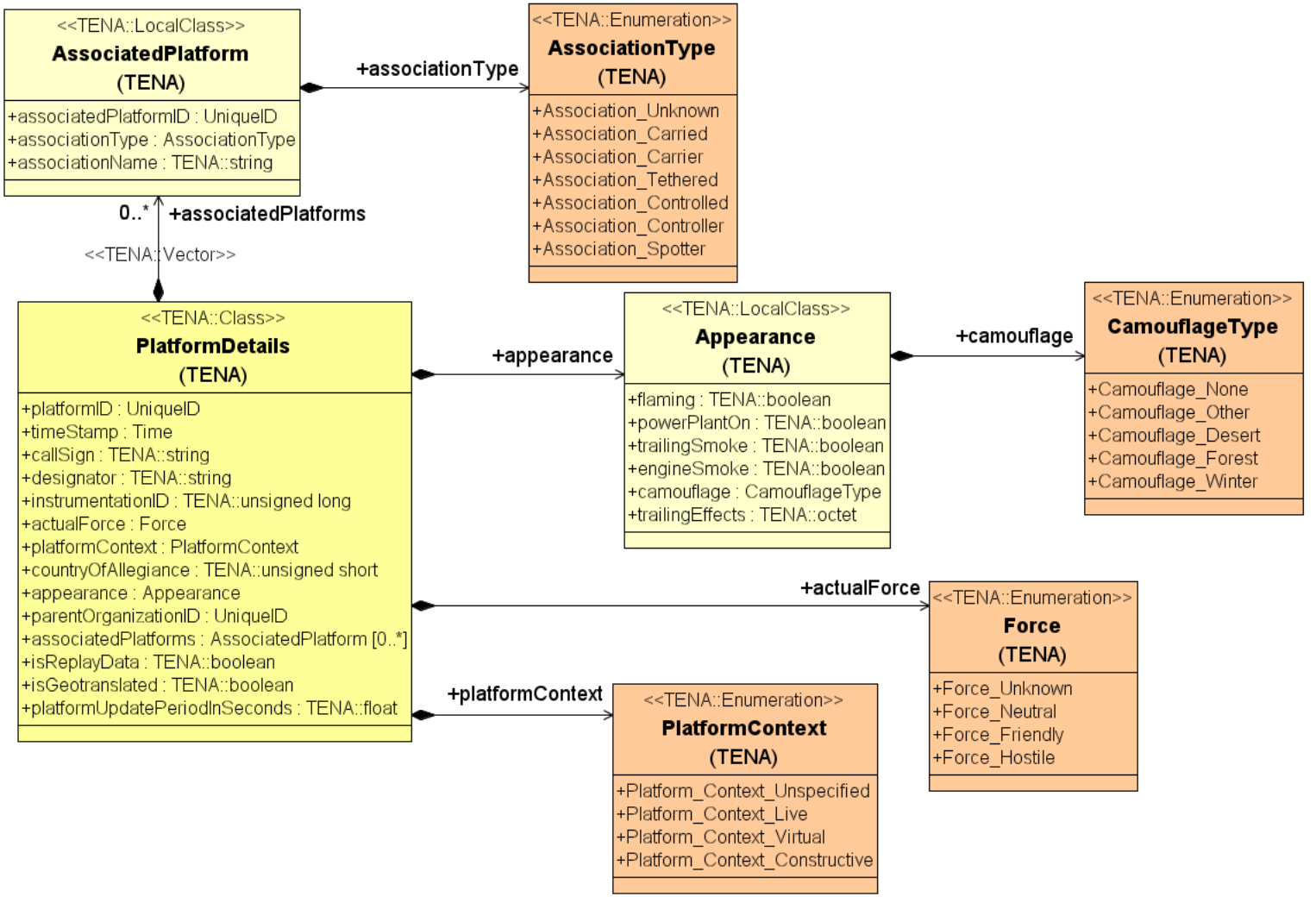
- **Case 2: Reading and writing in different coordinate systems**
  - Write in Geocentric (ECEF), read in Geodetic (latitude/longitude/altitude)



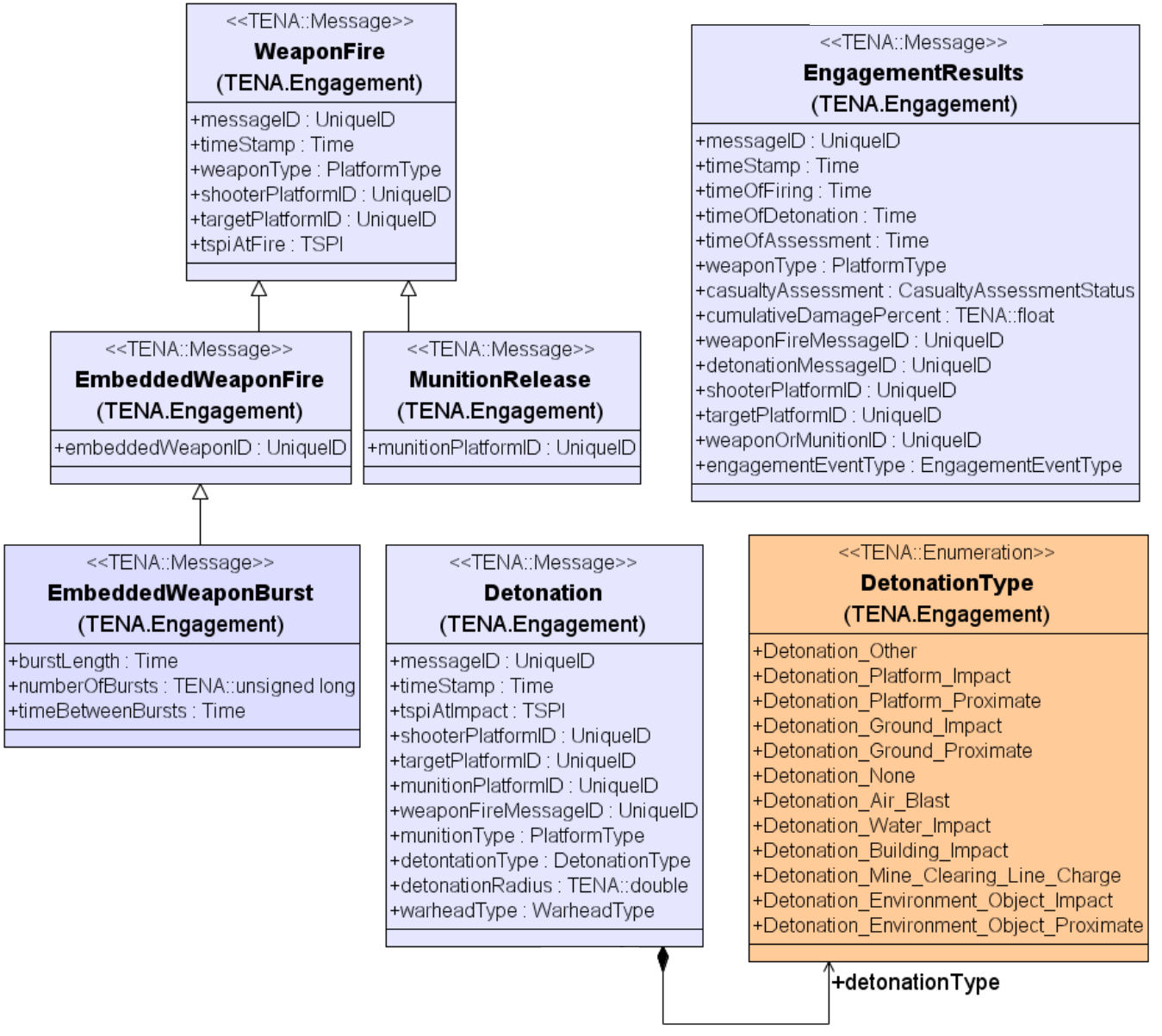
TENA-Platform-v3.1
(Current TENA Standard)



TENA-PlatformDetails-v3
(Current TENA Standard)



TENA-Engagement-v3.1
(Current TENA Standard)



Object Model Refinements

- **Many changes based on feedback from users will be implemented coincident with Middleware R6**
- **The TDL files (components) will be reduced to the following:**
  - TENA-TSPI-v5.tdl (includes all the TSPI-v4 components and the new time representations)
  - TENA-AMO-v2.tdl
  - TENA-MMO-v1.tdl
  - TENA-Platform-v4.tdl
  - TENA-PlatformDetails-v4.tdl
  - TENA-PlatformType-v2.tdl
  - TENA-UniqueID-v2.tdl
  - TENA-Munition-v3.tdl
  - TENA-EmbeddedSystem-v3.tdl
  - TENA-Engagement-v4.tdl
  - TENA-Exercise-v1.tdl
  - TENA-Radar-v3.tdl
  - TENA-GPS-v3.tdl
- **All changes have been coordinated with and approved by the AMT**
- **For more details see web site**

Web-Based Code Generation

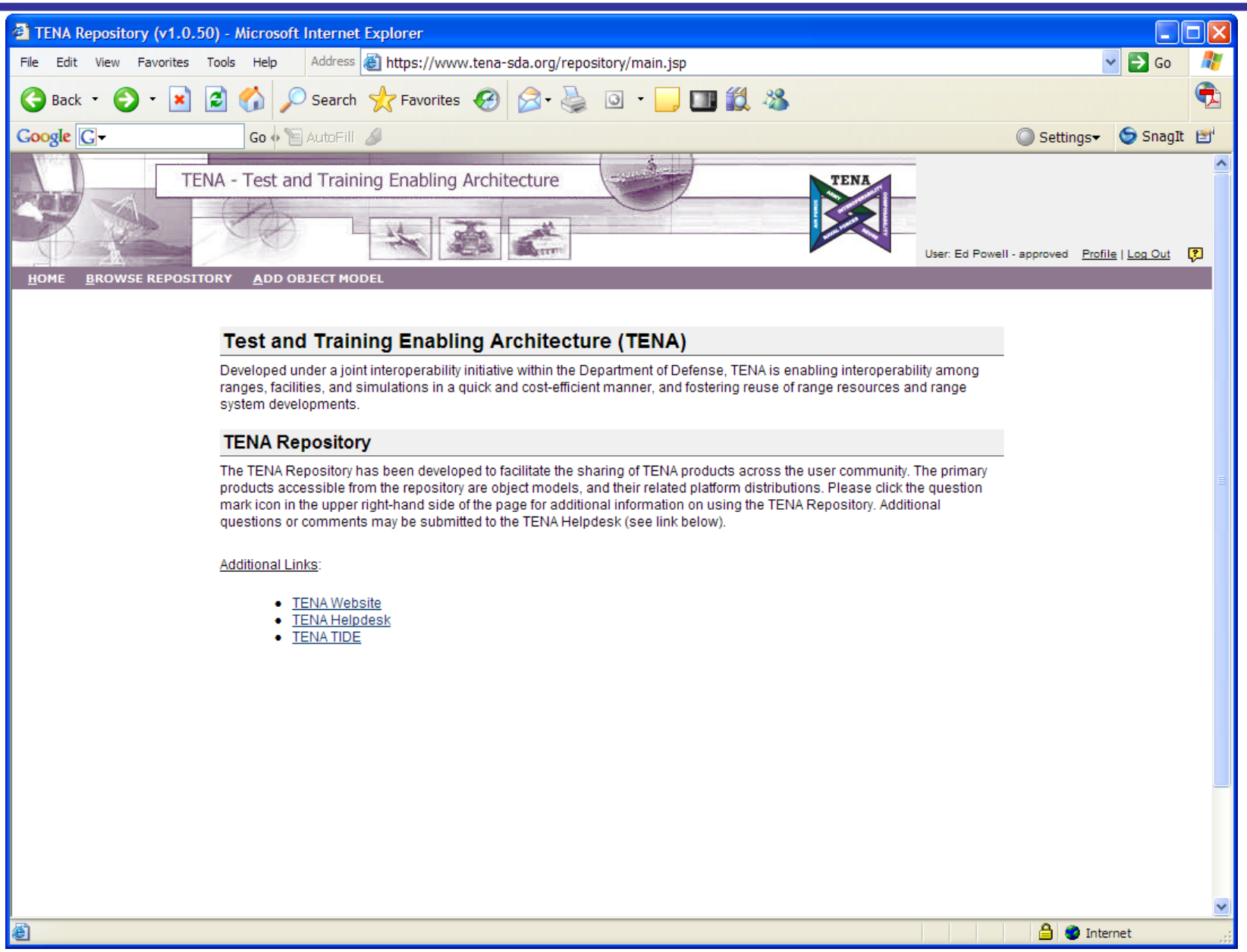
- **TDL-to-C++ compiler uses a Web front end, because it:**
  - Allows bug fixes and additions to the code generator without having to re-disseminate it to the community
  - Allows AMT to collect information on object models being designed so progress can be made toward the standard TENA Object Model
  - Allows collaboration with users on their object model designs
  - Allows code generator to be written for less than the full complement of TENA Middleware platforms, if necessary

Object Model Distributions

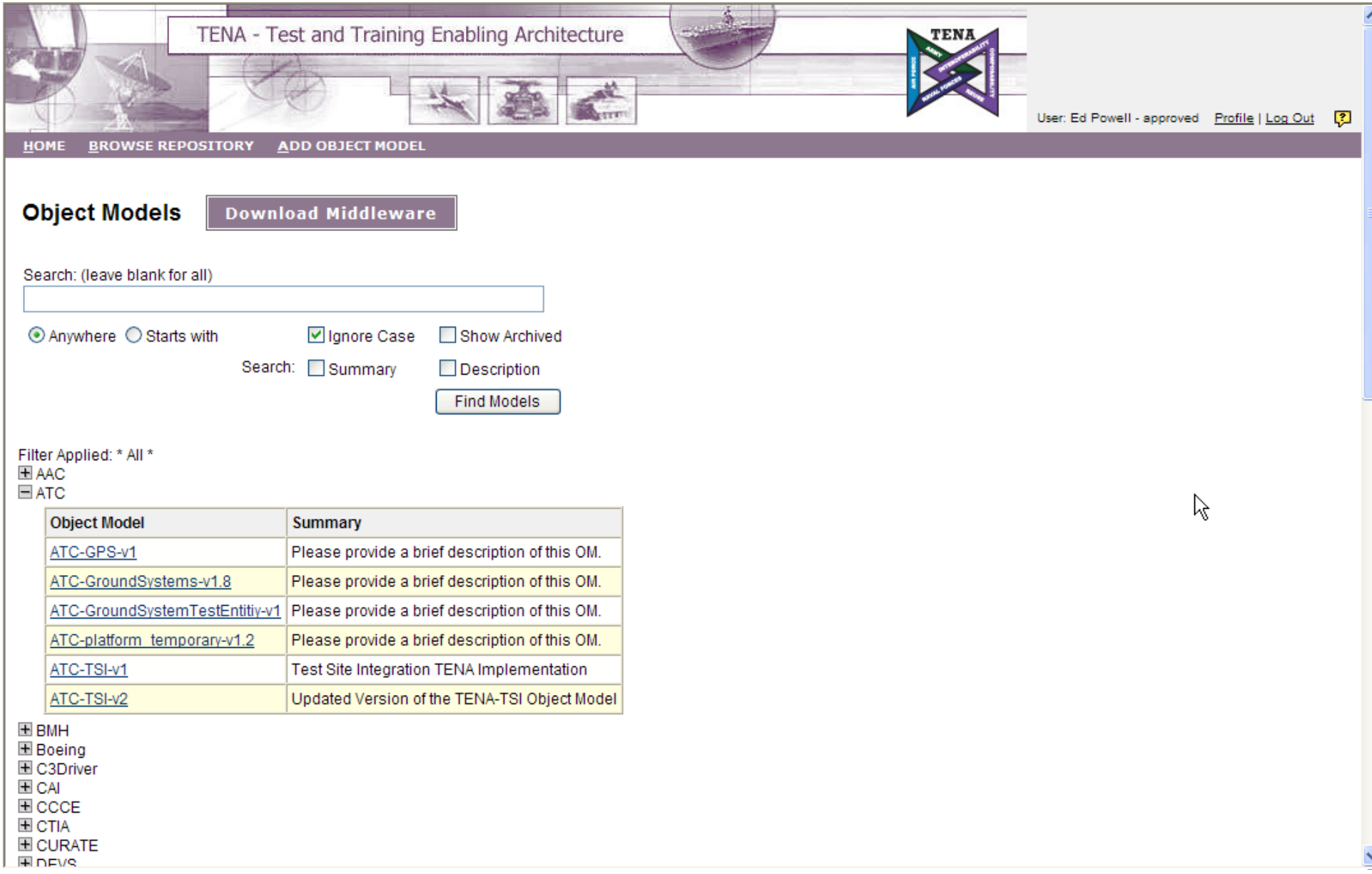
- **Two Types of Object Model Distributions**
  - *Object Model Definition* – specifies the types (e.g., classes, messages, enums) and their interface signatures and/or attributes
  - *Object Model Implementation* – Provides executable code that adheres to a particular definition
- **Object Model Components**
  - Object model definitions can "import" other definitions
  - Applications are required to install every object model definition and any pre-built implementations being used
  - Namespace changes with pre-built implementations complicates the automatic generation of "BasicImpl" applications
- **OM Distribution Bundles**
  - Currently developed mechanism for TENA Repository to bundle imported definitions and available implementations into a single downloadable file
  - Need to expand on this capability to automatically install all of the individual components

Web Site OM Support

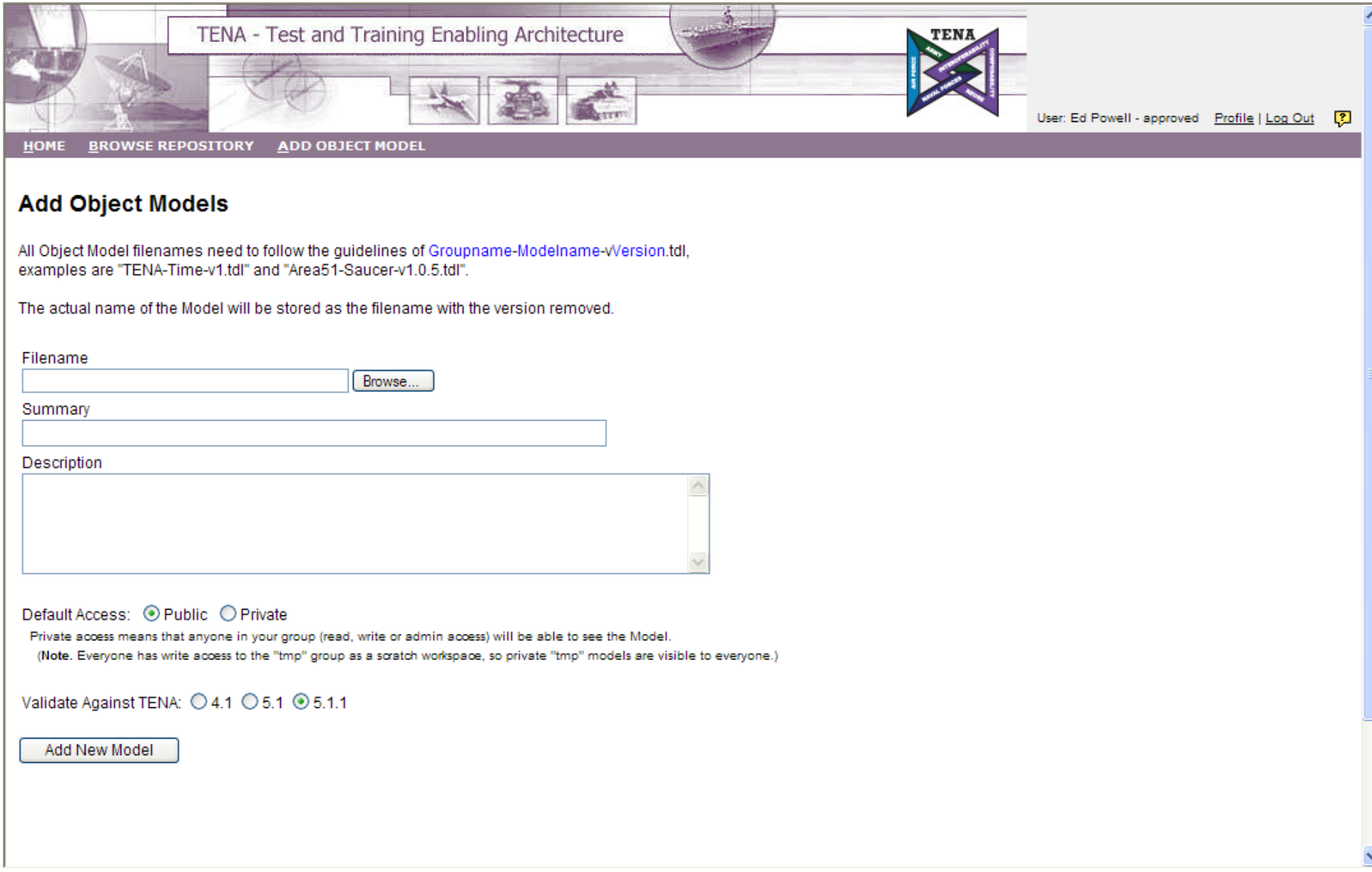
http://www.tena-sda.org/repository



Browse Repository



Upload TDL Files



Download Model Definition

Model Details

Model Name:	DIS-EntityType-v2
Access:	Public
Summary:	Version 2 of the DIS EntityType object model.
Description:	This is version 2 of the DIS EntityType object model developed by the TENA SDA project. This object model is based on the DIS standard, IEEE-1278.x. More detailed documentation of this object model is available in the TENA Wiki.
TDL File:	models/DIS/EntityType-v2/DIS-EntityType-v2.tdl View TDL
Uploaded by:	J. Russell Noseworthy on 09/23/2005 15:43:41

⊞ Attachments

Imports - *none*

Distributions: TENA-v5.1.1

⊞ *Select visible platforms*

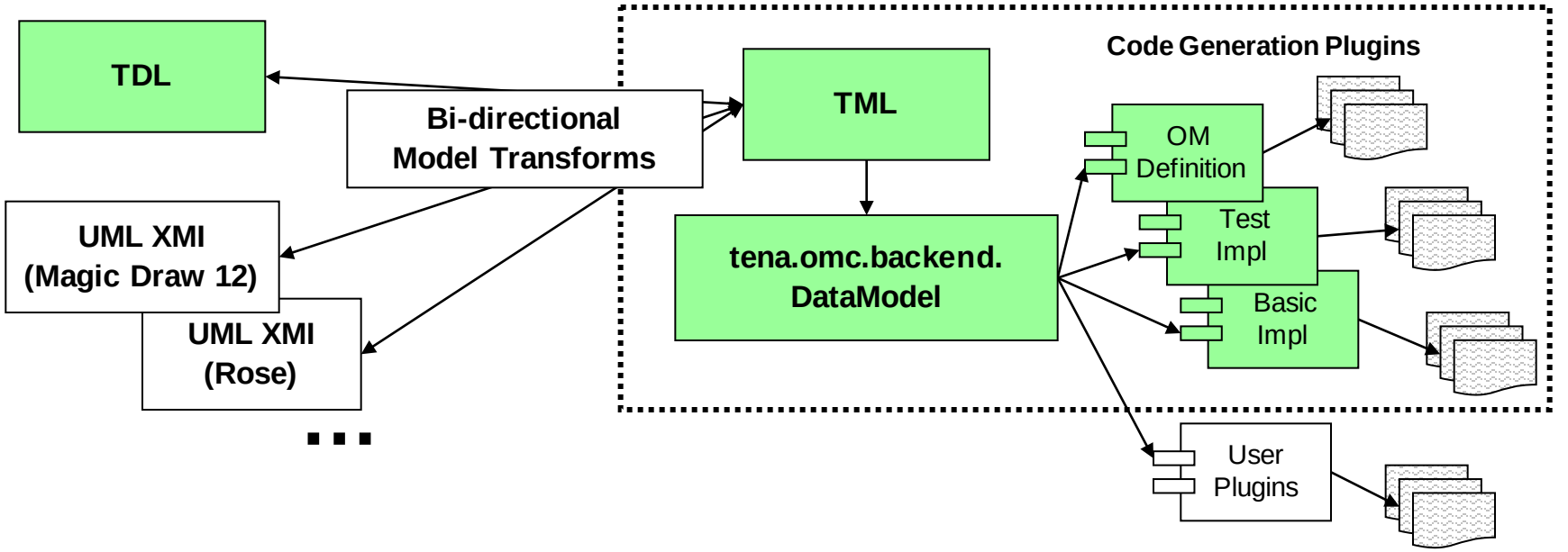
Distribution	Distribution State	Platform
Definition	Download (279.921KB) details	fc3-gcc34-d
Definition	Download (512.101KB) details	fc3-gcc34
Definition	Download (282.036KB) details	fc4-gcc40-d
Definition	Download (486.919KB) details	fc4-gcc40
Definition	Download (225.12KB) details	irix65-mipspro742m-d
Definition	Download (179.097KB) details	irix65-mipspro742m
Definition	Download (282.517KB) details	rhelws4-gcc34-d
Definition	Download (514.784KB) details	rhelws4-gcc34
Definition	Download (1.144MB) details	rh8-gcc32-d

Remember: Need to Download Definition and Implementation

Distribution	Distribution State	Platform
Definition	Download (1.126MB) details	fc3-gcc34-d
Implementation Src	Download (26.322KB) details	fc3-gcc34-d
Definition	Download (1.395MB) details	fc3-gcc34
Implementation Src	Download (26.977KB) details	fc3-gcc34
Definition	Download (1.165MB) details	fc4-gcc40-d
Implementation Src	Download (27.82KB) details	fc4-gcc40-d
Definition	Download (1.388MB) details	fc4-gcc40
Implementation Src	Download (28.199KB) details	fc4-gcc40
Definition	Download (862.533KB) details	irix65-mipspro742m-d
Implementation Src	Download (26.621KB) details	irix65-mipspro742m-d
Definition	Download (554.146KB) details	irix65-mipspro742m
Implementation Src	Download (26.599KB) details	irix65-mipspro742m
Definition	Download (1.129MB) details	rhelws4-gcc34-d
Implementation Src	Download (27.309KB) details	rhelws4-gcc34-d
Definition	Download (1.399MB) details	rhelws4-gcc34
Implementation Src	Download (30.263KB) details	rhelws4-gcc34
Definition	Download (2.587MB) details	rh8-gcc32-d
Implementation Src	Download (26.175KB) details	rh8-gcc32-d
Definition	Download (2.664MB) details	rh8-gcc32
Implementation Src	Download (26.845KB) details	rh8-gcc32
Definition	Download (2.584MB) details	rh9-gcc322-d
Implementation Src	Download (26.165KB) details	rh9-gcc322-d
Definition	Download (2.661MB) details	rh9-gcc322
Implementation Src	Download (26.822KB) details	rh9-gcc322
Definition	Download (319.866KB) details	rh9-ppc82xxgcc322
Implementation Src	Download (26.506KB) details	rh9-ppc82xxgcc322

Auto-Code Generation With TENA

- **Our desire is for the input to the TENA auto-code generator be standard XMI (generated from UML)**
- **Challenges: XMI not yet implemented in a standard way by tool vendors, and current auto-code generation capability is based on TDL**
- **Current Interim Solution – Use MagicDraw plug-in to create TDL from UML**
- **Next Steps**
  - Implement TENA Metamodel in Eclipse Modeling Framework using ECore representation – define TENA Modeling Language (TML)
  - Create XMI ↔ TML, TDL ↔ TML translators
  - API and framework being developed to support various "code generation plugins" used to automatically create specialized code based on FreeMarker templates



Domain-Specific Software Architecture – What is it?

- **Common Meta-Model**
  - What are "TENA Classes" and "TENA Objects?" (i.e., what are SDOs?)
  - What features do these objects have?
- **Common Object Model**
  - What are the standard TENA Classes?
  - It is a standard language for semantic interoperability
- **Common Infrastructure**
  - How are the TENA Objects managed and communicated?
  - Must support entire range event lifecycle
- **Common Technical Process**
  - What are the basic processes for initiating, conducting, and finishing communication about TENA objects?
  - Focused on the **technical** processes, not operational processes

TENA Common Infrastructure

- **Components:**
  - Repository
  - Logical Range Data Archive
  - Middleware
- **Purpose:**
  - Provide the common, standardized, software mechanism that makes communication about objects in the TENA Object Model as efficient and simple as possible throughout the entire range event lifecycle

Why are These Components Necessary for the Common Infrastructure?

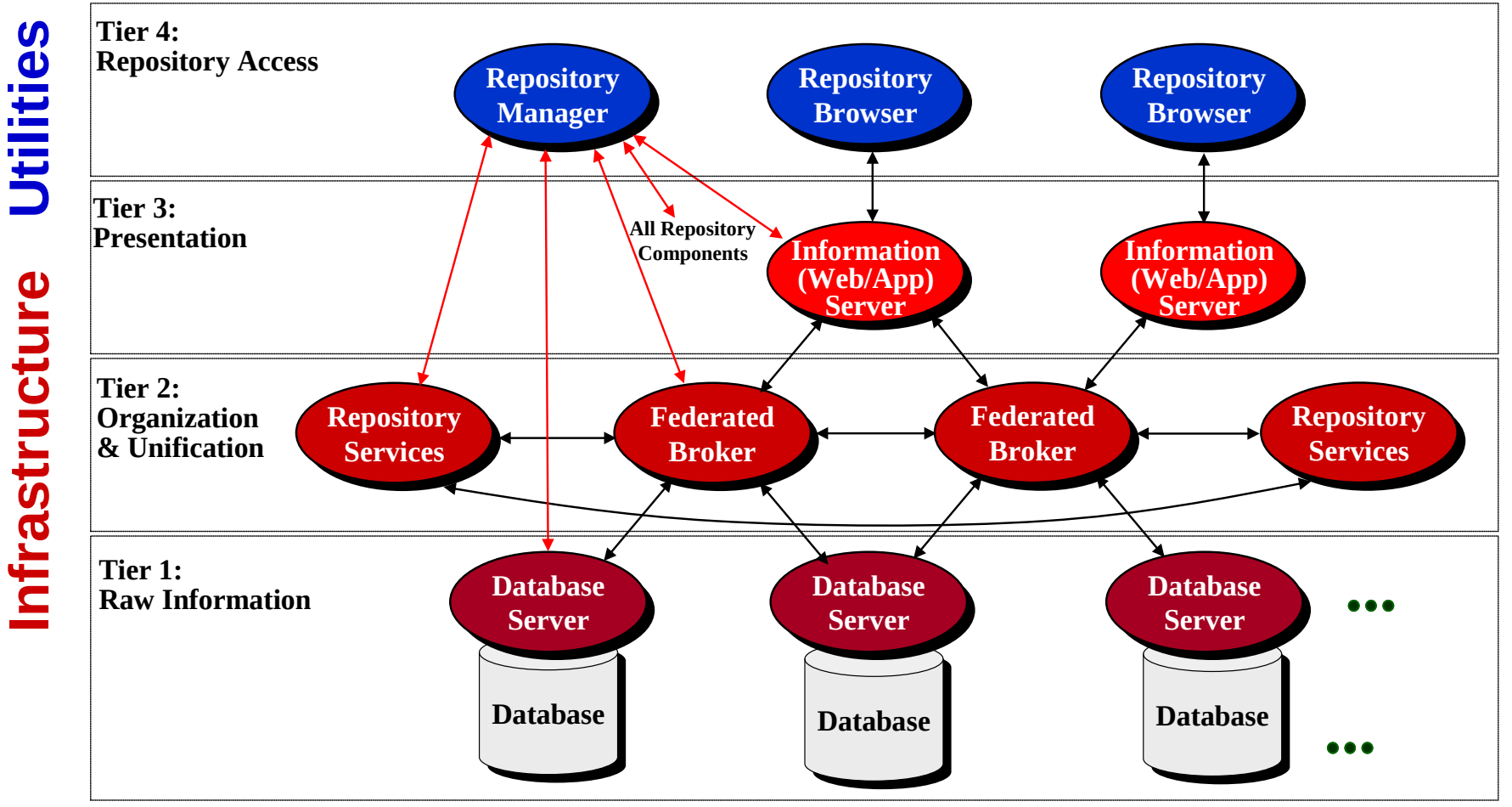
- **Communication needs to occur in two basic "modes"**
  - Communication between applications that are active simultaneously
    - Analogies: telephone, instant messaging
  - Communication between applications that are not active simultaneously
    - Analogies: mail, email
- **Non-Simultaneous communication requires management of persistent information**
- **Communication is necessary at different times, and these types of communication have different basic requirements**
  - Between exercises, always non-simultaneous → The Repository
  - During an event's lifetime for non-simultaneous comm. → The Logical Range Data Archive
  - During run-time for high-performance simultaneous comm. → The TENA Middleware

TENA Repository Purpose and Requirements

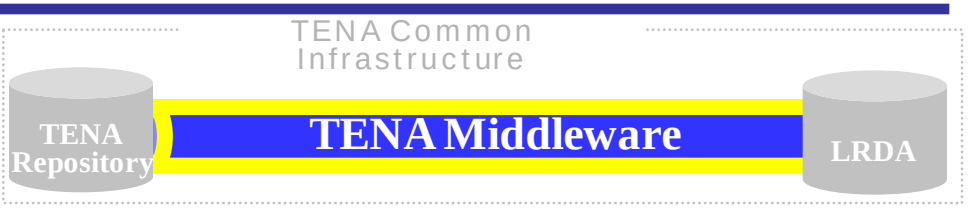
- **Purpose: to contain all the information relevant to TENA that is not specific to a given logical range**
- **Requirements:**
  - Store the TENA Object Model in all its forms including standard implementations
  - Store meta-data about all of its contents
  - Store TENA software (middleware, schemas, tools, gateways, reusable applications, and reusable components)
  - Store all TENA documentation
  - Store information from previous logical range executions for future reuse (including lessons learned)
  - Provide an easy-to-use secure interface to all of this information
- **The Repository is a database-of-databases, like the world-wide web.**
  - Except it has more meta-data, more security, more unification

TENA Repository Multi-Tiered Straw-Man Design

- This design is not "part of the architecture" — it is included to help illustrate the concept
- Obviously a web-based solution is the first step

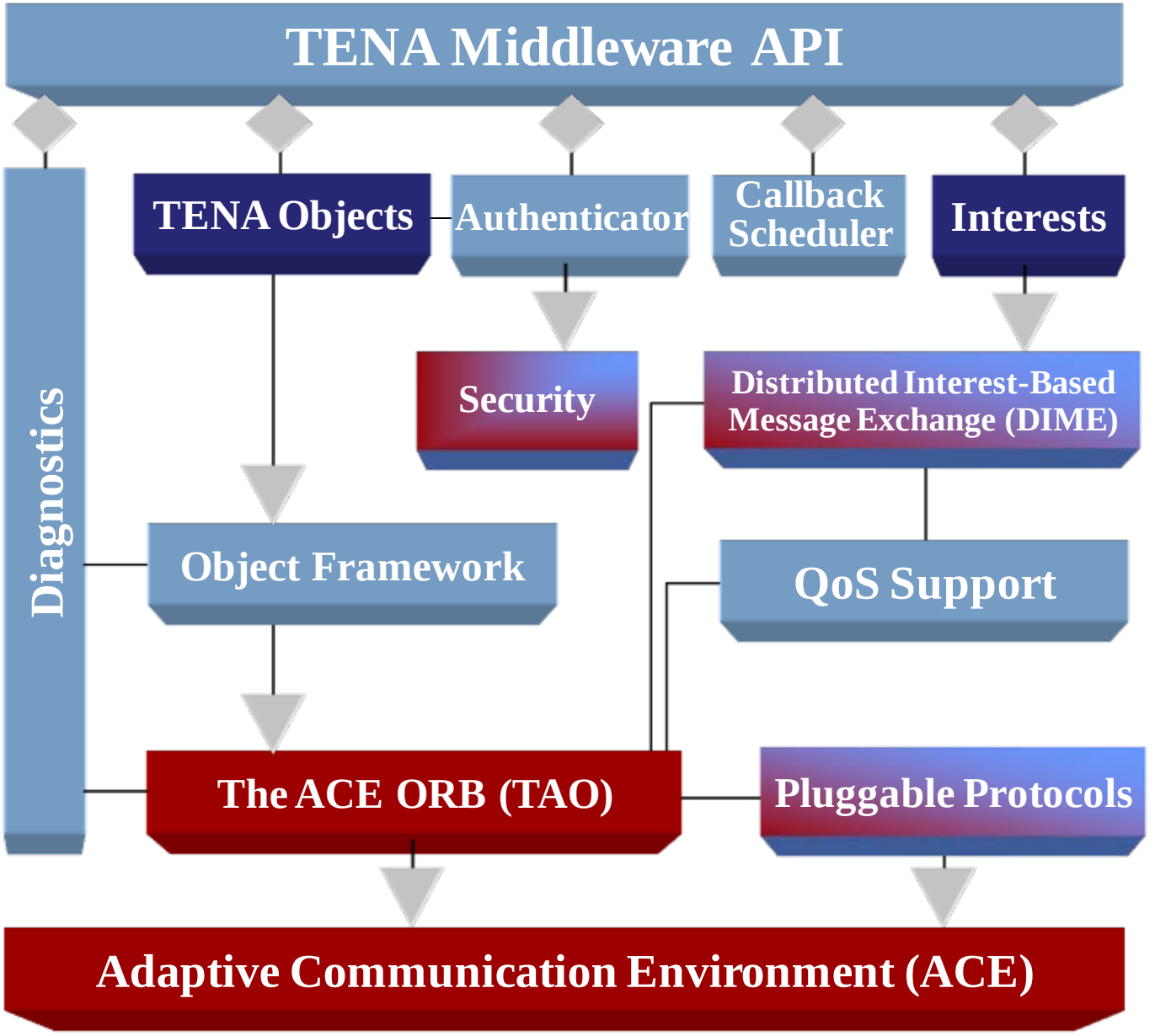


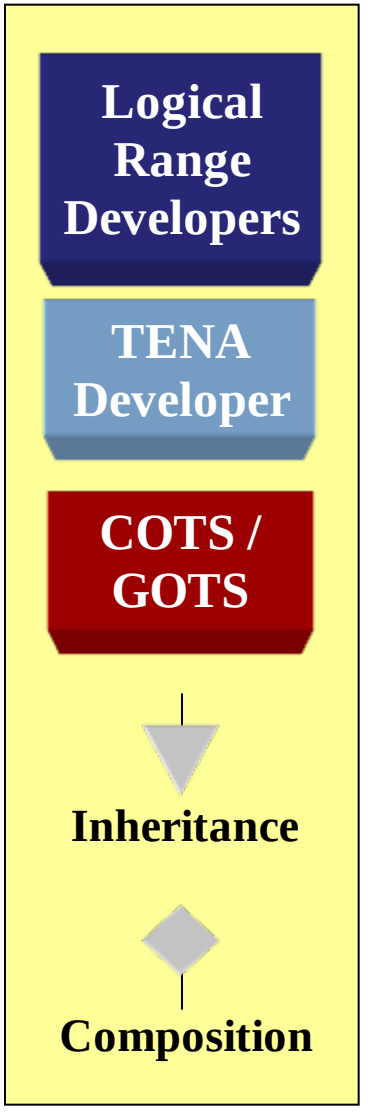
TENA Middleware Purpose and Requirements



- **Purpose: high-performance, real-time, low-latency communication infrastructure used by range resource applications and tools during execution**
- **Requirements:**
  - Fully support TENA Meta-Model
  - Be easy to use
  - Be highly reliable
  - Many varied communication strategies and media
    - Including management of quality-of-service
    - Including object-level security services
  - Be high-performance, including
    - Support multiple information filtering strategies
    - Support user-defined filtering criteria
  - Support a wide variety of range-relevant platforms (HW/OS/compiler)
  - Be technology neutral

TENA Middleware Current Design Overview





Supported Platforms

- Ardence ETS - NetAcquire (HW integrated Windows Real-Time OS) Microsoft Visual C++ 7.1 (bundled)
- Embedded Planet (embedded Linux OS) GCC 3.2.2 (bundled)
- Linux - Fedora Core 3 GCC 3.4.4 — Support for this platform is ending
- Linux - Fedora Core 4 GCC 4.0.2 — Support for this platform is ending
- Linux - Fedora Core 5 GCC 4.1.1
- Linux - Fedora Core 6 GCC 4.1.1 — New for R5.2.2
- Linux - Fedora Core 6, 64-bit GCC 4.1.1 — New for R5.2.2
- Linux - Red Hat 8 GCC 3.2 — Support for this platform is ending
- Linux - Red Hat 9 GCC 3.2.2 — Support for this platform is ending
- Linux - Red Hat Enterprise Workstation 4 GCC 3.4.4
- Linux - Red Hat Enterprise Linux 5 GCC 4.1.1 — New for R5.2.2
- Linux - SUSE 10.1 GCC 4.1.0
- Mac OS X 10.4.7 GCC 4.0.1 — New for R5.2.2 Universal Binary (Intel and Power PC) support
- Solaris 8 GCC 3.2.3 — Support for this platform is ending
- Solaris 10 Sun SPRO 5.8
- Solaris 10, 64-bit Sun SPRO 5.8
- Windows 2000 Microsoft Visual C++ .NET 2003 (aka Visual C++ 7.1) — Support for this platform is ending
- Windows Server 2003 Standard Microsoft Visual C++ .NET 2003 (Visual C++ 7.1)
- Windows Server 2003 Standard, 64-bit Microsoft Visual C++ .NET 2005 (Visual C++ 8.0) — New for R5.2.2
- Windows XP Microsoft Visual C++ .NET 2003 (Visual C++ 7.1)
- Windows XP Microsoft Visual C++ 2005 (Visual C++ 8.0)
- Windows Vista Microsoft Visual C++ 2005 (Visual C++ 8.0) — New for R5.2.2. User-specific HW/SW testing recommended prior to operational use

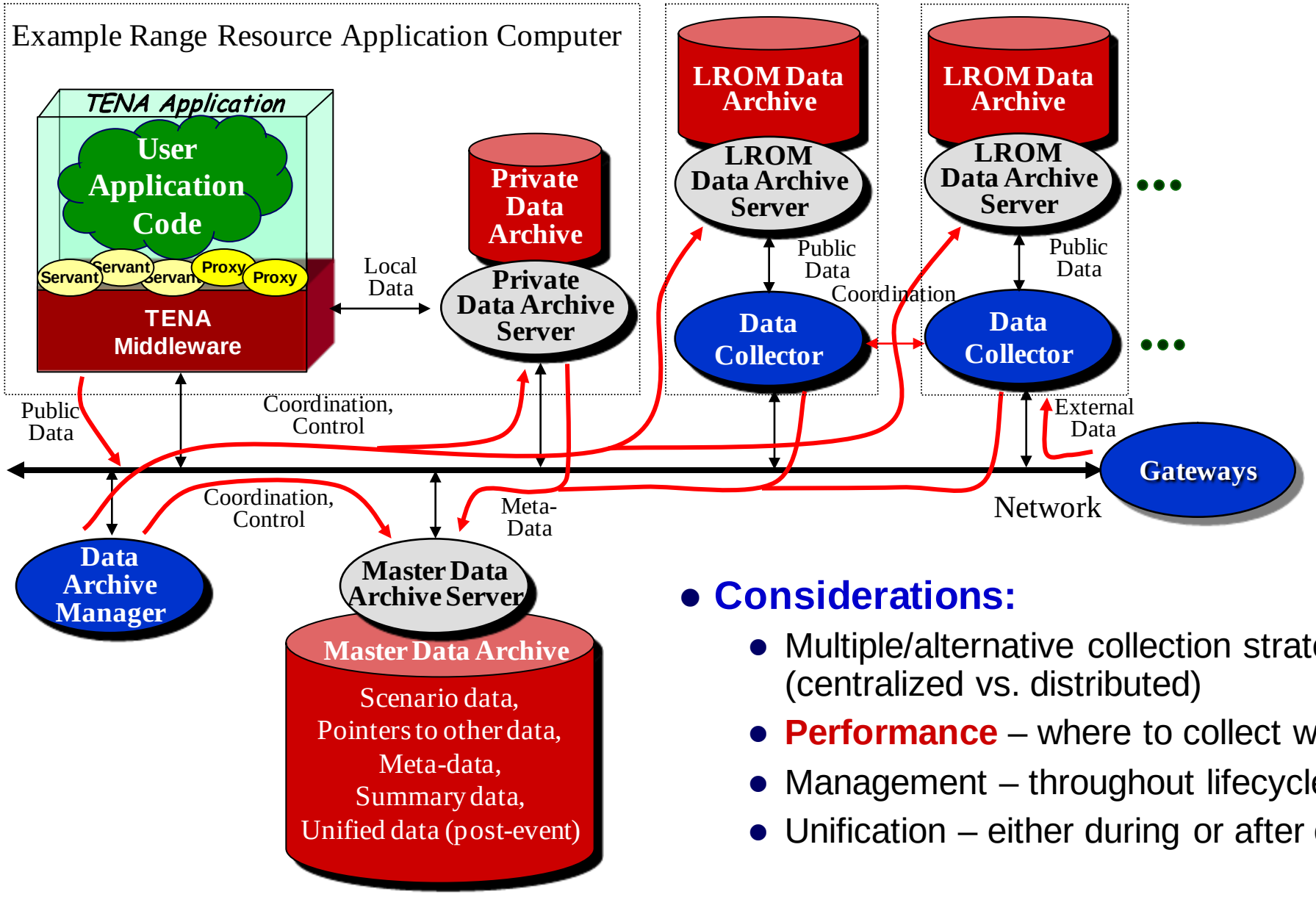
Release 6 Development Efforts

- **TENA Middleware Release 6 expected Summer 2007**
  - OM Consistency Checking
  - Enhanced OM Subsetting Support
  - Advanced Subscription Filtering
  - NNS and EM Fault Tolerance
  - Queued Publication State Delivery Policy
- **TENA Middleware Computer Platform Support (i.e., additional ports)**
  - New Windows OS (Vista) and compiler (Visual Studio .NET 2005)
  - New versions of Linux (RedHat Enterprise Workstation 5)
- **TENA Object Models**
  - Refined TSPI for new Time implementation
  - Refined PlatformType implementation to deal with user-identified issues
  - New packaging of OMs for r6 to minimize number of libraries
  - Refine AMO based on user testing

Logical Range Data Archive Purpose and Requirements

- **Purpose: store and provide for the retrieval of all of the information associated with a logical range execution**
- **Requirements:**
  - Store and serve initialization information
  - Store all data created in a logical range execution → **high-performance**
  - Store information at (possibly) multiple collection points → **distributed**
  - Support a "temporal" understanding of collected information → **temporal**
  - Support run-time queries as much as possible → **real-time**
  - Support post-event analytical queries
- **These things are non-trivial**
- **Does not have to be a single database running on a single computer (but could be)**
  - Perhaps a federated multi-database running on many computers throughout the logical range

Logical Range Data Archive Straw-Man Design

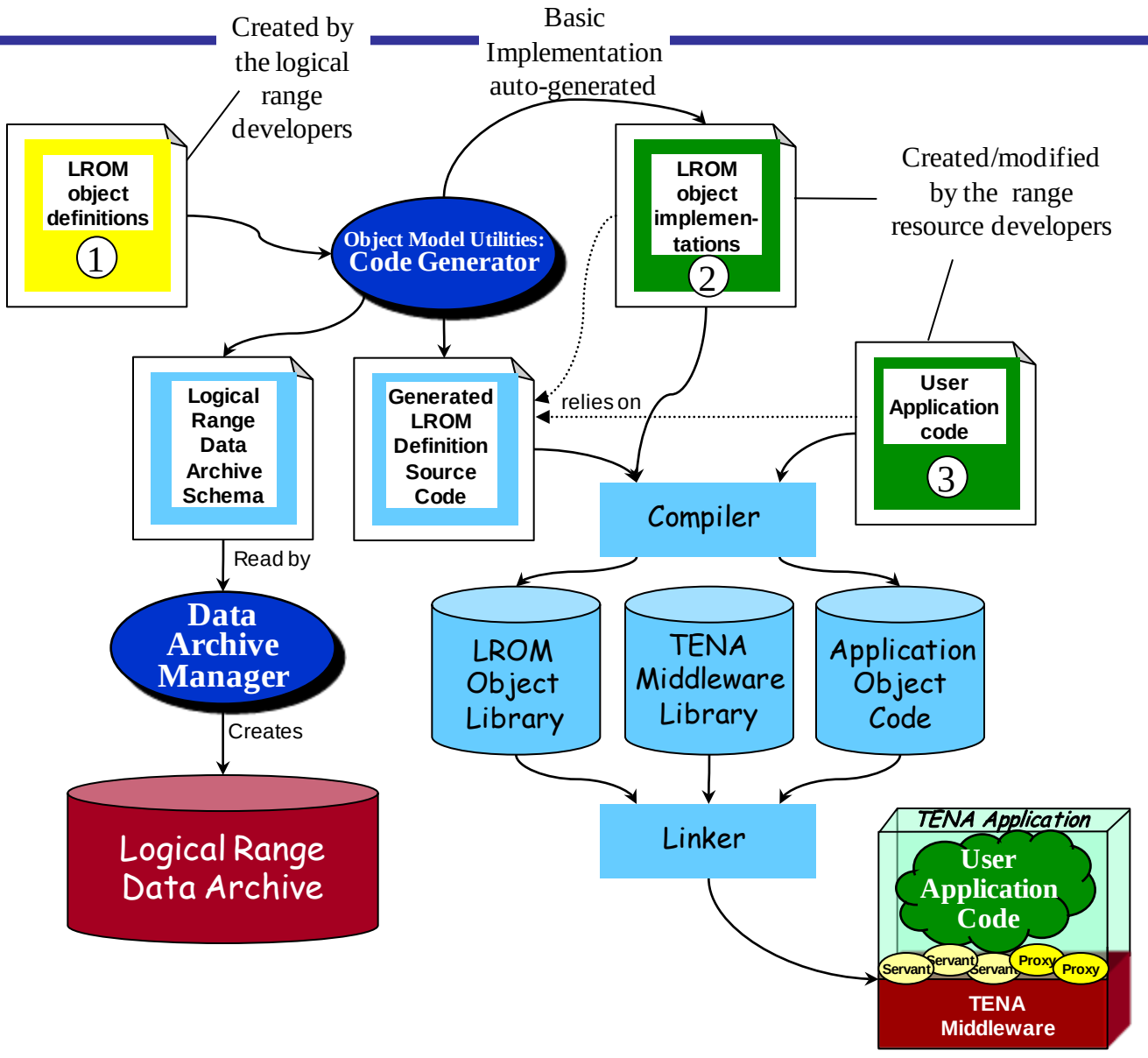


- **Considerations:**
  - Multiple/alternative collection strategies (centralized vs. distributed)
  - **Performance** – where to collect what?
  - Management – throughout lifecycle
  - Unification – either during or after event

Domain-Specific Software Architecture – What is it?

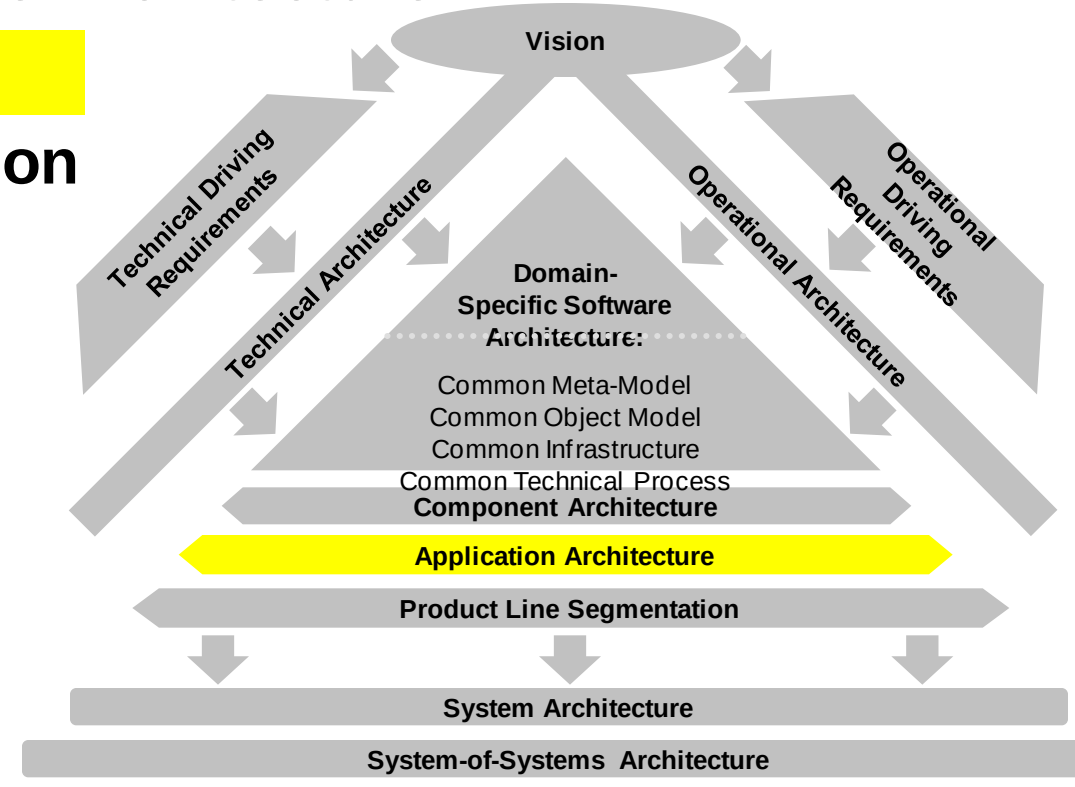
- **Common Meta-Model**
  - What are "TENA Classes" and "TENA Objects?" (i.e., what are SDOs?)
  - What features do these objects have?
- **Common Object Model**
  - What are the standard TENA Classes?
  - It is a standard language for semantic interoperability
- **Common Infrastructure**
  - How are the TENA Objects managed and communicated?
  - Must support entire range event lifecycle
- **Common Technical Process**
  - What are the basic processes for initiating, conducting, and finishing communication about TENA objects?
  - Focused on the **technical** processes, not operational processes

Creating a TENA Application



TENA Organization

- Technical Driving Requirements
- Operational Driving Requirements
- Technical Architecture View
- Operational Architecture View
- Domain-Specific Software Architecture
- **Application Architecture**
- Product Line Segmentation



TENA Application Architecture

- **Purpose:** Explains how applications should be built
- Emphasizes that the middleware and the LROM are linked into the application

APPLICATION CODE:

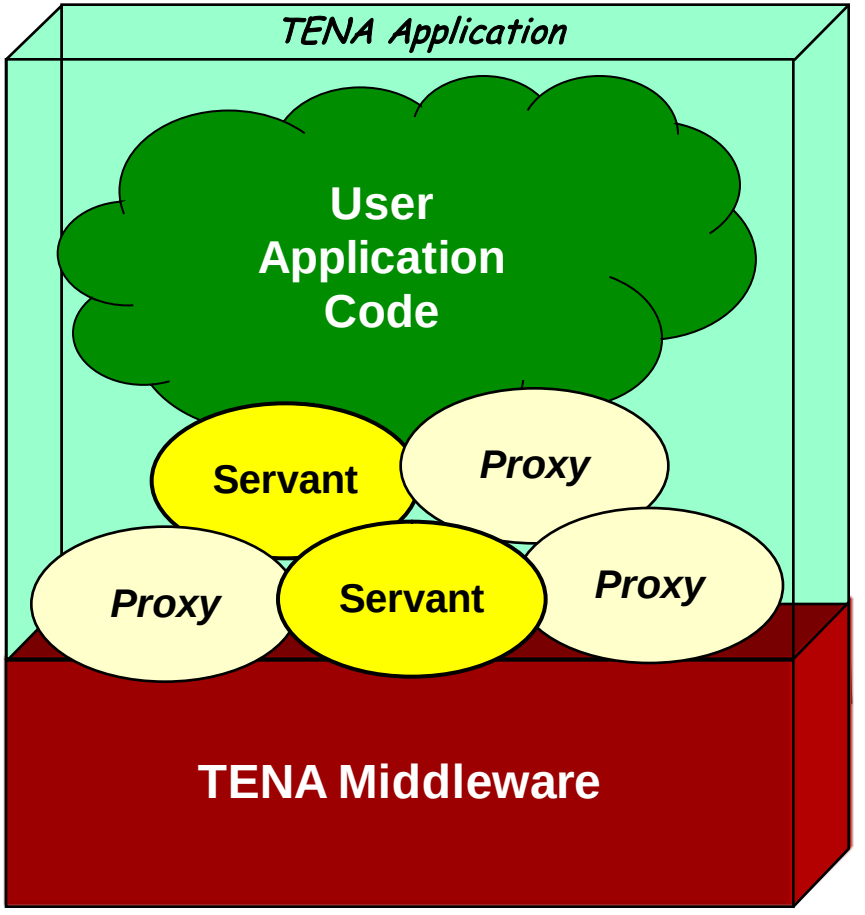
Specific to an individual application

OBJECT MODEL CODE:

Common across a given logical range

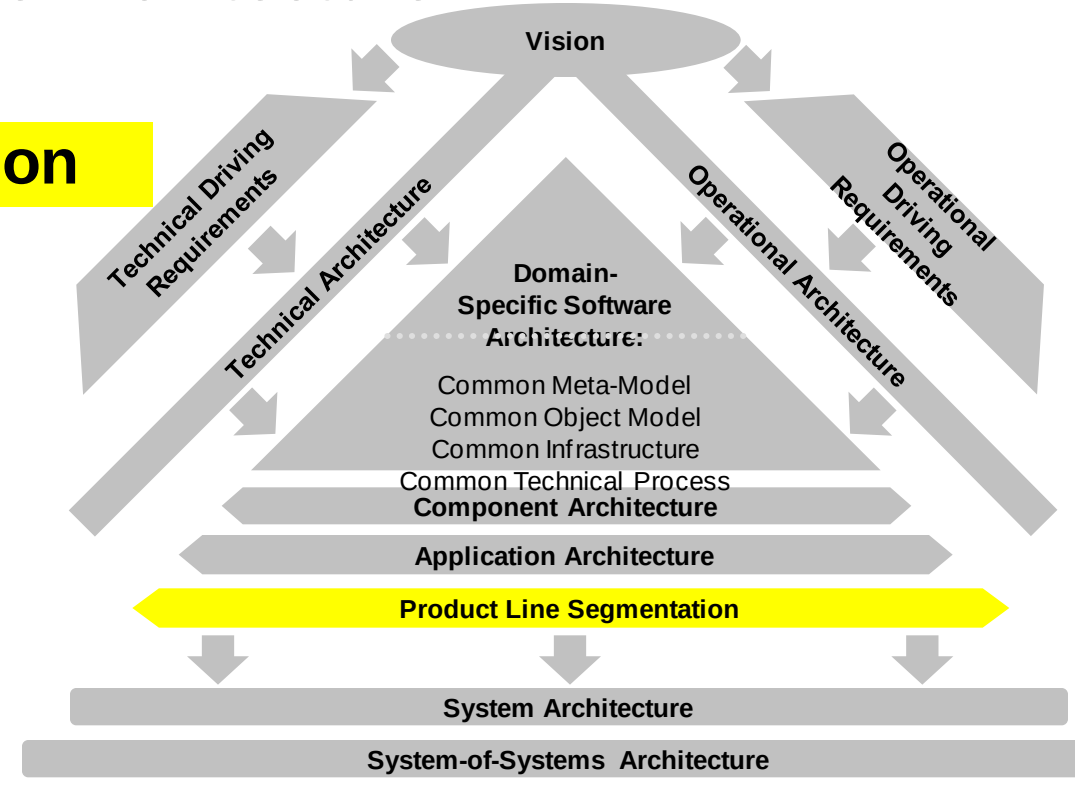
TENA CODE:

Common across all TENA applications



TENA Organization

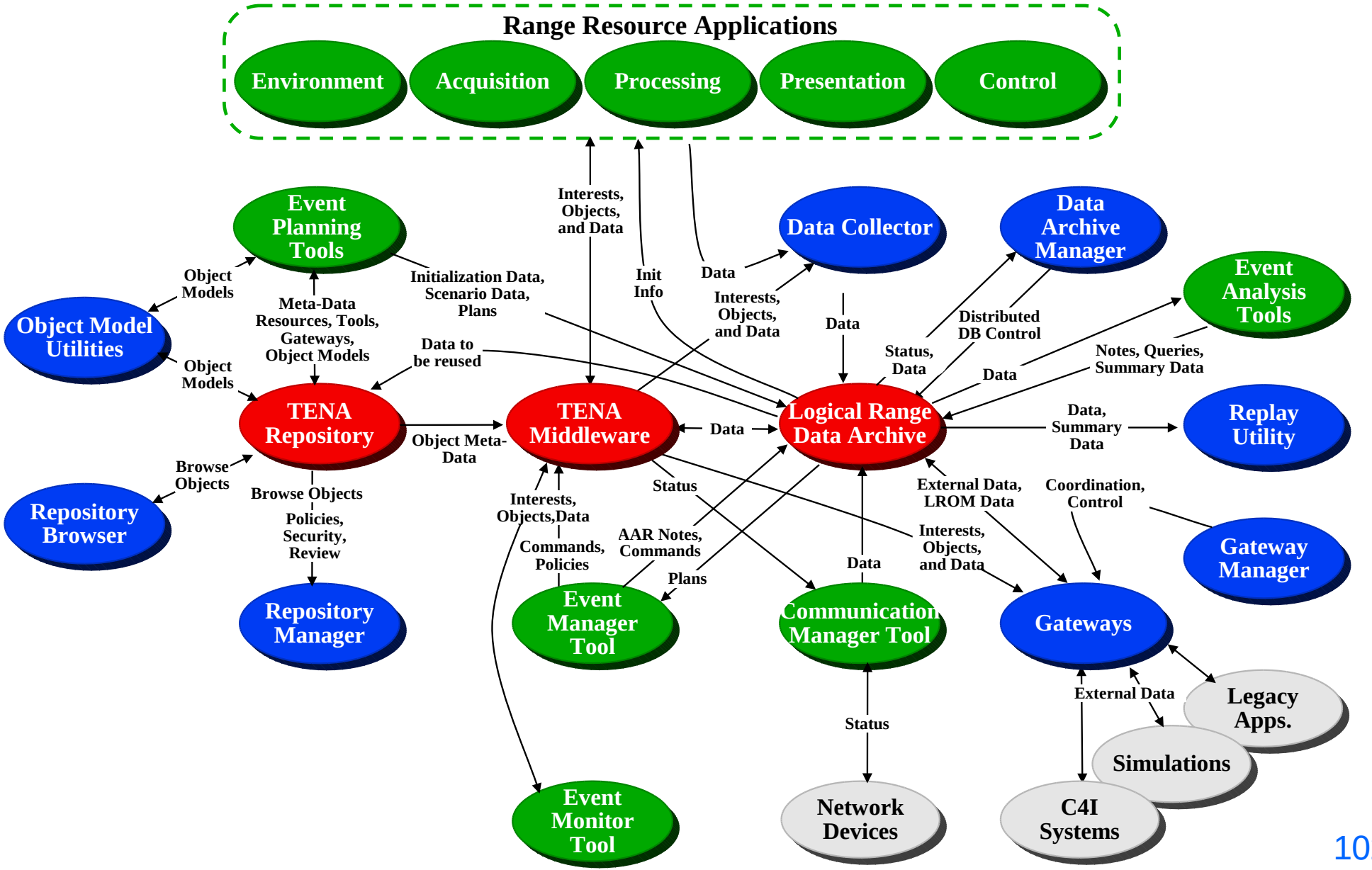
- Technical Driving Requirements
- Operational Driving Requirements
- Technical Architecture View
- Operational Architecture View
- Domain-Specific Software Architecture
- Application Architecture
- Product Line Segmentation



TENA Product Line

- **The Product Line is the only place that gives architectural advice on what to build rather than how to build it**
- **The Product Line is derived from an analysis based on the Operational Architecture and the Common Technical Process**
- **Products in the Product Line are organized into four basic categories:**
  - Range Instrumentation – does the work of the range
  - Utilities – help make TENA work
  - Tools – reusable applications that help perform tasks in the ConOps
  - Gateways – bridge TENA to other communication mechanisms/architectures

TENA Product Line Overview

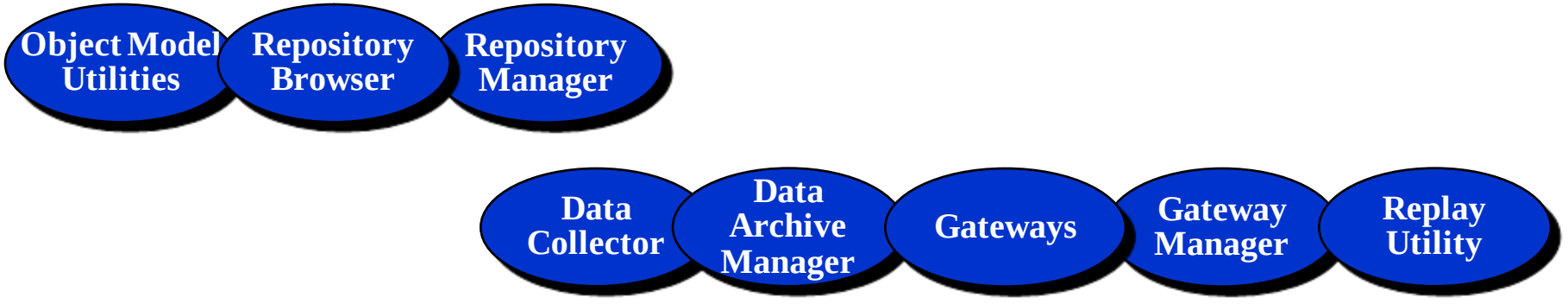


TENA Applications & Tools

- **Range Resource Applications**
  - Support the range infrastructure
    - Instrumentation and Sensor interfaces
- **TENA Tools**
  - Reusable applications that support the Logical Range Event Process
    - Test/Exercise Planner, Resource Manager, Test/Exercise Manager and Test/Exercise Analyzer

TENA Utilities: Purpose and Requirements

- **Purpose: To assist the user in planning for, creating, managing, and succeeding with a TENA Logical Range**
- **Requirements:**
  - Utilities should assist the user throughout the entire event life-cycle
  - Utilities should assist the user in dealing with the object model
  - Utilities should assist the user in dealing with the infrastructure
  - Many focused utilities are better than a few multi-featured tools
  - Some utilities are explicitly required by the JORD
- **In a perfect world, all of the utilities would be built upon a common set of reusable components**



TENA Integrated Development Environment (TIDE)

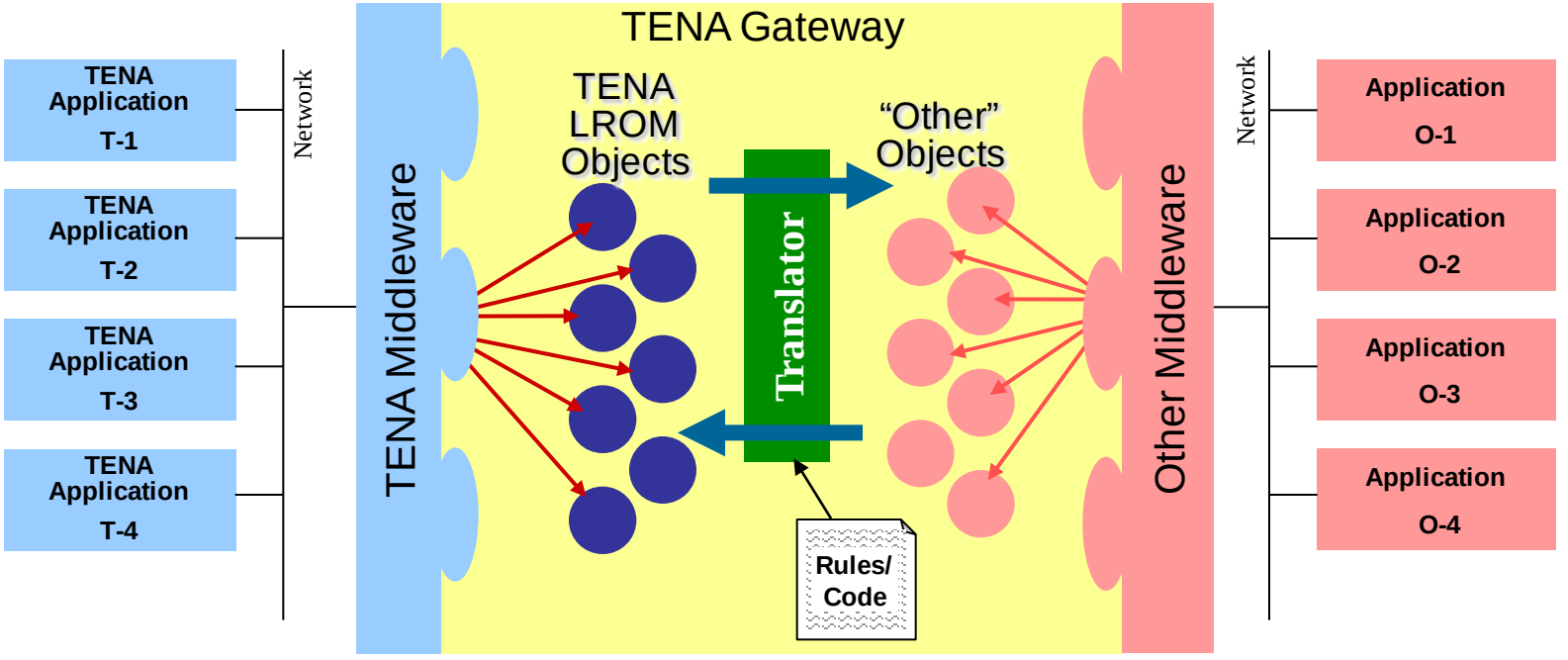
- **TIDE is a tool designed to assist developers in the creation, development, testing and deployment of TENA applications**
  - Based on the Eclipse Framework

- **Capabilities**
  - Catalog installed object models on a user's machine
  - Migrate user applications between object model versions
  - Migrate user applications between middleware versions
  - Browse and download object models available in the TENA Repository
  - Request object model distributions from the TENA Repository

- **TIDE 1.1 release**
  - Available at http://www.tena-sda.org/tide web site

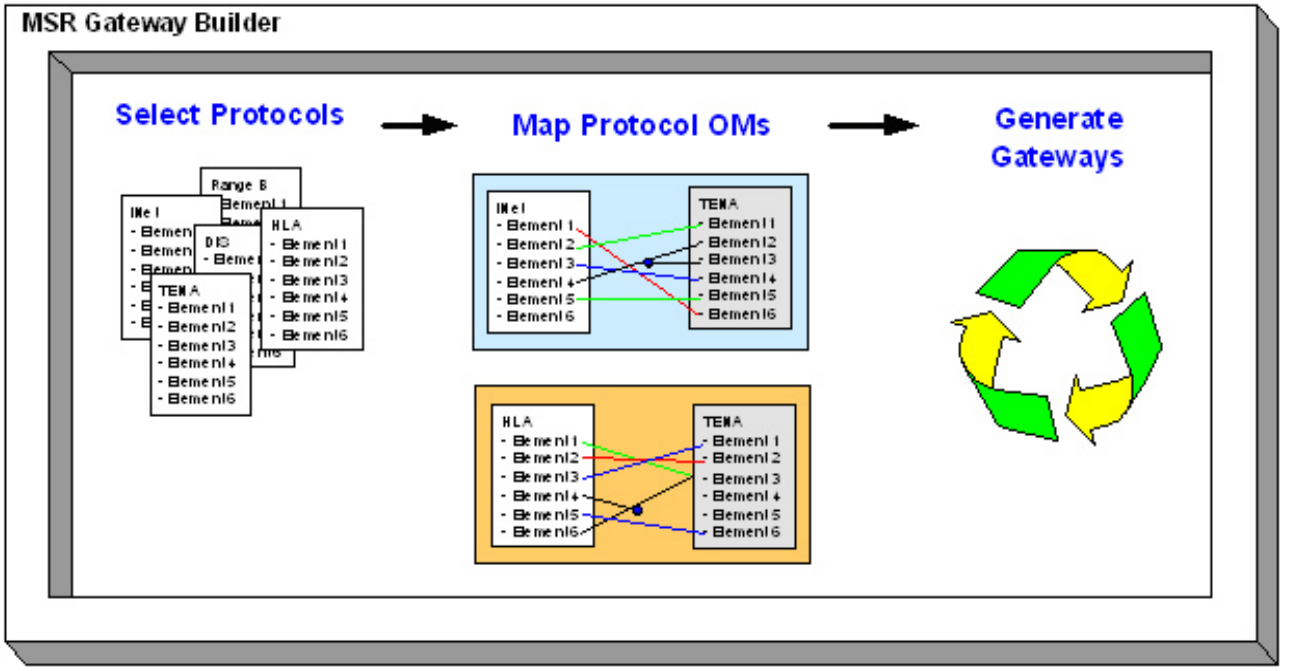
Gateways

- Gateways provide a means of bridging TENA systems to non-TENA systems
- Gateways are TENA applications but may also conform to other architectures
- The most important gateways will bridge TENA to the HLA and to C4ISR systems



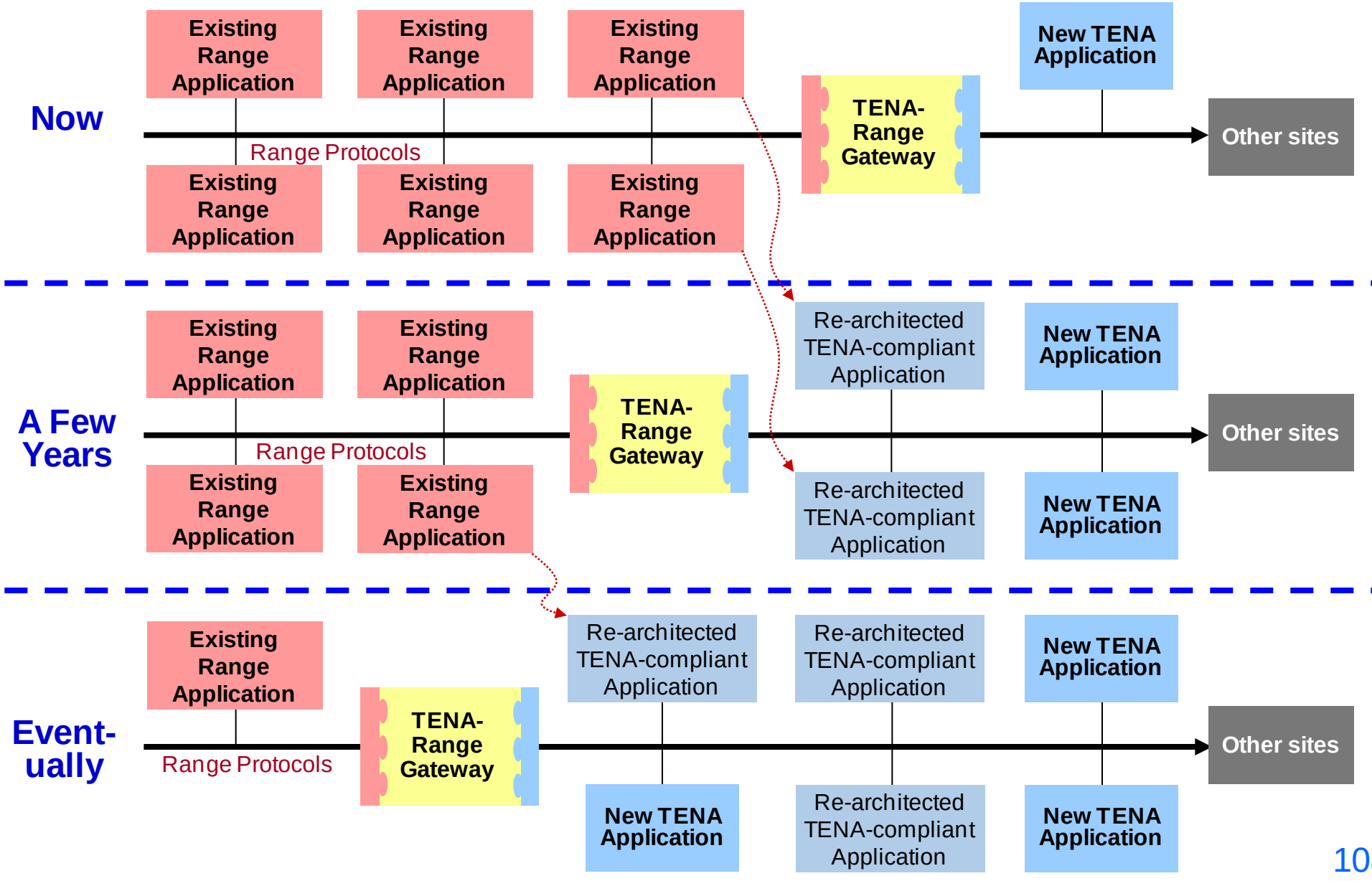
Gateway Builder

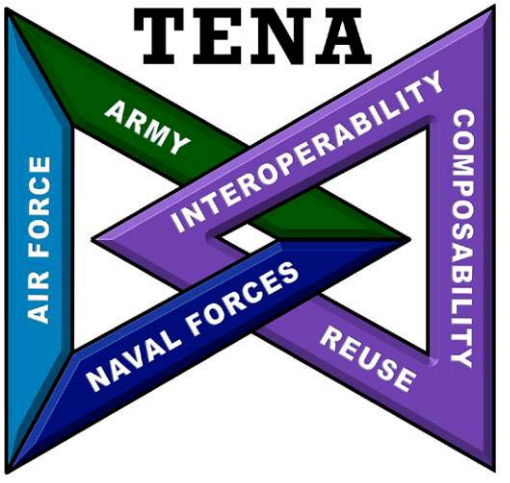
- MSR Program is focused on integration of distributed live, virtual, and constructive (LVC) systems into a common synthetic battle space that comprises various simulation protocols, training ranges, live systems and platforms
- Gateway Builder streamlines integration process and reduces time and effort of creating gateways
- Gateway Builder is a flexible, extensible, graphically driven tool that automatically generates gateways to bridge simulation and live protocols
- Gateway Builder supports mappings between TENA, DIS, and HLA and message-based protocols using any object model



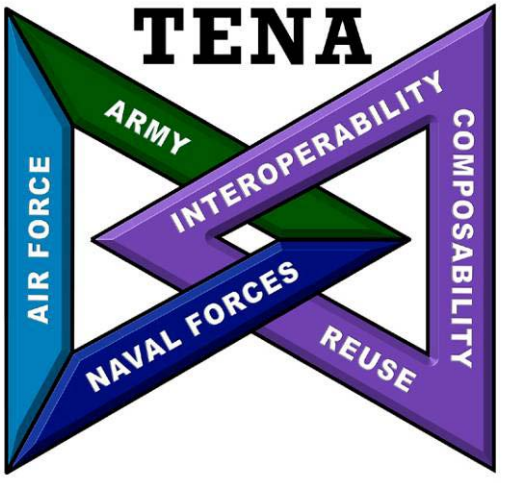
Gateway Builder Simplified Block Diagram

Gradual Deployment of TENA




TENA
ARMY
AIR FORCE
INTEROPERABILITY
COMPOSABILITY
NAVAL FORCES
REUSE

Questions?



Concluding Remarks

TENA Solutions to Interoperability Challenges

- **On-the-Wire Specification vs. API Standard**
  - API Standard allows future technological advances for data transmission to be much more cost-effectively incorporated
- **Single Reference Frame vs. Multiple Reference Frames**
  - Multiple Reference Frames allow different range systems to operate in the coordinate system most optimum for their range
- **Single Level vs. Multiple Levels of Compliancy**
  - Multiple Levels of Compliancy allow a more meaningful definition of compliancy to be used among Range engineers & investment managers
- **Run-Time Interpreter vs. Compile-Time Integration**
  - Compile-Time Integration allows for inconsistencies to be discovered when the software is being upgraded vice during the event
- **Hand-Coded vs. Auto-Code-Generated Interfaces**
  - Auto-Code-Generated Interfaces can be produced more reliably and tremendously faster than traditional hand-coded interfaces
- **Centralized (Client/Server) vs. Peer-to-Peer**
  - Peer-to-Peer gives more flexibility to exercise designers – can emulate client/server if necessary

Key Elements of TENA Revisited

- **TENA lowers the cost to integrate systems together**
  - Some systems made TENA-compliant <$20K for MC-02
- **TENA decreases the time to integrate systems together**
  - Auto-code-generator generated 50K+ lines of code in a few hours from a 4-page interface definition document
  - Legacy display system made TENA-compliant in 4.5 days for MC-02
  - Hydrophone instrumentation system made TENA-compliant in 2 days
  - HLA-compliant display system gateway made TENA-compliant in 1 day
- **TENA lowers the cost to reuse systems in future events**
  - Examples include VAST/IMPASS reusing Sunburst capability
  - Will be better realized in future JNTC events
- **TENA improves flexibility of integrating systems together**
  - Range applications can be optimally configured for the particular test event

Key Elements of TENA Revisited (cont.)

- **TENA improves reliability of integrating systems together**
  - Auto-code-generator ensures that every system has same baseline of source code
  - Standard, validated algorithms (such as coordinate translations or unit conversions) can be embedded in TENA rather than burden software applications of managing and performing translations
  - TENA Middleware performs data marshalling/demarshalling rather than burden software applications
- **TENA eases Deployment at the DoD Ranges**
  - TENA can be deployed gradually (system by system) rather than requiring all systems be redesigned
  - Providing on-site training at a number of ranges
- **TENA has a process to follow for sustainment/improvement**
  - Leverages CTTRA workshops and the Architecture Management Team (AMT)
  - Established on-line User Help Desk system to capture feedback from TENA users
  - Pursuing RCC standards, and investigating OMG standards
  - Working with T&E CTTRAP to determine TENA policy among Services

DoD Directive on TENA

Business Initiative Council TE-09
Common Test and Training Range Architecture Policy (CTTRAP)

- **Leverages lessons learned from past directives including Ada, HLA, and JTRS (mostly what not to do—no blanket mandate)**
- **Establishes a flexible process where the Services make the final determination on TENA compliancy for their systems on a case-by-case basis**
  - TENA compliancy must not adversely impact cost, schedule, or performance of the individual range system
- **All new range systems will be required to use TENA**
- **All existing range systems that are having their data distribution mechanism upgraded will be required to use TENA**
- **The Directive applies if the current version of TENA satisfies the interoperability requirements of the new or upgraded range system. If not, the interoperability requirements for the new system will be identified so the appropriate upgrades to TENA can be made by CTEIP**
- **OSD(P&R) and DTRMC will oversee the sustainment of TENA**

Key Functionality of TENA Beyond HLA

Standard Object Model

TENA provides for the managed evolution of a standardized Object Model (interfaces, data formats, data definitions, control commands, etc.)

Significance: *Range-community-wide agreed upon data formats, definitions, etc. promotes interoperability to a greater degree than the HLA specification*

High Performance and Reliability

TENA Objects are "compiled-in" when the application is made TENA-compliant

Significance: *Higher performance, plus higher reliability since any errors in data formats will be discovered during software compiling (pre-mission) rather than during the test mission (at run-time)*

Support for More Complex, Meaningful, User-Defined Object Models

TENA allows for objects to be composed of other objects (objects can contain other objects)

Significance: *Small "building block" objects (Time, Position, Orientation, etc.) can be standardized and reused to efficiently define other more complex objects, yielding more interoperability quickly at less cost than with the HLA*

TENA Middleware marshals/demarshals data, rather than relying on individual applications to do so

Significance: *Middleware marshaling makes it easier to integrate different computer platforms (Windows, Linux, Sun, etc.) in a distributed test event and avoid integration errors due to inconsistent user-written software*

TENA supports a more rich description language for defining the information that needs to be communicated

Significance: *Software interfaces can be designed more naturally and effectively for distributed test events*

(Future) Manages Persistent Data

TENA provides for the management and standardization of database information throughout the range event lifecycle, including scenario information and data collected during an exercise

Significance: *Interoperability is achieved before, during, and after a range event, leading to easier setup, initialization, and analysis, saving both time and money*

Summary of What We Have

An Architecture for Ranges, Facilities, and Simulations to Interoperate, to be Reused, to be Composed into greater capabilities

- **A Working Implementation of the Architecture**
  - TENA Middleware currently works on Windows, Linux, and Sun
- **A Process to Develop and Expand the Architecture**
  - CTTRA Workshops and AMT Meetings
- **A Technical Strategy to Deploy the Architecture**
  - Gateways provide interim solutions as TENA interfaces
- **A Definition of Compliancy**
  - Levels of compliancy to enhance communication among systems engineers and investment decision makers

Important Contact Information

- **Project Website: http://www.tena-sda.org**
  - Download TENA Middleware
  - Submit Helpdesk Case (http://www.tena-sda.org/helpdesk)
    - Use for all questions about the Middleware

- **TENA Feedback: feedback@tena-sda.org**
  - Provide technical feedback on TENA Architecture or Middleware
  - Ask technical questions regarding the TENA architecture or project
  - Provide responses to AMT action items
  - Request TENA training