KEANE

NDIA
NATIONAL DEFENSE INDUSTRIAL ASSOCIATION

Component-based Architecture And Modeling and Simulation

SBA Observations

- Dr Sega
  - Platform-centric• network centric
  - Common vision representation
    - Multiple function areas
    - Joint, interoperable, re-useable models
- Dr Dahmann
  - System focus • mission focus
  - Systems of systems pose new challenges
- Dr Castro
  - Multi-scale, multi-view
  - Contractors funded by systems, not by compatibility
- Mr Schade
  - Seamless data exchange/interface standards
  - Knowledge retention
- Ms. Zimmerman
  - Rapidly composable and scalable M&S
  - Strong CM focus
  - Build only what is needed
- Mr Lunceford
  - M&S best practices still a mystery
  - Begin shift of M&S from craft to scientific/engineering discipline

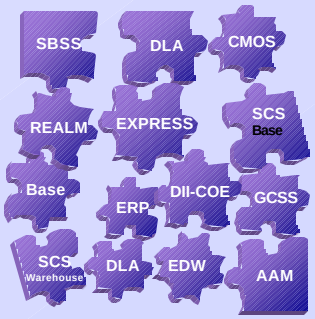
Today's Goals

- Discuss
  - Describe AF component-based strategy and approach
    - The picture and the pieces
  - Apply lessons learned to M&S
  - Components and the DoD capabilities vision
    - Components are a way of thinking about systems and organizations
      - Not just IT
  - Challenges
- Generate ideas, discussions, and excitement

Background Projects

Component-based Supply Chain Modernization

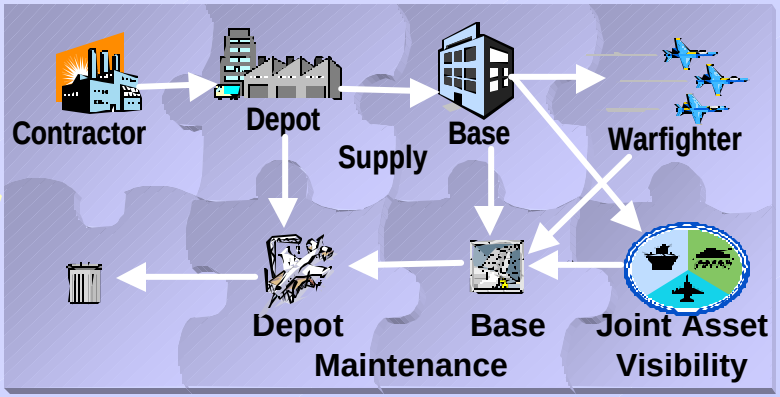
100+ Separate Logistics Systems



Today

Modernize Supply Chain

Integrated Logistics Enterprise



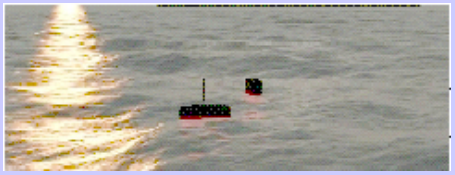
Tomorrow

Air Force Logistics

- Collection of stovepiped systems → Integrated picture across IL
- Pieces do not connect → Complete connectivity
- Picture not complete → Total Asset Visibility (TAV)
  - factory-to-foxhole

Background Projects

3D Component-based M&S Framework

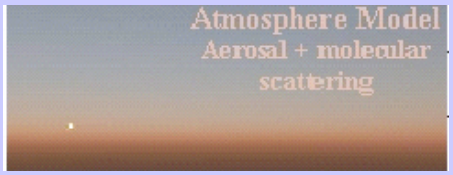


Arete Ocean Model
Ocean heights
Radiance function (point)

Component-based Models

Component-based Rendering

- Reflections
- Dynamic grid sampling
- Rendering



- Rendering
- Transmission

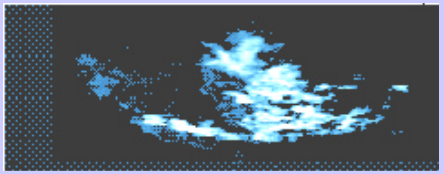
RenderWorld™ Visualization Framework

- Transmission

Sensor Motion
Object Interaction
Coordinate Transformations

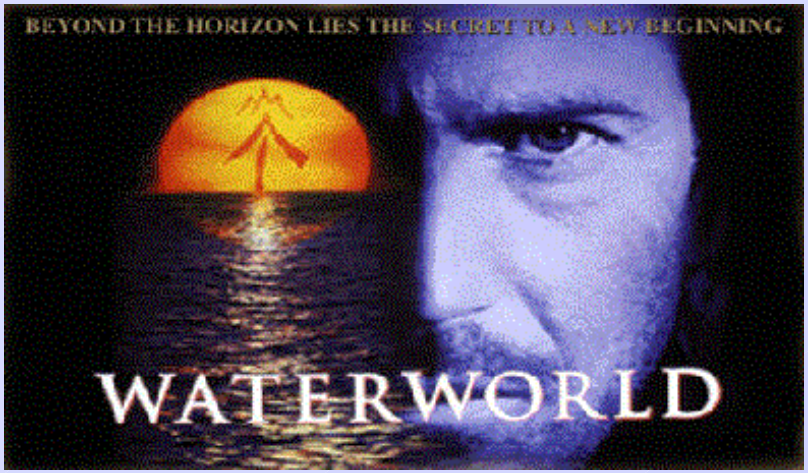
- Scattering
- Volumetric Integration

Arete d Model
Liquid wate t
Scattering phase function



Serves as reference – not recommendation

Background Projects
Component Based Environment Simulator

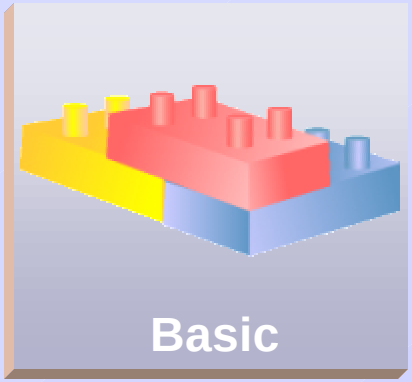


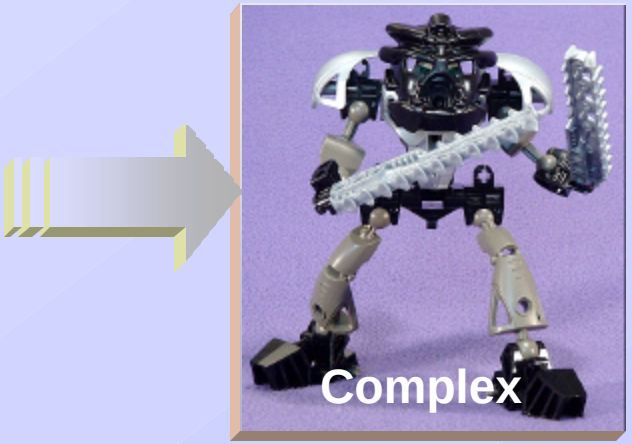
- Commercial Product (with DoD origins)
- Component architecture makes it very flexible
  - Plug in model to numerous rendering packages

What is a component?

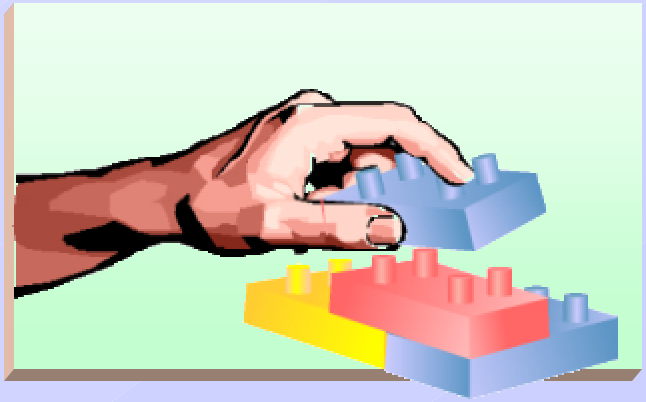
Software building blocks

- Structured interfaces
- Clear purpose
- Build complex apps



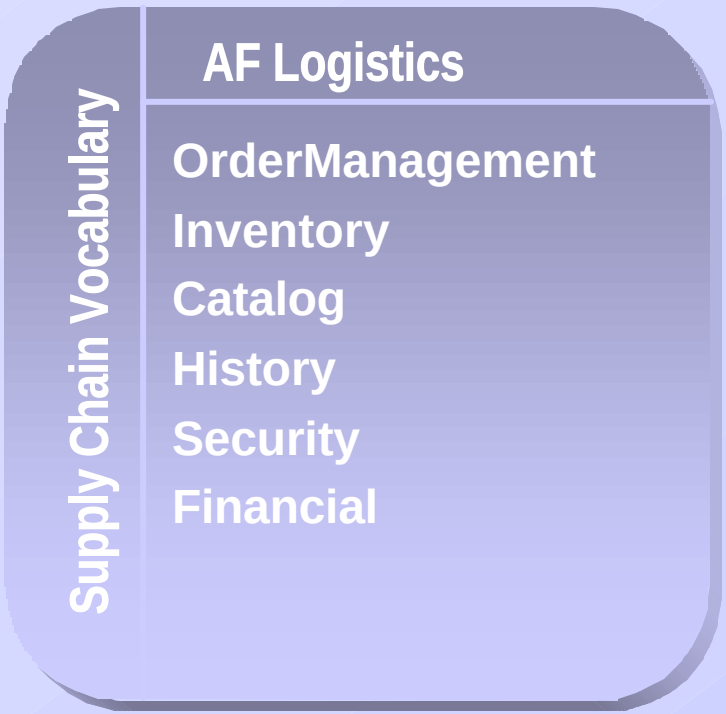


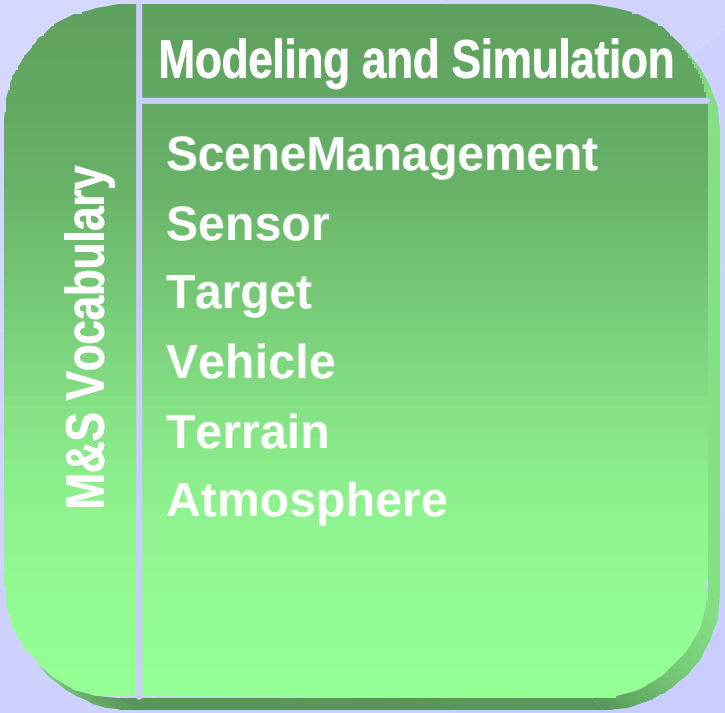
- **Examples**
  - Legos (complex and general purpose)
  - Dictionary in MS products



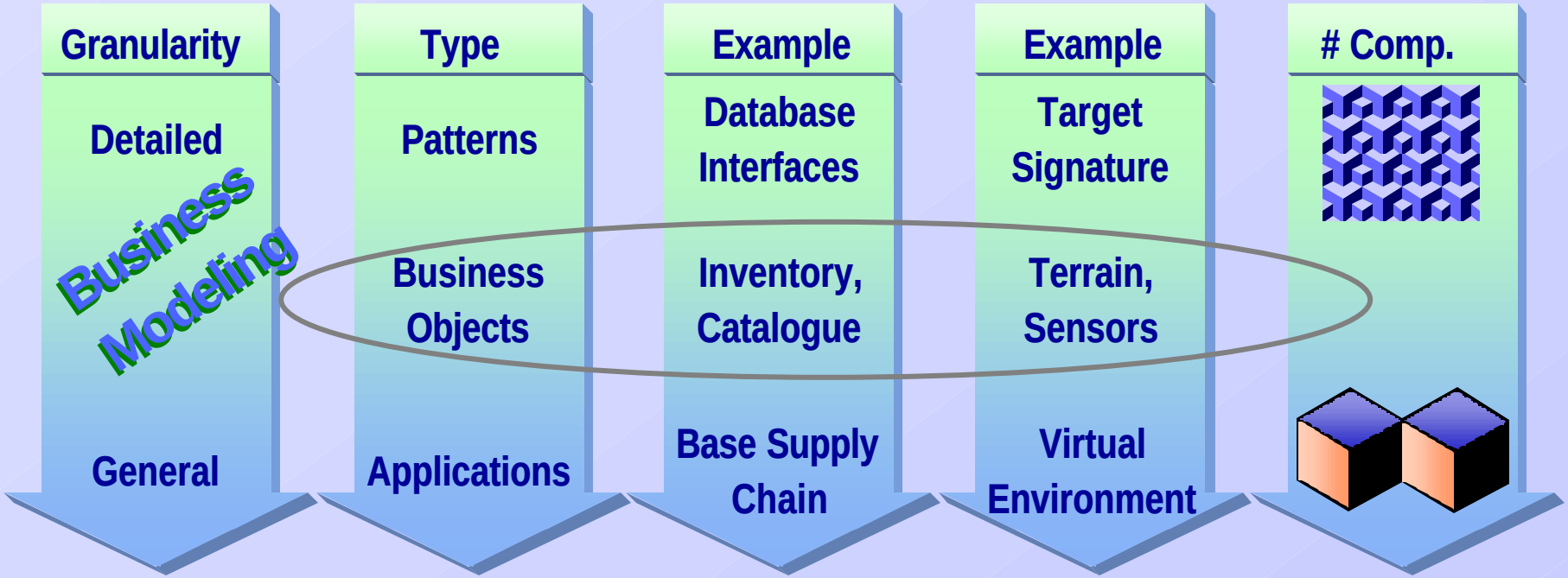
Component-based development

Components Examples





Component Granularity



Need **right** granularity:

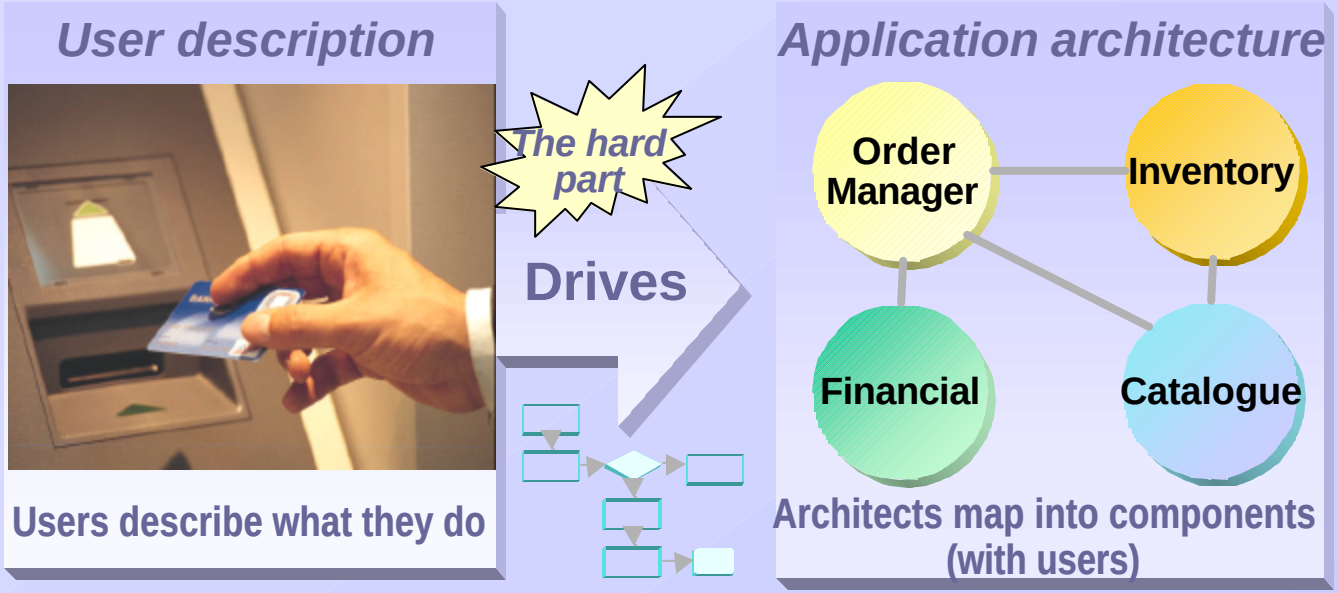
Too granular – exceedingly complex
Too general – limits reusability

How do you get components?

Use Case Modeling

Examples

- Withdraw $$
- Order item
- Fly thru

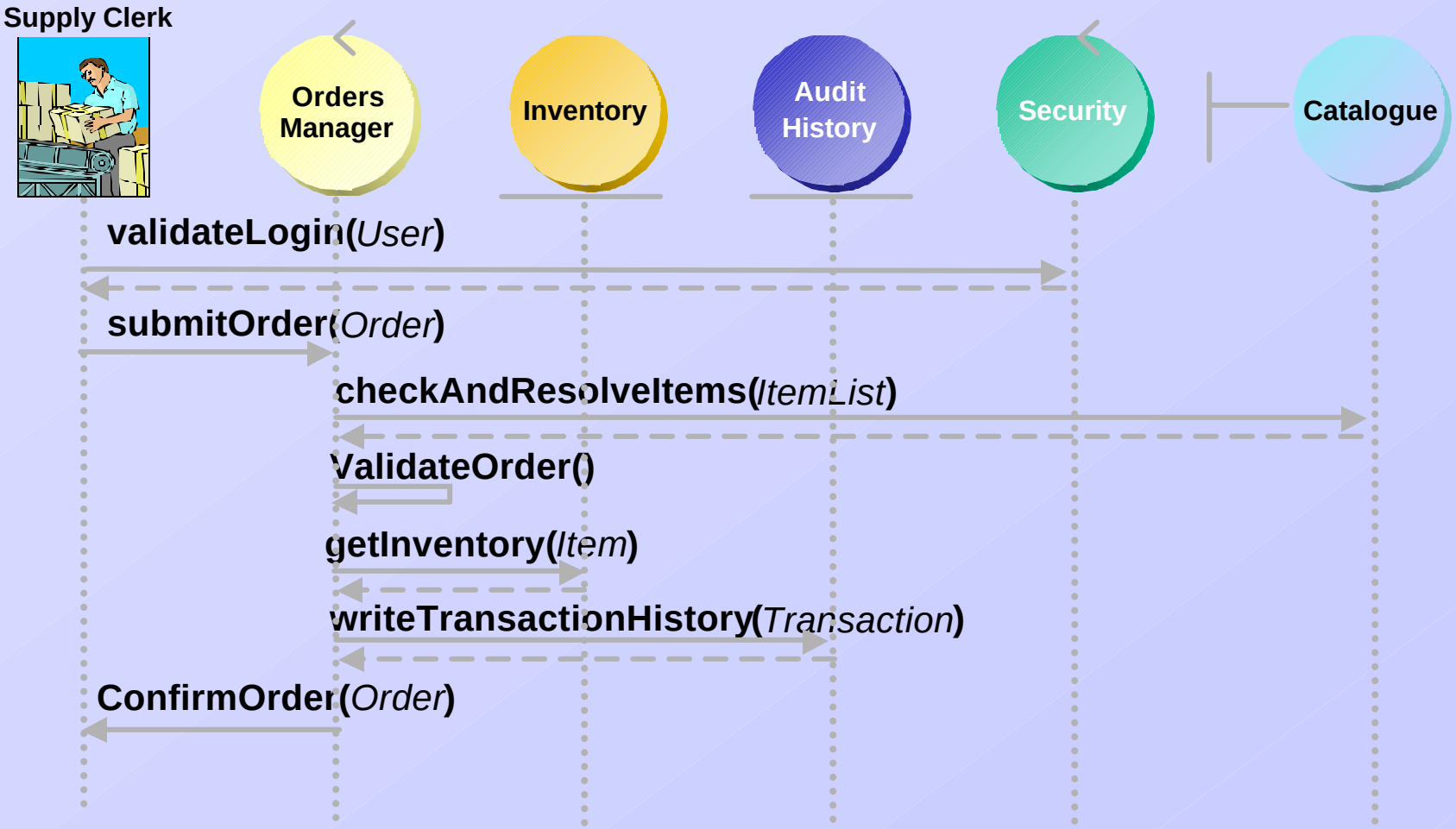


- **Use case modeling**
  - Text, activity diagrams
  - User-centric
- **Components and interactions**
  - Architects map into enterprise components

Mission-focused, not platform centric

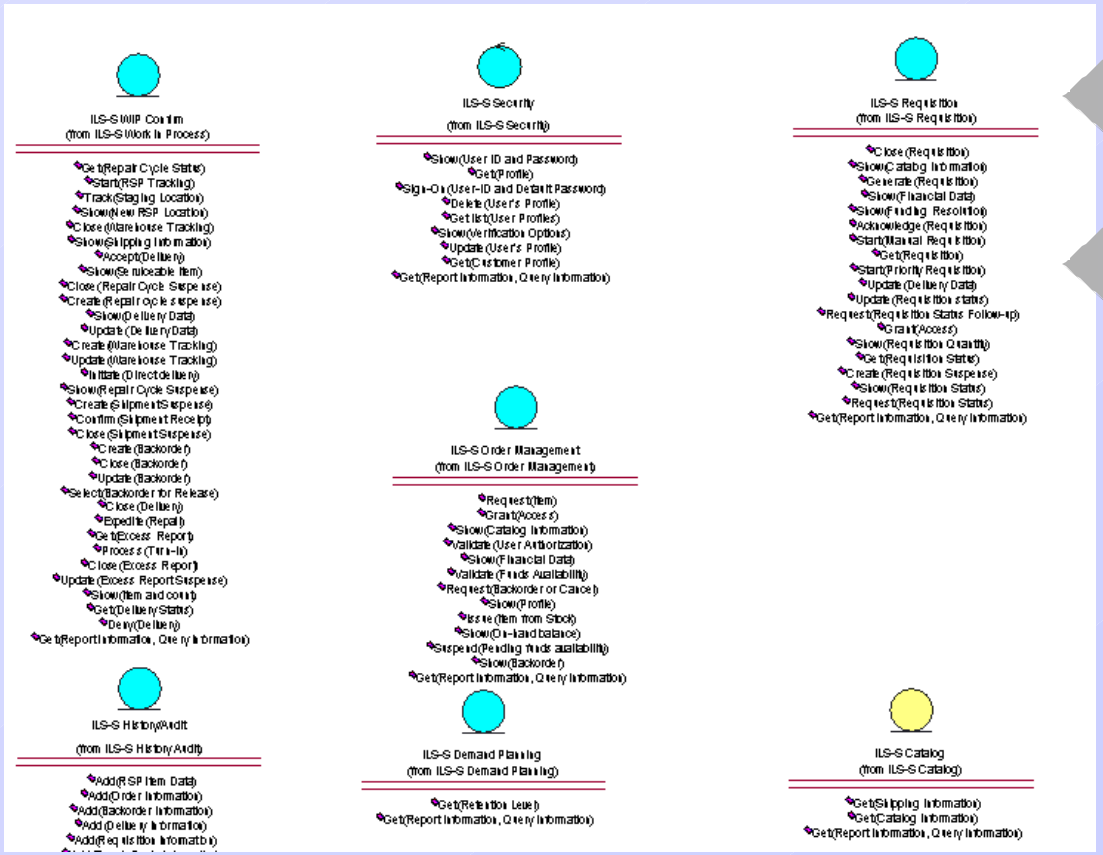
How components work together

(Order Item Use Case)



Use Case Modeling Results

Suite of components with Interfaces



Components

Interfaces

- **Interfaces define expected component behavior**

Components and Interfaces

Interfaces Define Component Behavior

- Plug and play architecture
  - Swap “approved” components in and out of scenario
  - Supports multiple modeling and visualization methods
    - Dynamic multi-scale modeling
    - Tunable rendering times
  - COTS insertion/interfaces
- Self-assembling, capability-focused applications
  - Components provide powerful toolkit
  - Use cases provide instruction manuals

Mission-focused, not platform centric

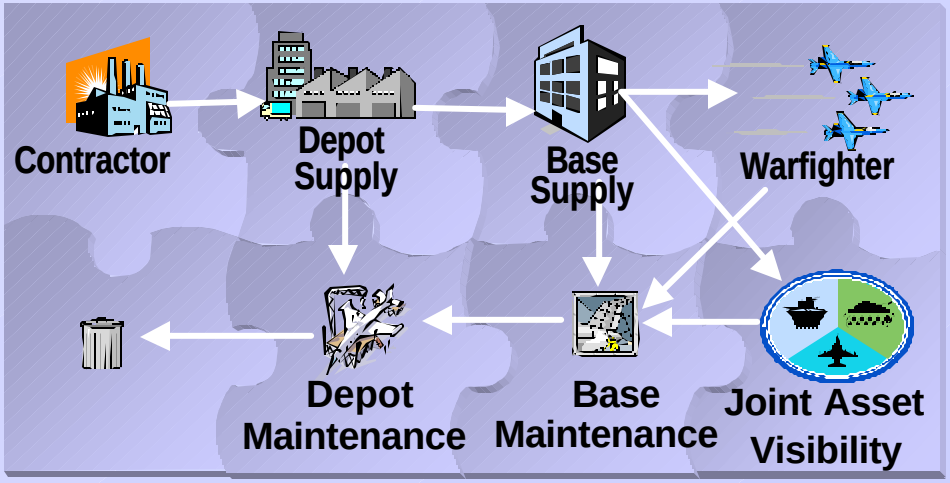
Challenges

- Human factors
  - Legacy people and legacy systems
    - Invite users to be part of change
  - Knowledge drain
    - SME and architects need to stay with projects
  - Cultural change
- Funding
  - DoD funds systems, not enterprise
  - Initial development costs significantly higher
- Technology
  - Component technology well-defined
  - Frameworks are immature
  - Emerging technologies from web services to intelligent agents to self organizing networks

But, DoD primed for transformation

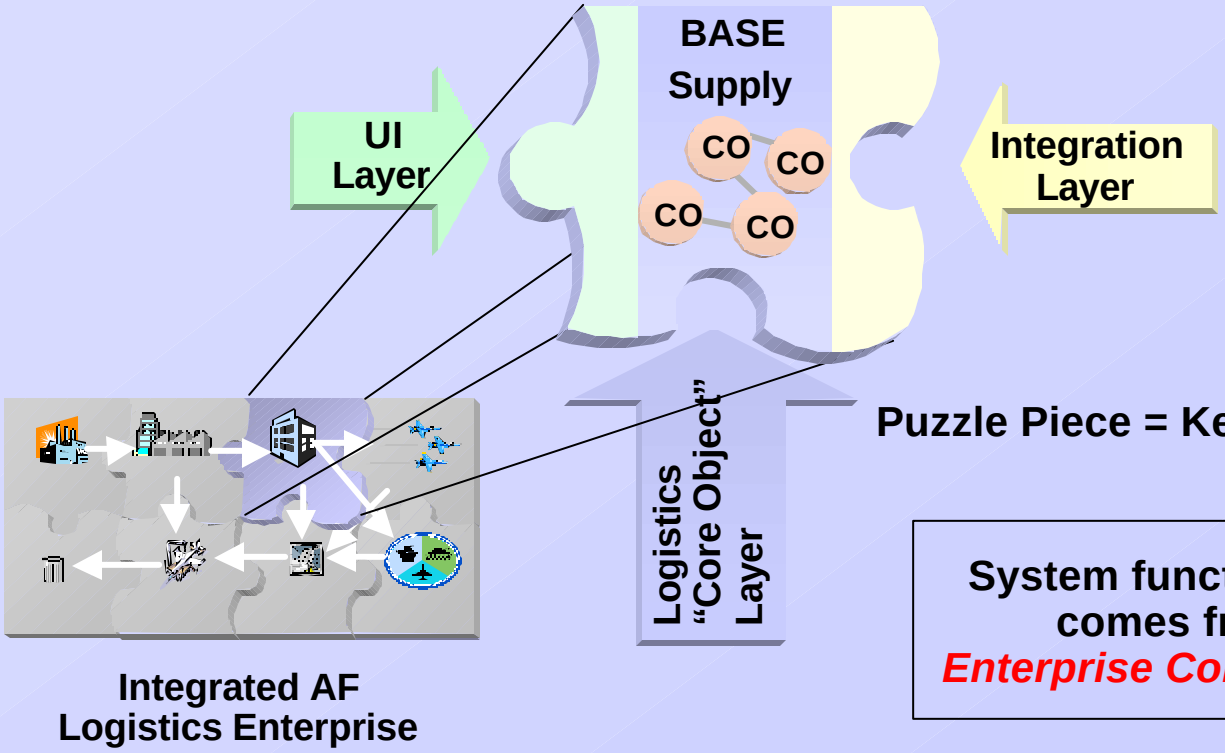
Resistance is futile

Putting it all together



Integrated Air Force Logistics Vision

Putting it all together

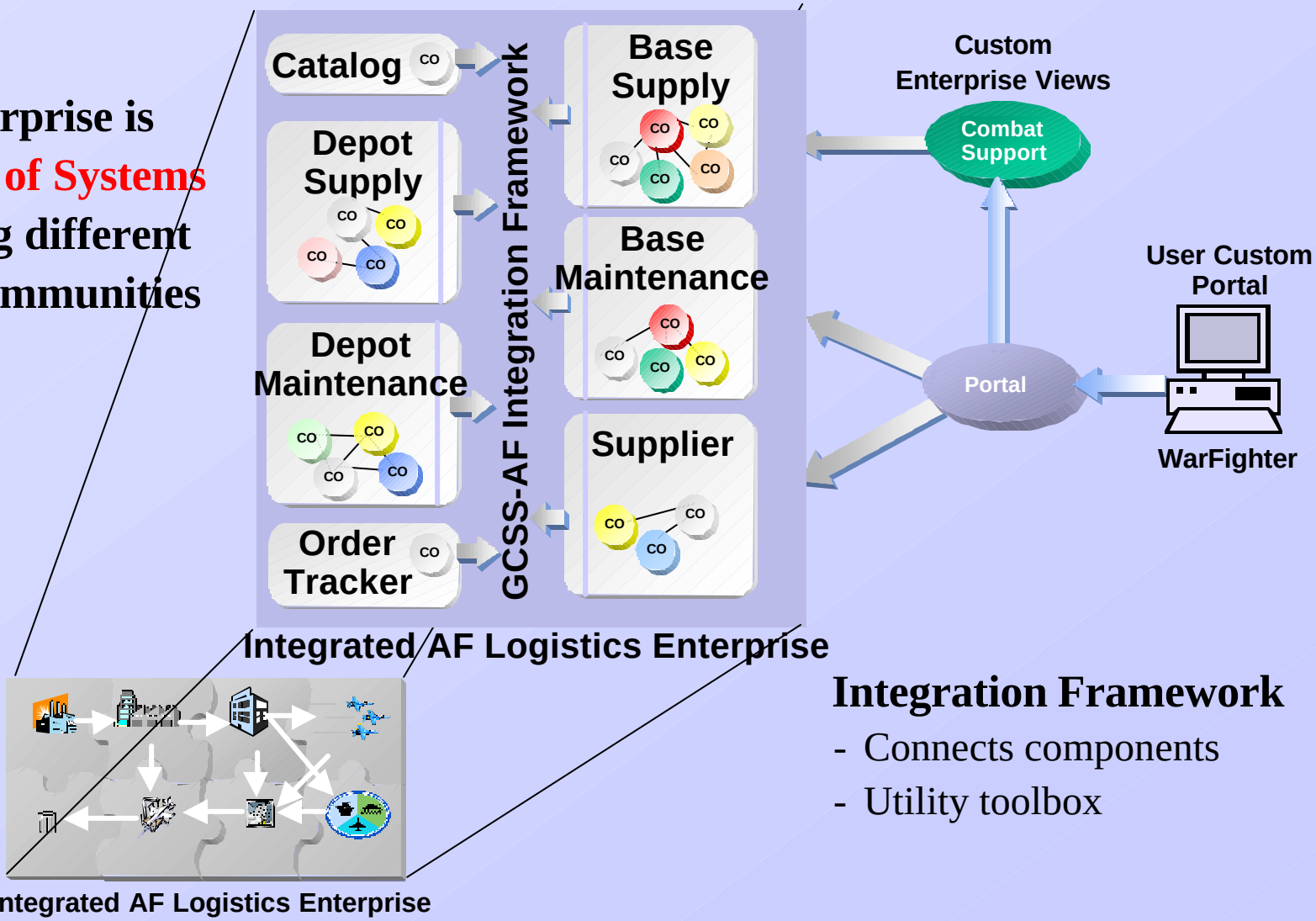


Puzzle Piece = Key Logistics *System*

System functionality comes from *Enterprise Components*

Putting it all together

Enterprise is System of Systems serving different user communities

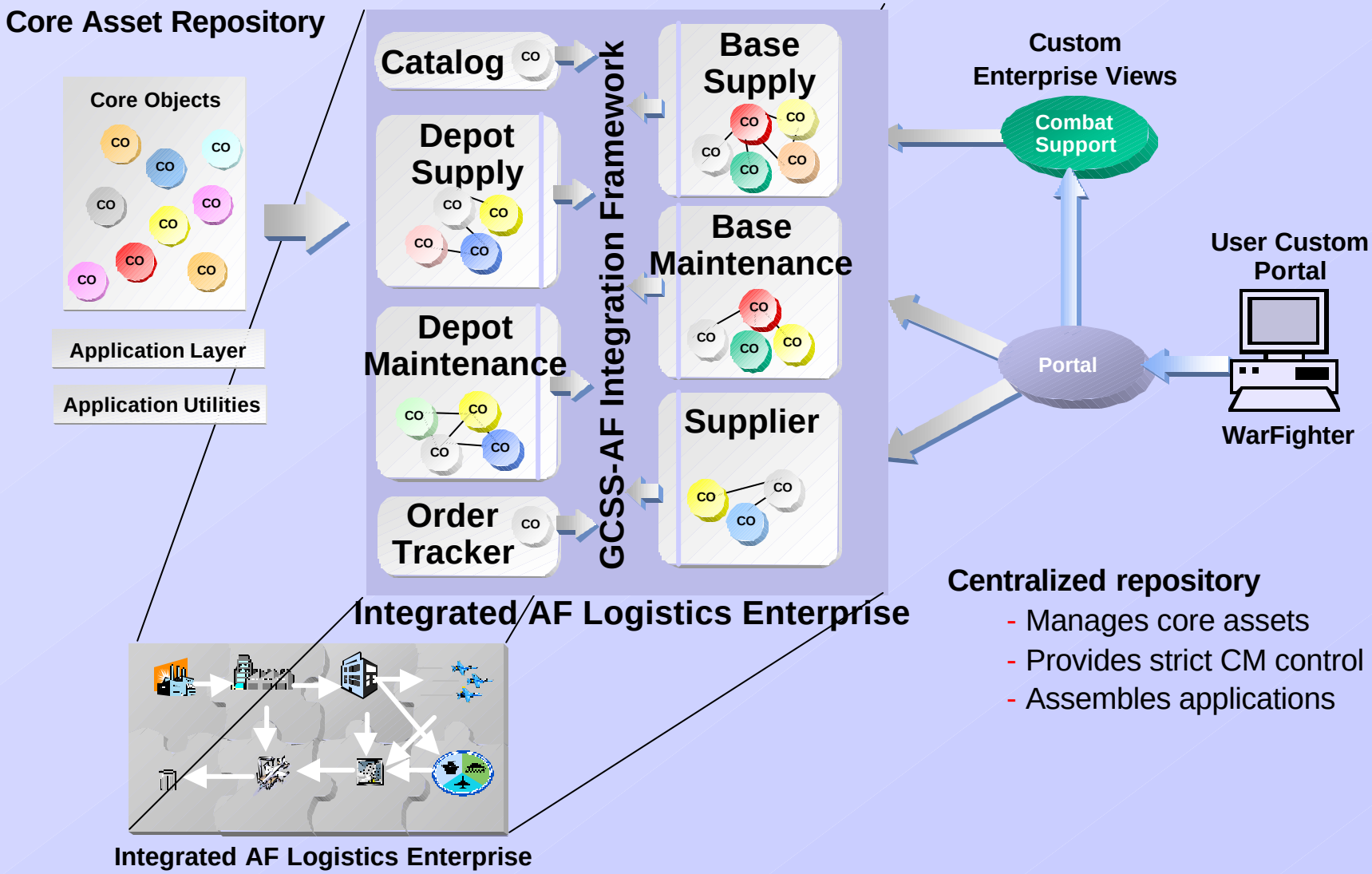


Integration Framework

- Connects components
- Utility toolbox

Putting it all together

Core Asset Repository

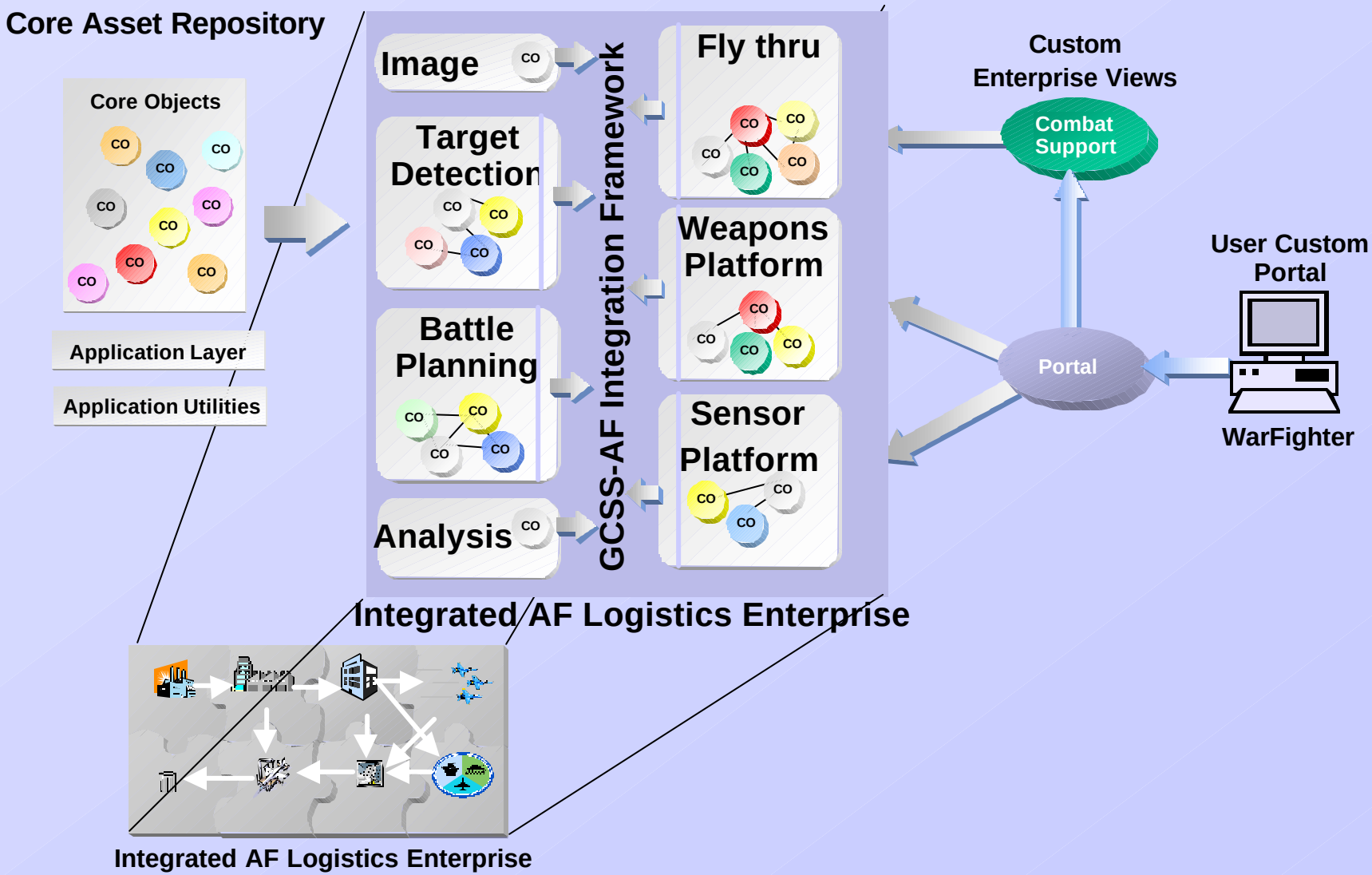


Centralized repository

- Manages core assets
- Provides strict CM control
- Assembles applications

Putting it all together

M&S Version



Key Factors

- Components represent a different way of thinking
  - Both **enterprise** and **mission-centric**
  - Collection of parts designed to work together
  - Applications assembled around requirements, then disappear
- Not just technology
  - Technical approach very mission-focused
  - Driven by **users**, not technology
- Requires architecture oversight and expertise
  - Design and configuration management key pieces

SBA Observations (Revisited)

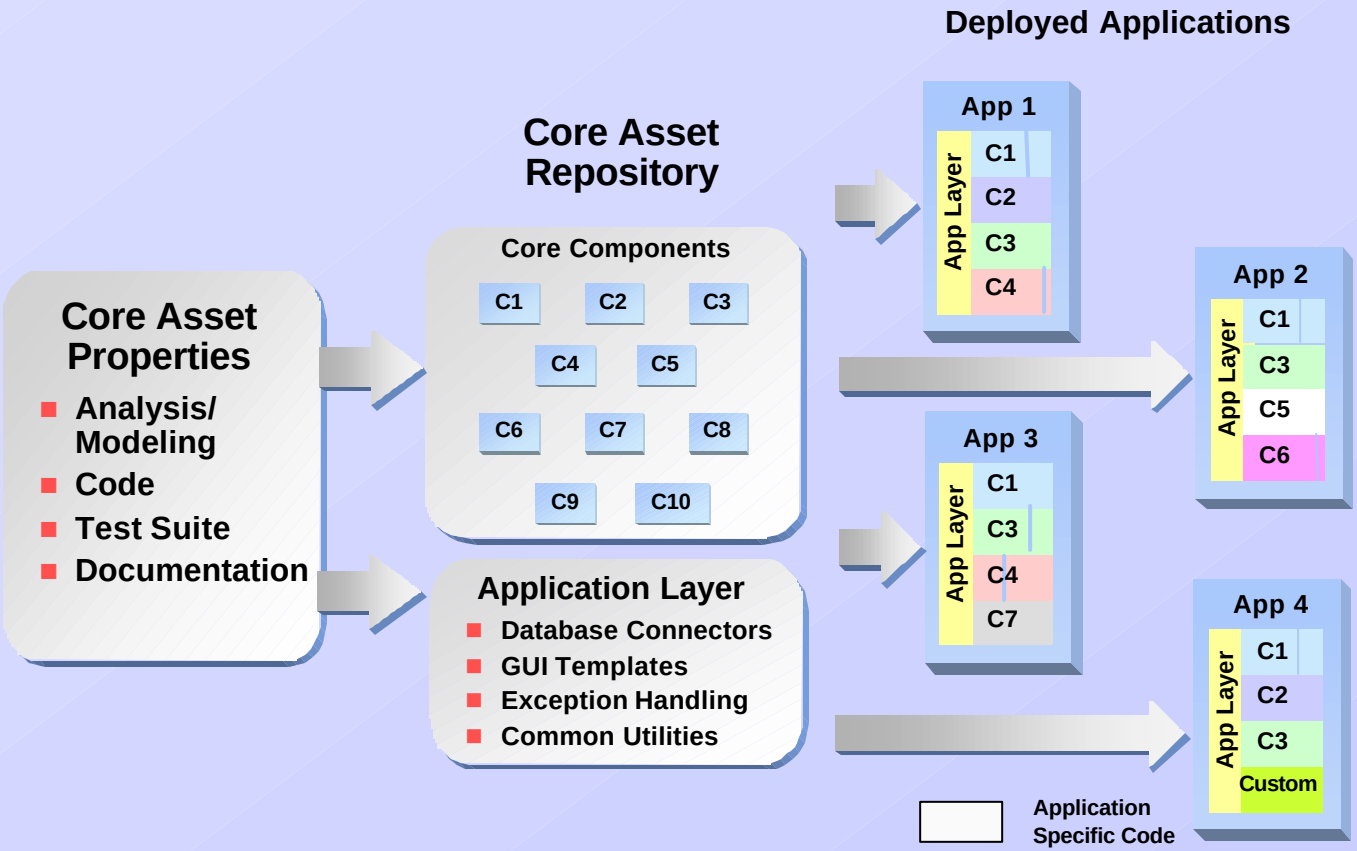
- Dr Sega
  - Platform-centric• network centric
  - Common vision representation
    - Multiple function areas
    - Joint, interoperable, re-useable models
- Dr Dahmann
  - System focus • mission focus
- Dr Castro
  - Multi-scale, multi-view
- Mr Schade
  - Seamless data exchange/interface standards
- Ms. Zimmerman
  - Rapidly composable and scalable M&S
  - Strong CM focus
  - Build only what is needed
- Mr Lunceford
  - Begin shift of M&S from craft to scientific/engineering discipline

What is a Software Product Line (SPL)?

- Software Product Line*
  - "a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way."

* From the book *Software Product Lines,* by Paul Clements and Linda Northrop

Software Product Line



Why a SPL?

- SPL provides established methodology for reusable component development across multiple applications
- Core Asset Repository extends well beyond centralized code
  - Standardized requirements for all objects
    - Interface and functional
  - Complete test cases
  - Integrated with development/CM environment
- SPL provides mechanism for formal testing and configuration management of components
  - Essential element to maintain plug-and-play capability
  - Ensures components always compatible with current architecture

Next Steps

- Embrace component-based approach
- Setup workshops to define scenarios and approach
  - Focus on few key capabilities
    - Keep others in mind
  - Determine how existing pieces fit into this approach
- Begin with a demonstrable prototype

- Onward and upward!

Thank You

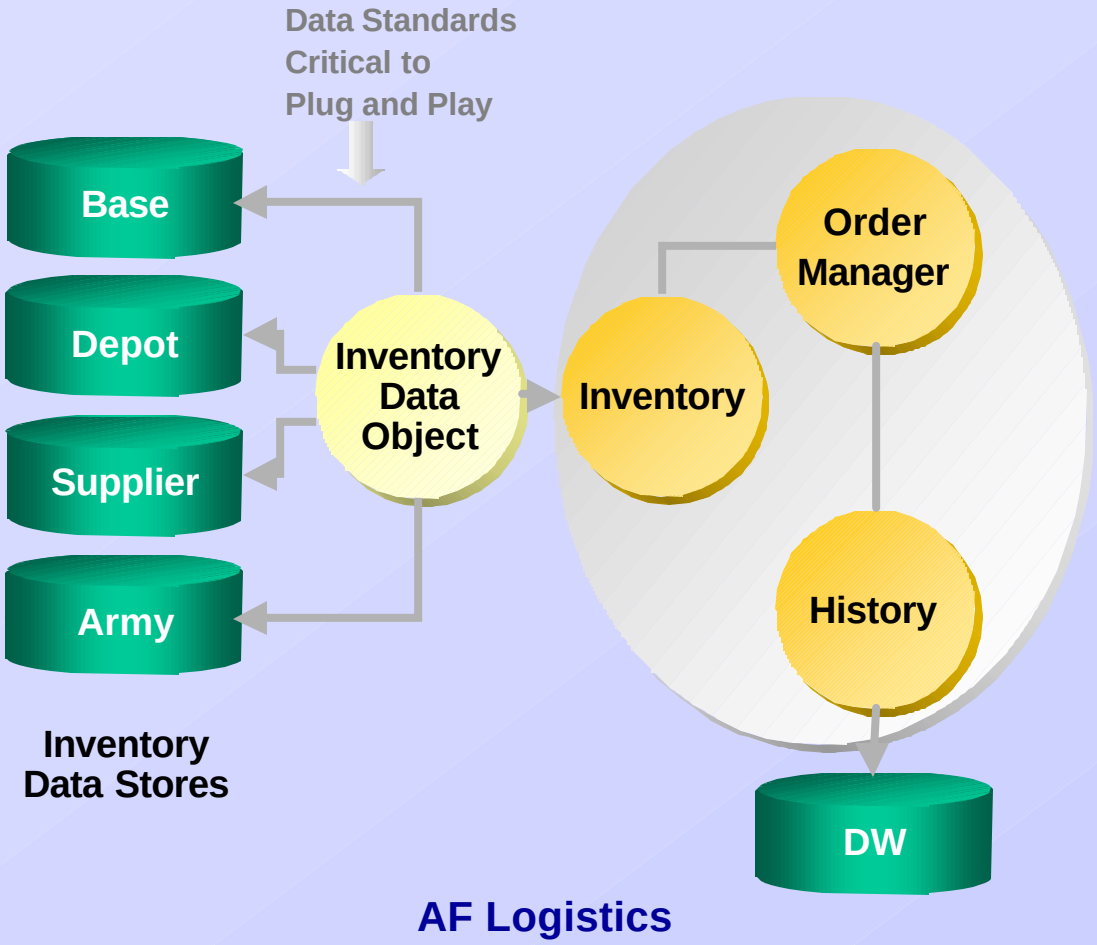
Me:

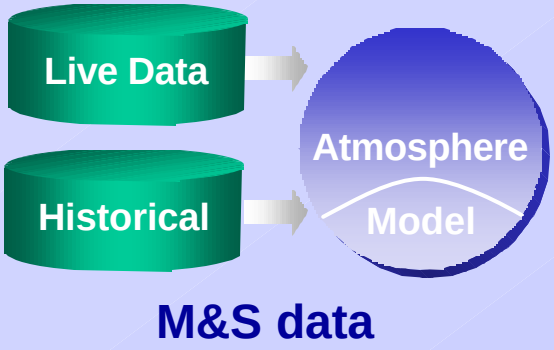
Keane, Inc
1410 Spring Hill Road
McLean, VA

Karl Garrison 703-655-5620 (C)
703-848-7200 (O)
karl_c_garrison@keane.com

But Where's The Data?

Application interfaces separate from data sources





- Data accessed through components
  - Some enterprise-wide
    - Enterprise data warehouses
  - Some local managed
    - Local operational data stores
    - Data volume
    - Security constraints