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

Integrated Logistics Enterprise

Modernize Supply Chain

Today

Tomorrow

Air Force Logistics

- Collection of stovepiped systems
- Pieces do not connect
- Picture not complete

- Integrated picture across IL
- Complete connectivity
- Total Asset Visibility (TAV)
  - factory-to-foxhole
Background Projects
3D Component-based M&S Framework

Arete Ocean Model
Ocean heights
Reflection function (point)

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

Supply Chain Vocabulary:
- AF Logistics
- Order Management
- Inventory
- Catalog
- History
- Security
- Financial

M&S Vocabulary:
- Modeling and Simulation
  - Scene Management
  - Sensor
  - Target
  - Vehicle
  - Terrain
  - Atmosphere
Component Granularity

<table>
<thead>
<tr>
<th>Granularity</th>
<th>Type</th>
<th>Example</th>
<th>Example</th>
<th># Comp.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detailed</td>
<td>Patterns</td>
<td>Database Interfaces</td>
<td>Target Signature</td>
<td></td>
</tr>
<tr>
<td>General</td>
<td>Business Objects</td>
<td>Inventory, Catalogue</td>
<td>Terrain, Sensors</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Applications</td>
<td>Base Supply Chain</td>
<td>Virtual Environment</td>
<td></td>
</tr>
</tbody>
</table>

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

User description
Users describe what they do

Application architecture
Architects map into components (with users)

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)

Supply Clerk

Orders Manager

Inventory

Audit History

Security

Catalogue

validateLogin(User)

submitOrder(Order)

checkAndResolveItems(ItemList)

ValidateOrder()

getInventory(Item)

writeTransactionHistory(Transaction)

ConfirmOrder(Order)
Use Case Modeling Results
Suite of components with 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

Integrated AF Logistics Enterprise

UI Layer

BASE Supply

Integration Layer

Puzzle Piece = Key Logistics System

System functionality comes from Enterprise Components
Putting it all together

Enterprise is System of Systems serving different user communities

Integrated AF Logistics Enterprise

Integration Framework
- Connects components
- Utility toolbox
Putting it all together
Core Asset Repository

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

Integrated AF Logistics Enterprise

Catalog
Depot
Base
Depot
Depot
Supply
Supply
Maintenance
Maintenance

Integrated AF Logistics Enterprise

Application Layer
Application Utilities

Core Asset Repository
Core Objects
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

Core Asset Properties
- Analysis/Modeling
- Code
- Test Suite
- Documentation

Core Components

App Layer
- Database Connectors
- GUI Templates
- Exception Handling
- Common Utilities

Deployed Applications

App 1
- C1
- C2
- C3
- C4

App 2
- C1
- C3
- C5
- C6

App 3
- C1
- C3
- C4
- C7

App 4
- C1
- C2
- C3
- Custom

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