What do “Services” Look Like in a Service Oriented Architecture?  
The Role of COTS  

Kevin B Kreitman, PhD  

The Aerospace Corporation  

March 2006
Information Systems Evolution

Responsiveness Requirements

Leadership

Value Net

Bureaucracy

Quality Improvement

Total Quality

Reengineering

Value Nets

Enterprise Architecture

Supply Chain Integration

Cross-Enterprise Architecture

Balanced Scorecard

Total Quality

Quality Improvement

Stovepipe mega-apps (ERP)

Stand-alone tools

Mainframe apps

1975-2004

THE AEROSPACE CORPORATION
Information Systems Evolution

- **Responsiveness Requirements**
  - **Leadership**
  - **Value Net**
  - **Bureaucracy**
  - **Quality Improvement**
  - **Total Quality**
  - **Reengineering**

- **Value Nets**

- **Mainframe apps**
  - **E-gov (agencies)**

- **Stovepipe mega-apps (ERP)**

- **1975-2004**
  - **Stand-alone tools**
  - **SOA**
  - **EAI**
  - **CORBA**
  - **RDBMS**
  - **Balanced Scorecard**
  - **Reengineering**
  - **Total Quality**
  - **Quality Improvement**
  - **Supply Chain Integration**
  - **Value Net**
  - **Cross-agency collaboration**
  - **Cross-Enterprise Architecture**
  - **NCOW**
  - **GPRA**
  - **EAI**
  - **SOA**
  - **CORBA**
  - **RDBMS**

**THE AEROSPACE CORPORATION**
Before SOAs…

• Stand-alone applications (mainframe…)
• Stovepiped applications (client-server…)
• Web-ified applications (web-based client-server)
Stovepiped “Web-ified” Applications on the Network

- Source of Directory Information
- User
- Network

- Presentation logic
- Directory
- RDBMS
- Document repository
- Database

- Maintains own ID/Password access
- Search engine
- Calendar
- Email
- Notifications

- DM
- RM
- WF
The Original Plan

• Highly granular, independent, composable services.

• Independently coded or configured.

• Aggregated “on the fly” or on short notice to produce end-to-end business process executable.

• Registered to find and reuse on the fly.

• Construct Web Services environment from open source components available on open platforms.
Service Oriented Architecture
Granular Services

Some information bus common services

Notifications
Data services
Security
Workflow (BPM)
Directory Services

Common (standard-based) messaging services

Network

(WS, JMS, other MOM)

Connect with other SOA / platforms
The Emerging Experience

• COTS Super-platforms enable coherent construction (proprietary elements included)
  – Advantage is rapid set-up, management. (This is harder than it looked…)

• Aggregation on the fly is a dream
  – At least partially due to trust issues and governance.

• Granularity is a double-edged sword
  – Requires agreement on “component” level architecture
  – Requires programming and Web Services expertise
  – Modern large COTS applications are offered with “Web Services Interfaces”
Superplatform with COTS Apps as Services

Common (standard-based) messaging services

Network

(WS, JMS, other MOM)
Advantages Of COTS

• Configure, don’t code

• Set-up and management
  – Time
  – Complexity
  – Accountability and assuredness

• Supported applications:
  – Protection from legal actions
  – Help and contractor/vendor experience
Example

• Custom-developed information analysis/selection application in complex, standalone DB application

• Redesigned as set of common and analytic services

• Discovered that common services could be provided by Super-Platform and approved ID management

• Discovered that analytic services could be configured and maintained in rules engine/BI application.
Samples of COTS Apps

• Content Management/lightweight Document Management

• Business Intelligence/Rules Engines

• Super Platforms (may include Business Process Management, CM, ID Management, Business Rules, Data Management, Service Registration, messaging, management dashboards, service agreement management, automated metrics . . .)
Observations

- Appropriate COTS can dramatically reduce development costs
  - But be careful of “false analogies”

- Super Platforms have made execution and management of SOAs realistic.
  - reduced complexity
  - increased transparency
  - reduced developer expertise requirements

- Critical for “WS” COTS components to
  - Have designed-in interfaces for WS compatibility
  - Use external Directories for ID management and access control (per emerging GIG-IA/Federal security models)