

From Architecture to Operations

Using Your Architecture Work in Operations

STEVEN H. DAM, PH.D., ESEP
CHRIS RITTER
SPEC INNOVATIONS
STEVEN.DAM@SPECINNOVATIONS.COM



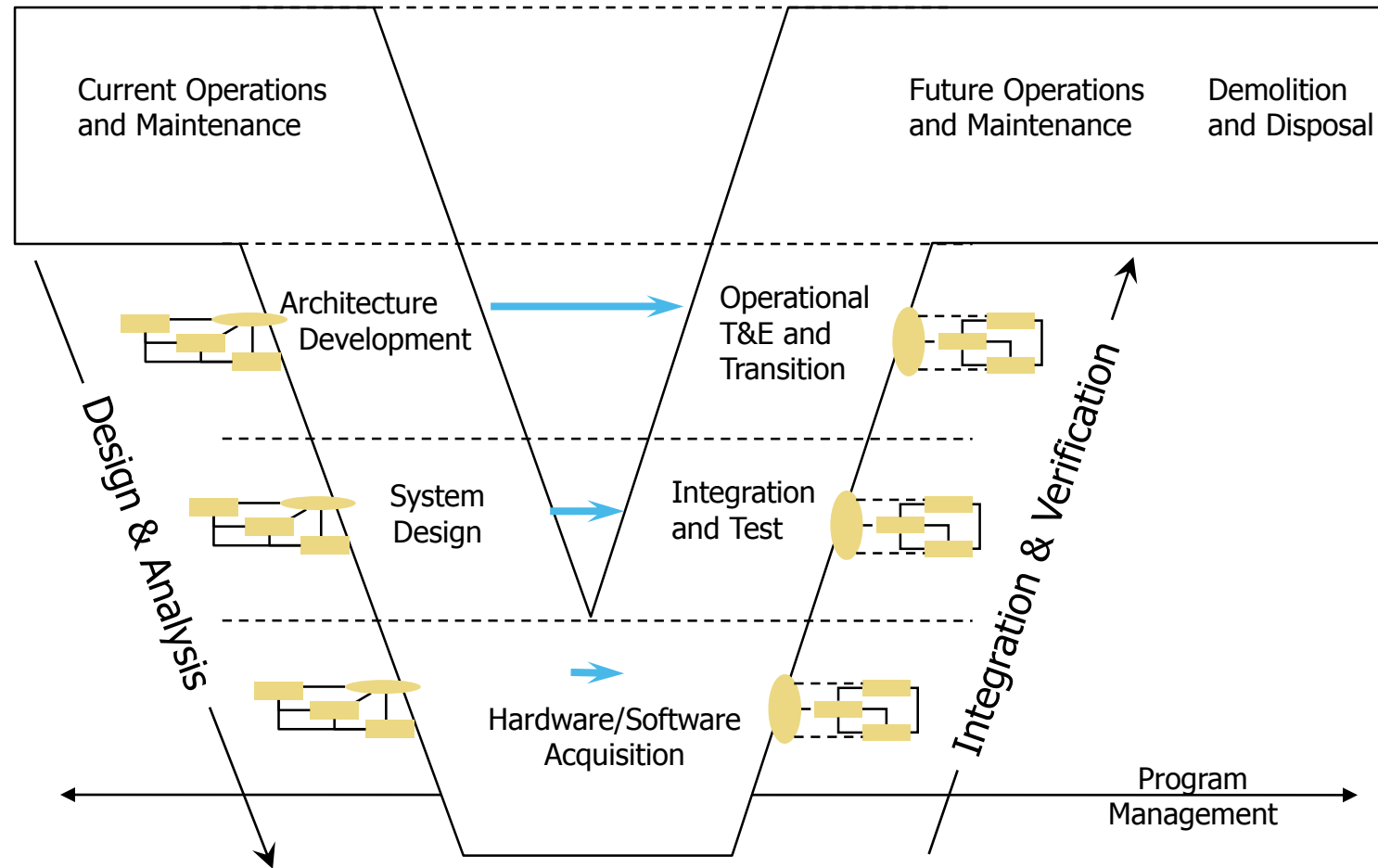
Outline

1. What Do We Do When We Develop Architectures?
2. How Is Architecture Useful to Operators?
3. How Can We Use Operational Information to Prepare for the Next Architecture?

1. WHAT DO WE DO WHEN WE DEVELOP ARCHITECTURES?



Product Lifecycle



Capturing As-Is Architecture Data

- Operational equipment
- Networks and other interfaces
- Operating processes and procedures
- Operating costs
- Operations personnel

We capture information that describes how current systems operate and where problems may occur

Developing To-Be Architecture Data

- Designs for new and modified equipment
- Designs of new and modified processes
- Plans for acquiring equipment
- Training needs

All this information will affect the operations of the as-is systems

When We Finish, What Do We Have?

- A complete database with all the information about the systems that operators (and maintainers) need to effectively and efficiently operate the systems

Why not put all that data to good use?

2. HOW IS ARCHITECTURE USEFUL TO OPERATORS?



How Can Architecture Be Useful

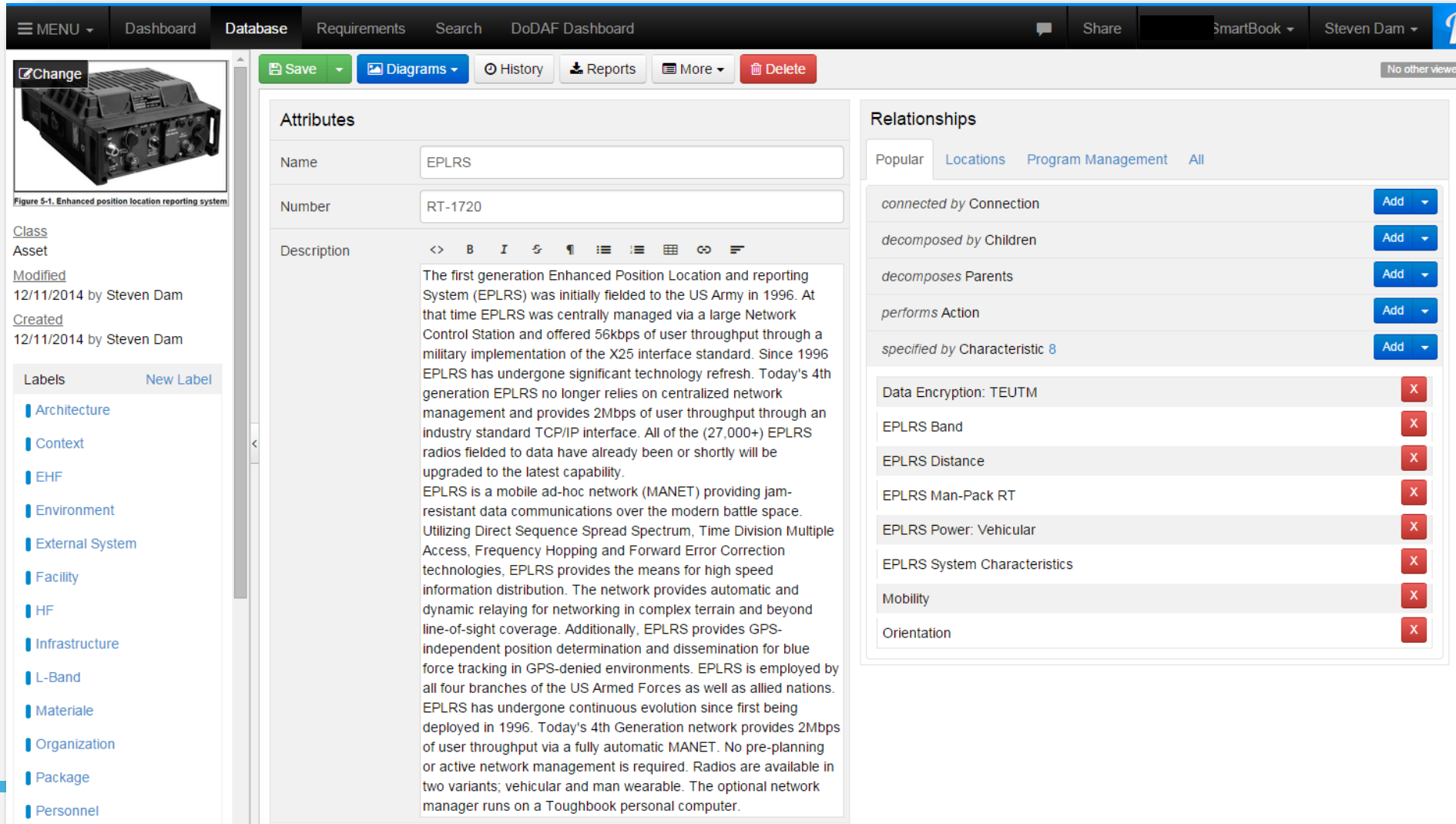
- For operators, architecture can provide the basis for understanding the limitations of and standard operating procedures (SOP) for using the components
- For maintainers, the architecture can provide a place to store critical information, such as reliability, availability, and maintenance procedure, for the components
- However, often this kind of information is either ignored by architects or occurs later in the lifecycle, after the architecture database has been ignored for years
- Too often architecture teams do not include the operational and maintenance perspectives

Capturing Operational Data

- Use cases, user stories, scenarios, threads, etc. are all ways to capture how a system is used or operated
- One way we capture this information is in a Concept of Operations (CONOPS)
 - However, most CONOPS are too high level and do not analyze the scenarios sufficiently to make sure they make any sense or even work in a realistic environment
 - An architecture-based CONOPS is then essential for this
 - Such a CONOPS can then form the basis for developing SOPs

- CONOPS ASSE Outline
 - - ❖ 0. [Version History](#)
 - ❖ 1. [Executive Summary](#)
 - ❖ 2. [Mission Description](#)
 - ❖ 2.1 ["As-Is" Mission Description](#)
 - ❖ 2.1.1 [Goals and Objectives](#)
 - ❖ 2.1.2 [Underlying Mission and Business Rationale](#)
 - ❖ 2.1.3 ["As-Is" Architecture](#)
 - ❖ 2.1.4 [Key Stakeholders and Expectations](#)
 - ❖ 2.1.5 ["As-Is" Gaps](#)
 - ❖ 2.2 ["To-Be" Architecture](#)
 - ❖ 2.2.1 ["To-Be" Operational Concept](#)
 - ❖ 2.2.2 ["To-Be" Functional Model](#)
 - ❖ 2.2.3 [Anticipated Performance Impacts of the "To-Be" Architecture](#)
 - ❖ 3. [System Operational Context and Reference Operational Architecture](#)
 - ❖ 3.1 [System Operational Context](#)
 - ❖ 3.2 [Reference Operational Architecture](#)
 - ❖ 4. [System Drivers and Constraints](#)
 - ❖ 4.1 [Performance Drivers and Constraints](#)
 - ❖ 4.2 [Constraints Resulting from Existing Systems and Infrastructure](#)
 - ❖ 4.3 [Doctrine](#)
 - ❖ 4.4 [Organizational Roles and Responsibilities](#)
 - ❖ 4.5 [Regulatory Requirements](#)
 - ❖ 5. [Operational Scenarios](#)
 - ❖ 5.1 [Scenario Characteristics and Descriptions](#)
 - ❖ 5.2 [Applying the Characteristics to Operational Scenarios](#)
 - ❖ 5.3 [Scenario Models and Validation](#)
 - ❖ 6. [Implementation Concepts Selected and Rationale](#)
 - ❖ 7. [Proposed System Operational Architecture](#)
 - ❖ 8. [Organizational and Business Impact](#)
 - ❖ 8.1 [Impact of the Changed Operational Architecture on Policies](#)
 - ❖ 8.2 [Impact of the Changed Operational Architecture on Processes and Procedures](#)
 - ❖ 8.3 [Impact of the Changed Operational Architecture on Organizations and Facilities](#)
 - ❖ 8.4 [Impact of the Changed Operational Architecture on Skills and Competencies](#)
 - ❖ 8.5 [Impact of the Changed Operational Architecture on Workload](#)
 - ❖ 9. [Risks and Technology Readiness Assessments](#)
 - ❖ 9.1 [Risks](#)
 - ❖ 9.2 [Technology Readiness Assessments](#)
 - ❖ [Appendix A: Concept Of Operations Document Approval](#)
 - ❖ [Appendix B: References](#)
 - ❖ [Appendix C: Key Terms](#)

Information Capture Example



The screenshot displays a software interface for managing an asset. The main content area is divided into three sections: Attributes, Relationships, and a Metadata sidebar.

Attributes:

- Name: EPLRS
- Number: RT-1720
- Description: The first generation Enhanced Position Location and reporting System (EPLRS) was initially fielded to the US Army in 1996. At that time EPLRS was centrally managed via a large Network Control Station and offered 56kbps of user throughput through a military implementation of the X25 interface standard. Since 1996 EPLRS has undergone significant technology refresh. Today's 4th generation EPLRS no longer relies on centralized network management and provides 2Mbps of user throughput through an industry standard TCP/IP interface. All of the (27,000+) EPLRS radios fielded to data have already been or shortly will be upgraded to the latest capability. EPLRS is a mobile ad-hoc network (MANET) providing jam-resistant data communications over the modern battle space. Utilizing Direct Sequence Spread Spectrum, Time Division Multiple Access, Frequency Hopping and Forward Error Correction technologies, EPLRS provides the means for high speed information distribution. The network provides automatic and dynamic relaying for networking in complex terrain and beyond line-of-sight coverage. Additionally, EPLRS provides GPS-independent position determination and dissemination for blue force tracking in GPS-denied environments. EPLRS is employed by all four branches of the US Armed Forces as well as allied nations. EPLRS has undergone continuous evolution since first being deployed in 1996. Today's 4th Generation network provides 2Mbps of user throughput via a fully automatic MANET. No pre-planning or active network management is required. Radios are available in two variants; vehicular and man wearable. The optional network manager runs on a Toughbook personal computer.

Relationships:

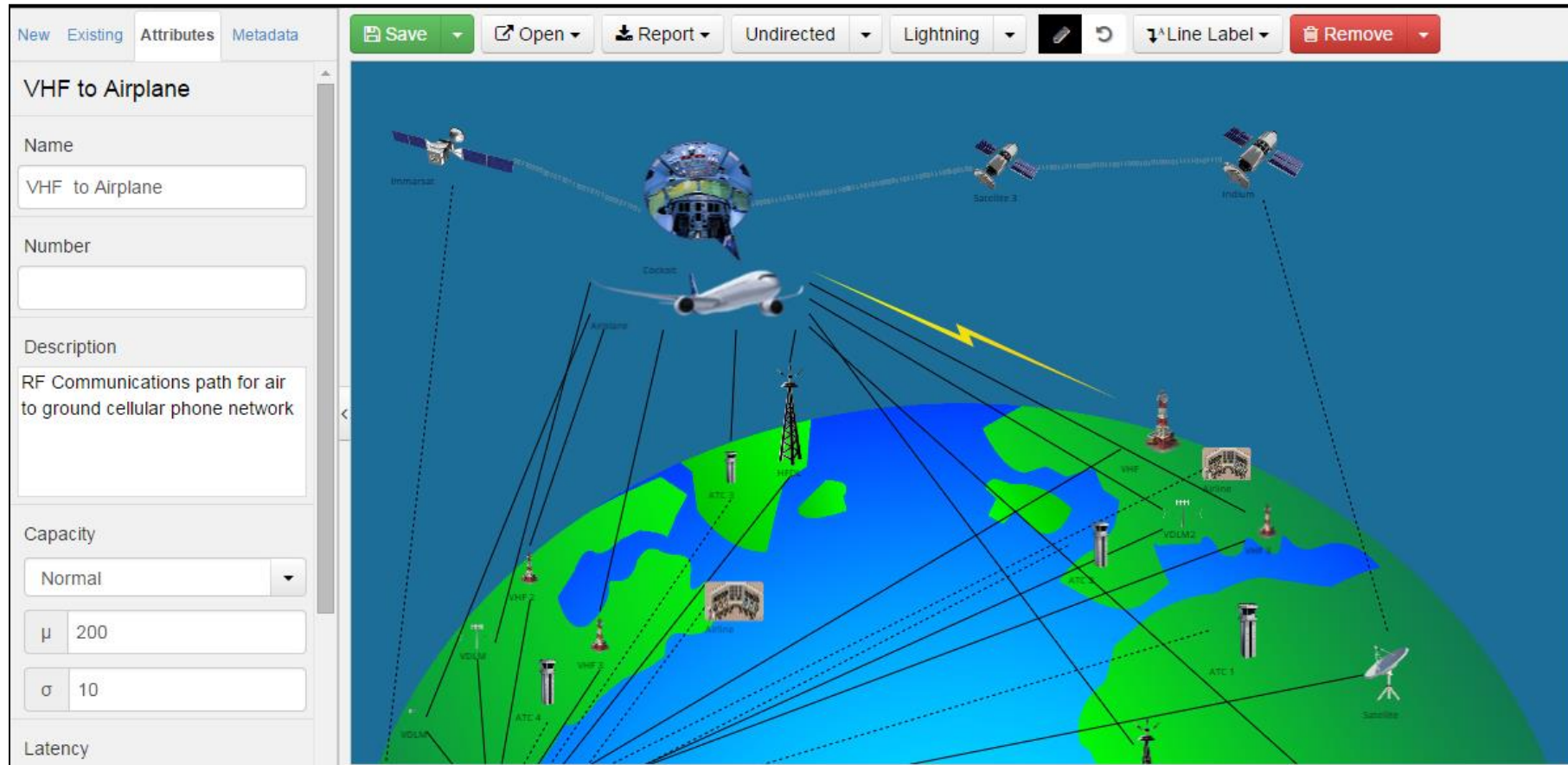
- connected by Connection (Add)
- decomposed by Children (Add)
- decomposes Parents (Add)
- performs Action (Add)
- specified by Characteristic 8 (Add)
- Data Encryption: TEUTM (X)
- EPLRS Band (X)
- EPLRS Distance (X)
- EPLRS Man-Pack RT (X)
- EPLRS Power: Vehicular (X)
- EPLRS System Characteristics (X)
- Mobility (X)
- Orientation (X)

Metadata Sidebar:

- Class: Asset
- Modified: 12/11/2014 by Steven Dam
- Created: 12/11/2014 by Steven Dam
- Labels: Architecture, Context, EHF, Environment, External System, Facility, HF, Infrastructure, L-Band, Materiale, Organization, Package, Personnel

- MBSE tool provides means to capture and relate information about any asset
- This would provide a means to capture and share information about the systems and components

Visualizing the Comms Architecture Example



- Physical architecture captured in MBSE tool
- Provides constraints for functional modeling and simulation

Next Steps

- Similar information and documentation can support the maintainers
 - However, we still need a database for the information that can then be reused for each perspective
 - Most tools currently focus on only one lifecycle phase (e.g., architecture, operations, or maintenance), but the overlapping information clearly means that we could use a single database with multiple views
 - Here is where the concept of MOSA helps us
-

Modular Open Systems Approach (MOSA)

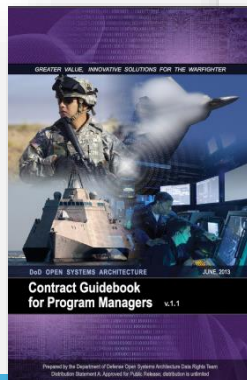
Distribution Statement A: OSA Contract Guidebook v.1.1
 Approved for Public Release; distribution is unlimited June 2013

EXECUTIVE SUMMARY

This Open Systems Architecture Contract Guidebook provides contract language targeted at the needs of the Program Manager. This document contains the basic elements to capture the benefits of an open architecture and an open business model. The essence of Open Systems Architecture (OSA) is organized decomposition, using carefully defined execution boundaries, layered onto a framework of software and hardware shared services and a vibrant business model that facilitates competition. OSA is composed of five fundamental principles:

1. Modular designs based on standards, with loose coupling and high cohesion, that allow for independent acquisition of system components;
2. Enterprise investment strategies, based on collaboration and trust, that maximize reuse of proven hardware system designs and ensure we spend the least to get the best;
3. Transformation of the life cycle sustainment strategies for software intensive systems through proven technology insertion and software product upgrade techniques;
4. Dramatically lower development risk through transparency of system designs, continuous design disclosure, and Government, academia, and industry peer reviews;
5. Strategic use of data rights to ensure a level competitive playing field and access to alternative solutions and sources, across the life cycle.

A mandate of OSA is that technical requirements be based to the maximum extent practicable on open standards. Where there are no standards, the OSA methodology creates them. At a minimum, technical standards and related specifications, requirements, source code, metadata, interface control documents (ICDs), and any other implementation and design artifacts that are necessary for a qualified contractor to successfully perform development or maintenance work for the Government are made available throughout the life cycle.



OSA Contract Guidebook for PMs

“The essence of Open Systems Architecture (OSA) is organized decomposition, using carefully defined execution boundaries, layered onto a framework of software and hardware shared services and a vibrant business model that facilitates competition.”



Defining Modular Open Systems Architecture



What: A technical architecture that leverages technical standards to support a modular, loosely coupled and highly cohesive system structure

How: Customer definition and ownership of product architecture; publication of key interfaces within the system

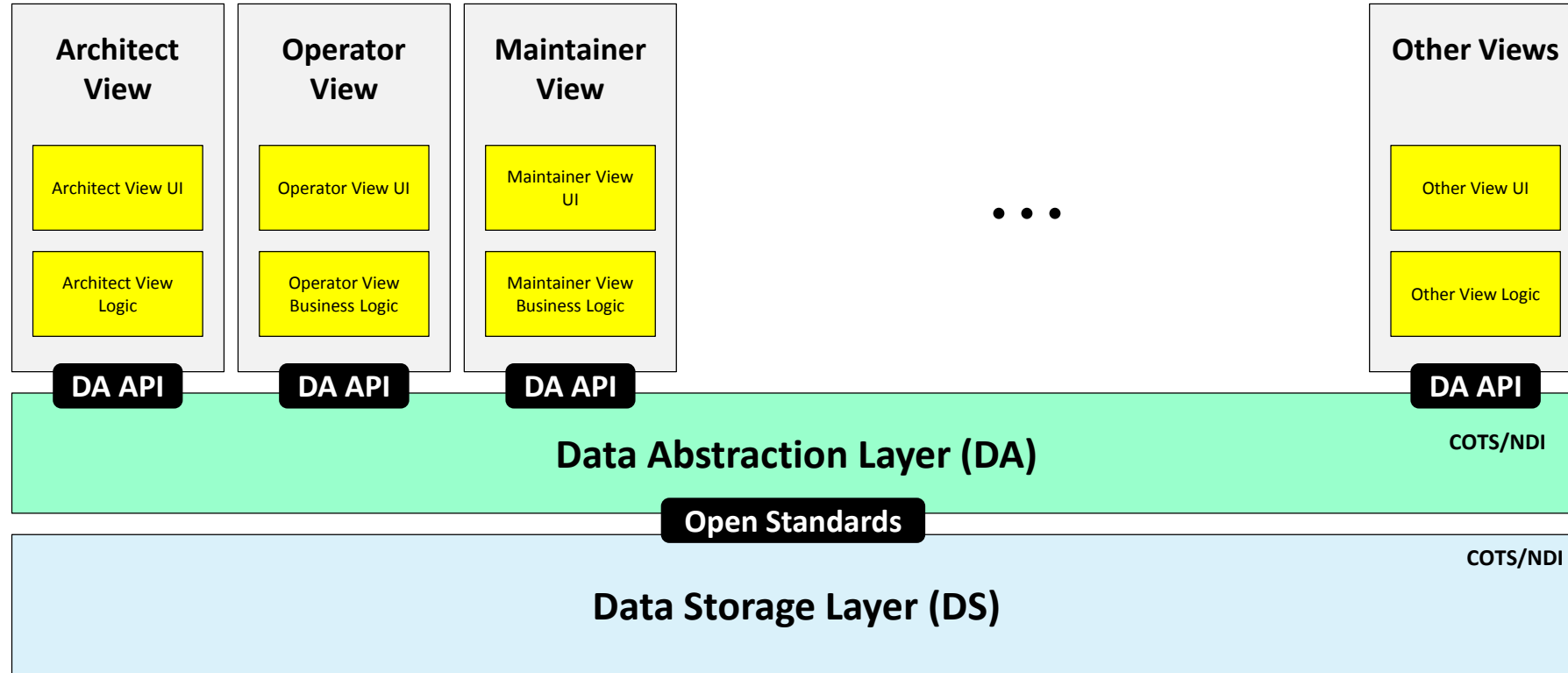
Why: Enables Open, Competitive Business Model – allowing components to be added, modified, replaced, removed or supported by different vendors throughout the life cycle – driving opportunities to enhance competition and innovation

OPNAV SE Conference
 1020141 Page 3

Distribution Statement A – Approved for public release by DCP/ISR. Distribution is unlimited.

**“Modular Open Systems Architecture
 in DoD Acquisition”
 Mr. Stephen Welby, DASD(SE)**

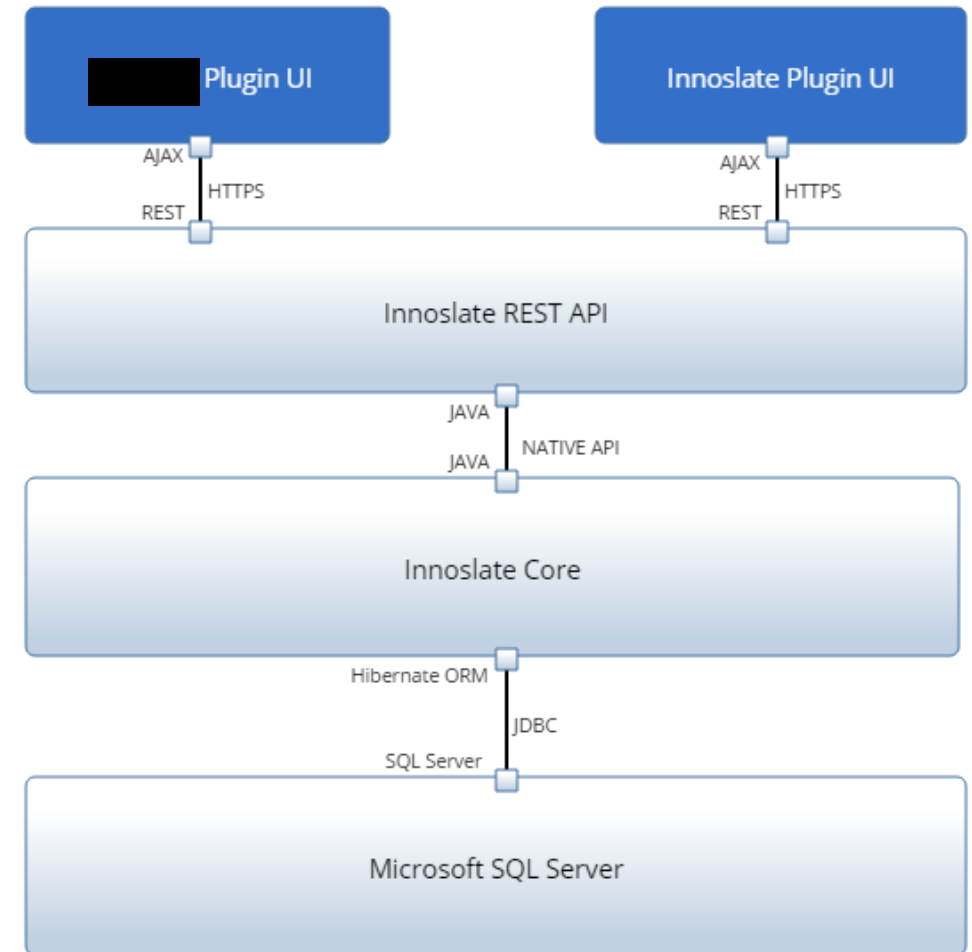
A MOSA Reference Architecture



Reference Architecture (RA) supports MOSA objectives through defined interfaces, open standards, and a severable modular design

Plugin Architecture Example

- Plugins are viewpoints of the Innoslate database
- Plugin features
 - Not a standalone application (requires Innoslate Core)
 - All authentication is through Innoslate Core with the options for:
 - Single-Sign-On CAC (Default)
 - Native Email/Password (Optional)
 - LDAP (Optional)
 - All data is stored in the U.S. Government managed MSSQL database using Innoslate Core
 - Innoslate REST API facilitates plugin data exchange



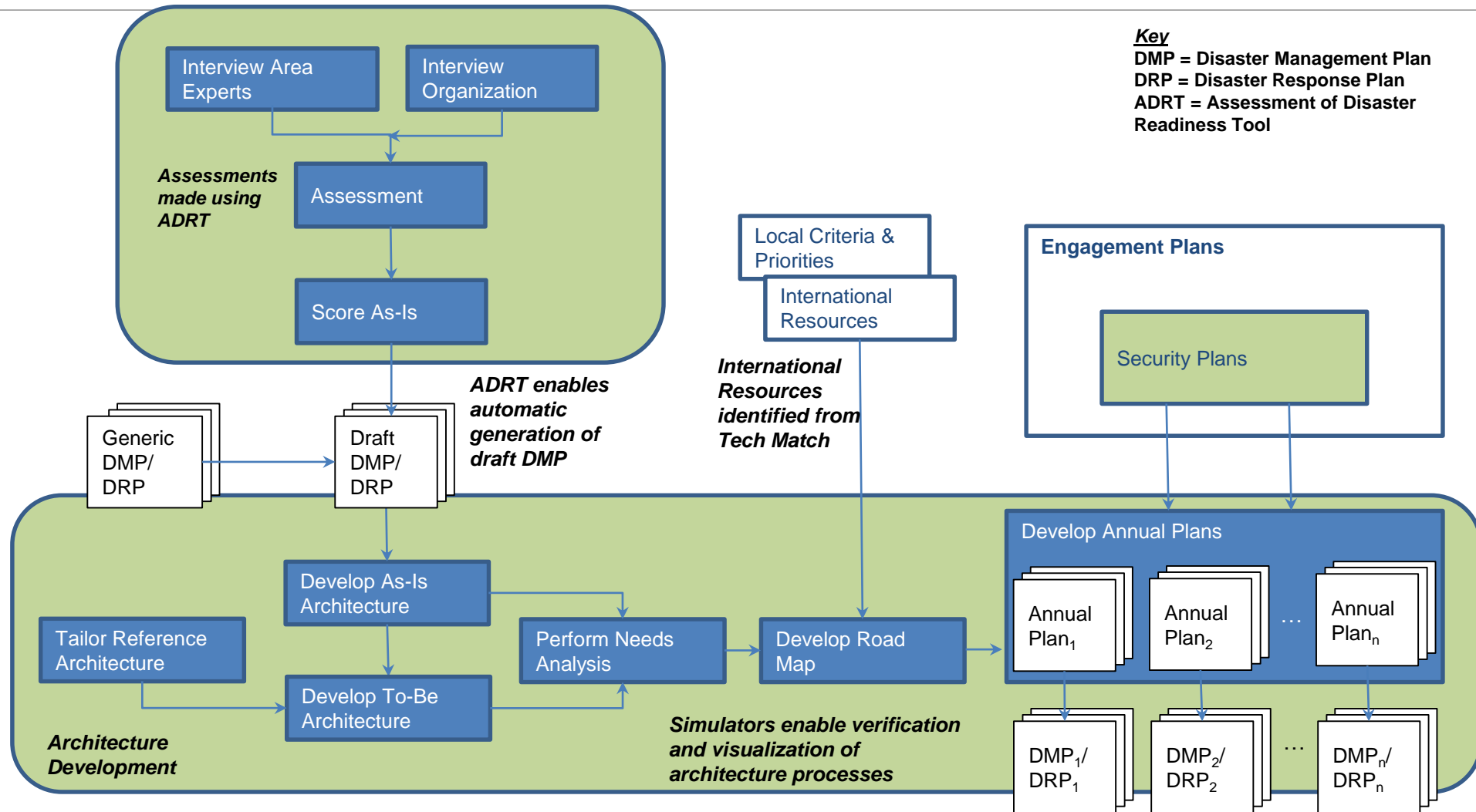
3. HOW CAN WE USE OPERATIONAL INFORMATION TO PREPARE FOR THE NEXT ARCHITECTURE?



Operational Data Utility

- Operational data can be used to identify problems areas and user needs
 - Issues may provide a window into the kinds of problems users face or unclear features
 - Feature suggestions may also be "hidden" in issues
 - Maintenance logs show where components do not meet their expected lifetimes or where they exceed them
- This data forms the basis for gap analysis that can then be turned into architecture improvements
- This approach requires the creation and maintenance of a reference architecture

Use of Other Tools to Feed Architecture Work



Summary

- Architecture work needs to include the perspectives of operators and maintainers
- We need to get more out of the architecture work that's performed
- Reusing and evolving the architecture databases into operational tools is essential to obtaining the full value for this work