Open Systems Architecture: Progress and Challenges

Forrest Shull, Harry Levinson, Thomas DuBois

Michael Bandor, Michael McLendon, Doug Schmidt

Software Solutions Conference 2015

November 16–18, 2015

Software Engineering Institute | Carnegie Mellon University

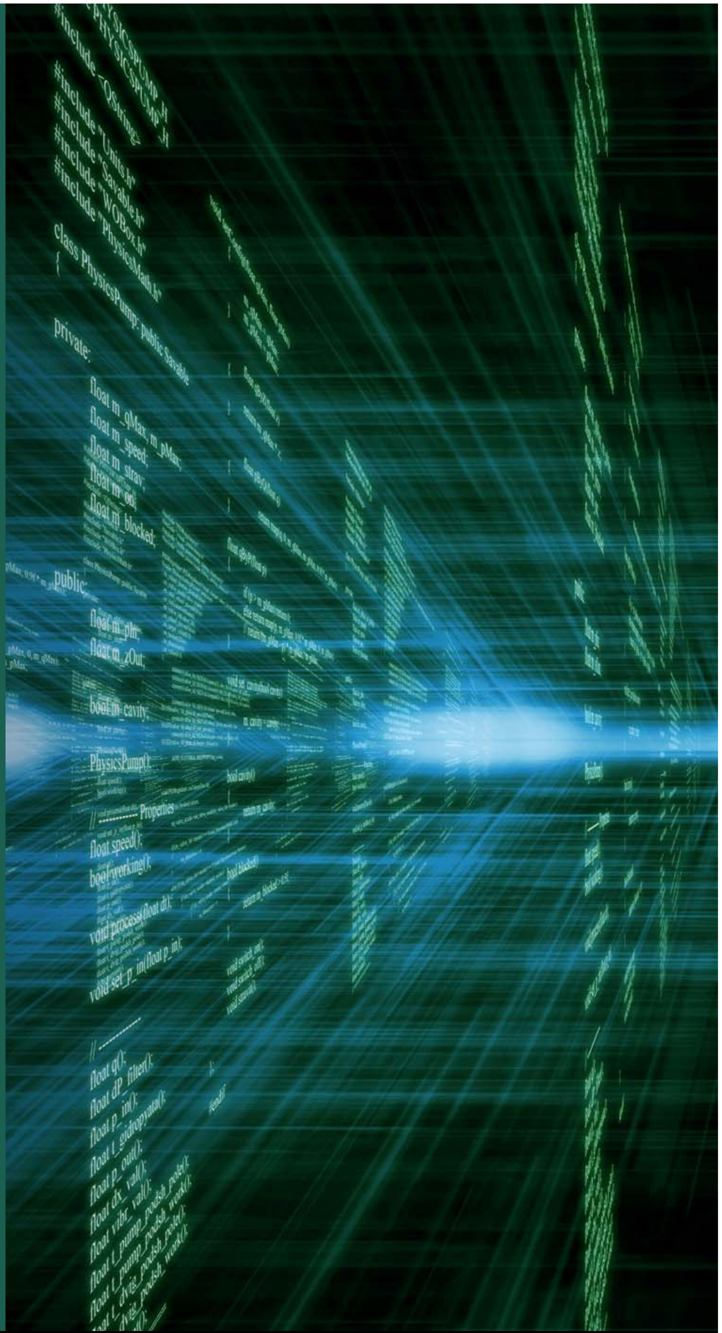
Agenda

Background

Panel Introduction and Discussion

Open Systems Architecture: Progress & Challenges

Background



What is the DoD facing?

DoD wants

- faster response to changes in threat
- technology for improved performance
- interoperability
- multiple, reliable sources of supply
- supportable, sustainable systems

But the DoD cannot afford to pay for it.

How is the DoD going to handle this situation?

DoD wants to take advantage of

- the marketplace
- the commonality within its reach
- potential efficiencies

From “Open Systems for Executives” Executive Workshop by Dr. Carol Sledge, Tricia Oberndorf, Software Engineering Institute, 2009

Open Systems Is An Answer

Standards and conformance management make the marketplace work for the DoD

A systems vision and common architecture perspective helps the DoD exploit commonalities and provides the guide for evolution.

A business strategy provides the framework for achieving efficiencies.

Open systems—
it makes (business) sense.

From “Open Systems for Executives” Executive Workshop by Dr. Carol Sledge, Tricia Oberndorf, Software Engineering Institute, 2009

Open Systems Approach – NDAA for FY15, Section 801

2) OPEN SYSTEMS APPROACH.—The term ''open systems approach'' means, with respect to an information technology system, an integrated business and technical strategy that—

(A) employs a ***modular design*** *and uses widely supported and consensus-based standards for key interfaces;*

(B) is subjected to successful validation and verification tests to ***ensure key interfaces comply*** *with widely supported and consensus-based standards; and*

(C) uses a ***system architecture that allows*** *<u>components to be added, modified, replaced, removed, or supported</u> by* ***different vendors throughout the lifecycle*** *of the system <u>to afford opportunities for enhanced competition and innovation</u> while yielding—*

(i) significant cost and schedule savings; and

(ii) increased interoperability.

What is an open system?

open system

A collection of interacting components designed to satisfy stated needs with the interface specification of components

- fully defined
- available to the public
- maintained according to group consensus

in which the implementations of components are conformant to the specification.

"Open Systems: What's Old is New Again" Oberndorf/Sledge
http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=18718

What is Open Systems Architecture (OSA)?

Open Systems Architecture is a strategy integrating business and technical practices to yield severable modules which can be competed

Components can be…
Added / Replaced / Removed
by different vendors

Open Business Model

Technical Architecture

OSA is Good Architecture Engineering

Most defining aspects of being an open system architecture are based on sound system architecture engineering:

- Good modularity, i.e. components are major architecture-level subsystems exhibiting properties of single abstraction, high cohesion, low coupling, high encapsulation
- Well-defined components and interfaces
- Use of interface standards rather than proprietary interfaces

However the benefits of OSA, such as increased competition and avoidance of vendor lock, require multiple suppliers of separately-procurable components to use the same:

- Component boundaries
- Interfaces
- Interface standards

Open system architectures are important, but other things are important too:

- Business and engineering-tradeoffs must be made

No system architecture is completely open.

Challenges Driving OSA Development & Use

Technical Challenges:

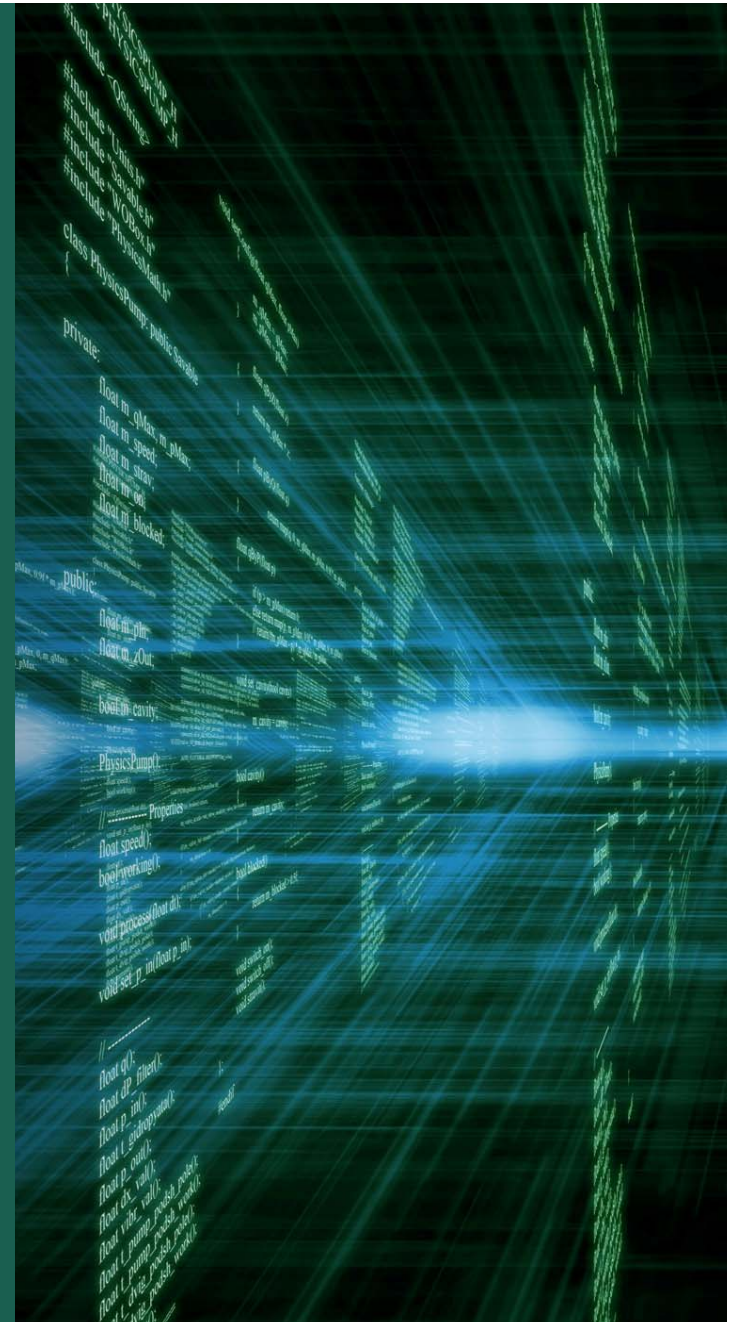
- Ad Hoc Architectures
- Legacy Code Base
- Proprietary Interfaces
- Low Interoperability, Maintainability, and Extensibility
- Isolation, removal, or replacement of US-only software for foreign military sales (FMS)

Acquisition Challenges:

- Vendor Lock
- Lack of Competition
- Lack of Data Rights
- Late Deliveries
- High Lifecycle Costs

Open Systems Architecture: Progress & Challenges

Panel Introduction & Discussion



Software Engineering Institute | Carnegie Mellon University

Panel Introduction

Moderators:

Forrest Shull *Carnegie Mellon/Software Engineering Institute (SEI)*

Harry Levinson *Carnegie Mellon/SEI*

Panel:

Thomas DuBois *The Boeing Company*

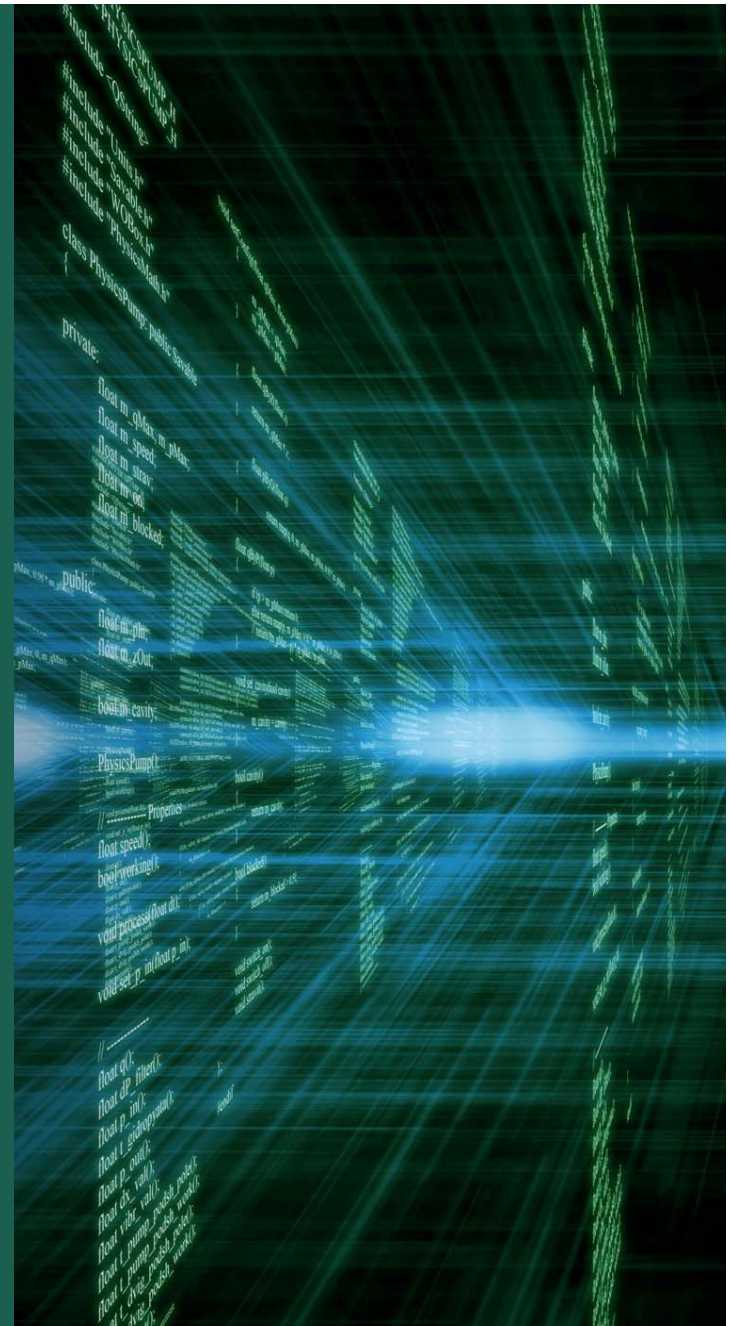
Michael Bandor *Carnegie Mellon/SEI*

Michael McLendon *Carnegie Mellon/SEI*

Doug Schmidt *Vanderbilt University & Carnegie Mellon/SEI*

Open Systems Architecture: Progress & Challenges

Panel Discussion



For Additional Information

Forrest J. Shull, Ph.D.

Harry Levinson

Software Engineering Institute

Carnegie Mellon University

Pittsburgh, PA 15213-3890

Voice: (703) 247-1372 - Shull

(412) 268-4148 - Levinson

FAX: 412 / 268-5758

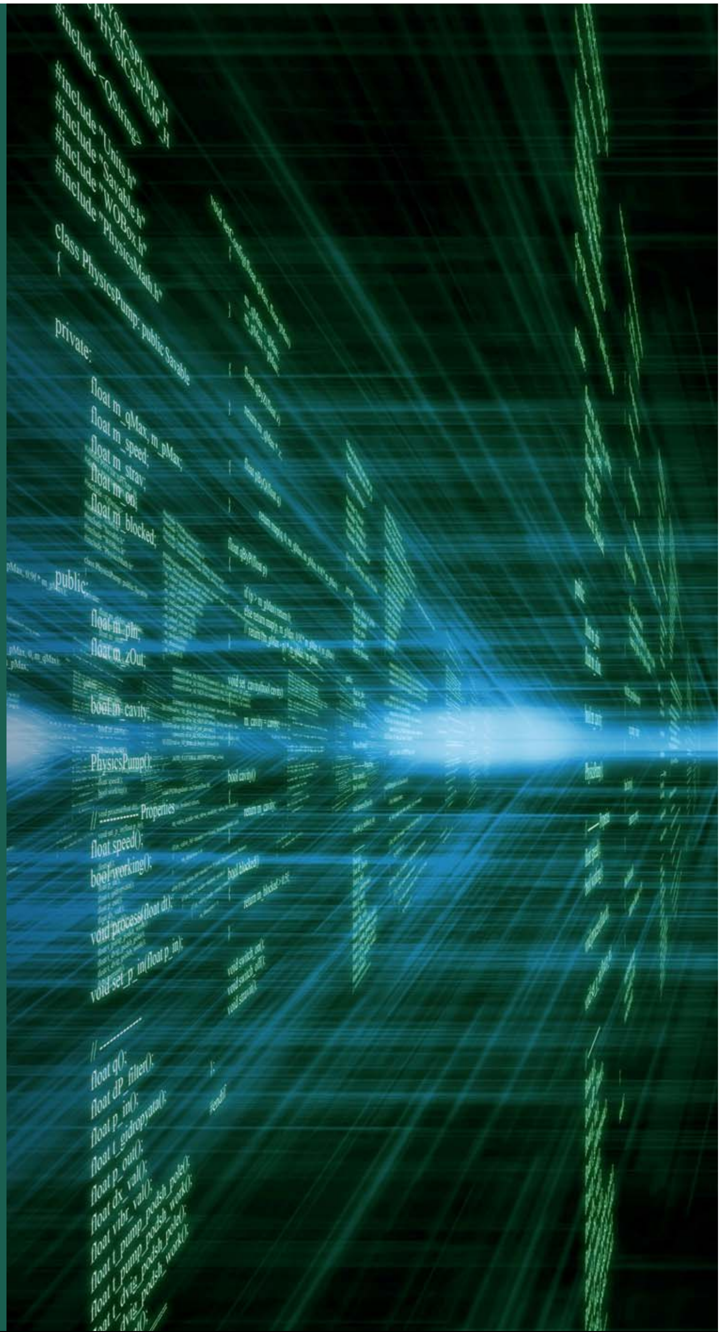
Email: fjshull@sei.cmu.edu

hll@sei.cmu.edu

www.sei.cmu.edu

Open Systems Architecture: Progress & Challenges

BACKUP SLIDES



Interface Standards

An *interface standard* is a widely-available document that specifies interfaces including services provided/required, protocols, message and data formats, etc.

- Standards may include ambiguities, implementation-dependent parts, and extensions that can result in incompatibilities and vendor lock.
- Use of implementation-dependent parts and extensions should be absolutely necessary, justified, encapsulated, and well-documented.

A interface *profile* is a set of one or more interface standards defining specific subsets (and potentially extensions) of these standards.

Compliance with the same interfaces or interface profiles promotes:

- ***Intra***operability is between two system-internal components
- ***Inter***operability is between a system-internal component and an external system

Any use of proprietary or vendor-specific profiles should be absolutely necessary, justified, hidden via encapsulation, and well-documented.

<u>M</u>odular Open Systems *Approach* – Modularity

An architecture is modular *to the degree to which* it consists of architectural components with the following properties:

1. **Architecture-Level** – Components are architecture-level subsystems.
2. **Single Abstraction** – Each component models (abstracts) the important aspects of a single relevant thing or concept in the application domain (e.g., avionics, sensors, propulsion, and weapons), user interface (e.g., screens, graphs, and icons), or technological domain (e.g., computers, networks, middleware, and OS).
3. **High Cohesion** – All of the parts of each component are necessary to implement the component's abstraction and the component does not contain any parts that are unrelated to its abstraction.
4. **Low Coupling** – The interfaces between these components are minimized in number, scope, and complexity (e.g., number and type of parameters).
5. **High Encapsulation** – The components are treated as black boxes that hide the implementations of their functionality behind well-documented visible interfaces that cannot be bypassed.