Towards Technical Reference Frameworks to Support Open System Architecture Initiatives

Douglas C. Schmidt
d.schmidt@vanderbilt.edu

Professor of
Computer Science

Institute for
Software Integrated
Systems

Vanderbilt University

Principal Researcher

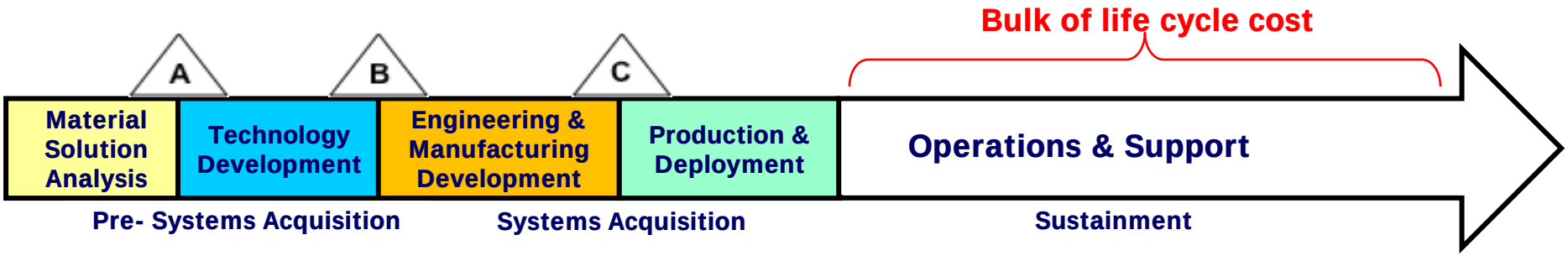
Software Engineering Institute

Carnegie Mellon University

The System of Systems Engineering Collaborators Information Exchange
December 15th, 2015

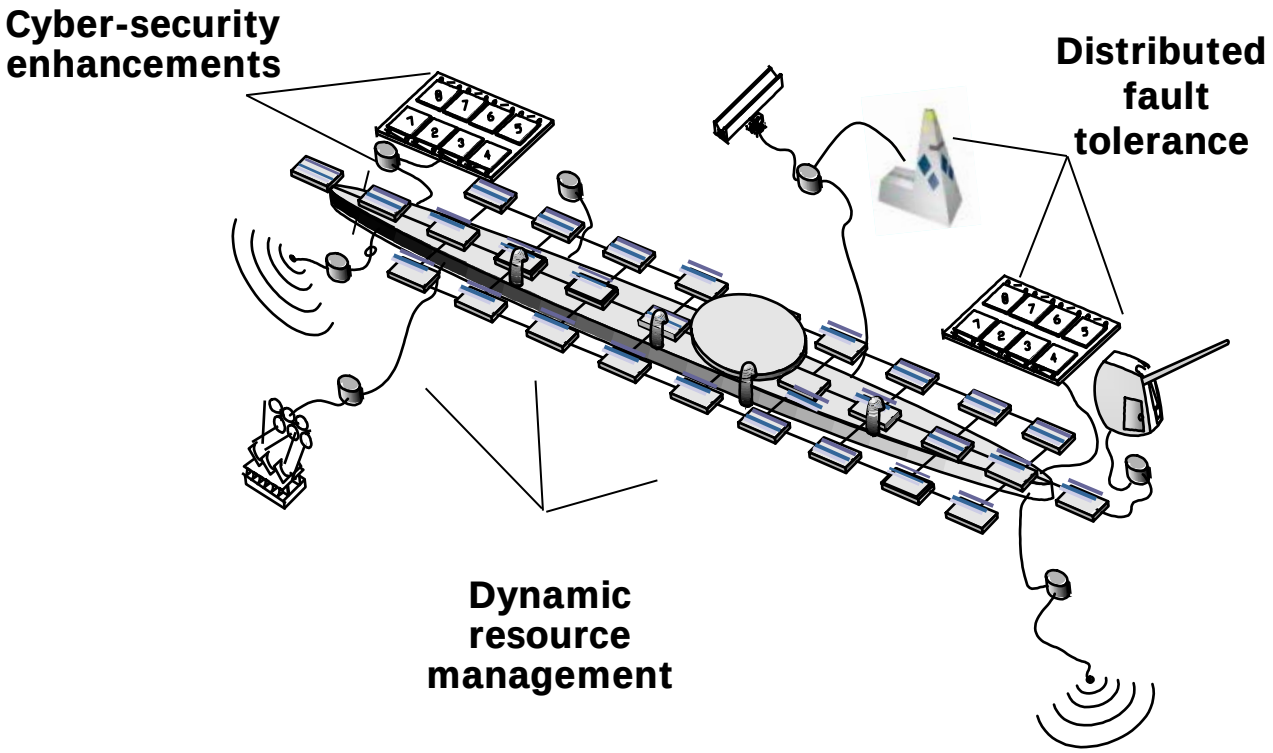
DoD Strategic Acquisition Goals

- Deliver *enhanced* integrated warfighting capability at *lower cost* across the enterprise & over the lifecycle



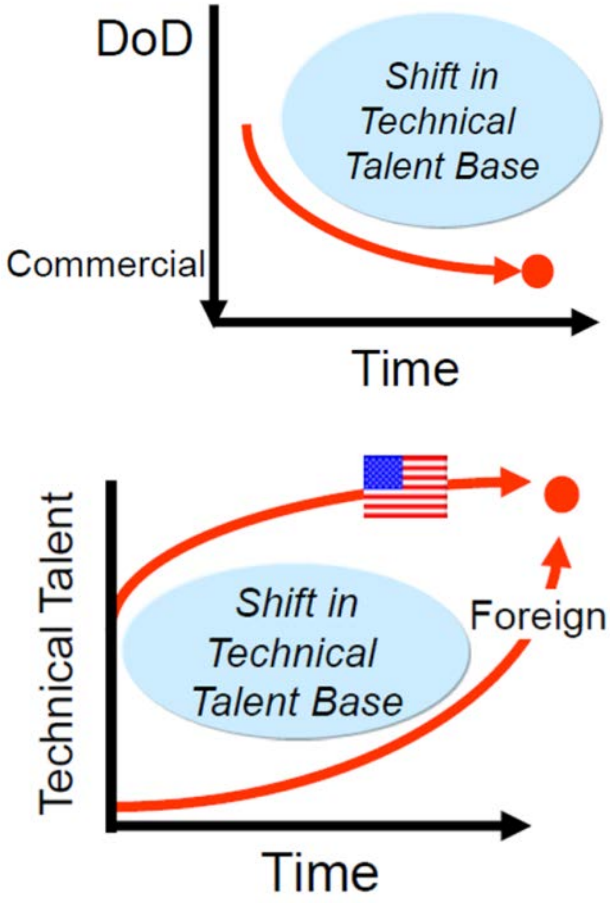
DoD Strategic Acquisition Goals

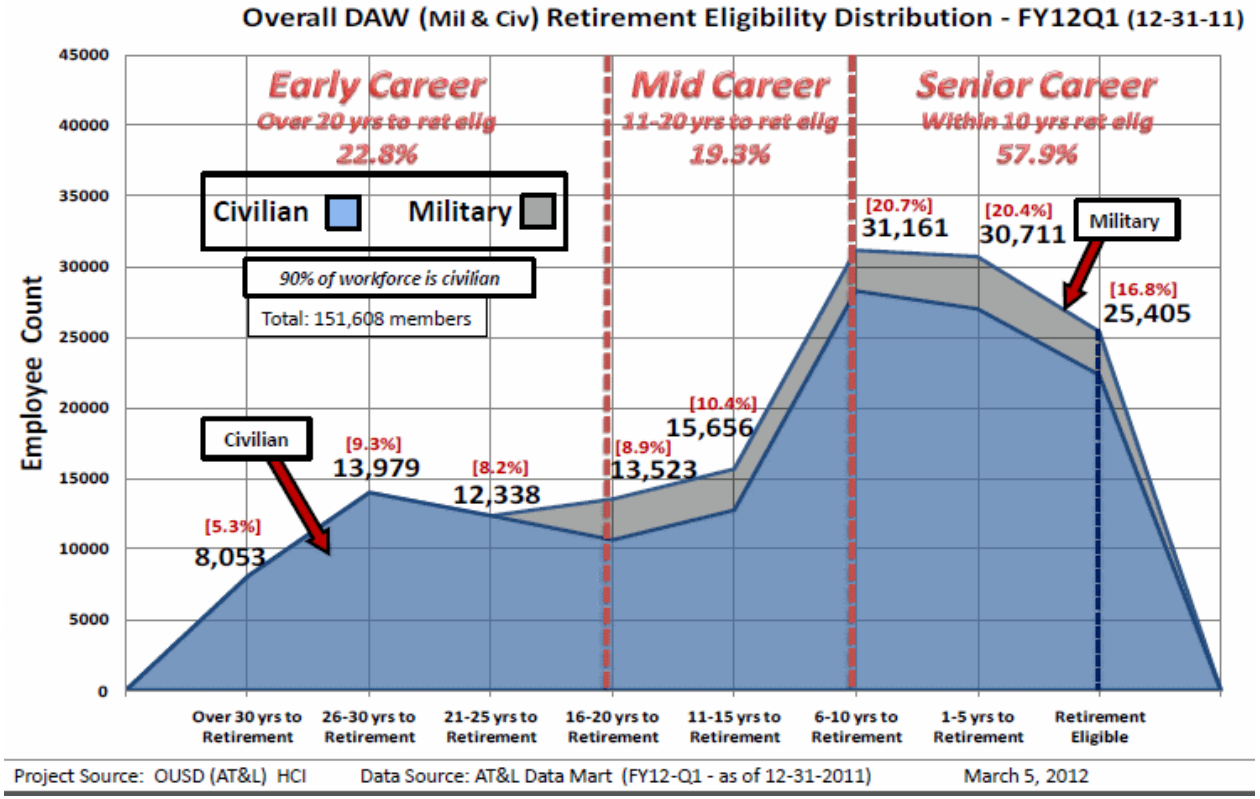
- *Reduce* cycle time of initial acquisition & new technology insertion



DoD Strategic Acquisition Goals

- Establish *sustainable* business & workforce strategies to support the other DoD acquisition goals

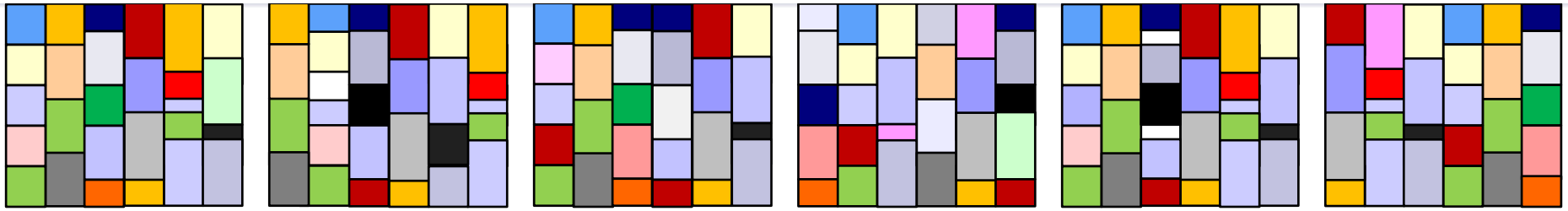




DoD Strategic Acquisition Goals

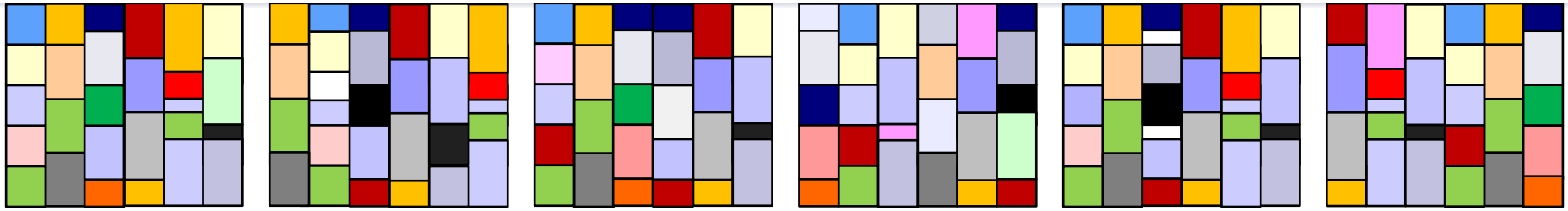
Alleviating the complexities & costs of software throughout the lifecycle is crucial to meeting DoD strategic acquisition goals

A Sampling of DoD Software Challenges



DoD cannot achieve its strategic acquisition goals when it must support too many software development activities, each implementing a unique solution

A Sampling of DoD Software Challenges



Drawbacks with stove-pipes

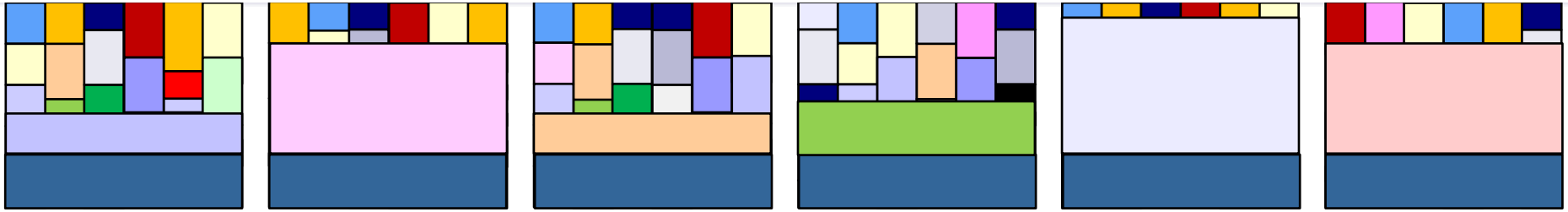
- Proprietary & vendor-locked
- Redundant to develop ... sustain
- Brittle & vulnerable to exploits
- Non-scalable tactical performance

A Sampling of DoD Software Challenges

Drawbacks wit

- Proprietary & ve
- Redundant to d
- Brittle & vulnera
- Non-scalable ta

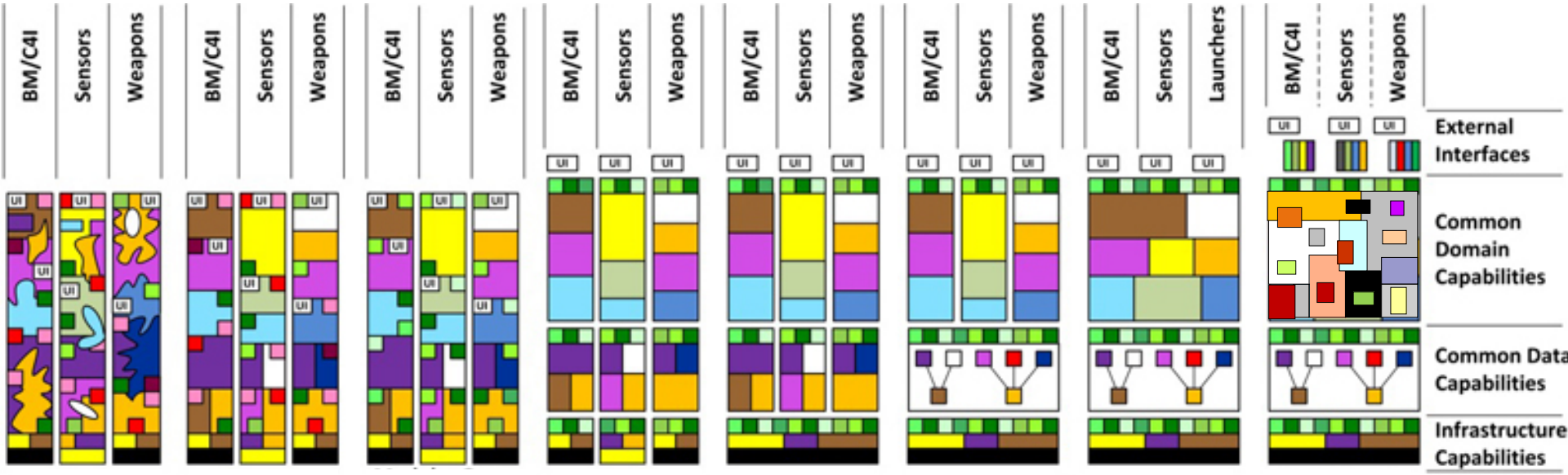
Solution: Open Systems Architecture (OSA)



Key tenets of OSA initiatives

- Published portable interfaces, protocols, & data formats
- Open standards
- Full design disclosure
- Modular, loosely coupled components
- An intentionally defined software/systems architecture

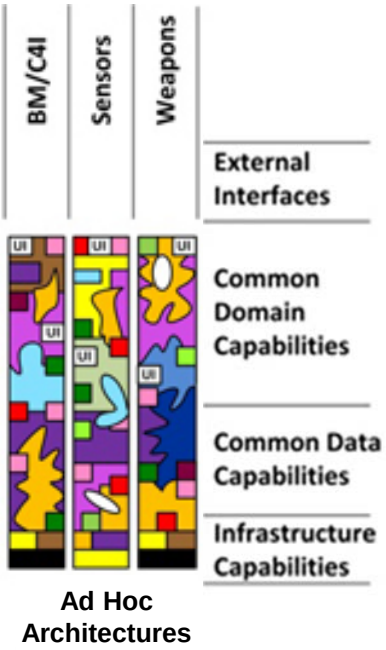
Evolution of DoD Combat Systems wrt the OSA Paradigm



In practice, production combat systems vary in terms of their progression along the continuum shown above

See blog.sei.cmu.edu/post.cfm/architectural-evolution-dod-combat-systems-359

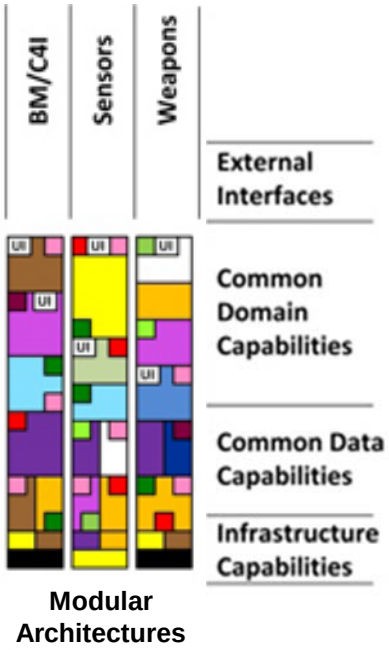
Evolution of DoD Combat Systems wrt the OSA Paradigm



Ad Hoc Architectures

Ad hoc architectures involve the separate development of each warfighter capability (such as BM/C4I, sensors, weapons, etc.) in a vertically stove-piped manner that lacks crisply-defined module boundaries

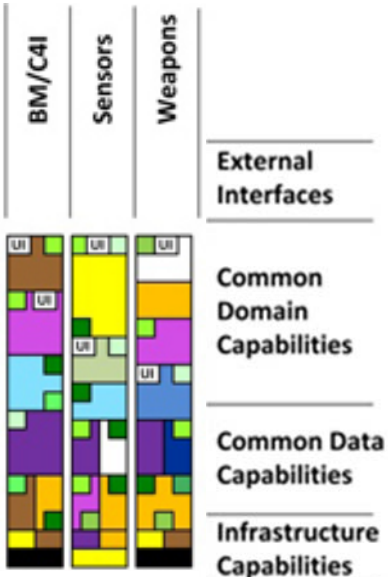
Evolution of DoD Combat Systems wrt the OSA Paradigm



Modular Architectures

Modular architectures define some crisp boundaries within their stove-pipes & transition away from top-down function-oriented decomposition to a more object-oriented & component-based decomposition

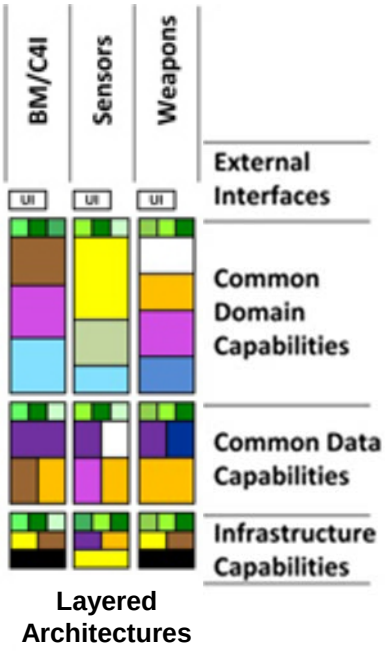
Evolution of DoD Combat Systems wrt the OSA Paradigm



Modular Open Systems Approach (MOSA) with Standard Key Interfaces

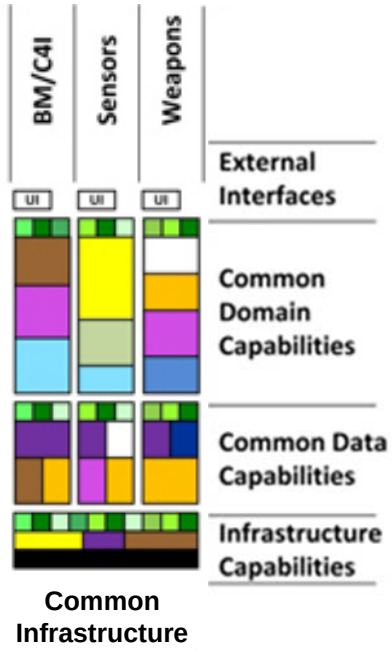
MOSA was the result of a well-defined, public standard approach with modular interfaces, designated key interfaces, & select open standards that allow programs a choice of vendors when a systems needs to be updated

Evolution of DoD Combat Systems wrt the OSA Paradigm



Layered architectures emerged as COTS began to mature & DoD programs began to purchase COTS directly from vendors & use them to layer systems so that they were no longer built entirely by integrators

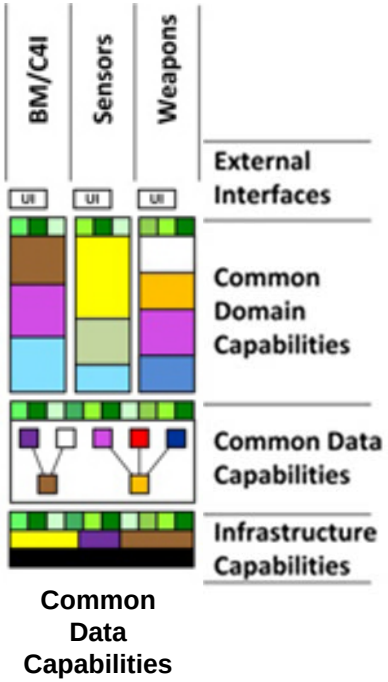
Evolution of DoD Combat Systems wrt the OSA Paradigm



Common infrastructure emerged due to the maturation of standards-based COTS middleware, operating systems, networks, & hardware

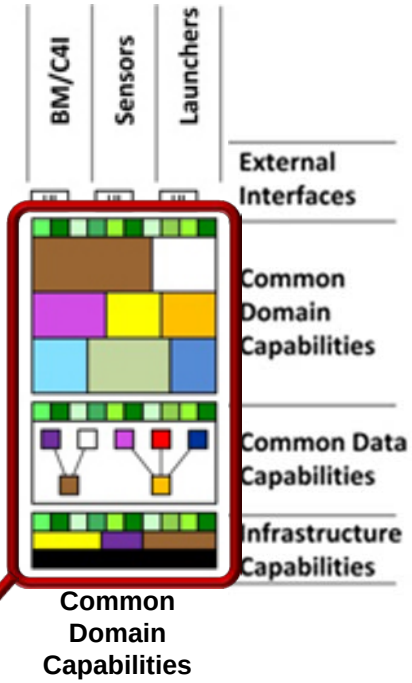
Evolution of DoD Combat Systems wrt the OSA Paradigm

Common data capabilities enable DoD programs to describe the information they have, the format of that information, the relationships, & dependencies among data types



Evolution of DoD Combat Systems wrt the OSA Paradigm

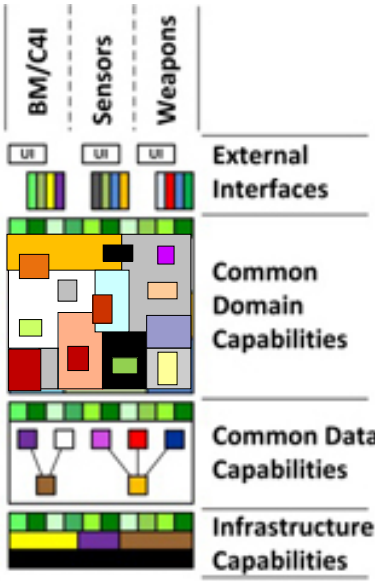
Common domain capabilities focus on the development of horizontally reusable services & components that address higher layers of the system stack



FACE™
Future Airborne Capability Environment

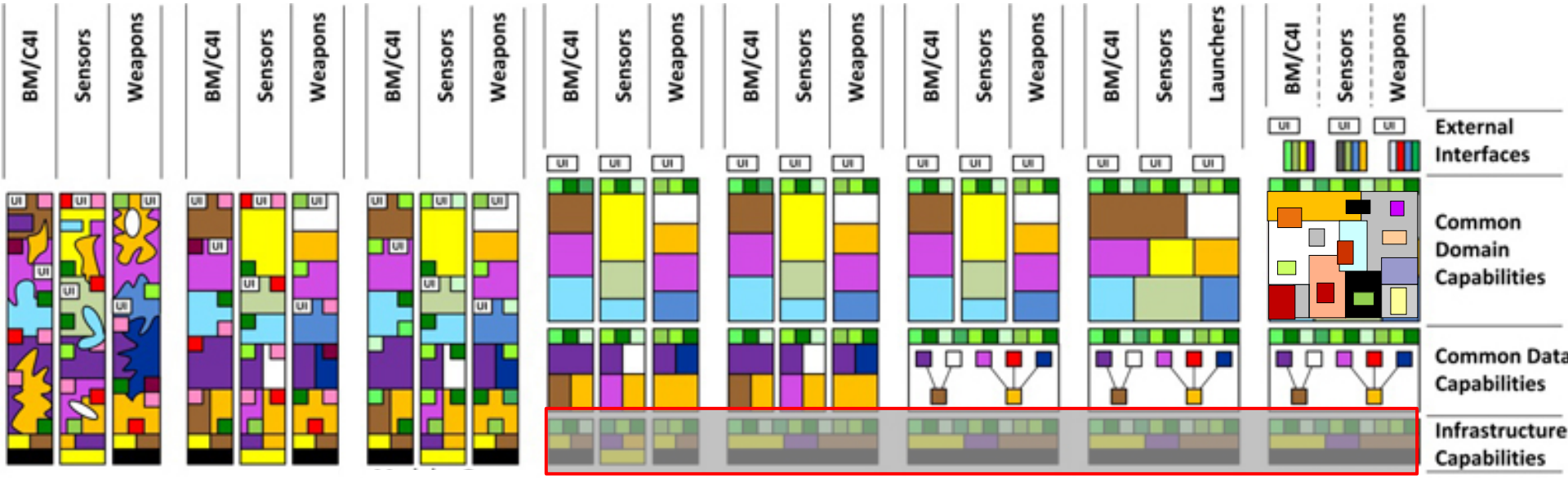
Evolution of DoD Combat Systems wrt the OSA Paradigm

Common domain capabilities via product-line architectures provides services that war fighters can reuse by building existing/new code atop common domain capabilities & allowing users to access/extend capabilities via systematic reuse



Common Domain Capabilities via Product-Lines

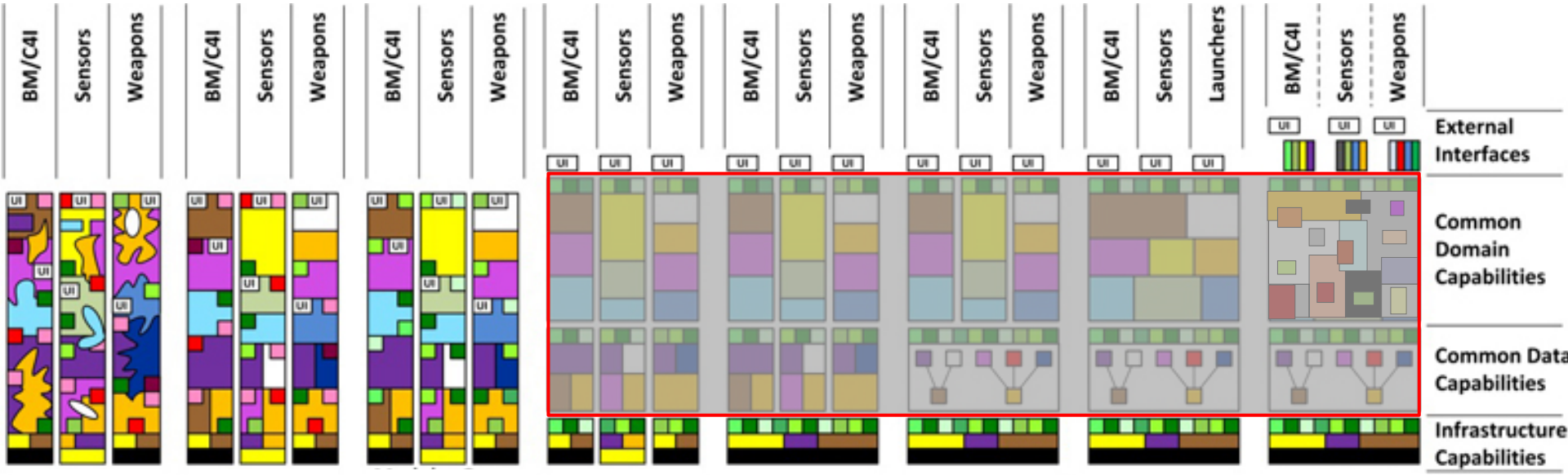
Evolution of DoD Combat Systems wrt the OSA Paradigm



Key Points

- OSA's been most successful at domain-independent *infrastructure* layer(s)
  - e.g., COTS products based on open standards like TCP/IP, POSIX, CORBA, DDS, etc.

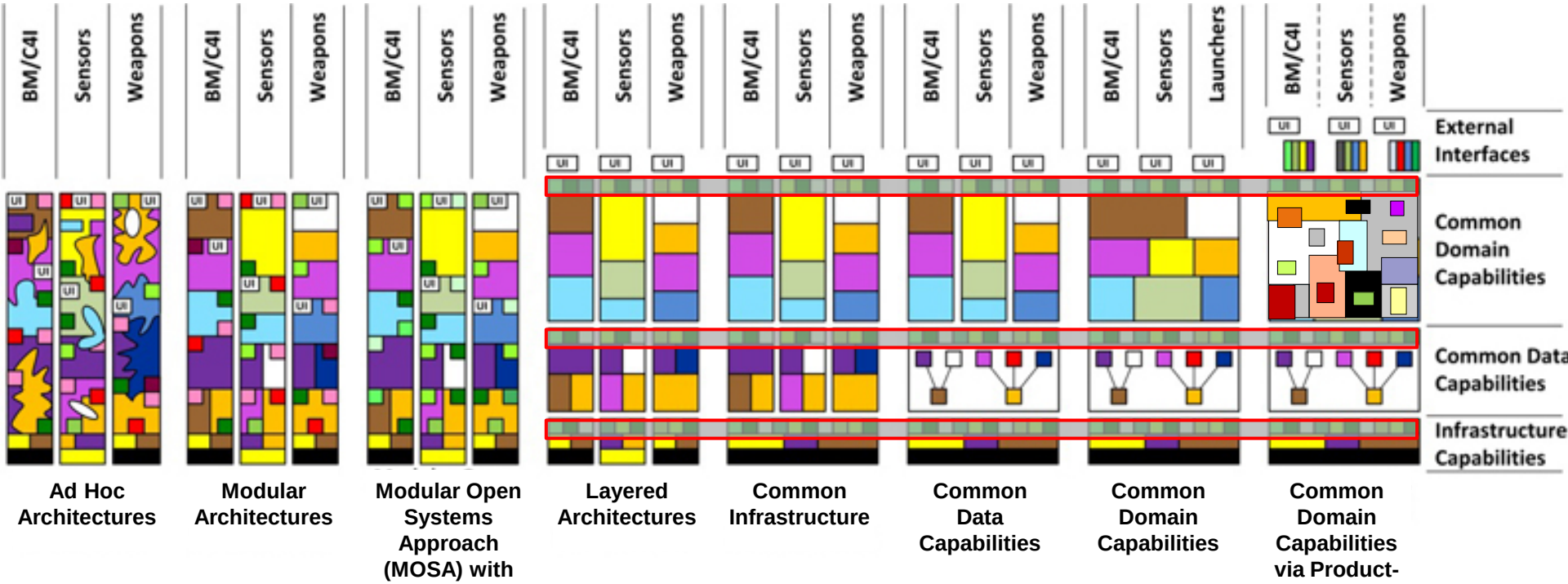
Evolution of DoD Combat Systems wrt the OSA Paradigm



Key Points

- OSA's been most successful at domain-independent *infrastructure* layer(s)
- Defining & adopting open standards for domain-specific layer(s) provide biggest payoff for OSA wrt reducing total ownership costs

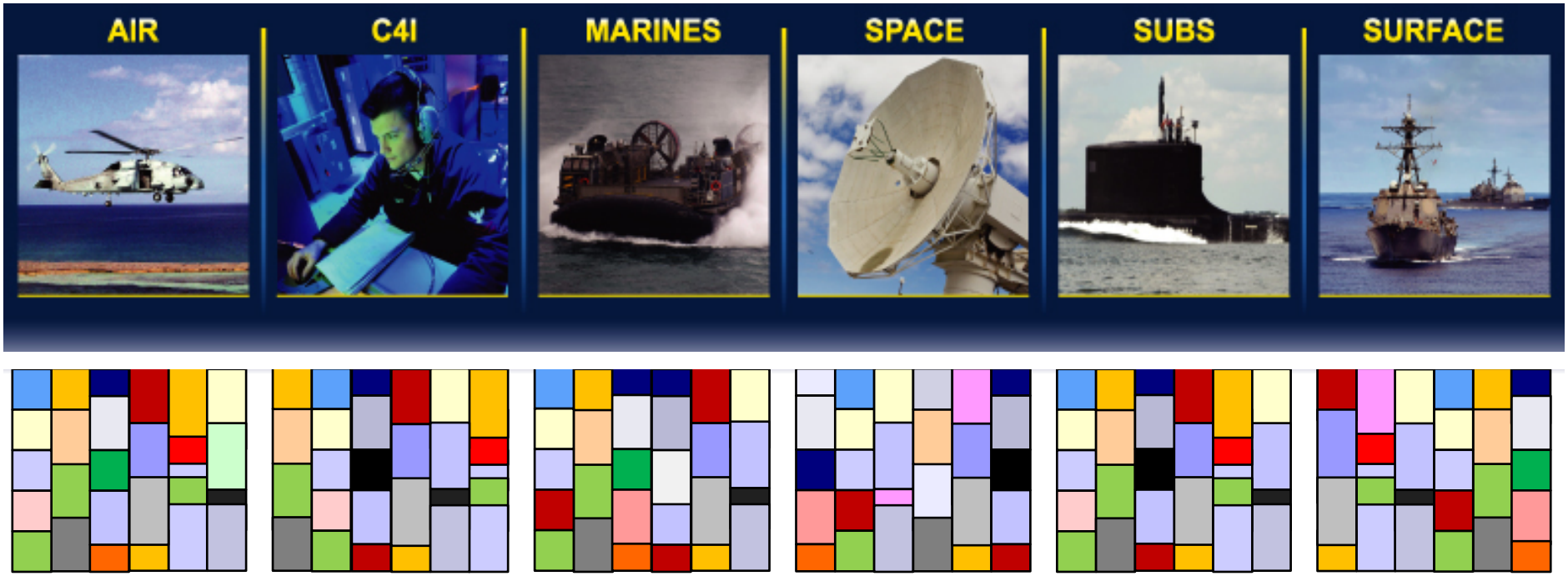
Evolution of DoD Combat Systems wrt the OSA Paradigm



Key Points

- OSA's been most successful at domain-independent *infrastructure* layer(s)
- Defining & adopting open standards for domain-specific layer(s) provide biggest payoff for OSA wrt reducing total ownership costs
- Some system components may never be realized via open standards & COTS
  - There's still significant value in publishing open domain-specific interfaces
    - e.g., help spur innovation, encourage competition, & avoid vendor-lock

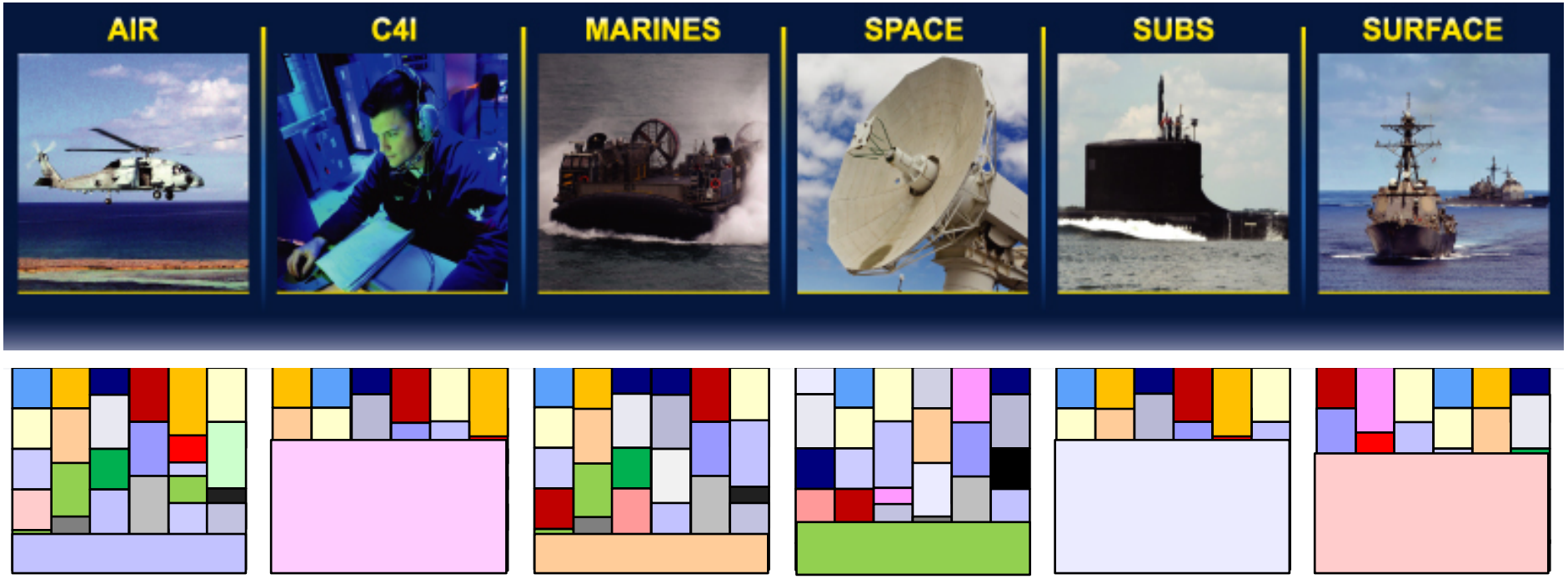
How Do We Get There From Here?



1. Divide programs of record into multiple *technical reference frameworks* that share common design & operational capabilities

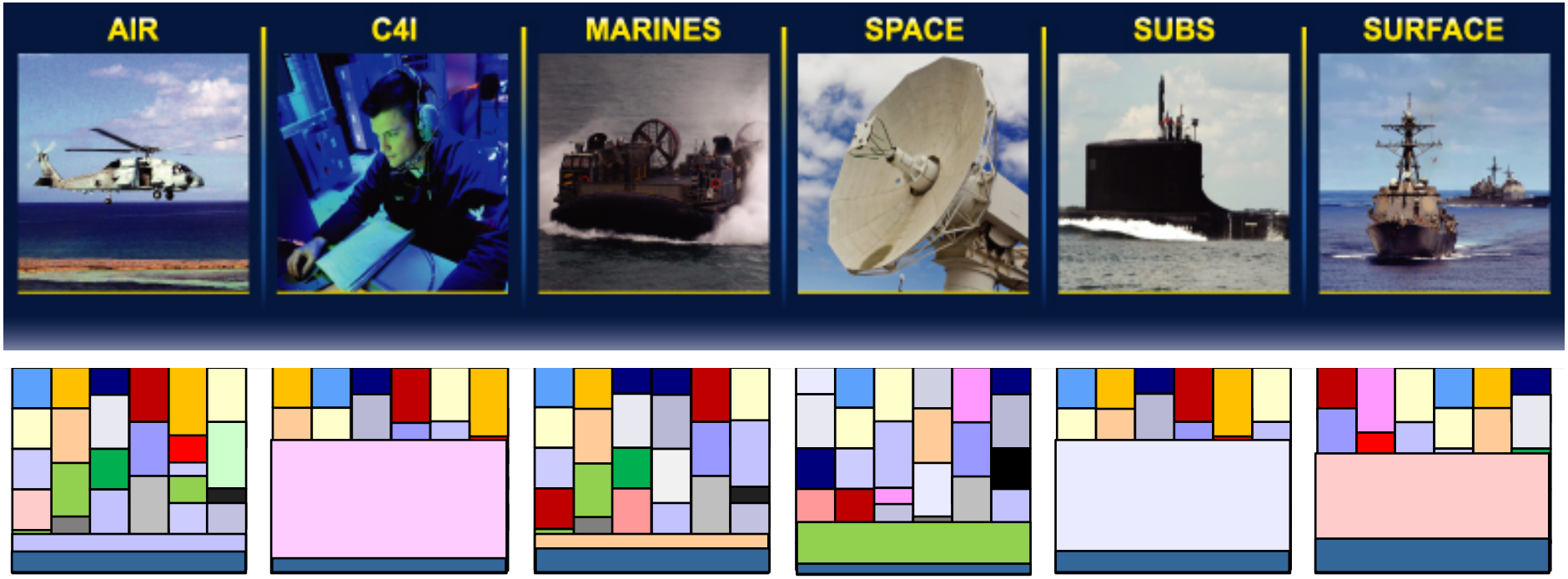
The Naval Open Systems Architecture Strategy (11/11/2012) identifies TRFs as "integrated sets of modular components that define common architectures for families of related warfighting systems to support improved competition & enable enterprise reuse"

How Do We Get There From Here?



1. Divide programs of record into multiple *technical reference frameworks* that share common design & operational constraints
2. Identify commonalities & incrementally evolve the technical reference frameworks

How Do We Get There From Here?



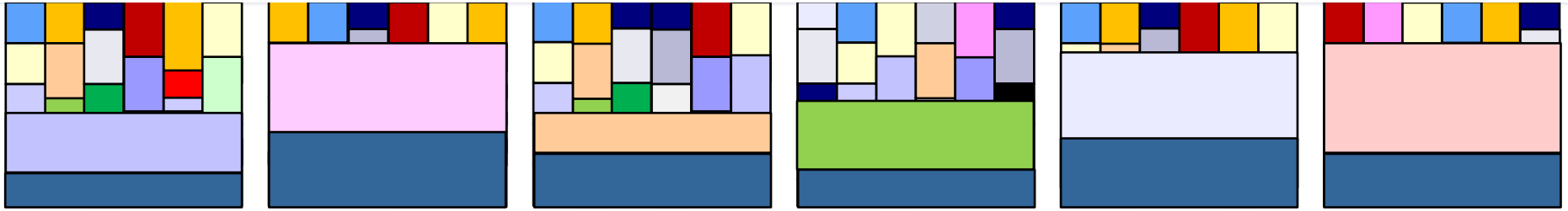
1. Divide programs of record into multiple *technical reference frameworks* that share common design & operational constraints
2. Identify commonalities & incrementally evolve the technical reference frameworks
3. Identify commonalities that span technical reference frameworks

Warning: amounts are only for illustrative purposes & shouldn't be construed as representative for specific domains

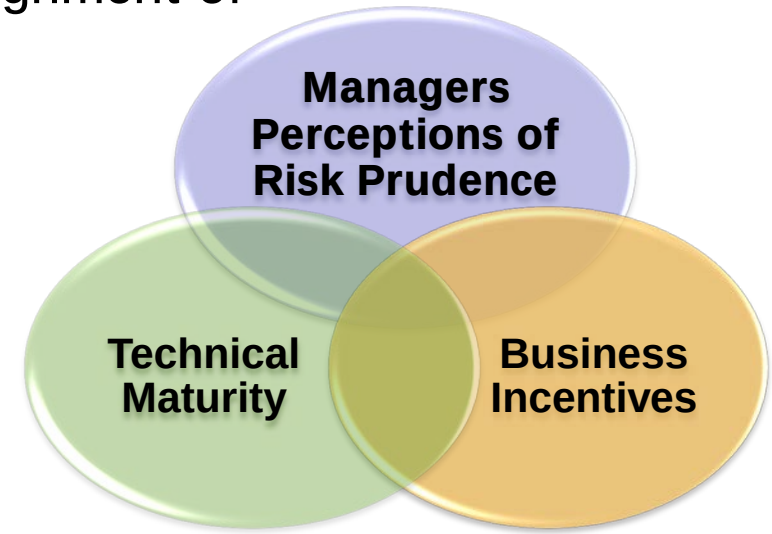
How Do We Get There From Here?

3. Identify commonalities that span technical reference frameworks
4. Expand commonality in both the technical reference frameworks & the broader OSA technical infrastructure
   - This is a stretch goal

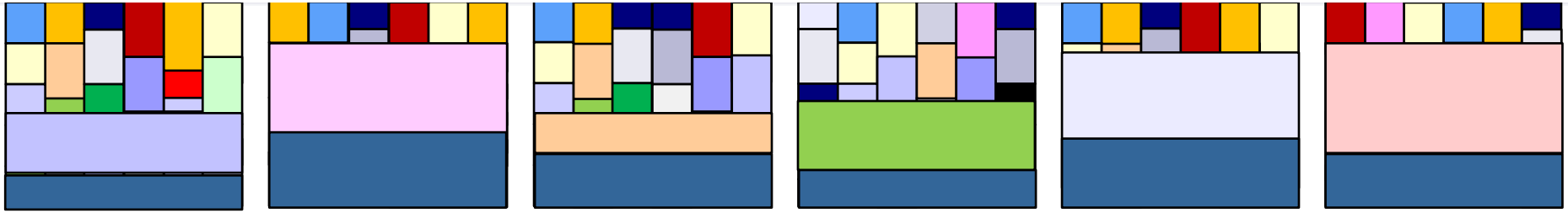
Some Examples of OSA Success Thus Far



OSA's more likely to succeed when there's alignment of



Some Examples of OSA Success Thus Far



Domain-Specific Services

Common Middleware Services

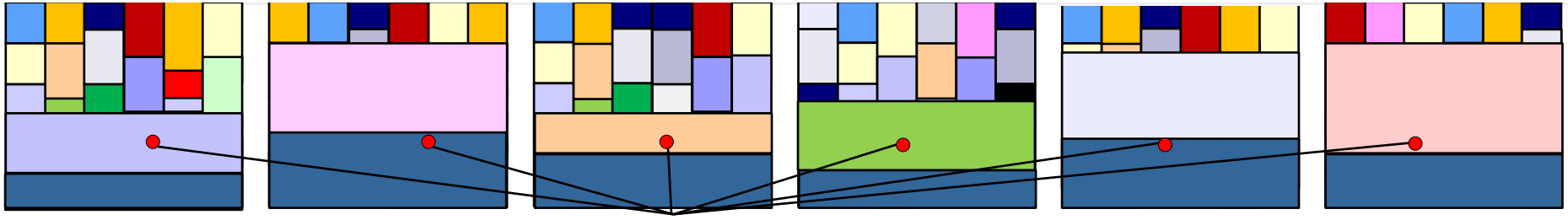
Distribution Middleware

Host Infrastructure Middleware

Operating Systems & Protocols

- The next 5 slides show examples of OSA successes in the domain-independent & domain-specific layers
- The examples are color coded as
  - **Dark green** – solid progress
  - **Light green** – some success, but more remains to be done
  - **Orange** – a work-in-progress, e.g., not widely fielded in programs of record (yet)

Some Examples of OSA Success Thus Far



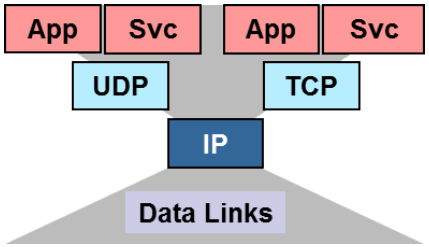
Domain-independent commonality

- Domain-Specific Services
- Common Middleware Services
- Distribution Middleware
- Host Infrastructure Middleware
- Operating Systems & Protocols

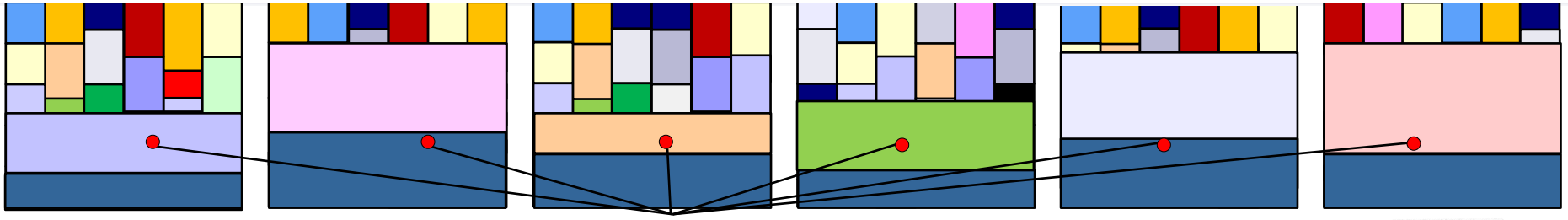
Provide mechanisms to manage endsystem resources, e.g., CPU scheduling, data storage, IPC, & memory management

IEEE POSIX®

Microsoft® Windows™



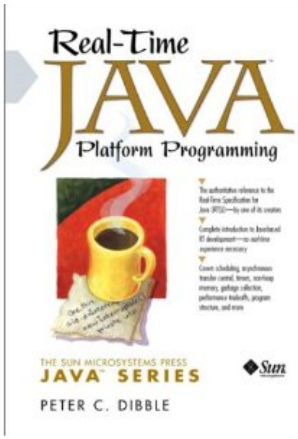
Some Examples of OSA Success Thus Far



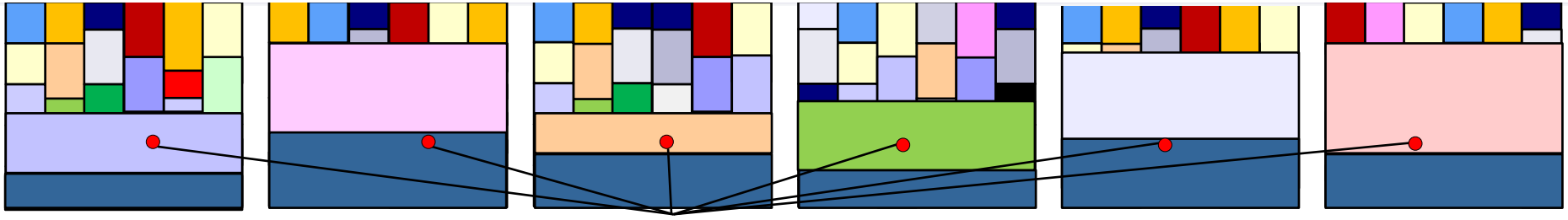
Domain-independent commonality

- Domain-Specific Services
- Common Middleware Services
- Distribution Middleware
- **Host Infrastructure Middleware**
- Operating Systems & Protocols

Encapsulates & enhances native OS mechanisms to create reusable network programming components



Some Examples of OSA Success Thus Far



Domain-independent commonality

- Domain-Specific Services
- Common Middleware Services
- **Distribution Middleware**
- Host Infrastructure Middleware
- Operating Systems & Protocols

Simplifies the programing of distributed components & automates/extends OS mechanisms end-to-end

DDS

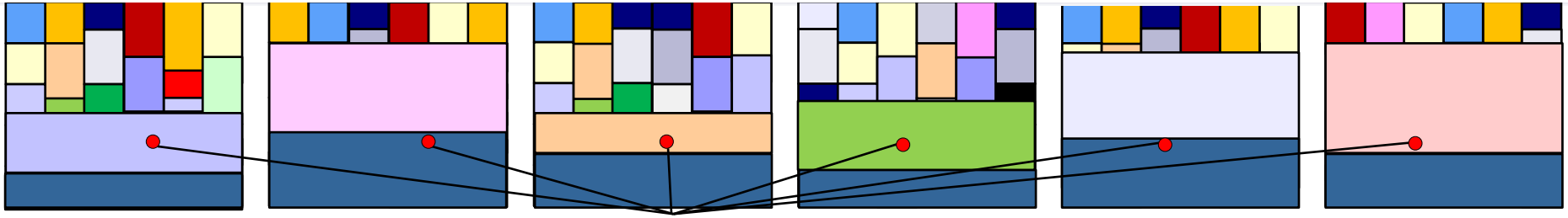
CORBA™

{JSON}

http://ws.apache.org

Apache <Web Services /> Project

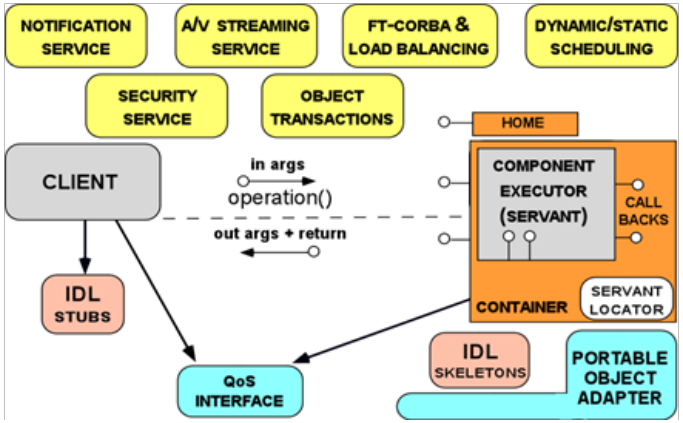
Some Examples of OSA Success Thus Far



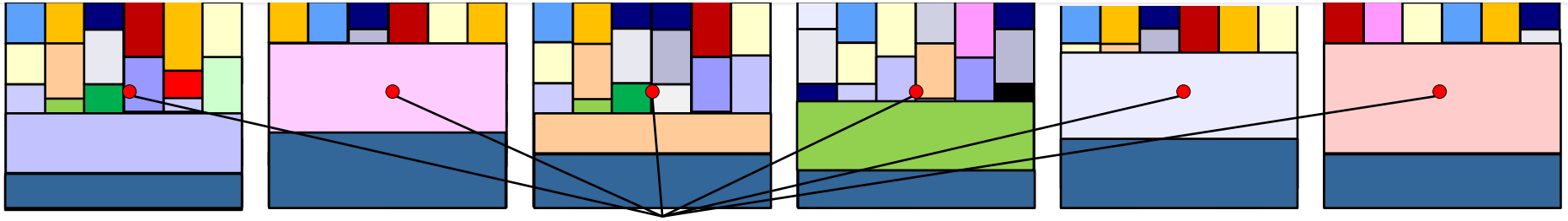
Domain-independent commonality

- **Domain-Specific Services**
- **Common Middleware Services**
- **Distribution Middleware**
- **Host Infrastructure Middleware**
- **Operating Systems & Protocols**

Defines reusable domain-independent services that simplify robust distributed computing



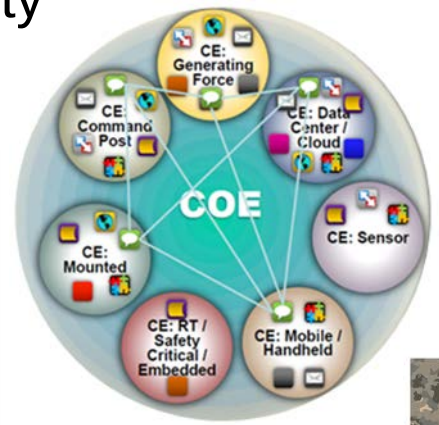
Some Examples of OSA Success Thus Far



- **Domain-Specific Services**
- **Common Middleware Services**
- **Distribution Middleware**
- **Host Infrastructure Middleware**
- **Operating Systems & Protocols**

Domain-specific commonality

Tailored to designated warfighter domains, e.g., C4ISR, avionics, air & missile defense, etc.

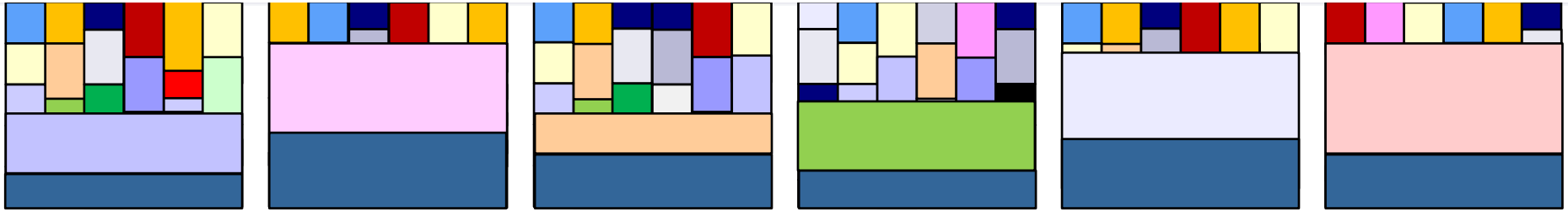


FACE™
Future Airborne Capability Environment

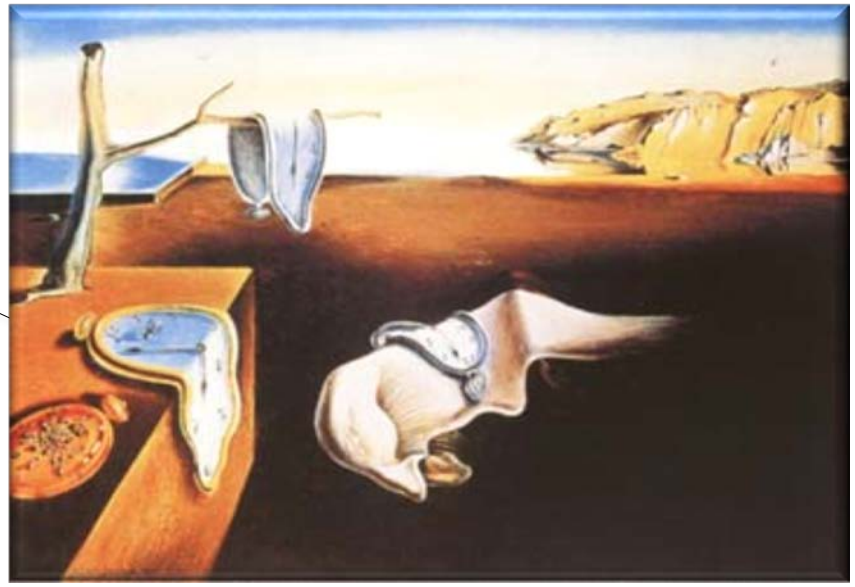
Some Impediments to Success of OSA Initiatives

Despite substantial technical advances during the past decade, affordable & dependable OSA-based solutions remain elusive

Some Impediments to Success of OSA Initiatives



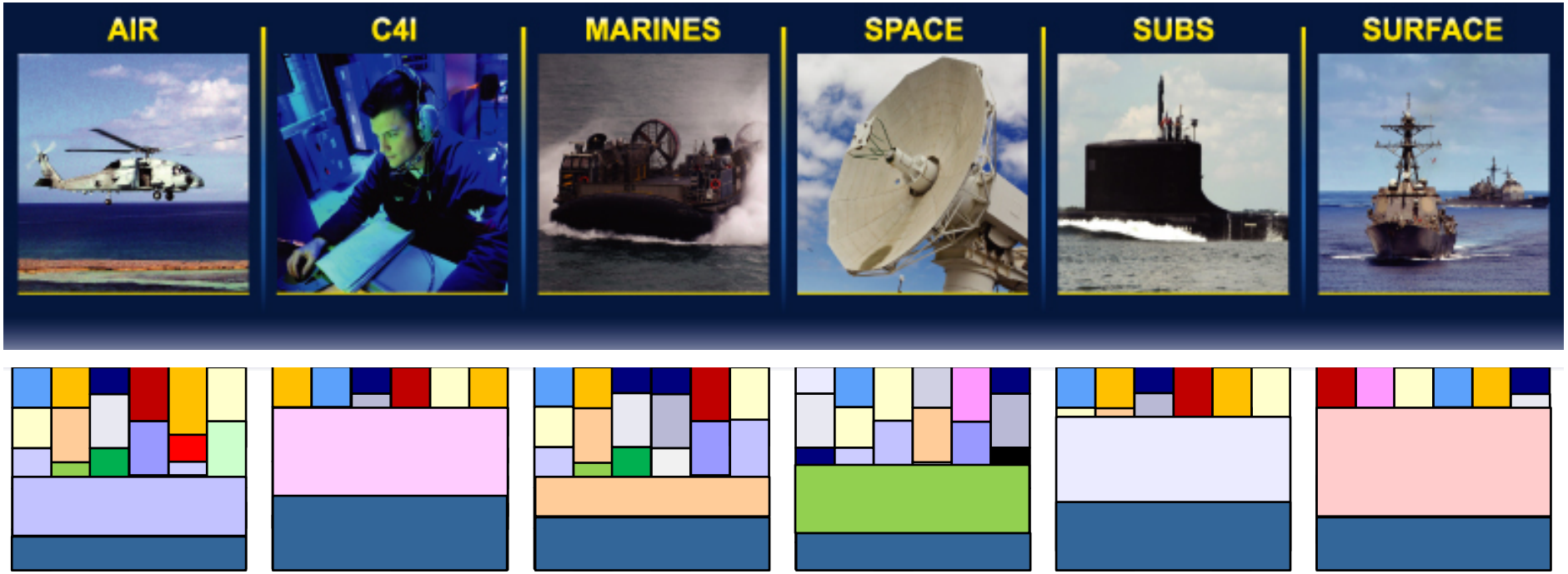
Glacially slow contracting processes impede timely delivery of capabilities that meet mission needs



Some Impediments to Success of OSA Initiatives

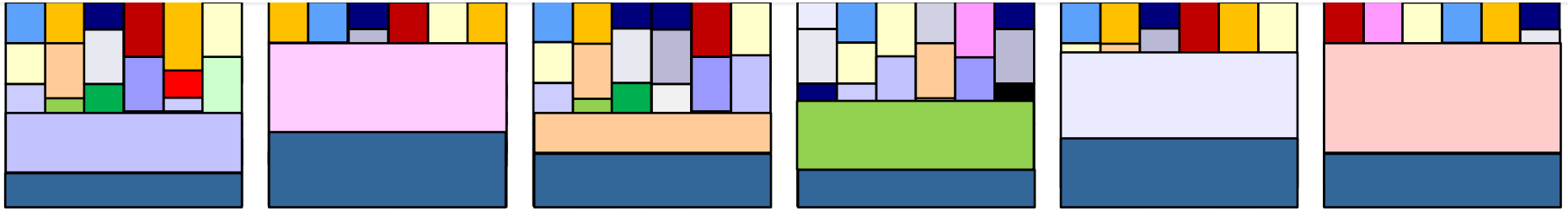
Contracting models that assume requirements can be fully defined up front are expensive when inevitable changes occur

Some Impediments to Success of OSA Initiatives



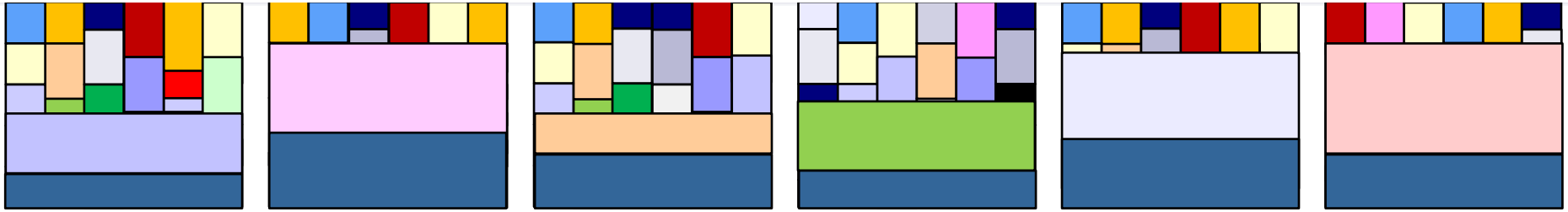
Quality-of-service (QoS) suffers when OSA initiatives use COTS standards & products that are ill-suited for mission-critical DoD combat systems

Some Impediments to Success of OSA Initiatives

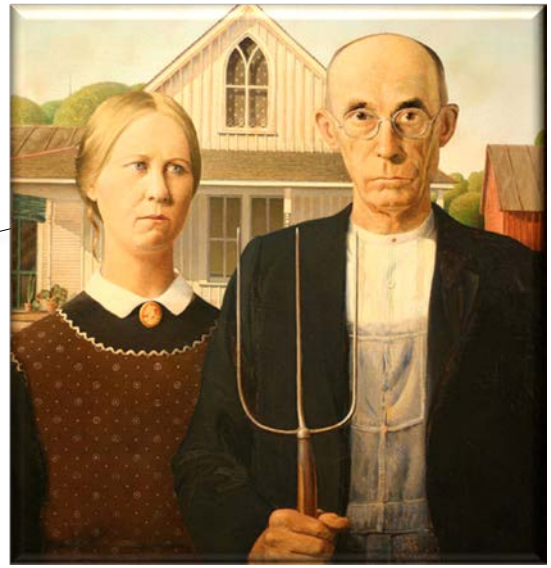


"Serialized phasing" of app & infrastructure development postpones identifying design flaws that degrade system QoS until late in lifecycle, i.e., during final system integration

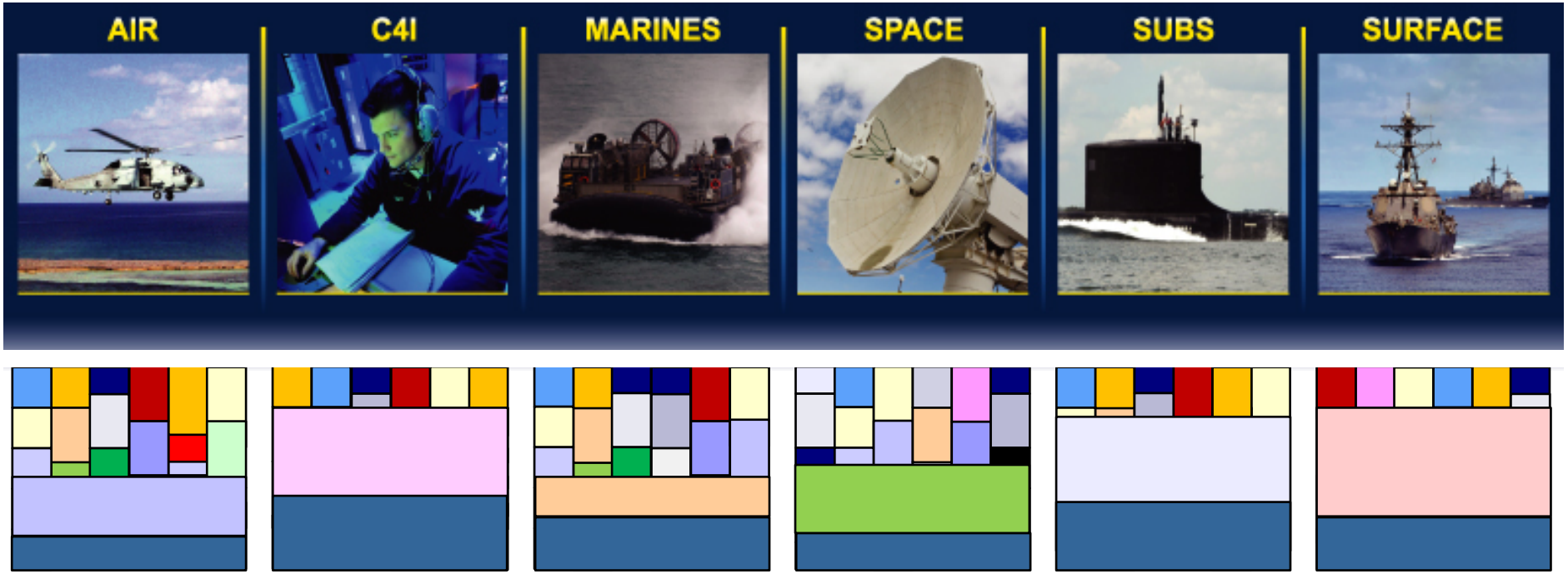
Some Impediments to Success of OSA Initiatives



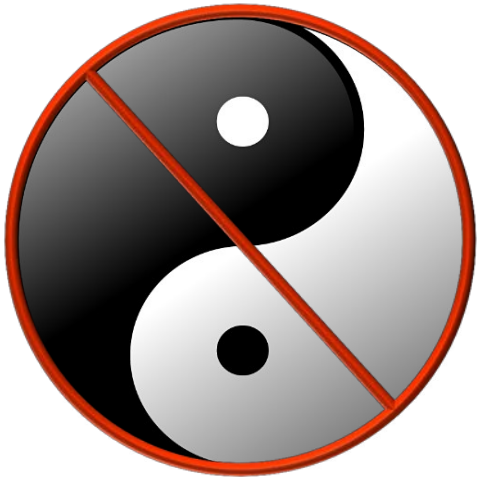
Rigid adherence to obsolete standards & ossified reference architectures limits application capabilities & impedes OSA technology refresh



Some Impediments to Success of OSA Initiatives

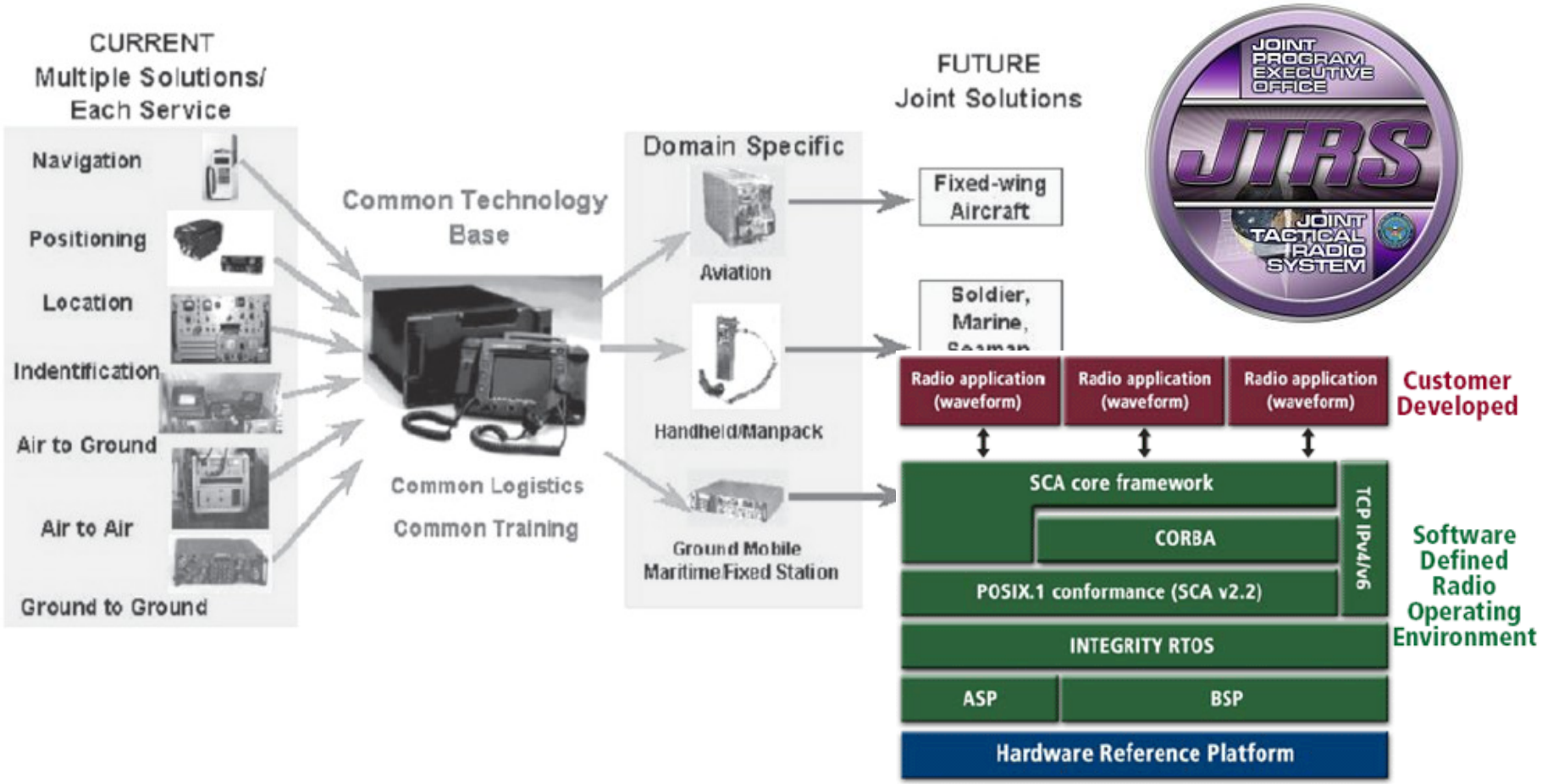


At the heart of these problems is the ***lack of an holistic approach*** that incentivizes competition in a targeted manner & aligns & balances key business, management, & technical drivers *at scale*



What Can We Learn from Our Failures?

Joint Tactical Radio System (JTRS) was a poster child for poor alignment between business, management, & technical drivers



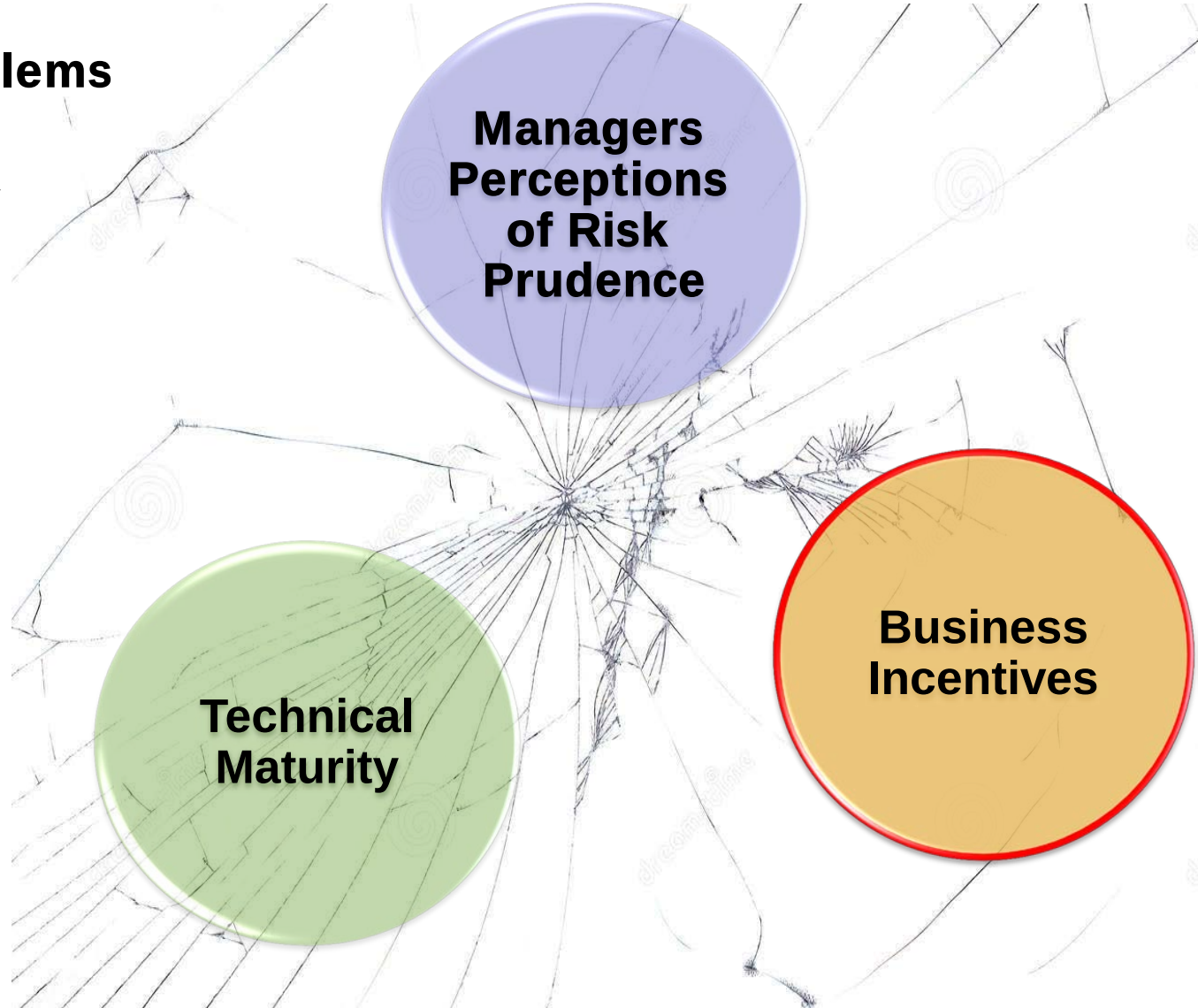
See blog.sei.cmu.edu/post.cfm/common-infrastructure-and-joint-programs-fourth-in-a-series

What Can We Learn from Our Failures?

Joint Tactical Radio System (JTRS) was a poster child for poor alignment between business, management, & technical drivers

Some key JTRS problems

- **Business** model disincentivized timely completion of design phase

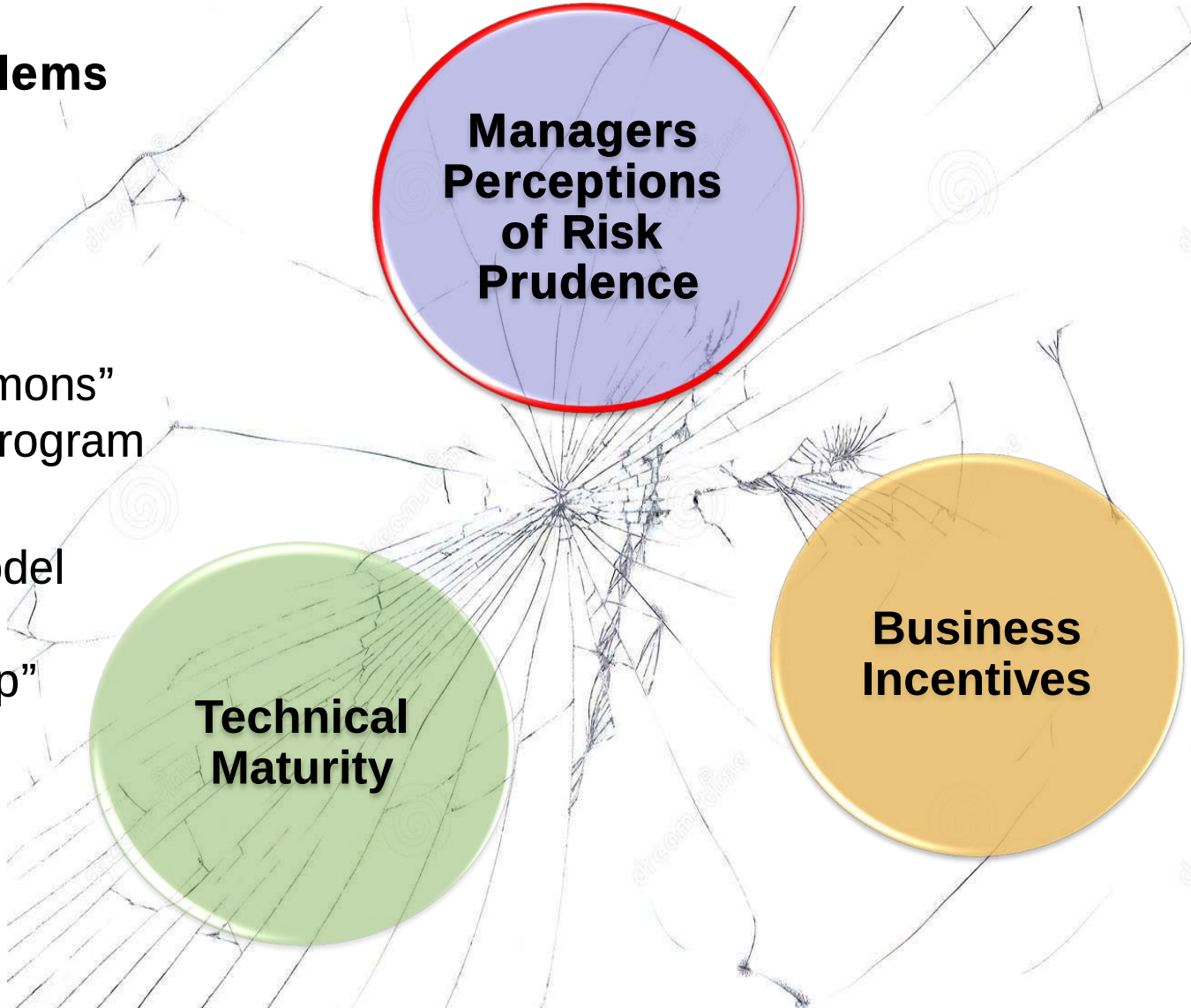


What Can We Learn from Our Failures?

Joint Tactical Radio System (JTRS) was a poster child for poor alignment between business, management, & technical drivers

Some key JTRS problems

- **Business** model disincentivized timely completion of design phase
- "Tragedy of the Commons" effects complicated program **management**
  - e.g., acquisition model fostered significant "requirements creep"

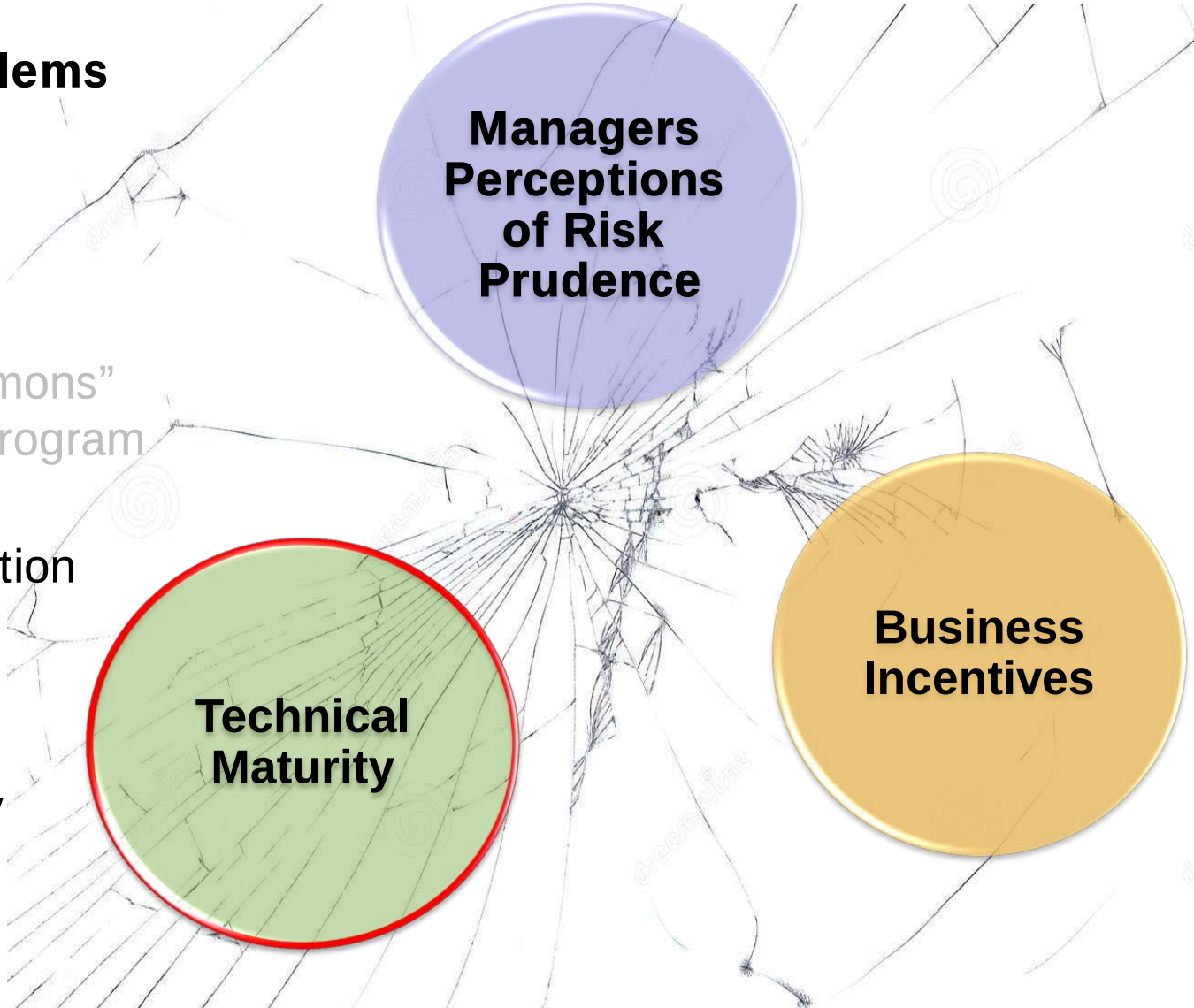


What Can We Learn from Our Failures?

Joint Tactical Radio System (JTRS) was a poster child for poor alignment between business, management, & technical drivers

Some key JTRS problems

- **Business** model disincentivized timely completion of design phase
- "Tragedy of the Commons" effects complicated program **management**
- Software Communication Architecture (SCA) **technical** standard was under-specified
  - Impeded portability & interoperability



How Can OSA Initiatives Be More Successful?

Key is Architecture-Led Iterative & Incremental Development (ALIID) approach

See blog.sei.cmu.edu/post.cfm/looking-ahead-the-sei-technical-strategic-plan

How Can OSA Initiatives Be More Successful?

Key is Architecture-Led Iterative & Incremental Development (ALIID) approach

Business Drivers

Achieving effective competition & broad acceptance of OSA economic aspects

How Can OSA Initiatives Be More Successful?

Key is Architecture-Led Iterative & Incremental Development (ALIID) approach

Management Drivers

Ensuring effective leadership & guidance of OSA initiatives to control risk

How Can OSA Initiatives Be More Successful?

Key is Architecture-Led Iterative & Incremental Development (ALIID) approach

Technical Drivers
Foundations of OSA development & sustainment

How Can OSA Initiatives Be More Successful?

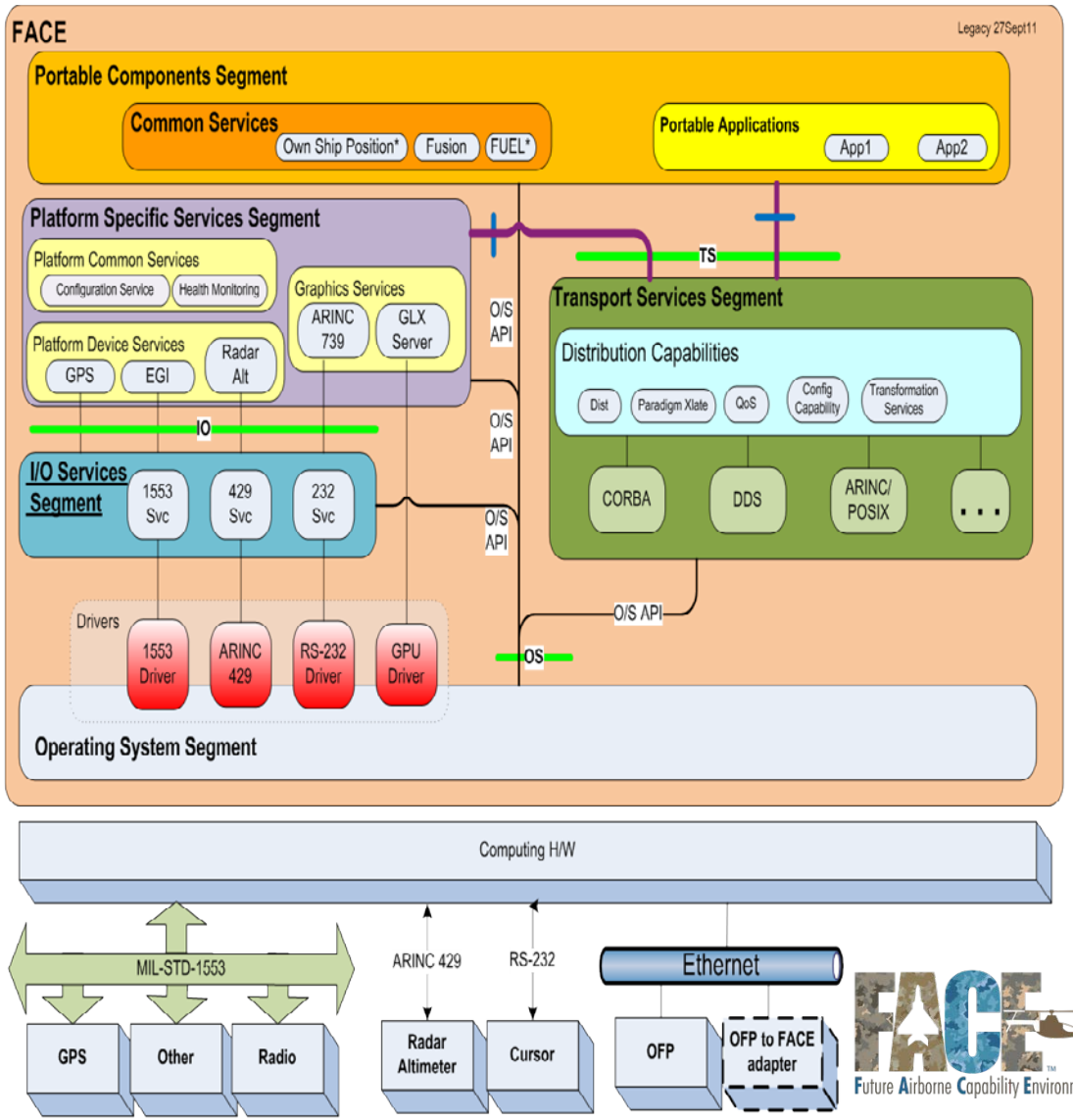
FACE is doing a good job at addressing these drivers

See blog.sei.cmu.edu/post.cfm/towards-common-operating-platform-environments-1

Competition Requires Economic & Value-based OSA

Key attributes

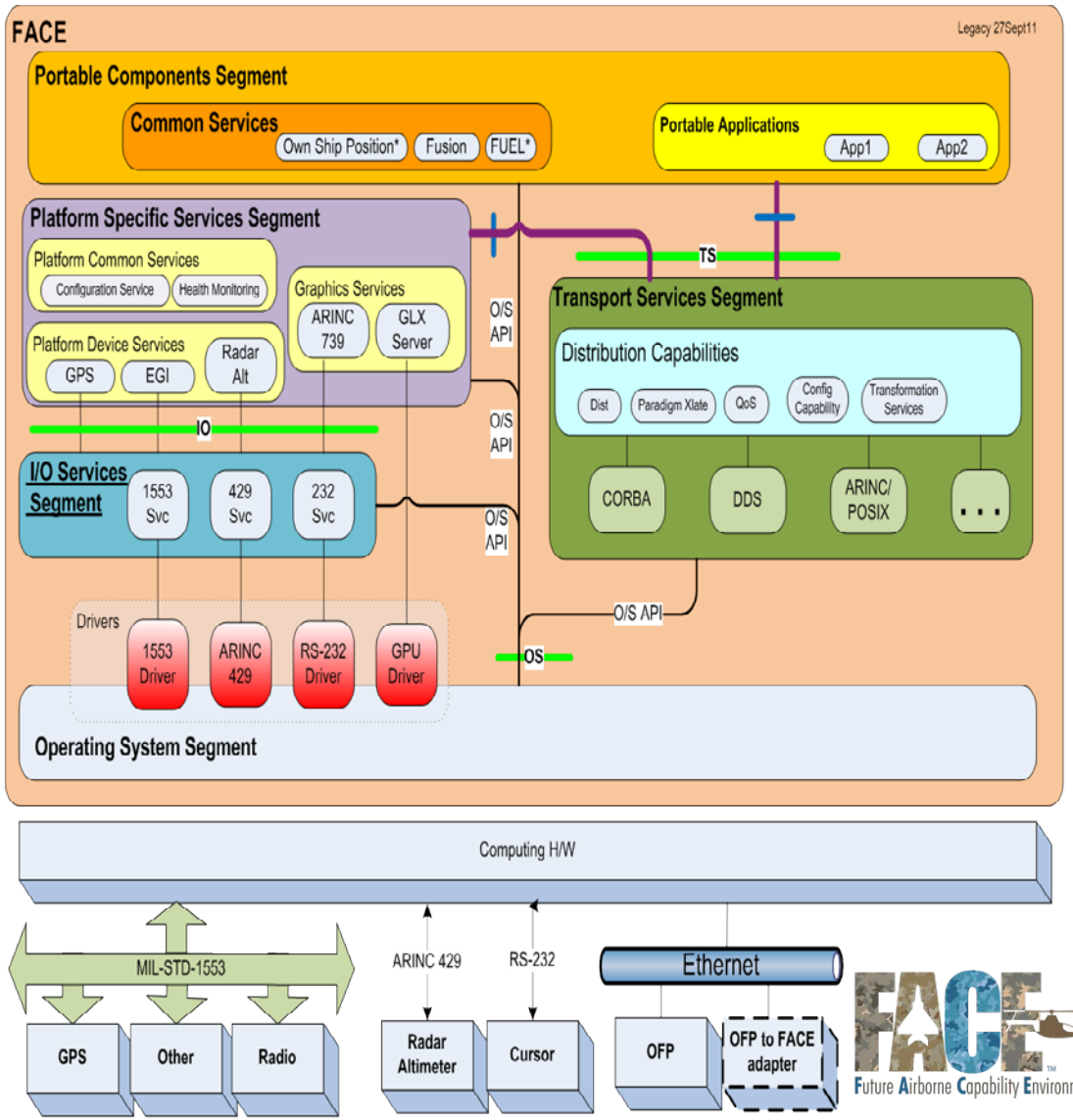
- *Crisply defined software & system technical architecture*
  - Technical reference frameworks enable competition at multiple system levels



Competition Requires Economic & Value-based OSA

Key attributes

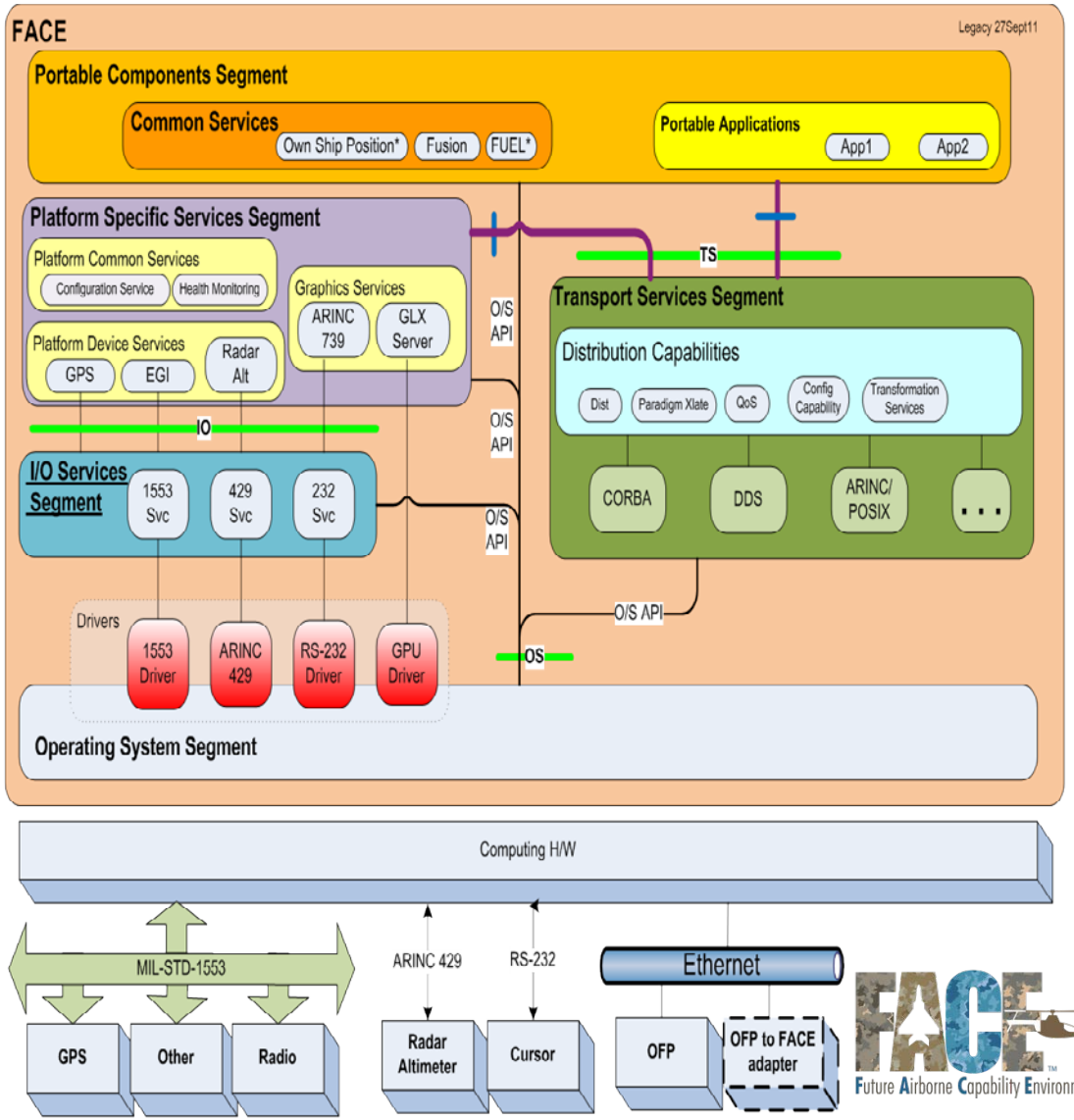
- *Crisply defined software & system technical architecture*
- *Modular innovation potential*
  - Economically-guided criterion for (de)composing technical reference frameworks into modules



Competition Requires Economic & Value-based OSA

Key attributes

- *Crisply defined software & system technical architecture*
- *Modular innovation potential*
- *Competitive evolutionary procurement processes*
  - Enable improvements throughout acquisition program lifecycles
  - Not just at infrequent down-selects



True competition requires robust ***interoperable*** *open system* ***architectures***

Concluding Remarks

"Big breakthroughs often happen when what is suddenly possible meets what is desperately necessary" – Thomas Friedman

Concluding Remarks

- OSA initiatives for DoD combat systems need a holistic vision & implementation strategy

Concluding Remarks

- OSA initiatives for DoD combat systems need a holistic vision & implementation strategy
- OSAs are achievable & valuable, though not easy to develop & sustain

Concluding Remarks

- OSA initiatives for DoD combat systems need a holistic vision & implementation strategy
- OSAs are achievable & valuable, though not easy to develop & sustain
- Alignment in business, technical, & management dimensions is essential for success

See blog.sei.cmu.edu/archives.cfm/category/common-operating-platform-environments-COPEs

Additional Information

SEI Blog

Software Engineering Institute | Carnegie Mellon

HOME AUTHORS

SEP 9 2013

Towards Affordable DoD Combat Systems in the Age of Sequestration

Common Operating Platform Environments (COPEs) Add comments

By Douglas C. Schmidt
Principal Researcher

Department of Defense (DoD) program managers and associated acquisition professionals are increasingly called upon to steward the development of complex, software-reliant combat systems. In today's environment of expanded threats and constrained resources (e.g., sequestration), their focus is on minimizing the cost and schedule of combat-system acquisition, while simultaneously ensuring interoperability and innovation. A promising approach for meeting these challenging goals is Open Systems Architecture (OSA), which combines (1) technical practices designed to reduce the cycle time needed to acquire new systems and insert new technology into legacy systems and (2) business models for creating a more

Search

Search

SEI Links

- Home Page
- Careers at the SEI
- SEI Technical Reports & Papers
- CERT Blogs
- SATURN Architecture Blog

Categories

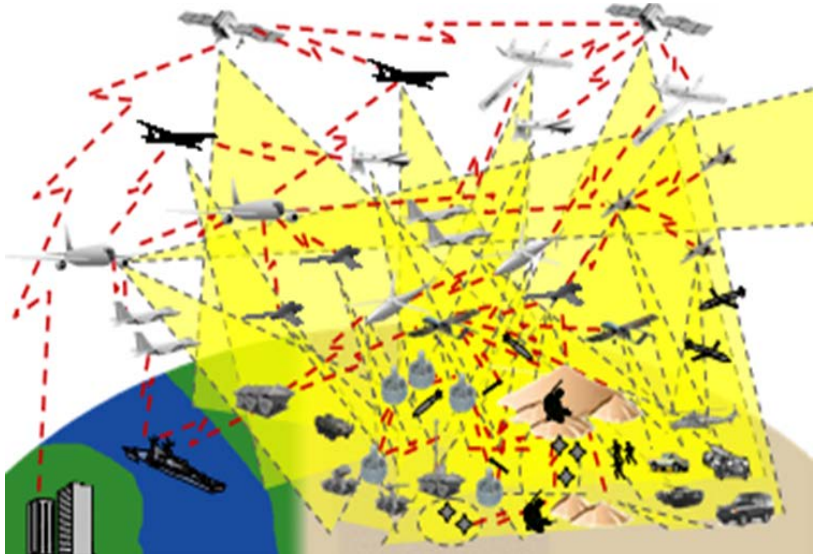
- Acquisition (30)
- Acquisition Dynamics (6)
- Agile (22)
- Architecture (11)

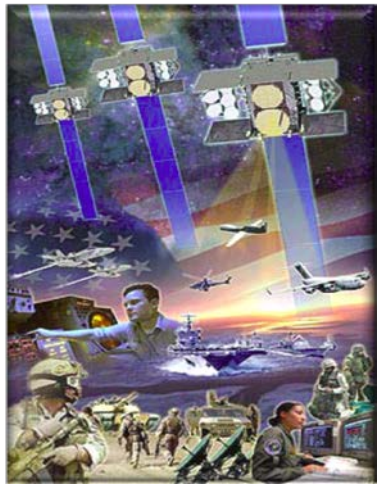
blog.sei.cmu.edu has more info on Open System Architectures

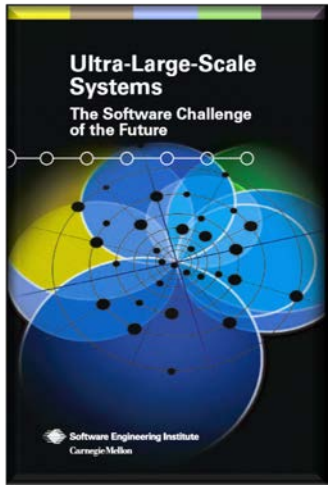
Additional Information

Ultra-large-scale (ULS) systems are socio-technical ecosystems comprised of software-reliant systems, people, policies, cultures, & economics that have unprecedented scale:

- # of software & hardware elements
- # of connections & interdependencies
- # of computational elements
- # of purposes & perception of purposes
- # of routine processes & “emergent behaviors”
- # of (overlapping) policy domains & enforceable mechanisms
- # of people involved in some way
- Amount of data stored, accessed, & manipulated
- ... etc ...





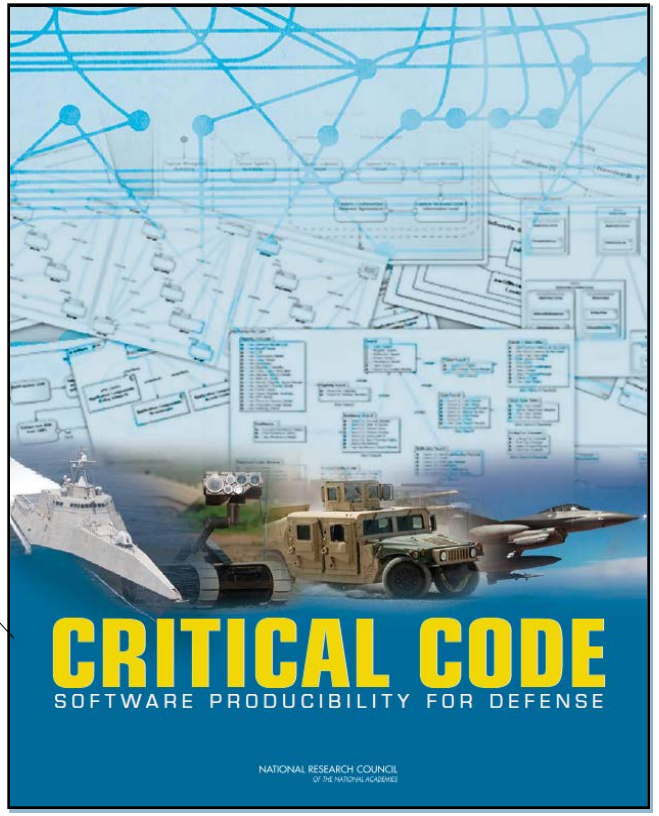


www.sei.cmu.edu/uls

Additional Information

NRC Report Critical Code: Software Producibility for Defense (2010)

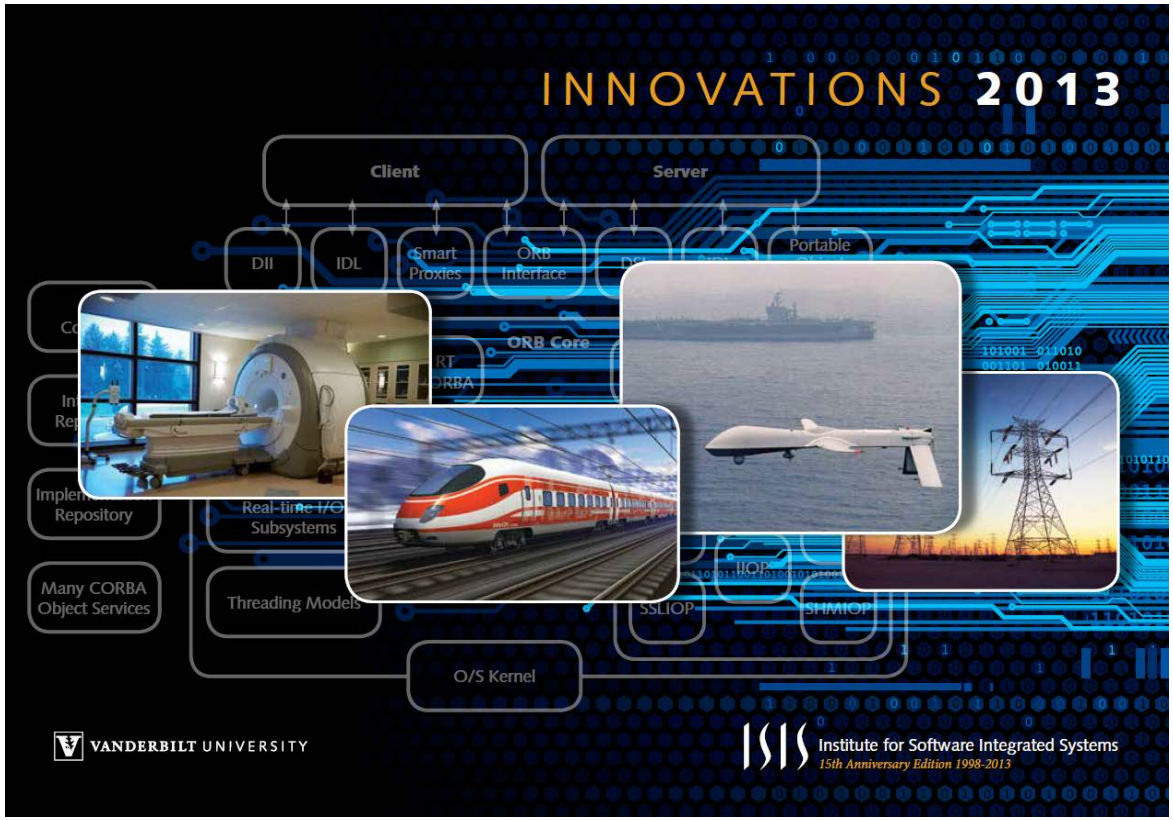
The report focuses on ensuring the DoD has the technical capacity & workforce to design, produce, assure, & evolve innovative software-reliant systems in a predictable manner, while effectively managing risk, cost, schedule, & complexity



Sponsored by Office of the Secretary of Defense (OSD) with assistance from the National Science Foundation (NSF), & Office of Naval Research (ONR),
http://www.nap.edu/openbook.php?record_id=12979&page=R1

Additional Information

- The Institute for Software Integrated Systems (ISIS) was established at Vanderbilt in 1998
- Research at ISIS focuses on systems with deeply integrated software that are networked, embedded, & cyber-physical
- Key research areas at ISIS:
  - Model-Integrated Computing
  - Middleware for distributed real-time & embedded (DRE) systems
  - Model-based engineering of cyber-physical systems
  - Wireless sensor networks
  - Systems security & privacy



engineering.vanderbilt.edu/innovations-2013 has more info on ISIS