How I stumbled across a Domain Overlay and why it's actually useful

22 March 2023

Richard Wise
Senior Research Engineer
richard.wise@gtri.gatech.edu

Michael Shearin
Senior Research Engineer
michael.shearin@gtri.gatech.edu

U.S.C. Title 10 §4401 MOSA Requirement

"A major defense acquisition program that receives Milestone A or Milestone B approval after January 1, 2019, shall be designed and developed, to the maximum extent practicable, with a ***modular open system approach*** *to enable incremental development and enhance competition, innovation, and interoperability."*
(*Office of Law Revision Council, United States Code*, n.d.)

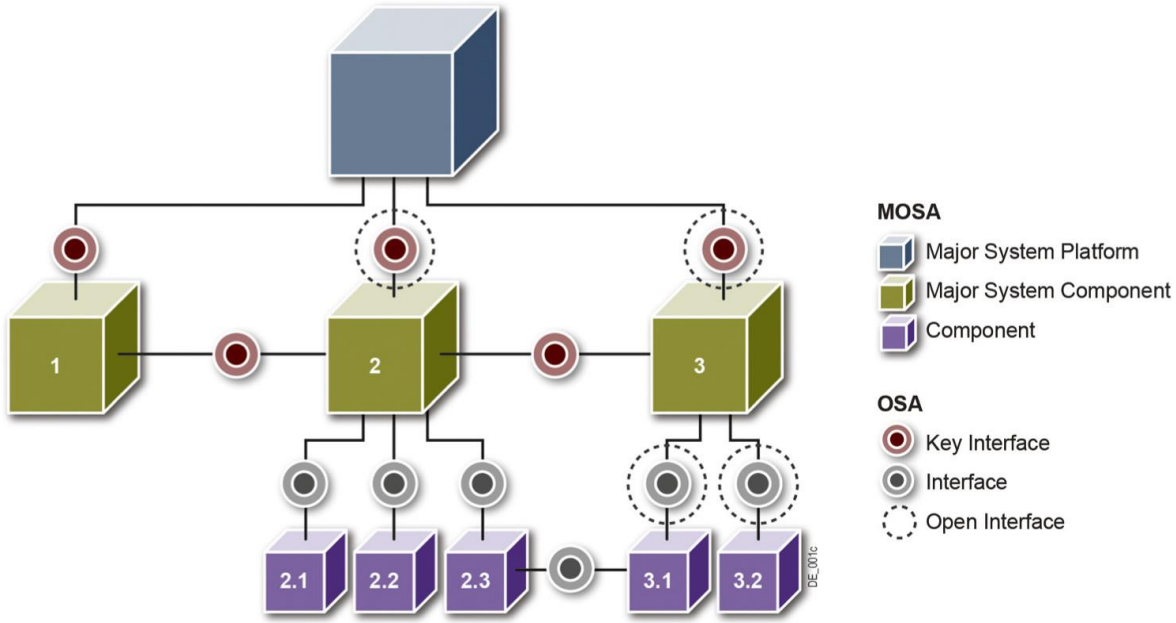
U.S.C. Title 10 §4401 Definition of MOSA

1. The term "modular open system approach" means, with respect to a major defense acquisition program, an integrated business and technical strategy that—
   A. employs a modular design that uses modular system interfaces between major systems, major system components, and modular systems;
   B. is subjected to verification to ensure modular system interfaces
      i. comply with, if available and suitable, widely supported and consensus-based standards; or
      ii. are delivered pursuant to the requirements established in subsection (a)(2)(B) of section 804 of the William M. (Mac) Thornberry National Defense Authorization Act for Fiscal Year 2021 -
   C. uses a system architecture that allows severable major system components at the appropriate level to be incrementally added, removed, or replaced throughout the life cycle of a major system platform to afford opportunities for enhanced competition and innovation while yielding -

(*Office of Law Revision Council, United States Code*, n.d.)

Two major themes in MOSA implementation

- Modularity
  - "Degree to which systems, major constitutive subsystems and components within a system, and major subsystems and components across subsystems can function as modules that can communicate across component boundaries and through interfaces and can be separated and recombined to achieve various effects, missions, or capabilities." (*Text - H.R.6395 - 116th Congress (2019-2020): William M. (Mac) Thornberry National Defense Authorization Act for Fiscal Year 2021 | Congress.Gov | Library of Congress*, n.d.)
- Interface Standardization
  - Logical and physical interface aspects are based on and comply with widely-supported, consensus-based standards
- Both lead to increased integration of components and interoperability between systems among other benefits



(*Zimmerman et al., 2019*)

The BUCK STOPS here!

- MOSA compliance is the responsibility of program offices where the Program Manager (PM) supported by Systems Engineers (SE) makes the ultimate decision on a particular implementation

OK…but where do you begin?

PM and SE MOSA strategy development responsibilities

As identified in the Acquisition Workforce Qualification Initiative:

1. Identify open system standards applicable to program/system
2. Direct the development of modular designs based on standards
3. Determine and institute enterprise investment best practices
4. Direct the strategic use of data rights
5. Implement life cycle sustainment practices
6. Verify system designs are available and shared between system development activities and program management
7. Determine risks and opportunities for open systems architecture application
8. Document

(*Modular Design*, n.d.)

PM and SEs need a starting point that is flexible for their specific program/specific technical and business needs

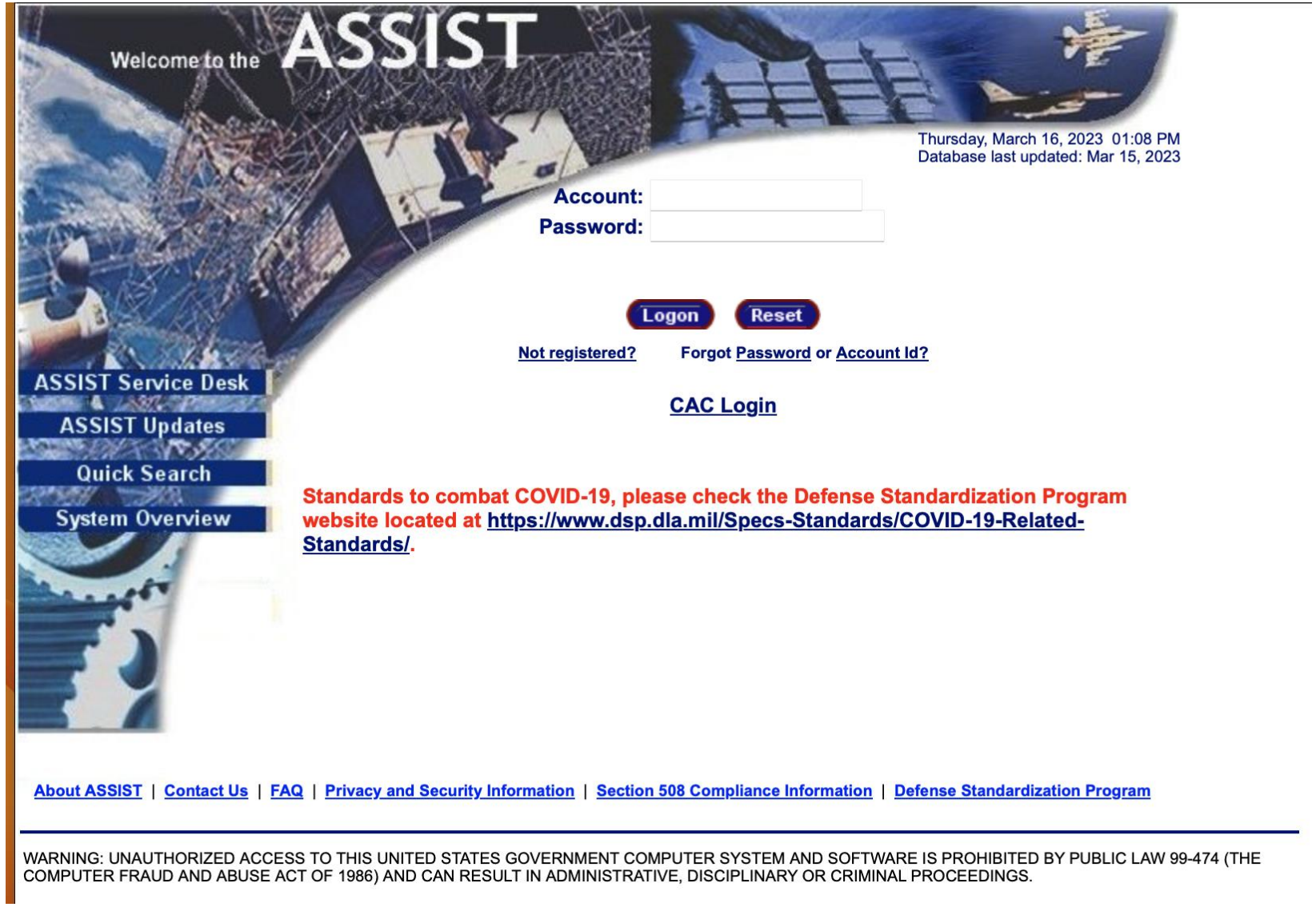
START HERE!

Start with Number 1

1. Identify open system standards applicable to program/system
2. Direct the development of modular designs based on standards
3. Determine and institute enterprise investment best practices
4. Direct the strategic use of data rights
5. Implement life cycle sustainment practices
6. Verify system designs are available and shared between system development activities and program management
7. Determine risks and opportunities for open systems architecture application
8. Document

(*Modular Design*, n.d.)

It’s easy, just use ASSIST…



Proceed with caution

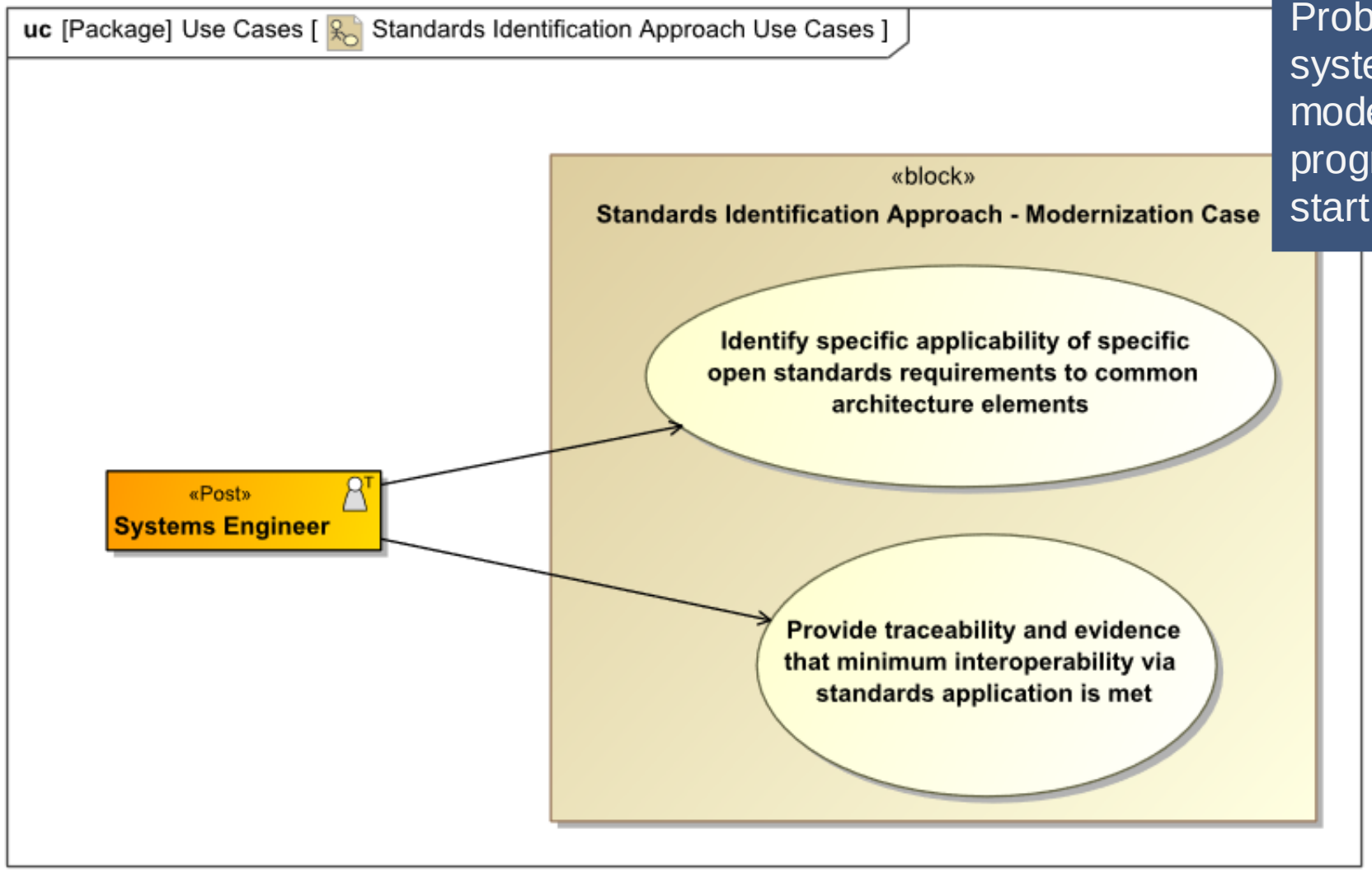
- Numerous consensus-based standards available
- Many standards…
  - require deep technical knowledge of the standard
  - overlap and compete and/or conflict with each other
  - present options that when chosen may not interoperate with other implementations of the same standard using different options
- Each standard…
  - talks about things uniquely within the scope of the standard
  - defines adherence of an implementation to the standard differently, e.g. compliance vs. conformance

HELP!

- Which standards and which parts of the standard are applicable to my system?
- I don't have time to become intimately familiar with all the standards
- How do I choose the standards that don't conflict?
- How do I choose an option that will interoperate with other implementations of the same standard?
- How do I decode all the concepts and terms into a coherent, consistent understanding?

We are here to help!

Standards Identification Approach



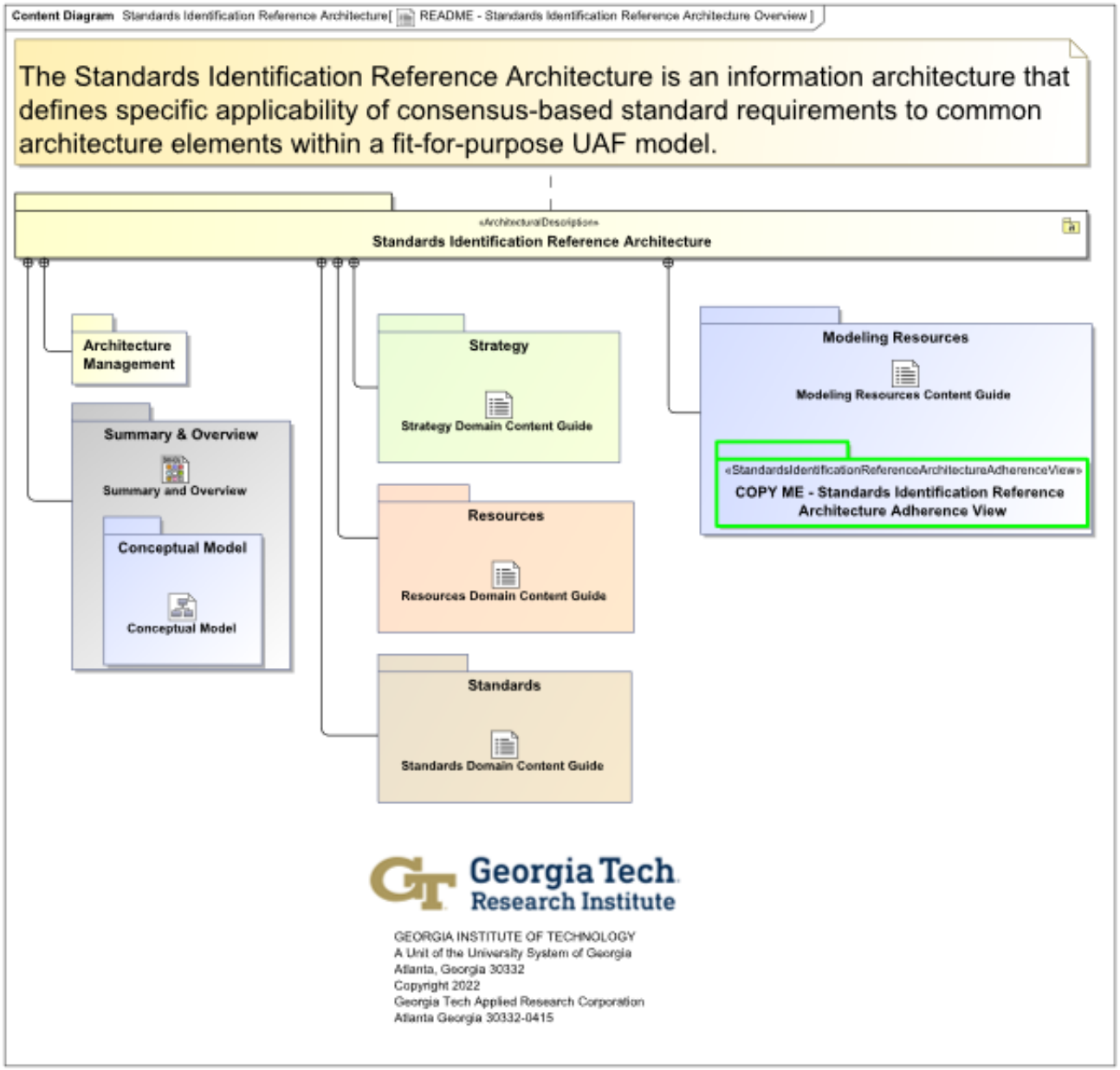
Problem scoped to system modernization programs vice new start

Design Challenges for the Approach

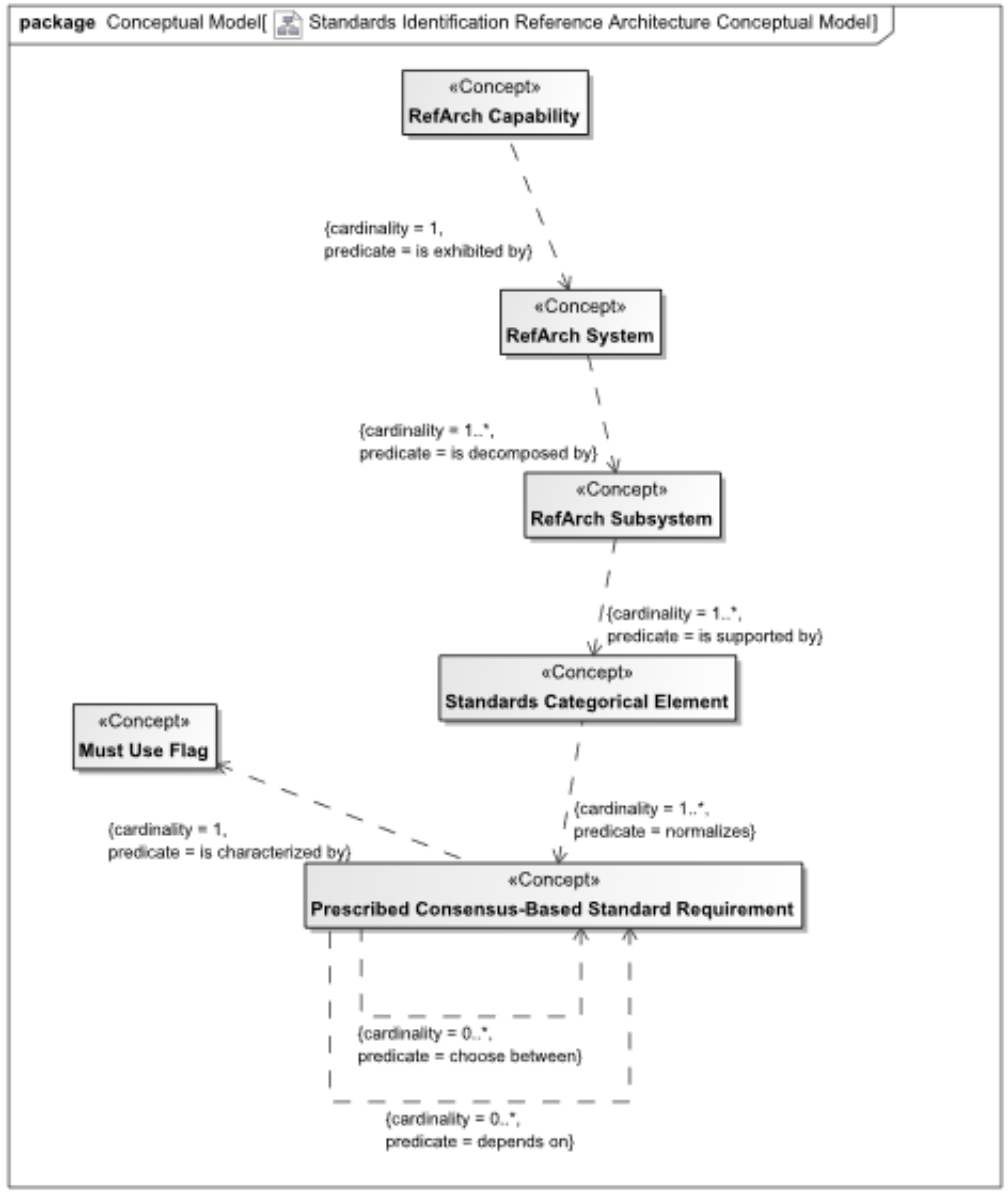
- Define how to normalize and unify the applicable concepts extracted from the relevant standards
- Create a foundation for the various standards to work together to a level of commonality understood and applicable to all systems within the domain of application and for the entire enterprise
- Give program offices the flexibility they need to choose the right standards based on their specific business and technical needs
- Enable traceability and evidence for minimum interoperability via standards application
- Make it reusable

Standards Identification Reference Architecture

- UAF fit-for-purpose information architecture designed to:
  - Identify, categorize, and normalize standards into common architecture elements within a capability domain
  - Present options, dependencies, and conflicts among standards
  - Support flexibility, innovation, and ability to meet unique program needs while constraining overall space to appropriate levels of commonality
  - Promote information discovery
  - Provide mechanism for traceability and evidence of adherence for referencing architectures



Reference Architecture Conceptual Model



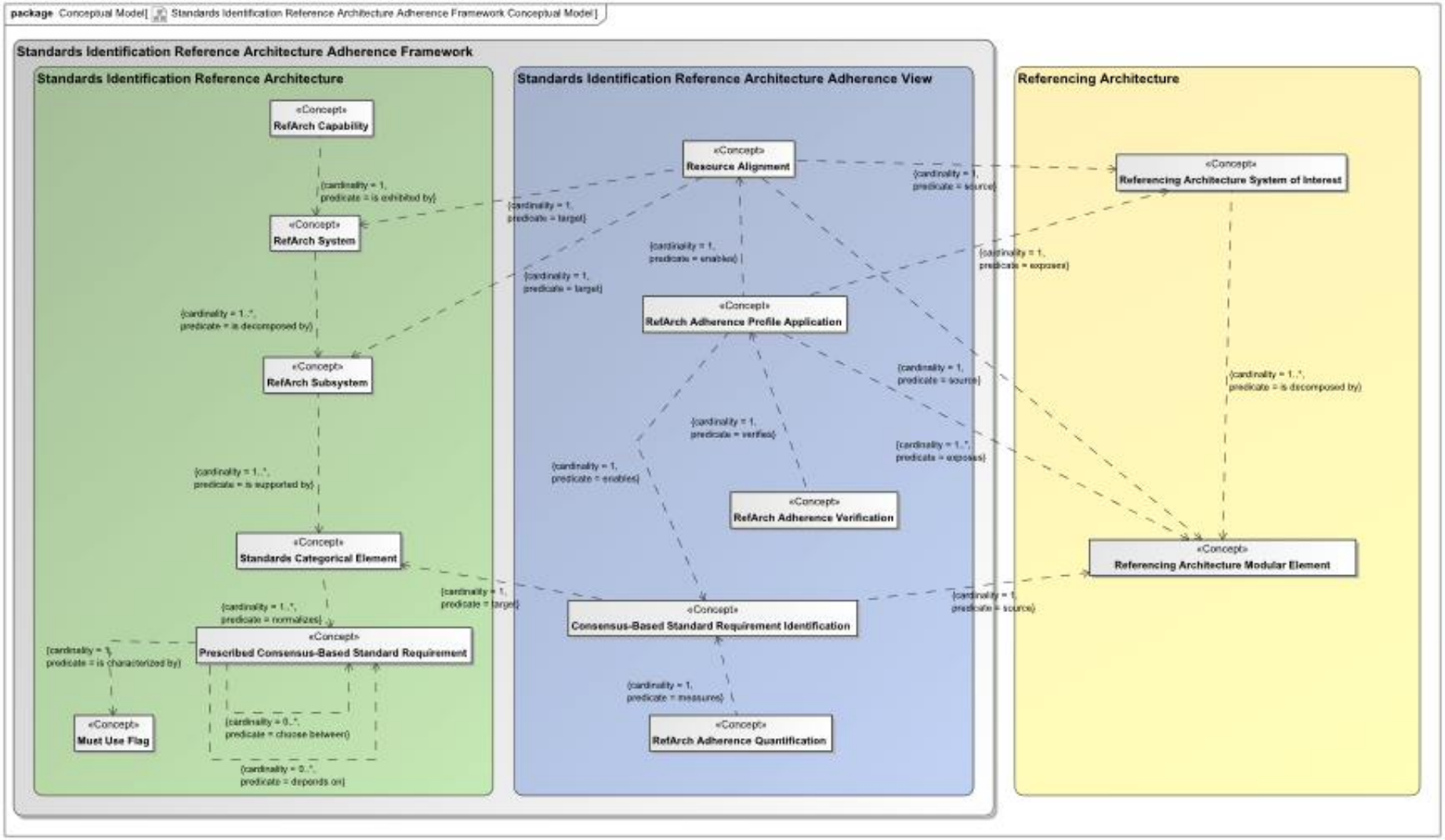
- Minimally constraining common system description
- Standards prescribed to maximize commonality among systems within a domain

- Standards Categorical Elements categorize and normalize standards under a common, well-understood architecture element
- Must Use Flag to indicate those standard requirements that must be used in order to satisfy minimum interoperability

This is great…but how do I, a Program Office SE, use it?

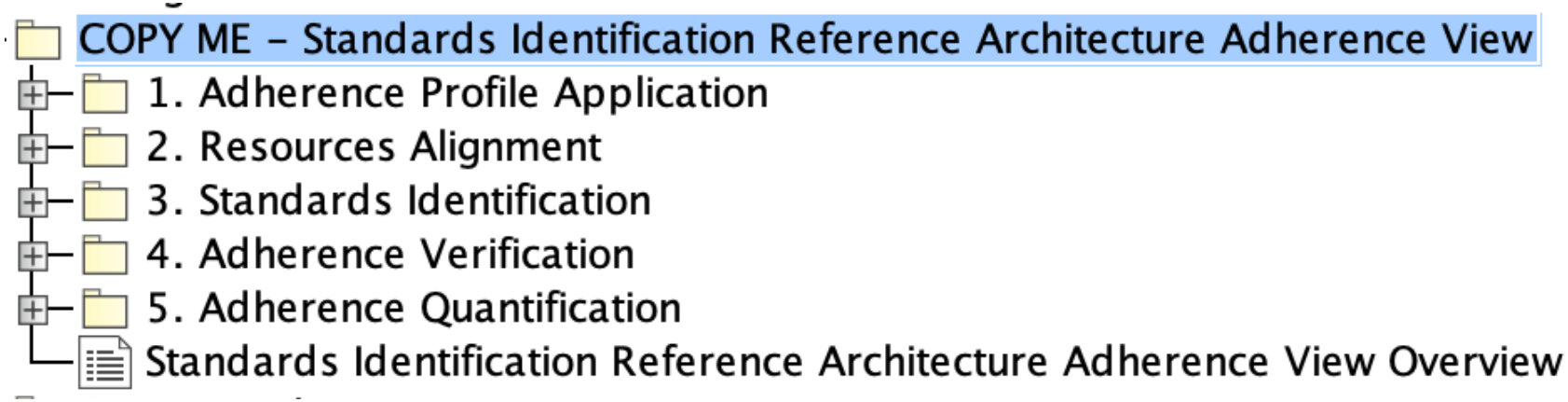
- Only certain parts of my architecture are concerned with complying to standards
- How do I discover and identify relevant standard requirements?
- How do I know if I'm using it the right way?
- How do I know if I'm any closer to MOSA compliance?

Standards Identification Reference Architecture Adherence Framework



Georgia Tech Research Institute

Standards Adherence Reference Architecture Adherence View



5 packages with smart packages, preconfigured matrices, and tables that guide a SE through:

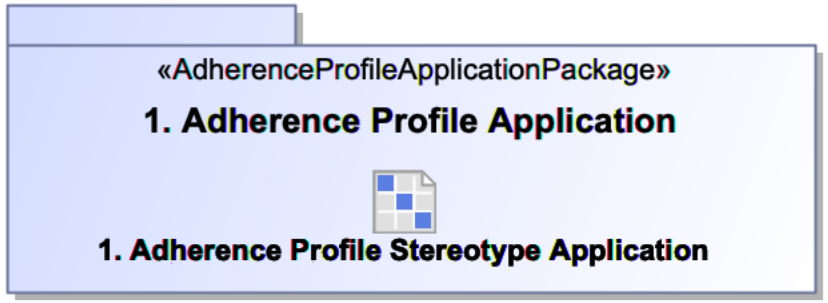
- Exposing relevant architecture elements
- Asserting capability and functional alignment
- Identifying relevant and applicable standards
- Choosing among options
- Declaring exceptions to required standards
- Analyzing verification of tracing to the Ref Arch
- Quantifying adherence to the Ref Arch

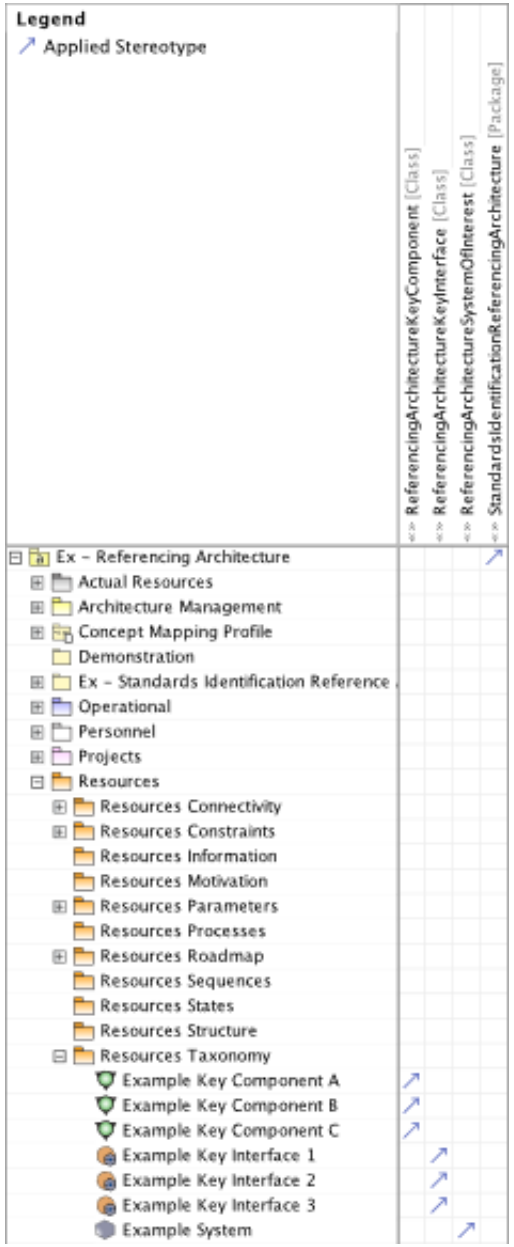
OK! Let’s GO!!

Georgia Tech Research Institute

Reference Architecture used as read-only project in Program Office Architecture

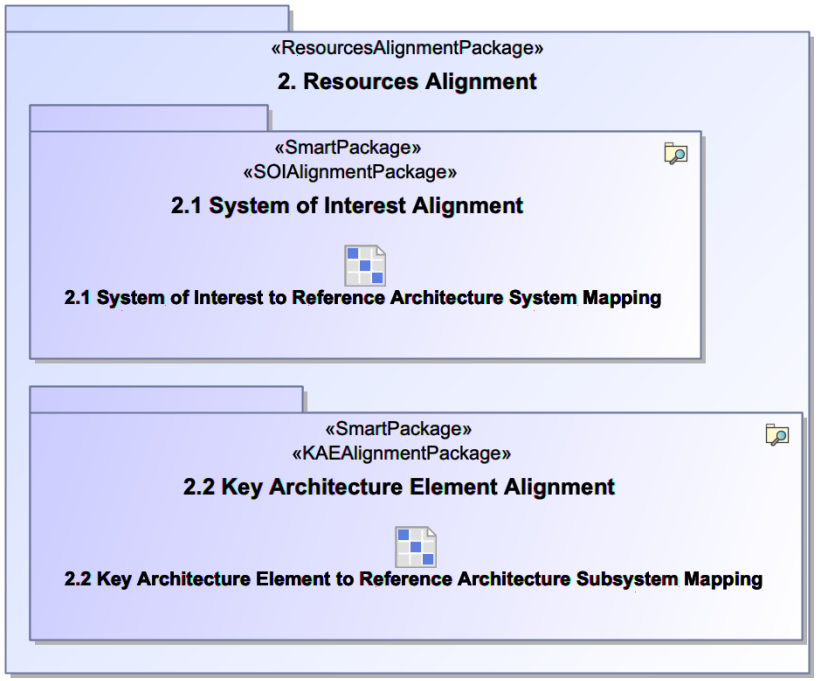
Apply Adherence Profile

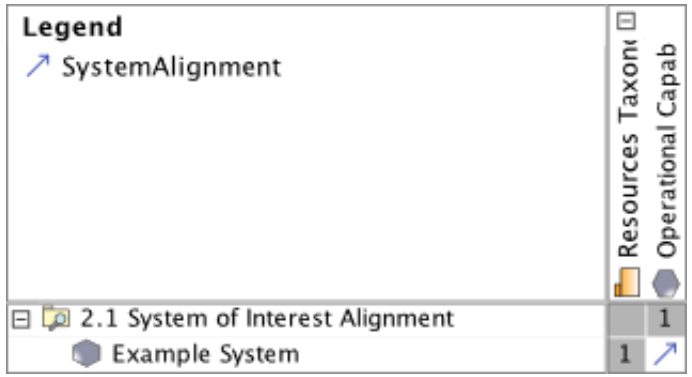


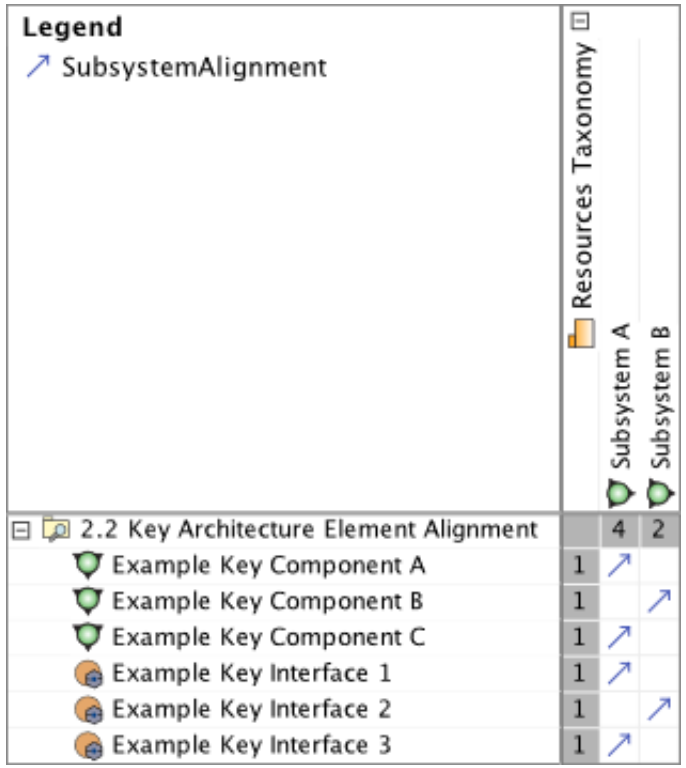


- Matrix preconfigured to facilitate applying information exposing stereotypes to relevant parts of the Program Office Architecture
- Not all components need be modularized and not all interfaces need comply with a standard
  - Only stereotype those that are needed

Functionally align Program Office Architecture to Reference Architecture

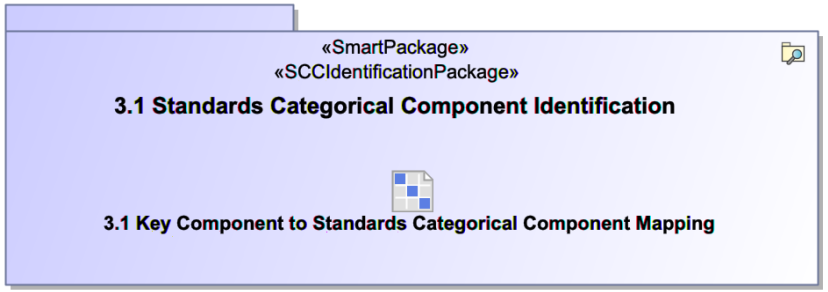


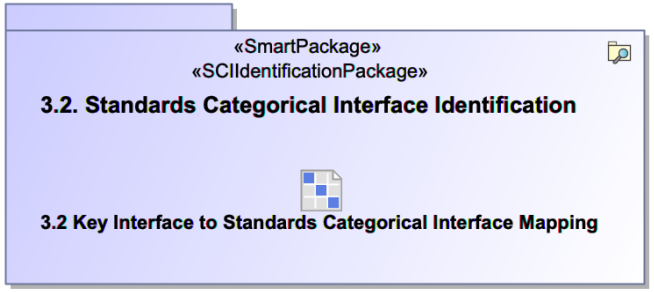


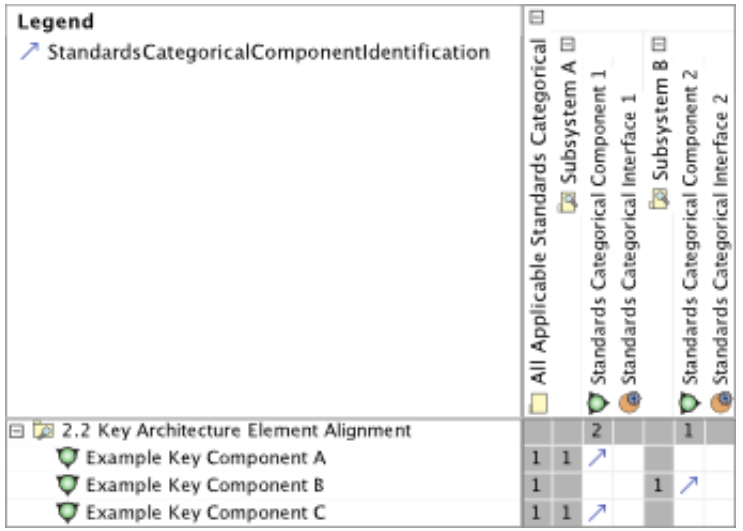


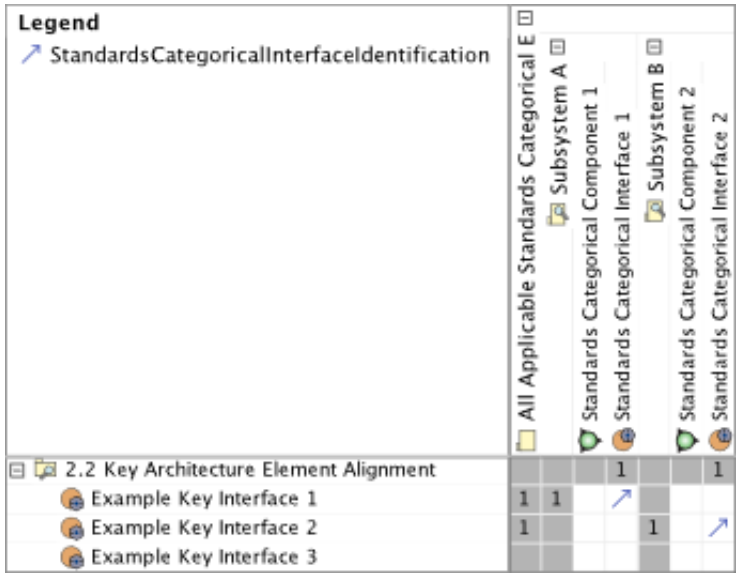
- Many kinds of systems may be defined in the Reference Architecture if the capability domain is broad
  - Align with the appropriate, specific system
- Not all subsystems in the Reference Architecture need to be aligned to a Program Office Architecture
  - Only the ones that match basic functional description of key components and interfaces

Identify applicable standards



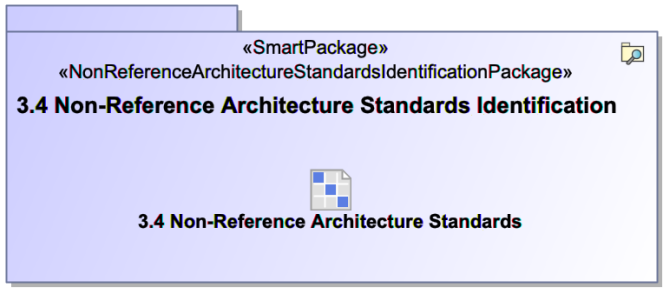


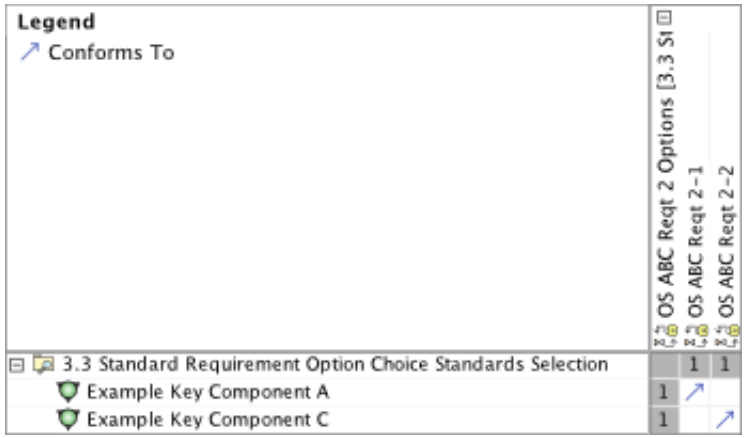


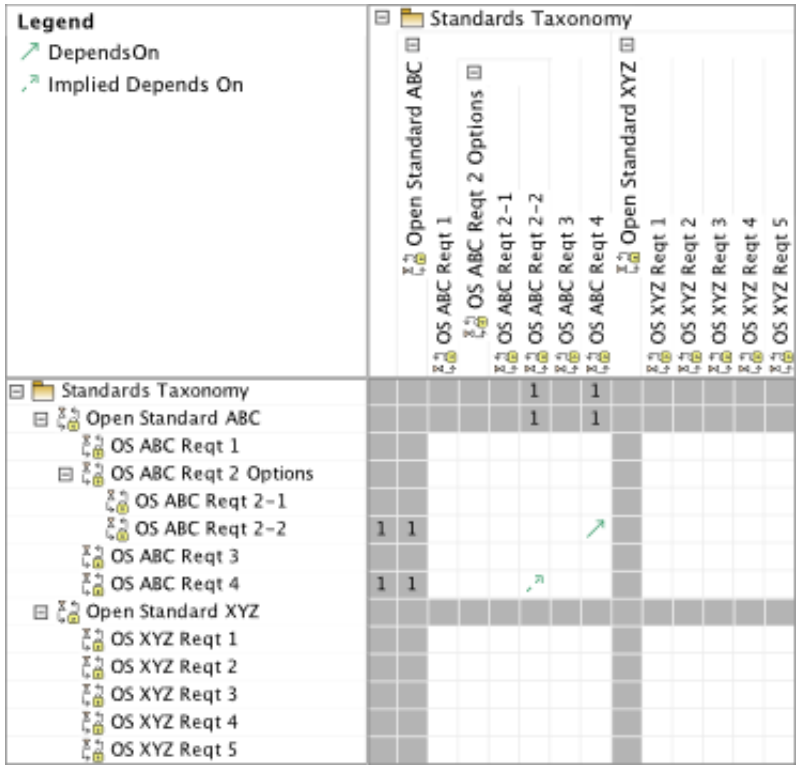


- Map Key Elements in the Program Office Architecture to Standards Categorical Elements that are logically and physically similar
- Constraints prevent mapping to Categorical Elements not in aligned subsystem
- Not all Key Elements will map
  - Can't know everything

Select standard requirement option choice

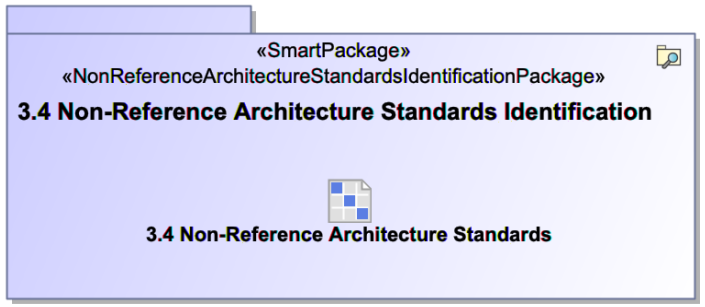


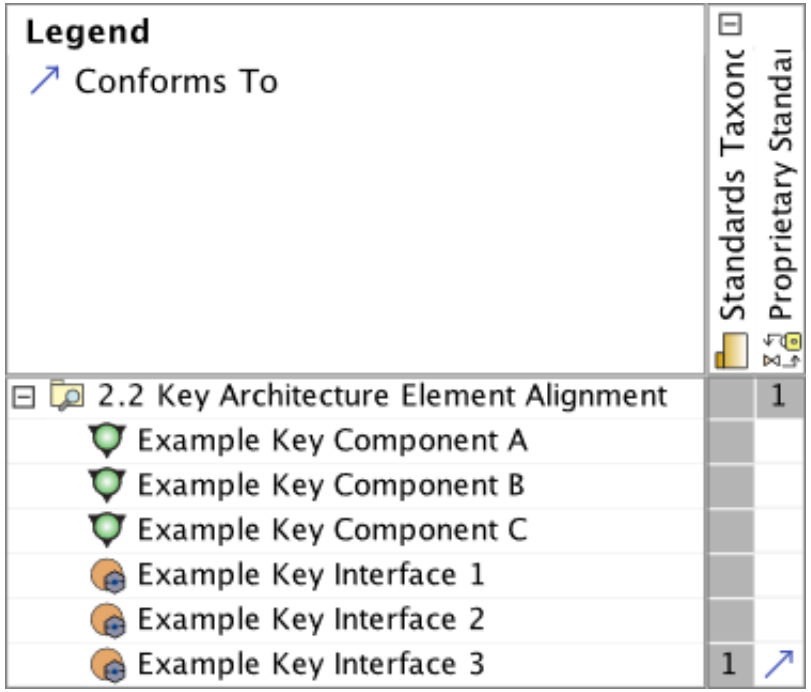




- Many standard requirements present options
  - Choose the one that makes the most business and technical sense
- Selections may result in additional, indirect standard requirements due to dependencies

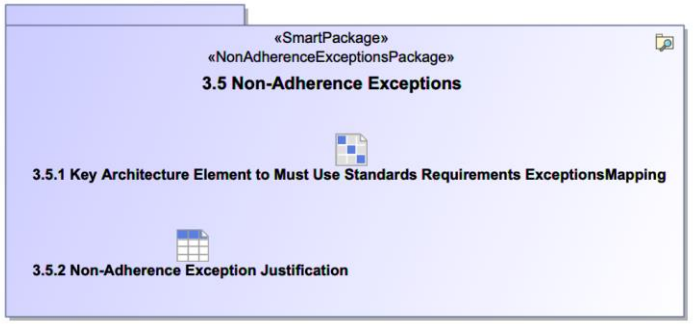
Map to standards not identified in Reference Architecture

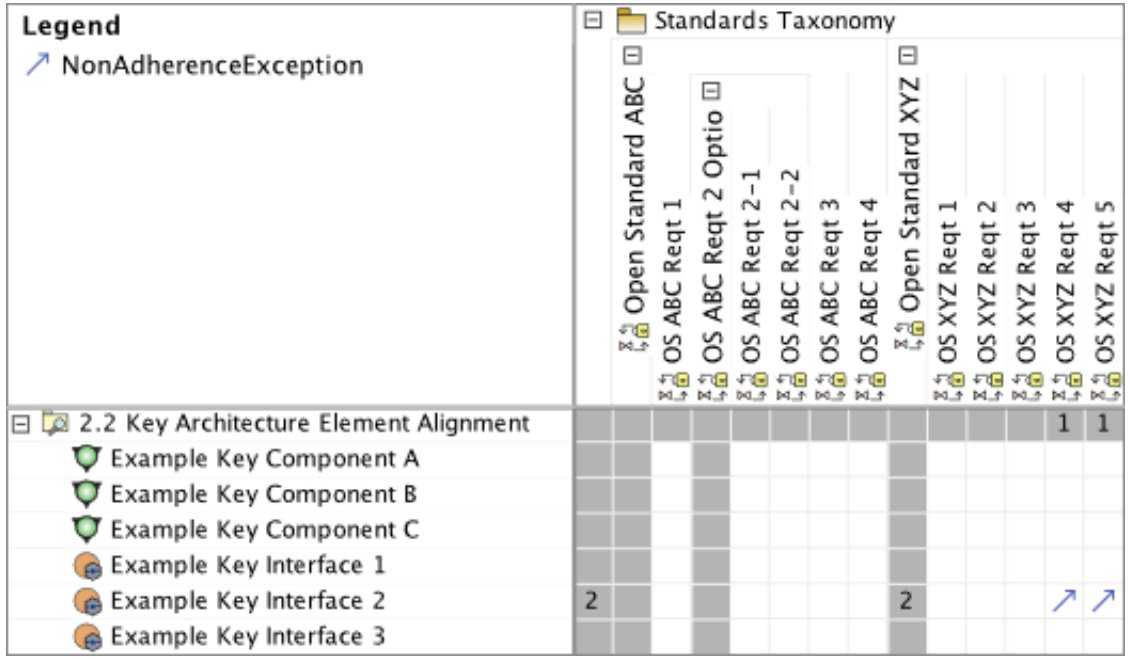




- Every key element must comply with a standard
  - Either identified in Reference Architecture
  - Or specific to the Program Office Architecture

Declare exceptions to must use requirements and provide justification





- It may not make technical or business sense to map to standard requirements flagged as must use
- Must declare exception and provide justification

#	Key Architecture Element	Non Adhering Must Use Reference Architecture Standard Requirement	Justification	Authority	Date
1	Example Key Interface 2	OS XYZ Reqt 4	Use could result in significant cost and schedule increase.	Gina Burdell	3/17/23
2	Example Key Interface 2	OS XYZ Reqt 5	Do not agree that this requirement aids in minimum interoperability.	Gina Burdell	3/17/23

View Standards Traceability Summary

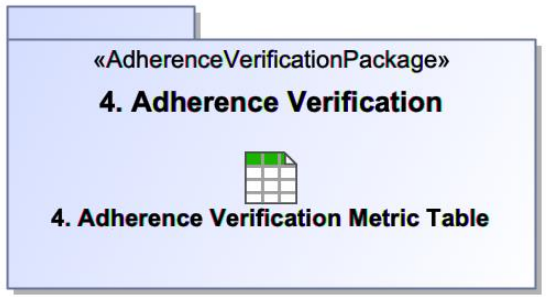
#	Name	Identified Reference Architecture Standards Categorical Elements	Identified Reference Architecture Standards	Chosen Reference Architecture Optional Standard Requirements	Choice Resultant Reference Architecture Standard Requirement	Must Use Reference Architecture Standard Requirements	Excluded Reference Architecture Must Use Standard Requirements	Referencing Architecture Specific Standard Requirements	All Applied Standards
1	Example Key Component A	Standards Categorical Component 1	OS ABC Reqt 2 Options OS ABC Reqt 1 OS ABC Reqt 4	OS ABC Reqt 2-1		OS ABC Reqt 1 OS ABC Reqt 2-1			OS ABC Reqt 1 OS ABC Reqt 2-1
2	Example Key Component B	Standards Categorical Component 2	OS ABC Reqt 3			OS ABC Reqt 3			OS ABC Reqt 3
3	Example Key Component C	Standards Categorical Component 1	OS ABC Reqt 2 Options OS ABC Reqt 1 OS ABC Reqt 4	OS ABC Reqt 2-2	OS ABC Reqt 4	OS ABC Reqt 1 OS ABC Reqt 2-2 OS ABC Reqt 4			OS ABC Reqt 1 OS ABC Reqt 2-2 OS ABC Reqt 4
4	Example Key Interface 1	Standards Categorical Interface 1	OS XYZ Reqt 1 OS XYZ Reqt 3			OS XYZ Reqt 1			OS XYZ Reqt 1
5	Example Key Interface 2	Standards Categorical Interface 2	OS XYZ Reqt 5 OS XYZ Reqt 2 OS XYZ Reqt 4			OS XYZ Reqt 5 OS XYZ Reqt 4 OS XYZ Reqt 2	OS XYZ Reqt 4 OS XYZ Reqt 5		OS XYZ Reqt 2
6	Example Key Interface 3							Proprietary Standard Rec	Proprietary Standard I

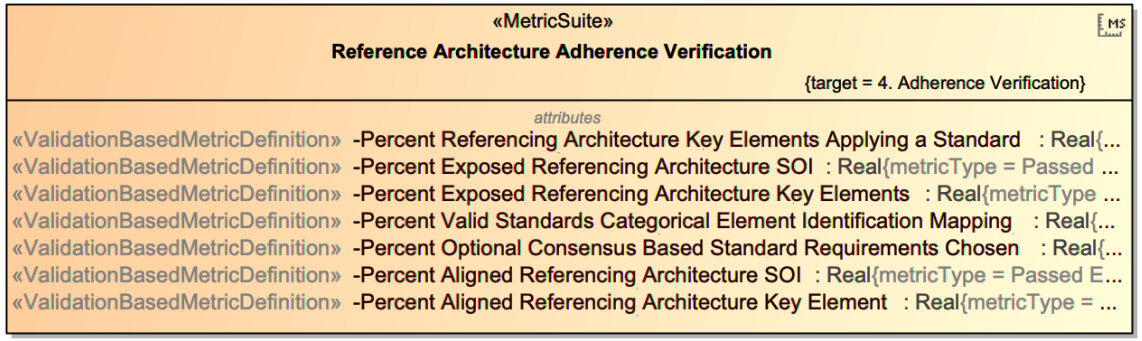
Easily derived requirements can be levied on vendors

Auto-Generated Requirements

#	△ Id	Name	Text	Refined By
1	STDREQ-1	STDREQ-1 Example Key Component A - OS ABC Reqt 1 Requirement	The Example Key Component A shall adhere to OS ABC Reqt 1.	Example Key Component A
2	STDREQ-2	STDREQ-2 Example Key Component A - OS ABC Reqt 2-1 Requirement	The Example Key Component A shall adhere to OS ABC Reqt 2-1.	Example Key Component A
3	STDREQ-3	STDREQ-3 Example Key Component B - OS ABC Reqt 3 Requirement	The Example Key Component B shall adhere to OS ABC Reqt 3.	Example Key Component B
4	STDREQ-4	STDREQ-4 Example Key Component C - OS ABC Reqt 1 Requirement	The Example Key Component C shall adhere to OS ABC Reqt 1.	Example Key Component C
5	STDREQ-5	STDREQ-5 Example Key Component C - OS ABC Reqt 2-2 Requirement	The Example Key Component C shall adhere to OS ABC Reqt 2-2.	Example Key Component C
6	STDREQ-6	STDREQ-6 Example Key Component C - OS ABC Reqt 4 Requirement	The Example Key Component C shall adhere to OS ABC Reqt 4.	Example Key Component C
7	STDREQ-7	STDREQ-7 Example Key Interface 1 - OS XYZ Reqt 1 Requirement	The Example Key Interface 1 shall adhere to OS XYZ Reqt 1.	Example Key Interface 1
8	STDREQ-8	STDREQ-8 Example Key Interface 2 - OS XYZ Reqt 2 Requirement	The Example Key Interface 2 shall adhere to OS XYZ Reqt 2.	Example Key Interface 2
9	STDREQ-9	STDREQ-9 Example Key Interface 3 - Proprietary Standard Reqt 1234.1 Requirement	The Example Key Interface 3 shall adhere to Proprietary Standard Reqt 1234.1.	Example Key Interface 3

Run analysis to determine if mappings have been done correctly

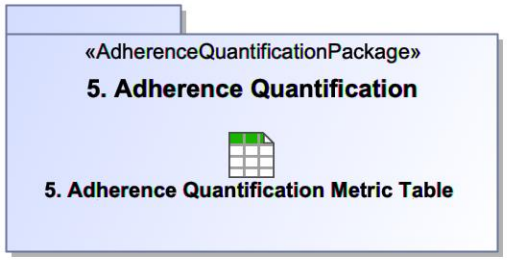


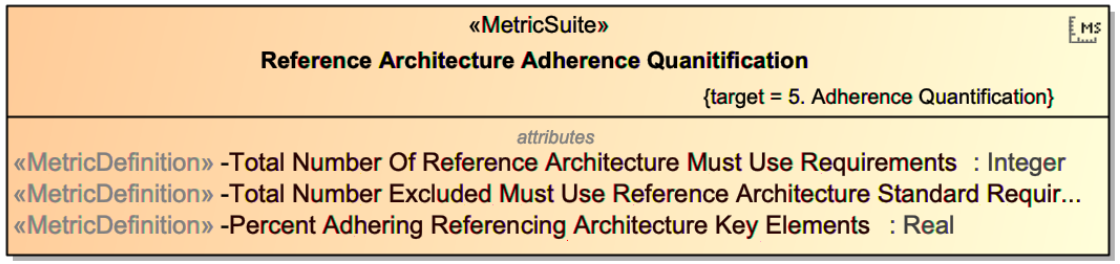


- Validation based metrics defined to to quantify degree of well-formedness to the meta-model underpinning the framework

#	Date	Scope	Percent Aligned Referencing Architecture SOI	Percent Aligned Referencing Architecture Key Element	Percent Exposed Referencing Architecture SOI	Percent Exposed Referencing Architecture Key Elements	Percent Valid Standards Categorical Element Identification Mapping	Percent Optional Consensus Based Standard Requirements Chosen	Percent Referencing Architecture Key Elements Applying a Standard
1	2023.03.07 09.52	Ex - Referencing Architecture	100	100	100	100	100	100	100

Run analysis to quantify degree of adherence to Reference Architecture



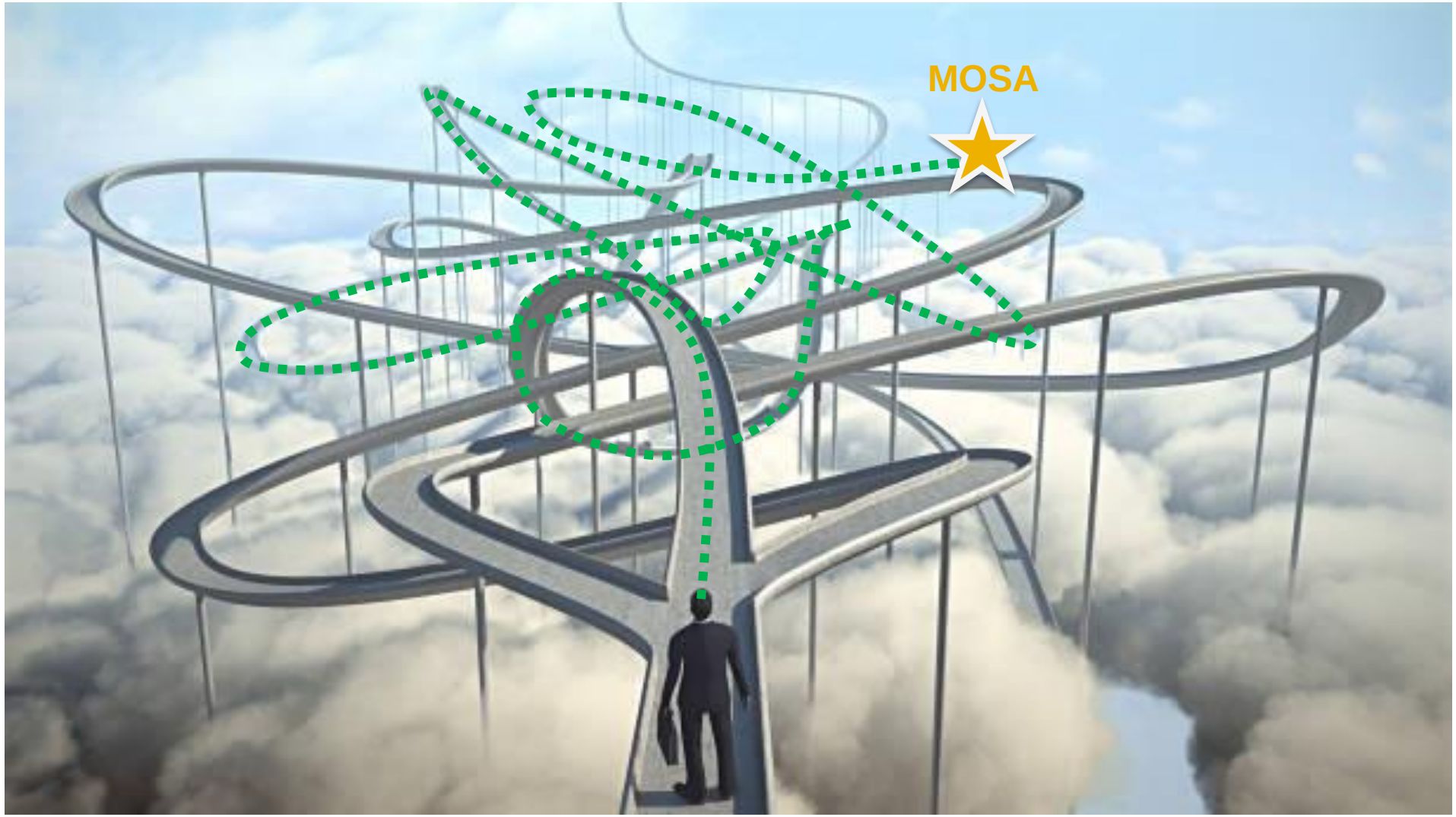


#	Date	Scope	Total Number Of Reference Architecture Must Use Requirements	Total Number Excluded Must Use Reference Architecture Standard Requirements	Percent Adhering Referencing Architecture Key Elements
1	2023.03.07 09.52	Ex – Referencing Architecture	10	2	80

$$Adherence = \left(1 - \frac{Number\ Excluded\ Must\ Use\ Requirements}{Total\ Number\ Must\ Use\ Requirements}\right) x\ 100$$

All good!

Almost there!

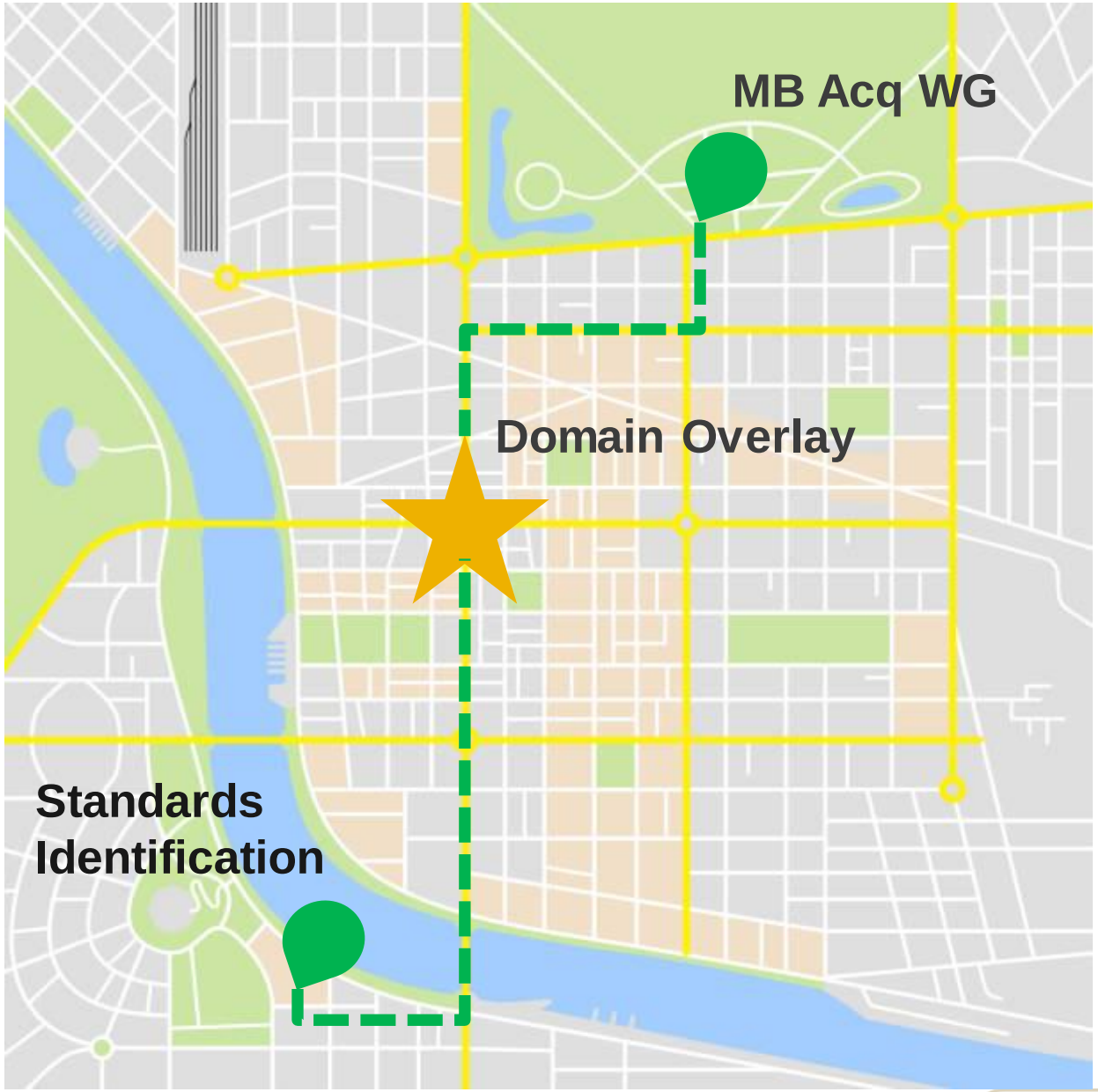


But wait, isn’t this a Domain Overlay?

Yes, yes it is…


ISN'T IT IRONIC?

But we developed it independently…



Characteristics of a Domain Overlay

- Domain Overlay: A collection of constructs needed to support analysis for a domain specific concern using a standardized approach.

- Characteristics:
  - Usually has associated regulations, governance that can be treated as pseudo requirements or constraints
  - Cross-cutting both viewpoints/rows & aspects/columns
  - Supports specific analysis associated with a Domain-Specific concern
  - Can be created independent of a specific solution architecture description
  - Can be applied or removed from a specific architecture description without impacting the AD, hence an overlay

(*Hart & Anderson, 2022*)

Characteristic 1

Domain Overlay -

- Usually has associated regulations, governance that can be treated as pseudo requirements or constraints

Standards Identification Reference Architecture Adherence Framework -

- Consensus-based standards prescribed to a capability domain are applied to Standards Categorical Elements and consequently constrain all referencing architectures

- Standards deemed “Must Use” within a capability domain must either be used or excepted with justification in the referencing architecture

Characteristic 2

Domain Overlay -

- Cross-cutting both viewpoints/rows & aspects/columns

Standards Identification Reference Architecture Adherence Framework -

- Most Standards Adherence Reference Architecture Adherence View cut across Resources viewpoint and Traceability aspect and Standards viewpoint and Traceability aspect

Characteristic 3

Domain Overlay -

- Supports specific analysis associated with a Domain-Specific concern

Standards Identification Reference Architecture Adherence Framework -

- Created with the purpose of providing a starting point for program offices to achieve MOSA compliance via identifying consensus-based standards

Characteristic 4

Domain Overlay -

- Can be created independent of a specific solution architecture description

Standards Identification Reference Architecture Adherence Framework -

- Adherence view is capability domain agnostic and consequently specific solution architecture agnostic

Characteristic 5

Domain Overlay -

- Can be applied or removed from a specific architecture description without impacting the AD, hence an overlay

Standards Identification Reference Architecture Adherence Framework -

- Adherence view annotates the referencing architecture description when applied

- The original AD is left unchanged when the Adherence view is removed

In summary

- The Standards Identification Reference Architecture Adherence Framework was developed to aid Program Office Systems Engineers in identifying the consensus-based standards applicable to their system and provide traceable and justifiable evidence for meeting minimum interoperability, i.e. adherence to the Reference Architecture
  - But it isn't finished…
- The Adherence Framework can be applied to any capability domain
  - Can it be generalized for other concerns…
- The Adherence Framework is a Domain Overlay
  - If it looks like a duck, swims like a duck, and quacks like a duck, then it probably is a Domain Overlay
  - The components of a Domain Overlay are intuitive and naturally discoverable
- The UAF grid made thinking about the problem easier
- UAFML provided a rich vocabulary to build on when architecting the framework

References

- *About AWQI*. (n.d.). Retrieved March 15, 2023, from https://www.dau.edu:443/tools/awqi/p?TermStoreId=dce24bd1-67d5-449a-a845-034b26f02d5e&TermSetId=d4eb1a2a-cab0-430d-a021-47313648a8a6&TermId=2df32fb9-8d95-40fd-9d47-3040988b07a9&UrlSuffix=About-AWQI
- Amira, D. (2012, October 16). *Everyone Is Butchering 'the Buck Stops Here.'* Intelligencer. https://nymag.com/intelligencer/2012/10/buck-stops-here-clinton-obama-truman.html
- *EWorkbook*. (n.d.). Retrieved March 15, 2023, from https://www.dau.edu/tools/awqi/p/eWorkbook
- Hart, L. (n.d.). *Actionable Architecture Using Aspect Modeling*.
- Hart, L., & Anderson, R. (2022). *OMG UAF Model-based Acquisition Analytic Viewpoint Overlays (AVO)*.
- *Modular Design*. (n.d.). Retrieved March 15, 2023, from https://www.dau.edu:443/tools/se-brainbook/Pages/Design%20Considerations/Modular-Design.aspx
- *Modularity (glossary)—SEBoK*. (n.d.). Retrieved March 15, 2023, from https://sebokwiki.org/wiki/Modularity_(glossary)
- *Office of Law Revision Council, United States Code*. (n.d.). OLRC Home. Retrieved March 15, 2023, from https://uscode.house.gov/browse/prelim@title10/subtitleA/part5/subpartF/chapter327/subchapter1&edition=prelim
- *Text—H.R.6395—116th Congress (2019-2020): William M. (Mac) Thornberry National Defense Authorization Act for Fiscal Year 2021 | Congress.gov | Library of Congress*. (n.d.). Retrieved March 15, 2023, from https://www.congress.gov/bill/116th-congress/house-bill/6395/text
- Zimmerman, P., Ofori, M., Barrett, D., Soler, J., & Harriman, A. (2019). Considerations and examples of a modular open systems approach in defense systems. *The Journal of Defense Modeling and Simulation*, *16*(4), 373–388. https://doi.org/10.1177/1548512917751281

Backup

Abstract

So you have to be MOSA compliant? What does that mean? A Modular Open Systems Approach (MOSA) according to Title 10 of the U.S. Code means that a compliant system will employ a modular design and that major systems interfaces be compliant with consensus-based standards. The better question then is, how do you get started? We have developed an approach in UAF that gives system developers a leg-up to MOSA compliance at least with respect to the identification and application of consensus-based standards. It was after we created this approach that we became aware of the Domain Overlay concept work being done by the OMG, and in a great sense of irony, validated our approach. This is the story of that journey, and how we see this approach being useful to our MBSE community.