Limits to the Use of the Zachman Framework in Developing and Evolving Architectures for Complex Systems of Systems

Philip Boxer, Suzanne Garcia

May 6th, 2009

Agenda

- The Enterprise/Architecture relationship
  - The demands of collaborative systems of systems
  - Limits to the use of the Zachman Framework & the consequences for DODAF 2.0
- Summary
**Limits to the Use of the Zachman Framework in Developing and Evolving Architectures for Complex Systems of Systems**

**Architectural Genres:**

different genres for different purposes

The primary interfaces across genres as evidenced by working group discussions:

- **Enterprise Architecture**
  - Enabler
  - Quality attributes
  - Mutually constraining

- **System architecture**
  - Software architect mostly on the receiving end

- **Software architecture**

These genres reflect a *supply-side perspective* on the enterprise

The Enterprise Architecture defines the way it creates value: *Zachman roots to DODAF*

**Source of coloured squares:** Zachman Framework, www.zifa.com

---

**Agenda**

- **The Enterprise/Architecture relationship**
  - The demands of collaborative systems of systems
- Limits to the use of the Zachman Framework & the consequences for DODAF 2.0
- **Summary**
Speech by Secretary Gates:
*There are two paradigms that must coexist*

The need for state of the art systems – particularly longer range capabilities – will never go away…

We also need specialized, often relatively low-tech equipment for stability and counter-insurgency missions.

- How do we institutionalize rapid procurement and fielding of such capabilities?
- Why do we currently have to go outside the normal bureaucratic process?

Our conventional modernization programs seek a 99% solution in years.

*Stability and counter-insurgency missions require 75% solutions in months.*

- The challenge is whether in our bureaucracy and in our minds these two different paradigms can be made to coexist.


---

**The three tempos:** analyzing the impact of the enterprise’s relation to customers’ changing demands
Managing diverging tempos: the readiness tempo has to be managed in its own right

The two paradigms are about diverging acquisition and demand/threat tempos
• Their coexistence depends on managing the readiness tempo in its own right

Managing the readiness tempo means:
• sustaining multiple collaborations between players able to address concurrent types of demand/threat
• building organizational agility into the supporting socio-technical infrastructures

Governance of a Collaborative SoS: involves multiple collaborations with a supporting infrastructure

The players in a collaboration can be spread across multiple enterprises and/or different parts of a single enterprise

Larger stakeholder context
Collaborations of Players
Supporting Infrastructure
Governance
Multiple value-creating relationships

It is the players participating in a particular collaboration who will define
• Their system-of-interest and its environment
• The stakeholders they judge to be relevant
• The way they want their collaboration supported by the infrastructure
And so... a demand-side perspective needs to be added

Collaborative SoS present a different order of complexity

This complexity arises because
- multiple collaborations between players exist concurrently,
- each with its own relationship to demand/threat, and
- supported by a shared infrastructure

It means adding a demand-side perspective on the collaborations

Managing both paradigms: means managing the relationship between the two 'V's

Effects on Demand/Threat
- Multiple Concurrent Collaborations... constrains what is possible up here
- Demand-Threat Tempo
- Mission Command
- Composite Capabilities
- Force Structures
- Demand-side Supply-side

Capability gap
- Requirement
- Design decomposition
- Acquisition
- System integration
- System components
- What happens down here...

Agenda

The Enterprise/Architecture relationship
The demands of collaborative systems of systems

➤ Limits to the use of the Zachman Framework & the consequences for DODAF 2.0

Summary

The demand-side perspective: creates gaps in Zachman

Limits to the use of the Zachman Framework
Philip Boxer & Suz Garcia, May 6th 2009
© 2009 Carnegie Mellon University

DODAF 2.0 Entities and Views:
what gets modeled?

Entities not modeled by DODAF 2.0:
the demand-side perspective is not included

The relationships to these entities are not dealt with in
DODAF 2.0 models
There is a table and a diagram on the page. The diagram is a flowchart representing the Zachman Framework, which is a framework for modeling the context of a system. The framework is divided into five perspectives: Scope, Collaborative Model, Business Model, System Model, and Technology Model. Each perspective is further divided into sub-perspectives and sub-sub-perspectives. The flowchart shows the relationships and dependencies between these perspectives and sub-perspectives.

The Agenda section outlines the topics to be covered in the presentation:

- The Enterprise/Architecture relationship
- The demands of collaborative systems of systems
- Limits to the use of the Zachman Framework & the consequences for DODAF 2.0

The Summary section will likely provide conclusions and takeaways from the discussion.
Summary: both supply-side and demand-side perspectives need to be modeled

Supporting the development of collaborative systems of systems involves modeling more than the supply-side entities in Zachman-rooted representations like DODAF 2.0

- Including a demand-side perspective means being able to account for
  - cross-cutting synchronization, not just hierarchical accountability
  - multi-enterprise development and co-evolution
  - inherent variation in the way user's demands emerge and evolve
  - the resultant tempo of the ongoing development of systems of systems

If you're a software architect...so what?

If you think/know you're involved in a SoS collaboration,

- It is likely that the requirements you are working to do NOT account for sufficient demand-side variety
  - Don't over-constrain your software architecture too early
  - Look for architectural mechanisms that can accommodate later information on interfaces and implementations
- Try to find out the level of awareness of SoS issues that is present on the part of your systems engineers
  - The more they are aware of their lack of control over organizational and technical interactions across the collaboration, the less likely they will be to pass down over-constraining architecture requirements to the software
  - If awareness of SoS issues is low, find out how they are planning to deal with some of the demand-side constructs discussed here
- Start thinking about your customers' "operations architecture" – the components and interfaces that they are operating with and that you are supporting with your software
  - Look for points of complementarity and conflict between your software architecture and your customer's "operations architecture"
NO WARRANTY
THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this presentation is not intended in any way to infringe on the rights of the trademark holder.

This Presentation may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

Contact Information
Philip Boxer, Suzanne Garcia
Research, Technology and Systems Solutions Program,
Software Engineering Institute, Carnegie Mellon University

Email: pboxer@sei.cmu.edu, smg@sei.cmu.edu

Mail: Software Engineering Institute
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
USA