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

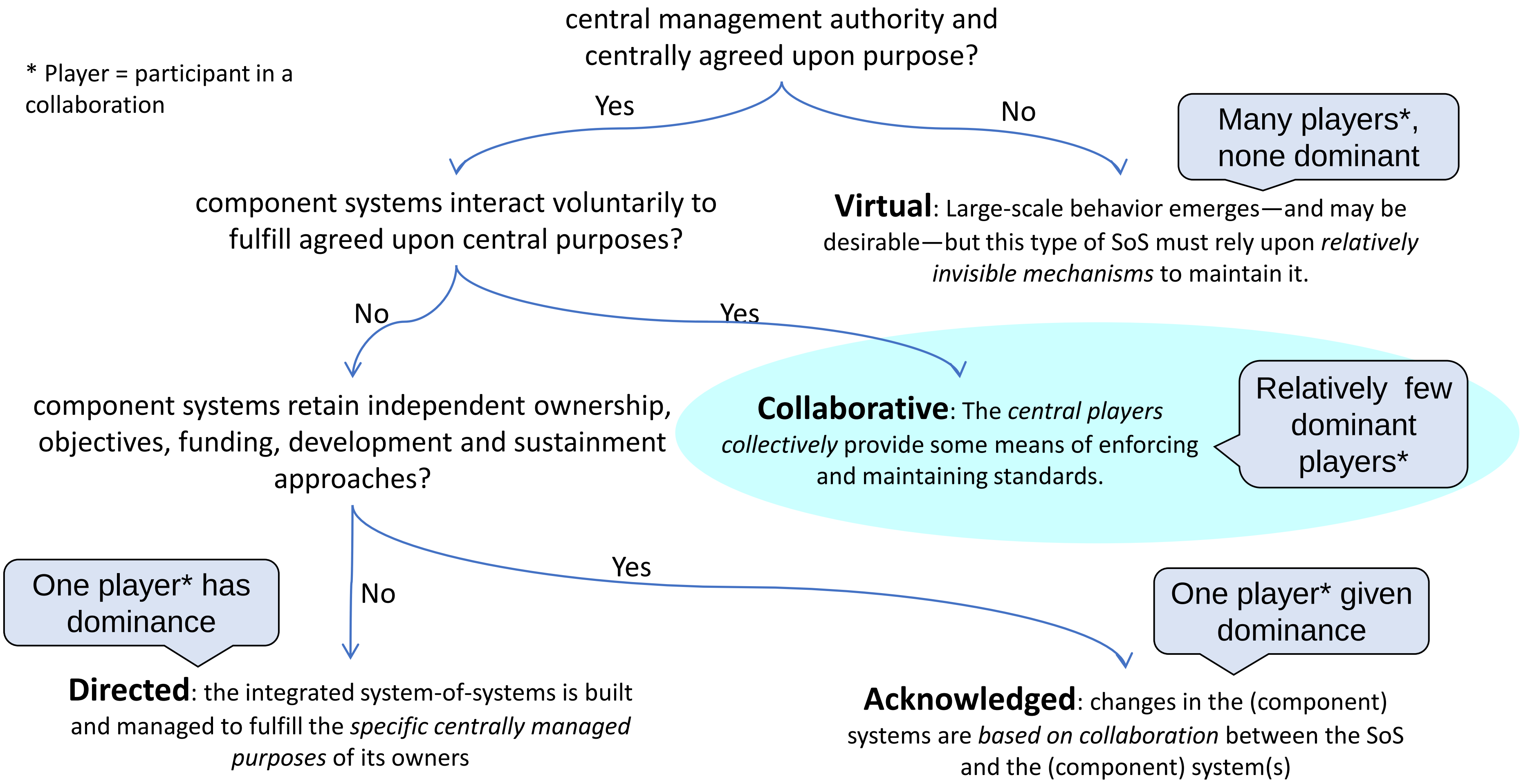
Philip Boxer

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

Systems of Systems: *4 Types defined by OSD SE Guide for Systems of Systems*



* Player = participant in a collaboration

central management authority and centrally agreed upon purpose?

Yes

No

Many players*, none dominant

Virtual: Large-scale behavior emerges—and may be desirable—but this type of SoS must rely upon *relatively invisible mechanisms* to maintain it.

component systems interact voluntarily to fulfill agreed upon central purposes?

No

Yes

Collaborative: The *central players collectively* provide some means of enforcing and maintaining standards.

Relatively few dominant players*

component systems retain independent ownership, objectives, funding, development and sustainment approaches?

No

Yes

One player* has dominance

Directed: the integrated system-of-systems is built and managed to fulfill the *specific centrally managed purposes* of its owners

One player* given dominance

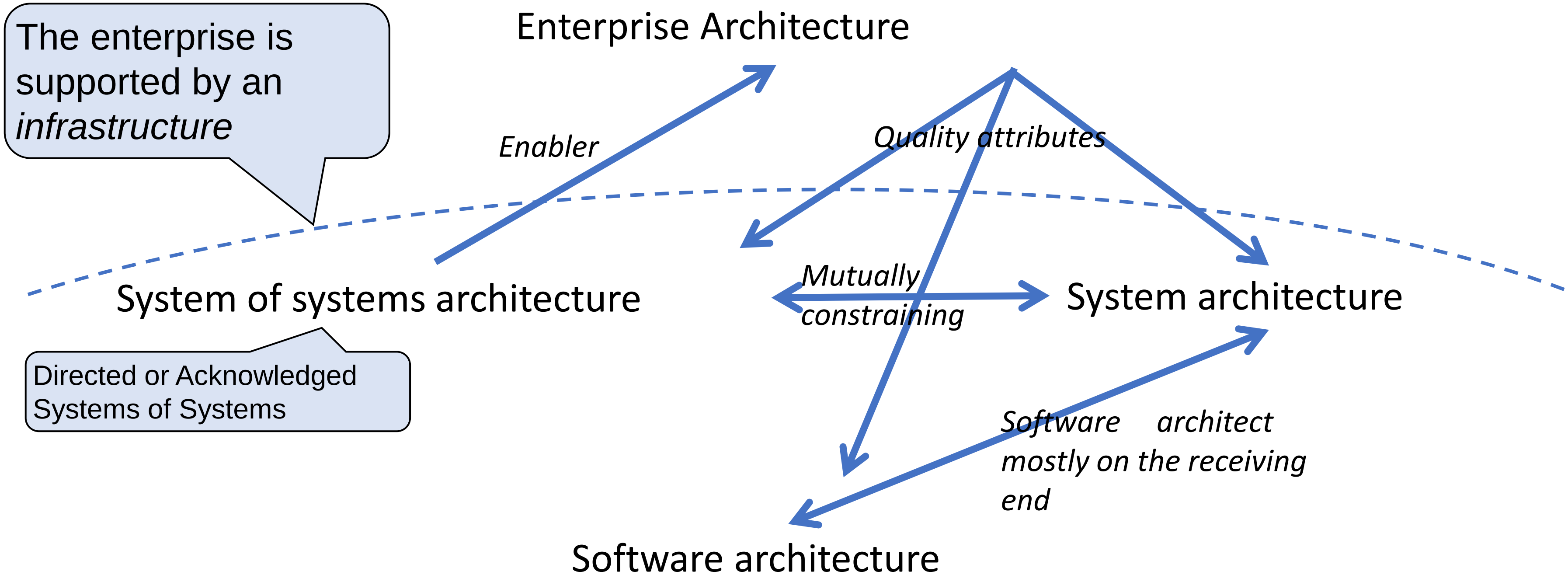
Acknowledged: changes in the (component) systems are *based on collaboration* between the SoS and the (component) system(s)

Source of definitions: Systems Engineering Guide for Systems of Systems, OSD, Version 1.0 August 2008. Brackets added.

Architectural Genres:

different genres for different purposes

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



The enterprise is supported by an *infrastructure*

Enterprise Architecture

Enabler

Quality attributes

System of systems architecture

Mutually constraining

System architecture

Directed or Acknowledged Systems of Systems

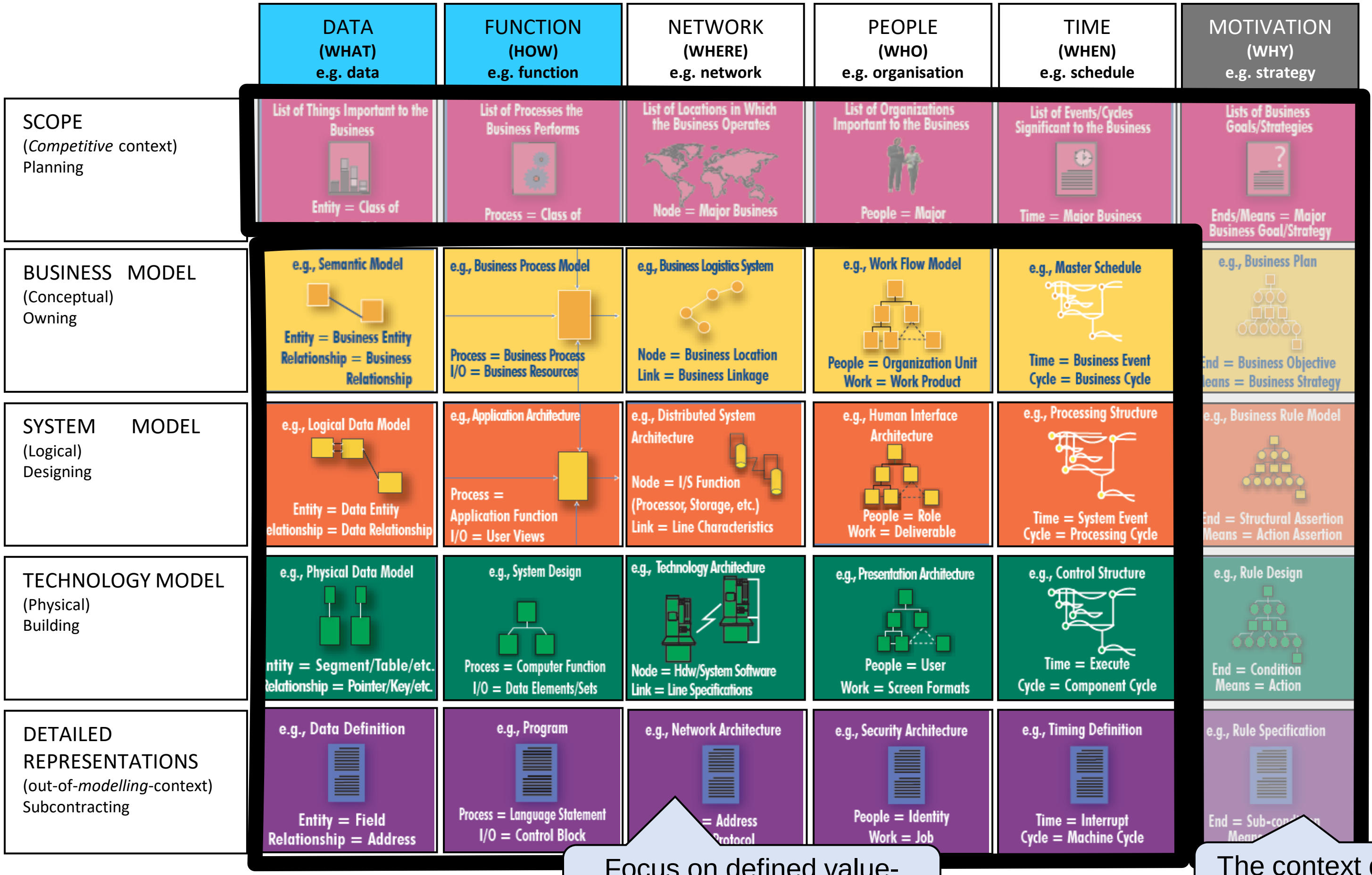
Software architect mostly on the receiving end

Software architecture

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

Source: U.S. Army Workshop on Exploring Enterprise, System of Systems, System, and Software Architectures, John Bergey, Stephen Blanchette, Jr., Paul Clements, Mike Gagliardi, John Klein, Rob Wojcik, Bill Wood, March 2009 TECHNICAL REPORT CMU/SEI-2009-TR-008

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



	DATA (WHAT) e.g. data	FUNCTION (HOW) e.g. function	NETWORK (WHERE) e.g. network	PEOPLE (WHO) e.g. organisation	TIME (WHEN) e.g. schedule	MOTIVATION (WHY) e.g. strategy
SCOPE (*Competitive* context) Planning	List of Things Important to the Business Entity = Class of	List of Processes the Business Performs Process = Class of	List of Locations in Which the Business Operates Node = Major Business	List of Organizations Important to the Business People = Major	List of Events/Cycles Significant to the Business Time = Major Business	Lists of Business Goals/Strategies Ends/Means = Major Business Goal/Strategy
BUSINESS MODEL (Conceptual) Owning	e.g., Semantic Model Entity = Business Entity Relationship = Business Relationship	e.g., Business Process Model Process = Business Process I/O = Business Resources	e.g., Business Logistics System Node = Business Location Link = Business Linkage	e.g., Work Flow Model People = Organization Unit Work = Work Product	e.g., Master Schedule Time = Business Event Cycle = Business Cycle	e.g., Business Plan End = Business Objective leans = Business Strategy
SYSTEM MODEL (Logical) Designing	e.g., Logical Data Model Entity = Data Entity elationship = Data Relationship	e.g., Application Architecture Process = Application Function I/O = User Views	e.g., Distributed System Architecture Node = I/S Function (Processor, Storage, etc.) Link = Line Characteristics	e.g., Human Interface Architecture People = Role Work = Deliverable	e.g., Processing Structure Time = System Event Cycle = Processing Cycle	e.g., Business Rule Model End = Structural Assertion Means = Action Assertion
TECHNOLOGY MODEL (Physical) Building	e.g., Physical Data Model ntity = Segment/Table/etc. Relationship = Pointer/Key/etc.	e.g., System Design Process = Computer Function I/O = Data Elements/Sets	e.g., Technology Architecture Node = Hdw/System Software Link = Line Specifications	e.g., Presentation Architecture People = User Work = Screen Formats	e.g., Control Structure Time = Execute Cycle = Component Cycle	e.g., Rule Design End = Condition Means = Action
DETAILED REPRESENTATIONS (out-of-*modelling*-context) Subcontracting	e.g., Data Definition Entity = Field Relationship = Address	e.g., Program Process = Language Statement I/O = Control Block	e.g., Network Architecture = Address Protocol	e.g., Security Architecture People = Identity Work = Job	e.g., Timing Definition Time = Interrupt Cycle = Machine Cycle	e.g., Rule Specification End = Sub-con Mean

Focus on defined value-creating relationships

The context defining that focus is the Enterprise

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

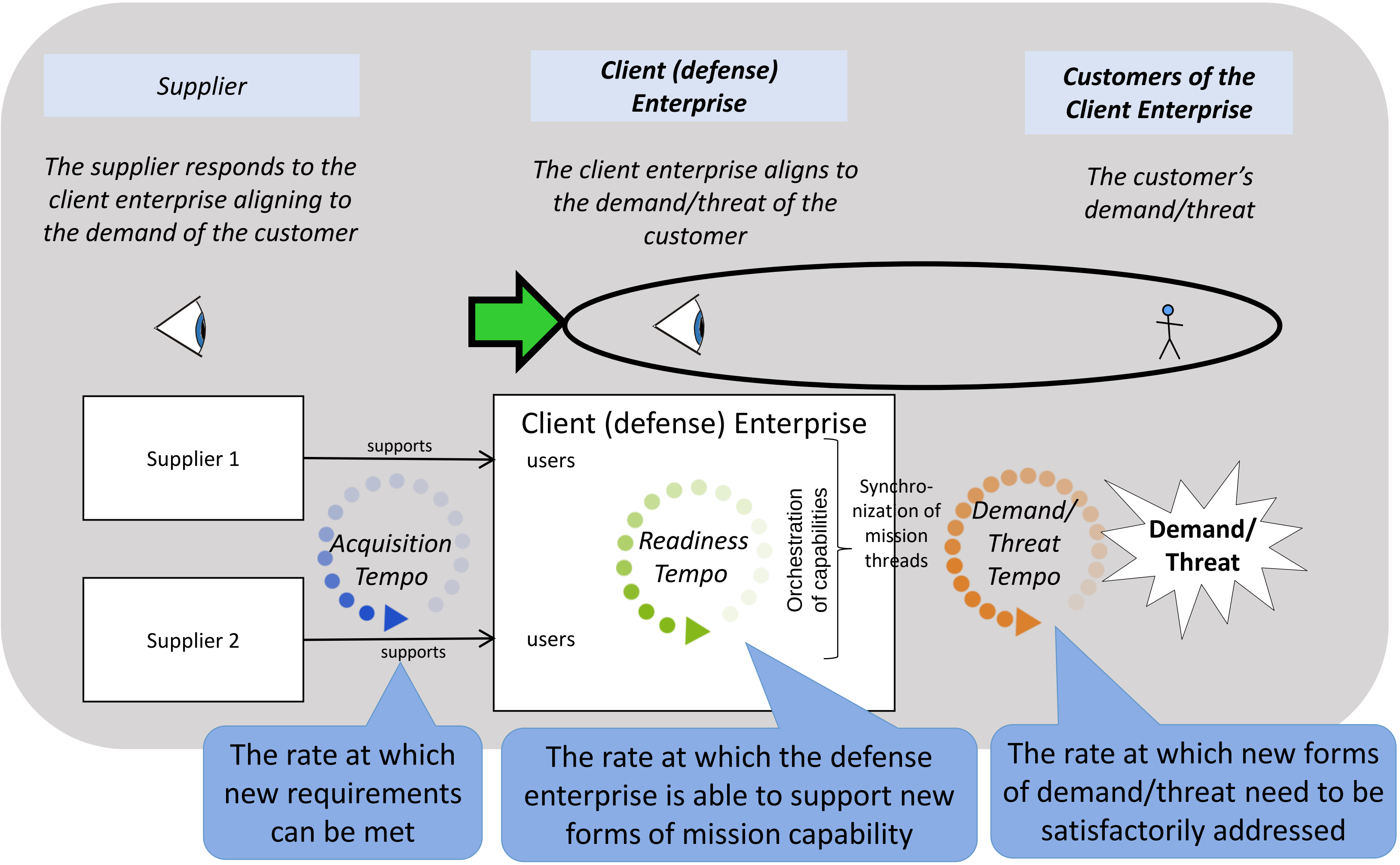
Speech by Secretary Gates:

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.

Extracted from speech delivered by Secretary of Defense Robert M. Gates,
National Defense University, Washington, D.C. September 29, 2008
http://www.defenselink.mil/speeches/speech.aspx?speechid=1279

The three tempos: *analyzing the impact of the enterprise's relation to customers' changing demands*



Supplier

The supplier responds to the client enterprise aligning to the demand of the customer

Client (defense) Enterprise

The client enterprise aligns to the demand/threat of the customer

Customers of the Client Enterprise

The customer's demand/threat

The rate at which new requirements can be met

The rate at which the defense enterprise is able to support new forms of mission capability

The rate at which new forms of demand/threat need to be satisfactorily addressed

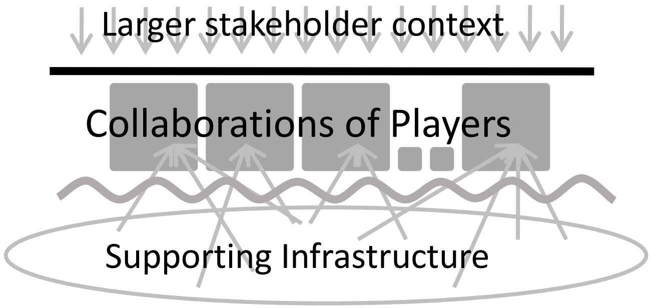
Managing diverging tempos: *the readiness tempo has to be managed in its own right*

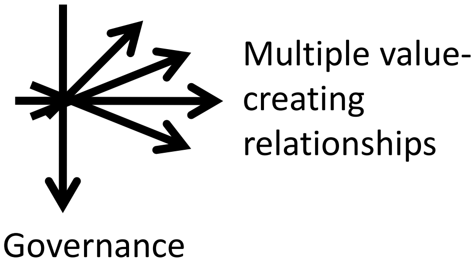
- Needing both of Gates' paradigms is about dealing with the consequences of diverging acquisition and demand/threat tempos
  - Their coexistence depends on managing the readiness tempo in its own right

- 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



It is the players participating in a particular collaboration who will define

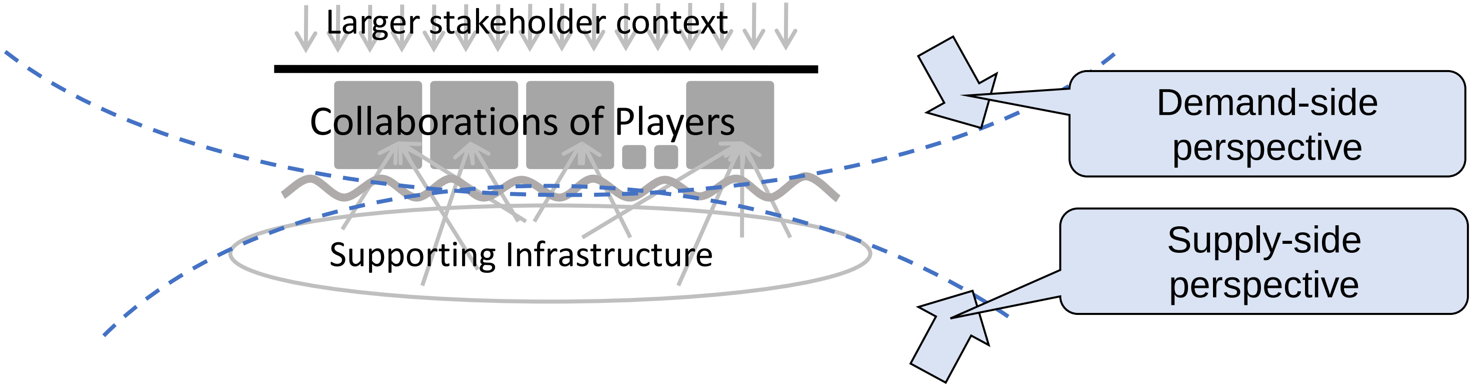
- Their system-of-interest and its environment
- The stakeholders they judge to be relevant
- The way they want their collaboration supported by the infrastructure

And so… *a demand-side perspective needs to be added*

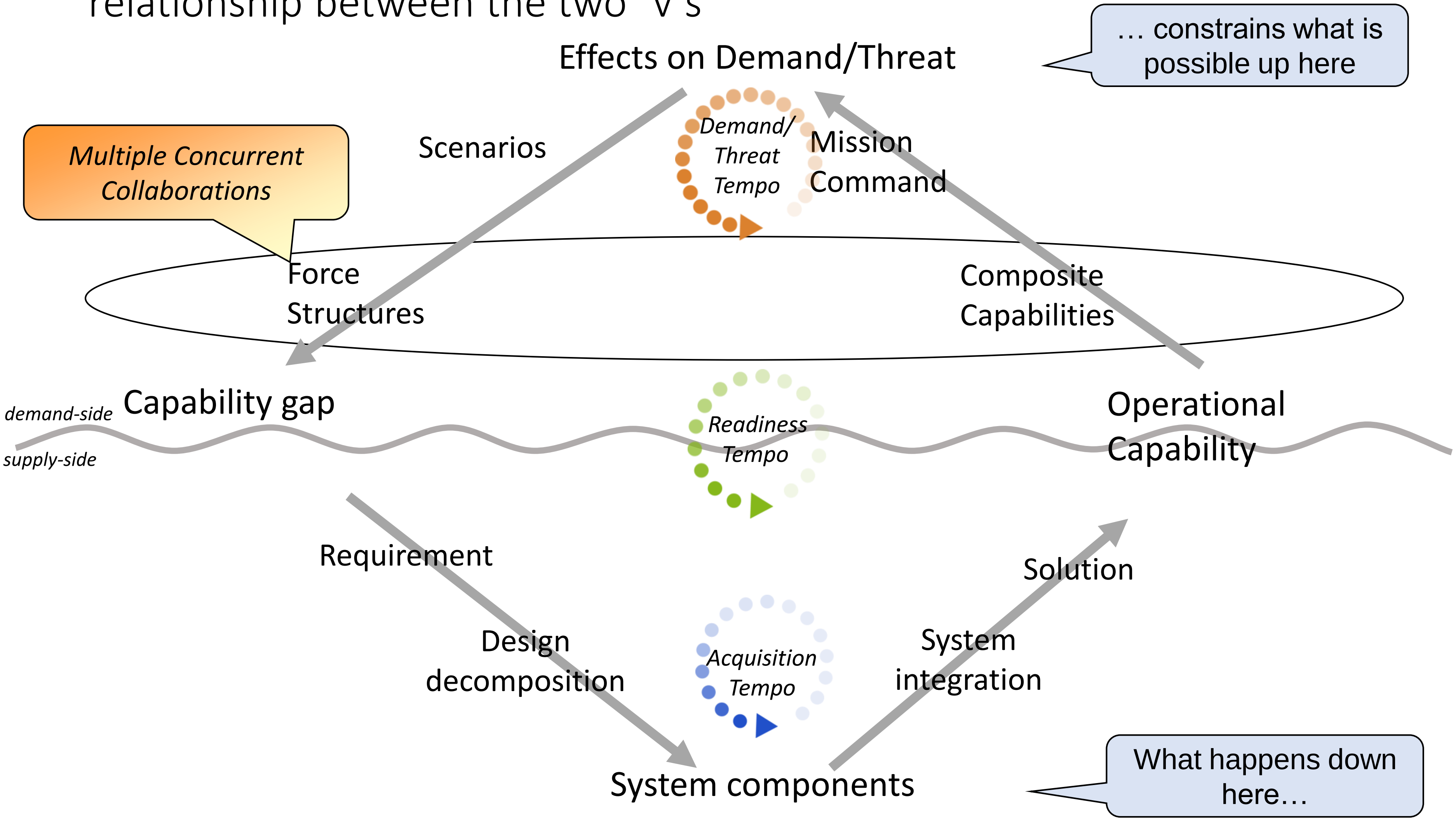
- Collaborative SoS present a different order of complexity

- 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


Larger stakeholder context
Collaborations of Players
Supporting Infrastructure
Demand-side perspective
Supply-side perspective

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



… constrains what is possible up here

Effects on Demand/Threat

Multiple Concurrent Collaborations

Scenarios

Demand/ Threat Tempo

Mission Command

Force Structures

Composite Capabilities

demand-side

Capability gap

supply-side

Readiness Tempo

Operational Capability

Requirement

Solution

Design decomposition

Acquisition Tempo

System integration

System components

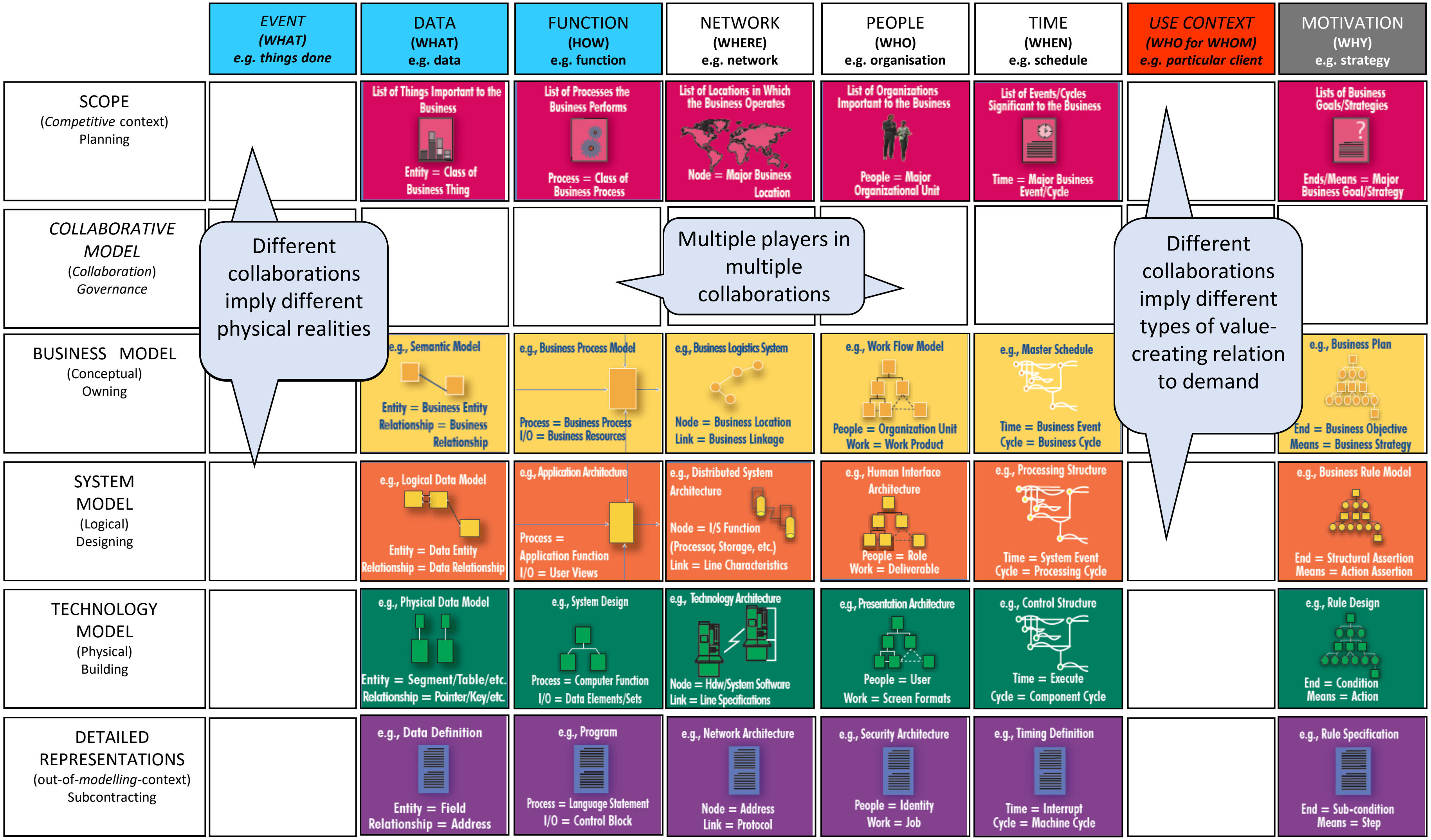
What happens down here…

Boxer, P.J. (2007) Managing the SoS Value Cycle, January 2007, http://www.asymmetricdesign.com/archives/85

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*



	EVENT ***(WHAT)*** ***e.g. things done***	DATA **(WHAT)** **e.g. data**	FUNCTION **(HOW)** **e.g. function**	NETWORK **(WHERE)** **e.g. network**	PEOPLE **(WHO)** **e.g. organisation**	TIME **(WHEN)** **e.g. schedule**	*USE CONTEXT* ***(WHO for WHOM)*** ***e.g. particular client***	MOTIVATION **(WHY)** **e.g. strategy**
SCOPE (*Competitive* context) Planning		List of Things Important to the Business Entity = Class of Business Thing	List of Processes the Business Performs Process = Class of Business Process	List of Locations in Which the Business Operates Node = Major Business Location	List of Organizations Important to the Business People = Major Organizational Unit	List of Events/Cycles Significant to the Business Time = Major Business Event/Cycle		Lists of Business Goals/Strategies Ends/Means = Major Business Goal/Strategy
COLLABORATIVE MODEL (*Collaboration*) *Governance*								
BUSINESS MODEL (Conceptual) Owning		e.g., Semantic Model Entity = Business Entity Relationship = Business Relationship	e.g., Business Process Model Process = Business Process I/O = Business Resources	e.g., Business Logistics System Node = Business Location Link = Business Linkage	e.g., Work Flow Model People = Organization Unit Work = Work Product	e.g., Master Schedule Time = Business Event Cycle = Business Cycle		e.g., Business Plan End = Business Objective Means = Business Strategy
SYSTEM MODEL (Logical) Designing		e.g., Logical Data Model Entity = Data Entity Relationship = Data Relationship	e.g., Application Architecture Process = Application Function I/O = User Views	e.g., Distributed System Architecture Node = I/S Function (Processor, Storage, etc.) Link = Line Characteristics	e.g., Human Interface Architecture People = Role Work = Deliverable	e.g., Processing Structure Time = System Event Cycle = Processing Cycle		e.g., Business Rule Model End = Structural Assertion Means = Action Assertion
TECHNOLOGY MODEL (Physical) Building		e.g., Physical Data Model Entity = Segment/Table/etc. Relationship = Pointer/Key/etc.	e.g., System Design Process = Computer Function I/O = Data Elements/Sets	e.g., Technology Architecture Node = Hdw/System Software Link = Line Specifications	e.g., Presentation Architecture People = User Work = Screen Formats	e.g., Control Structure Time = Execute Cycle = Component Cycle		e.g., Rule Design End = Condition Means = Action
DETAILED REPRESENTATIONS (out-of-*modelling*-context) Subcontracting		e.g., Data Definition Entity = Field Relationship = Address	e.g., Program Process = Language Statement I/O = Control Block	e.g., Network Architecture Node = Address Link = Protocol	e.g., Security Architecture People = Identity Work = Job	e.g., Timing Definition Time = Interrupt Cycle = Machine Cycle		e.g., Rule Specification End = Sub-condition Means = Step

Different collaborations imply different physical realities

Multiple players in multiple collaborations

Different collaborations imply different types of value-creating relation to demand

Source of gaps: Philip Boxer, Modeling structure-determining processes, http://www.asymmetricdesign.com/archives/59, December 2006

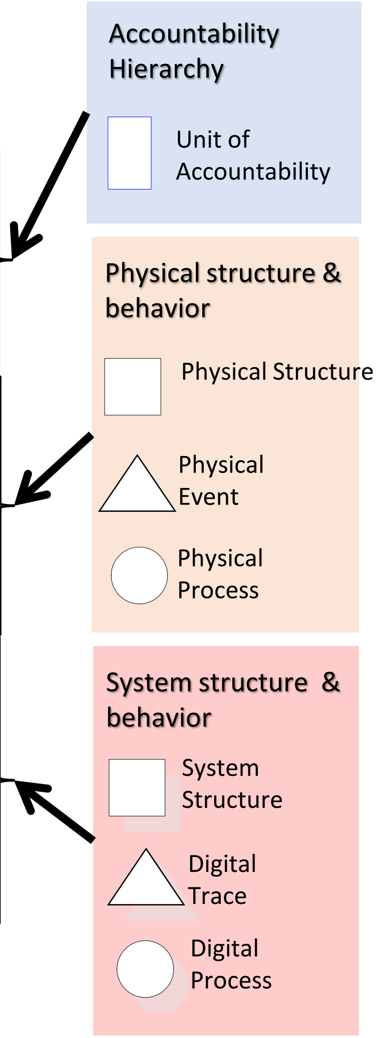
DODAF 2.0 Entities and Views:

what gets modeled?

Modeling Elements

DODAF TAXONOMY TYPES and CADM 2.0 PRINCIPAL INDEPENDENT ENTITIES	All Views (AV)		Operational View (OV)							System View (SV)											Tech View(TV)	
	1	{2}	1	{2}	{3}	{4}	{5}	{6}	{7}	{1}	{2}	{3}	{4}	{5}	{6}	{7}	{8}	{9}	{10}	{11}	{1}	{2}
Operational Nodes *Organizations, Types of Organizations, and Operational Roles*	🗎	●	🗎	●	●	●	●	●														
Performance Attributes	🗎	●	🗎		◉		◉	◉					◉		●	●	◉	◉	◉		◉	◉
Technical Standards *Info Processing, Info Transfer, Data, Security, and Human Factors*	🗎	●	🗎						◉	◉	◉		◉		◉		◉	◉		◉	●	●
Technology Areas	🗎	●	🗎															●				◉
Physical Nodes *Facilities, Platforms, Units, and Locations (including Features)*	🗎	●	🗎							◉	◉	◉			◉	◉	◉	◉			◉	◉
Triggers/Events	🗎	●	🗎		●			●							●				●			
Operational Activities *(and Tasks)*	🗎	●	🗎	◉	●		●	●						●								
Technology Areas	🗎	●	🗎															●				◉
Systems *Families-of-Systems, Systems-of-Systems, Networks, Applications, Software, and Equipment*	🗎	●	🗎							●	●	●	●	◉	●	●	●	●	●		◉	◉
Information Elements *(and Data Elements)*	🗎	●	🗎	◉	●		●		●				●		●				◉	●	◉	◉
System Functions	🗎	●	🗎										●	●	◉	◉	◉	◉	●		◉	◉

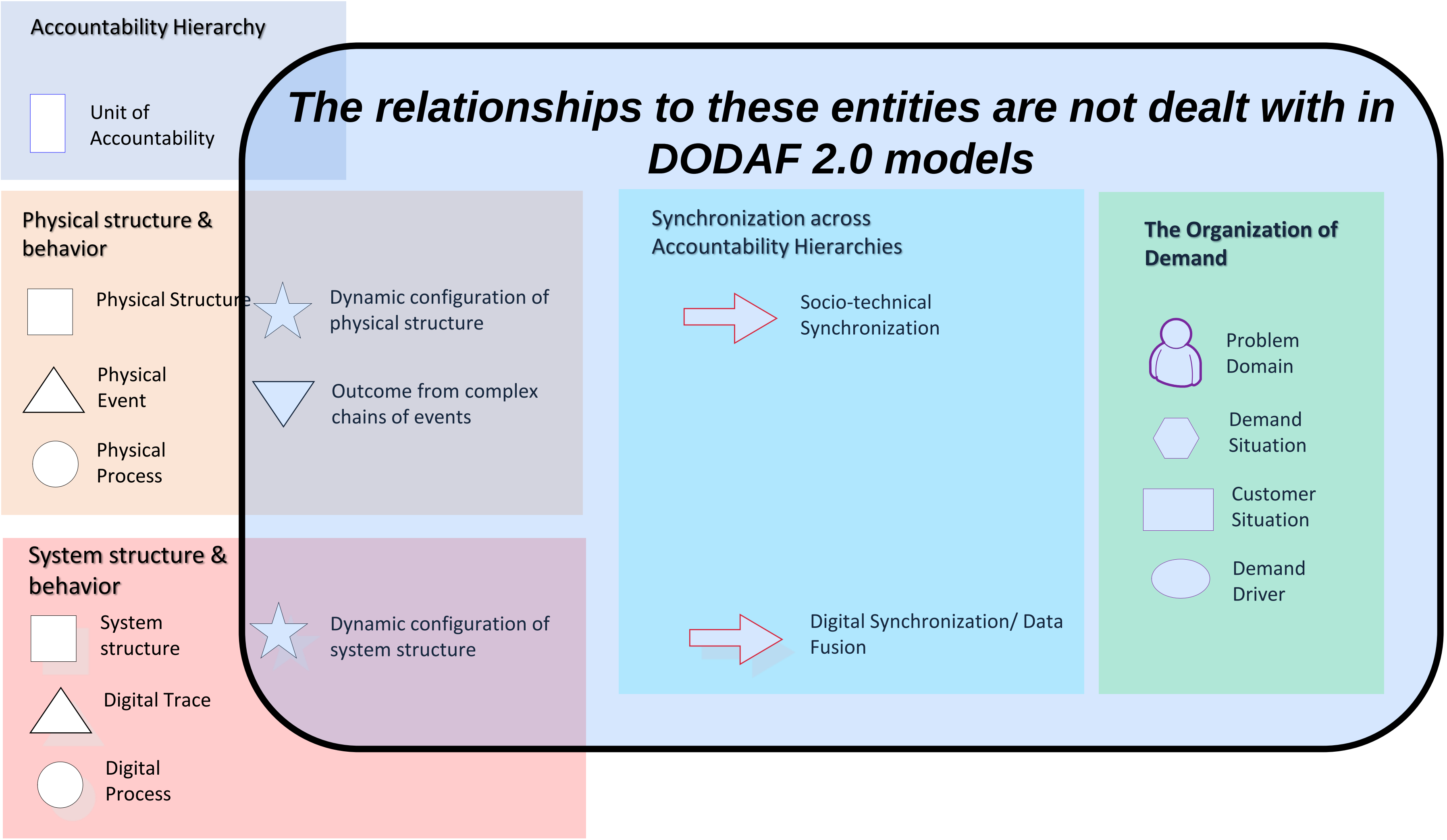
● = Taxonomy element plays a primary role   ◉ = element plays a secondary role   🗎 = unstructured text or graphics



Source: Fig 3-2, DoD Architectural Framework version 2.0 Volume III: Architecture Data Description, DOD Architecture Framework Working Group, July 2006

Entities not modeled by DODAF 2.0:

the demand-side perspective is not included



Accountability Hierarchy

Unit of Accountability

The relationships to these entities are not dealt with in DODAF 2.0 models

Physical structure & behavior

Physical Structure

Physical Event

Physical Process

Dynamic configuration of physical structure

Outcome from complex chains of events

System structure & behavior

System structure

Digital Trace

Digital Process

Dynamic configuration of system structure

Synchronization across Accountability Hierarchies

Socio-technical Synchronization

Digital Synchronization/ Data Fusion

The Organization of Demand

Problem Domain

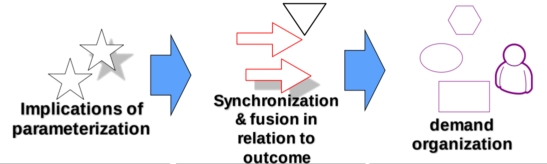
Demand Situation

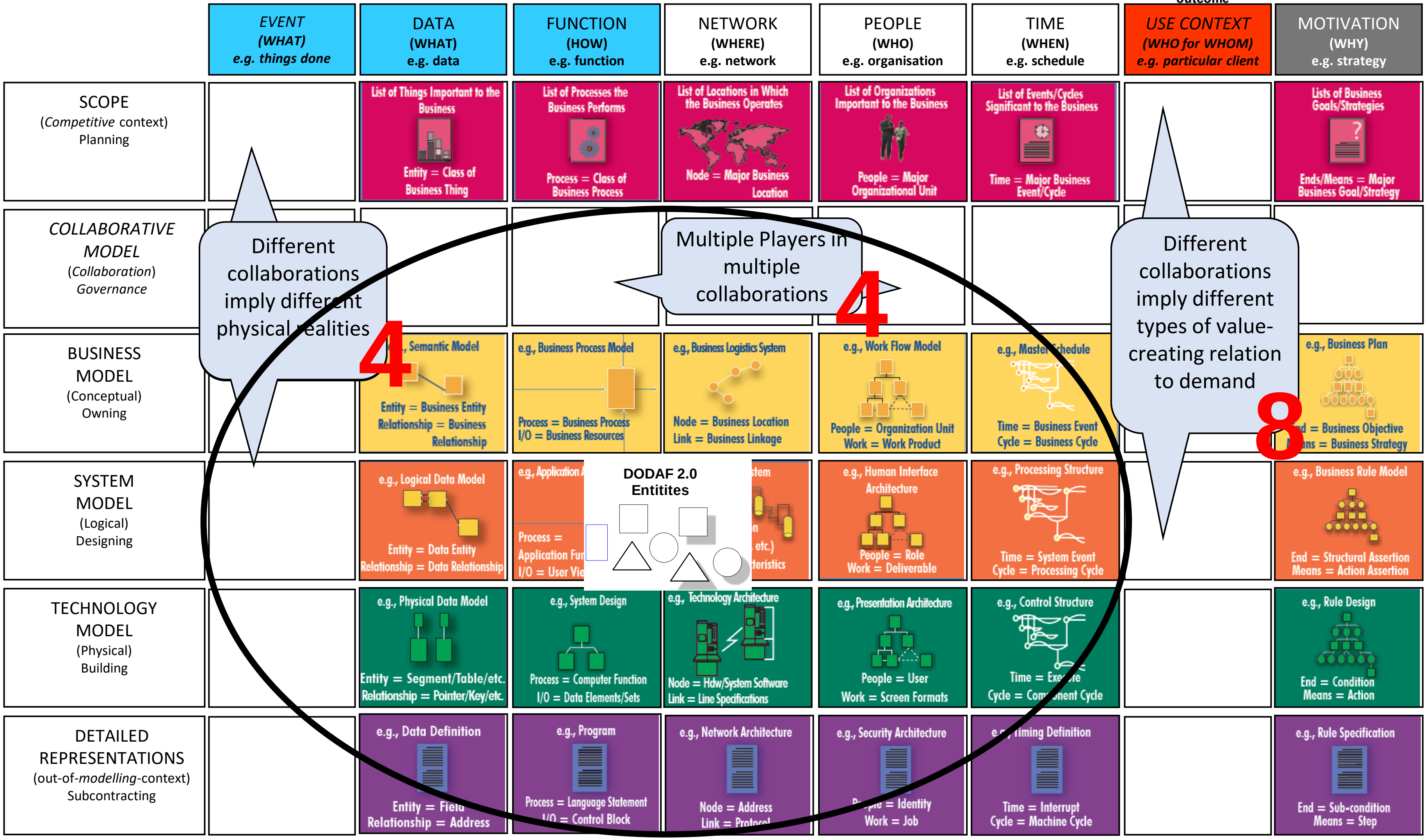
Customer Situation

Demand Driver

Describing the demand-side:

bridging the gaps


Implications of parameterization
Synchronization & fusion in relation to outcome
demand organization



	EVENT ***(WHAT)*** ***e.g. things done***	DATA **(WHAT)** **e.g. data**	FUNCTION **(HOW)** **e.g. function**	NETWORK **(WHERE)** **e.g. network**	PEOPLE **(WHO)** **e.g. organisation**	TIME **(WHEN)** **e.g. schedule**	*USE CONTEXT* ***(WHO for WHOM)*** ***e.g. particular client***	MOTIVATION **(WHY)** **e.g. strategy**
SCOPE (*Competitive* context) Planning		List of Things Important to the Business Entity = Class of Business Thing	List of Processes the Business Performs Process = Class of Business Process	List of Locations in Which the Business Operates Node = Major Business Location	List of Organizations Important to the Business People = Major Organizational Unit	List of Events/Cycles Significant to the Business Time = Major Business Event/Cycle		Lists of Business Goals/Strategies Ends/Means = Major Business Goal/Strategy
COLLABORATIVE MODEL (*Collaboration*) *Governance*								
BUSINESS MODEL (Conceptual) Owning		e.g., Semantic Model Entity = Business Entity Relationship = Business Relationship	e.g., Business Process Model Process = Business Process I/O = Business Resources	e.g., Business Logistics System Node = Business Location Link = Business Linkage	e.g., Work Flow Model People = Organization Unit Work = Work Product	e.g., Master Schedule Time = Business Event Cycle = Business Cycle		e.g., Business Plan End = Business Objective Means = Business Strategy
SYSTEM MODEL (Logical) Designing		e.g., Logical Data Model Entity = Data Entity Relationship = Data Relationship	e.g., Application A Process = Application Fur I/O = User Vie	stem n etc.) teristics	e.g., Human Interface Architecture People = Role Work = Deliverable	e.g., Processing Structure Time = System Event Cycle = Processing Cycle		e.g., Business Rule Model End = Structural Assertion Means = Action Assertion
TECHNOLOGY MODEL (Physical) Building		e.g., Physical Data Model Entity = Segment/Table/etc. Relationship = Pointer/Key/etc.	e.g., System Design Process = Computer Function I/O = Data Elements/Sets	e.g., Technology Architecture Node = Hdw/System Software Link = Line Specifications	e.g., Presentation Architecture People = User Work = Screen Formats	e.g., Control Structure Time = Execute Cycle = Component Cycle		e.g., Rule Design End = Condition Means = Action
DETAILED REPRESENTATIONS (out-of-*modelling*-context) Subcontracting		e.g., Data Definition Entity = Field Relationship = Address	e.g., Program Process = Language Statement I/O = Control Block	e.g., Network Architecture Node = Address Link = Protocol	e.g., Security Architecture People = Identity Work = Job	e.g., Timing Definition Time = Interrupt Cycle = Machine Cycle		e.g., Rule Specification End = Sub-condition Means = Step

Different collaborations imply different physical realities

4

Multiple Players in multiple collaborations

4

Different collaborations imply different types of value-creating relation to demand

8

DODAF 2.0 Entitites

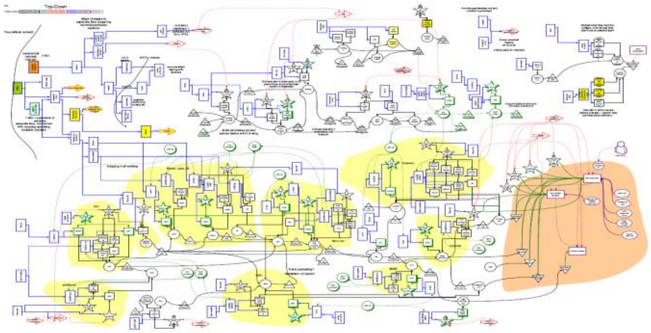
Source of gaps: Philip Boxer, Modeling structure-determining processes, http://www.asymmetricdesign.com/archives/59, December 2006

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

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”