
Re-Thinking Architecture

Leonard Fehskens
VP, Skills and Capabilities
The Open Group
l.fehskens@opengroup.org

“The most essential and fundamental principle of our work is the absolute authenticity of any of its employees’ qualifications and credentials. Each must be tops in his field, otherwise, well otherwise it’s mere quackery.”

“Just what is our work Dad?”

“It’s not so easy to define as one might think, lad. Not so easy.”

Sir Guy Grand (Peter Sellers)
and Youngman Grand (Ringo Starr)
in ***The Magic Christian***
Screenplay by Terry Southern

Agenda

- ❑ Introduction
- ❑ Brief History of the Idea of “Our Kind of” Architecture
- ❑ A Name for “Our Kind of” Architecture
- ❑ Why Do We Do Architecture?
- ❑ Towards a Good Definition
- ❑ Considering IEEE 1471 / ISO 42010
- ❑ (Re)Defining Architecture
- ❑ Conclusions

Agenda

- Introduction
- Brief History of the Idea of “Our Kind of” Architecture
- A Name for “Our Kind of” Architecture
- Why Do We Do Architecture?
- Towards a Good Definition
- Considering IEEE 1471 / ISO 42010
- (Re)Defining Architecture
- Conclusions

My Background

- ❑ MIT, 1964 – 1968, EE specializing in computer science
- ❑ Applied Data Research 1969 - 1973
- ❑ Digital Equipment Corporation 1973 - 1976
- ❑ Data General 1976 - 1978
- ❑ Prime Computer 1978 - 1985
- ❑ Digital/Compaq/HP 1985 – 2007
- ❑ The Open Group

Influences on my Thinking

- ❑ My experience in simulation and development methods
- ❑ System/360 architects' notion of architecture
- ❑ Digital's notion of architecture
- ❑ Compaq/HP's notion of solution architecture
- ❑ The Digital/Compaq/HP architecture methodology
- ❑ The question of how architecture differs from design
- ❑ The desire to professionalize the discipline
- ❑ The desire to meaningfully extend the value proposition for architecture into realms beyond IT
- ❑ My experience as an ITAC board member reviewing architecture and architecture methods as practiced.
- ❑ Conversations with members of The Open Group Architecture Forum and Business Architecture Working Group

“Truth in Presenting” Disclosure

- ❑ This talk will seem like a lot of abstract blather unless you’ve been “reflectively practicing” enterprise architecture for some time.
- ❑ Much of this material is “unconventional”.
 - Not to worry; you’ll hear more than enough of the conventional wisdom on this subject everywhere else.
 - This discipline is young enough that the idea that we already have all the right answers is presumptuous at best.
- ❑ Try to remain open-minded; draw your own conclusions about whether this alternative perspective helps you better understand the discipline.
- ❑ My thinking on this subject continues to evolve.

The Problem

- ❑ The problem is not that we don't have a definition of Enterprise Architecture;
- ❑ It's that we have too many definitions.

- ❑ It's hard to argue that Enterprise Architecture is a legitimate professional discipline if it means pretty much what anybody says it means.

Why Define Architecture?

- ❑ Is a precise definition of “our kind of” architecture really necessary?
 - Several thoughtful commentators have persuasively argued “no, just do it”.
- ❑ Until we have a more or less standard curriculum for training architects, I think it is.
- ❑ Otherwise, how can we:
 - Agree on why we do architecture?
 - Agree on what architects do?
 - Agree on what skills architects need?
 - Create a legitimate profession?
 - I.e., properly set expectations for the producers and consumers of architectural work products.

Why Rethink Architecture?

- The conventional wisdom is rapidly becoming that Enterprise Architecture is more than Enterprise IT Architecture.
 - There's a *lot* more to an enterprise than its IT; IT budgets generally represent about 2% of revenues.
 - An increasing number of enterprise architects believe that the rest of the enterprise, often generically referred to as “the business”, can and should be architected as well.
- To address the architectures of things outside the domain of IT, we need a concept of architecture that is not technological, and that is expressed in nontechnical language.
- Increasingly widespread acknowledgement of the importance and value of architecture is creating demand for a true profession.

Theory and Practice

- We need a theory of our kind of architecture to provide a sound foundation for a true profession.
- A theory of our kind of architecture would comprise:
 - What is architecture?
 - What is an enterprise?
 - What is enterprise architecture?
 - What is the value proposition for architecture?
 - How does architecture relate to “everything else”?
- “In theory, theory and practice are the same. In practice, they’re not.”

Methods

- ❑ For most practitioners, architecture is operationally defined as “what you get when you use whatever method you use to do architecture”.
- ❑ Most architecture methods can be categorized as:
 - Heuristics for producing architectural work products
 - Project management life cycle models for doing architecture work.
- ❑ The theory behind most methods is implicit at best.

The State of the Art – My Assessment

- ❑ Too many people think of architecture as something you're doing if you're really good at something.
 - “I'm really good at <x>, so I deserve to be called an ‘<x> architect’”.
- ❑ Architecture should be more than something people claim to be doing to justify a higher hourly rate.

The State of the Art – My Assessment

- ❑ Most definitions of “our kind of architecture” are just recycled definitions of design.
- ❑ There are so many and so varied definitions of “enterprise architecture” that there is a website devoted to collecting them.
- ❑ Virtually all of these definitions do not say what enterprise architecture is, they say what we hope enterprise architecture will do for us.
 - This is like defining a hammer as “something used to drive nails”.
 - While this is certainly correct in some sense, it is not very helpful in envisioning what a proper hammer looks like, and admits as a “hammer” all manner of devices which must be misused to drive nails.

The State of the Art – Other Assessments

- “In a sense, I define *architecture* as a word we use when we want to talk about design but want to puff it up to make it sound important.”
 - Martin Fowler, “Who Needs an Architect?”, *IEEE Software*, July/August 2003
- “In IT, we love titles and metaphors that come from the construction world. Of course, many of our ‘software engineers’ don’t know the first thing about engineering, and the quality of IT infrastructure that we casually call ‘plumbing’ would make a union plumber cringe. And architects? In IT, we think that means people who design systems.”
 - Frank Hayes, http://blogs.computerworld.com/frankly_speaking_real_architects

My Ultimate Goals

- ❑ Develop and justify a concept of architecture that clearly distinguishes it as a specific kind of design (in the most general, “Big D Design” sense), and that can be consistently and meaningfully applied across the entire domain of interest to us.
- ❑ Define this concept of architecture in language that is independent of the context within which architecture is applied, or the medium by which it is expressed, executed or implemented.
- ❑ Develop and justify a model of architecture in the enterprise context (i.e., “enterprise architecture”) that usefully applies to the entire enterprise, not just its IT assets, and to enterprises that are not “businesses”.

Where We're Going

- The Generalized Amdahl Blaauw Brooks (GABB) definition of architecture:
 - Those properties of a mission, its solution and their environment that are necessary and sufficient for the solution to be fit for purpose for its mission in that environment.
- The GABB applied to enterprise architecture:
 - Those properties of an enterprise, its mission, and their environment, that are necessary and sufficient for the enterprise to be fit for purpose for its mission in that environment, so as to ensure continuous alignment of the enterprise's assets and capabilities with its mission and strategy.

First Principles

- ❑ The meaning of “architecture” in “<x> architecture” should be independent of <x>, and the same as the definition of (unqualified) “architecture”.
- ❑ The definition of “<x> architecture” ought to be a straightforward elaboration of “the architecture of an <x>” or “architecture in the <x> context”.
- ❑ When this is not the case, I get concerned.
- ❑ However, I should note that some people assert it is neither possible nor desirable to satisfy these principles.

Some Fables to Keep in Mind

- The blind men and the elephant
 - Be careful about defining something as the part of it you are most familiar with.
- The “excellent” tailor
 - If you have to twist an idea into a pretzel to get it to apply to something it ought to apply to, maybe it’s the wrong idea.

And Remember...

- The specific words in a definition are important only to the extent that they effectively represent the concept of the definition.

Three ways we use the words “architecture”

- As the name of a discipline or practice
 - “I took a course on enterprise architecture.”
- As applied to a class of things
 - “An enterprise architecture enables business transformation.”
- As applied to a specific instance of a thing
 - “We used TOGAF as the basis for our enterprise architecture.”

- The meaning of architecture ought to be consistent across these three usages.

Agenda

- Introduction
- *Brief History of the Idea of “Our Kind of” Architecture*
- A Name for “Our Kind of” Architecture
- Why Do We Do Architecture?
- Towards a Good Definition
- Considering IEEE 1471 / ISO 42010
- (Re)Defining Architecture
- Conclusions

A Review of the Concept of Enterprise Architecture Up To Now

- ❑ Earliest Reference to Architecture in IT
- ❑ Software Architecture
- ❑ Architecture at Digital
- ❑ A Framework for IS Architecture
- ❑ Enterprise Architecture

The Roots of Enterprise Architecture: Processor Architecture

- ❑ Probable first use of “architecture” in an IT context in Amdahl, Blaauw, and Brooks “*Architecture of the IBM System/360*” (IBM Journal of Research and Development, April 1964).
 - “The term *architecture* is used here to describe the attributes of a system as seen by the programmer, i.e., the conceptual structure and functional behavior, as distinct from the organization of the data flow and controls, the logical design, and the physical implementation.”
- ❑ Their intent was to define the criteria for membership in a class of artifacts, i.e., System/360 processors, over a performance range of 50:1.

The Roots of Enterprise Architecture: Software Architecture

- Subsequently, in 1968, Edsger Dijkstra used “architecture” to mean the overall structure of a program, and this idea was further developed by David Parnas *et alia*, as part of the developing discipline of software engineering.
- Software architecture has been the subject of considerable academic research, on the premise that the structure of programs largely determines their properties.
 - Poorly structured → undesired properties
 - Well structured → desired properties
- These properties are largely independent of the purpose of a program. They are properties of a program as an end in itself, rather than as a means to an end.
- As such, software architecture is less about ensuring that programs “do what they’re supposed to do” than it is about making “well-behaved” programs.

The “Mythical Man-Month” on Architecture

“By the *architecture* of a system, I mean the complete and detailed specification of the user interface. For a computer this is the programming manual. For a compiler it is the language manual. For a control program it is the manuals for the language or languages used to invoke its functions. For the entire system it is the union of the manuals the user must consult to do his entire job.”

Frederick P. Brooks, Jr.
“The Mythical Man-Month”
1975

Architecture at **digital**

- ❑ Digital Equipment Corporation was generally recognized as one of the most intensively architectural engineering cultures in the industry.
- ❑ At Digital, architecture almost always meant architecture in the System/360 sense. It only rarely meant architecture in the software architecture sense.
- ❑ This concept of architecture was applied to processor families (DECsystem-10, PDP-11, VAX, Alpha), networking (DNA, DECnet), middleware (AIA, NAS), manageability (EMA, DECMcc/TMIP), ...
- ❑ This concept of architecture was such an integral part of the culture that it was never formally defined. However, in the late '80s I retrospectively defined it as:
 - A set of guarantees satisfied by a conforming implementation.

The Roots of Enterprise Architecture: A Framework for IS Architecture

- First application of architecture to enterprise wide (IT) systems by John Zachman in “*A framework for information systems architecture*” (IBM Systems Journal, Vol. 26, No. 3, 1987)
 - Note that Zachman did not call it “enterprise architecture”, or “information technology architecture”, he called it “information systems architecture”
 - Followed up by “*Extending and formalizing the framework for information systems architecture*” (IBM Systems Journal, Vol. 31, No. 3, 1992)
 - Zachman did not explicitly define architecture, alluding to it as “some logical construct (or architecture) for defining and controlling the interfaces and the integration of all the components of the system”.
 - Most of the paper addresses the question of how to represent such an architecture of an information system.

John Zachman's Latest Words on Architecture

- ❑ “... a set of descriptive representations relevant for describing a complex object (actually any object) such that an instance of the object can be created and such that the descriptive representation serves as a baseline for changing an object instance (assuming that the descriptive representations are maintained consistent with the instance).”
- ❑ “If it’s not complex and changing, you don’t need architecture”
- ❑ In his talk, Zachman essentially says that architecture is really engineering design.

The Roots of Enterprise Architecture: Enterprise Architecture

- First actual use of “enterprise architecture” by Steven Spewak in Enterprise Architecture Planning: Developing a Blueprint for Data, Applications and Technology (Wiley, 1992)
 - Note that the subtitle limits the scope to “data, applications and technology”.
 - Defines “enterprise” as “the term *enterprise* should include *all areas that need to share **substantial** amounts of data*”.
 - Doesn’t define “architecture”, just says “Architectures, in this context, are like blueprints, drawings or models”.

The Roots of Enterprise Architecture: Summary

- ❑ The idea of architecture in the IT domain is almost 45 years old.
- ❑ The idea of enterprise architecture is about 20 years old.
- ❑ Thinking about architecture has diverged into two communities:
 - Architecture as conceived by system engineers.
 - Architecture is about specifying the things that ensure something is what it's supposed to be or does what it's supposed to do.
 - Architecture as conceived by software academics.
 - Architecture is about internal structure.

The Roots of Enterprise Architecture: Two Distinct Worldviews

- ❑ Most of “our kind of” architects came out of the IT community, and most of them started as programmers.
- ❑ Because the software community is so much larger than the systems engineering community, thinking about architecture has come to be dominated by the software perspective. But:
 - The software concept of architecture has never been clearly distinguished from “general”, “high level”, “abstract” or “logical” internal design.
 - One might reasonably ask how ideas about the internal structure of programs apply to challenges like business transformation.

Reconciling Two Worldviews

- ❑ Is it possible to reconcile these two different concepts of architecture?
- ❑ I believe it is, by starting from and generalizing the System/360 architects' concept of architecture.
- ❑ Again, if you don't agree that this is a worthwhile or achievable goal, what follows will just be so much foolishness.

Recap and the Way Forward

- ❑ Most current definitions of architecture derived from the software concept of architecture are basically dressed up definitions of internal design.
- ❑ These definitions of architecture typically focus on what architecture should (or must) say about something; i.e., they define architecture in terms of its range (in the algebraic sense), or how it is represented.
- ❑ I have become convinced that architecture is better defined in terms of its domain, i.e., what things architecture should (or must) say something about.

It doesn't matter what you say about something if you say it about the wrong thing.

Agenda

- ❑ Introduction
- ❑ Brief History of the Idea of “Our Kind of” Architecture
- ❑ *A Name for “Our Kind of” Architecture*
- ❑ Why Do We Do Architecture?
- ❑ Towards a Good Definition
- ❑ Considering IEEE 1471 / ISO 42010
- ❑ (Re)Defining Architecture
- ❑ Conclusions

Variations on the Theme of “Our Kind of” Architecture

Chip Architecture

Security Architecture

Hardware Architecture

Management Architecture

Processor Architecture

Information Architecture

System Architecture

Software Architecture

Platform Architecture

Application Architecture

Storage Architecture

Solution Architecture

Network Architecture

Enterprise Architecture

Infrastructure Architecture

Business Architecture

A Name for “Our Kind of” Architecture

- ❑ “Our kind of” architecture needs a name for the discipline as a whole. We can’t just call it “architecture” because that is generally taken to mean civil architecture, addressing the architecture of buildings.
- ❑ Calling it “IT Architecture” or “IS Architecture” unduly circumscribes its applicability.
- ❑ The common thread to all of these subdomains within “our kind of” architecture is that they ultimately address the concerns of human enterprise, i.e., collaboration to achieve some shared goal.
- ❑ Hence, the discipline of “our kind of” architecture is legitimately called *enterprise architecture*.

Disambiguating “Enterprise Architecture”

- This creates some ambiguity:
 - Today most people use EA to mean the architecture of an enterprise’s IT assets.
 - We should use it to mean the architecture of the enterprise as a whole.
 - If we adopt this name for the discipline we can also mean the union of all the different architectural specializations of interest to an enterprise.
- Many professional words have similar ambiguity:
 - “medicine”
 - “law”

Disambiguating “Enterprise Architecture”

- The name of the discipline is enterprise architecture in the sense of “architecture in the enterprise context”.
- An area of specialization within the discipline is enterprise architecture in the sense of being about “the architecture of an enterprise”.

Agenda

- ❑ Introduction
- ❑ Brief History of the Idea of “Our Kind of” Architecture
- ❑ A Name for “Our Kind of” Architecture
- ❑ *Why Do We Do Architecture?*
- ❑ Towards a Good Definition
- ❑ Considering IEEE 1471 / ISO 42010
- ❑ (Re)Defining Architecture
- ❑ Conclusions

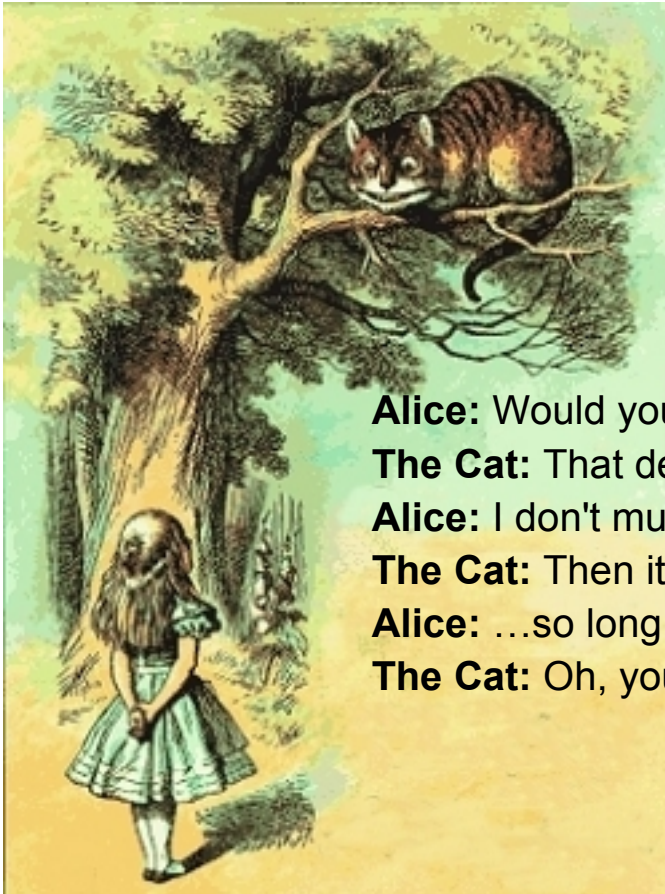
Why Do We Do Architecture?

- The conventional wisdom(s):
 - To ensure all the pieces work together, by looking at the “big picture”, and thus ensure structural integrity.
 - To provide “conceptual integrity”.
 - To address “nonfunctional requirements”.
 - To recognize and reuse patterns and styles that are acknowledged to represent best practices.
 - To define a (software) product family.
 - To master complexity.
 - To facilitate change via adaptability and agility.
 - To facilitate “business transformation”.

Why Do We Do Architecture?

- What I believe to be the real, primary reason:
 - Alignment - to ensure that something is what it's supposed to be, and does what it's supposed to do.
- Without alignment, achieving any other goals will not deliver the desired value.

The Cheshire Cat on Change



Alice: Would you tell me, please, which way I ought to go from here?

The Cat: That depends a good deal on where you want to get to.

Alice: I don't much care where

The Cat: Then it doesn't much matter which way you go.

Alice: ...so long as I get somewhere.

The Cat: Oh, you're sure to do that, if only you walk long enough.

Alice in Wonderland
Lewis Carroll

Change vs. Alignment

- The current obsession with change as the rationale for architecture is misplaced.
 - Change is a means to the end of alignment.
 - Focusing on change as an end in itself only makes it possible for you to thrash faster.
 - Continuous change that never achieves alignment is continuous misalignment.
- If it is not architecture that ensures alignment, what does?
 - The most common alternative proposed is requirements engineering.

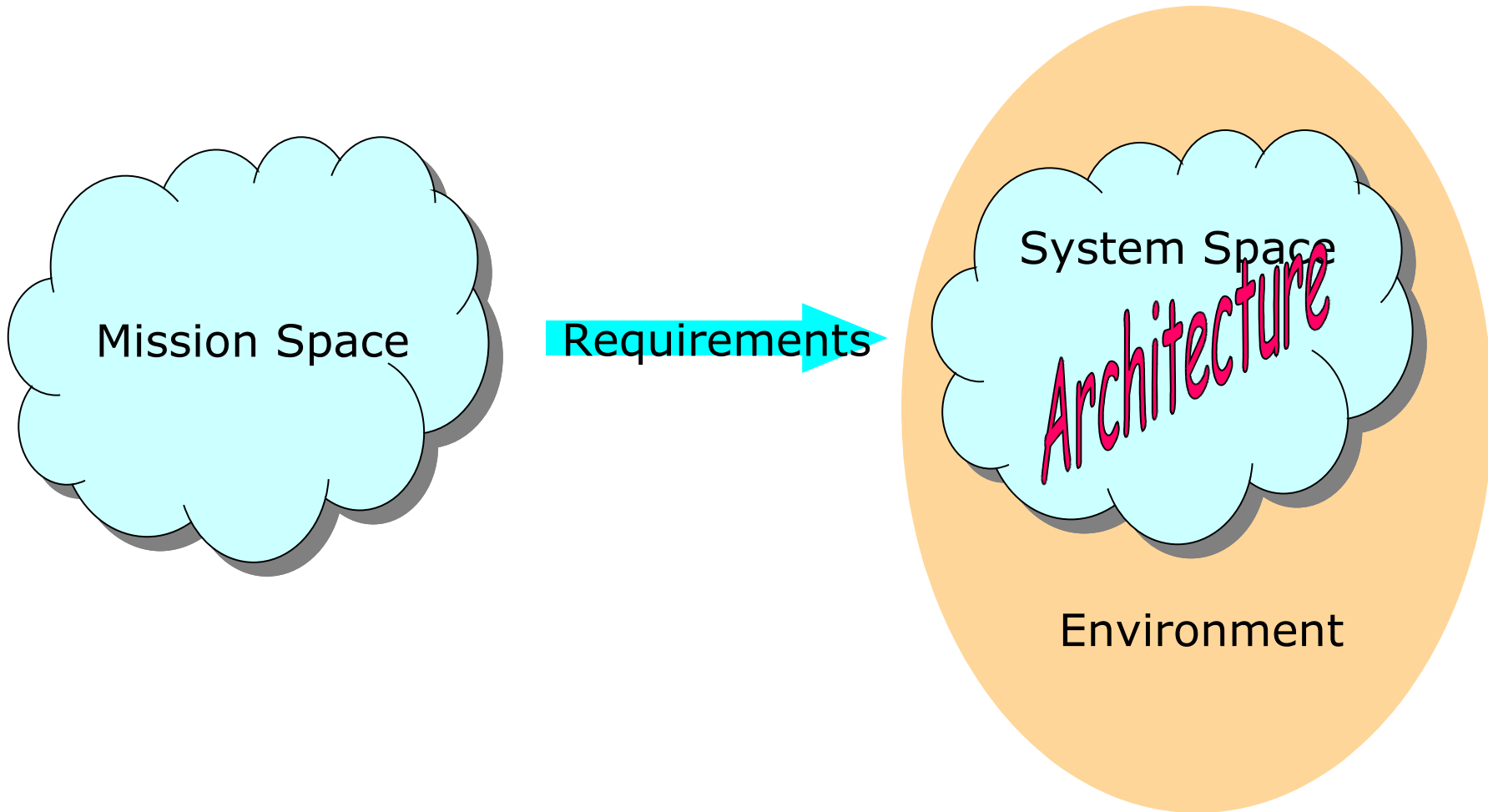
What About Requirements Engineering?

- Doesn't requirements engineering ensure alignment?
 - In practice it hasn't -- despite decades of work on requirements engineering, “business/IT alignment” and “requirements thrashing” remain apparently intractable challenges.
- This suggests that
 - The idea of “gathering” requirements is flawed.
 - The conventional model of the relationship between architecture and requirements is flawed.

Sidebar – Agile Development

- Extreme Programming and other forms of agile development (of software, of course) argue that Big Design Up Front (“BDUF”, of which architecture is a special case) cannot possibly succeed:
 - Stakeholders cannot know what they really want
 - The problem will change anyway before the project is completed.
- These are not problems with BDUF, they are problems with “Big Requirements Up Front”.
 - If requirements were stable and correct, BDUF would work just fine.

The Conventional View of the Relationship between Requirements and Architecture



What Requirements Engineering is Supposed to Do for Us

- ❑ Ensure that requirements are consistent, i.e., do not contradict or conflict with one another.
- ❑ Ensure that all necessary requirements have been specified.
- ❑ Ensure that no unnecessary requirements have been specified.
- ❑ Ensure that no requirements (unnecessarily) constrain the design and implementation.
- ❑ How is it possible to achieve these ends without some external frame of reference?

Heresy About Requirements

- ❑ Doing architecture in response to “gathered” requirements (by far the most common procurement model) generally hasn’t worked.
- ❑ Have we been putting the cart before the horse?
- ❑ Maybe we should first develop an architecture that specifies everything necessary to ensure alignment between the mission and its solution, and vet requirements against that architecture.
 - This would mean thinking of architectural principles as “upstream requirements”, and traditional requirements as “downstream requirements”.

What Comes with Essentials-based Alignment?

- ❑ Fitness for purpose
 - I.e., the desired value to the people expecting it.
- ❑ Only as much complexity as is necessary.
 - “Everything you need and nothing you don’t”
- ❑ Attention to the “-ities” that matter.
 - Adaptability and agility are only two of many “-ities”, and to assert, implicitly or explicitly, that they are *always* the *most* important is presumptive.
- ❑ Recognition of new mission opportunities.
 - The mission and its solution are duals of one another, and the mission can, and often should, be architected as well as the solution.

Agenda

- ❑ Introduction
- ❑ Brief History of the Idea of “Our Kind of” Architecture
- ❑ A Name for “Our Kind of” Architecture
- ❑ Why Do We Do Architecture?
- ❑ *Towards a Good Definition*
- ❑ Considering IEEE 1471 / ISO 42010
- ❑ (Re)Defining Architecture
- ❑ Conclusions

Attractive Nuisances

- ❑ Analogies with other architectural disciplines
 - Vitruvius' *utilitas, firmitas, venustas*
 - City planning as a metaphor for enterprise architecture
 - Blueprints
- ❑ Many architects use models, a special case of analogy, as an architectural tool.
- ❑ It is tempting to reason about architecture by analogy (e.g., “enterprise architecture is like city planning”) to apply what we learn by reasoning about the analog (e.g., city planning) to the thing itself (e.g., enterprise architecture).
 - The simpler the analog, the less likely we can do this, because relevant detail is missing.
 - The more complex the analog, the less likely we can do this, because irrelevant detail interferes.
- ❑ While models are analogies, not all analogies are models.
- ❑ Beware of any statement that begins “Architecture is just like ...”

Analogies, Models and Extrapolation

- The extrapolation principle:
 - If S is [enough] like M, then if X is true of M, X is also true of S.
- Models are a special case of analogies where we make considerable effort to understand how S is like M and for what Xs the extrapolation principle is valid.
- The best models consist entirely of Xs for which the extrapolation principle is valid.
- Bad models (and most analogies) include many Xs for which the extrapolation principle is not valid.

Risky Extrapolations

- ❑ A building is not a model of a program.
- ❑ A building is not a model of an enterprise.
- ❑ A program is not a model of an enterprise.
- ❑ A city is not a model of an enterprise.

- ❑ While an analogy may be an effective way to illustrate a concept, analogies are ultimately not helpful and can even be misleading as the basis for an in depth understanding of enterprise architecture.

Things related to, but not the same as, Architecture

- ❑ Program Management
- ❑ Strategy and Strategic Planning
- ❑ Governance
- ❑ Design, Development and Deployment
- ❑ Patterns
- ❑ Modeling
- ❑ Requirements Engineering and Management

Approximations of Architecture - 1

- Adoption of a specific vendor's product set
 - E.g., “our email architecture is Microsoft Exchange”
- A technical model for infrastructure
 - E.g., “our network architecture is TCP/IP”
- Adoption of a recognized pattern or structural style
 - E.g., “our application architecture is SOA”

Approximations of Architecture - 2

- Architectural decisions are abstract in nature
 - Architecture is about the “big picture”
- Architectural decisions are global in scope
 - A natural consequence of a high level of abstraction
- Architectural decisions are hard to change
 - A natural consequence of global scope

Properties a Definition of Architecture Ought to Have

- ❑ A good definition of architecture would be compatible with all these approximations, to the extent that they are correct.
- ❑ A definition of architecture ought to say, not just imply, what it “really” means.
- ❑ In particular, a definition of architecture ought to say how architecture achieves alignment.
- ❑ A definition of architecture ought to reflexively apply to itself, in the sense that it would directly express or at least strongly imply what the “architecture of architecture” is.
- ❑ A definition of architecture ought to be actionable; it ought to have sufficient substance that it is clear what “doing architecture” entails.

Questions a Definition of Architecture Ought to Help Us Answer

- ❑ Architecture is clearly a kind of design (in the sense of “Big D Design”). Exactly what kind of design is architecture?
- ❑ What sorts of things should be specified by an architecture, and what sorts of things should not be specified by an architecture?
 - I.e., what is the responsibility of the architect, and what is the responsibility of the downstream designer?
 - So, as much as possible, the language used to define a concept of architecture should be binary rather than a continuum.

Agenda

- ❑ Introduction
- ❑ Brief History of the Idea of “Our Kind of” Architecture
- ❑ A Name for “Our Kind of” Architecture
- ❑ Why Do We Do Architecture?
- ❑ Towards a Good Definition
- ❑ *Considering IEEE 1471 / ISO 42010*
- ❑ (Re)Defining Architecture
- ❑ Conclusions

The *de facto* Standard Definition of “Our Kind of” Architecture: ISO 42010 / IEEE 1471

- “The fundamental organization of a system embodied in its components, their relationships to each other and to the environment, and the principles guiding its design and evolution.”
 - IEEE Standard 1471: “IEEE Recommended Practice for Architectural Description of Software-Intensive Systems”
 - ISO/IEC WD1 42010: “Systems and software engineering — Architectural description”

Applicability of 42010

- “This International Standard applies to ‘systems that are man-made and may be configured with one or more of the following: hardware, software, data, humans, processes (e.g. processes for providing service to users), procedures (e.g. operator instructions), facilities, materials and naturally occurring entities (e.g. water, organisms, minerals).”
- “The principal users of this International Standard comprise stakeholders involved in processes throughout the systems and software life cycles ...”

Why not Just Accept 1471/42010?

- Many practicing solution and enterprise architects feel that the definition from IEEE 1471 is not well suited to their work
 - Even with the addition of principles, it is not significantly different from most definitions of internal design.
- The language of 1471's definition encourages thinking:
 - of the “system” as an end in itself rather than as a means.
 - about only the internal structure of the system.

1471 / 42010 and the Enterprise

- 1471 / 42010 was derived from ideas about software.
 - An enterprise, especially “the business” part of an enterprise, is not a “software intensive system”.
 - Architecting an enterprise is not solely, or even primarily, about “systems and software engineering”.
 - While software is important to most modern enterprises, it is not what most business people worry about.
 - The “everything is a system” metaphor is powerful but it’s still only a metaphor.
- To apply 1471 / 42010 to an enterprise requires considerable creativity in interpretation.

An Example of How 1471's Definition Violates One of my First Principles

- If we substitute “enterprise” for “system” in the 1471 definition, we get:
 - “The fundamental organization of an enterprise embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution.”
- This literally defines EA as the “org chart” with some editing guidelines.
 - This is not what most people intuitively understand EA to be about.
- Of course, you can liberally interpret this definition to make it mean what you already know it's supposed to mean ...

Important Interpretive Guidance from the 1471 FAQ - 1

- “Broadly speaking, an architecture is that which is essential or unifying about a system. It is that set of things about a system which largely determine the system's value, cost, and risk.”
- “...an architecture embodies ‘fundamental’ things about a system. We mean ‘fundamental’ in the sense of an abstraction of things that are important about the system as a whole—not in the sense of the top level in some arbitrary system/subsystem hierarchy.”

Important Interpretive Guidance from the 1471 FAQ - 2

- “‘Fundamental’ should be interpreted in the context of stakeholders and the environment. We cannot know what is fundamental about a system without knowing ‘fundamental to whom?’”
- “An architecture is *not just* the overall structure of physical components that make up a system. While physical structure is often a fundamental aspect of a system, it is not necessary.”

Why Isn't the 1471 / 42010 Definition Adequate?

- What 1471/42010 says and what the 1471 FAQ says it really means are not as well aligned as they could be.
 - The definition says fundamental, but the FAQ says important, essential, or unifying. Which is it?
 - The language of the definition is inherently structural (“fundamental organization”, “components”, “their relationships”), but the FAQ says it shouldn't be just about structure.
- 1471/42010 and the FAQ talk about the system, but they don't say anything about the system's mission.
 - The notion of fitness for purpose for the mission (i.e., alignment) is only very implicitly addressed by “value”, which is just another concern along with risk and cost, and seems to be considered an effect rather than a cause.
- What 1471/42010 says is not helpful in answering the questions a definition of architecture should help us answer.

Questions

- Is it necessary for something to be a “system” for it to have an architecture?
 - What are the consequences of thinking of something as a system without considering its mission? For one thing, “the system” too easily becomes the end rather than the means.
- What does “fundamental organization” mean?
- Which components and relationships?
 - Those that “embody the fundamental organization”?
 - How do we objectively recognize “embodiment”?
- Are there other things that architecture ought to say something about besides organization, components, relationships and principles?
 - Apparently, even the authors of 1471 believe so.

Agenda

- ❑ Introduction
- ❑ Brief History of the Idea of “Our Kind of” Architecture
- ❑ A Name for “Our Kind of” Architecture
- ❑ Why Do We Do Architecture?
- ❑ Towards a Good Definition
- ❑ Considering IEEE 1471 / ISO 42010
- ❑ *(Re)Defining Architecture*
- ❑ Conclusions

Please Bear In Mind...

“Right now, it's only a notion. But I think I can get money to make it into a concept. And later turn it into an idea.”

Partygoer in
Annie Hall

(Re)Defining Architecture

- ❑ The original concept of architecture articulated by the System/360 architects at IBM and adopted by the engineering community at Digital is more compatible with the way many practicing enterprise architects are coming to think of their discipline.
- ❑ Perhaps more importantly, a generalized Amdahl Blaauw and Brooks (GABB) definition of architecture appears to subsume the software concept of architecture, and offers a way to reconcile the two perspectives.

(Re)Defining Architecture

- ❑ Start from the idea of architecture as defining a class of acceptable solutions that address some goal, need, problem, ...
 - For the System/360, this was “a processor that presents the System/360 programming interface regardless of its implementation technology”.
- ❑ Recognize that to do so, an architecture must express the essential characteristics that define membership in that class.
- ❑ Recognize that these characteristics must be “mission-driven”, i.e., ensure that an instance of the class “is what it’s supposed to be” or “does what it’s supposed to do”.

Warning – Pitfall Ahead

- ❑ It is surprising how many people misinterpret the statement
 - “architecture is not solely or even primarily about structure”
- ❑ to mean
 - “architecture is not about structure”.
- ❑ Structure is relevant, it’s just not the *only* relevant thing.

(Re)Defining Architecture

- ❑ Explicitly address alignment.
 - Relate the role of architecture to the mission.
- ❑ Don't use inherently structural language.
 - Be more inclusive than just organization.
 - Be more inclusive than components and relationships.
- ❑ Be as specific as possible, but in the most general ways.
 - Use classes rather than lists of instances.
 - Use binary discriminators.
 - Say what you mean, don't just imply it.

Terminological Conventions

- I will use the following more or less interchangeably:
 - “Essential” and “necessary and sufficient”
 - “Suitability” and “fitness for purpose”
 - “Suitable” and “fit for purpose”
 - “Aligned with” and “fit for purpose for”
- I will use solution and mission to mean a thing (not necessarily an artifact) and its *raison d’etre*.
- Unless qualified (e.g., “Big D Design”), I will use design to mean the design activities that are conceptually proximate to implementation.

And Keep in Mind...

- The specific words in a definition are important only to the extent that they effectively represent the concept of the definition.

“When I use a word,” Humpty Dumpty said, in rather a scornful tone, “it means just what I choose it to mean - neither more nor less.”

“The question is,” said Alice, “whether you can make words mean so many different things.”

Alice In Wonderland
Lewis Carroll

Architecture

- “Those properties of a mission, its solution and their environment that are necessary and sufficient for the solution to be fit for purpose for its mission in that environment.”
 - Architecture binds the mission and its solution (“fit for purpose”).
 - “Necessary and sufficient” is a binary discriminator.
 - “Everything you need and nothing you don’t”
 - “Properties” is open ended and broadly inclusive.
 - Any relevant aspect that matters.
 - Language is nontechnical and nonstructural.
 - But inclusive of structure.

Observations about this Definition

- ❑ Architecture is as much about “why” as it is about “what” and “how”.
- ❑ An architecture defines a class of (acceptably equivalent) things.
- ❑ Architecture can be used to:
 - Recognize or classify things
 - Make things (artifacts)
- ❑ A thing’s essential properties are:
 - Anything and everything that matters
 - Structural, Behavioral, Functional, ...
- ❑ An architecture specifies these properties by saying what they
 - (Must/may/must not) x (be/have/do)
 - At the appropriate level of abstraction/generalality

Other Ways of Thinking About This Concept of Architecture

- ❑ Architecture is about “the stuff that matters”, regardless of what kind of stuff it is.
- ❑ An architecture is a class definition, even if the class contains only one instance.
- ❑ The architecture of a thing specifies those things (and only those things) that must be reproduced exactly to recreate an acceptably equivalent copy of that thing. This is what we really mean by “what matters”.
 - This is an effective way of thinking about what “necessary and sufficient” means.
 - It doesn’t matter if you don’t want or need to be able to make a copy.
 - It doesn’t have to be an exact copy.
 - The required degree of fidelity determines the required level of detail of the architecture.
 - “Acceptably equivalent” is determined by stakeholders’ concerns.

Reiterating: How we got here

- ❑ How can we be confident that a solution will be suitable for its mission, if those properties essential to its suitability for its mission have not been explicitly identified?
- ❑ If it is not architecture that ensures alignment, what does?
- ❑ If architecture does not specify essentials, what does?

Comparing 1471/42010 and GABB

IEEE 1471 / ISO 42010	Generalized Amdahl Blaauw Brooks
Fundamental	Necessary and sufficient (“essential”)
Organization	Properties
System	Mission and its solution
Embodied	Properties
Components	Properties
Relationships	Properties
Environment	Environment
Principles	Properties
Design	Fit for purpose (“suitable”) for its mission
Evolution	Fit for purpose (“suitable”) for its mission

Fundamental vs. Essential

- Fundamental does not mean essential:
 - Fundamental means foundational
 - Essential means necessary and sufficient
- 1471 / 42010 express “fundamental” in terms of the structure of the solution (“system”), while GABB defines “essential” relative to the mission (problem, need, opportunity, ...).
- A property can be fundamental (to the system) without being essential (for fitness for purpose for the mission), typically when the system is “wrong” (i.e., not fit for purpose for the mission)

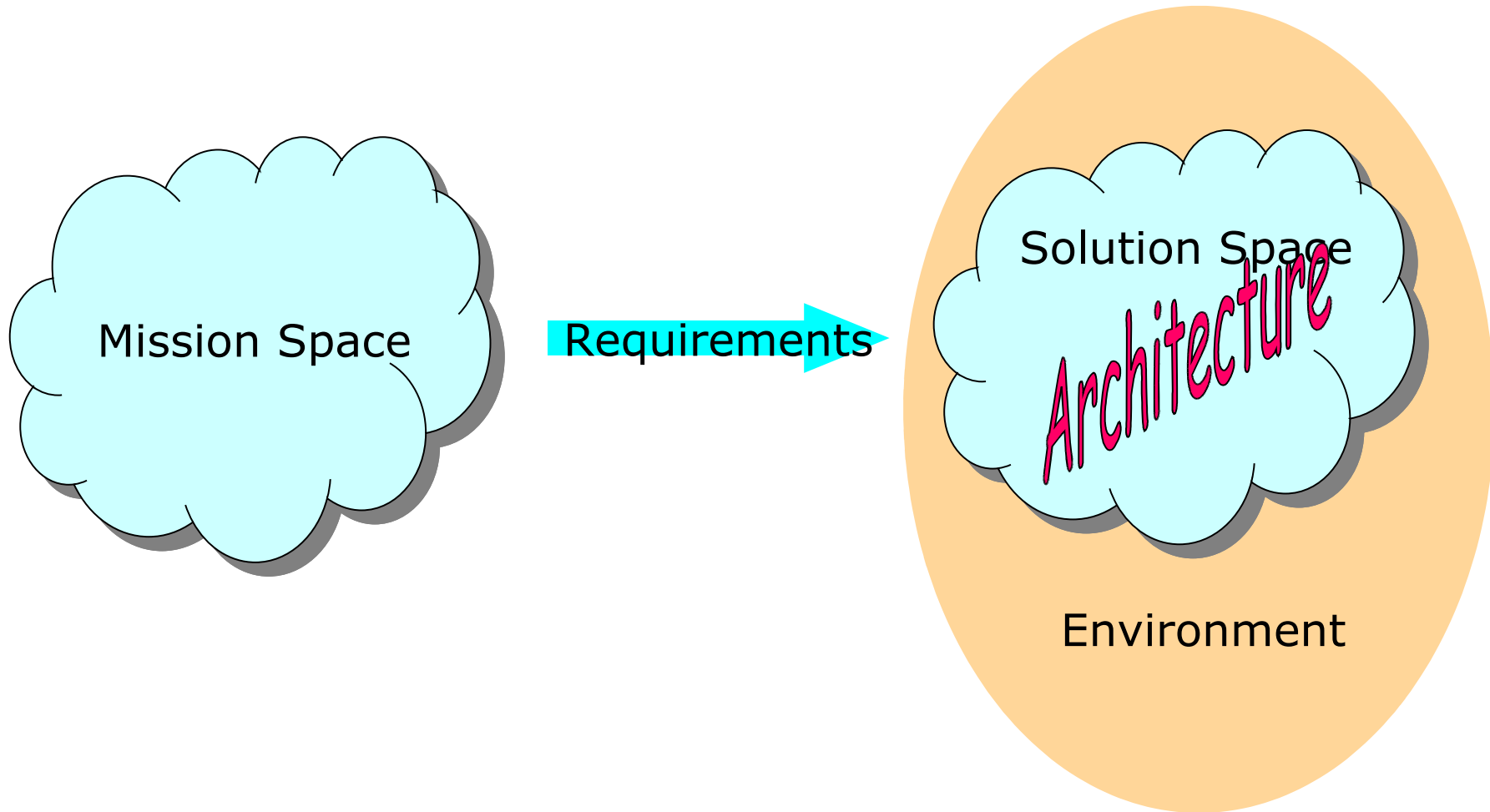
Fundamental vs. Essential

- ❑ Fundamentals may (but need not) be necessary, but they also may not be sufficient.
- ❑ Fundamental implies creating a foundation, but just being part of the foundation doesn't make something essential. Furthermore, something essential need not be fundamental.
- ❑ If fundamental “really” means essential, why not say essential (as it does in the FAQ)?

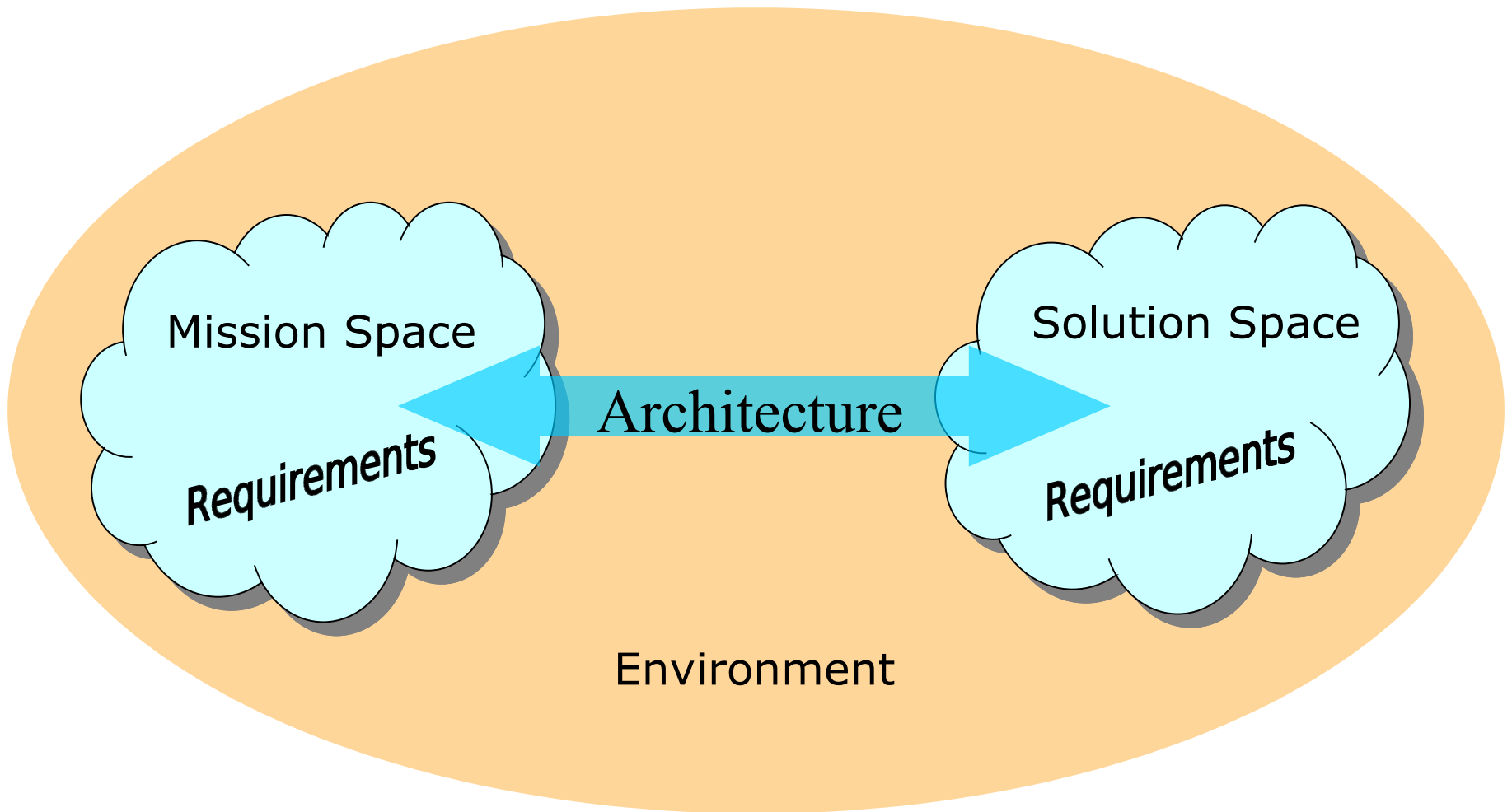
An Example

- ❑ What makes a chair a chair?
- ❑ How does a chair differ from:
 - A throne
 - A loveseat
 - A sofa
 - A bench
 - A stool
 - A large beanbag
 - A tree stump
 - A rock
- ❑ The set of properties that allows us to recognize a chair as a chair and not one of these other things is the architecture of a chair.
- ❑ Thinking about a chair this way makes you think about what a chair is for, not just how to build one, and if you really need a chair, or just something to sit on.

The Traditional Way We Think about Architecture



The Way GABB Implies We Should Think about Architecture



Agenda

- ❑ Introduction
- ❑ Brief History of the Idea of “Our Kind of” Architecture
- ❑ A Name for “Our Kind of” Architecture
- ❑ Why Do We Do Architecture?
- ❑ Towards a Good Definition
- ❑ Considering IEEE 1471 / ISO 42010
- ❑ (Re)Defining Architecture
- ❑ *Conclusions*

Curious Omissions

The GABB definition deliberately:

- ❑ Says nothing about technology.
- ❑ Says nothing about levels of abstraction.
- ❑ Says nothing about specific kinds of content.
 - In particular, says nothing about components and relationships, principles, standards, ...
- ❑ Says nothing about representation.
- ❑ Says nothing specific about scope.
- ❑ Says nothing about “business”.

The Standard Objections

- ❑ Why bring this up now? Isn't all this pretty much settled?
- ❑ Aren't there more important things to worry about?
- ❑ What are you, some kind of crackpot/wack job?
- ❑ You're obviously wrong, because nobody else says this.
- ❑ I don't like this word.
- ❑ I don't use this word to mean what you use it to mean.
- ❑ This definition doesn't mention <x>.
- ❑ This definition is too vague to be of any use.
- ❑ You're right, but nobody else will ever agree with you, unless you change it this way...

- ❑ But, so far, nobody has asserted that:
 - Architecture isn't about essentials (necessity and sufficiency).
 - Architecture isn't about alignment (fitness for purpose).
 - Architecture shouldn't be defined as a class.

Implications of this definition

- ❑ It is not meaningful to talk about “the architecture of something” without reference to its mission.
- ❑ The role of the architect is less to decide or design than to infer, from the mission, the essential properties of a solution, which bound the design space.
- ❑ In the real world, most missions are not well-defined. This means architecture can be as much about architecting the mission as it is about architecting its solution.

Implications of this definition

- Stakeholders are the people who decide what matters.
 - Because stakeholders decide what is essential, there can be no single standard “template” for all architectures.
 - Stakeholder concerns define both the breadth and depth of an architectural specification.
 - Different stakeholders may have different ideas of what the mission is, and what the necessary and sufficient properties of the solution are.

Implications of this definition

- Today, very little “pure” architecture in this sense is done. Most architecture is done as an integral, undifferentiated part of an intuitively bounded general design effort.
 - This means that most architectures are either underspecified or overspecified, if not both at the same time.
 - I suspect most architects/designers stop architecting/designing when they are satisfied that the implementation will be done “their way” rather than “the correct way”.
 - What gets included in an architectural specification is usually determined by what a method, deliverable, or template calls for rather than what is necessary and sufficient to ensure fitness for purpose.

Implications of this definition

- ❑ “Fit for purpose” is binary. “Necessary and sufficient” is binary. Therefore:
 - It is not meaningful to prioritize, or specify as optional, any elements of an architecture. An architectural element is either essential, or it is not. Nonessentials are not part of the architecture.
- ❑ It may be tempting to prioritize, or specify as optional, elements of the mission. This will only introduce noise and distraction.
- ❑ In the real world, you can never get this exactly right, but that does not justify using sloppier language to define architectural inclusion. You won’t approach what you don’t aspire to.

Two Critical Consequences

- Different stakeholders may have different ideas of what the mission is, and what the necessary and sufficient properties of the solution are.
 - If you don't address this from the outset, you will have serious problems later.
- Most architectures are either underspecified (insufficient) or overspecified (unnecessary), if not both at the same time.
 - Underspecified means that downstream design and implementation decisions may render the solution unfit for purpose.
 - Overspecified means that unnecessary constraints on downstream design and implementation decisions may have unnecessary and perhaps adverse consequences for things like cost, performance, ease of use, ...

Architecture vs. Design

Architecture	Design
Essentials dictated by the mission (problem, need or opportunity) and its environment	Decisions compatible with the architecture
A different architecture implies a different mission	Different designs may address the same mission
Defines a class of acceptable solutions	Defines a single specific solution
About suitability or fitness for purpose, as defined by the mission	About engineering optimization, within architectural constraints
Role of the architect is mostly to make correct inferences about the mission, solution and environment	Role of the designer is mostly to make correct decisions about the solution
Architecture is done by architects	Design is done by developers
Primary audience is mission and solution stakeholders, which usually includes designers and implementers	Primary audience is solution implementers
About the mission and solution in their environmental context, i.e., outward looking	About components and subsystems of the solution, i.e., inward looking

Applying this Concept of Architecture to the Enterprise

- Enterprise Architecture is, literally, the architecture of an enterprise. I.e.,
 - Those properties of an enterprise, its mission, and their environment, that are necessary and sufficient for the enterprise to be fit for purpose for its mission in that environment.
- The value proposition for Enterprise Architecture defined this way is:
 - Continuous alignment of an enterprise's assets and capabilities with its mission and strategy.

Integrating the Value Proposition with the Definition of Enterprise Architecture

- Those properties of an enterprise, its mission, and their environment, that are necessary and sufficient for the enterprise to be fit for purpose for its mission in that environment, so as to ensure continuous alignment of the enterprise's assets and capabilities with its mission and strategy.

Implications of this Definition of Enterprise Architecture

- ❑ Enterprise architecture is as much about what the enterprise needs, as it is about what the enterprise has.
- ❑ Alignment is a dynamic, not static, condition.
- ❑ Continuous alignment requires change, and hence agility, but change and agility are means, not ends.
- ❑ Enterprise architecture is not only about what the enterprise wishes to achieve, but also about how the enterprise wishes to achieve it. Architecture is not only about value, it is also about values.

IT-Centric vs. Enterprise Centric

IT and “everything else”,

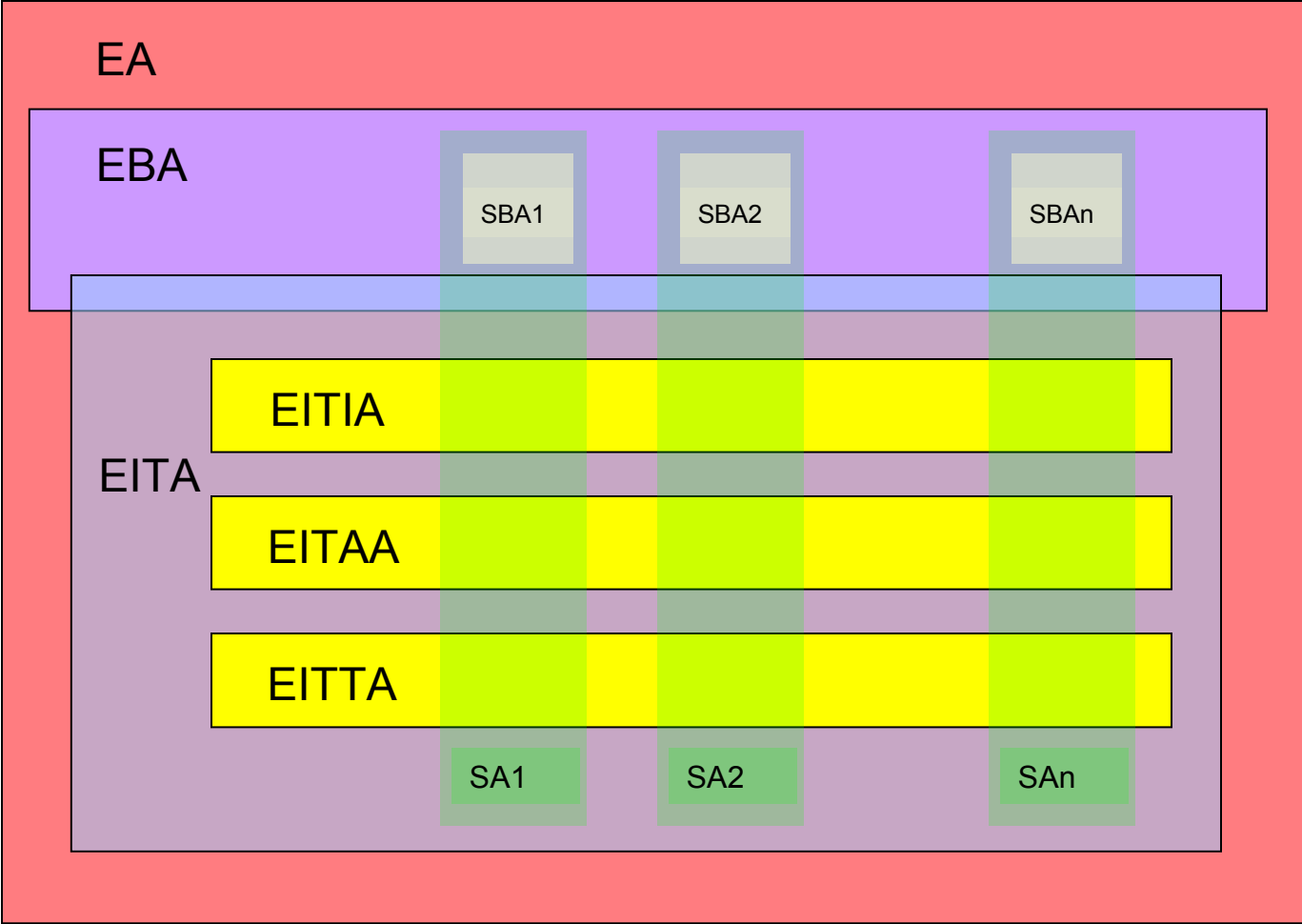
vs.

IT as one of many
collaborating strategic
assets.

Does “everything else”
have any internal structure
that is meaningful from the
enterprise architecture
perspective?



An IT-Centric Architecture of Enterprise Architecture



EBA: Enterprise Business Architecture

SAn: Solution Architecture <n>

SBA_n: Solution Business Architecture <n>

“The business” is implicitly defined as whatever isn’t IT.

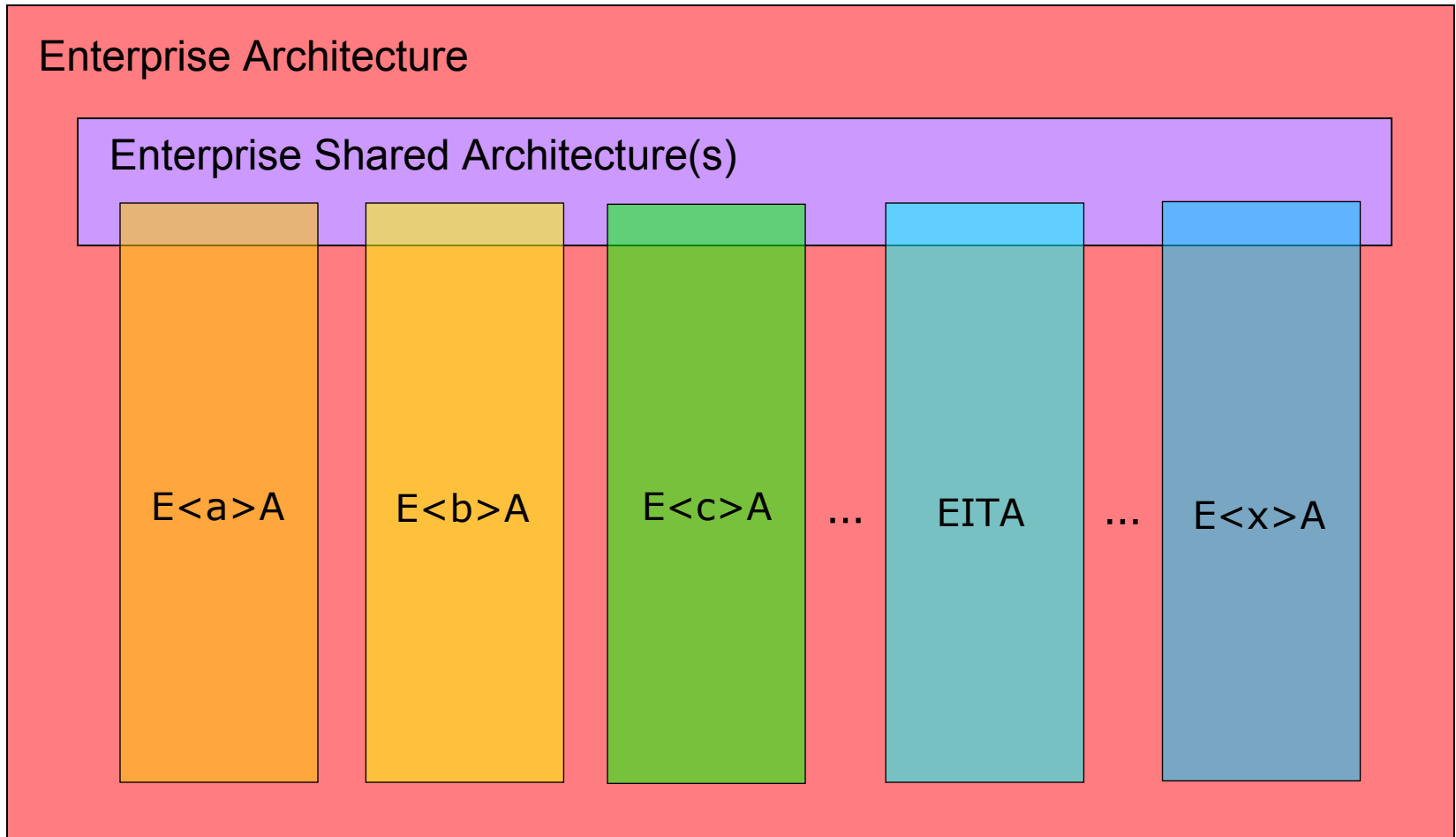
A Non-IT-Centric View of “The Business”

- “The Business” (nominally, the part of the enterprise which is “not IT”) is not monolithic.
- It comprises, for example:
 - Strategy
 - Finance
 - Legal
 - Marketing
 - Sales
 - Manufacturing
 - Shipping
 - Purchasing
 - Receiving
 - Engineering
 - Research
 - Product/Project/Program Management
 - Facilities
 - Human Resources
 - IT

A Non-IT-Centric View of “The Business”

- ❑ Each of these constituents of the enterprise conceivably has an architecture (just as IT does) that aligns its assets, processes, capabilities, etc., with its role in the enterprise’s mission.
- ❑ These architectures will share some (architectural) properties of the enterprise as a whole but will also have their own (architectural) properties that are compatible and synergistic with the shared properties, and the properties of the other architectures.

A non-IT-Centric Architecture of Enterprise Architecture



Testing Against My Goals

- ❑ Develop and justify a concept of architecture that clearly distinguishes it as a specific kind of design (in the “Big D Design” sense), and that can be consistently and meaningfully applied across the entire domain of interest to us.
- ❑ Define this concept of architecture in language that is independent of the context within which architecture is applied, or the medium by which it is expressed, executed or implemented.
- ❑ Develop and justify a model of architecture in the enterprise context (i.e., “enterprise architecture”) that usefully applies to the entire enterprise, not just its IT assets, and to enterprises that are not “businesses”.

Testing Against Desired Properties




- A good definition of architecture would be compatible with all these approximations, to the extent that they are correct:

- Adoption of a specific vendor's product set
- A technical model for infrastructure
- Adoption of a recognized pattern or structural style
- Architectural decisions are abstract in nature
- Architectural decisions are global in scope
- Architectural decisions are hard to change



- A definition of architecture ought to say, not just imply, what it means.



- In particular, a definition of architecture ought to say how architecture achieves alignment.

Testing Against Desired Properties

- ❑ A definition of architecture ought to reflexively apply to itself, in the sense that it would directly express or at least strongly imply what the “architecture of architecture” is.
- ❑ A definition of architecture ought to be actionable; it ought to have sufficient substance that it is clear what “doing architecture” entails.

Testing Against the Questions a Definition Should Help Us Answer

- ❑ Exactly what kind of design is architecture?
- ❑ What sorts of things should be specified by an architecture, and what sorts of things should not be specified by an architecture?

Reprise

- ❑ Introduction
- ❑ Brief History of the Idea of “Our Kind of” Architecture
- ❑ A Name for “Our Kind of” Architecture
- ❑ Why Do We Do Architecture?
- ❑ Towards a Good Definition
- ❑ Considering IEEE 1471 / ISO 42010
- ❑ (Re)Defining Architecture
- ❑ Conclusions

The Architecture of Architecture

- Mission, solution, environment
 - Continuous dynamic alignment
- What matters
 - Fit(ness) for purpose
 - Essential / necessary and sufficient

Thank You!

Can We Talk?