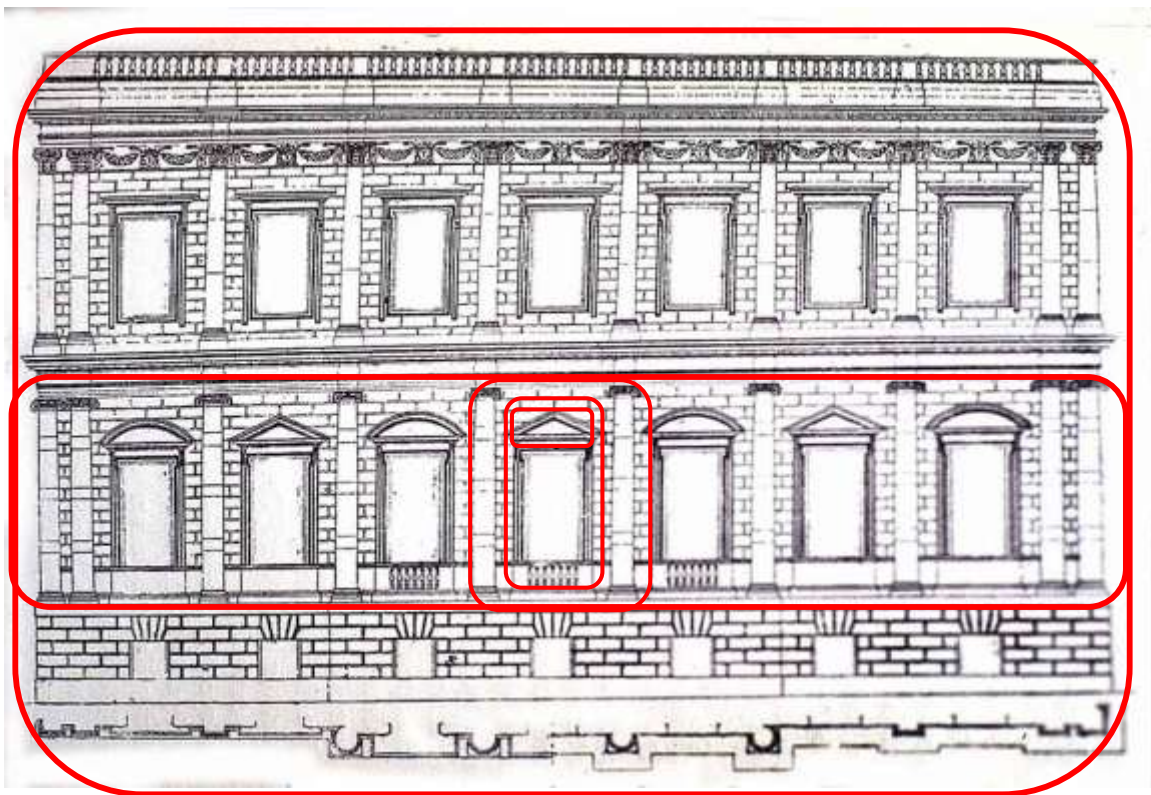


Urban and Enterprise Architectures: A Cross-Disciplinary Look at Complexity

Roger Sessions
Nikos A. Salingaros

Lecture Notes, IASA Webinar



This talk was originally presented as an IASA Webinar in July 2010. The present paper is a revised and expanded version that incorporates discussions during and after the online presentation.

Version: 1.0: 15 October 2010

Table of Contents

Executive Summary	3
Part I. Introduction	4
Author Bios	4
Bird's Eye Perspective	5
History Behind the Salingaros/Sessions Collaboration	5
Definition of Complexity	7
Implications of Complexity on Enterprise Architectures	7
Part II. Coherency in Urban Architectures	8
Salingaros on Coherency in Urban Architectures	8
Four Factors for Architectural Coherence	9
Factor 1. Scale Range	10
Factor 2. Scale Hierarchy	11
Factor 3. Scale Couplings	12
Factor 4. Scale Tightness	13
Architectures Without Coherence	14
Part III. Coherency in Enterprise Architecture	16
Applying Coherency to Enterprise Architecture	16
Factor 1. Scale Range	17
Factor 2. Scale Hierarchy	19
Factor 3. Scale Connections	19
Factor 4. Scale Tightness	20
Enterprise Architecture Systems with Coherence	21
Connection and Boundary Rules	22
Coherent Enterprise Architecture	22
Classic Enterprise Architecture	23
Part IV. Importance of Coherence	24
Coherent Architectures: Easier to Implement	24
Coherent Architectures: Easier to Maintain	25
Coherent Architectures: Easier to Adapt	26
Coherent Architectures: Less Complex	27
Summarizing Architectural Coherency	27
Who Cares About This?	28
Who violates these Rules?	28
Part V. Wrap-up	30
Where does SIP fit in?	30
Discussion	30
For more Information	35
Legal Notices	36
To Join the Discussion	36

Executive Summary

Enterprise architecture is an emerging discipline that looks at the relationship between business architectures and information and communication technology architectures. Enterprise architects are responsible for making sure that the technical architectures delivered by the information and communication technology architects closely match and meet the needs of the business systems they are designed to support.

Complexity is a major problem in enterprise architecture. Complexity obstructs business/IT alignment. Complexity obscures project vision. Complexity delays schedules, drives cost overruns, and hinders delivery of value. For enterprise architecture, complexity is the enemy.

Complexity is also a problem for urban architecture. In urban architecture complexity causes habitats that are destructive to the human psyche. Complexity can cause a variety of stress responses such as increased heart rate, sweating, and pupil dilation. For urban architecture, complexity is not just a matter of aesthetics; it is a matter of social wellbeing.

Although the symptoms of unmanaged complexity differ in these two fields, the laws governing architectural complexity are universal. Successful architectures follow the same basic principles whether those architectures describe biological, urban, or enterprise systems. And in all cases, the cost of ignoring these principles is disorder, dissolution, and ultimately chaos.

Urban architects have learned a great deal about these universal principles of designing successful architectures in the midst of high complexity. There are four main attributes of successful architectures. They are: 1) *scale range* (a range of scales from small to large), 2) *scale hierarchy* (some sort of containment of small scales within larger scales), 3) *scale connections* (strong connections are allowed only between elements at similar scales), and 4) *scale tightness* (connections decrease in number and strength at larger levels of scale). Architectures that exhibit all four of these properties are called *coherent* architectures. Architectures that lack one or more of these properties are called *incoherent* architectures.

This concept of architectural coherence is critical in enterprise architecture. Coherent enterprise architectures are easier to bring into business/IT alignment. They have more predictable budgets and schedules and are more likely to deliver business value. Enterprises that are built on a coherent enterprise architectural foundation are nimble, efficient, and adaptable. Those that are not are sluggish, inefficient, and lifeless.

Despite the obvious importance of coherency to enterprise architecture, this concept is new to the field. Organizations spend considerable time and money developing enterprise architectures and have no way to evaluate the results. Now they can. The amount of coherency the enterprise architecture has is a strong predictor of how much business value that architecture will ultimately deliver.

Part I. Introduction

Author Bios

Roger Sessions is a leading thinker in Enterprise Architecture. He is the CTO of ObjectWatch, a company he founded 15 years ago, a Fellow of the International Association of Software Architects, and a founder of The Consortium for Untangling Enterprise Complexity. He has written seven books including his most recent, *Simple Architectures for Complex Enterprises*, many articles and white papers. His most recent white paper is titled *The IT Complexity Crisis; Danger and Opportunity* (available at www.objectwatch.com). A frequent keynote speaker, Sessions has presented in countries around the world on the topic of Enterprise Architectural Complexity.

Nikos A. Salingaros is the author of *Anti-Architecture and Deconstruction* (2004), *Principles of Urban Structure* (2005), and *A Theory of Architecture* (2006), as well as numerous scientific papers. Both an artist and scientist, he is Professor of Mathematics at the University of Texas at San Antonio, and is also on the architecture faculties of universities in Holland, Italy, and Mexico. He designed the Commercial Center in Doha, Qatar in collaboration with Hadi Simaan and José Cornelio-da-Silva. Dr. Salingaros' theoretical work underpins and helps to link new movements in architecture and urbanism, such as New Urbanism, the Network City, Biophilic Design, Self-built Housing, and Sustainable Architecture. He is working with the Peer-to-Peer Foundation to promote self-built housing for the developing world. Dr. Salingaros collaborated with Christopher Alexander, helping to edit the four-volume *The Nature of Order* during its twenty-five-year gestation, a work that had a huge impact on the Patterns Movement in Software Architecture. In recognition of his efforts to understand architecture using scientific thinking, he was awarded the first grant ever for research on architecture by the Alfred P. Sloan Foundation, in 1997. Dr. Salingaros is a member of the INTBAU College of Traditional Practitioners, and is on the INTBAU Committee of Honor. Dr. Salingaros was included as one of the "50 Visionaries" selected by the UTNE Reader in 2008.

Bird's Eye Perspective

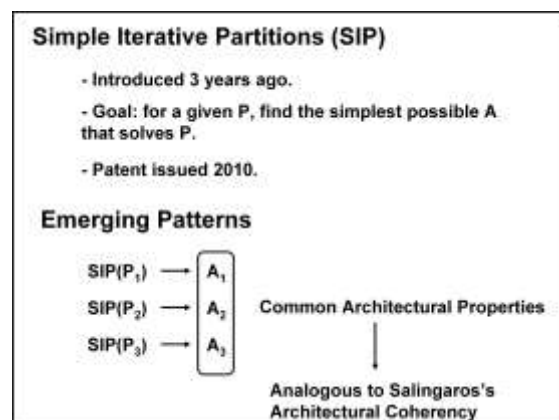
- What are we talking about?
 - Architectural Coherency
- Why?
 - Because Architectural Coherency reduces complexity
- How will we look at it?
 - We will look at Architectural Coherency in two different domains, Enterprise Architecture and Urban Architecture
- What is the promise?
 - We can architect systems that are more adaptable, more reliable, and less expensive to build

Complexity is a term that means different things to different people. So we will start out by describing what we mean by complexity. We will then look at how complexity in the urban architectural domain is related to an architectural attribute we call *architectural coherence*. We will then see how the concept of architectural coherence might apply in the field of enterprise architecture.

Once we understand what a coherent enterprise architecture looks like, we can compare that architecture to a typical “incoherent” enterprise architecture. Understanding the difference between coherent and incoherent architectures is important because architectural coherency is our main tool to manage complexity.

The concept of architectural coherency is cross disciplinary. We will be specifically discussing the concept as it has been realized in urban architecture and can be realized in enterprise architecture. As we will show, architectural coherence at the enterprise level results in systems that are significantly less complex, and as a result, more adaptable, more reliable, and less expensive to build and maintain.

History Behind the Salingaros/Sessions Collaboration



Sessions has been working on a methodology called SIP (Simple Iterative Partitions) for several years. The goal of SIP was to find the simplest possible enterprise architecture that solves a given business problem. SIP was issued a patent in 2010 nicknamed *the anti-complexity* patent, the first patent ever issued for a methodology to simplify an enterprise architecture.

One of the key points about SIP is that it is focused on simplifying an enterprise architecture using mathematical reasoning. Salingaros was interested in how this reasoning might apply to urban architectures and sent Sessions his work on coherency in urban architectures. Sessions saw in the concept of architectural coherence principles that could apply in helping to recognize “good” enterprise architectures.

Complexity, it should be noted, is not specifically linked to either enterprise or urban architectures. It is, instead, a mathematical topic and it should therefore be no surprise that the concept applies to a wide variety of disciplines.

The problem of complexity is itself rather complex. Nobody really understands complexity. We consider here two competing types of complexity. There is organizational complexity, which is requisite, and means that a system is composed of many parts. And you have unnecessary complexity which messes things up, that is, makes things difficult to understand and often results in a total breakdown of the system. Coherent complexity is essential for a system to work, and this competes against the destructive type of complexity that is making the system break down. Our aim is to get our hands around the different types of complexity, that is, to understand them and to hopefully intervene.

These problems go back a long time to at least Herbert Simon, Nobel prize winner in economics who looked at the role of partitioning in reducing complexity, but nothing was developed that was really applicable towards our goals. Christopher Alexander has been working for many years looking at complexity on the architectural and urban scales. Salingaros’s work is closely related to Alexander’s.

One of the important ideas Alexander introduced was *Patterns*, first in building architecture and then in software architecture. The idea of patterns threw tremendous light on design complexity.

Later, Alexander with his four-volume book *The Nature of Order* looked at the generated type of complexity, namely the iterative building up of complexity. Alexander realized that a key concept missing from *Patterns* was degenerative complexity. What is missing is actually the iterative development of partitioning.

Definition of Complexity

Complexity is the attribute of a system that makes that system difficult to use, understand, manage, and/or implement. Complexity is measured in some units defined by a simplification framework.

- CUEC – Standard Definitions of Common Terms
(www.cuec.info/standards/)

Complexity is an attribute of many systems across many domains. In general, complexity is what makes a system difficult to use, understand, or maintain. As defined by the Consortium for Untangling Enterprise, it is a measurable attribute that can and should be reduced.

Implications of Complexity on Enterprise Architectures

Question: Say two architectures, A_1 and A_2 both solve P . Which should we implement?

Answer: The least complex of A_1 and A_2 .

Problem: How do we know which is least complex without implementing them?

Enterprise architectures are intended to solve some specific business problem. So the enterprise architect must start by understanding the business problem and then devising a high-level architecture that solves this need. That architecture will typically include information about how software systems should be implemented and packaged. The architecture is then turned over to the Information and Communications Technology groups who then do more detailed plans on the actual software and data systems that need to be created.

For a given business problem, there are a large number of possible architectures that could solve that problem. Existing methodologies are non-directed, meaning that different groups analyzing the same problem even using the same methodology are highly likely to come up with different architectural solutions.

The question then becomes: if there are two or more proposed architectures — all of which solve the same business problem — how do you choose between them?

There are a large number of architectural attributes that one could look at in making this decision. One could, for example, choose the more secure of the two proposals. Alternatively, one could choose the more flexible of the two proposals.

In reality, there are so many attributes one could consider in choosing between the two candidate architectures that the choice most often comes down to non-rational decision making. The choice between the two architectures is usually not based on attributes of the architectures themselves, but on attributes of those championing the proposals.

Our assumption here, at the heart of our proposal, is that we can make a more rational choice based on the simplicity of the architectural proposals. If we have two proposals that both solve the same problem, then the simpler of the two is the better one.

One reason for this decision is that most other architectural attributes (security, flexibility, etc.) are, in fact dependent attributes that are linked, either positively or negatively, to the independent variable complexity. Thus, for example, systems that are more complex are less flexible. Systems that are more complex are also less secure. We shall explore the relationship between these dependent attributes and complexity later.

Part II. Coherency in Urban Architectures

Salingaros on Coherency in Urban Architectures

Geometric coherence is an identifiable quality that ties the city together through form and is an essential prerequisite for the vitality of the urban fabric

- Principles of Urban Structure by Nikos Salingaros, 2005, Techne Press. All remaining quotes taken from here as well.

Salingaros, in *Principles of Urban Structure*, identified *geometrical* (or *architectural*) coherence as an identifiable quality that ties the city together through form and is an essential prerequisite for the vitality of the urban fabric. Cities that have this coherence have great vitality. Cities that lack it seem lifeless.

Four Factors for Architectural Coherence

- Scale Range
- Scale Hierarchy
- Scale Connections
- Scale Tightness

There are a number of factors that Salingeros identified as necessary for architectural coherence, but these four seem particularly relevant to Information and Communications Technologies systems. The four factors for architectural coherence are:

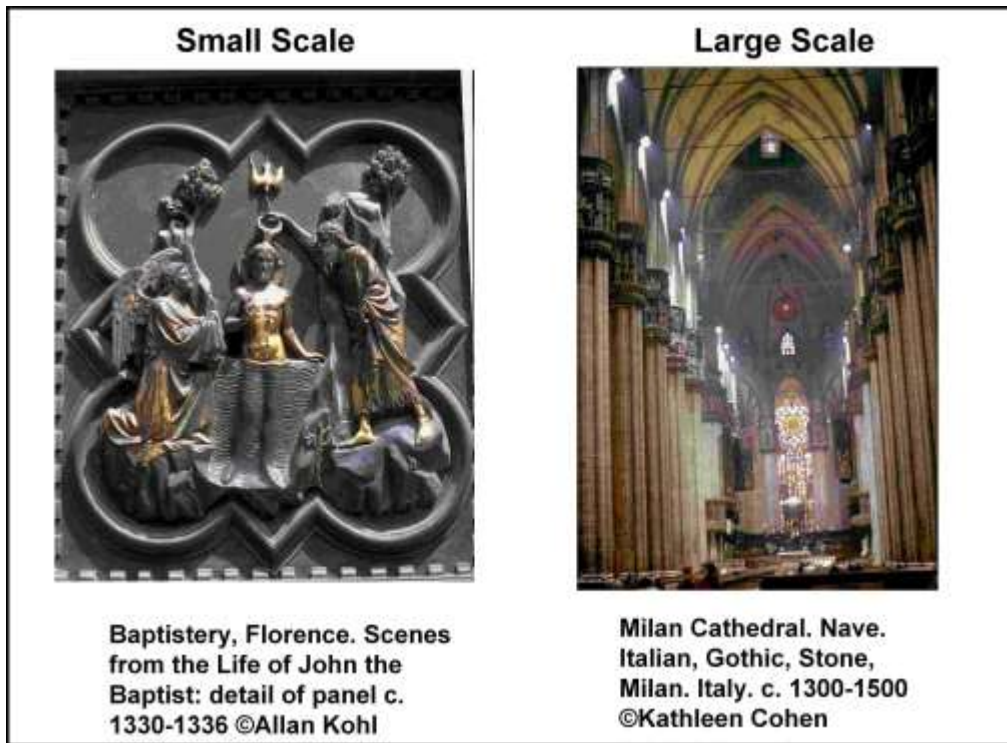
1. scale range
2. scale hierarchy
3. scale connections
4. scale tightness

Cities that have these factors within their organizational architecture have “coherence” and therefore vitality.

As we will see, these coherency factors are themselves hierarchical in that each builds upon the previous ones. Therefore, *scale range* is a basic notion. Without it, none of the other coherency factors are possible. If a particular architecture has *scale range*, then *scale hierarchy* is possible. And if it has *scale hierarchy*, then *scale connections* become possible. And if all of the first three coherency factors are present, then *scale tightness* is possible. And if (and only if) all four of these coherency factors are present, then we can say that the architecture is coherent.

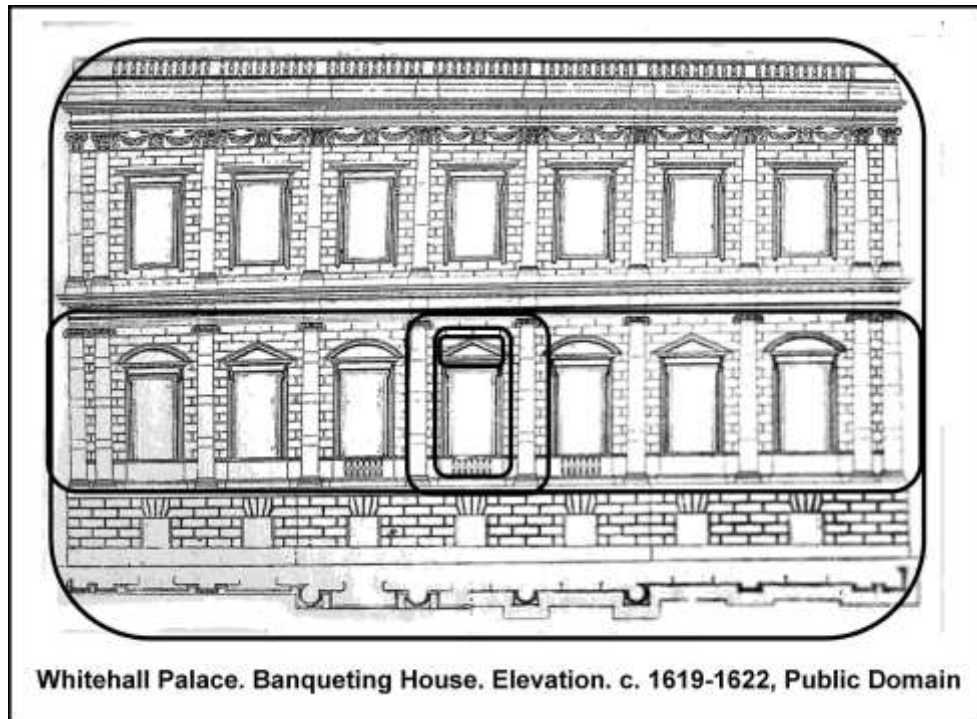
We’ll start looking at each of these factors from the perspective of urban architecture and then see how these same factors apply to Information and Communications Technology architectures. Information and Communications Technology architectures that have these same factors possess a vitality that is lacking in non-coherent architectures. We will see that vitality is a necessary attribute for a successful Information and Communications Technology architecture, that is, an architecture that is implementable, that delivers real value, and can be adapted to the future needs of the organization.

Factor 1. Scale Range



Scale range, the first requirement for architectural coherency, means that an architecture must have scales ranging from the very small to the very large and a number of scales in between. There should be a stepwise progression between scales so that the jump from scale to scale is never very large.

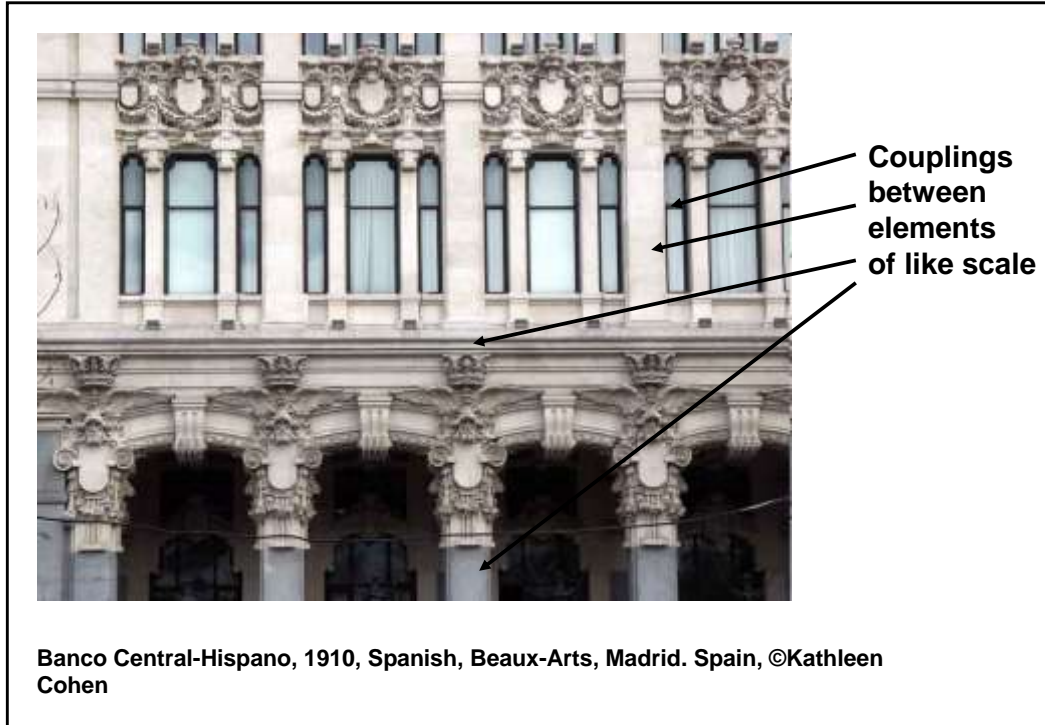
Factor 2. Scale Hierarchy



First of all, we can see in the above illustration that there is a range of scales ranging from the smallest, a brick, to the largest, the entire building. We also see that there is a number of scales in between. We have windows, window casements, and strong horizontal divisions. We also see that there is a stepwise progression from one scale to the next. In other words, there are no large jumps from one scale to the next. Thus this architecture meets the first factor, that of *scale range*. This is good, because *scale range* is a basic requirement. An architecture that lacks *scale hierarchy* cannot have any of the other three coherency factors.

Now let's look at *scale hierarchy*, the second requirement for coherency. *Scale hierarchy* means that each of the scales from small to large is related to every other scale, most likely in a hierarchical manner. Elements of smaller scale are contained within elements of larger scales. Bricks are contained within window casings. Window casings are contained within floors. Floors are contained within the building. This hierarchical order helps us make sense of the architecture.

Factor 3. Scale Couplings



Scale couplings refers to the requirement that like elements be connected to like elements. The first coupling shown above is a horizontal element connecting two window panes. The second coupling is a vertical element connecting two major vertical building subsets. The third coupling is a horizontal element connecting two major horizontal building subsets. The fourth coupling is a column that connects two entrance spaces. In all cases, we have elements being connected to other elements that are of like scale.

Factor 4. Scale Tightness



The final coherency factor is *scale tightness*. This means that at the highest level of scale, connections are very few and very loose. By “loose” we mean that the elements being connected are relatively independent of each other. In the picture above, the vertical edges of the building segments serve as the highest level connections, meaning that the elements they connect are at the largest dimension of scale. So we have very few of them and they are very loosely connected. We could replace a major building segment by another that contains the same smaller elements of scale, with little impact on the building segment to which it is connected.

Going down in scale, connections between window segments are greater in number and “tighter”. It would be harder to replace any window segment without forcing changes to the window segment to which it connects. And the lowest level couplings are those that connect the smallest elements of scale. We have here the largest number of these connections, and they are the tightest connections in the architecture.

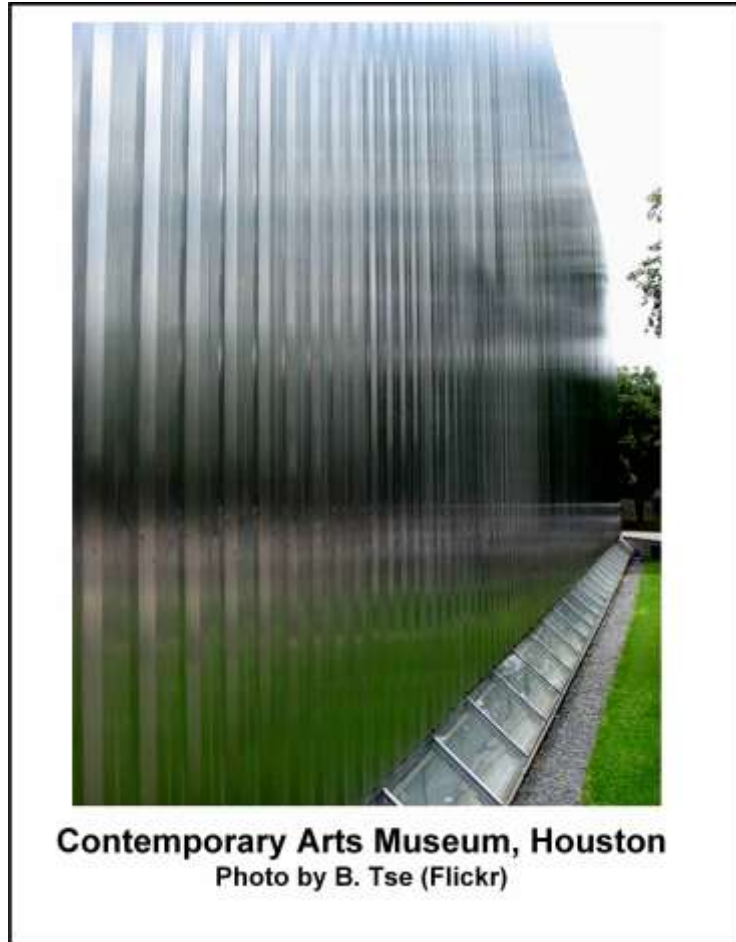
Notice that the building shown has all four coherency factors as do all of the buildings we have seen so far.

Now you might think that it is obvious that any building should be designed and built with all four scale factors in mind. But it turns out that many buildings do not have architectural coherence. Let’s consider two of these next.

Architectures Without Coherence



This building lacks all of the coherency factors. It has only a few scale ranges, far fewer than a building of this size calls for. The scales that it does have are not arranged hierarchically. The couplings are occurring between non-like elements. And the tightness of couplings does not follow the scale range.



This building is the Contemporary Arts Museum in Houston. It is quite unsettling when you first see it. In fact, it isn't even clear that you are looking at a building. The reasons for this are obvious. All four of the coherency factors are missing.

There are actually physiological reasons why humans and even mammals respond in a positive way to buildings that have coherence and in a negative way to buildings that lack coherence. This has to do with the concept of Biophilia first introduced by the pioneering biologist Edward O. Wilson. In our evolution, we are structured so as to recognize architectural coherence. The reason for this is that our natural environment, the one in which we evolved into human beings, is an environment that is resplendent in coherency. This means that we are hard-wired as perceptive machines to recognize coherency and even to require it.

Professor Wilson has posited that coherence actually leads to a healing effect on us. In the last ten years, laboratory experiments have demonstrated that there are healing properties to environmental coherence. So this isn't just esthetics, it is human biology.

The mathematical description of all of this can be recast in terms of "scale-free systems". A scale-free system has scaling coherence in all of the scales so that no single scale is prioritized; it doesn't jump out at you. The picture above shows a building that violates

this rule totally. You have this giant scale and some small vertical scale. That's only two scales and there is no hierarchy that links the small to the large scale. The built structure is defining the large scale above all. This is the opposite concept of scale-free.

Many listeners/readers will be familiar with scale-free and inverse power-law distributions. We find these in all stable complex systems. Namely, there are few things on the large scale, many things in the intermediate scale, and many, many things on the smaller scale.

All biological systems are built using scale-free and inverse power-law distributions. One such example is the mammalian lung, which has beautiful scaling coherence with eight levels of scale.

We also see these features in artificial systems, such as electrical distribution systems and the world-wide web. Most complex electronic devices have evolved this same inverse power-law distribution for the sake of efficacy and efficiency.

So we are looking at properties not just of Information and Communications Technology systems or urban systems, but invariant properties that all stable complex systems obey.

If we look at violations of these universal properties, we find that these lead to destabilization of systems. We see this in biological systems. For example, the healthy human heart beats by following an inverse power-law distribution. The first sign of an oncoming heart attack is when the electrocardiogram shows that the beats have stopped following this rhythm.

Part III. Coherency in Enterprise Architecture

Applying Coherency to Enterprise Architecture

- Scale Range: from small to large
- Scale Hierarchy: containment of small within large
- Scale Connections: connecting like to like
- Scale Tightness: connections get looser and fewer up the hierarchy.

Now that we have a better idea of what architectural coherence looks like, we can ask how these principles may be applicable to Enterprise Architecture. We have examined four requirements for architectural coherence:

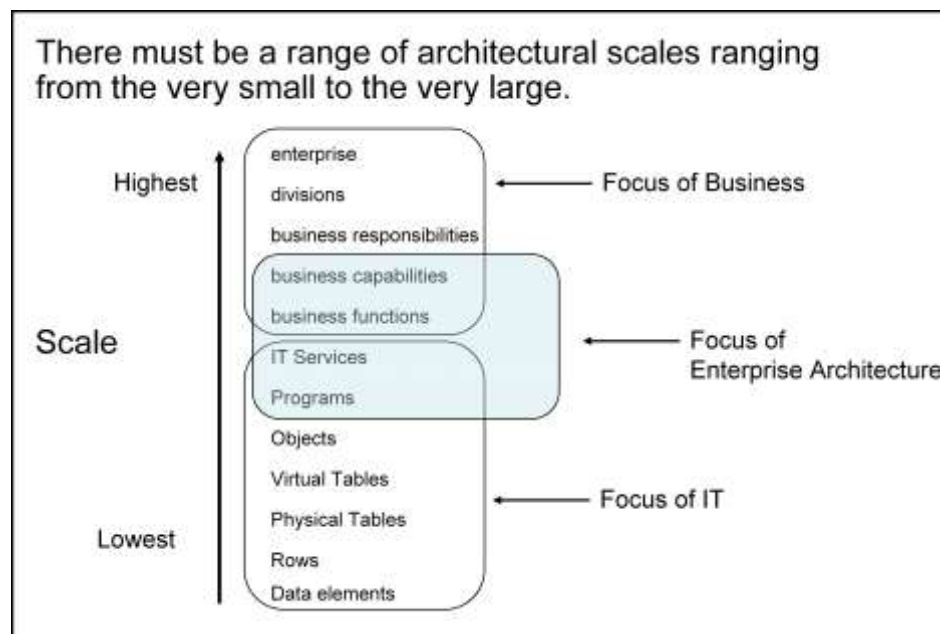
- *Scale range* requires a range of perceivable scales in the architecture. That range must extend all the way from the very small to the very large and there must exist

intermediate ranges, such that there appears to be a stepwise continuum from small to large.

- *Scale hierarchy* requires the elements at each scale to be hierarchically related to those elements at the next highest and the next lowest scales.
- *Scale connections* means that connections between elements occur within the same scale, so large elements are connected to other large elements, mid-sized elements are connected to other mid-sized elements, and small elements are connected to other small elements.
- *Scale tightness* means that there are more connections between smaller elements, and those connections are much tighter than the connections between larger elements. So as we move up the scale hierarchy (from small to large), we find fewer and fewer and weaker and weaker connections. As we saw with urban architectures, tighter connections are those that are harder to change without impacting the overall architectural stability.

Notice that we aren't saying that all enterprise architectures exhibit these four factors of architectural coherence. Far from it. Just as in urban architectures, *the overwhelming number of enterprise architectures are created without regard for (or even awareness of) what coherence is, what factors are involved, or why it is important.* These make up the bulk of those many systems that fail to deliver value to the businesses that have paid for them.

Factor 1. Scale Range



Scale range, for enterprise architectures, means that we have a range of scales that describe the system. For urban architectures, scale range meant that we could look at an architecture in detail, or in the large, or in any number of scales in between. *Scale range*

has the same meaning for enterprise architectures. We can look at these architectures in the large, that is, at the business level. We can look at these architectures in the minute, that is, the details of the data bits that are manipulated. And we can look at these architectures at any stage in between.

Note that different subsets of *scale range* are more closely associated with different groups in the enterprise. For example, the Business groups focus on how the enterprise is divided into divisions, how business responsibilities are divided up, and how business functions are organized into capabilities. The Information and Communications Technology group focuses more on how the software systems are put together and how they manipulate data. The Enterprise Architect focuses in the center scales, which connect the business and technology scales. But, unlike other groups in the enterprise, the enterprise architect needs an overall awareness of all of these scales. The enterprise architect is the glue that keeps the technical deliveries tightly focused on the business needs, and therefore is keeping an eye on all of the different architectural scales.

We should point out that there is considerable debate within the enterprise architectural community about exactly what enterprise architecture is. Some people see enterprise architecture as an all-embracing field that looks at every possible aspect of the enterprise, searching for opportunities to improve processes, relationships, or organization.

We, instead, take a much more focused perspective of enterprise architecture. Enterprise architecture is the field that ties together the business and the information and communications technology groups into a unified effort. In particular, the responsibility for producing a coherent architecture falls within the domain of enterprise architecture, because it requires a perspective that includes both the business and the information and communications systems. The enterprise architecture group owns all of the architectural coherency factors, since they all depend upon relationships that start at the business architecture and extend all the way down to the data architecture.

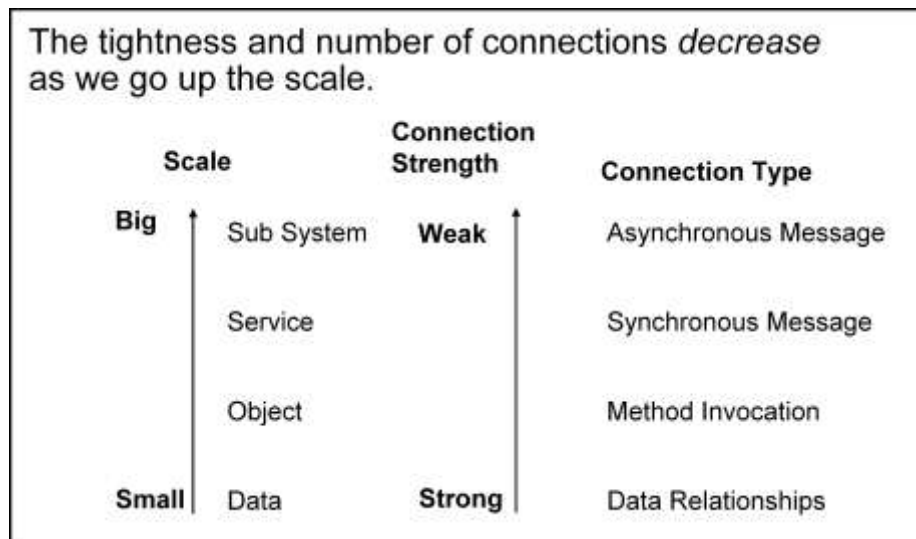
This does not mean that the enterprise architects are responsible for all the layers of the architecture. The enterprise architect, for example, does not define how the different data elements within a vertical hierarchy relate to each other. But the enterprise architect does define the boundaries that delineate what data elements are allowed to form relationships with what other data elements.

In organizations with non-existent or ineffective enterprise architecture groups, the business and information and communications technology groups tend to drift in separate directions and much of the money spent developing software systems is consequently squandered.

The third factor is *scale connections*, meaning that connections are formed at the same level of scale. Like is connected to like. Data is connected to data, objects to objects, and services to services. We don't see services connected to objects.

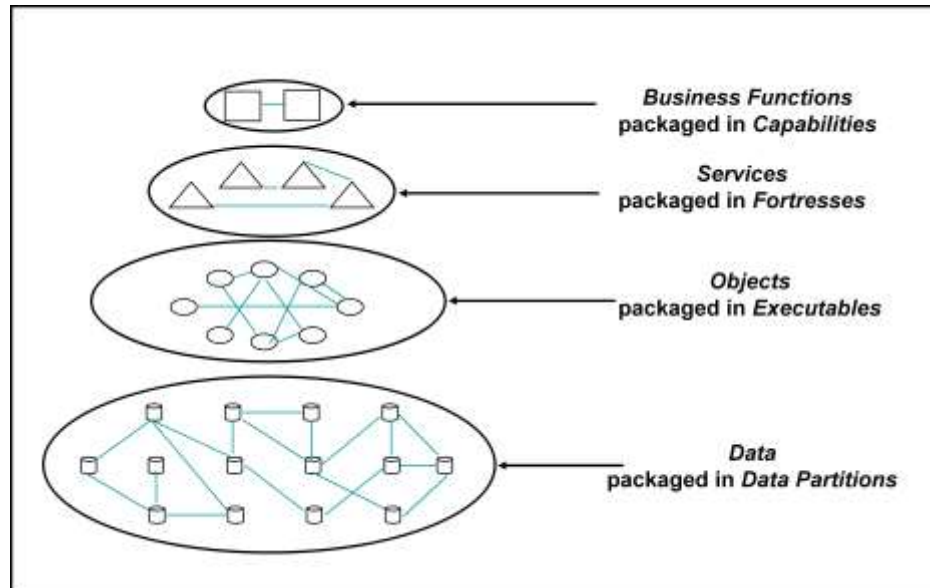
We also don't see connections traversing hierarchies. So we don't see data in one hierarchy connected to data in another hierarchy, or objects in one hierarchy connected to objects in another hierarchy. Thus, two services within a single hierarchy can be connected to each other, but two services in different hierarchies are not usually connected.

Factor 4. Scale Tightness



The fourth factor is *scale tightness*, which tells us that the number of connections and the tightness of those connections both decrease as we go up the scale. So we will see lots of data connections, for example, joined relational tables, and those connections are very tight. Tightness refers to the notion that breaking those connections is more likely to break the overall architecture. We will see fewer connections between objects, for example, method invocations, and those connections will be looser than the connections between data. Service connections are fewer and weaker yet. The weakest connections we see in software systems are found in the asynchronous message. These connections tie together autonomous subsystems. Such systems we often refer to as software fortresses, because they are autonomous, self-contained, and secure islands of software functionality.

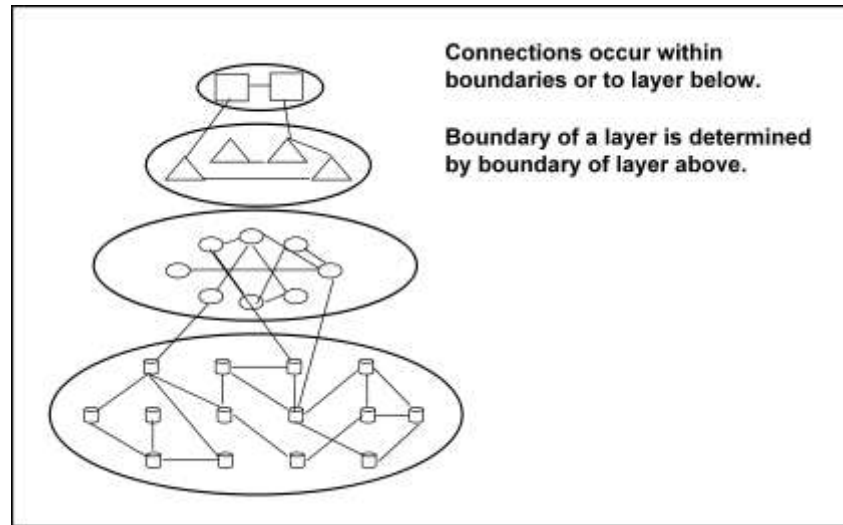
Enterprise Architecture Systems with Coherence



When we put all of the coherency factors together into a single enterprise architecture for a particular system, we get what looks like a snowman architecture, with the boundaries at each level reflecting the boundaries of the level above. It is this boundary reflection that puts this design project in the domain of enterprise architecture, because ultimately all boundaries are a reflection of the head of the snowman, representing the business capabilities.

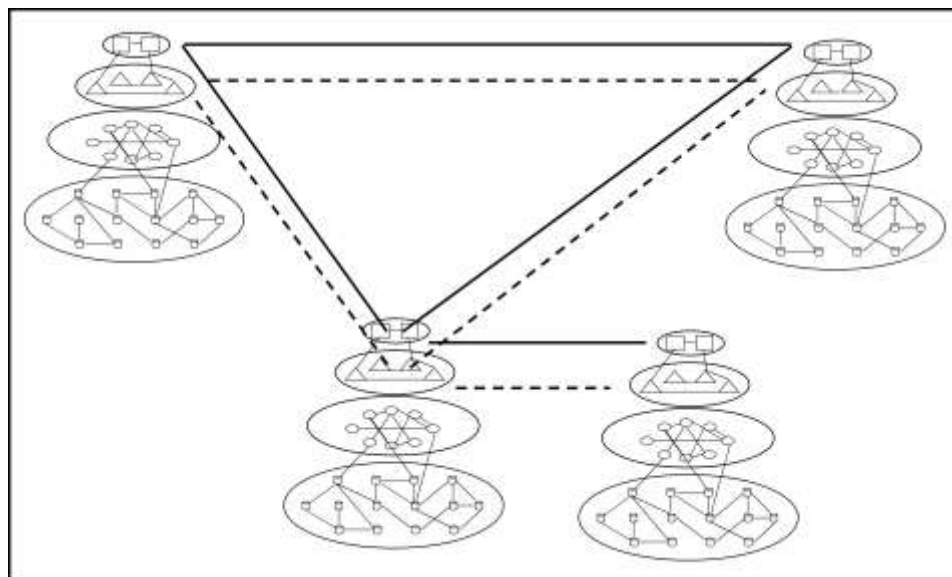
The “snowman” results naturally from accommodating the coherency factors. *Scale range*, for example, guarantees that we will have a number of balls making up the snowman’s body. *Scale hierarchy* guarantees that the balls get larger as we move from head to torso. *Scale connections* guarantee that each ball fully enwraps the elements within and that different snowmen are only connected to each other at the higher levels of the body, the level that Sessions describes as fortresses. We won’t describe the architectural concept of software fortresses here, but information on this is available in Sessions’s books *Simple Architectures for Complex Enterprises* and *Software Fortresses*.

Connection and Boundary Rules



The stability of the “snowman architecture” is a consequence of satisfying the four factors for architectural coherence. This occurs because of the predominance of connections within “balls” or scales, connecting “like to like”. The analogy to urban architectures is clear. Window panes are connected to other window panes, but not to doors, although window panes may be contained within doors.

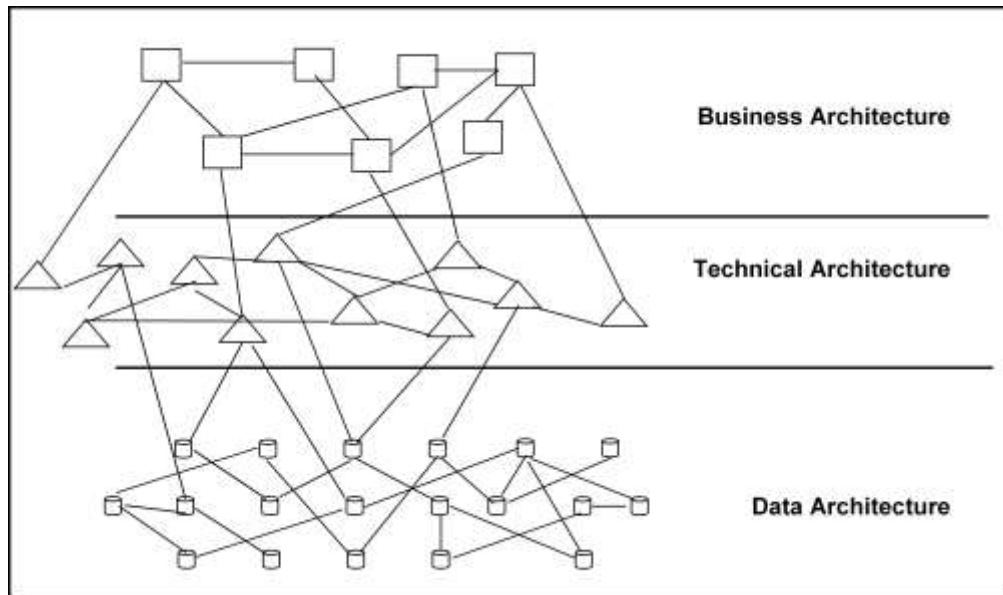
Coherent Enterprise Architecture



As we put together coherent systems into a full coherent enterprise architecture, we see groups of snowmen emerge. At the business level, they are connected by communications patterns between business capabilities, which reside at the head of the architecture. Following the scale connections coherency factor these connections are characteristically the weakest of all connections,. At the technical level, these connections are mirrored at

the highest level of the technical architecture, the software fortress level. Because of the restriction of scale connections, we see no other connections between snowmen at lower levels of the technical architecture. Of course, we see many connections of like to like within layers of the snowmen, but these connections stay confined within the boundaries of that layer.

Classic Enterprise Architecture



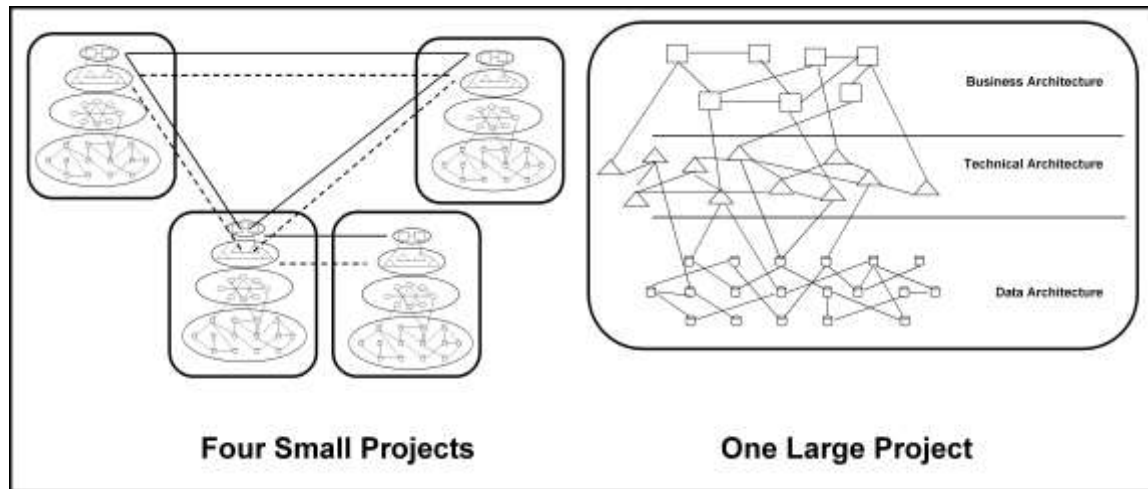
The “snowman” architecture that characterizes a coherent enterprise architecture is quite different than what we see in a classic enterprise architecture, shown above. The classic enterprise architecture has layers, like the coherent enterprise architecture, but that is where the similarities stop.

The classic enterprise architecture starts with the business architecture. This drives a technical architecture, in that every requirement of the business architecture must be supplied by the technical architecture. However, these layers are designed independently of each other. The only influence a layer has on the layer below is that each layer defines requirements that must be supplied by the layer below. But there is no coherence between the different layers, or even within layers.

The result of this is that there is no design relationship between the different layers and no rules that define how elements in one layer are related to elements in another layer. So, for example, there are no rules about how many elements there should be in the technical layer relative to the number of elements in the business layer, or which elements in the technical layer are allowed to connect with other elements in the technical layer, or what the strength of those connections should be. The result is an architecture that resembles an amorphous bowl of spaghetti more than an orderly gathering of snowmen.

Part IV. Importance of Coherence

Coherent Architectures: Easier to Implement



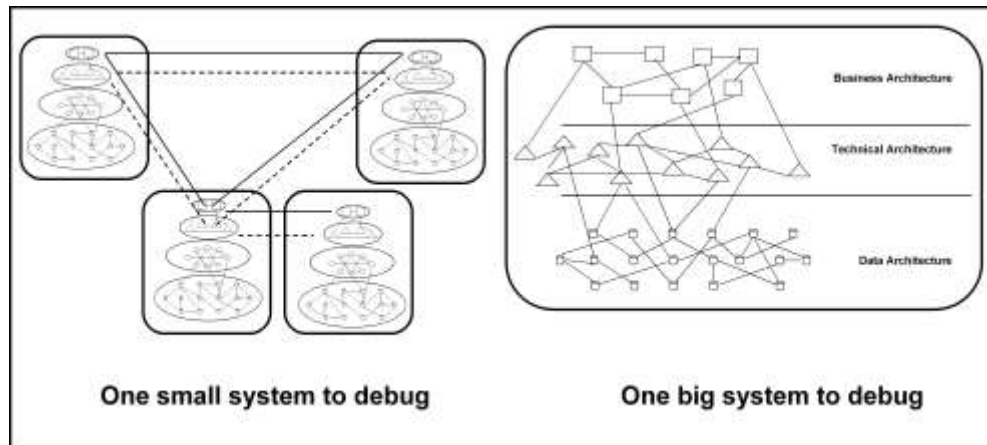
Comparing the coherent enterprise architecture to the classic enterprise architecture, we can immediately see why the former is so much easier to implement than the latter. In the classic enterprise architecture, a single, highly entangled project is being implemented. Such projects are highly complex, usually fail, and even when they succeed they result in highly fragile (and precarious) systems.

The coherent enterprise architecture, by contrast, is tackled as a number of smaller, simpler, autonomous projects with a few well-defined connections (or dependencies). Such projects are much easier to understand and have much higher success rates. The reason the connections are considerably fewer in the coherent enterprise architecture is that they occur much higher up in the hierarchy, at the highest level of the technical architecture. Not only are there very few connections, but these connections are quite loose, meaning that changes on either side of the connection have minimal impact on the architecture as a whole. This is one of the important reasons such architectures are so stable.

Comparing the two architectures (classic and coherent) side by side shows the value of an intelligently guided hierarchy, in which each scale of the architecture relates in a clear manner to the next scale up and down. In the classic architecture used for most of today's highly complex (and not so complex) systems, the number of scales is traditionally fixed at three (business, technical, and data) regardless of the overall size of the project, with connections between elements at different scales done haphazardly. Furthermore, there is no relationship between levels of scale. For large projects, the overall complexity of the system overwhelms the number of scales available to it, just as in urban architecture, the size of the John Hancock Tower (shown earlier) overwhelmed the number of scales available to it. This complexity makes the system impossible to understand, and whatever we cannot understand, we cannot implement.

In the coherent architecture, the number of levels of scale is dictated by the size of the project. Small systems may have two levels of scale. Large systems may have a dozen or more. And just as important as the number of scales is the set of relationships between those scales.

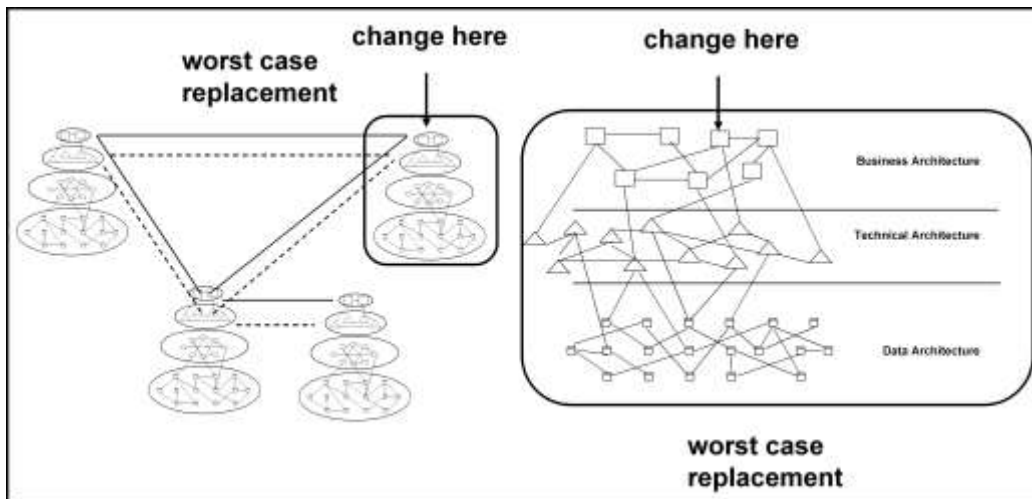
Coherent Architectures: Easier to Maintain



In the preceding discussion, we focused on the complexity (and therefore the cost) of building a system. But maintenance costs are equally important. Most large software systems cost more to maintain over their lifetime than they cost to build in the first place. So it is critical that our approach to designing these systems drives towards systems that are as easy (and therefore cheap) to maintain as possible.

In this area as well, coherent enterprise architectures are far superior to classic enterprise architectures. Since the systems are much smaller and the dependencies so much better defined, debugging the system becomes much easier. One is effectively debugging each single small simple system at a time rather than a massive large complex system. Which would you prefer to debug?

Coherent Architectures: Easier to Adapt



Adaptability refers to the ease with which the business can make changes in its systems to respond to changing conditions. Today's business world must be able to adapt at a pace unthinkable even a decade or two ago. Coherence is a critically important enabler for adaptability.

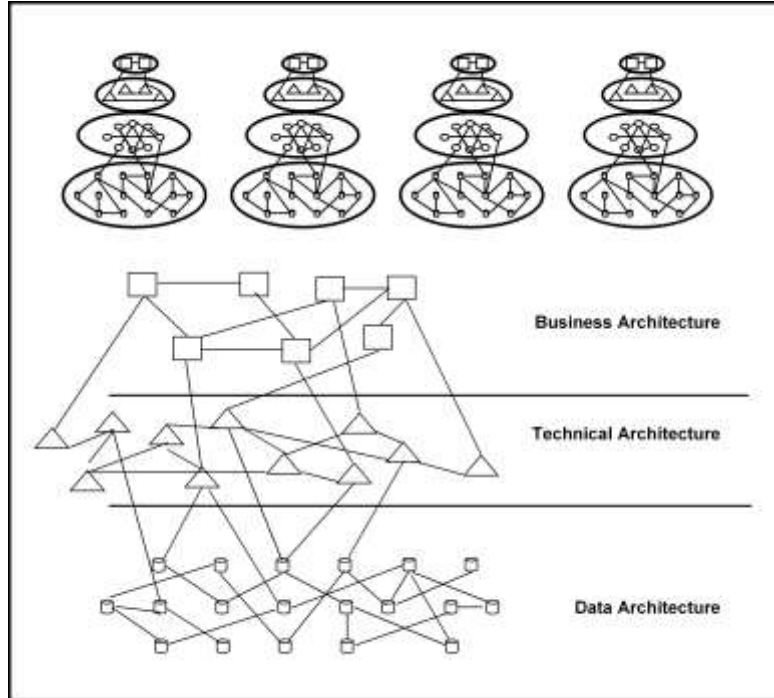
Business changes occur at the highest levels of the enterprise architecture. These business changes are then followed by changes at lower architectural levels, that is, lower balls, in the coherent architecture, or in the technical and/or data level, in the classic enterprise architecture.

For the classic enterprise architecture, these necessary technical changes become highly problematical to implement. Because of the haphazard approach to making connections, system dependencies are widespread and unpredictable. Changes in one small area of the technical architecture can break other areas of the technical architecture that seem at first glance to be completely unrelated. Changes in the data architecture are equally unpredictable. The result of this unpredictability to the effects of system interventions is that seemingly small changes at the business level can result in hugely expensive disruptions at the technical and data levels.

In the coherent enterprise architecture, such technical changes are much easier to implement. Most dependencies are bounded by the contours of the snowman, so even in the worst case, changes in the business architecture will force a replacement of an entire snowman, but are highly unlikely to cascade into other snowmen.

This discussion reveals the reasons why coherent enterprise architectures are far more adaptable than their classic brethren.

Coherent Architectures: Less Complex



The SIP methodology developed by Sessions includes mathematical approaches to measuring complexity. Using these approaches, we can prove that the coherent enterprise architecture is much less complex than the classic enterprise architecture. For a large enterprise architecture, the complexity reduction can easily be an order of magnitude. But it isn't necessary to do a methodical mathematical analysis to see the complexity reduction. A quick visual inspection of the two distinct approaches to enterprise architecture makes the complexity reduction readily apparent.

Summarizing Architectural Coherency

Architectural coherency crosses at least two domains.

A coherent EA is a better EA.

A coherent EA follows these rules:

The architecture has a range of scales, from data to business capabilities.

The levels of scale are arranged hierarchically.

Connections occur within a level and within boundary.

Connections become looser and fewer as you move up the hierarchy.

We have seen that the concept of architectural coherency crosses two different domains: *i*) enterprise architecture (EA), with its focus on business systems and information and communications technologies, and *ii*) urban architecture, with its focus on buildings and livability. These two domains do not at first seem closely related, and yet we find common rules across both. We can't help but wonder how many other domains will also find these rules relevant. Salingaros, following Alexander, claims that the rules governing architectural coherence turn out to be universal laws that govern many domains of learning and understanding.

Who Cares About This?

You do!

If you want systems that are cost effective.

If you want systems that are maintainable.

If you want systems that are maintainable.

The laws of architectural coherence are not just an intellectual exercise. In the enterprise architecture domain, these laws have pragmatic implications and deliver measurable business value. Coherent enterprise architectures are better than classic enterprise architectures in a number of important ways. A coherent enterprise architecture is simpler, more cost effective, easier to maintain, and adapts much more readily to changing needs.

Who violates these Rules?

Most enterprise architectures

They are a layered architecture, but the layers are not hierarchically related.

Most data architectures

They typically share data across a large swatch of functionality.

Most SaaS Systems

They lack the range of scale.

Most SOAs

They reuse services across multiple systems.

It might at first seem like the rules governing coherent enterprise architectures are obvious. Wouldn't you think that any enterprise architecture would follow these rules? It turns out that this is not the case, and very few enterprise architectures follow these rules. One example we have already discussed is the classic enterprise architecture, with its bowl of spaghetti motif and poor boundary definition. The architecture discussed here has a number of layers (three, to be exact), and there is no hierarchical relationship between these layers. But there are many other common examples of incoherent (that is, non-coherent) enterprise architectures as well.

Another common architecture that is incoherent is the standard approach to data architecture. Many organizations believe that there should be an organization-wide data architecture that is used by systems. The data architecture is typically highly interconnected and has no hierarchical relationship with other levels of the architecture. This makes the requirement that connections occur only within a hierarchical group completely meaningless.

We are seeing an increasing focus on Software as a Service systems, that is, cloud-based turnkey systems that provide prebuilt functionality and browser based user interfaces. These systems are often large, monolithic systems that offer few if any hierarchical scales. They are the software equivalent to the Houston Museum of Modern Art (discussed earlier), a large system you plop in place, and offering no insight into what happens on the other side of the imposing (and opaque) facade.

Service-oriented Architectures seem perfectly adapted to the concept of architectural coherence introduced here. They offer a nice technical boundary that can easily encompass multiple lower scales of a hierarchy. And they themselves can fit naturally underneath the business architecture. The problem with service-oriented architectures, however, is that they typically give a heavy emphasis to reusability. So it is common in service-oriented architectures to find generic services that promise to support a wide range of other systems. This focus on reusability creates a problem for architectural coherency. The almost maniacal message passing that is required to achieve this reusability totally erodes the clean boundaries required for coherency. Ironically, significant reusability is rarely achieved because of the difficulty in prognosticating the needs of future systems and also because of the unexpectedly high cost of developing generic functionality. In sacrificing coherency for reuse, we therefore end up with systems that provide neither.

Part V. Wrap-up

Where does SIP fit in?

Simple Iterative Partitions

A methodology for creating a Coherent Enterprise Architecture.

Not just any CEA, but the simplest possible CEA.

And we can prove it!

In a coherent enterprise architecture, one of the most critical pieces to get right is the business architecture, i.e., the head of the snowman. The functionality in the head will dictate the boundaries of each of the lower levels of hierarchy. But how does one determine the optimal configuration of functionality at this critical layer?

This is the problem that SIP attacks. SIP stands for *Simple Iterative Partitions* and is a methodology for determining the *optimal* coherent enterprise architecture given the specific goals and structural organization of a given enterprise. By *optimal* we mean the simplest possible coherent enterprise architecture that solves the problem at hand. Further, since SIP is based on mathematical models for complexity and coherency, the architecture delivered by SIP can be subjected to a rigorous validation analysis *before* the organization goes through the considerable expense of actually implementing the architecture.

SIP is an innovative approach to the problem of driving the optimal coherent enterprise architecture. In fact, it is so novel, it was granted a broad patent by the U.S. Patent and Trademark Office covering a wide range of claims in the area of enterprise architectural optimization. For those interested in learning more about SIP, or for information on the patent covering the SIP methodology, there is ample material freely available at Sessions's website, www.objectwatch.com.

SIP of course cannot remove every ounce of complexity. There is always a minimum amount of complexity needed to solve a given problem. For example, if we are designing a transportation system, a rock is simpler than a bus. The only problem with the rock is that it doesn't solve the problem. SIP doesn't guarantee to remove all complexity; only the extra complexity that detracts from the functions of the enterprise, which in themselves are considerably complex. SIP can't even guarantee to deliver a solution that seems simple. All that SIP can guarantee is that the SIP solution will be coherent and will be the simplest possible coherent solution that solves the problem.

Discussion

Salingeros: I would like to define the concept of hierarchy a little more precisely. There are many definitions of this word. One that is particularly negative is when that implies a

top-down command structure. This is not what we are talking about. We are talking about connections that can go either vertically upwards or vertically downwards. We can accommodate top-down command structures but we can also have a bottom-up structure. So we want to be sure that we are not interpreting hierarchy in a political way. Peer to peer structures are essentially bottom up. These connections have tremendous potential for many types of problems. So you can see that these ideas extend far beyond enterprise architecture.

We need to understand that we are not imposing our own notion of hierarchies of scale. We are discovering those hierarchies of scale that are inherent in the complex system we are trying to understand. This is also true for the connections. We don't impose our belief system on the connections, we discover the connections that are implicit (but perhaps not manifest) in the system itself. These connections may be vertical (within the hierarchy) or horizontal (within the boundaries of the hierarchy). There is a heuristic manner of defining the system and how it works and we don't want to come to the analysis with any prejudices.

This brings us back to Christopher Alexander's work on Patterns. The idea with patterns is that something has worked and we would like to apply it again, but we want the design of the system to be guided by the needs of the system itself.

Question: Are you going to mention that refactoring is a series of structure-preserving transformations, to use Alexander's terminology, that it does allow you to gradually turn the messy system into one with coherency?

Salingeros: If we can achieve a better understanding of how to apply Alexander's work on structure-preserving transformations, we could achieve a breakthrough in addressing complexity, especially in the field of enterprise architecture. Alexander's recent book, *The Nature of Order*, is a wealth of fantastic ideas. I certainly encourage everybody listening to this webinar or reading this paper to try to apply Alexander's transformations and get back to Roger and me and we would love to collaborate. It's opening up a tremendous world of opportunity that could well revolutionize enterprise architecture, not to mention urban architecture.

Question: Do you have examples in which you have re-factored an existing enterprise into snowmen?

Sessions: We have done a number of re-factorings with the SIP methodology, which is very much focused on finding not just a *coherent* enterprise architecture but the *optimal coherent* enterprise architecture. I won't try to discuss SIP here, but I do have a number of white papers on the subject at the ObjectWatch web site.

Question: Does Business to Business integration always occur at the head of the snowmen? What if you are using a lower level service, say, for example, a postal code check?

Sessions: With all rules, we need to realize that there could be exceptions. However we want to be very careful with any exceptions, because any embrace of exceptions weakens the coherency of the overall architecture. I generally believe that almost all integrations of this type should occur at the highest levels of the technical architecture. In the SIP methodology, these types of interactions (the postal code lookup) occur at the level of the software fortress. I discuss software fortresses in my recent book, *Simple Architectures for Complex Enterprises*.

Question: You have drawn a comparison between those rules that make an architecture pretty and those rules that make an enterprise architecture functional. What is your basis for making this correlation?

Sessions: For me, this is based on observations. As I have looked over a large number of enterprise architectures and tried to understand why some stand out as good architectures, a few characteristics emerge. The primary feature that emerges is simplicity, which explains why SIP focuses so much on this issue. But I noticed that as we look at the characteristics of these simple architectures, they seem to share a number of common properties. As I read Nikos's work on urban architecture, I discovered that these common properties could be described using the terminology Nikos introduced for urban architecture.

Salingaros: Yes, this requires a very careful answer. I will start with the basic premise that there are universal properties that have nothing to do with either enterprise or urban architectures. These universal properties are obeyed by all stable complex systems. It turns out that buildings that obey these universal properties are not necessarily esthetically appealing, but actually have a positive impact on our health, on our physiology. While I explained this earlier, it is worth repeating.

The reason we react positively in a physiological way to buildings that have these properties is that we evolved in an environment that exhibited these same properties. Nature is filled with systems that obey exactly the rules that we have been discussing. Enterprise Architecture, it is true, exists in a totally abstract realm, but it is still a complex system. As a complex system, we expect that the degree of stability the system has is directly related to how closely it obeys these natural laws for coherency.

That is the basis for the equivalence that we propose. This is not *our* premise. This is the basic premise of those working in complex systems since the early days of Herbert Simon. There are universal geometrical and mathematical properties that apply to complex systems regardless of the domain. Experimentally, we know that this is the case, not only for urban architectures and enterprise architectures but for all complex systems including biological and artificial.

Question: Most architects inherit their architectures. How do you take an existing incoherent architecture and turn it into a coherent architecture?

Sessions: When you have an existing architecture, you need to look at the entire architecture and see where you can have the most impact for the least effort. Typically you find complexity hot spots and these usually are areas in which there is a particular poverty of coherence. Sometimes relatively little effort in these areas can make a big difference. In the ideal world, you would be building the architecture from the ground up with coherency in mind. But you don't always have that luxury. It is a similar problem to what Nikos I'm sure faces. He would prefer to be able to build the building from the beginning with coherency, but sometimes he must do the best he can with existing architectures.

Salingaros: Let me describe the repair process that my group of architects/urbanists use when we need to fix something that is wrong. We start by trying to imagine what it would look like to have something totally new on the site. So we walk around the site, engaging our physical senses. We then try to see which aspects of the existing architecture give some positive feedback, which of them are neutral, which give negative feedback, and which feel totally deadly just to be near them. Unfortunately, many contemporary buildings fall in this last category. You can actually measure the anxiety these buildings cause by measuring physiological responses such as pupil dilation, skin temperature, and changes in the electrocardiogram. Then we sit down and try to figure out how to get rid of the worst parts of this urban environment and replace it, if possible. If you have the John Hancock tower, you obviously can't propose tearing down the whole tower, at least not until the end of its material life. So we make a series of compromises and go in and try to change the worst parts, which really corresponds to what Roger said. We try to fix the worst parts and leave those that are working okay. We are essentially doing very delicate microsurgery, but at the urban scale. Like Roger, we are trying to get the biggest bang for the smallest investment. I realize this is a very general statement, but I think any listener can draw the analogy with enterprise architecture.

Salingaros: I see a comment with respect to Richard Gabriel, a software pioneer, who collaborated with Alexander and has written a book about Christopher's work. I don't see a question, but I do recommend Richard Gabriel's book, which is a good read.

Question: Can you give some example of how service-oriented architectures violate the principles of architectural coherency?

Sessions: Service-oriented architectures don't have to violate the principles of architectural coherency. Service-oriented architectures can be models of architectural coherency. It's just that most aren't. There are many reasons why this is so. First of all, there is usually little relationship between the business architecture and that of the service-oriented architecture. There is certainly no relationship that is expressed hierarchically. Second, most architects are so focused on the elusive goal of reuse that they give up any hope of coherency. It is possible to achieve reuse and coherency, but it takes a much better understanding of the architectural issues of reuse than we usually see. But more typically we see architectures that sacrifice coherency in an attempt to achieve reuse and end up with neither reuse nor coherency. Now if a service-oriented architecture

is designed from the beginning with coherency as a goal, then service-oriented architectures can be a great way of achieving highly coherent architectures.

Sessions: Nikos, would you like to offer a wrap-up?

Salingaros: Yes, what we have seen is the universality of some principles that can be applied in a very specific manner and something that you, Roger, did not emphasize. You did not emphasize the difference between what is done today, is accepted practice, and what we recommend as a better, more efficient practice.

Let me start by emphasizing what I do. It is really shocking that the principles that we have outlined in this talk, which would allow us to build better buildings are routinely violated to build buildings that have shock value, that seem superficially interesting. This has led to a general deterioration of our cities. It does help tourism, since people must now go to considerable expense and travel great distances to find urban environments that feed our deepest, intuitive biological needs. We had here in the United States something wonderful that we just destroyed by imposing geometries that don't work and they don't work specifically because they violate the principles that we have outlined here for enterprise architecture.

Roger and I have been talking for a while now and he is telling me exactly the same thing: that wrong principles of enterprise architecture are routinely implemented by large corporations and the resulting systems are lousy, non-functional, fragile, and far more expensive than they need to be. And yet these bad practices are just repeated because there is an inertia in the system. People continue using the same approaches to enterprise architecture that they know, even though these systems keep delivering the same failed results in terms of systems delivered late, over budget, and missing fundamental functionality.

So we have these parallels in both enterprise architecture and building architecture where methods don't work, yet the people who make the major decisions keep repeating them. This webinar, then, is a call for sanity and logic. And since we offer an explicit set of guidelines and rules and explain exactly why these work, there is no excuse due to any lack of information. When we go to Roger's white papers, we see that yes, indeed, when applied to enterprise architecture these techniques do work. They give you a very efficient enterprise architecture that can be delivered at one tenth of the cost of the alternative that is huge, dysfunctional, and doesn't work.

The other parallel, of course, is with urbanism. When we build an urban environment following these rules people love it, housing prices go up, and yet, nearby we build the same giant buildings and suburban sprawl that is energy consuming and dysfunctional.

People follow the same failed models. There seems to be a terror in not continuing to do what has been done in the immediate past even though we know that what we are doing is highly deficient.

So Roger is too nice a guy to point this out. But I hope that the client, namely the corporation paying for the enterprise architecture will now think very logically in terms of choosing which type of enterprise architecture to design. And they should think in terms of the very concrete proposals that we have made here and developed further in Roger's white papers.

Sessions: Thank you, Nikos, for being part of this. It is always a great pleasure to be with you and thanks to all those who participated in this webinar.

For more Information

Both Salingaros and Sessions are members of the Consortium for Untangling Enterprise Complexity, a new group of people trying to work together to better understand ways of removing unnecessary complexity in enterprises. For information on the Consortium, go to www.cuec.info.

Sessions writes many white papers on the subject of enterprise complexity and SIP and both Salingaros and Sessions expect to continue this fascinating collaboration. If you send email to roger (at) objectwatch.com, he will add you to the notification list. You can also follow him on twitter (@RSessions). All of the Sessions white papers can be found at www.objectwatch.com.

SIP

- Reduce Complexity
- Develop Better Solutions
- Lower Costs

To leverage Roger Sessions' patented SIP™ methodology in your organization, email us: information@objectwatch.com

Thriving in a complex world is possible.
Let us show you the way.

Your consulting organization can become a SIP partner. Let's talk!

Legal Notices

ObjectWatch is a registered Trademark of ObjectWatch, Inc. This work is licensed under the Creative Commons Attribution 3.0 United States License. To view a copy of this license, visit <http://creativecommons.org/licenses/by/3.0/us/> or sent a letter to Creative Commons, 171 Second St. Suite 300, San Francisco, California, 94105, USA. Attribution of this work should be as follows: Used with permission from Roger Sessions and Nikos Salingaros. This work can be copied and redistributed as long as it is kept intact and no deletions, additions, or modifications are made without permission.

To Join the Discussion

Would you like to discuss this white paper? You may post comments and questions to the authors at this blog:

SimpleArchitectures.blogspot.com/2010/10/discussing-complexity.html.

If you would like to discuss this on twitter, use the hash tag #archcomplx.