White Paper

Enterprise Architecture meets Soft Systems - Paper 1

A Brief History of Soft Systems and why it matters for Enterprise Architects

WP0126 | January 2014

Ceri Williams

Ceri has thirty years in the IT industry, originally delivering complex control systems and subsequently broadening focus to Enterprise Architecture, Governance and transformation of the IT function. Working as a chief architect, consultant and coach, he enables FTSE 250 organizations to make medium and long term decisions on the shape of the Enterprise Architecture and positioning of the IT function.

He advocates putting people at the heart of technology and business change with focus on the human enablers and constraints. His work deals with the way in which rigorous engineering and architecture disciplines are integrated with the cognitive and behavioural capabilities of the people who practice them.

My previous set of White Papers on The Art of Judgment applied the ideas of Sir Geoffrey Vickers to Enterprise Architecture [Ref 1]. In particular, they considered the way in which we construct our reality and how the act of constructing that view changes our perception of what it is. They also considered the key areas where Judgment is exercised: in constructing the reality (what is and is not the case), the values we apply to it (what ought or ought not be) and what we decide to do about it. The natural link between Vickers' ideas, Systems Thinking [Ref 2] in general, and Soft Systems in particular, is the way in which we deal with a key challenge: how we construct our understanding of a problem and the way in which that understanding changes with every attempt to solve it – the core feature of a Wicked Problem [Ref 3].

Enterprise Architecture centers on the defining and resolution of Wicked Problems – these problems are difficult (or impossible) to solve because of incomplete, inconsistent and unstable requirements with complex interdependencies that are often difficult to recognize. For an Enterprise Architect, differentiating between difficult and impossible is a critical act of judgment. However, traditionally the methods and tools available to the Enterprise Architect are derived from Systems Engineering as originally conceived at Bell Telephone Laboratories in the first half of the 20th Century. The use of a Soft Systems approach equips the Enterprise Architect to effectively deal with a broader range of more complex problems - enabling more impossible problems to be seen as 'just' difficult.

Traditional Systems Engineering approaches work well on deterministic problems on a single system scale to deliver a new Information System. While they are necessary, they are not sufficient to deal with the complex challenges that Wicked Problems present – these are bread and butter for Enterprise Architects.

This Series of White Papers

This series of white papers explores the value of integrating Systems Engineering into the broader Soft Systems world in the context of Enterprise Architecture. It proposes the practical behaviors and values that Enterprise Architects can adopt in order to take their practice beyond the engineering domain and enhance their effectiveness. This first paper provides an overview of Soft Systems Method, its positioning alongside Systems Engineering Methods, and their significance for Enterprise Architects. Each of the subsequent seven papers covers a key element of the Soft Systems Method and its application to Enterprise Architecture:

[Paper 2]: The limits of an engineering approach to Enterprise Architecture: what Engineering is and is not good at. Contrary to popular belief, the Enterprise is not a complex engineering object. Complex, yes. Engineering, no. This paper explores the limits of adopting an Engineering-intensive approach to EA and how integrating it with Soft Systems provide a comprehensive set of tools for the vast majority of situations.

[Paper 3]: How the Enterprise Architect can recognize and respond to the Softs Systems challenge. This paper considers how the Enterprise Architect can recognize the situations when Soft Systems methods are likely to be helpful and when an Engineering approach may be more appropriate. It considers how Enterprise Architects can weave the ideas of Soft Systems methods into their business-as-usual practice without confusing their stakeholders.

[Paper 4]: Concepts, abstractions & simplification: modeling, but not as we know it. Most frameworks and visual languages name concepts such as ‘Conceptual Model’, ‘Logical Model’, or ‘Service Model’ without defining them in a way that can be put consistently and easily into practice by multiple Enterprise Architects over an extended time. This paper encourages the Enterprise Architect to be flexible, but clear about these definitions, and design the Meta-Model to be fit for purpose. It helps answer questions such as “how do we know when we’ve finished?” and “how can we describe this to be re-usable?”

[Paper 5]: Perspectives and viewpoints: choosing your window on the world. IEEE 1471 is the DNA of viewpoints, affording the Enterprise Architect with a systematic means to make the Architecture

intelligible. A key principle of a related discipline, Neuro-Linguistic Programming (NLP) is that “the meaning of the communication is the response that you get (not the one intended)”. This paper considers how Enterprise Architects are able to put themselves into the shoes of the stakeholders, and adjust their language, concepts, content and presentation to communicate effectively. It considers the strengths and weaknesses of the key IEE 1471 concepts of ‘constructed’ and ‘projected’ views as a way of representing multiple inconsistent realities in a way that can still be effective in a soft-systems context.

[Paper 6]: Mind the gap - the model vs. the real world. Polish-American scientist and philosopher Alfred Korzybski remarked that “the map is not the territory”, encapsulating his view that an abstraction derived from something, or a reaction to it, is not the thing itself. Korzybski held that many people do confuse maps with territories - that is, confuse models of reality with reality itself. This pitfall is particularly relevant for anyone practicing an approach that is based on an Engineering discipline. This paper considers these pitfalls and offers practical advice on how to avoid them by integrating with Soft Systems methods.

[Paper 7]: The role of creativity, instinct, intuition and experience. Most Methods and Frameworks are focused on structures, concepts and procedures, but without the very personal capabilities of creativity, instinct, intuition and experience, no method can deliver value. This paper considers how Enterprise Architects can leverage their personal capabilities and integrate them with more methodological constraints. It discusses how creativity can be systematic, and the conditions that recognize and exploit instinct and intuition.

[Paper 8]: Integrating Engineering, EA and Soft Systems Methods - adapting and leading with a Soft Systems approach. Summarizing the key elements of the previous papers, this paper proposes how Enterprise Architects can blend the best features of Soft Systems and Hard Systems to equip them for the real world that is only sometimes predictable and rational.

A (very) Short History of Soft Systems

The first lines of the Wikipedia entry covering Soft Systems [Ref 4] reads: “Soft systems methodology (SSM) is a systemic approach for tackling real-world problematic situations.” Soft Systems provide a framework for users to deal with the kind of messy problem situations that lack a formal problem definition. Enterprise Architecture deals with “real-world problematic situations” and routinely encounters “messy problem situations that lack a formal problem definition” – this is why a re-imagining of Enterprise Architecture as a blend of Soft Systems and Systems Engineering disciplines is now needed, and provides us with a complete set of concepts and tools with which to operate in a complex, people-centric environment.

The Soft Systems Methodology originally emerged in the 1960s in response to problems encountered in tackling management and organizational problems using a systems engineering approach. Again, from the Wikipedia entry: “The team found that Systems Engineering, which was a methodology so far only used for dealing with technical problems, proved very difficult to apply in real world management problem situations. This was especially so because the approach assumed the existence of a formal problem definition. However, it was found that such a unitary definition of what constitutes ‘the problem’ was often missing in organizational problem situations, where different stakeholders often have very divergent views on what constitutes ‘the problem’”. I would add that the Systems Engineering approach also makes a number of (usually unstated) assumptions. Specifically that:

1. The problem and solution space can be modeled as a single definitive version of ‘the truth’ that is common to all stakeholders
2. The environment (the world!) can be baselined to facilitate analysis and does not move on faster than the baseline and the problem solving work depending on can react
3. The time taken to assemble the baseline and develop a solution is short enough that the solution is relevant and valuable at the time it is implemented

Every movement has its gurus, and Soft Systems is no exception. The first mainstream work to encode and specialize the knowledge around Soft Systems centered around Lancaster University, UK in the mid-1960s pioneered by Prof Gwilym Jenkins and subsequently by Dr. Brian Wilson, before reaching the mass market through the work of Prof. Peter Checkland. A number of useful references are included at the end of this White Paper.

Despite the name, the Soft Systems Method does not differentiate between ‘Soft’ and ‘Hard’ systems. It does not even treat ‘Hard’ and

'Soft' as features of the problem under consideration – they are features of the relationship between the problem and the person interested in it. They relate to the way in which the problem analyst perceives and interacts with the situation. For this reason it provides the best reference point for Enterprise Architecture and an inclusive, systematic framework for integrating Engineering and Soft Systems approaches. For the sake of clarity in this series of papers, provided we accept that we construct our viewpoint to represent a 'system' and that 'Hard' and 'Soft' are not intrinsic to the system, we shall refer to 'Hard' and 'Soft' Systems.

For further reading and a very concise and complete account, see Ref [5].

Key Concepts

For the purpose of this series of white papers and in line with the general consensus in the field, Soft Systems and Hard Systems are treated as views of a system, rather than features of the system itself. Hard Systems are generally well suited to treatment with a Systems Engineering approach, Soft Systems with Soft Systems methods.

These viewpoints can be differentiated as follows:

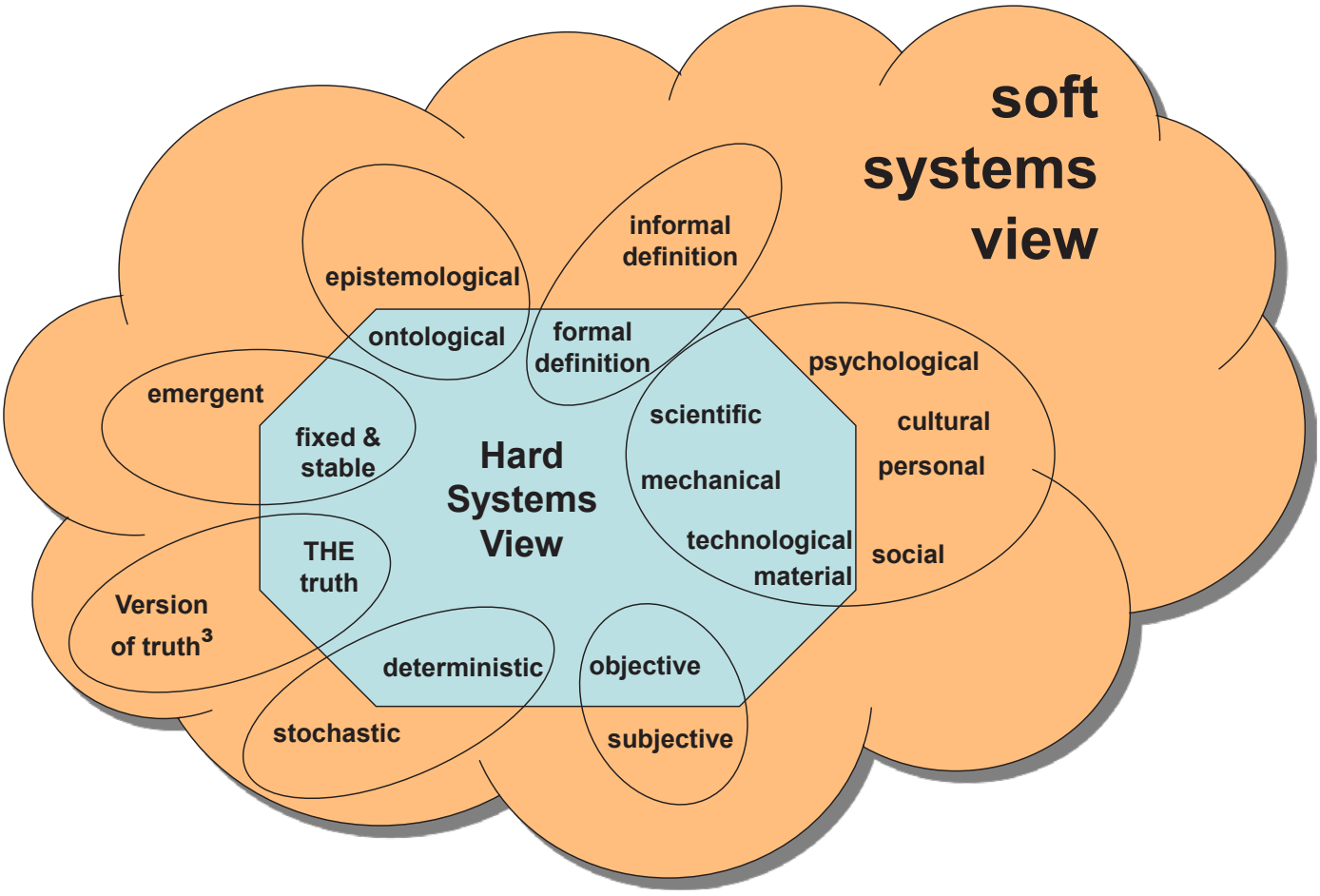

Figure 1: The Relationship between Soft and Hard System viewpoints

Soft System View	Hard System View
Inclusive of scientific, technological, mechanical, material, psychological, social and cultural domains.	Inclusive of scientific, technological, mechanical, material domains. Exclusive of psychological, social and cultural domains.
Accepts that Systems develop emergent properties that cannot be foreseen at the outset. Provides concepts and tools to cater for this.	Assumes fixed and defined System and environment in which it operates. Unanticipated changes to either require re-entry into the Systems Engineering process at some point.
Provides the ability to integrate Systems that exhibit features and behavior that may be random, stochastic (i.e. statistical) and deterministic (i.e. individual cases predictable by analysis).	Deals effectively with deterministic systems and environments in which they exist. Has limited ability to deal with stochastic systems.
Tolerant and accepting of subjectivity and multiple 'versions of the truth'. Treats all models as viewpoints that express how stakeholders perceive the system. Accepting of dissonant and inconsistent viewpoints.	Considers multiple viewpoints as filtered views of a single, objective, canonical definition of a system or problem. Assumes and requires common agreement across all stakeholders, convergence and consistency of viewpoints.
Conceives of 'System' as an epistemological entity – i.e. as made up of conceptual and mental schemas and models that determine the perception of what the system is. Considers the perceptual schemas are an integral part of the 'system'.	Conceives of 'System' as made up of ontological entities – i.e. representation of, or actual entities physically existing or proposed to exist in the real world. The 'system' is independent of the way in which it is described.
Integrates Systems and problems that can and cannot be represented by formal definitions. Formal definition may not be possible either because of the nature of the System or because there is no suitable formal language with which to describe it.	Requires that problems and Systems can be represented by formal definitions (i.e. having conventionally recognized form, structure or set of rules). Assumes that they are structured, well-formed and logical.
Recognizes the significance of stakeholder values and world views (Weltanschauung) and their impact on the scope and shape of the System.	Recognizes stakeholder values and world views only to the extent that they filter the information that represents the system and separates stakeholder concerns.

Table 1: Differences

The strong message from these differentiating features is that Hard Systems are an integrated subset of Soft Systems. This integration is even more apparent when the similarities between Soft System Methods and Systems Engineering are considered:

1. Reliance on some form of Conceptual Modeling: modeling is an integral part of both approaches. Models are used to explore and define concepts and as a means of capturing and communicating between stakeholders. Modeling is used as much to articulate the problem as define the solution. Systems Engineering typically uses (or aspires to use) more rigorous languages (e.g. ArchiMate® for Architecture), trading off inclusion (e.g. of non-Architect stakeholders) for rigor in specification. Soft Systems Methods use models that are targeted at facilitating engagement between diverse stakeholders, using informal diagrammatic and narrative models.

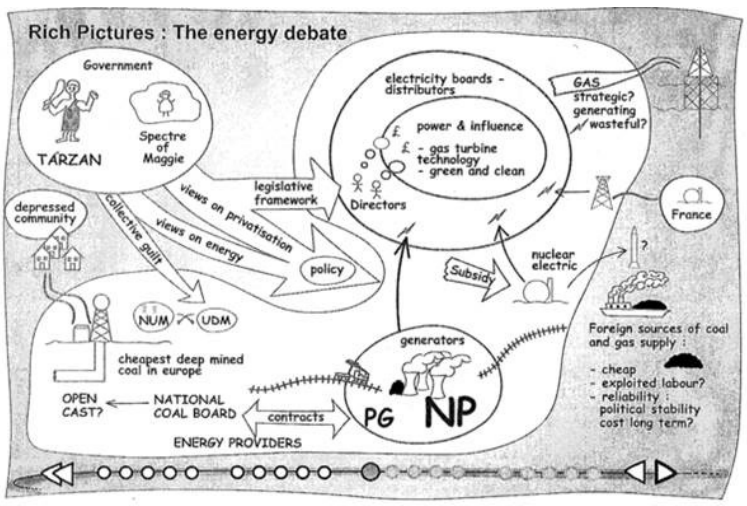

Figure 2: The Rich Picture

The Rich Picture is a good example of this [Ref 6] – it is typically very informal, engaging and expressive, providing context and situational information that relies little on abstraction and is directly meaningful to stakeholder day-to-day experience. Its primary purpose is to enable diverse stakeholders to engage with the problem and solution space and think deeply about both. A Rich Picture typically focuses on

the relationships between things rather than just cataloguing types of things.

2. Concern with System boundaries: the formal term for this in Systems Engineering terms is 'System of Interest'. Soft Systems methods do not have a specific name for this, however it regularly appears in Rich Pictures and built in to the notion of 'Environmental Constraints'. If there is a difference between these worlds, it is more in the way that Soft Systems methods are ready/willing/able to flex the boundary, or make it more porous. Both approaches recognize the concept of System of Systems, although again, Systems Engineering has rather more formal definition of it.

3. Facilitation of iterative analysis: Soft System methods make iterative model development a core and integral part of the approach. There is a clear expectation among stakeholders who engage in the process that the models will change – in fact, it is encouraged. Flexibility of the models is critical to the promotion of iterative analysis and creative problem solving. Systems Engineering, while it aspires to the use of models for similar effect, often, due to time constraints, fixes the models earlier in the exploration process whether as a 'straw' man' or 'draft' and only change under a well-argued (and sometimes courageous) challenge. If there is a difference between the approaches, it is that Soft Systems methods build in resistance to premature fixing of problem and solution definition, while Systems Engineering, although not by intent, builds in resistance to change for fear of undermining the integrity of the models.

4. Suitable for the analysis and specification of Information Systems: probably because Soft Systems methods emerged to address the shortcomings of Systems Engineering, its development from that world means that it is closely connected to the systems analysis of software intensive systems. Although it is not specialized for that purpose, it (or elements of it) has regularly been used in the Information Systems industry. Systems Engineering, on the other hand, is often specialized to deal with the specifics of software intensive systems as an element of a broader and more diverse set of systems.

5. Considers inter-dependency of System components as a critical feature: this is a major area of common ground, although often from different perspectives. Soft Systems methods consider the relationships between things to be the key feature that gives the things meaning. Systems Engineering considers the relationships almost as important, but mainly for the purpose of effectively managing inter-component dependencies, decoupling of systems and controlling of change. Enterprise Architecture often falls short in this area as the cataloguing of things takes precedence over understanding their relationships.

Soft Systems methods	Systems Engineering/TOGAF®
Enter the problem, situation	Feasibility Study and Concept Exploration/Preliminary Phase
Express the problem situation	Concept if Operations/Architecture Vision
Formulate root definitions of relevant systems	System Requirements/Conceptual Architecture (Business, Information Systems & Technology)
Build conceptual models of (human) activity systems	High Level Design/Logical Architecture
Compare the models with the real world	Optioneering and Tradespace Exploration/Opportunities and Solutions
Define changes that are desirable and feasible	Option Selection/Migration Planning
Take action to improve the real world situation	Implementation/Implementation & Governance

Table 2: Multi-step Processes

6. Implementation through an iterative multi-step approach: this is another major area of common ground, often with direct correspondence between the approaches. The table below provides a very approximate expression of the relationships, using TOGAF® to specialize Systems Engineering.

7. Reliance on categories/types of Stakeholder: both approaches codify the types of stakeholder and influences on the process. Soft Systems methods typically use the 'CATWOE' mnemonic to ensure inclusion, covering: Clients (beneficiaries or victims), Actors (enact the system), Transformation (transformations performed by the system), Worldview (values that give the system meaning), Owner (authority over the system), Environment Constraints (external constraints).

If we can re-imagine Enterprise Architecture as a subset of Systems Engineering, and Systems Engineering as a subset of Soft Systems, the Enterprise Architect can be well positioned to fully mobilize a rich set of concepts, techniques and tools to deal with an increasingly diverse, complex and big world.

The next white paper in this series explores in more detail the notion of the Enterprise as a complex engineering object and the limits of adopting an Engineering-intensive approach to Enterprise Architecture. It proposes how integrating it with Soft Systems provide a comprehensive set of tools for most situations.

References:

[1] The Art of Judgment Series. Orbus: http://www.orbussoftware.com/resources/authors/ceri-williams/

[2] Weinberg, Gerald M: An Introduction to General Systems Thinking. ISBN: 0-932633-49-8

[3] Wicked Problems: https://en.wikipedia.org/wiki/Wicked_problem

[4] Soft Systems Method: https://en.wikipedia.org/wiki/Soft_systems

[5] Checkland, P & Poulter, J: learning for Action – A Short Definitive Account of Soft Systems Methodology and its use for Practitioners, Teachers and Students. ISBN: 9780470025543

[6] Rich Pictures: http://systems.open.ac.uk/materials/T552/pages/rich/rp-what.html The Open University

Also, for books that will change your life:

[7] Vickers, G (1995) The Art of Judgment Centenary Edition. ISBN: 0-8039-7362-4

[8] Koberg, Dan & Bagnall, Jim: the Universal Traveller…a Soft-Systems guide to creativity, problem solving and the process of reaching goals. http://www.amazon.co.uk/Universal-Traveler-Soft-Systems-Creativity-Problem-Solving/dp/1560526793/ref=sr_1_1?ie=UTF8&qid=1389375638&sr=8-1&keywords=universal+traveller

Orbus Software
3rd Floor
111 Buckingham Palace Road
London
SW1W 0SR
United Kingdom

+44 (0) 870 991 1851
enquiries@orbussoftware.com
www.orbussoftware.com

orbus software