4-10-2008

The Role of the Information Architect: Conquering CognitiveParochialism

Antony Bryant
Leeds Metropolitan University, a.bryant@leedsmet.ac.uk

Rik Maes
University of Amsterdam, maestro@uva.nl

芽|Sprouts Working Papers on Information Systems ISSN 1535-6078

The Role of the Information Architect: Conquering Cognitive Parochialism

Antony Bryant
Leeds Metropolitan University, United Kingdom
Rik Maes
University of Amsterdam, The Netherlands

Abstract
The architectural metaphor has played an important role in many aspects of IS/ICT since the 1970s. One key influence in this has been Zachmanâ s ISA, first introduced in 1987. This is now a pivotal aspect of the domain, but it has developed in a lop-sided fashion with the structural features of the architectural trope effacing the cognitive ones. This paper focuses attention back on the neglected aspects of the architectural metaphor, arguing that a more comprehensive and accomplished conception of the role of the information architect and the nature of information architecture are critical for current IS/ICT practices.

Keywords: information architecture; information architect; Zachman framework, ISA

Permanent URL: http://sprouts.aisnet.org/5-10

Reference: Bryant, A., Maes, R. (2005). "The Role of the Information Architect: Conquering Cognitive Parochialism," University of Amsterdam, Netherlands . *Sprouts: Working Papers on Information Systems*, 5(10). http://sprouts.aisnet.org/5-10

1. INTRODUCTION

The notion of 'architecture' is widely used in the IS/ICT[3] world. A cursory trawl through recent offerings – commercial and academic, in print and electronic – produces terms as diverse as *business architecture, knowledge architecture, strategic architecture, governance architecture, IS architecture, IS competence architecture, IT architecture, network architecture, computer architecture*, and could certainly be further extended. The word itself definitely has a certain cachet in the context of IS/ICT; but why is it actually used and what does it actually mean or imply? Could it simply be replaced by simpler, perhaps less resonant terms such as *structure* or *framework*?

Our specific concern at this juncture is the term *information architecture*.[4] This has become an increasingly popular phrase in recent times, so it was no surprise that a query on Google returned more than 600,000 instances of the matching phrase. However a swift investigation of a small and random sampling of these instances indicated that in almost all cases the word 'architecture' could indeed be replaced by 'structure' or 'framework' without any loss of meaning. This is not to imply that the term 'information architecture' – or any of the other phrases that include the word in the domain of IS/ICT – should be replaced by simpler terms; rather we saw it as a spur for an investigation into the derivation and usage of the term, and this led us to develop an argument in favour of an enriched concept of the architectural trope in general, and 'information architecture' specifically.

Our intention is not primarily to draw attention to the aberrant ways in which the term *architecture* (in particular information architecture) and the architectural metaphor have come to be used with regard to IS/ICT; but rather to reconstruct the concept, and clarify and enhance its metaphorical power. To this end, we stress the important cognitive, perceptual and communicative aspects which have largely been lost, ignored or at least over-whelmed by

[3] Although phrases such as IS/ICT realm, domain of IS/ICT etc seem somewhat cumbersome, our use is deliberate since it invokes both the hardware (IT) aspects as well as the more critical, but less tangible systemic ones (IS).

[4] The architectural metaphor has received further impetus in the context of the *Service Oriented Architecture* and the *Model Driven Architecture*. Limitations of space and the objective of focusing our ideas on one aspect at this juncture prevent us from attending to these important areas in this paper.

current tendencies to focus on the structural, *con*structional and tectonic aspects of the architectural metaphor.

2. ZACHMAN'S ISA

All the academic disciplines associated with IS/ICT exhibit the characteristics of conceptual magpies: Eyeing up attractive and alluring concepts and terms in other – often related – disciplines; then applying them in some new context. Software engineering, a sub-discipline of Computing, exemplifies this in borrowing and adapting the term engineering itself, and then adding related terms such as *requirements, prototyping, specification* and *maintenance*. Generally the field of engineering was considered to be rich source for these conceptual borrowings as the various computer-related disciplines emerged in the 1960s. After all engineering could be regarded as a kindred discipline since computer hardware was clearly an engineered product, derived from work done by electronic engineers. The adoption of the engineering label did provoke considerable criticism and unease both from engineers themselves and from those within the computing domain. To paraphrase Star Trek, IS/ICT may be engineering but not as we know it! (See Bryant 2000 for a more detailed account of this.)

By the 1970s a new source of concepts was revealed. It was only a small step to move from the engineering metaphor to the construction metaphor. The activities required to develop software systems increased in scope as the nature of computer-based information systems developed to encompass a wider range of facilities and features. Such activities demanded a model for managing and co-ordinating, and those used in conjunction with major engineering projects were readily to hand. The idea of *constructing* computer-based information systems was a simple step to take. Moreover a key feature of construction projects was that the later assembly or building phases were based on a rigorous specification that had been designed, revised and presented in a formal manner, usually by architects of one sort or another. So by the early 1980s the generic metaphor of construction was enhanced by the more specific architectural metaphor within the IS/ICT context.

One of the landmark uses of this metaphor originated with John Zachman's paper in 1987, formulating the concept and the model of the 'Information Systems Architecture' [ISA]. Zachman's paper was not the first use of the architectural trope in this context, indeed Zachman himself noted in 1987 that it was no longer a novelty; and one of his motivations for writing the paper was to offer some clear and rational basis for use of the term given the lack of 'consistency in concepts or in specifications of *architecture*'. He sought to utilize the architectural metaphor but in a very specific way as the abstract to his article states.

> With the increasing size and complexity of the implementations of information systems, it is necessary to use some logical construct (*or architecture*) for defining and controlling the interfaces and the integration of all of the components of the system. *Stress added*

By the mid 1980s the nature of information systems had developed to the extent that they were no longer the relatively constrained and esoteric technical products of the 1960s, but on the contrary were critical components of their business or organizational context. An information system had to become an embedded component of its environment; and in order for this to be accomplished and sustained the preparation for, and development and performance of such systems had to take into account a wide, and possibly irreconcilable range of different perspectives and orientations. This was not merely an argument for additional meetings and consultations; rather it necessitated ways in which the entire process of conceptualizing, developing and operating such systems could be rendered understandable to and controllable by many different interests and cognitive groupings, each with specific assumptions, perspectives and modes of reasoning.

Zachman, in recognizing this, decided to draw his concepts and derive his model from what he termed 'classical architecture'. The starting point was that the initial model of a building or construction was often no more than a back-of-an-envelope sketch intended as a basis for negotiation between the client and the specialist (architect). Zachman stated that the purpose of this 'bubble chart' was as much to confirm to the owner that the architect understood what

was in the owner's[5] mind, as it was to provide a basis for the more technical aspects of the project. As such it was a device for focusing negotiation between these different parties, assisting in the resolution of any discrepancies or misunderstandings; and eventually serving as a crucial form of confirmation and contractual agreement. If such accord could be reached in this manner at this stage, then the project could be initiated; with the clients and users having some idea of what the finished product would look like or how it would function.

This confirmation and common grounding was of particular importance to the information systems professional since the mismatch between the initial concepts of a system on the part of the clients and users, and the finished product was an increasingly irksome, expensive and embarrassing issue. A graphical and graphic exemplification of this was to be found on the wall of many offices at the time. (see box 1)

One of the main factors underlying this issue was the essential nature of software itself, which as F.P. Brooks (1987) pointed out in a seminal article – dating from the same period as Zachman's – was complex, changeable, conformable and invisible. It was no wonder that software-based systems were often expensive failures, with research in the 1980s indicating that as much as 80% of government expenditure on software systems was wasted on systems that either were never completed or were not fit-for-purpose when delivered. The *Methodology Era* of the 1980s and 1990s was stimulated and propelled by precisely this state of affairs; modelling techniques, procedures, methods, methodologies and tools were all hailed as devices to be enlisted in the continuous battle with the essential properties of software and the miasma around user requirements.

The attraction of the architectural metaphor was that it seemed tailor-made for an intricate set of inter-related activities demanding constant oversight, highly technical specification, separation of concerns; but above all else a clear commitment to clarifying and satisfying client and user requirements. The engineering metaphor, interpreted and applied incisively

[5] NB Zachman uses the term *owner* with no explanation or discussion. We might now deem *client* or *user* more appropriate; and for the remainder of this article we shall use these latter terms. A client will be taken to be someone able to exert control on the project, particularly budgetary control; a user will be someone expecting to benefit from the completed project, but not necessarily with the influence of the client. In many cases clients will also be users, but of a specialized type.

and judiciously, could also serve this complex set of objectives and principles. Tony Hoare stressed precisely these aspects in his paper in the 1970s where he took issue with those software producers who simply adopted the term *engineer* without considering the full implications. For Hoare, to claim the title *engineer* was to be held to account against the highest principles of a profession. Professionals did not simply complete a specified task of work, they had to be imbued with a wide vision, a doctrine of service to the community, and a large degree of autonomy in their activities. Professionals also should have some form of certification from their own professional body; giving them independence from commercial and political pressures, but also subjecting them to sanctions, discipline and commendation by their peers. Needless to say, Hoare concluded that the software engineers of the 1970s had no claim to being professionals. In part this was because their sole concern was the software product itself, developed and supported in the narrowest sense.

The architectural metaphor seems to offer a way of encompassing Hoare's vision precisely by moving beyond the more limited and mechanistic view of the engineering one, which Hoare rightly criticizes. The architectural trope forces attention on the necessity to encompass something wider than the artefact itself. Although there still seems to be an inherent tendency to remain rooted to the *artefact* – whether it is the software as a whole or a set of software objects.[6]

This tension between product and process is also evident in Zachman's work. A careful reading of his 1987 paper reveals that his argument is founded on distinguishing between three distinct cognitive or perceptual domains, and consequently on the necessity of establishing mutual recognition and dialogue between them. Zachman suggests that the domains of *owner*, *designer*, and *builder* are cognitively distinct and potentially contradictory; and so it is imperative that some form of negotiation can be established and sustained with the aim of reaching a consensus: Or at least demonstrating recognition that there may be an inability to do so.

[6] *IEEE Software, September/October,1999* carried a section devoted to *re-evaluating* the architectural metaphor; the editor of this section, James Coplien, noted that 'focusing on objects had caused us to lose the system perspective'.

However this cognitive-oriented thread of his argument is interwoven with a more product-centred one. Hence he presents the Information Systems Architecture [ISA] as offering a developmental path for different systems abstractions or models. These abstractions are similar to 'a set of *architectural* representations that are produced during the process of constructing a building'. Zachman's aim is to provide a generic characterization of these abstractions, with his approach applying to 'any complex *engineering* product'. So the architectural and the engineering metaphors are here combined; although the primary stress is clearly on the former.

This interweaving of process (dealing with distinction perceptions) and products (abstractions or models) leads to a confusion that severely weakens the central argument. One of the most common readings of Zachman's papers is that the ISA is actually a series of top-down products, each adding further levels of detail; completely effacing the issue of perceptual discrepancy. Zachman must have realized this misunderstanding might arise since he specifically warns against this, stressing that each representation

> has a different *nature* from the others. They are not merely a set of representations, each of which displays a level of detail greater than the previous one.

But the fact that he had to give this explicit admonition indicates that the ISA was likely to evoke such interpretations. Moreover the later paper, by Sowa and Zachman (1992), continually uses phrases that themselves engender this misunderstanding. For example they mention that 'the architect's plans are the translations of the drawings into *detailed* specifications'; and 'the technology model which must adapt the information systems model to the *details* of the programming languages, I/O devices or other technology' (stress added). In fact page 592 of the 1992 paper uses the word *detail* in its description of *all* of rows of the revised ISA!

Indeed the 1992 paper specifies how the ISA itself can be understood in a variety of potentially contradictory ways. The ISA can be seen in terms of

- a model of 6 *levels* – i.e. a sequential series of linked abstractions – despite any protests by the authors to the contrary;
- as a 'taxonomy';
- as a basis for component design and '*periodic table* for information entities'.

A glance at the current uses of ISA and its associated website[7] indicates that in the intervening period the structural aspects have eclipsed the perceptual ones. The framework is widely used, and there are many project offices where the 6x6 grid is pinned up, each cell populated with a model, and each column seen as an increase in detail reading from top to bottom. The relationships between rows are submerged beneath the concern with adding further detail and focusing on structure. Zachman's framework has become part of the IS body of knowledge; with high visibility amongst practitioners, even more than amongst academics and researchers. If his objective of clarifying the ways in which the IS community apply the concept of architecture has been in any way successful, then it is an ironic accomplishment since it undermines the main strength of his initial argument. The structural has won out over the conceptual.

3. THE NATURE OF METAPHOR

A different source of the architectural metaphor derives from its application with regard to models of computer hardware. Here the term *computer architecture* predates Zachman by at least a decade. A key text of 1996 by Shaw & Garlan on ***software*** *architecture*, articulates a position for what the authors term 'an emerging discipline'. This proto-discipline is derived from the confluence of concepts drawn from the field of *hardware*, and those from the field of *design* as applied to software systems. The authors make no mention of Zachman or the term information architecture. This is perhaps understandable; but what is not so excusable is the failure to mention the work of F. P. Brooks who used the concept of architecture in the 1970s very much in the way that Shaw & Garlan used it two decades later.

[7] There is a website dedicated to Zachman's framework – The Zachman Institute for Framework Advancement (ZIFA) http://www.zifa.com/

There are however similarities – perhaps unwitting – between Zachman and Shaw & Garlan, as is clear from Barry Boehm's introduction to the latter which stressed the key point that the concept of architecture in this new context was used to emphasize and draw attention to the necessity for intermediate or bridging abstractions that connect 'characteristics of systems users needs to characteristics of systems that software engineers can build'. (vii) This is fully in tune with Zachman's argument about the need to offer links between the perceptually distinct realms of the 'owner, designer and builder'; and so aligns with what we may now regard as the neglected but critical aspect of Zachman's 1987 paper.

Given the background of many practitioners in the computing-IT field at the time, it was not too unexpected that the architectural metaphor should have been taken up predominantly in the relatively rigid structural sense, and initially applied to hardware. It made sense to talk about computer architecture in terms of hardware components, and also in the sense of families of hardware based on similar *architectures* – e.g. VAX, PDP, IBM 360, and the like. This allowed vendors to offer a range of *solutions* to customers, apparently responding to the specific requirements of any particular client; but in reality often doing no more than implementing standard facilities – CPU, registers, memory etc – within constrained product ranges. This allowed manufacturers and vendors to enforce a balance between offering the full standard range of required facilities, and going one step further and offering both distinctive features as well as an upgrade path.

The combination of the partial or erroneous interpretation of the ISA and the hardware-oriented use of the term, meant that by the late 1990s the architectural metaphor had largely come to be associated with structure and configuration. Computer architecture had, however, receded into the background as the technology had become ubiquitous and virtually transparent in the sense that it was almost taken for granted. Physical architectures now encompassed *networks* rather than individual machines; and here architecture was definitely meant in the sense of configuration, composition and component parts.

It might be thought that our concern with the ways in which the architectural metaphor has been taken up within IS/ICT is unduly portentous; after all *it is only a metaphor*, 'a figure of speech'. But this is severely to underestimate the potency and consequence of metaphors. Although it might be thought that metaphors are merely restricted linguistic devices, usually (consciously) employed for dramatic effects, this is an inadequate view, and effaces the role that metaphor plays in our lives. Fowler's *Modern English Usage* dating from the 1960s

(1965) noted that 'our vocabulary is largely built on metaphors; we use them, though perhaps not consciously, whenever we speak or write'. More recently Lakoff and his colleagues have been instrumental in establishing our understanding that the cognitive power of metaphor extends to all aspects of our social existence; neatly evoked by the title of Lakoff and Johnson's book *Metaphors we live by*.[8] This work not only underlines the ubiquitous nature of metaphors, but stresses their role and power in our thought processes. For Fowler metaphors are pervasive, but essentially passive. Lakoff and others now challenge this view, seeing metaphors as playing an *active* role in thought and cognition. In particular, metaphors are now seen as a crucial aspect of the reception and propagation of new ideas and more tangible facets of our existence; hence their critical role in accompanying the emergence of IS/ICT.A further characteristic of metaphors is that they take on a life of their own. They cannot be controlled, and may have unforeseen and unwelcome consequences. Software Engineering is a case in point. In his celebrated paper *No Silver Bullet*, F.P. Brooks noted the impact that the construction metaphor had on him when he first came across it in 1958. 'The metaphor shift was powerful, and accurate. Today … we freely use other elements of the metaphor, such as *specifications*, *assembly of components*, and *scaffolding*.' (p18) Writing in the mid-1980s, however, he argued that, although useful and powerful, this construction metaphor had outlived its usefulness, and needed to be replaced by an image of software development more akin to *growth* than construction. Brooks uses the term 'metaphor *shift'* when noting the impact of the building metaphor; implying that it replaced an existing metaphor. More than that, he takes it as axiomatic that metaphors operate with regard to software development. The particular shift he locates in the 1950s is one from *writing* software to *building* software; and these metaphors have a powerful impact on people's practices and cognition. The terms used are not simply words on a page, they have significant effects: As Brooks asserts when pointing out that perhaps the *construction* metaphor should be replaced by one concerned with *growth and nurture* - 'I have seen dramatic results since I began urging this technique (the growing of software using incremental development) on my project builders in my Software Engineering Laboratory class.' (p18) He continues in a similar vein, mentioning heightened morale, jumps in enthusiasm, redoubled efforts and so

[8] Lakoff has recently argued that the outcome of the US Presidential election owed a good deal to the power of the metaphorical imagery used by the Republicans and the paucity of that employed by the Democrats; a suggestion met with a good deal of scepticism.

on. He concludes that 'teams can *grow* much more complex entities in four months than they can *build*'.

If we follow the arguments of Brooks, Lakoff and others, then it becomes apparent that the current architectural metaphor – stressing structure to the virtual exclusion of all else – may be having potent and constraining ramifications on practices within IS/ICT. Challenging and enhancing this particular trope amounts to far more than an academic exercise centring on a *mere figure of speech*; it may well have profound consequences for those twin headline-grabbing concerns, *relevance* and *practice*. As such our intention is to draw attention to the ways in which the term *architecture* and the architectural metaphor have come to be used, particularly in the sense of *information architecture*. We are not doing this in pursuit of some form of conceptual purity, but because we wish to reconstruct the concept and clarify the metaphorical power, stressing the important cognitive and perceptual aspects which have largely been lost, ignored or at least over-whelmed by current tendencies to focus on the structural, *con*structional, and tectonic.

4. INFORMATION ARCHITECTURE – REMEDY AND REVISION

One of the appeals of the architectural metaphor, specifically alluded to by Zachman, seems to have been the prominence it gave to the ways in which architectural practices and artefacts coped with the issue of how best to facilitate an understanding between different parties to any complex (construction) project. In many senses these different participants are by definition non-like-minded, and consequently unlikely to arrive at a consensus without considerable effort and forethought. Architects and engineers have been here before, as perhaps have (some) doctors. Hoare in his 1975 paper specifically argued that one of the key features of claiming the status of a professional – as software developers appear to do in adopting the epithet software *engineer* – is that these professionals have a better understanding of their clients' requirements than do their clients themselves. Yet there is a fine line between specialists who know their patients' or clients' needs better than the latter do themselves, and those who simply lord it over their clients; the distinction is between insight and compassion on the one hand and high-handedness and arrogance on the other. The paradox is that often there is no middle way between these two orientations, since the

very nature of the problem/project context encompasses intricate technicalities that are almost certainly beyond the perceptual grasp of the non-specialist clients. It may well be *their problem* or *their requirement*, but by definition it is not theirs to specify without considerable guidance and manipulation by those more qualified. Other professions have established processes and procedures to cope with this concern; and so IS/ICT practitioners must do likewise if they are to lay claim to an equal status.

In his 1987 paper Zachman seems to suggest that the architectural metaphor – if translated into practice – will provide a basis for such guidance and mentoring; with the ISA offering a relevant scheme for promoting this. Without explicitly stating the argument, he intimates that practitioners with an understanding of the rationale and motivation underlying the ISA will be more likely to act with the insight and empathy required to encompass and reconcile the range of perspectives and cognitive abstractions given in the ISA itself. This in fact necessitates a major feat on the part of *information architects* who must be capable of overcoming their own cognitive parochialism in recognizing the limitations of other people's. They must then seek ways in which these disparate perceptions can be reconciled so that they satisfy client and user requirements – real and perceived. Moreover the aforementioned failure to appreciate and disseminate this crucial skill as part of the role of the architect within the domain of IS/ICT exposes a fundamental flaw in current practice and understanding.

To reiterate the earlier argument: Even though the term *information architecture* has achieved wide currency – together with a whole host of other terms incorporating *architecture*; in virtually all cases the term could be replaced – and intelligibility increased – by *structure* or *configuration*. The rationale for the metaphor centring on bridging abstractions has been overlooked, relegated or discarded. The predominant focus on the tectonic and mechanistic aspects of architecture has overwhelmed the semantic intricacies. If information architecture is to mean no more than information *configuration*, then it might be better to dispense with the term 'architecture' in this context altogether. But it will be far better to remedy our understanding and revive the full implications of the trope, also encompassing derived terms such as *information architect*. More explicitly, if the role and responsibility of an information architect is to mean anything, then it must encompass the

ability to relate and transcend varying perceptual abstractions in the sense alluded to by Boehm; and this is a crucial aspect of most facets of IS/ICT practice.

At this point it is pertinent to ask how well the practices within architecture itself measure up to this archetype. Do architects manage to combine this dual function of acting as technical specialist whilst also acting as broker between potentially contending interests? In fact it appears that architects themselves have battled with exactly similar arguments about the nature of their practices. Roland Barthes (1967), referring to the work of Kevin Lynch, draws attention to the ways in which urban architects have to use terms that relate to those understood and used by city dwellers. Barthes specifically points out that architects have to recognize and build upon the ways in which urbanites develop and impose schemas of discrete units that break-up and render the city itself as comprehensible and intelligible. In terms that are strikingly similar to those used by the patterns movement in software – itself derived from architecture – Barthes gives examples such as *path*, *edge*, *district*, *node*, and *landmark* which act 'like phonemes or semantemes' – i.e. fundamental units of meaning. He argues that this is a *Gestalt* as opposed to a structural conception of architectural practice, and that in general there is a conflict between the semantic view and the functional one.

This concept of a *semantic*, *Gestalt*, or cognitive practice resonates with Zachman's initial rationale, and Boehm's concern with bridging abstractions.[9] In both cases there is the recognition that people's perceptions are constitutive of their experienced and understood reality, and consequently that striving to model a context involves a negotiated dialogue between distinct perspectives. This is precisely the aspect to which we now draw attention. Practitioners – whether architects or information architects – have to adapt their practices accordingly, seeking to transcend their own perspectives in order to preside over dialogical processes aimed at achieving and sustaining some form of consensus; even if this is unlikely to be completely satisfactory for all those involved.

Umberto Eco (1973) develops these arguments in his work on *Function & Sign: Semiotics of Architecture*. He offers a very wide definition of architecture – which, with some minor

[9] NB 'bridging abstractions' can be interpreted both as a verb and a noun – the ambiguity is intentional.

modification, could readily include Information Systems (exclude the phrase 'three-dimensional')

> any type of design producing three-dimensional constructions destined to permit the fulfilment of some function connected with life in society (p182).

Like Barthes, Eco argues that architecture operates communicatively as well as functionally. He notes that as soon as people start to generalize from one occurrence to another – e.g. from the first cave in which they take shelter to a second cave they come across later – they generate models of generic concepts. At first these will operate on an individual or private level; but eventually such concepts will start to be used interactively and will then become *iconic principles*, and objects of communicative intercourse: Indeed they will make communication possible *per se*.[10]

It must be pointed out that some of the most fruitful and some of the most exasperating forms of communication occur when people with differing *iconic principles* enter into dialogue; particularly if they utilize the same *sign vehicles* (see below) but find that they are pointing in different directions. This is endemic to architectural practice, and even more so in IS/ICT. The adoption of engineering terms exemplifies this; hence the continuous and sustained disputes around the meaning of terms such as *requirements*, *maintenance*, and *prototyping* when taken from their earlier context and re-interpreted in a new one. With regard to ICT, its constituent *sign vehicles* have undergone enormous re-design and re-grounding in the past 40 years or so. In the 1970s and 1980s teaching people about the promise and potential of computer-based technology – often termed 'computer appreciation' – involved use of sign vehicles drawn from more mundane aspects of people's existence. Thus there was extensive use of analogies and metaphors based around *desk-tops, files, documents,* and the like. In the intervening period ICT has itself become not only part of our everyday existence, but has taken on a defining role so that many old concepts as well as new ones are couched in terms derived for computer-based technology. There is therefore a great deal of truth to say that we

[10] The argument that generalizations and concepts of 'sameness' are fundamental to social interaction is made even more forcefully in the work of Mary Douglas (1986), who adopts and adapts a Durkheimian position in arguing that only institutions – i.e. social groupings – can confer identity; 'sameness is conferred on the mixed bundle of items that count as members of a category; their sameness is conferred and fixed by institutions.' (p.53) So for Douglas, unlike Eco, there is no pre-social possibility of conferring *sameness*; but that is not a crucial distinction in the present context.

live in a computer age since so many of our *iconic* principles are computer-based, or more correctly ICT-based.

Eco argues that artefacts and models 'communicate the function to be fulfilled' (183), and that they signify this 'even when they are not being used'. Eco coins the phrase *sign vehicle*.

> Thus what our semiotic framework would recognize in the architectural sign is *the presence of a sign vehicle whose denoted meaning is the function it makes possible* … (184 – stress in original).

The sign vehicle is 'observable and describable apart from the meanings we attribute to them'. This distinction between the sign vehicle and the meaning

> permits us to recognize in architectural signs *sign vehicles capable of being described and catalogued*, which can denote precise functions provided one interprets them in the light of certain codes (184 – underlined part indicates stress added)

Eco offers the example of a flight of stairs, which denotes the possibility of going up. But this link between form and function operates through a codified connection and 'also a conventional conception of how one fulfils the function with the form'. Thus someone confronted by an elevator for the first time (e.g. Peter Sellers in the film *Being There*) would have no idea how the form fulfils the function of 'going up'. One cannot simply state the maxim that *form follows function*. It must be revised, in Eco's words

> *the form of the object must, besides making the function possible, denote that function clearly enough to make it practicable as well as desirable* … Then all the ingenuity of the architect or designer cannot make the new form functional (and cannot give form to a new function) *without the support of existing processes of codification* … (186).

The ramifications of this for information architecture are significant. Zachman's motivation in his 1987 paper can be seen to emanate from the historical context where IT was being used as a new, indeed revolutionary form for pre-existing functions. Thus in part his framework

can be seen as a response to the need to make the new forms functional within existing processes of codification. In the intervening period these 'existing processes' have developed sufficiently so that previously esoteric IS/ICT-based terms are now part of common parlance – e.g. input, database, computer network, internet and the like. To some extent this explains why the meaning of the term architecture in the IS/ICT context has altered. Yet Eco's central point, that the architect has to fulfil a primarily communicative role, must not be allowed to disappear simply because the technology is now part of our everyday life and habits.

Eco develops this argument stressing the distinction between *denotation* and *connotation*. Denotation is a far more direct or explicit aspect of signification; connotation is more diffuse and implicit. Thus a cave may have come to denote 'shelter', but in time it may also have 'begun to connote family or group, security, familiar surroundings, etc' (187). Eco stresses that this 'symbolic function' is no less functional or useful than its primary, denotative one. The connotative functions he deems secondary, not because they are less important, but based on the semiotic mechanism, 'in the sense that the secondary functions rest on the denotation of the primary function'. (188)

In the context of IS/ICT, architecture was incorporated to denote 'architecture-as-structure'; this was its primary function at least in the sense of being the original meaning. But it has failed to develop beyond this and form the basis for any widespread awareness of the secondary connotation of 'architecture-as-bridge-across-abstractions'. This is particularly manifest with regard to information architecture. In part this may because the concept of information architecture has not really stabilized. The massive developments in use and range of ICT have not given any firm foundation for an understanding of the term; and the development of the internet and world wide web have engendered a range of further uses of the term. This span encompasses those for whom IA relates to the *external presentation of information*, particularly websites, and those for whom IA refers to the *internal organisation* of (enterprise) information. Our critique is primarily directed against the latter, although those espousing the former also fail to engage with the cognitive aspects to which we draw attention.

For Eco, processes of denotation and connotation only operate within a context where they can draw support from 'existing processes of codification'. If there is no stability or at least temporary stasis then it is difficult to envisage how processes of codification might become established and so operate effectively. Thus it is hardly surprising that the concept of architecture has failed to develop connotatively.

In fact Eco admits that the term *code* is problematic in the context of visual representation. If the term is used with regard to linguistic codes, then related terms such as *syntactics, semantics, meta-language* have some clear relevance; but this is not necessarily the case in non-linguistic cases. There are inherent difficulties and paradoxes in applying concepts drawn from linguistic representation to non-linguistic ones. Eco warns against those who move from a phrase such as 'the semantics of architecture' to search for equivalents of concepts such as 'words', 'phrases' etc. (Barthes appears to do exactly this with regard to his *semantemes*.) Furthermore, Eco also argues that just because architecture '*can be described in terms of geometry does not indicate that architecture as such is founded on a geometric code*' (192/3). What is actually needed is an understanding of the base units of the architectural code, which is specific to itself; and Eco suggests three categories – technical, syntactic, and semantic.

The technical codes include engineering aspects such as 'beams, flooring systems, … etc.' The syntactic ones encompass 'typological codes concerning articulation into spatial types' and other syntactic conventions such as 'a stairway does not as a rule go through a window'. Semantic codes are the 'significant units of architecture, or the relations established between individual architectural sign vehicles … and their denotative and connotative meanings'. Eco outlines a scheme for these semantic codes, which we might now see as similar to Alexander's concept of patterns.[11] Moreover Eco points out that as these units are elaborated they become 'codifications of *already worked-out solutions*'. So in stead of being a source of innovation, it results in them acting as structural constraints on possibilities and potentialities. This might seem to be resonant with the ways in which proffered solutions in the realm of *information architecture* operate to present standardized implementations, rather than acting

[11] C Alexander, *A Pattern Language*, OUP 1977 – in fact Alexander's concepts have been somewhat misapplied within the context of IS/ICT – as we explain in a work currently in preparation – Maes & Bryant, 2005

as a basis for innovation and experiment. 'They establish not generative possibilities but ready-made solutions … fossilized forms'.

This standardized and constraining concept of structure differs significantly from the Chomskyan view of (linguistic) structures as *generative* and intrinsically engendering innovation and novelty. This view of structure as constraint may have a stronger rationale in the realm of IS/ICT than in architecture *per se*; after all it may be better to have a mass produced system that works within the confines of a tried-and-tested solution, than an idiosyncratic one that does not; but the danger is that this will blind people to the potential of new developments. As computer technology was developing in the 1960s and 1970s, Stafford Beer pointed out the significant fallacy in simply using this technology to add rigour or speed to existing activities in an enterprise –

> *'the question which asks how to use the computer in the enterprise, is, in short, the wrong question. A better formulation is to ask how the enterprise should be run given that computers exist. The best version of all is the question asking,* what, given computers, the enterprise now is. *(Beer, 1981, originally 1974 - stress in original)*

Generalizing this dictum, Beer is advocating a position that continually challenges the status quo with the potentialities of developing technologies. On the other hand Eco is arguing that the possibilities of innovation are not limitless, but are constrained by the necessity of some anchoring in 'existing rhetorical and ideological expectations'; without this the innovative directions would have no links to existing codes and so would preclude effective communication. The architect, however brilliant in terms of technical insight and innovation, has to retain a link with existing and prevalent codes and concepts in order to communicate ideas and persuade others. Eco argues that this means that the architect 'is continually obliged to be something other than an architect … forced to become something of a sociologist, a psychologist, an anthropologist, a semiotician …'. This is the counterpart to the point made earlier that the implicit assumption in Zachman's ISA is that the practitioner can and must strive to overcome the cognitive parochialism associated with any specific row of the ISA. Eco advocates that architects are 'obliged … to *think in terms of the totality'* and must do this regardless of the extent to which they 'have become a technician, a specialist,

someone intent on specific operations rather than general questions.' He offers the slogan that '*the architect should be designing for variable primary functions and open secondary functions'*.

5. THE ROLE OF THE INFORMATION ARCHITECT

This readily translates to the realm of IS/ICT, the role of the information architect, and the production of an information architecture. In all cases the outcomes are significantly different from currently accepted ideas. The predominant meaning of the term architecture in the IS/ICT realm is close to that of standardized problem characterizations codified in such a way that they are amenable to predetermined solutions. More critically, those who adopt or acquire the label of information architects tend to see themselves as narrowly confined specialists with a focus on specific (technical) operations – whether these are defined in terms of internal or external meanings of the term (see above). To adopt Eco's terms, the information architect must recognize the necessity to become something other than an information architect in the restricted and technicist sense. The complexities of the enterprise – internally-oriented or externally-oriented – cannot be forced into standardized formats or codes, although these formats and codes can offer a useful starting point for discussion or negotiation between cognitively alien participants.

The mandatory starting point must be the experiences and cognitive schemas of the users, clients, or general participants within the context of the enterprise. In the urban setting Barthes emphasizes that one has to begin with the ways in which urbanites develop and impose schemas of discrete units that break-up and render the city itself as comprehensible and intelligible. In similar fashion, the information architect has to start from the schemas and cognitive components 'developed and imposed' by those who comprise, act within, benefit or suffer from, and sustain the system – however that system is conceptualized. This will involve the practitioner in a role that will encompass non-technical activities and practices, sometimes involving subversions and subterfuge. Without such a realization the tendency will be simply for the practitioner to perpetuate the status quo; thus shoring up existing structures of dominance and concomitant 'regimes of truth', which will preclude any consideration of possibilities of doing things differently or doing different things. Account

has to be taken of Beer's advocacy for re-conceiving and renegotiating the nature of the enterprise.

If there is no attempt to 'think in terms of the totality', then the resulting architecture will often bear little resemblance to the reality of the enterprise. In Baudrillard's terms this is constructing a hyper-reality where the model is not simply inaccurate; but where it actually precedes and engenders the territory. The specific identity of the system is lost, overwhelmed by the 'precession of simulacra' derived from codified ready-made solutions that bear no resemblance to the specific context and are unintelligible to those involved with the system: Which is more-or-less to find ourselves back with the failure to build a simple rope-swing in a tree!

REFERENCES

Barthes, R., *Semiology and the Urban*, 1967; reprinted in N. Leach (ed) *Rethinking Architecture*, Routledge, 1997, pp166-172

Baudrillard, J., *Simulacres et Simulation*, Editions Galilée, 1981; in English, *Simulacra and Simulation*, University of Michigan Press, 1994

Beer, S., *The Brain of the Firm*, Wiley, 2nd edition 1981

Brooks, F.P., *No silver bullet: essence and accidents of software engineering*, IEEE Computer, 1987, 20:4

Bryant, A., *"It's Engineering Jim; but not as we know it"; Software Engineering, solution to the software crisis or part of the problem?, ICSE 2000*, Limerick, June 2000

Douglas, M., *How Institutions Think*, Syracuse UP, 1986

Eco, U., *Function and Sign: Semiotics of Architecture*, 1973, reprinted in N. Leach (ed) *Rethinking Architecture*, Routledge, 1997 pp182-201

Fowler, H.W., *Modern English Usage*, 2nd edition, revised by Sir Ernest Gowers, Oxford, 1965

Hoare, C.A.R. Software Engineering, *Computer Bulletin*, December 1975, pp6-7

Lakoff, G. & Johnson, M., *Metaphors We Live By*, University of Chicago Press, 1980

Maes, R., & Bryant, A., *Reconstructing the Concept of Architecture in Information Systems*, to appear, 2005

Shaw, M., & Garlan, D., *Software Architecture: Perspectives on an Emerging Discipline*; Prentice Hall, 1996

Sowa, J. F. & Zachman, J. A. *Extending and formalizing the framework for information systems architecture*, IBM Systems Journal 31:3, 1992

Zachman, J., *A Framework for Information Systems Architecture*, IBM Systems Journal, 26:3, pp454-470, 1987

http://www.uoregon.edu/~ftepfer/SchlFacilities/TireSwingTable.html

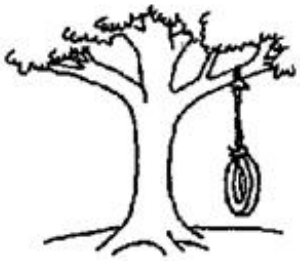
As proposed by the client

As specified in the project request

As designed by the analysts

As produced by the programmers

As installed at the user sitre



What the user wanted

INDEX

The Role of the Information Architect: Conquering Cognitive Parochialism

Antony Bryant[1] & Rik Maes[2]

ABSTRACT

The architectural metaphor has played an important role in many aspects of IS/ICT since the 1970s. One key influence in this has been Zachman's ISA, first introduced in 1987. This is now a pivotal aspect of the domain, but it has developed in a lop-sided fashion with the structural features of the architectural trope effacing the cognitive ones. This paper focuses attention back on the neglected aspects of the architectural metaphor, arguing that a more comprehensive and accomplished conception of the role of the information architect and the nature of information architecture are critical for current IS/ICT practices.

KEYWORDS

Information architecture; information architect; Zachman framework, ISA;

1 Professor of Informatics, School of Information Management, Leeds Metropolitan University, UK; a.bryant@leedsmet.ac.uk

2 Professor of Information and Communication Management, University of Amsterdam, Netherlands; maestro@uva.nl

芽|Sprouts

Working Papers on Information Systems | ISSN 1535-6078

Office:
Sprouts
University of Amsterdam
Roetersstraat 11, Room E 2.74
1018 WB Amsterdam, Netherlands
Email: admin@sprouts.aisnet.org

芽|Sprouts