Rubber Ducks, Nightmares, and Unsaturated Predicates: Proto-Scientific Schemata Are Good For Agile

Jenny Quillien
New Mexico University at Highlands
Santa Fe, New Mexico
jenny@jqsolutions.org

Dave West
New Mexico University at Highlands
Las Vegas, New Mexico
profwest@fastmail.fm

Abstract

Fine-grain case studies of scientific inquiry, lessons from linguistics on metaphoric thinking, the epistemology of Charles Sanders Peirce, recent work on architectural image-schemata, along with the computer world's own theorist, Peter Naur, all suggest that software developers (frequently dulled and desiccated from overdosing on 'Cartesian' methodologies) could benefit from imbibing a little mysticism—not the wave-your-hands woo-woo kind but the more ineffable hunch and gut side of human cognition. Scholarly publications in their final polished forms rarely admit that stories, jokes, eroticism, and dreams were the fertile seeds that germinated into 'serious' results. This essay looks to these 'closet' sources, non-reductionist, non-self conscious, metaphorical, and aformal modes of thought as the salvation of a profession gone awry. It is notably proto-scientific image-schemata that retain our attention as a pragmatic tool for improving the fecundity of Agile methodology, at its roots, so to speak. The necessary context is provided by Peter Naur's fundamental insights about software development as 'theory building' coupled with an elaboration of the Agile concept of storytelling.

The discussion starts with and, for reasons of length, mainly stays with architecture's Christopher Alexander who offers novel usages of image-schemata and whose older foundational work will be at least somewhat familiar. The reasoning laid out in the essay, however, is general enough to allow the reader to experiment with proto-scientific clues from any source, not just Alexander.

ACM Categories: K. Computing Milieu, K.7 The Computing Profession, K.7.0 General

General Terms: Design; Human Factors; Theory.

Keywords: Agile; Alexander; theory-building; stories.

1. How We Screwed Up

About fifteen years ago, Ward Cunningham introduced a few colleagues to Alexander's *The Timeless Way of Building* and *A Pattern Language*. Then Richard Gabriel, James Coplein, Kent Beck and the soon-to-be authors (Johnson, Vlissides, Gamma and Helm) of *Design Patterns* joined Cunningham to form the Hillside Group. Hillside, in turn, drove the early patterns movement and associated conferences and publications.

Although 'patterns' became a staple in the developer's diet, few remember an even earlier Alexandrian influence. *Notes on the Synthesis of Form*, Alexander's doctoral dissertation, attempted to rigorously define and quantize a 'science' of design. Cited during the 1968 NATO Conference on Software, *Notes* was to be an exemplar, adopted and adapted for software development. Current notions of software engineering and structured development derive from that conference, directly reflecting Alexander's ideas—*but only some of them*—and the distinction is crucial.

All of Alexander's writings have a dual nature:

(a) An attempt to understand the forces used by Nature to craft living systems, in other words, a focus on the world or the domain, expressed in rather mystical terms:

- The non-self conscious process
- *QWAN*, The Quality Without A Name
- Timeless Building
- Transcending the Pattern Language Gate
- Wholeness
- God

(b) A parallel attempt to define pragmatic elements that humans can use to achieve a similar end—a focus on implementation, expressed as concrete design directives:

- A calculus of design
- Resolution of forces
- Patterns
- Geometric Properties

The software community, uncomfortable with the metaphysical/mystical dimension of Alexander, tended to discretely whisk under the carpet the more mysterious phenomenological mother load, leaving the 'theater of their minds' feeling 'cleaner,' almost antiseptic, more brightly lit, with only a sparse cast of arid characters. It would be a 'good thing'—nay, a '*necessary* thing,' for us to take on board the full spectrum of perspectives as we ponder the possible contributions of Alexander's latest and still little known publication, *The Nature of Order*.

To profit (handsomely, we project) from Alexander's contributions (old and new), we need to consider them along with comparable ones from elsewhere, specifically: architect Bill Hillier's work on pathways and intelligibility, historian of science Gerald Holton's thoughts on nascent reflections, linguist M.L. Johnson's classic treatise on bodily knowledge, and, most importantly, our own subliminal intuitions. We need a revitalized understanding of what software development is really about.

2. So, What *Should* We Be Doing?

Peter Naur challenged the mainstream view of software development as a kind of 'production process' where programs and documentation constitute 'the deliverable.' Naur's alternative formulation, *Programming as Theory Building*, saw true development as the collective attainment of insights (a *Theory*) about the problems at hand and how they are addressed by the execution of the program.

> *"An affair of the world and how the program handles and supports it."*

The explicit inclusion of 'an affair of the world' and the emphasis on program 'behavior' as that which contributes to the problem solution, are critical departures from mainstream thinking. Both directly counter the prevailing idea that what matters in software is what goes on inside of the machine and not what is happening in the enveloping system.

Just how revolutionary Naur's ideas were can be appreciated by contrasting them with those of another computer science legend—Fred Brooks. For Brooks, author of the iconic piece *No Silver Bullet*, the essential difficulty confronting programmers was the inability to form a *conceptual construct*. Although there are some superficial similarities between Naur's *Theory* and Brooks' *conceptual construct*, they are semantically poles apart.

A *conceptual construct* is a model of how the program executes: i.e., flow of control, data structures, and algorithms—the step-by-step changes of state within a finite state machine. A *Theory*, in contrast, incorporates both what happens outside the computer and the *conceptual construct*.

The outer boundary of a *conceptual construct* is a set of requirements for what the computer is supposed to do. Development, therefore, has been concerned with 'building to specification' and making the process more formal and 'provable.' But. . .

> "*Where do the specifications come from?"*
>
> *"Why these requirements?"*
>
> *"What is the relationship between these specifications and the real world needs they supposedly reflect?"*
>
> *"Does an ambiguously stated expectation lose anything essential when straight jacketed by formal precision?"*

. . .These questions are vital if we are to have a robust *Theory* à la Naur.

Naur explicitly pierces the requirements boundary and asserts that 'affairs of the world' are primary concerns and what goes on inside of the machine is secondary. In fact, he really says that what happens inside of the machine is essentially irrelevant: it is the behavior of the executing program that determines the 'correctness' of the software. Indeed, it is quite possible to have provably correct software, that meets every requirement and yet fails to deliver the appropriate behavior.

Let's take an easy and commonly experienced example facing the authors at this moment: some of the specifications for Microsoft Word (which we are using to compose the original draft of this essay) include various types of auto-formatting, e.g., first letter capitalization and bullet point margins. Neither of us has ever used Word without experiencing a conflict between what we wanted to do and what Word wanted to do for us. The software performs correctly but the behavior of that 'correct' program is incorrect, annoying, and counter to supporting the 'affair of the world,' at hand which is to efficiently help us construct a conference essay. Word fails and it fails because the *Theory* behind it is impoverished, not allowing for instantiations of diverse particulars but only a rather dim-witted single end-user reminiscent of Joe The Plumber.

Agile owes a significant debt to Naur's ideas about theory building and kudos go to Alistair Cockburn for saying so. Common Agile values (simplicity, communication, feedback) and practices (pair-programming, collective code ownership, whole team, metaphor, on-site customer) directly support the creation of the kind of *Theory* that Naur advocates—but if and only if they are correctly interpreted and implemented. The caveat suggests that Agile, like Alexander, has suffered lopsided interpretation, with the formal getting more press than the aformal.

3. Telling Tales

Naur's *Theory Building* depends on crafting and relating stories. Storytelling, since time began, has been the mainstay of mankind when it comes to making collective sense of life, teaching, entertaining, and acculturating newcomers. And no wonder. Stories are effective, powerful, easily remembered, high bandwidth tools whose force resides in their primarily evocative nature. Even the simplest story, operating like a vacation snapshot, will bring to mind a dense association of other stories, circumstances, and knowledge.

Theories, à la Naur, are developed when a community tells stories about 'affairs of the world.'

what the world is like,
how it might be better if, and
I bet we could get a computer to help us with that.

The *Theory* continues in its development with stories about,

we need the computer to behave like this, or
here's how the computer will actually do it, and
here's how we'd know that the computer was doing that.

Theory is confirmed and Agile ratified with stories about

it works,
that really helps, and
here's another idea.

4. Agile Flunks Story Telling

One of the great tragedies of Agile is inadequate practitioner understanding and mastery of stories and metaphor. (Metaphor was essentially abandoned when Kent Beck removed it as an explicit practice.) Let's face it, User Stories (Agile's single most important practice supporting Theory Building) have devolved into being nothing but verbose expressions of a 'specification.'

"As a user I expect the system to allow me to login using an id and password."

"As a security manager I expect the system to incinerate anyone that incorrectly enters an id-password combination four times."

Those are not stories. A 'good' story, at minimum, means a full cast of characters, a plot, a description of interactions among the characters to advance the plot, cues, and outcomes. Consider how Robert Louis Stevenson, as he writes a letter to his Meredith, manages all aspects of 'story' in just three sentences.

Robert Louis Stevenson's Battlefield

"*For fourteen years I have not had a day's real health; I have wakened sick and gone to bed weary; and I have done my work unflinchingly. I have written in bed and written out of it, written in hemorrhages, written in sickness, written torn by coughing, written when my head swam for weakness; and for so long, it seems to me I have won my wager and recovered my glove. . . I was made for a contest, and the Powers have so willed that my battlefield should be this dingy, inglorious one of the bed and the physic bottle*."

The devolution of stories leaves us stuck with our old practices and diminishes our ability to directly link the enterprise with information technology systems that could really support it. It's more than a crying shame. Properly put to use, stories and Theory Building can bridge business practices like 'scenario-based planning,' and 'play scripts,' with software design. This same understanding would provide a sound foundation for emerging efforts in 'design thinking' and 'user experience design.'

To be fair, the full dunce cap isn't warranted. Obviously, we have developed some facility for telling stories about what happens inside the machine. Martin Fowler's *Analysis Patterns* starts to address specific domains. Works such as Eric Evan's *Domain-Driven Development* provide meta-patterns for telling stories about those 'affairs of the world' that we expect to handle with software. But, if we are weak when it comes to telling stories about the domain, we are *shockingly* weak when it comes to crafting stories at the boundary. Object stories and user stories rarely reflect the nature of the embedding system.

5. A Sojourn in Story Space

Stories are "*equipment for living.*"
S. I. Hayakawa

The relationship between *Theory* and the stories of different genres and perspectives that comprise it is shown in *Figure 1* on the following page. We see that stories make a transition from 'valid' (containing elements of truth but requiring interpretation) to 'reliable' (specific descriptions that produce consistent results when repeated).

Along the continuum, the story pane is partitioned, using a Venn type diagram, into three realms: 'affairs of the world,' transitional, and 'how the software handles or supports.'

The main types of stories are as follows (and we could elaborate by including additional story types such as heuristics or scenario planning).

Myths tend to be global in scope or to focus on epic events and individuals. Every organization tells itself stories about the organization as a whole and what it does.

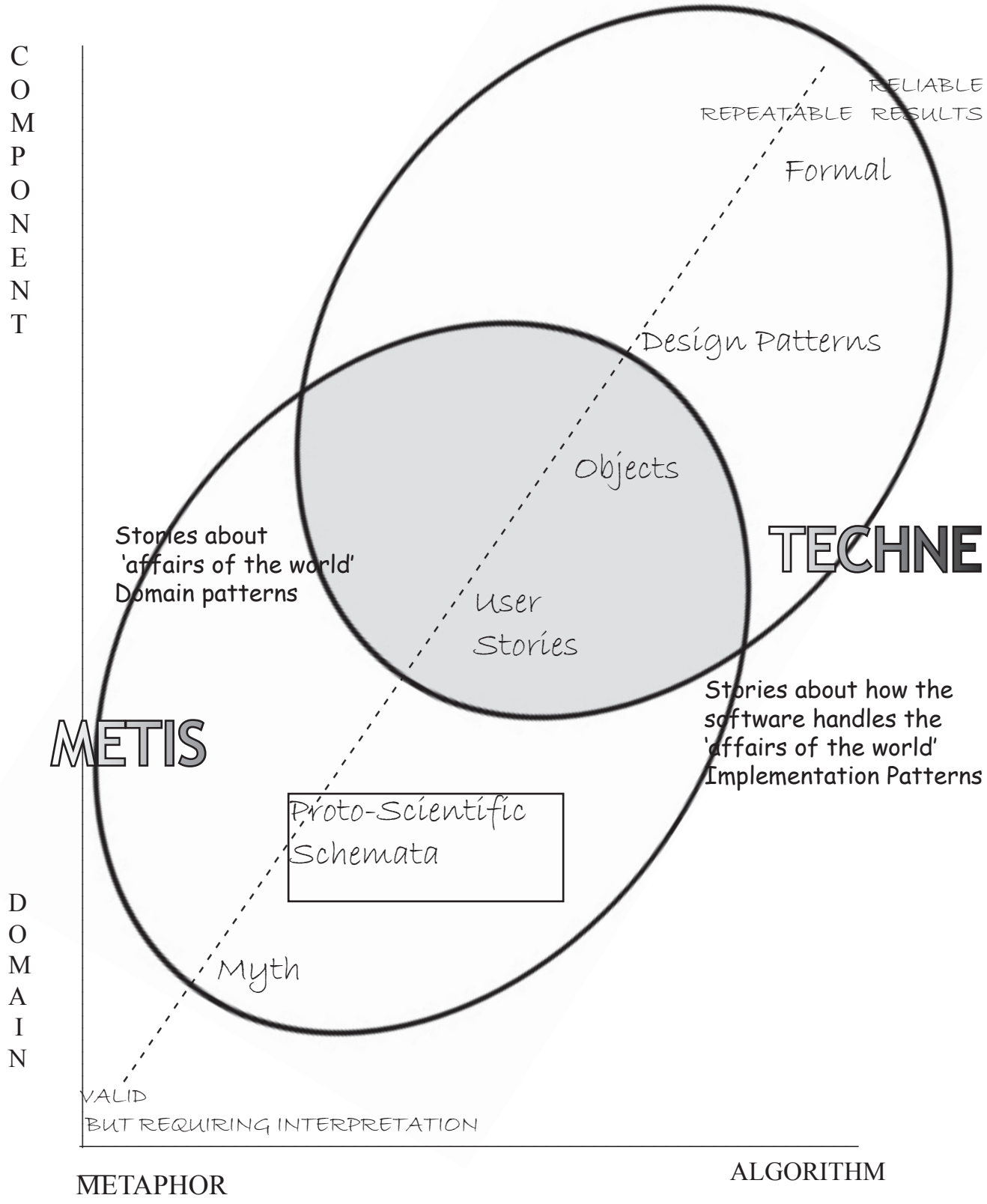


Figure 1. Story Space

Village Ingenuity

In the canton next to Geneva, which since time immemorial has made Swiss cheese, a small village specialized in also making the cheesecloth used in the aging process. The little town was isolated with many families dependent on this local craft for their livelihood. When their handmade product became obsolete, the families banded together and experimented with other outputs for their looms and know-how. Fabrics for furniture ran into too much competition but they persevered, eventually finding a niche in airplane seats covers which must be of incredible durability. Although successful at this stage, more learning was ahead. The founding family, faithful to their home town, eventually sold the company to the employees, who are now branching out to redesign the airplane seat itself, replacing the heavier frames with air cushions (covered of course with village-made fabrics) potentially saving airlines significant fuel cost. Collective loyalty to community and place, courage and deep craft knowledge of technique and material has allowed the village to successfully navigate the changing times.

Mythic stories might take the form of mission statements—valid, but subject to a lot of interpretation. Or, they might relate an event that solidified an organizational value.

Rubber Ducks

Cray Research held an annual event, Ducky Days, grounded in a story about their corporate values. Their first building in Minnesota had a fountain and one day a yellow rubber duck appeared, serenely floating about. A rather uptight, imported from outside, manager promptly issued a memo banning ducks as unbefitting the decorum of the company. Next day the fountain was overflowing with rubber ducks. The responsible manager soon departed.

Or, they speak of heroic individuals personifying values.

Walking the Talk

IBM values a culture of innovation and prudent risk taking. A senior VP was responsible for a significant loss. Watson was being interviewed and the topic of the VP and the loss came up. "*Will you fire Mr. X*?" Watson was asked. The response: "*Hell no, I just paid for his education!*"

All stories act as tropes or paradigms permanently on call to revitalize listeners and their predicaments. Stories travel. Stories support subtler modes of thought. Stories are the developer's life raft.

Proto-Scientific Image-Schemata (PSIS) help a story retain a global point of view while adding explanations about observed characteristics. [Does the reader remember *Just So Stories*?] PSIS are invaluable in improving domain stories and we will, in a later section, use them extensively in our case study of insurance companies.

Hard-wired in our deep structures of cognition are Gestalt categories, natural contents and lines of cleavage of experience: categories of shape, number, size, movement, constancy, continuity, and succession. At the conscious level we verbalizable perceptions of change and organize events into discontinuous chunks—like the ticks on a clock. In this conscious mode our field of attention is constructed. Underneath, however, is an un-verbalized flow of sensing and hunches in a diffused field of attention. Motility, intention, direction, kinetic centering and balance are sorts of non-conscious knowledge that are key to our making judgments through time and space, both real and imagined, from parallel parking our car to programming.

It is at this level that we can tap into the insights from other fields. Take the linguist M.L. Johnson who explored how our metaphors of path, blockage and enablement are rooted in our bodily knowledge of movement and impediment, and how our metaphors of compulsion and restraint removal are built around our kinetic understanding of force or pressure. (Using such metaphors to inform our ideas about proper Web-site navigation, organization, and inter-activity is an obvious possibility.)

Historians of science, such as Jacob Bronowski or Gerald Holton, go to great lengths to point out that the popular dichotomy between the 'rule of reason' and 'mystical conviction' is naïve and wrong-headed.

"Form is core rising to the surface."
Victor Hugo

Indeed, Mr. Hugo

Researcher, Otto Loewi, taking his dreams seriously, found that they told him how to design his experiment which unpacked his hunch that nervous impulses were substantially chemical rather than purely electrical in nature.

Elias Howe's design of the sewing machine came to him during a nightmare of being captured by cannibals, each with a spear with a hole in the tip.

Kekulé's discovered the circular benzene structure through dreams of Ourobos biting its tail.

Holton's detailed study of unpublished notes, letters, reminiscences, and conversations led him to a list of schemata which drive exploratory quests in science without necessarily being explicitly at issue in the research. Included in his list are: *atomicity* versus *continuum, evolution* versus *devolution, simplicity* versus *complexity, hierarchy* versus *unity, constancy* versus *evolution* versus *catastrophic change,* and *the explanatory efficacy of mathematics* versus *mechanical models.*

From architectural schemata we expect Hillier's work on space syntax and Alexander's image schemata (referred to as 'geometric properties' in *The Nature of Order*) to provide useful trail heads for systems story telling. Alexander offers essentially static structural schemata while Hillier recognizes how a world's underlying spatial grammar, particularly pathways versus 'fat' nodes (where we hang out), and degrees of accessibility and blockages, guide actions and understandings which lead, in turn, to policy and design decisions.

Our ultimate goal in software design is to create a 'virtuality' in which we can 'live,' not one that just establishes formally correct artifacts. If we employ the metaphor of a map, Alexander helps us with the layout of places, while Hillier helps us figure out how to navigate with ease. Notice though, that, Alexander-Hillier maps, and the stories told about them, cannot do more than tell us about our *Theory* of the world *at a specific time*—an important limitation. Important and problematic because stories contextualize and constrain later stories.

Why Are We Still Doing This?

> When we first theorized about airplane navigation we told stories about VORs and moving in straight lines from one VOR to another. Our air traffic control systems implemented this theory and despite the fact that our understanding of the world has changed to incorporate GPS, Great Circle routes, and computer assisted navigation, airlines still fly point-to-point, VOR-to-VOR zigzag routes with attendant congestion above each VOR. So we need be aware when telling Alexandrian stories about *centers* (a key 'property') that we don't find ourselves drawn into centralized IT systems that we don't really want.

User Stories bridge the 'affairs of the world' and software implementation. Specific and idiosyncratic rather than generic, this story type should remain ambiguous as long as possible, permitting novel interpretations to surface.

User stories detail the interactions among agents (some will be manifestations of executing computer code) as they collaborate on a discrete piece of work. They establish a unit of work—and a unit of *Theory*—with each story depicting a discrete 'affairs of the world.'

As a 'unit of work,' a User Story bounds an activity with three inter-related parts or aspects: 1) expanding the domain *Theory* and clarifying it through emerging contextualization as one story informs the next, 2) deciding (highly subject to change) what behaviors can and should be embodied in a software-driven artifact, and 3) making the initial decisions that will shape the *conceptual construct*.

The Product Backlog, along with the emerging executing software, comprise a gestalt comprehensive *Theory*. Stories, we would argue, should never be removed from the Product Backlog but simply be changed in some visual way to indicate that they are 'conjecture,' 'developing,' or 'confirmed.' In this way the Backlog provides both an overview of the state of the project while evoking the current *Theory*.

A User Story artifact, a story card for example, is but a placeholder for the conversation that disambiguates those affairs and limns the means by which the software will eventually 'handle and support.'

Objects are actors in User Story plays, anthropomorphized individual system elements engaged in productive activity. The object metaphor assures decomposition of the most complicated domain and the most equitable, simplest, distribution of workload across all domain elements.

Object stories are user stories from the point of view of the character—i.e., *"when I (an object) found myself in this situation I did this*." Object stories don't tell us anything about how the character was able to do what was done but they do provide clues as to how software objects might be implemented. From the stories we can discern:

- what that object needs to know and how he came to know it
- how object behaviors are invoked, and
- the nature of each discrete action expected of the object.

Object stories also preserve the verisimilitude between objects and their software implementations—a major requirement if we are to realize enterprise-software integration and integrity.

Patterns, as stories, are actually found across the entire transition zone, depending on whether they are stories about consistent program implementation (Design Patterns) or domain patterns (Analysis Patterns). Patterns tend to abstract similarities and codify structures for subsets of the domain. Patterns can deal with organizational structures and with structured interactions. They might be conceptual patterns or very low-level coding patterns. Kent Beck's *Best Smalltalk Practice Patterns* and the *Programming Pearls* books tell pattern stories right down to variable name.

In the software community we often think of patterns as another syntax, but in story telling we're pointing to a much wider human ability to think about motifs, such as Joseph Campbell who reduces all hero journeys down to the skeletal 'pattern' of 'the hero's journey.'

Formal Stories, such as data structures and algorithms, are, at least in principle, provably correct. X may equal Y but both are semantically empty. The proof aspect is of interest to theoretical computer scientists but most consumers of those stories only care about the fact that they consistently deliver expected results. Formal stories describe computer operations that are the actual mechanism behind how objects do what it is that they do.

6. Contextualization: Every Story Has Baggage

Bitter, with baggage, seeks same
Personal Ad*, Santa Fe Reporter*

Stories are nested, semantically contextualized, by all the stories to their left on the continuum defined in *Figure One,* and related to peer stories by virtue of a shared context.

Context provides the basis for consistency and for transition from simply valid stories to reliable stories. Context, reinforced by feedback from the emerging software, reinforces the plausibility of the developing *Theory*.

There is a subtly strong determinism in the way context shapes the telling of nested stories. Two examples:

Don't Provoke the Gods

> Imagine a South Sea island blessed with blue lagoons and handsome people who pray to jealous gods who resent mere mortals taking their stuff, like fish from the sea and trees from the forest. This divine resentment leads to occasional 'punishment' in the form of hurricanes or volcanic eruptions. (Although at this writing we may have it wrong about the island's location being in the South Seas. It could be Iceland and an irate god is using a local volcano to shut down air travel.)

Such myths will shape various South Sea rituals concerned with daily activities (building a house, for example) that include appropriate rites for propitiating the gods. In turn, within the context of these rites, specific practices evolve (using stilts for house support and curing logs before they are used).

A Myth Under Our Own Bed

> Need the readers of this essay 'imagine' our own cultural myths about heroic CEO's who steer their organizations by virtue of their near omniscience and flawless application of scientific management in a capitalist world unfettered by puny governments? (Think Ayn Rand.) This kind of myth leads to ritual behaviors, such as the development and deployment of monolithic integrated systems that put corporate information and controls in one place as an instrument to be played by such godlike figures. Embedded in such rituals are specific practices, such as offering the CEO appeasements in the form of obscene salaries and private planes and adopting formal procedures like RUP and formal models like UML. **(**A volunteer, perhaps, for writing up relationships between Tim Geithner, the Federal Reserve, and Goldman Sachs as a case in point? Or how about crafting the mythic story of *Too Big to Fail* according omnipotence beyond civil constraint. And does the reader recognize that Alexander's quaint stories about non self-consciousness in the African builder (who sees spirits in the mud bricks of his hut) apply just as well to our own goings-on?)

The business practices that end up being adopted because of their mythic influence (and the results that accrue from their use) have the effect of putting the enterprise in a technological straight jacket of ossified structure and infrastructure. Clearly we need better myths predicated on more perspicacious thinking and improved construction of stories that are derived from such myths. We need abductive and *metis-ical* thinking.

7. *Metis* versus *Techne*

So, clearly there is a qualitative difference in the kind of thinking, reasoning, and story telling required to develop domain stories as opposed to implementation stories.

Domain, Object and User Stories require what we will call *metis-ical* thinking (we're bad at it) and those about the machine and its innards require *techne-ical* thinking (we've got it down).

Metis (as a noun and from the French *métier* meaning profession, trade or craft) is a form of logic and 'thick' knowledge that is contextual, particular, and fine-grained: the kind of knowledge that can be acquired only by constant adaptation to changing circumstances and long practice at similar but rarely identical tasks (our Swiss village weavers, for example). Such 'thick' knowledge can only be acquired when teams are constantly exposed to the full complexity of task, materials, customers, and context. Perceptive innovations and productive adaptations are absolutely dependent on the kind of wisdom, experience, and 'thick' knowledge provided by *metis.*

Metis incorporates the kind of inference that Charles Sanders Peirce named 'abductive.' Peirce posited that no new idea could result from the mere application of deductive or inductive reasoning, but rather, new ideas come into being by way of "*leaps of the mind*" using "*inferences to the best explanation,*" in much the way that our familiar hero, Sherlock Holmes, unravels his mysteries. (*If* the butler did it, then all our collected but disparate clues fall into place.)

The bulk of software development literature and education concentrates, however, on *techne*, 'thin' reductionist forms of logic and knowledge that aim to be universal ways (math for instance) to render complexity more simple and 'legible.' Our currently 'thin' modes of teaching attempt to pre-codify everything in a vain attempt to protect the learner from surprise (and thereby also buffered from instructive disappointment), whereas a thicker apprenticeship system would aim for the opposite, teaching the learner to expect and embrace surprise and be attentive to subtle but meaningful differences. *Theory*, as conceived by Naur, is *metis-ical* first and *techn(e)-ical* only when absolutely necessary. We also need to re-cast many of our user stories and object stories in a more *metis-tical* light so that they foreshadow but do not unnecessarily constrain technical solutions.

from *Representation and the War for Reality*
by William Gass

"But the war for reality is not to be won or lost in the quarrels between historians and scientists or fictionists and philosophers, because each discipline has its thick and its think side, its solid terms, and their invisible relations. There must be data; there must be observations; there must be facts, incidents, events, the Thick side says, while the Thin reminds us that there also must be order, structure, analysis and argument, no philosophy; without arrangement and connection, no history; and without rhetoric, without patterns, without coherence, there is only the ordinary novel.

The war for reality is therefore a struggle between data and design, brute dumb fact and indifferent chance; and then ironically, there appears the thinnish impulse to introduce the laws of probability into this rubbish heap, and to publish, shortly, a careful catalogue of trash. If design wins the data are deformed, the system runs ahead of the load it was carrying, and there is a multiplication of artificial entities and metaphysical myths. Thins really wonder whether facts aren't entirely the creation of abstract schemes, like the golf courses, lakes, and tree-lined streets of desert real estate developers; and Thicks tend to think that concepts carve continuities into discrete chunks, that laws are lies of some system of society, some secrete legislature illegally elected."

Figure 2. Thick versus Thin

Metis-ical thinking and abductive reasoning are essential for dealing with the world outside the computer, for it is in this world that we encounter 'wicked problems' not resolvable with mere logic and inductive or deductive reasoning.

Apple's Conundrum

> Apple needs a business model which both guarantees a quality user experience and maximizes market share through third party contributions to the functionality of the platform (basically, lots of applications.) There are unknowns in the equation plus conflicts between the forces that have to be resolved. In order to control the quality of user experience Apple wants to limit applications to three languages (C++ which is obsolete, Objective C which little used, and Java Script which is popular but lacks respect). What happens as a result of that decision? Out the door go potential contributors who, rather than becoming loyal supporters, are alienated by a perceived arrogance on the part of Apple. Should these potential contributors decamp and head for another platform (say Google) then Apple loses market share.

Wicked problems elude known categories and approaches. Any attempt at devising a solution to a wicked problem changes your understanding; any solution changes both the problem and your thinking. Moreover, there is no clear rule for knowing when you have 'solved' the problem or what parameters the solution might exhibit.

Perhaps if experienced craftsmen (think *metis-ical*) at Apple increased the scope of their thinking to include the larger system in which their products are being used, they might by "*leaps of the mind*" intuit different kinds of decomposition, redefine the nature of application and platform, and make the choice of language irrelevant.

8. Thangkas and Other Tricks of the Trade

Remembering and sharing (even depicting) a plethora of stories is less intimidating than it sounds. In fact, the way the human mind is put together, we have no problem storing thousands of stories in memory, recalling them to mind when presented with appropriate triggers. Just think of all you know about your culture, your world-view, your profession, your business, your family, in story form.

That said, hundreds of stories are never 'in residence' in our heads at any given time. We need access to the stories, and we need some kind of meta-knowledge about the stories and how they are all inter-related. We also need some kind

Figure 3. The Thanka Over Dave's Desk

of 'index' to the stories so we can quickly find and use the one(s) of immediate interest.

The strategy of archiving or stockpiling stories, however, is a slippery slope, as any good librarian will tell you. You have to know a lot before you can begin to retrieve additional information and without it the information is as good as gone. Rather than a storage model, let us bank on the fact that stories are not frozen but change with time and the intentionality of the teller. [If a six year old tells how he fell in love with his first grade teacher, it is one story. If that same individual at sixty revisits the story, it is something else entirely.]

It is possible to compactly and visually represent a vast collection of continually unfolding stories. On the wall above the desk where the authors are working is a Buddhist Wheel of Life Thangka Painting. Around a hundred discrete images on this painting appear there, each capable of recalling stories. The images are arranged in particular ways which recall yet more stories. The overall gestalt brings to mind still more stories.

Agile workrooms employ similar (but less organized) visual artifacts for a similar reason—to create a kind of external memory with evocation-based retrieval. Big visible charts, information radiators, white boards, post-it notes, and story cards are artifacts that are deliberately placed on common walls so that everyone can simply look up, find the relevant 'trigger artifact,' and recall to mind the story(ies) they need for their immediate work. Less visible but playing the same role are tests, test suites, and the totality of the source code for the software (collective code ownership).

While the markers, acting as cues or clues, exist in the artifacts of our environment, the *Theory*, as Naur insists, exists only in the heads of those that participated in its construction and ongoing elaboration. If you break up a team, and the environment full of artifacts (a sort of external hard drive), the *Theory* itself is broken up and over time dissipates entirely.

Central to our argument is the realization that a proper *Theory* is comprehensive; i.e. it is comprised of *all* the stories. A necessary corollary is that there can be but one *Theory*, the *Theory* of the enterprise and the world in which it operates.

This assertion is in stark contrast to the situation in most organizations today where, at minimum, two contrasting theories compete; one of the business system and another of the IT system. (Thus preventing the long sought goal of business-IT integration.) In the worst case, there are multiple, episodic, ephemeral, idiosyncratic, and unrelated theories of discrete projects.

Everyone in the enterprise must share the same *Theory,* even if they understand some stories in more detail than others. When everyone shares the same *Theory*, and when the evocative artifacts are no longer isolated in an Agile team room but distributed across every bulletin board, logo, intranet, and corporate report, then, but only then, the enterprise will realize two significant benefits. One, there will always be a sufficient mass of theory-containing- minds to assure the preservation of knowledge and enculturation of newcomers. Two, the foundation will be laid to realize enterprise goals of adaptability and innovation.

9. Alexander's Schemata in *The Nature of Order* (or Alexander Ponders God)

Earlier, experienced story-tellers like Beck, Wirfs-Brock and Gabriel looked to Alexander as a potentially rich source of new metaphor and new idiom. Unfortunately, others, like the Gang of Four (Design Patterns) and The Three Amigos (UML), saw only a new form in which to express old ideas, thereby throwing out a lot of that proverbial baby with the bath water.

Our quest now will be to show how *The Nature of Order,* which carries forward much of the mystical and domain focused questions of Alexander's earlier works, can help us tell better stories.

Good Shape

Bad Shape

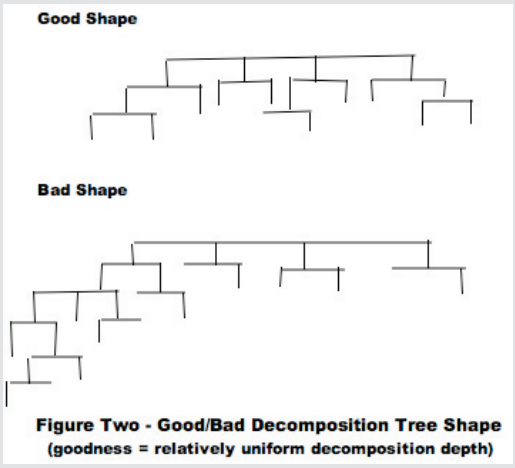


Figure 4. *Good Shape* in Chairs and Decomposition. Objects with Alexandrian *Good Shape are* usually composed of smaller elements themselves of *Good Shape,* often convex.

Alexander's magnum opus runs a hefty 2000 pages divided into four volumes, only the first two are pertinent to our discussion here.

Volume One presents fifteen image-schemata (labeled 'geometric properties') which, Alexander argues, are always present in natural artifacts made by Mother Nature (God, if you will). The presence of these same 'properties' universally characterizes successful man-made artifacts while their absence indicates failed design. (Figure 4 offers but one example.) It is Alexander's expectation that further research will provide mathematical proof of these properties, moving his insights away from loose qualifiers and more squarely within a 'scientific' discussion of, quite literally, the nature of order. However, a different and more modest interpretation is taken here: we limit ourselves to using these properties as convenient proto-scientific schemata of human perception—which in the epistemology of Charles Sanders Peirce are *unsaturated predicates* or *rhemic iconic qualifiers*. Peirce's idea of 'rheme' or 'unsaturated predicate' refers to our attention being on only one aspect of our experience. When, for example, we chip off a piece of old painted plaster or brush some red paint on a piece of wood or cardboard to take to the hardware store in order to purchase a matching color, we aren't concerned with the size, shape, or cost of the plaster, wood or cardboard, but only the 'redness' or 'blueness' of the color. 'Rhemic iconic qualifiers' are useful shorthand expressions for talking about a variety of related experiences (hence the expression 'unsaturated predicates'). Think of 'elasticity' as a rheme, i.e., a way of talking about how different materials act when we work with them or 'gravity' as our experience of various falling objects or, for that matter, our own bodies as we slip on ice or climb stairs. Alexander's '*echoes*,' (one of the 'properties') for example, allow us to speak about an experience of unity across a range of natural and man-made objects, city skylines, bluegrass music, sibling resemblances, or Shaker furniture. Coupling and Cohesions are examples of rhemes or unsaturated predicates that have been found to 'echo' across natural and man-made objects, including software objects.

Image-schemata offer ways of observing, knowing, communicating, and inventing. The risk, ever-present and ever so easy, is to reify these 'rhemes' into reductionist or operational recipes—a job they are ill suited to do.

Volume Two, the last to be published and, in significant ways, the least complete, adds the dimension of time to the initial static presentation of the fifteen properties given in Volume One. The properties are explored in the second tome as transformations, i.e., the actual mechanisms of living systems as they unfold over time. A video, for example, of a developing human fetus shows growth through increasing differentiation: parts increase in *contrast*, growth is through *local symmetries* at different *levels of scale*, both individual

cells and proto-organs develop *boundaries* or transition zones. Simultaneously we see that no *center*, be it heart, lungs, or pinky finger, will be isolated, but rather there is *not-separateness*. [Each of the words in italics is one of the fifteen properties.] At any point in the evolution of fetus, the fetus is whole unto itself, as is the born child who continues on to adulthood. Then, Alexander points out that a parallel process of differentiation and increased density of overlapping centers can be observed in the unfolding of successful buildings and towns. Alexander does not offer design instructions as to the order in which the properties should be conjured forth as inspiration. Rather it is the understanding or vision of the whole which guides the process. In Enterprise Domains we find something of a parallel. Paul Hawkins in *Growing a Business* speaks eloquently for the need of deep rooted knowledge and intuitive understanding of a business as a 'whole' if that business is going to continually learn, survive, and 'unfold' well. The failure rate and short life span in business speaks to the difficultly of such 'thick' domain knowledge.

10. What Every Schoolboy Knows. . .

Before we delve into domain stories (our Agile weakness) and how Alexandrian image-schemata (our illustrative example) can spur good thinking, four preliminary comments may provide a useful entry into the subject matter.

First, we want to underscore how 'un-Cartesian' the fifteen schemata really are, and, in that sense, helpful in shaking programmers out of their doldrums and habitual ruts. Alexander's properties are more in tune with what phenom-enologists call *lived space*, i.e., that day-to-day unexpressed and somewhat inexpressible sense of space that we all have as we orient ourselves from our source—our bodies—into the surrounding environment. User experience designers are grappling with exactly this kind of problem. *Roughness* (one of the fifteen properties), for example, honors the relaxation and disclosure of the full range of bodily sense and posture—fluid, not completely open, but not that narrow. Designers of all stripes want a brief, a project charter, with exactly this kind of *roughness*. Other properties, *good shape, positive space, contrast, boundaries, gradients*, clearly make more *lived space* sense than the razor thin A/not A categories from our geometry lessons. Alexandrian space reveals a rich field-like structure rather than space which is empty, transparent, unstructured and isotropic. Alexander would subscribe to the following quote from Newton's Latin version of *Optics, Query 20* (for Newton, too, had his moments of metaphysical wonder), "*Annon spatium universuum, sensorium est entis incorporei, viventis, et intelligentis*?" (Is not infinite space the sensorium of a being incorporeal, living and intelligent?) [Newton removed the phrase from the English version when he was in a knock-down-drag-out-fight with Leibniz, perhaps not wanting to appear as *too* mystical.]

As a second point, we need to make a special kind of spatial translation of Alexander's schemata. We need to think of 'space' as something else even though we use spatial metaphors (balance of power, span of control) when describing the enterprise domain. Several candidates come to mind: power, wealth, control, even market, but we will suggest another dimension, that of behavior, as the most appropriate candidate. For example (and an important example) an Alexandrian *center*, is not a point-center of a geometric circle but, rather, a focal zone created by the field around it, just as we experience 'centers of power' or 'centers of action,' in an enterprise through various kinds of behaviors.

Most of the time, in Alexander's work, *centers* are discussed in terms of two and three-dimensional space. However, when time is introduced, there is a sense of N-dimensional space unfolding, where the essence is preserved even as the manifestation of that essence is transformed. For example, if we take our story of the Swiss village and its original cottage industry, we can imagine that the *centers* have evolved quite significantly over time, but there is still continuity of an essence that remains alive.

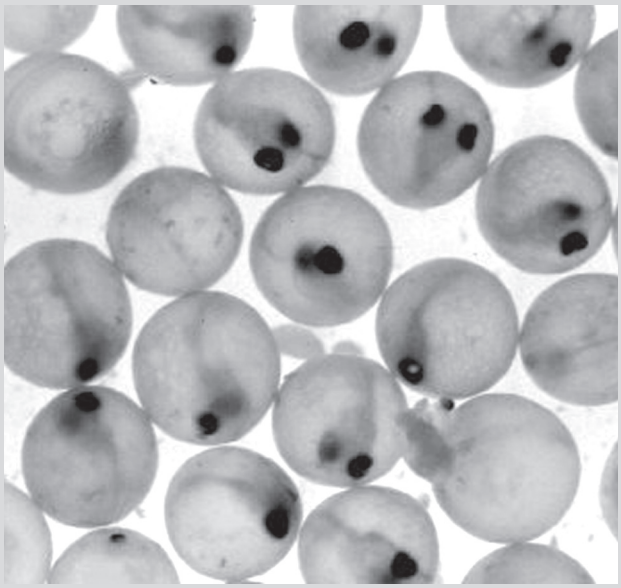


Figure 5. Fish eggs. *Roughness* is in how irregular three dimensional objects find their place and allow other to find their place.

Roughness in Schopenauer's Porcupine Story

One cold night the porcupines huddled together for warmth but pricked each other with their quills. After they shuffled in and out for a while, each found just the right spot for both warmth and comfort.

Third, a few authors (Régis Medina, for one) have looked to Alexander's new material as a way to improve implementation stories, and while we acknowledge these contributions, we feel the more insightful angle, and most certainly, the more propitious place to start, is to borrow insights from Alexander to tell stories about the domain.

And fourth, we must tell some mythic stories about the world as context for the Alexander inspired, domain stories.

The Universe

> We believe that the universe is a system composed of nested subsystems. We used to think that the Universe was something like massive clockworks, deterministic and mechanical in nature. But that was an earlier Zeitgeist, although businesses and management schools held on tight to that idea all the way through the 20th century. Now we've moved on to a better understanding and view the universe as a complex, living, system with emergent behavior, significant non-determinism, much closer to an organism or ecology than clockworks.

Of course, most of us are not going to be concerned with writing software for the universe or even Ultra-Large Systems. Instead we will be focusing on specific subsystems. And of course, some of those subsystems will be 'formal.' Examples of this kind of software would include operating systems, firmware, hardware drivers, etc. When developers deal with this kind of static system, they are best advised to use the techniques of software engineering as is. Our interest here, however, is in the design of [sub] systems (and supporting software) that are not formal in nature, but consistent with our contextualizing story of the Universe as Complex System. The vast majority of applications software fits into this category.

11. Excerpts from A Gecko's Tale

In order to show how we might tell stories about a specific domain using Alexandrian image-schemata, we will arbitrarily pick the domain of Insurance and use the Wheel of Life Mandala as a visualization tool.

Let's start with the core of the Mandala—a circle that is a c*enter*. Nestled inside are icons which act as contributing *centers*. These icons represent focal ideas and gather many stories of the primal forces involved. The act of *centering* the generative engine at the spatial center of the Wheel, draws on our primitive image-schemata to reinforce its essence as driver/creator of all else. The Mandala is further strengthened and defined by the *boundary* and its *good shape*. Our initial circle-as-center will *echo* across the Wheel as our work progresses.

An over arching story associated with this space is a variation on *Money Makes the World Go Round* with all the primal forces that generate the insurance domain. We can easily create visual reminders of such forces: money (represented by a U.S. dollar sign), people (a stick-figure icon), assets (a bank vault icon), and risk (a tornado/cyclone icon).

Contrast reflects the categorically quite different risks such as money, people, and assets. As *centers*, we expect all three to be complex and composed of nested centers at different *levels of scale*. We can tell stories about each to bring to mind their essential nature. For example, stories of the *Great Tsunami of 2008,* or *The San Francisco Earthquake, or The Great Chicago Fire*, reveal the nature and the composition of the Risk center. Similarly, we could tell stories about costs, profits, losses, expenses, and capital to better unpack the Money center.

The generative power of the core arises from *interlocked and ambiguously* expressed interactions and inter-relationships among the *centers* in that core. The interlock is sufficiently deep and the nuances of interaction are sufficiently ambiguous that we will never run out of stories to be told. This is a good thing, because new stories are the source of innovation (new products, re-designed processes, new organizational structures) and adaptability. Think of all the new process stories that had to be invented for on-line purchase of policies. Or, just what is an asset?

Jennifer Lopez's Lips Bring in Bucks

> Jennifer's lips, full, glossy, sensual, were the talk of Hollywood after the debut of her film, *Sex Kitten and Dracula*. This spurred Gecko into launching a new insurance product line around collagen and Botox mishaps. (We made this one up.)

Disambiguating the interactions, surfacing and removing the *deep interlock*s, even if possible, does not lead to greater understanding; it merely results in a reductionist, lowest common denominator, cookie-cutter systems design.

Stories about the interplay among the primal movers create the context for the other centers. In our insurance company (a structural center), actuaries arise from stories about the uneven distribution of risk across our customer base (people) and how that distribution affects costs and profits. For example, the sad saga about Sam exposes the *deep interlock* between human hormones and particular kinds of assets to define risk. Then there's Dave, a Harley man with gray hair, whose story exposes another different *deep interlock*.

He Never Reached Manhood

> Sam, on his sixteenth birthday, took his piggy bank and bought himself a 2000cc crotch rocket motorcycle

and took off for Denver. Taking his first hairpin turn at full throttle, he missed the curve, crashed into a boulder, and that was the end of young Sam.

Helmetless Dave

I won't wear a motorcycle helmet. Helmets impair my hearing and hasten fatigue thereby increasing the likelihood of an accident. Marginal Impact injury prevention is replaced by a greater chance of torque injury. And figures show that cowardly but experienced grumpy old guys like me, sans helmet, are at less overall risk than helmeted but reckless teenagers.

To develop a *Theory* of a large and complicated domain, we must partition the domain into more comprehensible pieces—decomposition. Stories and objects are the metaphorical knives we use to carve out our partitions, augmented by an understanding of properties (especially *centers* and *nascent centers* through *boundaries, contrast, void*), old programming concepts, like coupling and cohesion, and metaphors like behavior and homunculi.

Coupling and cohesion traditionally have been misused 'inside out,' as tools to aggregate computer functions and represented data (stuff inside the machine) and not as tools to partition the World. Here, however, we look to the kind of user stories and objects that emerge from telling stories about primal forces to more accurately reflect the natural elements of a domain.

Behavior provides a criteria for decomposition that maps to (and translates) Alexander's spatially defined properties; it represents Naur's 'executing software,' and is a feature common to software, human, and abstract entities in our domain.

Behavior-based decomposition of a business enterprise begins with the people, or more precisely, with job roles and their attached behaviors (duties, responsibilities, and tasks). Processes are simply ordered collections of discrete behaviors.

In our insurance company we see behavior centers named Agent, Underwriter, and Claims Adjuster. We need to extend our thinking about behavior to the 'inanimate' elements of the system as well. For example, what are the behaviors of a document? Using action verbs forms, a document:

- •Collects a set of pages
- •Adds, deletes pages from itself
- •Orders the pages in collaboration with the individual pages
- •Provides a page on request
- •Identifies itself
- •Presents itself, also in cooperation with individual pages

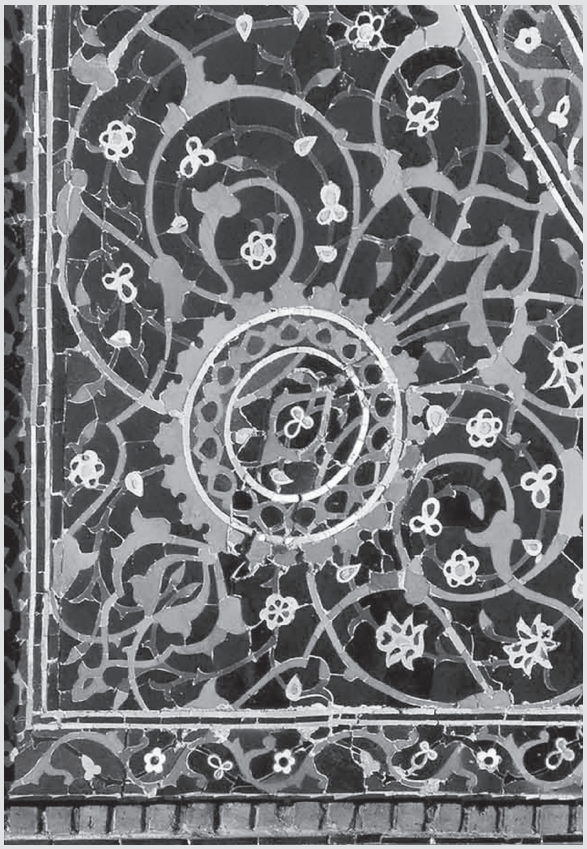


Figure 6. *Deep Interlock and Ambiguity* in a tile from Samarkand and the lack of this 'property' in Sherlock Holmes tiles from Baker Station in the London underground. Alexandrian *deep interlock and ambiguity* is about two centers being joined by a third which belongs ambiguously to both. A meandering stream or coastal wetlands would be examples from nature.

Figure 7. *Positive Space.* 'Goodly shaped centers will create around them well (positive) shaped space which units the separate centers into a larger cohesive one. Even in movement, dancers or aikido practitioners who are 'centered' in themselves will form a fluid but continually positive space between themselves.

We want strong centers, and to strengthen them we now use our old friend, cohesion. But cohesion, in this case, means that the behaviors are appropriately aggregated from the perspective of the domain. For example, we do not ask actuaries to investigate claims even though they possess information about accidents that would allow them to do so. *Centers* are strengthened (cohesion is increased) when there are *boundaries, good shape, contrast,* and *simplicity and inner calm.*

Centers do not exist in isolation (Alexander's *not-separateness*) but are coupled to other centers. As we know, loose coupling is preferred. A number of properties contribute to loose coupling of centers: *positive space* (which loosely couples adjoining centers, such as the dancers in *Figure 7*) and *void* are two. [For *void*, think of how a public square (*void*) decouples surrounding parts of a city by offering multiple pathways among them. In contrast, two parts of a city connected by a single bridge are tightly coupled. In software, multiple objects connected by a single name space (bridge) are tightly coupled, whereas the same objects connected by a directory (a container of multiple possible connections or, in Hillier's lingo a 'carrier space') are decoupled.]

Other properties, *echoes, gradients,* and *alternating repetitions,* provide additional ways to couple centers, or at least establish relationships among them. These are more akin to the way that we organize departments or processes as relationships among centers. We apply similar coupling and cohesion principals here because we would like our processes to be as adaptable and flexible as possible. Examples would include:

• The way that Cray Research *echoed* both the physical layout and the organizational structure of the small Chippewa Falls office where Seymour Cray and two associates wrote the Cray One operating system across their entire campus complex in Minnesota.

•Well crafted, and addictive, games offer a challenge *gradient* that can ascend as user experience and expertise increases.

•The *alternating repetitions* that generate the rhythms of agile development: build (test & code), then improve (refactor) or listen (user stories and on-site customer), then interpret (test and code, again), or then, do (sprint), and reflect (retrospective).

Deep interlock and ambiguity is a 'property' whose name seems to suggest stronger coupling than might be desirable. But in fact it captures the essence of another well-known software principle—the association. (A Reservation center resolves the *deep interlock* and attribute dependencies between Room and Event centers, establishes appropriate loose coupling, and preserves the *ambiguity* of 'what event might be in what room when' until a concrete instance must be scheduled.)

The astute reader will notice that Alexander's schemata brings us, by another route, within range of the behavioral approach to object design à la Ward Cunningham, Kent Beck, Rebecca Wirfs-Brock, Nancy Wilkerson, and Dave West that was generally dismissed as informal, messy, artsy, and hopelessly devoid of strong manly type systems.

Nor would this behavioral approach be foreign to scientific inquiry.

Maxwell's Demon

> In a thought experiment about equilibrium in gas pressure, Maxwell's demon is a hypothetical homunculus that has a behavioral obsession of standing in the doorway connecting two compartments in order to admit or block the passage of individual molecules between the two chambers. The demon's goal? Balance out the distribution. If provided with information about the speed of individual molecules (and a bit of malice), Maxwell's demon would be able to violate the second law of thermodynamics.

A final example, this time of thinking about design and foreshadowing implementation. The properties of interest are: *levels of scale, echoes,* and *roughness*. We will also return to our Thangka metaphor.

The outermost circle of the Wheel of Life depicts the cycle of life, the sequence of stages from birth to birth. The circle itself is a sequence of segments, each segment representing a life phase and each segment being a cycle unto itself. So the Circle of Life is an object, specifically, an ordered collection that consists of phases, that are themselves objects, most often also an ordered collection.

The Wheel of Life for our insurance company is a process, or more accurately, an ordered collection of processes identified as:

Identify Prospect
Collect policy information
Rate policy
Underwrite policy
Issue Policy
Collect premiums
Accept claim
Terminate policy

Roughness comes into play when you recognize that each of the component parts of the overall process are stronger to the extent they are autonomous and not tightly coupled, yet allowing the others to 'find their place.' The elements themselves are diversely shaped and loosely coupled, making it possible to completely re-organize or restructure the entire process merely by changing components, or the interfaces among the components. Organizations that ruthlessly pursue total integration (à la SAP) remove *roughness* and hence adaptability.

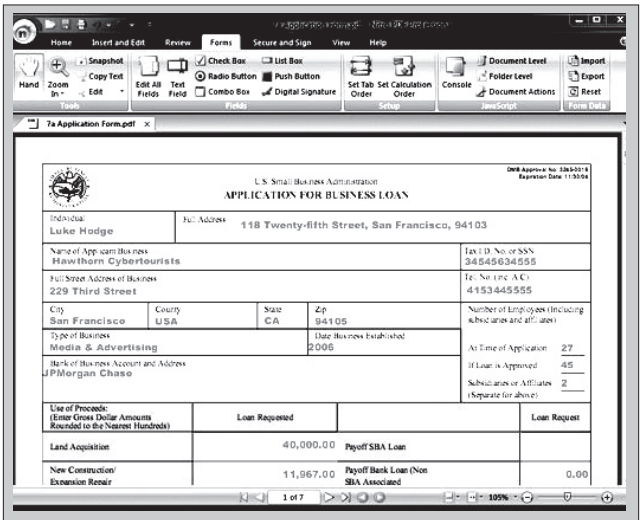
Echoing our 'collect policy information,' (which is an ordered collection of processes), display data collection forms must accept input, validate input, and submit validated input. Similarly, the issue policy process would be a process collection: add page, create page, add 'boilerplate,' add fact, and add text string. In this example we see both *echoes* and *levels of scale* as individual objects, such as data entry form and policy, have the same 'ordered collection of objects' aspect. The form is an ordered collection of text strings and entry fields and another, unseen, ordered collection of validation rules. The policy is an ordered collection of pages, and a page is an ordered collection of text strings, glyphs, and values. A text string is an ordered collection of characters. A validation rule is an ordered collection of operators, constants, and variables.

12. Have You Heard The One About. . .

In our insurance example, we have used image-schemata-inspired thinking to decompose a complicated domain and design components that could be directly and simply implemented as software. There is not an algorithm, data structure, mathematical formalism, or logical argument in sight. We have shown the potential of *metis-ical* thinking and laid a foundation for developing the following corollary insights.

The first is theoretical and speculative. One of the more mystical '*metis-ical* topics in *The Nature of Order* is 'unfolding.' Alexander, true to his dual nature approach, is looking to emulate the natural behavior of a seed (as it unfolds through several stages to become a flower or tree), to create a kind of evolutionary design of the artificial. The behavior of the seed is grounded in the structure of its DNA which is an ordered collection of genes, which are ordered collections of bases. The ability of a single instance of a seed to unfold into its ultimate form is encoded in its DNA, so too is the variability of forms from essentially identical instances of DNA. Variation in form comes from context—environmental factors that replace or activating different members of the ordered collections, such that different unfoldings occur.

In our example of the data collection form we have something that is metaphorically and behaviorally similar to the seed. A form is essentially a nested set of ordered collections (à la DNA string, gene, base). A finished, behaviorally active, form unfolds from the operation of the elements in those nested ordered collections. Variability of form (order form, job application, insurance application, etc.) requires simple addition, deletion, or substitution of elements in one or more of the ordered collections—as to which change will be appropriate, that will be determined by the stories that contextualize the unfolding.



"Things are stories."
Maurice Merleau-Ponty

Figure 8. A Form 'Application for Loan'

Second, pragmatic benefits have also been obtained. Adaptability is readily apparent. An enterprise (if understood and modeled behaviorally) can change any of its processes or any of its component objects (including translating them from the tangible to the digital) by adding, deleting, substituting, or re-ordering elements of its nested ordered-collections. The property, *simplicity and inner calm (*a de-cluttering of all unnecessary centers leaving the remaining ones stronger*)*, is also quite real. The core of the data entry form—which is all possible forms—can be implemented in less than 100 lines of Smalltalk code. Those hundred lines are distributed across five objects (classes) and roughly thirty methods, so every code statement or method is trivial to understand.

Third, we have reinforced Agile stories and a Lean principle, by using those stories to delay the departure from the ambiguous (ambiguity being simply maximum possibility).

Fourth, Peter Naur's insights and Agile's concept of storytelling provided the bridge to cross from familiar mind sets to those that are strange. We have also opened the door for consideration of other essential sources of insight. Networks, weak links, design thinking, and complex adaptive systems are but four such sources.

In essence, we have shown why more stories should be told, and object and user stories recast as ambiguous and dynamic through the use of proto-scientific schemata. And, really, folks, stories are first and foremost, a way to improve our sense-making skills.

The moral of our tale is that we have a choice. Machines are sterile. Dead as dead can be. Only living systems are fecund, intellectually productive, generative. The most compelling enterprise need, today, is the ability to innovate, adapt, and learn. These are characteristics of living systems, not machines.

Software can be either dead or 'alive' in how it 'handles or supports' and it all depends on the philosophical presup-positions—our myths of the World—that contextualize our *Theory*. We can choose.

Our profession would do well to eschew what we think we know and diligently explore what others are learning about living systems. Software developers could nurture their inner artist. And, just as other kinds of artists are drawn to the marginal, counter cultural, night life areas of a city, we need a software development environment that is similarly loose, permissive, and tolerant of the subliminal. We need story telling as our primal social action where an utterance to an audience is a transfer to the collective.

Acknowledgments

We thank André Demailly, Charles R. Adams, and Mary West for their invaluable comments on early drafts.

13. Annotated Bibliography

Alexander, Christopher. *Notes on the Synthesis of Form,* Cambridge: Harvard Paperback, 1964. Alexander's published Ph.D. exposes his struggle between his training in math and logic and mystical intuitive sources of knowledge.

Alexander, Christopher. *The Timeless Way of Building*, New York: OUP, 1979. The Big Picture on 'the why' of architecture—which is to achieve *QWAN*. The Quality Without a Name is redefined in his latest work as God.

Alexander, Christopher, Sarah Ishikawa, Murray Silverstein, with M. Jacobson, I. Fiksdahl-King, S. Angel. *A Pattern Language*, New York: OUP, 1977. The how-to book which supplements *The Timeless Way* on 'the why' of architecture.

Alexander, Christopher. *The Nature of Order*, Berkeley, California: CES, 2003-2005. Four volume magnum opus attempting quite literally to decipher the nature of order in the universe and how to harness its secrets.

Bronowksi, Jacob. *The Visionary Eye: Essays in the Arts, Literature, and Science*, Cambridge, Massachusetts: MIT Press, 1978. Bronowski makes the pitch that art and science are but two compatible and mutually reinforcing sides of man's inquisitive mind.

Brooks, Frederick P. "No Silver Bullet: Essence and Accidents of Software Engineering," *Computer*, Vol.20, No.4 (April 1987). As explained in the essay, he looks at the conceptual construct behind programming.

Gabriel, Richard. *Patterns of Software*, Oxford University Press, 1996. The use of patterns thinking in software.

Gass, William. *Habitations of the World*, Cornell University Press, 1985. Essays on writing and source of quotes by Gass and Robert Louis Stevenson.

Geertz, Clifford. *Toward an Interpretative Theory of Culture*, New York, N.Y.: Basic Books, 1973. A collection of essays, the first of which looks at 'thick descriptions' as a major methodology in ethnography.

Hawkins, Paul. *Growing A Business*, New York: Simon & Schuster, 1988. A book for entrepreneurs speaking about how a business 'unfolds,' but getting the sequence of business decisions right requires in-depth understanding and 'feel' for the industry.

Hillier Bill and Hanson J. *The Social Logic of Space*, Cambridge: Cambridge University Press, 1984. Basic text on space syntax. Provides concepts such as pathways, access, and intelligibility absent in Alexander's work.

Holton, Gerald. *The Scientific Imagination*. Cambridge, Harvard Press, 1998. Case studies of inquiry which reveal the subliminal schemata at work.

Johnson, M.L. *The Body in the Mind: The bodily basis of meaning, imagination and reason*, Chicago: University of Chicago Press, 1987. Classic text on the role of bodily metaphors in thought.

Medina, Régis www.regismedina.com Régis Medina of Crossbow Labs in France keeps an interesting blog on the using concepts from *The Nature of Order* in implementation design.

Naur, Peter. "Programming as Theory Building," *Microprocessing and Microprogramming*, 15, (1985).

Peirce, Charles S. *Values in a Universe of Chance, Selected Writings of Charles S. Peirce*, editor Philip Wiener, Doubleday, 1958.

Quillien, Jenny. *Delight's Muse: on Christopher Alexander's The Nature of Order*, Ames, Iowa: Culicidae Press, 2008. A beginner's set of cliff notes on the main themes of the four volumes.

Quillien, Jenny. *Haunted Spaces for Lesser Gods: A more modest interpretation of Christopher Alexander's Findings in Book Four of The Nature of Order*, paper prepared for EDRA (Environmental Design and Research Association), Washington D.C., June 2010. Available from the author upon request. Paper proposes a secular explanation rather than Alexander's theistic one.

Scott, James. *Seeing Like a State: How Certain Schemes to Improve the Human Condition have Failed*, New Haven: Yale University Press, 1999. An account of how the State has required legibility in order to govern and imposed 'thin' techne over 'thick' metis knowledge.

Shank, Robert. *Tell Me A Story: A New Look at Real and Artificial Memory*, Chicago: Athenaeum Press, 1991. A look at how and why stories serve human cognition so well.