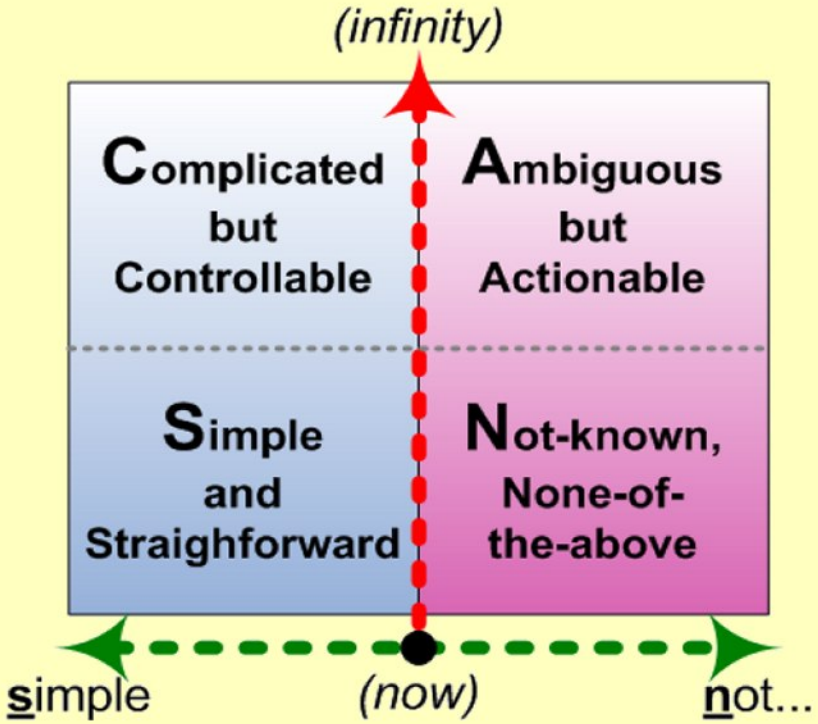
SCAN

a framework for sensemaking and decision-making - *the Tetradian weblogs*



Tom Graves

Tetradian Weblogs Series

SCAN framework

The Tetradian weblogs

Tom Graves

This book is for sale at http://leanpub.com/tp-scan

This version was published on 2014-05-31

Also By Tom Graves

The enterprise as story

Mapping the enterprise

Everyday enterprise-architecture

The service-oriented enterprise

Doing enterprise architecture

Enterprise Canvas

Fazendo Arquitetura Empresarial

Contents

Introduction

This ebook is a collection of articles from the Tetradian weblog[1], on the SCAN framework. This updated edition includes all weblog-posts on SCAN up to the end of January 2014; there'll be further new editions of this ebook from time to time, to incorporate later posts and practical experience.

SCAN is a simple yet powerful framework for sensemaking and decision-making, and, recursively, for assessment and review of processes and results of sensemaking and decision-making. The first version of the framework, with an initial focus only on sensemaking, was born in November 2011 with a catch-phrase that came up out of the blue one morning: **"Let's do a quick SCAN on this"**.

In essence, SCAN is a form of *context-space mapping* – a means by which we make sense of a context, and work towards resolving our requirements in that context, by bouncing back and forth between the 'problem-space' and 'solution-space' in a disciplined way. There's more detail on context-space mapping as a generic technique in my book *Everyday Enterprise-architecture: sensemaking, strategy, structures and solutions*[2].

[1] http://weblog.tetradian.com

[2] http://tetradianbooks.com/2010/05/everydayea/

A key part of the background behind SCAN had been almost a decade's-worth of frustration and worse in trying to use the well-known Cynefin framework in my work in enterprise-architecture and the like. Cynefin is well-suited for sensemaking in some specific aspects of complex adaptive systems, but was not (and I believe still is not) a good fit for enterprise-architectures, particularly around uniqueness and real-time action. I summarised some of the key differences in approach and purpose in the post **Comparing SCAN and Cynefin**.

The first version of SCAN wasn't quite right, of course, but "with a little help from my friends" – as **SCAN – an Ambiguous correction** explains – we started to home in towards a more consistent and usable frame.

Real-world examples help in this, too - as in **Using SCAN: some quick examples**.

Of course, there are plenty of other frameworks that incorporate some form of systematic sensemaking and decision-making. **On SCAN, PDCA, OODA and the acronym-soup** summarises comparisons with two of the better known frameworks: Deming's PDCA (Plan, Do, Check, Act) and John Boyd's OODA (Observe, Orient, Decide, Act).

Ensuring that the Simple stays simple provides a quick example of using SCAN to explore differences in what is experiences as 'simple', and what is not.

By this point I needed to describe more of the theoretical background behind SCAN. **SCAN left**, **SCAN right** shows

how the visual layout aligns with the often over-hyped yet still-useful concept of 'left-brain' versus 'right-brain'; whilst **Domains and dimensions in SCAN** provides more explanation of the underlying axes of the framework, and the resultant implicit sensemaking-'domains'.

The four-part series **On sensemaking in enterprise-architecture** explores two of the key focus-areas for SCAN – uniqueness, and sensemaking and decision-making in real-time action – using the ditching of US Airways Flight 1549 ('The Miracle on the Hudson') as a real-world example.

In **Real-time sensemaking with SCAN** and **Belief and faith at the point of action**, I started to delve more into what happens to sensemaking and decision-making as we move closer and closer to the actual point of action. This is also where SCAN starts to cover decision-making as well as sensemaking.

Then in **Decision-making – belief, fact, theory and practice**, and in the four-part series **Decision-making – linking intent and action**, the decision-making side of SCAN becomes much more explicit, with its own version of the SCAN frame.

Recursion is a key theme in SCAN, and also a key differentiator from sensemaking in Cynefin and suchlike. In **Using recursion in sensemaking** I showed how to use SCAN recursively to explore itself, and explore the exploring of itself, at the same time as being used to explore whatever concern it was being used to explore. Recursion can be very useful, but it can sometimes be a bit tricky to get the head

around…

Competence, non-competence and incompetence uses the decision-making view of SCAN to clarify the knotty problem of incompetence and 'any-centrism' in enterprise-architectures and the like; whilst **Requisite-variety and stormy weather** uses the sensemaking-view of SCAN to unravel the equally-knotty myth of 'control', and introduces the concept of 'variety-weather' – the way in which the underlying factors in a context may themselves undergo unpredictable change, making the supposed certainty of 'control' an inherent impossibility.

The brief two-part series **Rules, principles and the Inverse-Einstein Test** and **Rules, principles, belief and faith** again uses SCAN's decision-making view – and, in particular, the crucial boundary between 'order' and 'unorder', Simple versus Not-simple – to describe the different roles of rules and principles in real-time business decisions. **Checklists and complexity** extends this to show how checklists can be used to straddle that inherently-complex real-time boundary between Simple and Not-simple; whilst **Complex, complicated, and Einstein's dice** extends it further again to to explore how to cope with the context's 'variety-weather'.

The next few posts explore more-extended forms of context-space mapping, using cross-links to other sensemaking-frames. **Order, unorder and effectiveness** applies a useful cross-map with Five Elements – another frame I use a lot in my work – to explore a key relationship between

SCAN's sensemaking-modes and dimensions of enterprise-effectiveness; **Sensemaking and the swamp-metaphor** describes another cross-map to another framework – the 'swamp-metaphor' – that provided key concepts in the theoretical background to SCAN; and **Sensemaking – modes and disciplines** provides detailed 'how-to' instructions on tactics to move around within the sensemaking / decision-making space in real-time practice.

Rounding off the first edition of the ebook, **Enterprise-architecture is wicked** shows how to use SCAN to help in making sense of wicked-problems; whilst one of the two examples in **Unbreaking** describes how SCAN can be used in resolving wicked-problems in actual enterprise-architectures. (The other example uses the Five Element frame described above – with both examples illustrating how context-space mapping works in real-world practice.)

To start this second edition, there's a stream of posts about re-exploring and extending SCAN itself. The first of these is **More keywords for SCAN**, which picks out some alternate terms that could be used for each of the S, C, A and N domains. **A simpler SCAN** presents a radical rethink of how to describe the relationships between the domains, shifting from central-axes to side-axes. **SCAN as decision-dartboard** and **SCAN - some recent notes** show some simple yet very practical applications for SCAN in project-planning and the like. **Problem-space, solution-space and SCAN** and **Knowable and discoverable** explore some alternate ways to frame the type of context-space mapped via

SCAN. And **SCANning the toaster** gives a light-hearted yet definitely real-world worked-example of what actually happens in the sensemaking / decision-making / action context-space.

Four posts explore rather more of the historical background behind the development of SCAN. **On metaframeworks in enterprise-architecture** introduces the concept of a *metaframework* – a framework to create other context-specific frameworks – whilst **Metaframeworks in practice, Part 4: Context-space mapping and SCAN** shows how it applies to SCAN. **Methods, mechanics, approaches** describes one of the earlier models that provides part of the foundations for SCAN, whilst **SCAN and Causal Layered Analysis** shows how SCAN can be used in conjunction with Causal Layered Analysis, a popular techniques for futures-work.

Next, a series of posts on one of the more difficult and controversial aspects of SCAN - the way it tackles the 'Not-known', or, as some would put it, 'the Chaotic'. **Coinspiring in the Chaotic domain** revisits some old and occasionally difficult territory, though from several new directions, whilst **Obliquity, serendipity and purpose** and **Working with 'I don't know'** assess tactics on how to approach and *use* the so-called 'Chaotic'. And two posts – **Control, complexity and chaos** and **Control, complex, chaotic** – explore some of the key distinctions that enforce the requirement for different tactics in different parts of the SCAN context-space.

Whilst most organisations are still structured as disparate sections and silos, much of the work of enterprise-architecture takes place **Between the boxes**, or repairing the damage caused by fiefdoms fragmenting the context-space and **Dotting the joins** – hence why we need to expend so much effort in then 'joining the dots' all over again… (An expanded version of the *Dotting the joins* article was published in the August 2013 edition of the *Journal of Enterprise Architecture*[3]; it's also available on the Tetradian weblog[4], but unfortunately can't be re-published here, for copyright reasons.)

Three posts explore some other practical concerns within enterprise-architecture, and how SCAN can help to highlight some of the respective themes. **On human applications in enterprise architecture** looks at human-based 'applications' such as Amazon's 'Mechanical Turk'; **Principles and checklists** show how to create run-time support for the 'Not-known' parts of the context-space in SCAN; and **Best practices - adapt, then adopt** describes the practical limitations, yet real usefulness, of so-called 'best-practices'. We then have two other posts – **Reframing entropy in business**, and **More on reframing entropy in business** – that explore the *practical* adaptation in business of the physics-notion of 'entropy'.

Continuing on with the theme of practical applications for SCAN, we next have **The over-certainties of certifi-**

[3] https://www.globalaea.org/news/136731/JEA-August-edition-now-available.htm

[4] http://weblog.tetradian.com/dotting-the-joins-jea-version/

cation, explaining *why* so many would-be 'certification'-schemes risk causing more problems than they solve. Along the same general themes, two posts explore the limits to which enterprise-architecture and the like can, could or should ever be described as a 'science': **The craft of knowledge-work, the role of theory and the challenge of scale** and **The science of enterprise architecture**. After that, we use SCAN to assess the question **Can complex systems be architected?**, how to **Avoid delusions of strategy**, and how to contrast **Efficient versus effective** within the business context.

Finally, for this second edition, a multi-part example of how SCAN can be used in tackling issues at much larger scope and scale. These form part of a futures-exercise at literally global scale, *Four Principles For A Sane Society*. Two of these principles are outlined in the posts **Four principles - 1: There are nor rules**, and **Four principles - 4: Adaptability is everything**. There's an overview of the whole thing in **Four principles for a sane society: Summary**; and, in **Four principles for a sane society: An addendum**, some responses to various all-too-common 'objections' that arise against this kind of futures-work. This edition ends with **A kind of manifesto**, which summarises the direction that my own enterprise-architecture is starting to take.

Enough for now, anyway: on to the posts, and let them speak for themselves.

Contents for the Sample edition

This ebook is a collection of articles from the Tetradian weblog[5], on the SCAN framework. The current full edition of the ebook includes all of the weblog-posts on SCAN summarised in the Introduction. This sampler-edition contains the following posts:

[5] http://weblog.tetradian.com

“Let’s do a quick SCAN on this”

What’s a *quick* way to start making sense of some context? – fast enough to help in making good decisions fast, too?

If you’ve been watching this blog or any of my other writing, you’ll know I’ve been working on this one for years, worrying at it like a dog with a bone. My first books were actually about a variant of this, three decades and more ago. In recent years, mostly for enterprise-architecture, I’ve developed or re-used a number of approaches: the tetradian dimensions, classic Five Elements, context-space mapping, all the different themes that came together for Enterprise Canvas, and much more.

There’s a lot there, with a heck of a lot of theory behind it. But that’s the problem: there’s a lot of it, and there’s a heck of a lot of theory. Sigh…

What’s *really* needed is some sensemaking-structure that’s quick, simple, snappy, that’s as easy to grasp, and as ubiquitous in use, as something like SWOT.

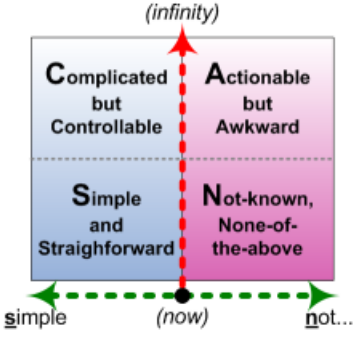
And I think I may have found one, with this:

“Let’s do a quick SCAN on this.”

It's a form of sensemaking that actually comes down to two very quick questions:

- Is it simple, or not?
- How much time do we have to make it work?

Visually, this gives us a simple sort-of-two-axis-framework that looks like this:



SCAN: sensemaking

(You could use the plus-or-minus symbol ± as a quick scrawl for this - particularly as it's both visually-descriptive and gives that sense of uncertainty yet sorting things out, "plus-or-minus a bit, here or there…".)

And *you* choose what each of those domains will mean. It's your context, hence your choice: intentionally, there are no preset 'special definitions' here.

Looking at anything in that space, is it ***Simple***, Straightforward, makes Sense; or ***Not-known***, Not-sure, Not-certain,

a kind of niggling 'None-of-the-above'? That's our horizontal-axis: a simple sort into what we already know how to handle, and what we don't; what we're certain about (for now, at any rate), and what we're not.

(The key here is what we might call the *Inverse Einstein test*. Einstein once said that craziness is doing the same thing and expecting the different results. On the simple side of the scale, that's true: if we do the same thing, we should always get the same results. But if we do the same thing and *don't* get the same results, that automatically pushes us to the other side of the scale – for now, at least, until we've had a chance to do some sensemaking about it.)

If there's no time at all to make sense, but we're dealing with something that doesn't make sense, that's what we'd do: split it straight away into the stuff that that we know, and the stuff that we don't, and then get on with it straight away with the bits that we *can* do. The important part is to *not* throw the 'stuff-that-we-don't-know' into the too-hard basket: instead, we keep it to one side, clearly labelled 'None-of-the-above'.

When we *do* have time – our vertical-axis here – we apply the same test, but stretch it out a bit. Think of it as the Blu-Tack® school of sensemaking, perhaps, because we always start with this sticky blob called 'Not-known' or 'None-of-the-above'. Or perhaps like stretching pizza-dough. Anyway, what we're doing is stretching that 'None-of-the-above' out into four rather more distinct domains:

– Is it **Simple** and **Straightforward**? – we know what to do,

and we can do it fast, with simple rules or simple guidelines.

(In practice, this means that we could probably do it with what we already have, or with something that we can buy off the shelf or train people to do in a couple of days or so. Keep it simple, keep it cheap, keep it working: that's what we'd expect here - or aim for, at any rate.)

– Is it **Complicated** but still **Controllable**? – it might take us a bit of analysis and an algorithm or two, but we're certain we can make it into a predictable, predefined, packageable process. One of the keys here is that it'd be 'fit-and-forget': once we've solved it, it *stays* solved.

(This'll likely take some significant time, and possibly some serious costs, but importantly it'd be a *once-off* investment: once this kind of problem is solved, we shouldn't need to do it again. We hope…)

– Is it **Actionable** but **Awkward**, always a bit Amorphous and self-Adapting, sometimes almost an 'Anything-goes'? – we know how to do it, but we have to watch for patterns, textures, trends, work with experimentation and emergence. It's always similar, or sort-of-similar, but we can never be certain that it'll be the same. Typically anything that involves working with real people, or anything that connects directly with the real-world, is going to have at least some of this. The key difference from the Complicated is that we can't *solve* it as such, we have to keep '*re-solving*' it, time after time.

(This is *not* going to be a once-off investment: we're going

to have to go through the same loops time after time, yet likely with subtle differences every time. Which means there'll be an ongoing training effort, and ongoing costs. Which is fine, once we know it: what placing something here will do is help us accept that fact.)

– Is it still **Not-known** or **None-of-the-above**? – Not-sure, Not-certain, No-sense-yet, even No-idea or Not-a-clue…? The point is that all of those are fine: that's what sense-making is for, after all. Whenever any kind of unplanned-for-change happens, we're always going to have a bit of this; likewise in innovation, where we'll actually *want* to go into this space of 'the unknown'. The crucial concern is that, rather than hiding these items away in the 'too-hard basket' and vainly hope that they'll disappear on their own, this gives us a known place where we keep track of them, where we *do* acknowledge that they exist, and work on them as best we can.

(In practice, this is the realm of skill and experience - the troubleshooters and trailbreakers and mapmakers and make-it-up-as-we-go-along improvisers who work *with* the uncertainty and create new pathways that others can follow. People who can do this reliably and well are often hard to find, hard to grow, rarely come cheap, and rarely fit well with anyone else's rules: so if this part of our business depends on this, we need to respect that fact, and plan accordingly.)

Note that we *always* end up with some of what's going on still in the Not-known area: it gives us something to go back

to later, if you like.

And we can also apply the same SCAN on each of the areas that we've found, separating *those* out into their own Simple, Complicated, Awkward and still-Not-known. May well be some interesting surprises there, too – sometimes *useful* surprises as well.

What we *do* with this depends on the business, and the context. Scientists would classically aim to follow a path from idea to hypothesis to theory to law, which in effect is None-of-the-above (a new idea) to Awkward (an uncertain hypothesis) to Controllable (a more certain theory) to Simple (a 'scientific law'). Many businesses will want to push to make everything as Simple as possible too, because that's often where the profit is mostly easily made. But others – an advertising-agency, for example – will often *want* to explore out in the idea-space of 'None-of-the-above'; and one person's Simple might well be another's confusingly-Complicated that would collapse in chaos whenever time runs short. In other words, it all depends on what the needs might be; and those, in part, are what we aim to find out here.

So this isn't a fixed 'one framework fits all': it's much more fluid than that, more adaptable to the way that real people work in real business contexts. The aim here is that this gives us a quick, simple, straightforward way to get started, to make sense and map out what's happening, and hence make quick choices that *do* make sense in practice.

There's a lot more to this, of course, and a lot more ways

we can use this, as I'll explore in subsequent posts on this. But for now, that's it – all we need for basic sensemaking in business is to start with the question:

"Let's do a quick SCAN on this?"

Comments or questions, anyone?

(*Note*: In case anyone's wondering where this comes from, its real roots are a sort-of Jungian model I described in the chapter 'Can't we explain this scientifically?' in my book *Inventing Reality*, first published way back in 1986. Perhaps take a look at the original text[6]: it may amuse. Or something. :grin:)

Source (Tetradian weblog)

- *Date*: 2011/11/08
- *URL*: lets-do-a-quick-scan-on-this[7]
- *Comments*: (none)
- *Categories*: Business, Complexity / Structure, Enterprise architecture
- *Tags*: Business, decision-making, disruption, Enterprise architecture, Knowledge, SCAN, sense-making

[6] http://www.tomgraves.org/3science
[7] http://weblog.tetradian.com/lets-do-a-quick-scan-on-this

SCAN – an Ambiguous correction

Yup, I admit: I got it wrong. (Well, the kind of 'wrong' that happens often in early-stage development-work, anyway. :grin:)

In my initial version of the SCAN sensemaking-framework[8], I wasn't happy with the 'A' keyword for the 'not-certain but we do have time to make it sort-of work' domain (upper-right quadrant). I'd started with *Agile*, but that's more about a set of methods that we use in that domain. I'd floundered around with a whole bunch of different keywords, but settled for the awkward '*Actionable but Awkward*' because I couldn't think of anything else.

Courtesy of a blog-post I was reading when a comment[9] from the incomparable Cynthia Kurtz[10] about the previous post[11] came in, this is the all-too-obvious alternative: ***Ambiguous***.

(Duh… I shoulda seen that one *much* earlier… oh well… [beats self with horse-whip, remembers eventually that all

[8] http://weblog.tetradian.com/lets-do-a-quick-scan-on-this/

[9] http://weblog.tetradian.com/comparing-scan-and-cynefin/comment-page-1/#comment-70767

[10] http://www.storycoloredglasses.com

[11] http://weblog.tetradian.com/comparing-scan-and-cynefin/

that does is hurt… :wry-grin: – dumps horse-whip, gets back to writing…])

More to the point, in another comment[12], Cynthia shows that it would work better if I switch the statement round:

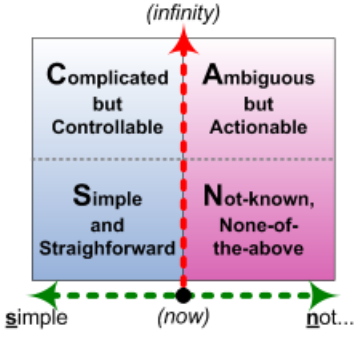
> I love ambiguous. Only I would say it as "***ambiguous but actionable***" matching your other three as what-it-is then what-you-can-do. And it is perfect that the N-spot does away with what-you-can-do because it's not that simple there. I also like how you have "but" on the top and "and" on the bottom, which is meaningful.

I hadn't noticed that distinction between 'but' versus 'and' – entirely accidental, to be honest – but again, she's right. It's only when we have time to argue that we can afford the luxury of 'but'; when time is compressed to almost-nothing, all we have time for is the improviser's[13] 'Yes, and…'.

Anyway, courtesy of those two exactly-to-the-point comments from Cynthia Kurtz, here's the updated core-graphic:

[12] http://weblog.tetradian.com/comparing-scan-and-cynefin/comment-page-1/#comment-70790

[13] http://www.creativeemergence.com/improv.html



SCAN: sensemaking

More posts about how to use this in real-world practice will be coming up Real Soon Now, I promise!

Hope it's useful, anyway?

Source (Tetradian weblog)

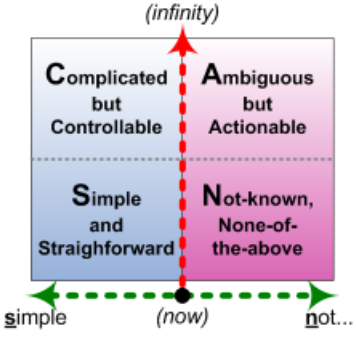
- *Date*: 2011/11/10
- *URL*: scan-an-ambiguous-correction[14]
- *Comments*: 3
- *Categories*: Business, Complexity / Structure, Enterprise architecture
- *Tags*: Business, decision-making, disruption, Enterprise architecture, Knowledge, SCAN, sense-making

[14] http://weblog.tetradian.com/scan-an-ambiguous-correction

Using SCAN: some quick examples

Yeah, right. 'SCAN'. Yet another pretty acronym. What's the point? What's the use? Gimme some real examples, huh?

This one's a follow-up to the previous post "Let's do a quick SCAN on this"[15], in which I introduced the SCAN frame for sensemaking at business-speed:



SCAN: sensemaking

(The above is the updated core-graphic – see 'SCAN – an Ambiguous correction[16]'.)

[15] http://weblog.tetradian.com/lets-do-a-quick-scan-on-this/
[16] http://weblog.tetradian.com/scan-an-ambiguous-correction/

So: some real examples. Let's get started.

Requirements definition

Let's say we're looking at requirements for a new IT-system, and we need to clarify the difference between 'shall' and 'should' in the requirements-specification.

As soon as we say it's 'new', that tells us that there are unknowns. In SCAN terms, *we start from Not-known.* And we start pulling outward from Not-known, into the other spaces.

What is there that's certain, that we know is **Simple** and straightforward? For example, what rules and regulations and standards *must* apply to this? In requirements terms, that's mandatory: that's going to be a 'shall'. We can say that straight away: we don't have to think about it.

What is there that, however **Complicated** it might be to do it, the system still has to deliver against that requirement? That's probably going to be a 'shall' as well, because it's over on the same side of the 'controllable' fence as the Simple. But we might have to spend a bit more time thinking about this.

What is there that's a bit **Ambiguous**? – that we *know* is going to be a requirement of some kind, but it's not particularly clear or definite as yet. That's probably going to be a 'should' – desirable but not mandatory. But again, we might have to spend a bit more time thinking about it.

So we keep digging down into the '**Not-known**', pulling out requirement after requirement, stretching them out into one of the other three categories.

And note that there's still uncertainty about both Complicated and Ambiguous: *we're going to have to spend more time on each requirements-item there*, to determine whether they really are a 'shall' or a 'should'.

Yet to quote a great comment[17] by Cynthia Kurtz on the previous post:

> Give me ten years and I can work my way into making just about anything work (if it doesn't kill me first). Give me ten minutes and it had better be simple.

So if we *don't* have the time to explore further, we treat each item just as they are: Complicated gets squeezed down into the enforced 'shall' of Simple, and Ambiguous gently drops back into the acceptance of a less-enforceable 'should', where it may still remain somewhat 'Not-known' right up until the last moment.

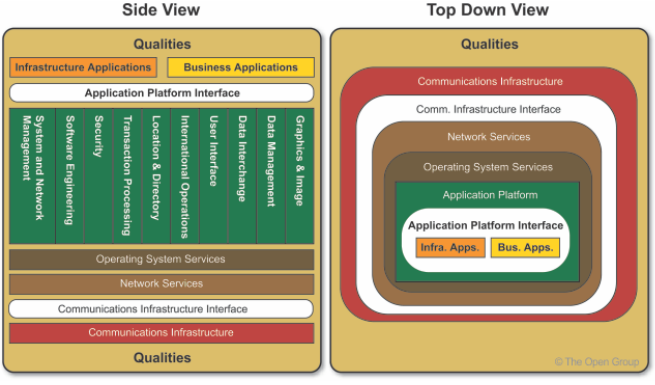
That's how Agile-style requirements-processes work: just before the start of the sprint (or whatever the development-cycle is called), we compress everything down to Simple, or Not-until-next-iteration. At the end of the cycle, we allow

[17] http://weblog.tetradian.com/scan-an-ambiguous-correction/comment-page-1/#comment-70870

ourselves the *time* to re-assess what we've done, and re-explore the requirements-space for items that we could do in the next cycle. We push the time-box back-and-forth: stretch to review the Complicated and the Ambiguous, the 'shall' and the 'should'; pull some of the ambiguities across to 'what we we think we can do'; and then compress it back down again to get the work done in the available time.

EA reference-framework

Reference-frameworks are a commonly-used tool for governance in enterprise-architectures. In effect, they're another type of requirements-specification, but one that straddles across a whole suite of projects, programmes or portfolios, and often aim to apply for several years across the whole of that space. It's a tool to manage risk, opportunity and cost over the longer term.



18

[18] http://pubs.opengroup.org/architecture/togaf9-doc/arch/chap44.html

TOGAF TRM Orientation Views ((c) The Open Group)

(The example above is the raw 'unpopulated' shell for the TOGAF 9 'Technical Reference Model'. For real-world use, specific technologies would be defined for each of the cells in this framework, as the standard reference-framework for the organisation's IT technology-architecture.)

A reference-framework is typically used to specify particular technologies for use in particular contexts, in IT and beyond. As with the requirements, there's a balance between 'shall' and 'should' and the real-world: the reference-framework says what we want to happen, the real-world tells us what we have, and there's then the governance-negotiation that goes on between the two. Hence TOGAF Phase G[19], for example, and the delicate diplomacy needed around architecture-dispensations or waivers and the like. We then also use the reference-framework later on, when reviewing previous dispensations, to see what we *should* have done 'in a perfect world', and explore possibilities to bring whatever-it-is back into line with the intended architecture.

The catch is that most reference-frameworks take a Simple view: everything is portrayed as an 'is-a', a 'shall', a 'must-be'. Which tends to bring on a lot of fights, and denigration about 'the dreaded architecture-police', because the real-world just isn't that simple… And this tension is only going to get worse as the business-space becomes further fragmented with outsourcing, cloud, 'bring-your-

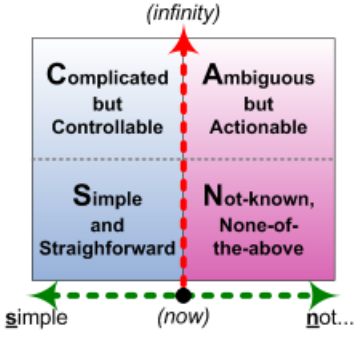
[19] http://pubs.opengroup.org/architecture/togaf9-doc/arch/chap15.html

own-technology' and more. We need a better, more flexible way to define and use reference-frameworks.

To reduce the fights, we can use a SCAN to help us identify where we *must* stand our ground, architecturally speaking, and where it's safe to back off and let people 'do their own thing'.

That time-axis in SCAN is important. If we know we don't have time, we're forced into a straightforward split between the Simple – what we know we can do, what we know we can support – and the 'Not-known' – otherwise described as either "you ain't havin' it" or "you're on your own, bud, we ain't touchin' it". Which, yes, sometimes that's the only choice we have. And in that case, the reference-framework would describe what's supported, and what isn't: which also means that there needs to be the governance to support those constraints in real-world practice – and the clout to back it up without brooking any argument.

Yet most real-world contexts demand a bit more flexibility – *which also means that we need the time to support that flexibility*, and the governance to support that flexibility, too. Given that stretching of time, we can give a somewhat more sophisticated assessment that covers the full SCAN.



SCAN: **sensemaking**

In effect, we extend the simple 'true/false' – it is or it isn't, 'we can' versus 'we can't' – to a more modal[20] logic of possibility and necessity:

- ***is-a*** **(Simple)** - "we're a Microsoft house" (that's what we do, so don't expect it to be cheap or certain or even doable with anything else)
- ***is-sometimes-a*** **(Complicated)** – "we prefer Windows, and we do that best, but we can also support Microsoft packages on Mac, UNIX and Red-Hat Linux" (it'll cost extra time and money, but we know it'll work)
- ***is-believed-to-work*** **(Ambiguous)** – "there are [these listed] equivalent packages on Windows and on [these listed] other operating-systems: we've been told they work, but we haven't yet tested them

[20] http://en.wikipedia.org/wiki/Modal_logic

ourselves" (and if you want us to test them, it'll cost both time and money, and may not work anyway)
- ***none-of-the-above*** (**Not-known**)- "sure there's plenty else out there, but we don't know much of anything about it" (we have no idea what it'll cost even to find out more about it, and no idea if it'll work at all with what we have)

A practical catch here is that most current EA toolsets don't support this kind of modal-logic in reference-frameworks (or anything else, for that matter). Usually the nearest we have for this are composition- or aggregation-relationships: they're sort-of-usable as a workaround for this, but they're not quite the same, and can be misleading if we're not careful.

Anyway, much the same happens with reference-frameworks as in Agile-development: over time, there's a steady migration of some (but not all) from Complicated to Simple, and some (but not all) Ambiguous to Complicated. Yet some will *always* remain Complicated; some will *always* remain Ambiguous; and even more, there will *always* be some that's Not-known. A repeated, recursive SCAN helps to clarify what will move between those 'domains', and when.

Source (Tetradian weblog)

- *Date*: 2011/11/11

- *URL*: using-scan-quick-examples[21]
- *Comments*: (none)
- *Categories*: Business, Complexity / Structure, Enterprise architecture
- *Tags*: Business, decision-making, disruption, Enterprise architecture, Knowledge, SCAN, sense-making

[21] http://weblog.tetradian.com/using-scan-quick-examples

Domains and dimensions in SCAN

What are the sensemaking-domains in SCAN? What are the boundaries between those domains?

A great challenge in an earlier comment[22] from Roger Sessions[23], where he asked me for the mathematical basis for those domains and boundaries. I think he was a bit shocked when I said[24] there wasn't one – but in fact there is such a basis, sort-of, and it's worth summarising here, out on the surface rather than buried away in the comments.

(Whilst working on this I've realised that in some ways this is a repeat of the section 'The structure of SCAN' in the post 'On SCAN, PDCA, OODA and the acronym-soup[25]'. But it's probably worth having the extra detail, anyway.)

The dimensions of SCAN

There are three distinct dimensions to SCAN:

[22] http://weblog.tetradian.com/ensuring-that-the-simple-stays-simple/comment-page-1/#comment-71367

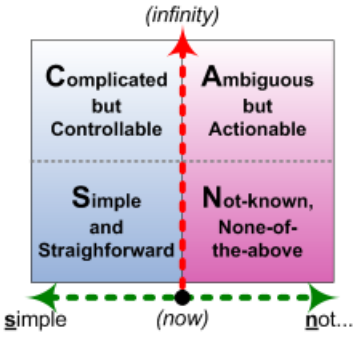
[23] http://twitter.com/RSessions

[24] http://weblog.tetradian.com/ensuring-that-the-simple-stays-simple/comment-page-1/#comment-71384

[25] http://weblog.tetradian.com/on-scan-pdca-ooda-acronym-soup/

- *modality*[26] – the extent of perceived 'controllability' versus 'possibility and necessity'
- *available-time* – the amount of time remaining before an action-decision must be made
- *repeatability* – ability to reliably recreate the same perceived results

The SCAN frame is usually shown with four apparent domains, derived from the first two dimensions above:



SCAN: sensemaking

In part, though, this format is mainly for people who are more comfortable with a simple two-axis matrix, or who need to translate across from other domain-oriented frameworks such as Cynefin or the Jungian-based 'swamp analogy[27]'. This layout can be somewhat misleading in that the boundaries between apparent domains are not

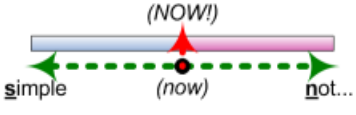
[26] http://en.wikipedia.org/wiki/Modal_logic
[27] http://www.tomgraves.org/3science

straightforward, and that third dimension of repeatability *does* also need to be taken into account. We'll explore how this works in practice in the rest of this article.

Dimension of modality

The 'horizontal' dimension for SCAN is a scale of *modality* of the logic used for sensemaking and decision-making. Modality in this sense is the scope of possibility and necessity; a scale of modality ranges from a simple 'yes/no' or 'true/false' choice, to an infinity of possibilities. We make a choice from the palette of possibilities on offer in the context, in accordance with what we perceive as the necessity in the context.

In principle, for SCAN, we should draw this as a horizontal graded spectrum of 0..*n* possibilities for choice, from left (choice of 0..1) to right (choice of 0..infinity). In practice, though, we can put an explicit boundary at the 0..1 point, because the way the choices are usually addressed will change radically on either side of this point. In SCAN, we describe this distinction as a Simple choice, or a Not-simple choice:



SCAN: time-compression

Anything that relies on absolute repeatability regardless of agent, on identical circumstances, or on any true/false logic, must by definition be constrained to the Simple side of the scale. This includes almost all machines, most IT, and any rule-bound human context.

The key point is that on the Simple side, there's only one choice: do it, or don't do it. Very straightforward. (Whether it actually *works*, in terms of creating the required results, is another story, of course…) Once the options start to multiply, the choices become more Complicated, but as long as the choice-mechanism is still some form of true/false, it still remains 'controllable' – we just need more *time* to sift through the options and factors and make the 'right choice'.

But once the options become contextual, or dependent on the skill and capabilities of the agent, or for any reason *cannot* be absolutely repeatable, that pushes us over the boundary to where a Simple true/false logic is unlikely to work. In other words, it's Not-simple. And we start to need other ways to work with it – ways that are usually *not* available from machines or IT, or from inexperienced human trainees. The further over into the Not-simple that it gets, the higher level of skill it will require to get to the equivalent of 'repeatable' results – the same perceived outcomes reached via different routes.

One important complication arises from different experiences of 'simple' versus 'complex'[28]. The definition for Sim-

[28] http://weblog.tetradian.com/human-view-of-simple-complicated-complex/

ple here is the use of true/false choice-logic, of true/false rules and so on. However, many people experience that as anything *but* 'simple' – especially where rigid rules are applied in contexts that have high natural variability and hence *need* greater modality. In those contexts, more fluid patterns and guidelines are often experienced as 'simple', because it's easier to use them to achieve the same perceived outcomes.

In those types of circumstances it may be better to change the horizontal scale from a 'mathematical' one of modality', to a more subjective scale of what is *experienced* as 'Simple' versus 'Not-simple'. We do, however, need to be really clear and explicit as to which type of scale we're using!

Dimension of available-time

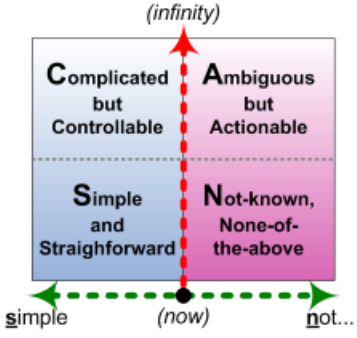
The 'vertical' dimension for SCAN is a straightforward scale of *time-available-for-decision*. We can use a linear or logarithmic scale scale for this: the choice probably doesn't matter, although since the time-available can potentially stretch to infinity, a logarithmic scale might make more sense in practice.

The key point here is that as we gain more time before a decision must be made, we also gain the ability to assess a broader range of options. When the time is tightly focussed, we *must* focus on 'right here, right now', the *specific* point of action, using only what is available at the time. When the time is less tightly focussed, we get to have more choice

about how to tackle decisions, about what can be used to enact those decisions, and so on.

It's a continuous spectrum, of course, so any 'boundary' we put along along that vertical axis is going to be somewhat arbitrary. One easy way to partition the timescale is the classic three-way split between Strategy (far-future), Tactics (near-future) and Operations (NOW!). Another – which would obviously be a better fit with the notion of a two-axis matrix – is 'Time-to-think' versus 'No-time-to-think'.

If we use the latter, and combine it with the Simple/Not-simple split on the 'horizontal-axis', that gives us a conventional 'four-domain' layout for SCAN. So let's use that for now - *always remembering that the position of the vertical-axis boundary between 'domains' is an arbitrary choice.*



SCAN: sensemaking

The horizontal-axis here still has that boundary between 0..1 true/false decision-logic to the left, and a true 0..*n* modal-logic to the right.

So on the left, Time-to-think ('Complicated') gives us the rules and categories that we would use in a true/false logic when there's No-time-to-think ('Simple'). In other words, it allows its 'world' to be more Complicated, but it still expects it to be 'controllable', for there to be an identifiable, repeatable 'right answer' to every context-related question. In terms of systems-theory, this is the kind of space where we would expect to find 'hard-systems' models in use.

And over on the right, Time-to-think ('Ambiguous') gives us the patterns and 'seeds' – and also the support to work *with* the uncertainty – for when we work with inherent-uncertainty when there's No-time-to-think ('None-of-the-above'). It allows its 'world' to be Ambiguous, uncertain, yet also still 'actionable', understandable, describable in some sense. In terms of systems-theory, this is the kind of space where we would expect to find 'soft-systems' models in use, or concepts of 'complex-adaptive-systems' or 'emergent-systems' and the like.

When it gets down to No-time-to-think in that modal space, though, there often *isn't* any way to understand it, or even describe it: it's too context-specific, too unique to the context, the person or both, often dependent on *personal* skills and experience that only 'make sense' to that individual person. It's often not sharable as such within anyone else, and there's no obvious way to make it fit into any predefined category, or even into any identifiable pattern. Hence the label 'Not-known', or 'None-of-the-above': it simply *is*. And yet that also *is* the domain in which perhaps

most human work is actually done – and hence should *not* be ignored…

Dimension of repeatability (variety)

Perhaps the real point about the None-of-the-above domain is that it's probably the closest we have to 'the real-world': *everything else is an abstraction.*

What we *actually* have in sensemaking – and, hence, its derived decision-making – is a myth of abstraction: the belief that abstractions are somehow 'real'. Something 'makes sense' because we *choose* that it should 'make sense' in that way: just how much it actually *is* 'real' is often an open question…

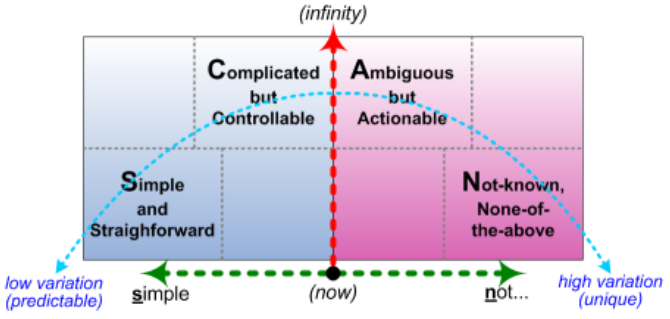
In practice, there's a spectrum of certainty of abstraction, from idea to hypothesis to theory to 'law' – lowest-certainty to highest-certainty. The closer we get to 'law' (in the scientific sense, at least), the more predictable the context should be – assuming that the 'law' is an accurate abstraction, of course. The more predictable it is, the more we can rely on its repeatability – where the same actions deliver the same results. Conversely, the less predictable it is, the less we can rely on the same actions delivering those same results. Which gives us a spectrum of repeatability.

That spectrum of repeatability does sort-of line up with abstraction, but even more so with the *variety* – in the

cybernetics sense – in the actual context. When there's a mismatch between the type and range of variety that the abstraction can cope with, versus the actual variety in the context – the 'system' – then we're likely to get systemic failure.

This is crucial in enterprise-architecture and the like, because most IT and the like, and most systems that depend on 'command and control', are almost by definition constrained to the amount of variety that can be covered by their own true/false logic. The whole point of conventional command-and-control is that it *doesn't* permit any variety beyond its own scope. And when the real-world *does* happen to contain greater variety – which, to be blunt, it often does – then, again, the system will fail. (Though likely that it'd be the real-world, rather than the inadequate abstraction, that would be blamed for the failure…)

When we put all this together in SCAN, that spectrum of repeatability – or, inversely, of variety – ends up somewhat like this:



SCAN: repeatability

When there is low variety in the overall system, it's relatively Simple to set up straightforward rules, and enact those rules in real-world practice with high to very high probability of repeatable results – including the same results from different agents that use the same rules.

As the variety increases, we need *time* to be able to identify the various factors and feedback-loops. The system becomes more Complicated, but up to a certain point will still be able to deliver repeatable results with different agents that follow the same more-complicated systemic rules.

As variety continues to increase, there is a crucial crossover point where doing the same thing can no longer be guaranteed to deliver the same results, sometimes even with the same agent. The crossover into Ambiguous *automatically* occurs whenever an unaccounted-for factor enters into the supposedly-predictable picture. We can sometimes work out new rules for that new factor, but in some cases there will *always* be uncertainty, ambiguity –

and trying to force the system to fit the lower-variety assumptions of the Complicated 'domain' will usually cause systemic failures such as 'wicked-problems[29]' and the like. 'Soft-systems' and 'emergent-systems' methods will help to address the issues here, often developing patterns and guidelines for use at real-time, but as the Complicated 'domain', all of these techniques take *time* – which may not be availability.

As time-available becomes compressed towards real-time, or variety increases still further towards non-repeatable uniqueness, we end up being forced into a 'domain' that's probably best summarised as Not-known or None-of-the-above. Increasingly, the equivalents of 'repeatable' results on *not* repeating the same actions – and it takes increasing levels of skill to guide the context towards desired 're-peatable' results. Conversely, repeating the *same* actions can deliver different results – which again, with skill, may return highly-desirable uniqueness.

Overall, though, note the transitions here: both Simple and Not-known, at the opposite ends of the scale, can work well at or near real-time, whereas Complicated and Ambiguous, in the mid-range, require *time* to execute. If 'control' is required at real-time, it *must* be Simple: there is no other option. Any other choice either requires more time, or an acceptance that 'control' will not work in the context. Again, this point has huge implications for enterprise-architectures.

[29] http://en.wikipedia.org/wiki/Wicked_problem

A possibly-simpler summary

Some quick follow-on points from all of the above:

- In the real world, 'control' is a myth – we can simulate some of it, but it often constrains ability to cope with real-world variety such that it's likely to break down.
- When it *does* break down at the Operations level, at or near real-time, we're automatically forced into a None-of-the-above context, which requires skill and experience to bring the context back to a simulation of 'control'.
- If the skill and experience are not available, or are excluded, the overall system *is* going to fail – as happens often in misguided attempts at IT-centric 'business process reengineering'.
- Analysis and experimentation *take time to execute* – they're not viable when time-available is compressed down to the real-time level.
- In most real-world systems, time-available will vary: there are periods of intense time-pressure, where only the Simple and the None-of-the-above will work; and there are periods when the time-pressure eases off, which can be used for review and reassessment via a move 'into' the Complicated and/or Ambiguous 'domains', to prepare for the next high-intensity part of the work-cycle.

I'll stop there for now, but there'll be more on this in the upcoming final part of the 'on sensemaking[30] in enterprise[31]-architecture series[32]', and in other future posts.

Source (Tetradian weblog)

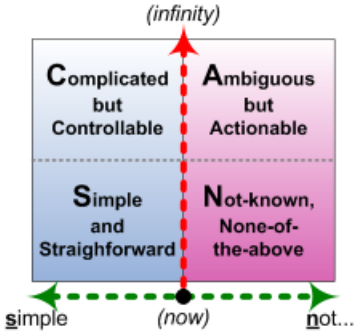
- *Date*: 2011/11/18
- *URL*: domains-dimensions-in-scan[33]
- *Comments*: (none)
- *Categories*: Complexity / Structure, Enterprise architecture, Knowledge
- *Tags*: ambiguous, chaotic, complex, complicated, Enterprise architecture, not-known, SCAN, sense-making, simple

[30] http://weblog.tetradian.com/on-sensemaking-in-ea-1/
[31] http://weblog.tetradian.com/on-sensemaking-in-ea-2/
[32] http://weblog.tetradian.com/on-sensemaking-in-ea-3/
[33] http://weblog.tetradian.com/domains-dimensions-in-scan

Real-time sensemaking with SCAN

What do we do when we don't know what to do? – and how do we ensure that whatever we do is the right thing to do? How do we make sense *fast*, at *business*-speed?

I've been tussling with this one for quite a while, most recently culminating with a simple sensemaking framework called SCAN:



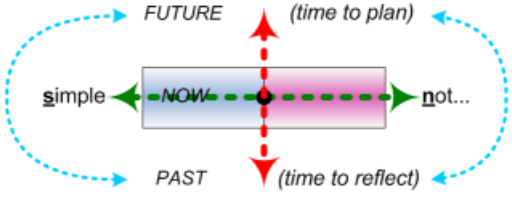
SCAN: **sensemaking)**

The horizontal green-line axis here represents the decision-type, from a simple true/false choice to a not-so-simple modal choice of possibility and necessity; the vertical red-line axis is the amount of time available before *must* make

a choice and take action.

(For more on SCAN and its technical background, see the posts '"Let's do a quick SCAN on this"[34]' and 'Domains and dimensions in SCAN[35]'.)

In a sense, though, that red line of 'available-time' goes *both* sides of the 'now', extending outward both into future plans and past record:



SCAN: past and future

Time and distance and even social-distance all compress down towards the point of decision, the moment of action, the *now*. That 'now'-moment is the only one that matters: prior to that point, every 'decision' may be nothing more than a vague statement of intent, which may not actually happen in practice – as I know only too well…

At each moment of 'right *here*, right *now*', it's always *our* responsibility - our 'response-ability', our individual and personal *ability to respond*.

The 'now' is the still-point at the centre of action. Yet it's an *active* stillness, and there *are* still choices there in that

[34] http://weblog.tetradian.com/lets-do-a-quick-scan-on-this/
[35] http://weblog.tetradian.com/domains-dimensions-in-scan/

moment. So what we aim for in this kind of real-time sensemaking is to create just enough space to enhance that 'ability to respond' - enough space to enable appropriate choice for appropriate action.

If we *don't* create that space for choice, the only 'choices' we have come from habit - which may not be appropriate to to the context - or the various 'hard-wired' reflex-responses, such as 'fight', 'flight' or 'freeze'.

(The other hard-wired natural-reflex is 'fornicate', but we'd, uh, best leave that out of the conversation for now…? :wry-grin:)

Whilst it's easy enough to describe what goes on either side of that choice-point, it's surprisingly hard to describe the choice-point *itself* without sounding somewhat mystical. Rather like the cosmological moment of the Big Bang, it's both technically and literally a moment of chaos, within which the 'normal rules' break down, and which contains within itself every possibility and every other point.

This is the literal meaning of Pan, by the way – 'the every-thing'. If we can't cope with this infinity of (im)possibility, we're like to fall into panic. And that's what leads to those three reflex-responses – each of which rejects the uncertainty in their own distinct way:

- *fight*: grab at a single possibility and 'take control' (whether or not that single chosen option is appropriate to the needs of the context)

- *flight*: 'run away' from the choice (such as to a 'considered-sensemaking'[36] framework which cannot work at real-time, and hence leads to some variant of 'analysis-paralysis')
- *freeze*: do nothing and hope that the need for choice will go away (which only works if there's no actual need for choice or action)

What we need to do instead is stay *within* the 'chaos' for as long as we can, to allow the *appropriate* choice to emerge from *and with* the context itself. Describing this as 'act / sense / respond' is way too simplistic: it's more like a real-time dance of choice and action, a transitory yet immensely powerful condition of *flow* that is often experienced as a kind of 'no-time' that is seemingly *beyond* time.

(People who *can* hold that space are often described - or derided - as 'eccentric', 'the crazy ones[37]'. Yet 'eccentric' is literally away from the centre - and that's the place where change can happen, because that distance also provides leverage for change. Being seen as 'eccentric' can be difficult at times, but it's certainly important...)

What I've been working on over the past few days or so is trying to a detailed mapping of what actually happens in the real-time space, using this specific question of real-time sensemaking as the 'target problem' to keep in focus.

[36] http://weblog.tetradian.com/comparing-scan-and-cynefin/
[37] http://www.youtube.com/watch?v=4oAB83Z1ydE

(As usual, I've gone back to first-principles to do this, so in effect I've been watching myself at work whilst doing this work. What I've been seeing may not be the way that others do this, of course, but it actually does match up quite well with what's in the rather eclectic mix literature that I happen to know, from Lao Tse's *Tao Te Ching* to Csíkszentmihalyi's *Flow*, and from *Zen and the Art of Motorcycle Maintenance*, to *The Art of Scientific Investigation*. So no claims to be 'academic' as such, but that isn't the point: I'm a practical toolmaker, not a 'pure' theorist, after all.)

For this, I've used the 'time-compressed' version of SCAN, in which everything is squeezed down to a real-time choice of tactics, between 'Simple' and 'Not-simple':

SCAN: time-compression

The crucial boundary on this dimension is what I've called 'the Inverse Einstein test':

- if we do the same thing and get the *same* results, it's on the Simple side of the story – so we would attempt to use Simple-side tactics
- if we do the same thing and get *different* results, it's on the Not-simple side of the story – so we would need to use tactics from the Not-simple side

In real-time sensemaking we actually swing back and forth between these 'domains', using a variety of real-time checks to tell use which side we need to be on at any one moment. They're different disciplines: but by swinging back-and-forth in a *conscious* and *deliberate* way, we maintain an *overall* discipline at all times.

(First-hand example: doing a formal back-massage. At first, I'll follow the rules, following the standard sequence of moves and work-patterns, using that pattern itself as a focus. At some point it switches into that '*flow*-state', and I'll find myself doing something subtly different, applying pressure in a different way, following kind of 'inner instructions' that seem to come through my hands themselves. Then, just as suddenly, the '*flow*-state' fades, leaving me feeling a bit lost, like I don't where I am, I don't know what to do. That's when the key-phrase 'Don't Panic!' comes in, and reminds me to go back to 'the rules' - back to the Simple-side - and follow that pattern until the '*flow*-state' returns. Which it may not, of course - but at least I'll have done *something* useful simply by following 'the rules'.)

If I use the tags '[S]' for Simple-side, and '[N]' for the Not-simple side, these are some of the points I've noticed during this week about that real-time back-and-forth:

– [S] is about following the instructions, following 'the rules'; [N] is about allowing 'the answers' to arise in whatever way *they* seem to choose.

– [N] is what we do while that 'inner knowing' lasts; [S] is

what we do when the knowing fades.

– *Both* sides need calm, and need discipline – including the discipline about how and when to switch back and forth between them.

– [S] has notions of 'truth', of 'control', of certainty, "I know what to do"; [N] calls for a kind of faith, a lot of trust, perhaps Susan Jeffers[38]' "feel the fear and do it anyway" – and often a difficult balance between "*do* something, don't just stand there!" and "*don't* 'do something, just stand there…".

– In a rework of the old slogan "think global, act local", [S] seems to focus on '*act* local', whilst [N] seems to allow the broader space of '*aware* global' – no time to stop and think at real-time, yet use that deep-space of 'the everything' to help maintain the big-picture awareness.

– [S] seems to work best with rules or checklists – which is hardly surprising since in essence it thrives on real-time certainties. Some of the rules and checklists I use a lot in real-time sensemaking for enterprise-architecture include:

- allow the uncertainty to *be* uncertain (i.e. keep gently returning to the Not-simple side)
- don't try to control – allow 'the answers' to arise in their own way
- use the 'checklist for checklists[39]' to create checklists

[38] http://www.susanjeffers.com/home/index.cfm
[39] http://www.projectcheck.org/checklist-for-checklists.html

on-the-fly with whatever ideas I've gleaned from the Not-simple side
- use quick enquiry-techniques such as 'Five Whys[40]' to push into the Not-simple side for new ideas and information
- use Five-Whys to move *up* the scale of abstraction towards core-purpose
- use Five-Hows to move *down* the scale of abstraction towards real-world implementation
- use the R5 set of system-thinking principles – rotation, reciprocation, resonance, recursion, reflexion – to look for factors and patterns in the context
- use the REAL checklist – reliable, efficient, appropriate, elegant – to test for effectiveness themes (sometimes extended to 'LEARN' with the addition of 'integrated')
- use the tetradian set – physical, virtual/conceptual, relation/emotional, aspirational/spiritual – to review asset-dimensions in a context
- use the Five Elements set – Purpose, People, Preparation, Process, Performance – to assess balance across strategy, tactics and operations (which also aligns with the Tuckman project-lifecycle sequence 'forming, storming, norming, performing, adjourning')

– Almost by definition, [N] doesn't seem to have any clear patterns: the only 'patterns' I see myself doing all too often

[40] http://en.wikipedia.org/wiki/5_Whys

on that side are ones about how to *avoid* making sense, such as running away to check emails or make yet another cup of tea…

Anyway, that's it for the moment. It's just a work in progress, as usual, but make of it what you will.

Source (Tetradian weblog)

- *Date*: 2011/11/28
- *URL*: real-time-sensemaking-with-scan[41]
- *Comments*: (none)
- *Categories*: Complexity / Structure, Enterprise architecture, Knowledge
- *Tags*: ambiguous, chaotic, complex, complicated, Enterprise architecture, not-known, SCAN, sense-making, simple

[41] http://weblog.tetradian.com/real-time-sensemaking-with-scan

Belief and faith at the point of action

What is it that drives decisions at the exact moment of choice and action? – even in the most mundane, everyday action? If the choice-point itself is a true moment of chaos – a point where literally anything is possible – then what is it that guides us through each of those infinitesimal yet ubiquitous moments?

A lot of this is still tentative, very much 'a work in progress'. Yet what I've found myself returning to again and again over the past few days, whilst working on the design and workflows for the SCAN app[42], is a pairing of two words: *belief*, and *faith*.

[Don't worry, I'm not going to go all religious on you. (Well, probably not, anyway!) This is still the same enterprise-architecture exploration about the context of SCAN, about sensemaking and decision-making at real-time, particularly in what some would term the 'Chaotic domain'.

Minor warning, though: this is written in English, and from the perspective of an Anglo culture. I think (believe? guess?) that what follows is close to generic across all

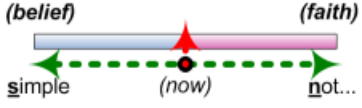
[42] http://weblog.tetradian.com/four-ea-app-ideas-anyone-interested/

human cultures, but note that you may well need to do some translation here, both linguistic and cultural.]

Where SCAN's 'Simple' and 'Not-simple' are about about how to describe sensemaking, belief and faith seem more about decision-making – the actual moment of *choice* that immediately precedes each moment of action. In other words, decision-making in real-time. And because sense-making, decision-making and action are all intertwined with each other within real-world practice, belief and faith also map onto the SCAN frame in much the same way as for real-time sensemaking.

[There's also a mapping to the full SCAN, that extends this outward to the scope where there is more time available for review, but I'll describe that in another post.]

In short, *belief* maps to the known, the certain, the Simple; whilst *faith* maps to the unknown, the uncertain, the Not-simple:



SCAN: belief and faith

As in sensemaking, the crucial distinction occurs where the modality[43] of the decision-choice changes from a Simple deontic[44] true/false to a Not-simple true alethic[45] logic of

[43] http://en.wikipedia.org/wiki/Modal_logic
[44] http://en.wikipedia.org/wiki/Deontic
[45] http://en.wikipedia.org/wiki/Alethic_modality

'possibility and necessity':

– over on the left-side, *belief* provides a straightforward black-or-white choice: true or false, right versus wrong, culturally 'proper' versus 'politically incorrect';

– over on the right, choices are more blurry, more uncertain, more 'shades of grey' – or more colourful, perhaps – and the only guide we have is *faith* or trust that what we do is right. (Right in its own way, but still 'right' in some sense.)

Both of these are actually about the individual, about 'I'. Which it should be, of course, because that's all we have at the exact point of action: our own choice, and our own 'response-ability'.

Belief is fast, and importantly doesn't demand any personal skill as such: the whole point is that they're deemed to be 'true' for all who enact them, regardless of who or what enacts them. (A belief may be *believed* to apply only to self – such as 'nothing goes right for me' – but is still held as an 'absolute truth' in that sense.) This has both advantages and disadvantages, mainly relating to how well the belief *does* match up to actual reality. Advantages include:

- simple beliefs are useful when the person enacting them has only a limited level of skill and 'response-ability' – "just follow the instructions, kid…"
- even for those with skill, simple beliefs are useful as a structured fallback for whenever the faith falters in

the context and in one's own ability – "when all else fails, follow the instructions"
- advising acceptance that some contexts *are* constrained by 'laws' of some kind – particularly the physical-world constraints implied by 'scientific law' and the like
- beliefs are also useful as a disciplined means to temper excess enthusiasm – "trust to Allah, but tie the camel first"

A classic example of a structured belief of that last type is the checklist[46] – mapping out essential safety-checks and other 'known truths' prior to or during any activity that is inherently uncertain.

The disadvantages of 'prepackaged' belief-structures are more complex, and often rather more subtle:

- the usefulness of beliefs ultimately depends on the myth of 'control', the myth of predictability and certainty – none of which may be valid in a real-world context
- beliefs themselves can and do act as perceptual filters, potentially rendering invisible essential contrary information from the context
- as guides for choice and action, beliefs can apply inappropriate constraints to action in any given context – following 'the letter of the law' rather than 'the spirit of the law'

[46] http://gawande.com/the-checklist-manifesto

- in much the same way, beliefs can be used to *evade* difficult or challenging choices – for example, 'morals' as 'the lazy-person's ethics'

Faith is often the only choice-mechanism available whenever the context is inherently uncertain. It also correlates closely to *skill* – so much so that, in essence, 'skill' is a proxy for the real-world reliability of faith in one's own ability to work with the inherent uncertainties of a given type of real-world context. In other words, skill is what determines whether we really *can* do what we believe or hope we can do in that kind of context.

Sometimes, though, it isn't about skill: it's just about faith, or trust. Every change of belief requires 'a leap of faith'; innovation or experimentation always requires us to accept that *we don't know* what the outcome will be. (That's very different from belief, where we *do* expect the outcome to be what we expect.) A modal-logic of possibility and necessity is the only place where 'the impossible' first becomes possible – and thence, through skill, becomes probable, then predictable, and eventually something resembling certain, a kind of 'law' in its own right. It may end up as a checklist or some other pre-packaged set of beliefs – but it always *starts* with faith, in the midst of a moment of inherent uncertainty.

As with belief, there are disadvantages to faith too: not least what we might describe as 'misplaced faith', where lack of skill – or plain old lack of awareness – leads to inappropriate outcomes. Whether we like them or not,

sometimes the constraints of belief *do* apply – such as in most (though *not* all) assertions of 'scientific law' and the like.

So in practice we need to be able to bounce back and forth along SCAN's 'horizontal' axis of modality. Sometimes we need to hold to a Simple true/false belief; sometimes we need to let go into the Not-simple world of faith and trust. And of course, recursively, there are no set rules about which one should always apply at any given moment – which means that this too is a skill in itself. It's Complicated, perhaps, or Complex[47]… yet in real-time action we don't even have time for either of those. All we have is *this* decision, right *here*, right_ now_ – no time for anything else. Belief that we *know* what to do; or faith that the results we need will arise from within the chaos itself.

All of which means that, as enterprise-architects, we need to understand how belief and faith work within our organisation and enterprise, and provide structures to support them in real-world practice.

Enterprise-architecture implications

It's essential to draw a distinction here between the *individual* and the *organisation*. Belief and faith are expressed in practice directly by the individual, or indirectly by proxy,

[47] http://weblog.tetradian.com/sccc-simple-complicated-complex-chaotic/

such as via the design or operation of a semi-autonomous machine or IT-system. Yet in an organisational context, it's the *collective* belief and faith that we want expressed in action – expressed *by* the individuals *on behalf of* the organisation, the collective.

In effect, that's the key role of organisational culture – and despite the wishes of executives and others, it's not as simple as it looks… For enterprise-architects, it also means that we often have to address aspects of organisation-architecture that are more usually the territory of HR and change-management and the like – which means that we have to tread carefully at times, and engage in some potentially-challenging negotiations. But the payoff is an enterprise-architecture that really *works* – for everyone.

The *organisation's* **beliefs-in-action** are expressed in definitive statements such as work-instructions, reporting-relationships and business-rules. One of the architectural concerns here is to provide support such that these business-rules and the like are actually implemented in practice, in real-time decision-making.

To make this work, we in effect need each individual to take up those shared-beliefs as if they are their own *personal* beliefs. This is especially important wherever these rules must normally be followed 'to the letter' – such as in regulatory compliance.

It's crucial to understand, though, that rules cannot be imposed onto individuals from outside, whether by fiat or threat of force. Although as an organisation we can give

ourselves the *illusion* that this has been done, it rarely works in practice: instead, there will usually be a myriad of small 'failures', ranging from unconscious errors to covert rebellion, which effectively sabotage the intended functional impact of the rules. (The former will tend to occur more often in collective-oriented cultures, the latter especially so in individual-oriented cultures.)

What *does* work is to engage people in the rules – the 'why' as much as the 'how' and 'what'. To use the terms from Hagel, Brown and Davison's *The Power of Pull*[48], we create that engagement by shifting from 'push' to 'pull'. In an enterprise-architecture, we do this by treating organisational-beliefs in much the same way as for organisational-*values*. The Enterprise Canvas[49] model describes a generic structure[50] for this purpose:

- *create awareness* of the rules-structure, its purpose and rationale, and the context for its use
- *build capability* to apply the rules-structure in real-time practice
- *apply the rules-structure* in run-time decisions
- *verify and validate* the usage of the rules-structure
- *derive lessons-learned* from the (attempted) usage of the rules-structure

[48] http://www.amazon.com/Power-Pull-Smartly-Things-Motion/dp/0465019358

[49] http://weblog.tetradian.com/tag/enterprise-canvas/

[50] http://weblog.tetradian.com/ecanvas-as-service-viability-checklist/

Working with HR, change-management, process-management and others, we create what is in effect a PDCA[51]-type learning-loop, to develop, apply and revise the business-rules and other belief-structures for the organisation.

The **faith-in-action** side of that decision-making modality-spectrum deals with anything that *isn't* covered appropriately by business-rules and the like – which is a *large* part of most real-world organisational contexts. For enterprise-architecture, the two key focus-areas are *skills-development*, to enhance individual 'response-ability'; and *vision, values and principles*, to enhance consistency in decision-making across the collective.

The skills-issue is one that is almost completely missing from most current-enterprise-architectures, especially those of an IT-centric bent. That's rapidly becoming a lethally-dangerous oversight – see the Sidewise post 'Where have all the good skills gone?[52]' – and one that we *need* to address, working in conjunction with HR, organisational-development units and suchlike. EA will come into the picture by mapping out skills-requirements and competency-levels needed within enterprise-capabilities[53]; the actual skills-development would usually be out of scope for EA, of course, though overall much of it would follow that generic structure for values as above.

The values-issue is one I've been pushing for a very long

[51] http://en.wikipedia.org/wiki/PDCA
[52] http://sidewise.biz/2009/07/skills/
[53] http://weblog.tetradian.com/on-function-capability-service/

time as the true core of the enterprise-architecture[54]: for example, it forms the topmost layer of abstraction in Enterprise Canvas[55], and thence acts as the anchor for the generic structure described above for values-management services. The reason why it's important is that if the organisation isn't clear about its values, then what will be used instead – as the drivers for 'faith'-type decision-making – will be whatever values happen to be around for that individual. Which could be anything at all. Including not just a destructive 'me-first', but a *really* destructive 'me-only'. In other words, *not* a good idea… clarity on values *matters*.

A lot more that could be said on all of that, but I'd probably best leave that for the moment. The only point that *does* need to be added here is the importance of *story* – the enterprise *as* story, the enterprise *is* the story[56] – as the 'glue' that holds all of this together.

Overall, the real point here is this: that at the point of action – and despite whatever we might plan beforehand – decisions seem to be taken primarily on the basis of belief, or of faith or trust. Which means that, architecturally, we *need* to design for that fact. Not a trivial point, then.

Source (Tetradian weblog)

[54] http://www.slideshare.net/tetradian/vision-role-mission-goal-a-framework-for-business-motivation

[55] http://weblog.tetradian.com/ecanvas-as-service-viability-checklist/

[56] http://weblog.tetradian.com/the-enterprise-is-the-story/

- *Date*: 2011/12/03
- *URL*: belief-and-faith-at-point-of-action[57]
- *Comments*: 3
- *Categories*: Business, Complexity / Structure, Enterprise architecture, Knowledge, Society
- *Tags*: belief, decision-making, effectiveness, enterprise, Enterprise architecture, faith, paradigm, SCAN, sense-making, trust, values, worldview

[57] http://weblog.tetradian.com/belief-and-faith-at-point-of-action

Decision-making – belief, fact, theory and practice

In what ways do ideology and experience inform decision-making in real-time practice? How do we bridge between the intentions we make before and after action, with the decisions we make at the point of action itself? And what implications does this have for our enterprise-architectures?

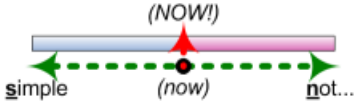
This extends the previous post on real-time decision-making, 'Belief and faith at the point of action[58]', to crosslink with the earlier ideas on SCAN[59] and sensemaking, and especially about where there *is* more time available to review and reflect on action.

[A gentle warning and polite request: much of this is still 'work in progress', so do beware the rough edges and knobbly bits, and use it with some caution; and whilst I do need critique on this, please don't be *too* quick to kick down the scaffolding that's holding it all together. Fair enough?]

The previous post was about how options for sensemaking become more constrained as we approach real-time. Right

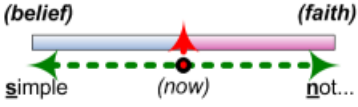
[58] http://weblog.tetradian.com/belief-and-faith-at-point-of-action/
[59] http://weblog.tetradian.com/lets-do-a-quick-scan-on-this/

at the point of action, the options reduce to either a Simple interpretation in terms of of true/false categories, versus a Not-simple interpretation based on a modal-logic of possibility and necessity, which is much harder to explain or even to describe to anyone else. In SCAN we'd depict that compression as follows:



SCAN: time-compression

In much the same way, decision-making becomes compressed down to Simple belief versus Not-simple faith – neither of which are *actually* explainable, and both of which, at the root, are primarily emotional rather than 'rational':



SCAN: belief and faith

In both sensemaking and decision-making, the crucial distinction – indicated in SCAN by where the red-line time-axis crosses the green-line axis of decision-modality – is what I've termed the 'Inverse Einstein test'. Einstein is said to have asserted that "insanity is doing the same thing and expecting different results": but whilst that's true in a simple rule-based world, it's *not* true – or not necessarily

true, anyway – in a more complex world where many things are context-specific or even inherently unique.

So our 'horizontal' test is this: if doing the same thing leads to the same results – *or is believed to lead to the same results* – then it's a Simple decision; if doing the same thing leads to different results, or if we need to do different things to get the same results, it's Not-simple.

[Yes, I do know that that's a Simple true/false distinction across a spectrum that in reality is fully modal. If you want to apply the appropriate recursion here, please feel free to do so: I thought it wisest here to keep it as simple as possible, because this can get complicated real fast, and unless we're careful to keep the complexities at bay we could end up with a right old chaos of confusion. Which is, yes, yet another recursion… Hence best to keep it simple for now, as best we can, acknowledge that much of it *isn't* Simple, and allow the recursions to come back in later when there's a bit more space to work with it.]

The crucial point about real-time is that there's no time available for a distinct sensemaking-stage: decision links directly to action, and vice-versa. (That's *why* it's called 'decision': the same linguistic roots as 'incision', it's literally 'cutting away', 'cutting apart', the cutting-edge for action in the 'now'.)

For sensemaking to take place, there *must* be a gap in time between one decision to the next. The key to John Boyd's

'Observe, Orient, Decide, Act' (OODA[60]) loop – which, importantly, is also not a loop[61] as such – is that it still allows distinct sensemaking ('Orientation') to take place, but keeps it as close to real-time as possible: that's what's meant by 'getting inside the opponent's OODA loop'.

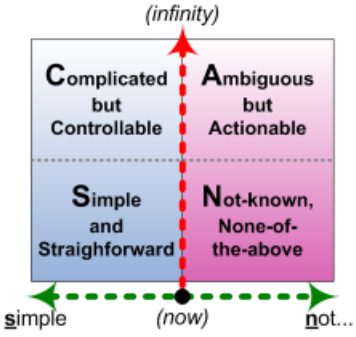
As time-available – the red-line 'vertical'-axis in SCAN – extends outward either side of real-time, the OODA-'loop' can become recursive, and thence, given enough time, simplified-out to a Deming-style 'Plan, Do, Check, Act' (PDCA[62]) continuous-review cycle, such as is also implied in the US Army's 'After Action Review[63]':

- "What was supposed to happen?" – what was our Plan?
- "What actually happened?" – what did we Do?
- "What was the source of the difference?" – what do we need to Check?
- "What do we need to do different next time?" – about what do we need to Act?

As I've described in other posts, sensemaking-choices tend to split as described in SCAN: there's a 'bump' on the path, indicated by the jump between simple true/false logic versus fully-modal logics of 'possibility and necessity' on the 'horizontal' axis, contrasted with a much

[60] http://en.wikipedia.org/wiki/OODA_loop
[61] http://www.dbrmfg.co.nz/Thinking%20Process%20Cloud%20OODA.htm
[62] http://en.wikipedia.org/wiki/PDCA
[63] http://en.wikipedia.org/wiki/After_action_review

smoother spectrum of choices as available-time extends in the 'vertical'-axis. Although the 'vertical' boundaries are less clear-cut than the 'horizontal' ones, this gives us the four SCAN quadrants – Simple, Complicated, Ambiguous, Not-Known:



SCAN sensemaking

Those distinctions determine the appropriate tactics for sensemaking, as described in those earlier posts.

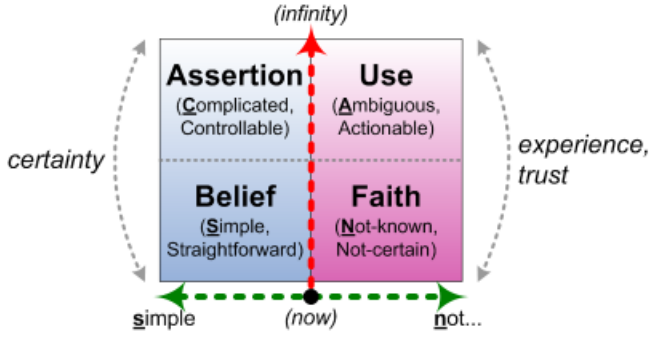
Decision-making seems to follow a similar, closely-related pattern – though that's the part I'm having trouble pinning down right now.

(Boyd's OODA is in part another attempt to pin down the same relationships; likewise Snowden's Cynefin, if rather less so. Jung's frame of 'psychological types[64]' is probably a closer fit than Cynefin for this: I've used a generic decision-types adaptation[65] of it for some decades now, though it's

[64] http://en.wikipedia.org/wiki/Psychological_type
[65] http://www.tomgraves.org/3science

still not quite right. Hence this exploration here.)

So again, it's 'work-in-progress', but this is where I've come to at present:



SCAN: **decision-making**

It's a decision-making frame based on the same horizontal (decision-modality) and vertical (time-available) axes as in SCAN, and hence the same sort-of-quadrants but with a decision-oriented re-labelling: Belief (Simple), Assertion (Complicated), Use (Ambiguous) and Faith (Not-known).

On the left-side of the Inverse-Einstein test, the mechanism that links Assertion and Belief is a drive for *certainty*, for 'control'. On the right-side, linking Use or 'usefulness' with the real-time openness of Faith, is more a focus on *experience*, underpinned by a deeper kind of *trust* – a trust which is often conspicuously absent in any concept of 'control'.

[For this post I'll focus more on what happens across

the horizontal-axis, the relationships between theory and practice, or 'truth' versus 'usefulness'. I'll explore more closely the interactions along the vertical-axis - between what we *plan* to do versus what we *actually* do - in a following post.]

In terms of decision-making tactics:

- on the **left**-side, **theory takes precedence over practice** – or, in some contexts, ideology rules, which is much the same
- on the **right**-side, **practice takes precedence over theory**

In essence, this is CP Snow's classic 'The Two Cultures[66]', the sciences (left-side) and the arts (right-side). Notice, though, that *technology sits on the right, not the left*: it *uses* theory, but that isn't its actual base – hence the very real dangers in the often-misleading term 'applied science'.

Bridging the gap, from left to right, is *praxis*, "the process by which a theory, lesson, or skill is enacted, practised, embodied, or realized"; and from right to left, is *pragmatics*[67], "a process where theory is extracted from practice". As enterprise-architects would be all too aware, the latter always starts from *pragma*[68], from "what is expedient rather than technically ideal": and it usually includes the

[66] http://en.wikipedia.org/wiki/The_Two_Cultures
[67] http://en.wikipedia.org/wiki/Pragmatism
[68] http://en.wiktionary.org/wiki/pragma

joys of 'realpolitik', of carefully filtering reality to fit in with other people's prepackaged assumptions…

That boundary denoted by the Inverse Einstein Test is all too real: whether the beliefs in question are 'scientific', religious, political or whatever, the 'need' for certainty will often trigger huge resistance against anything that doesn't fit its assumptions. For example, there's a very close mapping between this frame and the classic scientific-discovery sequence of **idea > hypothesis > theory > law**, which align with Faith, Use, Assertion and Belief respectively.

In real scientific practice[69], it's not a linear sequence, there's a lot of back-and-forth between each of the steps. And in principle, it *should* be a continuous-improvement cycle, a broader-scope form of PDCA. But as Thomas Kuhn[70] and many others have documented, that same 'need' for certainty often places a near-absolute barrier between supposed 'scientific law' and any new ideas – in other words, between Belief and Faith – that brings that cycle to a sudden halt, sometimes for years, decades or even centuries. All too often, in practice, if we take the real-time 'short-cut' from Belief to Faith, we will be forcibly forbidden to return along the same path: instead, we're forced to go 'the long way round', via Use and Assertion (hypothesis and theory) – which we may not have time to do. Which is a very real problem. And one that applies as much in enterprise-architecture as in any other field – as we've seen with the

[69] http://www.archive.org/details/artofscientifici00beve

[70] http://en.wikipedia.org/wiki/The_Structure_of_Scientific_Revolutions

inane IT-centrism that has dominated the discipline for far too long.

It gets complicated…

What I've been seeing, as I've explored this frame, is a whole stream of often-subtle misunderstandings and 'gotchas' that I've noticed time and again in practice in enterprise-architecture and elsewhere. These seem to be where many unnecessary complications and confusions arise – so it's worth noting them here.

For example, **fact arises from *experience***: its basis is on the right-side of this frame – *not* the left. What's on the left-side often purports to be fact: yet it's not fact as such, but *interpretation* of fact – a very important difference. The left-side operates on information, an interpretation of raw-data – but it often has no means to identify the source or validity of that information, or its method of interpreting it. (This is the same inherent problem whereby a logic is incapable of assessing the validity of its own assumptions: by definition, it *must* call on something outside of itself to test those premises.) So on the left-side, there's actually no difference between 'real' and 'imaginary' – which can lead to all manner of unpleasant problems if the left-side is allowed to over-dominate in any real-world context…

Importantly, there's *no real difference here* between '**objective**' **versus** '**subjective**': that distinction is actually another dimension that's somewhat orthogonal to this

plane. What I *feel*, or *sense*, is subjective, but it's still a fact; whereas how I *interpret* that feeling or sensation is not a fact – it's an interpretation. Telling someone that they should or shouldn't feel something is just plain daft: the feeling itself is a fact – something about which we *don't* actually have any choice – whereas the 'should' is an interpretation arbitrarily imposed by someone else.

[What we *do* in response to a feeling is a choice – literally, a 'response-ability' – and is something that *can* be guided by 'shoulds' and the like: but not the feelings themselves. That's a *very* important distinction which, sadly, surprisingly few people seem to understand…]

There is a specific sense in which subjective versus objective aligns somewhat with the 'less-time' versus 'more-time' on the SCAN *vertical*-axis. More-time means more time available for experimentation and analysis – and that can allow us to identify what's shared ('objective fact') across many people's experience, versus experiences that are more specific and personal ('subjective fact').

But there seems instead to be a tendency to conflate the objective/subjective distinction with the SCAN *horizontal*-axis – objective-fact as 'truth' on the left-side, subjective-fact as 'not-truth' on the right-side. There *are* ways in which that conflation can work – it's at the core of the Jungian frame, for example – but we need to be careful about it. Using that conflation to dismiss all subjective-fact as 'irrelevant' – as the classic 'command and control' models would do – not only makes no sense at all, but is

extremely unwise in real-world practice…

There also several other key distinctions across either side of the Inverse-Einstein test:

– '**science**' **versus technology**, which also parallels **ideology versus practice**: on the left-side, there's an assertion that something *is* 'true', whereas on the right-side we proceed *as-if* it's true – which is not the same at all.

– **organisation versus enterprise**: the nature of an organisation is that it's about left-side themes such as control, beliefs, repeatability and certainty; the nature of an enterprise is that it's *not* certain, "a risky venture" and suchlike – with all that that implies.

– **structure versus story**: most structures within current enterprise architectures will, again, have a left-side focus on providing repeatability and certainty; story and other forms of narrative-knowledge provide an alternate kind of 'structure' that holds many of the right-side themes together

– **sameness versus uniqueness**: another key enterprise-architecture theme, sameness and repeatability is very much a left-side theme, whereas uniqueness is just as much a right-side theme

– '**best-practice**' **versus** '**worst-practice**': the notion of 'best-practice' assumes that practice that worked well in one context will be directly applicable to another, the same success repeatable in another; by contrast, maintenance engineers and others who work extensively with unique

or near-unique contexts share their learning more through 'worst-practice', stories of what *didn't* work in a given context. (I think I first heard that one from Dave Snowden? – credit where credit's due, anyway.)

The trade-offs across each of these dichotomies all have direct implications for the design and structure of any enterprise-architecture.

Implications for enterprise-architecture

Take a look at those dichotomies again: which side do you think is emphasised by current enterprise-architectures?

The obvious answer is that, almost invariably, the left-side is given priority over the right.

However, this has huge consequences for the effectiveness of the overall enterprise, and for the enterprise-architecture that describes it:

- interpretation takes priority over fact: never a good idea…
- theory and ideology takes priority over practice and experience: that's almost a definition of (misused) Taylorism…
- the need for (spurious) 'certainty' and 'control' takes priority over trust of anything or anyone: ditto on Taylorism…

- the reliance on true/false decision-methods can render the organisation unable to cope with any form of uniqueness
- the need to force-fit everything into sameness of *content* – 'best practice', IT-centric BPR and the like – fails to grasp the differences of *context*
- the over-focus on organisation – 'the letter of the law' – literally kills off the spirit of enterprise…

Look at most of our existing EA toolsets, too: can you find *any* toolset that's actively designed around anything other than true/false logic? Other than in rare model-types such as ORM[71] (Object-Role Modelling), there's no means to describe modality in relationships – hence, for example, no directly-supported way to describe a *usable* reference-model that allows for real-world ifs, buts and perhapses.

And whilst every toolset focusses on structure – and most do that very well, too – how many of those toolsets also help us to focus on the counterpart of story? They might support few use-cases, perhaps, but that's about it: there's a *huge* gap in capability there…

What we *need*, urgently, is a better balance between structure and story, between theory and practice, between organisation and enterprise. And without adequate support in the toolsets, that means that we have to create that balance ourselves.

[71] http://www.orm.net/

The crucial point is that this balance is not an 'either/or', but a much more modal 'both/and':

- theory *and* experience
- 'objective' *and* 'subjective'
- 'science' *and* technology
- certainty *and* trust
- true/false *and* fully-modal
- organisation *and* enterprise
- structure *and* story
- sameness *and* difference
- 'sense' and 'nonsense[72]'
- certainty *and* uncertainty

We will only achieve a real effectiveness in the architecture via a fully-nuanced 'both/and' balance across all of these dimensions, and more.

So take a careful look at your own organisation, your own enterprise-architectures and the like: where is it out of balance, in this sense? In SCAN terms, how much does it over-emphasise the left-side at the expense of the right-side? And what can (and must) you do to bring it back into a better balance overall?

[72] http://www.nytimes.com/2009/10/06/health/06mind.html

Source (Tetradian weblog)

- *Date*: 2011/12/19
- *URL*: decisionmaking-belief-fact-theory-practice[73]
- *Comments*: 5
- *Categories*: Complexity / Structure, Enterprise architecture, Knowledge
- *Tags*: Business, chaos, complexity, effectiveness, enterprise, Enterprise architecture, Knowledge, methodology, SCAN, taxonomy

[73] http://weblog.tetradian.com/decisionmaking-belief-fact-theory-practice

Using recursion in sensemaking

This was such a good question from Paul Beckford, in one of his comments[74] on the previous post[75], that I thought it was worthwhile bringing it out into more accessible form here:

> "I don't understand the recursion you speak of and the real time nature of decision making and how that is different from 'considered' decision making."

I'll deal with the easy bit first: real-time versus 'considered'. Let's use a really simple (and, at present, topical) example: New Year's Resolutions.

- Did you make any New Year's Resolutions? If you did, that's a 'considered' decision, at some distance from the point of action – an *intent*.

[74] http://weblog.tetradian.com/more-on-principles-and-decision-time/#comment-79905

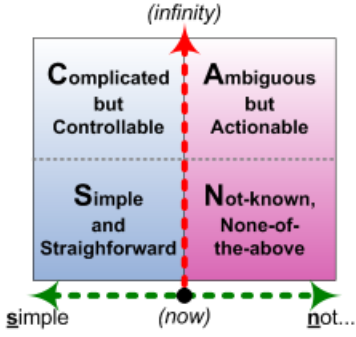
[75] http://weblog.tetradian.com/more-on-principles-and-decision-time/

- Assuming you did make a New Year's Resolution, did you actually keep to it, in terms of what you *actually* do and did? – because that's a real-time decision.

Given the above, notice how well (or not) the 'considered' decision-making lines up with the actual decisions made at the point of action. Overall, that's an important part of *enterprise-effectiveness*. That's what I've been working on, with the SCAN posts and the like.

(There's also how review-processes such as PDCA and After Action Review etc link up with all of this: how the review of what we intend versus what we actually did is used to challenge and re-align the linkage between what we intend and what we do next time. If there *is* a 'next time', of course: it gets even trickier if there isn't…!)

The other point: *recursion*. For this context, recursion occurs when we use a framework on itself, to review or work with or refine itself. Let's use just the sensemaking side of the SCAN frame for this, it should (I hope!) be a safe and uncontroversial example.



SCAN: **sensemaking**

So, we would say that this frame has four domains:

- Simple
- Complicated
- Ambiguous
- Not-known, None-of-the-above

And the boundaries of those domains are defined by two axes:

- *horizontal*: modality – true/false on left, uncertain ('possibility/necessity') on right
- *vertical*: distance in time (or time-available-until-irrevocable-decision) – from point-of-action to potentially-infinite time-available

At first glance, that's a really simple categorisation. Note the word *'Simple'*.

Then we notice that our Simple categorisation starts to get *Complicated*. The boundaries between the domains aren't as fixed as they might at first seem: although there's a definite 'bump' on the horizontal axis (what I've termed the 'Inverse-Einstein test'), it's actually a continuous spectrum of modality, from predictable to somewhat-variable to a lot of variation to everything inherently-unique with no pattern at all.

(The Inverse-Einstein test: on the 'order' side (Simple/Complicated), if we do the same thing, we expect to get the same result; on the 'unorder' side (Ambiguous/None-of-the-above), if we do the same thing, we may get a different result, or we may need to do different things in order to get the same result.)

And the vertical axis is always a completely continuous spectrum: there is a clear transition *somewhere*, between the 'Newtonian' (Complicated/Ambiguous) and 'quantum' (Simple/Not-known) levels, but we can't define explicitly where it is.

Then our Simple-but-also-Complicated categorisation starts to get *Ambiguous* as well: we'll see this especially when we use cross-maps, such as that one about skill-levels, where each skill-level represents a different *mix* of 'order' or 'unorder', again with no clear boundaries, and with a fair few emergent-properties arising as well.

And then we recognise also that there are aspects in this Simple-and-Complicated-and-Ambiguous categorisation that are inherently-unique, scattered all the way through every-

thing we're looking at, with some bits that are definitely *Not-known* or None-of-the-above. (In fact that's the whole point of this kind of exploration, trying to make sense of those Not-known items and come to some useful actionable decisions about them.)

And, yes, once we dig deeper, we'll find that the same kind of pattern recurs at another level, and then deeper again, and so on.

Fractal, self-similar, recursive; Simple, Complicated, Ambiguous, None-of-the-above, all of them weaving through each other, all at the same time.

That's what I mean by recursion here: the framework used to explore itself, and explore the exploring of itself, and – of course – of what it is itself being used to explore.

Makes sense? I hope? :grin:

Source (Tetradian weblog)

- *Date*: 2012/01/15
- *URL*: using-recursion-in-sensemaking[76]
- *Comments*: 1
- *Categories*: Business, Complexity / Structure, Enterprise architecture

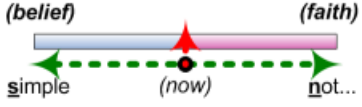
[76] http://weblog.tetradian.com/using-recursion-in-sensemaking

- *Tags*: Business, chaos, complexity, decision-making, effectiveness, enterprise, Enterprise architecture, SCAN, sense-making

Rules, principles, belief and faith

Following on from the previous post 'Rules, principles and the Inverse-Einstein Test[77]', there's an important corollary about real-time sensemaking and and decision-making – it was in my notes for the post, but I forgot to include it, so I'll do it as a separate post here.

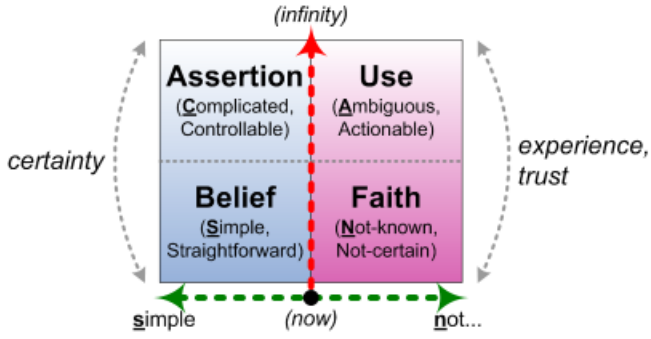
This connects back to the previous work on SCAN, and, in particular, relates to how sensemaking and decision-making[78] work *at the point of action*[79]:



SCAN: belief and faith

This probably also relates to the 'considered' sensemaking and (would-be) decision-making that come into play when there is more separation-in-time from the moment of action itself:

[77] http://weblog.tetradian.com/rules-principles-and-inverse-einstein/
[78] http://weblog.tetradian.com/decisionmaking-belief-fact-theory-practice/
[79] http://weblog.tetradian.com/belief-and-faith-at-point-of-action/



SCAN: **decision-making**

(In that diagram above, 'Assertion' is closely related to ***analysis***, and 'Use' to ***experiment***.)

In both those diagrams, the green horizontal-axis represents the *modality* of the context – the extent to which the context ranges from certain-repeatability and predictability ('simple') to an increasingly-ambiguous and uncertain realm of possibility and necessity ('not-simple'). The black dot where the red vertical distance-in-time axis crosses the modality-axis is, in effect, the boundary-condition of the Inverse-Einstein test: to the left ('*simple*', a context of ***order***), doing the same thing will always lead to the same result; to the right ('*not-simple*', a context of ***unorder***), doing the same thing *may* lead to different results, with the probability of different-results increasing as we go further to the right.

(Or, to put it another way, over on the left, the *variety-*

weather[80] is calm enough that 'control' will seem to be achievable; over to the right, the variety in the context is greater than that of the would-be control-system, and hence certainty of control is *not* possible. Note that the variety of variety itself means that the notion of 'control' is rarely more than a questionably-convenient fiction whose nominal validity will itself vary over time. In reality, in almost all real-world systems, indefinitely-guaranteed certainty of control is not achievable even in theory, let alone in practice. But I digress… again… :grin:)

Another way to view this in terms of perspectives, as in my post 'Inside-in, inside-out, outside-in, outside-out[81]':

- ***belief*** and ***assertion*** ('analysis') tend to take an **inside-out** view: they apply ***rules*** or ***algorithms*** – predefined, closed-logic, and often *self-centric* sets of assumptions - *onto* the overall context, and in effect attempt to force the context to fit the assumptions
- ***faith*** and ***use*** ('experiment') tend to take an **outside-in** view: they use preselected **values**, **principles**, **patterns** and **guidelines** as open-logic, externally-aware filters onto the overall context, and in effect attempt to adapt *to* the context, often in real-time
- *heuristics* – semi-open filters such as 'best-practice' or past-experience – tend to straddle the boundaries between these two modes

[80] http://weblog.tetradian.com/requisite-variety-and-stormy-weather/
[81] http://weblog.tetradian.com/inside-in-inside-out-outside-in-outside-out/

The core advantage of belief is that it enables us to proceed, often at very high speed, with absolute certainty – *if* the belief-logic is valid for the context. The catch with belief is that it is *very* prone to circular-reasoning, in part because the 'inside-out' view precludes visibility of any information (or *anything*, really) that does not fit its predefined assumptions. (Hence, for example, the frustratingly-persistent predominance of IT-centrism in much of so-called 'enterprise'-architecture…) Also, almost by definition, belief-based decision-making has no means *within itself* to verify the validity of its own logic or assumptions – which is *why* it is so prone to circular-reasoning.

The core advantage of faith is that it can cope with almost any potential variation in the context – dependent on skill-level. The catch with faith is that it can easily become 'blind faith', based on principles that may actually have little or no connection with concrete reality. Also, almost by definition, it can be all but impossible to demonstrate a repeatable chain-of-reasoning in faith-type sensemaking and decision-making: this often renders reliability of action within the system highly dependent on individual skill.

Given those respective advantages and disadvantages, it should be clear that neither belief-based decision-making nor faith-based decision-making would be sufficient on its own for acting on any real-world system: ***we need a balance between belief and faith to guide real-time decision-making***.

Source (Tetradian weblog)

- *Date*: 2012/07/16
- *URL*: rules-principles-belief-and-faith[82]
- *Comments*: 4
- *Categories*: Business, Complexity / Structure, Enterprise architecture, Knowledge
- *Tags*: belief, complexity, decision-making, enterprise, Enterprise architecture, faith, inverse-Einstein test, principles, rules, SCAN, sense-making

[82] http://weblog.tetradian.com/rules-principles-belief-and-faith

More keywords for SCAN

Some notes that came up for me almost a month back, on the SCAN[83] framework for sensemaking and decision-making, and that I hadn't gotten round to documenting until now.

I was reminded that I *hadn't* posted these notes when I saw Mark Foden refer to the Cynefin framework in his talk at the Integrated EA conference[84], to illustrate the difference between a context that's merely complicated, and one that's genuinely complex – the latter often because it includes a variety of 'people-issues' and suchlike. The Cynefin framework *is* quite useful for illustrating that difference.

Yet over the past few years – as I've described in quite a few of my previous posts, such as 'A human view of Simple, Complicated and Complex[85]', and 'Ensuring that the Simple stays simple[86]' – I've become increasingly wary of the term 'complex': the problem is not in the term itself, but in

[83] http://weblog.tetradian.com/tag/scan/

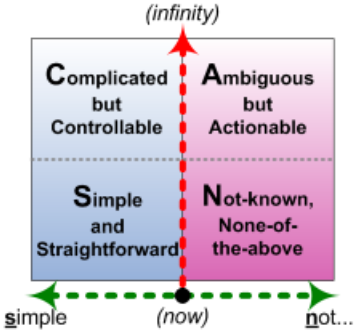
[84] http://weblog.tetradian.com/at-integrated-ea-2013/

[85] http://weblog.tetradian.com/human-view-of-simple-complicated-complex/

[86] http://weblog.tetradian.com/ensuring-that-the-simple-stays-simple/

the fact that people interpret it in so many different ways. For example, much of what IT-folks might call 'complexity' – because it's, well, *complex*, isn't it? – would be described instead by aficionados of complexity-science merely as high levels of 'complicatedness' – the key distinction being that it expects a predictable and repeatable result. Hence a *lot* of unnecessary confusion and, occasionally, heated argument, that really doesn't help anyone at all. That's one of the key reasons why, in SCAN, I switched over to using 'Ambiguous' instead: it's perhaps not as precise a term as 'complex', but at least it's unambiguous about the existence of the ambiguous!

Which brings me back to those notes, because I'd been building small collection of alternate keywords for each of the 'domains' in SCAN. None of these keywords *replace* anything: it's just that, following the same principles of context-space mapping that underlie SCAN itself, they add extra layers of richness to the sensemaking by providing a kind of contrast, cross-reference, cross-check and suchlike. Anyway, here's the list so far:



'S' domain: *Simple and Straightforward* (high-repeatability, at or near moment of action):

- Step-by-step
- Speed [don't feel, don't think, just *do*]
- Switch [as in predefined flowchart, 'no thinking required']

'C' domain: *Complicated but Controllable* (high-repeatability, at distance from moment of action):

- Correct
- Calculable
- Certain
- Confirmable
- Conformity
- Compliance
- Certifiable

'**A**' domain: *Ambiguous but Actionable* (low-repeatability, at distance from moment of action):

- Adaptable
- Adjustable
- Amenable
- Assessable
- Ask [as in the uncertainty of experimentation, rather than the (apparent) certainty of calculation]
- Angst-laden! :wry-grin:

'**N**' domain: *Not-known, None-of-the-above* (low-repeatability, at or near moment of action):

- No idea
- No time to think
- Newness [as in (near)-unique, therefore no certain rules]

Any other suggestions that you'd add to this?

[***Update***: in the comments, Nigel Green suggested:

- A = Adoption-led (As in LiT Adoption Engineering), Agile-practice, Answers-emerge
- C = Capability-constrained (Expertise required), Constrained (bounded), Calculated (formula applied), Conforming (to rules)

- N = Nascent, Non-conforming, Next-practice, Not yet, Non-compliant
- S = Sorting, Sifting, Stacking, Shelving (as in archive), Squirreling (filing)]

Source (Tetradian weblog)

- *Date*: 2013/03/14
- *URL*: more-keywords-for-scan[87]
- *Comments*: 10
- *Categories*: Complexity / Structure, Knowledge
- *Tags*: complexity, context-space mapping, SCAN, sense-making, sensemaking

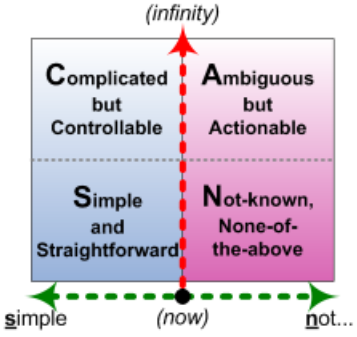
[87] http://weblog.tetradian.com/more-keywords-for-scan

A simpler SCAN

The SCAN[88] frame provides a simple means to make sense of complexity and decision-making in almost any context. Yet is there perhaps an even simpler way to describe the principles and practice behind it?

I've been nibbling at that question for quite a while, and it's now gotten to the point where it's probably worth describing what I've come up with so far.

To start with, here's the sensemaking-version of the usual SCAN frame:



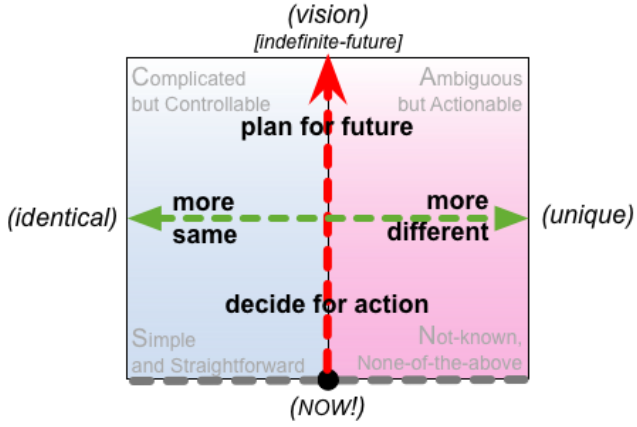
It shows the boundary-conditions: the 'Inverse-Einstein boundary' between doing the same thing leading to same

[88] http://weblog.tetradian.com/tag/scan/

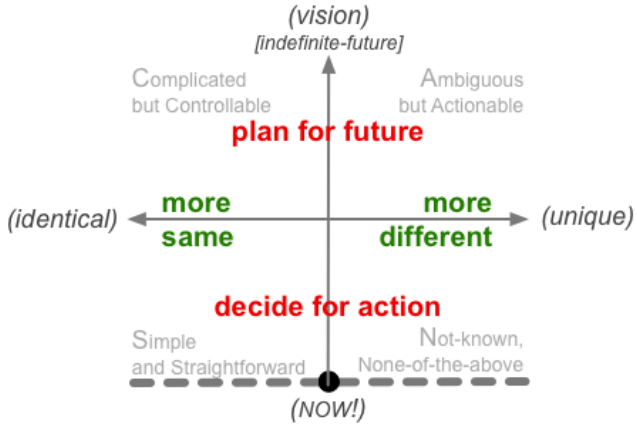
results (left) versus different results (right); and the transition from 'rational' or 'considered' decision-making at distance from the point-of-action (above the horizontal-axis) to emotion-based decision-making close to or at real-time (below the axis). It shows the 'domains' implied by those boundary-conditions: Simple, Complicated, Ambiguous, Not-known - hence **SCAN**.

It works well, and provides a useful focus for cross-comparison with all manner of other ways to make sense of a context. Yet it doesn't really describe *what those distinctions look like in practice.*

So let's focus instead on what happens *along each of the axes.* Which gives us something more like this:



Or, even simpler, like this:



In effect, we've changed the emphasis from boundary-conditions (vertical and horizontal) to the spectra (horizontal and vertical) that extend either side of those boundary-conditions. In other words, the actual axes, rather than the effective boundaries that occur (and often move, dynamically) along those axes.

[It's the *same* frame, giving us the *same* implied 'domains' in the same relative positions: it's just a different way of looking at it, that's all.]

Between sameness and uniqueness

Either side of the Inverse-Einstein boundary, we bounce back and forth between same and different[89]:

- the extreme of **sameness**__ is when everything in scope is exactly ***identical***
- the extreme of ****difference** is when everything in scope is utterly ***unique***, sharing no sameness with anything else in that scope

In between, everything has its own distinct mix of *sameness* and *uniqueness*.

And the catch there is that methods that assume sameness - as most IT-systems still do, for example - may not work well, or at all, with any kind of difference. Conversely, methods that assume that everything's unique tend to be very difficult to scale. (See the post 'Scalability and uniqueness[90]' for more detail on that.)

The range and mix of sameness and difference has very real impacts on system-design:

- conventional rule-based machines and IT enable high scaling of sameness, but in practice will often struggle even with small degrees of difference

[89] http://weblog.tetradian.com/same-and-different/
[90] http://weblog.tetradian.com/scalability-and-uniqueness/

- humans can handle increasing levels of difference, dependent on skill[91], but on their own can only work at a human pace

Any system that needs high-scalability (handling large numbers of items at speed), *and* must also cope with medium to high levels of variation and difference in those items, will usually need a mix of automated *and* 'manual' processes - and the ability to detect when and how to switch between those processes, as appropriate, in accordance with the level of difference and the skill-levels of the people working within those processes.

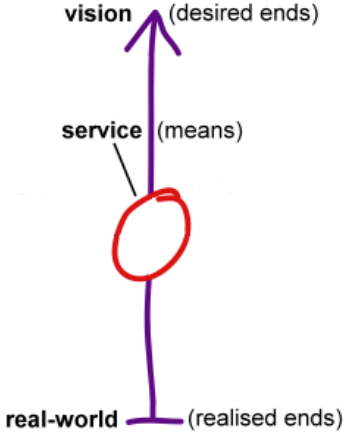
Note that the real-world *always* contains a mix that encompasses the entirety of that spectrum, from identical to unique. And since 'control' inherently relies on repeatability, which in turn relies on assumptions of 'sameness', **no automated system will ever be able to provide complete 'control' in any complete real-world context**. In practice, the *illusion* of 'control' is provided - or propped up - by artificially-imposed constraints on what is considered 'real' and what is not, or what is considered 'in scope' or not. *_However,* those constraints do not remove the uncertainty*: it's still part of reality, hence still always exists, regardless of those artificial constraints. When difference from the real-world breaches those arbitrary constraints - which, by definition, it _must* do at some point - then the illusion of 'control' that underpins the design of the system will be

[91] http://weblog.tetradian.com/over-certainties-of-certification/

shattered, and the system itself will fail. No matter how desirable 'sameness' and certainty may be, ***difference and uniqueness are a fact of life***: hence *we always need to be aware of and design for that fact* - and not pretend that it doesn't exist…

From plan to action

In the other direction, the vertical-axis in SCAN, we work with the tension between the difference between what we ***plan to do***__ versus what we ***actually do***. In many ways, this is the same tension as that between desired-ends and realised-ends that, in Enterprise Canvas[92], underpins and provides the ultimate motivation for any service:



[92] http://weblog.tetradian.com/tag/enterprise-canvas/

Moving 'upward', the vertical-distance here represents effective 'distance in time' from the point of action - the moment at which all previous decisions and plans are enacted, applied into real-world action. Or *should* be enacted, rather, because the reality is that often they're not: there can sometimes be a very large gap between what was intended, versus what is actually done at the time.

The point here is that there's another very real boundary somewhere along this axis, driven largely by the fact that the real-world won't sit around and wait for us to make up our minds what we're going to do.

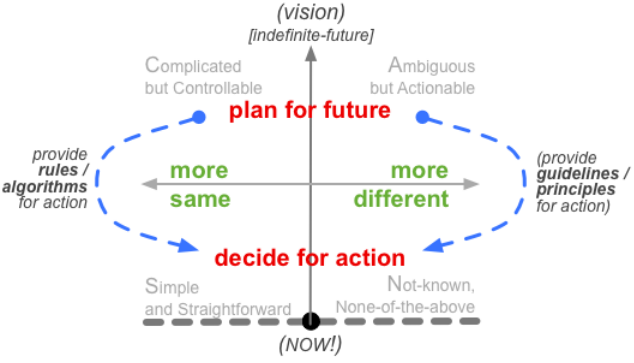
This applies to *everything*: not just to people, but to machines and IT-automation too. Whichever way we do it, there will always a somewhat-tortuous trade-off between speed and complexity: there will always be real constraints on just how complicated any 'control-law' can be, even for the fastest of real-time automation.

For rule-based machines and IT, the effect is that we'll usually have to work out all of those complexities and complications at some distance away from real-time, through ***analysis*** and ***experiment***. We then repackage those understandings into ***rules*** and ***algorithms***__ that *are* compact enough to run at real-time - a translation from **Complicated** to **Simple**.

It's sort-of the same for real-people, though one important difference is that, given sufficient skill, people can also tackle inherent-difference, often all the way to inherent-uniqueness, in real-time action. Decision-making for the

latter is supported by ***guidelines*** and ***principles*** - a translation from **Ambiguous** to **Not-known**, something that few machines or IT-systems can handle. Again, these guidelines and principles are developed via analysis and experimentation, at some distance in time from the real-time action.

The aim is that the ***plan for the future*** translates as directly as practicable to ***decisions for real-time action*** - though the paths to do this translation can vary somewhat, mainly dependent on whether we're dealing with real-time sameness ('Simple') or difference ('Not-known'). We can summarise those two types of 'translations' in visual form as follows:



The real complication here, though, is that, for humans, decision-making at or close to real-time is *fundamentally* different to that that's used in most 'considered' decision-making: ***real-time decision-making is primarily emo-***

tional, not 'rational'. For humans, that transition between decision-mechanisms is the core boundary-condition along this vertical-axis of distance-in-time.

Those emotion-based decision-mechanisms are capable of handling only very limited numbers of choices or options in real-time. In practice, the typical limit is somewhere around five to nine options - dependent on a wide variety of factors that I won't detail here. This is one of the key reasons why action-checklists and sets of principles for real-time use need to be constrained to no more than that size: too many options tends to trigger either a shutdown, a total stop, or a breakdown into what is colloquially described as 'chaos' or 'panic' - neither of which lead to viable response to the respective conditions.

(Yes, I do know I'm over-simplifying here - but the aim is to keep the description as simple as possible, after all!)

Summary

This simpler version of SCAN has essentially the same two axes as before, but with (I hope!) more easily-understood descriptors:

- *modality* (horizontal-axis, vertical boundary): towards ***more-same*** and *identicality*, or towards ***more-different*** and *uniqueness*
- *distance-from-action* (vertical-axis, horizontal boundary): from ***plan-for-the-future*** to ***decide-for-action***,

linking between the *vision* for the context (indefinite-future) and the real-time *NOW!*

I hope that makes SCAN even easier to use, anyway.

Source (Tetradian weblog)

- *Date*: 2013/06/07
- *URL*: a-simpler-scan[93]
- *Comments*: (none)
- *Categories*: Complexity / Structure, Enterprise architecture
- *Tags*: complexity, decision-making, Enterprise architecture, SCAN, sense-making

[93] http://weblog.tetradian.com/a-simpler-scan

SCAN as ‘decision-dartboard’

How to use SCAN[94] as a ‘decision-dartboard’ in work-planning? That was probably the highlight in a great conversation outside the IRM-EAC[95] conference in London earlier this week with Kai Schlüter[96], enterprise-architect at Danish engineering conglomerate Danfoss[97].

Kai heads up a team of about 30 architects who support some 500 software-developers worldwide. As I mentioned in the summary of his talk at the UnicomEA 2013[98] conference, he takes agile-development almost to extremes: turnround for most fixups in two hours or less, turnround for business-changes often less than a day. By intent, they don’t use a ‘proper’ repository-based EA toolset: instead, their most important EA tools are the whiteboards in every room.

In short, *fast*.

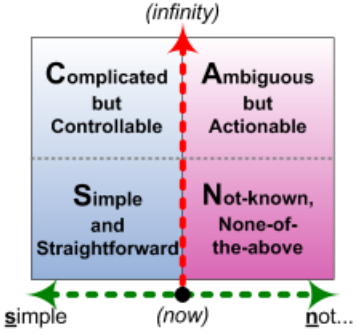
But not always - because it *can’t* always work that way.

[94] http://weblog.tetradian.com/tag/scan/
[95] http://www.irmuk.co.uk/eac2013/
[96] http://twitter.com/chbrain
[97] http://www.danfoss.com/
[98] http://weblog.tetradian.com/at-unicom-ea-2013/

Which means that they have to decide - fast - what they *can* do fast, and what they can't.

And that's where SCAN comes into the picture.

In its standard form, SCAN denotes *ways of interpreting and deciding* within an overall context-space[99]:



We view the context in terms of two dynamic dimensions[100] - sameness versus difference, and time-distance from the moment of action - which in effect gives us four quadrants: Simple, Complicated, Ambiguous, Not-known. Yes, it looks much like the all-too-typical two-axis frame so beloved by so many consultants, but as you'll see if you work your way

[99] http://weblog.tetradian.com/tag/context-space-mapping/
[100] http://weblog.tetradian.com/a-simpler-scan/

back through some[101] of[102] the[103] various[104] posts[105] here[106] on[107] the[108] SCAN[109] framework[110], there are a whole lot of subtleties and nuances and overlays and suchlike that, in principle, really ought to be taken into account whenever we use the framework.

Me being me, of course, I worry away at all of that detail, finding new layers, new correspondences, new applications. I worry away at getting it right, making it as versatile as possible.

Kai being Kai, of course, he doesn't waste time on worrying. Instead, he instead strips it right back to the core: "How can I use it for *this* need, right *here*, right *now*? Most of all, how can I use it *fast*?"

(I'll have to paraphrase Kai's description a bit at this point - please correct me if I get it wrong, Kai?)

In his work-planning sessions, he doesn't use SCAN as a context-space map: instead, he uses it as a kind of 'decision-dartboard'. He's partitioned his team's work-methods in terms of the SCAN quadrants:

[101] http://weblog.tetradian.com/on-metaframeworks-in-ea/
[102] http://weblog.tetradian.com/methods-mechanics-approaches/
[103] http://weblog.tetradian.com/principles-and-checklists/
[104] http://weblog.tetradian.com/control-complexity-and-chaos/
[105] http://weblog.tetradian.com/best-practices-adapt-then-adopt/
[106] http://weblog.tetradian.com/reframing-entropy-in-business/
[107] http://weblog.tetradian.com/working-with-i-dont-know/
[108] http://weblog.tetradian.com/more-keywords-for-scan/
[109] http://weblog.tetradian.com/control-complex-chaotic/
[110] http://weblog.tetradian.com/scalability-and-uniqueness/

- *Simple*: it's routine, we know how to do this, just do it
- *Complicated*: we'll need to do some analysis first, but it's within known space, and we can give an exact time-estimate
- *Ambiguous*: some parts of it aren't clear, we'll need to do some experiments, but we'll deliver something usable within a known-time-box - though it might need another update later
- *Not-known*: some of it is novel, new, in unknown territory, we can't guarantee usable results but we'll see what we can come up with within the set time-frame

And then, as Kai put it, "We throw projects at it, and see where they stick. Thirty projects, and in twenty-five minutes that's the whole project-plan done." Wow! In a word, *brilliant*.

Was SCAN *designed* to work this way, as a simple (some would even say simplistic) categorisation-framework? Definitely not.

Is it appropriate for SCAN to be *used* in this way? Definitely yes.

(Though preferably with a solid understanding of how SCAN 'should' be used - which Kai certainly has.)

That's actually the difference between 'design-intent' versus 'affordance[111]': SCAN wasn't *designed* to be used in

[111] http://en.wikipedia.org/wiki/Affordance

this way, but it *affords* the possibility of using it that way, in a way that really does work well. So if you know what you're doing with it, just do it. Kinda simple, really.

I love seeing new stories about how the tools and techniques I'd developed are being used to deliver real results in real-world practice - especially if they're being used in ways that I didn't expect. So keep those stories coming, y'hear? :-)

[***Update:*** Kai sent through a few corrections and updates that are well worth including here…]

Despite some inaccuracy in the numbers quite accurate how we use it.

To sort the numbers for you:

1. Roughly 30 Architects (bit more at the moment)
2. Roughly 500 People in IT (+ externals, so 500 Developers might be true, or not, we do not know exact)
3. 85% of all severity 1 defects recognize-to-fix < 2 hours
4. Daily Delivery to Production (that is not necessarily one day from Idea-to-Production
5. best case was 25 potential projects in 30 minutes. :)

The 3 ways to use the delivery darting (some are projects, but not all):

1. we use EPIC SCAN[112] to understand why we are where we are. So very literally the EPIC of how we got where we are.
2. we use WISE SCAN[113] to understand where we want to go. So very literally if it is WISE to go where we thing we should go.
3. we use PACE SCAN[114] to understand at what speed we can change. So very literally what is the PACE to change.

The process is simple: gather a group of clever people (Architects in our case) and let them play a variant of Planning Poker till there is agreement on one Architecture Item SCAN value (consensus on S, C, A or N) and then move to the next.

Tools we use for it: Whiteboard, Excel and SharePoint (in that order).

Source (Tetradian weblog)

- *Date*: 2013/06/15
- *URL*: scan-as-decision-dartboard[115]

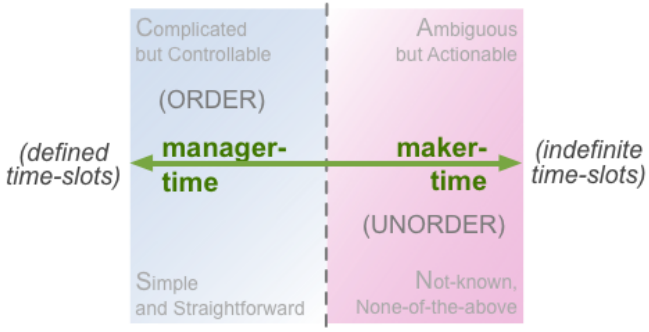
[112] http://socialea.chickenbrain.de/2013/02/dreamtime-epic-scan.html
[113] http://socialea.chickenbrain.de/2013/02/dreamtime-wise-scan.html
[114] http://socialea.chickenbrain.de/2013/02/dreamtime-pace-scan.html
[115] http://weblog.tetradian.com/scan-as-decision-dartboard

- *Comments*: 2
- *Categories*: Complexity / Structure, Enterprise architecture
- *Tags*: decision-making, Enterprise architecture, kai schluter, SCAN, sense-making

SCAN - some recent notes

Still somewhat in catch-up mode, but seems like this'd be the right time to summarise a few ideas that have come up in recent months on the SCAN[116] framework for sensemaking and decision-making.

First of these is a simple cross-map with the **manager-time**, **maker-time**[117] dichotomy:



Managers, as per classic 'scientific management', tend to view the world in discrete blocks of time. Their world is *certain*, is *definite* - if often quite a long way removed from the

[116] http://weblog.tetradian.com/tag/scan/
[117] http://www.paulgraham.com/makersschedule.html

real-time action. To business-managers and line-managers, meetings and suchlike should occur exactly on-schedule, to a predefined timeline, and each meeting should last exactly the time specified in the schedule. (The word '*should*' tends to occur quite a lot here…) To project-managers and production-managers, every task should take place within a similarly predefined timeline or schedule, and each task should complete in no more than a known amount of time - all defined, no doubt, by the dreaded time-and-motion men[118] or their more up-to-date equivalents. And if anyone fails to match up to these expectations - if anything takes any time longer than expected - they're derided as being 'behind schedule'.

Makers, meanwhile, live in a very different world - and one that, by definition, is much closer to the real-time action. Here, tasks take as long as they take: and the more uncertainty there is about what needs to be done, the more uncertain is the estimate of how long it will take.

For makers, there's often also a real problem with interruptions. With knowledge-work especially - such as in enterprise-architecture, where we'll often need to 'get our head around' some very large chunk of some context or system - it can take hours to build up the mental picture, and sometimes a mere moment's interruption to lose the whole lot and have to start again from scratch. Being forced to break off in the middle of something for some manager's meeting can cause a huge hit against productivity: a couple

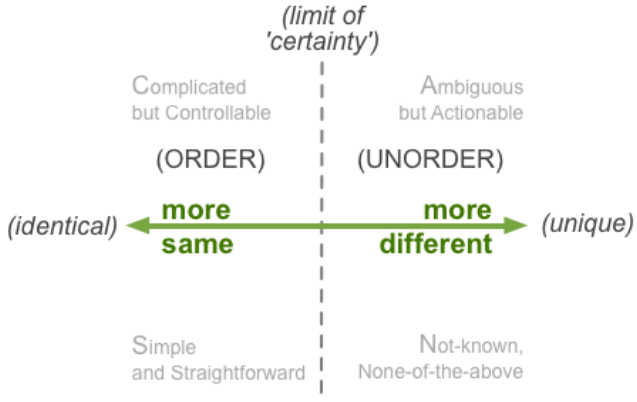
[118] http://en.wikipedia.org/wiki/Time_and_motion_study

of meaningless meetings in the middle of either side of the business-day - to satisfy some manager's need for 'certainty' as to how the schedule's going - can easily kill off any possibility of usable work in that entire day. Not good from *anyone's* perspective… yet all too common, too.

This is one reason why I so much like Kai Schlüter's 'SCAN as decision-dartboard[119]': it's a simple, practical way to show managers where each task *really* sits on that axis of certainty-to-uncertainty, from manager-time to maker-time - and hence how each respective task needs to be managed, in terms of time and uncertainty.

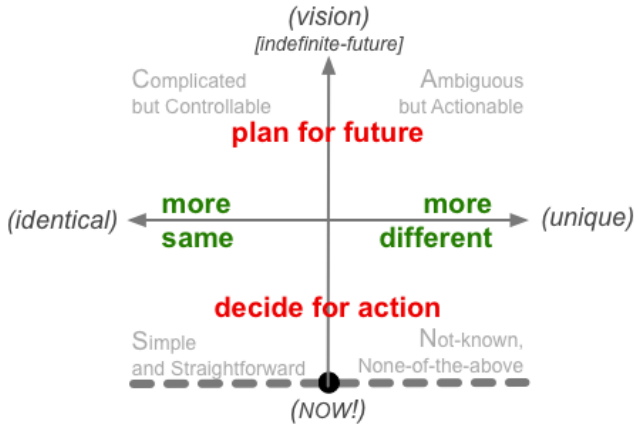
Next item is **an alternate way to indicate the SCAN axes**. The usual way I've shown these is as the delimiters or boundaries between the SCAN domains, crossing over in the centre of the context-space map. The horizontal-axis I've usually shown as a double-headed arrow:

[119] http://weblog.tetradian.com/scan-as-decision-dartboard/



...and the vertical-axis as a single-headed arrow, anchored at the base-point of 'Now!':

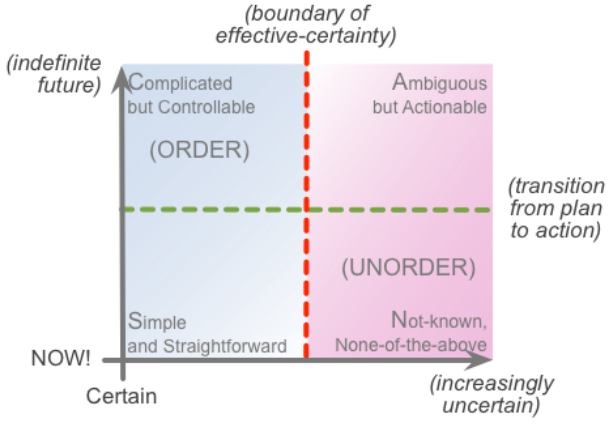
...which we would bring together as:



But actually, the horizontal axis should perhaps be shown as a single-headed arrow too, because whilst the right-hand side extends to infinity, the left-hand side represents an absolute limit such as absolute-certainty or absolute-predictability. We'd still need, though, to show those (dynamically-changing) boundary-conditions along the two axes, that in effect demarcate the SCAN domains:

- doing the same thing should lead to the *same* result (left-side) or can lead to a *different* result (right-side)
- "in theory there's no difference between theory and practice" (above the line); "in practice there is [a difference]" (below the line)

Which, overall, gives us this:



I'm not saying we should always use this instead of the centre-axis layout: it's just that for some purposes - some types of sensemaking and decision-making exercises - it may be more usable or easier to understand.

Next, a set of **keywords to clarify what happens in each 'domain'**. For me, these have been around for a very long time: the first version, which in those days I described as R4 and, somewhat later, as R5, dates way into the tag-end of the last century. These days I perhaps should describe the set as R8, because it now has eight keywords - two for each of the SCAN domains:

– ***Simple***

- *regulation* - reliance on rules and standards
- *rotation* - systematically switch between different elements in a predefined set, such as in a work-

instruction or checklist (or in the elements in a framework such as SCAN itself)

– ***Complicated***

- *reciprocation* - interactions should balance up, even if only across the whole
- *resonance* - systemic delays and feedback-loops may cause amplification or damping

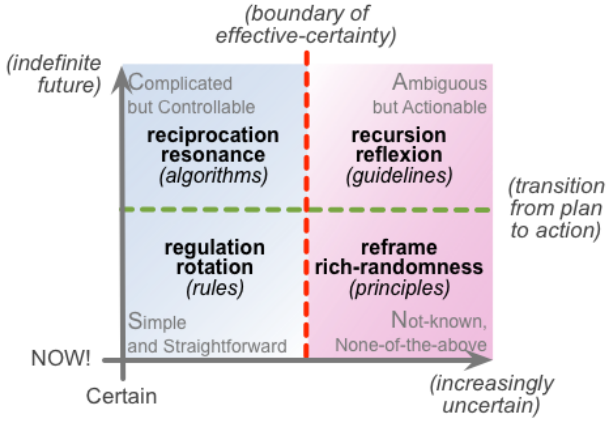
– ***Ambiguous*******

- *recursion* - patterns may be 'self-similar' at every level of the context
- *reflexion* - intimations of the nature of the whole may be seen in any part of or view into that whole

– ***Not-known*******

- *reframe* - switch between nominally-incompatible views and overlays, to trigger 'unexpected' insights
- *rich-randomness* - use tactics from improv and such-like to support 'structured serendipity'

Or, in visual form:



Which, to illustrate, is *recursively* just another *reframe* for SCAN - exactly as described for the 'Ambiguous' and 'Not-known' domains above.

Finally, some **extensions to SCAN**, created by Kai Schlüter[120] (who also developed the 'SCAN as decision-dartboard' technique mentioned earlier). Each of these extensions is actually a separate frame or checklist that we would use *in parallel* with SCAN, to trigger further insights - again as per the *reframe* method for the SCAN 'Not-known' domain. And as we'll see, each extension is labelled via a meaningful acronym for the respective context: EPIC SCAN, WISE SCAN and PACE SCAN.

EPIC SCAN[121] is focussed on the epic problems of unneeded-complexity as "the root-cause of too-high costs":

[120] https://twitter.com/chbrain

[121] http://socialea.chickenbrain.de/2013/02/dreamtime-epic-scan.html

- ***E**mergent Complexity* - consequence of many small and unrelated decisions.
- ***P**erverse Complexity* - consequence of clumsy attempts to reduce complexity.
- ***I**rreducible Complexity* - consequence of the real complexity of the demand environment.
- ***C**ontrived Complexity* - consequence of deliberate creation to benefit some stakeholders.

WISE SCAN[122] is about the factors that help us to be wise in guiding business-transformations:

- ***W**orth* - the future capability must be worthwhile to trigger a transformation.
- ***I**nformed* - the future capability must contain all the relevant information, as much as needed, containing the necessary facts.
- ***S**imple* - the future capability must be the most simple solution which fits the purpose.
- ***E**nvironment* - the future capability must be embedded in the greater context.

PACE SCAN[123] explores the drivers behind the pace of work, and critical factors in business-transformation:

- ***P**eople* - to transform people will make the difference between success and failure.

[122] http://socialea.chickenbrain.de/2013/02/dreamtime-wise-scan.html
[123] http://socialea.chickenbrain.de/2013/02/dreamtime-pace-scan.html

- ***A**daption* - to transform the complete solution must be adapted to reach fit to purpose.
- ***C**ommunication* - to transform communication is key to secure that the targets will be reached.
- ***E**mphatic* - to transform it is required to sense also the unspoken which enables to deliver and help that people and solution form a perfect fit-to-purpose environment.

I really love it when I see people taking up ideas that I'd developed, and then building on those ideas to do something even better with them. What Kai has done here with SCAN has been amongst the best I've seen, and especially well-suited to his very fast-paced business-driven architectures. Many thanks indeed, Kai! :-)

Hope this is useful, anyway.

Source (Tetradian weblog)

- *Date*: 2013/10/07
- *URL*: scan-some-recent-notes[124]
- *Comments*: 1
- *Categories*: Complexity / Structure, Enterprise architecture
- *Tags*: decision-making, Enterprise architecture, SCAN, sense-making, sensemaking

[124] http://weblog.tetradian.com/scan-some-recent-notes

Problem-space, solution-space and SCAN

Uh-oh, Ric Phillips[125] is back…

And I *do* mean that as a compliment! :-) Too often, when people 'get it wrong', it's because they haven't bothered to read the description, or to think, or both. But when someone like Ric - a real master of simplicity in enterprise-architecture[126] - 'gets it wrong', it's much more likely to be *my* fault than his: that *I* haven't explained it well enough. Which means that, yeah, I *do* need to have a quick go at this one, to see if I can explain it any better.

Here was Ric's comment[127] to my previous post 'Can complex systems be architected?[128]', talking about the SCAN[129] framework and its depiction of complexity:

> What is the thinking behind the selection of

[125] https://twitter.com/ricphillips

[126] http://eapatterns.tumblr.com/

[127] http://weblog.tetradian.com/can-complex-systems-be-architected/#comment-6760

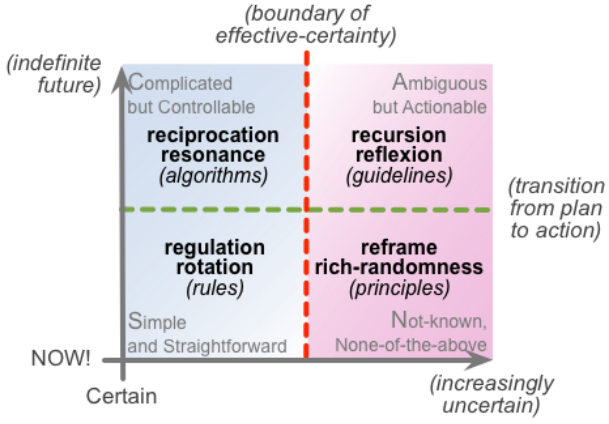
[128] http://weblog.tetradian.com/can-complex-systems-be-architected/

[129] http://weblog.tetradian.com/tag/scan/

> complexity as a function of time over certainty?
>
> It does rather give the impression that levels of disorder are a psychological product. That doesn't scan (no pun intended) with the characterisation of complexity as a state of affairs distinct from complication or confusion.
>
> How does one make an objective assessment of the levels of certainty that pertain to any organisational system (leaving aside for a moment the deeply interesting question you have discussed about where to draw the boundaries of the system itself)?

What??? - well, that'd be my first response, anyway: to me, in at least two places there, he's completely, completely, *completely* missed the point. And yet, wait a minute, hang about, this is one of the sharpest guys I know: if *he's* missed the point, then even at best that point is really *seriously* 'hidden in plain sight' - so well 'hidden in plain sight' that it can't be seen. Oops… *definitely* my fault… sorry…

So… kinda start again, then. Let's start with the diagram in question:



And in essence, what Ric's done, in his first sentence above, is to 'place' complexity 'in' the upper-right region, marked 'Ambiguous but Actionable':

> What is the thinking behind the selection of complexity as a function of time over certainty?

I can understand how people could interpret in this way what I've said before about complexity. And it's even more likely to happen with people who've previously come across the Cynefin framework, which - as I (mis?)understand it, anyway - really *does* 'place' specific types of 'state of affairs' (to use Ric's term) in specific 'domains' of the framework.

But it's not what I mean here, at all - and although the difference can be horribly subtle, it's absolutely crucial

in terms of getting real value from what SCAN can actually show us. If you don't 'get' this point, SCAN falls back to seeming to be just another simplistic two-axis categorisation-framework. Which, yeah, is sort-of useful for some things, but nothing special: there are plenty of other alternatives for that kind of use - Cynefin being one of them, of course.

Instead, there are two keys to getting the best use out of SCAN:

- *it's a metaframework*[130], *not a framework*: its value comes in the way we *use* it - meld it, change it, work *with* it - much more than in what it shows on its nominal surface
- as a metaframework, *it applies to itself* - for example, we can apply any or all of those principles listed in the diagram above *to the framework itself*, and *while we're using it*

Yeah, that kind of recursion can get tricky… but that's where the tool's real power resides. If you *can* get your head around it, it's enormously powerful: it can be really simple - it's just a two-axis framework - yet almost literally as rich as you could possibly want it to be.

The other crucial, *crucial* point here is that it's ***a tool for context-space mapping***. It's not a map of the problem-space, as such; it's not a map of the solution-space, as such;

[130] http://weblog.tetradian.com/metaframeworks-pt4-csm-and-scan/

it's a *framework* - graphic(s) and method *combined* - to help map out the *overall* context-space, both 'problem-space' *and* 'solution-space', *both at the same time*, and dynamically changing over time.

Perhaps even more, it's means to map out how *we* perceive *and interact with* our own understandings of and actions in that context-space. That's *why* I describe it as 'a framework for sensemaking and decision-making': we use it as an active guide within a kind of interactive dance *with* the context, making sense of something, deciding what to do, doing, making sense, deciding, doing, in just about any order or sequence or recursion of making sense, deciding and doing.

Which, in a way, should help to answer Ric's second paragraph:

> It does rather give the impression that levels of disorder are a psychological product. That doesn't scan (no pun intended) with the characterisation of complexity as a state of affairs distinct from complication or confusion.

The answer is, yes, *both* are true. In physics at least, there *is* "complexity as a state of affairs distinct from complication or confusion"; yet when we're dealing with sensemaking and decision-making, then yes, *perceived* "levels of disorder" are indeed "a psychological product". (Another important subtlety here, though, is Cynthia Kurtz[131]'s concept

[131] http://www.storycoloredglasses.com/

of 'unorder' - in effect, 'a state of affairs not amenable to (the usual notions of) order' - which is *not* the same as 'disorder'.

And it also answers the question in Ric's third paragraph:

> How does one make an objective assessment of the levels of certainty that pertain to any organisational system?

– because the short-answer here is that when we're dealing with the type of context-space that we usually work on in most enterprise-architectures and the like - a complex sociotechnical system - there's probably no such thing as "an objective assessment": it's *always* going to be some kind of complex dance between 'objective' and subjective. Which, in turn, also covers the second half of that last paragraph:

> (leaving aside for a moment the deeply interesting question you have discussed about where to draw the boundaries of the system itself)

– because, again, the process of identifying[132] 'the boundaries of the system[133]' is itself as recursive and fractal as anything else in this context.

[132] http://weblog.tetradian.com/whats-the-size-of-an-enterprise/
[133] http://weblog.tetradian.com/boundary-of-identity-vs-control/

(As an aside, this comment[134] just came through from Kai Schlüter[135], which probably illustrates much better than I could the point about 'objective' versus 'subjective' when working on a context-space map:)

> I like this statement: "Almost by definition, 'objective assessment' is somewhere between unlikely and impossible within a sociotechnical complex-system." and yet, it is possible to create a consensus based 'subjective assessment' where all people in the room agree to the statement, which is close enough to objective to be really useful, because the involved people are buying into the assessment. You have written yourself about it in the 'SCAN as decision dartboard[136]' post which I use by (mis)using SCAN to come to a fast assessment on Past, Present and Future to drive decisions in the needed speed (typically time boxed to one hour, plenty challenges to look at in that one hour). I do though never try to look for the perfect truth, just a barely good enough version works.

(See also the last part of the post 'SCAN - some recent

[134] http://weblog.tetradian.com/can-complex-systems-be-architected/#comment-6766

[135] http://twitter.com/chbrain

[136] http://weblog.tetradian.com/scan-as-decision-dartboard/

notes[137]' for more detail and links on how Kai uses SCAN for high-speed sensemaking and decision-making in a very busy multi-project software-development / maintenance context.)

One more point, and then I'd better stop for now. The start-point is a revisit of Ric's first sentence above:

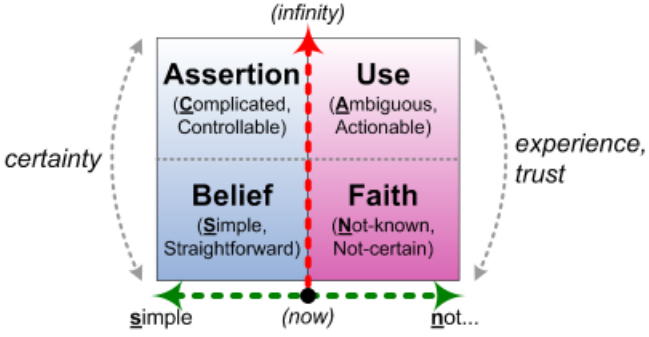
> What is the thinking behind the selection of complexity as a function of time over certainty?

The key here is that the vertical axis in SCAN isn't a metric of time as such: it's *time-available-before-moment-of-action*. In a sense, it's a measure of how much time we have available to us to sit back and make sense of something, before we have to take action based on a decision based on that sensemaking.

When we're further away from the moment-of-action, we have at least *some* time to carry out some kind of analysis and/or experiment. But when we get right close to the action, *there is no time to think* - we have to 'do'. And yet, even right up close to and *in* the moment-of-action, there *is* still a form of sensemaking and decision-making that's going on - but it *isn't* the same as that we used for analysis and experimentation. In that sense, yes, sensemaking and decision-making in relation to complication (such as anal-

[137] http://weblog.tetradian.com/scan-some-recent-notes/

ysis) and VUCA[138]-complexity (such as experimentation) *_is _qualitatively* different from that used at or close to the moment-of-action - in the same way, in turn, that sensemaking and decision-making are *qualitatively* different either side of that 'boundary of effective-certainty' across the horizontal-axis. This is illustrated in one of the early variants of SCAN:

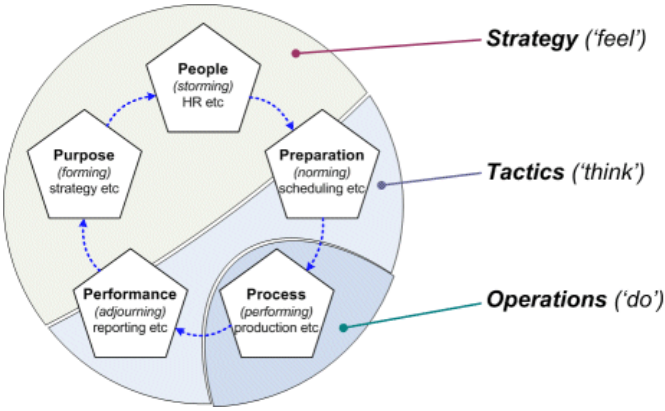


(To be honest, the labels in each domain there probably take more explanation than they're worth, but at least it illustrates the point. :-|)

Perhaps more usefully, we can also see much the same happening in the Five Elements set (another metaframework that's equally fractal and recursive). Each of the 'domains' here has a significantly-different perspective on time: starting from Purpose, they're respectively far-future, people-time, near-future, NOW!, and past. But they also

[138] http://en.wikipedia.org/wiki/Volatility,_uncertainty,_complexity_and_ambiguity

have significantly-different emphases in terms of sense-making and decision-making, which in some ways do also correspond to usage of very different parts of the brain:



If we cross-map that to SCAN, the main parallel is with the 'Preparation'/'Performance' ('think') and 'Process' ('do') domains, which respectively line up above and below the 'transition from plan to action' boundary on the SCAN vertical-axis. They involve *different* thinking-processes, *different* sensemaking, *different* decision-making, and, to a significant extent, *different* forms or modes of 'doing'. In effect, the whole Five Element cycle - starting from 'Purpose' - runs from a potentially-infinite time-before-moment-of-action, all the way down the vertical-axis of the SCAN frame to the moment-of-action at 'Process', very briefly dips beyond that, as 'the past', at the moment between 'Process' and 'Performance', and then all the way back up again, connecting past with intent or initial-

purpose. (We just need to remember, again, that the whole thing is highly fractal…)

Yeah, I know, it's really hard to explain just in words-and-pictures-on-a-page, and I've possibly made it worse in this yet-another-long-ramble - but I hope it clarifies it a *bit*, at least?

Over to you, anyway.

Source (Tetradian weblog)

- *Date*: 2013/12/04
- *URL*: problem-space-solution-space-scan[139]
- *Comments*: 2
- *Categories*: Complexity / Structure, Enterprise architecture
- *Tags*: complexity, enterprise, Enterprise architecture, ric phillips, SCAN

[139] http://weblog.tetradian.com/problem-space-solution-space-scan

Knowable and discoverable

"Profound garbage" - that was how he described my work. Well, I can't complain: at least he described it as 'profound'… :-)

Over the past few days there's been a fair amount that's gone all kinda philosophical and 'meta' on me, and it seems worthwhile to explore a bit about what's been coming up. Don't worry, though: it *does* have real practical applications in the everyday, as you'll see later.

[*An urgent aside*: Some of what follows does, in passing, happen to mention the Cynefin framework. It even references back, in passing, to some themes that came and went in some of the earlier versions of Cynefin. But it's not about Cynefin, at all. *This does not critique Cynefin, at all.* Please don't let's go there again, okay? 'Nuff said… :-( ]

One of the threads in this happened a few days back, when someone - I forget who - posted a piece that contrasted 'knowable' with 'discoverable'. The point there was that 'knowable' applies to the more certain side of the things that we come across - things that we can predict, perhaps, or calculate, or derive via some form of formal-logic - whereas 'discoverable' was more about the things that we

can't predict, but more just come across in passing. (Like this whole thread, I guess? - recursion again… :-))

What whoever-it-was was saying was that 'knowable' could be used as a near-synonym for 'complicated', whereas 'discoverable' could be used as a near-synonym for 'complexity'. Yet for me it didn't quite gel: that kind of 'discovering' seemed to me to fit more closely with what's sometimes referred to as 'creative-chaos', the Not-known space where new ideas arise. But there was no real anchor for me to take that idea any further, so - as usual - I just kinda let it sit, and quietly brew away by itself.

Then yesterday, unbeknownst to me, a colleague posted a Tweet that was, yes, arguably critical about Cynefin, and included a pointer to my two-and-some-years-ago post 'Comparing SCAN and Cynefin[140]'. One of the responses that came up from that included that comment about 'profound garbage' - though the first I heard of any of this was when all of these Tweets starting appearing in my Twitterstream, apparently addressed to me. Ouch…

As I said, I'm very careful these days to try to keep out of any discussion on the merits or otherwise of Cynefin: it really isn't my business, and I prefer to keep it that way. What was *much* more interesting was the reason why that critic had dismissed my work in that way: it was, he said, because he was "looking for coherence". At that point I *did* reply, to ask what he meant by the term 'coherence', what he understood it to be. Carefully-polite, to which I received

[140] http://weblog.tetradian.com/comparing-scan-and-cynefin/

an equally-polite response, in which he pointed to the work of H. H. Joachim[141], "generally credited with the definitive formulation of the coherence theory of truth[142]".

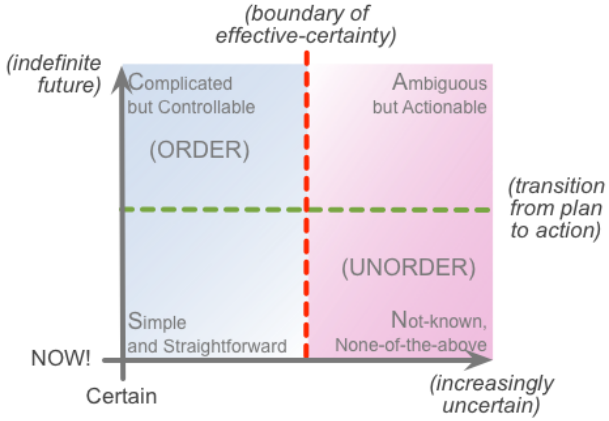
Okay, fine, yes, does look interesting: so I chased up the respective Wikipedia entries. But I was, uh, profoundly unimpressed: to me it just seemed to be one of those more-unfortunate examples of circular-reasoning, with 'truth' defined as coherency and self-consistency, and coherency in turn defined as 'truth'. Presumably I was missing something? Either way, I sent a reply saying that I took a rather different view of 'truth' and coherency, aligned more with Paul Feyerabend[143]'s *Against Method*[144] - in other words, that's there's a kind of "anything goes" involved in the story, that 'truth' is, in essence, more an artificial construct of its own time and context, and that apparent attributes such as coherency or self-consistency might well prove in practice to be somewhat misleading. And there we left it: no big deal for either of us, no further argument, we each went our separate ways.

Yet this morning, it all kinda came together in one of those classic in-the-shower insights: ***coherency, as purported-'truth', is fundamentally inappropriate for complexity***. Coherency is not something that we should expect in true-complexity: or, to frame it the other way round - using Cynefin terminology here - if it's fully self-consistent, it's

[141] http://en.wikipedia.org/wiki/Harold_Joachim
[142] http://en.wikipedia.org/wiki/Coherence_theory_of_truth
[143] http://en.wikipedia.org/wiki/Paul_Feyerabend
[144] http://en.wikipedia.org/wiki/Against_Method

not Complexity, it's merely Complicated. Perhaps more to the point, trying to assert coherency as a 'desired-attribute' for something that incorporates inherent-ambiguities and uncertainties - such as a wicked-problem or a pattern - could well be a very dangerous tactic: trying to apply Complicated-domain methods to such contexts is *not* a wise move…

Cynefin has some useful things to say about this, but as I said above, I'll keep out of that: it's someone else's model, not mine, after all, so best I don't go there. What I'll do instead is link it to SCAN[145]:

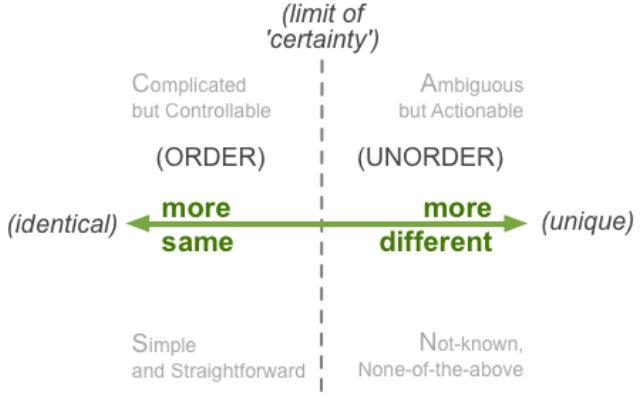


The key to SCAN isn't the domain-names or any of the other labels: don't get hung-up on those, they're literally just labels, nothing more than that. (We could call them all

[145] http://weblog.tetradian.com/tag/scan/

FRED, if you prefer - in homage to Rowland Emmet[146] and his Fantastically Rapid Electronic Device. :-)) Instead, as I described in the post 'SCAN - some recent notes[147]', what matters is those two axes.

The 'horizontal' axis is about, well, a variety of themes such as ***certainty versus uncertainty***, predictability versus unpredictability, sameness versus uniqueness, order versus unorder, or, in this case above, coherence versus non-coherence - they're all similar enough (and, often related-enough) that you can pick any of them and it'll come out much the same. Place most-certain or most-predictable or most-same or whatever over on the left-hand side; its opposite extends out to the right, towards infinity:



[146] http://en.wikipedia.org/wiki/Rowland_Emmett
[147] http://weblog.tetradian.com/2013/10/07/scan-some-recent-notes/

It's rare in most real-world situations that we'll ever get absolute certainty, absolute sameness, absolute predictability. Instead, what we aim for is certain-*enough*, predictable-*enough*, and so on, for the requirement at hand. And there's a real boundary - sometimes sharp-cut, more often somewhat-blurry - between certain-enough and not-certain-enough. That's what I've indicated above as the 'limit of 'certainty'', or 'boundary of effective-certainty'.

The point here is that the tactics and techniques we need to use are often radically-different either side of that boundary: for example, certainty-dependent logic-based methods that aim towards a single 'solution' *really* don't work with inherently-uncertain and continually-morphing wicked-problems. Which, if we go back to that initial comment about ***'knowable' versus 'discoverable'***, does kinda fit here: 'knowable' on the left, 'discoverable' on the right.

But which… I dunno… still isn't quite right yet? Not quite enough? So, keep going…

Which brings us to the other axis in SCAN, the 'vertical' axis. This is about the amount of time we still have available to us for sensemaking and decision-making before we *must* take action. We place 'NOW!' at (and as) the baseline, with time-available stretching upwards, again potentially to infinity. (For those of us who procrastinate a lot - I wouldn't be looking at myself here, now would I? :-(- the time-before-decision too often really *does* extend to infinity…)

And once again there's a very real boundary here - often

easily identifiable in humans, if perhaps less so with machines and IT-systems - where, as we get closer towards the 'NOW!', the techniques and tactics change: decisions *for or _about action tend to be distinctly different to decisions _within* action, at or close to the moment-of-action itself.

We might describe this difference as ***'plan' versus 'action'***, or 'theory' versus 'practice' - a point nicely illustrated by the old joke that "In theory there's no difference between theory and practice; in practice, there is".

The effective 'position-in-time' of that boundary is highly contextual, often dynamic and, again, in some cases also somewhat blurry, but it's real all right: there *is* a real difference between what we do in planning, versus what

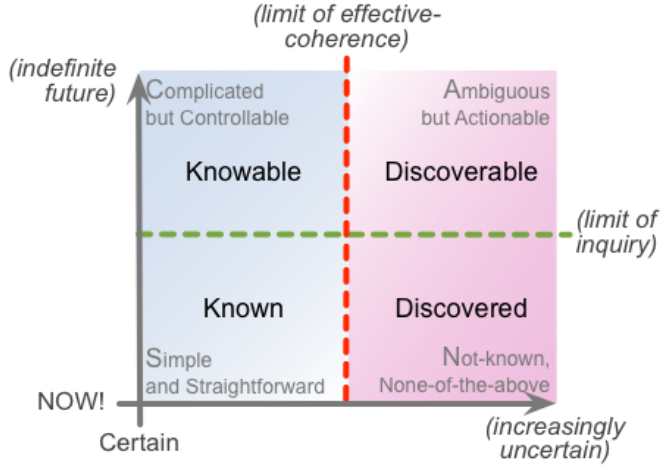
we do to adapt that plan to what's *really* happening right-here-right-now.

Yet if that's so, how do we describe that difference? What came up the other day was a hint from the Cynefin framework, or more specifically out of its past. These days, two of its domains are labelled 'Simple' and 'Complicated', which I've likewise used for the two 'left-side' domains in SCAN.

(In part this is intentional, for compatibility-reasons, since in my interpretation of what Cynefin was/is aiming to do, the two frameworks are fairly similar here. To me at least, they diverge a lot over on 'the other side': but again, I'm not talking about Cynefin here - just that one fact about the domain-labels.)

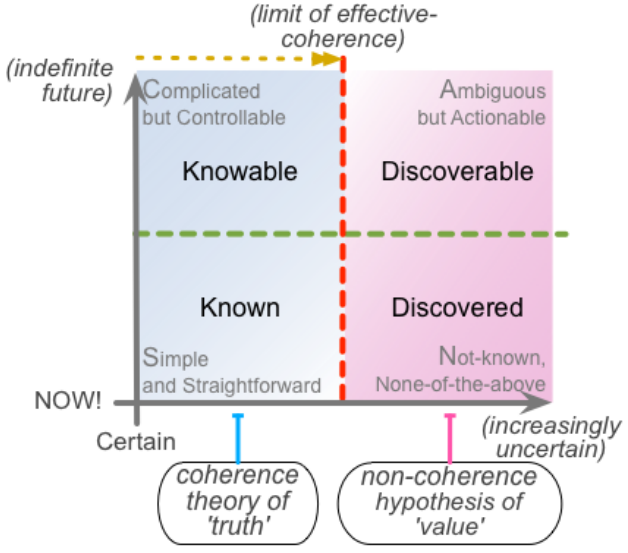
Yet if we go further back in Cynefin's history, back to the days when it was focused more on knowledge-management concerns rather than complexity in general, those two domains were respectively labelled 'Known' (Simple) and 'Knowable' (Complicated).

Which, in turn, suggests another way to look at that 'knowable versus discoverable' dichotomy mentioned earlier - and that this time *does* resolve my qualms about it. Because what it suggests is yet another SCAN crossmap:



I've used the 'limit of effective coherence' label on the horizontal-axis boundary, to link to back to the other half of the conversation that started this thread - but I could of course use any other equivalent, such as the original of 'boundary of effective-certainty'. It really *is* all 'much of a muchness' here.

Anyway, let's look at this in a bit more detail - horizontal-axis first:



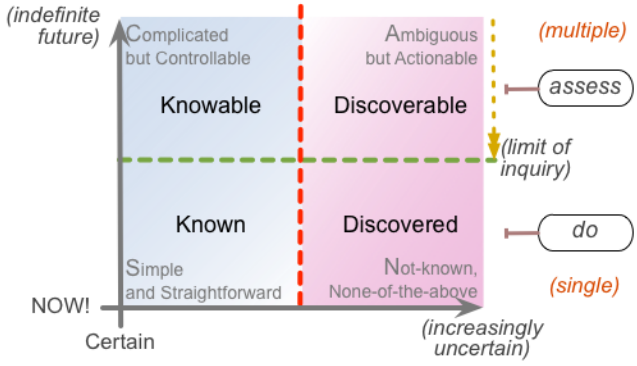
Going back to that initial conversation, about coherence and 'truth', in essence it's about certainty again - things that we can 'know for certain'. There are things that are *somewhat*-uncertain, and that we can *sort-of* link together in a coherent way, even though the terminology or values or whatever kind of differ in some ways from each other. There are lots of real-world examples of that, even in the sciences: our best current theories of cosmology still give a significantly different date than our current best theories of star-formation, for example. But we still can get enough coherence between the various theories to be able to send a space-probe to exact positions in our own solar-system: in other words, 'good-enough theory' - or, in effect, the 'left-

side' of that 'limit of effective-coherence'.

To get the best results, we might have to do a bit of tweaking, working 'inside the box' - the kind of things that analysts do. And whatever results we get here, we can *know* with reasonable certainty that we'll get the same results in future if we do the same things in the same way. More or less, anyway.

Over on the other side of that boundary, we can't *know* in the same way: instead, we have to kinda *discover* it. For example, we can't *solve* a wicked-problem - reduce it to something 'known' - but we can dynamically and repeatedly *re-solve* it, in accordance with its own strange dance: and the nature of that dance is what we can discover, by careful observation and in-person engagement and the like. The difference between theory and hypothesis, in fact: whilst we might hope that a theory would remain certain, the only thing we can be certain of with an hypothesis is that it's likely to change!

Moving to the other direction - the vertical-axis:

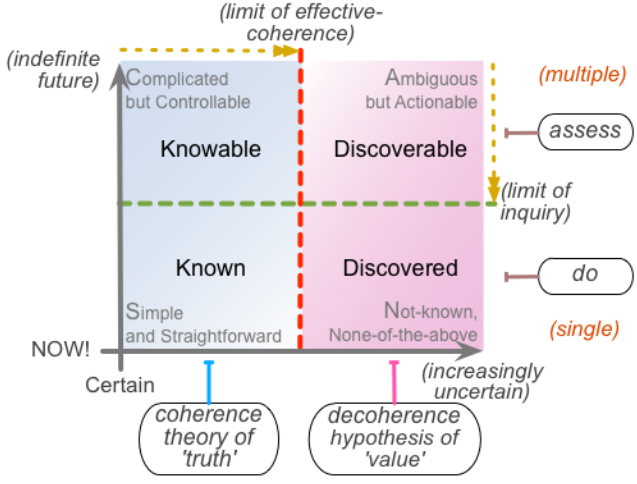


In this case it's probably easiest look at this by coming *downward*, rather than 'upward' from the 'NOW!"-base. Within that *'knowable and/or discoverable'* space - the analysis / experimentation loop - we can assess multiple options and multiple possibilities. As we get closer towards the 'NOW!'-point, though, the options get squeezed, until we finally hit a limit for inquiry: we have to take action. Then *within* the action, the 'do'-space, we do still have options and possibilities - yet ultimately only one at a time, *at* the exact moment-of-action.

Overall, this gives us four distinct domains: Known, Knowable, Discoverable, Discovered (or Discover_ing_ - active, in contrast to the often somewhat-passive interactions with the purportedly-preordained Known). Note, though, that the boundaries between the domains are, again, often highly dynamic and contextual; and the whole thing is itself often deeply recursive, looping through each of the domains at wildly-varying speeds, as shown in the 'SCAN-

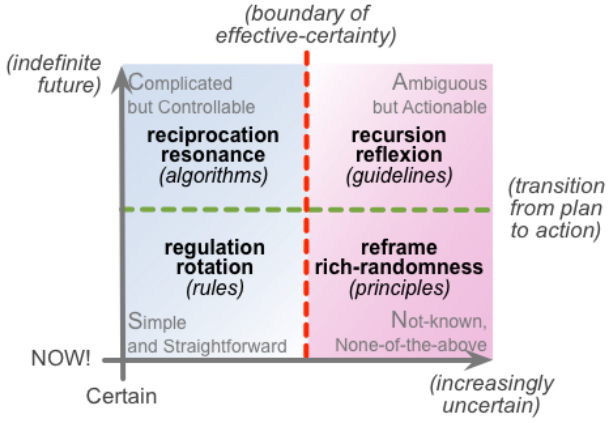
ning the toaster[148]' post. When we *do* have time to explore, we can explore the limits - theoretical and practical - of the Knowable and the Discoverable. But when we're tight on time, we'd better stick to what is Known - or face the real and often very-personal challenges of Discovering, and work with what it is that we've Discovered in that process.

We can summarise the whole thing visually as follows:



And, using one of the other diagrams from the 'SCAN - some recent notes' post, we can summarise the typical types of tactics that come into play dynamically within each of those domains:

[148] http://weblog.tetradian.com/scanning-the-toaster/



That's about it for now, really.

Practical implications

The main point here is that - as with SCAN in general - this forms a very quick, practical checklist: ***Known, Knowable, Discoverable, Discovered*** (or Discovering).

Anything on the Known / Knowable side is going to give us what is effectively 'doing the same thing leads to the same results'. We might need to tweak it a bit to do that, and iron out a few minor uncertainties, but it'll get there. What it *won't* do - and for the most part isn't intended to do - is give us anything different, anything new, anything that we don't already know or that isn't directly derivable from what we already.

Anything on the Discovered / Discoverable side is going to be, well, always a bit messy and uncertain. Kinda like the World Wide Web, always somewhat in flux, and so well described as "always a little bit broken". It's the only place where we'll find anything new: so if we *need* something new, something different, we must face what face up to that uncertainty. Another of the key points from that is that once we cross over to the right-side of that boundary, *we cannot guarantee results*. High-probability does *not* mean "is going to happen"; low-probability does *not* mean "won't happen"; 'uncertain' *means* 'uncertain', not "is wrong because it isn't certain". Above all, *don't* try to 'control' it, because attempting to do so will *always* make things worse.

Anything on the Knowable / Discoverable side of the 'plan / action' boundary means that we still have time to do something, to analyse a bit more, experiment a bit more, build a sort-of certainty in the form of rules - for the Known side - or guiding-principles - for the Discover side. The catch is that the longer we stay there, the less we'll actually get *done* - because that 'doing' only happens when we get right down to the 'NOW!', the moment-of-action.

Anything on the Known / Discover side of the 'plan / action' boundary allows us to get things done - but only because we're not still trying to plan. 'Decision' literally means 'to cut away': and to get things done we have to pare everything right down to the bone, right down to *one* option, right-here-right-now - *one* choice which, at

the moment-of-action, is actually no *choice* at all. Which tells us that if we ask people to make choices *at* the moment-of-action, we're actually forcing them *away* from the moment-of-action - which, by definition, slows things down.

So, to put it at its simplest, *know where you are* at all times within this context-space. Know the 'rules' that apply here, the principles that apply here; watch the recursions as the Known, the Knowable, the Discoverable, the Discovering, all loop and weave through each other, often seemingly all in the same moment. And practice at it, and practice at it, until you don't even notice it at all - yet it's still always there.

Interesting, I hope? Over to you for comments, anyway.

Source (Tetradian weblog)

- *Date*: 2014/01/22
- *URL*: knowable-and-discoverable[149]
- *Comments*: (none)
- *Categories*: Complexity / Structure
- *Tags*: chaos, complexity, decision-making, effectiveness, SCAN, sense-making

[149] http://weblog.tetradian.com/knowable-and-discoverable

SCANning the toaster

How about a nice, simple, first-hand everyday example of SCAN[150] in action? :-)

Here's the context: toasting my mid-morning snack - that old English delicacy, a hot-cross bun[151].

Except that the toaster didn't work.

So here's the toaster, with the two halves of my hot-cross bun still sitting in the slots:

And here's what happened (or didn't)...

1. I put the bun-halves into the toaster, push down the lever, watch the timer-lights (the five LEDs on the

[150] http://weblog.tetradian.com/tag/scan/
[151] http://en.wikipedia.org/wiki/Hot_Cross_Bun

right-side of the control-panel) count down until the buns pop up.

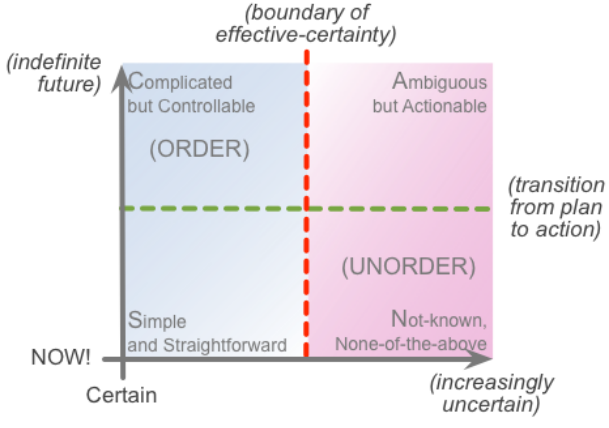
2. I pick the buns out of the slots, drop them on the plate, get ready to spread butter on them, and realise that they're not toasted. Not even hot. Hmm… must have missed something?
3. I check the plug: yep, that's in. In fact, the timer-setting (the red '5' in the control-panel display) has been on the whole time, so it must have been plugged-in anyway.
4. I put the buns back in the slots; push down the lever, checking carefully that it *is* all the way down; check that the timer-lights *are* on, and counting down correctly, until the buns pop up again at the right time.
5. Except they're still cold. I park the buns on the plate, and stop to think. What's gone wrong?
6. Without putting the buns back in, I push the lever down again. The timer-lights come on, doing the timing correctly, but the heater-strips inside the toaster don't glow. The toaster's working overall, but the part that actually does the toasting isn't.
7. Hypothesis: maybe there's been a mains-spike, and its microcontroller needs a reboot?
8. Pull the mains-plug out to force it to do a reboot: yep, the control-lights come on okay.
9. Test it again, pushing down the lever. Nope, no joy: heating-strips still aren't glowing, so no heat.

10. Hypothesis: there's a large breadcrumb shorting it somewhere.
11. Visual inspection: nope, can't see anything.
12. Rattle it about a bit anyway to shake loose any crumbs, then empty the crumb-tray. (Yes, I did pull out the mains-plug first - don't panic!)
13. Second visual inspection: nope - still can't see anything.
14. Test it again, pushing down the lever.
15. Still no joy: control-panel says its working, heater-strips show it isn't.
16. Get hold of a brighter light and a wooden skewer, to do a more careful inspection, with a bit of additional prodding. Nope, can't find anything at all.
17. Quick test again, more in hope than anything else. Still isn't working.
18. Hypothesis: there's a wire come loose somewhere?
19. No, stop - just stop. I'm not equipped for this - I don't have the right test-gear or anything.
20. And I'm running out of time, too. Forget it.
21. Okay, it's going to be *cold* cross bun, then. An unwanted culinary experience…
22. Wait a moment, what about the little grill-oven, won't that work as a toaster as well? Even if only one side at a time?
23. Check its controls: hey, it *will* do both sides at once!
24. How long will it need? Three minutes for a first try, perhaps?

25. Put the bun-halves on the slide-out tray, close the lid, set the timer, watch and wait until the timer rings.
26. Properly toasted? No, not quite…
27. Okay, give it another couple of minutes, I'd guess.
28. Set the timer, start it going, watch and wait again.
29. Done?
30. Yes, done. Hooray!

All terribly trivial, I know. But even at this level, it's a really good illustration of how much we jump around between sensemaking, decision-making and action. In other words, right into SCAN territory.

So let's remap the whole thing onto SCAN, and see what it shows us about how that jumping-around actually works. First, here's the SCAN frame:

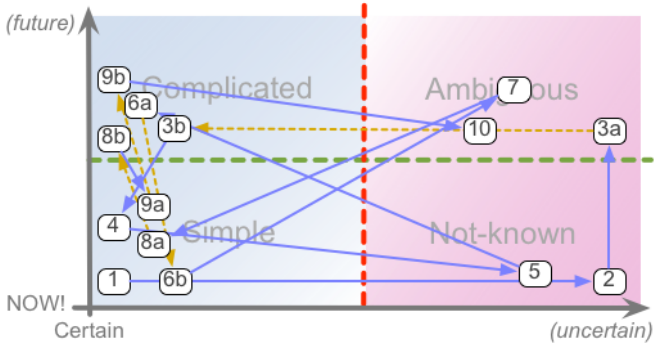


And here's the remap, step by step:

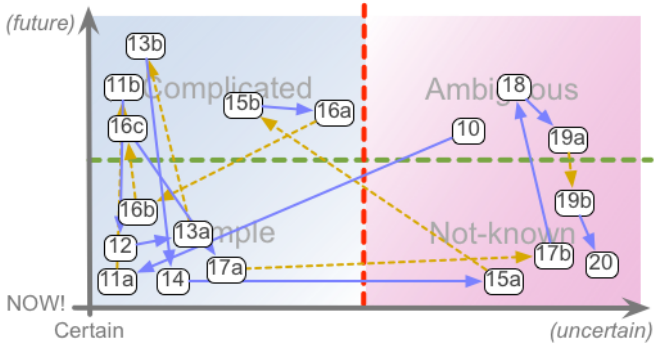
1. Standard procedure for toasting. [*Simple*]
2. Not toasted. [*Not-known*]
3. Experiment, then analyse. [*Ambiguous* to *Complicated*]
4. Repeat standard procedure for toasting, with explicit confirmation of each step. [*Simple*, slightly back from real-time]
5. Still not working. [*Not-known*]
6. Repeat the standard procedure [*Simple*] followed by analysis [*Complicated*]
7. Hypothesis: reboot required? [*Ambiguous*]
8. Do standard procedure for reboot [*Simple*] and analysis [*Complicated*]
9. Repeat the standard procedure for toasting [*Simple*] and analysis [*Complicated*]
10. Hypothesis: breadcrumb short-circuit? [*Ambiguous*]
11. Standard procedure for setup of visual inspection [*Simple*] followed by analysis [*Complicated*]
12. Standard procedure to clear crumbs [*Simple*]
13. Repeat procedure for visual inspection [*Simple*] and analysis [*Complicated*]
14. Repeat standard procedure for toasting [*Simple*]
15. Still no joy [*Not-known*] so analyse results [Complicated]
16. Extend test-algorithm [*Complicated*], apply modified procedure [*Simple*] and analyse results [*Complicated*]

17. Repeat standard procedure for toasting [*Simple*] without success [*Not-known*]
18. Hypothesis: loose wire? [*Ambiguous*]
19. Self-doubt [*Ambiguous* to *Not-known*]
20. Abandon the investigation [*Not-known*]
21. Standard procedure for 'Disappointment'… [*Simple*]
22. New idea [*Not-known*] for new option [*Ambiguous*]
23. Standard procedure to list equipment-capabilities [*Simple*] and compare with requirements [*Complicated*]
24. Guesstimate for timing [*Ambiguous*]
25. Modified standard procedure for toasting, adapted for grill-oven [*Simple* but with some uncertainty]
26. Review outcome [*Not-known* to *Ambiguous*]
27. Analyse results of review [*Complicated*, with some uncertainty]
28. Repeat modified standard procedure for toasting, with amendments [*Simple*, with some uncertainty]
29. Done? [*Not-known* to *Ambiguous* to *Complicated*]
30. Success and completion [*Simple*]

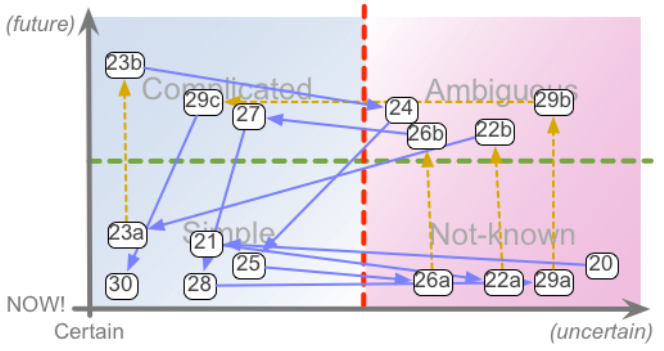
Or, in visual form - steps 1-10:



Steps 10-20:



And steps 20-30:



Did I really do all of that, just to get a toasted tea-cake? Yikes… :-)

Practical application

Again, it's a trivial example - and intentionally so, of course. Rather, the real value here is more if we take what we've just seen above - the looping-around and constant jumping from one mode to another, much as in an indisciplined version of John Boyd's OODA[152] - and instead turn the whole thing the other way round: ***provide support for disciplined sensemaking, decision-making and action***.

(Some of this gets kinda technical again - sorry… But if you interpret and compare it to the real-world example, it probably makes a bit more sense? I hope?)

[152] http://en.wikipedia.org/wiki/OODA_loop

First, if we want known, certain, repeatable results, at high speed, we need to be in the **Simple** domain. Various corollaries arise from this:

– We will need clear, precise, unambiguous, preferably 'Yes/No' instructions that apply directly to the specified context. There can only a small number of these instructions applicable at any one time, and even fewer choices: typically a maximum of half-a-dozen for humans, and precisely one instruction or choice-point at a time for automated systems.

– Because there is (almost) no time to think or calculate when within the Simple domain, those instructions will need to be developed and tested in another domain - usually but not always the Complicated domain.

– The Simple domain delivers fast, and reliably, *only* when the assumptions behind its instructions *do* match the real-world context. When those assumptions do *not* match up with the real-world, the action will 'fail' - in other words, deliver 'unexpected results', or, in SCAN terms, an inherent and enforced transit into the Not-known domain. Repeating the same actions after failure will usually make things worse. Action in the Simple domain therefore *needs sensors and/or other mechanisms to identify when its assumptions no longer apply* - in other words, when the context has moved 'outside the box' of the predefined assumptions.

– When a risk of failure is detected at run-time, the Simple domain can sometimes slow down - move away somewhat from real-time - so as to incorporate what's usually a

fairly rudimentary amount of the Complicated-domain's methods of analysis, and thereby attempt to avoid full failure. (In SCAN terms, the action moves to the right - less-certain - and upward - less-rapid - yet still remains 'within' the effective bounds of the Simple domain.) In effect, there is a trade-off here between uncertainty, reliability and speed: as uncertainty increases, either speed or reliability will need increasingly to be sacrificed.

When we don't know what's happened and we're running at full speed, we are, by definition, in the **Not-known** domain. We can also *choose* to move there, "in order to remember something we never knew". Various corollaries arise from this:

– *Everything* that happens here is in some way unknown, literally 'out of control' or 'outside of the box' - or, arguably no 'box' at all. Out towards the extremes of this domain, there is no such thing as reliability or predictability, because there is no identifiable link between anything and anything else. Machines that rely on repeatability will usually fall into a 'chaotic' state here, and will typically need to be pulled out of that state by external means such as a forced reboot. The equivalent in humans is often described as 'panic' - a literal expression of the Greek root 'pan-', as 'the everything', all at once.

– The Not-known domain is - by definition - inherently unpredictable; and whilst everything does happen here in real-time, time itself can pass in an unpredictable way. Where some kind of known action - 'productivity' - *is*

required, yet transits into Not-known are probable or possible, *'hooks' should be provided to make it easier to return to the Simple domain* and minimise 'wasted time' in the Not-known. An emergency-action checklist[153] is a classic example of such a 'hook' - potentially reducing uncertainty sufficiently to return towards the Simple domain.

– More broadly, principles and patterns can provide 'seeds' around which some form of meaning can coalesce, or provide some of reference-anchor or direction. These are often especially important where it's desirable to remain 'in' the Not-known domain - such as for ideastorming, or to encourage serendipity - or for contexts that depend on unpredictability, such as improvisation in theatre or music.

– Because there is (almost) no time to assess within the Not-known domain, those principles and patterns will need to be developed and tested in another domain - usually in the Ambiguous domain, for principles, or in a combination of Ambiguous and Complicated for action-checklists.

– For many purposes, such as in idea-generation for experimentation, it can be useful or necessary to slow down - to 'drop out of the chaos' somewhat - so as to enable capture of the imagery, ideas and experiences. We might likewise need to constrain uncertainty somewhat. (In SCAN terms, the action moves to the left - less-uncertain - and upward - less-rapid, yet still remains 'within' the effective bounds of the Not-known domain.) In effect, there is a trade-off here between comprehensibility, possibility and speed:

[153] http://weblog.tetradian.com/principles-and-checklists/

as the requirement for comprehensibility or applicability increases, either possibility or speed will need increasingly to be sacrificed.

In the same way that both Simple and Not-known are a matched-pair at the point-of-action, Complicated and Ambiguous have the same kind of matched-pair relationship *away* from the point-of-action, respectively supporting analysis and experimentation.

For the **Complicated** domain, the core purpose is to develop 'rules' and suchlike that are compact enough to be usable at run-time in the Simple domain. Various points arise from this:

– Note that, by definition, its work takes place at some distance from the action: *the Complicated domain does not act directly on the real-world*. This can cause the misleading belief that "there's always time for more analysis": the reality is that such 'analysis-paralysis' - avoidance of commitment to action - can itself cause failure.

– The focus within the Complicated is on analysis, identifying 'correct' or self-certain answers *within* a specified algorithm. It may have considerable difficulty to 'bootstrap' itself with sufficient *uncertainty* to assess the validity of its own assumptions. (In SCAN terms, moving towards the right of the frame, yet still within the bounds of the Complicated domain.) In general, a transit to the Ambiguous - usually intentional - may be needed for this purpose.

– Unlike the Simple domain, the Complicated domain can

handle *any* required level of 'complicatedness', including delay-effects and feedback-loops, as long as it fits within the core requirements of effective-certainty and effective-predictability. However, the algorithms will be *actionable* only to the extent that they can be partitioned into 'chunks' that *can* be applied in the Simple domain, as per above. Another trade-off applies here: increasing complicatedness in general enforces slower response and greater difficulty in partitioning into actionable 'chunks': increased logic-processor speeds can mask this trade-off somewhat, but by definition can never eliminate it.

For the **Ambiguous** domain, the core purpose is two-fold: to develop patterns and principles that can be actionable in the Not-known domain; and provide some means - typically, hypothesis and experiment - to support development of actionable algorithms in the Complicated domain and subsequently for the Simple domain. Various points arise from this:

– Note that, by definition, its work takes place at some distance from the action: *the Ambiguous domain does not act directly on the real-world.* This can cause the misleading belief that "there's always time for more experiments": the reality is that such avoidance of commitment to action can itself cause failure.

– The focus within the Ambiguous is on experiments to point towards a usable hypothesis. It may have considerable difficulty to reduce the ambiguities and uncertainties enough to make it *usable* as a base for algorithms

- especially for systems which *require* certainty, such as computer-based systems - or even explainable with consistent terminology and suchlike. (In SCAN terms, this implies moving towards the left of the frame, if still within the bounds of the Ambiguous domain.) In general, a transit to the Complicated may be needed for this purpose - if only to test the hypothesis to the point of failure.

– Unlike the Not-known domain - which, by definition, is inherently particular and unique - the Ambiguous domain can handle *any* required level of complexity and generality, as long as the exploration *does* for arbitrary levels of uncertainty and unpredictability. However, the outcomes of the exploration will be *actionable* only to the extent that they can be simplified down into principles or patterns that *can* be applied in the Not-known domain, as per above, and/or packaged into hypotheses that can be tested in the Complicated domain and further adapted for the Simple domain. Another trade-off applies here: increasing need for generality and pattern-identification for re-use enforces slower response and risks a tendency to suppress usable and valid uniqueness.

Overall, as described in the later part of the post 'SCAN - some recent notes[154]', we can summarise the kind of techniques or checks that we'd use in each domain in terms of a simple set of keywords:

– *Simple*

[154] http://weblog.tetradian.com/scan-some-recent-notes/

- *regulation* - reliance on rules and standards
- *rotation* - systematically switch between different elements in a predefined set, such as in a work-instruction or checklist (or in the elements in a framework such as SCAN itself)

– *Complicated*

- *reciprocation* - interactions should balance up, even if only across the whole
- *resonance* - systemic delays and feedback-loops may cause amplification or damping

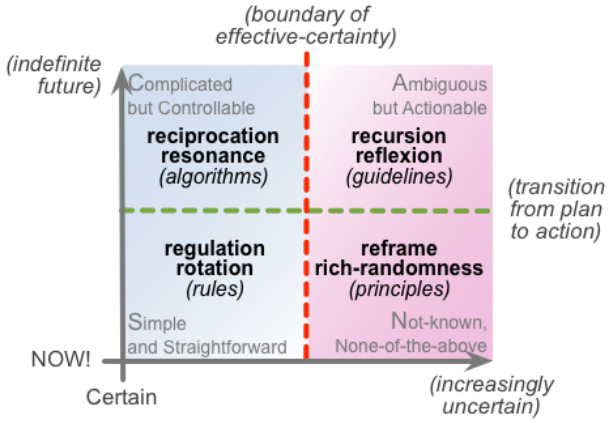
– *Ambiguous*

- *recursion* - patterns may be 'self-similar' at every level of the context
- *reflexion* - intimations of the nature of the whole may be seen in any part of or view into that whole

– *Not-known*

- *reframe* - switch between nominally-incompatible views and overlays, to trigger 'unexpected' insights
- *rich-randomness* - use tactics from improv and such-like to support 'structured serendipity'

Which we could again summarise in visual form as follows:



Probably the main point I'd like to emphasise this time, though, is how *fast* all of this switching-around really is. It isn't that we sit in some specific domain for long periods at a stretch: instead, as can be seen in the toaster example above, the whole thing is fractal, recursive, continuously looping through and round and within itself, often so fast that it can be hard to catch what's really going on. The real advantage of something like SCAN is that it provides 'just enough framework' to be able to *see* all of this happening, *as* it happens - and thence start to gain more choice about how we do our sensemaking, decision-making and action.

Anyway, more than enough for now: time to get back to practice, I'd suggest? (Or, in my case right now, to bed. And *without* a butter-laden hot-cross-bun! :-))

Source (Tetradian weblog)

- *Date*: 2014/01/17
- *URL*: scanning-the-toaster[155]
- *Comments*: 2
- *Categories*: Complexity / Structure
- *Tags*: chaos, complexity, SCAN

[155] http://weblog.tetradian.com/scanning-the-toaster