the Tetradian weblogs

SCAN for sense making and decision-making

Making sense in real-world uncertainty

Tom Graves

SCAN for sensemaking and decision-making

Making sense in real-world uncertainty

Tom Graves

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

This version was published on 2022-08-22

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

© 2022 Tom Graves

Contents

SCAN: SAMPLE

This is a sample of the content from the Tetradian ***SCAN Sensemaking and Decision-making*** anthology.

This anthology from the Tetradian weblog covers the **SCAN** model-type and its related uses and modelling methods. SCAN is used to guide sensemaking and decision-making in contexts with varying levels of complexity, uncertainty and uniqueness.

This book is the first of a two-part series, showing the initial SCAN developments and usages up to 2014. The second part in the series, *Updates on SCAN sensemaking: New developments for the SCAN model*, describes further developments and usages from 2014 onwards.

This sample contains around one-tenth of the content from the full anthology. The complete book includes about 35 posts and 55 images from the weblog. Those posts are split into four groups:

- *SCAN: Core and Origins* - describes the underlying concepts, structures and methods for the core SCAN framework.
- *SCAN: Sensemaking* - explores how to use the SCAN frame to support sensemaking.
- *SCAN: Decision-making* - illustrates how to use the SCAN frame to support real-world and real-time decision-making.
- *SCAN: Theory and Practice* - presents the theory that underpins the SCAN framework, and practical examples of SCAN in use.

For further information on enterprise-architectures and more, visit the **Tetradian weblog** at weblog.tetradian.com[1]. The weblog cur-

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

rently includes some 1400 posts and more than a thousand images, and is at present the world's primary source on *whole-enterprise architecture* - methods, principles and practices for architectures that extend beyond IT to the whole enterprise.

For more ebooks and anthologies on enterprise-architecture and more, visit the **Tetradian website on Leanpub** at leanpub.com/u/tetradian[2]. (Each anthology contains around 30-40 posts from the weblog.)

Some books are also available in print format, from all regular book-retailers. For more details, see the 'Books' section on the main **Tetradian website** at tetradian.com/books/[3].

Unless otherwise stated, all text, images and other materials in this anthology are Copyright © Tom Graves / Tetradian 2006-2022.

[2] https://leanpub.com/u/tetradian

[3] http://tetradian.com/books/

“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.

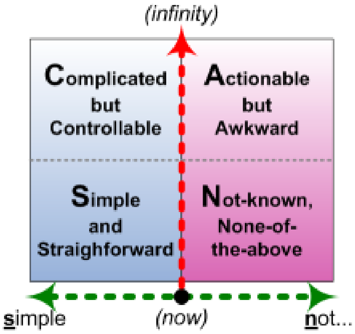
And I think I may have found one, with this:

“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:


(infinity)
Complicated but Controllable
Actionable but Awkward
Simple and Straighforward
Not-known, None-of-the-above
simple
(now)
not...

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 sensemaking 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[4]: it may amuse. Or something. :grin:)

[4] http://www.tomgraves.org/3science

Source (Tetradian weblog)

- *Date*: 2011/11/08
- *URL*: lets-do-a-quick-scan-on-this[5]
- *Comments*: (none)
- *Categories*: Business, Complexity / Structure, Enterprise architecture
- *Tags*: Business, decision-making, disruption, Enterprise architecture, Knowledge, SCAN, sense-making

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

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[6] from Roger Sessions[7], where he asked me for the mathematical basis for those domains and boundaries. I think he was a bit shocked when I said[8] 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[9]'. But it's probably worth having the extra detail, anyway.)

The dimensions of SCAN

There are three distinct dimensions to SCAN:

- *modality*[10] – the extent of perceived 'controllability' versus 'possibility and necessity'
- *available-time* – the amount of time remaining before an action-decision must be made

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

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

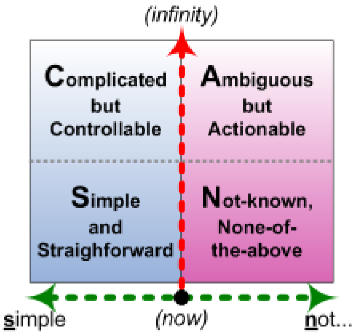
[8] http://weblog.tetradian.com/ensuring-that-the-simple-stays-simple/comment-page-1/#comment-71384

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

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

- *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:


(infinity)
Complicated but Controllable
Ambiguous but Actionable
Simple and Straighforward
Not-known, None-of-the-above
simple
(now)
not...

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[11]'. This layout can be somewhat misleading in that the boundaries between apparent domains are not 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.

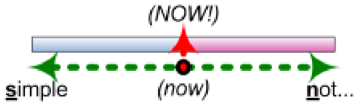
Dimension of modality

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

[11] http://www.tomgraves.org/3science

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'[12]. The definition for Simple 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.

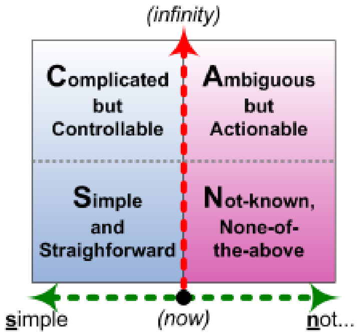
The key point here is that as we gain more time before a decision must be made, we also gain the ability to assess a broader range of options. When the time is tightly focussed, we *must* focus on 'right here, right now', the *specific* point of action, using only what

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

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.*


(infinity)
Complicated but Controllable
Ambiguous but Actionable
Simple and Straighforward
Not-known, None-of-the-above
simple
(now)
not...

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.*

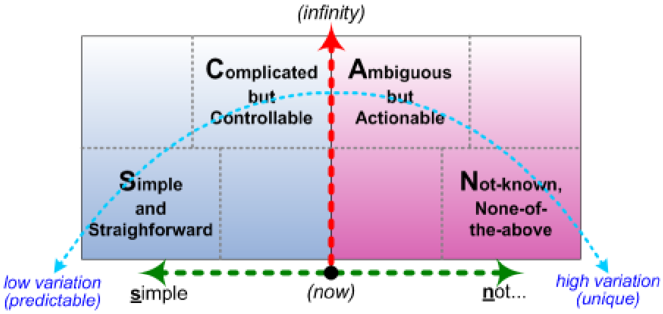
What we *actually* have in sensemaking – and, hence, its derived decision-making – is a myth of abstraction: the belief that abstractions are somehow ‘real’. Something ‘makes sense’ because we *choose* that it should ‘make sense’ in that way: just how much it actually *is* ‘real’ is often an open question…

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:


(infinity)
Complicated but Controllable
Ambiguous but Actionable
Simple and Straighforward
Not-known, None-of-the-above
low variation (predictable)
simple
(now)
not...
high variation (unique)

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 cross-over 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[13]' 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

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

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 'repeatable' 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.

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[14] in enterprise[15]-architecture series[16]', and in other future posts.

Source (Tetradian weblog)

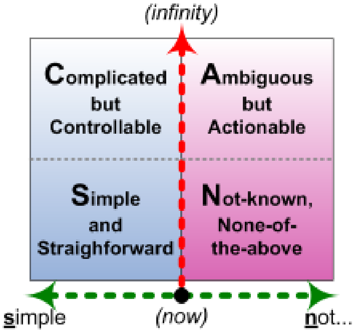
- *Date*: 2011/11/18
- *URL*: domains-dimensions-in-scan[17]
- *Comments*: (none)
- *Categories*: Complexity / Structure, Enterprise architecture, Knowledge
- *Tags*: ambiguous, chaotic, complex, complicated, Enterprise architecture, not-known, SCAN, sense-making, simple

[14] http://weblog.tetradian.com/on-sensemaking-in-ea-1/
[15] http://weblog.tetradian.com/on-sensemaking-in-ea-2/
[16] http://weblog.tetradian.com/on-sensemaking-in-ea-3/
[17] 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:


(infinity)
Complicated but Controllable
Ambiguous but Actionable
Simple and Straighforward
Not-known, None-of-the-above
simple
(now)
not...

SCAN: sensemaking)

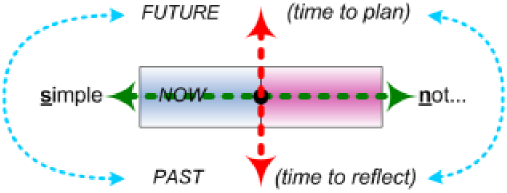
The horizontal green-line axis here represents the decision-type, from a simple true/false choice to a not-so-simple modal choice of possibility and necessity; the vertical red-line axis is the amount of time available before *must* make a choice and take action.

(For more on SCAN and its technical background, see the posts '"Let's do a quick SCAN on this"[18]' and 'Domains and dimensions in SCAN[19]'.)

In a sense, though, that red line of 'available-time' goes *both* sides

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

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 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 everything'. 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'[20] 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[21]'. 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,

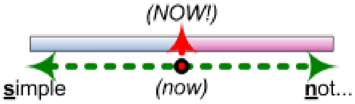
[20] http://weblog.tetradian.com/comparing-scan-and-cynefin/

[21] http://www.youtube.com/watch?v=4oAB83Z1ydE

using this specific question of real-time sensemaking as the ‘target problem’ to keep in focus.

(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[22]' "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

[22] http://www.susanjeffers.com/home/index.cfm

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[23]’ to create checklists on-the-fly with whatever ideas I’ve gleaned from the Not-simple side
- use quick enquiry-techniques such as ‘Five Whys[24]’ 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’)

[23] http://www.projectcheck.org/checklist-for-checklists.html
[24] http://en.wikipedia.org/wiki/5_Whys

– Almost by definition, [N] doesn't seem to have any clear patterns: the only 'patterns' I see myself doing all too often 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[25]
- *Comments*: (none)
- *Categories*: Complexity / Structure, Enterprise architecture, Knowledge
- *Tags*: ambiguous, chaotic, complex, complicated, Enterprise architecture, not-known, SCAN, sense-making, simple

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

SCAN as ‘decision-dartboard’

How to use SCAN[26] as a ‘decision-dartboard’ in work-planning? That was probably the highlight in a great conversation outside the IRM-EAC[27] conference in London earlier this week with Kai Schlüter[28], enterprise-architect at Danish engineering conglomerate Danfoss[29].

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[30] 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*.

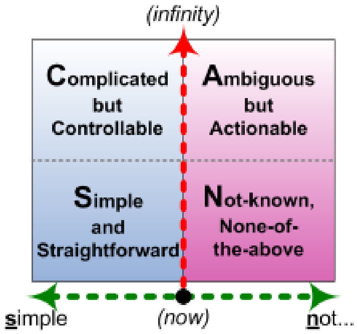
But not always - because it *can’t* always work that way.

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[31]:

[26] http://weblog.tetradian.com/tag/scan/
[27] http://www.irmuk.co.uk/eac2013/
[28] http://twitter.com/chbrain
[29] http://www.danfoss.com/
[30] http://weblog.tetradian.com/at-unicom-ea-2013/
[31] http://weblog.tetradian.com/tag/context-space-mapping/


(infinity)
Complicated but Controllable
Ambiguous but Actionable
Simple and Straightforward
Not-known, None-of-the-above
simple
(now)
not...

We view the context in terms of two dynamic dimensions[32] - 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 back through some[33] of[34] the[35] various[36] posts[37] here[38] on[39] the[40] SCAN[41] framework[42], 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

[32] http://weblog.tetradian.com/a-simpler-scan/
[33] http://weblog.tetradian.com/on-metaframeworks-in-ea/
[34] http://weblog.tetradian.com/methods-mechanics-approaches/
[35] http://weblog.tetradian.com/principles-and-checklists/
[36] http://weblog.tetradian.com/control-complexity-and-chaos/
[37] http://weblog.tetradian.com/best-practices-adapt-then-adopt/
[38] http://weblog.tetradian.com/reframing-entropy-in-business/
[39] http://weblog.tetradian.com/working-with-i-dont-know/
[40] http://weblog.tetradian.com/more-keywords-for-scan/
[41] http://weblog.tetradian.com/control-complex-chaotic/
[42] http://weblog.tetradian.com/scalability-and-uniqueness/

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:

- *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[43]’: SCAN wasn’t *designed* to be used in this way, but it *affords* the possibility of using it that way, in a way that really

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

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[44] 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[45] to understand where we want to go. So very literally if it is WISE to go where we thing we should go.

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

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

3. we use PACE SCAN[46] 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)

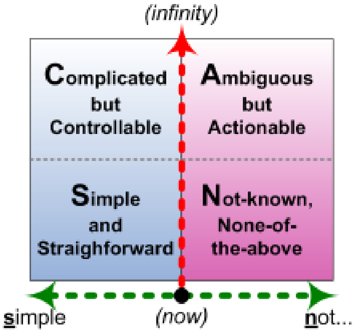
- *Date*: 2013/06/15
- *URL*: scan-as-decision-dartboard[47]
- *Comments*: 2
- *Categories*: Complexity / Structure, Enterprise architecture
- *Tags*: decision-making, Enterprise architecture, kai schluter, SCAN, sense-making

[46] http://socialea.chickenbrain.de/2013/02/dreamtime-pace-scan.html
[47] http://weblog.tetradian.com/scan-as-decision-dartboard

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"[48], in which I introduced the SCAN frame for sensemaking at business-speed:


(infinity)
Complicated but Controllable
Ambiguous but Actionable
Simple and Straighforward
Not-known, None-of-the-above
simple
(now)
not...

SCAN: sensemaking

(The above is the updated core-graphic – see 'SCAN – an Ambiguous correction[49]'.)

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

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

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

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[50] 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.

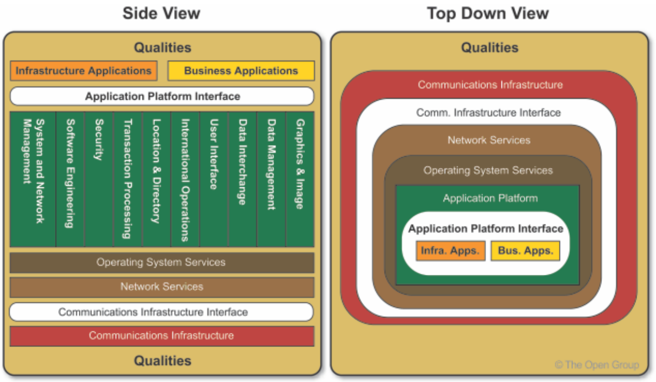
So if we *don't* have the time to explore further, we treat each item just as they are: Complicated gets squeezed down into the enforced 'shall' of Simple, and Ambiguous gently drops back into the acceptance of a less-enforceable 'should', where it may still remain somewhat 'Not-known' right up until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

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



[51]

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[52], 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

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

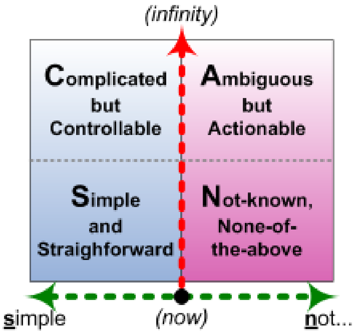
[52] http://pubs.opengroup.org/architecture/togaf9-doc/arch/chap15.html

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-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.


(infinity)
Complicated but Controllable
Ambiguous but Actionable
Simple and Straighforward
Not-known, None-of-the-above
simple
(now)
not...

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[53] 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 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

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

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[54]
- *Comments*: (none)
- *Categories*: Business, Complexity / Structure, Enterprise architecture
- *Tags*: Business, decision-making, disruption, Enterprise architecture, Knowledge, SCAN, sense-making

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