On Defining Agile Systems Engineering

Rick Dove, dove@parshift.com

Abstract: The author was asked to define Agile Systems Engineering, and found it necessary, for context, to first develop some thoughts on what engineering and systems engineering might be. This led to thoughts on differentiating engineering from crafting and arting. Engineering, crafting, and arting are activities carried out by artifact creators with different purposes, but at core all are constrained and enabled by the laws of coherence. This article does not suggest what those laws might be; but does conclude that the definition of agile systems engineering is rooted in what it does, not how it does it.

A definition of Agile Systems Engineering is necessarily context dependent on Systems Engineering, which is in turn context dependent on Engineering. I will approach this perhaps differently than others.

I choose to view engineering fundamentally as an intelligence directed activity that designs artifacts (broadly meant) which exhibit design coherence (key word).

Webster defines coherence as:

1: The quality or state of cohering, such as:

- systematic or logical connection or consistency.
- integration of diverse elements, relationships, or values.

2: The property of being coherent: united as or forming a whole.

Engineering is generally defined relative to a domain. But in all cases it is fundamentally an intelligence directed activity conducted by biological or artificial intelligence sources. It is the activity (designing) and the outcome (coherence) that are defining. Designing a coherent algorithm is an engineering activity. Designing a mathematical proof is an engineering activity. Designing a coherent strategy is an engineering activity. The work of a building architect is an engineering activity. An artificial intelligence designing new molecules which exhibit coherence is an engineering activity.

Engineering, crafting, and arting are activities carried out by artifact creators with purpose. The essential semantic difference between the three is in the purpose of the activity, but not necessarily in the perfect completion of the result. Frank Lloyd Wright engineered the structural coherence of Falling Water, considered one of his crowning achievements; which ironically resonated with its name, as the structural coherence was inadequate to sustain the stresses and had to be subsequently re-engineered years after it was built. Nevertheless, Wright's initial design activity was engineering, and arting, with quite a bit of coherence in both aspects.

Engineering, crafting, and arting are all subject to laws of coherence (whatever they might be fundamentally). In the end, the elements employed in artifact creation must work together in concert to achieve a purpose.

Defining Engineering: designing something coherent that can function sufficiently to satisfy a purpose if and when constructed according to the design. I did not say an *intended* purpose – what is designed may actually fulfill a different purpose than intended, it is nevertheless an engineering activity. The purpose of a system is what it does. Engineering does not carry a requirement of beauty or user appreciation. Its defining requirement is functional coherence. Engineers have engineered things that don't get used, for lots of reasons: something

better was engineered by someone else, or the intended purpose ceased to exist. Nevertheless, an engineering activity occurred. The elements of design are peculiar to the domain of the engineering activity: physical materials, laws of nature in various domains, mathematics, computer instructions, to name a few.

Defining Crafting: The exercise of procedures to construct a functional artifact. It may function as art, and/or as a usable engineered artifact; but there is little engineering in the activity, except that which is exercised by the craftsman to interpret a design intent.

Defining Arting: in the traditional sense arting is the engineering and crafting of something with the purpose of causing a human emotional reaction by an observer or participant. Art doesn't have to have lasting value or lasting existence – performance art, for instance, is art in the moment. Arting can be applied to both engineering and crafting, and is the emotion-generating component of an otherwise rational activity. As to the *art of systems engineering*, a potentially ambiguous phrase. I prefer to apply it to the generated user emotion – a rich topic for needed discussion at another time.

We speak of system science as a missing element in codified systems engineering. The systems science of most relevance are the laws of system coherence. What are the universal laws of coherence, and what are the domain dependent laws of coherence? The concept of system coherence is basic to all engineering, crafting, and arting.

Systems engineering is engineering by the definition above. Systems engineering is fundamentally about total-system coherence. But as it is currently codified, it has a large focus on craft, and some arm waving about art.

Systems engineering distinguishes its domain of engineering in the *systems* context.

Agile systems engineering distinguishes the type of systems engineering with the adjective *agile*. It is called *agile* systems engineering because it exhibits agility – not because it has a practice dogma (like the craft of agile software development). It is an organic complex system (process) motivated by self preservation to evolve suitably in a potentially uncooperative environment. An understanding of what the phrase *agile systems engineering* means is captured at the encompassing conceptual level, not at the procedural or best practice level. This understanding starts with succinct statements of need and intent.

Need: Effective system engineering in the face of uncontrolled change.

Intent: Effective response in a CURVEd systems engineering operational environment: Capricious, Uncertain, Risky, Variable, and Evolving (Dove 2017).

The word *effective* means that a valued result from resource employment is obtained. At one extreme, a project canceled before completion should provide valuable and employable learning and artifacts. At the other extreme, a deployed system should provide sustainable relevance beyond a break even ROI. In the middle, responding to changes in the engineering operational environment should sustain forward progress. The word *should* is used to indicate a necessary objective for an agile systems engineering process: value delivery.

Uncontrolled change encompasses uncontrollable change, but is not limited to that which can't be controlled, just what isn't controlled regardless of reason.

The definition of agile systems engineering is rooted in what it does, not how it does it. The how can be satisfied many ways. Agile systems engineering responds effectively in CURVE environments, operates asynchronously and potentially simultaneously in at least seven life cycle stages, appears to behave according to nine operating principles, and melds target system, development system, and learning system into one interdependent system (Dove 2017).

Enabling and Practicing Systems Engineering Agility

The theme for the 2018 quarter two issue of INSIGHT is Enabling and Practicing Systems Engineering Agility. About eleven articles are wanted that address *agile systems engineering* process experiences that range across mixed-discipline projects, organizational/cultural transformation, open system architecture employment, product line engineering, lessons learned, and other issues and solutions ready for sharing. Contact the author for the call-for-articles guidance, with first draft articles due 30-November-2017.

References

Dove, Rick. 2017. "Agility in Systems Engineering – Findings from Recent Studies." Paradigm Shift International working paper, 15 April. www.parshift.com/s/ASELCM170515-AgilityInSE-Findings.pdf.