33rd Annual INCOSE international symposium

hybrid event

Honolulu, HI, USA
July 15 - 20, 2023

Contrasting and Comparing Agile Systems Engineering and Agile Software Engineering

IS23 Panel
Monday 17-July, 15:30-17:00

Rick Dove dove@parshift.com
Kerry Lunney kerry.lunney@thalesgroup.com.au
Duncan Kemp duncan.kemp735@mod.gov.uk
Robin Yeman robinyeman@gmail.com

Download this file: www.parshift.com/t/p1.pdf

Panel Context

Agile engineering, of any kind, is a principle-based method for designing, building, sustaining, and evolving purpose-fulfilling creations when knowledge is uncertain and operational environments are dynamic.

Principles are abstractions for what needs to be accomplished and why, without constraints or directions on how.

How those abstractions manifest operationally depends upon the engineering context.

Single-domain software engineering is different than multi-domain systems engineering.

While tactical methods necessarily vary among different engineering domains (the how part), strategies for achieving common goals (the what and why parts) are domain independent.

Panel

Panelists will offer their views on selected agility-enabling strategies with considerations for fitting them differently in Agile Systems Engineering and Agile Software Engineering.

Panel Purpose:

- **Mitigate confused thinking that agile software methods are appropriate for agile Systems Engineering.**
- **Show how SE and SW have a different perspective on methods**

90 Minute Plan

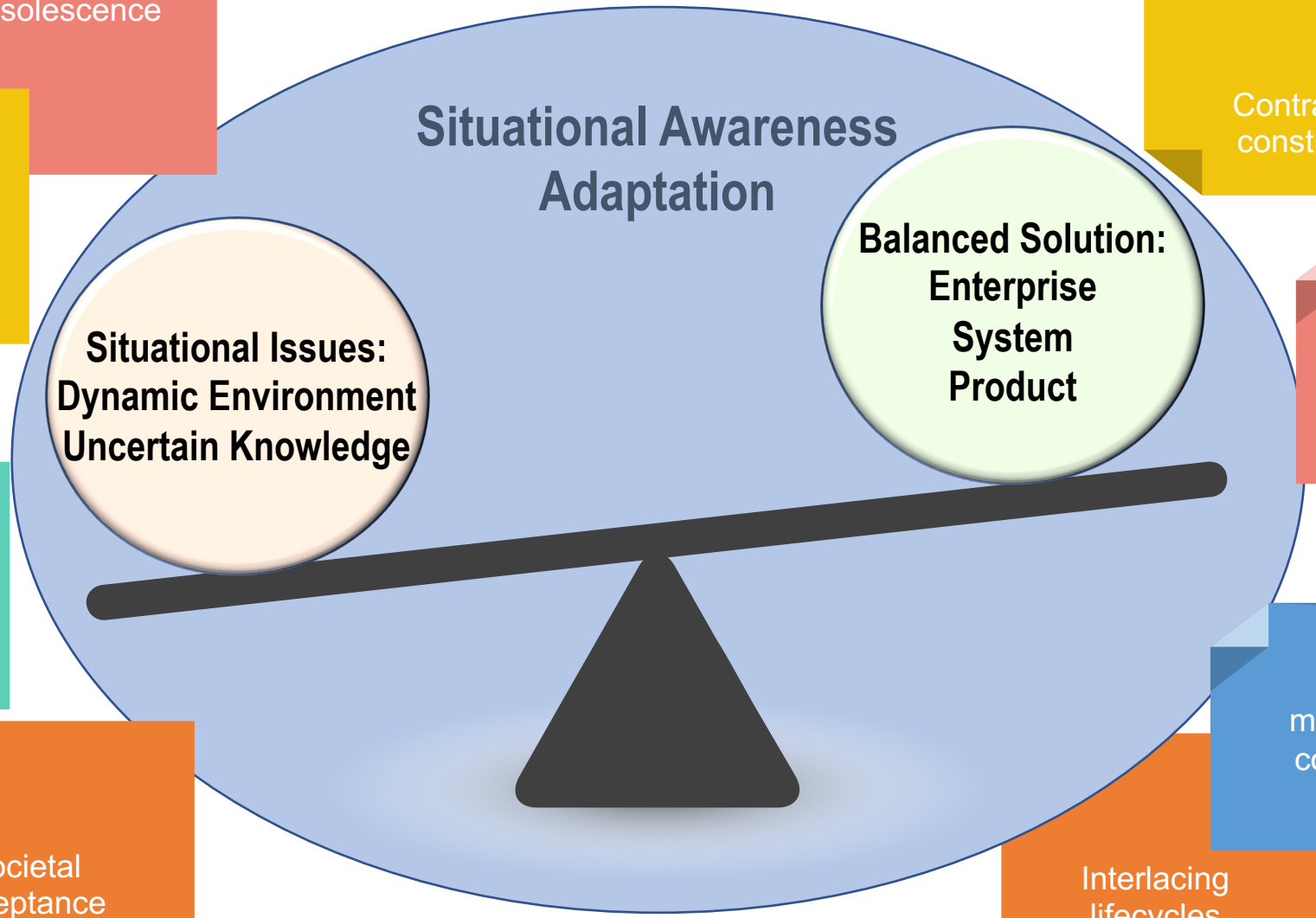
3 mins: Context
44 mins: Four different points of view
3 mins: One-slide summary
40 mins: Audience questions, comments, contributions

Agility is Essential for Situational Awareness Adaptation

Kerry Lunney

Thales Australia Country Engineering Director / Chief Engineer

Systems Engineering Agility Challenges



Supply chain disruptions

Obsolescence

Maintaining fidelity of models & prototypes

Reductionism focus

Handling long-lead items

Changing interdependencies across in-service life

Societal acceptance stability

Contractual constraints

Varying customers' expectations

Limited access to end users

Safety & security (including cyber) factors

Knowledge sharing barriers

Change management complexities

Interlacing lifecycles

Inflexible baselines

Misconceptions Leading to Limited Engineering Agility

"One size fits all" agility strategy

Maximising reuse

Frequent engagements

Reduction in cycle times

Adoption of tool / methodology reporting to provide an agile approach

Limited Engineering Agility

Temporal considerations are key to being agile

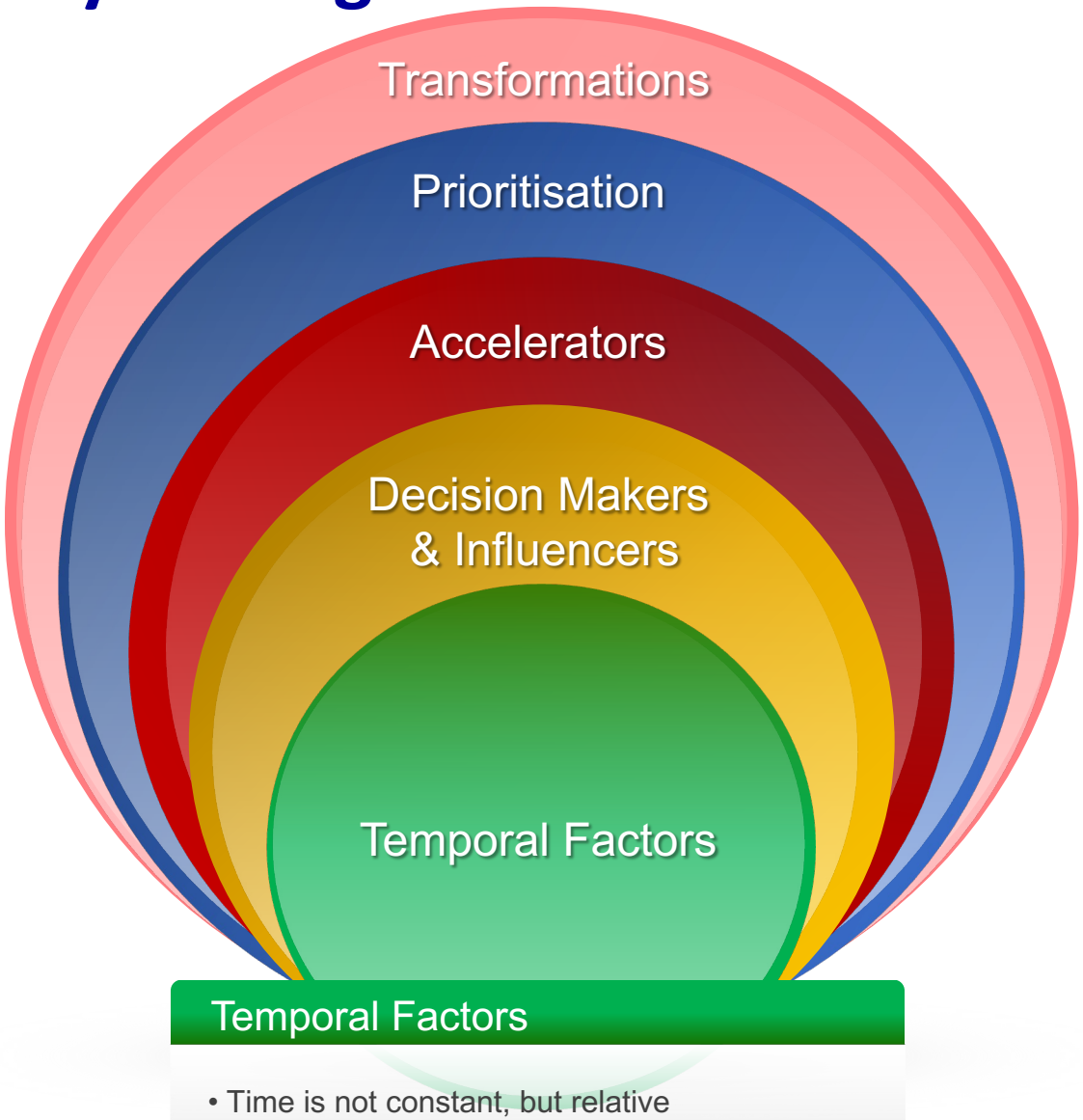
Engineering Agility Through Situational Awareness Adaptation

Accelerators

Rapid catalysts for adaptations – Minimal Viable Product/Capability (MVP/C) strategies / Mental models / Knowledge sharing / Tailoring / "Safe" labs & environments / Flexible Chain of Command / End user accessibility / Pilot strategies / Protected core features / Frequent engagements / Mission thread preservation

Decision Makers & Influencers

- Over-arching governance
- Decision making empowerment to those closer to the problem/activity
- Multiple levels of delegation & responsibilities
- Re-balance customer expectations



Temporal Factors

- Time is not constant, but relative
- Situational awareness changes with time
- Adaptations need to be fluid but controlled

Transformations

Enabling adaptations – Digital transformations / MVP/MVC / Models, prototypes, digital twinning / Agile tools & methodologies / Supply chain optimisation / Feedback loops coupled with decision authority / Aids for test & recovery / Reuse & variability / Robust change management

Prioritisation

- Common shared vision
- Re-alignment of goals
- Playbook
- Enabling pathway from accelerators to transformations

Take aways

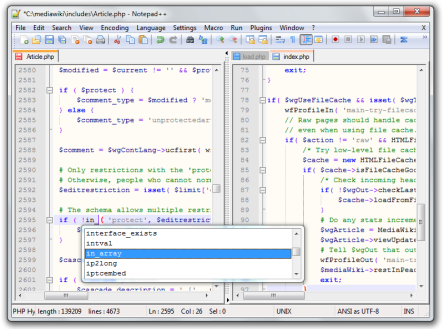
Engineering Agility is required for situational awareness adaptations

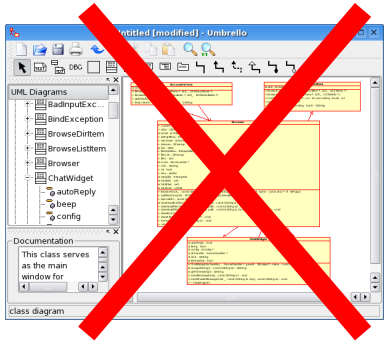
The dynamics of time constantly impact agility approaches

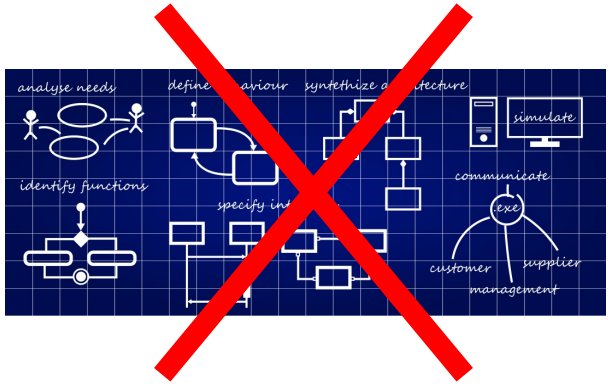
Temporal Factors + Decision Makers/Influencers + Accelerators + Prioritisation + Transformations → ↑ Engineering Agility

Professor Duncan Kemp, CEng, FIET, INCOSE Fellow
UK Ministry of Defence

A long time ago in a galaxy far-far away







Agile Systems Engineering

Truly Agile, and the closest view to the Agile SW Approach.

Not true Agile, as the requirements remain fixed, but offers significant benefits in cycle time, cost and risk reduction.

Agile Systems Engineering
A high tempo approach, with a rapidly changing environment and requirements, using operational feedback to drive development.

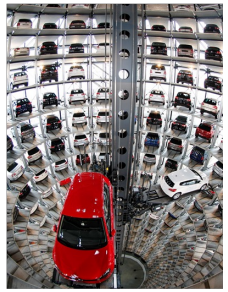
Rapid Development
Using a mixture of processes, tools and radical tailoring to reduce the time spent on conventional SE in order to meet a defined requirement.

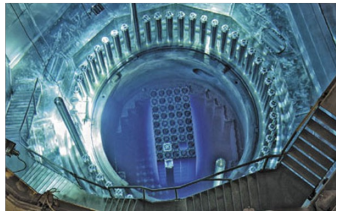
Agile Design
Using Agile software engineering techniques, combined with model based approaches, to improve SE document/model development (in terms of time to develop, quality of products and stakeholder buy in).

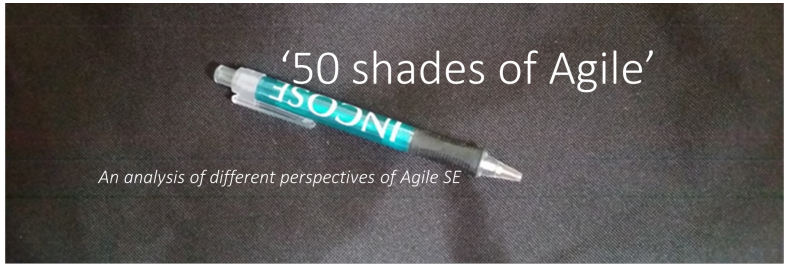
Snake Oil
Using Agile as a means to sell methods, tools or consultants with no clear understanding of how, or whether, this will succeed.

Whilst not Agile SE, they can help manage the uncertainty of early SE tasks.

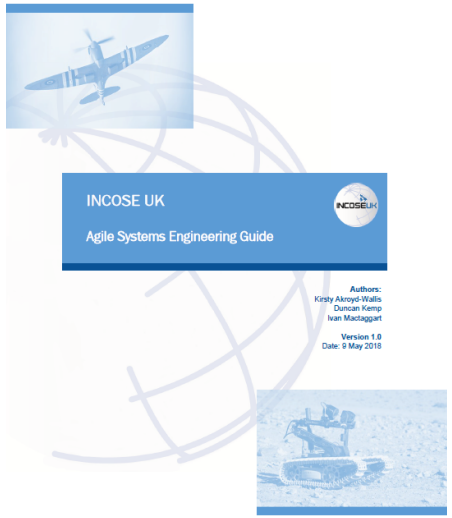
Although these approaches may win business, they will ultimately fail, and risk devaluing the wider Agile brand.

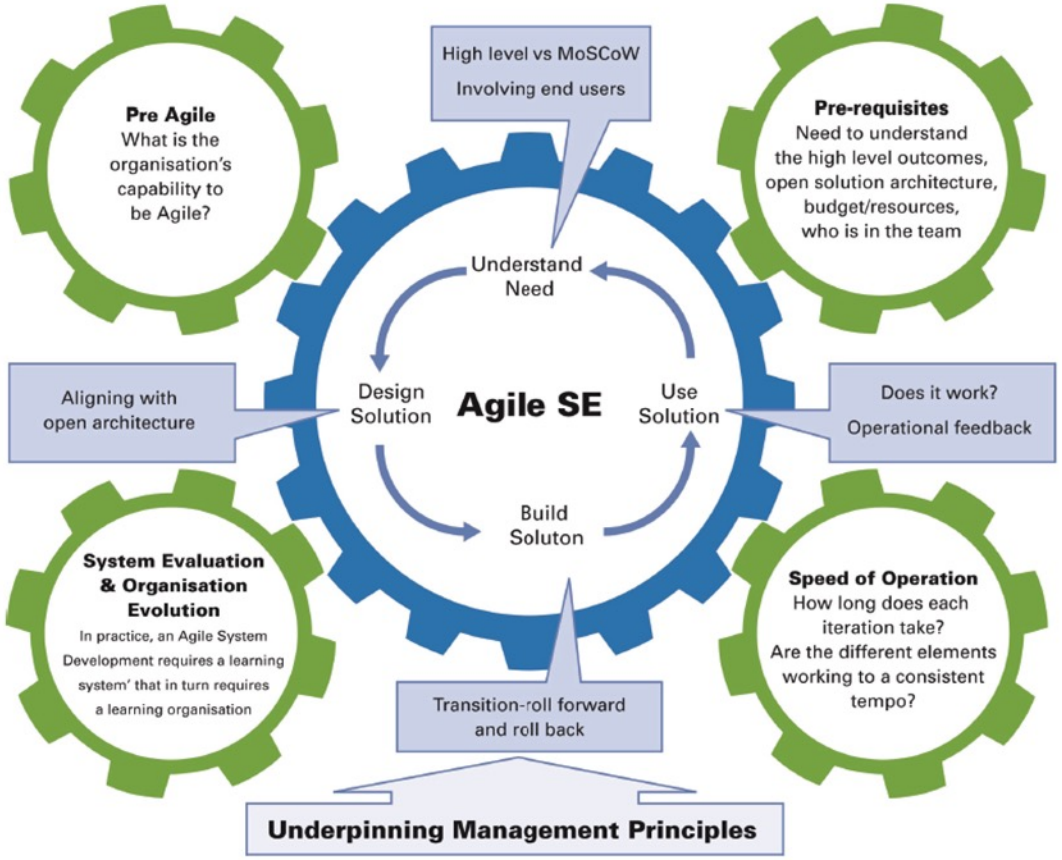






Agile Systems Engineering





Take aways

There is a lot of confusion as to what Agile Systems Engineering is

Vanilla Agile Software Engineering techniques will not work on Agile Systems

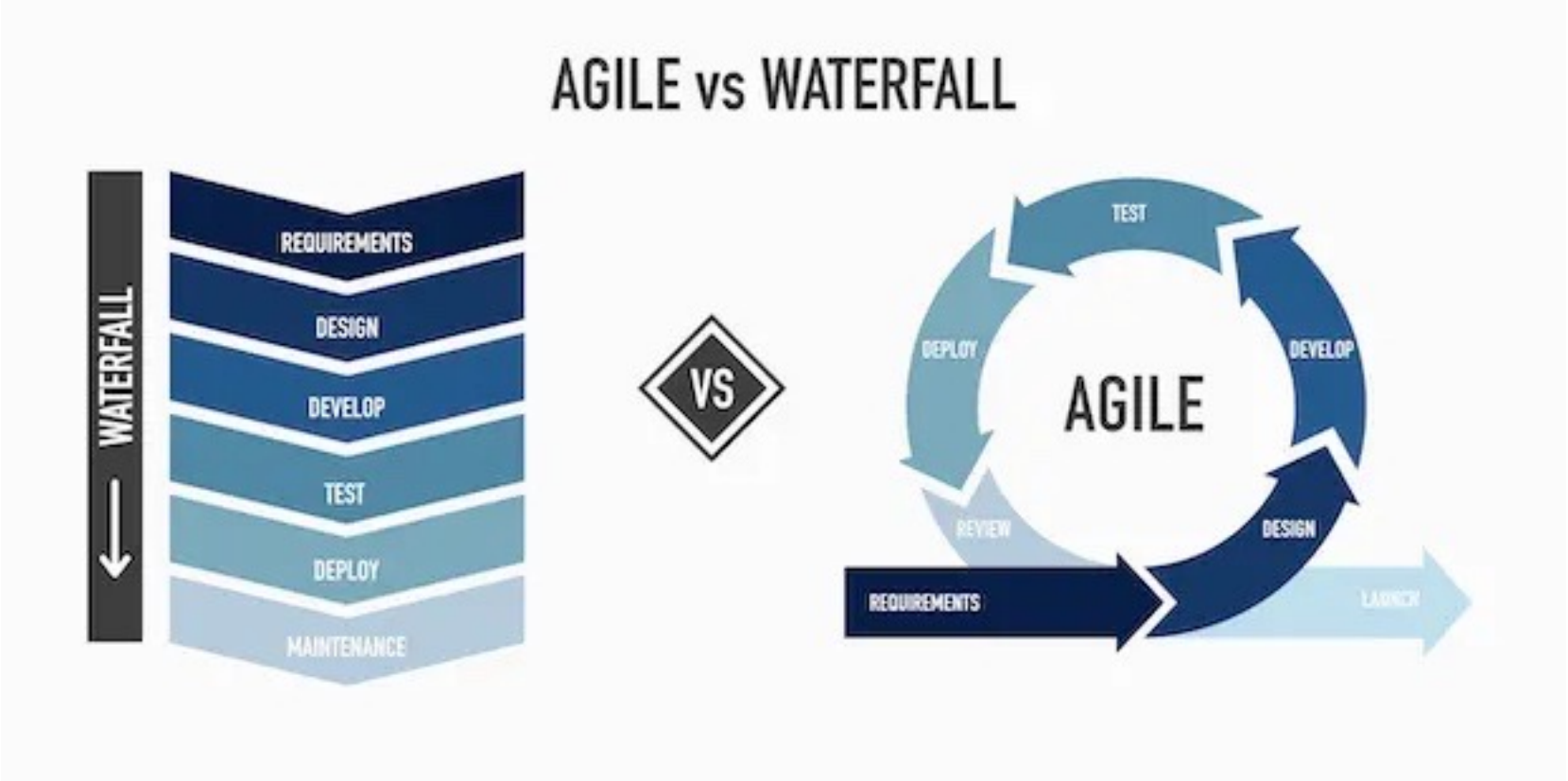
But it is possible, and necessary, to increase the agility of SE

Robin Yeman, SAFe Fellow / Systems Engineering PHD Candidate

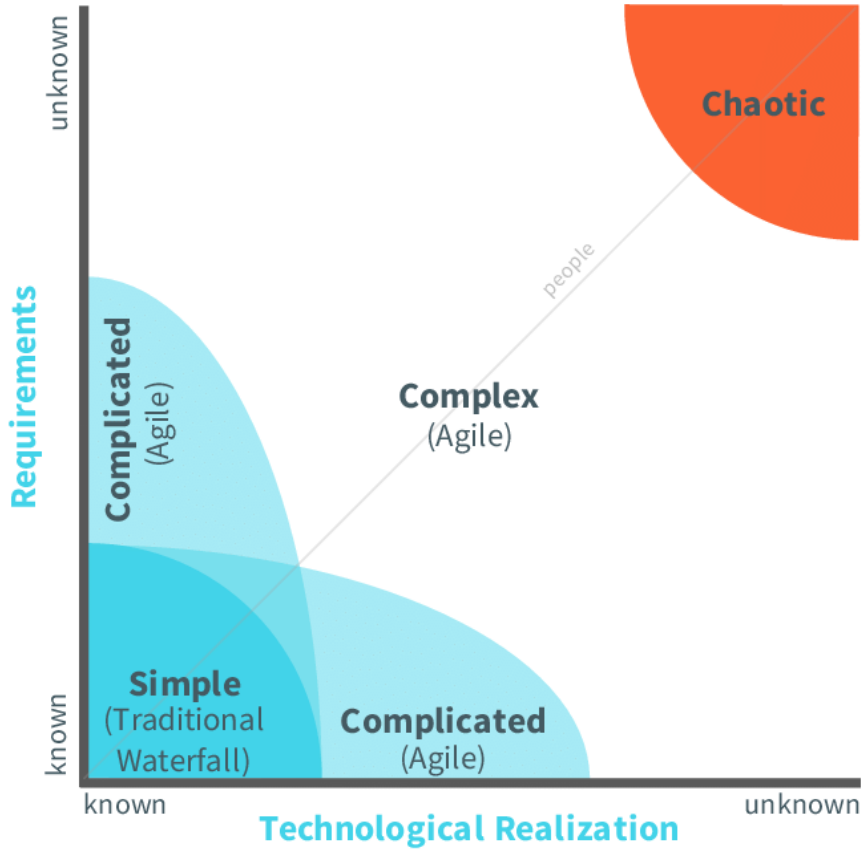
Agile is an empirical lifecycle

Agile is a Lifecycle

Waterfall is a predictive lifecycle based on phase gates, Agile is an empirical lifecycle based on objective data.

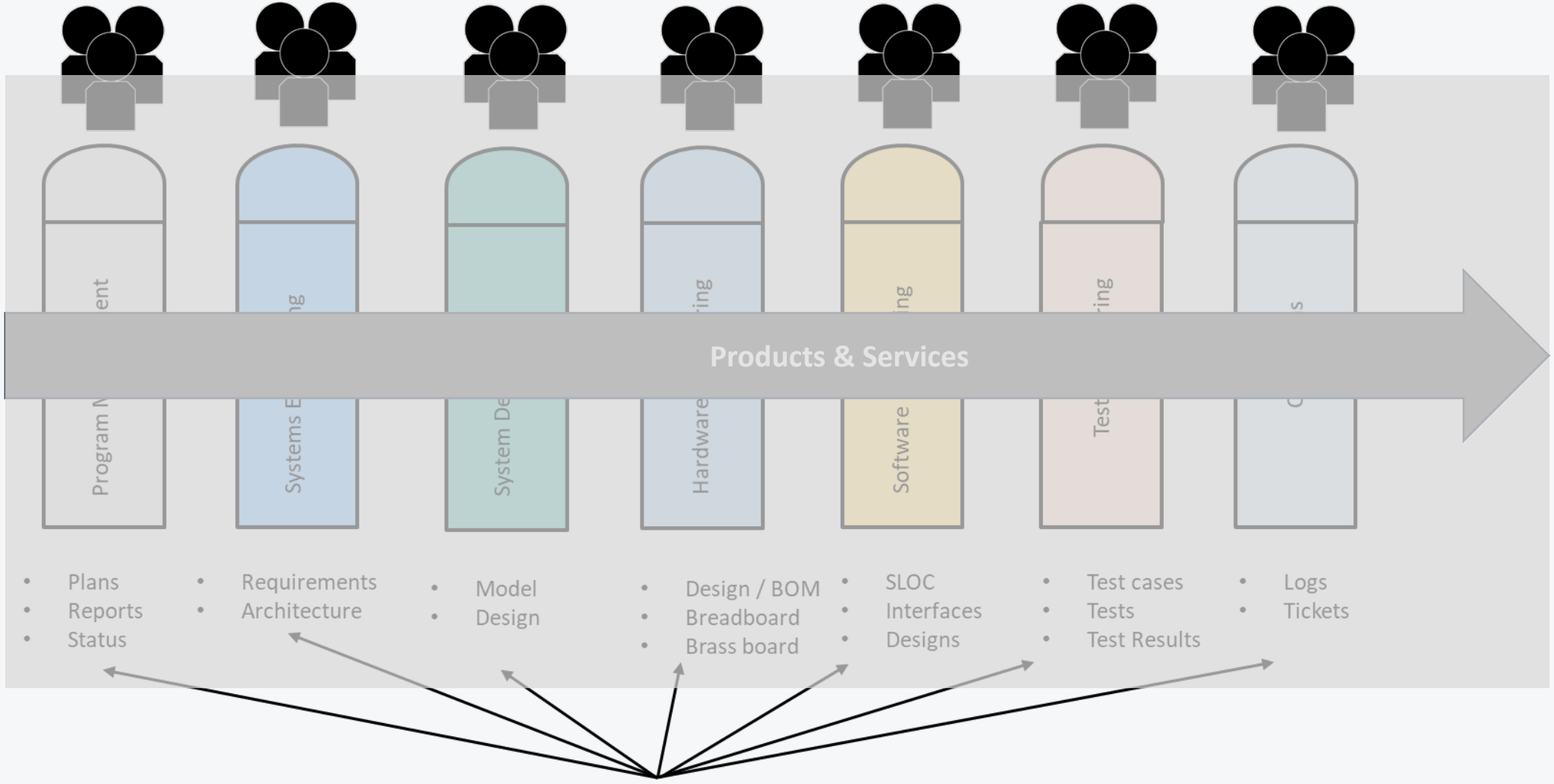


Begin with Why

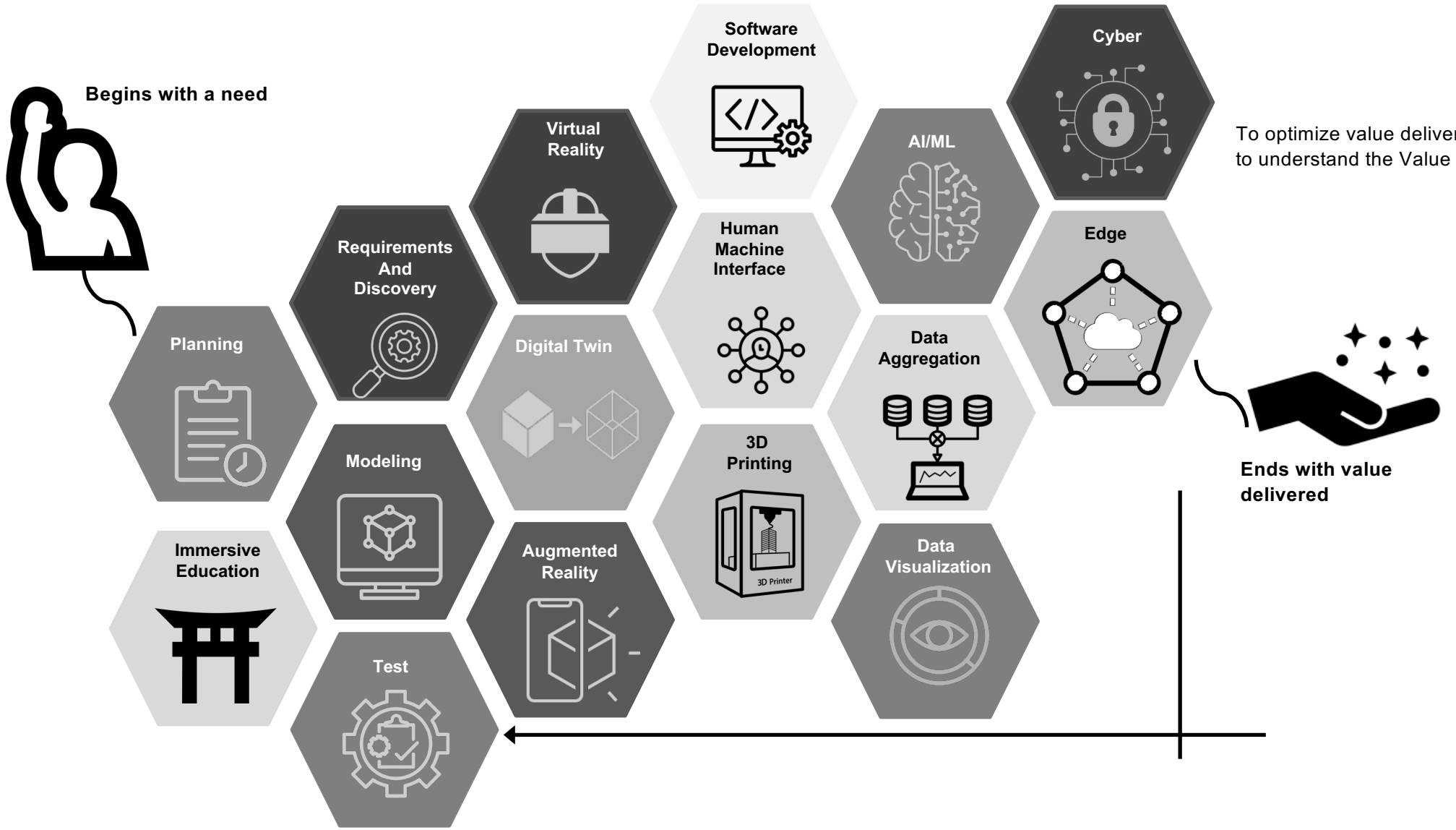


*The motivation to migrate to Agile is a demand for **faster** development, difficulty managing **change**, and increased product **complexity***

Organization and Common Language



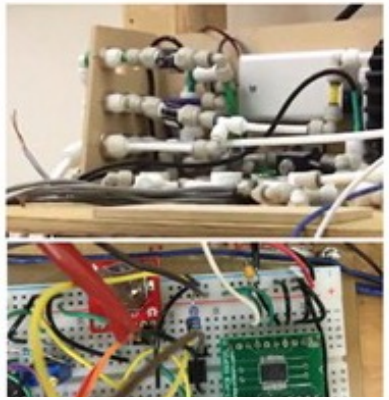
Use all the tools in Value Stream



Agile Engineering

Plywood Prototype	Hydraulic Prototype	"Alpha" Near-Final Prototype	"Beta" Fully-Functional Verification Ready

• Wooden support frame
• Power-on
• Installation features
• Time-based filter replacement
	• Functional hydraulics
	• In-line sensors installed
	• Read sensors
	• Flow-based filter expiration
		• 'Rev 1' hardware
		  • PCBA
		  • Sheet metal
		  • Cables
		• Internet connection
		• Logging
		• Secondary GUI screens
			• 'Rev A' hardware
			• Service screens
			• Reliability improvements
			• Final GUI improvements
			• Bug fixes



https://www.alten.com/mechanical-electrical-engineering-agile-method/

Take aways

Stop thinking of Agile as a software practice

Move from a predictive to empirical lifecycle

Use all the tools in your toolbox

SE Agility Primer (wip)

Eight Strategic Aspects

Tuesday @ 1:30

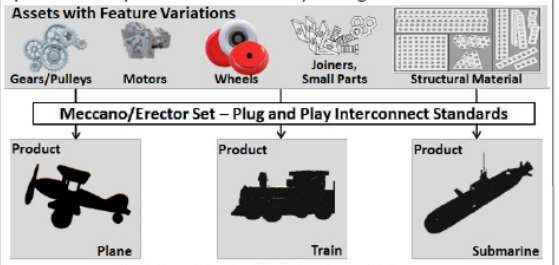
Product Line Architectures

Needs: Facilitated product and process experimentation, modification, and evolution.

Behaviors: Composable and reconfigurable product and process designs from variations of reusable assets.

Discussion: One fixed process approach won't fit all projects, so an appropriate process should be easy to compose and evolve according to context and usage experience. Variations of reusable assets are built over time as features are modified for different contextual usage.

A hallmark of agile systems engineering is iterative incremental development, which modifies work in process as suitability is repetitively evaluated. The agility of the process depends upon the agility of the product – so both process and product can be easily changed.



Notional Agile Architecture Pattern

Attentive Situational Awareness

Needs: Timely knowledge of emergent risks and opportunities.

Behaviors: Active monitoring and evaluation of relevant internal and external operational-environment factors.

Discussion: Are you doing things right (internal awareness) and doing the right things (external awareness)? Having the agile capability for timely and cost-effective change does little good if you don't know when that ability should be exercised. Situational awareness can be enhanced with systemic methods and mechanisms.

Alert in-the-moment constant attention

Common-Mission Teaming

Needs: Coherent collective pursuit of a common mission.

Behaviors: Engaged collaboration, cooperation, and teaming among all relevant stakeholders.

Discussion: Collaboration, cooperation, and teaming are not synonymous, and need individual support attention. Collaboration is an act of relevant information exchange among individuals, cooperation is an act of optimal give and take among individuals, and teaming is an act of collective endeavor toward a common purpose.

Tightly integrated coherent operation

Continual Integration & Test

Needs: Early revelation of system integration issues.

Behaviors: Integrated demonstration and test of work-in-process.

Discussion: Discovering integration issues late in development activities can impact cost and schedule with major rework. Synchronizing multiple domain engineering activities via continual integration and test provides faster and clearer insight into potential system integration issues.

SpaWar iteratively evolving unmanned technology integration platform.

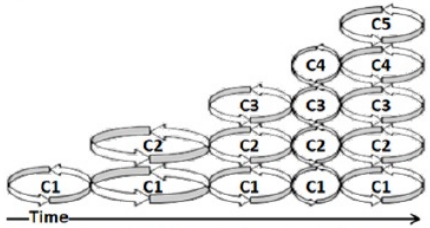
Iterative Incremental Development

Needs: Minimize unexpected rework and maximize quality.

Behaviors: Incremental loops of building, evaluating, correcting, and improving capabilities.

Discussion: Generally increments *create* capabilities and iterations add and augment features to *improve* capabilities.

- Increment cycles are beneficially timed to coordinate events such as integrated testing and evaluation, capability deployment, experimental deployment, or release to production.
- Increments may have constant or variable cadence to accommodate management standards or operational dynamics.
- Iteration cycles are beneficially timed to minimize rework cost as a project learns experimentally and empirically.



Iterative capability improvements (looping) and incremental capability additions (successive development periods)

Attentive Decision Making

Needs: Timely corrective and improvement actions.

Behaviors: Systemic linkage of situational awareness to decisive action.

Discussion: Empower decision making at the point of most knowledge. As a counter example, technical debt (a term for knowing something needs correction or improvement but postponing action) is situational awareness without a causal link to prompt action.

John Boyd's OODA loop

Shared-Knowledge Management

Needs: Accelerated mutual learning and single source of truth for internal and external stakeholders.

Behaviors: Facilitated communication, collaboration, and knowledge curation.

Discussion: There are two kinds of knowledge to consider. Short time frame operational knowledge: what happened, what's happening, what's planned to happen. Long time frame curated knowledge: what do we know of reusable relevance, e.g., digital artifacts, lessons learned, and proven practices.

Depicted books represent information containers of any kind; but typically digital

Being Agile: Operations Concept

Needs: Attentive operational response to evolving knowledge and dynamic environments.

Behaviors: Sensing, responding, evolving.

Discussion: Agile systems engineering is not about doing Agile, it is about being agile. Being agile is a behavior, not a procedure – a behavior sensitive to threats and opportunities in the operational environment, decisive when faced with threat or opportunity, and driven to improve these capabilities. Deciding how to implement any of the core aspects, even this one, should be done with sense-respond-evolve principles in mind as aspect objectives.

Three principles that operationalize agility

Strategic Aspects 1/2

Product-Line Architecture

What: Composable and reconfigurable designs from variations of reusable assets.
Why: Facilitated product & process experimentation, modification, and evolution.

Iterative Incremental Development (and delivery)

What: Incremental loops of building/evaluating/correcting/improving/delivering capabilities.
Why: Minimize unexpected rework and maximize quality.

Attentive Situational Awareness

What: Active monitoring/evaluation of internal & external threats/opportunities.
Why: Timely knowledge of emergent risks and opportunities.

Attentive Decision Making

What: Systemic linkage of situational awareness to decisive action
Why: Timely corrective and improvement actions.

Strategic Aspects 2/2

Common-Mission Teaming

What: Engaged collaboration, cooperation, and teaming among stakeholders.
Why: Coherent collective pursuit of a common mission.

Shared-Knowledge Management

What: Facilitated communication, collaboration, and knowledge curation.
Why: Accelerated mutual learning and single source of truth for all stakeholders.

Continual Integration and Test

What: Integrated demonstration and test of work-in-process.
Why: Early revelation of system integration issues.

Being Agile: Operations Concept

What: Sensing, responding, evolving.
Why: Attentive response to evolving knowledge and dynamic environments.

Common Strategies, Different Tactics

Provocative examples: many contrasts can be listed in each row depending upon project context.

Strategic Aspect	Software Engineering	Systems Engineering
Product-Line Architecture	• Standard interface • COTS object-oriented integrated dev environment	• Proprietary interface • MOSA design effort
Iterative Incremental Development	• Tight • Test time stability	• Loose • Test facility conflicts
Attentive Situational Awareness	• Introspective • Frequent user feedback	• Extrospective • Sparse user feedback
Attentive Decision Making	• Simple • Few people	• Complicated • Many people
Common-Mission Teaming	• Homogeneous • Shared language	• Heterogeneous • Different languages
Shared Knowledge Management	• Code libraries • Integrated tools	• PLM • Federated tools
Continual Integration and Test	• Common platforms • Synchronous	• Proprietary platforms • Asynchronous
Being Agile: Operations Concept	• Do Agile • Many COTS Options	• Be agile • Home grown and tailored

Take aways

Common goals and strategies (why & what)

Context dependent tactics (how)

Discussing contrasts can provoke thinking and understanding

Last Words

Kerry
- **Engineering Agility is required for situational awareness adaptations**
- **The dynamics of time constantly impact agility approaches**
- **Temporal Factors + Decision Makers/Influencers + Accelerators + Prioritisation + Transformations → ↑ Engineering Agility**

Duncan
- **There is a lot of confusion as to what Agile Systems Engineering is**
- **Vanilla Agile Software Engineering techniques will not work on Agile Systems**
- **But it is possible, and necessary, to increase the agility of SE**

Robin
- **Stop thinking of Agile as a software practice**
- **Move from a predictive to empirical lifecycle**
- **Use all the tools in your toolbox**

Rick
- **Common goals and strategies (why & what)**
- **Context dependent tactics (how)**
- **Discussing contrasts can provoke thinking and understanding**