NDIA

2023 Systems & Mission Engineering Conference

# Systems Engineering Agility – Eight Strategic Aspects

October 16-19, 2023 – NDIA Systems & Mission Engineering Conference
Norfolk, Virginia

**Rick Dove**
**Chair, INCOSE Agile Systems & Systems Engineering Working Group**

**Abstract:** Agile software development has pioneered and proliferated methods for managing software projects (e.g. Scrum et al.) and engineering software products (e.g. XP et al.) when knowledge is uncertain and environments are dynamic. The success of these approaches is challenging other engineering disciplines to find better ways to navigate their development activities through similar uncertainties and dynamics.

Agile engineering, of any kind, employs strategies for designing, building, sustaining, and evolving purpose-fulfilling creations when knowledge is uncertain and operational environments are dynamic. Strategies address what needs to be accomplished and why, without constraints or directions on how. Eight core strategic aspects, discussed as behaviors in fulfilment of needs, will be shown to enable and amplify systems engineering agility.

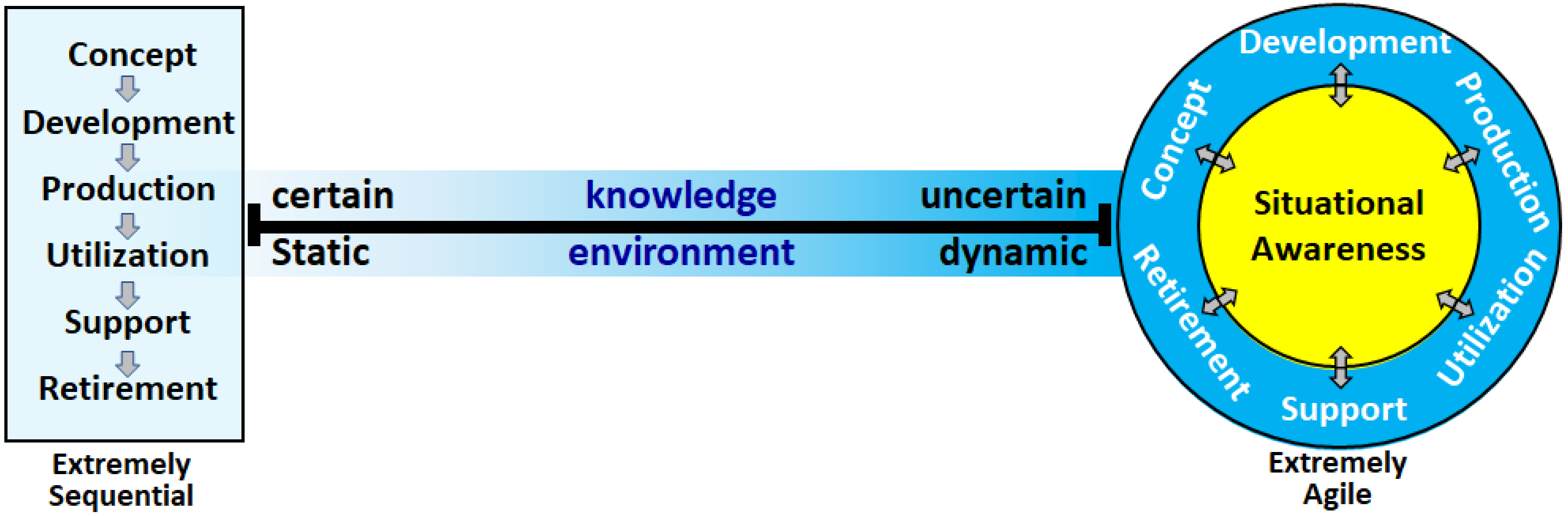
Agile software development methods (process tactics) necessarily leverage the nature of software engineering. A software product is created by engineers who are supported by an integrated hierarchy of many tools (computers, code compilers, user interfaces, development platforms, et al.) that gives them fast turn-around control over design, fabrication, and verification. Piecewise functional prototypes can be created and tested in minutes, and deployed into evolving user product in hours and days.

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. This presentation offers eight strategic aspects, each of which can individually improve capability to deal with uncertain knowledge and dynamic environments in any engineering process; but to have something intended as an agile engineering process at either domain or system level requires multiple aspects operating in concert. Individual aspects are strategic concepts that can tactically manifest over a range of intensity. Thus, the degree of agility is a product of how many of these aspects are operational as well as how effectively each one contributes to the agility required by the operating environment.

The eight strategic aspect for discussion are: Product Line Architectures, Iterative Incremental Development, Attentive Situational Awareness, Attentive Decision Making, Common-Mission Teaming, Shared-Knowledge Management, Continual Integration and Test, and Being Agile: the Operations Concept.

**Bio:** Rick Dove is an independent researcher, systems engineer, and project manager generally focused in the system security and system agility areas. He chairs the INCOSE working groups for System Security Engineering, and for Agile Systems and Systems Engineering; and leads INCOSE's Future of Systems Engineering (FuSE) project areas for both systems engineering security and systems engineering agility. He is an INCOSE Fellow, and author of Response Ability – the Language, Structure, and Culture of the Agile Enterprise.

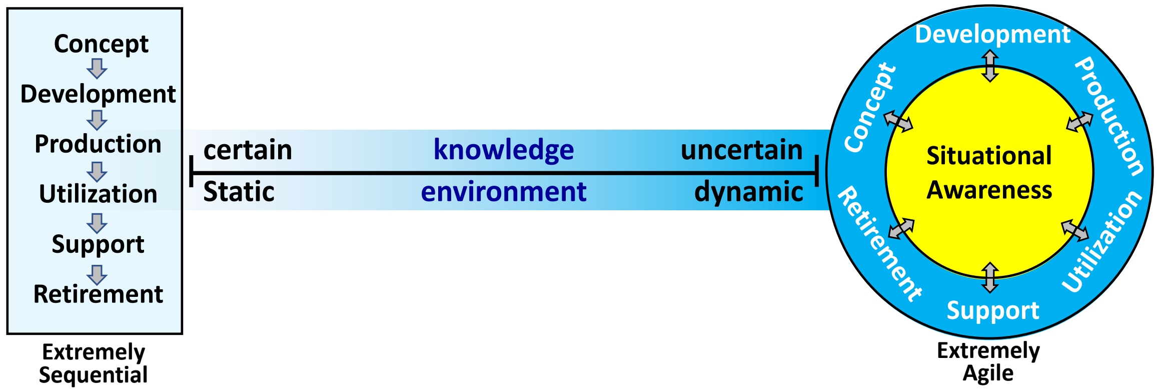
# Systems Engineering Life Cycle Spectrum
# Sequential to Agile



**Agile systems engineering is systems engineering as it is known through ISO/IEC/IEEE standards, the Vee model, and the INCOSE Handbook.**

**What distinguishes it as "agile" systems engineering is its leverage of situational awareness in driving continual evolution.**

# Chapter 2 – Context



INCOSE’s Vision 2035 expressed a fundamental **need**: “Systems engineering anticipates and effectively responds to an increasingly dynamic and uncertain environment.”

Ashby’s law expresses a timeless **need**: “When the variety or complexity of the environment exceeds the capacity of a system the environment will dominate and ultimately destroy that system.”

**Asynchronous Stage Activity**

- You’re using a personal computer in the morning = utilization stage.
- In the afternoon an SSD (Solid State Drive) is installed = production/deployment stage.
- Which replaces the Hard Drive = retirement.
- Next day the BIOS are adjusted for optimal SSD performance = support stage.
- Supplier is creating a new widescreen monitor based on market demand research = development stage.
- Supplier is always dreaming up product line extensions = concept stage.

**Concurrent Stage Activity** `www.parshift.com/s/AgileSE-206.pdf` **(SE agility at Tesla)**

- You’re driving a year-old Tesla = utilization stage.
- Simultaneously Tesla is downloading an AI upgrade = production/deployment stage.
- Simultaneously that upgrade is replacing an older capability = retirement stage.
- Simultaneously engineers are creating a market-desired faster charge capability = development stage.
- Simultaneously engineers are dreaming up tomorrow’s upgrade = concept stage.
- Simultaneously Tesla is asking to schedule a part-replacement house call based on monitored stats = support stage.

## Systems Engineering Agility

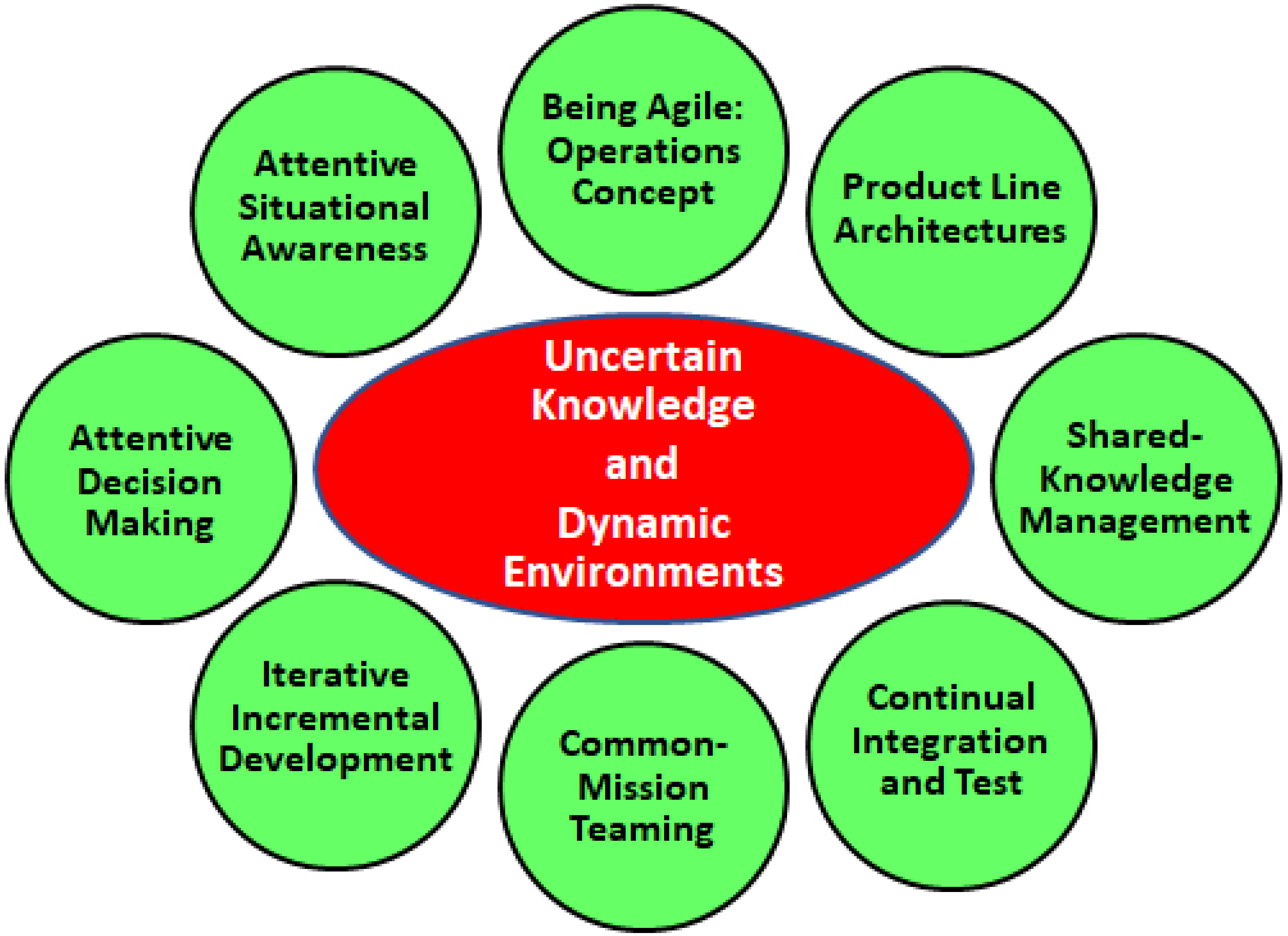
**Agile systems engineering is a strategy-driven method
for designing, building, sustaining, and evolving systems
when knowledge is uncertain and/or environments are dynamic.**

**Agile systems engineering is *being* agile, not *doing* agile.**

**Agile System Engineering is a what, not a how;
a strategic intent, not a tactical method.**

**There are many different methods that can be adopted, adapted, or crafted
to suite project contexts and organizational cultures;
but all share the same goals and strategies for being.**

## Eight Strategic Aspects Enable Agility


Being Agile: Operations Concept
Product Line Architectures
Shared-Knowledge Management
Continual Integration and Test
Common-Mission Teaming
Iterative Incremental Development
Attentive Decision Making
Attentive Situational Awareness
Uncertain Knowledge and Dynamic Environments

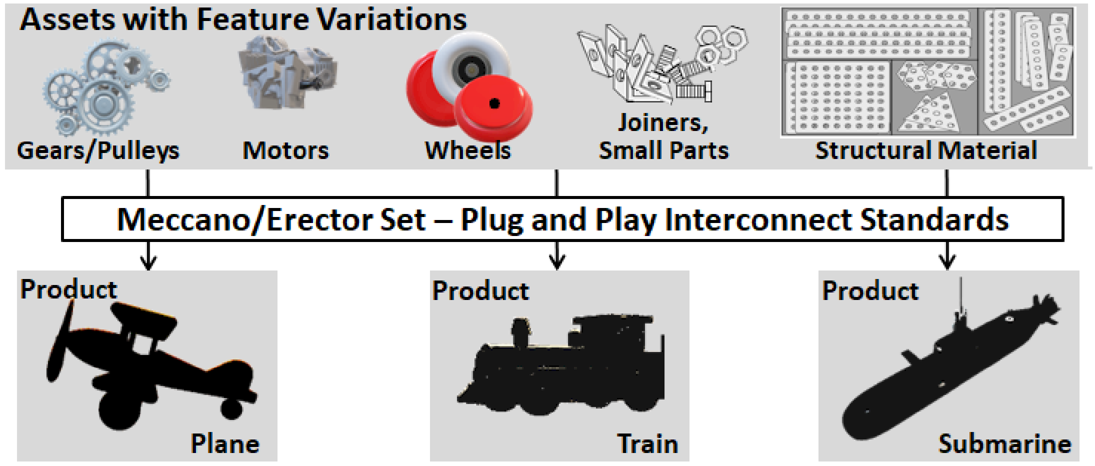
# Adaptable Modular Architectures

**Why/Needs**: Facilitated product and process experimentation, modification, and evolution.

**What/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 on the agility of the product – so both process and product can be easily changed.



Notional Agile Architecture Pattern

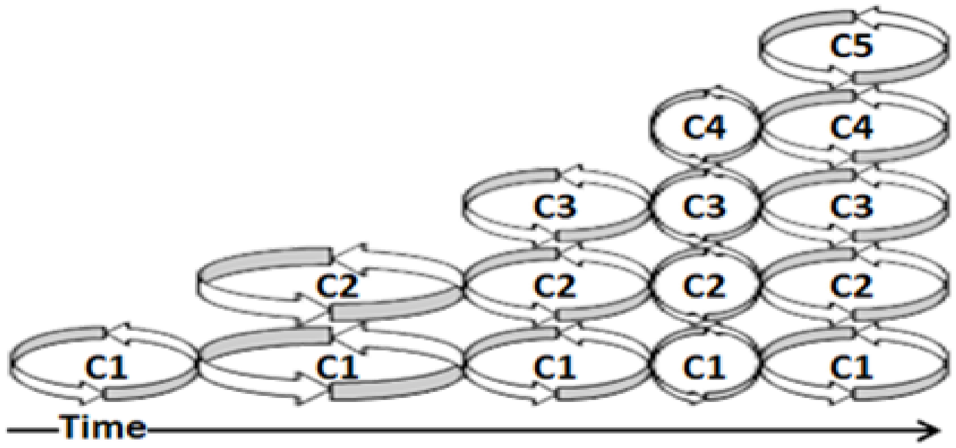
## Iterative Incremental Development

**Why/Needs**: Minimize rework, maximize quality, drive innovation.

**What/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, experimental deployment, release to production, or delivery.
- 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 columns)

## Attentive Situational Awareness

**Why/Needs**: Timely knowledge of emergent risks and opportunities.

**What/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.

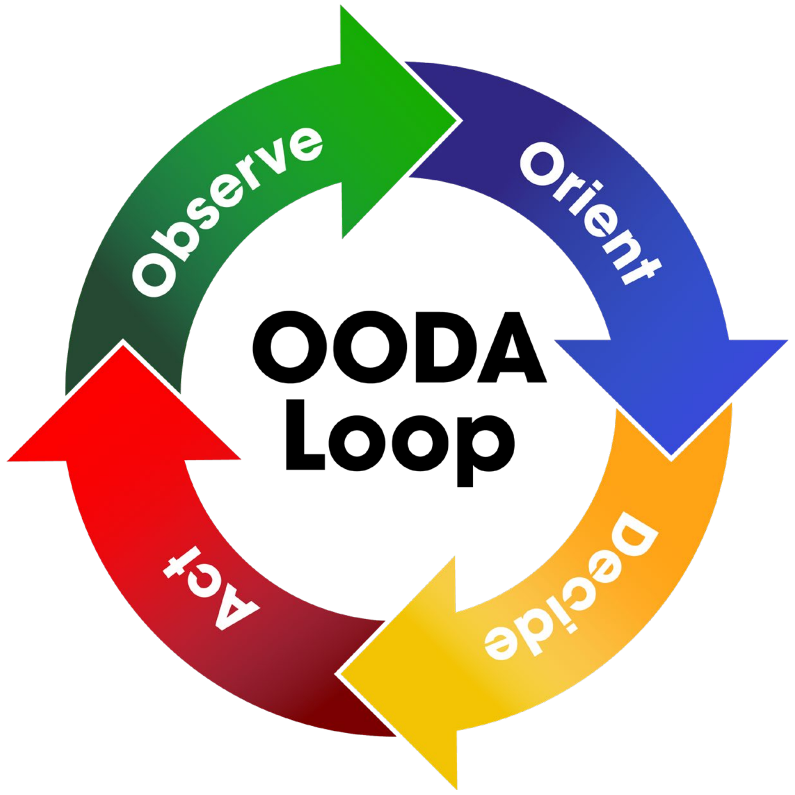
Alert in-the-moment constant attention

## Attentive Decision Making

**Why/Needs**: Timely corrective and improvement actions.

**What/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

## Common-Mission Teaming

**Why/Needs**: Coherent collective pursuit of a common mission.

**What/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

## Shared-Knowledge Management

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

**What/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

## Continual Integration & Test

**Why/Needs**: Early revelation of system integration issues.

**What/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.

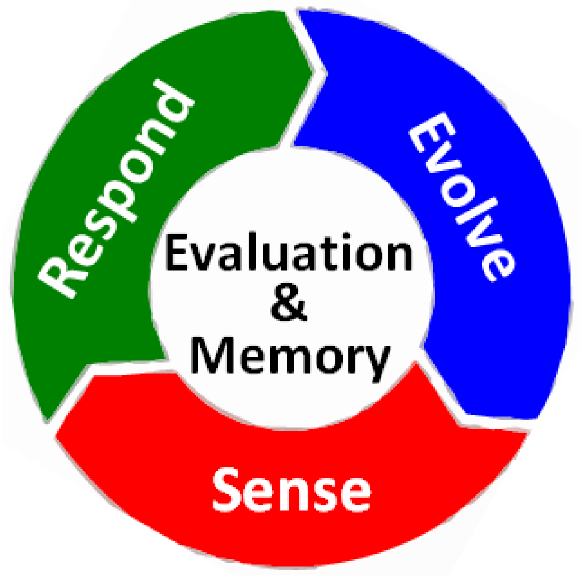
SpaWar iteratively evolving unmanned technology integration platform.

## Being Agile: Operations Concept

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

**What/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

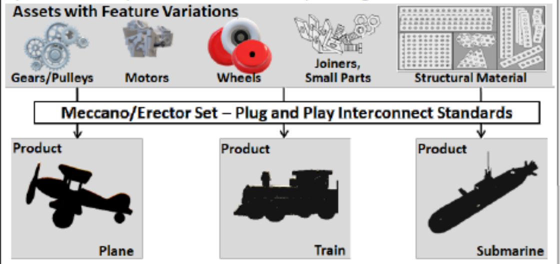
## Adaptable Modular Architectures

**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 things being done right (internal awareness) and are the right things being done (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 test and demonstration 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.

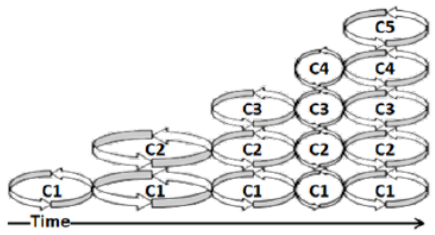
## Iterative Incremental Development

**Needs**: Minimize rework, maximize quality, drive innovation.

**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.



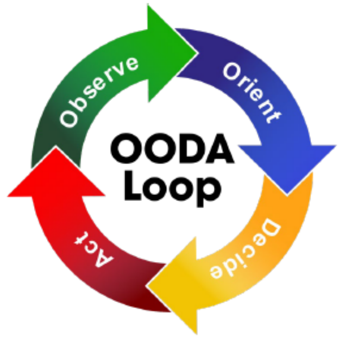
Iterative capability improvements (looping) and incremental capability additions (successive development periods)

## 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.

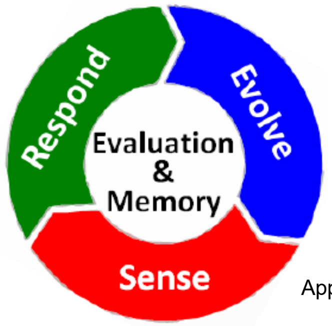
Depicted books represent information containers of any kind; but typically digital

## 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

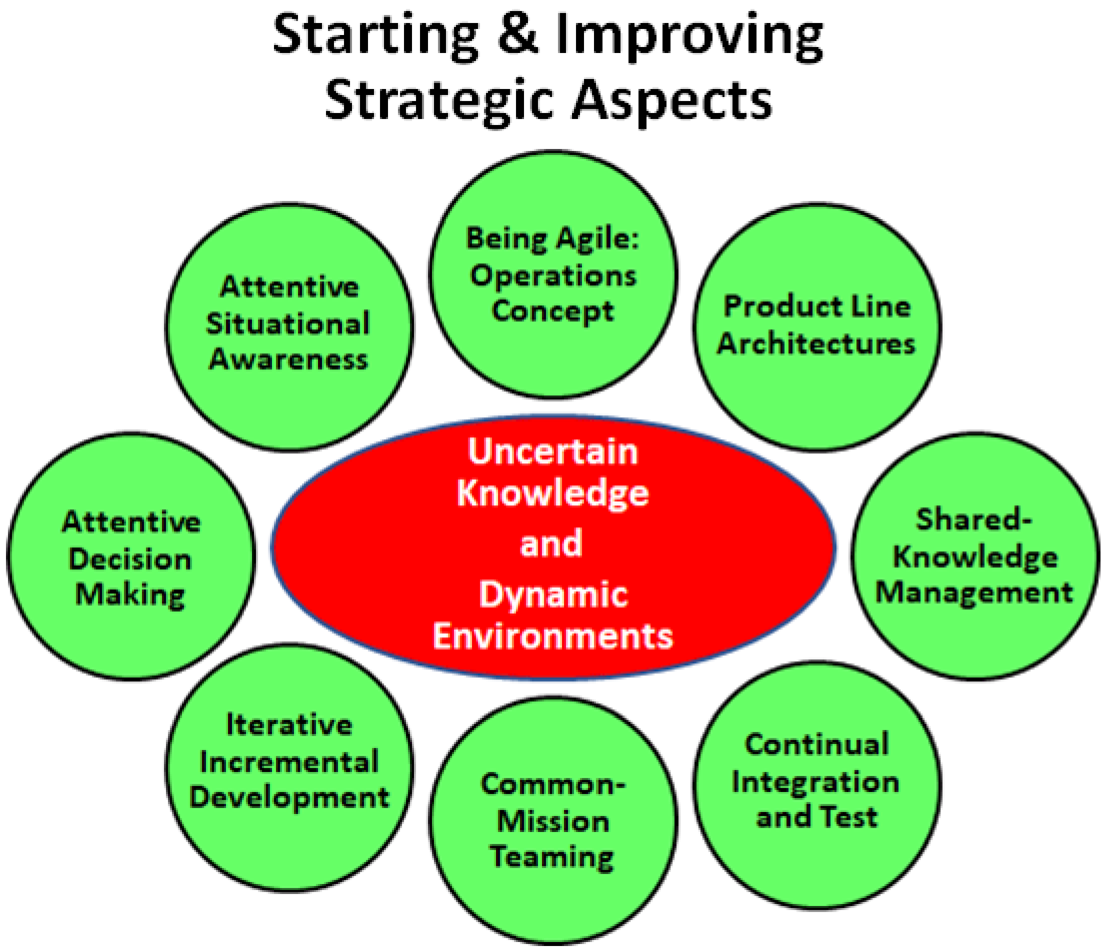
# Aspect Application is Context Dependent
## Common Strategies, Different Tactics

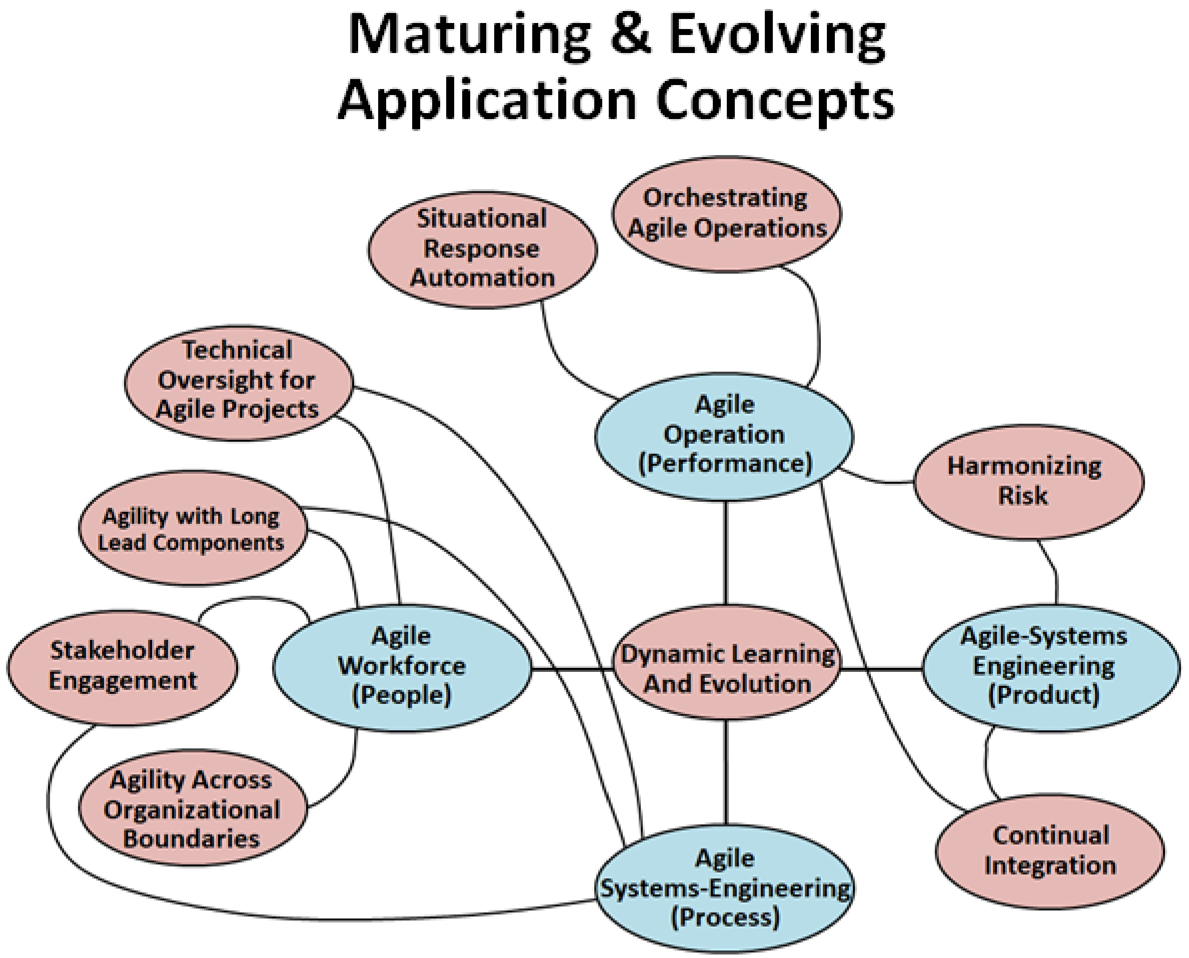
**Single-domain software engineering is different than multi-domain systems engineering.**

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

# Agile SE is a Journey


Starting & Improving Strategic Aspects
Attentive Situational Awareness
Being Agile: Operations Concept
Product Line Architectures
Attentive Decision Making
Uncertain Knowledge and Dynamic Environments
Shared-Knowledge Management
Iterative Incremental Development
Common-Mission Teaming
Continual Integration and Test


Maturing & Evolving Application Concepts
Situational Response Automation
Orchestrating Agile Operations
Technical Oversight for Agile Projects
Agile Operation (Performance)
Harmonizing Risk
Agility with Long Lead Components
Stakeholder Engagement
Agile Workforce (People)
Dynamic Learning And Evolution
Agile-Systems Engineering (Product)
Agility Across Organizational Boundaries
Agile Systems-Engineering (Process)
Continual Integration

**Large organizations likely have units working in both early and advanced stages**

## Supporting Material

1. Fundamentals of Agile Systems Engineering – Part 1 & Part 2. Dove, R., R. LaBarge. International Council on Systems Engineering. International Symposium, Las Vegas, NV, June 30-July 3, 2014. www.parshift.com/s/140630IS14-AgileSystemsEngineering-Part1&2.pdf
2. Introduction to the Agile Systems Engineering Life Cycle MBSE Pattern. Schindel, W., R. Dove. Proceedings International Symposium. International Council on Systems Engineering. Edinburgh, Scotland, July 18-21, 2016. www.parshift.com/s/160718IS16-IntroToTheAgileSystemsEngineeringLifeCycleMBSEPattern.pdf
3. [Case Study:] Agile Systems Engineering Process Features Collective Culture, Consciousness, and Conscience at SSC Pacific Unmanned Systems Group. Dove, R., W. Schindel, C. Scrapper. Proceedings International Symposium. International Council on Systems Engineering. Edinburgh, Scotland, July 18-21, 2016. www.parshift.com/s/ASELCM-01SSCPac.pdf.
4. Case Study: Agile Hardware/Firmware/Software Product Line Engineering at Rockwell Collins. Dove, R., W. Schindel, R. Hartney. Proceedings 11th Annual IEEE International Systems Conference. Montreal, Quebec, Canada, April 24-27, 2017. www.parshift.com/s/ASELCM-02RC.pdf
5. Case study: Agile SE Process for Centralized SoS Sustainment at Northrop Grumman. Dove, R, W. Schindel, M. Kenney. Proceedings International Symposium. International Council on Systems Engineering. Adelaide, Australia, July 17-20, 2017. www.parshift.com/s/ASELCM-03NGC.pdf.
6. Case Study: Agile Systems Engineering at Lockheed Martin Aeronautics Integrated Fighter Group. Dove, R., W. Schindel, K. Garlington. International Council on Systems Engineering, International Symposium, Washington, DC, July 7-12, 2018. www.parshift.com/s/ASELCM-04LMC.pdf
7. Agile Systems Engineering Life Cycle Model for Mixed Discipline Engineering. Dove, R., W. Schindel. Proceedings International Symposium. International Council on Systems Engineering. Orlando, FL, July 20-25, 2019. www.parshift.com/s/ASELCM-05Findings.pdf
8. Systems Engineering the Conditions of the Possibility (Towards Systems Engineering v2.0). Willett, K.D. Proceedings International Symposium. International Council on Systems Engineering. July 20-22, 2020. www.researchgate.net/publication/343306942_Systems_Engineering_the_Conditions_of_the_Possibility_Towards_Systems_Engineering_v20
9. Agility in the Future of Systems Engineering (FuSE) – A Roadmap of Foundational Concepts. Willett, K.D., R. Dove, A. Chudnow, R. Eckman, L. Rosser, J.S. Stevens, R. Yeman, M. Yokell. Proceedings International Symposium. International Council on Systems Engineering. July 17-22, 2021. www.parshift.com/s/210717IS21-FuseAgilityRoadmap.pdf
10. Complete Dan Rasky Interview (SpaceX Secrets), Knowledge @ NASA. https://www.youtube.com/watch?v=MxIiiwD9C0E
11. Agile at Tesla with Joe Justice, eleven 25 minute mind-boggling audio podcasts across the enterprise. https://munwaic.com/podcasts/agile-tesla/

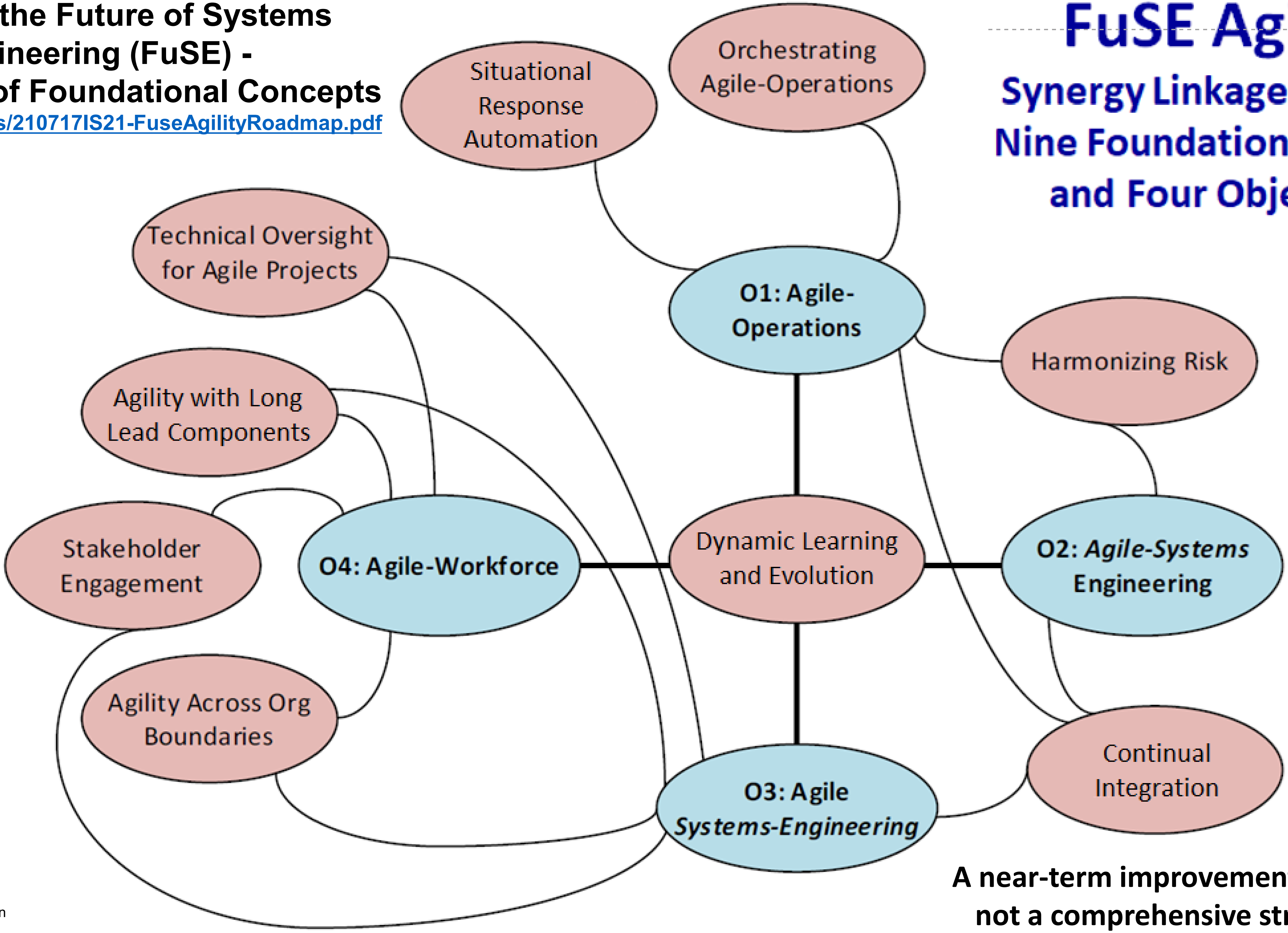
## Background

## Agility in the Future of Systems Engineering (FuSE) - A Roadmap of Foundational Concepts

www.parshift.com/s/210717IS21-FuseAgilityRoadmap.pdf

## FuSE Agility

**Synergy Linkage Between Nine Foundation Concepts and Four Objectives**


Situational Response Automation
Orchestrating Agile-Operations
Technical Oversight for Agile Projects
O1: Agile-Operations
Harmonizing Risk
Agility with Long Lead Components
Stakeholder Engagement
O4: Agile-Workforce
Dynamic Learning and Evolution
O2: Agile-Systems Engineering
Agility Across Org Boundaries
O3: Agile Systems-Engineering
Continual Integration

**A near-term improvement foundation, not a comprehensive strategy web.**

Approved for Public Distribution

# FuSE Agility Roadmap Concepts

| Concept Title | General Problem to Address | General Needs to Fill | General Barriers to Overcome |
|---|---|---|---|
| **1. Dynamic Learning and Evolution** | Insufficient learning and knowledge management processes;<br>barriers to learned-knowledge application. | Situational awareness and learning embedded in lifecycle processes; timely/affordable learning-application; knowledge management. | Unclear *what to do* or *where to do it* beyond learning ceremonies and contract obligation satisfaction. |
| 2. **Technical Oversight** | Traditional technical oversight methods are counterproductive in agile programs. | An interactive approach that reveals relevant knowledge for guidance and decision making. | Oversight traditions; standard contract wording; disrespect for oversight. |
| 3. **Stakeholder Engagement** | Timeliness and depth of stakeholder collaborative engagement. | Discovery of true requirements and integration conflicts. | Time involved; travel cost; inconvenient scheduling; lack of motivation. |
| 4. **Agility Across Organizational Boundaries** | Incompatible siloed cultures and languages. | Common language; less handoffs; product-based teams; common metrics. | Functional organizational silos. |
| 5. **Agility with Long Lead Components and Dependencies** | Components and external dependencies with long lead times complicate schedule coordination and disrupt technical performance. | Scheduling and acquisition techniques that better align with agile-SE principles. | [False] justification that long-lead items prohibit the use of agile-SE. |
| 6. **Continual Integration** | Late discovery of integration and requirements issues. | Minimize risk and rework with fast learning; maximize stakeholder engagement. | Development effort and expense; technologies for integrating/testing software prior to HW being ready. |
| 7. **Orchestrating Agile Operations** | Coherence among loosely coupled multi-actor outcomes. | Dynamic operational coordination in real-time. | Ability to encode self-learning;<br>adaptive logic as decision-support for people and for autonomous decision making. |
| 8. **Situational Response Automation** | Decision and action too slow. | Continual dynamic adaptation within cyber-relevant time. | Complicatedness of encoding autonomous governance and adjudication logic and rules; situational awareness that provides necessary inputs. |
| 9. **Harmonizing Risk in Agile Operations** | Agility focus is principally loss avoidance | Expand awareness and operational realization of both the negative side of risk (loss) and the positive side of risk (opportunity, seek gain, optimize). | Silo-thinking and predominance of looking at risk only in terms of loss. |