Domain Independent
Agile Systems Engineering Life Cycle Model Fundamentals
Discovery Project

An INCOSE Discovery Workshop Series

Rick Dove
575-586-1536, rick.dove@parshift.com

Chair, INCOSE Agile Systems & Systems Engineering Working Group
Adjunct Professor, Stevens Institute of Technology

Last Updated: 19-Nov-2016
The Time Has Come

INCOSE Vision 2025 Elements

- Resilient Systems
- Composable Design
- Adaptable and Scalable Methods

Top Five INCOSE CAB Priorities:
1) SE Professional development
2) Agile/Expedited methods
3) Effective Trade Studies
4) Product lines, re-use
5) Better Value proposal for INCOSE and SE

CAB: Corporate Advisory Board
Clarifying the Issues of CAB Agile-SE Priority

What the CAB workshop clarified on Agile Expedited Methods priority:

1. Clarity/consistency on what agile means independent of the software practice.
2. Guidance on when/where to use an agile approach.
3. Integrating agile approach concepts with planned approach concepts.
4. Systems as works in process after deployment
5. How to pivot a project effectively when feedback dictates a path change.
6. Short cycle constant evolution – e.g., counter-IED “systems”
7. Long cycle constant evolution – e.g., 20-year design/build for complex plants.
8. Meaningful WIP measures when an agile approach is employed.
9. Dealing with hardware design/build timeframes and sunk costs.
10. Case studies.

NOTES:

• Universal dissatisfaction among this group with the Agile SW Manifesto as a guide for agile SE.
• Conclusion: all needs are being addressed by the Agile Sys & SE WG, or will be in the Agile SE Life Cycle Model project.
What is an SE Life Cycle Model?


3.2.1 System life cycle model

Every system, whatever the kind or size, inherently follows some life cycle, evolving from its initial conceptualization through its eventual retirement…

A life cycle model, then, is a decision-linked conceptual segmentation of the definition of the need for the system, its realization as a product or service, and its utilization, evolution and disposal.

A system life cycle model is typically segmented by stages to facilitate planning, provisioning, operating and supporting the system-of-interest. These segments provide an orderly progression of a system through established decision-making gates to reduce risk and to ensure satisfactory progress.

As stated before, it is the need to make a decision to specific criteria before a system can progress to the next stage that is the most important reason for using a life cycle model.

Notes:

• Implies, but does not say, an SOI is in one and only one stage at any time.
• An Agile SE Life Cycle Model is distinguished from waterfall by allowing non-sequential stage progression and multiple-stage activities simultaneously.
• Key is the decision criteria that permits/demands any stage’s process activity.
Section 5.5.5 (p. 32):
“... to convey the idea that one can jump from a stage to one that does not immediately follow it, or revert to a prior stage or stages that do not immediately precede it.”

“Further, the text in the model indicates that one applies, at any stage, the appropriate life cycle processes, in whatever sequence is appropriate to the project, and repeatedly or recursively if appropriate.”

“While this may seem to be a total lack of structure, indeed it is not.”

“Rather, the structure has well defined parts that can be juxtaposed as needed to get the job done, flexibly but still in a disciplined manner, just as a real structure would be created.”

Seven asynchronously-invoked stages can be engaged repetitively and simultaneously to achieve benefit when engagement criteria are met.
Project: Agile SE Life Cycle Model (ASELCM) Fundamentals

Objectives:

A. Discover generic principle-based life-cycle stages/processes/activities that can be intuitively embraced and applied, rather than compromised by situational reality factors, for dealing with uncertain, unpredictable, evolving SE environments.

B. Cover a variety of SE project types, e.g.:
   1. discovery (verifying requirements and solution feasibility),
   2. programmatic (Systems and SoS from proven components),
   3. approach (e.g., ICSM methodology and product line architecture),
   4. quick reaction (rapid development and fielding),
   5. evolving (continuous change of system operational viability and opportunity, rapid sequential generations).

C. Recognize that ASELCM process activities within multiple life cycle stages may be occurring simultaneously, particularly after initial deployment.

Agreement Processes

Organizational Project-Enabling Processes

Project Processes

Technical Processes

Special Processes

Acquisition

Supply

Life Cycle Model Management

Infrastructure Management

Project Portfolio Management

Human Resource Management

Quality Management

Project Planning

Project Assess and Control

Decision Management

Risk Management

Configuration Management

Information Management

Measurement

Stakeholder Requirements Definition

Requirements Analysis

Architectural Design

Implementation

Integration

Verification

Transition

Validation

Operation

Maintenance

Disposal

Tailoring

20 Processes of Interest
Project Artifacts (Products)

1. An instructive technical report describing a generic Agile SE Life Cycle Model with supporting exemplar case studies. The model will support rather than supplant common agile systems-and-software SE processes.

2. Pattern Based SE Modeling (PBSE) will illustrate configurations aligned to the case studies (next slide).


4. Collateral technical information in briefer form and focus is anticipated as papers targeted for relevant SE journals and conferences.

5. At least two SMEs at each Host with insightful understandings.
Pattern-Based System Engineering (PBSE)

Pattern Class Hierarchy
Adapted from: Bill Schindel IS05 paper.

Some Level 2 Candidates:
- ICSM: Incremental Commitment Spiral Model
- OSA: Open System Architecture PM concept
- EVO: Evolutionary Project Management
- RD: Rapid Development/Fielding
- QRC: Quick Reaction Capability
- LVC: Live-Virtual-Constructive
- Scrum: Scrum PM concept
- Wave: Wave model
- SAFe
Strategies

1. The project will be guided by ISO/IEC TR 24748-1:2010 and recognize six primary continuous and potentially simultaneous stages of process activity: Research, Concept, Development, Production, Utilization, and Support. A seventh terminal stage, Retirement, may be considered if anything unique to agile SE is discovered during the project. Guidance will also be taken from ISO/IEC 15288-2008 to specifically analyze 19 selected Processes.

2. Workshop Hosts will provide discussion and presentation of one completed agile-SE experience for analysis, and a discussion/presentation of one SE approach in need of more agility to fuel a synthesis exercise based on accumulated learning.

3. Non-Host Traveling Participants may fill out workshops to max 20 total participants, with each participant, Host and non-Host, required to attend a minimum of 3 workshops.
Strategies²/²

4. **With a structured analysis approach**, analyze experience from employed agile SE practices in both defense and commercial SE projects that involve combined aspects of software, hardware, and wetware (management, engineering, operator, maintainer). Management includes supplier and acquirer project management aspects.

   - Discover and justify ("why" reasoning) common necessary and sufficient agile SE needs and reality factors, independent of what agile SE practice may be entrenched, favored, under consideration, or subsequently adopted.
   - Discover and justify ("why" reasoning) principle-based stages, processes, and activities that satisfy the project objectives.

5. **With a structured synthesis approach**, apply discovery and provide benefit to workshop hosts and participants with an application of accumulated learning to a relevant host opportunity or problem.

Reality check: We don’t *solve* the item-5 Host challenge at this workshop, but we do cause actionable thought. Values to Host come from workshop preparation, thoughtful engagement in the workshop, application of what was learned, attendance at minimum-two additional workshops, and eventual Case Study co-authored development.
Iconic Agile Architecture Pattern (AAP)
System Response-Construction Kit
Participants will construct AAP from Host discussion

Example: Scrum Agile Architecture Pattern (AAP)


Pro forma only – not comprehensive
### Example: Scrum Response Situation Analysis (RSA)


<table>
<thead>
<tr>
<th>Change Domain</th>
<th>Proactive</th>
<th>Reactive</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Creation (and Elimination)</strong></td>
<td>• requirements</td>
<td>• unanticipated expertise requirement</td>
</tr>
<tr>
<td></td>
<td>• experiments</td>
<td>• development activity-sequence priority change</td>
</tr>
<tr>
<td></td>
<td>• next sprint activity</td>
<td>• system/sub-system design change</td>
</tr>
<tr>
<td><strong>Improvement</strong></td>
<td>• process effectiveness</td>
<td>• 2x (or half x) project scope change</td>
</tr>
<tr>
<td></td>
<td>• risk/uncertainty reduction</td>
<td>• x to y engineers distributed across n to m locations</td>
</tr>
<tr>
<td><strong>Migration</strong></td>
<td>• new technology/tools that will impact infrastructure</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• lean SE process principles</td>
<td></td>
</tr>
<tr>
<td><strong>Modification (of Capability)</strong></td>
<td>• new team member unfamiliar/uncomfortable with agile SE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• new environmental situation</td>
<td></td>
</tr>
<tr>
<td><strong>Correction</strong></td>
<td>• wrong requirement</td>
<td>• non-compliant supplier</td>
</tr>
<tr>
<td></td>
<td>• wrong design</td>
<td>• inadequate developer</td>
</tr>
<tr>
<td></td>
<td>• inadequate implementation</td>
<td></td>
</tr>
<tr>
<td><strong>Variation</strong></td>
<td>• expertise and skill levels among team members</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• allowable deliverable performance range</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• customer availability, interaction, involvement expertise</td>
<td></td>
</tr>
<tr>
<td><strong>Expansion (of Capacity)</strong></td>
<td>• 2x (or half x) project scope change</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• x to y engineers distributed across n to m locations</td>
<td></td>
</tr>
<tr>
<td><strong>Reconfiguration</strong></td>
<td>• unanticipated expertise requirement</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• development activity-sequence priority change</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• system/sub-system design change</td>
<td></td>
</tr>
</tbody>
</table>
### Reality Factors

**Human Behavior:**
Non-team behavior, error, expediency, uncommitted customer rep, ...

**Organizational Behavior:**
Change in stakeholders, organizational priorities, resource access, ...

**Technology Pace:**
Evolving technology, testing trade-offs, ...

**Complexity:**
Large project with many involved simultaneously, emergent interaction affects, different ...

**Globalization:**
Partners/teams with different ethics, cultures, infrastructures, ...

**Partially-Agile Enterprise Concepts:**
Outsourcing, COTS affects, COTS supply/supplier affects, agile software practice-thinking dominance on HW/SW project...

**Agile Customers/Competitors/Adversaries:**
Continuous external-knowledge evolution, continuous external innovation, ...

---

Participants will construct Reality Factors from Host discussion

**Example: Scrum Environmental Reality Factors**

RSA exercises often assume a reasonably behaved and supportive environment, and tend to focus on the system’s internal functional response situations. This framework tool moves the analysis into the external environment.
Principle (Others) – where were these observed?

Participants will construct principles-observed analysis

~10 Principle Candidate Will Be Provided at Each Workshop

Principle 01 – where was this observed?

Principle 02 – where was this observed?

Principle 03 – where was this observed?

Principle 04 – where was this observed?

Principle 05 – where was this observed?

Principle 06 – where was this observed?

Principle 07 – where was this observed?

Principle 08 – where was this observed?

Principle 09 – where was this observed?

Principle 10 – where was this observed?

2015 workshops revealed initial candidate principles that might be necessary and sufficient to cope with a CURVE SE environment.

These will evolve through each subsequent workshop as they are confirmed/denied/augmented in what is observed in each subsequent workshop.

Each workshop will use the latest evolution of candidate principles for analysis.

Keep in mind that the principles being sought are context-independent – i.e., they should be applicable in any-and-all Agile SE processes.

This necessitates stating the principles as high level concepts – with generic intent.

Each workshop will provide examples of how these generic principles are exhibited in context-specific application.
Planned (Roughly) Workshop Agenda

------------- Day 1 – 8 hours of structured work starting at 8:00am, room open at 7:30.
2:00 – Introductions, objectives, workshop agenda structure, tools and processes.
2:00 – Host process presentation/discussion of SE CURVE situation and SE process.

Lunch (one hour lunch allows informal conversation)

2:00 – Breakout: analysis of SE-process RSA/RF (two teams doing identical analysis).
2:00 – Brief-out: Analysis results, discussion, and refinement.

Dinner (host-funded for all participants) at time TBD.

------------- Day 2 – 8 hours of structured work starting at 8:00am, room open at 7:30.
1:00 – Breakout: refinement of yesterday’s salient learning.
3:00 – Host presentation and Q&A of 20 processes.

Lunch (one hour lunch allows informal conversation).

2:00 – Breakout: Identifies observed process-guidance principles.
2:00 – Brief-out: observed process-guidance principles with discussion.

------------- Day 3 – 8 hours of structured work starting at 8:00am, room open at 7:30.
0:30 – Review Agile Architecture Pattern (AAP) concepts.
1:00 – Breakout: Develop AAP process representation, and refine principles as time permits.
0:30 – Brief-out: AAP process representation
1:00 – Host presentation/discussion and Q&A of process challenge (in any form wished).
1:00 – Break out: identify RSA issues for challenge area.

Lunch (one hour lunch allows informal conversation).

1:00 – Breakout cont.: describe the application of principles to address the RSA issues.
1:30 – Brief-out and wrap up.
0:30 – General reflection on the workshop process, tools, learning, and results.
1:00 – Private reflection among Host personnel, what might be applied?
Two different operational environments defining necessary agile counterpoint for the systems they encompass

It is counterproductive to have an agile development process if you don’t have an agile product architecture
Action Plan

• 10-15 (TBD) three-day structured workshops will be conducted at host sites in the US and Europe to analyze a variety of different types of agile SE experiences.

• The first four workshops were completed in 2015.

• Traveling participants must participate in at least 3 workshops. Host sites must provide at least two participants that will attend 2 additional workshops.

• Host sites will include both defense and commercial organizations.

• Workshops will analyze a host life-cycle experience, and then use accumulated learning to synthesize a host-chosen SE approach in need of more agility.

• Hosts will be expected to prepare a discussion presentation covering the processes to be analyzed and synthesized.

• Workshops will have up to 20 participants plus briefers. Participants are drawn first from current, previous and planned Hosts.

• Workshop output will be synopsized and available to participants within 30-days of each workshop.

• No system-functional details need be revealed, only SE life-cycle process and activity procedures. Proprietary and classified projects should not be a problem.
Outcomes and Benefits

Workshop Hosts:
• Diagnostic analysis of an agile SE process:
  • Revelation of strengths and weaknesses.
  • Valuation of what is being done and why.
  • Illumination of agility improvement opportunities.
• Action-learning synthesis applied to a challenging Host situation in need of more agility.
• Understanding of necessary and sufficient enabling principles for any type of agile SE process on any type of project.
• Insightful competency developed among at least two host participants for knowledgeable internal leadership.
• Influence where things are going, compatible with your environment.

Traveling Participants:
• Insightful competency for transformational leadership.
• Bench-mark exposure to HW/SW/FW/WW agile SE processes.

Systems Engineering Community:
• Generic principle-based framework for knowledgably evaluating, choosing, tailoring, integrating, and evolving agile SE.
• Means to address SE dynamics with resilient & composable processes.
• Clarified agile-SE compatibility with 15288 and INCOSE Handbook.
Status


Next

Host identification/scheduling (yours?)

Attention to balance with commercial sector and Europe.

Workshops will occur about 4-8 weeks apart, 4-5/year.

Project Leadership:

• Rick Dove, prior agile-fundamentals workshop series involvement
• Kevin Forsberg, V diagram and INCOSE Handbook involvement
• Bud Lawson, systems engineering text-book involvement
• Jack Ring, prior agile-fundamentals workshop involvement
• Garry Roedler, 15288 involvement
• Bill Schindel, PBSE concept involvement

More information at [www.parshift.com/ASELCM/Home.html](http://www.parshift.com/ASELCM/Home.html)

Ask us to schedule a Webinar to help your organization get involved
References


Abstract

For many, the word Agile, with a capital A, is used as a noun, referring to a family of software development processes based on principles published as the Agile Software Development Manifesto (Fowler 2001). To the INCOSE Agile Systems and Systems Engineering (AS&SE) working group, the word agile has a small a, and is an adjective referring to a capability for operational adaptability in an uncertain and unpredictable evolving environment. Fundamental concepts of agile capability were developed throughout the nineties in projects led by Lehigh University and funded by DoD. See (Dove 2014) for that history. This discussion will review fundamental architecture and design principles that enable agile capability, relate these fundamentals to domain-independent agile systems engineering, and review an INCOSE traveling-workshop series planned for 2015 that will develop a generic agile systems-engineering life cycle model.

We are all very familiar with architectures that accommodate and facilitate real-time structural change. Think of the construction sets we grew up with: Erector/Meccano sets, Tinker Toy, Lego, and other. Each of these construction sets consists of different types of components, with constraints on how these components can be connected and interact. This basic architectural pattern enables reconfiguration, augmentation, and evolution of both the engineering process and the engineered system.

Systems engineering is a disciplined activity that delivers engineered solutions to problems and opportunities – often involving multiple stakeholders, coordination across multiple engineering disciplines, and complexity in both problem and solution. Unlike other engineering disciplines, systems engineering also deals with the social, political, and technical aspects of managing projects that span multiple disciplines.

There is no a priori reason to expect domain specific software development practices to be applicable in domain independent systems engineering. For a simple disconnect example see (Carson 2013). Nevertheless, the ball is in motion toward the goal of a domain-independent agile systems-engineering discipline. Perhaps many different balls are in motion, as the pressure to do systems engineering under accelerating environmental dynamics is not waiting for a common disciplined understanding.

Life cycle, as a systems engineering term, demarcates the progressive maturity of a system through a linear sequence of stages. Here the argument is against the continued notions of non-repeating stages and of single-state existence; instead, a life cycle framework that employs progressively concurrent repeated stages, with diminishing emphasis on the lower stages as maturity through primary stages progresses.

It is time to develop an agile systems-engineering life cycle model. This model, if a single one is sufficient, must take into account four or five types of SE projects: discovery (verifying requirements and solution feasibility), programmatic (Systems and SoS from proven components), approach (e.g., ICSM methodology and product line architecture, Boehm 2014), quick reaction (rapid development and fielding), and evolving (continuous change of system operational viability and opportunity, rapid sequential generations).

The INCOSE AS&SE working group project will develop an agile systems-engineering life cycle model, with guidance from the ISO/IEC 15288 Standard for Systems and Software Engineering; identifying fundamental principle-based activities and processes that provide the agility observed in case studies. This model will justify the application of these principles, activities, and processes by identifying common systems-engineering environmental situations in need of agile response capability across a variety of systems engineering domains and project types.

A method called Realsearch (Dove 1998), so called because it employs real people solving real problems in real time, refined and socialized the original agile system fundamentals discovered and organized in the nineties. It is a process of traveling, structured, collaborative workshops where participants visit host sights by invitation. The process engages first in a collaborative exercise of situation analysis on local examples of agile process, then engages in collaborative identification of principles employed locally that enable agile capability, and finally engages in an exercise that applies learnings to an open problem in need of an agile process solution. A series of such workshops begins in 2015, designed to converge on a fundamental agile systems-engineering life cycle model applicable to the INCOSE systems engineering community.