Balancing Agility and Discipline: Bridging the Gaps Between Software and Systems Engineering

Dr. Richard Turner
Stevens Institute of Technology
rturner@stevens.edu

Pacific Northwest Software Quality Conference
Portland, OR • October 20, 2014
The Systems Engineering Research Center

The SERC is a US Department of Defense University-Affiliated Research Center (UARC)

The networked national resource to further systems research and its impact on issues of national and global significance

The systems research and impact network
SERC Collaborating Universities
A Balancing Act to Support a Bridge

The Problem
Fundamentals
Balancing as Bridge
An Evolving Framework
Admiring the View

Photo: Andrew Hall, PortlandBridges.com

© 2014 Richard Turner
The Current (and Future) Environment

"THE WORLD IS CHANGED..."
## Some software-intensive system trends

<table>
<thead>
<tr>
<th>Traditional Development</th>
<th>Current/Future Trends</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standalone systems</td>
<td>Everything connected</td>
</tr>
<tr>
<td>Stable requirements</td>
<td>Rapid requirements change</td>
</tr>
<tr>
<td>Rqts. determine capabilities</td>
<td>COTS capabilities determine rqts.</td>
</tr>
<tr>
<td>Control over evolution</td>
<td>No control over COTS evolution</td>
</tr>
<tr>
<td>Enough time to keep stable</td>
<td>Ever-decreasing cycle times</td>
</tr>
<tr>
<td>Stable jobs</td>
<td>Outsourced jobs</td>
</tr>
<tr>
<td>Failures not critical</td>
<td>Failures critical</td>
</tr>
<tr>
<td>Reductionist systems</td>
<td>Complex, adaptive, emergent SOSs</td>
</tr>
<tr>
<td>Repeatability-oriented process, maturity models</td>
<td>Adaptive process models</td>
</tr>
</tbody>
</table>

© 2014 Richard Turner
Short history of SE

1930’s   General Systems Theory (1930’s)
1940’s   Term SE coined at Bell Labs
1950’s-60’s-70’s  SE’s heyday (NASA, etc.)
(80’s-00’s) Vee-model, large non-space systems, modeling
Short history of SwE

1950’s Thesis: SW Engineering Is Like HW Engineering

1960’s Antithesis: Software Crafting

1970’s Synthesis and Antithesis: Waterfall Processes

1980’s Synthesis: Productivity and Scalability

1990’s Antithesis: Concurrent vs. Sequential Processes

2000’s Antithesis and Partial Synthesis: Agility, Value, Lean

2010’s Thesis: Ubiquity and temporality?

Adapted from Boehm, “A View of 20th and 21st Century Software Engineering,” ICSE ’06 Keynote
Comparing Systems and Software Engineering

**Similarities (Siblings?)**
- Youth (age?)
- Identity
- Practicality
- Respect (not!)

**Differences (Rivalry?)**
- Education
- Vocabulary
- Values
- Mental Models

Fundamental Concepts

\[
\left(\frac{-\hbar^2}{2m} \nabla^2 + V\right)\psi = i\hbar \frac{\partial \psi}{\partial t}
\]

\[
\Delta x_i \Delta p_i \geq \frac{\hbar}{2}
\]
Boehm’s Fundamental System Success Theorem

A system will succeed if and only if it makes winners of its success-critical stakeholders

Proof of “if”:
- Everyone significant is a winner.
- Nobody significant is left to complain.

Proof of “only if”:
- Nobody wants to lose.
- Prospective losers will refuse to participate or will counterattack.
- The usual result is lose-lose.
Boehm’s System Success Realization Theorem

Making winners of your success-critical stakeholders requires

- Identifying all of the success-critical stakeholders
- Understanding how each stakeholder wants to win
- Having the success-critical stakeholders negotiate among themselves a win-win set of product and process plans
- Controlling progress toward the negotiated win-win realization, including adapting it to change
Fundamentals of Discipline

- Training
- Planning
- Architectures
- Basic Skills
- Infrastructure
- Experience
- Practice principles
- Alignment
- Quality

© 2014 Richard Turner
Fundamentals of Agility...

Flow

Improvisation

Creativity on the Fly

Adaptation

Insight

© 2014 Richard Turner
The Conundrum as of 2003

Traditionalists (CMMI, PM-BoK, SysE)

- Prefer stability, consistency, control, and stepwise refinement; no work is lost, revisited, or repeated.
- Designed to manage cost and schedule as well as engineering
- Big system focus, thus BDUF is expected
- Overly process-bound (MIL-STD-499, MIL-STD-1521)
- Often equated to systems engineering – a hold-over from the 60s and 70s (NASA SE successes), but now part of the problem

Agilists (XP, Scrum, Crystal)

- Handles rapid change/increased complexity
- Handles emergent, vague, volatile requirements
- Avoids late integration and validation problems
- Little or no documentation
- Seen as hacking or sloppy
- Rapid development cycles and refactoring confuse CM and QA folks
The Conundrum as of 2003

Stereotypical Systems Engineers

Geeky white men
(Note pocket protector)

Stereotypical SW Engineers

Weapon-toting
code monkeys!
More realistic media representatives?
A Decade Later: SW Development

Small teams are best; scaling has proven difficult
Poor discipline when implementing agile practices
Agile management (i.e. Scrum) falsely equated with agile dev/ops
SW doesn’t speak business and executive management’s language;
Focus on the small (TDD, MMF) with little understanding of the large
(continuous integration implications at the system level)
Flow and WIP have become more important, but rarely at scale or in
middle management or the executive suite
Wariness toward explicit architectures; “surreptitiously” in SDEs
Size and complexity still growing; legacy SW exploding; backwards
compatibility drives technical debt; SW languages more dangerous
A Decade Later: Systems Engineering

The Vee Model still reigns supreme; seen as essential “bad” waterfall
SE decisions often made without SW input leading to poor SW
Integrating the SW and System life cycles has proven challenging
More stovepipes have exacerbated complexity issues
Software portion of capability continues to increase while life cycle
remains hardware/system focused
Human components in complex systems are (still) unpredictable
No real answer for building/verifying evolving Systems of Systems
Scarcity of qualified systems engineers causes resource bottlenecks
SE seems fixed on creating “agile systems engineering” as a fix
Need for Balance in Both

Acceleration of change expands the cone of uncertainty

Software’s power is critical to systems, but challenges -ilities

Software must expand focus to the system level

SE must reconcile to software values

SE must continue to

• Involve, sound out, and manage a broad spectrum of stakeholders
• Create, maintain, and evolve holistic, essential, and viable concepts of operations, governance models, and architectures
• Resolve technical issues that cross discipline boundaries
• Ensure the –ilities are considered and balanced across the system
## Observations on the components of success

<table>
<thead>
<tr>
<th>Manageable pieces</th>
<th>Efficient and continuous communication</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Evolutionary development</td>
<td>• Architecture in keeping with size</td>
</tr>
<tr>
<td>• Maintain process flow with WIP limits and small batch sizes</td>
<td>• Collaboration is a primary concern</td>
</tr>
<tr>
<td>• Short iterations delivering value</td>
<td>• Multiple communication paths</td>
</tr>
<tr>
<td>• Rolling planning horizon; Reality-based iteration planning</td>
<td>• Continuous integration</td>
</tr>
</tbody>
</table>

### Emphasize human component

<table>
<thead>
<tr>
<th>Efficient and continuous communication</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Architecture in keeping with size</td>
</tr>
<tr>
<td>• Collaboration is a primary concern</td>
</tr>
<tr>
<td>• Multiple communication paths</td>
</tr>
<tr>
<td>• Continuous integration</td>
</tr>
</tbody>
</table>

### Demonstrate progress

<table>
<thead>
<tr>
<th>Efficient and continuous communication</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Quality/performance is everyone’s responsibility</td>
</tr>
<tr>
<td>• Tests are written before any other artifacts (design, code)</td>
</tr>
<tr>
<td>• Capabilities (requirements) are defined by the tests (empirical evidence) that validates them</td>
</tr>
</tbody>
</table>
Key components of a bridge

- Superstructure
- Substructure
The Two Strong Piers of Our Bridge Substructure

Software Engineering Discipline

Systems Engineering Discipline
Many Means of Developing the Superstructure

Creativity
Agility
Values
Leanness
An Illustration of the Need for Balance
A Takeaway from the Collapse

“The Tacoma Narrows bridge failure has given us invaluable information...It has shown [that] every new structure [that] projects into new fields of magnitude involves new problems for the solution of which neither theory nor practical experience furnish an adequate guide. It is then that we must rely largely on judgment and if, as a result, errors, or failures occur, we must accept them as a price for human progress.”

Othmar Ammann, a leading bridge designer and member of the Federal Works Agency Commission investigating the collapse of the Tacoma Narrows Bridge

For software, I believe this supports the idea that the discipline of traditional analysis, architecture, and design must be balanced by agility and it’s learning and feedback to rapidly identify risks and errors and correct them as early as possible. We have the advantage that software as a development medium allows us to do this, and to not take advantage of this ability is malpractice.
A New Framework for Balancing: The Incremental Commitment Spiral Model

Cumulative Level of Understanding, Product and Process Detail (Risk-Driven)

RISK-BASED STAKEHOLDER COMMITMENT REVIEW POINTS:
Opportunities to proceed, merge phases, backtrack or terminate

Risk-Based Decisions
Acceptable
Too High, Unaddressable
High, But Addressable
Negligible

Evidence-Based Review Content
- A first-class deliverable
- Independent expert review
- Shortfalls are uncertainties and risks

© 2014 Richard Turner
What is the ICSM?

Risk-driven framework for determining and evolving best-fit system life-cycle processes

Integrates the strengths of phased and risk-driven spiral process models

Synthesizes together principles critical to successful system development

- Stakeholder value-based guidance
- Incremental commitment and accountability
- Concurrent multidiscipline engineering
- Evidence and risk-driven decisions

Principles used by 60-80% of CrossTalk Top-5 projects, 2002-2005
ICSM Nature and Origins

Integrates hardware, software, and human factors elements of systems life cycle

- Concurrent exploration of needs and opportunities
- Concurrent engineering of hardware, software, human aspects
- Concurrency stabilized via anchor point milestones

Developed in response to a variety of issues

- Clarify “spiral development” usage
  - Initial phased version (2005)
- Provide framework for human-systems integration

Integrates strengths of current process models

- But not their weaknesses

Facilitates transition from existing practices

Incremental Commitment in Gambling

Total Commitment: Roulette
- Put your chips on a number
  - E.g., a value of a key performance parameter
- Wait and see if you win or lose

Incremental Commitment: Poker, Blackjack
- Put some chips in
- See your cards, some of others’ cards
- Decide whether, how much to commit to proceed
ICSM Phased View

<table>
<thead>
<tr>
<th>ICM Anchor Points / DoD Milestones</th>
<th>ECR/MDP</th>
<th>VCR/MDD</th>
<th>FCR/A</th>
<th>DCR/B1</th>
<th>Development &amp; Operations</th>
<th>Anchor Point Milestones</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage I: Incremental Definition</td>
<td>Exploration / Need and Opportunity</td>
<td>Initial Scoping</td>
<td>Concept Definition</td>
<td>System Life-Cycle Architecture and Operations</td>
<td>Increment 1 Development</td>
<td>Operations</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Increment 2 Foundations Baseline</td>
<td>Development</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Increment 3 Foundations Baseline</td>
<td>Operations</td>
</tr>
<tr>
<td>ICM Lifecycle Phases / DoD Phases</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Operations</td>
<td></td>
</tr>
<tr>
<td>Activities</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Operations</td>
<td></td>
</tr>
<tr>
<td>Concurrent risk-and-opportunity-driven growth of system understanding and definition</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Operations</td>
<td></td>
</tr>
<tr>
<td>Evaluation of evidence of feasibility to proceed</td>
<td>Feasibility Evidence</td>
<td></td>
<td></td>
<td></td>
<td>Operations</td>
<td></td>
</tr>
<tr>
<td>Stakeholder review and commitment</td>
<td>High, but addressable</td>
<td>Acceptable</td>
<td>Risk 7</td>
<td>Risk 7</td>
<td>Risk 7</td>
<td>Risk 7</td>
</tr>
<tr>
<td>MOP = Materiel Decision Preparation</td>
<td>ECR = Exploration Committee Review</td>
<td>VCR = Valuation Committee Review</td>
<td>FCR = Foundations Committee Review</td>
<td>DCR = Development Committee Review</td>
<td>OCR = Operations Committee Review</td>
<td></td>
</tr>
<tr>
<td>AoA = Analysis of Alternatives</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Operations</td>
<td></td>
</tr>
<tr>
<td>CDD = Capability Development Document</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Operations</td>
<td></td>
</tr>
</tbody>
</table>

Synchronize, stabilize concurrency via FEDs

Risk patterns determine life cycle process

© 2014 Richard Turner
Different Risk Patterns Yield Different Processes
Anchor Point Feasibility Evidence Descriptions

Evidence provided by developer and validated by independent experts that:

If the system is built to the specified architecture, it will

• Satisfy the requirements: capability, interfaces, level of service, and evolution
• Support the operational concept
• Be buildable within the budgets and schedules in the plan
• Generate a viable return on investment
• Generate satisfactory outcomes for all of the success-critical stakeholders

All major risks resolved or covered by risk management plans

Serves as basis for stakeholders’ commitment to proceed

Synchronizes and stabilizes concurrent activities

Can be used to strengthen current schedule- or event-based reviews
Risk-Driven Scalable Spiral Model: Increment View for each level of systems-of-interest

- **Unforeseeable Change (Adapt)**
  - Rapid Change
  - Agile Rebaselining for Future Increments
  - Deferrals
  - Short, Stabilized Development of Increment N
  - Artifacts
  - Verification and Validation (V&V) of Increment N
  - Future Increment Baselines
  - Increment N Transition/Operations and Maintenance
  - Future V&V Resources

- **Foreseeable Change (Plan)**
  - Short Development Increments
  - Stable Development Increments
  - Current V&V Resources
  - Continuous V&V

- **High Assurance**

© 2014 Richard Turner
Reality for Larger, Complex Systems
The ICSM Principles

Stakeholder value-based guidance

Incremental commitment and accountability

Concurrent system engineering

Evidence and risk-driven decisions

Good example: Symbiq Medical Infusion Pump

Counter example: Healthcare.gov
Example ICSM Commercial Application: Symbiq Medical Infusion Pump

Winner of 2006 HFES Best New Design Award
Described in NRC HSI Report, Chapter 5
Symbiq IV Pump ICSM Process - I

Exploration Phase

- Stakeholder needs interviews, field observations
- Initial user interface prototypes
- Competitive analysis, system scoping
- Commitment to proceed

Valuation Phase

- Feature analysis and prioritization
- Display vendor option prototyping and analysis
- Top-level life cycle plan, business case analysis
- Safety and business risk assessment
- Commitment to proceed while addressing risks
Symbiq IV Pump ICSM Process - II

Foundations Phase

• Modularity of pumping channels
• Safety feature and alarms prototyping and iteration
• Programmable therapy types, touchscreen analysis
• Failure modes and effects analyses (FMEAs)
• Prototype usage in teaching hospital
• Commitment to proceed into development

Development Phase

• Extensive usability criteria and testing
• Iterated FMEAs and safety analyses
• Patient-simulator testing; adaptation to concerns
• Commitment to production and business plans
Ways that ICSM Might Have Helped healthcare.gov
Healthcare.gov and the 4 principles

Stakeholder Value-Based Guidelines

- System stakeholders, numerous and powerful, found it difficult to reconcile win conditions and success models.
- Stakeholder win conditions were not negotiated, documented, and continuously stabilized to ensure that the system would remain acceptable.
- There were no shared understanding of values, assumptions, and expectations; this allowed degeneration into blame assignment after the fact.

Incremental Commitment and Accountability

- The complexity of the system (>5000 pages in ACA and >78,000 health plans) made the probability of ambiguous and conflicting requirements extremely high.
- Top-level HHS leadership had no significant experience in overseeing a systems development effort like healthcare.gov and little understanding of commitments.
- The development was set up with an all-or-nothing delivery date, resulting in a low probability of success given the complexity and visibility of the system.
- Expectations were initially set extremely high and never managed with respect to the reality of the developing system.
Healthcare.gov and the 4 principles

Concurrent Multidisciplinary Engineering

• Concurrency is fundamental to roll out a large system in a relatively short time frame.

• Adequate expertise in human factors, insurance processing, architecture, and cognitive science was lacking.

• Requisite supply chain communications and engineering process adaptability were absent before contracting $600M of work to 55 companies.

Evidence- and Risk-Based Decisions

• It is doubtful that validated evidence of the system capabilities meeting the anchor-point milestone criteria was presented to any authority by the developers in the last year of development.

• Decisions were made for political reasons rather than technical reasons, and the technical ramifications were not fully understood.

• Schedule was a key driver, yet it appears that there was no prioritization of key requirements and features to drop requirements or incrementally roll out capability.

• The risks associated with the magnitude and visibility of the site should have led to a collaborative, incremental development strategy not a single deployment.
Admiring the View

There is a clear need for a bridge…
It needs to reach out from both directions…
It can build on deep, experienced discipline…
It must adopt a more holistic and flexible framework…
Guidance exists to build it…
There are interested parties to invest in it…
There are myriads of workers beginning to work on it…
The crossing traffic can only increase, so the bridge must be ready to grow…
For more information….

Tutorial on Wednesday and/or the book:

*The Incremental Commitment Spiral Model: Principles and Practices for Successful Systems and Software*

- Available from Amazon, InformIT, all the usual suspects….

Co-authors:

Barry Boehm
- boehm@usc.edu

Jo Ann Lane
- jolane@usc.edu

Sue Koolmanojwong
- koolmano@usc.edu
Questions?