SAND2012-2039 C

SAND2012-2039C

Understanding Agile Concepts and the Role of QA

World Conference on Quality and Improvement
Session ISE 10
May 23, 2012 9:30am-10:30am

Dr. David E Peercy depeerc@sandia.gov
Sandia National Laboratories
ASQ Software Division Regional Councilor, Region 14

The Global Voice of Quality™

Topics

- Understanding Agile Concepts
  - Agile Background
  - Agile Manifesto Values and Principles
  - Some Thoughts/Conclusions from a nuclear weapons study
- ASQ Software Division’s Agile Position Statement
- Quality (Assurance) in an Agile Environment

Agile "Approach/Method" Background

- The "Agile" approach to software engineering
  - Agile Alliance (http://www.agilealliance.org/) in 2001
  - publication of the Agile Manifesto
  - a discipline of developing software where the practices adhere to the Agile Manifesto Values and its Principles
- Agile/agile methods are described[1] as being
  - lightweight processes with short iterative cycles
  - actively involve users to establish, prioritize, and verify requirements
  - relies on tacit knowledge within a team as opposed to documentation
  - methods are an outgrowth of rapid prototyping and rapid development experiences
  - problem of change is exacerbated by long development cycles
- Some thoughts – what about?
  - applications with significant security/safety concerns – such as medical devices, commercial aircraft, nuclear weapons?
  - balance between the application of Agile concepts and regulation practices?
  - Agile concepts and practices integrated into a software engineering life cycle?

[1]Boehm, B., Turner, R., Balancing Agility and Discipline ,Addison-Wesley, 2004

Agile Values

V1: Individuals and interactions are valued over processes and tools.

This does not mean processes and tools should not be valued

V2: Working software is valued over comprehensive documentation.

This does not mean documentation should not be valued

V3: Customer collaboration is valued over contract negotiation.

This does not mean contracts and appropriate negotiation should not be valued

V4: Responding to change is valued over following a plan.

This does not mean plans should not be valued

ASQ

Agile Principles

P1: Customer is highest priority – early and continuous delivery of valuable software

P2: Welcome changing requirements, even late in development

P3: Deliver working software frequently

P4: Business people and developers must work together daily throughout the project

P5: Build projects around motivated individuals, good environment and support, trust

P6: Face-to-face conversation is the best communication method

Agile Principles

P7: Working software is the primary measure of progress

P8: Agile processes promote sustainable development; the sponsors, developers, users should be able to maintain a constant pace indefinitely

P9: Continuous attention to technical excellence and good design enhances agility

P10: Simplicity--the art of maximizing the amount of work not done--is essential

P11: The best architectures, requirements, and designs emerge from self-organizing teams

P12: At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly

Agile Method Examples[1]

SQAS31.01.00-2007

	Applicability of Agile Method Practices to Weapon/Weapon-Related Software Projects
Agile Method	**Overall Assessment of Applicability**
Scrum	**Limited:** • some project team interaction concepts are applicable • does not directly address software development / support methods
Adaptive Software Development (ASD)	**Limited:** • very little evidence of use • targeted domain is e-business • might be useful for research software and/or concept development, but not for production software
Lean Development (LD)	**Limited:** • good evidence of manufacturing project management success • does not directly address software development / support • proprietary aspects may limit use
Crystal	**Medium to High:** • some verification activities would need enhancement • critical systems would need to add specialized practices • Crystal Orange for production • Crystal Clear for research
eXtreme Programming (XP)	**Limited:** • the almost total focus on frequent delivery, and the activities around frequent delivery (e.g., co-location of customer and team) make XP limited in its use • might be used in research and concept efforts, but the co-location aspect is still not viable
Dynamic Systems Development Method (DSDM)	**High:** • provides detailed practices that can be generally tailored for use in weapon/weapon-related software projects • flexible application of principles makes this method applicable
Feature Driven Development (FDD)	**Limited:** • could be integrated within a more comprehensive life cycle approach • probably more useful in prototyping and research applications

[1]SQAS31.01.00 Report, "Applying Agile Methods to Weapon/Weapon-Related Software", Nuclear Weapons Complex, Software Quality Assurance Subcommittee, April, 2007.

ASQ

SQAS Study Conclusions[1]

C1: Philosophical – but some applicability
Most agile methods reviewed provide more of a philosophical and project management approach than a software development approach; with only a few exceptions, these approaches have some applicability to weapon/weapon-related software development and/or support

C2: Significant Variation – need for a standard definition
There is significant variation in the capabilities provided by the various agile methods; this supports the need for a standard such as IEEE 1648 to provide better guidance as to what methods can legitimately be called "agile"

C3: Critical Software Not Addressed – but practices can be integrated
None of the agile methods directly address practices that might be needed for development of software for critical applications, although such practices might be integrated as needed

[1]SQAS31.01.00 Report, "Applying Agile Methods to Weapon/Weapon-Related Software", Nuclear Weapons Complex, Software Quality Assurance Subcommittee, April, 2007.

ASQ

SQAS Study Conclusions

C4: Non-applicable Methods Use Non-applicable Principles

The agile method practices that are most non-applicable to weapon/weapon-related software are the same ones that support certain aspects of the principles that are either limited or non-applicable

C5: Require Good People

All agile methods reviewed appear to require (or at least need) motivated, creative, and talented software developers for success; it might be reasoned that all projects would be more successful with this baseline requirement

C6: Crystal and Dynamic Systems Development Method Best

Of the agile methods reviewed, the most applicable for weapon/weapon-related use appear to be Crystal and Dynamic Systems Development Method; other methods have more limited application

C7: Methods Most Applicable to Prototype/Concept Phase

The agile methods reviewed all appear to have some reasonable applicability to research and prototype/concept development phases of weapon/weapon-related software

ASQ Position Statement Background

ASQ_Position_Statement_Background

- **August 2009 Project Formation**
  - ASQ Software Division convened a small group of ASQ software experts in 2009 to develop a position statement concerning the use of agile practices for software projects
- **May 2010 Draft Approval**
  - Draft statement completed and submitted for approval at the WCQI in St. Louis, May 24-26, 2010. Division Management Council approved the draft and recommended that the position statement be shared with the members of the ASQ Software Division for further comment before removing the draft status
- **2nd Q 2010 Draft Publication**
  - ASQ Software Division was invited to comment on the statement through LinkedIn. Draft statement was published in the ASQ Software Division Newsletter Software Quality Live, 2nd Quarter 2010.
  - Comments ranged in detail and from simple "really like it" to "no valuable content". Minor constructive changes were incorporated but essentially did not change the basic statement.
- **2011 June - Final Published Version**
  - Software Quality Professional, Vol 13 No. 3, June 2011, pp 39-40

ASQ Position Statement Summary

Position Statement provides three propositions and a short section on Rationale for those three propositions.

SUMMARY

"A preferred approach is to incorporate practices based on stakeholder needs and expectations, project properties, methods for conducting the project work, and the value such methods provide to the effectiveness and efficiency of the project's properties within the context of system project success."

The ASQ position on the use of Agile techniques for software development is that they are the preferred practices whenever they provide the best overall value for all stakeholders.

SQP-ASQ_Stateme
t_Vol13_No3_June201

ASQ Position Statement Propositions

Proposition 1

- **Stakeholders Interests are of Value**
    - A preferred approach for a software project is one the stakeholders feel supports their interests.
- **Stakeholders Are Those with Benefits or Cost**
    - Stakeholders include any entity receiving benefits or incurring costs from a software project.
- **Stakeholders Interests Need Balance**
    - There are competing interests among stakeholders, and a balance of those interests is critical in selecting software engineering practices for a software project.

ASQ Position Statement Propositions

Proposition 2

- **Values are Preferred, but Preferences are Not Excused**
  - The Agile Manifesto prefers one value over another, but it does not excuse a project from appropriately addressing both sides of the balance to support the best interests of the project and its stakeholders.

- **Values and Principles that Balance Stakeholder’s Interests are Valued**
  - As such values and principles provide guidance for balancing each stakeholder’s interests, they are considered of value to the software project. http://www.agilemanifesto.org/

ASQ Position Statement Propositions

Proposition 3

- **RE-Evaluate All Practices Periodically**
  - In the spirit of continual improvement, all software practices should be re-evaluated periodically as to their effectiveness and efficiency.

- **Agile Methods Fit Many Environments**
  - Management and engineering practices will change over time as technologies and expectations change. Agile methods have evolved to include “extreme” implementations of good practices that may fit many of today’s environments.

- **Some Agile Methods Fit Some Environments Better Than Others**
  - Variants of good practice, including the agile practices, suit some business environments better than others, and may be combined in ways to increase the probability of project success.

- **Agile Manifesto Philosophies Can Be Useful for Improvement**
  - Philosophies such as those stated in the Agile Manifesto, as well as those underlying other methodologies, can be useful in supporting an organization’s ongoing improvement efforts.

Role of QA Within Agile Context

Quality (Assurance) in an Agile Environment

sduncan@computer.org

with

PS - Integration of ASQ Position Statement (PS) Thoughts/Discussion Points

First, Some Definitions (ASQ)

- **QA**: “The planned and systematic activities implemented in a quality system so that quality requirements for a product or service will be fulfilled.”
- **QC**: “The observation techniques and activities used to fulfill requirements for quality.”

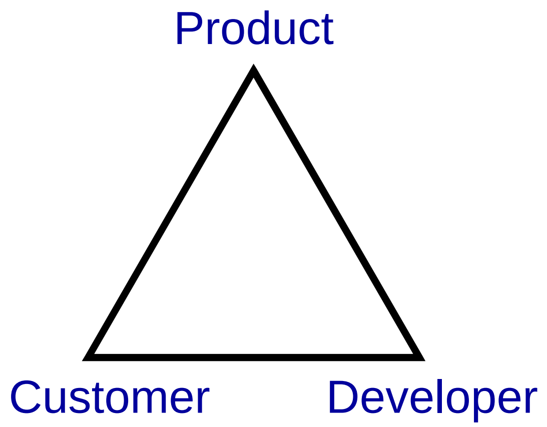
Generally

- **QA** focus is on preventing defects
- **QC** focus is on finding and correcting defects

Prevention (QA) vs **Correction (QC)**

Quality in an Agile Environment

- QA & QC are intertwined and performed so iteratively and frequently, it is as if they are one
- **The essence of all Agile activity is**
  - Expanding the bandwidth & frequency of communication through direct, continuous feedback
  - Being open to & prepared for change by facilitating, not avoiding, it



PS - communication is not only within the developer team but with the customer as well – are there situations/timing where transparency might not be the best solution?

And, how does/does not Agile facilitate sustained change in a product – such as during maintenance/support?

Essential Agile Quality Topics

• Collaboration	- *not just professional cooperation*
• Planning	- *time-boxed estimation*
• Commitment	- *under commit, over deliver*
• Communication	- *most powerful face-to-face*
• Development	- *short, easily tested episodes*
• Delivery	- *production-ready results*
• Reflection	- *continuous improvement*

Done *early, frequently,* with a

near-term emphasis

PS – discuss “delivery” and what that might mean to a customer, good and bad?

Collaboration

Not just professional cooperation

- Trusting working relationships
- Offer/ask for help without reservation
- Shared commitment and responsibility
- Developers and testers work together daily
- Understand one another’s work

PS – great concepts –
are there any potential issues/concerns?

ASQ

Planning

Time-boxed, near-term emphasis

- Short work “episodes”
- Easily verified tasks/results
- Less detail until as much is known as can be at the “last responsible moment”
- Testing an integral part of each day’s work
- Everyone can support “test”

PS – interesting concepts –
are there any potential issues/concerns?

Commitment

Under commit, over deliver

- Trust, built on commitments met
- Commitment, built on achievement
- Achievement, built on realistic assessment
- Assessment, built on visibility
- Take on more when confident about meeting existing commitments

PS – how to balance customer schedule and contractual commitments with the concept of under commitment and builds based on known estimates and realistic assessments?

What is the "basis" for quality estimates?

Communication

Most powerful face-to-face

- Fastest way to clarify intent
- Easiest way to assess feedback
- Clearest way to confirm understanding
- Lowest level of overhead
- Encourage "sustainable" documentation

PS – what situations might prevent face-to-face and what could one do to still achieve the intent of this communication principle?

What does "sustainable" mean?

ASQ

Development

Short, easily-tested episodes

- Iterative happens in the head, not on the calendar
- Risk = time until code is “done”
- Critical practices
  - ➢ Test-driven design
  - ➢ Continuous integration
  - ➢ Automated regression testing
  - ➢ Pairing (or something like it)

PS – quality principle is to find defects early to prevent propagation to later stages – learn by doing!

Is that what is being suggested here?

Delivery

Production-ready results

- Definition of “done” is a must
- Done = it *could* go into production (but it might not)
- All acceptance criteria met (so there has to be some)
- View each iteration as a “delivery”
- What “value” has the iteration produced?

PS – what is meant by “value” in this case? How does this “value” relate to the agile values?

What are some potential pitfalls related to the definition of “done”?

And, of course, back to the question of what “delivery” means?

ASQ

Reflection

Continuous improvement

- Agile's approach to *Kaizen*[1]
- Whole team contributes
- Keep, change, add, learn
- Prioritize and estimate
- Better quality, clearer commitments, greater visibility, higher trust & satisfaction
- Management's role: do the same

[1]Kaizen Philosophy: Continuous Incremental Improvements
Kaizen Method Foundation Elements:

1. Teamwork
2. Personal Discipline
3. Improved Morale
4. Quality Circles
5. Suggestions for improvement

PS – how do "lessons learned" work in your organizations?

Values & (condensed) Principles

Individuals & Interactions

Working Software

- [1]Progress = working software, not documentation
- Work together daily, employing face-to-face conversation
- Satisfy the customer and welcome change
- [2]Deliver working software early, continuously & frequently
- Become more effective by tuning & adjusting behavior to produce a sustainable, constant pace
- Build self-organizing teams from motivated individuals, who trust & are trusted, within an effective, supported environment
- [3]Simplicity & good design based on technical excellence

Customer collaboration

[1] what is minimally required?
[2] is this always possible?
[3] well of course!

Responding to change

Q&A

PS

As usual, the question isn’t whether this all sounds good. Clearly, it does!

It is mostly what we expect of a team-oriented project – but, what precise measures of success/continual improvement can we possibly define?

And how are “values” actually determined – and for whom?

Any ideas?

Measuring "Agility"

- **Measuring agility (from Wikipedia Reference:** http://en.wikipedia.org/wiki/Agile_software_development**)**
- While agility can be seen as a means to an end, a number of approaches have been proposed to quantify agility. *Agility Index Measurements* (AIM)[18] score projects against a number of agility factors to achieve a total. The similarly named *Agility Measurement Index*,[19] scores developments against five dimensions of a software project (duration, risk, novelty, effort, and interaction). Other techniques are based on measurable goals.[20] Another study using fuzzy mathematics[21] has suggested that project velocity can be used as a metric of agility. There are agile self-assessments to determine whether a team is using agile practices (Nokia test,[22] Karlskrona test,[23] 42 points test[24]).
- While such approaches have been proposed to measure agility, the practical application of such metrics has yet to be seen.
- Historically, there is a lack of data on agile projects that failed to produce good results. Studies can be found that report poor projects due to a deficient implementation of an agile method, or methods, but none where it was felt that they were executed properly and failed to deliver on its promise. "This may be a result of a reluctance to publish papers on unsuccessful projects, or it may in fact be an indication that, when implemented correctly, Agile Methods work." [25] However, there is agile software development ROI data available from the DACS ROI Dashboard. [26]

(18)"David Bock's Weblog : Weblog". Jroller.com. http://jroller.com/page/bokmann?entry=improving_your_processes_aim_high. Retrieved 2 April 2010.
(19) "Agility measurement index". Doi.acm.org. http://doi.acm.org/10.1145/1185448.1185509. Retrieved 2 April 2010.
(20) Peter Lappo; Henry C.T. Andrew. "Assessing Agility". http://www.smr.co.uk Retrieved 6 June 2010.
(21) Kurian, Tisni (2006). Agility Metrics: A Quantitative Fuzzy Based Approach for Measuring Agility of a Software Process, *ISAM-Proceedings of International Conference on Agile Manufacturing'06(ICAM-2006)*, Norfolk, U.S.
(22) Joe Little (2 December 2007). "Nokia test, A scrum-specific test". Agileconsortium.blogspot.com. http://agileconsortium.blogspot.com/2007/12/nokia-test.html. Retrieved 6 June 2010.
(23) Mark Seuffert, Piratson Technologies, Sweden. "Karlskrona test, A generic agile adoption test". Piratson.se. http://www.piratson.se/archive/Agile_Karlskrona_Test.html. Retrieved 6 June 2010.
(24) "How agile are you, a scrum-specific test". Agile-software-development.com. http://www.agile-software-development.com/2008/01/how-agile-are-you-take-this-42-point.html. Retrieved 6 June 2010.
(25) David Cohen, Mikael Lindvall, Patricia Costa "Agile Software Development", *Data & Analysis Center for Software*, January 2003

Other References of Some Interest

1) Abrahamsson, P., Salo, O., Ronkainen, J., Warsta, J., "Agile Software Development Methods: Review and Analysis," VRR Publications 478, Espoo 2002.
2) Agile Alliance: **http://www.agilealliance.org/**
3) Agile Manifesto, 2001, **http://www.agilemanifesto.org/**
4) Ambler, Scott, Agile Modeling and eXtreme Programming (XP) **www.agilemodeling.com**
5) Boehm, B., Turner, R., Balancing Agility and Discipline ,Addison-Wesley, 2004Clements, Paul, Ivers, James, Little, Reed Nord, Robert Stafford, Judith,"Documenting Software Architectures in an Agile World" **http://www.sei.cmu.edu/publications/documents/03.reports/03tn023.html**
6) CMMI, "Capability Maturity Model Integration," Version 1.1, CMU/SEI-2002-TR-028(continuous) and CMU/SEI-2002-TR-029 (staged), Software Engineering Institute. **http://c2.com/cgi/wiki?CapabilityMaturityModel**
7) IEEE 1648, "Recommended Practice for the Introduction and Use of Agile Software Development Methods," draft, April 2006.
8) Ornburn, Steve, Kane, David, "Anti-Matter and Matter or Reconcilable Differences?" **http://www.sstc-online.org/proceedings/2002/SpkrPDFS/WedTracs/p752.pdf**
9) Paulk, Mark, Website for several articles on Agile Concepts: **http://www.cs.cmu.edu/~mcp/**
10) Pettichord, Bret, "Testers Should Embrace Agile Programming" **http://www.io.com/~wazmo/papers/embrace_agile_programming.html**
11) Radding, Alan, "Extremely Agile Programming: Agile Programming Systems" **http://www.computerworld.com/softwaretopics/software/appdev/story/0,10801,67950,00.html**
12) Software Quality Professional, "ASQ Software Division Position Statement on Agile Methods for Software Projects," American Society for Quality (ASQ), Vol 13 No. 3, June 2011, pp 39-40 **http://asq.org/pub/sqp/**
13) SQAS31.01.00-2007, "Applying Agile Methods to Weapon/Weapon-Related Software," Nuclear Weapons Complex Software Quality Assurance Subcommittee, April 2007.