The Agile Enterprise Architect

Working effectively with Sprint Teams, Backlogs and Scrum Masters

Guy B. Sereff

20 November 2013

About The Presenter

Guy B. Sereff

- Author, Speaker and Technology Practitioner
- Vice President / Enterprise Architecture
- Technology Industry Experience
  - *Application Research & Development (12 years)*
  - *Large-Scale Technology Management (8 years)*
  - *Global Enterprise Architecture (7 years)*
- Enterprise Architecture Domain Experience
  - *Business Architecture*
  - *Information Architecture*
  - *Application Architecture*
  - *Solution Architecture*
  - *Architecture Governance*
- Pragmatic Blend of Strategy and Tactical Execution

http://www.linkedin.com/in/guysereff

Agenda

Agile Conceptual Overview

Challenges to the Agile Enterprise

Becoming an Effective Agile Enterprise Architect

- Study Up on the Use/Misuse of Agile
- Tackle the Hard Stuff
- Plan and Prepare Ahead
- Work on the Front Lines
- Reduce Friction

Recommended Next Steps

Questions and Closing Comments

Agile Conceptual Overview

Organizations have amassed a heterogeneous array of technology gadgets, software packages, utilities and applications over the course of operations

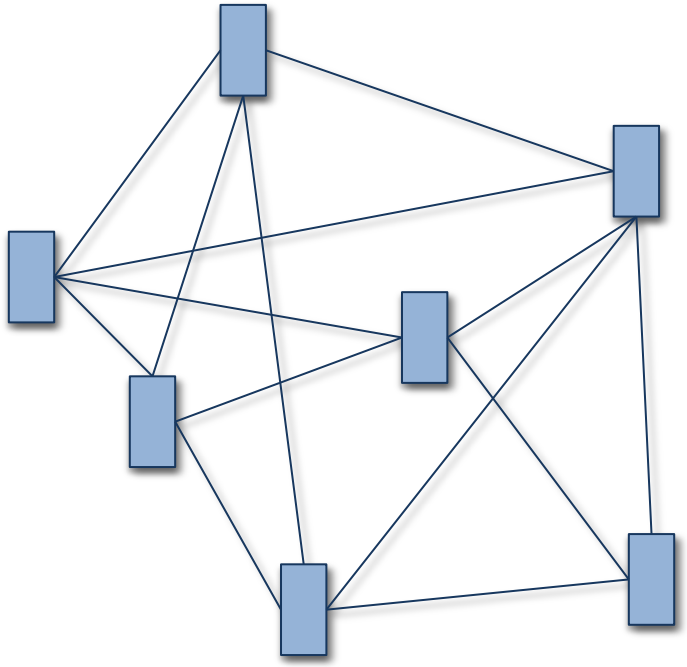
Today's rapid global economic pace makes it very difficult to:

- Stay on top of the vast number of technical components deployed across a firm
- Execute an effective portfolio management strategy
- Rapidly deploy innovative solutions

This collection of technology 'widgets' often represents years of what was, at the time, a thoughtful investment in meeting the immediate needs of the organization

- But what about *today*?
- What about *tomorrow*?

Many organizations have turned to rapid software delivery methods, typically under the banner of 'Agile'



The Agile Manifesto

- The Agile Manifesto was created in 2001 by a group of seventeen leading software engineering methodology practitioners
- Comprised of four simple, yet powerful unifying values statements
- The purpose: significantly improve the software delivery process and to ease the persistent cycle of tension between the community of software consumers and the community of software producers
- Emphasis is on valuable activities with the potential of much better outcomes than traditional methods had previously produced

The Manifesto for Agile Software Development*

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- Individuals and Interactions over processes and tools
- Working Software over comprehensive documentation
- Customer Collaboration over contract negotiation
- Responding to Change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

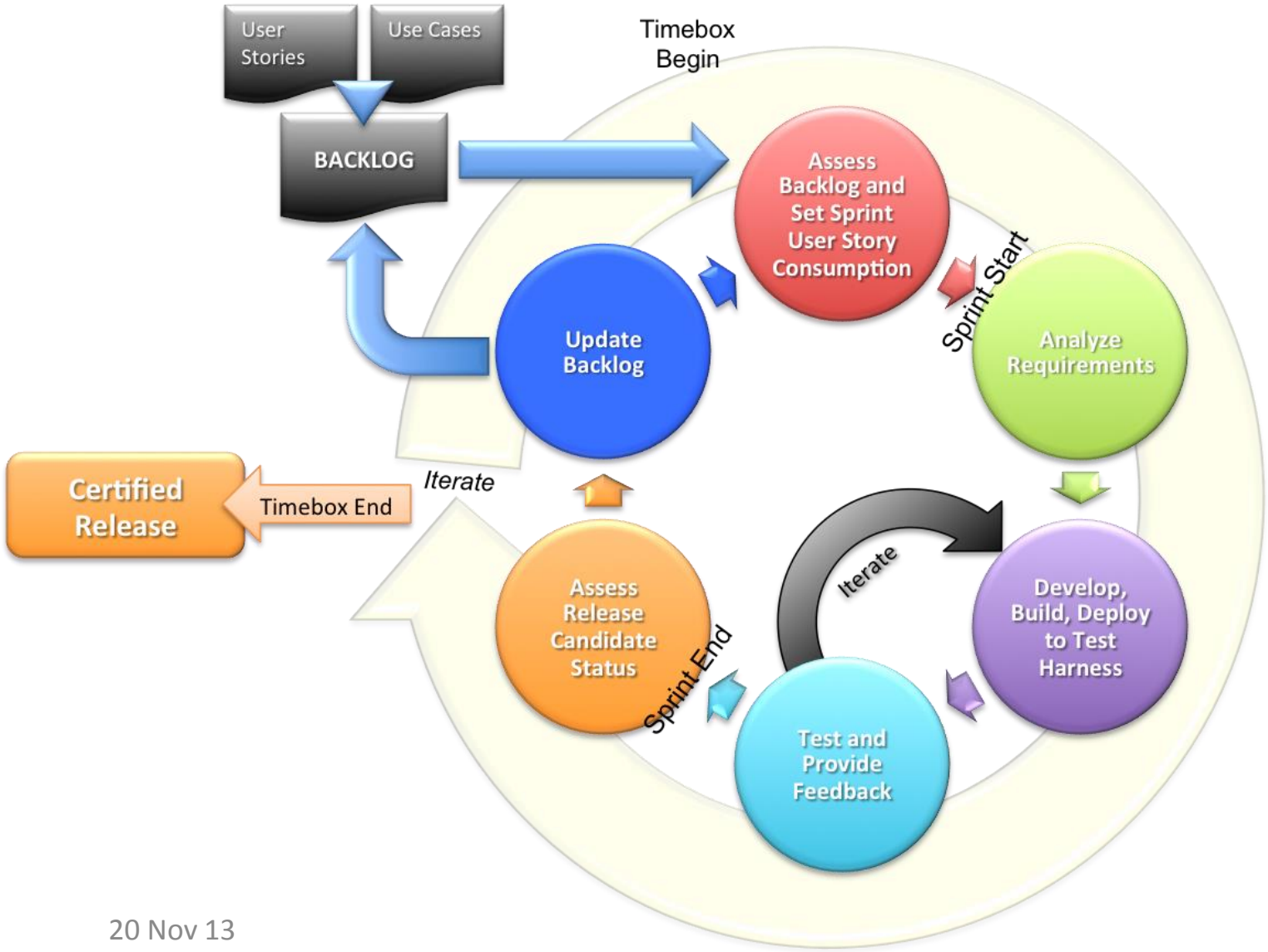
*Beck, Cockburn, Fowler *et al*. (2001). *The Agile Manifesto*.

Agile Software Principles*

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

*The Agile Alliance's Twelve Principles of Agile Software

Generalized Agile Delivery Life Cycle



- Technicians and end-user representatives iterate back and forth between requirement definitions and test results
- Designers and developers iterate back and forth between approach and implementation details
- Numerous sprints are conducted within the overall project timebox, producing some form of a 'working' system at the end of each sprint cycle
- Simultaneous sprint teams can be leveraged, although they often require additional levels of cross-functional platform integration and validation work

Taking the Enterprise Agile

- Once an organization has had some success with Agile, they are often tempted to try it on a much broader scale
- This presents a p
  and complex like
  behave in a nimb
- Common approa
  - Islands of Ag
  - Integrated A
  - Hybrid Agile

For our purposes, we'll define an *Agile Enterprise* as an organization that has been able to successfully implement Agile methods in scale, specifically in terms of software development and solution delivery.

In this context, 'in scale' means that a significant portion of software delivery follows a defined Agile methodology to deliver critical or strategic solutions.

Repeatable patterns of success are evident and spread across more than one part of the organization.

Challenges to Becoming an Agile Enterprise

Four Impeding Organizational Memory Patterns*:

- **Waterfall Thinking**
  Classic cascading project management techniques that are deeply embedding in the organization's psyche and contrary to iterative delivery approach methods
- **Command and Control**
  Managers believe they know best about <u>everything</u> and make dictation from 'on high' regardless of input from the community of individual contributors
- **Commitment to Defying the Laws of Nature**
  Engineers and solution providers yield to the pressure to make promises that they will deliver humanly impossible results...again
- **Hiding Reality**
  Continually communicating an untrue or overly optimistic status in hopes that someone figure something out before the truth gets out and the problems will magically go away

Additional Considerations

- Organizations that lack a culture of transparency or don't tolerate the delivery of bad news well will be disappointed with their ability to adopt Agile in scale
- Hiring a certified *Scrum Master* doesn't make the organization Agile; becoming an Agile Enterprise requires a commitment from top to bottom
- Developers and team members must be ready, willing and able to do 'Agile' and not have it forced on them; Too many 'voices of doom' or 'devil's advocates' around the table will lead to a self-fulfilling prophecy of failure, or at best lackluster results

*Schwaber, *The Enterprise and Scrum*

Agile Enterprise Scalability Gating Factors*

- **Geographic Distribution**
  Colocation is optimal; Where will the team be physically located?
- **Team Size**
  Optimal teams size is ty… big will the teams be ar… be?
- **Regulatory Compliar…**
  Agile teams tend to res… aspects are non-negoti…
- **Domain Complexity**
  Agile complexity grows with the complexity of the domain; How complex are the problems to be solved with Agile?
- **Technical Complexity**
  Agile can be applied to new or legacy platforms of varying … How complex are the … will be involved?
- **…ion**
  …ms can be challenging to …; What is the organizational …ile participants and
- **…ity**
  …gile; What is the current … organization and how hard is
- *Agile takes time to do well and should not move the organization away from its strategic vision; How will fundamental enterprise principles regarding architecture, reuse and strategic alignment be incorporated?*

Focus on Architecture

Ensure that the solution delivered is maintainable, extensible, and high-performing.

Many agile approaches ignore architecture, or assume it can be derived by merely refactoring.

This results in well-structured code but ignores the bigger picture.

Focusing on the architecture is also essential to coordinating multiple teams working together.

Ivar Jacobson International

*Adapted from *Disciplined Agile Development (Ambler & Line)*

Becoming an Effective Agile Enterprise Architect

- *Another Paradox:* Agile teams working in short, tightly wound cycles to deliver working software need architectural guidance, but there's often little time allowed for it...
  ...and often no appetite for it!
- Enterprise Architects can not afford to ignore the growth of Agile within their organization
- Unattended Agile programs risk delivering solutions that are in conflict with the organization's architectural strategy
- Enter the '*Agile Enterprise Architect'* – a proactive, pragmatist who focuses on maintaining the spirit and benefits of Agile while protecting the broader interests of the enterprise are protected

An ***Agile Enterprise Architect*** is an actively engaged Enterprise Architect who effectively guides and influences organizations through the Agile Software Delivery process, providing the appropriate level of design oversight and reference architecture governance without impeding solution delivery velocity or the level of delivered functionality.

Study Up on the Use/Misuse of Agile

Understand the organization's existing Agile Methodology definition, implementation and maturity level

- Gain a full understanding of the organization's Agile Software Delivery methodology
  - Strict adherence to a particular Agile methodology with academic precision
  - Adaptive or hybrid Agile method
  - Obscure set of practices that are Agile in name only
- Understand the current Agile implementation approach and build a solid functional knowledge by observing and engaging in actual Agile sprint activities
  - Avoid disrupting the flow of progress
  - Build a knowledge base and earn credibility in order to influence future behavior
  - Monitor an Agile delivery cycle from start to finish, observing the process, roles and collaboration dynamics

Tackle the Hard Stuff

- Be bold enough to tackle the more difficult issues head on
  - Focus the team's energy on solving the most significant architectural problems first
  - Avoid taking short-sighted architectural 'short-cuts' or delaying fundamental decisions
  - Recognize the exacerbating impact of 'regressive design'
- Convey the downstream impact of critical architectural issues
  - Outline the level of difficulty involved in architectural realignment
  - Distinguish between critical architectural issues and those of lesser impact
  - Be prepared with workable alternatives that work within the constraints of the delivery cadence

Address the challenges of Enterprise Agile head-on within the context and culture of the organization

Plan and Prepare Ahead

Be prepared for rapid deployment ahead of time with reusable artifacts and available components

- Most Enterprise Architecture organizations produce numerous models and artifacts
  - Establish a central repository with the ability to export architecture content in native formats
  - Prepare pre-populated UML models for rapid PIM to PSM transformation
- Establishing over-arching architectural approach
  - Create a Design Framework and identify non-negotiables
  - Clarify decision rights ahead of time
- Provide reusable Service Oriented Architecture (SOA) components based on the Reference Architecture
  - Make services easy to discover through a centralized service repository
  - Publish consistent API's and internal SDKs
  - Support rapid scaling and provisioning
  - Configuration over customization

Work on the Front Lines

- Practice *Architecture by Wandering Around*:
  - Hands-on engagement with the Agile delivery teams
  - Active collaboration and problem solving
- This is more than...

  ...monitoring daily scrum calls while multi-tasking and listening for occasional architectural questions

  ...being on 'stand by', waiting to engage only from a reactive 'as needed' basis
- Become a formal member of the Agile team
  - Own sprint deliverables and become a contributing resource
  - Be accountable to *both* the Agile and Enterprise Architecture stakeholders
- Take the opportunity to be a catalyst for change and build credibility through value creation

Spend time with/as a Solution Architect through an entire Sprint or Delivery Cycle to keep a realistic point of view

Reduce Friction

Provide a means for Low-Friction Reference Architecture Adoption

- Work aggressively to reduce Reference Architecture adoption friction
  - Make adoption easier than perpetuating non-compliant environments
  - Eliminate adoption bottlenecks and chokepoints
  - Eliminate any ambiguity from the Reference Architecture and control the amount of variation
- Leverage first-hand Agile engagement to identify where Enterprise Architecture guidance breaks down
- Focus on providing both *Ready-to-Consume* and *Easy-To-Consume* technical components
- Think Process Engineering
  - Engage Business Process optimization experts
  - Eliminate as many manual steps as possible
  - Leverage industrial strength reengineering approaches such as *Lean Six Sigma*

Potential Benefits

- Better alignment of stakeholder needs across the organization
- Stronger ability to provide architectural influence to Agile delivery teams in real-time rather than at post-activity reviews and checkpoints
- Establishment of best-practices that benefit the Agile community as well as other parts of the Enterprise
- Acceleration of Reference Architecture adoption
- Strengthened Enterprise Architecture credibility and accountability

You've got to think about big things while you're doing small things, so that all the small things go in the right direction.

– Alvin Toffler

Recommended Next Steps

Study up (or brush up) on the concepts of Agile, particularly how it is practiced within your organization today

Look for limitations to Agile scalability at the Enterprise level today and determine its impact on architecture principles

Roll those sleeves up and get engaged in real Agile project work to build an accurate point of reference

Identify Enterprise Architecture process ‘decelerators’ and do what it takes to move the organization from legacy speed to digital speed

Becoming an Agile Enterprise Architect is not easy.

Some of the effort requires process changes that may or may not be hard to adopt, depending on the size of the organization and its ability to change.

The core message is to:

- ***Take a realistic view of what is happening within the organization***
- ***Engage in an impactful way***
- ***Be prepared to remove roadblocks***

Any Questions?

Orbus Software Group

@OrbusSoftware

Download this presentation and accompanying white paper from:
www.orbussoftware.com/downloads