White Paper

Enterprise Architecture and Agile Development, Contradiction or Synergy?

WP0145 | May 2014

Karl Schulmeisters

Karl Schulmeisters is Technology Advisor for the Carver Global Health Group where he provides expert leadership in Business Process Architecture, Enterprise Architecture and Cloud Computing. Karl's current emphasis is on the impact and integration of disruptive technologies into traditional enterprise IT organizations: Cloud, Mobility, Consumerized IT, Machine Learning/Big Data and Social Media.

Karl Schulmeisters is an internationally recognized speaker – his most recent speaking engagement was at the Congress on the Future of Engineering Software in St. Petersburg Russia. He welcomes your comments at *karl.schulmeisters@cg-hg.com*

Agile Methodologies have recently become popular across the spectrum of technology activity from Agile Start-ups. But is all this talk of Agile methods with short 'sprints' and a dependency on 'user stories' consistent with the Enterprise Architecture's goal of having a comprehensive map of all the relevant technologies and the relevant business and technology roadmaps? A common perception is "no". As one author put it:

> *"many agile methods are perceived as architecturally weak, disconnected from the realities of delivering large systems in complex enterprise environments"*[1]

Enterprise Architecture is also focused heavily on the processes of change and complexity management.

In this paper I will discuss where there are conflicts between traditional Enterprise Architecture approaches and Agile Methodologies, and where the two align, as well as some best practices for integrating the two.

[1] http://www.agilearchitect.org/agile/index.asp Andrew Johnston.

Principles of Agile Methods

The Agile movement started with The Manifesto for Agile Software Development in 2001:

> ***Definition:*** *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. http://www.agilemanifesto.org/

From this manifesto evolved twelve principles of software development:

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.

Now you will notice the above is about software development. But since the Agile Manifesto was released, the focus on customer input, changing requirements and self-motivated small teams has spread throughout the business community. MIT Sloan School of Management's Center for Information Systems Research has specific efforts focused on "Agility research"[2] which has led to the Business Agile Enterprise (BA-E) concept. B-AE sets 10 (as opposed to the above 12) competencies[3]

1. Business & IT Convergence

2. Competencies that drive Superior Value from IT

3. IT portfolio asset class management (Enterprise Architecture)

4. Governance (direction, control & distributed authority for decision making)

5. Senior management leadership involvement

6. Business control of information (intellectual capital & information visibility)

7. Business process understanding (component business modeling)

8. Business case (business value documentation throughout lifecycle)

9. Key business performance indicators (consistent & continuous benefits valuation)

10. External relationships (external assets, skills & competencies to act on opportunities)

One of these models is primarily about technology and the other more about business. Clearly the models need to be bridged. Which of course is the primary role of Enterprise Architecture: Integrating Technology and Business architectures and roadmaps to create a more holistic interaction with technology.

[2]Business Agility & IT Portfolios," MIT Sloan School of Management, Center for Information Systems Research, Summer Session 3, June 2006,

[3]http://en.wikipedia.org/wiki/Business-agile_enterprise#Business-Agile_Proficiencies Wikipedia, Business-Agile Proficiencies

Conflicts Between Agile and Enterprise Architecture

In the introduction I suggested that Enterprise Architecture is somewhat contrary to the Agile approach. From the outset the Agile Manifesto and Enterprise Architecture have a conflict. Where Enterprise Architecture places an emphasis on the architecture process and the clear documentation of Technologies and Business goals, Agile explicitly de-emphasizes those in preference to Individuals, Interactions and Working Software. Similarly Agile's implementation principle of welcoming changing requirements and Face-to-Face vs. written documentation of information are directly at odds with the Enterprise Architecture approach. Agile also cites simplicity as a key implementation principle, but many systems are inherently complex and one of the roles of Enterprise Architecture is the management of this complexity. On the other hand, the principles of cooperation between business and development, and simplification of architectures with a focus on excellence is completely consistent.

Enterprise Architecture is also typically focused on the organization itself – with senior management or business leadership aka "stakeholders" as the "customer" whereas Agile explicitly places the priority on the satisfaction of the end customer.

Similarities Between Agile and Enterprise

Probably the two key similarities between the Agile approach to technology and the Enterprise Architecture approach are the focus on goals and outcomes, and the effort to simplify where possible. Both Agile and Enterprise Architecture explicitly seek to reduce redundancy and unnecessary technology. Both approaches also measure their success by evaluating the outcomes and how close they come to the customers' goals even if their definitions of customers are different.

As mentioned above, the goals of cooperation between business and development is a key motivator of Enterprise Architecture, and the need to continually keep the Enterprise Architecture roadmaps, and technology architectures tuned and up-to-date is also a core value of Enterprise Architecture.

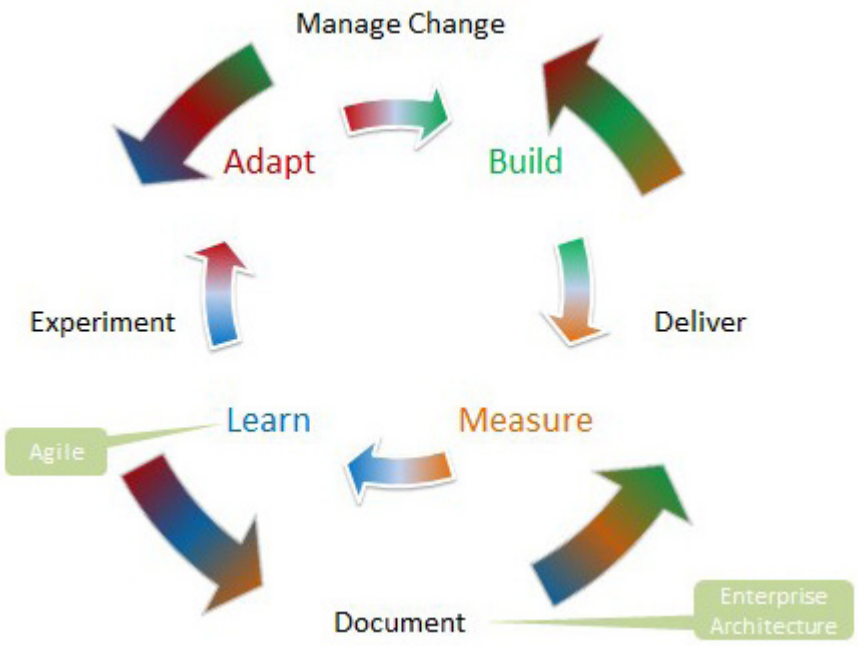
Sustainable development can also be seen in the goals of Enterprise Architecture. The creation of roadmaps both for business goals as well as technology goals and capabilities is a form of insuring that development anticipates the needs and changes on the horizon. The primary goal is to enable a proactive response in place of a reactive one, but proactive addressing of changes and requirements is part of how sustainability is implemented. A continuous environment of reactivity is sustainable neither at the staff level, nor at the financial level, given how much more reactive development costs in comparison with proactive development.

Integrating Agile Methodologies into Enterprise Architecture

So how then to reconcile these two approaches, since in a large organization the lack of an Enterprise Architecture increases cost and puts the company at competitive risk, yet the failure to adopt Agile principles also risks the same?

Various organizations have offered suggestions. Already mentioned has been the MIT Sloan School of Management's "Business Agile Enterprise", the Agile Alliance[4] offers the PUMA methodology, Gartner suggests blending Lean and Agile methodologies[5] etc.

If we take the "first principles" approach, looking at both Enterprise Architecture and Agile approaches we can visualize them as:



Both processes Build, Measure, Learn, and Adapt as part of their ongoing cycle. Both processes Document, Deliver, Manage Change, and Experiment as part of their ongoing cycle. But they start at different points in this cycle and place emphasis on different aspects of the process.

[4]http://www.entreprise-agile.com/en/Puma-en.htm Agile Alliance, Jean-Pierre Vickoff

[5]http://blogs.gartner.com/nathan-wilson/agile-and-lean-what-do-they-really-mean/ Nathan Wilson October 18, 2013

Agile starts with:

- Learning about the customer requirements,
- Experiments with implementing them,
- Adaptations based on customer feedback
- Managing the change of that adaption
- Building the iteration
- Delivering it to the customer
- Measuring the customer's response
- Documenting the response and the defects

And Repeats.

Enterprise Architecture starts by:

- Documenting the business and technology visions
- Measuring the difference between the vision and existing architecture
- Delivering an opportunity map
- Building a migration plan
- Implementing governance and change management
- Experimenting and learning from the implementation and repeats[6]

But the two processes tend to flow in somewhat opposite directions and have different starting points. We can integrate them by looking at what the processes are seeking to accomplish:

- They seek solutions that meet the goals of all stake holders, customers, business leadership, and technology managers alike -
    - In a way that maximizes stakeholder value (customer and business alike)
    - That work
    - That manage change and complexity in a way that minimizes constraints on future efforts

[6]http://pubs.opengroup.org/architecture/togaf9-doc/arch/ The Open Architecture Group Framework Architecture Development Cycle

Avoiding Common Failures

In my previous series of white papers on *Enterprise Architecture : Inside Out or Outside In* I offered the guidance that the kickoff of any Enterprise Architecture effort needed to deliver value to stakeholders quickly as a way of justifying the effort. That was an application of "Agile" principles to an Enterprise Architecture approach. The goal was to offer an EA solution to stakeholders that worked, that quickly delivered value, and that enabled future EA efforts.

The goal was to avoid the common Enterprise Architecture failure pitfalls: namely high cost, slow delivery of results and the creation of unnecessary processes. Agile's common failures are that

- The result is often inadequately documented
- The result is often inadequately tested
- Future customer requirements ask for major redevelopment

The reason is that Agile intentionally brings the "event horizon" in closer by focusing only on what is in the current "sprint". This can lead to architectural solutions that are not easily extensible. And at the same time, if the solution is built too generically, future performance might be compromised. Similarly test cases that address only the current "sprint".

Definition: *Agile Sprint*

> A sprint (or iteration) is the basic unit of development in Scrum. The sprint is a "time-boxed" effort; that is, it is restricted to a specific duration. The duration is fixed in advance for each sprint and is normally between one week and one month, although two weeks is typical.
>
> Schwaber, Ken. Agile Project Management with Scrum. Microsoft Press ISBN 978-0-7356-1993-7

Definition: *Product and Sprint Backlog*

> The product backlog is an ordered list of requirements that is maintained for a product. It consists of features, bug fixes, non-functional requirements, etc.—whatever needs to be done in order to successfully deliver a viable product.
>
> The sprint backlog is the list of work the Development Team must address during the next sprint. The list is derived by selecting product backlog items from the top of the product backlog until the Development Team feels it has enough work to fill the sprint.
>
> Higgins, Tony. Authoring Requirements in an Agile World. BA Times March 31, 2009

Requirements (User Stories) often do not sufficiently test their interaction with future features. Finally, while having immediate customer feedback can enable faster adaptation and deliver a product closer to the customer's vision, it can also result in the customer losing sight of the overall vision and asking for otherwise conflicting features.

To avoid the pitfalls of both the traditional Enterprise Architecture as well as those of Agile methodologies a synthesis is required, particularly on large scale and complex projects. Enterprise Architecture by its nature intentionally slows and manages some forms of change. Agile by its nature intentionally increases the rate of change and speed of delivery.

By using Enterprise Architecture as a structural framework on which to implement Agile solutions, it is possible to integrate the two. Enterprise Architecture thus becomes one of the customers in an Agile development cycle – adding in requirements to both the Product and Sprint Backlogs. Requirements such as architectural principles, technologies to be used, types of testing requirements as well as the level of documentation to be delivered.

Similarly Agile principles of iterative development with short Time-to-Value for customers can be applied to an Enterprise Architecture implementation effort as well. Rather than identifying all of the technologies in an organization, an Agile EA effort would opt to focus on a particular high value segment of the enterprise and identify a particular value point (such as reduced technology redundancy, or better alignment with business roadmaps) to deliver on. And then repeat that cycle iteratively.

Summary

Enterprise Architecture and Agile Methodologies do have some conflicting goals. In part these goals come about because of both differing focal points (process repeatability and documentation for EA vs. adaptive change and short delivery cycles for Agile) different starting points in the development process (learning for Agile, documentation for Enterprise Architecture), as well as different process flows.

However their fundamental goals are similar:

They seek to create solutions -

- that meet the goals of all stakeholders, customers, business leadership, and technology managers alike
- In a way that maximizes stakeholder value (customer and business alike)
- That work
- That manage change and complexity in a way that minimizes constraints on future efforts.

If the two approaches are balanced against each other and placed in the right context with respect to each other – they can and do enhance the achievement of those goals:

Enterprise Architecture as a framework and customer of Agile brings a rigor, consistency and overarching vision that reduces the likelihood of an Agile process failing to test and document the changes it makes along the way.

Agile Methodology applied to implementation of Enterprise Architecture increases the perception of value delivered by EA, and speeds up the achievement of tangible results.

Orbus Software
3rd Floor
111 Buckingham Palace Road
London
SW1W 0SR
United Kingdom

+44 (0) 870 991 1851
enquiries@orbussoftware.com
www.orbussoftware.com