American Council for Technology | Industry Advisory Council

Planning for Success: Agile Software Development in Federal Agencies

Emerging Technology Shared Interest Group (SIG)

Date Released: August 2013

Many federal agencies are contemplating or just beginning to adopt agile software development. Disruptive for any organization, agile adoption is particularly so for federal agencies where waterfall software development is most familiar and deeply entrenched in practices from procurement to deployment.

So how do we get started with agile? The good news: agile has been around long enough for us to understand best practices, pre-requisites for success, and strategies for overcoming obstacles. This paper is intended to serve as a "quick start" guide to adopting agile in the federal agency setting, and provides the foundation to build a successful agile practice.

3040 Williams Drive, Suite 610, Fairfax, VA 22031
www.actgov.org ● (p) (703) 208.4800 (f) ● (703) 208.4805

Advancing Government through Collaboration, Education and Action

American Council for Technology-Industry Advisory Council (ACT-IAC)

The American Council for Technology (ACT) is a non-profit educational organization established in 1979 to improve government through the efficient and innovative application of information technology. In 1989 ACT established the Industry Advisory Council (IAC) to bring industry and government executives together to collaborate on IT issues of interest to the Government.

ACT-IAC is a unique, public-private partnership dedicated to helping Government use technology to serve the public. The purposes of the organization are to communicate, educate, inform, and collaborate. ACT-IAC also works to promote the profession of public IT management. ACT-IAC offers a wide range of programs to accomplish these purposes.

ACT-IAC welcomes the participation of all public and private organizations committed to improving the delivery of public services through the effective and efficient use of IT. For membership and other information, visit the ACT-IAC website at www.actgov.org.

Emerging Technology SIG

The Emerging Technology SIG serves the federal CXO and government executive community responsible for identifying, assessing, and deploying emerging technology and maturing it to become a major component of the IT & business strategy. The ET SIG actively monitors the emerging technology landscape, evaluates game-changing technologies, provides insight on high-potential technologies to accelerate awareness and adoption, and shares lessons learned across the government IT community.

Disclaimer

Copyright

Further Information

For further information, contact the American Council for Technology-Industry Advisory Council at (703) 208-4800 or www.actgov.org.

Table of Contents

Executive Summary

Agile software development is a framework for delivering working software frequently and efficiently. It can fulfill the promise of software development speed, flexibility and delivery success in a landscape notable for major project failures. Widely accepted as best practice in the commercial sector, agile software development is ripe for adoption in the Federal Government where computer systems are large, agency problems are complex and ever-changing, and IT budgets are shrinking.

This paper provides:

- An overview of the nature and components of agile software development,
- Requirements and best practices for adopting agile, and
- Strategies to overcome hurdles inherent to Federal Government agencies.

Nature and Components of Agile Software Development

Agile has evolved as an efficient means of reliably developing and delivering software. The paper reviews key tenets of agile development that have emerged from *The Agile Manifesto* and which drive the productivity and quality that is the hallmark of good agile development.

Requirements and Best Practices for Implementing Agile

Next, the paper explains the essential foundation elements for agile adoption (an IT champion, a willing customer, and a commitment to transparency) and the building blocks to successful agile implementation. These building blocks include team skill, selection of projects for early success, and establishing discipline and sound agile practices around scoping, development, testing, and delivery of software.

Adopting Agile in the Federal Agency Environment

Finally, the paper addresses the unique challenges facing agile adoption in the federal agency setting and pathways to overcoming obstacles and attaining success. The paper offers both examples of agile implementation success in the federal setting, as well as a discussion of complementary software development techniques to address specific development scenarios. In addition, the paper suggests other resources agencies can leverage – books, blogs and a growing community of practitioners – on their path to successful agile adoption.

Acknowledgement

We would like to thank the Federal agency IT executives and expert practitioners who were interviewed as part of the writing of this paper. Their views were taken into account and some are included as unattributed quotes to amplify key points. We especially thank a handful of IT executives who reviewed multiple early drafts and provided excellent feedback and encouragement.

1 What is Agile Software Development?

1.1 Background

Since the 1950s, software professionals have advocated the benefits of "iterative and incremental" development.[1] By the mid-1990's, numerous iterative development methodologies had emerged, from Rapid Application Development (RAD) and Rational Unified Process (RUP), to Scrum,[2] Digital System Design (DSD) and Crystal Clear. Despite the availability of iterative development methodologies, 1980s and 1990s software projects predominantly used non-iterative – or Waterfall – methodologies with disastrous results. Waterfall methodologies are driven by the belief that by creating better requirements prior to writing software, projects would be less risky, faster and cheaper. In reality, extensive written requirements have not contributed to low-risk, low-cost, high-speed software development projects.

A 1999 review of failure rates in a sample of earlier DoD projects drew grave conclusions: "Of a total $37 billion for the sample set, 75% of the projects failed or were never used, and only 2% were used without extensive modification."

-- Larman & Basili[1]

1.2 Emergence of Agile

In 2001, a handful of visionary software engineers applied the term "agile" to software in creating *The Agile Manifesto* as an umbrella set of principles spanning the various iterative development methodologies.[3] Their goal was to spur professionals to continue advancing methods for delivering software more reliably. *The Agile Manifesto* suggests something counterintuitive: that we embrace the inherent uncertainty in creating software; that we learn by doing rather than specifying. Its principles are anchored in the axiom that by *producing working software regularly, in collaboration with users, solutions can be created more reliably and efficiently*.

1.3 Agile Defined

Agile is not a methodology but, rather, a discipline – the discipline of regularly producing working software. *The Agile Manifesto* defines a set of principles, and the term "agile" is used to describe methods and practices adhering to those principles. An agile development discipline includes principles defined in Table 1 below.

[1] Larman, \Craig & Victor R. Basili. (June 2003), "Iterative and Incremental Development: A Brief History." IEEE Computer (IEEE Computer Society) 36 (6): 47-56.
[2] Schwaber, Ken. (March 10, 2004). *Agile project management with scrum*. Microsoft Press.
[3] Beck, Kent, et al. (2001). The agile manifesto. Retrieved September 5, 2012 from http://agilemanifesto.org/

Table 1: Principles of an Agile Development Discipline

Principles	*Description*
Work iteratively	Working software is demonstrated in time-boxed intervals of several weeks or less.
Test thoroughly	Software is tested rigorously and often.
Discover requirements	Teams discover requirements while implementing, detailing only enough to understand, implement, and test.
Involve the customer	The customer (product owner) is intimately involved in making priority decisions.
Favor software over documents	Working software replaces extensive documentation.
Establish cadence	Teams work to a steady pace, not relying on last minute heroics for delivery.
Build just enough	Each turn of the solution does just enough, avoiding unnecessary complexity and non-essential features.
Embrace learning	Teams strive constantly to improve, learning from the past, and incorporating those lessons into future actions.
Take accountability	Teams take responsibility for their work, skill, methods, and commitments.

Too often, agile is used in a way that implies ad hoc behavior. Nothing could be further from the truth. Agile is not:

- An excuse to avoid planning or creating a business justification;
- A pass to ignore software engineering practices (e.g., agile requirements management, code management, build, testing);
- A license to change direction constantly, thrashing teams;
- An effort devoid of leadership, clear roles or structure; or
- The right to “make it up as we go along.”

Make no mistake: agile is formal, disciplined, and hard to do well. But, for those who choose to embrace agile, the rewards are outstanding software delivered safely and efficiently – and just as importantly – happy customers, teams, and management.

2 Best Practices for Agile Development

2.1 Agile Organizational Baseline

Agile development takes root in organizations where there is a commitment to the discipline. Because it requires cross-team collaboration, agile is the sort of discipline that you cannot do halfway. Ensuring an agile organizational baseline will increase the likelihood that agile adoption succeeds.

The Agile Organizational Baseline	
WHAT	**WHY**
An IT champion: a leader within the IT organization who is deeply experienced as a successful manager of software projects and is a student of software development process.	While agile is not all about processes, a working professional knowledge of various software development lifecycles will help the organization transition to agile.
A willing customer: a customer executive with a pressing business need not currently being met who is willing to commit resources and patience to early projects.	The customer doesn't have to be enlightened about software development, that's the IT champion's job, but they do need to have the will to help agile flourish and to engage in the adoption process.
A commitment to transparency: a willingness to share information continuously in the interest of performance improvement.	You cannot evaluate, correct, and improve what you cannot see, and the speed and effectiveness of agile depends on continuous improvement. Agile requires transparency across all project dimensions: team skill and performance, project progress, and solution suitability.

2.2 Building Blocks for Agile Success

With the agile organizational baseline in place, agile success next depends on the development effort itself. The development team members are the agile standard-bearers and must have the skill, attitude and drive to model the agile discipline for the organization. Project selection is also important. Early project "wins" can accelerate the pace of agile adoption and reinforce its feasibility and value. Finally, agile practices reinforcing delivery discipline and transparency can build the foundation of trust between the IT organization and the business that will pay continuous dividends.

2.3 Skilled Agile Development Team

Effective software development requires skill that can be learned, but gaining expertise requires years of both doing and studying. Agile development builds on decades of software engineering knowledge and involves multiple related skills, e.g., analysis, programming, testing, and engineering.

Because inexperienced teams attempting agile development will make mistakes, and mistakes can be fatal to a project in its early stages, hiring skilled team members with thorough knowledge of, and a track record of success using, agile practices is essential. And while hiring a ready-made skilled team may not be possible, hiring skilled individuals in key positions will go a long way to ensuring success.

"Having the right skills for the problem is absolutely essential."

-- A federal IT leader

Importance of Coaching

Expert agile coaches can have a big impact also. Newcomers to agile will need training to understand key concepts, but training alone may not be enough. Agile coaches provide leverage for the IT champion and greatly improve the chance for successful delivery. Working closely alongside teams, coaches look for problem areas, identify skill gaps, and guide teams through unfamiliar territory. Over multiple sprints, releases, and projects, coaching raises the efficiency and effectiveness of the organization.

Culture of Continuous Improvement

Agile methods insist that teams reflect upon their performance, pinpoint problem areas for improvement, and implement necessary changes. For example, Scrum[13] ends each sprint with a *retrospective*. Scrum is a framework for development team collaboration most often used within agile methodologies. The retrospective is a formal meeting when customers and technical teams jointly identify areas for improvement. Whether skill, process, technology, or decision-making, no project element should be off-limits to examination.[4]

2.4 Suitable Initial Agile Projects

Choosing projects for success is particularly important in the early stages of agile adoption. Because the organization is making a dramatic change by adopting agile, it helps to minimize other risks that often contribute to failed projects. When evaluating initial projects for agile development, look for a willing product owner with a business need to show results quickly, where there are opportunities to create quick releases and early wins. Consider the following:

- Is there an urgent need for a solution to a business problem for which the scope can be contained, or a first iteration can be scoped and delivered?
- Is the problem-space and solution-space well enough understood to get started, but too large or complicated to tackle all at once?
- Can a team of ten or less people deliver the solution in less than six months?

[4] Poppendieck, Mary & Tom Poppendieck. (May 18, 2003). Lean software development: An agile toolkit. Chapter 2: Amplify Learning. Addison-Wesley Professional.

- Is the team ready to immerse themselves, and engage the customer, in an agile approach?

Projects for which the answer to these questions is “yes” pose the lowest risk and may prove good starting points for agencies seeking to adopt agile.

2.5 Sound Agile Practices

Engage a Committed Product Owner

Expert agile practitioners emphasize the importance of having a *product owner* working closely with the team. *Scrum* defines the product owner as the “voice of the customer”, ensuring that the team delivers value by defining and prioritizing features (or *user stories*). The product owner does not need to be a customer, but it helps. The agile team must have consistent and timely participation by the product owner - in many cases, daily involvement.

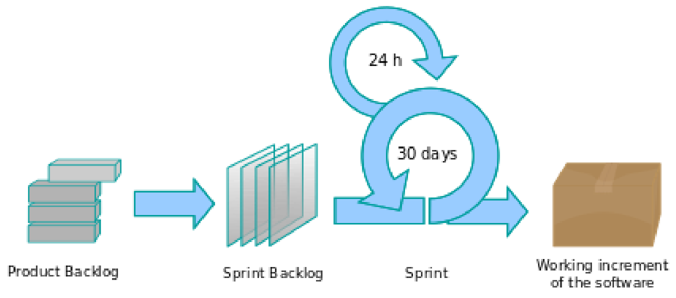
> “The biggest myth is no documentation; [but] they have to get comfortable with the fact that they're not going to get 500 page requirements document; you have to trust that the code is good enough.”
>
> -- A federal IT leader

Embrace Agile-Style Requirements

Agile methods rely on the assertion that requirements evolve through the act of creating working software, and that premature requirements efforts waste precious resources and frequently lead the team to deliver the wrong features. Because agile methods focus on delivering working software in many, small, time-boxed chunks of effort, only the most important requirements are needed next. However, agile requirements management is not lax. It is well-documented and disciplined, regardless of agile methodology.[5, 6]

Deliver Working Software Frequently

Frequent software delivery is the cornerstone of agile practice. Agile methods deliver working software frequently in time-boxed efforts (usually, four weeks or less), called *sprints*. A solution should be ready for production in less than six to nine months – sooner is



[5] Cohn, Mike. (March 11, 2004). *User stories applied: For agile software development*. Addison-Wesley Professional.
[6] Martin, Robert C. (October 25, 2002). *Agile software development: Principles, patterns and practices*. Prentice Hall.

better. Many solutions will require multiple releases to complete valued features.

The key to frequent software releases is the *use of automation* to accelerate testing, manage rapidly changing code, and (sometimes) track work requests.

Automating test cases is the only feasible way to test software as extensively and frequently as is required by agile and ensure that every sprint and release delivery is high quality. Agile embraces the idea that requirements can change continuously, which means that an ever-growing volume of unit, system, and regression tests must be applied to the code base continuously. Manual testing would not be able to meet the speed requirement of agile. So automation must be employed to handle the task. This approach ensures quality even when the software is changing rapidly.

Continuous integration (CI) avoids the costly and painful problems created when teams merge their code changes infrequently, or as was once common, near the end of the project. Teams practicing CI merge their changes as soon as they are complete and rely on automation to catch mistakes.[7] CI software automates checking all code out of the source repository, building and configuring the software package, running all unit and integration tests, pushing the software into other environments, and generating reports and metrics. Mature agile projects run this cycle many times a day – every time a code change is checked into the code base.

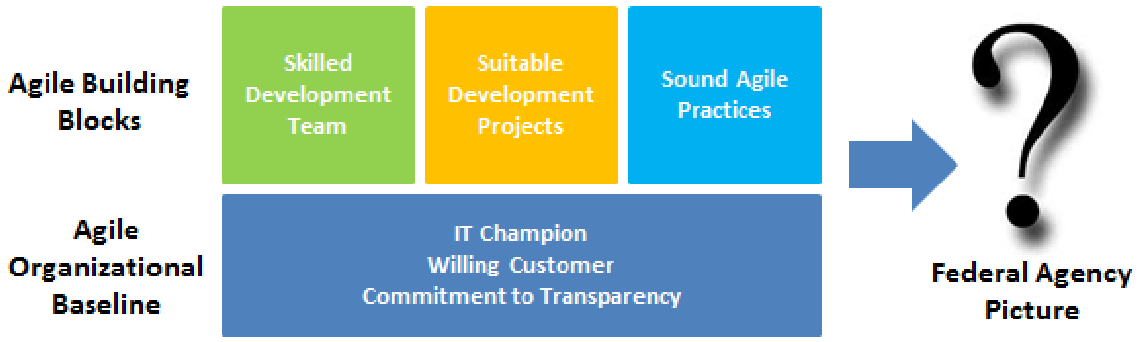
Agile teams also rely heavily on tools for managing requirements and work requests (features and bugs). For large-scale projects, automated tools may be the only means of tracking user stories and work requests effectively. By automating project tracking and integrating it with CI platforms, teams have a complete view of a project's health and progress. For many projects, however, use of information "radiators" – such as sticky notes on a project board – is sufficient. In all cases, it is imperative that agile projects have a well-understood means of displaying progress.

Automation and continuous integration are predicated on infrastructure flexibility and access to multiple environments. Agile requires more environments and environment flexibility because it emphasizes heavy and continuous testing, and it parallelizes activities, thus confounding environment sharing. This imperative puts stress on traditional operations support organizations. Fortunately, mitigating tension at the apps/ops boundary is easier today than ever before. Include operations staff early and continuously in the agile projects, and leverage the cloud for development environments.

[7] Duvall, Paul M. (July 9, 2007). Continuous integration: Improving software quality and reducing risk. Addison-Wesley Professional.

3 Agile in the Federal Agency Environment

With an understanding of the agile organizational baseline and a review of the best practices for implementing the agile discipline, how can we apply the agile model in the Federal Agency setting? Without question, flexibility in procedures and evolution of culture in the federal setting will be necessary to support agile adoption. But many IT leaders in the Federal Government are committed to finding a way to take advantage of the quality, speed, and efficiency benefits of agile development.[8]



Key to implementing agile successfully in the government environment is understanding the structural hurdles and established patterns of behavior that will need to be overcome in order to set the stage for agile success. The guidance that follows should help agencies in two ways: to accelerate the agile adoption process, and to discern when an agile approach needs to be complemented with other management techniques.

3.1 Be Prepared to Counter the Culture of Scrutiny

Many agencies are reluctant to raise project problems and risks because of the external scrutiny that bad news can bring. Agile methods demand honest introspection so that teams can improve, yet this can put agencies at odds with the desire to minimize their exposure to scrutiny. Agencies can avoid this dilemma in two ways: execute projects of a manageable size (or large projects in manageable increments) and manage risks, not issues.

Think small to start: projects with deliverables of a modest size attract less interest and are inherently less risky. Even if something goes wrong, most customers are willing to experiment on a small scale, and can absorb failure if it happens.

Manageable scope and small teams all increase the odds of success. Agile methods insist on demonstrating working software every four-six weeks or less to keep scope and complexity manageable. If a potential project seems too large, consider the

[8] (July 27, 2012) Software development: Effective practices and federal challenges in applying agile methods. GAO-12-681. U.S. GAO. Retrieved September 5, 2012 from http://www.gao.gov/products/GAO-12-681

possibility of scoping an initial, high-value solution component. Selecting the scope for an initial release of a larger project can be difficult when functional requirements are complex and numerous. Seasoned agile practitioners can help agencies select the minimum essential set of functions to include in an initial release and then demonstrate how the agile approach can deliver business value quickly.

Manage risks: risks are early hints that something bad could happen without intervention; issues are thorny problems impacting progress, often visible ahead of time as risks. Preventing risks from turning into issues – managing risk – is routine, inexpensive, and painless, and should be part of any project activity. Teams may resist the shift from managing issues to managing risks for fear of reprisal. Make risk management activity an early focus so that team members understand what the practice entails and can train themselves to look over the horizon and plan proactively to identify and mitigate risks. Remove the fear of reprisal and the shift to risk management is swift and natural. Teams relish the success of identifying and managing risks rather than full blown problems.

3.2 Rethink Procurement Practices and Progress Measurement[9]

A bias to waterfall development, demanding *big requirements up front (BRUF),* permeates federal procurement practices, measurement procedures, and belief systems. A recent federal procurement emphasis on doing more firm fixed price (FFP) contracting is exacerbating the problem, as FFP contracting is implemented with a BRUF requirements mindset.

Traditional progress measurement also reinforces waterfall methods. For example, earned value management (EVM) requiring percentage completion estimates forces teams to track questionable indicators of progress and document-centric gate reviews force alignment to waterfall phases such as requirements, design, and test. As noted at the beginning of this paper, requirements *cannot* be known in advance for most software systems. Agencies pursuing agile

Agile Return on Investment

In 2011, the Office of the Chief Information Officer at a large Federal agency decided to abandon its traditional under-performing development practices and adopt agile. It wasn't easy. They cut ties with existing vendors and embarked on adopting agile practices while simultaneously altering their procurement process and working to rebuild trust relationships with their customers.

The change to agile paid huge dividends. In less than 18 months, the agency's development group delivered four major software releases successfully, with additional deliverables to follow. When their program is complete, they'll have invested two years and $12 million to complete a program scope that was originally projected to take 5 years and cost over $60 million, a significant return on their investment in adopting agile.

[9] Kundra, Vivek. (Dec 9, 2010), 25 point implementation plan to reform federal information technology management. The White House. Retrieved September 5, 2012 from

must push back consistently and forcefully against these biases.

In the absence of full-scale procurement and progress measurement practice reform, what strategies can agencies use to begin to adopt agile? Consider developing a performance work statement structured to indicate essential, high-level business requirements, total budget, high-level deliver-by dates, and ***how*** the work will be performed (i.e., in small releases of tested software, iteratively, using agile methods).[10] Rapid, frequent delivery of software, flexibility to incorporate evolving requirements, transparency, and thorough evaluation of team performance and deliverables are inherent in the agile framework. These attributes should help allay fears that, without BRUF, agencies run the risk of not getting what they want or need.

3.3 Be Strategic in Accessing Required Skill

Software creation requires substantial skill, yet the Federal Government cannot hire enough skilled workers to lead the volume of software projects it funds. Lower pay scales, cyclical need within an agency for major software efforts, and late adoption of innovation put the Federal Government at a disadvantage in competing for and keeping skilled developers. So while skill might be available outside the Federal Government, the Federal Government's drive to in-source has not been able to meet the federal IT demand for skilled workers.

Agencies must learn to strike the right balance between skilled federal leadership and skilled contractor support on project teams. Agencies should not undertake risky software projects without demonstrated leadership, and should take advantage of available private sector skill for execution when necessary. It's also important to ensure that agile talent is developed internally for the longer term. In order to reap the benefits of agile, agencies should look to cultivate the role and skill of the product owner. Product owners should be collaborative, result-oriented individuals with good communication skills and deep domain expertise. It is the product owner who will partner with the agile development team to continuously manage and prioritize business requirements for development.

https://cio.gov/wp-content/uploads/downloads/2012/09/25-Point-Implementation-Plan-to-Reform-Federal-IT.pdf

[10] Jordan, Joseph G. & Steven VanRoekel. (June 14, 2012). Contracting guidance to support modular development. The White House. Retrieved September 5, 2012 from http://www.whitehouse.gov/sites/default/files/omb/procurement/guidance/modular-approaches-for-information-technology.pdf

4 When More than Agile is Required

While the agile framework can provide an excellent foundation for all software development, there are times when complementary methods may be more effective for some components of the development lifecycle. For certain situations – such as operations, production support, and maintenance activities – agile planning and management techniques may slow progress. In these instances, Lean methods, such as software Kanban, might be employed instead of Scrum to manage work.[11] Many agile techniques, however, do apply to maintenance activities – including testing, build, and deployment automation.

Other situations, such as legacy modernization (LM), demand modified requirements discovery and testing processes.[12] The first step in any sizable LM effort is to replace the existing system with a functionally equivalent system in the new architecture and technology. This situation differs in that the existing system is the predominant source of requirements, not users or customers. Agile practices still apply to development and testing, particularly automation of regression testing to ensure new code will function effectively in the old system. Also, testing must consider comparison of the replacement system against the legacy system and validate migration of legacy data.

Lastly, as mentioned earlier, applying agile to very large software programs will require concerted effort to break the large effort into smaller projects and mitigate dependencies between them. This is an area of evolving thinking.[13, 14, 15] Agencies seeking to apply agile to a major initiative would be well-served to engage the advice of a seasoned agile practitioner or coach as part of the procurement process. This expertise can help shape the scope and size an initial release so as to attract providers offering an agile approach. As an alternative, agencies might consider initiating multiple, independent agile projects as a means of embedding the agile approach in the organization.

[11] Anderson, David J. (April 7, 2010). Kanban: Successful evolutionary change for your technology business. Blue Hole Press.
[12] Estes, Don (Ed.). (December 2010). *The journey to successful application modernization and rationalization*. Cutter IT Journal.
[13] *Larman, Craig & Bas Vodde.* (February 5, 2010)*. Practices for scaling lean & agile development*. Addison-Wesley Professional.
[14] Leffingwell, Dean. (March 8, 2007). *Scaling software agility*. Addison-Wesley Professional.
[15] Ambler, Scott & Mark Line. (June 2, 20120). *Disciplined agile delivery*. IBM Press.

5 The Journey to Becoming Agile

This paper highlights conditions necessary for adopting best practices for becoming agile, and some recommendations to promote successful agile adoption in the Federal Government environment. The requirements for successful agile adoption and demands of the agile framework for software development may seem intimidating to early adopters. The guiding principle of agile is delivering working software frequently to stakeholders for evaluation or release. With this foundation principle in place, the surrounding framework simply serves to support delivery, enabling speed without jeopardizing code quality, allowing for requirements flexibility without introducing delay, and promoting continuous evaluation of team and code performance through retrospectives, ensuring that teams refine their skill and improve delivery with each successive release.

The track record of success for well-implemented agile development is impressive, including examples in the Federal Government. NASA's Jet Propulsion Lab uses agile approaches and tools to coordinate the activities of developers across three teams working on extremely complex initiatives.[16] The U.S. Department of Veterans Affairs[17] and the Central Intelligence Agency[18] have looked to agile to drive productivity and transparency in their software development. Under extreme time pressure and using an agile team, tools and methods, the NTIA and FCC[19] successfully delivered a critical policy initiative - the National Broadband Map - that includes many technical innovations.

There is a very large body of excellent knowledge to help those with the desire to learn. References to a few of these resources are listed in the appendix, and numerous resources are available online. In addition, there is an active community of agile professionals locally, within the public sector, and on the Internet. This community is very supportive of new adopters. Join in and, as one federal "agilista" said, "keep it fun!"

[16] Goff-Dupont, Sarah. (May 14, 2012). *NASA's Jet Propulsion Laboratory launches Atlassian into space*. Atlassian Blog. Retrieved April 25, 2013 from http://blogs.atlassian.com/2012/05/nasa-atlassian-development-tools/

[17] Thibodeau, Patrick. (January 22, 2013). *Feds turn to agile development as budget cuts loom*. Computerworld. Retrieved April 25, 2013 from http://www.computerworld.com/s/article/9235966/Feds_turn_to_agile_development_as_budget_cuts_loom

[18] Wailgum, Thomas. (August 4, 2008). Inside the CIA's Extreme technology makeover, part 1. CIO.Com. Retrieved April 25, 2013 from http://www.cio.com/article/441116/Inside_the_CIA_s_Extreme_Technology_Makeover_Part_1?page=2&taxonomyId=3154

[19] Bastian, Zachary & Michael Byrne. (2012). *The National Broadband Map: A case study on open innovation for national policy.* Washington, D.C. Woodrow Wilson International Center for Scholars. Retrieved April 25, 2013 from http://wilsoncenter.org/publication/the-national-broadband-map-case-study-open-innovation-for-national-policy

6 Appendix

The Agile Manifesto states four values and twelve principles. The complete manifesto is at http://agilemanifesto.org. The four values are:

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.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler

James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick

Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

7 Authors & Affiliations

Primary Authors
Lawrence Fitzpatrick, Computech, Inc.
Christine Swistro, Computech, Inc.

Contributors
Sandra Van Valkenburg, CGI Federal
Robert Clarke, ASPE, Inc.

We would like to thank the federal agency IT executives and expert practitioners who were interviewed as part of the writing of this paper. Their views were taken into account and some are included as unattributed quotes to amplify key points. We especially thank a handful of IT executives who reviewed multiple early drafts and provided excellent feedback and encouragement.