Deloitte.
Insights

Agile in Government

A playbook from the Deloitte Center for Government Insights

ABOUT THE DELOITTE CENTER FOR GOVERNMENT INSIGHTS

The Deloitte Center for Government Insights shares inspiring stories of government innovation, looking at what's behind the adoption of new technologies and management practices. We produce cuttingedge research that guides public officials without burying them in jargon and minutiae, crystalizing essential insights in an easy-to-absorb format. Through research, forums, and immersive workshops, our goal is to provide public officials, policy professionals, and members of the media with fresh insights that advance an understanding of what is possible in government transformation.

Deloitte Consulting LLP's Technology Strategy & Architecture and Systems Integration teams help accelerate business performance by managing the complexity associated with large-scale technology change. The Systems Integration practice helps clients zero in on their toughest challenges across the technology lifecycle. The practice offers a complete range of industry-leading services, including application architecture, custom systems development, and solution and platform integration, as well as program management, functional, and testing services. Additionally, Deloitte's Technology Strategy & Architecture specialists help clients develop strategies and implement systems that build business value and drive performance.

CONTENTS

Introduction | **2**
An Agile playbook for government

When Agile meets government | **4**
Closing the culture gap

Scaling Agile for government | **8**
Using Agile on large, complex projects in government

Going Agile: The new mind-set for procurement officials | **14**
How does Agile change the role of the acquisition officer?

The art of points-based procurement for Agile projects | **18**

Successful Agile in government | **24**
Supporting the product owner

Bringing Agile benefits to a waterfall project | **32**
Visual design simulations

Managing an Agile program? Consider an AMO | **36**

Agile by the numbers | **42**
A data analysis of Agile development in the US federal government

Introduction

An Agile playbook for government

By Deborah Sills and Warren Miller

In February 2001, 17 serious software geeks gathered at a ski resort in the Wasatch Mountains of Utah. Did they ski? A little. But mostly they talked about how to build software that best meets customers' needs rather than merely conforming to written requirements. Over a three-day period, they wrote a short but influential document called "The Agile Manifesto," and in doing so, seem to have spawned a revolution in the way that many software systems get built. Agile stressed collaboration, adaptation, and iterative reviews—useful approaches in an era of rapid change.

The best ideas typically solve real problems. Agile software development came into being in the early 2000s to address perceived shortcomings with the traditional waterfall approach.

Today, Agile and its variants are perhaps the most common approach to software development. So why hasn't the public sector embraced the use of Agile to the same extent the private sector has?

The answer may lie in the nature of government procurement and other unique features of the public sector. This collection of articles focuses on tackling the thorniest challenges of Agile within the context of the public sector.

In working with our clients, we heard many who were interested in taking advantage of the benefits of Agile, but weren't sure how to proceed. In addition, while there are lots of resources out there that explain how Agile works, there doesn't appear to be much that looks specifically at how to apply Agile in the government context.

This is no small gap. Government buys a lot of IT, but the Agile process doesn't naturally align with public purchasing.

For valid reasons, public procurement seeks certainty. Buyers want to define exactly what they are buying. They want to know what it will cost and when it will be delivered. Understandably, government contract managers want this written into the contract, providing clarity on both sides on what is expected—and legal recourse if it isn't delivered. The essence of Agile, however, is adaptation and flexibility. You generally don't start with highly detailed specifications. Instead, solutions evolve through a collaborative, iterative process.

This Agile playbook covers everything from deciding when Agile does (and doesn't) make sense, to thinking about the contracting process, to how to price Agile, to how best to manage an Agile project. It draws on Deloitte's rich experience of working with public officials using both waterfall and Agile techniques—and everything in between. Our hope in offering this collection is that it can help public officials make informed choices.

A final word of caution. Agile is not a magic bullet. While everyone wants every software procurement to succeed, there is always a risk of failure. No technique is perfect. The better you understand the pitfalls of the journey—whichever journey you choose—the more likely you are to arrive at the destination.

The best IT projects, whether Agile or waterfall, typically involve a close relationship between clients and contractors. After all, vendors and public officials both want projects to be successful. The end goal should always be the same: software that works for the end user, built in a timely fashion at a reasonable cost.

We hope you find the articles in this playbook helpful in thinking about how Agile might work for your public sector organization.

Deborah Sills is a principal with Deloitte Consulting LLP's Public Sector management team.

Warren Miller is a managing director with Deloitte Consulting LLP's Federal technology practice

Agile in Government

When Agile meets government

Closing the culture gap

By: William D. Eggers and John O'Leary

INCREASINGLY, governments are looking to procure software built using the Agile methodology. This could not only demand changes to the procurement process, but also set up a potential culture clash between the rules-driven public sector and the more relaxed technology provider.

For an Agile procurement to be successful, the public sector business users and the Agile developers should work harmoniously. This means bridging a culture gap. From attire to work hours, from documentation to compensation, public sector workers and Agile developers typically have a different way of doing things. Learning to work together despite those differences requires some hard work on both sides.

The language gap

Agile literally has a language all its own. "How many story points is this epic?" "What is the team's velocity?" Instead of project managers, there are product owners and scrum masters. If you're a procurement veteran or a government business manager, your reaction to this is likely a big eye roll.

That was certainly Alistair Montgomery's reaction. Montgomery is an IT manager for Transport for London (TfL). A long-time public official, years of developing IT projects with "waterfall" development left Montgomery far more comfortable in the world of fixed project plans and timelines.[1] When TfL first proposed using Agile development in the summer of 2015, he was deeply skeptical.

"I was set in my ways," he says. "I thought [Agile development] was a gimmick filled with buzz terms and would never work. There were no Gantt charts. No plan. No set delivery. And when I heard about 'scrum masters,' I said, 'Come on, this is ludicrous.'"

Fortunately, Montgomery says, he was outvoted. "Within two weeks, I was a complete convert," he recalls, laughing. What changed his mind? "It was fun," he says. "There was no hierarchy—it was all peer to peer—and I could see progress quickly. That's why people have been won around to this."

Montgomery quickly learned that the new language actually reflects a new way of doing things, and Agile has been successfully adopted at the agency. But to partner successfully, public officials have to be ready to learn the language of Agile.

The rules gap

Of course, government procurement has a language of its own as well—the Federal Acquisition Regulation in the US federal government, Defense Federal Acquisition Regulation Supplement (DFARS) and Procedures, Guidance, and Information (PGI) in the US military, the Federal Acquisition Streamlining Act (FASA) for "small" acquisitions—and databases such as the Federal Procurement Data System, Federal Business Opportunities (FBO), Integrated Award Environment (IAE), and System for Award Management (SAM.gov). This "alphabet soup of acronyms" is the language of rules, regulation, and documentation—an alien world for many adherents to Agile who "just get it done." Part of a procurement officer's job is to use his or her expertise to meet rules in such a way that both the agency and the vendor can focus on the end result, rather than just the rules.

The trust gap

Traditional procurement contracts are designed to protect the government from scams—the underlying assumption being that every vendor might be a potential rip-off artist. Agile requires trust—and lots of it. Procurement contracts should facilitate Agile's process. This means that lawyers and contract officers should craft contracts that prioritize the success of the collaboration, rather than crafting more punitive documents that create an impregnable fortress of words that ostensibly protects the client. In contrast, in the more collaborative Agile approach the close client interactions and regular demonstration of progress help build customer trust. In addition, the shorter time frames can shift power to the procurers.

For example, agencies that issue small, incremental contracts have the ability to switch vendors at any iteration. While not without some disruption for the agency, this may be preferable to being "married" to an unresponsive vendor for a long-term mega-project. Traditionally, agencies have tried to write penalty-laden contracts, but this often incentivizes vendors to deliver the bare minimum that meets the letter of the contract. A well-written Agile contract uses the power of competition to keep suppliers eager to deliver software that works.

For an Agile procurement to be successful, the public sector business users and the Agile developers should work harmoniously. This means bridging a culture gap.

Meanwhile, vendors can set expectations in stages, nipping growing demands in the bud, educating the agency as to how much each feature or additional request might cost.

The contract gap

In traditional procurement, the contract covers everything—prices, delivery, and system performance. But in Agile procurement, the final product emerges through a joint effort during the process. The contract isn't really the key to success, contract *management* is. In many cases, governments aren't used to providing the sort of ongoing input and support through contract administration, creating a gap.

A procurement officer can't just award a contract and expect successful delivery. Regular engagement of the business owner and procurement officer throughout the process is essential. After all, Agile was originally created as a method for in-house software development in the private sector. This meant that the "business users" and the "IT developers" worked for the same company. Agile wasn't envisioned as a contracting approach. Part of the Agile process is frequent—as often as every two weeks—demonstrations that share progress and challenges on the software. Intimate, hands-on involvement is critical to monitor progress and avoid unpleasant eleventh-hour surprises.

The risk gap

In a similar way, traditional government frameworks rely on contractual safeguards to minimize risk to taxpayers. Even today most public software buyers want a contract that spells out in minute detail precisely what the application will do, its total cost, and the delivery date. These protections are comforting to all involved—though they rarely translate into reality.

Procurement officers face a serious dilemma. Even if a "contracting shop" is on board with the Agile philosophy, many simply lack the experience to mitigate risk in an Agile environment. Moving from the theory to practice can be tricky. Contracting officers (COs) need viable options.

COs have been trained to hold contractors accountable for nonperformance through "hooks" in the contract. Strict definitions clarify what counts as nonperformance in terms of time, cost, and scope. Agile can't operate well with so many constraints—to some extent, you have to have trust in your partner.

Trust is a risk. The CO would take the hit if a project went sideways. So, while COs can work with a team, they still often reflexively shrink away from the trust that their position now requires. One of the key roles the CO can play is to make sure that the contract is being monitored. A "trust and verify" approach can protect taxpayers and lead to working software. But it makes new demands and requires a new skill set for COs.

Recognizing this learning curve, the United States Digital Service has published the TechFAR to actively encourage less strict interpretations of contracting rules, to ensure that COs have the flexibility and skills they need.[2] COs may feel suspicious of talking with contractors about problems before writing requests for proposals. They don't want to seem to favor any specific bid. However, informal conversations with experts can help COs understand the possible solutions and write more sophisticated contract proposals. There are several tools available to help COs navigate unfamiliar Agile terrain.

Working with a vendor to produce great software using the Agile methodology requires a true working partnership. Technical development doesn't happen in isolation, but in close and frequent connection with business users and subject matter experts. Bridging the culture gap between government and Agile tech providers is key to fostering success.

ENDNOTES

1. William D. Eggers, *Delivering on Digital: The Innovators and Technologies That Are Transforming Government*, (New York City: Deloitte University Press, 2016), pp. 84–89.

2. United States Digital Service, "The TechFAR handbook for procuring digital services using Agile processes," https://playbook.cio.gov/techfar/.

John O'Leary is a researcher with the Deloitte Center for Government Insights, where he oversees state and local government research.

William D. Eggers is the executive director of Deloitte's Center for Government Insights, where he is responsible for the firm's public sector thought leadership.

Agile in Government

Scaling Agile for government

Using Agile on large, complex projects in government

By: Deborah Sills, Kevin Tunks, and John O'Leary

Can Agile work at government's scale?

GOVERNMENT is increasingly looking to use Agile to quickly deliver technology that meets users' needs. But what about applying Agile in the kinds of large, complex IT projects so common in government?

Scaling Agile presents unique challenges. A big project can require multiple teams. So how do you fit the pieces together, particularly if the final design isn't fully envisioned at the start? How can you get the same sort of quick decision making that allows small-scale Agile to rapidly deliver working prototypes? How do you ensure that interdependencies between teams are accounted for without creating the sort of bureaucracy Agile seeks to eliminate?

It is true that Agile often involves small project teams—often under 10 people—working on narrow projects with timeframes measured in weeks or months rather than years. The Agile approach, however, can also be used on large megaprojects within sizable organizations—but it can be tricky. Agile is more than just a way of developing software; at its core, Agile is about creating high-performance teams, and as such can be well suited for use within large government agencies. Agile is about bring

ing out the very best in people, both on the business side and the technical side, to work together to solve real business problems.

Building a successful Agile model

Successfully building software through Agile relies on three elements. First, a clearly defined set of business problems and a vision of what's needed to solve those problems are important. Second, because business leaders and technologists generally bring different perspectives and working styles to the task, steps should be taken to ensure their viewpoints are integrated. Third, the creation of the solution should be an evolving journey between the business side and the technology side who collaborate to constantly redefine the best available outcome as quickly as possible. As a result, while there are various "flavors" of Agile that employ various methods and implementation approaches, they all share certain characteristics, including:

- Delivering capability in short, "time-boxed" iterations
- Encouraging continuous learning and embracing the inevitable changes in requirements that result through the process
- Driving an active partnership between the government agency, relevant IT groups, and vendors to co-create the best possible outcomes within time and budget constraints

These Agile characteristics can challenge traditional government systems, particularly on larger projects. In essence, you want to create a space to "Let Agile be Agile" while still satisfying the demands of the larger organization. Some of the key challenges that typically need to be addressed in scaling Agile include:

1. Multiyear road maps with many sub-tasks
2. Governance
3. Cross-team dependencies/coordinating multiple work communities
4. End-to-end functionality

AGILE AT SCALE IN GOVERNMENT

Several examples show how Agile is being applied at scale in government:

The FBI case management system: Following the 9/11 attacks, the FBI sought to build a new virtual case file management system to facilitate information-sharing. The project was initially undertaken using a traditional "waterfall" approach. Between 2000 and 2010, the FBI spent hundreds of millions of dollars on multiple iterations of the project with disappointing results. The agency subsequently adopted an Agile approach for the project, a change that required no small amount of culture change.[1] Using the Agile approach, the agency was able to deliver a working system.

Texas Health and Human Services Commission: The Texas Health and Human Services Commission is one of the largest agencies in one of the nation's largest states, with a budget of roughly $30 billion per year.[2] So its decision to embrace Agile as a way of maintaining one of its core IT systems—its integrated eligibility and case management system—required a significant amount of cultural change. After a successful pilot effort, the agency more broadly embraced Agile, including educating staff to deliver better software faster. This investment in people readiness was critical to the effort's success.[3]

California Child Welfare Services: The California Health and Human Services Agency had been working for years on a traditional waterfall request for proposal (RFP), and then chose to switch to an Agile procurement approach. According to Stuart Drown, the deputy secretary of innovation and accountability at the Government Operations Agency, "In 2015, we were about to release an RFP for a $500 million-dollar project. We were on the seventh version, and had worked on it for nearly three years. In a very dramatic switch . . . our leaders decided to go to an Agile model or Agile approach."[4] The agency's approach includes breaking the project into sequenced modules and employing multiple vendors. It will take time to see how this effort at massively scaled Agile will play out, but it does indicate a serious shift in public sector thinking toward applying Agile to large, complex projects.

Agile distinguishes itself in part by ensuring frequent assessment of progress, and these iterative sprints allow for regular demonstrations of business value.

The basic principle is simple: Create an *overall framework* that allows small Agile teams to work in time-boxed, iterative sprints in close connection with business users. Agile distinguishes itself in part by ensuring frequent assessment of progress, and these iterative sprints allow for regular demonstrations of business value. The framework should ensure frequent delivery of working software from across the Agile teams to help demonstrate that coordination is happening at scale.

Regardless of the specific Agile approach, as with any large public sector IT problem, the potential exists for challenges, as mentioned earlier. Below we discuss the four key challenges and strategies to address them:

MULTIYEAR ROAD MAPS WITH MANY SUB-TASKS

In Agile, the precise features of the final product emerge through a process of joint discovery rather than through a theoretical design vision. One of the most challenging aspects of Agile at scale may be the need for multiyear road maps in light of an uncertain, evolving future end state.

Teams can do all the right things, with all the right controls and following best practices, but get nowhere without a guiding vision. With the help of multiyear road maps, project leads can help teams remain on track to deliver business value, constantly refining the product's next minimum viable product (MVP) release. Large organization initiatives often involve multiple objectives with long time horizons. Implementations should inform long-term business and product road maps while maintaining basic Agile principles, including the ability to adjust the end vision as new information comes to light during the build. This requires a "loose-tight" long-term road map that ensures the project provides consistent incremental improvements to business value throughout the long and winding journey.

The road map is primarily geared toward framing the work in executive-level terms. It should help teams understand the sequence of work needed, including balancing the user-visible features with the underlying enabling capabilities—a.k.a. the "plumbing" needed to support working software that solves the most pressing business needs.

GOVERNANCE

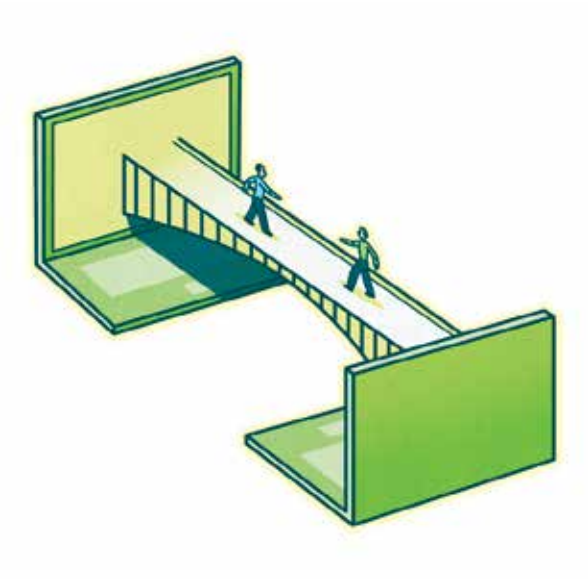
Large-scale Agile implementations involve multiple Agile teams from disparate functions, often in dispersed locations. These resources are often pulled together from various organizational entities, all of which could have a stake in their use. In scaled Agile, it is critical to coordinate these teams under some form of common governance. For example, additional roles are often needed to facilitate communication and resolve conflicts. And strong leadership via the governance framework is critical.

Multiple Agile coaches and product owners should regularly convene in a "scrum of scrums" to facilitate cross-community communication and provide fast decisions. The Agile community of leaders provides a forum for communicating across teams, reporting, and consolidating tracking. This provides a more integrated and up-to-date message framework that allows senior management and interested stakeholders to receive real-time updates about progress during and after attending in-progress product demonstrations of working system capabilities. The regular cadence of demonstrations, along with direct two-way executive and business stakeholder communication and velocity reports from user and technical story completion rates, becomes the primary metric for quantifying progress, and it helps to build confidence that the project as a whole is on track.

CROSS-TEAM DEPENDENCIES/ COORDINATING MULTIPLE WORK COMMUNITIES

Often, a team requires completed components from another team before proceeding further. Such cross-team dependencies create the need for additional coordination when planning upcoming iterations. The ordering of the sprints becomes important, but there should be a balance between flexibility and adherence to the overarching plan.

In Agile at scale, project leaders need to keep an eye on the big picture while the teams absorb themselves into sprints. The most common approach is to divide groups of teams into an "initiative" and multiple initiatives into a project. With different teams working on different features, someone has to ensure that the user interface—the overall "look and feel" of the various components—is eventually consistent. There is a particular risk here that user experience (UX) teams can get ahead of back-end services to the point that expectations held by the product owner are not achievable. Sometimes, addressing cross-team dependencies simply requires diligent communication and frequent validation through demos that exercise slices of functionality. Each Agile team typically holds a brief (usually 15–30 minutes) daily morning planning session to identify the work completed on stories and tasks, work planned for that day, and what blockers are impeding their progress. This information can then be fed up to a regular feature of scaled Agile where all of an initiative's Agile coaches meet for a "scrum of scrums" to discuss progress, blockers, and needs.



One of the most challenging aspects of Agile at scale may be the need for multiyear road maps in light of an uncertain, evolving future end state.

END-TO-END FUNCTIONALITY

Traditional Agile relies on functional and unit testing within sprints as well as other quality control processes. In Agile at scale, an additional testing layer is recommended, which provides end-to-end product testing to identify issues between distinctive components built by Agile teams. The increasingly closely aligned practice of "DevOps"[5] and micro-segmentation architecture[6] are natural partners of Agile management and enable teams to increase velocity. To be sure, "bad code doesn't scale," but perfect code also doesn't exist. The goal is to write good code quickly and receive feedback often. Agile at scale is something of a balancing act that avoids overly prescriptive top-down standardization and allows rapid adoption of best practices, but without creating a free-for-all "Wild West" atmosphere that becomes a trouble-shooting and maintenance nightmare. Coders typically hate it when someone else writes better code than they do. As a result, when teams are cross-pollinated, their natural competitive tendencies are engaged in a healthy way, resulting in more innovation and opportunity via self-organization. Agile at scale may require more guardrails than a smaller project; however, providing latitude for innovation within reasonable boundaries is important.

When multiple teams are building a large system, it is imperative that the approach accommodate end-of-sprint demonstrations that exercise real capability and functionality (albeit incomplete) from the beginning. This favors an approach where vertical (through the tech stack) slices of functionality are sequenced instead of building the system up in layers from the bottom. Delivering in vertical slices also facilitates the ironing out of kinks related to build, promotion, provisioning, and configuration that are sometimes ignored and can lead to surprises at the end, which is the antithesis of Agile.

While some Agile teams release code into production much more frequently than traditional development, others operate over several sprints before reaching sufficient MVP for a new code release. For example, the teams at Texas Health and Human Services issued functional releases frequently, roughly every 5–6 weeks, rather than in a disruptive "big bang" rollout once every 6–12 months.

Agile at scale isn't one-size-fits-all

The term Agile covers a wide array of approaches, and no single cookbook approach can be applied for Agile at scale. Multiple frameworks exist for scaling Agile, but the first order of business may be setting realistic expectations. Large IT projects are more challenging than small projects. Agile can boost your success rate, but Agile isn't a magic wand that makes all the challenges of megaprojects disappear. From vendor selection to staffing, Agile at scale—like any IT project at scale—is a management challenge as much as a technical challenge.

Agile methods require increased involvement from affected stakeholders and, in some cases, dedicated stakeholder resources.

With large projects, governments are experimenting with "modular contracting," which breaks the large task into smaller chunks, as is being done at California's Child Welfare Services (see sidebar, "Agile at scale in government"). While this enables a broader pool of potential contracting partners, the bad news is that not all these

partners may have the wherewithal to capably tackle the coordination aspects of a large project. Moreover, using a slew of small contractors without an overarching "general contractor," can compound the problem of coordinating multiple work streams. From the government agency's perspective, even if multiple sub-vendors are involved, there may be a great benefit in having "one throat to choke."[7] This doesn't mean that the public agency can simply offload the responsibility for success to an external project management office vendor; rather, it means that the government should dedicate needed resources to oversight, and strongly consider a structure that has a central locus for oversight, a role that requires experience in partnering with government on large engagements.

A strong primary contractor can identify issues earlier and help ensure that the government agency and the vendors fulfill their part of the bargain in providing subject-matter expertise, access to needed technology infrastructure, and other elements crucial to success. Agile methods require increased involvement from affected stakeholders and, in some cases, dedicated stakeholder resources. Develop a mutual understanding of the frequency and type of involvement needed to specify requirements, review progress, answer questions, provide feedback, and encourage sign-on. At Texas Health and Human Services, users and subject-matter experts worked directly with the developers, in essence co-designing the software. At the same time, there were strong protocols in place for assignment of developers, coding standards, quality control, and the like.

Much of the success of Agile at scale depends on setting proper expectations and diligently applying project management methods to meet those expectations.

Deborah Sills is a principal with Deloitte Consulting LLP's Public Sector management team.

Kevin Tunks is an information technology professional at Deloitte Digital within Deloitte Consulting LLP.

John O'Leary, based in Boston, MA, leads state and local research for the Deloitte Center for Government Insights.

Agile in Government

Going Agile: The new mind-set for procurement officials

How does Agile change the role of the acquisition officer?

By: John O'Leary and William D. Eggers

GOVERNMENT organizations are increasingly looking to partner with vendors who use Agile to deliver software systems. But for government to successfully take advantage of what Agile has to offer requires a change in mind-set for procurement officials.

For most things that government buys, most people aren't overly concerned with the process of *how* it is made. From office furniture to computers, how the item being purchased was built doesn't really matter.

But if you want to buy software that is developed through an Agile process, you need to alter your procurement process. The procurement officials, the lawyers, and the purchasing agency's business leaders need to embrace a new way of thinking about their role.

Why? Because the Agile process combines design with development and user acceptance. In other words, the software's final design emerges through a collaborative effort between developers and business users. So the traditional procurement approach, heavy on functional

specifications written up front, isn't consistent with the Agile approach.

The new mind-set of procuring for Agile involves many major shifts in thinking. Five of the most important shifts include:

Shift 1: From contract-centered to project-centered

The Agile Manifesto, which kick-started the Agile movement in 2001, explicitly talks about the relative value of various aspects of software development. The manifesto's signers declared that they had come to value:

- Individuals and interactions *over* processes and tools
- Working software *over* comprehensive documentation
- Customer collaboration *ov-er* contract negotiation
- Responding to change *over* following a plan

The signers of the manifesto acknowledge: "While there is value in the items on the right, we value the items on the left more."[1] Those familiar with government will quickly recognize that public procurement is strongly weighted to the items on the right.

For procurement officials looking to use Agile, this has clear and obvious implications. The contract has always been a cornerstone of public software procurement, the document that defines the relationship between a government agency and a vendor. Traditionally, a well-written contract, including detailed specifications, was seen as critical to a successful engagement.

This makes intuitive sense, and a linear, rules-based approach certainly *feels* safe. But experience teaches us that that feeling is often an illusion. The Standish Group's CHAOS Report routinely shows that Agile projects have a higher success rate than linear waterfall projects, and waterfall is more likely not just to go over budget, but to fail in delivering software that works for users.[2] Paper safeguards are of little use if they don't result in successful projects. Good stewards of government focus on ensuring value from an investment. That may mean rethinking the massive requirement-laden contracts of yesteryear.

In the old contract-centric world, the contracting agency spends months, maybe years, soliciting and documenting user requirements, then "tosses" this blueprint over the wall. The vendor collects it, and many months, or years, later, delivers a final product—sometimes deeply flawed. While any number of sanctions in the contract make it appear "tough-nosed," these sanctions may prove difficult to enforce as agencies find that the software "is what they asked for, just not what they really needed." This contract-centric approach too often leads to disappointment and disagreement.

Shift 2: The vendor doesn't run the project, the agency does

In Agile, there is no blueprint and there is no wall. The agency and the vendor partner to build a system. The vendor's assistance may include project management as well as the heavy lifting of development—but throughout the process the government agency actively participates, helping to ensure that the final result will meet its needs.

More than a well-written contract or massive spec document, for Agile to succeed there must be a leader at the agency with a vision for what the application is going to do: Whom does the software support? What is the business challenge being addressed? How will data enter and leave the system? The Agile process turns this vision into working software.

Shift 3: You aren't just buying software, you are entering a relationship

Agile software development requires software buyers to rethink the role of the contract. Instead of serving as the ultimate blueprint for the projects—detailed specs, precise price, firm deadlines—the contract becomes a guide for structuring the relationship between the government agency and the vendor. The shift in mind-set is profound. The agency is no longer looking to buy a "thing"—in this case a new software system. Instead, the agency is entering a relationship to jointly *design and build* a new software system.

Hence, the contract doesn't primarily define the software. The contract's main purpose is to define the expectations of the relationship. This can include pricing associated with a series of performance reviews, defining "done," and clarifying the role of agency representatives and the vendor. One of the biggest challenges for IT vendors can be insufficient access to subject matter experts and managers empowered to make quick decisions—these should be spelled out in advance.

Shift 4: From "lump sum, fixed price" to incremental pricing

No builder can provide a precise fixed cost bid on a house without a highly detailed set of blueprints. The same is true when building software. Because Agile doesn't provide precise specifications up front, it's somewhere between difficult and impossible to calculate an accurate fixed price in advance. This means there will likely need to be some form of incremental pricing, which could entail a time and materials approach, or breaking the project into smaller chunks, or paying for "development points."

Shift 5: From contract management to performance monitoring

Traditional contract management focused on the terms of the contract: Are the correct number of people on the task and are their hours properly documented? Agile leaders can still track that sort of compliance if they'd like, but more important is performance monitoring. One Agile principle states: "Working software is the primary measure of progress."[3] After every Agile "sprint," hands-on review of the software is critical. One project manager told us, "Paper reviews are mostly worthless. You've got to regularly see demos of the software as it is being developed."[4] Bye-bye, Gantt charts. Hello, demos.

The road ahead

Agile isn't the right approach for every project, and there is considerable variation within the various "flavors" of Agile. But for those procurement officials looking to buy software developed through Agile, a significant shift in mind-set is in order.

TOOLS FOR AGILE THINKING

For those looking to dig deeper, here are some additional resources on contracting for Agile development. Also see other articles in our **Agile series**.

18F RESOURCES

18F houses a treasure trove of information on making Agile work in government. The **modular contracting guidebook** helps government executives implement modular contracting at their agencies and enable Agile development. It has also created a **modular contracting resources** page on GitHub. Also check out its **blog** series that covers a wide array of topics on Agile development.

WHITE HOUSE RESOURCE

You can also go back to the 2012 contracting guidance from the White House on **modular development**. The guidance, although published a few years ago, is still relevant and could be helpful for executives who want to dive into Agile.

TECHFAR HANDBOOK

The **handbook** highlights the flexibilities in the Federal Acquisition Regulation (FAR) that can help agencies implement different plays from the digital services playbook. It also focuses on how to use contractors to support an iterative, customer-centric software development process.

AGILE CONTRACTING IN ACTION

See the 18F **Agile Delivery Services Blanket Purchase Agreement** page for more details about how the US federal government is trying to align acquisition practices with Agile delivery practices. The 18F also has an ongoing blog series on **Agile BPA**.

ENDNOTES

3. "The Agile manifesto," http://agilemanifesto.org/, accessed March 28, 2017.
4. The Standish Group, *Chaos*, 2014.
5. "Principles behind the Agile manifesto," http://agilemanifesto.org/principles.html, accessed March 28, 2017.
6. Interview with Kevin Tunks (software project manager), September 29, 2016.

William D. Eggers is the executive director of Deloitte's Center for Government Isights, where he is responsible for the firm's public sector thought leadership.

John O'Leary, based in Boston, MA, leads state and local research for the Deloitte Center for Government Insights.

Agile in Government

The art of points-based procurement for Agile projects

By: John O'Leary and Robert Tross

Project pricing: The perennial contracting challenge

As more government organizations turn to Agile development, contracting officers may find themselves at a crossroads. How can the procurement function effectively support the purchase of software in the absence of precise design specifications up front?

At the heart of the challenge is pricing.

In one direction lies the traditional approach to acquiring software solutions, with its emphasis on functional specifications that are typically tightly defined up front and the ability to write a "fixed price" contract. This pricing scheme can be comfortable, but it isn't always well aligned with the Agile approach, in which a final design emerges over time through a collaborative, iterative process. It also may not be as safe as one might imagine, as change orders can drive the final project price above that initial "fixed" price.

In the other direction, there's a new way of buying software development services, one based on the notion that work can be measured and sold in units known as points. This pricing approach, though typically well aligned with Agile, doesn't have a long-standing track record or framework for execution in government procurement.

WHAT'S THE POINT OF POINTS-BASED ESTIMATES?

In Agile, points are a way to compare the effort between different tasks. The point scale is arbitrary, but it gives a sense of the relative amount of effort required to produce a specific piece of work.

Estimating the points associated with a chunk of work is best approached as a group activity, to reflect the wisdom and experience of the team. One of the best ways to estimate with a group is for each member to make an estimate in isolation, and then to compare estimates. If there is close consensus, great. But if not, a discussion can explore *why* each individual made the estimate he or she did.

In order to accurately reflect the lack of precision associated with point estimates, some teams use a point scale that is a modified Fibonacci sequence, such as 1, 2, 3, 5, 8, 13, 20, 40, 80, and so on. The idea behind this is that the larger a piece of work, the more difficult it can be to estimate the effort required with precision. Instead of wasting time arguing over whether a task is a 16 versus a 17, the team can have the much easier debate of whether to score it a 13 or a 20.

But why use points at all? There are several possible reasons. First, productivity levels can vary, making points a more reliable indicator of effort than estimates based strictly on time. For instance, depending on who is involved, a five-point story might take three days, or it might take six. Using points can also limit extraneous factors, such as pressure to complete a sprint by a certain date, that can disrupt an estimate based strictly on time. In addition, people just seem to be more effective at estimating *relative* values rather than *absolute* values.

A key reason to consider a points-based pricing approach, however, is that, because it is aligned with Agile development, it can facilitate the contracting relationship, ultimately leading to a better value for the contracting agency. We call this the "art" of points-based contracting because there is no cookie-cutter recipe for success. Contracting officers and agency heads need to use their wisdom, judgment, and experience to make sound choices.

Fortunately, the territory ahead isn't wholly uncharted. Agile has enough history in the public sector to provide some insight into the pros and cons of points-based procurement, as well as some practical ideas for getting started.

Aligning pricing and the Agile development approach

Agile is a methodology for putting usable new software in people's hands as quickly as possible. The basic approach is to break projects up into small, time-limited chunks called sprints (see the sidebar "Glossary of Agile terms."). Each sprint addresses at least one user requirement, or story, quickly resulting in a working prototype. Users get frequent demos of the software as it is being developed, with a chance to offer their feedback to the development teams. The teams then generate a new version that incorporates this user feedback. This process is repeated until the functionality is right.

Agile's iterative process can offer a number of advantages. For one thing, it can uncover problems or shortcomings sooner than a traditional "waterfall" approach. Sometimes, people don't realize what they really need until they have something tangible to look at and/or work with. Other times, development teams run into unexpected roadblocks. Agile acknowledges these realities and provides a way to address them before they undermine a project.

Then, there's customer satisfaction. Because users have input during Agile development, the resulting software is typically more in line with their expectations. The Agile method helps users gain value from a project earlier, rather than wait for everything to be delivered at the end—including nasty surprises. Further, an Agile approach can boost confidence in the development team and project sponsors. It can also help set appropriate customer expectations: Based on their experience with past iterations, users learn what the development team is realistically able to do.

The greatest benefit of Agile, however, may be that it makes room for and facilitates change. Few things typically remain constant during the course of a big software development project. If priorities shift, an Agile approach can accommodate the situation with minimal

The greatest benefit of Agile, however, may be that it makes room for and facilitates change.

friction. In contrast, a fixed-fee contract generally operates on the implicit assumption that the original design will meet users' needs and that nothing will change. Sometimes, in fact, this turns out to be the case. However, when priorities do shift, a fixed-fee project can sometimes require post-contract changes, including change orders, and may generate costly late surprises—some of the very issues that Agile is intended to mitigate.

While it may be helpful that Agile can accommodate change, how can a contracting officer price for that? How do you hit a moving target?

Translating Agile to procurement

How can you price something where the final output is expected to evolve? Various pricing approaches deal with this in different ways, each with its own set of pros and cons.

Time and materials: One possibility is time and materials—which can be an effective approach for an Agile effort. But to the government, this approach may feel like writing a blank check.

Fixed fee: It is possible to use a fixed-fee contract for Agile development, but from the contractor's perspective, this can be risky. After all, in Agile, the expectation is that the design will evolve, and if a government agency has free range to add features, the result could be a recipe for scope creep. Moreover, traditional fixed-price contracts lock in functionality—but in Agile development, locked-in functionality specs may fail to account for the type of design evolution that Agile is designed to produce. In fact, an attempt to rigidly specify functionality at the outset is fundamentally misaligned with how Agile builds software.

Small chunks of fixed fee: An agency could break an effort into smaller chunks for the purpose of issuing a series of small, fixed-price contracts, one for each chunk. This approach is somewhat consistent with Agile's incremental development approach. As a pricing mechanism, however, this can be burdensome, as few parties to a transaction likely have much appetite for putting every two- or four-week sprint out to bid. It also might give incumbent vendors an unfair advantage over competitors who lack familiarity with the current project or who do not have an ongoing relationship with the product owner. Alternatively, jumping from vendor to vendor based on the bid on a particular chunk of work can be both administratively costly and risky, given that not only does each component have to work, but they all have to work together. Like a car, software can be built by multiple vendors, but significant work must be done to ensure that all the pieces mesh.

Points-based Agile procurement: This approach is similar to the "small chunks of fixed fee" approach, but it more fully embraces the language, methodology, and working pace of Agile development.

In points-based Agile procurement, development teams estimate project complexity through a system of points based on how much work is required to complete similar tasks. By adding up the points for each story in a project, experienced development teams can gauge the overall velocity of their efforts. For example, if over a series of four sprints an Agile team delivers 80, 100, 120, and 100 story points, its velocity is 100, and the team can reasonably estimate that in future sprints it will likely be able to produce roughly 100 points—all other things being equal (days in sprint, holidays, number of staff, and so on). These points form an arbitrary but internally consistent scale: A 20-point programming challenge should take twice as much effort as a 10-point one. Points put the focus on complexity, not on the time required. After all, some teams may be more productive than others, taking less time to complete the same task.

Points-based procurement, at its core, is all about empowering delivery teams—specifically, the product owners—and breathing flexibility into the Agile process. The agency ultimately governs how much it is willing to pay for a given number of development points, and then empowers product owners to deal with the day-to-day complexity of determining what trade-offs and decisions to make with respect to product features.

If contracting teams acquire software solutions based on points—under a blanket purchase agreement, for example—they can largely avoid change orders, budget overruns, and other costly distractions. The focus instead shifts to the programming challenge and the desired functionality. Long-term, points-based procurement can help alleviate other classic challenges of

long-term development planning, as it places more authority for planning directly into the product owner's hands. Long lead times (and their attendant risks) give way to sprints. Gantt charts and arbitrary deadlines are replaced by iterative processes, demos and other Agile ceremonies, and feedback loops. Change resistance and eleventh-hour training yield to ongoing, high-touch collaboration between the business's product owner and the development team, whether internal, external, or a combination of the two.

Put another way, points-based procurement allows government contracting officers to acquire the *capabilities* of an Agile team, without caring about whether that capability is delivered by 8 people or 12 people, and without relying on a specific set of requirements or functionality. This way, executives could gain an extension of their own teams who can direct more attention to the business activities that align with the organization's mission.

Ultimately, the use of points-based procurement can facilitate fixed-price purchasing without the need for creating functional specifications. In other words, a government agency and vendor with a good track record could agree on a fixed-fee contract that buys a certain number of story points without dictating what functionality those points will be put toward. Then, the client's business leaders and Agile team can prioritize what to work on based on "bang for the buck"—that is, which stories will produce the most business value relative to story points consumed.

Challenges of a points-based approach

None of this means that traditional requirements-based software procurement is about to disappear. Specificity is a response to accountability, after all, and hours and days are typically easier to measure against external requirements. At some point, moreover, the subject of price does need to come up. How much a "point" costs relates back to the cost of the developer resources.

The biggest barrier to points-based pricing may be that a successful points-based approach requires a shared understanding between delivery and procurement—meaning that the procurement team should develop a good understanding of how points relate to effort. This means not only becoming comfortable with concepts such as "velocity," "story," and "epic," but actually becoming familiar with how much technical effort is required to accomplish certain tasks. Is it reasonable that building a user interface should require 15 points? Why would adding a mobile capability to an app require 40 points? Only experience developed over time can allow someone to be confident that these point-based fees are reasonable.

In addition, there would need to be an education effort for political leaders, so that they can feel secure that the public interest is being served and that the agency is receiving appropriate value for the price it is paying.

In sum, there are many ways to price a contract for Agile development. You can use fixed fee. You can use time and materials or cost-plus contracts. You can break a big project into a series of small fixed-fee chunks. You can use a points-based approach, including one that includes a firm fixed price per user story point delivered. Or you can start with time and materials for the first few sprints, and then move to a "points-based" system for later sprints. All these approaches have benefits and limitations, and the best pricing structure could depend on many factors: the nature of the work, the organization's appetite for experimentation, and the level of trust between an agency and its vendors, to name a few.

Setting the stage for points-based procurement

With all that said, if Agile is gaining steam in your organization and if your procurement organization seems open to the approach, you likely have a few openings for bringing points-based estimation into the public procurement realm. The key may be to take a stepwise approach, documenting effective practices as you go. Steps to consider include:

Get firsthand experience. Before attempting points-based procurement, focus on learning how to work with Agile contractors. Once your contract team develops a history of making deliverables in an Agile way, they—and you—could be more comfortable buying a design that's not completely laid out.

Look to long-term projects. Even under a traditional contract, longer-term Agile projects can provide more opportunity for government employees and contracting personnel to work together and understand their own internal velocities. At the same time, conversations would tend to be less about price and more about the complexity of a piece of functionality. That depth of understanding can form the basis for making points-based procurement worthwhile.

Keep it simple—at first. Consider starting your points-based procurement journey with a time-and-materials contract for Agile development. From there, you might move to fixed-price contracts, and finally on to points-based contracting as the contract team's experience grows. Another option: Consider using a points-based contract for a project that involves a single technology platform with a sophisticated software development kit. The lessons from this experience might then extend to projects with more diverse technologies or the need for more custom development.

Although Agile is no stranger to government organizations, a points-based measurement and pricing technique can be a significant departure from the traditional procurement mind-set. However, that needn't stop contracting officers from exploring ways to apply what Agile has to offer. The results could well be greater flexibility, transparency, and value for the public.

GLOSSARY OF AGILE TERMS

Ceremony. A meeting to mark a milestone in the development effort, such as a kickoff meeting at the beginning of a sprint or a demonstration of the team's work at the end of a sprint. ("Join us at the ceremony on Friday to take a look at the new credit card payment feature.")

Epic. A series of user stories. ("The first story is to let constituents pay online with a credit card. The next story is that they can use other online payment platforms. Then they'll be able to search for the records they want. Then they'll be able to download each record in PDF format.")

Points. A measure of the relative complexity of a sprint, as assessed by the development team based on experience. ("It generally takes a little over two weeks to build credit card functionality into a website like this. Given that history, this sprint is worth 22 points.")

Product owner. A representative of the contracting organization who writes stories and works closely with the vendor to implement them. ("The product owner has been on-site with the development team, helping them understand the business objectives.")

Sprint. A brief span of development work, usually lasting less than six weeks. ("In this two-week sprint, we'll build the capability for this website to accept credit card payments.")

Story. A high-level description of a user requirement. ("As a constituent, I want to be able to pay for my government services online with a credit card.")

Velocity. How quickly a development team completes work, as measured in points. ("We produce an average of 10 points' worth of work each week. So with that velocity, this 22-point story will take us just over two weeks to do.")

DISCOVER MORE ABOUT AGILE PROCUREMENT

There's a growing body of literature on the subject of points-based software acquisition. To learn more about this rapidly-maturing area, consider starting with the following resources:

ACT-IAC

The American Council for Technology (ACT) and Industry Advisory Council (IAC) published a document for the procurement of Agile IT services.

18F

The US General Services Administration maintains a website of information on modular contracting and Agile development. Here, among other things, you can find the Agile Delivery Services Blanket Purchase Agreement describing how the US federal government is trying to align acquisition practices with Agile delivery practices.

THE WHITE HOUSE

The contracting guidance that the White House published in 2012 still seems relevant for agencies who want to dive into Agile. Another White House publication, the TechFAR handbook, explains how government organizations can use Agile principles to acquire digital services in the context of the Federal Acquisition Regulation (FAR).

John O'Leary, based in Boston, MA, leads state and local research for the Deloitte Center for Government Insights.

Robert Tross, of Arlington, VA, is a senior manager with Deloitte's Federal/Government Technology practice.

Agile in Government

Successful Agile in government

Supporting the product owner

By: John O'Leary, Roberto Cota, and Gabrielle Otis

WHILE a growing number of government organizations are looking to take advantage of the benefits of Agile development, not all of them may be fully prepared for the level of commitment needed from their own people and stakeholders to enable success. This paper looks at some common pitfalls of under-resourcing and under-supporting one of the most crucial roles in Agile, the *product owner*.

The product owner: The linchpin of Agile

The product owner (PO) is a role unique to Agile, and a capable, well-supported PO is critical to the success of Agile development implementations.[1] The PO has three key functions in the Agile process:

ROLE DISTINCTIONS: PRODUCT OWNER VS. PROJECT MANAGER

Product owner

- Is closely integrated with the development team
- Serves as the "eyes and ears" of the agency and the end-user stakeholder groups
- Is intimately familiar with the intended use of the software
- Acts on a blend of the current end-state vision of the software as well as the near-term results of the spring team(s)
- Is knowledgeable about the end users' culture, work habits, and workflows
- Has access to senior decision makers
- Has the authority to make on-the-spot lower-level decisions that do not require escalation
- Has enough "political capital" to guide the project team through the inevitable challenges, and to maintain the focus of senior leadership in light of competing agency priorities
- Communicates effectively, especially to senior managers and end users, acting as a cheerleader for the project or an advocate for the development team, or raising an alarm early if the project veers off target

Product manager

- Is skilled in traditional project management
- Acts primarily on collected data and forecast estimates (as opposed to deep team interactions)
- Focuses on tracking to a project budget and timeline
- Focuses on reporting and documentatio
- Focuses on resource planning and timing of events such as user testing, training, and implementation rollouts

1. The PO is responsible for representing the business's and end users' interests to the development team, including managing the backlog and prioritizing the work of sprint teams. The focus here is on usability and adoptability.
2. The PO plays a key role in the milestone demos that an Agile team will be providing throughout the process—acting as both the recipient of the demos from the development team as well as conduit, participating in demos to senior management. The PO in this role may either copresent the features as they develop or act as tour guide for senior management, answering key questions to help fully illuminate how the project is coming along.
3. The PO is also responsible for ensuring that decisions on choices that arise during the development process get communicated to the development team in a timely manner. This doesn't mean that the PO has to make every call on his or her own—few POs will or even should have that sort of authority—but they do need to be able to command the attention of senior management to streamline the decision-making process rather than allow lingering questions to fester.

For those new to Agile, the PO role is often confused with the more traditional project manager role—but they are different. While the project manager focuses on resource management, keeping the project on track in terms of time and expense, the PO maintains a laser focus on ensuring that the software delivers the business benefits envisioned by the agency. More than anyone else on the public sector side of the equation, the PO should understand the ins and outs of how the software will actually work and provide value to end users.

Why can the product owner role be especially difficult in government?

The PO role is challenging in almost any Agile development effort, whether public or private. A good PO has to bridge the gaps between business users, developers, the IT department(s), senior management, and even external stakeholders such as advocacy groups and third-party providers. A PO has to be part business analyst, part technologist, and full-time diplomat—a communicator who commands respect throughout the organization, while often operating without a large staff, extensive authority, or a lofty title.

While the PO role is a linchpin role in most Agile projects, it may be even more challenging in the public sector, for a couple of key reasons. The first is that skilled managers of the sort described above are scarce resources in government—or in any organization—and an employee who fits that description is almost always engaged in a critical role already. It is very hard for senior management to reallocate such a talent—but it is crucial for the success of an Agile project.

The second is that the PO will have to battle an environment more rule bound than the private sector. While politics and bureaucracy do exist in the private sector, they are rarely as intense as their public sector counterparts. Unlike private entities, our democratic institutions require a level of transparency and fair process, as well as protocols to limit the likelihood of corruption. This red tape, some of it legislatively required, can hinder execution. In addition, the public sector often presents a high number of stakeholders that have to be consulted throughout the process. In a private organization, many of these obstacles either do not exist or can be eliminated by a single stakeholder such as the company's CEO.

Perhaps the most common Agile mistake is to name someone as PO without relieving him or her of other duties. The truth is that a PO needs full-time (or very near full-time) focus to successfully manage an Agile implementation; it's very difficult to effectively handle PO responsibilities while holding down a "day job." Yet too often we see decision makers short-changing the time commitment required of a PO, with predictably bad results. Sometimes POs are overwhelmed or frustrated, doing neither of their jobs particularly well. Sometimes the PO simply moves on to a different role, leaving the project in limbo. And sometimes the PO is able to

Perhaps the most common Agile mistake is to name someone as PO without relieving him or her of other duties.

successfully juggle the demands of two roles—but, as months pass, eventually succumbs to "product owner fatigue."

PO fatigue can afflict anyone in that linchpin role, but it is particularly acute when management has insufficiently cleared the plate of the employee being asked to take on the challenge. Too often, we hear of cases where management takes the view of "Everybody is stressed, everybody has a lot of meetings—deal with it." Then, as the country song says, you'll only miss them when they're gone.

Another mistake can be to fail to support the PO, most often by failing to establish mechanisms that allow critical decisions to be made quickly. In this scenario, the PO and developers identify "either/or" decision points: The product will either be made to work on mobile devices, or it won't. It will allow employees to change records with an audit trail, or it will require manager sign-off.

It will allow field workers to override assignments, or it won't. These decisions are often "above the pay grade" of the PO. The challenge for the PO can then become, who needs to be involved in this decision, and how can I get them to make an informed decision in a timely manner?

It is at this point that the PO can most often suffer a metaphorical nervous breakdown, because the vast majority of these decisions mean trade-offs, and these trade-offs create winners and losers. In government, the complex web of checks and balances means that different stakeholders will likely have different perspectives—but there is often no clear mechanism by which these different views may be reconciled. It often isn't even clear initially who needs to be involved in making the call. Thus a PO may scurry about on an endless loop, chasing down senior officials from all over government—from secretariat-level officials in the business unit to county administrators to IT architects—but no decision gets made. Or, even worse, decisions get made and then unmade. Further, all of this takes a lot of time, and the "Agile" project may slow down or stall altogether, depending on the nature of the decision.

Supporting the product owner: Strategies for senior sponsors

One approach to mitigate this challenge is to establish a senior steering committee, comprising several senior representatives empowered to meet and deliver a decision. The PO can raise these "either/or" decision points, present the pros and cons of various options, and help the senior steering committee engage in dialogue to reach a decision. There still remains the challenge of timeliness—the group isn't much good if it takes a month to get them all together—but having such a committee can be a huge benefit to both the PO and the project.

The senior steering committee can also act as a sounding board for project demos throughout the milestones of an Agile project. In Agile, actual demonstrations of working software are typically the key measure of progress toward success—as opposed to Gantt charts, Excel spreadsheets, or multicolored graphs showing a work stream as red, amber, or green. However, to be meaningful, these demos need to be seen by the right people. While the PO would review these demos, it is important that a broad cross-section of senior management likewise has a window into the project. This not only reduces the likelihood of unpleasant surprises down the road, it also allows for creative input along the way. With the PO at the table along with representatives of the development team, these demos can not only provide senior leaders with a better understanding of how the project is progressing but also offer an opportunity to discover previously unthought-of ways the system could work even better.

In addition to the senior steering committee, many successful Agile projects have one or more key senior "sponsors." Whether formally named as such or not, these senior leaders make themselves available to POs when the latter hit organizational roadblocks. Sometimes the PO just needs to be able to walk into an office or pick up the phone and make a quick call to someone who knows what is going on, and who can provide guidance without having to wait for the next monthly meeting.

Without sufficient leadership support, the PO may have a difficult time helping to guide an Agile project to a successful conclusion.

The product owner support structure: Context for success

In addition to leadership support, POs need a support structure to help them succeed over the long haul. While some Agile projects can be completed in just a few sprints, more often, a project can stretch out over months or even years. Or, even if an initial project is completed quickly, additional features are added that effectively extend the life of the project.

There are several ways to support a PO throughout the journey.

TOOL 1: CREATE A PO JOB DESCRIPTION

"Don't worry, of course you'll be freed up from your regular duties. We aren't going to name someone to replace you since this is just a temporary assignment, but surely so-and-so can step up for a while." Assurances of this sort should set off alarm bells for any would-be PO.

A written role description for POs can give them confidence that they are being asked to focus on this new role. This would also help clarify the PO role responsibilities. Often, the PO would keep the same manager, so it is especially important that expectations are negotiated and articulated in advance. In addition, POs are typically go-to people in an organization, and, without a written description, people might continue to go to them with all the old problems, and expect them to keep the shop running as they always have. While a 100 percent focus on

the new role might be ideal, an 80–90 percent commitment is usually a good compromise that reflects reality. A word of caution, however: For large implementation projects, the PO should really be engaged 100 percent of the time and, in some cases, will require a support team.

While the coding portion of an Agile project may be swift, the PO's role could last two or more years, stretching from planning through implementation. It is critical that the new role is communicated to the rest of the agency team, and one of the best ways to do that is through a new PO job description.

TOOL 2: CONSIDER COMMUNICATION AND CHANGE MANAGEMENT SUPPORT

Because the PO is often the key conduit of communication between the business and the development team, it is critical to consider how much communication might be required. For example, if a state is implementing new software in a county-administered system, the PO may need to both gather input from and communicate changes to all the county stakeholders—which could be scores of leaders across the state, all with different concerns and priorities.

In the absence of readily available information, there can be a tendency for everyone to call or email the PO directly. Instead, agencies should think strategically about which messages need to be communicated by the PO, and which directly by the leadership team or senior steering committee. Because the PO's focus should be on technology delivery and adoption, it is typically important that change management, communication, process redesign, and training not all fall on this individual's plate without proper support.

Moreover, even if the PO is a strong communicator on an individual level, she or he likely has little or no experience building a communication plan and even less capacity to execute that plan. Large-scale communications might include regular email updates, periodic conference calls, and training calls. If the new system will have impact on citizens, the press may have inquiries, and a PO can easily become overwhelmed by the amount of change management internally and externally. In these situations, establishing a separate communication and/or change management role and delegating those responsibilities to these professionals could alleviate the load on the PO. The PO would still be accountable for the communications that go out as well as the change management strategies that get implemented but would have the needed support.

TOOL 3: CREATE A TRANSPARENT PROJECT MANAGEMENT ORGANIZATION (PMO)

The PMO in an Agile project can be large or small, run by a vendor or internally, and may play a critical or more secondary role. However it is established, a PMO website can leverage technology to make information accessible to key stakeholders, and the PMO can be a great way to communicate mass project updates. If stakeholders feel in the dark about the process, the negative impact of any problem could be multiplied.

Part of transparency is committing to the use of clear, jargon-free language in PMO reports. In too many cases, PMO reports end up going unread, such as the famed "TPS report" from the movie *Office Space*. Clear reporting from the PMO and other collaboration tools can streamline operations and serve as the central point for project progress, burndown rates, and so on.

TOOL 4: HOLD AN OPEN PO SELECTION PROCESS

The PO role can often be seen as a plum assignment, showing the confidence that an organization is placing in the individual. For that reason, it is often best to invite interested parties to apply for the role. In the absence of a formal process, those not selected may feel overlooked or jealous. In addition, through a selection process, you may learn of others who are willing to step up to this challenge and thus might be able to play a supporting role. An open call for candidates, including external candidates, can help ensure the best candidate is selected while typically generating goodwill.

Even if the project sponsor has "the perfect person" in mind, it is often best to go through a process. Not only does it embody a principle of fair treatment, it also provides an opportunity to meet other possible contributors and explore key questions about the structure of the role.

TOOL 5: IDENTIFY BACKUP AND COVERAGE

The PO role is critical, but POs need vacation, get sick, and have unexpected emergencies, just like anyone else. Because of this, it is typically important that a backup be in place—someone who is close enough to the project to keep the wheels turning during brief absences. In larger projects, there might be a need to establish this position more permanently, as a deputy PO who can also help the PO in day-to-day activities.

TOOL 6: COLOCATION AND MOBILE SUPPORT

Agile puts a high emphasis on regular, face-to-face communications. If possible, it may be best to colocate the business user group and development team. If that is not possible, POs often need space in both locations to accommodate the working hours of a vendor project team. Indeed, committing the resources to getting a good space for the team can be important, as public agencies often operate near capacity for space. In addition, it can help if the PO has access to mobile technologies such as a laptop and smartphone to enable close communication at all times. This may not be the standard approach for the government agency, but is key to enable the work demands of the PO.

TOOL 7: TRAINING

If the new PO needs training in Agile, a software language, or project management, try to get these trainings done prior to the project launch. Also, be prepared to commit training resources to the other Agile team members as well as to key members of the senior leadership team and business stakeholders with whom the Agile team may regularly interact, including the contracting office. Given the unique terminology of Agile (scrum master, velocity, story points), those without some training may feel excluded and resentful of the Agile team, setting up barriers to collaboration.

Table 1 lists the possible project features that will help POs achieve success.

Table 1. Product owners in Agile

There are many different approaches to Agile, but most of them will entail similar functions for the agency's product owner. A successful Agile project typically needs rapid decision making and responsiveness to dynamic requirements.

Realistic expectations	The PO needs sufficient time to devote to leading the Agile project.	It will always be tempting to think the PO can lead the Agile project while also maintaining prior responsibilities. This can be a big mistake.
Sufficient support	The PO may need support for communication, training, and other change management activities, as well as support during absences.	With all the focus on building software, it is easy to underestimate the resources the PO will need to ensure successful adoption by staff and customers.
Senior leadership access	The PO may need access to key sponsors who can use their authority to help clear roadblocks.	Government often has a multiplicity of key stakeholders. Consider establishing a senior steering committee to help gather critical leaders.
Decision-making authority	To be successful, Agile teams need quick turnaround on decisions.	Multiple stakeholders make decision making difficult. Leaders should empower the PO to make most decisions close to where the actual work is being performed. A clear escalation path should be defined to quickly resolve tougher questions.
Governance	A framework is needed for reviewing milestone demos, making key decisions, and resolving issues with the vendor.	Getting business leaders, technical architects, developers, and the client all on the same page isn't always easy—but it is important to delivering ultimate success.

Agile success comes with commitment

Why are more and more government entities looking to use Agile? Perhaps some have seen Agile deliver success in prior jobs in the private sector. Or perhaps government leaders have had a bad experience with Waterfall, one in which a big, thick document of requirements—often challenging to read, or at least open to different interpretations—was handed over to a development team, which then worked in isolation for months, only to deliver software that failed to satisfy user needs and hence required a long, laborious, contentious rewrite.

But for Agile to succeed, the public sector partner should be an active participant throughout the development process, with the PO playing a big role in that. The PO typically needs the support and attention of senior leaders; access to support for change management, communication, and training; and the ability to bridge the gap between users' needs and the development team.

ENDNOTES

7. Though a product owner may not be a named role in all variants of Agile, most all will include a role that covers these essential functions.

John O'Leary, based in Boston, MA, leads state and local research for the Deloitte Center for Government Insights.

Roberto Cota has over 18 years of experience delivering large-scale enterprise transformation projects, 15 of which have been spent in health and human services agencies across the United States.

Gabrielle Otis is a manager with Deloitte Consulting LLP.

Agile in Government

Bringing Agile benefits to a waterfall project

Visual design simulations

By: Phong Khanh Huynh and John O'Leary

The best of both worlds?

Not every project is a good candidate for Agile software development approaches, and not every organization is interested in undergoing the sort of deep cultural change needed to adopt "pure" Agile. For those trying to realize the benefits of Agile within a waterfall project, simulations may be the answer.

A simulation is a compromise between the purely "paper" reviews of traditional waterfall and the full-on demonstrations of working software that are the hallmark of pure Agile. A simulation gives management a tour of a visual prototype of the application, showing them how various screens look and feel, and allowing them to do a hands-on walkthrough of the process workflow. This helps mitigate the risks of last-minute surprises that can occur with waterfall approaches.

These simulations aren't full-blown working code. They are merely visual prototypes, and are not hooked up to a database or test environment. However, they do lay out in visual fashion the basic steps the system is performing. In a simulation, for example, after a record is retrieved in a case management system, a worker can see what would happen if the information was approved by

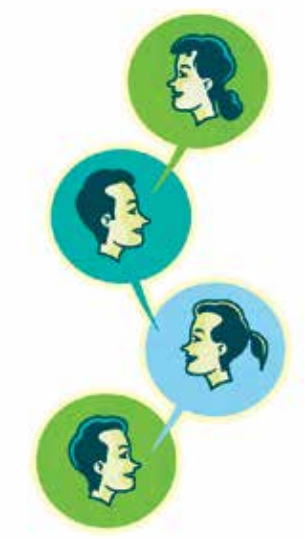
WATERFALL AND AGILE METHODOLOGIES: PROS AND CONS

	Pros	Cons
Waterfall	• Allows for budget predictability • Fits traditional procurement systems • Mitigates demands on client staff	• Reliance on paper design documentation reviews can mask issues • Difficult to "see" how the system is shaping up until very late in the project • Poses a large risk of rejection during user-acceptance testing due to the risk that the system is not what was expected • Massive written specifications are difficult to digest • There could be gaps between what the documenter intended and how the readers interpreted it
Agile	• Early and frequent user input from design through implementation reduces the risk of last-minute surprises	• Procurement and payment approaches are less well established • Places high demands on client personnel • Requires significant cultural change for true Agile adoption

the system, or what the next steps would be if there is a problem with the information. While the actual coding for all of the various exceptions is not done for a simulation, this visual walkthrough often gives management enough insight to offer important feedback early—when the project can make course corrections without incurring significant delays or cost impacts.

Simulations can be a marked improvement over the functional specifications and design documentation required for waterfall, which can run upward of thousands of pages and be open to multiple interpretations. It is difficult, if not impossible, to read and absorb 1,000-page design documents and tie them together into a working understanding of how a system will function once developed. In contrast, when walked through a visual simulation, project staff and sponsors are able to ask important probing questions, and they come away with a greater awareness of whether the design is on the right track. Granted—unlike with other, "purer" forms of Agile—building simulations can require a significant documentation effort in itself. But the advantage is that simulations can help to ensure that the product works in accordance with expectations and not merely in conformance to written contractual requirements.

Simulations are significantly more robust than mere wireframes, and they do represent a certain amount of investment. However, they can provide enormous value by reducing the possibility of painful and costly surprises down the line. As an added benefit, simulations can be used to identify gaps in business logic. They provide a way for teams to collaboratively review progress early and throughout the design phase—an approach consistent with Agile development principles.



SIMULATIONS: AN EFFECTIVE TOOL FOR WATERFALL PROJECTS

Visual design and system simulations can enhance a waterfall project in many ways, including helping to:

- Gather meaningful, collaborative reviews of a design as it is emerging
- Prompt course corrections early in the process
- Encourage client sponsor/executive buy-in
- Enable earlier development of training plans and materials
- Prompt new "what if" thinking that often leads to broader testing scenarios or other improvements
- Improve user adoption and acceptance. Giving users the opportunity to "see" and provide input on the new system can lead to a more satisfactory end product. At the same time, users' participation in the process will help in gaining their buy-in during the later rollout

A recipe for success

While simulations are a useful tool, for them to be effective, both the client and the vendor need to use them within a process that allows for their potential benefits to become reality. When considering incorporating simulations into a waterfall project, here are some tips to keep in mind:

- **Use simulations throughout the design phase**. Set the up-front expectation that, in addition to the usual "paper" progress reviews featuring Gantt charts and spreadsheets, simulations will be part of the review process at predetermined regular intervals. These events must be participatory and in-depth, not a superficial show-and-tell, and project stakeholders—from senior leaders to front-end users—must provide input.
- **Involve product owners**. Plan to have product owners highly engaged during design and "test-review" simulations.
- **Plan for changes**. Since simulations should prompt significant thought and discussion, it is critical to build in cycle time to account for enhancements.
- **Control the backlog**. As with any project, scope creep is a possibility. To control this, the discipline needed in the initial phases must apply in the later phases as well. Management will likely need to make trade-offs to keep the project scope under control.
- **Don't forget end-to-end performance**. Test the simulations in modular fashion, but don't forget the importance of end-to-end performance. This includes identifying cross-dependencies, integrating with third-party vendors, integrating converted data, and testing across all aspects of the system.
- **Simulate the important parts**. Stick to simulations for core workflows and high-traffic areas. Remember, 20 percent of the system will typically cover 80 percent of the functionality.
- **Build basic business logic**. To show how various sections are linked, it is important to have some business logic built into simulations. This enables reviewers to understand how work flows through the system, rather than just seeing it in isolated pockets.
- **Don't throw away the design documents**. Rely on traditional specifications and design documentation for nonfunctional requirements and aspects of the system that cannot be simulated, such as business rules, interfaces, and other technical system components. In essence, a simulation's role is to serve as design documentation for the user interface, the navigation field definition, and the association with the underlying data model.

Some cautions

As with any tool, simulations won't solve every problem or clarify every gray area. For example, in any waterfall project—with or without simulations—if the specifications and requirements are not well-documented and clearly defined, the project will struggle. While simulations are helpful in allowing users to see the workflow

and interact with a visual prototype, it is important to remember that they aren't full-blown operating software. They will not allow users to test how well the new system connects with data sources, or the effect that volume may have on system performance. The testing life cycle remains critical to validating those aspects of the project and ensuring adherence to written design specifications.

Waterfall with simulations

Since waterfall with simulations is based on waterfall, it has most of waterfall's advantages: It is both familiar and predictable, and mitigates demands on the client workforce. That said, there are also some key differences, both positive and negative.

The primary advantage of waterfall with simulations over "pure" waterfall is that simulations can provide early confirmation that the project is heading in the right direction. Additionally, the ability to walk through visual demonstrations can allow users and developers to identify additional innovations and improvements not contained in the original specifications. The use of visual prototypes limits late surprises and gives the project a head start on familiarizing users with the system, gaining client executive/sponsor buy-in, and developing training.

On the other hand, while useful, simulations aren't fully operational, and true end-to-end functionality won't exist until late in the project. In addition, not all of the work involved in building simulations will directly contribute to the actual software build. While the screen design and some other aspects of a simulation can readily translate elsewhere and be "reused," some of the effort of building a simulation will go unleveraged.

In essence, a portion of the cost of building simulations can be thought of as an investment in a much more robust form of progress reporting. Simulations are far less subject to differing interpretations than written reports, which is one reason why they are so effective in limiting risks and encouraging more helpful user input. But to reap their benefits, organizations need to allow for flexibility and be prepared to dedicate resources to course corrections and enhancements that arise through the simulation process.

Phong Khanh Huynh is a Deloitte Consulting LLP principal based in the Costa Mesa, CA office.

John O'Leary, based in Boston, MA, leads state and local research for the Deloitte Center for Government Insights.

Agile in Government

Managing an Agile program? Consider an AMO

By: Sam Martin and Kareem Abdelsadek

A COMMON misconception is that adopting Agile means embracing chaos, where planning, well-defined processes, and project oversight are abandoned in favor of uncontrolled madness. Given that projects launch without detailed specifications and comprehensive schedules, management may fear that an Agile project will be the IT equivalent of the Wild West.

After all, how can you manage a project in which the final output is not based on an initial design, but emerges over time through iterations? Without detailed specs and hard deliverables, how can you ensure the accountability of contractors? Senior leaders also tend to be reluctant to abandon the comfort of traditional project reporting due to concerns that a lack of control could result in a costly failure.

The concern is legitimate. Indeed, if Agile is attempted without *proper* oversight, a project can decay into a chaotic and unproductive collection of activities with disappointing results. Agile projects can fail like any other.

If done well, however, Agile methodologies can provide greater visibility for management into how a project is progressing, more enforceable process controls for how the effort is conducted, and a stronger correlation between a customer's needs and the work delivered. And

Agile can provide all of this while also cultivating a culture of innovation and employee empowerment.

An "Agile management office," or AMO, represents a way to transform the traditional program management office (PMO) to better support programs executing (or experimenting with) Agile.

Common pitfalls of a traditional PMO

The traditional PMO took on its contemporary form in the 1950s, and was established to centralize management of business projects. Built on a command-and-control model, a traditional PMO emphasizes consistent process execution from the top down—and reams of documentation. The discipline of "project management for IT" matched up reasonably well with the waterfall approach to development, focusing on controlling a project's project scope, cost, and schedule—all three of which were predetermined at a project's start and served as the basis for performance reporting.

In essence, the PMO was established to answer questions centered on process: "Did we meet spec?," "Did we deliver on time?," and "Did we meet budget?" Missing was any metric regarding working software that actually met the end user's needs.

The risks and shortfalls of a traditional PMO in a waterfall setting are significant. Oftentimes, step-by-step plans don't play out as originally designed; yet the PMO continues to report against them. As a result, waterfall IT projects and their PMOs are often blindsided by problems that come to light very late in the project. Indeed, too often, business customers are disappointed with the traditional approach's results, where slow development times yield systems that technically meet spec but fail to deliver value for users.

The mismatch between traditional PMO techniques and an Agile project manifests itself from the very start. An Agile project isn't born with detailed specs or a comprehensive schedule for completion. Instead, the final product emerges through a collaborative and iterative process in which business users provide continuous feedback to developers. However, in order to satisfy the PMO's demands, organizations often perform impressive contortions in an attempt to fit Agile development's square peg into the PMO's round hole.

The question facing senior management is thus: How do we adapt the PMO to survive (and thrive) in an Agile development world? How can we maintain visibility into project delivery using metrics appropriate for Agile? How do we maintain a coherent set of processes while enabling innovation by the project teams? These are critical questions, because the ability to coordinate multiple workstreams running in parallel can often determine the success or failure of an IT effort.

What makes an AMO different?

The switch from a PMO to an AMO is much more than a superficial name change. An AMO is designed to operate as an effective way to monitor the progress of Agile projects. It measures success in terms identical to the Agile development teams' goal: working software that meets users' needs. Because of this, an AMO, while performing an oversight function similar to a PMO, operates in a fundamentally different manner.

Four key features distinguish an AMO from a traditional PMO:

- **Tracking**. An AMO tracks team productivity and product delivery by applying lean estimation techniques, emphasizing the delivery of working software.
- **Coordination**. An AMO fosters collaboration between teams, ensuring that efforts are coordinated and aligned.
- **Prioritization**. An AMO ensures that teams focus on the important things first—features that will drive working software that delivers business value. (These features may change over the course of the project, however!)
- **Governance**. An AMO provides a lightweight form of governance that focuses squarely on the project's strategic vision while allowing flexibility at the task level.

Shared language is essential

Numerous methodologies, each with its own vocabulary, have emerged to define a delivery process based on the Agile Manifesto. While the AMO approach does not prescribe any particular methodology, it is compatible with popular approaches, such as Scrum and the Scaled Agile Framework.

Table 1. A traditional PMO versus an AMO: Compare and contrast

Program management office (PMO)	Agile management office (AMO)
Tracking	
• Red/amber/green (RAG) reports • Gantt charts • Status of work compared to the plan	• Burndown charts • Story points per sprint/velocity measurement • Delivery of working software
An AMO tracks a project using tools and methods that emphasize visible progress through working software.	
Coordination	
• Matrixed hierarchy of managed projects • Centralized communication among teams • Consistency from the top down	• Flat structure with self-organizing teams • Direct cross-project collaboration • Continuous experimentation and innovation
An AMO aims to empower those closest to the work to collaborate and innovate to deliver the product.	
Prioritization	
• Detailed specifications/requirements • Value tracked in a detailed, long-term road map • Fixed schedule and prioritization	• Development of a high-level product backlog • Value delivery forecast in a conceptual road map • Frequent re-evaluation of business needs
An AMO focuses on defining high-level plans and commits to regular reprioritization at the end of each iteration.	
Governance	
• Documentation-driven • Approval required at each phase gate • Regular cadence-driven	• Value-driven • Infrequent intervention • Flexible and adaptable to meet team needs
AMO governance strives to be lightweight, relinquishing day-to-day control to focus on value prioritization.	

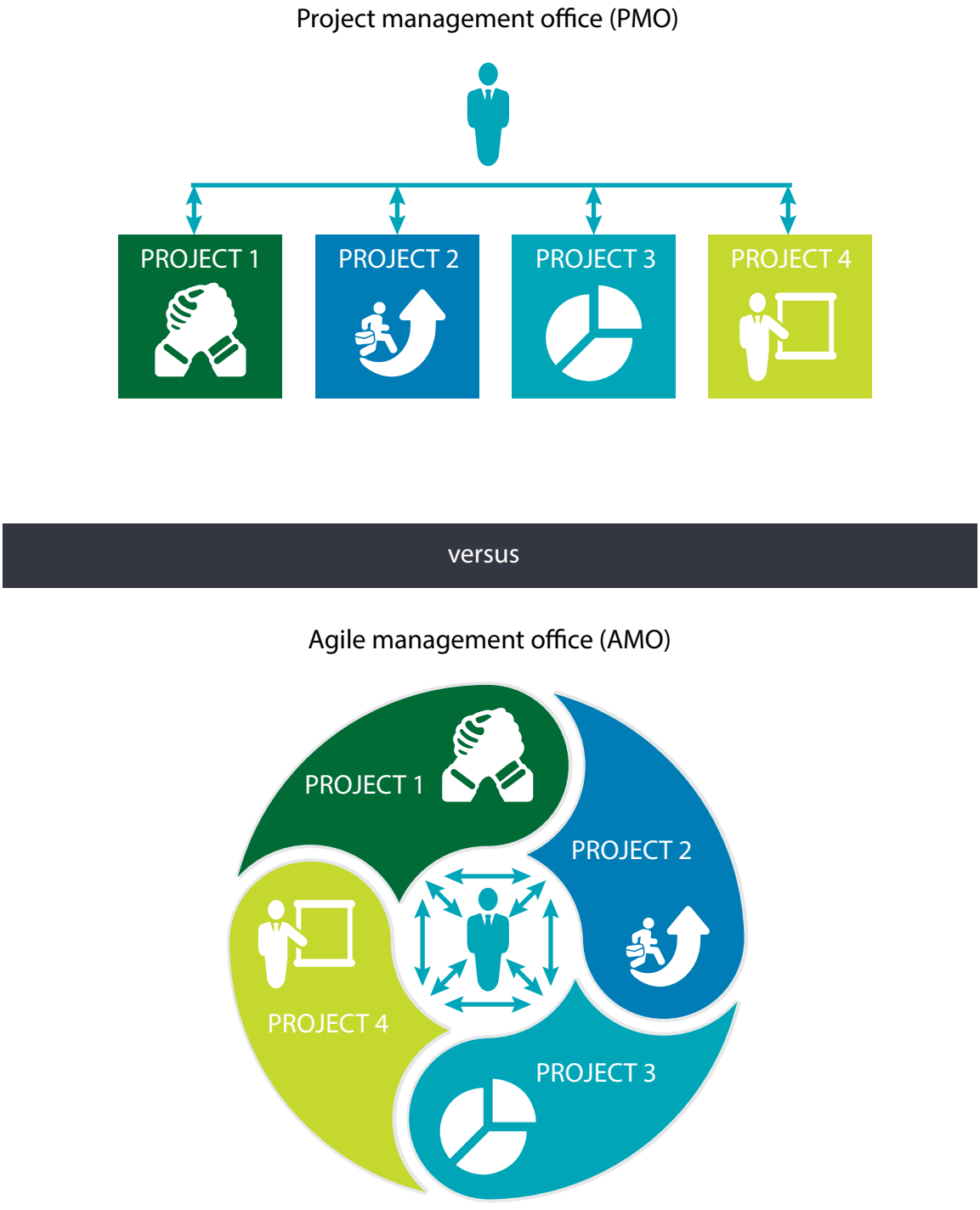
Source: Deloitte analysis.

The net result of these four characteristics is accountability for delivering the desired result.

TRACKING

An AMO can track delivery during an Agile project in two distinct ways. The first is by documenting the work delivered by each Agile team in terms of "story points" completed during each time-boxed iteration, each of which typically lasts two to four weeks. Story points reflect a relative estimate of task complexity, estimated directly by the development team. The number of story points each team produces per iteration, known as "velocity," can make it possible to more reliably forecast future iterations' output and to estimate the time re

Figure 1. An AMO facilitates coordination among Agile delivery teams



Source: Deloitte analysis.

quired to deliver functionality. For those unfamiliar with Agile, these terms may sound odd, but in practice, they can lead to more accurate planning estimates than predefined resource hours or task durations.

The other measure of progress is the demonstration of working software. Unlike a traditional waterfall approach, where working software is often seen for the first time months or years into a project, Agile emphasizes the rapid creation and frequent demonstration of working software to allow business leaders to react to the software early in the process.

COORDINATION

In projects involving multiple teams, it is critical that teams are aligned in their efforts. An AMO serves to shepherd the work planning process, facilitating coordination among Agile delivery teams.

In lieu of a command-and-control model, an AMO enables and empowers project teams to make decisions and select enabling tools and processes within AMO-designated guidelines. Instead of serving as the focal point for all communication, the AMO creates communication channels that enable and promote the free flow of information among project teams. Cross-project dependencies, process improvements, and the outcomes of "failed" experiments and retrospectives can thus quickly disseminate among teams. The AMO also provides program and project coaching to observe, capture, and socialize leading practices and guard against common pitfalls without prescribing a rigid delivery approach.

In line with Agile values, an AMO does not coordinate Agile project teams so much as it enables the teams to coordinate among themselves. This is a subtle but important difference, as it empowers the teams, enabling a thriving culture of innovation and experimentation, rather than putting a central management entity in charge. By empowering teams to drive the day-to-day activities, the AMO is better able to objectively monitor delivery from end to end and identify impediments and process bottlenecks, with a view to eliminating waste and delay. For instance, the AMO can champion productivity-enhancing initiatives or update project delivery guidelines to implement system-wide improvements.

By empowering teams to drive the day-to-day activities, the AMO is better able to objectively monitor delivery from end to end and identify impediments and process bottlenecks, with a view to eliminating waste and delay.

PRIORITIZATION

An AMO guides the prioritization of work in a manner consistent with Agile execution at scale, using a product backlog that includes all business, technology, or infrastructural capabilities to be developed. The business and IT work collaboratively to create, maintain, and prioritize the backlog, typically in a shared, integrated life cycle management tool.

Product planning provides a long-term vision at a conceptual level. A high-level road map depicts the sequence in which the strategic objectives should be developed and provides a high-level timeline for the business value the software delivers. Strategic objectives enter the backlog as "epics," a conceptual depiction of desired business functionality. At regular intervals, a combined panel of business and IT leaders review and prioritize epics in the pipeline, which are then allocated to one or more cross-functional teams for implementation. For example, organizational leadership may call for expanded cybersecurity coverage in light of recent cyberattacks on similar organizations. This work may entail significant refactoring across multiple IT systems, requiring resources to suspend new development work. Because Agile development teams plan frequently and with a shorter-term view (two to three months), these prioritized cybersecurity enhancements can be scheduled for implementation more rapidly, and with less disruption, than in the traditional model.

Delivery is synchronized among teams along a shared iterative cadence in which teams release and integrate work at a frequent and predictable pace. A direct line of sight can be established from the day-to-day work at the team level to the progress made towards meeting enterprise strategic objectives. For many organizations, the introduction of collaborative progress-tracking tools can yield unprecedented visibility into the results of their investment as well as enabling more engaging interactions with the business customer.

GOVERNANCE

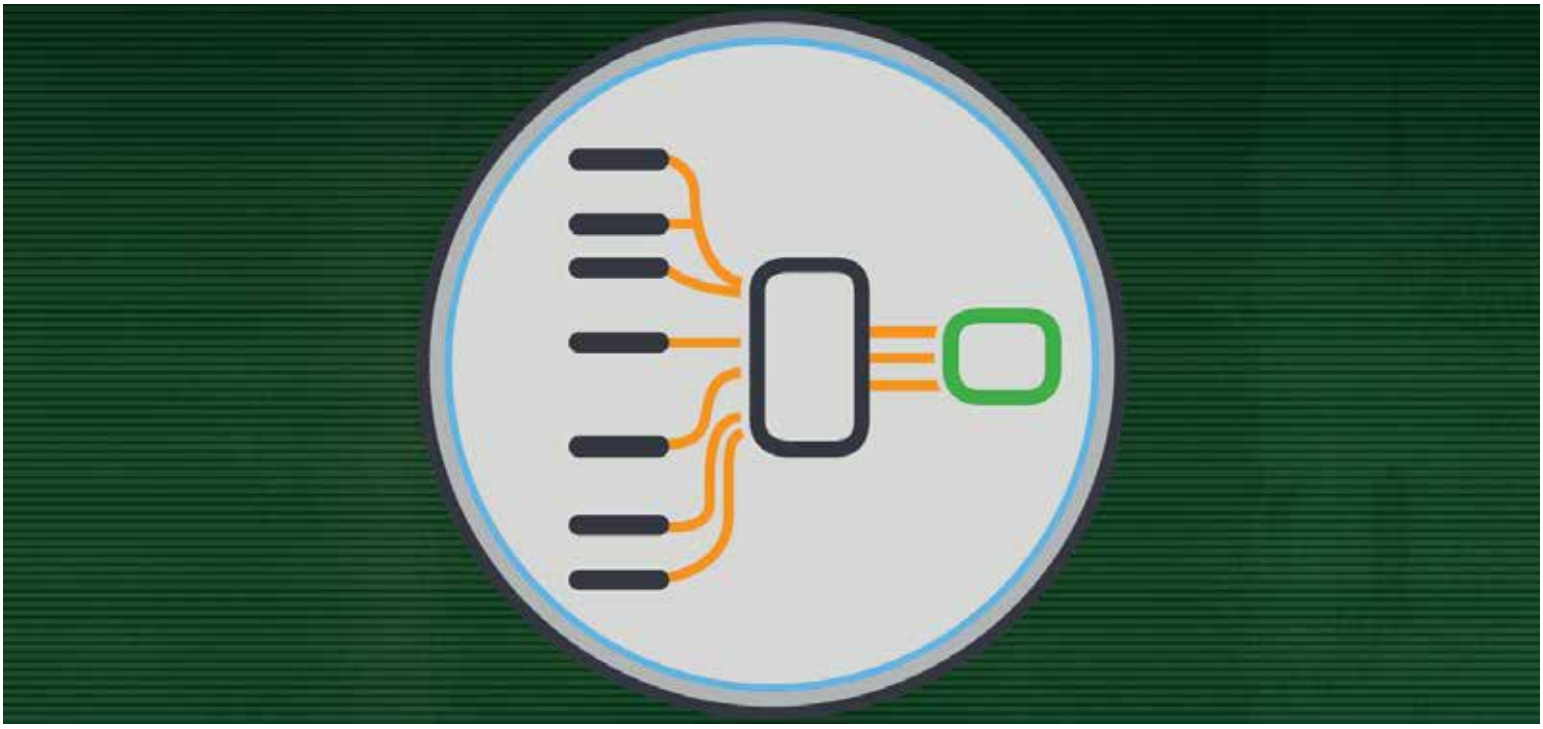
An AMO adopts a lean approach to governance, streamlining the delivery of planned work while ensuring alignment with business priorities. Regular product demonstrations at both the program and project levels take the place of traditional phase-gate governance reviews, freeing teams to rapidly release functionality without requiring formal approval. Risks, cross-project impacts, and dependencies are identified and mitigated in regular integrated planning sessions without formal governance board review. Instead of funding projects and programs designed to deliver one specific product or system, the AMO allocates funding to long-standing value delivery streams, a network of teams that shepherds a constant flow of strategic objectives from conception through business use. The AMO ensures that the flow of new requirements to the value streams aligns with the enterprise strategy, and monitors delivery throughput using highly visible Agile metrics and reports.

Is an AMO for me?

The decision of whether or not to establish an AMO will often depend on an organization's experience with Agile, the scale or complexity of the work being delivered, and leadership's willingness to help drive a sometimes difficult journey of culture change. Before committing to an AMO transformation, leadership should be willing to cede control of day-to-day team-level processes and refocus on more frequently identifying and prioritizing strategic objectives. At organizations that decide to move forward with an AMO, the potential benefits include greater visibility into the work being delivered and the ability to reprioritize delivery to meet business needs. The bottom line: An organization can use an AMO to effectively steer the ship while empowering and engaging teams, recognizing that the best ideas often come from those closest to the work.

Sam Martin is a senior consultant for Deloitte Consulting LLP's Federal Technology Consulting practice. He is based in Arlington, VA.

Kareem Abdelsadek is a manager in Deloitte Consulting LLP's Federal Technology Consulting practice. He is based in New York City.



Agile in Government

Agile by the numbers

A data analysis of Agile development in the US federal government

By: Peter Viechnicki and Mahesh Kelkar

THERE is little question that the Agile approach has been making inroads in government. Originally developed as an approach for building software in the private sector, government has also embraced Agile, at least in concept.

In 2012, the US Office of Management and Budget (OMB) issued guidance that "encourages agencies to shift away from the bloated, multi-year projects so common in the past" and to adopt a more modular approach.[1] The US federal government's "Digital Services Playbook" urges officials to "build the service using Agile and iterative practices."[2] Indeed, Agile Software Development Life Cycle (SDLC) textbooks have been seen on employees' desks at the highest levels of the US government.[3]

But how deep-seated and fundamental is this shift? Have Agile methods taken root within the federal IT community? How much IT procurement is Agile-based, and how much is still based on the traditional waterfall approach?

These are the questions that the Deloitte Center for Government Insights set out to investigate by analyzing hard numbers. We used a variety of federal data sources to dig deep into what is taking place with respect to Agile and federal IT. (See sidebar, "Data sources: The hard numbers that describe Agile's reach.")

Data sources: The hard numbers that describe Agile's reach

Rather than rely on anecdotes, our investigation of Agile combined data from four major federal sources.

OMB

To understand how much money the federal government is spending on its IT projects and how long they last, we analyzed data on 1,029 major IT projects compiled by the Office of Management and Budget on its ITDashboard.gov portal. These projects included only those that agencies listed in their Exhibit 300 submissions, approximately several hundred projects per year.

USASPENDING.GOV

To get a more complete view of run-of-the-mill federal IT purchases, we analyzed contract transactions from the Department of Treasury's USASpending.gov portal, restricting our analysis to transactions with product service code (PSC) 70.

OPM

To understand the makeup of the federal contracting workforce, we used the Office of Personnel Management's Fedscope data cube from March 2016.

FEDBIZOPPS

Finally, to understand how the language of federal IT procurement is changing, we pulled the full text of solicitations from the FedBizOpps portal, again restricting ourselves to procurements coded as PSC 70.

What we found is that the Agile movement coincides with a long-standing trend toward less expensive and more nimble IT projects, but most likely did not itself cause those trends.

What we found is that the Agile movement coincides with a long-standing trend toward less expensive and more nimble IT projects, but most likely did not itself cause those trends. For example, between 2004 and 2015, the duration of major federal IT projects went from an average of nine years to less than two years. But most of this shift took place prior to the embrace of Agile approaches. In fact, in 2011, fewer than 10 percent of major federal IT projects described themselves as "Agile" or "iterative." This number, however, has grown rapidly in the past few years: In 2017, fully 80 percent of major federal IT projects are now describing themselves as "Agile" or "iterative."[4]

One explanation for the increase in self-reported adopting of Agile and/or incremental approaches could be a desire to appear compliant with leadership rather than an actual underlying shift in IT procurement approach. While an openness to Agile, incremental IT procurement seems real, there continues to be a strong role for traditional waterfall practices as well. We conducted an analysis of more than 3,000 procurement documents, and found evidence that both waterfall and Agile approaches are currently reflected in federal IT projects.

In the following pages, we'll review the evidence in more detail and offer some insight about what the future may hold for the Agile movement and federal IT acquisitions.

Major federal software projects have been getting shorter in duration and smaller in cost for more than a decade

The Agile movement is all about flexibility through iterative approaches—which often entails breaking large projects into smaller and shorter chunks. But we find that the trend toward smaller, cheaper, and more iterative projects in federal IT development actually started years ago, as far back as 2004, before "Agile" was widely recognized as a distinct software development methodology in government. According to our analysis of data from the OMB, major software projects[5] have been getting both shorter and smaller in value every year (figures 1 and 2). In fact, the average cost for line-item projects in constant dollars[6] fell from $143.5 million in 2004 to only $1.72 million in 2015. Average project duration fell over the same period, from 108 months to 7.9 months.[7] Shorter IT development projects are widely perceived as less risky,[8] matching one of Agile's core tenets, "fail fast."

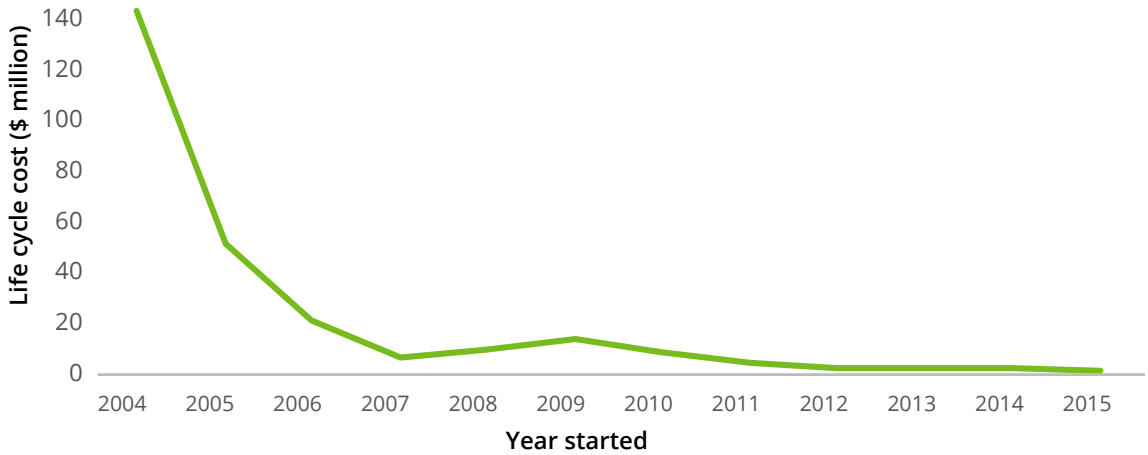
Improvements in IT procurement timelines and prices are not tied to Agile methods

To what extent are Agile methodologies responsible for the trends we see toward less expensive and shorter major projects? Self-reported use of Agile or iterative methods is definitely on the rise among major federal IT projects (figure 3). By 2017, some 80 percent of these major systems were reporting using Agile or iterative SDLC methods.

Since they self-report their choice of SDLC method, the owners of these projects may be responding to encouragement from OMB to use Agile methods. Or the reporting may reflect guidance and high-profile success stories from projects by the United States Digital Service (USDS) or GSA's 18F division.

But it's not possible to determine with confidence the extent to which Agile methods caused the reductions we're seeing in IT development times and costs for major IT projects.[9]

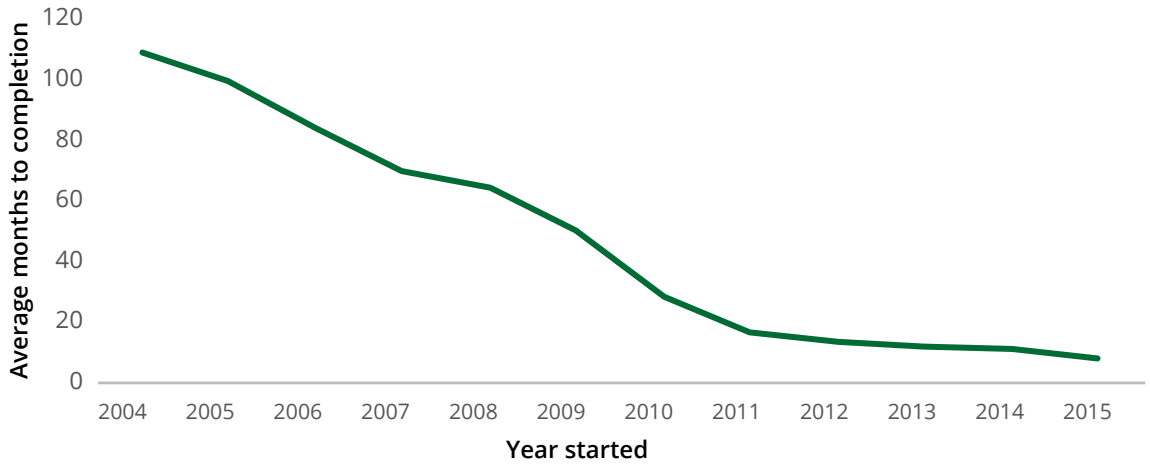
Figure 1. Major federal software projects are getting smaller



Note: The life cycle cost in 2015 was $4.09 million.

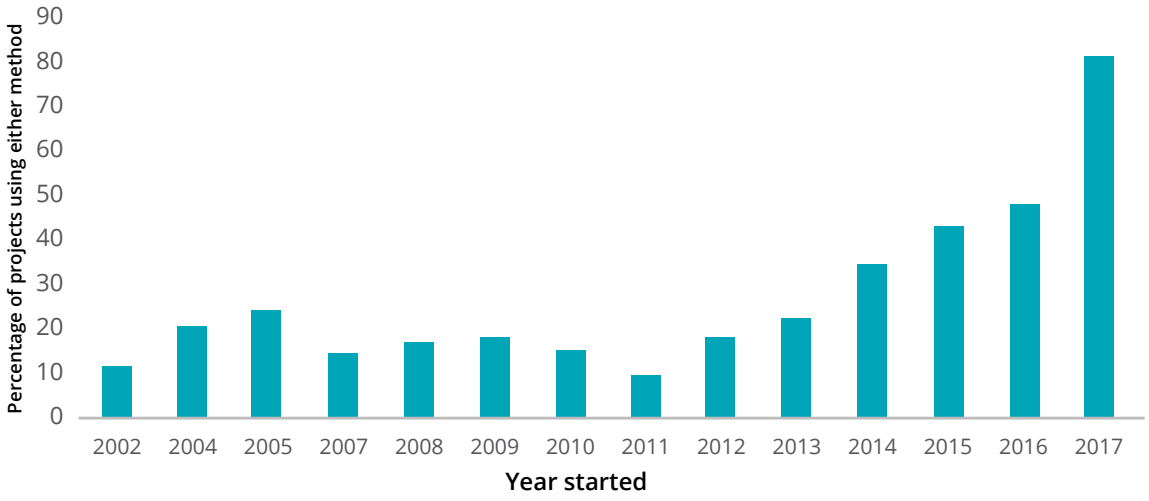
Source: Deloitte analysis of Office of Management and Budget's ITDashboard.gov.

Figure 2. Major federal software projects are also getting shorter



Source: Deloitte analysis of Office of Management and Budget's ITDashboard.gov.

Figure 3. Major federal IT projects are now overwhelmingly characterized as "Agile" or "iterative"



Source: Deloitte analysis of ITDashboard.gov projects.

Are contracting officers asking for Agile in their solicitations?

Are contracting officers (COs) requesting Agile methods in IT solicitations? Do the procurement solicitations that are now routinely self-described as "Agile" or "incremental" truly reflect a fundamentally different approach?

To find out, we analyzed statements of work from FedBizOpps, the primary source for federal contracts over $25,000. Our analysis shows that the majority of solicitations do not currently specify a particular SDLC, although some solicitations do specifically request Agile methods. We see no evidence of an increasing preference for Agile methods in solicitations at the expense of waterfall.

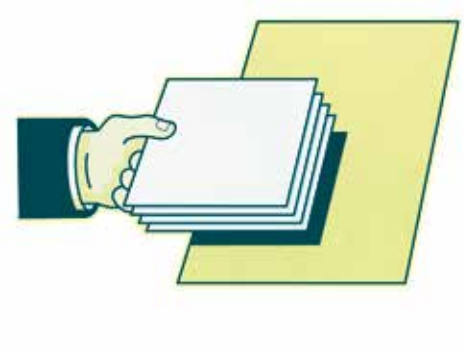
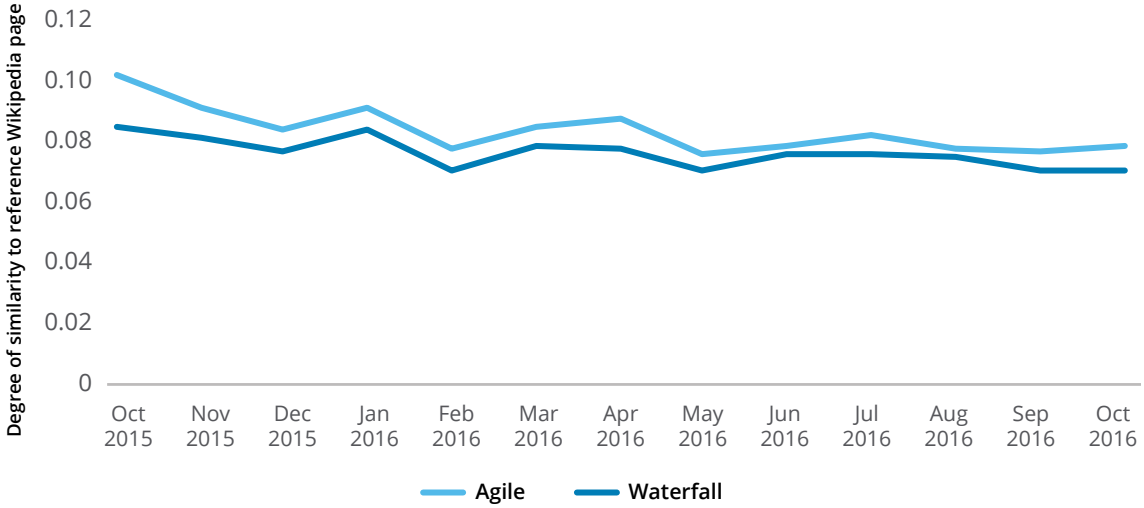


Figure 4. Agile and waterfall coexist in solicitations



Source: Deloitte analysis of solicitations from FedBizOpps.gov, October 2015–October 2016.

Using text analysis, a form of artificial intelligence, we studied more than 3,000 solicitations for software procurements between October 2015 and October 2016.[10] Figure 4 shows the similarity between the language in those solicitations and a separate set of reference documents about the Agile method. For comparison, we also measured the similarity of the solicitations' language to the language in reference documents about waterfall methods. The amount of "Agile language" included in solicitations is stable over time. Importantly, our analysis found no apparent decline in terminology related to waterfall methods.

Simply put, the majority of current solicitations appeared broad enough to cover both Agile and waterfall methods. Few and far between were "pure" Agile solicitations, though they did fit within the Federal Acquisition Regulations (FAR) and our text analysis method could detect them. Our finding is consistent with statements from proponents of Agile methods who point out that the flexibility of the FAR as they stand today allow for the use of Agile approaches.

The Agile movement came to government when IT procurement was already changing

It seems clear that Agile techniques have taken root in parts of the federal IT ecosystem, but they are by no means ubiquitous or exclusive. While we have seen a spike in self-reporting projects as "Agile" or "incremental," there is no evidence that COs are specifically requesting Agile methods in solicitations, and waterfall continues to play a significant role in IT procurements. How can we make sense of these seemingly contradictory indicators?

It seems clear that Agile techniques have taken root in parts of the federal IT ecosystem, but they are by no means ubiquitous or exclusive.

The most likely explanation is that high-profile delays, cost overruns, and cancellations[11] of major federal IT systems in the 2000s, coupled with the 2008 economic downturn, led to a general avoidance of lengthy projects with huge price tags. Somewhat later, the Agile methodology became popular inside the core of the federal government—the White House and agencies closely tied to its policy agenda, such as OMB, US Digital Service, and 18F. Both policy leaders and federal contracting officers appear to have become more open to new approaches

to software contracting. These separate trends coincided, and with encouragement from the White House, more and more owners of major government systems began labeling their methods as "Agile," even though they had been moving toward shorter, smaller projects since at least 2004. In addition, there has been a proliferation in the "flavors" of Agile, meaning that aspects of Agile methods can be (and are being) incorporated into traditional procurement vehicles.

Regardless of what label is applied, the fundamental trends toward smaller, shorter IT projects is undeniable. As the federal contracting workforce continues to refine more Agile approaches to IT procurement, we are likely to see, at the very least, the fundamental principles of Agile continue to play an important role in federal IT.

ENDNOTES

8. The OMB guidance is available at https://obamawhitehouse.achives.gov/blog/2012/06/14/greater-accountability-and-faster-delivery-through-modular-contracting, and the description is from the June 14, 2012 OMB blog anouncing it, available at https://obamawhitehouse.archives.gov/blog/2012/06/14/greater-accountability-and-faster-delivery-through-modular-contracting.
9. The US Digital Service, *Digital services playbook*, https://playbook.cio.gov/ , play no. 4, accessed April 6, 2017.
10. Robert Tross (senior manager, Deloitte Consulting LLP), interview with the authors, 2016.
11. Deloitte analysis of data from ITDashboard.gov, https://www.itdashboard.gov/drupal/data/datafeeds.
12. Data from OMB's ITDashboard.gov, which only tracks major IT projects—that is, those receiving a line item in the annual US federal budget. We restricted our analysis to projects whose status was "completed" and were coded as primarily software projects.
13. We use 1994 (the first year of the ITDashboard time series) as the base year and then depreciate subsequent years using the consumer price index.
14. Deloitte analysis of data from ITDashboard.gov, https://www.itdashboard.gov/drupal/data/datafeeds.
15. See, for example, Chris de Leon and Philip Petrina, *The road to Agile: PennDOT's transformation to iterative and Agile methods*, Pennsylvania Department of Transportation, 2016, http://Agilesummit.harrisburgu.edu/lib/pdf/2016-05-09-AgileSummit-RoadtoAgile-ChrisandPhilFinalVersion-NoNotes.pdf.
16. From the available data on ITDashboard.gov, we do not see clear evidence that Agile projects have significantly shorter durations and costs than waterfall or other SDLC projects. Nor did we find any data that correlates success rates with software methodology in federal IT.
17. Text analysis of documents attached to 3,063 solicitations from FedBizOpps.com for products coded "70" between October 2015 and October 2016. The lines in the graphic show the degree of similarity by month of solicitation to reference documents about Agile development and waterfall development (the Agile and waterfall pages on Wikipedia).
18. Government Accountability Office, *Implementation of reform legislation needed to improve acquisitions and operations*, November 4, 2015, http://www.gao.gov/assets/680/673508.pdf.

Peter Viechnicki, Deloitte Services LP, is a strategic analysis manager and data scientist with the Deloitte Center for Government Insights.

Mahesh Kelkar, Deloitte Services LP, is a research manager with the Deloitte Center for Government Insights.

ACKNOWLEDGEMENTS

A number of colleagues within Deloitte member firms generously contributed their time and insights to this report. In no particular order, the authors would like to thank **Howard Byrd** and **Eric Uhlir** for their contributions to this research. We would also like to extend a special thanks to **Gregory Marlow**, whose assistance was critical to the analysis of data on federal solicitations scraped from FedBizOpps. Thanks also to **Manu Choubey** from the data sciences team for helping us pull and analyze data from USASpending data portal. **William Eggers, John O'Leary,** and **Dave Noone** from the Deloitte Center for Government Insights provided valuable inputs and critical editorial help at important junctures.

CONTACTS

Deborah Sills
National managing principal, public sector
Deloitte Consulting LLP
dsills@deloitte.com
+1 303 298 6603

Warren Miller
Managing director
Deloitte Consulting LLP
wmiller@deloitte.com
+1 571 858 0612

William D. Eggers
Executive director, Deloitte Center for Government Insights
Deloitte Services LP
+1 571 882 6585
weggers@deloitte.com

John O'Leary
Deloitte Center for Government Insights
Deloitte Services LP
+1 617 437 3576
jpoleary@deloitte.com

Sign up for Deloitte Insights updates at www.deloitte.com/insights.

Follow @DeloitteInsight

Contributors

Editorial: Junko Kaji, Aditi Rao, Nikita Garia, Abrar Khan

Creative: Sonya Vasilieff, Molly Woodworth

Promotion: Haley Pearson

Artwork: Hoey Comics

About Deloitte Insights
Deloitte Insights publishes original articles, reports and periodicals that provide insights for businesses, the public sector and NGOs. Our goal is to draw upon research and experience from throughout our professional services organization, and that of coauthors in academia and business, to advance the conversation on a broad spectrum of topics of interest to executives and government leaders.

Deloitte Insights is an imprint of Deloitte Development LLC.