FEDERAL DEVOPS SUMMIT

JUNE 28, 2017 | MARRIOTT METRO CENTER | WASHINGTON, DC

On behalf of the Advanced Technology Academic Research Center, I am proud to announce the release of a White Paper documenting the MITRE-ATARC DevOps Collaboration Symposium held on June 28, 2017 in Washington, D.C. in conjunction with the ATARC Federal DevOps Summit.

I would like to take this opportunity to recognize the following session leads for their contributions:

MITRE Chairs: Michelle Casagni, Melissa Heeren

Challenge Area 1: Scaling Agile

Government Lead: Asghar Noor, U.S. Small Business Administration
Industry Lead: Rizwan Tanoli, Acuity
MITRE Lead: Michelle Casagni

Challenge Area 2: Rugged/Secure DevOps

Government Lead: Jennifer Hoover, TSA, U.S. Department of Homeland Security
Industry Lead: Chuck Svoboda, Red Hat
MITRE Lead: Michael Kristan

Challenge Area 3: Assessing DevOps

Industry Lead: Bob Dodson, Eliassen Group
MITRE Lead: Rick Cagle

Challenge Area 4: Right-Sizing Documentation for Developers and Decision-Makers

Government Lead: Kevin Long, U.S. General Accountability Office
MITRE Lead: Diane Hanf

Challenge Area 5: Acquisition Model for Agile

Government Lead: Thomas Thompson, U.S. Department of Homeland Security
Industry Lead: Anil Karmel, C2 Labs
MITRE Lead: Melissa Heeren

Below is a list of government, academic and industry members who participated in these dialogue sessions:

Challenge Area 1: Scaling Agile

John Asbra, Pivotal Software; Tina Chang, USDA; Pryalal Karmakar, DHA; Ken Laskey, MITRE; Jacques Malebranche, GSA; Nicholas Pesce, MITRE; Sahar Sadeghian, MITRE

Challenge Area 2: Rugged/Secure DevOps

Durga Anakala, HUD; David Axinn, Pivotal Software; R.E. Baum, GitLab; Sean Burgess, USDA; Peter Burkholder, GSA; Mark Galpin, JFrog; Chris Goehring, Pivotal Software; Kyle Mach, DHA; Delbert Miller, DoS; James Mysliwiec, USAID; James Paschal, DoD; Mark Roth, USAF

Challenge Area 3: Assessing DevOps

Gary Alderman, HUD; Kevin Bierschenk, IRS; Dallas Blair, HUD; Alex Bois, Eliassen Group; Julio Burgo, MITRE; Thomas Chester, SEC; Mary Curtis, EPA; Brandon Dube, TTB; John Griffith, MITRE; Thomas Hoffmann, IRS; Pamela Isom, USPTO; Deepak Kunal, USPTO; Daniel Ng, FDA; Atika Tabassum, Deloitte; Audrey Winston, MITRE; Annie Xiong, Deloitte

Challenge Area 4: Right-Sizing Documentation for Developers and Decision-Makers

Antoinette Bowser, DoS; Chris Campbell, GSA; Tina Chen, EPA; Greg Elin, GovReady; Tricia Iveson, 10Novate; Clark Misul, IRS; Dennis Nelson, DLA; Amanda Racer, Census; Deanna Stanley, MITRE; Mark Webber, MITRE

Challenge Area 5: Acquisition Model for Agile

Eric Blank, EPA; Curtis Jackson, EPA; Giancarlo Osorio, DHA; Scott Overend, Eliassen Group; Brad Wintermute, FDA

Thank you to everyone who contributed to the MITRE-ATARC DevOps Collaboration Symposium. Without your knowledge and insight, this White Paper would not be possible.

Sincerely,

Tom Suder
President, Advanced Technology Academic Research Center (ATARC)
Host organization of the ATARC Federal DevOps Summit

2017 Federal DevOps Summit Report

Michelle Casagni, Melissa Heeren, Rick Cagle, Diane Hanf, Michael Kristan, Justin F. Brunelle

The MITRE Corporation[1]

Tom Suder and Tim Harvey

The Advanced Technology Academic Research Center

[1] APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. CASE NUMBER 17-3231-3.

Executive Summary

The second ATARC (Advanced Technology Academic Research Center) Federal DevOps Summit was held on June 28, 2017 in Washington, D.C. During this summit, five MITRE-ATARC Collaboration sessions provided representatives of industry, academia, government, and MITRE the opportunity to discuss challenges the government faces using DevOps and Agile processes for software development and delivery. The goal of these sessions is to create an interactive forum for participants to exchange ideas on best practices, recommendations, success stories, barriers and requirements to advance the adoption of DevOps and Agile within the government.

Participants ranged from Director, CIO, and other executive levels from government and industry to practitioners from government, industry, and MITRE. Each collaboration session had a MITRE, government, and industry lead to drive the discussions with session participants towards addressing challenge areas in DevOps and Agile, as well as identifying courses of action to be taken to enable government and industry collaboration with academic institutions.

This white paper summarizes the discussions in the collaboration sessions and presents recommendations for government, academia, and industry while identifying intersecting points among challenge areas. The sessions identified key overarching themes which are summarized below.

Culture

Adoption of Agile and DevOps practices requires a shift in mindset and a change in processes. Several themes related to culture were discussed across the collaboration sessions:

- Leadership buy-in;
- Start small;
- Success is contagious; and
- Fear of failure.

Leadership buy-in is necessary for Agile to succeed in an organization. Communication must go up, down, and across to include all stakeholders, leadership, and development teams. Processes and procedures are going to change when adopting Agile and DevOps, and that change needs to be realized and accepted. Buy-in for Agile – scaled or not – needs to come from the top down with leadership in place to drive teams towards delivering value.

Starting small and having impact that shows value to users will be contagious. As an organization begins to adopt DevOps, start small and grow as successes are achieved and others begin to see the value in Agile development. A first step could be adopting Agile approaches to systems development life cycle (SDLC), upon which further cultural transformation could be based. Establish specific measures to benchmark the starting point, and later demonstrate what is working and where to focus specific efforts aimed at further growth and maturity of any DevOps transformation.

Fear of failure results in a slowdown of development and adoption of new processes. Managing Agile should be light-handed with the realization that failure will happen. The Agile culture supports failing early and often to allow time for correction and more likelihood of success. Instead of saying “fail fast”, the motto should be “learn fast”.

People and Processes

Agile and DevOps is less about technology and more about people and processes. Legacy regulation, business rules, procedures, and development techniques that drive current processes are a mismatch with Agile and DevOps principles. Re-examining these components of the organization is critical to any needed organizational change to support modern software systems environments.

It is important that all stakeholders understand that they need to embrace change and simplicity, garner and react to rapid feedback, and be product (outcome) focused. In an Agile and DevOps environment, the rate of change is increased so that discussions and decisions occur more frequently and – more importantly – less formally. Therefore, there is a need to empower teams up front to make smart choices.

It was further suggested through the collaboration sessions that Agile is not just for development, it can be used for cross-cutting practices to include organizational processes. Also, government ownership of the development process will ensure consistency across the program and teams.

Training

Training should not be viewed as only necessary for developers; stakeholders and program/project managers should also be trained on the concepts and unique aspects of Agile and DevOps to effectively execute, adopt, and manage them. Since the fundamentals of Agile and DevOps are not yet widely understood, education will go a long way to helping stakeholders understand what is important in the new deployment/development paradigm and help them better understand the changes and commitment needed to adopt these methods.

TABLE OF CONTENTS

1 Introduction

The second ATARC (Advanced Technology Academic Research Center) Federal DevOps Summit was held on June 28, 2017 in Washington, D.C. During this summit, five MITRE-ATARC Collaboration sessions provided representatives of industry, academia, government, and MITRE the opportunity to discuss challenges the government faces using DevOps and Agile processes for software development and delivery. The goal of these sessions is to create an interactive forum for participants to exchange ideas on best practices, recommendations, success stories, barriers and requirements to advance the adoption of DevOps and Agile within the government.

Participants ranged from Director, CIO, and other executive levels from government and industry to practitioners from government, industry, and MITRE. Each collaboration session had a MITRE, government and industry lead to drive the discussions with session participants towards addressing challenge areas in DevOps and Agile, as well as identifying courses of action to be taken to enable government and industry collaboration with academic institutions.

The MITRE Corporation is a not-for-profit company that operates multiple Federally Funded Research and Development Centers (FFRDCs)[2]. ATARC is a non-profit organization that leverages academia to bridge between government and corporate participation in technology[3]. MITRE works in partnership with ATARC to host these collaborative sessions as part of the Federal DevOps Summit.

This white paper is a summary of the results of the collaboration sessions and identifies suggestions and recommendations for government, industry, and academia while identifying cross-cutting issues among the challenge areas.

2 Collaboration Session Overview

Each of the five MITRE-ATARC collaboration sessions consisted of a focused and moderated discussion of current problems, gaps in work programs, potential solutions, and ways forward regarding a specific challenge area. The sessions for this summit addressed:

- Scaling Agile – when should you scale and what are the different methods of scaling Agile?
- Rugged/Secure DevOps - how do you continuously modernize applications in a DevOps environment with security baked in?
- Assessing DevOps – how well is your DevOps process working and how do you improve it?
- Right Sizing Documentation for Developers and Decision makers – Agile doesn't mean NO documentation; how to incrementally develop documents and plans.

---

[2] https://www.mitre.org/about/corporate-overview

[3] http://www.atarc.org/about/

- Acquisition Model for Agile - applying Agile development practices with tailored acquisition models.

This section outlines the goals, themes, and findings of each of the collaboration sessions.

2.1 Scaling Agile

This session focused on defining what it means to scale Agile and when is it appropriate to scale. It also discussed challenges, lessons learned and best practices for scaling Agile.

2.1.1 Session Goals

- What does scaling Agile mean?
- When should you scale?
- What are different models/methods of scaling Agile to include lessons learned and best practices?

2.1.2 Session Summary

The opening remarks to kick off the session outlined the goals and specified that the session was not focused on one method for scaling Agile. Several different methods were listed for the participants benefit, which included:

- Scaled Agile Framework (SAFe);
- Disciplined Agile Delivery (DAD);
- Large Scale Scrum (LeSS);
- Scrum of Scrums (SoS); and
- Enterprise Scrum.

The session started with a discussion on the challenges faced by several organizations integrating new development using Agile into legacy systems and legacy development environments. Some challenges included different tools, contracts, and software architectures. Lessons learned from another organization suggested that communication is key. There needs to be coordination between program management, development teams, and leadership to manage expectations and needs across all teams. There should be a shared vision with a group understanding of the goal(s) so that everyone is driving to the same outcome, no matter what development style is being used. It was indicated that everyone wants success, so there should be an agreement on what success looks like and how it can be measured.

The discussion on using different tools and technology stacks across teams raised another concern within the group. Some expressed that different technology stacks could create development silos while others questioned if this was the root of the problem for scaling Agile to multiple teams. The challenges faced by scaling Agile is not about the technology stack or tools, it is about the people and the processes. The recommendation of a shared vision and the fact that teams need to communicate and collaborate was raised again, pointing out that things are going to change and that change needs to be realized and accepted. Buy-in for Agile, scaled or not, needs to come from the top down with leadership in place to drive teams towards delivering value.

There were several points made about how to grow Agile organizationally. One member of the group looked up a definition for scaling Agile online finding that there are two modes: *vertical scaling* is about dealing with resistance up and down the chain and *horizontal scaling* is bringing it to different parts of an organization. It was agreed that scaling Agile is about bringing Agile principles into the whole organization and not just for development; processes can also be Agile in nature. This aligns with the view that Agile strategies should be tailored to address scaling challenges faced by development teams as well as adopting agility across the organization[4]. To scale, it was recommended to start small and be successful. Starting small and having impact that shows value to users will be contagious. It is also necessary for stakeholder mentality to consider value to the consumer, not just a focus on schedule, budget and code produced. Once there are enough "small successes", plan on how to grow, but don't scale just to scale.

2.1.3 Recommendations

When to Scale: Most organizations tend to scale Agile when the teams are getting big. It was recommended that Agile organizations have a shared vision and definition of success that should be understood and agreed upon by all teams. There should also be a standard to follow that includes measures for success that leads to a cadence where everyone is seeing value.

Lessons Learned: Leadership buy-in is necessary for Agile to succeed in an organization. As an organization begins to adopt Agile, start small and grow as successes are achieved and others begin to see the value in Agile development. Communication and training is also critical. Communication must go up, down, and across to include all stakeholders, leadership, and development teams. Training shouldn't be viewed as only necessary for developers; stakeholders and program/project managers should also be trained on the concepts and unique aspects of Agile to effectively execute adopt and manage Agile processes.

Governance: Managing Agile should be light handed with the realization that failure will happen. The Agile culture supports failing early and often to allow time for correction and more likelihood of success.

Methods for Scaling Agile: There are several different methods for scaling Agile. Choosing the method that works best for an organization may not mean using one specific method; some organizations adopt best practices from different frameworks to tailor an Agile process that meets their needs and culture.

2.2 Rugged/Secure DevOps

This session focused on answering questions surrounding the integration of security into a DevOps culture and environment.

---

[4] Ambler, S. & Lines, M. "Scaling Agile Software Development". https://disciplinedagileconsortium.org/Resources/Documents/ScalingAgileSoftwareDevelopment.pdf

2.2.1 Session Goals

- Identify ways to continuously modernize applications in a DevOps environment with security baked in
- Identify possibilities to break down monoliths and cut down batch sizes
- Identify when to get Information Assurance (IA) involved in sprints and how to move IA from waterfall to Agile
- Identify how to integrate open source products into government operational ecosystems
- Identify the role of Authority to Operate (ATO) for DevOps tools
- Identify the ability to significantly shorten the time to achieving ATOs

2.2.2 Session Summary

The session began with participants stating the challenges in their organization with respect to security and DevOps. There were several common themes that were discussed. In a fast paced – but cyber-centric – environment, the rules are constantly changing due to emerging threats. Thus, patches and Security Technical Implementation Guides (STIGs) are constantly changing. Some organizations have the additional burden of having to conform to compliance guidance of multiple organizations. Once an environment has been certified and accredited, it won't need re-accreditation unless an emergency has been identified or a major release/change occurs. It is unclear how many of these accredited environments are immutable in their architecture and implementation. Smaller releases were not considered to be material changes and thus not subject to recertification.

The participants expressed frustration with the existing development and operations path. There is a general fear of failure and thus a resulting slowdown in development to accommodate for additional processes. Several participants are working in an environment where the target production environment is a closed off environment due to sensitive processing so automated steps to employ Continuous Integration (CI) and Continuous Delivery (CD) across air gaps is currently impossible. Adding to this challenge is the claim that security processes and threats are not always transparent, many of the controls in the Risk Management Framework (RMF) have little to do with software development and cannot be automated, and there is a culture of being told "no", whether it be security teams, administrators, finance personnel, or others. There is a general misconception among the community that open source software is less secure than closed source and that automated scanning tools result in too many false positives, slowing the automation pipeline.

The discussion moved on to questions that were posed by members of the room. Many of these questions are based on the challenges previously mentioned. One individual asked if it was a good practice to involve developers in Change Configuration Boards (CCBs) to which the answer was "yes, if the process is Agile enough." Most of the questions revolved around the role of the security person such as "what is the right taxonomy to use when developing software that a security expert would resonate with?" and if there are tools and automation steps that can automatically be triggered to generate valid bodies of evidence that would be used in a security accreditation package. This

created a debate around whether or not the velocity of incremental releases creates too much verbosity and if manual inspections are better than automated since automated testing results in a sense of control. The discussion moved on to questions about refactoring, breaking up monoliths to more reusable microservices that can be individually certified, and how to scan these microservices using newer technologies (such as containers).

The group went on to discuss improvements in two areas: processes and people. The consensus in the room is that DevOps with an eye towards ruggedness and security is an evolution of the Agile development process with the need for security at all phases of the development cycle. Unlike Agile where breaking a build is forbidden, breaks and failures are tolerated but won't automatically be pushed to production and it is expected that some commits and check-ins will not make it through the complete CI/CD pipeline. On a security front, the output of the CI/CD pipeline can include artifacts and tests that will validate a continuous ATO accreditation so long as all the requirements are satisfied in the pipeline, otherwise it will not deploy to production. Rather than thinking of DevOps as a phase such as DevSecOps or SecDevOps, bringing it in everywhere results in the more ubiquitous, but clunky SecDevSecOpsSec title. This aligns with the culture of treating security as a functional bug in software. All this, of course, will require an analysis of time and money to implement.

From a people perspective, the goal of adopting DevOps is to empower developers up front to make a smart choice in the first place. Finding a change agent or a small team of excited people, representing a small vertical slice of the organization, is a good way to start small with implementing DevOps in a new organization. Some ideas that were floated around include enlightening the other roles in the organization through educational opportunities. One suggestion was building 5-minute lightning talks to highlight the perspective of a developer, operator, or security professional. Everyone should be a stakeholder, including customers and security professionals. This also extends to contractors augmenting the organic staff and mandating that they follow certain policies and processes to be open and collaborative and to define how they shall deliver products. These discussed items help push back against a culture of fear where instead of saying "fail fast" with respect to security, the motto is "learn fast".

2.2.3 Recommendations

- Testing and securing
    - Look at how finance industry is doing security
    - Test your application in simulated, but real-world environments such as cyber test ranges
    - Employ Continuous Monitoring in a production environment
    - Let security person decide what artifacts are needed and automate whenever possible
    - Determine what is necessary to be submitted for reaccreditation. "No need to submit user interface (UI) changes", for example
- Automate Configuration Change Board (CCB) process
    - Connect tracker software to CI pipeline to feed back into requirements boards

- Deploying to Software as a Service (SaaS) or Platform as a Service (PaaS) may reduce the number of required controls vs Infrastructure as a Service (IaaS)
- Use vendors to monitor upstream of open source community
- Create a developer golden image which is a development environment with pre-approved products and libraries installed to minimize assessment of development libraries

2.3 Assessing DevOps

The *Assessing DevOps* session proposed two fundamental questions. Firstly, "are you ready for DevOps?" for attendees who were contemplating, but had not yet undertaken, a transformation of their software development lifecycle. It also asked, for those who were further along in exploring DevOps, "how well is your DevOps process working?"

This group represented a great deal of diversity of familiarity and implementation experience with fundamentals and tenets of DevOps. Among those early in their considerations of adoption, questions of "what is the scope of DevOps?", "what is not suitable for DevOps?", and "what are others doing for their assessments?" were foremost on their minds. Beyond those who were still considering a DevOps assessment for their portfolio and programs, many were interested in discussing initial approaches, differing techniques and priorities, and best practices being employed by other adopters.

The group moved quickly from the early discussion of the opening questions, and settled on the narrower goal of addressing a framework for understanding how to begin transforming organizational structure and culture to prepare for adopting DevOps.

2.3.1 Session Goals

- Are you ready for DevOps?
- How well is your DevOps processing working?
- Where should you begin a DevOps transformation?

2.3.2 Session Summary

The session began with introductions, which included organizational and domain perspectives, and interests in the goals of the session, as well as a statement of how far along each representative and their organizations were in exploring DevOps. These early inputs included perspectives from those only just considering beginning their DevOps adoption, as well as from those already engaged in some aspect of DevOps transformation, such as deploying Infrastructure-as-Code (IaC), CI, or CD, and so forth. Many were interested in comparing techniques, learning from each other, and discussing emerging best practices or any approach to roadmaps for adoption. Several also expressed interest in understanding how to migrate and manage legacy applications still under active development, and how to integrate compliance requirements into DevOps tools and processes.

As the conversation opened, and the group began the work of the session to develop the topic, discussion moved toward where to begin when considering a DevOps assessment. Seasoned adopters offered the advice to start with understanding the state of the organization. In several cases, participants suggested identifying and cataloging any

existing work to automate parts of the software development lifecycle (SDLC), as kernels around which a CI, testing, or deployment practice could be structured. Others built on this, adding assessing other components of the organization critical to any needed organizational change, such as people, processes, and tools already in place. Many supported adopting Agile approaches to SDLC processes as a first step, upon which further cultural transformation could be based. Further suggestions included starting small (even as a potential "skunk-works" approach) and establishing specific measures to benchmark the starting point, and later demonstrate what is working, and where to focus specific efforts aimed at further growth and maturity of any DevOps transformation.

From this initial conversation, the group shaped and refined a more specific goal for the session focusing on taking a first step toward DevOps adoption, and defining how to assess their existing organizational structure and culture, and legacy investments.

With the ice broken, and the evolution of the session goal, most of the session consisted of a deep dive led by the session's industry Lead. The lead guided the group through a discussion of an example cultural assessment framework, with the acronym "STAR". The framework consists of the Stories found in organizational cultures, the collection of Taboos unique to each organization, the collection of shared Artifacts to be found in each culture, as well as any defining Rituals practiced. For each area, members of the session offered both positive and negative examples.

The session participants began with consideration of what Stories were shared in cultures to explain what happened in the past, regardless of whether the events were positive or negative. By sharing information in the form of these Stories, an organization reveals what members consider important. The audience shared several examples of the various anecdotes illustrating pain felt across the respective organizations, and what factors are driving them in considering DevOps to address concerns. For example, a story that resonated with much of the audience was how "… it took two months of work just for the stage gates…" or "… it takes nine months to start up a development environment", often referred to in DevOps as work bottlenecks, where a backlog can develop and become a productivity problem. Other negative stories referenced "… it worked in development (or test) …" or "… environments are always down!", referring to the difficulties with consistency and stability of SDLC environments. Still other examples were offered, referencing the frustrations with non-productive work "We have 20 documents we've produced that are nothing but 'shelf-ware' at best!" or with release planning "We can't stuff enough in a release!".

The audience was then challenged to reflect on what positive Stories might sound like, and what points they might address. Greater confidence in responsibilities, environments, features, and releases might sound like "I know who to go to for…", or "We know our environment is working!", or "sure, we can add that to tomorrow's release!", or users saying, "I knew that was coming."

Following the framework, the audience then discussed the cultural Taboos, those elements of an organization's culture that depict what members consider sacred, or forbidden. Some ideas brought forward were among those considered "sacred" or positive, such as "involving customers", having more "frequent releases", or "automating a position". However, the audience seemed to resonate more with the negative, or

"forbidden" aspect of the Taboo concept. Many were familiar with the risks of "changing a methodology", or with the worries about "missing a gate or a rollout." Others knew all too well the concerns with "silos", "delays in procuring and implementing" solutions, and "customers being excluded" from prioritizations.

Next, the audience weighed in on the cultural nature of Artifacts, or those symbols from the past to which members of an organization defer, and in which they may place value. The audience offered only a few positive examples, both tangible (passing tests, approvals as solutions pass through Stage Gates), and more abstract (decreasing Technical Debt), or even incidental (congratulatory emails). More examples came to mind from a more negative perspective, including "shelf-ware" documentation, long backlogs, sizable release notes, "Email Storms", and so on.

Lastly, the group turned to consider any long-standing routines or standardized practices referred to as Rituals. These reveal what is respected and treasured, or what is dreaded and despised. Positive examples discussed included "Thumb tests", having security baked-in, "Test Result Reviews", and the ability to "deploy anytime". Teams strive to "Fail Fast and Fail Early", and even failures under these conditions would prove to be worthy of formalized celebrations by the team. Specific reviews, such as "Metrics Reviews" for assessing measures of success, or "Operational Readiness Reviews (ORR)", and formalized approvals, such as "Security Validations" and "Milestone Approvals" would also be included in positive examples.

Negative examples included "Blame Storming", "Schedule beatings", "Pager-Duty" or "On-Call" status. Development may suffer from "extended code freeze", "ever-growing deployment windows", "re-baselining", and "emergency releases".

With a short time remaining in the session, the members agreed to turn the discussion over to a different topic, namely a bit of brainstorming on various Measures upon which recommendations could be made. Nominations were made for consideration of "Tech Debt", the overall "Quality", addressing "Defects", the "Percentage of Feature Complete" of the solution, and the "Velocity" or rate of value delivery. Success measures included some more traditional metrics, such as the Mean Time to Recover (MTTR) after an event or an outage, and Mean Time Between Failure", Also considered were newer measures, such as "the percentage of features accepted by the users".

2.3.3 Recommendations

- Conduct assessments both before you start a DevOps initiative and periodically afterwards
- Assessments need to be holistic, including review of the culture of the organization with respect to the appetite for change
- Identify appropriate metrics to monitor your DevOps performance; start with a limited set and inspect the list regularly to ensure the metrics are relevant and useful

2.4 Right Sizing Documentation for Developers and Decision Makers

This session focused on how to incrementally develop documents and track progress in an Agile environment.

2.4.1 Session Goals

The goals of the session were to share experiences and establish recommendations for the following statements:

- Agile doesn't mean "no documentation";
- Incrementally developing and updating documents and plans;
- Documenting ground truth about system – balance between ad-hoc developer discussions and final decisions;
- How do large priorities of management get translated and prioritized into work of teams? and
- Tracking progress using Agile Metrics and Earned Value Management(EVM).

2.4.2 Session Summary

- Agile doesn't mean "no documentation"

What Agile does mean, though, is that the pace of communications and documents needs to be aligned across the development/deployment cycle to generate documentation that is meaningful in an Agile, DevOps environment. Documentation is one way to unite the various stakeholders – such as security, development, operations, and test – that have a need to influence and shape a system. It should provide salient facts about the system/software under development and feed evaluations and key decisions that determine the next iteration of that system. Having no documentation would make it difficult to orchestrate these stakeholders or create the right artifacts that form the core of evidence that stakeholders need to do their jobs and/or create an audit trail. While development techniques have changed (taking advantage of technology to automate many repetitive tasks) the requirements for documentation have not been re-examined to support modern software systems environments.

The group tended to agree that legacy regulation, business rules, procedures, and development techniques that drive their current documentation needs are a mismatch with Agile and DevOps principles. There is a definite need for education across all stakeholders as to what Agile and DevOps culture is about to correct the misconception that Agile or DevOps means "no documentation". Because the fundamentals of Agile and DevOps are not yet widely understood, education will go a long way to helping stakeholders understand what is important in the new deployment/development paradigm and help them better understand that documentation is not discarded but realigned using lean principles. At the core of this understanding of the new process, the group felt it important that all stakeholders understand that they need to embrace change and simplicity, garner and react to rapid feedback, and be product (outcome) focused.

- Incrementally developing and updating documents and plans

Session stakeholders shared experiences that many documents that are generated across different stakeholder teams contain the same type of data. So rather than separate teams generate the same information for different documents many advocated for reuse from

other teams, teams should have a single source of information, and consider having a documentation pipeline that aligns with the progress of the systems as it goes through the pipeline. Most importantly the group converged on the idea of devising a documentation strategy, planning, and getting agreement from all the stakeholders on sufficient documentation for each stage of the DevOps pipeline and executing it. In the same manner that the system would be developed iteratively, documentation would be developed iteratively, reuse documentation from other teams, and optimize the generation of the documentation by not only generating what is newer than previously documented, but also by encouraging the reviewers of the document to review the document incrementally.

As the group explored documentation user needs, they felt that all DevOps stakeholders should agree to set a threshold timeline at which documents would be reviewed, especially for security documentation needs which seemed to be most voluminous. Some advocated for a documentation agreement one year ahead of development, or a cadence that matches the development cycle. In that agreement, DevOps stakeholder should set a scope for the deliverables and capture what the group determines is important to report. Looking at the documentation holistically, reporting requirements can then be allocated across teams, especially ensuring that the report focuses on proof of business value rather than descriptions of how the system was built.

- Documenting ground truth about the system – balance between ad-hoc developer discussions and final decisions

Regardless of development style, there are communications needs that capture team/project activities, that update stakeholders about ongoing services that they use, that provide guidance to teams/projects on needed compliance and to keep management informed. In an Agile and DevOps environment, the rate of change is increased so that discussions and decisions occur more frequently – and more importantly – less formally. Since the purest implementation of Agile and DevOps is complete (1) involvement of key stakeholders and (2) transparency on the state of the system, this does not pose any issues; ideally all stakeholders would be aware of ad hoc discussions and final decisions. But not all stakeholders have participated in this manner in the past.

Benefits of Behavior Driven Development

Teams already using TDD or ATDD may want to consider BDD for several reasons:

- BDD offers more precise guidance on organizing the conversation between developers, testers and domain experts
- notations originating in the BDD approach, in particular the given-when-then canvas, are closer to everyday language and have a shallower learning curve compared to those of tools such as Fit/FitNesse
- tools targeting a BDD approach generally afford the automatic generation of technical and end user documentation from BDD "specifications"

Agile Alliance

Particularly, security stakeholders have been less involved in system development, only becoming more involved when it is time to certify a system so the only way to truly understand the state or history of the evolution of the system is to review lots of documentation. While the need for evidence to support a certification decision is not undervalued, the group felt that much documentation of system state can be foregone if security is an early stakeholder that presents compliance needs early, shaping the development of the system and encouraging system changes early in the process when it is less expensive to do so. To support a more lightweight, high contact operational style that can support leaner, more meaningful documentation, the group talked about active, communications principles and supporting tools such as Behavior Driven Development which applies a discipline that enable automation of documentation.

- How do large priorities of management get translated and prioritized into work for teams?

The group discussed the many levels of management priorities — upper, team, mid-management — identifying that these priorities come at multiple times via multiple messengers. This has direct impact on an organization's configuration management ability. Understanding the priorities goes a long way to helping stakeholders understand what impacts management priorities will have on the deployment/development process. Capturing the priorities and determining their impacts to the product under development provides a foundation for being responsive to management priorities and being able to show how those priorities have been addressed. Keeping simplicity and transparency at the core, have these priorities at the ready to communicate back to management.

- Tracking progress using Agile Metrics and EVM

The team quickly agreed that metrics should articulate value – business value, product value, system value – and that all the key metrics that should be captured and reported is business value. The group articulated business value as showing that the product demonstrated is useful and meets expectations. Metrics need not be complicated. The EVM discussion was very short with the participants recommending "Don't do it"; EVM is being adapted to include Agile principles which are fundamentally different than the waterfall approach that it typically supports. Until an equivalent regimen is adopted, the group felt that EVM was not a good match to support their Agile activities.

2.4.3 Recommendations

- Educate all stakeholders so that they understand that fundamental to Agile and DevOps is more rapid system change, complete stakeholder involvement, and high process and system state transparency which – if adhered to – can mean that teams operate with less documentation
- Have early agreement on what documents are important and have meaningful document revisit rates
- Use Behavior Driven Development approaches and tools that enable teams to have automated documentation generation using a central repository accessible by all stakeholders

- Treat Management priorities like a requirement; manage using Agile configuration management techniques – determine impact keeping analysis simple and transparent to show responsiveness
- Metrics should articulate value; use demonstration of value rather than documentation as proof of delivery

2.5 Acquisition Model for Agile

This session focused on how to address the acquisition challenges in software intensive systems developed using the Agile software development paradigm.

2.5.1 Session Goals

- Understand how to apply Agile development practices within the Defense Acquisition System
- Identify tailored acquisition models with focused guidance for accelerated acquisition
- Understanding where and how to tailor programs

2.5.2 Session Summary

This session began with an overview of acquisition challenges by the government lead. These included the need to accept that traditional waterfall processes have failed to deliver systems on-time and on-budget, and adapt acquisition processes to support modern development paradigms, such as Agile and DevOps. Several challenge areas were discussed, including:

Challenges in Agile Contracting

- How to keep bias out of one vendor's work impacting the next acquisition?
- How can competitors work together?
- Requirements being written by one company can lead to less than optimal team writing requirements (because the best teams will want to be able to compete on the resultant work)

Agile challenges in government

- Color of Money constraints
    - E.g., moving to cloud services when money has been allocated for capital expenditures
- Fiscal year boundaries
- Little motivation for innovation
    - E.g., programs being reviewed negatively for not spending their budgets as opposed to being rewarded for saving money

One major issue identified was that many program management and acquisition professionals are not deeply conversant in information technology. The participants described the need for IT expertise to "buy the right stuff." That includes getting the right people with the right skills.

Another topic of discussion was when, in an Agile program, it is appropriate to use a Statement of Objectives (SOO) versus a Statement of Work (SOW). For an iterative Agile development effort, a well-written SOO may be preferred.

There was a great deal of discussion about the actual source selection process. For example, when considering past performance of a vendor who will be leveraging subcontractors, should the subcontractors' past performance be considered as well? The team determined that it depends; the subcontractors' past performance on similar work should be considered, but not on work that was very different from the current source selection. The issue of cross functional teams from multiple vendors and all associated issues need to be addressed in the contract to prevent problems during execution (e.g., Scrum Master from one company, team member from another: How can Scrum Master avoid "directing work?").

The team brought up the need for Product Owners who are trained and passionate to help avoid scope creep. The wrong product owner may not understand the technology or may not understand user needs. The session participants suggested that the contract have the vendor pay for product owner training which would lead to team building and solid expectation setting.

Ultimately, the idea of government ownership of the system and processes emerged as a major theme. This ownership requires appropriate training, support from leadership, and a deep understanding of and commitment to Agile and its processes.

2.5.3 Recommendations

- Provide basic Agile training to all stakeholders
- Send Acquisition professionals to technical conferences (recommendation to have 1-2 sessions at upcoming ATARC conferences)
- Work to find ways to address the "color of money" issue so that programs are not penalized if money is spent in another category if that is the best use of funds (e.g., the transition from capital outlay to services when transitioning from data centers to the cloud)
- Develop a template for IT Acquisition professionals that includes a common language for Agile, defines roles and responsibilities, identifies key personnel, outlines an appropriate reporting structure for Agile, and provides sample contract language for Agile acquisition

3 Conclusion & Summit Recommendations

The June 2017 Federal DevOps Summit highlighted several challenges facing the Federal Government's adoption of DevOps and Agile practices. A common set of themes emerged across the individual sessions: culture, people and processes, and training. Success stories are coming from government adoption efforts and with continued collaboration and sharing best practices and recommendations for mitigation of these challenges will evolve.

The following are the key overarching themes from the collaboration sessions.

Culture

Adoption of Agile and DevOps practices requires a shift in mindset and a change in processes. Several themes related to culture were discussed across the collaboration sessions:

- Leadership buy-in;
- Start small;
- Success is contagious; and
- Fear of failure.

Leadership buy-in is necessary for Agile to succeed in an organization. Communication must go up, down, and across to include all stakeholders, leadership, and development teams. Processes and procedures are going to change and that change needs to be realized and accepted. Buy-in for Agile – scaled or not – needs to come from the top down with leadership in place to drive teams towards delivering value.

Starting small and having impact that shows value to users will be contagious. As an organization begins to adopt DevOps, start small and grow as successes are achieved and others begin to see the value in Agile development. A first step could be adopting Agile approaches to SDLC, upon which further cultural transformation could be based. Establish specific measures to benchmark the starting point, and later demonstrate what is working, and where to focus specific efforts aimed at further growth and maturity of any DevOps transformation.

Fear of failure results in a slowdown of development and adoption of new processes. Managing Agile should be light handed with the realization that failure will happen. The Agile culture supports failing early and often to allow time for correction and more likelihood of success. Instead of saying “fail fast”, the motto should be “learn fast”.

People and Processes

Agile and DevOps is less about technology and more about people and processes. Legacy regulation, business rules, procedures, and development techniques that drive current processes are a mismatch with Agile and DevOps principles. Re-examining these components of the organization is critical to any needed organizational change to support modern software systems environments.

It is important that all stakeholders understand that they need to embrace change and simplicity, garner and react to rapid feedback, and be product (outcome) focused. In an Agile and DevOps environment, the rate of change is increased so that discussions and decisions occur more frequently and – more importantly – less formally. Therefore, there is a need to empower teams up front to make smart choices.

It was further suggested through the collaboration sessions that Agile is not just for development, it can be used for cross-cutting practices to include organizational processes. Also, government ownership of the development process will ensure consistency across the program and teams.

Training

Training should not be viewed as only necessary for developers; stakeholders and program/project managers should also be trained on the concepts and unique aspects of Agile and DevOps to effectively execute, adopt, and manage them. Since the fundamentals of Agile and DevOps are not yet widely understood, education will go a long way to helping stakeholders understand what is important in the new deployment/development paradigm and help them better understand the changes and commitment needed to adopt these methods.

4 Acknowledgements

The authors of this paper would like to thank The Advanced Technology Academic Research Center and The MITRE Corporation for their support and organization of the summit. The authors would also like to thank the session leads and participants that helped make the collaborations and discussions possible. A full participant list is maintained and published by ATARC on the FedSummits web site[5].

[5] https://www.fedsummits.com/devops/june-2017/