FEDERAL DEVOPS SUMMIT

MARCH 1, 2018 | 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 March 1, 2018 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: DevOps Implementation with Cloud

Government Chair: Surendra Babu, U.S. Department of Education
Academic Chair: Harvey Hyman, St. Leo University
Industry Chair: Bob Dorch, Dynatrace
MITRE Chair: Rick Cagle

Challenge Area 2: Integration of Security into the DevOps Culture

Government Chair: John Jediny, GSA
Industry Chair: Nick Markowski, Onyx Point
MITRE Chair: Michael Kristan

Challenge Area 3: DevOps in Health

Industry Chair: Chris PreDieu, IBM
MITRE Chair: Richard Eng

Challenge Area 4: DevOps and Test

Industry Chair: Tim Reaves, Eliassen Group
MITRE Chair: Jennifer Flam
MITRE Chair: Seth Goldrich

Challenge Area 5: DevOps Culture

Government Chair: Tobi Edler, GSA
Industry Chair: Eric Chen, CGI Federal
Industry Chair: Lisa Zellers, CGI Federal
MITRE Chair: Diane Hanf

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

Challenge Area 1: DevOps Implementation with Cloud

Majed Ayoub, DHS; David Charbonneau, EPA; Jeremy Cohn, SSA; Ronald Davis, NIH; Tony Frangieh, DoS; Angel Gonzalez, LOC; John Henderson, MITRE; Monica Hunt, U.S. Navy; Madhav Komaragiri, HRSA; Cynthia Larkins, USDA; Doug Meyer, HRSA; Naga Molleti, HHS; Sebastian Murry, U.S. Navy; Kyalo Musau, VA; Bruce Nguyen, JFrog; Chase Noel, U.S. Navy; Hemal Patel, DoS; Hoa Pham, LOC; Marty Russell, DHS; Maria Smyslova, VA; William Stevenson, FDA; Puru Subedi, DOT

Challenge Area 2: Integration of Security into the DevOps Culture

Tom Alal, SSA; Jim Bielski, MITRE; Joshua Camire, USDA; Roger Castillo, DOJ; Rashid Chowdhury, SSA; Denis Danilin, Treasury; Larry Eichenbaum, Chef Software; Kelly Enochson, MITRE; Mark Galpin, JFrog; Feda Handac, CNCS; Steven Harkness, SSA; Russ Holmes, Onyx Point; Mark Johnson, NIH; Alan Jones, DHS; Durga Kodali, HHS; Rupesh Kumar, IRS; Shouming Li, HHS; Neeraja Mareddy, USDA; Alexey Massalskiy, LOC; Sameka McNiel, NOAA; William Rankin, Census; Thomas Reaves, EPA; Stephen Reilly, HHS; David Shepard, CMU SEI; Parmvir Singh, ARL; Susan Sons, Indiana University; Joe Stewart, HHS; Alexey Svotnev, Peace Corps; Kevin Wheatley, DOJ; Siobhan Williams, Treasury

Challenge Area 3: DevOps in Health

Dan Bailey, Arlington County Government; Laura Fleming, NIH; Rachel Gardell, Peace Corps; John Griffith, MITRE; Tom Harrell, NIH; Jacques Malebranche, GSA; Mojgan Pedoeim, IBM

Challenge Area 4: DevOps and Test

Clis Acosta, DHS; Charlton Apgar, MITRE; Sara Bauer, BLS; Greg Encinias, Arlington County Government; Stephen Esmacher, SSA; Aaron Kemmer, MITRE; Jonathan Lewis, USAID; Margo Lord, MITRE; Simmons Lough, USPTO; Frank John McQuaid, Census; Sompop Noiwan, Red Hat; Mark Ottaviani, SSA; Amy Reedy, TSA; Lihua Shen, HHS; Carol Taylor, Treasury; Peter Tung, Census; Manohar Yerreddu, HHS

Challenge Area 5: DevOps Culture

Yonas Berhe, IRS; John Broderick, BLM; Hari Chinthalapale, Census; Ivor D'Souza, NIH; Paul Davis, CNCS; Gianna Dusch, Census; Charles Fabry, USDA; Michaela Hilliard, SSA; Don Johnson, OSD; Jerry Kurtze, JFrog; Kevin Lawson, TSA; Wei Ma, NIH; Omar McNeil, TSA; Melissa Merens, CNCS; Scott Overend, Eliassen Group; Amanda Racer, Census; Renee Rawls, DHS; Sahar Sadeghian, MITRE; Steve Smith, SSA; Amy Smith, CNCS; Deanna Stanley, MITRE; Mark Webber, MITRE; Audrey Winston, MITRE

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

FEDERAL SUMMITS

MARCH 2018
FEDERAL DEVOPS SUMMIT REPORT*

July 5, 2018

Michelle Casagni, Melissa Heeren, Rick Cagle, Richard Eng, Jennifer Flamm, Seth Goldrich, Diane Hanf, Michael Kristan, Justin F. Brunelle
The MITRE Corporation

Tim Harvey and Tom Suder
The Advanced Technology Academic Research Center

Contents

1 ABSTRACT

The third ATARC (Advanced Technology Academic Research Center) Federal DevOps Summit was held on March 1, 2018 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, CTO, 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.

Challenge Areas

This year's collaboration sessions revealed that there is overall acceptance of the need to move to a DevOps framework and embrace automation. Some key takeaways include the following summarized topics in this section.

Across the collaboration sessions there was general agreement that there is a need for automation of development, test, deployment and operational processes. This is a paradigm shift from previous summits [5, 4].

There is agreement that the transition to a DevOps framework would shorten delivery times, increase security (due to frequent, even daily scanning and code review), and reduce life-cycle cost of ownership. Even so, many government agencies are overwhelmed by the impediments to automation. Some of these impediments are listed below:

- Perception that a domain is too specialized or safety critical to risk DevOps processes;
- Implementing DevOps processes for legacy systems is an intimidating challenge;

- Replacing "one and done" Authority to Operate (ATO) with incremental, ongoing security implementation and review (with all the cultural impacts that implies);
- Traditional stove-piped waterfall organizations are not set up for the rapid cadence of Agile and DevOps (e.g., contractual requirements such as Test Readiness Reviews (TRR), and Factory and Site Acceptance Tests);
- Recruiting, training, and retaining DevOps and Agile expertise is a challenge in government IT due to government salary caps and limited career opportunities;
- The government budgeting process and organizational dynamics work against adoption of new processes like DevOps and Agile; and
- Manual processes are still deeply embedded in government IT projects.

Recommended Solutions

Despite the challenges, several recommended best practices are emerging for government DevOps and Agile implementations. The recommendations from this summit's sessions are listed below:

- Start small – move one application into a DevOps environment[1] to learn and demonstrate success and build on these early successes to build confidence and gain upper management support;
- Embrace risk – learn from mistakes and make incremental improvements;
- Break down silos among traditionally stove-piped organizations – pull individuals from different organizations to build cross-functional teams;
- Provide training on Agile and DevOps philosophies and methodologies to all practitioners, not just to those in specific roles; and
- Encourage cultural change with active leadership engagement, communities of practice, and continuing to make changes to grow and sustain the DevOps culture.

[1] Terminology in this domain is evolving; for the purposes of this paper, *DevOps environment* refers to the cultural, organizational, and technical aspects of DevOps.

2 INTRODUCTION

The third ATARC (Advanced Technology Academic Research Center) Federal DevOps Summit was held on March 1, 2018 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, CTO, 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)[11][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.

3 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:

- DevOps Implementation with Cloud – How do cloud operations, environments, and service models impact decisions for provisioning DevOps toolchains?

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

- Security and DevOps – How and when do government organizations assess and security-harden applications in a DevOps environment?
- DevOps in Healthcare Information Technology (IT) – Does DevOps live up to its hype when it comes to Healthcare IT?
- DevOps and Testing – Given the unique mission capabilities supported by the Federal information technology landscape, what test strategies and test tactics may enable the DevOps pipeline?
- DevOps Culture – DevOps requires operations, development and security/certification culture changes; how big is the cultural divide and how do government organizations successfully span it?

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

3.1 DevOps Implementation with Cloud

The *DevOps Implementation with Cloud* session considered what a cloud environment brings to the adoption of DevOps. It started with considering whether to combine the two practices if the organization is considering DevOps and how to acquire cloud provisioning and DevOps toolchains. The group also discussed obstacles or barriers to success of this approach, which cloud services models to consider, and advantages and risks to each. Attendees ranged in perspective from "considering either or both cloud and DevOps, but have not experimented yet" to "have tried one, but not the other" to "am sharing my experience with the combination – both good and bad."

3.1.1 Session Goals

This session had the following discussion topic goals:

- Cost and contracting – moving from capital expenditure to operational expenditure;
- To cloud or not to cloud;
- Choosing the right cloud service model (e.g., Infrastructure as a Service (IaaS) vs Software as a Service (SaaS) vs Platform as a Service (PaaS); DevOps as a Service);
- Understanding multi-cloud environments and integrating cloud management platforms; and

- Fully understanding ramifications of choosing vendor native verse agnostic.

3.1.2 Challenges

The session began with brief introductions around the room to share context about each attendees' organizational and domain perspectives on both DevOps and cloud usage within their respective organizations. To structure the conversation, the session adopted the same framework used to describe the challenges to prospective attendees. The floor was opened briefly to allow additional topics to be proposed and considered by the panel and the group. As a result, three additional topics were added to the challenges framework and session goals for discussion: Infrastructure-as-Code (or "IaC"); cloud hybrid and cross-domain structures; and roles and toolchains.

3.1.3 Session Summary

The session kicked off with a discussion of issues surrounding what impacts a cloud-based approach has on costs for developing DevOps and challenges in developing contract artifacts consistent with traditional acquisition approaches. Central to the organizational change needed is a philosophical shift from traditional *CAPEX* to *OPEX* models for acquisition. The term CAPEX (or "capital expenditures") models refers to the purchasing, sustaining, depreciating, and scheduling replacement or retirement of "things" – in this case, computer hardware. Conversely, a use of OPEX (or "operational expenditures") models refers to the rates of services and "utilities" acquired on an "as-needed" basis, and paid for on a schedule – such as a monthly bill for electricity.

DevOps enables a paradigm shift from traditional waterfall methodology to an agile incremental continuous integration and continuous delivery (CI/CD) model. There was broad agreement that a cloud environment offers substantial advantages to DevOps implementation in terms of established cloud characteristics of "on-demand self-service," "autonomic scaling in response to surge needs," and the repeatability of IaC. However, attendees widely acknowledged that traditional methods of acquisition would not continue to suffice in the context of these same characteristics. Cultural embrace of DevOps and cloud concepts, coupled with practical organizational transformation, would have to precede any evolution of acquisition approaches.

The discussion of acquisition of cloud shifted to acquisition of the DevOps toolchain. Several representatives of organizations with more DevOps experience related details on tools incorporated into their DevOps software development lifecycle processes; primarily

focused on "Orchestration" tools, citing examples such as Cloudbees[4] or Jenkins[5] for CI, Artifactory[6] for artifact management, and Cloud Formation[7] for automating cloud infrastructure provisioning. The question of "How much to buy?" led back to an oft-referenced academic observation that "there is no prescriptive way to incorporate DevOps," but several stories of why an agency experimented with DevOps were shared, along with which DevOps characteristics the agency focused on first, and in some cases, how successful or not their results had been. Some time was devoted to a related question of what regulations agencies are subject to when considering DevOps or cloud provisioning.

Regulations and mandates bridged naturally into the next topic planned: whether or not to go to a cloud environment at all. While it was agreed that "doing DevOps" did not require doing anything specifically with a cloud environment, cloud characteristics (as discussed previously) offer attractive opportunities for automation and environmental consistency. The primary driver identified for DevOps was for increased delivery tempo, and when combined with a primary driver for cloud provisioning being presumptive cost savings, the powerful combination of "faster/cheaper" emerged from the discussion. Among the concerns brought up during the conversation, again, the group focused on incompatibilities with current acquisition approaches. An example briefly mentioned was the concern of loss of control and ownership of assets (e.g., "If my S3-bucket goes away, I lose data!") [3, 2].

Following another interesting path for a DevOps and Cloud discussion, the group took up the topic of choosing the right cloud service model to best support a DevOps approach. Discussion began by framing the NIST defined simple cloud service models of IaaS, PaaS, and SaaS [10]. Attendees shared perspectives from their experiments and existing efforts, including drivers for choosing one cloud services model over another. Some examples of challenges were shared and led to discussion across experience and implementation levels of the various attendees.

Cases in which cloud migration represented prohibitive expenses (due to anything from large compute needs to large migrations of data and high volume or frequent throughput during DevOps cycles) were discussed in terms of the pros and cons of the sliding scales of control versus responsibilities for elements of the computing stack across the models. Experience shows that compute, storage, and network resources constitute major cost factors in the cloud [1]. Some suggested that a dedicated connection may provide for greater security and predictable network utilization cost. Others introduced scenarios deemed too complex,

[4] https://www.cloudbees.com/
[5] https://jenkins.io/
[6] https://jfrog.com/artifactory/
[7] https://aws.amazon.com/cloudformation/

such as systems with dependencies on legacy capabilities requiring a coordinated migration to both DevOps and cloud provisioning for multiple sub-systems. Cultural, organizational, and personnel barriers to success were also raised, such as when the primary challenge for an organization turned out to be a need for skills and experience not available in-house or where the organizational culture was observed to be so frozen and stagnant as to make such a transformation more expensive than the projected technical and operational return-on-investment could address.

Scaling needs arose as a concern, and was discussed next. The impact of the choice of cloud services model on the mechanisms identified for easy scaling, such as the combination of IaaS and IaC was discussed in considerable detail. Contrasting points of view for how to either implement DevOps in PaaS or SaaS were also given, along with the raised concern of risk of "lock-in" as one made use of cloud-native PaaS and SaaS instead of the more generic IaaS approach. This led again to the topic of outsourcing DevOps CI, testing, delivery, and deployment entirely to either a Cloud Services Provider (CSP) or a third-party consumer of CSP capabilities. Unfortunately, the conversation wrapped up for the segment with the group discussing the challenges of "outsourcing" of DevOps itself (the concept of "DevOps-as-a-Service") so the additional proposed topics of understanding multi-cloud environments and integrating cloud management platforms for DevOps and roles and toolchains were not discussed. The group would recommend these be considered for topics for a future session.

3.1.4 Recommendations

The participants in the *DevOps Implementation with Cloud* collaboration session identified the following important findings and recommendations:

- While DevOps does not require cloud implementation, cloud characteristics such as "on-demand self-service," "autonomic scaling," and IaC offer rich potential to empower DevOps adoption, facilitating from the necessary cultural shift to provisioning, tooling, management, and DevOps growth;
- As organizations budget and plan for a cloud investment in their DevOps provisioning, they should make use of the various estimating tools CSPs offer for understanding costs in terms of up-time for the resource and for both the volume and frequency of any data moving into or out of the cloud environment;
- Start small – move one application into a DevOps-cloud environment to learn and to demonstrate success to the enterprise;

- Consider outsourcing DevOps to a CSP, but be aware that a dedicated connection may be required to control network cost in managing high volume and throughput; and
- Consider employing a cloud agnostic and universal DevOps toolchain that leverages infrastructure as code and binary repository management to enable seamless CI/CD pipeline to build highly resilient mission critical production operation.

3.2 Integration of Security into the DevOps Culture

The *Integration of Security into the DevOps Culture* session focused on methods by which teams utilizing (or aiming to utilize) DevOps practices can integrate security practices into their DevOps Culture[8].

3.2.1 Session Goals

This session had the following discussion goals:

- How do government organizations continuously modernize applications in a DevOps environment with security baked in?
- How to do an assessment of security in real-time?
- How do government organizations get Information Assurance (IA) involved in sprints? How do you move IA mindset from waterfall to agile?
- What is the role of ATOs for DevOps tools? and
- Building trust between security and DevOps team.

"Developers care about security." It is a discussion topic that sparked an in-depth conversation among participants in the room. Security is everyone's responsibility. It is not just the responsibility of a Chief Information Security Officer (CISO) or InfoSec organization, but also that of the operators, managers, and developers in a DevOps culture. As discussion into this topic went deeper, there was a finding that security is not just a feature or a set of regression tests. Rather, it is a process in which the goal of complete security is unreachable; getting close to reaching complete security is the objective.

There was general agreement that security is important in the DevOps and government community. The question is where it falls among competing priorities, especially among

[8]Since industry has not converged on a standard name for these concepts, this paper is avoiding the labels DevSecOps, SecDevOps, and DevOpsSec.

developers. Automated tooling and processes can help inject security into a pipeline but the room began to question if checks, mitigations, and best practices were truly cyber security focused, or instead compliance and mission assurance focused. One participant admitted "your average developer is interested in security, maybe less about compliance." Regardless, following some leading practices and changing culture will increase the quality of software produced in the government which will lead to positive outcomes and greater agility.

3.2.2 Challenges

The participants identified a few questions up-front which steered the discussion. Those key themes included the following:

- How do government organizations define and identify high risk projects?
- How do government organizations get security teams on board with a DevOps environment?
- How do government organizations influence processes in place to adapt to DevOps?
- How do government organizations measure success? and
- How do government organizations build a system security plan up-front and apply it to a CI/CD environment?

3.2.3 Session Summary

The discussions occurred against a backdrop of security professionals being frequently outnumbered by software developers. Given that – in many organizations – there is little incentive for security, developers frequently prioritize features and bug fixes over other tasks, which means other approaches need to be taken to help reduce the likeliness of future errors. When creating blocks of code, developers and managers need to be aware that the blocks of code should be small and manageable enough to fit within the cognizance of a developer and tester's memory. Otherwise logic flaws can appear due to complexity of the code base and contribute to security holes.

The group identified challenges, either hypothetical or from personal experience. Many of these challenges are non-technical in nature. Communication challenges is a common theme, especially with varying degrees of maturity in software development and operations teams. Feedback loops frequently do not include security professionals. Once security is engaged, there is a fear that the cost and amount of effort related to security tasks (ATOs,

assessments, & tests) will cause deliveries to be missed or outweigh the value of undertaking the effort at all. In other cases, operational teams are unaware when a security fix is being introduced into production due to communication breakdowns. Some of this is attributed to a legacy mindset for assessments, individuals in the organization who object to change, people who have valid reservations but have been burned in the past, and the sheer amount of an organization's technical debt.

Given all of this, how does one reduce the burden of security and compliance in the DevOps process while getting more people on board? This question should be shared with senior stakeholders (such as the CISO) on a regular basis to get buy-in and to keep them aware of progress towards improvement. One suggestion was to prioritize security in the development process. Increasing communication opportunities, from formal meetings to brown bag sessions, will lead to improved communications. For example, the organization can add a security component to the user stories and sprints. Furthermore, in every story or change ticket, the team should enumerate what it would take to test. Every ticket should be scored by security and maintenance teams. Security related bugs should be scored higher than other bugs to help keep the emphasis on being secure. Another suggestion was to find better ways to inherit controls and prove that those controls are inherited so that time isn't spent on assessing those items. For example, using accredited base images and platforms as a starting point can allow users and developers to focus on a subset of security items. Leveraging the community and industry will allow the government to share the burden. For example, instituting bug bounty programs, utilizing accredited commercial platforms (e.g., FedRAMP [7]) for development and operations, and using industry tools for scanning and assessment will help.

3.2.4 Recommendations

The group came up with several suggestions to address security challenges, summarized into several key statements.

- Slow down. Build time into the development and deployment process to adequately address security risks
- Compartmentalize as much as possible, perhaps adopting a microservices based architecture
  - This can reduce code complexity, software logic, and reduce the attack surface of a component

  - Build a dependency tree of all components and make sure that teams understand all the dependencies and associated risks in software

- Develop metrics, at least with compliance, that can be measured so that teams can begin to optimize as quickly as possible
- Automate as many security checks and processes (e.g., compliance as code) as possible to allow for humans to focus on exploratory testing and outside-the-box thinking
  - Run security scans before commit, preferably in real-time in the development environment as they code
  - Repeat testing over time to build confidence
- Break down silos among traditionally stove-piped organizations
  - Pull individuals from different organizations (management, infrastructure, etc.) together into red teams
  - Have larger post-mortems that involve everyone

If all members of the team accept shared responsibility of risk, the team members become security experts and stakeholders in the success of the delivery of software. For those who are concerned that they may only have one security person and perhaps ten software developers and operators, sharing the ownership will allow an organization to claim that they really have eleven security professionals on their team.

3.3 DevOps in Health IT

The *DevOps in Health IT* group had limited experience with DevOps. Industry participants had the most experience. Many of the government members were interested in learning about DevOps. The group moved quickly to addressing and understanding the Federal Health IT challenges to adopting DevOps and any other new technologies successfully.

3.3.1 Session Goals

This session had four goals:

- How is applying DevOps to Health IT different from applying it to other domains?
- What are the unique challenges in Health IT?

- What are the lessons learned from applying DevOps in a Health IT environment? and
- What things should be avoided and what should be encouraged?

3.3.2 Challenges

The collaboration session discussions identified the following challenges:

- Government processes are difficult to change without support from senior leadership;
- Healthcare IT is less risk-tolerant; and
- Budget cycles prevent adoption of long-term strategies.

3.3.3 Session Summary

The session began with introductions. Seven of the nine participants identified as being from the government. Their experience with DevOps ranged from one to five years. Most of the participants were interested in learning about DevOps and whether it would be the solution to their IT modernization problems.

The government participants pointed out that Healthcare IT is different because the systems have consequences that are – in effect – "life or death." Therefore, the tolerance for risk taking and learning from mistakes is limited. This is counter to the philosophy underlying DevOps of continuous learning and improvement. The benefits of DevOps and Agile processes are limited without a culture of experimenting and learning. Government processes are engrained and difficult to change without support from senior leadership. Therefore, DevOps requires support from leadership and will be suboptimal if it is solely a grassroots initiative.

Due to federal acquisition practices, many of the technical details related to DevOps may be obfuscated from the acquisition package. This may lead to awarding a contract to the least cost acceptable vendor rather than to a capable vendor. The Federal Information Security Modernization Act (FISMA) [6] compliance was mentioned as an issue along with privacy concerns about protecting patient information. Security is an issue that needs to be addressed as part of DevOps implementation up front.

A recurring theme throughout the session was that the annual budget cycles makes it hard to adopt long-term strategies. The government budgeting process works against adoption of new processes like DevOps and Agile. Budgeting is often short-term focused and works against creating a long-term partnership with vendors. Multiple reviews and approvals by

individuals far removed from the actual development and operations work may limit the ability to introduce DevOps process improvements. Unless there is senior level sponsorship, there is no incentive to take risks to change the status quo.

Recruiting, training, and retaining DevOps and Agile expertise is a challenge in government health IT due to government salary caps and limited career opportunities. Recruitment is a challenge because base salaries and bonuses in the commercial sector compete against stability and mission focus in government health IT. This makes it difficult to attract staff with healthcare domain and IT skills into government service. Government technical training budgets and opportunities are limited compared to the commercial sector.

Attracting highly skilled health IT professionals committed to the healthcare mission who are willing to work for less financial remuneration than the commercial sector limits the pool of skilled staff available to implement in government DevOps. Several participants indicated that the medical researchers were experts at creating software applications. This is due to their commitment to the mission and domain expertise.

The Department of Health and Human Services (DHHS) framework was suggested to the group because "... it is compatible with DevOps EPLC (enterprise performance life cycle)" [12, 13]. Participants will consider whether the framework can be adopted by their organization.

A lack of clarity about the business, mission, and purpose were cited as additional reasons for why health IT is different than other domains. Many of the participants indicated that healthcare is a complex domain unlike more mature domains like accounting and finance. Bridging the gap between healthcare domain knowledge and IT knowledge is tougher because there are so few individuals with both skillsets.

Although DevOps is about streamlining process, decision making by the people who do the work, and automating manual processes; many manual processes are still involved in government IT projects. Other observations by the participants included the following:

- The operations team might not agree to use DevOps principles – the operations team may have the attitude that they "...do not have to because you cannot fire Government";
- Legacy IT and data is another hurdle to the adoption of new software engineering approaches that disrupt the existing status quo;
- Healthcare and medicine is a special domain and requires an expert translator to work between the healthcare domain and DevOps stakeholders;
- Much of the software engineering expertise rests with vendors; and

- Automating legacy system testing is a challenge and will likely remain a manual and time-consuming process.

The non-government participants pointed out that the differences and challenges to implementing DevOps in Healthcare IT were challenges with the government acquisition process, culture, and organizational dynamics.

3.3.4 Recommendations

The participants in the *DevOps in Health IT* collaboration session agreed that a cultural change is required to fully implement and realize the benefits of DevOps. Although risk taking is discouraged, the participants agreed that taking small risks and building on early successes is necessary to build confidence and gain upper management support to make changes. Things that should be encouraged include:

- Continuous learning – move away from the culture of name, blame, and shame to learning by doing and learn from mistakes and make incremental improvements;
- Automate the development and operations processes to reduce time currently spent on manual processes;
- Remove bottlenecks, such as reviews by management who are removed from the technical work;
- Empower staff to make decisions and improvements; and
- Evaluate at the value stream of the activities to remove inefficiencies.

3.4 DevOps and Test

The *DevOps and Test* session explored the following over-arching areas in progression:

- What are the goals of software test in a DevOps pipeline?
- What are the impeding issues and concerns which must be overcome to achieve those goals? and
- What solutions are available? What are the specific recommendations to the community at large?

3.4.1 Session Goals

The team shared many goals, some of which were generally applicable to software testing. Other goals which were specific to software testing in the DevOps context. In the most general sense, software test must clearly ensure that software is functioning as intended. Variations on this theme include: satisfying explicit requirements, meeting business – or mission – needs, meeting or exceeding the expectations of the user base, not failing once deployed, and not causing the failure of other components as deployed.

Goals of test in the DevOps context include: enabling frequent deliveries, shortening delivery cycles, creating process repeatability, identifying bugs as early as possible, improving the code quality, ensuring quality throughout the lifecycle, and enabling decision-makers; (e.g., go/no-go deployment decision). These goals all appear to roll up under two larger themes or rollup-goals:

- deliver mission value faster and
- reduce lifecycle cost of ownership.

A derived goal is a need to increase test automation – increase the adoption across projects and systems; increase its penetration within systems once adopted; increase its fidelity and efficiency and effectiveness once fully embraced. Acceptance Test Driven Development (ATDD) and Behavior Driven Development (BDD) are considered the state-of-the-practice to best enable test automation, and can be considered as further derived goals, as well as solution enablers.

3.4.2 Challenges

The issues and concerns discussed herein are largely roadblocks or impediments. They can be summarized and categorized into four major areas:

- Organizational Inertia;
- Barriers Perceived as Immovable;
- Legacy; and
- Misinterpreted or Misapplied Directives.

3.4.3 Session Summary

The discussions in the session built on the issues and concerns itemized in Section 3.4.2.

3.4.3.1 Organizational Intertia Several impediments can generally be categorized as resistance-to-change. Specific examples of organizational inertia include: formal Independent Verification and Validation (IV&V) requirements, contractual cadences that include items such as Test Readiness Reviews (TRR), and Factory and Site Acceptance Tests. A major source of the inertia is a reluctance to invest early in DevOps enabling test practices, technologies, and techniques. Test Driven Development requires up-front investment in a complex system, with lesser initial measurable value as perceived by some stakeholders. There is little argument that bugs and issues are most efficiently and effectively resolved when found earliest in the lifecycle. Other organizational barriers include the traditional demarcation of development and test teams, which is far from optimal. Further, the traditional allocation of the most creative and proficient engineers to the development team with the test team as a de facto second class is arcane and counter-productive.

3.4.3.2 Barriers Perceived as Immovable Certain mandates (e.g., contractual or legal) present barriers to fully employing DevOps enabling test tactics. In many cases, there are certification processes and requirements for independent (from the development organization) testing to receive accreditation or authorization to operate. Most stakeholders agree that there is limited return on investment when an independent team simply repeats an existing set of test scripts in the same environment to get the same results. Other potentially insurmountable barriers include instances where a test environment that mimics the production is infeasible (e.g., space-based systems, weapons systems). Safety-critical applications and systems may also not be ideal candidates for a legitimate DevOps pipeline because their test strategies and tactics have additional mandates.

3.4.3.3 Legacy There are significant impediments to adopting DevOps enabling test procedures in the case of legacy software systems. In this case, "legacy" means software where there is no baseline of automated tests or that if such a baseline of automated tests exists, then it is inadequate. The best approaches to introduce test automation are not clear. The answer to the "how much testing is enough?" question is elusive.

3.4.3.4 Misinterpreted or Misapplied Directives Some test automation best practices may be misconstrued. One prevalent example is "coverage for the sake of coverage". It is often simple to cover lines and branches of source code, writing tests that assert nothing of value or even nothing at all just to increase the coverage metrics. Teams may not always employ

white-box[9] vs. black-box[10] unit testing optimally, and the over-reliance on unit tests at the expense of a robust set of integration tests is an anti-pattern.

3.4.4 Recommendations

The goals of *DevOps and Test* are achievable and the issues and concerns may be overcome in most cases. This collaboration session offers a set of specific recommendations.

- Adopt the BDD syntax (e.g., Gherkin[11] in lieu of formal shall statements as a requirements baseline. Express the business/mission need as well as the technical functionality
- Provide training on Agile and DevOps philosophies and methodologies to all practitioners, not just to those in specific roles (e.g., ScrumMaster, Product Owner, Test Lead)
- Elevate test artifacts as first-class, subject to same review scrutiny and quality standards as source artifacts.
- Apply the Agile/Scrum industry best practice of cross-functional teams that co-own both the test and source baselines of work products and artifacts
- Define workflows for development activities (e.g., sprint cycles) that include test gates. For example:
  - Acceptance test written,
  - Source written and passes acceptance test,
  - Test and source code reviewed,
  - Static analysis executed and results dispositioned,
  - Security scan complete,
  - Merge request submitted, etc.
- Provide visibility to all stakeholders via metrics around test:
  - Ratios and raw counts of Source Lines of Code (SLOC) and Test Lines of Code (TLOC);

[9] https://en.wikipedia.org/wiki/White-box_testing
[10] https://en.wikipedia.org/wiki/Black-box_testing
[11] http://behat.org/en/latest/user_guide/gherkin.html

  - Derived lifecycle cost savings metrics as return on investment for adopting TDD, BDD; and
  - Time to execute automated test baseline vs. manual test baseline.

- Prioritize introducing automated test to legacy software systems based on traits to include risk and complexity. Mandate automated tests for new capabilities, modifications, and/or bug fixes going forward

3.5 DevOps Culture

This collaboration session focused on a DevOps culture "lifecycle". The participants discussed what drives an organization to change and examined best practices for strategizing to make cultural change. They considered how to initiate and grow a DevOps culture, sustain and nurture it, and looked at the distractors and challenges that might be encountered over the course of the of the DevOps culture journey.

3.5.1 Session Goals

Acknowledging that DevOps requires operations, development, and security/certification culture changes, this session had the goal to answer a single, overarching question – How big is the cultural divide and how do we successfully span it?

3.5.2 Challenges

The collaboration session elaborated on a series of challenges:

- DevOps drivers;
- Strategizing to make a culture change;
- Initiating and growing a DevOps culture;
- Sustaining and nurturing a DevOps culture nurture; and
- Distractors and challenges.

3.5.3 Session Summary

This section elaborated on the high level topics identified in Section 3.5.2.

3.5.3.1 DevOps Drivers An organization's interest in DevOps may be initiated because management is aware of others' embracing DevOps and may be curious about how it might apply to their organization. More often, the system developers and testers may have already embraced Agile and Lean principles and are operating at a speed that operations cannot match.

When this occurs, DevOps becomes a natural organization progression path to increase the overall speed of delivery. Another driver of change in organizations whose time to market is critical for ensuring competitive edge is that the leaders in all levels are incentivized to move to DevOps because there are obvious intrinsic and extrinsic rewards to do so. Similarly, in any type of organization seeking to take advantage of mobile and cloud technologies, the leaders see the value in being able to adapt faster as mission needs and workforce habits change. To be successful at staying relevant, organizations realize that they must be nimble (or learn how to be) and DevOps appears to be a very attractive mechanism to enable them to be agile. In organizations that must tolerate high occurrences of disruption, DevOps promises highly disciplined adaptation at speed. On the flip side, for an organization focused on performance, DevOps becomes an attractive basis to support optimizing.

3.5.3.2 Strategizing to make a Culture Change While DevOps tools may have been in use by parts of an organization, spreading their use creates a social engineering challenge – that of changing culture. Often the new cultural norm must be envisioned and robust strategies need to be followed to ensure success.

Discussions immediately focused on how important it was to get buy-in, align to common goals, and establish a high trust environment. Experiences in the group cited getting buy-in at managerial levels and having functional managers promote showcase studies to provide members of the organization with credible examples of how the DevOps culture can yield success. "Homegrown" examples are especially helpful. Leaders at all levels should invite employees to see what DevOps can do for them, encourage them to experiment and learn in a practical way about DevOps and evangelize about the desirable outcomes that the organizations can have. Leaders should define their objectives and make investments to ensure that employees get educated about DevOps and the Agile and Lean principles in play.

DevOps culture changes affect many stakeholders, all of whom need to adjust to new cultural norms. The organization should take the time to revisit the value that the organization provides and determine if there are gaps and opportunities that DevOps can fix/address. As a start the organization can conduct stakeholder analysis. This step is a precursor to strategizing all the possible touchpoints that will need to culturally change to embrace DevOps.

Some session participants found value stream mapping to be a useful tool to revisit organizational value. They then use the results to plan the change to align with the discovered values of the organization. This approach is also embraced by other organizations who are part of the commercial DevOps movement.

Another important step to strategizing for cultural change is to develop a shared lexicon. One participant who was embarking on cultural change urged that change agents and leaders should read books on the topic. The participant suggested The Phoenix Project: A Novel About IT, DevOps, Helping Your Business Win [8] (which is an instructive story that helps stakeholders understand how organizations can change to get positive outcomes) and Leading Change [9] (which provides a practical eight-step process to make the change).

3.5.3.3 Initiating and Growing a DevOps Culture The audience pointed out that as organizations start the change, the people themselves, how they currently work, and their roles drive different tactics for effecting the culture change. Not all people will be like-minded. There are pairings that often drive adjustment to an initiation to DevOps. The group described the pairings as having varied, sometimes opposing cultural norms that would need to be addressed:

- Development vs. Operations,
- Management roles vs. Functional roles,
- Contractor vs. Government,
- IT vs. Business,
- Internal users vs. Public,
- Companies vs. Individuals,
- Tenured vs. New to organization,
- Union vs. non-union,
- In-person vs. Virtual,
- Pro DevOps vs. Anti DevOps, and
- Task worker vs. Knowledge worker.

One very important action for leaders and managers to focus on is to set up cross-functional teams and empower them quickly. Leaders should encourage teams to use methods such as nominal group techniques[12] so that all opinions are considered in their decision-making. The colocation of cross functional teams accelerates learning and understanding for the other roles on the team, which makes them more effective at understanding how to meet their DevOps goals. The agile cultural goal of doing everything face-to-face is not always achievable with large geographically dispersed teams, but the digital world offers tools that can help an organization set up virtual environments in which collaboration can take place. One way to support dispersed groups is to have in-person kick offs and then use video chat, text chat, and to level the communications field several times a week. This approach, encouraging everyone on the team to communicate this way, improves appreciation for and reinforce continuous, clear communications using electronic tools.

Cultural change is sustained by practice and the Community of Practice construct is an excellent way to connect stakeholders, exchange practice, and amplify good behaviors through feedback and sharing to show the DevOps virtuous cycle[13] in action. It is also a good way to provide evidence of success to those who may be resistant to change. Most organizations start with a small team practicing DevOps. The single most important way to further expand successfully is to put some thought into how to scale the success and then do it. All through the expansion period leaders should support the culture change by coaching individuals and having well-advertised, easily accessible resource centers. As new staff onboard, new staff should be infused with different experiences by having them rotate through different team positions. This helps them grow empathy for the team; they will be more understanding and productive in problem solving because they understand more viewpoints of the problem. For those already in the organization, team members can adapt their current skill sets. This will reduce the fear of losing their jobs (which can be a distractor from effective culture change).

Leaders should work to identify pain points and address them quickly. Where it makes sense, team agreements can be created to help maintain understanding of dependencies as team members ebb and flow.

3.5.3.4 Sustaining and Nurturing a DevOps Culture Nurture Once the DevOps culture is pervasive, establishing a cadence is a good way to sustain the change. Organizations should

[12] This technique focuses on win-win techniques rather than majority rule techniques which tend to alienate team members. Read about how it works at `http://www.sixsigmadaily.com/nominal-group-technique/`.

[13] `http://www.hightechinthehub.com/2012/04/virtuous-cycle-of-devops/`

continue to increase empathy by regular role swapping and collocate teams as much as possible to facilitate face-to-face interactions. One of the larger DevOps culture changes is that of not punishing mistakes. Organizations should use a tool to assess and continue to evolve their organization. With any culture, it is best to reward desired behavior to reinforce it. The reward system should be visible and continuous, showing what is appreciated so that people will be motivated to strive for it.

Organizations should avoid restructuring. It is important to ensure that thoughtful deliberation is used to determine how the culture change will be on-ramped, sustained, and nurtured.

3.5.3.5 Distractors and Challenges For any culture change to be successful, it is necessary for the current stakeholders to see the value. This is done through change agents first understanding the value themselves then making a concerted effort to show the value – through demonstrations and by citing the successes of others.

Often, the fact that the agile team delivery has increased but implementation speed cannot keep up is a good motivator to move to DevOps. The challenge with bringing operations along to embrace the culture change usually lies in negotiating with their leadership and managers (they usually belong to a different organizational entity than development) and determining the best way to help them to embrace DevOps. For those who claim: "We are in flight; cannot stop to make changes", change agents can explain that others are successfully making those changes in flight and that it is possible to make change without stopping. DevOps culture change can start small and expand.

Concerns about lack of accountability, poor awareness of DevOps, perception of losing power, and having a shared responsibility model can all be addressed by education.

If an organization structure does not seem to lend itself to DevOps, one approach is to build matrixed teams and give them a dedicated space to prove out how DevOps can work.

If, after immersion in the DevOps practice, a team member is still not able to embrace the new paradigm, it may be necessary for that individual to find a different role or organization. If the resistance is caused by anxiety about losing one's job due to automation, it can be communicated, for example, that automated scripts must be maintained and written. Some people who may seem to be displaced will move into such positions or other new positions created by the DevOps culture.

3.5.4 Recommendations

Traversing the culture "lifecycle" yielded rich discussion on techniques to strategize for, on-ramp, nurture, and address resistance/distractors. Highlights include:

- Importance of leadership in strategizing and setting the tone of the culture change;
- Being sensitive to the differing cultures caused by previous stove-piped organizations of work and roles;
- Growing cultural change by using communities of practice; and
- Applying techniques and education for addressing resistance and distractors.

4 Conclusions & Summit Recommendations

The March 2018 Federal DevOps Summit highlighted several challenges facing the Federal Government's adoption of DevOps and Agile practices, and presented recommendations to address those challenges

This year's collaboration sessions revealed that there is overall acceptance of the need to move to a DevOps framework and embrace automation. Some key takeaways include the highlights included in this section.

4.1 Challenge Areas

Across the collaboration sessions there was general agreement that there is a need for automation of development, test, deployment and operational processes. This is a paradigm shift from previous summits [5, 4].

There is agreement that the transition to a DevOps framework would shorten delivery times, increase security (due to frequent, even daily scanning and code review), and reduce life-cycle cost of ownership. Even so, many are overwhelmed by the impediments to automation. For example:

- Perception that a domain is too specialized or safety critical to risk DevOps processes;
- Implementing DevOps processes for legacy systems is an intimidating challenge;
- Replacing "one and done" ATO with incremental, ongoing security implementation and review (with all the cultural impacts that implies);

- Traditional stove-piped waterfall organizations are not set up for the rapid cadence of Agile and DevOps (e.g., contractual requirements such as TRR, and Factory and Site Acceptance Tests);
- Recruiting, training, and retaining DevOps and Agile expertise is a challenge in government IT due to government salary caps and limited career opportunities;
- The government budgeting process and organizational dynamics work against adoption of new processes like DevOps and Agile; and
- Manual processes are still deeply embedded in government IT projects.

4.2 Recommended Solutions

Despite the challenges, several recommended best practices are emerging for government DevOps and Agile implementations (listed in this section).

- Start small – move one application into your DevOps environment to learn and demonstrate success and build on these early successes to build confidence and gain upper management support ;
- Embrace risk – learn from mistakes and make incremental improvements;
- Break down silos among traditionally stove-piped organizations; Pull individuals from different organizations to build cross-functional teams;
- Provide training on Agile and DevOps philosophies and methodologies to all practitioners, not just to those in specific roles; and
- Encourage cultural change with active leadership engagement, communities of practice, and continuing to make changes to grow and sustain the DevOps culture.

ACKNOWLEDGMENTS

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 its web site[14].

REFERENCES

[1] J. F. Brunelle, S. Anand, G. Barmine, M. Spina, K. Warren, A. Winston, M. Javid, A. Kemmer, C. Kim, S. Masoud, T. Harvey, and T. Suder. August 2017 ATARC Federal Cloud & Data Center Summit Report. Technical report, The MITRE Corporation; The Advanced Technology Academic Research Center, 2017.

[2] J. F. Brunelle, S. Anand, R. Cagle, C. Kim, M. Kristan, M. Spina, K. Warren, T. Harvey, and T. Suder. February 2017 ATARC Federal Cloud & Data Center Summit Report. Technical report, The MITRE Corporation; The Advanced Technology Academic Research Center, 2017.

[3] J. F. Brunelle, D. Davis, N. Gong, D. Huynh, M. Kristan, M. Malayanur, T. Harvey, and T. Suder. July 2016 ATARC Federal Cloud & Data Center Summit Report. Technical report, The MITRE Corporation; The Advanced Technology Academic Research Center, 2016.

[4] M. Casagni, M. Heeren, R. Cagle, D. Hanf, M. Kristan, J. F. Brunelle, T. Suder, and T. Harvey. 2017 Federal DevOps Summit Report. Technical report, The MITRE Corporation; The Advanced Technology Academic Research Center, 2017.

[5] M. Casagni, M. Heeren, D. Hanf, M. Kristan, S. Kuwana, T. Suder, and T. Harvey. 2016 Federal DevOps Computing Summit Report. Technical report, The MITRE Corporation; The Advanced Technology Academic Research Center, 2016.

[6] Department of Homeland Security. Federal information security modernization act (fisma). `https://www.dhs.gov/fisma`, 2016.

[7] FedRAMP PMO. FedRAMP. `https://www.fedramp.gov/`, 2015.

[8] G. Kim, K. Behr, and G. Spafford. *The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win.* IT Revolution Press, 2013.

[14] `http://atarc.org/devops`

[9] J. P. Kotter. *Leading Change*. Harvard Business Review Press, 2012.

[10] F. Liu, J. Tong, J. Mao, R. Bohn, J. Messina, L. Badger, and D. Leaf. NIST Cloud Computing Reference Architecture. Technical Report Special Publication 500-292, National Institute of Standards and Technology, 2011.

[11] The MITRE Corporation. FFRDCs – A Primer. `http://www.mitre.org/sites/default/files/publications/ffrdc-primer-april-2015.pdf`, 2015.

[12] U.S. Department of Health and Human Services. Enterprise Performance Life Cycle Framework. `https://www.hhs.gov/sites/default/files/ocio/eplc-lifecycle-framework.pdf`, 2012.

[13] U.S. Department of Health and Human Services. Enterprise Performance Life Cycle (EPLC). `https://ocio.nih.gov/PM/Pages/EPLC.aspx`, 2018.