Defense Innovation Board (DIB)
Software Acquisition and Practices (SWAP) Study

Vignettes

v0.2, 4 Mar 2019

To help illustrate some of the issues facing the Department in the area of software acquisition and practices, the SWAP study solicited a set of "vignettes" on different topics of relevance to the study. These vignettes represent "user stories" contributed by study team members and collaborators, and the views expressed here do not necessarily reflect the views of the Subcommittee (though they are consistent with the overall themes contained in the report). The intent of these vignettes is to provide some additional points of view and insights that are more specific and, in some cases, more personal.

List of vignettes:

1. Implementing Continuous Delivery: The JIDO Approach
2. F22: DevOps on a Hardware Platform
3. Marking It Hard to Help: A Self-Denial of Service Attack for the SWAP Study
4. DDS: Fighting the Hiring Process Instead of Our Adversaries
5. Kessel Run: The Future of Defense Acquisitions is #AgileAF
6. JMS: Seven Signs Your Software (Program) Is In Trouble

Vignette #1 – Implementing Continuous Delivery: The JIDO Approach

Forrest J. Shull, SWAP Study Team

v0.1, 6 Feb 2019

One theme that emerges from the work in this study is that DoD certainly does have successes in terms of modern, continuous delivery of software capability; however, in too many cases, these successes are driven by heroic personalities and not supported by the surrounding acquisition ecosystem. In fact, in several cases the demands of the rest of the ecosystem cause friction that, at best, adds unnecessary overhead to the process and slows the delivery of capability. The Joint Improvised Threat Defeat organization (JIDO), within the Defense Threat Reduction Agency, is a compelling example.

JIDO describes itself as "the DoD's agile response mechanism, a Quick Reaction Capability (QRC) as a Service providing timely near-term solutions to the improvised threats endangering U.S. military personnel around the world[1]." As such, the speed of delivery is a key success criteria, and JIDO has had important improvements in this domain as referenced by the additional (following) JIDO vignette in this series. Central to accomplishing these successes has been the adoption of a DevSecOps solution along with a continuous Authorization to Operate (ATO) process, which leverages the automation provided by DevSecOps to quickly assess security issues.

At least as important as the tooling are the tight connections that JIDO has enabled among the stakeholder groups who have to work together with speed to deliver capability. JIDO has personnel embedded in the user communities associated with different COCOMs, referred to as Capability Data Integrators (CDIs). These personnel are required to be familiar with the domain, familiar with the technology, and forward-leaning to envision technical solutions to help warfighter operations. Almost all CDIs have prior military experience and are deployed in the field, moving from one group of users to another, helping to train them on the tools that are available and understanding at the same time what they still need. CDIs have tight reach back to JIDO, and are able to identify important available data that can be leveraged by software functionality and can be developed with speed through the DevSecOps pipeline.

JIDO has also focused on knocking down barriers among contractors and government personnel. JIDO finds value in relying on contract labor that can flex and adapt as needed to the technical work, with effort spent on making sure that the mix of government personnel and multiple contractor organizations can work together as a truly integrated team. To accomplish this, JIDO has created an environment with a great deal of trust between government and contractors. There are responsibilities that are inherently governmental, and tasks that can be delegated to the contractor. Finding the right mix requires experimentation, especially since finding the personnel with the right skillset on the government side is difficult.

Despite these successes bringing together stakeholders within the JIDO team, there are sometimes substantial difficulties working within the rest of the acquisition ecosystem, since on many dimensions the agile/DevSecOps approach does not work well with business as usual. For example, there have been instances where the Services or the Joint Chiefs push back on

[1] JIDO SecDevOps Concept of Operations, v1

solutions that were created to address requirements from the field. Thanks to the CDIs, JIDO can create a technical solution that answers identified requirements from warfighters in the field, but that does not mean it will get approval for deployment. There is a mismatch and potential for miscommunication when the organizations that control deployment do not own the requirements themselves.

Also, because JIDO operates in an agile paradigm in which requirements can emerge and get re-prioritized, it is difficult for them to justify budget requests upfront in the way that their command chain requires. JIDO addresses this today by creating notional, detailed mappings of functionality to release milestones. Since a basic principle of the approach is that capabilities being developed can be modified or re-prioritized with input from the warfighter, this predictive approach provides little or no value to the JIDO teams themselves. Even though JIDO refuses to map functionality in this way more than 2 years out, given that user needs can change significantly in that time, the program has had to add bodies just to pull these reports together.

JIDO has no problem showing value for the money spent. They are able to show numbers of users and, because they have personnel embedded within user communities, can discuss operational impact. As mentioned above, their primary performance metric is “response from the theater.” Currently, JIDO faces a backlog of tasks representing additional demand for more of their services, as well as a demand for more CDIs. Despite these impactful successes, the surrounding ecosystem unfortunately provides little in the way of support and much that hinders the core mission. It is difficult to see how these practices can be replicated in other environments where they can provide positive impact until these organizational mismatches can be resolved.

OPERATIONALIZING DEVSECOPS AT JIDO

A CASE STUDY ON THE MISSION AND BUSINESS IMPACT OF DEVSECOPS

In 2016 the Joint Improvised-Threat Defeat Organization's (JIDO) expanding counter-threat mission necessitated an evolving digital transformation to respond to the warfighter's "tactical edge" Latest Time of Value (LTOV) needs in hours versus days or weeks. JIDO sought to create a DevSecOps enabled Path-to-Production with the intent of realizing mission and business performance impacts originally achieved by Silicon Valley.

THE MISSION

Today, the Defense Threat Reduction Agency's (DTRA), Joint Improvised-Threat Defeat Organization (JIDO) enables Department of Defense (DoD) actions to counter improvised threats with tactical responsiveness. Such action is conducted via anticipatory, rapid acquisition in support of Combatant Commands' efforts to prepare for, and adapt to, battlefield surprise. Current mission sets support counter-terrorism (CT), counter-insurgency (CI), and other related mission areas, including counter-improvised explosive device (IED). JIDO is the DoD's agile response mechanism; a Quick Reaction Capability (QRC) providing timely near-term solutions to the improvised threats endangering U.S. military personnel around the world.

THE CHALLENGE

In 2016 JIDO's expanding counter-threat mission necessitated an evolving digital transformation to respond to the warfighter's "tactical edge" Latest Time of Value (LTOV) needs in hours versus days or weeks. This led JIDO's Mission IT to reimagine its service delivery model, commonly referred to as the "Path-to-Production"; to operationalize the true potential of an Agile approach to software development. JIDO would seek a Path-To-Production that would reduce time to market, build more reliable products, and recover quicker from failure. This had to be accomplished while supporting a big-data solution on three classified networks for 9,600+ active monthly users in the DoD and Intelligence Community (IC).

KPI	Result
Deployment Frequency	↑ 891% Increase
Availability AQL	↑ 3HR (MONTHLY)
Lead Time	↓ 93%
MTTR	↓ 74%
Initial System Authorization	↓ 75%

Table 1: Top 5 DevSecOps KPIs

THE APPROACH

JIDO's approach to deliver mission capabilities faster and more securely to the warfighter was to unify principles from an Agile Software Development Lifecycle (SDLC) with ITIL-based Release and Operations Management while incorporating the automation techniques in Site Reliability Engineering (SRE), creating an end-to-end process to deliver capabilities. The approach encompassed the IT organization's people, processes, and technology; leveraging the latest advances in open source and industry-proven technologies to create a DevSecOps enabled Path-to-Production. By automating repeatable processes, Quality Assurance (QA) and enforcement of security requirements throughout, the organization can ensure a greater level of compliance with federal policies and guidelines while achieving all the benefits of speed, reliability, and efficiency. The approach also ensures a greater level of configuration management across the networks as well as providing advantages in governance. JIDO avoided traditional organizational barriers of policy and bureaucracy in favor of high trust, high collaboration, and secure Agile best practices that span mission, functional, development, and operations and maintenance

domains. Mission capabilities would be delivered faster and more securely, operate with enhanced reliability, and meet stakeholder needs at less cost.

THE SOLUTION

JIDO integrated its people across functional development, engineering, and cybersecurity teams. The DevSecOps teams followed processes geared towards continuous improvement, achieving common focus, and shared priorities. On the technology side, JIDO developed a continuous integration / continuous deployment (CI/CD) pipeline and by "shifting security to the left" and integrating numerous automated security checks into the build process it prevents human error, ensures code supply chain management and enforces policies. Development occurs in an unclassified environment and is continuously delivered to air-gapped classified production networks. JIDO developers and operations engineers deliver fully tested, commit-level increments of new code on a security-hardened, patched, and approved infrastructure enclave for both capability and infrastructure. When automating nearly all the security controls associated with the Application Development Security Technology Implementation Guide (STIG), JIDO has defined a risk-managed software delivery pipeline that minimizes manual human review of every software update or modification. This process provides a real-time view of the systems, networks, and vulnerabilities to the Authorizing Official (AO) while delivering immediate value to operational warfighters.

THE IMPACT

The benefits of DevSecOps for JIDO include both quantitative and non-quantitative impacts to the speed of delivery. The results were disruptive, yielding positive impact across all KPIs measured.

Non-Quantitative Impact

Non-Quantitative results were captured based on observed changes in JIDO's Mission IT culture:

- **Multidisciplinary teams:** Team synergy increased due composition of varied but complimentary experience, increased interaction between teams, daily forced communication, qualifications, and skills. Job satisfaction among participants increased due to visibility of impact.
- **Shared Focus:** Teams began working in a focused manner sharing communications, common priorities and working towards common goals**.**
- **Security:** An organization will never know the disasters that did not occur. Now, the team has a real time understanding of vulnerabilities in its custom code and a rapid way to respond. Therefore, the team can patch custom code the same day a vulnerability is observed where previously it could take weeks or even months. The AO also has an improved understanding of technical risks on the network with transparent dashboards verse static reports.

Quantitative Impact

Over a 6-month period, JIDO measured Key Performance Indicator (KPI) impact against pre and post-DevSecOps enablement:

KPI	Definition	Legacy	DevSecOps Enabled	%/$ Impact
Availability Acceptable Quality Level (AQL)	Service Level Acceptable Quality Level (AQL) for Average Operational Availability of services	99.5%	99.9%	**+3 HRS MONTHLY UPTIME**
Continuous Authorization	Average time to complete code deployment after initial A&A	23 Days	6 Hours	**92% FASTER**
Deployment Frequency	The frequency new code reach customers	11	98 Releases	**891% INCREASE**
Initial System Authorization	Cybersecurity risk assessment threshold determination for pipeline including major system design and compliance with DoD Risk Management Framework	12 Months	3 Month	**75% REDUCTION**
Lead Time Reduction	The time from the start of a development cycle (the first new code) to deployment is the change lead time	169.83 Days	12 Days	**93% REDUCTION**
Mean Time to Provision	The average time that it takes to add additional services to an environment	6 Months	2 Hours	**99.79% REDUCTION**
Mean Time to Recovery	The average time from deployment failure to recovery	15.5 Minutes	4 Minutes	**74% REDUCTION**
Operating Cost	Change in operating costs based on leveraging open source tooling vs legacy COTs dependent architecture	$1.8M	$150K	**91.66% REDUCTION**

Table 2: DevSecOps KPIs

Vignette #2 – F22: DevOps on a Hardware Platform

Craig W. Ulsh, SWAP Study Team

v0.3 , 10 Jan 2019

The F-22A Raptor program recognized a need for greater speed and agility and took action. In mid-2017, the Vice Chief of Staff of the Air Force and Air Force Acquisition leadership realized the F-22A Raptor modernization efforts where not delivering at a speed that would keep pace with emerging threats. Air Force leadership secured the expertise of the Air Force Digital Service (AFDS). A joint team assessed the program and captured a series of observations and recommendations. The overarching assessment was:

> *"The Air Force must move faster, accept a greater amount of risk, and commit to radical change with how the F-22A modernization effort is managed and technology is implemented. Competitors are moving faster, and blaming poor vendor performance will not help the F-22A Raptor remain the dominant air superiority platform."*

The F-22A program office came to the realization that change was needed. The traditional F-22 acquisition process was slow and cumbersome, with initial retrofits taking six years to deliver. The program recognized the following symptoms:

- Requirements were static and rigidly defined
- Capability was delivered in large, monolithic releases
- Change was avoided and treated as a deviation from well-guarded baselines
- Too much focus on intensive documentation
- Marathon test events

More specifically, a number of issues were identified that are common among weapons systems:

Development practices. Development processes were matched to the traditional acquisition process. Large feature sets, multiple baselines, highly manual developer testing tools, and limited focus on continuous software infrastructure upgrades contributed to the slow capability delivery cycle. Several specific recommendations were made under the overarching recommendation for the software development teams to adopt modern software practices.

Planning. Several inefficiencies were identified in the planning process including lack of metrics for estimation of effort, inability to prioritize, and inefficient use of developer time. Again, recommendations to adopt modern agile software processes were proposed.

Organization. Organizational gaps included poor collaboration across teams, lack of incentives for engineering talent, and competing priorities across multiple vendors.

Contracts. The single most significant observation is the failure to prioritize.

In November 2017, the F-22 program office took several steps to accelerate the F-22A modernization efforts. In response to outdated development practices, the program office restructured the TACLink 16 and TACMAN programs into a single agile development stream. To

properly match the contractor's effort with a new development approach, a "level of effort" for prime development labor was adopted. To address some of the planning concerns, steps were taken to adjust program alignments and authorities.

The F-22A Raptor program has made positive steps in adopting a more modern software approach, but the results are yet to be seen. The program office has learned lessons during the transition, including:

- Culture change has been the biggest hurdle
- Recognizing and accepting that things will go wrong
- Security controls limit flexibility and communication

The program is on the right track with a sound plan to accelerate delivery. But, the program office also noted, in the immortal words of Mike Tyson, "Everyone has a plan until they get punched in the face."

Vignette #3 – Making It Hard to Help:
A Self-Denial of Service Attack for the SWAP Study

Richard Murray, SWAP Study Team

v0.4, 3 Mar 2019

The Department makes use of advisory committees consisting of a mixture of government, industry, and academic experts, all trying to help. However, the Department can make it extremely difficult for these groups to function, an example of what we refer to on the Defense Innovation Board (DIB) as a "self-denial of service attack."[2] The DIB Software Acquisition and Practices (SWAP) study is itself a case in point.

<rant>

The DIB SWAP study clock started ticking when the 2018 NDAA was signed on 12 December 2017. We had our first SWAP discussion at the Pentagon on 16 January, before we had officially been requested by the Under Secretary of Defense for Acquisition & Sustainment[3] to start, but knowing this was coming (and using the DIB Science & Technology [S&T] subcommittee to ramp up quickly). We identified potential subcommittee members by 12 February and we were officially charged to carry out the study on 5 April 2018. The one-year Congressionally-mandated end date was thus set as 5 April 2019. The DIB S&T subcommittee sent in the list of suggested subcommittee members. Then we started waiting…

On 24 May, after a DIB meeting, one of the SWAP co-chairs found out that there had been no movement on these positions. He sent a note to the DIB's Executive Director, expressing disappointment and reiterating the importance of getting these members on board early in the study. The Executive Director tried to push things along. More waiting…

The first activity in which any new member of the SWAP subgroup participated was on 1 November 2018, a full 30 weeks after our 52 week countdown started and nine months after we had identified the people we wanted to help with the study. Even getting people on board this late took repeated "interventions" by the DIB staff and, in the end, only 2 of the 4 people that we hoped could help were able to participate in the study. The timing was such that we had already visited 5 of the 6 programs with which we met, written seven of the eight concept papers that we generated, and held three of the four public meetings that provided input for our report.

Why did things take so long? These people were ready to help, had served in government advisory roles in the past, and provided incredibly valuable input in the end (but only in the end). Maybe we need some sort of "FACA Pre ✓" that allows the DoD to make use of people who are willing to help and all we need to do is ask...

---

[2] The DIB first heard this term from one of the military instructors at the Air Force Academy and we now use it all time time.

[3] On 2 January, 2018, the Secretary of Defense delegated the direction of the DIB for the SWAP study to the Undersecretary of Defense (Acquisition, Technology and Logistics), now the Undersecretary of Defense (Acquisition and Sustainment) (USD (A&S)). On April 5, 2018, the USD(A&S), HON Ellen Lord, signed a Terms of Reference that officially launched the SWAP study with an end date of April 5, 2019.

Another example: the SWAP study decided to use Google's G-Suite as our means for writing our report. It had some nice features for collaboration and several of us were familiar with using it. Setting up a G-Suite site is fast and easy, and a member of the study had previously created a site in a matter of minutes and had a fully operational, two-factor authenticated set of accounts up and running in less than a week. It turns out that the Department has the authority to create official G-Suite sites and so we just needed to get permission to use it.

Our request went in ~10 April 2019. The site was created on 8 August 2019, 17 weeks after our request. As near as we can tell, the only thing that happened during the four months that it took to get the site working was that people said "no" and then other people had to spend time figuring out why they said no and either convincing them that this really was useful and a good solution for the study's needs, and/or going above their heads.

A major theme from the beginning of the SWAP study, and more generally the DIB's overall work, has been that DoD technology needs to move at the speed of (mission) need, faster than our adversaries and, certainly, not that much slower than what has proven possible and effective in the private sector. If the Department wants to take advantage of people who can help it be more effective in development and delivery of technology for improving national security, it should figure out how to *quickly* put together groups of people from inside and outside government, provide them with modern collaboration environments, and let them spend their time providing service to the Department instead of struggling with the bureaucracy.

</rant>

Vignette #4 – DDS: Fighting the Hiring Process Instead of Our Adversaries

Sean Brady, Kevin Carter, Maj Justin Ellsworth, SWAP Study Team

v0.1, 30 Jan 2019

In novelist James Patterson and former President Bill Clinton's political thriller, *The President is Missing*, a terrorist group threatens to unleash cyber-warfare on the Western world bringing about the "Dark Ages." The President (in the story) must sneak away from the White House incognito, engage in shootouts, survive an ambush on Memorial bridge, and assemble the best computer scientists from our government and military to take out the impending computer virus before it strikes.

At this point, the novel introduces a top "white hat hacker" who joins the President's team. She impresses the FBI with her hacking abilities and they hire her on the spot. In a sensational thriller, full of suspended disbelief, this was by far the most unbelievable.

There's no way government hiring works that effectively or efficiently.

We know because we tried.

The Defense Digital Service (DDS) is an organization within the Pentagon tasked with driving a giant leap forward in the way the Department builds and deploys technology and digital services. One of DDS's most visible programs is "Hack the Pentagon," the first bug bounty program in the history of the Federal government. Bug bounties (also known as crowd-sourced hacking challenges) allow the government to leverage a diverse range of talent from private citizens to strengthen our nation's security posture in exchange for a monetary reward for finding security issues. Bug bounties are an integral part of private sector security strategies at companies including Microsoft, Google, Twitter, and Facebook.

The winner of one of these Hack the Pentagon challenges was a 17-year-old high school student, who beat out 600 other invited hackers by reporting 30 unique vulnerabilities to the Department. After the challenge, he expressed interest in interning so he could help contribute to our nation's security outside of the challenges.

DDS staff spent the next eight months and approximately 200 man hours trying to navigate the hiring process to bring the hacker onboard. DDS engaged with the Washington Headquarters Service, the Air Force internship program, and U.S. Army Cyber HR organizations to identify applicable hiring authorities and more importantly, the HR specialists who could help drive the hiring actions for a non-traditional, but obviously qualified, candidate.

Unfortunately, what we found was a system ill-equipped to evaluate technical expertise (especially when demonstrated through experience or skill rather than certifications or education) and resistant to leveraging the full flexibilities and authorities provided.

Twice the hacker's resume was rejected as insufficient to qualify him at the necessary grade level for using direct hire authority. Ultimately, the candidate lengthened his resume to a total of five pages which a classifier reviewed and determined would qualify him for the GS-4 level, which equates to less than $16 per hour. (For what it's worth, the GS-5 level only requires "experience

that provided a knowledge of data processing... gained in work such as a computer operator or assistant, [or] computer sales representative…" according to the OPM GS-2210: Information Technology Management Series General Schedule Qualification Standards.) To be clear - he would have qualified if he had worked a year at Best Buy.

Oh, and did we mention he landed on TIME's List of the 25 Most Influential Teenagers of 2018? He is currently studying computer science at Stanford University.

We recognize that it is unreasonable to expect a classification specialist to understand and translate the experience listed in a resume to the education, demonstrated knowledge, and specialized experience requirements that must be met for each grade level in each job series.

The classification specialist may not have known how this particular candidate's listed experience developing *"mobile applications in IonicJS, mobile applications using Angular, and APIs using Node.js, MongoDB, npm, Express gulp, and Babel,"* met or did not meet the classification requirements of *"experience that demonstrated accomplishment of computer-project assignments that required a wide range of knowledge of computer requirements and techniques* pertinent to the position to be filled."

That's why DDS provided a supporting memo to the classifier that identified where the candidate's resume and classification guide matched. However, the HR office refused to accept the supporting document despite OPM guidance that *"It is entirely appropriate (and encouraged!) to use Subject Matter Experts (SMEs) outside of HR to rate and rank applicants and determine the most highly qualified candidates for a position."*

Thankfully, our story, like *The President is Missing* has a happy ending. When it became clear that we would lose the hacker to a competing offer from the private sector, leadership at some of the highest levels of the Pentagon intervened and ordered their HR office to make the hire. With sufficient visibility and the right people assigned, the hacker's original (one page) resume was reviewed and used to hire him at a reasonable, but still below market, rate. We were ultimately able to hire him, but the process required escalation and is not scalable for more than a small number of hires.

The hacker, now 18, joined DDS as an employee during the summer of 2018 and during that time identified numerous vulnerabilities that threatened the security of information and potentially the safety of our nation.

His story was not isolated to one HR specialist or one service. The Department, we made it as hard as possible for him to join (all while the private sector offered higher salaries and housing stipends). Hiring him did not require a new law or regulation; it required an understanding of his technical abilities, trust in those who evaluated him, and leadership that prioritizes people over process.

Vignette #5 – Kessel Run: The Future of Defense Acquisitions Is #AgileAF

Dan Ward

v0.1, 25 Feb 2019

I've seen the future, and it's #agileAF.

That's the hashtag used by an Air Force software company known as Kessel Run – the "AF" stands for Air Force, by the way. And I did say "software company," which is how members of this military unit describe their organization. Kessel Run does not look like any other program office the Air Force has ever seen. That is its great strength. That is its great peril. And that is why it is the future.

What's so great about Kessel Run? For starters, they deliver. As one example from many, in less than 130 days they fielded an accredited Secret Internet Protocol Router (SIPR) cloud-native DevOps platform at Al Udeid Air Base, then replicated the instance at Shaw Air Force Base and fielded another DevOps platform at Osan Air Base in Japan. Don't worry if that last sentence sounded like technobabble – the point is they put stuff into the field quickly. In contrast, the previous program charged with addressing this need (which went by the catchy name "AOC 10.2") spent $430M over ten years before being terminated "without delivering any meaningful capability," to quote Senator John McCain. But while Kessel Run's ability to field operational software is noteworthy, their organizational achievement and the culture they built just might be the real breakthrough.

It turns out disruptive new technologies do not merely require cutting edge tech. They also require new "organizational architectures," to use Professor Rebecca Henderson's term, and very specific cultural features.

Easier said than done, of course. Building and sustaining these innovative structures inside a large legacy organization like the US military requires replacing existing standards and norms. That's even harder than it sounds and is why so many large companies fail to make the switch.

Despite the difficulty, the Kessel Run team seems to have cracked the code and built a unique organization that operates at warp speed. The most visible difference between Kessel Run and business-as-usual military program offices is their location. Rather than spending all their time on the military base they are *technically* assigned to, Kessel Run personnel operate from a brightly lit WeWork office in downtown Cambridge, MA. The conference rooms have Star Wars themed names, instead of mil-standard room numbers. The walls are covered in multi-colored sticky notes. The view of Boston is spectacular. You get the picture.

Only slightly less visible is their approach to contracting. Instead of handing the work over to a major defense contractor, they built a collaborative partnership with a small-ish software company named Pivotal. Together they use DevOps methods like pair programming where Air Force coders work side-by-side with Pivotal coders to produce software that runs on classified military systems and supports real-world military operations.

Where people sit and how they collaborate are just the tip of the iceberg. The Kessel Run culture is the product of hundreds of thoughtful design decisions that continually reinforce principles of learning, collaboration, critical thinking, and agility. The details of these decisions are beyond the scope of this short vignette, but the fact that Kessel Run continues to do the hard work of deliberately crafting and maintaining its culture is absolutely foundational to their story.

Their story is happening right now, so saying "the future is #agileAF" is actually an observation about the present. Kessel Run's approach is what right looks like *today*. They are the new standard of military acquisition excellence, and already the other services are starting to follow suit. Just last month the US Naval Institute's blog had a post titled The Navy's Kessel Run. When your program office's name gets used in a headline like that, it's a sure sign you're doing something right.

Some skeptical commentators have expressed concern about the risks inherent in a high-speed operation like Kessel Run. In response, let's hear from the 4-star commander of US Strategic Command, General John Hyten. He's responsible for the nation's nuclear arsenal, and is precisely the type of serious, thoughtful, risk-averse leader we want in charge of nuclear weapons. If anyone has a definitive professional opinion on Kessel Run's risk profile, it's General Hyten.

On several occasions Gen Hyten has stated that what keeps him up at night is the thought that the US military's technology community has "lost the ability to go fast." This inability to move quickly increases the likelihood of operational shortfalls and degrades our nation's overall defense posture. In General Hyten's assessment, going too slow is far riskier than going too fast. He sounds quite comfortable with Kessel Run's pace.

In a similar vein, Secretary of the Air Force Heather Wilson submitted a report to Congress in October 2018 that described Kessel Run's achievements to date. She wrote "The use of Agile DevOps methodologies… is proving successful and we are able to rapidly deliver cloud native applications that increase operational utility… *We believe we have demonstrated the ability to continuously deliver software that adds value to the warfighter*." (emphasis added)

So, the question is not whether the Kessel Run team delivers good results or addresses the needs of the operational community. They clearly do. Instead, the question is how long it will take the Department of Defense to adopt this organizational innovation on a larger scale. How long will the DoD wait before making Kessel Run-style organizations and culture the default rather than the exception?

Replicating the Kessel Run culture requires more than giving all your conference rooms Star Wars themed names and putting military personnel into civilian clothes. In fact, the best way to replicate the Kessel Run culture is to *not* replicate it exactly. The wisest imitators will use Kessel Run's example for illumination, not imitation. They will learn from Kessel Run's practices, not simply cut and paste them onto existing organizational structures. The wisest imitators will commit to having the difficult, ongoing conversations about values, attitudes, and beliefs that lead to genuine culture

shifts. They will do the hard work of establishing and maintaining a healthy culture that unleashes people's talent and enables them to do their best work.

Kessel Run is not perfect, of course. They have collected a number of critics and skeptics alongside their fans and supporters. Interestingly, no critics see the project's shortcomings more clearly and pointedly than the Kessel Run members themselves. The team is very aware they are still learning, still experimenting, still making mistakes and identifying opportunities for improvement. They are the first to tell you that Kessel Run has problems and struggles. They are quick to agree with some of their critics about ways the program can and should improve. That is the thing I admire most about this team. That just might be the most important practice for the rest of us to follow. And that is precisely why the future is #agileAF.

Vignette #6 – JMS: Seven Signs That Your Software (Program) Is In Trouble

Richard Murray, SWAP Study Team

v0.2, 3 Mar 2019

The DIB SWAP study visited the JMS (JSpOC [Joint Space Operations Center] Mission System) program in August 2018. The team was open and cooperative, and the people working on the project were highly capable and well-intentioned. At the same time, our assessment of the program was that it was doomed to failure. Because the JMS program was restructured after our visit, we felt it was OK to spell out the problems as examples of what can go wrong.

While there were many issues that led to the failure of the JMS program, the following seven are ones that are not a function of the program per se, but rather the process that created it. We thus call these out as general things to look for as indications that your software (program) may be in trouble.

1. The problem is being made harder than it needs to be. JMS Increment 2 has a budget of just under $1B. The basic function of the JMS system is to track objects in space. While there are engineering challenges to doing this with the proper precision, the basic problem is *not that hard*. Our sense was that the project could be converted to an "app" within AOC Pathfinder, or something equivalent. Assign 20-30 [50? 100?] programmers (+ 20% program management, administration) to work on it for 3 years at $10-20M/year, with first capability due in 6 months, increments every 2 weeks (based on user feedback). Interface to existing data sources (via SW interfaces), run in the cloud, use a scalable architecture that can get to 1M objects in the next year or two. Make sure that the "app" architecture can accept a commercial product if one is available that meets the needs of the user (there were some indications this might have already be happening). Target budget: $10-20M/year for first 5 years, $5-15M/year in perpetuity after that.

2. The requirements are outdated. Many of the requirements for JMS Increment 2 appear to trace back to its original inception circa 2000 and/or its restart in 2010. Any software program in which a set of SW requirements were established more than 5 years ago should be shut down and restarted with a description of the desired end date (list of features with specifications) and a prioritization of features that should be targeted for simplest usable functionality.

3. The program organizational structure is designed to slow things down.

Any software program with more than 1 layer of "indirection" between the prime contractor/integrator and the companies doing the engineering work should be shut down and restarted with a set of level-of-effort style contracts that go directly from the system integrator to the companies delivering code. The system integrator should own the architecture, including the design specifications for the components that plug into that architecture.

4. The program contract structure is designed to slow things down even more. The program had at least a dozen contracts with all sorts of small companies and national labs. It was apparently treated as a "COTS" integration problem with lots of pieces, but it was implemented in a way that seemed designed to ensure that nobody could make any progress.

5. The program is implementing "waterfall with sprints" (otherwise known as agile BS). The program was implementing "sprints" of ~six-nine months (agile BS detector alert!). Sprints have hundreds of tasks spread across six development teams. Just coordinating was taking weeks. For a while the program had used four week sprints, but infrastructure was not available to support that cadence. Test happens after delivery of software, with very little automation.

6. The program management office is too big and doesn't know enough about software. We were told there were 200-260 Full Time Equivalents (FTEs) in the program office. The overall program management should be limited 10-20% of the size of the program, so that resources are focused on the development team (including system architects, user interface designers, programmers, etc.), where the main work gets done. The program office must have expertise in software programs so that they are able to utilize contract and oversight structures that are designed for software (not hardware).

7. OT&E is done as a tailgate process. As an ACAT1 program, JMS was mandated to have the Air Force Operational Test and Evaluation Center (AFOTEC) as the test organization. This required the program to freeze its baseline, do the tests, and then wait 120 days for a report. The consequence of this approach was to slow down the program even more.

The JMS program has undergone major changes to address the issues above, and so the criticisms here should be taken as an example of some of the signs that a program is in trouble.

Vignette 1 – Implementing Continuous Delivery: The JIDO Approach

Forrest Shull

One theme that emerges from the work in this study is that DoD certainly does have successes in terms of modern, continuous delivery of software capability; however, in too many cases, these successes are driven by heroic personalities and not supported by the surrounding acquisition ecosystem. In fact, in several cases the demands of the rest of the ecosystem cause friction that, at best, adds unnecessary overhead to the process and slows the delivery of capability. The Joint Improvised-Threat Defeat Organization (JIDO), within the Defense Threat Reduction Agency, is a compelling example.

JIDO describes itself as "the DoD's agile response mechanism, a Quick Reaction Capability (QRC) as a Service providing timely near-term solutions to the improvised threats endangering U.S. military personnel around the world."[11] As such, the speed of delivery is a key success criterion, and JIDO has made important improvements in this domain. Central to accomplishing these successes has been the adoption of a DevSecOps solution along with a continuous ATO process, which exploits the automation provided by DevSecOps to quickly assess security issues.

At least as important as the tooling are the tight connections that JIDO has enabled among the stakeholder groups that have to work together with speed to deliver capability. JIDO has personnel embedded in the user communities associated with different COCOMs, referred to as Capability Data Integrators (CDIs). These personnel are required to be familiar with the domain, familiar with the technology, and forward-leaning in terms of envisioning technical solutions to help warfighter operations. Almost all CDIs have prior military experience and are deployed in the field, moving from one group of users to another, helping to train them on the tools that are available, and at the same time understanding what they still need. CDIs have tight reachback to JIDO and are able to identify important available data that can be leveraged by software functionality and can be developed with speed through the DevSecOps pipeline.

JIDO has also focused on knocking down barriers among contractors and government personnel. JIDO finds value in relying on contractor labor that can flex and adapt as needed to the technical work, with effort spent on making sure that the mix of government personnel and multiple contractor organizations can work together as a truly integrated team. To accomplish this, JIDO has created an environment with a great deal of trust between government and contractors. There are responsibilities that are inherently governmental and tasks that can be delegated to the contractor. Finding the right mix requires experimentation, especially since finding the personnel with the right skillset on the government side is difficult.

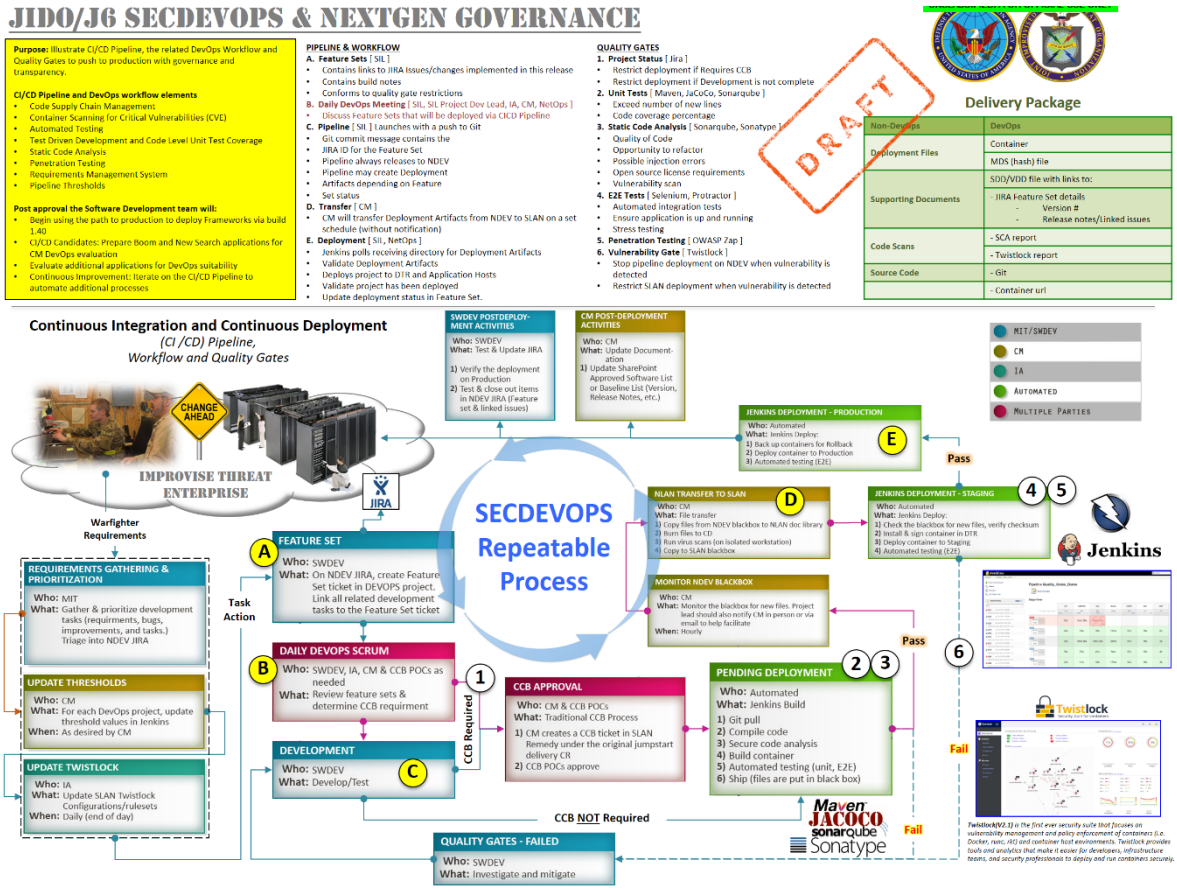
Despite these successes at bringing together stakeholders within the JIDO team, stakeholders in the program management office (PMO) sometimes describe substantial difficulties in working with the rest of the acquisition ecosystem, since on many dimensions the Agile/DevSecOps approach does not work well with business as usual. For example, they describe instances where the Services or the Joint Chiefs push back on solutions that were created to address requirements from the field. Thanks to the CDIs, JIDO can create a technical solution that answers identified

[11] JIDO SecDevOps Concept of Operations, v1.

requirements from warfighters in the field, but that does not mean it will get approval for deployment. There is a mismatch and potential for miscommunication when the organizations that control deployment don't own the requirements themselves.

Also, because JIDO operates in an agile paradigm in which requirements can emerge and get re-prioritized, it is difficult for the organization to justify budget requests upfront in the way that their command chain requires. JIDO addresses this today by creating notional, detailed mappings of functionality to release milestones. Since a basic principle of the approach is that capabilities being developed can be modified or re-prioritized with input from the warfighter, this predictive approach provides little or no value to the JIDO teams themselves. Even though JIDO refuses to map functionality in this way more than 2 years out, given that user needs can change significantly in that time, the program has had to add headcount just to pull these reports together.

JIDO has no problem showing value for the money spent. It is able to show numbers of users and, because it has personnel embedded with user communities, can discuss operational impact. As mentioned above, JIDO's primary performance metric is "response from the theater." Currently, JIDO faces a backlog of tasks representing additional demand for more of its services, as well as a demand for more CDIs. Despite these impactful successes, the surrounding ecosystem unfortunately provides little in the way of support and much that hinders the core mission. It is difficult to see how these practices can be replicated in other environments where they can provide positive impact, until these organizational mismatches can be resolved.



Slide image received from former DTRA-JIDO chief technology officer.

Vignette 2 – F22: DevOps on a Hardware Platform

Craig Ulsh and Maj Zachary McCarty

The F-22A Raptor program recognized a need for greater speed and agility and took action. In mid-2017, the F-22 Program Office realized the F-22A Raptor modernization efforts were not delivering at a speed that would keep pace with emerging threats. Program leadership secured the expertise of the Air Force Digital Service (AFDS). A joint team assessed the program and captured a series of observations and recommendations. The overarching assessment was:

> The Air Force must move faster, accept a greater amount of risk, and commit to radical change with how the F-22A modernization effort is managed and technology is implemented. Competitors are moving faster, and blaming poor vendor performance will not help the F-22A Raptor remain the dominant air superiority platform.

The F-22A Program Office realized that change was needed. The F-22 acquisition process, steeped in the traditional DoDI 5000 model, was slow and cumbersome, with initial retrofits taking at least 6 years to deliver. The program recognized the following symptoms:

- Requirements were static and rigidly defined.
- Capability was delivered in large, monolithic releases.
- Change was avoided and treated as a deviation from well-guarded baselines.
- The development team placed too much focus on intensive documentation.
- Separate programs with separate contracts drove inefficiencies and conflicting interests.
- Insufficient automation for incremental testing resulted in marathon test events. More specifically, the team identified a number of issues that are common among weapon systems:

Development practices. Development processes were matched to the traditional acquisition process. Large feature sets, multiple baselines, highly manual developer testing tools, and limited focus on continuous software infrastructure upgrades contributed to the slow capability delivery cycle. The team made several specific recommendations under the overarching recommendation for the software development teams to adopt modern software practices.

Planning. Several inefficiencies were identified in the planning process including lack of metrics for estimation of effort, inability to prioritize, and inefficient use of developer time. Again, the team proposed that the program adopt modern agile software processes.

Organization. Organizational gaps included poor collaboration across teams, lack of incentives for engineering talent, and competing priorities across multiple vendors.

Contracts. The single most significant observation is the failure to prioritize.

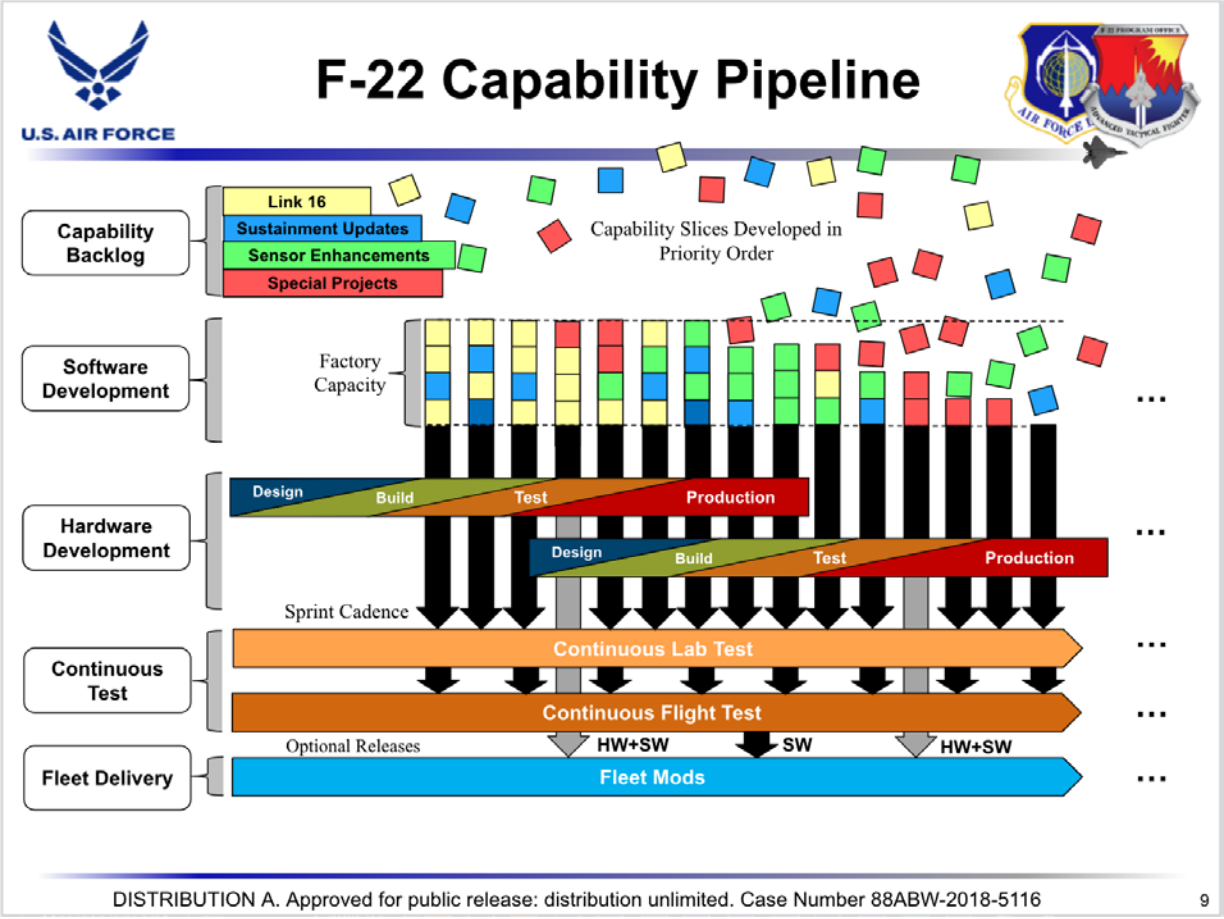
In November 2017, the F-22 Program Office took several steps to accelerate the F-22A modernization efforts. In response to outdated development practices, the program office restructured TACLink 16 and TACMAN programs into a single agile development stream. To properly match the contractor effort with a new development approach, a “level of effort” for prime

development labor was adopted. To address some of the planning concerns, steps were taken to adjust program alignments and authorities.

The F-22A Raptor program has made positive steps in adopting a more modern approach to both hardware and software acquisition. Perhaps the best example is a new contract structure that allows for quick reaction to emerging requirements and changing user priorities while incentivizing a long-time incumbent contractor for continuous improvement. The Program Office has learned lessons during the transition to more agile approaches, including:

- Culture change has been the biggest hurdle.
- The program must recognize and accept that things will go wrong.
- Security controls limit flexibility and communication.

The program is on the right track with a sound plan to accelerate delivery. But the program office also noted, in the immortal words of Mike Tyson, “Everyone has a plan until they get punched in the face.”



Slide image received for briefing from F22A Raptor Program Office.

Vignette 3 – Making It Hard to Help:
A Self-Denial of Service Attack for the SWAP Study

Richard Murray

DoD makes use of advisory committees consisting of a mixture of government, industry, and academic experts, all trying to help. However, the Department can make it extremely difficult for these groups to function, an example of what we refer to on the Defense Innovation Board (DIB) as a "self-denial of service attack."[12] The DIB SWAP study is itself a case in point.

<rant>

The DIB Software Acquisition and Practices (SWAP) study clock started ticking when the 2018 NDAA was signed on 12 December 2017. We had our first SWAP discussion at the Pentagon on 16 January 2018, before we had officially been requested by the Under Secretary for Defense (Acquisition and Sustainment) to start, but knowing this was coming (and using the DIB Science & Technology [S&T] committee to ramp up quickly). We identified potential subcommittee members by 12 February, and we were officially charged to carry out the study on 5 April 2018. The one-year Congressionally-mandated end date was thus set as 5 April 2019. The DIB S&T subcommittee submitted the list of suggested subcommittee members. Then we started waiting…

On 24 May, after a DIB meeting, one of the SWAP co-chairs found out that there had been no movement on these positions. He sent a note to the DIB's Executive Director, expressing disappointment and reiterating the importance of getting these people on board early in the study. The Executive Director tried to use this note to push things along. More waiting…

The first activity in which any new member of the SWAP subgroup participated took place on 1 November 2018— a full 30 weeks after our 52-week countdown started and 9 months after we had identified the people whom we wanted to enlist in to help in our study. Even this took repeated interventions by the DIB staff and, in the end, only two of the four people who we hoped could help were able to participate in the study. The timing was such that we had already visited five of the six programs with which we met, written seven of the eight concept papers that we generated, and held three of the four public meetings that provided input for our report.

Why did things take so long? These people were ready to help, had served in government advisory roles in the past, and provided incredibly valuable input in the end (but only in the end). Maybe we need some sort of "FACA Pre ✓" that allows DoD to make use of people who are willing to help and all we need to do is ask.

Another example: the SWAP study decided to use Google's G Suite as the means for writing our report. It had some nice features for collaboration and several of us were familiar with using it. Setting up a G Suite site is fast and easy, and a member of the study had previously created a site in a matter of minutes and had a fully operational, two-factor authenticated set of accounts

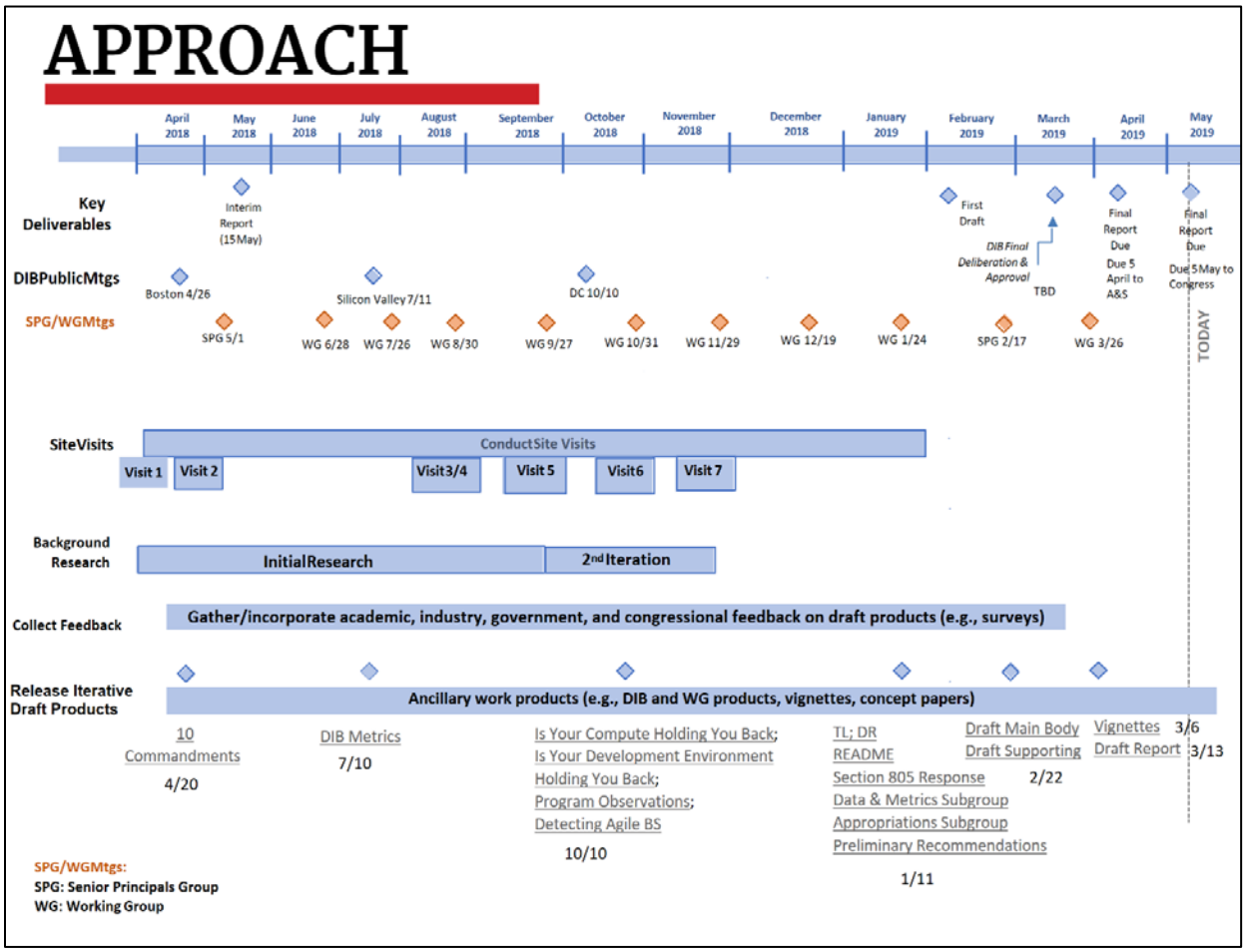
[12] The DIB first heard this term from one of the military instructors at the Air Force Academy and we now use it all the time.

up and running in less than a week. It turns out that the Department has the authority to create official G Suite sites and so we just needed to get permission to use it.

Our request went in ~10 April 2018. The site was created on 8 August 2018, 17 weeks after our request. As near as we can tell, the only thing that happened during the 4 months that it took to get the site working was that people said “no” and then other people had to spend time figuring out why they said no and either convincing them that this really was useful and a good solution for the study’s needs and/or going above their heads.

A major theme from the beginning of the SWAP study, and more generally in the DIB’s overall work, has been that DoD technology must move at the speed of (mission) need, faster than our adversaries and, certainly, not that much slower than what has proven possible and effective in the private sector. If the Department wants to take advantage of people who can help it be more effective in development and delivery of technology for improving national security, it should figure out how to *quickly* put together groups of people from inside and outside government, provide them with modern collaboration environments, and let them spend their time providing service to the Department instead of struggling with the bureaucracy.

</rant>



SWAP study schedule (used for briefings).

Vignette 4 – DDS: Fighting the Hiring Process Instead of Our Adversaries

Sean Brady, Kevin Carter, Justin Ellsworth

In novelist James Patterson and former President Bill Clinton's political thriller, *The President Is Missing*, a terrorist group threatens to unleash cyber-warfare on the Western World, bringing about the "Dark Ages." The President (in the story) must sneak away from the White House incognito, engage in shootouts, survive an ambush on Memorial Bridge, and assemble the best computer scientists from our government and military to take out the impending computer virus before it strikes.

At this point, the novel introduces a top "white hat hacker" who joins the President's team. She impresses the FBI with her hacking abilities and the Bureau hires her on the spot. In a sensational thriller that constantly demands suspended disbelief, this was by far the most unbelievable.

There's no way government hiring works that effectively or efficiently.

We know because we tried.

The Defense Digital Service (DDS) is an organization within the Pentagon tasked with driving a giant leap forward in the way DoD builds and deploys technology and digital services. One of DDS's most visible programs is Hack the Pentagon, the first bug bounty program in the history of the federal government. Bug bounties (also known as crowd-sourced hacking challenges) allow private citizens to harness their diverse range of talents to contribute and strengthen our nation's security posture in exchange for a monetary reward for finding security issues. Bug bounties are an integral part of private-sector security strategies at companies including Microsoft, Google, Twitter, and Facebook.

The winner of one of these Hack the Pentagon challenges was a 17-year-old high school student, who beat out 600 other invited hackers by reporting 30 unique vulnerabilities to the Department. After the challenge, he expressed interest in interning so he could help contribute to our nation's security outside of the challenges.

DDS staff spent the next 8 months and approximately 200 man hours trying to navigate the hiring process to bring the hacker onboard. DDS engaged with the Washington Headquarters Service, the Air Force internship program, and U.S. Army Cyber HR organizations to identify applicable hiring authorities and, more important, the HR specialists who could help drive the hiring actions for a non-traditional, but obviously qualified, candidate.

Unfortunately, what we found was a system ill-equipped to evaluate technical expertise (especially when demonstrated through experience or skill rather than certifications or education) and resistant to leveraging the full flexibilities and authorities provided.

Twice the hacker's resume was rejected as insufficient to qualify him at the necessary grade level for using direct hire authority. Ultimately, the candidate lengthened his resume to a total of five pages, which a classifier reviewed and determined would qualify him for the General Schedule (GS)-4 level, which equates to less than $16 per hour. (For what it's worth, the GS-5 only requires "experience that provided a knowledge of data processing ... gained in work such as a computer

operator or assistant, [or] computer sales representative…" according to the OPM GS-2210: Information Technology Management Series General Schedule Qualification Standards). We like to point out that he would have qualified if he had worked a year at Best Buy.

Oh, and did we mention he landed on *TIME*'s List of the 25 Most Influential Teenagers of 2018? He is currently studying computer science at Stanford University.

We recognize that it is unreasonable to expect a classification specialist to understand and translate the experience listed in a resume into the education, demonstrated knowledge, and specialized experience requirements that must be met for each grade level in each job series.

The classification specialist may not have known how this particular candidate's listed experience developing *"mobile applications in IonicJS, mobile applications using Angular, and APIs using Node.js, MongoDB, npm, Express gulp, and Babel,"* met or did not meet the classification requirements of *"experience that demonstrated accomplishment of computer-project assignments that required a wide range of knowledge of computer requirements and techniques pertinent to the position to be filled.*"

This is why DDS provided a supporting memo to the classifier that identified where the candidate's resume and classification guide matched. However, the HR office refused to accept the supporting document despite OPM guidance that *"It is entirely appropriate (and encouraged!) to use Subject Matter Experts (SMEs) outside of HR to rate and rank applicants and determine the most highly qualified candidates for a position."*

Thankfully, our story, like *The President Is Missing*, has a happy ending. When it became clear that we would lose the hacker to a competing offer from the private sector, leaders at some of the highest levels of the Pentagon intervened and ordered their HR office to make the hire. With sufficient visibility and the right people assigned, the hacker's original (one-page) resume was reviewed and used to hire him at a reasonable but still below-market rate. We were ultimately able to hire him, but the process required escalation and is not scalable for more than a small number of hires.

The hacker, now 18, joined DDS as an employee during the summer of 2018 and during that time identified numerous vulnerabilities that threatened the security of information and potentially the safety of our nation.

His story was not isolated to one HR specialist or one service. As a Department, we made it as hard as possible for him to join (all while the private sector offered higher salaries and housing stipends). Hiring him did not require a new law or regulation; it required an understanding of his technical abilities, trust in those who evaluated him, and leadership that prioritizes people over process.

Vignette 5 – Kessel Run: The Future of Defense Acquisitions Is #AgileAF

Dan Ward

I've seen the future, and it's #agileAF.

Kessel Run's lab director welcomes new engineers. [U.S. Air Force photo by Todd Maki]

That's the hashtag used by an Air Force software company known as Kessel Run—the "AF" stands for Air Force, by the way. And I did say "software company," which is how members of this military unit describe their organization. Kessel Run does not look like any other program office the Air Force has ever seen. That is its great strength. That is its great peril. And that is why it is the future.

What's so great about Kessel Run? For starters, it delivers. As one example from many, in less than 130 days Kessel Run fielded an accredited Secret Internet Protocol Router (SIPR) cloud-native DevOps platform at Al Udeid Air Base, then replicated the instance at Shaw Air Force Base and fielded another DevOps platform at Osan Air Base in Japan. Don't worry if that last sentence sounded like technobabble—the point is they put stuff into the field quickly. In contrast, the previous program charged with addressing this need (which went by the catchy name "AOC 10.2") spent $430 million over 10 years before being terminated "without delivering any meaningful capability," to quote Senator John McCain. But while Kessel Run's ability to field operational software is noteworthy, its organizational achievement and the culture the team has built just might be the real breakthrough.

It turns out disruptive new technologies do not merely require cutting-edge tech. They also require new *organizational architectures*, to use Professor Rebecca Henderson's term, and very specific cultural features.

Easier said than done, of course. Building and sustaining these innovative structures inside a large legacy organization like the U.S. military requires replacing existing standards and norms. That's even harder than it sounds and is why so many large companies fail to make the switch.

Despite the difficulty, the Kessel Run team seems to have cracked the code and built a unique organization that operates at warp speed. The most visible difference between Kessel Run and business-as-usual military program offices is their location. Rather than spending all their time on the military base they are *technically* assigned to, Kessel Run personnel operate from a brightly lit We Work office in downtown Cambridge, MA. The conference rooms have Star Wars–themed names instead of Mil-Standard room numbers. The walls are covered in multi-colored sticky notes. The view of Boston is spectacular. You get the picture.

Only slightly less visible is Kessel Run's approach to contracting. Instead of handing the work over to a major defense contractor, team members built a collaborative partnership with a small-ish software company named Pivotal. Together they use DevOps methods like pair programming,

where Air Force coders work side-by-side with Pivotal coders to produce software that runs on classified military systems and supports real-world military operations.

Where people sit and how they collaborate are just the tip of the iceberg. The Kessel Run culture is the product of hundreds of thoughtful design decisions that continually reinforce principles of learning, collaboration, critical thinking, and agility. The details of these decisions are beyond the scope of this short vignette, but the fact that Kessel Run continues to do the hard work of deliberately crafting and maintaining its culture is absolutely foundational to its success story.

That story is happening right now, so saying "the future is #agileAF" is actually an observation about the present. Kessel Run's approach is what right looks like *today*. Kessel Run is the new standard of military acquisition excellence, and already the other Services are starting to follow suit. Just last month the U.S. Naval Institute's blog had a post titled The Navy's Kessel Run. When your program office's name gets used in a headline like that, it's a sure sign you're doing something right.

Some skeptical commentators have expressed concern about the risks inherent in a high-speed operation like Kessel Run. In response, let's hear from the four-star commander of U.S. Strategic Command, General John Hyten. He's responsible for the nation's nuclear arsenal and is precisely the type of serious, thoughtful, risk-averse leader we want in charge of nuclear weapons. If anyone has a definitive professional opinion on Kessel Run's risk profile, it's General Hyten.

On several occasions General Hyten has stated that what keeps him up at night is the thought that the U.S. military's technology community has "lost the ability to go fast." This inability to move quickly increases the likelihood of operational shortfalls and degrades our nation's overall defense posture. In General Hyten's assessment, going too slow is far riskier than going too fast. He sounds quite comfortable with Kessel Run's pace.

In a similar vein, Secretary of the Air Force Heather Wilson submitted a report to Congress in October 2018 that described Kessel Run's achievements to date. She wrote "The use of Agile DevOps methodologies … is proving successful and we are able to rapidly deliver cloud native applications that increase operational utility. … *We believe we have demonstrated the ability to continuously deliver software that adds value to the warfighter.* " (emphasis added.)

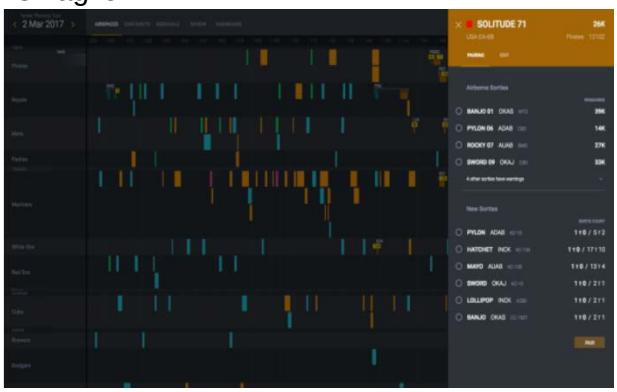
So the question is not whether the Kessel Run team delivers good results or addresses the needs of the operational community. It clearly does. Instead, the question is how long it will take the Department of Defense to adopt this organizational innovation on a larger scale. How long will DoD wait before making Kessel Run-style organizations and culture the default rather than the exception?

Replicating the Kessel Run culture requires more than giving all your conference rooms Star Wars-themed names and putting military personnel into civilian clothes. In fact, the best way to replicate the Kessel Run culture is to *not* replicate it exactly. The wisest imitators will use Kessel Run's example for illumination, not imitation. They will learn from Kessel Run's practices, not simply cut and paste them onto existing organizational structures. The wisest imitators will commit to having the difficult, ongoing conversations about values, attitudes, and beliefs that lead to

genuine culture shifts. They will do the hard work of establishing and maintaining a healthy culture that unleashes people's talent and enables them to do their best work.

Kessel Run is not perfect, of course. It has collected a number of critics and skeptics alongside its fans and supporters. Interestingly, no critics see the project's shortcomings more clearly and pointedly than the Kessel Run members themselves. The team members are very aware they are still learning, still experimenting, still making mistakes and identifying opportunities for improvement. They are the first to tell you that Kessel Run has problems and struggles. They are quick to agree with some of their critics about ways the program can and should improve. That is the thing I admire most about this team. That just might be the most important practice for the rest of us to follow. And that is precisely why the future is #agileAF.

Whiteboard on which tanker refueling operations were planned. [Photo by U.S. Air Force]



The tanker refueling planning app that replaced the AOC's whiteboard. [Photo by U.S. Air Force]

Air Force Kessel Run Headquarters in Boston, MA. [U.S. Air Force photo by J.M. Eddins Jr.]

Vignette 6 – JMS: Seven Signs That Your Software (Program) Is in Trouble

Richard Murray

The DIB SWAP study visited the JMS (JSpOC [Joint Space Operations Center] Mission System) program in August 2018. The JMS team was open and cooperative, and the people working on the project were highly capable and well-intentioned. At the same time, our assessment of the program was that it was doomed to failure. Because the JMS program was restructured after our visit, we felt it was OK to spell out the problems as examples of what can go wrong.

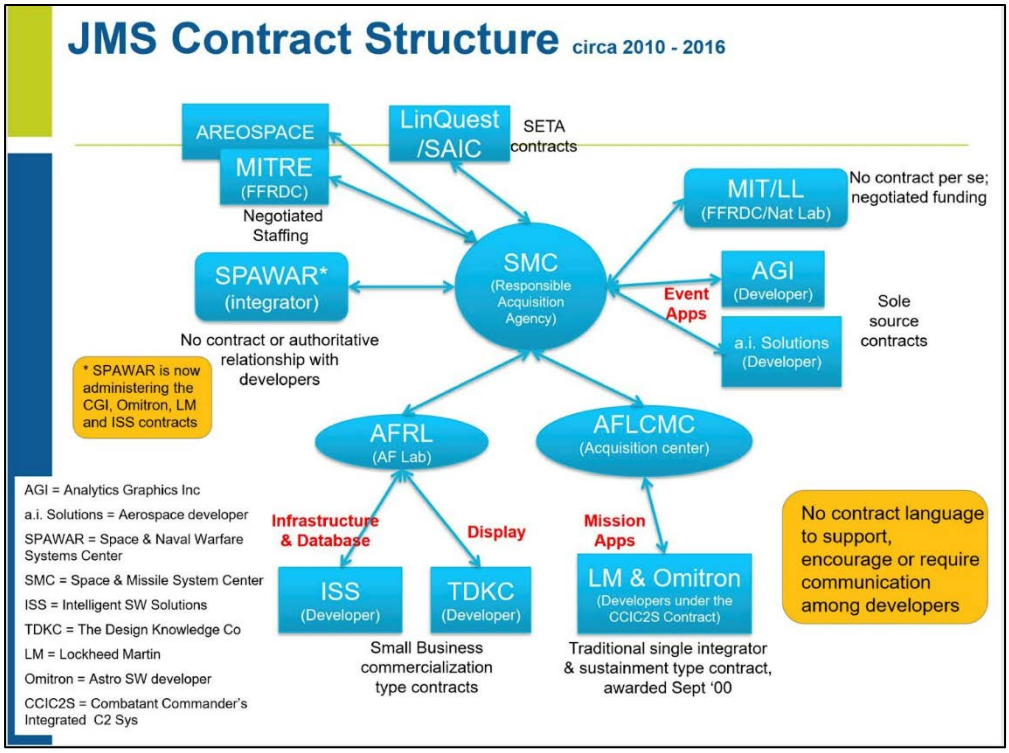
While there were many issues that led to the failure of the JMS program, the following seven are ones that are not a function of that program *per se*, but rather of the process that created it. We thus call these out as general things to look for as indications that your software (program) may be in trouble.

1. The problem is being made harder than it needs to be. JMS increment 2 had a budget of just under $1B. The basic function of the JMS system was to track objects in space. While there are engineering challenges to doing this with the proper precision, the basic problem is *not that hard*. Our sense was that the project could be converted to an "app" within AOC Pathfinder, or something equivalent. Assign 20–30 [50? 100?] programmers (+ 20% program management, administration) to work on it for 3 years at $10–20M/year, with first capability due in 6 months and increments every 2 weeks (based on user feedback). Interface to existing data sources (via software interfaces), run in the cloud, and use a scalable architecture that can get to 1M objects in the next year or two. Make sure that the app architecture can accept a commercial product if one is available that meets the needs of the user (there were some indications this might have already been happening). Target budget: $10–20M/year for first 5 years, $5–15M/year in perpetuity after that.

2. The requirements are outdated. Many of the requirements for JMS increment 2 appeared to trace back to its original inception circa 2000 and/or its restart in 2010. Any software program in which a set of software requirements was established more than 5 years ago should be shut down and restarted with a description of the desired end state (list of features with specifications) and a prioritization of features that should be targeted for simplest usable functionality.

3. The program organizational structure is designed to slow things down. Any software program with more than one layer of indirection between the prime contractor/integrator and the companies doing the engineering work should be shut down and restarted with a set of level-of-effort–style contracts that go directly from the system integrator to the companies delivering code. The system integrator should own the architecture, including the design specifications for the components that plug into that architecture.

4. The program contract structure is designed to slow things down even more. The program had at least a dozen contracts with all sorts of small companies and National Labs. It was apparently treated as a COTS integration problem with lots of pieces, but it was implemented in a way that seemed designed to ensure that nobody could make any progress.



JMS contract structure. [Photo courtesy of former JMS program office]

5. The program is implementing "waterfall with sprints" (otherwise known as Agile BS). The program was implementing "sprints" of ~6–9 months (Agile BS detector alert!). Sprints had hundreds of tasks spread across six development teams. Just coordinating was taking weeks. For a while the program had used 4-week sprints, but infrastructure was not available to support that cadence. Test happened after delivery of software, with very little automation.

6. The program management office is too big and does not know enough about software. We were told there were 200–260 FTEs in the program office. The overall program management should be limited to 10–20% of the size of the program so that resources are focused on the development team (including system architects, user interface designers, programmers, etc.), where the main work gets done. The program office must have expertise in software programs so that it is able to utilize contract and oversight structures that are designed for software (not hardware).

7. OT&E is done as a tailgate process. As an ACAT1 program, JMS was mandated to conduct operational test, a process that nominally required the program to freeze its baseline, do the tests, and then wait 120 days for report. The Operational User Evaluation conducted in early 2018 was terminated early by the Air Force due to poor performance of the system. The OT&E process being used by the program added information to support the termination decision, but it is important to note that had the program not been terminated the tailgate nature of the evaluation was one that would have added further delays.

The JMS program has since undergone major changes to address the issues above, so the criticisms here should be taken as an example of some of the signs that a program is in trouble.