Software is Never Done:
Refactoring the Acquisition Code for Competitive Advantage

Defense Innovation Board (v3.3, 12 Mar 2019)

J. Michael McQuade and Richard M. Murray (co-chairs)
Gilman Louie, Milo Medin, Jennifer Pahlka, Trae Stephens

Extended Abstract

U.S. national security increasingly relies on software to execute missions, integrate and collaborate with allies, and manage the defense enterprise. The ability to develop, procure, assure, deploy, and continuously improve software is thus central to national defense. At the same time, the threats that the United States faces are changing at an ever increasing pace, and the Department of Defense's (DoD's) ability to adapt and respond is now determined by its ability to develop and deploy software to the field rapidly. The current approach to software development is broken and is a leading source of risk to DoD: it takes too long, is too expensive, and exposes warfighters to unacceptable risk by delaying their access to tools they need to ensure mission success. Instead, software should enable a more effective joint force, strengthen our ability to work with allies, and improve the business processes of the DoD enterprise.

Countless past studies have recognized the deficiencies in software acquisition and practices within DoD, but little seems to be changing. Rather than simply reprint the 1987 Defense Science Board (DSB) study on military software that pretty much said it all, the Defense Innovation Board's (DIB's) congressionally mandated study[1] on Software Acquisition and Practices (SWAP), has taken a different approach. By engaging Congress, DoD, federally funded research and development centers (FFRDCs), contractors, and the public in an active and iterative conversation about how DoD can take advantage of the strength of the U.S. commercial software ecosystem, we hope to move past the myriad reports and recommendations that have so far resulted in little progress. Past experience suggests we should not anticipate that this report will miraculously result in solutions to every obstacle we have found, but we hope that the two year conversation around it will provide the impetus for figuring out how to make the changes for which everyone is clamoring.

In this iteration of our report, we emphasize three fundamental themes:

1. **Speed and cycle time are the most important metrics for managing software**. To maintain advantage, DoD needs to procure, deploy, and update software that works for its users at the speed of mission need, executing more quickly than our adversaries. Statutes, regulations and cultural norms that get in the way of deploying software to the field quickly weaken our national security and expose our nation to risk.
2. **Software is made by people and for people, so digital talent matters**. DoD's current personnel processes and culture will not allow its military and civilian software capabilities to grow

[1] Section 872 of the FY18 National Defense Authorization Act (NDAA) directed the Secretary of Defense to "direct the Defense Innovation Board to undertake a study on streamlining software development and acquisition regulations." The DIB-SWAP members were charged to "review the acquisitions regulations applicable to, and organizational structures within, the Department of Defense…; review ongoing software development and acquisition programs…; produce specific and detailed recommendations…; and produce such additional recommendations for legislation." See Section 872 of the FY18 NDAA at https://www.congress.gov/115/plaws/publ91/PLAW-115publ91.pdf.

nearly fast or deep enough to meet its mission needs. New mechanisms are needed for attracting, educating, retaining, and promoting digital talent, and for supporting the work- force to follow modern practices, including developing software hand in hand with users.

3. **Software is different than hardware (and not all software is the same)**. Hardware can be developed, procured, and maintained in a linear fashion. Software is an enduring capability that must be supported and continuously improved throughout its lifecycle. DoD's acquisition process and culture need to be streamlined for effective delivery and oversight of multiple types of software-enabled systems, at scale, and at the speed of relevance.

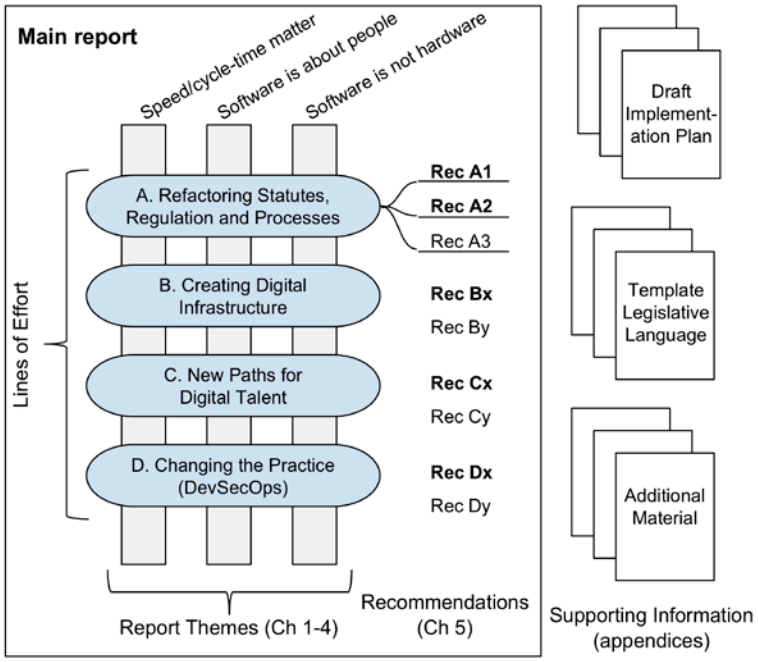
To take advantage of the power of software, we recommend four primary lines of effort:

A. **Congress and DoD should refactor statutes, regulations, and processes for software**, enabling rapid deployment and continuous improvement of software to the field and providing increased insight to reduce the risk of slow, costly, and overgrown programs.

B. **OSD and the Services should create and maintain cross-program/cross-service digital infrastructure** that enables rapid deployment, scaling, testing, and optimization of software as an enduring capability; manage them using modern development methods; and eliminate the existing hardware-centric regulations and other barriers.

C. **The Services will need to create new paths for digital talent (especially *internal* talent)** by establishing software development as a high-visibility, high-priority career track and increasing the level of understanding of modern software within the acquisition workforce.

D. **DoD and industry must change the practice of how software is procured and developed** by adopting modern software development approaches, prioritizing speed as the critical metric, ensuring cyber protection is an integrated element of the entire software lifecycle, and purchasing existing commercial software whenever possible.



Report structure. The main report provides an assessment of the current and desired states for software acquisition and practices, as well as a review of previous reports and an assessment of why little has changed in the way DoD acquires software, with emphasis on three fundamental themes. The report's recommendations are broken into four lines of effort, with a set of key recommendations provided for each (bold), along with additional recommendations that can provide additional improvements. For each recommendation, a draft implementation plan is presented, and potential legislative language is also provided.

Table of Contents

v3.3, 12 Mar 2019

Supporting Information (separate document, ~180 pages)[2]

[2] Page numbers reflect this document and "TBD" for docs that were either previously released, released separately, or will be included in subsequent releases.

Chapter 0. README

In 2011, Marc Andreessen claimed in an op-ed for *The Wall Street Journal* that "Software Is Eating the World."[3] He argued that *every* industry (not just those considered to be "information technology") would be transformed by software – bytes rather than atoms. Eight years later, it is clear he was right.

This transformation is happening in defense, and we are not prepared for it. Software is leveling the playing field with our rivals, eroding the advantages we have spent many decades accruing. Software is the focal point of many important advances in national security technology, including data analytics, artificial intelligence, machine learning, and autonomy. Software is ubiquitous, part of everything the Department of Defense (DoD) does from logistics to management to weapons systems. U.S. national security superiority is critically dependent on the capabilities of the DoD's software.

If this is true, then DoD must be able to develop, procure, assure, deploy, and continuously improve software faster than our adversaries. Unfortunately, DoD still treats software much like hardware, and misunderstands the relationship between speed and security. As a result, the vast majority of the DoD's software takes far too long, costs far too much, and is far too brittle to be competitive in the long run. If DoD does not take steps to modernize its software acquisition and development practices, we will no longer have the best military in the world, no matter how much we invest or how talented and dedicated our armed forces may be.

The good news is that there are organizations within DoD that have already acknowledged the risks of falling further behind in software and are leveraging more modern acquisition and development practices with notable success. The Defense Digital Service (DDS), the Defense Innovation Unit (DIU), the Joint Improvised Threats Defense Organization (JIDO), and the Air Force's Kessel Run demonstrate that DoD has the ability to ship world-class software. The challenge remains doing this at scale.

DoD needs to build on these foundations to create an ecosystem and standard operating procedures that enable the practices of great software without requiring employees to "hack the system." To do that, we must address the prioritization, planning, and acquisition processes and policies that create the worst bottlenecks for deploying capability to the field at the speed of relevance. And we must address all the practices that not only put the U.S. Armed Forces at risk and reduce the efficiency of the DoD's operations, but also drive away the very people who are most needed to develop this critical capability.

Our adversaries are already doing this. China actively leverages its private industry to develop national security software (particularly in artificial intelligence, or AI), recruits top students under the age of 18[4] to work on "intelligent weapons design," and poaches U.S. software talent directly from the United States. In Russia, Vladimir Putin has told the students of his country that, "artificial intelligence is the future, not only for Russia, but for all humankind….Whoever becomes the

[3] Marc Andreessen, "Why Software Is Eating the World," *The Wall Street Journal*, August 20, 2011, 1.
[4]Stephen Chen, "China's Brightest Children Are Being Recruited To Develop AI 'Killer Bots,'" South China Morning Post, November 8, 2018,

leader in this sphere will become the ruler of the world." We can and must compete with software and the people who make it, not only to maintain U.S. military superiority, but also to ensure that the power that software represents is used in accordance with American values.

What this report is about. This report summarizes the current assessment of the Defense Innovation Board's (DIB) Software Acquisition and Practices (SWAP) study. The DIB was charged by Congress[5] to recommend changes to statutes, regulations, processes, and culture to enable the better use of software in DoD. We took an iterative approach, mirroring the way modern software is successfully done, releasing a sequence of concept papers describing our preliminary observations and insights (the latest versions of these are included in Appendix E). We used those to encourage dialogue with a wide variety of individuals and groups to gain insights into the current barriers to implementing modern software effectively and efficiently. This document captures key insights from these discussions in an easy-to-read format that highlights the elements that we think are critical for the DoD's success and serves as a starting point for continued discussions required to implement the changes that we recommend here.

This report is organized as follows:

- **Extended Abstract:** a two-page summary of the key takeaways from the report.
- **README** (this document): a more detailed executive summary of the report. If your boss heard about the report or read the extended abstract, thought it was intriguing, and asked you to read the entire report and provide a short summary, cut and paste this chapter and you should be good-to-go. (A README file is used by the open source software community to provide essential information about a software package.)
- **Recommendations Cheat Sheet:** A list of the primary lines of effort and key recommendations, so you can pretty much stop at that point—or better yet, stop after suggesting to your boss she adopt them all.
- **Chapters 1-4:** short descriptions of key areas and topics. If you attach the extended abstract to any one of these as a preface, it should be comprehensible.
- **Chapter 5:** a more detailed description of the recommendations and our rationale.
- **Supporting Information:** To ensure that the main body of the report satisfies the staple test[6] and the takeoff test,[7] we put most of the additional information generated during the study in a set of appendices. These provide a wealth of examples and evidence, but we took care to put our essential arguments up front for less wonky types. Some highlights:
  - **Draft implementation:** For each recommendation, additional information on the background, desired state, stakeholders, and actions to be taken
  - **Legislative language:** Template language for new or revised statutes, aligned with our recommendations
  - **FAQ** (frequently asked questions): a list of the most common questions that we get about the study and our attempt to answer them. (Question #1: hasn't all of this been recommended before? A: yes…)

[5] 2018 NDAA, Sec. 872. Defense Innovation Board analysis of software acquisition regulations.
[6] Any report that is going to be read should be thin enough to be stapled with a regular office stapler.
[7] Reports should be short enough to read during takeoff, before the movies start and drinks are served.

Note: if you are reading any portion of the report in paper form, a navigable version is available at http://innovation.defense.gov/software (hyperlink version coming soon).

Key themes. The rise of electronics, computing, and networking has forever transformed the way we live: software is a part of almost everything with which we interact in our daily lives, either directly through embedded computation in the objects around us or indirectly through the use of information technology through all stages of design, development, deployment, and operations. Our military advantage, coordination with allies and partners, operational security, and many other aspects of the DoD are all contingent upon our software edge and any lack thereof presents serious consequences. Software drives our military advantage: what makes weapons systems sophisticated is the software, not (just) the hardware.

Commercial trends show what is possible with software, from the use of open source tools to agile development techniques to global-scale cloud computing. Because of these changes, software can be developed, deployed, and updated much more quickly, which means systems need to be in place to support this speed. But modern software development requires a new set of skills and methodologies (e.g., generalist software engineers, specialized product management, DevOps and DevSecOps, agile development). Hence, the policies and systems surrounding software must be transformed to support software, not Cold-War era weapon manufacturing.

Our adversaries are active players in the world of software and so they will increasingly have the potential to develop weapons systems faster than we can, capitalizing on any advantages in software development. Meanwhile, they exploit our vulnerabilities via cyber-attacks to steal, undermine, and inhibit our capabilities. The incoming generation of military and civilian personnel began life digitally plugged-in, with an innate reliance on software-based systems. They will demand new concepts of operations, tactics, and strategies to maintain the edge they need. If the Department can refactor its acquisition processes and adjust its culture and personnel policies before it is too late, this software-savvy generation can still set the Department on the right course.

As we studied the methods that the private sector has used to enable software to transform its operations and consider how to best apply those practices to the defense enterprise, three primary themes emerged as the basis for our recommendations:

1. Speed and cycle time are the most important metrics for software.
2. Software is made by people and for people, so digital talent matters.
3. Software is different than hardware (and not all software is the same).

Speed and cycle time are the most important metrics for software. Most DoD software projects are currently managed using “waterfall” development processes, which involve spending years on developing requirements, taking and selecting bids from contractors, and then executing programs that must meet the listed requirements before they are “done.” This results in software that takes years to reach the field and is often not well matched to the current needs of the user or tactics of our adversaries, which have often changed significantly while the software was being written, tested, and accepted. Being able to develop and deploy faster than our adversaries means that we can provide more advanced capabilities, respond to our adversaries’ moves, and be more responsive to our end users. Faster reduces risk because it demands focus on the critical

functionality rather than over-specification or bloated requirements. It also means we can identify trouble earlier and take faster corrective action which reduces cost, time, and risk. Faster leads to increased reliability: the more quickly software/code is in the hands of users, the more quickly feedback can focus on efforts to deploy greater capability. Faster gives us a tactical advantage on the battlefield by allowing operation and response inside our adversaries' observe–orient–decide–act (OODA) loops. Faster is more secure. Faster is possible.

Software is made by people and for people, so digital talent matters. Current DoD human resource policies are not conducive to attracting, retaining, and promoting digital talent. Talented software developers and acquisition personnel with software experience are often put in jobs that do not allow them to make use of those talents, particularly in the military where rotating job assignments may not recognize and reward the importance of software development experience. As Steve Jobs observed,[8] one of the major differences between hardware and software is that for hardware the "dynamic range" (ratio between the best in class and average performance) is, at most, 2:1. But, the difference between the best software developer and an average software developer can be 50:1, or even 100:1, and putting great developers on a team with other great developers amplifies this effect. Today, in DoD and the industrial base that supports it, the people with the necessary skills exist, but instead of taking advantage of their skills we put them in environments where it is difficult for them to be effective. DoD does not take advantage of already existing military and civilian personnel expertise by offering pay bonuses, career paths that provide the ability to stay in their specialization, or access to early promotions. Skilled software engineers and the related specialties that are part of the overall software ecosystem need to be treated like a kind of special forces; the United States must harness their talent for the great benefits that it can provide.

Software is different than hardware (and not all software is the same). Over the years, Congress and DoD have developed a sophisticated set of statues, regulations, and instructions that govern the development, procurement, and sustainment of defense systems. This process was developed in the context of the Cold War, where major powers developed aircraft carriers, nuclear weapons, fighter jets, and submarines that are extremely expensive, last a very long time, and require tremendous access to capital and natural resources. Software, on the other hand, is something that can be mastered by a ragtag bunch of teenagers with very little money – and can be used to quickly destabilize world powers. Currently most parts of DoD develop, procure, and manage software like hardware, assuming that it is developed based on a fixed set of specifications, procured after it has been shown to comply with those specifications, "maintained" by block upgrades, and upgraded by replaying this entire procurement process linearly. But software development is fundamentally different than hardware development, and software should be developed, deployed, and continuously improved using much different cycle times, support infrastructure, and maintenance strategies. Testing and validation of software is also much different than for hardware, both in terms of the ability to automate but also in the potential vulnerabilities found in software that is not kept up to date. Software is never "done," and must be managed as an enduring capability that is treated differently than hardware.

[8] Steve Jobs, "Steve Jobs: The Lost Interview," interview by Robert X. Cringely for the 1995 PBS documentary, *Triumph of the Nerds*, released to limited theaters in, 2012, video.

Primary lines of effort: the most important things to do. DoD's current approach to software is a, if not *the*, major driver of cost and schedule overruns for Major Defense Acquisition Programs (MDAPs). Congress and DoD need to come together to fix the acquisition system for software because it is a primary source of its acquisition headaches.

Bringing about the type of change that is required to give DoD the software capabilities it needs is going to take a significant amount of work. While it is possible to use the current acquisition system and DoD process to develop, procure, assure, deploy, and continuously improve DoD software, the statutes, regulations, processes, and culture are debilitating. The current approach to acquisition was defined in a different era, for different purposes, and only works for software projects through enormous effort and creativity. Congress, the Office of the Secretary of Defense, the Armed Services, defense contractors, and the myriad of government and industry organizations involved in getting software out the door need to make major changes (together). Here are the four primary lines of effort that we recommend be undertaken:

A. **Refactor statutes, regulations, and processes for software,** enabling rapid deployment and continuous improvement of software to the field and providing increased insight to reduce the risk of slow, costly, and overgrown programs. The management and oversight of software development and acquisition must focus on different measures and adopt a quicker cadence.

B. **Create and maintain cross-program/cross-service digital infrastructure** that enables rapid deployment, scaling, testing, and optimization of software as an enduring capability; manage them using modern development methods; and eliminate the existing hardware-centric regulations and other barriers.

C. **Create new paths for digital talent (especially internal talent)** by establishing software development as a high-visibility, high-priority career track with specialized recruiting, education, promotion, organization, incentives, and salary.

D. **Change the practice of how software is procured and developed** by adopting modern software development approaches, prioritizing speed as the critical metric, ensuring cyber protection is an integrated element of the entire software lifecycle, and purchasing existing commercial software whenever possible.

None of these can be done by a single organization within the government. They are going to require a bunch of hard-working, well-meaning people to work together to craft a set of statutes, regulations, processes, and (most importantly) a culture that recognizes the importance of software, the need for speed and agility (theme 1), the critical role that smart people have to play in the process (theme 2), and the impact of inefficiencies of the current approach (theme 3). In many ways this mission is as challenging as any combat mission: while participant's lives may not be directly at risk in defining, implementing, and communicating the needed changes to policy and culture, the lives of those who defend our nation ultimately depend on the ability of the Department to redefine its approach to delivering combat-critical software to the field.

Refactor statutes, regulations, and processes, streamlined for software. Congress has created many workarounds to allow DoD to be agile in its development of new weapons systems, and DoD has used many of these to good effect. But the default statutes, regulations, and processes

that are used for software too often rely on the traditional hardware mentality (repeat: software is different than hardware) and those practices do not take advantage of what is possible with modern software (or frankly necessary, given the threat environment). We think that a combination of top-down and bottom-up pressure can break us out of the current state of affairs, and creating a new acquisition pathway that is tuned for software (of various types) will make a big difference. To this end, Congress and DoD should prototype and, after proving success, create mechanisms for ideation, appropriation, and deployment of software-driven solutions that take advantage of the unique features of software (versus hardware) development (start small, iterate quickly, terminate early) and provide purpose-fit methods of oversight. As an important aside, note that throughout this study our recommendations adhere to this guiding axiom—start small, iterate quickly—the same one that characterizes the best of modern software innovation cycles (see the "DIB Ten Commandments of Software" in Appendix E for more information about the DIB's guiding principles for software acquisition).

Create and maintain cross-program/cross-service digital infrastructure. Current practice in DoD programs is for each individual program to build its own infrastructure for computing, development, testing, and deployment, and there is little ability to build richer development and testing capabilities that are possible by making use of common infrastructure. Instead, we need to create, scale, and optimize an enterprise-level architecture and supporting infrastructure that enables creation and initial fielding of software within six months and continuous delivery of improvements on a three- month cycle. This "digital infrastructure," common in commercial IT, is critical to enable rapid deployment at the speed (and scale) of relevance. In order to implement this recommendation, Congress and DoD leadership must figure out ways to incent the Services and defense contractors to build on a common set of tools (instead of inventing their own) *without* just requiring that everyone use one DoD-wide (or even service-wide) platform. Similarly, OSD is going to have to define non-exceptions-based alternatives to (or at least pathways through) Joint Capabilities Integration and Development System (JCIDS), Planning, Programing, Budget and Execution (PPB&E), and Defense Federal Acquisition Regulation Supplement (DFARS)[9] that are optimized for software. The Director, Operational Test and Evaluation (DOT&E) will need new methods for operational test and evaluation that match the software's speed of relevance, and Cost Assessment and Program Evaluation (CAPE) is going to have to capture better data and leverage artificial intelligence/machine learning (AI/ML) as a tool for cost assessment and performance evaluation. Finally, the Services are going to need to identify, champion, and measure platform-based, software-intensive projects that increase software effectiveness, simplify interconnectivity among allies, and reform business practices. Subsequent chapters in our report provide specific recommendations on each of these areas.

Create new paths for digital talent (especially internal talent). The biggest enabler for great software is providing great people with the means to contribute to the national security mission. While the previous recommendations speak to providing the tools and infrastructure DoD technologists need to succeed, it is equally important that the Department's human capital strategies allow them to even do this work consistently in the first place. Driving the cultural transformation to support modern, cloud-based technology requires new types of skills and competencies, changing ratios

[9] Common DoD acronyms are defined in Appendix F (Acronyms and Catch Phrases).

of program managers to software engineers, moving from waterfall development to agile development, and dealing with all of the change management that comes with it. This is not an easy task, but arguably one of the most important. While compensation is a major driver in attracting competitive talent, DoD must also make changes in the roles, methodologies, cultures, and other aspects of the transformation that industry is undergoing and that the government must as well.

Increasing developer talent is not the only workforce challenge. DoD must also change how programs and contractors are managed, which goes beyond just moving to agile development. The government must have experts well steeped in the software development process and architecture design to adequately manage both organic activities and contracted programs. They must have the skills to detect when contractors are going down the wrong path, choosing a bad implementation approach, or otherwise being wasteful. This is perhaps the argument for ensuring we have software development experience natively in the government, rather than relying primarily on external vendors; unless there are software-knowledgeable members on the core team, it is impossible to effectively monitor and manage outsourced projects. This is even truer with the movement to DevSecOps.

In implementing this change in the workforce, it is particularly important to provide new career paths for digital talent and enable the infrastructure and environment required to allow them to succeed. The current GS system favors time-in-grade over talent. This simply will not work for software. The military promotion system has the same problem. As with sports, great teams make a huge difference in software and we need to make sure those teams have the tools they need to succeed and reward them appropriately -- through recognition, opportunities for impact, career advancement, and pay. Advanced expertise in procurement, project management, evaluation and testing, and risk mitigation strategies will also be needed to create the types of elite teams that are necessary. A key element of success is finding ways to keep talented people in their roles (rather than transferring them out because it is the end of their assignment), and promote people based on their abilities, not based on their years of service.

Change the practice of how software is procured and developed. The items above are where we think Congress and the Department should focus in terms of statutory, regulatory, and process changes. But a major element is also the need to change the *culture* around software within Congress, DoD, and the defense industrial base. We use the term “DevSecOps” as our label for the type of culture that is needed: iterative development that deploys secure applications and software into operations in a continuing (and continuous) fashion.

Numerous projects and groups have demonstrated the ability to implement DevSecOps within the existing acquisition system. But the organizations we previously mentioned - DDS, JIDO, DIU, and Kessel Run - are the exception rather than the rule, and the amount of effort required to initiate and sustain their activities is enormous. Instead, DoD must make legacy programs that use outdated techniques for developing software fight for existence (and in most cases replace them with new activities that embrace a DevSecOps approach).

Getting started now. The types of changes that we are talking about will take years to bring to complete fruition. But it would be a mistake to spend two years figuring out what the answer should look like, spend another two years prototyping the solutions to make sure we are right,

then spend two to four more years implementing the changes in statutes, regulations, processes, and culture that are actually required. Let's call that approach the "hardware" approach. Software is different than hardware and therefore, the approach to implementing change for software should be different as well.

Indeed, most (if not all) of the changes we are recommending are not new and not impossible to do. The 1987 Defense Science Board Task Force on Military Software,[10] chaired by legendary computer scientist Fred Brooks, wrote an outstanding report that already articulated much of what we are saying here. And the software industry has already implemented and demonstrated the utility of the types of changes we envision. The problem appears to be in getting the military enterprise to adopt a software mindset and implement a DevSecOps approach in a system that was intended to make sure that things would not move too quickly.

Many of our DoD issues could be addressed by adopting existing best practices of the private sector for agile development, software as a service, use of modern (cloud) infrastructure, tools, computing and shared libraries, and software logistics and support delivery systems for software maintenance, development, and updating (patching). We do not need to study these, we need to get going and implement them. Here are some specific suggestions for what to do starting *now*:

- FY19 (create): High-level endorsement of the vision we articulate here, and support for activities that are consistent with the desired end state (i.e., DevSecOps and enterprise-level architecture and infrastructure); identify and launch programs to move out on the priority recommendations (start small, iterate quickly). If you are reading this and are in a position of leadership in your organization, pass this on to others with your seal of approval and a request for your team to develop 2-3 plans of action for how it can be applied in your domain. If someone comes to you with a proposal that aligns with the objectives we have outlined here, find a way to be on the front line of changing DoD to a "culture of yes."
- FY20 (deploy): Initial deployment of authorities, budgets, and processes for software acquisition and practices reform. Execute representative programs according to the lines of effort and recommendations in this report, implement now, measure results, and modify approaches. Implement this report in the way we implement modern software.
- FY21 (scale): Streamlined authorities, budgets, and processes enabling software acquisition and practices reform at scale. In this time frame, we need a new methodology to estimate as well as determine the value of software capability delivered (and not based on lines of code).
- FY22 (optimize): All DoD software development projects transition (by choice) to software-enabled processes, with talent and ecosystem in place for effective management and insight.

In the remainder of this report we provide a rationale for the approach that we are advocating. Chapter 1 makes the case for why software is important to DoD, including a taxonomy of the different types of software that need to be considered (not all software is the same). In Chapter 2, we describe how software is developed in the private sector and what is required in terms of workforce, infrastructure, and culture. Chapter 3 is an attempt to understand what has already

[10] Defense Science Board Task Force, *Military Software* (Washington, DC: Office of the Under Secretary of Defense for Acquisition, September 1987), https://apps.dtic.mil/dtic/tr/fulltext/u2/a188561.pdf.

been said by other studies and groups, why the situation has not changed, and how we think this study can potentially lead to a different outcome. Chapters 4 and 5 contain our recommendations for how to move forward. In Chapter 4 we present three alternative paths to consider: doing the best we can with the current system, streamlining statutes, regulations, and processes so that they are optimized for software (instead of hardware), and making more radical changes that create whole new appropriation categories and acquisition pathways. Finally, Chapter 5 describes the path that we recommend be taken, broken out along the lines of effort described above, and with a set of 10 key recommendations (a detailed set of action plans for implementing those recommendations is included in Appendix A).

A two page summary (“cheat sheet”) of the lines of effort and recommendations are given next.

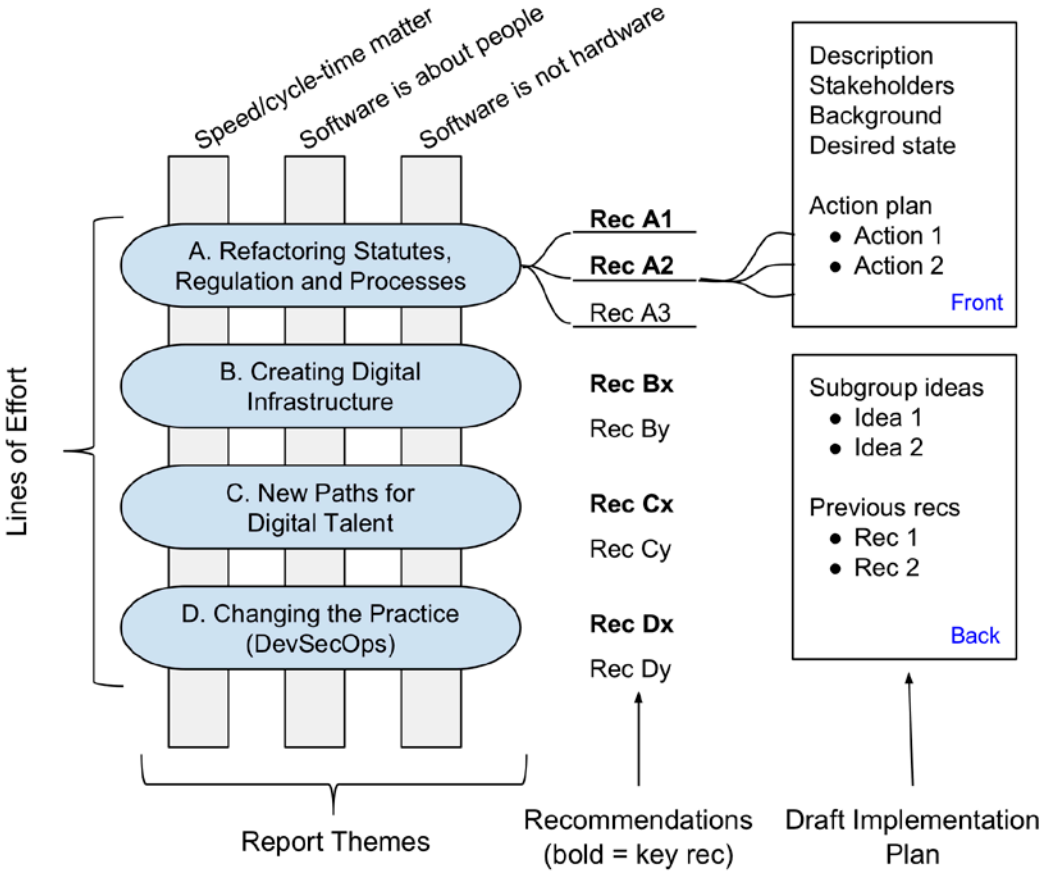
//PAGE INTENTIONALLY LEFT BLANK//

DIB SWAP Study
Recommendations "Cheat Sheet"
v2.1, 12 Mar 2019

This sheet contains a list of the recommendations for the Defense Innovation Board's (DIB) Software Acquisition and Practices (SWAP) study. The recommendations below include input from the following sources:

- DIB Guides for Software (Appendix E)
- SWAP working group reports (Appendix F)
- Previous software acquisition reform studies (starting with the 1987 DSB study)

The recommendations are organized according to four major lines of effort and each recommendation contains background information, a proposed owner for implementing the recommendation as well as a more detailed action plan, a list of other offices that are affected, and additional details. The following diagram documents this structure:



For each recommendation, a draft implementation plan can be found in Appendix A that gives more detail on the rationale, supporting information, similar recommendations, specific action items, and notes on implementation. Potential legislative language to implement selected recommendations is included in Appendix B.[11]

[11] Appendix B is not yet finalized or all-inclusive

The Ten Most Important Things to Do (Starting Now!)

Line of Effort A (Congress and OSD): Refactor statutes, regulations, and processes for software	
A1	Establish a new acquisition pathway (Sec 805) for software that prioritizes continuous integration and delivery of working software in a secure manner, with continuous oversight from automated analytics.
A2	Create a new appropriations category that allows (relevant types of) software to be funded as a single budget item, with no separation between RDT&E, production, and sustainment.
Line of Effort B (OSD and Services): Create and maintain cross-program/cross-service digital infrastructure	
B1	Establish and maintain digital infrastructure within each Service or Agency that enables rapid deployment of secure software to the field and incentivize its use by contractors
B2	Create, implement, support, and use fully automatable approaches to testing and evaluation (T&E), including security, that allow high confidence distribution of software to the field on an iterative basis
B3	Create a mechanism for Authority to Operate (ATO) reciprocity within and between programs, Services, and other DoD agencies to enable sharing of software platforms, components and infrastructure and rapid integration of capabilities across (hardware) platforms, (weapons) systems, and Services
Line of Effort C (Services): Create new paths for digital talent (especially *internal* talent)	
C1	Create software development groups in each Service consisting of military and/or civilian personnel who write code that is used in the field and track individuals who serve in these groups for future DoD leadership roles
C2	Expand the use of (specialized) training programs for CIOs, SAEs, PEOs, and PMs that provide (hands-on) insight into modern software development (e.g., agile, DevOps, DevSecOps) and the authorities available to enable rapid acquisition of software
Line of Effort D (Acquisition Offices and Contractors): Change the practice of how software is procured and developed	
D1	Require access to source code, software frameworks, and development toolchains – with appropriate IP rights – for all DoD-specific code, enabling full security testing and rebuilding of binaries from source
D2	Make security a first-order consideration for all software-intensive systems, under the assumption that security-at-the-border will not be enough.
D3	Shift from the use of rigid lists of requirements for software programs to a list of desired features and required interfaces/characteristics, to avoid requirements creep, overly ambitious requirements, and program delays

Additional context provided in Chapter 5 and draft implementation plans in Appendix A.

Chapter 1. Not All Software is the Same, but it all Matters for National Security

This chapter provides a high-level vision of why software is critical for national security and the types of software we are going to have to build in the future. We also provide a description of different types of software, where they are used, and why a one-size-fits-all approach will not work.

1.1 Where Are We Coming from, Where Are We Going?

While software development has always been a challenge for the Department, today these challenges are greatly affecting our ability to deploy and maintain mission critical systems to meet current and future threats. In the past, software simply served as an enabler of hardware systems and weapons platforms.

Software now defines our mission critical capabilities and our ability to sense, share, integrate, coordinate, and act. Software is everywhere and is in almost everything that the Department operates and uses. Software drives our weapons systems; command, control, and communications systems; intelligence systems; logistics; and infrastructure and it drives much of the backroom, enterprise process that make the Department function. If the new domain that we are fighting in is cyber, then our ability to maintain situational awareness and our ability to fight, defend, and counter threats will be based on the capabilities of our software. In this new domain software is both the enabler as well as the target of the fight.

As our military systems become increasingly networked and automated, as autonomy becomes more prevalent, as we become more dependent on machine learning and artificial intelligence, then our ability to maintain superiority will be directly linked to our ability to field and maintain software that is better, smarter, and more capable than our adversaries software. In this new world, digital threats are more prevalent and, in many cases, more effective than physical and kinetic threats alone. Digital capabilities bring new dimensions to asymmetric and hybrid warfare and nation states are investing in new capabilities to gain parity, if not superiority, over the United States. Even our ability to defend against new physical and kinetic threats like hypersonics, energetics, and biological weapons will be based on software capabilities. We need to identify and respond to these new threats as they happen in near real time. Our ability to do so will be based on our ability to develop and push new software defined capabilities to meet those threats on time scales that greatly outpace our adversaries' ability to do so.

The ability to meet future threats requires us to rethink how we develop, procure, assure, deploy, and maintain software. We can no longer take years to develop software for our major systems. Software cannot be an afterthought to hardware and it cannot be acquired, developed, and managed like hardware. DoD's current procurement processes treat software programs like hardware programs. Its acquisition and development approaches are also antiquated and do not meet the demands of the Department. Fixing our software approach in the Department is more than just making sure that we get control over cost and budget, it is about our ability to maintain our fighting readiness and our ability to win the fight and counter any threat regardless of domain and regardless of adversary.

1.2 Weapons and Software and Systems, Oh My! A Taxonomy for DoD

Not all software systems are the same and it is important to optimize development processes and oversight mechanisms to the different types of software that are used by DoD. We distinguish here between two different aspects of software: their *operational function* (use) and their *implementation platform*. To a large extent, a given operational function can be implemented on many different computational platforms depending on whether it is a mission support function (where high bandwidth connectivity to the cloud is highly likely) or a field-forward software application (where connectivity many be compromised and/or undesirable).

The following glossary of terms provides some characteristics and important properties of the types of software-enabled systems that deliver the DoD's mission:

- *Enterprise systems*: very large-scale software systems intended to manage a large collection of users, interface with many other systems, and generally used at the DoD level or equivalent. These systems should always run in the cloud and should use architectures that allow interoperability, expandability, and reliability. In most cases the software should be commercial software purchased without modification to the underlying code, but with DoD-specific configuration. Examples include: e-mail systems, accounting systems, travel systems, and HR databases.

- *Business systems*: essentially the same as enterprise systems, but operating at a slightly smaller scale (e.g., for one of the Services). Like enterprise systems, they are interoperable, expandable, reliable, and probably based on commercial offerings. Similar functions may be customized differently by individual Services, though they should all interoperate with DoD-wide enterprise systems. Examples include: software development environments, Service-specific HR, financial, and logistics systems.

- *Combat systems*: software applications that are unique to the national security space and used as part of combat operations. Combat systems may require some level of customization that may be unique to DoD, not the least of which will be specialized cybersecurity considerations to enable them to continue to function during an adversarial attack. (Note that since modern DoD enterprise and business systems depend on software, cyber-attacks to disrupt operations have the potential be just as crippling as those aimed at combat systems.)

  We further break down combat systems into subcategories:

  - *Logistics systems*: any system that is used to keep track of materials, supplies, and transport as part of operational use (versus Service-scale logistics systems, with which they should interoperate). While used actively during operations, logistics systems are likely to run on commercial hardware and operating systems, allowing them to build on commercial-off-the-shelf (COTS) technologies. Platform-based architectures enable integration of new capabilities functions over time (probably on a months-long or annual time scale). Operation in the cloud or based on servers is likely.

- *Mission systems*: any system used to plan and monitor ongoing operations. Similar to logistics systems, this software will typically use commercial hardware and operating systems, but may be run in a more localized form (such as an air operations center) that precludes the use of some types of cloud computing infrastructure, but may still heavily leverage cloud technologies, at least in terms of critical functions. These systems should be able to incorporate new functionality at a rate that is set by the speed at which the operational environment changes (days to months).

- *Weapons system*: any system that is capable of the delivery of lethal force, as well as any direct support systems used as part of the operation of the weapon. Note that our definition differs from the standard DoD definition[12] of a weapons system, which also includes any related equipment, materials, services, personnel, and means of delivery and deployment (if applicable) required for self-sufficiency. The DoD definition would most likely include the mission and logistics functions, which we find useful to break out separately. Software on weapons systems is traditionally closely tied to hardware, but as we move to greater reliability of software-defined systems and distributed intelligence, weapons systems software is becoming increasingly hardware independent (similar to operating systems for mobile devices, which run across many different hardware configuration).

We also define several different types of computing platforms on which the operational functions above might be implemented:

- *Cloud computing*: computing that is typically provided in a manner such that the specific location of the compute hardware is not relevant (and may change over time). These systems will typically be running on commercial hardware and using commercial operating systems, and the applications running on them will run even as the underlying hardware changes. The important point here is that the hardware and operating systems are generally transparent to the application and its user.

- *Client/server computing*: computing provided by a combination of hardware resources available in a computing center (servers) as well as local computing (client). These systems will usually be running on commercial hardware and using commercial operating systems.

- *Desktop/laptop/tablet computing*: computing that is carried out on a single system, often by interacting with data sources across a network. These systems will usually be running on commercial hardware and using commercial operating systems.

- *Embedded computing*: computing that is tied to a physical, often-customized hardware platform and that has special features that requires careful integration between software and hardware.

Note that a single software system may have multiple components or functions that cross these definitions and there may be components of an integrated system that have elements that cross

[12] The Department of Defense, *DoD Dictionary of Military and Associated Terms* (Washington, DC: Department of Defense, as of February 2019), 252.

these definitions. The key point is that each type of software system will have different requirements in terms of how quickly it can/should be updated, the level of information assurance that is required, and the organizations that will participate in development, testing, customization, and use of the software. Different statutes, regulations, and processes may be required for different types of software (and these will differ greatly from what is used for hardware).

Having defined systems that deliver effects and the kinds of computing platforms on which software is hosted, we now distinguish between four primary types of software, which we use throughout the rest of the report so they we differentiate the approaches that are needed:

- **Type A (Commercial-Off-The-Shelf (COTS) apps):** The first class of software consists of applications that are available from commercial suppliers. Business processes, financial management, human resources, software development and collaboration tools; accounting and other "enterprise" applications in DoD are generally not more complicated nor significantly larger in scale than those in the private sector. Unmodified commercial software should be deployed in nearly all circumstances. Where DoD processes are not amenable to this approach, those processes should be modified, not the software.

- **Type B (Customized Software):** The second class of software constitutes those applications that consist of commercially available software that is customized for DoD-specific usage. Customizations can include the use of configuration files, parameter values, or scripted functions that are tailored for DoD missions. These applications will generally require (ongoing) configuration by DoD personnel, contractors, or vendors.

- **Type C (COTS Hardware/Operating Systems):** The third class of software applications is those that are highly specialized for DoD operations but run on commercial hardware and standard operating systems (e.g., Linux or Windows). These applications will generally be able to take advantage of commercial processes for software development and deployment, including the use of open-source code and tools. This class of software includes applications that are written by DoD personnel as well as those that are developed by contractors.

- **Type D (Custom SW/HW):** This class of software focuses on applications involving real-time, mission-critical, embedded software whose design is highly coupled to its customized hardware. Examples include primary avionics or engine control, or target tracking in shipboard radar systems. Requirements such as safety, target discrimination, and fundamental timing considerations demand that extensive formal analysis, test, validation, and verification activities be carried out in virtual and "iron bird" environments before deployment to active systems. These considerations also warrant care in the way application programming interfaces (APIs) are potentially presented to third parties.

We note that these classes of software are closely related to those described in the 1987 DSB study on military software, where they categorized software as "standard" (roughly capturing types A and B), "extended" (type C), "embedded" (type D), and "advanced" (which they categorized as "advanced and exploratory systems," which are not so relevant here).

1.3 What Kind of Software Are We Going To Have To Build?

The competitor that can realize software-defined military capability the fastest wins all future conflicts. We must shorten our development cycles from years to months so that we can react and respond within the observe-orient-decide-act (OODA) loop of the threats we face. Agile methodologies enable (see "Detecting Agile BS" in Appendix E for more information about agile methodologies) this rapid cycle approach and in addition to development we will need to test and validate software in real-time as part of the integrated approach agile demands. Quality assurance needs to be a continuous and fully integrated process throughout every phase of the software cycle. We need to build software logistic trains that are able to develop and deploy software and provide updates as quickly as modern day commercial companies so that we can respond to new threats (especially when the target will be our software). We must treat software as a continuous service rather than as block deliverables. It is important to have the agility in our procurement approach that will allow program managers to change priorities based on the needs and timing of the end users.

In the near feature, the DoD's acquisition and use of business systems should closely mirror industry and the private sector. The DoD should modify its processes to mimic industry's best practices (see Section 2.1 for examples of best practices in industry) rather than try to contract and maintain customized software.

DoD should also adopt commercial logistics and mission planning software (COTS) wherever possible and reduce its reliance on government-off-the-shelf (GOTS) solutions. Good logistics and mission software reduces process complexity, improves situational awareness, reduces costs, simplifies planning while improving speed of delivery and streamlines performance.

Software defined systems should be easier to develop, maintain, and upgrade than classic embedded systems. A well-designed system would allow new capabilities that can be delivered directly to edges of the network from the cloud in the same way new capabilities are delivered to consumer smart devices.

DoD should manage software by measuring value delivered to the customer rather than by monitoring compliance with requirements. Accountability should be for delivering value to the customer and solving customer needs, not by complying with obsolete contracts and requirements documents.

Program managers must identify potential problems earlier (ideally, within months) and take corrective action quickly. Troubled programs need to fail quickly, and we need to learn from them. As we witnessed throughout our work on this study, many software programs are too big, too complex, too long, and have too many requirements. Development needs to be staged and follow the best practice of smaller, quicker deliverables with higher frequency of updates and new features. Initially, program development should focus on developing the minimum viable product delivered more quickly to the customer than traditionally run programs.

Software developers within our defense community need the same modern tools, systems, environments, and collaboration resources that commercial industry has adopted as standard. Without this, we are undermining the effectiveness of our software developer base, and our ability to attract and retain our software human capital, both within DoD and among our suppliers. With the introduction of new technologies like machine learning and artificial intelligence and the ever-

increasing interdependency between networked heterogeneous systems, software complexity will continue to increase logarithmically. We need to continuously invest in new development tools and environments including simulation environments, modeling, automated testing and validation tools. We must invest in research and development into new technologies and methodologies for software development to help the Department keep up with ever growing complexity of defense systems.

1.4 What Are the Challenges That We Face (and Consequences of Inaction)?

The world is changing. The United States used to be the dominate supplier of software and the world leader in software innovation. That is no longer the case. Due to the global digital revolution driven by the consumer and commercial markets, countries are building their own indigenous software capabilities and their own technology clusters. Countries like China are making huge investments in AI and cyber. China's 2030 plan envisions a $1 trillion dollar artificial intelligence industry in China. They want to become a cyber superpower and are investing in their capital markets, universities, research centers, defense industry, and commercial software companies.

The long-term consequence of inaction is that our adversaries' software capabilities can catch and surpass ours. If that happens, then our adversaries will be able to develop new capabilities and iterate faster than we can. They can respond to our defense systems faster than we can respond to theirs. If their algorithms and AI becomes superior to ours, it means that they can hold a decisive advantage where any of our systems goes up against any of theirs. And if their cyber capability becomes superior to ours, they can shut us down, cause chaos, and continue to steal our secrets at their choosing and without repercussion – especially if we cannot attribute those attacks. Our adversaries' software capabilities are growing rapidly. If we do keep pace, we could lose our defense technology advantage within a decade or much sooner.

Chapter 2. What Does It Look Like to Do Software Right?

In many cases, the software acquisition approaches and practices in place within DoD today look strange and perplexing to those familiar with commercial software practices. While the mission-, security-, and safety-critical nature of DoD's software in the context of embedded weapons will have an impact on practices, the extreme degree of divergence from contemporary commercial practice has been an area of focus. Our case studies, site visits, and other study activities allowed a closer look into the reasons for divergence and whether the absence of many commercial best practices is justified.

2.1 How It Works in Industry (and Can/Should Work in DoD): DevSecOps

Modern software companies must develop and deliver software quickly and efficiently in order to survive in a hyper-competitive environment. While it is difficult to characterize the entire software industry, the following set of practices—based on documented approaches at Google[13]—are representative of commercial environments where the delivery of software capability determines the commercial success or failure of the company. These practices generally hold true in other industries where companies have unexpectedly found themselves in the software business due to an increasing reliance on software to provide their key offerings – e.g. automotive, banking, healthcare, and many others. In any environment, software engineering practices must be matched with the recruitment and retention of talented software expertise. These practices must be honed over time and adapted to lessons learned.

Generally, successful software companies have developed best practices in three categories:

Software development. These are software engineering practices that include source code management, software build, code review, testing, bug tracking, release, launch and post-mortems. Some of the key best practices that are applicable to DoD software programs include:

- All source code is maintained in a single repository that is available to all software engineers. There are control mechanisms to manage additions to the repository but in some cases all engineers are culturally encouraged to fix problems, independent of program boundaries.
- Developers are strongly encouraged to avoid "forking" source code and focus work on the main branch of the software development.
- Code review tools are reliable and easy to use. Changes to main source code typically require review by at least one other engineer and code review discussions are open and collaborative.
- Unit test is ubiquitous, fully automated, and integrated into the software review process. Integration, regression, and load testing are also widely used and these activities should be an integrated automated part of daily workflow.
- Releases are frequent - often weekly. There is an incremental staging process over several days, particularly for high-traffic, high-reliability services.

[13] Fergus Henderson, "Software Engineering at Google" (arXiv:1702.01715 [cs.SE], January 31, 2017).

- Post-mortems are conducted after system outages. The focus of the post-mortem is on how to avoid problems in the future and not about affixing blame.

Project management. Software projects must contribute to the overall aim of the business and efforts must be aligned to that end goal.

- Individuals and teams set goals, quarterly and annually. Progress against those goals are tracked, reported, and shared across the organization. Goals are mechanisms to encourage high performance but can be decoupled from performance appraisal or compensation.
- Organic project approval process. Significant latitude to initiate projects is given at all levels, with oversight responsibility given to managers and executives to allocate resources or cancel projects.

People management. Given the scarce number of skilled software engineers, successful software companies know how to encourage and reward good talent. Some examples include:

- Clear separation between engineering and management roles, with advancement paths for both. Similar distinctions are made between technical management and people management. The ratio of software engineers to product managers and program managers ranges from 4:1 to 30:1.
- Mobility throughout the organization is encouraged. This allows for the spread of technology, knowledge, and culture throughout the company.

In addition to these specific software development practices, another common approach to managing programs in industry is to move away from the typical DoD specifications and requirements approach towards a portfolio management approach. The portfolio management approach allows program managers to make agile decisions based on evolving needs and capabilities. Using a portfolio management approach, a program manager has a list of features and capabilities ranked by need, risk, cost, resource, and time. This list of capabilities is two to three times larger than what generally can be accomplished within a given time frame, a given budget, and a set of resources. Program managers make decisions about feature mix, matching investments to needs, and balancing risk against performance. Needs are driven tactically by end users and strategically by the Services. Capabilities are tested and delivered on a continuous basis, and maximum automation is leveraged for testing.

In industry, software programs initially start as a minimum viable product (MVP). A minimum viable product has just enough features to meet basic minimum functionality. It provides the foundational capabilities upon which improvements can be made. MVPs have significantly shorter development cycles than traditional waterfall approaches. The goal of MVPs is to get basic capabilities into users hands for evaluation and feedback. Program managers use the evaluation and feedback results to rebalance and re-prioritize the software capability portfolio.

Portfolio success is measured based on performance of the *delivery* of capabilities as measured against users' need and strategic objectives within an investment cycle. Value is determined by output measurements rather than process measurements. Portfolio value is the aggregation of total value of all of the capabilities delivered divided by total cost invested within a period of time.

Blending higher risk/higher reward capabilities with lower risk/lower reward capabilities is the art of good portfolio management. Within a given period of time, program managers will use diversification to spread risk and rewards. Good program managers identify troubled projects early and are encouraged either to quickly correct the problems or to quickly abandon failing efforts so that remaining resources can be husbanded and then reallocated to other priorities.

Software budgets are driven by time, talent, compute resources, development environment, and testing capabilities required to deliver capabilities. The capability and cost of talent varies greatly between software engineers, designers, programmers, and manager. The quality of engineering talent is the single largest variable that determines cost, risk, and time of a software project. Good portfolio managers must take inventory of the range of software talent within a program and carefully allocate that talent across the portfolio of capabilities development.

2.2 Empowering the Workforce: Building Talent Inside and Out

One of the biggest barriers to the software capabilities the Department so desperately needs is how the Department manages the people necessary to build that capability. DoD cannot compete and dominate in defense software without a technical and design workforce within the Department that can both build software natively and effectively manage vendors to do the same, using the proven principles and practices described above. Some of the Department's human capital practices actively work against this critical goal.

If the Department wants to be good at software, it must be good at recruiting, retaining, leveraging, managing, and developing the people who make it. When we look at private sector organizations and institutions that effectively use software to fulfill their mission, each of them:

- Understands the software professionals that it has, understands at a high level what it needs, and understands the gap between the two; we say "at a high level" because the DIB believes that the gap is large enough that it is much more important to begin closing the gap than it is to measure the gap to too much precision;
- Has a strategy to recruit the people and skills it needs to fulfill its mission, understanding what it uniquely has to offer in a competitive market;
- Has clear understanding of the competencies required by software professionals in the organization and the expectations of these professionals at each level in the organization;
- Has defined career ladders for both uniformed (via the military occupational specialty code or its equivalent) and civilians (via the GS system) that map software competencies and expectations from entry level to senior technical leadership and management;
- Offers opportunities for learning and mentorship from more senior engineering and design leaders;
- Counts engineering and design leaders among its most senior leadership, with the ability to advocate across silos for the needs of the software and software acquisition workforce and support other senior leaders in understanding how to work with both;
- Supports a cadre of leadership able and empowered to create a culture of software management and promote common approaches, practices, platforms and tools, while retaining the ability to use judgement about when to deviate from those common approaches and tools;

- Is able to reward software professionals based on merit and demonstrated contribution rather than time in grade.

The above are not descriptors for the software workforce in today's DoD.

The DoD has, however, long recognized that medicine and law require specialized skills, continuing education, and support and made it not only possible but desirable and rewarding to have a career as a doctor or lawyer in the armed forces. In contrast, software developers, designers, and managers in the services must practice their skills intermittently and often without support as they endure frequent rotations into other roles. We would not expect a trained physician to constantly rotate into deployments focused on aviation maintenance, nor would we interrupt the training of a lawyer to teach her human resources. Who would be comfortable being treated by a physician who worked in an institution that lacked common standards of care and provided no continuing education? And though software is often a matter of life and death, the DoD's current human capital practices do all of these.

The process to retool human capital practices to meet the challenge of software competency in the DoD must start with the people the DoD already has who have software skills or who are interested in acquiring them. Unlike medicine, software skills can be acquired through self-directed and even informal training resources, and the Department has individuals, military and civilian, who have taken it upon themselves to gain technical skills outside of or in addition to formal DoD training. This kind of initiative and aptitude, especially when it results in real contribution to the mission, should be rewarded with appropriate career opportunities for advancement in this highly sought-after specialty. As we have witnessed during site visits for this study, there are also many individuals with more formally recognized software skills who are working with determination and even courage to try to deliver great software in service of the mission, but whose efforts to practice modern software techniques are poorly supported, and often actively blocked. Changes to policy that make clear the Department's support for these practices will help, but they must be married with support for the individuals to stay and grow within their chosen field. Possible human capital pathways might include:

- A core Occupational Series (Civilian) for software development that includes subcategories to address the various duties found in modern software development (e.g., developers/engineers, product owners, designers, etc.)
- A secondary specialty series/designator for military members for software development. Experts come from various backgrounds and a special secondary designator or occupational series for service members would be invaluable to tapping into their expertise even if they are not part of the core "Information Technology" profession.
- A Special Experience Identifier or other Endorsement for acquisition professionals (military and civilian) that indicates they have the necessary experience and training to serve on a software acquisition team. This Identifier or Endorsement needs to be a mandatory requirement to lead the acquisition team for any software procurement. Furthermore, this Identifier or Endorsement needs to be expanded to the broader team working the software procurement to include legal counsel, contract specialists, and financial analysts.

In addition to supporting the people DoD has today, both those already working in software and those who could, the DoD will need to attract and retain many more, and more qualified, software developers and, particularly, more software leaders. Again, the creation of defined career ladders that recognize and reward the appropriate competencies for each of the major specialties on a software team is table stakes for effective recruitment. Also effective will be the demonstrated ability to leverage, recognize, and reward software developers more flexibly than DoD currently allows for so that the strongest contributors can be put on the most critical projects and can be retained within DoD even when their skills become highly valued in the private sector. In addition, our recommendations contain over a dozen ways that DoD can improve its technical recruiting, including the idea of giving all new recruits a software aptitude test to identify potential trainees.

2.3 Getting It Right: Superior National Security AND Better Oversight

Getting software right in DoD is not as simple as recognizing that it is a national security priority, and changing the practices of the development teams is also insufficient. Oversight (and budgeting and finance) must also change.

Agile projects that use modern software approaches can be expected to deliver value to the user faster than alternative approaches. Oversight of monolithic, waterfall projects has generally focused around whether the team hit pre-determined milestones that may or may not represent actual value or even working code, and trying to figure out what to do when they do not. When evaluating and appropriating funds to agile projects, it is more suitable to judge the project on the speed by which it delivers working code and actual value to users. In a waterfall project, changes to the plan generally reflect the team falling behind and are cause for concern. In a project that is agile and takes advantages of the other approaches the DIB recommends (including software reuse), the plan is intended to be flexible because the team should be learning what works as they code and test. Successful projects will develop metrics that measure value to the user, which involves close, ongoing communication with users. Source Lines of Code (SLOC) does not equal value. (SLOC ≠ value).

Have a leader and hold them accountable. Great program outcomes generally emerge from exceptional leaders who are fully on the hook for delivering on their vision. The mythology around the impact of top founders is widely and commonly accepted with regards to private companies.

This is just as applicable to the public sector as it is to the private sector and has become somewhat of a lost art form. Many of the most noteworthy defense programs over the past decade have been shepherded by exceptional “founders.” Kelly Johnson with the U-2, F-104, SR-71. Paul Kaminski with stealth technology. Admiral Hyman Rickover with the nuclear navy. Harry Hillaker with the F-16. Bennie Schriever with the intercontinental ballistic missile. The list goes on. The United States Digital Service recognized this with Play 6 of the Digital Services Playbook[14] - Assign One Leader and Hold That Person Accountable. We would do well to remember this part of our history and work this into our oversight plan.

[14] https://playbook.cio.gov/#plays_index_anchor

Speed increases security. As we have learned from the cyber world, when we are facing active threats, our ability to have faster detection, response, and mitigation reduces the consequences of an attack or breach. In the digital domain, where attacks can be launched at machine speeds, where Artificial Intelligence (AI) and Machine Learning (ML) can probe and exploit vulnerabilities in near real-time, our current ability to detect, respond, and mitigate against digital threat leaves our systems completely vulnerable to our adversaries.

> The Department of Defense (DOD) faces mounting challenges in protecting its weapon systems from increasingly sophisticated cyber threats. This state is due to the computerized nature of weapon systems; DOD's late start in prioritizing weapon systems cybersecurity; and DOD's nascent understanding of how to develop more secure weapon systems. DOD weapon systems are more software dependent and more networked than ever before…. Potential adversaries have developed advanced cyber-espionage and cyber-attack capabilities that target DOD systems.
>
> GAO-19-128: Published: Oct 9, 2018. Publicly Released: Oct 9, 2018

The DoD must operate within our adversaries' digital OODA loop. Much like today's consumer electronic companies, the DoD needs the ability to identify and mitigate evolving software and digital threats and to push continuous updates to fielded systems in near real-time.

We must be able to do so without sacrificing our abilities to test and validate software. To accomplish this, we need to re-imagine the software development cycle as a continuous flow rather than discrete software block upgrades. We need to not only modernize to the agile methodology of software development, but we must also modernize our entire suite of development and testing tools and environments. We need to be able to instrument our fielded systems so that we can build accurate synthetic models that can be used in development and test. The Department needs to be able to patch, update, enhance, and add new capabilities faster than our adversaries' abilities to exploit vulnerabilities.

2.4 What's the R&D Strategy for Our Investment?

The nature of software development may radically change in the near future. It is incumbent that the Department of Defense adequately fund R&D programs to advance the fields of computer science including: computer programming, artificial intelligence/machine learning, autonomy, quantum computing, networks and complex systems, man-machine interfaces and cybersecurity.

Today, computers are controlled by programs that are comprised of sets of instructions and rules that human programmers write. AI and ML changes how humans teach computers. Instead of providing computers with programmed instructions, humans will train or supervise the learning algorithm being executed on the computer. Training is inherently different than programming. Data becomes more important than code. Training errors are very different than programming errors. Hacking AI is very different than hacking code. The use of synthetic environments and digital twins may also become increasingly important tools to train a computer. The impact of AI

and ML on software development will be profound and necessitates entirely new approaches and methods of developing software.

Computer architectures are also evolving. Networks of distributed dissimilar computing and sensors greatly increases complexity which could significantly impact software and system reliability. This is especially important given the proliferation of sensors and the importance of networked and cloud enabled systems in modern warfare. The Department will need new tools and approaches to testing and validation.

New computing technologies are also on the horizon. Experts may agree that we are many years away from developing a Universal Quantum Computer (UQC), nevertheless, the United States cannot afford to come in second in the race to develop the first UQC. The challenge is not only confined to the development of the UQC hardware but in developing QC programming languages and software. We also need to continue to invest in new quantum resistant technologies such as cryptography and algorithms and apply those technologies as soon as possible to protect today’s data and information from tomorrow's UQC attacks.

The field of computer science continues to advance with the discovery and development of new computer architectures and designs. We have already seen the impact of new architectures such as cloud computing, GPU (Graphics Processing Units), low-power electronics and Internet of Things (IoT) on computing. New architectures are being studied and developed by both industry and academia. The DoD should not only continue to invest in the development of new architectures but also to invest in new methods for quicker adoption of these technologies.

Given today's challenge of cybersecurity and software assurance, R&D must continue into developing more trusted computing to thwart future cyber-attacks and being able to execute software with assurance on untrusted networks and hardware.

The DoD should invest in new approaches to software development (beyond agile), including the use of computer assisted programming and project management. While the Agile development process is currently the best practice in industry, managing the software cycle is still more art form than science. New analytical approaches and next generation management tools could significantly improve software performance and schedule predictability. The Department should fund ongoing research as well as support academic, commercial, and development community efforts to innovate the software process.

//PAGE INTENTIONALLY LEFT BLANK//

Chapter 3. Been There, ~~Done~~ Said That: Why Hasn't This Already Happened?

DoD and Congress have a rich history of asking experts to assess the state of DoD software capabilities and recommend how to improve them. A DoD joint task force chaired by Duffel in 1982 started their report by saying:

> Computer software has become an important component of modern weapon systems. It integrates and controls many of the hardware components and provides much of the functional capability of a weapon system. Software has been elevated to this prominent role because of its flexibility to change and relatively low replication cost when compared to hardware. It is the preferred means of adding capability to weapon systems and of reacting quickly to new enemy threats
>
> *Report of the DoD Joint Service Task Force on Software Problems, 1982.*

Indeed, this largely echoes our own views, though the scope of software has now moved well beyond weapons systems, the importance of software has increased even further, and the rate of change for software is many orders of magnitude faster, at least in the commercial world.

Five years later, a task force chaired by Fred Brooks began its executive summary as follows:

> Many previous studies have provided an abundance of valid conclusions and detailed recommendations. Most remain unimplemented. … the Task Force is convinced that today's major problems with military software development are not technical problems, but management problems.
>
> *Report of the Task Force on Military Software, Defense Science Board, 1987.*

This particular assessment, from over 30 years ago, referenced over 30 previous studies and is largely aligned with the assessments of more recent studies, including this study.

And finally, in its 2000 study on DoD software, DSB Chair Craig Fields commented that

> Numerous prior studies contain valid recommendations that could significantly and positively impact DOD software development programs. However the majority of these recommendations have not been implemented. Every effort should be made to understand the inhibitors that prevented previous recommendations.
>
> *Defense Science Board Task Force on Defense Software,* 2000.

The problem is not that we do not know what to do, but that we simply are not doing it. In this chapter we briefly summarize some of the many reports that have come before ours and attempt to provide some understanding of why the current state of affairs in defense software is still so problematic. Using these insights, we attempt to provide some level of confidence that our recommendations might be handled differently (remembering that "hope is not a strategy").

3.1 Brief summary and assessment of 37 years of reports on DoD software

The following table lists previous reports focused on improving software acquisition and practices within DoD:

Date	Org	Short title / Summary of contents
Jul'82	DoD	Joint Service Task Force on Software Problems ● 37 pp + 192 pp Supplementary Info (SI); 4 major recommendations (recs) ● Software represents important opportunity ● DoD should take a lead in embedded software
Sep'87	DSB	Task Force on Military Software 41 pp + 36 pp SI; 38 recs Vision for rapid development and deployment of software, moving away from waterfall model
Dec'00	DSB	Task Force on Defense Software TBD: XX pp + YY major recs TBD: 2-3 line summary of what the report covers and key insights/takeaways.
2004	RAND	Attracting the Best: How the Military Competes for Information Technology Personnel TBD: XX pp + YY major recs TBD: 2-3 line summary of what the report covers and key insights/takeaways.
Feb'08	NCMA	Generational Inertia - An Impediment to Innovation? TBD: XX pp + YY major recs TBD: 2-3 line summary of what the report covers and key insights/takeaways.
Mar'09	DSB	Task Force on Department of Defense Policies and Procedures for the Acquisition of Information Technology 68 pp + 2 pp dissent + 15 pp SI; 4 major recs with 13 subrecs TBD: 2-3 line summary of what the report covers and key insights/takeaways.
2010a	NRC	Achieving Effective Acquisition of Information Technology in the Department of Defense TBD: XX pp + YY major recs TBD: 2-3 line summary of what the report covers and key insights/takeaways.
2010b	NRC	Critical Code: Software Producibility for Defense TBD: XX pp + YY major recs TBD: 2-3 line summary of what the report covers and key insights/takeaways.
Jul'16	CRS	The Department of Defense Acquisition Workforce: Background, Analysis, and Questions for Congress TBD: XX pp + YY major recs TBD: 2-3 line summary of what the report covers and key insights/takeaways.

Dec'16	CNA	Independent Study of Implementation of Defense Acquisition Workforce Improvement Efforts TBD: XX pp + YY major recs TBD: 2-3 line summary of what the report covers and key insights/takeaways.
Feb'17	SEI	DoD's Software Sustainment Study Phase I: DoD's Software Sustainment Ecosystem - *For copies please contact the Office of the Deputy Assistant Secretary of Defense for Materiel Readiness, Pentagon.* 101 pp + 5 major recs Since the time in the early 1980s when software began to be recognized as important to DoD, software sustainment has been considered a maintenance function. After almost four decades, DoD is also at a tipping point where it needs to deal with the reality that software sustainment is not about maintenance, but rather it is about continuous systems and software engineering for the life cycle to evolve the software product baseline. This report recommends changing that paradigm to enable the innovation needed to address a rapidly changing technology environment, specifically through investments in human capital, better performance measurement of software sustainment, and better visibility for the software portfolio.
Mar'17	BPC	Building a F.A.S.T. Force: A Flexible Personnel System for a Modern Military TBD: XX pp + YY major recs TBD: 2-3 line summary of what the report covers and key insights/takeaways.
Feb'18	DSB	Design and Acquisition of Software for Defense Systems 28 pp + 22 pp SI; 7 (high-level) recs + ~32 subrecs Transition to the use of software factories and continuous iterative development for DoD software; expand acquisition workforce knowledge of software
2018	2016 NDAA	Section 809 Panel - Streamlining and Codifying Acquisition [comparison] 1,275 pages, 93 recommendations Comprehensive review of Title 10, FAR, DFARS and recommendations on what needs to change
Apr'19	DIB	Software is Never Done; Refactoring the Acquisition Code for Competitive Advantage (this document) 32 pp + 150 pp SI; 4 lines of effort, ~10 recommendations (+ the next 16) Speed/cycle time as key metrics, build digital talent and infrastructure, avoid one-size-fits-all

Studies dating back to at least 1982 have identified software as a particular area of growing importance to the DoD, and software acquisition as requiring improvement, and the frequency and urgency of such studies identifying software acquisition as a major issue requiring reform has increased markedly since 2010. Notable recent examples include the 2010 studies by the National Research Council on *Achieving Effective Acquisition of Information Technology in the Department of Defense* and *Critical Code: Software Producibility for Defense*, the 2017 study conducted by Carnegie Mellon Software Engineering Institute (SEI) on DoD's Software Sustainment Ecosystem, and the 2018 DSB study on *Design and Acquisition of Software for Defense Systems*.

The properties of software that contribute to its unique and growing importance to DoD are summarized in this quote from the 2010 *Critical Code* study:

> This growth is a natural outcome of the special engineering characteristics of software: Software is uniquely unbounded and flexible, having relatively few intrinsic limits on the

> degree to which it can be scaled in complexity and capability. Software is an abstract and purely synthetic medium that, for the most part, lacks fundamental physical limits and natural constraints. For example, unlike physical hardware, software can be delivered and up-graded electronically and remotely, greatly facilitating rapid adaptation to changes in adversary threats, mission priorities, technology, and other aspects of the operating environment. The principal constraint is the human intellectual capacity to understand systems, to build tools to manage them, and to provide assurance—all at ever-greater levels of complexity.
>
> *Critical Code: Software Producibility for Defense,* NRC, 2010

Prior studies (e.g., [SEI2017]) have commented on the fact that much of DoD software acquisition policy is systems- and hardware-oriented and largely does not take these unique properties into account.

The lack of action on most of the software recommendations from these studies has also been a subject of perennial comment. The DSB's 2000 study noted this phenomenon:

> [Prior] studies contained 134 recommendations, of which only a very few have been implemented. Most all of the recommendations remain valid today and many could significantly and positively impact DoD software development capability. The DoD's failure to implement these recommendations is most disturbing and is perhaps the most relevant finding of the Task Force. Clearly, there are inhibitors within the DoD to adopting the recommended changes.
>
> *Task Force on Defense Software,* Defense Science Board, 2000.

The situation has not changed significantly since then despite additional studies and significant numbers of new recommendations. There is little to suggest that the inhibitors to good software practice have changed since 2000, and it is likely that the pace of technological change and capabilities provided by software have only increased since then.

Major categories of prior recommendations. The DIB-SWAP study team conducted a literature review of prior work on DoD software acquisition and extracted the specific recommendations that had been made, binning them according to major topics. The focus of the effort was on recent studies, with the bulk of the work since 2010, resulting in 139 recommendations that were extracted and categorized.

A few prevailing themes stood out from this body of work, representing issues that were commented upon in multiple studies:

- Contracts: contracts should be modular and flexible
- Test and evaluation: test and evaluation should be incorporated throughout the software process with close user engagement
- Workforce: software acquisition requires specific skills and knowledge along with user interaction and senior leadership support
- Requirements: requirements should be reasonable and prioritized; some advocacy for alternative requirement documentation (product vision)
- Acquisition strategy/oversight: DoD should encourage agencies to pursue business process innovations.

The three areas which were dealt with most often in the prior studies were acquisition oversight, contracting, and workforce. These three topics alone accounted for 60 percent of all of the recommendations we compiled. We summarize the major recurring prior recommendations in each of those areas as follows:

Recommendations from recent work in acquisition oversight:

- Ensure non-interruption of funding of programs that are successfully executing to objective (rather than budget), while insulating programs from unfunded mandates.
- Durations should be reasonably short and meaningful and should allow for discrete progress measurement.
- Design the overall technology maturity assessment strategy for the program or project.
- Encourage program managers to share bad news, and encourage collaboration and communication.
- Require program managers to stay with a project to its end.
- Empower program managers to make decisions on the direction of the program and to resolve problems and implement solutions.
- Follow an evolutionary path toward meeting mission needs rather than attempting to satisfy all needs in a single step.

Recommendations from recent work in contracting:

- Requests for proposals (RFPs) for acquisition programs entering risk reduction and full development should specify the basic elements of the software framework supporting the software factory, including code and document repositories, test infrastructure, software tools, check-in notes, code provenance, and reference and working documents informing development, test, and deployment.
- Establish a common list of source selection criteria for evaluating software factories for use throughout the Department.
- Contracting Officers (KOs) must function as strategic partners tightly integrated into the program office, rather than operate as a separate organization that simply processes the contract paperwork.
- Develop and maintain core competencies in diverse acquisition approaches and increase the use of venture capital type acquisitions such as Small Business Innovative Research (SBIR), Advanced Concept Technology Development (ACTD), and Other Transaction Authority (OTA) as mechanisms to draw in non-traditional companies.

Recommendations from recent work on workforce issues:

- The service acquisition commands need to develop workforce competency and a deep familiarity of current software development techniques.
- The different acquisition phases require different types of leaders. The early phases call for visionary innovators who can explore the full opportunity space and engage in intuitive decision-making. The development and production phases demand a more pragmatic orchestrator to execute the designs and strategies via collaboration and consensus decisions.
- U.S. Special Operations Command (USSOCOM) must develop a unique organizational culture that possesses the attributes of responsiveness, innovation, and problem solving necessary to convert strategic disadvantage into strategic advantage.

- Encourage employees to study statutes and regulations and explore innovative and alternative approaches that meet the statutory and regulatory intent.
- Rapid acquisition succeeds when senior leaders are involved in ensuring that programs are able to overcome the inevitable hurdles that arise during acquisition, and empower those responsible with achieving the right outcome with the authority to get the job done while minimizing the layers in between.

To help illustrate the continuity of the history of these issues and the lack of progress despite consistent, repeated similar findings, we consider the case of recommendations related to software capabilities of the acquisition workforce (areas where we are also recommending change).

Calls to improve DoD's ability to include software expertise in its workforce have a long history. DoD studies dating back to 1982 have raised concerns about the technical competencies and size of DoD's software workforce [DSB'82, DSB'87]. In 1993, the DoD Acquisition Management Board identified a need to review the DoD's software acquisition management education and training curricula. This study concluded that no existing DoD workforce functional management group was responsible for the software competencies needed in the workforce and that software acquisition competencies were needed in many different acquisition career fields. However, the Board asserted that no new career field was needed for Software Acquisition Managers. In 2001, the same concerns regarding the software competencies of the DoD acquisition workforce once again surfaced. The DoD Software Intensive Systems Group conducted a software education and training survey of the acquisition workforce. This survey demonstrated that less than 20 percent of the ACAT program staff had taken the basic Software Acquisition Management course (SAM 101) and that less than 20 percent of the ACAT program staff had degrees in computer science, software engineering, or information technology. The specific recommendations from this analysis included: (1) institute mandatory software intensive systems training for the workforce; (2) develop a graduate-level program for software systems development and acquisition; and (3) require ACAT 1 programs to identify a chief software/ systems architect.

A year later, Congress mandated that the Secretary of each military department establish a program to improve the software acquisition processes of that military department. Subsequently each Service established a strategic software improvement program (Army 2002, Air Force 2004, and Navy 2006). These Service initiatives have continued at some level. However, with the sunsetting of the Software Intensive Systems Group at the OSD level, the enterprise focus on software waned. During this same period, the Navy started the Software Process Improvement Initiative (SPII), which identified issues preventing software-intensive projects from meeting schedule, cost, and performance goals. This initiative highlighted the lack of adequately educated and trained software acquisition professionals and systems engineers.

In 2007, OSD issued guidance to create the Software Acquisition Training and Education Working Group (SATEWG) with a charter to affirm required software competencies, identify gaps in Defense Acquisition Workforce Improvement Act (DAWIA) career fields, and to develop a plan to address those gaps. This group was composed of representatives from the Services, OSD, and other organizations, including Carnegie Mellon SEI. The group developed a software competency framework that identified four key knowledge areas and 29 competencies that could inform the different acquisition workforce managers as to the software competencies to be integrated into their existing career field competency models. There has been no follow-on effort to evaluate the progress of the SATEWG or its outcomes.

Today, in the absence of a DoD-wide approach to describing, managing, and setting goals against a common understanding of needed software skills, each Service (as well as software sustainment organizations) has evolved its own approach or model for identifying software competencies for its workforce.

This historical context highlights two key points. First, DoD has long recognized the challenges of addressing the technical competencies and size of the software workforce across the life cycle. However, there is limited evidence of the outcomes from these different efforts. Secondly, this history clearly indicates that acquiring software human capital and equipping that workforce with the necessary competencies is a persistent and dynamic challenge that demands a continuous enterprise strategy.

3.2 Our Interpretation of Why Nothing Happened but Why We Think Our Report Will Matter

Given the long and profound history of inaction on past studies, we have attempted to create our own “Theory of (Non)Change.” Why does the Department struggle to step up to rational, generally agreed-upon change? We offer the following three drivers:

The (patriotic and dutifully) frozen middle. Our process in executing this study has been to talk to anyone and everyone we could within various departments of the DoD and the Services, to gather as many different perspectives as possible on what is needed, and to find out what is working and what needs to be stomped upon. As with many change management opportunities we find significant top-down support for what we are trying to do, especially from those who see the immediate need for more, better, faster mission capability and those who are directly frustrated at the command level by the current processes that are just not working. At the other end, we see digital natives demanding change but with limited power to make it happen; people who are fully enmeshed in how the tech world works, people who have all the expectations that have been created by their private sector lifestyle and economy. And then we have *the middle,* who are dutifully following the rules, and have been trained and had success defined for a different world. For *the middle,* new methodologies and approaches introduce unknown risks, while the old acquisition and development approaches built the world's best military. We question neither the integrity nor the patriotism of this group. They are simply not incentivized to the way we believe modern software should be acquired and implemented, and the enormous inertia they represent is a profound barrier to change.

Unrequited Congress. Congress is responsible for approving and overseeing the Department's development programs. While it is clear that Congress takes its oversight role seriously, it does so knowing that to have oversight requires something to oversee, and it understands its fundamental responsibility is to enable the Department to execute its mission. But oversight matters, and recommendations for change that do not also provide insight into how new ways of doing things will allow Congress to perform its role are a very tough sell. In addition, there is a sense of unrequited return from past changes and legislation such as Other Transaction Authorities (OTAs), pilot programs and special hiring authorities. In many cases, Congress believes it has

already provided the tools and flexibilities for which the Department has asked. It is perhaps unreasonable to expect a positive response to ask for more when current opportunities have not been fully exploited.

Optimized Acquisition (for something else!).

> *Knowing was a barrier which prevented learning.*
>
> Frank Herbert

While some may (justifiably) argue that the current acquisition system is not optimized for anything, it is the product of decades of rules upon rules, designed to speak to each and every edge case that might crop up in the delivery of decades-long hardware systems, holds risk elimination at a premium, and has a vast cadre of dedicated practitioners exquisitely trained to prosper within that system. This is a massive barrier to change and informs our recommendations that to argue for major new ways of acquiring software and not just attempt to re-optimize to a different local maximum.

What we are trying to do that we think is different. Given the long history of DoD and Congressional reports that make recommendations that are not implemented, why do we think that this report is going to be any different? Our approach has been to focus not on the report and its recommendations *per se*, but rather in the series of discussions around the ideas in this report and the people we have interacted with inside the Pentagon and at program site visits. The recommendations in this report thus serve primarily as documentation of a sequence of iterative conversations and the real work of the report is the engagements before and after the report is released.

We also believe that there are some ideas in the report that, while articulated in many places in different ways, are emphasized differently here. In particular, a key point of focus in this report is the use of speed and cycle time as the key drivers for what needs to change and optimizing statutes, regulations, and processes to allow management and oversight of software. We believe that optimizing for the speed at which software can be utilized for competitive advantage will create an acquisition system that is much better able to provide security, insight, and scale.

Finally, we have tried to make this report shorter and pithier than previous reports, so we hope people will read it. It also is staged so that each reader, with their specific levels of authority and responsibility, can navigate an efficient path to reaching their conclusions on how best to support what is contained here.

3.3 Consequences of Inaction: Increasing Our Attack Surface and Shifting Risk to the Warfighter

So, what happens if history does, in fact, repeat itself and we again fail to step up to the changes that have been so clearly articulated for so long? Certainly by continuing to follow acquisition processes designed to limit risk for the hardware age, we will not reduce risk but instead will simply transfer that risk to the worst possible place—the warfighter who most needs the tools in her arsenal to deliver the missions we ask her to perform. But in addition, as we have continually

stressed throughout this study, there are several real differences in today's world compared to the environment in which past efforts were made.

First, and most important, weapons systems, and the bulk of the operational structure on which DoD executes its mission, are now fundamentally software (or software-defined) systems, and as such, delays in implementing change amplify the capability gaps that slow, poor, or unsupportable software creates. Second, the astonishing growth of the tech sector has created a very different competitive environment for the talent most needed to meet DoD's needs. Decades ago, DoD was the leading edge of the world's coolest technology and passionate, skilled software specialists jumped at the chance to be at that edge. That is simply not the case today and while a commitment to national security is a strong motivator, if the changes recommended in this study are not implemented, the competitive war for talent, *within our country,* will be lost.

The modern software methodologies enumerated in this report – and the recommendations concerning culture, regulation and statute, and career trajectories that enable those methodologies – are the best path to providing secure, effective, and efficient software to users. Cyber assurance, resilience, and relevance are all delivered much more effectively when done quickly and incrementally, using the tools and methods recommended in this study.

Finally we call attention back to Section 1.4 (What are the challenges that we face [and consequences of inaction]?). To summarize: "The long-term consequence of inaction is that our adversaries' software capabilities can catch and surpass ours...Our adversaries' software capabilities are growing as ours are stagnating."

//PAGE INTENTIONALLY LEFT BLANK//

Chapter 4. How Do We Get There From Here: Three Paths for Moving Forward

The previous three chapters provided the rationale for why we need to *do* (not just say) something different about how DoD develops, procures, assures, deploys, and continuously improves software in support of defense systems. The private sector has figured out ways to use software to accelerate their businesses and DoD should accelerate its incorporation of those techniques to its own benefit, especially in ensuring that its warfighters have the tools they need in a timely fashion to execute their missions in today's software-dominated environment. In this chapter, we lay out three different paths for moving forward, each under a different set of assumptions and objectives. A list of some representative, high-level steps are provided for each path, along with a short analysis of advantages and weaknesses.

4.1 Path 1: Make the Best Out of What We've Got

Congress has provided DoD with substantial authority and flexibility to implement the mission of the DoD. Although difficult and often inefficient, it is possible to implement the major goals of this report making use of the existing authorities and, indeed, there are already examples of the types of activities that we envision taking place across OSD and the Services. In this section, we attempt to articulate a path that builds on these successes and does not require any change in the law nor major changes in regulatory structure. The primary steps required to implement this path should focus on changing the practice by which software is developed, procured, assured, and deployed as well as updating some of the regulations and processes to facilitate cultural and operational changes.

To embark on this first path, DoD should streamline its processes for software, allowing more rapid procurement, deployment, and updating of software. OSD and the Services should also work together to allow better cross-service and pre-certified ATOs, easier access to large-scale cloud computing, and use of modern tool chains that will benefit the entire software ecosystem. The acquisition workforce, both within OSD and the Services, should be provided with better training and insight on modern software development (one of the more frequent recommendations over the past 37 years) so that they can take advantage of the approaches that software allows that are different than hardware. Most importantly, government and industry must come together to implement a DevSecOps culture and approach to software, building on practices that are already known and used in industry.

The following list provides a summary of high-level steps that require changes to DoD culture and process, but could be taken with no change in current law and relatively minor changes to existing regulations:

- Make use of existing authorities such as OTAs and mid-tier acquisition (Sec 804) to implement a DevSecOps approach to acquisition to the greatest extent possible under existing statutes, regulations, and processes.
- Require cost assessment and performance estimates for software programs (and software components of larger programs) to be based on metrics that track speed and cycle time, security, code quality, and useful capability deliver to end users.

- Create a mechanism for ATO reciprocity between Services and industrial base companies to enable sharing of software platforms, components and infrastructure and rapid integration of capabilities across (hardware) platforms, (weapons) systems, and Services.
- Remove obstacles to DoD usage of cloud computing on commercial platforms, including Defense Information System Agency (DISA) cloud access point (CAP) limits, lack of ATO reciprocity, and access to modern software development tools.
- Expand the use of (specialized) training programs for chief information officers (CIOs), service acquisition executives (SAEs), program executive officers (PEOs), and program managers (PMs) that provide (hands-on) insight into modern software development (e.g., agile, DevOps, DevSecOps) and the authorities available to enable rapid acquisition of software.
- Increase the knowledge, expertise, and flexibility in program offices related to modern software development practices to improve the ability of program offices to take advantage of software-centric approaches to acquisition.
- Require access to source code, software frameworks, and development toolchains, with appropriate intellectual property (IP) rights, for all DoD-specific code, enabling full security testing and rebuilding of binaries from source.
- Create and use automatically generated, continuously available metrics that emphasize speed, cycle time, security, and code quality to assess, manage, and terminate software programs (and software components of hardware programs).
- Shift the approach for acquisition (and development) of software (and software- intensive components of larger programs) to an iterative approach: start small, be iterative, and build on success – or be terminated quickly.
- Make security a first-order consideration for all software-intensive systems, under the assumption that security-at-the-border will not be enough.
- Shift from a list of requirements for software to a list of desired features and required interfaces/characteristics to avoid requirements creep or overly ambitious requirements.
- Maintain an active research portfolio into next-generation software methodologies and tools, including the integration of machine learning and AI into software development, cost estimation, security vulnerabilities, and related areas.
- Invest in transition of emerging approaches from academia and industry to creating, analysis, verification, and testing of software into DoD practice (via pilots, field tests, and other mechanisms).
- Automatically collect all data from DoD weapons systems and make available for machine learning (via federated, secured enclaves, not a centralized repository).
- Mandate a full program review within the first 6-12 months of development to determine if a program is on track or requires corrective action or deserves cancellation.

This path has the advantage that the authorities required to undertake it are already in place and the expertise exists within the Department to begin moving forward. We believe that the there is strong support for these activities at the top and bottom of the system, and existing groups (e.g., DDS, JIDO, Kessel Run) have demonstrated that the flexibilities exist within the existing system to develop, procure, deliver, and update software more quickly. The difficulty in this path is that it requires individuals to figure out how to go beyond the default approaches that are built into the current acquisition system. Current statutes, regulations, and processes are very complicated, there is a “culture of no” that must be overcome, and hence using the authorities that are available

requires substantial time, effort, and risk (to one's career, if not successful). The risk in pursuing this path is that change occurs too slowly or not at scale, and we are left with old software that is vulnerable and cannot serve our needs. Our adversaries have the same opportunities that we do for taking advantage of software and may be able to move more quickly if the current system is left in place.

4.2 Path 2: Tune the Defense Acquisition System to Optimize for Software

While the first steps to refactoring the defense acquisition system can be taken without necessarily having to change regulations, the reality of the current situation is that Congress and DoD have created a massive body of laws and regulations that are just slowing things down. This might be OK for hardware, but it is definitely not OK for (most types of) software, as we have articulated in the previous three chapters. This second, more difficult path to software acquisition and practice reform, should focus on rewriting selected pieces of old code (= legislation and regulations) that are doing more harm than good. These changes would apply to both software that is acquired as well as software that is built.

The following list provides a set of high-level steps that require some additional changes to DoD culture and process, but also modest changes in current law and existing regulations. These steps build on the steps listed in path 1 above, although in some cases they can solve the problems that the previous actions were trying to work around.

- Refactor and simplify Title 10 and the defense acquisition system to remove all statutory, regulatory, and procedural requirements that generate delays for acquisition, development and fielding of software while adding requirements for continuous (automated) reporting of cost, performance (against updated metrics), and schedule.
- Create streamlined authorization and appropriation processes for defense business systems (DBS) that use commercially-available products with minimal (source code) modification.
- Plan, budget, fund, and manage software development as an enduring capability that crosses program elements and funding categories, removing cost and schedule triggers that force categorization into hardware-oriented regulations and processes.
- Replace JCIDS, PPBE, and DFARS with a "PEO Digital" in each Service that uses portfolio management and direct identification of warfighter needs to decide on allocation priorities.
- Create, implement, support, and require a fully automatable approach to T&E, including security that allows high-confidence distribution of software to the field on an iterative basis (with frequency dependent on type of software, but targets cycle times measured in weeks).
- Prioritize secure, iterative, collaborative development for selection and execution of all new software programs (and software components of hardware programs) (see DIB's Detecting Agile BS as an initial view of how to evaluate capability).
- For any software developed for DoD, require that software development be separated from hardware in a manner that allows non-prime vendors to bid for software elements of the program on a performance-based basis.
- Shift from certification of executables, to certification of code, to certification of the development, integration, and deployment toolchain, with the goal of enabling rapid fielding of mission-critical code at high levels of information assurance.

- Require CIOs, SAEs, PEOs, PMs and any other acquisition roles involving software development as part of the program to have prior experience in software development.
- Restructure the approach to recruiting software developers to assume that the average tenure of a talented engineering will be 2-4 years, and make better use of highly qualified experts (HQEs), intergovernmental personnel act employees (IPAs), reservists, and enlisted personnel to provide organic software development capability.
- Establish a Combat Digital Service (CDS) unit within each combatant command consisting of software development talent that can be used to manage command-specific IT assets, at the discretion of the combatant commander.

This path takes a more active approach to modifying the acquisition system for software by identifying those statutes, regulations, and processes that are creating the worst bottlenecks and modifying them to allow for faster delivery of software to the field. We see this path as one of removing old pieces of code (statutory, regulatory, or process) that are no longer needed or that should not be applied to software, as well as increasing the expertise in how modern software development works so that software programs (and software-centric elements of larger programs) can be optimized for speed and cycle time. Pursuing this path will allow faster updates to software and will improve security and oversight (via increased insight). In many cases, the Department is already executing some of the actions required to enable this path. The weakness in this path is that software would generally use the same basic approach to acquisition as hardware, with various carve-outs and exceptions. This runs the risk that software programs still move too slowly due to the large number of people who have to say yes and the need to train a very large acquisition force to understand how software is different than hardware (and not all software is the same).

4.3 Path 3: An Acquisition Pathway and New Appropriations Category for Software to Force Change in the Middle

The final path is the most difficult and will require dozens of independent groups to agree on a common direction, approach, and set of actions. At the end of this path lies a new defense acquisition system that is optimized for software-centric systems instead of hardware-centric systems, and that prioritizes security, speed, and cycle time over cost, schedule, and (rigid) requirements.

To undertake this path, Congress and OSD must write new statutes and regulations for software, providing increased (and automation-enabled) insight to reduce the risk of slow, costly, and overgrown programs and enabling rapid deployment and continuous improvement of software to the field. Laws will have to be changed, and management and oversight will have to be reinvented, focusing on different measures and a quicker cadence. OSD and the services will need to create and maintain interoperable (cross-program/cross-service) digital infrastructure that enables rapid deployment, scaling, testing, and optimization of software as an enduring capability; manage them using modern development methods; and eliminate the existing hardware-centric regulations and other barriers. Finally, the Services will need to establish software development as a high visibility, high-priority career track with specialized recruiting, education, promotion, organization, incentives, and salary.

The following list of high-level steps required to pursue this path, building on the steps listed in the previous paths:

- Establish a new acquisition pathway (Sec 805) for software that prioritizes the ability to rapidly field and iterate new functionality in a secure manner, with continuous oversight based on automated reporting and analytics, and utilizing IA-accredited commercial development tools.
- Create a new appropriations category that allows (relevant types of) software to be funded as a single budget item, with no separation between RDT&E, production, and sustainment; remove cost and schedule triggers associated with hardware-focused regulations and processes.
- Establish and maintain digital infrastructure within each Service or Agency that enables rapid deployment of secure software to the field and make it available to contractors at subsidized cost.
- Plan and fund computing hardware (of all types) as consumable resources, with continuous refresh and upgrades to the most recent, most secure operating system and platform components.
- Create software development groups in each Service consisting of military and/or civilian personnel who write code that is used in the field and track individuals who serve in these groups for future DoD leadership roles.

This path attempts to solve the longstanding issues with software by creating an appropriations category and acquisition pathway that is fine-tuned for software. It will require a very large effort to get the regulations, processes, and people in place that are required to execute it effectively, and there will be missteps along the way that generate controversy and unwanted publicity. In addition, it will likely be opposed by those currently in control of selling or making software for the DoD, since it will require that they retool their business to a very new approach that is not well-defined at the outset.

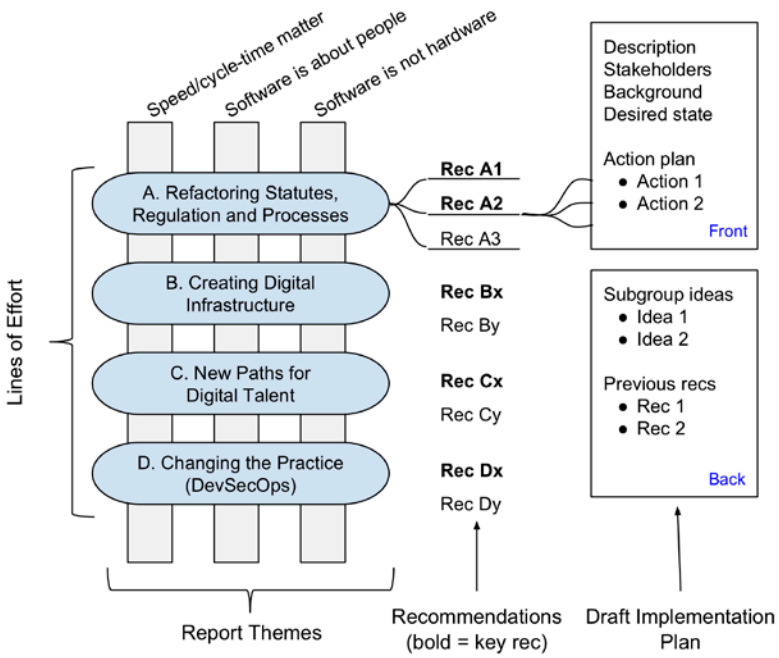
//PAGE INTENTIONALLY LEFT BLANK//

Chapter 5. What Would the DIB Do: Recommendations for Congress and DoD

In this final chapter we lay out our recommendations for what Congress and DoD should do to implement the type of software acquisition and practices reform that we believe is needed for the future. Our recommendations are organized according to four primary lines of effort:

A. Congress and OSD should refactor statutes, regulations, and processes for software
B. OSD and the Services should create and maintain cross-program/cross-service digital infrastructure
C. The Services should create new paths for digital talent (especially internal talent)
D. Acquisition offices and contractors must change the practice of how software is procured and developed

For each of these lines of effort, we have identified the 2-3 most important recommendations that we believe Congress and DoD should undertake. These are our "Top Ten" recommendations. These ten recommendations were chosen not because they solve the entire problem, but because they will make the biggest difference; without them, substantial change is not likely. In addition, we have identified 16 more recommendations for consideration once the execution of the first ten recommendations is successfully underway. For each recommendation, a draft implementation plan is provided in Appendix A that gives a list of actions required to implement the recommendation, as well as more detail on the rationale, supporting information, and similar recommendations from other studies. Potential legislative and regulatory language to implement selected recommendations is included in Appendix B. While we have tried hard to provide specific actions, owners and target dates that will drive an implementation plan for each recommendation, we recognize that in the end, owners will be decided by the Department's response to our study and owners will use our actions as a starting point to their own Implementation Plan,



Recommendation structure. For each line of effort, a set of primary recommendations (bold) are provided, along with a set of additional recommendations for consideration. Each recommendation contains a draft implementation plan that includes a background information on the rationale, vision, and stakeholders.

5.1 The Ten Most Important Things To Do (Starting Now!)

In this section we lay out what we believe are the most important steps for Congress and DoD to take to fully leverage the opportunities presented by software and the private sector's strength in modern development practices. Our commitment to these steps will directly impact the Department's ability to achieve the 2018 National Defense Strategy[15] goals of increased lethality, stronger alliance while positioning for new partnerships, and reformed business practices for better performance and affordability - all of which

Line of effort A. Congress and OSD should refactor statutes, regulations, and processes for software, providing increased insight to reduce the risk of slow, costly, and overgrown programs, and enabling rapid deployment and continuous improvement of software to the field. Reinvent management and oversight, focusing on different measures and a quicker cadence.

> **Recommendation A1.** Establish new acquisition pathway(s) for software that prioritizes continuous integration and delivery of working software in a secure manner, with continuous oversight from automated analytics

Current law, regulation, and policy, and internal DoD processes make DevSecOps-based software development extremely difficult, requiring substantial and consistent senior leadership involvement. Consequently, DoD is challenged in its ability to scale Agile SW development practices to meet mission needs. The desired state is that programs have the ability to rapidly field and iterate new functionality in a secure manner, with continuous oversight based on automated reporting and analytics, and utilizing IA-accredited commercial development tools.

[TBD: Summarized actions]

[TBD: Summarized supporting information]

> **Recommendation A2.** Create a new appropriations category that allows (relevant types of) software to be funded as a single budget item, with no separation between RDT&E, production, and sustainment.

Current law, regulation, and policy treat software acquisition as a series of discrete sequential steps; accounting guidance treats software as a depreciating asset. These processes are at odds with software being continuously updated to add new functionality and create significant delays in fielding user-needed capability. The desired state is that programs are better able to prioritize how effort is spent on new capabilities versus fixing bugs / vulnerabilities, improving existing capabilities, etc. Such prioritization can be made based on warfighter / user needs, changing mission profiles, and other external drivers, not constrained by available sources of funding.

Implementation of this recommendation could be accomplished by having USD(A&S) submit a legislative proposal to create a new appropriations category for software and software-intensive programs for approval by the House and Senate Armed Services Committees and funding by the

[15] https://dod.defense.gov/Portals/1/Documents/pubs/2018-National-Defense-Strategy-Summary.pdf&sa=D&ust=1552454086241000&usg=AFQjCNHF0VZL0skCFY8w_ouUq52nPDx4mA

House and Senate Appropriations Committees. The DoD Comptroller, working with CAPE, would need to make necessary modifications in supporting PPBE systems to allow use and tracking of new software appropriation. USD(A&S), in coordination with the Service Acquisition Executives (SAEs) should select the initial programs that will use the new software appropriation from those that are currently using DevSecOps. Budget exhibits for the new software appropriation, replacing the current P-Forms and R-Forms, should be prepared by USD(A&S) working with USD(C), CAPE, and the Appropriations Committees, and those programs selected for using the new appropriation category should begin using the exhibits upon selection into the category. Finally, FASAB in coordination with USD(A&S) and USD(C) will need to change the audit treatment of software for this category to achieve the following: (1) separate category for software instead of being characterized as property, plant, and equipment; (2) default setting that software is an expense, not an investment; and (3) there is no "sustainment" phase for software.

This recommendation builds on the recommendations in the Ten Commandments on Software and our Visit Observations and Recommendations that budgets for software (and software- intensive) programs should support the full, iterative life-cycle of the software. In addition, the Acquisition, Appropriations Strategy, Contracting, and Sustainment and Maintenance subgroups all had recommendations that support this approach. The basic approach advocated here was also articulated in the 1987 Defense Science Board task for on military software, the GAO studies in 2015 and 2017, and is consistent with the Portfolio Management Framework Recommendations 41 and 42 of the Section 809 Panel.

Line of Effort B. OSD and the Services should create and maintain cross-program/cross-service digital infrastructure that enables rapid deployment, scaling, and optimization of software as an enduring capability, managed using modern development methods in place of existing (hardware-centric) regulations, and providing more insight (and hence better oversight) for software-intensive programs.

Recommendation B1. Establish and maintain digital infrastructure within each Service or Agency that enables rapid deployment of secure software to the field and incentivize its use by contractors.

Currently, DoD programs each develop their own development and test environments, which requires redundant definition and provisioning, replicated assurance (including cyber), and extended lead times to deploy capability. Small companies have difficulties providing software solutions to DoD because those environments are not available outside the incumbent contractor or they have to build (and certify) unique infrastructure from scratch.

[TBD: Summarized actions]

[TBD: Summarized supporting information]

Recommendation B2. Create, implement, support, and use fully automatable approaches to testing and evaluation (T&E), including security, that allow high confidence distribution of software to the field on an iterative basis

To deliver SW at speed, rigorous, automated testing processes and workflows are essential. Current DoD practices and procedures often see OT&E as a tailgate process, sequentially after development has completed, slowing down delivery of useful software to the field and leaving existing (potentially poorly performing and/or vulnerable) software in place. The desired state is that development systems, infrastructure and practices are focused on continuous, automated testing by developers (with users). To the maximum extent possible, system operational testing is integrated (and automated) as part of the development cycle using data, information and test protocols delivered as part of the development environment. Testing and evaluation/ certification of COTS components done once (if justified) and then ATO reciprocity (Rec B3) is applied to enable use in other programs, as appropriate.

[TBD: Summarized actions]

[TBD: Summarized supporting information]

Recommendation B3. Create a mechanism for Authority to Operate (ATO) reciprocity within and between programs, Services, and other DoD agencies to enable sharing of software platforms, components and infrastructure and rapid integration of capabilities across (hardware) platforms, (weapons) systems, and Services.

Current software acquisition practice emphasizes the differences among programs: perceptions around different missions, different threats, and different levels of risk tolerance mean that components, tools, and infrastructure that have been given permission to be used in one context are rarely accepted for use in another. The lack of ATO reciprocity drives each program to create their own infrastructure, repeating time- and effort-intensive activities needed to certify elements as secure for their own specific context. The desired state is that modern software components, tools, and infrastructure, once accredited as secure within the DoD, can be used appropriately and cost-effectively by multiple programs. Programs can spend a greater percentage of their budgets on developing software that adds value to the mission rather than spending time and effort on basic software infrastructure. Accreditation of COTS components is done once and then made available for use in other programs, as appropriate.

[TBD: Summarized actions]

[TBD: Summarized supporting information]

Line of effort C. The Services should create new paths for digital talent (especially internal talent) by establishing software development as a high-visibility, high-priority career track and increasing the level of understanding of modern software within the acquisition workforce.

Recommendation C1. Create software development units in each Service consisting of military and civilian personnel who develop and deploy software to the field using DevSecOps practices.

The DoD's capacity to apply modern technology and software practices to meet its mission is required in order to remain relevant in increasingly technical fighting domains, especially against peer adversaries. While DoD has both military and civilian software engineers (often associated with maintenance activities), the IT career field suffers from a lack of visibility and support. The Department has not prioritized a viable recruiting strategy for technical positions, and there is no comprehensive training or development program that prepares the technical and acquisition workforce to adequately deploy modern software development tools and methodologies. The desired state is that DoD recruits, trains, and retains internal capability for software development, including by service members, and maintains this as a separate career track (like DoD doctors, lawyers, and musicians). Each Service has organic development units that are able to create software for specific needs and that serve as an entry point for software development capability in military and civilian roles (complementing work done by contractors). The Department's workforce embraces commercial best practices for the rapid recruitment of talented professionals, including the ability to onboard quickly and provide modern tools and training in state-of-the-art training environments. Individuals in software development career paths are able to maintain their technical skills and take on DoD leadership roles.

[TBD: Summarized actions]

[TBD: Summarized supporting information]

Recommendation C2. Expand the use of (specialized) training programs for CIOs, SAEs, PEOs, and PMs that provide (hands-on) insight into modern software development (e.g., agile, DevOps, DevSecOps) and the authorities available to enable rapid acquisition of software.

Acquisition professional have been trained and had success in the current model, which has produced the world's best military but this model is not serving well for software. New methodologies and approaches introduce unknown risks, and acquisition professionals are often not incentivized to make use of the authorities available to implement modern software methods. At the same time, senior leaders in DoD need to be more knowledgeable about modern software development practices so they can recognize, encourage, and champion efforts to implement modern approaches to software program management. The desired state is that senior leaders, middle management, and organic and contractor-based software developers are aligned in their view of how modern software is procured and developed. Acquisition professionals are aware of all of the authorities available for software programs and use them to provide flexibility and rapid delivery of capability to the field. Program leaders are able to assess the status of software (and software-intensive) programs and spot problems early in the development process, as well as provide continuous insight to senior leadership and Congress. Highly specialized requirements are scrutinized to avoid developing custom software when commercial offerings are available that are less expensive and more capable.

[TBD: Summarized actions]

[TBD: Summarized supporting information]

Line of effort D. Acquisition offices and contractors must change the practice of how software is procured and developed by adopting modern software development approaches, prioritizing speed as the critical metric, ensuring cyber protection is an integrated element of the entire software lifecycle, and purchasing existing commercial software whenever possible.

> **Recommendation D1.** Require access to source code, software frameworks, and development toolchains, with appropriate IP rights, for all DoD-specific code, enabling full security testing and rebuilding of binaries from source.

For many DoD systems, source code is not available to DoD for inspection and testing, and DoD relies on suppliers to write code for new compute environments. As code ages, suppliers are not required to maintain codebases without an active development contract and "legacy" code is not continuously migrated to the latest hardware and operating systems. The desired state is that DoD has access to source code for DoD-specific software systems that it operates and uses to perform detailed (and automated) evaluation of software correctness, security, and performance, enabling more rapid deployment of both initial software releases and (most importantly) upgrades (patches and enhancements). DoD is able to rebuild executables from scratch for all of its systems, and has the rights and ability to modify (DoD-specific) code when new conditions and features arise. Code is routinely migrated to the latest computing hardware and operating systems, and routinely scanned against currently-known vulnerabilities. Modern IP language is used to ensure that the government can use, scan, rebuild, and extend purpose-built code, but contractors are able to use licensing agreements that protect any IP that they have developed with their own resources. Industry trusts DoD with its code and has appropriate IP rights for internally developed code.

[TBD: Summarized actions]

[TBD: Summarized supporting information]

> **Recommendation D2.** Make security a first-order consideration for all software-intensive systems, under the assumption that security-at-the-border will not be enough.

[TBD: Background + desired state]

[TBD: Summarized actions]

[TBD: Summarized supporting information]

> **Recommendation D3.** Shift from the use of rigid lists of requirements for software programs to a list of desired features and required interfaces/characteristics, to avoid requirements creep, overly ambitious requirements, and program delays

Current DoD requirements processes significantly impede its ability to implement modern SW development practices by spending years establishing program requirements and insisting on satisfaction of requirements before a project is considered “done”. This impedes rapid implementation of features that are of the most use to the user. The desired state is that rather than a list of requirements for every feature, programs should establish a minimum set of requirements required for initial operation, security, and interoperability, and place all other desired features on a list that will be implemented in priority order, with the ability for DoD to redefine priorities on a regular basis.

[TBD: Summarized actions]

[TBD: Summarized supporting information]

5.2 The Next Most Important Things to Tackle

There are a large number of changes that will need to be made to fully realize the vision that 37 years of studies have articulated. This study solicited input from a wide range of stakeholders in the defense software enterprise, including OSD and Service leaders, industry participants in our visits and roundtables, and FFRDC personnel who helped put together our report to help identify the recommendations that we should make. The list of recommendations below are the next 0x10 (16) recommendations that we believe can be implemented after the ones above are solidly underway (like software, implementing recommendations is never “done”). We list these second not because they are dependent on the primary recommendations but simply to emphasize the urgency of the Top Ten.

ID	Recommendation
A3	Require cost assessment and performance estimates for software programs (and software components of larger programs) be based on metrics that track speed and cycle time, security, code quality, and functionality.
A4	Refactor and simplify Title 10, DFARS, and DoDI 5000.02/5000.75 to remove statutory, regulatory, and procedural requirements that generate delays for acquisition, development, and fielding of software while adding requirements for continuous (automated) reporting of cost, performance (against updated metrics), and schedule
A5	Create streamlined authorization and appropriation processes for defense business systems (DBS) that use commercially-available products with minimal (source code) modification
A6	Plan, budget, fund, and manage software development as an enduring capability that crosses program elements and funding categories, removing cost and schedule triggers associated with hardware-focused regulations and processes.
A7	Replace JCIDS, PPBE, and DFARS with a "PEO Digital" in each Service that uses portfolio management and direct identification of warfighter needs to decide on allocation priorities for software capabilities.
B4	Prioritize secure, iterative, collaborative development for selection and execution of new software development programs (and software components of hardware programs), especially those using commodity hardware and operating systems.
B5	Remove obstacles to DoD usage of cloud computing on commercial platforms, including DISA CAP limits, lack of ATO reciprocity, and access to modern software development tools

B6	Shift from certification of executables for low and medium risk deployments to certification of code/architectures and certification of the development, integration, and deployment toolchain.
B7	Plan and fund computing hardware (of all types) as consumable resources, with continuous refresh and upgrades to the most recent, most secure OS and platform components
C3	Increase the knowledge, expertise, and flexibility in program offices related to modern software development practices to improve the ability of program offices to take advantage of software-centric approaches to acquisition
C4	Restructure the approach to recruiting digital talent to assume that the average tenure of a talented engineering will be 2-4 years, and make better use of HQEs, IPAs, reservists and enlisted personnel to provide organic software development capability, while at the same time incentivizing and rewarding internal talent.
D4	Create and use automatically generated, continuously available metrics that emphasize speed, cycle time, security, and code quality to assess, manage, and terminate software programs (and software components of hardware programs)
D5	Shift the approach for acquisition and development of software (and software-intensive components of larger programs) to an iterative approach: start small, be iterative, and build on success - or be terminated quickly.
D6	Maintain an active research portfolio into next-generation software methodologies and tools, including the integration of machine learning and AI into software development, cost estimation, security vulnerabilities and related areas
D7	Invest in transition of emerging approaches from academia and industry to creating, analysis, verification, and testing of software into DoD practice (via pilots, field tests, and other mechanisms)
D8	Automatically collect all data from DoD weapons systems and make available for machine learning (via federated, secured enclaves, not a centralized repository)

5.3 Monitoring and Oversight of the Implementation Plan

It would be naive to believe that just listing the recommendations above will somehow make them quickly and easily implemented after 37 years of previous, largely consistent recommendations have had relatively minor impact. We believe that DoD should use these recommendations (and the ones that preceded them) to create an implementation plan for review by stakeholders (including the DIB, if there is interest). This implementation plan might use as its starting point the proposed implementation plans that we have articulated in Appendix R, with agreement by the Secretary of Defense, the Undersecretaries of Defense, the Service Chiefs, CAPE and DOT&E to support the creation and execution of the implementation plan within 60 days of delivery of this report to Congress.

We propose the following timeline for implementing the recommendations proposed here:

- Within 60 days after delivery of this report to Congress: Define a detailed implementation plan and assign owners for each of the top recommendations to begin right now.
- FY19 (create): High-level endorsement of the vision of this report, and support for activities that are consistent with the desired end state (i.e., DevSecOps and enterprise-level architecture and infrastructure); identify and launch programs to move out on the priority recommendations (start small, iterate quickly).
- FY20 (deploy): Initial deployment of authorities, budgets, and processes for software acquisition and practices reform. Execute representative programs according to the lines of effort

and recommendations in this report. Implement this report in the way we implement modern software: implement now, measure results, and modify approaches.

- FY21 (scale): Streamlined authorities, budgets, and processes enabling software acquisition and practices reform at scale. In this time frame, we need a new methodology to estimate as well as determine the value of software capability delivered (and not based on lines of code).
- FY22 (optimize): All DoD software development projects transition (by choice) to software-enabled processes, with talent and ecosystem in place for effective management and insight.

5.4 Kicking the Can Down The Road: Other Things That We Could Not Figure Out How to Fix

Despite the fairly comprehensive view that we have attempted to take in this study regarding how to improve the defense software enterprise, there are a number of challenges remaining that we were not able to address. We summarize these here for the next study (or perhaps one 37 years from now) to consider as they continue this path forward.

Over-oversight. The Department of Defense's sprawling software enterprise has many oversight actors, spanning the Congress, the Office of the Secretary of Defense, Service or Component leadership, and other executive branch actors like the Government Accountability Office. These actors each take frequent oversight action in attempts to improve the software in specific programs and also make well intended efforts to improve the health of the overall system. However, these oversight actions focus primarily on addressing the behavior of the people developing and maintaining the software, overlooking the fact that the oversight itself is equally part of the DoD's software problem. Ultimately, we can't fix software without fixing oversight.

There are at least two categories of problems when it comes to software oversight: structural and substantive.

From a structural perspective, there are too many actors involved in oversight. A program manager, tasked with leading a software development effort, can have as many as 17 other actors who can take some form of oversight action on their program. Most of these individuals do not possess the authority to cancel a program unilaterally but all have the ability to delay or create uncertainty while seeking corrective action for their concerns. These oversight actors often have overlapping or unclear roles and authorities, as well as competing interests and incentives. This means that in addition to the necessary checks and balances required between organizations, there is debate and active competition inside each of the organizations with, for example, various offices in OSD arguing among themselves in addition to arguing with Congress and the Services. Further, there is significant personnel turnover within these positions, meaning that any consensus tends to be short lived.

Substantively, the various oversight actors do not possess a shared understanding of what constitutes good practice for software or its oversight. Relatedly, these actors do not share a common vision for what the DoD's software enterprise should look like today or in the future. The majority of oversight attention and action is placed on individual programs rather than considering portfolios in aggregate or the performance of the system as a whole. This program oversight is highly subjective in nature, relying on reports and PowerPoint slides composed of narratives and custom

created data. Worse, this oversight operates primarily on the conventional wisdom associated with the oversight of hardware programs, considering cost, schedule, and performance using decades old heuristics.

Without understanding what good looks like, or the right questions to ask, oversight actors risk enacting poor fixes. These actions can also be at odds with stated policy. Oversight actions are always more powerful than written policy, meaning that disparities between the two create the risk of cognitive dissonance or a shadow policy environment. Disparities also put program leadership in the unfair position of having to resolve the competing priorities of others, with the knowledge that failure to do so will lead to more blame and action from above.

Structural and substantive problems lead to oversight that is inconsistent and confusing, making it essentially impossible to systematically identify symptoms, determine root causes, or implement scalable fixes. This, in turn, allows everyone involved in DoD software development and maintenance to feel aggrieved, blame everyone other than themselves for systemic issues, and continue their behavior without reflection or change, thus perpetuating the cycle.

The approach by oversight organizations both on the Hill and in the DoD should be that policy is treated as the current hypothesis for how best to ship code that DoD's users need. Through the use of data driven governance, each program should then be tested against that policy while also being a test of the policy. The hypothesis, and policy, must be continually updated based on standard data that is recognized by, and accessible to, all oversight actors. Implementing such an approach is within the power of the oversight community but would be challenging and appears unlikely given current culture and practices. Regardless, those involved in the oversight of DoD software should not expect meaningfully improved outcomes for that software until the oversight practices used to improve that software are themselves improved.

Promotion practices. Software is disproportionately talent-driven. Access to strong engineering talent is one of the most important factors that determine the success or failure of software projects. All that our rivals have to do to surpass us in national security applications of software such as AI, autonomy, or data analytics, is to leverage their most talented software engineers work on those applications. And yet in DoD, as much as we struggle to attract those technical talent, we also struggle to elevate the talent we have.

The companies and institutions who are winning the software game recognize the importance of identifying and cultivating talented software leaders (whether they are engineers, management, or strategists working closely with contractors) and actively promote and reward employees based on merit and demonstrated contribution. In contrast, human capital practices in DoD, sometimes by design and sometimes by habit and culture, narrowly limit how technical talent can be evaluated to time in grade. The Department needs to figure out how to recognize when civilians and service members show an aptitude for software and software management, and be able to promote, reward, and retain these individuals outside of the current constraints.

Using commercial software whenever possible. DoD should not build something that it can buy. If there is an 80 percent commercial solution, it is better to buy it and adjust—either the requirements or the product—rather than build it from scratch. It is generally not a good idea to over-

optimize for what we view as "exceptional performance,"[16] because counter-intuitively this may be the wrong thing to optimize for as the threat environment evolves over time. Similarly, actions should be taken to ensure that the letter and spirit of commercial preference laws (e.g., 10 USC 2377, which requires defense agencies to give strong preference to commercial and non-developmental products) are being followed.

There is a myth that the U.S. private sector—where much of the world's software talent is concentrated—is unwilling to work on national security software. The reality is that DoD has failed to give meaningful government contracts to commercial software companies, which has generally led to companies making a *business decision* to avoid it. DoD's existing efforts to target the commercial software sector are governed by a "spray and pray" strategy, rather than by making concentrated investments.[17] The DoD seems to love the idea of innovation, but doesn't love taking sizeable bets on new entrants or capabilities. It is interesting to note that Palantir and SpaceX are the only two examples since the end of the Cold War of venture-backed, DoD-focused businesses reaching multi-billion dollar valuations. By contrast, China has minted around a dozen new multi-billion dollar defense technologies companies over the same time period. Some of these problems are purely cultural in nature and require no statutory/regulatory changes to address. Others likely will require changes we list in the recommendations below.

That said, in many cases, there will not be an obvious "buy" option on the table. DoD and the Services should also work together to prioritize interoperable approaches to software and systems that enable rapid deployment, scaling, testing, and optimization of software as an enduring capability; manage them using modern development methods; and eliminate selected hardware-centric regulations and other particularly problematic barriers. The Services should find ways to better recognize software as a key area of expertise and provide specialized education and organizational structures that are better tuned for rapid insertion and continuous updates of software in the field and in the (back) office.

[16] From the 2018 Summary of the National Defense Strategy: *Deliver performance at the speed of relevance. Success no longer goes to the country that develops a new technology first, but rather to the one that better integrates it and adapts its way of fighting. Current processes are not responsive to need; the Department is over-optimized for exceptional performance at the expense of providing timely decisions, policies, and capabilities to the warfighter. Our response will be to prioritize speed of delivery, continuous adaptation, and frequent modular upgrades. We must not accept cumbersome approval chains, wasteful applications of resources in uncompetitive space, or overly risk-averse thinking that impedes change. Delivering performance means we will shed outdated management practices and structures while integrating insights from business innovation.*

[17] While the overall funding commitments are large—$2 billion dollars from DARPA for AI, for example—those commitments have resulted in few, if any, contracts for private companies other than traditional defense contractors. They have therefore failed to create significant incentives for the commercial tech sector to invest in government applications of AI.

Software is Never Done:
Refactoring the Acquisition Code for Competitive Advantage

Defense Innovation Board

SUPPORTING INFORMATION[18]

v2.0, 12 Mar 2019

This document contains the supporting information for the Defense Innovation Board (DIB) Software Acquisition and Practices (SWAP) study. This information is in preliminary form and should be read along with the main (draft) report.

Contents:

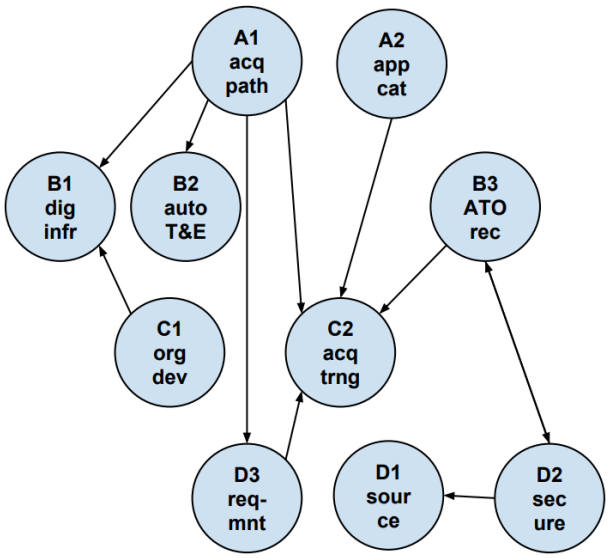
[18] Page numbers reflect this document and "TBD" for docs that were either previously released, released separately, or will be included in subsequent releases.

Appendix A: Draft Implementation Plan (Recommendation Summaries)

v1.0, 12 Mar 2019

The following pages contain summaries for each recommendation that give more detail on the rationale, supporting information, similar recommendations, specific action items, and notes on implementation. The beginning of each recommendation summary includes the recommendation statement, proposed owner, background information, description of the desired state, proposed role for Congress, and a short "action plan" describing how the recommendation might be implemented. The remainder of the summary contains a list of recommendations from the DIB Guides (contained in Appendix E of the supporting information), a list of recommendations from the working group reports (Appendix F of the supporting information), and some related recommendations from previous reports.

The recommendations listed here are relatively decoupled, but there are some dependencies between them, as shown to the right. In this figure, an arrow leading from one recommendation toward a second recommendation means that the first implementation depends at least somewhat on the implementation of the second. Hence by choosing one recommendation and following the arrows, the list of all recommendations that should also be implemented can be obtained.



Primary Recommendation A1 – New Acquisition Pathway

Line of Effort	Refactor statutes, regulations, and processes for software
Recommendation	**Establish new acquisition pathway(s) for software that prioritizes continuous integration and delivery of working software in a secure manner, with continuous oversight from automated analytics**
Stakeholders	A&S, HASC/SASC, USD(C), CAPE, DOT&E, R&E/DT, SAE, Service FM & PA&E, Joint Staff
Background	Current law, regulation, and policy, and internal DoD processes make DevSecOps software development extremely difficult, requiring substantial and consistent senior leadership involvement. Consequently, DoD is challenged in its ability to scale DevSecOps software development practices to meet mission needs.
Desired State	Tailored software-specific pathways that provide guidance to acquisition professionals to navigate the acquisition and requirements lifecycle to rapidly deliver capabilities. Each pathway streamlines the processes, reviews, and documents based on the type of IT/SW capability. Programs choosing these pathways have the ability to rapidly field and iterate new functionality in a secure manner, with continuous oversight based on automated reporting and analytics, and utilizing IA-accredited commercial development tools. Rapid acquisition authority available for software already in use and accredited - especially when purchased as a capability delivery (as a service). Over time, this becomes the default choice for software and software-intensive programs/program elements.
Congressional Role	This should become the primary pathway that DoD chooses to use for software and software-intensive programs and should provide Congress with the insight required to oversee software projects that move at a much faster pace than traditional HW programs, with traditional metrics and milestones replaced by more software-compatible measures of progress.

Draft Implementation Plan		Lead Stakeholder	Target Date
A1.1	(optional) Submit legislative proposal using Sec 805 to propose new acquisition pathways for two or more classes of software (e.g., application, embedded), optimized for DevSecOps	USD(A&S), in coordination with USD(C) and CAPE	Q3 FY19
A1.2	Create new acquisition pathway(s) for two or more classes of software, optimized for DevSecOps (based on A2c.1 or Appendix B.1)	HASC, SASC	FY20 NDAA
A1.3	Develop and issue a Directive Type Memorandum (DTM) for the new software acquisition pathway	USD(A&S)	Q1 FY20
A1.4	Issue Service level guidance for new acquisition pathway	SAE	Q2 FY20
A1.5	Select pilot programs using DevSecOps to convert to or utilize new SW acquisition pathway	USD(A&S), with SAEs	Q2 FY20
A1.6	Develop and implement training at Defense Acquisition University on new software acquisition pathway for all acquisition communities (FM, Costing, PM, IT, SE, etc.)	USD(A&S)	Q3 FY20

A1.7	Convert DTM to DoD Instruction, incorporating lessons learned during pilot program implementation	USD(A&S)	Q4 FY20

SWAP working group inputs related to this recommendation

Acq	Define software as a critical national security capability under Section 805 of FY16 NDAA “Use of Alternative Acquisition Paths to Acquire Critical National Security Capabilities”.
Acq	Create an acquisition policy framework that recognizes that software is ubiquitous and will be part of all acquisition policy models.
Acq	Create a clear, efficient acquisition path for acquiring non-embedded software capability. Deconflict supplemental policies.
Acq	Develop an Enterprise-level Strategic Technology Plan that reinforces the concept of software as a national security capability and recognizes how disruptive technologies will be introduced into the environment on an ongoing basis
Acq	Additionally, take all actions associated with Rec A2a to refactor and simplify those parts of Title 10, DoD 5000 and other regulations and processes that are still in force for software-intensive programs.

Related recommendations from previous studies

DSB87	Rec 13: The Undersecretary of Defense (Acquisition) should adopt a four-category classification as the basis of acquisition policy [standard (COTS), extended (extensions of current systems, both DoD and commercial), embedded, and advanced (advanced and exploratory systems)]
DSB87	Rec 14: USD(A) should develop acquisition policy, procedures, and guidance for each category.
GAO'17	Prioritize investments so that projects can be fully funded and it is clear where projects stand in relation to the overall portfolio.
DSB’09	The USD (AT&L) should lead an effort, in conjunction with the Vice Chairman, Joint Chiefs of Staff, to develop new, streamlined, and agile capabilities (requirements) development and acquisition processes and associated policies for information technology programs

Primary Recommendation A2 – New Appropriation Category

Line of Effort	Refactor statutes, regulations, and processes for software
Recommendation	**Create a new appropriations category that allows (relevant types of) software to be funded as a single budget item, with no separation between RDT&E, production, and sustainment.**
Stakeholders	A&S, HAC-D/SAC-D, HASC/SASC, USD(C), CAPE, SAE, Service FM & PA&E, FASAB, OMB
Background	Current law, regulation, and policy treat software acquisition as a series of discrete sequential steps; accounting guidance treats software as a depreciating asset. These processes are at odds with software being continuously updated to add new functionality and create significant delays in fielding user-needed capability.
Desired State	Programs are better able to prioritize how effort is spent on new capabilities versus fixing bugs / vulnerabilities, improving existing capabilities, etc. Such prioritization can be made based on warfighter / user needs, changing mission profiles, and other external drivers, not constrained by available sources of funding.
Congressional Role	This should become the primary pathway that Congress uses to fund software and software-intensive programs and should provide Congress with the insight required to oversee software projects that move at a much faster pace than traditional HW programs, with traditional metrics and milestones replaced by more software-compatible measures of progress.

	Draft Implementation Plan	Lead Stakeholder	Target Date
A2.1	(optional) Submit legislative proposal using Sec 805 to create a new appropriations category for software and software-intensive programs	USD(A&S), with USD(C) and CAPE	Q3 FY19 for FY20 NDAA
A2.2	Create new appropriation category for software-intensive programs, with appropriate reporting and oversight for software (based on Action A2.1 or Appendix B.1)	HAC-D, SAC-D, with OSD, HASC, SASC	FY20 NDAA, FY20 budget
A2.3	Select initial programs using DevSecOps to convert to or use new SW Appropriation in FY20	USD(A&S), with Service Acquisition Executives	Q4 FY19
A2.4	Define budget exhibits for new SW appropriation (replacement for P- and R-forms; see App C)	USD(A&S), with USD(C), CAPE, HAC-D, SAC-D	Q4 FY19
A2.5	Change audit treatment of software with these goals: (1) separate category for software instead of being characterized as property, plant, and equipment; (2) default setting that software is an expense, not an investment; and (3) there is no “sustainment” phase for software.	FASAB, with USD(A&S) and USD(C)	End FY20
A2.6	Make necessary modifications in supporting PPB&E systems to allow use and tracking of new software appropriation	USD(C) and CAPE	Q1 FY21

A2.7	Ensure programs using new software appropriation submit budget exhibits in the approved format.	SAE with USD(C), CAPE	FY 22 POM

SWAP concept paper recommendations related to this recommendation

10C	Budgets should be constructed to support the full, iterative life-cycle of the software being procured with amount proportional to the criticality and utility of the software.
Visits	Construct budget to support the full, iterative life-cycle of the software

SWAP working group inputs related to this recommendation

Acq	Revise 10 USC 2214 to allow funding approved by Congress for acquisition of a specific software solution to be used for research and development, production, or sustainment of that software solution, under appropriate conditions.
App	A new multi-year appropriation for Digital Technology needs to be established for each Military Defense Department and the Fourth Estate.
App	Components will program, budget, and execute for information and technology capabilities from one appropriation throughout lifecycle rather than using RDT&E, procurement, or O&M appropriations -- often applied inconsistently and inaccurately -- allowing for continuous engineering
Con	Congress establishes new authority for contracting for SW development and IT modernization
M&S	Revise 10 USC 2460 to replace the “software maintenance” with “software sustainment” and use a definition that is consistent with a continuous engineering approach across the lifecycle
M&S	A DoD Working Group should be established to leverage on-going individual Service efforts and create a DoD contracting and acquisition guide for software and software sustainment patterned after the approach that led to creation of the DoD Open Systems Architecture Contracting Guide
M&S	Acquisition Strategy, RFP/Evaluation Criteria, and Systems Engineering Plan should address software sustainability and transition to sustainment as an acquisition priority.
Con	Manage programs at budget levels, allow programs to allocate funds at project investment level
Con	Work with appropriators to establish working capital funds so that there is not pressure to spend funds quicker then you're ready (iterative contracts may produce more value with less money

Related recommendations from previous studies

GAO15	3. Assigning resources to all activities. The schedule should reflect the resources (labor, materials, travel, facilities, equipment, and the like) needed to do the work, whether they will be available when needed, and any constraints on funding or time.
GAO17	Hold suppliers accountable for delivering high-quality parts for their products through activities including regular supplier audits and performance evaluations of quality and delivery.
CSIS18	Performance Based Logistics (PBL) contracts should have a duration that allow for tuning and re-baselining with triggered options and rolling extensions.
809	Rec. 41: Establish a sustainment program baseline, implement key enablers of sustainment, elevate sustainment to equal standing with development and procurement, and improve the defense materiel enterprise focus on weapon system readiness.
809	Rec. 42: Reduce budgetary uncertainty, increase funding flexibility, and enhance the ability to effectively execute sustainment plans and address emergent sustainment requirements.

Additional Recommendation A3 – Cost Assessments and Performance Estimates

Line of Effort	Refactor statutes and regulations for software
Recommendation	Require cost assessment and performance estimates for software programs (and software components of larger programs) of appropriate type be based on metrics that track speed and cycle time, security, code quality, and functionality.
Stakeholders	CMO, USD(A&S), Service CMOs and SAEs
Background	Current software cost estimation and reporting processes and procedures in DoD have proven to be highly inaccurate and time consuming. New metrics are required that match the DevSecOps approach of continuous capability delivery and maintenance and provide continuous insight into program progress.
Desired State	Program oversight will re-focus on the value provided by the software as it is deployed to the warfighter/user, and will rely more heavily on metrics that can be collect in a (semi-)automated fashion from instrumentation on the DevSecOps pipeline and other parts of the infrastructure. Specific metrics will depend on the type of software rather than a one-size-fits-all approach.
Congressional Role	Congress needs to emphasize the need for new software acquisition reporting that focuses on value provided for the investment in software, and frequency of deployments to the warfighter/user. Congress needs to work with CAPE and USD(A&S) to provide feedback on meaningful content and level of detail in reporting.

Draft Implementation Plan		Lead Stakeholders	Target date
A3.1	Hire a small team (3-4) programmers to implement required software and provide them with a modern development environment	CAPE, DDS	Q4 FY19
A3.2	Identify low-level metrics that are already part of standard commercial development environments (see Appendix C for reporting form and Appendix E.2 (DIB Metrics) for initial lists)	CAPE, SAO	MVP Q4 FY19, then quarterly
A3.2a	Speed and cycle time: launch → initial use, cycle time	Dev team, users	
A3.2b	Code quality: unit test coverage, bug burn-rate, bugs-in-test:bugs-in-field	Dev team, users	
A3.2c	Security: patch → field, OS upgrade → field, HW/OS age	Dev team, users	
A3.2d	Functionality: user satisfaction, number/type of features/cycle	Dev team, users	
A3.2e	Cost: head count, software license cost, compute costs	Dev team, users	
A3.3	Identify 3-5 ongoing programs that are collecting relevant metrics and are willing to partner with CAPE	CAPE, A&S, CMO, SAEs	In parallel with A6.2
A3.4	Create a mechanism to transfer and process low-level metrics to PMO on a continuous basis with selectable levels of resolution across the program	CAPE, SAO, PMO	MVP Q4 FY19, then quarterly

A3.5	Begin reporting metrics to Congress as part of annual reporting; iterate on content, level, format	CAPE, Comp, A&S	FY2020
A3.6	Use initial results to establish expectations for new proposed software or software-intensive projects and integrate use of new cost and performance estimates into contract selection	A&S, SAO, CAPE	FY2020
A3.7	Establish ongoing capability within CAPE to update metrics on continuous basis, with input from users (of the data)	CAPE	FY2021
A3.8	Identify and eliminate remaining uses of ESLOC as metric for cost and schedule estimation of software/software-intensive programs	CAPE, SAEs	FY2022

SWAP working group inputs related to this recommendation

Con	Revise estimation models - source lines of code are irrelevant to future development efforts, estimations should be based on the team size and investment focused (Cultural)

Related recommendations from previous studies

DSB87	Rec 12: Use evolutionary acquisition, including simulation and prototyping, as discussed elsewhere in this report, to reduce risk.
SEI'01	Effort Estimation: • Utilize most likely effort estimates in proposals and status reports'; • Find ways to promote the use of accurate effort estimation and productivity evaluation; • Lowest cost is not equivalent to best value. Question outliers.
OSD'06	Adjust program estimates to reflect "high confidence"—defined as a program with an 80 percent chance of completing development at or below estimated cost—when programs are baselined in the Stable Program Funding Account.
SEI'10	Don't require PMO to adopt contractors' estimate for the program—or else use the difference as PM "reserve"
SEI'10	Change from traditional 50% estimation confidence level to 80% level
SEI'10	DoD should consider use of Vickrey "second price" auction mechanism for acquisition proposal bidding
SEI'15	Use the government's cost estimates (using say an 80% confidence level) rather than contractors' estimates as the basis for program budgets and place the difference (if the government's estimate is larger) in a reserve fund available to program managers with sufficient justification. Contractors' estimates should be acquired using mechanisms that promote accurate estimates, e.g., using Vickrey auctions, the Truth-Revealing Incentive Mechanism (TRIM), or more standard methods of review and acceptance by independent third parties.
DSB18	Rec 3b: The MDA with the Cost Assessment and Program Evaluation office (CAPE), the USD(R&E), the Service Cost Estimators, and others should modernize cost and schedule estimates and measurements.
DSB18	Rec 3b.1: [DoD] should evolve from a pure SLOC approach to historical comparables as a measurement, and should adopt the National Reconnaissance Office (NRO) approach (demonstrated in Box 5) of contracting with the defense industrial base for work breakdown schedule data to include, among others, staff, cost, and productivity.
DSB18	Rec 3c: The MDA should immediately require the PM to build a program-appropriate framework for status estimation.

Additional Recommendation A4 – Simplify Laws and Policies

Line of Effort	Refactor statutes and regulations for software
Recommendation	Refactor and simplify Title 10, DFARS, and DoDI 5000.02/5000.75 to remove statutory, regulatory, and procedural requirements that generate delays for acquisition, development, and fielding of software while adding requirements for continuous (automated) reporting of cost, performance (against updated metrics), and schedule.
Stakeholders	USD(C), CAPE, SAE, Service FM & PA&E, Joint Staff
Background	Current law, regulation, and policy, and internal DoD processes make Agile SW development extremely difficult, requiring substantial and consistent senior leadership involvement. Consequently, DoD is challenged in its ability to scale Agile SW development practices to meet mission needs.
Desired State	Programs have the ability to rapidly field and iterate new functionality in a secure manner, with continuous oversight based on automated reporting and analytics, and utilizing IA-accredited commercial development tools.
Congressional Role	Change 10 USC § 2341a

Draft Implementation Plan		Lead Stakeholders	Target Date
A4.1	Submit legislative proposal to simplify Title 10 for software	USD(A&S)	Q3 FY19
A4.2	Convene working group with stakeholders and develop and issue a Directive Type Memorandum (DTM) for the new simplified SW acquisition process	USD(A&S)	Q1 FY20
A4.3	Issue Service level guidance for new simplified SW acquisition process	SAE	Q1 FY20
A4.4	Identify pilot programs using Agile to convert to or utilize new simplified SW acquisition process	USD(A&S), in coordination with Service Acquisition Executives	Q1 FY20
A4.5	Convert DTM to DoD Instruction, incorporating lessons learned during pilot program implementation.	USD(A&S)	Q1 FY20
A4.6	Develop and implement training at Defense Acquisition University on new simplified SW acquisition process for all acquisition communities (FM, Costing, PM, IT, SE, etc.)	USD(A&S)	Q1 FY20

SWAP working group inputs related to this recommendation

Acq	Ensure appropriate integration of a data strategy and the Department's Cloud Strategy. Examine a Steering Committee approach for management. [dup]
Acq	Examine the organizational structure with the intent of achieving a more responsive and flat organizational model that de-conflicts roles and responsibilities between the DoD CIO, the USD(A&S) and the CMO regarding software.
Acq	Re-focus the software acquisition workforce on teaming and collaboration, agility, improved role definition, career path advancement methods, continuing education and training opportunities, incentivization, and empowerment.
Acq	Increase flexibility and agility for software programs by eliminating mandated content for acquisition strategies and authorities in Section 821 of the FY16 NDAA except for MDAPs.
Acq	Eliminate hardware-centric cost, fielding and performance goals in 10 USC 2488 (established by Sec 807 of the FY17 NDAA) for software-intensive programs.
Acq	Eliminate Nunn-McCurdy breaches (10 USC 2433) for software-intensive programs and replace

	with continuous evaluation of software performance metrics.
Acq	Remove statutory definition of "major system" for software-intensive programs in 10 USC 2302 and 2302d to remove confusion since most software in weapons systems inherently functions together to fulfill a mission need.
Acq	Develop language for 10 USC 2366a that allows exemption for software-intensive programs, where DOT&E must justify adding the program for oversight with the MDA and must streamline the process.
Acq	Only require DOT&E oversight for software-intensive programs when requested by the SAE, USD(A&S) or Congress, or if the program is an MDAP.
Acq	For the 4th estate, combine all three authorities for DBS under the DoD CMO. After one year conduct assessment and make a determination if this should be applied to the Services as well.
Acq	Eliminate the separate annual funding certification process for defense business system from 10 USC 2222 or require that funding certification be merged in to the PPBE process
Acq	Replace annual configuration steering board (CSBs) for software-intensive programs with board (or equivalent entities) established by the CAE, PEO, or PM [FY09 NDAA Sec 814; DoDI 5000.02]
Acq	Expand the FAR 39 (Acquisition of IT) to allow for one area to drive technology purchases. Unless otherwise stated, no other FAR rules would apply
Acq	Rewrite FMR Volume 2A, Chapter 1, Section 010212(B) to [1] acknowledge that, for the purpose or modifying or enhancing software, there is no technically meaningful distinction between RDT&E, Procurement, and O&M; [2] eliminate the $250,000 barrier between expenses and investments (i.e., stop explicitly tying to a dollar threshold the determination of whether software is an expense or an investment.
Acq	Revise or eliminate DoDI 8330.01 to eliminate the following elements for software-intensive programs: [1] NR KPP required; [2] DoD specific architecture products in the DoDAF format that are labor intensive and of questionable value; [e] Interoperability Support Plans (ISPs) required, where DoD CIO can declare any ISP of “special interest”; [2] requirement of DT authority to provide assessments at MS C; [5] mandates JITC to do interoperability assessments for IT with “joint, multinational, and interagency interoperability requirements”
Acq	Revise PfM policy (DoDD 7045.20) to consider the role of data and metrics, as well as additional portfolios (like NC3), and determine authority for the policy.
Con	Separate Contract requirements (scope, PoP, and price) from technical requirements (backlog, roadmap, and stories)
Con	Use SOO vs SOW to allow the vendor to solve the objectives how they are best suited
Con	Establish clear and intuitive guidelines on how and when to apply existing clauses
Con	Have standard clause applications for each of the above that must be excepted vs accepted
D&M	Congress could establish, via an NDAA provision, new data-driven methods for governance of software development, maintenance, and performance. The new approach should require on demand access to standard [and real-time?] data with reviews occurring on a standard calendar, rather than the current approach of manually developed, periodic reports. [dup]
M&S	Title 10 USC 2460 should be revised to replace the term software maintenance with the term software sustainment and definition that is consistent with a continuous engineering approach across the lifecycle [dup]

Req	The Joint Staff should consider revising JCIDS guidance to focus on user needs, bypassing the JCIDS process as needed to facilitate rapid software development. Guidance should specifically account for user communities (e.g. Tactical Action Officer (TAO), Maritime Operations Center (MOC) director) that do not have one specific PoR assigned to them, but use multiple systems and data from those systems to be effective
Req	The Joint Staff should consider revising JCIDS guidance to separate functionality that needs high variability from the functionality that deemed “more stable” (e.g. types of signals to analyze vs. allowable space for the antenna). Then implement a “software box” approach for each, one in which the contours of the box are shaped by the functionality variability
Req	The Joint Staff should consider revising JCIDS guidance to document stable concepts, not speculative ideas. The Joint Staff should consider revising JCIDS guidance to document stable concepts, not speculative ideas. Acknowledge that software requirement documents will iterate, iterate, iterate. JCIDS must change from a “one-pass” mentality to a “first of many” model that is inherently agile delegating approval to the lowest possible level

Related recommendations from previous studies

DSB87	Rec 21: DoD should examine and revise regulations to approach modern commercial practice insofar as practicable and appropriate.
NPS'16a	Program offices spend far too much time generating paperwork and navigating the bureaucracy rather than thinking creatively about program risks, opportunities, and key elements of their strategies
NDU'17	Develop and maintain core competencies in diverse acquisition approaches and increase the use of venture capital type acquisitions such as Small Business Innovative Research (SBIR), Advanced Concept Technology Development (ACTD), and Other Transaction Authority (OTA) as mechanisms to draw in non-traditional companies
NDU'17	Encourage employees to study statutes and regulations and explore innovative and alternative approaches that meet the statutory and regulatory intent
Sec 809	Rec. 62: Update the FAR and DFARS to reduce burdens on DoD’s commercial supply chain to decrease cost, prevent delays, remove barriers, and encourage innovation available to the Military Services.
Sec 809	Rec. 74: Eliminate redundant documentation requirements or superfluous approvals when appropriate consideration is given and documented as part of acquisition planning.
Sec 809	Rec. 75: Revise regulations, instructions, or directives to eliminate non-value added documentation or approvals.
Sec 809	Rec. 90: Reorganize Title 10 of the U.S. Code to place all of the acquisition provisions in a single part, and update and move acquisition-related note sections into the reorganized acquisition part of Title 10.

Additional Recommendation A5 – Streamlined Processes for Business Systems

Line of Effort	Refactor statutes and regulations for software
Recommendation	Create streamlined authorization and appropriation processes for defense business systems (DBS) that use commercially-available products with minimal (source code) modification.
Stakeholders	CMO, USD(A&S), Service CMOs, SAEs, DoD CIO
Background	Current DoD business processes are minimally standardized due to a high number of legacy systems that inhibit business process reengineering. In addition, solicitation for new business systems often insist on customization because DoD is "different", resulting in hard-to-maintain systems that become obsolete (and possibly insecure) quickly.
Desired State	DoD uses standard commercial packages for enterprise and business services, changing its processes to match those of large industries, allowing its systems to be updated and modified on a much faster cadence. The only specialized defense business systems should be those for which there is no commercial equivalent (to include cases in which minor modifications would be required) and there is a funded internal capability to maintain and update the software at a near-commercial cadence.
Congressional Role	

Draft Implementation Plan		Lead Stakeholders	Target Date
A5.1	Revise DBS certification process guidance	CMO, with USD(A&S), Service counterparts	Q1 FY20
A5.2	Select 4 projects for COTS implementation	CMO, with Service CMOs and business process owners	Q1 FY20
A5.3	Implement COTS opportunities, with contracts in place	Services, with CMO oversight	Q1 FY21
A5.4	Submit legislative change proposal (if Title 10 §2222 is a hindrance)	CMO, with USD(A&S) and Service counterparts	FY21

SWAP concept paper recommendations related to this recommendation

10C	Use commercial process and software to adopt and implement standard business practices within the services
D&D	For common functions, purchase existing software and change DoD processes to use existing apps

Related recommendations from previous studies

DSB87	Rec 15: The USD(A) and the ASD(Comptroller) should direct Program Managers to assume that system software requirements can be met with off-the-shelf subsystem and components until it is proved that they are unique.
Sec 809	Rec 16: Combine authority for requirements, resources, and acquisition in a single, empowered entity to govern DBS portfolios separate from the existing acquisition chain of command

Sec 809	Rec 18: Fund defense business systems (DBSs) in a way that allows for commonly accepted software development approaches

Additional Recommendation A6 – Enduring Capability

Line of Effort	Refactor statutes, regulations, and processes for software
Recommendation	Plan, budget, fund, and manage software development as an enduring capability that crosses program elements and funding categories, removing cost and schedule triggers associated with hardware-focused regulations and processes.
Stakeholders	USD(A&S), USD(C), SAE, Service FM, HASC, SASC
Background	The current approach to acquiring software is based on projects that have a beginning and end. However, many missions are "enduring capabilities" and need software program and portfolio management that continually and perpetually deliver across the spectrum of new capability, incremental enhancements and life-cycle sustainment. The Department should pilot and then scale methods for appropriating software budgets for these enduring capability programs as an ongoing, regularly evaluated expense, with continuous oversight, rather than large , multi-year development contracts
Desired State	The Department can manage software acquisition as an activity requiring continuous development, deployment and sustainment, recognizing that software systems are long-lived and have a continuous need for a level of activity to evolve capabilities and address vulnerabilities. Assessment of progress will be maintained throughout the software lifespan by means of continual user engagement with working software, rather than at large-scale milestone gates that do not map well to the underlying technical activities.
Congressional Role	N/A

Draft Implementation Plan		Lead Stakeholder	Target Date
A6.1	Modify FMR to implement this continuous funding approach	USD(C)	Q4 FY19
A6.2	Select and launch five programs to be managed as enduring capability two year pilot projects	USD (A&S) with SAE	Q4 FY19
A6.3	Work with FASAB to create an audit treatment of enduring capability software that; ● has a category distinct from Property, Plant, and Equipment ● defaults to treating software as an expense, not an investment; and ● does not distinguish between development and sustainment	USD(A&S) with USD(C)	Q4 FY20

SWAP concept paper recommendations related to this recommendation

10C	Budgets should be constructed to support the full, iterative life-cycle of the software being procured with amount proportional to the criticality and utility of the software.
D&D	Treat software development as a continuous activity, adding functionality continuously

Additional Recommendation A7 – Portfolio Management

Line of Effort	Refactor statutes, regulations, and processes for software
Recommendation	Replace JCIDS, PPB&E, and DFARS with a portfolio management approach to software programs, assigned to "PEO Digital" or equivalent office in each Service that uses and direct identification of warfighter needs to decide on allocation priorities for software capabilities.
Stakeholders	USD(A&S), CAPE, *JCS*, USD(C), SAE, Service FM & PAE
Background	The current requirements process often drives the development of exquisite requirements that tend to be overly rigid and specific, and attempt to describe the properties of systems in dynamic environments years in advance. The speed of requirements development and analysis is out of sync with the pace of technology and mission changes. Most importantly, requirement documents that are developed are often disconnected with the end user requirements.
Desired State	Software programs are managed using a portfolio approach, in which resources are available for reallocation across programs and funding categories based on the importance and opportunities of given elements of the portfolio.
Congressional Role	Congress should approve and monitor metrics of success defined within different portfolios and measure the progress against those metrics in determining allocations of funding to different portfolios (with the decisions within a portfolio made by the portfolio office and held accountable for those decisions).

Draft Implementation Plan		Lead Stakeholders	Target Date
A7.2	Select pilot capability areas in each service to place under portfolio management by PEO Digital (or equivalent)	SAEs	Q3 FY19
A7.1	Issue guidance for management of software portfolios with a “PEO Digital” or similar office with OSD and/or the Services.	USD(A&S) SAE	Q4 FY19
A7.3	Stand up PEO Digital or equivalent office with necessary resources allocated and aligned	SAE	Q1 FY20
A7.4	Implement new portfolio management methods for pilot program capability areas	PEO Digital	Q3 FY20
A7.5	Determine intermediate successes of, or required modifications to, portfolio management approach	PEO Digital	Q1 FY21
A7.6	Establish portfolio management approach as standard work for software	PEO Digital, SAE	FY22

SWAP working group inputs related to this recommendation

App	Within each Component-unique Budget Activity (BA), Budget Line Items (BLINs) align by functional or operational portfolios. The BLINs may be further broken into specific projects to provide an even greater level of fidelity. These projects would represent key systems and supporting activities, such as mission engineering.
App	By taking a portfolio approach for obtaining software intensive capabilities, the Components can better manage the range of requirements, balance priorities, and develop portfolio approaches to enable the transition of data to information in their own portfolios and data integration across portfolios to achieve mission effects, optimize the value of cloud technology, and leverage and transition to the concept of acquisition of whole data services vice individual systems.
App	This fund will be apportioned to each of the Military Departments and OSD for Fourth Estate execution.

App	Governance: management execution, performance assessment, and reporting would be aligned to the portfolio framework—BA, BLI, project.
Req	OSD and the Joint Staff should consider creating “umbrella” software programs around “roles” (e.g. USAF Kessel Run)

Related recommendations from previous studies

OSD'06	Transform the Planning, Programming, and Budgeting and Execution process and stabilize funding for major weapons systems development programs.
DSB’09	The USD (AT&L) aggressively delegate milestone decision authority commensurate with program risk
DSB’09	The USD (AT&L) consider a more effective management and oversight mechanism to ensure joint program stability and improved program outcomes
DSB’09	Consolidate all acquisition oversight of information technology under the USD (AT&L) by moving into that organization those elements of the OASD (NII)/DOD CIO and Business Transformation Agency responsible for IT acquisition oversight. The remainder of OASD (NII)/DOD CIO is retained as it exists today, but should be strengthened as indicated in the previous recommendation.
Sec 809	Rec 36: Transition from a program-centric execution model to a portfolio execution model
Sec 809	Rec 37: Implement a defense-wide capability portfolio framework that provides an enterprise view of existing and planned capability, to ensure delivery of integrated and innovative
Sec 809	Rec. 38: Implement best practices for portfolio management.
Sec 809	Rec. 39: Leverage a portfolio structure for requirements.

Primary Recommendation B1 – Digital Infrastructure

Line of Effort	Create and maintain cross-program/cross-service digital infrastructure
Recommendation	**Establish and maintain digital infrastructure within each Service or Agency that enables rapid deployment of secure software to the field and incentivize its use by contractors**
Stakeholders	A&S, CIO, SAE, CMO
Background	Currently, DoD programs each develop their own development and test environments, which requires redundant definition and provisioning, replicated assurance (including cyber), and extended lead times to deploy capability. Small companies have difficulties providing software solutions to DoD because those environments are not available outside the incumbent contractor or they have to build (and certify) unique infrastructure from scratch.
Desired State	Programs will have access to, and be stakeholders in, a cross-program, modern digital infrastructure that can benefit from centralized support and provisioning to lower overall costs and the burden for each program. Development infrastructure supporting CI/CD and DevSecOps is available as best of breed and GOTS provided so that contractors want to use it, though DoD programs or organizations that want or need to go outside of that existing infrastructure can still do so.
Congressional Role	Congress should track the availability, scale, use, and cost effectiveness of digital infrastructure, with the expectation that overall capacity will expand while unit costs decrease over time. Sufficient funding should be provided on an ongoing basis to maintain and upgrade digital infrastructure to maintain best of breed capability that accelerates software development.

	Implementation Plan	Lead Stakeholder	Target Date
B1.1	Designate organization(s) responsible for creating and maintaining the digital infrastructure for each Service's digital infrastructure. Explore the use of tiered approaches with infrastructure at Service or Program level, as appropriate.	DoD CIO, USD(C) and Services (SAE and Service CIO)	Q3 FY19
B1.2	Designate organization responsible for creating and maintaining digital infrastructure for DoD agencies and organizations, including joint digital infrastructure available to the Services.	USD(A&S), with CIO, CMO	Q3 FY19
B1.3	Provide resources for digital infrastructure, including cloud solutions, pre-approved "drop-ship" local compute capability, approved development environments (see DIB Compute Environment concept paper, Appendix I [Glossary])	USD(A&S), SAE with CAPE, USD(C)	FY20 budget
B1.4	Define baseline digital infrastructure systems and implement procurement and deployment processes and capability	Responsible organizations from B1.1, B1.2	Q2 FY20
B1.5	Implement digital infrastructure and provide access to ongoing and new programs.	Responsible organizations from B1.1, B1.2	Q3 FY20
B1.6	Identify acquisition programs to transition to digital infrastructure	SAE	Q2 FY20

B1.7	Transition programs to digital infrastructure	SAE, PEO, PM	Q4 FY20

SWAP concept paper recommendations related to this recommendation

10C	Make computing, storage, and bandwidth and programmers abundant to DoD developers and users.
D&D	Use validated software development platforms that permit continuous integration & delivery evaluation (DevSecOps platform)
Visits	Separate development of mission level software from development of IA-accredited platforms

SWAP working group inputs related to this recommendation

T&E	Build the enterprise-level digital infrastructure needed to streamline software development and testing across the full DoD software portfolio.

Related recommendations from previous studies

DSB87	Rec 16: All methodological efforts, especially STARS, should look to see how commercially available software tools can be selected and standardized for DoD needs.
SEI'01	Infrastructure: In distributed development activities, get high quality, secure, broadband communications between sites. It is an enabler, not a cost.

Primary Recommendation B2 – Automated Testing and Evaluation

Line of Effort	Create and maintain cross-program/cross-service digital infrastructure
Recommendation	**Create, implement, support, and use fully automatable approaches to testing and evaluation (T&E), including security, that allow high confidence distribution of software to the field on an iterative basis**
Stakeholders	DOT&E, USD(A&S), DDR&E(AC), SAE, Service Test Agencies
Background	To deliver SW at speed, rigorous, automated testing processes and workflows are essential. Current DoD practices and procedures often see OT&E as a tailgate process, sequentially after development has completed, slowing down delivery of useful software to the field and leaving existing (potentially poorly performing and/or vulnerable) software in place.
Desired State	Development systems, infrastructure and practices are focused on continuous, automated testing by developers (with users), with frequency dependent on type of software, but targets cycle times measured in weeks. To the maximum extent possible, system operational testing is integrated (and automated) as part of the development cycle using data, information and test protocols delivered as part of the development environment. Testing and evaluation/ certification of COTS components done once (if justified) and then ATO reciprocity (Rec B3) is applied to enable use in other programs, as appropriate.
Congressional Role	DOT&E should provide annual reports to Congress that describe the availability, scale, use, and effectiveness of automated T&E, with the expectation that level/depth of testing will increase at the same time as speed and cycle time are being improved.

Draft Implementation Plan		Lead Stakeholders	Target Date
B2.1	Establish procedures for fully automated testing on digital infrastructure (Rec B1), updating DoDI 5129.47 and Service equivalents.	USD(A&S), DOT&E, with Service Testers	Q1 FY20
B2.2	Establish processes for automated security testing, including zero-trust assumptions, automated penetration testing, and red teams for vulnerability scanning.	USD(A&S), DOT&E, with Service Testers	Q1 FY20
B2.3	Identify initial programs to use tools and workflows	SAE	Q1 FY20
B2.4	Implement minimum viable product (MVP) tools and workflows on digital infrastructure (Rec B1)	SAE, DOT&E, with PMOs	Q2 FY20
B2.5	Migrate initial programs to digital infrastructure using automated T&E	PEO, with Responsible Organizations	Q3 FY20
B2.6	Use tools and workflows, identify lessons learned and improvements (using DevSecOps iterative approach)	Service Testers, with PEO/PM	Q4 FY20
B2.7	Modify tools and workflows, document procedures	Responsible Organizations, Service Testers	Q4 FY20

SWAP concept paper recommendations related to this recommendation

10C	Automate testing of software to enable critical updates to be deployed in days to weeks, not months or years.
D&D	Create automated test environments to enable continuous (and secure) integration and deployment to shift testing and security left
Visits	Automate testing of software to enable critical updates to be deployed in days to weeks, not months or years (also requires changes in testing organization)
Visits	Add testing as a service

SWAP working group inputs related to this recommendation

Acq	DOT&E should use test data collected through existing test methodologies present in a software-intensive programs and not recommend or prescribe additional independent one-time test events.
Acq	One time IOT&Es or cybersecurity test events should not be recommended for software-intensive systems except in specific circumstances if warranted
T&E	Build the enterprise-level digital infrastructure needed to streamline software development and testing across the full DoD software portfolio. [dup]
T&E	DoD should expand DOT&E's current capability to obtain state-of-the-art cyber capabilities on a fee- for-service basis

Related recommendations from previous studies

DSB87	Rec 27: Each Service should provide its software Using Commands with facilities to do comprehensive operational testing and life-cycle evaluation of extensions and changes.
SEI'12	Merge Agile and security best practices (e.g., integrate vulnerability scans into continuous integration process, leverage automated test cases for accreditation validation, adhere to secure coding standards)
SEI'16	Employ concurrent testing and continuous integration.
USDS	When issuing a solicitation, it should explain the Agile software development process. The solicitation should also describe the required testing of functional requirements and make it clear that testing should be integrated into each sprint cycle
IDA'18a	Analysis of planned operational test lengths indicates that the test scope is generally not long enough demonstrate operational reliability with statistical confidence

Primary Recommendation B3 – ATO Reciprocity

Line of Effort	Create and maintain cross-program/cross-service digital infrastructure
Recommendation	**Create a mechanism for Authority to Operate (ATO) reciprocity within and between programs, Services, and other DoD agencies to enable sharing of software platforms, components and infrastructure and rapid integration of capabilities across (hardware) platforms, (weapons) systems, and Services.**
Stakeholders	DoD CIO, A&S, Service CIOs, DISA
Background	Current software acquisition practice emphasizes the differences among programs: perceptions around different missions, different threats, and different levels of risk tolerance mean that components, tools, and infrastructure that have been given permission to be used in one context are rarely accepted for use in another. The lack of ATO reciprocity drives each program to create their own infrastructure, repeating time- and effort-intensive activities needed to certify elements as secure for their own specific context.
Desired State	Modern software components, tools, and infrastructure, once accredited as secure within the DoD, can be used appropriately and cost-effectively by multiple programs. Programs can spend a greater percentage of their budgets on developing software that adds value to the mission rather than spending time and effort on basic software infrastructure. Accreditation of COTS components is done once and then made available for use in other programs, as appropriate.
Congressional Role	N/A

Draft Implementation Plan		Lead Stakeholder	Target Date
B3.1	Issue guidance making reciprocity the default practice in DoD with limited exceptions and update DoDI 8510.01 to reflect updated risk management framework. Exceptions should require signoff by the DoD CIO to discourage their use.	DoD CIO, with Service CIOs	Q3 FY19
B3.2	Establish DoD-wide repository for ATO artifacts with tools and access rules that enable Services to identify existing ATOs and utilize them when possible.	DoD CIO, with Service CIOs, DISA	Q4 FY19
B3.3	Implement procedures and access controls so that Authorizing Officials have visibility over other programs that are using compatible ATOs	DoD CIO, with Service CIOs, DISA	Q2 FY20
B3.4	Implement mechanisms to allow FedRAMP and other non-DoD security certifications to be used for DoD ATO when appropriate based on intended use and environment	DoD CIO, with FedRAMP	Q4 FY20

SWAP working group inputs related to this recommendation

Sec	As security is “baked in” to software during the development process, people must be educated about what that means as different tools look at different security aspects.
Sec	People must learn to appreciate that speed helps increase security. Security is improved when changes and updates can be made quickly to an application. Using automation, software can be reviewed quickly.
Sec	The AO must also be able to review documentation and make a risk decision quickly and make that decision on the process and not the product.

Related recommendations from previous studies

SEI'12	Define criteria for reaccreditation early in the project.
SEI'12	Leverage long accreditation approval wait time with frequent community previews.
SEI'12	Don’t apply all the information assurance controls blindly.

Additional Recommendation B4 – Prioritize Modern SW Development Methods

Line of Effort	Create and maintain cross-program/cross-service digital infrastructure
Recommendation	Prioritize secure, iterative, collaborative development for selection and execution of new software development programs (and software components of hardware programs), especially those using commodity hardware and operating systems.
Stakeholders	USD(A&S), USD(C) DOT&E, SAE, Service Test Agencies
Background	Despite 37+ years of recommendations to stop using waterfall development for software programs, DoD continues to make use of hardware-centric approaches to development for software and software-intensive programs. While portions of the DoD 5000.02 Instructions apply to "Defense Unique Software Intensive" programs and "Incrementally Deployed Software Intensive" programs, these are still waterfall processes with years between the cycles of deployments (instead of weeks). These processes may be appropriate for some (though not all) embedded systems, but are not the right approach for DoD-specific software running on commercial hardware and operating systems.
Desired State	DoD makes use of commercial software (without customization) whenever possible. When DoD-specific software development is required, contractors with demonstrated ability in the implementation of modern software development processes (eg, agile, DevOps, DevSecOps) are prioritized in the selection process and a contract structure is used that enables those methods to be successfully applied. For those applications for which hardware and software development are closely coupled, modern methods are still used as appropriate, especially in terms of information assurance testing.
Congressional Role	Congress should review metrics for performance on software (and software-intensive) programs with the expectation that modern methods of software able to deliver software to the field quickly, provide rapid and continuous updates of capability, perform extensive automated testing, and track metrics for speed and cycle time, correctness, and security.

Draft Implementation Plan		Lead Stakeholders	Target Date
B4.1	Establish metrics for evaluation of software development environments, following DSB 2018 recommendations on software factors and DIB "Development Environment" and "Agile BS Detector" Concept Papers	USD(A&S) with TBD	Q3 FY19
B4.2	Issue Directive-Type Memorandum (DTM) to specify DoD's default software development approach is secure, iterative, modular, and collaborative	USD(A&S)	Q3 FY19
B4.3	Update DoDI 5000.02 and 5000.75 to specify DoD's default software development approach is secure, iterative, modular, and collaborative	USD(A&S)	Q1 FY20
B4.4	Update courseware at Defense Acquisition University to specify DoD's default SW development approach is secure, iterative, modular, and collaborative	USD(A&S)	Q2 FY20

SWAP concept paper recommendations related to this recommendation

10C	Adopt a DevOps culture for software systems.
D&D	Require developers to meet with end users, then start small and iterate to quickly deliver useful code

Visits	Adopt a DevOps culture: design, implement, test, deploy, evaluate, repeat

SWAP working group inputs related to this recommendation

Con	Use collaborative tools and libraries so that all content is available to all parties at all times
Con	Use an agile process to manage structure and technical requirements
Sec	As security is “baked in” to software during the development process, people must be educated about what that means as different tools look at different security aspects.
Wkf	Incentivize defense contractors to demonstrate their ability to leverage modern software methodologies
Wkf	Contractor Reform. Adjust future NDAA’s to add incentives for defense contractors to use modern development practices. (See FY18NDAA / §§873 & 874)

Related recommendations from previous studies

DSB87	Rec 17: DoD should devise increased productivity incentives for custom-built software contracts, and much such incentivized contracts the standard practice.
DSB87	Rec 18: DoD should devise increased provide incentives on software quality.
DSB87	Rec 23: The USD(A) should update DoD Directive 5000.29, “Management of Computer Resources in Major Defense Systems”, so that it mandates the iterative setting of specifications, the rapid prototyping of specified systems, and increment development.
DSB87	Rec 24: DoD STD 2167 should be further revised to remove any remaining dependency on the assumptions of the “waterfall” model and to institutionalize rapid prototyping and incremental development
DSB87	Rec 29: The USD(A) should develop economic incentives, to be incorporated into standard contracts, to allow contractors to profit from offering modules for reuse, even though built with DoD funds.
DSB87	Rec 30: The USD(A) should develop economic incentives, to be incorporated into all cost-plus standard contracts, to encourage contractors to buy modules and use them rather than building new ones.
DSB87	Rec 31: The USD(A) and ASD(Comptroller) should direct Program Managers to identify in their programs those systems, components, and perhaps even modules, that may be expected to be acquired rather than built; and to reward such acquisition in the RFP’s.
SEI'12	Make sure Agile project teams understand the intent behind security requirements and organize the backlog accordingly
SEI'12	Ensure Agile development processes produce and maintain “just enough” design documentation.
SEI'12	Make sure there is at least one person with strong security analysis expertise on the Agile project team
SEI'12	Foster Agile project team and accrediting authority collaboration.
SEI'12	Leverage unclassified environments for Agile development and community previews.
SEI'12	Agile and the information assurance community must join forces to continue improving information assurance processes.
GAO'16a	Establish a department policy and process for the certification of major IT investments’ adequate use of incremental development, in accordance with OMB’s guidance on the implementation of FITARA.
NPS'16a	Systems leveraging open architectures and incremental designs can focus on delivering initial

	capability quickly, and then iterate improvements over time. The DoD can tailor acquisition processes for each major type of system to streamline each program's path through focused guidance
SEI'16	Ensure that the RFP contains language that allows the use of Agile. One promising approach that is consistent with Agile is to make sure the original contract is written with Agile in mind and contains sufficient flexibility to permit a wide scope of activity that could be modified as the situation develops. Agile program managers (PMs) could establish contract vehicles that allow for collaborative discussions to resolve and address dynamic developments over the life of the effort.
DSB18	Requests for proposals (RFPs) for acquisition programs entering risk reduction and full development should specify the basic elements of the software framework supporting the software factory, including code and document repositories, test infrastructure, software tools, check-in notes, code provenance, and reference and working documents informing development, test, and deployment
DSB18	Rec 1: A key evaluation criterion in the source selection process should be the efficacy of the offeror's software factory.
DSB18	Rec 1a: Establish a common list of source selection criteria for evaluating software factories for use throughout the Department
DSB18	Rec 1b: Competing contractors should have to demonstrate at least a pass-fail ability to construct a software factory
DSB18	Rec 1c: Criteria for evaluating software factories should be reviewed and updated every five years.
DSB18	Rec 5e: Defense prime contractors must build internal competencies in modern software methodologies.
DSB18	Rec 2: The DoD and its defense industrial base partners should adopt continuous iterative development best practices for software, including through sustainment.
DSB18	Rec 2c: [DoD should] engage Congress to change statutes to transition Configuration Steering Boards (CSB) to support rapid iterative approaches (Fiscal Year (FY) 2009 National Defense Authorization Act (NDAA), Section 814).
DSB18	Rec 2d: [DoD] should require all programs entering Milestone B to implement these iterative processes for Acquisition Category (ACAT) I, II, and III programs.
DSB18	Rec 4a: For ongoing development programs, the USD(A&S) should immediately task the PMs with the PEOs for current programs to plan transition to a software factory and continuous iterative development.
DSB18	Rec 4c: Defense prime contractors should incorporate continuous iterative development into a long-term sustainment plan
DSB18	Establish a common list of source selection criteria for evaluating software factories for use throughout the Department.
FCW'18	Contractors would allow government to develop past performance reports with less documentation and less contractor opportunity to appeal their ratings
USDS	Agile software development is the preferred methodology for software development contracts that contribute to the creation and maintenance of digital services, whether they are websites, mobile applications, or other digital channels
USDS	Although Part 39 does not directly speak to Agile software development practices, it endorses modular contracting principles where information technology systems are acquired in successive, interoperable increments to reduce overall risk and support rapid delivery of incremental new functionality
USDS	With Agile software development, requirements and priorities are captured in a high level Prod-

	uct Vision, which establishes a high level definition of the scope of the project, specifies expected outcomes, and produces high level budgetary estimates.
USDS	Under Agile software development, the Government retains the responsibility for making decisions and managing the process; it plays a critical role in the IPT as the Product Owner by approving the specific plans for each iteration, establishing the priorities, approving the overall plan revisions reflecting the experience from completed iterations, and approving deliverables.
USDS	OMB's 2012 Contracting Guidance to Support Modular Development states that IDIQ contracts may be especially suitable for Agile software development because they provide a high level of acquisition responsiveness, provide flexibility, and accommodate the full spectrum of the system lifecycle that provide both development and operational products and services. BPAs may work with Agile software development using modular contracting methods. Additionally, standalone contracts or single award contracts may be used.
USDS	The Agile process works only if there are appropriate dedicated resources, as the process can be labor intensive. Agencies need to ensure adequate resources are applied to manage their contracts irrespective of the strategy used. Strong contract management ensures projects stay on course and helps prevent the agency from becoming overly reliant on contractors.

Additional Recommendation B5 - Cloud Computing

Line of Effort	Create and maintain cross-program/cross-service digital infrastructure
Recommendation	Remove obstacles to DoD usage of cloud computing on commercial platforms, including DISA CAP limits, lack of ATO reciprocity, and access to modern software development tools.
Stakeholders	DoD CIO, Service CIOs, USD(A&S)
Background	Lack of ATO reciprocity and current DoD procedures for cloud are obstacles to leveraging modern infrastructure and tools.
Desired State	DoD developers and contractors are able to use modern cloud computing environments and commercial development tools quickly, with a single certification that is transferable to other groups using the same environment, tools
Congressional Role	N/A

Draft Implementation Plan		Lead Stakeholders	Target Date
B5.1	Rescind Cloud Access Point (CAP) policy and replace with policy that ensures security at scale (including end-to-end encryption)	DoD CIO	Q3 FY19
B5.2	In conjunction with Rec B4, allow transfer of ATOs for commercial platforms between programs and services	DoD CIO	Q3 FY19
B5.3	Create specifications and certification process for approval of standard development tools (w/ ATO reciprocity)	DoD CIO	Q4 FY19
B5.4	In conjunction with Rec B1, establish a common, enterprise ability to develop software solutions in the “easy-to-acquire-and-provision” cloud that is fully accredited by design of the process, tools, and pipeline	USD(A&S)	Q1 FY20

SWAP working group inputs related to this recommendation

Acq	Include an approach for enterprise-level DevSecOps and other centralized infrastructure development and management, approach for shared services, and applications management.
Inf	Establish a DoD enterprise ability to procure, provision, pay for, and use cloud that is no different from the commercial entry points for cloud computing.
Inf	DoD should establish a common, enterprise ability to develop software solutions in the “easy-to-acquire-and-provision” cloud that is fully accredited by design of the process, tools, and pipeline.

Related recommendations from previous studies

Sec 809	Rec. 43: Revise acquisition regulations to enable more flexible and effective procurement of consumption-based solutions.

Additional Rec B6 - Certify Code/Toolchain

Line of Effort	Create and maintain cross-program/cross-service digital infrastructure
Recommendation	Shift from certification of executables for low and medium risk deployments to certification of code/architectures and certification of the development, integration, and deployment toolchain.
Stakeholders	USD(A&S), SAE, DoD CIO, Service CIO
Background	Today, the typical focus of security accreditation on programs is to certify each version of the code that is intended for release. This works against the goal of frequent updates since the more versions of software that are created, the more often the time and expense of the certification have to be borne by the program.
Desired State	The Department will accredit software infrastructures that are capable of producing quality code when used appropriately, enabling each version of the code produced on that infrastructure to be treated as certifiably secure (within appropriate limits, e.g. for versions that do not entail major architectural changes). This this change in certification, DoD will enable rapid fielding of mission-critical code at high levels of information assurance.
Congressional Role	N/A

Draft Implementation Plan		Lead Stakeholders	Target Date
B6.1	Identify and use commercial certification procedures for security assessments and deployment mechanisms that can be used for DoD software programs	CIO	Q4 FY19
B6.2	Identify 3 lead programs for initial implementation of certification procedures	A&S, SAE	Q1 FY20
B6.3	Expand certification procedures to 10 additional sites, spanning all Services and multiple OSD offices; update procedures with each new certification to streamline process.	A&S, SAE with CIO	Q3 FY20
B6.4	Update DoDI 8501.01, Risk Management Framework for DoD Information Technology, to reflect revised certification procedures.	CIO with SAE, A&S	Q4 FY20

SWAP working group inputs related to this recommendation

Acq	Exempt the DoD from the Clinger Cohen Act, 40 U.S.C. 1401(3)
Inf	DoD should establish a common, enterprise ability to develop software solutions in the “easy-to-acquire-and-provision” cloud that is fully accredited by design of the process, tools, and pipeline.

Related recommendations from previous studies

SEI'12	Use common operating environment (COE), software development toolkits (SDKs) and enterprise services to speed up accreditation time.
SEI'12	Apply a risk-based, incremental approach to security architecture.
SEI'12	Leverage design tactics such as layering and encapsulation to limit impact of change.

SEI'13	For an SoS or for the more likely case of a system or component that participates in an existing SoS, an effective risk management approach should • scale to size and complexity of systems of systems • incorporate dynamics • integrate across full life cycle: requirements to sustainment • focus on success as well as failure

Additional Recommendation B7 – Hardware as a Consumable

Line of Effort	Digital
Recommendation	Plan and fund computing hardware (of all types) as consumable resources, with continuous refresh and upgrades to the most recent, most secure OS and platform components.
Stakeholders	USD(A&S), SAE, DoD CIO, Service CIO
Background	Current information technology (IT) refreshes take 8-10 years from planning to implementation, which means that most of the time our systems are running on obsolete hardware that limits our ability to implement the algorithms required to provide the level of performance required to stay ahead of our adversaries. Maintaining legacy code for different variants that have hardware capabilities ranging from 2 to 12 years old is an almost impossibly large spread of capability in computing, storage, and communications. From a contracting perspective, this change would require DoD to provide a stable annual budget that paid for new hardware and software capability (see Commandment #3), but this would very likely save money over the longer term.
Desired State	Whenever possible, applications are run in the cloud, so that algorithms can be run on the latest hardware and operating systems. For weapons systems, a continuous hardware refresh mentality is in place that enables software upgrades, crypto updates, and connectivity upgrades to be rapidly deployed across a fleet on an ongoing basis. The adoption rate of the latest hardware and operating system versions is tracked and targets are set for maintaining hardware and operating system "readiness". The paradigm for computing hardware from current Property, Plant, and Equipment categorization (as investments with depreciation schedules) is modified to treat hardware as an expense.
Congressional Role	Provide funding for ongoing replacement of computing hardware as a consumable with a 2-4 year lifetime. Track "readiness" of currently deployed software capability in part by measuring age of the hardware and operating systems on which software is being run.

Draft Implementation Plan		Lead Stakeholders	Target Date
B7.1	Establish funds for initial existing weapons platforms involving computing hardware to replace hardware every 2-4 years (like oil)	CIO with USD(C), SAE	Q1 FY20
B7.2	Establish draft guidance for determining when to update hardware and operating systems to balance cost with risk/capability.	CIO	Q2 FY20
B7.3	Work with FASAB to change audit treatment of software with these goals: (1) Separate category for software instead of being characterized as Property, Plant, and Equipment; (2) Default setting that software is an expense, not an investment; and (3) there is no "sustainment" phase for software.	USD(A&S), in coordination with USD(C)	Q4 FY20
B7.4	Modify DoD Financial Management Regulation (FMR) to capture changes in how hardware is purchased and retired from service.	USD(C)	Q1 FY21

SWAP concept paper recommendations related to this recommendation

10C	Move to a model of continuous hardware refresh in which computers are treated as a consumable

	with a 2-3 year lifetime
Visits	Make use of platforms (hardware and software) that continuously evolve at the timescales of the commercial sector (3-5 years between HW/OS updates)

Related recommendations from previous studies

Sec 809	Rec. 44: Exempt DoD from Clinger–Cohen Act Provisions in Title 40:
Sec 809	Rec. 56: Use authority in Section 1077 of the FY 2018 NDAA to establish a revolving fund for information technology modernization projects and explore the feasibility of using revolving funds for other money-saving investments.

Primary Recommendation C1 – Organic Development Groups

Line of Effort	Create new paths for digital talent (especially internal talent)
Recommendation	**Create software development units in each Service consisting of military and civilian personnel who develop and deploy software to the field using DevSecOps practices.**
Stakeholders	USD(A&S), USD(P&R), SAE, Service HR
Background	The DoD's capacity to apply modern technology and software practices to meet its mission is required in order to remain relevant in increasingly technical fighting domains, especially against peer adversaries. While DoD has both military and civilian software engineers (often associated with maintenance activities), the IT career field suffers from a lack of visibility and support. The Department has not prioritized a viable recruiting strategy for technical positions, and there is no comprehensive training or development program that prepares the technical and acquisition workforce to adequately deploy modern software development tools and methodologies.
Desired State	DoD recruits, trains, and retains internal capability for software development, including by service members, and maintains this as a separate career track (like DoD doctors, lawyers, and musicians). Each Service has organic development units that are able to create software for specific needs and that serve as an entry point for software development capability in military and civilian roles (complementing work done by contractors). The Department's workforce embraces commercial best practices for the rapid recruitment of talented professionals, including the ability to onboard quickly and provide modern tools and training in state-of-the-art training environments. Individuals in software development career paths are able to maintain their technical skills and take on DoD leadership roles.
Congressional Role	Congress should receive regular "readiness" reports that include organic software development capability and provide budget required to maintain desired capability level and resources for modern software development.

	Draft Implementation Plan	Lead Stakeholders	Target Date
C1.1	Exercise existing acquisition and cybersecurity hiring authorities to increase the number of software developers in DoD programs with vacant positions.	SAE, PEO, with CIO (cyber excepted service ability)	Immediately
C1.2	Create new military occupational specialty (MOS) and core occupational series plus corresponding career tracks for each Service; use to grow digital talent for DevSecOps	J1 and comparable X1 for each Service with USD(P&R)	Q1 FY20
C1.3	Create regulations to allow standard identification, recruitment, and onboarding of experienced civilian software talent, especially on rotation from private sector roles	USD(P&R)	Q1 FY20
C1.4	Create mechanism for tracking software development expertise and use as preferred experience for promotion into software engineer and acquisition roles	A&S, CIO	Q2 FY20
C1.5	Obtain additional manpower authorizations for military and civilian SW developers.	USD(A&S), with USD(P&R), SAE	FY20, FY21

C1.6	Stand up one or more software factories within each Service, tied to field needs that can be satisfied through organic software development groups.	SAEs, with PEOs Digital	FY20 (pilot), FY21 (scale)

SWAP concept paper recommendations related to this recommendation

10C	Establish Computer Science as a DoD core competency
D&D	Hire competent people with appropriate expertise in software to implement the desired state and give them the freedom to do so (“competence trumps process”)

SWAP working group inputs related to this recommendation

M&S	The definition of “core capabilities” in 10 USC 2464 should be revisited in light of warfighter dependence on software intensive systems to determine the scope of DoD’s core organic software engineering capability, and we should engage with Congress on the proposed revision to clarify the intent and extent of key terminology used in the current statute.
M&S	Revise industrial base policy to include software and DoD’s organic software engineering capabilities and infrastructure. Start enterprise planning and investment to establish and modernize organic System Integration Labs (SILs), software engineering environments, and technical infrastructure; invest in R&D to advance organic software engineering infrastructure capabilities.
Wkf	Develop a core occupational series based on current core competencies and skills for software acquisition and engineering.
Wkf	Overhaul the recruiting and hiring process to use simple position descriptions, fully leverage hiring authorities, engage subject matter experts as reviewers, and streamline the onboarding process to take weeks instead of months
Wkf	Embrace private-sector hiring methods to attract and onboard top talent from non-traditional backgrounds that may require special authorities to join the Department
Wkf	Develop a strategic recruitment program that targets civilians, similar to the recruitment strategy for military members, [including] prioritizing experience and skills over cookie-cutter commercial certifications or educational attainment
Wkf	Establish an alliance across the services that incentivizes and provides software practitioners a modern engagement platform (e.g. a chatOps platform) to connect across services, share their skills, communicate through knowledge channels, gather pain points, and develop solutions leveraging the full enterprise.
Wkf	Allow for greater private-public sector fluidity across the workforce while empowering the existing workforce to create a place where they want to work
Wkf	Modify Title 10, §1596a to create a new Computer-language proficiency pay statute.
Wkf	Pilot a cyber hiring team with the necessary authorities to execute report recommendations and that can serve as a Department-wide alternative to organization’s traditional HR offices and will provide expedited hiring and a better candidate experience for top tier cyber positions.

Related recommendations from previous studies

DSB87	Rec 26: Each Service should provide its software Product Development Division with the ability to do rapid prototyping in conjunction with users.
DSB87	Rec 36: Establish mechanisms for tracking personnel skills and projecting personnel needs.
DSB87	Rec 37: Structure some office careers to build a cadre of technical managers with deep technical mastery and broad operational overview.
SEI'10	Improve compensation and advancement opportunities to increase tenure.

Primary Recommendation C2 – Acquisition Workforce Training

Line of Effort	Create new paths for digital talent (especially internal talent)
Recommendation	**Expand the use of (specialized) training programs for CIOs, SAEs, PEOs, and PMs that provide (hands-on) insight into modern software development (e.g., agile, DevOps, DevSecOps) and the authorities available to enable rapid acquisition of software.**
Stakeholders	USD(A&S), DoD CIO, SAE, Service CIO
Background	Acquisition professional have been trained and had success in the current model, which has produced the world's best military but this model is not serving well for software. New methodologies and approaches introduce unknown risks, and acquisition professionals are often not incentivized to make use of the authorities available to implement modern software methods. At the same time, senior leaders in DoD need to be more knowledgeable about modern software development practices so they can recognize, encourage, and champion efforts to implement modern approaches to software program management.
Desired State	Senior leaders, middle management, and organic and contractor-based software developers are aligned in their view of how modern software is procured and developed. Acquisition professionals are aware of all of the authorities available for software programs and use them to provide flexibility and rapid delivery of capability to the field. Program leaders are able to assess the status of software (and software-intensive) programs and spot problems early in the development process, as well as provide continuous insight to senior leadership and Congress. Highly specialized requirements are scrutinized to avoid developing custom software when commercial offerings are available that are less expensive and more capable.
Congressional Role	Prioritize experience with modern software development environments in approval of senior acquisition leaders.

Draft Implementation Plan		Lead Stakeholders	Target Date
C2.1	Leverage existing training venues to add content about modern software development practices	USD(A&S), SAEs with DAU	Q4 FY19
C2.2	Create and provide training opportunities via boot camps and rotations for acquisition professionals to obtain hands-on experience in DevSecOps programs	A&S with SAEs, USD(P&R)	FY20 (MVP) FY21 (scale)
C2.3	Develop additional training opportunities for key leaders about modern software development practices	USD(A&S), SAE	Q2 FY20
C2.4	Create software continuing education programs and requirements for CIOs, SAEs, PEOs and PMs modeled after MCLE (Minimum Continuing Legal Education) for lawyers	A&S, DAU	Q3 FY20

SWAP working group inputs related to this recommendation

Con	Provide training to KOs, PMs, and leadership to understand the value and methods associated with agile and modular implementation
Wkf	Create a software acquisition workforce fund (similar to the existing Defense Acquisition Workforce Development Fund (DAWDF)) ... to hire and train a cadre of modern software acquisition experts.
Wkf	Pilot development programs that provide comprehensive training for all software acquisition professionals, developers, and associated functions.
Con	Provide training to KOs, PMs, and leadership to understand the value and methods associated with agile and modular implementation
Con	Educate PMs and KOs on Open Source, proprietary, and Government funded code

Related recommendations from previous studies

DSB'09	All CIOs should approve IT acquisition program manager training and certification and advise the personnel selection process.
DSB'09	The USD (AT&L) shall direct the Defense Acquisition University, in coordination with the Information Resources Management College, to integrate the new acquisition model into their curriculum.
DSB18	USD(A&S) should task the PMs of programs that have transitioned successfully to modern software development practices to brief best practices and lessons learned across the Services.
DSB18	Rec 5d: The USD(A&S) and the USD(R&E) should direct the Defense Acquisition University (DAU) to establish curricula addressing modern software practices leveraging expertise from the DDS, the FFRDCs, and the University Affiliated Research Centers (UARCs).
DSB18	Rec 5g: DoD career functional Integrated Product Team (IPT) leads should immediately establish a special software acquisition workforce fund modeled after the Defense Acquisition Workforce Development Fund (DAWDF), the purpose of which is to hire and train a cadre of modern software acquisition experts across the Services.
DSB18	Rec 5h: PMs should create an iterative development IPT with associated training. The Service Chiefs should delegate the role of Product Manager to these IPTs.
DSB18	Rec 5b: The Service Acquisition Career Managers should develop a training curriculum to create and train [a] cadre [of] software-informed PMs, sustainers and software acquisition specialists.
809	Rec 27: Improve resourcing, allocation, and management of the Defense Acquisition Workforce Development Fund (DAWDF)
809	Rec. 59: Revise the Defense Acquisition Workforce Improvement Act to focus more on building professional qualifications.

Additional Recommendation C3 – Increase PMO Experience

Line of Effort	Create new paths for digital talent (especially internal talent)
Recommendation	Increase the knowledge, expertise, and flexibility in program offices related to modern software development practices to improve the ability of program offices to take advantage of software-centric approaches to acquisition.
Stakeholders	USD(A&S), SAE, USD(P&R)
Background	Acquisition professionals do not always have experience and insights into modern software development environments, especially in the opportunities (and limitations) for continuous integration/continuous deployment (CI/CD), automated testing (including security testing), and modern cloud-computing architectures. New methodologies and approaches introduce unknown risks, while the old acquisition and development approaches built the world's best military. Program offices not incentivized to adopt new approaches to acquisition and implementation of software, and inertia represents a barrier to change.
Desired State	Program management offices have staff available with experience in modern software development environments and who are able to make creative (but legal) use of available authorities for acquisition of software to fit the needs of modern software development solutions. Management of most types of software relies on (continuous) measurement of capability delivered to the field rather than being tied to satisfaction of objectives. Time and cost are used as constraints with schedule of delivery of features replanned at each iteration cycle based on warfighter/user feedback.
Congressional Role	N/A

Draft Implementation Plan		Lead Stakeholders	Target Date
C3.1	Establish list of skills and experience needed by program office staff to be considered “fully staffed” for a software program.	A&S with SAEs, USD(P&R)	Q4 FY19
C3.2	Modify Position Descriptions for those in leadership positions in software acquisition programs to prioritize and reward prior experience in software development.	USD(A&S), SAE, Service HR	Q1 FY20
C3.3	Create and provide training opportunities via boot camps and rotations for acquisition professionals to obtain hands-on experience in DevSecOps programs [duplicate of C2.2]	A&S with SAEs, USD(P&R)	Q2 FY20 (MVP) FY21 (scale)
C3.4	Modify PM training requirements to obtain DAU Level IIII certification to include hands-on experience with modern software development.	USD(A&S)	Q3 FY20
C3.5	Evaluate readiness level of software (and software-intensive) program offices by comparing experience/skill sets available with the list of needed skills from C3.1 (hint: consider tracking those skills sets; see Action C1.2)	A&S with SAEs, USD(P&R)	Q4 FY20 (MVP) FY21 (scale)

SWAP concept paper recommendations related to this recommendation

D&D	Hire competent people with appropriate expertise in software to implement the desired state and give them the freedom to do so (“competence trumps process”) [dup]

SWAP working group inputs related to this recommendation

Acq	Lead tester from either DOT&E or JITC (preferably both, if JITC is being used as test org) must be a subject matter expert in the subject being tested, similar to how qualified test pilots run test flights (health records, financial systems, etc.)
Wkf	Empower a small cadre of Highly Qualified Experts and innovative Department employees to execute changes
Wkf	Create a software acquisition workforce fund (similar to the existing Defense Acquisition Workforce Development Fund (DAWDF)) ... to hire and train a cadre of modern software acquisition experts. [dup]
Wkf	Provide Agile, Tech and DevSecOps coaches in Program Offices to support transformations, adoption of modern software practice and sharing lessons across the enterprise
Wkf	Develop a core occupational series based on current core competencies and skills for software acquisition and engineering. [dup]
Wkf	Modernize Position Description and Hiring Practices. Modifying Existing Language - Title 5, Part III, Subpart D, Chapter 53, the addition of this pilot program needs to be added.
Wkf	Develop a Modern Academy. Modification Language - Title 10 §1746: This section should be added under the Defense Acquisition University, however, the HQE Cadre from Proposal #1 will lead the development of this pilot training program. Note: Tied with FY18 NDAA §891
Wkf	Private-Public Sector Fluidity. Modification Language - Title 5, §§3371-3375: Expand the Inter-Government Personnel Act and allow more civil service employees to work with non-Federal Agencies and Educational Institutions. Modification Language - Title 10, §1599g: Expand the Public-Private Talent Exchange Program and modify the language to reduce the “repayment” period from 1:2 to 1:1 ratio.
Wkf	Establish Workforce Fund. New Legislation - Similar to DAWDF, but the primary use will be for hiring and training a cadre of modern software acquisition experts.

Related recommendations from previous studies

DSB18	Rec 5a: The service acquisition commands (e.g., the LCMC, the NAVAIR, the U.S. Naval Sea Systems Command (NAVSEA), and the AMC) need to develop workforce competency and a deep familiarity of current software development techniques.
DSB18	Rec 5a.2: Services acquisition commands should use this cadre early in the acquisition process to formulate acquisition strategy, develop source selection criteria, and evaluate progress.
DSB18	Over the next two years, the service acquisition commands need to develop workforce competency and a deep familiarity of current software development techniques.
Sec 809	Rec. 40: Professionalize the requirements management workforce.
Sec 809	Rec. 46: Empower the acquisition community by delegating below threshold reprogramming decision authority to portfolio acquisition executives.
NPS'16a	The growth of rapid acquisition organizations gives acquisition executives new avenues to meet their top priority and rapid capability demands. However, these organizations may also have negative effects on traditional acquisition organizations. The DoD’s top talent will flock to the rapid acquisition organizations so that they can work on high-priority programs with minimal restrictions and likely achieve greater success.

NPS'16a	Contracting Officers (COs) must function as strategic partners tightly integrated into the program office, rather than operate as a separate organization that simply processes the contract paperwork
NPS'16b	Culturally, the acquisition community needs to embrace the available tools as opportunities, while being selective with procurement methods and adaptive to the market environment
CSIS'15	Rapid acquisition succeeds when senior leaders are involved in ensuring that programs are able to overcome the inevitable hurdles that arise during acquisition, and empower those responsible with achieving the right outcome with the authority to get the job done while minimizing the layers in between
CSIS'15	Rapid acquisition is fundamentally an ongoing dialogue between the acquisition and operational communities about what the real needs of the warfighter are and what the art of the possible is in addressing them.
GAO'17	Empower program managers to make decisions on the direction of the program and to resolve problems and implement solutions.
GAO'17	Hold program managers accountable for their choices.
GAO'17	Require program managers to stay with a project to its end.
GAO'17	Encourage program managers to share bad news, and encourage collaboration and communication.
SEI'15	5. Government Personnel Experience. Government personnel with extensive experience in developing and managing acquisition strategy and technical architecture should be dedicated and available to a program throughout its duration.
SEI'10	Improve qualifications of acquisition staff emphasizing software expertise.
SEI'10	Assign PMs, DPMs, and other key positions for the program’s duration and into deployment. Use civilians if military rotations are not amenable.
OSD'06	Fully implement the intent of the Packard Commission. Create a streamlined acquisition organization with accountability assigned and enforced at each level.
OSD'06	Realign responsibility, authority and accountability at the lowest practical level of authority by reintegrating the Services into the acquisition management structure.
OSD'06	Seek legislation to retain high-performance military personnel in the acquisition workforce to include allowing military personnel to remain in uniform past the limitations imposed by the Defense Officer Personnel Management Act and augment their pay to offset the “declining marginal return” associated with retired pay entitlement.
OSD'06	Request that the White House Liaison Office create a pool of acquisition-qualified, White House pre-cleared, non-career senior executives and political appointees to fill executive positions, to provide leadership stability in the Acquisition System.
OSD'06	Immediately increase the number of federal employees focused on critical skill areas, such as program management, system engineering and contracting. The cost of this increase should be offset by reductions in funding for contractor support.
OSD'06	Establish a consistent definition of the acquisition workforce with the Under Secretary of Defense for Acquisition Technology and Logistics, working with the Service Secretaries to include in that definition all acquisition-related budget and requirements personnel.

Additional Recommendation C4 – Recruiting (Transient) Digital Talent

Line of Effort	People
Recommendation	Restructure the approach to recruiting digital talent to assume that the average tenure of a talented engineer will be 2-4 years, and make better use of HQEs, IPAs, special hiring authorities, reservists, and enlisted personnel to provide organic software development capability, while at the same time incentivizing and rewarding internal talent.
Stakeholders	USD(A&S), SAE
Background	Current DoD personnel systems assuming that military and government employees will "grow through the ranks" and that individuals will stay in government service for long periods of time. The attractions of the private sector creates challenges in retaining personnel that are not likely to be overcome, so a different approach is needed.
Desired State	DoD takes advantage of all individuals who are willing to serve, whether for a long period or a short period and amplifies the ability of individuals to make a contribution during their time in government. Internal talent is recognized and retained through merit-based systems of promotion and job assignment.
Congressional Role	Support and encourage the use of existing authorities to hire digital talent in creative ways that match the intent of Congress and solve the need for more flexible arrangements in which talented individual move in and out of government service (without creating unnecessary barriers).

Draft Implementation Plan		Lead Stakeholders	Target Date
C4.1	Exercise existing hiring authorities to increase the number of highly skilled SW people in DoD program, such as the Cyber Excepted Workforce.	SAE, PEO	Starting now
C4.2	In conjunction with Recs C1 and D3, create a database of individuals in enlisted, officer, reserve, and civilian positions with software development skills and experience for internal recruiting use to SW squadrons & PAOs	CMO?, Service HR groups?	Q3 FY19
C4.3	Within organic software programs, create processes for maintaining release cadence under the assumption of up to 25% turnover per year	PMOs	Q4 FY19
C4.4	Require software-intensive project proposals to include a plan for maintaining cadence-related metrics in face of up to 25% turnover of staff	SAEs	Q4 FY19
C4.5	Identify bottlenecks in providing security clearances for software developers are target granting of interim clearances within 1 month of start date.	???	Q1 FY20
C4.6	Revise GS and military promotion guidelines for software developers to allow rapid promotion of highly qualified individuals with appropriate skills, independent of "time in grade"	USD(P&R)	FY20 for FY21 NDAA
C4.7	Obtain additional funding for military, civilian SW developers, including existing personnel, HQEs, IPAs, reservists, and direct commissioning	USD(A&S), USD(P&R), SAE	FY21

SWAP concept paper recommendations related to this recommendation

10C	Establish Computer Science as a DoD core competency

SWAP working group inputs related to this recommendation

Wkf	Develop a core occupational series based on current core competencies and skills for software acquisition and engineering.
Wkf	Overhaul the recruiting and hiring process to use simple position descriptions, fully leverage hiring authorities, engage subject matter experts as reviewers, and streamline the onboarding process to take weeks instead of months
Wkf	Embrace private-sector hiring methods to attract and onboard top talent from non-traditional backgrounds that may require special authorities to join the Department
Wkf	Develop a strategic recruitment program that targets civilians, similar to the recruitment strategy for military members, [including] prioritizing experience and skills over cookie-cutter commercial certifications or educational attainment

Related recommendations from previous studies

DSB87	Rec 34: Do not believe that DoD can solve its skilled personnel shortage; plan how best to live with it, and how to ameliorate it.
Sec 809	Rec. 45: Create a pilot program for contracting directly with information technology consultants through an online talent marketplace.
SEI'10	Divide large acquisition development efforts into multiple smaller, shorter duration programs.

Primary Recommendation D1 – Source Code Access

Line of Effort	Adopt DevSecOps practices and approaches
Recommendation	**Require access to source code, software frameworks, and development toolchains – with appropriate IP rights – for DoD-specific code, enabling full security testing and rebuilding of binaries from source.**
Stakeholders	USD(A&S), CIO, SAE
Background	For many DoD systems, source code is not available to DoD for inspection and testing, and DoD relies on suppliers to write code for new compute environments. As code ages, suppliers are not required to maintain codebases without an active development contract and "legacy" code is not continuously migrated to the latest hardware and operating systems.
Desired State	DoD has access to source code for DoD-specific software systems that it operates and uses to perform detailed (and automated) evaluation of software correctness, security, and performance, enabling more rapid deployment of both initial software releases and (most importantly) upgrades (patches and enhancements). DoD is able to rebuild executables from scratch for all of its systems, and has the rights and ability to modify (DoD-specific) code when new conditions and features arise. Code is routinely migrated to the latest computing hardware and operating systems, and routinely scanned against currently-known vulnerabilities. Modern IP language is used to ensure that the government can use, scan, rebuild, and extend purpose-built code, but contractors are able to use licensing agreements that protect any IP that they have developed with their own resources. Industry trusts DoD with its code and has appropriate IP rights for internally developed code.
Congressional Role	N/A

Draft Implementation Plan		Lead Stakeholders	Target Date
D1.1	Work with industry to modernize policies for software code ownership, licensing, and purchase. See 2018 Army IP directive as an example.	USD(A&S)	Q3 FY19
D1.2	Modify FAR/DFARS guidance to require software source code deliverables for GOTS and for government-funded software development. Obtain rights for access to source code for COTS wherever possible (and useful).	USD(A&S)	Q3 FY20
D1.3	Modify DoDI 5000.02 and DoDI 5000.75 to make access to code and development environments the default.	USD(A&S)	Q3 FY20
D1.4	Develop a comprehensive source code management plan for DoD including the safe and secure storage, access control, testing and field of use rights.	USD(A&S), with CIO	Q4 FY20

SWAP concept paper recommendations related to this recommendation

10C	Every purpose-built DoD software system should include source code as a deliverable.
D&D	Require source code as a deliverable on all purpose-built DoD software contracts. Continuous development and integration, rather than sustainment, should be a part of all contracts. DoD personnel should be trained to extend the software through source code or API access

Related recommendations from previous studies

DSB87	Rec 22: DoD should follow the concepts of the proposed FAR 27.4 for data rights for military software, rather than those of the proposed DoD 27.4, or it should adopt a new “Rights in Software” Clause as Recommended by Samuelson, Deasy, and Martin in Appendix A6.
DSB18	Rec 6b: Availability, cost, compatibility, and licensing restrictions of [the proposed software factory] framework elements to the U.S. Government and its contractors should be part of the selection criteria for contract award.
DSB18	Rec 6c: all documentation, test files, coding, application programming interfaces (APIs), design documents, results of fault, performance tests conducted using the framework, and tools developed during the development, as well as the software factory framework, should be delivered to the U.S. Government at each production milestone; OR escrowed and delivered at such times specified by the U.S. Government (i.e., end of production, contract reward).
DSB18	Rec 6d: Selection preference should be granted based on the ability of the United States to reconstitute the software framework and rebuild binaries, re-run tests, procedures, and tools against delivered software, and documentation.

Primary Recommendation D2 – Security Considerations

Line of Effort	Adopt DevSecOps practices and approaches
Recommendation	**Make security a first-order consideration for all software-intensive systems, under the assumption that security-at-the-border will not be enough.**
Stakeholders	USD(A&S), CIO, DDS, SAE, DDR&E(DT), DOT&E
Background	Multiple GAO, DoDIG, and other reports have identified cybersecurity as a major issue in acquisition programs.
Desired State	TBD
Congressional Role	TBD

Draft Implementation Plan		Lead Stakeholders	Target Date
D2.1	Adopt standards for secure software development and testing that use a zero-trust security model	CIO, with DDS	Q3 FY19
D2.2	Develop, deploy, and require the use of IA-accredited (commercial) development tools for DoD software development	CIO, PEO Digital	Q4 FY19
D2.3	Establish automated penetration testing as part of OT&E evaluation (integrated with program development)	DOT&E	Q1 FY20
D2.4	Establish red team responsible for ongoing vulnerability testing against any defense software system	CIO with DDS	Q2 FY20
D2.5	Establish security as part of the selection criteria for software programs	A&S with CIO, SAEs	Q3 FY20

SWAP concept paper recommendations related to this recommendation

10C	Only run operating systems that are receiving (and utilizing) regular security updates for newly discovered security vulnerabilities.
10C	Data should always be encrypted unless it is part of an active computation.
D&D	Create automated test environments to enable continuous (and secure) integration and deployment to shift testing and security left

SWAP working group inputs related to this recommendation

Sec	People must learn to appreciate that speed helps increase security. Security is improved when changes and updates can be made quickly to an application. Using automation, software can be reviewed quickly.
Sec	The AO must also be able to review documentation and make a risk decision quickly and make that decision on the process and not the product.
T&E	Establish a statutory "Live Fire" requirement on software-intensive systems as there is on "Covered Systems" for protecting our warfighters from kinetic threats. “Shoot at it” before design is complete and certainly before it is put into the operational environment.
T&E	Establish a federation of state-of-the-art cyber testing capabilities from non-profit institutions to support trusted, survivable and resilient defense systems and ensure the security of software and hardware developed, acquired, maintained, and used by the DoD.
T&E	Establish cyber security as the “4th leg” in measurement of Acquisition system/program performance: Cost, Schedule, Performance, Cyber Security.
T&E	Develop mechanisms to enforce existing software and cyber security policies (from cradle-to-grave) that are not (now) being adequately enforced.
T&E	Ensure each DoD Component is responsible for representing its own forces and capabilities in a digital modeling environment (e.g., M&S, digital twin, etc.), making them available to all other DoD users, subject to a pre-defined architecture and supporting standards. DIA will represent threat forces and capabilities in a digital form consistent with this architecture/standards. Programs are required to use DIA-supplied threat models, unless sufficient justification is provided to use other.

Related recommendations from previous studies

DSB’09	In the Services and agencies, the CIOs should also have strong authorities and responsibilities for system certification, compliance, applications development, and innovation.
DSB’09	The DOD CIO, supported by CIOs in the Services and agencies, should be responsible for certifying that systems and capabilities added to the enterprise do not introduce avoidable vulnerabilities that can be exploited by adversaries.
Sec 809	Rec. 77: Require role-based planning to prevent unnecessary application of security clearance and investigation requirements to contracts.

Primary Recommendation D3 – Software Features

Line of Effort	Adopt DevSecOps practices and approaches
Recommendation	**Shift from the use of rigid lists of requirements for software programs to a list of desired features and required interfaces/characteristics, to avoid requirements creep, overly ambitious requirements, and program delays**
Stakeholders	USD(A&S), Joint Staff, SAEs
Background	Current DoD requirements processes significantly impede its ability to implement modern SW development practices by spending years establishing program requirements and insisting on satisfaction of requirements before a project is considered "done". This impedes rapid implementation of features that are of the most use to the user.
Desired state	Rather than a list of requirements for every feature, programs should establish a minimum set of requirements required for initial operation, security, and interoperability, and place all other desired features on a list that will be implemented in priority order, with the ability for DoD to redefine priorities on a regular basis.
Congressional Role	Modify relevant statutes to allow the use of evolving features over rigid requirements and develop alternative methods for obtaining information on program status (See Rec A2, Action A2.4)

Draft Implementation Plan		Lead Stakeholders	Target Date
D3.1	Modify requirements guidance by memo to shift from a list of requirements for software to a list of desired features and required interfaces/characteristics.	USD(A&S)	Q4 FY19
D3.2	Update CJCSI 3170.01H (JCIDS requirements process) to reflect contents of guidance memos	Joint Staff	Q1 FY20
D3.3	Modify DoDI 5000.02 and DoDI 5000.75 (or integrate into new DoDI 5000.SW)	USD(A&S)	Q2 FY20
D3.4	Define and use new budget exhibits for software programs using evolving lists of features in place of requirements (see also Rec A2).	USD(A&S), with USD(C), CAPE, HAC-D, SAC-D	Q3 FY20

SWAP concept paper recommendations related to this recommendation

10C	Adopt a DevOps culture for software systems.
10C	All software procurement programs should start small, be iterative, and build on success – or be terminated quickly.
D&D	Accept 70% solutions in a short time (months) and add functionality in rapid iterations (weeks)

Related recommendations from previous studies

SEI'01	Ensure that all critical functional and interoperability requirements are well specified in the contract (statement of work, Statement of Objectives).
SEI'01	Handle requirements that have architectural consequences as systems engineering issues—up front.
SEI'12	Ensure requirements prioritization of backlog considers business value and risk.
GAO'17	Match requirements to resources—that is time, money, technology, and people—before undertaking new development efforts.

Additional Recommendation D4 – Continuous Metrics

Line of Effort	Change the practice of how software is procured and developed
Recommendation	Create and use automatically generated, continuously available metrics that emphasize speed, cycle time, security, user value and code quality to assess, manage, and terminate software programs (and software components of hardware programs).
Stakeholders	USD(A&S), CAPE, SAE, Service Cost Orgs
Background	Current program reporting requirements are largely manual, time consuming, and provide limited insight into the SW health of a program. New metrics are required that match the DevSecOps approach of continuous capability delivery and maintenance and provide continuous insight into program progress.
Desired State	Program oversight will re-focus on the value provided by the software as it is deployed to the warfighter/user, and will rely more heavily on metrics that can be collect in an automated fashion from instrumentation on the DevSecOps pipeline and other parts of the infrastructure. Specific metrics will depend on the type of software rather than a one-size-fits-all approach.
Congressional Role	N/A (but see Rec A3)

Draft Implementation Plan		Lead Stakeholder	Target Date
D4.1	Modify acquisition policy guidance to specify use of automatically generated, continuously available metrics that emphasize speed, cycle time, security, and code.	USD(A&S)	Q3 FY19
D4.2	Modify cost estimation policy guidance to specify use of automatically generated, continuously available metrics that emphasize speed, cycle time, security, and code.	CAPE	Q3 FY19
D4.3	Develop specific measure of software quality, value and velocity and the tools to implement the automatic generation and reporting	DDS, with CAPE, CIO, USD(C)	Q4 FY19
D4.4	Modify DoDI 5000.02, DoDI 5000.75, and DoDI 5105.84 to reflect use of updated methods, remove earned value management (EVM) for software programs	A&S	Q1 FY20

SWAP working group inputs related to this recommendation

Acq	Revise DFARS Subpart 234.201, DoDI 5000.02 Table 8, and OMB Circular A-11 to remove EVM requirement
Con	Allow for documentation and reporting substitutions to improve agility (agile reporting vs EVM) (Cultural and EVM Policy)
Con	Establish a clear definition of done targets for software metrics for defense systems of different types (code coverage, defect rate, user acceptance) (Cultural)
D&M	Congress could establish, via an NDAA provision, new data-driven methods for governance of software development, maintenance, and performance. The new approach should require on demand access to standard [and real-time?] data with reviews occurring on a standard calendar, rather than the current approach of manually developed, periodic reports. [dup]

D&M	DoD must establish the data sources, methods, and metrics required for better analysis, insight, and subsequent management of software development activities. This action does not require Congressional action but will likely stall without external intervention and may require explicit and specific Congressional requirements to strategically collect, access, and share data for analysis and decision making.
T&E	Establish requirements for government-owned software to be instrumented such that critical monitoring functions (e.g., performance, security, etc.) can be automated as much as possible, persistently available, and such that authoritative data can be captured, stored, and reused in subsequent testing or other analytic efforts.

Related recommendations from previous studies

DSB87	Rec 19: DoD should develop metrics and measuring techniques for software quality and completeness, and incorporate these routinely in contracts.
DSB87	Rec 20: DoD should develop metrics to measure implementation progress.
Sec 809	Rec 19: Eliminate the Earned Value Management (EVM) mandate for software programs using Agile methods
MITRE'18	Elevate Security as a Primary Metric in DoD Acquisition and Sustainment

Additional Recommendation D5 – Iterative Development

Line of Effort	Change the practice of how software is procured and developed
Recommendation	Shift the approach for acquisition and development of software (and software-intensive components of larger programs) to an iterative approach: start small, be iterative, and build on success or be terminated quickly.
Stakeholders	USD(A&S), USD(P&R), SAE, Service HR
Background	Current language DoD acquisition guidance is largely based around a hardware-centric paradigm, with a well-defined start and end and sequential lifecycle activities.
Desired State	Software acquisition in the DoD follows an iterative approach, with frequent deployment of working software, supported by a DevSecOps infrastructure that enables speed through continuous integration / continuous deployment. Software projects are continuously evaluated by the quality of their deployed capability and are terminated early if they are found to be non-performant. Software is never "complete". Programs are viewed as an ongoing service rather than a discrete project.
Congressional Role	Authorize and track software programs that utilize iterative methods of development rather than milestone-based progress. Recognizing that the distinction between RTD&E, procurement, and sustainment is not appropriate for many types of software, identify new ways of providing oversight while enabling much more flexibility for programs.

Draft Implementation Plan		Lead Stakeholders	Target Date
D4.1	Issue guidance immediately changing default for acquisition programs to be Agile DevSecOps	USD(A&S)	Q3 FY19
D4.2	Issue guidance immediately changing default for acquisition programs to be Agile DevSecOps	SAE	Q3 FY19
D4.6	Select three software programs widely perceived to be in dire straits and go through a program termination exercise to identify new potential solutions and the blockers to more effectively terminating non-performing programs.	USD A&S	Q1 FY20
D4.3	Modify DoDI 5000.02 and DoDI 5000.75 (or DoDI 5000.SW) to reflect more iterative approaches for software development	USD(A&S)	Q2 FY20
D4.4	Modify Service acquisition policy to reflect more iterative approaches for software development	SAE	Q2 FY20
D4.5	Build a Congressional Reporting Dashboard that would be available to the four Defense Committees to show the progress of DoD and Services DevSecOps programs, including speed and cycle time, code quality, security, and user satisfaction..	USD (A&S)	Q4 FY20

SWAP concept paper recommendations related to this recommendation

10C	Adopt a DevOps culture for software systems.
10C	All software procurement programs should start small, be iterative, and build on success – or be terminated quickly.
D&D	Accept 70% solutions in a short time (months) and add functionality in rapid iterations (weeks)
D&D	Take advantage of the fact that software is essentially free to duplicate, distribute, and modify
D&D	Treat software development as a continuous activity, adding functionality continuously across its life cycle

Visits	Spend time upfront getting the architecture right: modular, automated, secure
Visits	Start small, be iterative, and build on success – or terminate quickly

SWAP working group inputs related to this recommendation

Con	Treat procurements as investments “what would you pay for a possible initial capability”
Con	Leverage incentives to make smaller purchases to take advantage of simplified acquisition procedures
Con	Use modular contracting to allow for regular investment decisions based on perceived value
Con	Streamline acquisition processes to allow for replacing poor performing contractors
T&E	Develop the enterprise knowledge management and data analytics capability for rapid analysis/ presentation of technical RDT&E data to support deployment decisions at each iterative cycle.

Related recommendations from previous studies

DSB18	Rec 2b: [DoD programs should] establish MVP and the equivalent of a product manager for each program in its formal acquisition strategy, and arrange for the warfighter to adopt the initial operational capability (IOC) as an MVP for evaluation and feedback
DSB18	Rec 2a: [DoD programs should] develop a series of viable products (starting with MVP) followed by successive next viable products (NVPs);
DSB18	Rec 3a: The MDA (with the DAE, the SAE, the PEO, and the PM) should allow multiple vendors to begin work. A down-select should happen after at least one vendor has proven they can do the work, and should retain several vendors through development to reduce risk, as feasible.
NDU'17	Prioritize technical performance and project schedules over cost. Maintain aggressive focus on risk identification and management across all elements of the open system and resolve technical problems as rapidly as possible.
GAO'17	Follow an evolutionary path toward meeting mission needs rather than attempting to satisfy all needs in a single step.
GAO'17	Ensure that critical technologies are proven to work as intended before programs begin. Assign more ambitious technology development efforts to research departments until they are ready to be added to future generations (or increments) of a product.
OSD'06	Change DoD’s preferred acquisition strategy for developmental programs from delivering 100 percent performance to delivering useful military capability within a constrained period of time, no more than 6 years from Milestone A. This makes time a Key Performance Parameter.
OSD'06	Direct changes to the DoD 5000 series to establish Time Certain Development as the preferred acquisition strategy for major weapons systems development programs.

Additional Recommendation D6 – Machine Learning and AI Methodologies

Line of Effort	Change the practice of how software is procured and developed
Recommendation	Maintain an active research portfolio into next-generation software methodologies and tools, including the integration of machine learning and AI into software development, cost estimation, security vulnerabilities and related areas.
Stakeholders	USD(R&E), USD(A&S)
Background	Software is essential to national security and DoD needs to stay ahead of adversaries on emerging SW development practices.
Desired State	DoD benefits from a feedback loop between research and practice, in areas important to retaining the ability to be able to field innovations in software-enabled technologies: Mission needs and a practical understanding of the acquisition ecosystem inform research programs in emerging technologies. At the same time, results emerging from research can impact the department's warfighting and other systems thanks to high-quality and modular software systems, a DevSecOps infrastructure capable of moving fast, and other enablers.
Congressional Role	N/A

Draft Implementation Plan		Lead Stakeholders	Target Date
D6.1	Designate a responsible person or organization to sponsor software research.	USD(R&E)	Q4 FY19
D6.2	Stand up a Chief Engineer for Software to direct the implementation of next generation software methodologies and tools.	SAEs	Q4 FY19
D6.4	Direct the Principal Civilian Deputy to the SAE to implement the acquisition infrastructure for DevSecOps, allowing quick incorporation of new technologies into DoD systems, implemented by someone with software development experience	SAEs	Q4 FY19
D6.6	Create a documented DoD Software strategy, perhaps patterned on the DoD cyber strategy[19], with ties to other existing national and DoD research strategies, and with involvement of A&S and the Services.	USD(R&E)	Q4 FY19
D6.5	Make acquisition data collected continuously from DevSecOps infrastructure and tools available to researchers with appropriate clearances, as a testbed for AI, ML, or other technologies. (See Recs A6, D2)	USD(A&S)	Q4 FY20

Related recommendations from previous studies

DSB18	Rec 7a: Under the leadership and immediate direction of the USD(R&E), the Defense Advanced Research Projects Agency (DARPA), the SEI FFRDC, and the DoD laboratories should establish research and experimentation programs around the practical use of machine learning in defense systems with efficient testing, independent verification and validation (IVV), and cybersecurity resiliency and hardening as the primary focus points.

[19] https://media.defense.gov/2018/Sep/18/2002041658/-1/-1/1/CYBER_STRATEGY_SUMMARY_FINAL.PDF

Additional Recommendation D7 – Transition Emerging Approaches

Line of Effort	Change the practice of how software is procured and developed
Recommendation	Invest in transition of emerging approaches from academia and industry to creating, analysis, verification, and testing of software into DoD practice (via pilots, field tests, and other mechanisms).
Stakeholders	USD(A&S), USD(R&E), Service Digital PEOs
Background	Software is essential to national security and DoD needs to stay ahead of adversaries in implementing emerging SW development practices. Research work at universities and in the private sector, along with best practice implementation from the private sector can provide value tools and methods to be deployed across the DoD.
Desired State	Development and test technology, tools and methods that are being created and used in the private sector and academia are known and visible to the PEOs Digital who enable transition into service programs. DoD labs are investing internally and externally maturing software development and analysis tools.
Congressional Role	N/A

Draft Implementation Plan		Lead Stakeholders	Target Date
D7.1	Create community of practice, code repositories and other mechanism to keep all practitioners knowledgeable about latest trends and capabilities in software development, testing and deployment.	USD(A&S)	Q4 FY19
D7.2	Invest in and engage with academic and private sector efforts to transition tools to do software engineering: creating, analyzing, verifying, testing and maintaining software.	Service Digital PEOs	FY20
D7.3	Invest in and engage with academic and private sector efforts to transition tools to do software engineering: creating, analyzing, verifying, testing and maintaining software.	USD(R&E)	FY20

SWAP working group inputs related to this recommendation

Req	OSD should consider identifying automated software generation areas which can apply to specific domains

Related recommendations from previous studies

OSD'06	Direct the Deputy Director for Research and Engineering to coordinate service science and technology transition plans with the appropriate military service.
OSD'06	Direct the Deputy Director for Research and Engineering to actively participate in the Joint Capabilities Acquisition and Divestment process to reemphasize technology push initiatives.

Additional Recommendation D8 – Collect Data

Line of Effort	Change the practice of how software is procured and developed
Recommendation	Automatically collect all data from DoD national security systems, networks, and sensor systems and make available for machine learning (via federated, secured enclaves, not a centralized repository).
Stakeholders	USD(A&S), USD(P&R), SAE, Service HR; CMO, CAPE, DOT&E R&E/DT
Background	DoD discards or does not have access to significant amounts of data for its systems and has not established an infrastructure for storing data, mining data, or making data available for machine learning. Current analytical efforts are siloed and under-resourced in many cases.
Desired State	DoD has a modern architecture to collect, share, and analyze data that can be mined for patterns that humans cannot perceive. Data is being used to enable better decision-making in all facets of the Department, providing significant advantages that adversaries cannot anticipate. Data collection and analysis is done without compromising security and DoD, with minimum exceptions, should have complete data rights for all systems (developed with industry).
Congressional Role	N/A

Draft Implementation Plan		Lead stakeholders	Target Date
D8.1	Develop comprehensive data strategy for the DoD taking into account future AI/ML requirements	CDO with USD(A&S), SAE	Q1 FY20
D8.2	Implement a minimum viable product (MVP) that collects and analyzes the most critical data element for 1 or more programs	CDO with USD(A&S), SAE	Q3 FY20
D8.3	Create digital data infrastructure to support collection, storage, and processing	CDO with USD(A&S), SAE	Q1 FY21
D8.4	Require that all new major systems should specify a data collection and delivery plan.	A&S	Q2 FY21
D8.5	Implement data collection requirements for new sensor and weapon system acquisition	A&S	FY21

SWAP concept paper recommendations related to this recommendation

10C	All data generated by DoD systems - in development and deployment - should be stored, mined, and made available for machine learning.

Related recommendations from previous studies

DSB18	Rec 7b: [USD(R&E)] should establish a machine learning and autonomy data repository and exchange along the lines of the U.S. Computer Emergency Readiness Team (US-CERT) to collect and share necessary data from and for the deployment of machine learning and autonomy.
DSB18	Rec 7c: [USD(R&E)] should create and promulgate a methodology and best practices for the construction, validation, and deployment of machine learning systems, including architectures and test harnesses.

Appendix B: Legislative Template Language and Changes for Recommendations

v1.1, 12 Mar 2019

Appendix B.1 provides a template for the type of legislative language that could represent a new category/pathway to procure, develop, deploy and continuously improve software for DoD applications. This template is designed to serve as an example of how the types of changes we envision might be implemented and has not been reviewed or endorsed by the Department. It is written to be consistent with 2016 NDAA Section 805 (Use of alternative acquisition paths to acquire critical national security capabilities).

Appendix B.2 provides possible legislative changes that can be used to support the individual recommendations in the report, focused on the Top Ten Recommendations listed in the Recommendations Cheat Sheet and in Chapter 5.

Appendix B.1: Response to 2016 NDAA Section 805
Template Language for Recs A1, A2

v1.1, 12 Mar 2019

This document is a template for the type of legislative language that could represent a new category/pathway to procure, develop, deploy and continuously improve software for DoD applications. This template is designed to serve as an example of how the types of changes we envision might be implemented and has not been reviewed or endorsed by the Department. It is written to be consistent with 2016 NDAA Section 805 (Use of alternative acquisition paths to acquire critical national security capabilities).

SEC. [???]. SPECIAL PATHWAYS FOR RAPID ACQUISITION OF SOFTWARE APPLICATIONS AND UPGRADES.

(a) GUIDANCE REQUIRED.—Not later than [90, 180, 270] days after the date of the enactment of this Act, the Secretary of Defense shall establish guidance authorizing the use of special pathways for the rapid acquisition of software applications and upgrades that are intended to be fielded within one year.

(b) SOFTWARE ACQUISITION PATHWAYS.—

(1) The guidance required by subsection (a) shall provide for the use of proven technologies and solutions to continuously engineer and deliver capabilities in software. The objective of an acquisition under this authority shall be to begin the engineering of new capabilities quickly, to demonstrate viability and effectiveness of those capabilities in operation, and continue updating and delivering new capabilities iteratively afterwards. An acquisition under this authority shall not be treated as an acquisition program for the purpose of section 2430 of title 10, United States Code or Department of Defense Directive 5000.01.

(2) Such guidance shall provide for two rapid acquisition pathways:

(A) APPLICATIONS.—The applications software acquisition pathway shall provide for the use of rapid development and implementation of applications and other software and software improvements running on commercial commodity hardware (including modified or ruggedized hardware) operated by the Department; and

(B) EMBEDDED SYSTEMS.—The embedded systems software acquisition pathway shall provide for the rapid development and insertion of upgrades and improvements for software embedded in weapon systems and other military-unique hardware systems.

(c) EXPEDITED PROCESS.--

(1) IN GENERAL.—The guidance required by subsection (a) shall provide for a streamlined and coordinated requirements, budget, and acquisition process that results in the rapid fielding of software applications and software upgrades to embedded systems in a period of not more than [one year] from the time that the process is initiated. It shall also require the collection of data on the version fielded and continuous engagement with the users of that software, so as to enable engineering and delivery of additional versions in periods of not more than one year each.

(2) EXPEDITED SOFTWARE REQUIREMENTS PROCESS. –

(A) Software acquisitions conducted under the authority of this provision shall not be subject to the Joint Capabilities Integration and Development System Manual and Department of Defense Directive 5000.01, except to the extent specifically provided in the guidance required by subsection (a).

(B) The guidance required by subsection (a) shall provide that –

(1) Requirements for covered acquisitions are developed on an iterative basis through engagement with the user community, and utilization of user feedback in order to regularly define and prioritize the software requirements, as well as to evaluate the software capabilities acquired;

(2) The requirements process begins with the identification of 1) the warfighter or user need, 2) the rationale for how these software capabilities will support increased lethality and/or efficiency, and 3) the identification of a relevant user community;

(3) Initial contract requirements are stated in the form of a summary-level list of problems and shortcomings in existing software systems and desired features or capabilities of new or upgraded software systems;

(4) Contract requirements are continuously refined and prioritized in an evolutionary process through discussions with users that may continue throughout the development and implementation period;

(5) Issues related to lifecycle costs and systems interoperability are considered; and

(6) Issues of logistics support in cases where the software developer may stop supporting the software system are addressed.

(3) RAPID CONTRACTING MECHANISM.— The guidance required by subsection (a) shall authorize the use of a rapid contracting mechanism, pursuant to which --

(A) a contract may be awarded within a [90-day] period after proposals are solicited on the basis of statements of qualifications and past performance data submitted by contractors, supplemented by discussions with two or more contractors determined to be the most highly-qualified, without regard to price;

(B) a contract may be entered for a period of not more than one-year and a ceiling price of not more than [$50 million] and shall be treated as a contract for the acquisition of commercial services covered by the preference in section 2377 of title 10, United States Code;

(C) a contract shall identify the contractor team to be engaged for the work, and substitutions shall not be made during the base contract period without the advance written consent of the contracting officer;

(D) the contractor may be paid during the base contract period on a time and materials basis up to the ceiling price of the contract to review existing software in consultation with the user community and utilize user feedback to define and prioritize software requirements, and to design and implement new software and software upgrades, as appropriate;

(E) a contract may provide for a single one-year option to complete the implementation of one or more specified software upgrades or improvements identified during the period of the initial contract, with a price of not more than [$100 million] to be negotiated at the time that the option is awarded; and

(F) an option under the authority of this section may be entered on a time and materials basis and treated as an acquisition of commercial services or entered on a fixed price basis and treated as an acquisition of commercial products, as appropriate.

(4) EXECUTION OF RAPID ACQUISITIONS. The Secretary shall ensure that —

(A) software acquisitions conducted under the authority of this provision are supported by an entity capable of regular automated testing of the code, which is authorized to buy storage, bandwidth, and computing capability as a service or utility if required for implementation;

(B) processes are in place to provide for collection of testing data automatically from [entity specified in (A)] and using those data to drive acquisition decisions and oversight reporting;

(C) the Director of Operational Test and Evaluation and the director of developmental test and evaluation participate with the acquisition team to design acceptance test cases that can be automated using the entity specified in (A) and regularly used to test the acceptability of the software as it is incrementally being engineered;

(D) acquisition progress is monitored through close and regular interaction between government and contractor personnel, sufficient to allow the government to understand progress and quality of the software with greater fidelity than provided by formal but infrequent milestone reviews;

(E) an independent, non-advocate cost estimate is developed in parallel with engineering of the software, and is based on an investment-focused alternative to current estimation models, which is not based on source lines of code;

(F) the performance of fielded versions of the software capabilities are demonstrated and evaluated in an operational environment; and

(G) software performance metrics addressing issues such as deployment rate and speed of delivery, response rate such as the speed of recovery from outages and cybersecurity vulnerabilities, and assessment and estimation of the size and complexity of software development effort are established that can be automatically generated on a [monthly, weekly, continuous] basis and made available throughout the Department of Defense and the congressional defense committees.

(5) ADMINISTRATION OF ACQUISITION PATHWAY.—The guidance for the acquisitions conducted under the authority of this section may provide for the use of any of the following streamlined procedures in appropriate circumstances:

(A) The service acquisition executive of the military department concerned shall appoint a project manager for such acquisition from among candidates from among civilian employees or members of the Armed Forces who have significant and relevant experience in modern software methods.

(B) The project manager for each large software acquisition as designated by the service acquisition executive shall report with respect to such acquisition directly, and without intervening review or approval, to the service acquisition executive of the military department concerned.

(C) The service acquisition executive of the military department concerned shall evaluate the job performance of such manager on an annual basis. In conducting an evaluation under this paragraph, a service acquisition executive shall consider the extent to which the manager has achieved the objectives of the acquisition for which the manager is responsible, including quality, timeliness, and cost objectives.

(D) The project manager shall be authorized staff positions for a technical staff, including experts in software engineering to enable the manager to manage the acquisition without the technical assistance of another organizational unit of an agency to the maximum extent practicable.

(E) The project manager shall be authorized, in coordination with the users of the equipment and capability to be acquired and the test community, to make trade-offs among life-cycle costs, requirements, and schedules to meet the goals of the acquisition.

(F) The service acquisition executive or the defense acquisition executive in cases of defense wide efforts, shall serve as the decision authority for the acquisition.

(G) The project manager of a defense streamlined acquisition shall be provided a process to expeditiously seek a waiver from Congress from any statutory or regulatory requirement that the project manager determines adds little or no value to the management of the acquisition.

(6) OTHER FLEXIBLE ACQUISITION METHODS. – The flexibilities provided for software acquisition pathways under this section do not preclude the use of acquisition flexibilities otherwise available for the acquisition of software. The Department may use other transactions authority, broad agency announcements, general solicitation competitive procedures authority under section 879 of the National Defense Authorization Act for Fiscal Year 2017, the challenge program authorized by section 2359b of title 10, United States Code, and other authorized procedures for the acquisition of software, as appropriate. Such authorities may be used either in lieu of or in conjunction with the authorities provided in this section.

(d) FUNDING MECHANISMS. --

(1) SOFTWARE FUND.—

(A) IN GENERAL.—The Secretary of Defense shall establish a fund to be known as the [‘‘Department of Defense Rapid Development of Effective Software Fund’’] to provide funds, in addition to other funds that may be available for acquisition under the rapid software development pathways established pursuant to this section. The Fund shall be managed by a senior official of the Department of Defense designated by the [Under Secretary of Defense for Acquisition and Sustainment]. The Fund shall consist of amounts appropriated to the Fund and amounts credited to the Fund pursuant to section [???] of this Act.

(B) TRANSFER AUTHORITY.—Amounts available in the Fund may be transferred to a military department for the purpose of starting an acquisition under the software acquisition pathway established pursuant to this section. These funds will be used to fund the first year of the software acquisition and provide the Department an opportunity to field software capabilities that address newly discovered needs. A decision to continue the acquisition on other funds will be made based upon the progress demonstrated after the first year. Any amount so transferred shall be credited to the account to which it is transferred. The transfer authority provided in this subsection is in addition to any other transfer authority available to the Department of Defense.

(C) CONGRESSIONAL NOTICE.—The senior official designated to manage the Fund shall notify the congressional defense committees of all transfers under paragraph (2). Each notification shall specify the amount transferred, the purpose of the transfer, and the total projected cost and funding based on the effort required each year to sustain the capability to which the funds were transferred. The senior official will also notify the congressional defense committees at the end of the one-year timeframe and report on the fielded capabilities that were achieved. A notice under this paragraph shall be sufficient to fulfill any requirement to provide notification to Congress for a new start.

(2) PILOT PROGRAM. The Secretary may conduct a pilot program under which funding is appropriated in a single two-year appropriation for lifecycle management of software-intensive and infrastructure technology capabilities conducted under the authority of this section. The objective of the appropriation software pilot program would be to provide 1) greater focus on managed services versus disaggregated development efforts, 2) additional accountability and transparency for information centric and enabling technology capabilities, and 3) flexibility to pursue the most effective solution available at the time of acquisition; 4) much greater insight into the nature of software expenditures across the DOD enterprise; 5) an improved ability to measure costs and program performance;

Appendix B.2: Additional Legislative and Regulatory Changes for Top Ten Recommendations

v0.1, 12 Mar 2019

B.2. Legislative changes supporting top ten recommendations

The following sections provide possible legislative changes that can be used to support the recommendations in the report, focused on the Top Ten Recommendations listed in the Recommendations Cheat Sheet and Chapter 5.

Recommendation A1. Establish new acquisition pathway(s) for software that prioritizes continuous integration and delivery of working software in a secure manner, with continuous oversight from automated analytics

See Appendix B.1, Section (a)-(c).

Recommendation A2. Create a new appropriations category that allows (relevant types of) software to be funded as a single budget item, with no separation between RDT&E, production, and sustainment.

See Appendix B.1, Section (d).

Topic	**Digital Technology Management Appropriation**
Subgroup	Appropriations
Background	DoD continues to acquire and fund information-centric systems using processes designed for hardware-centric platforms. Current funding decision processes and data structures do not effectively support leading software development practices. Differentiating continuous iteration and continuous delivery of software workload into hardware-defined phases (Research, Development, Test & Evaluation (RDT&E), Procurement, or Operations and Maintenance (O&M)) hinders the ability to deliver holistic capabilities and services and does not enable real-time resource, requirements, performance, and schedule trades across systems without significant work.
Statute, Regs	FY20 NDAA
Changes	Proposed Language for New Appropriation: For expenses necessary for the continuous lifecycle management (requirements, research, development, test and evaluation, procurement, production, modification, and operation and maintenance) of software, software-based services and supporting technologies to include requisite hardware for Department of Defense business and information warfare capabilities, as authorized by law. [$000,000,000] to remain available for obligation until [September 30, 2022]: Provided, that the funds are available for software, and electronic tools, systems, applications, resources, or an applicable emerging technology, acquisition of services, business process re-engineering activities, functional requirements development, technical evaluations, and other activities in direct support of acquiring, developing, deploying, sustaining, enhancing, and modernizing software and information technology capabilities. Proposed Language for New Authorization: To further enable HR 2810's intent of

	streamlining and improving the efficiency and effectiveness of software acquisition in order to maintain defense technology advantage, funds are hereby authorized to be appropriated for fiscal year 2021 for use by the Armed Forces and other activities and agencies of the Department of Defense for expenses, not otherwise provided for, for a new Department of Defense appropriation within the [TBD if this resides under an established appropriation, or new] title called Digital Technology Management, as specified in the funding table in section [???]. This appropriation replaces all other appropriation types used in executing lifecycle management of software, software-based services, and supporting technologies to include requisite hardware for Department of Defense business and information warfare capabilities. The Department of Navy Information Warfare Capabilities and Department of Air Force Business Operations Capabilities will pilot the effort, and if successful will be expanded in 2022 to all information centric capability areas across the Department of Defense. The Department of Navy and the Department of Air Force shall identify the affected programs and transition all existing funds—RDT&E, Procurement, and O&M to the new appropriation for execution in FY 2021. Reporting to Congress continues using the existing [TBD or new] budget display. Baseline and Progress reporting on the effectiveness of the appropriation structure in executing shall be reported semi-annually to the Defense Appropriations Committees.
Sec 805?	Yes, covered by response to Section 805 template language

Recommendation B1. Establish and maintain digital infrastructure within each Service or Agency that enables rapid deployment of secure software to the field and make available to contractors at subsidized cost.

No legislative changes needed.

Recommendation B2. Create, implement, support, and require a fully automatable approach to test and evaluation (T&E), including security, that allows high confidence distribution of software to the field on an iterative basis (with frequency dependent on type of software, but targets cycle times measured in weeks).

Topic	**Live fire / survivability / lethality testing**
Subgroup	Acquisition Strategy
Background	There is no exemption for software-intensive programs to conduct survivability/ lethality/live fire testing to move beyond LRIP OR to modify these requirements to reflect their nature as software intensive programs. Any covered system may require Live Fire Test & Evaluation (LFT&E). Includes major systems in the definition which may or may not be software programs (per the § 2302 definition). Otherwise, a waiver must be sent to the Congressional committees before Milestone B.
Statute, Regs	10 USC §2366 and DoDI 5000.02
Changes	First, elimination of the Major Systems from Title 10 U.S.C. § 2302 helps to solve the identified challenges. Further, consider language for Title 10 2366a which allows exemption for software intensive programs, where DOT&E must justify adding the program for oversight with the MDA and must streamline the process.
Related Recs	A1, B2

Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Statutory DOT&E authority**
Subgroup	Acquisition Strategy
Background	OT&E has been able to essentially stop programs as they move through the development (acquisition) process. OT&E testers are often not SMEs in the systems they are conducting testing oversight which can negatively impact testing. 1. Statutory authority assumes use of waterfall methodology; relies on infrequent, major test events instead of the continuous testing that agile uses. 2. Also assumes a separate test team (and even organization) as opposed to testers being embedded in an agile team.
Statute, Regs	Title 10 U.S.C. §2399
Changes	1. DOT&E oversight is only when requested by the SAE or USD(A&S), or Congressionally directed, unless MDAP. 2. DOT&E will utilize, to the greatest extent possible, test data collected through existing test methodologies present in the program and will not recommend or prescribe additional independent one-time test events. 3. One time IOT&Es or cybersecurity test events will not be recommended for software intensive systems unless in specific circumstances if warranted
Related Recs	A1, B2
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**DoD Interoperability Policy**
Subgroup	Acquisition Strategy
Background	Directs various things that should be reconsidered for IT/Software: 1. NR KPP required 2. DoD specific architecture products in the DoDAF format which are labor intensive and of questionable value 3. Interoperability Support Plans (ISPs) required, where DoD CIO can declare any ISP of “special interest” 4. Requires DT authority to provide assessments at MS C 5. Mandates JITC to do interoperability assessments for IT with “joint, multinational, and interagency interoperability requirements”
Statute, Regs	DoDI 8330.01
Changes	Direct revision of DoDI 8330.01 or, potentially, elimination of it
Related Recs	A1, A7, B2, D3
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Cyber testing**

Subgroup	Testing and Evaluation
Background	T&E must strive for continuous software testing, automated and integrated into the development cycle to the fullest extent possible, across the entirety of the DoD’s software portfolio.
Changes	Establish a statutory ("Live Fire") requirement on software-intensive systems as there is on "Covered Systems" for protecting our warfighters from kinetic threats. “Shoot at it” before design is complete and certainly before it is put into the operational environment.
Related Recs	B2, D2
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Cyber testing infrastructure**
Subgroup	Testing and Evaluation
Background	DoD lacks the enterprise digital infrastructure needed to test the broad spectrum of software types and across the span of T&E to support developmental efficiency (in DT) and operational effectiveness (in OT).
Changes	Establish a federation of state-of-the-art cyber testing capabilities from non-profit institutions to support trusted, survivable and resilient defense systems and ensure the security of software and hardware developed, acquired, maintained, and used by DoD.
Related Recs	B2, D2, E2
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Cyber policies**
Subgroup	Testing and Evaluation
Background	T&E must strive for continuous software testing, automated and integrated into the development cycle to the fullest extent possible, across the entirety of the DoD’s software portfolio.
Statute, Regs	N/A
Changes	Develop mechanisms to enforce existing software and cyber security policies (from cradle-to- grave) that are not (now) being adequately enforced.
Related Recs	B2, D2
Sec 805?	Yes, covered by response to Section 805 template language

Recommendation B3. Create a mechanism for ATO reciprocity within and between services to enable sharing of software platforms, components and infrastructure and rapid integration of capabilities across (hardware) platforms, (weapons) systems, and Services.

No legislative changes needed.

Recommendation C1. Create software development groups in each Service consisting of military and/or civilian personnel who write code that is used in the field and track individuals who serve in these groups for future DoD leadership roles.

Topic	**Core capabilities**
Subgroup	Sustainment and Modernization
Background	Title 10 USC 2464 establishes a key imperative for DoD to establish core Government Owned Government Operated (GOGO) capabilities as a ready and controlled source of technical competence and resources for national security. DoD's focus has traditionally been on hardware and therefore there has seen significant Service and DoD enterprise focus on hardware GOGO capabilities and infrastructure for core. However, there has been significantly less upfront acquisition focus and visibility on what core means for software intensive systems and the associated GOGO software engineering capability.
Statute, Regs	Title 10 USC 2464
Changes	The definition of "core capabilities" in 10 USC 2464 should be revisited in light of warfighter dependence on software intensive systems to determine the scope of DoD's core organic software engineering capability.
Related Recs	C1, D1
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Establishment of a Core "Digital Delivery" Occupational Series**
Subgroup	Workforce
Background	Develop a core occupational series based on current core competencies and skills for software acquisition and engineering.
Statute, Regs	Title 10, §1607, §1721
Changes	Modifying Existing Language - Title 10, §1721. Need to add this Core Occupational Series to the list of "Designation of Acquisition Positions" or Consider Using Existing Language: Title 10, §1607 to add this occupational series fit within this established Defense Intelligence Senior Level model.
Related Recs	C1
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Empower Implementation Cadre**
Subgroup	Workforce
Background	Empower a small cadre of Highly Qualified Experts and innovative Department employees to execute changes from this report
Changes	New Legislation - This will be critical to avoid a repeat of the past 35+ years of continuous admiration of the problem.
Related Recs	C1
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Private-Public Sector Fluidity**
Subgroup	Workforce
Background	Encourage greater private-public sector fluidity within its workforce. Federal employees who come from the private sector bring with them best practices, modern methodologies, and exposure to new technologies. Federal employees who leave bring their understanding of the DoD mission and constraints.
Statute, Regs	Title 5, §§3371-3375, Title 10, §1599g
Changes	Modification Language - Title 5, §§3371-3375: Expand the Inter-Government Personnel Act and allow more civil service employees to work with non-Federal Agencies and Educational Institutions. Modification Language - Title 10, §1599g: Expand the Public-Private Talent Exchange Program and modify the language to reduce the “repayment” period from 1:2 to 1:1 ratio.
Related Recs	C1
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Computer Language Proficiency Pay**
Subgroup	Workforce
Background	TBD
Statute, Regs	Title 10, §1596a
Changes	New Language - Title 10, §1596a - Use this language to create a new Computer-language proficiency pay statute.
Related Recs	C1
Sec 805?	Yes, covered by response to Section 805 template language

Topic	Pilot a Cyber Hiring Team
Subgroup	Workforce
Background	TBD
Changes	New Legislation - Team will have all the necessary authorities to execute recommendations called out in this report. The team will serve as a Department-wide alternative to organization's traditional HR offices and will provide expedited hiring and a better candidate experience for top tier cyber positions.
Related Recs	C1
Sec 805?	Yes, covered by response to Section 805 template language

Recommendation C2. Expand the use of (specialized) training programs for CIOs, SAEs, PEOs, and PMs that provide (hands-on) insight into modern software development (e.g., agile, DevOps, DevSecOps) and the authorities available to enable rapid acquisition of software.

Topic	Establish Workforce Fund
Subgroup	Workforce
Background	Create a software acquisition workforce fund similar to the existing Defense Acquisition Workforce Development Fund (DAWDF).
Changes	New Legislation, similar to DAWDF, but the primary use will be for hiring and training a cadre of modern software acquisition experts.
Related Recs	C2
Sec 805?	Yes, covered by response to Section 805 template language

Recommendation D1. Require access to source code, software frameworks, and development toolchains, with appropriate IP rights, for all DoD-specific code, enabling full security testing and rebuilding of binaries from source.

Topic	Development Tools and Data
Subgroup	Workforce
Background	As discussed in appendices of this report, the existing data that the Department collects regarding software acquisition is insufficient for answering many questions of interest. A more robust data set would also allow the application of Artificial Intelligence/Machine Learning techniques that could provide value.
Changes	SECDEF shall establish a policy for the capture and use of software acquisition data, and standards for the use of tools that could automate the data collection.

Related Recs	D1, D2, D5, D7, D8
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Core capabilities**
Subgroup	Sustainment and Modernization
Background	Title 10 USC 2464 establishes a key imperative for DoD to establish core Government Owned Government Operated (GOGO) capabilities as a ready and controlled source of technical competence and resources for national security. DoD's focus has traditionally been on hardware and therefore there has seen significant Service and DoD enterprise focus on hardware GOGO capabilities and infrastructure for core. However, there has been significantly less upfront acquisition focus and visibility on what core means for software intensive systems and the associated GOGO software engineering capability.
Statute, Regs	Title 10 USC 2464
Changes	The definition of "core capabilities" in 10 USC 2464 should be revisited in light of warfighter dependence on software intensive systems to determine the scope of DoD's core organic software engineering capability.
Related Recs	C1, D1
Sec 805?	Yes, covered by response to Section 805 template language

Recommendation D2. Make security a first-order consideration for all software-intensive systems, under the assumption that security-at-the-border will not be enough.

Topic	**Cyber testing**
Subgroup	Testing and evaluation
Background	T&E must strive for continuous software testing, automated and integrated into the development cycle to the fullest extent possible, across the entirety of the DoD's software portfolio.
Statute, Regs	TBD
Changes	Establish a statutory ("Live Fire") requirement on software-intensive systems as there is on "Covered Systems" for protecting our warfighters from kinetic threats. "Shoot at it" before design is complete and certainly before it is put into the operational environment.
Related Recs	B2, D2
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Cyber testing infrastructure**
Subgroup	Testing and Evaluation
Background	The DoD lacks the enterprise digital infrastructure needed to test the broad spectrum of software types and across the span of T&E to support developmental efficiency (in DT) and operational effectiveness (in OT).
Changes	Establish a federation of state-of-the-art cyber testing capabilities from non-profit institutions to support trusted, survivable and resilient defense systems and ensure the security of software and hardware developed, acquired, maintained, and used by the DoD.
Related Recs	B2, D2, E2
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Acquisition metrics**
Subgroup	Testing and Evaluation
Background	The DoD lacks the enterprise data management and analytics capability needed to support the evaluation of test data in accordance with the pace of modern iterative software methods.
Changes	Establish cyber security as the “4th leg” in measurement of Acquisition system/program performance: Cost, Schedule, Performance, and Cyber Security.
Related Recs	A3, D2
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Cyber policies**
Subgroup	Testing and Evaluation
Background	T&E must strive for continuous software testing, automated and integrated into the development cycle to the fullest extent possible, across the entirety of the DoD’s software portfolio.
Changes	Develop mechanisms to enforce existing software and cyber security policies (from cradle-to- grave) that are not (now) being adequately enforced.
Related Recs	B2, D2
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Cyber threat models**
Subgroup	Testing and Evaluation
Background	The DoD lacks the resources needed to adequately emulate advanced cyber adversaries, to support fielding of trusted, survivable, and resilient software-intensive defense systems.
Changes	Ensure each DoD Component is responsible for representing its own forces and capabilities in a digital modeling environment (e.g., M&S, digital twin, etc.), making them available to all other DoD users, subject to a pre-defined architecture and supporting standards. DIA will represent threat forces and capabilities in a digital form consistent with these architecture/standards. Programs would be required to use DIA-supplied threat models, unless sufficient justification is provided to use others.
Related Recs	D2, E3
Sec 805?	Yes, covered by response to Section 805 template language

Recommendation D3. Shift from a list of requirements for software to a list of desired features and required interfaces/characteristics, to avoid requirements creep, overly ambitious requirements, etc.

Topic	**DoD Interoperability Policy**
Subgroup	Acquisition Strategy
Background	Directs various things that should be reconsidered for IT/Software: 1. NR KPP required 2. DoD specific architecture products in the DoDAF format which are labor intensive and of questionable value 3. Interoperability Support Plans (ISPs) required, where DoD CIO can declare any ISP of “special interest” 4. Requires DT authority to provide assessments at MS C 5. Mandates JITC to do interoperability assessments for IT with “joint, multinational, and interagency interoperability requirements”
Statute, Regs	DoDI 8330.01
Changes	Direct revision of DoDI 8330.01 or potentially elimination of it
Related Recs	A1, A7, B2, D3
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**MDAPs**
Subgroup	Acquisition Strategy
Background	Specific to the establishment of cost, fielding, and performance goals for MDAPs under section 2448a of title 10 introduced by Section 807 of the FY17 NDAA. Does not distinguish software intensive programs from any other type of program. Also this provision was a reaction to programs not following guidance for affordability already established in the DODI 5000.02.
Statute, Regs	Title 10 § 2448a through Section 807 of the FY17 NDAA
Changes	Eliminate this provision from statute. There is policy which already exists that covers this in the DoDI 5000.02 (note: DSD just signed out a memo on this)
Related Recs	A1, A3, A6, D3
Sec 805?	Yes, covered by response to Section 805 template language

Topic	**Configuration Steering Boards (CSB)**
Subgroup	Acquisition Strategy
Background	Must occur on at least an annual basis per the current statute (MDAPs). The Services tend to implement them for programs other than MDAPs based on 5000.02, and long-standing cultural factors.
Statute, Regs	FY 2009 NDAA, section 814; DoDI 5000.02
Changes	Other boards (or equivalent entities) established by the CAE or as delegated, the PEO or PM may fulfill the requirement of the CSB as long as the board (or equivalent entity) meets at least once a year and addresses the requirements in (c)(1).
Related Recs	A1, A6, A7, D3
Sec 805?	Yes, covered by response to Section 805 template language

WORKING DOCUMENT//DRAFT

Software is Never Done:
Refactoring the Acquisition Code for Competitive Advantage

Defense Innovation Board

SUPPORTING INFORMATION

v1.0, 19 Feb 2019

This document contains the supporting information for the Defense Innovation Board (DIB) Software Acquisition and Practices (SWAP) study. This information is in preliminary form and should be read along with the main (draft) report.

Contents:

Appendix B.1: Acquisition Strategy Subgroup Report

8 February 2019

This appendix examines pain points, obstacles, change ideas, and future vision for the Defense Innovation Board (DIB) Software Acquisition and Practices (SWAP) Study in the area of Acquisition Strategy and Oversight (i.e., *Acquisition Environment*). In 2017 the Office of the DASD(C3CB) under the ASD(A) commissioned an IT acquisition study with Deloitte. The study recommended the following attributes of an effective and efficient IT acquisition structure:

- *Fast* to incorporate current technology and make efficient use of Agency resources
- *Flexible* and adaptable to support rapid changes in technology and input from stakeholders about capability needs
- *Collaborative* to seek stakeholder involvement and input to be incorporated throughout

In a previous study completed in September 2016, Deloitte also provided key findings on commercial IT practices. Findings were taken into consideration when forming the proposals following in this appendix. The team recognizes that DoD is falling short of the preferred attributes outlined above with the current IT acquisition structure, in addition to multiple statutory, regulatory, and cultural issues that currently hinder an effective and efficient DoD acquisition environment that would benefit from reform.

Pain points

Acquisition Policy Environment. The DoD lacks a cohesive acquisition policy architecture and robust policy for software acquisition. Existing policies, to include tangential or supplemental policies that are integral to the operation of the defense acquisition system, do not fit well together and result in discrepancies, conflicts, and gaps. The defense acquisition system is monolithic, compiled in pieces as needs arose instead of as an integrated and evolving environment. It has proven unable to keep up with or remain ahead of the pace of change and technological advancements that require speed and agility. While it has regularly been revised, the changes tend to be conservative and incremental, requiring the agreement of too many parties protecting narrow interests and who are reluctant to relinquish authority or evolve. The system remains focused on oversight and situational control rather than insight and trust. The policies, practices, and documents become quickly entrenched and manifest themselves in the form of the Department's culture, leading to additional bureaucracy and decreased levels of organizational trust, that are difficult to rapidly reverse. Furthermore, the environment is risk averse, seeking out what is perceived to be the "safest" route to get things done, stifling the innovation and risk- taking that's required to maintain an advantage over adversaries.

As an example, one DoD weapons system program, which is implementing a DevSecOps pipeline to enable agile capability releases, informed us it took 18 months to get approval of a Test and Evaluation Master Plan (TEMP). The process within the TEMP drove them into sequential developmental and operational test - which is antithetical to continuous delivery under the DevSecOps concept.

Governance and Management. The Department lacks a strategic approach that recognizes software's criticality as the backbone and nervous system of the Department's mission and operations, often leading to widespread duplication of capabilities that could be consolidated and scaled at an enterprise level (whether Service-enterprise or OSD-enterprise). This absence of any strategy, compounded by a long-

standing lack of organizational trust in the Department, is exemplified by various situations in the software environment. For example, the lack of reciprocity on matters such as security standards, architecture, and compliance methods – my way is “better” (insert “less expensive,” “more efficient,” “more effective”) than your way, or, “our requirements / processes are unique,” regardless of validity. Further, the DoD issues separate policies on matters such as cloud, architecture, and risk management, with no unified approach at the strategic level. Management and governance of these matters takes the form of prolific numbers of senior working groups (or equivalent) that make few decisions but have frequent meetings. The DoD’s lack of an overarching strategic plan for key technologies, with a robust decision making framework that pushes responsibility and authority down to the lowest executable level, creates inefficiency, duplication, and waste.

Organization and Culture. The DoD lacks an organizational structure with clear responsibility and authority for software acquisition and management; there are confusing roles and responsibilities between DoD CIO, USD(A&S), and the DoD CMO. This state of ambiguity leads to overlap, inefficiency, and unnecessary bureaucracy; and it is replicated at the Service level. The result is a slow, rigid, siloed organization unable to adapt in the present and plan for the future in order to maintain competitive advantage. The DoD is not a change-ready environment and the acquisition system was not designed for rapid change. DoD employees tend to receive change mandates rather than participating in them. A case in point is that when DoD issues a policy, the Services will implement their own supporting version or “supplemental guidance”, which expands the policy and introduces multiple layers of bureaucracy, eliminating any semblance of flexibility that was intended by the original policy issued. For example, the Department issued DoD Instruction 5000.75 in February 2018, a tailored requirements and acquisition approach for business systems. Subsequently, the Army produced accompanying implementation guidance – 91 pages – which introduces additional forms, templates, processes, and time constraints.

Desired (end) state An acquisition system that enables rapid delivery of cost-efficient, relevant software capability through the application of creative compliance and fact-based critical thinking under a logical and minimal policy framework. The Department treats software as a national security capability and continuously retrains the workforce to be able to adapt to an ever-changing technology environment, embraces continuous collaboration between user and developers, embraces changing requirements, accepts and take risks, and deliver adversary- countering capabilities to the warfighter. Executing the approach requires an end state with an efficient contracting environment; a culture that rewards informed risk-taking and fast failures; the use of limits or guardrails instead of prescriptive requirements that limit creativity; outcome-based metrics that focus on value vs. execution against a plan; and a move away from traditional funding models and compliance-driven management.

Obstacles The Department operates with a general lack of urgency regarding its software – it is not recognized or treated as a national security capability. There is an aversion to informed risk-taking regarding new and innovative approaches to doing business and adopting emerging (or even simply relevant) technologies, even though it’s risky, or riskier, to continue using outdated technologies that are not secure or facing obsolescence in the face of evolving threats. Dramatic changes in policy or process are viewed as risky yet our current ways of operation are not despite a known degradation in strategic advantage previously enjoyed over adversaries. The inability to evolve and support rapid changes in technology and input from stakeholders about capability needs is bred through organizational silos and stovepipes that stifle the collaboration necessary to develop and operationalize software. Further, stakeholder involvement is limited by following restrictive controls, timelines, and processes in a sequential manner that impedes progress and results in a lower state of readiness. The duplication of

authorities and responsibilities among organizations both horizontally and vertically, within the defense acquisition system only exacerbates an already complex environment where a protectionist culture is ingrained and the workforce is not incentivized to change. In its endeavors to improve the status quo, "help" from Congress over the past decades translates into entrenched policies, processes, and procedures – "cultural norms" that are difficult to reverse.

Ideas for Change

*Acquisition Policy Environment***.** Define software as a critical national security capability under Section 805 of FY16 NDAA "Use of Alternative Acquisition Paths to Acquire Critical National Security Capabilities". Create an acquisition policy framework that recognizes that software is ubiquitous and will be part of all acquisition policy models. Recommend the creation of a clear, efficient acquisition path for acquiring non-embedded software capability. Reconcile and resolve discrepancies among supplemental policies that lead to conflicts. Consider the following tenets in development of a reformed software acquisition policy:

- Emphasis on quickly delivering working software
- Encourage projects and pilot efforts that serve to reduce risk and complexity - fail fast
- Reimagine program structures and program offices – i.e., accommodate move to "as-a-service" capabilities, agile, micro-services, and micro-applications
- Iterative, incremental development practices based on agile methods
- Rapid adoption of emerging technologies through piloting or prototyping
- Elimination of traditional A, B, C milestones; replaced by more sprint-centric decision points
- Elimination of arbitrary phases or merge phases to reflect rapid, agile development methods
- Tailor in requirements (statutory, regulatory – i.e., documentation) rather than tailor out; start with a minimum set
- No big-bang testing with sequential DT/OT; move to fully integrated test approaches driven by automated testing as well as regular, automated cybersecurity scanning
- Use a "guardrail-based" (upper / lower limit) approach for software requirements rather than defining every requirement up front
- Track value-driven outcome metrics which can be easily and continuously generated rather than measuring execution against a plan

*Governance and Management - Software as an Asset***.** Develop an enterprise-level Strategic Technology Plan that reinforces the concept of software as a national security capability. Include an approach for enterprise-level DevSecOps and other centralized infrastructure development and management, an approach for shared services, and applications management. The plan should recognize how disruptive technologies will be introduced into the environment on an ongoing basis. Ensure appropriate integration of a data strategy and the Department's Cloud Strategy. Examine a Steering Committee approach for management.

*Organization and Culture Reform***.** Examine roles and responsibilities with the intent to streamline reconcile, and resolve discrepancies for software acquisition and management among the DoD CIO, the USD(A&S) and the CMO. Re-focus the software acquisition workforce on teaming and collaboration, agility, improved role definition, career path advancement methods, continuing education and training opportunities, incentivization, and empowerment. Involve them in the change process.

Proposed Legislative/Regulatory Language

Any topic with an "*" was an idea derived either wholly or in part from engagements with the FY18 NDAA Section 873 and 874 agile pilot programs.

STATUTORY

TOPIC	OVERVIEW / ISSUE	STATUTE	PROPOSAL
Acquisition Strategy	Acquisition Strategies mandated by Section 821 of the FY16 NDAA for MDAPs, MAIS, and Major Systems, mandates content for acquisition strategies and authorities, the content in terms of how the provision is mandated does not allow for much flexibility and agility in content.	Section 821, FY16 NDAA	1. Eliminate for all except MDAPs 2.Keep overall definition of content with the A&S (listed as AT&L) in (b)(1) for consistency across the Services 3. Section (b)(2) authority should reside with the Service Chiefs
MDAPs	Specific to the establishment of cost, fielding, and performance goals for MDAPs under section 2448a of title 10 introduced by Section 807 of the FY17 NDAA. Does not distinguish software intensive programs from any other type of program. Also this provision was a reaction to programs not following guidance for affordability already established in the DODI 5000.02.	Title 10 § 2448a through Section 807 of the FY17 NDAA	Eliminate this provision from statute. There is policy which already exists that covers this in the DoDI 5000.02 (note: DSD just signed out a memo on this)
Nunn McCurdy	Nunn McCurdy is not an effective tool for restructuring MDAPs. There is little evidence to show that programs emerging from Nunn McCurdy breaches are drastically restructured for improvement, and rarely are programs cancelled. Some go through more than one Nunn McCurdy (though, a rare occurrence). Perception is a reporting / paperwork bureaucratic exercise that does not positively impact behavior.	10 U.S.C. § 2433	1.Consider elimination of Nunn McCurdy 2. Consider replacement of Nunn McCurdy w/ different focus and outcomes mandated (note: requires additional discussion)

Statutory Definition – Major System	The purpose and intent of this term is confusing. The term is separate and distinct from MDAP and MAIS. Typically ACAT II programs are affiliated with the "major system" designator, but ACAT II is a policy designation not a statutory designation. These systems do not do mandated statutory reporting like MDAPs. Per 10 U.S.C § 2302: "The term "major system" means a combination of elements that will function together to produce the capabilities required to fulfill a mission need. The elements may include hardware, equipment, software or any combination thereof, but excludes construction or other improvements to real property. A system shall be considered a major system if (A) the conditions of section 2302d of this title are satisfied, or (B) the system is designated a "major system" by the head of the agency responsible for the system." Dollar thresholds are defined in 10 U.S.C. 2302d; 41 U.S.C § 109 (the title 10 and title 41 thresholds are different)	10 U.S.C § 2302 10 U.S.C § 2302d 41 U.S.C § 109	Eliminate definition from title 10. Other agencies may use the definition in Title 41; recommend keeping Title 41.

Live fire / survivability / lethality testing*	There is no exemption for software-intensive programs to conduct survivability / lethality / live fire testing to move beyond LRIP OR to modify these requirements to reflect their nature as software intensive programs. Any covered system may require LFT&E. Includes major systems in the definition which may or may not be software programs (per the § 2302 definition). Otherwise, a waiver must be sent to the congressional committees before MS B. *Note: awaiting feedback / add'l info from AIAMD PMO*	Title 10 U.S.C. § 2366; and DoDI 5000.02	First, elimination of the Major Systems from Title 10 U.S.C. § 2302 helps to solve the identified challenges. Further, consider language for Title 10 2366a which allows exemption for software intensive programs, where DOT&E must justify adding the program for oversight with the MDA and must streamline the process. *Note: awaiting feedback / add'l info from AIAMD PMO*

Statutory DOT&E authority*	DOT&E has been able to essentially stop programs as they move through the development (acquisition) process. DOT&E testers are also not often SMEs in the systems they are conducting testing oversight on which negatively impacts testing. 1. Statutory authority assumes use of waterfall methodology; relies on infrequent, major test events instead of the continuous testing that agile uses 2. Also assumes a separate test team (and even organization) as opposed to testers being embedded in an agile team.	Title 10 U.S.C. § 2399	1. DOT&E oversight is only when requested by the SAE or USD(A&S), or Congressionally directed, unless MDAP. 2. DOT&E will utilize, to the greatest extent possible, test data collected through existing test methodologies present in the program and will not recommend or prescribe additional independent one-time test events. 3. One time IOT&Es or cybersecurity test events will not be recommended for software intensive systems unless in specific circumstances if warranted 4. Lead tester from either DOT&E or JITC (preferably both, if JITC is being used as test org) must be a subject matter expert in the subject being tested, similar to how qualified test pilots run test flights (health records, financial systems, etc.)
Clinger Cohen Act (CCA)*	1. CCA compliance process is outdated 2. Has become a time-consuming burden for programs that is layered on top of DoD's robust resources, requirements, and acquisition system. This renders many CCA requirements redundant with other laws, regulations, and policies. 3. Checklist-driven; provides limited strategic value; recognized as more of a hurdle than an enabler to capability delivery	40 U.S.C. § 1401(3)	Exempt the DoD from the Clinger Cohen Act, 40 U.S.C. 1401(3)

Business Systems Acquisition Reform *	DoD has three different governance entities: a business organization (CMO), an IT organization (CIO), and an acquisition organization (A&S) involved in providing oversight of business systems. Further, the annual certification requirement for DBS investments leads to unnecessary delays and is duplicative of the POM in the PPBE process	Title 10 U.S.C. § 2222	1. For the 4th estate combine all three authorities for DBS under the DoD CMO. After one year conduct assessment and make a determination if this should be applied to the Services as well. 2. Eliminate the separate funding certification process from 10 U.S.C. § 2222; or 3. If not eliminated, require the funding certification to be merged into the PPBE process
Configuration Steering Boards (CSB)	Must occur on at least an annual basis per the current statute (MDAPs). The Services tend to implement them for programs other than MDAPs based on 5000.02, and long-standing cultural factors.	FY 2009 NDAA, section 814; DoDI 5000.02	Other boards (or equivalent entities) established by the CAE or as delegated, the PEO or PM may fulfill the requirement of the CSB as long as the board (or equivalent entity) meets at least once a year and addresses the requirements in (c)(1).

Appropriations Accounts supporting IT Acquisition*	Agile acquisition is hindered by the appropriations environment. We must allow for more flexibility in appropriations account definitions for IT programs.	10 U.S.C. § 2214	Proposed language: "Funding for software solution acquisition does not adhere to the same standard development categories as other major programs. Funding approved by Congress for acquisition of a specific software solution may be used for research and development, production, or sustainment of that software solution. Provided that the software solution being acquired is the same software solution for which funding was appropriated, that funding may be accessed without respect to the appropriations account and without engaging in transfer of funds under the standard reprogramming process. If funding for one software solution is used for a different software solution, it must undergo a transfer of funds under the standard reprogramming process."
Expand FAR 39 to cover all IT purchasing regulations*	FAR 39 is too general. Further, for more streamlined acquisition of IT all rules governing it would be contained in one place. Purchasing speed is also too slow. Would allow for government-wide IT best practices and increase commodity / government-wide purchasing.	Title 49, Chapter 1, part 39	Expand the FAR 39 (Acquisition of IT) to allow for one area to drive technology purchases. Unless otherwise stated, no other FAR rules would apply

REGULATORY / POLICY

TOPIC	ISSUE	REG / POLICY	PROPOSAL
Earned Value Management (EVM) *	1. Earned Value Management (EVM) techniques are difficult (resource intensive) to implement; are neither designed nor well suited to effectively on measure an agile project; EVM cannot easily accommodate fluid requirements and shifting baselines. 2. EVM is lagging, not leading. 3. EVM does not measure product quality or user acceptance, which are hallmarks of the agile software development approach.	DFARS Subpart 234.201 DoDI 5000.02 Table 8 OMB Circular A-11 (not high priority)	Revise DFARS Subpart 234.201, DoDI 5000.02 Table 8, and OMB Circular A-11 to remove EVM requirement

FMR rules supporting IT acquisition*	IT/AIS that are not embedded in weapons systems and/or major end item procurements are budgeted according to the investment and expense criteria, these criteria do not enable agile acquisition or recognize the lifecycle nature of IT	FMR Volume 2A, Chapter 1, Section 010212(B)	Rewrite FMR Volume 2A, Chapter 1, Section 010212(B): 1. Acknowledge that, for the purpose of modifying or enhancing software, there is no technically meaningful distinction between RDT&E, Procurement, and O&M. 2. Eliminate the $250,000 barrier between expenses and investments (i.e., stop explicitly tying to a dollar threshold the determination of whether software is an expense or an investment. If the recommendations listed under "Appropriations Accounts Supporting IT Acquisition" are adopted, there should no longer be a need to make this determination for intra-program transfers.)

DoD Interoperability Policy*	Directs various things that should be reconsidered for IT/Software: 1. NR KPP required 2. DoD specific architecture products in the DoDAF format which are labor intensive and of questionable value 3. Interoperability Support Plans (ISPs) required, where DoD CIO can declare any ISP of "special interest" 4. Requires DT authority to provide assessments at MS C 5. Mandates JITC to do interoperability assessments for IT with "joint, multinational, and interagency interoperability requirements"	DoDI 8330.01	Direct revision of DoDI 8330.01 or potentially elimination of it
PfM Policy	Outdated (Sept 2008). Does not consider role of data and metrics, additional portfolios (like NC3) since 2008	DoDD 7045.20	Determine authority for policy; direct revision of DoDD 7045.20

Appendix B.2:

Appropriations Subgroup Report – released previously January 11, 2019

Appendix B.3: Contracting Subgroup Report

v0.2, 6 Feb 2019

The contacting challenges faced by the DoD today are almost entirely cultural. This premise is asserted by instances of excellence throughout the Department where effective contracting methods have been executed (DDS, DIU, Kessel Run).

That said, rather than attempting to battle each cultural challenge as they arise, it is easier to create a new modern acquisition platform from which to execute contracts that starts from a point of “how should it be done” as a product of “what should we be buying”.

The historical acquisition system was created to prevent fraud. The new priority is to establish technical superiority over our adversaries. While the prevention of fraud continues to be, and always will be, important, as a singular priority it serves to undermine the current identified need of speed and efficiency, which results in technical excellence for the Department.

Pain Points

Individual contracts are subject to review processes designed for large programs (of which they are likely enabling). This limits the agility of individual contract actions, even when modular contracting approaches are applied. In addition, the acquisition process is rigid and revolves around templates, boards, and checklists thus limiting the ability for innovation and streamlining execution.

Contracts focus on technical requirements instead of contractual process requirements. The contract should address overall scope (required capability), Period of Performance and price. The technical execution requirements should be separate and managed by the product owner or other technical lead.

Intellectual Property (IP) rights are often genetically incorporated without considering the layers of technology often applied to a solution. A single solution might include open source, proprietary software, and government custom code. The IP clauses should reflect all of the technology used.

Desired state

The desired state is an acquisition model that is liberated from the decades of policy and regulations that singularly focus on fraud prevention and provides for efficiency allowing the DoD to keep pace with the private sector and adversaries. This can be accomplished through a new authority Congress establishes a separate *new* authority for contracting for software development and IT modernization.

Obstacles

- Requires act of Congress ⇒ work with Armed Service Committees Staffers
- There is no infrastructure to support this ⇒ establish policy for guidance
- There are no Contracting Officers with specific certifications ⇒ Leverage current certifications

- Could cause confusion on implementation (what applies, what doesn't) ⇒ A&S issues guidance

Ideas for change

Congress establishes a separate *new* authority for contracting for software development and IT modernization

To address "Individual contracts being subject to review processes designed for large programs":

- Treat procurements as investments "what would you pay for a possible initial capability" (cultural).
- Manage programs at budget levels, allow programs to allocate funds at a project investment level (policy).
- Work with appropriators to establish working capital funds so that there is not pressure to spend funds quicker then you're ready (iterative contracts may produce more value with less money) (statute).
- Leverage incentives to make smaller purchases to take advantage of simplified acquisition procedures (cultural).
- Revise estimation models - source lines of code are irrelevant to future development efforts, estimations should be based on the team size, capability delivered, and investment focused (cultural).
- Allow for documentation and reporting substitutions to improve agility (agile reporting vs EVM) (cultural and EVM policy).
- Provide training to contracting officers, program managers, and leadership to understand the value and methods associated with agile and modular implementation (cultural).

To address "*Contracts focus on technical requirements instead of contractual process requirements*":

- Separate contract requirements (scope, PoP, and price) from technical requirements (backlog, roadmap, and stories) (cultural).
- Use statement of objectives (SOO) vs statement of work (SOW) to allow the vendor to solve the objectives how they are best suited (cultural).
- Use collaborative tools and libraries so that all content is available to all parties at all times (cultural).
- Use an agile process to manage structure and technical requirements (cultural).
- Establish a clear definition of done for the end of a sprint (code coverage, defect rate, user acceptance) (cultural).
- Use modular contracting to allow for regular investment decisions based on realized value (cultural).
- Streamline acquisition processes to allow for replacing poor performing contractors (cultural).
- Provide training to contracting officers, program managers, and leadership to understand the value and methods associated with agile and modular implementation (cultural).

To address "*Intellectual Property (IP) rights which are often genetically incorporated without considering the layers of technology often applied to a solution*":

- Establish clear and intuitive guidelines on how and when to apply existing clauses (cultural).
- Educate program managers and contracting officers on open source, proprietary, and government funded code (cultural).
- Have standard clause applications for each of the above that must be excepted vs accepted (cultural).

Proposed Legislative/Regulatory Language

(1) Authority

(a)Additional Forms of Transactions Authorized.—

The Secretary of Defense and the Secretary of each military department may enter into transactions (other than contracts, cooperative agreements, and grants) under the authority of this subsection for the purposes of acquiring Software Development and IT Modernization projects.

(1) The authority of this section—

(A)may be exercised for a transaction for a prototype project, and any follow-on production contract or transaction that is awarded pursuant to subsection (f), that is expected to cost the Department of Defense in excess of $100,000,000 but not in excess of $500,000,000 (including all options) only upon a written determination by the senior procurement executive for the agency as designated for the purpose of section 1702(c) of title 41, or, for the Defense Advanced Research Projects Agency or the Missile Defense Agency, the director of the agency that—

(i)

the requirements of subsection (d) will be met; and

(ii)

the use of the authority of this section is essential to promoting the success of the prototype project; and

(B)may be exercised for a transaction for a Software Development and IT Modernization project, and any follow-on production contract or transaction that is awarded pursuant to subsection (f), that is expected to cost the Department of Defense in excess of $500,000,000 (including all options) only if—

(i)the Under Secretary of Defense for Research and Engineering or the Under Secretary of Defense for Acquisition and Sustainment determines in writing that—

(I)

the requirements of subsection (d) will be met; and

(II)

the use of the authority of this section is essential to meet critical national security objectives; and

(ii)

the congressional defense committees are notified in writing at least 30 days before such authority is exercised.

(C) The authority of a senior procurement executive or director of the Defense Advanced Research Projects Agency or Missile Defense Agency under paragraph (2)(A), and the authority of the Under Secretaries of Defense under paragraph (2)(B), may not be delegated.

(D)Applicability of Procurement Ethics Requirements.—

An agreement entered into under the authority of this section shall be treated as a Federal agency procurement for the purposes of chapter 21 of title 41.

(2) Exercise of Authority by Secretary of Defense.—

In any exercise of the authority in subsection (a), the Secretary of Defense shall act through any element of the Department of Defense that the Secretary may designate.

(A)

Subsections (e)(1)(B) and (e)(2) of such section xxxx shall not apply to projects carried out under subsection (a).

(B)

To the maximum extent practicable, competitive procedures shall be used when entering into agreements to carry out the prototype projects under subsection (a).

(3) Appropriate Use of Authority.—

(1) The Secretary of Defense shall ensure that no official of an agency enters into a transaction (other than a contract, grant, or cooperative agreement) for a SW Development or IT Modernization project under the authority of this section unless one of the following conditions is met:

(A)

There is at least one nontraditional defense contractor or nonprofit research institution participating to a significant extent in the prototype project.

(B)

All significant participants in the transaction other than the Federal Government are small businesses (including small businesses participating in a program described under section 9 of the Small Business Act (15 U.S.C. 638)) or nontraditional defense contractors.

(C)

At least one third of the total cost of the prototype project is to be paid out of funds provided by sources other than other than [1] the Federal Government.

(D)

The senior procurement executive for the agency determines in writing that exceptional circumstances justify the use of a transaction that provides for innovative business arrangements or structures that would not be feasible or appropriate under a contract, or would provide an opportunity to expand the defense supply base in a manner that would not be practical or feasible under a contract.

(2)

(A)

Except as provided in subparagraph (B), the amounts counted for the purposes of this subsection as being provided, or to be provided, by a party to a transaction with respect to a SW Development or IT modernization project that is entered into under this section other than the Federal Government do not include costs that were incurred before the date on which the transaction becomes effective.

(B) Costs that were incurred for a SW Development or IT modernization project by a party after the beginning of negotiations resulting in a transaction (other than a contract, grant, or cooperative agreement) with respect to the project before the date on which the transaction becomes effective may be counted for purposes of this subsection as being provided, or to be provided, by the party to the transaction if and to the extent that the official responsible for entering into the transaction determines in writing that—

(i)

the party incurred the costs in anticipation of entering into the transaction; and

(ii)

it was appropriate for the party to incur the costs before the transaction became effective in order to ensure the successful implementation of the transaction.

(2) Payments

(a)Advance Payments.—

The authority provided under subsection (a) may be exercised without regard to section 3324 of title 31.

(b)Recovery of Funds.—

(1)

A cooperative agreement for performance of basic, applied, or advanced research authorized by a transaction authorized by subsection (a) may include a clause that requires a person or other entity to make payments to the Department of Defense or any other department or agency of the Federal Government as a condition for receiving support under the agreement or other transaction.

(2)

The amount of any payment received by the Federal Government pursuant to a requirement imposed under paragraph (1) may be credited, to the extent authorized by the Secretary of Defense, to the appropriate account established under subsection (f). Amounts so credited shall be merged with other funds in the account and shall be available for the same purposes and the same period for which other funds in such account are available.

(c)Support Accounts.—

There is hereby established on the books of the Treasury separate accounts for each of the military departments for support of Software Development and IT Modernization projects provided for in cooperative agreements containing a clause under subsection (d) and Software Development and IT Modernization projects provided for in transactions entered into under subsection (a). Funds in those accounts shall be available for the payment of such support.

(3)Education and Training.

The Secretary of Defense shall—

(1)

ensure that management, technical, and contracting personnel of the Department of Defense involved in the award or administration of transactions under this section or other innovative forms of contracting are afforded opportunities for adequate education and training; and

(2)

establish minimum levels and requirements for continuous and experiential learning for such personnel, including levels and requirements for acquisition certification programs.

(4)Regulations.—

The Secretary of Defense shall prescribe regulations to carry out this section.

(i)Protection of Certain Information From Disclosure.—

(1)

Disclosure of information described in paragraph (2) is not required, and may not be compelled, under section 552 of title 5 for five years after the date on which the information is received by the Department of Defense.

(2)

(A)

Paragraph (1) applies to information described in subparagraph (B) that is in the records of the Department of Defense if the information was submitted to the Department in a competitive or noncompetitive process having the potential for resulting in an award, to the party submitting the information, of a cooperative agreement for Software Development and IT Modernization projects authorized by transaction authorized by subsection (a).

(B) The information referred to in subparagraph (A) is the following:

(i)

A proposal, proposal abstract, and supporting documents.

(ii)

A business plan submitted on a confidential basis.

(iii)

Technical information submitted on a confidential basis.

(5)Records

Comptroller General Access to Information.—

(1)

Each agreement entered into by an official referred to in subsection (a) to carry out a project under that subsection that provides for payments in a total amount in excess of $5,000,000 shall include a clause that provides for the Comptroller General, in the discretion of the Comptroller General, to examine the records of any party to the agreement or any entity that participates in the performance of the agreement.

(2)

The requirement in paragraph (1) shall not apply with respect to a party or entity, or a subordinate element of a party or entity that has not entered into any other agreement that provides for audit access by a Government entity in the year prior to the date of the agreement.

(3)

(A)

The right provided to the Comptroller General in a clause of an agreement under paragraph (1) is limited as provided in subparagraph (B) in the case of a party to the agreement, an entity that participates in the performance of the agreement, or a subordinate element of that party or entity if the only agreements or other transactions that the party, entity, or subordinate element entered into with Government entities in the year prior to the date of that agreement are cooperative agreements or transactions that were entered into under this section or section xxxx of this title.

(B)

The only records of a party, other entity, or subordinate element referred to in subparagraph (A) that the Comptroller General may examine in the exercise of the right referred to in that subparagraph are records of the same type as the records that the Government has had the right to examine under the audit access clauses of the previous agreements or transactions referred to in such subparagraph that were entered into by that particular party, entity, or subordinate element.

(4)

The head of the contracting activity that is carrying out the agreement may waive the applicability of the requirement in paragraph (1) to the agreement if the head of the contracting activity determines that it would not be in the public interest to apply the requirement to the agreement. The waiver shall be effective with respect to the agreement only if the head of the contracting activity transmits a notification of the waiver to Congress and the Comptroller General before entering into the agreement. The notification shall include the rationale for the determination.

(5)

The Comptroller General may not examine records pursuant to a clause included in an agreement under paragraph (1) more than three years after the final payment is made by the United States under the agreement.

(6)Definitions.

In this section:

(1)

The term “nontraditional defense contractor” has the meaning given the term under section 2302(9) of this title.

(2)

The term “small business” means a small business concern as defined under section 3 of the Small Business Act (15 U.S.C. 632).

(a) Follow-on Contracts or Transactions.—

(1)

A transaction entered into under this section for a SW Development or IT modernization project may provide for the award of a follow-on contract or transaction to the participants in the transaction. A transaction includes all individual SW Development or IT modernization project subprojects awarded under the transaction to a consortium of United States industry and academic institutions.

(2)A follow-on production contract or transaction provided for in a transaction under paragraph (1) may be awarded to the participants in the transaction without the use of competitive procedures, notwithstanding the requirements of section 2304 of this title, if—

(A)

competitive procedures were used for the selection of parties for participation in the transaction; and

(B)

the participants in the transaction successfully completed the prototype project provided for in the transaction.

(3)

A follow-on production contract or transaction may be awarded, pursuant to this subsection, when the Department determines that an individual prototype or prototype subproject as part of a consortium is successfully completed by the participants.

(4)

Award of a follow-on production contract or transaction pursuant to the terms under this subsection is not contingent upon the successful completion of all activities within a consortium as a condition for an award for follow-on production of a successfully completed prototype or prototype subproject within that consortium.

(5)

Contracts and transactions entered into pursuant to this subsection may be awarded using the authority in subsection (a), under the authority of chapter 137 of this title, or under such procedures, terms, and conditions as the Secretary of Defense may establish by regulation.

(b)Authority To Provide Prototypes and Follow-on Production Items as Government-furnished Equipment.—

An agreement entered into pursuant to the authority of subsection (a) or a follow-on contract or transaction entered into pursuant to the authority of subsection (f) may provide for follow-on items to be provided to another contractor as Government-furnished equipment.

Proposed Policy for Implementation:

SOFTWARE DEVELOPMENT USING AGILE BEST PRACTICES.

(a) In General.—This policy governs software development activities within the Department of Defense or military departments to be developed using agile acquisition methods as provided for under NDAA 2020 Section XXX.

(b) Streamlined Processes.—Software development activities identified under subsection (a) shall be developed without incorporation of the following contract or transaction requirements:

(1) Earned value management (EVM) or EVM-like reporting.

(2) Development of integrated master schedule.

(3) Development of integrated master plan.

(4) Development of technical requirement document.

(5) Development of systems requirement documents.

(6) Use of information technology infrastructure library agreements.

(7) Use of software development life cycle (methodology).

(c) Roles And Responsibilities.—

(1) IN GENERAL.—Selected activities shall include the following roles and responsibilities:

(A) A program manager that is authorized to make all programmatic decisions within the overarching activity objectives, including resources, funding, personnel, and contract or transaction termination recommendations.

(B) A product owner that reports directly to the program manager and is responsible for the overall design of the product, prioritization of roadmap elements and interpretation of their acceptance criteria, and prioritization of the list of all features desired in the product.

(C) An engineering lead that reports directly to the program manager and is responsible for the implementation and operation of the software.

(D) A design lead that reports directly to the program manager and is responsible for identifying, communicating, and visualizing user needs through a human-centered design process.

(2) QUALIFICATIONS.—Shall establish qualifications for personnel filling the positions described in paragraph (1) prior to their selection. The qualifications may not include a positive education requirement and must be based on technical expertise or experience in delivery of software products, including agile concepts.

(3) COORDINATION PLAN FOR TESTING AND CERTIFICATION ORGANIZATIONS.—The program manager shall ensure the availability of resources for test and certification organizations support of iterative development processes.

(d) Plan.— DPAP shall develop a plan which shall include the following elements:

(1) Definition of a product vision, identifying a succinct, clearly defined need the software will address.

(2) Definition of a product road map, outlining a noncontractual plan that identifies short-term and long-term product goals and specific technology solutions to help meet those goals and adjusts to mission and user needs at the product owner's discretion.

(3) The use of a broad agency announcement, other transaction authority, or other rapid merit-based solicitation procedure.

(4) Identification of, and continuous engagement with, end users.

(5) Frequent and iterative end user validation of features and usability consistent with the principles outlined in the Digital Services Playbook of the U.S. Digital Service.

(6) Use of commercial best practices for advanced computing systems, including, where applicable—

(A) Automated testing, integration, and deployment;

(B) compliance with applicable commercial accessibility standards;

(C) capability to support modern versions of multiple, common web browsers;

(D) capability to be viewable across commonly used end user devices, including mobile devices; and

(E) built-in application monitoring.

(e) Program Schedule.—Shall ensure that each activity includes—

(1) award processes that take no longer than three months after a requirement is identified;

(2) planned frequent and iterative end user validation of implemented features and their usability;

(3) delivery of a functional prototype or minimally viable product in three months or less from award; and

(4) follow-on delivery of iterative development cycles no longer than four weeks apart, including security testing and configuration management as applicable.

(f) Oversight Metrics.—Shall ensure that the selected activities—

(1) use a modern tracking tool to execute requirements backlog tracking; and

(2) use agile development metrics that, at a minimum, track—

(A) pace of work accomplishment;

(B) completeness of scope of testing activities (such as code coverage, fault tolerance, and boundary testing);

(C) product quality attributes (such as major and minor defects and measures of key performance attributes and quality attributes);

(D) delivery progress relative to the current product roadmap; and

(E) goals for each iteration.

(g) Restrictions.—

(1) USE OF FUNDS.—No funds made available for the selected activities may be expended on estimation or evaluation using source lines of code methodologies.

(2) CONTRACT TYPES.—The Secretary of Defense may not use lowest price technically acceptable contracting methods or cost plus contracts to carry out selected activities under this section, and shall encourage the use of existing streamlined and flexible contracting arrangements.

(h) Definitions.—In this section:

(1) AGILE ACQUISITION.—The term “agile acquisition” means acquisition using agile or iterative development.

(2) AGILE OR ITERATIVE DEVELOPMENT.—The term “agile or iterative development”, with respect to software—

(A) means acquisition pursuant to a method for delivering multiple, rapid, incremental capabilities to the user for operational use, evaluation, and feedback not exclusively linked to any single, proprietary method or process; and

(B) involves—

(i) the incremental development and fielding of capabilities, commonly called “spirals”, “spins”, or “sprints”, which can be measured in a few weeks or months; and

(ii) continuous participation and collaboration by users, testers, and requirements authorities.

Appendix B.4:

Data and Metrics Subgroup Report – released previously January 11, 2019

Appendix B.5: Infrastructure Working Group Report

v0.1, 11 February 2019

Despite several years of effort to "move DoD to the cloud," significant friction still exists for the DoD to easily leverage the required compute, storage, and bandwidth infrastructure that the commercial world so readily enjoys. The major obstacle is not at all technical, but is broadly one of accessibility: the ability to specify, contract for, pay for, connect to, secure, and continuously monitor sufficient modern computing infrastructure. Modern computing infrastructure refers primarily to cloud-based computing technologies and stacks. "Cloud-based" does not necessarily presuppose commercial cloud, but could also be on premises or hybrid cloud solutions. Similarly, "computing technologies and stacks" can run the full spectrum from infrastructure, to platform, to function, to software as a Service (IaaS, PaaS, FaaS, SaaS).

Pain Points and Obstacles

How much cloud do I need? Countless developers and IT professionals have wrestled with this question, and often the answer is to "dive in," move some apps, see what is needed, and then scale and tweak from there. The Department's culture hampers our ability to even take a "leap of faith" like this. We must be able to precisely size and cost our cloud requirements before ever starting to experiment or prototype. It should become more clear why this analysis paralysis exists as the below pain points are outlined and considered.

How do I buy cloud? Oh, just head on over to FedRAMP, pick an approved provider, sign up and you're on your way… FedRAMP? Is that a cloud? What about GovCloud, cloud.gov (not the same thing by the way), and MilCloud (is that version 1.0 or 2.0?)? What's the difference between AWS GovCloud and Azure Government? Can I just sign up with a credit card like a normal private citizen and start hosting my compute and data in the cloud? Sadly, the answer is a definitive and resounding NO! Even if you know which "government-approved" cloud you're moving to, it's just not easy to contract for it or buy it.

There is not space here to answer all these rhetorical questions. For a good description of the difficulty of buying cloud, please refer to the DoD Cloud Acquisition Guidebook at https://www.dau.mil/tools/t/DoD-Cloud-Acquisition-Guidebook. Here the Defense Acquisition University (DAU) outlines the multiple activities that need to be accomplished to contract for cloud services. Starting with the dreaded IT Business Case Analysis (BCA), moving on to applying the DoD Cloud Security Requirements Guide (SRG - more on this soon), to getting an Authority to Operate (ATO), ensuring DISA approves of your Boundary Cloud Access Point (BCAP) and your Cyber Security Service Provider (CCSSP), and lastly to applying the DFARS supplementary rule to your cloud contact. No friction here right?

How do I know my cloud is secure? Easy. FedRAMP pre-evaluates and approves Cloud Service Providers (CSSPs) for Information Impact Levels (IILs) 2, 4, 5, and 6 (don't ask about levels 1 and 3; apparently we over specified and they aren't necessary any longer). Whew, now things are making sense… Not so fast, the FedRAMP IILs are for US Government cloud use, but not DoD![1] We need FedRAMP+ for DoD use, and DISA doesn't evaluate Cloud Service Providers (CSPs), only Cloud Service Offerings (CSOs). Huh? Be sure to go through the DoD Cloud Computing SRG, ensure those extra security controls are in place for FedRAMP+, and you're on your way. Again,

[1] Don't ask… we know DoD is part of the US Government.

not so fast Program Manager (or small business owner)! How are you and your customers going to access the fancy new cloud you just finally got on contract?

How do I access my cloud? The cloud, sort of by definition, implies ease of access, right? The National Institute of Standards and Technology (NIST) definition in SP 800-145 defines cloud computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." Well, if you're a DoD user, you need to ensure you've got a BCAP in place between your application/service and your users. It's OK and accurate to immediately envision bottleneck and single point of failure here.[2] Mis-configuring and under-provisioning BCAPs is the norm rather than the exception, so even with all that compute and storage in the cloud that you somehow ran the contracting gauntlet to get, you're going to severely lack adequate bandwidth and likely suffer from significant latency. Friction++.

How do I pay for cloud? The best part of cloud computing is that I can only pay for what I use. A true consumption-based cost model. Just like a utility. Not so for Government and DoD though. The Anti-Deficiency Act doesn't allow us to pay for cloud computing like a utility. A common way around this is to pay a third party contractor to buy the cloud service for us. This results in a situation where we estimate the highest charges we could ever incur in a year, add a bit of padding to that (say 20-30%), pay the third party, and we've paid for our cloud. What happens if we don't use it all up by the end of the year? Nothing (i.e. no refunds). Money spent. The third party contractor makes (quite?) a bit of extra profit for "taking the risk off the government." So much for consumption-based payments.

Desired State

The ability to provision, pay for, consume, access, and monitor cloud computing (compute, storage, and bandwidth) the same way any commercial organization does. It is understood that there are unique DoD security requirements, but that should only affect cloud pricing (say 1.5 to 2 times commercial, worst case), and not any of the other procedures to easily access cloud computing technologies and resources.

Obstacles

Significant obstacles remain to easily leverage commercially equivalent compute, storage, and bandwidth infrastructure. Contracting, security procedures (not necessarily requirements), network access (i.e. a modern technological approach to BCAP), and billing all loom large. The most important of these is the DoD's inability to contract and pay for cloud computing on a consumption basis.

Ideas for change

Establish a DoD enterprise ability to procure, provision, pay for, and use cloud that is no different from the commercial entry points for cloud computing. The Joint Enterprise Defense Infrastructure (JEDI) Cloud initiative is a bold attempt at this solution and should be awarded. Cloud.gov (which is ironically hosted in GovCloud) is another promising program that is already very straightforward to provision and buy, but is limited to IIL 2 data and applications. The objective cloud procurement and billing contract must include the ability to truly pay for consumption of cloud services and not be

[2] There are better ways to do this, like zero trust networks. The commercial world has some really good examples and architectures that don't require this man-in-the-middle attack called a BCAP which actually breaks end-to-end encryption by design…

artificially limited by the Anti-Deficiency Act. Modern software demands the ability to consume and pay for cloud services just as we do any other utility.

In addition to this, the DoD should establish a common, enterprise ability to develop software solutions in the “easy-to-acquire-and-provision” cloud that is fully accredited by design of the process, tools, and pipeline. Said another way, the DoD should stop the security accreditation of individual applications, but *should instead invest in accrediting the ability to produce software*. The pipeline, automated tooling, procedures, and operational monitoring and auditing of software should be the focus and target of security accreditation, not each individual application and version of an operating system or application.

Another essential and necessary, though not sufficient, change that must occur is to adopt modern commercial approaches to software and system security in the cloud that does NOT involve BCAPs, Internet Access (choke) Points (IAPs), or CSSPs that cannot be performed entirely by trusted commercial entities. The DoD must adopt modern cloud security approaches such as zero trust networks[3], micro-segmentation, and eliminate the perimeter approach to network security and trust that is based on assigned IP address or network connection point. Perimeter-based security cannot scale to accommodate the bandwidth, traffic, and latency demands of modern cloud access, applications, and services. Furthermore, it is a failed architectural practice that has proven to be readily exploitable by adversaries and is especially vulnerable to insider threats.

[3] https://www.oreilly.com/library/view/zero-trust-networks/9781491962183/ch01.html

Proposed Legislative/Regulatory Language

Provide explicit policy and guidance that allows cloud computing resources to be acquired and paid for by consumption and on demand. This will require an amendment to or reconsideration of the Anti-Deficiency Act to consider compute, storage, and bandwidth as a utility.

The following are excerpts from the 809 Panel report that can help address the recommendations made here.

> Panel 809 - Volume 3 of 3 January 2019 Implementation Legislative Branch Revise appropriation law and budgeting rules to address the unique aspects of buying consumption-based solutions. Recommendation 49 provides the flexibility necessary for these changes. Executive Branch Create a new subcategory of services called consumption-based solutions in FAR Part 37, Service Contracting, and add a reference (pointer) in FAR Part 39, Acquisition of Information Technology.43 Agency-specific regulations, policies, and guidance regarding service contracting are not applicable to contracts for consumption-based solutions or hybrid contracts when the primary purpose is to procure consumption-based solutions. The following is the definition of consumption-based solutions: Any combination of hardware/equipment, software, and labor/services that together provide a seamless capability that is metered and billed based on actual usage and predetermined pricing per resource unit, and includes the ability to rapidly scale capacity up or down. Consumption-based solutions must be measurable/meterable on a frequent interval customary for the type of solution (e.g., hourly, daily, weekly). The contractor is required to notify the government when consumption reaches 75 percent and 90 percent of the contract funded amount. New services or features can be added to contracts for consumption-based solutions at the discretion of the contracting officer without conducting a new competition, provided the amount of these new services or features does not exceed 25 percent of the total contract value. Update the Product Service Code (PSC) data architecture to accommodate consumption-based solutions as a new data type. **Add a new contract type called fixed-price resource units to FAR Subpart 16.2. The fixed-price resource units contract type: Establishes a fixed price per unit of measure (e.g., one hour of computing resource as shown in Table 3-1 below). Sets a ceiling for the overall contract value against which consumption of individual resource line items will be charged. Is the preferred contract type for consumption-based solutions, and when used for those procurements should not require special approvals. Can be incrementally funded.**
>
> 43 The term consumption-based solutions was chosen in favor of consumption-based services because lessons learned from utility services contracting indicated that including the word “services” would cause confusion and result in attempts to improperly apply all Service Contracting (i.e., FAR Part 37) rules to the new purchasing category.
>
> Sets a maximum unit price for each resource unit and captures price reductions when commercial catalog prices are reduced. Is permitted for use under commercial item/service acquisition in FAR Part 12: Acquisition of Commercial Items.
>
> Develop IT solutions training and a corresponding certification/designation for DoD acquisition professionals based on the existing DITAP, which is part of the FAC-C Core-Plus specialization in digital services. Refresh training content and individual certifications at least annually. Include instruction on how to conduct cost/price analysis for consumption-based solutions. This training curriculum is for commercial IT solutions and does not apply to weapon systems acquisition.

Note: Draft regulatory text can be found in the Implementation Details subsection at the end of Section 3. SECTION 3: IT PROCUREMENT Due to the limited interaction between commercial and DoD information technology (IT) markets, the two now operate at substantially different paces of technological advancement. Because the commercial IT market has outpaced the DoD market for decades, DoD regularly acquires outdated and inferior technology, often at higher prices and slower rates. DoD's slower acquisition pace has a direct effect on warfighting capability in a defense era defined by technological edge. Warfighters, and their support commands, are often operating with less functionality and at higher operating costs. This market 1 GAO, Weapon System Acquisitions: Opportunities Exist to Improve the DOD's Portfolio Management, GAO-15-466, August 2015, Highlights, accessed November 26, 2018, https://www.gao.gov/assets/680/672205.pdf. Report of the Advisory Panel on Streamlining and Codifying Acquisition Regulations Volume 3 of 3 | January 2019 Page EX-4 | Volume 3 Executive Summary segregation is caused by the vastly different way in which DoD and the wider federal government acquire IT. Rather than operating in the private-sector market of readily available options, DoD often creates detailed, intricate and unique requirements for its IT systems and services. DoD must acknowledge its acquisition system suffers from processes and procedures that are obsolete, redundant, or unnecessary and work to move quickly enough to keep pace with private-sector innovation. The recommendations in Section 3 offer strategies for transforming DoD's IT acquisition from both the top down and bottom up. Strategic revisions to how DoD understands and acquires IT are integrated with smaller-scale changes that restore efficiency to routine processes that have become bogged down by layers of bureaucracy. None of the actions recommended in Section 3 alone will solve the challenges associated with IT market segregation; however, together they offer a series of changes that can better align DoD acquisition with private-sector practices. Allowing DoD to buy in a manner similar to private-sector companies will reduce barriers to sellers in the marketplace. Rec. 43: Revise acquisition regulations to enable more flexible and effective procurement of consumption-based solutions. Rec. 44: Exempt DoD from Clinger–Cohen Act Provisions in Title 40. Rec. 45: Create a pilot program for contracting directly with information technology consultants through an online talent marketplace.

Appendix B.6: Sustainment / Modernization Subgroup Report

v0.2, 11 Feb 2019

Improving the materiel readiness of our fielded weapon systems and equipment is an imperative across the Department in accordance with the new National Defense Strategy.[4] The time is now to shift from our traditional, hardware-centric focus and identify what core[5] means for software intensive weapon systems and associated software engineering capabilities. Software is a foundational building material for the engineering of systems, enabling almost 100 percent of the integrated functionality of cyber-physical systems, especially mission- and safety-critical software-reliant systems. More simply, these systems cannot function without software.

For fielded weapon systems and military equipment, software life-cycle activities follow somewhat predictable cycles of corrective, perfective, adaptive, and preventative modifications while major modifications drive new periods of development. Software development activities, even those following agile methods, encounter a phase where the program transitions from adding new features to supporting and sustaining day-to-day use and operations. At that point, development changes and signals a move to "sustainers" within the organic industrial base. Therefore, sustainment may be defined as the sum of all actions and activities necessary to support a weapon system or military equipment after it has been fielded.

Prioritizing the transition to software sustainment during requirements and engineering development is critical to timely, effective, and affordable sustainment, regardless of how software engineering organizations are structured and resourced. Software sustainment organizations must be engaged and embedded at the earliest design stages to ensure we can keep pace with new capabilities as systems become operational. Lastly, access to software source code, emphasizing an early focus on designing for sustainment, and investment into establishing and modernizing system integration laboratories, are just a few of the challenges faced by the DoD software enterprise.

Pain points

Applying a hardware maintenance mindset to software hinders the DoD's ability to better leverage the organic software engineering infrastructure. DoD maintenance policies and maintenance-related Congressional statutes have traditionally been optimized for hardware and are difficult to change due to long standing policies, practices, inertia, and incentives. The goal of hardware maintenance is to repair and restore form, fit, and function. This mindset does not align well with the ever evolving nature of software. The scope of software engineering for sustainment mitigates defects and vulnerabilities, fact-of-life interface changes, and add new enhancements. Software is never done and any time it is "touched," it triggers the software engineering development life cycle which produces a new configuration. Therefore, any system that is dependent on software to remain operational, is always in a state of continuous engineering during sustainment (or O&S phase of the life cycle).

DoD's acquisition process is not emphasizing an upfront focus on design for software sustainment and a seamless transition to organic sustainment. It is critical that software be designed to be more affordably sustained with high assurance and the ability to integrate changes and enhancements

[4] "Summary of the 2018 National Defense Strategy" (Washington, DC: Department of Defense, 2018), https://dod.defense.gov/Portals/1/Documents/pubs/2018-National-Defense-Strategy-Summary.pdf.

[5] As defined in 10 USC 2464, *Core logistics capabilities.*

more rapidly to provide a continual operational capability to the warfighter. Moreover, software must be decoupled from hardware to the greatest extent possible in order to enable leveraging rapid and continuous hardware improvements. We need to place increased emphasis in acquisition on designing in software sustainability with a consistent emphasis on how DoD contracts for software as well as the span of requirements, architecture, design, development, and test. Additionally, this includes making provisions for timely access to the necessary range of software technical data to enable timely and effective organic software engineering and rapid re-hosting. It is essential that the DoD and industry work collaboratively to meet the increasing software sustainment demand.

Public Private Partnerships (PPPs) provide one means to leverage DoD and industry capabilities as a team to deliver warfighter capability. However, PPPs and other options are not being considered up front and leveraged across DoD as an inherent element of the acquisition and engineering strategy of programs. This team strategy may facilitate mutual access to the technical data inherent in executing the software development life cycle.

Limited visibility of the DoD organic software engineering infrastructure, capabilities, workload, and resources. Title 10 USC 2464 establishes a key imperative for DoD to establish core Government Owned Government Operated (GOGO) capabilities as a ready and controlled source of technical competence and resources for national security. DoD's focus has traditionally been on hardware and therefore there has seen significant Service and DoD enterprise focus on hardware GOGO capabilities and infrastructure for core. However, there has been significantly less upfront acquisition focus and visibility on what core means for software intensive systems and the associated GOGO software engineering capability. For the traditional DoD hardware-centric model, core capability is based on individual weapon systems or platforms at the depot level. All systems operate interdependently in a net-centric environment, where force structure and execution of mission capabilities are products of a system-of-systems capability. In a software intensive environment "Go to War" analysis of what core means as it relates to software requires more strategic thinking about core than just focusing on individual weapon systems or platforms (aircraft, ship, tank, etc.) as hardware. The hardware-centric focus on weapon systems likely underestimates the scope and magnitude of what should be considered a core requirement in a software intensive systems operational environment.

Desired State. Require government integrated software sustainment participation from the very beginning of development activities.

Ideas for Change

- Title 10 USC 2460 should be revised to replace the term software maintenance with the term software sustainment and a definition that is consistent with a continuous engineering approach across the lifecycle.
- DoD should establish a capability for visibility into the size and composition of DoD's software sustainment portfolio, demographics, and infrastructure to better inform enterprise investment and program decisions.
- A DoD working group should be established to leverage on-going individual Service efforts and create a DoD contracting and acquisition guide for software and software sustainment patterned after the approach that led to creation of the DoD Open Systems Architecture Contracting Guide.

- Acquisition Strategy, RFP/Evaluation Criteria, and Systems Engineering Plan should address software sustainability, re-hosting, and transition to sustainment as an acquisition priority. The engineering strategy and plan should engage software sustainment engineers upfront and co-locates government software sustainment engineers on the contractor software development teams to enable effectively and timely transition to an organic sustainment capability.
- The definition of "core capabilities" in 10 USC 2464 should be revisited in light of warfighter dependence on software intensive systems to determine the scope of DoD's core organic software engineering capability, and we should engage with Congress on the proposed revision to clarify the intent and extent of key terminology used in the current statute.
- The DoD should revise industrial base policy to include software and DoD's organic software engineering capabilities and infrastructure. Start enterprise planning and investment to establish and modernize organic System Integration Labs (SILs), software engineering environments, and technical infrastructure; invest in R&D to advance organic software engineering infrastructure capabilities.
- **Revisions to the Definition of Depot-Level Maintenance and Repair**

Section 2460 of title 10, United States Code, is amended—

(1) in subsection (a), by striking "maintenance classified by the Department of Defense as of July 1, 1995" and inserting "sustainment and software engineering (including requirements definition, architecture, design, development and coding, integration and test, and all other related software engineering-related activities) for fielded software to correct faults and vulnerabilities, make continuous capability upgrades, improve performance or other attributes, or adapt the product to a modified environment without regard to the type of system, funding source, means (organic software engineering, contractor, Public Private Partnership, etc.), and organizational location and alignment"

§2460. Definition of depot-level maintenance and repair

(a) IN GENERAL.—In this chapter, the term "depot-level maintenance and repair" means (except as provided in subsection (b)) material maintenance or repair requiring the overhaul, upgrading, or rebuilding of parts, assemblies, or subassemblies, and the testing and reclamation of equipment as necessary, regardless of the source of funds for the maintenance or repair or the location at which the maintenance or repair is performed. The term includes (1) all aspects of software ~~maintenance classified by the Department of Defense as of July 1, 1995,~~ sustainment and software engineering (including requirements definition, architecture, design, development and coding, integration and test, and all other related software engineering-related activities) for fielded software to correct faults and vulnerabilities, make continuous capability upgrades, improve performance or other attributes, or adapt the product to a modified environment without regard to the type of system, funding source, means (organic software engineering, contractor, Public Private Partnership, etc.), and organizational location and alignment, as depot-level maintenance and repair, and (2) interim contractor support or contractor logistics support (or any similar contractor support), to the extent that such support is for the performance of services described in the preceding sentence.

(b) EXCEPTIONS.—(1) The term does not include the procurement of major modifications or upgrades of weapon systems that are designed to improve program performance or the nuclear refueling or defueling of an aircraft carrier and any concurrent

complex overhaul. A major upgrade program covered by this exception could continue to be performed by private or public sector activities.

(2) The term also does not include the procurement of parts for safety modifications. However, the term does include the installation of parts for that purpose.

Appendix B.7: Requirements Subgroup Report

v0.7, 7 Feb 2019

The Department of Defense (DoD) in 2003 institutionalized the identification and validation of requirements via the Joint Capability Integration and Development System (JCIDS). Created to support the statutory responsibility of the Joint Requirements Oversight Council (JROC), it is one of three processes (Acquisition, Requirements, and Funding) that support the Defense Acquisition System (DAS). Considered revolutionary in its design, moving DoD from a threat-based to a capability-based model, it has begun to show its age in today's era of software-intensive systems intending to leverage agile software practices. These evolving agile practices upend traditional industrial-age process attempts to credibly and accurately predict a future 15-20 years away, necessitating unimaginable precision and foresight upfront in support to capability development. The requirement process, writ large, must adapt to support delivering capabilities at the speed of relevance; processes, cultures, and expectations of the Service and Joint Force requirement communities.

Pain points

A byproduct of top-level requirement flow down is rigidity and over specificity at the derived requirements level, that greatly hinders agile software design. Capability validated by the JROC does not proscribe requirement allocation to either hardware or software solutions. However, the resulting flowdown of derived requirements incorporated into the source selection/contract award and the subsequent allocation of these between hardware and software by the prime can ultimately discourage software design flexibility. The decisions, often made years before software coding even begins, locks the prime and the government into a proscribed path that often does not produce the desired warfighter capability within the needed time frame. Preserving software design flexibility must be a key component throughout the requirements validation process. "Requirers" will need to learn to settle for "less" not "more" at capability need inception.

Too often exquisite requirements, intended to be 100 percent correct, are levied on a system that in turn drives extensive complex software requirements and design, affecting development, integration, and system test. Today's requirements process more closely mimics the "big-bang" theory often vilified by industry, government, and Congress. As the warfighting community loses faith in the acquisition community's ability to meet their commitments through timely incremental improvements, the temptation to "gold-plate" a requirement becomes more prevalent. Likewise, as the acquisition community is forced to defend shifting warfighter priorities in budget deliberations and Congressional engagements, the temptation to "lock requirements down early" permeates acquisition strategies. With both of these choices in play, exquisite requirements must be described perfectly at capability inception in order to maintain a low-risk acquisition program - obviously an impossible outcome.

Data sets are siloed within programs - a common Law of Requirements is that programs of record (PoR) try to avoid dependencies with other PoRs. By tying SW to a PoR, it becomes nearly impossible to transfer that code across systems and data environments. Data "lakes," "pools," and "ponds" will be the foundation for future weapon system data repositories, and the requirements process must be flexible enough to accommodate this new archetype. Breaking

from the past mold of tying software code to a program of record and a specific data environment frees the program manager from the arduous task of integrating seams across multiple PORs.

Example. The Navy operates forward at sea and on-shore at maritime operations centers (MOCs). Command and control between sea and shore is a key aspect of how they fight – they need shared battlespace awareness at aligned actions across distributed units at best. However, the systems afloat and ashore are not always the same because ships need systems that are hardened for combat at sea. If a new algorithm can help manage supply and logistics on the cloud ashore, it may not run the same at sea because different system exists afloat. Extrapolating across Services, the USAF writes an algorithm to optimize F-16 maintenance, however it is highly unlikely that the Navy can pick it up and apply it to F-18s. This depends on the vertical integration of the algorithm, data, and system (PoR).

Desired State. Go from Sailor (Airman, Rifleman, etc.)-stated need to software delivery in their hands within days to support future conflicts. This necessitates a process for concept/requirements determination/setting that takes advantage of the agility in software development and software products to increase the agility and modifiability in our systems. Requirements flow down must also maintain a broad-based approach into the lowest levels of design. We also note that one of the overarching agile principles is that “increments are small.” Fast requirements, fast deployments and fast test cycles for usefulness are tough to accomplish with huge, monolithic software projects. Start small, stay small! Finally, recognizing that documenting and contracting for a moving target is not easy but must be done.

Obstacles. Breaking the tyranny of siloed PoRs will require a concerted effort across the Department, Combat Support Agencies, and will require Congressional engagement and support. Considerable cultural barriers must also be overcome as the algorithms themselves become capability, and the methods used to document, validate, and maintain currency enter the mainstream. Complexity and dependencies among multiple elements prevent widespread usage of Family-of-Systems (FoS) and System-of-Systems (SoS) requirement documents. Government requirements and acquisition communities take on extra oversight burden when they take a FoS or SoS approach because they have to manage all the pieces coming together effectively. Lastly, current statutory guidance does not promote, encourage, or reward the use of agile software development practices or environments.

Ideas for Change

- The Joint Staff should consider revising JCIDS guidance to separate functionality that needs high variability from the functionality that deemed “more stable” (e.g., types of signals to analyze vs. allowable space for the antenna). Then implement a “software box” approach for each, one in which the contours of the box are shaped by the functionality variability

- OSD should consider identifying automated software generation areas that can apply to specific domains

- The Joint Staff should consider revising JCIDS guidance to document stable concepts, not speculative ideas.

  - Specifying needed capabilities is important up front, however it must be acknowledged that initial software requirements need to be "just barely good enough" for the situation at hand or, in other words, "document late"

  - Acknowledge that software requirement documents will iterate, iterate, iterate. JCIDS must change from a "one-pass" mentality to a "first of many" model that is inherently agile delegating approval to the lowest possible level

- The DoD should consider instituting a distributed model-based approach to requirements development extended across the enterprise

  - The model should be used to develop result-based metrics for requirement evaluation

- The Joint Staff should consider revising JCIDS guidance to focus on user needs, bypassing the JCIDS process as needed to facilitate rapid software development. Guidance should specifically account for user communities (e.g. Tactical Action Officer (TAO), Maritime Operations Center (MOC) director) that do not have one specific PoR assigned to them, but use multiple systems and data from those systems to be effective

- OSD and the Joint Staff should consider creating "umbrella" software programs around "roles" (e.g. USAF Kessel Run)

Potential DRAFT Legislative/Regulatory Language

No recommendations at this time.

Appendix B.8: Security Accreditation/Certification Subgroup Report

v0.1, 28 Jan 2019

The Department's current Security Certification and Accreditation (C&A) process is a complicated and time-consuming process that is measured in months and years. The process is typically seen as a serial process that occurs after development with a checklist mentality. While this fits with a waterfall approach to development, the Department is changing to an agile, DevSecOps approach. The overall security paradigm must change from one where updates to software happen optimistically on a yearly basis to one where software is updated weekly or daily in response to emerging threats and this is recognized as more secure than the slow, static process. Additionally, we must strive to accredit the process, tools, and platforms to allow and enable continuous authority to operate (ATO) when software changes meet the required thresholds.

Pain points

Complex, time-consuming, and misapplied process. Although developing and operating software securely is a primary concern, the means to achieve and demonstrate security is overly complex and hampered by inconsistent and outdated/misapplied policy and implementation practices (e.g. overlaying historical DoD Information Assurance Certification and Accreditation Process (DIACAP) process over Risk Management Framework (RMF) controls for individual pieces of software versus system accreditation). The sense is that the Certification and Accreditation (C&A) process is primarily a "check-the-box" documentary process, adds little value to the overall security of the system, and is likely to overlook flaws in the design, implementation, and the environment in which the software operates.

No way to calculate total costs of C&A process. The Department needs to be able to calculate the true and component costs for implementing the RMF and C&A in order to identify inefficiencies, duplicative capabilities, and redundant or overlapping security products and services that are being acquired or developed. Absent a set of metrics it is difficult to prioritize risk areas, investments, and evaluating risk reduction and return on investment.

Lack of top-down security requirements. The Department has not decomposed security requirements from an enterprise level to a mission level to a functional implementation level. Programs waste resources implementing security controls that should be inherited.

Lack of automation. The C&A process is predominantly a manual process which makes it a very low process. Programs must plan in terms of months and years to get a product through the security accreditation process. This slow process does not provide the warfighter the timely, modern solutions that are needed.

Desired state

Accredit the process, not the product. Done correctly, security is applied from the beginning of software development using automated tools. Before transitioning into operations, an Authorizing Official (AO) reviews the process under which the software was developed and accepts the risk as determined from various scans and tests. The AO signs a Continuous Authority to Operate (ATO) so that as long as the process remains intact and is continuously operationally monitored, the subsequent software releases are accredited.

Obstacles

Two primary obstacles are culture change and workforce skills. The current security culture is that security is a checkbox activity at the end of the development process. As RMF is implemented, this is beginning to change the culture of security from compliance to continuous risk assessment. However, the process is still very manual. The culture change needs to include using automation to speed up risk assessment and continuous risk monitoring of operational software.

The other obstacle is the security and accreditation workforce skill set. While tools can provide reports and speed up security activities like scans and code analysis, it takes a particular skill set to understand those inputs and recommend or make a risk decisions. The current security workforce must be trained in these new skills.

Ideas for change

Embrace DevSecOps. The Department should embrace DevSecOps (not just DevOps) and provide the necessary resources to develop the common software components and automation to assemble, test, accredit, and operate software systems. DevSecOps also includes policy-supported processes, certified libraries, tools, and an operational platform (with appropriately instrumented run-time software), and a toolchain reference to implementation to produce “born secure” software.

Automate, Automate, Automate! The Department needs to provide automated tools and services needed to integrate continuous monitoring with the development lifecycle, enable continuous assessment and accreditation, and delegate decision making at the lowest level possible. Examples of automation are using static code analysis during the “build” stage, running automated unit tests, functional test, regression tests, integration tests, and resiliency/performance tests during the “test” stage, using dynamic code analysis, fuzzing scans, running container security scans, STIG compliance scans, and 508 compliance scans during the “secure” stage, and running continuous monitoring tools and ensuring logs are being pushed to the appropriate entity during the “monitoring” and “operational” stages.

Define top-down implementation requirements. The Department needs to ensure that each Joint Capability Area (JCA) flows-down its strategy, best practices, and implementation requirements/guidance for security and accreditation to allow the Component responsible for implementing the software to appropriately tailor RMF and plan the development, accreditation, and operation of the software. Furthermore, each JCA should endeavor to clearly state its risk profile and tolerance so that the RMF can be applied effectively and appropriately mitigate identified risks.

Education is necessary at all levels. As security is “baked in” to software during the development process, people must be educated about what that means as different tools look at different security aspects. They must also be educated in what it means to bring different security reports together and make a risk decision, both during development, and continuously during operations.

Culturally, people must learn to appreciate that speed helps increase security. Security is improved when changes and updates can be made quickly to an application. Using automation, software can be reviewed and updated quickly. The AO must also be able to review documentation and make a risk decision quickly and make that decision on the process and not the product and document it in a “Continuous Authority to Operate.”

Appendix B.9: Test and Evaluation Subgroup Report

v0.3, 11 Feb 2019

The fundamental purpose of DoD test and evaluation (T&E) is to provide knowledge that helps decision makers manage the risk involved in developing, producing, operating, and sustaining systems and capabilities. While colloquially referred to as a single construct, T&E is composed of two distinct functions: obtaining the data and assessing the data. This distinction is important because the T&E community will report "pain points" in both functions. There are also two major types of test: Developmental Test (DT) and Operational Test (OT). DT, by nature, is "experimental," performed on behalf of the Program Management Office (PMO), supporting a formative evaluation and identifying design elements that will drive mission critical capability to inform the evolution of component and system design. OT is "evaluative," performed by and on behalf of the warfighter, supporting a summative evaluation of system capabilities to support warfighting missions across the operational envelope.

Because T&E has historically occurred toward the end of, often, a long and costly acquisition process (e.g., requirements, design, development, etc.), it can be perceived as simply adding time and cost to an already late and over-budget effort; PMOs therefore can view this "last step" T&E as simply making the situation worse. And if T&E finds a system substantially defective, necessitating expensive re-engineering of the design late in developing, it adds to the perception that T&E simply adds cost and time to project execution. A continuous iterative T&E model is clearly called for, occurring alongside design and development, where T&E can both; catch defects early so they can be solved quickly and cheaply and inform/shape system requirements based on early feedback from the warfighter. Experience shows that active, early involvement by independent testers – combined with a PMO who responds to the independent testers' advice – makes a positive difference to program outcomes. We have seen this in modern iterative approaches, such as agile development, applied effectively in the DoD, especially in Major Automated Information Systems (MAIS).[6] Taken together, these observations point to the need to move away from what can be a linear waterfall process segregated by siloes, to a more iterative and collaborative model that fuses all development, test, processes, tools, and information to enable the continuous delivery of tested capability. T&E can then be viewed as saving time/cost in development, instead of adding time/cost.

Pain Points and Obstacles

The DoD lacks the enterprise digital infrastructure needed to test the broad spectrum of software types and across the span of T&E to support developmental efficiency (in DT) and operational effectiveness (in OT). Digital models of test articles (e.g., "Digital Twins") are not always available and not built to common standards. T&E environments, including threat surrogates or models, are often program-focused and funded, with short-term development goals and narrowly-scoped capabilities defined by the program. Building (and re-building) representative T&E environments is time and cost prohibitive for individual programs and results in duplicative infrastructure investments across DoD. Moreover, current T&E practices in the Services, including those focused on software-intensive systems, do not adequately test systems in Joint and Coalition environments, nor do they consistently use appropriate risk-based, mission-focused testing.

[6] FY16 DOT&E Annual Report.

The DoD lacks the enterprise data management and analytics capability needed to support the evaluation of test data in accordance with the pace of modern iterative software methods. As data required to make informed acquisition decisions continues to grow due to higher resolution measurements, higher acquisition rates, and other additional requirements for software intensive systems (e.g., interdependency, need to operate in system-of-systems, family-of-systems, Joint, and Coalition environments, etc.), the need for a T&E infrastructure to collect, aggregate, and analyze this data must likewise evolve to keep pace. More timely data fusion will require improvements in data management techniques, access speeds, data access policies, data verification techniques, and the availability of more intelligent and agile tools. Without this infrastructure, and within the current paradigm, we are failing to adequately gather and analyze these highly diverse and complex datasets, which leads to invalid assessments of acquisition program progress and system performance, undercuts mission readiness, and places warfighters at risk. This gap becomes an even more prominent choke point in an iterative cycle. Thus, even if we mitigate the first pain point with modernized realistic test environments, and had the capability to collect the appropriate mix/quantity of data in testing, we would still not have the analytics horsepower to turn around an assessment to support the pace of an Agile/DevSecOps iterative cycle.

The DoD lacks the resources needed to adequately emulate advanced cyber adversaries, to support fielding of trusted, survivable, and resilient software-intensive defense systems. Various oversight entities (e.g., NDAAs, GAO Reports, etc.) have acknowledged this gap, and past DOT&E Annual Reports have documented a significant number of adverse cyber findings in OT that should not require an operational environment to discover. While the gap exists now (in the absence of modern software methods), it will become an even more prominent choke point in a rapid development and operational fielding paradigm. We do not have the advanced cyber test resources (manpower, methods, and environment) to support a true Agile/DevSecOps approach to developing, testing, and fielding the broad range of software-intensive systems needed by DoD now and in the future, in an environment increasingly populated by advanced cyber adversaries.

The DoD lacks a modern software intellectual property (IP) strategy to support T&E in a rapid software development and fielding environment. Overcoming this pain point is critical to overcoming all of the three previously described pain points. Specifically, none of the previously described pain points is fully achievable without sufficient access to necessary technical data associated with the software deliverables. Software acquisition processes are and will continue to be suboptimal (with respect to time and risk) without access to relevant technical data and this gap will become an even more prominent choke point in an Agile/DevSecOps-based paradigm without that access. A modern software IP strategy must include access to software environments (e.g., source code, build tools, test scripts, cybersecurity artifacts/risk assessments, etc.) so tests are repeatable, extendable, and reusable. This strategy will also have to strike a balance with the IP rights of the innovator (usually industry) to ensure continued engagement of DoD with leading-edge technology organizations.

A modern software IP strategy would support the three previously described pain points via:

- Enhance our ability to operationalize the concept of “digital twins,” with sufficient access to the source code of a given system (balancing DoD and innovator IP rights), so as to be able adequately represent that system.
- Support the instrumentation of software-intensive systems as needed during testing.
- Support cyber vulnerability assessments and the assignment of risks to residual vulnerabilities, via access to system data (e.g., code, technical data, etc.).

Desired state

While the DOD does a fair amount of “integrated testing” now (across DT and OT), that is not the same as “integrating T&E with the Voice of the End User continuously and alongside software development.” T&E must strive for continuous software testing, automated and integrated into the development cycle to the fullest extent possible, across the entirety of the DoD’s software portfolio. The qualifier, “fullest extent possible” is important, as many experts have acknowledged that no single “one size fits all” approach will work best across the entire DoD software portfolio all of the time.[7,8] In this envisioned state, independent testers would work alongside developers and operators to help software development programs succeed and deliver capability at the speed of need. T&E would no longer be perceived as “slowing things down” or “costing money post-development” because it occurs toward the end of a highly linear and inefficient process, but would instead be associated with saving time and money during development. This vision, applied across the entire DoD software portfolio (i.e., beyond just IT or MAIS) requires the right kinds of tools, architectures and standards (see first three pain points), access to the right kind of data (see second and fourth pain points), and an ability to partner with and work alongside the developer, while yet maintaining independence and objectivity in our assessments.

Ideas for change

Build the enterprise-level digital infrastructure needed to streamline software development and testing across the full DoD software portfolio. Beyond the DevSecOps platform (or Digital Technology concept), the DoD requires a digital *engineering* infrastructure to streamline integration and testing. This suggests that the DevSecOps platform must be made available to all DoD software developers and:

- Integrated with (systems-level) model-based/digital engineering infrastructure, including digital twin(s),
- Integrated with existing T&E infrastructure (e.g., open-air ranges, labs, and other test facilities),
- Integrated with comprehensive tactical/mission-level infrastructure, and
- Available to others who could benefit (e.g., analysis, training, planning, etc.).

Even with this kind of complete testing infrastructure providing the capability to collect the appropriate mix/quantity of data in testing, we would still not have the analytics horsepower to turn around an assessment sufficiently rapidly to support the pace of an Agile/DevSecOps iterative cycle. We must develop the enterprise knowledge management and data analytics capability for rapid analysis/presentation of technical data to support deployment decisions at each iterative cycle.

Finally, to advance our cyber test resources such that we can achieve overmatch to our most capable adversaries while yet supporting the pace of the modern software development, the DoD should expand DOT&E’s current capability to obtain state-of-the-art cyber capabilities on a fee- for-service basis. This provides a straightforward way to acquire skilled cyber personnel from leading institutions (e.g., academia, university affiliated or federally funded research and development centers, etc.), to help the DoD to keep pace with advanced cyber adversaries.

[7] 2018 Defense Science Board Task Force on Design and Acquisition of Software for Defense Systems.

[8] Boehm and Turner, 2009. Balancing Agility and Discipline: A Guide for the Perplexed. Addison-Wesley. Boston, MA.

Appendix B.10: Workforce Subgroup Report

V0.3, 7 Feb 2019

DoD's workforce (civilian, military, and supporting contractor personnel) is our most valuable resource. The workforce's capacity to apply modern technology and software practices to meet the mission is the only way we can remain relevant in increasingly technical fighting domains, especially against our sophisticated peers, Russia and China.

Improved management of the Department's software acquisition talent will also drive success across the other subgroups and sections of this report. Policies, processes, and bureaucratic practices are never a sufficient substitute for competence.

The Department's challenges are well documented and well known by the software acquisition and engineering professionals who suffer most from the accrued technology, cultural, and leadership debt. The Workforce Subgroup identified prevalent pain points, but focused on providing concrete and actionable solutions for improving the recruitment, retention, development, and engagement of the workforce.

Pain Points

The Department's reputation as an employer is a weakness rather than a strength. Candidates base their employment decision on a variety of factors, but the organization's reputation and day-to-day work are chief among their considerations. The demand, and competition with the private sector, for an experienced and qualified workforce, is increasing as threats to our data security become more sophisticated. DoD has a reputation as an antiquated employer that rewards time in grade rather than competence and most often outsources its technical execution. Technical employees often serve as oversight or move away from "hands-on-keyboard" as they advance in their careers; no longer contributing to creative or innovative execution.

The Department does not adequately understand which competencies and skill sets are possessed and needed within its software acquisition and engineering workforce. Without the ability to distinguish the workforce, the DoD cannot effectively drive human capital initiatives. Furthermore, there is no enterprise-wide talent management system to manage the workforce (e.g., geographically, skills, etc.), which leads to bureaucratic silos and the inability to leverage the Total Force.

The Department has not prioritized a comprehensive recruiting strategy or campaign targeting civilians (90 percent of the acquisition workforce) for technical positions. When candidates do apply, they face an "overly complex and lengthy hiring process (that) frequently results in the Government losing potential employees to private sector organizations with more streamlined hiring processes," according to the President's Management Agenda.[9]

There is no comprehensive training or development program that prepares the software acquisition and technical workforce to adequately deploy modern development tools and methodologies within our dynamic environments. Hiring top technical talent into the Department will never be a silver

[9] "President's Management Agenda: Modernizing Government for the 21st Century," (Washington, DC: Office of Management and Budget, April 2018), 20, https://www.whitehouse.gov/omb/management/pma/.

bullet. The Department also needs to consider how to equip, reward, promote, and empower its existing workforce.

The Department is unable to leverage modern tools that are common in the private sector and our personal lives (e.g., cloud storage, collaborative software, etc.) due to bureaucratic barriers. Top talent expects access to these tools to meet mission demands, and their absence may discourage qualified candidates from applying or staying. Although the Department has pockets of innovation and entrepreneurship within rapid fielding offices across the services, this culture has not scaled to the larger acquisition programs and offices. Long-cycle times, bureaucratic silos, and information-hoarding prevail.

Desired State

The Department requires a workforce capable of acquiring, building, and delivering software and technology in real time, as threats and demands emerge. This workforce should resemble successful technology companies that must move quickly to meet market challenges. They do so by promoting an agile culture, celebrating innovation, learning from calculated failures, and valuing people over process.

The Department's workforce embraced commercial best practices for the rapid recruitment of talented professionals. Once on boarded quickly, they will use modern tools and continuously learn in state-of-the-art training environments, bringing in the best from industry and academia, while pursuing private-public exchange programs to broaden their skill sets.

Obstacles

The bureaucratic culture of the Department creates significant barriers compared to a commercial sector ecosystem that moves at the speed of relevance. These barriers are now ingrained within the institution, perpetuating a risk-averse environment that represents the most significant obstacle to reform. While there are minor legislative solutions to achieving the desired state, we believe that the Department has the necessary authorities and flexibilities, but has shown lack of impetus to move to the modern era of talent management.

While small pockets of expertise and progress exist, the Department as a whole lacks sufficient understanding of current software development practices and talent management models that support them. Studies on the workforce dating back 35 years that show "limited evidence these different efforts had any lasting impact or resulted in meaningful outcomes."[10]

Ideas for Change

Foundational. Taking into account history and the significant challenges with changing the culture in a bureaucracy, the Department should *empower a small cadre of Highly Qualified Experts and innovative Department employees to execute changes* from this report. This cadre is empowered with the authority to create, eliminate, and change policies within the Department for organizations beyond themselves. If needed, create a software acquisition workforce fund similar to the existing

[10] McLendon, Michael H.; Shull, Forrest; Miller, Christopher, "DoD's Software Sustainment Ecosystem: Needed Skill Sets," (Naval Postgraduate School, Monterey, California, April 30, 2018).

Defense Acquisition Workforce Development Fund (DAWDF). As called out by the Defense Science Board, the purpose of this fund will be to hire and train a cadre of modern software acquisition experts. This fund should also be used to provide Agile, Tech, and DevSecOps coaches in Program Offices to support transformations, adoption of modern software practice and sharing lessons across the enterprise.[11]

Workforce Foundations. The Department must develop a core occupational series based on current core competencies and skills for software acquisition and engineering. This occupational series should encompass all workforce roles required for modern software development and acquisition - engineers, designers, product managers, etc. Additionally, the Department should create a unique identifier or endorsement of qualified (experience & training) individuals who are capable of serving on an acquisition for software. This includes the development of a modern talent marketplace (and associated knowledge and skill tags/badges) to track these individuals. The competencies for this series should be flexible enough to evolve alongside technology, something that has constrained the 2110 IT Series.

Contractor Reforms. Defense contractors develop the majority of software in the Department. The Department should incentivize defense contractors that demonstrate modern software methodologies; this may take the form of software factory demonstrations and rapid software delivery challenges when evaluating proposals. Additional consideration should be given to contractors with demonstrated excellence creating commercially successful software.

Recruitment and Hiring. The Department must overhaul its recruiting and hiring process to use simple position titles and descriptions, educate hiring managers to leverage all hiring authorities, engage subject-matter experts as reviewers, and streamline the onboarding process to take weeks instead of months. The Department needs to embrace private-sector hiring methods to attract and onboard top talent from non-traditional backgrounds (e.g., hackers and entrepreneurs). Too often, these types of candidates are passed over or require special authorities to join the Department, due to lack of education or regular pay stubs. Furthermore, the Department must develop a strategic recruitment program that targets civilians, similar to its recruitment strategy for military members. This includes prioritizing experience and skills over cookie-cutter commercial certifications or educational credentials.

Development, Advancement, Engagement, and Retention. The Department must pilot development programs that provide comprehensive training for all software acquisition professionals, developers, and associated functions. Programs should be built in partnership with academia and industry, leveraging commercial training solutions rather than custom and expensive Federal solutions. This will include continuing education courses to help the workforce stay current and ensure technical literacy across the acquisition workforce. The Department must emphasize promoting and rewarding those that have proven both commitment and technical competence. Continually looking outside the Department is demoralizing and insulting to existing professionals that demonstrate innovation, excellence, and the ability to deliver already. The Department should incentivize and provide software practitioners access to modern engagement and collaboration platforms to connect, share their skills and knowledge, and develop solutions leveraging the full enterprise.

[11] Design and Acquisition of Software for Defense Systems," Defense Science Board, Feb. 2018, https://www.acq.osd.mil/dsb/reports.htm

Finally, the Department should encourage greater private-public sector fluidity within its workforce. Federal employees who come from the private sector bring with them best practices, modern methodologies, and exposure to new technologies. Federal employees who leave bring their understanding of our unique mission and constraints, helping the private sector develop offerings and services that meet our needs.

Proposed Legislative/Regulatory Language

1. **Establishment of a Core "Digital Delivery" Occupational Series.** *Modifying Existing Language* - Title 10, §1721. Need to add this Core Occupational Series to the list of "Designation of Acquisition Positions" or *Consider Using Existing Language:* Title 10, §1607 to add this occupational series fit within this established Defense Intelligence Senior Level model.

2. **Empower Implementation Cadre.** *New Legislation* - This will be critical to avoid a repeat of the past 35+ years of continuous admiration of the problem.

3. **Contractor Reform.** Adjust future NDAA's to add incentives for defense contractors to use modern development practices. (See FY18NDAA / §§873 & 874)

4. **Modernize Position Description and Hiring Practices.** *Modifying Existing Language* - Title 5, Part III, Subpart D, Chapter 53, the addition of this pilot program needs to be added.

5. **Develop a Modern Academy.** *Modification Language* - Title 10 §1746: This section should be added under the Defense Acquisition University, however, the HQE Cadre from Proposal #1 will lead the development of this pilot training program. Note: Tied with FY18 NDAA §891

6. **Private-Public Sector Fluidity.** *Modification Language* - Title 5, §§3371-3375: Expand the Inter-Government Personnel Act and allow more civil service employees to work with non-Federal Agencies and Educational Institutions. *Modification Language* - Title 10, §1599g: Expand the Public-Private Talent Exchange Program and modify the language to reduce the "repayment" period from 1:2 to 1:1 ratio.

7. **Computer Language Proficiency Pay.** *New Language* - Title 10, §1596a - Use this language to create a new Computer-language proficiency pay statute.

8. **Develop a Strategic Recruitment Strategy for Civilians.** New Legislation

9. **Pilot a Cyber Hiring Team.** *New Legislation* - Team will have all the necessary authorities to execute recommendations called out in this report. The team will serve as a Department-wide alternative to organization's traditional HR offices and will provide expedited hiring and a better candidate experience for top tier cyber positions.

10. **Establish Workforce Fund.** *New Legislation* - Similar to DAWDF, but the primary use will be for hiring and training a cadre of modern software acquisition experts.

Appendix C: Analysis the Old Fashioned Way: A Look at Past DoD Software Projects

v1.0, 6 Jan 2019

The Department has been building and buying software for decades. The study's initial idea was to take a cutting edge machine learning tool, hook it up to the Department's databases, and do an analysis across all of the plentiful software data collected over the years.

Unfortunately, initial attempts at analysis quickly led to the realization that the Department had never strategically collected data on its software. The data that have been collected cover only a subset of the systems the Department acquires and are typically collected by hand, with all the potential for erroneous or missing values that that implies. The granularity at which data are collected also does not typically support insight into specific questions of acquisition performance. Without massive data calls, enormous amounts of PDF scanning, and an impossible number of non-disclosure agreements, a comprehensive analysis would not be possible.

Instead, the SWAP members broke the analysis into two main efforts:

1. Analysis of the available data in order to test the board's hypotheses as they evolve. Subject Matter Experts who are familiar with the existing data and its constraints explored the available data in search of insights that would confirm or refute the board's hypotheses about DoD software acquisition performance. These results are described in this appendix.

2. Application of cutting edge machine learning and other modern analytical techniques to datasets from outside of the DoD, to support reasoning about the type of insights that could be gained and reported, if the Department had access to more comprehensive data about its software. These results are described in Appendix D.

C.1 Data Used in This Analysis

The focus of this study is on software-intensive programs – and the specific software scope within these programs – presenting top-level insights into software acquisition performance. We focused our analysis on a few major data sources collected by the Department, which can provide insight on these issues.

The data in our first source are known as Software Resources Data Reports (SRDRs). The SRDR data were selected for use because they are specifically focused on the software activities of DoD acquisition programs. The SRDR is a contract data deliverable that formalizes the reporting of software metrics data and is the primary source of data on software projects and their performance. The SRDR reports are provided at the project level or subsystem level, not at the DoD Acquisition Program level. The data points included in the analyses reported here are representative of software builds, increments, or releases. In many cases, there are multiple data points in the set that represent different subsystems or projects from the same program.

The SRDR applies to all major contracts and subcontracts, regardless of contract type, for contractors developing or producing software elements that meet specific criteria[12] and with a projected software effort greater than $20M.

SRDR reports are designed to record both the estimates and actual results of new software development efforts or upgrades, with the goal of supporting cost estimation. The reports collect many characteristics about software activities in both structured and unstructured formats. The primary data analyzed in our work were size, effort, and schedule. Notably absent from the SRDRs are any data about quality. Defect data have been optional until recently and hence were not reported.

Other data sources used to explore some of the assumptions and recommendations of the DIB are the IPMR (Integrated Program Management Report) and SAR (Selected Acquisition Report) datasets. Programs in these datasets fall into the category of Major Defense Acquisition Programs (MDAPs). These datasets include:

1. Software development effort measured in labor hours, software size, and development activity duration metrics delivered as mandated respective to contractual agreements.

2. Software development performance as identified within each contract report. However, each contract contained common elements supporting both software and non-software activity on contracts. These were treated in proportion to the weight of software activity cost on contract. These reports contain data for measuring contractor's cost compared to budget baselines on Department acquisition contracts as well as projections of cost at completion.

3. Planned and executed schedule milestone dates reported to the Department at the aggregate program level as required by acquisition policy. This information is included as a part of a comprehensive summary of total program cost, schedule, and unit cost breach information.

These software development effort metrics, contract performance, and program level schedule data represent the best source of product development, contract cost, and schedule performance information available on various projects throughout DoD. In addition, these datasets are also independently validated by agencies within the Department and subject to audits that require maximum fidelity to accounting standards.

It is worth noting that these datasets provide the best available information on DoD software acquisition, but are mainly limited to contract cost and budget performance (versus technical functionality performance) and were collected by hand. This scenario seems to address larger structural and cultural problems:

- The Department has no real acquisition data system that holds anything more than top-level data on our largest programs.

[12] Specifically, "within acquisition category (ACAT) I and IA programs and pre-MDAP and pre-MAIS programs, subsequent to milestone A approval."

- There is no automated collection of acquisition data, despite the fact that software tools and infrastructures, from which data can be automatically extracted, are integral parts of the state of the practice in the software industry.

- For much of the limited software-specific data that we do have (for example, source lines of code, or SLOC), this study has argued that they do not provide meaningful technical insight. Metrics like SLOC are not what the private sector would use to assess and manage programs.

- Leadership often relies on experience and trusted advisors because timely, authoritative data are not available for real analysis.

C.2 Software Development Project Analysis

One area of analysis focused on the SRDR data to describe, at an enterprise- or portfolio-level, what the Department is able to say about its software based on the software-specific data. As described above, SRDR data are more project- or subcomponent-focused versus program- or contract-focused; indeed, it is not easy and perhaps not possible to create a program-level understanding of software activities from the SRDR data.

The results reported here address 3 three questions:

1. How well do software projects perform in terms of effort and schedule?

2. Is there a difference in project performance related to the size of the project and the use of agile development?

3. How long do software projects take to reach completion?

The source of the data was the May 2018 compilation file published by members of the Software Resources Data Report Working Group. This file contains 3993 submissions that yielded 475 initial reports of planning estimates, 598 reports of final actual values, and 295 pairs of initial and final reports. Upon further investigation, 131 pairs contained full lifecycle information and therefore serve as a better dataset for studying effort and schedule growth. Thus, while we base our conclusions in this section on the best available data for software, it is important to keep in mind the data represent only a small subset of the Department's software.

The results presented below were primarily based on common statistical methods. Although a variety of additional explorations were conducted, the results were not found to be stable or to have achieved high confidence. These included dynamic simulation modeling, causal learning, and analysis with repetitive partitioning and regression trees.

Software Project Effort and Schedule Performance

In the current DoD acquisition lifecycle, substantial effort goes into defining requirements upfront in extensive detail, and projecting the cost and schedule for achieving the capabilities so described. Despite that, it is often said that the Department has problems acquiring the software capabilities it needs within budget and schedule. This analysis explored whether there was support for this conventional wisdom.

DoD projects in the dataset generally do indeed experience substantial effort growth. As seen in the following figure, the median number of estimated hours is 22,250 while the median number of actual hours is 30,120. (Note that the vast majority of points lie above the green line, indicating that actual values were greater than estimated.) The median rate of growth is 25%. However, there are some projects that expend less than their estimated effort, sometimes by a substantial amount as reflected by the points within the red circle. Unfortunately, based on the data reported we cannot discern whether they delivered the full committed functionality or not.

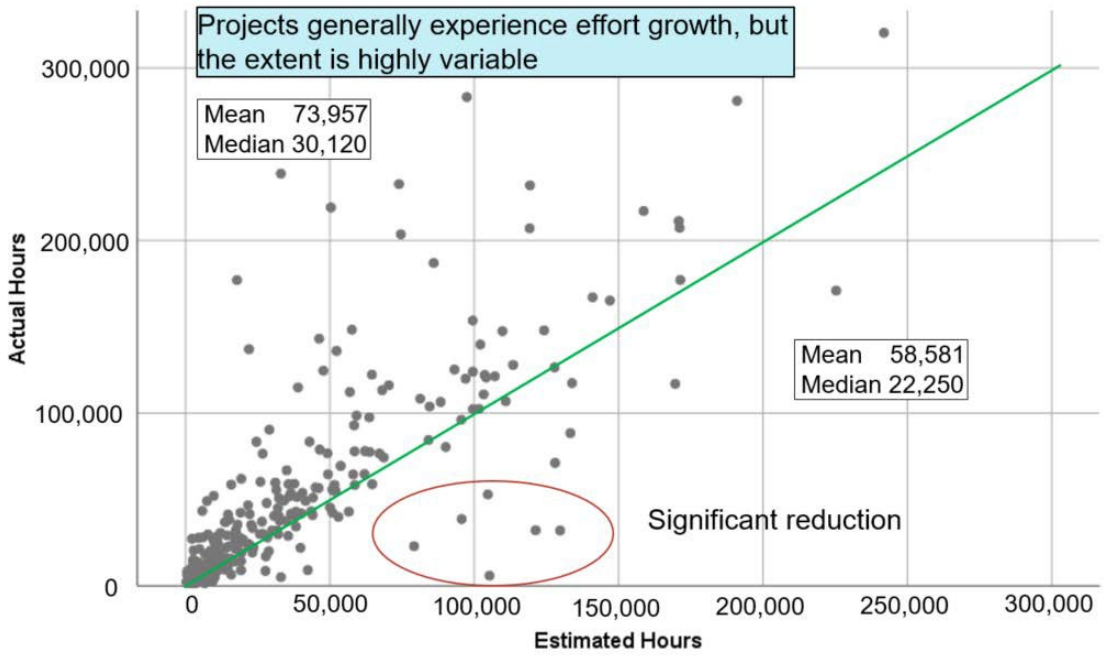


Figure 1. Estimated and actual project hours for project with less than 300,000 estimated hours.

The growth in project duration is generally not as large as the growth in effort. The median planned duration is 28 months and the actual duration is 34.9 months. The median growth in duration is 12%.

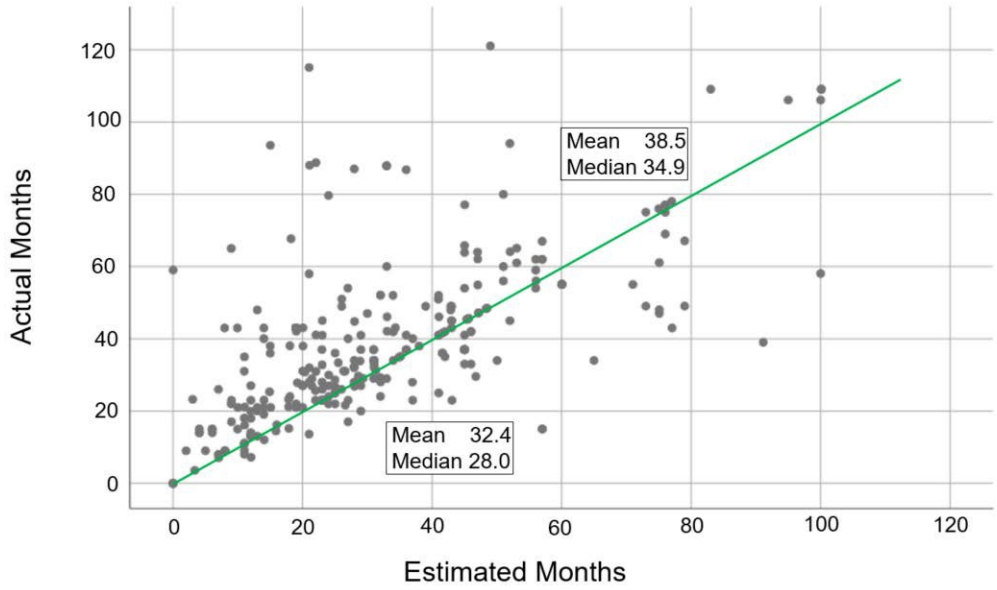


Figure 2. Estimated and actual project duration.

Interestingly, effort and duration growth are only weakly correlated and the highly skewed nature of their distributions means that averages create a more negative impression of performance than may be warranted. That is, the average exaggerates the degree of growth across the portfolio of projects. Nonetheless, in the data we have available, overruns of effort and duration are the norm.

Does Project Size Affect Performance?

The DIB has recommended that software programs should start small. The next analysis examined the historical data available to test whether small programs performed better than large ones, at least in terms of delivering capabilities on time and within budget.

To perform this analysis, projects were categorized in terms of their estimated equivalent source lines of code (ESLOC)[13] and effort. ESLOC is not collected but computed from the detailed SLOC measures that are collected: ESLOC combines the different sources of lines of code, new, modified, reused, and autogenerated, into a single count. Projects that were in the lower and upper quartiles on both effort and ESLOC measures were labelled as small and large projects respectively. This yielded 53 small and 55 large projects. An analysis of variance was conducted for growth in effort and duration.

The results found that small projects do not outperform large projects. Large projects do have less effort growth on a percentage basis but more growth in terms of raw hours. Surprisingly, schedule growth is very similar. Variation in performance overwhelms any apparent difference and the results do not achieve statistical significance.

[13] Elsewhere in this report, we reflect on the problems inherent with using SLOC as a measure. However, this is a key measure that has been collected historically by the department and so represents the best available data for this analysis.

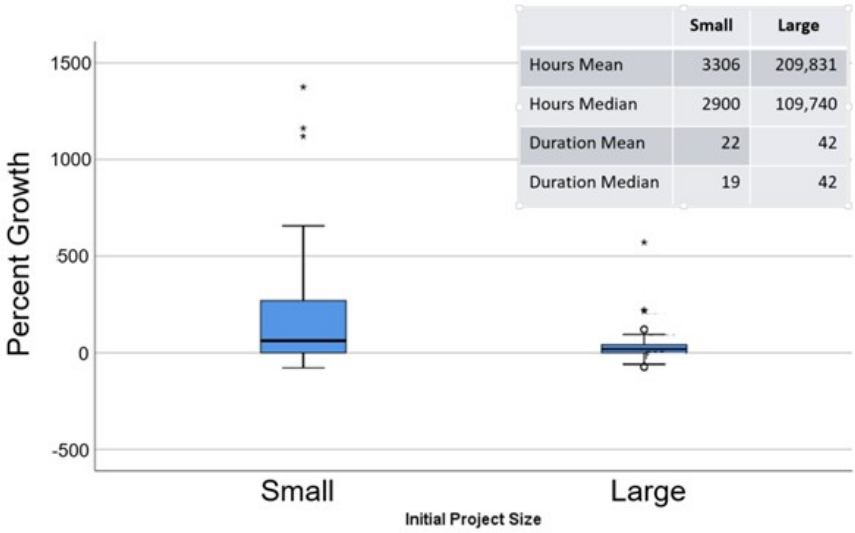


Figure 3. Effort growth by project size.

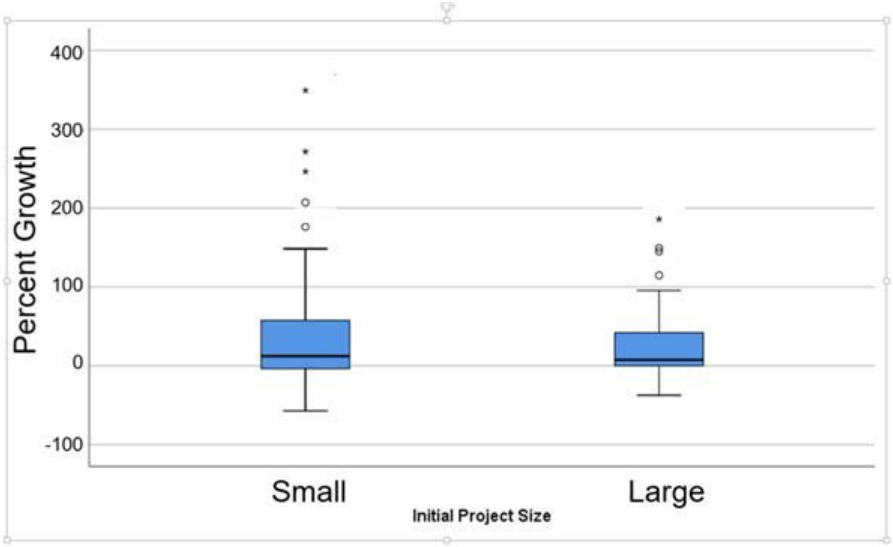


Figure 4. Duration growth by project size.

The fact that small projects still experience the same growth as large projects does not negate the advice that projects should start small, iterate often, and be terminated early if unsuccessful, since this can still result in significant savings in costs for projects that are not performing well.

Do Development Approaches Affect Performance?

There is much interest in the software development community and the DoD in the use of Agile methods. While the most recently updated SRDR form explicitly calls out measures for Agile projects, this has not been the case for the historical SRDR data upon which these analyses rely. Furthermore, the identification of the development approach is captured in an open text field. This necessitated interpretation and grouping of the entries in order to perform this analysis. A significant number of projects reported using “Waterfall,” “Incremental,” “Spiral,” or “Iterative” approaches. The remainder suggest use of a customized or hybrid approach. For the analysis here, “Waterfall” is compared to “Incremental,” “Spiral,” and “Iterative” projects.

Again, using ANOVA, the results indicate that effort growth does not significantly vary by development approach. However, duration growth is significantly less for projects using incremental development approaches as compared to waterfall (28% v 70% on average).

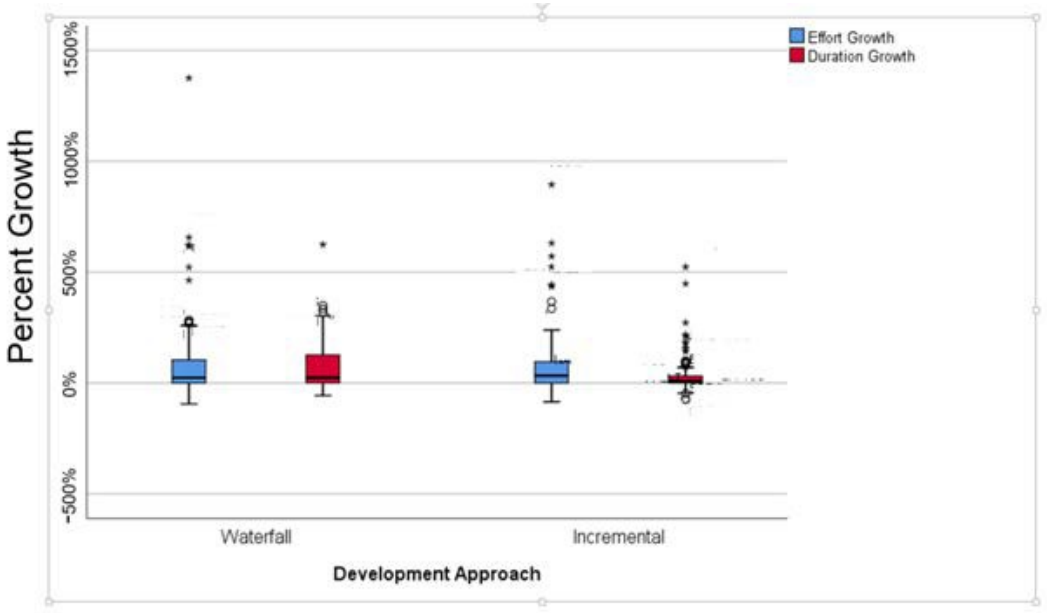


Figure 5. Effort and duration growth by development approach.

How Long Does It Currently Take to Complete a Project/Deliver Software?

As can be seen in the following figure, it is very rare for a project to complete in 12 months or less. Out of 371 projects used for this analysis, only 21 (6%) completed in this timeframe.

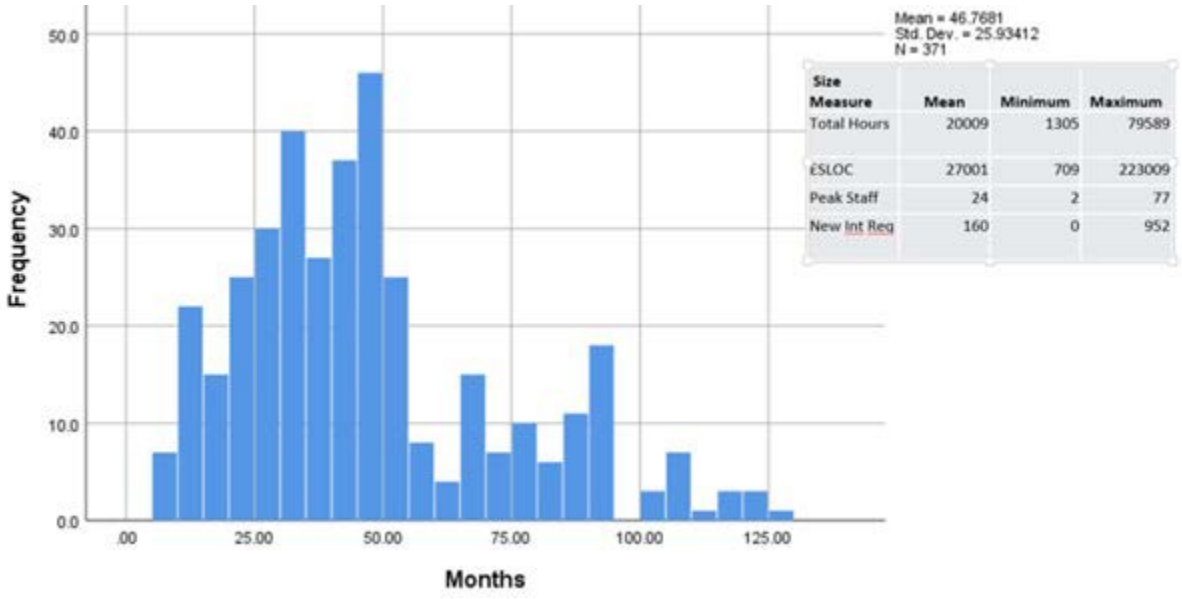


Figure 6. Actual duration for 371 AIS, Engineering, and Real-time projects.

Additional Insights from the SRDR Data

The preceding analyses were guided by the recommendations and proposed measures in DIB authored documents. In the course of performing those analyses, other questions and issues were posed and investigated. Briefly, these findings are:

1. Extreme variability in project performance confounds the identification of statistically significant results. This was noted above and is most likely actually due to performance and reporting inconsistencies.

2. Planned values can be useful for establishing expectations regarding reported actual effort and duration. That is, planned and actual values tend to be highly correlated with each other.

3. Planning for reuse is associated with significantly more schedule growth as compared to projects that do not plan for reuse.

The last one deserves more explanation as it is a somewhat counterintuitive result. Based on 275 projects that reported either no plan for code reuse or did plan for code reuse, the growth analysis showed no statistically significant differences in effort growth, but a significant difference in the amount of duration growth. Projects planning for code reuse had 52% duration growth as compared to only 20% for those that did not plan for code reuse. This phenomenon has been noted before and attributed to over-optimism about the amount and ease of code reuse. As the ability to reuse code falls short, unplanned effort and time go into producing new or modified code to compensate for the unrealized code reuse. Why effort growth is not significantly different is but likely at least partially related to the extreme variability in the performance measures.

Recommendations for Improving SRDR Data for Use

Issues regarding the data quality of SRDR data used here hampered the analyses. As is noted earlier, there is a substantial reduction from the number of submissions in the system to the number of usable records. At its most extreme there are 131 high quality pairs (262 records) out of the 3993 submissions included in the compilation dataset. That is, roughly 93% of the data is discarded.

The following recommendations are offered for improving SRDR data for use in addition to supporting the needs of the DOD cost community. Briefly, they are:

1. Leverage data collection and reporting from automation within the software environments (software factory). Minimize the need for manual entry and transformation.

2. Capture information about the quality of the delivered system.

3. Make the data more broadly available and encourage analyses into DoD software challenges (DIB Recommendation A6).

4. Identify the information needs of the stakeholders and intended users of the data beyond the cost community.

C.3 Software Development Data Analyses

A second investigation focused on cost and schedule performance data reported on recently completed and ongoing software development efforts within DoD. As these data provided insights *within* programs (and allowed understanding how values changed over time), we expected that this analysis would allow for deeper dives that could better explain how software acquisition occurs in programs.

This information was extracted from IPMRs, which are deliverables required by most contracts. The team also reviewed SARs for the large ACAT I programs to gain perspective on programs as they evolve over time.

Poor Data Quality and Inconsistent Data Reporting

There are approximately 130 ACAT I programs reporting research and development (R&D) contract performance over the past 10 years. We discarded from our analysis:

- Contracts for which the first IPMR report showed 65% (or about two-thirds) completed in work scope, reasoning that too much of the work had occurred before data collection began;
- Contracts for which the latest IPMR reported work that was less than 70% complete, reasoning that we would not have the ability to evaluate a significant portion of work completed.

146 contracts (35%) did not meet these data quality criteria out of the total of the 413 ACAT I program development contracts for which we have data (Figure 7). The fact that more than one-third of contracts do not meet this criterion implies that DoD would benefit from improving the quality and consistency of software development performance reporting. DoD cannot comprehensively assess the performance and value of the billions of dollars in investment without insight into a third of the complete portfolio.

Additionally, there are many data that are of limited utility due inconsistencies related to reporting. These have to do with problems with filing the mandated regular reports, and a lack of contextual data (i.e., metadata) being collected in a readily analyzable form. The DIB Software Metrics Recommendations contain recommended best practices on data collection and metrics definitions to not only capture data, but to establish standards meant to enhance software development performance.

Cost and Schedule Data

The resulting list of contracts was prioritized based on the budget assigned to the software-specific development efforts, and the top 46 contracts with the largest budgets were included in this study. These 46 contracts covered roughly half of the total dollar scope for all development programs in our dataset, and thus provided a reasonable sample size for our analysis. In addition, 35 contracts for smaller ACAT II and ACAT III software intensive Command and Control (C2) and Automated Information System (AIS) programs were included in this analysis. This resulted in the study capturing 81 total contracts valued at $17.9B in software development cost over the past

10 years (2008-2018). This study did not attempt to qualify or quantify the reasons for cost and schedule growth, recognizing that growth is not always indicative of poor performance by the program and/or contractor.

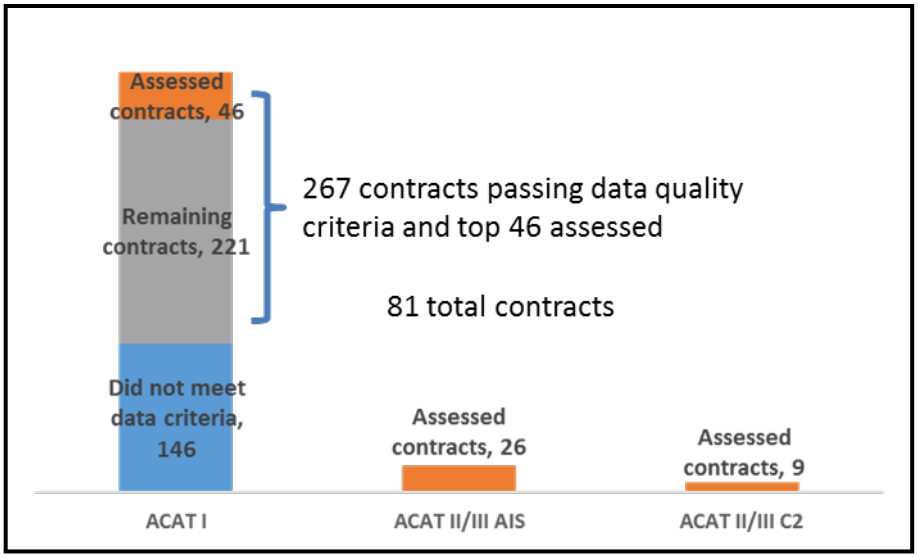


Figure 7. Results of Contract Selection Process

The 81 total contracts included in this analysis covered the portfolio of DoD programs, including software intensive C2 and AIS programs as well as aircraft, radars, land vehicles, and missile weapon systems, as shown in Figure 8.

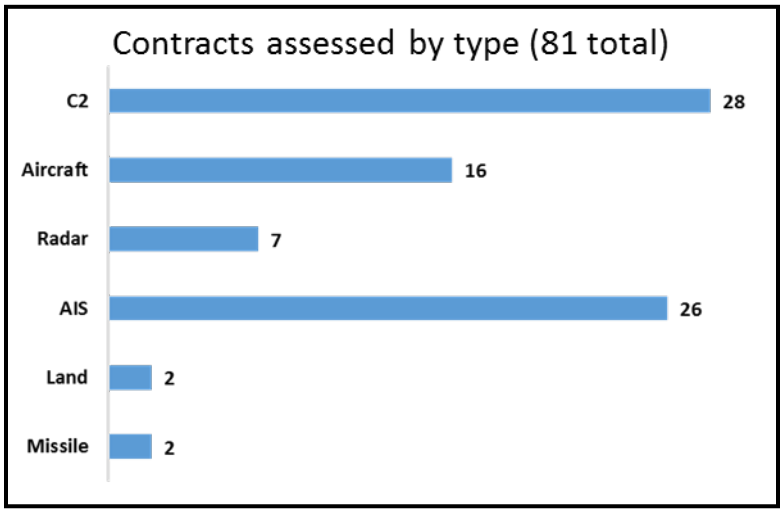


Figure 8. Contracts Analyzed by Weapon System Type

Large Software Cost Growth

The analysis of IPMR data found that on average, the contracts experienced 138% cost growth. The total combined value of the software development budgets within these contracts was $7.6B at the time of initial reporting. By the time these contracts reported the latest (or in some cases,

final) performance baseline, the software development budget total grew by $10.4B. Based on the analysis completed, significant software development cost growth was experienced across all platform and program types, resulting in a second observation: In general, the DoD struggles to minimize software development cost growth across the complete portfolio of projects. Figure 9 provides a summary of the 81 contracts evaluated, organized by project and by platform type. Note that the cost growth of “C2 Program A05” was truncated in the figure as it was an outlier in the analysis.

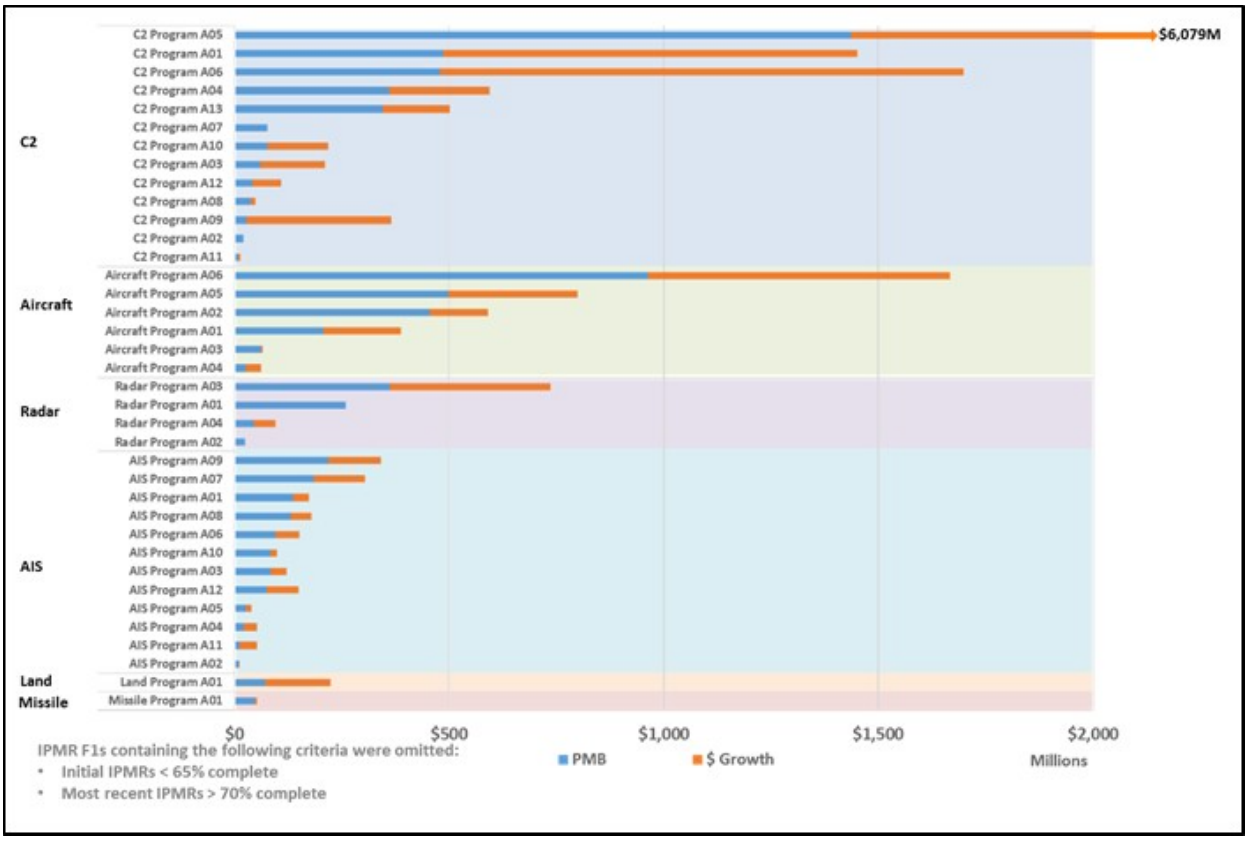


Figure 9. Contract Software Development Cost Growth by Program and by Platform

The study team used information provided by SARs and other relevant acquisition documentation to calculate project schedule growth. Figure 10 illustrates both dimensions of cost and schedule performance and identifies programs for which actual performance exceeds more than twice the baseline cost and schedule. Two programs, “AIS Program A01” and “C2 Program A02,” experienced cost or schedule growth so extreme that the bounds of the diagram axis plots were exceeded. This figure also supports the second observation that recent software development programs experience significant cost growth. The DIB SW Commandment 3 addresses cost growth by advocating that software budgets be planned upfront to support the full lifecycle versus the current funding lifecycle, defined around Planning, Programming, Budgeting, and Execution (PPB&E).

Figure 10. Software Development Cost Growth vs. Program Schedule Growth

Long Planned Durations and Frequent Re-baselining

The third study observation results from a deeper look into programs with high cost growth. This research found that in numerous instances, program baselines shifted (re-baselined) during the contract period of performance. The contracts with what appear to be significant “re-rebaselining” (i.e., multiple recurring increases to the expected cost) were analyzed in further detail.

SAR program milestones and available open source data were evaluated to provide a scale of time and functionality. It is observed that the software development effort crosses the same percent complete, as defined by the Earned Value Management (EVM) metric as the ratio of Budgeted Cost of Work Performed (BCWP) to Budget at Completion (BAC), multiple times. This represents an incremental method of adding cost, which is presumably associated with the addition of technical scope and requirements, which can result in a doubling or tripling of the total original budgeted value of the software development effort.

Figure 11 provides an example of this behavior, showing the “C2 Program A01” program effort that appears to re-baseline several times. The software development effort crosses the same percent complete point multiple times.

DIB Software Commandment 2 provides the recommendation that software development should begin small, be iterative and build on success; otherwise, be terminated quickly. DoD programs that take this approach are likely to see an improvement in performance once scope and requirements can be delimited through successful iteration. The behavior demonstrated in Figure 11 seems to indicate that to some extent, at least some programs are already behaving in an iterative way that better suits the technical work of software evolution. Unfortunately, our reporting mechanisms are not suited to reflect this reality, and in fact cannot differentiate a reasonable approach to incremental development from problematic cost or schedule growth. Looking just at the top-line numbers, these instances could be interpreted as excessive cost growth on the program, representing a problem from the Department’s point of view since the predictability of performance against cost and schedule baselines are normally taken as indicators of success.

What this scenario seems to point to is a need to improve our metrics collection to better reflect the underlying technical reality of software, where good performance often leads to a demand for new capabilities and new scope, as well as better educating our decision makers about how to interpret the results.

Thus this example provides more information about associated reporting issues tied to observation 5, that budgets should be contracted to support the full, iterative lifecycle of the software being procured with amounts definitized proportionally to the criticality and utility of the software.

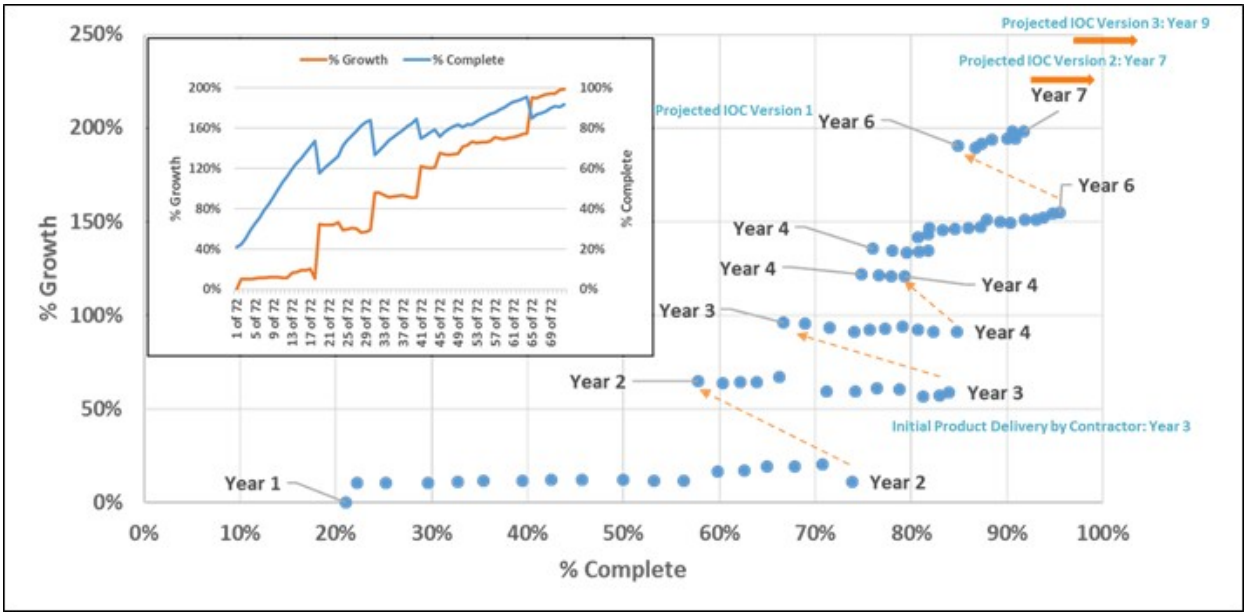


Figure 11. C2 Program A01 Performance Measurement Re-baselining

Agile Software Development Can Improve Program Performance

This study researched the performance of agile development methods that are implemented in existing programs. IPMRs do not explicitly state the type of development effort being used (incremental, agile, etc.). However, an article published in the journal Defense Acquisition provided an instance where agile development was applied and considered a success story. Although this article did not name the program, we were able to identify the most likely candidate, “Aircraft Program A05,” by matching the timeline presented in the article against the timeline of contracts that we could see in the program data.

The IPMR data for this program are shown in Figure 12. The contract work completed using an agile approach are shown in blue and represent a 21% cost reduction when compared to the initial budgeted value. This is in contrast to the contracts that seem to adopt a waterfall development methodology, i.e., contracts with planned long durations, which are shown in shades of orange and represent a 129% cost growth compared to the initial budgeted cost.

This analysis supports the fourth study observation that agile development may reduce cost growth compared to more traditional waterfall approaches. The DIB SW Commandment 2 also

advocates that agile approaches seen in commercial development result in faster deployment of functionality and cost savings which we observe in this instance.

Though a comparison of cost is one facet of performance, more research is required to increase the certainty that better overall performance and results were achieved with agile methods.

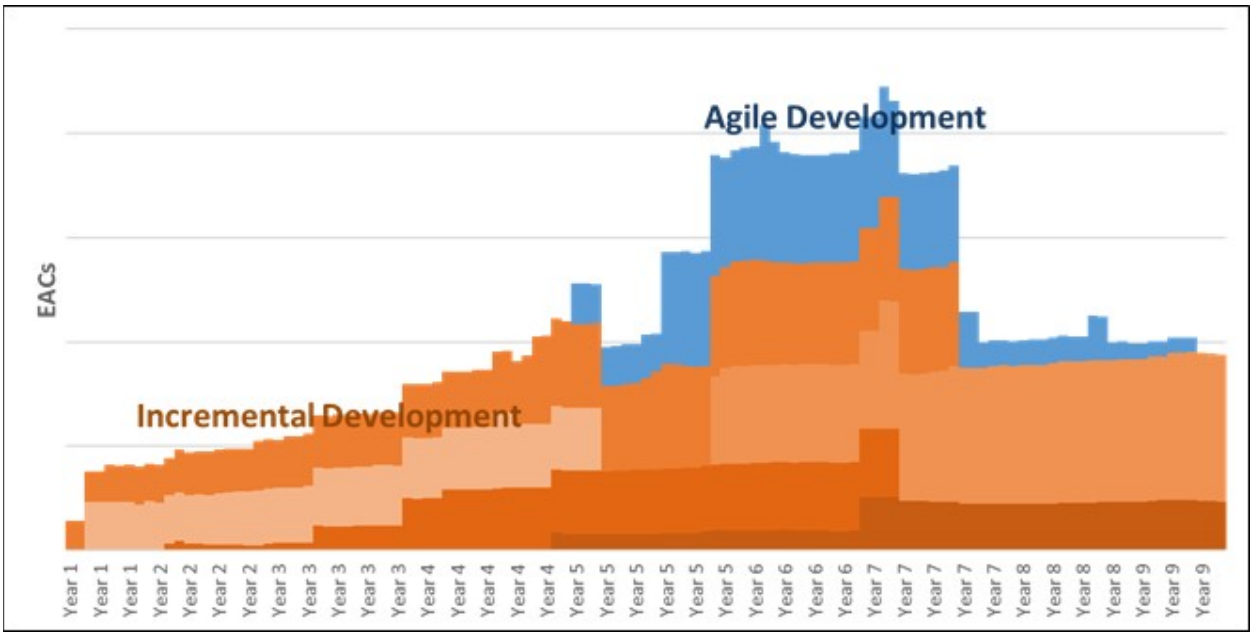


Figure 12. Aircraft Program A05: Incremental vs. Agile Development Efforts

Cost and Schedule Analysis Summary

In important ways, this analysis was typical of other efforts that aim to use Department data to examine the performance of acquisition. Due to the limited nature of the data available, our best analyses typically take months to create, with substantial time needed to find the data, to collect them, and to compile them into a structured format from multiple siloed and restricted systems.

The observations taken from data analysis of DoD program cost and schedule performance support the supposition that the current state of software acquisition is highly problematic and unsustainable relative to affordability and functionality. The DIB SW Commandments 2, 3, and 4 provide recommended measures to contain growth and increase the opportunity for cost savings by detaching software development from a hardware manufacturing industrial model and integrating software development and operations to quickly provide functionality to users and meet changing needs dictated by a dynamic global environment.

The preceding sections have described specific conclusions from the analyses our team conducted. Equally important, however, are the types of analyses we were *unable* to conduct given the data that were available.

A notable omission is that the Department is unable to address questions of *how much* software it has. Not in terms of software size but in terms of an index of how many important software systems have been acquired or are being sustained by the Department: There is no DoD or Service framework for describing the types of software intensive systems, or any inventory /

catalogue of the software in use. As a result, it is challenging to comprehend the scope and magnitude of the DoD software enterprise, and to design appropriate solutions for issues such as infrastructure or workforce that can meet the magnitude of the problem. Although done at a smaller scale, NASA's software inventory is an example of such an inventory model that is used to make strategic decisions for a federal agency.[14]

There is a large and growing body of work on software analytics, the automated or tool-assisted analysis of data about software systems (usually collected automatically) in order to make decisions. Conferences such as Mining Software Repositories[15] and Automated Software Engineering[16] annually showcase the best of the new research in these areas, and these methods are having a practical impact in commercial and government environments as well. A summary of software analytic applications lists several important questions that can be explored in this way: to name just a few, "using process data to predict overall project effort, using software process models to learn effective project changes, … using execution traces to learn normal interface usage patterns, … using bug databases to learn defect predictors that guide inspections teams to where code is most likely to fail."[17] Without access to its own software data, the DoD is missing the opportunity to exploit another area of research that could provide practical benefit for improving acquisition.

In a later section of this report (Appendix D), we provide the results of a small study that was undertaken to demonstrate potential practical impacts that could be achieved if software data access could be possible in the future.

[14] NASA Engineering Handbook (https://swehb.nasa.gov/display/7150/SWE-006+-+Agency+Software+Inventory#_tabs-6).

[15] https://2018.msrconf.org/

[16] http://ase-conferences.org/

[17] T. Menzies and T. Zimmermann, "Software Analytics: So What?," in *IEEE Software*, vol. 30, no. 4, pp. 31-37, July-Aug. 2013. DOI: 10.1109/MS.2013.86

Appendix D: Machine Learning Exploration

Linda Harrell, John Piorkoski, Phil Koshute, Erhan Guven, Marc Johnson (JHU/APL)
Vladimir Filkov, Farhana Sarkar, Guowei Yang, Anze Wang (UC Davis)
Steven Lee (Rotunda Solutions)
v0.2, 18 Feb 2019

D.1 Introduction

The Defense Innovation Board (DIB) Software Acquisition and Practices (SWAP) study chartered an exploratory study to explore the use of modern tools in data analytics and Machine Learning (ML) to provide insights into cost, time, and quality of Department of Defense (DoD) software projects. The data analytics and ML effort were performed by a team from academia (University of California Davis (UC-Davis)), a university affiliated research center (The Johns Hopkins University Applied Physics Laboratory (JHU/APL)) and industry (Rotunda Solutions). Since a suitable DoD data set was not available, the three teams leveraged existing data sets that were readily available to perform ML experiments and quickly get results.

ML models were created to predict the cost, time, and other aspects of software projects and gain a deeper understanding of the potential impact of project characteristics on overall project budget and effort. The models were trained with different data sets and were constructed to predict different performance metrics throughout the software development lifecycle.

The JHU/APL team developed ML models to predict software project duration and effort using the commercially available International Software Benchmarking Standards Group (ISBSG) Development and Enhancement (D&E) Repository of completed software projects. The UC-Davis team developed ML models to forecast software project duration, effort, and popularity using the publicly available GitHub repository of open-source projects. Finally, Rotunda Solutions created a defect density ML model to capture the code complexity and predict potential risk of code modules using a publicly available NASA dataset.

Additionally, the Rotunda Solutions team identified a number of opportunities for harnessing ML and Artificial Intelligence (AI) to improve the software acquisition process during different phases of the procurement cycle. This research effort is referred to as the Opportunities for Analytic Intervention. Rotunda Solutions also started development of a conceptual mock-up to explore some of these opportunities.

Overall, the three ML model development approaches demonstrated promising results aimed at improving predictions of software cost, time, and quality during different life-cycle phases.

- The JHU/APL team identified features (software metrics) that can support predictions of duration and effort at the project onset and shows that ML models have very good accuracy even with as few as 5 to 15 important features, most of which can be easily collected. It also shows how the prediction accuracy increases slightly by also including the effort expended in different life-cycle phases (e.g., planning, specification, design, build, test, and implementation). Since this analysis addresses the whole software lifecycle, the APL effort is referred to as the Software Life-Cycle Prediction Model.
- The UC-Davis team shows how monitoring of software development activities over time via automated tools that capture metrics (such as the number of lines of code, the number

of commits, and team size) can support accurate forecasts of duration, software effort (SWE), and software popularity. Additionally, the UC-Davis analysis showed that the ML models could obtain very good forecasting accuracy only 6 months after code development has started. Hence the UC-Davis ML model can serve as an early warning indicator. Since this analysis leveraged data obtained during software development activities to forecast future outcomes, it is referred to as the Software Development Forecasting Model.

- The Rotunda Solutions defect density model automatically processed code files and output code complexity metrics to aid efficient resource allocations and risk mitigation.

Interestingly, despite the differences in the approaches taken by JHU/APL and UC-Davis, the teams shared similar conclusions. For instance, both teams identified the team size and the project timing as being important features for the predictions.

Section D.2 of this document describes the methodology applied to the APL Software Life-Cycle Prediction Model and the UC-Davis Software Development Forecasting Model. Section D.3 summarizes the major findings of all three analyses. Section 4 offers implications of these study results for DoD programs.

D.2 Methodology

The approaches taken for the APL Software Life-Cycle Prediction Model and the UC-Davis Software Development Forecasting Model were complementary. Table 2.1 summarizes key aspects of the two approaches. These aspects include:

ML Techniques. Both studies leveraged readily available commercial or open- source ML techniques. This enabled the teams to meet the task's quick reaction turn-around timeline and also ensures that DoD government personnel and contractors can apply a similar approach when they develop their own prediction models for software projects. Although the teams developed several types of ML models, this report focuses on those with the best results: the APL Random Forest (RF) and the UC-Davis Neural Network (NN) models.

Data Sets. The APL team leveraged the 2018 International Software Benchmarking Standards Group (ISBSG) Development and Enhancement (D&E) Repository of completed software projects. This diverse database contains thousands of software projects that are described by a rich set of features that span the whole software lifecycle, but most of these projects have less than one year in duration or less than two years of effort. The UC-Davis team mined the GitHub collaborative project development and repository site, which contains historical trace data captured from millions of open-source software projects. The resulting database includes hundreds of thousands projects of various sizes. Its feature set is not as rich as in the ISBSG database, but it automatically tracks development metrics including commits, discussions, and other activities.

Target Variables. The APL team focuses on predicting software project duration and effort, two of the three metrics of greatest interest to the DIB. On the other hand, the UC-Davis team aims to predict the project duration (via its proxy months committed), the number of software commits (which is an incomplete proxy for software effort), and the number of stars (which is an indicator of the popularity of a project in GitHub).

Project Tiers and Boundaries. Large differences between proposal estimates and actual outcomes for software development duration and effort cause the biggest challenges for the DoD; small deviations are much more manageable. To reflect this perspective, both studies gathered their target variables into discrete tiers with boundaries shown in Figure 2.1.

Performance Metrics. Both studies assessed the performance of their models with confusion matrices (which shows the distribution of predictions in terms of predicted and actual tiers) and overall accuracy.

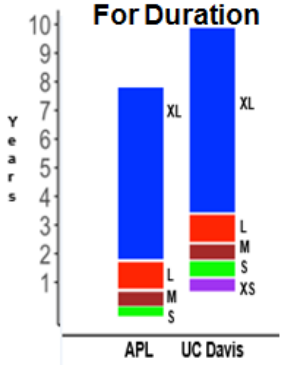
Table 2.1. Key Aspects of APL and UC-Davis Studies

Parameter		**APL Software Life-Cycle Prediction Model**	**UC-Davis Software Development Forecast Model**
Data Set		2018 ISBSG D&E Repository	2018 GitHub Repository
Number of Projects (after preprocessing)		2,818	Approx. 127,000
Number of Features (after reduction)		176	36
Target Variables for …	Duration	Project Duration	Months Committed
	Effort	Effort	Total Number of Commits
	Popularity	N/A	Number of Stars
ML Techniques		Off-the-shelf (NB, SVM, RF)	Off-the-shelf (MR, NB, RF, NN)
Results: Overall Accuracy; Confusion Matrices		Overall accuracy: Yes Confusion Matrix: 4 tier	Overall accuracy: Yes Confusion Matrix: 5 tier
Prediction Snapshots		Early concept development and procurement; Software development in process	After 6 months of software development ; Most recent software development
Feature Reduction		Yes	Yes

Definitions: NB = Naive Bayes, SVM = Support Vector Machines, MR = Multivariate Regression, NN = Neural Networks

APL Classification Tiers

Project Size	Software Effort
Small	2 months to 0.999 year
Medium	1 year to 1.999 years
Large	2 years to 7.999 years
Extra Large	>= 8 years



UC Davis Classification Tiers

Project Size	Number of Commits	Number of Stars
Extra Small	30 - 90	0 - 2
Small	91 - 140	3 - 7
Medium	141 - 217	8 - 22
Large	218 - 384	23 - 90
Extra Large	385 - 3469	91 - 31168

Figure 2.1. Classification Tier Boundaries

Prediction/Forecasting Snapshots. APL made predictions at two project phases (snapshots). The first snapshot is at onset, which includes features that are available or can be estimated during the concept, proposal, and procurement stage. The second is after software development has

been underway; it can include additional features as they become available. UC-Davis made predictions at three snapshots, corresponding to the time elapsed for each project: 6 months from first commit, 12 months from first commit, and most recent snapshot (1/1/2018). The most recent snapshot is taken to be the actual outcome (even if the project is still under development). For simplicity, the results with the 12-month snapshot are not discussed herein.

Feature Importance Ranking and Reduction. The APL RF and UC-Davis NN models both determined feature importance by evaluating the importance of each feature to the overall accuracy prediction and developed corresponding models with only the top ranked features.

Pre-Processing and Feature Selection. The pre-processing actions taken by the APL and UC-Davis are discussed in separate reports.

Project Context (Cluster) Creation. To fine-tune their predictive models, UC-Davis used an Autoencoder NN to group projects into four similarity clusters (i.e., contexts). A separate model NN was trained for each cluster. This technique allows for greater accuracy when project context is known early on, by, for example, tracking project metrics from the start.

D.2 Key Results and Findings

APL Software Life-Cycle Prediction Model

Table 3.1 shows the performance of the APL models that predict software project duration and effort with all features included. Even with minimal data cleaning, model tweaking, or sensitivity studies, and using a very sparse and unevenly distributed data set, the ML models predict a project's size tier with an overall accuracy ranging from 57% to 74%. These are impressive results for a quick-turnaround exploratory analysis.

As expected, the prediction estimates once development is underway are better than the predictions at program onset. This is because additional features, such as the effort expended in various life-cycle phases, help to improve predictions. However, with the features included in this analysis, the improvement was slight.

Even when the ML model does not correctly predict the size of the software project, the prediction is most often in adjacent tiers rather than significantly further away. This is evident in the confusion matrix in Table 3.2 and the additional confusion matrices provided in separate reports. This is important because it indicates that incorrect predictions still tend to be fairly close (e.g., an extra large project predicted as large or vice versa).

Table 3.1. Performance Summary for APL Prediction Models (with all features)

Model	Overall Accuracy
Predicting Duration at Project Onset	57%
Predicting Duration after the Project is Underway	58%
Predicting Effort at Project Onset	68%
Predicting Effort after the Project is Underway	74%

Table 3.2. APL Confusion Matrix for Predicting Effort as Project is Underway (with all features)

Accuracy values are shown as a percent of all projects of a given class		Predicted Class			
		S	**M**	**L**	**XL**
Actual Class	**Small (S)**	80	18	2	0.1
	Medium (M)	23	59	18	0.6
	Large (L)	2	20	73	5
	Extra Large (XL)	0.1	0.8	14	85

Table 3.3 identifies the most important features that influence the predictions. Naturally, the ranking of importance for each feature varies slightly for the predictions of duration and effort and for the two different phases (at project onset versus while the software development is underway), but the discrepancies are generally slight. Encouragingly, the features in this table are generally easy to obtain or estimate: function point standards, team size, software type, project implementation date, scope, programming language. The only feature category that is time consuming to gather is the functional size estimate. Each of the features in these tables is further described in the APL report.

Table 3.3 Most Important Features for ML Accuracy Predictions

Category of Feature	**Most Important Features**	**Project Phase**
Software Size	Functional Size, Relative Size, Adjusted Function Points	Project Onset
Standards for Function Point Estimates	Function Point Standards, Count Approach	Project Onset
Team	Maximum Team Size, Team Size	Project Onset
Type of Software	Industry Sector, Organization Type, Application Type, Business Area	Project Onset
Timing	Year of Project, Implementation Date	Project Onset
Scope	Project Activities, Development Type	Project Onset
Programming Language	Primary Programming Language, Language Type, Development Platform	Project Onset
Incremental Effort	Effort in the Planning Phase, Effort in Specify Phase, Effort in Design Phase, Effort in Build Phase, Effort for Implementation, Effort in Test Phase	When the Project is Underway
Cost	Total Project Cost	When the Project is Underway

Figure 3.1 depicts the accuracy prediction with small subsets of the most important features, and shows how the accuracy increases as additional features are added. This figure shows that although the database includes 176 features, very good predictions can be obtained using only as few as 5 to 15 features. These features are captured in Table 3.3.

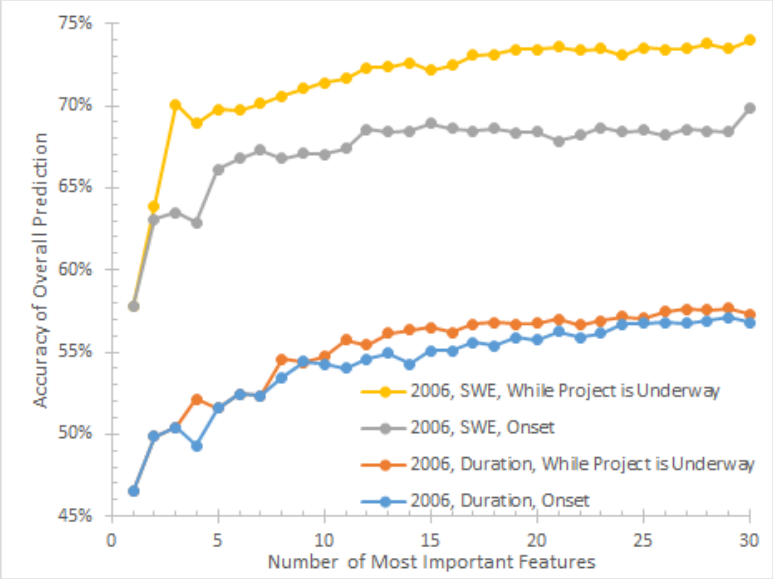


Figure 3.1. Accuracy of APL's Software Project Duration and Software Effort (with reduced, prioritized feature set)

The APL Software Life-Cycle Prediction model results clearly show that ML models can quickly be developed and trained using only a relatively small number of projects, a very small number of features, and a large amount of missing data. Furthermore, the resulting predictions for a software project's duration and total effort can be reasonably accurate at the project onset, and can then improve slightly over time by tracking the effort that is expended over the lifecycle. Only about 5 to 15 features are required to achieve reasonable predictions. The most important features for the predictions were identified; most of them are easy to obtain or estimate.

UC-Davis Software Development Forecasting Model

UC-Davis developed models that predict project duration, number of commits, and popularity using all available historical data of completed projects in the January 2018 snapshot, starting from the first commit of software. Table 3.4 shows the best-case overall prediction accuracies that can be obtained with these models and all of this data. The best-case overall accuracy of the prediction estimate for project duration is 84% and the best-case overall accuracy of the prediction estimate for the number of commits is 72%. Predictions for popularity were less accurate. These results indicate that the features in the GitHub database will be very useful for predicting software project duration and to a lesser extent the predictions for the number of commits. It appears that additional features will be necessary to improve the predictions for software popularity.

Additionally, Table 3.4 also shows that the best-case overall accuracy results for these models vary for different context clusters of similar projects. For instance, the accuracy values for each target variable increase within certain clusters; accuracy is greater in Cluster 1 by 16% for project duration and by 24% for number of commits and in Cluster 4 by 13% for popularity. These increases suggest that clustering projects based on similar context can increase the best-case prediction accuracy and that different models may be necessary to best predict different project contexts. The descriptions of these different clusters are not available at this time, but it would be valuable to investigate this further in order to understand the project characteristics that distinguish the clusters.

Table 3.5 shows the best-case overall accuracy of the UC-Davis models that use only the 9 most important features from the full project lifetime. These results are very close to those of the models that use all available features, indicating that the reduced feature set is sufficient for accurate predictions.

Table 3.4. Full Lifetime (Best-Case) Prediction Accuracy

Target Variable	All Projects	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Number of Projects	126,799	21,462	31,918	55,065	18,354
Project Duration (months committed)	84%	99.5%	83%	80%	78%
Number of Commits	72%	96%	70%	62%	69%
Popularity (number of stars)	49%	46%	48%	42%	62%

Table 3.5. Full Lifetime (Best-Case) Prediction Accuracy with Reduced Feature Set

Target Variable	**All Features** (All Clusters)	**9 Most Important Features** (All Clusters)
Project Duration (months committed)	84%	84%
Number of Commits	72%	74%
Popularity (number of stars)	49%	48%

Table 3.6 shows the accuracy results of the forecasting models, which predict the target variable in the final snapshot using features from a snapshot taken 6 months after project starts. These results are averaged over each of the 4 clusters (i.e., include 126,799 projects). These forecasting results show that data from only the first 6 months into a project can predict future outcomes, reaching accuracies of approximately 50% for both project duration and number of commits.

Table 3.7 identifies the most important features that influenced the UC-Davis predictions and forecasting. This table shows that features related to teams and commit activity are the most important for the UC-Davis models.

Table 3.6. Forecasting Accuracy (Averaged Over All Clusters)

Target Variable	Prediction of target variable at last snapshot given 6 month snapshot	Prediction of target variable at last snapshot given all data
Project Duration (months committed)	53%	84%
Number of Commits	50%	72%
Popularity (number of stars)	41%	49%

Table 3.7. Most Important Features for the UC-Davis Predictions and Forecasting

Feature Category	Most Important Features
Commit Activity Data	First Commit Date, Months Committed

Team Member Data	Team Size, Number of Commenters, Number of Pull Request Mergers, Average Months Active, Standard Deviation (SD) Months Active, Average Commits per Month, SD Commits per Month

In summary, the UC-Davis analysis shows excellent results for being able to forecast project duration and the number of commits only 6 months into a project. Only 9 features are required to achieve these forecasts. The most important features for the predictions were identified; all of them easily obtained with automation tools that track software development activities. Additionally, UC-Davis uncovered clusters of projects that if better understood could lead to improved models and accuracy predictions.

Rotunda Solutions Investigation of Opportunities for Analytic Intervention

The Rotunda Solutions effort focused on identifying strategic opportunities to leverage ML and AI at key points in the overall DoD procurement process. It extended academic research and state-of-the-art quality management principles to identify opportunities to improve the likelihood of successful software development outcomes. It also developed initial conceptual mock-ups to explore potential applications, including a defect prediction platform.

Rotunda Solutions adopted a basic stage-gate model to represent the general structure and stages of a DoD procurement and project development effort. Multiple opportunities are identified in each stage where analytics, ML, and other modern techniques can assist project managers. First, analytics can provide metrics and insights to support the project manager's yes/no/hold decision for whether the project should move to the next development stage. Second, analytics and ML can facilitate the search and interpretation of DoD procurement and development data sets so that decision makers have better access to historical data. Third, analytics can be run on this historical data to provide insights that can inform future projects. The application of modern techniques within a basic stage-gate model for a typical DoD procurement and development project can be envisioned as follows.

Stage 1: Idea Generation/Need Analysis. Analyze the internal unstructured documents from the program office and communications between suppliers and procurement officials. Then apply problem identification analytics to define the problem to be solved, considering the following 5 major groups/factors: need spotting, solution spotting, mental invention, market research, and trend. The literature shows a clear trend in savings of time and resources during the development process by maximizing the effectiveness of the idea generation stage.

Stages 2 and 3: Proposal Development and Response. Analyze internal unstructured documents from the program office and communications as they relate to proposal development and response. Use qualitative techniques such as focus groups, in-depth interviews, and surveys to determine factors associated with development success and failure. Additionally, use natural language processing (NLP) techniques to prepare the documents for further analysis. Both methods can identify key mechanisms and characteristics of software development success.

Stage 4: Contract and Award. Identify keywords through analysis of prior software contracts. Use NLP and topic extraction on legal documents surrounding the final selection of the supplier, contract vehicles, set-asides, and all stipulations to determine content. This can increase the ease of detecting associations between numerous demographic and supplier characteristics and

software development performance. It also provides the ability to build a grading system and general profile of contractors and their performance on projects.

Stage 5: Software Development. Gather representative data regarding project management metrics, code base, and development metrics, and compile a list of metrics that can help identify the likelihood of success of a DoD software development project. This helps the DoD in two ways: first by identifying projects that are likely to succeed or fail in each stage; and second by informing cost and time estimates for future software acquisition projects. Alternatively, analyze code to inform the development of ML tools to assist project managers and developers understand the state of their code. Potential benefits of this analysis include tools that can rapidly identify errors and increase efficiency for automation, audits, process checkpoints, and standardization.

Stage 6: Implementation. Harness available information on users, development, delivery personnel, and performance metrics of the software system. Measure the efficacy of the deployed or implemented software systems through metrics such as dependability, system performance, extensibility, and cross-platform functionality. This provides a post-mortem analysis of the efficiency and effectiveness of the software and the development process, allowing DoD to learn from past experience and increase the likelihood of future development success.

Conceptual Mock-Ups

Rotunda Solutions aims to help the DoD in four ways: (1) understand the potential impact of variables, decisions, and project characteristics on project budget and effort, based on historical data of similar projects; (2) make data-informed project decisions pertaining to the adjustment of project structure, methods, and other details; (3) create and explore what-if scenarios to promote better planning; and (4) encourage transparency and traceability of factors and decision-points affecting project performance. To this end, a number of concepts offer potential for further development and exploration. For instance, the concept of an “intelligent” burn-down chart is especially intriguing. Given sufficient sprint data and historical trend data, effort estimation tools and ML algorithms can be leveraged to make real-time predictions and issue alerts when estimates of team effort needs a closer review. Also, a defect prediction algorithm may be able to support risk mitigation activities and improve resource allocations.

Focus Area: Defect Prediction Platform

Software defect prevention is an essential part of the quality improvement process; timely identification of defects is important for efficient resource allocation, increased productivity, and risk mitigation, yet complete testing of an entire system is generally not feasible due to budget and time constraints. Studies show that the majority of software bugs are often contained within a small number of modules. To more rapidly identify these modules, Rotunda Solutions developed a system to automatically process code files and output code complexity metrics. They built off extensive industry research and tested representative NASA software modules using NN, SVM, Gaussian mixtures, and ensembles of ML techniques. The NN model performed best and was selected for production.

The NN model consists of 8 hidden layers, each layer becoming smaller until converging on a single probability to represent the existence of defects in the file. This model learns to assign importance weights to each of the 17 features and to combine these features in non-linear ways

to identify any potential defects. The NN can then be used to give a probability of defects for future files. This could help the management team in three ways: (1) to recognize the likeliest modules to have defects and allocate corrective resources effectively; (2) to provide an overview of the riskiest code modules to identify opportunities to re-architect the application; and (3) to understand the risk of deployment in production by an automated code complexity review.

Conclusions

The Rotunda Solutions exploration outlined the potential benefits of harnessing ML/AI throughout the DoD software acquisition lifecycle. These benefits include increased accuracy of budget predictions, comprehensive planning, mitigation of expensive defects, and transparency. Rotunda Solutions also identified many opportunities and applications that may improve DoD software development and estimation practices.

D.3 Implications of the Study Results for DoD

This ML study demonstrated promising results by creating models with publicly available software project data. It uncovered a promising approach (the APL Life-Cycle Prediction Model) that can be used to develop good predictions of software duration and effort in the early stages of software procurement and development. The study also uncovered another approach (the UC-Davis Forecasting Model) that can further improve project estimates once software development has been underway for 6 months or more. Finally, the Rotunda Solutions defect density model can highlight modules requiring additional resources and risk mitigation efforts.

The generalizability of these models to DoD software projects requires validation. For instance, a pilot study could be conducted with a small subset of DoD projects. Ultimately, strategies can be developed to enable DoD leadership to effectively leverage ML models.

One strategy could entail a strong centralized mandate for DoD software development teams to provide project data to DoD oversight personnel for evaluation with the APL and UC-Davis models.

A second, more streamlined and evolutionary strategy is to provide these models as tools for DoD software development teams to use as part of best practices to guide their development plans. This strategy would alleviate the exchange of data and would allow a more collaborative community effort to refine the models and resulting software development performance over time.

D.4 Caveats and Limitations

It is important to note that there are significant differences between the software repositories used in this work and important classes of software acquired by the DoD. For example, embedded software used in DoD weapons platforms is typically marked by high complexity, with low tolerance for reliability, availability, safety, and security issues. Although the testbeds on which the ML approaches were applied do contain some NASA software, only a small subset at best of the systems providing data are expected to have similar characteristics. As a result, it is important to view these results as showing a potential method that would be applicable to DoD programs and could learn characteristics of interest within that environment. While the method may be of interest, the specific results summarized may not directly carry over to some types of software present in the DoD environment.

Jan 11, 2019 5

Software is Never Done: Refactoring the Acquisition System for Competitive Advantage

Defense Innovation Board

TL;DR (v1.5, 11 Jan 2019)

Key themes:

- **Software is ubiquitous and U.S. national security relies on software.** Well-equipped and well-trained warfighters provide the capability necessary to defend the nation, but software critically enables that mission. The ability to develop, procure, assure, and deploy software is central to national defense and integrating with allies and partners.
- **Speed and cycle time are the most effective metrics for software.** Software is a critical element of the Department's approach to executing missions, collaborating with allies, and managing its operations. DoD needs to deploy & update software at the speed of (mission) need, and execute within the OODA loop of our adversaries to maintain advantage.
- **Software is made by people, for people, so digital talent matters.** DoD's current personnel processes and culture will not allow its military and civilian software capabilities to grow nearly enough. New mechanisms are needed for attracting, educating, retaining, and promoting digital talent, and providing the ecosystem that enables them to succeed.
- **Software is different than hardware (and not all software is the same)**. Hardware can be developed, procured, and maintained. Software is an enduring and evolving capability that must be supported and continuously improved throughout its lifecycle. The DoD acquisition process and culture need to be streamlined for effective delivery and oversight of multiple types of software-enabled systems, at scale, and at the speed of relevance.

Why it matters:

- **The threats that the U.S. faces are changing rapidly**, and DoD's ability to adapt and respond is defined by its ability to develop and deploy software to the field rapidly.
- **The current approach to software development is a leading source of risk to DoD;** it takes too long, is too expensive, and exposes warfighters to unacceptable risk.
- **Software should enable a more effective force,** strengthening our ability to work with allies, and improving the business processes of the Department.

Who needs to do what to fix this:

- **Congress**: **Create new statutes streamlined for software,** providing increased insight to reduce the risk of slow, costly, and overgrown programs, and enabling rapid deployment and continuous improvement of software to the field.
- **OSD**: **Create cross-program/cross-service digital infrastructure** that enables rapid deployment, scaling, testing, and optimization of software as an enduring capability; manage them using modern development methods; and eliminate the existing hardware-centric regulations and other barriers.
- **Services**: **Establish SW development as a high visibility, high priority career track** with specialized recruiting, education, promotion, organization, incentives, and salary.
- **Contractors**: **Adopt DevSecOps practices/culture;** prioritize speed as the critical metric.

Timeline for action: We need to start now (FY19-FY20); inaction has serious consequences.

Table of Contents[1]

v1.4, 5 Jan 2019

List of Vignettes (tentative)

Supplemental Information (separate document)

[1] Draft outline for the report; page numbers reflect target length for each chapter

End of Overall Report: 32 pages (main) + 160 pages (SI) = 192 pages

Chapter 0. README

V1.4, 11 Jan 2019

Software is ubiquitous in the world around us and U.S. national security is critically dependent on the capabilities of its software. To maintain our military advantage, the Department of Defense (DoD) must be able to develop, procure, deploy, and continuously improve software faster than our adversaries. Recognizing that not all "software" is the same – it can range from off-the-shelf, non-customized applications to highly-specialized, embedded code running on custom hardware – it is critical that the right tools and methods be applied for each type. Commercial industry has demonstrated that software can have a transformative impact on business and society. Companies that thrive take advantage of software, computing, and networking – and the rapid cycles of improvement they allow – to the maximum extent possible. At the present time, DoD's software prioritization, planning, and acquisition processes are among the worst bottlenecks for deploying capability to the field at the speed of relevance. This puts the U.S. Armed Forces at risk, reduces the efficiency of DoD operations, and drives away the very people who are most needed to develop software that is critical to national security.

What this report is about. This manifesto describes the output of the Defense Innovation Board (DIB) Software Acquisition and Practices (SWAP) study. The DIB was charged by Congress[1] to recommend changes to statutes, regulations, processes, and culture to enable the better use of software in the DoD. We took an iterative approach, releasing a sequence of concept papers describing our preliminary observations and insights (the current versions of which are included in Appendix A) and using those to encourage dialogue with a wide variety of individuals and groups to gain insights into the current barriers to implementing modern software. This report attempts to capture key insights from these discussions in an easy-to-read format that highlights the elements that we think are critical for the Department's success.

This report is organized as follows:

- **TL;DR:** a one-page summary of 12 months of work for those not likely to read the full report; please take the time to read it. (TL;DR is Internet slang for "too long; didn't read").
- **README** (this document): a more detailed five-page summary of the report. If your boss read the TL;DR, thought it was intriguing, and asked you to read the entire report and provide a short summary, cut and paste this chapter and you should be good-to-go. A final bonus page has a list of the recommendations, so you can pretty much stop at that point – or better yet, stop after suggesting to your boss she adopt them all. (A README file is used by the open source software community to provide essential information about a software package.)
- **Chapters 1-4:** short descriptions of key areas we felt were important to expound upon. If you attach the TL;DR to any one of these as a preface, it should be comprehensible.
- **Chapter 5:** a more detailed description of the recommendations and our rationale.

[1] 2018 NDAA, Sec. 872. Defense Innovation Board analysis of software acquisition regulations.

- **Supplementary Information:** To ensure that the main body of the report satisfies the staple test[2] and the takeoff test,[3] we put most of the additional information generated during the study in a set of appendices. These provide a wealth of examples and evidence, but we took care to put our essential arguments up front for less wonky types.

Key Themes. In order for the report to be useful, we felt we should come up with a few key themes that could be used to drive home the message of the report. Here they are (again):

1. Software is ubiquitous and U.S. national security relies on software.
2. Speed and cycle time are the most effective metrics for software.
3. Software is made by people, for people, so digital talent matters.
4. Software is different than hardware (and not all software is the same).

Software is ubiquitous and U.S. national security relies on software. The rise of electronics, computing, and networking has forever transformed the way we live: software is a part of almost everything that we interact with in our daily lives, either directly through embedded computation in the objects around us or indirectly through the use of information technology through all stages of design, development, deployment, and operations. Our military advantage, coordination with allies and partners, operational security and many other aspects of the DoD are all contingent upon our software edge and the lack thereof presents serious consequences. Software drives the competitive advantage: what makes weapons systems sophisticated is the software, not (just) the hardware.

Commercial trends show what is possible with software, from the use of open source tools to agile development techniques to global-scale cloud computing. Our adversaries are active players in the world of software and so they are increasingly able to develop weapons systems faster than we can, capitalizing on their advantage in software development. Meanwhile, they exploit our vulnerabilities via cyber-attacks to steal, undermine, and inhibit our capabilities. The incoming generation of military and civilian personnel began life digitally plugged-in, with an innate reliance on software-based systems. They will demand new concepts of operations, tactics, and strategies to maintain the edge they need. If the Department can reform its acquisition processes and adjust its culture and personnel policies before its too late, this software-savvy generation can still set the Department on the right course.

Speed is the ultimate software metric. Being able to develop and deploy faster than our adversaries means that we can provide more advanced capabilities and be more responsive to our end users. Faster reduces risk by focusing on the critical functionality rather than over-specification and bloated requirements. It also means we can identify trouble earlier and take faster corrective action which reduces cost, time, and risk. Faster leads to increased reliability: the more quickly software/code is in the hands of users, the more quickly feedback can focus efforts to deploy greater capability, sooner. Faster gives us a tactical advantage on the battlefield because we can operate and respond inside our adversaries' observe–orient–decide–act (OODA) loops.

[2] Any report that is going to be read should be thin enough to be stapled with a regular office stapler.
[3] Reports should be short enough to read during takeoff, before the movies start and drinks are served.

Software is about people. As Steve Jobs observed,[4] one of the major differences between hardware and software is that for hardware the "dynamic range" (ratio between the best in class and average performance) is, at most, 2:1. But, the difference between the best software developer and an average software developer can be 50:1, or even 100:1, and putting great developers on a team with other great developers amplifies this effect. Today, in DoD and the industrial base that supports it, the people with the necessary skills exist, but instead of taking advantage of their skills we put them in environments where it is difficult for them to be effective. In DoD proper, we do not take advantage of already existing military and civilian personnel skill sets by offering pay bonuses, the ability to stay in their specialization, or access to early promotions. Skilled software engineers and the related specialities that are part of the overall software development team need to be treated like Special Forces and the United States must harness their talent for the great benefits that it can provide.

Software is different than hardware. Over the years, Congress and DoD have developed a sophisticated set of statues, regulations, and instructions that govern the development, procurement, and sustainment of defense systems. This process was developed in the context of the Cold War, where major powers developed aircraft carriers, nuclear weapons, fighter jets, and submarines that are extremely expensive and require tremendous access to capital and natural resources. Software, on the other hand, is something that can be mastered by a ragtag bunch of teenagers with very little money – and can be used to destabilize world powers. Currently most parts of DoD develop, procure and manage software like hardware, assuming that it is developed based on a fixed set of specifications, procured after it has been shown to comply with those specifications, and then "maintained" by block upgrades and new procure- ments. But software development is fundamentally different than hardware development, and software should be developed, deployed, and continuously improved using much different cycle times, support infrastructure, and maintenance strategies. Testing and validation of software is also much different than hardware, both in terms of the ability to automate but also in the potential vulnerabilities found in software that is not kept up to date. Software is never "done," and must be managed as an enduring capability that is treated differently than hardware.

The First Three Things to Do. The Department's current approach to software is a major driver, if not *the* major driver, of cost and schedule overruns for major defense acquisition programs (MDAPs). Congress and DoD need to come together to fix the acquisition system for software because it is the primary sources of its acquisition headaches.

Bringing about the type of change that is required to give DoD the software capabilities it needs to stay ahead is going to take a significant amount of work. While it is possible to use the current acquisition system and DoD process to develop, procure, deploy, and continuously improve DoD software, the statutes, regulations, processes, and culture are debilitating for software. The current approach to acquisition was defined in a different era, for different purposes, and only works for software projects through enormous effort and creativity. Congress, the Office of the Secretary of Defense, the Armed Services, defense contractors, and the myriad of government

[4] Steve Jobs - The Lost Interview, 2012.

and industry organizations involved in getting software out the door need to come together and make major changes. Here are the three most important things to start with:

1. **Create new statutes streamlined for software** that provide more insight while enabling rapid deployment and continuous improvement of software to the field (bear with us).

2. **Create cross-program/cross-service digital infrastructure** that enables rapid deployment, scaling, and optimization of software as an enduring capability, managed using modern development methods in place of existing (hardware-centric) regulations.

3. **Create new paths for digital talent (especially *internal* talent)** by establishing software development as a high visibility, high priority career track with specialized recruiting, promotion, organization, incentives, and salary.

None of these can be done by a single organization within the government, so they are going to require a bunch of hard-working, well-meaning people to work together to craft a set of statutes, regulations, processes, and (most importantly) a culture that recognizes the importance of software (theme 1), the need for speed and agility (theme 2), the critical role that smart people have to play in the process (theme 3), and the impact of inefficiencies of the current approach (theme 4). In many ways this mission is as challenging as any combat mission: while participant's lives may not be directly at risk in defining, implementing, and communicating the needed changes to policy and culture, the lives of those who defend our nation ultimately depend on the ability of the Department to redefine its approach to delivering combat-critical software to the field.

New statutes, streamlined for software. Congress has created lots of workarounds to allow the DoD to be agile in its development of new weapons systems, and the Department has used many of these to good effect. But the default statutes, regulations, and processes that are used for software too often rely on the traditional hardware mentality (repeat after me: software is different than hardware) and those practices do not take advantage of what is possible (or frankly necessary, given the threat environment) with modern software. We think that a combination of top-down and bottom-up pressure can break us out of the current state of affairs, and creating a new acquisition pathway that is tuned for software (of various types) could make a big difference. To this end, Congress and DoD should prototype and eventually create mechanisms for ideation, appropriation, and deployment of software-driven solutions that take advantage of the unique features of software (versus hardware) development (start small, iterate quickly, terminate early) and provide purpose-fit methods of oversight.

Cross-program/cross-service software digital infrastructure: We need to create, scale, and optimize an enterprise-level architecture and supporting infrastructure that enables creation and initial fielding of software within 6 months and continuous delivery of improvements on a 3 month cycle. This "digital infrastructure," common in commercial IT, is critical to enable rapid deployment at the speed (and scale) of relevance. In order to implement this recommendation, Congress and Department leadership must figure out some ways to incentivize the Services and defense contractors to build on a common set of tools (instead of inventing their own) *without* just requiring that everyone use one DoD-wide (or even service-wide) platform. Similarly, OSD is going to have to define some non-exceptions-based alternatives to (or at least pathways through) JCIDS,

PPB&E, and DFARS[5] that are optimized for software. DOT&E will need new methods for operational test and evaluation that match the software's speed of relevance, and CAPE is going to have to capture better data and leverage AI/ML as a tool for cost assessment and performance evaluation. Finally, the Services are going to need to identify, champion, and measure platform-based, software-intensive projects that increase software effectiveness, simplify interconnectivity among allies, and reform business practices. Subsequent chapters in our report provide specific recommendations on each of these areas.

New paths for digital talent. The biggest enabler for great software is providing great people with the means to contribute to the national security mission. While the previous recommendations speak to providing the tools and infrastructure DoD technologists need to succeed, it is equally important that the Department's human capital strategy allow them to even do this work consistently in the first place. Particularly important is to provide new career paths for digital talent and enable the infrastructure and environment required to allow them to succeed. The current GS system favors time-in-grade over talent, and this simply will not work for software. The military promotion system has the same problem. As with sports, medicine, and law, great teams make a huge difference in software and we need to make sure those teams have the tools they need to succeed and reward them appropriately -- through recognition, opportunities for impact, and pay. Advanced expertise in procurement, project management, evaluation and testing, and risk mitigation strategies will also be needed to create the types of elite teams that are needed. To get started, Congress should create a two-year national security waiver from the GS system in selected digital technology areas required for software and the Services should use this and other authorities to identify and nurture civilian and military talent with software development expertise. A key element of success is finding a way to keep talented people in their roles (rather than transferring them out because it is the end of their assignment), and promote people based on their abilities, not based on their years of service.

The Next Ten Things to Do. The items above are what we think Congress and the Department should focus on as the first three things to accomplish. Without dramatic change, the rate at which we can make improvements is far outpaced by the rate at which the problem itself gets worse. With demonstrated progress on these three there is then a long list of other things that need to be done, ranging from changing the law to changing the way people work. We created a list of 30 recommendations for change that we thought were important, and then asked everyone with whom we interacted in the building on this report to vote on the ones they thought would make the most difference. Here is the current snapshot of the top 10 recommendations based on that voting:

Rank	**Recommendation**	□	□
	This table will be filled in for the final report		
	The items here will come from a longer list of recommendations (see cheat sheet)		
	The order will be determined using a leaderboard, hosted on [TBD service]		

[5] Common DoD acronyms are defined in Appendix F (Acronyms, Inside Jokes, and Catch Phrases).

	Participants in SWAP study activities will be allowed to cast a vote		
	More details coming later; look at the full list of options (cheat sheet) for now.		

More details on these (as well as top 10 lists for the biggest barriers and the most useful tools that are not currently available for use) are included in Chapter 5 (What Would the DIB Do) and the supplementary information.

Getting Started Now. The types of changes that we are talking about will take years to bring to complete fruition. But it would be a mistake to spend two years figuring out what the answer should look like, spend another two years prototyping the solutions to make sure we are right, then spend two to four more years implementing the changes in statutes, regulations, processes, and culture that are actually required. Let's call that approach the "hardware" approach. Software is different than hardware and the approach to implementing change for software should be different as well.

Many of our DoD issues could be addressed by adopting existing best practices of industry for agile development, software as a service, use of modern (cloud) infrastructure, tools, computing and shared libraries, and software logistics and support delivery systems for software maintenance, development, and updating (patching). We do not need to study these, we need to get going and implement them. Here are some specific suggestions for what to do starting *now*:

- FY19 (create): High-level endorsement of report vision and support for activities that are consistent with the desired end state (i.e., DevSecOps and enterprise-level architecture and infrastructure). If you are reading this and are in a position of leadership in your organization, pass this on to others with your seal of approval and a request for your team to develop 2-3 plans of action for how it can be applied in your domain. If someone comes to you with a proposal that aligns with the objectives we have outlined here, find a way to say yes.
- FY20 (deploy): Initial deployment of authorities, budgets, and processes for SWAP reform. Choose immediate representative programs to act according to the themes, flavors, and recommendations in this report, implement now, measure results, and modify approaches. Let's implement this report the way we implement modern software.
- FY21 (scale): Streamlined authorities, budgets, and processes enabling SWAP reform at scale. In this time frame, we need a new methodology to estimate as well as determine the value of software capability delivered (something not based on lines of code).
- FY22 (optimize): All DoD software development projects transition (by choice) to software-enabled processes, with talent & ecosystem in place for effective management & oversight.