

Jan 11, 2019

5

**Software is Never Done: Refactoring the Acquisition System for Competitive**

Department of Defense

OFFICE OF PREPUBLICATION AND SECURITY REVIEW

**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<sup>1</sup>**

v1.4, 5 Jan 2019

<u>Chapter 0. README</u>	1
<ul style="list-style-type: none"><li>• Key themes and what needs to be done, in 5 short pages</li><li>• Interlude: Recommendations Cheat Sheet (preview of Chapters 4, 5)</li></ul>	
<u>Chapter 1. Who Cares: Why Does Software Matter for the DoD?</u>	6
<ul style="list-style-type: none"><li>• Weapons and Software and Systems, oh my! A taxonomy for DoD</li><li>• Where are we coming from, where are we going?</li><li>• What kind of software will we need to build?</li><li>• What challenges do we face (and consequences of inaction)?</li></ul>	
<u>Chapter 2. I Don't Get It: What Does It Look Like To Do Software Right?</u>	11
<ul style="list-style-type: none"><li>• How it works in industry (and can/should work in the DoD): DevSecOps</li><li>• Empowering the workforce: building talent inside and out</li><li>• How doing software right enables superior national security &amp; more insight for Congress</li><li>• Eye on the prize: What's the R&amp;D strategy for our investment?</li></ul>	
<u>Chapter 3. Been There, Done That: Why This Hasn't Already Happened?</u>	16
<ul style="list-style-type: none"><li>• Brief summary and grade assignment of the 40 reports that came before us</li><li>• Breaking the spell: Why nothing happened before, but why this time could be different</li><li>• Consequences of inaction: Increased attack surface, shipping risk to the warfighter</li><li>• While you were out: A quick look at the competition, state and non-state actors</li></ul>	
<u>Chapter 4. How Do We Get There From Here: Three Paths for Moving Forward</u>	21
<ul style="list-style-type: none"><li>• Option 1: Make the best of what we've got (just like we make our warfighters do)</li><li>• Option 2: Tune and tweak the system to optimize for software</li><li>• Option 3: A new acquisition pathway for software to force change in the middle</li></ul>	
<u>Chapter 5. What Would the DIB Do: Our Recommendations for Congress and DoD</u>	26
<ul style="list-style-type: none"><li>• The first three things to do (starting <i>now</i>)</li><li>• The next ten things after that (by popular demand)</li><li>• Kicking the can down the road: Other things that we couldn't figure out how to fix</li></ul>	
Acknowledgements	30
<b>List of Vignettes</b> (tentative)	
1. Kessel Run: Insourcing Software Development for Mission-Critical Applications	
2. Continuous ATOs: The JIDO Approach	
3. F22: DevOps on a Hardware Platform	
4. JMS: Seven Signs That Your Software Is In Trouble	
5. Kessel Run: How Not to Incentivize the DoD Workforce	
6. SWAP Study: Making It Hard to Volunteer to Help	
7. What Happens When Congress Gets Inside DoD's OODA Loop	
<b>Supplemental Information</b> (separate document)	

---

<sup>1</sup> Draft outline for the report; page numbers reflect target length for each chapter

<u>Appendix A.</u> DIB Guides for Software	S1
<ul style="list-style-type: none"><li>● <a href="#">Ten Commandments of Software</a></li><li>● <a href="#">Metrics for Software Development</a></li><li>● <a href="#">Do's and Don'ts for Software</a></li><li>● <a href="#">Detecting Agile BS</a></li><li>● <a href="#">Is Your Development Environment Holding You Back?</a></li><li>● <a href="#">Is Your Compute Environment Holding You Back?</a></li><li>● <a href="#">Site Visit Observations and Recommendations</a></li><li>● How To Defend Your Agile Budget (tentative)</li><li>● How to Know You're Getting Your Money's Worth (tentative)</li></ul>	
<u>Appendix B.</u> SWAP Working Group Reports (DIB remix)	S41
<ul style="list-style-type: none"><li>● Acquisition Strategy</li><li>● Appropriations</li><li>● Contracts</li><li>● Data and Metrics</li><li>● Infrastructure</li><li>● Modernization/Sustainment</li><li>● Requirements</li><li>● Security Certification/Accreditation</li><li>● Testing and Evaluation</li><li>● Workforce</li></ul>	
<u>Appendix C.</u> Analysis the Old-Fashioned Way: A Look at Past DoD SW Projects	S71
<ul style="list-style-type: none"><li>● Earned value data analysis</li><li>● SRDR data analysis</li></ul>	
<u>Appendix D.</u> <del>Replacing</del> Augmenting CAPE with AI/ML	S91
<ul style="list-style-type: none"><li>● Analysis of ML/AI Intervention Points in DoD Software Acquisition</li><li>● Machine Learning for Software Development Effort Estimation</li><li>● Metrics for success for open source software</li></ul>	
<u>Appendix E.</u> Top 10 Lists: Recommendations, Obstacles, Tools	S111
<u>Appendix F.</u> Acronyms and Catch Phrases	S135
<u>Appendix G.</u> Required Content That Nobody Ever Reads	S120
<u>Appendix L.</u> Legislative and Regulatory Language Templates	S130
Index [ <i>substantive</i> index to entire report, to allow people to find what they are looking for]	S150

End of Overall Report: 32 pages (main) + 160 pages (SI) = 192 pages