DEPARTMENT OF THE AIR FORCE
WASHINGTON DC

OFFICE OF THE ASSISTANT SECRETARY

25 APR 2019

MEMORANDUM FOR ALL ACQUISITION PERSONNEL

FROM: SAF/AQ

SUBJECT: Toyota, DevSecOps, and the Chief Software Officer

Teammates:

I hope this memo finds you well and excited about the amazing things happening in Air Force acquisition. If you've been on the fence—or just waiting to see what's next—find a way to add your personal spark to the engine. I know it takes time for empowerment to reach all levels of an organization, but stay positive and patient—radical delegation (below the Senior Materiel Leader level) is our next focus area to energize envelope-pushers and barrier-breakers!

The 2030 threat we face is a Pandora's box of many possibilities: artificial intelligence, quantum systems, biotech, and ubiquitous sensing are poised to be technological titanics whose bow waves will rock national security. Our acquisition system must be able to ride them agilely through perilous waters. No matter which sails first, we will need software excellence to fuel the high-stakes regatta. On our watch, it is a universal accelerant.

Good news! We are off to a great start on software reform. Kessel Run, Kobayashi Maru, Space Camp, LevelUP, Bespin, B-21, F-22 Pipeline, Unified Platform, Protected Tactical Enterprises Service, to name a few, are all fully agile. Hats off to you all! If you're out in the field wanting to shift your current software program—or just get in the know for a future one—we want to make this easy and provide tools to succeed. But you probably know more about agile Dev(Sec)Ops than you think.

DevSecOps is mostly lean automotive manufacturing ideas applied to software development. A bit of history: in the early 1950s, the automotive industry underwent its second manufacturing revolution—Toyota's kanban and andon cord systems—since Henry Ford's creation of the moving assembly line 40 years earlier. It doesn't get the same notoriety as Ford, but it introduced two important ideas into manufacturing: (i) **early anything, like late anything, is bad**, and (ii) **ubiquitous production line quality control—vice final assembly only—keeps problems small and manageable and velocities, high**.

Rather than having parts produced, delivered, and stored ahead of assembly, Toyota's just-in-time, just-in-quantity kanban system lowered carried inventory and leaned the production floor, increasing output while lowering cost. Combined with continuous quality inspections—and line halts to address early-stage issues before they traveled—these new manufacturing ideas thrived and spread to other industries. In hindsight, they may seem obvious, but I'm sure the first worker who pulled the production-halting andon cord took a deep breath first, as did the manager who let stockpiles deplete to record lows in the name of efficiency and agility!

How does this tie to software? In the 1970s, "waterfall" software development created a production line process where one module of code was completed and handed to the next, sequentially, without quality being fully assessable till the end, often with insufficient time to debug. Development cycles took so long that early modules became long-term "carried inventory", resulting in higher coding overhead and the digital equivalent of scrap, rework, and repair (SRR). With the automotive industry staring everyone in the face, why didn't they shrink the waterfall to ripples using lean principles? Why is it possible now?

Great question!

We have something early software developers were missing: modern software production tools. Walk a modern automotive line and the tools come from myriad companies pushing new boundaries of capability, automation, and efficiency for lean manufacturing. As such, few are standalone; they are designed to be integrated, globally optimized, and continuously improved, especially in the era of machine learning. When software development was in its waterfall phase, analogous tools did not exist. It wasn't until major internet search companies couldn't update increasingly-complex code quickly enough in the late 1990s that speedier Cloud-based development was born. Firstly, it increased coding velocity (as much as a factor of 200!), lowering carried inventory and all its parasitic effects. Secondly, it allowed final assembly tasks—quality, security, and test—to be distributed across the software production line so that smaller increments of code could be fielded rapidly and continuously. Big endgame problems were replaced by smaller early-game ones, increasing time for new functionality by 20-30% and reducing time for security by 40-50%. The rest you know; digital production in this century has seen the same revolution physical production did in the last. The name "software factory" is a moniker well chosen.

I am so proud of our software programs, but the path to actual coding is a machete march through a virgin jungle. We do not have standard software development tools approved for everything we acquire—bombers to base defense—and as a result, every software factory/program has to evaluate, select, and secure its own development tools. In this case, trailblazing is hard duplicative work!

Contrast this scenario with information technology (IT): any new Air Force organization can make phone calls, send emails, and hold video teleconferences at all levels of classification on day one because of pre-approved communication devices for all Air Force information. We want them expert in their mission, not IT. We are missing that critical function for software development. When a new software program initiates, it should have a pre-approved development environment next to its phones (well, not literally). But evaluation, selection, and security across the Air Force enterprise would be a full-time job—let's make it one.

I'm creating a new position, the Air Force Chief Software Officer (CSO), to identify, secure, and approve software development infrastructure that is hardened and pre-accredited for continuous authority to operate (ATO)—across the Air Force—**so that we code faster and**

more efficiently. This will include, but is not limited to, Cloud deployments and Cloud-native apps, hardened containers with continuous ATO, commercial technology adoption, and training on microservices, DevSecOps, Agile and Cloud-native, etc. As such, the CSO will work with Air Force and Department of Defense Chief Information Officers, Office of the Secretary of Defense and Service officials, and industry leaders to speed accreditations, enable ATO reciprocity, and share/learn best practices. The CSO will also serve as the three-letter advisor to me on software matters.

So how is this different from PEO Digital? PEO Digital remains responsible for reviewing and publishing best coding practices and certifying how the Air Force, writ large, is doing on software (similar to how we review service acquisitions today). The CSO is responsible for the infrastructure on which we code and, consequently, will work closely with each PEO to make coding on day one easy and inexpensive.

So, how does the CSO work with you? The CSO will be a service provider to our programs, software factories, laboratories, and depots. Program Executive Officers and Program Managers retain full authority to select their own software tools; therefore, it is critical the CSO provide tools with delineating advantages: speed, economy, and lower overhead. This will allow the CSO to grow as value is demonstrated. Given greater licensing across the Air Force should significantly lower cost, a small percentage of demonstrated savings will be used for onsite CSO subject matter experts to keep tools current and track requirements for updates. I will announce our selectee once hired, hopefully, very soon!

I hope this makes your pivot to DevSecOps easier in our quest to become a fully agile Air Force. Lofty ambition? True. But there is one place you can have your head in the cloud and feet on the ground simultaneously: mountaintops. Let's climb this one together!

So proud of what you are accomplishing!

Cheers,

William B. Roper, Jr.
Assistant Secretary of the Air Force
(Acquisition, Technology & Logistics)