*Written Testimony of Dr. Richard M. Murray*

**U.S. House Committee on Armed Services**
**Subcommittee on Cyber, Information Technologies, and Innovation**

**"Too Critical to Fail: Getting Software Right in an Age of Rapid Innovation"**
Wednesday, 13 March 2024, 9:00am

Chairman Rogers, Ranking Member Smith, and Distinguished Members of the Subcommittee:

Thank you for inviting me to speak with you this morning on the topic of software development in the Department of Defense (DoD).

My name is Richard Murray and I am a Professor of Control & Dynamical Systems and Bioengineering at the California Institute of Technology. From 2016 to 2021 I was a member of the Defense Innovation Board (DIB) and I co-chaired, along with Michael McQuade, the DIB's study on Software Acquisition and Practices (SWAP), which was established as part of the 2018 National Defense Authorization Act (NDAA). Our report, titled "Software is Never Done" was released in May 2019.[1]

One of the key findings in our report was that Congress and DoD have been talking about the importance of software and struggling with how to make better use of software in support of national security for decades. In many ways, the DIB report of 2019 was just a rephrasing of a 1987 Defense Science Board Task Force on Military Software,[2] written 32 years earlier, which itself identified over 30 previous studies on this same topic. As we described in our report (Chapter 3: Been There, ~~Done~~ Said That), we know what we need to do, we just need to figure out how to actually do it. While software is always changing, and perhaps never more so than it is now, its importance to the DoD mission is clear and we must continue to find ways to take advantage of the power that software provides to improve our lives and protect our country.

In our 2019 report, we identified three key themes that I believe are still the most important points to make about getting software right:

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).

[1] https://media.defense.gov/2019/May/01/2002126689/-1/-1/0/SWAP%20COMPLETE%20REPORT.PDF
[2] https://apps.dtic.mil/sti/tr/pdf/ADA188561.pdf

I'd like to speak to each of these briefly, reiterating what was said in the SWAP report:

*"Speed and cycle time are the most important metrics for software.* [...] 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.* [...] DoD human resource policies are [often] 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. [...] 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 [often] put them in environments where it is difficult for them to be effective."

*"Software is different than hardware (and not all software is the same).* Over the years, Congress and DoD have established a sophisticated set of statutes, regulations, and instructions that govern the development, procurement, and sustainment of defense systems. [...] 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."

In preparing for this meeting, I took the opportunity to read through recent GAO reports on the implementation of the recommendations of the SWAP study. I was pleased to see that in the five years since the SWAP report was released, Congress and DoD have made substantial progress

in implementing our recommendations, including establishing new acquisition pathways for software and exploring new appropriation categories that allow software to be funded as a single budget item with no separation between RDT&E, production, and sustainment. These are important steps and they should be continued and accelerated if we are going to get software right. It is particularly important to identify ways to apply modern development practices to software that is attached to hardware (such as a major weapons system) since the capability of those systems is often software-enabled, and it would be a mistake to let hardware-centric development practices impede the ability to take advantage of the speed and cycle time that are possible for software.

In addition to these important actions that focused on the acquisitions process, DoD has implemented actions on many other primary and secondary recommendations from our report. Some of the most important are developing guidance on authority to operate (ATOs), including ATO reciprocity, which allows ATO certification to be shared across program boundaries, and continuous ATOs (cATOs), which enable continuous integration/continuous deployment (CI/CD) pipelines. As of April 2023, it appears that guidance for cATOs has not yet been published, and I would encourage DoD to do so, if this is not yet completed.

Another area of high importance is recruiting digital talent. DoD reports progress in establishing software factories that combine personnel from specialized teams, but it appears they have yet to establish a separate career track for software developers (unlike what exists for doctors, lawyers, and musicians), nor take advantage of commercial best practices for recruitment of talented personnel. A well-defined career track for software developers, including service members, will allow DoD to train and retain the skilled workforce required to design, build, and test modern software-enable systems. Modern software development teams also need to take advantage of the ability to recruit highly talented developers who may not have a traditional career path and whose grade and pay will need to be flexible in order to recruit and retain them. Demonstrated talent, rather than “time in grade”, should be a deciding factor in promoting software developers.

Finally, an area that is completely different today than it was five years ago is the role of artificial intelligence (AI) in military systems. The implications of AI will be profound across all areas of society and the field is changing so rapidly that it is impossible to predict how AI is going to progress over anything more than the next few years. What is clear is that the use of AI-based

tools is going to be essential for all areas of national security, and the US must maintain its leadership position in this critical technology area.

In the context of software development, AI is poised to revolutionize the way we write, test, and deploy code. Modern large language models (LLMs) are already capable of writing code based on descriptions of the desired function and are being used in industry to speed development. In the future, they will be integral to testing, deployment, and protection of software as well. These developments will “democratize” software development in ways that have the potential to drastically reduce cycle time for deploying new code, both for the US and for our adversaries. At the same time, software that is used in military systems is too critical to fail, and so we must find ways to harness these tools that fit the increased levels of security and safety that are required in safety- and mission-critical systems. DoD must stay on top of these developments and take advantage of current US leadership being developed in the commercial sector.

In closing, I would like to reiterate the key themes that I believe we should use to guide getting software right in an age of rapid innovation:

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).

Congress, working with DoD, plays an essential role in moving forward and breaking us out of the cycle of repeating history for another 32 years.

Thank you for your attention, and I look forward to our discussion.