A Software Epiphany by the US Defense Community Might Provide an Unexpected Boost for Agile Software Development

Recent events in the US defense community hold the promise of a major boost in the use of agile software development methods and a concomitant improvement in the effectiveness of defense software procurement and development. Software has become a critical component of major systems. Airplanes are being characterized as “flying computers.” Automotive systems have hundreds of thousands of lines of code. Software for virtual prototyping is being used to design next-generation systems, reducing their development time, risks, and costs. Software is a major component of defense programs for autonomous vehicles, artificial intelligence, and data analytics—including machine learning.

The present defense procurement process starts with the prospective user community drafting a set of detailed, rigid requirements, and specifications for the product of interest. Proposals are then solicited and a contract awarded to the lowest credible bidder. The contractor next develops a detailed, long-term plan, executes the plan, tests the final product, and delivers it for final testing to the user community. This is all closely supervised by a government contracting office. The same process is generally used for both software and hardware. Few CiSE readers will be surprised to hear that this process doesn’t work very well for developing complex software. Fred Brooks pointed this out over 30 years ago in his classic paper, “No Silver Bullet.” It’s very hard for users to draft valid, detailed requirements for complex software, and harder yet to deliver software that meets them. If the first time the users see the software is when the final version is delivered to them, it’s highly likely that the software won’t meet their initial requirements. It’s virtually guaranteed that the software won’t meet their current needs, which will have evolved

Douglass Post
Carnegie Mellon University
Software Engineering Institute

Richard Kendall
Consultant
Recognizing this, the US defense community asked several advisory groups—including the Defense Science Board (DSB), the Defense Innovation Board (DIB), and the Intelligence and National Security Alliance (INSA)—to analyze the problems and recommend solutions. These advisory groups strongly engaged commercial high tech industries such as Google, Facebook, IBM, Microsoft, and others. The advice from the advisory groups is very similar. They all recognized that software development is different from hardware procurement, and that software is never finished. They strongly recommend replacing the traditional software acquisition process with variants of disciplined agile methods that emphasize

- requirements based on “goals and outcomes and use cases” rather than long lists of detailed, rigid specifications; and
- small development teams (~10–15 professionals), who develop and deliver workable software iteratively in short sessions (2–4 weeks), and continual or continuous software builds and tests.

The DSB refers to development teams with these attributes as “software factories.” Senior government officials are now beating the drums to help accelerate the paradigm transition from rigid “waterfall” methodologies to agile methodologies.

Why should this be of interest to the computational science community? After all, almost anyone who isn’t being funded by the defense community, and who depends on developing working software for a living, probably is already using some type of agile process. At one level, agile methods are common sense because they involve continual evaluation of what has been accomplished and what next steps are needed to meet the project goals. The reason that this potential paradigm shift is important to the CiSE community is that the paradigm shift described above will affect a large portion of the scientists, engineers, and computer scientists engaged in computing of any kind in the US and abroad. The US defense budget is approximately $700 billion. If the defense community adopts agile methods, major parts of the rest of US government will likely follow, as will parts of the international computing community that look to the US defense community as a model system. The defense community supports a significant fraction of US scientific research, engineering, and product development, with applications ranging from machine learning to engineering design. It’s one of the major supporters of commercial and academic computer science and software engineering. If agile methods are fully adopted by the defense community, a working knowledge of the new paradigm will be required by a significant portion of computational science, computational engineering, and computer science graduates and undergraduates. They’ll need formal training and credentials in agile methodologies if they expect to work in the defense community. In addition, there will be research opportunities in computer science and computer engineering to improve software productivity. In the past, defense software development methods were pretty inflexible. There wasn’t much room for developing software except by the book (such as rigid waterfall methodologies). Now, a potential revolution is growing in a large portion of the federal software development community, and that can’t help but open up many new opportunities for innovation.

REFERENCES


ABOUT THE AUTHORS

Douglass Post is a principal researcher at the Carnegie Mellon University Software Engineering Institute, and an associate editor in chief of CiSE. Contact him at post@sei.cmu.edu or post@ieee.org.

Richard Kendall is a consultant for the US Department of Defense High Performance Computing Modernization Program. Contact him at richard.p.kendall.ctr@mail.mil.