SWAP Program Visits: Questions and Observations

Programs Reviewed

Reviewed 6 programs to date:

- Next Generation fighter jet
- Next Generation ground system
- Kessel Run—AOC Pathfinder
- Space tracking system
- Naval radar system
- Cross-service business system

What we hope to understand:

- Why is the software the way it is?
- How have you gone about developing and deploying it?
- What constraints/obligations have you been under and what would be your recommendations to change those?

Standard Questions

- What is the coding environment and what languages/SW tools do you use?
- What do the software and system architectures look like?
- What is the computational environment (processing, comms, storage)?
- How is software deployed and how often are updates delivered to the field?
- What determines the cycle time for updates?
- How does software development incorporate user feedback? What is the developer-user interface? How quickly are user issues addressed and fixed?
- How long does it take to compile the code from scratch?
- How much access does the DoD have to the source code?
- How is testing done? What tool suites are used? How much is automated? How long does it take to do a full regression test?
- How is cybersecurity testing done? How are programs/updates certified?
- What does the workforce look like (headcounts, skill sets)? How many programmers? How much software expertise is there in the program office?
- What is the structure of the contract with the government? How are changes, new features, and new ideas integrated into the development process?

Preliminary Observations

- Software is being delivered to the field 2-10X slower than it could be due to outdated requirements, test requirements, and lack of trust in SW
- Many systems are using legacy hardware and outdated architectures that make it much harder to exploit advances in computing and communications

- Program requirements were often formulated 5+ years ago (when the threat environment + available technologies were very different => wasted effort)
- New capabilities and features are added in multi-year (multi-decade?) development “blocks” instead of continuously and iteratively
- Most program offices don’t have enough expertise in modern SW methods
- Most SW teams are attempting to implement DevOps and “agile” approaches, but in most cases the capabilities are still nascent (and hence fragile)
- Transition to DevOps is often hindered by a gov’t support structure focused on technical performance in a waterfall setting (“waterfall with sprints”)
- Information assurance (IA) is complex, difficult, and not yet well architected
- Test, certification and IA are almost always linear “tailgate” processes instead of being integrated into a continuous delivery cycle.

What should be done differently in future programs?

- Spend time upfront getting the architecture right: modular, automated, secure
- Make use of platforms (hardware and software) that continuously evolve at the timescales of the commercial sector (3-5 years between HW/OS updates)
- Start small, be iterative, and build on success – or terminate quickly
- Construct budget to support the full, iterative life cycle of the software
- Adopt a DevOps culture: design, implement, test, deploy, evaluate, repeat
- 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)
- Have a local team of DoD software experts who are capable of modifying or extending the software through source code or API access
- Separate development of mission level software from development of IA-accredited platforms