A U.S. Soldier assigned to 4th Battalion, 27th Field Artillery Regiment, 2nd Armored Brigade Combat Team, 1st Armored Division looks at information on a laptop during Rotation 25-07 at the National Training Center, Fort Irwin, California, April 30, 2025. (U.S. Army photo by Spc. William Vu, Operations Group, National Training Center)

Reflections as a Tanker Developer

By CPT Tate Bowers

Introduction

Trying to integrate programmatic solutions into the operational force is extremely difficult at best, borderline impossible at worst. Poor institutional memory means that projects will be forgotten within a few changes of command, and strong opinions mean that what one commander emphasizes may be useless for any following. There are also the endless problems of funding, resourcing, and providing infrastructure for developers and what they produce regardless of the emphasis behind it. This article is a personal retrospective on dealing with some of these problems as an accidental participant in a live experiment on integrating Cyber Capability Developers (17Ds) into the United States Army Forces Command (FORSCOM).

Background

Following my graduation from the Cyber Network Operator Development Program (CNODP) and entry into the cyber branch as a developer, I joined the cavalry for a second time. I was slotted as Cyber Electromagnetic Activities (CEMA) lead in the Third Cavalry Regiment (3CR). Fortunately, my direct superiors recognized the value in my work as a developer and granted me the freedom to continue developing with the aim of aiding 3CR during my expected tenure of one year, task organized under the S3 section.

I was given the following limitation: "You may do anything you want, with a budget of $0," meaning I would have access to a standard Army image, with a standard Army laptop, using standard Army tools, and without administrator privileges. My superior did not have development or programming experience, none of the people I worked with knew programming, and my position was unlikely to be backfilled, making follow-on support or even a right-seat-left-seat unlikely. Based on operational requirements, I would also need to spend time assisting with the S3.

As far as I can tell, I was almost literally inside a speculative whitepaper titled "Cyber Support to Corps and Below." With a set timeframe and scarce specialized resources, I was tasked with making things "better."

Situation

My experience as a tank platoon leader over the past two years allowed me to pinpoint areas where automation could be most effectively applied. I suggested applying Z-scores to analyze the data, aiming to identify units

that might be misreporting their vehicles' uptime and to pinpoint specific vehicles being overlooked relative to others in the same population. My proposal was met with confusion, and I was instead encouraged to use Power BI and given broad freedom to proceed.

While Power BI is extremely powerful, it frequently does not play nicely with Army outputs. For example, weapons data is compiled in the Defense Talent Management System (DTMS) as many small tables in the same spreadsheet, separated by multiple lines. Power BI accurately views each of these as a separate data table, but that is supremely unhelpful when trying to generate analyses across all of the tables. If I was going to generate anything, I needed to preprocess the data first.

As a result, I pitched a plan to automate generating the commander's dashboard, a collection of weapons, physical training, medical, and other training data that was compiled and generated on a per-troop basis. Given the issues in the weapons data, I would need to reorganize the data anyway, so it was mostly downloading and manipulating data already present in DTMS. If I did one, I'd have the process mostly down for all of them.

The question became: What is the best way for me to manipulate the data and produce a useful output? I decided to try to focus on a few qualities to help decide the approach.

1. Portability. No downloading outside tools or systems. If IT or security needs to get involved, there's going to be a commander who says, "No" just because it's easier than saying, "Yes," and the spin-up process for 3CR was already tenuous enough.
2. Longevity. Don't rely on systems that are likely to change. The advantage of DTMS was that the data was already present, and I could do my best to make my system data format independent. I preferred not to use 3rd party integrations or newer Army systems. I fully admit that this is the ultimate in the "Not Built Here" philosophy. However, given that I didn't have a development organization or any coworkers who were programmers, I suspected that bullheadedness would kill the project the moment I left if anything broke or appeared to break.
3. Ease-of-Use. Limit the number of clicks to make the system work. I wanted to avoid installations, certificates, and anything else that required even acknowledging a pop-up. It had to be obvious that the system was easier than any alternative, including telling a private to do it by hand.

As a result of one, most real programming languages were immediately ruled out. Trying to get the option to compile code into a functional executable would have taken far longer than I likely had.

Tool-wise, I needed to work with systems that could interface easily with Excel, since many Army systems refuse to output in Comma Separated Value (.csv) format, hampering raw data manipulation. This narrowed it down to two designs: a very complex batch or PowerShell script, or a Visual Basic for Applications (VBA) script. I considered that IT would potentially block PowerShell but probably not macros or VBA, so I had my approach selected.

Within a couple of weeks, I'd managed to complete the project to the point where the weapons data was being output in a dense grid—ideal for Power BI processing—then extracted and output as a readable table. Then I moved onto the next data source. And repeat.

The final script processed, manipulated, and did basic statistics on five of the six data sources, then ranked the troops in order. The most surprising aspect of this code was that nearly half of the VBA script was dedicated to outputting the data as the commander desired, including a specified shape format, color scheme, bolding, naming convention, and merged cells. By the end, I'd validated the data output with each squadron commander, created a Power BI-friendly and labeled table, and generated a colorful data display that mirrored the existing commander's dashboard down to the troop level. I reduced the process's complexity to the point that after setting the save location, a user required only four keystrokes to generate the desired output.

In the end, most of the time was spent generating documentation, getting feedback, getting buy in from the commanders, and trying to simplify the process. In the end, it simplified to copy the script, paste the script into the VBA macro window, and press run. That said, compromises were made—hopefully from a programmer's perspective rather than the users':

1. Everything is contained in one file, a well-known anti-pattern. This is useful in this specific instance because it reduces the number of clicks for users to interact with the system, but it also means that complexity is severely limited. It also means that reusability and code sharing with more complex systems requires manual intervention.
2. I tried to reduce the impact of changing unit names across systems with multiple variants of names. There are hundreds of constants declared in lookup tables to try to reduce the risk. The code searches for how everything is organized through several levels to minimize the effects of simple changes.
3. It's still a VBA script. I would have preferred to use a different language, but it seemed like the most feasible solution. Copy and pasting a script into a VBA window is gross from a "proper implementation" perspective, but … it works. And it's something that people can understand, even with little knowledge about computers. The required knowledge level and Excel experience is minimal for the effect. Future modifications are not something I envy someone else doing, but I tried to document everything thoroughly and highlighted parts I was not proud of for whatever reason.
4. It contains no imports. Enough said.

Results and Lessons Learned

In the three months since leaving that position, my project has continued to be used, even enduring a change

of command and near complete turnover of the S3 shop. I am still waiting for the long-term sustainment of that adoption, but current results are promising. Trying to understand more, I think the following reasons are why the automated dashboard has a higher chance of long-term adoption than expected:

1. To maintain the same standard without my tool, Soldiers down to the troop level would need access to DTMS to compile the same amount of data. While orderly rooms could certainly be tasked with it, that task would be an irritant for approximately 40 administrative troops (and their commanders) that had been previously done by one expendable Captain in the S3.
2. Medical data, which only the Senior Enlisted Leader (SEL) was authorized to handle, was integrated. I generated a second script that produces anonymized data that aided in providing reports, for which the SEL was very enthusiastic as they cannot easily delegate the responsibility.
3. It nests well with higher-command requirements. The previous method (relying on subordinate units to self-report data) presented inaccurate reports that were overly optimistic. Personnel in the 1st Cavalary Division were starting to leverage DTMS for metrics and promotions more than it had been previously used. As a result, leadership was more directly affected by DTMS and self-reporting being out of sync.
4. The system is easy to use, and it is hard to mangle. VBA is not opened by default, so people need to generate local copies rather than editing the master copy directly. This is certainly not guaranteed long term, but it is a useful consideration, and it makes it less likely for unrecoverable problems to occur.
5. Everyone knows Excel, and it's not going to disappear any time soon. There is no feeling that it is its "own" special thing.
6. It's extremely low latency from the user's perspective, and it can feel as fast or faster than copy-paste (thanks to Army computing resources). Due to a requirement to generate the dashboard for each troop, the person running the script sees it constantly working rather than just a loading bar or freezing.

From an experimental perspective, do I think that the last year was successful? To a degree, yes, I do. I produced a tool that was integrated well into the unit and alleviated a tedious portion of the job from the S3 while increasing transparency and accuracy. Do I think it was the best use of my time? Probably not. While the actual process of adoption was the hardest and longest part, I basically spent a year playing with VBA data processing rather than contributing to projects in support of cyber operations that could better utilize my skills. I am uniquely qualified to interface with a cavalry unit as a 17D, but I still had a hard time communicating with and understanding the unit's priorities. "Do whatever you want" frequently means "Do something we want but we can't define what we want because we don't know."

In my opinion, the hardest thing about this was getting the Army to change their processes. The Army has a problem with adopting new solutions to old problems, especially when functional solutions exist, regardless of how slow they are. To a small degree, VBA, in the current environment, helps alleviate this. Everyone knows Excel; VBA is just one more thing it can do. If Python was everywhere—and people had to use it—people would feel the same way. Fundamentally, people don't necessarily like learning new things (especially administrative personnel in FORSCOM), and maintaining and advertising capabilities to the wider Army is extremely difficult. People will spend three hours hand editing documents to avoid spending 10 minutes navigating the deeper Word settings pages. The solution needs to be so simple from the user's perspective that they don't have an excuse to go the harder route. "Living off the land" is not just a term for Offensive Cyber Operations; it's also a tedious, relatively slow, and limited collaboration approach towards software development. Yet, the institutional adoption process for my tool was even slower than this development philosophy.

From a higher-level perspective, this may argue against some of the Army's attempts to put development organizations in contact with big Army units. Despite the limitations that I experienced, I was able to generate multiple iterative systems to assist with processing various types of data. However, a remarkably simple and self-evidently useful solution proved most effective and was widely adopted. It took time to build trust that the system would be sustainable and wouldn't break. If a more distant organization, like the Army Software Factory, were to take on a project like this, I think they would struggle even more to gain traction. Commanders have very strong opinions on format, and that can often take longer to properly build than the data itself. They also often have a strong preference for things built in their own organization. Everywhere has its own standard operating procedures (SOPs) for a reason, even if those procedures are functionally identical to previously published documents. Trying to get an organization to use a process or tool that is made by someone else, someone not trusted and known and just seen by email, is a significant lift. While data processing itself is not that hard, it is extremely difficult to establish trust and utility in that data and requires navigating complex formal and informal systems.

Capt. Tate Bowers graduated from West Point in 2018 with a degree in computer science, commissioning as an Armor Officer. He served for two years as a tank platoon leader in 1-1CAV before being transferred to the cyber branch as a Cyber Warfare Officer (17A). After completing the Basic Skill Level Exam to become a 17D, he served as a Cyber Electromagnetic Warfare Officer in 3CR for a year before attending Cyber Captain's Career Course.

Sources:

Bowers, T. M. (2025). 3CR automations. R2D2. https://code.levelup.cce.af.mil/tate.m.bowers.mil/3cr_automations