NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

CAPSTONE APPLIED PROJECT REPORT

ORGANIC SOFTWARE DEVELOPMENT: A CASE STUDY FOR AGILE DEVELOPMENT

March 2023

By: **Daniel Kern**

Advisor: **Charles K. Pickar**
Co-Advisor: **Jeffrey R. Dunlap**

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** March 2023	**3. REPORT TYPE AND DATES COVERED** Capstone Applied Project Report

4. TITLE AND SUBTITLE ORGANIC SOFTWARE DEVELOPMENT: A CASE STUDY FOR AGILE DEVELOPMENT	**5. FUNDING NUMBERS**
6. AUTHOR(S) Daniel Kern	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

This project examines the efforts and results of United States Strategic Command's (USSTRATCOM) Targeting Process Improvement Working Group (TPIWG) as a case study. The TPIWG conducted an evaluation of USSTRATCOM J52/J2T Joint Targeting Division's current target maintenance processes and identified areas of improvements. After identification of areas requiring improvement, the TPIWG conceived the idea for and began design and development of an automated change detection software program for target maintenance.

Methodology includes an analysis of what processes were identified for improvement and how change was implemented, how well software development processes were implemented and adhered to, analysis of the process to secure funding support from the U.S. Air Force and subsequent contracting for full time support of the software.

This case study documents challenges, innovative ideas, risks taken and faced by the TPIWG during the course of process analysis, software development and implementation phases. Documentation of challenges to sustainment funding from a cost, schedule, and performance perspective. Exploration of pros and cons to having the completed work then contracted to a secondary party, not part of the TPIWG. Identification of future challenges contractors may face in the sustainment or future improvements of the software. Lastly, this case study tries to determine what the overall improvement and benefit is for the warfighter.

14. SUBJECT TERMS target, target maintenance process, software development, automated change detection, United States Strategic Command, USSTRATCOM, Targeting Process Improvement Working Group, TPIWG	**15. NUMBER OF PAGES** 63
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

ORGANIC SOFTWARE DEVELOPMENT: A CASE STUDY FOR AGILE DEVELOPMENT

Daniel Kern, Lieutenant Commander, United States Navy

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN PROGRAM MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
March 2023

Approved by: Charles K. Pickar
Advisor

Jeffrey R. Dunlap
Co-Advisor

Robert F. Mortlock
Academic Associate
Department of Defense Management

THIS PAGE INTENTIONALLY LEFT BLANK

ORGANIC SOFTWARE DEVELOPMENT: A CASE STUDY FOR AGILE DEVELOPMENT

ABSTRACT

This project examines the efforts and results of United States Strategic Command's (USSTRATCOM) Targeting Process Improvement Working Group (TPIWG) as a case study. The TPIWG conducted an evaluation of USSTRATCOM J52/J2T Joint Targeting Division's current target maintenance processes and identified areas of improvements. After identification of areas requiring improvement, the TPIWG conceived the idea for and began design and development of an automated change detection software program for target maintenance.

Methodology includes an analysis of what processes were identified for improvement and how change was implemented, how well software development processes were implemented and adhered to, analysis of the process to secure funding support from the U.S. Air Force and subsequent contracting for full time support of the software.

This case study documents challenges, innovative ideas, risks taken and faced by the TPIWG during the course of process analysis, software development and implementation phases. Documentation of challenges to sustainment funding from a cost, schedule, and performance perspective. Exploration of pros and cons to having the completed work then contracted to a secondary party, not part of the TPIWG. Identification of future challenges contractors may face in the sustainment or future improvements of the software. Lastly, this case study tries to determine what the overall improvement and benefit is for the warfighter.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

AAF	Adaptive Acquisition Framework
AFLCMC	Air Force Life Cycle Management Center
CJCSI	Chairman of the Joint Chiefs Instruction
COTS	Commercial off the Shelf
DevSecOps	Development, Security and Operations
DIA	Defense Intelligence Agency
DIB	Defense Innovation Board
DOD	Department of Defense
ETF	Electronic Target Folder
GAO	Government Accountability Office
JEMA	Joint Enterprise Modeling & Analytics
JP	Joint Publication
JWICS	Joint Worldwide Intelligence Communications System
MIDB	Modern Integrated Database
MVP	Minimum Viable Product
NDAA	National Defense Authorization Act
NGA	National Geospatial-Intelligence Agency
RFI	Request for Information
RMT	Record Message Traffic
RESPROD	Responsible Producer
SABER	Simplified Acquisition of Base Engineer Requirements
USSTRATCOM	United States Strategic Command
TPIWG	Target Process Improvement Working Group
USA	United States Army
USAF	United States Air Force
USN	United States Navy

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

This research was designed to examine the application of the agile software development process and the benefits it provides to software acquisition. The research examines a real-world application to inhouse development of a software solution at USSTRATCOM J52 for the purpose of automating and enhancing the quality of target maintenance. The research explores the concept of a software factory and how they can add cost and time saving measures to development of software.

This case study identifies mechanisms within the DOD that can be applied to enhance a combatant command's ability to develop software tools, enhancing a combatant command's ability to field new software in a rapid manner to keep pace with changing mission needs and adversary advancements.

The scope of this study is limited to the TPIWG experience, application of concepts, and end results being currently used. The TPIWG experience is being shared first hand as well as supporting data that is available at the UNCLASSIFIED level. Primary resources being used are *United States Government Accountability Office* publications.

Lessons learned from the case study are looked at retrospectively, providing justification for alternate pathways. The conclusion of the case study presents a case for software development opportunity and sustainment from specialists.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

This project would not have been possible without the support and assistance of my advisors, Dr. Charles Pickar and Mr. Jeffrey Dunlap, CAPT USN Ret. I would like to thank my Division Chief, Colonel Jeffrey Dyball, USAF, for his faith in my leadership and trust in innovative ideas to champion a new software concept, aiding in rapid target maintenance. It was a tremendous honor to work with the TPIWG team, specifically LCDR Michael Moore, USN, and MAJ Ryan Pretty, USA, and the many individuals who dedicated countless hours to rework software coding and championed acceptance and funding to create a program of record. Finally, I would like to thank my strongest supporters, my family, who attended many Little League practices and took trips to the zoo, allowing me to dedicate time to this body of work.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Intelligence is the ability to adapt to change.

—Professor Steven Hawking

The speed of technological development dramatically increased throughout the twentieth century, mostly in the realm of physical hardware, and militaries innovated in ways never before seen. Now over twenty years into the Twenty First Century, there is a new realm included in military development, the digital realm. Since the advent of computers, more information has been available and created, allowing militaries to know more but also process larger quantities of information at faster rates.

In the current environment there is often too much data available and humans are not always quick enough to find, analyze, and apply the information in a relevant timeframe. There are countless databases in which information is stored, all maintained by different agencies, with access limited to specific user groups, data is not always extractable, data formatting is not uniform, software systems are slow to be improved, and the creation of new software is not quick enough to replace outdated legacy systems.

The Department of Defense is working to share more information among agencies, improve its software development, and streamline processes. Services have developed software factories to answer the needs of individual units, combatant commands, and component commands, to access more information and process larger quantities of data faster.

A. BACKGROUND

1. Target Process Improvement Working Group (TPIWG)

The United States Department of Defense uses JP 3-60 Joint Targeting to outline how the department shall conduct targeting for warfare. CJCSI 3370.01C Target Development Standards outlines how targets should be developed and what information is doctrinally required within a target folder. Once a target has been developed, it remains mostly stagnant until it is nominated for strike or possibly looked at years later. Target

maintenance is a process that is not required by the Joint Staff, but it is conducted at USSTRATCOM. The purpose of target maintenance in the past has been to revisit a target on a periodic basis and repeat the target development process to review intelligence concerning the target in the years since the record was created or last reviewed. The target maintenance process was time consuming, cumbersome, and extremely duplicative.

On 1 August 2021, USSTRATCOM J52 issued an order to conduct a review of the target maintenance process. The Target Improvement Working Group (TPIWG) was created and given the responsibility to review the current process, make recommendations for changes in the process, streamline procedures wherever possible, and to automate where feasible, to reduce personnel work hours spent on maintenance without lowering work quality.

2. Old Maintenance Process

In the past, USSTRATCOM target maintenance was conducted on a periodic basis as part of a fixed schedule. The average target would be revisited every two years, manually reviewing all available intelligence material to determine if the target retained the same function as it was originally nominated for. The results of this in-depth review would then be annotated in the corresponding Electronic Training Folder (ETF) maintained in MIDB.

The maintenance process provided a guaranteed periodic review of all targets, allowing the command to have confidence in the viability of all targets. Targets that no longer met the viability criteria were removed and no longer maintained.

The legacy maintenance process lacked timeliness, negatively impacting the efficiency of the overall targeting process. The two-year ETF review timeline often meant targets were 12 or more months out of date when the review process renewed. The speed at which new intelligence was being reviewed at USSTRATCOM was not conducted at the speed of relevance. Often analysts would open an ETF for review and would find RESPROD made updates months prior, changing important data, which in turn would drive changes to targeting strategy and additional operational changes.

Additionally, the Strategic Targeting Division lacked an in-depth understanding of all metrics involved in the maintenance process to update an ETF. An excel spreadsheet was maintained attempting to track data on individual targets, but was too large, complex, hard to read, and data was undecipherable. Many questions and data points were left unanswered. How long did it actually take an analyst to conduct maintenance? What was the source of the change? How often were there changes to targets? How much analytical rigor was required? Did everything need to be reviewed for maintenance? How often did targets need to be deleted? Are collections correctly focused based on types and/or sources of changes? Leadership did not have the answers to these questions and steps needed to be taken to develop a mechanism to capture and track the abundance of data that was needed.

3. Process Review

Immediately after the TPIWG was announced, the group convened to discuss the way ahead. The TPIWG was designated to be directed by a single active duty targeting officer. Key individuals within the division were designated as primary support to the process but all hands were available to the effort. The group established a list of problems which needed to be solved with the maintenance process**.** Potential ideas were listed for each issue to allow for group discussion and reflection to occur. All personnel were assigned to areas of their resident expertise or areas of significant training. The group was fortunate to have two individuals with extensive coding experience and their recommendation for an automated change detection database concept was agreed upon by the group.

The process review was a period of 30 days during the month of August 2021. This time period consisted of problem framing, solution research, planning, and drafting proposed implementation. This process was the sole focus of TPIWG participants, rapidly planning the solution due to the fact all individuals were targeting analysts and familiar with the problem at hand. An agile program management style was adopted to allow all members to have equal participation and to aid in conflict management as disagreements would occur often.

4. New Process Recommendation

With the TPIWG's desire to move forward with an automated database used to track intelligence updates and for analysts to add remarks based on data ingested, a new analytical framework was designed to focus analytical rigor. This analytical workflow took into account DIA analytic and tradecraft standards, which helped shape the end result, as seen in Figure 1.

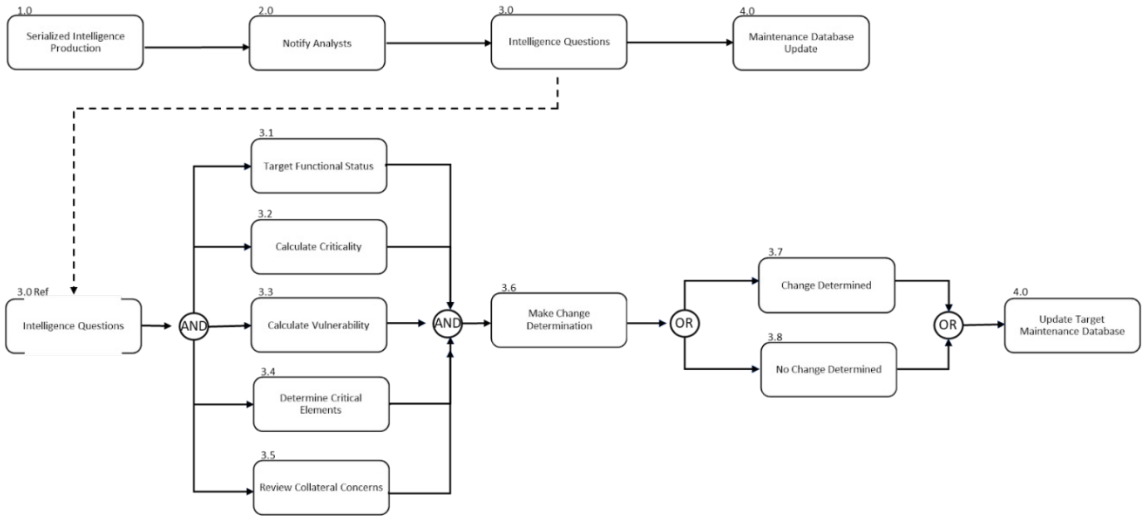

Figure 1. Target Maintenance Functional Flow Block Diagram

Figure 1 shows the final analytical thought design within a simplified version of the change detection database. Five analytic questions must be answered when a change is detected in the database:

- Functional Status – Was there a change to the core function of a target?
- Calculate Criticality – Was there a change to the criticality of the target within the target system?
- Calculate Vulnerability – Was there a change to the vulnerability of the target?

- Determine Critical Elements – Was there a change, addition, or removal to any of the critical elements of the target?
- Review Collateral Concerns – Was there a change, addition, or removal to any of the collateral concerns of the target?

The previously identified questions helped reduce the amount of data an analyst would have to look at, by information the analyst of new intelligence reporting though the change detection database, then focusing intelligence analysis in five critical areas. Analysts experienced as end users of the targeting software advised through the problem-solving stage prior to the project receiving approval by the Strategic Targeting Division leadership.

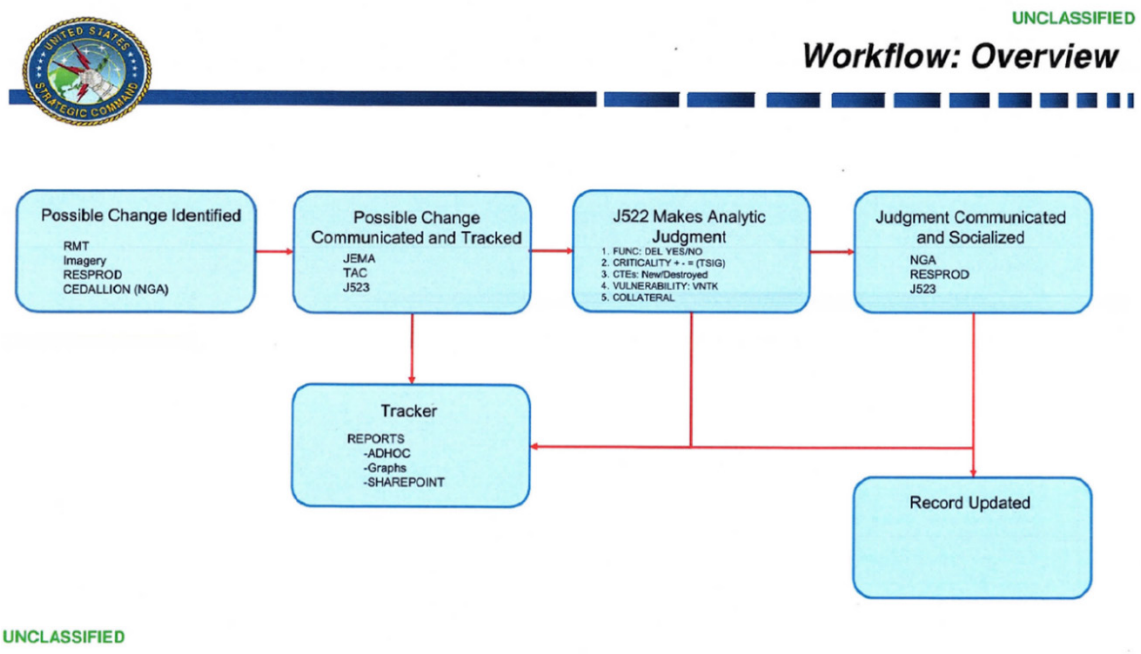

Figure 2. Target Maintenance Workflow Overview Diagram

The analysis and database workflow were combined, as seen in Figure 2, identifying the different touch point expected to occur. Figures 3 and 4 further refined the expected analysis and database workflows, indicating information flow automated by the change detection database and manual input required by the analyst.

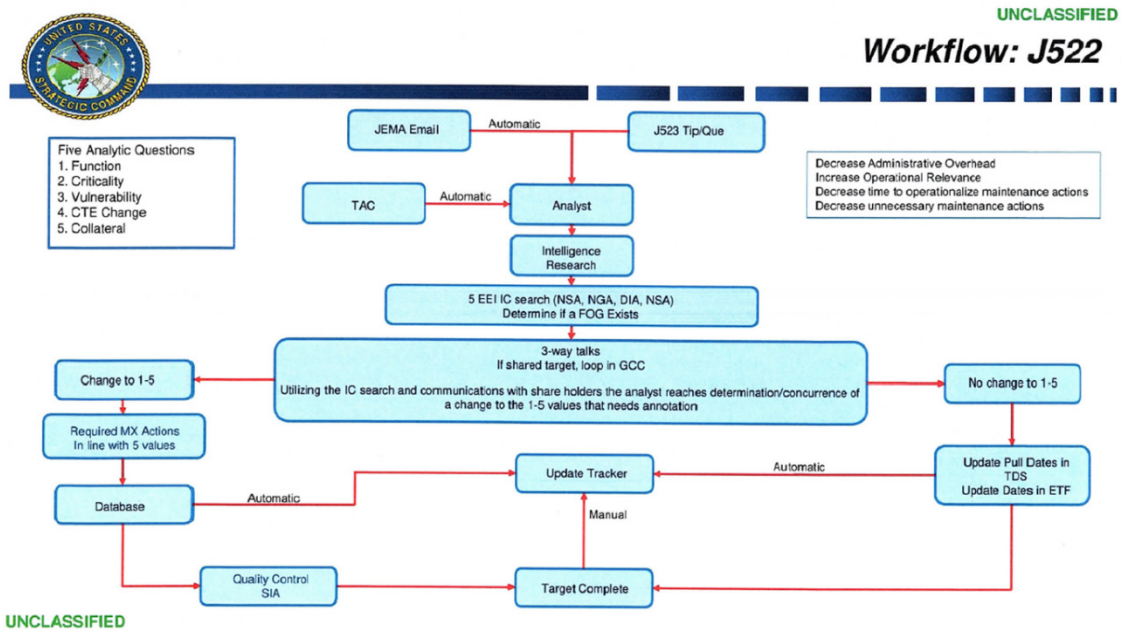

Figure 3. Target Maintenance Workflow Diagram J522

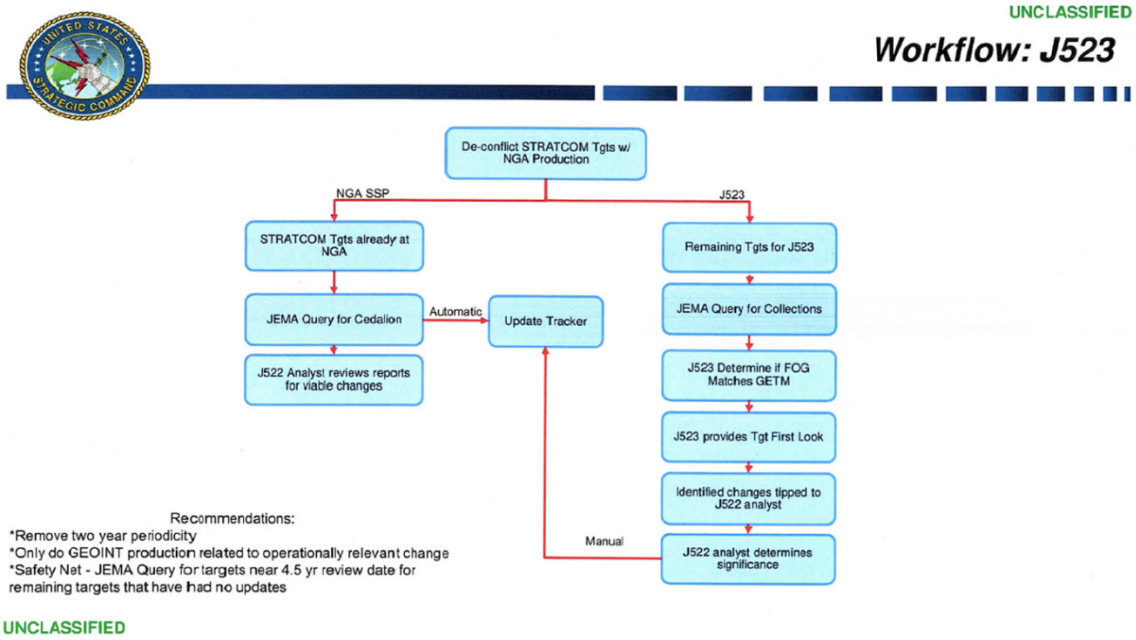

Figure 4. Target Maintenance Workflow Diagram J523

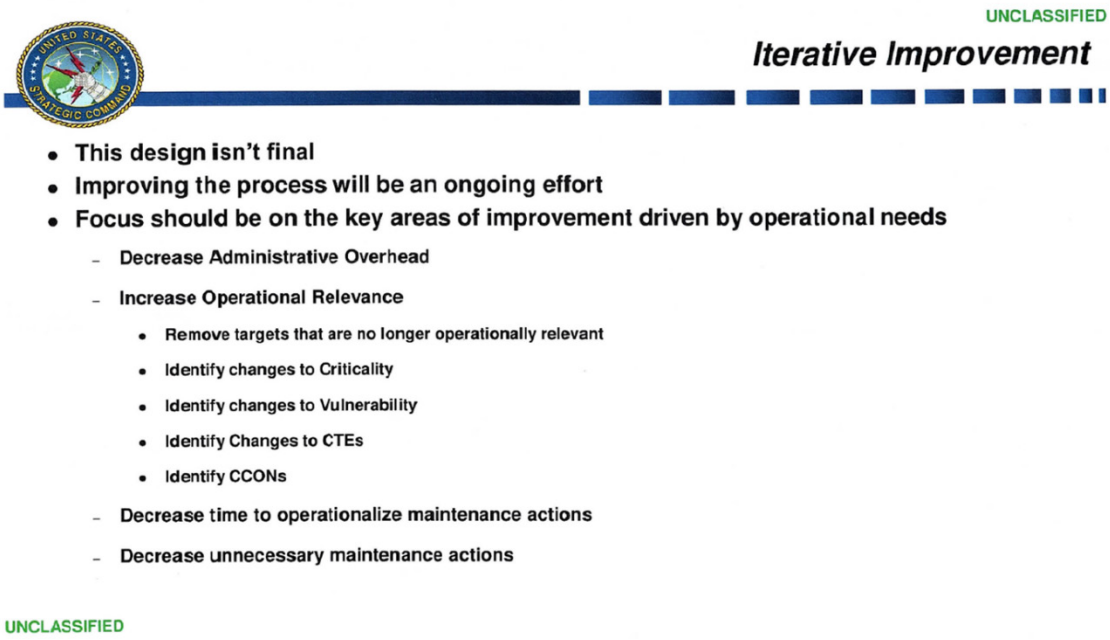

Figure 5. Target Maintenance Iterative Improvement

Figure 5 depicts the TPIWG agile process pitch though iterative software development, continuous data collection in a learning environment and the end goals the group was trying to achieve.

5. Summary

The aforementioned background provides important context for the case study that will be presented in this report. Software development is incredibly difficult and complex by any means. The description of how process solutions and software development was completed in-house by intelligence analysts to support the creation of an automated tool for target maintenance tracking and data collection was provided as a practical example for the scope and application of this report.

B. PROBLEM STATEMENT

The DOD, as a hierarchical organization, often has a top-down directive approach to problem solving and process implementation. The DOD is slow to adapt to innovation, especially when it comes to new problem-solving methods geared towards empowering individuals or small groups to carry out large tasks. Software in particular can be very

complex, depending on what it is used for, or solutions it is created for. Rapid creation and testing of new software and applications needs to become a normal activity to keep pace with an evolving environment.

C. RESEARCH OBJECTIVES

The objective of this case study is to explore the Agile software development cycle and software factory model, highlighting the benefits it provides to software development and acquisition, delivering better capabilities to the warfighter faster. While the Agile process and software factory concepts are not new, there remains significant room within the DOD for additional and more widespread use.

D. RESEARCH QUESTIONS

Primary Research Question: The primary question that will be answered in this case study is, "How was the Agile Development Cycle applied and what were the benefits it provided to the software development process?"

Secondary Research Question: The secondary question outlined is, "How can a software factory construct or mindset provide more relevant, faster, and complex software to a combatant command mission set?"

E. PURPOSE/BENEFIT

The purpose of this case study is to provide a real-world example where processes and newer concepts were applied to develop an automated tool with practical application for the warfighter by using the agile software development process. Software acquisition can be time consuming, complex, deal in the realm of proprietary coding, and lack the immediate benefit for the end user or warfighter. This case study will identify mechanisms within the DOD that can be applied to enhance a combatant command's ability to develop software tools, enhancing a combatant command's ability to field new software in a rapid manner to keep pace with changing mission needs and adversary advancements.

F. SCOPE/METHODOLOGY

The scope of this study is limited to the TPIWG experience, application of concepts, and end results being currently used. The TPIWG experience is being shared first hand as well as supporting data that is available at the UNCLASSIFIED level. Primary resources being used are *United States Government Accountability Office* publications.

G. THESIS STATEMENT

Combatant Commands should make use of agile software development for smaller software applications, improving on existing software, or utilize software factory capabilities to develop larger software needs to reduce the risk inherent of in-house software development.

H. SUMMARY

This chapter outlined how the TPIWG process played out in solving the target maintenance process at USSTRATCOM. Group approaches were explained in addition to expected outcomes. The background previously laid out will give the reader the background to understand what processes or tools the TPIWG may or may not have applied or how well those processes worked in the rest of this case study.

THIS PAGE INTENTIONALLY LEFT BLANK

II. AGILE SOFTWARE DEVELOPMENT

If everyone is moving forward together, then success takes care of itself.

—Henry Ford

1. What is Agile Software Development

Agile software development was conceived in the private sector in the early 2000s as a means to develop software solutions quickly in short, but definitive, blocks of time. Software engineers met to try and solve the problem of software project taking too long to complete, often being cancelled due to company requirements changing before a solution completed development. Software engineers met in Snowbird, Utah during the winter of 2001 to discuss possible ways to speed up the development process, completing projects to achieve the requirements they sought to achieve. From this meeting an agile manifesto was born, outlining four items (Nyce, 2017):

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

The engineers wanted to ensure the free flow of ideas and communication to ensure actual needs are met. This is important because when software development teams become focused on process and tools, they can lose sight of what the spirit of the objective at hand. The software engineers focused on working software over comprehensive documentation because overemphasizing documentation will cause a project to grow, adding too many details or lose necessary details in the noise of documentation. The engineers believed it was important to work with the customer, ensuring their needs are being met as opposed to focusing on determining how work will be completed and work billed. Lastly, the engineers want developers to be responsive to the changing needs of the customer and

responsive to what they learn during the development process, becoming for adaptive to changes than sticking to a rigid plan.

According to the Tech Beacon, after the meeting in Snowbird, the agile process thought out by the software engineers began allowing teams to develop a fast delivery approach that enabled users to get some of the business benefits of the new software faster and enabled software teams to get rapid feedback on the software's scope and direction (Varhol, 2022).

Mr. John Adam wrote, "Agile software development is an iterative approach to creating software products based on quickly releasing a minimum viable product (MVP) and then adjusting it and adding features and functionalities in stages based on user behavior and feedback" (Adam, 2022). The purpose of creating a MVP as quickly as possible, is to meet the needs of the customer, establishing a basic framework, then creating a final product with the end user involved in the process. Figure 6 shows the modern agile development process, with each loop representing a new iteration in the development of a software program. Developing an MVP by the end of sprint cycle 1 and improving upon the program with each successive iteration.

Figure 6. Agile Development Cycle. Source: Adam (2022).

Before the agile manifesto, software was created by a development team without much end user feedback and after years of development it would be rolled out, often with many bugs, end user frustration, not meeting original requirements or the requirements changed. Mr. Adam further details as software becomes more complex, it is more difficult to detail the required specifications and create the best option for the end user (Adam, 2022). In essence, building the perfect product in a vacuum never works.

The federal government has been trying to incorporate agile software development as best practices for the acquisition community for many years. In 2012 the GAO published a report titled, *Effective Practices and Federal Challenges in Applying Agile Methods*. This publication was written due to the DOD $76 billion investment in IT projects in FY 2011 and a congressional desire to reduce lengthy IT projects incurring costly overruns (Powner, 2012). The result of the study identified 32 best practices and approaches to apply to agile software development projects (Powner, 2012). The GAO does not set rules but provides guidance and best practices for acquisition professionals and DOD members to follow. The findings in the GAO report were available for anyone to apply to current or new projects that were beginning at the time.

Mr. Pearsons from the Government Accountability Office (GAO) has described Agile software development as, "having the potential to save the federal government billions of dollars and significant time, allowing agencies to deliver software more efficiently and effectively for American taxpayers" (Pearsons, 2020). The GAO also describes Agile software development as, "an iterative product development and delivery cycle occurring continuously through a software product's life cycle," and shown in Figure 7. The GAO further adds that software is continuously evaluated through its many iterations for functionality, quality, and customer satisfaction (Pearsons 2020). The agile cycle is applicable to software programs that will have distinct features, of which many, will be discovered through the iterative process (Pearsons, 2020).

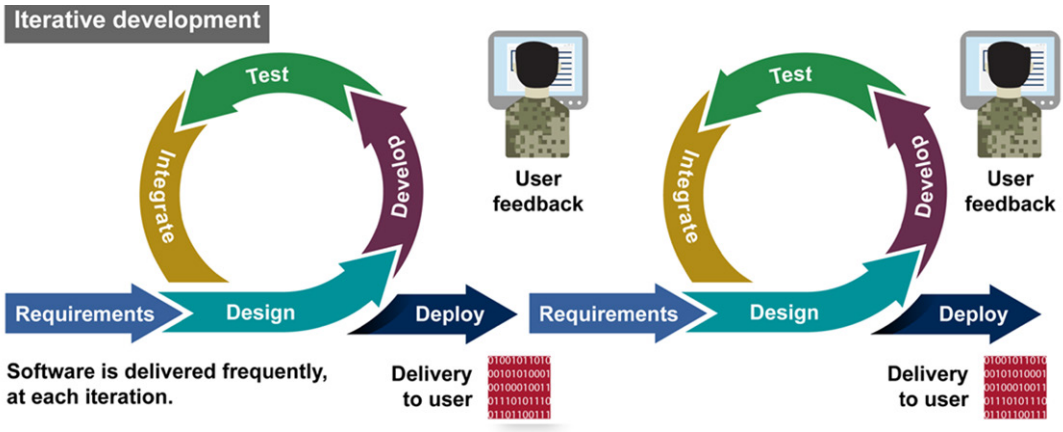

Figure 7. The Department of Defense's Agile Development Process. Source: Pearsons (2020).

The DOD has invested time, effort and money into discovering how agile can become and effective tool for use and best ways to implement in its own projects and also acquisition programs. *The Agile Software Acquisition Guidebook* provides practices and lessons learned from the FY 2018 NDAA, "providing program managers with information on developing acquisition strategies for Agile software development and providing an understanding of Agile practices" (Cummings 2020). Agile has proven itself to be a useful process to allow teams to successfully guide themselves in the dynamic environment that is software development. The DOD is taking the time to study, learn and inform its employees of the benefits it offers.

2. DOD Software Acquisition

In January 2020, the DOD established an Adaptive Acquisition Framework (AAF), including software acquisition. As outlined by the Defense Accountability Office (DAO), "the AAF emphasizes several principles that include simplifying acquisition policy, tailoring acquisition approaches, and conducting data-driven analysis" (Oakley, 2021). Further the DAO states, "the AAF comprises six acquisition pathways, each tailored for the characteristics and risk profile of the capability being acquired" (Oakley, 2021).

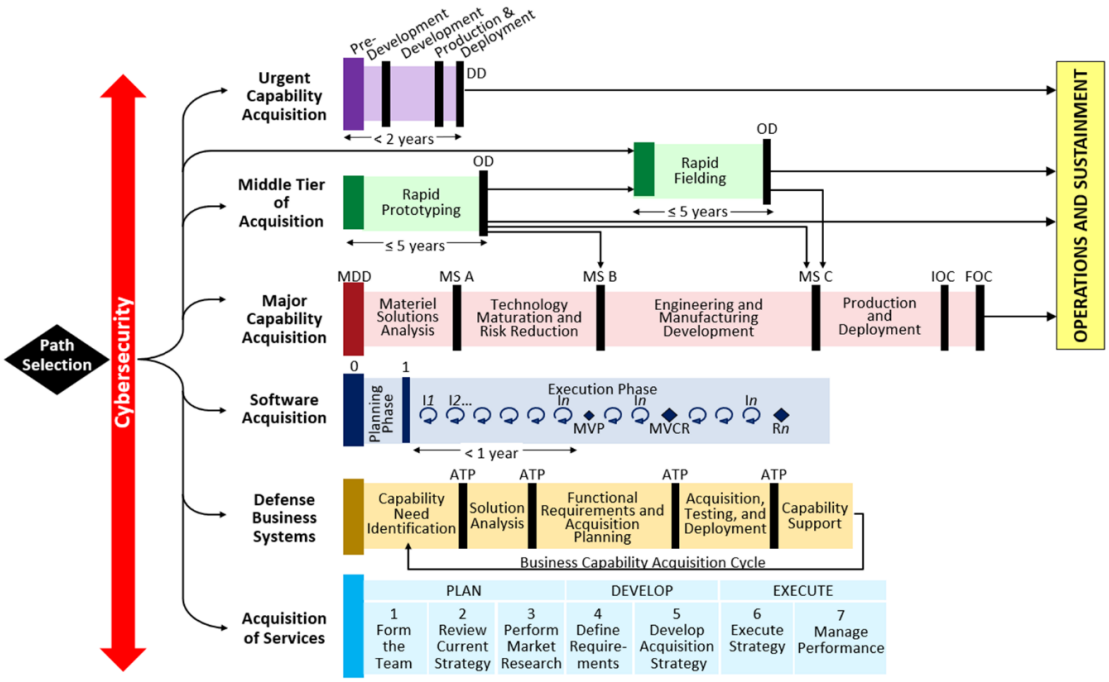

Figure 8. Adaptive Acquisition Framework Pathways. Source: Defense Acquisition University (2022).

Software has become one of the most important components of DOD systems; DOD software development practices have not kept pace with leading industry practices. The Defense Innovation Board (DIB) released a report in May 2019 that emphasized the need for DOD to deploy software quickly and develop a workforce to follow modern software development practices (Oakley, 2021).

The DOD has made agile software development a significant pathway option for the acquisition process. It has codified and standardized a specific pathway for software acquisition on the principles of agile development. This is a declaration by the DOD for agile to become the development process of choice when it comes to developing or acquiring software.

3. How was the Agile Concept Applied to the TPIWG

Agile was first applied to the TPIWG from the onset, as the group began to self-organize and set up a framework to quickly frame the problem and begin identifying solutions for the group to work toward. As outlined in the GAO Agile Assessment Guide,

agile teams should be self-organizing, meaning they are empowered to collectively own the entire product and decide how work will be accomplished (Pearsons 2020). An agile team's duties should be well defined, and authorities should highlight the importance of cross-functionality to allow for autonomy and team stability. The more encouragement and latitude the team is given, the better it can address technical issues in creative ways. If teams are not self-organizing or self-managing, the teams may be inefficient, causing program cost increases and schedule slips (Pearsons, 2020).

The TPIWG members had limited coding experience and chose to leverage existing enterprise services as much as possible to achieve an automated software solution as quickly as possible (Defense Acquisition, 2022). TPIWG personnel identified how a system architecture would be created, gathering data from established intelligence databases through an existing program, Joint Enterprise Modeling & Analytics (JEMA), to use as an intermediary between data sources and a database created in Microsoft Access (Defense Innovation Board 2022). JEMA is a platform that supports the ingestion of virtually any data source and normalizes data into a common data format, allowing analysts to easily build complex analytic models with disparate data sources (NASK, 2022).

As in line with the fourth agile core principle of responding to change, learning occurred during the TPIWG process as analysts were trained locally to use JEMA in the workspace. JEMA allows analysts to easily extract information from established databases or data residing in excel spreadsheets, and provides the ability to manipulate the data using the functions offered. The use of the JEMA tool allowed the developers to change on the fly as they better understood the capabilities of the tool and how it could be best utilized in supporting the change detection database.

On September 1, 2021, the TPIWG adopted an agile process, creating a weekly cycle for software development and testing, with each week representing a single iteration as shown in figures 6 and 7. Figure 9 showcases the activities scheduled during September 2021, with each week ending with an end user information session, detailing progress and allowing the user to use the program and provide feedback.

4. Version I Development

The first version of the change detection database began 1 September 2021, continuing through the month on a weekly agile cycle. As shown in Figure 9, the September schedule shows concurrent processes that were ongoing during the software development phase. Throughout this period, development, continual refinement, testing and feedback occurred. Analysts from within the division, who were the end users of the product, provided feedback at the end of each weekly cycle. The TPIWG likely had an over abundance of user input into the development, as this was developed by users to begin with.

Figure 9. TPIWG Schedule September 2021

Through development, feedback and training more was learned as the group developed the change detection database. Mr. Pearsons wrote in the GAO publication Agile Software Development, "agile software development is well suited for programs where the end goal is known, but specific details about their implementation may be refined along the way" (Pearsons, 2020). The TPIWG develop a program with an end goal in mind but a substantial amount of learning on how to structure the program and code for the functionality occurred along the way. The group learned how different database

information interacted with each other, critical information needed by the end user, and how best to write code to link formation. The TPIWG demonstrated the ability to respond to change and customer collaboration, two of the four agile manifesto core values. During the month of September 2021, as testing occurred, a soft rollout was provided for controlled live operational testing and user interface feedback. Figure 10 depicts the October 2021 scheduled, showcasing the database soft opening after only six weeks of development.

As the software went live in October 2021, the development team began working on creating reports to show metrics (Figure 11) to help leadership understand target maintenance data the database was designed to capture. This was a requirement levied by leadership but could not be completed until after the change detection database was developed. Metrics were actually developed from the learning process which occurred during the agile software development process. The TPIWG identified what information was readily apparent and discoverable, but also during each weekly agile cycle, the TPIWG ask what data could become discoverable as development continued.

The weekly agile cycle consisting of software updates, feedback and training continued until mid-November 2021. The development cycles at this point were refining the user interface, button functionality and bug fixes. At the end of November 2021, it was decided to transition the change detection database from a development phase to a sustainment phase, occurring by December 2021 (Figure 11). Analysts felt comfortable enough using the database and maintenance was occurring at a more expedient pace.

Figure 10. TPIWG Schedule October 2021

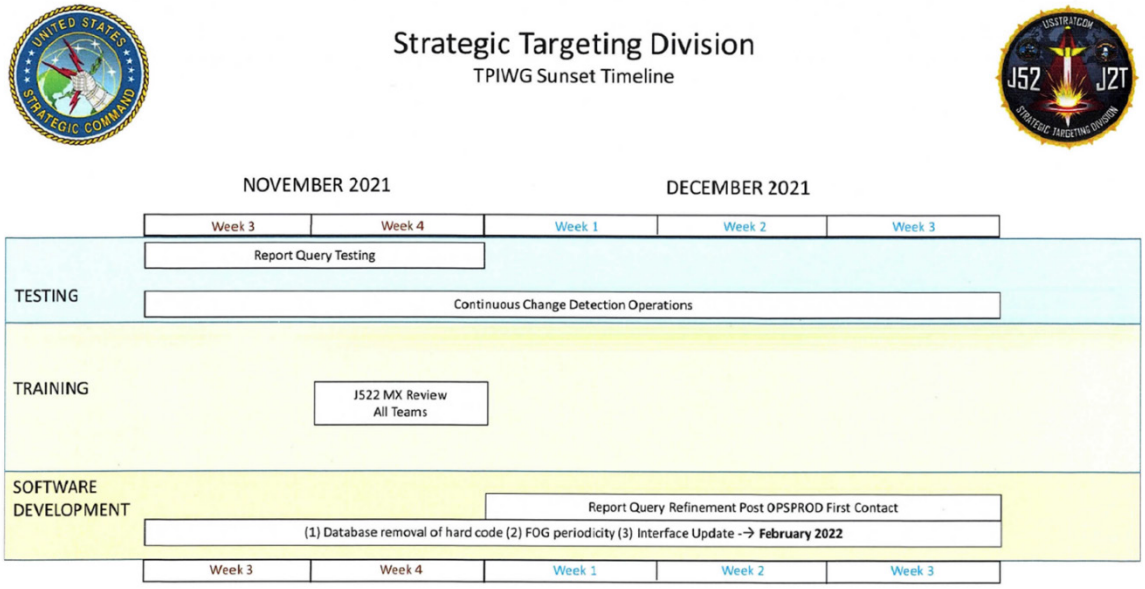

Figure 11. TPIWG Sunset Schedule 2021

The TPIWG schedules depict concurrent lines of effort as each weekly iterative cycle occurred, culminating in a MVP being released live after 45 days of development. The database then entered an operational phase, where the development cycles became longer but now focused on providing improved capabilities to the end user. The TPIWG

did not develop a system frame work or system documentation from the onset. The goal was to use an agile process to reduce the amount of time the division was in a stand down from conducting target maintenance. The agile process allowed the TPIWG to work as the pace it did and adapt to change from user feedback and information discovery along the way. Another positive to the agile process was that users were very familiar with the operation of the database before it went live. One the database was live users could use it as intended then provide operational use feedback to help improve the operational functionality.

5. Why Was Agile Chosen

The TPIWG chose to approach the creation of a change detection database with the agile process because of the flexibility it offered. The group had a very basic idea of how to accomplish what was needed and what it should look like, but the group did not know exactly what right would be. Agile offered the TPIWG the ability to create a MVP quickly but then to improve upon the MVP in an iterative fashion and add in emerging requirements as they became apparent during the development of the change detection database.

As mentioned, each week the database was being developed, a full agile cycle would occur. During each cycle, the TPIWG would learn from the challenges of coding Microsoft Access to carry out the functions needed. JEMA coding would need to be changed depending how the databases interacted or effects from new functions added from a new cycle. Each cycle provided an environment to learn from errors and make changes to only a small set of coding, instead of waiting to the end of a project, finding out it didn't work, then trying to discover where the errors were. It is easier to look through a smaller pool of work for issues than a year's worth of work.

The TPIWG was also intently focused on making a product that the end users were comfortable using and had a hand in developing. It was important to have end users be involved with the process at every possible step, offering their opinions on what the database should do, how it should process and display information. The TPIWG was shaking up an entrenched target maintenance process, developing a new software tool to help automate new information, maintain targets, track status and metrics. User feedback

provided insight into what the database should do and the front end of the software was ultimately designed using feedback.

Developing the back end of the database was derived from working groups with the J56 and a lot of trial and error. It went quickly, providing the TPIWG with valuable insight into how intelligence community databases worked but also provided the J56 with situations and work they had never tried.

There were many deliverables that needed to occur during the development process, and each TPIWG member had roles and responsibilities. To manage all of the deliverables and keep the team on track, I fulfilled the role of a scrum master. A scrum master is the "role responsible for gluing everything together and ensuring that scrum is being done well, meaning they help the product owner define value, the development team deliver the value, and the scrum team to get to get better" (West, 2022). Scrum is one of many ways to manage agile development, in this case it worked well because of the many aspects the team was working on and the need to coordinate them in concert. An important aspect of scrum is being a transparent leader, communicating any issues the team may be facing. Also in scrum, are agile sprint cycles, a timebox of one month or less during which the team produces a software increment (West, 2022). This also meant leading daily standups to detail activity and expected next day activity, essentially informing all of what was occurring.

6. Development, Security and Operations

DevSecOps is an organizational software engineering culture and practice that aims at unifying software development, security, and operations. The main characteristic of DevSecOps is to automate, monitor, and apply security at all phases of the software life cycle: plan, develop, build, test, release and deliver, deploy, operate, and monitor. The benefits of adopting DevSecOps include reduced time from development to deployment, more robust security, and faster capability at the speed of relevance (Sherman 2022). DevSecOps/Tooling represents the set of capabilities enabling the continuous integration and delivery (Cl/CD) of secure software as produced through a software factory (Sherman 2022).

7. Summary

The TPIWG applied the agile process of developing a software program in an iterative fashion. Development and learning both occurred quickly, adapting feedback into the follow-on cycle and culminated with the rapid release of the change detection database. The DevSecOps process was not utilized by the TPIWG as a COTS Microsoft Access program, residing on JWICS, was used as the basis for the change detection database. Leading the team, meant following a scrum method to keep the project on track.

The TPIWG is a good example of how to apply the agile software development process to rapidly create a software program with end user feedback, reducing the overall development time and meeting end user requirements, while incorporating additional items learned during the process. However, the creation of software by the end user should not occur. Potentially significant risk was unnecessarily taken by J52, pausing target maintenance, to develop a new process, then developing a software solution.

The DOD has centers at the service level for the purpose of rapidly developing and fielding software programs. These centers are called software factories and they have increased in numbers and their application among individual services. Software factories can rapidly develop, incorporate end user feedback, developing in a secure environment and provide support for future updates. Additionally, software factories have a culture of software development knowledge base and creativity within that craft.

III. BENEFIT TO THE WARFIGHTER

In the case of the TPIWG, there were a number of positive benefits to the warfighter, and a few drawbacks associated with implementing new software. In this case, the warfighter is defined as USSTRATCOM decision makers, targeting analysts, and individuals who benefit from the automated change detection database.

1. Benefit #1

The automated change detection process allows new intelligence to be identified, and targeting analysts to receive notifications of the identified information. This benefit allows analysts to be apprised of all relevant target information as serialized information is published on JWICS. The automation adds confidence to decision-makers thanks to the understanding that all new intelligence results in rapid updates, rather than on a two-year review cycle. The resulting process provides a yearly review of all targets, cutting the previous review timeline in half.

2. Benefit #2

The database tracks all changes, allowing for a history of changes detected to be stored and categorized by ETF. By housing all the data in a single location, updated automatically, this benefit reduces the need for cumbersome excel spreadsheets and manual input of information at different times in an ETF's existence. The data is extractable as needed to provide historical context and a timeline of analytical touchpoints.

3. Benefit #3

Automatic indication of RESPROD or other DOD targeting analyst adding information or making data field changes in MIDB. This notification allows a USSTRATCOM analyst to be apprised if a RESPROD is updating an ETF based on another intelligence source that is not currently automated. It also notifies the analyst if another combatant command or component command is adding/editing remarks, for example if the target is being added to another target list. These notifications are

particularly beneficial as the DOD targeting community is attempting to share and coordinate the workload for target development and maintenance.

4. Benefit #4

Analytical rigor is standardized and captured in distinct categories. As mentioned previously in this case study, the five intelligence questions which must be tripped for a 'change' or 'no change' to be assessed, are clearly identified with each analytical touchpoint. In the previous process for target maintenance, analytical rigor and tracking detailed work would vary among the targeting teams and also among individual analysts.

5. Benefit #5

The change detection database provides automated reports from all of the data metrics it is asked to track. The strategic targeting division now has the ability to create reports on demand tracking the status of target maintenance.

Examples of reports include:

- average time per analyst to conduct a 'change' or 'no change' determination
- average time per team to conduct a 'change' or 'no change' determination
- source a change was initiated from – NGA, DIA, RMT, or in house analysts
- identification of 'false changes' by source
- how many targets have been reviewed by fiscal year, month, week, day, analyst, team
- tracks intelligence agency support to USSTRATCOM strategic targeting division

Figure 12. Target Maintenance Workflow Metrics

6. Benefit #6

The combination of the automated change detection database and the five intelligence factors to follow, has provided a significant time savings and accountability metric for analysts. In the first six months of operation, the database metrics reflected an average of 45 minutes spent updating each ETF. Previous average update times took as long as two weeks.

7. Benefit #7

The learning process during the TPIWG agile cycle led to the discovery of the JEMA tool. This tool helped analysts capture and manipulate data from existing databases or retain in excel spreadsheets. The JEMA tool helps reduce the personnel and hours spent answering RFIs for senior leaders and is still in use by the Strategic Targeting Division as of the date of this case study.

8. Drawback #1

Bugs still exist in this new program that continue to require effort to resolve. The system is prone to crashing when too many users access simultaneously

9. Drawback #2

Support is currently being conducted by a single individual. There is high risk having a single point of failure in the case the change detection database fails. This is contrary to the Defense Innovation Board recommendations, "hiring competent people with appropriate expertise in software to implement the desire state and give them the freedom to do so" (Defense Innovation Board 2022).

10. Summary

This section of the case study highlighted specific benefits and drawbacks for the warfighters at USSTRATCOM. Analysts now have more oversight into developing intelligence, activity in an ETF, and less time devoted to reviewing an entire record, allowing analysts the ability to update ETFs quickly, as new intelligence is reported. Leadership now has more awareness of the target maintenance process, more fidelity in data associated with tracking the process, and the confidence targets remain current on a timelier basis. The benefits of this new program will continue to show value as the command moves forward with using the software. New data, as it is captured, can be shared with the J7 for exercise planning and scenario development with the J3. USSTRATCOM is continually refining the information it uses to test how leadership thinks while playing out scenarios. Having the most up-to-date targeting data will make the command's training more relevant and representative with real world data.

The DOD is becoming more innovative when approaching the creation of new software to suit specific needs not addressed by the commercial sector. Individual service components have created software factories, small hyper focused units, with the intent to develop new or better software tailored to end user needs through a fast and focused agile cycle. The work done by the TPIWG embodied the software factory model, working closely with the end user (service members), rapidly adapting changes and ensuring mission is being met with properly working software. It would seem natural for a new program like this to make the jump into a rapid paced development atmosphere and driven at the services level to success.

IV. SOFTWARE FACTORY

The DOD is becoming more reliant on software for daily operations, vehicles, aircraft, and weapon systems. Development of software has been cumbersome and slow in the past, often not meeting the design mark for an intended application. Adoption of agile software development has been slow or non-existent within the DOD. Acquisition practices within the DOD are ingrained within the community, second nature because it is the way it has always been done. However, change is on the horizon. Instead of forcing all of DOD acquisition to change at once, pockets of 'what right looks like' are coming into existence and producing results in minimal time. Software factories are emerging as large agile engines to rapidly develop and field software solutions for the front-line forces. Software factories are an alternative to developing solutions in-house, reducing risk by allowing professionals in agile environments, build a software application and provide the long term sustainment necessary to update software through its life cycle.

The United States Army is capitalizing on small areas of software development excellence and taking it to a new level. The Army's Futures Command has developed a unique approach to solving software issues, creating its own software factory to tackle software development.

The Army Software Factory concept intends to derive organic cultivation of software developers and coders from within Army ranks, working with individuals from industry, to help solve software problems or develop new software for the Army. Many soldiers have the requisite skill sets and the Army wants to capitalize on it (Thayer 2021). The Army is offering its entire force the opportunity to apply to become a member of the Software Factory. Applicants do not need to have a background in coding or have any type of formal training, however many have experience from their natural curiosity from childhood or in their spare time (Marci 2021). The Army will then place them into a cohort for six months of formal training, with four cohorts conducting the formal training spaced throughout a calendar year.

1. Unique Service Approaches to Software Development

The Navy is taking a similar, yet slightly different approach with a software factory. The Navy has established the Forge at the University of Maryland College Park with teams from the defense industry, small business, government, and academia—all taking a new approach to quickly solving the Navy's problems and outfitting sailors with new and more lethal capabilities through rapid software development, certification, testing and fielding (Eckstein 2021). Naval vessel capability has been largely measured by the quality in naval architecture and weapon systems installed (Lavelle 2020). The Navy has now recognized that to keep pace with adversary capability, it must close the capability gap. Instead of fixing software issues for fleet feedback items during re-baselining of major systems, the Forge is to tackle and provide updates to software based on feedback from the fleet.

The Air Force has taken an approach which is more akin to what the Army is trying to establish through the Futures Command. The Air Force adopted the DevSecOps process through a new command named Kessel Run, which works directly with industry partners and Air Force personnel to help develop secure software to fit Air Force needs in a fast and cost-effective manner. The focus of Kessel Run was to help support the Combined Air Operations Center (CAOC), in the U.S. Central Command, with their software needs. Defense contractors were unable to fulfill the needs of the CAOC in a timely manner, and an Air Force officer developed an Air Force organic software development capability with the Kessel Run program. Fielding the KRADOS software within three weeks as a viable solution for operational forces (Katz and Ising, 2021).

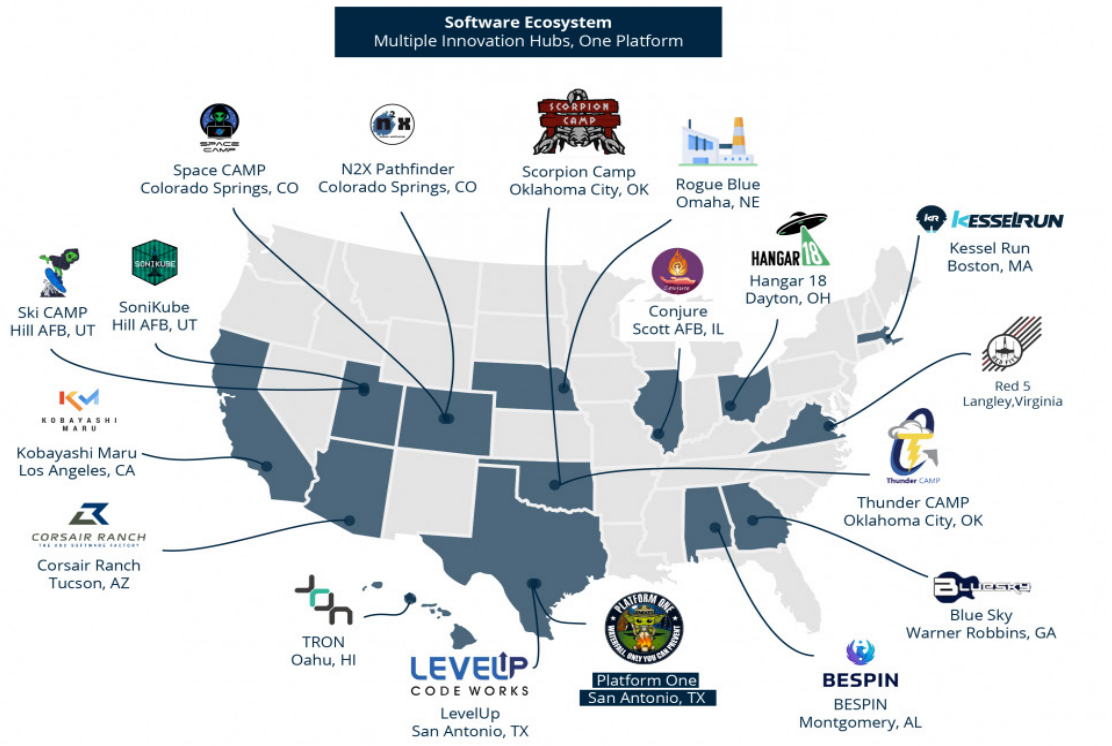

Figure 13. Software Factory Ecosystem. Source: Software Factories (2022).

The Air Force was the first of the services to adopt a software-centric development center or software factory, to help fill software needs organically, but also to enforce industry practices to help reduce the time and cost for new software development (Kessel Run 2022). The Army software factory is trying to emulate the Air Force approach but also incorporate more soldiers into the process by offering tours with full training for those who are accepted to the program. The Air Force has career fields for program management and software program management and not specifically bringing airmen into the process to work side by side with industry, but instead the Air Force manages and steers the software development in the desired direction. From joint experience, the Army likes to be involved in processes and also own the process. So, seeing the Army bring in soldiers to work directly in the process, with industry, providing field experience and feedback directly into the development at hand, is the right way to do it, but with the Army approach.

2. Soldier-centric Design

The approach by the Army is placing the soldier at the center of the design. By placing the soldier at the center, the Army is allowing the soldier to think critically and solve problems while working in a dynamic software development environment. Soldiers coming from the field have a clear understanding of what a soldier's needs are, operating environment, systems capabilities and limitations, problems that need to be solved, and a soldier's sanity check. Soldier-centered design places the soldiers, who will ultimately use a system, squarely in the center of the design process and ensures that their needs are the foremost consideration when making design trade-offs and decisions (Knepshield 2021). Soldiers who are coming from the field have a unique perspective of understanding daily life and how business is done day to day. A soldier developer has years of experience to provide insight into what the needs are in the field and how it will work for another soldier on the front line. The next perspective a soldier can provide in the development of software is what the operating environment is like (Siter 2020). The operational perspective can provide insight into what software actually needs to do or provide for the soldier in the field. Soldiers have operational familiarity of equipment being used, how current processes work, needs for new software, and what is practical in the field.

3. The Army Cohort Model

The Army software factory has developed a cohort model, grouping new participants as they enter the program and progress through their tours. The cohorts begin every six months with 25 active-duty Army and five Department of the Army civilians, totaling 30 individuals. The cohort progresses through the program's three phases at roughly the same timeline but along different paths, based on individual tracks. Cohort members are given 18 months to complete phases 1 and 2.

The first phase is an industry technical accelerator, designed to provide a baseline and a common understanding of each of their respective tracks. The basics of DevSecOps is also taught to all cohort members. There are four possible tracks each individual can take: Product Manager; User Experience / User Interface Designer; Software Engineer; and Platform Engineer.

The second phase begins with cohort members' placement into software development teams and assigned a current software problem being faced by the Army. Industry experts are then paired with the teams to coach and mentor them while also contributing to the task at hand. The teams then begin to scope and frame out the problems leading to design solutions.

At the conclusion of the second phase, cohort members will graduate to become a member of the software factory team. The third phase begins once they become full members of the software factory, continuing as members of software development teams, working on solving problems for the Army. What individuals will do after the completion of their 36-month tour is still being developed by the Army, but the plan is for these individuals to continue working to solve Army problems, supporting other commands (U.S. Army 2021).

4. Future Force Design

Teaching soldiers the new skills to bring to bear on problems the Army is facing will lead to a change in how the force approaches problem solving and at what level problems can be solved. The Army software factory is nearing the end of its first year and will have produced 120 trained individuals to operate in software product management, user experience/interface design, software engineering, and platform engineering. Over the next ten years, if the software factory numbers remain static, the Army will have 1,200 trained individuals put back into the Army with unique training and understanding of software development. These individuals in turn can educate their gaining commands in the process and efficiencies with how the software factory works, but also working on developing or solving issues at their level will be extremely beneficial.

5. Combatant Commands Need Software Factory Capabilities

If USSTRATCOM had the capability to utilize capacity at a software factory, the change detection database could easily have been completed within a similar timeframe and to DevSecOps standards. What a software factory can do, that the TPIWG could not, is develop interoperability with other proprietary software used by the command, where data intersects and used by multiple entities. The TPIWG used JEMA in the version I of

the change detection database because access was not granted between the programs and JEMA was an easy intermediary to extract data, but it could not extract data in the fidelity needed. Version II of the software does a better job at extracting data, no longer utilizing JEMA, but through arduous work with J56 to gain access to databases the command maintains, the change detection database is more capable. A software factory would have a centralized knowledge of how all software works, interacts and shares data within the command, creating solutions with each iteration of the agile development cycle.

Software factories are expensive to operate, having an overhead cost a combatant command could not justify. However, if a partnership is built with a geographically collocated software factory (Figure 12) and spare capacity is contracted out to fill the needs of a combatant command, this presents a lower cost solution to solve software needs in a fraction of the time it would take to contract out and wait for software development. This will also reduce risk of having non-specialized individuals attempting to solve and build a software solution.

V. TPIWG CHALLENGES

The TPIWG transitioned the change detection database from a working group environment to a sustainment environment under the care of the J52 Automation team. The automation team currently maintains the database as is, making larger changes and updates but also researching contracted sustainment options.

Long term sustainment funding vehicles are few and far between for software and applications. The J52 is not appropriated funding for software and development and neither is the J56 for that matter. This lack of funding means that gaining and sustaining viable products for the organization can be a challenge when "innovating from within." Therefore, J52 use of Microsoft Access is demonstrative of "creative" use of tools that the command makes available and not really innovative. In short, J52 logically tied our innovation to a capability gap that exists among current applications. The next step is to contract support for the larger enterprise of applications that will be sufficient for developing the solution that replaces the Microsoft Access product with a sustainable application.

"Sustainability" is the key concern in this situation and has three definitions for the J52 division:

- The organization is billeted with the appropriate skill sets to use and maintain the application;
- Funding available for enhancing, changing, ensuring that the application is viable for operation in the workplace (e.g. IA/cyber sec updates);
- The organization can use the application without the risk of the application introducing risk of mission failure (e.g., massive data corruption events/ errors that either have an actual or perceived impact to the organization's mission).

1. Challenges of program funding when the end requirements are uncertain from a Cost/Schedule/Performance perspective: Is there a color of money issue?

There are currently no color of money issues with the change detection database. J52 has obtained simplified acquisition of base engineer requirements (SABER) funding though the USAF. Lockheed Martin is the primary contractor to work on the SABER contract, funded by AFLCMC, but with a scope of work limitations (Figure 14). J52 has neither levied a requirement beyond the scope of research and development for SABER, nor levied a user requirement to Lockheed Martin beyond the scope of their work. The challenge is getting J52 objectives met within the scope of research and development with SABER to deliver a MVP. The other challenge is getting Lockheed Martin to deliver on a timetable that provides utility and buy-in from the user community.

ATTACHMENT 1a - SCHEDULE OF MILESTONES
STTR funds will not exceed $1.25M

The below chart should detail the proposed milestone list. Milestones will correspond to a measurable event's completion, e.g., baseline execution plan, test plan, prototype production or a component thereof, final report, etc. Status reports shall not be milestones. The milestone description should indicate demonstrated completion. Associated milestone payments must reflect comprehensive completion costs.

TASK *Max 15 milestones.*	EXPECTED DELIVERY (MONTH AFTER CONTRACT AWARD)	DELIVERABLE	ACCEPTANCE CRITERIA	PAYMENT *Max STTR funds $1.25M; non- STTR Federal funds, which have no max, must be identified.*
Deliver automated target change detection solution	Award + 3 months	Produce, test, and deliver a change detection module on JWICS that notifies analyst groups of changes to target records.	The application detects record changes in databases, notifies users of change by type, and displays the data that has changed.	$250,000
Track role-based user actions on change detection	Award + 4 months	Knowledge management (tracking and analysis) capability within the application	The application tracks user actions concerning record change, tracks the amount of time users spend actioning the change	$250,000
Create metrics for assessing organizational measures of performance and measures of effectiveness	Award + 6 months	MOP: total number of changes detected (by analyst, by region, by target type, by RESPROD), total number of analyst actions taken MOE: total time analysts spend in change detection, total number of "noise" reports eliminated/filtered, analytic data on regional analysts' time compared to detected volume of change.	The application provides metrics/data points that demonstrate its capability to detect change, and provides insights into how effective the organization is structured given the volume of record change compared to available manpower.	$150,00
Analyze data change over time	Award + 9 month	Advanced analytic capability to aggregate change data over user-defined periods of time.	The application captures data change over time and analyzes it for trend analysis and insights.	$250,000
Create knowledge about the operational environment that eludes current analytic capabilities	Award + 12 months	Advanced analytic capability that provides insights into the change data, extrapolating the meaning of the change given the physical and functional characteristics.	The application describes the types of data that are changing over user-defined periods of time and provides a baseline for analysts to determine analytic significance of the change.	$250,000
Create reports that assess relative level of effort across the user community and enable leader decisions	Award + 15 months	Reporting function that formats reports (MOP, MOE, change data analysis), including metrics and data points, that enable leader decisions.	The application produces reports routinely (daily, weekly, monthly) and on-demand.	$100,000

Figure 14. J52 SABER Schedule of Milestones

2. Positives and negatives of conducting work in-house

Positives of doing the work in house: J52 controls the speed, detail, accuracy, prioritization of Microsoft Access development and is LOW COST.

Negatives of doing the work in house: J52 personnel are not software developers, which means that inherent limitations of Microsoft Access are imputed into the methodology and model. Microsoft Access databases are difficult to manage over time. If the J52 single individual, not an exaggeration, can no longer update the product, then a mission risk exists for the division.

3. Challenges are faced by brining on a contractor in this type of environment

The challenges of bringing a contractor into the workplace relate to security, and the fact that J52 does not have a JWICS development environment. This means the software must be developed and deployed elsewhere. This is a challenge shared with Lockheed Martin as well. Lockheed Martin has conducted business with USSTRATCOM long enough, that there are business practices and procedures in place to mitigate this issue. However, getting both the developer and user in front of the software, in the workplace where the user works, is the single greatest challenge and most important to solve. Direct developer and user interaction leads to the best results, both in developing the software to a fully viable product and in user acceptance.

VI. SUMMARY

This case study has covered the TPIWG process at USSTRATCOM, Agile software development, DevSecOps, creative development and the exciting software factory prospects being developed by individual services. It is important to recognize the reliance on software within the DOD and its use for large quantities of data and across all weapon systems. Software needs will only increase with the increased reliance and the timeframe that software will be need to be fielded or upgraded will only decrease.

Exploring creative solutions to design and field better software, quicker and at a lower price is beneficial not just for the DOD but for developers and contractors who support the DOD. Partnering to increase efficiencies and understanding of needs will create a better environment for problem solving and useful tool creation for the customer. Software factories provide the capability previously mentioned and should be leveraged by more DOD agencies and combatant commands. Services have always leveraged the ingenuity of their personnel to solve problems and the services have now captured the ability of their members to solve software problems in an environment which allows them to focus solely on the problems, teaching the DevSecOps and agile processes at software factories.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. RECOMMENDATIONS

It is recommended that combatant commands adopt agile development models to build software solutions and managing its development or to make use of the service level software factories, expanding the use of the joint force, creating solutions faster, at a lower cost, reduced risk and with proper security protocols.

In-house software development can become cumbersome if the properly trained individuals are not on the product team. With the properly trained individuals, adopting an agile development method will help keep the team focused and on track to accomplish its initial task. Agile offers a model where a team can design and implement in an iterative fashion, and doing it yourself is always cheaper than paying someone else to design and create a product.

Lower costs can be achieved through a more efficient process at the software factory level rather than seeking proposals to requests for bids in software development. Individuals at the software factories are trained professionals who are well versed in software management, coding, software solutions, software development problem solving and management of information technology systems. Software factories also have the required development ecosystems to provide the security necessary for the DevSecOps process with the speed of an agile development cycle. Further software development timeframes will be shortened and customers will have a lower level of risk along this software development path. Higher risks are incurred when doing the work in house or contracting to a third-party vendor.

Combatant Commands do not need to contract with software factories full time, as this may not be necessary. However, as project such at the TPIWG arise, combatant commands could rent spare capacity from software factories. If the scope of a software requirement increases, full time production may be recommended by the software factory to ensure risk remains low. Software factory use should be explored further, allowing specialists in software development to properly create solutions for customers.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Adam, J. (2022, March 22). What is Agile Software Development? Krusche & Company. https://kruschecompany.com/agile-software-development/

Agile Alliance (n.d.) Agile Manifesto. Retrieved June 14, 2022. https://www.agilealliance.org/agile101/the-agile-manifesto/

Agile Alliance. (n.d.) What is Agile. Retrieved June 14, 2022. https://www.agilealliance.org/agile101/

Army Software Factory Information Sheet (n.d.) U.S. Army. Retrieved March 27, 2022, Chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/ https://afcwebsite.blob.core.usgovcloudapi.net/uploads/assets/ Army_Software_Factory_Web_Infosheet_5_MAY_21_0b764e291d.pdf

Cummings, S. A. (2020) *Agile Software Acquisition Guidebook.* Office of the Under Secretary of Defense for Acquisition and Sustainment

Defense Acquisition University. (n.d.) Adaptive Acquisition Framework Pathways. Retrieved March 28, 2022, https://aaf.dau.edu/aaf/aaf-pathways/.

Defense Acquisition University. (n.d.). Software Acquisition. *Planning Phase*. Retrieved March 27, 2022, https://aaf.dau.edu/aaf/software/planning-phase/.

Defense Innovation Board Do's and Don'ts for Software (n.d.) Defense Innovation Board. Retrieved March 29, 2022, chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://media.defense.gov/2018/ Oct/09/2002049593/-1/-1/0/DIB_DOS_DONTS_SOFTWARE_2018.10.05.PDF

Eckstein, M. (2021, April 12). Navy Software Factory, The Forge, Wants to Reshape How Ships Get Upgraded. *USNI News.* https://news.usni.org/2021/04/12/navy-software-factory-the-forge-wants-to-reshape-how-ships-get-upgraded

Katz, B. and Ising P. (2021, January 12) Kessel Run Deploys KRADOS to Air Operations Center, Kessel Run. https://kesselrun.af.mil/news/Kessel-Run-Deploys-KRADOS.html

Kessel Run (n.d.) Kessel Run. Retrieved March 27, 2022. https://kesselrun.af.mil/

Lavelle, Sean (2020, February 27). Fixing the Navy's Software. *War on the Rocks.*https://warontherocks.com/2020/02/fixing-the-navys-software/

Marci, K. (2021, September 15) The Next Generation of Soldiers Will Be Software Engineers. *Government CIO Media.* https://governmentciomedia.com/next-generation-soldiers-will-be-software-engineers

NASK. (n.d). JEMA. Retrieved March 25, 2020, *https://www.nask.world/jema/*.

Nyce, C. M. (2017, December 8) The Winter Getaway That Turned the Software World Upside Down. *The Atlantic.* https://www.theatlantic.com/technology/archive/2017/12/agile-manifesto-a-history/547715/

Oakley, S. S. (2021). Status of Challenges Related to Reform Efforts: DOD Software Acquisition (GAO-21-105298). Government Accountability Office.

Pearsons, T. M. (2020). *Agile Assessment Guide.* (GAO-20-590G). Government Accountability Office.

Pearsons, T. M. (2020) *Agile Software Development.* (GAO-20-713SP). Government Accountability Office.

Powner, D. A. (2012) *Effective Practices and Federal Challenges in Applying Agile Methods* (GAO-12-681). Government Accountability Office.

Savage-Knepshield, P. A. (2012). Designing Soldier Systems & Evaluation Techniques.

Sherman, J. B. (n.d.) DOD Enterprise DevSecOps Fundamentals. Retrieved March 29, 2022. chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://software.af.mil/wp-content/uploads/2021/05/DOD-Enterprise-DevSecOps-2.0-Fundamentals.pdf

Siter, B. (2020, January 3). Soldier Centered Design proving vital to kitting Soldiers faster and more efficiently. *U.S. Army.* https://www.army.mil/article/231425/soldier_centered_design_proving_vital_to_kitting_soldiers_faster_and_more_efficiently.

Software Factories (n.d.) Assistant Secretary of Acquisition Chief Software Office Retrieved March 29, 2022) https://software.af.mil/software-factories/.

Thayer, R. L. (2021, November 19) Army's software factory builds apps to improve the lives of soldiers, gain edge on battlefield. *Stars and Stripes.* https://www.stripes.com/branches/army/2021-11-19/army-software-factory-technology-apps-soldiers-austin-texas-3683582.html

Thiry, M. (2010). Agile Program Management. *PMI® Global Congress 2010.* *https://www.pmi.org/learning/library/agile-management-myths-analyze-components-6562*

Varhol, P. (n.d.) To agility and beyond: The history and legacy of agile development. *Tech Beacon.* Retrieved June 16, 2022. https://techbeacon.com/app-dev-testing/agility-beyond-history-legacy-agile-development

West, D. (n.d.) Scrum roles and the truth about job titles in scrum. Retrieved June 16, 2022. https://www.atlassian.com/agile/scrum/roles

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California