A Case Study: Introducing eXtreme Programming in a US Government System Development Project

Ann Fruhling[1], Patrick McDonald[1], Christopher Dunbar[2]
[1]University of Nebraska at Omaha,[2]FGM, Inc.
afruhling@unomaha.edu, *pmmcdonald@unomaha.edu, dunbar@fgm.com*

Abstract

The US Military's ability to meet its mission critical requirements calls for increased agility in its information technology development process. The purpose of this case study is to better understand how agile principles, specifically eXtreme Programming (XP) practices, can be effectively introduced and implemented into government organizations that have historically embraced the plan-driven traditional software development environment. In particular, we studied the eXtreme Programming process to develop a new capability for USSTRATCOM's premier knowledge management system, SKIWeb. There were several lessons learned that will assist practitioners in future implementations of XP and new research questions emerged that suggest further study.

1. Introduction

Rapidly changing operational requirements and a subsequent growing need for a leaner U.S. military force have both government leaders and defense contractors seeking faster and better ways of getting critical information and decision-making tools into the hands of decision makers. This has led to a strategy of decentralization of information, a key component of which has become the migration toward net-centric, service-oriented architectures (SOA) for military information systems [11]. While this type of infrastructure seems well suited to the task of supporting emergent requirements, the linear nature and high-density documentation required by plan-driven software development methods traditionally employed by the military typically do not [1]. Attempts to address this drawback have come in the form of spiral development approaches that essentially break the traditional waterfall method down into smaller iterations based on risk or priority of functionality [13].

More recently, Agile software development methods (i.e. eXtreme Programming [3], Scrum [7,17]) have aimed to further streamline the development process and bring significant improvements such as timely delivery of required functionality. Agile styles also purport to embrace changing or unclear requirements while promoting user and developer interaction [4].

While the use of methods, like eXtreme Programming (XP) exist in government and military environments, they have yet to reach their potential. Still, today, traditional plan-driven methods are the mainstay, partly because they are considered by some to have less risk and they support CMMI level-5 certification [2]. (Most DoD projects are required by law to be CMMI level-3 compliant.) Also, plan-driven methods are more consistent with the hierarchical structured military culture. Never-the-less, there is a noticeable movement towards agile approaches.

The purpose of this paper is to better understand how XP practices can be effectively introduced and implemented into government organizations that have historically embraced the plan-driven traditional software development environment. We present a case study and the lessons learned where a defense contractor implemented an XP development process pilot, referred to as Phase 1, to develop a new capability for U.S. Strategic Command's (USSTRATCOM's) premier knowledge management system, SKIWeb. The paper is structured as follows. We begin with background information on USSTRATCOM and SKIWeb, the U.S. military's call for more agility in its system development processes, and an overview of Agile methods and XP practices. Next, we discuss our research method and the Phase 1 introduction and implementation of the XP process. Subsequently, we present the survey results and participants feedback from the interviews. Lastly, we conclude with a discussion of the lessons learned.

1530-1605 2008 U.S. Government Work Not Protected by U.S. Copyright

2. Background

2.1. USSTRATCOM and SKIWeb

USSTRATCOM, headquartered at Offutt Air Force Base, Nebraska, is a unified command comprised of members from all four branches of the U.S. military. Its role in the armed services is as a "global integrator," with responsibilities including information operations, strategic warning and missile defense, and global command and control functions [14]. SKIWeb, or the Strategic Knowledge Integration website, is an event tracking and blogging tool used at USSTRATCOM. Its purpose is to facilitate the sharing of information regarding military and world events between members of the strategic command and intelligence communities [10].

SKIWeb represents a significant paradigm shift in military operations in that everyone in or associated with the command can access and use the system, creating direct lines of communications between the lowest ranking specialist and the highest-ranking commander. The intent of the system is to capitalize on the idea that everyone in the command may have some insight or information that can be used to accomplish USSTRATCOM missions more effectively.

SKIWeb is one of many "Command and Control" systems at USSTRATCOM. The Command and Control Software Engineering and Support project known as C2SES is supported by a team of external contractors.

2.2. Agile Methods and Government Contracting

The term 'agility' is a key attribute when describing how to measure a military unit's ability to meet varying sets of mission requirements. Agility in the context of the military attribute is the ability to quickly adapt and adjust to changing conditions. According to US Defense Department CIO John G. Grimes, "…succeeding in the new strategic environment requires levels of responsiveness and agility never before demanded of our forces. The U.S. Defense Department must transform from its historical emphasis on ships, guns, tanks and planes to a focus on information, knowledge, and actionable intelligence." [8] SKIWeb is one example of the kind of system envisioned by Secretary Grimes. Migration towards information-based warfare, combined with a need for greater agility stemming from rapidly changing requirements, calls for change in the way military information systems are developed.

2.3. Overview of Agile Software Development and eXtreme Programming

The Agile method encompasses a set of principles stressing valuation of early and continuous delivery of functionality, collaboration, and responsiveness to change over heavy documentation, negotiation, and plan-driven project management [4]. On one hand, some claim that this is in direct contrast with traditional plan-driven development models and is essentially "undisciplined hacking" [12], and on the other hand, others argue that Agile methods' focus on short delivery cycles and continuous refinement [3.4] and therefore, Agile methods are as rigorous as plan-driven approaches. The Agile Manifesto points out that formal planning and documentation are still essential, but should be pared down to the essentials in creating environments that are more responsive to change [9].

The Agile method adopted for the SKIWeb project at USSTRATCOM is eXtreme Programming (XP). XP focuses on the application of "programming techniques, communication, and teamwork," and utilizes a core set of twelve principles [3]. The twelve principles are: Incremental Planning, Small Releases, Simple Design, Test-first Development, Refactoring, Pair Programming, Collective Ownership, Continuous Integration, Sustainable Pace, On-Site Customer, Uniform Coding Standard, and System Metaphor. They are further described in the XP Phase 1 section.

3. Research Method

This research represents a single descriptive case study. We chose a descriptive case study, because it is particularly well suited to support learning by answering 'how' and 'why' questions [15]. Case studies often explain the reasons for a problem, the background of a situation, what happened, and why.

We gathered data from a variety of sources during the study. We attended team meetings, reviewed meeting minutes, prepared a post-implementation evaluation survey and interviewed all key stakeholders involved in the XP project. The roles of the researchers were

both observer and mentor.

4. XP Phase 1

The intention of Phase 1 was to pilot an XP process to implement a new SKIWeb system capability. At the end of the pilot, the project team would review the process, make adjustments and continue phasing in the XP development process across the project team.

Two experienced developers, a development team lead, and a government functional manager (customer) were initially assigned to the XP project. Later in the XP project, a Human Factors Engineer (HFE) was included. The developers were responsible for designing and building the actual end product. The development team lead was the direct supervisor of the developers and was responsible for the XP process facilitation. The government functional manager was the main end-user community representative. In the context of this project, the HFE was responsible for the design of the user interface and developing story boards for new capabilities. Also, two project managers (a software engineering team manager and the software development team manager) and a project documentation coordinator attended most of the weekly meetings. All participants on the XP project were external contractors with the exception of the government functional manager.

At the initiation of Phase 1, the XP team was given training on the general goals of Agile methods and detailed information on the practices of XP in particular. One of the researchers met with the XP team weekly to observe, discuss issues and provide guidance. In addition, one of the developers had previous experience using XP practices.

At the conclusion of Phase 1, the researchers met with all the key stakeholders to debrief the XP process. The meetings with the key stakeholders were divided according to their role in Phase 1. In other words, the researchers met with the developers and lead developers, the user, the HFE, and the management in four separate meetings. Evaluation surveys were distributed and compiled. The researchers also analyzed notes and observations during Phase 1.

Phase 1 began August 7, 2006 and ended October 25, 2006. Four "drops" or iterations of new capabilities software were completed. It was intended for the the drops to occur every two weeks, however, due to holidays, vacations, production problems, and other enterprise-wide constraints some drops occurred at three week intervals.

The XP developers sat next to each other in the same cubicle. Each developer had her own computer, however there was room for the two of them to sit together and work on the same computer. The development team lead sat across the aisle. The functional government manager had an office in the same building and floor. Both of the developers were new hires in the past year. The development team lead was a new hire a few weeks before the XP pilot. The government functional manager had been with the project team for over a year and had a good understanding of the system. The HFE was located on a different floor in the same building. Most of the time, the developers and usually the team lead met with the user collaborator on a daily basis and sometimes more than once a day if needed. Throughout the pilot, the project team met on a weekly basis to assess the XP process and discuss issues. The researcher participated in these weekly meetings. Status reports on the progress of the development effort were also reported to the government officials at other weekly meetings.

The two managers, for the entire development project team, thoughtfully considered which enhancement capability would be the best candidate for the XP pilot process. The main task that was selected for the project was to implement an advanced search capability for SKIWeb. After the two developers for the XP team were told which new capability would be assigned to the XP project they did some initial planning and divided the capability into subtasks that they thought could be accomplished in the two week cycles. The four subtasks were 1) Implement "and-or-not" search functionality; modify GUI for aesthetics, 2) Implement free text search when creating User Defined Event Logs, 3) Enhance Advanced Search filter capability; and 4) Implement advanced search for Add Related Events capability. During the third week of Phase 1 an additional subtask surfaced which was to revise the existing GUI, and therefore, the HFE was requested to become involved in the XP process and fifth subtask was added.

During Phase 1, the researchers made the following observations. 1) Developers and the government functional manager, referred to as the "user collaborator", had competing work assignments which interrupted the work flow. 2) The user collaborator had a search capability in mind that the developers did not currently have

all the technical expertise to implement, so the developers had to learn new technologies during the task. 3) The user collaborator went on vacation when the XP team was ramping up and gaining momentum. There was a back-up user collaborator assigned during his absence. 4) Several internal C2SES communication issues surfaced and had to be addressed during Phase 1. 5) The HFE was not an initial participant of the XP pilot team.

5. Findings

The findings from this case were formulated from several sources which include the researchers' observations, weekly status meetings, interviews and surveys.

After the interviews with the participants in the XP project were completed, the researcher asked each stakeholder (including the HFE) to complete a survey. The intention of the survey was to further document the feedback and use the data as a future benchmark. Seven of the eight individuals returned the survey. Three respondents did not answer all the questions and were therefore they were not used in the statistical analysis.

The three individuals who did not fully complete the survey were not directly involved in the day-to-day process. The one individual who did not return the survey was a "back up" user collaborator. The survey asked basic demographic questions such as role in the organization, years of IT experience, and experience with agile software development process. Open-ended questions were asked about team communication, XP task selection, and suggestions on future changes and improvements to the process. T survey also asked the individual's perception of the level of management support and the success of Phase 1. The first page of the survey included a description of each of the practices. (These were explained in the training at the onset of the project.) The definitions given for each of the practices are presented in Table 1.

Survey questions that were directly related to the twelve agile practices addressed two areas. First, we asked the respondent to indicate the usage amount of each of the XP practices during Phase 1. A score of 1 indicated the practice was not implemented and a score of 7 indicates that the practice was fully utilized. An average of the responses and the standard deviation are reported, however, the statistical validity is limited due to the small sample size. The results are shown in Table 2.

Table 1 – Extreme Programming Practices [3]. [6]

Practice	Practice Overview
Incremental Planning	Also "Planning Game." Work is planned in chunks, with the on-site customer playing a large part in requirements determination and work prioritization
Small Releases	Development team puts core functions into production early and then builds upon them using feedback from users and customer collaborator
Simple Design	Simple and small pieces of design allow frequent changes to be made as necessary.
Test-first Development	Writing and running unit tests as code is written to make sure the system is working properly. Frequent user acceptance tests to ensure the system is fulfilling user requirements.
Refactoring	A technique used to improve code without altering functionality. Focuses on simple, clean, non-repeating code that can be easily changed.
Pair Programming	Two programmers developing production code at the same time on one machine.
Collective Ownership	Everyone owns all of the code, allowing necessary changes to be made by anyone at any time.
Continuous Integration	Integrating changes as they are complete to detect system failures as soon as possible.
Sustainable Pace	Developers keep a normal work schedule to remain productive, and interested in the project.
On-site Customer	An on-site customer available for requirements clarification and business decisions that should not be made by the Developer
Uniform Coding Standard	Coding standards (language, formatting, syntax) agreed upon by the team at the start of a project. Facilitates communication and ease of development.
System Metaphor	An over-arching description of the system's functionality, purpose, and conventions, providing customers and developers with common ground on which they can stand

Table 2 – Extent of Implementation

Practice	Avg	StDev
On-site Customer	**6.2**	**0.8**
Small Releases	**6.0**	**0.0**
Sustainable Pace	**6.0**	**0.8**
Incremental Planning / Planning Game	**5.8**	**0.4**
Coding Standards	**5.5**	**1.0**
Simple Design	**4.8**	**0.5**
Collective Code Ownership	**4.5**	**2.6**
Refactoring	**4.0**	**1.8**
Continuous Integration	**3.7**	**2.3**
System Metaphor	**3.6**	**1.8**
Test-Driven Programming	**1.8**	**1.0**
Pair Programming	**1.0**	**0.0**

The results show that the XP principles On-site Customer, Small Releases, and Sustainable Pace were the most used practices. On the other hand, Test-Driven Programming and Pair Programming were essentially not used at all. Also, there was the broad range of responses for Collective Code Ownership, Continuous Integration, Refactoring, and System Metaphor which is evident from the standard deviation.

Second, we asked each respondent to rate their perception of the importance of each XP practice to the success of the project. Table 3 presents the results. A rating of one indicates that the practice was perceived to be not important at all and 7 rating indicates that the practice was extremely important.

Table 3 – Importance of Practice to Success of Phase 1

Practice	Avg	StDev
Incremental Planning / Planning Game	**6.4**	**0.5**
On-site Customer	**6.2**	**0.4**
Small Releases	**6.2**	**0.8**
Continuous Integration	**6.0**	**0.8**
Sustainable Pace	**5.5**	**0.6**
Collective Code Ownership	**5.5**	**1.3**
Test-Driven Programming	**5.3**	**1.5**
Coding Standards	**5.0**	**0.8**
Simple Design	**5.0**	**1.4**
Refactoring	**4.8**	**1.5**
System Metaphor	**4.2**	**1.6**
Pair Programming	**1.8**	**1.5**

The high average scores and low standard deviations suggest overwhelming support from the majority of the project team for Incremental Planning, On-site Customer, Small Releases, and Continuous Integration. Pair programming was rated as not being very important.

The USSTRATCOM environment may provide insights as to why the respondents indicated XP practices were or were not fully adopted and also why some XP practices were rated more important than others. For example, in the current environment, the team appeared to have already gravitated toward three of the agile practices without formally recognizing them as such. For instance, the researcher observed that Small Releases, Sustainable Pace, and On-Site Customer were already in place as a matter of course, though perhaps not to the extent specified in XP. However, once recognized, these were indicated in the survey as being fully adopted. Small releases were practiced as enhancement work done on the SKIWeb system is typically organized into chunks by functionality and then prioritized by senior leadership. Contract work is normally held to a limited number of hours per week with no overtime which supports a sustainable pace. The on-site customer practice was implemented as regular contact with the customer is required to approve work done and set goals for the next iteration. That said, perhaps the biggest questions surround the practices that were not readily adopted.

Only one XP practice was given a low rating both in terms of usage and importance – Pair Programming. This is partly attributable to the fact that at the introduction and training meetings for the Phase 1 the developers openly stated their reluctance to follow the pair programming process as defined by the XP process. The developer that was the most vocal was the one who had past XP experience. She had already made her mind up that she wasn't willing to do this and the team lead did not challenge her comment probably because he was still new to the team. The two developers assigned to Phase 1 did state they worked closely on the tasks, sometimes independently and sometimes they worked together to solve logic and coding problems. Because the project team lead did not enforce the XP pair programming practice, it was neither used nor deemed important.

Table 4 compares the two survey results. Table 4 shows that there are disparities between the attributed importance of some of the XP practices and the extent to which they were used

by the project team. To measure this, the average importance rating was subtracted from the usage rating; a negative delta indicates that the practice was perceived to have more importance than was placed upon it during Phase 1, and a positive delta suggests the practice was less important than the emphasis placed upon it, with the distance from zero indicating the relative magnitude.

Table 4 – Usage / Importance Comparison

Practice	Usage	Importance	Delta
Test-Driven Programming	1.8	5.3	-3.5
Continuous Integration	3.7	6.0	-2.3
Collective Code Ownership	4.5	5.5	-1.0
Refactoring	4.0	4.8	-0.8
Pair Programming	1.0	1.8	-0.8
Incremental Planning / Planning Game	5.8	6.4	-0.6
System Metaphor	3.6	4.2	-0.6
Simple Design	4.8	5.0	-0.3
Small Releases	6.0	6.2	-0.2
On-site Customer	6.2	6.2	0.0
Sustainable Pace	6.0	5.5	0.5
Coding Standards	5.5	5.0	0.5

In other words, of the twelve practices, Test-Driven Programming and Continuous Integration of code stood out as being perceived to be important but rarely used, whereas Sustainable Pace and Coding Standards were considered to be slightly less important than the emphasis placed on them. A possible reason why Test-Driven Programming may have been ranked high, but rarely used is because the project team has a designated tester for quality assurance. Therefore, they relied on this person to do the testing and were not in the habit of writing explicit test cases upfront. Management may have viewed this as redundant effort.

The results of Continuous Integration may be due to the type of task selected. Since the new search capability had well-defined boundaries the developers did not encounter the opportunity to perform continuous integration. As for Sustainable Pace, there was not a change in the number of hours the developers worked per week. In fact, team members seldom have to work overtime as reported by the project manager, regardless of the development methodology used. Currently, the team does not strictly enforce all Coding Standards, so, this may be why it was perceived as unimportant, yet the developers on the XP team expressed their dissatisfaction with the lack of enforcement of coding standards in the overall project and independently instituted a higher level of standards among the two of them. The response from the developers may have been a way for them to express their concern and get management's attention as they knew the results were going to be presented to those who could enforce changes.

In addition to the measurements discussed above, key stakeholders also provided responses to open-ended survey questions and participated in companion interviews to discuss their experiences with Phase 1, with the underlying questions being an assessment on how the XP process went, what issues arose, the lessons learned, and what changes should be considered for Phase 2. Reported issues were generally consistent between respondents, and fell into themes of Management, Communication, Planning, and the XP Principles.

6. Discussion and Lessons Learned

6.1. Management

One common response from the post-process interviews was that the XP team spent too much time working on unrelated tasks that caused considerable distraction and time away from doing their work on the XP tasks. This impacted the success of Phase 1. The survey result that measured the perceived success of the Phase 1 was an average rating of 4.8 (1-low and 7- high) and a standard deviation of .8. In the USSTRATCOM contracting environment, it is not uncommon for development teams to be actively participating in multiple projects

simultaneously and even for team members to be assigned to multiple projects with different teams. This was the case with the team participating in the Phase 1. Occasionally, there were competing priorities that interrupted work flow. This sometimes made it difficult to instill any sort of daily routine for the project team. Several members indicated that more management involvement was needed in order to keep the team functioning and on track.

Also, Phase 1 took place during the summer when several of the other IT developers not on the XP pilot went on vacation which required the XP developers to cover for them. In addition, the project team supported the maintenance of SKIWeb as well as new enhancements. One of the developers on the XP team had to work on a critical production problem in the middle of Phase 1. Although, it was recommended at the onset of Phase 1 that the developers involved should only focus on the XP tasks. This stipulation became impossible as Phase 1 progressed.

Another issue that caused problems was scope creep. About the third week of Phase 1 it was determined that the user interface would need considerable changes and it would be helpful to have HFE involvement. The HFE was not available to assist in the timely manner that the XP team needed. Further, this person seemed to be reluctant to participate in the XP process. Perhaps, partly due to the perception that the XP process was infringing on the HFE's current job responsibility as the chief designer of all the user interface modifications. Also the HFE was not included in the initial task selection discussion and was not integrated into the XP Pilot team originally.

In summary, there are three key findings that resulted from stakeholders' feedback and researchers' observations. First, the XP team must work on only XP tasks and not be pulled off for other tasks. This is important so that they are able to focus on the tasks at hand and to deliver the new capabilities in iterative cycles and on schedule. The XP project team did meet the deadlines, but there was an increased amount of stress to do so. Second, if management agrees to additional requirements (which is understandable) that cause substantial scope creep, for example the major redesign of the user interface, then the expectations of what will be delivered in the future iterations needs to be re-prioritized and adjusted. Lastly, the HFE and other technical experts need to be available to accommodate the needs of the XP developers in a timely manner and as needed. Management should include technical experts in the discussion of task selection and project initiation.

6.2. Communication

The most frequent point made by interviewees was the importance of timely, complete, and accurate communication among the development project team. This was especially relevant between the developers and the HFE. There were times, when e-mails were sent and not answered quickly enough, meetings were scheduled and not everyone was available to attend. Therefore, decisions were made, but not always completely and accurately communicated to everyone. Perhaps, the reason why so much emphasis was placed on communication was that in the XP environment, communication is key. One survey response provided the insight that s/he perceived there was resistance to change from the traditional hierarchical communication to the more collaborative style used in XP.

Further clarification is also needed on the process of involving technical experts (i.e. HFEs, DBAs, Security experts, etc.) and then communicating the expectations to the technical experts by their management. Suggestions included explicitly setting ground rules for communications (i.e. response time, availability for meetings, etc.) and defining an internal issue resolution process before embarking on the next Phase of the XP process roll-out.

On the other hand, there was a consensus on the satisfaction with the communication improvements gained between the Government and the Contractor through the introduction of the user collaborator. From the researcher's observations and the feedback in the interviews, the communication process was only an issue among the internal technical staff. And in fact, the communication between the technical staff and the user collaborator increased both in depth and breadth. The user collaborator interacted with more technical staff members than in the past and more frequently. The main lesson learned is that there needs to be timely, accurate and complete communications among the team members, management, and the user collaborator to effectively and successfully introduce agile methods. This is explicitly stated as a key component of agile methods and this study only further demonstrates the importance and value of communication. Further, it is important that the team lead have the expertise and time to

champion the project on a daily and sometimes hourly basis to facilitate the communication process.

6.3. Planning

A few individuals raised issues regarding the appropriateness of the task selected for Phase 1. Since SKIWeb is a system that is already in use, most, if not all, of the software development tasks are enhancements to add new functionality or address problems with existing segments. In this case, the task was the addition of a new search capability. The IT managers who identified this new capability recognized that technical Oracle expertise would be needed and this was taken into account on the selection of the developers. However, the developers quickly realized they did not have the depth of technical expertise to quickly implement the requested search algorithms using Oracle. Thus, a significant portion of their productive time had to be dedicated to learning advanced features of Oracle and at the same time maintaining responsibility for progress. This created extra stress and reluctance on fully designing the new features as the user collaborator envisioned. The lesson learned is that it is important to assess the technical expertise of the developers and evaluate if they have the skill set for the tasks assigned.

Another issue that arose was the XP team composition and the barriers encountered when it became clear that certain user interface system designs would need HFE support. The HFE was not originally considered a key stakeholder in the initial XP project charter. This decision was discussed; still, as the managers understood the requirement they did not expect the search capability to impact the user interface to the level that it required HFE involvement. This assumption did not hold true and as the user collaborator and developers worked on the second iteration, new ideas on improvements to the user interface design became apparent and interaction with the HFE became necessary. The HFE was reluctant to get involved since s/he was not included in the initial XP pilot team. This caused problems and miscommunication that could have been avoided. The lesson learned is that there needs to be a thorough understanding of the impact and boundaries of the selected XP tasks and the roles and expectations of the technical experts need to be clearly defined and understood by everyone. Thus, the technical experts should always be considered key stakeholders.

6.4. XP Principles

As indicated in the survey responses, Incremental Planning received high marks for both importance and usage. However, feedback from the companion interviews and responses to open-ended survey questions suggested a need for improvement. For example, half the respondents indicated a need to spend more time on requirements determination and systems engineering before handing the project over to the XP team. While this seems directly at odds with the principles and practices of XP, it might also be interpreted as the lack of recognition of the differences between XP and the plan-driven development approach. Thus, this suggests there needs to more in-depth Agile/XP training prior to execution the next phase of the XP process and additional mentoring and team lead oversight during Phase 2.

In the end, small-scale Incremental Planning was generally well-received, and verbal responses from the government customers indicated an overall sense that the XP pilot had sped up the deployment of new features and capabilities without introducing additional re-work or defects. In fact, the customer received smaller increments of new features every two to three weeks, instead of having to wait the usual 6-8 week intervals for the entire enhancement.

Although there were no quantitative results captured, it was the perception of the development team lead that the end product produced by the XP team was of a better quality, had fewer defects, and better met customers' needs.

7. Conclusion

In summary, there were several lessons learned and contributions from this case study. The lessons learned centered around the importance of team member communication, understanding the XP practices and processes, and on-going management involvement and planning.

The main limitation of this study is that the results focus on one organization at one point in time and therefore, they can not be generalized. In addition, the small sample size limits the amount of interpretation that can be made from the survey statistics. Never the less, this study

makes an insightful contribution to XP practitioners, especially those involved in government and military installations. Practitioners can incorporate the lessons learned in their planning and implementation of XP development team projects. Further research is needed for us to better understand the tradeoffs in risk and value between traditional and agile development approaches in government and military environments. [2].

8. Acknowledgements

The researchers would like to thank USSTRATCOM and Mr. Richard Corbin for their support on this research and the opportunity to work with the C2SES project team.

9. References

[1] Alleman, G, Henderson M., Seggelke R. "Making Agile Development Work in a Government Contracting Environment - Measuring velocity with Earned Value," adc, p. 114, Agile Development Conference (ADC '03), 2003.

[2] Armstrong, D. Interview, USSTRATCOM, December 16, 2006.

[3] Beck, K. Extreme Programming Explained. Boston: Addison_Wesley, 2000.

[4] Beck, et al. "Manifesto for Agile Software Development." 2001. Agile Alliance. http://www.agilemanifesto.org

[5] Boehm, B. and Turner, M. "Management Challenges to Implementing Agile Processes in Traditional Development Organizations." IEEE Software 22.5 (2005) 30-39.

[6] Cao, L., Mohan, K., Xu,P., and Ramesh, B. "How Extreme does Programming Have to be? Adapting XP Practices to Large-scale Projects," In 37th Hawaii International Conference on System Sciences, January 2004.

[7] Cockburn, A. Agile Software Development. Reading, MA: Addison-Wesley. 2001.

[8] Grimes, John G. "Which Emerging Technology Will Have The Biggest Impact on Your Organization in the Future?" Signal, AFCEA's International Journal. September 2006.

[9] Highsmith, J., 2001. "History: The Agile Manifesto." Manifesto for Agile Software Development. http://agilemanifesto.org/history.org

[10] Kehler, Robert C., Lt General, USAF. Address to the Armed Forces Communications and Electronics Association. Washington, D.C., 16 June 2006. Transcript: http://www.stratcom.mil/Spch&test/CD_AFCEA_16Jun06.html

[11] "Net-Centric Checklist, Version 2.1.3. May 12, 2004." Retrieved June 3rd, 2006 from http://www.defenselink.mil/nii/org/cio/doc/NetCentric_Checklist_v2-1-3_May12.doc

[12] Paulk, Mark. "Extreme Programming from a CMM Perspective." IEEE Software Nov./Dec. 2001: 19-26.

[13] Potok, T. E. "Extensions to the spiral model to support joint development of complex software systems," Proceedings of the 30th Annual Southeast Regional Conference. April 1992.

[14] U.S. Strategic Command (USSTRATCOM). "US Strategic Command History." http://www.stratcom.mil/about-ch.html

[15] Yin, R.K. Case Study Research: Design and Methods, Sage Publications. Newbury Park. 1984.