17th ICCRTS

"Operationalizing C2 Agility"

Agile Development of Shared Situational Awareness: Two Case Studies in U.S. Air Force and Army

Topic 4: Collaboration, Shared Awareness, and Decision Making
Topic 3: Data, Information and Knowledge

Mark Adkins
Christopher Steinmeyer
William P. Loftus

Accenture
200 Federal Street, Suite 400
Camden, NJ 01803

POC: Mark Adkins
m.adkins@accenture.com
520.906.0432

Agile Development of Shared Situational Awareness: Two Case Studies in U.S. Air Force and Army

Abstract

The world of a fast changing threat, the tradition method of software development is invalid; requirements for unknown threats cannot be identified ahead of time. In fact doing so only gives an advantage to an opponent, since understanding of future threats become the catalyst for an opponent to change strategies to exploit a perceived weakness. In this scenario, the opponent has significantly compromised the decision command process. Using agile software methodology enables the development of C2 to not force requirements too early and project an end position. Agile software methodology is a force multiplier for the Department of Defense effort to create agile command and control (C2) environments. Two case studies show an agile methodology in use in the U.S. Army and U.S. Air Force, each study in different phases of the product life cycle. In the first case the capability to develop an environment with a rapid “smart” index and retrieve is explored. The second case is in its fifth year and uses a large distributed team to develop capability for different elements within the Air Force. A critical area of exploration is to understand how a command and control capability needs to evolve with the requirements of the mission and users. Working with the objective to develop 85% of functionality and to deploy it quickly to multiple user groups who are immediately able to use the capability is achievable. The DoD acquisition community needs to embrace these objectives and develop measures of effectiveness. The warfighters and acquisition professionals have a laudable goal to develop, deploy and integrate capability in a timely manner. Using an agile software develop methodology is a mechanism to develop agile C2.

Keyword Agile methods, Shared Situational Awareness, Command and Control

Introduction

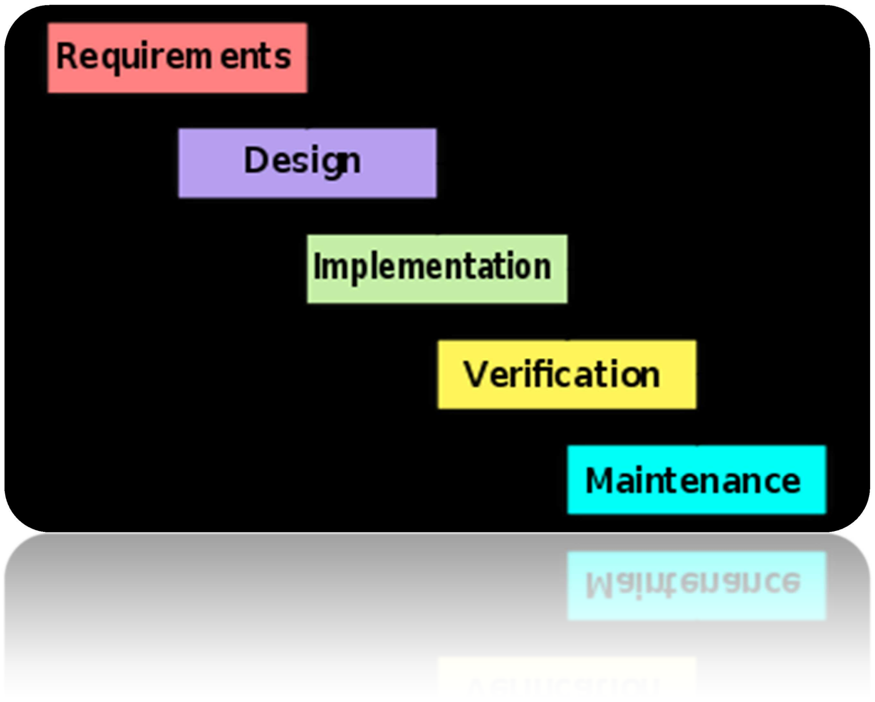
The federal government's major information technology projects are 21 months behind and 12 percent are more than four years late (Government Accountability Office (GAO), 2008). In the last two decades, the U.S. Department of Defense has been transforming to benefit from advanced information technology while challenging acquisition processes. The structure of the United States Air Force is changing with Headquarters United States Air Force Program Directive 06-09, entitled "Implementation of the Chief of Staff of the Air Force Direction to Establish Force Component Organization." The U.S. Army TRADOC Pamphlet (Pam) 525-7-18 is providing 2015-2024 timeframe logistic C2 capabilities. These dynamic changes within the U.S. Army and Air Force illustrate how a family of software development processes that embody principles of the Agile Manifesto bridge the gap between delivery delays reported by the GAO and continuous delivery cycles required by the warfighter in rapidly changing environments.

A recent article (Northern, Mayfield, Benito, & Casagni, 2011) provides a handbook for implementing Agile software development methods in DoD IT Acquisition. A large body of work details various development processes using agile software development methods and argues for one prescription over another (Schwaber & Beedle, 2002, Schwaber, 2007). The intent of this work is to provide a brief overview of agile software development within the context of two Department of Defense environments in the United States then argue agile software develop methods are a key enabler for Agile Command and Control (C2). This article reports five years of experiences using Agile software methods to deliver capability for warfighters at four different United States Air Force Component Number Air Forces (C-NAF) environments and a U.S. Army system operational in the field.

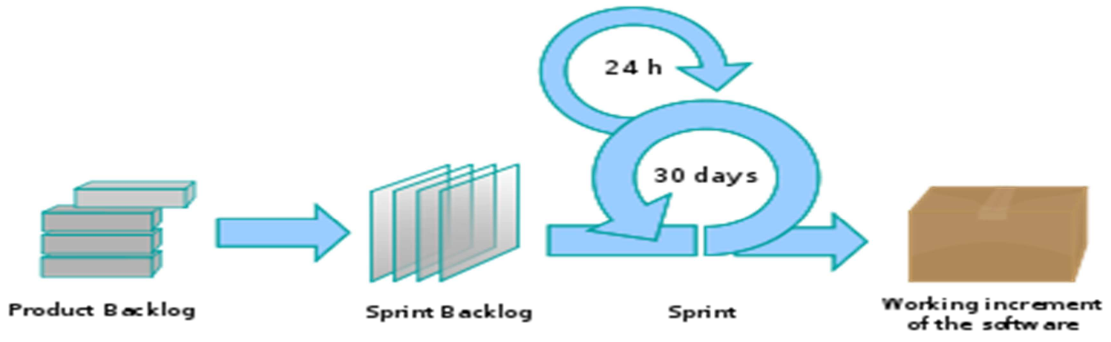
C2 systems are required to enable the commanders to focus on priority issues with appropriate and sufficient information to make decisions quickly while monitoring tasks that the staff is executing. The staff requires enhanced situational awareness and collaboration capability to plan, monitor, and sustain operations. These elements enable a staff to actively think and innovate with a focus on analysis, planning, and execution — not on a battle rhythm synchronized around static tools such as briefing slides and spreadsheets. Delivering situational awareness (SA) capability quickly to meet the warfighter's need can increase the speed of command over a robust, networked grid (Alberts & Hays, 2003). One force multiplier is technology blended with leadership, intelligence and training. The effective use of warfighting tools can achieve a significant strategic advantage and could make the difference between winning and losing battles, in the air, on land and at sea.

Agile Software Develop Methods

Fighting wars of today and tomorrow requires actions that occur concurrently. Agile software development fits the evolving model of agile C2 by providing frequent working software deliveries, allowing for the changing of requirements at any phase of the software development cycle and allowing development team members to work directly with the warfighter. In DoD the guiding principles of Agile software development are that the developers and the warfighter have to work together to produce useable software in short periods of time. These periods for software development are typically two to four weeks of development. Using this agile software development process with the subject matter experts (SMEs) typically gets a level of the capability to use and refine before having to transition out of their job. The Agile development style is counter to the "waterfall" development that better aligns with the typical Plan of Action and Milestone cycle of acquisition typical of the U.S. Department of Defense.



Adkins, Grosse, Baldwin, Coats and Kruse (2008) report a case analysis of the employment of a Network Centric Command Decision Service (netCDS) for 7th and 13th Air Force. This case study expands on Kruse and Adkins (2005) work to focus on how a software development process is operationalizing command and control agility with the U.S. Army and U.S. Air Force. Northern, Mayfield, Benito and Casagni (2011) provide a detailed review of the Agile Manifesto (2002) and a number of agile methods. The focus here is on generic agile methods where product backlog is negotiated with the stakeholders and maintained by the development team solution owner, with sprint backlogs owned by the development team after a selection process and sprints being the execution of capability development in a continuous integration environment. Additional tools used by the teams during sprints to keep down meetings and optimize development time are daily team scrums, twice-weekly scrum of scrum meetings and a series of weekly meetings for program management, senior working group and technical working groups. In addition, at the end of each sprint there is a sprint review where stakeholders within the organization and external to the organization gather using meeting software to share screens and verbalize accomplishments over the sprint. After a sprint review, a retrospective team meeting occurs along with a backlog selection meeting and a task break down meeting for the next sprint.



The amount of effort required to use an agile methodology is huge (Schwaber, 2007), but the transparency of the methodology enables a development team to produce at a rapid pace. Delivering a software product at the end of a 30 day cycle that can be used by a stakeholder in an operational environment is a challenge as the functionality of the software product grows. The following two case studies use a similar approach to an agile methodology to develop two different software products for two different branches of military service. In particular, take note of the factors of Project Team structure, project cadence, tools, stakeholder benefit, quality, risk management and delivery for each of the agile studies.

Using Agile methodologies for the US Army

Project A is a semantic knowledge management and collaboration software application designed to centralize the knowledge capital of our stakeholder's organization as well as facilitate the development of their analytical products. The application recently completed a User Acceptance Test and is targeted for deployment in mid-July of this year. The initial deployment will be on the Non-secure Internet Protocol Router Net (NIPRNET) only and will support roughly 100 users within the stakeholder's organization. There is an expectation, however, that the application will be operational on Secure Internet Protocol Router Net (SIPRNET) and will utilize a multi-tenant architecture that will support several thousand users across the Army by fall of 2013. The Project is in the middle of its second year with follow-on expected for FY13. The project has followed Agile software development methodologies since inception

Team Organization

The project team consists of 3 full-time software engineers (the Team) and 3 part-time managers:

- Project Manager (PM): The PM provides executive oversight over the project. Ideally, the PM takes a very hands-off approach and his/her influence on the project is curtailed.

- Solution Owner (SO): The SO's primary responsibility is to act as the 'stakeholder advocate' for the project. She is responsible for gathering stakeholder requirements, and maintaining those requirements in a prioritized list (the Backlog).
- Scrum Master (SM): The SM is the developers' advocate. He is responsible for tracking the Team's progress during the sprint as well as removing any impediments that may be hindering their progress.
- The Team: The size of the team has fluctuated over the life of the project ranging from 3 to 8 developers depending on the requirements that were being implemented and the delivery schedule. At no time have any two developers or managers resided in the same U.S. City – the team is 100% distributed.

Project Cadence

Like all agile projects, Project A follows a structured schedule of meetings. Scrum meetings – status meetings with the development team and management - occur daily and are time boxed to 15 minutes. During scrum, each 'committed' team member reports on what they did in the last 24 hours, what they will do in the next 24 hours and if there is anything impeding their progress. Sprint Reviews – stakeholder meetings at the end of a development cycle – are held at the stakeholder's site and occur the first Thursday of the month.

Tools

The two primary requirements documents – the project backlog and sprint backlog – are maintained using the Agile project management application Rally, however, Atlassian's GreenHopper was initially used. Rally was chosen over GreenHopper as Rally provided better fidelity around developer time usage and structures the projects more closely to the Agile process adopted within the organization. The increased fidelity allows accurate estimates of development velocity during a sprint based on the time metrics gathered from previous sprints. Hudson is the Continuous Integration environment as it is Open Source, contains a rich plugin library and robust community support.

Stakeholder Benefit

Requirements for DoD software applications with the U.S. Army at this time are in constant flux. Very rarely do the requirements that are laid out at the beginning of a multi-year software project align with the operational requirements of the organization at the time of delivery. Often on major projects using other software development menologies the requirements gathering phase of the project takes a year as there are often multiple site visits, many interviews with subject matter experts that need execution in a coordinated manner that can take considerable time. Once information is acquired an execution process is required to develop a collaborative product of requirements that is vetted among principle stakeholders in a serial manner that takes considerable time and is challenging to get full agreement on before modeling and development can begin.

Operational needs change rapidly and the software design to meet those needs must change equally rapidly to ensure that the solution is as effective as possible. Traditional software development methodologies typically contain multi-month development cycles and can, and often do, diverge from the needs of an organization. Adjusting and re-prioritizing stakeholder needs during a sprint review is a successful method to mitigate risk. Typically, the development goals are no more than 30 days off from the stakeholder's prioritized requirements.

Quality

One of the greatest benefits that Agile methods provide to a stakeholder is an increase in quality over traditional software development techniques. Agile stresses testing early and often in the development lifecycle. This helps to identify defects in the application or architectural shortcomings before they are too entrenched in the application to be easily fixed or are discovered too late in the delivery phase to be fixed without jeopardizing the delivery schedule. Project A utilizes a Continuous Integration (CI) application called Hudson that automatically runs a suite of unit and integration tests on the application with the introduction of new code into the application. This serves to protect the application from the introduction of defects to formerly functioning areas of the application as new features implementation occurs.

Risk Management

Risk mitigation is another area that Project A benefits from as a result of the shortened delivery cycles of an Agile methods' project. Typically with a 30 day development cycle the stakeholder is informed of potential risks and issues before the risk becomes an issue and before the issue becomes unrecoverable or requires significant resource allocation for reparation. On several occasions during the execution of Project A, identifying infrastructure constraints, early on, allowed the team to take corrective action before the risk had a large negative impact on the project.

Delivery

One of the tenants of Agile is to deliver functioning software at the end of each development iteration; for Project A this is every 30 days. Putting functioning software in our stakeholder's hands with each Sprint Review has two major benefits for the stakeholder. First, the generation of a thorough and accurate set of requirements for the next development iteration is possible. Second, the delivery cycle provides an assurance that stakeholder funding is utilized judiciously and the project is on track.

Using Agile methodologies for the U.S. Air Force

Project B is network centric Commander's Decision Services (netCDS) that began using an agile methodology in 2007 for creating capabilities that provide timely access to shared information on the status of forces, systems and supplies from the Wing level up to the Numbered Air Force (NAF) and Coalition (Adkins, Grosse, Baldwin, Coats & Kruse, 2008). Currently, there are several discreet elements of netCDS offered to provide a collaborative view of functions. First, a master events log (MEL) allows critical events tracking by event type and geographic location then assigned out to various staff elements via an alert system. The netCDS MEL capability capitalizes on decades of action research on collaboration in a military context (Briggs, Adkins, Mittleman, Kruse, Miller & Nunamaker, 1999; Adkins, Burgoon & Nunamaker, 2002; Adkins, et al, 2001; Kruse & Adkins, 2005). Second, a "Commander's Situational Awareness" capability allows functional areas to create individual and command level views of various data and information required to operate. A map view provides geographical information while a "Rolling Operational Widget" allows temporal views of various data and information. Conceptually, netCDS focuses on extracting data from services and allowing single point data entry with multiple views. The Project is in the middle of its sixth year and has been following Agile software development methodologies since inception.

Team Organization

Over the last 5 years, the project team has moved from a single small relatively collocated group to three or four groups working with different stakeholders located across five time zones. In total, the whole team has had over 30 members at various times in the projects life cycle. The current team configuration uses a Program Management Group, Senior Working Group, Technical Working Group, Scrum of Scrums and 3 individual scrums to manage the coordination of Project B. Team members execute the following roles during the Sprint Cycle that ranges from 2 weeks to 4 weeks depending on the operational tempo required by the warfighter. At times any one of the three teams could be on a two-week sprint cycle while the other two are on 30 day sprint cycles.

- <u>Project Manager (PM):</u> The PM provides executive oversight over the project. Ideally, the PM takes a very hands-off approach and his/her influence on the project is curtailed.

- <u>Solution Owner (SO):</u> The SO's primary responsibility is to act as the 'stakeholder advocate' for the project. He/she is responsible for gathering stakeholder requirements, maintaining those requirements in a prioritized list (the Backlog).

- <u>Scrum Master (SM):</u> The SM's role is the developers advocate. He/she are responsible for tracking the Team's progress during the sprint as well as removing any impediments that may be hindering their progress.

- <u>Enterprise Architect (EA):</u> The EA's are responsible for the global technical direction of project A.

- <u>Quality Control (QC):</u> The QC's are responsible for planning tests with the teams, executing tests and establishing a timeline for story completion to assure quality.

- <u>The Team:</u> The size of any single team and the combined team has fluctuated over the life of the project ranging from 3-25 developers depending on the requirement execution and the delivery schedule. Developers work in collocated groups and across locations – using a combination of distributed and co-located teams.

Project Cadence

Like all agile projects, Project A follows a structured schedule of meetings. Scrum meetings – status meetings with the development team and management - occur daily and are time boxed to 15 minutes. During scrum, each 'committed' team member reports on what they did in the last 24 hours, what they will do in the next 24 hours and if there is anything impeding their progress. Shared resources who focus on user-interface and database changes work diligently to balance time across teams. Two days a week a "Scrum of Scrums" occurs for coordination across teams as well as a Senior Working Group meeting. Once a week the Technical Working Group meets for one hour and a Program Management meeting occurs.

Sprint Reviews – stakeholder meetings at the end of a development cycle – occur consistently on a specific day of the week at the beginning of the month using Defense Connect Online (DCO) and a teleconference number to include stakeholders in locations around the globe that often cross over seventeen of time zones. After a sprint review, the team takes a few hours for retrospective to discuss what went well and what can be improved with specific actions. Then the team moves into a full backlog selection meeting to choose user stories for the next sprint and wraps up the meeting process with a task break down meeting that includes the majority of the development team. All information discussed during meetings documentation is posted internally using a variety of document management tools.

Tools

The three primary requirements documents – roadmap, the project backlog and sprint backlog – are maintained using the Agile project management application Rally, however, VersionOne was initially used and is still used by a project team to coordinate with external developers. Hudson is used as the continuous integration environment and defects are tracked using Atlassian's JIRA. In addition, the teams capitalize on wiki and chat tools such as, campfire for collaboration and coordination.

Stakeholder Benefit

Decision service capabilities for DoD emerge over time and rarely get defined early in the project. Working on a multi-year, multi-stakeholder software project provides a lot of opportunity for alignment with the operational requirements of the organization at the time of delivery. Operational needs change rapidly and the software design to meet those needs must change equally rapidly to ensure that the solution is as effective as possible. Traditional software development methodologies typically contain multi-month development cycles and can, and often do, diverge from the needs of an organization. Accomplishing risk mitigation by adjusting and re-prioritizing stakeholder needs during a sprint review is successful. Thus, the development goals are no more than 30 days off from the stakeholder's prioritized requirements. By implementing agile processes, stakeholders at the NAF, Wing and squadron level have been able to capitalize on command and control capabilities continuously. Capabilities are fielded while the same individuals who requested the functionality are in position to use the capability and adjust as the unknown becomes known.

Quality

One of the greatest benefits that Agile provides to a stakeholder is an increase in quality over traditional software development techniques. Agile stresses testing early and often in the development lifecycle. This helps to identify defects in the application or architectural shortcomings before they are too entrenched in the application to be easily fixed or are discovered too late in the delivery phase to be fixed without jeopardizing the delivery schedule. Project B utilizes a Continuous Integration (CI) application called Hudson that automatically runs a suite of unit and integration tests on the application with the introduction of new code into the application. This serves to protect the application from the introduction of defects to formerly functioning areas of the application as new features implementation occurs.

Risk Management

Risk mitigation is another area that Project B benefits from as a result of the shortened delivery cycles of an Agile project. With a 30 day development cycle the stakeholder is informed of potential risks and issues before the risk becomes an issue and before the issue becomes unrecoverable. On several occasions, we were able to identify risks, primarily due to infrastructure constraints, early on and take corrective action before the risk had a negative impact on the project.

Delivery

One of the tenants of Agile is to deliver functioning software at the end the development iteration; for Project B this is every 30 days. A release schedule has a deployment team installing operational capability every 30 days across several locations. Deploying functioning software to stakeholders each Sprint has two major benefits for the stakeholder. First, having a capability to use and work with live software in their environment allows them to generate a more thorough and accurate set of needs for the next development iteration. Secondly, the aggressive release schedule provides an assurance resources are executing judiciously and the project is on track.

Limitations

First, these are two case studies so by design there is limited control within the subject environments. Limiting factors due to resources for the team and when resources come available to the teams are beyond the control of the researchers. Second, these two cases were selected with a convenience method. Third, each author is a member of the delivery team and brings bias to the cases.

Future Research

In the future, a structured meta-analysis should be considered as a method to gain knowledge on using agile software development methodology to create agile C2. A rigorous analysis of agile methods needs to be conducted to determine the variations of various agile, scrum, extreme programming, etc. methodologies. With the delineations within the various agile methods, field experiments may be conducted to establish knowledge claims on the effects different agile methodologies can have on outcomes in a variety of project teams.

Conclusion

The thesis of this work is that agile software methodology is a force multiplier for the Department of Defense effort to create agile command and control (C2) environments. The two case studies show an agile methodology used with two different services and in two phases of the product life cycle. A critical area of exploration is to understand how a command and control capability needs to evolve with the requirements of the mission and users. Working with the objective to develop 85% of functionality deployed quickly to multiple user groups who are immediately able to use the capability is achievable. Northern, Mayfield, Benito and Casagni (2011) argue that the acquisition community will need to embrace these objectives and development measure of effectiveness to execute. This is a laudable goal for DoD acquisition professionals and end users with specific needs to develop combat coding capabilities.

In general, over the last thirty years one of the most challenging tasks in creating software has been to get a group of subject matter experts (SMEs) together to develop a set of requirements that all agree on for execution in a timely manner. Then the second challenge is to build the project, test, deploy and integrate before the original SMEs have a permanent change of station (PCS) so you can get feedback on improvements to the capability. Past efforts to tackle this issue have lead to the development of group support systems to facilitate large group agreement on complex ideas (Nunamaker, Dennis, Valacich, Vogel & George, 1991). To benefit the warfighter in a fast pace diversified threat situation flexible software development methodologies need to exploration by acquisition professional and warfighter to create opportunities to develop sustainable agile C2 that is used and reused across the Department of Defense.

Project A has taken great ideas from stakeholders, built a fast prototype environment to solicit additional feedback from more stakeholders and increase user “buy in” for the project. Now the project is moving into a production environment and looking to support a sustainment capability as well as to create new functionality while maintaining a quality controlled capability. As the stakeholder’s mission and focus change as the result of a constantly shifting operational environment the team has been able to deliver high-quality software on a regular basis, keep a tight bound on the gap between the stakeholder’s requirements and the feature set of the application while using successful risk mitigation.

Project B accomplished the same objective as in project A to develop a prototype system then expand capabilities as warfighter needs drove functionality. The project is in the fifth year of development and testimony to the agile development methodology that there is longevity, scalability and flexibility in that several project teams operate on the same code base producing a variety of functionality. The biggest challenges with this project are sustaining quality while driving functionality and meeting an aggressive delivery schedule. Over the last five years the team has proven successful.

The constantly shifting operational environment of the U.S. Army and Air Force provides a rich environment to deliver high-quality software on a regular basis, and to keep a tight bound on the gap between requirements and the feature set of the application. Using the agile methodology to deliver capability to the Air Force each delivery provides 70-90% of what the stakeholder wants for command and control. Adjusting backlog stories will get a full capability within a time-period so a user can see change and impact before they have to move to another job. This development process provides capability that utilizes a refinement process driven by stakeholders.

References

Adkins, M., Kruse, W. J., Damianos, L. E., Brooks, J., Younger, R. E., Rasmussen, E., Rennie, Y. M., Oshika, B., & Nunamaker, J. F., Jr. (2001). Experience using Collaborative Technology with the United Nations and Multi-National Militaries: Rim of the Pacific 2000 Strong Angel Exercise in Humanitarian Assistance. In R. H. Sprague, Jr. (Ed.) Proceedings of the Thirty-Fourth Hawaii International Conference on Systems Sciences, Los Alamitos, CA: IEEE Computer Society Press.

Adkins, M., Burgoon, M., & Nunamaker, J. F., Jr. (2002). Using group support systems for strategic planning with the United States Air Force: The effects of a facilitator's using GSS to structure communication to increase quality output and improve group member satisfaction with the interaction process. Decision Support Systems 34 (3), 315-337.

Adkins, M., Grosse, G., Baldwin, R., Coats, R., & Kruse, W. J. (2008). Network-Centric Command Decision Services (netCDS) for the Component Numbered Air Force (7th Air Force Korea and 13th Air Force). Presented at 13th ICCRTS: C2 for Complex Endeavors, Bellevue, WA http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA486834&Location=U2&doc=GetTRDoc.pdf

Alberts, D.S. Garstka, J.J. & Stein, F.P. (1999). Network centric warfare: Development and leveraging information superiority. Washington, DC: National Defense University Press.

Briggs, R. O., Adkins, M., Mittleman, D. D., Kruse, W. J., Miller, S., & Nunamaker, J. F., Jr. (1999). A technology transition model. Journal of Management Information Systems, 15 (3), 151-195.

Government Accountability Office. (2008). OMB and agencies need to improve planning, management, and oversight of projects totaling billions of dollars. Retrieved from http://www.gao.gov/new.items/d081051t.pdf

Kruse, J. & Adkins, M. (2005). The Technology Trap. Proceedings of the U.S. Naval Institute, 131(8).

Northern, C., Mayfield, K., Benito, R. & Casagni, M. (2011). Handbook for Implementing Agile in Department of Defense Information Technology Acquisition. MITRE Technical Paper. http://www.mitre.org/work/tech_papers/2011/11_0401/

Nunamaker, J. F., Dennis, A. R., Valacich, J. S., Vogel, D. R., George, J. F. (1991). Electronic meeting systems to support group work: Theory and practice at Arizona. Communications of the ACM, 34(7), 40-61.

Schwaber, K. (2007). The enterprise and scrum, Microsoft Press: Redmond, WA

Schwaber, K. & Beedle, M. (2002). Agile software development with scrum. Prentice Hall: New York.

Zelibor, T.E. (December, 2003). 'FORCEnet' is Navy's future: Information sharing from seabed to space. Armed Force Journal, 48-50.