Driving Major Change

The Balance Between Methods and People

David M. Moore, COL, U.S. Army, Project Manager Battle Command

Portia Crowe, U.S. Army, Chief Engineer PM BC-Strategic

Robert Cloutier, Ph.D., Stevens Institute of Technology

Abstract: Successful system of systems interoperability includes a disciplined and responsive system engineering process that focuses on both near-term deliveries to new and current software baselines and longer-term development that sets conditions for enhanced warfighter effectiveness. The foundation of this success relies on flexible and rapid development methodologies and the creation and sustainment of a collaborative social environment by which various communities unify to provide capabilities in a common framework. In the context of new strategy development, the intent of this article is to describe the challenges of implementing an innovative and collaborative environment in the context of scaling an agile system engineering method to a large combined effort.

"Technical problems we can solve; social challenges are much harder." These words of wisdom stated by our Project Manager Battle Command (PM BC) Technical Manager have proven true many times over. The core meaning of this statement is that engineers tend to focus on innovative methods and best practices as a means to increase productivity, reduce defects, increase cycle time, et. al. The most critical processes and methods to success really involve unifying and sustaining a productive social component – a good team with clarity of mission, unity of purpose, and organized to clear objectives.

The backdrop for what has become more of a social endeavor than a technical shift is the Battle Command (BC) "collapse" strategy. The Army has been developing unique and essentially stove-piped digital command and control solutions for many years. Nearly every specific staff function (artillery, air defense, aviation, etc.) has built a unique system for its sole purpose. While it must be noted that Army doctrine drove this design, the sum effect is that our unit commanders and staff are separated by their information systems. The collapse strategy implements a material design approach that brings the Army's family of uniquely distinct tactical functional applications with unique data storing and sharing mechanisms and collapses these systems towards a consolidated software product line. The desired benefits of the collapse strategy beyond the operational warfighter value included reduction in software development and hardware procurement costs.

To launch a strategy about a core product, the BC leadership first had to create conditions for unified effort as well as establish overarching system engineering processes to control progress and gain irreversible momentum. Irreversible momentum is achieved by establishing annual build plans and driving immediate redirection of effort towards these near term goals. Overarching system engineering processes at the PM BC level had to be established in parallel to subordinate project plan adjustments. As part of the BC effort, agile methodologies were adopted and built into the broader organizational culture.

A Shared Innovator's Environment

An aggressive rapid approach is a key measure of keeping our capability relevant with deployed warfighters. One barrier to innovation is a program having centralized control over the development environment. Innovation is more broadly adopted when all can participate with a degree of independence and recognition of shared value by unity of effort.

The main effort of the BC strategy was to build on the most promising software framework within the BC portfolio. This strategy achieved immediate gains but also advanced a limitation that this architecture was proprietary. To mitigate this limitation, PM BC negotiated for government purpose rights (GPR) within the next year. This allowed all developers to maintain the BC software framework. This approach also created the most internal social friction and demanded very deliberate and significant system engineering. Subordinate Project Managers (PMs) and their contractors who needed to shift to this new architecture needed significant training. PM BC continues to mediate between contractors to maintain as much momentum as possible and to keep on the annual build schedules. In time and with the release of GPR, the social friction associated with this course of action should diminish but success has demanded significant leader interaction to maintain support and keep the system engineering process on track.

Similar to each subordinate PM maintaining a thick client system, each PM was also developing unique web service frameworks and a unique presentation layer. To unify this unacceptable condition, the PM chose a new product development effort and built a common web service environment usable at the tactical level that extended the commander's collaborative reach to anyone with a computer and browser ("BC Web"). Select functions are targeted for this environment with the goal of eventually building collaborative capability in a web service environment. To maximize the ability to team with a broad range of development partners, PM BC chose a government open source framework that already had momentum in the Intelligence domain. This choice reinforced unified effort within the Army and created the conditions for any command to build with a program of record in a collaborative development environment.

With the intent to collapse disparate BC thin client systems, a third-party environment became necessary to allow other developers to create, or re-create, their capabilities using a set of standard tools and guidelines in a common framework. The BC Web thin client team stood up an environment that includes a third-party software development kit, third-party widget test and development area using Defense Information Systems Agency (DISA) Rapid Access Computing Environment, authorization and authentication, widget security checklist, and widget and training style guides that collectively provide an integrated secure single environment from development to deployment. The team was able to completely stand up this environment within six months in accordance with the Army CIO/G6 guidance and policies for a common operating environment [1]. One of the main efficiencies of using a common operating environment is that Battle Command, Distributed Common Ground Systems-Army, and now the DISA Joint Command and Control initiative will provide a common core framework and capabilities for broad applicability for enterprise and tactical users, and will allow for optimal sharing of information, infrastructure, and development costs using the Ozone Widget Framework.

Through innovation and placing value on a new product, we are trying to motivate users and developers to create, share, and enhance capabilities and allow for efficiencies in products, services, and processes at a monumental pace, getting users what they need. Through these efforts, our users are seeing similar functionality to common web features (i.e., social networking, mapping features, app store concept). An aggressive strategy, modern technologies, and warfighter needs have imposed a new business model that requires an innovative environment that allows for growth, rapid development and lean testing, integration, and deployment of capabilities. Chris Jones, a widget developer, states, "As a developer, the use of the standard set of tools increases productivity, enables rapid development and deployment of capabilities, and ensures that I maintain the rigors of software governance, test, and validation provided within the environment of BC Web."

Methodology Used

The traditional acquisition process used to develop military technology is not aligned with the speed, agility, and adaptability of new IT capabilities in today's information age [2]. To provide speed of delivery of capabilities to warfighters, we choose to implement an Agile Systems Engineering (ASE) approach that encompasses agile principles as well as brings agility to the entire lifecycle process. Parallel efforts of development, testing and integration with short iterations while stacking priorities are part of the agility structure implemented. The beneficial impact of agile systems engineering is to work smarter and provide immediate benefit and value to the users. It is a highly collaborative method that needs the stakeholders to work together to be successful. The agile systems engineering method values customer interaction and collaboration over process [3].

Future Trends in Systems Engineering
Platform to enterprise (customer emphasis)
Dominant prime to strategic teaming (Execution internal & external)
Compressed delivery schedules
Increased reliance on systems engineering for unknown space
Improvements in collaboration
Increased number of complex systems, emergence and rapid change
Increased customer requests for system engineering support earlier in life cycle
Increased emphasis on users and end value
Understanding of what is attainable

Table 1: Future Trends in Systems Engineering

The BC team understands the challenges of an agile approach from historical knowledge and use in other programs. However, we needed to expand this knowledge to introduce concepts and process to traditional thinkers to invoke the broader community effort. A traditional development approach starts with a defined system with specific functionality as opposed to an ASE approach where adaptive and emergent requirements and system capabilities can be undefined in the beginning and later evolve. Although agile software development is the most popular agile discipline, we needed agility across the spectrum of the program's lifecycle in a rapid and flexible manner. We incorporated ASE as a lightweight loop-back process with short and rapid cycles with priority of requirements and close user and stakeholder collaboration.

ASE offered us a way to incorporate other functions into our 30-day sprint cycle. Within one month, the 20-person contract and government team created enough momentum to demonstrate capabilities at a BC scrum (user and developer integration and feedback) event which included user stories and plans for refinement of capabilities.

Requirements from a much larger community were prioritized and the team was able to complete about 10-15 requirements a sprint. We also worked with the open Ozone community on requirements, standards, and governance. Risks were continuously monitored on a weekly basis during integrated product team meetings. Through these activities, we were able to invoke discipline in our agile processes. Testing and integration were continuously performed on each 30-day sprint build. Through the agile methodology, we had a process that started the security accreditation process early as it is usually the longest lead time. The concurrent planning and execution of security accreditation and training modules earlier than traditional waterfall processes allowed us to provide the system to a beta unit for feedback much earlier than anticipated. A success to rapid widget development in the BC Web environment was a Communications-Electronics Research, Development, and Engineering Center research and development social networking capability that was ready for deployment for the beta unit test. The rapid development of widgets from third-party developers into a common marketplace with no middle integrator enables development of capabilities at a much faster pace with the efficiencies of using common tools. That environment is of high value to our users.

Challenges and Lessons Learned

Creating strategy is nowhere near as hard as implementing strategy. Engineering teams tend to focus on controlling processes, schedule, and risk. The value of a new strategy is only self evident to the creator. All others must be lead along the journey. Leaders and key staff must be given the means to see the vision and work towards a common objective. Solid system engineering processes and scaling up agile methodologies is hard work. Leaders who undertake strategy development must be confident that key leaders and staff who are essential to success will pick up the pace and deliver results that lead subordinates through ambiguous and challenging times.

Judicious selection of software architecture and framework are crucial choices to launch a project with momentum. This paper identifies two key elements of a strategy with two distinct start points. One leveraged a proprietary solution with near-term promise of opening the framework up while the other began open. The more open and ubiquitous an architecture is, the better. The more closed, the greater the challenge to success.

Battle Command's adoption of scrum sessions as a broad means to unify users and developers on common visualizations was essential. Scrum sessions gave a means for project managers and developers to visualize how their traditionally stove-piped software would work in the new environments. Developers were able to derive accurate requirements as a result of scrum sessions. An interesting side effect of scrum sessions was accelerated software development. By putting users and software developers together, management was sidelined. Visible angst existed as both government and commercial managers lost an element of control as these groups became excited and began to turn iterations very quickly. Getting the middleman out of the way at certain times has its benefits.

This is a people business. When change is implemented, people assess their posture against this change and will judge themselves a winner or a loser. They will get on board, actively or passively fight change, or seek a means to ride the fence, ready to shift from side to side depending on their own assessment of probability of success. Dedicated leadership at many levels is needed to overcome these dynamics. Top leaders must engage not only immediate leaders but also engineers and managers at all levels. Leaders must personally communicate the strategy, the plan, and seek feedback at every level. Sitting in the office and publishing implementing e-mails will not lead to long-term success.

At the right time, the leader must get tough. When a risk assessment warrants it, a leader must dive as deep as he can stand to draw out barriers to success. Very often these investigations will not only give technical insight into barriers to success but will also reveal subordinate interactions between government and contractors that may be barriers to success.

Do not jump to adverse social conclusions. Implementing major change is hard. It may be easy at some point to label a key leader or engineer a non-supporter and then seek methods to minimize their adverse impact. In reality, these people are most likely struggling to fit their skills and personality into the new strategy. A personnel change may be warranted if unacceptable risk persists, but this does not mean the person in charge was seeking to undermine the strategy. A leader should seek to align subordinates to their strengths if a change in strategy has marginalized an individual's value.

Nothing beats personal presence. It is a leader's responsibility to put himself in front of his subordinates when he implements change. The leader must be available to take private and public shots from his subordinates. A leader must get out, explain, and internalize how people are feeling about change. The real issues will never come in the leader's office, but brew in the cubicles of subordinates and contractors affected by change. Subordinates may not like what the leader is doing but they will always respect genuine personal engagement.

Innovation is emergent and dynamic and BC realized that it is typically a bottom-up approach in which people involvement is critical. We realized that to gain stakeholder buy-in, the team thrived off of empowerment and involvement in the methodology and process. To overcome cultural challenges, we worked with leadership for buy-in of the agile process that lead to stakeholder ownership of the process and encouraged every member of the team to participate in sprint reviews and creation of the environment. Through this process, we found innovation came naturally and was accepted more openly. The rapid and aggressive approach also brought a higher number of risks than a traditional process, so we had to adjust our tolerance for acceptance and balance it with value to our users. As more developers enter this space with unique requirements, this becomes more complex. Lessons learned also included setting expectations up front for all participants. For example, to minimize costs, the team was asked to maintain a lean attitude up front so that, collectively, we would ensure all IT dollars would provide value. We also found that culture plays a much larger role than technology and can significantly hinder or provide momentum to organizational efficiency and effectiveness. We found that common beliefs and shared logic was very important for success.

Conclusion

The BC collapse strategy is driving significant positive strides that will increase commander and staff operational effectiveness. The strategy's focus on robust, collaborative solutions places Army software development on a path to successfully supporting the warfighter in highly variable and uncertain operational environments. By breaking down system designs that have stove-piped the Army's warfighting functions, employing a human-centered collaborative approach has proven to support the way the commander and staff desire to interact. The condition PM BC seeks to create is one where the strengths of its vendors are not marginalized because of governmental barriers to effective collaboration and open competition. Adopting a commercial competitive model, characterized by rapid cycle times that quickly deliver innovation to the field, is how PM BC programs will remain relevant to the warfighter.

Technical problems can be solved; social problems are much tougher. With any change, new processes must be built or modified and then reinforced. Beneath repeatable system engineering processes and agile methodologies are people. Strategy and its supporting development processes begin and end with people. A leader must ensure his team is well trained, given a clear mission and objectives, and are resourced to execute. The BC software development mission goes one step further as our systems are used in combat. A leader will visibly display this emotional commitment to the warfighter and seek to generate and sustain this commitment in the organization. In Battle Command, this has not been difficult. The unique challenge is convincing the organization that by supporting change, the warfighter will be more lethal and survivable. Software is much about method but in the end, it is mostly about people. People drive success or failure in any organization. Active leadership at all levels drive this success.

REFERENCES

1. US Army CIO G6. Common Operating Environment Architecture. October 2010. <http://ciog6.army.mil/LinkClick.aspx?fileticket=udbujAHXmKO%3D&tabid=79>.
2. Report to Congress. A New Approach for Delivering Information Technology Capabilities in the Department of Defense. Office of the Secretary of Defense. November 2010.
3. Turner, Richard. Toward Agile Systems Engineering Processes. CrossTalk, April 2007.

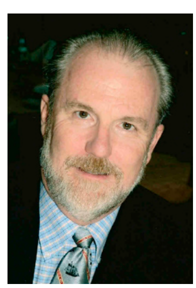
ABOUT THE AUTHORS

COL David Moore assumed command as the PM BC in August 2007. PM BC develops and sustains Army and Joint software command and control capabilities from tactical to strategic levels. The scope of PM BC includes the lifecycle management of software products that include Command Post of the Future, Advanced Field Artillery Data Systems, Global Command and Control System – Army, Battle Command Sustainment and Support System, and a common software effort. PM BC also serves as the Army's Component Program Management Office for Joint Command and Control as well as manages the Army's Common Hardware Systems procurement effort. COL Moore's previous acquisition assignments have focused on warfighting solutions for combat vehicle, soldier equipment, and software command and control systems for the joint force and coalition partners.

PEO C3T PM Battle Command
ATTN: SFAE-C3T-BC
6007 Combat Drive, 5th Floor
APG, MD 21005-1846
E-mail: PAOPEOC3T@conus.army.mil

Portia Crowe is the chief engineer and technical director for the Army, Program Executive Office C3T-PM Battle Command-Strategic. Crowe is a Ph.D. candidate at Stevens Institute of Technology, Hoboken, NJ and currently conducting research in agile systems engineering. She has received numerous Army awards for successful implementation and fielding of strategic programs, and for various research and development projects. Crowe has experience in systems and software engineering and tactical and strategic program lifecycle management.

PEO C3T PM Battle Command
ATTN: SFAE-C3T-BC-TMD
6007 Combat Drive, 5th Floor
APG, MD 21005-1846
E-mail: PAOPEOC3T@conus.army.mil



Robert Cloutier, Ph.D. is an associate professor of systems engineering in the School of Systems and Enterprises at Stevens Institute of Technology. Cloutier has more than 20 years experience in systems engineering and architecting, software engineering, and project management in both commercial and defense industries. Industry roles include lead avionics engineer, chief enterprise architect, lead software engineer, and system architect on a number of efforts and proposals. Cloutier's research interests include model-based systems engineering and systems architecting using Unified and Systems Modeling Languages, reference architectures, systems engineering patterns, and architecture management. Cloutier has a bachelor's degree from the U.S. Naval Academy, an MBA from Eastern College, and a doctorate in systems engineering from the Stevens Institute of Technology.

School of Systems and Enterprises
Stevens Institute of Technology
Hoboken, NJ 07030
Phone: (201) 216-5378
E-mail: robert.cloutier@stevens.edu