Eric Peck, Lynndee Kemmet, Ray McGowan, C. Reed Hodgin, and Bart Peintner

The Agility Imperative
Emerging Knowledge Management Requirements for Stability Operations in the U.S. Army

David Kilcullen said it well in *The Accidental Guerilla,* when he argued that we have entered a new era of warfare in which the "enemy" is often not nation-states but non-state actors who are far more agile and adaptive than large nation-states.[1] As a result, succeeding in this era of warfare will require us to become more agile as well so that we can adapt quickly to changing environments.

U.S. soldiers and their commanders are aware of this need to be flexible. In both Iraq and Afghanistan, they have witnessed firsthand the enemy's ability to adapt quickly to changing circumstances in order to overcome efforts to defeat them. Insurgent groups have shown themselves to be mobile, capable of using simple technology to thwart sophisticated Western technology, and adept at quickly using what information they gather to guide their actions. Unburdened by any rule-bound system of how they must acquire and use resources, insurgent groups are often one step ahead of coalition forces.

As in any conflict, one of the most valuable assets is information. Military forces of most nations, including the U.S., are well aware of this, and the collection and management of information has long been considered critical for the successful planning and execution of missions. Advances in technology have made it possible for modern militaries to collect a mass of data, but process has not kept pace with technological advancement.

Despite the sophisticated new technology available to coalition forces in Iraq and Afghanistan, insurgents still often seem to know things sooner and to act on

Brigadier General Eric Peck is Commander of the Kansas Army National Guard.

Lynndee Kemmet is a researcher and project manager for the ADT project at the Network Science Center at West Point.

Ray McGowan is a Senior Engineer at the U.S. Army Communications-Electronics Research Development Engineering Center.

C. Reed Hodgin is President of AlphaTRAC, Inc.

Bart Peintner is a Senior Computer Scientist at SRI International.

that information faster than the coalition forces fighting them. Collecting information has become much easier, but the "actionable" use of that information remains a challenge. Soldiers and commanders are now overloaded with data, and sifting through it to find what is and is not of value at any given moment is difficult. Moreover, sharing information remains a challenge, partly because of technological barriers between systems and partly because of barriers created by information-sharing policies.

Civilian and military researchers are aware of these challenges and of the urgent need to overcome them. Millions of dollars are poured into efforts to find technical solutions to information overload and to overcome roadblocks to information-sharing. The development process of new technology for the battlefield is too often cumbersome and not related to the needs on the ground. This is particularly true for military units deployed on nontraditional missions. Because these units are still such a small part of the military, there has been little support for creating software programs that improve the process of collecting and sharing data that are useful for economic development and nation-building efforts. However, the need for this sort of information is growing and becoming increasingly important in warfare and economic development.

The U.S. military is quite skilled in traditional warfare, also called kinetic or regular warfare. But, as we have seen in Afghanistan, these traditional approaches to warfare are far less effective when U.S. forces find themselves in the midst of an insurgency where the enemy consists of small, decentralized, mobile groups of insurgents. Counterinsurgency operations have become the norm in Afghanistan, and a large part of this counterinsurgency involves U.S. troops carrying out economic development and nation-building projects. Young soldiers trained for combat have found themselves assigned to such duties as rebuilding economic infrastructure and serving as liaisons to local government councils. These are tasks for which they have little training and for which it is difficult to find information on how to do them most effectively.

Because agriculture is the dominant industry in Afghanistan, the U.S. military sent special National Guard units called Agribusiness Development Teams (ADTs) into Afghanistan with the specific mission of rebuilding that nation's agriculture industry. ADTs are company-size units that include both soldiers who are experts in agriculture and soldiers who are able to provide security for the team. The ADTs are modeled on the National Guard's State Partnership Program (SPP), which has been in operation for two decades. Under the SPP, state National Guards develop partnerships with "sister" countries and provide training and assistance, including development assistance. An added benefit of SPP teams is the positive relationship built between U.S. military personnel and civilian and military personnel of the host nation.

ADTs seem to be an effective model for the U.S. military units that provide development and nation-building assistance in highly unstable and underdeveloped nations. But ADT commanders have made it clear that the model has knowledge management challenges that must be overcome. For example, these units have been hampered by the inability to share information with their civilian counterparts on development efforts. Even though information on ADT projects is mostly unclassified, it is put into classified military systems; once there it cannot be shared, even with civilian personnel of U.S. government agencies, such as the U.S. Department of Agriculture and USAID. It most certainly cannot be shared with Afghan government officials or international nongovernmental organizations on the ground, and this lack of information-sharing makes the coordination of civil and military efforts nearly impossible. Such coordination is vital in enabling the military to eventually transfer development efforts to the civilian sector.

Information on the projects and efforts of military teams engaged in development efforts is also not easily shared across military units, or from an outgoing ADT to its replacement ADT. This hinders consistency in development efforts. It also means that the lessons learned by one ADT in terms of which strategies are most effective on the ground are lost, rather than being shared with other military units. In addition, valuable information collected by ADTs and units on similar missions is not shared with researchers, which prevents them from conducting analyses that could lead to better strategies for these teams.

The data collection and management software programs developed for ADTs will help resolve the problems described above. But because ADTs are already on the ground and need this information support now, the normal defense department process of developing and fielding new technologies will likely be of little help to ADTs. That process can take years to put new technologies into the hands of soldiers; moreover, most current information technologies in the military are aimed at units on traditional military missions. This essentially means that most information technologies focus on the collection and management of classified data without concern for the need to share information with civilian counterparts.

Information technologies for ADTs must be developed quickly and placed in the hands of soldiers as rapidly as possible. These software programs also must be flexible so they can easily be modified to meet the varying needs of development-type missions. Developers therefore must work closely with ADT personnel to ensure that what they create actually meets the needs of the ADTs.

In the traditional technology development process, technology developers, whether civilian or military, usually design systems to address what they see as a problem, not necessarily what soldiers have told them is a problem. Developers then take their new creation and “toss it over the wall” to soldiers, who often toss it to the side because it was designed for the wrong problem or it addresses the right problem but in the wrong way. Moreover, because the development-to-deployment process is so long, when a new technology reaches soldiers, it often turns out that the problem it was designed to address no longer exists or has changed.

A more effective approach, which was used in the case described in this article, is agile development. ADTs face the challenge of operating in rapidly changing environments and therefore need timely and relevant information for mission planning. Agile development, as described in the 2001 *Manifesto for Agile Software*

Development, is essentially a process that emphasizes interaction between the developer and the user.[2] Developers remain open to adapting a design based on user feedback. This process should be rapid: users provide feedback, then designers make quick adjustments and return the adjusted product to users for additional feedback. This process can reduce the time between development and deployment of new technologies from several years to one year or even, as in the case studies discussed below, to less than a year. Because users are part of this development process, the technology they receive meets their needs more effectively. If the users' needs change, the developers quickly modify the technology to meet that new need.

With funding from the Defense Advanced Research Projects Agency and the office of the secretary of defense, and under the guidance of the ADT Project at the Network Science Center at West Point and the Communications-Electronics Research Development Engineering Center, two technologies were developed for specific use by units on ADT-type reconstruction missions. Both technologies were developed with constant interaction and feedback from ADT users, and both were in the field within the first year of development, a rapidity made possible by leveraging previous technologies and responding rapidly to users' needs.

These two technologies—Mobile Task Assistant and XCapture—help soldiers collect data about their missions. In the case of ADTs, this includes collecting information on development projects and training programs conducted to help Afghan farmers. These technologies make it easier for soldiers to share this information with other military units, and with civilian government officials and non-governmental organizations. This information-sharing allows lessons learned about how to approach stability and how to conduct reconstruction/development operations to be passed from one military unit to another. It also improves civil/military coordination of reconstruction efforts. Sharing lessons learned and removing information barriers to civil/military coordination will go far in helping unstable and underdeveloped regions get back on their feet.

NATIONAL GUARD AGRIBUSINESS DEVELOPMENT TEAM KNOWLEDGE MANAGEMENT SYSTEM: A CASE STUDY OF AGILE SUPPORT FOR NON-KINETIC EFFECTS

One of the emerging challenges in military mission command is the ability to use the available data and information across missions and issue areas to support commanders' decisionmaking. The "data-to-decision" paradigm refers to this ability to use data and information to make decisions. The current paradigm is not perfect, but it is effective when applied to most traditional war fighting conditions. This is because the user community and the subject-matter experts understand what critical information is needed to support their missions and their workflow. They also understand how these information requirements serve as either lagging or leading indicators of how successfully they are achieving their objectives. In some cases, this information might include knowledge of how these indicators tie to outcomes within a particular mission or campaign plan when it is a kinetic (tactical) mission.

The picture is not as clear on the non-kinetic side of tactical engagements. In most areas that fall outside the tactical intelligence domain, which includes people and infrastructure, the U.S. military has only a small knowledge base. The military needs a wider range of information in order to understand and address the complex interactions and relationships that occur among the entities in the political, military, economic, social, infrastructure, and information domain. One reason for this is that irregular warfare, meaning such things as counterinsurgency and stability operations, has forced the U.S. Army to support missions such as economic development and nation-building, which traditionally would be performed by government agencies like the state department or USAID. No civilian agency could hope to safely undertake reconstruction work in Afghanistan, for example, because of the unstable conditions. Although tasked with this reconstruction mission, the military has neither the information nor the tools to carry out critical tasks.

This case study addresses how this challenge might be overcome. It describes the application of the agile systems engineering approach to the development of knowledge management/information systems designed for use by ADTs in Afghanistan. This case shows the benefits of this innovative approach in developing and applying technologies that can help military teams involved in non-kinetic missions. Much has been learned in this particular case about how to apply an agile development approach most effectively.

AGILE SYSTEMS ENGINEERING

Agile software development is one aspect of the larger agile systems engineering process. The *Manifesto for Agile Software Development* details the guidelines used by agile software developers, and focuses on the following values:

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

These values apply directly to any effort in which the primary objective is to produce technology rapidly for a specific customer. The process works when the customer—in this case the ADT—is dedicated, knowledgeable, and can prioritize the functions that have the most value. Moreover, it results in architecture designed to meet up-to-date requirements in an environment where the systems are largely emerging and can change rapidly. These values lead to the 12 principles that guide agile development:

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- We welcome changing requirements, even late in development. Agile processes harness change to the customer’s competitive advantage.

- We rapidly deliver working software from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers work together daily throughout the project.
- We build projects around motivated individuals, giving them the environment and support they need and trusting them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a rapid pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity, the art of maximizing the amount of work not done, is essential.
- The best architecture, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then adjusts its behavior accordingly.

Applying the agile manifesto to systems engineering means increasing the speed with which new products and systems are designed and delivered to the customer. It also entails addressing the uncertainty in future user needs, operating conditions, and functional requirements.[3] Developing information technologies to meet the needs of ADTs was an ideal opportunity to test this innovative approach to technology development and deployment.

The development-focused mission of ADTs is different from traditional software development processes found in the agile manifesto, in that ADTs primarily collect unclassified information on their reconstruction efforts, and they need to share that information with other military units and civilian counterparts that also are working to rebuild Afghanistan. Most military information technologies are not designed to deal with unclassified information, and definitely are not designed to share information. As mentioned above, two technologies were developed specifically for the ADT Project. One is XCapture, an automated after action review technology created by AlphaTRAC, Inc., that gathers information on missions, including lessons learned. The other is Mobile Task Assistant (mTA), developed by SRI International, which focuses on collecting and sharing information on projects undertaken by ADTs. These two technologies will make it easier for ADTs to share information, and the information collected will be used to create a rich database on ADT missions. This database will make it possible for researchers to analyze and learn from the ADT missions, and thus to improve future strategies for military teams involved in stability and reconstruction efforts. Application of the agile development process to the development of XCapture and mTA is described below.

XCapture

XCapture is an automated after action review technology that is designed to improve the information collected about military missions in order to improve future missions. This automatic system for creating reports gathers and compares what was supposed to happen on a mission and report what really occurred. The system asks what lessons were learned, and then archives the information in a way that makes it useful for the development of artificial intelligence, training, and operational support. It is designed to create reports automatically and in various formats.

The first step in the development of XCapture was to build on an existing technology known as AlphaACT, which was the result of a Defense Advanced Research Projects Agency SBIR project conducted by AlphaTRAC. It is web-based software that goes beyond data management to support and improve the actual decision-making process. The system uses artificial intelligence to capture and use the experiences of successful decisionmakers. AlphaACT also creates a communitywide knowledge base, which increases the sharing of knowledge and experiences among decisionmakers. AlphaACT is currently being used for emergency response training and is being adapted for small unit leader decisionmaking training.

Experience in commercial software development over the last 15 years has shown that an adaptive approach helps ensure both project and end-user satisfaction. The agile development methodology has been shown to be most effective in this environment. In the agile development process, "user stories" first establish the needs and requirements of product users and ultimately drive product development and modifications. They do so through what is termed a "scrum process," which involves a series of cycles (called sprints) that last two to four weeks, during which a number of user stories are selected for development and testing. The process allows the development team to respond quickly to emerging and evolving product needs and to address bugs quickly and methodically.

Each sprint involves a team working through a full software development cycle. The cycle includes planning, analysis of requirements, design, coding, unit testing, and acceptance testing, the latter being when a working product is demonstrated to stakeholders. The agile development process minimizes overall risk by identifying and adapting to evolving user needs and allows the project to adapt to changes quickly. Team composition for the XCapture development project is cross-functional and self-organizing. Team members take responsibility for tasks that deliver the functionality a sprint requires.

The XCapture team includes multiple customer representatives, such as National Guard soldiers and representatives of the Agribusiness Development Team program who make a personal commitment to be available to developers whenever problems or questions arise throughout the process. At the end of each sprint, the customer representatives review progress and reevaluate priorities.

The XCapture development team collaborates with the full ADT unit in two ways. First, intensive training, testing, and feedback sessions take place during

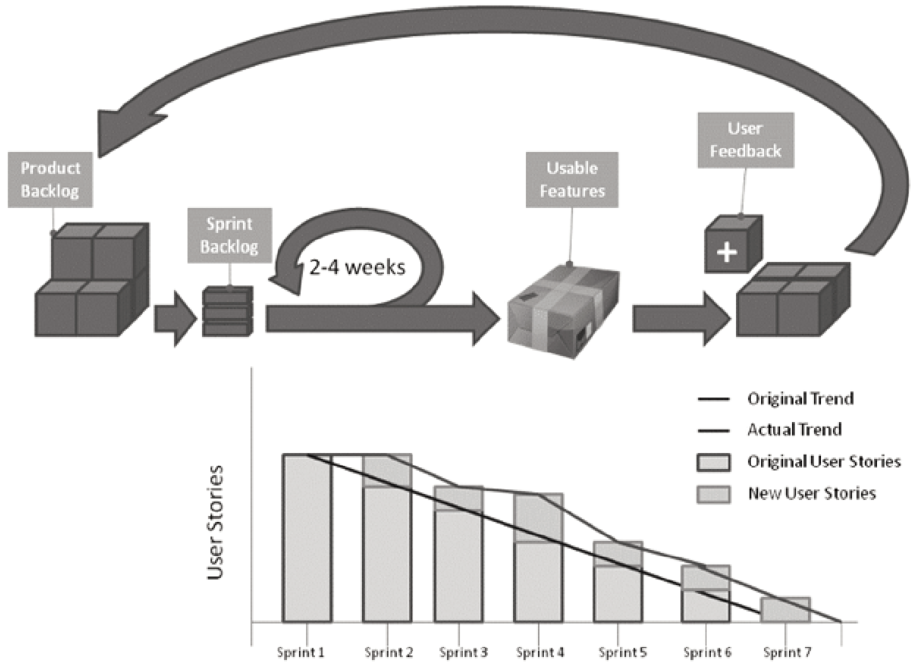


Figure 1. The Agile Product Development Process

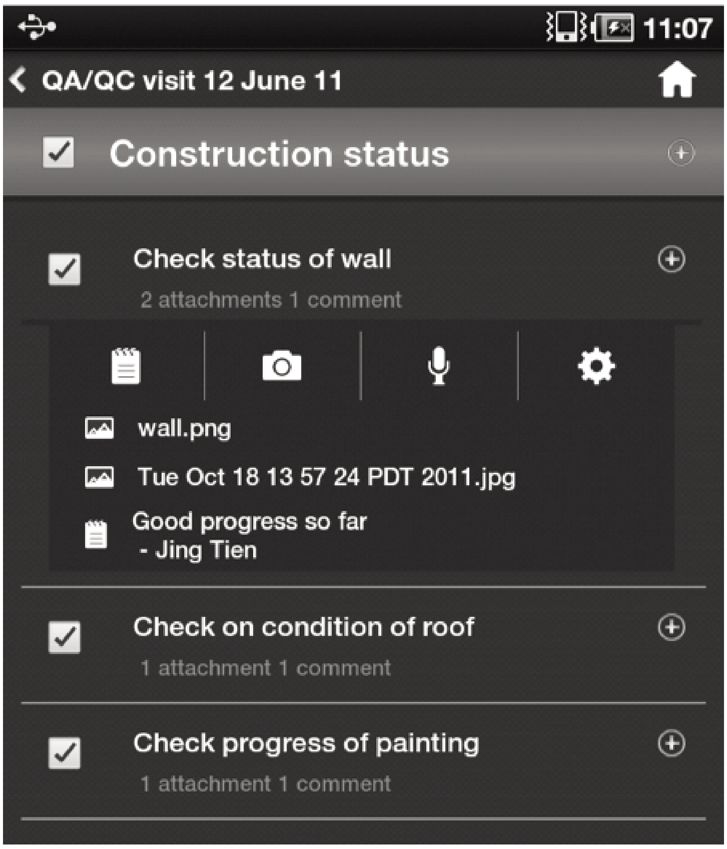
workshops that take place periodically at ADT home station or mobilization sites. Second, soldiers at the home station and in the field conduct ongoing application testing, all the while providing feedback through email and internal social forums.

The XCapture development process begins with a sprint kickoff session. The tasks to be accomplished in each sprint are based on the stakeholders' needs as they are defined at the beginning of the sprint. These needs are expressed as user stories, which essentially serve as user requirements during the agile development process. Stakeholders include the intended XCapture end users and other interested parties, such as leaders, managers, security personnel, infrastructure managers, third-party users, system administrators, and even the development team itself.

During the kickoff session, the team chooses user stories that will be addressed during the sprint and assigns a task leader and task team for each user story. Addressing some user stories may require more than a single sprint. These stories are termed "epics" and are broken into sub-units, each of which can be addressed in a single sprint.

The team meets daily during the sprint. The customer representative and interested stakeholders can observe at these meetings, during which team members report what they did the previous day, what they intend to do today, and what their roadblocks are—a process that exposes problems as they arise.

Each sprint culminates in a sprint review that features the product modifications or extensions for each user story. Unit and integrated testing data and results


11:07

QA/QC visit 12 June 11

Construction status

Check status of wall
2 attachments 1 comment

wall.png
Tue Oct 18 13 57 24 PDT 2011.jpg
Good progress so far
- Jing Tien

Check on condition of roof
1 attachment 1 comment

Check progress of painting
1 attachment 1 comment

Figure 2. Mobile Task Assistant

are also presented. The customer representatives review each modification or extension and declare whether or not the user's need was met to their satisfaction. The user story is then either retired or goes into further development. The development cycle continues with a kickoff for the next sprint.

In addition to the testing done within each sprint, the entire team breaks from the development process to test the system thoroughly after each major release of product improvements. The agile development process as applied to the XCapture project is illustrated in Figure 1 (previous page).

Table 1 (following page) presents an illustrative sample of the initial user stories that were generated to begin the XCapture development project in June 2011.

MOBILE TASK ASSISTANT

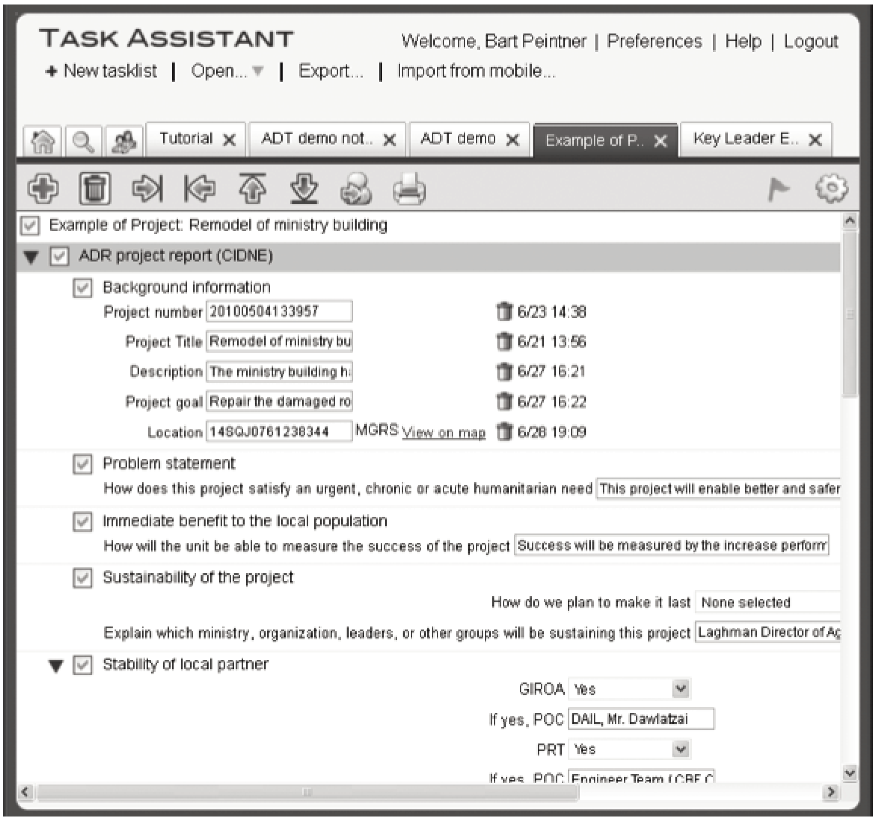
Mobile Task Assistant (mTA) supports ADTs in developing and managing agribusiness projects by facilitating data collection, data sharing, and report generation. This Android-based mTA app was developed over the course of eight months using the agile process described above. mTA is the mobile companion to the (pre-

Type	Time frame	Priority	Description
Epic	Mid-term	High	As an ADT member, I need the AAR tool to quickly capture lessons learned from an ADT mission, so I can learn from a mission and not repeat mistakes.
Story	Mid-term	High	As an ADT member, I need the AAR tool to quickly capture the mission plan/order, so I can understand what was supposed to happen on a mission and find AARs relevant to my mission.
Story	Mid-term	High	As an ADT member, I need the AAR tool to quickly capture a description of an ADT mission, so I can understand what really happened on a mission and find AARs relevant to my mission.
Epic	Mid-term	High	As an analyst/planner, I need the AAR tool to archive mission information and lessons learned in an accessible, minable database, so I can easily access, mine, organize, and use the information.
Epic	Mid-term	High	As an ADT Leader, I need the AAR tool to make the information immediately available, so the information can be used for upcoming mission planning and team workups.
Epic	Mid-term	High	As an ADT member, I need the AAR tool to be easy to use in the warfighters' environment—whether that is the field, a forward operations base, a rear base, or a home station—so a warfighter will be able to and want to conduct the AAR.
Epic	Long	Medium	As an ADT Leader, I need the AAR tool to be adaptable to an individual unit's use, so the team can do things its way.
Epic	Mid-term	High	As an ADT Leader, I need the AAR tool to coach the team and team leader through the AAR in real time, so the AAR is consistent, complete, and rapidly developed.
Epic	Mid-term	High	As an ADT member, I need the AAR tool to incorporate new knowledge (especially free form input) into the system right away, so new knowledge and lessons learned can bring value to the warfighter right away.
Epic	Mid-term	High	As an ADT Leader, I need the AAR tool to generate standardized AAR reports, so the reports can be distributed and used within existing systems and infrastructure.

Table 1. Sample of initial user stories that were generated to begin the XCapture development project in June 2011

existing) Task Assistant system, a web-based collaborative task management technology that helps teams coordinate efforts, gather and share data, and disseminate knowledge and procedures.

Along with its web-based counterpart, mTA helps ADT team members capture notes, images, audio recordings, and GPS locations more efficiently, and then to organize that information and make it accessible to both the current team and future units. At the base, soldiers develop forms and instructional guidelines using the web-based interface. The forms and guidelines are then downloaded onto the



TASK ASSISTANT

Welcome, Bart Peintner | Preferences | Help | Logout

+ New tasklist | Open... | Export... | Import from mobile...

Tutorial × | ADT demo not.. × | ADT demo × | Example of P.. × | Key Leader E.. ×

☑ Example of Project: Remodel of ministry building

☑ ADR project report (CIDNE)

☑ Background information

Project number	20100504133957	6/23 14:38
Project Title	Remodel of ministry bu	6/21 13:56
Description	The ministry building h	6/27 16:21
Project goal	Repair the damaged ro	6/27 16:22
Location	14SQJ0761238344 MGRS View on map	6/28 19:09

☑ Problem statement

How does this project satisfy an urgent, chronic or acute humanitarian need: This project will enable better and safer

☑ Immediate benefit to the local population

How will the unit be able to measure the success of the project: Success will be measured by the increase perform

☑ Sustainability of the project

How do we plan to make it last: None selected

Explain which ministry, organization, leaders, or other groups will be sustaining this project: Laghman Director of Ag

☑ Stability of local partner

GIROA: Yes

If yes, POC: DAIL, Mr. Dawlatzai

PRT: Yes

If yes, POC: Engineer Team (CBE C

Figure 3. Web-based Task Assistant view of a mission tasklist for the ADTs

Android devices for use in the field. When returning from the field, soldiers fill out forms and provide images, and sound recordings. These items are merged into the shared Task Assistant database, which is then used to generate reports and support future field missions.

Need for agile development

The compressed timeline of less than one year to deploy this new technology necessitated the agile development approach. The classical approach of gathering requirements, developing the software, then throwing it over the wall was infeasible for three main reasons. First, gathering requirements for the software required interaction with several National Guard units, some of which were deployed and therefore initially unavailable, some of which were too inexperienced at the start of the project to give valuable feedback, and others that had intense pre-deployment training schedules that limited our access.

Second, the utility of the software depended primarily on how easily it fit into the workflow of the mobile teams. ADT teams cannot afford to spend their time interacting with the software and device—the technology had to perform its functions without compromising the personal relationships between the team and local residents. Therefore, the design and development process had to center on eliciting design-enabling knowledge from the target users.

Finally, the people driving the project had not fully understood the concept of operations for the software and device. Arguably, this is true for nearly all "ground-up" software development, which is a primary cause of failure for the waterfall method of product development. The mTA app represented a substantial change in team operations, in that its effects and primary values could not be anticipated, even by experienced domain experts. Furthermore, just as technology experts do not initially understand the domain, domain experts do not initially understand what is possible in software development. Hence, it is difficult to develop a product vision that fits into both time and resource constraints, and the vision must adapt over time. Seeing interim versions of the product shows the domain expert what is possible and helps generate new ideas.

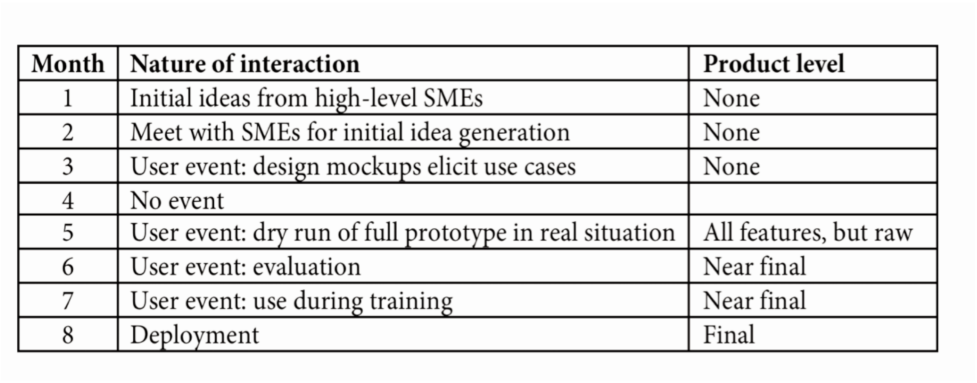
For the reasons just mentioned, the development and design processes necessarily began before the first "requirements gathering" user event, and therefore well before developers had a complete picture of the eventual uses for the software. The lack of a complete picture had a strong effect on the order and priorities of the initial development efforts. The team's initial development efforts focused on the underlying client-server interactions and the internal structure of the app. The lack of a fully specified design actually promoted an agile-friendly design environment; the infrastructure that was first developed supported a wide range of layouts and navigation strategies.

Based on experiences on other Personal Assistant that Learns or PAL programs, developers had been working successfully with users to define the concept of operations in their domains. Initial meetings were held with the ADT members to understand their roles and to begin to develop the concept of operations. Developers had a series of meetings with ADT members returning from theater, ADT members preparing for their second deployments, and ADT members who were just beginning to train for their first deployment.

Tailoring the standard agile methodology to the situation

While the initial infrastructure was being developed, design efforts focused on mocking up several ideas to present at the first meeting with the user. The first user meeting did not require a prototype, but developers believed that presenting a variety of highly realistic mockups, along with demos of the existing web-based tool, would elicit more valuable information than a single real but highly limited example of working code. This first meeting gave developers the input they needed to commit to a first design of the layout and navigation.

Developers then tailored the standard agile process in one other significant way: they did not hold the typical 15-minute weekly meetings, as Task Assistant provides collaboration and task management services that serve the same function as these meetings. Using the tool, developers efficiently tracked who was doing



Month	Nature of interaction	Product level
1	Initial ideas from high-level SMEs	None
2	Meet with SMEs for initial idea generation	None
3	User event: design mockups elicit use cases	None
4	No event	
5	User event: dry run of full prototype in real situation	All features, but raw
6	User event: evaluation	Near final
7	User event: use during training	Near final
8	Deployment	Final

Table 2. ADT User Events

what, gathered ideas for improvements, reviewed requirements elicited at the last user event, defined short-term goals, and reported on the results of testing efforts. As a result, almost no time was spent in meetings coordinating, and meeting time was used instead to discuss user feedback, brainstorm solutions, and argue for different technical approaches—all of which moved the product forward quickly.

Iterations and ADT user events

Developers were fortunate to go through several iterations with the ADT users—a surprising number in fact, given the schedules and training demands of recently deployed and soon-to-be-deployed soldiers. Each of the iterations consisted of a user event, a goal-setting session, and a development sprint.

Table 2 outlines the nature of each user event and the level of prototype available for each. After each event, developers updated the design, reprioritized features, and set goals for the next release. Several factors influenced which improvements to make between events:

- **Level of uncertainty.** Uncertainty in the value or design of a feature is cause for inclusion in the next user event, given that it may be necessary to maximize the number of remaining iterations to get the feature right.
- **Difficulty of implementation.** Complex elements of the product require more testing and user time to get issues ironed out.
- **Visibility of problem or solution.** When users perceive improvements in the product from session to session, and when they understand that their feedback finds expression in the product, their feedback improves in quantity and level of insight. Conversely, an unaddressed deficiency that users perceive as major reduces their faith in the process and the development team.

The natural progression of the user events tends toward an increasingly capable product, an increasingly refined concept of operations, and a general slowing of the rate of change in each. In the case of mTA, one or two additional iterations would have significantly improved the initial delivery. These iterations have continued post-deployment, as developers continue to improve the system based on feedback from the field.

Results

This agile development process quickly produced a working prototype. Although technical and usability issues remained at the time of delivery, initial reports from the field, which outlined the positives and negatives of their experience with mTA, showed clearly that the ADT members were pleased with the development process and were eager to continue it. Developers have continued to produce iterations, sending new releases to the field every few months.

The agile process uncovered several non-software issues that had to be addressed to enable effective use. A lot of time was spent exploring such things as gloves, rugged cases, and input devices—for example, keyboard, stylus, and pouches—that had to be tried by a number of soldiers before something workable was found.

Other issues stemmed from limitations in the field, such as the lack of wireless connectivity, the need for and the challenge in creating hard cases for the device, and the wide variety of lighting conditions. Early interaction with the users forced developers to tackle these issues early enough to be able to design a solution into the final product. The lack of wireless connectivity on base, for example, added a significant technical hurdle that required several months to resolve.

The final version of the app differs significantly from the initial design. Some features originally marked as "must have" did not make it into the system because no user session ever showed the need for them. Implementing those features would have taken time away from higher priority tasks and cluttered the interface, making it harder to learn to use. In addition, features that were secondary and somewhat hidden in the web interface (e.g., displaying a location field) ended up being prominent, easily created features of mTA. All of these differences point to the value of the iterative, agile process. Using the classic development process, few challenges would have been identified or solved by the time users required.

CONCLUSION

Over the past decade, we have overcome many planning, execution, and informational challenges that we faced when conducting both stability and response-and-recovery operations, but many challenges remain in terms of improving collaboration between military and civilian government agencies, and between those agencies and NGOs and the private sector. Particularly challenging in both stability operations and response-and-recovery operations is the impact of our inability to share information in real time, or close to real time. We face policy, procedural, and technology barriers, in addition to the normally degraded infrastructure that accompanies both situations. The ability to address the barriers and challenges faced in these complex situations will require our continued efforts to reduce the time it takes to produce solutions by applying the agile development process to more than just technology solutions.

Traditional technology development followed a classical format that involved creating requirements—often without input from end users—building technology to meet those requirements, tossing the new technology "over the wall" to soldiers, and then leaving them to "like it or lump it." The typical result has been the development of technologies that users neither liked nor needed. Or, more accurately, the development of technologies no longer needed because the situation had changed long before the technology ever made it from drawing room to the field.

The agile development process takes a different view, which essentially is that the end user—in this case the warfighter—is all that matters, not what the technology developers think they want or should want. The agile approach requires constant interaction with users during the development process, and making constant adjustments as the needs of the user change. As this case study of XCapture and mTA show, when effectively applied, the agile development approach can lead not only to faster deployment of new technologies but also to the creation of technologies that meet users' needs more effectively, are viewed more favorably, and therefore are used more widely by warfighters.

The success of this approach in developing technologies that actually provide benefits to soldiers on ADT reconstruction missions provides lessons that leaders in both military organizations and civilian agencies should learn. Far too often, managers at the top of organizations have no real understanding of the situation on the ground or of the needs of their personnel who are out there conducting operations, whether they be stability operations or disaster response and recovery. Information-sharing, both across organizations and among the levels within an organization, is critical for coordinating efforts.

What is needed, then, is a new approach that supports the development of technologies and the adoption of policies that can improve information-sharing and coordination. This approach must force managers at the top to allow those on the ground a greater voice in decisions, not only regarding the development of technologies that meet their needs but in changing policies in ways that can improve their ability to share information and coordinate with others on the ground. This case study shows how quickly needs can be met when leaders at the top of organizations, end users on the ground, and developers engage in constant dialogue in which each listens to and understands the needs of the others.

---

1. Kilcullen, David. *The Accidental Guerilla: Fighting Small Wars in the Midst of a Big One*. Oxford, England: Oxford University Press, 2009.
2. Manifesto for Agile Software Development, http://agilemanifesto.org. See also Lingfeng Wang, "Agility counts in developing small-size software," *Potentials*, IEEE, vol. 26, no. 6 (Nov.-Dec. 2007): 16-23.
3. D. Oxenham. "Agile approaches to meet complex system of system engineering challenges: A Defence Perspective," System of Systems Engineering (SoSE), 2010 5th International Conference, pp. 1-6, 22-24, June 2010. V. Rahimian, and R. Ramsin, "Designing an agile methodology for

mobile software development: A hybrid method engineering approach," Research Challenges in Information Science, 2008. Second International Conference, June 2008; A.M. Sen and K. Hemachandran, "Elicitation of Goals in Requirements Engineering Using Agile Methods," Computer Software and Applications Conference Workshops, 2010 IEEE 34th Annual, July 2010.