2

Acquisition & Management Concerns for Agile Use in Government Series

Agile Culture in the DoD

Software Engineering Institute
Carnegie Mellon University

Acquisition & Management Concerns for Agile Use in Government

This booklet is part of a series based on material originally published in a 2011 report titled *Agile Methods: Selected DoD Management and Acquisition Concerns* (CMU/SEI-2011-TN-002).

The material has been slightly updated and modified for stand-alone publication.

Booklet 1: Agile Development and DoD Acquisitions

Booklet 2: Agile Culture in the DoD

Booklet 3: Management and Contracting Practices for Agile Programs

Booklet 4: Agile Acquisition and Milestone Reviews

Booklet 5: Estimating in Agile Acquisition

Booklet 6: Adopting Agile in DoD IT Acquisitions

Agile Culture in the DoD

Every organization has common things that are visible and reflect its basic assumptions, shared values, espoused values, and artifacts as shown in Figure 1. Together these form a view of the organization's culture [Schein 2009]. Artifacts are the most visible elements of a culture. They include the products of the groups and everything an organization experiences, from the facilities and properties of the physical environment to the group's language, reports and documents, posters and presentation materials, myths, stories, and rituals.

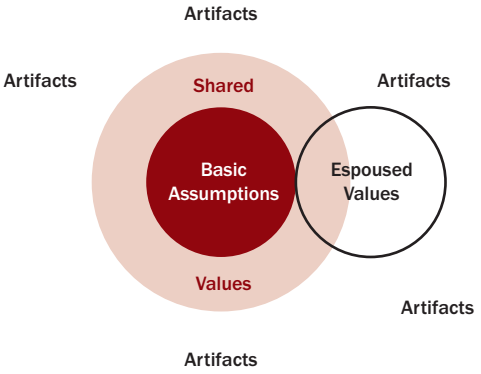

Figure 1: Key Elements of Culture[1]

"Information radiators" are a good example of an artifact of Agile culture. The big, visible version of these charts can take many forms, for example: a storyboard of user stories that highlights the iteration's status, burn down charts that show the number of requirements and stories that are completed, status of the continuous integration builds, roadblock charts that show obstacles and who has responsibility for working these, and so on. Distributed teams also need information radiators.

One of our reviewers commented, "A large flat-panel screen in each lab/team room is a wonderful information radiator for distributed teams." In an Agile culture, it is critical that the artifacts are dynamic, actionable expressions of the project at work, not static documents (e.g., project mission statements that are not expected to change frequently). Cultural artifacts are easy to see but can be hard to interpret, and they alone do not represent the essence of a culture.

The Agile Manifesto is a good example of the espoused values of the Agile Alliance, the originators of the term "Agile methods," and it generally reflects the values to which Agile teams aspire. Espoused values (things we say we do), norms, and rules make up the everyday operations of a workplace. Espoused values become shared when they have been proven as solutions and the group accepts the solution's value. Solutions (actions and values) that are repeatedly used with success eventually become basic assumptions—general inferences that are not grounded in substantiated research but are unquestionable [Schein 2009].

[1] Adapted from Edgar Schein's Organizational Culture and Leadership, 1992

For example, continuous integration may start as an espoused value in an Agile organization (this is a rule we will follow), but until there are sufficient tooling mechanisms to support continuous integration, it will not become a shared value, a common practice that most of the organization would not readily abandon. Should continuous integration become automatically accepted within the group without its use being discussable, it would become a basic assumption for that particular culture.

The relationship between values and assumptions is important. "If the espoused values are reasonably congruent with the underlying assumptions, then the articulation of those values… can be helpful in bringing the group together, serving as a source of identity and core mission." [Schein 2009] In situations where there is tight coupling or congruency, the Agile cultural identity will be very strong. In those situations where there is less congruence, evidenced by more explanation, rationalization, and aspiration around the values, the Agile cultural identity will be weaker.

In some of the organizations we interviewed, we saw evidence of strong shared values related to Agile principles, particularly in programs where Agile had been successfully used for more than a couple of years. We saw other cases, particularly where the Agile adoption was relatively recent, of a weaker Agile cultural identity. In one program, with a strong Agile identity, prospective employees are likely to be interviewed by all their potential team members, not just the team lead. The team makes a judgment as to the fit of the employee not only in terms of software development skills, but also in terms of their perceived ability and willingness to adopt pair programming and test-driven development practices; both of these are anchors of Agile practices in that organization.

In a more traditional project, full team involvement in interviewing and deciding on new members is not as common an occurrence. One reviewer, who shares the practice of team interviewing with this program, commented, "In almost all cases that I have seen, the team is better off without adding a team member versus adding one who does not jell with the rest of the team or work well in a team environment."

An expected, but still striking difference we saw in Agile programs versus more traditional software programs was the attitude toward requirements. In the more traditional programs that some of our interviewees had worked on, a tremendous amount of effort, energy, and attention was paid to refining requirements statements to a low level of detail and obtaining multiple levels of approval of them before doing any prototyping or visible design activities. The confidence of the development team in how accurate the requirements were in terms of the user's needs was low.

However, they could not substantively move forward with the design until the complete systems requirements had been approved. A reviewer who had a similar experience said, "To date, I have not seen where the rigor applied to our requirements has brought any value to our product." The attitude toward requirements in the programs we interviewed using Agile was much more focused on a definition of only enough capability to bring acknowledged value to an end user, with much less focus on ultimate completeness of the requirements set. In the program using Agile, the team

was confident that with the user involvement that they knew they could count on, they would discover additional requirements as the project evolved and that all the requirements they worked on would provide definable value to the customer.

The shared value of prioritizing end-user involvement to build confidence in requirements early allows both the acquirer and developer to permit appropriate ambiguity early in the project. This is contrary to what is more typical in a traditional acquirer/development team, which strives for completeness of a requirements specification prior to significant design work. Although this is a false completeness (almost everyone we spoke with acknowledged that the complexity of today's systems makes it impossible to completely know a system's requirements prior to design), the shared values of the acquirer/developer team are focused on the completion aspect of the requirements rather than allowing evolutionary learning about them.

Here is what one colonel tells his subordinate PMs about requirements and release planning:

> Planning Release 1… [should contain the] smallest field-able amount with tangible ROI that the customer will agree to, and
>
> - should demonstrate that the system will work end-to-end… or a logical chunk of it that can be built upon.
> - Pick the easiest requirements that have the best quality data.
> - Fight for this. The customer wants it all in the first release… every time.
> - Give success earlier, validate assumptions, buy time to further analyze the harder things.
> - If budget cuts take everything except the release 1 money, you are still a success.
>
> Planning Release 2 thru 'n'… use a model.
>
> - You must know: the interfaces required, the quality of the data required, the hardware required, and the software required… with costs and estimates of complexity for each.
> - Create a model based on the variables above… use it for estimates going forward; continuously refine.
> - If you have to go around the world to field each increment at each unit, it means many trips; [that must be] synced with [the] deployment schedule… [which will cost lots of] $$$… think this thru carefully.

Culture is ingrained in the organization, and is intertwined with everything that the organization is and does, including these dimensions:

- organizational structure
- leadership style
- rewards system
- communication and decision-making styles
- staffing model

Table 1 compares the above cultural dimensions as reflected in typical Agile and traditional DoD environments.[2] This is not a comprehensive characterization, but is intended for purposes of comparison. It also provides an informal way for a DoD organization to understand how closely its current culture reflects Agile cultural elements. Although this table may seem like it advocates Agile elements as superior to traditional DoD elements, for adopters of Agile, the point and the challenge is that the Agile adoption is likely to occur within the traditional DoD culture and steps will need to be taken to mitigate risks related to cultural conflict that is likely to arise.

An Agile culture runs counter to the traditional DoD acquisition culture in many ways, from oversight and team structure to end-user interaction throughout development. In our interviews with successful DoD Agile projects, we have consistently confirmed that adopting Agile methods requires a mindset change for government program management offices and other acquisition entities the projects interact with, such as the Office of the Secretary of Defense (OSD). The first step in effecting a culture change of this type is to understand the differences in assumptions, shared values, and artifacts that make up the cultures of Agile projects and those of more traditional projects as executed in the DoD [Schein 2009].

[2] These are generalizations and by nature will not be present as stated in every Agile or DoD traditional program situation. Some DoD program offices, for example, have created excellent collaboration structures through the judicious use of integrated product teams.

Table 1

	AGILE DOD	TRADITIONAL DOD
Organizational Structure	Flexible and adaptive structures Self-organizing teams Collocated teams or strong communication mechanisms when teams are distributed	Formal structures that are difficult to change Hierarchical, command-and-control based teams Integrated product teams that have formal responsibilities
Leadership Style	Facilitative leadership Leader as champion and team advocate	Leader as keeper of vision Leader as primary source of authority to act
Rewards System	Team is focus of reward systems Sometimes team itself recognizes individuals	Individual is focus of the reward system
Communications and Decision Making	Daily stand-up meetings Frequent retrospectives to improve practices Information radiators to communicate critical project information Evocative documents to feed conversation "Just enough" documentation, highly dependent on product context	Top-down communication structures dominate External regulations, policies and procedures drive the focus of work Indirect communications, like documented activities and processes, dominate over face-to-face dialogue Traditional, representational documents used by the PMO throughout the development life cycle to oversee the progress of the developer PMO oversight tools focused on demonstrating compliance vs. achieving insight into progress
Staffing Model	Cross-functional teams including all roles across the life cycle throughout the lifespan of the project Includes an Agile advocate or coach who explicitly attends to the team's process	Uses traditional life-cycle model with separate teams, particularly for development and testing Different roles are active at different defined points in the life cycle and are not substantively involved except at those times

Agile culture, by contrast, is the culture of just enough. "Just enough" is, of course, contextually defined. Just enough documentation for a mobile game application versus just enough documentation for a missile navigation system is likely to be quite different, both in volume and in character. An Agile approach would dictate just enough

- detail and process planning to ensure compliance throughout the program
- review cycles to ensure the plan is well understood by all stakeholders
- reviewers to ensure the plan represents the desired level of compliance for the program

Implications of Adopting Agile for DoD Planning Approaches

Agile practitioners conceptualize a software development project as a value-seeking activity in which knowledge is continually being acquired about the product and the user needs that make it valuable. While most mature Agile development organizations recognize that a certain degree of upfront analysis is important, at the core of Agile is a fundamental belief that knowledge acquisition will and must progress throughout the course of a software development project.

One group of interviewees emphasized that, unlike other programs they had worked on, they were allowed to accept that the requirements will change as a routine—rather than exception— condition. Agile encourages interaction among the entire stakeholder team to determine what is required to satisfy the current user needs. Thus, they do enough analysis so that user stories (their way of expressing requirements) assigned to the current iteration can be accomplished, but allow for continuing accumulation of data for stories that will be implemented at a later date. This approach also allows them to not waste effort on items that may change or might be interpreted incorrectly. Another group said that Agile allows them to get a product out in a timely manner because the user environment is constantly changing; they could never know all the requirements.

As the users, environment, and technology change, updates and changes can be made to the original product. Non-Agile DoD programs have explicit mechanisms for addressing change, of course. However, generally speaking, those change processes, once a baseline has been posted, require a significant amount of time and effort to navigate prior to a change being actionable.

This reflects a mindset of change as an exception, versus change as routine. Agile practices focus on creating a development environment that acknowledges and supports continuous knowledge acquisition. For this reason, detailed upfront requirements specifications and detailed, long-range, task-level project plans are not used within Agile development projects. Agile projects generally reflect a multi-level approach to both requirements specification and project planning.[3]

[3] Note that multi-level, rolling wave planning is not unique to Agile projects; however, it is one of the practices that have become the basis of several Agile methods.

Rolling wave planning, an element of earned value management, is a particular approach to resource planning and management that is prevalent in DoD acquisition [Defense Acquisition University 2011]. A time period for a wave is defined for the project (often three months, but not always), and the next wave is planned in significant detail, while waves that follow it are generally planned at a much higher level.

The thinking, which is similar to that in the Agile community, is that learning occurs as we go and detailed planning beyond a certain window is counterproductive. However, where rolling wave practice departs from Agile practice is in the foundation of each. Rolling waves assume a complete and static set of requirements that will be worked off in a pre-planned fashion. Agile practice assumes that, as the developers and users learn together, requirements priorities and even the need for them will shift, and each iteration provides an opportunity to realign the developer's priorities to the user's needs.

Interviewees working this way did not provide direct cost comparisons between using baseline change requests (BCR) and shifting requirements priorities between iterations. However, those who had worked previously in more traditional programs did state that the decreased level of formality at least provided a sense of collaboration between user and developer that was missing in other programs they had worked.

Conclusion

To sum up, Agile is a different culture, but one that can be incrementally adopted and tailored to the existing conditions within the DoD. Things like oversight, cross-acquisition/development team structure, and end-user interaction throughout development are areas where incremental steps must be taken in many settings, rather than trying to adopt all of Agile at once. In our interviews, we saw a variety of adoption approaches, several of which were explicitly incremental to ease the cultural transition. The use of Agile methods and the oversight of Agile programs require a mindset change, not just a practice change, for the DoD and other government entities.

For example, a PMO that embraces the Agile principle that values operating code over extensive documentation may require a different set of CDRLs when formulating a contract. This not only requires a change in perspective, but also the creation of appropriate governance models, via tailoring DoD 5000.02 and CDRLs from such events as SRR, PDR, CDR, etc. The PMO involved may have to seek waivers from higher up the acquisition chain, and these higher-ups must also understand Agile methods if they are to understand what they are waiving. One of our reviewers cited a recent contract using Agile methods, in which they were bounded by an SDR milestone, but obtained approval to have IDRs (Incremental Design Reviews) beyond that time instead of the traditional PDR and CDR cycle.

At the recent AFEI Forum on Agile in DoD, acquisition strategies that promote Agile practices via tailoring of regulations and standards were seen as one of the ways that Agile methods could more usefully be employed.[4]

One of the barriers to this kind of tailoring is the uncertainty of program managers as to whether the desired tailoring will be approved, since these types of tailoring reflect a different approach than is expected by DoD policy makers. In programs that wish to take advantage of the benefits seen by other Agile programs in the DoD, both the government PMO and the contractor developer need to be aware that different skills sets or skill mixes will likely be needed in programs using Agile. Agile takes a lot of strong, focused team and management oversight at the middle and low levels. To achieve the benefits of any of the Agile methods commonly in use, the DoD acquisition organization attempting Agile methods for the first time will have to:

- plan for them
- train themselves in Agile concepts as well as ensure their developers are adequately trained and skilled in Agile methods
- anticipate the changes needed in their environment and business model, especially as they relate to increased end-user involvement
- work hard to make the changes a reality

If the team can stay together, the work of sustaining Agile approaches is usually much less than the effort required to initiate them.

[4] NDIA/AFEI. Program, DoD Agile Development Conference. NDIA/AFEI, December 14, 2010, Alexandria, VA.

References

[Defense Acquisition University 2011]
Defense Acquisition University. “Foreword.” Defense Acquisition Guidebook. Defense Acquisition University, 2011. https://acc.dau.mil/CommunityBrowser.aspx?id=314709. Accessed July 13, 2011.

[Schein 2009]
Schein, Edgar H. Organizational Culture and Leadership, 3rd ed. Jossey-Bass Publishers, 2004.

DM17-0009

About the SEI

For more than three decades, the Software Engineering Institute (SEI) has been helping government and industry organizations acquire, develop, operate, and sustain software systems that are innovative, affordable, enduring, and trustworthy. We serve the nation as a federally funded research and development center (FFRDC) sponsored by the U.S. Department of Defense (DoD) and based at Carnegie Mellon University, a global research university annually rated among the best for its programs in computer science and engineering.

Contact Us

Software Engineering Institute
4500 Fifth Avenue, Pittsburgh, PA 15213-2612

Phone: 412.268.5800 | 888.201.4479
Web: www.sei.cmu.edu | www.cert.org
Email: info@sei.cmu.edu