RG Perspective

Effective Agile: A Practitioners Guide for Program Managers

Abstract

Agile development has the potential to delivering higher quality software more quickly. To assist the Department of Defense in successfully applying Agile for enterprise business systems, Program Managers need to understand why and how to apply Agile techniques in the definition and delivery of software projects.

www.teamrg.com | 3rd September 2019

Contents

The Challenges in Agile Adoption in the Department of Defense 2
Identify where and how Agile will – and will not – be applied 3
Establish Agile support at all levels within the Defense organization 3
Define the Agile partnership between Government and Contractor 4
Collaboratively define outcomes, the solution, and the "Definition of Done" 5
The "Definition of Done" for Agile deliverables 7
Ensure the project team has the right Knowledge, Skills and Abilities (KSA) 7
Recognize how to apply Agile to "green-field" and "brown-field" applications 8
Allow flexibility in the selection and application of Agile methodologies 9
Move from Implementation + Sustainment to Continuous Development 9
Consider that Agile can be used by co-located, remote, and hybrid delivery teams 10
Assess how much change the End User can absorb 10
Ensure consistency in Continuous Development of Enterprise Business Systems 11
Understanding the trade-offs in CI/CD 11
Conclusion 12

Effective Agile: A Practitioners Guide for Program Managers

Successfully delivering Agile projects requires a sound understanding of the principles and application of Agile techniques.

- Agile methodologies can potential enable the Department of Defense (DoD) to deliver enterprise business systems software projects more quickly and with higher quality and usability, enhancing the ability to more cost-effectively support the needs of the warfighter.
- A key challenge is that Agile methodologies are often counter to the existing software acquisition and delivery approaches, requiring a change in mindset at every level of program delivery. Failing to understand Agile increases the risk of projects not achieving their desired goals and benefits.
- Program managers looking to deliver using Agile play a critical role in setting up projects for success, including the scoping, definition, design, and structure of Agile project teams and ensuring appropriate governance is applied to retain the focus on delivering high quality working software to end-users.

The Challenges to Agile Adoption in the Department of Defense

In order to move beyond simply spreading around a few Agile buzzwords and truly affect the fundamental changes necessary to successfully adopt Agile, the Department of Defense (DoD) must change its mindset and restructure its processes and procedures to transform its operations to better support adaptive approaches necessary to accelerate the delivery of capabilities to the warfighter.

Agile practices, processes, and culture are largely counter to long-established government development practices and procedures. Traditional waterfall is based on establishing comprehensive requirements upfront, predicated on heavy predictive plans, and tailored in excruciating detail to eliminate all unforeseen risks. This approach can have the opposite effect of making project more difficult to deliver as requirements are static and the lag in producing them fails to reflect the pace of technological and organizational change or embrace more adaptive, iterative models of project definition and delivery such as Agile.

Agile is more a philosophy than a proscriptive methodology and encompasses a range of well-tested tools and techniques which can be adapted and applied to a wide variety of projects based on goals, scale, and the expertise of the government and contractor project team. This flexibility is both its strength and a challenge – it takes experience to select the right people, tools, and techniques to fit both the project and team, and then continue to refine those as the project progresses based on continual monitoring and feedback.

Done well, Agile is a powerful and effective approach for the acquisition and delivery of software systems. Creating good software is only the start; the ability to adopt and embrace *use* of the software is also necessary.

There is a dark side. The fluidity of Agile also allows people to claim expertise and hide behind jargon, which requires the acquisition community become proficient at spotting "Agile BS" (also the succinct title of this useful paper from the Defense Innovation Board) A common misconception is that Agile is

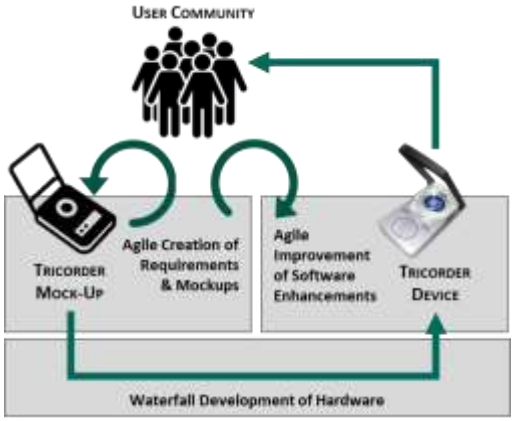
about coding to the exclusion of design and architecture, a myth which has been used as an excuse to justify "code-like-hell" project delivery since the early days of Agile adoption.

In reality, the foundational scoping and definition activities which – when done properly – quickly build alignment between Government and Contractor are a crucial step to set the project up for success and provide the guidance and guardrails to keep the Agile project on track to achieving the desired business outcomes. While Agile work products and reports are distinctly different to those of a traditional Waterfall project, they provide an effective means for the DoD to track the productivity, quality, and acceptance of Agile projects.

Identify where and how Agile will – and will not – be applied

The Defense Science Board paper (Defense Science Board : Design and Acquisition of Software for Defense Systems) provides a good overview of when Agile should and should not be applied. Some software lifecycle activities which have traditionally been regarded as Waterfall only can be done effectively and efficiently following Agile principles, including architecture and design and sustainment.

For example, Agile can be applied to design and definition of a new piece of integrated communications (let's call it a Tricorder) to work closely with end users to iterate on mock-ups for its functionality and user interfaces.



For the Tricorder, Agile will be used to produce the design and requirements, and hardware construction would follow a traditional process. Once produced, Agile can be used to iterate on the software, as well as capture the requests for enhancements to the hardware and refine the physical design to feed into the manufacturing process.

To structure a proposal for the Tricorder, the acquisition contracting officer would identify where Agile is applicable and, once confirmed those conditions are met, determine what information is required for each activity to ensure it achieves the desired outcomes. (See Table A for a guide of when and when not to choose Agile.)

The example above is intended to illustrate the importance of understanding the strengths and limitations of Agile, selecting the appropriate approach for each phase of work, and ensuring that the dependencies and flow of information are well defined to minimize disruption with each workstream. A task order or contract should be structured to clearly identify the activities or projects within the program which are to follow a specific approach and clearly outline the dependencies, responsibilities, and deliverables between them.

Establish Agile support at all levels within the Defense organization

The current DoD acquisition process generally prescribes a specific approach developing and delivering software based on a Waterfall methodology. Agile devolves decisions to the level within the project team most appropriate to make them to maintain velocity of development.

Software factories as described by the Defense Science Board paper on software acquisition require consistent application of Agile best practices to deliver high-quality software – otherwise it's more like an artist's collective, which can produce interesting individual works of highly variable quality and with little consistency.

To improve acquisition and delivery of successful Agile projects, higher levels of Government need to provide an enterprise-wide management framework which establishes a consistent set of practices and shared knowledge for Agile program leadership. An organizational-level best practice is to establish an **Agile Enterprise Governance Team** which maintains the framework and provides expertise and guidance to projects in the development portfolio. Agile standards focus on *frameworks* and *outcomes*, leaving the selection of *how* and *what* to deliver to the project teams.

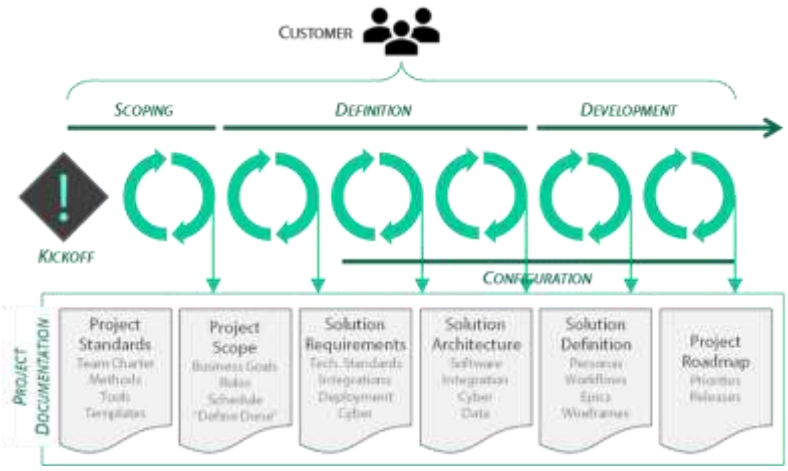
This flexibility ensures a level of consistency across projects while allowing teams to choose the techniques and technologies which best fit their specific project needs. It also ensures that at transition points for projects, the backlog, designs, code, and architecture are in a consistent format and set of tools. Without appropriate governance, each Agile project team could set up their project in whatever way they want, which can restrict the ability to transition that work or information to a different team or Contractor in the future. This also complicates oversight and collaboration, as the Government representatives will have to learn whatever tools, formats, and interfaces the project team is using in order to participate effectively in team activities.

By comparison, creating a well-defined framework for Agile delivery which enables teams to tune their approach to fit their project needs improves collaboration, creates consistency, and help assure effective governance. For a list of functions to consider for the Agile governance framework at the program level, see Table B. For decision-making activities at the project level, see Table D.

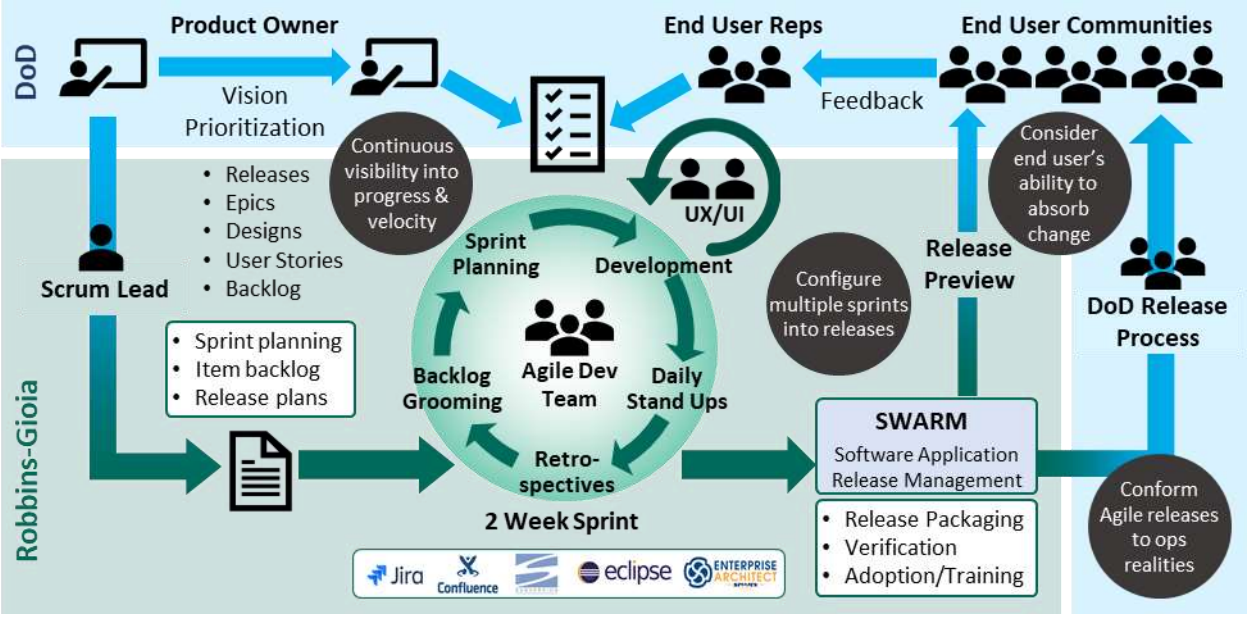
Define the Agile partnership between Government and Contractor

Effective Agile requires shifting the traditional Government-Contractor relationship to embracing more of a genuine partnership in systems delivery, including collaborating in establishing and articulating goals, outcomes, roles and responsibilities, quality expectations, and the specific Agile methodology to deliver the capabilities required for the defense business system project.

Sustainable success with Agile requires recognizing that the methodology is *design-centric* and not, as many interpret it, *coding-centric*. A truly Agile approach focuses on ensuring that what is to be delivered is well understood, usually through iterative refinement with Business Owners and End Users, before the bulk of coding begins. This ensures that what is produced is *functional software* requiring the *minimum amount of coding* effort and which delivers needed business outcomes and is readily adopted by the end user community.



A collaborative, sustainable partnership between government and contractor is achievable when goals, outcomes and requirements are clearly communicated, project teams consistently follow a well-defined Agile approach, team-member roles and responsibilities are understood yet flexible as needed, and resource commitments are made and kept. In addition, efforts to identify and mitigate potentially disruptive factors can greatly enhance the project team's ability to deliver the capabilities needed to support the warfighter timely and efficiently.



It's important for the government to recognize and address the challenge posed by external government project influencers such as DISA, IA, and others outside the PMO who may not be willing to adapt to an Agile approach. To mitigate the disruptive impact on the Agile methodology, relationships with those organizations will have to be influenced and cultivated to ensure their interactions are adapted to align with Agile. If not managed successfully this may hamstring the ability of the DoD to realize the benefits of the Agile methodology.

For an RFI/RFP, we recommend using Agile-friendly language and focusing on describing the business objectives and outcomes to be achieved. Our recommendation is the contract language ensure compliance with Government expectations, standards, and practices and ensure that the Contractor understands the processes and responsibilities for delivery. A list of recommendations for fostering a Government-contractor partnership is included in Table C.

Collaboratively define outcomes, the solution, and the "Definition of Done"

Most Agile methodologies provide for numerous opportunities for Business Owners, Product Owners, and End User Representatives to ensure that the project team is building the right solution, in effect providing guardrails to keep development on track to achieve the desired business outcomes.

A key first step during the acquisition and project initiation is to ensure that the initial scoping of the project establishes the business outcomes and priorities, and that coding doesn't start until architecture, definition, and preliminary design confirm that the project team and Government customer are aligned.

For example, a "minimum viable product" (MVP) as described by the Defense Science Board requires the security, integration, performance, and usability objectives are attained in order for it be "viable".

Part of the plan to get to the first release (the MVP) is to ensure that the project team understands why, what and how to build the software to meet the business objectives.

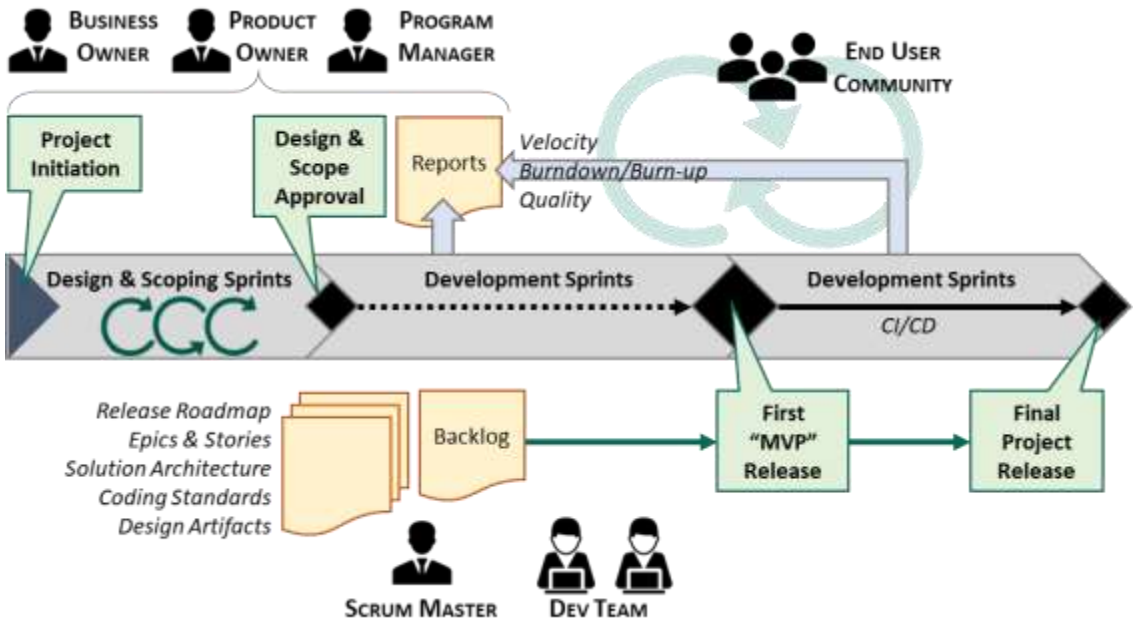
Establishing this understanding within the project team requires the:

- **Business Owners** accept the release plan and sign off on the Epics and Stories to be included in each release (essentially the business priorities to achieve required outcomes)
- **Technical Authorities** sign off on the solution architecture, including software, system, security, integration, and data architectures, and coding and development standards
- **Product Owner** and **End User Representatives** accept the Epics, Stories and any preliminary designs such as wireframes and mock-ups which describe the workflows and features.

Where this approach differs from current DoD project documentation is the level of detail. The artifacts created above – perhaps with exception of architecture documents – start at a high level and are then refined by the project team as part of iterative delivery during the first rounds of "design sprints", which also helps to:

- Embed Agile cadence and behavior within the project team and stakeholders
- Maintain constant engagement of Government in setting and adjusting priorities
- Ensure strong adherence to the basic Agile/Sprint methodology and principles
- Adapt/adopt tools as needed throughout but consistently follow the Agile framework/philosophy.

These deliverables and the reviews can be included as SDS in the contract to provide checkpoints to ensure that the team knows what it's doing and what success looks like before starting to code.



A common mistake on Agile projects is "coding to an MVP", which focuses on the end-user and eschews the creation of design and solution documentation and coding standards described above. This approach, often described as "faux Agile", can be time consuming and frustrating for Business Owners and End User representative as multiple iterations are required to refine the design through re-writing code, rather than confirming them earlier in design and description. It also makes compliance with security and other technical standards harder, potentially requiring refactoring or re-

architecting of the software. The result is significantly more work for the project team and a less stable and maintainable code base for the software produced.

The "Definition of Done" for Agile deliverables

In general, for each deliverable – whether a document or code - the Definition of Done (DxD, which we use here to avoid confusion with the DoD) describes what is required for the work to be regarded as complete and count towards velocity. The DxD is determined for each project by the team and confirmed during the planning phase for each activity. This requires gaining agreement with the final recipient of the expectations and acceptance criteria before the work is begun. For the DxD at each process iteration, see Table E.

> The "Definition of Done" sets the conditions for contractual success.

The ability of a Contractor to meet Government expectations of any contract executed using Agile is greatly improved when two conditions are met:

- Requirements are clearly defined yet not over specified – they focus on what is required at a business workflow and outcomes level, without specifying how those will be achieved
- Government stakeholders and subject-matter experts participate effectively and timely as Agile team members or in support of the team, as appropriate to their roles in the project, and consistent with how successful Agile projects are executed

The lack of either one of these can make the difference between failure and success. The focus of contractual language should focus on outcomes, expectations and compliance with Government standards and Agile best practices. The contract should also outline how the Agile delivery will be overseen for compliance and define the method of measurement in Agile terminology for productivity (velocity), quality (rework), and completeness (acceptance and the "definition of done." Performance can be incented by choosing the appropriate contract type.

Contract language can provide guidance and set guardrails for contractors and can help create an environment that fosters effective, collaborative government-contractor relationships. Nonetheless, among the most important factors for driving success of Agile projects will be the clarity with which the government identifies requirements and desired outcomes while avoiding prescription of how those are achieved, and the degree and timeliness of engagement by key government stakeholders and SMEs throughout the life of the project.

Ensure the project team has the right Knowledge, Skills and Abilities (KSA)

Successful Agile is about people. Team dynamics, collaboration and trust are critical to productivity and quality. A realistic assessment of team experience for both Contractor and Government participants is required in order to be able to set an Agile project team up for success.

Agile is a discipline which requires both experience and practice to do well. While some of the principles are relatively simple, there is a plethora of different methodologies, tools, and techniques which can be applied in a myriad of combinations. For this reason, teams should always have a core of experienced Agile practitioners and supplement them with coaches and mentors as needed.

> Don't try and run an Agile project with only novices on both Customer and Development sides.

The experience level of the team and of individuals in specific roles can (and should) determine the Agile methodology and tools to be used on a project. Additionally, the distribution of design, development, and testers in a team is entirely dependent on project scope. The Project Manager and Scrum Master need a level of experience in order to appropriately size and shape the delivery team.

Knowledge. Paper qualifications are less useful in Agile than practical experience. Each team member, based on role, will have practiced Agile, ideally on multiple projects of various scales and scope. Where team members lack experience, they need to be paired with more experienced team members or be provided external mentoring and coaching.

Skills. Agile suits having multi-disciplinary members (those with Design, Dev & Test experience) who can therefore communicate effectively with others based on a shared understanding of need, and also respect the importance of different roles and experience in contributing to team success. Technical skills, such as design, coding, testing, architecture, and using Agile toolsets are obviously essential but need to be paired with a similar range of "soft" skills and abilities including communication, collaboration, and leadership.

Abilities. Agile is highly collaborative and requires all team members understand their roles and commitments to the project. Effective collaboration and teamwork require establishing "trust through delivering" and "respect through understanding". See Table F for a list of attributes of good Agile team members.

For Agile leaders and managers, it's essential to be able to foster a high-performance culture based on trust, collaboration, and commitment. Some of the core attributes which make for an effective Agile leader are the ability to:

- Facilitate discussion and look within the team for expertise and opinions on best practices and team performance
- Encourage and guide teams to work together to solve problems
- Be open to discussion and feedback, specifically for ways to improve performance
- Create a culture of openness, where issues can be quickly raised and addressed without penalty

There are a similar set of managers and leadership traits which can quickly undermine Agile teams, some of which are:

- Seeking to manage by authority, rather than lead through competence
- Creating a "culture of fear" through a hostile culture to bad news, critiques, issues or suggestions
- Having a rigid focus on metrics over encouraging team performance in achieving outcomes
- Taking credit for team successes without visibly sharing the credit across the team
- Being openly critical of the team, individuals, the project, or customer

Equally, there's a long list of individual attitudes and behaviors which are undesirable and counter-productive for Agile team members and which can undermine the success of a project, which can be found on Table G.

Recognize how to apply Agile to "green-field" and "brown-field" applications

In most enterprise environments will required a new or replacement business system to connect or integrate with legacy systems. Agile projects are no different in needing to define the architecture of the solution, including the system integrations, data architecture, and security compliance.

Brown-field Applications. If poor documentation of legacy systems is a known factor, the project scope and plan need to allow enough time to capture *sufficient detail* (not perfect detail) to proceed with design and development. Depending of the complexity in this may involve the legacy architecture work proceeding in parallel with ongoing systems development. The objective of this

work is not to fully document the legacy system, but to describe in enough detail the interaction between the legacy system and the project software application so that development can begin without risking significant re-work in the future.

One Agile technique for this is to treat the legacy system as a *non-human persona*. This then describes the interactions as Epics and Stories in addition to the technical description in the project's solution architecture. See Table H for some additional Agile good practices for documenting integration with legacy.

Green-field Applications. The creation of systems and architecture documentation for new applications follows similar lines, and serve as guardrails to keep a solution on track to meet the desired business outcomes:

- Create the initial solution architecture at a high level with sufficient detail to make key decisions on the implementation and minimize the risk of later re-work
- As project progresses, continue to refine and update the level of detail in the architecture to support development and reflect the actual implementation
- Maintain all architecture and solution definition documents as part of project library.

Allow flexibility in the selection and application of Agile methodologies

It can be very difficult to determine which methodology will best suit a specific project, as Agile execution is as much about the competence and behavior of the team over any particular techniques. Being too proscriptive could have the effect of excluding robust alternatives, limiting fair and open competition and potentially result in lower project performance and higher costs for the Government.

To fully realize the benefits of this for a new project, the Government should focus on setting the goals and measuring the output and outcomes of an Agile project and determining that the Contractor is capable and able to achieve their productivity and quality objectives.

To ensure consistency, it's not uncommon to proscribe the management tools for the Agile process, including requirements and backlog management, code control, and automated testing. However, having the Contractor be responsible for selecting their Agile delivery model has benefit:

- During the acquisition process, the Government can compare proposal from different vendors and enquire as to their past performance and depth of experience. This can help identify the proficient from the less capable contractors.
- During delivery, the Contractor cannot blame any issues arising from the chosen methodology on the Government, such as inexperience with a specific tool, or the poor fit of a particular technique to the given project.

The only conditions where a specific Agile methodology should be mandated by the Government are when a Contractor is being brought on to supplement an existing project which already has an established Agile methodology, tools, and development and management processes, and any new methodology would be disruptive to delivery.

Move from Implementation + Sustainment to Continuous Development

Traditional DoD enterprise business systems projects have followed the model of a lengthy definition and implementation period followed by an even longer sustainment phase, in which system capabilities are maintained but rarely updated, modified, or enhanced without additional contracting and budgetary effort.

The DoDI 5000.75 BCAC Phase 5 objective is "to provide enduring support for the capability established by the business system [including] active engagement in both functional and technical opportunities for ***continuous process improvement***", which can be interpreted to support an Agile approach to the ongoing sustainment and maintenance of business systems.

As the DoD has identified, in the commercial sector it is standard to incorporate adaptive, corrective, and perfective initiatives as part of ongoing system support. "Agile sustainment" can be regarded as a continuation of the development process which happens in parallel with first major release to the user community and associated change management supporting ongoing adoption of the system.

Acquisition needs to allow for a broader range of activities to be permitted to enable continuous process modernization. When contracting for Agile sustainment, the Government should standardize the language around expectations to provide the Contractor with flexibility (and responsibility) to deliver the highest quality of service. For some of the considerations we have identified, see Table I.

Consider that Agile can be used by co-located, remote, and hybrid delivery teams

While co-location of the entire product development team is ideal, it is often not practical depending on the project and location of team members, end-users, and end-user representatives. Facilities may also not be able to support effective development, especially at remote locations with limited connectivity or infrastructure. Access to talent may also be constrained by geography, and in the case of multiple contractors contributing to a single project this may be exacerbated by including team members from several different companies. In addition, Government representatives might be spread across the continental US or the world.

Most software and consulting firms have had to adapt to being unable to completely co-locate Agile project teams, and instead often have "blended" structures with some teams co-located, some partially remote, and others completely remote. See Table J for some of the best practices for managing dispersed Agile teams.

For the Government, enabling this requires that:

- Government contributors, including business and end-user representatives, have the time, training, and equipment to participate effectively remotely
- Project teams must be able to have access to collaboration tools to be effective on/off DoD sites
- Preferred tools and technologies can work with DoD environments
- Government facilities have enough network bandwidth and connectivity to support remote working.

Assess how much change the End User can absorb

For many enterprise applications, frequent changes to workflows and user interfaces can create stress on the end-users and undermine productivity and adoption, and potentially safety and security. During project scoping and design, the amount of change and impact on end user communities needs to be thoroughly researched and understood. This informs the decision around how software releases will be delivered to users, and the viability of applying continuous development methods within the project.

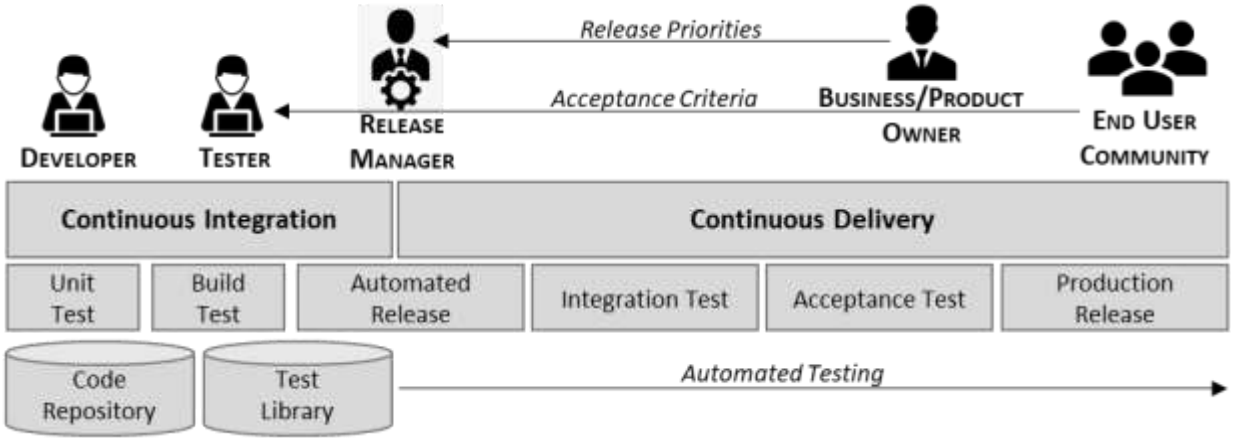
If the user community is unable to absorb change – or is constrained in the frequency of release by technology or operational factors – then investing in a fully configured continuous development solution is not likely to show much benefit. Some of the factors which Agile leaders need to consider when determining the viability of continuous development are whether:

- Releases require substantial changes in user interfaces and workflows, requiring dedicated training and change management to ensure adoption and correct operation
- The IT organization can work with the software project team to ensure timely and adequate testing and quality assurance of software being released, to minimize the risk and impact of defects
- The project has enough resources to properly prepare the user communities for each release, including training, coaching, shadowing and any related activities including installation, data migration/preparation, and business process transition.

Ensure consistency in Continuous Development of Enterprise Business Systems

While Continuous Integration/Continuous Delivery (CI/CD) is a common element of Agile methodologies including Extreme Programming (XP) and DevOps, it also places significant demands on the development team and infrastructure. Developers and testers need to set up the automated test environment for multiple daily builds and releases, which requires a significant amount of time and system resources. For complex business systems, this can involve hundreds of automated test cases being created and maintained, requiring significant input from the Business Owner and representatives of the end-user communities.

When implementing a CI/CD pipeline, responsibility for providing the environment can depend on whether the full production environment is Government or Contractor hosted. For example, for a FedRAMP-approved Cloud service the entire pipeline would be provided by the Contractor. If the



application is hosted by DISA or the Air Force Common Computing Environment (CCE), the governing organization will need to establish release and testing procedures which support CD. If the policies and procedures do not support the desired frequency of release, the then effort and cost to establish a CI/CD environment will largely be wasted. Table K contains some pro's and con's of using CI/CD.

Understanding the trade-offs in CI/CD

Like many Agile techniques, implementing CI/CD requires understanding the trade-offs which are incurred in transitioning from a traditional test-release process. The core principle of Ci/CD is the use of automated testing defined in parallel with systems development.

While in traditional test-release the test scripts are written and then executed once development is complete, in CI/CD the test scripts are developed in parallel with feature development and executed as needed during development. A key element here is the difference between a "tester" and a "test developer": the latter are writing scripts and code to execute testing, rather than following a set series

of test steps. This requires an enhanced skillset which translates into being a more expensive – albeit more productive – resource.

Another aspect of CI/CD while programs need to consider is it requires a range of additional software tools and processes which need to be configured and maintained. Tools like Jenkins, Maven, Selenium, SonarQube, and GitLab need to be integrated into a pipeline and ideally used consistently by multiple projects and systems.

This release infrastructure requires a significant amount of IT resources to support in terms of skills and services (storage, compute, transfer). Organizations and projects looking to "save money and reduce resources" by adopting CI/CD are likely to be disappointed. The objective of CI/CD is not to reduce costs in development – it is to improve quality and speed and *reduce costs of rework and process inefficiency.*

Delivering a more stable, functional, and responsive business system generally returns benefits far greater than a 10%-25% increase in the cost of development. Making the business case for implementing CI/CD needs to look beyond the development process and identify and quantify the net benefit to the business of improved system responsiveness and performance.

Defining and managing the CI/CD process at an organization (or at least portfolio) level provides significant economies of scale which can counter some of the additional costs of maintenance. As discussed earlier, consistent processes and discipline assists Agile teams in predictable delivery of quality software.

Conclusion

We believe that establishing a well-constructed enterprise Agile framework enables consistent high performance from Agile project teams, helps build a coherent level of capability and competence across both Government and Contractors, and promotes rather than constrains creativity and innovation required to manage portfolios of Agile projects for consistent delivery and best practice and achieve the ideal of a "software factory".

The key themes we believe Program Managers need to consider in adopting Agile within the DoD are:

1. Agile is a design-centric methodology, and acquisitions and contracting need to ensure there is a focus on the architecture, design, and End User adoption in addition to code development
2. Properly scoping and defining the solution is an essential early activity to establish alignment between Government and Contractor on project objectives and foster effective collaboration
3. Anything that disrupts team dynamics will negatively impact successful delivery (including incompetence, infighting, lack of discipline, and use of inappropriate tools and techniques)
4. Agile is team-driven not top-down. Success requires creating a framework – and not being overly proscriptive – to leverage flexibility, and setting the right metrics and measures to incent high performance
5. Agile can readily be continued into sustainment to provide ongoing enhancement and development throughout the lifespan of the software.

Agile is not an excuse to avoid good program management, architecture and design – in fact, Agile is reliant on strong management discipline, a sound architecture, and effective use of design principles to deliver high quality working software for the Government.

Want to know more?

Please contact:

Liam Speden

Chief Technology Officer, Robbins-Gioia LLC

Liam.Speden@TeamRG.com

References

1. DoDI 5000.75 Business Systems Requirements and Acquisition, US Department of Defense, February 2017.
2. Defense Innovation Board, Recommendations, https://innovation.defense.gov/Recommendations/

www.teamRG.com

Contact Us

RG HQ
99 Canal Center Plaza, Suite 300
Alexandria, VA 22314
703.548.7006

About RG

RG partners with clients to test and refine every solution to meet their exact needs. We take pride in tackling complex management challenges with fresh and innovative insights and in transforming

Table A: When and when not to use Agile

Use Agile When…	DO NOT Use Agile When…
• Qualified, well-seasoned, well-disciplined team members will be staffed. • Interfaces are well-defined. Deliverables can be reasonably distributed in work packets achievable in 2-4 week periods. • Responsiveness to customer requests defines success. • The customer or customer representative is available for close collaboration throughout the project. • Scope can be adjusted to fit schedule. • The customer can reprioritize and add requirements as they progress. • Work is ground-breaking with steps defined by progress resulting in estimates that are not expected to be reliable. • Incremental results have significant value. • Process is by nature iterative, allowing for cumulative results though sprints, and for overlapping planning for the next sprint while work takes place in the current sprint.	• Work is commoditized and can be delivered by lower-skilled staff. • Inexperienced individuals are principal candidates. • Deliverables cannot be reasonably distributed in work packets achievable in 6-week periods. • Success is defined as absolute adherence to fully scoped customer requirements identified completely in advance of development. • The customer is not available for close collaboration throughout the project. • Critical (or many) steps involve long lead times or lots of specialized resources. • Incremental results have little or no significant value for anyone. • Process is best implemented as linear (waterfall model in software) or spiral. • Contract requirements or regulations mandate the use of a specific life-cycle process and attendant documentation that does not follow Agile tenets, and under which no ready allowance can be made for an Agile approach[1]. • No contract requirements or regulations preventing the use of Agile or making its employment impractical apply.

1. "When to Use Scrum for software projects," Kevin Thompson, *Cprime, (*http://www.cprime.com/community/articles/whentousescrum.html)
2. Program Management for Agile, Robbins-Gioia, LLC, 2017 (https://cdn2.hubspot.net/hubfs/3452025/White%20Papers/program-management-agile.pdf)

Table B: Functions of Agile governance framework at the program level

☐	Set the DoD's strategic goals, objectives as business objective and associated performance measures applicable for Agile projects within the program portfolio
☐	Coordinate development initiatives across the organization, ensuring that projects have a change management plan to drive adoption in user communities and realize desired business outcomes
☐	Define program "templates" for Agile projects, which contain standardized descriptions for roles, phases, architecture, design, coding, deliverables, and metrics based on typical types of Agile software application
☐	Set the delivery standards for Agile projects, including standardized tools for delivery management, tracking, code management, and other project assets
☐	Ensure that projects are delivering software which complies with DoD technical standards
☐	Identify best practices from project teams, capture those practices in the framework, and communicate those out to active project teams
☐	Maintain a common repository of system assets, such as architecture and project documents, which can be used as references for acquisition and delivery teams

Table C: Recommendations for fostering partnership between DoD & Contractor

☐	Identify goals, intended business goals and end-state, and define business and high-level functional requirements in clear, concise terms, ideally as Epics and Stories with associated Personas
☐	Clearly define Agile project roles and responsibilities for contractor & government personnel as appropriate
☐	Set expectations for formal briefings and other collaboration mechanisms both inside and outside the program team, including expected levels of interaction with end user communities
☐	Request description for how Contractor will engage stakeholders, business owners, and end users to improve technical quality and ensure change readiness is included and integral to the solution and project approach
☐	Establish evaluation criteria that allow the government to award the contract to the bidder that demonstrates the fastest, lowest risk path to the minimum viable product (MVP), and then to the final full production release
☐	Provide Contractors with key details that will help determine project scope and complexity, including: Expected time from award to first deliverables, MVP, and final full release; expected level of engagement with Government representatives and end users; numbers and locations of end users; the systems that must be integrated with and method of integration
☐	Clearly identify what technical standards need to be complied with, for example for cybersecurity, production deployment, or data management.
☐	Detail the Agile delivery framework to be used (if any) and best practices for definition, design and development, including how productivity, quality, and acceptance will be measured
☐	Set expectations for how the Agile project will be kept on track, including the guardrails put in place, such as phases and design reviews, to ensure that the project delivers what is needed
☐	Specify Agile roles & responsibilities which organization (Government or Contractor) will cover them
☐	Identify the Government stakeholders, subject experts, end users and other representatives who must participate, and provide any time boxes or constraints on participant commitment and engagement
☐	Require the Contractor to develop and gain government approval for the initial Project Definition, Prioritized Backlog, and the Release Plan: • The initial Project Definition establishes a shared understanding of Government priorities with the Contractor and Agile project team • The Contractor's Release Plan outlines the path to the first functional release of a minimum viable product (MVP) (http://bit.ly/MinViaProd), defined as a coherent set of features to deliver usable software, including integration and security • How the Agile team (Contractor and Government) will collaboratively manage the Prioritized Backlog during project delivery • Which project artifacts require Government approval and signoff, and by whom: solution architecture, Epics, Stories, Use Cases, interface designs, prototypes, acceptance criteria, etc.
☐	Contractor is to provide government with milestones/reports that will communicate program status and provide tripwires to overruns/slipped schedules • Require Government acceptance of primary Agile measures (e.g., burndown/burnup, etc.) • Require outcomes to be measurable, including quality (based on Government expectations/ definitions)
☐	Specify the Contractor is responsible to ensure End User adoption by following Organizational Change Management best practices from the beginning of the project (a Change Management plan)
☐	Do not specify how the contractor is to meet government business/technical requirements – let it emerge from the Agile methodology

☐	Reinforce the controls available to the Government to ensure that projects are delivering viable software which meets the business need through regular solution design and architectural reviews.
☐	Consider ways to mitigate the potential disruptions to Agile projects caused by budget and funding cycles, and turnover in Government personnel
☐	Ensure commitment to follow Agile methodology is obtained from all government participants.

Table D: Agile decision-making activities at the project level

☐	The Product Owner (PO), Project Manager (PM), and Scrum Master (SM) use the business outcomes to define Agile project goals & measures of success, including required level of adoption by End User communities
☐	PO and agree on the Agile methodology to be used, and with the Lead Developer define and set up the project, process, and code management tools based on Program guidelines
☐	The Lead Developer sets out the coding standards and templates to comply with Program standards
☐	The User Experience/User Interface (UX/UI) design lead determines the design guide, based on the organizational guidelines provided by the Program
☐	The Solution Architect defines the architecture to comply with technical & integration requirements (security, integration, data, deployment, software, etc.)
☐	The PO and Solution Architect create the Epics and high-level User Stories for the core Persona workflows
☐	The PO and SM set out the design, development and release plans
☐	Development team, including developers, testers, and designers determine how each User Story is deconstructed into Use Case & assigns story points

Table E: Definition of Done

NOTE: This table is to be considered RG proprietary information	
Process Iteration	**Done is defined as…**
Business Level	Wireframes, screen mockups, requirements, high level test criteria, and fully developed epics agreed upon by the Product Owner, Program Manager, and Scrum Master and placed in the backlog for development
Sprint Level	Product Owner acceptance of the Epic/Story/Function following demo. Then moves on to Integration.
Increment Level	Epic/Story/Function considered complete following successful integration and User/Representative acceptance and placed into release backlog
Release Level	When the release has been deployed to the production environment and all documentation has been submitted and accepted
Reference: Agile Alliance Definition of Done	

Table F: Attributes of good Agile team members

☐	Having a high level of organization, self-discipline, and commitment to the project
☐	Ability to see the big picture, not simply what's on their current plate. Having a good understanding of the high-level project goals or functionality allows for seeing new ways to complete the "story" more efficiently
☐	Willingness to speak up if something doesn't feel right, or if problems or risks are anticipated (i.e. must be comfortable with asserting their opinions)
☐	Humility: anyone who thinks they know all the answers isn't an Agile thinker and won't be a good team player. Willingness to acknowledge when change is needed and to ask for help (i.e. too much arrogance can be detrimental to the team)
☐	Must be flexible or adaptable in the approach and solution (i.e. cannot be locked into the mentality of "this is how we've always done this" or "This is how we did it on the last job")
☐	Desire to continue self-education to keep current on new technology, concepts, tools, etc.
☐	Must be an effective team worker and communicator, be attentive to detail, able to take constructive feedback and adjust behavior accordingly
☐	Willingness to share lessons learned from past experience to fellow team members. Acting as a mentor does not have to be a formally established position to pass on lessons learned

Table G: Attitudes and behaviors undesirable for Agile team members

☐	Are "Lone wolves" who don't like working in a team environment
☐	Prioritize their own work over that of the team
☐	Are unable to manage their own work, requiring micro-management to complete tasks
☐	Are unwilling to help others (or worse, undermine the work of others)
☐	Struggle with clear, concise, honest communications with others
☐	Don't like receiving feedback or criticism of their work
☐	Believe that their way is the only way to do something
☐	Are unable to follow the project-defined discipline and prefer to follow a previously learned approach
☐	Have a low sense of completion, regularly miss deadlines, don't show up for meetings, etc.

Table H: Agile good practices for documenting integration with legacy

☐	As part of project scoping, identify legacy systems which need to be interfaced with and the dependencies and interactions with user workflows
☐	Focus on what is relevant to deliver the current project, and determine the appropriate and necessary level of detail in documentation
☐	Define in parallel with ongoing scoping, definition, and delivery, update continuously so they become part of the overall project library of design, architecture, and descriptive assets
☐	Refine iteratively the legacy integration descriptions until "done" is achieved (to avoid over-analysis)
☐	Maintain the description as part of the project library of "living" documents, including updating it to reflect and changes in the legacy business systems (this helps identify any impact on development as a change in the non-human persona needs or behavior may impact their Epics and Stories)

☐ Ensure consistency of architecture and legacy systems integration documentation across multiple projects in same customer or enterprise environment

Table I: Considerations when contracting for Agile sustainment

☐ The Government should include explanation of "adaptive, corrective, and perfective activities" to ensure that the Contractor adequately addresses these requirements
☐ Recognize that technology upgrades may enable code to be rewritten and simplified to enhance performance even if not "broken", such as re-writing of stored procedures in databases and refactoring code to take advantage of new libraries. These can deliver performance improvements to end-users without interrupting daily workflow and processes and also make the system easier and cheaper to maintain
☐ Allow the Contractor to adjust the team structure to fit the needs of ongoing sustainment. For example, sustainment often requires a lighter UX capability than during system development as emphasis is on maintenance and enhancement of the production UI and code
☐ Ensure Contractor tracks end-user generated requests and defects as well as team-identified enhancements along with other continuous improvement opportunities in a common backlog along with the assessment of potential impact and value to the Air Force
☐ The Government and Contractor should regularly review the common backlog and prioritized them based on business value, and share the list with Business Owners for validation
☐ The Contractor should be capable of adding additional Agile teams to undertake additional high priority work under the overall direction of the sustainment program and within the same Agile development management framework
☐ The Contractor should establish a process to monitor the results of enhancements implemented from the common backlog and report on continued benefits delivered to the Air Force
☐ The Contractor should seek innovations and improvements to process and technology for sustainment and record the outcomes so that these become routine and part of the adopted agile process

Table J: Best practices for managing dispersed Agile teams

☐ Establish team discipline early and enforce consistently
☐ Provide common and consistent processes and tools for remote & co-located teams
☐ Ensure team members are properly trained and equipped
☐ Foster open communication and train team members on effective collaboration
☐ Enforce team discipline through participation in daily, weekly, and other scheduled meetings
☐ Factor in travel for key activities for remote team members and for core collaboration events (for example scoping/design/release workshops)
☐ Use web-based conferencing and collaboration tools, which enable sharing of interactive content and the gathering and recording of working sessions, for project meetings and customer collaboration
☐ Scheduled travel to conduct in-person interview, workshops, and other facilitated sessions.

Table K: Pros and cons of CI/CD options

CI/CD Option	Pro	Con
Govt/BES-provided CI/CD pipeline and development environment	Consistent E2E environment makes configuration management easier Gov can ensure Contractor compliance with required standards and technologies from Dev to Production	Reduces flexibility of Contractor to manage & adapt development and testing environment Gov must be responsive to resolve issues/requests across the entire CI/CD environment
Contractor-provided CI/CD pipeline and development environment	Contractor can set up and manage environments to suit team behavior & productivity Sole responsibility for development & automated testing sits with Contractor	Gov must accept Contractor testing is sufficient to go into Production without Gov-hosted CV&I environment Have to maintain tight configuration management control between Contractor Dev/Test and Gov Production environments
Hybrid approach, where the contractor-provided CI/CD pipeline is connected to a BES-owned and managed CI/CD pipeline that is the single path to release into production	Contractor can set up and manage environments to suit team behavior & productivity Test environment mirrors the final Production environment so should reduce need for final CV&I testing	Require well-defined transfer of release objects from Contractor to Gov environments Need to maintain tight configuration management control between Contractor and Government environments