BEST PRACTICES WHITE PAPER

Optimizing Agile for Your Organization

Jenny Stuart, Vice President of Consulting, Construx Software

Version 1, September 2008

Contributors

Earl Beede, Professional Software Engineer

Jerry Deville, Professional Software Engineer

Steve McConnell, CEO and Chief Software Engineer

Steve Tockey, Principal Consultant

Many organizations are interested in becoming Agile and wonder where to start. They want to ensure that their Agile adoption will achieve the desired benefits, goals, and objectives. This white paper will outline the major organizational, cultural, and project considerations that are critical to a succesful Agile adoption. Construx provides a starting point that works for many projects and organizations.

Contents

Getting Started with Agile

More and more companies are interested in becoming Agile. The motivations for this range from a desire to be more responsive to the changing market or customer needs, a history of challenged or failed projects using more traditional approaches, or an interest in responding to this growing trend in the industry. Whatever the motivation, it can be difficult to know where to start as there are numerous Agile methodologies and every adopting organization has different goals, challenges, and constraints.

Select Your Agile Approach

The first step to optimizing Agile is to select the Agile approach that is the best fit for the adopting organization and its projects. This choice alone can be confusing because there are numerous software development approaches under the umbrella of Agile—from Evolutionary Delivery to Extreme Programming (XP) to Scrum to Lean. It can be difficult to understand which is the best one for any specific organization.

The most fundamental issue to resolve is determining the desired balance between predictability and responsiveness in a development effort. Traditional software development approaches focus on providing long-range predictability of the combination of cost, schedule, and features. Some of the most agile approaches focus on flexibility and responsiveness at the expense of predictability. In the middle of this range are agile approaches that seek to balance responsiveness and predictability. The spectrum of agile approaches is outlined below in Figure 1.

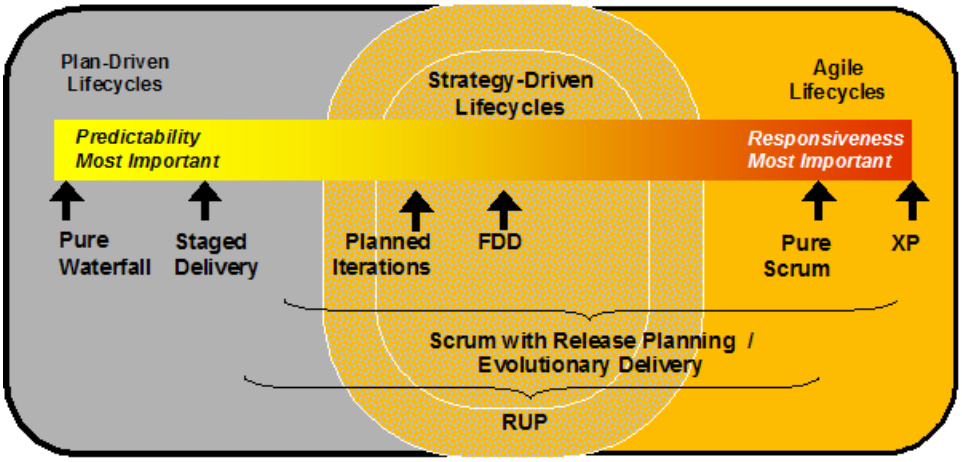

Figure 1 *Summary of Agile Approaches*

If you are unsure of where to begin, Construx recommends starting by evaluating Scrum. Scrum is an Agile management framework for software development and is a good place to start for many organizations for the following reasons:

- As a management framework, Scrum makes is possible for an organization to become more agile while keeping the development practices that are a best fit for the organization.
- While a few core principles must be adopted, Scrum can be extended and modified to work in a number of situations
- It can be utilized with Release Planning and Product Planning if a more Strategy-Driven or Plan-Driven approach is desired

Scrum is currently the most commonly implemented Agile approach and has been used successfully in a wide variety of organizations.

Once the particular Agile approach has been selected, it is important to ensure that the approach is adapted so that it works for the organization. If Scrum is selected, a key to success is implementing the fundamentals of Scrum while also modifying and extending it to meet the specific needs and challenges of the adopting organization. Construx's *10 Keys to Successful Scrum Adoption* white paper provides guidance on how to adapt and adopt Scrum.

Provide Guidance on Using Agile

Many organizations need to use a combination of Agile and more traditional approaches to support different types of projects. For example, a company in a regulated environment may wish to use an Agile approach to conduct projects that deliver software for internal use or nonregulated products, while it uses a traditional approach for developing software that is regulated.

To support the use of multiple approaches in the organization, it can be useful to create an Agile ScoreCard that team members can complete at the start of a project to help management decided whether the project is a good fit for the selected Agile approach. An Agile ScoreCard supports the analysis of individual projects to determine whether or not they should use an Agile approach and how agile they should be.

Some of the areas that should be considered for inclusion on an Agile ScoreCard are the following:

- Sponsor/executive buy-in
- Team buy-in
- Team's experience using Agile
- Team's experience in business area
- Team size
- Level of geographic distribution
- How exploratory the circumstances are
- Level of expected requirements change
- Need for long-range predictability about schedule and features
- Strength of desire for early value delivery
- Availability of customer resources to support Agile
- Requirements familiarity
- Longevity of the software
- Regulatory requirements
- Need for transparency
- Level of coordination required with other groups
- System complexity
- Technology familiarity
- Dependencies on major infrastructure elements

An example of the criteria for a ScoreCard is outlined in Table 1 and an example ScoreCard is outlined in Table 2.

Table 1 *Agile ScoreCard Criteria*

Area	Ranking Description
Project sponsor buy-in to Agile development	1=active resistance; 5=Agile champion
Team buy-in to Agile development	1=active resistance; 5=all team members are Agile champions
Team's experience in using Agile development	1=low; 5=high
Team's experience in the business area	1=low; 5=high
Availability of customer	1=largely unavailable, infrequent interactions; 5=daily face-to-face access
Requirements familiarity	1=very well known; 5=completely unknown
Regulatory requirements	1=high degree of regulatory requirements; 5=no regulatory requirements

Area	Ranking Description
Need for long-range predictability about combination of schedule and features	1=high need for predictability; 5=no need for predictability
Coordination required with other groups	1=high degree of coordination; 5=no coordination, project is independent
Team size	1=50+ people; 5=1-7 people
Team location	1=multiple sites, multiple time zones, 5=co-located in same work area
Consequence of error in this software	1=risk to human safety; 5=nuisance only

Score of:

11-23 Agile development not recommended.

24-35 Likely not a good match for Agile development. Recommend sequential approach instead. If Agile is chosen, project should be viewed as a high risk project.

36-45 Possible match for Agile development. Scores of 3 or lower should be viewed as possible risk areas.

46+ Good match for Agile development. Scores of 3 or lower should be viewed as possible risk areas.

Table 2 *Agile ScoreCard Example*

Area	Points	Comments
Project sponsor buy-in to Agile development	4	Management is cautiously optimistic about Agile.
Team buy-in to Agile development	3	Team is willing to try Agile, but members are not champions.
Team's experience in using Agile development	2	Some of the team members, but not all, have used Agile previously.
Team's experience in the business area	5	Team is very senior personnel.
Availability of customer	3	Customer is readily available, but usually via e-mail; face-to-face only weekly.
Requirements familiarity	5	Project is a research project.
Regulatory requirements	4	Project takes place in a regulated industry, but this particular software isn't subject to regulation.

Area	Points	Comments
Need for long-range predictability about combination of schedule and features	3	Project is viewed as open-ended, so need is more for transparency than predictability.
Coordination required with other groups	5	Independent project; no coordination required.
Team size	5	5-person project.
Team location	5	All on same floor of same building.
Consequence of error in this software	3	Risk is to quality of business decision-making, but there's no safety risk.
TOTAL	47	

Cultural and Organizational Considerations

There are a number of cultural and organizational considerations during a transition to Agile approaches. These considerations may influence the Agile approach that is selected and will certainly influence how that approach is implemented in the organization.

Align the "Customer" with the Teams

Having a strong Voice of the Customer (VoC) is critical to the successful implementation of any methodology. However, as organizations look to become more agile, the need for this tends to come into sharp relief. Many of the Agile approaches require an on-site customer or a customer representative who is involved with the team on a daily basis.

The incremental delivery of software that occurs on either an Agile or a Strategy-Driven life cycle increases the frequency and level of contact necessary between the VoC and the team throughout the software development life cycle. As a result, most organizations find that the move toward Agile increases the organization's visibility into any issues that currently exist related to obtaining, understanding, and validating the needs of the end users. It is important, therefore, to understand who in the organization represents the voice of the customer and to ensure that person can effectively communicate the customer's needs to the team.

The more agile the organization wishes to be, the more it will need to commit to integrating the customer representative into the Agile team. Regardless of whether the customer representatives are Product Managers, End Customers, Business Analysts, or Subject Matter Experts, it is critical to identify how the organization obtains its customer input and to assign a customer representative to each team. Individuals in this role must be able to identify, synthesize, prioritize, and communicate user requirements so that they reflect the priorities and needs of the entire user community.

This focus on integrating the VoC into the team can sometimes be as simple as assigning another individual to the Agile team, or it can be as complex as restructuring the organization and modifying existing processes that separate the customer from the development team. It can be further complicated by the fact that it can be hard to find a single person who is both good at communicating user requirements and can be fully dedicated to the team.

Some of the ways that Construx has seen organizations change to provide a strong Voice of the Customer include:

- Assigning a Product Manager directly to a team as the individual who is capable of representing the needs of the user community and the only person designated to do so.
- Assigning a Business Analyst as the primary contact for the VoC, and establishing a set of Subject Matter Experts (SMEs) who are appointed to represent the needs of different parts of the user community.
- Designating two people to represent the Voice of the Customer. This approach is most commonly implemented by separating the VoC into a strategic VoC and a tactical VoC. For example, in Scrum the Product Owner role could comprise two components: the Release Product Owner (RPO) and the Sprint Product Owner (SPO). The Release Product Owner is the strategic VoC and focuses on longer term strategy, owns the vision, drives the Product Roadmap, and participates in key events during the sprints, such as Sprint Planning meetings and Sprint Review meetings. The Sprint Product Owner performs a more tactical role and is the person who is part of the daily Scrum activities. The SPO also engages the Release Product Owner as needed to make tactical decisions.

Every adopting team and organization needs to determine the best way to establish a clear Voice of the Customer given its unique goals, constraints, and culture. This is a significant issue, as most organizations find that identifying and assigning a customer representative who can provide business guidance and prioritization is the hardest change they will make when becoming agile.

Understand Customer Commitment Implications

Many Agile approaches are based on the assumption that we do not know what 'done' is for the project. Organizations typically take a fixed-budget or fixed-schedule approach, and then the team works to deliver the highest value features within that constraint. This development strategy may be an issue for organizations that wish to provide detailed product roadmaps to their customers.

If a highly agile approach such as XP is selected, it can be difficult to provide customer commitments because the approach is focused on responding to change rather than providing long-term visibility. Although this approach is very appropriate for research projects or exploratory projects, many organizations find that it is not a good fit for a

product release where the organization wants to provide some visibility into future features and functionality to the end customers and the market.

The need to make external customer commitments for certain projects and the level of requirements stability can influence which Agile approach is selected and the amount of up-front time that is spent doing requirements and design work. The more the requirements for a project can be well understood and the more they will remain stable, the less agile the project needs to be, and the more predictable the outcome will be.

One way that organizations often balance responsiveness with predictability is to expand the out-of-the-box Agile approach with some level of product planning. For example, an organization might adopt Scrum and extend it with Release Planning and Product Planning. If the requirements are relatively well understood and stable, the Product Roadmap could commit to high-level or marquee features with a fair degree of confidence for the first release. Later releases would provide a higher degree of uncertainty or commitment to only one or two marquee features.

In evaluating Agile approaches, it is important to understand the necessary level of customer commitments that an organization needs to make, when those commitments need to occur, and what the general level of requirements stability and certainty is on projects. These factors will influence the Agile approach selected and may require changes to the way the organization currently makes external commitments.

Understand Financial Accounting Implications

The assumption that we do not know what "done" is for a project can also be an issue for organizations with a strict annual budgeting cycle. Such organizations make financial decisions based on the expectation that projects will deliver a certain set of features and functions.

When adopting Agile, an organization needs to understand the financial accounting implications of the move to Agile. In some cases, these processes can be changed to enable the organization to become more agile. In other cases, the level of Agile must be selected to work within the budgeting constraints.

One common change that organizations make in becoming more agile is to provide an abstraction layer between annual budgeting and specific projects, such as clear business goals, rather than a specific detailed feature set. This approach enables the team to focus on achieving the business goals and, at the same time, provides the team with the flexibility to change the means of meeting those business goals. This approach can be coupled with a specific budget or target delivery date to ensure the business has predictability on those aspects of the project.

Major Project Considerations

In addition to cultural and organizational considerations, some considerations need to be made at the project level for a successful transition to Agile.

Learn to Decompose Project Work

When an organization is adopting Agile, teams can have difficulty understanding how to decompose the work so that the pieces are small enough to complete in a single increment. This focus on smaller segments is a significant shift from creating large requirements that could take months to complete. In most Agile approaches, the team begins by discussing the highest priority features and looking for small requirements that can be completed in one increment. Depending on the methodology selected, this can mean looking for work that can be completed in the span of one to six weeks. Even beyond this, the detailed work for an increment tends to be decomposed into tasks that can be completed in the span of half a day to two days.

For large and complex systems and for people new to Agile, this shift in thinking can be difficult. When adopting Agile, an organization needs to understand the significance of this shift and ensure the team has the support it needs during the transition and is given time to go through the learning curve. It is equally important to focus on having the team implement end-user-visible functionality starting from the first increment, as this is one of the most valuable changes in the shift to Agile.

Understand and Address Architectural Implications

Becoming more agile does not mean ignoring design and architecture; rather, the process changes how teams and the organization approach design and architecture. The specific Agile approach that is selected will have an impact on how the organization oversees the architecture.

More Strategy-Driven approaches, such as Feature-Driven Development or Scrum using Release Planning and Product Planning, spend more time early in the project building architectural models for the system. With XP or Pure Scrum, the architecture and design of the system emerges as the project progresses.

In moving to Agile, team members need to understand the level of architectural oversight and compliance that is required by the organization. In some organizations, the team simply needs to oversee the architecture for the product and does not have to comply with external constraints. In other organizations, regulatory compliance constraints require that the design be documented and traced to the requirements and to test cases. In other organizations, there are architectural standards that must be met and reviews that validate that compliance.

When adopting an Agile methodology, the organization needs to understand the architectural implications of the Agile approach that is selected and determine how to ad-

dress them. The constraints within the organization do not typically mean that a specific Agile approach cannot be used, but rather that changes need to be made from the out-of-the-box approach so that it works for the organization.

Some examples of how organizations have addressed this issue include

- Assigning an architect to the team who oversees the architecture and design work and is a liaison to external oversight committees to ensure the architecture is solid and fits within organizational or system architectural guidelines
- Allowing the architecture and design to emerge during the project
- Using the first one or two increments to shape the general architectural direction
- Incrementally capturing the overall system architecture and high-level design documents during the project

When adopting an Agile approach, each team must identify the architectural constraints and implications of the change and determine how these will be addressed. Because it is not uncommon for organizations to have different types of projects that require different levels of architectural oversight and control, what the organization requires may vary from project to project.

Increase Investment in Supporting Infrastructure

Most Agile approaches release software—either an internal or external release—on a more frequent basis than waterfall, staged-delivery, or other plan-driven approaches. This frequent release of software (or systems) requires that there is sufficient supporting infrastructure in place to ensure a high-quality product is being released. When adopting Agile, an organization might need to invest some time and effort to improve or establish the infrastructure and practices that are critical for Agile development.

Common infrastructure investment areas include

- Establishing ongoing daily or continuous builds that are validated with an automated smoke test
- Implementing an automated unit testing framework
- Building automated system-level regression tests
- Automating the release and promotion process to fully automate software releases

Establishing this support infrastructure is necessary to ensure that the system is always integrated, deployable, and at a known level of quality. Without this level of assurance, it can be difficult to ensure that the software being built is sufficiently stable and at the necessary quality level to meet the Agile criteria of delivering software at a near releasable state at the conclusion of each increment. Over time, the lack of this infrastructure can reduce the productivity of a team because they end up spending time investigating defects that are introduced when new features are added, changes are made

to existing features, or the design and architecture are updated based on end-user feedback on the system functionality.

Most organizations that are adopting Agile do not have the luxury of starting with the development of a new system. Because completely retrofitting existing systems with comprehensive unit and system tests is impractical, it is important that the adopting organization have a plan to begin building this infrastructure for existing systems.

The most effective approach is generally to start by fully automating the daily build process and then to begin incrementally adding automated unit and system tests. This may require implementing automation tools for both unit and system testing if they are not already in place.

Construx has seen numerous organization start by requiring automated unit tests for new development while selectively back-filling areas that will benefit the most from automated regression tests. In support of this approach, we recommend that organizations profile their systems based on a set of criteria to identify the most important areas to both put automated testing infrastructure in place and to back-fill tests.

Keys to a Successful Transition

Construx has worked with numerous project teams and organizations to implement specific Agile practices and adopt an Agile approach. We have identified some keys to success that have helped organizations effectively adopt these new approaches.

Make Use of Pilots

It is generally more effective to begin an organization-wide adoption of Agile with one or more pilot projects. Pilot projects are an opportunity to try the selected approach on a trial basis within the organization to determine what works well out of the box, what needs to be modified, and what additional practices or processes are necessary to make the selected Agile approach work within any organizational constraints and with the culture. Pilots are a chance for an organization to learn in a small deployment and tailor the approach before a wider deployment occurs.

The number of pilots necessary depends on a number of items, including the amount of variation in the projects that occur in the organization, the difference between the approach the organization uses today and the new approach selected, any variability due to geographic distribution, and the amount of experience that individuals in the organization have with Agile. It is not uncommon for organizations to conduct at least two pilot projects to determine the major changes that need to occur and then incrementally deploy the new approach throughout the organization.

Use a Coach During the Transition

Many organizations that are interested in becoming more agile have historically used a traditional software development life cycle such as waterfall. Often, individuals and teams making this shift are concerned about how to use the Agile approach, how to implement the organizational or team structure changes necessary to make it successful, and what specific modifications to the Agile approach are necessary to make it successful.

Assigning an internal or external coach to a team during the transition is a very effective way of helping the team move quickly through the learning curve. Attributes of a good coach include that they

- Have a deep understanding of and experience with the Agile approach being adopted
- Have experience helping individuals and teams implement new practices
- Are respected within the organization *(which is especially true for internal coaches)*

When Agile is being adopted across an entire organization, the coach is used whenever a team is formed by people who are not experienced with Agile.

Implement the Process First, Tools Will Follow

The field of software development has seen, over the last 20 years or so, a series of tools, methodologies, and languages that were going to revolutionize how we develop software. Many of these have been successful in improving productivity. However, the improvement has been an incremental improvement rather than an order of magnitude improvement. Although good tools support effective software development, they are really only an aid to doing the underlying work—for example, eliciting, analyzing, and organizing requirements or using the Scrum approach.

In the case of requirements, the process of first defining the problem to be solved, agreeing upon a decision process to help choose among competing solutions, and negotiating how to select the essential few requirements from the trivial many are the primary challenges. Documenting requirements in a tool such as Doors or Requisite Pro is a small part of the overall effort necessary to figure out what the requirements are, review them for completeness/consistency, make sure they are detailed enough, and ensure they satisfy users' needs. Similarly, when implementing an Agile approach, the main challenges are understanding how to decompose work into small chunks, adopting new roles and ways of working together, agreeing upon what it means to bring software to a near releasable state, and implementing the infrastructure necessary to support frequent releases.

Most people who have studied tool implementation suggest working on an organization's process first, and then finding a tool to support that process. The tool is ineffective without a process for using it effectively and efficiently.

When adopting Agile, it is generally more effective to determine the approach that will be used and implement the software engineering practices that support its effective application in the organization. Once that has been completed, the organization can look for a tool or suite of tools that support the processes.

Learn from Experience

While implementing an Agile approach it is important to be prepared to learn and adapt it based on experience. It generally takes a few increments to tailor and expand the approach so that it meets the unique needs of the project team. Most Agile approaches use short increments which provide a built-in opportunity to learn from experience, make refinements or modifications, and validate that the desired outcome is achieved.

Retrospectives are a powerful technique for learning by experience and improving during a project. A best practice is to conduct a retrospective at the conclusion of each increment to identify the following:

- What went well
- What should be changed to improve the team's productivity and effectiveness
- One or two changes the team wants to try in the next increment

If the increment was a month in duration, the first few retrospectives typically last about half a day. As the team becomes established, retrospectives can become as short as one or two hours. Shorter increments may have shorter retrospectives as well.

When adopting Agile across an organization, retrospectives from individual projects should be leveraged to provide feedback such as suggesting changes that can be made to improve the entire Agile adoption and sharing best practices that can be used across teams. Some common approaches include using brownbag lunch meetings or lunch-and-learn sessions across teams, providing feedback to a centralized team that is supporting the change effort, and including people from teams that are starting to use Agile as observers at the retrospectives.

Once the initial adoption is complete, it is still important for the organization to continuously learn from the experiences of the Agile teams and adapt the methodology. What works perfectly for the size of the organization, complexity of projects, and external constraints today might not meet the needs of the organization in the future.

About Construx

This white paper was created by Construx Software Builders, Inc. Construx Software is the market leader in software development best practices training and consulting. Construx was founded in 1996 by Steve McConnell, respected author and thought leader on software development best practices. Steve's books *Code Complete*, *Rapid Development*, and other titles are some of the most accessible books on software development with more than a million copies in print in 20 languages. Steve's passion for advancing the art and science of software engineering is shared by Construx's team of seasoned consultants. Their depth of knowledge and expertise has helped hundreds of companies solve their software challenges by identifying and adopting practices that have been proven to produce high quality software—faster, and with greater predictability.

Steve McConnell, CEO/Chief Software Engineer

steve.mcconnell@construx.com
+1(425) 636-0100

Jenny Stuart, VP Consulting

jenny.stuart@construx.com
+1(425) 636-0108

Matt Peloquin, CTO

matt.peloquin@construx.com
+1(425) 636-0104

Steve Tockey, Principal Consultant

steve.tockey@construx.com
+1(425) 636-0106

Mark Nygren, COO/VP Sales

mark.nygren@construx.com
+1(425) 636-0110

For more information about Construx's support for software development best practices, please see our website at www.construx.com, contact us at consulting@construx.com, or call us at +1(866) 296-6300.

Construx Software Builders, Inc.
10900 NE 8th Street, Suite 1350
Bellevue, WA 98004
U.S.A.