Adopting an Agile methodology
Requirements-gathering and delivery
# Table of contents

1. Introduction .................................................................................................................. 1
2. Traditional methodologies requirements-gathering process ........................................... 2
3. Agile methodologies requirements-gathering process ...................................................... 3
4. Reaping the benefits of Agile ....................................................................................... 10
5. Introducing Agile to your organization and building a roadmap for success .................. 13
6. Conclusion .................................................................................................................... 17
1. Introduction

Gathering requirements is the cornerstone to any successful project. Requirements are critical for scoping, defining, estimating, and managing the project. In projects employing Agile, achieving desired results requires a measured approach that may be unique to each organization. For organizations with little or no experience employing Agile methods, the risk is high for delays and overruns. In this comprehensive overview of requirements gathering and project delivery, we explore an approach and framework that can help drive success through open communication with stakeholders around an active project management framework designed to best fit the organization.

In a global economy, organizations are forced to adapt quickly to evolving technology and a fluid corporate landscape. Few of these companies have shown the capability to quickly pivot their software production or manufacturing units to consistently deliver end-user products that are of high quality and on schedule. The capabilities required of a high-performing organization to provide increasing development velocity with frequent and effective communication cannot be implemented overnight. It requires an organizational commitment to change and cultural adaptability for progress.

The trend across industries is toward the Agile methodology of development; it provides a greater flexibility within a development unit to confront a constantly changing market demand and competitive landscape. A company needs to maintain a high standard of work, focus on cost reduction, and promote alignment between business and information technology. The evidence demonstrates how this shift has continued to develop and, as The Economic Times wrote at the end of 2012:

“While less than one percent of IT departments had even heard about Agile in 2000, Gartner and Forrester statistics indicate that 60-80 percent of software development teams now use Agile as their primary method for creating software. This is higher than in 2009, when Forrester research said 45 percent of development teams used Agile methods to write code or create products.”

While a greater number of corporations are aware of Agile and are employing its concepts to develop product, Agile will not provide the anticipated “turn-key” solution without proper planning and communication.

One of the key differentiators for the Agile methodology is the requirements-gathering process and the iterative approach that fosters alignment through increased communication between IT and business. In a more traditional approach, requirements-gathering and analysis can become a lengthy process that can last a year or more. A project team spends the time trying to define every detail of how a system should work prior to designing the product.

On the other hand, Agile provides a high level requirements process to define what the system should provide, not how the system should work. The “what” is required to start development early and show progress and, as the project continues, drives alignment between the success criteria and the end result. The iterative process promotes communication between and within teams to establish how the system should work. Iteration fosters communication between IT and business, software developers and business analysts, and creators and end users. The development teams gain greater insight into what is required and a better understanding of the desired outcome. It is acceptable for developers to contribute to the requirements gathering process. The goal of an Agile requirements gathering and delivery process is to promote communication, alignment, and early customer involvement to meet the goals and needs of the organization.
2. Traditional methodologies - requirements-gathering process

With technology evolving at a dramatic rate over the past several decades, organizations have a growing need to adapt to faster product life cycles while producing a higher quality of work just to stay on par with industry peers. Improving the customer experience through technology is now a cornerstone of market success. This cycle of continuous improvement is required for companies to keep pace and continuously enhance their platforms.

The migration from traditional requirements-gathering processes to Agile or other iterative processes can be challenging and confusing for an organization. The shift from document-centric and business requirements-centric processes can be difficult to adopt when the communication of the new processes is not clearly outlined by either the business or the IT organization. Time-honored methodologies, such as Waterfall, have seen a decline in use due to the compulsion for lengthy change processes and extensive documentation. Traditionally, using Waterfall-style methodologies could stall development by involving a lengthy requirements-gathering process or phase with heavy business interaction. Once completed with the requirements-gathering, the business would retreat from the project, leaving minimal interaction with the development team until project completion. Because the pace of business is constantly in flux, the end product could no longer meet the needs of the business if there isn’t continuous communication.

These traditional SDLC requirements processes provide a number of documentation formats for capturing both “functional” (what it does) and “technical” (how it’s built) business requirements. A project team ends up spending months if not over a year in the requirements-gathering phase, then additional months discussing and gathering technical requirements in the design phase, all before anyone starts coding. Eventually, due to the governance surrounding the typical Waterfall approach, the development team is handed a lengthy document that tells them what to do. They are not given any input into the design. Ultimately, lack of communication from phase to phase between the business and the development team can lead to a misinterpretation between what is written and where their client needs truly lie. Creating a product that does not fully fit the needs of the organization can cause additional expenses, constrict their resource plan, and cause significant delays in deployment. The risk of a Waterfall-style methodology is that removing the critical aspect of communication and transparency can lead to costly delays in product releases and delivery of a lower quality product. As Tom Mellor of State Farm points out:

“It was often futile for the business to try to develop complete requirements upfront, and as a result there were many problems that resulted from people not getting what they needed or wanted. I knew there had to be a better way than to think we could look into a crystal ball in new product development.”

Displaying transparency and finished product early in the development process frees the organization of the rigor and structure of a phased approach. It relieves the pain points and down time most organizations face when dealing with rigid structure. Waterfall’s lack of transparency will lead to vague status reports regarding where the development team is in terms of percentage complete and how the end product is taking shape. By providing functional requirements upfront using Agile, the development team will begin to understand what the product could do and why it is should be built. Providing the technical requirements in an iterative process provides business analysts and developers the opportunity to deliberate the best possible way to deliver the requested features. The enhanced communication through a skilled cross-functional team results in a higher quality product and increased satisfaction from the end-users.
3. **Agile methodologies requirements-gathering process**

The timeframe required to look into a “crystal ball” is quickly closing as the rate of technological advancement increases and as customers are expecting a quicker turnover in a product cycle. The below image demonstrates an example of the difference between pursuing traditional Waterfall methodologies versus an Agile-style system implementation:

The objective of using Agile is simple: deliver higher quality products, promote collaboration through an incremental process, and optimize program budgets. Agile is intended to promote transparency within the development process, respond quickly to inevitable changes in business requirements, and deliver functional software. Agile recognizes that an implementation project will experience bumps in the road. Agile actually expects it. The development team is prepared for iterative change, and the requirements-gathering process should reflect this reality.

Not every change is good for all members of a development team. The iterative approach to requirements gathering impacts the QA team more severely. Test scripts, which are often written in parallel to business requirements and development activities in an Agile environment, require strict adherence to a static processes. Vigilant project management and knowledgeable business stakeholders will need to clearly define scope in order to minimize re-work by the project team. Managing scope creep, even inevitable changes along the way, can go a long way in minimizing the impact to the project budget.
While an Agile approach is designed for flexibility, it is important to understand the critical need for structure and project planning. The scope of work is broken into separate buckets with the largest, called Epics, defining the major items of development. The Epics are then segmented into smaller pieces of work, called Stories, and within the Stories, individual tasks. Project planning within Agile requires careful consideration of dependencies between each of the Epics and Stories, then prioritization based on business demand. Finally, the Epics and Stories are slotted into blocks of development time, called Sprints, for development. The project team must have a clear understanding of this framework and stridently fight to maintain the sprint scope based on planning estimates of team capacity and project scope.

The project team develops estimates in an inception phase around planned deliverables, which tie back to their business case. Within the inception phase, the team is busy allocating time to develop a high-level overview of business requirements, establishing project controls, and communicating with the stakeholders and other business leadership. This phase will set the stage for how the project will move through the remaining phases of Development, Testing/Stabilization, and Deployment. The high-level overview of the requirements allows a project team to adequately scope the project. This scope will drive how stories are grouped into the sprint plan. The development team must be realistic in its approach in order to accurately assess total effort required to deliver the required functionality within each sprint. As stated earlier, the development team expects the business to change. By accounting for future iterations of business requirements in later sprints, the team prepares for unforeseen future improvements to the product. These iterations are not considered a distraction, but a healthy and necessary part of the requirements-gathering process. It should lead to a better product.

Another critical outcome of the inception phase is a complete list of stories. To reduce future rework, it is important to systematically gather a complete picture of desired functionality, integration points, internal/external stakeholders, and potential dependencies and impediments that are external to the project. Accurately pinning down scope early in the project allows for more precise estimating of the overall effort, which would remediate project delays and overflow stories into future sprints.

Requirements gathered within the inception phase will typically not be as thoroughly detailed as the completed requirements that come from the development phase, but it is in a development team’s best interest to capture as much as possible during this inception phase so that they are prepared for development. The information collected will serve as a framework for the detailed requirements-gathering process within a given sprint. In conversations with business stakeholders, the team must document what the business partners consider as important functionalities for the end product. In addition, the team should remain vigilant about documenting discussions of processes and functionality that are out of scope. These scope-stretching topics and functionalities will be raised again during detailed requirements-gathering. Being able to quickly point toward details from that previously made decision can save the project from a time-wasting digression. This is not to say that the scope will not change during the project (after all, the scope itself can be somewhat iterative), but addressing scope creep when it arises and referring to the original decision can streamline conversations and place the team back on track to complete the requirements-gathering process.

As we discussed earlier, a shift will occur in how and where requirements-gathering takes place in the Agile process. Business analysts first using Agile may find themselves in uncharted territory; the pace and expectations are vastly different. Waiting days for answers from the business gets cut down to hours in many cases.

It is important for the business to provide skilled subject matter experts (SMEs) with the authority to make decisions on a tight timeframe to eliminate gaps in development and team velocity. The right person for this type of
role is one who can apply logic and critical thinking, while still reaching a conclusion within a reasonable timeframe. For those decisions requiring elevation to superiors, the SMEs must have the understanding of the business to lay out the options and provide insight into how the decision would impact the project. The business sponsors should supply resources with a balanced understanding of the vision for the project, the decision-making ability to shape the vision, and the support and direction from the organization to see it through.

Along with skilled resources at the SME level, the business should provide a proper framework to scope the scale of the project. Within the requirements gathering process, the business is responsible for supplying the following:

**Assumptions:** Data that will not be subject to user changes

- Scenarios for which this application will be valid. For example: ask the question “Is this application or feature only to be used for a certain product, line of business, state, or transaction?” Make sure the business accurately defines its use. If the solution is not to be used for every situation, then get clear agreement as to the boundaries.
- Determine the source of data that will be used within an application or system. What data will be sourced from another system, and could that cause complications? Often, this type of data is used as a reference and may have to be modified only in the source system. Nailing down the data that is in scope will help quantify the scope of work.

**Definitions:** Required data fields for the functionality

- What data elements are needed to process rules or make decisions? Data gathering must be evaluated on a need basis. If a rule is not using a data element, then the team should question the importance of the data. A team should always balance the usefulness of the data against the effort required to obtain it.
- Construct the data fields with standards in such a way to limit user error. Use of conditional fields, formatting, and validation can effectively drive data accuracy.

**Process Flows:** Define user input and system response

- Set standards for inputting data in a specific order and make key decisions defining logical screen layout and flow. The team should not be afraid to challenge the data flow in the current system—the flow in the current process or system may not be the most logical or efficient.
- Clearly defining the desired system response helps frame data field specifications. This will help in clarifying required elements and formatting requirements.
- Defining the rules properly is the path to increasing the accuracy with which the process actually does happen. Rarely are new rules being created, so the rules that are defined are typically what the process owners and SME’s expect to happen. The discussions will uncover the differences between how the rules should work, and hopefully lead the project to work within certain process boundaries. Be wary—these discussions can clearly become a breeding ground for scope creep. Requirements-gathering workshops can open the door to a flood of new ideas. The goal is determining which ideas fit within the defined scope, which ones should be evaluated in the change order process, and which ones are candidates for future releases.

**Screen Descriptions:** Verbal description and graphical prototype of the screen

- Simple screens can often be covered through basic field definition. Unless the organization is requiring a particular screen layout, allow developers the input into aspects of design. The developer may have insight into methods of design and configuration.
- A prototype may be the best way to get agreement and signoff on the requirements when dealing with more complicated screen configurations. It may also smooth the communication process with the developers. The screen prototype, if thoroughly detailed, could be the requirements artifact.
Business Rules: Validation rules and data types for each user action

- Improving the accuracy of data input should reduce future re-work. A variety of tools are at the disposal of a development team. If a field is determined to have value, then careful consideration needs to be given to all likely scenarios. The conditions that govern data entry into specific fields should be outlined in detail. It should be clear whether or not certain data entry should be required at all times. If it is not required, then what business rules cause the field to be conditionally required and/or conditionally displayed to the user? Working out these questions and others will bring standardization across the platform.

- Use design options to guide the user. Field labels should be thought out and titled with care. They should be succinct, while providing enough description to avoid misunderstanding. The type of input for each field should be readily apparent to the user. A field that requires numeric input should be labeled as such or with enough clues to avoid improper data entry. Hover text can help with providing explanation, but must be used judiciously--too many pop-ups on a page will distract a user and become a nuisance over time.

- Displaying error messages upon saving screen changes should be used only when no other viable alternative is available. Efficiency improves with the reduction of error message instances.

Considerations for packaged solutions

- Don’t re-invent the wheel. Take advantage of developed out-of-the-box (OOTB) functionality to accelerate the requirements-gathering process.

- With a packaged solution, requirements should focus on the deltas from the OOTB product. Writing requirements for OOTB functionality that is to be retained in the final product increases the likelihood of the developers and testers discovering errors. There is no need to introduce extra work through the requirements process. The assumption made, in this situation, is that anything not covered in the requirements will remain unaltered from the base product. After all, the organization selected the product because at least a fairly significant amount of its OOTB functionality met its business needs.

- Make sure that each change from OOTB product is sufficiently justified by a business case. Does the OOTB functionality not meet the business’s needs? Encourage discussion between the development team and the business to determine the real business value for the change. If the change provides value, then prioritize it accordingly within a given sprint. If the value is not apparent, then discuss the costs of developing the change against other scheduled features. Also, drive home the concept that the new system does not have to look and feel like the old one. The decision was made to replace the legacy system for a valid reason.

- Not all requirements are system based. Instances will exist where a simple process re-design can provide a solution to a problem. A business analyst plays an important role in determining whether a stated business need is a system requirement or a process consideration. When implementing a new system, the organization should not shy away from opportunities to retire old processes for new ones that potentially increase productivity and efficiency.

Agile processes will promote understanding between the business and IT by laying out clear expectations at each checkpoint of the project. The first step of bringing synergy to development starts with a conversation earlier to allow business and IT to provide input into potential solutions. The business is able to clearly outline expectations by confirming the solution that the development team is going to deliver. The confirmation step will also provide a chance for the development team to outline assumptions and define the required data to be utilized for delivering the product functionality. It will be important for the business analysts (BA’s) to act as the voice of the business throughout the development step in order to confirm the business needs are being met and to resolve any issues early in the development phase rather than waiting until a demonstration. A best practice during the development phase is for the BA’s to define the process flows for user input and changes.

The demonstration of the system will provide the development team with the opportunity to show off the features delivered in a sprint while the users get the opportunity to view the product’s functionality. The business gets to validate that the needs set by the business were met through the development process. If the business is not
satisfied, then the organizational stakeholders should provide input into identifying the necessary corrections. Misses that are discovered during the demo should not be viewed as failures, but rather as opportunities that can be leveraged to improve the existing requirements and possibly refine the requirements-gathering process going forward. The misses could be a missing requirement, an invalid assumption, or a breakdown of the communication process. The current requirements-gathering process should be reassessed to evaluate the level of detail covered in the requirements discussions, the capability of the Subject Matter Experts and Business Analysts to understand business process, the level of input from the proper decision-makers, and the detail captured within the requirements themselves. Misses are chances to fix a problem early in the process prior to releasing finished code into a production environment for end users.

As we described earlier, the Agile development process is broken into sprints, and the work broken into epics, stories, and tasks. Development is a fast-paced and iterative process by which the cross-functional development team interacts to build requirements, develop code, and test features. The image below illustrates the high level process for how a sprint will flow.

The figure above presents the iterative process within a sprint. During the inception phase, which is prior to development sprints, high level functional requirements will be gathered to provide requirements outlining what the system is being built for and what functionality is being delivered (e.g., “As a customer, I want to perform a [specified task] in order to achieve [goal]). This process is meant to give the development team a starting point, but not all of the detailed requirements. For example, during inception, a requirement for the need to capture an additional data element may be identified. While that is likely a sufficient level of detail for inception, more details are needed for development. During development for this topic, specific details will need to be defined. Examples of these details may include formatting, determining how the data is to be populated, screen location, and any potential validations of data entered into the field. Once stories for a sprint are determined, collaboration between BA’s and developers will be vital to providing detailed requirements for how the system needs to function. This is an important distinction between Agile and traditional methodologies because by providing detailed requirements
for how the system should work in an iterative process it gives developers a chance to provide input from a technical perspective and provides users early and frequent interaction. The benefit is that users are seeing the design and functionality much earlier in the process rather than operating outside the black box of development.

The timing of completion of the requirements must be given careful consideration. While each sprint tends to be viewed as a complete unit of work, some front-loading of the requirements is necessary in order for development to begin on Day 1 of each sprint. This suggests that the requirements documentation and sign-off must be complete for at least some pieces of the planned work for each sprint prior to the start of the sprint. Typically, the last week or so of each sprint should be devoted to preparations for development work to start promptly at the beginning of the next sprint.

Several key steps and rules have been defined by the Executive PMO Council to keep in mind when approaching the requirements-gathering process:

- Gather detailed requirements by defining technical specifics of the features to be built.
- Force prioritization and keep the total number of points constant – If business partners add to scope, reprioritize a comparable functionality.
- Prioritize capabilities according to user needs and value.
- Re-scope or cancel requirements if the effort to build exceeds benefits – Revisit the backlog to ensure that the business case for building the rest of the capabilities is still relevant and sound.
- Front-load pieces of functionality that users can react to (e.g., interface, data displays, reports) to address the “I know it when I see it” nature of the client-needs articulation. Allowing users to see early on what capabilities they are getting reduces rework costs since functionality has not yet been integrated with many systems. “Build buy-in and create evangelists.”

The output of the above steps will be a final deliverable capable of demonstrating the newly developed functionality. This step will provide users the chance to see how the technical requirements that are developed operate in the product. This will lead to their ability to ultimately sign off on this functionality or to revisit the requirements to improve the functionality.

Three key aspects to the requirements management process documents are clear documentation, context, and control:

- Requirements documentation may take many forms, including spreadsheets, diagrams, text documents defining requirements in simple language, or use cases where particular scenarios are defined together with required outcomes for each scenario. (Most projects will include some combination of documentation methodologies.) Measures of success include the ability for each key stakeholder – business decision-maker, developer, and QA tester – to have a common understanding of the expected functionality based on documentation content. The content, not the format, is the driving force here.
- Clear documentation provides the “what” and context provides the “when” and “where.” Context helps to set the stage for the conditions, screens, timing, and users for which particular functionality will be applicable. Screenshots (actual or mock-ups) and workflow diagrams can be helpful in telling the story.
- The control component allows for a foundational understanding between business and IT. Control supports proper documentation and traceability throughout the project. While Agile clearly does not require the volume of documentation as found in traditional approaches, the faster pace of change requires discipline in order to adapt to the continuing requirement changes through development and QA.

Tracking the requirements development process is important to keeping the process moving forward in accordance with the project plan. The tracking process allows for communication across the team about the status of the development of new requirements, as well as updates to existing requirements. The tracking method can take the form of a simple spreadsheet or may become more sophisticated based on the availability of tools. Whatever
method is chosen, however, all members of the team should have ready access and a solid understanding of how it works. Assignments for ownership of specific requirements documents to BA's, along with targeted due dates, clearly define responsibility and allow for development teams to adequately allocate resources. Identification of dates for completion and sign-off will allow developers to know when their work can begin. Setting dates for requirement updates will allow the QA team to monitor changes so that test scripts can be revised.

Additionally, an effective defect management process is one of the cornerstones of effective Agile projects. It is imperative that everyone in the project has the opportunity (and obligation) to report a defect. This is not simply a function of QA. Effective defect management starts with adequately defining the defect. This will include describing the steps used to discover the defect, the expected outcome, and the actual outcome. Screenshots are clearly one of the best methods to demonstrate the issue. Anyone reporting a bug should be encouraged to outline any potential solution. Once the defect has been reported, proper categorization will drive how the team manages the defect. For example, defects that are examples of functionality not working as defined in the requirements are to be handled as bugs and assigned to developers for remediation. Defect mitigation should be prioritized based upon severity, but in some instances stopgap measures can be applied. The definition of bug severity and expected turn-around timelines by severity should be agreed upon at the outset of the project.

Items that are reported as defects, but are not defined by non-working functionality, are typically enhancements. These can take the form of missed requirements or simply an improvement opportunity identified by the business during a sprint. Once again, a severity approach should be employed to determine whether the additional work required should be treated as a significant value add, a “nice-to-have,” or not a meaningful addition. Collaboration between the project team and the business will be necessary to make decisions on these items. The perceived importance based on value to the organization must be weighed against impacts on project timelines and budget.

**Incorporating offshore or remote teams**

With the ever-expanding opportunity to leverage the skills of offshore resources to reduce total development cost, evaluation of how this staffing model works within the Agile framework is worthy of discussion. The ability to tap into a greater network of talent can only serve to broaden the skill set of the project team. The wealth of experience and knowledge can be added to the team and, when properly managed, can improve overall team performance.

Not having the team together in one physical location can create risks. One consideration to mitigate these risks and facilitate communication between onshore and offshore is to use a web-based Agile project management tool such as Rally, VersionOne, Scrumworks or something that generates reports to keep track of user stories and their progress electronically. Agile project teams are often collocated in one area to promote communication. Utilizing offshore resources may enhance the value proposition, but may also introduce some risk to the project. It is important for project leadership to employ management techniques that minimize the risk of communication breakdowns, but maximize the return of lower labor costs and around-the-clock development. Employing a partially overlapping schedule between the onshore and offshore teams can improve productivity with the proper management in place. If the members of the team are working at the same time for at least of few hours of the day, then off shore team members can attend some meetings together and build team rapport. These overlapping schedules can be used to keep working moving forward. Requirements gathered one day can often be developed overnight by the off-shore team and be ready for review by the local project team the next morning. Consider using the offshore team for defect management; having the offshore team review outstanding defects and perform some bug fixes can greatly improve team velocity. Implementing a continuous cycle of work with offshore resources is attainable, but only with active involvement by management and constant communication among all resources.
4. Reaping the benefits of Agile

**Key benefits of Agile methodology when gathering requirements**

The following are five of the major key advantages of gathering requirements using Agile methodology:

1. The project scope is tightly managed, but allows for adjustments as the business adapts to change and discovers opportunity to reprioritize the functionality they need when gathering requirements.
2. **Fully functional software** is delivered quickly in 2-4 week sprints to enable visualization of requirements.
3. High level of collaboration and communication between business and development pods increases the quality of the end product based on high quality requirements.
4. The frequent deployment of code allows business users to see the end product as it’s developed and confirm that requirements are documented correctly.
5. Team resources (developers, testers, architects) are utilized throughout the entire delivery process to make sure developers and testers are in sync with business analysts who document requirements.

The benefits of Agile include fostering communication between business and IT through demonstrations and structured meeting schedules, adding additional transparency into completed and to-be-completed work, and speeding up the development timeframe. While Agile provides significant advantages over traditional methods of development, it does have its pitfalls. If not properly managed, the disadvantages can be readily apparent. The iterative requirements process can expand scope beyond the initial estimate. The demands are high from the SMEs and decision makers. If decision making is not made and approvals are not reached within the tight time frames required by the sprint, then deferred work will pile up and threaten timelines. In addition, not having fully allocated resources or a continuous rotation of resources on the development team can impact team performance. Project management is crucial to avoid these predictable negative outcomes and to facilitate the successful performance of the project and the project team.

One of the biggest benefits of Agile is having weekly, even daily, updates that allow users to adjust their requirements in order to achieve their desired goals. The additional transparency provided in an Agile environment allows for greater user input throughout the development lifecycle. The iterations allow for problems to be fixed early and to reduce the possibility for rework late in the development process. Plus, if something does fall through the cracks, then it would likely be identified during the bi-monthly or monthly demonstrations. This process should save cost in the long run while providing soft benefits, like when the scheduled demonstrations create excitement for the eventual release.

As we mentioned earlier, the collaborative environment promotes communication between business and IT. Creating structured daily or weekly meetings with stakeholders fosters relationship development with a focus on developing the product together. The small cross-functional teams clustered together on the work floor forces quick interaction on key issues. While collocation has minimal impact in a waterfall environment, it is a necessary feature in Agile. The collaboration required throughout the requirements process creates a “we” environment that supports a goal-oriented team. The benefit to having a team working together is not always quantifiable, but the synergy created by having BA’s and Developers working in close quarters to provide the best solution is going to be invaluable to an organization as they tackle larger and more complex projects.

The benefits are not only to the collective team, but to the growth of individual team members. BA’s will gain a greater understanding of what is possible for development to create and be able to better influence the business for what is possible and realistic. Developers will inherently gain greater perspective to what the business is trying to
do and potentially offer solutions that the business did not know were a possibility. The benefit provided by parallel requirements gathering, development, and testing generates a focus on providing a working solution that is only successful if the team is working together rather than focusing on individual achievement.

Traditional methodologies are clearly subject to the misunderstanding that SME’s are able to fully grasp the end-to-end solution without seeing developed product. This leads to a significant gap in understanding and, ultimately, a gap in the finished requirements documentation. Agile allows the SME’s to synthesize their knowledge of the business needs with their ever-growing understanding of how the solution will work. As the project progresses, the project team and SMEs will increase the speed in which they come to a correct solution through a better understanding of the end product. Allowing the team to see the fruits as well as the missteps of the early work promotes an environment of continuous refinement and improvement.

Not following the Agile principles outlined above increases the inherent risk that fluctuations in resource availability will cause cascading scheduling impacts over time. There is a risk existing within waterfall when there is a span of time between the gathering of requirements and development. If the gap becomes too large, then those who participated in gathering the requirements will not be available during development. This can lead to lost time as replacement resources attempt to get acclimated to the project, understand the rationale behind decisions that were made, and work to provide meaningful input to solve issues and address questions. If the replacement resources disagree with previous decisions, then this can serve to further delay the project. While Agile clearly does not fully mitigate this risk, it does help to reduce it through the rapid development cycle and iterative change.

Agile does introduce other areas of concern that require constant attention from Agile leaders. One of the biggest areas that can lead to trouble implementing Agile during the requirements phase is inadequate structure around change control and project management. Because of user interaction throughout the process, Agile can encourage excessive requirement changes that can invoke scope creep into a project. The impact for excessive change to requirements impacts a development team new to Agile in a greater way since they are not likely to have been developing their code in a modular format more adaptable to requirement changes. Because of the risk to impacting project timelines and scope, there is a need for BAs and project management to determine the difference between “nice-to-have” requirements – those items that, if capacity permitted, could be included in development – versus necessary requirements.

Separating those “nice-to-have” requirements is easier said than done during the high-paced iterative process. Based on cost constraints, business partners must separate “nice-to-have” features from the functionality required to go-live, or request an increase in project budget. If the business is to make any changes that would, in some way, increase scope, then the change control process should be followed on each occasion involving the proper business leadership. IT organizations are particularly well suited to make this type of decision within Agile since the business has been a constant presence throughout development. The decision needs to be based on the business’s actual experience with the system being developed as they have the opportunity to review the system as it evolves.

While ongoing interaction between the business and IT during development is a key advantage, it can become an impediment to maintaining project momentum if the project keeps requiring change controls. Usually, skilled BA’s are able to see where changes need to be made almost immediately, since they have been seeing the system progress on a regular basis. If the project has not implemented proper change controls, then the process of change can cause impacts to the project. It is critical to have a change control process that has appropriate checks and balances to document necessary requirements versus “nice-to-have” requirements. Changes that are essential to delivering a functional system can be incorporated into planning while “nice-to-have” requirements can be brought in if capacity is available or can be planned for future releases.

Beyond managing the change control process, the risk for failed adoption is high when not establishing proper communication protocols for gathering requirements using Agile. “The company would need buy-in from key opinion leaders who could communicate the need for taking a new tack to other employees.”

Communication from
key stakeholders is critical to a successful adoption of Agile methodology because it gives clear direction to the team members during change. Also important, though, is communication to key stakeholders from the project team; ensuring leadership with the rate of adoption and any concerns that could impact the development process. Agile presents concerns for any organization, but with proper oversight in place from a stakeholder and project management perspective, the risks can be reduced and mitigated for a successful implementation.

Implementing Agile does not come without risks given its iterative requirement process. While flexibility is a powerful advantage of the Agile environment, delays can pile up from unstructured project management, which can manifest itself into time lost in development, testing, and product owner approval. It is vitally important that the project leadership manages change by understanding both the cost to the project and the impact of a proposed change. If an approved change puts previously approved requirements in conflict or simply makes them obsolete, then it is essential to mitigate and communicate that requirements change in order to move the project forward. Changing requirements remain a critical aspect to the flexibility behind Agile development. It is one of the methodology’s great advantages. Change, and the time and cost associated with change, must be considered and monitored respectively.

When change occurs, communication within the team and within the stakeholders becomes nearly as important. Project management must play an active role in informing the various moving parts of an Agile project. When assessing the pros and cons of using Agile, it is important to understand that the issues that arise can be actively managed and even taken as an opportunity to make changes to the process in order to minimize risks. The customization of Agile is dependent on identifying how the concerns impact the development team, and addressing the concerns to minimize or eliminate the impact.
5. **Introducing Agile to your organization and building a roadmap for success**

Implementing Agile within an organization always starts with a proper planning effort. As mentioned previously, the Agile team will continue running into foreseeable roadblocks if a planning effort is not established at the onset of the project. The planning should start prior to appointing a development team or building a backlog of work. When implementing Agile across an organization, it will be critical to establish organizational direction and definitions. This will provide the framework necessary for successful adoption of the methodology and to create a baseline from which the teams will work. Leadership should take a deliberate approach in building an Agile foundation, which can be scaled throughout the organization. This foundation should include architectural standards, change controls, and project management guidelines. Without this foundation, the Agile team may make these processes up as they go along, which could delay development and impact whether Agile can properly scale throughout organization.

Successfully defending the project from scope creep requires clearly defining scope and goals from the outset. The risks can become even more profound in an Agile project. The iterative nature allows for multiple “oh, by the way” opportunities. Providing a clear change control process with decision-making responsibilities outlined and accountability transparent within the project allows for the greatest chance for success. It is the management of change, both deemed in-scope and out-of-scope, which will determine the success of the project.

To help manage the scope, the team must accurately estimate their capacity. To do so, the project requires clear definition of roles and responsibilities. The nature of Agile, however, suggests a shift away from the typical focus of the hierarchical nature of a team. While leadership roles might not always be defined, the success of the team lies in the regular rhythm of accountability that is driven among and across the teams. The most successful teams flourish because they hold themselves accountable for accomplishing work, and the leadership role is viewed as another role within the team.

To hold a team to the defined backlog or status reporting, the organization needs to invest in sufficient project management tools. The project management tools of choice may depend on the organization’s corporate culture. The choice of tool for the organization may not determine the success or failure of the project, but as a matter of planning it may be helpful to think of what features will best fit the organization. Fortunately, an abundance of materials exist that can help an organization choose the proper tools to fit its desired needs.

As the size of the development team and the project increases, so does the need for more robust workflow tools. Whiteboards and spreadsheets no longer serve the need for transparency and accurate traceability. According to *The Economic Times*:

> “As companies shift from small projects and teams engaged in Agile to more complex projects and potentially distributed teams, there is a need to shift from paper and spreadsheets to tools that provide workflow, persistence and traceability. At the same time, caution must be taken to avoid stifling teams’ ability to establish a productive flow.”
Once the projects begin meeting the predetermined success criteria across various Agile projects, the organization should migrate to a tool that can consolidate work effort across all projects and provide a structured status report for project management and key stakeholders. Status reports and measures of completion for Agile should look different from those used for Waterfall, considering that each method of development reports out on different success criteria. Outlining the success criteria while using a consolidated project management tool, when possible, should provide stakeholders and leadership with the right amount of visibility into Agile adoption.

Investing in a tool, and the decision regarding which tool to utilize, will be dependent on what shape the Agile methodology takes. Previously, Waterfall style teams would feel the impact of implementing the Agile methodology when they had to start collaborating on a daily basis. The change in direction is gradual, but as teams mature it is important to provide more robust tools to assist with the flow of information between the business and IT. The correct tool is one that can provide a workflow that assists the team members working in an iterative process because it outputs live updates about progress made on work. For example, Agile uses tasks, which generally do not get broken down into smaller pieces of work considering the risks and of transferring ownership. With a tool in place to facilitate workflow, the risk of process delays related to task ownership can be mitigated in conjunction with standard daily team meetings (Stand-ups). Plus, a tool can provide the documentation that critics often say is insufficient in Agile. With the correct tools in place, associated documents will be tracked automatically within the tool, providing sufficient traceability and documentation for project management and key stakeholders.

Early on in the Agile adoption period the project team will enter various stages of team development. It is important that the Agile team finds some of its own way, given the constraints put on it through effective project management. The team needs to identify components of the Agile methodology that best fit their team personalities and the corporate culture within their respective organizations.

The implementation and adoption of the Agile process should be fluid and iterative, just like the development of requirements. This allows the team to learn their strengths and their weaknesses and adapt accordingly to best utilize their capacity. After completing a sprint, all members of the team should hold a retrospective to discuss the successes, opportunities, and failures from a recently completed sprint. Retrospectives help drive an evolving adoption of the Agile process. Taking the time to properly diagnose the underlying reasons for successes and failures will help the team succeed in future sprints. By leveraging what worked and minimizing or eliminating what didn’t work, then the team will improve their overall velocity. This will guide the team toward accountability and achieving the goals or completing the tasks— not just implementing the process. If a piece of the process cannot be clearly tied to achieving the goal, strong consideration should be given to eliminating it.

Clear communication and education for the project team will be essential in order to drive a clear understanding that the process will be flexible with collaboration at all levels of the project team. One of the biggest impacts will be to the requirements-gathering process because of the increased transparency and collaboration between business and IT. Everyone on the team will need to subscribe to the idea that is not acceptable to complete a requirements document and “throw it over the wall.” It is an iterative process; a collaboration between the Business Analysts and the product owners, the developers, and the testers to accurately and efficiently document the business needs.

When building an Agile team, be prepared to separate the academic Agile methodology from the Agile that will be implemented within the organization. For instance, the Agile documentation process will likely not be as intense and detailed as with traditional methodologies, but will still require some documentation of requirements, tests scripts, and project management. Less documentation is required in large part due to the shorter cycle between requirements and documentation. The focus is not on documentation requirements perfectly, but rather producing production worthy features. If something is found to be missing in the requirements, then the gap is much easier to resolve one week later than it is one year later.

This leads to another misconception about the Agile process: change is encouraged. Agile is not about encouraging more change by the business, but rather preparing for change and getting to the inevitable change sooner rather
than later in the process. In fact, change should not be solicited. If a particular set of requirements is agreed upon and signed off, then the team should move on to the next task. Allowing for requirements to change during the gathering process and suggesting a change is a different matter altogether.

Changes to already approved requirements are inevitable given the nature of Agile. As long as the team has implemented effective change-management protocols and is transparent with project leadership, most critical issues can be addressed without derailing the project.

Instability is expected when teams are adapting to change. When the entire team is impacted by a change in requirements, it can cause tension. The requirements strategy should be outlined at the onset of the project. Discussions should take place to develop a process to move the requirements to development, then from development to test. It should include the entire project team, not just the Business Analysts. It will not only promote communication within the team, but also provide ownership and accountability for the process across the team.

It takes time for teams to evolve into a high functioning Agile unit. Blending personal style and Agile techniques takes time, but the transition can be made easier by selecting the correct resources and providing a strong project management structure. Over time, several fully functioning Agile units will handle project work. By providing the proper structure, the organization will have a scalable process that can be used throughout future projects.

A few key things should be considered before committing to Agile Methodology. Firstly, the organization must decide whether an organization or team is ready for change. Preparing for a shift to an Agile environment should involve understanding how the organization or business unit operates as a team, rather than individual achievement. While it is important to have solid performers on a pilot team, it is more important to understand and evaluate the risks posed by individualism prior to implementation. If the business unit is focused on individual goals, then they will need to be slowly transitioned to sharing in the success of their team. The shared success of the team will be critical during the requirements process to avoid hand-off of requirements and the misconception that there is an elimination of the responsibility to see a task through. The culture will be critical in developing a change-management strategy. In order to increase the chance for successful adoption, the company must understand how its guiding principles—how it manages projects and people—can best be adapted within the change-management system.

For example, if the organization does not rely on change requests for minor alterations to project requirements, then they should not have an elaborate change request process for the Agile projects. It is necessary to work within the system that is currently in place, while carefully thinking out modifications from the previous Waterfall methods, to provide the best possible opportunity for Agile adoption and organizational success. The same goes for picking the correct project to pilot and the team resources to populate the roles. Agile may not work in every organizational culture, but it is important for companies to acknowledge how their teams are currently functioning and adopt a methodology that will best complement their culture and skill sets.

When preparing the organization for Agile adoption, it is essential to identify the specific project and team members within the organization to gain experience within an Agile environment before rolling out to the rest of the organization. An ideal team is one that best suits the Agile methodology of being accepting of change, goal-oriented and team-oriented.

The pilot project team will play an important role in identifying what pieces of Agile will work best for them. The team members could also act like champions for change. It will be critical for managers and stakeholders to understand the pilot process itself, as well as what will make the project a success. They should be open to spending months modifying and maturing the process before it is fully adopted and functional. By selecting the right resources across the organization, the pilot project will avoid many project pitfalls, including missed
communication, lack of stakeholder involvement, and lack of motivation. As an added benefit, the adoption process could continue as the pilot team markets the methodology and its successes across the floor.

Once the project passes the success criteria set forth in planning, the organization should be fully prepared to roll out the customized methodology enterprise-wide. A plan should be in place to scale the process, if possible, when the given the go ahead from IT leadership. Also, the key stakeholders and project team should conduct a retrospective encompassing the entire project, from pre-planning to release. The focus should be on more than what went well and what needs improvement, but what parts of the process can be re-used and reformatted to different parts of the organization. The outputs of a retrospective should be identified as soft benefits. The outlined risks for each new project adopting Agile should be mitigated, if possible, through the next project’s planning sessions.

Once Agile is an established methodology within the organization, there could be tension between the traditional project management group and the Agile project management group. Both will report on different things in their status reports. The traditional project management group may feel threatened by the organizational change of face. Additionally, external vendors and suppliers, which are working in a certain way, may be confused by the change in pace.

Transparency in actions and active communication are the best ways to manage these risks, both internally and externally. There is a place for Waterfall. Not every project is a candidate for Agile. The goal, as always, is to put the organization and its people in the best position to succeed.

In order for an Agile implementation to be strong it will require collaboration, strong individual contribution and independence, but all with the best interest of the team in mind. Once a pilot project is in progress there will be a need to assess the current team’s capabilities and comfort with the adopted process. The best way to gauge the comfort level with the new requirements process is to simply ask. The initial Agile project is not an end in itself, but rather a beginning to implement the process in other parts of the organization. Communicating with the Agile team and allowing their input into the process can improve the transition. The Agile team should own the process and, ultimately, the success of a project.
6. Conclusion

Ultimately, an organization will need to be cognizant of its place among its competition. It is the organization’s responsibility to determine whether need for change is necessary. While a traditional methodology may remain effective in some organizations and industries, the competitive landscape for delivering improved consumer experiences continues to grow and intensify. The risk is that the laggards will be left behind without the capability to deliver high quality products at an ever increasing pace. According to Tom Mellor of State Farm:

“If you’re not focused on continuous improvement and adapting your development organization to a cultural model of efficiency and production, it does nothing but benefit us as your competitor.” vi

The consumer landscape continues to change and grow in importance. In order to keep pace, an organization will need to focus on a cycle of retrospection and application of lessons learned.

Executive leadership is always looking for performance improvement opportunities; even the most profitable and well run departments should strive to improve year after year. When reviewing ways to best optimize a budget, whether is yearly or a multi-year plan, the goal is to balance speed to market and accuracy of scope completion. An organization should not forget the high value gained from speed to market, as long as it produces a quality product. If a competitor delivered 80% of a similar product and released in six months while another delivered 100% at twelve months, who won in the market? This question is rarely easy to answer on a forward-looking basis, but tends to be quite clear retrospectively. Carefully balancing these factors could be the difference between success and failure.

One of the primary goals of Agile when gathering requirements is to provide organizations with flexibility and a more efficient use of their budgets. The use of Agile, especially during the requirements process, will allow projects to develop more efficiently by having work tracks running in parallel rather than waiting for fully developed requirements with little interaction with the business. This change is not easily adaptable for larger organizations and difficult to pursue organically. The shift in culture is something that will require internal knowledge, a longer runway of time to fully engage and transition, and the ability to commit significant FTE resources. Resource constraints can cause major impacts not only because of capacity, but the ability to learn a new process while continuing to deliver their work at a higher velocity than previously done before.
For more information about Agile methodology, please contact:

Russ Riggen  
Principal, Advisory Services  
+1 630 561 3626  
russell.riggen@us.pwc.com

Ricky Raisinghani  
Director, Advisory Services  
+1 312 298 2695  
ricky.raisinghani@us.pwc.com

Kyle Ryan  
Senior Associate, Advisory Services  
+1 267 330 1642  
kyle.j.ryan@us.pwc.com
Works cited


ii Voelker, Michael P. “Two Steps Forward.” PropertyCasualty360, April 14, 2011.


iv Voelker.

v The Economic Times, August 6, 2012.

vi Voelker.