Usefulness of Agile Technique on Support Teams

Charles Edeki, Ph.D[1], Amanda Prince[2]

[1]Bronx Community College, City University of New York
Department of Business and Information System
2155 University Avenue, Bronx, New York 10453.
[2]School of Science and Technology, Department of Information Technology
American Public University
111 W. Congress Street, Charles Town, WV 25414

Abstract
Can Agile be used on support teams? The Agile software development methodology has become increasingly popular for building applications by software development project teams. Many companies are moving from the traditional waterfall approach to the faster agile approach for their software development projects. Support teams have a lot of on their plate dealing with production issues as well as bug fixes and enhancements. This study examined the possibility of using Agile on support teams to help manage these development tasks while still ensuring that production issues are handled as they come up.

Keywords**:** Agile, waterfall model, software development, software bug.

Introduction

Production support teams have several things they are responsible for. First and foremost, they must keep production systems and/or applications up and running 24/7. In addition to that, they work on bug fixes and sometimes small enhancements.

This study considered what the agile development methodology is about and the challenges that support teams facing to accomplish software development tasks without a development methodology in place. Several approaches to incorporating agile development tasks in the software projects were described. The goal of these approaches was to allow support team members time to focus on development tasks to be able to get them completed and not be constantly distracted by production issues. Agile development methods are used by many companies and many development projects are seeing lots of success when using it. Support teams need something to help with the challenges they face regarding development tasks. The study looked into these challenges to research the question of can agile be used on support teams.

The Problem

Support teams are getting more and more projects transitioned over to them to support. The rate to which they are getting new applications to support is possibly due to many projects moving to agile which allows projects to be completed more quickly. Support teams may benefit from adopting similar agile practices to aid in the growing number of bug fixes and enhancements due to the increase of applications. Development teams have been using agile for some time, but can agile also be used on support teams?

Production support teams are generally responsible for making bug fixes, small enhancements requested by customers, and dealing with production issues. There is a lot of responsibility to manage a 24/7 production environment. Speediness, precision, reaction time and problem-solving are crucial and usually the support team is working on code they didn’t create. That can cause many challenges to a team trying to accomplish other tasks at the same time such as enhancements. McLaughlin (2008) wrote that, the struggle typically comes in the form of pulling people away from development tasks to make whatever hot fixes are necessary. This is clearly disruptive to the development efforts. Oftentimes, a team member is being pulled away to work on something that’s very unpredictable. This disrupts their rhythm and velocity. It makes planning difficult. Context switching makes developers less productive.

With all the distractions of production issues, it is difficult to make much headway on development efforts. As production issues arise, the development efforts are pushed to the side over and over, and typically, not a lot of progress is made. This leads to customer frustration with not getting their requests completed, as well as, team member frustration where they feel like they do not get things accomplished.

Agile Software Development Methodology

Incremental software development methods have been tracked back to 1957 at IBM (Nicholson, 2013). Since the mid 1990’s, agile software development has become increasingly popular. Dennis & Wixom & Tegarden (2012) describe agile as:

"Focusing on streamlining the system-development process by eliminating much of the modeling and documentation overhead and the time spent on those tasks. Instead, projects emphasize simple, iterative application development. All agile development methodologies follow a simple cycle through the traditional phases of the system development process (p. 14)."

One of the biggest benefits to agile development is that it provides opportunities to assess the direction throughout the development lifecycle. This is done through iterations where the teams must present a potentially shippable project at the end of each increment. Agile Methodology (2015) states:

"by focusing on the repetition of abbreviated work cycles as well as the functional product they yield, agile methodology is described as "iterative" and "incremental". In waterfall, development teams only have one chance to get each aspect of a project right. In an agile paradigm, every aspect of development — requirements, design, etc. — is continually revisited. When a team stops and re-evaluates the direction of a project every two weeks, there's time to steer it in another direction. This "inspect-and-adapt" approach to development greatly reduces development costs and time to market. Because teams can develop software at the same time they're gathering requirements, "analysis paralysis" is less likely to impede a team from making progress. And because a team's work cycle is limited to two weeks, stakeholders have recurring opportunities to calibrate releases for success in the real world. Agile development helps companies build the right product. Instead of committing to market a piece of software that hasn't been written yet, agile empowers teams to continuously re-plan their release to optimize its value throughout development, allowing them to be as competitive as possible in the marketplace. Agile development preserves a product's critical market relevance and ensures a team's work doesn't wind up on a shelf, never released."

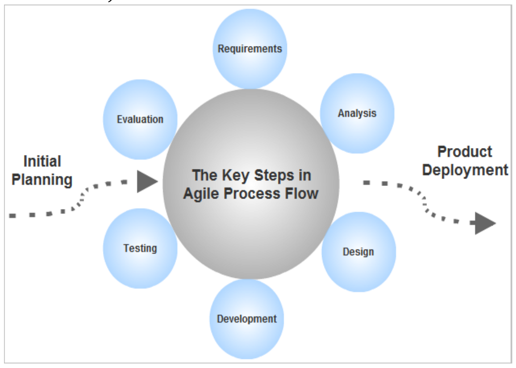

Figure 1. : Agile Methodology– Agile Process Flow Steps

Agile Benefits and Challenges for Support Teams

While these are some overall benefits to using agile, there are some benefits that could directly help support teams. Most support teams have a backlog of bugs and enhancements. Part of incorporating agile would entail reviewing the entire list and assigning points based on their complexity. Unlike taking bugs as they come or as the team fixes them, this approach enables the team to review the bugs in a structured way (Darbha, 2012).

Another specific benefit to support teams is that a sprint is time-bound and shouldn't last longer than three or four weeks, but can be as short as one day. Agile requires that bugs be fixed and verified within the sprint so they don't spill over. This may help bring a sense of urgency and goal-setting that otherwise can be difficult in the support scenario (Darbha, 2012).

It's even more important in support than in development to remember that the goal is to deliver working code at the end of each time box. Bird (2011) states that if some code is not working, or you're not sure if it is working, then extend the deadline, back some of the changes out, or pull the plug on this release and start over. Don't risk a production failure in order to hit an arbitrary deadline. If the team is having problems fitting work into time boxes, then stop and figure out what you're doing wrong – the team is trying to do too much too fast, or the code is too unstable, or people don't understand the code enough – and fix it and move on.

Visibility is another benefit that agile can bring to support teams. Sometimes support teams do not get a lot of visibility unless issues are escalated. Because senior management is involved in the sprint planning and meetings, management is able to focus on the team's successes.

It is harder to introduce agile practices into a support team because there are a lot of technical requirements and some cultural changes that need to be made. But most maintenance teams have little to lose and lots to gain from borrowing from what agile development teams are doing. Agile methods are designed to help small teams deal with a lot of change and uncertainty, and to deliver software quickly – all things that are at least as important in maintenance as they are in development (Bird, 2011).

On support teams, it's not easy to see how big changes can be broken down into small steps that can be fit into short time boxes. Bird (2011) says, it is harder for support because you have to be more careful in understanding and untangling dependencies before you make changes, and you have to be more careful not to break things. The code and design will sometimes fight the kinds of changes that you need to make, because you need to do something that was never anticipated in the original design, or whatever design there was has been lost over time and any kind of change is hard to make.

Sometimes work doesn't fit into short time boxes. Short iterations might work for bug fixes and small enhancements, but sometimes you need to make bigger changes that have lots of dependencies. While Agile teams building new systems can stub out incomplete work and keep going in steps, maintenance teams have to get everything working all at once – it's all or nothing. You can use tools to figure out how much of a dependency mess you have in the code and what kind of changes you need to make to get out of this mess. If you are going to spend "weeks, months, or even years" to make changes to a system, then it makes sense to take time upfront to understand and break down build dependencies and isolate run-time dependencies, and put in test scaffolding and tests to protect the team from making mistakes as they go along. All of this can be done in time boxed steps. Changing data in a production system, especially data shared with other systems, isn't easy either. You need to plan out API changes and data structure changes as carefully as possible, but you can still make data and database changes in small, structured steps. Agile development teams follow incremental design and development to help them discover an optimal solution through trial-and-error. Support teams work this way for a different reason – to manage technical risks by breaking big changes down and making small bets instead of big ones. Working this way means that you have to put in scaffolding (and remember to take it out afterwards) and plan out intermediate steps and review and test everything as you make each change. Sometimes it might feel like you are running in place, that it is taking longer and costing more. But getting there in small steps is much safer, and gives you a lot more control.

Teams working on large legacy code bases and old technology platforms will have a harder time taking on these ideas and succeeding with them.

Agile development is not without some challenges. One major criticism is that if it is not carefully managed, the project may never actually complete. It could just become a never ending cycle of iterations that keeps changing the application. Another criticism is the lack of actual documentation up front could lead to a lack of documentation when the project is completed. These challenges can be overcome with proper guidance and leadership.

Possible Approaches to Incorporating Agile

Agile development is best suited for small projects. Typically, a support team only works on bug fixes and small enhancements. Because they don't usually work on large projects, agile development is definitely something to consider. The current, most common approach, to development by support teams is as time is allowed, or when there are not production issues. But as discussed earlier, this can lead to things not getting accomplished very quickly. In an effort to try to get more tasks completed, using agile development methods by support teams could be considered by using several different approaches. Cottmeyer (2009) describes three approaches:

Approach One: Alternate the teams between support iterations and new development iterations. The team would establish a steady velocity (every other week) based on their new development work and that steady velocity could be measured against the remaining backlog to balance the scope and end date. If the team is not 100% consumed with support during a given sprint, they can use the extra time to get ahead of the game on their upcoming development sprints.

Approach Two: Assuming there is some historical data on how much time is spent doing support, allocate a fixed amount of bandwidth to support activities each sprint. For example, each team would allocate 30% of their time to support activities and velocity would emerge based on the time they have remaining to do new development.

Approach Three: Have one team responsible for development and one team that is responsible for support. That would allow the development team to get into a groove writing new code and the support team to establish patterns for how much and how quickly they can get through support tickets. Team members could rotate in and out of the support team and back onto the new development team to allow everyone time doing both aspects.

Davies (2001) has another approach:

One person is allocated permanently to a Client Support role and each day a developer pairs with that person to provide technical assistance. This developer is "exposed": this means interruptible for customer and client support questions. The exposed developer may need to ask another developer for help but they will have a better understanding of who is the best person to ask.

There are several approaches that could be tried to incorporate agile into a support team. As with any development team working on any agile project, the methodology can be adjusted to fit the situation. Not all approaches would work for all teams and all type of projects.

Conclusion

In conclusion, agile development methods can be used on support teams in several ways. Using one of the described approaches, support teams could find time to work on small pieces of the enhancement or bug fix within each iteration. Some of the approaches even dedicate a team member's time completely to development tasks instead of being pulled between production issues and development. While it may not be easy to move a support team to agile, there are a lot of benefits. The biggest challenge for support teams is finding that dedicated time to work on development. If they are able to find an approach to get the time needed, they could make progress within each iteration and get the development tasks completed. This benefits the customer because they get their enhancements and bug fixes completed and it also helps the team members feel like they are getting things accomplished.

References

1) Agile Methodology. Retrieved January 4, 2015 from http://agilemethodology.org/.
2) Bird, Jim. (2011, October 12). You Can't be Agile in Maintenance. *Building Real Software*, Retrieved from http://swreflections.blogspot.com/2011/10/you-cant-be-agile-in-maintenance.html.
3) Cottmeyer, Mike. (2009, February). Handling Support On Agile Teams. *Leading Agile*, Retrieved from http://www.leadingagile.com/2009/02/handling-support-on-agile-teams/.
4) Darbha, Shweta. (2012, February 22). Can Support and Maintenance Teams Become Agile? *Scrum Alliance*, Retrieved from https://www.scrumalliance.org/community/articles/2012/february/can-support-and-maintenance-become-agile.
5) Davies, Rachel. (2001, May 3). Extreme Support, Retrieved from http://cf.agilealliance.org/articles/system/article/file/1253/file.pdf.
6) Dennis, A., & Wixom, B.H., & Tegarden, D. (2012). *Systems Analysis & Design*. John Wiley & Sons, Inc.
7) Linders, Ben. (2010, Winter). Process improvement, The Agile Way! *Methods and Tools,* Retrieved from http://www.methodsandtools.com/archive/archive.php?id=115.
8) McLaughlin, Mike. (2008, September 23). The Agile Coach on Production Support. *Agile Exectuive Blog*, Retrieved from http://blogs.versionone.com/agile_management/2014/04/08/the-agile-coach-on-production-support/.
9) Nicholson, Laurence. (2013, April). Agile in Project Management: A Brief Overview. *PM World Journal, 2:4*.

a. http://pmworldjournal.net/wp-content/uploads/2013/04/pmwj9-apr2013-nicholson-agile-project-management-brief-history-commentary.pdf