2014

Amy Comrey

INFSYS 6840

11/8/2014

Agile Requirements and Team Collaboration



**Agile Requirements and Team Collaboration**

By: Amy Comrey

**Abstract**

The term agility means the capability to respond to changes and supply the business with effective results. One of the most important steps in providing a result is developing useful requirements. Requirements change regularly due to fluctuating business conditions. New ideas come to fruition, new systems come into play, and users may change their mind. Since these types of changes are normal in software development, agile methodology is able to include these modifications in the final output of the project. According to recent research, surveys show that 42% of agile projects are successful (Sutherland). Gathering requirements for agile methods contrasts with other traditional requirements gathering in that it requires a greater deal of team collaboration and confirmation of requirements. Agile development is more than responding to changes in requirements-it is also about changing the entire process of development. This paper will examine the agile requirements gathering process as well as team collaboration throughout the analysis phase. Focus will be placed on user stories and back-log prioritization and customer involvement in the agile requirements gathering process.

**Introduction**

Analyzing a system is the process we use to begin realizing the needs of the customer. Successful system development depends on the ability to extricate the requirements a system must fulfill. In more traditional forms of requirements gathering, although customers (final users) are included in the process of developing a system, the system still tends to end up as the vision of the designers, with little attention to customer requests. This has led to projects being labeled as “failed” since these systems were not built to meet customer requirements. With the increased use of agile requirements gathering methods, customers have been able to be more involved with their systems throughout the project lifecycle which has led to greater success. To establish a sound set of requirements, many things have to be taken into account such as: the problem or the user needs, budget, resources and even political aspects (Konaté). This takes a great amount of proficiency and skillfulness within a team; therefore, team collaboration is crucial in ensuring that all of the appropriate individuals and groups are included in the requirements gathering process as well as ensuring all issues are investigated and resolved.

Analysis of a system can be extremely time consuming and costly. Customer involvement is crucial in the requirements process, as this can greatly reduce the amount of time and money spent on a project. “The savings come through a better product for the user with fewer defects. If the IT development group can deliver a quality product, even in its first phase, to an organization sooner, IT is then delivering higher value to their organization.” (Yasin) Many information technology (IT) professionals admit that issues with the final system or solution lead straight back to the analysis and design phases of the life cycle. PM Solutions released a report naming requirements definition as the number one cause of project failure. (Strategies for Project Recovery) This includes unclear requirements, lack of agreement, lack of priority, contradicting requirements, and ambiguous or imprecise requirements. In agile requirements gathering, the customer becomes part of the team’s regular meetings which allows the project to be altered on a smaller scale over time.

User stories also assist in the requirements gathering process as they define requirements at a high-level and contain enough information for developers to produce an estimate on the amount of effort that will be needed to implement the feature. User stories also assist in back-log prioritization as each requirement is analyzed and placed in order of priority to the customer. Through examining the agile requirements gathering process, we will see the impact team collaboration has on the systems analysis process.

**Requirements Gathering**

Requirements gathering is the first step towards analyzing a system. Since no system is perfect, the team has to identify what is exactly necessary. This requires input from all portions of a team, but the main focus should be on the customer. There are many methods teams may use to gather requirements, including: meetings or interviews with the customer and other stakeholders, user observation, and prototyping. Any system we examine-manual or automated-has inputs and outputs which the team needs to identify since the new system that will be created will have comparable ins and outs. Once these are identified, team members can examine what could cause the system to run more efficiently and effectively.

**Traditional Requirements**

With traditional requirement gathering, teams gather requirements at the beginning of the project and typically create a document that lists out everything in detail. The customer then reviews and approves this documentation before the project begins. Once the project begins, change requests have to be submitted if new requirements are needed. These change requests assess what the impact of the change will be, and the team decides if the modifications are necessary. This can easily set a project back if a team does not agree with the changes. Even if the changes are needed, the setback could be enormous.

**Agile Requirements**

As mentioned earlier, change is expected in a business setting, and agile methodology allows these changes to be planned for and estimated (Appendix 1). With this methodology, a high-level outline of project requirements is worked. Detailed requirements are then assembled in the development phase. Agile uses an iterative form of implementation with every iteration taking place usually about every one to four weeks. With each iteration, the customer and team decide which requirements will be implemented for the particular iteration. When the iteration is complete, this process is repeated until the project is considered complete. Team collaboration is essential in this type of requirements gathering and could cause many issues if not kept in check. Issues like inadequate social contact, geographically distributed practitioners, cultural and time differences can cause problems with communication and project management (Sinha). This can eventually cause further frustrations. As Khuzadi mentions in his article on collaboration management: Sometimes companies struggle, and they are sometimes very inefficient in generating customer interest for the future as they complete projects over schedule and do not invent anything new. These companies by default still remain as chief establishments though since they finished the project. This counterbalances by sometimes offsetting the risk of finding and working with another team (Khuzadi).

**Requirements Modeling**

Without clear goals, teams should not proceed with a project. Requirements definition conversations often center on steps to do, techniques to use, or methods to follow (Holtzblatt). There are several resources and tools a team can use to lay-out and structure the requirements process. Entity-Relationship Diagrams can be used in database design modeling and data flow diagrams are helpful in showing a process from start to end. The ultimate goal of these tools is to collect information on how to improve a new system for the customer. According to the Project Management Institute (PMI), 37% of project failures are attributed to inaccurate requirements gathering (Requirements Management). To understand a customer’s desires, many methods can be used to collect requirements. As mentioned before, setting a meeting with the customer is one of the best and most used ways to gather information about an information system. This helps to see how the current system works and interacts with other components or other systems. During this meeting, the team could also watch the user (customer) walk through the process of what they currently do. This helps fill in any gaps that could be missed by human error. Meeting with the customer allows the team to understand current problems and possible opportunities as well as discuss whether certain changes are viable options. Sometimes the customer will provide process documentation, but this is normally a formal procedure. Individuals tend to have different routines on how to access data, and sometimes outside factors affect or influence how they go through a process. For this reason, meeting with the customer and taking part in the process is critical to project success.

**Requirements Organization and Prioritization**

Many developers focus just on how to elicit and gather requirements, but do not actually focus on organizing requirements and prioritizing them. There has been an upward trend in using Kanban, an efficient method of handling the software development process. Kanban is an advanced variation of the reorder point (ROP) method (Im). Kanban is similar to Scrum task boards, except Kanban looks at a more granular level. As mentioned before, the software development process is a pipeline with inputs and outputs. Hopefully those outputs are generally improving business. Within that pipeline, we are analyzing requirements developing, and testing. In agile, we are repeating this consistently. While Kanban is a relatively simple process, it has helped many businesses identify bottlenecks in processes that hampers innovation. For example: when testers can only test five features a week, but developers can produce ten features a week, only five features will output per week because the testers are acting as a bottleneck (What is Kanban). When this occurs, it creates detachment from the market and market value drops. Quality will also undergo problems in this case as well because the testers do not want to be the bottleneck, so they may start processing these as fast as possible, creating errors. When these errors are implemented in production, bugs cause issues with users which wastes time and money since you paid the developer to write the bug, then you pay the developer to fix the bug, then you have to go through full testing to ensure the bug is fixed in all environments.

The further along a problem continues in the pipeline, the more exponential the cost to the company. Kanban can improve the process and can be incredibly simple. We have this on our wall at home: it is a board on the wall with sticky notes under columns labeled “To Complete”, “In Progress”, and “Completed”. To limit work projects, numbers can be placed at the top of each column which limits how many projects can be placed under that column at any one time. This allows the team to see each step of a process and reveals the bottlenecks. Also, by limiting the projects, this also allows teams to address each problem and reorganize resources before it is out of control and allows for a continual system review (Esparrago).

Kanbans generally evolve over time, but one of the most important steps in creating this board is agreeing on the organization and prioritization of events. Teams must agree on: “Initial WIP limits and policies for changing or temporarily breaking them; a process for prioritizing and selecting features; policies for different classes of service (e.g. "standard", "expedite", "fixed delivery date"); and frequency of reviews.” (What is Kanban) For example, teams need to know what estimates are necessary and when choosing requirements to work, which requirements will be of highest priority.

**Scrum Teams**

Scrum methodology is synonymous with agile methodology. In this methodology, projects are divided down to sprints that typically last between one and four weeks. At the end of every sprint, customers and stakeholders meet on the project and plan for the next sprint. Projects can then be adjusted as needed to meet the project direction (Lee). Many teams run into issues when manager’s plan a team member’s workload capacity without speaking to them first about estimates on time. Instead of confirming with team members first, the manager expects certain tasks to take a certain amount of time, and they allocate work based on their assumptions. This causes problems as some tasks are much more challenging and the manager is only judging the task on length, but not on level of ease. Scrum teams eliminate this approach, as work is assigned to the whole team, and team members estimate their own “time” instead of a manager doing so. Typically teams do not actually measure in time, but rather in effort for each story (Scrum Effort).

There are three primary roles for Scrum Teams: Product Owner, Development Team Member, and Scrum Master (Shankarmani). The Product Owner takes down estimates during sprint planning meetings and prioritizes items in the backlog, which is a thorough analysis document that contains every requirement for the new system (Scrum Backlog). The Product Owner can then analyze and predict future releases based on team member estimates.

**Team Collaboration**

With Scrum teams, there is a term called “planning poker”. Basically, this means that although the team is estimating their project times, all members should reveal their effort/time estimates together to reduce influences from outside sources or other team members. This is called “showing their hands” which is like poker (Scrum Effort). Team members will obviously take into account different factors and will estimate different, so these estimates will go through several revision before a final agreement is reached. There is also a well-known fable named the chicken and the pig which describes commitment to a project. In the fable, a dish is needing to be made of ham and eggs. The pig provides the ham, which he must sacrifice his life for. The chicken provides the eggs which he can just produce. The pig is fully committed while the chicken is only involved, but both are needed for the dish (Scrum Meeting). In projects teams are the people committed to the project, but the customers and high-level management are only participants. This is important to remember in team collaboration, as the active members should only be those committed to the project. The “chickens” are always welcome to attend team meetings, but should only appear as guests. In meetings, the “pigs” should be required to give updates on progress. Progress can be measured in various ways. Typically in Scrum, task boards are used. Kanban boards are also increasing in popularity to reflect user stories, as they can provide greater detail on what state each task is in, which can help keep the chickens comfortable knowing things are getting done.

**User Stories: Kanban**

In using the agile methodology, it is very important that urgent requirements are planned to implement first. This assists a project in that if time runs out then the higher priority items are already completed. Due to the regular delivery of product through iterations, customers can be more involved in the final product. Kanban boards are a lot like Scrum task boards. They both have similar columns showing advancement of user stories (tickets) from a “backlog”. With Scrum teams, when all tasks are complete, the user story moves into completed status as well. With Kanban, we are tracking the progress of user stories across the board, and the requirements should be understood, so there is no need for separate lanes to keep certain tasks and user stories associated with each other (Scrum to Lean Kanban). Bugs that are found can be fixed sooner since each iteration is one to four weeks and the team no longer has to wait until the end of the project for the final testing cycles to begin.

User stories help prioritize which requirements will be completed first. These are developed typically by having customers/users write high level requirements down on notecards. These will form the higher level requirements needed from which more detailed information will be gathered. User stories can have functions-for example-how a system must work, and constraints-for example-something not fixed to a certain function, such as performance. When requirements seem to be out of scope, team meetings can be held to decide if the requirement will still be included. As progress occurs in a project, more details will be needed than what is on the story cards. These details are generally kept in a separate document along with exceptions.

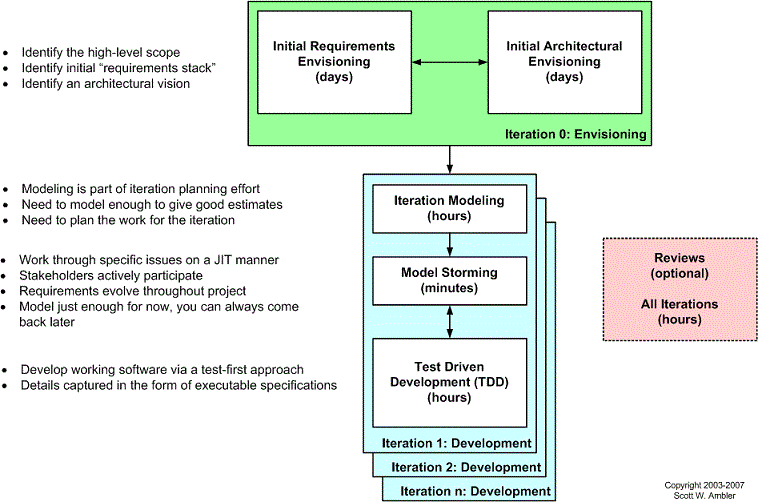
A good way to go about prioritizing user stories is to think about the system with three things in mind. First, is the user story/requirement required for the underlying architecture of the system, in other words, is this piece necessary for the system to even get off the ground? Second, how much business value does the user story provide to the end users? More business value means more importance should be placed on that user story. Finally, is the user story difficult to complete or will it require a specific amount of learning in order to complete? It is important to work through more difficult user stories early in the development of a new application for two reasons. First, if it is a critical piece of the application that cannot be figured out, then the team knows early on they either need better resources, more training, or they need to understand that piece will not be implemented in a timely manner or at all. Second, by implementing the harder pieces of the system earlier, the team will make better design decisions and learn from completing that story, and they will then be able to use that learning to complete future stories faster. Looking at these three things together can help with the overall prioritization of the backlog.

**Conclusion**

Agile requirements gathering is the best way to have customer involvement throughout the software development process and this process allows requirements to change. Requirements gathering is the first step towards analyzing a system, and since no system is perfect, the team has to identify what is exactly necessary. Teams must effectively collaborate in order to achieve success with a project as requirements can change during each iteration. Agile development also requires teams to effectively and efficiently include the customer in all parts of the lifecycle process. This leads to a better product as the customer can make changes and update ideas throughout.

There are many methods teams may use to gather requirements. User stories can help prioritize requirements and Kanban boards can help track user story progress. Any system we examine has inputs and outputs which the team needs to identify since the new system that will be created be comparable to the old system. We examined the advantages and disadvantages of both traditional and agile requirements gathering methods. With agile methods, customers are consistently involved in the development process. It is easiest to meet with customers and ask them questions regarding which requirements they expect and it is also helpful to have a workshop to watch their processes. While agile may not necessarily show immediate benefits and savings of cost and time, typically customers and business benefit from decreases in errors, and a better overall product.

Appendix 1: Agile Methodology Example



**Works Cited**

Refereed Journals:

Esparrago, Romeo A., Jr. "Kanban." *Production and Inventory Management Journal* 29.1 (1988): 6.

Farooq, Umer, et al. "Requirements Gathering for Collaboration." *International Journal of Human –*

*Computer Studies* 67.4 (2009): 297-312.

Holtzblatt, Karen, and Hugh Beyer. "Requirements Gathering: The Human Factor." *Communications.*

May 1995.

Im, Jin H., and Richard J. Schonberger. "The Pull of Kanban." *Production and Inventory Management*

*Journal* 29.4 (1988): 54.

Khuzadi, M., "The collaboration management culture," *Aerospace and Electronic Systems Magazine,*

*IEEE*, vol.25, no.12, pp.4, 15, Dec. 2010.

Konaté, Jacqueline, and Abd El Kader Sahraoui. "Collaborative Requirements Elicitation: A Process-

Centered Approach." *Group decision and negotiation* 23.4 (2014): 847-77.

Lee, Rich C. "The Success Factors of Running Scrum: A Qualitative Perspective." *Journal of Software*

*Engineering and Applications* 5.6 (2012): 367.

Sinha, Vibha, Bikram Sengupta, and Satish Chandra. "Enabling Collaboration in Distributed

Requirements Management." *IEEE Xplore*. Sept. 2006.

Shankarmani, Radha, et al. "Agile Methodology Adoption: Benefits and Constraints." *International*

*Journal of Computer Applications* 58.15 (2012) *ProQuest.* 10 Nov. 2014.

Sutherland, Jeff, Neil Harrison, and Joel Riddle. "Teams that Finish Early Accelerate Faster: A Pattern

Language for High Performing Scrum Teams". IEEE, 2014. 4722-4728.

Web Resources:

Appendix 1: "User Stories: An Agile Introduction." *Agile Modeling*. 2014.

"Requirements Management." *Project Management Institute*. Aug. 2014.

"Scrum Backlog." *Scrum Methodology*. 5 Sept. 2011.

"Scrum Effort Estimation and Story Points." *Scrum Methodology*. 21 Nov. 2009.

"Scrum to Lean Kanban: Some Problems and Pitfalls." *Javalobby*. 10 Oct. 2013.

"Scrum Meeting." 06 Mar. 2014.

"Strategies for Project Recovery." *PM Solutions Research*. 2011.

"User Stories." *Extreme Programming*. 2000.

"What Is Kanban?" *Kanban Blog Atom*. 2014.

Yasin, Rutrell. "Why Agencies Are Learning to Love Agile Development." *Technology, Tools and*

*Tactics for Public Sector IT*. GCN, 01 July 2013.