Preliminary Investigation of Stand-up Meetings in Agile Methods

E.Hasnain, Dr T.Hall

***Abstract*—In recent years, agile methods have become more popular in the software industry. Agile methods are a new approach compared to plan-driven approaches. One of the most important shifts in adopting an agile approach is the central focus given to people in the process. This is exemplified by the independence afforded to developers in the development work they do. This paper investigates the opinions of practitioners about daily stand-up meetings in the agile methods and the role of developer in that. For our investigation we joined a yahoo group called "Extreme Programming". Our investigation suggests that although trust is an important factor in agile methods. But stand-ups are not the place to build trust.**

***Index Terms*—Agile methods, Developer, Stand-up meetings, Trust.**

I. Introduction

This paper investigates the opinions of practitioners about daily stand-up meetings in the agile methods and the role of developer in that.

The main motivation for this investigation is that although there is plenty of discussion about stand-ups reported in different internet discussions and blogs, much of this is based on internal anecdote and personal experience. Very little research on the impact and importance of stand-ups is reported in literature. This means that managers are using stand-ups without any evidence of benefit. Some of the examples taken from fishbowl website are given below:

Keith Pitty [1] said: *"Stand-up meetings have certainly helped team communication and awareness in my experience. As long as they're short, informal and to the point. I'd much rather have them than not. It's a way of gently and succinctly keeping the team's joint focus on the goal."*
Another example comes from Phil Wilson [1]: "*We stopped having them at my place of work due to poor management, which drove all the developers nuts. As a result, the developers now have them so we know what's going on, the managers know we have them and don't come, and project management suffers."*
Both of above examples show that stand-up meetings are making effects in the work practice. Even though people are facing problems with them, but still they are good way of communication.
This paper is divided into five sections. The first section is introduction. The second section presents background with relation to our investigation. The third section describes the approach that we have used in order to conduct our investigation. The fourth section presents our results to the questions we have asked in the discussion. The fifth section is discussing our results while final section is a conclusion.

II. Background

The Stand-up in agile methods is a short meeting that managers and developers are required to attend every day. In a stand-up meeting usually everyone stands up in a circle to avoid long discussions**.** A stand up meeting aims to **communicate problems, and promote team focus** [2]. Furthermore the purpose of the stand-up is to communicate problems, not to solve those [3]. During stand-ups team members learn what other developers are working on and struggling with and how they can help each other to make the whole team succeed [4]. Stand-up meetings are meant to be short and so the detail of problems is not discussed during the meeting [3]. According to Yip [5] the main structure of stand-ups centers on developers answering the following questions:

- What did I do yesterday?
- What obstacles do I have?
- What am I going to do today?
- What else should the team know about?

III. Method

This investigation elicits some preliminary information about stand-ups from practitioners. We joined a Yahoo discussion group called "Extreme Programming". The main reason to join this group is because it is an effective way to get information from XP practitioners from different locations and different cultural backgrounds. The group has both technical and non-technical discussions. The group also has some well known practitioners as members.In our first message we posted the following questions:

1. What role does developer plays in stand-up meetings?
2. How do we know or how can we find out that if developer has said something in daily stand-up meeting has he/she managed to done that work next day?
3. Is there any accountability of developer that what he/she can say?
4. We always say that in stand-up meetings developers should be asked
   following questions:-
   • what did I do yesterday?
   • What am I going to do today?

Later different people replayed to our questions. This is pilot study that will lead us to see the reaction to different people from our investigation. If it is successful we can use questionnaire in the future to ask the feedback from the same group.

Eisha Hasnain is PhD student at Brunel University, Uxbridge, Middlesex UB8 3PH United Kingdom (phone: 00447921439040; e-mail: Eisha.Hasnain@brunel.ac.uk).
Dr Tracy Hall is Reader at Brunel University, Uxbridge, Middlesex UB8 3PH United Kingdom (e-mail: Tracy.Hall@brunel.ac.uk).

IV. Results

Total Number of Replies to our questions	7			
Who Replied	Replier 1	Replier 2	Replier 3	Replier 4
Number of Replies	3	2	1	1

Table 1 Number of replies

In order to obtain more information about stand-up meetings and developer we joined a Yahoo group called "Extreme Programming". We posted our messages on Monday 22nd November 2007. Overall we received 7 replies from 4 different people. Details of all these can be found on Table 1.

The first reply was from Replier 1 who is one of the known personalities of XProgramming.com. In his message he said *"The role of developer in stand-up meetings is to represent him or herself. It is to be a developer. The developer is supposed to tell the truth. Is there some reason why you would not trust them to do so?"* one of the interesting facts in his discussion is the word "Truth". According to Replier 1 Developers are supposed to tell the truth.

Another interesting thing that Replier 2 comes up in his answer to our questions that: "*It is the purpose of the stand-up to facilitate the work of the project. It does not have the purpose of monitoring the developers."*

According to Replier 3 *"In agile software development, everybody is supposed to work together on things, so anybody who is lying about what they have done or intend to do will quickly be discovered. Furthermore, the community formed by a group of people working together for the same objectives (commonly called a team) is great motivation to be cooperative. People who cannot work that way typically quit agile organizations and move to more traditional organizations where they can keep their work more private."*

Replier 4 in his first reply said: *"One of the things that can be surprising about XP and agile development is that team members are expected to behave professionally. You shouldn't have to check to make sure they're doing a good job; each developer should be doing a good job because that's the professional thing to do. This allows people to take risks and make mistakes, which is a more enjoyable way to work and leads to higher quality and productivity."*

In another reply Replier 4 suggested that "*The thing to remember is that the stand up meeting isn't for management, but for the team. Here, we've had days when our manager (who is also a team member) couldn't be there due to illness, vacation, etc. That doesn't stop us from having stand up meetings.The folks at our stand-up meeting are our co-workers, not our cow-workers. Trust is required; otherwise we couldn't work on a team effectively. Also, there is no real reason to lie, nor any real way to get away with it. The other agile practices (team code ownership, pair programming, frequent check-ins of all-test-passing code, etc) also provide feedback on what the members of the team are doing. Someone who consistently lied about what they got done would quickly be discovered and at the very least talked to. They won't keep it up very long, either due to peer pressure or potential lack of employment.*

For me, stand-ups provide a good feedback mechanism to keep from lying to myself. I could (and in other, non-XP jobs have) easily spend an entire 8-hour day reading XP mailing lists and browsing web-sites. Without stand-up meetings, all I had to do was to get some progress between weekly updates with my manager, and I could at least look like I was getting stuff done. On the other hand, I cannot bring myself to say, day after day, "Yesterday, I got nothing done except web surfing, today I'll work on what I said I was going to work on yesterday, and I'm being blocked by my own laziness". So the stand-ups put pressure on me to actually get work done.

If the organization was agile from the beginning, then people who cannot work that way tend to leave before they have very much valuable knowledge. Also, the collaborative nature of agile tends to avoid anybody knowing anything that nobody else knows about.

If the organization is just switching to agile, they are usually making the change because the traditional approach was not achieving the quality or throughput necessary to stay competitive. The people who do not see that the change is worth becoming competitive are generally part of the problem, so the company is better off without them. I have seen many cases of quality and throughput improving noticeably a few months after the most experienced developer leaves. If you follow any team sports, there are lots of examples of teams improving when their statistically best player moves on. Are you assuming that it's possible to prevent mistakes? It's not, of course. The difference is how you treat them. Some organizations see mistakes as infractions to be punished. This doesn't prevent mistakes, but it does cause people to be afraid of making mistakes. They hide them and blame mistakes on others. This dysfunctional behavior makes it hard for people to work together.

The agile approach is to accept that everybody makes mistakes. Rather than blaming people for mistakes, we try to understand the root causes of mistakes and fix those things. It's not personal and everybody works together.

The stand-up meeting is similar. In agile development, the whole team is a single organism that is dedicated to producing great software. The stand-up meeting is a mechanism that everybody on the team (not just developers) uses to communicate about their status. There is no need to lie because there is no punishment for error. If somebody is constantly struggling, the team as a whole will help solve the problem by identifying the root cause that's leading to his or her trouble.

That's the way it's supposed to work, anyway. I don't generally see problems in this area. You would have to have a crippling level of distrust to lie in a stand-up meeting. Such teams generally aren't interested in agile development; agile development is inherently non-controlling and thus feels unsafe to a distrustful organization."

In another reply from Replier 1 explained how trust is created within agile methods. He suggested that *"If the team is doing as we suggest, they meet every week with their customer, agree on what is to be accomplished, and meet next week to report what was actually accomplished. It's that practice that generates the trust -- and the cross-checking*

-- That makes the stand-up practice work. To deliver software every week, there are other practices that are necessary. These are, to a very close approximation, the practices of XP, or practices very like XP's. What you find out about trust from stand-ups is that if they work, they are evidence of trust in action. They are a geiger counter clicking away because you're in the presence of trust. Trust is a result of agile methods, not a prerequisite. Agile methods deliver running tested software every week or two or four. That creates trust."

V. DISCUSSION

A lot of different opinions came after discussion. According to Replier 1, the main motivation for stand-ups is to discuss different aspects of daily work. So Replier 1 said *“Developers answer the questions. If something is in the way, the team, or the ScrumMaster if they have one, removes the obstacle. The developer works on what he said he would work on. Tomorrow he tells us what he worked on. If something changed, so that he worked on something else, he tells us what he worked on.”* So this is also answering to one of our asked question as well. But this also suggests that stand-up meetings are also a place to see the progress of developer on certain project. So he/she is telling if he/she failed to do something or successful in completing whatever he/she said yesterday.

Another important concept that came under discussion was that stand-up meetings are not for monitoring the developer but they are just for daily project progress. Like Replier 2 said *“One of the things that can be surprising ------ higher quality and productivity.”* So this means that we are trusting team members to behave professionally. But again there are so many conflicting views about this as well, like what is professionalism? Is it that you are working on time or is it that you have finished what you said you will do? All these take us to another conflicting idea of trusting. This suggests that we are expecting someone to behave professionally as we trust that person.

Further talking about trust Replier 4 mentioned *“A basic trust must exist between an employer and an employee that the employee will do his job. Especially for programmers, the customer often doesn't have the technical skill to tell if the programmer did anything. Iterative methods, such as XP or scrum build trust between the customers and the programmers, because they require the programmers to deliver working software every iteration. Trust between programmers is necessary for the team to function, but it's not built by the daily stand-up. Pair Programming and Collective Code Ownership help build trust between the programmers, as does the fact that they are working together and will rapidly learn if they can rely on each other. The daily stand-up is a common way to make sure that everyone is on the same page about what's going on, but it's not a trust-building tool.”* Trust is the final output of all these discussions. But how do we know that trust is present or we need to create it. If it is present how can we test its level and if we need to create it how can we create it?

VI. CONCLUSIONS

After all this discussion this can be suggested that although trust is an important factor in agile methods. But stand-ups are not the place to build trust. Just like Replier 4 said that Pair Programming and Collective Code Ownership can help in building trust. We suggest that stand-up meetings can be place to test trust not to create trust. But agile methods and its recognition of people as their primary factors can be trust creative. Further to this discussion we can suggest that as a result of this pilot study we can run a whole questionnaire in order to get some more information about how study can help in creating or destroying trust within organization or team members. Future work will be to make a set of questionnaire that can help us to give more information about agile methods, trust and developer. Then we can post our questionnaire to extreme programming group to get feedback and answers.

ACKNOWLEDGMENT

Sincere thanks to all members of eXtreme Programming yahoo group who helped us in this studies.

REFERENCES

[1] Wilson, P., Pitty, K. (2003) *Stand-up Meeting Antipatterns* [Online] URL: http://fishbowl.pastiche.org/2003/11/19/standup_meeting_antipatterns [10th November 2007]

[2] Beck, K. (2006) *Daily Stand-up Meetings.* [Online] URL: http://www.extremeprogramming.org/rules/standupmeeting.html. [5th November 2007]

[3] Beck, K., Fowler, M. (2001) Planning Extreme Programming. *Addison Wesley Professional*. P-160.

[4] Laplante, P. (2003) Opinion: Stand and Deliver: Why I Hate Stand-up Meetings. *ACM Queue*. 1(7). 7-9.

[5] Yip, J. (2003) It’s Not Just Standing Up: Patterns for Daily Stand-up Meetings. *ThoughtWorks, Inc*.