Scrum in a Land Far, Far Away…

(or Using Scrum Across Dispersed Sites)

Rick D. Anderson
rick.david.anderson@gmail.com

Abstract

The agile project management methodology Scrum was designed for teams that are co-located. Unfortunately that is not a common reality in the high technology world we live in. Indeed, companies continue to expand geographically and overseas, more employees are working from home and many large corporations now hire the best talent regardless of project location.

When co-location is not possible, Scrum can still be used with some tailoring. This presentation will talk about Dispersed Scrum – how to use Scrum in with remote workers. Several successful techniques will be shared on how to use Scrum when the project team is not co-located including dealing with team communication, local site leadership, handling Daily Scrum Meetings and ScrumMaster and Product Owner challenges. Most of the techniques presented will be valuable to anyone who works with employees at remote sites regardless of methodology.

The paper assumes a basic understanding of the Scrum methodology. Primers on Scrum can easily be found on the web (see reference [8]) or in book form [9].

Biography

Rick Anderson has been with Wind River for 4 years and is the Engineering Director of the Android Solutions Technology Center, a world-wide group of experienced Android engineers who bring forth unique features into the Android ecosystem. In his over 25 years of embedded, real-time experience, Rick has worked at Tektronix, Interactive Systems and Lucent Technologies in a variety of engineering and management roles.

As a recognized expert in Software Engineering processes, Rick has presented numerous papers on improving software quality and productivity. He is currently serving on the program review committee for the Pacific Northwest Software Quality Conference and he chairs the Wind River Solutions & Services Software Technology Leadership Council. Rick is a certified Agile Scrum Master, an Agile coach and is a member of the IEEE and the American Society of Quality.

Rick graduated summa cum laude with a B.S. in Management and a B.S. in Computer Science from Oregon State University. In this spare time, Rick coaches competitive youth sports and leads Boy Scout high adventure activities.

Copyright Rick Anderson, 2013

1. Introduction

In a perfect world, product development teams are all located in the same city, building and even floor. They sit side-by-side and through the incredible power of co-location, they collaborate to dream-up incredible innovations, as well as design and implement killer products. Unfortunately for most of us, this perfect world does not exist. More often than not, development teams are spread across multiple and geographically separated sites. The expectations of these so-called "dispersed teams" are no different than their co-located counterparts – they must develop high quality products on time and with robust features sets.

While the number of dispersed teams continue to grow, so too does the move towards agile software development. Scrum, which combines software development and project management methodologies, seems to be the method that is making the most traction in the industry today. If one goes back to the original Scrum methodology as outlined by Hirotaka Takeuchi and Ikujiro Nonaka in 1986 [11], or even the modern versions captured by Dr. Jeff Sutherland in 1993 and Ken Schwaber in 1995 [9,12], one sees that Scrum is designed for co-located teams.

There are many challenges when using Scrum with dispersed teams. These include communication issues such as how does the ScrumMaster run the Daily Stand-Up Meeting, physical location issues such as how to show a demo to a Product Owner who isn't in the room and a variety of team organization challenges around training, accountability and schedule tracking.

So how does one then do Scrum with a dispersed team? While many experts have said this is not possible, with tailoring of process and setting clear expectations with your team, Scrum can successfully be used with dispersed teams.

This paper will further outline the challenges of using Scrum with a dispersed team and then provide specific techniques to overcome these. By following this advice, there is no reason a dispersed team cannot be as successful as a co-located one and remain agile at the same time. While most of the advice contained herein is targeted at those who have teams in multiple geographies and time zones, much of the practice outlined here applies also to Scrum teams who are potentially regionally co-located, but still cannot work together face-to-face on a daily basis. In addition, regardless of whether Scrum is used or not, those working with dispersed sites on a regular basis will probably find good advice in this paper as well.

2. Why Dispersed Teams?

In this paper, we will use the term "dispersed teams" to denote any product development team that is not co-located within face-to-face meeting distance of each other. This "dispersed team" could be at home workers, workers who are remote in the same region or country, or workers who are spread across multiple countries and time zones. For the most part, we will assume this dispersed team is spread across multiple geographies and time zones.

Before we dive in, one might ask "What is driving the increase in dispersed teams?" There are a few trends that seem to be causing more dispersed teams. First, outsourcing of work to low-cost geographies continues to occur as businesses focus on driving down expenses. Second, corporations are increasingly willing to hire the best candidate regardless of location, as long as they can report to one of the company's local offices. Third, companies continue to purchase other companies and often keep their local sites open, thereby creating a more dispersed company. Finally, more companies are allowing employees to work from home to avoid long commutes and help reduce pollution. While this does not contribute to geographically dispersed teams, it does create teams that are not always co-located.

3. Typical Solutions for Dispersed Teams

There are three primary methods of performing Scrum with a dispersed team. These are:

Integrated Scrum – This is your typical Scrum process. It consists of a single Scrum for the team, where dispersed team members participate just like they are co-located. Allowances must be made for the Daily Stand-Up, planning, retrospective and demo meetings. The integrated Scrum approach can work with a local or regionally dispersed team, but is rarely practiced with a globally dispersed one that is spread between multiple time zones (especially more than eight time zones away).

Isolated Scrums – In this model, each major local site runs their own Scrum process. These are not coordinated (hence the name, isolated). This works well when sites don't depend on each other. Local ScrumMasters and Product Owners drive the work. The isolated Scrum approach is rarely chosen because work often cannot be clearly separated by site with no dependencies between sites.

Scrum of Scrums – Each dispersed team runs its own Scrum process, but the Scrums are synced (start and end on time) and the outputs roll-up into a master Scrum process. Local ScrumMasters (and if possible, Product Owners) drive the work, along with an overall ScrumMaster who is driving the roll-up of the individual Scrums. This process can be complicated and dependencies between groups/Scrums must still be dealt with. This can be one of the best methods for using Scrum with large teams.

This paper will propose a fourth method which is a hybrid of the Scrum of Scrum and Integrated Scrum methods. This method has worked well for the author over the past six years. Let's call it a Dispersed Scrum.

Dispersed Scrum – A single Scrum across all sites (like an Integrated Scrum), but run locally by each major local site (like a Scrum of Scrums). In this model, you need one ScrumMaster per site. There is a single Sprint Backlog used by all teams, and during their local Daily Stand-Up Meeting, the local ScrumMaster updates the Sprint Backlog for all local employees. For example, in North America we have an 11am PDT Stand-Up run by ScrumMaster A who is located in North America, and in China we have a 2am PDT Stand-Up (but represents a local work time for them) run by ScrumMaster B who is located in China.

The Dispersed Scrum has the advantage of keeping a single Scrum which is managed locally and integrated across all sites. The following table shows the methods and trade-offs:

Method	# of Scrums	ScrumMaster	Coordinated
Integrated Scrum	1 ☺	At main site only	Yes ☺
Isolated Scrum	2 or more	At local site ☺	No
Scrum of Scrums	2 or more	At local site + main site SM ☺	Yes ☺
Dispersed Scrum	1 ☺	At local site ☺	Yes ☺

A Dispersed Scrum does not solve all problems, however. The planning, demo and retrospective meetings are still a challenge if individuals are located in different time zones. The majority of the planning step generally takes place over a 24 hour period where each site contributes. Specific user stories are selected from the product backlog prior to planning and loosely assigned to each individual and they are responsible for coming up with the task list and estimates for each task. This is supplemented by peer review instead of good practices like Planning Poker. These are added by the local ScrumMaster into the Sprint Backlog. Work generally starts immediately on the next sprint (it doesn't wait for the 24 hour period to end). Given planning is iterative across multiple time zones, it often takes 48 or even 72 hours to get a four week sprint completely planned out by the head ScrumMaster.

The Daily Stand-Up meeting updates a single Sprint Backlog, so it's easy to see what work was accomplished by co-workers at other sites the day before. Roadblocks are added to a common roadblock sheet and worked by local teams where possible, or saved for another team if appropriate. For demo and retrospective meetings, we put these back to back and attempt to get together as many people as possible even if it's an early or late meeting (once a month is generally deemed acceptable for a non-working hours meeting).

For the remainder of the paper, we will cover some of the key areas for success in using Scrum across dispersed sites. For each area, we'll share one or more challenges and then some techniques for success.

4. Crisp Team Communication

Challenges:

#1: Scrum and agility require a faster pace, which can stress a dispersed team. Said another way, agility often exposes communication weakness in a group.

#2: Scrum requires daily interaction, yet time zones and distance get in the way.

#3: Scrum means everyone must be involved. This can be more difficult for the introverted engineer. [Note that this is not technically a challenge that is shared only with dispersed teams.]

Techniques for Success:

#1: All email which requires an answer must be acknowledged within your working day. Where possible, email is answered during your work day. But if the answer is not available, you at least communicate receipt of the email and a target date when you can respond. This "rule" helps sets clear expectations to all team members around the speed at which communication must occur in the group.

Beyond this base rule, asking team members to check email early in the morning (7am when they first get up) and late at night (7pm or 9pm before they go to bed) greatly increases email turnover and the speed at which the team can get things done. This is generally a quick check to answer easy questions, not a lengthy email session. In some cases, we only ask for this level of commitment during critical times during the project or we rotate the responsibility among team members if it can be rotated. Another technique is to work split shifts (6am-10am and 2pm-6pm) which some actually find refreshing. The highest performing teams have incorporated some form of this secondary rule.

#2: Over communicate. In all communications, reiterate the question, provide an executive summary of the answer, then respond with or point to details and quantitative data to back up the answer. Document everything using common tools and common workspaces. It is important to put team information into a well-organized tool so the question and answer can be found by team members on their own. We've used wiki's and Jive (http://www.jivesoftware.com) successfully within our team. The team should work towards perfection in their communication, as every miscommunication costs the team 24 hours.

Perfection, in this case, means the responder does not have to ask follow-up questions to get the information they need (the communication cycles is limited to one question and one answer exchange). We use the phrase: "Perfect communication every time."

#3: Standardize on the communication tools used and the formats expected. For example, where are status reports kept, when are they due, what format should they be in? Beyond the typical Scrum questions "work done yesterday, work expected today and roadblocks", what do you need the team to report on and track to be successful. Be very clear about what is required and help the team by creating templates. This is especially important when more information is required than what is typically found in a Sprint Backlog. Again, tools like Jive and wikis can be helpful, although the information does have to be well organized and managed by the team to be of use. Recognize that because people are dispersed, having tools in the cloud can often be useful.

#4: Make it clear to the team that there is no hierarchy in the team and that everyone must speak up. In some organizations, a junior engineer would report to a senior engineer and very little of the junior engineers' communications would be passed along unfiltered. In order to be agile, this cannot happen. Every team member needs to speak-up quickly and internal team responses should be unfiltered. This idea might take some encouragement, mentoring and socializing within the team, but it is required if you want to truly be agile.

#5: Cultural and language training for all Scrum team members can go a long way towards helping the team work together more closely. Such training improves communication and helps the team members understand why their co-workers might be behaving in a way they are not normally use too.

5. Accountability with Bottom-Up Estimation

Challenges:

#1: Scrum exposes your best and your worse estimators. While this is not necessarily a unique issue for dispersed teams, off-shore resources often do not have the same training and experience doing estimation.

#2: Related, Scrum also exposes your highest and lowest productivity engineers. It is more difficult to "hide" your performance. While this is also not necessarily a unique issue for dispersed teams, it can bring into question the value of dispersed resources.

#3: The Product Owner is watching more closely when using Scrum. This can be intimidating to those not use to working with a real customer or a strong marketing person. This is exacerbated when dispersed team members do not speak the same native language as the Product Owner.

Techniques for Success:

#1: When using Scrum, it is very important to discuss the meaning of accountability and schedule ownership. Everyone needs to understand they own a piece of the schedule and if they do not deliver on schedule, it sets the whole team back. Scrum team members must understand the importance of doing solid estimation up front and providing feedback on others estimates if they feel they are incorrect. In my experience, many team members are afraid to add new tasks in the middle of the sprint, or to raise roadblocks early. The ScrumMaster needs to encourage this behavior. Additional training might be required here. The ScrumMaster must be willing to work with and mentor those who are struggling in any of these areas.

#2: Emphasize the schedule every day in your Daily Stand-Up meeting. Show the burn down and/or burn up chart. Highlight progress. Let the team know if they are ahead or behind. Engineers like to stay ahead of schedule. And of course, they don't like falling behind. Open and visual communication helps ensure positive behavior.

#3: Don't plan 100% of your available hours during your planning meeting. If you do, any unestimated tasks (the #1 source of missed schedules) will surely put you behind because you cannot recover. This also gives you a little flexibility to take on some stretch goals or to deal with emergencies that occur during your sprint.

#4: Use six hour work days. On my teams, tasks are estimated assuming you are doing nothing but working on the task (so called "heads down estimation"). But the reality is that the typical engineer has email to respond too, meetings and other interruptions during a typical work day. By planning for a six hour work day, you don't have to estimate with "overhead" built into your estimates. I have generally found this to be a more accurate way to perform estimation.

#5: Product Owners should recognize the enormous influence they might have over Software Engineers who are not use to working with a customer or a customer representative. Providing lots of positive feedback and give-and-take dialog will help establish a partnership of trust. The ScrumMaster should watch these relationships carefully and help both sides understand roles, project goals and convey ways to work together more efficiently.

6. Daily Stand-Up Meeting

Challenge:

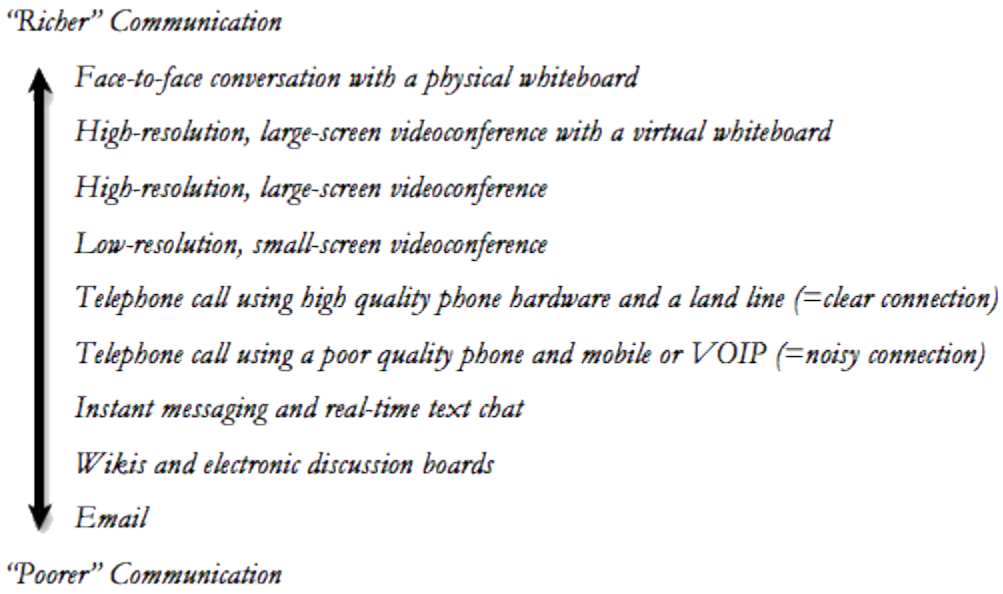
#1: A co-located Daily Stand-Up is impossible with a dispersed team.

Techniques for Success:

#1: Have local Daily Stand-Up Meetings. Using the Dispersed Scrum model described above, each major local site has a Daily Stand-Up meeting run by a local ScrumMaster. In essence, you have multiple Daily Stand-Ups. The Dispersed Scrum method uses a single sprint backlog, burn down chart, roadblocks list, etc. so all team members see everything else the team is doing. These process assets are stored in a commonly accessible place (the cloud is good given its accessible regardless of location). It is important to make sure they are under configuration management control and backed-up.

#2: Use the most appropriate tools designed for distributed work. While email is often an engineers' first choice for communication, it is often the worst choice if you wish to have a robust conversation.

Peter Deemer in his document on Distributed Scrum [6] talks about a "richness" scale for communication which can be seen here:



Source:http://www.goodagile.com/distributedscrumprimer/commsdiag.gif.

The basic idea is that visual communication (face-to-face, video conference) is better than verbal communication (telephone) is better than interactive written (chat rooms, text messaging) is better than non-interactive written (email).

When dispersed, video conferencing is great for Daily Stand-Up Meetings if you have the equipment and the bandwidth because you can see the other team members. Skype, WebEx, Google Hangouts and other tools and services exist if you don't have dedicated video equipment. When video conferencing doesn't work, text messaging using IRCs, IM or Chatrooms can work almost as well. When combined with searchable logs and the ability to go back into history, these even become more valuable.

Having status typed-up ahead of the meeting start, using an agreed upon format (for example, task/hours worked/hours remaining), starting on time and proceeding in a set order helps things move along. This method can also overcome some language barrier issues that might exist with video and spoken communication. I've also seen teams record a Daily Stand-Up meeting and post this video for dispersed teams. Whatever method you use, really try to maximize team communication while keeping the meeting short and to the point.

#3: Continue to have weekly group meetings. While this is often the responsibility of the Engineering Manager or Technical Lead/Architect instead of the ScrumMaster, having this additional interaction when you have a dispersed team remains critical. Having short training sessions, design reviews and other sharing sessions despite having a dispersed team helps keep the team together and productive.

7. Strong Local Site Management

Challenge:

#1: It can be hard for leaders (ScrumMasters, Product Owners or Engineering Managers) to understand exactly what is going on at dispersed sites. They have no eyes and ears to see the folks and hear the hallway conversations. There is often no one person you can go to for issues that fall between typical job duties. In an international setting, understanding local laws, customs and culture might be difficult.

Technique for Success:

#1: Having a strong designated leader at your remote site helps greatly. This can be any combination of a local ScrumMaster, Product Owner, Engineering Manager or Technical Leader. These individuals can act as the local translator if language issues are involved, they can be your eyes and ears, and they can help explain local laws, customs and culture. Finding the right leader is critical and once you find them, keep them! They will become very critical to your success. Have weekly 1:1's with them to really understand the heartbeat of the dispersed site.

8. Self-Organizing Teams

Challenge:

#1: Problem solving must be pushed down to the individual engineer with a dispersed team. Take an extreme example, where every team member is located in a different location. In this scenario, there is often no one to ask for help, and certainly no one sitting next to you that can help you. Co-located teams have a much larger "pool of knowledge" sitting next to them. Further, the number of common work hours might be limited due to geographically different time zones for project team members.

Techniques for Success:

#1: Build empowered and self-organizing teams. To build a really strong team, you must let the team figure out things for themselves. One of the great investments you can make here is to teach the team problem solving techniques. These add tools to their toolbox. Examples of some great problem solving techniques to be taught include Fishbone Diagrams and 5 Whys. Having weekly technical sharing meetings also helps and is a good forum for quick-n-dirty tips around tools and other areas.

#2: Leverage your retrospectives by listening to your team and driving problems to root cause. Sharing of lessons learned is so obvious, but rarely done well. See previous PNSQC paper by this author and others for some great suggestions on holding good post-project reviews [1], [2]. Also see Martin Fowlers article on Lessons Learned when using Agile with Offshore Development teams [10].

#3: While a strong ScrumMaster or Engineering Manager could solve most problems themselves, this doesn't help make the team stronger. Instead, identify problems, schedule a meeting and act as a moderator while letting the team figure the problem. This guided problem solving approach takes a bit longer, but makes the team stronger.

#4: The Four Hour Rule. This rule says that if you have spent more than four hours trying to solve a problem and you still haven't done so, you should ask for help. This means raise it as a roadblock and alert the team you are stuck. It's amazing how well this simple rule works at removing roadblocks by using the collective knowledge of the team and not letting a problem churn in an unproductive manner.

#5: Adopt a Defect-Free Mentality. This concept is one where you push quality upstream and the Software Engineers focus on producing work products free of defects. In essence, they try to put the SQE team out of business since there are no defects to find. It's a mental mindset that is lacking in most organizations. See this PNSQC paper for more on this [3].

9. ScrumMaster Issues

Challenge:

#1: It's difficult for a ScrumMaster to manage a dispersed team. They might have to attend daily stands-up in weird time zones, it's hard to get a real feel for the status of the project when you can't touch the product, and you don't hear the hallway conversations or see the panic on their faces.

Techniques for Success:

#1: Follow the Dispersed Scrum model. This says you should have one ScrumMaster at each local site. A local ScrumMaster can overcome many of the challenges a typical project faces. Certainly all ScrumMasters on the project must work closely together as well.

#2: Rotate the ScrumMaster role between those that have the right traits (see below). This strengthens the team. And there is no reason why the ScrumMaster can't be outside of Software Engineering, like Software Quality Engineering or Technical Publications.

A good ScrumMaster must be:

- Organized and prompt
- Helpful
- Patient and flexible
- Accountable
- Be detail oriented
- Have good communication skills

#3: Use the right communication medium for the job. As previously discussed, engineers have a tendency to use email for their default communication device because it's fast and easy. However, often times it is not the best method for achieving timely resolution. The preference for agility is interaction, so my encouragement to teams is to use the phone or video conference first, then some interactive text messaging method next and email last.

#4: Learn the local language, customs and culture. Certainly this is easier said than done, but there are books, websites and courses on all of these. The more a ScrumMaster understands and can communicate with the team, the more effective they will be.

#5: Visit your remote teams at least once per year and work in their time zone every once in a while to increase overlapping communication time. Sometimes a simple shift in work hours goes a long ways towards making a good ScrumMaster a great one.

10. Product Owner Issues

Challenges:

#1: A remote Product Owner has the same challenges as a remote ScrumMaster. This includes time zone and communication issues. Product Owners who cannot see and touch their product as it is being developed is certainly a challenge that must be overcome.

#2: Not having an active Product Owner is equally challenging. While this is not specific to dispersed teams, the challenge becomes great when you are dispersed.

Techniques for Success:

#1: See all the ScrumMaster Techniques for Success above. The Distributed Scrum Primer has a whole section on trust and the benefits of the Product Owner travelling to dispersed sites [6, and specifically page 2].

#2: Make sure your Product Owner understand their role. Level with them on the time commitment, participation expectations, communication within the group, etc. Be sure there is agreement up front on how planning and product demonstrations will be taken care of.

#3: Make adjustment to the product demo meeting so it's still relevant for all. Which method is best depends on your product and team location. There are a few different ways to do this:

- Assuming your product can be controlled via remote means, perform the demo live over the Internet.
- When the product can't be controlled remotely, using a live video feed via the Internet is next best. The Product Owner can ask questions in real time and the engineers can adjust the demo on the fly.
- Have the same physical hands-on demo at all dispersed sites, but at the same time. With this approach, the "demo master" guides everyone step-by-step on the execution of the demo.
- Video the demo ahead of the meeting and share the video via WebEx or other sharing mechanism during the demo meeting.

Related to this, be sure to still celebrate project success. While it might not be possible to have ice cream or a team lunch with a dispersed team, there are other mechanisms that can be utilized to share incremental milestones on your projects.

#4: Supplement the primary Product Owner with a local Product Owner. This might be an Engineering Manager, a Technical Leader/Architect, a Product Owner from another team, or anyone else who can represent the owner. Certainly the supplemental Product Owner must work closely with the Product Owner on all decisions.

11. Separate Software Quality Engineering Teams

Challenges:

#1: On many project teams, the Software Engineering (SWE) team is co-located and the Software Quality Engineering (SQE) team is also co-located, but not necessarily together. Specifically in many organizations, the SQE function is off-shore.

#2: Is the SQE team part of the Scrum or not? While the answer to this seems obvious and it is not specific to dispersed teams, it seems this question comes up a few times a year.

Techniques for Success:

#1: Have a separate SQE function. It remains a best practice to have a team responsible for product quality, writing automated test cases and watching over the Software Engineering teams work as a voice of the customer.

#2: Have the SQE team 100% integrated in the Scrum. So while they can be a separate role or group, they are absolutely part of the Scrum team. SQE retains their typical duties of test planning, test case creation and test case execution. A focus on daily builds and test automation is recommended. Part of the SQE group's roles should be to push quality upstream (software process improvement). There are many articles being written about how SQE processes can be modified to work with Scrum.

#3: As much as possible, treat the SWE and SQE teams as the same. They should be equals as far as job status, pay, responsibility, etc. If you have Software Engineering leaders, have Software Quality Engineering leaders. Their value is the same.

#4: Challenge the SQE team to do more than just manual testing. They are SWEs, but their role is to write test code instead of product code. Also encourage them to push quality upstream (software process improvement). These items give the SQE role more variety and make it more exciting.

12. Track Plan versus Actual

Challenge:

#1: Scrum normally tracks the original estimate and hours remaining (what did you do yesterday, what will you do today, how many hours are remaining). This makes it hard to identify slipping tasks and those who have estimation problems. While this challenge is not specific to dispersed team, it can be a bit more difficult to identify those who have estimation issues if you aren't in a daily stand-up with all team members every day.

Technique for Success:

#1: Track actual work hours. With just a minor change, your Daily Stand-Up Meeting and Sprint Backlog sheet can be modified to keep plan versus actual hours. I found this to be a huge win in tracking the success of my projects and helping those who needed it with estimation and problem solving.

Below is an example of a spreadsheet-based Sprint Backlog. You will see that it shows the Initial Estimated Hours during the planning meeting (Initial Est'd Hrs column), but also the running total of the number of hours worked on the task (Hrs Worked column). Each day during the Daily Stand-Up Meeting, individuals report the task they worked on, the # of hours they worked on it and the number of hours remaining. The # of hours worked is added to the # of hours in the Hrs Worked column. The # of hours remaining get recorded in the appropriate day column (for example, 14-Sep). Looking at this table during your retrospective, it would be easily to see that the "Add alpha blending feature" was under-estimated.

Backlog Item	Task	Resource	Task Status	Initial Est'd Hrs	Hrs Worked	10-Sep	11-Sep	12-Sep	13-Sep	14-Sep
Jelly Bean Security	Test and debug	Rick	Not Started	9	0	9	9	9	9	9
	Add alpha blending feature	Rick	Completed	12	18	12	12	6	6	0
	Gingerbreak security exploit	Rick	In Progress	6	0	6	6	6	6	4
	Combine existing code & open source	Rick	Completed	6	6	6	0	0	0	0
	Video	Rick	Not Started	12	0	12	12	12	12	12
	Demo image creation	Rick	Defer	12	0	12	12	12	12	0

13. Other Tips for Success

There are several other tips for working with a dispersed Scrum team that won't be covered here, but include:

- Use of appropriate tools that can be used across geographies (cloud-based tools come to mind)
- Daily builds used by all sites
- Good configuration management techniques (CM becomes very important to project success; consider having a dedicated person responsible for this, potentially at each dispersed site).
- Visit your team on a regular basis. Getting the team together even when dispersed helps with team morale and productivity.

14. Conclusion

This paper has presented a number of challenges that a dispersed Scrum team faces as compared to a co-located Scrum team. For each of these challenges, one or more Techniques for Success have also been given. Hopefully you will find these helpful in your efforts to become more agile. I'd like to leave you with the following final suggestions:

- Building a high performance dispersed Scrum team takes lots of patience and work, recognition that it will not happen overnight (nor in your first sprint) and two-way trust (empower the team, hold them accountable).
- Create an environment where "we are all in this together".
- Use retrospectives as constructive events.
- Celebrate success

Good luck!

15. References

[1]. Anderson, Rick D. and Gillmore, Bill, 1997. "Maximizing Lessons Learned: A Complete Post-Project Review Process". 15th Annual Pacific Northwest Software Quality Conference Proceedings, page 129. http://www.uploads.pnsqc.org/proceedings/pnsqc1997.pdf

[2]. Lavell, Debra, 2007. "Facilitating Effective Retrospectives", 25th Annual Pacific Northwest Software Quality Conference Proceedings, page 329. http://www.uploads.pnsqc.org/proceedings/pnsqc2007.pdf

[3]. Anderson, Rick D., 2007. "Flexing Your Mental Muscle to Obtain Defect-Free Products". 25th Annual Pacific Northwest Software Quality Conference Proceedings, page 49. http://www.uploads.pnsqc.org/proceedings/pnsqc2007.pdf

[4]. Buecker, Inge, 2008. "Architecture tip: Adapting the Scrum project management method for geographically separated teams". http://www.ibm.com/developerworks/library/ar-scrumremote/

[5]. Rothman, Johanna, 2012. "Agile Lifecycles for Geographically Distributed Teams, Part 1". http://www.jrothman.com/blog/mpd/2012/01/agile-lifecycles-for-geographically-distributed-teams-part-1.html

[6]. Deemer, Pete. "The Distributed Scrum Primer", http://www.goodagile.com/distributedscrumprimer/index.html

[7]. Puopolo, John, 2007. "Be There or Be Square – A Case for Collocated Teams", http://www.scrumalliance.org/community/articles/2007/august/be-there-or-be-square .

[8]. Deemer, Pete and Benefield, Gabrielle, 2007. "The Scrum Primer – An Introduction to Agile Project Management with Scrum", http://www.scrumprimer.com/

[9]. Schwaber, Ken and Beedle, Mike, 2002. "Agile Software Development with Scrum".

[10]. Fowler, Martin, 2006. "Using an Agile Software Process with Offshore Development". A list of several lessons learned from ThoughtWorks experience.

[11]. "New New Product Development Game". Harvard Business Review 86116:137–146, 1986. January 1, 1986. Retrieved March 12, 2013.

[12]. Sutherland, Jeffrey Victor; Schwaber, Ken (1995). *Business object design and implementation: OOPSLA '95 workshop proceedings*. The University of Michigan. p. 118. ISBN 3-540-76096-2.

Others:

http://msdn.microsoft.com/en-us/library/jj620910.aspx

http://www.nicta.com.au/pub?doc=4637

http://www.solutionsiq.com/Portals/93486/docs/distributed-scrum-why-some-teams-make-it-work.pdf