Nine Low-Tech Tips for Project Management

Moss Drake

drakem@dmcdental.com

Abstract

Project management can be tough, especially when technology gets in the way of the goal: to develop software that provides value to the users. Creating a complex software system becomes even more difficult when your processes rely on other complex systems.

Experience shows that project management is made easier by reducing complexity whenever possible. Considering this, perhaps it is better to choose the simpler, low-tech solution when one is available. Additionally, some low-tech practices can lead to higher visibility, more democratic processes, and simpler solutions.

This paper is an experience report on nine simple, low-tech solutions used for project management at our company that helped improve our process, reduced complexity and kept the project focused on the goals rather than the technology. These tips include using story cards, prioritizing the backlog, sizing stories, using information radiators for project status, collaborating using drawing and simple screen mockups, and increasing face-to-face interaction.

Biography

Moss Drake is a software developer and project leader at Dentist Management Corporation in Portland, Oregon. He has been developing software for over twenty-five years, focusing on solutions for the health care and insurance industries.

Moss has a B.S. in Computer Science and a B.A. in English Literature from Oregon State University.

1. Introduction

“Somewhere today, a project is failing.” (DeMarco 1987) These are the first words in chapter one of Tom DeMarco's book *Peopleware*. His thesis is that the major hurdles encountered in software development projects are not so much technological as they are sociological. For years, people have been developing and redeveloping the same or similar projects, and yet one somewhere today is in trouble. The problem is that high-tech people, enamored with the technology, look to code themselves out of a problem rather than interact with the team. DeMarco points out that because we go about software development in teams and projects and other tightly knit working groups, we are mostly in the human communication business. In 1987, this may have been a cry in the wilderness, but by the time of the Agile Manifesto (Beck and al. 2001) in 2001, it was commonly accepted. As the manifesto proclaims, the focus should be on “individuals and interactions over processes and tools.” (Beck and al. 2001)

Peopleware proposes changing physical workspaces and mental attitudes to promote communication, but communication can also be improved by simpler project management processes. Project management can be tough, especially when technology gets in the way of the goal of developing software that provides value to the users. Creating a complex software system becomes even more difficult when processes rely on other complex systems. Project management is easier if you can reduce complexity whenever possible. Considering this, it is better to choose a simpler, low-tech solution when one is available. Additionally, some low-tech practices can lead to higher visibility, more democratic processes, and simpler solutions.

2. Background

Dentists Management Corporation (DMC) is a subsidiary company of Moda Health. DMC develops clinical and business management systems for dental offices. Our development team is small, about six or seven people. Projects usually last six months to a year, although multiple endeavors are in play at any time. In addition to customer-facing projects, DMC must also adapt to regulatory changes such as HIPAA and the Affordable Care Act.

The path that led to our current project management process was not a quick one. Having explored multiple project management methods over the years, including modified waterfall methods and a couple failed attempts at Agile, we now use Scrum for most of our non-regulatory projects. We found that some of the techniques learned during our initial Scrum training continue to shine through as excellent practices, especially because they are so simple. Other aspects, however, were more opaque at the time. It took repetition, over multiple sprints and projects, before we realized the full value of the activity.

3. Low-Tech Tips

In his *Ignite* talk, Adam Light asserts, “Scrum moves organization from the individual level to the team level.” (Light 2013) For managing projects, this means:

- Helping people take their ideas into a shared work space in a structured and efficient way.
- Getting the right people working on the right things at the right time.
- Building on the shared ideas to create better solutions

In short, it means collaboration: moving the project management process from the project leader to the team. The Scrum Master becomes a team member who has the role of facilitating the progress of the project rather than the project leader.

The Scrum Master provides the tools and environment to make the project easier, and allows the team to self-manage. Self-managing, however, is not a goal but a habit, and like any habit it takes time to become innate. The nine tips discussed in the following sections provide a set of good habits that lead to self-managing teams. They all have the common goal of using low-tech methods to make ideas visible and shared, enhancing collaboration.

3.1. Take a Field Trip

At the beginning of every project, we develop a vision document. The goal of this document is to provide a vision of how the product will work once the project is complete. The contents of this document are words and diagrams, pages and pages of writing in an attempt to bring imagination to life. The vision document is supposed to be the inspiration for the project, but often it is a flawed vision only approximating the actual results.

There are two good reasons why a vision document cannot possibly hope to communicate everything. The first is because of the fuzzy front end (New product development n.d.). A vision document is meant to inspire, but the project could be killed at any point so only a limited amount of work is spent defining the edges of the system. Second, while the vision document may talk about personas, it often does not describe actual system users in the way that a novel might bring a character like Harry Potter to life.

One way to fully realize the environment and characters that will populate your software solution is to take a field trip to sites where the software will be used. Visiting a site brings concrete images to the abstract ideas summarized in the vision document. While it may not be feasible or affordable to bring the entire team on site, even video and phone calls give an enhanced understanding of the environment. The goal is provide a deep impression that will help with development later in the project.

The most obvious observations to make are the workflows. The business analyst is best at this process. For the technical staff, the environment may play a factor in developing a solution. For example, dental operatories must be hygienic, so keyboards and mice are wrapped in disposable plastic covers. Since this makes touch screens unfeasible, we had to investigate voice and motion recognition solutions. In other offices, we discovered that staff had such cramped working spaces that they did not have desk space for their paper notes, so they ended up holding them in one hand and typing with the other, which raised a usability issue.

The field trip is a good time to put faces to personas. Previously you may have thought of the receptionist as “Receptionist 1,” but now you know her as Betty, who is extremely good at multi-tasking until it comes to her software. We would not have imagined the reception desk was such a hub of activity until we saw Betty interrupted nearly fifteen times in a fifteen-minute interval. Observing this not only impressed on the programmers and testers the importance of avoiding any software failures, but also provided data for system response times. Betty would be extremely unhappy to take time to call for support if something went wrong with the software.

There are other ways to observe the system: shadowing through screen-sharing sessions, conference calls, setting up role-playing scenarios at work, but none of these bring the situation to life as well as a field trip.

3.2. Make the Story Cards Work Overtime

Numerous articles on Scrum discuss story cards. They usually explain how the cards contain a user story, an informal statement about the value of a feature in the system to someone. A common form of the statement is:

As a <role>, I want <goal/desire> so that <benefit>. (Wikipedia entry User story n.d.)

The articles explain that the cards are merely an artifact to begin the conversation about the full requirements of the feature. Some teams transcribe the stories into spreadsheets, documents or project management systems.

At this point, some people may toss out the cards, but wait! Even if the story is documented electronically, the physical card has value for the project. Since the principal goal of story cards in Agile development is to raise visibility and focus for each feature, keeping story cards posted in a public place is a good way to do this. Think of the card as an icon for the full feature.

The card can also be a shortcut for the additional details associated with each story. For example, on each card we write a unique tracking ID, the story size, and the initial priority. If the story has a particular champion, the PO writes that person’s name on the back of the card.

As the project progresses, each story card will become a familiar artifact, bringing joy or dread, memories and ideas. The story card becomes memorable token that stands for the full feature. The position of the card can also carry meaning, like a flag. Turn the card sideways to remove it temporarily from the action. Turn it upside down to indicate that the feature is in distress. If conversations about the story become contentious, use the story card as a talking stick, allowing each person to speak their point of view as long as they hold the card.

Even when a story is tracked electronically it is useful to revert to low-tech by printing the summaries on labels and putting them on sticky notes. For multiple, parallel projects use different colored cards for each one.

A word of warning about using story cards: watch out for the nighttime janitor or strong air conditioning. If there is a mysterious gap in the backlog, you might want to look around on the floor. This emphasizes the need to keep an electronic copy of the stories in a spreadsheet or requirement management system.

3.3. Use an Insertion Sort for Prioritization

To be successful, Scrum requires a prioritized backlog of stories. Teams focus their efforts during the sprint on the items with the highest priority. Often, explanations of Scrum imply that the Product Owner (PO) already has an optimally prioritized the backlog (Cohn, Learn About the Scrum Product Backlog n.d.). That sounds like an easy request, until you realize the process required to achieve prioritization. Supposedly, the PO sets the priority, but that assumes a fully informed PO, which may be asking a lot of one person. Each stakeholder involved with the project has his or her own set of priorities, and backroom lobbying for features can cause conflicts. It is better to have a transparent prioritization process.

There are many ways to prioritize the backlog, but some tend to exclude participation more than others do. For example, when using a spreadsheet, database or document to organize the priority, only one person at a time can make modifications, so it is not an inclusive process. In planning meetings, no matter how large the display, the entire backlog cannot be seen at once, so attendees end up seasick watching the screen bounce around while one person sets the priorities. In other cases, when someone has assembled a printed document with the complete backlog, the prioritization process becomes a bubble sort on paper, stepping through the list, comparing items and swapping them if they are in the wrong order; not the most efficient method. In all of these cases, providing a pre-processed list allows some priorities to slip by without any discussion.

A simpler solution is a physical process that involves an insertion sort. In this method, the PO convenes a prioritization meeting and anyone who has a stake in the project can attend. The meeting space must have enough space to display all the story cards on a wall or window. The process steps are:

1. The PO places a random story on the wall, and then pulls another from the backlog.
2. The PO asks whether the current story has higher or lower priority than the one on the wall.
3. People in the room may offer their advice, but the PO has the final authority for placing the story card.
4. If the current story has a higher priority than the one on the wall then the PO places it to the left (meaning higher priority), otherwise it is placed to the right (lower priority). All stories must have a unique priority, so they should all be at the same level horizontally (see Figure 1).
5. Leave space between the cards so it is easier to organize them if a subsequent story is prioritized between two others.
6. One by one, the PO continues with the remaining stories, reading them aloud, getting advice, and then placing them into the correct priority.
7. When all stories have been prioritized, write the initial priority on the front of each story card for tracking purposes because priorities change over time. It is also useful to track the priority in a spreadsheet or requirement management system.

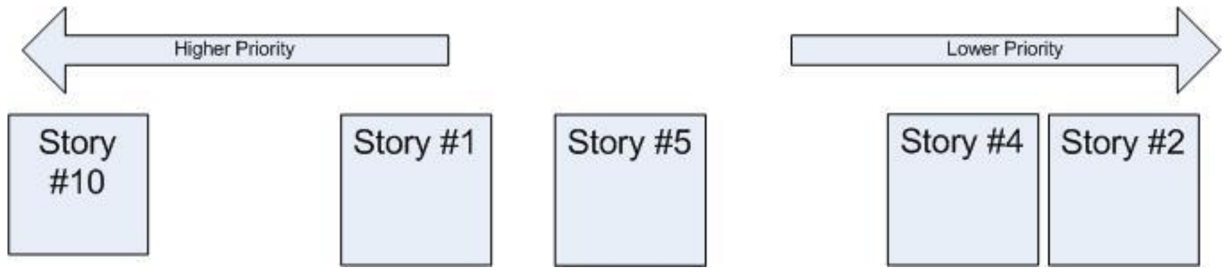

Figure 1 Uniquely Prioritized Stories

Using this process, our team has been able to prioritize up to 50 stories in a two-hour meeting. It is not recommended to try for more than this since people get decision fatigue and attention spans decline. Depending on the size of the team and of the stories, it may not be necessary to prioritize beyond the top 50 items. Since the process is an insertion sort, it is easy to split the process into multiple sessions.

The benefits of this approach are:

- Each story has its own moment under the spotlight—there is no a chance for it to be lost.
- It is a more democratic process, where the stakeholders can lobby the PO on behalf of their preferred stories and the PO becomes more fully informed.
- The unique prioritization avoids the situation where multiple stories end up with the same priority. It forces people to acknowledge the constraints of time and resources early in the project. It also avoids vague priorities where some stories are “high,” some are “medium” and some might be “medium-high.”
- The space allows everyone to participate, and the spatial orientation helps people better understand the scope of the project.

There are some potential variations to this process. When two project teams are working in tandem, it is acceptable to create two physical lines. This may expose dependencies. The prioritization process can also help define the minimal viable release points. The PO can move along the horizontal axis to identify the earliest acceptable release point. We maintained the prioritization as we moved the stories to the information radiator (see section 3.5). When a story is added to the backlog, it will have the lowest priority until the next sprint planning meeting when there will be an opportunity to re-prioritize.

3.4. Keep the Sizes Relative

In Scrum, the product backlog shows the amount of estimated effort required to deliver each story. While there are other methods of estimating, such as Mike Cohn's Planning Poker (Cohn, Planning Poker: Agile Estimating n.d.), T-Shirt sizing is a simpler way to get a baseline idea of effort. The human brain is better at estimating relative sizes than absolute sizes. Relative sizing is simpler since less information is required. Team members need only to recognize which stories are larger than others. The goal is to give the stories T-Shirt sizes (XL, L, M, S, and XS) relative to each other. Then, after several iterations of tracking the stories completed in a sprint, the team will have a better idea of the schedule for the remaining stories.

A problem, however, is introduced when some team members try to correlate relative sizes with absolute amounts, such as “S = one day”, while others may use a different scale. Workdays are a subjective unit of measure and depending on other commitments, ideal days will vary. Using a simple mechanical method avoids this problem.

At DMC, only the Scrum team participates in the sizing process. Team members arrive at a sizing session with an idea of the stories and with any information that might inform their estimates.

The sizing process is similar to the one described above for prioritizing stories, except the Scrum Master leads the session. After an arbitrary story card is placed on the wall, the Scrum Master pulls the next card, reads it aloud, and asks if people think this story is larger, smaller, or about the same size as the one the wall. The vertical axis represents the size of the stories with higher on the wall indicating more effort; lower on the wall is less effort. People in the meeting say “higher” or “lower” and the Scrum Master follows their suggestions. If the team cannot agree on the size, then it is open for a five-minute discussion, and the Scrum Master decides the final position.

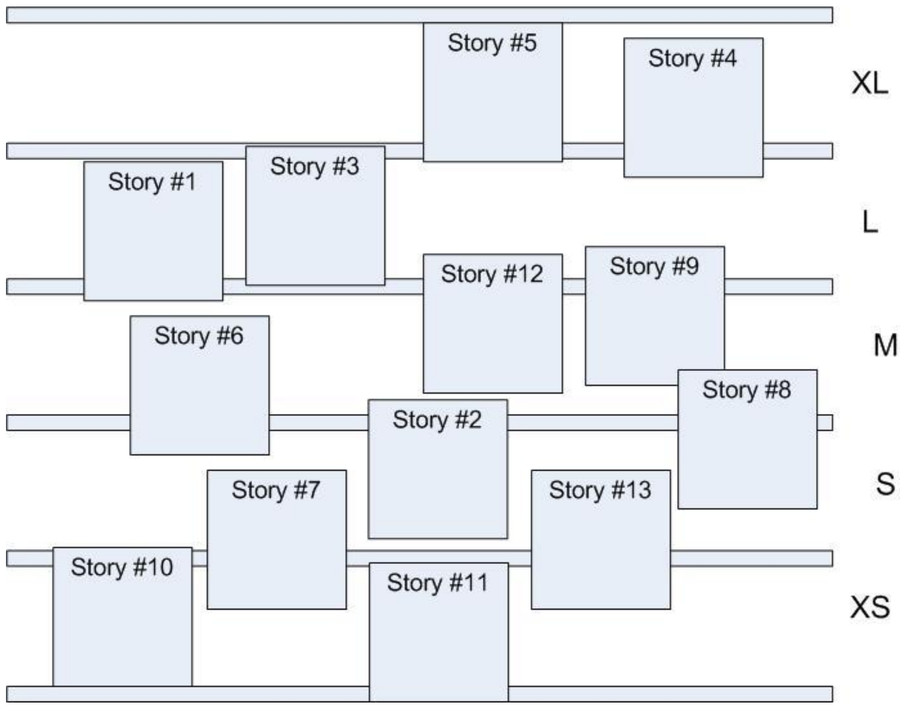

Figure 2 Relatively Sized Stories

After all the stories have been placed on the wall relative to each other, divide the wall into five equal vertical sections. Mark the sections XL, L, M, S and XS, respectively. Drawing the five sections is a physical method for categorizing the stories by size (See Figure 2).

Sizing the stories this way creates a visual map of the project in terms of effort. Reviewing it may highlight red flags hidden in the project. The Scrum Master should watch for L and XL stories that have lower priorities; in the last third of the project, for example. Large and XL stories toward the end of a project represent riskier work. Software projects have enough uncertainty that teams will want to avoid planning large stories at the end of a project. The PO may want to think about raising their priority, dropping them from the project, or trying to break them into smaller features.

Since the stories sizes are relative, the majority of the stories should fall into the small, medium and large categories. If there is a larger ratio of XL stories then the team should re-examine the stories that fall between L and XL. Perhaps the team has overlooked some details of the stories, or perhaps they are epics that should be split into multiple, smaller stories.

Note: relative sizing only works for stories considered during the same session, but the simplicity of the process makes it especially accessible for teams and team members who are new to Scrum.

3.5. Broadcast with Information Radiators

First coined by Alistair Cockburn, an information radiator is the “generic term for any of a number of handwritten, drawn, printed or electronic displays which a team places in a highly visible location, so that all team members as well as passers-by can see the latest information at a glance” (13Ag). Information radiators provide succinct visual messages for anyone who is in the area; similar to how a construction zone sign proclaims “187 days without an accident” to both workers and those who drive by. The goal is to convey current project status to people who are not involved on a day-to-day basis and to provide enough information to avoid interrupting the team with questions. Posting this information also implies that the team acknowledges the status of the project and is willing to share it with anyone.

Information radiators are one of the best low-tech devices for broadcasting the status of a project. Since they are visibly posted, it is a passive process. No one has to seek out the status or refresh a web page. As Cockburn mentions “online files and web pages generally do not make good information radiators, because an information radiator needs to be visible without significant effort on the part of the viewer.” (Cockburn 2004)

At DMC, the information radiator shows:

- The current sprint’s stories and related tasks
- A Kanban-style board with work under progress
- The current build number and timestamp
- The most recent release number and timestamp
- All the stories from the product backlog

Other items that might be useful to post are:

- A graphical display of the number of tests planned versus passing
- Any planned resolutions discussed during the most recent sprint review
- The status of any of the team members if they will be on vacation or otherwise unavailable

At a glance, anyone can tell how many more items are in the backlog for the current project and which stories are under development during a sprint.

When we first started using an information radiator the workspace presented some challenges. It lacked large areas of windows or walls, and the facilities staff was reluctant to hang whiteboards due to the newness of recent renovations. Also, the development area was in a quieter spot, so not many people passed by.

Fortunately, we overcame those problems. Instead of a whiteboard, we used the cubicle walls for posting the backlog. Since it was difficult to get sticky notes to adhere to fuzzy cubicle walls, putting a long piece of strapping tape on the cubicles made a non-porous surface that worked better. The development area lacked space to display everything together, so we compromised by posting the backlog in one area, and the sprint status items in a slightly different area, but still visible from one spot. Moreover, passing traffic increased when sales materials were stored in an area just past development. Now the sales and training staff pass by the information radiator on a daily basis. Recently I saw a sales person looking at the cards, understanding what we are working on.

The biggest benefit of the information radiator is for the team itself. It creates a sense of completion as the backlog whittles down, and the sight of a once-daunting project dwindling down to a small tail of outstanding stories is a sure morale booster.

3.6. Draw Out Ideas

Reviewing architecture and design documents can be tedious and boring since the process is passive: sitting in meetings trying to evaluate someone else's design work. Participants easily check out, often keeping their eyes and hands occupied by doodling in their notebooks while the meeting uses their ears and brain.

Drawing, however, is an excellent approach to engage the brain in multiple ways, making a more creative and memorable meeting for everyone. A recent Wall Street Journal article began "Employees at a range of businesses are being encouraged by their companies to doodle their ideas and draw diagrams to explain complicated concepts to colleagues." (Silverman n.d.) Drawing can be used to promote engagement, visual thinking, and enhance note taking.

I used to prepare diagrams or schema using Visio and hand them out in meetings only to realize that people were not fully engaged. I had gone through the journey of creating the visual, but for everyone else it was just a static image. Looking at a picture can be a fleeting event and the less energy one puts into it, the less memorable it will be.

In design meetings now, however, I arrive with an image in my head, and make sure I have a space to draw it out. The process is more engaging and we share the journey as the drawing is created. Describing the design and drawing it out simultaneously combines two methods of learning, sight and sound, helping to reinforce the ideas. This technique has a number of benefits.

- Telling stories while drawing helps people who learn using different styles. Drawing is a process and a result, your mind carves out the lines as you examine the subject.
- The images will transcend language: drawing out ideas works even when colleagues do not share a common first language or have the same level of technical understanding.
- Drawing is participatory: The brainstorm is directed and scoped, but democratic. Others may draw on the board or paper as well. It helps foster involvement in the critical thinking process, whereas a completed and polished image may give the impression that it is beyond criticism.
- Drawing brings out creativity. As the cartoonist Lynda Barry says about her graphic memoir/how-to book *What It Is*, "drawing or expressing your image is a therapeutic biological change that occurs in your body and makes you say 'ah!'" (Barry 2008).
- Studies have shown that drawing enhances retention. In *Moonwalking with Einstein,* Jonathan Foer devotes a chapter to discussing mind maps and how they correlate with memory palaces (Foer 2011). A memory palace is a visual mnemonic device used to help organize and recollect bits of information. As the team works with the image, they inadvertently create spatial memory palaces of the system in their minds. Later, when writing code or testing, the image can help clarify details and provide a context within the larger system.

To take it to the next level and get people involved, give them a pen and ask them to include their ideas in the drawing. If people object, saying they cannot draw, suggest instead that they "make marks on paper."

In addition to using drawing in meetings, drawing helps with other parts of the design process. Kevin Cheng, author of *See What I Mean* (Cheng, See What I Mean: How to Use Comics to Communicate Ideas 2012) promotes the idea of drawing comics for business reasons. Creating storyboards and scenarios as comics not only engages people in the process, but also saves time and effort. The sequential comic frames are simply another way to represent the steps in the user stories. Drawings can also be used to visualize the user experience, which is the main idea behind his web comic “OK/Cancel.” (Cheng, OK/Cancel n.d.)

Save the drawings at least until the end of the project. Even when working on a whiteboard you can still take a photo of the board and the memory of drawing it will stay with people.

3.7. Simplify Design with Paper Prototypes

Requirements people, designers, and developers think on different levels. When they meet to plan what needs to be done they focus on different aspects of the specifications rather than on the problem that needs to be solved.

One way to avoid this is to embrace paper prototypes. When the team goes to work on a screen design, the fastest way to get everyone’s participation is to use paper, markers, scissors and tape or glue stick. In a just a few minutes, one team member can sketch out an initial screen and then let others add to the work. Using black markers enforces simple screen design and removes colors and fonts from the equation. If people make mistakes or suggest changes, tape and scissors make it easy to rearrange the screen. A photocopier can be used to duplicate parts if necessary.

Working interactively on the page turns it into a shared design problem. Everyone in the team can participate in naming items, and the business analyst can remind the team of the real-world terms. Once the prototype takes shape, testers can apply their tests to the paper: How many characters? Is this required? Where do you go when you click on this?

Paper prototyping has been used for years, and it continues to be useful in this age of mobile devices. Whether you are designing a screen for a tablet, a smart phone, or a computer, the basic rules are the same. Using a paper model makes initial design easier. Print out an oversize template of an iPhone, photocopy it, and then use that to draw in the screens with markers. To replicate the user experience, you can stack up the pages, hold them in your hand like a phone, and then flip through the story.

3.8. Increase Communication by Pairing

Many Scrum proponents endorse creating cross-functional teams, but they do not always describe in detail the team mechanics. The high-level description is usually something like “Scrum relies on a self-organizing, cross-functional team. The Scrum team is self-organizing in that there is no overall team leader who decides which person will do which task or how a problem will be solved.” (Cohn, What is Scrum Methodology? n.d.) As a result, team members may end up creating their own personal silos, with developers writing code, testers checking it, the business analyst reviewing, all without collaboration. Consequently, the hand-off points become discrete, and the communication between team members is limited.

Similar to cross-functionality, many Agile evangelists also point to pair programming as a way to create better code with fewer bugs than when programmers work alone. But, why limit pairing to only developers? By pairing developers with testers or testers with the technical writer, or the business analyst with the developer, you can achieve benefits similar to pair programming for the entire team.

For example, we occasionally have the developer write the rough draft of the user documentation. This process helps the developer better understand the user workflows and often uncovers gaps in functionality. Moreover, as the technical writer sees early versions of the system, he can look at it from the point of view of documenting user workflow and suggest changes to the UI to improve usability.

In the same way, asking a developer to sit with the tester while creating the initial test plan increases the informational bandwidth between them. The developer can point out areas that may need more scrutiny or were touched in development but not obviously connected to the main functions of the code. And, the tester may notice areas where the developer may not have fully understood the implementation of a feature. As they create the test plan, the developer may recognize that the tester is not working with the same mindset and steps that he assumed while writing the code.

These pairings are a great way to shorten the loop. Throughout the development process, the goal of the Scrum framework is to increase participation, ideas, and the general quality of the product through shortened feedback loops. This means each minor iteration of the process should have feedback, and meeting face to face is the best way to accomplish this.

Managers and Scrum Masters can do the following to make pairing sessions successful:

- Make sure that the team has time to pair and suggest including pairings as tasks attached to stories. This raises the visibility of the pairings and allocates the time.
- Ensure that both sides get encouragement. Developers like to show off their code, but can be protective of it. If the interaction is confrontational, it will not work and will not become a habit. The Scrum Master should plan to attend the first couple pairings and encourage a positive outcome.
- Keep the sessions concise. When working on complex problems, pair programming and other types of pairing is useful, but difficult. It is two different minds working together on the same problem and sometimes people pull in different directions. Set aside time for the pairing, but avoid forcing the issue. Let the team self-organize.

The result of pair programming, testing, writing, and other sessions is that members of the cross-functional team will get a better understanding of the specialties of the other members in their group and seeing other people's perspectives helps everyone create better solutions.

3.9. Keep a Project Diary

All of the previous tips have provided ways to increase project visibility, participation, and collaboration. Those ideas were designed for a disparate audience: the team, managers, other employees, customers, and executives. This last idea, however, is to help you collaborate with one specific person: your future self.

Keeping a log of activities is not a new idea. Projects were tracked when Stone Age people drew pictures of a hunt on cave walls. Most tracking consists of structured metrics used to show managers and executives that the project is on target for the impending future release. Keeping a diary, however, is a way to track where the project has *been*.

A daily diary, like a personal diary, is meant for your eyes, and should be written for you as the audience. It can contain many bits of information that might not incorporate well into the project process. For example, if you have a complicated email exchange, do not leave it in your email client. Spend time extracting the exchange into paragraphs and annotate with pertinent details. Paste everything into the diary, and date it. Later, when someone asks, “What were we thinking at the time?” the diary will act as a personal reference to jog your memory.

Another use of a diary is to note exceptional efforts by team members. If someone on the team has a shining moment, make a note of it in your project diary. For supervisory staff who write employee performance reviews or provide feedback to Human Resources for employee assessments, these examples in the diary will prevent a lot of head scratching later and will benefit everyone.

Most importantly, the project diary is a way to provide a history for you. After many projects, you may wonder what you have accomplished, and how you got there. Since the diary entries were personal for you, you can include ideas, guesses, feelings and emotions among the timeline and events. During retrospectives, the diary can help you prepare your notes. Years later, when you recognize a situation, the project diary provides a reference, of not only how the situation was handled, but also your thoughts at the time.

4. Conclusion

The process of developing software is essentially one of taking ideas and making them tangible. Since it is usually accomplished with a team of people assigned to create a complex software solution, it is necessary for the team members to communicate their ideas and opinions to others. High-tech teams often navigate toward high-tech solutions to accomplish this goal, but those solutions come at a cost and can complicate the project management process in unexpected ways.

Even Agile projects are complex. Although Scrum reduces some of the complexity, developing modern software solutions is fraught with risks and unforeseen dependencies. Simplifying the project management process frees the team to focus on creating the solution, rather than occupying themselves with administrative tasks. Additionally, involving the team more in the project management process achieves the Agile goals of collaboration, short feedback loops, and user-centric development.

While all of the tips presented in this paper have been used in conjunction with Scrum, they are not limited to an Agile project framework.

Many of these tips may not be new to you, but I hope that you see them in a new light. The low-tech goals of collaboration, increased communication, and democratic processes align with the tenets of the Agile Manifesto to promote face-to-face communication, people over process, and collaboration over silos.

References

Agile Alliance. http://guide.agilealliance.org/guide/radiator.html (accessed 07 09, 2013).

Barry, Lynda. *What It Is.* Drawn & Quarterly, 2008.

Beck, Kent, and et al. *Agile Manifesto Principles.* 2001. http://agilemanifesto.org/principles.html (accessed 07 07, 2013).

Cheng, Kevin. *OK/Cancel.* http://okcancel.com/ (accessed 07 07, 2013).

—. *See What I Mean: How to Use Comics to Communicate Ideas.* Rosenfeld Media, 2012.

Cockburn, Alistair. *Crystal Clear: A Human-Powered Methodology for Small Teams.* Addison-Wesley Professional, 2004.

Cohn, Mike. *Learn About the Scrum Product Backlog.* Mountain Goat Software. http://www.mountaingoatsoftware.com/scrum/product-backlog (accessed 07 07, 2013).

—. *Planning Poker: Agile Estimating.* Mountain Goat Software. http://www.mountaingoatsoftware.com/topics/planning-poker (accessed 07 07, 2013).

—. *What is Scrum Methodology?* Mountain Goat Software. http://www.mountaingoatsoftware.com/topics/scrum (accessed 07 07, 2013).

DeMarco, Tom and Lister, Timothy. *Peopleware: Productive Projects and Teams.* New York: Dorset House Publishing Co., 1987.

Foer, Joshua. *Moonwalking with Einstein: The Art and Science of Remembering Everything.* New York: The Penguin Press, 2011.

Light, Adam. "Why Scrum Works (and How to Sustain it)." *Ignite.* 05 16, 2013. http://igniteshow.com/videos/ignite-tao-v4-may-162013-adam-light (accessed 07 07, 2013).

New product development. Wikipedia. http://en.wikipedia.org/wiki/Fuzzy_front_end#Fuzzy_Front_End (accessed 07 07, 2013).

Silverman, Rachel Emma. "Doodling for Dollars." *Wall Street Journal.* http://online.wsj.com/article/SB10001424052702303978104577362402264009714.html (accessed 07 07, 2013).

"Wikipedia entry User story." http://en.wikipedia.org/wiki/User_story (accessed 07 07, 2013).