Agile Scrum - The 30 Day Sprint and The Daily Scrum Meeting

Many of us have experienced projects that drag on much longer than expected and cost more than planned. Companies looking to improve their software development processes are now exploring how Agile can help their Enterprise more reliably deliver software quickly, iteratively and with a feature set that hits that mark. While Agile has different "flavors", Scrum is one process for implementing Agile. This newsletter is one in a series of newsletters that will discuss the Agile Scrum process and will end with variants of Scrum that can be used to aid in improving your software releases. Here are the prior newsletters in this series:

- **Feb 2008: Agile Scrum - An Overview**
  http://www.pragmaticsw.com/newsletters/Newsletter_2008_02_SP.htm
- **Mar 2008: Agile Scrum - Team Composition**
  http://www.pragmaticsw.com/newsletters/Newsletter_2008_03_SP.htm
- **Apr 2008: Agile Scrum - Understanding Scrum Rules**
  http://www.pragmaticsw.com/newsletters/Newsletter_2008_04_SP.htm
- **May 2008: Agile Scrum - Scrum Kickoff and Product Backlog**
  http://www.pragmaticsw.com/newsletters/Newsletter_2008_05_SP.htm

30-Day Sprints

Agile differs from standard Waterfall development in that development has a smaller timeframe with a smaller feature set. Agile Scrum implements releases every 30 days (called 30 day sprints). In the purest implementation of Scrum, the 30 days is 30 calendar days. We have found that 30 working days works best.

Upon completing a sprint, you can move the software to production (if production-ready) or move into another 30 day sprint to implement additional features. At the conclusion of the PAD Planning Week, you should have a set of detailed requirements and estimates and the Scrum Master can create a project plan that contains each feature (represented by a work order) and the individual assignments. The items that appear in the project plan for the sprint are referred to the **Sprint Backlog**.

Each feature should have a priority and should be worked in priority order so that if the team falls behind, you can ensure the highest priority items make it into the sprint. Features not completed at the end of the sprint will be reprioritized for possible inclusion in the next sprint.

Best Practices for 30-Day Sprints

Create Test Cases before Coding Begins

As each feature is detailed, it is important to create a set of test cases before coding begins. Developers should always run all the established test cases before making the feature available to the Software Quality Engineer(s) for testing. This approach will drastically reduce defect count and significantly speed up the quality assurance testing.

Daily Code Builds

Each day, each team member should check their code into their source control system if their code is compilable (never check in code that is not compilable). If SQL script changes were made, these also need to be checked into the source control system. It is wise to utilize an automated build tool that can run at the end of day and will do a GET on all source code and SQL Script changes that were made. It can compile the code into DLLs and run the SQL scripts to upgrade your database. This allows you to have a new build on your quality assurance server daily so that your Software Quality Engineer(s) can test new features and

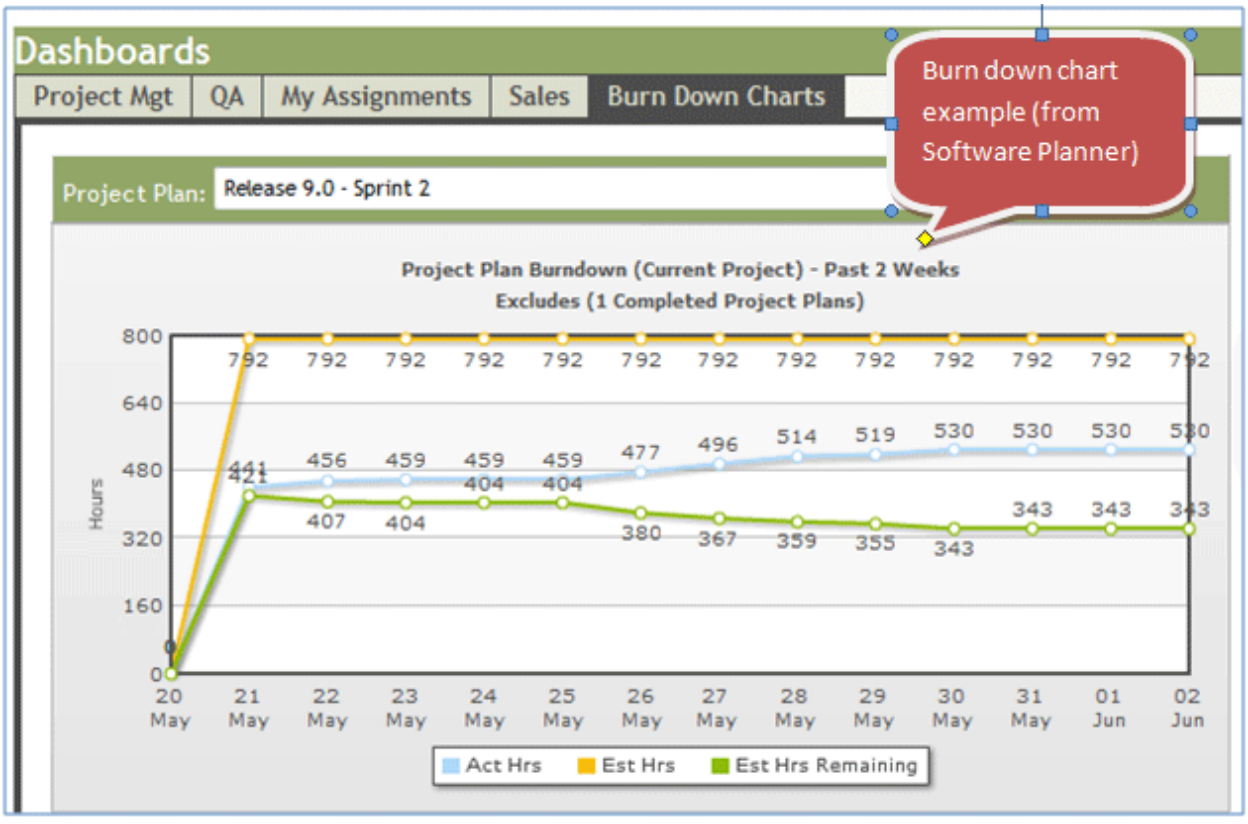
run regression when needed. There are a number of tools available for automatic builds, a popular one is called **Automated Build Studio** by Automated QA (see it at http://www.automatedqa.com)

Code Inspection

It is important to know that code should not be considered complete until it is fully coded, fully unit tested, all established test cases have been run, code has been refactored (if needed), and technical documentation is written (when needed). Upon indication that code is complete for a feature, the team should do a code inspection by reviewing the code in the source control system related to the feature. The code review should look for standards adherence, identification of logic errors or performance problems, and reusability. If the code inspection finds failures, defects should be created and assigned -- allowing the developer to fix the issues before the code is tested by the Software Quality Engineer(s).

Daily Hours Entry

Each day, every team member should log the hours they worked on each task during that day. When logging time, it is important to enter the percentage complete or estimated remaining hours (this is the preferred method). By entering the estimated remaining hours for each task worked on, your team will know how many hours remain for all tasks and can determine if you are progressing on a pace to finish the sprint with all the desired features. From a metrics perspective, inspect burn down charts daily. A burn down chart is simply a chart that shows day-by-day the number of estimated hours, actual hours and estimated hours remaining. As the sprint progresses, you should see the estimated hours trend downwards and it should be on pace so all the committed work is accomplished in the sprint.



Daily Scrum Meeting

Each day, your team should hold a Daily Scrum Meeting. In the purest version of Scrum, this daily meeting is restricted to 15 minutes and each team member is asked 3 questions:

"What did you do yesterday?"
"What will you do today?"

"Are there any roadblocks or anything impeding your progress?"

In our experience, 15 minutes is not always enough time to have a good dialog and a meaningful meeting. Many days we will complete this in 15 minutes, but most often it takes about 30 to 45 minutes. To speed this up, we have each team member post a summary of what they did yesterday and a summary of what they plan to do tomorrow in a daily discussion forum that is automatically distributed to all team members (see example below). This allows you to spend the Daily Scrum meeting talking about roadblocks, design decisions, and impediments to progress.

What's Next?
Upcoming newsletters will discuss the following topics:

- Agile Scrum - Reporting and Metrics
- Agile Scrum - Retrospectives
- Agile Scrum - Site specific variants of Scrum

Helpful Templates

Below are some helpful templates to aid you in developing software solutions on-time and on-budget:

- **Pragmatic Agile Development -http://www.pragmaticsw.com/PADOverview.pdf**
- **Software Development /QA Templates -http://www.pragmaticsw.com/Templates.asp**
- **Software Planner - http://www.SoftwarePlanner.com**
- **Agile Training - http://www.PragmaticSW.com/Services.asp**

About the Author

Steve Miller is the President of Pragmatic Software (http://www.PragmaticSW.com). With over 23 years of experience, Steve has extensive knowledge in project management, software architecture and test design. Steve publishes a monthly newsletter for companies that design and develop software. You can read other newsletters at http://www.PragmaticSW.com/Newsletters.asp.