Measuring the Impact of Scrum on Product Development at Adobe Systems

Peter Green
Adobe Systems
peterg@adobe.com

Abstract

Over the past several years scrum has grown to become the most commonly used product development method at Adobe Systems. Large desktop software products like Premiere Pro and After Effects, Platform tools like Adobe AIR, and Software as a Service products like Acrobat Connect and Omniture SiteCatalyst are using scrum to become more effective at delivering the right solutions to customers with higher quality. This paper discusses the methods Adobe Systems is putting in place to measure the impact scrum has had on these teams.

1. Introduction

Moving to a new software development process often comes with the promise of highly productive teams, better software quality, and products that are better fit for customers. While there is ample evidence that agile processes such as Scrum and XP result in such benefits [2], there can be multiple challenges in attempting to measure and analyze the results within a working business. Agile methods emphasize individual conversations over tracked documents, self-management, rapid response to change, and working software as the primary metric [3]. Given this environment, it is often difficult to gather meaningful metrics describing the before state versus the current state, and hence to use data to quantify the changes the intuitive sense of improvement that teams may feel is in place using the new methods.

In 2005, two teams at Adobe Systems began using scrum [4]. Today, over 1/3 of the company's developers are using scrum, and it is the most common development methodology in use at Adobe. As the process has grown at the company, we have attempted to put a few measurements in place to back up the sense that the process is working.

There are three measurement areas that we predicted would provide good indicators of the effect of scrum on a specific project: 1) subjective ratings by scrum practitioners of how the process has affected them and how well their team is doing the process, gathered through a semi-annual survey, 2) defect data, such as defect counts, deferral rates, and cycle-to-cycle defect curve information, gathered from defect management systems, and 3) Net Promoter Scores [1], gathered from prerelease customer surveys. While we would prefer to use customer satisfaction ratings from the products in the field, such data is not available for the most recent releases.

In attempting to measure these three areas, we found that there was a lack of data in at least one of the three areas for most teams. This made it very difficult to correlate across the teams to identify key drivers for successful scrum adoption. Additionally, only moderate correlation between specific scrum practice adherence and high customer satisfaction/low defect rates may indicate that there are important drivers of success that don't fall into one of the three measurement areas. The paper will discuss the data available in the three measurement areas, what steps we are taking to remedy the lack of data in key areas for future adopters of scrum at Adobe, and what conclusions we are able to draw based on the more limited data set we have available today.

1.1 Sample set

Table 1.1 below demonstrates the challenge faced in comparing the three measurement areas across multiple product teams. For the purposes of this paper, the team names are redacted and replaced with generic terms to protect the teams' identity, but the generic names are used consistently throughout the paper (team 1 is always team 1) We have 26 teams that have data in at least one of the three categories. Of those 26, 25 have data from the scrum survey,

1530-1605/11 $26.00 © 2011 IEEE

nine have defect data, and three have customer satisfaction data.

2. Scrum survey

Within the agile community, there is much discussion of what it means to actually be "doing scrum". There have been a few attempts to create a test to determine how well teams have adhered to the principles of scrum (see Nokia Test/Scrum But Test [5]), but we have found these tests to be inadequate for our needs.

In order to better ascertain how well teams are implementing the process, we chose to create a more comprehensive survey than what we found in use in the community at the time. While other industry thought leaders have since produced similar surveys [6], we have decided to continue to use the Adobe Scrum Survey rather than change to the standard surveys and lose the ability to track improvement in specific areas. The Adobe Scrum Survey is administered semi-annually. The most recent version of the survey had 220 respondents from 25 different development teams.

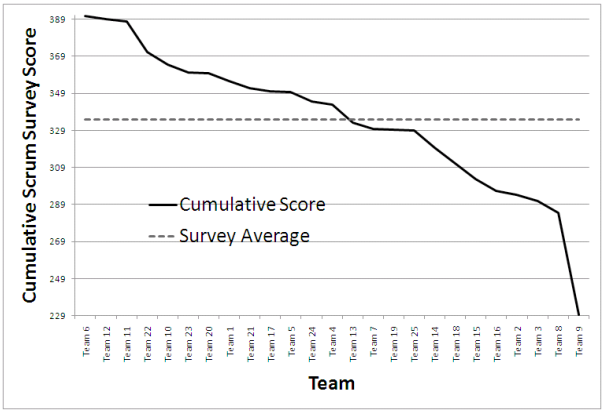
The first section of the survey measures respondents' opinion of scrum's effectiveness as a process for their team (Scrum Effectiveness section). The second section is a scrum scorecard where respondents are asked to rate several areas of standard scrum process rules to ascertain how well the teams are implementing scrum (Scrum Scorecard section).

The survey consists of statements that respondents are asked to rate their agreement with on a scale of one to ten. There are a total of 50 statements across the various categories, yielding a maximum score of 500. The highest cumulative actual team score in the most recent survey was from Team 6, with a score of 391, representing an average agreement of 7.818 out of 10 across all questions. We can use Team 6's score to "grade on a curve", applying a coefficient to the other team's scores that normalize them to a scale where the Team 6 is equal to 100%. Table 2.0 shows the scores of all teams with these data points.

Table 2.0: Cumulative, average, and normalized scores for Adobe Scrum Survey:

Team	Score	Average	Normalized
Team 6	390.9	7.818	100%
Team 12	388.9	7.778	99%
Team 11	387.7	7.753	99%
Team 22	371.4	7.428	95%
Team 10	364.7	7.293	93%
Team 23	360.5	7.209	92%
Team 20	359.7	7.193	92%
Team 1	355.8	7.115	91%
Team 21	351.8	7.036	90%
Team 17	350.2	7.003	90%
Team 5	349.5	6.989	89%
Team 24	344.7	6.894	88%
Team 4	343.0	6.859	88%
Team 13	333.1	6.662	85%
Team 7	329.9	6.597	84%
Team 19	329.3	6.586	84%
Team 25	329.0	6.580	84%
Team 14	319.9	6.396	82%
Team 18	311.0	6.221	80%
Team 15	302.7	6.053	77%
Team 16	296.3	5.925	76%
Team 2	294.2	5.884	75%
Team 3	290.7	5.814	74%
Team 8	284.6	5.692	73%
Team 9	229.7	4.593	59%

Chart 2.0: Cumulative Survey Scores per team, ranked highest to lowest and graphed:



In analyzing the data from the survey, we found it beneficial to examine average scores per question and area, as a baseline for how the company is doing as a whole, as well as to examine teams whose scores were among the best at the company, as a way to show individual teams what was a potential target score at the company. Additionally, we can use these two categories (average and high) to correlate specific metrics, either correlating specific survey questions or general survey categories between

average teams and high teams, as well as correlating survey data with data in the defect and customer satisfaction areas

To determine which scores to use in the "high" category, we took the top 5 teams in the cumulative survey scores and provide their answers as a guide post for what was possible. Since the top 5 represent 25% of all responding teams, they are referred to in the tables at Top Quartile, sometimes abbreviated as Top Q.

2.1. Scrum effectiveness section

In the Scrum Effectiveness section of the survey, respondents are asked to rate how strongly they agree with several aspirational statements related to project goals on a scale of 0-10. In all categories, teams saw some improvements after moving to scrum. When we narrow the results to those teams that scored in the top 25% on the Scrum Scorecard section, we see much stronger effectiveness:

	Avg.	Top Q
The quality of our software has improved since implementing scrum.	6.44	7.57
The communication on our team has improved since implementing scrum.	7.03	8.35
We deliver a better product to our customers since implementing scrum.	6.37	7.81
I have a better work/life balance since our team implemented scrum.	5.11	5.96
My job allows me to put my skills and talents to good use.	6.31	7.44

In addition to the aspirational statements, respondents were asked two general impression questions. The first is a Net Promoter Score question for the process (How likely are you to recommend scrum to friends or colleagues?), the second is a question borrowed from Yahoo scrum surveys: "If the question were solely up to you, would you continue using scrum on your project?"

	Avg.	Top Quartile
Net Promoter Score	-3	44
Would use if up to you (% Yes)?	80%	100%

Note the high correlation between satisfaction with the results of scrum and doing the scrum process well. It is unclear from the data whether the Scorecard data has a causal relationship upon the Effectiveness data, or whether the causation is reversed or simply similarly symptomatic of other factors such as effective leadership or strong team members. However, from the data we can conclude that Adobe teams that effectively do scrum have the viewpoint that their quality, communication, product fitness, and work life balance are significantly improved as a result.

2.2. Scrum scorecard section

In the Scrum Scorecard section of the survey, respondents are asked to rate how strongly they agree with several aspirational statements related to effective implantation of scrum on a scale of 0-10. This section serves both to provide overall trends and to provide feedback to individual teams that they can utilize in their retrospectives. This section is divided into major areas of the process: Discipline & Improvement, Product Owner, Scrum Master, Scrum Team, Sprints, Agile Best Practices, and Agile Values. Let us examine these by category. In the results below, we show the Adobe Average score, the score of those respondents whose cumulative scores were in the top quartile overall, and the delta between the two. This allows us to identify what key areas good scrum teams do really well at when compared to average scrum teams.

The first sub-area of the scorecard asks respondents to rate two high level statements. The first statement, simply asking if respondents feel that their teams follow the rules, is interesting, as individuals will have a wide variance in what their understanding of the rules might be. However, it does provide valuable information since team trust is significantly hampered when team members don't feel that they follow their own rules, regardless of their understanding of what those rules might be.

The second statement is key to the survey as one of the main pillars of scrum is continuous improvement. If teams see low scores on this question, it would indicate a significant gap in how effectively they were doing scrum.

Statement	Avg	Top Q	Δ
Discipline, Improvement Category	**6.56**	**7.90**	**1.34**
Our team adheres to the rules of scrum	6.24	7.40	1.16
In general, our scrum implementation is improving over time	6.88	8.40	1.52

The Product Owner section of the Scorecard highlights an area in need of improvement at the company. Note that even teams in the top 25% have lower scores in this section than most other sections, and that with the exception of the velocity question, have significantly lower deltas than in other sections:

Statement	Avg	Top Q	Δ
Product Owner Category	**6.00**	**6.84**	**0.84**
There is no question as to who the product owner is on my team.	6.96	7.32	0.36
The product owner inspires the team with their vision and goals for our product and the current release.	5.87	6.71	0.84
The product owner keeps the product backlog well groomed, prioritized, and organized	5.59	6.48	0.89
The product owner clearly communicates the priority and goals of items in the product backlog during sprint planning	6.14	7.21	1.07
The product owner is readily available to answer questions as they arise during the sprint.	6.7	6.88	0.18
We use team velocity to determine the probable scope or release date of our current release	4.74	6.42	1.68

The Scrum Master section of the Scrum Scorecard shows higher scores than other areas, indicating that teams are generally satisfied with how the Scrum Master role is being fulfilled. In this category there were smaller deltas between the average teams and the best teams. Unlike the Product Owner section, the scores were universally good as opposed to universally poor:

Statement	Avg	Top Q	Δ
Scrum Master	**7.06**	**8.02**	**0.96**
Our Scrum Master has a good understanding of the theory and rules of scrum.	7.48	8.24	0.76
Our Scrum Master helps us be more productive as a team	7.11	8.13	1.02
Our Scrum Master keeps the daily scrum meeting brief and on topic	7.03	8.05	1.02
Our Scrum Master helps us stay on track to meet our commitments for the sprint	7.09	7.98	0.89
Our Scrum Master makes sure that impediments are raised and resolved quickly	7.3	7.92	0.62
Our Scrum Master shields the team from external influences during the sprint	6.41	7.68	1.27
Our Scrum Master leads the team to improve our processes and teamwork during meetings like the daily scrum and sprint retrospectives	7.02	8.18	1.16

The Scrum Team section is also among the highest scores of any of the Scorecard sections. Adobe has traditionally had strong teams in place, so this is not a surprising result. Notice the significant delta between average and top teams in the area of team ownership of the sprint burndown artifact.

Statement	Avg	Top Q	Δ
Scrum Team	**6.98**	**8.01**	**1.03**
Our scrum team is cross functional, or in other words, includes the various skill sets (UX, Dev, QE, Etc.) required to deliver a potentially shippable increment each sprint	7.34	6.88	-0.46
Our scrum team decides how much work to commit to in a single sprint	7.84	8.90	1.06
Our scrum team keeps the sprint backlog up to date and accurate throughout the sprint	6.43	6.92	0.49
The sprint burndown chart is a valuable tool for our scrum team to help us know if we're on track for meeting our sprint commitments	5.85	7.97	2.12
Our scrum team's daily scrum meeting is a valuable use of my time	6.93	8.38	1.45
Our scrum team regularly makes team decisions without needing to escalate to management.	7.04	8.41	1.37
Our managers trust the scrum team to do the right thing in the sprint	6.99	8.16	1.17
Members of our scrum team work together to solve problems rather than trying to solve them on their own	7.39	8.46	1.07

The Sprints sub-area has relatively high scores as well. A few interesting standouts are the strict adherence to sprint boundaries and a clear and respected definition of done. Any project can choose to fix either the date or the scope but not both (on short iterations, adding resources has no benefit or a negative impact) [7]. Scrum literature recommends dropping scope so that the scrum teams can establish a rhythm [8]. Notice that the successful scrum teams are very disciplined in this area compared to the average scrum team, indicating that a willingness to slip a date may be indicative of a larger leadership issue on the team. Also, notice the large deltas in the two Definition of Done statements, indicating that teams that clarify and adhere to a definition of done for the items completed in a sprint is a strong factor in the overall success of scrum for those teams:

Statement	Avg	Top Q	Δ
Sprints	**6.97**	**8.22**	**1.25**
Our team uses sprints that are 30 calendar days or less.	9.07	10.00	0.93
At sprint planning meetings, all of our team members actively participate in the discussion on how to decompose product backlog items into the sprint backlog	7.49	8.60	1.11
Our team never slips the date of a sprint review in order to finish a backlog item	7.27	8.80	1.53
Our team has agreed to a "definition of done" for everything that is demonstrated at a sprint review	6.64	8.13	1.49
Items that don't meet the definition of done are always put back on the product backlog and never demonstrated at the sprint review.	6.04	7.06	1.02
Our definition of done challenges us to improve the quality of our software	6.59	8.35	1.76
The primary activity of our sprint review meeting is the demonstration of working software	6.68	7.66	0.98
Sprint retrospective meetings result in specific changes to improve things in the next sprint	6.6	7.95	1.35
Recommended changes from the retrospective are regularly implemented	6.33	7.42	1.09

The Agile Best Practices sub-area attempts to narrow our results to three types of team: those using scrum alone, those using scrum with other agile practices, or those using agile practices but not scrum. The teams that scored high on the remainder of the section were significantly better at two areas than the average team: Planning Poker and Story Points. A very surprising result is that the best scrum teams scored significantly lower than the average scrum teams in the areas of Continuous Integration and automated test coverage, while still outscoring the average teams when rating their own product quality. This is the only area of the survey where we found negative deltas like these:

Statement	Avg	Top Q	Δ
Agile Best Practices	**6.08**	**6.92**	**0.85**
Our product backlog is made up primarily of user stories	5.67	7.88	2.21
Stories/Features that are approaching implementation are decomposed into several very small vertical slices.	5.72	7.53	1.81
Items in our product backlog are estimated in relative size using a unit such as Story Points	6.63	8.74	2.11
Our team conducts a short release planning phase at the beginning of a release to agree on the vision, goals, initial product backlog, release estimates, etc.	5.7	6.94	1.24
Our team uses a wide band Delphi estimation technique (such as planning poker) for release level estimates	5.67	7.70	2.03
Our team uses early defect removal techniques such as peer reviews, pair programming, or Test Driven Development.	6.23	7.28	1.05
Our team has specific quality improvement goals with metrics that are reviewed regularly	5.54	5.95	0.40
Our team runs automated tests on every build that validate most of the functionality in the product	6.59	3.86	-2.73
Our team uses continuous integration	6.95	6.44	-0.51

2.3. Scrum survey conclusions

The scrum survey's primary goal is to help teams measure their own process and effectiveness. Several teams take advantage of the data by comparing their scores to previous iterations of the survey and comparing their team's score to Adobe averages and top teams. These teams take advantage of Adobe's internal scrum coaching resources to help them analyze the data, provide insight into possible changes to their process, and track progress over time.

In addition to its value to individual teams, we use the data to examine larger trends at the company. We track the Net Promoter Score from iteration to iteration of the survey to assess overall effectiveness of scrum adoption by Adobe teams. We also use the individual categories as an indicator of areas that are in need of focus. For example, the low scores in the product owner category led to the creation of a

training course tailored for product owners, a blog post by our internal trainers on the responsibilities of the product owner, which was followed up by a good discussion on Adobe's internal agile email distribution list.

Finally, we can use the data to pinpoint areas that good scrum teams are doing a lot better than average scrum teams as potential areas of focus. The 10 statements with the highest deltas are a great list of items for consideration in retrospectives and coaching visits as practices to consider doing or doing better. Many are not officially part of the scrum framework, but are other agile techniques that work well within the scrum framework. This may indicate that the teams in question have moved further along in the agile adoption curve, starting with scrum and adding other agile practices through inspection and adaptation, or that they began their agile adoption with other agile practices in place and when adding scrum as a framework around these practices, were able to move quickly to a more effective overall approach.

Ten Survey Statements with the highest delta between average and high performing scrum teams at Adobe Systems:

Delta	Statement
2.21	Our product backlog is made up primarily of user stories
2.12	The sprint burndown chart is a valuable tool for our scrum team to help us know if we're on track for meeting our sprint commitments
2.11	Items in our product backlog are estimated in relative size using a unit such as Story Points
2.03	Our team uses a wide band Delphi estimation technique (such as planning poker) for release level estimates
1.81	Stories/Features that are approaching implementation are decomposed into several very small vertical slices.
1.76	Our definition of done challenges us to improve the quality of our software
1.68	We use team velocity to determine the probable scope or release date of our current release
1.53	Our team never slips the date of a sprint review in order to finish a backlog item
1.52	In general, our scrum implementation is improving over time
1.49	Our team has agreed to a "definition of done" for everything that is demonstrated at a sprint review

3. Defect data

Defect data at Adobe is collected from several defect tracking systems. Unfortunately, 2/3 of the teams that responded to the survey do not have data in these systems for one of three reasons. First, some were using a defect tracking system that did not feed into the Adobe corporate metrics portal. Second, some teams never setup the release dates in the system that allow for cycle-to-cycle comparisons pre and post scrum. Finally, some teams were building new "1.0" products with scrum, and were newly formed teams with no historical data to compare against. In addition to these challenges in drawing correlation, the project timelines of the teams measured in the survey range in length from 2 weeks to 18 months, so "Cycle" has a very different meaning between these teams.

The limited data renders any conclusions drawn by correlating the survey data with the defect data of dubious value. However, there are some interesting results that we can observe from those teams that are tracking defect data in these systems.

Of the 25 teams that took the survey, eight had data in the tracking system. In addition, one team that adopted scrum had data in the system but did not take the scrum survey.

For the purposes of this paper, we will describe three defect metrics used by most Adobe teams.

The first is total defects found in system test, which we use to measure the quality of code being delivered to the system test process. As early defect removal techniques such as Test Driven Development, pair programming, and peer reviews are adopted, we expect the overall defects found during system test to drop as more defects are found early.

The second metric is total defects deferred, which we have found is highly correlated to both the number of defects found in system test and the number of defects discovered by customers in the field. Therefore, the number of deferred defects is a good indicator of the level of quality experienced by customers.

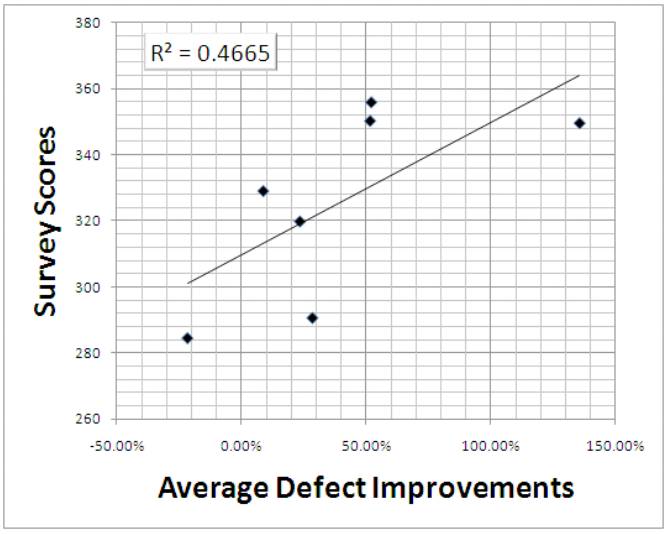
The third defect metric is the number of open defects on any given day of the release. This metric is visualized using a chart showing the open defect curve for the current release compared to the same relative date in the release cycle before implementing scrum. This graphic is primarily useful for teams that have long release cycles, and tends to demonstrate how close to "potentially shippable software" the team actually is at any given moment. A traditional waterfall type project with a model where defects are

fixed as late as possible in the release will have a steep defect curve as the project reaches its code complete or beta milestone, with a steep curve down towards zero. Using agile methodologies such as scrum, which have the goal of delivering high quality software on a frequent heartbeat, we should see a much gentler curve undulating up and down due to the short iterations during which the team is attempting to reach a state of potential releasability. We can measure peak-to-peak total open defects on these charts as a way to show the amount of improvement from cycle-to-cycle.

3.1. Defect rate improvements

All nine teams tracking cycle-to-cycle defect improvements are on relatively long release cycles (12-18 months), and use scrum to release to prerelease testers under Non-Disclosure Agreements at the end of each sprint. Seven of the nine teams saw improvements in all areas of defect rates after implementing scrum. One team saw improvements in two of the three areas, with a decline in the third. The one team that saw declines in all three areas, "Team 8", also had very low scores on the scrum survey (second to lowest). When taking the average improvement across the three defect areas, and comparing it to the scrum survey scores, we saw a relatively strong correlation:

Correlation between Defect Rate Improvements and Scrum Survey Cumulative scores.



We also examined several statements from the Scrum Scorecard section of the survey that we would expect would correlate to improved defect rates, and measured the actual correlation between high scores in these areas and greater improvement in defect rates. We expected to find that specific engineering practices would be the strongly correlated. What we actually found was the following in order of highest correlation (using R^2) to lowest:

R^2 Value	
0.52	The primary activity of our sprint review meeting is the demonstration of working software
0.44	Our team never slips the date of a sprint review in order to finish a backlog item
0.38	Our team has specific quality improvement goals with metrics that are reviewed regularly
0.24	Our team has agreed to a "definition of done" for everything that is demonstrated at a sprint review
0.2	Items that don't meet the definition of done are always put back on the product backlog and never demonstrated at the sprint review.
0.03	Our team runs automated tests on every build that validate most of the functionality in the product
-0.08	Our team uses continuous integration
-0.63	Our team uses early defect removal techniques such as peer reviews, pair programming, or Test Driven Development.

We believe that the highly correlated items on this list are valid findings. However, as the correlation drops, and with personal experience working with these specific teams being measured, we believe the small sample size does not permit us to draw strong conclusions about the importance of engineering techniques and defect improvement. We are aware of other teams that are not represented by this data that would skew the correlations in the areas of agile engineering techniques higher than those in this sample set. In our recommendations for future scrum adopters at Adobe, this will be an area in which we hope to improve so that we have better statistical samples sizes in the future.

The table below lists the nine teams tracking defect improvements before and after scrum implementation, in order of largest average improvement to lowest. For the Defects Found in Test metric, we used the following equation, where I represents the improvement percentage, D_1 represents the total number of defects found in test during the pre-scrum cycle, and D_2 represents the total number of defects found in test during the post-scrum cycle. We used similar equations for calculating percentage improvements in total deferred defects and peak open defects for the cycle.

$$I = 1 - (D_2 \div \ _1)$$

Percentage improvement in defect metrics categories after implementing scrum:

Team	Defects Found in Test	Defects Deferred	Peak Open Defects	Average Improvement
Team 5	97.00%	122.00%	187.00%	135.33%
Team 17	52.00%	52.80%	51.00%	51.93%
Team 1	38.40%	36.00%	80.00%	51.47%
Team 14	25.00%	25.00%	35.00%	28.33%
Team 3	16.00%	7.00%	47.00%	23.33%
Team 26	12.00%	41.70%	14.00%	22.57%
Team 8	5.00%	37.00%	-16.00%	8.67%
Team 25	-13.00%	-43.00%	-9.00%	-21.67%

Note that these metrics will tend to favor teams who were not doing particularly well in any of the measurement categories prior to implementing scrum. In at least one case, Team 25, the team already had very low defect numbers before implementing scrum, and so while their average improvement is not as extreme as some of the other teams, this does not necessarily indicate an area needing focus.

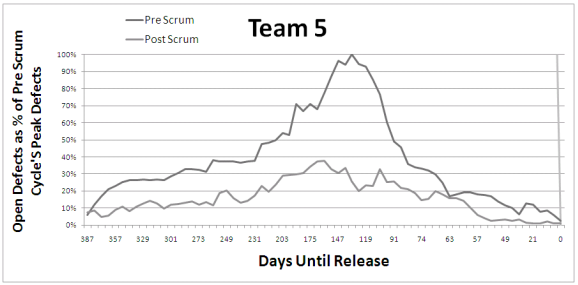
3.2. Cycle-to-cycle comparative defect graphs

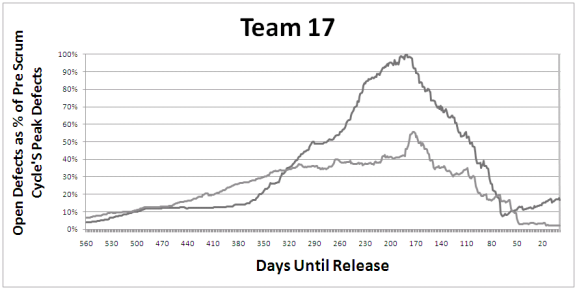
Below are the cycle-to-cycle defect graphs showing open defects at any given point in the given cycle. For each graph, the X axis represents total number of days prior to the final release build, and the Y axis represents total open defects on that date. For the purposes of this paper, we are representing the Total Defect rates as a percentage of the peak rates from the Pre-Scrum cycle. For example, if in the cycle before implementing scrum, a team's peak open defect count was 200, and in the cycle after implementing scrum, the peak open defect count was 100, these would be normalized so that the series line for Pre-Scrum showed a peak of 100% and the series line for Post-Scrum showed a peak of 50%.

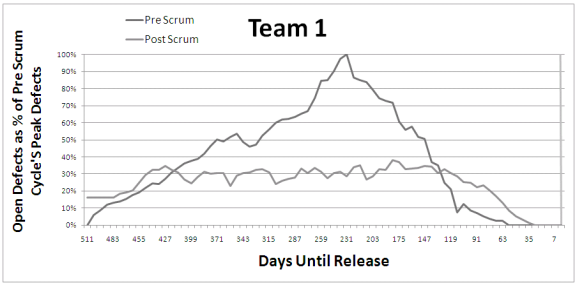
Notice that none of the teams are maintaining the strictest definition of done, where the project is bouncing off of zero defects at each sprint boundary. Also notice that in nearly all cases, while the defect curve is less extreme after moving to scrum, the curve still suggests that the teams have not fully moved away from a waterfall cycle mentality, where an entire phase of the project is devoted to removing defects and not adding new functionality.

One contributing factor to this shortcoming is that several teams that have adopted scrum are part of larger suite products with their own SDLC requirements. Due to these requirements, scrum teams are often required to adhere to a "Feature Complete" milestone weeks or months before the final release, where new features are not permitted to be developed without a hefty Engineering Change Order approval process. The teams are aware that they will need to stop developing features on that specific date, and that they will be allowed to remove defects after that date, and so you can see the natural result – quality drops as they attempt to squeeze in as much functionality prior to the deadline as possible, knowing they can "clean up the mess" after the milestone passes. Thus, adhering to the "old rules" of waterfall robs scrum of its efficacy as predicted by Eliyahu Goldratt in Beyond the Goal [9]. This is an area that the company is working hard to address in future cycles.

The graphs for the three teams with the most significant improvements are presented to illustrate what Adobe's best scrum teams are accomplishing:







3.3. Defect conclusions

Though there is not as large a sample set of data to work with as in the survey area, there is still compelling evidence that a move to scrum has had a net positive impact on the total number of defects found in system test, the total number of defects deferred, and the peak defects per cycle, and that there is a rough correlation between defect improvements and high scores on the subjective scrum survey. Additionally, the specific scrum constraints of delivering working software at the end of each sprint, and working within strict time boxes, along with the management best practice of setting high level quality goals for a team, tend to be positively correlated to improved defect rates.

4. Customer satisfaction

While we had only eight teams with both Scrum Survey data and Defect Improvement data, we have even fewer with good Customer Satisfaction data, with three teams that have adopted scrum measuring Net Promoter Scores from customers for both the current cycle and the cycle before adopting scrum. Two of these three have data in all three areas, including scrum survey data and defect data, and one has NPS scores and defect data but not scrum survey data. With such a limited data set, we can draw few conclusions from the data. Inclusion in this paper is to serve as a snapshot of the current state of progress towards our goal of being able to measure and correlate all three areas.

Improvements in Net Promoter Score after adopting scrum:

Team	Improvement in Net Promoter Score
Team 1	+8 points
Team 17	+16 points
Team 26	+14 points

5. Conclusions and future use of this data

Through the measurement of subjective data from teams adopting scrum, the measurement of comparative defect data rates between pre-scrum adoption and post-scrum adoption product cycles, and the measurement of customer satisfaction through Net Promoter Scores, we have been able to draw a few key conclusions about the efficacy of scrum as a framework at Adobe Systems.

There is a large spectrum of how effectively scrum has been adopted by teams across Adobe, from teams that are rating close to an average score of eight out of possible 10 on survey questions, all the way down to teams rating below five. By their own admission, some Adobe teams have adopted scrum in name only, not actually changing their processes and environments to support the key practices, artifacts, roles, and environment described by scrum's original inventors and current thought leaders. On the other side of the spectrum, there are teams that are making great progress by applying the key tenets of scrum on their projects. We plan to use these successful scrum implementers as examples to other teams looking to adopt scrum, asking key leaders from these teams to help mentor leaders whose teams are just adopting the process. At the same time, we are working with teams that have lower scores to identify if these lower scores are preventing them or slowing them from delivering better products to their customers, and providing input and coaching to improve on key areas.

In general, Adobe employees using scrum like the process, with 80% of all adopters saying that they would continue using scrum if the decision was solely up to them. For teams that are using it effectively, the individuals like the process even more, with100% of those that are on the Top Quartile saying they would continue to use it if the decision were up to them.

Survey data is a useful tool for teams to use in retrospection activities. It also provides data on overall adoption trends at the company and areas in need of general improvement and focus for the scrum adoption community at Adobe. The semi-annual delivery of the survey allows teams to measure trends in areas of focus to ensure they are making progress.

Data indicates that teams that do well at adopting scrum generally experience an improvement in three key areas of defect measurement. First, the total number of defects found in system test tends to go down for teams successfully adopting the process. A drop in defects found in system test, with other factors such as team size and cycle length remaining constant, indicates that quality processes have moved up stream from the quality assurance team to the software engineering team, where it is more efficient [10]. Next, the total defects deferred from a release had a strong correlation with both the total number of defects found in system test, and with how successfully the team had implemented scrum. We have found that deferred defects tend to be very strongly correlated to defects found by customers in

the field, so this particular metric is a good predictor of the perceived quality of the software we release. Finally, teams successfully implementing scrum tend to see a significant reduction in the peak number of open defects at any given point in the cycle. This has a few key advantages. First, it tends to mean decreased risk, as defect removal is one of the riskiest areas of software development, and the large mountain of defects faced by traditional software teams represents a mountain of unknowns. . Next, finding and removing defects earlier and more cost effectively leads to more productive teams as team members are free to develop new functionality without the added delays of having to work around broken areas of the code base. Finally, the much gentler slope of the defect curve tends to mean fewer deferrals will need to be made in order to ship the software, as one common way to reduce the large queue of defects is to defer defects that we would not normally be comfortable deferring just to make the date.

Finally, a few teams have started measuring cycle-to-cycle Net Promoter Scores. We think that this is one of the purest measures of success as it is right at the source of our revenue. While there is very limited data in this area, all three teams measuring NPS saw a significant increase in their scores after adopting scrum.

Adobe's adoption of scrum has been a largely grass roots effort. There have been no executive level mandates to use the framework. Adobe has funded one full time trainer/coach, supporting the spreading of the scrum framework for teams that are interested in taking advantage of it.

Adobe has grown rapidly through several acquisitions throughout its history. In the experience of the author, Adobe tends to take a hands-off approach when acquiring these companies, believing that imposing a single process or culture will decrease the newly acquired teams' ability to deliver the solutions that made them valuable targets in the first place. This approach has many benefits that the company believes outweigh the drawbacks, but there are drawbacks. One such drawback is that different groups use vastly different development approaches with different measurements, making it difficult to draw conclusions across teams.

As scrum adoption continues to grow from its current early adopter stage to the early majority stage, we will need to have more data to describe and indicate what is most important in implementing the framework as we move from influencing visionaries to pragmatists [11]. Our initial efforts in this area have yielded some promising indicators, but with limited statistic relevance due to lack of more data. In the future, we plan to apply the approach of measuring subjective impressions of the process, defect rate improvements, and customer satisfaction data as a standard practice for new adopters. As our training and coaching of new scrum team matures, we are seeing the need of spending more time with teams to get them successfully launched. Part of this increased involvement with new teams will include strong recommendations that they measure these three areas in order to determine the efficacy of their adoption.

[1] Reichheld, Frederick F., "The One Number You Need to Grow", Harvard Business Review, Dec2003, Vol. 81 Issue 12, p46-54.

[2] Larman, Craig, Agile and Iterative Management: A Manager's Guide, Addison-Wesley Professional, p63-100, August 21, 2003.

[3] Beck, Kent, Beedle, Mike, van Bennekum, Arie, Cockburn, Alistair, Cunningham, Ward, Fowler, Martin, Grenning, James, Highsmith, Jim, Hunt, Andrew, Jeffries, Ron, Kern, Jon, Marick, Brian, Martin, Robert C., Mellor, Steve, Schwaber, Ken, Sutherland, Jeff, and Thomas, Dave, "Manifesto for Agile Software Development", February 13, 2001, web, <http://www.agilemanifest.org>, August 1, 2010.

[4] Green, Peter, and Smith, Peter, "Scrum @ Adobe", Scrum Alliance, March 17, 2009, web, <http://www.scrumalliance.org/resources/612>.

[5] Vode, Bas, and Sutherland, Jeff, "Scrum But… Test", June 28, 2009, web, <http://antoine.vernois.net/scrumbut/>.

[6] Cohn, Mike, and Ruben, Kenny, Comparative Agility, June 2007, web, <http://comparativeagility.com/>.

[7] Brooks, Fredrick P., "The Mythical Man Month: Essays on Software Engineering, 2nd Edition", Addison Welsley, August 12, 1995, p16-21

[8] Scwhaber, Ken, and Sutherland, Jeff, "The Scrum Guide", February 2010, web, <http://www.Scrum.org>, p10.

[9] Goldratt, Eliyahu M., Beyond the Goal, Your Coach in a Box, September 1, 2005, audio.

[10] Kaner, Cem, James Bach, and Bret Pettichord, "Lessons Learned in Software Testing: A Context-Driven Approach", Wiley. 2001 p4

[11] Moore, Geoffrey A., Crossing the Chasm, Harper Business Essentials, 1991, p42-46.