Scrum Metrics for Hyperproductive Teams: How They Fly like Fighter Aircraft

Scott Downey
RapidScrum LLC
scott@rapidscrum.com

Jeff Sutherland, Ph.D.
Scrum Inc.
jeff@scruminc.com

Abstract

Scrum teams use lightweight metrics like story points, the burndown chart, and team velocity. Recent work with hyperproductive teams shows they are like modern jet fighters.. They have two engines that produce velocity--alignment of the team and team spirit. A fighter aircraft is inherently unstable and must constantly correct to stay within the flight envelope—those parameters where the plane flies properly. Failing to watch Scrum metrics and trim the flight of the team can result in a hyperproductive crash into waterfall performance, typically 5-10 times slowerl.

Here we describe several metrics that can develop and sustain hyperproductive teams--velocity, work capacity, focus factor, percentage of found work, percentage of adopted work, original commitment, final commitment, commitment accuracy, estimate accuracy, and target value contribution increase. The unique contribution of this paper is to demonstate how a light touch and lightweight strategy can be used to compare teams with different story point reference scales.

1. Background

The average Scrum team delivered a 35% improvement in velocity at Yahoo [1] where teams properly coached delivered 300-400% improvements. The best Scrum Master at MySpace peaked at 1680% of initial velocity after 20 weeks and averaged 450% increase in velocity over 10 Sprints. Most teams are less successful. The highest performing team ever recorded was a Borland team audited by Bell Labs. They were 50 times faster than waterfall team industry average [2].

Currently, the best Scrum teams in the world average 750% gains over the velocity of waterfall teams with much higher quality, customer satisfaction, and developer experience. We have see this in the U.S. [3], Russia [4], the Netherlands and India [5], and from Software Productivity Research data on agile teams [6]. The problem addressed in this paper is that over 90% of Scrum teams never deliver this capability.

Agile teams have trouble measuring performance. Over 50% of teams do not know their velocity of production and have difficulty in finding ways to improve and measure this rate. Even when teams know their velocity, management cannot compare the performance of two teams.

Velocity on Agile teams is typically measured in story points. Teams pick a small reference story and assign it an arbitrary number of points. All other stories are estimated relative to the reference story using the wide-band delphi estimation technique commonly known as “planning poker” [7] Planning poker provides faster and more accurate estimates with less variance than hourly estimates but has the disadvantage that it is not usually comparable across teams. While function points are the preferred metric for productivity research they require more training, expertise, and time than is usually available to Agile teams [8]

The lack of adequate attention to metrics can prevent teams from systematically improving and reaching a hyperproductive state. This state is defined as at least 400% better than the average waterfall team. Today we have many documented hyperproductive teams running faster than this [4, 9-11].

2. Scrum is an Ecosystem

Experienced agile coaches recognize that Scrum is based on complex adaptive systems theory. It is not a methodology, process, or procedure. It is a framework based on enforcement of simple constraints that will cause a average team to self-organize into a hyper-productive state [12].

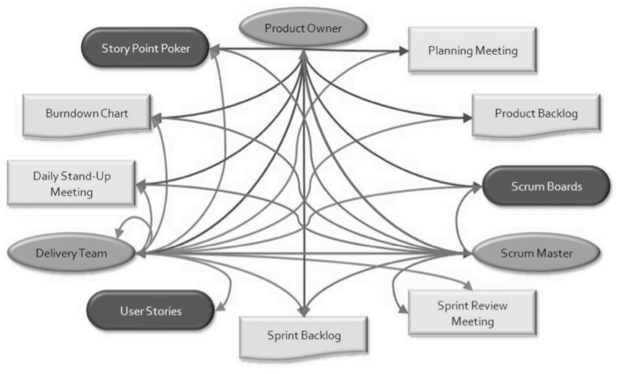


Figure 1. Scrum is an ecosystem.

Any system will settle into the lowest possible energy state. Consider the water in a toilet. It is without motion and flat. When you flush the toilet you introduce energy into the system and enforce constraints which cause the water to swirl into the same motion every time. As soon as the energy input stops, the water returns to a flat and motionless state.

The difference between the highest and lowest performing software development teams is 1:2000 [13]. This is more than two orders of magnitude greater than the difference between the best and worst developer on a project [14]. The average software development team is in a placid state where velocity is slow, quality is low, customers are unhappy, and management is upset. We want to introduce energy into the team and enforce constraints that systematically product high velocity, high quality, happy managers, and ecstatic customers.

Scrum meetings are designed to raise the communication level of a team in order to align their focus and facilitate team spirit. This introduces an energy flow into the system which is constrained by the ordering of the product backlog, the required ready state of user stories, a strong definition of done, and continuous process improvement through removal of impediments. Velocity of the team, quality of the software, satisfaction of the users, and revenue for the company will always increase several hundred percent if communication saturation goes up and Scrum constraints are properly enforced. Waste will be flushed from the system and the team will go from strength to strength.

When implementing Scrum, it is therefore essential to understand Scrum as an ecosystem of interdependent parts. The balance of each part with the other needs to be inspected daily. A simple set of metrics provides a dashboard similar to an aircraft cockpit. Watching altitude, direction, speed, and rate of descent can keep you on track even in heavy weather.

3. Current state

People are often measuring hours of work accomplished or tasks completed without being able to clearly demonstrate forward progress on the product owner's roadmap or demonstrate process improvement that increases value contribution. Management cannot compare performance of Agile teams straightforwardly. Productivity and quality are less than 25% of what they could be with properly functioning teams.

There are, however, a few teams that have broken through the barrier of mediocre performance. As an example, here we have data on five teams from MySpace in California. Teams at MySpace worked on a variety of projects, from SEO and framework standards to internal tools and user features that manage profiles and accounts for hundreds of millions of users building their personal web pages.

3.1. Establishing baseline velocity

The baseline velocity (100%) is established for a team during the first Sprint. The Product Owner presents the prioritized Product Backlog in the Sprint Planning meeting. This is estimated using Planning Poker and story points [7]. The team selects what can be accomplished during the Sprint and the Product Owner determines exactly what is "Done" at the end of the Sprint. The sum of the original estimates for the approved work is the baseline Velocity.

Velocity is defined as:
$V = \sum$ of original estimates of all accepted work

At MySpace, the baseline velocity is often perceived by the Team as being too low, as they had previously felt a sense of achievement for motion instead of completion. This delta serves to highlight the scale of suboptimization which will be overcome with successful application of the Scrum framework.

3.2. Daily Stand-Up Modifications

In order to collect data indicating progress during the Sprint and get new Teams operational more quickly, a few modifications to the standard Daily Stand-Up format were necessary.

The first is that we structure the meeting around the Sprint Backlog. Most Teams use a standard format wherein each individual answers the standard three questions:

1. *What did you do yesterday?*
2. *What are you going to do today?*
3. *What, if anything, is blocking you?*

We instead shift the focus of the meeting from the individuals to the Sprint Backlog. Starting with the highest priority card which is not yet completed in each Daily Stand-Up, the entire team discusses their collective contribution toward completing that card. They then estimate their collective contribution's complexity in story points as if the previous day's contribution had been presented during the Sprint Planning meeting as the entire goal of the body of work. The team then collectively plans the fastest and most effective way to share the work in order to move that card into the Done column as quickly as possible. Finally, we discuss anything that blocks the work or has the potential to slow it down for any reason. So the restructured Daily Stand-Up questions become:

1. What did WE achieve yesterday on Priority 1?
2. What was OUR contribution on Priority 1 worth in Story Points?
3. What is OUR plan for completing Priority 1 today?
4. What, if anything, is blocking US or has the potential to slow US down today?

These questions are then repeated for each lower Priority card remaining in the Sprint Backlog until either all cards have been discussed or the 15 minute allotted time has elapsed, whichever comes first.

These modifications serve several purposes. Shifting the focus from the individual to the priorities helps them begin to function more as a team. It encourages consideration of how to effectively subdivide the work for quicker completion, overcoming the often difficult to observe technical silos that specialists tend to prefer.

We also find better quality updates and more attentive participation from all team members as a result of question 2. Because each team member now has a need to understand the complexity that has been resolved in order to vote on it, updates on the order of *"Yesterday, I worked on card 1. Today, I will keep working on card 1. No impediments."* are no longer tolerated by the team. They become a self-policing group, both demanding quality updates and full attention from all team members to keep the meeting efficient.

Through sheer repetition, we also find that the quality and speed of estimation in Story Points improves more quickly using this method, especially for groups that may have previously been unfamiliar with them.

The more detailed discussions of achievements aide in cross-training the entire group more quickly, as they will hear and be asked to estimate the work of teammates with specialties that may differ significantly from their own. This also quickens the Team's learning about one so that they can move through the *Forming, Norming, Storming, Performing* phases rapidly.

Finally, and critically, it overcomes fractional thinking. For example, it is typical for an engineer who is working on a task estimated as 5 story points to report 1 point per day for 5 days if s/he feels that the work is progressing smoothly. This creates a false sense of uniformity in the rate of complexity resolution and often masks estimation inaccuracies which could be discussed in Retrospectives to help the entire Team become better at the initial estimates.

2.3 MySpace Team Data

Data on five teams at MySpace is summarized in Figure 2. The solid curve in the middle of the graph is average velocity for all teams for each Sprint. The upper and lower curves show the maximum and minimum achievement from the data.

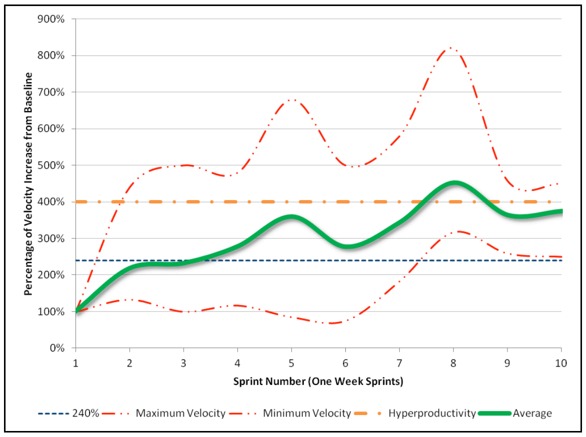


Figure 2. Velocity of MySpace Teams by Sprint

The lower dotted line is 240% percent of baseline velocity. This threshold was used to recognize that

Teams had achieved a level of proficiency with the Scrum Framework so that the Agile Coach could begin gradually returning control from the Shock Therapy model to the team members. This threshold was usually crossed in 3-5 one-week Sprints. Teams that achieve this typically went on to surpass 400% (upper dotted line) into a hyper-productive state in later Sprints. The low data points were from the only team in this data set where the MySpace Agile Coach did not assume the Scrum Master role. The existing Scrum Master failed to enforce constraints.

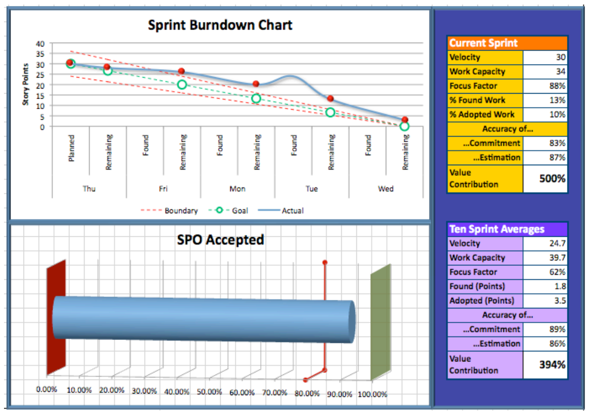
These teams were all monitored by a carefully selected set of metrics that was used to analyze performance in real time. Flying these teams into the hyperproductive state required careful balance of the altitude, speed, direction, and rate of descent on the burndown chart at all times. Failure to do this causes a hyperproductive team to spiral out of control. This result is velocity that descends to baseline level.

3. A simple set of metrics can help create and maintain a hyperproductive state for all teams

Good metrics help the team to measure their own performance. They help management compare the performance of multiple teams with apples to apples metrics. They lay down a framework for consistent data collection so that measured hyperproductivity is clear.

Good metrics also give the ScrumMaster a clear framework as a basis to advise the team. They measure the impact of modification of the environment (removal of impediments) on team performance.

When we say team value contribution is up 200%, we want it clear and demonstrable what we mean. TVC+ (targeted value contribution increase) allows us to compare the increase in horsepower of the team with the increase in revenue generated by the Product Owner's backlog. Team measurements show how well the team satisfies the product owner requirements.



Current Sprint	
Velocity	30
Work Capacity	34
Focus Factor	88%
% Found Work	13%
% Adopted Work	10%
Accuracy of...	
...Commitment	83%
...Estimation	87%
Value Contribution	500%

Ten Sprint Averages	
Velocity	24.7
Work Capacity	39.7
Focus Factor	62%
Found (Points)	1.8
Adopted (Points)	3.5
Accuracy of...	
...Commitment	89%
...Estimation	86%
Value Contribution	394%

Figure 3. Sprint Burndown Chart

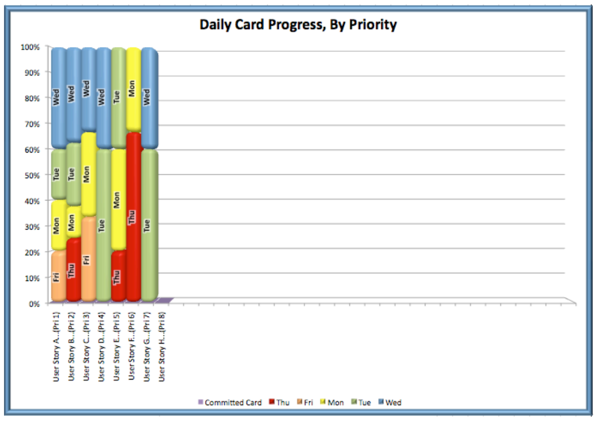


Figure 4. Daily Card Progress

Daily card progress allows us to see the rainbow of time. Red is first day of sprint and blue is last day. Why did the team start priority 2, 5, and 6 before priority 1 or 3? It enables the ScrumMaster to facilitate a discussion on priorities.

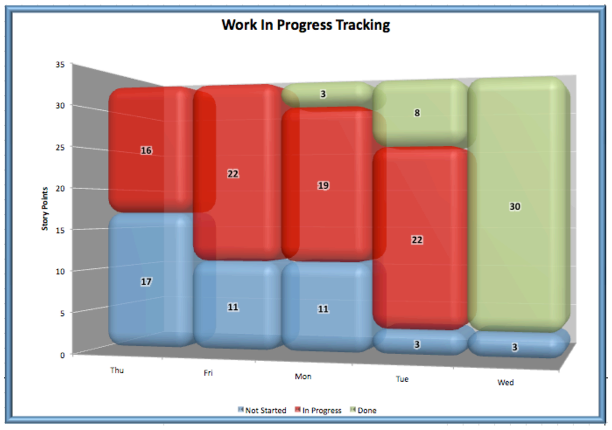


Figure 5. Work in Progress

Blue is work not started. Red is work in progress. Green is work considered Done.

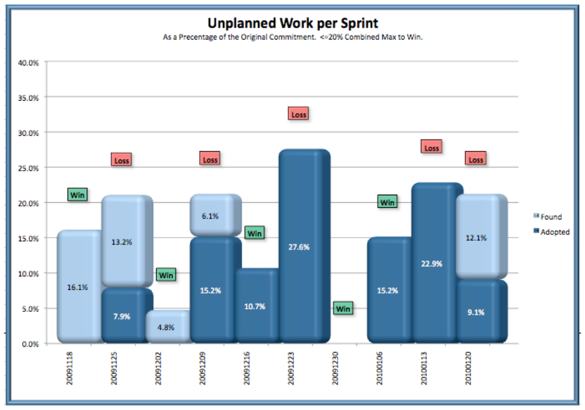
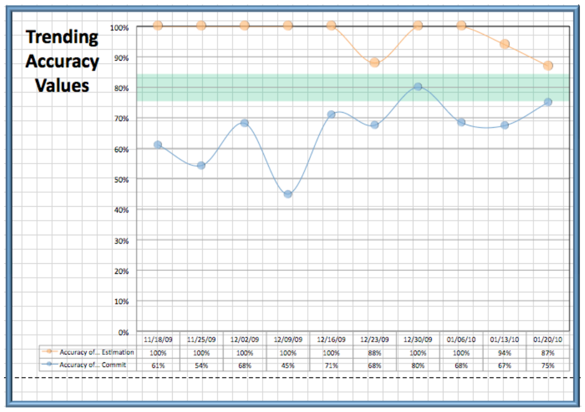


Figure 6. Unplanned Work

Careful tracking of unplanned work is essential to detecting and removing waste from a hyperproductive team. A Sprint can only be considered a Win if at least 80% of the original commitment was approved by the Product Owner, and the combined surprise work remains at a level of 20% or less of the original commitment.



	11/18/09	11/25/09	12/02/09	12/09/09	12/16/09	12/23/09	12/30/09	01/06/10	01/13/10	01/20/10
Accuracy of... Estimation	100%	100%	100%	100%	100%	88%	100%	100%	94%	87%
Accuracy of... Commit	61%	54%	68%	45%	71%	68%	80%	68%	67%	75%

Figure 7. Trending Accuracy Values

4. Conclusions

Hyperproductive Scrum teams need a simple set of metrics to provide subtle control to the team. Without these metrics, performance of the team can be unstable and loss of control will result in lowered velocity. Super-performing teams typically abandon hours as a means of tracking progress as it introduces waste into the system, lowers velocity, and reduces predictability. The simple set of metrics introduced here are easy to implement and have a powerful effect on performance of hyperproductive Scrum teams.

5. References

[1] G. Benefield, "Rolling Out Agile at a Large Enterprise," in *HICSS'41, Hawaii International Conference on Software Systems*, Big Island, Hawaii, 2008.

[2] J. O. Coplien, "Borland Software Craftsmanship: A New Look at Process, Quality and Productivity," in *5th Annual Borland International Conference*, Orlando, FL, 1994.

[3] M. Cohn, *User Stories Applied : For Agile Software Development*: Addison-Wesley, 2004.

[4] J. Sutherland, A. Viktorov, J. Blount, and N. Puntikov, "Distributed Scrum: Agile Project Management with Outsourced Development Teams," presented at the HICSS'40, Hawaii International Conference on Software Systems, Big Island, Hawaii, 2007.

[5] J. Sutherland, G. Schoonheim, and M. Rijk, "Fully Distributed Scrum: The Secret Sauce for Hyperproductive Offshored Development Teams," in *Agile 2008*, Toronto, 2008.

[6] C. Jones, "Development Practices for Small Software Applications," Software Productivity Research 2007.

[7] M. Cohn, *Agile Estimation and Planning*: Addison-Wesley, 2005.

[8] J. Sutherland, *Scrum Handbook*: Scrum Foundation, 2010.

[9] J. Sutherland and I. Altman, "Take No Prisoners: How a Venture Capital Group Does Scrum," in *Agile 2009*, Chicago, 2009.

[10] J. Sutherland, G. Schoonheim, N. Kumar, V. Pandey, and S. Vishal,

"Fully Distributed Scrum: Linear Scalability of Production Between San Francisco and India," in *Agile 2009*, Chicago, 2009.

[11] C. Jakobsen and J. Sutherland, "Scrum and CMMI – Going from Good to Great: are you ready-ready to be done-done?," in *Agile 2009*, Chicago, 2009.

[12] M. Beedle, M. Devos, Y. Sharon, K. Schwaber, and J. Sutherland, "Scrum: A Pattern Language for Hyperproductive Software Development," in *Pattern Languages of Program Design*. vol. 4, N. Harrison, Ed., ed Boston: Addison-Wesley, 1999, pp. 637-651.

[13] L. Putnam and W. Myers, *Industrial Strength Software: Effective Management Using Measurement*: IEEE, 1997.

[14] J. Spolsky. (2005, Hitting the High Notes. *Joel on Software*.