

# THE IMPACT OF AGILE. **QUANTIFIED.**



Agile and lean are built on a foundation of continuous improvement: You need to inspect, learn from and adapt your performance to keep improving. Enhancing performance begins with having accurate, comprehensive data. The multitenant architecture of Rally Software is uniquely positioned to provide access to anonymized industry benchmarking data from tens of thousands of agile teams.

Rally Software from Broadcom provide performance metrics and benchmarking data for individual teams, teams of teams and even whole business units, departments and organizations.

## Background

These insights give you real-world numbers to make an economic case for getting the resources you need and getting your people to commit to change.

### About the findings

Though people have made agile recommendations for many years, we've never been able to say how accurate they actually are or how much impact a particular recommendation might make. The findings in this document were extracted by looking at nonattributable data from more than 160,000 projects, 50,000 agile teams, and 13,000 active teams using the Rally Software.



## The Software Development Performance Index

The Software Development Performance Index (SDPI) is a balanced measurement framework researched and developed in cooperation with the Software Engineering Institute (SEI) at Carnegie Mellon University. The SDPI measures performance across the key dimensions of Quality, Productivity, Predictability, and Responsiveness. The framework's data and surveys include a formula for calculating performance measurements and guidance for how to use the metrics based on your context.

## Correlation: not necessarily causation

The findings in this document are extracted by looking for correlations between decisions or behaviors (keeping teams stable, setting your team sizes to between five and nine, keeping your work in process—WiP—low, etc.) and outcomes as measured by the dimensions of the SDPI. As long as the correlations meet certain statistical requirements,\* we report them here. However, correlation does not necessarily mean causation. For example, just because we show that teams with low average WiP have one-quarter as many defects as teams with high WiP, doesn't necessarily mean that if you lower your WiP, you'll reduce your defect density to one-quarter of what it is now. The effect may be partially or wholly related to some other underlying mechanism.

## About the four dimensions of performance



### Responsiveness

Based on time in process (or time to market): The amount of time that a work item spends in process.



### Quality

Based on defect density:  
The count of defects divided by man days.



### Productivity

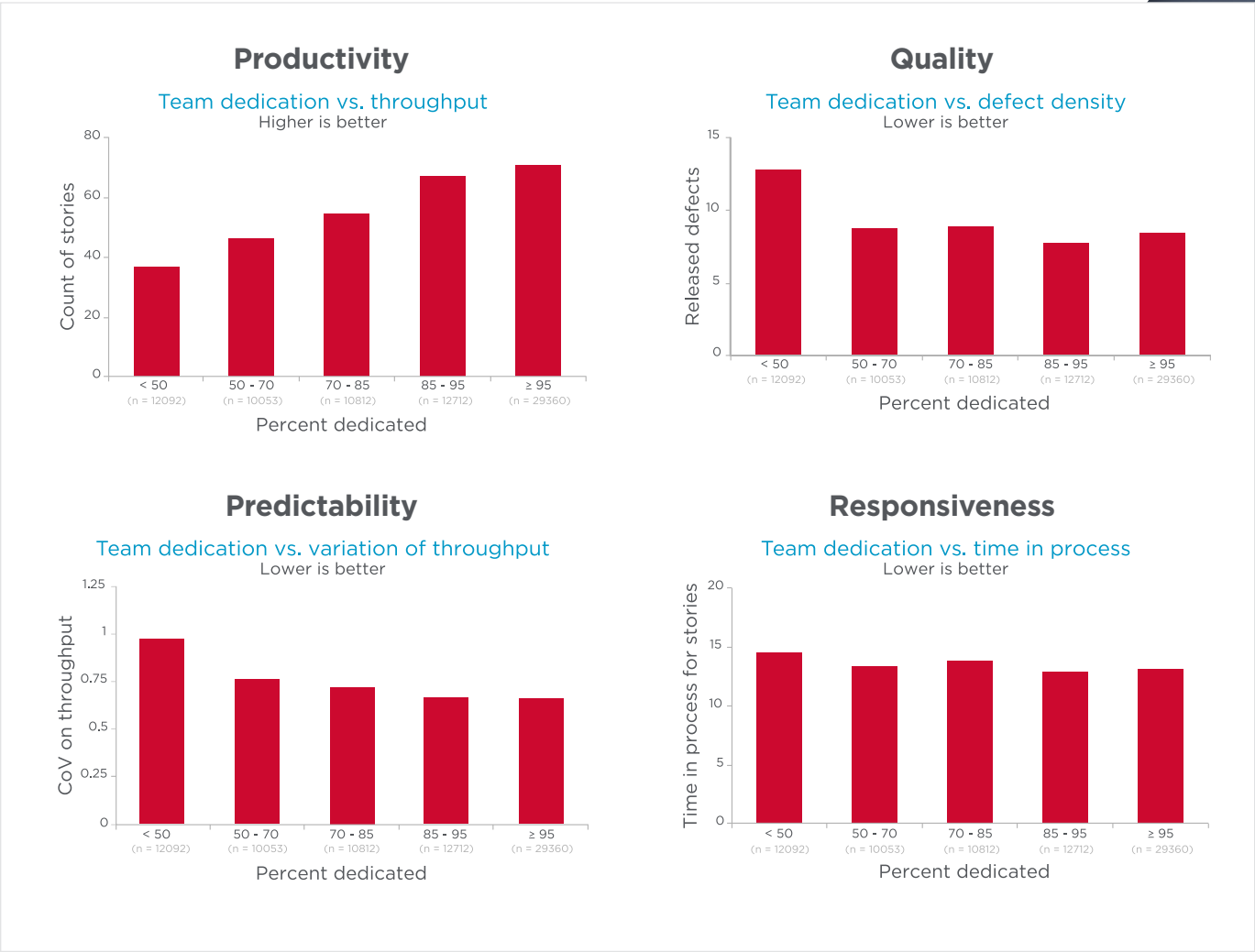
Based on throughput/team size:  
The count of user stories and defects completed in a given time period.



### Predictability

Based on throughput variability: The standard deviation of throughput for a given team over three monthly periods divided by the average of the throughput for those same three months.

# Double Your Productivity



If people are dedicated to only one team rather than multiple teams or projects, they stay focused and get more done, leading to better performance. But which aspect of performance is impacted most?

The answer is Productivity. We can see that there is almost a two to one difference in throughput between teams that are 95% or more dedicated compared with teams that are 50% or less dedicated.

Dedicating people to one team also has an impact on Predictability and Quality, but mostly in the extreme. You can see from the charts showing the variability of throughput and defect density, the effect is most prominent for the group that is less than 50% dedicated.

We can see that there is almost a **2:1 difference** in throughput between teams that are **95% or more dedicated** compared with teams that are **50% or less dedicated**.

On a positive note, the recommendation that we dedicate people to one team is widely followed. You can see in the histogram that the highest spike is in the far right. This is the count of the number of team quarters where 99% or better of the work was done by people who are dedicated to this one team. The next bar to the left is the 98 to 99% group, and it's the second highest. This histogram shows that we are consistently dedicating people to one team.

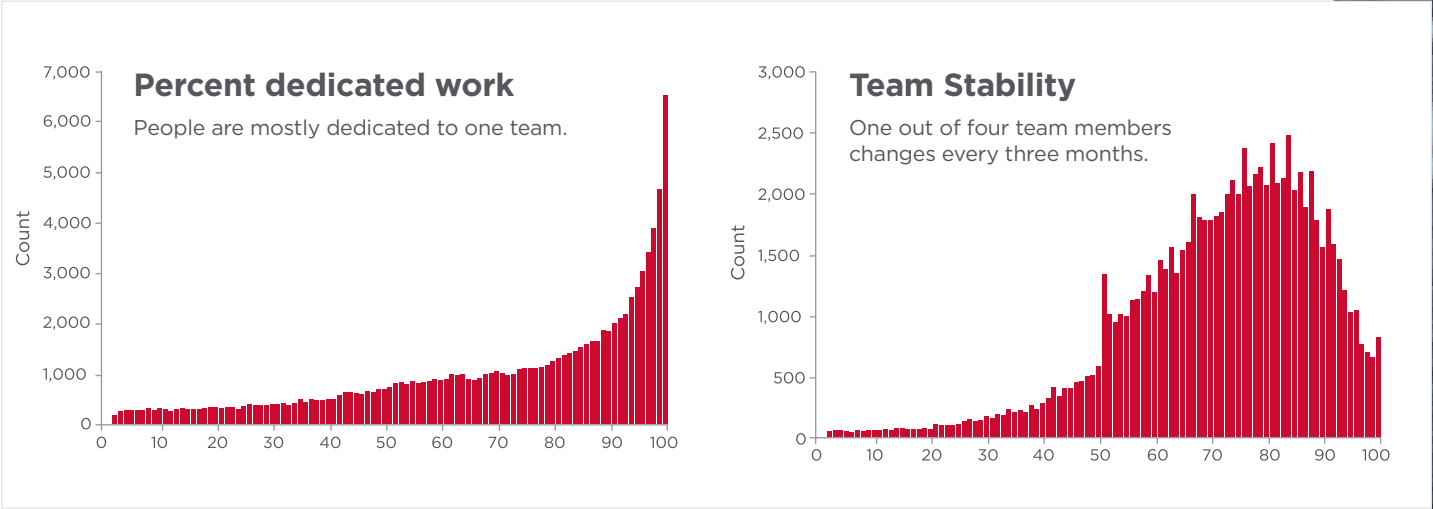
However, the story is not so good for the agile recommendation of keeping teams stable. The stability metric measures how many of the team members stay the same from one quarter to the next. This histogram shows that very few teams actually have 100% stability. The median of this data is at 74.8%, which means that roughly one out of four people on these teams changes every three months.

Teams are very unstable.

**Key Findings**


Stable teams are associated/correlate with:

- 60% better Productivity.**
- 40% better Predictability.**
- 60% better Responsiveness.**



# Double Your Productivity

Unstable teams are associated with lower performance, which makes sense. If we shift the teams around, we have to train new team members. While we are ramping them up, we're not getting work done. Again, Productivity (throughput effect of up to 60%) is most impacted. But Predictability (variability of throughput effect of up to 40%) and Responsiveness (time-in-process effect of up to 60%) also show a significant effect.



**Recommendations:**

Dedicate people to a single team.

Keep teams intact and stable.



$$a^2 + b^2 = c^2$$

# Improve Quality by 250%

We looked at teams that followed four different estimating processes.

**The first group**, which is only three% of our teams, did no estimating, even though 90% or more of their work was put into iterations.

**The second group** is doing Full Scrum. They are consistently putting story points on their stories before adding them to iterations, and they are also consistently breaking those stories down into tasks and making task-hour estimates. This group represents the vast majority of our teams: 79%.

**The third group** we have labeled Lightweight Scrum, and it represents 10% of the teams in the study. Some agile coaches suggest that mature teams may be able to skip task breakdown and task-hour estimating without hurting performance. Let's see if the data bears this out.

**The fourth and last group** is teams that are not doing story point estimation but are doing task-hour estimates. They do all of their estimating in hours. We were a bit surprised to see that 8% of the teams in the study were doing this, because we know of no agile coaches who recommend this process. We believe that these are teams that have come from a pre-agile world and started to use Rally Software with little or no coaching. They did their estimates in hours before they started using Rally, and that's what they're used to.

Teams that follow the **Full Scrum** process perform better than most alternatives, but **Lightweight Scrum** is actually better overall.

Process Type	Percent of Teams
No estimates	3%
Full Scrum Story points and task hours	79%
Lightweight Scrum Story points only	10%
Hour-oriented Task hours only	8%

## Key Findings

Teams doing Full Scrum have **250%** better Quality than teams doing no estimating.

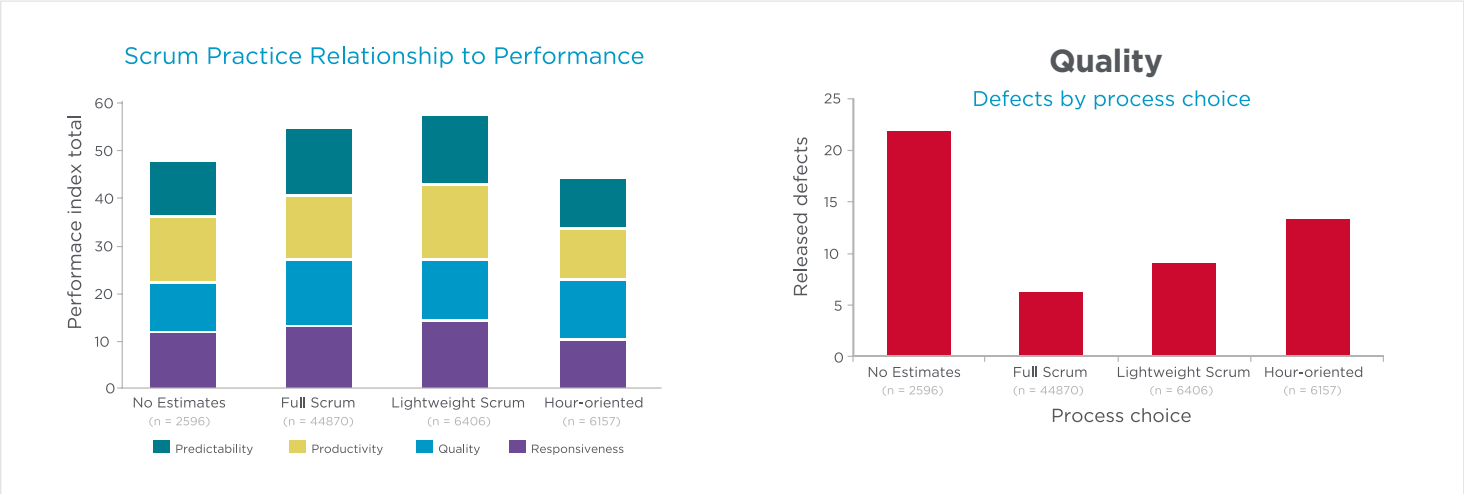
Lightweight Scrum performs better overall, with better Productivity, Predictability and Responsiveness.



What we found when we compared these various process choices is that teams that follow the Full Scrum process perform better than most alternatives, but Lightweight Scrum is actually better overall. This chart shows a score for each of the four dimensions added together.

It's interesting to note that the group that we believe has received the least coaching (task-hour estimates only) performs the worst, and the coaching recommendation for mature teams (Lightweight Scrum) performs best.

There is one dimension where Full Scrum outperforms Lightweight Scrum, and that is the dimension of Quality. There is a 250% difference in defect density between the best and worst process choices, so that's pretty dramatic.\*



**Recommendations:**

Experienced teams may get best results from Lightweight Scrum.

If new to agile or most strongly focused on Quality, choose Full Scrum.



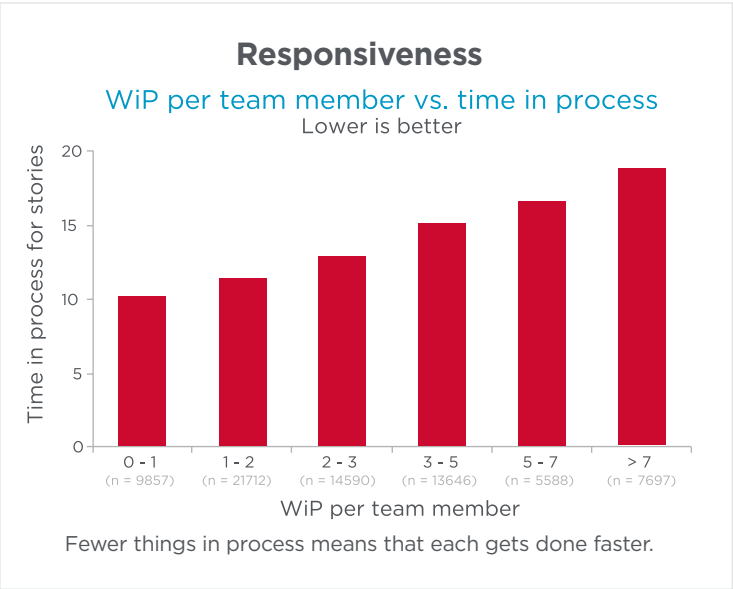
# Cut Time to Market in Half

Coaches tell you that lower WiP is always better. Is that really true?

WiP is the measure of the number of simultaneous work items that are in process at the same time.

Let's look at the relationship between WiP per team member and time in process (TiP). The group on the far left is very good at controlling their WiP. They have, on average, less than one work item per team member in process. The group on the far right is not controlling WiP very well at all. They have seven or more work items per team member in process at the same time. So a team of five would have a WiP of 35 or more.

Queuing theory (Little's Law in particular) predicts that there will be a linear relationship between WiP and TiP, and sure enough, we see these results. The TiP for teams that poorly control their WiP is up to two times as long as teams that control their WiP very well. This makes intuitive sense. The more focused you are on a few things, the quicker you'll get each one done.



## Key Findings

Teams that aggressively control WiP:

Cut time in process in half | Have one quarter as many defects | But have **34%** lower Productivity

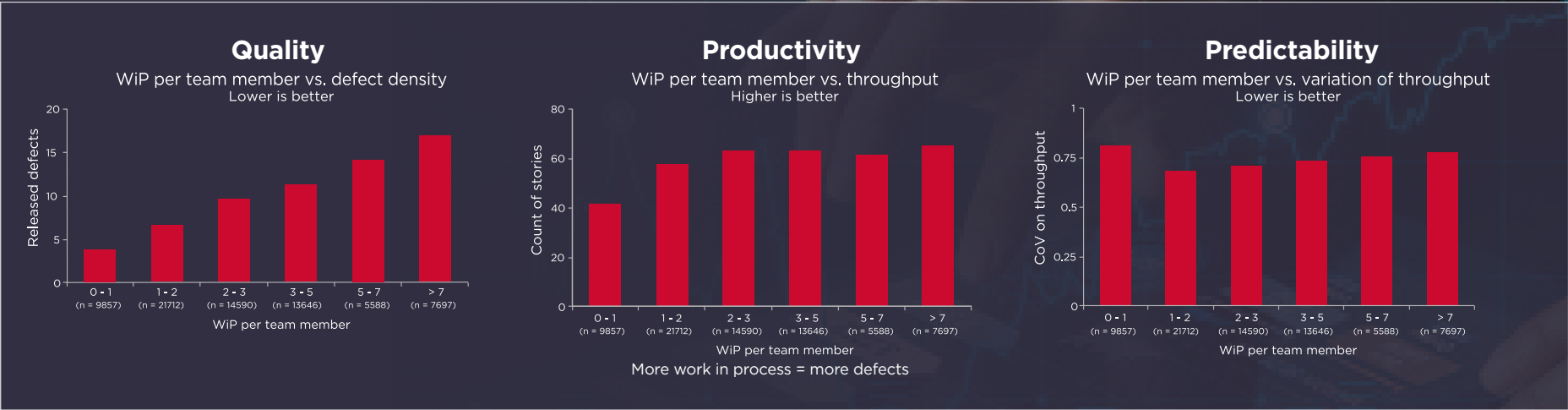
We discovered a huge effect on Quality for teams that have low WiP. Teams with the lowest WiP have four times better Quality than teams with the highest WiP.\*

Queuing theory also predicts that if you lower WiP too much, you'll have a negative impact on Productivity. This too makes sense. If some work gets blocked, there isn't enough other work to do. The two groups on the left of the productivity chart have pushed their WiP so low that they've negatively impacted their throughput.

In fact, teams with very low WiP have 34% lower Productivity.

In summary, if your WiP is already high, then by all means drive it lower. However, if your WiP is already low, consider your economic model before you decide to drive it lower. If you're at risk for missing a market window, then drive your WiP as low as possible by focusing on just a few things. But if Productivity is the primary driver of your economic model, don't push your WiP to extremely low levels because if work gets blocked, you won't have any Productivity.

Teams with the lowest WiP have **four times better Quality** than teams with the highest WiP.



### Recommendations:

If your WiP is high, reduce it.

If your WiP is already low, consider your economic drivers.

If Productivity drives your bottom line, don't push WiP too low.

If time to market drives your bottom line, push WiP as low as it will go.

# Balance Your Team Performance

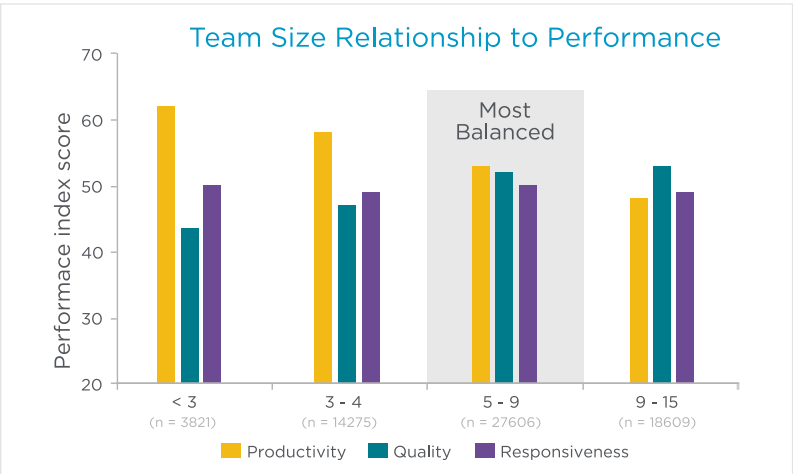
Agile recommends that the ideal team size is seven, plus or minus two. How ideal is this, when we actually look at the data?

Teams that are smaller than the recommended size tend to have better Productivity, but also tend to have worse Quality. There is little effect on Responsiveness.

## Does organization size matter?

Yes and no. It turns out that organizations of different sizes tend to make different choices. Smaller organizations tend to have a higher proportion of smaller teams, which makes sense.

Larger organizations tend to choose Full Scrum more than smaller organizations. These choices explain most of the differences we see in the variation in performance between larger and smaller organizations.



## Key Findings

Compared to teams of the recommended size (5-9), small teams of 1-3 people have:

**40%** less Predictability

**17%** lower Quality

But **17%** more Productivity

Teams larger than the recommended size have better Predictability and little effect on other dimensions.



## Recommendations:

Set up teams of 7, plus or minus 2 people, for the most balanced performance.

If you are doing well with larger teams, there's no evidence that you need to change.

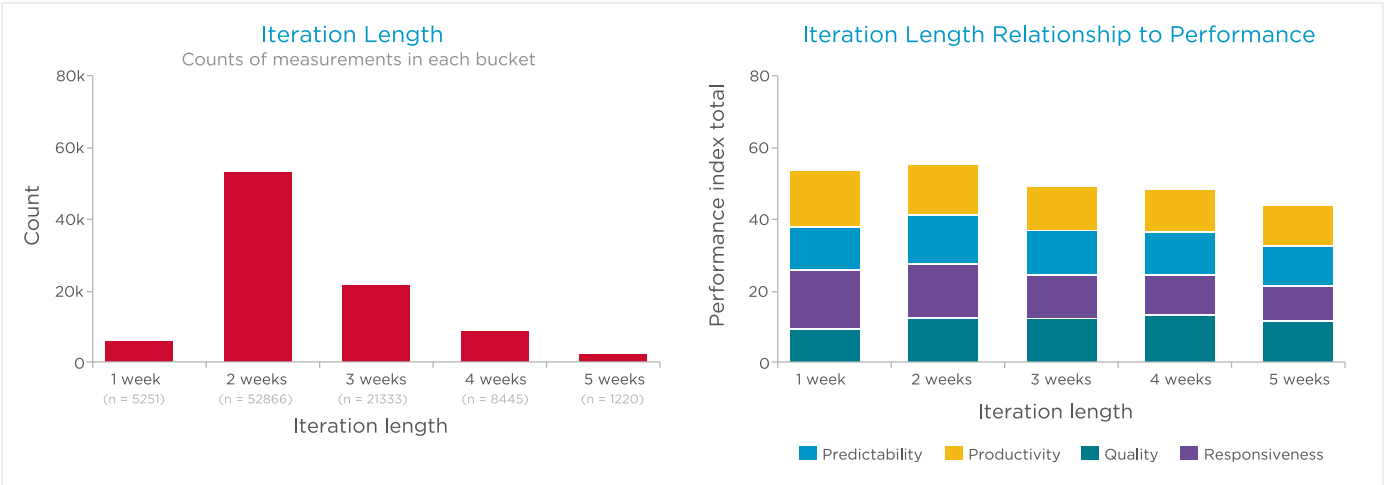
# Iteration Length

When Scrum first came out, four weeks was the recommended time frame for sprints. Over time, this has drifted toward two weeks. Is two the right answer? The overwhelming majority of teams in our sample practice two-week iterations.

## Crowd wisdom or shared delusion?

Iteration length	Teams using
1 week	6.2%
2 weeks	59.1%
3 weeks	23.4%
4 weeks	9.8%
5+ weeks	1.5%

Two-week iterations have the best overall performance. Teams practicing one-week iterations have almost equal performance but lower Quality. Compared to teams practicing four-week iterations, two-week iterations are higher in three out of four measurements. Quality is slightly higher with four-week iterations.



### Key Findings

Teams using two-week versus four-week iterations have:

**14%** more Productivity.

**8%** more Predictability.

**26%** more Responsiveness.

But Quality was **5%** lower.



### Recommendations:

Use two-week iterations for the best balanced Performance.  
Shorter iterations correlate with higher Productivity and Responsiveness.

# Ratio of Testers to Developers

Do you know if you have enough testers? What is the impact to overall Quality? We looked at ratios of testers to developers that range from no dedicated testers up to one-to-one testers to developers.

Our conclusion is that more testers lead to better Quality but lower Productivity and Responsiveness.



## Recommendations:

Testing practices, regardless of testers-to-developers ratio, still support overall performance.

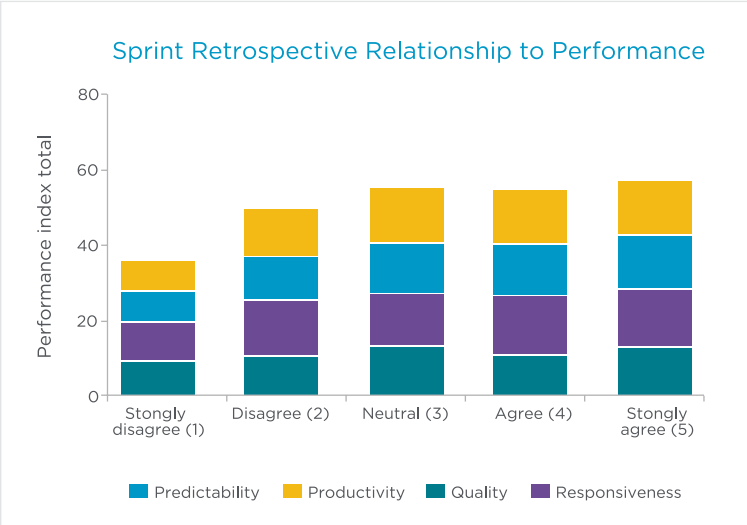
## Key Findings

Teams with up to **1 tester** per developer have **20%** higher Quality than those with less than **.3 testers** per developer. But they had **12%** less Productivity and **15%** less Responsiveness. Interestingly, teams with no testers have:  
The best Productivity | Almost as good Quality |  
But much wider variation in Quality.



# Retrospectives

Our research indicates that effective retrospectives result in teams with 20% higher balanced performance than teams that don't conduct retrospectives.



## Key Findings

Teams that strongly agree that they have sprint retrospectives have **24%** more Responsiveness and have **42%** higher Quality with less variability.

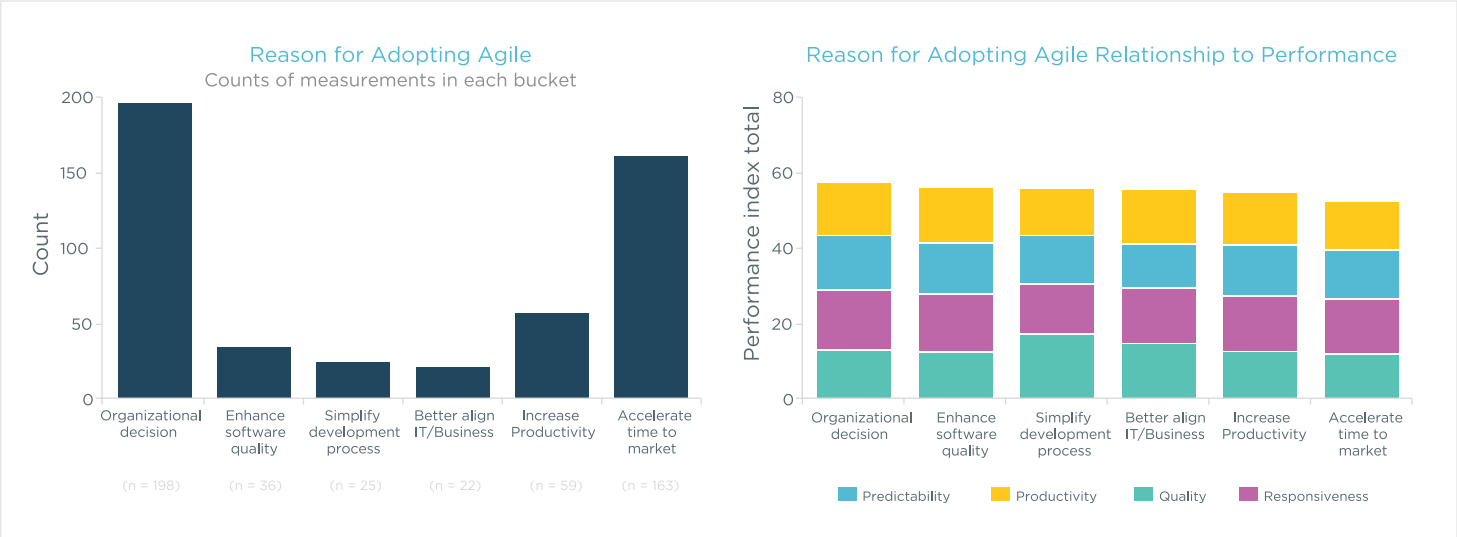


## Recommendations:

Consistent and effective retrospectives where learnings are applied for future improvement can significantly impact teams' performance in future sprints.

# Motive

We compared performance with the main reason to adopt agile. Interestingly, those people who indicated this was an organizational decision (versus simplifying the development process or increasing productivity, for example) performed the best. This is probably because more coaching and training was involved for those people whose organization as a whole supported the move to agile.



## Key Findings

Motive has a small but statistically significant impact on performance.

Extrinsic motivation does not have a negative impact on performance.

Although teamwork is selected four times more than talent, skills and experience, the latter correlate with higher overall performance.



## Recommendations:

Executive support is critical for success with agile—identify and invest in developing team members for higher overall performance.



# Rally Software Insights

## Support smart decisions across your enterprise

We found that customers using Rally Software on an ongoing basis increased their balanced team performance by 9% after 25 weeks.

### Key Findings

Teams using Rally Software for more than **25** weeks have **30%** faster average TiP.



**Want to go agile? Get a purpose-built agile tool.**

Program Manager, Insurance Industry



### Recommendations

Continue to inspect and adapt using Rally Software data insights to continually improve performance, crosspollinate learnings and see which teams need more investment.



**Rally made things so easy, I was worried that we had missed something**

Development Product Owner Lead, large financial services company

To learn more about Rally Software, visit:

<https://www.broadcom.com/rally>

# Appendix: Useful Definitions

Time buckets	Each metric is calculated for a particular time bucket. The charts in this document are all shown for time periods of three months in length.
Percentile scoring	Each raw metric has a unique distribution, and for some metrics higher is better, whereas lower is better for others. To make it easier to interpret the metric and enable the aggregation of dissimilar units into a single index, raw metrics are converted into a percentile score across the entire distribution of all similar metrics. Higher is always better for percentiles.
Calculating the Index	The SDPI is made up of several dimensions. Each raw metric is percentile scored, and one or more of those are averaged to make up a particular dimension. To calculate the overall SDPI, we take the average of the contributing dimensions' scores.
Team size	We heuristically extract the team membership by looking at who is working on what items and who is the owner of those work items, along with which Rally Software project/team those work items are in. We then determine what fraction of each team member's time is dedicated to each team. The team size is the sum of these fractions.
Responsiveness score from Time in Process (TiP)	TiP is the amount of time (in fractional days) that a work item spends in a particular state. Weekends, holidays and nonwork hours are not counted. We attribute a work item to the bucket where it left that state. You can think of this as the time bucket where work was completed. We then take the median TiP of all the work items in that time bucket. While other parameters are possible, we only look at the TiP of user stories and we define "in Process" as ScheduleState equals "In-Progress" or "Completed."
Quality score from defect density	<p>Defect density is the count of defects divided by man days, where man days is team size times the number of workdays in that time bucket. This results in a metric that represents the number of defects per team member per workday.</p> <p>We look at both the defects found in production as well as those found in test and other areas as indicated by the "Environment" field in Rally Software. We sense whether or not defects are typically being recorded in Rally for each of these types, for each team over a time period, and only use it if it passes this test. We'll take either as the Quality score or the average of the two if both are reliably recorded.</p>
Productivity score from throughput/team size	Throughput is simply the count of user stories and defects completed in a given time period. The Productivity score is the percentile scoring of this throughput normalized by the team size. While defects are shown in the drill-down charts, currently only user stories contribute to the Productivity score.
Predictability score from throughput variability	Throughput variability is the standard deviation of throughput for a given team over three monthly periods divided by the average of the throughput for those same three months. This is referred to as the coefficient of variance (CoV) of throughput. Again, we only look at user stories for this Predictability score.