4 Metrics to Measure Software Project Team Productivity
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is Agile and Scrum</td>
<td>3</td>
</tr>
<tr>
<td>Why Use Scrum metrics</td>
<td>8</td>
</tr>
<tr>
<td>4 Scrum Metrics to Measure a Team’s Performance</td>
<td>10</td>
</tr>
<tr>
<td>Sprint Velocity</td>
<td>10</td>
</tr>
<tr>
<td>Effort Estimation Variance</td>
<td>12</td>
</tr>
<tr>
<td>Scope Change</td>
<td>14</td>
</tr>
<tr>
<td>Defect Leakage</td>
<td>16</td>
</tr>
<tr>
<td>Benefits of Agile Software Development</td>
<td>18</td>
</tr>
</tbody>
</table>
The word “Agile” has the following meanings in the Oxford English Dictionary (OED):

- able to move quickly and easily
- able to think quickly and in an intelligent way

Simply speaking, these are the cornerstones of your business success. If you are not flexible to obstacles and do not make decisions quickly – you are most likely on the competitors’ heels.

Agile mindset is totally about flexibility, simplicity, and high speed.

Agile is a complex of software development methodologies that are based on an iterative approach. It allows you to deliver software incrementally instead of all at once.
Agile is about any development process that corresponds with the Agile Manifesto principles. This is a collection of software development concepts established by 14 leading tech specialists back in 2001.

To be short, **Agile Manifesto values** are as follows:

- **Individuals and interactions** over processes and tools
- **Responding to change** over following a plan
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
Put it simply, the Agile Scrum framework is a lightweight methodology which helps a cross-functional and self-managing team to deliver value. It’s also scalable and can be used by 2 or more Scrum teams using Scaled Agile frameworks for more complex projects. **Scrum has a system of roles:**

**Product Owner**
An apologist who perfectly knows the Product Goal. Being a part of a Scrum team, the Product owner keeps the needs of many stakeholders in the Product Backlog, which should represent the Product Goal.

**Scrum Master**
A person who helps to establish and maintain a Scrum framework for the team. This person helps Developers accomplish and deliver sprint increments without disturbances and distractions.

**Developers**
A self-managing team of specialists that have the required skill set to deliver value each sprint. In the context of software development, Developers may carry out such activities as prototyping, UI/UX design, software development, testing, etc.
Apart from roles, Scrum also has 4 essential events (ceremonies):

**Sprint Planning**
The first meeting includes all Scrum team members (Scrum Master, Product Owner, and Developers) to collaborate, discuss the work required to achieve Sprint Goal.

**Daily Scrum**
A day-to-day meeting during a sprint often takes around 15 minutes. It is organized to schedule the day’s work program towards the Sprint Goal.

**Sprint Review**
A demonstration meeting at the end of a sprint to show off the product developed during the last sprint to customers and stakeholders.

**Sprint Retrospective**
A final meeting before a new sprint where a Scrum team discovers the process, what went right, and how to improve communication, tools, and the process itself to increase the delivery quality and effectiveness.
### WHAT IS AGILE AND SCRUM

The framework includes three main artifacts that refer to a kind of work that should be done to finish sprints and the project. They help make internal information transparent for all team members.

<table>
<thead>
<tr>
<th><strong>Product Backlog</strong></th>
<th><strong>Sprint Backlog</strong></th>
<th><strong>Increment</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>The list of required actions that should be completed during the entire project development process to deliver a high-quality product. The commitment of Product Backlog is a Product Goal.</td>
<td>All the tasks created for a particular sprint should be done within one iteration. A Sprint Backlog is created based on Product Backlog during a Sprint Planning meeting. The commitment of Product Backlog is a Sprint Goal.</td>
<td>A sum of all Product Backlog elements that were completed during a sprint and the value of all previous sprints. By the end of a sprint, an Increment should be “Done” which means that it is functioning and meets the acceptance criteria of a Scrum team. The commitment of Product Backlog is a Definition of Done.</td>
</tr>
</tbody>
</table>
Why do you need Scrum metrics?

Agile productivity metrics provide Product Owners, Scrum Masters, and developers with the visibility they need to assess quality and track productivity throughout the software development lifecycle (SDLC).

So, Agile project metrics first help evaluate the project development process and indicate positive and negative points that influence the Scrum team, process, and overall performance.

What’s more, when you detect some problems that slow down the development process, Scrum performance metrics will help you determine the root cause so that you can eliminate it quickly.

However, you should clearly realize that NOT all Scrum metrics and reporting can be efficiently implemented for any kind of a project.
WHY USE SCRUM METRICS

At Ascendix, we built a list of rules and principles that one should follow to make the most out of each metric and use it efficiently to get valuable results and insights.

#1 A metric should be used by the team
Metrics should not be implemented just to follow the Agile Scrum methodology. They should be used by the team to learn, improve, and deliver a more valuable product to customers.

#2 A metric should be based on a conversation
Again, use Agile software development metrics wisely to start a discussion about the process and roadblocks that negatively affect a team’s performance. Metrics are NOT about numbers; they are about changes and analysis.

#3 A metric should answer concrete questions
All Scrum metrics that you implement in a project should answer specific questions so that you can understand what should be improved.

#4 A metric should be used together with other metrics
If you use a single metric to evaluate a project, you may bump into a tunnel vision that obfuscates the reality due to the lack of analytics data. This way, we recommend using several Agile productivity metrics together to get a balanced picture of your project activity.

#5 A metric should be easy to calculate and understand
In case you find it difficult to use a metric, then it most likely will not become useful in guiding daily activities, even if it provides great insights into your team’s work.
Sprint Velocity

Sprint velocity is one of the most wide-spread Scrum metrics which allows you to get a historic overview of how much value you have delivered at every sprint. In terms of one sprint, it is the amount of work (story points) your developers can complete under ideal circumstances.

With this knowledge, you can plan projects and predict how much work can be completed in the next sprint. The more accurate you are with this estimate, the more realistic your plans will be.

Everything you need is just to sum the story points completed at all sprints and divide them by the number of sprints.
AN EXAMPLE OF A SPRINT VELOCITY CHART

Sprint 2: 9 Completed
Sprint 3: 9 Completed
Sprint 4: 11 Completed
Sprint 5: 11 Completed
Sprint 6: 8 Completed
Sprint 7: 1 Completed
Sprint 8: 6 Completed

Incomplete

Completed
Effort Estimation Variance

Effort estimation variance is one of the most effective Agile Scrum reporting metrics. It allows you to track actual efforts made over the estimated hours to improve future performance.

Simply put, this metric enables you to figure out over/underestimation issues, identify their root causes, and eliminate such challenges within the future sprints.

So, there are **two simple formulas** that allow you to calculate over/underestimation values:

\[
\text{Overestimate} = \frac{(E-A)}{E} \times 100
\]

\[
\text{Underestimate} = \frac{(A-E)}{E} \times 100
\]

- **A** - actual effort
- **E** - estimated effort
AN EXAMPLE OF AN EFFORT ESTIMATION VARIANCE CHART

- **Sprint 2**: 69%
- **Sprint 3**: 43%
- **Sprint 4**: 6%
- **Sprint 5**: 19%
- **Sprint 6**: 2%

Hours:

- Estimated
- Actual
- Over Estimation
- Under Estimation
Scope Change (SC)

Scope change is another valuable and complementary Agile project metric that allows you to analyze root causes of no meeting commitments. Literally, scope change helps you track the number of stories added/removed within the project during sprint or release.

This metric has 2 key indicators “descope” and “scope increase” that help you determine when and why any mistakes or changes were made. Here is how you can calculate them:

\[
\text{Descope} = \frac{D}{C} \times 100
\]
\[
\text{Scope increase} = \frac{A}{C} \times 100
\]

- **D** - removed SP
- **A** - added SP
- **C** - committed SP
AN EXAMPLE OF A SCOPE CHANGE CHART

- **% Increase of Scope**
- **% Decrease of Scope**

<table>
<thead>
<tr>
<th>Sprint</th>
<th>% Increase</th>
<th>% Decrease</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sprint 2</td>
<td>34%</td>
<td></td>
</tr>
<tr>
<td>Sprint 3</td>
<td>0%</td>
<td></td>
</tr>
<tr>
<td>Sprint 4</td>
<td>44%</td>
<td></td>
</tr>
<tr>
<td>Sprint 5</td>
<td>50%</td>
<td></td>
</tr>
<tr>
<td>Sprint 6</td>
<td>0%</td>
<td></td>
</tr>
<tr>
<td>Sprint 7</td>
<td>0%</td>
<td></td>
</tr>
</tbody>
</table>
Defect Leakage

As your software product grows, the number of bugs reported will naturally increase and influence the final estimation, client happiness, and team satisfaction.

Defect Leakage is a valuable metric which allows you to track the bug ratio by summarizing the number of defects detected when a sprint or release is done.

So, it is one of the key Agile metrics which allows you to track the summarized number of defects detected when a sprint or release is done.

Primarily, this metric is about the efficiency of quality assurance processes as most bugs become transparent at the stage of testing the delivered software.
AN EXAMPLE OF A DEFECT LEAKAGE CHART

Sprint 2 Sprint 3 Sprint 4 Sprint 5 Sprint 6 Sprint 7

5% 13.33% 20% 33.33% 27.27%
#1 Prime quality product

Agile software development with Scrum is divided into multiple sprints during which clients/product owners can make changes to the requirements and functionality.

#2 High customer satisfaction

As the clients are constantly involved in the decision-making and development processes throughout the entire project delivery, they become fully satisfied with the process, result, and reduced time-to-market period.

#3 Controllability

Agile project management provides high transparency, feedback integration, and daily progress reports that help managers and stakeholders better control the project development process.
**#4 Better project predictions**

It becomes easier to indicate and predict risks when the project development is divided into multiple sprints with continuous improvements and alterations.

**#5 Low risks**

Advanced risk management is a hallmark of Agile software development with Scrum as small sprints and continuous delivery allow to better predict the upcoming development phases and reduce potential risks.

**#6 Continuous improvement**

As one of the 12 key concepts of the Agile Manifesto, continuous improvement means that managers and developers perform error correction sessions from sprint to sprint and enhance the delivery quality.
Contact Us

Looking for a software development company experienced in both creating top technology solutions and effectively managing projects? Answer a few simple questions and get a ball-park estimate.

Where we are

📞 +1 888-346-3276 Ext 2
📞 +1 972-889-8090
✉️ sales@ascendix.com
èmes ascendixtech.com
📍 Ascendix Corporate Office
   12222 Merit Drive Suite 1760
   Dallas, Texas 75251