Rework

Tags: Flow

Rework is when an engineer rewrites or deletes their own code that is less than three weeks old.

Some Rework is expected. For example, a Rework rate of 9-14% for a senior engineer can be normal.

Who can use this?

Core	Plus
✓	✓

But unusual spikes in Rework can indicate various problems. For example, an engineer could be stuck, or your project's specifications may be inadequate.

Knowing when your team's Rework rate spikes helps you have timely conversations to surface potential problems.

In this article

How is Rework calculated?

Flow calculates Rework by examining the context of code surrounding the Rework. Flow looks at git diffs before and after a Rework to identify the context of code. A git diff is a block of four or more lines of unchanged code before or after changes.

Flow determines if multiple changes occur in the same hunk of code or in separate hunks of code. A hunk is a continuous block of code. A hunk can include removed, added or unchanged lines of code.

back to top

What is the difference between Rework and New work?

The following example shows the difference between Rework and New work.

An example of New work

As long as there are git diffs between each of your code changes, the changes are considered New work.

In the example, lines 69-72 and lines 73-76 are unchanged. These sets of four lines are used as git diffs to provide context around the code. Between these git diffs is a hunk of added code in lines 72 and 73. These added lines are considered New work because they are changed code between git diffs.

An example of Rework

In the example, the deleted line 68 is Rework. It is followed by a git diff. But because there is no git diff before line 68, it is considered Rework.

Similarly, the deleted lines 77-79 and the added lines 78-108 are also considered Rework. There is no git diff after these hunks.

Learn more about causes of code churn and what to do about them.

back to top

How does Rework impact productivity?

Jason checked in the following javascript code on Monday:

```
1  // Populate the welcome text
2  $("#welcome").text("Hello Customer");
```

On Tuesday, he decided to tweak his code and checked in this change:

```
1  // Populate the welcome text
2  var name = "Hello Customer";
3  $("#welcome").text(name);
```

Notice that the last line changed. So Jason reworked one line of code. Or to put it another way, he gets no credit for the line of code he wrote yesterday.

On Wednesday he decided to tweak it again and checked the following code in:

```
1  // Populate the welcome text
2  var is_new = false;
3  var name = (is_new ? "Hello New Customer" : "Hello Customer")
4  $("#welcome").text(name);
```

Now he's changed the last two lines of code. Again, Jason gets no credit for yesterday's change and he loses credit for the original line of code he checked in on Monday. In effect, Jason reworked 100% of his code this week.

Simply put, Jason's contributions this week didn't net any new lines of code. He's doing work, but that work isn't represented as New work.

In our simple example, the net result was that Jason took three days to get this feature right. Now in all fairness this may or may not be his fault. The product manager may not have been clear. The spec may have changed. Or maybe Jason got the requirements wrong. Or there could be something else going on in the team's workflow that helps explain where this trend is coming from.

> **Tip**: Certain coding workflows, such as test-driven development (TDD) may have more rework in general. Understanding how your team codes and viewing those trends over time rather than raw percentages can be helpful to understand this big picture and identify problematic spikes.

As Jason's manager, look a little deeper as to why he keeps rewriting the same lines of code over and over again. If you're on the lookout for spikes in Rework, you can diagnose problems early and keep your team from getting

discouraged. As always, have conversations with Jason and the rest of the team to understand when patterns are to be expected and when there may be something to be concerned about.

If you need help, please contact Pluralsight Support.