How to Kill Agility: Nine Metrics Mistakes

ca technologies

Effective metrics and measurements are critical to running a high-performance business. Properly applied, they lead you to better insights, better decisions and better business outcomes. They provide feedback to spark improvement and create learning opportunities. They help you identify the right outcomes that drive you toward your business goals. Unfortunately, many businesses misuse these powerful tools in ways that actively destroy the agility they seek to create. In this paper, we highlight nine mistakes organizations make involving agile measurement at enterprise scale—and how to do it right.

The metrics systems we used before agile emphasized the wrong feedback loop. So agilists fittingly threw them out and replaced them with qualitative insight, which works well on smaller teams. But agile is going through another environmental shift. It's scaling up to larger projects and being adopted by larger organizations. In these environments, it's not enough to have only qualitative insight—you have to complement it with appropriate quantitative insight.

People perceive agile in different ways: as a manifesto of values or list of principles, as an emphasis on collaboration (like Scrum), as a repackaging of iterative development concepts, or as a focus on adapting to change versus following a plan.

It used to be very expensive to move between the different stages of the software lifecycle. Compilers ran for hours. Testing was labor-intensive. Distributing a finished product involved physical media and could take months. In this environment, it was critical to minimize the number of times you went through these costly transitions.

Fittingly, the emphasis for feedback was on the process and hope—by improving the checklists you used to stage-gate each of the transitions, maybe you could reduce the rework that overlapped these expensive boundaries. Similarly, the success of a project required, at a minimum, that it be finished before funding ran out, so there was similar emphasis on the plan feedback.

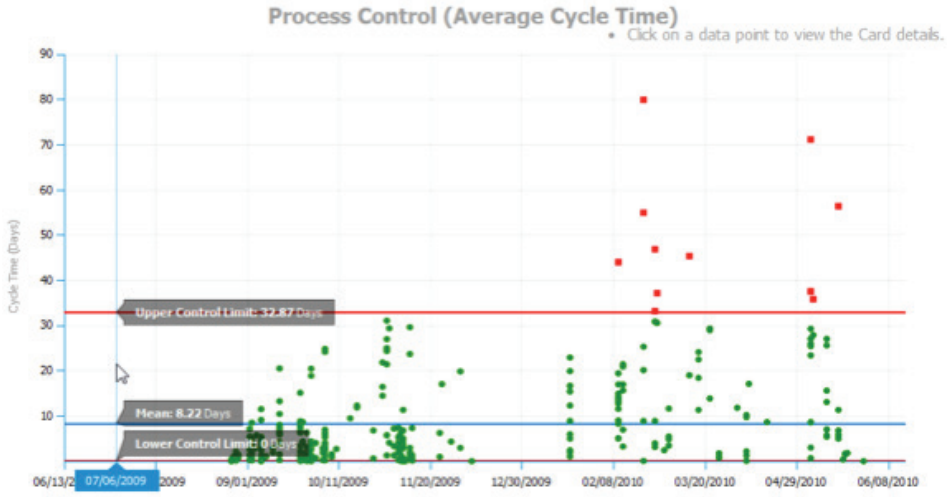
Then the environment changed. The costs of compilation, testing, and distribution have been driven close to zero. The biggest threat to success is not that you'll run out of money but that you'll miss your market window. Now, organizations embrace rework. Rather than try to build it right the first time, it's much better to build something minimally usable—and rework it based on user feedback. We no longer value feedback on the process or the plan as much as we used to; our most valuable feedback is now about the product.

Mistake One:

Using Measurement as a Lever to Drive Behavior

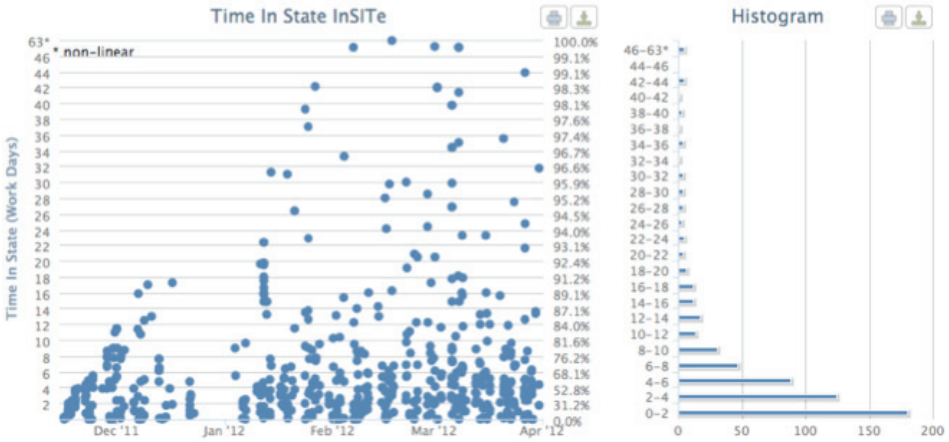
If feedback emphasis is critical to agile success, the key to effective agile measurement is to think of it in terms of feedback—not as a traditional lever to motivate behavior. Using measurement as a lever often devolves into keeping score, which is where the dark side of measurement starts.

Here's an example to illustrate the subtle but important distinction between the terms feedback and lever. Below is a chart found in an ALM tool. It's an attempt to borrow a concept from the manufacturing world that is often misapplied to the agile domain. There are several problems with this chart but for this example, pay attention to the red line and the red dots.



Each dot on the chart represents a particular user story. How high up on the chart they appear is proportional to how long the story took to be completed—the higher the story, the longer it took. That red upper control limit line and those red dots scream, "This is bad!" The stories in red took too long and are literally out of control.

What will happen the next time you show this chart? There will probably be fewer red dots, but why? Wishful thinking: people will have deeply analyzed their process and made necessary changes. But what's more likely is that they will just game the system to make sure they don't have any red dots. Maybe they'll split stories artificially instead of where they deliver value. This is bad because it's wasteful and doesn't improve the process. But what really hurts is that you've now hidden data from yourself—you're making decisions with a distorted picture of reality.



Now take a look at the chart above. Conceptually, it's very similar to the first chart but lacks the red line and red dots. With this visualization, teams can explore their evidence to learn and improve. You can hover over a dot and get the details about what happened, which should spark discussion. You can talk about probabilities to help gauge risk. You'll see that 95 percent of all stories finish in 28 days—maybe that will help you make a service-level agreement commitment.

The takeaway here is to use metrics as feedback to improve yourself and expose opportunities to learn. Never try and change people's behaviors by building rules and targets around metrics.

Mistake Two:

Unbalanced Metrics

If you try to measure agile development in a pre-agile way, you're bound to fail. The second common mistake organizations make with agile measurement is in using single-dimensional metrics, failing to consider the various aspects of effective delivery.

The need for this is fairly readily apparent. If you focus all measurement on one aspect of performance, such as productivity, you'll likely see other aspects suffer, such as quality, customer satisfaction and predictability.

It's important to launch a metrics feedback effort with at least one measure from each of these four areas:

- Do it fast
- Do it right
- Do it on time
- Keep doing it

In the diagram, you can see we've populated each of these with at least one outcome subdimension.

These outcomes dimensions form the foundation of the Software Development Performance Index (SDPI), which quantifies insights about development work and provides feedback on how process and technology decisions impact a team's performance. Of the metrics in each of the quadrants shown above, productivity, responsiveness, quality and predictability can be extracted from agile software like CA Agile Central from CA Technologies®. For customer satisfaction and employee satisfaction, you'll want to augment the approaches (like those described in mistake four) that automatically collect information.

If your metrics remain imbalanced, you won't be measuring the right outcomes in ways that fully align to your business goals. The way to counter unbalanced metrics is, of course, to consider a properly balanced set of measurements—both in how they're reported and in management's actual behaviors in response to the data.

Mistake Three:

Believing That Metrics Can Replace Thinking

Lord Kelvin says, "... when you cannot express it in numbers, your knowledge is of a meagre and unsatisfactory kind ..." But your teams hear, "We don't trust you."

Mistake three happens when organizations believe that quantitative insight can replace qualitative insight. One of the worst behaviors associated with this mistake is having policies that automatically trigger based on metrics data, such as one company that automatically considered a team at risk if they delivered less than 50 percent of a commitment, regardless of circumstances. One of the primary principles of agile is to trust the folks closest to the work because they have the most context to make decisions. In an agile world, the sort of quantitative evidence that you get from metrics must complement and augment qualitative insight, rather than seek to replace it.

In the best of circumstances, you can move in a virtuous cycle between qualitative and quantitative. Frequently, you'll want to validate a performance hypothesis you have (qualitative insight) by using metrics (quantitative insight). The results of that analysis will lead to more questions. Creating a cycle of hunches that you validate via metrics is very powerful and can lead to huge leaps forward in decision making. This requires support for ad-hoc analysis. Connectors from CA like the Excel® plugin, as well as APIs, data access toolkits, custom grids, custom reports and custom dashboard panels all help you avoid making mistake three. Remember, metrics don't steer you—they illuminate the road so you can select the most advantageous route to take. Metrics have the best impact when you make them accessible to people who can use them to make more effective decisions.

Mistake Four:

Too-costly Metrics

Sometimes it's too expensive or burdensome to get the exact metric you want. Even when the actual cost—the time it takes your developers to record extra information that helps calculate a certain metric—is low, the perceived burden can be much greater, especially in an agile environment. Developers are often your most valuable resources and you don't want to burden them unnecessarily. You should only ask your developers to input manual data if they believe it leads to a measurement whose value exceeds the perceived burden (see mistake one).

What this means is that in many cases, qualitative insight is your best alternative—but that's OK, maybe even desirable (see mistake three). However, you can turn a qualitative perception into quantitative insight via a survey. That's why two of the SDPI dimensions are customer satisfaction and employee engagement. To capture a lot of value with little difficulty, we recommend using a very lightweight metric. In many cases, this could be as simple as a single question, such as net promoter score.

When assessing the value of a particular metric, make sure you include both the actual cost and the perceived burden on your team members to record that data. Qualitative insights can become valuable quantitative data with minimal effort. And you can passively capture lots of extremely valuable data from CA Agile Central just by doing the right thing.

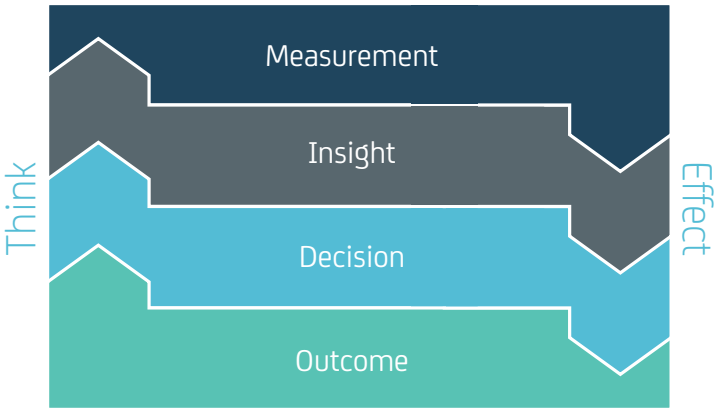
Mistake Five:

Using a Convenient Metric

Better measurement leads to better insights, which lead to better decisions and eventually better outcomes. With this chain of effect in mind, many people start by choosing measurements that are easy to acquire. But measuring what's easy can often drive the wrong behavior.

The most common version of this mistake is something we call the velocity trap. Agile teams frequently estimate the size of stories in story points to make effective commitments at the iteration planning session. When they finish the work, the sum of those points for work actually completed is called velocity and is a powerful tool for the team to use to inform future commitments. Unfortunately, it's also an incredibly convenient measure that leadership can quickly grasp. As a result, teams are frequently expected to commit to at least as many points as they did before, ignoring the context of the new work that's coming. What's worse is that some teams are even evaluated based on their contribution to velocity compared to other teams. The natural result of this is a continual story-point inflation and the utter loss of velocity as a tool that helps teams make better decisions.

So while the learning flow goes from measures to outcomes, the best way to make measurement decisions is to start by identifying the desired outcomes.



Applying this to the velocity trap, the organization may have goals to be more predictable or more productive (see mistake two), and the teams should be trusted to make the decisions that allow them to best support the organization's goal—without worrying about how their numbers look. As you work with teams, you can help them identify better metrics that fit with the end goals of the business.

Mistake Six:

Artifact Illusions

Modern tooling solutions provide a near-infinite amount of data that can be used to inform decisions by teams and leadership. Unfortunately, much of this data is simply a reflection of how people use it. For example, many Scrum teams show time-in-process metrics of ten business days. This might seem like something important on the surface but further analysis will show it to be a natural side effect of the data. In many ways, this is an extension of mistake three but deserves an entry of its own because it's so prevalent.

This mistake is of special concern because we have a tendency to trust the data summarized in lists, tables and graphs because of the polish it appears to have. We rarely remember to take the time to dig into the underlying structure and understand if the data is entirely indicative of the insights we're trying to achieve (see mistake five).

A common practice to help avoid this mistake is to sketch out the entire chain of measures and aggregations that leads to a given metric you intend to use. Frequently, you'll find something in this exercise that's a side effect of how people use a tool or a natural exhibition of the process being used. As a result of these explorations, you can better identify which symptoms are hiding a meaningful root cause and which are merely illusions in the data.

Mistake Seven:

Unclear Usage

We love our dashboards and visualizations. Unfortunately, the people who build the dashboards are rarely the people that have to use them every day. And frequently, the people expected to use the dashboards don't even know the purpose of the dashboard, much less the meaning of the data and the decisions it's expected to drive. Plus, many dashboards are built to provide as much information in as small a screen as possible.

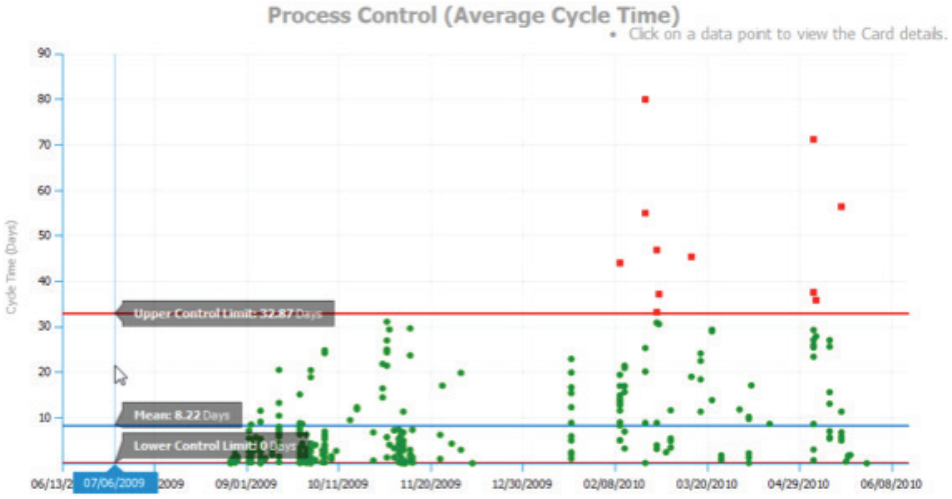
Fixing this one is pretty simple but takes a lot of diligence. Ask yourself some key questions like: Can everyone explain what data is used to calculate the measure? Can everyone explain why the results are valuable to the organization? If not, it's time to simplify the dashboard and perhaps even the underlying metrics. After all, the purpose of metrics is to improve decisions (see mistake five) and we need to ensure that people are consistently looking at the right data to support the specific decisions that need to be made now.

Mistake Eight:

Bad Analysis

Imagine setting a service level agreement thinking that you would miss it in just 0.2 percent of your cases, but in reality you miss it in 1.0 percent of your cases. If you had allocated $200,000 as a contingency, you'd actually need $1 million. Yikes. Though the percentages seem small, clearly their impact in dollars is not. This kind of bad analysis is actually a fairly easy mistake to make.

Here's the same chart we discussed in mistake one—using measurement as a lever to drive someone else's behavior—where we looked at how coloring the dots red is a lever rather than feedback. Each dot represents a user story and the higher the dot, the longer it took to be completed. The upper control limit is supposed to be a 2 standard deviation, or 97.7 percent threshold, but the data isn't normally distributed so it's really only about a 92 percent threshold.



Many phenomena in the knowledge-work domain exhibit fat-tail behavior; the tail is frequently what interests us. Careless statistical analysis will hugely underestimate this and is just one example of the way in which bad analysis can drive misinformed decisions. Be sure to invest in the expertise that enables you to do correct analysis.

Mistake Nine:

Forecasting Without Discussing Probability and Risk

Too often, teams get burned by giving their boss a linear projection as the anticipated completion date for their current backlog. Inevitably, the boss takes this as a commitment. After all, they have little other information on which to base their plans: The numbers say this is when we should finish, right? But there's a lot of uncertainty in a linear estimate. At best, it's an average projection with a 50-percent degree of certainty. It's much better to think in terms of probability and shared risk between the team and the stakeholders—something called probabilistic forecasting.

In any innovation process, there's a limit to what you can know ahead of time. Although software projects start with a vision, agile development allows software projects to unfold rather than be planned upfront. We can forecast what we know but the unknown often has a larger impact on delivery time. Other industries understand this well—insurance companies, for example, have to manage cash flow in the face of uncertainty about future risks. We accept that risks and delays are a natural part of software innovation; mitigate the associated costs and forecast accordingly. Linear forecasts based on current knowledge will always fail to give an acceptably accurate result in the face of high uncertainty and risks. Managing uncertainty by measuring how much remains in the project is the only way to know if a linear forecast projection (burndown) is an accurate prediction.

Accepting that risks and delays will always be a part of software innovation stresses the importance of agile ceremonies that help raise the visibility of risks and dependencies, and create alignment among and across teams. Quarterly big room planning is a common, effective method to make sure this happens early and often.

Learn How Agile Can Quantifiably Improve Your Software Delivery

Now that you know the nine mistakes that can kill agility in your organization, it's time to get real data about the incredible benefits of adopting agile—so you can make an economic case and drive better results.

For more information, please visit **ca.com/agile**

ca technologies

Connect with CA Technologies at ca.com

CA Technologies (NASDAQ: CA) creates software that fuels transformation for companies and enables them to seize the opportunities of the application economy. Software is at the heart of every business, in every industry. From planning to development to management and security, CA is working with companies worldwide to change the way we live, transact and communicate—across mobile, private and public cloud, distributed and mainframe environments. Learn more at **ca.com**.

CS200-246619