

Theory: What are you trying to do?

by improving :

we will increase value delivered as follows:

Test - how will you show that you've not only changed, but become more effective? i.e.delivered more value.

Question answered by this Diagnostic

it should answer a specific, clear question for a particular role or group. If there are multiple questions, design other metrics

Clear and Meaningful Name - this should be well chosen to avoid ambiguity, confusion, oversimplification.

Appropriate audience level and usage

indicate intended usages at various levels of the organization. Indicate limits on usage, if any.

Basis of measurement

clearly state what is being measured, including units. Labeling of graph axes must be clear rather than brief.

Assumptions about data or data collection should be identified to ensure clear understanding of data represented

Expected trend - what designers expect to see happen. Once the metric is proven, document common trends.

When to stop using it (for a Diagnostic)

when will it outlive its usefulness, become misleading or extra baggage? Design this in from the start.

How to game this metric - think of natural ways people will warp behavior/information to yield more 'favorable' outcomes

Warnings - recommend balancing metrics, limits on use, dangers of improper use.

A Good Agile Metric or Diagnostic ...

1 Affirms and reinforces Agile principles.

Supports the customer-intimate and value focused traits that reinforce Agile principles. This requires that people who understand Agile participate in metrics design. The truism "you get what you measure" reminds us that counterproductive behaviors may ensue if you reinforce the wrong things (ex: overtime, % utilization, paperwork).

2 Follows trends, not numbers.

Measure "one level up" [5] to ensure you measure aggregated information, not sub-optimized parts of a whole. (Ex: the financial community has a mature sense of typical industry "shapes" or trends). Aggregate above the individual team level for upper management use. To promote process health, do not track at levels more granular than "a team", and "an iteration".

3 Belongs to a small set of metrics and diagnostics.

A "just enough" metrics approach is recommended: too much information can obscure important trends. Aggregate upward to roll interrelated or competing factors into a single big picture (see previous item).

4 Measures outcome, not output.

In an Agile environment where simplicity or "maximizing the amount of work not done" is promoted, the most spectacular outcome might be achieved by reducing planned output while maximizing delivered value. Outcomes are measured in terms of delivered Customer *value*.

5 Is easy to collect.

For team-level diagnostics the ideal is "one button" automation - where data is drawn from operational tools (i.e. the Product Backlog, acceptance test tools, code analyzers). For management use, avoid rework (ex: powerpoints) and manipulation of lower level data, aggregation is preferable.

6 Reveals, rather than conceals, its context and significant variables.

Should be visibly accompanied by notes on significant influencing factors, to discourage false assumptions and facilitate improvement.

7 Provides fuel for meaningful conversation.

Face-to-face conversation is a very useful tool for process improvement. A measurement isolated from its context loses its meaning. Note: It's a good sign when people talk about what they've *learned* by using a metric or diagnostic.

8 Provides feedback on a frequent and regular basis.

To amplify learning and accelerate process improvement, metrics should preferably be available at each iteration retrospective, and at key periodic management meetings.

9 May measure Value (Product) or Process.

Depending on where problems lie, diagnostics may measure anything suspected of inhibiting effectiveness. Consider the appropriate audience for each metric, and document its context/assumptions to encourage proper use of its content. And remember: you get what you measure!

10 Encourages "good-enough" quality.

The definition of what's "good enough" in a given context must come from that context's Business Customer or their proxy, not the developers.