YOU GET WHAT YOU MEASURE!

©Ravindar Gujral, MNG Technologies/Philly Kaizen

Why Measure

- They help quantify where you are and need to be
- Help share best practices
- Benchmarking
- Common measurement techniques ensure standards across teams

  Remember what Taiichi Ohno said about Standard Work

Before I dive in

Taiichi Ohno on Standard Work

There is something called standard work, but standards should be changed constantly. Instead, if you think of the standard as the best you can do, it's all over. The standard work is only a baseline for doing further kaizen. It is kai-aku [change for the worse] if things get worse than now, and it is kaizen[change for the better] if things get better than now. Standards are set arbitrarily by humans, **so how can they not change?**

Lets see some metrics

- Velocity
- Size of code/LOC (Line of code)
- Code coverage
- Unplanned feature Change
- Schedule and Budget Compliance

Velocity

This is a good measure but is often used incorrectly.

- Velocity measures how much product backlog effort a team can handle in one sprint

- This can be estimated by viewing previous sprints or can also be established using commitment-based planning

Velocity

BUT..

It makes you start thinking of projects instead of products

Is used as a performance measure.

Maybe..

You should focus on Cycle Time: Time of request/defect/change to delivery to customer

Use velocity as a capacity measurement not performance measurement

Code Size

Function points and line of code will ensure we are on track

BUT..

Good software is measured against how FEW features provide value to customer

You could write thousands of line of code while not delivering a single valued feature to the customer!

Maybe..

This is just a bad metric?

Code Coverage

This is one of the Thou Shall rules -

> You shall have 80% code coverage. This is not a bad measure just applied incorrectly.

BUT..

> I could write hundreds of test that don't add any value or assert on responses, giving you 90% code coverage!

Maybe..

> You should educate team on the value of TDD?

Unplanned Feature

Unplanned Features slow us down, especially late feature changes which deviate from what we had planned, SO

Lets create a change control-board!

Unplanned Features

BUT..

Change is not a bad thing

This metric places the greatest value in following a plan that was made at the time of greatest ignorance

Maybe..

You planned too early? Agile methods and practices might help you.

Good software is measured by how tolerant it is to change!

Schedule and Budget Compliance

Aims to reduce schedule and budget overruns

- Calculations based on deviation to the original plan

BUT..

- Assuming scope will be fixed is not correct
- It uses time and cost as a measure for value delivered

Schedule and Budget Compliance

Maybe..

You should TimeBox and meet schedule on a cadence. If you timebox it will always be met

Biggest waste in software is extra features

Will force you to simplify process/product

Will reduce work in progress, provide early feedback

Thoughts

- Eliminate waste: Unwanted or erroneous metrics can add waste
- Empower teams so that they can continuously improve
- Create a stop the line culture
- Take the system view

  Use measurements to improve the system instead of a way to find someone to blame

Example Zara *

Design to Store in 2 Weeks

Twice Weekly orders: deliver Globally 2 Days after order

On Hanger priced, ready to sell

Manufactures in small lots at western european labor rates, mostly at coops in spain

RESULTS	Zara	Industry
New Items introduced / year	11,000	3,000
Items sold at full price	85%	60-70%
Unsold Items	<10%	17-20%
% sales spent on advertising	0.3%	3-4 %
% sales spent on IT	0.5%	2%

** From presentation by M. Poppendieck*

Q&A

Most of the problems we encounter (perhaps 90%) are the result of multiple influences, they generally cannot be attributed to a single cause. Assigning blame for a problem to the last person involved is worse than counterproductive, it will probably make the bad situation worse. **E Deming**

Be careful what and how you measure.