Software
Metrics
@sleepyfox, 2014
@sleepyfox
(Vulpes Urbanus)
Programming in 1977
Software Metrics

@sleepyfox, 2014
Software Metrics

@sleepeyfox, 2013
Dark Side of Metrics

@sleepyfox – jDays 2013
(Plot) Overview

• The rise of metrics and the measurement culture
• What went wrong
• Why it went wrong
• A story of Love, Life and Betrayal!
Tolle numerum omnibus rebus et omnia pereunt[1]
MISTAKES
It could be that the purpose of your life is only to serve as a warning to others.
Why?

- Trained as a Scientist (Physics & Astro.)
- 25 years in IT industry
- Most of my career as a Methodologist
- Projects fail with alarming regularity
- Metrics programmes don't seem to tell us why, or have the effects we expect...
Standish Group Chaos report[2]
Metrics programmes

• Software engineering is difficult*
• Symbian: How do we measure 'readiness to release'?
• CASRE – Computer Aided Software Reliability Estimation
• Origin: NASA's JPL
CASRE

- **ENUDs**
- Estimated Number of Unresolved Defects, Defects discovered per unit time
- Why doesn't this work reliably?
“You can't control what you can't measure”[4]
Lord Kelvin
Gilb's Law

"Anything you need to quantify can be measured in some way that is superior to not measuring it at all."

-- Tom Gilb
The success of measurement culture

- Renaissance
- Industrial revolution
- Information age
- Eradication of poverty, hunger and strife
Birth of Science
Birth of Science
Causal Determinism

• Popularly: 'Cause and effect'
• Newtonian Mechanics
• 'For everything that happens there are conditions such that, given those conditions, nothing else could happen'[^6]
• 'Any state (of an object or event) is completely determined by prior states'[^7]
Science of Management
"There is nothing new to be discovered in physics now. All that remains is more and more precise measurement."[8]
-- Lord Kelvin, 1900
"The more important fundamental laws and facts of physical science have all been discovered, and these are so firmly established that the possibility of their ever being supplanted in consequence of new discoveries is exceedingly remote." -- Albert Abraham Michelson, 1903
The end is in sight for Physics

- Grand Unified Theory of Everything
- That will reduce all science to a single equation, short enough to be printed on the face of a T-shirt” – Dr. Stephen Hawking
Measurement and Metrics in the Software world

Mostly these break down to:

• Productivity
• Quality
• Leading measures (estimation)
Productivity

- LOC or KLOC (*trailing*)
- Requirements
- Function points (*trailing*)
- Features
- Man-days (*as a leading measure this is estimation*)
- Story points
- Velocity
Quality

- Defect detection rate/defect injection rate
- Defect fix rate
- Defect escape rate
- Percentage of tests running to passing ratio
- Complex model-based metrics e.g. COCOMO
- Cyclomatic complexity
- Code test coverage %
Leading measures (estimates)

- Requirements (?)
- Man-days
- Story points
  - Planning poker
- #NoEstimates
  - Estimation as waste (Muri)
The downfall of counting
Proxy metrics
Hawthorne effect[9]

Also known as the Observer Effect, or the Measurement effect:
“Measuring the system changes the system”
Planning

• “Failing to plan is planning to fail” – W. Churchill

• “Plan the work and work the plan and the plan will work” – Anon

• “It is better to have a bad plan than no plan.” – Garry Kasparov
Lando's Casino
Cloud City, Bespin
Where the deal gets worse all the time
The Dark Side of planning

• “Plans are useless but planning is indispensable.” – Dwight D. Eisenhower
• “No plan survives contact with the enemy.” – Field Marshal von Moltke
• “For the word plan, substitute the word 'guess'...” – David Heinemeier Hansson
Yip's corollary

- Gilb's Law:
  "Anything you need to quantify can be measured in some way that is superior to not measuring it at all."

- Yip's corollary to Gilb's Law:
  "Anything you need to quantify can be measured in some way that is inferior to not measuring it at all."
Problems with the measures

- **Granularity issue**
  *All items are not the same size*

- **Inter-rater reliability**
  *Your 3pt story is my 1pt story*

- **Hawthorne effect**
  *Observing the system changes the system*

- **Experimenter's bias**
  *169 kinds of bias*[^10]
Goodhart's Law

• "Any observed statistical regularity will tend to collapse once pressure is placed upon it for control purposes." -- Goodhart's original 1975 formulation[11]

• Most popular formulation is:
"When a measure becomes a target, it ceases to be a good measure."
The Darker Side of Metrics

Douglas Hoffman, BACS, MBA, MSEE, ASQ-CSQE
Software Quality Methods, LLC.
24646 Heather Heights Place
Saratoga, California 95070-9710
doug.hoffman@acm.org

Abstract

There sometimes is a decidedly dark side to software metrics that many of us have observed, but few have openly discussed. It is clear to me that we often get what we ask for with software metrics and we sometimes get side effects from the metrics that overshadow any value we might derive from the metrics information. Whether or not our models are correct, and regardless of how well or poorly we collect and compute software metrics, people’s behaviors change in predictable ways to provide the answers management asks for when metrics are applied. I believe most people in this field are hard working and well intentioned, and even though some of the behaviors caused by metrics may seem strange, odd, or even silly, they are serious responses created in organizations because of the use of metrics. Some of these actions seriously hamper productivity and can effectively reduce quality.

This paper focuses on a metric that I’ve seen used in many organizations (readiness for release) and some of the disruptive results in those organizations. I’ve focused on three different metrics that have been used and a few examples of the behaviors elicited in organizations using the metrics. For obvious
Defect find/fix rate

This model is based on several assumptions that don’t hold:
1) all defects that are found are reported,
2) there is a goal of fixing (or resolving) all known defects,
3) when all known defects are fixed the product is ready to release,
4) there are reasonable resolutions for all fixed defects.
Observed behaviours

- Duplicates
- “Unassigned”
- Withholding
- “Pocket lists”
- Post-fix reporting
Bug bonus

Our goal is to write bug-free software. I'll pay a ten-dollar bonus for every bug you find and fix.

Yahoo! We're rich!

Yes!!! Yes!!!

I hope this drives the right behavior.

Yes!!!

I'm gonna write me a new minivan this afternoon!
Automated program proving

"Even perfect program verification can only establish that a program meets its specification... Much of the essence of building a program is in fact the debugging of the specification." -- Fred Brookes[13]
"Management by numerical goal is an attempt to manage without knowledge of what to do, and in fact is usually management by fear."[14]

"It is wrong to suppose that if you can’t measure it, you can’t manage it - a costly myth"[15]
WE CANNOT SOLVE OUR PROBLEMS WITH THE SAME THINKING WE USED WHEN WE CREATED THEM

-Albert Einstein
Causation fallacy

"That every effect has a cause, and we can tell which"[16]
– Gerald Weinberg
Correlation vs. causation

Fig. 1
IS FACEBOOK DRIVING THE GREEK DEBT CRISIS?

- 750m users
- 16.82
- Yield on 10-year Greek government bonds

- 3.6
- Number of active Facebook users
- 5.5m users

2005 - 2011
Causal fallacy

I USED TO THINK CORRELATION IMPLIED CAUSATION.

THEN I TOOK A STATISTICS CLASS. NOW I DON'T.

SOUNDS LIKE THE CLASS HELPED. WELL, MAYBE.

Credit: xkcd - http://xkcd.com/552/
Hierarchical reductionism

“Even though we can use reductionism to trace a problem back to its root cause, we cannot apply a constructionist view to build a system that prevents that problem happening in the first place.”[17]

– Richard Dawkins
Cognitive biases

"In the psychology of uncertainty we find that the perception of rare events is subjected to severe framing distortions: people are aggressive with risks that hit them "once every thirty years" but not if they are told that the risk happens with a '3% a year' occurrence."

– Nassim Nicholas Taleb[18]
Halley's comet

We can predict the return of Halley's Comet to the second, but not next week's weather, or (sadly) the necessary features, quality, time or resources necessary to deliver a software project. **Why not?**
HELP ME

OBI WAN
Uncertainty principle

Werner Heisenberg, 1926

\[ \Delta x \Delta p \geq \frac{h}{2} \]
Goedel's completeness theory

Kurt Gödel, 1931

“Any effectively generated theory capable of expressing elementary arithmetic cannot be both consistent and complete. In particular, for any consistent, effectively generated formal theory that proves certain basic arithmetic truths, there is an arithmetical statement that is true, but not provable in the theory.”[19]
Turing halting problem

Alan Turing, 1936
"Given a description of an arbitrary computer program (and its input), decide whether the program finishes running or continues to run forever"

    while (true) continue;

vs.

    print "Hello, world!";
Complexity

- Complexity = large numbers?
Two-body problem
Three-body problem
Water molecule
WAT?
Mandlebrot set

$$z_{n+1} = z_n^2 + c$$
Algorithmic complexity[21]

Large Limits to Software Estimation

J. P. Lewis
Disney TSL
3100 Thornton Ave.,
Burbank CA 91506 USA
zilla@computer.org

Abstract
Algorithmic (KCS) complexity results can be interpreted as indicating some limits to software estimation. While these limits are abstract they have nonetheless undermined claims occasionally made by commercial software estimation advocates. Specifically, if it is accepted that algorithmic complexity is an appropriate definition of the complexity of a programming project, then claims of purely objective estimation of project complexity, development time, and programmer productivity are necessarily incorrect.

Keywords
Estimation and metrics, project management, risks, ethical issues.

Introduction
Among the important practical problems in software engineering is software estimation — the estimation of development schedules and the assessment of productivity and quality. Is it possible to apply mathematical and scientific principles to software estimation, so that development schedules, productivity, and quality might be objectively ascertained or estimated rather than being a matter of opinion?

The debate over this question is familiar. In the case of development schedules, for example, many programmers find it self evident that accurate and objective estimates are not possible. One reader of an early version of this paper commented, “Software practitioners know about poor predictability from empirical evidence. I don’t need to prove it…”

On the other hand, there are a large number of design methods, de...

Similarly, a book promoting a software estimation package [15] states that "...software estimating can be a science, not just an art. It really is possible to accurately and consistently estimate costs and schedules for a wide range of projects," etc.

The answer to our question ('can software be objectively estimated?') has both practical and ethical implications. Newspaper headlines frequently describe the cancellation of costly software projects that are behind schedule and over budget. With computer programs now widely deployed in socially critical roles it is recognized that software professionals have a responsibility to make accurate and truthful characterizations of prospective software systems.

Given the existence of the various software methodologies and processes alluded to above, it would be easy to conclude that the problem is merely that these methods are not being practiced. On the other hand, considering the wide variety of competing methodologies and the well considered critiques of some of these methodologies [1, 2, 3, 4, 8], one may be tempted to adopt an outside perspective and ask whether all of the stated goals of these methodologies are possible even in principle.

In this paper we will look at software estimation from the point of view of algorithmic or KCS (Kolmogorov-Chaitin-Solomonoff) complexity. Section two introduces the notion of algorithmic complexity. In sections three and four we will find that algorithmic complexity results can be directly interpreted as indicating that software complexity, development schedules, and productivity cannot be objectively and feasibly estimated and so will re...

Agile environment
Mathematical limits to software estimation

- Program size and complexity cannot be objectively estimated a priori, therefore:
- Development time cannot be objectively predicted.
- Hence absolute productivity cannot be objectively determined.
- Even approximate estimators are suspect: **there is no estimator** that produces a correct fixed bound on the complexity of all programs.
Paradigm shift

"A new scientific truth does not triumph by convincing its opponents and making them see the light, but rather because its opponents eventually die, and a new generation grows up that is familiar with it."

– Max Plank, 1948
Be Mindful!

Whoaah... Steady girl... What's the matter? You smell something?
Rational responses

• Move from predictive to reactive models of software development
• Focus on metrics that are aligned with business objectives, not proxies

$$\textdollar\textdollar\textdollar\textdollar\textdollar$$-Driven Development

• Monitor side-effects, expect gaming
• Be mindful of biases, you can't compensate!
“You can't control what you can't measure”[4]
40 years on...

“You cannot control what you cannot measure”

“In my reflective mood, I’m wondering, was its advice correct at the time, is it still relevant, and do I still believe that metrics are a must for any successful software development effort? My answers are no, no, and no.”

-- Tom DeMarco[22]
Thank you

@sleepyfox
agile-environment.com

http://creativecommons.org/licenses/by-sa/4.0/
References


References


[8] Lord Kelvin, in an address to the British Association for the Advancement of Science, 1900

[9] Hawthorne effect
http://en.wikipedia.org/wiki/Hawthorne_effect

References

http://en.wikipedia.org/wiki/Goodhart%27s_law
References


[18] “Errors, Robustness and the Fourth Quadrant”, Nassim Nicholas Taleb, 2009

References


http://scribblethink.org/Work/Softestim/softestim.html