

Fall 2018

Bruce F. Webster

CS 428 –
Webster readings #4

Lies, damned lies, and project metrics [[Part I](#), [Part II](#), [Part III](#)] (Baseline, 2008)

- ◇ Same problem as project estimation, but for a project already underway
 - ◇ Most organizations are very bad at predicting when a given project will ship
 - ◇ Usually rely on ‘metrics’ that aren’t at all useful
- ◇ A meaningful, useful project metric has three key qualities:
 - ◇ Informative/predictive: tells you something important and/or when you will deliver
 - ◇ Objective: should yield the same value regardless of who is doing the measuring
 - ◇ Automated: can be done quickly and without direct human intervention
- ◇ Almost all major metrics used in IT projects lack one, two, or all three qualities

Metric “Laws”

- ◇ **Weinberg’s Law of Metrics:** “That which gets measured, gets fudged.”
 - ◇ We will distort work and reporting to achieve required or valued metrics
- ◇ **The Metric Law of 90s:** “The first 90 percent of a development project takes 90 percent of the schedule. The remaining 10 percent of the project takes the other 90 percent of the schedule.”
 - ◇ We tend to focus on low-hanging fruit in order to make metrics look good
- ◇ **The Metric Law of Least Resistance:** “The more human effort required to calculate a metric, the less often (and less accurately) it will be calculated, until it is abandoned or ignored altogether.”
 - ◇ Hence the need for automation (cf. classic joke about drunk looking for keys)
- ◇ **Must-read book:** “Measuring and Managing Performance in Organizations” by Robert D. Austin (Dorset House, 1996)

The challenge:



◇ Why is project completion so hard to predict?

- ◇ The amount of analysis (gathering relevant subject-matter information) that still has to occur
- ◇ The amount of invention (novel problem solving) that still has to occur (cf Armour, as usual)
- ◇ The amount of discovery (e.g., running into roadblocks and dead ends) that still has to occur (again, Armour)
- ◇ The adequacy of the current architecture, design and implementation
- ◇ The amount of actual coding that still has to occur
- ◇ The amount of quality engineering (testing, reviews, etc.) that still has to occur

The challenge (cont.):



◆ Why is project completion so hard to predict?

- ◆ Any and all remaining external dependencies (availability of resources, availability of technologies, deliveries from vendors and other projects, etc.)
- ◆ The talent, experience and productivity of your IT engineers and managers, as well as turnover among those employees
- ◆ The amount of business process re-engineering required to put this system into production, as well as the degree of resistance or cooperation among the affected business units
- ◆ The complexity, cohesion and comprehensibility of the overall system
- ◆ Other factors, such as scope creep, conflicting requirements, changes in business or market needs, budget constraints, or internal politics

Potential approach to useful metrics

- ◇ First, instrumentation: automated collection of wide range of metrics/characteristics over time
 - ◇ Result: time-stamped history for each metric/characteristic
 - ◇ These should be automated and objective
 - ◇ Can be tied to configuration management system and run on a regular basis
- ◇ Second, heuristics: use data collected
 - ◇ After project is done and with known timeline, use Bayesian analysis to see which combination of metrics best anticipate milestone completion
 - ◇ Use human analysis as well to look for correlations between metrics and actual progress (or lack thereof)
 - ◇ Refine set of metrics/characteristics for next project and see how well they predict progress