So I hope you've all read the article, 'Eight Secrets in Software Measurement'. If you haven't, do so now. There are spoilers ahead. I would hate to ruin the pleasure you'll receive from this required reading. Grab a cup of tea, curl up by the fireplace, and lose yourself in the magical world of 'Software Measurements'. It's worth it.

I'll start with the title. Betsy Clark (not related to Allan, as far as I'm aware) and I have a slight disagreement over the word choice. What she calls *secrets* are what I call *common sense tips*. The points she makes do not seem like furtive things whispered into the ears of managers - they have more in common with a business management course. She alludes to this of course, putting 'secrets' in quotations in the introductory paragraph, but they remain strongly unquoted in the title. A good marketing technique, I suppose.

Nevertheless, they are good pieces of advice. Even the one that references Lance Armstrong (Clark evidently couldn't see into the future) has merit if you strip away that analogy. Measurements are just a tool; it's how you use them that matters.

If I may continue with the tool analogy, Maslow's Hammer states 'if the only tool you have is a hammer...treat everything as if it were a nail.' Measurement should not be the only tool you use. If I'm trying to debug my software, I would rather have a tool that shows me the values of my variables than a tool telling me I have 50,000 bugs (a metric that says 'you're a lousy programmer' isn't as useful when trying to ship software).

That said, I believe that the difference between good companies and great companies is that the latter will make the most out of the tools available. A great company will effectively record metrics about software use and put it to good use.

But what are these metrics? From reading the paper, I'm not sure - they are never explicitly defined. Clark occasionally references these measurements (i.e. Secret 6 references individual progress reports), but rarely are there technical examples of the report (the brief summary of the one at the end does briefly count). At no point does she suggest recording cyclomatic complexity, function load analysis, program load time, or even the basic lines of code. This article is not written as a technical how-to, it seems, but rather for managers who are spearheading this transition into recording metrics.

This is not a bad thing. It approaches the issue from a different perspective than I would take. It does, however, mean that Clark's statements occasionally seem without merit. She will state something as fact, but only backs up her claims with anecdotes.

For example, when Clark says 'it's often easy to get people enthused about measures' in point 3, I have to disagree. Although I cannot vouch for every single person, nor of Clark's persuasive skills, I cannot see the general populace getting excited over measurements. Most people will resent putting in extra work to log/maintain software metrics. Even if the data collection process is completely automated, somebody has to manually shift through the data and draw relevant conclusions. Nobody wants that.

Clark suggests trying to originate a cultural change. Her suggestion for how to achieve this comes in the next section - drafting individuals to help convince others. Which implies that the individuals responsible for dealing with the data should come internally. Should a programmer, who has been hired to program, have to deal with the measurements? Should a manager, whose primary purpose is to lead, take time to analyse this data?

Clark seems to suggest so, but I take a different approach. Depending on the situation, I would assume it beneficial to hire an expert in the measurement business. Somebody who knows exactly what the data implies, does not have other responsibilities on hand, and can afford to make suggestions without risking job security. This would be dependent on the availability of money, and the willingness of the business to put their metrics into another's hand, but if measurements are as important as Clark states, then surely it should be worth it.

But, you might say, 'does this mean we should hire consultants?' And I would respond 'Probably, but only if money allows,' and then playfully pat you on your head. And then you, in your youthful brashness, would say 'But isn't that what Clark is? She says she's a consultant in the first paragraph. Doesn't this mean it's all just a complex job recruitment scheme?' And I would say 'Probably. Go play with your code and don't worry about it.'

Cheesy hypothetical conversations aside, I think it is important to hire somebody who knows what they are doing. There is a lot of data to sift through and numerous ways to interpret it. Worse, there are fewer ways to interpret it in a way to get meaningful results (and not just a small increase in numbers - tangible results that arise from a shift in company policy). Best hire somebody who has experience with these numbers.

But if you cannot afford or do not wish to hire a consultant, then such a person must come from the inside. If you are a startup, you'll likely not have the luxury of excess money.

Except her situations do not take into account the worse case scenarios. Clark says 'at least one person steps up as an early adopter.' If we follow Roger's Bell Curve ![Roger's Bell Curve] (http://upload.wikimedia.org/wikipedia/en/4/45/DiffusionOfInnovation.png), we can see that 13.5% of the population will be these early adopters (16% if you include the innovators in that). It is likely that one of your colleagues will be an early adopter, but what if there is not?

What if nobody wants to step forward and begin using measurements as frequently as Clark suggests? Or, a more realistic scenario given the numbers, what if the early adopter is unpopular? A person whose use won't convince the rest of the team that they should switch? The situation leads to a scenario where you, as the metric guru, must persuade these people to switch. Clark offers no solution - her eight secrets are more broad generalizations than concrete plans of action.

In such a situation, I would suggest attempting to persuade an individual with sway to your side. Show them the metrics, show why they can be used, and even show them 'Eight Secrets in Software Measurement'. Once they switch, you'll be able to make headway in the same way that Clark suggests.

But Clark doesn't consider the 'what-if'. Her experience is justification, and we'll take her word for it. The nitty gritty details are hinted at, but never explored.

Take for instance the statement 'I've worked with clients who have implemented detailed progress measures that begin with the individual, measuring actual progress against a plan." What are the progress measures? Lines of code? Features implemented? Complexity measurements? Each of these have their pros and cons, and to hold an individual accountable against these will likewise have its pros and cons. Holding them accountable to lines of code could encourage frivolous or bad coding habits, or holding them against complexity measurements could encourage overcomplicating simple tasks.

What about if the estimates are bad? We discussed in the lecture about how bad most of us are at estimation. If we measure our progress, and realize that it does not approach where our plan says we should be, should the individuals still be held accountable for not achieving it, or should the estimation be held accountable? These specifics are absent, probably because they are dependant on the individual case, but they need some sort of addressing to form a concrete argument.

Does the lack of support for her statements matter? You could say probably not. This is an article written over eleven years ago, and I have no idea about the impact it had in the software world. It could be that every individual utters a prayer to Clark at the beginning of every work day or that this paper was nothing but a drop in the sea. It doesn't matter.

It's the content that matters. And while she makes these claims, she professes that they are secrets meant to be implemented if you want success. But when does she provide proof or go into detail about these secrets? Hardly, if ever. In the last point, she provides some of the elusive figures - a single characteristic (project's percentage of turnover). It doesn't satisfy my curiosity about these figures. Where is my data?

It's not here. To be fair, Clark never promised that it would be here. But if I try follow her steps towards introducing measurements, and something fails along the way, I will have no way of knowing where I went wrong. And that lack of accountability does not bode well for an individual trying to introduce measurements into a project. Unless I go and hire Clark. Then everything will be fine.