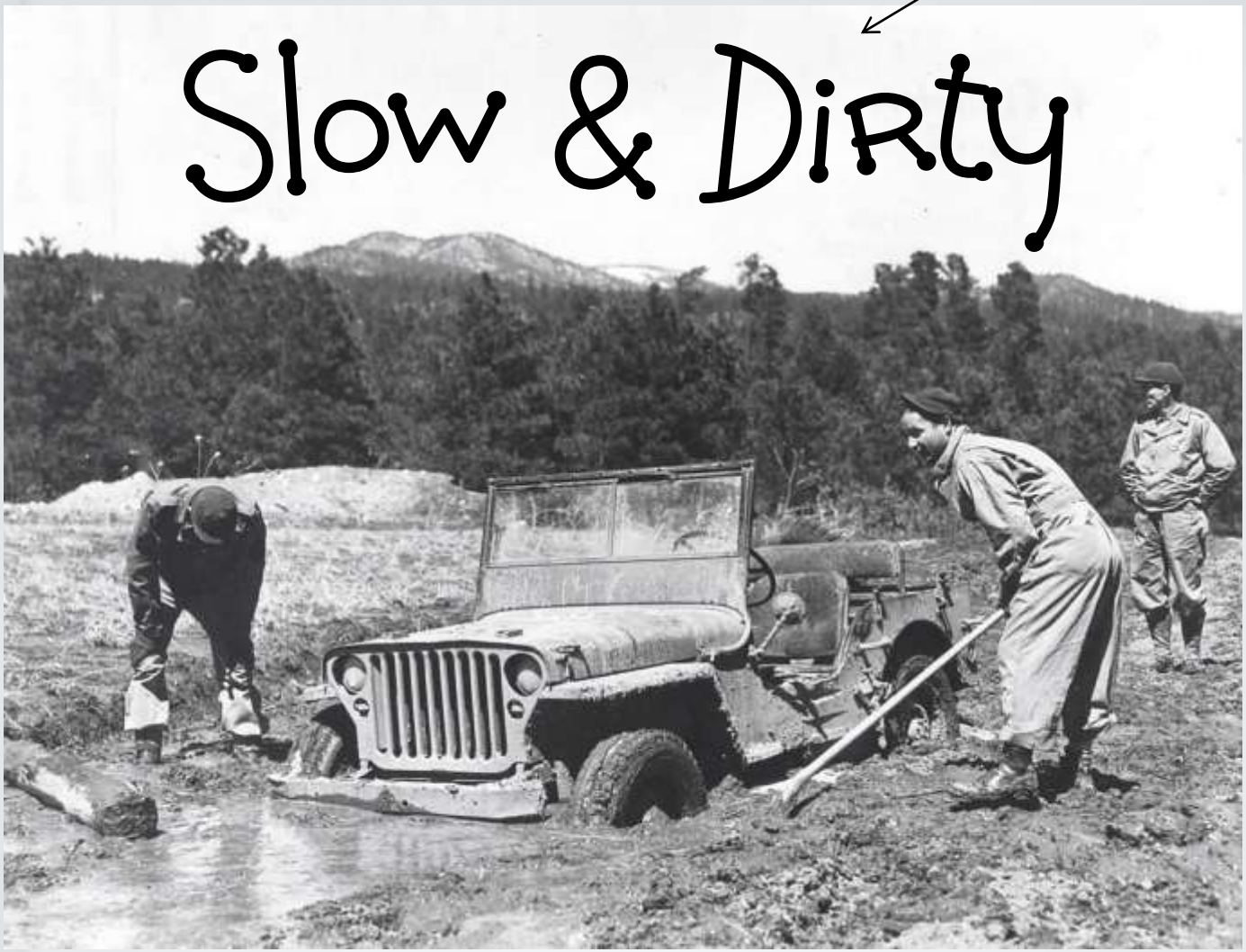
codemanship

A RANT



© Codemanship Ltd 2011

Please support Astrid Byro in her Everest challenge to raise money for Bletchley Park

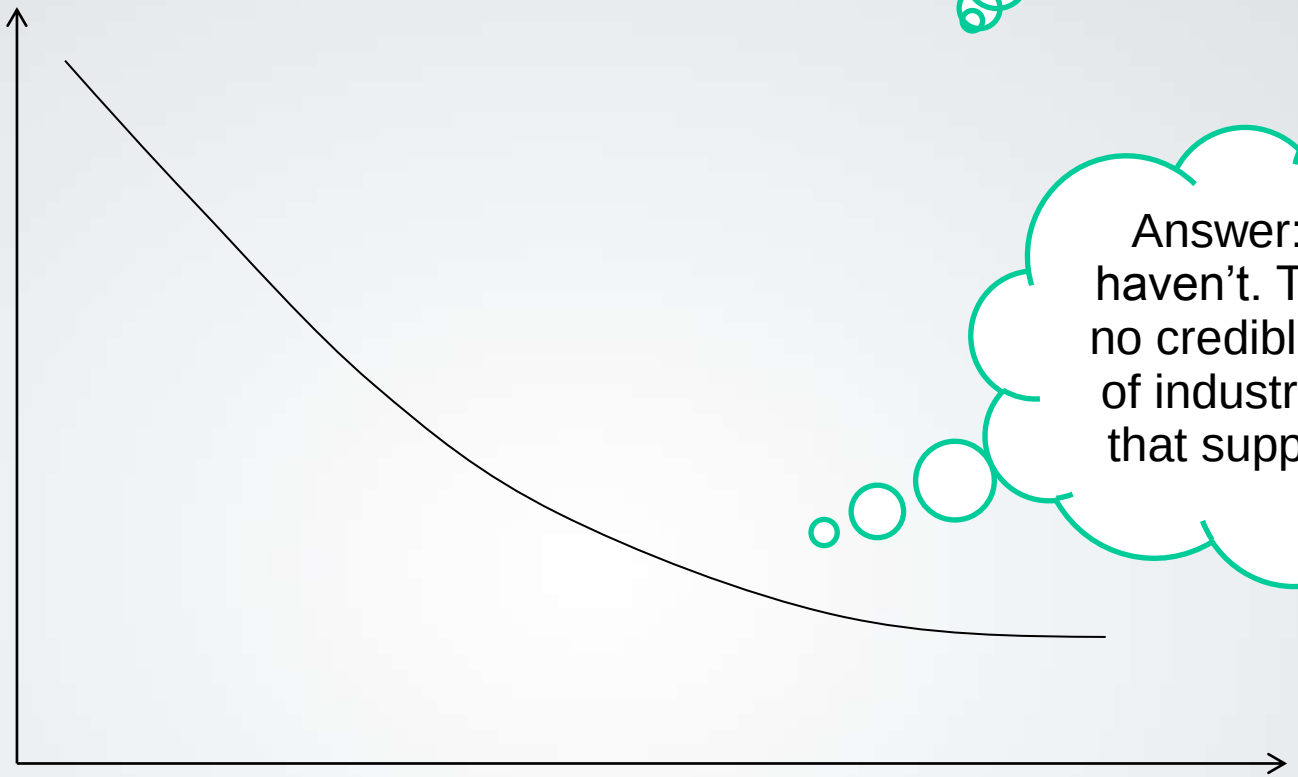
Twitter: @MsAsti

http://www.justgiving.com/Astrid-Byro

© Codemanship Ltd 2011

"Where have I seen this graph before?"

Relative Development Effort (person-hours)



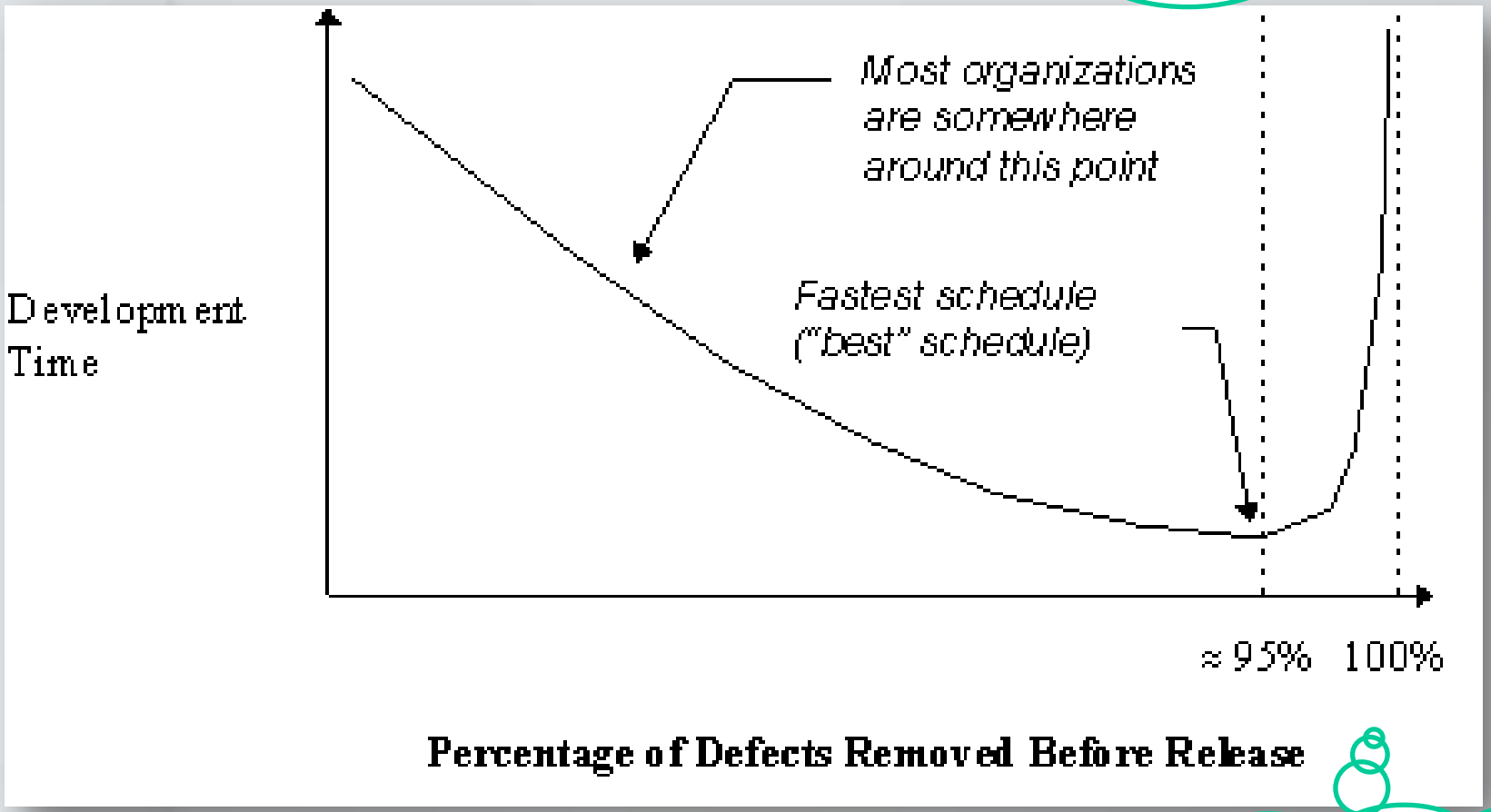
Answer: You haven't. There's no credible body of industry data that supports it

Defects/KLOC

© Codemanship Ltd 2011

codemanship

What data we do have paints a compelling picture of the opposite relationship between quality and speed



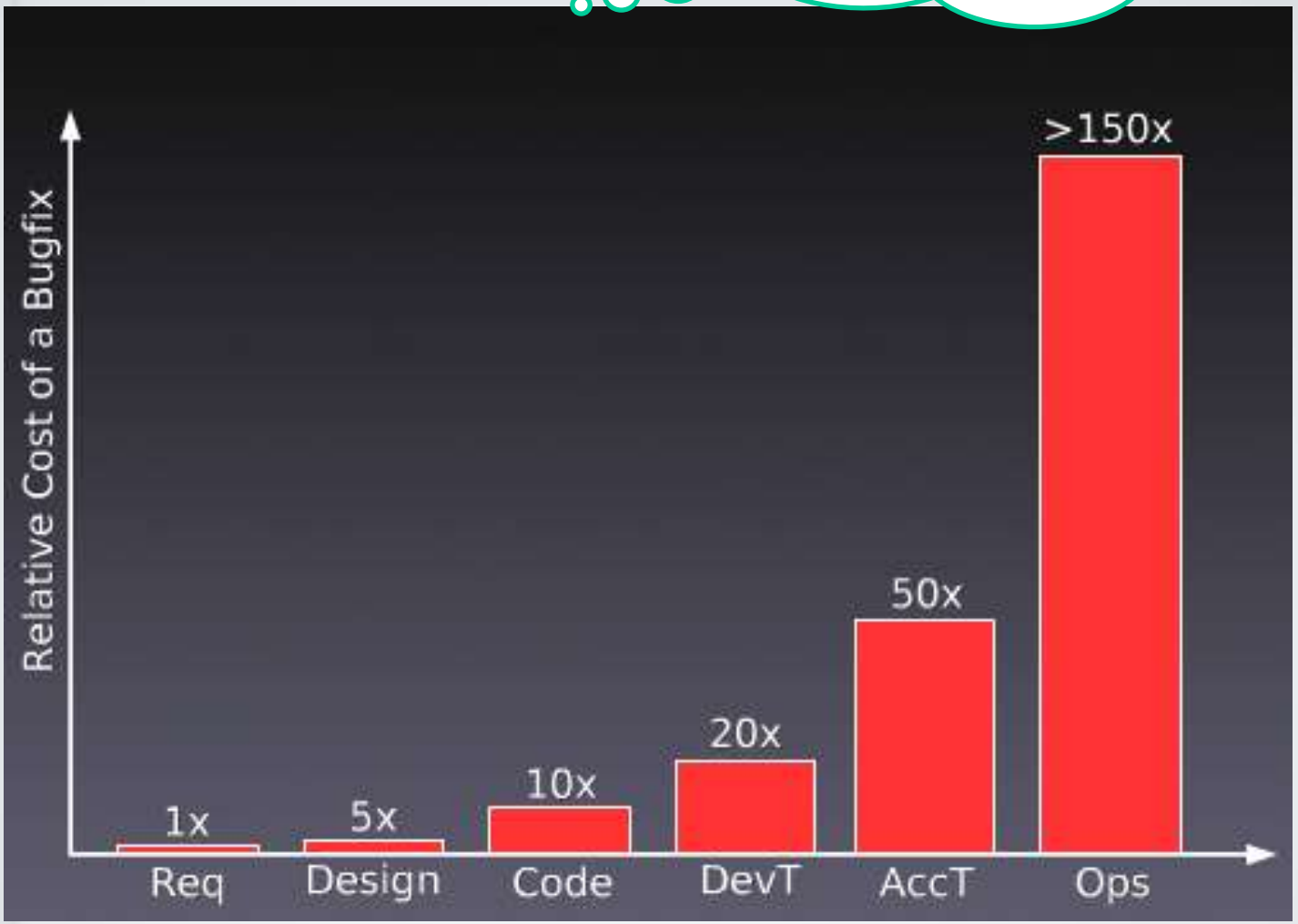
Steve McConnell, Software Quality At Top Speed, 1996

When considering whether to put more effort into quality or less, you need to know what side of this curve you're on. 99%+ of us are to the left of it

© Codemanship Ltd 2011

codemanship

Again, we have a mountain of data that suggests that time devoted to catching problems earlier easily pays for itself in time saved later

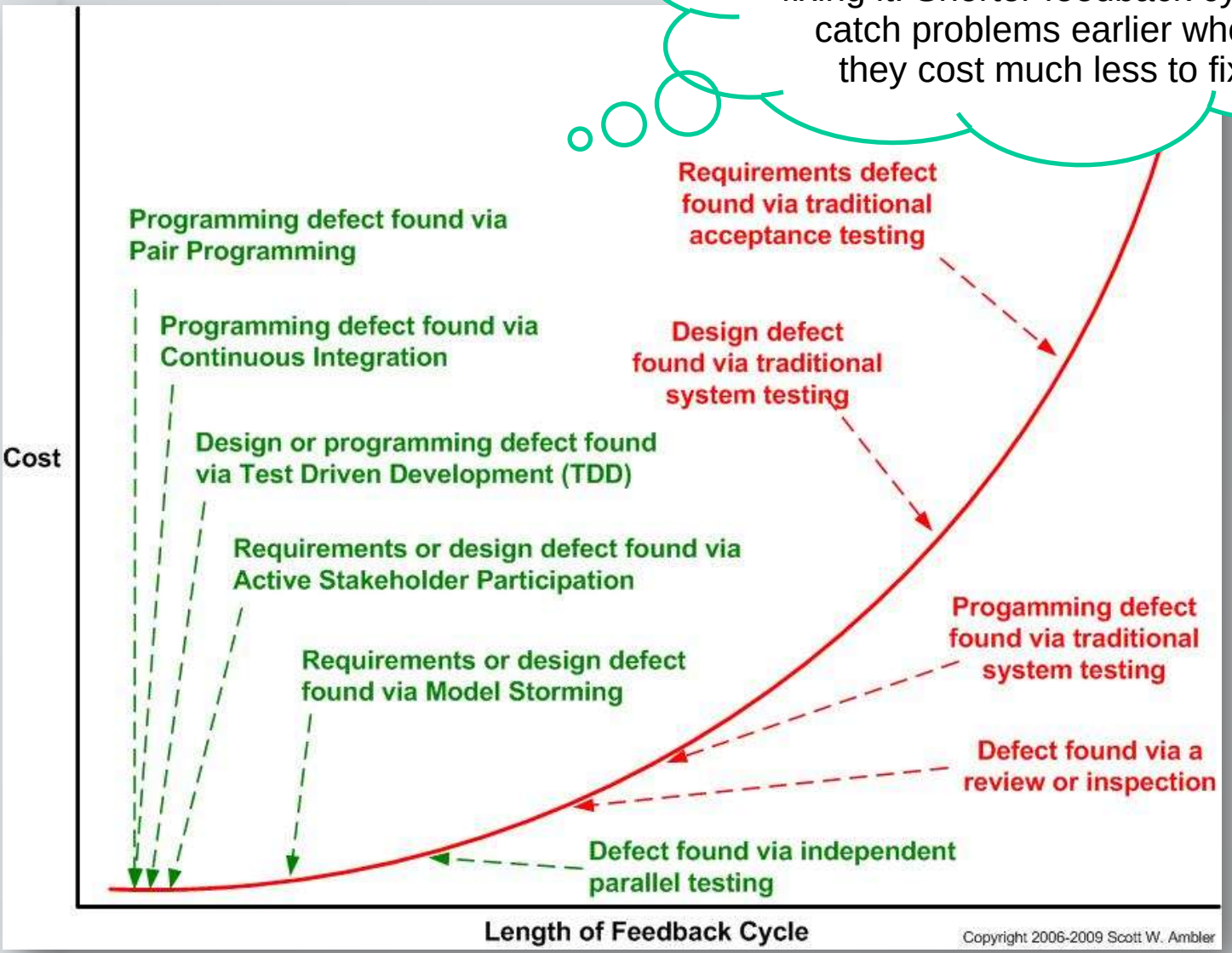


Barry Boehm, 2007

© Codemanship Ltd 2011

codemanship

What makes a bug more expensive to fix is the length of the feedback cycle involved in fixing it. Shorter feedback cycles catch problems earlier when they cost much less to fix

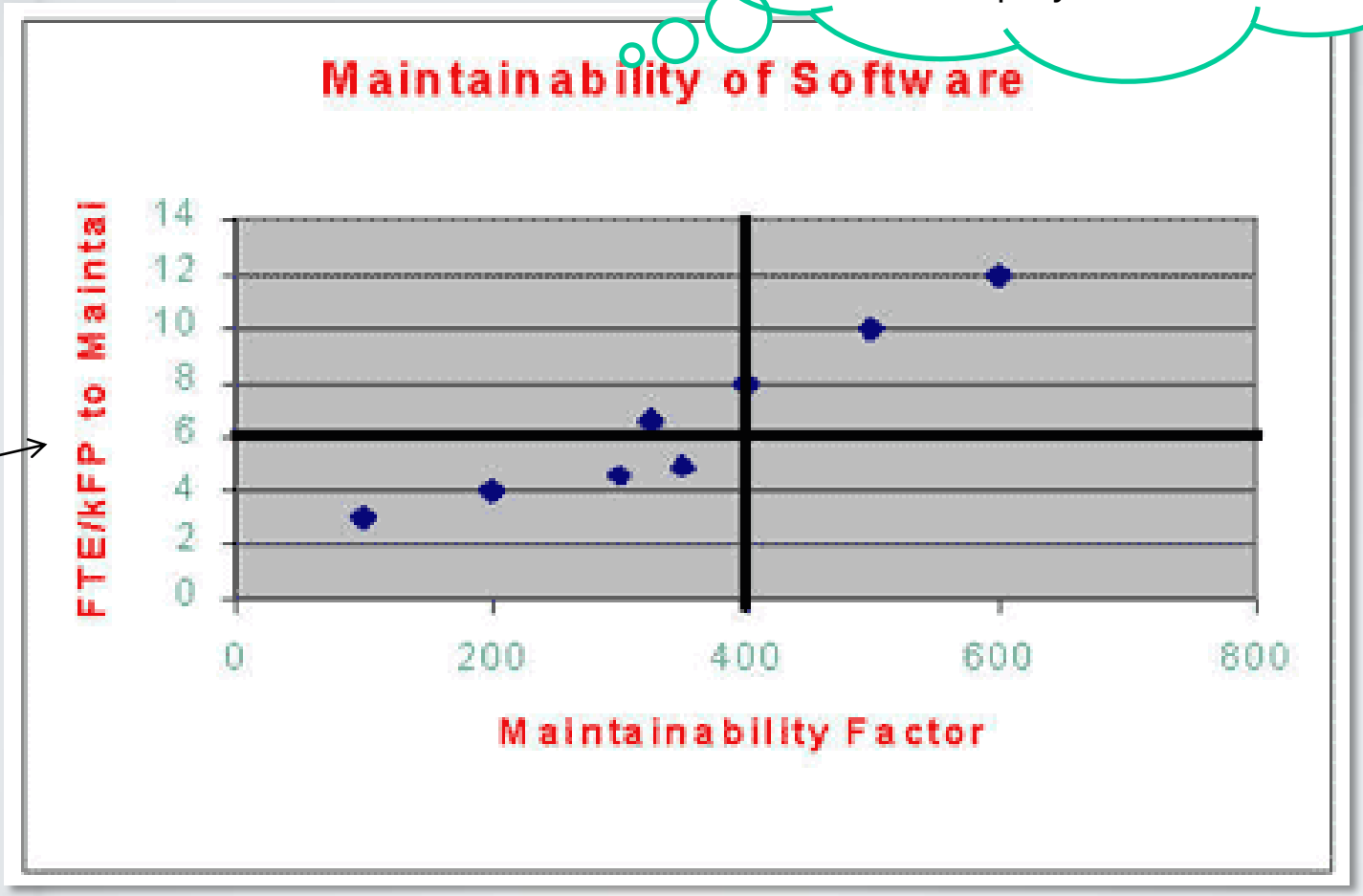


http://www.ambysoft.com/essays/whyAgileWorksFeedback.html

© Codemanship Ltd 2011

codemanship

Bugs aren't the only problems that cost us more later. Time invested early in making code easier to maintain can also pay big dividends as the code rapidly evolves.



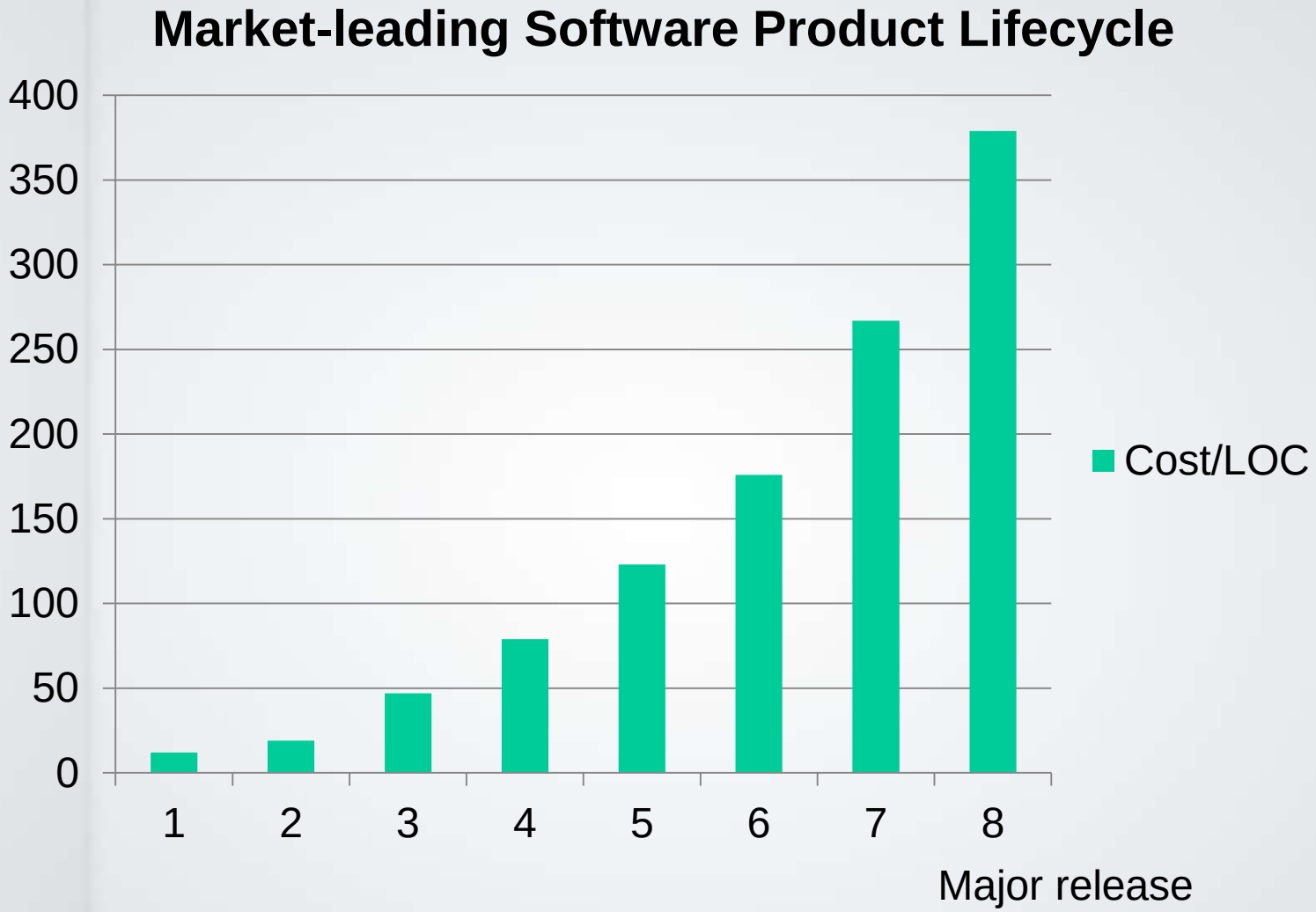
Full Team Effort to maintain

http://www.lalcrest.co.uk/cost.php

© Codemanship Ltd 2011

codemanship

A leading software company found that by the 8th version of their only product, a line of code cost 20x as much to write/change

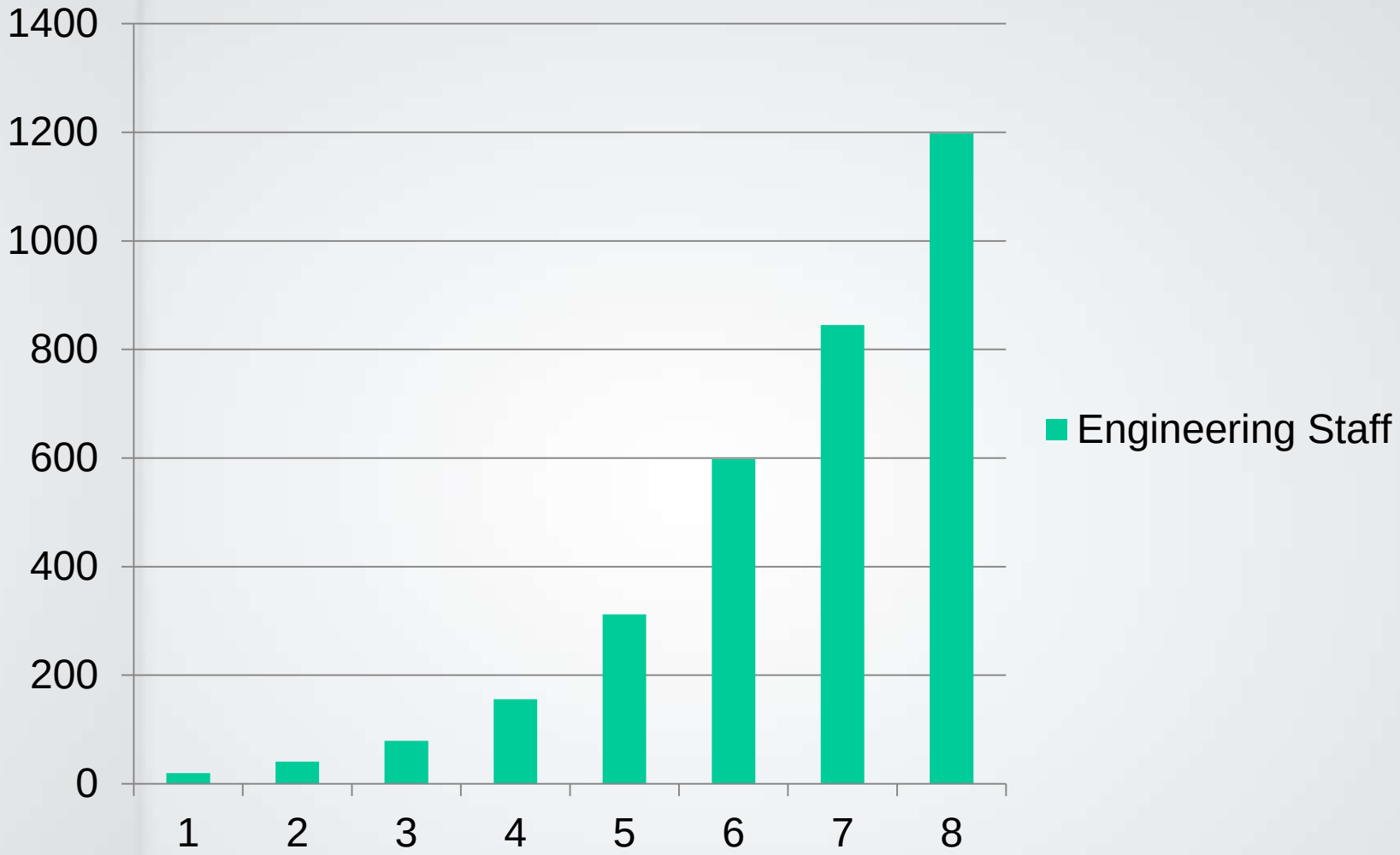


© Codemanship Ltd 2011

codemanship

Engineering staffing costs spiralled as they hired more and more people to achieve less and less

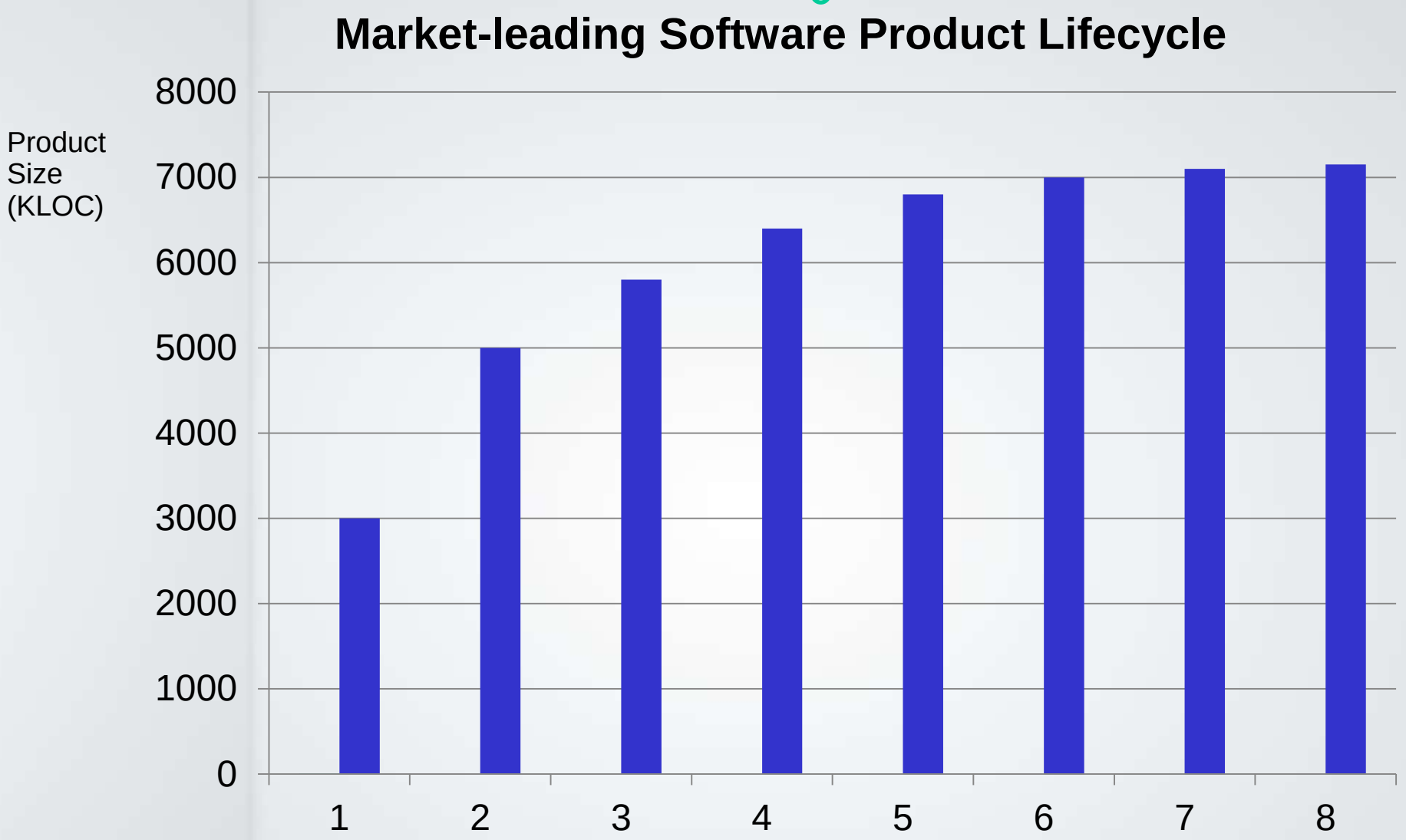
Market-leading Software Product Lifecycle



© Codemanship Ltd 2011

codemanship

The evolution of the product slowed to a snail's pace, and today they are lagging far behind their competitors



© Codemanship Ltd 2011

codemanship

Try this thought experiment: imagine two teams A and B who compete to guess a 4-digit number. Team A guesses all 4 digits at a time, team B guesses one digit at a time. Worst case, team A may need 10,000 guesses, but team B would need max 40 guesses. Which team would you bet on?

Now let's make team A 10x as "productive" as team B: for every guess team B gets, team A get 10. Which team would you bet on now?

© Codemanship Ltd 2011

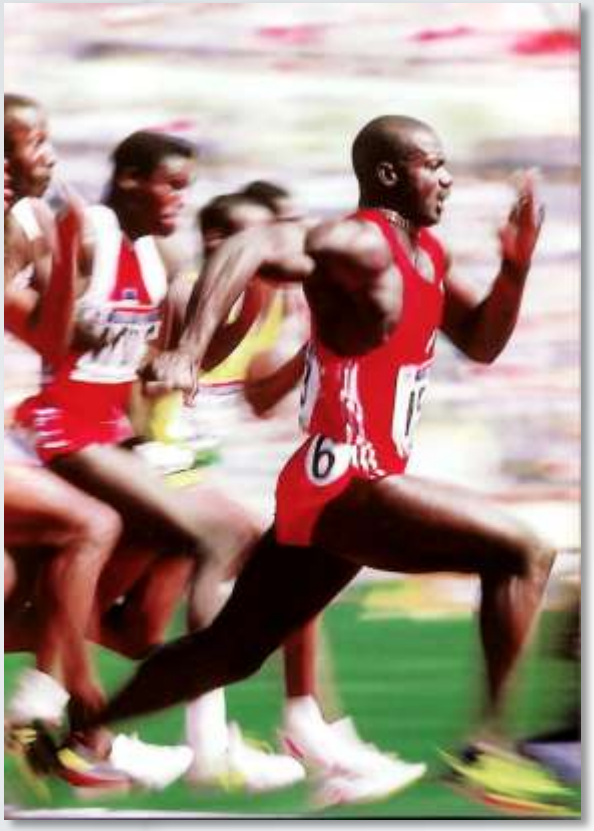
“It is not how fast we deliver that defines the winners and losers, it is how fast we learn from what we deliver”

Me, Whenever

© Codemanship Ltd 2011

codemanship

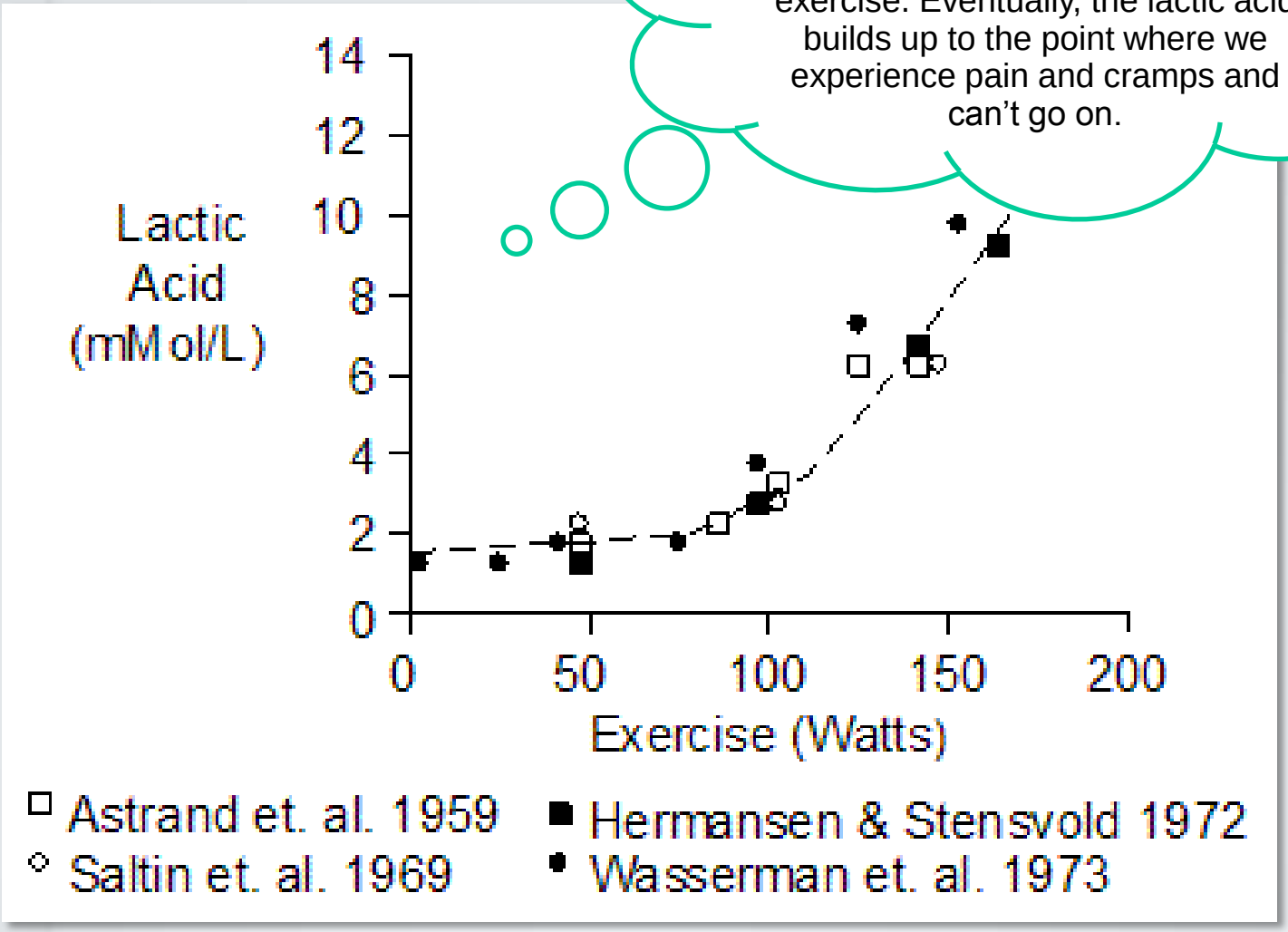
Consider the different way we would start a marathon as opposed to a 100m sprint



© Codemanship Ltd 2011

codemanship

With the higher exertion of a sprint, lactic acid builds up faster in our muscles – until we can't take in oxygen fast enough to get rid of the lactic acid. We call this **anaerobic** exercise. Eventually, the lactic acid builds up to the point where we experience pain and cramps and can't go on.

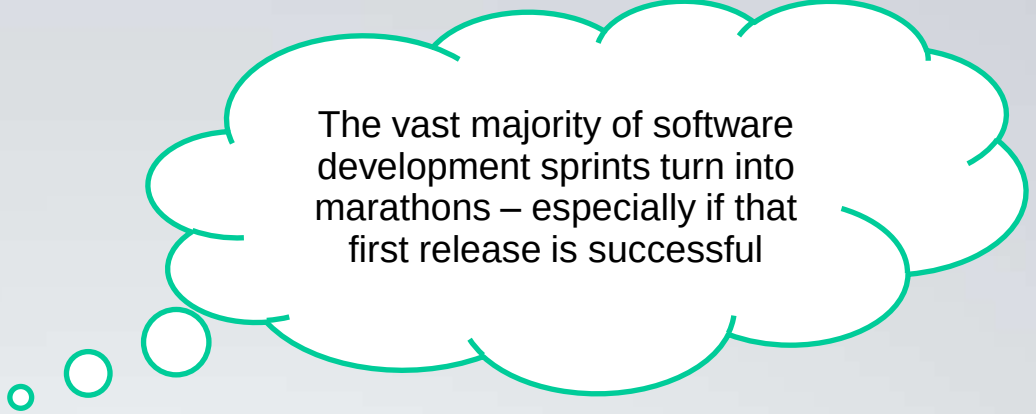


© Codemanship Ltd 2011

codemanship

When we run a marathon, we must not exert ourselves to the point where we're doing anaerobic exercise, as it's not sustainable. Start a marathon at a sprint, you may take an early lead, but you'll end up being carried off on a stretcher.

© Codemanship Ltd 2011



Anaerobic Software Development

When teams write code at an unsustainable pace, code smells build up faster, making progress increasingly difficult and painful

© Codemanship Ltd 2011

codemanship

This is a software development marathon being run right now. The winners in this race will be decided by who sets the most sustainable pace of innovation – not who sets the fastest initial pace . The winner will not out-deliver their competition. They will **out-learn** them.

BBC iPlayer

YouTube

media

vimeo

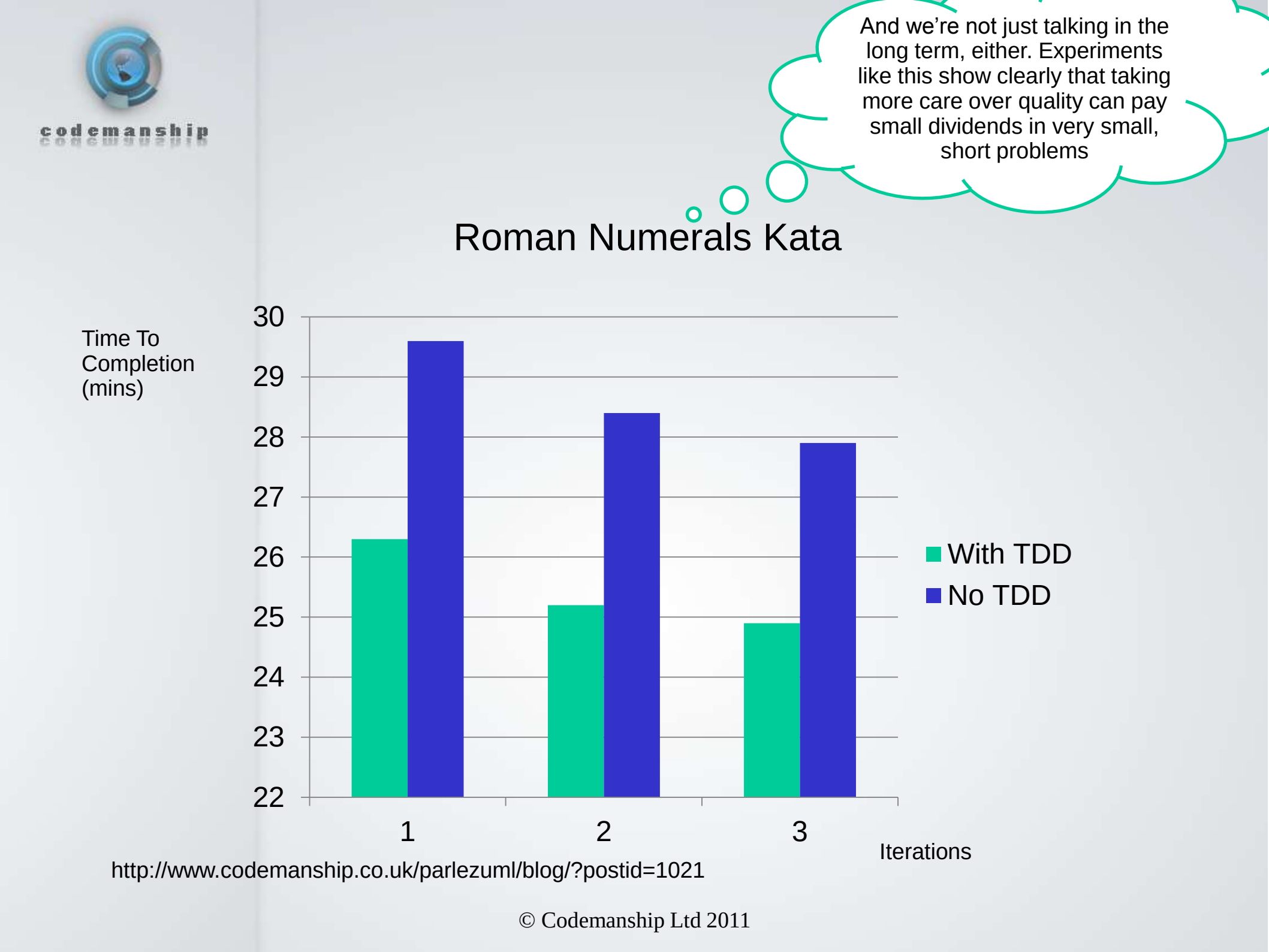
sky

Joost™

hulu™

4 on demand

© Codemanship Ltd 2011


codemanship
And we're not just talking in the long term, either. Experiments like this show clearly that taking more care over quality can pay small dividends in very small, short problems
Roman Numerals Kata
Time To Completion (mins)
30
29
28
27
26
25
24
23
22
1
2
3
Iterations
With TDD
No TDD
http://www.codemanship.co.uk/parlezuml/blog/?postid=1021
© Codemanship Ltd 2011

codemanship

Do not be seduced by the illusion of “done”

Of course, one way to deliver earlier without taking more care is to widen the goal posts – e.g., testing less thoroughly, or just ignoring problems

© Codemanship Ltd 2011

codemanship

"The financial impact of software quality problems is in no way diminished by our ability to ignore them"

Me, Just Now

The problem with this strategy is that the business consequences of quality issues have no respect for your desire to ignore them. The fiends!

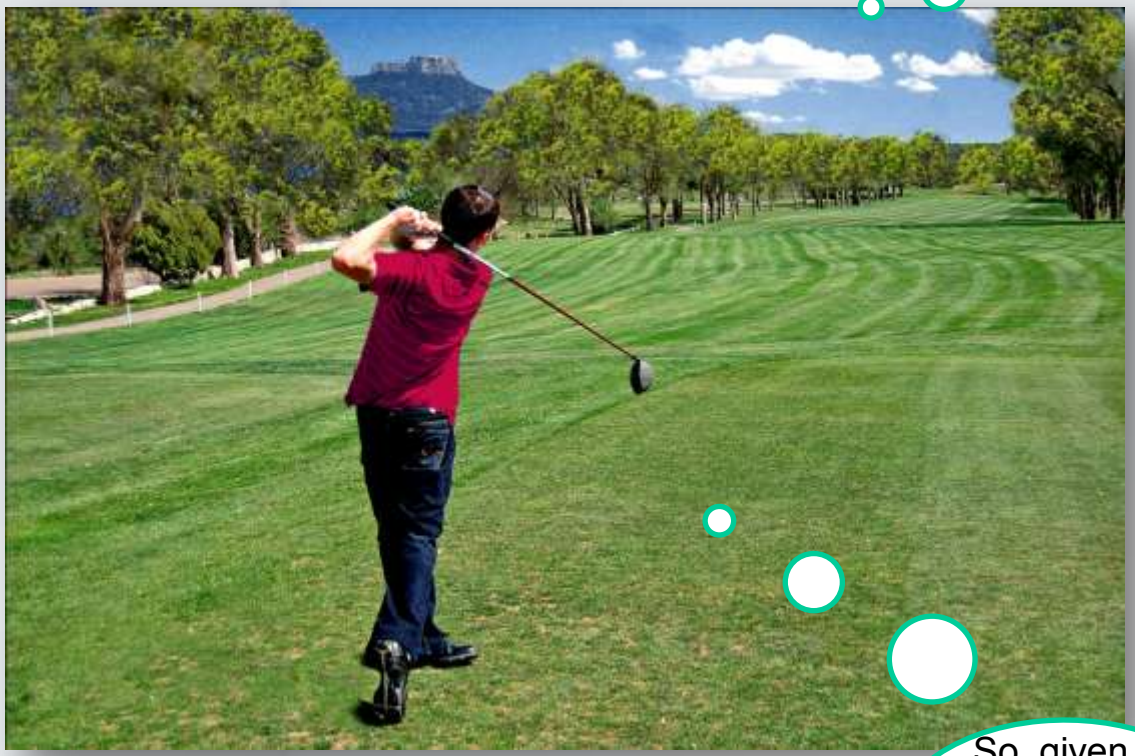
© Codemanship Ltd 2011

codemanship

© Codemanship Ltd 2011

codemanship

Well, what of it? Let's just accept that we're almost certainly going to need to take a few swings at it before we get the ball in the hole. NOBODY gets it right first time.



I would say it's a given

So, given that we're – to whatever extent – building the wrong thing, and we can only learn that by delivering *something*, and given that it would take no more time or money to build it right, and that we can be pretty sure that we'll need to evolve our first release further and that our ability to do that will depend on the quality of the code…

© Codemanship Ltd 2011

codemanship

…why would you choose to take *less* care over quality?

© Codemanship Ltd 2011

codemanship

Ah, but what about…

facebook

twitter

bebo

flickr

But what about all these massively successful start-ups who we know hacked out their software during all-night pizza-and-caffeine-fuelled code orgies?

Surely we should just get *anything* to market quickly by any and all means, and then we can fix all the problems with all that lovely Web 2.0 bubble money that will come flooding in?

© Codemanship Ltd 2011

codemanship

And what about my Great Aunt Doris, who smoked 60 a day and lived to be 103?

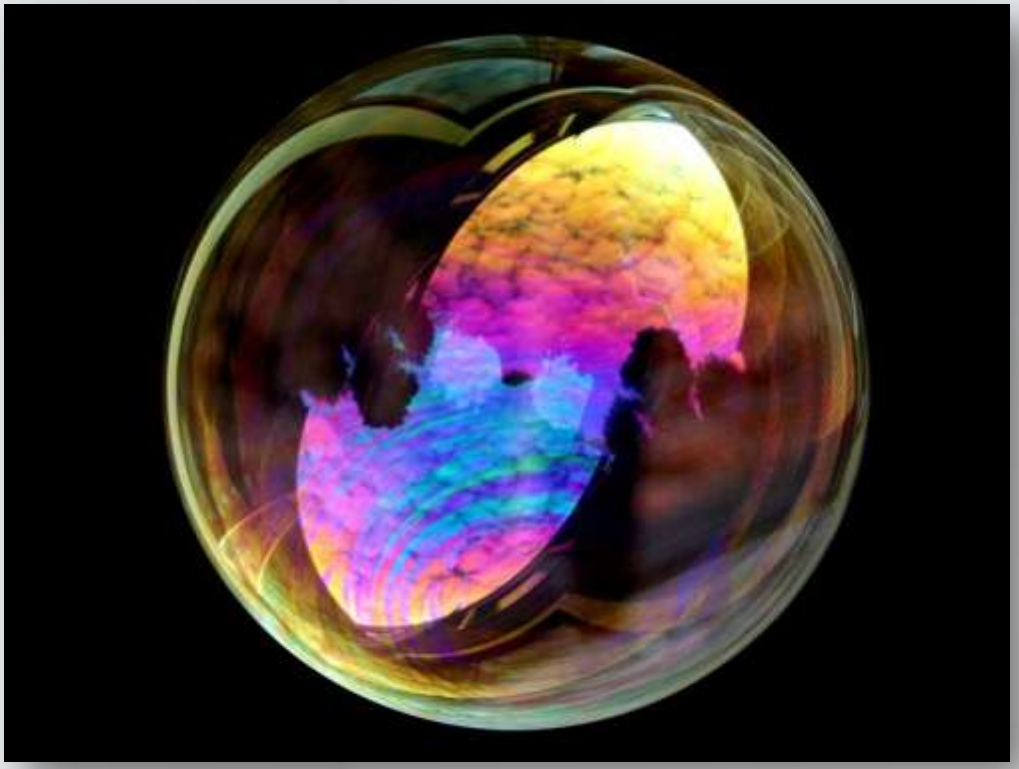
© Codemanship Ltd 2011

codemanship

Or the guy who bet his entire life savings on one hand of poker and became a millionaire?

© Codemanship Ltd 2011

codemanship



Successes like Facebook, Twitter etc are **statistical aberrations**, distorted even more by the Web 2.0 bubble, which enables them to buy their way out of hugely expensive early mistakes with other people's money

Mistakes that kill less lucky companies all the time!

© Codemanship Ltd 2011

So, In Summary

- Business models learned from statistical aberrations are Fool's Gold
- For the overwhelming majority, we are on the left of McConnell's curve, and **better = quicker**
- Get to market sooner by doing a better job of something simpler
- Expect to have to play more than one hand before you win anything
- When it comes to discussing this topic with managers and customers, **grow a pair**

© Codemanship Ltd 2011

codemanship

Because start-ups must eventually become stay-ups

© Codemanship Ltd 2011

www.codemanship.com

@jasongorman

© Codemanship Ltd 2011