

Facts and Fallacies of Software Engineering

CT-SPIN, July 2005

Ben van der Merwe

Stephen Quirke



STRUCTURE

Part 1 – Management

Part 2 – Requirements &
Estimation

Part 3 – Coding & Testing



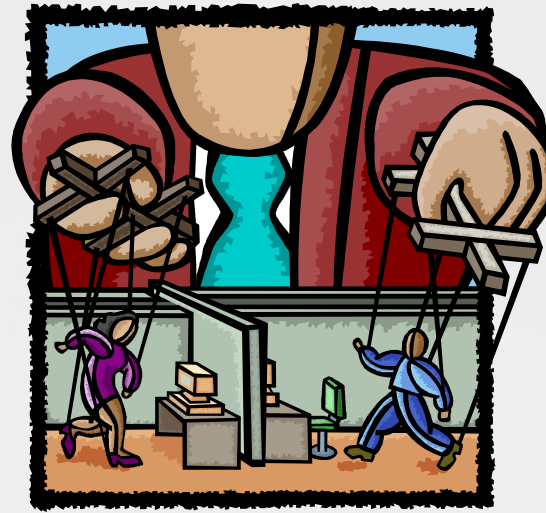
☆

Facts and Fallacies of Software Engineering



Robert L. Glass
Foreword by Alan M. Davis

Source: Facts and Fallacies of Software Engineering
Robert L. Glass, Addison-Wesley, 2003



Part 1 MANAGEMENT





Management (1/3)

“The most important factor in software work is the quality of the programmers.”

Management (1/3)

“The most important factor in software work is the quality of the programmers.”

Glass – FACT 1

- Estimates for productivity variances between best and worst programmers range from 5 to 28 fold.
- Quote: “We don’t know how to identify the “best” people.”
- Key Issues:
 - Recruitment, Upskilling & Training
 - Environment and resources. One study showed that the best performers had 1.7 times more room, 2.6 times better performance (*Peopleware*, 1999)



Management (2/3)

“You can manage quality into a software product.”

Management (2/3)

“You can manage quality into a software product.”

Glass – FALLACY 2

- Key Question: “Who’s responsibility is quality?”
- Chief enemy of quality is schedule pressure – which is most often applied by management.
- Discussion points:
 - If the fallacy is that it is management’s job, should management remove obstacles, provide resources and then get of the way?
 - If creative people are intrinsically motivated, how should management then approach quality?



Management (3/3)

“Hype (about tools and techniques) is the plague on the house of software.”

Management (3/3)

“Hype (about tools and techniques) is the plague on the house of software.”

Glass – FACT 5

- It has been a long time since the last true breakthrough
- There is very little supporting evidence (studies) for most claims
- The highest payoff is for processes that improve reuse (10% to 35%)
- Related fact: “Software developers talk a lot about tools but seldom use them.”
- Discussion point: hype leads to sales. Who makes money selling you the best programmers?



Discussion - Management

“The most important factor in software work is the quality of the programmers.”

“You can manage quality into a software product.”

“Hype (about tools and techniques) is the plague on the house of software.”



Part 2 REQUIREMENTS & ESTIMATION





Requirements & Estimation (1/3)

“Software estimation usually occurs at the wrong time.”

Requirements & Estimation (1/3)

“Software estimation usually occurs at the wrong time.”

Glass – FACT 9

- Key question: “What causes run-away projects?”
 - Answer: unstable requirements and poor estimation.
- The first phase of a project is to determine what the requirements are.
 - Why do we make estimates before we understand what the problem is that we are trying to solve?
- Quote: “You don’t understand ... we’ve already announced the release date.”
- Related fact: “Estimation is done by the wrong people.” (Marketers and Management)



Requirements & Estimation (2/3)

“Explicit requirements
"explode" as implicit
(design) requirements for a
solution evolve.”

Requirements & Estimation (2/3)

“Explicit requirements "explode" as implicit (design) requirements for a solution evolve.”

Glass – FACT 26

- When things gets messy..
- Growth factor of up to 50
- How do we manage requirements when they are continuously being discovered?
- Quote: “Traceability has proven to be an illusive Grail.”
- Discussion points:
 - What are the implications for estimation?
 - What is the use of traceability in the first place?



Requirements & Estimation (3/3)

“There is a disconnect between software management and their programmers.”

Requirements & Estimation (3/3)

“There is a disconnect between software management and their programmers.”

Glass – FACT 13

- Many problems have their origin at the very beginning – ill-defined scope, unrealistic schedule, *missing requirements*.
- Engineers see problems coming long before management – 72% of the time.
- Ultimate disconnect: knowledge of problems not passed on to management.
- Solutions?
 - Quote: “Very strong correlation between level of productivity and feeling of control.”



Discussion – Requirements & Estimation

“Software estimation usually occurs at the wrong time.”

“Explicit requirements "explode" as implicit (design) requirements for a solution evolve.”

“There is a disconnect between software management and their programmers.”



Round 3 CODING & TESTING





Coding & Testing (1/3)

“Programming can and should be egoless.”

Coding & Testing (1/3)

“Programming can and should be egoless.”

Glass – FALLACY 3

- *Psychology of Computer Programming* (Weinberg 1971)
- Glass: Fundamental human trait, natural thing
- Quote: “Contemplate the notion of an egoless manager”
- Discussion points:
 - How to deal with the defensive programmer?
 - Loss of objectivity, e.g. business vs. technical case
 - How do we approach appraisals and reviews? No ego no responsibility – it’s just the team?



Coding & Testing (2/3)

“Errors tend to cluster.”

Coding & Testing (2/3)

“Errors tend to cluster.”

Glass – FACT 49

- Best way to know where errors are, is to look where they have been found in the past.
- Typical quote: “About 80% of the errors come from 20% of the modules and about half the modules are error-free.”
- Problem areas: programmer ability (experience, knowledge), complexity
- Discussion points:
 - Are we keeping statistics on our errors?
 - How do you deal with an error-prone programmer?



Coding & Testing (3/3)

“Software developers talk a lot about tools, but seldom use them.”

Coding & Testing (3/3)

“Software developers talk a lot about tools, but seldom use them.”

Glass – FACT 25

- Causes:
 - Learning curve, initial productivity loss
 - Programmers going back to what they know
- “Not Invented Here” phenomenon? Or do we simply don’t have the time?
- Lack of standard toolchest.
- Intangible benefits
- Open source – a tool for every taste?



Discussion – Coding & Testing

“Programming can and should be egoless.”

“Errors tend to cluster.”

“Software developers talk a lot about tools, but seldom use them.”

More information, search for book:
<http://print.google.com>

**Next CT-SPIN
Meeting:
17 August 2005**

<http://www.spin.org.za>

