[simula . research laboratory]

How we know what isn't so

Common myths in daily life and software development

Why do we believe in myths?

Magne Jørgensen
Simula Research Laboratory

The paper clip was invented by a Norwegian

[simula . research laboratory]

Most communication is non-verbal

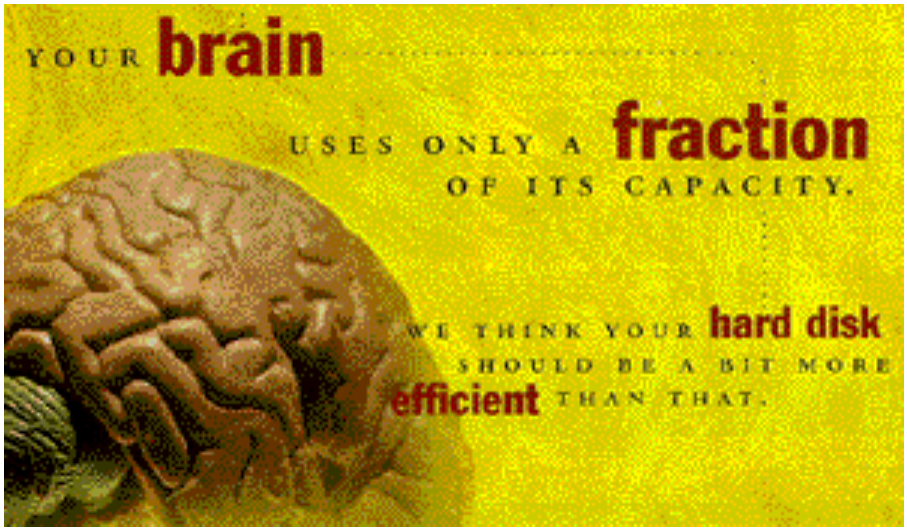
[simula . research laboratory]

Most of the heat is lost through the head

[simula . research laboratory]

Children get more hyper-active with sugar

We use only 10% of our brain



Short men are more aggressive (The Napoleon Complex)

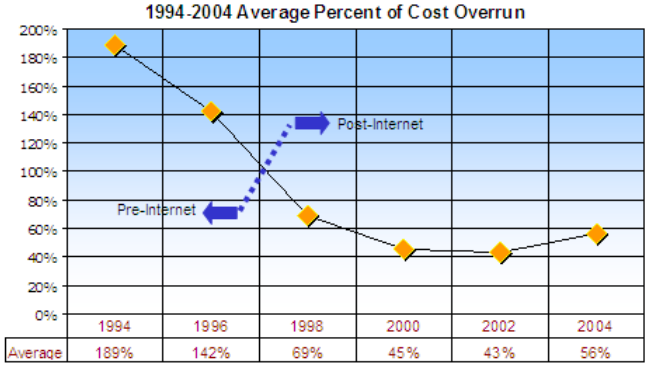
[simula . research laboratory]

There is a software crisis (Chaos Report: 189% average cost overrun)

THE STANDISH GROUP

Cost Overruns

1994-2004 Average Percent of Cost Overrun



Year: 2004, Source: CHAOS Database: CHAOS surveys conducted from 1994 - 2004.
Results: show average percent of cost above their original estimate.

[simula . research laboratory]

Why did we believe in this study?

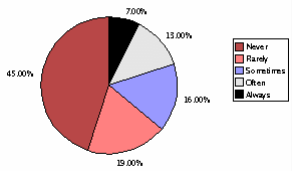
- Most other studies show cost overruns of about 30-40%. Nobody I know work in organizations where a project on average costs 2,9 times more than estimated
- The selection process is not documented (probably they were asking for ”problem projects”)
- They confuse 89% overrun with 189% of original estimate
- **Possible reasons for the acceptance of the results of this very low quality study**
  - We do not check the study quality, especially not the selection process
  - It fits our purpose, e.g., to sell consultancy services or to defend that there are people even worse than me/us to estimate cost
  - Fits the feeling that estimation is difficult
  - We know of examples with that larger cost overrun (confirmation bias)

Jim Johnson
Standish Group

Standish Group - again

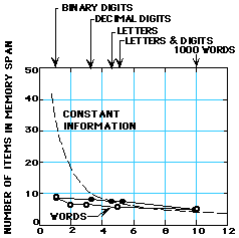
- **45% of the features are never used**
  - Much used in the agile community to argue against the waterfall model and in favor of agile or incremental methods
  - Original study not available. Reference to a stsudy presented by Jim Johnson on XP 2002
  - According to www.agilemodeling.com/essays/examiningBRUF.htm the study was as follows:
    - **The Standish Group looked at a subset of traditional teams which eventually delivered into production and asked the question, “Of the functionality which was delivered, how much of it was actually used?”**
  - How would you interpret the question? Is it about features never used by anybody, not even once?
  - Who did they ask? The developers? How good will their knowledge be about the users? Is it based on measurement?
  - Does the results sound reasonable? Is this what **typically** happens?

Features and Function Usage



”Magic Seven, Plus or Minus Two”

- **The optimum number of elements (e.g., in user interfaces of routine calls) is 7 plus/minus two).**
  - *“One guideline that needs closer attention is the "Magic Seven, Plus or Minus Two" (7±2) rule of thumb. This principle has often been applied to determine the number of items in a navigation menu on a web page. It arose to satisfy a tactical need to make quick design choices and to objectively justify navigation to site stakeholders.”*
  - Based on studies by Miller on cognitive capacity, i.e., the **maximum** we are able to recognise visually without dividing into groups or store an recall in certaint contexts)
  - Has hardly anything to do with the **optimum** number.
  - The results are clearly misused. Probably due to a wish for simple rules in complex isssues.



Exercises

How do you think the following myths were created and spread? (some of them are more in the category ”meaningless claims” or ”outrageous simplifications”

- **“Adding manpower to a late software project makes it later"** *The Mythical Man-Month* **(Brooks’s lov)**
  - Described even by Brooks himself as an "outrageous oversimplification“. This myth has been used to give support for useless estimation models.

- **The difference between the best and the worst programmer is 10:1**
  - Based on a study from 1968 with experienced developers and one task
  - Clearly, the ratio is as large as you want it to be, just add incompetent, inexperienced programmers

Exercises …

- **"Happy teams are productive teams"**
  - Very useful in order to convince an employee to invest more mony in happy developers
  - Very little documentation on "happy-productive", e.g., because it is difficult to measure.
  - Results are more in favor of satisfaction with task, co-workers and management.
  - "Happy teams" seem to have both the highest and the lowest productivity.
- **Brainstorming should be done in groups**
  - Evidence in favor of the opposite (see especially the classic studies by Zajonk)
  - BUT, information/ideas should be shared and discussed in groups
- **The largest part of software development is mainteance (or testing or ….)**
  - Lack of precision means that there is not much information is such statements

Exercise

- **The cost of changes (or correction of errors) increases with a factor of 10 with each phase**
  - Many varints of this claim. The mildest is that it always costs more to correct an error in later phases.
- Alistair Coburn writes at www.xprogramming.com/xpmag/cost_of_change.htm
  - *The first thing is to establish that it really does cost x10 or similar to detect and handle mistakes as we proceed across r, p, t, f, (i.e. exponential) and that XP really is well suited to handle that exponential growth in cost. It is not the case that the x10 doesn't exist, or that XP nullifies it. XP deals with its existence, is "well aware of it" to use my preferred anthropomorphizing locution.*
  - *r=requirements; p=programming, t=test, f=field (10*10*10=1000 ganger høyere i produksjon enn i krav!)*
- Where is evidence in support of ten times as much? If this is true, it would beneficial to find all errors early, regardless of how much it costs to find the last error.

Why and how are myths created?

- Meeting a need or desire. We want them to be true and don't look for disconfirming evidence.
- Lack of precision/misunderstood research.
- We are more conserned about whether something sounds correct, than about asking what this really means.
- Self-fulfilling claims (we see it because we believe it).
- It is easier to find confirming and than to find representative evidence.
- Political and business-related reasons. Deliberate creation of myths.
  - Repetition.
  - Presented by authorities.
- To understand is to accept. De-accepting is more difficult (see next study)

Creation of a myth: Risk willing programmers are better

1. **Based on your experience, do you think that risk willing programmers are better than risk averse programmers?** 1 (strongly agree) – 10 (strongly disagree)
2. **Evaluation 1**: Group A received information about a confirming study and presented one own argument in support of risk willingness. Group B, the opposite.
   ➔ Group A: 3,3 - Group B: 5,4
3. **Evaluation 2**: Both groups received the information that the study was there to mislead them and were asked to update their judgment.
   ➔ Group A: 3,5 - Group B: 5,0
4. **Evaluation 3**: Two weeks later.
   ➔ Group A: 3,5 - Group B: 4,9

Things will never go “back to normal” when we make a decision (e.g. started believe something)

http://www.ted.com/talks/lang/eng/dan_gilbert_asks_why_are_we_happy.html

How to become a myth buster (Evidence-Based Software Engineering)

1. A
   -
2. D ort
   th
   - ve
3. S e
   by
   -
4. S