Keep it simpel

Jakob Majkilde

Estimering

version2

IT-NYHEDER | BLOGS | IT-JOB | COMMUNITY | EMNER

EMNER *Projektledelse* Se kommentarer (10)

1.500 gigantiske it-projekter undersøgt: Hvert sjette ender katastrofalt

It-projekter fejler 20 gange hyppigere end andre store projekter. Ny dansk forskning dokumenterer milliardtab for skatteborgere, investorer og virksomheder.

Af ***Christian Loiborg*** *Mandag, 22. august 2011 - 6:59*

Rejsekortet, Digital Tinglysning og Amanda er blandt de store it-skandaler, der har fået hug i medierne. Men den slags afsporede it-projekter er langt mere almindelige end tidligere antaget. Og det private erhvervsliv er ikke bedre end den offentlige sektor.

Et omfattende internationalt forskningsprojekt, med en dansk Oxford-professor i spidsen, viser nemlig, at et ud af seks it-projekter ender katastrofalt.

Resultatet for de mislykkede it-projekter er en tredobling af prisen og en forsinkelse på 70 procent. Den type projekter er 20 gange så sandsynlige at finde bla

Bent Flyvbje er professor på Univers y of Oxford i Engl

3. Reducer kompleksiteten

Sørg for at reducere kompleksiteten i projektet. Jo mere komplekst et projekt er, jo større er risikoen for, at projektet vil fejle. Reduceringen skal ske på alle måder, der er mulige. Kompleksiteten kan for eksempel mindskes ved at bruge allerede etableret teknologi, opdele projektet i selvstændige moduler og undgå låne-finansiering.

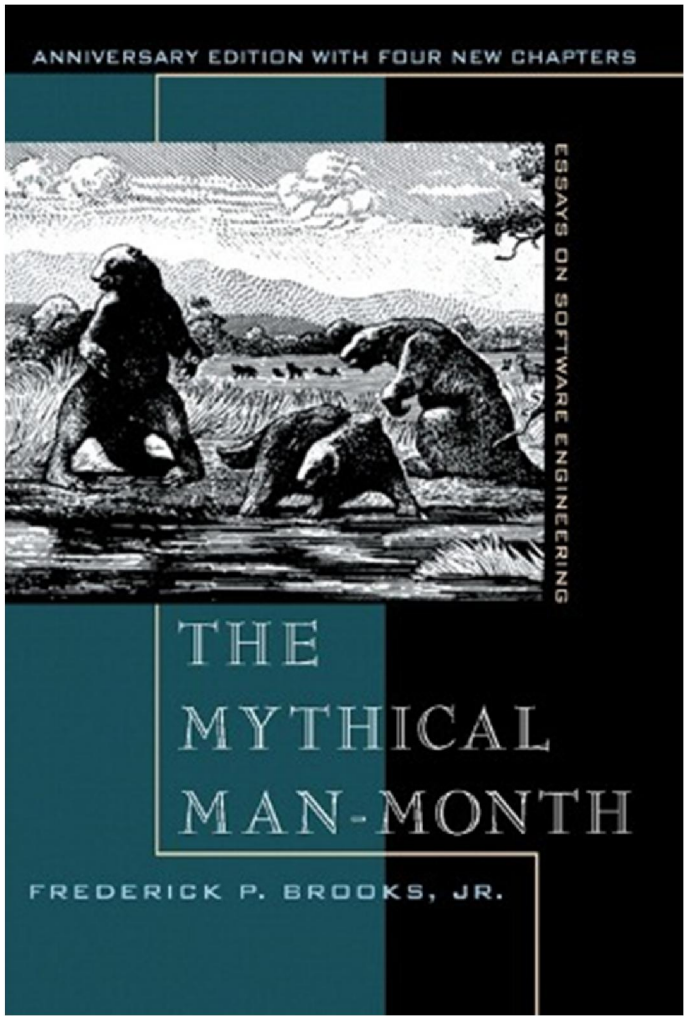
4. Reducer størrelsen af projektet

Størrelsen af projektet i tid er mere afgørende end den fysiske størrelse. Jo længere tid, et

er. Efter 30 måne

Successiv kalkulation?

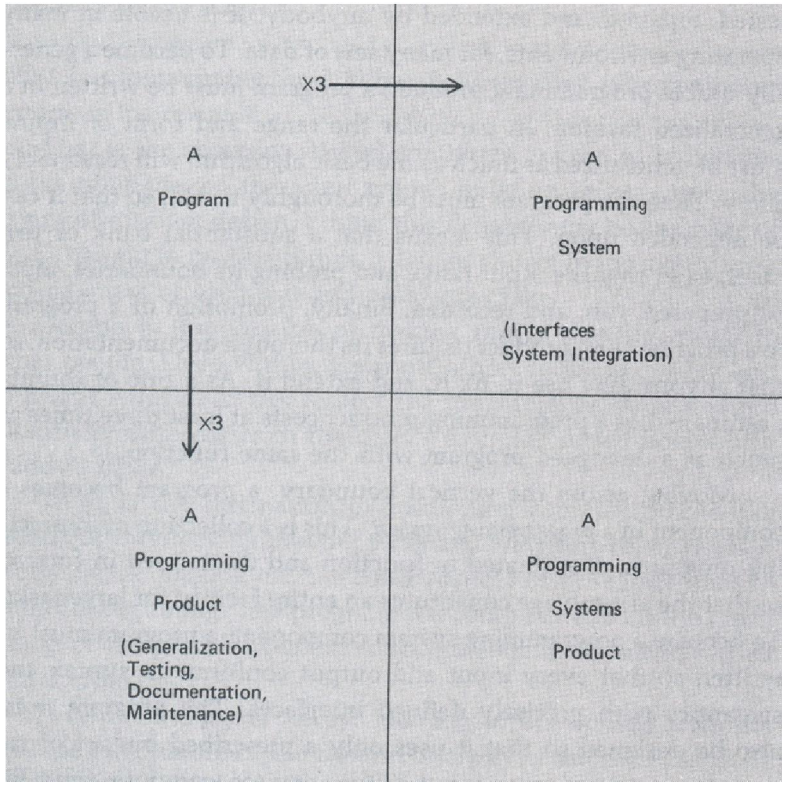
The Mythical Man-month



Bemærk: illustrationer i denne præsentation er taget fra bogen

Fra programmør til bruger = x9

- Program
  - Din kode
- Komponent i et system
  - 100% styr på interface
  - Optimeret og dokumenteret
  - Alle kombinationer er testet
  - Tager 3x ekstra tid
- Produkt
  - UI optimeret, input validering
  - Brugervejledning m.v.
  - Tager 3x ekstra tid



Estimering af store projekter

- Estimering
  - 1/3 Planning
  - 1/6 Coding
  - 1/4 Component test
  - 1/4 System test
- Brug så få personer muligt!

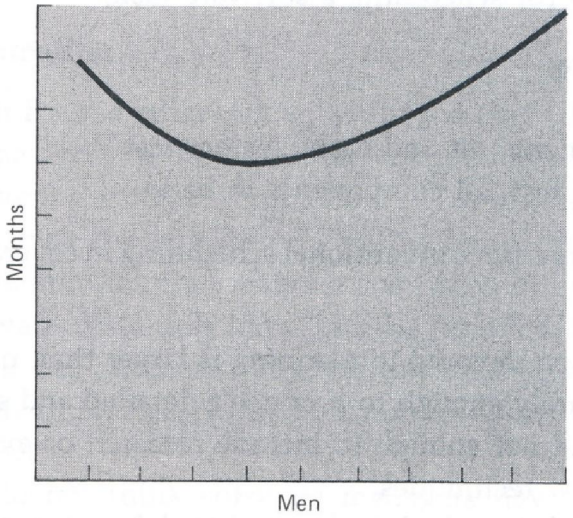
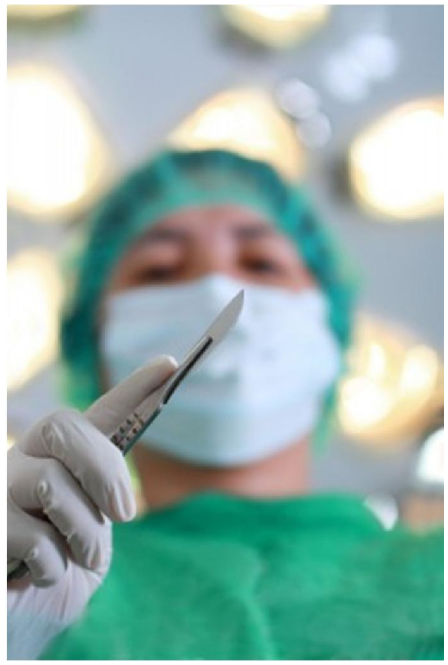


Fig. 2.4 Time versus number of workers—task with complex interrelationships

The Surgial Team

- Én programmør
  - Chefen
  - Koder, strukturerer og dokumenterer
- Copilot / Toolsmith
  - Sparing, Stand-in
  - Debugging, tools, components
- Sekretær
  - Administration
  - Dokumentation
- Tester



Småt er godt – simpelt er bedre

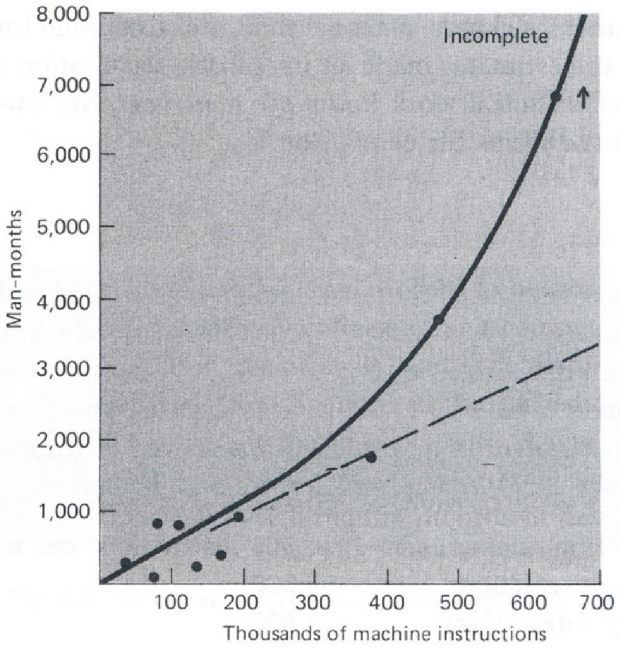


Fig. 8.1 Programming effort as a function of program size

```
Dim i As Integer
Print i / i
```

Use *"object"*

```
Print getValue( i ) / o.getDiv( i )
```

Code reuse

- Genbrug kode
  - Hurtigere udvikling
  - Koden er testet
  - Mindre kompleksitet
- Kræver små simple intuitive komponenter der er gennemtestet og dokumenteret
  - sinus(degrees) -> Bliver genbrugt
  - getProject(id, options) -> Bliver ikke genbrugt
- OOO er et must!
- Husk: Skal andre genbruge din kode = 3x tid

Reuse: Keep it simple

- Simple komponenter
  - Er nemmere at dokumenterer
  - Er ofte mere intuitive
- Low-level komponenter
  - Er nemmere at generalisere
- Dokumentation!
- Test!

”I keep having to bend things even on the fifth use of my own personal user-interface”

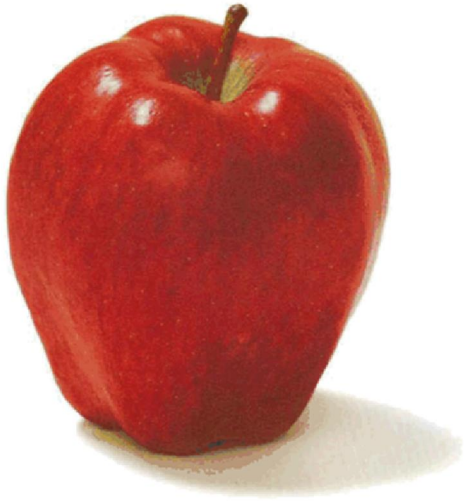
”Reuse it something that is far easier to say than to do. Doing it requires good design and very good documentation”

Den nemme løsning er aldrig den rigtige!

- Simplicity is the key (Apple / Google)
- The simplest explanation is most likely the correct one
- Simplicity is the ultimate sophistication.
  - Leonardo Da Vinci
- Complexity kills. It sucks the life out of developers, it makes products difficult to plan, build and test, it introduces security challenges and it causes end-user and administrator frustration.
  - Ray Ozzie
- Bad programmers worry about the code. Good programmers worry about data structures and their relationships.
  - Torvalds, Linus

MAY 5 1978

Simplicity is the ultimate sophistication.



Introducing Apple II, the personal computer.

Applikations udvikling

- Eksempel: Sync database / mini LEI
  - Generel import/export. Understøtter Notes, Text, Excel og kan nemt udvides med ADO el. lign. Både scheduleret og ad-hoc
  - Kompleks kode
  - Standard app og komponent = 9x udviklingstid
  - Fjernet fra kilden = mere abstrakt
  - Virker alligevel ikke – kunden har *altid* andre behov...

Komponent baseret udvikling

- LEGO
  - Små komponenter i stedet for applikationer
  - Brug dem til at bygge en custom app hos kunden
- Eksempel: Sync agent
  - Ny database med personer der synkroniseres med names.nsf via agent
  - Opdaterer kun hvis nødvendigt
  - Full log og mail ved fejl
  - Kan laves på 30 min.
    med komponenter

Opsummering

- Keep it simple
  - Simpel funktionalitet & simpelt UI: nemmere at dokumenterer, nemmere at bruge, mindre support
  - Simpel kode: færre fejl, mindre udvikling, bedre performance
  - Simpelt projekt: få personer, kortere udvikling, mindre overhead
- Hvis koden er kompleks – så lav det om!