Programvarukonstruktion: Lectures

Torbjörn Sonning
epltos@epl.ericsson.se

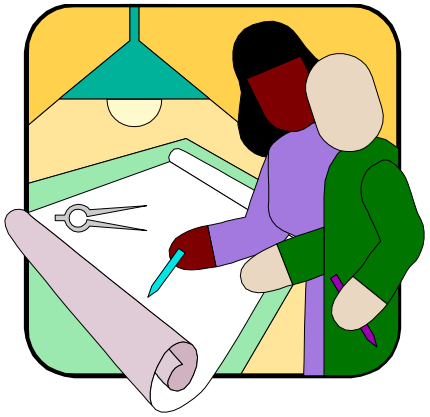
http://www.cs.umu.se/kurser/TDBB12/

Contents

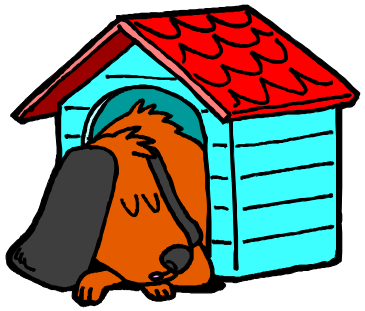
The Software Engineering Challenge

Programming is an art… isn't it?

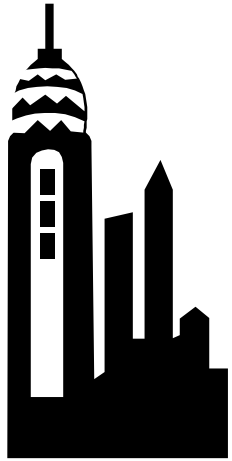
What are Software Engineers Constructing?



Dog houses?



Skyscrapers?



Cities?

Software “Dog Houses”

- One or a few programmers
- Multiple roles per person
- Ad hoc planning
- Completed within a few months
- Single customer
- Non-distributed program
- 5 - 100 kLOC
- Little reuse
- No maintenance

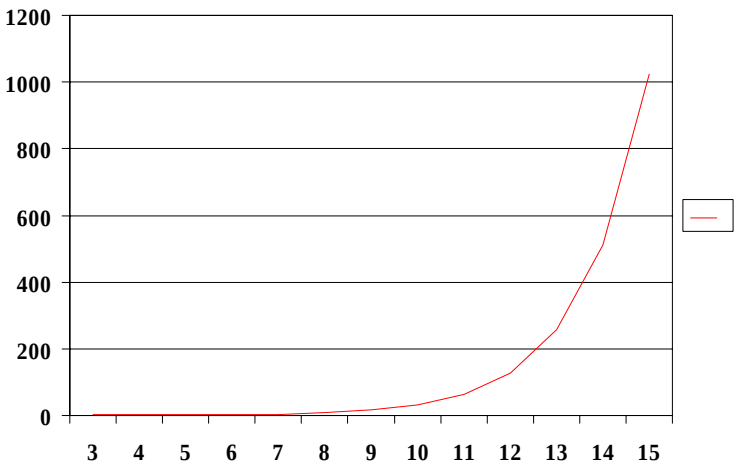
Software “Skyscrapers”

- 10-100 people. Project managers, architects, designers, programmers etc.
- Few roles per person
- Thorough planning
- Completed within a 6-18 months
- Typically single customer
- Distributed or non-distributed program
- 100 - 1000 kLOC
- Copy/paste reuse from already built skyscrapers
- Maintenance is the customers problem

Software “Cities”

- 100 - 1000 people. Product planners, project managers, architects, designers, programmers etc.
- One or two roles per person
- Thorough planning
- Completed within a 12-24 months
- General product, no specific customer
- Systems of many programs and processors
- 1000 - 10000 kLOC
- Planned and thoroughly organized reuse
- Product life time maintenance (5 - 20 years)

Complexity vs. lead time



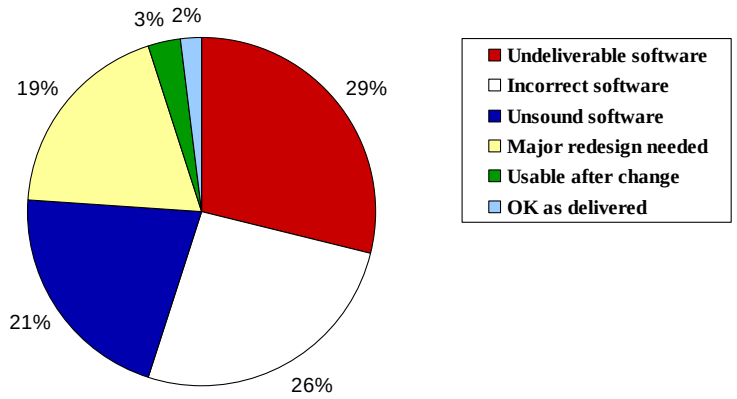
The Art of Software Engineering

- Functionality
- Appearance
- Usability
- Performance
- Stability
- Understandability
- Testability
- Reusability
- Maintainability
- …..

To be Successful, Consider that...

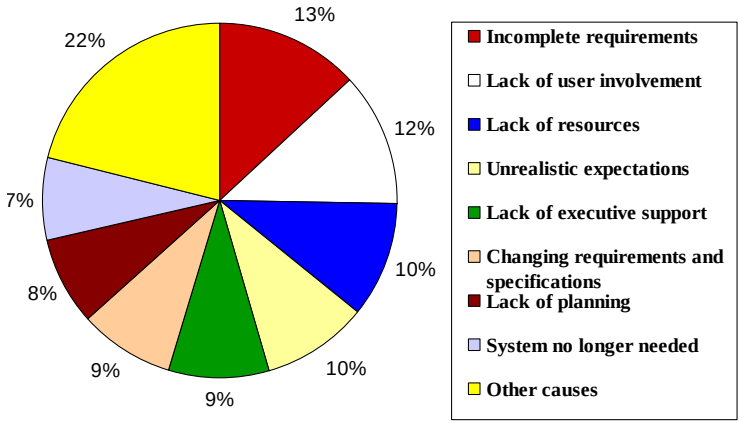
- Systems are increasingly complex
- Customer expectations and customer requirements don't always match
- Software projects typically exceed budgets and schedules
- Time To Market (TTM) is crucial for competition
- Software failures may harm our lives
- Bugs means loss of confidence
- ...

The Software Crisis is not Over



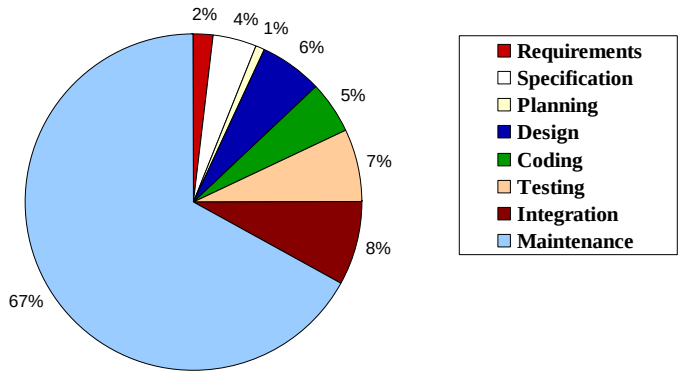
Study from ...

Why Do Projects Fail?



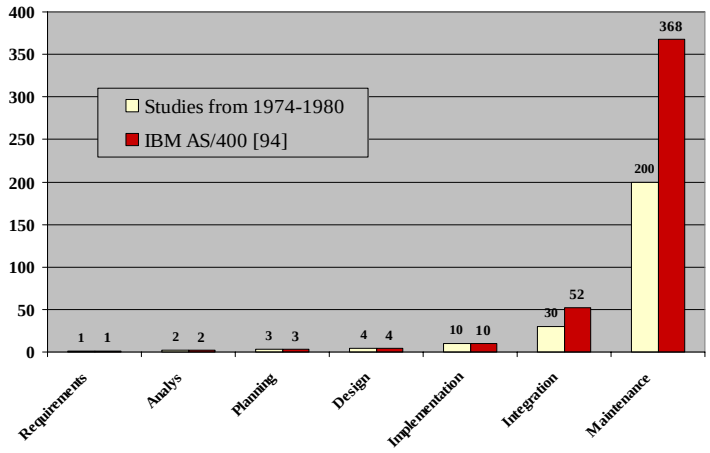
Study by the Standish Group involving 350 companies from 1994/95, see [Pfleeger 98].

Relative Costs of Development Phases



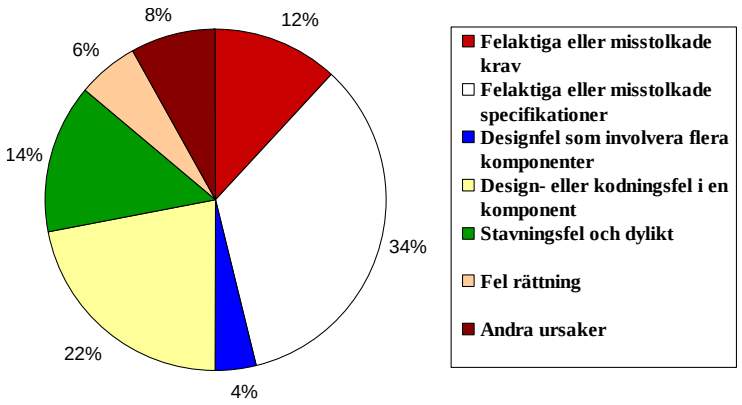
Compiled data from 1976-1981, see [Schach 97].

Relative Costs of an Error



See [Schach 97].

Causes of Errors

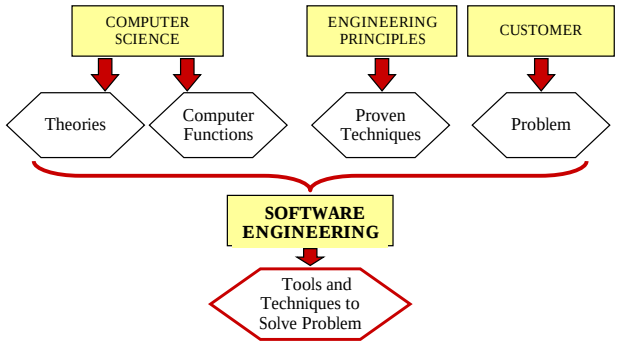


Study from 1978, see [GoRu 95].

What is Software Engineering?

"The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines."

Definition proposed by Fritz Bauer at the NATO conference '68 in Garmisch [NRB 76]



But ...

" ... we all tell each other and ourselves that software engineering techniques should be improved considerably, because there is a crisis. But there are a few boundary conditions which apparently have to be satisfied. I will list them for you:

1. We may not change our thinking habits.
2. We may not change our programming tools.
3. We may not change our hardware.
4. We may not change our tasks.
5. We may not change our organizational set-up in which the work has to be done.

Now under these five immutable boundary conditions, we have to try to improve matters. This is utterly ridiculous. ..."

Comment by Edsger Dijkstra at the NATO conference '69 in Garmisch [BuRa 70]

Some projects work, others don't

Why?

No1 Key to Success

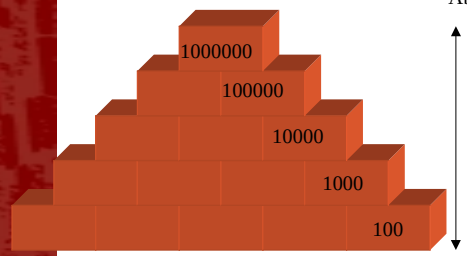
Concepts:
Abstractions used for communication

- Between people
- Over time

Productivity comes from the power of concepts

Feasibility study

Project sponsor



Abstract

Concrete

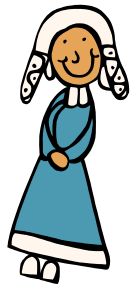
Signal coordination

Software thread

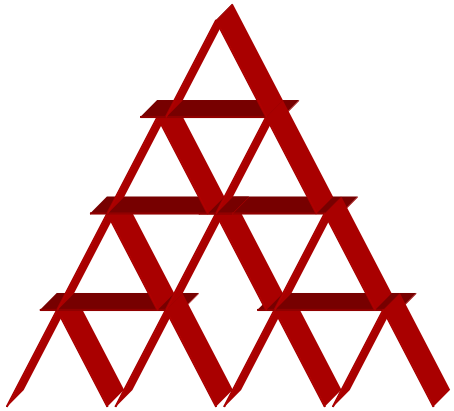
Function Framework

Iterator

How many words do you need to explain a concept to your mother ?



Confusion comes from the use of undefined concepts



Remember...

- There is rarely a single “true” definition of a concept.
- Don’t get into arguments about the “true” meaning.
- If the concept is not well established, qualify it with the context in which it is defined.
- Examples of vague but popular concepts: Project, process, program, module, system, subsystem...

To be best...

Successful Business

Ways of Working

Tools, Principles, Practices, Patterns

Language, Concepts, Terminology

… we must have a powerful and unambiguous way to communicate.

To be best...

Successful Business

Ways of Working

Tools, Principles, Practices, Patterns

Language, Concepts, Terminology

… we must use the most efficient techniques.

To be best...

Successful Business

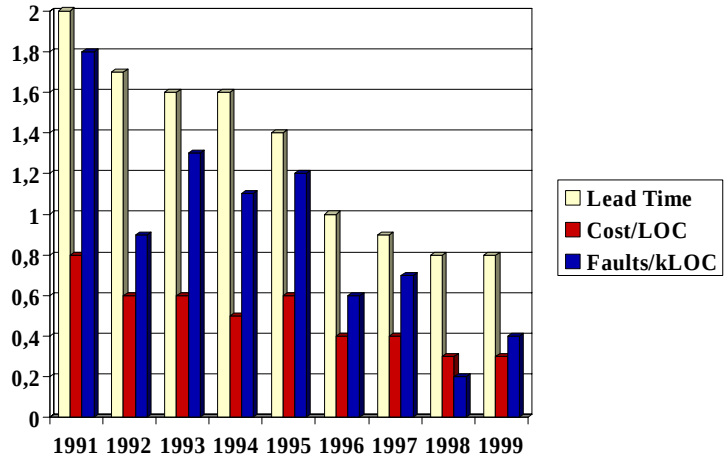
Ways of Working

Tools, Principles, Practices, Patterns

Language, Concepts, Terminology

… we must continuously improve our ways of working.

Continuous Improvement



The Challenge!

I we want to be best, doing it the same way as last time is not good enough!

A Little History (1)

1940 Construction of the first computers
Hardware dominates

1950 Single batch programs

1960 More complex programs
Software crisis
- More application domains
- Not enough educated programmers
- No common programming languages
- No special project management
- No quality assurance
- Quality = efficiency
- ...

1968 NATO conference in Garmisch gives “birth” to Software Engineering
- Engineering approach to software production
- Systematic production of large software systems of high quality

A Little History (2)

1970 Life-cycle models
Structured programming and design
General purpose languages
Requirements definition languages
Modularization

1980 Many new methods, languages, and tools
First programming environments
Project management support

1985 Object-orientation, GUIs
Reuse, Re-engineering, ...
Process modeling
Distributed computing

1990 OOA/D, Software architecture, CORBA
Patterns, Internet computing, CMM, PSP

1995 Java, UML
T-PSP, ...