Real Software Engineering

Glenn Vanderburg
LivingSocial

glv@vanderburg.org
@glv

Forty-Four Years of Software Engineering

Software Engineering Doesn't Work!

How We Got Here

REPORT ON A CONFERENCE SPONSORED BY THE
NATO SCIENCE COMMITTEE
ROME, ITALY, 27th to 31st OCTOBER 1969

SOFTWARE ENGINEERING TECHNIQUES

CHAIRMAN : PROFESSOR P. ERCOLI
CO-CHAIRMAN : PROFESSOR DR. F.L. BAUER

EDITORS : J.N. BUXTON and B. RANDELL

APRIL 1970

Unlike the first conference, at which it was fully accepted that the term software engineering expressed a need rather than a reality, in Rome there was already a slight tendency to talk as if the subject already existed.

And it became clear during the conference that the organizers had a hidden agenda, namely that of persuading NATO to fund the setting up of an International Software Engineering Institute.

However things did not go according to their plan. The discussion sessions which were meant to provide evidence of strong and extensive support for this proposal were instead marked by considerable scepticism, and led one of the participants, Tom Simpson of IBM, to write a splendid short satire on "Masterpiece Engineering".

It was little surprise to any of the participants in the Rome conference that no attempt was made to continue the NATO conference series, but the software engineering bandwagon began to roll as many people started to use the term to describe their work, to my mind often with very little justification.

—Brian Randell

“Premature maturity”

Engineering: Caricature and Reality

“Engineering is straightforward and predictable”

[Programming] is not some kind of engineering where all we have to do is put something in one end and turn the crank.

—*Bruce Eckel*

The conversion of an idea to an artifact, which engages both the designer and the maker, is a complex and subtle process that will always be far closer to art than to science.

—*Eugene S. Ferguson,* Engineering and the Mind's Eye

The *defined process control model* requires that every piece of work be completely understood. A defined process can be started and allowed to run until completion, with the same results every time.

The *empirical process control model* provides and exercises control through frequent inspection and adaptation for processes that are imperfectly defined and generate unpredictable and unrepeatable outputs.

"Engineers produce drawings and documents."

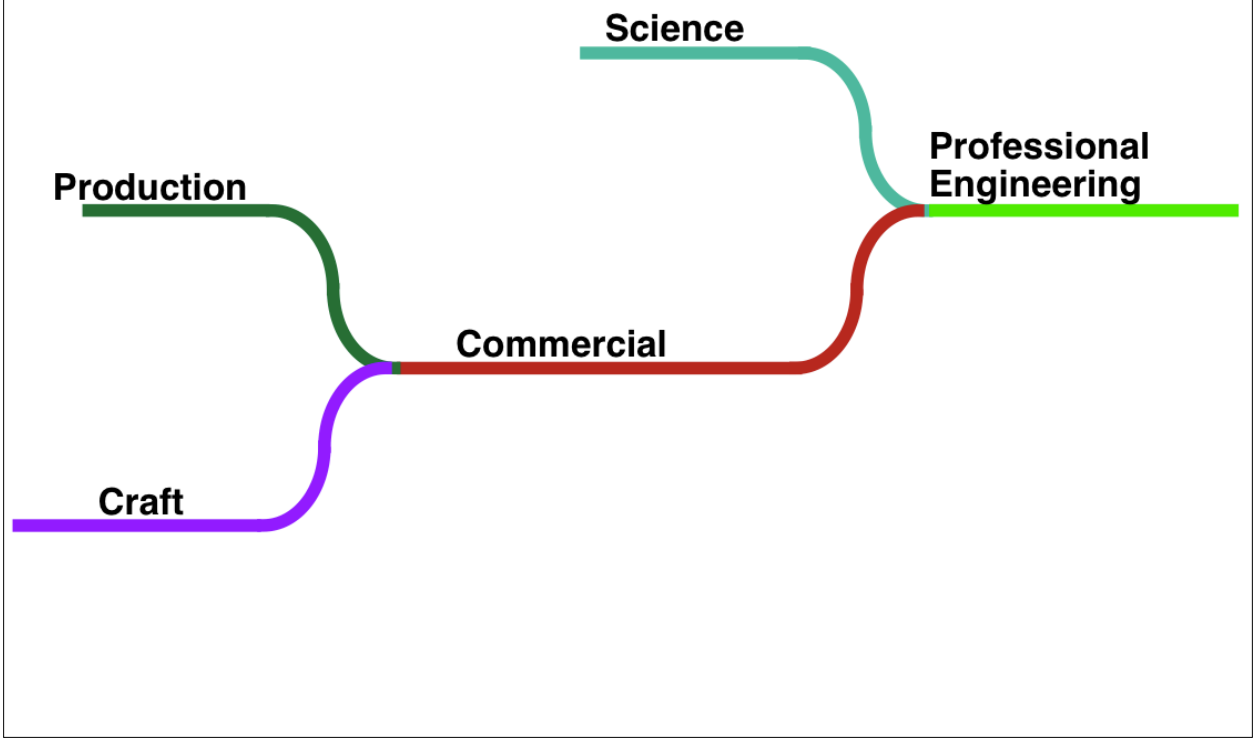
In engineering ... people design through documentation.

—*David Parnas*

Although the drawings appear to be exact and unequivocal, their precision conceals many informal choices, inarticulate judgments, acts of intuition, and assumptions about the way the world works.

—*Eugene S. Ferguson,* Engineering and the Mind's Eye

"Engineering is applied science."



@skilldrick
Nick Morgan

I've invented a new term "Parasitic Credibility". Where certain fields attempt to link themselves to science in order to sound more real.

Van Vleck was very concerned that the practice of engineering be put on a firmer scientific basis. He led a vigorous shift of American engineering education away from design toward applied science. The pendulum swung too far; reaction set in; and the teaching of design has been contentious ever since.

—*Fred Brooks,*
The Design of Design

Aeroplanes are not designed by science, but by art, in spite of some pretence and humbug to the contrary. [...]

There is a big gap between scientific research and the engineering product which has to be bridged by the art of the engineer.

—*J. D. North*

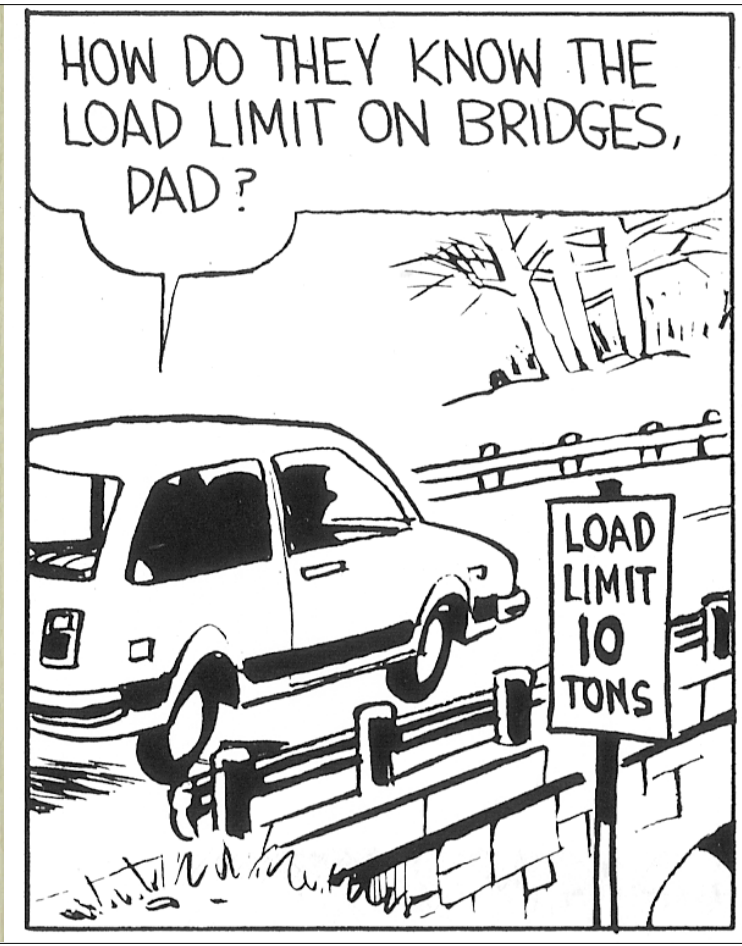
Engineers frequently have to make decisions of great practical consequence in the face of incomplete and uncertain knowledge.

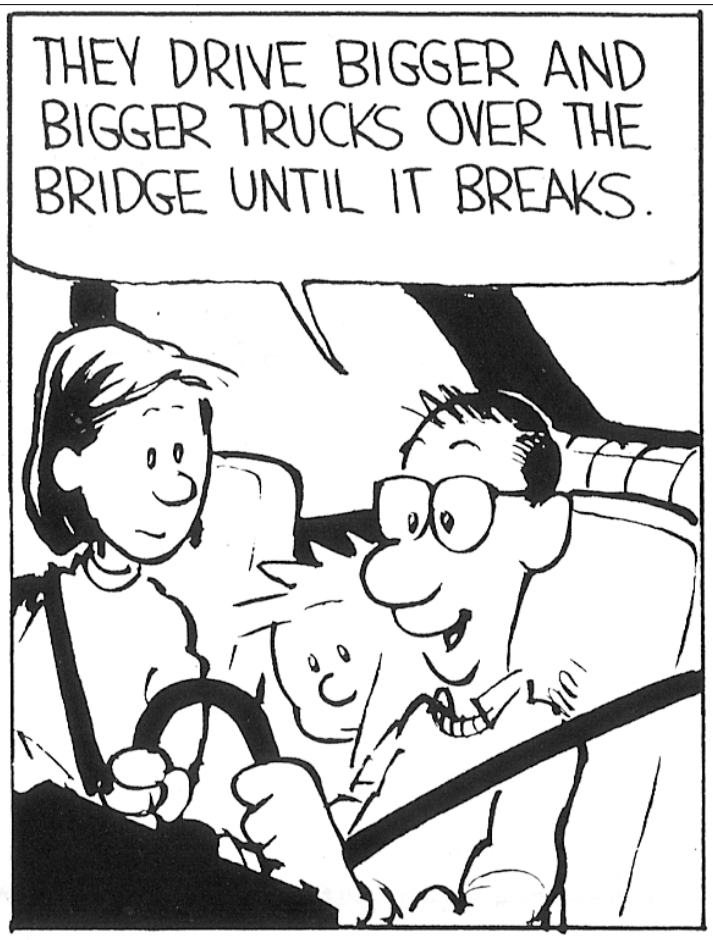
—*Walter Vincenti,*
What Engineers Know
and How They Know It

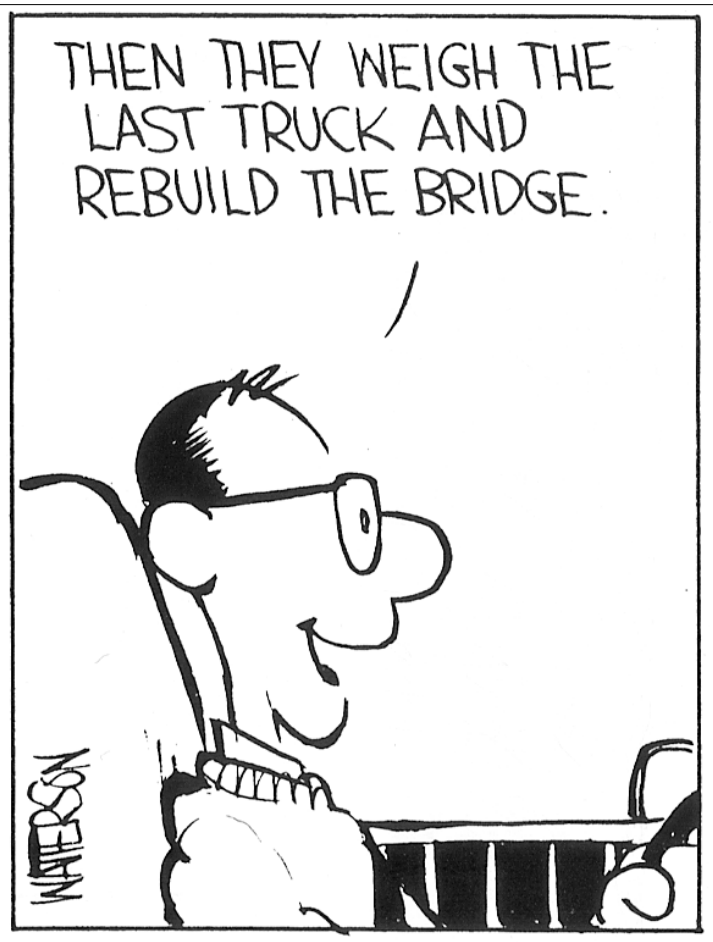
"Engineers **prove** their designs with mathematics!"

Structural analyses (indeed, any engineering calculations) must be employed with caution and judgment, because mathematical models are always less complex than actual structures, processes, or machines.

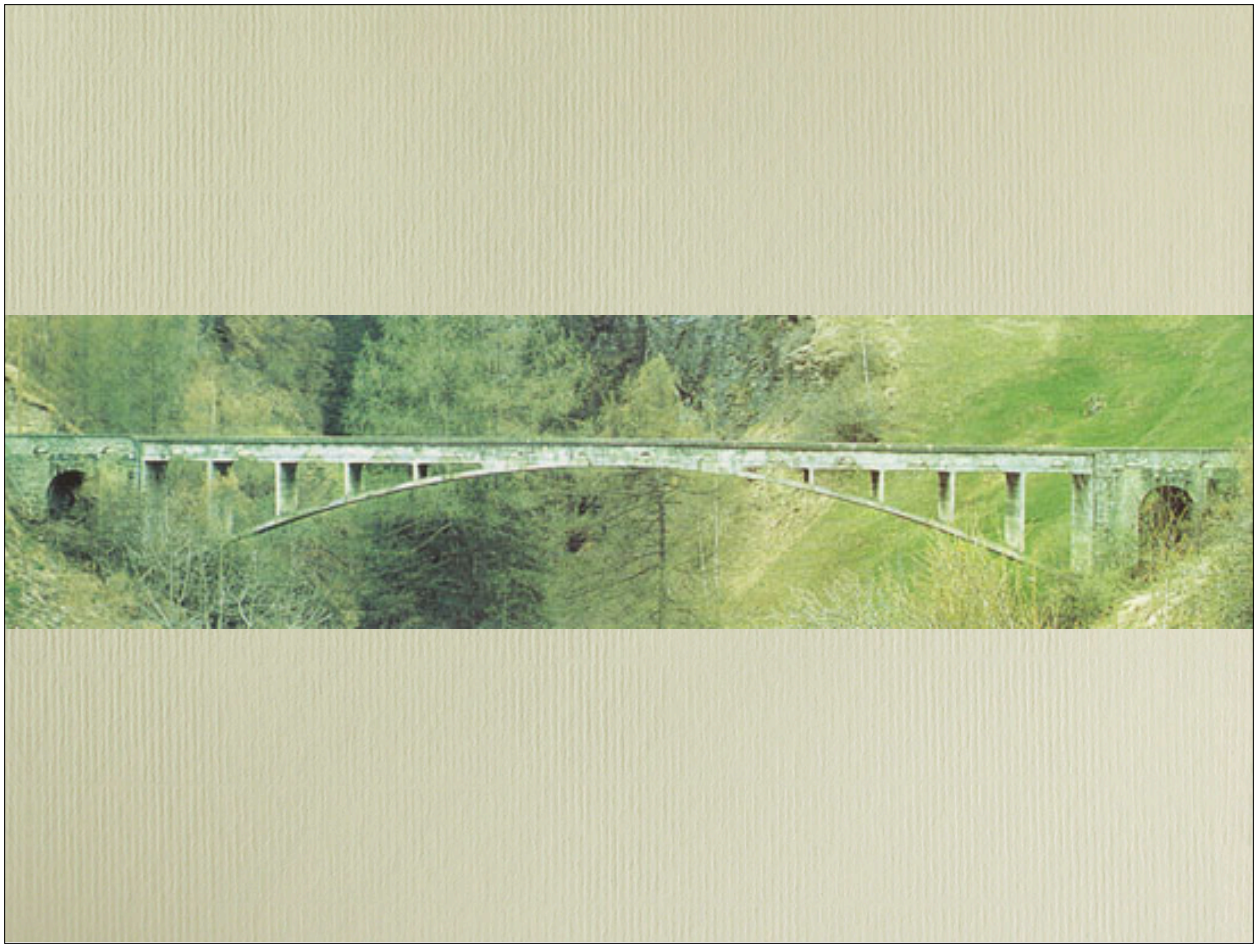
—*Eugene S. Ferguson,*
Engineering and the Mind's Eye




THEY DRIVE BIGGER AND BIGGER TRUCKS OVER THE BRIDGE UNTIL IT BREAKS.


THEN THEY WEIGH THE LAST TRUCK AND REBUILD THE BRIDGE.
WATTERSON


OH. I SHOULD'VE GUESSED.
DEAR, IF YOU DON'T KNOW THE ANSWER, JUST TELL HIM!



Mathematical modeling was introduced as a cost-saving measure.

Engineering is not the art of constructing. It is rather the art of not constructing: or, it is the art of doing well with one dollar what any bungler can do with two.

—*Arthur Mellen Wellington*

Different Engineering Disciplines are *Different*

- Different materials, physical effects, forces
- Different degrees of complexity in requirements, designs, processes, and artifacts
- Varied reliance on formal modeling and analysis vs. experimentation, prototyping, and testing
- Varied use of defined and empirical processes

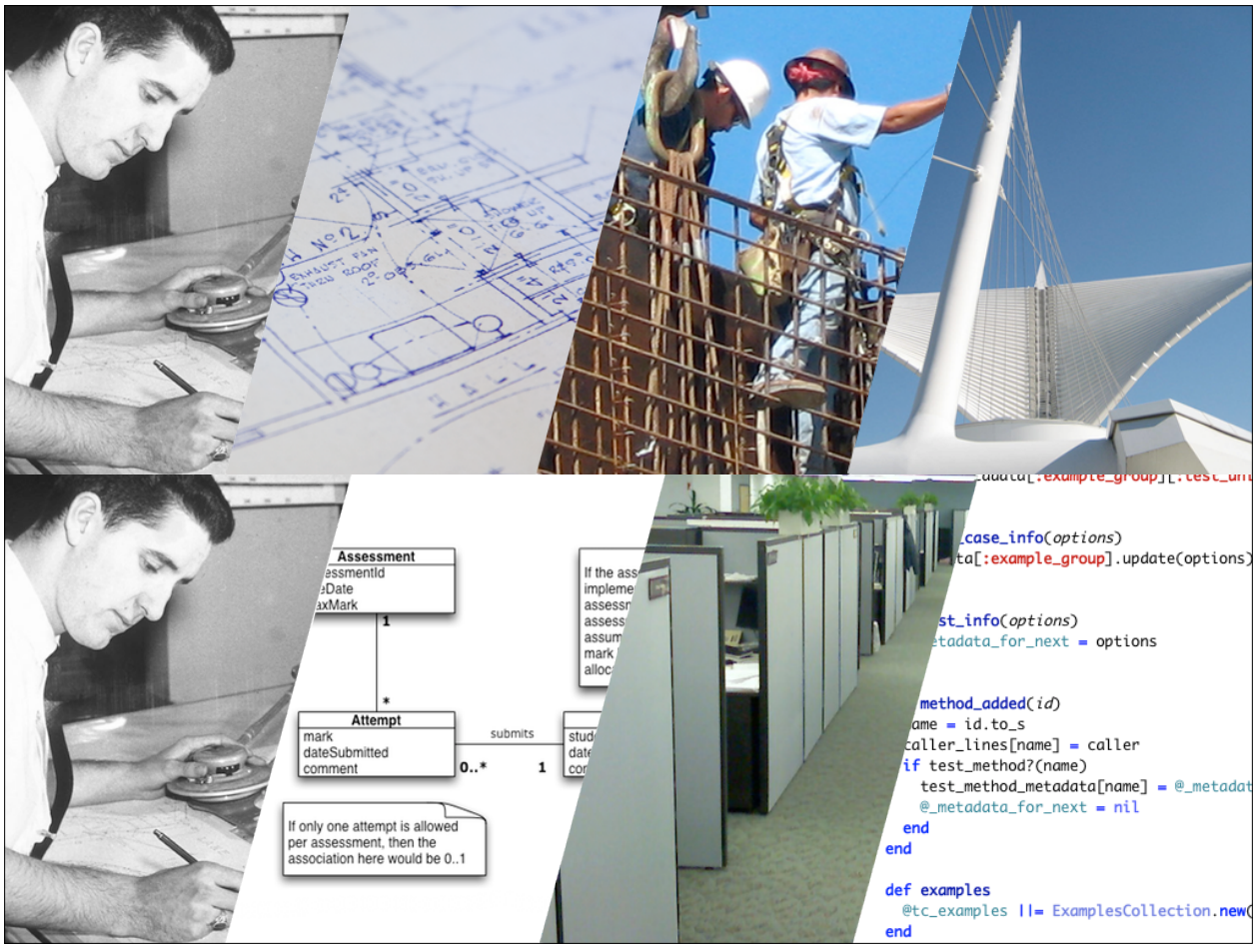
Structural engineering is
the science and art
of designing and making,
with economy and elegance,
[...] structures so that they
can safely resist the forces to
which they may be subjected.

—*Structural Engineer's Association*

Real
Software Engineering

Software engineering is
the science and art
of designing and making,
with economy and elegance,
[...] systems so that they can
readily adapt to the situations to
which they may be subjected.

Software engineering will be *different* from other kinds of engineering.



Condition							Value N(T))
N(p(T)) = Pass							Pass
N(p(T)) = Fail							Fail
N(p(T))≠ Pass ∧ N(p(T))≠ Fail	r(T)= N(p(T))	N(p(T)) = L					Pass
		N(p(T)) ≠ L					N(p(T))+1
	r(T)≠ N(p(T))	r(T)≠ esc	r(p(T))≠ N(p(p(T)))	r(p(T))= esc	p(p(T))= _		N(p(T))
					p(p(T)) ≠_	r(p(p(T)))= N(p(p(p(T))))	N(p(T))–1
						r(p(p(T))) ≠ N(p(p(p(T))))	N(p(T))
				r(p(T)) ≠ esc			N(p(T))
			r(p(T)) = N(p(p(T)))				N(p(T))-1
		r(T)= esc	r(p(T))=esc	N(p(p(T))) = esc			N(p(T))
				N(p(p(T))) ≠ esc			Fail
			r(p(T)) ≠ esc				N(p(T))

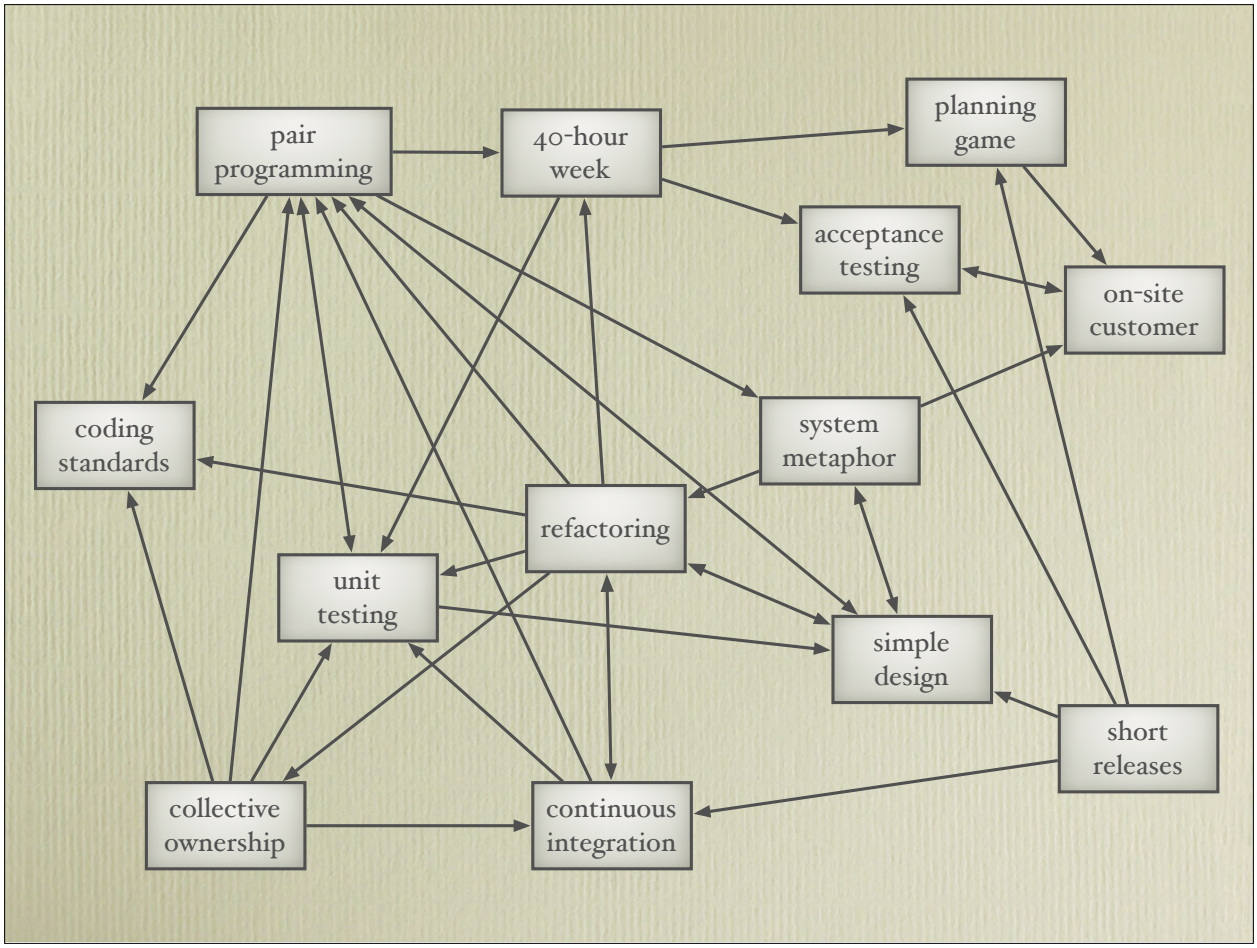
divided by	2	3	4	7
9	4.5	3	2.25	1.29
10	5.0	3.33	2.5	1.43
11	5.5	3.66	2.75	1.57
12.6	6.3	4.2	3.15	1.8
22	11.0	7.33	5.5	3.14
100	50.0	33.33	32	14.29

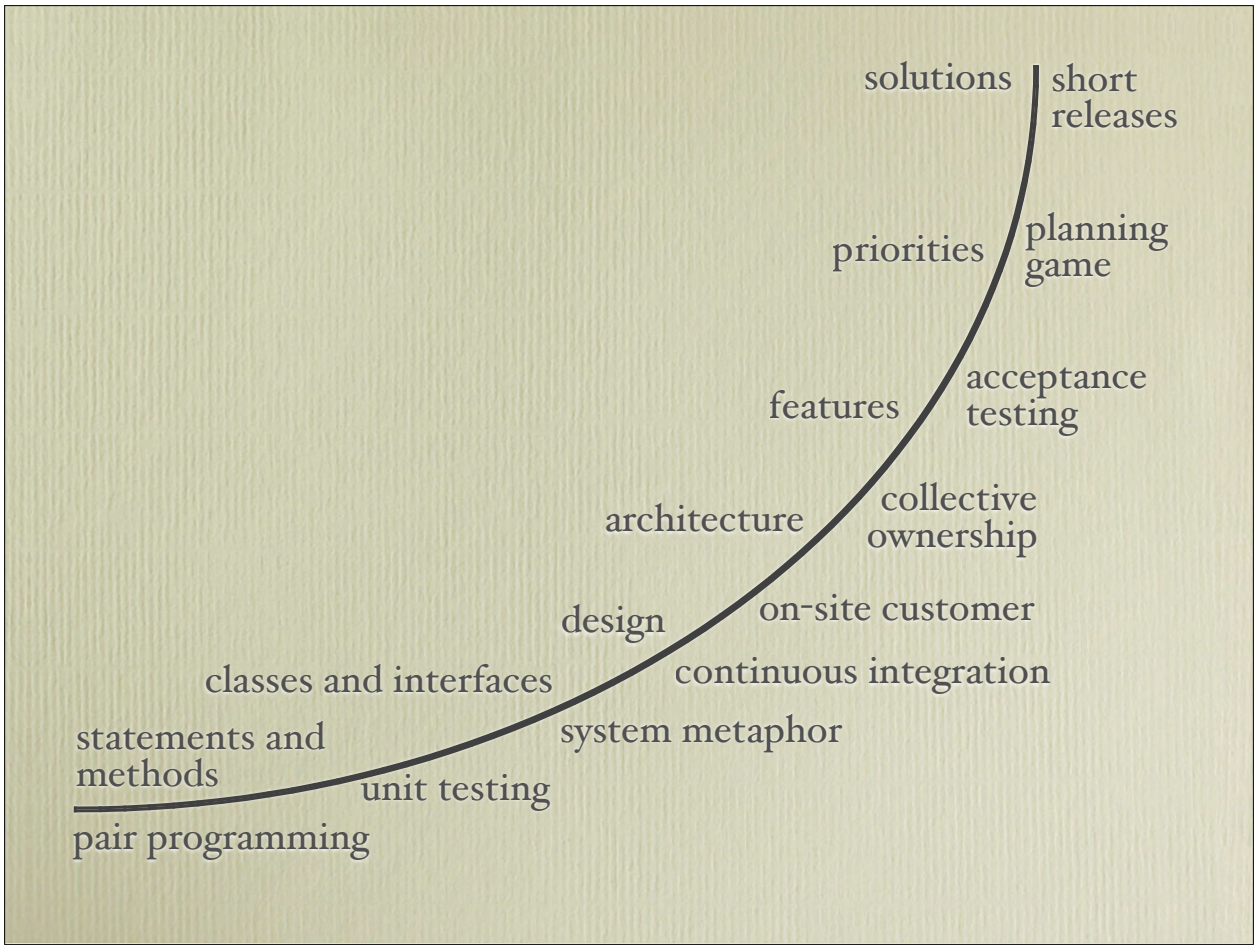
```
Feature: Addition
  In order to avoid silly mistak
  As an error-prone person
  I want to divide two numbers

  Scenario Outline: Divide two nu
    Given I have entered <input_
    And I have entered <input_2>
    When I press "divide"
    Then the result should be <r

  Examples:
    | input_1 | input_2 | result  |
    | 10      | 2       | 5.0     |
    | 12.6    | 3       | 4.2     |
    | 22      | 7       | ~=3.14  |
    | 9       | 3       | <5      |
    | 11      | 2       | 4<_<6   |
    | 100     | 4       | 32      |
```

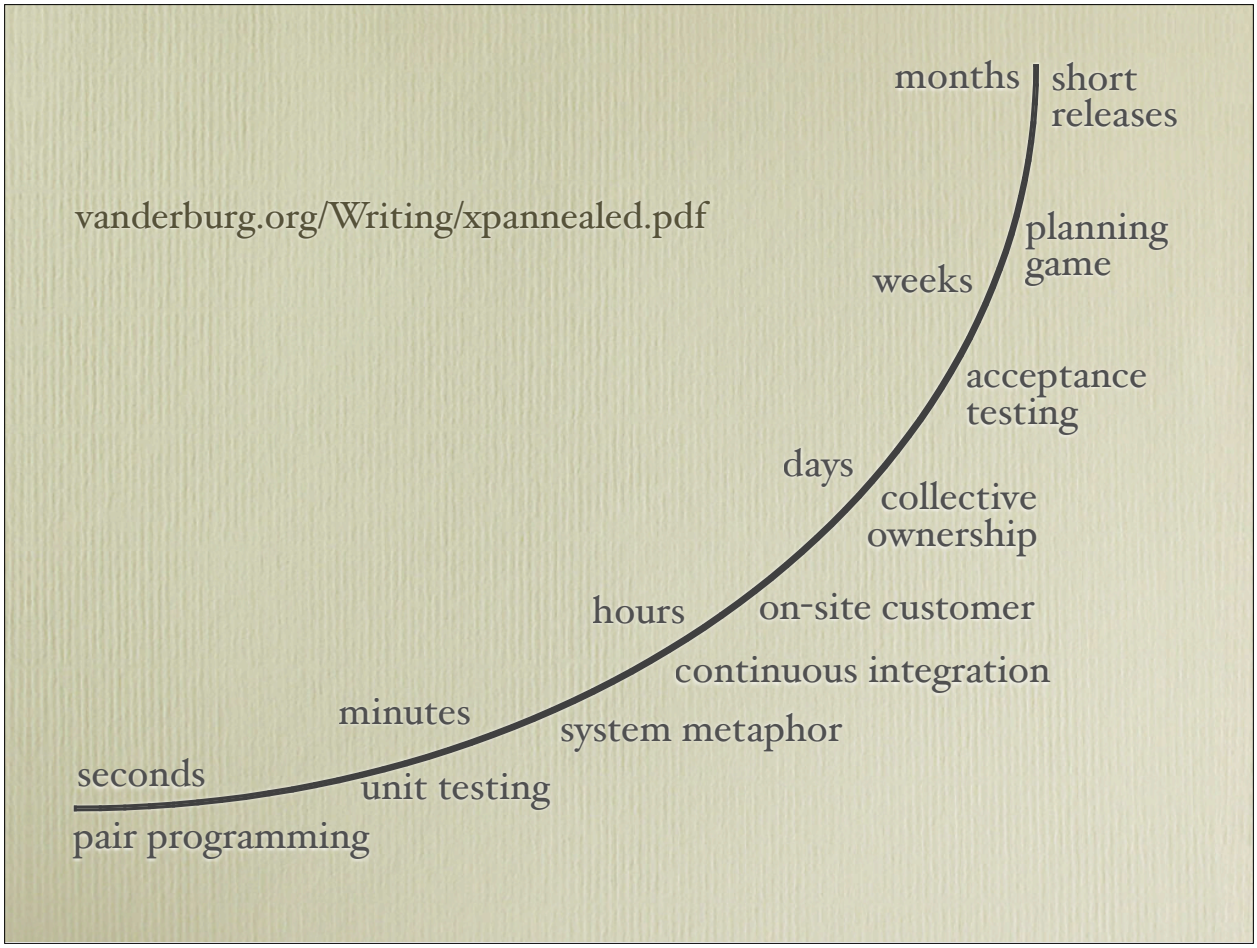
eg.Division		
numerator	denominator	quotient?
10	2	5.0
12.6	3	4.2
22	7	~=3.14
9	3	<5
11	2	4<_<6
100	4	32 expected / 25 actual

```
    assert_in_delta  3.1
    assert           5
    assert           4..
    assert_equal     32,
  end
end
```

```
ers" do
do
should == 5.0
should == 4.2
should be_close(3.14, 0.01)
  (9 / 3)    .should < 5
  (11/2)     .should satisfy{|n| n > 4 && n < 6 }
  (100 / 4) .should == 32
  end
end
```


pair programming
40-hour week
planning game
acceptance testing
on-site customer
coding standards
system metaphor
refactoring
unit testing
simple design
short releases
collective ownership
continuous integration


solutions
short releases
priorities
planning game
features
acceptance testing
architecture
collective ownership
design
on-site customer
classes and interfaces
continuous integration
statements and methods
system metaphor
unit testing
pair programming



Discarding Incorrect Assumptions

The Reality of Software Engineering

Growing Up

Real Software Engineering

Glenn Vanderburg
LivingSocial

glv@vanderburg.org
@glv

Photo Credits (all from Flickr):

Seattle Municipal Archives:	/photos/seattlemunicipalarchives/2713475713/
Will Scullin:	/photos/wscullin/3770015203/
Bill Jacobus:	/photos/billjacobus1/122497422/
kke:	/photos/kesta/1818986646/
Carol Shergold:	/photos/carolshergold/1921464196/
Bill Abbott:	/photos/wbaiv/3236672907/