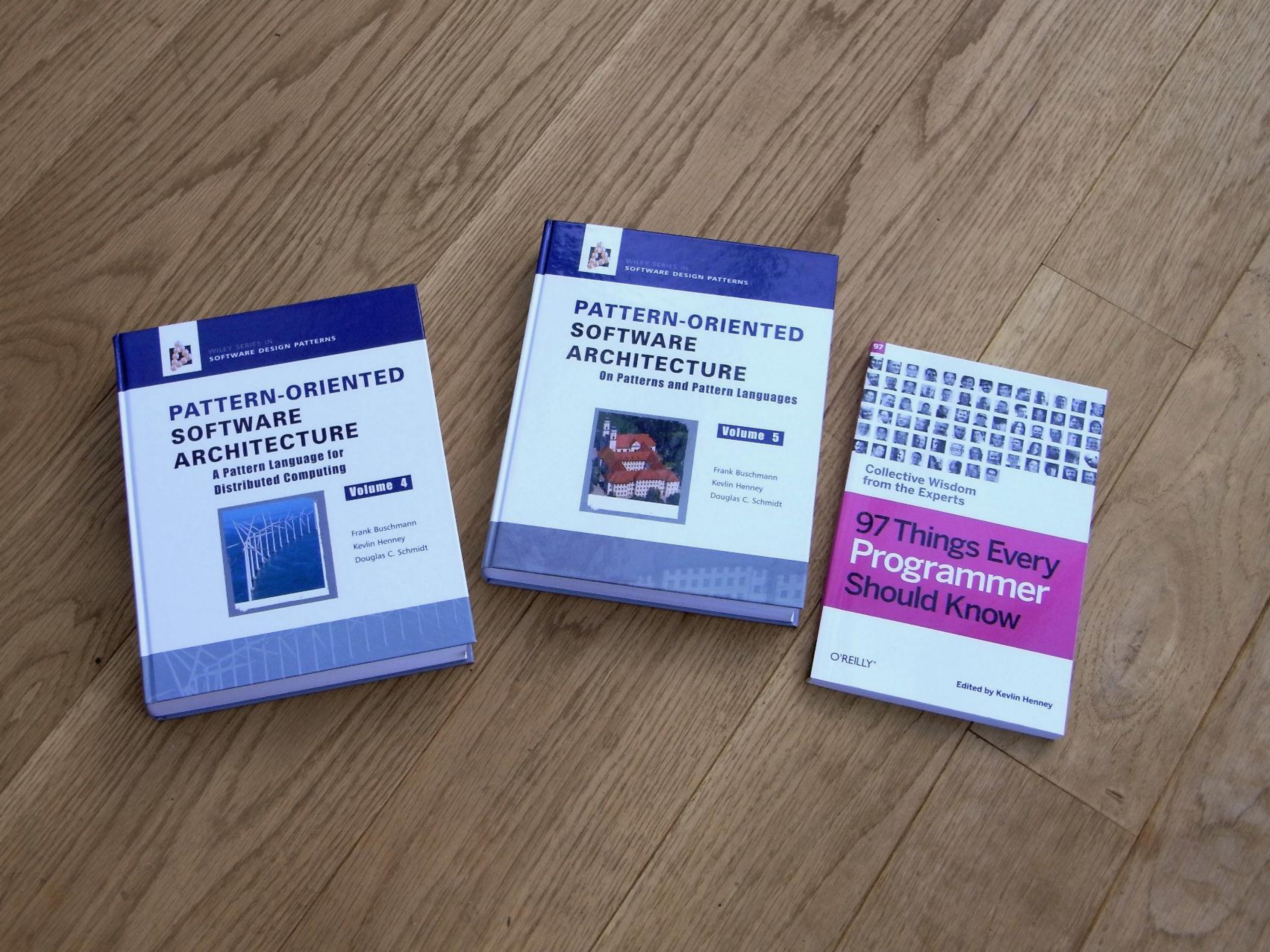
Putting the "re" into Architecture

Kevlin Henney
kevlin@curbralan.com
@KevlinHenney


WILEY SERIES IN SOFTWARE DESIGN PATTERNS
PATTERN-ORIENTED SOFTWARE ARCHITECTURE
A Pattern Language for Distributed Computing
Volume 4
Frank Buschmann
Kevlin Henney
Douglas C. Schmidt
WILEY SERIES IN SOFTWARE DESIGN PATTERNS
PATTERN-ORIENTED SOFTWARE ARCHITECTURE
On Patterns and Pattern Languages
Volume 5
Frank Buschmann
Kevlin Henney
Douglas C. Schmidt
97
Collective Wisdom from the Experts
97 Things Every Programmer Should Know
O'REILLY
Edited by Kevlin Henney

2011 PREVIEW The big stories to look out for next year

NewScientist

25 December 2010/1 January 2011

CHRISTMAS & NEW YEAR SPECIAL

Robothespians
The roast of Christmas future
Victorian science tales
Sozzled superconductors
Sequence my Stilton
Extreme beer
Horror lizards
Steampunk Lego
Booze myths busted
Secret Santa's secret
Jingle hells
Electric elk
Plus much, much more...

£3.95
US/CAN$5.95
No2792/93

News, ideas and innovation www.NewScientist.com The best jobs in science

No design system is or should be perfect.

That which is overdesigned, too highly specific, anticipates outcome; the anticipation of outcome guarantees, if not failure, the absence of grace.

William Gibson
All Tomorrow's Parties

```
interface Iterator
{
    boolean set_to_first_element();
    boolean set_to_next_element();
    boolean set_to_next_nth_element(in unsigned long n) raises(…);
    boolean retrieve_element(out any element) raises(…);
    boolean retrieve_element_set_to_next(out any element, out boolean more) raises(…);
    boolean retrieve_next_n_elements(
        in unsigned long n, out AnySequence result, out boolean more) raises(…);
    boolean not_equal_retrieve_element_set_to_next(in Iterator test, out any element) raises(…);
    void remove_element() raises(…);
    boolean remove_element_set_to_next() raises(…);
    boolean remove_next_n_elements(in unsigned long n, out unsigned long actual_number) raises(…);
    boolean not_equal_remove_element_set_to_next(in Iterator test) raises(…);
    void replace_element(in any element) raises(…);
    boolean replace_element_set_to_next(in any element) raises(…);
    boolean replace_next_n_elements(
        in AnySequence elements, out unsigned long actual_number) raises(…);
    boolean not_equal_replace_element_set_to_next(in Iterator test, in any element) raises(…);
    boolean add_element_set_iterator(in any element) raises(…);
    boolean add_n_elements_set_iterator(
        in AnySequence elements, out unsigned long actual_number) raises(…);
    void invalidate();
    boolean is_valid();
    boolean is_in_between();
    boolean is_for(in Collection collector);
    boolean is_const();
    boolean is_equal(in Iterator test) raises(…);
    Iterator clone();
    void assign(in Iterator from_where) raises(…);
    void destroy();
};
```

```
interface BindingIterator
{
    boolean next_one(out Binding result);
    boolean next_n(in unsigned long how_many, out BindingList result);
    void destroy();
};
```

Public APIs, like diamonds, are forever.

Joshua Bloch
"Bumper-Sticker API Design"
http://www.infoq.com/articles/API-Design-Joshua-Bloch

All architecture is design but not all design is architecture. Architecture represents the significant design decisions that shape a system, where significant is measured by cost of change.

Grady Booch

Firmitas

Utilitas

Venustas

Uncertainty

Change

Learning

Satisfaction

Sufficiency

Sustainability

Sustainable development, which implies meeting the needs of the present without compromising the ability of future generations to meet their own needs.

Brundtland Report of the World Commission on Environment and Development

repair

refactoring

re-evaluation

remembering

revision

re-engineering

rewriting

retrospection

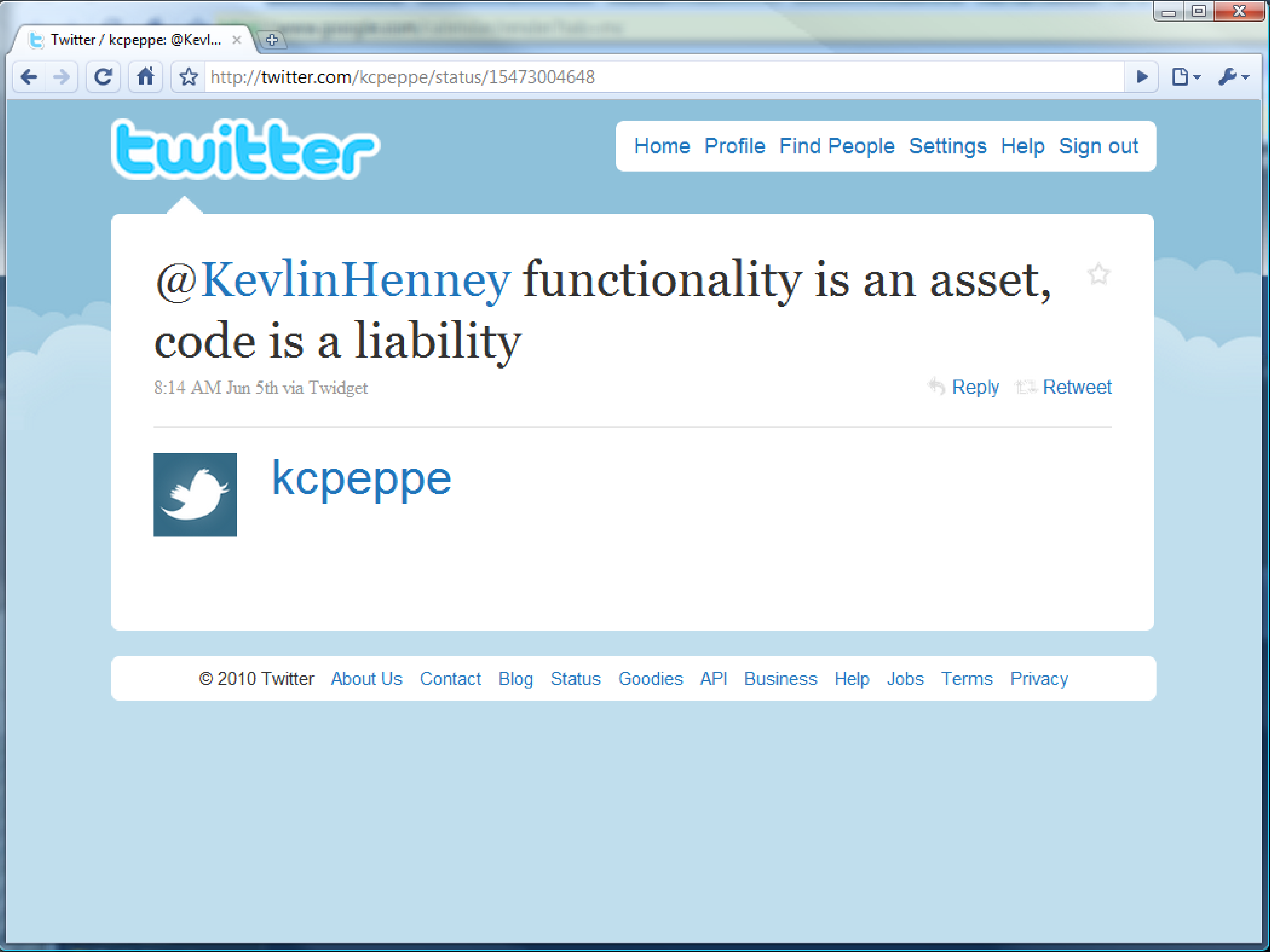
reduction

reaction

recovery

reuse

55
0
5
10
15



Twitter / kcpeppe: @Kevl...
http://twitter.com/kcpeppe/status/15473004648
twitter
Home
Profile
Find People
Settings
Help
Sign out

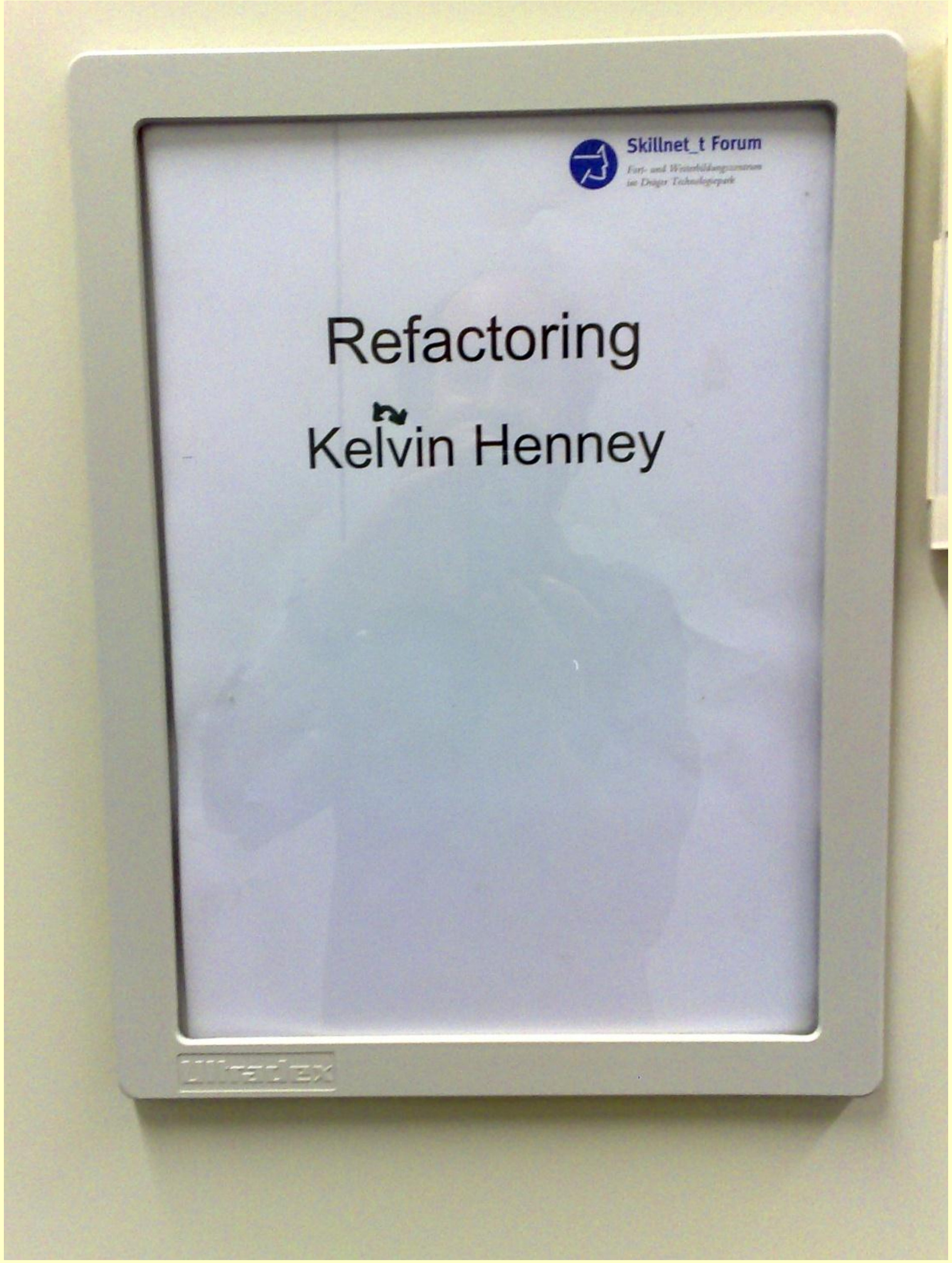
@KevlinHenney functionality is an asset, code is a liability

8:14 AM Jun 5th via Twidget

kcpeppe

It is better to be roughly right than precisely wrong.

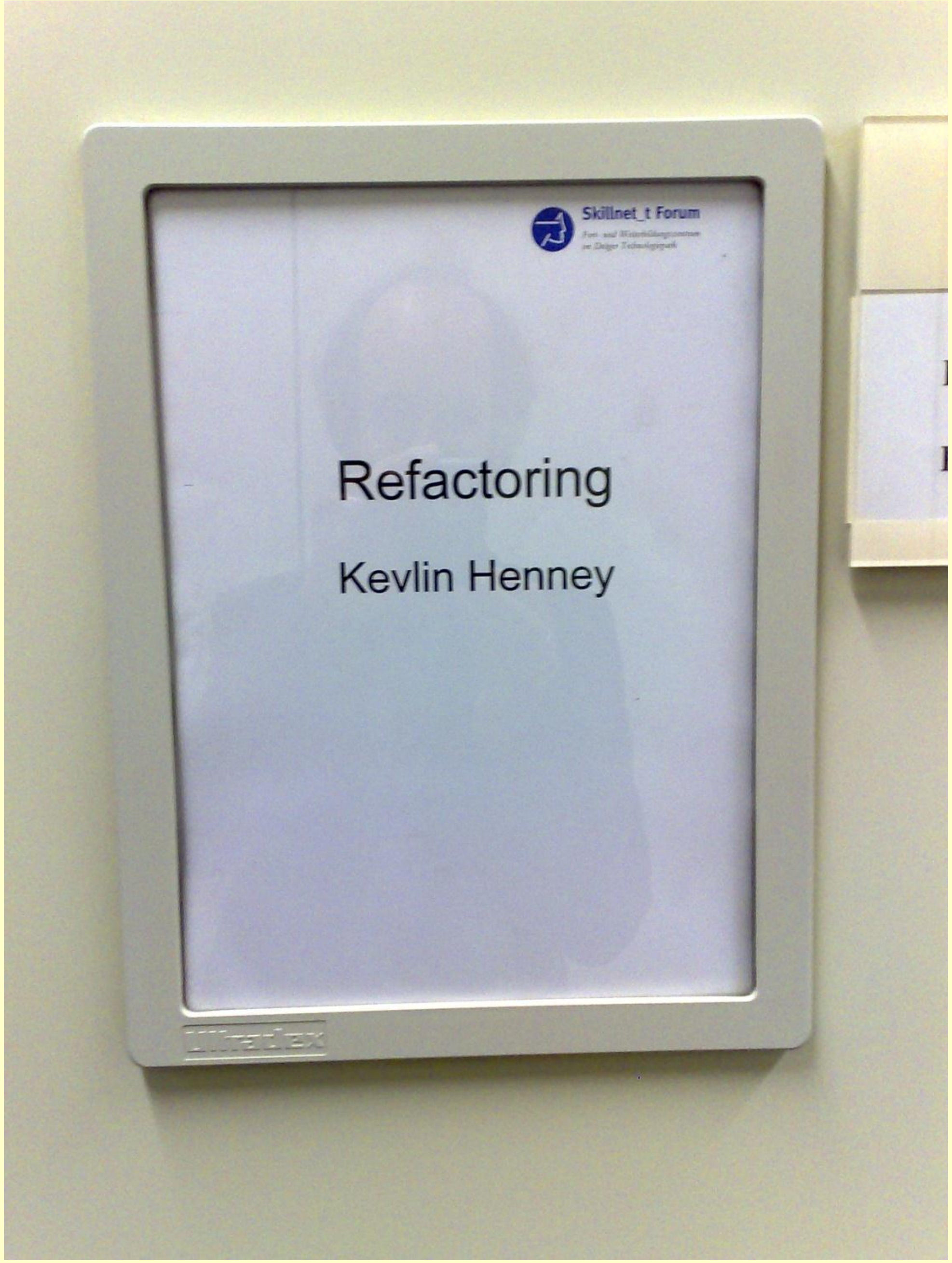
John Maynard Keynes

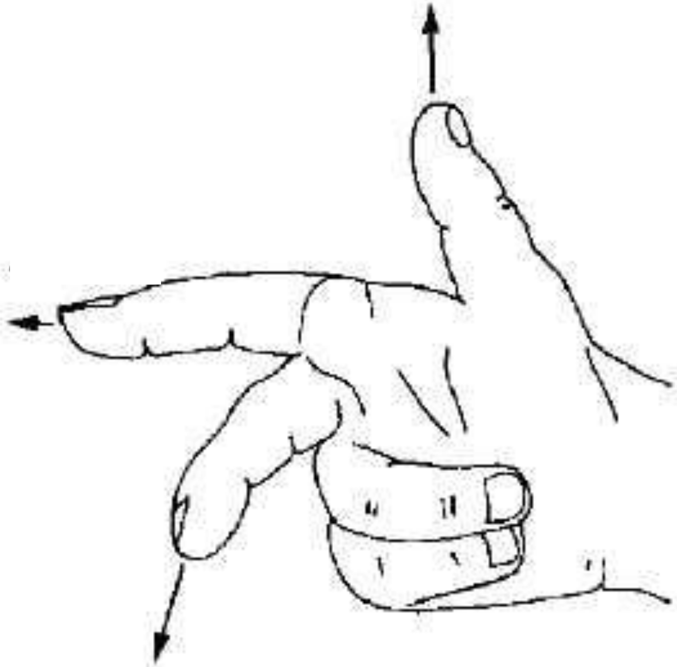


Skillnet_t Forum
Fort- und Weiterbildungszentrum
im Dräger Technologiepark

Refactoring

Kelvin Henney


Skillnet_t Forum
Fort- und Weiterbildungszentrum
im [illegible] Technologiepark
Refactoring
Kevlin Henney



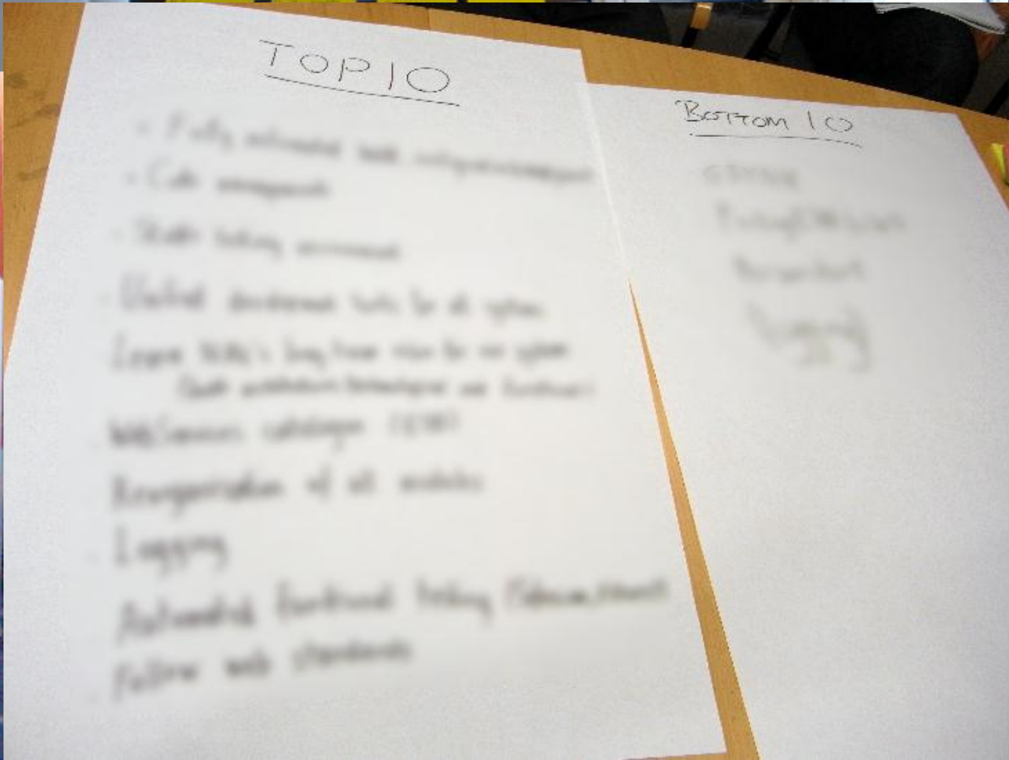
Functional

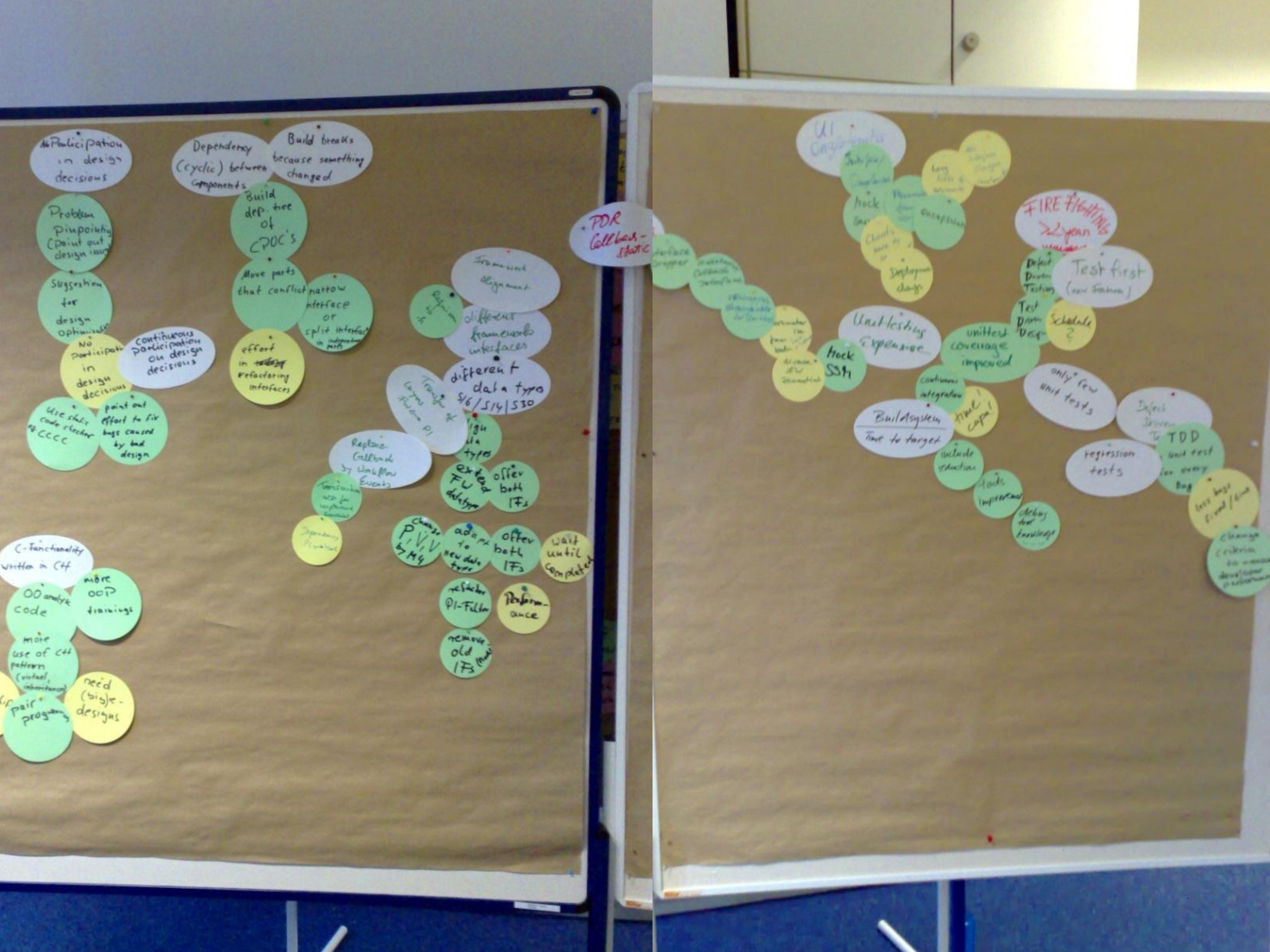
Operational

Developmental

model based code generation / testing
Code generation beyond UML
too much implementation detail
Less detail in UML models
reduce amount of queue's (complexity)
Holocot classes general methods
Data/Command transfer via Objects and queue's
Message parser
Testlanguage for Ventilation
Domain Specific Language
DSL ?
USE ANOTHER LANGUAGE
EARLIER CODE REVIEWS
Interface Definition
Documentation of Interfaces
Architektur/ Dependencies
Release management
Writing Requirements
tool chain
time / duration
Traceability TEST
Testsystem (Unittests)
framework
Clear Case
UCM config management
error handling
Different base

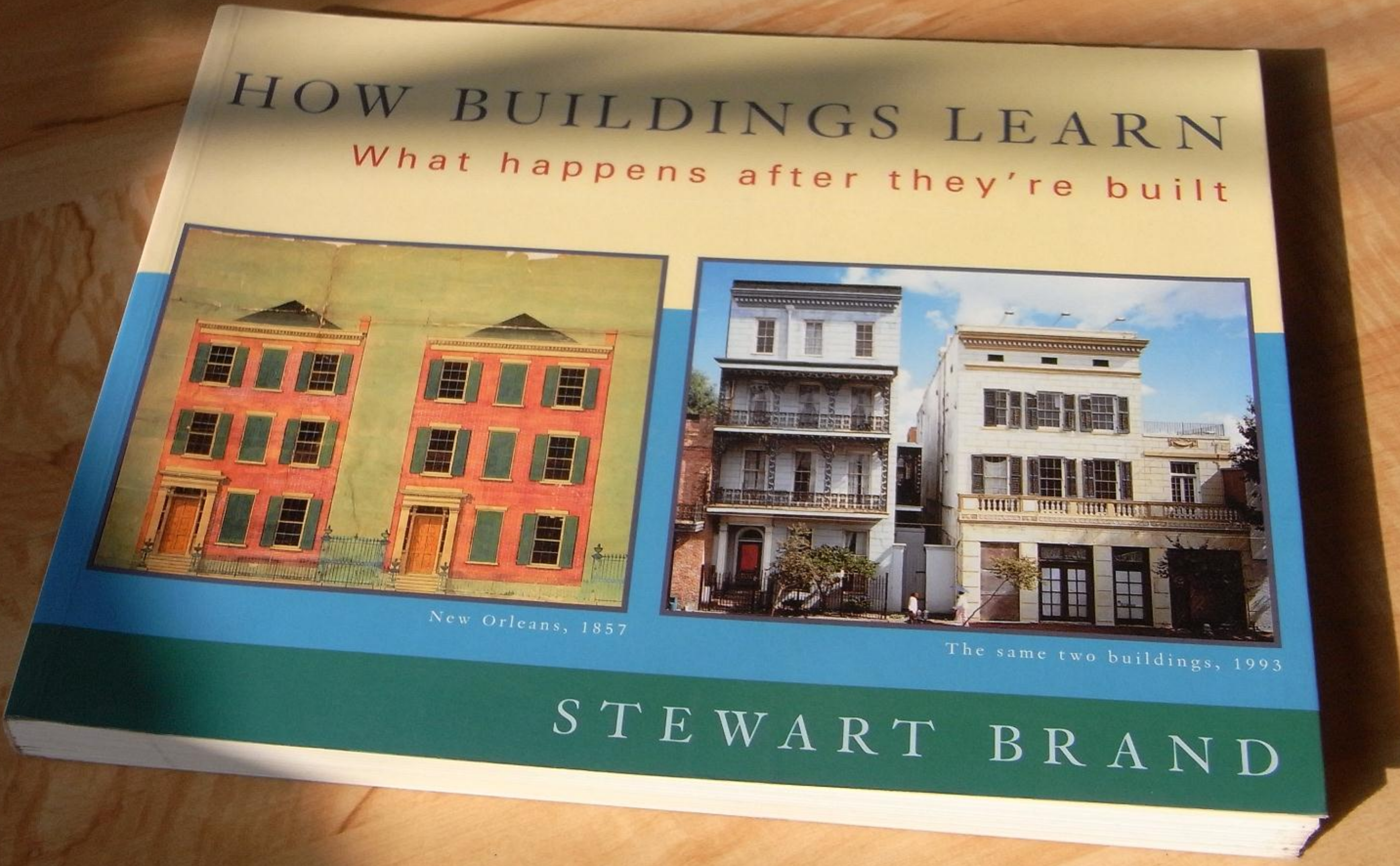
SCRUM


TOP 10
BOTTOM 10
Logging


No Participation in design decisions
Dependency (cyclic) between Components
Build breaks because something changed
Problem pinpointing (point out design issue
Build dep. tree of CPOC's
Suggestion for design optimizati
Move parts that conflict
narrow interface or split interface in independent parts
No participation in design decisions
continuous participation on design decisions
effort in ~~testing~~ refactoring interfaces
Use static code checker e.g. CCCC
point out effort to fix bugs caused by bad design
PDR Callback-static
Framework Alignment
different framework interfaces
different data types S16/S14/S30
Replace Callback by Workflow Events
align data types
extend FW datatypes
offer both IFs
change P, V, V by MG
adapt to new data type
offer both IFs
wait until completed
refactor PI-Filter
Performance
remove old IFs
Dependency
C-Functionality written in C++
OO analyze code
more OOP trainings
more use of C++ pattern (virtual, inheritance)
need (big)-designs
pair programming
UI
Mock
Deployment days
Interface Wrapper
Mock SSH
Unittesting expensive
unittest coverage improved
Continuous integration
time! capa!
Buildsystem
Time to target
include education
tools improvement
FIRE FIGHTING >2 years
Defect Driven Testing
Test first (new features)
Test Driven Dev
Schedule ?
only few unit tests
Defect Driven Te
TDD
unit test for every Bug
regression tests
less bugs fixed / time
change criteria to measure developer performance

Prediction is very difficult, especially about the future.

Niels Bohr



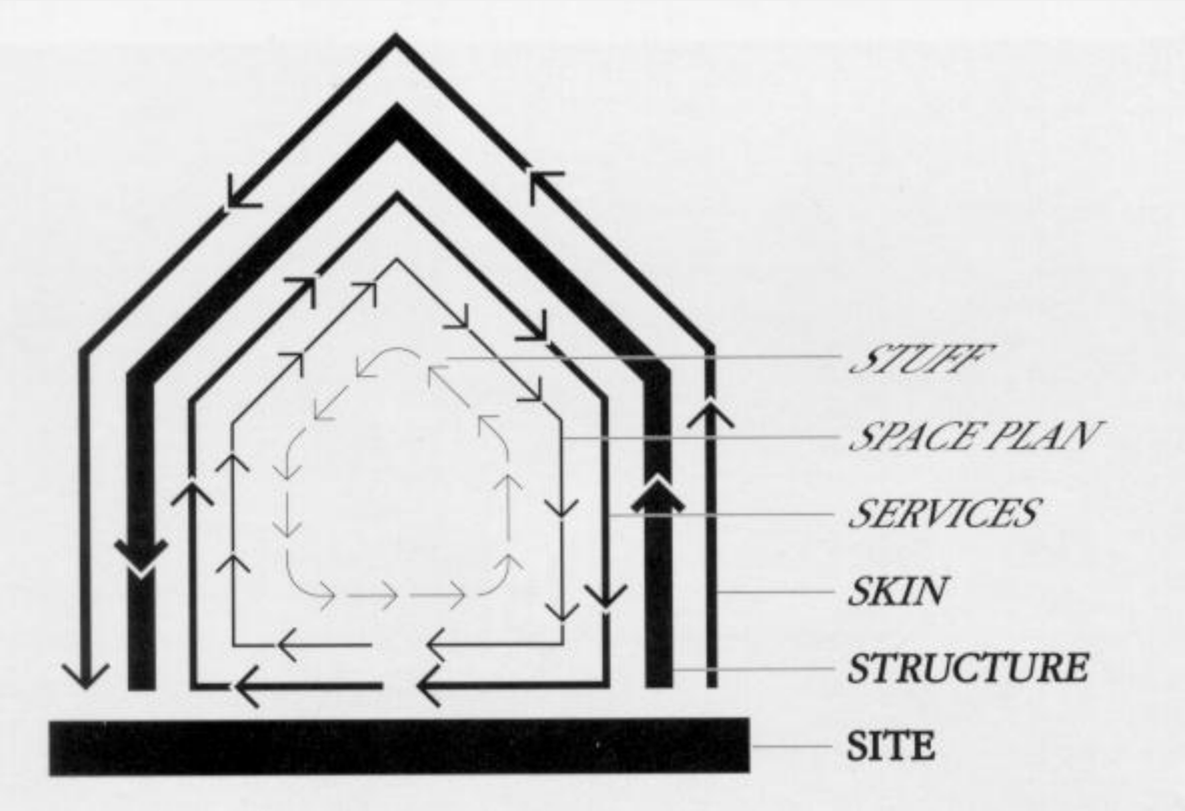
HOW BUILDINGS LEARN

What happens after they're built

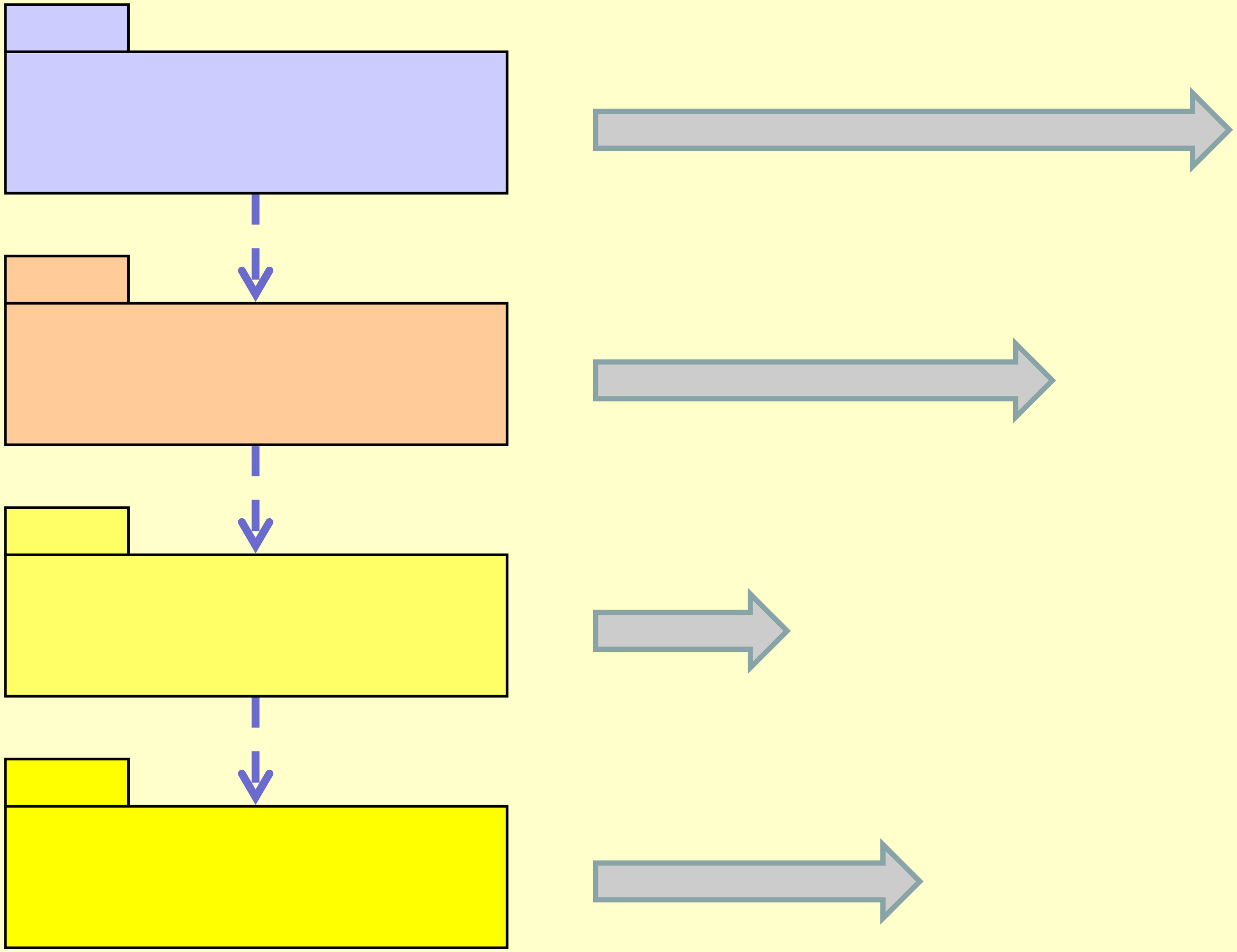
New Orleans, 1857

The same two buildings, 1993

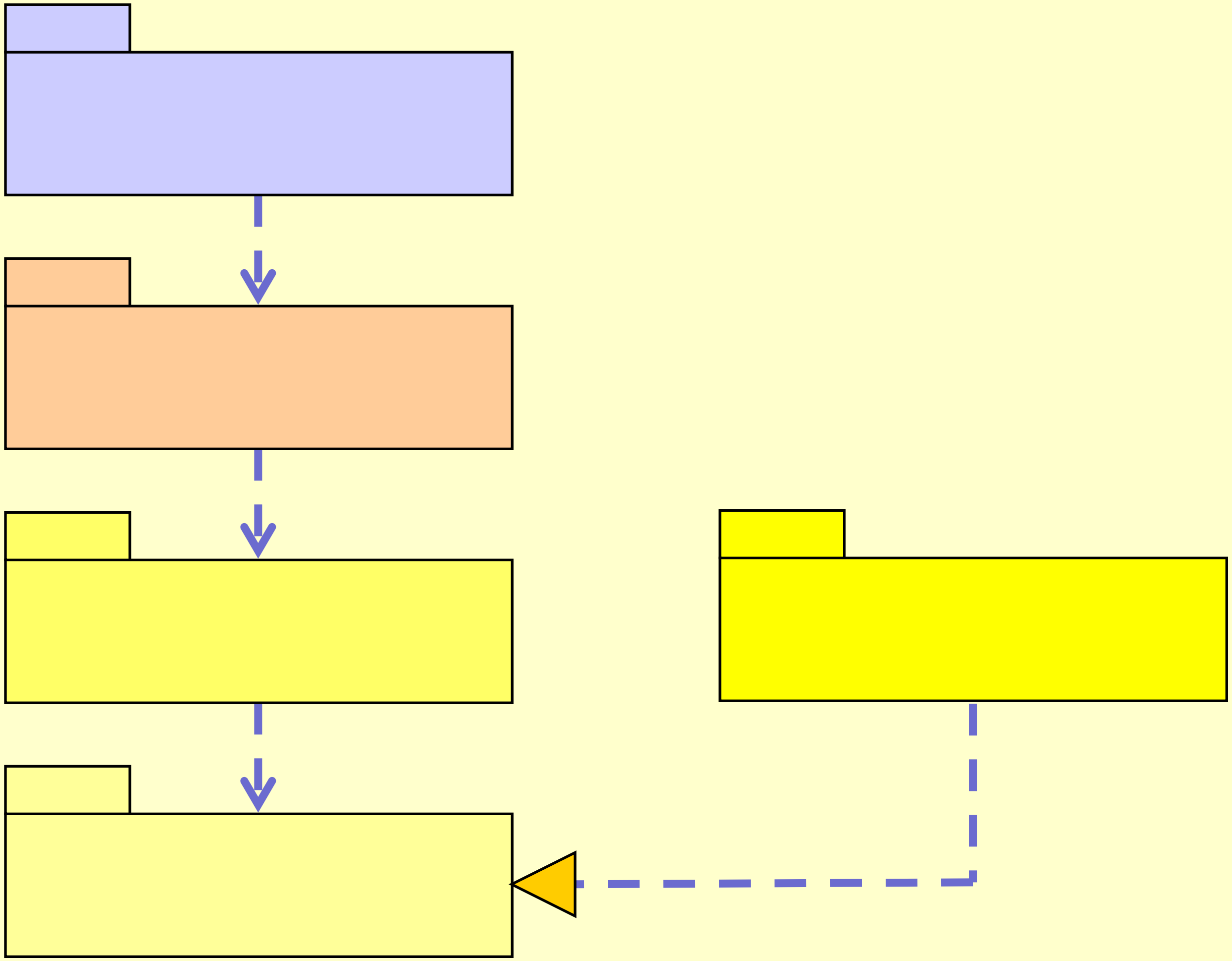
STEWART BRAND

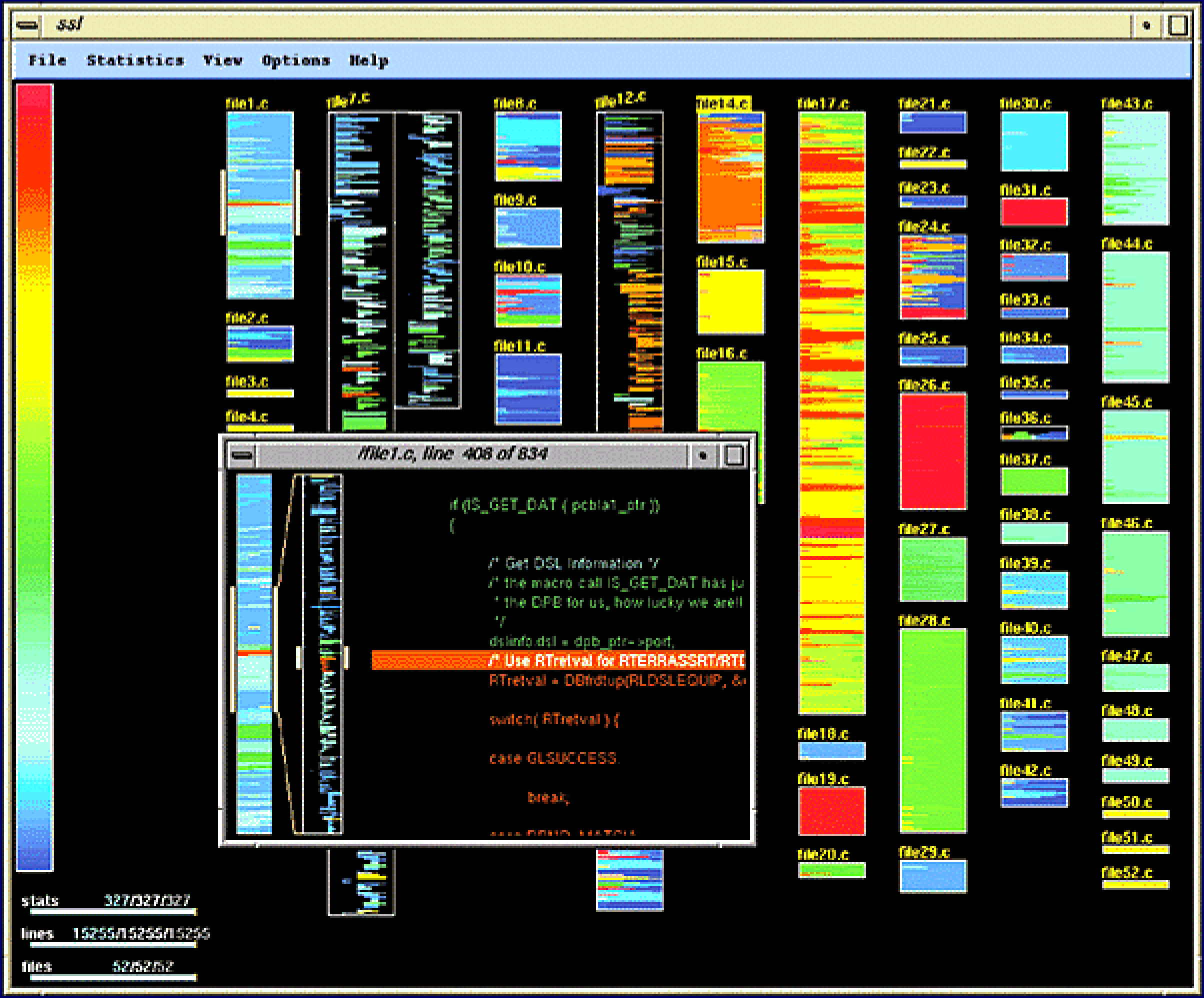


Stewart Brand, *How Buildings Learn*
See also *http://www.laputan.org/mud/*



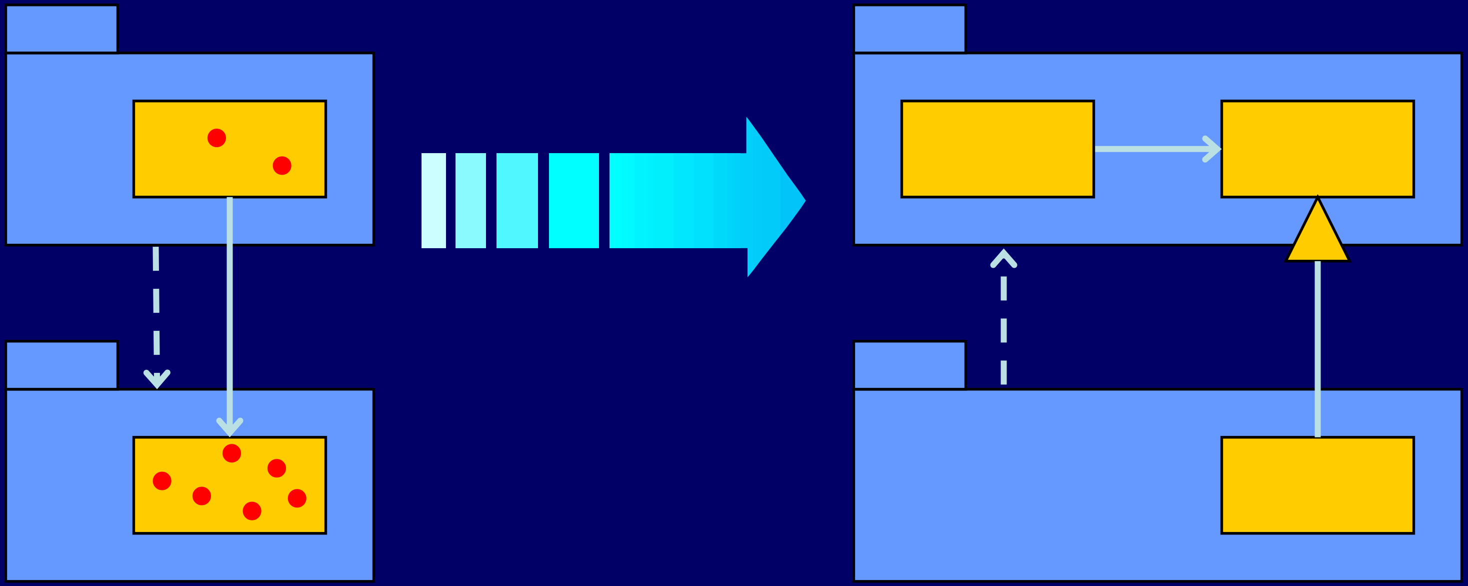
Rate of change





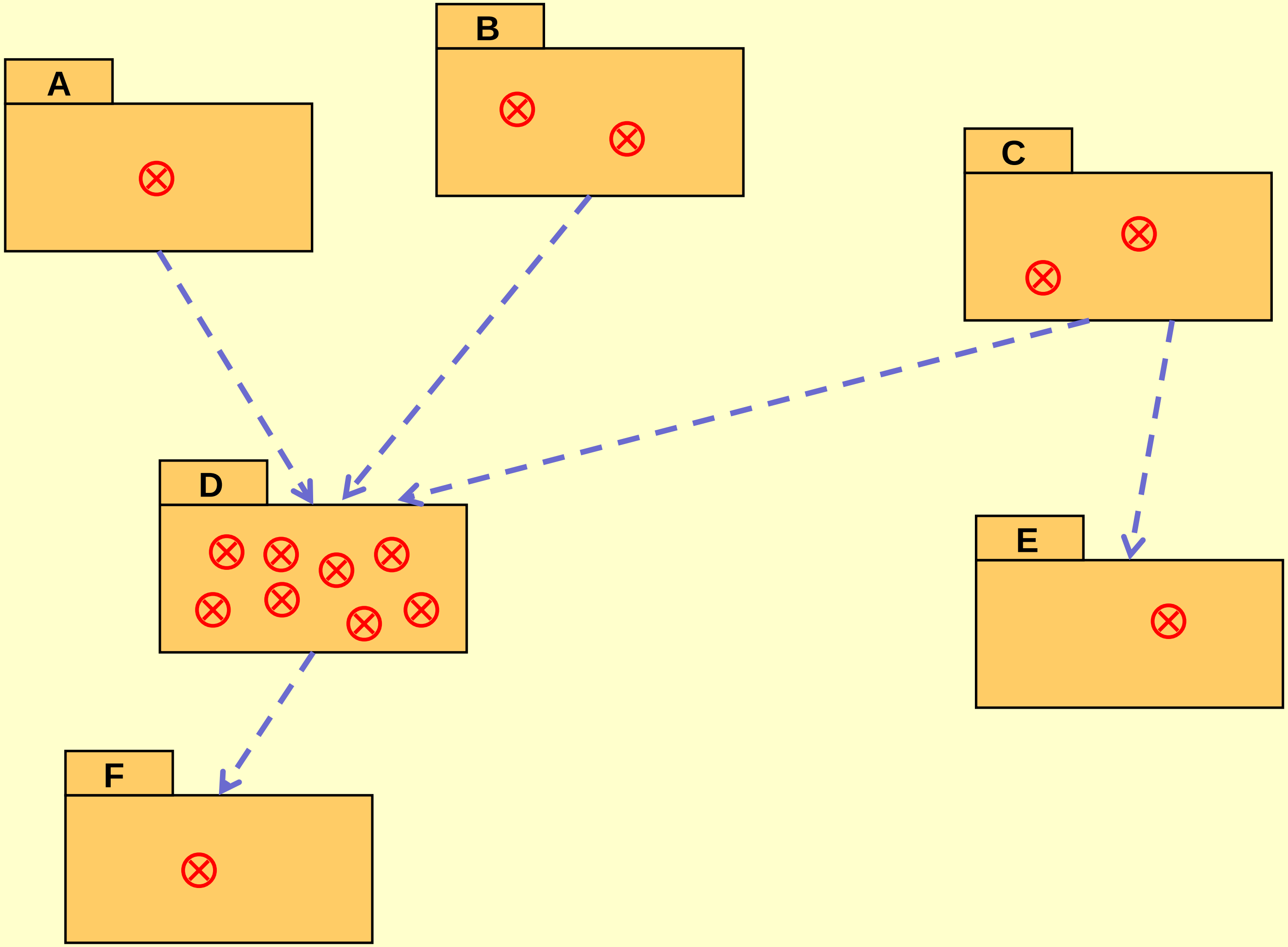
```
if (IS_GET_DAT ( pcblat_ptr ))
{

      /* Get DSL Information */
      /* the macro call IS_GET_DAT has ju
       * the DPB for us, how lucky we are!!
       */
      dslinfo.dsl = dpb_ptr->port;
      /* Use RTretval for RTERRASSRT/RTE
      RTretval = DBfrdtup(RLDSLEQUIP, &

      switch( RTretval ) {

      case GLSUCCESS:

            break;

      case DBNO_MATCH:
```

Thomas Ball and Stephen G Eick
"Software Visualization in the Large"



Scenario buffering by dot-voting possible changes and then readjusting dependencies

BUGRAPPORT


A
B
C
D
E
F

If all you could make was a long-term argument for testing, you could forget about it. Some people would do it out of a sense of duty or because someone was watching over their shoulder. As soon as the attention wavered or the pressure increased, no new tests would get written, the tests that were written wouldn't be run, and the whole thing would fall apart.

Kent Beck
Extreme Programming Explained

How much test coverage should your code have? 80%? 90%? If you've been writing tests from the beginning of your project, you probably have a percentage that hovers around 90%, but what about the typical project? The project which was started years ago, and contains hundreds of thousands of lines of code? Or millions of lines of code? What can we expect from it?

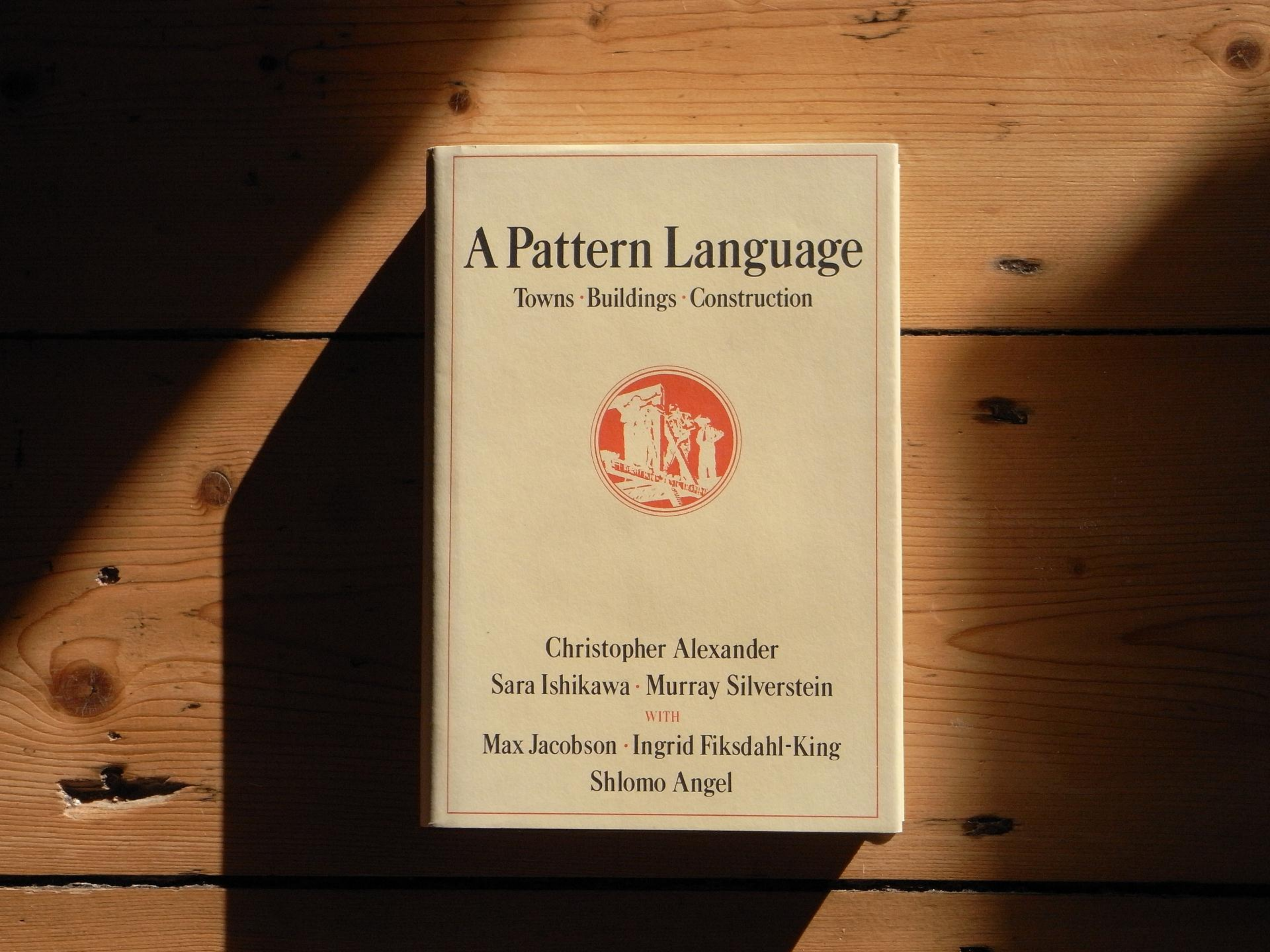
One of the things that I know is that in these code bases, one could spend one's entire working life writing tests without doing anything else. There's simply that much untested code. [...]

Changes occur in clusters in applications. There's some code that you will simply never change and there's other areas of code which change quite often. The other day it occurred to me that we could use that fact to arrive at a better metric, one that is a bit less disheartening and also gives us a sense of our true progress.

Michael Feathers, "A Coverage Metric That Matters"

http://blog.objectmentor.com/articles/2010/05/28/a-coverage-metric-that-matters

All of this has happened before, and it will happen again.



A Pattern Language

Towns · Buildings · Construction

Christopher Alexander

Sara Ishikawa · Murray Silverstein

WITH

Max Jacobson · Ingrid Fiksdahl-King

Shlomo Angel

THE SINGLETON
Single Malt Scotch Whisky
of Dufftown
DUFFTOWN ESTD 1896
MATURED FOR 12 YEARS
12
PRODUCT OF SCOTLAND
DISTILLED MATURED AND BOTTLED IN SCOTLAND
PERFECTLY BALANCED, NATURALLY RICH & SMOOTH
DUFFTOWN DISTILLERY, DUFFTOWN, KEITH
BANFFSHIRE AB55 4BR
1 Litre e
40%vol

Command Pattern
ACTOR EXECUTES COMMANDS FROM
QUEUE IN A SEPARATE THREAD SEQUENTIALLY
COMMAND IS SENT TO THE ACTOR
NEED TO KNOW WHEN THE COMMAND
BEEN EXECUTED
D A "BLOCKING COMMAND" TO THE
OR WAIT FOR THE BLOCKING COMMAND
BE EXECUTED WHEN IT HAS BEEN
ECUTED YOU CAN BE SURE THAT THE
EVIOUS COMMAND WAS ALSO BEEN
ECUTED
EXAMPLE
TEST
SYSTEM
ACTOR
Peppes Pizza
1970
AMERICAN WAY
Margherita
New Orleans
2222 5555
www.peppes.no

The real problem with modular parts is that we took a good idea — modularity — and mixed it up with reuse. Modularity is about separation: When we worry about a small set of related things, we locate them in the same place. This is how thousands of programmers can work on the same source code and make progress. We get in trouble when we try to use that small set of related things in lots of places without preparing or repairing them.

Richard Gabriel
"Mob Software: The Erotic Life of Code"
http://www.dreamsongs.com/MobSoftware.html

repair

refactoring

re-evaluation

remembering

revision

re-engineering

rewriting

retrospection

reduction

reaction

recovery

reuse