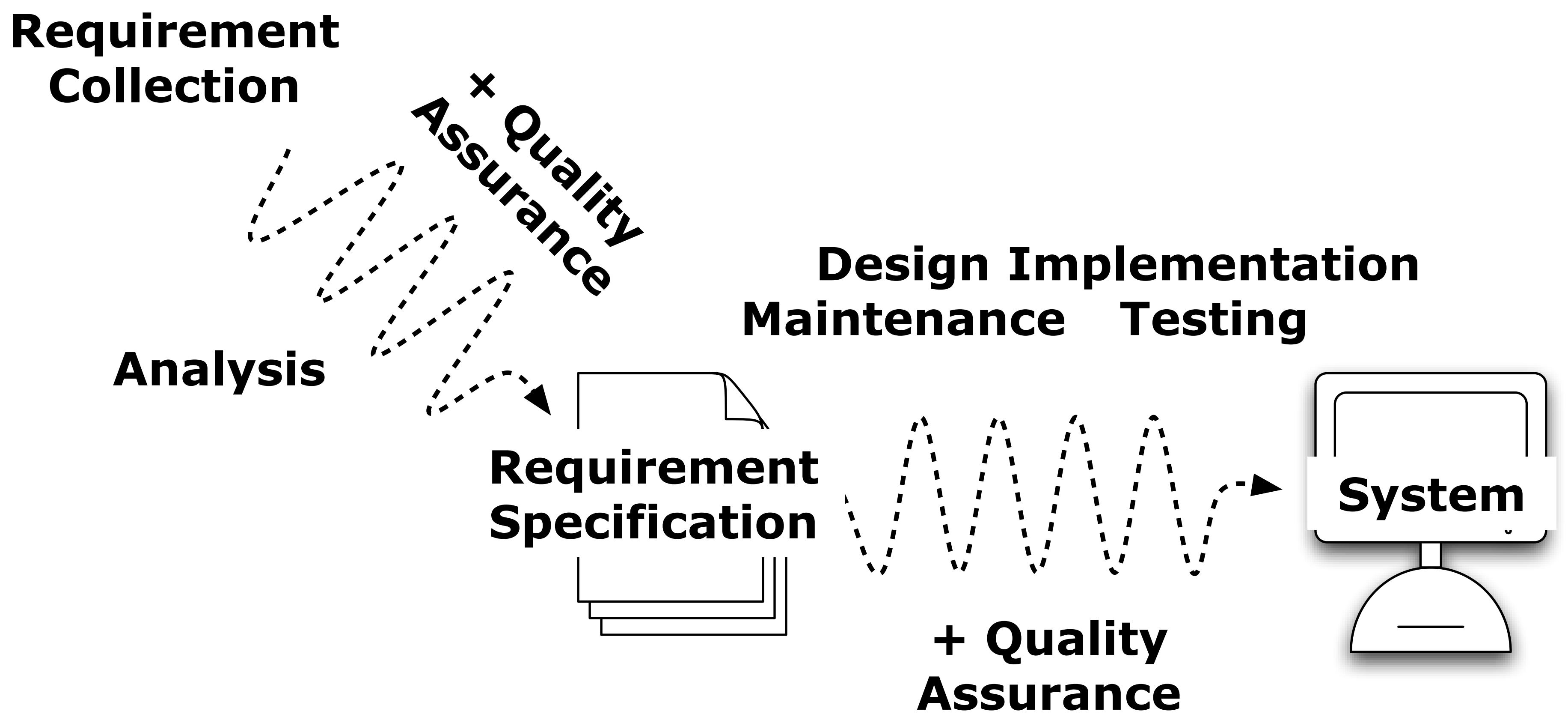
CHAPTER 12 – Conclusion

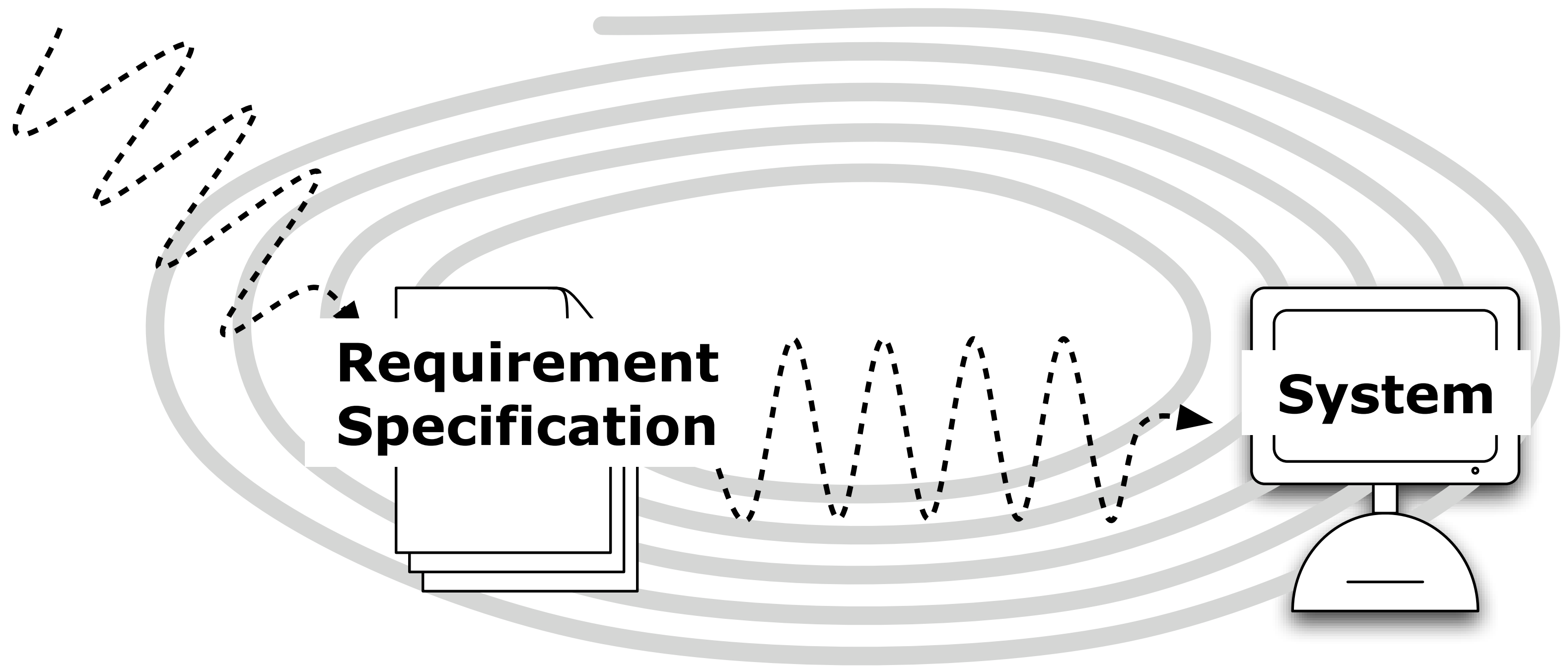
- Overview
  + 1.Introduction
  + 2.Requirements
  + 3.Software Architecture
  + 4.Project Management
  + 5.Design by Contract
  + 6.Testing
  + 7.Formal Specification
  + 8.Domain Modeling
  + 9.Software Quality
  + 10.Software Metrics
  + 11.Refactoring
- Articles
  + The Quest for the Silver Bullet
  + The Case of the Killer Robot
- Professional Ethics
  + Cases
- The future: Software Engineering Tools

Software Product & Process



- **Software Process:**
  + Requirements Collection + Analysis + Design + Implementation
    + Testing + Maintenance + Quality Assurance
- **Software Product:**
  + Requirements Specification (= functional & non-functional)
    + System (= executable + code + documentation)

Evaluation Criteria

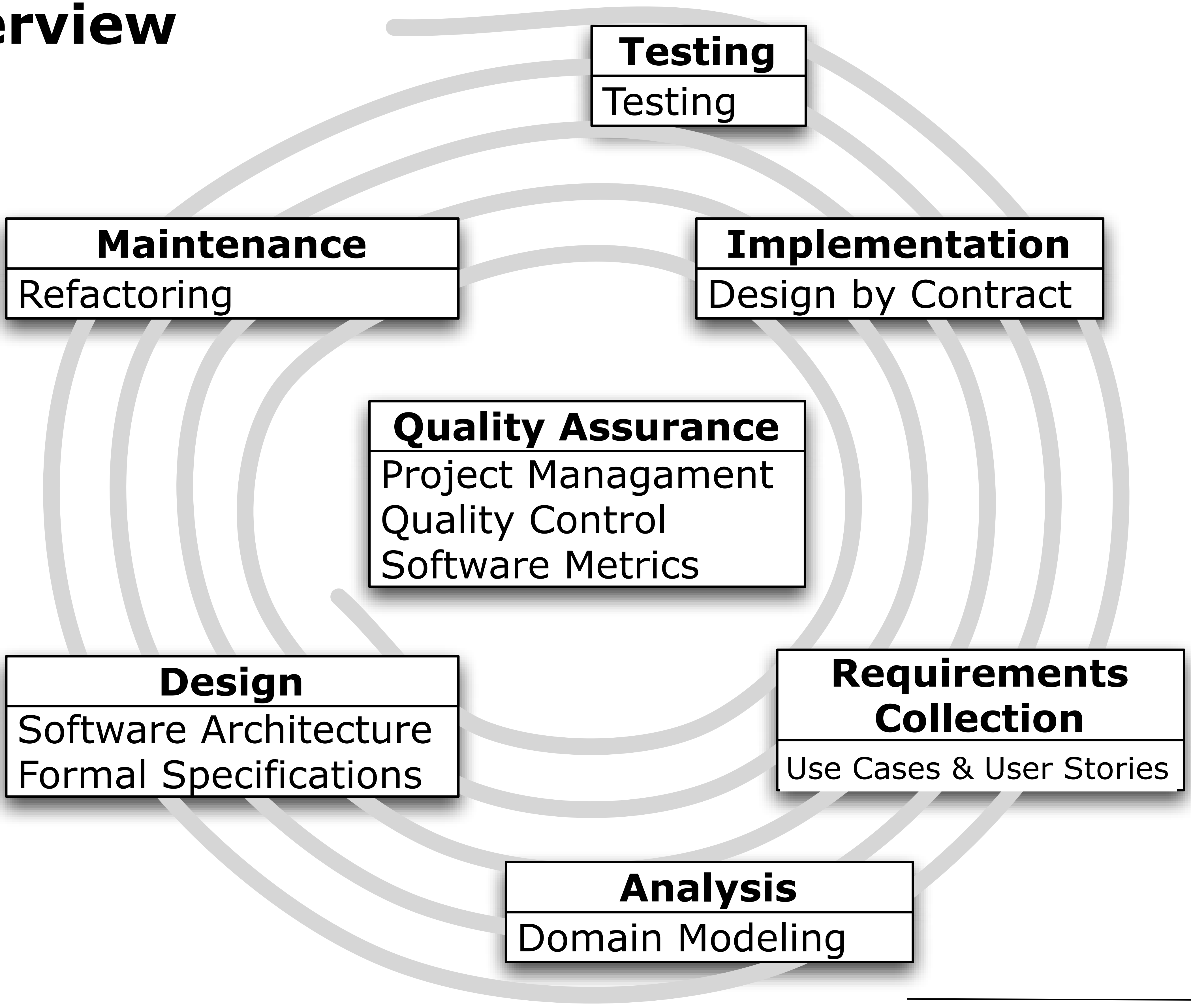


2 evaluation criteria to assess techniques applied during process

Correctness
- Are we building the right product? = VALIDATION
- Are we building the product right? = VERIFICATION

Traceability
- Can we deduce which product components will be affected by changes?

Overview


Testing
Testing
Maintenance
Refactoring
Implementation
Design by Contract
Quality Assurance
Project Managament
Quality Control
Software Metrics
Design
Software Architecture
Formal Specifications
Requirements Collection
Use Cases & User Stories
Analysis
Domain Modeling

Requirements

- Use Cases
  + = Specify expected system behavior as a set of generic scenarios
- User Stories
  + = Express expected functionality with the behaviour driven template
    - As a <user role> I want to <goal> so that <benefit>.

- Are we building the system right?
  + Well specified scenarios help to verify system against requirements
- Are we building the right system?
  + Validation by means of CRC Cards and role playing.
  + Safety Critical ⇒ Failure Mode and Affect Analysis (FMEA)

- Traceability? Requirements ⇔ System
  + Via proper naming conventions

- Traceability? Requirements ⇔ Project Plan
  + Use cases & User stories form good milestones

Software Architecture

- Software Architecture
  + = Components & Connectors describing high-level view of a system.
  + Decomposition implies trade-offs expressed via *coupling* and *cohesion*.
  + Proven solutions to recurring problems are recorded as *patterns*.
- Architecture Tradeoff Analysis Method (ATAM)
  + Review: identify risks, non-risks, sensitivity points and trade-off points

- Are we building the system right?
  + For the *non-functional* parts of the requirements

- Traceability?
  + Extra level of abstraction may hinder traceability

Project Management

- Project Management
  + = plan the work and work the plan
  + PERT and Gantt charts with various options
  + Critical path analysis and monitoring

- Are we building the system right?
  + Deliver what’s required on time within budget
  + Calculate risk to the schedule via optimistic and pessimistic estimates
  + Monitor the critical path to detect delays early
  + Plan to re-plan to meet the deadline

- Traceability? Project Plan ⇔ Requirements & System

  + The purpose of a plan is to detect deviations as soon as possible
  + Small tasks + Milestones verifiable by customer

Design by Contract

- Contractual Obligations Explicitly recorded in Interface
  + pre-condition = obligation to be satisfied by invoking method
  + post-condition = obligation to be satisfied by method being invoked
  + class invariant = obligation to be satisfied by both parties

- Are we building the system right?
  + Recorded obligations prevent defects
  + and ... remain in effect during changes
- Consumer-driven contract testing
  - Test distributed components in isolation via contractual obligations
- Traceability?
  + Obligations express key requirements in source code

- Liskov Substitution Principle?

	stronger	weaker	equal
{I'} vs. {I}			x
{P'} vs. {P}		x	x
{Q'} vs. {Q}	x		x

Testing

- Automated Regression Testing
  + = Deterministic tests (no user intervention), answering whether the system did regress (red = failing tests) or not (green = all tests pass)

- Are we building the system right?
  + Tests only reveal the presence of defects, not their absence
    yet ... Tests verify whether a system is as right as it was before
- Traceability?
  + Link from requirements specification to system source code

- Test techniques
  + Individual test are white box or black box tests
    - **White box**: exploit knowledge of internal structure
      > e.g., path testing, condition testing
    - **Black box**: exploit knowledge about inputs/outputs
      > e.g., input- and output partitioning + boundary conditions
  + Code *Coverage* to measure the strength of a test suite
    - Line - statement - MC/DC - mutation

Formal Specifications

- Input/Output Specifications
  + = include logic assertions (pre- and postconditions + invariants) in algorithm
    > prove assertions via formal reasoning

- State-Based Specifications
  + = Specify acceptable message sequences by means of state machine

- Are we building the system right?
  + Makes verification easier
    > generation of test cases
    > deduction of contractual obligations
- Traceability?
  + Extra intermediate representation may hinder traceability

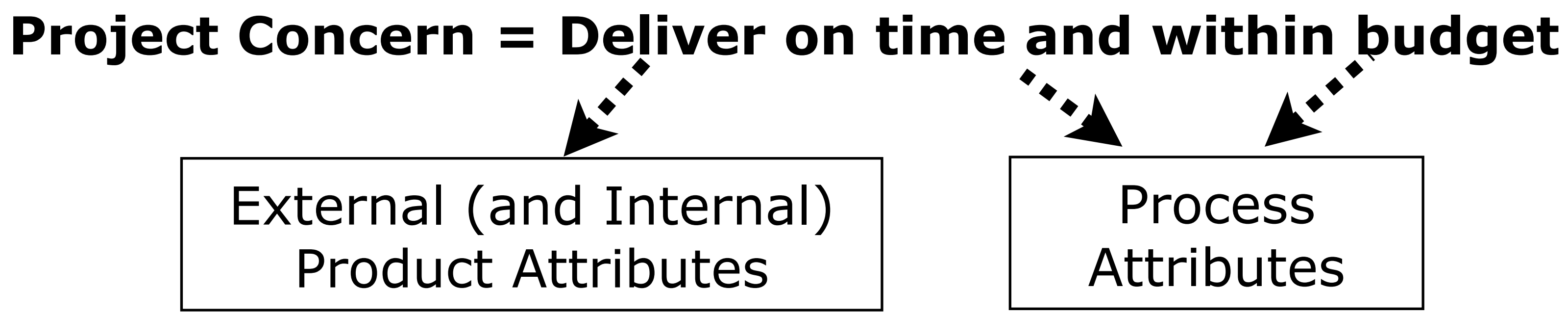
Domain Modeling

- CRC Cards
  + = Analyse system as a set of classes
    - ... each of them having a few responsibilities
    - ... and collaborating with other classes to fulfill these responsibilities
- Feature Model
  + a set of reusable and configurable requirements for specifying system families (a.k.a. product line)

- Are we building the system right?
  + A robust domain model is easier to maintain (= long-term reliability).
- Are we building the right system?
  + CRC Cards and role playing validate use cases.
  + Feature diagrams make product differences (and choices) explicit

- Traceability?
  + Via proper naming conventions

Quality Control


Project Concern = Deliver on time and within budget
External (and Internal) Product Attributes
Process Attributes

- Quality Control
  + = include checkpoints in the process to verify quality attributes
  + Formal technical reviews are very effective and cost effective!
- Quality Standards (ISO9000 and CMM)
  + = Checklists to verify whether a quality system may be certified

- Are we building the system right?
  Are we building the right system?
  + Quality Control eliminates coincidence.
- Traceability?
  + Only when part of the quality plan/system

Software Metrics

- Effort and Cost Estimation
  + = measure early products to estimate costs of later products
  + algorithmic cost modeling, i.e. estimate based on previous experience

- Correctness?
  + Algorithmic cost modeling provides reliable estimates (incl. risk factor)
- Traceability?
  + Quantification of estimates allows for negotiations

- Quality Assurance
  + = quantify the quality model
  + Via internal and external product metrics
- Correctness & Traceability?
  + Software metrics are too premature too assure reliable assessment

Refactoring

- Refactoring Operation
  + = Behaviour-preserving program transformation
  + e.g., rename, move methods and attributes up and down in the hierarchy
- Refactoring Process
  + = Improve internal structure without altering external behaviour
- Code Smell
  + = Symptom of a not so good internal structure
  + e.g, complex conditionals, duplicated code
- Are we building the system right?
  + Behaviour preserving ⇒ as right as it was before (cfr. tests)
- Are we building the right system?
  + Improve internal structure ⇒ cope with requirements mismatches.
- Traceability?
  + Renaming may help to maintain naming conventions
  + Refactoring makes it (too) easy to alter the code without changing the documentation

CHAPTER 12 – Conclusion

- Overview
  + 1.Introduction
  + 2.Requirements
  + 3.Software Architecture
  + 4.Project Management
  + 5.Design by Contract
  + 6.Testing
  + 7.Formal Specification
  + 8.Domain Modeling
  + 9.Software Quality
  + 10.Software Metrics
  + 11.Refactoring
- Articles
  + The Quest for the Silver Bullet
  + The Case of the Killer Robot
- Professional Ethics
  + Cases
- The future: Software Engineering Tools

Assignment: Study an Article of your Choice

- Find and read both of the following articles. Pick the one you liked the most, study it carefully and compare the article with the course contents.

- The Quest for the Silver Bullet
  + [Broo87] Frederick P. Brooks, Jr. “No Silver Bullet: Incidents and Accidents in Software Engineering” IEEE Computer, April 1987.
  + See also [Broo95] Frederick P. Brooks, Jr. “The Mythical Man-Month (20th anniversary edition)” Addison-Wesley.
    - The article is more than 15 years old. Yet, it succeeds in explaining why there will never be an easy solution for solving the problems involved in building large and complex software systems.

- The Killer Robot Case
  + [Epst94] Richard G. Epstein, "The use of computer ethics scenarios in software engineering education: the case of the killer robot.", Software Engineering Education: Proceedings of the 7th SEI CSEE Conference
    - The article is a faked series of newspaper articles concerning a robot which killed its operators due to a software fault. The series of articles conveys the different viewpoints one might have concerning the production of quality software.

Software Engineering & Society

Lives are at stake
(e.g., automatic pilot)

Huge amounts of money
are at stake
(e.g., Ariane V crash)

Software became Ubiquitous
Our society is vulnerable!
⇒ Deontology, Licensing, ...

Corporate success or failure is at stake
(e.g., telephone billing,
VTM launching 2nd channel)

Your personal future is
at stake (e.g., Y2K lawsuits)

Code of Ethics

- Software Engineering Code of Ethics and Professional Practice
  + ACM-site: http://www.acm.org/serving/se/code.htm
  + IEEE-site: http://computer.org/tab/swecc/code.htm
- Recommended by
  + IEEE-CS (Institute of Electrical and Electronics Engineers - Computer Society)
  + ACM (Association for Computing Machinery)

- "Software Engineering Code of Ethics is Approved", Don Gotterbarn, Keith Miller, Simon Rogerson, Communications of the ACM, October 1999, Vol42, no. 10, pages 102-107.
  + Announces the revised 5.2 version of the Code

- "Using the New ACM Code of Ethics in Decision Making", Ronald E. Anderson, Deborah G. Johnson, Donald Gotterbarn, Judith Perrolle, Communications of the ACM, February 1993, Vol36, no. 2, pages 98-104.
  + Discusses 9 cases of situations you might encounter and how (an older version of) the code address them

Code of Ethics: 8 Principles

  + ACM-site: http://www.acm.org/serving/se/code.htm
  + IEEE-site: http://computer.org/tab/swecc/code.htm
- 1. PUBLIC
  + Software engineers shall act consistently with the public interest.
- 2. CLIENT AND EMPLOYER
  + Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.
- 3. PRODUCT
  + Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
- 4. JUDGMENT
  + Software engineers shall maintain integrity and independence in their professional judgment.
- 5. MANAGEMENT
  + Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
- 6. PROFESSION
  + Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
- 7. COLLEAGUES
  + Software engineers shall be fair to and supportive of their colleagues.
- 8. SELF
  + Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

Case: Privacy - Description

- Case Description
  + You consult a company concerning a database for personnel management.
  + Database will include sensitive data: performance evaluations, medical data.
  + System costs too much and company wants to cut back in security.

- What does the code say?
  + 1.03. Approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy or harm the environment. The ultimate effect of the work should be to the public good.
  + 3.12. Work to develop software and related documents that respect the privacy of those who will be affected by that software.

    > Situation is unacceptable.

Case study: Privacy - Solution

- Applicable Clauses
  + 1.02. Moderate the interests of the software engineer, the employer, the client and the users with the public good.
  + 1.04. Disclose to appropriate persons or authorities any actual or potential danger to the user, the public, or the environment, that they reasonably believe to be associated with software or related documents.
  + 2.07. Identify, document, and report significant issues of social concern, of which they are aware, in software or related documents, to the employer or the client.
  + 6.09. Ensure that clients, employers, and supervisors know of the software engineer's commitment to this Code of ethics, and the subsequent ramifications of such commitment.

- Actions
  + Try to convince management to keep high security standards.
  + Include in contract a clause to cancel contract when against the code of ethics.
  + Alarm other institutions if you later hear that others accepted the contract.

Case study: Privacy - Solution

If you are an independent consultant, how can you ensure that you will not have to act against the code of ethics?

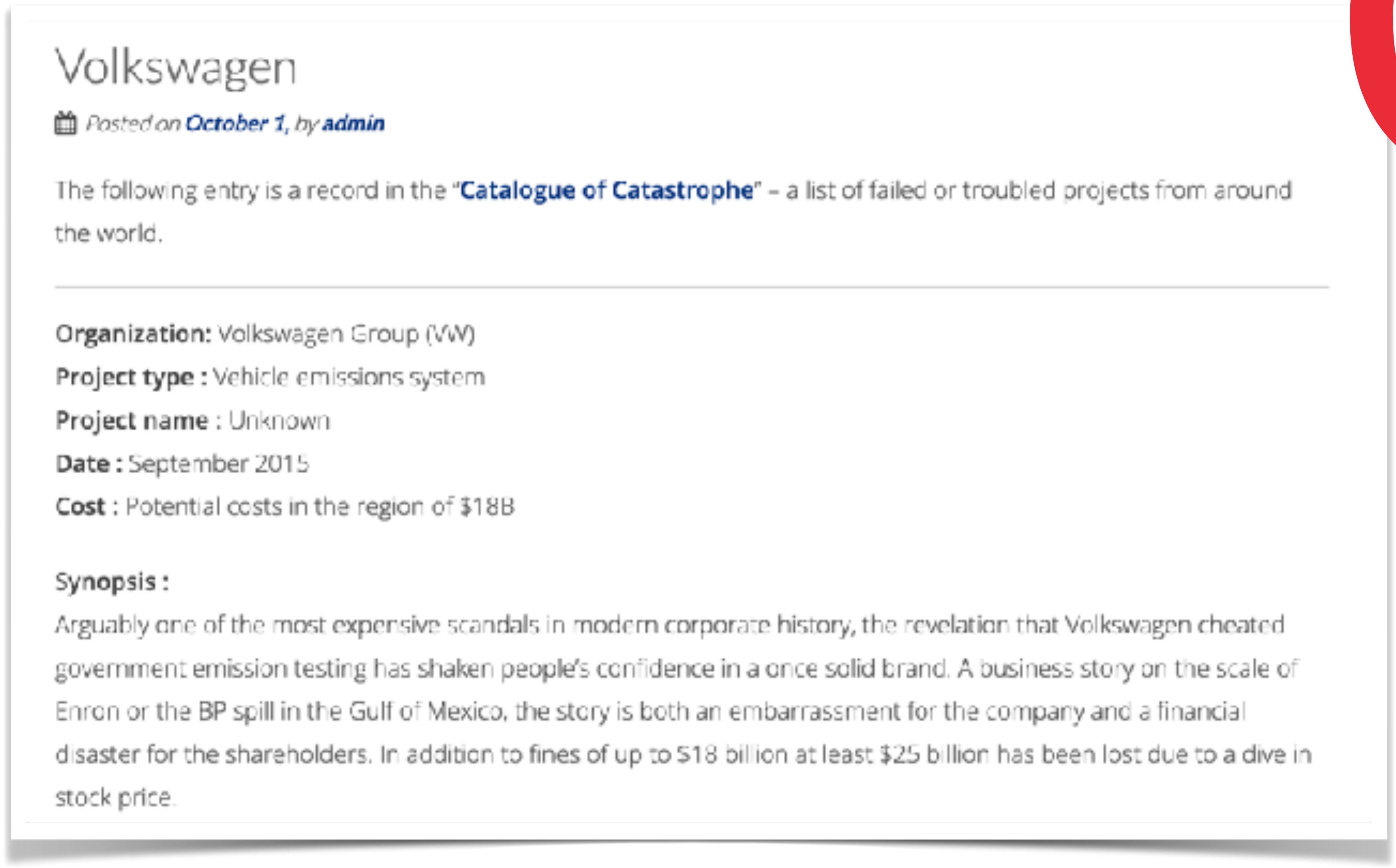
- Actions
  - + …
  - + Include in contract a clause to cancel contract when against the code of ethics.
  - + …

Case: Unreliability

- Case Description
  + You’re the team leader of a team building software for calculating taxes.
  + Your team and your boss are aware that the system contains a lot of defects. Consequently you state that the product can’t be shipped in its current form.
  + Your boss ships the product anyway, with a disclaimer “Company X is not responsible for errors resulting from the use of this program”.

- What does the code say?
  + 1.03. Approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy or harm the environment. The ultimate effect of the work should be to the public good.
  + 5.11. Not ask a software engineer to do anything inconsistent with this Code.
  + 5.12. Not punish anyone for expressing ethical concerns about a project.
    > Disclaimer does not apply: can only be made in “good conscience”.
    > In court you can not be held liable.

VW emissions scandal



Volkswagen

Posted on October 1, by admin

The following entry is a record in the "Catalogue of Catastrophe" – a list of failed or troubled projects from around the world.

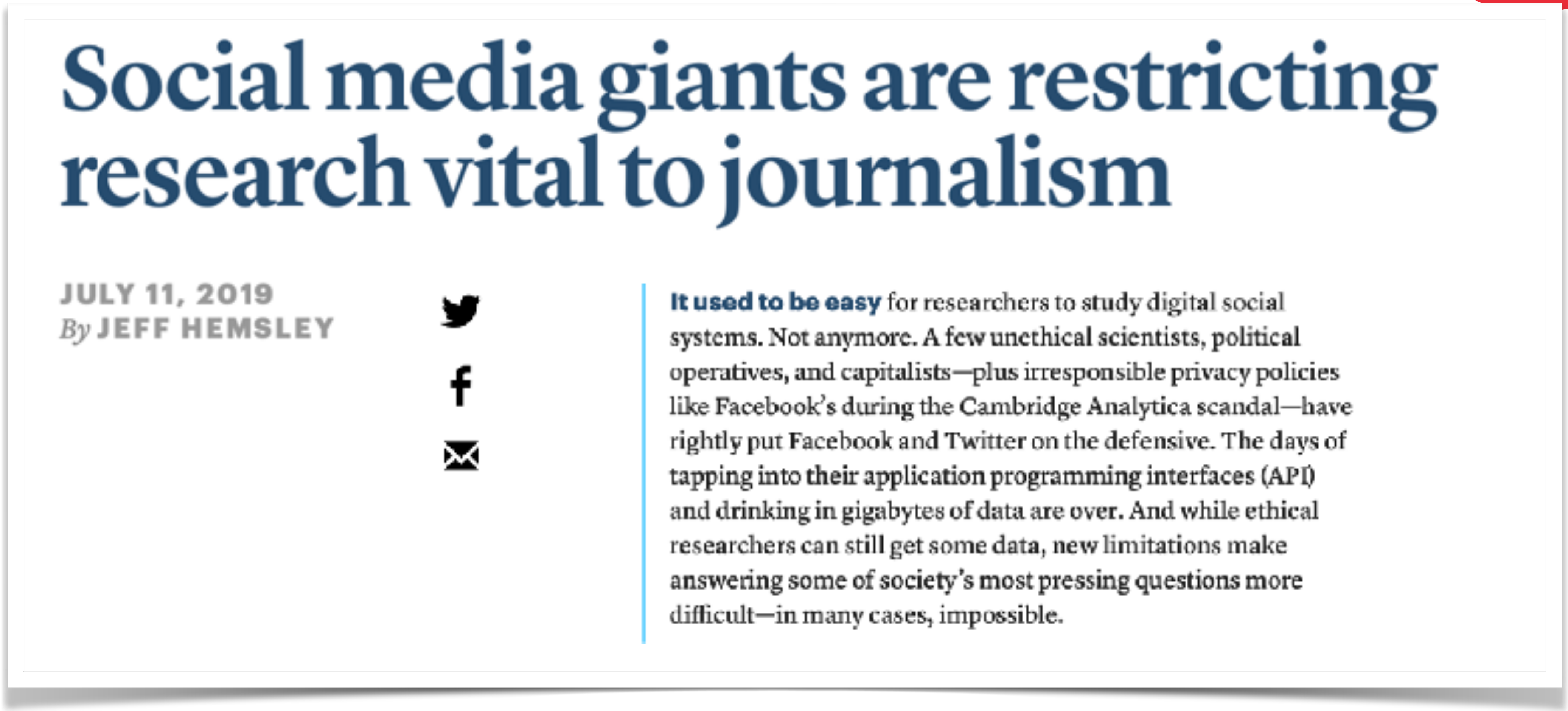
Organization: Volkswagen Group (VW)
Project type : Vehicle emissions system
Project name : Unknown
Date : September 2015
Cost : Potential costs in the region of $18B

Synopsis :
Arguably one of the most expensive scandals in modern corporate history, the revelation that Volkswagen cheated government emission testing has shaken people's confidence in a once solid brand. A business story on the scale of Enron or the BP spill in the Gulf of Mexico, the story is both an embarrassment for the company and a financial disaster for the shareholders. In addition to fines of up to $18 billion at least $25 billion has been lost due to a dive in stock price.

Your mission should you choose to accept.

- You are a software engineer working for volkswagen. Your management asks to install a so called "defeat device" into the car to circumvent emission tests.

Facebook / Twitter API



Social media giants are restricting research vital to journalism

JULY 11, 2019
By JEFF HEMSLEY

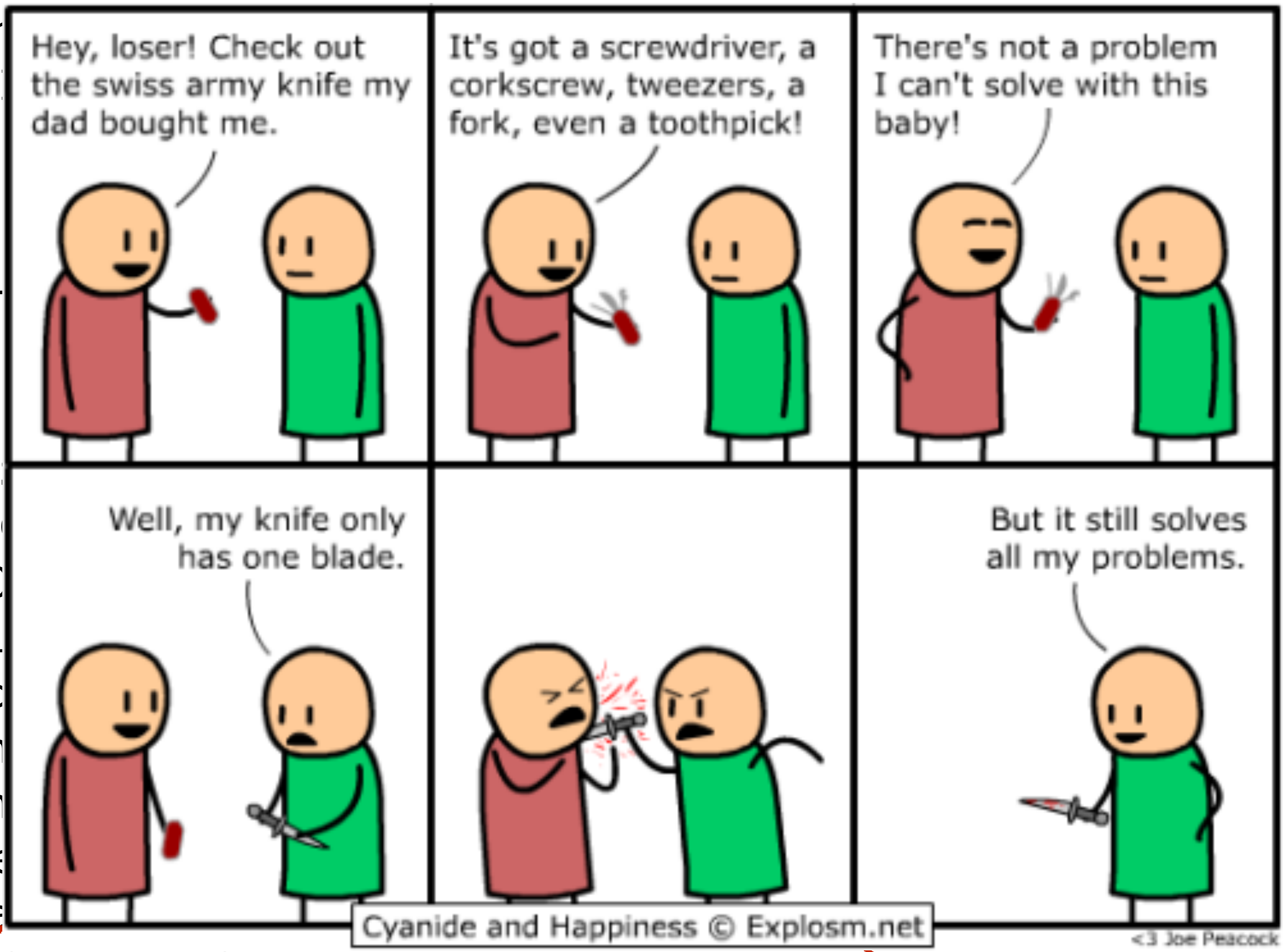
It used to be easy for researchers to study digital social systems. Not anymore. A few unethical scientists, political operatives, and capitalists—plus irresponsible privacy policies like Facebook's during the Cambridge Analytica scandal—have rightly put Facebook and Twitter on the defensive. The days of tapping into their application programming interfaces (API) and drinking in gigabytes of data are over. And while ethical researchers can still get some data, new limitations make answering some of society's most pressing questions more difficult—in many cases, impossible.

Your mission should you choose to accept.

- You are a master thesis student and you are asked to inject "spy software" on the API of big social media for research purposes.

CHAPTER 12 – Conclusion

- Over
  + 1.
  + 2.
  + 3.
  + 4.
  + 5.
  + 6.
  + 7.
  + 8.
  + 9.
  + 10
  + 11
- Artic
  + Th
  + Th
- Profe
  + Ca
- The future: Software Engineering Tools


Hey, loser! Check out the swiss army knife my dad bought me.
It's got a screwdriver, a corkscrew, tweezers, a fork, even a toothpick!
There's not a problem I can't solve with this baby!
Well, my knife only has one blade.
But it still solves all my problems.
Cyanide and Happiness © Explosm.net
<3 Joe Peacock

Innovation

Underlying Technology

1908 — patent on paper filter

1946 — commerical piston espresso machine

2000 — nespresso

2001 — senseo

Business Models

1475 — Kiva Han coffee house (Constantinople)

(Vienna)
1529 — European coffee house

1971 — Starbucks (seattle)

Innovation

Underlying Technology

1908 — patent on paper filter

1940 — commerical piston espresso machine

2000 — nespresso
2001 — senseo

Business Models

1475 — Kiva Han coffee house (Constantinople)

(Vienna)
1529 — European coffee house

1971 — Starbucks (seattle)

Technology changes every 20 years
...
Underlying business models rarely change!

Innovation in ICT

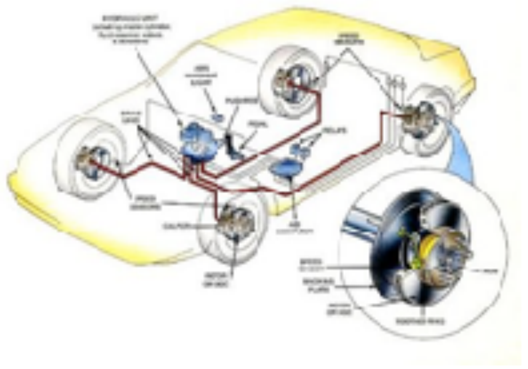
ENIAC, 1945

IBM PC, 1981

NEC ultralite, 1989

iPad, 2010

Underlying Technology



Embedded

Google www.taxonweb.be YouTube Drupal facebook twitter

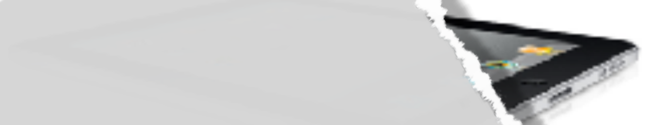
Internet

Innovation in ICT

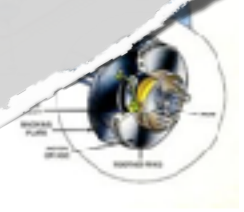
ENIAC, 1945

IBM PC, 1981

NEC ultralite, 1989


iPad, 2010

Underlying Technology



Embedded

Google YouTube Drupal facebook twitter

Internet

Technology changes every 5 years

...

Underlying business models change often!

Market pressure in ICT

YAHOO!

altavista

Google

Measure of innovation

- # products in portfolio younger than 5 years
  + in ICT usually more than 1/2 the portfolio

Significant investment in R&D

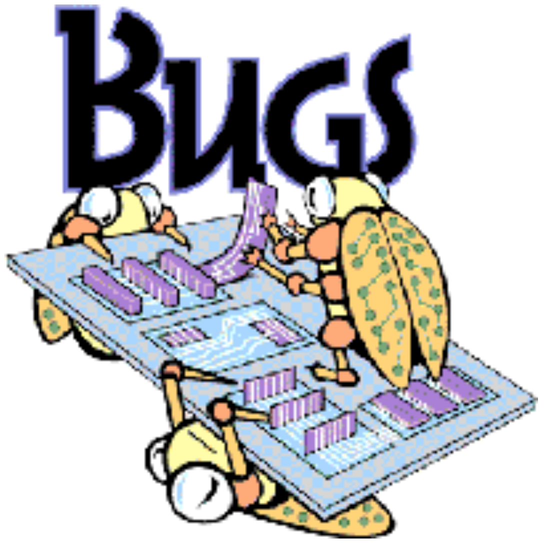
- more products … faster

RELIABILITY ⟷ AGILITY

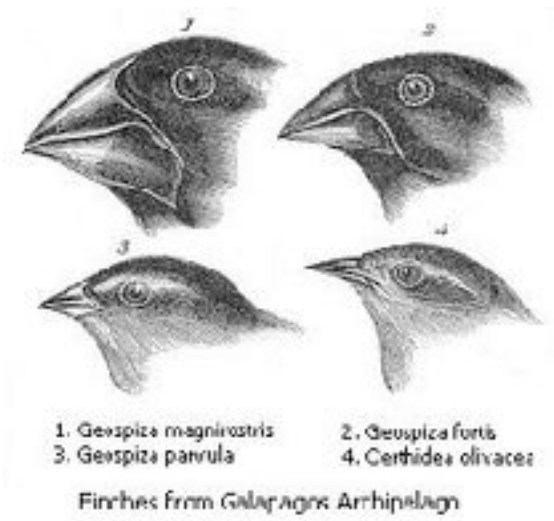
Reliability vs. Agility

Software is vital to our society ⇒ Software must be reliable

Traditional Software Engineering
Reliable = Software without bugs



Today's Software Engineering
Reliable = Easy to Adapt



On the Origin of Species

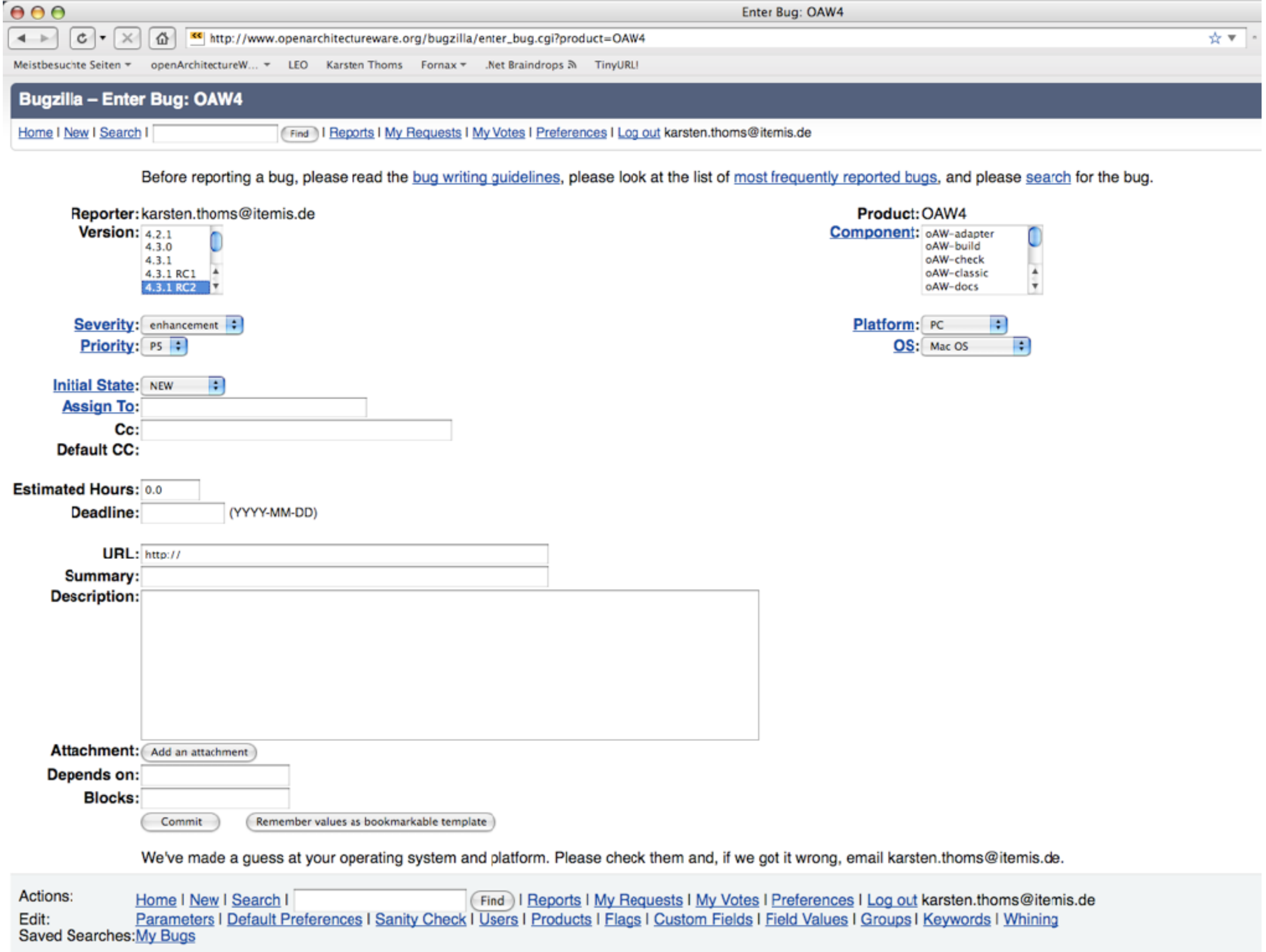
Striving for RELIABILITY

(Optimise for *perfection*)

⟷

Striving for AGILITY

(Optimise for *development speed*)



Enter Bug: OAW4

http://www.openarchitectureware.org/bugzilla/enter_bug.cgi?product=OAW4

Meistbesuchte Seiten | openArchitectureW... | LEO | Karsten Thoms | Fornax | .Net Braindrops | TinyURL!

Bugzilla – Enter Bug: OAW4

Home | New | Search | [Find] | Reports | My Requests | My Votes | Preferences | Log out karsten.thoms@itemis.de

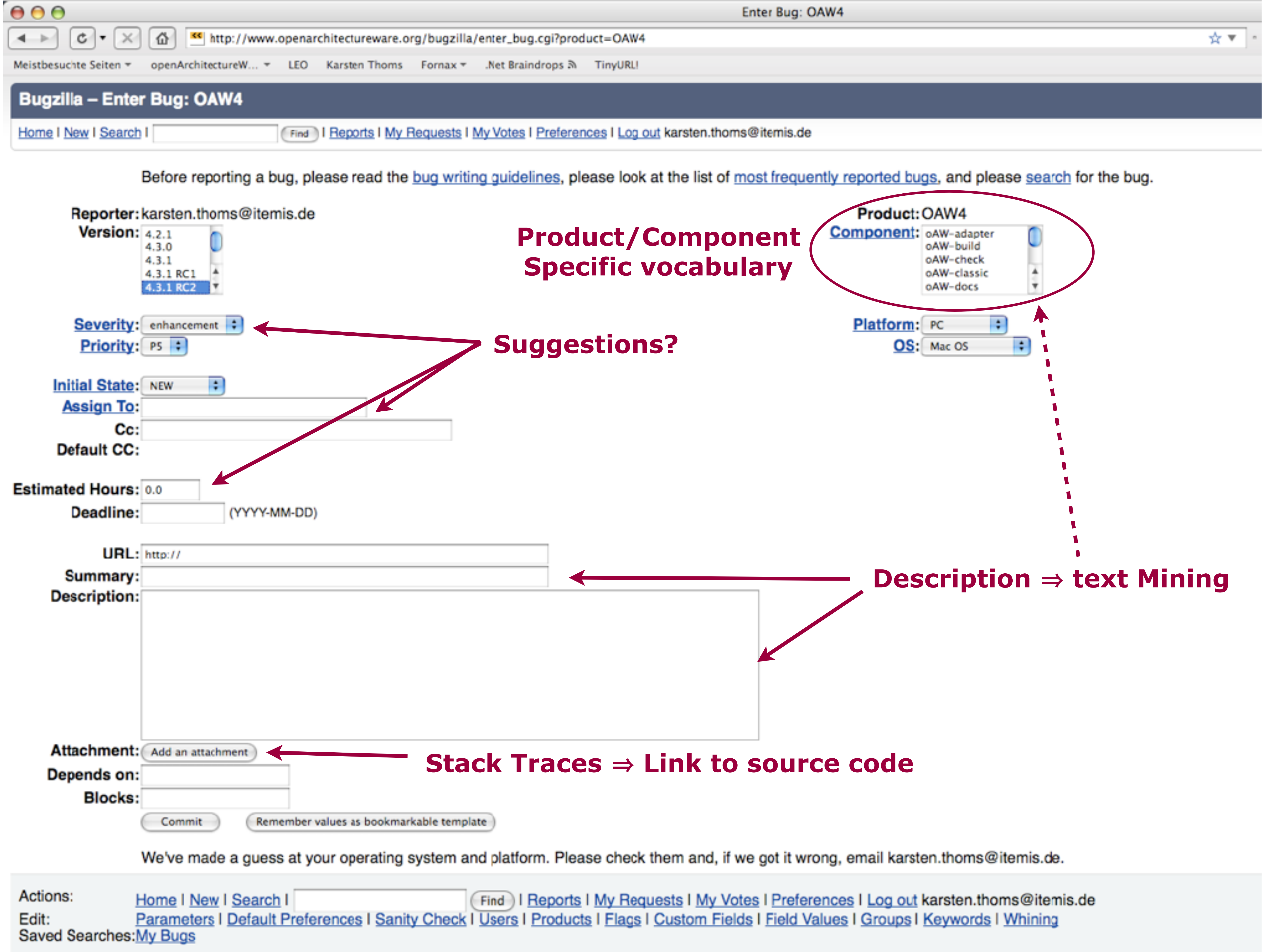
Before reporting a bug, please read the bug writing guidelines, please look at the list of most frequently reported bugs, and please search for the bug.

Reporter:	karsten.thoms@itemis.de	Product:	OAW4
Version:	4.2.1, 4.3.0, 4.3.1, 4.3.1 RC1, 4.3.1 RC2 (selected)	Component:	oAW-adapter, oAW-build, oAW-check, oAW-classic, oAW-docs
Severity:	enhancement	Platform:	PC
Priority:	P5	OS:	Mac OS
Initial State:	NEW		
Assign To:			
Cc:			
Default CC:			
Estimated Hours:	0.0		
Deadline:	(YYYY-MM-DD)		
URL:	http://		
Summary:			
Description:			
Attachment:	Add an attachment		
Depends on:			
Blocks:			

[Commit] [Remember values as bookmarkable template]

We've made a guess at your operating system and platform. Please check them and, if we got it wrong, email karsten.thoms@itemis.de.

Actions: Home | New | Search | [Find] | Reports | My Requests | My Votes | Preferences | Log out karsten.thoms@itemis.de
Edit: Parameters | Default Preferences | Sanity Check | Users | Products | Flags | Custom Fields | Field Values | Groups | Keywords | Whining
Saved Searches: My Bugs



Enter Bug: OAW4

http://www.openarchitectureware.org/bugzilla/enter_bug.cgi?product=OAW4

Meistbesuchte Seiten | openArchitectureW... | LEO | Karsten Thoms | Fornax | .Net Braindrops | TinyURL!

Bugzilla – Enter Bug: OAW4

Home | New | Search | Find | Reports | My Requests | My Votes | Preferences | Log out karsten.thoms@itemis.de

Before reporting a bug, please read the bug writing guidelines, please look at the list of most frequently reported bugs, and please search for the bug.

Reporter:	karsten.thoms@itemis.de
Version:	4.2.1, 4.3.0, 4.3.1, 4.3.1 RC1, [✓ 4.3.1 RC2]
Severity:	enhancement
Priority:	P5
Initial State:	NEW
Assign To:	
Cc:	
Default CC:	
Estimated Hours:	0.0
Deadline:	(YYYY-MM-DD)
URL:	http://
Summary:	
Description:	
Attachment:	Add an attachment
Depends on:	
Blocks:	

Product:	OAW4
Component:	oAW-adapter, oAW-build, oAW-check, oAW-classic, oAW-docs
Platform:	PC
OS:	Mac OS

Commit | Remember values as bookmarkable template

We've made a guess at your operating system and platform. Please check them and, if we got it wrong, email karsten.thoms@itemis.de.

Actions: Home | New | Search | Find | Reports | My Requests | My Votes | Preferences | Log out karsten.thoms@itemis.de

Edit: Parameters | Default Preferences | Sanity Check | Users | Products | Flags | Custom Fields | Field Values | Groups | Keywords | Whining

Saved Searches: My Bugs

Product/Component Specific vocabulary
Suggestions?
Description ⇒ text Mining
Stack Traces ⇒ Link to source code

Bug Report Triaging

Artificial Intelligence Inside

Question	Cases	Precision	Recall
Who should fix this bug?	Eclipse, Firefox, gcc	eclipse: 57% firefox: 64% gcc: 6%	—
How long will it take to fix this bug?	JBoss	depends on the component many similar reports: off by one hour few similar reports: off by 7 hours	
What is the severity of this bug?	Mozilla, Eclipse, Gnome	mozilla, eclipse:67% - 73% gnome: 75%-82%	mozilla, eclipse:50% - 75% gnome: 68%-84%

Bug Report Triaging

Artificial Intelligence Inside

Question	Cases	Precision	Recall
~~Who should fix this bug?~~	~~x, gcc~~	~~eclipse: 57%~~ ~~firefox: 64%~~ ~~gcc: 6%~~	—
How long will it take to fix this bug?	JBoss	depends on the component many similar reports: off by one hour few similar reports: off by 7 hours	
What is the severity of this bug?	me	mozilla, eclipse:67% - 73% gnome: 75%-82%	mozilla, eclipse:50% - 75% gnome: 68%-84%

Irrelevant for Practitioners

Internal vs. External Bug Reports

Story Points (Planning Poker)

1/2	1	2	3	5	8	13	20	40	100	∞

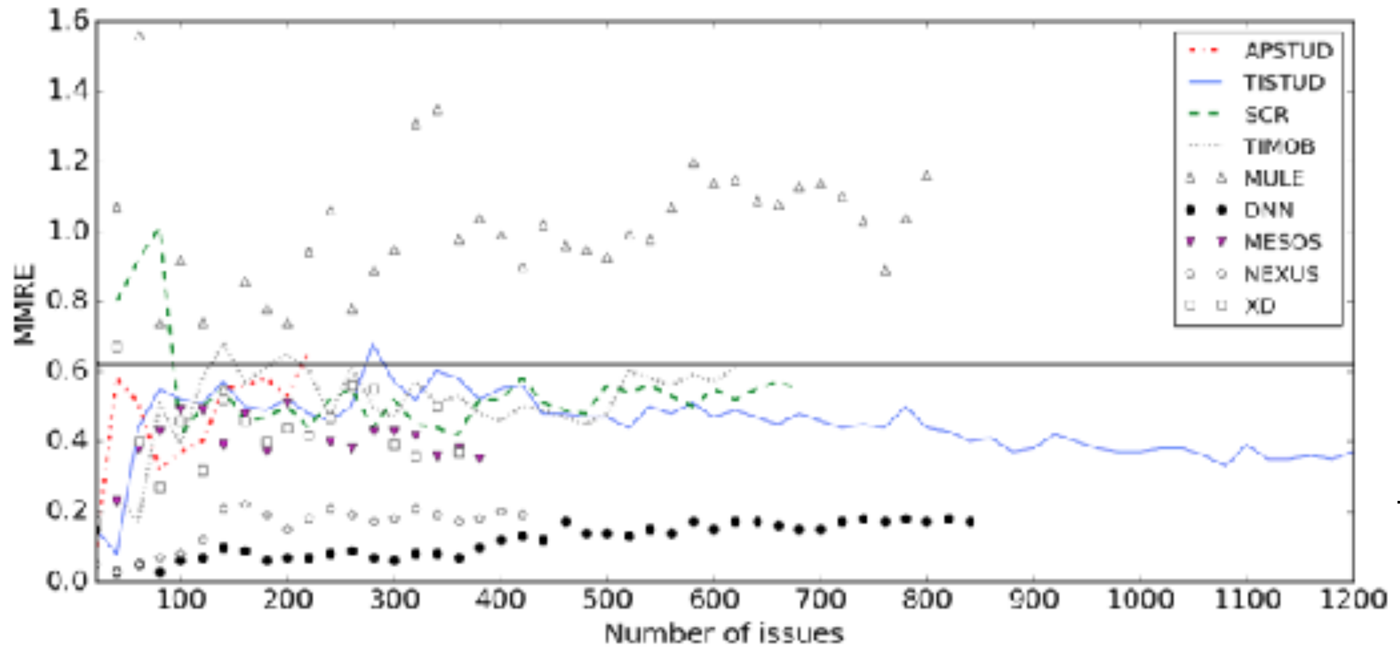
Results

Project	Total est.	Correct est.	MMRE
APSTUD	228	143	0.61
XD	360	223	0.42
MESOS	387	223	0.39
NEXUS	421	341	0.16
TIMOB	634	380	0.6
SCR	**699**	**413**	**0.5**
MULE	805	416	1.07
DNN	858	669	0.16
TISTUD	1215	810	0.35
mean	*623*	*402*	*0.47*

Human
MMRE: **0.48**

(*) Mean Magnitude
of Relative Error

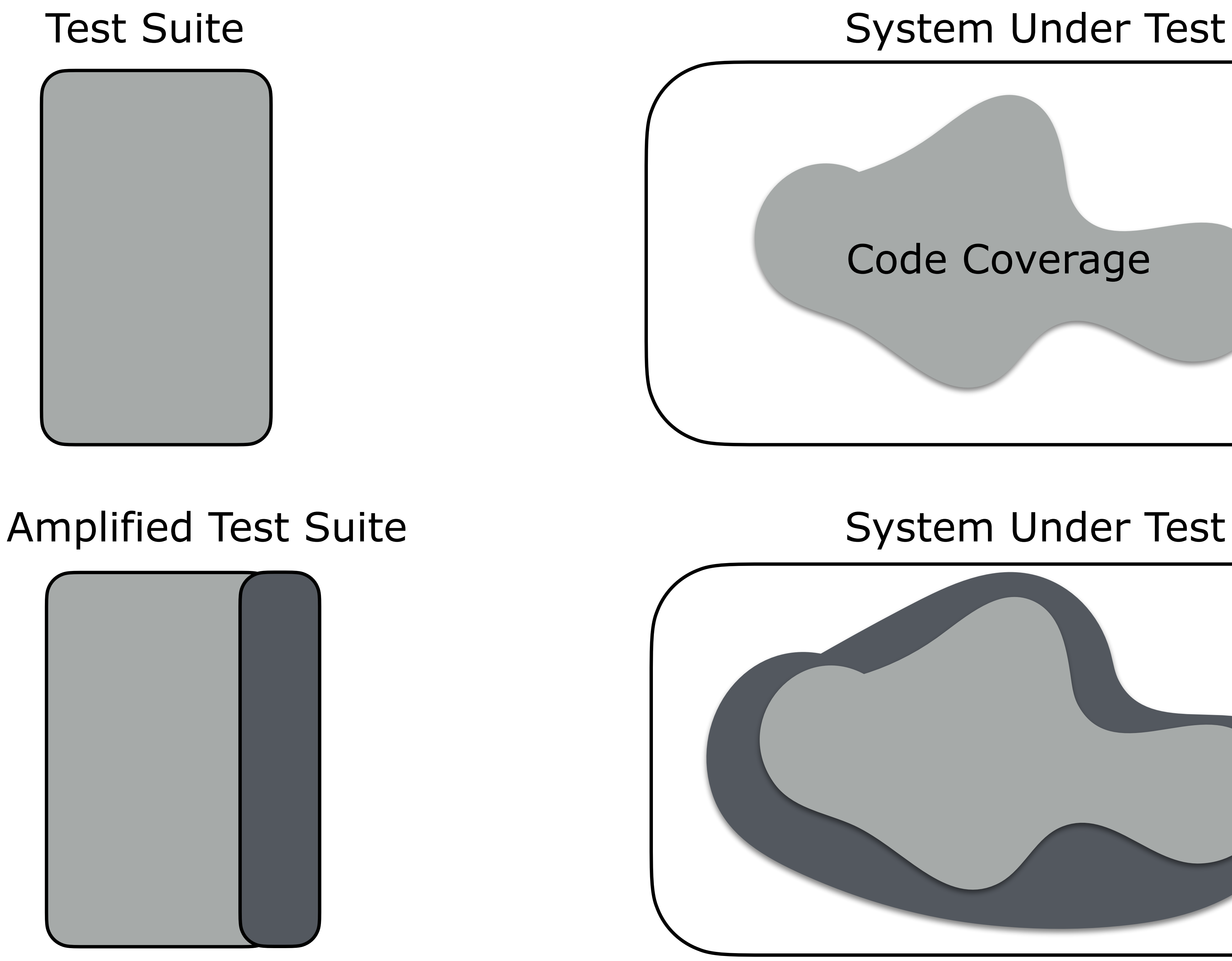
Learning Curve



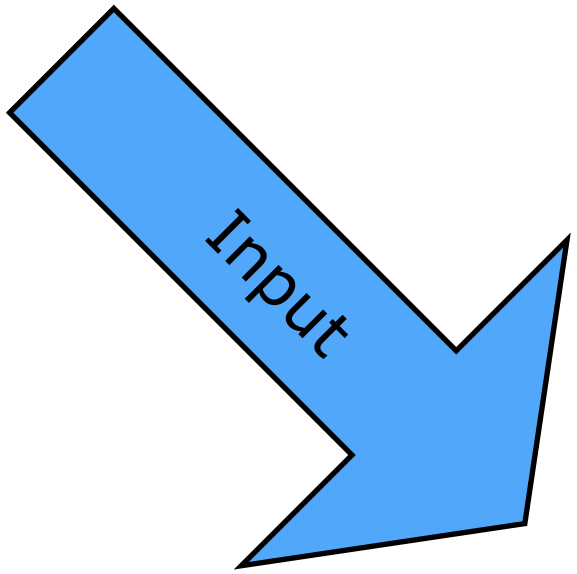
12.Conclusion

“in vivo” Validation

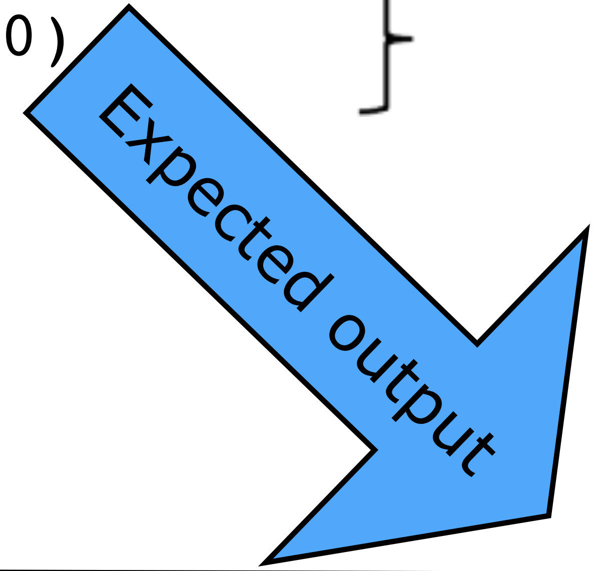
Test Amplification



Example - testDeposit



```
def testDeposit (self) :
  self.b.set_owner('Iwena Kroka')
  self.b.deposit(10)
  self.assertEqual(self.b.get_balance(), 10)
  self.b.deposit(100)
  self.b.deposit(100)
  self.assertEqual(self.b.get_balance() , 210)
```



Example - testDeposit_amplified (1/2)

Assertion Amplification

```
def testDeposit_amplified (self) :
  self.b.set_owner('Iwena Kroka')
  self.b.deposit(10)
  self.assertEqual(self.b.
          get_transactions(), [10])
  self.assertFalse(self.b.is_empty () )
  self.assertEqual(self.b.owner, 'Iwena Kroka')
  self.assertEqual(self.b.get_balance(), 10)
  …
```

Assertion Amplification = (re)generate appropriate assertions to verify the actual state of the object under test by observing the run-time behaviour.

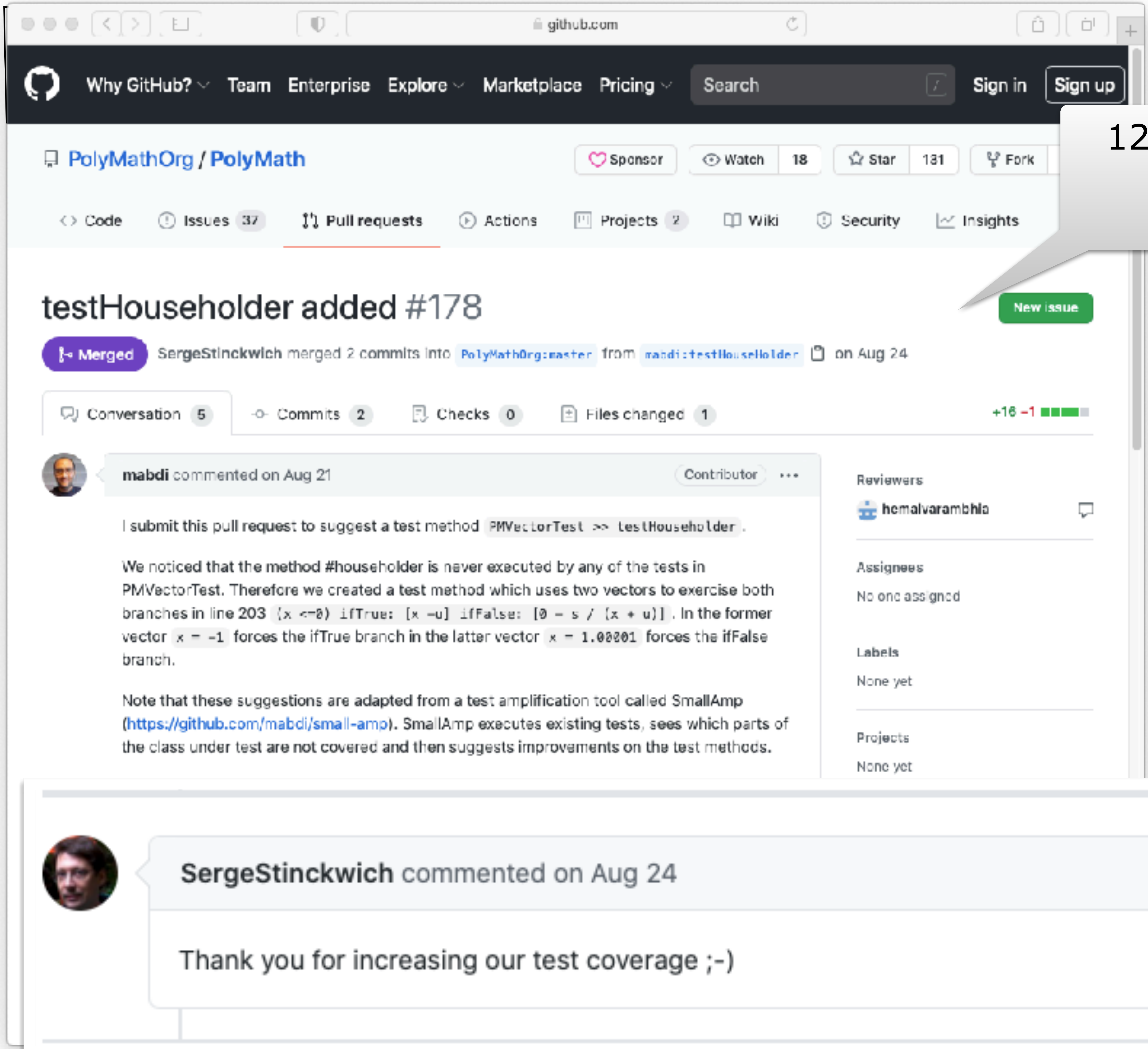
Example - testDeposit_amplified (2/2)

Input Amplification

```
def testDeposit_amplified (self) :
  self.b.set_owner('Iwena Kroka')
  self.b.deposit(10)
  self.assertEqual(self.b.
          get_transactions(), [10])
  self.assertFalse(self.b.is_empty () )
  self.assertEqual(self.b.owner, 'Iwena Kroka')
  self.assertEqual(self.b.get_balance(), 10)
  with self.assertRaises(Exception):
     self.b.deposit(-56313)
  self.b.deposit(100)
  self.b.set_owner('Guido van Rossum')
  self.assertEqual(self.b.
           get_transactions(), [10])
…
```

Input Amplification = Transform the original test method(*); forcing previously untested paths.
(*) Change the set-up of the object under test, providing parameters that represent boundary conditions; inject calls to state-changing methods

⇒ Brute force but optimize via increase in code coverage


github.com
Why GitHub? Team Enterprise Explore Marketplace Pricing Search Sign in Sign up
PolyMathOrg / PolyMath
Sponsor Watch 18 Star 131 Fork
Code Issues 37 Pull requests Actions Projects 2 Wiki Security Insights
New issue

testHouseholder added #178

Merged SergeStinckwich merged 2 commits into `PolyMathOrg:master` from `mabdi:testHouseholder` on Aug 24

Conversation 5 · Commits 2 · Checks 0 · Files changed 1 · +16 −1

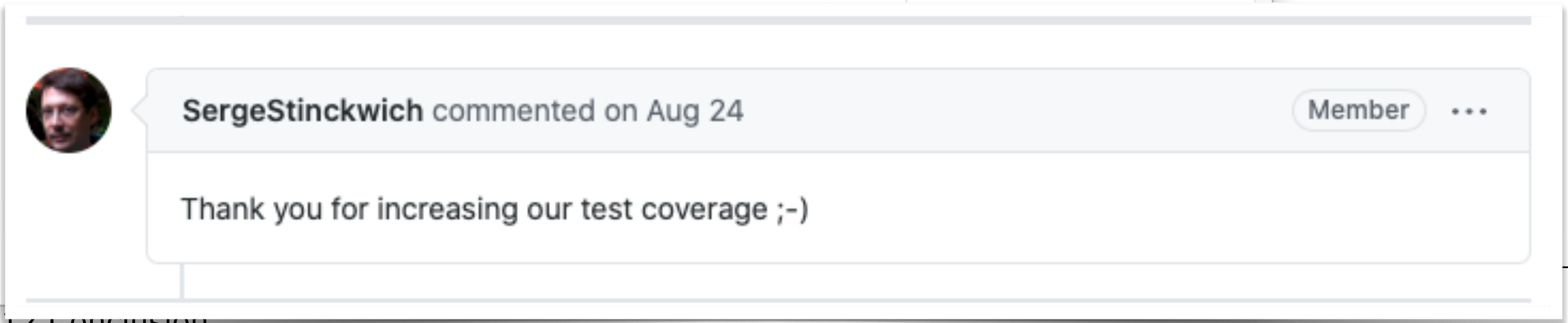
mabdi commented on Aug 21 · Contributor

I submit this pull request to suggest a test method `PMVectorTest >> testHouseholder`.

We noticed that the method #householder is never executed by any of the tests in PMVectorTest. Therefore we created a test method which uses two vectors to exercise both branches in line 203 `(x <=0) ifTrue: [x -u] ifFalse: [0 - s / (x + u)]`. In the former vector `x = -1` forces the ifTrue branch in the latter vector `x = 1.00001` forces the ifFalse branch.

Note that these suggestions are adapted from a test amplification tool called SmallAmp (https://github.com/mabdi/small-amp). SmallAmp executes existing tests, sees which parts of the class under test are not covered and then suggests improvements on the test methods.

Reviewers: hemalvarambhia
Assignees: No one assigned
Labels: None yet
Projects: None yet

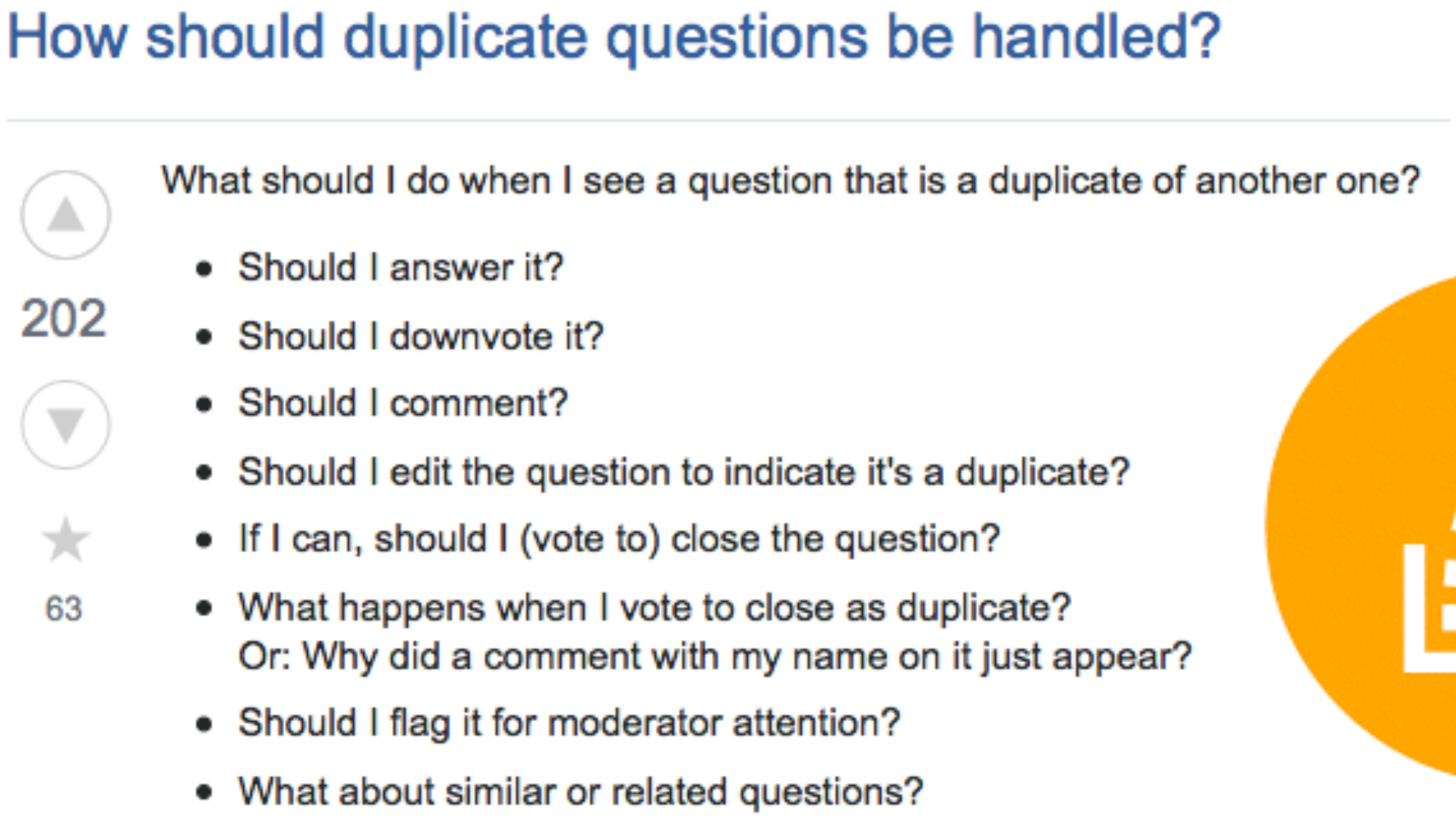


SergeStinckwich commented on Aug 24 · Member

Thank you for increasing our test coverage ;-)

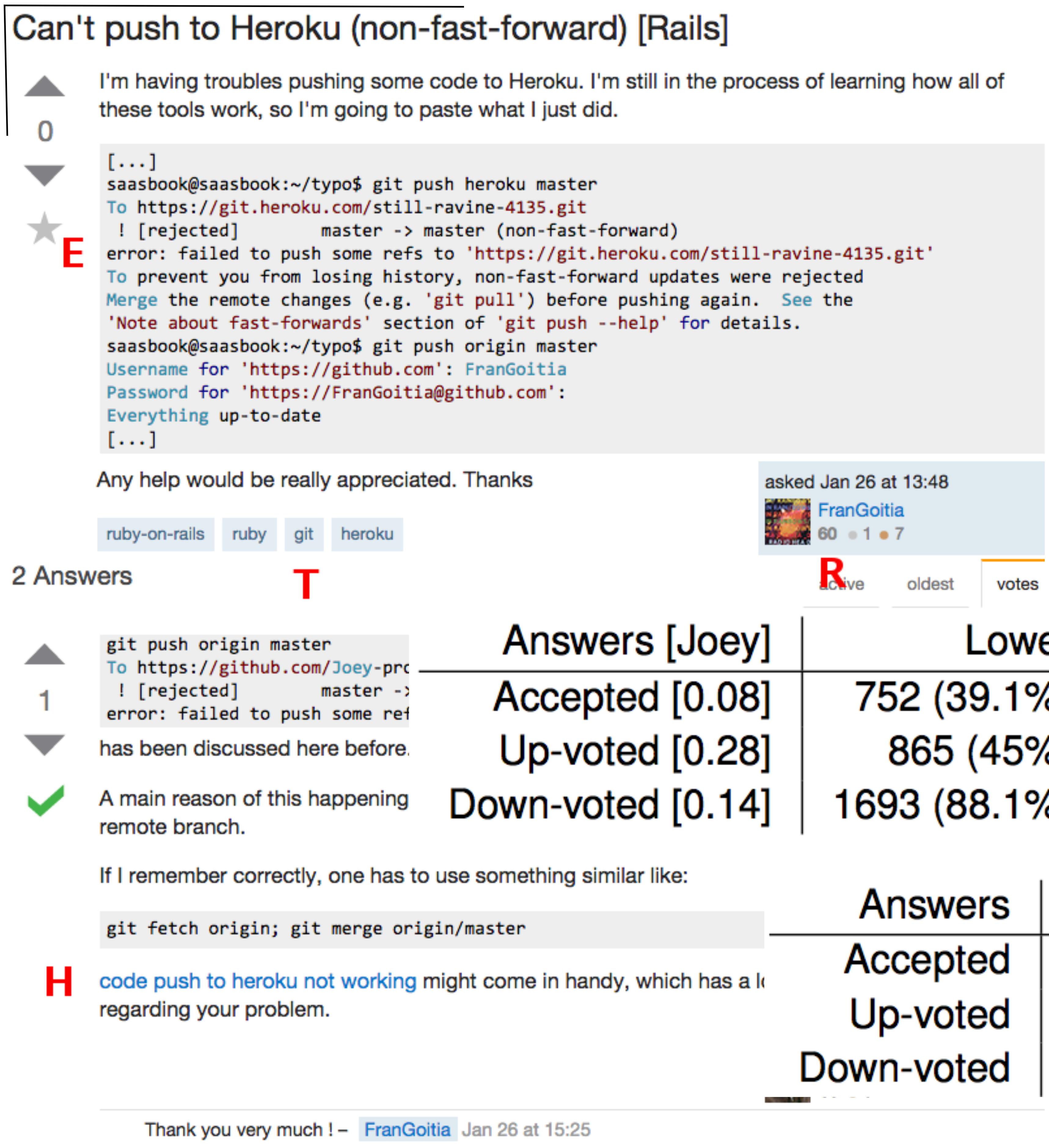
12 pull requests
9 merged
3 pending

Q&A support


How should duplicate questions be handled?

What should I do when I see a question that is a duplicate of another one?

- Should I answer it?
- Should I downvote it?
- Should I comment?
- Should I edit the question to indicate it's a duplicate?
- If I can, should I (vote to) close the question?
- What happens when I vote to close as duplicate?
  Or: Why did a comment with my name on it just appear?
- Should I flag it for moderator attention?
- What about similar or related questions?



Can't push to Heroku (non-fast-forward) [Rails]

0

I'm having troubles pushing some code to Heroku. I'm still in the process of learning how all of these tools work, so I'm going to paste what I just did.

```
[...]
saasbook@saasbook:~/typo$ git push heroku master
To https://git.heroku.com/still-ravine-4135.git
 ! [rejected]        master -> master (non-fast-forward)
error: failed to push some refs to 'https://git.heroku.com/still-ravine-4135.git'
To prevent you from losing history, non-fast-forward updates were rejected
Merge the remote changes (e.g. 'git pull') before pushing again.  See the
'Note about fast-forwards' section of 'git push --help' for details.
saasbook@saasbook:~/typo$ git push origin master
Username for 'https://github.com': FranGoitia
Password for 'https://FranGoitia@github.com':
Everything up-to-date
[...]
```

E

Any help would be really appreciated. Thanks

ruby-on-rails ruby git heroku

asked Jan 26 at 13:48
FranGoitia
60 ● 1 ● 7

2 Answers

T

R

active oldest votes

1

```
git push origin master
To https://github.com/Joey-pro
 ! [rejected]        master -
error: failed to push some ref
```

has been discussed here before.

A main reason of this happening
remote branch.

If I remember correctly, one has to use something similar like:

```
git fetch origin; git merge origin/master
```

H

code push to heroku not working might come in handy, which has a l
regarding your problem.

Thank you very much ! – FranGoitia Jan 26 at 15:25

Answers [Joey]	Lower	Equal	Higher
Accepted [0.08]	752 (39.1%)	3 (0.2%)	1167 (60.7%)
Up-voted [0.28]	865 (45%)	3 (0.2%)	1054 (54.8%)
Down-voted [0.14]	1693 (88.1%)	0 (0%)	229 (11.9%)

Answers	Joey	Answer_Bot
Accepted	4/50 (0.08)	1/13 (0.08)
Up-voted	14/50 (0.28)	0
Down-voted	7/50 (0.14)	3/13 (0.23)

Summary (i)

- You should know the answers to these questions
  + Name 3 items from the code of ethics and provide a one-line explanation.
  + If you are an independent consultant, how can you ensure that you will not have to act against the code of ethics?
  + What would be a possible metric for measuring the amount of innovation of a manufacturing company?
  + Explain the 2 main steps of test amplification: input amplification and assertion amplification

Summary (i) - Continued

“No Silver Bullet”

- What’s the distinction between essential and accidental complexity?
- Name 3 reasons why the building of software is essentially a hard task? Provide a one-line explanation.
- Why is “object-oriented programming” no silver bullet?
- Why is “program verification” no silver bullet?
- Why are “components” a potential silver bullet?

“Killer Robot”

- Which regression tests would you have written to prevent the “killer robot”?
- Was code reviewing applied as part of the QA process? Why (not)?
- Why was the waterfall process disastrous in this particular case?
- Why was the user-interface design flawed?

Summary (ii)

- Can you answer the following questions?
  + You are an experienced designer and you heard that the sales people earn more money than you do. You want to ask your boss for a salary-increase; how would you argue your case?
  + Software products are usually released with a disclaimer like “Company X is not responsible for errors resulting from the use of this program”. Does this mean that you shouldn’t test your software? Motivate your answer.
  + Your are a QA manager and are requested to produce a monthly report about the quality of the test process. How would you do that?
  + Why is “explainable Artificial Intelligence” so important when creating bots for software engineering tasks?
- When you chose the “No Silver Bullet” paper
  + Explain why incremental development is a promising attack on conceptual essence. Give examples from the different topics addressed in the course.
  + “Software components” are said to be a promising attack on conceptual essence. Which techniques in the course are applicable? Which techniques aren’t?
- When you chose the “Killer Robot” paper
  + Recount the story of the Killer Robot case. List the three most important causes for the failure and argue why you think these are the most important.

Summary (i)

- You should know the answers to these questions
  - Name 3 items from the code of ethics and provide a one-line explanation.
  - If you are an independent consultant, how can you ensure that you will not have to act against the code of ethics?
  - What would be a possible metric for measuring the amount of innovation of a manufacturing company?
  - Explain the 2 main steps of test amplification: input amplification and assertion amplification

When you chose the “No Silver Bullet” paper

- What’s the distinction between essential and accidental complexity?
- Name 3 reasons why the building of software is essentially a hard task? Provide a one-line explanation.
- Why is “object-oriented programming” no silver bullet?
- Why is “program verification” no silver bullet?
- Why are “components” a potential silver bullet?

When you chose the “Killer Robot” paper

- Which regression tests would you have written to prevent the “killer robot”?
- Was code reviewing applied as part of the QA process? Why (not)?
- Why was the waterfall process disastrous in this particular case?
- Why was the user-interface design flawed?

Summary (ii)

- Can you answer the following questions?
  + You are an experienced designer and you heard that the sales people earn more money than you do. You want to ask your boss for a salary-increase; how would you argue your case?
  + Software products are usually released with a disclaimer like “Company X is not responsible for errors resulting from the use of this program”. Does this mean that you shouldn’t test your software? Motivate your answer.
  + Your are a QA manager and are requested to produce a monthly report about the quality of the test process. How would you do that?
  + Why is “explainable Artificial Intelligence” so important when creating bots for software engineering tasks?

When you chose the “No Silver Bullet” paper

  + Explain why incremental development is a promising attack on conceptual essence. Give examples from the different topics addressed in the course.
  + “Software components” are said to be a promising attack on conceptual essence. Which techniques in the course are applicable? Which techniques aren’t?

When you chose the “Killer Robot” paper

  + Recount the story of the Killer Robot case. List the three most important causes for the failure and argue why you think these are the most important.