

# Foundations of Software Engineering

Quality-Assurance Process

Christian Kästner

# Foundations of Software Engineering

How to get developers to  
[write tests | use static analysis | appreciate testers]

Christian Kästner



# Agenda

- QA in the context of process
- Case study: QA at Microsoft from 1980 to today
- Case study: Adopting a static analysis tool at Ebay
- Embedding QA in a process
- Social aspects of QA

# Learning Goals

- Understand process aspects of QA
- Describe the tradeoffs of QA techniques
- Select an appropriate QA technique for a given project and quality attribute
- Decide the when and how much of QA
- Overview of concepts how to enforce QA techniques in a process
- Select when and how to integrate tools and policies into the process: daily builds, continuous integration, test automation, static analysis, issue tracking, ...
- Understand human and social challenges of adopting QA techniques
- Understand how process and tool improvement can solve the dilemma between features and quality

# QA Process

# QA Process Considerations

- We covered several QA techniques:
  - Formal verification (15-112)
  - Unit testing, Test driven development (15-214)
  - Various forms of advanced testing for quality attributes (GUI testing, fuzz testing, ...)
  - Static analysis
  - Dynamic analysis
  - Formal inspections and other forms of code reviews
- But: When to use? Which techniques? How much? How to introduce? How to establish a quality culture? How to ensure compliance? Social issues? What about external components?

You are here: [Home](#) > [Shop](#)



Big  
75mm 67 grams

6.00 € / P

[Buy](#) [Detail](#)



Small  
60mm 53 grams

4.00 € / P

[Buy](#) [Detail](#)



Viking Norwik

75mm 134g-402g | 134g 7€ | 201g 8€ | 268g 9€ | 335g 10€ | 402g 11€ |

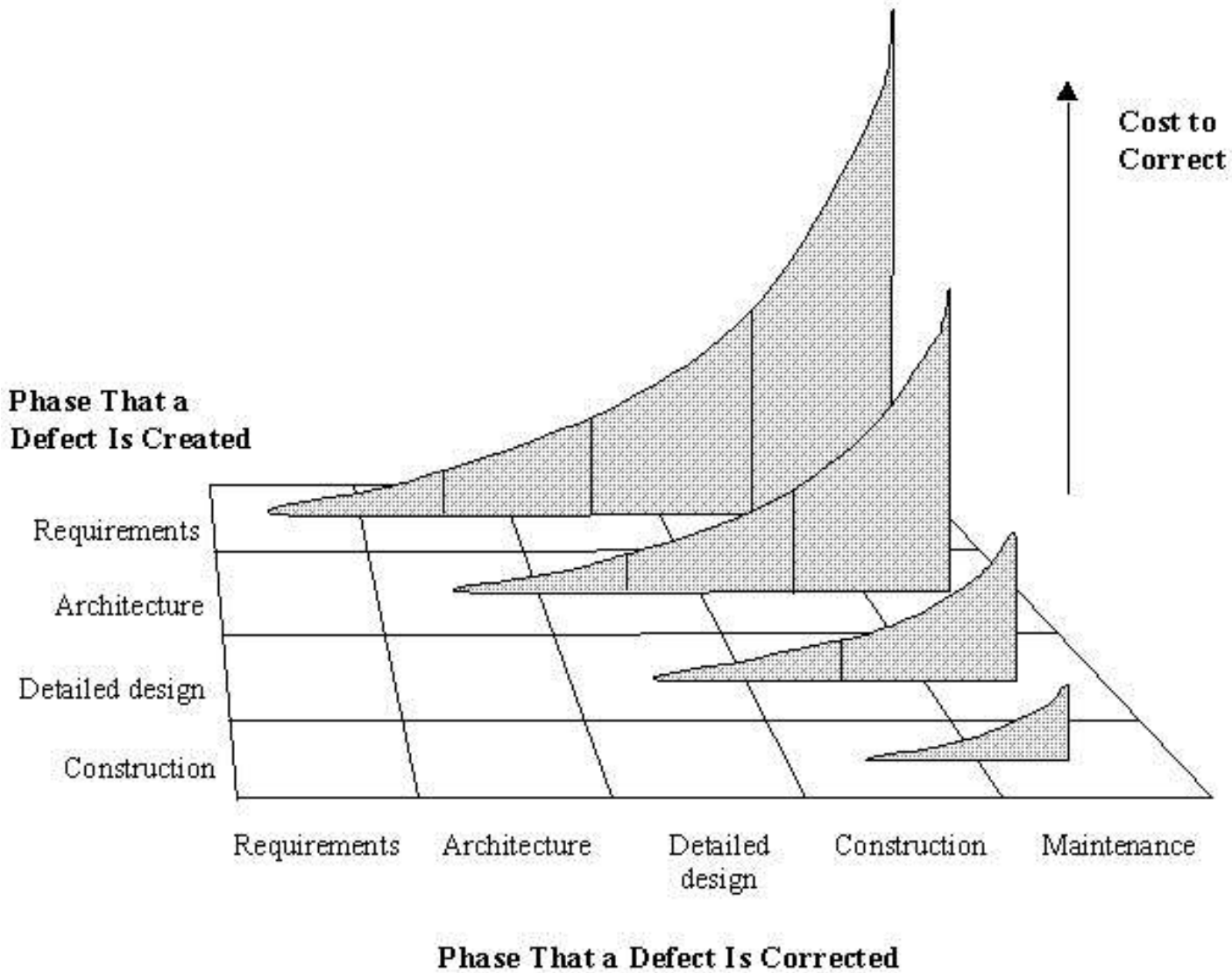
6.00 € / P

[Buy](#) [Detail](#)







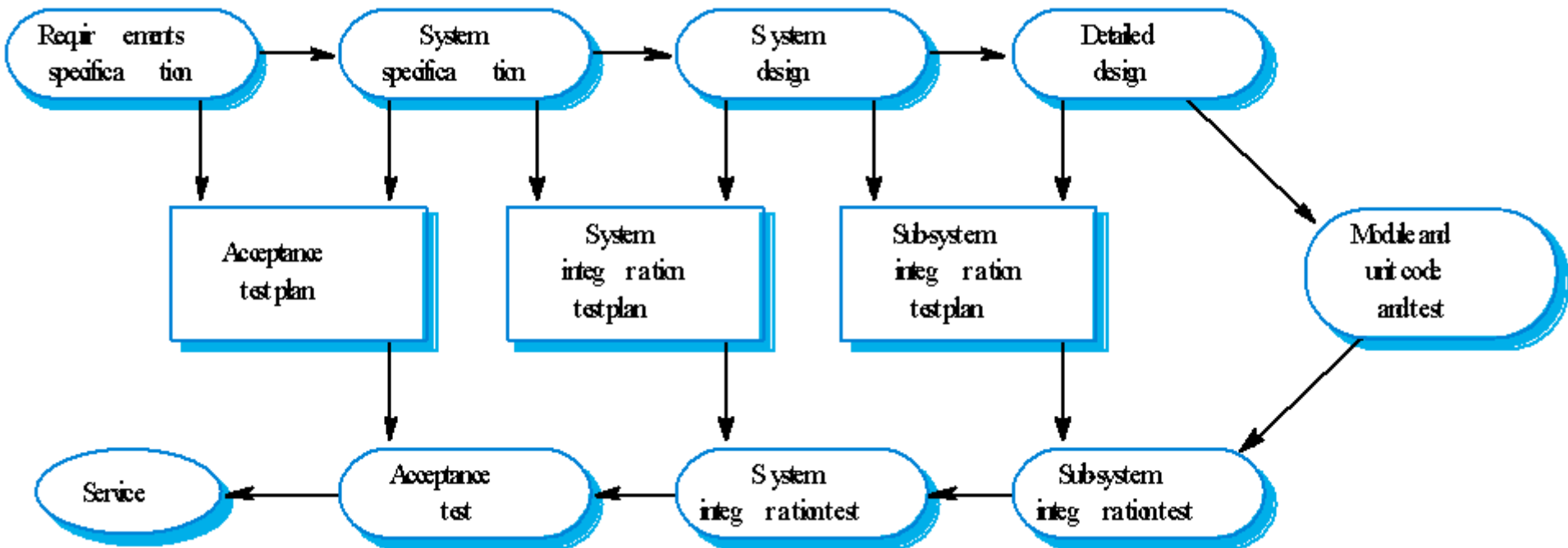


Copyright 1998 Steven C. McConnell. Reprinted with permission from *Software Project Survival Guide* (Microsoft Press, 1998).

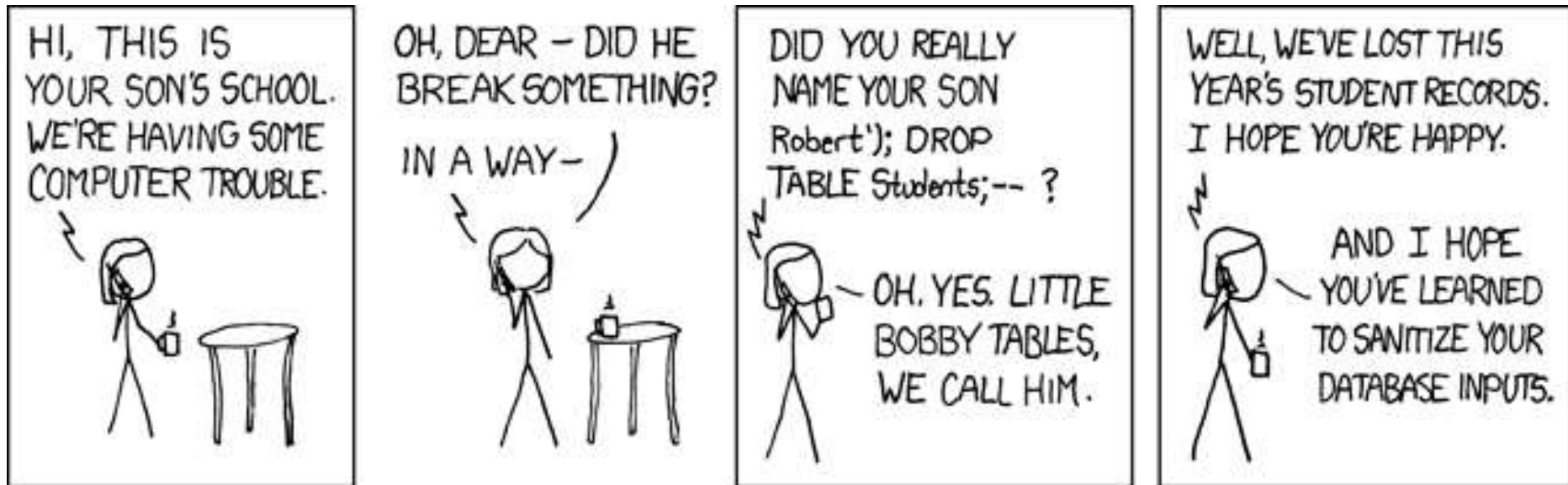
# Qualities and Risks

- What qualities are required?  
(requirements engineering)
- What risks are expected?
  
- Align QA strategy based on qualities and risks

# Example: Test plans linking development and testing



# Example: SQL Injection Attacks



Which QA strategy is suitable?

<http://xkcd.com/327/>

# Example: Scalability



twitter

Home Public Timeline Help

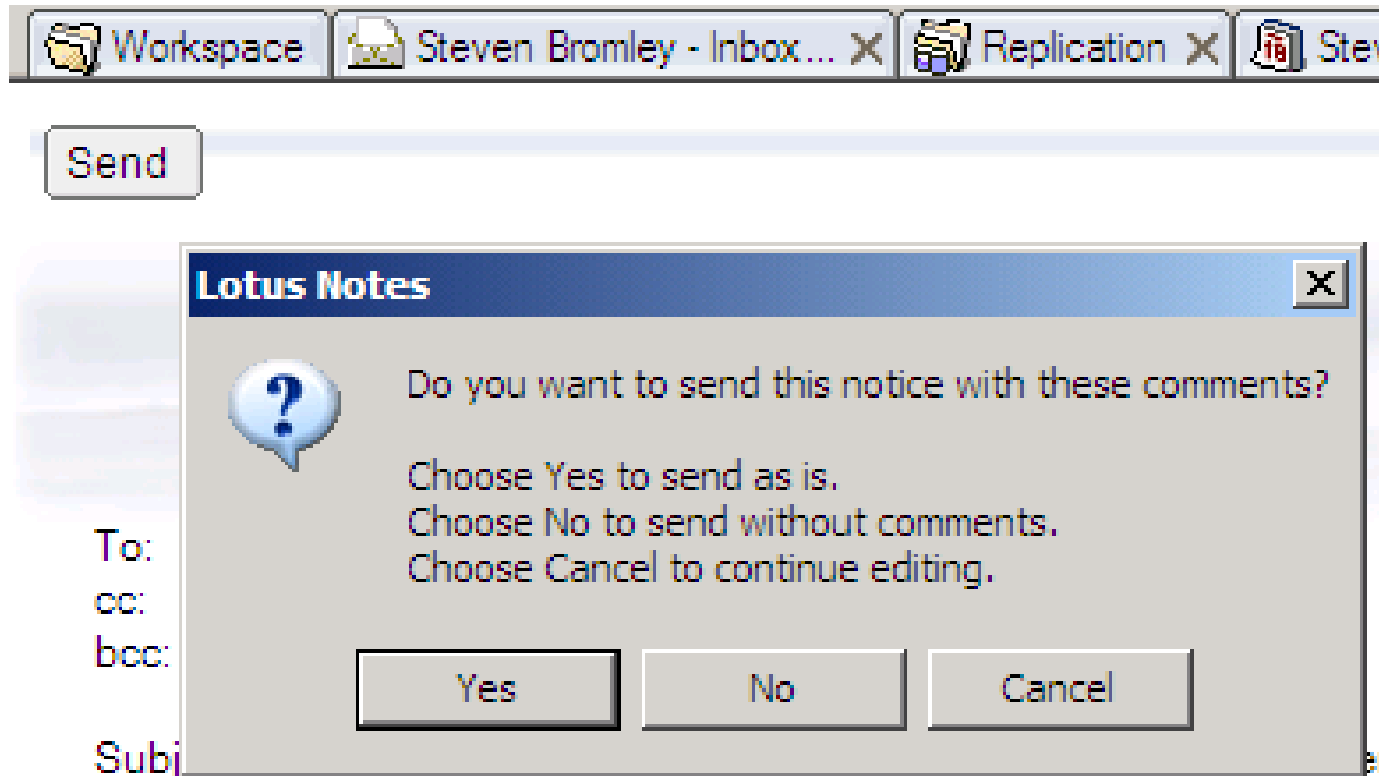
**Twitter is over capacity.**  
Too many tweets! Please wait a moment and try again.

Which QA strategy is suitable?

© 2009 Twitter About Us Contact Blog Status API Help Jobs TOS Privacy

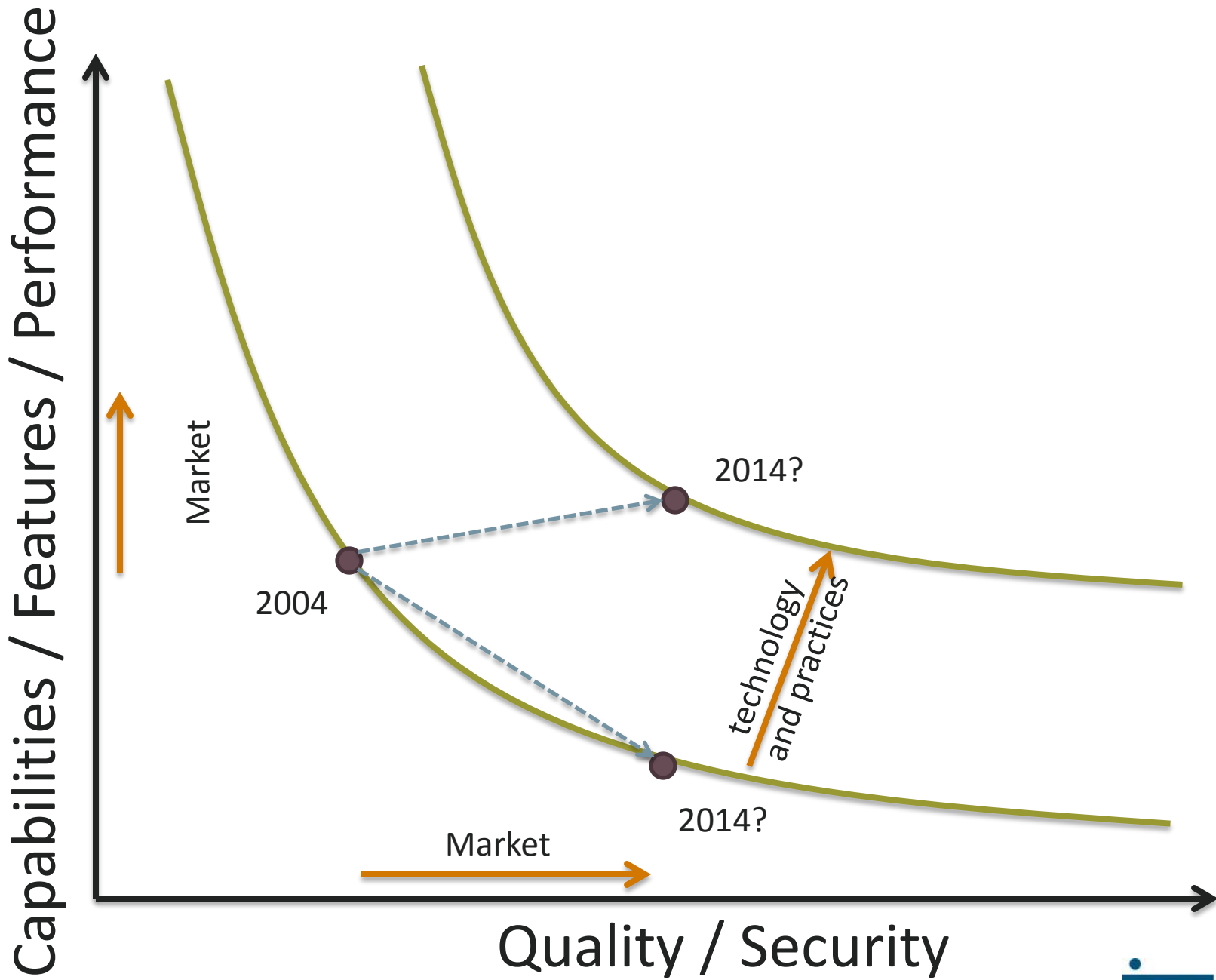
for  
ARE  
:CH

# Example: Usability



Which QA strategy is suitable?





# QA Tradeoffs

- Understand limitations of QA approaches
  - e.g. testing vs static analysis,  
formal verification vs inspection, ...
- Mix and match techniques
- Different techniques for different qualities



# Case Study: QA at Microsoft

A problem has been detected and windows has been shut down to prevent damage to your computer.

THREAD\_NOT\_MUTEX\_OWNER

If this is the first time you've seen this Stop error screen, restart your computer. If this screen appears again, follow these steps:

Check to make sure any new hardware or software is properly installed. If this is a new installation, ask your hardware or software manufacturer for any windows updates you might need.

If problems continue, disable or remove any newly installed hardware or software. Disable BIOS memory options such as caching or shadowing. If you need to use Safe Mode to remove or disable components, restart your computer, press F8 to select Advanced Startup Options, and then select Safe Mode.

Technical information:

\*\*\* STOP: 0x00000011 (0x00234234,0x00005345,0x05345345,0xFFFFFFFF)

NATIONAL BESTSELLER

"A meticulous accounting of how Microsoft operates...A blueprint for any company...  
facing fast-paced markets and hounding competition." —Business Week

# Microsoft SECRETS

How the World's Most Powerful Software Company

Creates Technology,

Shapes Markets,

and Manages People

Michael A. Cusumano

Richard W. Selby

WITH A NEW PREFACE BY THE AUTHORS

Throughout the case studies,  
look for nontechnical challenges  
and how they were addressed  
(social issues, process issues, ...)

# Microsoft's Culture

- Hiring the best developers
  - “Microsoft can achieve with a few hundred top-notch developers for what IBM would need thousands”
- Giving them freedom
- Teams for products largely independent
- Relatively short development cycles
  - Version updates (eg. Excel 3->4) 1-2 month
  - New products 1-4 years
  - Driven by release date
- Little upfront specification, flexible for change and cutting features



# Early Days (1984): Separate testing from development

- after complaints over bugs from hardware manufacturers (eg. wrong computations in BASIC)
- customers complained about products
- IBM insisted that Microsoft improves process for development and quality control
- Serious data-destroying bug forced Microsoft to ship update of Multiplan to 20000 users at 10\$ cost each
- Resistance from developers and some management (incl. Balmer):  
“developers could test their own products, assisted on occasion by high school students, secretaries, and some outside contractors”
- Hired outside testers
- Avoided bureaucracy of formal inspections, signoff between stages, or time logging
- Separate testing group; automated tests; code reviews for new people and critical components

# Early Days (1986): Testing groups

- “Developers got lazy”, relied on test team for QA
- “Infinite defects” - Testers find defects faster than developers can fix them
- Late and large integrations (“big bang”) - long testing periods, delayed releases
- Mac Word 3 desaster: 8 month late, hundreds of bugs, including crashing and data destroying bugs; 1M\$ for free upgrades
- Pressure on delivering quality grew

# 1989 Retreat and “Zero defects”

- see memo

# Zero-Defect Rules for Excel 4

- All changes must compile and link
- All changes must pass the automated quick tests on Mac and Windows
- Any developer who has more than 10 open bugs assigned must fix them before moving to new features

# Testing Buddies

- Development and test teams separate, roughly similar size
- Developers test their own code, run automated tests daily
- Individual testers often assigned to one developer
  - Testing their private releases (branch), giving direct, rapid feedback by email before code is merged

# Testers

- Encouraged to communicate with support team and customers, review media evaluations
- Develop testing strategy for high-risk areas
- Many forms of testing (internally called): unstructured testing, ad hoc testing, gorilla testing, free-form Fridays

# Early-mid 90s

- Zero defect goal (1989 memo)
- Milestones (first with Publisher 1.0 in 1988)
- Version control, branches, frequent integration
- Daily builds
- Automated tests (“quick autotest”) - must succeed before checkin
- Usability labs
- Beta testing (400000 beta testers for Win 95) with instrumentation
- Brief formal design reviews; selected code reviews
- Defect tracking and metrics
- Developers stay in product group for more than one release cycle

# Metrics

- Number of open bugs by severity
  - Number of open bugs expected to decrease before milestone
  - All know severe bugs need to be fixed before release
  - Severity 1 (product crash), Severity 2 (feature crash), Severity 3 (bug with workaround), Severity 4 (cosmetic/minor)
  - Metrics tracked across releases and projects
- Performance metrics
- Bug data used for deciding when “ready to ship”
  - Relative and pragmatic, not absolute view
  - “The market will forgive us for being late, but they won't forgive us for being buggy”



# Challenges of Microsoft's Culture

- Little communication among product teams
- Developers and testers often “not so well read in with software-engineering literature, reinventing the wheel”
  - Long underestimated architecture, design, sharing of components, quality metrics, ...
- Developers resistant to change and “bureaucracy”

# Project Postmortem

- Identify systematic problems and good practices (10-150 page report)
  - document recurring problems and practices that work well
  - e.g.,
    - breadth-first → depth-first & tested milestones
    - insufficient specification
    - not reviewing commits
    - using asserts to communicate assumptions
    - lack of adequate tools → automated tests
    - instrumented versions for testers and beta releases
    - zero defect rule not a priority for developers
- Circulate insights as memos, encourage cross-team learning

# Process Audits

- Informal 1-week audits in problematic problems
- Analyzing metrics, interviewing team members
- Recommendations to pick up best practices from other teams
  - daily builds, automated tests, milestones, reviews

# The 2002 Trustworthy Computing Memo

<http://news.microsoft.com/2012/01/11/memo-from-bill-gates/>

# Code Reviews

- Own code review tools (passaround style)
- Internal studies on how effective reviews are
- Internal tools to improve code reviews

A problem has been detected and windows has been shut down to prevent damage to your computer.

The problem seems to be caused by the following file: SPCMDCON.SYS

PAGE\_FAULT\_IN\_NONPAGED\_AREA

If this is the first time you've seen this Stop error screen, restart your computer. If this screen appears again, follow these steps:

Check to make sure any new hardware or software is properly installed. If this is a new installation, ask your hardware or software manufacturer for any windows updates you might need.

If problems continue, disable or remove any newly installed hardware or software. Disable BIOS memory options such as caching or shadowing. If you need to use Safe Mode to remove or disable components, restart your computer, press F8 to select Advanced Startup Options, and then select Safe Mode.

Technical information:

\*\*\* STOP: 0x00000050 (0xFD3094C2,0x00000001,0xFBFE7617,0x00000000)

\*\*\* SPCMDCON.SYS - Address FBFE7617 base at FBFE5000, DateStamp 3

# SLAM/SDV (since 2000)



- Goal: Reducing blue screens, often caused by drivers
- Driver verification tool for C
- Model checking technology
- Finds narrow class of protocol violations
  - Use characteristics of drivers (not general C code)
  - Found several bugs in Microsoft's well tested sample drivers
- Fully automated in Microsoft compiler suite
- Available for free
- Enforcement through driver certification program

Ball, Thomas, Vladimir Levin, and Sriram K. Rajamani. "A decade of software model checking with SLAM." *Communications of the ACM* 54.7 (2011): 68-76.

# SLAM

- Compelling business case: eliminated most blue screens
- Based on basic science of model checking: originated in university labs with public funding



# Annotation

- How to motivate developers, especially with millions of lines of unannotated code?
- Microsoft approach:
  - Require annotations at checkin (e.g., Reject code that has a `char*` with no `__ecount()`)
  - Make annotations natural, like what you would put in a comment anyway
    - But now machine checkable
    - Avoid formality with poor match to engineering practices
  - Incrementality
    - Check code  $\leftrightarrow$  design consistency on every compile
    - Rewards programmers for each increment of effort
      - Provide benefit for annotating partial code
      - Can focus on most important parts of the code first
      - Avoid excuse: I'll do it after the deadline
  - Build tools to infer annotations
    - Inference is approximate and so annotations may need to be changed, but saves work overall.
    - Unfortunately not yet available outside Microsoft

# SAGE

- White-box fuzz testing (symbolic-execution-based test generation)
- Especially for security issues in file and protocol parsing routines
  - “found many previously-unknown security vulnerabilities in hundreds of Microsoft applications, including image processors, media players, file decoders and document parsers”
- In-house SMT constraint solver (Z3)
- From research project to large-scale deployment
  - Running at scale on 200 machines

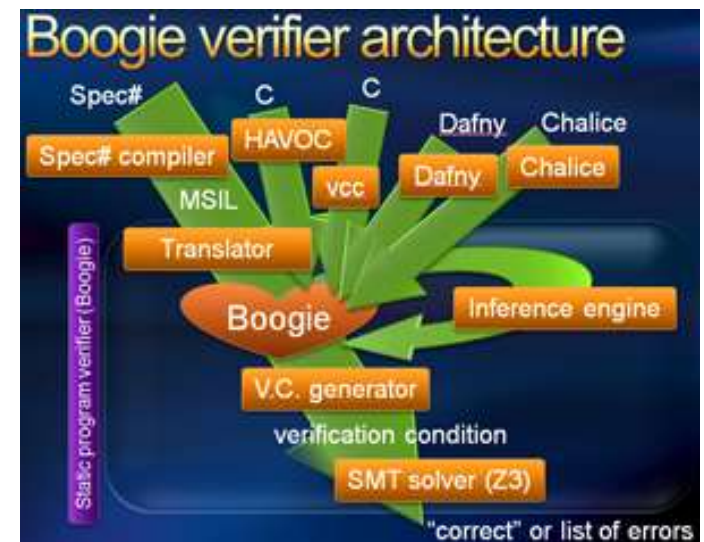
Bounimova, Ella, Patrice Godefroid, and David Molnar. "Billions and billions of constraints: Whitebox fuzz testing in production." In Proceedings of the 2013 International Conference on Software Engineering, pp. 122-131. IEEE Press, 2013.

# Bug prediction

- Metrics
- Mining software repositories
- Example results:
  - Distributed development not critical, but organizational distance is
- Now prioritizing testing effort

# Boogie, Dafny, ...

- Intermediate Verification Language
- “Usable formal verification”
  - Dafny language...
- Active research today...



# Case Study 2:

## Introducing Static Analysis at Ebay

Jaspan, Ciera, I. Chen, and Anoop Sharma. "Understanding the value of program analysis tools." *Companion to the 22nd ACM SIGPLAN conference on Object-oriented programming systems and applications companion*. ACM, 2007.

# Findbugs in 214

- We forced everybody to use Findbugs
- Has it found bugs?
- Who is still using Findbugs?
- Why not?

# Ebay: Prior Evaluations

- Individual teams tried tools
  - On snapshots
  - No tool customization
  - Overall negative results
  - Developers were not impressed: many minor issues (2 checkers reported half the issues, all irrelevant for Ebay)
- Would this change when integrated into process? i.e. incremental checking
- Which bugs to look at?

# Ebay: Goals

- Find defects earlier in the lifecycle
  - Allow quality engineers to focus on different issues
- Find defects that are difficult to find through other QA techniques
  - security, performance, concurrency
- As early as feasible: Run on developer machines and in nightly builds
- No resources to build own tool
  - But few people for dedicated team (customization, policies, creating project-specific analyses etc) possible
- Continuous evaluation



# Ebay: Customization

- Customization dropped false positives from 50% to 10%
- Separate checkers evaluated separately
  - By number of issues
  - By severity as judged by developers; iteratively with several groups
- Some low-priority checkers (e.g., dead store to local) was assigned high priority – performance impact important for Ebay

# Ebay: Enforcement policy

- High priority: All these issues must be fixed (e.g. null pointer exceptions)
  - Potentially very costly given the huge existing code base
- Medium priority: May not be added to the code base. Old issues won't be fixed unless refactored anyway (e.g., high cyclomatic complexity)
- Low priority: At most X issues may be added between releases (usually stylistic)
- Tossed: Turned off entirely

# Ebay: Cost estimation

- Free tool
- 2 developers full time for customization and extension
- A typical tester at ebay finds 10 bugs/week, 10% high priority
- Sample bugs found with Findbugs for a comparison

# Aside: Cost/benefit analysis

- Cost/Benefit tradeoff
  - Benefit: How valuable is the bug?
    - How much does it cost if not found?
    - How expensive to find using testing/inspection?
  - Cost: How much did the analysis cost?
    - Effort spent running analysis, interpreting results – includes false positives
    - Effort spent finding remaining bugs (for unsound analysis)
- Rule of thumb
  - For critical bugs that testing/inspection can't find, a sound analysis is worth it, as long as false positive rate is acceptable.
  - For other bugs, maximize engineer productivity

# Ebay: Combining tools

- Program analysis coverage
  - Performance – High importance
  - Security – High
  - Global quality – High
  - Local quality – medium
  - API/framework compliance – medium
  - Concurrency – low
  - Style and readability – low
- Select appropriate tools and detectors

# Ebay: Enforcement

- Enforcement at dev/QA handoff:
- Developers run FindBugs on desktop
- QA runs FindBugs on receipt of code, posts results, require high-priority fixes.

# Ebay: Continuous evaluation

- Gather data on detected bugs and false positives
- Present to developers, make case for tool

# Incremental introduction

- Begin with early adopters in small team
- Use these as champions in organization
- Support team: answer questions, help with tool.



# Case Study 3: Google's Tricorder

# Integrate Static Analysis in Review Process

- Static analysis as bots in code review tool
  - Automatically applied on each commit
  - Results visible to author and reviewers
- Lightweight checkers, easy to add and modify
- Feedback buttons to indicate ineffective checkers

Sadowski, Caitlin, et al. "Tricorder: Building a program analysis ecosystem."  
2015 IEEE/ACM 37th IEEE International Conference on Software Engineering.

```
package com.google.devtools.staticanalysis;
```

```
public class Test {
```

▼ Lint Missing a Javadoc comment.

Java  
1:02 AM, Aug 21

[Please fix](#)

[Not useful](#)

```
public boolean foo() {  
    return getString() == "foo".toString();  
}
```

▼ ErrorProne String comparison using reference equality instead of value equality  
(see <http://code.google.com/p/error-prone/wiki/StringEquality>)

StringEquality  
1:03 AM, Aug 21

[Please fix](#)

Suggested fix attached: [show](#)

[Not useful](#)

```
}  
  
public String getString() {  
    return new String("foo");  
}  
}
```

```
package com.google.devtools.staticanalysis;
```

```
public class Test {
```

▼ Lint Missing a Javadoc comment.

Java  
1:02 AM, Aug 21

[Please fix](#)

[Not useful](#)

```
public boolean foo() {  
    return getString() == "foo".toString();  
}
```

▼ ErrorProne String comparison using reference equality instead of value equality  
(see <http://code.google.com/p/error-prone/wiki/StringEquality>)

StringEquality  
1:03 AM, Aug 21

[Please fix](#)

//depot/google3/java/com/google/devtools/staticanalysis/Test.java

```
package com.google.devtools.staticanalysis;
```

```
public class Test {  
    public boolean foo() {  
        return getString() == "foo".toString();  
    }  
}
```

```
    public String getString() {  
        return new String("foo");  
    }  
}
```

```
package com.google.devtools.staticanalysis;
```

```
import java.util.Objects;
```

```
public class Test {  
    public boolean foo() {  
        return Objects.equals(getString(), "foo".toString());  
    }  
}
```

```
    public String getString() {  
        return new String("foo");  
    }  
}
```

Apply

Cancel

# QA within the Process

# QA as part of the process

- Have QA deliverables at milestones (management policy)
  - Inspection / test report before milestone
- Change development practices (req. developer buy-in)
  - e.g., continuous integration, pair programming, reviewed checkins, zero-bug static analysis before checking
- Static analysis part of code review (Google)
- Track bugs and other quality metrics

# Defect tracking

- Issues: Bug, feature request, query
- Basis for measurement
  - reported in which phase
  - duration to repair, difficulty
  - categorization
    - > root cause analysis
- Facilitates communication
  - questions back to reporter
  - ensures reports are not forgotten
- Accountability

Bug List: (48 of 200) [First](#) [Last](#) [Prev](#) [Next](#) [Show last search results](#) [Search page](#) [Enter new bug](#)

**[Eclipse]**  
Bug#: 160502

Product: Platform  
Component: Runtime  
Status: REOPENED  
Resolution: platform-runtime-inbox  
Assigned To: <platform-runtime-inbox@eclipse.org>

Hardware: PC  
OS: Linux  
Version: 3.2.1  
Priority: P3  
Severity: blocker  
Target Milestone:

Reporter: [Clare Carty](#)  
<ccarty@ca.ibm.com>  
Add CC:  
CC: [ccarty@ca.ibm.com](#)  
[john\\_erhorne@ca.ibm.com](#)  
 Remove selected CC

QA Contact:   
URL:   
Summary: JVM crash at random intervals on SUSE 9 with Sun JRE 1.5  
Status:   
Whiteboard:   
Keywords: jvm

Attachment	Type	Created	Size	Actions
<a href="#">screenshot of crash</a>	image/jpeg	2006-10-11 12:14	131.55 KB	<a href="#">Edit</a>
<a href="#">Create a New Attachment</a> (proposed patch, testcase, etc.)				<a href="#">View All</a>

Bug 160502 depends on:  [Show dependency tree](#)  
Bug 160502 blocks:

Votes: 0 [Show votes for this bug](#) [Vote for this bug](#)

# Enforcement

- Microsoft: check in gates
  - Cannot check in code unless analysis suite has been run and produced **no errors** (test coverage, dependency violation, insufficient/bad design intent, integer overflow, allocation arithmetic, buffer overruns, memory errors, security issues)
- eBay: dev/QA handoff
  - Developers run FindBugs on desktop
  - QA runs FindBugs on receipt of code, posts results, require high-priority fixes.
- Google: static analysis on commits, shown in review
- Requirements for success
  - Low false positives
  - A way to override false positive warnings (typically through inspection).
  - Developers must buy into static analysis first



# Reminder: Continuous Integration

Jenkins  admin | log out

Jenkins ENABLE AUTO REFRESH [add description](#)

- [New Job](#)
- [People](#)
- [Build History](#)
- [Project Relationship](#)
- [Check File Fingerprint](#)
- [Manage Jenkins](#)
- [My Views](#)
- [Disk usage](#)

**Build Queue**

No builds in the queue.

**Build Executor Status**

#	Status
1	Idle

All +	S	W	Name	Last Success	Last Failure	Last Duration	
			<a href="#">FOSPL</a>	1 hr 40 min (#186)	6 days 8 hr (#164)	47 sec	
			<a href="#">IVM</a>	2 days 19 hr (#288)	12 days (#279)	4 min 35 sec	
			<a href="#">IVMBranch</a>	3 mo 19 days (#139)	3 mo 25 days (#125)	4 min 27 sec	
			<a href="#">IVMBranchEval</a>	3 mo 24 days (#70)	3 mo 28 days (#57)	12 min	
			<a href="#">IVMBranchTest</a>	3 mo 24 days (#110)	3 mo 19 days (#118)	11 min	
			<a href="#">IVMTest</a>	2 days 19 hr (#160)	10 days (#155)	12 min	
			<a href="#">TypeChef</a>	21 days (#354)	7 hr 54 min (#357)	16 min	
			<a href="#">variational</a>	1 yr 2 mo (#11)	1 yr 2 mo (#3)	3 min 43 sec	

Icon: [S](#) [M](#) [L](#)

[Legend](#)
[RSS for all](#)
[RSS for failures](#)
[RSS for just latest builds](#)

[Help us localize this page](#)
Page generated: Jan 29, 2013 10:41:11 PM [REST API](#) [Jenkins ver. 1.500](#)

# Automating Test Execution

```
ckaestne@kastner-desktop:~/work/TypeChef/TypeChef$ sbt "project FeatureExprLib" test
Detected sbt version 0.12.2
[info] Loading global plugins from /usr0/home/ckaestne/.sbt/plugins
[info] Loading project definition from /usr0/home/ckaestne/work/TypeChef/TypeChef/project/project
[info] Loading project definition from /usr0/home/ckaestne/work/TypeChef/TypeChef/project
[info] Set current project to TypeChef (in build file:/usr0/home/ckaestne/work/TypeChef/TypeChef/)
[info] Set current project to FeatureExprLib (in build file:/usr0/home/ckaestne/work/TypeChef/TypeChef/)
[info] Compiling 10 Scala sources to /usr0/home/ckaestne/work/TypeChef/TypeChef/FeatureExprLib/target/scala-2.10/test-classes...
[info] + FeatureExpr.parse(print(x))==x: OK, passed 100 tests.
[info] + FeatureExpr.and1: OK, passed 100 tests.
[info] + FeatureExpr.and0: OK, passed 100 tests.
[info] + FeatureExpr.andSelf: OK, passed 100 tests.
[info] + FeatureExpr.or1: OK, passed 100 tests.
[info] + FeatureExpr.or0: OK, passed 100 tests.
[info] + FeatureExpr.orSelf: OK, passed 100 tests.
[info] + FeatureExpr.a eq a: OK, passed 100 tests.
[info] + FeatureExpr.a equals a: OK, passed 100 tests.
[info] + FeatureExpr.a equivalent a: OK, passed 100 tests.
[info] + FeatureExpr.a implies a: OK, passed 100 tests.
[info] + FeatureExpr.creating (a and b) twice creates equal object: OK, passed 100 tests.
[info] + FeatureExpr.creating (a or b) twice creates equal object: OK, passed 100 tests.
[info] + FeatureExpr.creating (not a) twice creates equal object: OK, passed 100 tests.
[info] + FeatureExpr.applying not twice yields an equivalent formula: OK, passed 100 tests.
[info] + FeatureExpr.Commutativity wrt. equivalence: (a and b) produces the same object as (b and a): OK, passed 100 tests.
[info] + FeatureExpr.Commutativity wrt. equivalence: (a or b) produces the same object as (b or a): OK, passed 100 tests.
[info] + FeatureExpr.taut(a=>b) == contr(a and !b): OK, passed 100 tests.
[info] + FeatureExpr.featuremodel.tautology: OK, passed 100 tests.
```

# Continuous Integration with Travis-CI

The screenshot shows the Travis CI interface. On the left, there's a sidebar with a search bar and a list of recent repositories. The main area displays the build details for the rails/rails repository, including the current build status (successful), commit information, and a build matrix table. On the right, there's a sidebar showing the status of various workers and their queues.

**Recent Repositories:**

- diasporg/diaspora #209 (Failed)
- rubinius/rubinius #815 (Success)
- robgleeson/ed #31 (Success)
- niku/frange #4 (Success)
- tedsuo/raaraa #48 (Success)
- holman/play #84 (Failed)
- crnc/sift.js #35 (Success)
- BonzaiProject/Bonzai #19 (Success)

**rails/rails Build Details:**

Build: ● 1995  
Commit: f3e079e (master)  
Finished: about 6 hours ago  
Duration: 1 hr 33 min 32 sec  
Author: Vijay Dev  
Message: Merge pull request #4248 from andrew/2012 Updated copyright notices for 2012

**Build Matrix:**

Job	Duration	Finished	Rvm	Env
<span style="color: green;">●</span> 1995.1	19 min 5 sec	about 6 hours ago	1.9.3	GEM=railties
<span style="color: green;">●</span> 1995.2	12 min 38 sec	about 6 hours ago	1.9.3	GEM=ap,am,amo,ares,as
<span style="color: green;">●</span> 1995.3	16 min 57 sec	about 6 hours ago	1.9.3	GEM=ar:mysql
<span style="color: green;">●</span> 1995.4	12 min 55 sec	about 6 hours ago	1.9.3	GEM=ar:mysql2
<span style="color: green;">●</span> 1995.5	12 min 34 sec	about 6 hours ago	1.9.3	GEM=ar:sqlite3
<span style="color: green;">●</span> 1995.6	19 min 23 sec	about 6 hours ago	1.9.3	GEM=ar:postgresql

**Workers and Queues:**

- Queue: Common: No jobs
- Queue: NodeJs: No jobs
- Queue: Php: No jobs
- Queue: Rails: No jobs
- Queue: Erlang: No jobs
- Queue: Spree: No jobs

# Social Aspects

# Social issues

- Developer attitude toward defects
- Developer education about security
- Using peer pressure to enforce QA practices
  - Breaking the build – various rules

# Social issues

- Developer vs tester culture
- Testers tend to deliver bad news
- Defects in performance evaluations?
- Issues vs defects
- Good test suits raise confidence, encourage shared code ownership

# Reporting Defects

- Reproducible defects
- Simple and general
- One defect per report
- Non-antagonistic
  - (testers usually bring bad news)
  - State the problem
  - Don't blame

Thread: To much is to much

Thread Tools

Search Thread

Display

21-05-2012, 15:05

#1

bauchy •

ined  
aten Date: Apr 2012  
ts: 8**To much is to much**

I am fed up of all the bugs.  
 Production never update i must go through the army to updaye the production  
 disconnection very often  
 loss of gold and forge point  
 loss of life points of soldiers without fighting  
 diasapearing soldiers  
 and at las but not least my copper foundry diasapered while i was trying to change its emplacement.

YOU ARE SORRY FOR ALL THESE INCONVIENIENCE,I KNOW, YOU ARE GOING TO SAY IT IS BECAUSE IT S A BETA,I KNOW  
 BUT THIS GAME SUCKS FROM THE TOP TO THE BOTTOM,PAY 10 MONKEYS AS DEVELLOPERS AND YOU WILL HAVE THE SAME RESULTS.  
 BY THE WAY IF YOU WANT TO TEST A CAR BEFORE BUYING IT YOU DO NOT HAVE TO PAY,HERE WITH THE DIAMONDS OPTION IS YOU WANT TO BUY THE CAR OK, YOU WANT TO TEST IT OK SO YOU MUST PAY.  
 SO PLEASE THIS TIME NO APOLOGISE,I NEED EXPLANATION AND NOT AS BETA BLA BLA BLA.  
 IS INNO CIE ARE BELONGING TO BANKSTERS GANG?

*Last edited by Carasus; 23-05-2012 at 02:13.*

21-05-2012, 15:14

rmerlynch •

adier-General



you DO NOT have to buy diamonds. its your choice, and you should r  
 diamonds are paying for this game to be developed. if your so upset  
 bottom then don't let the door hit you in the a\$\$ on the way out ☺

**FORGE OF EMPIRES****NOT REGISTERED YET?**

ON THE FORUM YOU CAN  
 BE A PART OF THE  
 COMMUNITY AND  
 PARTICIPATE IN TOPICS  
 REGARDING THE GAME.  
 YOU CAN ALSO FIND NEW  
 GUILD MEMBERS AND  
 DISCUSS THE WORLDS  
 DEVELOPMENT.

**SO DONT FORGET TO  
 CREATE AN ACCOUNT!**



#2



- *To:* [debian-devel@lists.debian.org](mailto:debian-devel@lists.debian.org)
  - *Subject:* Reporting 1.2K crashes
  - *From:* Alexandre Rebert <[alexandre.rebert@gmail.com](mailto:alexandre.rebert@gmail.com)>
  - *Date:* Tue, 25 Jun 2013 01:28:10 -0400
  - *Message-id:* <[CAF1AS2itHonB5KtnqNnX5zat4Bh7ytr0dG2tzX6BSKzboVSMzA@mail.gmail.com](mailto:CAF1AS2itHonB5KtnqNnX5zat4Bh7ytr0dG2tzX6BSKzboVSMzA@mail.gmail.com)>
- 

Hi,

I am a security researcher at Carnegie Mellon University, and my team has found thousands of crashes in binaries downloaded from debian wheezy packages. After contacting [owner@bugs.debian.org](mailto:owner@bugs.debian.org), Don Armstrong advised us to contact you before submitting ~1.2K bug reports to the Debian BTS using [maintonly@bugs.debian.org](mailto:maintonly@bugs.debian.org) (to avoid spamming `debian-bugs-dist`).

We found the bugs using Mayhem [1], an automatic bug finding system that we've been developing in David Brumley's research lab for a couple of years. We recently ran Mayhem on almost all ELF binaries of Debian Wheezy (~23K binaries) [2], and it reported thousands of crashes.

Our goal here is to make our bug reports as complete and accurate as possible. To minimize duplicates, we are reporting only one crash per binary, and at most 5 crashes per package. This amounts to ~1.2K crashes. Moreover, to ensure accuracy, we confirmed all the crashes by re-running them in a fresh unstable installation. Finally, we also filter out assertion failures for now, as they seemed less important. In short, every report is reproducible and actionable.

You can download the list of affected packages, with their maintainers [3], generated with `dd-list`, as well as a sample bug report for `gcov-4.6` [4]. The bug report contains:

- 1) the bug report that will be mailed to [maintonly@bugs.debian.org](mailto:maintonly@bugs.debian.org) (`report.txt`)
- 2) a testcase reproducing the crash in `./crash/`
- 3) information about the crash in `./crash_info/`: a core dump (`core`), the output of the crash (`crash_output.txt`), the `dmesg` of the crash (`dmesg.txt`), as well as the exit status (`exit_status.txt`).

This is a lot of bugs, and we want to make sure we're doing bug reports right, so that we don't make anyone angry by spamming the BTS with bad reports. Please let us know if the reports are good enough to proceed with the filing, or if any additional information should be

# Summary

- Developing a QA plan:
  - Identify quality goals and risks
  - Mix and match approaches
  - Enforce QA, establish practices
- Case study from Microsoft
- Integrate QA in process
- Social issues in QA

# Further Reading

- Cusumano, Michael A., and Richard W. Selby. "Microsoft secrets." (1997).
  - Book covers quality assurance at Microsoft until the mid 90s (and much more)
- Ball, Thomas, Vladimir Levin, and Sriram K. Rajamani. "A decade of software model checking with SLAM." *Communications of the ACM* 54.7 (2011): 68-76.
  - An overview of SLAM at Microsoft
- Jaspan, Ciera, I. Chen, and Anoop Sharma. "Understanding the value of program analysis tools." *Companion OOPSLA*. ACM, 2007.
  - Description of eBay evaluating FindBugs
- Sadowski, C., van Gogh, J., Jaspan, C., Söderberg, E., & Winter, C. Tricorder: Building a Program Analysis Ecosystem. ICSE 2015
  - Integrating static analysis into code reviews at Google in a data-driven way
- Sommerville. *Software Engineering*. 8<sup>th</sup> Edition. Chapter 27
  - QA planning and process improvement, standards