Empirical Software Engineering at Microsoft Research

Christian Bird
Brendan Murphy
Nachiappan Nagappan
Thomas Zimmermann
Empirical Software Engineering Goals

• Gain insight from product, process, people, and customers
  – Gather quantitative & qualitative data
  – Empirical studies

• Empower software development teams
  – Build tools
  – Provide best practices
What makes empirical software engineering research at Microsoft unique?

- Easy access to industrial data
- Easy access to engineers
- Near term impact
- Collaborations with Microsoft and external researchers
Collaborations of the ESE Group

Main locations of the ESM Group:
Redmond, Cambridge (UK)

Collaborations with other MSR Labs:
Microsoft Research India (Bangalore),
Microsoft Research Asia (Beijing),
European Microsoft Innovation Center (Aachen, Germany)

Interns 2007-2010: U of Virginia (Ray Buse); U of California, Santa Cruz (Ken Hullett); National Institute of Technology, Tiruchirappalli, India (Kalaikumaran Ramamurthy); Stanford U (Philip Guo); Boğaziçi U, Turkey (Ayse Tosun); Darmstadt U of Technology (Andreas Johansson); North Carolina State U (Lucas Layman, Mei Nagappan)

Visitors 2007-2010: Hong Kong U of Science and Technology (Sung Kim); U of Zurich (Harald Gall, Martin Pinzger); Saarland U (Andreas Zeller); North Carolina State U (Laurie Williams); U of Maryland (Victor Basili); Darmstadt U of Technology (Neeraj Suri).
The ESE Group at Microsoft

Socio-technical congruence
- Contribution behavior, technical relationships, and quality
- Does organizational structure affect bugs?
- Investigating the effects of geographic distribution

Bug reporting and triage
- What makes a good bug report?
- Characterizing bug reassignments and fixed bugs
- Collaboration and information needs in bug reports

Data-driven software engineering
- Failure-prediction/risk analysis
- Assessment of test-driven development and unit testing
- Analytics for software development
Outline

Socio-technical congruence

• Contribution behavior, technical relationships, and quality
• Does organizational structure affect bugs?
• Investigating the effects of geographic distribution

Bug reporting and triage

• What makes a good bug report?
• Characterizing bug reassignments and fixed bugs
• Collaboration and information needs in bug reports

Data-driven software engineering

• Failure-prediction/risk analysis
• Assessment of test-driven development and unit testing
• Analytics for software development
“Design and programming are human activities; forget that and all is lost.”

- Bjarne Stroustrup
DOES DISTRIBUTED DEVELOPMENT AFFECT QUALITY?
Distributed development

• Global development is becoming more common.
• There are many challenges, lots of research
  – Multiple workshops, now a conference (ICGSE)
  – Prior research enumerates many of the hurdles and effects on time to market, code size, process difficulty
• What is the effect on code quality?

This study focuses on the effect of distributed development on software quality in Windows Vista
Examining binaries

Level of distribution is the smallest entity that 75% of the commits come from

© Microsoft Corporation
Examining binaries

Level of distribution is the smallest entity that 75% of the commits come from
Effect on quality

Use linear regression to examine effect on post-release failures

**Model:** Levels of distribution

<table>
<thead>
<tr>
<th>Name</th>
<th>Diff Buildings</th>
<th>Diff Cafeterias</th>
<th>Diff Campuses</th>
<th>Diff Localities</th>
<th>Diff Continents</th>
<th>Number of Devs</th>
<th>Failures</th>
</tr>
</thead>
<tbody>
<tr>
<td>foo.dll</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>bar.exe</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>baz.sys</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>hello.exe</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>23</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>% increase</th>
<th>Std Err.</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Constant)</td>
<td></td>
<td>0.09</td>
<td>p &lt; 0.001</td>
</tr>
<tr>
<td>Diff Buildings</td>
<td>2.6%</td>
<td>0.42</td>
<td>p = 0.493</td>
</tr>
<tr>
<td>Diff Cafeterias</td>
<td>3.9%</td>
<td>0.18</td>
<td>p = 0.016</td>
</tr>
<tr>
<td>Diff Campuses</td>
<td>6.3%</td>
<td>0.29</td>
<td>p = 0.019</td>
</tr>
<tr>
<td>Diff Localities</td>
<td>8.3%</td>
<td>1.23</td>
<td>p = 0.457</td>
</tr>
<tr>
<td>Diff Continents</td>
<td>-3.9%</td>
<td>0.26</td>
<td>p = 0.101</td>
</tr>
<tr>
<td>Num Devs</td>
<td></td>
<td>0.00</td>
<td>p &lt; 0.001</td>
</tr>
</tbody>
</table>
Why did distributed development work?

- Not outsourcing
- Liaisons and face to face meetings
- Senior engineers started in Redmond
- **Daily synchronous communication**
- Consistent process and tools
- End to end ownership
- Common schedules
- Organizational integration
INFLUENCE OF ORGANIZATIONAL STRUCTURE ON SOFTWARE QUALITY

Nachiappan Nagappan, Brendan Murphy, Victor Basili @ ICSE 2008
Motivation

• **Conway’s Law:**
  – “Organizations that design systems are constrained to produce **systems which are copies of the communication structures** of these organizations.”

• **Brooks** argues in the Mythical Man Month:
  – product quality is strongly affected by that structure.

• Little empirical evidence for relationship between organizational structure and software quality
Organizational Metrics

- # of Engineers who edited
- # of Engineers who left
- Org level with 75% ownership
- # Engineers in owning org
- # of Organizations making edits
- % of edits made by owning org
- # of edits
## Defect Prediction

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-Release Bugs</td>
<td>73.80%</td>
<td>62.90%</td>
</tr>
<tr>
<td>Org. Structure</td>
<td>86.20%</td>
<td>84.00%</td>
</tr>
</tbody>
</table>

Show stronger relationship with software quality than traditional measures.
SOCIAL NETWORK BASED DEFECT PREDICTION
Quality assurance is limited...

...by time...

...and by money.
Defect prediction

Software element → Model
  PCA
  Regression
  Bayes

Prediction →
  Classification
  Ranking

Metrics
Churn
Dependencies
...
Are these relationships indicative of failures?
Social Network Analysis: Centrality

- **Degree** - number of edges incident to a node
- **Closeness** - distance to all other nodes
- **Betweenness** - number of shortest paths flow through a binary
- **Eigenvector** – Similar to PageRank algorithm
Software Networks

Contribution Network  Socio-Technical Network  Dependency Network
## Vista Model Results

<table>
<thead>
<tr>
<th>Network</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Score</th>
<th>Nagel.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dependency</td>
<td>0.707</td>
<td>0.547</td>
<td>0.617</td>
<td>0.284</td>
</tr>
<tr>
<td>Contribution</td>
<td>0.774</td>
<td>0.650</td>
<td>0.706</td>
<td>0.506</td>
</tr>
<tr>
<td>Combined</td>
<td>0.787</td>
<td>0.681</td>
<td>0.730</td>
<td>0.504</td>
</tr>
<tr>
<td>Socio-technical</td>
<td>0.769</td>
<td>0.705</td>
<td>0.736</td>
<td>0.520</td>
</tr>
</tbody>
</table>

*Best results when using social & technical relationships*
Replicated Study on Eclipse

Network based prediction models work on data sets other than Windows Vista

- Precision: 75% - 82%
- Recall: 78% - 86%
The ESE Group at Microsoft

Socio-technical congruence
- Contribution behavior, technical relationships, and quality
- Does organizational structure affect bugs?
- Investigating the effects of geographic distribution

Bug reporting and triage
- What makes a good bug report?
- Characterizing bug reassignments and fixed bugs
- Collaboration and information needs in bug reports

Data-driven software engineering
- Failure-prediction/risk analysis
- Assessment of test-driven development and unit testing
- Analytics for software development
Backyard Safari Bug Vacuum
Price: $19.82

Big Bunch O' Bugs
Price: $8.25

Best Ever Bug Jar
Price: $8.12

Watch a Bug
Price: $7.59
Bugs! Bugs! Bugs!

by Bob Barner
Bug Tracker

- Draw the bug.
- Can it fly?
- Where did you find it?
- How many legs?
- Does it sting?
BUG TRACKING 101
Your software fails...

The application Keynote has unexpectedly quit.
The system and other applications have not been affected.
Would you like to submit a bug report to Apple?
Submit Report... Cancel

This program has performed an illegal operation and will be shut down.
If the problem persists, contact the program vendor.

VS02 caused an invalid page fault in module MRB.DLL at 014f:7f860ed.
Registers:
EAX=00000004 CD-014f EIP=7f860ed EFLG5=00010216
EBX=00000004 ES=0157 ESP=0074f448 EBP=0074f498
ECX=00003a6e DS=0157 ESI=00750000 EDX=131f
EDX=82414568 EB=0157 EDI=82215b94 GS=0000
Byte at CS:EIP:
53 4b 88 83 ec 03 0f 34 c4 01 42 94 a8 01 00 00
Stack dump:
000f4d4 824145d8 7f862214 0074f444 00004c4a
824145d4 0074f444 7f862214 0074f444 0074f494
0074f494 00caca2a 00000000 824145d8 00005c4d
000f4d4

Application "gedit" (process 25321) has crashed due to a fatal error.
(Segmentation fault)

Please visit the GNOME Application Crash page for more information

Close Submit a bug report Debug
Bug 31021

Summary: Tree - Selection listener stops default expansion
Product: [Eclipse] Platform
Component: SWT
Status: RESOLVED WONTFIX
Severity: normal
Priority: P3
Version: 2.1
Target Milestone: ---
Hardware: PC
OS: Windows XP

Description:

R20030205

Run the following example. Double click on a tree item and notice that it does not expand.

Comment out the Selection listener and now double click on any tree item and notice that it expands.

```java
public static void main(String[] args) {
    Display display = new Display();
    Shell shell = new Shell(display);
    shell.setLayout(new FillLayout());
    Tree tree = new Tree(shell, SWT.BORDER);
    for (int i = 0; i < 4; i++) {
        TreeItem item = new TreeItem(tree, SWT.NONE);
        item.setText("Item " + i);
        for (int j = 0; j < 4; j++) {
            TreeItem subItem = new TreeItem(item, SWT.NONE);
            subItem.setText("item " + j);
            for (int k = 0; k < 4; k++) {
                TreeItem subsubItem = new TreeItem(subItem,
                        SWT.NONE);
                subsubItem.setText("item " + k);
            }
        }
    }
    tree.addSelectionListener(new SelectionAdapter() { });
    shell.setSize(200, 200);
    shell.open();
    while (!shell.isDisposed()) {
        if (!display.readAndDispatch())
            display.sleep();
    }
    display.dispose();
}
```

------- Comment #1 From Steve Northover 2004-10-07 19:39:16 --0400-------

This behavior can't be changed now because application code now relies on it. Sorry.
CHARACTERIZING WHICH BUGS GET FIXED IN WINDOWS

Philip J. Guo, Thomas Zimmermann, Nachiappan Nagappan, Brendan Murphy @ ICSE 2010
Not all bugs get fixed

• Bug #1 occurs rarely and affects only a few users.
  – Changes required to fix Bug #1 could be large and expensive.

• Fixing Bug #2 could introduce new bugs
  – Code changes that fix bugs are up to twice as likely to introduce new bugs as other kinds of changes.

• Users are relying on existing behavior
  – Fixing the bug could break their systems.
How do factors related to **people** and **bug report edits** affect whether a bug is successfully resolved as **FIXED**?

→ Inefficiencies in bug triaging process
→ Inform design of tools and policies
Bug handling is pervasive

24,000+ Microsoft employees reported and handled Windows bugs

(vs. 2,000+ Windows developers)
Methodology

Qualitative survey

- “In your experience, how do each of these factors affect the chances of whether a bug will get **successfully resolved as FIXED**?
- 358 out of 1,773 responded.

Quantitative data analysis

- All bug reports for Windows Vista
- Reputation, interest, reassignments, distance
- Logistic regression model for (1) fixed bugs and (2) bugs with “excessive” reassignments
- Manual inspection of bug reports

Follow-up survey

- Reassignment patterns, bug pong
- 118 out of 397 responded
Reputation

“A big influence [on bug fixes] is the reputation of the person opening the bug. If submitter has a history of submitting high quality bugs then new bugs from that person get better attention [...]”

From anonymous survey respondent
Quantifying reputation

For each bug, calculate opener’s reputation by aggregating over all bugs in the past.

\[
\text{bug opener reputation} = \frac{|\text{OPENED} \cap \text{FIXED}|}{|\text{OPENED}| + 1}
\]

Reputation vs. Bugs fixed
Got myth? Reassignments

Reassignments decrease the chances of bugs getting fixed.

**SURVEY:** How does the bug report's activity affect the chances of it being successfully fixed?

- No reassignments
- Few reassignments
- Many reassignments
Reassignments vs. Bugs fixed

Windows Vista

% Fixed

0%

# assignees

© Microsoft Corporation
The need for reassignments

“Bugs many times are exposed in the UI [user interface], but are not caused by the team writing the UI code. These bugs can pass down several layers of components before landing on a lower level component owner.”

From anonymous survey respondent
SURVEY: Other qualitative factors

Textual quality of bug report:
“Just to re-emphasize: The quality of the bug description is very important. Not necessarily filling in the dozens of fields in the bug database with all sorts of crap (build numbers, dates, classifications, etc) - but just the plain-text description of the problem, the implication and maybe even the potential solution.”

Perceived customer/business impact:
“Customer impact can be a very big impact on a bug if evidence exists to show the cost of not fixing the bug for one or more customers.”

Rank and seniority of bug opener:
“A bug opened because something went wrong on a VPs [vice president’s] laptop has better chance [of being fixed] than a bug opened because the same thing happened to an intern.”

Interpersonal skills of bug opener:
“One other ‘soft’ factor is the speaking skill persuasiveness of the developer (or other representative) when arguing for the bug.”
Recommendations

• Train and incentivize employees to write higher-quality bug reports

• Improve awareness of developers’ expertise to minimize reassignments

• Improve communication and trust amongst people in different teams and locations

• Encourage more objectivity in prioritizing and handling bugs
What makes a good bug report?

Thomas Zimmermann, Member, IEEE, Rahul Premraj, Nicolas Bettenburg, Member, IEEE, Sascha Just, Member, IEEE, Adrian Schröter, Member, IEEE, and Cathrin Weiss

Abstract—In software development, bug reports provide crucial information to developers. However, these reports widely differ in their quality. We conducted a survey among developers and users of APACHE, ECLIPSE, and MOZILLA to find out what makes a good bug report. The analysis of the 466 responses revealed an information mismatch between what developers need and what users supply. Most developers consider steps to reproduce, stack traces, and test cases as helpful, which are, at the same time, most difficult to provide for users. Such insight is helpful for designing new bug tracking tools that guide users at collecting and providing more helpful information. Our CUEZILLA prototype is such a tool and measures the quality of new bug reports; it also recommends which elements should be added to improve the quality. We trained CUEZILLA on a sample of 289 bug reports, rated by developers as part of the survey. The participants of our survey also provided 175 comments on hurdles in reporting and resolving bugs. Based on these comments, we discuss several recommendations for better bug tracking systems, which should focus on engaging bug reporters, better tool support, and improved handling of bug duplicates.

Index Terms—Testing and debugging, distribution, maintenance, and enhancement, human factors, management, measurement.

1 INTRODUCTION

Bug reports are vital for any software development. They allow users to inform developers of the problems encountered while using a software. Bug reports typically contain a detailed description of a failure and they occasionally hint at the location of the fault in the code (in the form of patches or stack traces). However, bug reports vary in their quality of content; they often provide length of descriptions, formatting, and presence of stack traces and attachments (such as screenshots). To find out which matter most, we asked 872 developers from the APACHE, ECLIPSE, and MOZILLA projects to:

1. Complete a survey on important information in bug reports and the problems they faced with them. We
The ESE Group at Microsoft

Socio-technical congruence
- Contribution behavior, technical relationships, and quality
- Does organizational structure affect bugs?
- Investigating the effects of geographic distribution

Bug reporting and triage
- What makes a good bug report?
- Characterizing bug reassignments and fixed bugs
- Collaboration and information needs in bug reports

Data-driven software engineering
- Failure-prediction/risk analysis
- Assessment of test-driven development and unit testing
- Analytics for software development
What metrics are the best predictors of failures?

What is the data quality level used in empirical studies and how much does it actually matter?

I just submitted a bug report. Will it be fixed?

How can I tell if a piece of software will have vulnerabilities?

Do cross-cutting concerns cause defects?

If I increase test coverage, will that actually increase software quality?

Are there any metrics that are indicators of failures in both Open Source and Commercial domains?

Should I be writing unit tests in my software project?

Is strong code ownership good or bad for software quality?

Am I using branching in my SCM correctly?

Who adopts new language features and why?

Does Test Driven Development (TDD) produce better code in shorter time?

Does Distributed/Global software development affect quality?
TOWARDS ANALYTICS FOR SOFTWARE DEVELOPMENT

Ray Buse, Thomas Zimmermann @ FoSER 2010
Mining Software Repositories

+

Empirical Software Engineering
Mining Software Repositories + Empirical Software Engineering → Software Development Analytics
Analytics

“Use of analysis, data, and systematic reasoning to make decisions”

Financial services
Retail
Manufacturing
Health care
Energy
And more...
Web analytics 2.0
A long time ago

This Many People Have Been to MY Page:

0123456789
Slide by Ray Buse
Mining Software Repositories + Empirical Software Engineering → Software Development Analytics
Stakeholders have different needs

- Researcher
- Developer
- Tester
- Dev Lead
- Test Lead
- Manager
A single tool is not enough

- Surveys
- Qualitative data
- Benchmarking
- Measurements
Make data actionable and accessible
#1: Data collection

Integration

Data focused

Scenario focused
#2: Data quality
#3: Privacy
#4: Understand user needs
#4: Understand user needs

Developers: well studied
#4: Understand user needs

Managers: not many studies

Developers: well studied
#4: Understand user needs

- Managers: not many studies
- Communication: not many studies
- Developers: well studied
#5: User experience
#6: Education
Make data actionable and accessible

- Education
  - Understand user needs
- Privacy
  - Data quality
  - Data collection
- User experience
Empirical Software Engineering Group (ESE)

The Empirical Software Engineering research area activities at Microsoft Research focus on understanding various software development issues from an empirical perspective. We are involved in doing practical studies on large software systems. All our work is done in conjunction with Microsoft product teams such as Windows and Visual Studio.

Our current interests are in the areas of:

- Software Reliability: Predicting Failures/Failure-proneness, Test Prioritization, Failure Analysis.
- Software Process: Organizational impact on quality, Agile software development, Global software development, Effort estimation
- Empirical Studies: Unit Testing, Inspections, Assertions, Test Driven Development

The ESE group is part of Research in Software Engineering (RiSE).

Publications

Jacek Czerwonka, Rajiv Das, Nachiappan Nagappan, Alex Tarvo, and Alex Teterev, CRANE: Failure Prediction, Change Analysis and Test Prioritization in Practice - Experiences from Windows, in International Conference on Software Testing (ICST) 2011, IEEE, 30 March 2011


http://research.microsoft.com/ese