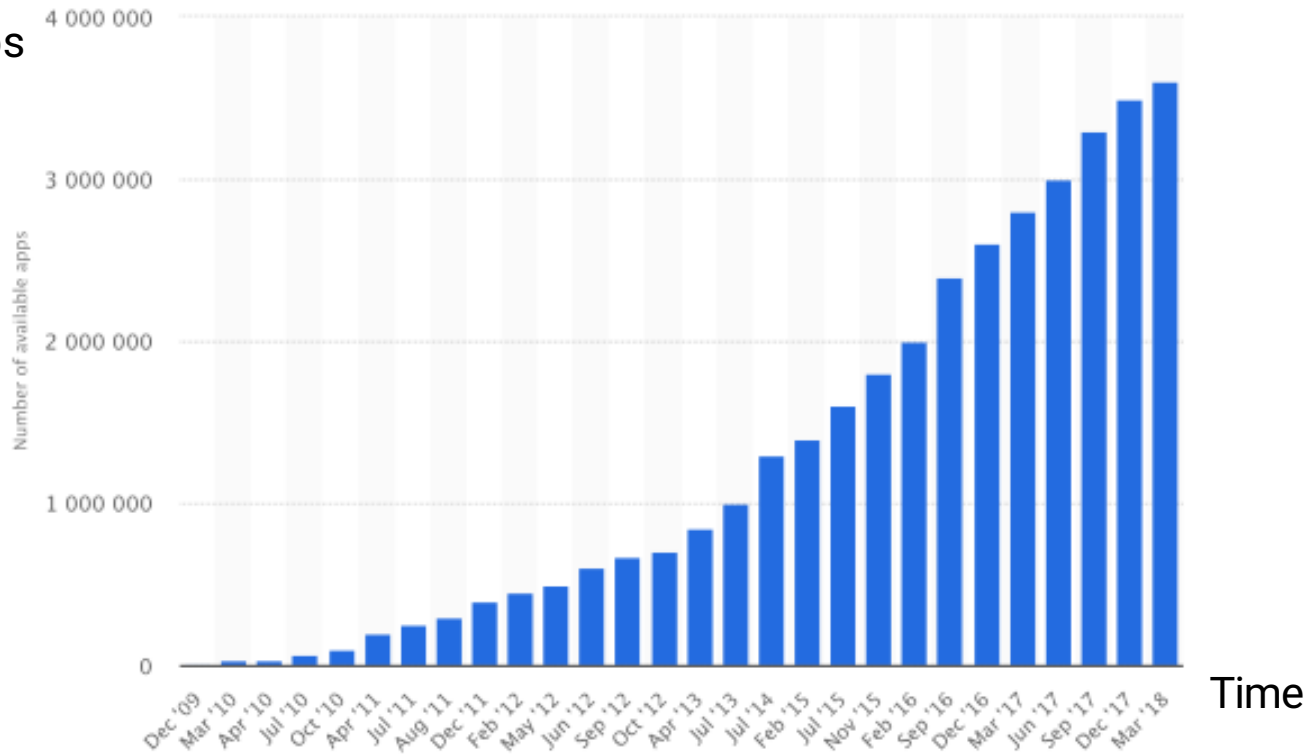
Large-scale Analysis of Framework-specific Exceptions in Android Apps

Lingling Fan, *Ting Su, Sen Chen, Guozhu Meng, Yang Liu, Lihua Xu, Geguang Pu, Zhendong Su*

ACM SIGSOFT Distinguished Paper Award (ICSE2018)

NASAC2018
Shenzhen, China

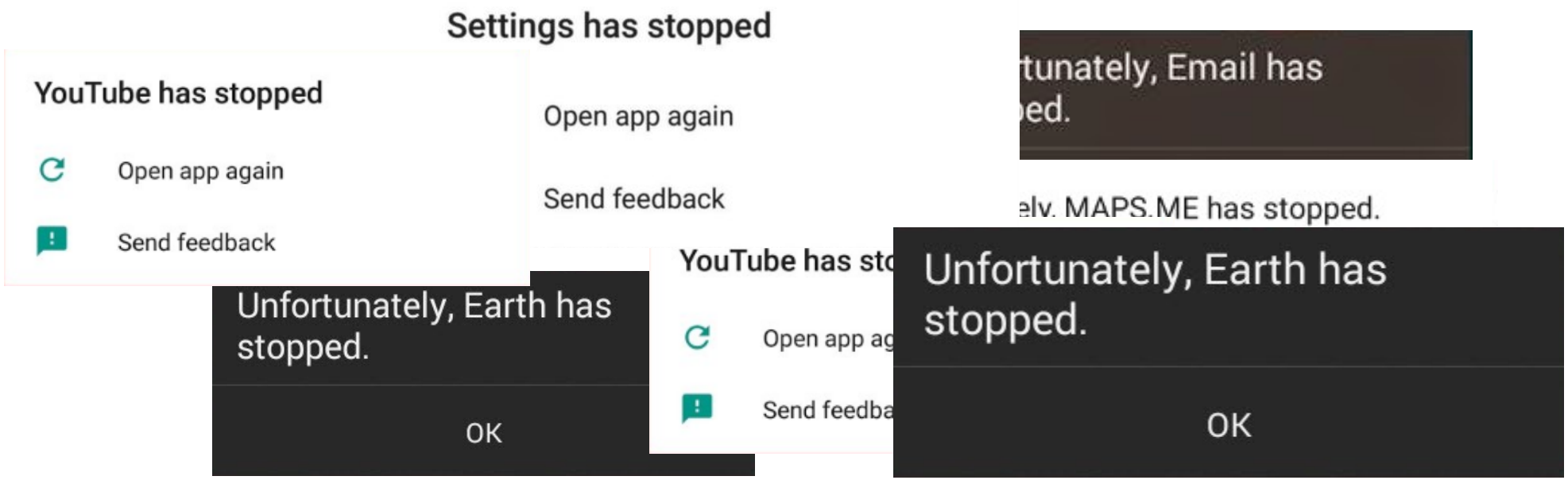
Mobile app is continuously increasing



[statista.com]

One of the key priority is to prevent fail-stop errors, e.g., crash

However……



Apps still suffer from crashes.

Customer complaints

A Lee-Koo

★★☆☆☆ 30 May 2018

Keeps crashing on Android 9 Developer Preview 2, please fix

walter itsadavisthang Youwouldntunderstand

★★☆☆☆ 25 May 2018

Keeps crashing, works shotty !

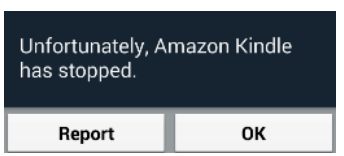
Brittonee Deleveaux

★☆☆☆☆ 27 May 2018

2241

Good for socializing with friends and family but becoming too strict and bias on behavior standards. App now crashes a lot too. Some people are targeted to be banned for 30 days and some are not penalized for the same infraction. Bans cannot be challenged and Face-book carelessly blocks you from

Framework-specific Crash for Android Apps




```
java.lang.RuntimeException: Unable to resume activity {*}:
java.lang.NumberFormatException: Invalid double: “”
    at android.app.ActivityThread.performResumeActivity(…)
    ….
Caused by: java.lang.NumberFormatException: Invalid double:“”
    at java.lang.StringToReal.invalidReal(StringToReal.java:63)
    at java.lang.StringToReal.parseDouble(StringToReal.java:248)
    ….
```

Root exception

Crash signaler

An example of exception trace

Crash Signaler ⇨

- **App**
- **Framework**
- **Lib**

NOTE: We do not consider exceptions caused by the bugs of framework itself.

With the understanding of framework crashes

Developers: avoid and fix crashes

Researchers: improve bug detection tools

However, existing studies on functional bugs analysis:

- Small scale (AST'11, ICST'14)
- Different goals (ICST'14, MSR'15)

  (1) generate testing oracles

  (2) investigate bug hazards of exception-handling code

Analyzing (framework-specific) crashes is challenging

- *Lack of comprehensive dataset*
  - *No publicly available data*
  - *Only 16% issues contain exception traces on Github and Google Code*

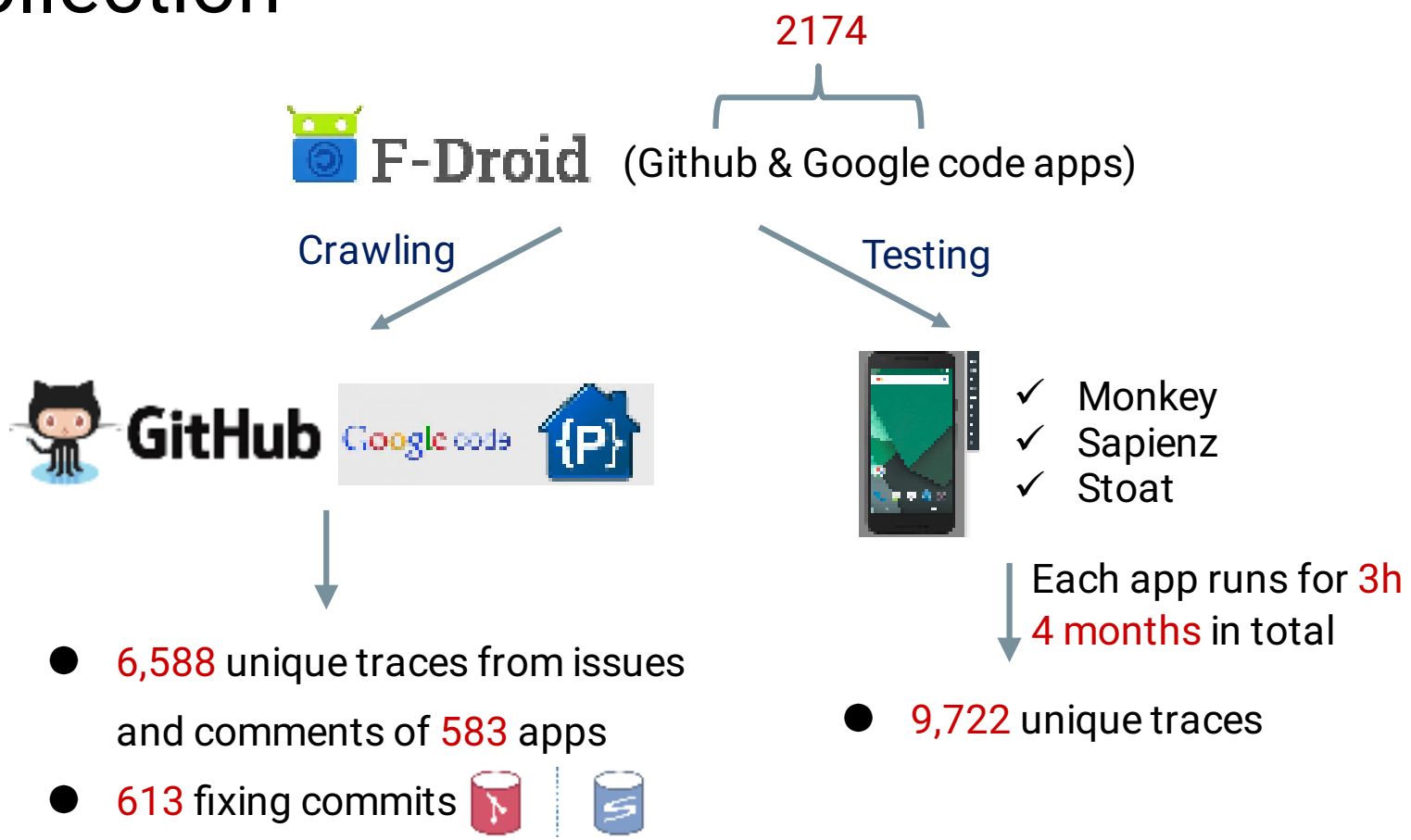
- *Lack of tool support*
  - *Crash reproducing tools*
  - *Failure localization tools*

- *Substantial human effort*
  - *Require understanding of Android framework*

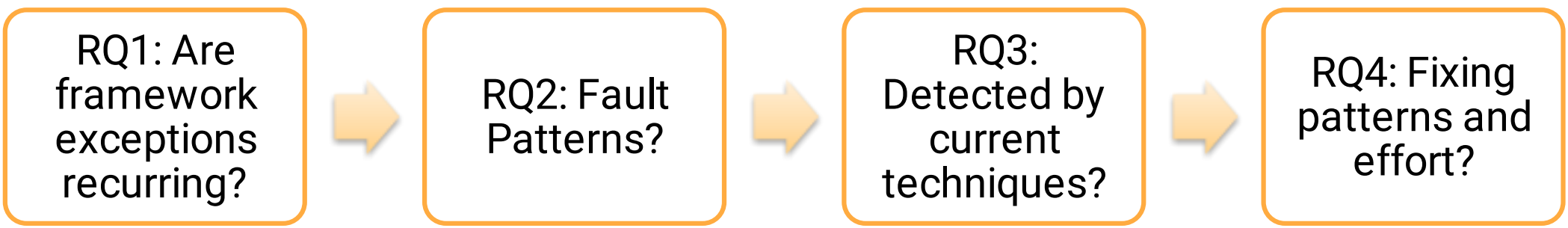
Contribution

- First empirical study to characterize Android framework-specific exceptions
  - 11 fault categories
- Evaluate the state-of-the-art bug detection techniques
  - Static & dynamic tools
- Prototype tools to demonstrate the usefulness of findings
  - Stoat+ & Exlocator
- Publicly available dataset

Data Collection

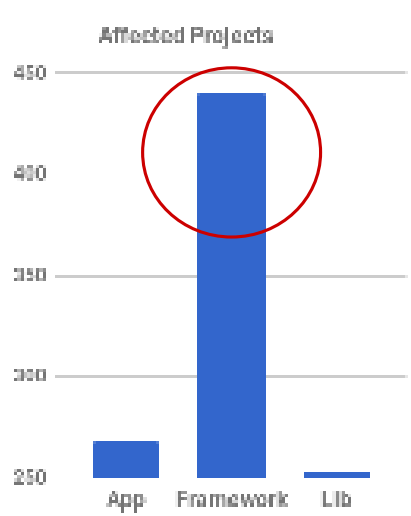


Research Questions

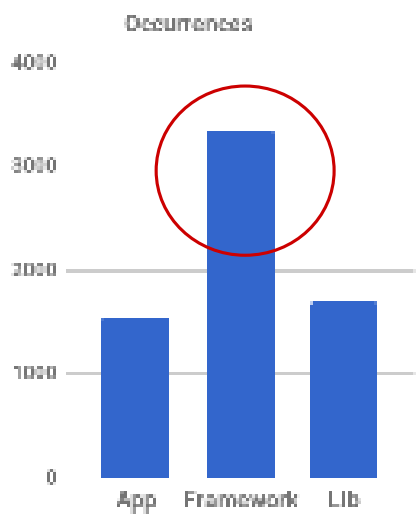


RQ1: Are framework exceptions recurring?

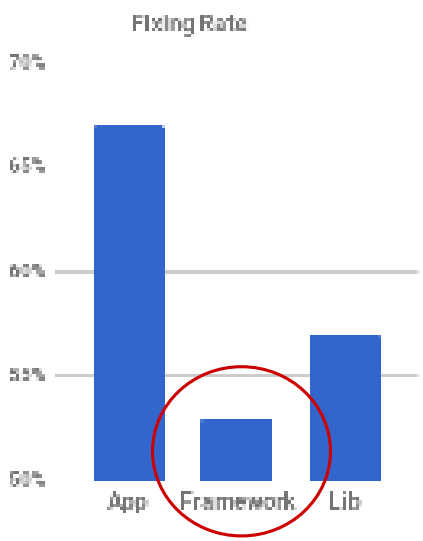
Based on 6,588 unique exceptions from Github and Google code



Affected projects



occurrences

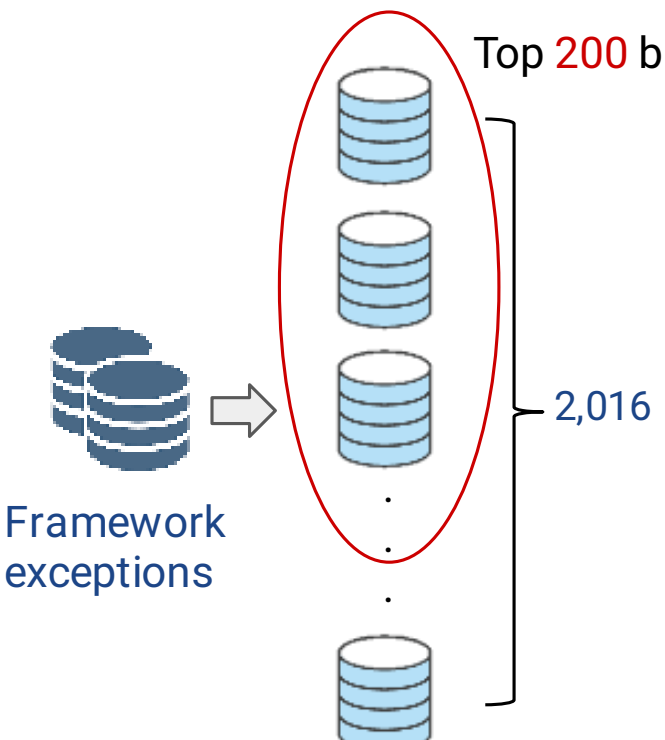


Fixing rate

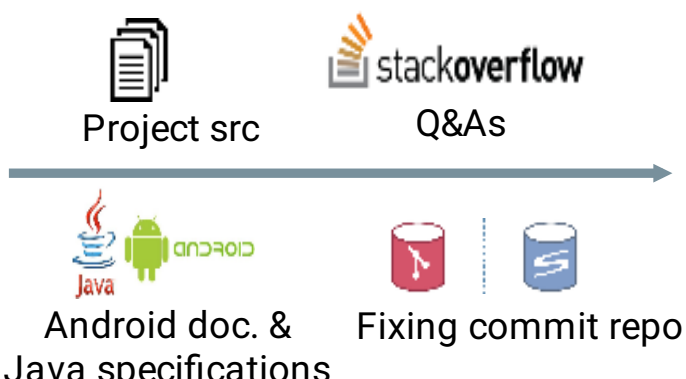
Yes, framework exceptions are more recurring and pervasive

RQ2: Taxonomy of Framework Exceptions

Bucket: repository for exceptions that are thrown from the same location of Android framework



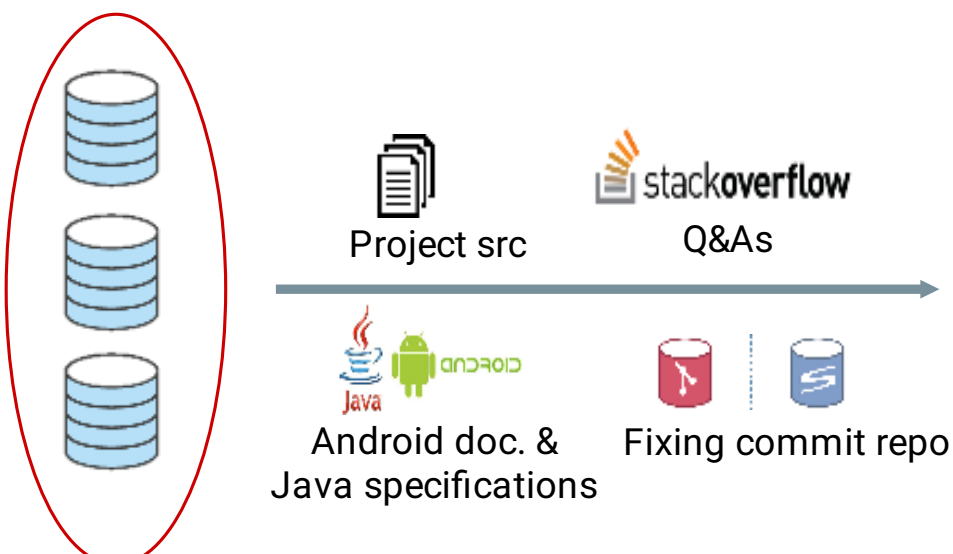
Top 200 buckets occupy over 80% exceptions



11 fault categories covering 84.6% of all framework exceptions

RQ2: Taxonomy of Framework Exceptions

Bucket: repository for exceptions that are thrown from the same location of Android framework



11 fault categories covering 84.6% of all framework exceptions

RQ2: Taxonomy of Framework Exceptions

Category	Occurrence	#S.O. posts
API Updates and Compatibility	68	60
XML Layout Error	122	246
API Parameter Error	820	819
Framework Constraint Error	**383**	**1726**
Index Error	950	218
Database Management Error	128	61
Resource-Not-Found Error	1303	7178
UI Update Error	327	666
Concurrency Error	372	263
Component Lifecycle Error	**608**	**1065**
Memory/Hardware Error	**414**	**792**

Developers make more mistakes on Lifecycle Error, Framework Constraint Error and Memory/Hardware Error.

RQ3: Auditing bug detection tools

Static Tools → **75** different exception instances from 11 categories

Tools	Android support	# Detected (out of 75 exceptions)	# Rules for Android
Lint	✔	**4**	281
FindBugs	✔	0	0
PMD	✔	0	3
SonarQube	✔	0	0

- Existing static analysis tools are ineffective in detecting framework exceptions

RQ3: Auditing bug detection tools

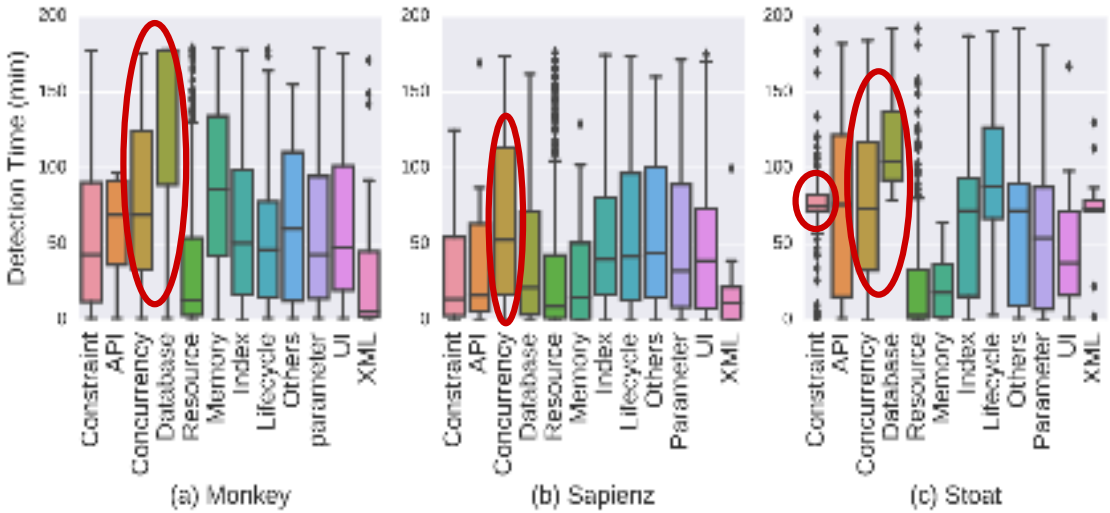
Dynamic Tools

- 2104 apps (4560 versions)
- Each runs for 3h

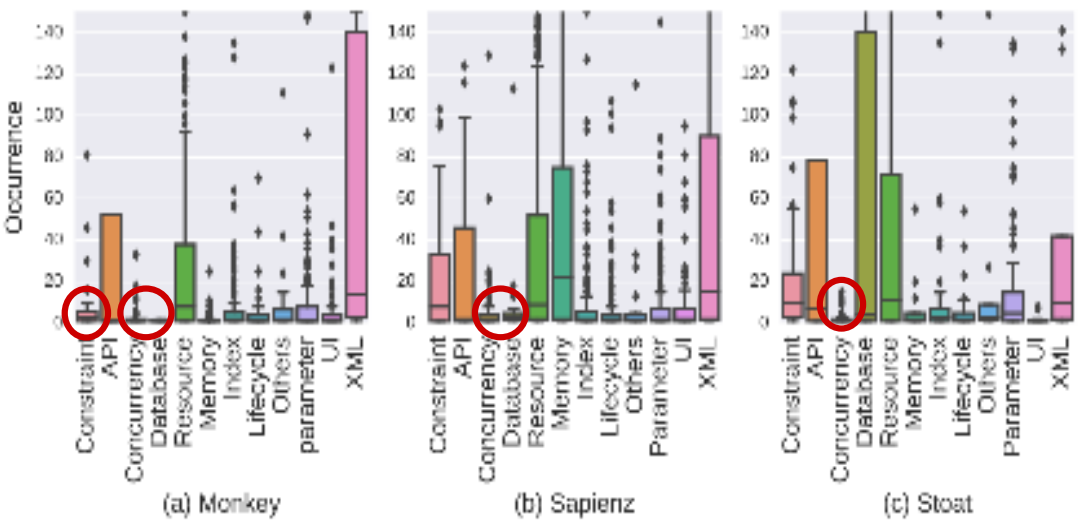
Tools	Approach	# unique framework exceptions
Monkey	Random	1842
Sapienz	Search-based	2342
Stoat	Model-based	1438

Metrics: Detection time & Occurrence

RQ3: Auditing bug detection tools



Detection time: The time of detecting an exception for the first time

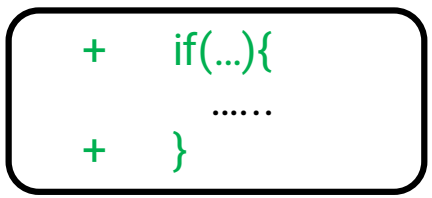


Occurrence: The times of an exception detected during 3 hours

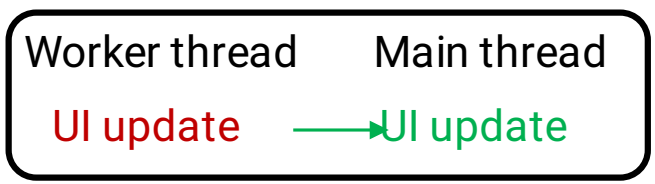
- Dynamic testing tools are still far from effective in detecting database, framework constraint and concurrency errors

RQ4: Fixing Patterns

1. Refine Conditional Checks

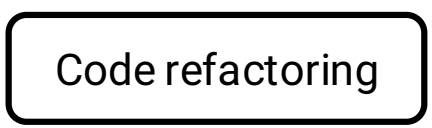


2. Move Code into Correct Thread



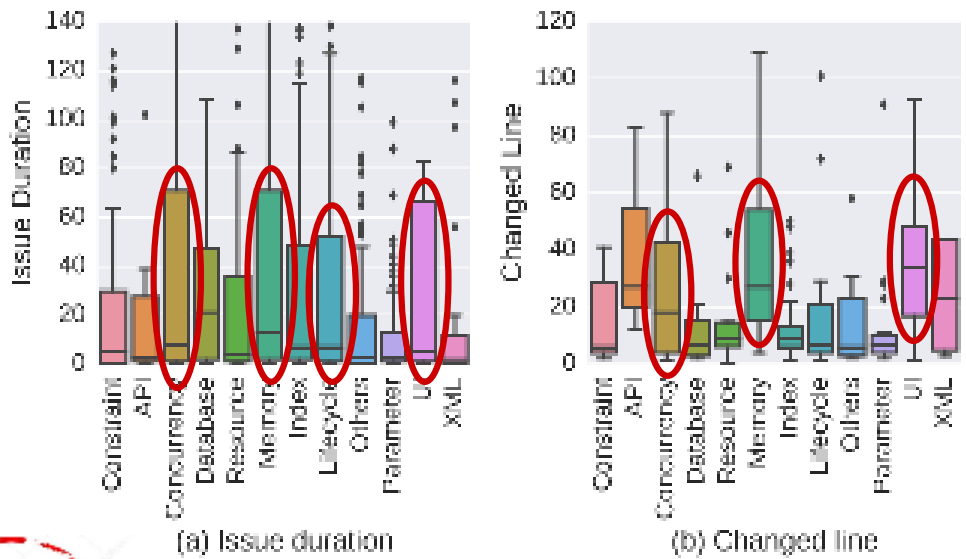
3. Work in Right Callbacks

4. Adjust Implementation Choices



RQ4: Fixing Efforts

- Issue duration: The time cost to fix the issue (day)
- Changed line: Exclude "//...", "@Override", "import *.*"
- Closing rate: The percent of issues being closed



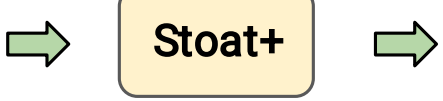
Category	Closing Rate
API Updates and Compatibility	93.9%
XML Layout Error	93.2%
API Parameter Error	88.5%
Framework Constraint Error	87.7%
Index Error	84.1%
Database Management Error	76.8%
Resource-Not-Found Error	75.3%
UI Update Error	**75.0%**
Concurrency Error	**73.5%**
Component Lifecycle Error	**58.8%**
Memory/Hardware Error	**51.6%**

- Lifecyle, Concurrency, UI update and memory errors are more difficult to fix

Applications

(1) Improving Bug Detection

- Meaningful corner cases
  - e.g., “"” and “%”
- Enforce environment interplay
  - Screen rotation
  - Start an activity and quickly back
  - Put the app at background for a long time and navigate to it again



3 previously unknown bugs

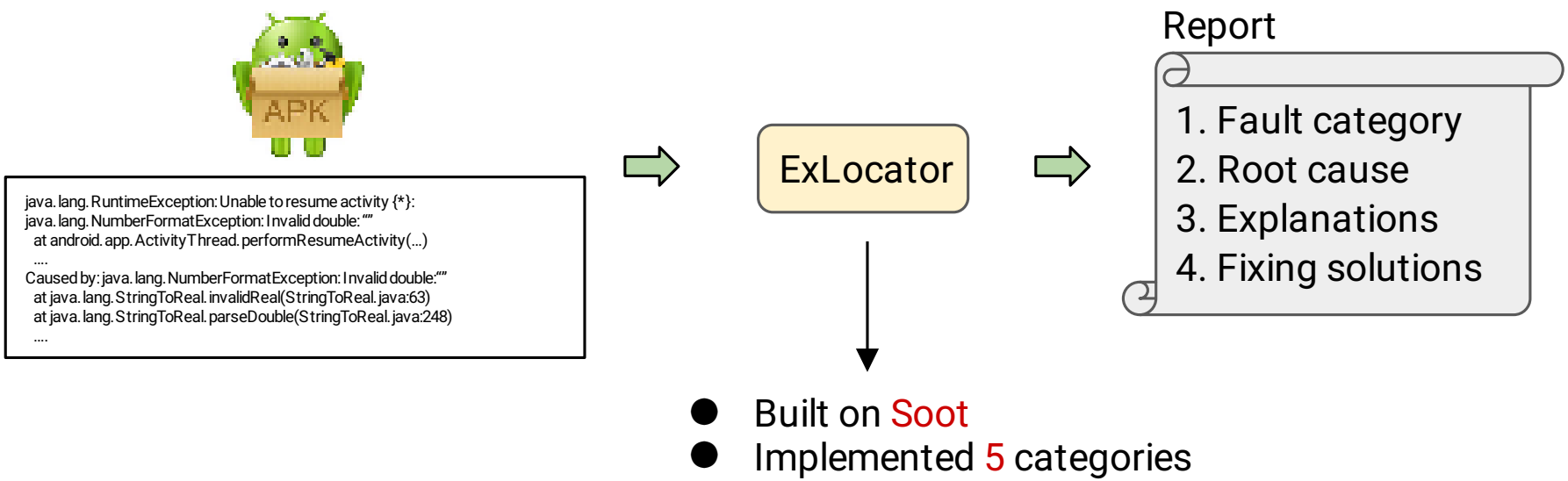
- Parameter error

- UI update error
- Lifecycle error

https://github.com/tingsu/Stoat

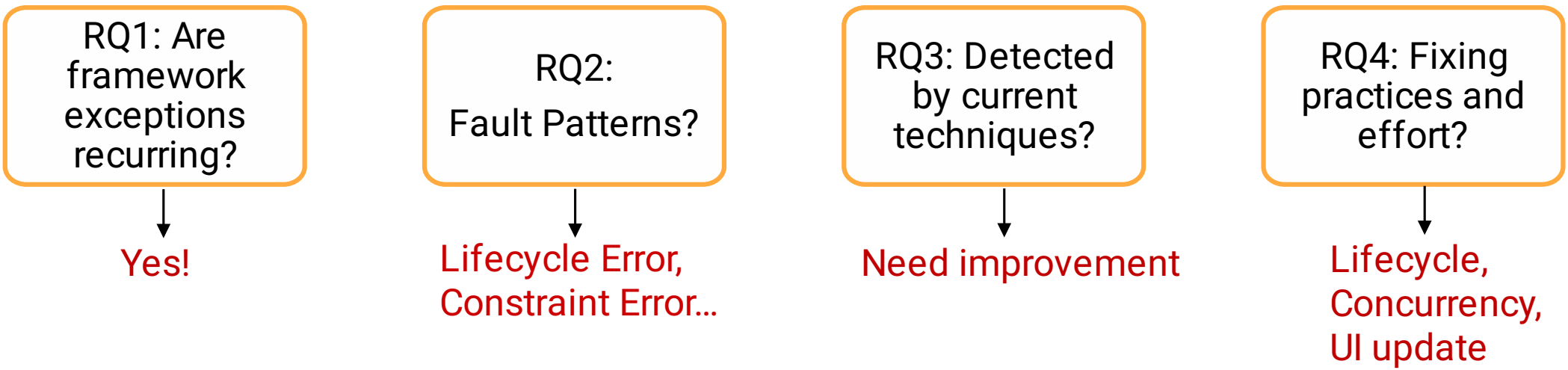
Applications

(2) Enabling Exception Localization



25 out of 27 exceptions (92% precision) are correctly located by comparing the patches from the developers.

Conclusions



- First large-scale analysis of Android framework-specific exceptions
- Supporting follow-up research on bug detection, fault localization and patch generation
- Large-scale and reusable dataset available on

https://crashanalysis.github.io/Dataset-CrashAnalysis

Contact: ecnujanefan@gmail.com

Homepage: http://www.sqslab.com/llfan/