Fixr: Mining and Understanding Bug Fixes for App-Framework Protocol Defects

University of Colorado Boulder

MUSE Demonstration Workshop: Technical Talk
February 1, 2016
The Android framework is constantly changing
The Android framework is constantly changing

My app seems broken on the new Android 6.0.1. How do I update my app?
The Android framework is constantly changing

My app seems broken on the new Android 6.0.1. How do I update my app?

As an app developer, what happened in the last Android update that I need to be aware of now?
The Android framework is constantly changing.

The Big Android Chart: Android Platform Version History

My app seems broken on the new Android 6.0.1. How do I update my app?

100s of API packages, 1,000s of API classes, 10,000s+ of API methods (as of API 23)

As an app developer, what happened in the last Android update that I need to be aware of now?
Do bugfixes trend?

My app seems broken on the new Android 6.0.1. How do I update my app?
Do bugfixes trend?

Hypothesis: App fixes for a framework update have a time signature
Do bugfixes trend?

Hypothesis: App fixes for a framework update have a time signature
Do bugfixes trend?

Hypothesis: App fixes for a framework update have a time signature
**Hypothesis:** App fixes for a framework update have a **time signature**

A **bugfix pattern** for changes in how an app should use method $m$? 

**Do bugfixes trend?**

**Help!** My app seems broken on the new Android 6.0.1. How do I update my app?
API Usage
Trend Analysis
Need: Find API usage trends—APIs that devs are “touching” in commits over time.
Need: Find API usage trends—APIs that devs are “touching” in commits over time.
Need: Find API usage trends—APIs that devs are “touching” in commits over time.

Enables an analyst to explore API trends to find interesting patterns of API usage.
Need: Find API usage trends—APIs that devs are “touching” in commits over time.

Enables an analyst to explore API trends to find interesting patterns of API usage

anyone interested in bug patterns and fixes for Android protocol defects (e.g., devs, quality assurance engineers, security auditors)
Need: Find API usage trends—APIs that devs are “touching” in commits over time.

Enables an analyst to explore API trends to find interesting patterns of API usage.

Demonstration on Wednesday for anyone interested in bug patterns and fixes for Android protocol defects (e.g., devs, quality assurance engineers, security auditors).
API Usage
Trend Analysis
Is the trend relevant?
Hypothesis: **Bugfix commits** can be found in the corpus
Is the trend relevant?

Hypothesis: **Bugfix commits** can be found in the corpus

Oh, I have in my code:
```java
void onDestroy() {
    mContext.unregisterReceiver(this);
}
```
Is the trend relevant?

Hypothesis: **Bugfix commits** can be found in the corpus

Oh, I have in my code:

```java
void onDestroy() {
    mContext.unregisterReceiver(this);
}
```

I want to find relevant commits, such as:

```java
void onDestroy() {
    mContext.unregisterReceiver(this);
}

void onStop() {
    c.unregisterReceiver(this);
}
```
void onDestroy() {
    c.unregisterReceiver(this);
}

void onStop() {
    c.unregisterReceiver(this);
}
Need: Find **commits** that are relevant to a code snippet
Relevant Commit Search

Need: Find **commits** that are relevant to a code snippet

```java
import android.media.MediaScannerConnection;

public class Query extends AppCompatActivity implements MediaScannerConnection.

private MediaScannerConnection mConnection;

protected void onCreate() {
    MediaScannerConnection.scanFile(this,  
        new String[] { "" },  
        null,  
        null);
}
```

```java
void onDestroy() {
    c.unregisterReceiver(this);
}
```

```java
void onStop() {
    c.unregisterReceiver(this);
}
```
Relevant Commit Search

Need: Find **commits** that are relevant to a code snippet

Enables an **analyst** to explore commits to find potential bugfixes
Relevant Commit Search

Enables an **analyst** to explore commits to find potential bugfixes
Relevant Commit Search

Enables an analyst to explore commits to find potential bugfixes
Relevant Commit Search

Enables an analyst to explore commits to find potential bug fixes.
Relevant Commit Search

Enables an analyst to explore commits to find potential bugfixes

Demonstration on Wednesday
Relevant Commit Search

Enables an analyst to explore commits to find potential bugfixes.

Demonstration on Wednesday

Specifies a bug condition and a fixing transformation.

Repair specification
Are repair specifications applicable?
Are repair specifications applicable?

Hypothesis: API repairs are applicable “in the wild”
Are repair specifications applicable?

Hypothesis: API **repairs** are applicable “in the wild”
Are repair specifications applicable?

Hypothesis: API repairs are applicable “in the wild”
Semantic Code Search-and-Repair
Semantic Code Search-and-Repair

Need: Find repositories satisfying the (semantic) bug condition and patch
Semantic Code Search-and-Repair

Need: Find repositories satisfying the (semantic) bug condition and patch
Semantic Code Search-and-Repair

Need: Find repositories satisfying the (semantic) bug condition and patch

Enables an analyst to apply repairs “in the wild”
Fixr Workflow

API Usage Trend Analysis → Relevant Commit Search → Semantic Search-and-Repair → Patched Github
Inspect the trends to find common patterns of API usage.
Inspect the trends to find common patterns of API usage

Write a query: snippet of Java code that uses the API in a relevant way
Inspect the trends to find common patterns of API usage

Inspect the commits to understand the bug and the fix

API Usage Trend Analysis

Write a query: snippet of Java code that uses the API in a relevant way

Semantic Search-and-Repair

Patched Github
**Fixr Workflow**

1. **API Usage Trend Analysis**
   - Inspect the trends to find common patterns of API usage

2. **Relevant Commit Search**
   - Inspect the commits to understand the bug and the fix

3. **Semantic Search-and-Repair**
   - Write a repair specification: specifies what is the bug and how it can be repaired

4. **Patched Github**
   - Write a query: snippet of Java code that uses the API in a relevant way
Fixr Workflow

Inspect the trends to find common patterns of API usage

Inspect the commits to understand the bug and the fix

API Usage Trend Analysis

Write a query: snippet of Java code that uses the API in a relevant way

Relevant Commit Search

Semantic Search-and-Repair

Write a repair specification: specifies what is the bug and how it can be repaired

Patched Github
Feature Extraction on Commits
Feature Extraction on Commits

API Usage Trend Analysis

Relevant Commit Search

Semantic Search-and-Repair
Feature Extraction on Commits

API Usage Trend Analysis

![Graph showing API usage trend analysis]

Relevant Commit Search

```java
void onDestroy() {
    c.unregisterReceiver(this);
}
```

```java
void onStop() {
    c.unregisterReceiver(this);
}
```

Semantic Search-and-Repair

![Icon representing semantic search-and-repair]
Feature Extraction on Commits

API Usage Trend Analysis

Relevant Commit Search

void onDestroy() {
    c.unregisterReceiver(this);
}

void onStop() {
    c.unregisterReceiver(this);
}

Semantic Search-and-Repair
Feature Extraction on Commits

Github

DATA

API Usage Trend Analysis

void onDestroy() {
    c.unregisterReceiver(this);
}

void onStop() {
    c.unregisterReceiver(this);
}

Relevant Commit Search

Semantic Search-and-Repair

repair

specification
Feature Extraction on Commits

Need: Extract features from code commits

API Usage Trend Analysis

Relevant Commit Search

Semantic Search-and-Repair

Github

DATA

void onDestroy() {
    c.unregisterReceiver(this);
}

void onStop() {
    c.unregisterReceiver(this);
}
Fixr Contributions
Fixr Contributions

Github ➔ DATA
Extract commit features at scale
Fixr Contributions

Extract commit features at scale

Find API usage patterns over time
Fixr Contributions

Extract commit features at scale

Find API usage patterns over time

Index commit feature documents
Fixr Contributions

Extract commit features at scale

Find API usage patterns over time

Index commit feature documents

Search-and-repair platform for Android apps
Commits + Time Series = Trends
Commits + Time Series = Trends

Semantic Repair Specification
Commits + Time Series = Trends

Semantic Repair Specification

Android API Evolution
Fixr Contributions

Extract commit features at scale

Find API usage patterns over time

Index commit feature documents

Search-and-repair platform for Android apps
Fixr Contributions

Extract commit features at scale

Find API usage patterns over time

Index commit feature documents

Search-and-repair platform for Android apps
Extracting commit features
Approach
Extracting commit features

Approach

Github

Data Extraction
Extracting commit features
Approach

Github

Data Extraction

Commit Extraction
Extracting commit features

Approach

Github

Data Extraction

Commit Extraction

Java source code
Extracting commit features

Approach

Github Data Extraction

Commit Extraction

Feature Extraction

Java source code

Data Extraction
Extracting commit features

Approach

Github

Data Extraction

Commit Extraction

Feature Extraction

Java source code

API calls, imports
Extracting commit features

Approach

Data Extraction

Commit Extraction

Feature Extraction

Github

Java source code

API calls, imports

eclipse
Extracting commit features

Approach

Data Extraction

Commit Extraction

Feature Extraction

Feature Processing

Github

Java source code

API calls, imports

source code

API calls, imports

Commit Extraction

Feature Extraction

Feature Processing

Github

Java source code

API calls, imports

Commit Extraction

Feature Extraction

Feature Processing

Github

Java source code

API calls, imports

Commit Extraction

Feature Extraction

Feature Processing

Github

Java source code

API calls, imports

Commit Extraction

Feature Extraction

Feature Processing

Github

Java source code

API calls, imports

Commit Extraction

Feature Extraction

Feature Processing

Github

Java source code

API calls, imports

Commit Extraction

Feature Extraction

Feature Processing

Github

Java source code

API calls, imports

Commit Extraction

Feature Extraction

Feature Processing

Github

Java source code

API calls, imports

Commit Extraction

Feature Extraction

Feature Processing

Github

Java source code

API calls, imports
Extracting commit features

Approach

1. Github
2. Commit Extraction
3. Feature Extraction
4. Feature Processing

Data Extraction

- Java source code
- API calls, imports
- Parent-child commit relations
For a Java file, pairs of extracted features from the commits that change that file.
For a Java file, pairs of extracted features from the commits that change that file.
For a Java file, pairs of extracted features from the commits that change that file.
For a Java file, pairs of extracted features from the commits that change that file.
For a Java file, pairs of extracted features from the commits that change that file.
For a Java file, pairs of extracted features from the commits that change that file.

A Spark-based data processing pipeline with custom data targets.
The scale of feature processing

Commit Extraction → Feature Extraction → Feature Processing

- Github
- Java source code
- API calls, imports
- Parent-child commit relations

Data Extraction

Bit vectors

{JSON}
commit documents
The scale of feature processing

Commit Extraction → Feature Extraction → Feature Processing

16K repos with 510K code commits
The scale of feature processing

2.83M changed files in 3 hours

16K repos with 510K code commits

Commit Extraction → Feature Extraction → Feature Processing

Data Extraction

Github

2.83M changed files in 3 hours

16K repos with 510K code commits

Commit Extraction

Feature Extraction

Feature Processing

parent-child commit-relations

API calls, imports

Java source code

bit vectors

commit documents

{JSON}
The scale of feature processing

- **2.83M** changed files in **3 hours**
- **16K repos with 510K code commits**
- **125M code features in 28 hours**

Diagram:
- Github
- Commit Extraction
- Feature Extraction
- Feature Processing
- API calls, imports
- Java source code
- parent-child commit-relations
- bit vectors
- commit documents
The scale of feature processing

- 2.83M changed files in 3 hours
- 1.39M commit-relations in 4 hours
- 16K repos with 510K code commits
- 125M code features in 28 hours
- 1.39M commit-relations in 4 hours
- 2.83M changed files in 3 hours
The scale of feature processing

- **2.83M** changed files in **3 hours**
- **16K** repos with **510K** code commits
- **1.39M** commit-relations in **4 hours**
- **125M** code features in **28 hours**
- **4.22M** documents

**Data Extraction**
- Commit Extraction
- Feature Extraction

**Source code**
- Java
- API calls, imports
- parent-child commit-relations

**Bit vectors**
The scale of feature processing

2.83M changed files in 3 hours

1.39M commit-relations in 4 hours

16K repos with 510K code commits

125M code features in 28 hours

Note: Measurements on a cluster with relatively limited resources (4 workers)

4.22M documents
Fixr Contributions

Extract commit features at scale

Find API usage patterns over time

Index commit feature documents

Search-and-repair platform for Android apps
Fixr Contributions

Extract commit features at scale

Find API usage patterns over time

Index commit feature documents

Search-and-repair platform for Android apps
API Usage Trend Analysis
Approach
Research Question: Do bugfixes exhibit a time signature?
Research Question: Do bugfixes exhibit a time signature?

Extract patterns of API usage
API Usage Trend Analysis

Approach

Research Question: Do bugfixes exhibit a time signature?

Extract patterns of API usage

Association Rule: If method $m_1, m_2, ..., m_k$ are involved in a commit, then method $m$ is also involved with high probability.
Research Question: Do bugfixes exhibit a time signature?

Extract patterns of API usage

Association Rule: If method $m_1, m_2, \ldots, m_k$ are involved in a commit, then method $m$ is also involved with high probability.

Compute time signature for each pattern
API Usage Trend Analysis
Approach

Research Question: Do bugfixes exhibit a time signature?

Extract patterns of API usage

Association Rule: If method $m_1, m_2, \ldots, m_k$ are involved in a commit, then method $m$ is also involved with high probability.

Compute time signature for each pattern

Fraction of commits matching a rule over time.

![Graph showing trend analysis](image)
API Usage Trend Analysis
Status

Github -> Data Extraction
API Usage Trend Analysis

Status

[Diagram of Github linked to Data Extraction]

API calls involved per commit

+setTag(…)
-registerReceiver(…)
-getTag(…)

Github → Data Extraction
API Usage Trend Analysis
Status

Github

Data Extraction

API calls involved per commit

Trend Analysis

+setTag(...)
-registerReceiver(...)
-getTag(...)

API Usage Trend Analysis

Status

Github

Data Extraction

API calls involved per commit

Trend Analysis

API Pattern Mining

+setTag(...)
-registerReceiver(...) 
-getTag(....)
API Usage Trend Analysis
Status

API calls involved per commit

Github

Data Extraction

Trend Analysis

API Pattern Mining

If unregisterReceiver(...) then registerReceiver(...) with strength 0.91

+setTag(...)

-registerReceiver(...)

-getTag(...)
API Usage Trend Analysis
Status

Github
Data Extraction
API calls involved per commit
API call patterns
Trend Analysis
Temporal Mining

+setTag(…)
-registerReceiver(…)
-getTag(…)

If unregisterReceiver(…)
then registerReceiver(…)
with strength 0.91
API Usage Trend Analysis

Status

Data Extraction

API calls involved per commit

API call patterns

Trend Analysis

Temporal Mining

+setTag(…)
-registerReceiver(…)
-getTag(…)

If unregisterReceiver(…)
then registerReceiver(…)
with strength 0.91
API Usage Trend Analysis
Status

API calls involved per commit

Data Extraction

Github

API call patterns

Temporal Mining

Pattern Violations

+setTag(...) -registerReceiver(...) -getTag(...)

If unregisterReceiver(...) then registerReceiver(...) with strength 0.91
API Usage Trend Analysis

Status

Data Extraction

API calls involved per commit

API call patterns

Trend Analysis

Temporal Mining

Pattern Violations

+setTag(…)
-registerReceiver(…)
-getTag(…)

If unregisterReceiver(…)
then registerReceiver(…)
with strength 0.91
API Usage Trend Analysis

Status

Data Extraction → API Pattern Mining → Temporal Mining → Pattern Violations

- setTag(…)
- registerReceiver(…)
- getTag(…)

If unregisterReceiver(…)
then registerReceiver(…)
with strength 0.91

16K repos with 510K code commits
API Usage Trend Analysis

Status

Data Extraction

API calls involved per commit

API Pattern Mining

Temporal Mining

Pattern Violations

16K repos with 510K code commits

16.5K API methods on 1.4M commit-relations

If unregisterReceiver(...) then registerReceiver(...) with strength 0.91
API Usage Trend Analysis

Status

Github

Data Extraction

API calls involved per commit

API call patterns

Trend Analysis

API Pattern Mining

Temporal Mining

Pattern Violations

16K repos with 510K code commits

16.5K API methods on 1.4M commit-relations

+setTag(...)

-regReceiver(...)

-getTag(...)

If unregisterReceiver(...) then registerReceiver(...) with strength 0.91

1200 API association rules in 23 minutes
API Usage Trend Analysis

Status

- **Github**
- **Data Extraction**
- **API Calls Involved per Commit**
- **API Pattern Mining**
- **Temporal Mining**
- **Pattern Violations**

**API Call Patterns**

- `+setState(...)`
- `-register(...)`
- `-getTag(...)`
- `if unregister(...)
  then register(...)`

**65 Trending Rules**

**16K Repos with 510K Code Commits**

**16.5K API Methods on 1.4M Commit Relations**

**1200 API Association Rules in 23 Minutes**
API Usage Trend Analysis

Status

16K repos with 510K code commits

16.5K API methods on 1.4M commit relations

+setTag(…)
-registerReceiver(…)
-getTag(…)

If unregisterReceiver(…)
then registerReceiver(…)

1200 API association rules in 23 minutes

Temporal Mining

Pattern Violations

API call patterns

Trend Analysis

60 violations manually examined from 8 rules

25 bugs

65 trending rules
Alarm example:
A commit violating a mined association rule
Alarm example:
A commit violating a mined association rule

Rule: setTransactionSuccessful → endTransaction
Alarm example:
A commit violating a mined association rule

Rule: setTransactionSuccessful → endTransaction
- 1556 commits matching this rule
- 81 potential violations of this rule
- 5 manually identified bugs, 40 false alarms
Alarm example: A commit violating a mined association rule

**Rule:** setTransactionSuccessful → endTransaction

- 1556 commits matching this rule
- 81 potential violations of this rule
- 5 manually identified bugs, 40 false alarms
Alarm example:
A **commit** violating a mined association rule

**Rule:** `setTransactionSuccessful → endTransaction`

- 1556 commits matching this rule
- 81 potential violations of this rule
- 5 manually identified bugs, 40 false alarms

```java
public void syncUsers(List<com.ch_linghu.fanfoudroid.data.User> users){
    SQLiteDatabase mdb = mOpenHelper.getWritableDatabase();
    mdb.beginTransaction();
    for(com.ch_linghu.fanfoudroid.data.User u:users){
        if(existsUser(u.id)){
            updateUser(u);
        }else{
            createUserInfo(u);
        }
    }
    mdb.setTransactionSuccessful();
}
```

https://github.com/fanfoudroid/fanfoudroid/blob/66d7f7d621cffd290f0d0291e49128b6bef109e4/src/com/ch_linghu/fanfoudroid/data/db/TwitterDatabase.java
Alarm example: A commit violating a mined association rule

Rule: `setTransactionSuccessful → endTransaction`

1556 commits matching this rule
81 potential violations of this rule
5 manually identified bugs, 40 false alarms

```java
/* 同步用户,更新已存在的用户,插入未存在的用户 */

public void syncUsers(List<com.ch_linghu.fanfoudroid.data.User> users) {
    SQLiteDatabase mDb = mOpenHelper.getWritableDatabase();
    mDb.beginTransaction();
    for (com.ch_linghu.fanfoudroid.data.User u : users) {
        if (existsUser(u.id)) {
            updateUser(u);
        } else {
            createUserInfo(u);
        }
    }
    mDb.setTransactionSuccessful();
}
```

https://github.com/fanfoudroid/fanfoudroid/blob/66d7f7d621cfd290f0d0291e49128b6bef109e4/src/com/ch_linghu/fanfoudroid/data/db/TwitterDatabase.java
Alarm example: A commit violating a mined association rule

**Rule:** `setTransactionSuccessful \rightarrow endTransaction`

1556 commits matching this rule
81 potential violations of this rule
5 manually identified bugs, 40 false alarms

```
/* 同步用户，更新已存在的用户，插入未存在的用户 */

public void syncUsers(List<com.ch_linghu.fanfoudroid.data.User> users) {
    SQLiteOpenHelper mDb = mOpenHelper.getWritableDatabase();
    mDb.beginTransaction();
    for (com.ch_linghu.fanfoudroid.data.User u : users) {
        if (existsUser(u.id)) {
            updateUser(u);
        } else {
            createUserInfo(u);
        }
    }
    mDb.setTransactionSuccessful();
}
```

**Bug:** Missing call to `endTransaction` in a try-finally block. Database may end in an inconsistent state.

https://github.com/fanfoudroid/fanfoudroid/blob/66d7f7d621cfd290f0d0291e49128b6bef109e4/src/com/ch_linghu/fanfoudroid/data/db/TwitterDatabase.java
Another alarm example:
A commit violating a mined association rule
Another alarm example:
A commit violating a mined association rule

**Rule**: `obtainStyledAttributes → recycle`

- 4210 commits matching this rule.
- 467 potential violations of this rule
- 8 manually identified bugs, 2 false alarms
Another alarm example:
A **commit** violating a mined association rule

**Rule:** `obtainStyledAttributes → recycle`

- 4210 commits matching this rule.
- 467 potential violations of this rule
- 8 **manually identified bugs**, 2 false alarms
Another alarm example:
A commit violating a mined association rule

**Rule:** `obtainStyledAttributes` → `recycle`

4210 commits matching this rule.
467 potential violations of this rule
8 manually identified bugs, 2 false alarms

“Be sure to call `recycle` when done with …”
Another alarm example:
A commit violating a mined association rule

Rule: `obtainStyledAttributes → recycle`

- 4210 commits matching this rule.
- 467 potential violations of this rule
- 8 manually identified bugs, 2 false alarms

```java
public MaterialColorPalette(Context context, AttributeSet attrs) {
    super(context, attrs);

    TypedArray array = context.getTheme()
        .obtainStyledAttributes(attrs, R.styleable.MaterialColorPalette, 0, 0);
    int normalColorsResId =
        array.getResourceId(R.styleable.MaterialColorPalette_color_palette_normal, 0);
    int pressedColorsResId =
        array.getResourceId(R.styleable.MaterialColorPalette_color_palette_pressed, 0);

    if (normalColorsResId != 0) {
        mNormalColorList = getResources().getIntArray(normalColorsResId);
    }
    if (pressedColorsResId != 0) {
        mPressedColorList = getResources().getIntArray(pressedColorsResId);
    } else {
        mPressedColorList = mNormalColorList;
    }

    LinearLayoutManager layoutManager = new LinearLayoutManager(getContext());
    layoutManager.setOrientation(LinearLayoutManager.HORIZONTAL);
    setLayoutManager(layoutManager);
    setAdapter(new ColorPaletteAdapter());
}
```


"Be sure to call recycle when done with ..."
Another alarm example:
A **commit** violating a mined association rule

**Rule:** `obtainStyledAttributes → recycle`

- 4210 commits matching this rule.
- 467 potential violations of this rule
- 8 manually identified **bugs**, 2 false alarms

```
public MaterialColorPalette(Context context, AttributeSet attrs) {
    super(context, attrs);

    TypedArray array = context.getTheme()
        .obtainStyledAttributes(attrs, R.styleable.MaterialColorPalette, 0, 0);
    int normalColorsResId =
        array.getResourceId(R.styleable.MaterialColorPalette_color_palette_normal, 0);
    int pressedColorsResId =
        array.getResourceId(R.styleable.MaterialColorPalette_color_palette_pressed, 0);

    if (normalColorsResId != 0) {
        mNormalColorList = getResources().getIntArray(normalColorsResId);
    }

    if (pressedColorsResId != 0) {
        mPressedColorList = getResources().getIntArray(pressedColorsResId);
    } else {
        mPressedColorList = mNormalColorList;
    }

    LinearLayoutManager layoutManager = new LinearLayoutManager(getActivity());
    layoutManager.setOrientation(LinearLayoutManager.HORIZONTAL);
    setLayoutManager(layoutManager);
    setAdapter(new ColorPaletteAdapter());
}
```


“Be sure to call `recycle` when done with ...”
Another alarm example:
A **commit** violating a mined association rule

**Rule:** `obtainStyledAttributes \rightarrow recycle`

4210 commits matching this rule.
467 potential violations of this rule
8 manually identified **bugs**, 2 false alarms

```
public MaterialColorPalette(Context context, AttributeSet attrs) {
    super(context, attrs);
    TypedArray array = context.getTheme()
        .obtainStyledAttributes(attrs, R.styleable.MaterialColorPalette, 0, 0);
    int normalColorsResId =
        array.getResourceId(R.styleable.MaterialColorPalette_color_palette_normal, 0);
    int pressedColorsResId =
        array.getResourceId(R.styleable.MaterialColorPalette_color_palette_pressed, 0);
    if (normalColorsResId != 0) {
        mNormalColorList = getResources().getIntArray(normalColorsResId);
    }
    if (pressedColorsResId != 0) {
        mPressedColorList = getResources().getIntArray(pressedColorsResId);
    } else {
        mPressedColorList = mNormalColorList;
    }
    LinearLayoutManager layoutManager = new LinearLayoutManager(
        layoutManager.setOrientation(LinearLayoutManager.VERTICAL);
    layoutManager.setAdapter(new ColorPaletteAdapter()));
}
```

“Be sure to call **recycle** when done with …”

**Bug:** Missing call to `array.recycle()` before `array` goes out of scope.

Test time signatures for trends
Basic test for trending. A rule is trending iff there is a time interval that exceeds three standard deviations from the mean.
Test time signatures for trends

Strength: 0.921907
Matches/Violating: 1818 / 154
Test time signatures for trends

No clear trend for

\{\text{hasNext}, \text{iterator}, \text{remove}\} \Rightarrow \text{next}
Fixr Contributions

- Extract commit features at scale
- Find API usage patterns over time
- Index commit feature documents
- Search-and-repair platform for Android apps
Fixr Contributions

Extract commit features at scale

Find API usage patterns over time

Index commit feature documents

Search-and-repair platform for Android apps
Research Question: Can bugfix commits be found in the corpus with indexed search?
Research Question: Can bugfix commits be found in the corpus with indexed search?

Oh, I have in my code:

```java
MediaScannerConnection.scanFile(context, p, m, l)
```
Research Question: Can bugfix commits be found in the corpus with indexed search?

Oh, I have in my code:

```java
MediaPlayerConnection.scanFile(context, p, m, l)
```
Relevant Commit Search

[Diagram showing a connection from a Github icon to a Data Extraction box]
Relevant Commit Search

Idea: Create and index commit feature documents
Idea: Create and index commit feature documents
Relevant Commit Search

Oh, I have in my code:

```java
MediaScannerConnection.scanFile(context, p, m, l)
```

Idea: Create and index commit feature documents
Relevant Commit Search

Oh, I have in my code:

```java
MediaScannerConnection.scanFile(context, p, m, l)
```

Idea: Create and index commit feature documents
Oh, I have in my code:

```java
MediaScannerConnection.scanFile(context, p, m, l)
```
Relevant Commit Search

Oh, I have in my code:

```java
MediaScannerConnection.scanFile(context, p, m, l)
```

**Idea:** Create and index commit feature documents
Relevant Commit Search

Oh, I have in my code:

```
MediaScannerConnection.scanFile(context, p, m, l)
```

Idea: Create and index commit feature documents
Oh, I have in my code:

```
MediaScannerConnection.scanFile(context, p, m, l)
```

**Idea:** Create and index commit feature documents
Relevant Commit Search

Oh, I have in my code:

```
MediaPlayerConnection.scanFile(context, p, m, l)
```

Idea: Create and index commit feature documents

Challenge: Index “useful” features

GitHub

Data Extraction → JSON

Solr
Oh, I have in my code:

```java
MediaScannerConnection.scanFile(context, p, m, l)
```

**Idea:** Create and index commit feature documents

**Challenge:** Index “useful” features

For each commit-relation, we index:

- Features of the parent commit
- Features of the child commit
- Parent-to-child patch
- Commit messages
Can *bugfix commits* be found in the corpus with indexed search?
Can bugfix commits be found in the corpus with indexed search?

Do the results improve using more fine-grained features?
16K repos with 2.83M Java files

1.39M commit-relation documents
16K repos with 2.83M Java files

1.39M commit-relation documents

5 bug pattern queries
Relevant Commit Search
Experimental Setup

16K repos with 2.83M Java files
1.39M commit-relation documents

5 bug pattern queries

3 granularity levels
  method names
  method names + arity
  method names + arity + “fix” in commit message
Empirical Evaluation
Empirical Evaluation

Bug Pattern

- View.setTag
- MediaScanner
- Connection.scanFile
- getSystemService(CAMERA_SERVICE)
- GoogleApiClient
  - .Builder
- MediaPlayer
  - .setDataSource
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
</tr>
<tr>
<td>MediaScanner</td>
<td>name</td>
</tr>
<tr>
<td>Connection.scanFile</td>
<td>+arity</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
</tr>
<tr>
<td>getSystemService (CAMERA_SERVICE)</td>
<td>name</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
</tr>
<tr>
<td>GoogleApiClient.Builder</td>
<td>name</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
</tr>
<tr>
<td>MediaPlayer</td>
<td>name</td>
</tr>
<tr>
<td>.setDataSource</td>
<td>+arity</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
</tr>
<tr>
<td>summary</td>
<td></td>
</tr>
</tbody>
</table>
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
<th>Query Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>16</td>
</tr>
<tr>
<td>MediaScanner</td>
<td>name</td>
<td>26</td>
</tr>
<tr>
<td>Connection.scanFile</td>
<td>+arity</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>18</td>
</tr>
<tr>
<td>getSystemService</td>
<td>name</td>
<td>24</td>
</tr>
<tr>
<td>(CAMERA_SERVICE)</td>
<td>+arity</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>23</td>
</tr>
<tr>
<td>GoogleApiClient.Builder</td>
<td>name</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>2</td>
</tr>
<tr>
<td>MediaPlayer.setDataSource</td>
<td>name</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>10</td>
</tr>
<tr>
<td><strong>summary</strong></td>
<td></td>
<td><strong>20</strong></td>
</tr>
</tbody>
</table>
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
<th>Query Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>16</td>
</tr>
<tr>
<td>MediaScanner</td>
<td>name</td>
<td>26</td>
</tr>
<tr>
<td>Connection.scanFile</td>
<td>+arity</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>18</td>
</tr>
<tr>
<td>getSystemService(CAMERA_SERVICE)</td>
<td>name</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>23</td>
</tr>
<tr>
<td>GoogleApiClient.Builder</td>
<td>name</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>2</td>
</tr>
<tr>
<td>MediaPlayer.setDataSource</td>
<td>name</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>10</td>
</tr>
<tr>
<td>summary</td>
<td></td>
<td>20</td>
</tr>
</tbody>
</table>

*dominated by feature extraction*
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
<th>Query Time (sec)</th>
<th>Commits Retrieved (num)</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
<td>138</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
<td>86</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>16</td>
<td>19</td>
</tr>
<tr>
<td>MediaScanner</td>
<td>name</td>
<td>26</td>
<td>75</td>
</tr>
<tr>
<td>Connection.scanFile</td>
<td>+arity</td>
<td>30</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>18</td>
<td>11</td>
</tr>
<tr>
<td>getSystemService (CAMERA_SERVICE)</td>
<td>name</td>
<td>24</td>
<td>82</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>14</td>
<td>82</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>23</td>
<td>139</td>
</tr>
<tr>
<td>GoogleApiClient.Builder</td>
<td>name</td>
<td>30</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>MediaPlayer.setDataSource</td>
<td>name</td>
<td>45</td>
<td>192</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>15</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>10</td>
<td>21</td>
</tr>
</tbody>
</table>

**summary** | 20

*dominated by feature extraction*
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
<th>Query Time (sec)</th>
<th>Commits Retrieved (num)</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
<td>138</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
<td>86</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>16</td>
<td>19</td>
</tr>
<tr>
<td>MediaScanner Connection.scanFile</td>
<td>name</td>
<td>26</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>30</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>18</td>
<td>11</td>
</tr>
<tr>
<td>getSystemService (CAMERA_SERVICE)</td>
<td>name</td>
<td>24</td>
<td>82</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>14</td>
<td>82</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>23</td>
<td>139</td>
</tr>
<tr>
<td>GoogleApiClient.Builder</td>
<td>name</td>
<td>30</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>MediaPlayer.setDataSource</td>
<td>name</td>
<td>45</td>
<td>192</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>15</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>10</td>
<td>21</td>
</tr>
</tbody>
</table>

**summary** 20

out of 1.39M commit-relations <0.01%
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
<th>Query Time (sec)</th>
<th>Commits Retrieved (num)</th>
<th>Bugs in First 10 (num)</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
<td>138</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
<td>86</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>16</td>
<td>19</td>
<td>1</td>
</tr>
<tr>
<td>MediaScanner Connection.scanFile</td>
<td>name</td>
<td>26</td>
<td>75</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>30</td>
<td>51</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>18</td>
<td>11</td>
<td>5</td>
</tr>
<tr>
<td>getSystemService (CAMERA_SERVICE)</td>
<td>name</td>
<td>24</td>
<td>82</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>14</td>
<td>82</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>23</td>
<td>139</td>
<td>1</td>
</tr>
<tr>
<td>GoogleApiClient.Builder</td>
<td>name</td>
<td>30</td>
<td>23</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>10</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>MediaPlayer.setDataSource</td>
<td>name</td>
<td>45</td>
<td>192</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>15</td>
<td>135</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>10</td>
<td>21</td>
<td>0</td>
</tr>
<tr>
<td><strong>summary</strong></td>
<td><strong>name</strong></td>
<td><strong>20</strong></td>
<td><strong>2</strong></td>
<td><strong>2</strong></td>
</tr>
</tbody>
</table>

*out of 1.39M commit-relations <0.01%*
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
<th>Query Time (sec)</th>
<th>Commits Retrieved (num)</th>
<th>Bugs in First 10 (num)</th>
<th>Fixes in First 10 (num)</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
<td>138</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
<td>86</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>16</td>
<td>19</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>MediaScanner</td>
<td>name</td>
<td>26</td>
<td>75</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Connection.scanFile</td>
<td>+arity</td>
<td>30</td>
<td>51</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>18</td>
<td>11</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>getSystemService (CAMERA_SERVICE)</td>
<td>name</td>
<td>24</td>
<td>82</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>GoogleApiClient.Builder</td>
<td>+arity</td>
<td>14</td>
<td>82</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>23</td>
<td>139</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>MediaPlayer</td>
<td>name</td>
<td>30</td>
<td>23</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>.setDataSource</td>
<td>+arity</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>name</td>
<td>45</td>
<td>192</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>15</td>
<td>135</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>10</td>
<td>21</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>summary</strong></td>
<td><strong>name</strong></td>
<td><strong>20</strong></td>
<td><strong>2</strong></td>
<td><strong>2</strong></td>
<td><strong>1</strong></td>
</tr>
</tbody>
</table>

*Out of 1,39M commit-relations <0.01%*
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
<th>Query Time (sec)</th>
<th>Commits Retrieved (num)</th>
<th>Bugs in First 10 (num)</th>
<th>Fixes in First 10 (num)</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
<td>138</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
<td>86</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>16</td>
<td>19</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>MediaScanner</td>
<td>name</td>
<td>26</td>
<td>75</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Connection.scanFile</td>
<td>+arity</td>
<td>30</td>
<td>51</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>18</td>
<td>11</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>getSystemService</td>
<td>name</td>
<td>24</td>
<td>82</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>(CAMERA_SERVICE)</td>
<td>+arity</td>
<td>14</td>
<td>82</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>23</td>
<td>139</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>GoogleApiClient.Builder</td>
<td>name</td>
<td>30</td>
<td>23</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>MediaPlayer.setDataSource</td>
<td>name</td>
<td>45</td>
<td>192</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>15</td>
<td>135</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>10</td>
<td>21</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>summary</td>
<td></td>
<td>20</td>
<td></td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Bugs and fixes in triaging first 10
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
<th>Query Time (sec)</th>
<th>Commits Retrieved (num)</th>
<th>Bugs in First 10 (num)</th>
<th>Fixes in First 10 (num)</th>
<th>Cumulative Score in the First 30% (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
<td>138</td>
<td>3</td>
<td>0</td>
<td>69</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
<td>86</td>
<td>1</td>
<td>1</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>16</td>
<td>19</td>
<td>1</td>
<td>1</td>
<td>44</td>
</tr>
<tr>
<td>MediaScanner</td>
<td>name</td>
<td>26</td>
<td>75</td>
<td>5</td>
<td>1</td>
<td>69</td>
</tr>
<tr>
<td>Connection.scanFile</td>
<td>+arity</td>
<td>30</td>
<td>51</td>
<td>2</td>
<td>0</td>
<td>66</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>18</td>
<td>11</td>
<td>5</td>
<td>2</td>
<td>67</td>
</tr>
<tr>
<td>getSystemService</td>
<td>name</td>
<td>24</td>
<td>82</td>
<td>1</td>
<td>0</td>
<td>88</td>
</tr>
<tr>
<td>(CAMERA_SERVICE)</td>
<td>+arity</td>
<td>14</td>
<td>82</td>
<td>1</td>
<td>0</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>23</td>
<td>139</td>
<td>1</td>
<td>0</td>
<td>32</td>
</tr>
<tr>
<td>GoogleApiClient</td>
<td>name</td>
<td>30</td>
<td>23</td>
<td>7</td>
<td>0</td>
<td>65</td>
</tr>
<tr>
<td>.Builder</td>
<td>+arity</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>0</td>
<td>82</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MediaPlayer</td>
<td>name</td>
<td>45</td>
<td>192</td>
<td>0</td>
<td>0</td>
<td>53</td>
</tr>
<tr>
<td>.setDataSource</td>
<td>+arity</td>
<td>15</td>
<td>135</td>
<td>0</td>
<td>0</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>10</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>57</td>
</tr>
<tr>
<td><strong>summary</strong></td>
<td></td>
<td>20</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

*bugs and fixes in triaging first 10*
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
<th>Query Time (sec)</th>
<th>Commits Retrieved (num)</th>
<th>Bugs in First 10 (num)</th>
<th>Fixes in First 10 (num)</th>
<th>Cumulative Score in the First 30% (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
<td>138</td>
<td>3</td>
<td>0</td>
<td>69</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
<td>86</td>
<td>1</td>
<td>1</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>16</td>
<td>19</td>
<td>1</td>
<td>1</td>
<td>44</td>
</tr>
<tr>
<td>MediaScanner</td>
<td>name</td>
<td>26</td>
<td>75</td>
<td>5</td>
<td>1</td>
<td>69</td>
</tr>
<tr>
<td>Connection.scanFile</td>
<td>+arity</td>
<td>30</td>
<td>51</td>
<td>2</td>
<td>0</td>
<td>66</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>18</td>
<td>11</td>
<td>5</td>
<td>2</td>
<td>67</td>
</tr>
<tr>
<td>getSystemService (CAMERA_SERVICE)</td>
<td>name</td>
<td>24</td>
<td>82</td>
<td>1</td>
<td>0</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>14</td>
<td>82</td>
<td>1</td>
<td>0</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>23</td>
<td>139</td>
<td>1</td>
<td>0</td>
<td>32</td>
</tr>
<tr>
<td>GoogleApiClient.Builder</td>
<td>name</td>
<td>30</td>
<td>23</td>
<td>7</td>
<td>0</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>0</td>
<td>82</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MediaPlayer.setDataSource</td>
<td>name</td>
<td>45</td>
<td>192</td>
<td>0</td>
<td>0</td>
<td>53</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>15</td>
<td>135</td>
<td>0</td>
<td>0</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>10</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>57</td>
</tr>
</tbody>
</table>

### Summary

- **Granularity**
  - `name`
  - `+arity`
  - `+“fix”`

- **Query Time (sec)**
  - `summary`: 20

- **Commits Retrieved (num)**
  - `summary`: 2

- **Bugs in First 10 (num)**
  - `summary`: 2

- **Fixes in First 10 (num)**
  - `summary`: 0

- **Cumulative Score in the First 30% (%)**
  - `summary`: 69

### Note

The fraction of score in the first 30% is shown in the last column.
## Empirical Evaluation

<table>
<thead>
<tr>
<th>Bug Pattern</th>
<th>Granularity</th>
<th>Query Time (sec)</th>
<th>Commits Retrieved (num)</th>
<th>Bugs in First 10 (num)</th>
<th>Fixes in First 10 (num)</th>
<th>Cumulative Score in the First 30% (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
<td>138</td>
<td>3</td>
<td>0</td>
<td>69</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
<td>86</td>
<td>1</td>
<td>1</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>16</td>
<td>19</td>
<td>1</td>
<td>1</td>
<td>44</td>
</tr>
<tr>
<td>MediaScanner</td>
<td>name</td>
<td>26</td>
<td>75</td>
<td>5</td>
<td>1</td>
<td>69</td>
</tr>
<tr>
<td>Connection.scanFile</td>
<td>+arity</td>
<td>30</td>
<td>51</td>
<td>2</td>
<td>0</td>
<td>66</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>18</td>
<td>11</td>
<td>5</td>
<td>2</td>
<td>67</td>
</tr>
<tr>
<td>getSystemService (CAMERA_SERVICE)</td>
<td>name</td>
<td>24</td>
<td>82</td>
<td>1</td>
<td>0</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>14</td>
<td>82</td>
<td>1</td>
<td>0</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>23</td>
<td>139</td>
<td>1</td>
<td>0</td>
<td>32</td>
</tr>
<tr>
<td>GoogleApiClient .Builder</td>
<td>name</td>
<td>30</td>
<td>23</td>
<td>7</td>
<td>0</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>0</td>
<td>82</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MediaPlayer .setDataSource</td>
<td>name</td>
<td>45</td>
<td>192</td>
<td>0</td>
<td>0</td>
<td>53</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>15</td>
<td>135</td>
<td>0</td>
<td>0</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>+“fix”</td>
<td>10</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>57</td>
</tr>
<tr>
<td><strong>summary</strong></td>
<td></td>
<td>20</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bug Pattern</td>
<td>Granularity</td>
<td>Query Time (sec)</td>
<td>Commits Retrieved (num)</td>
<td>Bugs in First 10 (num)</td>
<td>Fixes in First 10 (num)</td>
<td>Cumulative Score in the First 30% (%)</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
<td>------------------</td>
<td>-------------------------</td>
<td>------------------------</td>
<td>------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>View.setTag</td>
<td>name</td>
<td>14</td>
<td>138</td>
<td>3</td>
<td>0</td>
<td>69</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>17</td>
<td>86</td>
<td>1</td>
<td>1</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>+&quot;fix&quot;</td>
<td>16</td>
<td>19</td>
<td>1</td>
<td>1</td>
<td>44</td>
</tr>
<tr>
<td>MediaScanner</td>
<td>name</td>
<td>26</td>
<td>75</td>
<td>5</td>
<td>1</td>
<td>69</td>
</tr>
<tr>
<td>Connection.scanFile</td>
<td>+arity</td>
<td>30</td>
<td>51</td>
<td>2</td>
<td>0</td>
<td>66</td>
</tr>
<tr>
<td></td>
<td>+&quot;fix&quot;</td>
<td>18</td>
<td>11</td>
<td>5</td>
<td>2</td>
<td>67</td>
</tr>
<tr>
<td>getSystemService (CAMERA_SERVICE)</td>
<td>name</td>
<td>24</td>
<td>82</td>
<td>1</td>
<td>0</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>14</td>
<td>82</td>
<td>1</td>
<td>0</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>+&quot;fix&quot;</td>
<td>23</td>
<td>139</td>
<td>1</td>
<td>0</td>
<td>32</td>
</tr>
<tr>
<td>GoogleApiClient.Builder</td>
<td>name</td>
<td>30</td>
<td>23</td>
<td>7</td>
<td>0</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>0</td>
<td>82</td>
</tr>
<tr>
<td></td>
<td>+&quot;fix&quot;</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MediaPlayer.setDataSource</td>
<td>name</td>
<td>45</td>
<td>192</td>
<td>0</td>
<td>0</td>
<td>53</td>
</tr>
<tr>
<td></td>
<td>+arity</td>
<td>15</td>
<td>135</td>
<td>0</td>
<td>0</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>+&quot;fix&quot;</td>
<td>10</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>57</td>
</tr>
<tr>
<td>summary</td>
<td></td>
<td>20</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Bugs and fixes can be found in the first 10 results with feature-based document search.
Fixr Contributions

Extract commit features at scale

Find API usage patterns over time

Index commit feature documents

Search-and-repair platform for Android apps
Fixr Contributions

- Extract commit features at scale
- Find API usage patterns over time
- Index commit feature documents
- Search-and-repair platform for Android apps
Are repair specifications applicable?

Hypothesis: API repairs are applicable “in the wild”
Are repair specifications applicable?

Hypothesis: API repairs are applicable “in the wild”
Are repair specifications applicable?

Hypothesis: API repairs are applicable “in the wild”

Bug found “in the wild”
(at HEAD as of January 2016 in a project active as of August 2015)
Bug
Bug

MediaScannerConnection.scanFile(context, p, m, l)
if context points to an object of type Activity or l can reach an Activity
if `context` points to an object of type `Activity` or 1 can reach an Activity
if context points to an object of type Activity or l can reach an Activity
If `context` points to an object of type `Activity` or `l` can reach an `Activity`
if context points to an object of type Activity or l can reach an Activity

- MediaScannerConnection.scanFile({context}, {p}, {m}, {l})

+final {MSCCWrapper} {fresh w} = new {MSCCWrapper}({context}, {p}, {m});
+final MediaScannerConnection {fresh msc} = new {MSCCWrapper}({l}, {w});
+{w}.startConnection({msc});

syntactic transformation language
if context points to an object of type Activity or l can reach an Activity

Repair

-MediaScannerConnection.scanFile({context}, {p}, {m}, {l})

+final {MSCCWrapper} {fresh w} = new {MSCCWrapper}({context}, {p}, {m});
+final MediaScannerConnection {fresh msc} = new {MSCWrapper}({l}, {w});
+{w}.startConnection({msc});
if context points to an object of type Activity or 1 can reach an Activity

- MediaScannerConnection.scanFile({{context}}, {{p}}, {{m}}, {{l}})

+final {{MSCCWrapper}} {{fresh w}} = new {{MSCCWrapper}}({{context}}, {{p}}, {{m}});
+final MediaScannerConnection {{fresh msc}} = new {{MSCCWrapper}}({{l}}, {{w}});
+{{w}}.startConnection({{msc}});
Search-and-repair platform for Android apps

Approach
Search-and-repair platform for Android apps

Approach

Github
Search-and-repair platform for Android apps

Approach

Github + repair specification
Search-and-repair platform for Android apps

Approach

Github + repair specification → Relevant Commit Search
Search-and-repair platform for Android apps

Approach

Github + repair specification → Relevant Commit Search → Relevant Commits
Search-and-repair platform for Android apps

Approach

Github + repair specification → Relevant Commit Search → Relevant Commits
Search-and-repair platform for Android apps

Approach
Search-and-repair platform for Android apps

Approach

- Github
  - Relevant Commits
  - Repair specification

- Relevant Commits
  - Repair specification

→ Relevant Commit Search

→ Static Analysis

→ Relevant Commits
Search-and-repair platform for Android apps

Approach

Github + repair specification → Relevant Commit Search

Relevant Commits + repair specification → Static Analysis

Relevant Commits

Static Analysis

Relevant Commits

patch locations
Search-and-repair platform for Android apps

Approach

- Github
- Relevant Commits
- Relevant Commit Search
- Static Analysis
- Patch locations
Search-and-repair platform for Android apps

Approach

Github

Relevant Commits

+ repair specification

Relevant Commit Search

Static Analysis

+ repair specification

patch locations

Relevant Commits

Relevant Commits

patch locations
Search-and-repair platform for Android apps

**Approach**

1. **Github** + **repair specification** → **Relevant Commit Search** → **Relevant Commits**
2. **Relevant Commits** + **repair specification** → **Static Analysis** → **patch locations**
3. **Relevant Commits** + **repair specification** + **patch locations**
Search-and-repair platform for Android apps

Approach

- Github + repair specification → Relevant Commit Search → Relevant Commits
- Relevant Commits + repair specification → Static Analysis → patch locations
- Relevant Commits + repair specification + patch locations → Abstract Syntactic Patch
Search-and-repair platform for Android apps

Approach

- Github
- Relevant Commits
- Relevant Commit Search
- Static Analysis
- Abstract Syntactic Patch
- Patched Github
Search-and-repair platform for Android apps

Approach

Decouples search into **syntactic** filtering before applying **semantic** queries.
Search-and-repair platform for Android apps

Approach

1. Github + Relevant Commits

2. Relevant Commit Search

3. Static Analysis

4. Abstract Syntactic Patch

5. Patched Github
Search-and-repair platform for Android apps

Approach

- Converts **semantic** queries into **syntactic** locations.

  - **Static Analysis**
  - **Abstract Syntactic Patch**

  - **Relevant Commit**
  - **Patch**
  - **Patched**

- +

  - Github
  - Relevant Commits
  - Relevant Commits
  - Relevant Commit
  - Relevant Commits
  - Relevant Commits

- +

  - Repair specification
  - Patch locations
  - Repair specification
  - Patch locations

- +

  - Patched Github
Search-and-repair platform for Android apps

Status: API repairs applicable “in the wild” today?

Github + repair specification → Relevant Commit Search → Relevant Commits

Relevant Commits + repair specification → Static Analysis → patch locations

Relevant Commits + repair specification + patch locations → Abstract Syntactic Patch → Patched Github
A search-and-repair platform for Android apps

16K repos with 510K code commits

API repairs applicable “in the wild” today?

Patch locations

Relevant Commit Search

Static Analysis

Abstract Syntactic Patch

Patched Github
Search-and-repair platform for Android apps

16K repos with 510K code commits

3 repair specifications → Relevant Commit Search → Relevant Commits

Relevant Commits + repair specification → Static Analysis → patch locations

Relevant Commits + repair specification + patch locations → Abstract Syntactic Patch → Patched Github

API repairs applicable “in the wild” today?

16K repos with 510K code commits

3 repair specifications

Relevant Commits

Static Analysis

Abstract Syntactic Patch

Patched Github
Search-and-repair platform for Android apps

Status: API repairs applicable “in the wild” today?

- Github
- Relevant Commits
- Repair specification

 Relevant Commit Search

- Static Analysis
- Abstract Syntactic Patch

48 repos with relevant commits

32
Search-and-repair platform for Android apps

Status: API repairs applicable “in the wild” today?

48 repos with relevant commits

Github

+ repair specification

Relevant Commit Search

22 repos buildable at HEAD

Static Analysis

Abstract Syntactic Patch

Patched Github

32
Search-and-repair platform for Android apps

Status: API **repairs** applicable “in the wild” today?

**Difficulty**: Enormous variability in source code organization and “cleanliness”.

Like to work with Leidos in Phase 2 on this issue.

48 repos with relevant commits

22 repos buildable at HEAD

48 repos with relevant commits

**48 repos with relevant commits**

**Status**: API repairs applicable “in the wild” today?

**Relevant Commit Search**

**Static Analysis**

**Abstract Syntactic Patch**

**Difficulty**: Enormous variability in source code organization and “cleanliness”.

Like to work with Leidos in Phase 2 on this issue.
Search-and-repair platform for Android apps

Status: API repairs applicable “in the wild” today?

- Github
  + repair specification
  → Relevant Commit Search

- Relevant Commits
  + repair specification
  → 22 repos buildable at HEAD
  → Static Analysis

- 48 repos with relevant commits
- 6 alarms in 5 repos with 5 bugs

**Difficulty**: Enormous variability in source code organization and “cleanliness”.

Like to work with Leidos in Phase 2 on this issue.
Search-and-repair platform for Android apps

Status: API **repairs** applicable “in the wild” today?

- **48** repos with relevant commits
- **6** alarms in **5** repos with **5** bugs

1. Github + repair specification → Relevant Commit Search
2. Relevant Commits + repair specification → Static Analysis
3. Relevant Commits + repair specification + patch locations → Abstract Syntactic Patch

32
Search-and-repair platform for Android apps

Status: API repairs applicable “in the wild” today?

- Github
- Relevant Commits
- Relevant Commit Search
- Static Analysis
- Abstract Syntactic Patch
- Patched Commits

48 repos with relevant commits
6 alarms in 5 repos with 5 bugs
22 repos buildable at HEAD
22 repos buildable at HEAD
32 repos with relevant commits
829 repos with relevant commits

5 repos patchable
Search-and-repair platform for Android apps

Status: API repairs applicable “in the wild” today?

- Github
- Relevant Commits
- Relevant Commit Search
- Static Analysis
- Abstract
- Patched

48 repos with relevant commits
6 alarms in 5 repos with 5 bugs

+ repair specification
22 repos buildable at HEAD

+ repair specification
+ patch locations

Compiles and loads without crash

32 repos with relevant commits
22 repos buildable at HEAD
5 repos patchable
Fixr Contributions

- Extract commit features at scale
- Find API usage patterns over time
- Index commit feature documents
- Search-and-repair platform for Android apps
Experience with the Corpus

We transitioned our scripts and corpus with Android Github repos to Leidos in April 2015

Need to have full git repos

Using corpus from a crawl in June 2015
Fixr Lessons Learned, Challenges, and Improvement Goals
Fixr Lessons Learned, Challenges, and Improvement Goals

Feature extraction time is compute intensive.

**Goal:** Incrementalize commit processing
Fixr Lessons Learned, Challenges, and Improvement Goals

Feature extraction time is compute intensive.  
**Goal:** Incrementalize commit processing

Manual interpretation of rules needed.  
**Goal:** Richer online tool to examine and analyze rules.  
File-by-file extraction of API associations.  
**Goal:** Richer association of local method context with API changes.
Fixr Lessons Learned, Challenges, and Improvement Goals

**Goal:** Incrementalize commit processing

Feature extraction time is compute intensive.

**Goal:** Richer online tool to examine and analyze rules.

Manual interpretation of rules needed.

**Goal:** Richer association of local method context with API changes.

File-by-file extraction of API associations.

Kinds of extracted features limits kinds of queries.

**Goal:** Investigate finer-grained features.
Fixr Lessons Learned, Challenges, and Improvement Goals

Feature extraction time is compute intensive.  
**Goal:** Incrementalize commit processing

Manual interpretation of rules needed.  
**Goal:** Richer online tool to examine and analyze rules.  
File-by-file extraction of API associations.  
**Goal:** Richer association of local method context with API changes.

Kinds of extracted features limits kinds of queries.  
**Goal:** Investigate finer-grained features.

Variance in source code organization limits buildability and thus analysis.  
**Goal:** Investigate normalized build systems.
Fixr Lessons Learned, Challenges, and Improvement Goals

**Feature extraction time is compute intensive.**

**Goal:** Incrementalize commit processing

**Manual interpretation of rules needed.**

**Goal:** Richer online tool to examine and analyze rules.
**File-by-file extraction of API associations.**

**Goal:** Richer association of local method context with API changes.

**Kinds of extracted features limits kinds of queries.**

**Goal:** Investigate finer-grained features.

---

Most difficulties anticipated but the magnitude of challenge with Big Data not necessarily expected
Overall status of the **Fixr** project

- **Deltar**: Inferring Semantic Deltas and Repair Specifications
- **Prepair**: Deriving Probabilistic Repair Specifications
- **Harvestr**: Social Validation and Mining of Fixes
- **Patchr**: Detecting Potential Bugs and Synthesizing Patches

Key terms:
- semantic
- statistical-semantic
- syntactic
- social
- interaction
- commit
- repair specification
- semantic facts
- probabilistic repair specification
- fix
- patch

**Github**
Overall status of the Fixr project

Deltar: Inferring Semantic Deltas and Repair Specifications

Prepair: Deriving Probabilistic Repair Specifications

Harvestr: Social Validation and Mining of Fixes

Patchr: Detecting Potential Bugs and Synthesizing Patches

Github

semantic
statistical-semantic
syntactic
social

interaction
commit

FixrDB

user-centered
big data analytics

fix
patch
repair specification
probabilistic repair specification
semantic facts
repair specification
Overall status of the \textbf{Fixr} project

\textbf{Github} interaction commit

\textbf{Deltar}: Inferring Semantic Deltas and Repair Specifications

\textbf{Prepair}: Deriving Probabilistic Repair Specifications

\textbf{FixrDB}

\textbf{Harvestr}: Social Validation and Mining of Fixes

\textbf{Patchr}: Detecting Potential Bugs and Synthesizing Patches

\textbf{FixrDB}

\textbf{symbolic program analysis}

semantic statistical-semantic syntactic social

user-centered big data analytics
Overall status of the **Fixr** project

**Deltar: Inferring Semantic Deltas and Repair Specifications**

**Prepair: Deriving Probabilistic Repair Specifications**

**Harvestr: Social Validation and Mining of Fixes**

**Patchr: Detecting Potential Bugs and Synthesizing Patches**

**Github**

**FixrDB**

symbolic program analysis

numerical-probabilistic program analysis

semantic
statistical-semantic
syntactic
social

interaction
commit

user-centered big data analytics

fix

repair specification

semantic facts

probabilistic repair specification

patch
Overall status of the Fixr project

**symbolic program analysis**
- Deltar: Inferring Semantic Deltas and Repair Specifications
- Prepair: Deriving Probabilistic Repair Specifications
- Harvestr: Social Validation and Mining of Fixes
- Patchr: Detecting Potential Bugs and Synthesizing Patches

**numerical-probabilistic program analysis**
- syntactic
- statistical-semantic
- semantic
- social

**interaction**
- commit

**FixrDB**
- repair specification
- semantic facts
- fix
- patch

**user-centered big data analytics**
- program synthesis
Overall status of the **Fixr** project

- **Deltar**: Inferring Semantic Deltas and Repair Specifications
- **Prepair**: Deriving Probabilistic Repair Specifications
- **Harvestr**: Social Validation and Mining of Fixes
- **Patchr**: Detecting Potential Bugs and Synthesizing Patches

**FixrDB**

- Symbolic program analysis
- Numerical-probabilistic program analysis
- Semantic
- Statistical-semantic
- Syntactic
- Social

**Interaction**

- Github

**Software engineering for big data**

- User-centered big data analytics
- Program synthesis
Overall status of the **Fixr** project

**Phase 1**

- **Deltar: Inferring Semantic Deltas and Repair Specifications**
- **Prepair: Deriving Probabilistic Repair Specifications**
- **Harvestr: Social Validation and Mining of Fixes**
- **Patchr: Detecting Potential Bugs and Synthesizing Patches**

**FixrDB**

- symbolic program analysis
- numerical-probabilistic program analysis
- semantic
- statistical-semantic
- syntactic
- social
- interaction
- commit

**software engineering for big data**

**user-centered big data analytics**

**big data analytics**
Overall status of the Fixr project

**Ongoing to Phase 2**

**Phase 1**

- **Github**
- **Harvestr**: Social Validation and Mining of Fixes
- **Deltar**: Inferring Semantic Deltas and Repair Specifications
- **Prepair**: Deriving Probabilistic Repair Specifications
- **Patchr**: Detecting Potential Bugs and Synthesizing Patches

**Keywords**
- Symbolic program analysis
- Semantic
- Statistical-semantic
- Syntactic
- Social

**Techniques**
- Program analysis
- Program synthesis
- User-centered big data analytics
- Software engineering for big data
- Big data analytics
- Numerical-probabilistic program analysis

**FixrDB**

**FixrDB**

- Interaction
- Commit

**Fixr** project

**FixrDB**

- Fix
- Repair specification
- Semantic facts
- Probabilistic repair specification
Overall status of the Fixr project

**Phase 1**

- **FixrDB:**
  - Syntactic
  - Statistical-semantic
  - Semantic

- **Deltar:** Inferring Semantic Deltas and Repair Specifications
  - Repair specification
  - Semantic facts

- **Prespair:** Deriving Probabilistic Repair Specifications
  - Probabilistic repair specification

- **Harvestr:** Social Validation and Mining of Fixes
  - Social
  - Interaction
  - Commit

- **Patchr:** Detecting Potential Bugs and Synthesizing Patches
  - Program synthesis

**Phase 2**

- **Github:**
  - Interaction

**Ongoing to Phase 2**

Semantic differencing for bug conditions
Overall status of the Fixr project

- **Phase 1**
  - FixrDB: Collecting and storing fixes for analysis
  - Harvestr: Social validation and mining of fixes
  - Deltar: Inferring semantic deltas and repair specifications
  - Prepair: Deriving probabilistic repair specifications
  - Patchr: Detecting potential bugs and synthesizing patches

- **Phase 2**
  - Semantic differencing for bug conditions
  - Spec mining: Callback orderings

- **Interaction**
  - Commit
  - Interaction

- **Tools**
  - Github
  - FixrDB
  - Deltar, Harvestr, Prepair, Patchr

- **Keywords**
  - Fixr
  - Repair
  - Specification
  - Probabilistic repair
  - Spec mining
  - Callback orderings
  - Semantic differences
  - Social validation
  - Big data analytics
  - Program synthesis
  - Software engineering for big data
  - User-centered big data analytics
  - Program analysis
  - Numerical-probabilistic program analysis
  - Symbolic reasoning
Overall status of the Fixr project

**Github**

**Phase 1**
- Ongoing to Phase 2

**Phase 2**

- **Deltar**: Inferring Semantic Deltas and Repair Specifications
- **Prepair**: Deriving Probabilistic Repair Specifications
- **Harvestr**: Social Validation and Mining of Fixes
- **Patchr**: Detecting Potential Bugs and Synthesizing Patches

**FixrDB**

- software engineering for big data
- user-centered big data analytics
- program synthesis

**semantic differencing for bug conditions**

**spec mining callback orderings**

**inferring app-specific event control-flow**

**interaction**

**commit**

**repair specification**

**probabilistic repair specification**

**numerical-probabilistic program analysis**
Overall status of the Fixr project

Deltar: Inferring Semantic Deltas and Repair Specifications

Prepair: Deriving Probabilistic Repair Specifications

Harvestr: Social Validation and Mining of Fixes

Patchr: Detecting Potential Bugs and Synthesizing Patches

FixrDB

Phase 1

Ongoing to Phase 2

semantic differencing for bug conditions

spec mining callback orderings

inferring app-specific event control-flow

spec mining fix transformations

software engineering for big data

user-centered big data analytics

program synthesis

Github

semantic
statistical-semantic
syntactic
social

interaction

commit
Overall status of the Fixr project

- **Github interaction**
- **Commit**
- **Fixr**
- **Patchr**
- **Harvestr**
- **Deltar**
- **Prepair**
- **FixrDB**
- **Phase 1**
- **Phase 2**

**Ongoing to Phase 2**

**semantic differencing for bug conditions**

**spec mining callback orderings**

**inferring app-specific event control-flow**

**spec mining fix transformations**

**Deltar: Inferring Semantic Deltas and Repair Specifications**

**Prepair: Deriving Probabilistic Repair Specifications**

**FixrDB**

**Further pattern mining (e.g., clustering)**

**Harvestr: Social Validation and Mining of Fixes**

**Patchr: Detecting Potential Bugs and Synthesizing Patches**

**FixrDB**

**user-centered big data analytics**

**program synthesis**

**interaction**

**commit**
Fixr Phase 1

Extract commit features at scale

Find API usage patterns over time

Index commit feature documents

Search-and-repair platform for Android apps
Fixr Phase 1

Extract commit features at scale
Find API usage patterns over time
Index commit feature documents
Search-and-repair platform for Android apps
**Fixr Phase 1**

1. Extract commit features at scale
2. Find API usage patterns over time
3. Index commit feature documents
4. Search-and-repair platform for Android apps