

# New Frontiers of Mining Software Repositories

## **Usability** and **Information Delivery**

Miryung Kim  
University of California, Los Angeles

**UCLA**

# Why software **specification inference**?



focus

cooperative and human aspects of SE.....

# What Makes APIs Hard to Learn? Answers from Developers

**Martin P. Robillard**, *McGill University*

A study of obstacles that professional Microsoft developers faced when learning to use APIs uncovered challenges and

**M**ost software projects reuse components exposed through APIs. In fact, current-day software development technologies are becoming inseparable from the large APIs they provide. To name two prominent examples, both the Java Software Development Kit and the .NET framework ship with APIs comprising thousands of classes supporting tasks that range from reading files to managing complex process workflows.

An API is the interface to implemented func- and interviewing developers about the obstacles

## Response categories for API learning obstacles

Main category	Subcategories/descriptions	Associated respondents	
Resources	Obstacles caused by inadequate or absent resources for learning the API (for example, documentation)	50	
	Examples	Insufficient or inadequate examples	20
	General	Unspecified issues with the documentation	14
	Content	A specific piece of content is missing or inadequately presented in the documentation (for example, information about all exceptions raised)	12
	Task	No reference on how to use the API to accomplish a specific task	9
	Format	Resources aren't available in the desired format	8
	Design	Insufficient or inadequate documentation on the high-level aspects of the API such as design or rationale	8
Structure	Obstacles related to the structure or design of the API	36	
	Design	Issues with the API's structural design	20
	Testing and debugging	Issues related to the API's testing, debugging, and runtime behavior	10
Background	Obstacles caused by the respondent's background and prior experience	17	
Technical environment	Obstacles caused by the technical environment in which the API is used (for example, heterogeneous system, hardware)	15	
Process	Obstacles related to process issues (for example, time, interruptions)	13	

What Makes APIs Hard to Learn? Answers from Developers, Robillard 2009

# Learning Barriers for Developers



# Six Learning Barriers in End-User Programming Systems

Andrew J. Ko, Brad A. Myers, and Htet Htet Aung  
*Human-Computer Interaction Institute*  
*Carnegie Mellon University, Pittsburgh, PA 15213 USA*  
*ajko@cmu.edu, bam+@cs.cmu.edu, hha@cs.cmu.edu*

## Abstract

*As programming skills increase in demand and utility, the learnability of end-user programming systems is of utmost importance. However, research on learning barriers in programming systems has primarily focused on languages, overlooking potential barriers in the environment and accompanying libraries. To address this, a study of beginning programmers learning Visual Basic.NET was*

## 2. Prior Research on Learning Barriers

One way to understand learning barriers is to study the learner. For example, imagine Jill, a user interface designer who just began learning VB. Shortly after starting, she realizes that she must learn about event handlers to proceed. This poses a potential learning barrier. From an attention-investment perspective [2], she will weigh the cost, risk, and reward of overcoming the barrier, and if the risk of failure outweighs the

# Six Learning Barriers for Developers

<b>Barrier Type</b>	
Design	I don't know what I want the computer to do...
Selection	I think I know what I want the computer to do but I don't know what to use
Coordination	I think I know what things to use but I don't know how to make them work together...
Use	I think I know what to use, but I don't know how to use it...
Understanding	I thought I knew how to use this but it didn't do what I expected...
Information	I think I know why it didn't do what I expected but I don't know how to check

# Six Learning Barriers for Developers

Barrier Type	
Design	I don't know what I want the computer to do...
Selection	I think I know what I want the computer to do but I don't know <b>what to use</b>
Coordination	I think I know what things to use but I don't know <b>how to make them work together...</b>
Use	I think I know what to use, but I don't know how to use it...
Understanding	I thought I knew how to use this but it didn't do what I expected...
Information	I think I know why it didn't do what I expected but I don't know how to check



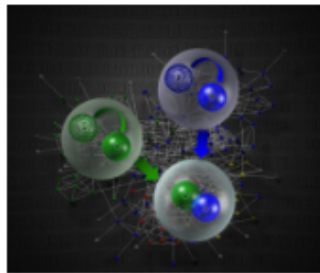
# Rise of Mining Big Code



May 5, 2014

## DARPA Launches 'Big Code' Initiative

George Leopold



The U.S. Defense Advanced Research Projects Agency is attempting to take big data analytics to the next level through a "big code" project designed to improve overall software reliability through a

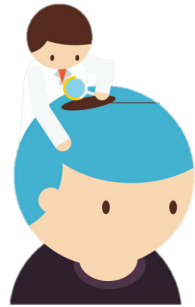
large-scale repository of software that drives big data.

The [DARPA "big code" initiative](#), formally known as Mining and Understanding Software Enclaves, or MUSE, seeks to leverage software analysis and big data analytics to improve the way software is built, debugged and verified.



Learning from "Big Code"

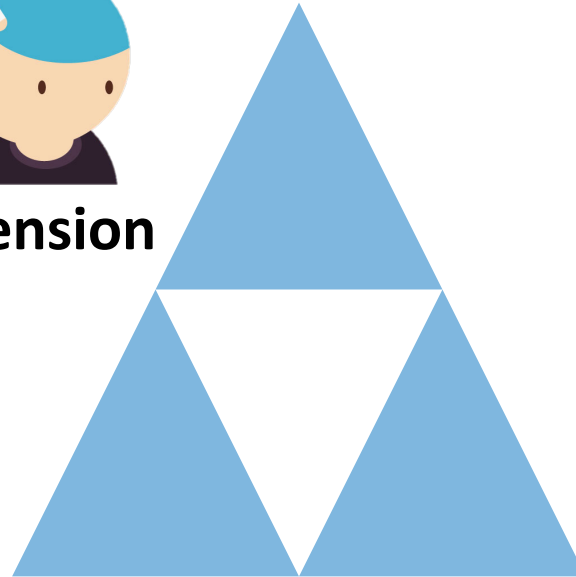
# Information Delivery and Usability



**1. Comprehension**

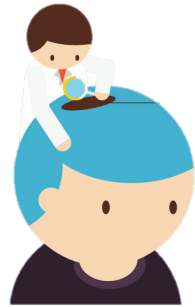


**2. Interactive Navigation**

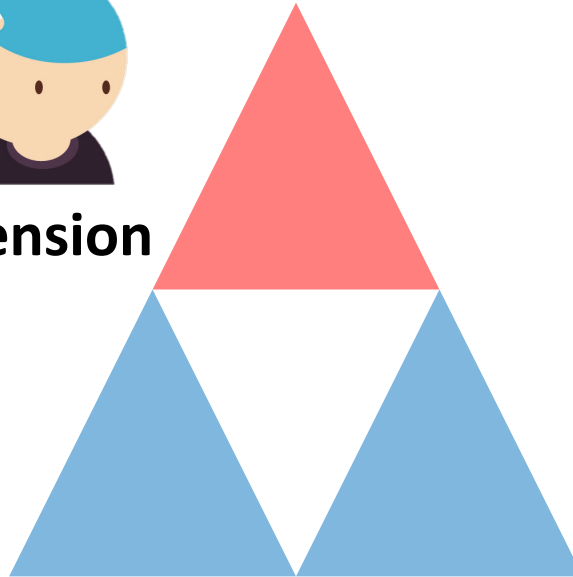


**3. Fit Developer Workflow**

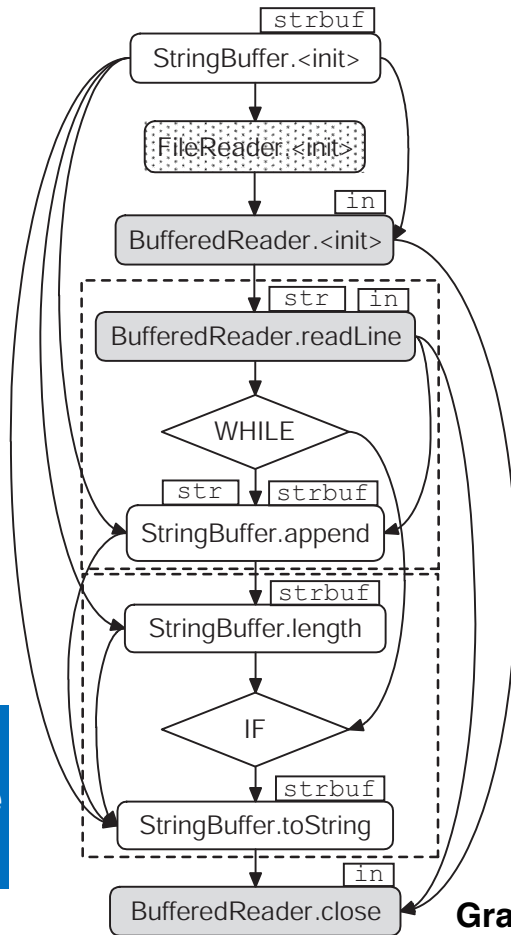
# Information Delivery and Usability



**1. Comprehension**

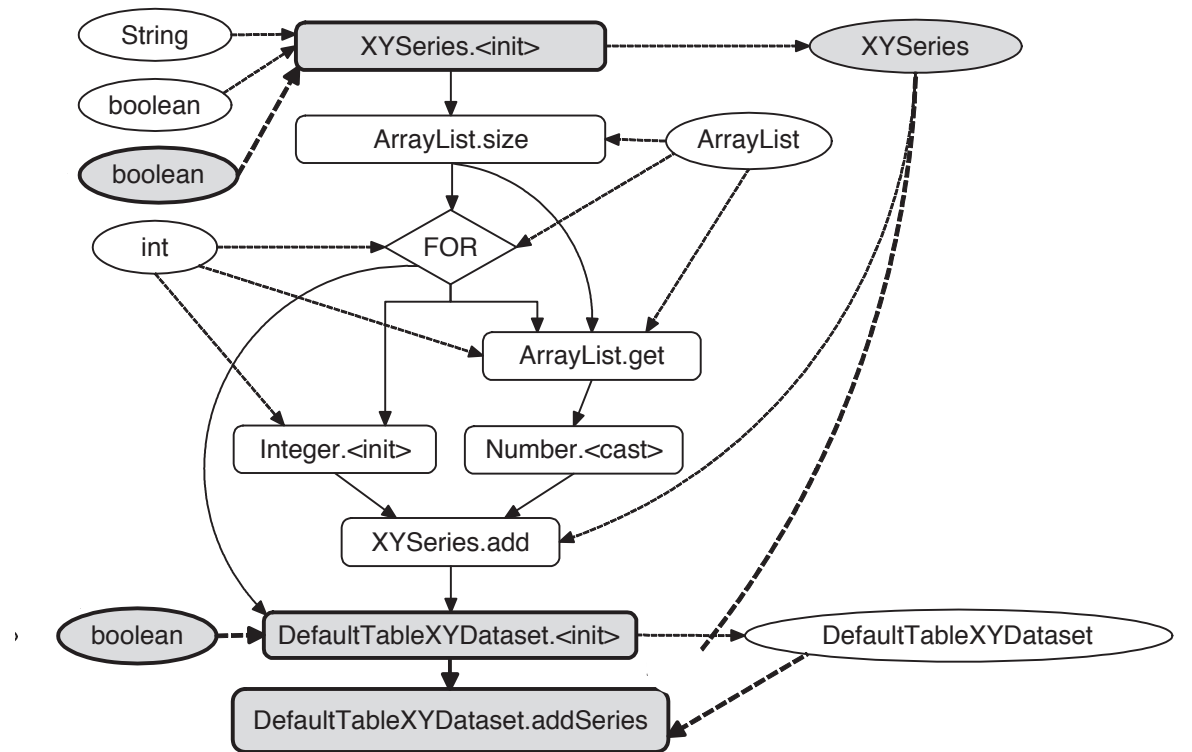


# Not Easy to Comprehend



Graph-based Mining of Multiple Object Usage Patterns, Nguyen et al. 2009

# Not Easy to Comprehend



# What Makes a Good Code Example?

- “It shouldn’t model something extremely **specific**”
- “It must be able to show **multiple uses**.”
- “a good example is easy to **understand** and read.”
- “**less irrelevant**, unrelated stuff in the example is better”
- “clear **naming** of variables”

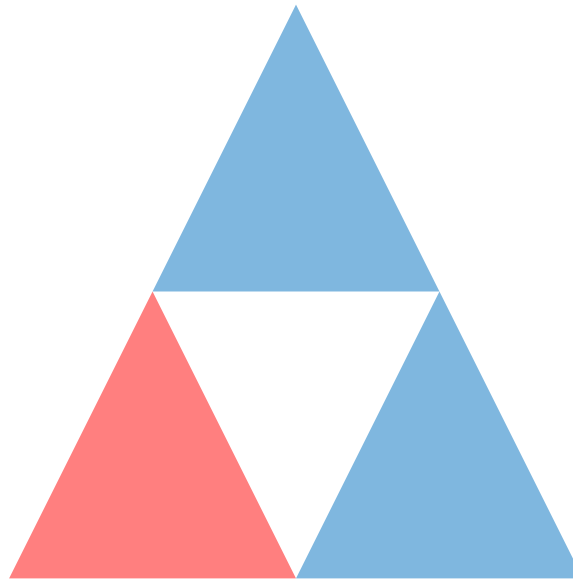
# What Makes a Good Code Example?

- “It shouldn’t model something extremely **specific**”
- “It must be able to show **multiple uses**.”
- “a good example is easy to **understand** and read.”
- “**less irrelevant**, unrelated stuff in the example is better”
- “clear **naming** of variables”

```
BufferedInputStream b;//initialized previously:  
ObjectInputStream stream =  
    new ObjectInputStream(b);  
try {  
    Object o = stream.readObject();  
    //Do something with o  
} catch(IOException e) {  
} finally {  
    stream.close();  
}
```

**Synthesized Code Example**

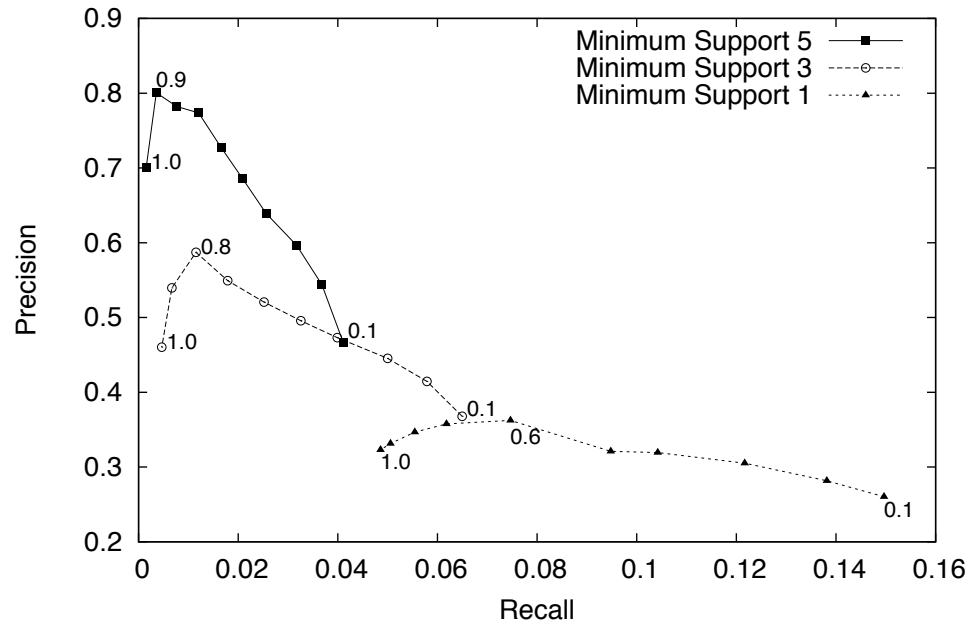
# Information Delivery and Usability



## 2. Interactive Navigation



# Precision and Recall are Not Enough



Setting	Precision (%)	Recall (%)	Rank
	Top 5	Top 5	
k = 1	80	91	3
k = 2	79	92	4
k = 3	80	91	4
k = ∞	74	91	4
No Slicing	65	81	9

Mining Version Histories to Guide Software Changes, Zimmermann et al. 2004

Are Code Examples on an Online Q&A Forum Reliable? Zhang et al. 2018

# Not Easy to Navigate and Compare

The screenshot displays a search interface for the API `SqlConnection.Open`. On the left, a list of 12 usage patterns is shown, with the top three having counts of 252, 232, and 100. The selected pattern (rank 54) includes `SqlConnection.Open`, `SqlConnection.BeginTransaction`, `SqlCommand.Parameters.Add`, and `SqlCommand.ExecuteReader`. The right panel shows a 'Code' tab with a flow graph. The graph starts with `SqlConnection.Open` (100% to `SqlConnection.BeginTransaction`), which leads to `SqlCommand.Parameters.Add` (100%). From there, the flow branches into three paths: `SqlCommand.ExecuteScalar` (7.78%), `SqlCommand.ExecuteNonQuery` (7.04%), and `SqlCommand.ExecuteReader` (90%). All three paths converge to `SqlTransaction.Commit` (100%).

1. Search box; 2. Usage pattern 1 for `SqlConnection.Open`: conducting a simple reading operation after opening a `Sql` connection; 3. Usage pattern 5: conducting a transaction operation after opening a `Sql` connection; 4. Sample code Tab: showed code snippets associated with the selected usage pattern shown in the left panel; 5. Graphical representation of usage pattern: aggregated view of API calls that appears before calling the searched API and API calls that appear after calling the searched API with probability of invocation relationship.

**API of interest**

**Clicked sequence**

**Contexts of methods**

**Pattern**

**Pattern rank**

**Whether the API is invoked in this sample**

className	methodName	Signature	Similarity	Invoked
1 ZenKitContextMenuProvider	buildContextMenu	void--IMenuManager	93.61%	Invoked
2 PetriNetContextMenuProvider	buildContextMenu	void--IMenuManager	86.66%	Invoked
3 FlowEditorContextMenuProvider	buildContextMenu	void--IMenuManager	81.11%	Invoked
4 EntityRelationContextMenuProvider	buildContextMenu	void--IMenuManager	92.22%	Invoked
5 SequenceChartContextMenuProvider	buildContextMenu	void--IMenuManager	92.22%	Invoked

**Click**

**Methods with the clicked sequence  
Highlighted with background color**

**Methods without the clicked sequence**

**Similarity to the current programming context**

```

Source code of selected method.
public void buildContextMenu( IMenuManager manager) {
    GEFActionConstants.addStandardActionGroups(manager);
    IAction action;
    action = getActionRegistry().getAction( ZenModelerConstants.ZEN_KIT_ZOOM_IN );
    if ( action.isEnabled() )
        manager.appendToGroup( GEFActionConstants.GROUP_EDIT, action);
}

```

MAPO: Mining and Recommending API Usage Patterns, Zhong et al. 2009

# Why Don't Software Developers Use Static Analysis Tools to Find Bugs?

Brittany Johnson, Yoonki Song, and Emerson Murphy-Hill  
North Carolina State University  
Raleigh, NC, U.S.A.  
bijohnso,y song2@ncsu.edu,emerson@csc.ncsu.edu

Robert Bowdidge  
Google  
Mountain View, CA, U.S.A.  
bowdidge@google.com

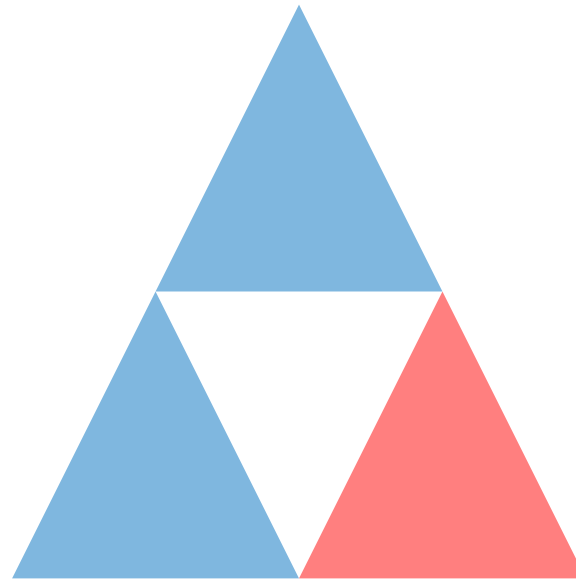
***Abstract***—Using static analysis tools for automating code inspections can be beneficial for software engineers. Such tools can make finding bugs, or software defects, faster and cheaper than manual inspections. Despite the benefits of using static analysis tools to find bugs, research suggests that these tools are underused. In this paper, we investigate why developers are not widely using static analysis tools and how current tools could potentially be improved. We conducted interviews with 20 developers and found that although all of our participants felt that use is beneficial, false positives and the way in which the warnings are presented, among other things, are barriers to use. We discuss several implications of these results, such as the need for an interactive mechanism to help developers fix defects.

There are many situations where a developer may consider using a static analysis tool to find defects in their code. Let us consider a developer, Susie. Susie is a software developer at a small company. She wants to make sure that she is following the company's standards while maintaining quality code. She needs a way of checking her code in her IDE, before submitting it to the general code repository, without worrying about any outside dependencies that she has no control over. Susie decides that her best bet is to install a static analysis tool. She decides to install FindBugs because she likes the quality of the results and the fact that bugs can be found as she types; at first, she is very happy with her decision and

# Disuse of Static Analysis Tools

- **Large volumes of false positives and warnings** outweigh true positives in volumes
- **Custom Navigation and Filter** Users would like to configure the ways that you see and filter results
- **Actionable Understandability** A developer not being able to understand what the tool is telling her is a barrier to use
  - **Quick Fixes** *“if you can tell me it’s an error, you should be able to tell me how to fix it.”*

# Information Delivery and Usability



## 3. Fit Developer Workflow

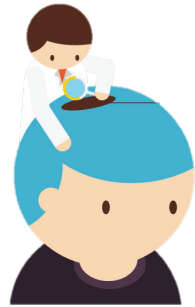
# Must Fit Developer Workflow

- Developers opportunistically interleave **web foraging of online resources**, learning, and writing code.
- Programmers search for code very frequently, conducting an average of 5 search sessions with **12 queries each workday**.
- 7% of respondents reused or modified code examples from **Stack Overflow** daily, 40% did at least weekly, and 62% did at least monthly.



Two Studies of Opportunistic Programming: Interleaving Web Foraging, Learning, and Writing Code, Brandt et al. 2009  
How Developers Search for Code: A Case Study, Sadowski et al. 2015  
How do developers utilize source code from stack overflow? Wu et al. 2018

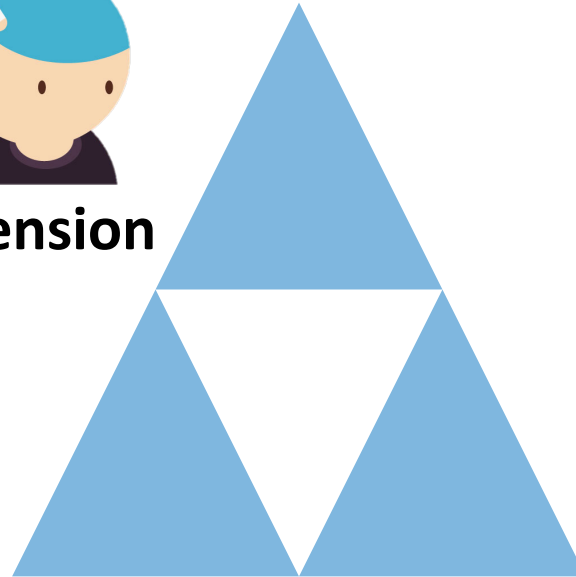
# Information Delivery and Usability



**1. Comprehension**

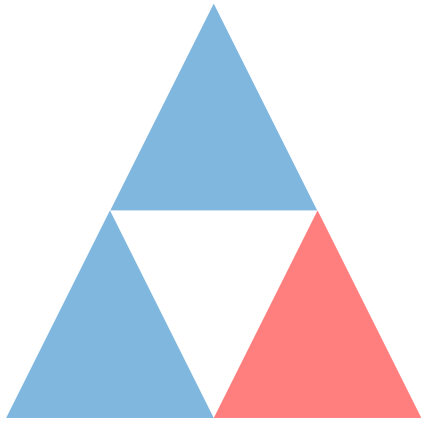


**2. Interactive Navigation**



**3. Fit Developer Workflow**





**Fit Developer Workflow**

## Part 1

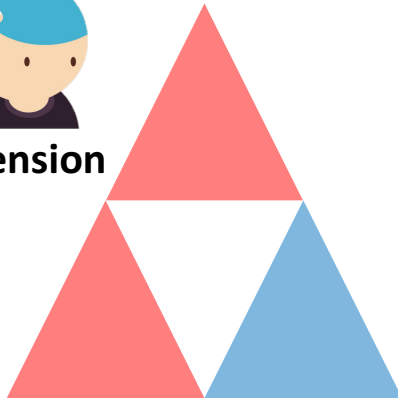
API Usage Mining from GitHub  
API Misuse Detection in Stack  
Overflow [ICSE 2018]



**Comprehension**



**Interactive Navigation**



## Part 2

Visualization of Code Examples at  
Scale [CHI 2018]

# “How do I write data to a file using FileChannel?”



2



You're probably interested in a `FileChannel`. `Channel`s were designed to perform bulk IO operations to and from `Buffer`s.

Ex:

```
FileChannel fileOut = new FileOutputStream(file).getChannel();
fileOut.write(ByteBuffer.wrap("Whatever you want to write".getBytes()));
```

share improve this answer

answered Apr 8 '12 at 19:39



Jeffrey

35.7k ● 7 ● 60 ● 111

---

Actually, I want to maintain a large buffer (whose size I can mention) and periodically flush it. – [Arpssss](#) Apr 8 '12 at 19:43

---

@Arpssss You can maintain a `ByteBuffer` and periodically write it to the file system. You don't have to create your `ByteBuffer` inline like that. – [Jeffrey](#) Apr 8 '12 at 19:47

---

Thanks. Is it possible to use charbuffer of NIO ? – [Arpssss](#) Apr 8 '12 at 20:05

---

`FileChannel` cannot write a `CharBuffer`. You can, however, use `ByteBuffer#putChar` to put characters

# “How do I write data to a file using FileChannel?”

▲ You're probably interested in a `FileChannel`. `Channel`s were designed to perform bulk IO operations to and from `Buffer`s.

2

▼ Ex:

✓  

```
FileChannel fileOut = new FileOutputStream(file).getChannel();
fileOut.write(ByteBuffer.wrap("Whatever you want to write".getBytes()));
```


**This example forgets to close the FileChannel object properly.**

 35.7k ● 7 ● 60 ● 111

---

Actually, I want to maintain a large buffer (whose size I can mention) and periodically flush it. – [Arpssss](#) Apr 8 '12 at 19:43

---

@Arpssss You can maintain a `ByteBuffer` and periodically write it to the file system. You don't have to create your `ByteBuffer` inline like that. – [Jeffrey](#) Apr 8 '12 at 19:47 

---

Thanks. Is it possible to use charbuffer of NIO ? – [Arpssss](#) Apr 8 '12 at 20:05

---

`FileChannel` cannot write a `CharBuffer`. You can, however, use `ByteBuffer#putChar` to put characters into it. – [Jeffrey](#) Apr 8 '12 at 20:08

## *“How do I write data to a file using FileChannel?”*

▲ Somewhat like this:

7

▼

```
short[] payload = {1,2,3,4,5,6,7,8,9,0};
ByteBuffer myByteBuffer = ByteBuffer.allocate(20);
myByteBuffer.order(ByteOrder.LITTLE_ENDIAN);

ShortBuffer myShortBuffer = myByteBuffer.asShortBuffer();
myShortBuffer.put(payload);

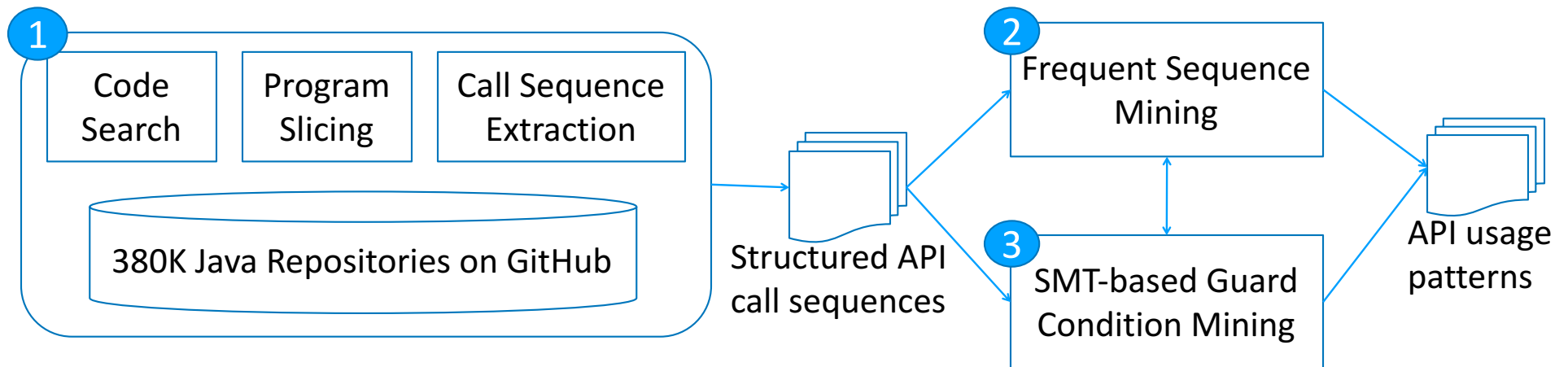
FileChannel out = new FileOutputStream("sample.bin").getChannel();
out.write(myByteBuffer);
out.close();
```

✓

**This example forgets to handle potential exceptions such as IOException and FileNotFoundException.**

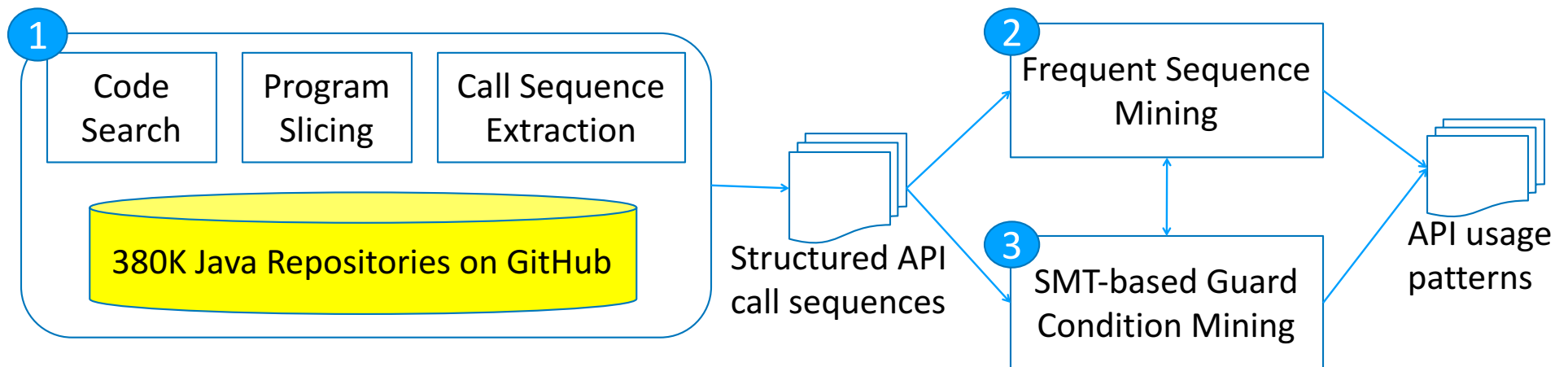
# API Usage Mining from GitHub

- We mine API usage patterns from 380K GitHub projects.



# Insight 1: Mining a Large Code Corpus

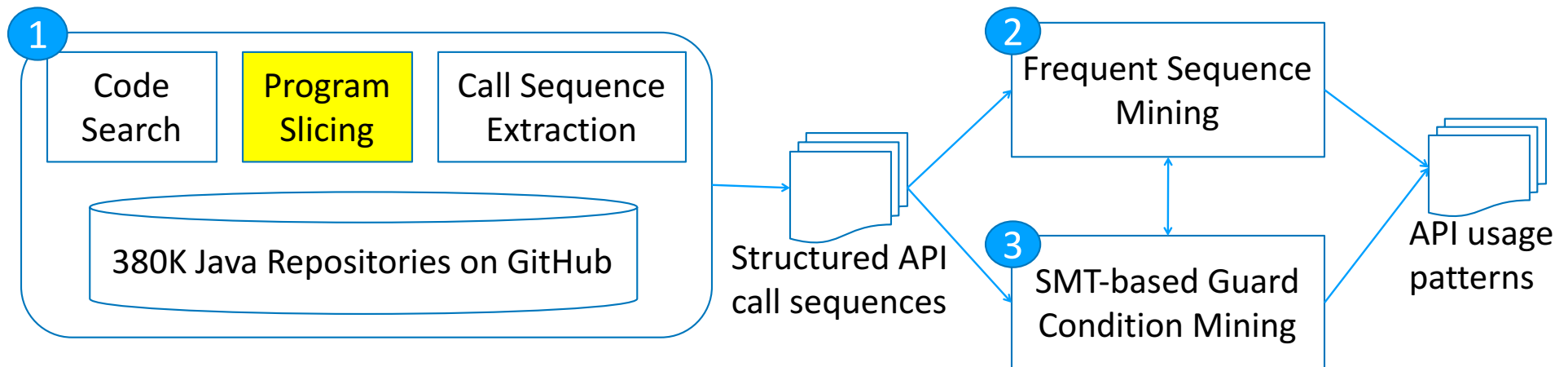
- Our code corpus includes 380K GitHub projects with at least 100 revisions and 2 contributors.



Dyer et al. Boa: A language and infrastructure for analyzing ultra-large-scale software repositories. ICSE 2013.

# Insight 2: Removing Irrelevant Statements via Program Slicing

- We perform backward and forward slicing to identify data- and control-dependent statements to an API method of interest.



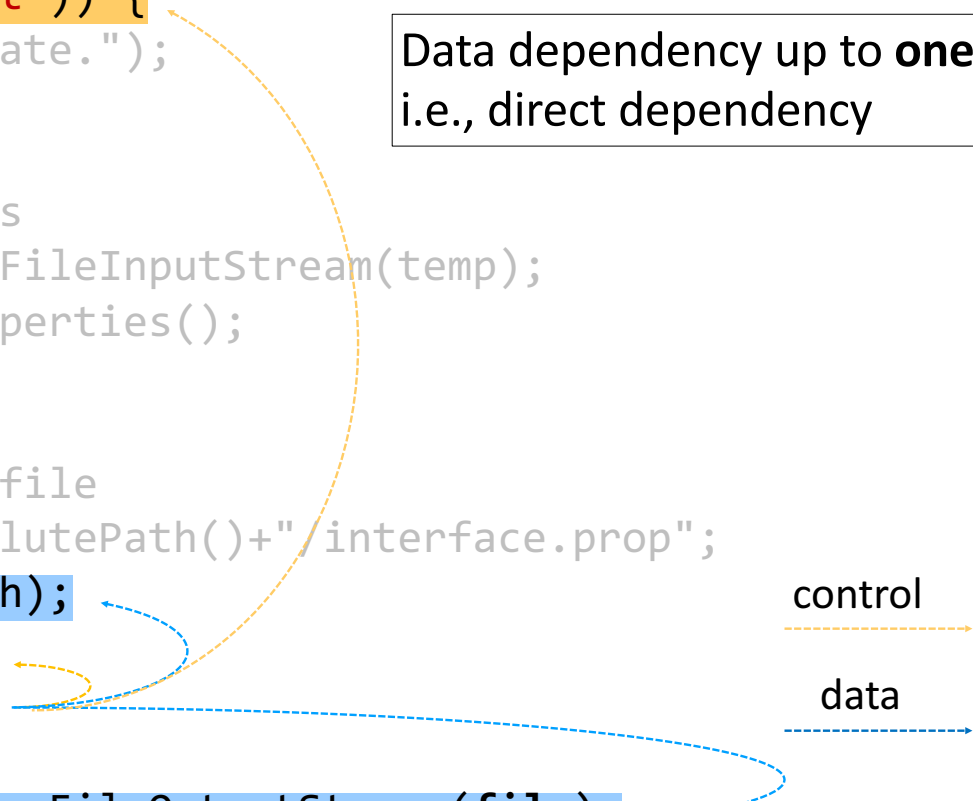
```
void initInterfaceProperties(String temp, File dDir) {  
    if(!temp.equals("props.txt")) {  
        log.error("Wrong Template.");  
        return;  
    }  
    // load default properties  
    FileInputStream in = new FileInputStream(temp);  
    Properties prop = new Properties();  
    prop.load(in);  
    ... init properties ...  
    // write to the property file  
    String fPath=dDir.getAbsolutePath()+"/interface.prop";  
    File file = new File(fPath);  
    if(!file.exists()) {  
        file.createNewFile();  
    }  
    FileOutputStream out = new FileOutputStream(file);  
    prop.store(out, null);  
    in.close();  
}
```

Data dependency up to **one** hop,  
i.e., direct dependency

The focal API  
method

control

data





```
void initInterfaceProperties(String temp, File dDir) {  
    if(!temp.equals("props.txt")) {  
        log.error("Wrong Template.");  
        return;  
    }  
    // load default properties  
    FileInputStream in = new FileInputStream(temp);  
    Properties prop = new Properties();  
    prop.load(in);  
    ... init properties ...  
    // write to the property file  
    String fPath=dDir.getAbsolutePath()+"/interface.prop";  
    File file = new File(fPath);  
    if(!file.exists()) {  
        file.createNewFile();  
    }  
    FileOutputStream out = new FileOutputStream(file);  
    prop.store(out, null);  
    in.close();  
}
```

Data dependency up to two hops

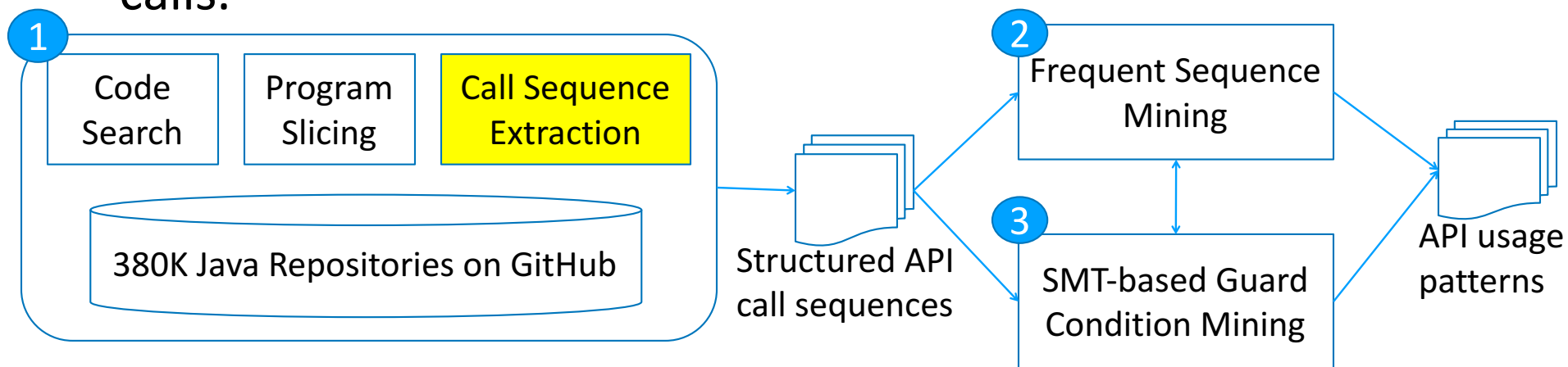
The focal API method

control

data

# Insight 3: Capture the Semantics of API Usage

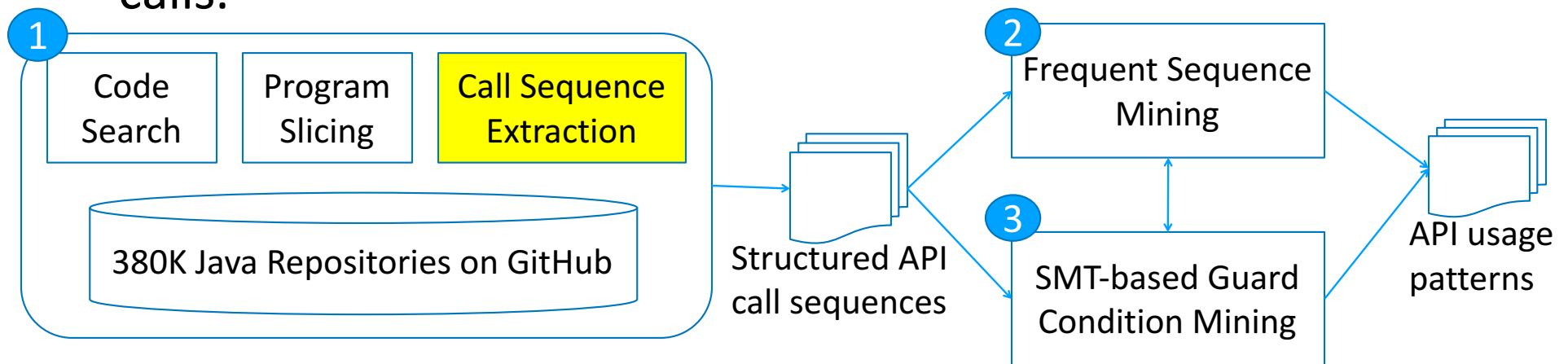
- It is important to capture the temporal ordering, enclosing control structures, and appropriate guard conditions of API calls.



```
new File (String); try { new FileInputStream(File)@arg0.exists(); }  
catch (IOException); }
```

# Insight 3: Capture the Semantics of API Usage

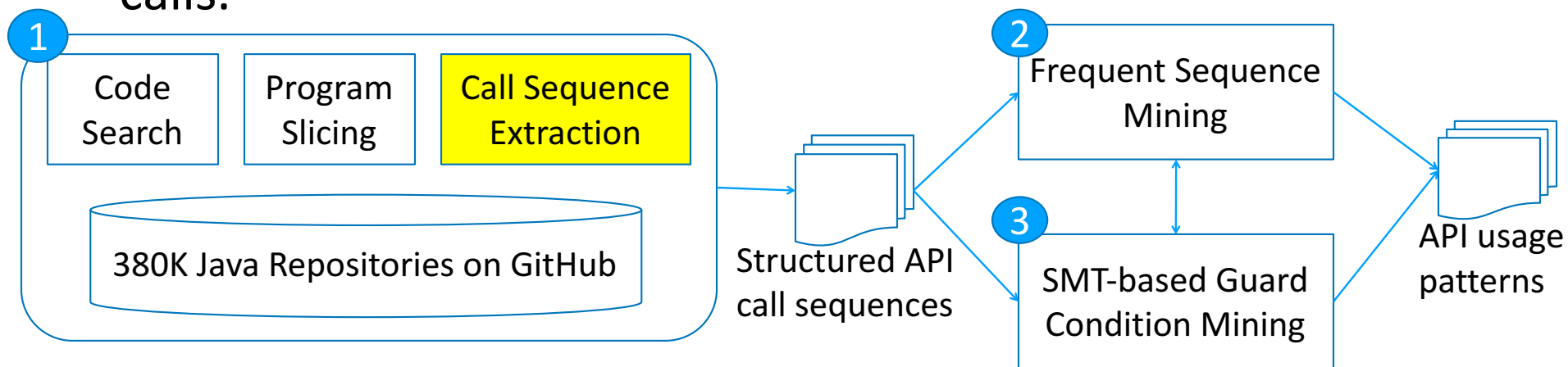
- It is important to capture the temporal ordering, enclosing control structures, and appropriate guard conditions of API calls.



```
new File (String); try { new FileInputStream(File)@arg0.exists(); }  
catch (IOException); }
```

# Insight 3: Capture the Semantics of API Usage

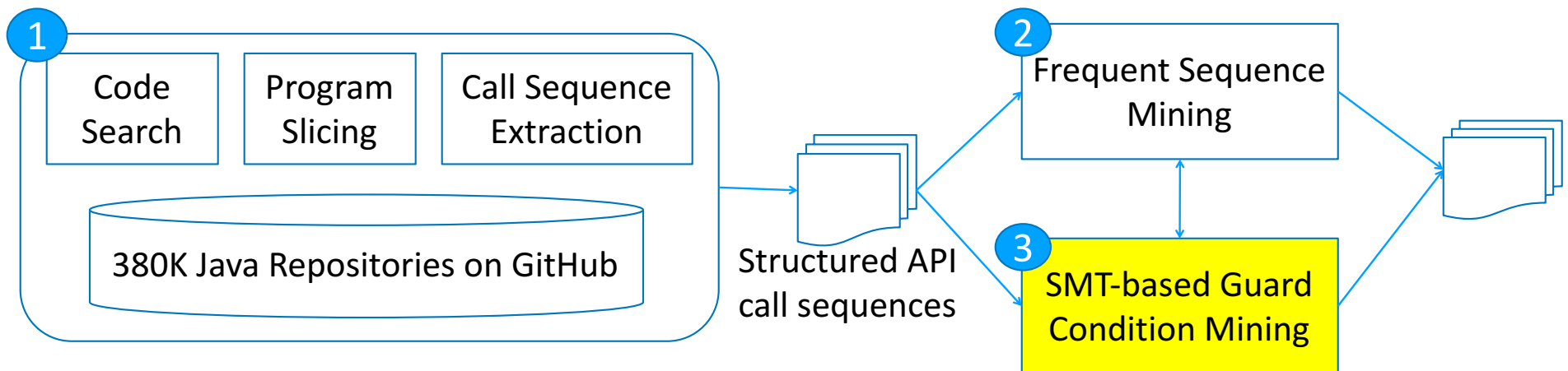
- It is important to capture the temporal ordering, enclosing control structures, and appropriate guard conditions of API calls.



```
new File (String); try { new FileInputStream(File)@arg0.exists(); }  
catch (IOException); }
```

# Insight 4: Variations in Guard Conditions

- GitHub developers may write the same predicate in different ways.



**Two equivalent guard conditions for substring(int):**

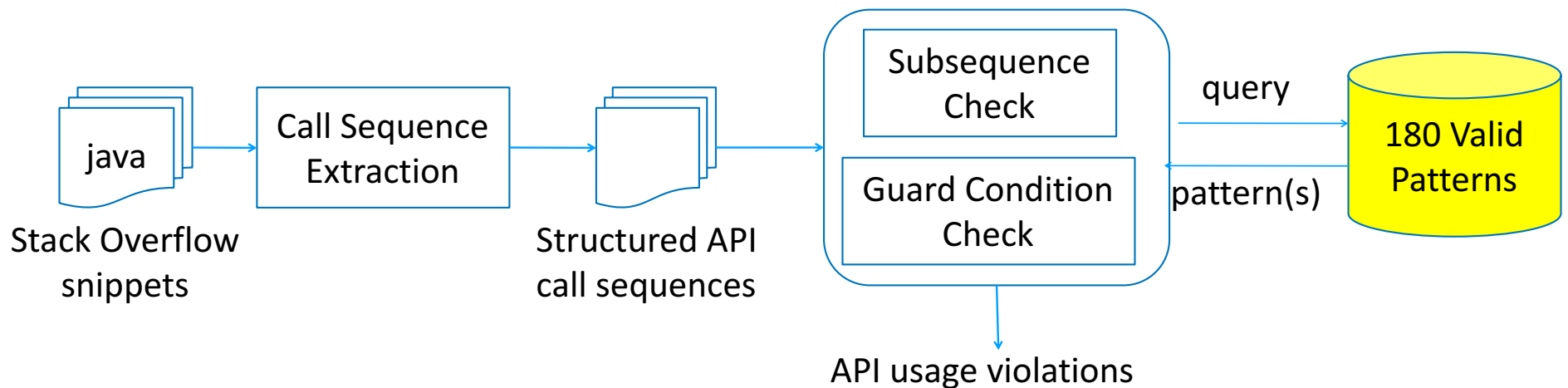
**$\text{arg0} \geq 0 \ \&\& \ \text{arg0} \leq \text{rcv.length}() \Leftrightarrow \text{arg0} > -1 \ \&\& \ \text{arg0} < \text{rcv.length}() + 1$**

# Yet Another API Usage Mining Tool?



# API Misuse Detection in Stack Overflow

- We examine 220K SO posts with 180 confirmed patterns.
- => 31% of SO posts contain API usage violations!



Dataset: <http://web.cs.ucla.edu/~tianyi.zhang/examplecheck.html>





# ExampleCheck [FSE'18 Demo]

The screenshot shows a Stack Overflow page with a question and an answer. A pop-up window is overlaid on the right side of the page, highlighting specific features of the 'ExampleCheck' tool. The pop-up window contains the following elements:

- Potential API Misuse**: A red circle with the number 1 points to the title 'Potential API Misuse'.
- API misuse description**: A red circle with the number 2 points to the text 'You may want to check whether the receiver of `getAsString()` is not equal to null. You may also want to handle the potential Exception thrown by `getAsString()` by using a try-catch block here. 117 Github code examples also do this.'
- Fix suggestion**: A red circle with the number 3 points to the code snippet:

```
try {
    if (match_number!=null) {
        match_number.getAsString();
    }
} catch (Exception e) {
```
- Like or dislike**: A red circle with the number 4 points to the thumbs up and thumbs down icons, which show 4 likes and 2 dislikes.
- Supporting GitHub examples**: A red circle with the number 5 points to the text 'See this in a GitHub example:' followed by links to 'ybonnel/gson', 'Aleks-Ya/hh-java-api', and 'ezterry/TTRSS\_android\_ezterry'.
- Pagination for multiple misuses**: A red circle with the number 6 points to the pagination controls at the bottom of the pop-up window, which show buttons for 1, 2, 3, and 4 pages.

The background Stack Overflow page shows a question with 0 votes and an answer with 1 vote. The answer contains a code snippet for parsing JSON and a comment: 'just to note, i just tested it and it works fine. - faljbou...'. The Chrome browser logo and 'chrome Your Answer' are visible in the bottom left corner.



# ExampleCheck

Remove from Chrome

Offered by: Tianyi Zhang

★★★★★ 0 | [Developer Tools](#) | 26 users

Overview

Reviews

Related

stackoverflow Questions Developer Jobs

try this out, I did not test it, but from what i see in your code, alliances is not an a json array, also objects based on what i see in your json docume

0

```

JsonObject rootobj = root.getAsJsonObject();
JsonElement match_number = rootobj.get("match");
JsonObject alliances = rootobj.getAsJsonObject("alliances");
JsonElement blue = alliances.getAsJsonObject("blue");
JsonElement red = alliances.getAsJsonObject("red");
System.out.println(match_number.getAsString());

```

share improve this answer edited At

just to note, i just tested it and it works fine. - faljbour

add a comment

**Potential API Misuse**

You may want to check whether the receiver of `getAsString()` is not equal to null. You may also want to handle the potential Exception thrown by `getAsString()` by using a try-catch block here. 117 Github code examples also do this.

```

try {
    if (match_number!=null) {
        match_number.getAsString();
    }
} catch (Exception e) {
}

```

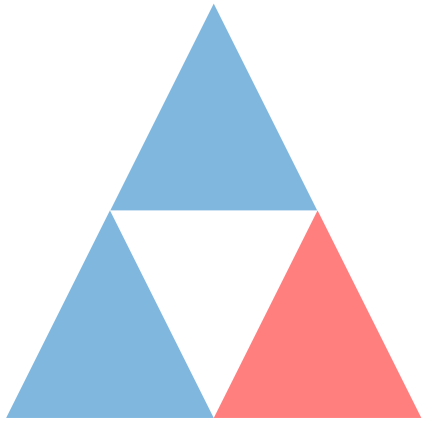
See this in a GitHub example:  
[ybonnel/gson](#)  
[Aleks-Ya/hh-java-api](#)  
[ezterry/TTRSS\\_android\\_ezterry](#)

1 2 3 4



# chrome





**Fit Developer Workflow**

## Part 1

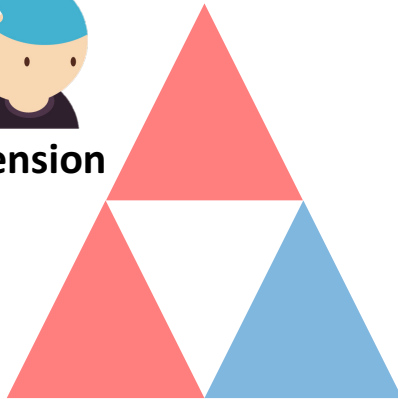
API Usage Mining from GitHub  
API Misuse Detection in Stack  
Overflow [ICSE 2018]



**Comprehension**



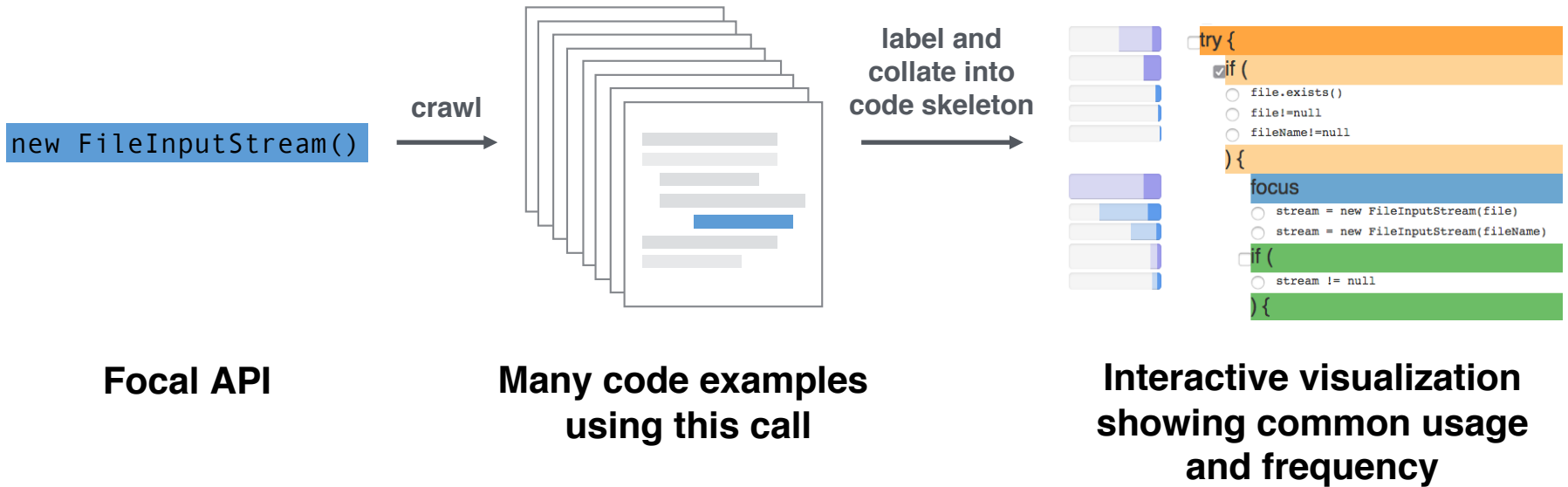
**Interactive Navigation**



## Part 2

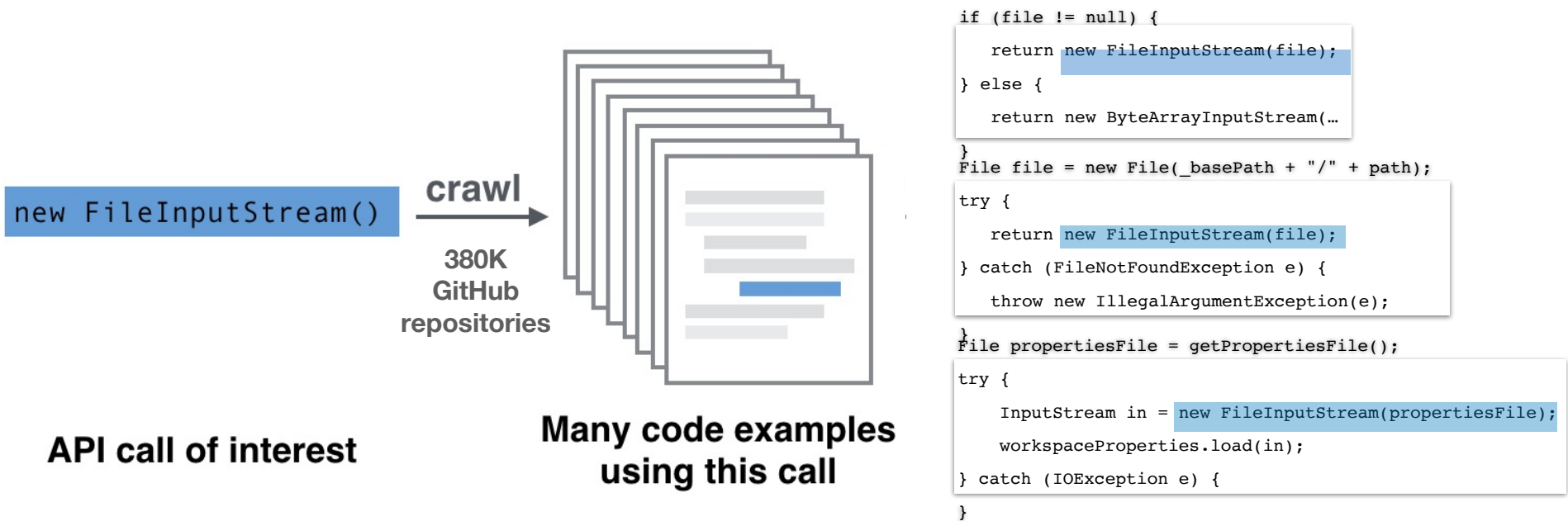
Visualization of Code Examples at  
Scale [CHI 2018]

# Example: Visualizing Examples at Scale



Demo: <http://example.cs.ucla.edu:3000>

# Mining API Usage from a Large Corpus



# Program Slicing and Labeling

```
private void getLatestVersion() {
// TODO Auto-generated method stub
File temp = new File(Environment.getExternalStorageDirectory().toString() + "/pdTemp");
try {
List<File> listMain = IoUtils.extractZipResource(new FileInputStream(pdZipPath), temp, true);
if (listMain.size() != 0) {
for (File f : listMain) {
if (f.isDirectory()) folderName = f.getName();
if (f.getAbsolutePath().toLowerCase().contains("droidparty_main.pd")) {
foundmainPd = true;
dpMainfileName = f.getName();
InputStream is = new FileInputStream(f);
BufferedReader reader = new BufferedReader(new InputStreamReader(is));
String line;
while ((line = reader.readLine()) != null) {
String version;
if (line.contains(" version: ")) {
Log.d("LatestVersionLine", line);
version = line.substring(line.lastIndexOf(":") + 1, line.length() - 1);
this.latestVersion = Float.parseFloat(version);
break;
} else {
version = "0";
this.latestVersion = Float.parseFloat(version);
}
}
reader.close();
Log.d("LatestVersion", latestVersion + "");
break;
}
}
}
if (!foundmainPd) {
closePd();
}
} else {
closePd();
}
} catch (Exception e) {
e.printStackTrace();
}
}
```

```
private void getLatestVersion() {
// TODO Auto-generated method stub
File temp = new File(Environment.getExternalStorageDirectory().toString() + "/pdTemp");
try {
List<File> listMain = IoUtils.extractZipResource(new FileInputStream(pdZipPath), temp, true);
if (listMain.size() != 0) {
for (File f : listMain) {
if (f.isDirectory()) folderName = f.getName();
if (f.getAbsolutePath().toLowerCase().contains("droidparty_main.pd")) {
foundmainPd = true;
dpMainfileName = f.getName();
InputStream is = new FileInputStream(f);
BufferedReader reader = new BufferedReader(new InputStreamReader(is));
String line;
while ((line = reader.readLine()) != null) {
String version;
if (line.contains(" version: ")) {
Log.d("LatestVersionLine", line);
version = line.substring(line.lastIndexOf(":") + 1, line.length() - 1);
this.latestVersion = Float.parseFloat(version);
break;
} else {
version = "0";
this.latestVersion = Float.parseFloat(version);
}
}
}
reader.close();
Log.d("LatestVersion", latestVersion + "");
break;
}
}
}
if (!foundmainPd) {
closePd();
}
} else {
closePd();
}
} catch (Exception e) {
e.printStackTrace();
}
}
```

Labeled Code Examples

[Ko et al. 2004, Duala-Ekoko & Robillard 2012]

# Code Canonicalization

`new FileInputStream()`

crawl  
380K  
Github  
repositories



API call of interest

Many code examples  
using this call

## API Skeleton

```
declarations  
try {  
  pre method call  
  if ( ... ) {  
    focus  
    if ( ... ) {  
      post method call  
    }  
  }  
} catch ( ... ) {  
  exception handling call  
}  
finally { ...
```

## Labeled Code Examples

```
if (file != null) {  
  return new FileInputStream(file);  
} else {  
  return new ByteArrayInputStream(...  
}
```

```
File file = new File( String );  
try {  
  return new FileInputStream(file);  
} catch (FileNotFoundException e) {  
  throw new IllegalArgumentException(e);  
}
```

```
File file = getPropertiesFile();  
try {  
  InputStream stream = new FileInputStream(file);  
  workspaceProperties.load(stream);  
} catch (IOException e) {  
}
```

# Examplore Interface

Mutual Alignment

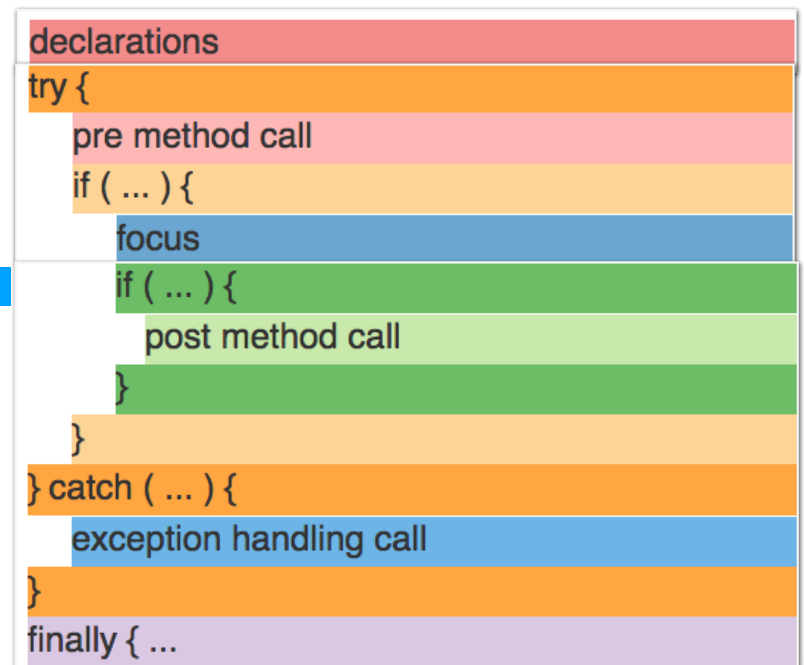
Abstraction  
API Skeleton

```
return new FileInputStream(file);
```

```
return new FileInputStream(file);
```

```
InputStream stream = new FileInputStream(file);
```

3





# Exampler Interface

Mutual Alignment

Abstraction  
API Skeleton

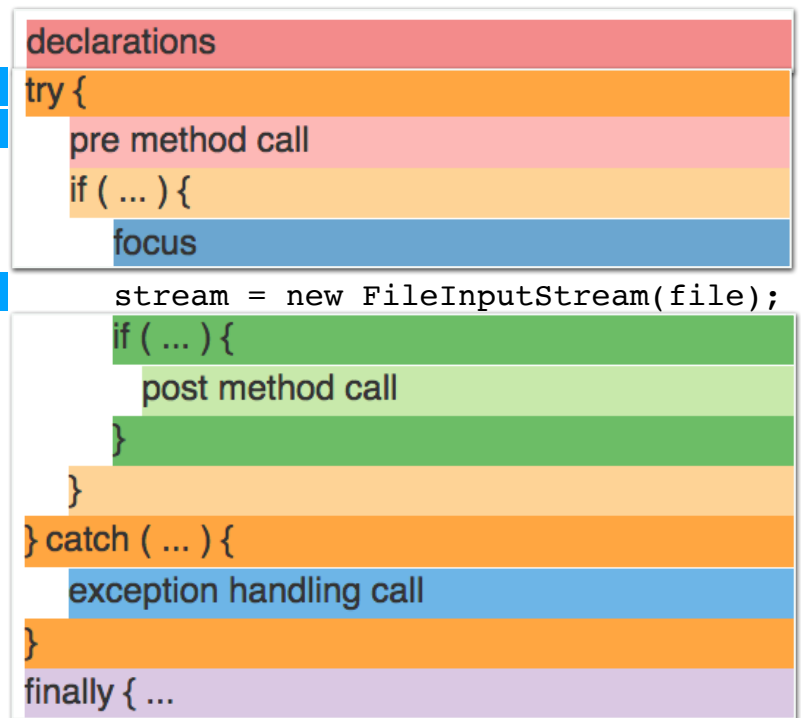
```
File file = new File(String);
```

```
File file = getPropertiesFile();
```

1

1

3



# Examplore Interface

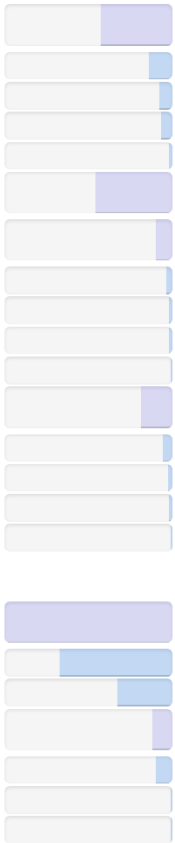
Mutual Alignment

Abstraction  
API Skeleton

```
if (file != null) {  
    return new FileInputStream(file);  
} else {  
    return new ByteArrayInputStream(...  
}
```

```
File file = new File(String);  
try {  
    return new FileInputStream(file);  
} catch (FileNotFoundException e) {  
    throw new IllegalArgumentException(e);  
}
```

```
File file = getPropertiesFile();  
try {  
    InputStream stream = new FileInputStream(file);  
    workspaceProperties.load(stream);  
} catch (IOException e) {  
}
```



- declarations
  - File file = new File(String)
  - File file = new File(\*)
  - File file = new File(\*,String)
  - String fileName = Properties.getProperty(String)
- try {
  - pre method call
    - file.length()
    - file.getName()
    - file.getAbsolutePath()
    - file.deleteOnExit()
  - if (
    - file.exists()
    - file!=null
    - fileName!=null
    - !(file.isDirectory()) && !(visited.contains(file,))
- ) {
  - focus
    - stream = new FileInputStream(file)
    - stream = new FileInputStream(fileName)
  - if (
    - stream != null
    - null != stream
    - stream.read(outputByte,0,4096) != -1

# Examplore Interface

## Cross-example counts for FileInputStream

Toggle Labels Fold Code Show Default Show Less Show More Show All

Counts

Blocks of options

- declarations
  - File file = new File(String)
  - File file = new File(\*)
  - File file = new File(\*,String)
- try {
  - pre method call
    - file.length()
    - file.getName()
    - file.getAbsolutePath()
  - if (
    - file.exists()
    - file!=null
    - fileName!=null
  - focus
    - stream = new FileInputStream(file)
    - stream = new FileInputStream(fileName)
  - if (
    - stream != null
    - null != stream
    - stream.read(outputByte,0,4096) != -1

## 100 concrete examples from GitHub

Reset Active Filters:

[Link to the GitHub source code](#)

```
@Override
public void readFromFile(String filename) throws IOException {
    in = new FileInputStream(filename);
    prop.load(in);
}
```

[Link to the GitHub source code](#)

```
private synchronized InputStream openStream() throws IOException {
    if (file != null) {
        return new FileInputStream(file);
    } else {
        return new ByteArrayInputStream(memory.getBuffer(), 0, memory.getCount());
    }
}
```

[Link to the GitHub source code](#)

```
public InputStream getResourceContents(String path) {
    File file = new File(_basePath + "/" + path);
    try {
        return new FileInputStream(file);
    } catch (FileNotFoundException e) {
        throw new IllegalArgumentException(e);
    }
}
```

[Link to the GitHub source code](#)

```
public InputStream getInputStream() throws MessaainaException {
```

# Examplore Interface

The interface includes a top navigation bar with buttons: Toggle Labels, Fold Code, Show Default, Show Less, Show More, Show All. On the left, there is a 'Counts' section with a vertical bar chart. Below it, 'Blocks of options' are listed with checkboxes and radio buttons. The 'declarations' block is highlighted in red. The 'try {' block is highlighted in orange, and the 'if (' block is checked with a red arrow. The 'focus' block is highlighted in blue, and the 'stream = new FileInputStream(file)' line is selected with a red arrow. The 'if (' block below it is highlighted in green. On the right, a 'Reset' button is next to a red-bordered box containing 'Active Filters: must have stream = new FileInputStream(file); some guard condition,'. Below this are two code snippets with red arrows pointing to specific lines. The first snippet is for 'private synchronized InputStream openStream()' and the second is for 'public InputStream getInputStream()'. Both snippets have red arrows pointing to the 'if' conditions and the 'return new FileInputStream()' lines.

Toggle Labels Fold Code Show Default Show Less Show More Show All

Counts

Blocks of options

- declarations
- File file = new File(String)
- File file = new File(\*)
- try {
- if (
- file.exists()
- file!=null
- ) {
- focus
- stream = new FileInputStream(file)
- if (
- stream != null
- ) {

Reset Active Filters: must have stream = new FileInputStream(file); some guard condition,

Link to the GitHub source code

```
private synchronized InputStream openStream() throws IOException {  
    if (file != null) {  
        return new FileInputStream(file);  
    } else {  
        return new ByteArrayInputStream(memory.getBuffer(), 0, memory.getCount());  
    }  
}
```

Link to the GitHub source code

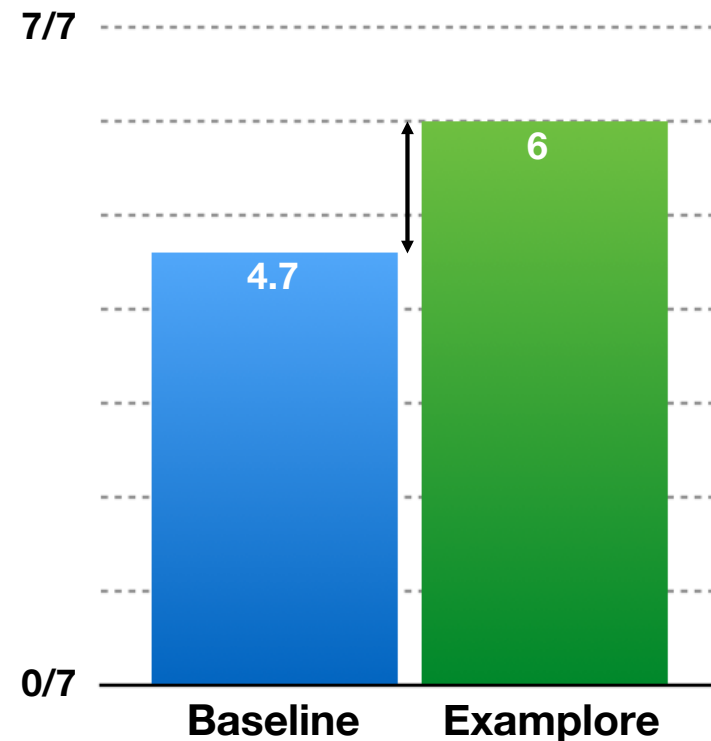
```
public InputStream getInputStream() {  
    if (exists()) {  
        if (_file.isFile()) {  
            try {  
                return new FileInputStream(_file);  
            } catch (FileNotFoundException e) {  
                throw new RuntimeException(e);  
            }  
        } else if (_file.isDirectory()) {
```

Tool is available at <http://examplore.cs.ucla.edu:3000/>

# Lab Study Results

- Examplore users investigated many relevant examples.
- Baseline users often answered based on one example or by guessing.

Average # of correct answers on **Q1-7**



Mean difference is statistically significant  
(paired t-test:  $t=3.02$ ,  $df=15$ ,  $p\text{-value}<0.01$ )

# Lab Study Results

For **Q8**, 88% of participants gave valid comments about the StackOverflow answer.

The majority of participants' critiques...

- (Using the baseline) were about style and the mechanics of adaptation
- (Using Examplore) were about safety


- Q8. How might you **modify this code example on Stack Overflow** if you were going to copy and paste it into your own solution to the original prompt?

▲ This function is very useful to read a whole file into memory. See this example,

```
1 File = new File("/anywhere/anyfile");
  InputStream is = new FileInputStream(file);
  long fileSize = file.length();
  byte[] bytes = new byte[(int)fileSize];
  int offset = 0;
  int count=0;
  while (offset < fileSize) {
    count=is.read(bytes, offset, fileSize-offset);
    if (count >= 0)
      offset += count;
    else
      throw new IOException("Can't read file "+file.getName());
  }
  is.close();
  // Now bytes has all the complete file.
```

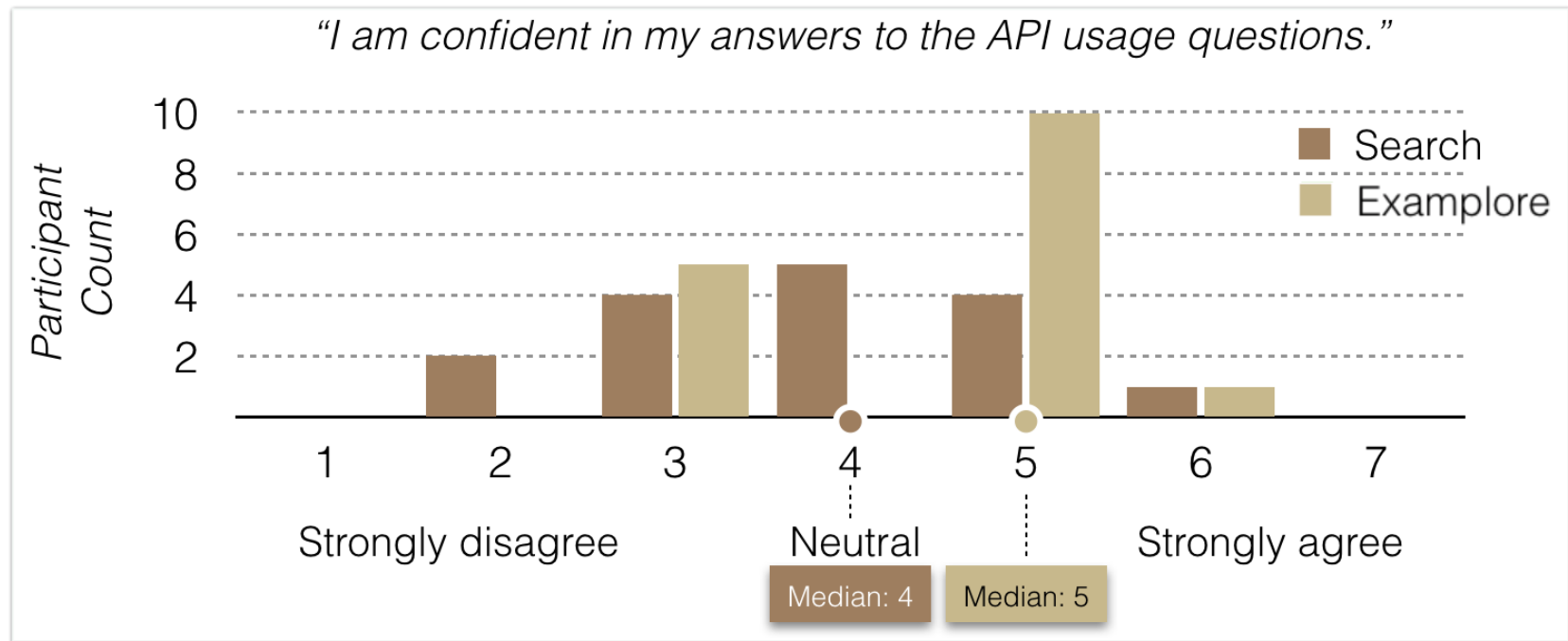
share improve this answer

answered Aug 4 '09 at 12:42

 ZZ Coder  
56.2k • 22 • 112 • 147

add a comment

# Lab Study Results



# Summary

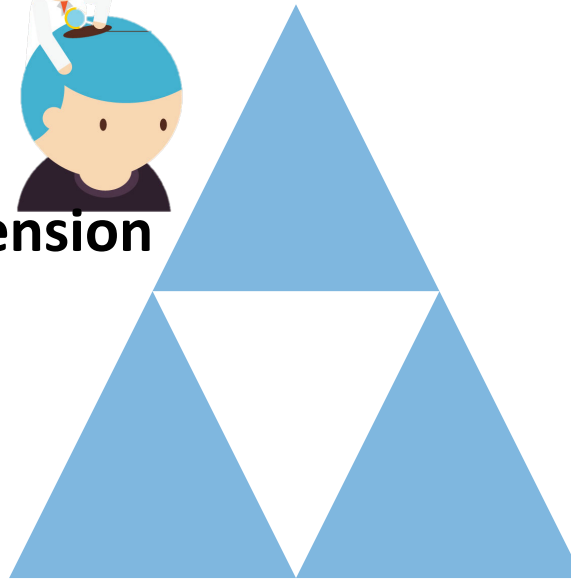
It's time to **go beyond** measuring **precision** and **recall** of software specification inference techniques



## 1. Comprehension



## 2. Interactive Navigation



## 3. Fit Developer Workflow





**UCLA**

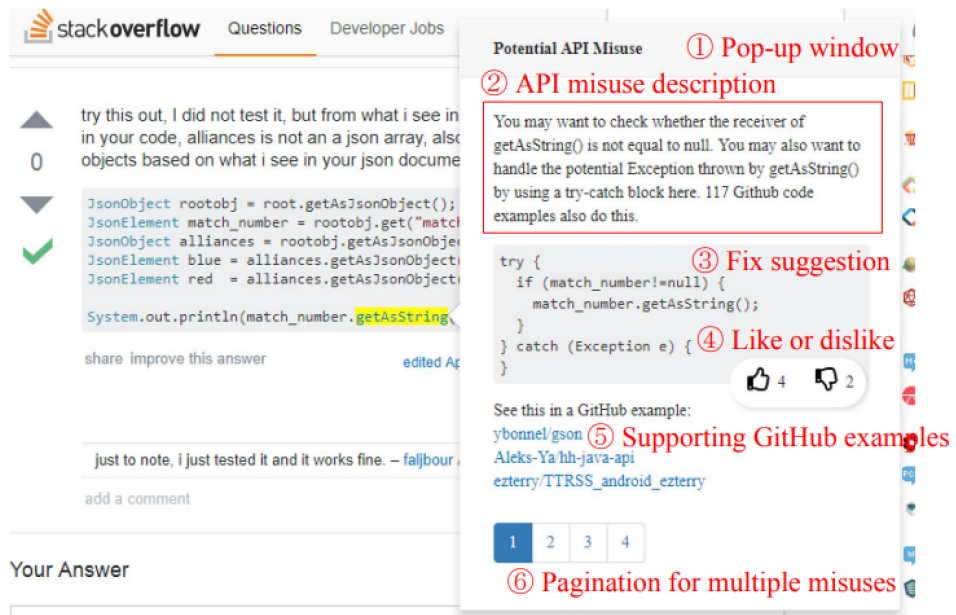
**Berkeley**  
UNIVERSITY OF CALIFORNIA

**IOWA STATE  
UNIVERSITY**

  
**GEORGE FOX  
UNIVERSITY**

**Collaborators: Tianyi Zhang, Elena Glassman, Bjoern Hartmann, Ganesha Upadhyaya, Hriday Rajan, Anastasia Reinhart**

# ExampleCheck and Examplelore



stackoverflow Questions Developer Jobs

try this out, I did not test it, but from what i see in your code, alliances is not an a json array, also objects based on what i see in your json docume

```
JsonObject rootobj = root.getAsJsonObject();
JsonElement match_number = rootobj.get("match");
JsonObject alliances = rootobj.getAsJsonObject("alliances");
JsonElement blue = alliances.getAsJsonObject("blue");
JsonElement red = alliances.getAsJsonObject("red");

System.out.println(match_number.getAsString());
```

share improve this answer edited At

just to note, i just tested it and it works fine. – fallbour

add a comment

Your Answer

**Potential API Misuse** ① Pop-up window

② API misuse description

You may want to check whether the receiver of `getAsString()` is not equal to null. You may also want to handle the potential Exception thrown by `getAsString()` by using a try-catch block here. 117 Github code examples also do this.

```
try {
    if (match_number!=null) {
        match_number.getAsString();
    }
} catch (Exception e) {
```

③ Fix suggestion

④ Like or dislike

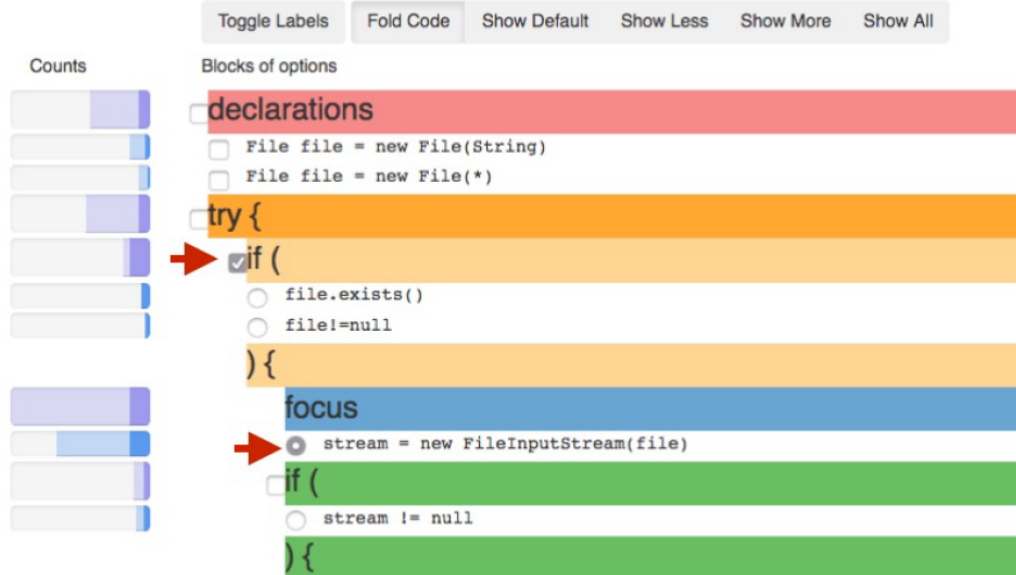
See this in a GitHub example:

ybonnel/gson ⑤ Supporting GitHub examples

Aleks-Ya'hi-java-api

ezterry/TTRSS\_android\_ezterry

⑥ Pagination for multiple misuses



Counts

Toggle Labels Fold Code Show Default Show Less Show More Show All

Blocks of options

declarations

- File file = new File(String)
- File file = new File(\*)

try {

- if (
- file.exists()
- file!=null

) {

focus

- stream = new FileInputStream(file)
- if (
- stream != null

) {



chrome web store

Tool is available at  
<http://examplelore.cs.ucla.edu:3000/>