Reducing friction between clients and maintainers of software libraries

PhD Defense

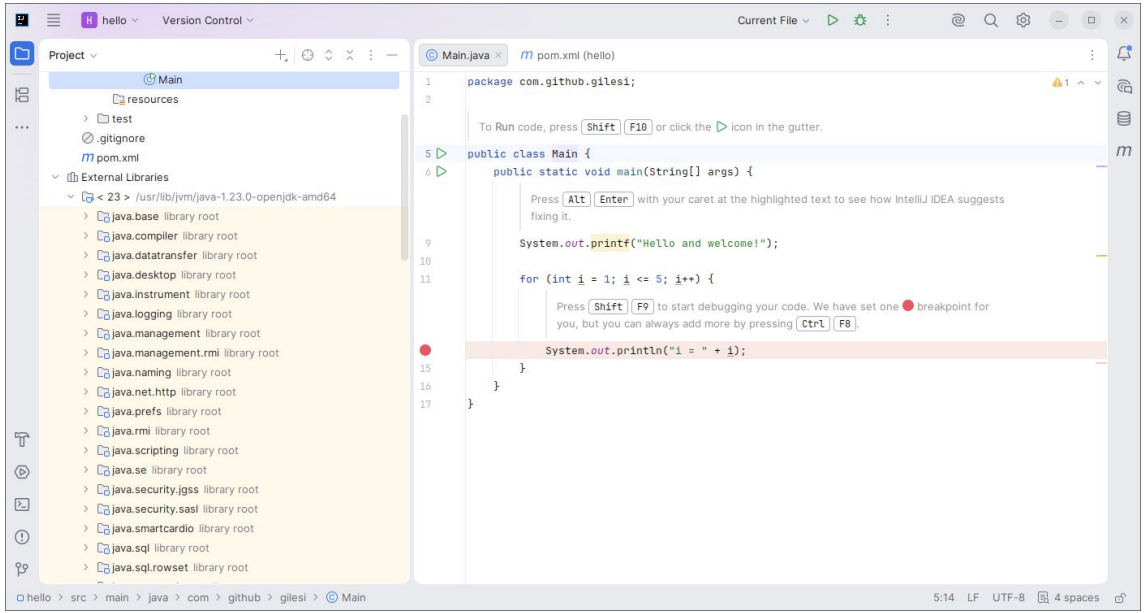
1st April, 2026

2:00 PM
Amphi LaBRI
Talence, France

Mme Anne Etien	Professeure des universités	Université Lille 1	Rapporteure
M. Jannik Laval	Maître de Conférences	Université Lumière Lyon 2	Rapporteur
M. Djamel Eddine-Khelladi	Chargé de Recherche	CNRS	Examinateur
M. Hicham Lakhlef	Professeur des universités	Bordeaux INP	Examinateur
M. Jean-Rémy Falleri	Professeur des universités	Bordeaux INP	Directeur
M. Thomas Degueule	Chargé de Recherche	CNRS	Co-directeur

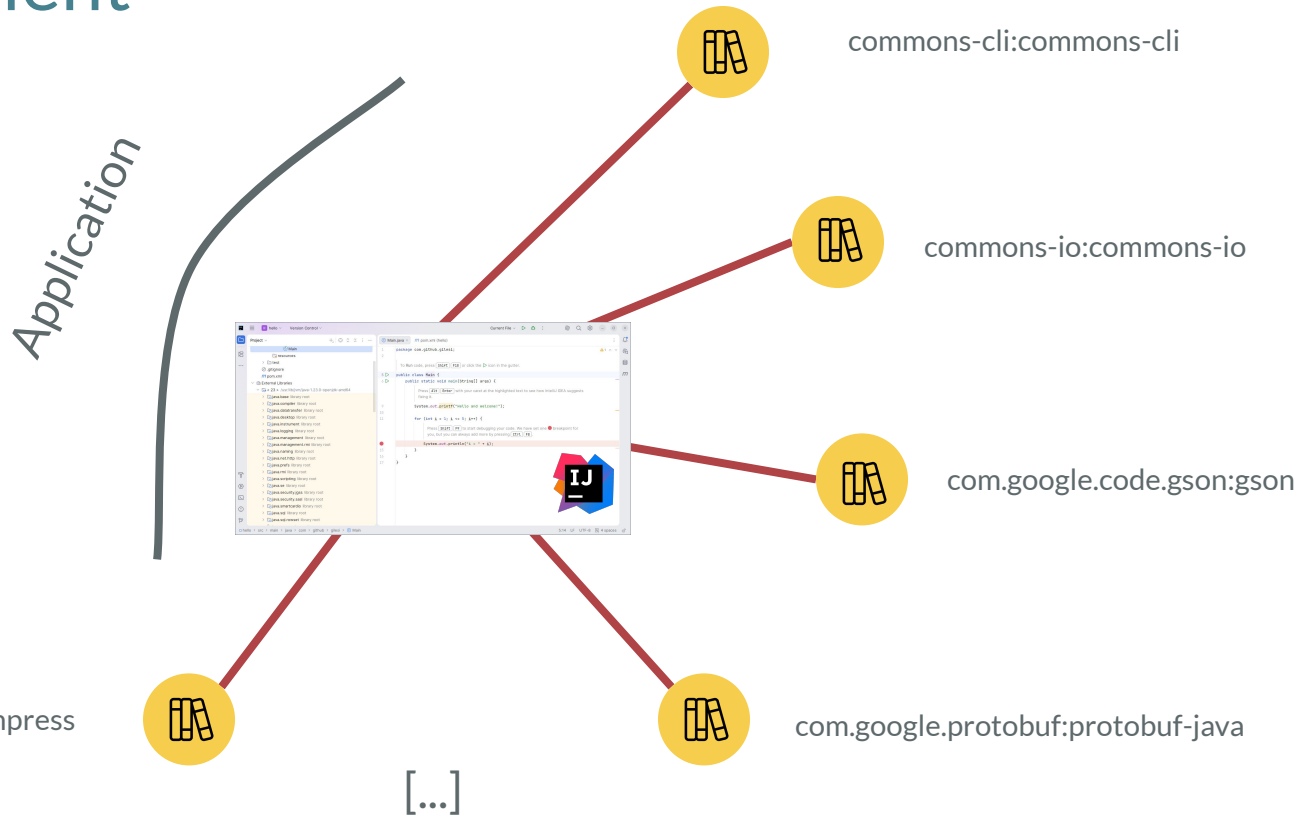
us A ge driven software
Li brary
E vo-
lutio n ANR ALIEN (ANR--21--CE25--0007)

Software



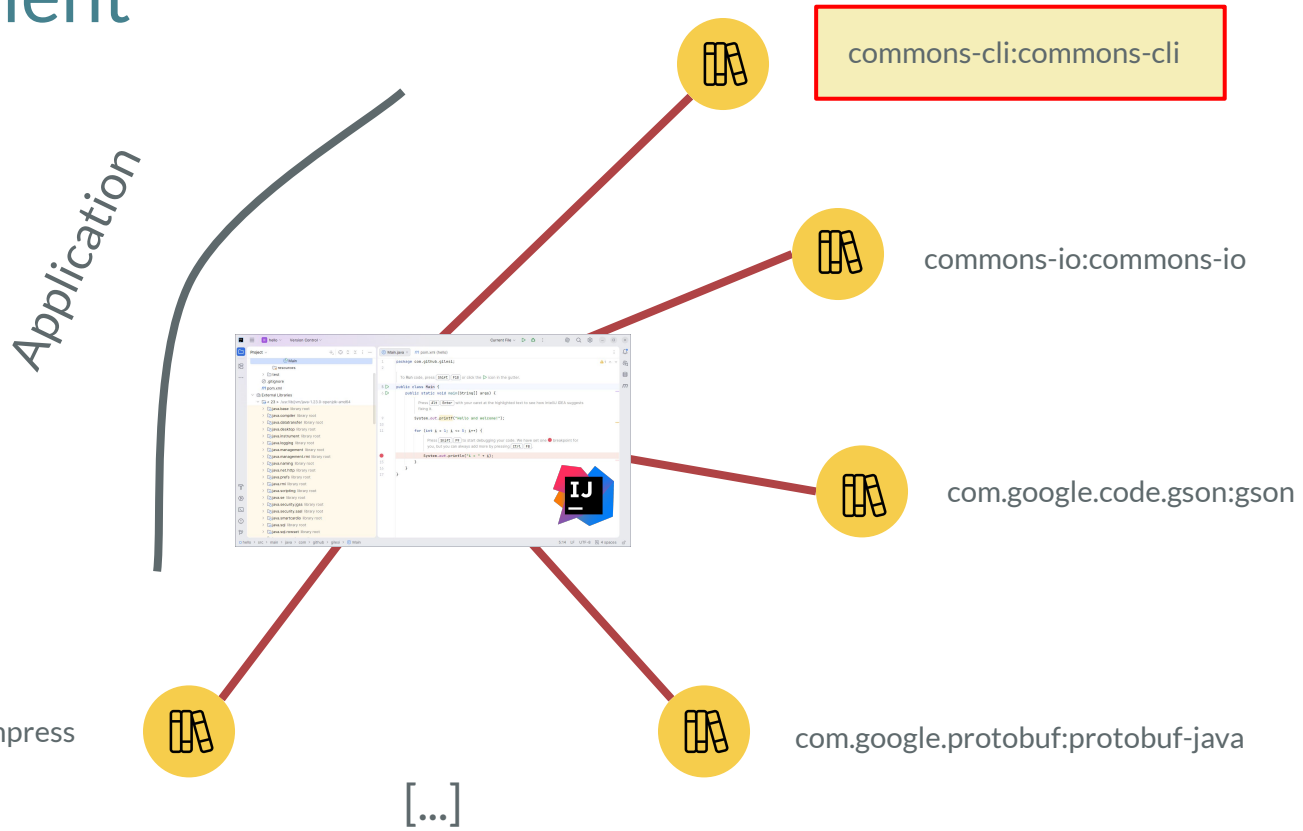
Application

Software… Client



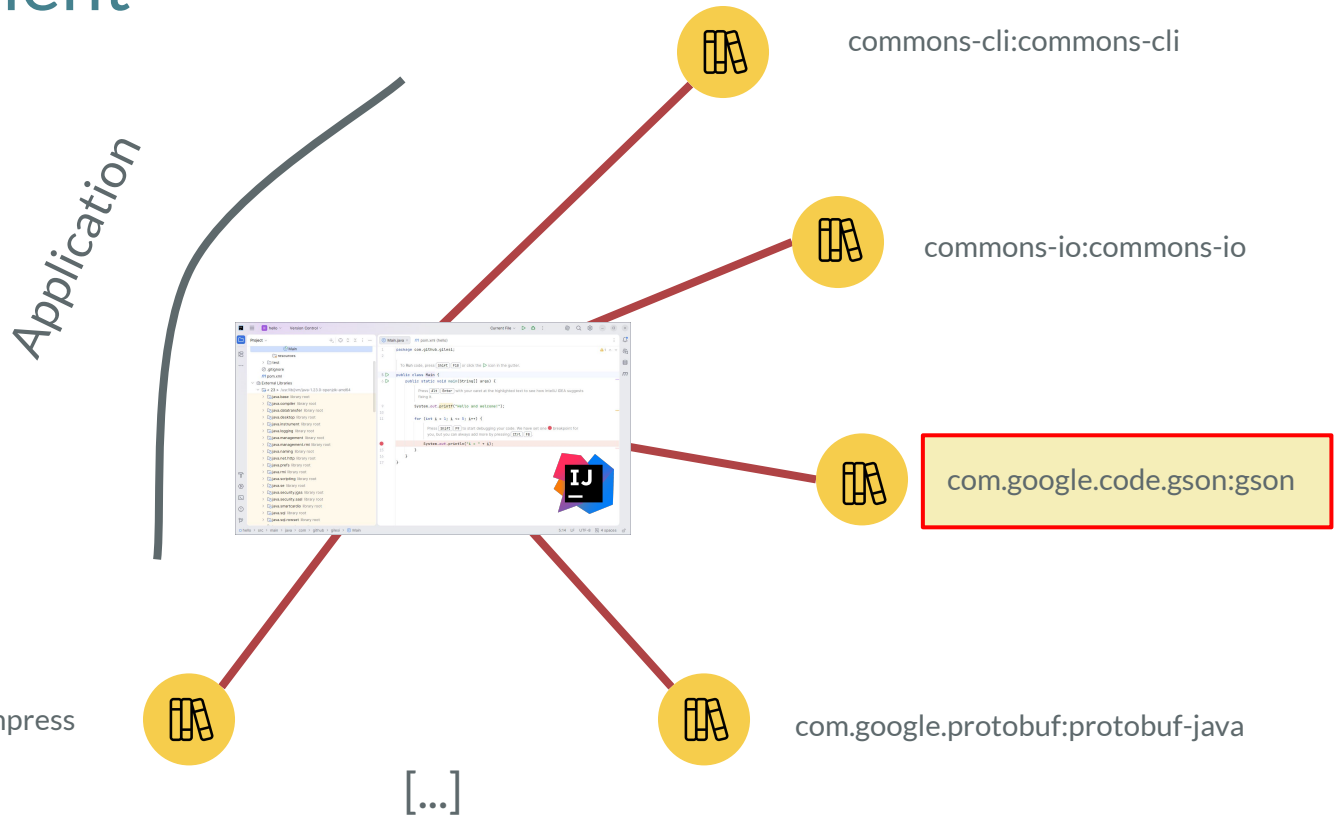
https://github.com/JetBrains/intellij-community/network/dependencies?q=ecosystem%3AMaven

Software… Client



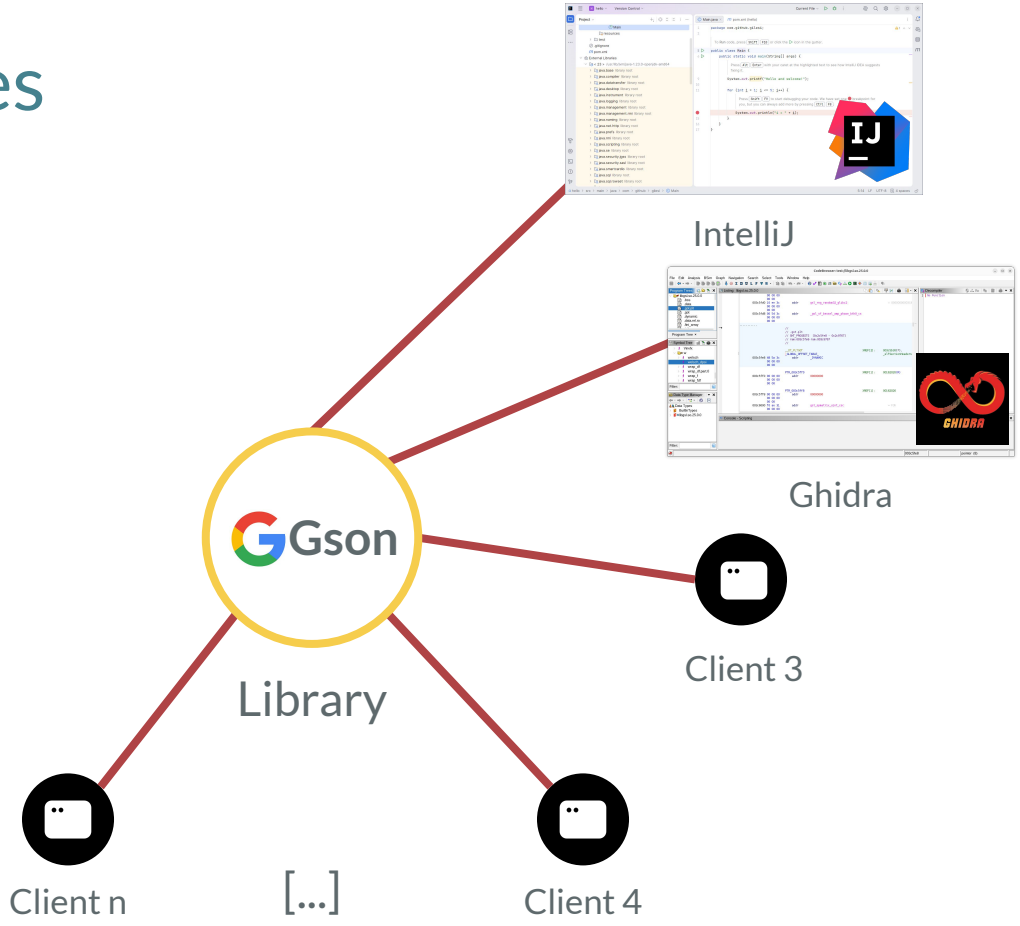
https://github.com/JetBrains/intellij-community/network/dependencies?q=ecosystem%3AMaven

Software… Client



https://github.com/JetBrains/intellij-community/network/dependencies?q=ecosystem%3AMaven

Software Libraries



What is a software library?

```
ArrayList<Integer> lst = new ArrayList<Integer>();
lst.add(42);
lst.add(1337);
```

Client

```
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  private int size;
  ...
}
```

API

Client Maintainers

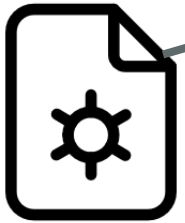
Library Maintainers

Consumption of Software Libraries

- Identifier: *e.g. com.google.code.gson:gson*
- Version: *e.g. 1.19*
- Managed through a package manager: *e.g. Maven/Gradle*

```
<dependencies>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.13.2</version>
    <scope>test</scope>
  </dependency>
  <dependency>
    <groupId>com.github.gilesi.samples</groupId>
    <artifactId>samplelibrary</artifactId>
    <version>1.0.0-SNAPSHOT</version>
  </dependency>
  <dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter-engine</artifactId>
    <version>5.11.0-M2</version>
    <scope>test</scope>
  </dependency>
</dependencies>

...
```

Application

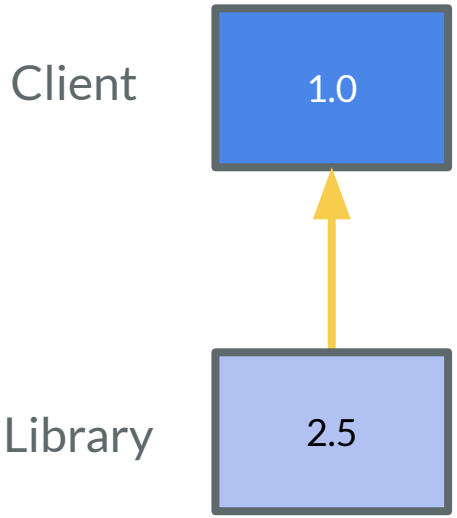


Dependency Declaration

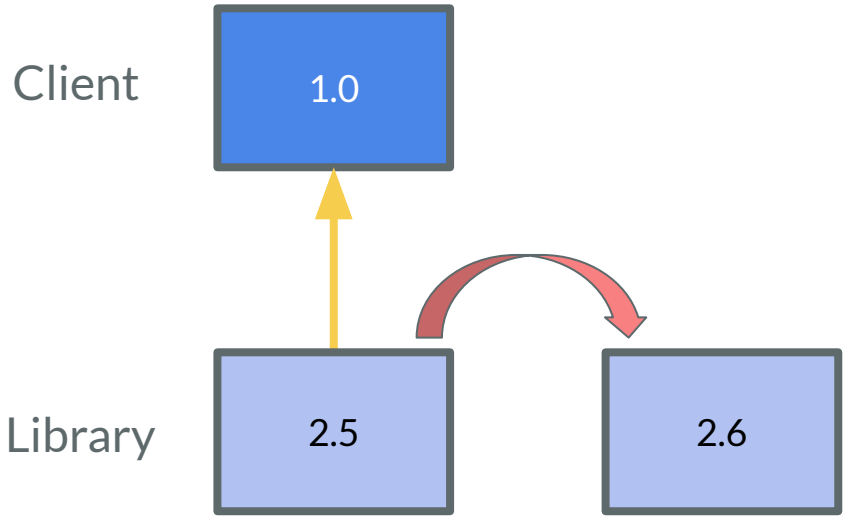
JAR

Package Repository (Online)

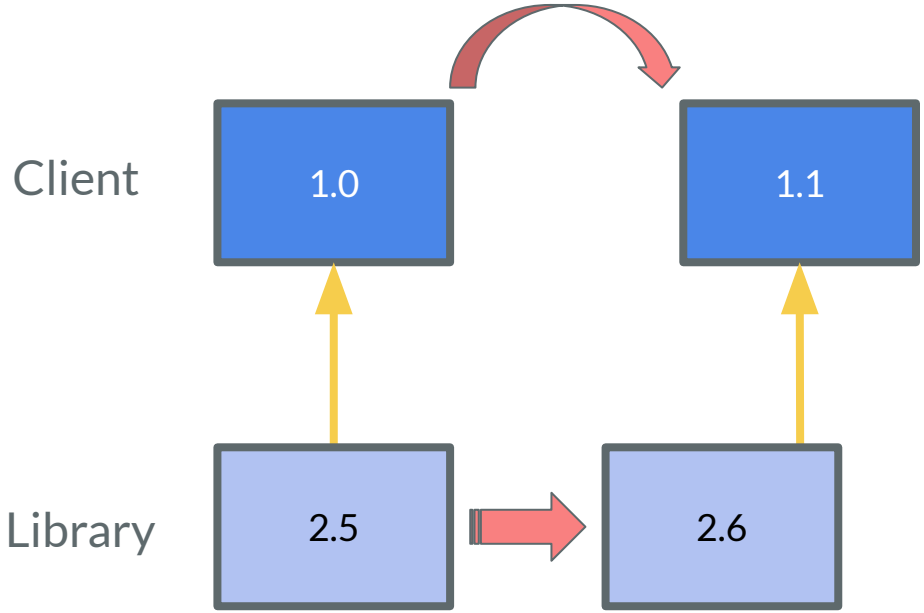
Co-evolution



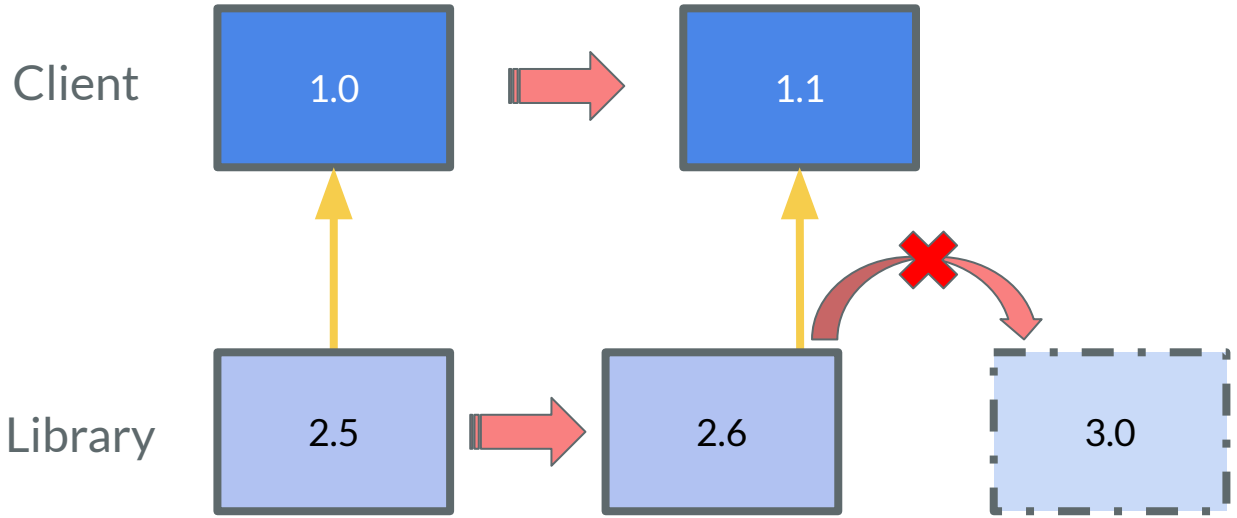
Co-evolution



Co-evolution



Problems linked with co-evolution



Problems linked with co-evolution: Library Maintainer Side

```
ArrayList<Integer> lst = new ArrayList<Integer>();
lst.add(42);
lst.add(1337);
```

Client

Library

```
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  public boolean addIfNotExist(E e) { ... }
  private int size;
  ...
}
```

API

Problems linked with co-evolution: Library Maintainer Side

```
ArrayList<Integer> lst = new ArrayList<Integer>();
lst.add(42);
lst.add(1337);
```

Client 1

?

Client 2

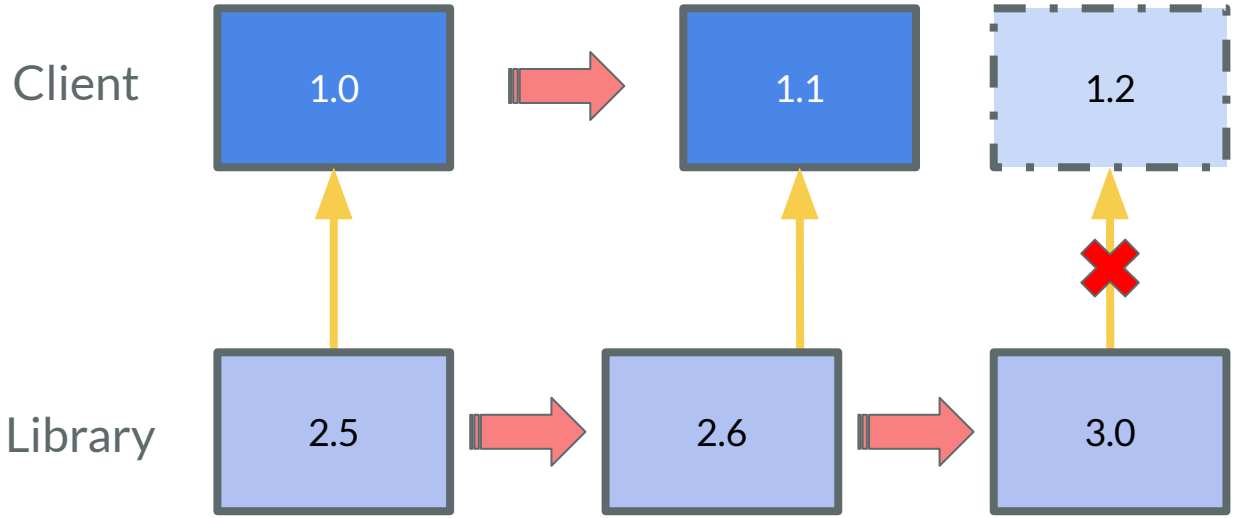
Library

```
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  public boolean addIfNotExist(E e) { ... }
  private int size;
  ...
}
```

API

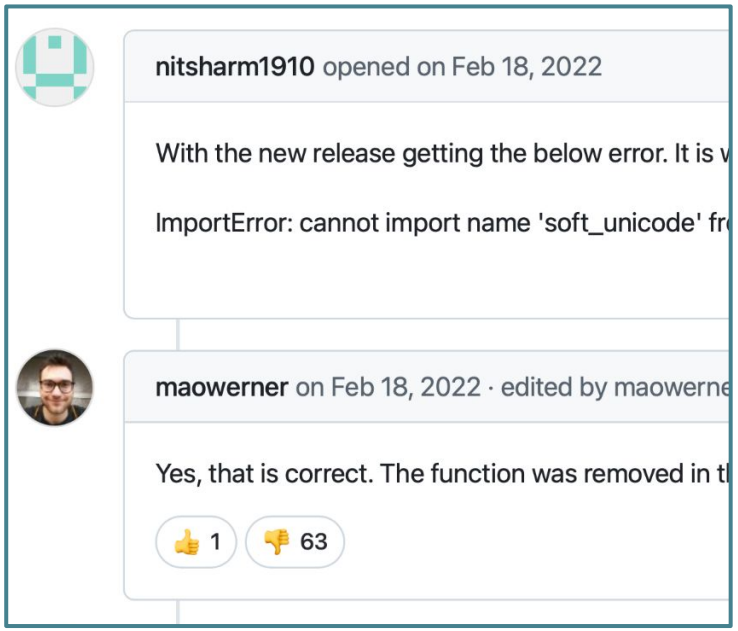
As a library maintainer, can I safely evolve my API?

Problems linked with co-evolution



Problems linked with co-evolution: Client Maintainer Side

- Breaking Changes
  - Undocumented or Documented
  - Caught during migration or post mortem
  - Hard to debug/troubleshoot
  - Can possibly block migration of the client
  - Client prevented from getting fixes and enhancements

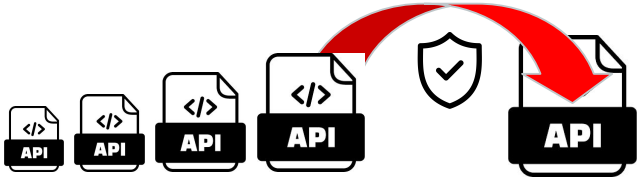


https://github.com/pallets/markupsafe/issues/284

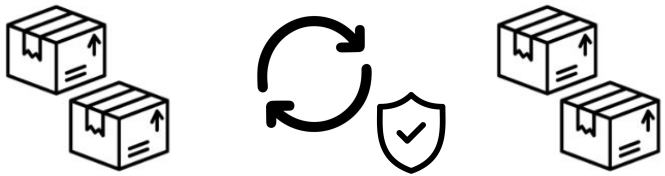
As a client maintainer, can I safely update my dependencies?

Problem statements

→ As a library maintainer, can I safely evolve my API?



→ As a client maintainer, can I safely update my dependencies?



Lightweight Syntactic API Usage Analysis with UCov

Gustave Monce, Thomas Couturou, Yasmine Hamdaoui, Thomas Degueule, and Jean-Rémy Falleri.

In Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension (ICPC '24). Association for Computing Machinery, New York, NY, USA, 426–437.

Application Programming Interface (API)

```
public class ArrayList<E> implements List<E> {
  public ArrayList<E>() {
     size = 0;
     internalArray = new E[10000];
  }

  public boolean add(E e) {
     internalArray[size++] = e;
  }

  private int size;
  private E[] internalArray;
  ...
}
```

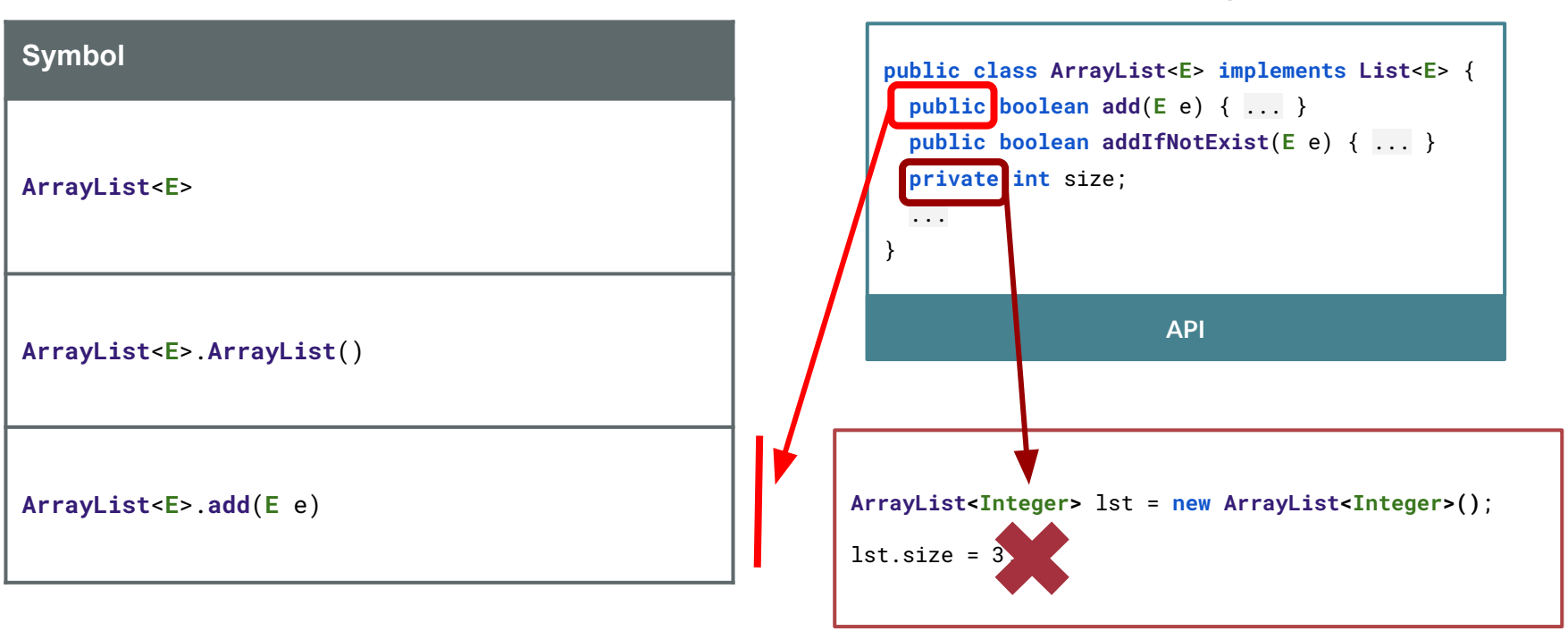
Application Programming Interface (API)

```
public class ArrayList<E> implements List<E> {
  public ArrayList<E>() {
     size = 0;
     internalArray = new E[10000];
  }

  public boolean add(E e) {
     internalArray[size++] = e;
  }

  private int size;
  private E[] internalArray;
  ...
}
```

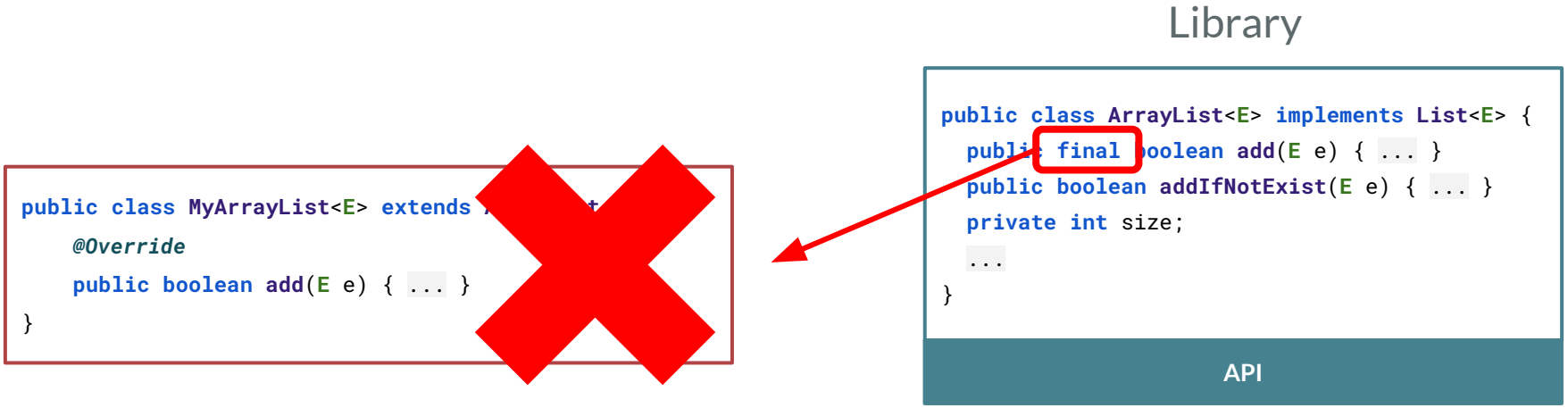
Application Programming Interface (API)

```
public class ArrayList<E> implements List<E> {
  public ArrayList<E>() {
     size = 0;
     internalArray = new E[10000];
  }

  public boolean add(E e) {
     internalArray[size++] = e;
  }

  private int size;
  private E[] internalArray;
  ...
}
```

Application Programming Interface (API)

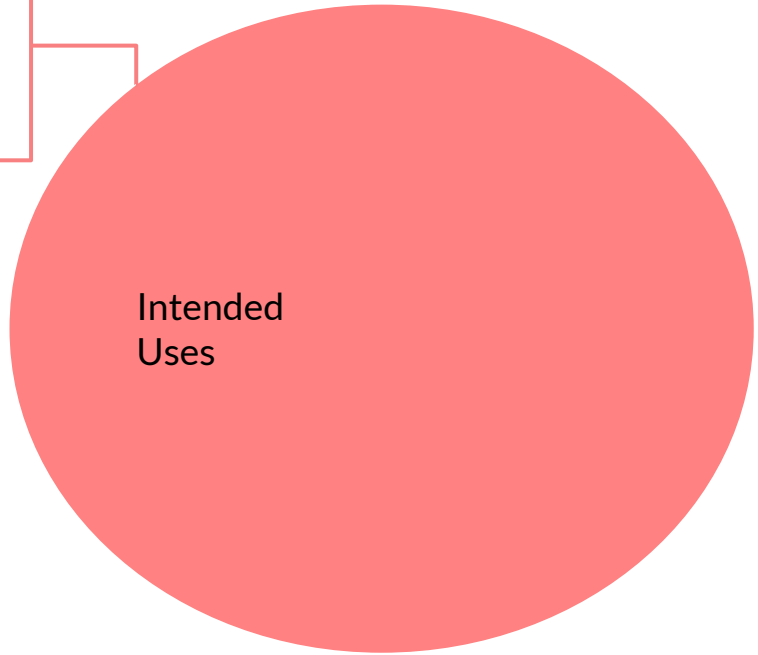


Application Programming Interface (API)



API Design & Usage

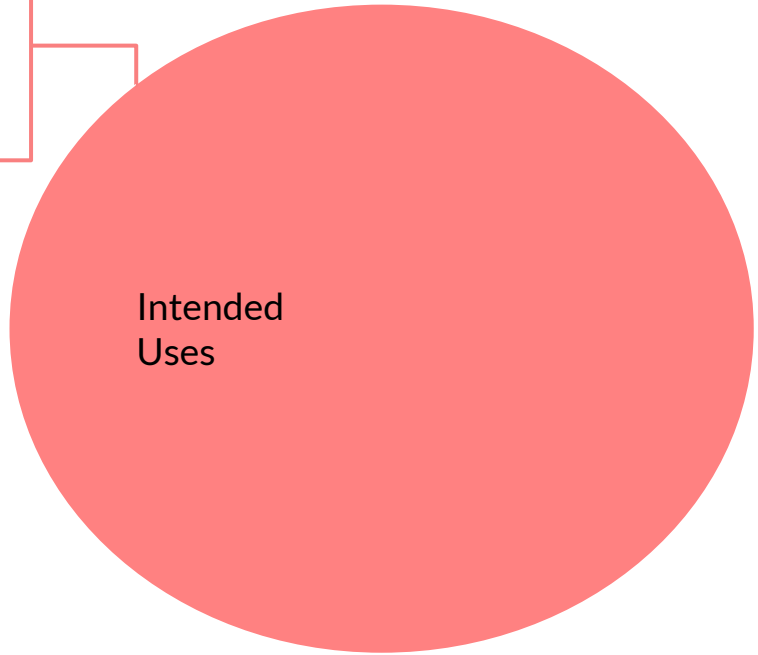
```
new ArrayList<Integer>();
lst.add(0);
lst.add(2);
lst.add(6 * 2);
lst.get(0);
```



API Design & Usage

```
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  private int size;
  ...
}
```

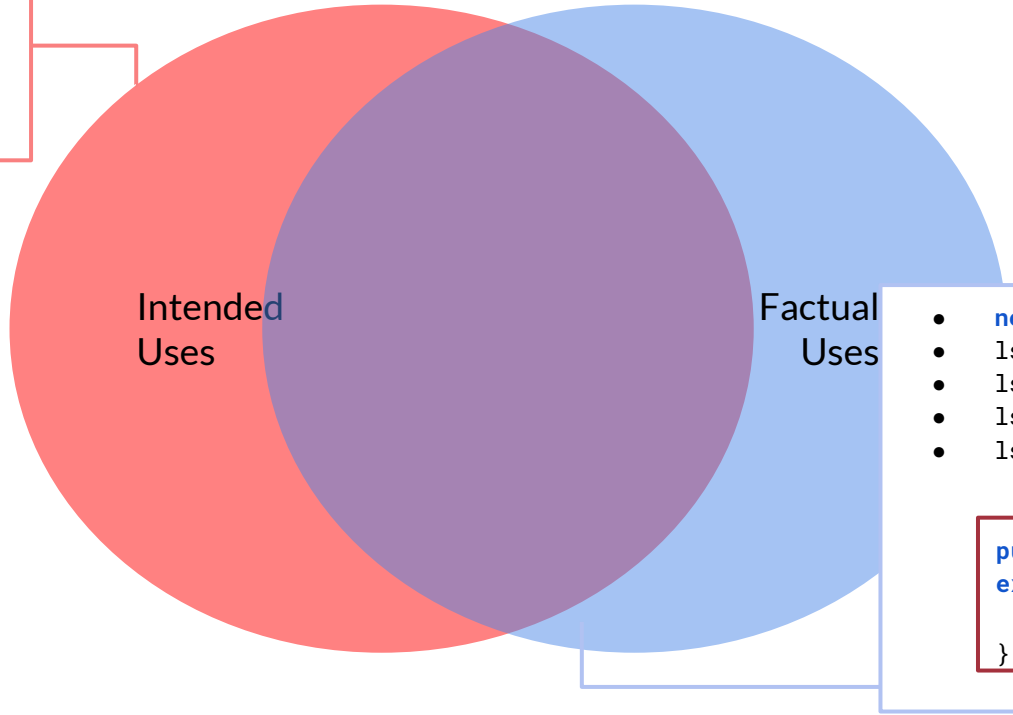
- `new ArrayList<Integer>();`
- `lst.add(0);`
- `lst.add(2);`
- `lst.add(6 * 2);`
- `lst.get(0);`



API Design & Usage

```
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  private int size;
  ...
}
```

```
• new ArrayList<Integer>();
• lst.add(0);
• lst.add(2);
• lst.add(6 * 2);
• lst.get(0);
```



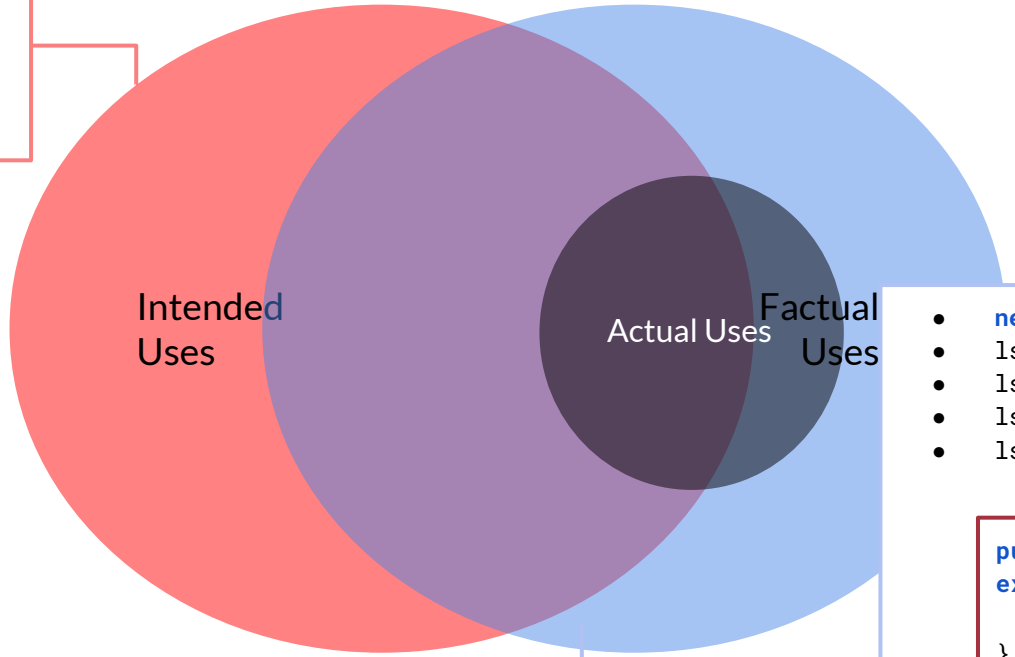

```
• new ArrayList<Integer>();
• lst.add(0);
• lst.add(2);
• lst.add(6 * 2);
• lst.get(0);
```

```
public class NewArrayList<E>
extends ArrayList<E> {
   @Override { ... }
}
```

API Design & Usage

```
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  private int size;
  ...
}
```

```
• new ArrayList<Integer>();
• lst.add(0);
• lst.add(2);
• lst.add(6 * 2);
• lst.get(0);
```



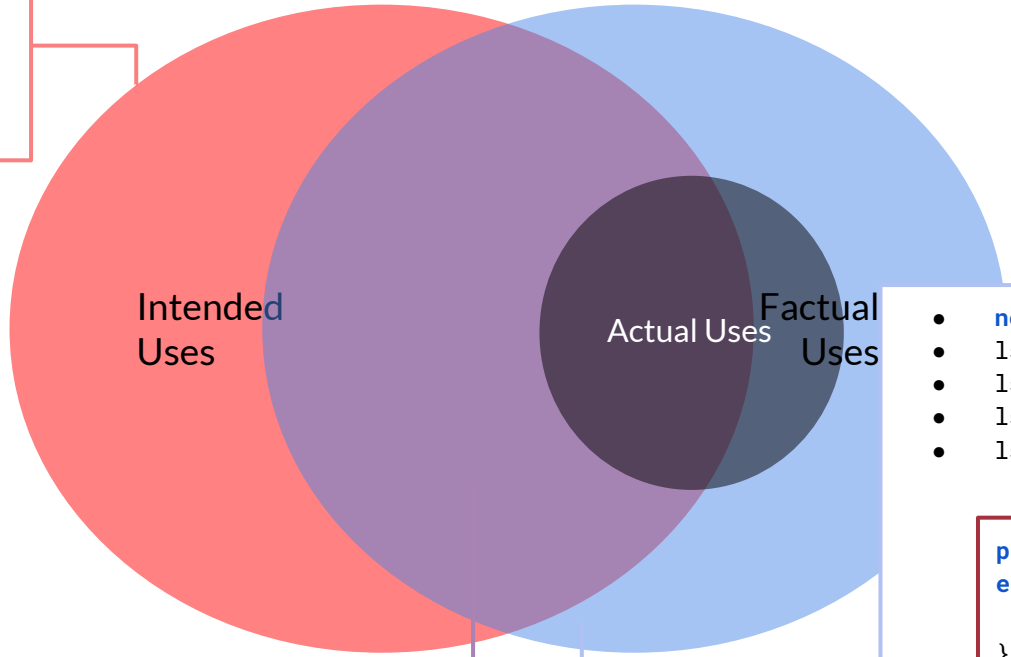

```
• new ArrayList<Integer>();
• lst.add(0);
• lst.add(2);
• lst.add(6 * 2);
• lst.get(0);
```

```
public class NewArrayList<E>
extends ArrayList<E> {
   @Override { ... }
}
```

API Design & Usage

```
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  private int size;
  ...
}
```

```
new ArrayList<Integer>();
lst.add(0);
lst.add(2);
lst.add(6 * 2);
lst.get(0);
```



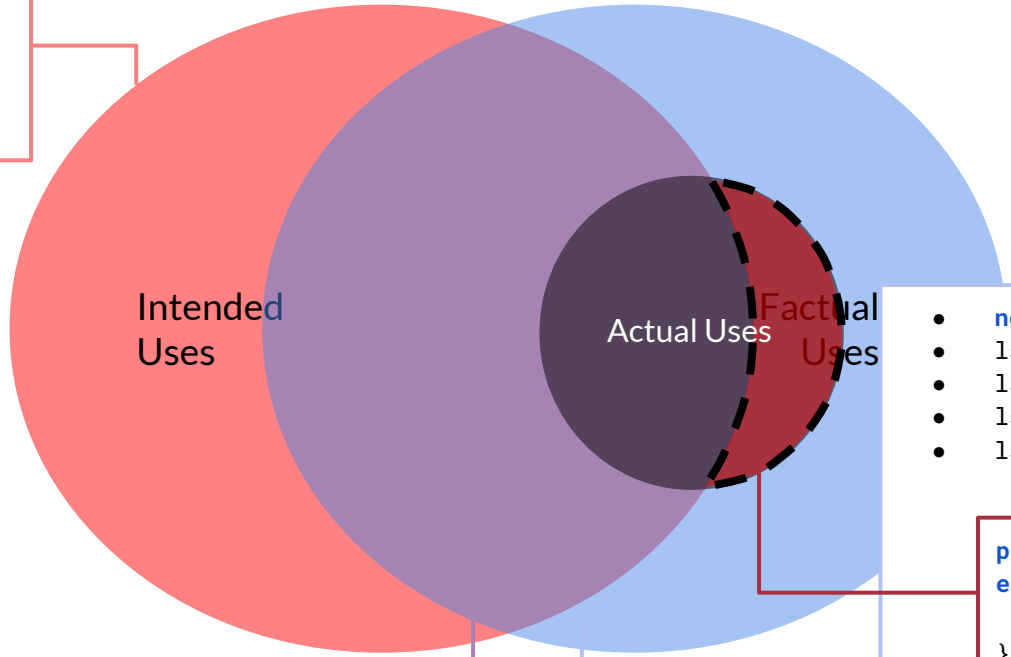

```
new ArrayList<Integer>();
lst.add(0);
lst.add(2);
lst.add(6 * 2);
lst.get(0);
```

```
public class NewArrayList<E>
extends ArrayList<E> {
   @Override { ... }
}
```

API Design & Usage

```
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  private int size;
  ...
}
```

```
new ArrayList<Integer>();
lst.add(0);
lst.add(2);
lst.add(6 * 2);
lst.get(0);
```




```
new ArrayList<Integer>();
lst.add(0);
lst.add(2);
lst.add(6 * 2);
lst.get(0);
```

```
public class NewArrayList<E>
extends ArrayList<E> {
   @Override { ... }
}
```

Software Library Consumers

```
ArrayList<Integer> lst = new ArrayList<Integer>();
lst.add(42);
lst.add(1337);
```

Clients

```
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  private int size;
  ...
}
```

API

Software Library Consumers

```java
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  private int size;
  ...
}
```

API

```java
ArrayList<Integer> lst = new ArrayList<Integer>();
lst.add(42);
lst.add(1337);
```

Clients

```java
@Test
public void testAdd() {
    ArrayList<Integer> lst = new ArrayList<Integer>();
    lst.add(42); assertEquals(42, lst.get(0));
}
```

Tests

Software Library Consumers

```java
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  private int size;
  ...
}
```

API

```java
ArrayList<Integer> lst = new ArrayList<Integer>();
lst.add(42);
lst.add(1337);
```

Clients

Using the ArrayList library

Below's example showcases a simple program that creates a list, and adds and retrieves a few elements

```java
ArrayList<Integer> lst = new ArrayList<Integer>();
lst.add(0);
for (int i = 0; i < 2024; i++)
    lst.add(lst.get(i - 1) + i);
```

Documentation/Samples

```java
@Test
public void testAdd() {
    ArrayList<Integer> lst = new ArrayList<Integer>();
    lst.add(42); assertEquals(42, lst.get(0));
}
```

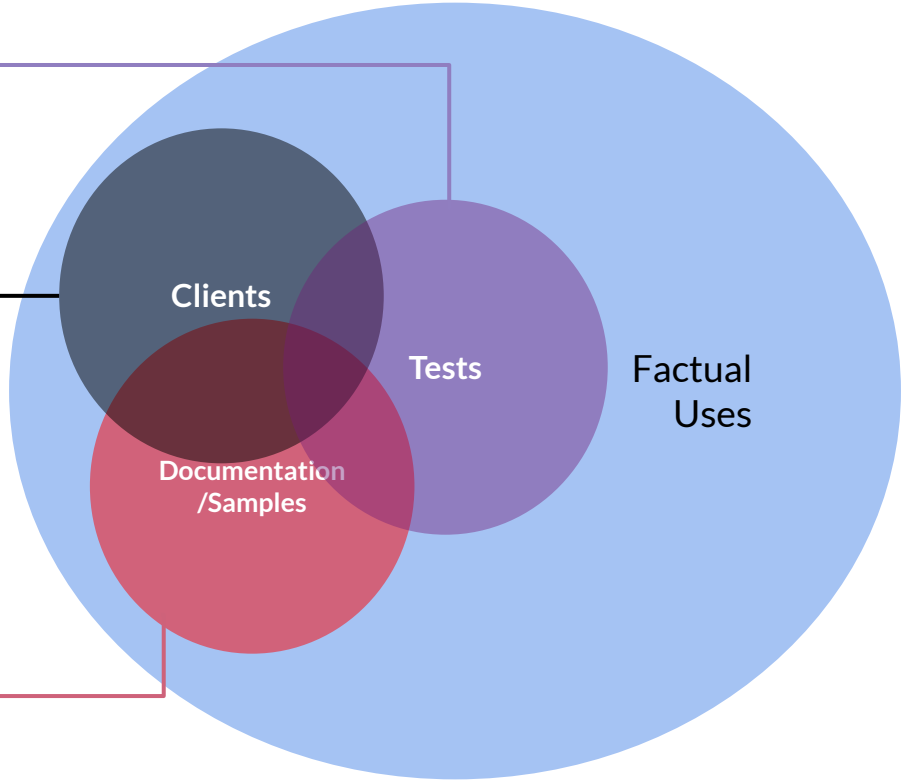
Tests

API Design & Usage

```
assert lst.size() == 2;
...
```

```
new ArrayList<Integer>();

lst.add(0);
lst.get(0);
```

```
public class
NewArrayList<E>
     extends ArrayList<E>
```



Using the ArrayList library

Below's example showcases a simple program that creates a list, and

```
ArrayList<Integer> lst = new ArrayList<Integer>();
lst.add(0);
for (int i = 0; i < 2024; i++)
    lst.add(lst.get(i - 1) + i);
```

Symbol-based modeling of uses (Existing Work)

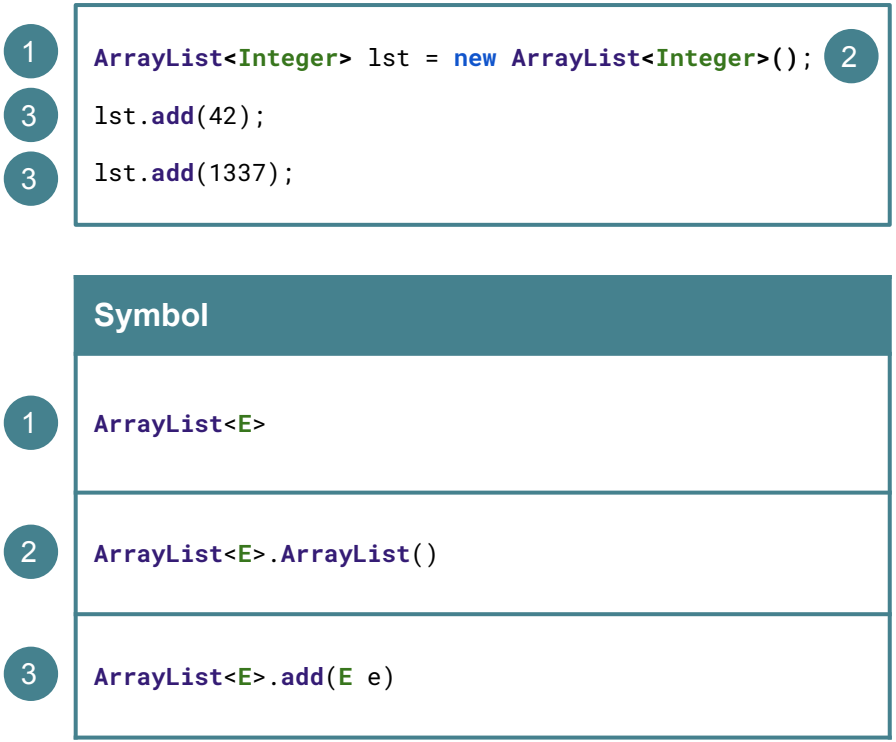
```
public class ArrayList<E> implements List<E>
{

  public boolean add(E e) { ... }

  private int size;

  ...

}
```

API

Symbol
`ArrayList<E>`
`ArrayList<E>.ArrayList()`
`ArrayList<E>.add(E e)`

Symbols

Symbol-based modeling of uses (Existing Work)



To identify notable hotspots and coldspots, i.e., parts of the APIs that are over- or under-utilized:

- Stylos, et al. *"Improving API documentation using API usage information"* (VLHCC09)
- Sawant, et al. *"fine-GRAPE: fine-grained APi usage extractor – an approach and dataset to investigate API usage"* (ESE17)
- Thummalapenta , et al. *"SpotWeb: Detecting Framework Hotspots and Coldspots via Mining Open Source Code on the Web"* (ASE08)

To address the problem of mining and recommending unordered or sequential API usage patterns and protocols to users:

- Nguyen, et al. *"FOCUS: A Recommender System for Mining API Function Calls and Usage Patterns"* (ICSE19)
- Robillard, et al. *"Automated API Property Inference Techniques"* (TSE13)
- Zhong, et al. *"MAPO: Mining and Recommending API Usage Patterns"* (LNPSE09)
- Wang, et al. *"Mining succinct and high-coverage API usage patterns from source code"* (MSR13)
- Buse, et al. *"Synthesizing API usage examples"* (ICSE22)

Symbol-based modeling of uses (Existing Work)

1
3
3

```
ArrayList<Integer> lst = new ArrayList<Integer>();
lst.add(42);
lst.add(1337);
```

2

Symbol
1 `ArrayList<E>`
2 `ArrayList<E>.ArrayList()`
3 `ArrayList<E>.add(E e)`

Symbols

```
public class MyArrayList<E> extends ArrayList<E> {
    @Override
    public boolean add(E e) { ... }
}
```

1

2

Symbol
`ArrayList<E>`
`ArrayList<E>.add(E e)`

1

2

Symbols

Syntactic Usage Model (SUM) (Contribution)

```
public class ArrayList<E> implements List<E>
{

  public boolean add(E e) { ... }

  private int size;

  ...

}
```

API

Exported Symbol
`public class ArrayList<E>`
`public ArrayList<E>()`
`public boolean add(E e)`

Syntactic Usage Model (SUM) (Contribution)

New!

```
public class ArrayList<E> implements List<E>
{

  public boolean add(E e) { ... }

  private int size;

  ...

}
```

API

Exported Symbol	Interaction (Use)
public class ArrayList<E>	*Referenced*
	Instantiated
	Extended
public ArrayList<E>()	*Invoked*
public boolean add(E e)	*Invoked*
	Overridden

SUM

Definition of each symbol and allowed interactions

Syntactic Usage Footprint (SUF) (Contribution)

1 3 3

```
ArrayList<Integer> lst = new ArrayList<Integer>();
lst.add(42);
lst.add(1337);
```

2

Symbol	Interaction
1 `public class ArrayList<E>`	*Referenced*
2 `public class ArrayList<E>`	*Instantiated*
2 `public ArrayList<E>()`	*Invoked*
3 `public boolean add(E e)`	*Invoked*

SUF 1

Syntactic Usage Footprint (SUF) (Contribution)

1 3 3

```
ArrayList<Integer> lst = new ArrayList<Integer>(); 2
lst.add(42);
lst.add(1337);
```

```
public class MyArrayList<E> extends ArrayList<E> {
    @Override
    public boolean add(E e) { ... } 2
}
```

1

Symbol	Interaction
1 `public class ArrayList<E>`	*Referenced*
2 `public class ArrayList<E>`	*Instantiated*
2 `public ArrayList<E>()`	*Invoked*
3 `public boolean add(E e)`	*Invoked*

SUF 1

Symbol	Interaction
`public class ArrayList<E>`	*Extended*
`public boolean add(E e)`	*Overridden*

SUF 2

API

```
public class ArrayList<E> implements List<E> {
  public boolean add(E e) { ... }
  private int size;
  ...
}
```

SUM

Symbol	Interaction
`public class ArrayList<E>`	*Referenced*
	Instantiated
	Extended
`public ArrayList<E>()`	*Invoked*
`public boolean add(E e)`	*Invoked*
	Overridden

SUF 1

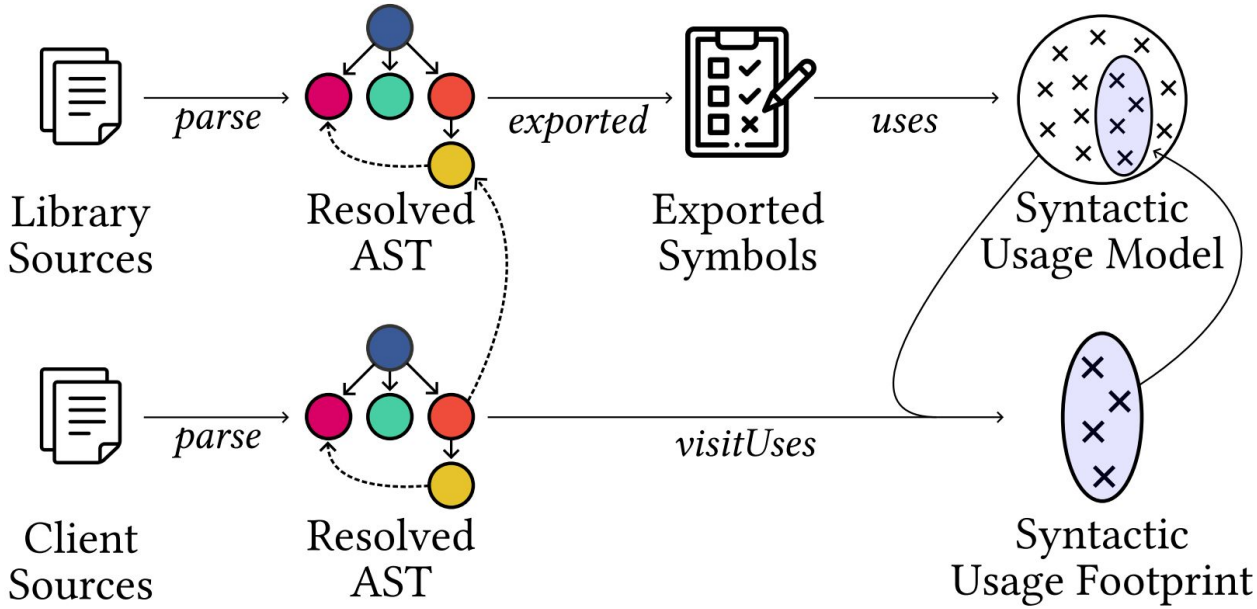
Symbol	Interaction
`public class ArrayList<E>`	*Referenced*
	Instantiated
`public ArrayList<E>()`	*Invoked*
`public boolean add(E e)`	*Invoked*

SUF 2

Symbol	Interaction
`public class ArrayList<E>`	*Extended*
`public boolean add(E e)`	*Overridden*

UCov: SUMs & SUFs for Java

github.com/
alien-tools/ucov



Usage model in UCov

Symbol Type	Use	Example	Resolved API Symbol
Type	Reference	`String s`	`java.lang.String`
		`void f(Integer i)`	`java.lang.Integer`
		`Integer f()`	`java.lang.Integer`
		`void f()throws IOException`	`java.io.IOException`
		`catch (IOException e)`	`java.io.IOException`
		`List<String> l`	`java.lang.String`
		etc.	*etc.*
Class	Instantiation	`new Integer(42)`	`java.lang.Integer`
	Inheritance	`class T extends Thread`	`java.lang.Thread`
Interface	Implementation	`class R implements Runnable`	`java.lang.Runnable`
		`Runnable r = ()→{...}`	`java.lang.Runnable`
	Extension	`interface R extends Runnable`	`java.lang.Runnable`
Constructor	Invocation	`new Integer(42)`	`java.lang.Integer(int)`
Method	Invocation	`"a".length()`	`java.lang.String.length()`
	Static invocation	`String.valueOf(42)`	`java.lang.String.valueOf(int)`
	Overriding	`@Override void run()`	`java.lang.Thread.run()`
		`Runnable r = ()→{...}`	`java.lang.Runnable.run()`
Field	Field read	`Integer.MAX_VALUE`	`java.lang.Integer.MAX_VALUE`
	Field write	`Point.x = 2`	`java.awt.Point.x`

Inspired and refined from the work of Qiu et al.

Dong Qiu, Bixin Li, and Hareton Leung. "*Understanding the API usage in Java.*" Information and Software Technology, 73:81–100, May 2016. ISSN 09505849. doi:10.1016/j.infsof.2016.01.011.

Evaluation: Case Study

zenodo.org/records/
10571867

Classical

```
//commons.apache.org
//proper/commons-cli/usage.html


CommandLineParser parser = new
DefaultParser();
Options options = new Options();
options.addOption("a", "all",
  false, "do not hide entries");
options.addOption("C", false,
  "list entries by columns");
```

commons-cli

Framework

```
//sparkjava.com
//documentation#routes


get("/", (rq,rs) → { ... });
put("/", (rq,rs) → { ... });
post("/", (rq,rs) → { ... });
```

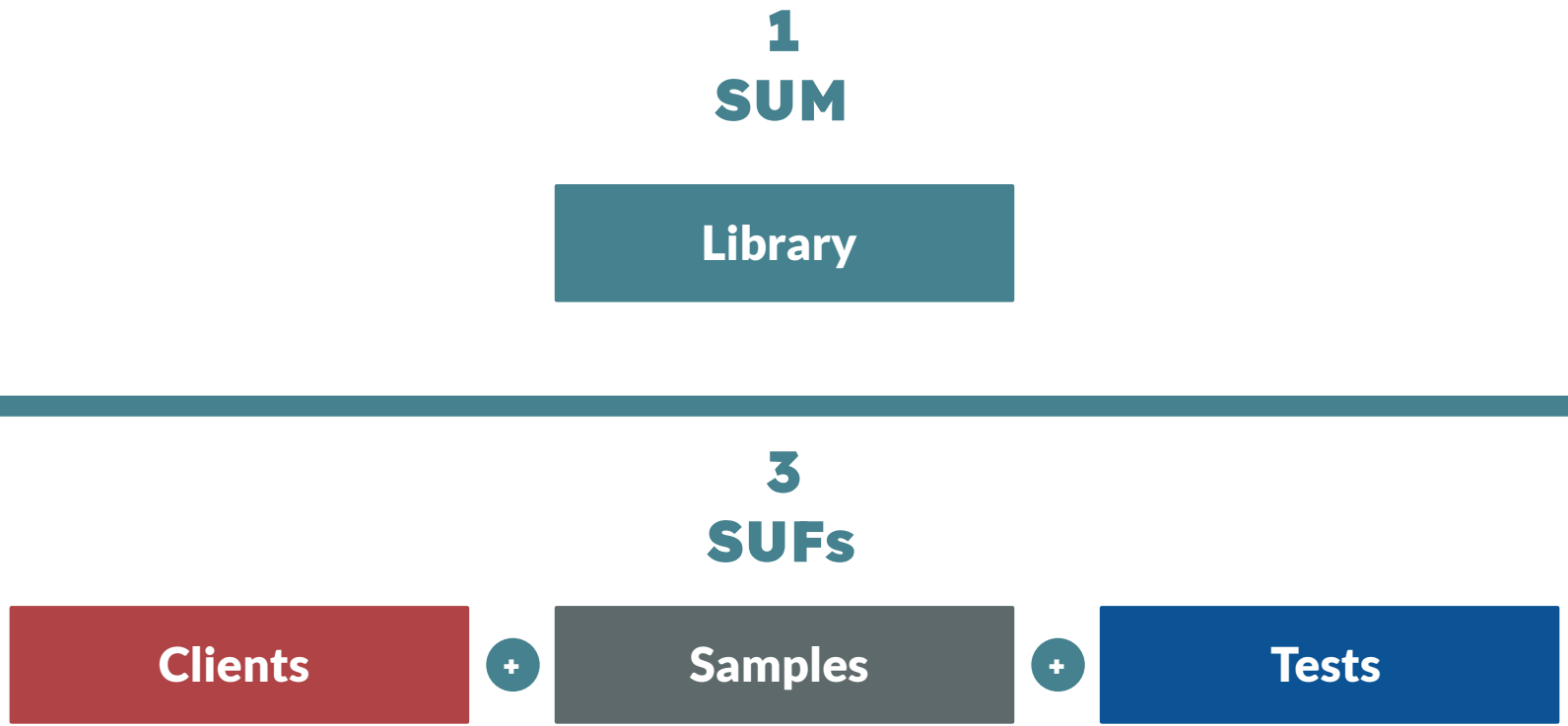
sparkjava

Fluent

```
//jsoup.org/cookbook
//input/load-document-from-url


Document doc = Jsoup
  .connect("http://example.com")
  .data("query", "Java")
  .userAgent("Mozilla")
  .cookie("auth", "token")
  .timeout(3000)
  .post();
```

jsoup

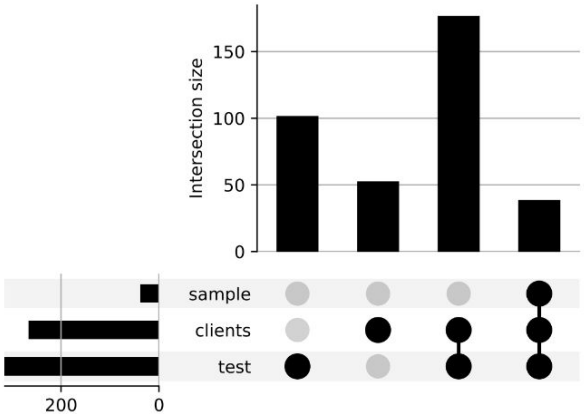
Evaluation: Methodology



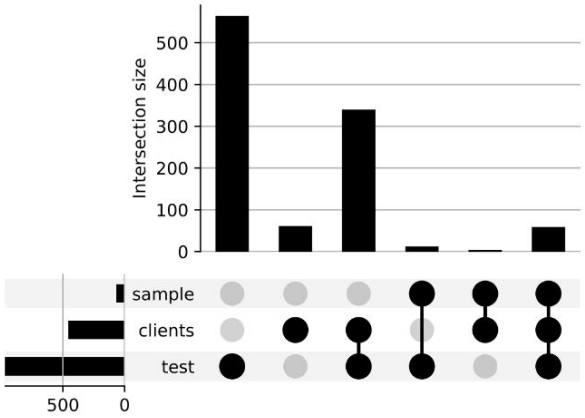
Results: Library & Syntactic Usage Model (SUM) metrics

	commons-cli	jsoup	spark
SUM - # API symbols	291	1138	771
SUM - # Legal interactions	755	2941	1960
Stars	309	10.4k	9.5k
Commits	1374	1889	1067
Size (LoC)	6307	27817	11298
Contributors	46	97	100
Clients	49k	131k	30k

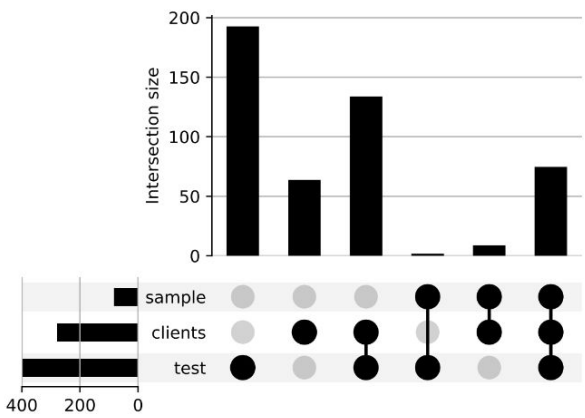
Results: Syntactic Usage Footprint (SUF) intersections



commons-cli

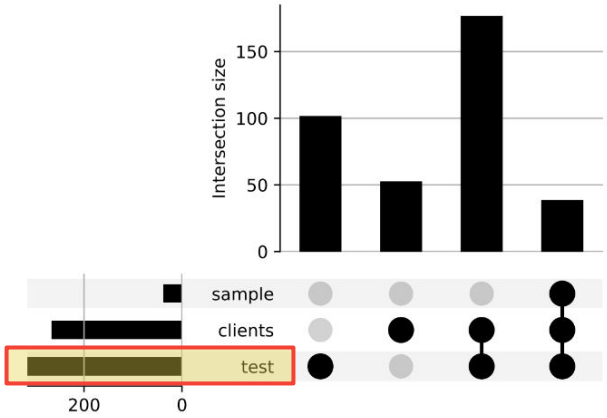


jsoup

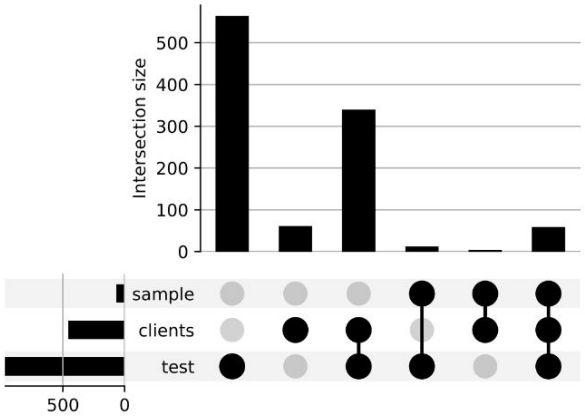


sparkjava

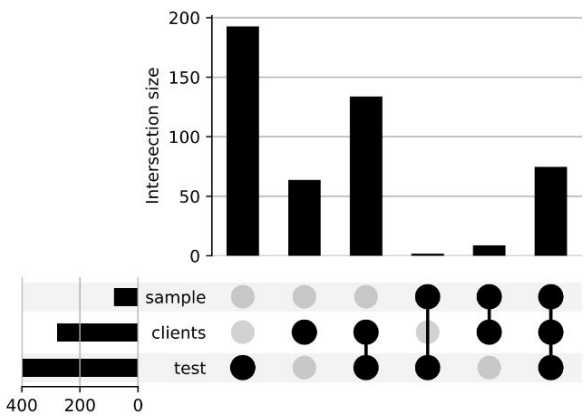
Results: Syntactic Usage Footprint (SUF) intersections



commons-cli

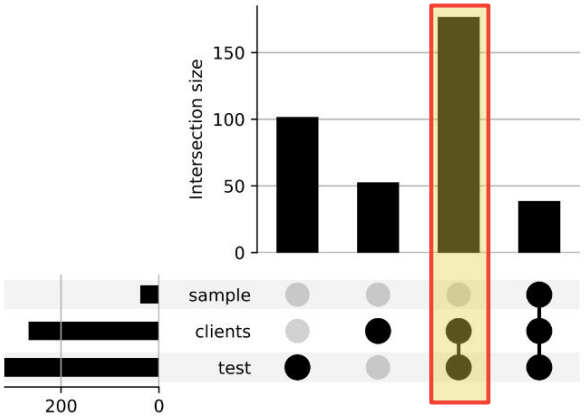


jsoup

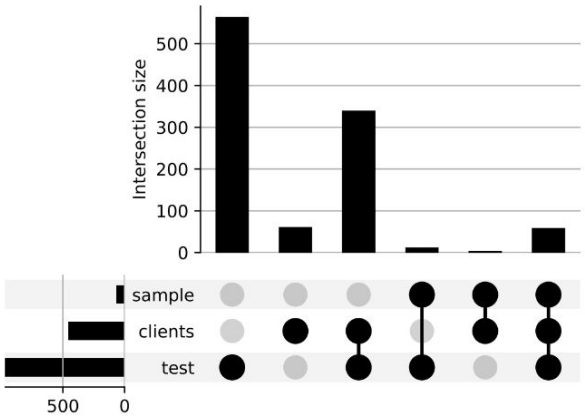


sparkjava

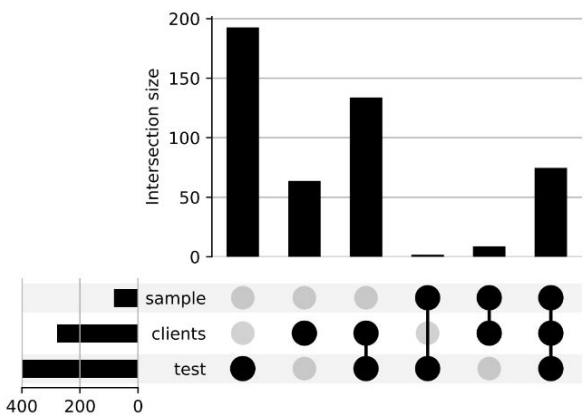
Results: Syntactic Usage Footprint (SUF) intersections



commons-cli

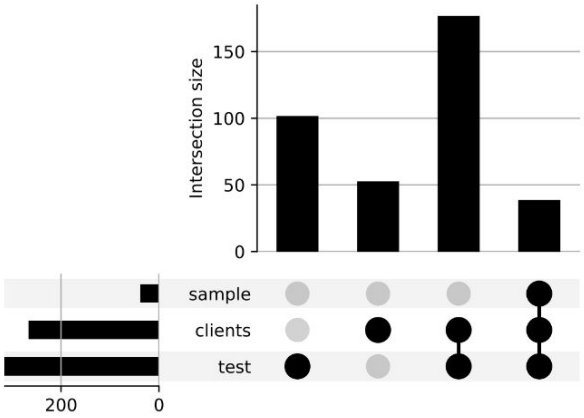


jsoup

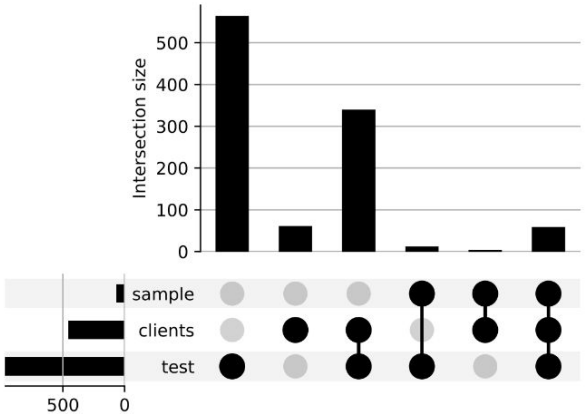


sparkjava

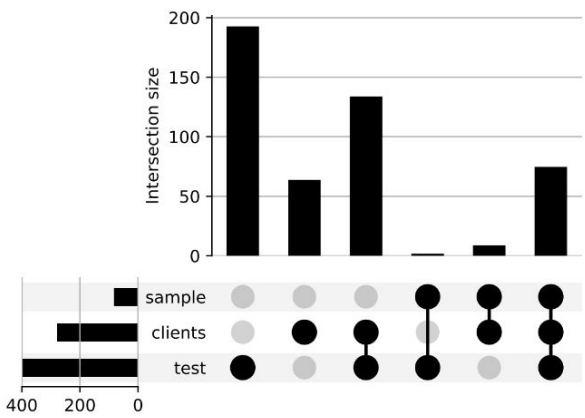
Results: Syntactic Usage Footprint (SUF) intersections



commons-cli



jsoup



sparkjava

Unexpected use in JSoup

Library

```
// src/main/java/org/jsoup/internal/StringUtil.java

/**
 A minimal String utility class. Designed for <b>internal</b>
 jsoup use only - the API and outcome may change without
 notice.
 */
public final class StringUtil {
    ...
}
```

Sample (& Clients too!)

```
// src/main/java/org/jsoup/examples/HtmlToPlainText.java

package org.jsoup.examples;

public class HtmlToPlainText {
    ...
    private static class FormattingVisitor
                        implements NodeVisitor {
        ...
        public void tail(Node node, int depth) {
            String name = node.nodeName();
            if (StringUtil.in(name, ...))
                append("\n");
            else if (name.equals("a"))
                append(String.format(" <%s>",
                        node.absUrl("href")));
        }
        ...
    }
}
```

Unexpected interaction in commons-cli

Library

```
// src/java/org/apache/commons/cli/Parser.java

public class PosixParser extends Parser {

    /**
     * @param stopAtNonOption specifies whether to stop interpreting
     * the arguments when a non option has been encountered and to add
     * them to the CommandLines args list.
     */
     public CommandLine parse(Options options, String[] arguments,
                boolean stopAtNonOption) throws ParseException { ... }
}
```

```
// java -jar ./main.jar -d -t

PosixParser p = new PosixParser();
Options o = new Options();
o.addOption("t", false, "display current time");
CommandLine c = p.parse(o, args, false);
```

Client

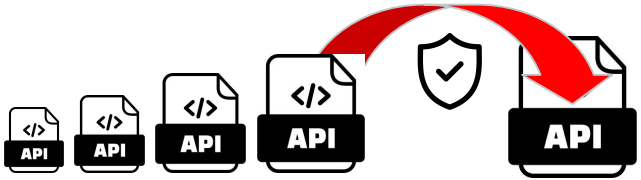
```
package org.obolibrary.robot;

/**
 * A custom CommandLineParser that ignores
 * unrecognized options without throwing an
 * exception. See
 * http://stackoverflow.com/a/8613949
 */
public class ExtPosixParser extends PosixParser {
  ...
  @Override
  protected void processOption(final String arg,
                final ListIterator iter)
       throws ParseException { ... }
}
```

UCov

- Interaction-based modeling of uses
  - Helps understanding the current API fully
- Support for discussing API design
  - Helps drive changes on the current API
- Support for discussing alignment with in-the-wild uses
  - Helps understand actual consumer usages in the wild, spot unexpected interactions and highlight flaws in the current design, tests, or documentation

✔ As a library maintainer, can I safely evolve my API?

→ Lightweight Syntactic API Usage Analysis ***with UCov***

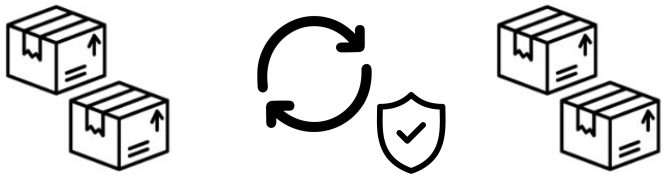


Problem statements

✔ As a library maintainer, can I safely evolve my API?

- Lightweight Syntactic API Usage Analysis ***with UCov***

→ As a client maintainer, can I safely update my dependencies?

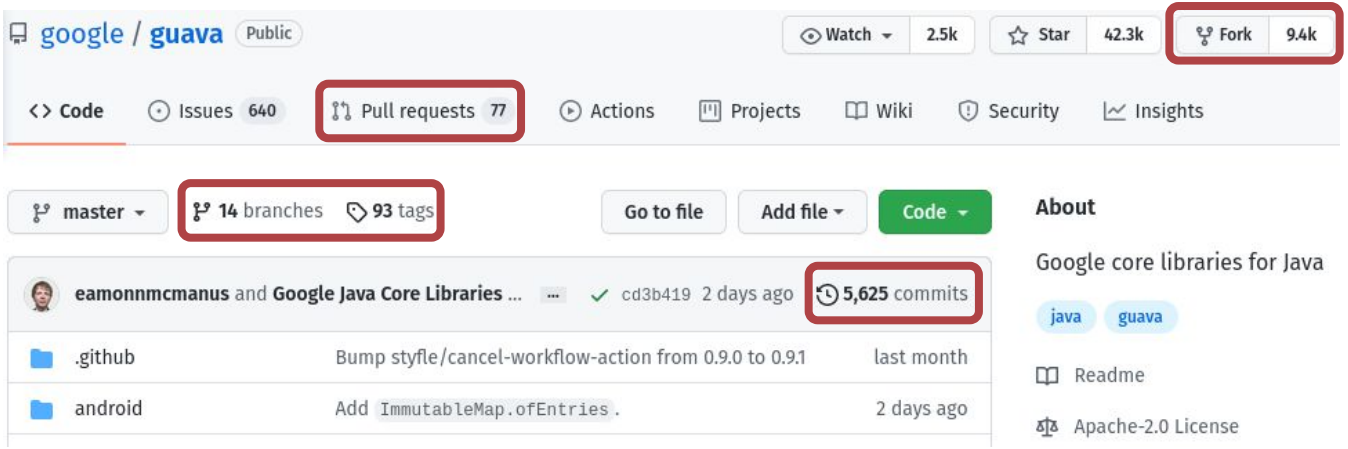


Client–Library Compatibility Testing with API Interaction Snapshots

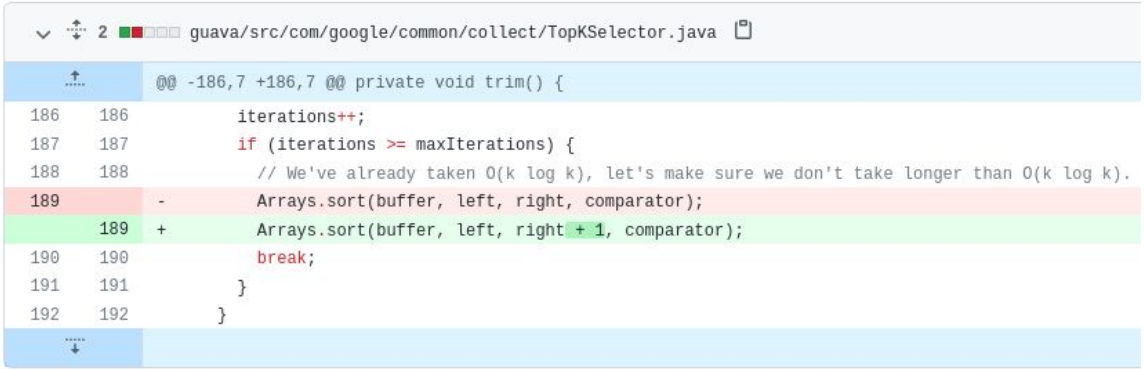
Gustave Monce, Thomas Degueule, Jean-Rémy Falleri, Romain Robbes.

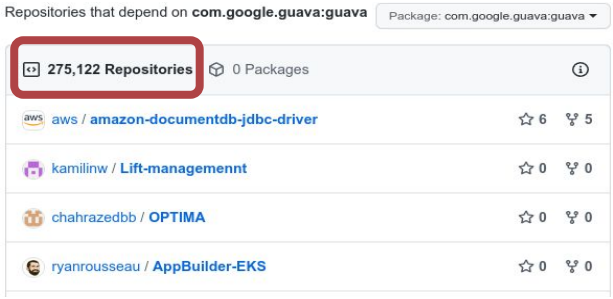
In Proceedings of the 41st IEEE International Conference on Software Maintenance and Evolution, NIER Track (ICSME-NIER 2025). IEEE, 791-796.

Software Libraries



Version	Repository	Usages	Date
30.1.1-jre	Central	2,839	Mar, 2021
30.1.1-android	Central	488	Mar, 2021
30.1-jre	Central	1,611	Dec, 2020
30.1-android	Central	424	Dec, 2020
30.0-jre	Central	1,490	Oct, 2020
30.0-android	Central	359	Oct, 2020
29.0-jre	Central	2,799	Apr, 2020
29.0-android	Central	382	Apr, 2020
28.2-jre	Central	1,785	Jan, 2020
28.2-android	Central	142	Jan, 2020
28.1-jre	Central	1,827	Aug, 2019
28.1-android	Central	80	Aug, 2019
28.0-jre	Central	1,482	Jun, 2019
28.0-android	Central	56	Jun, 2019
27.1-jre	Central	1,442	Mar, 2019





Library Upgrades

- Manually
- Using automated bots/tasks
  - *Can* seemingly report compatibility status
  - Relies on unit tests

Syntactic vs Behavioral Breaking Changes

Syntactic BCs

- Affect API signatures
- Trigger compile/link-time errors
- Extensive literature
- Can be detected statically with good accuracy (e.g., japicmp, RevApi, **Roseau**)

```
- public String  fetch(String productId) {
+ public Product fetch(String productId) {
```

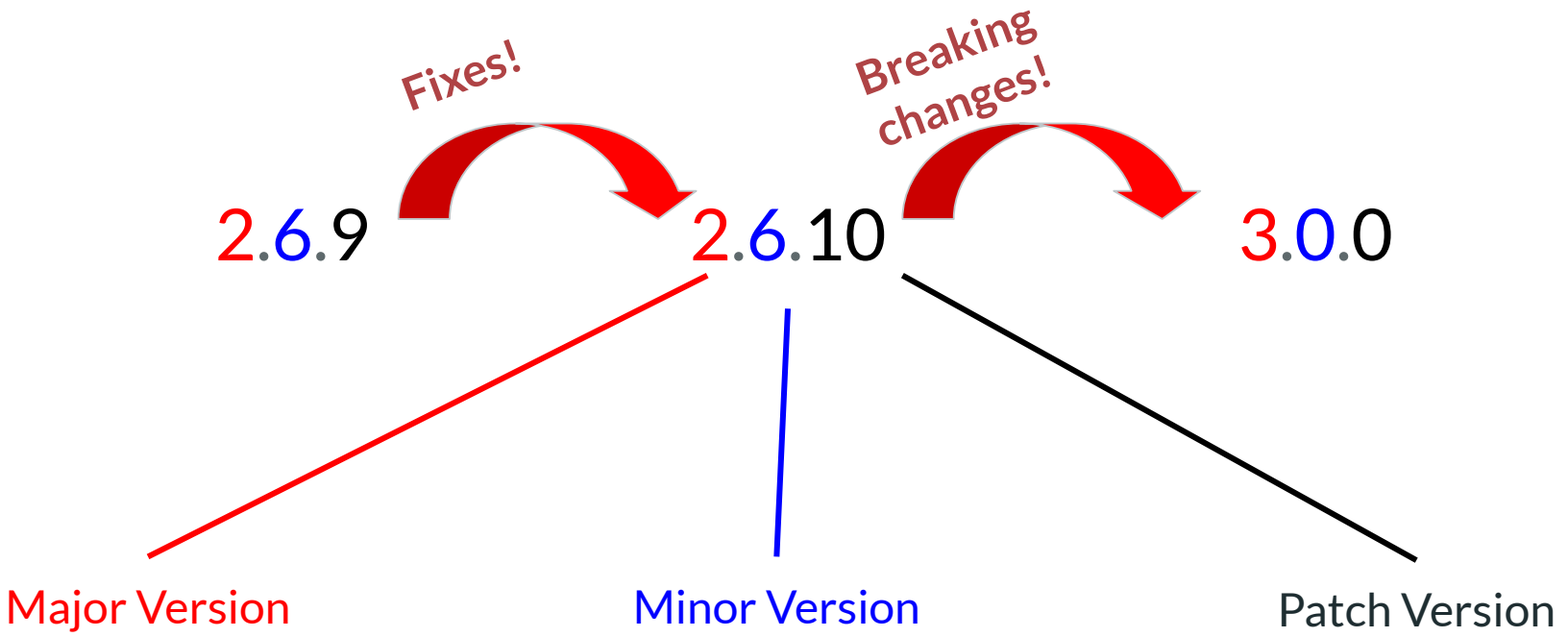
Syntactic vs Behavioral Breaking Changes

Behavioral BCs

- Undecidable
- Manifest as run-time errors, exceptions, etc.
- Hard to estimate impact on client code
- Very few approaches

```
- return Arrays.sort(buf, left, right, comparator)
+ return Arrays.sort(buf, left, right + 1, comparator)
```

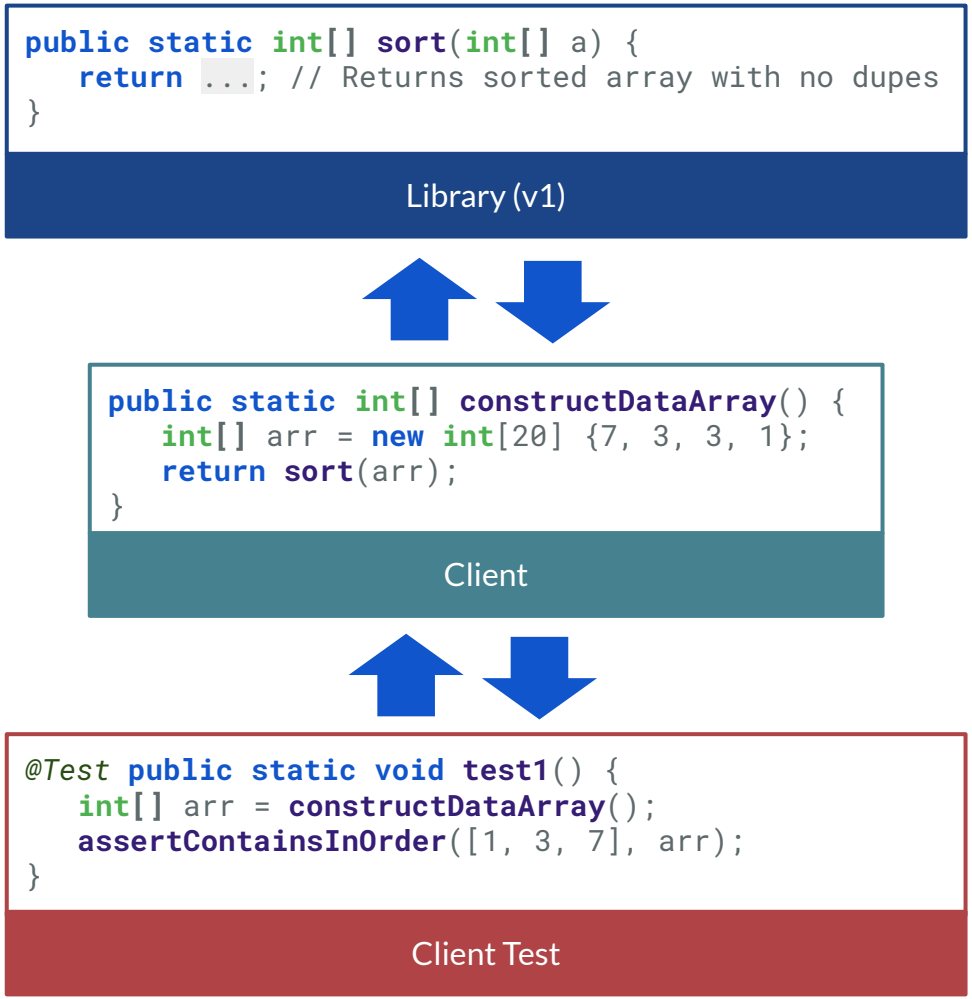
Semantic Versioning



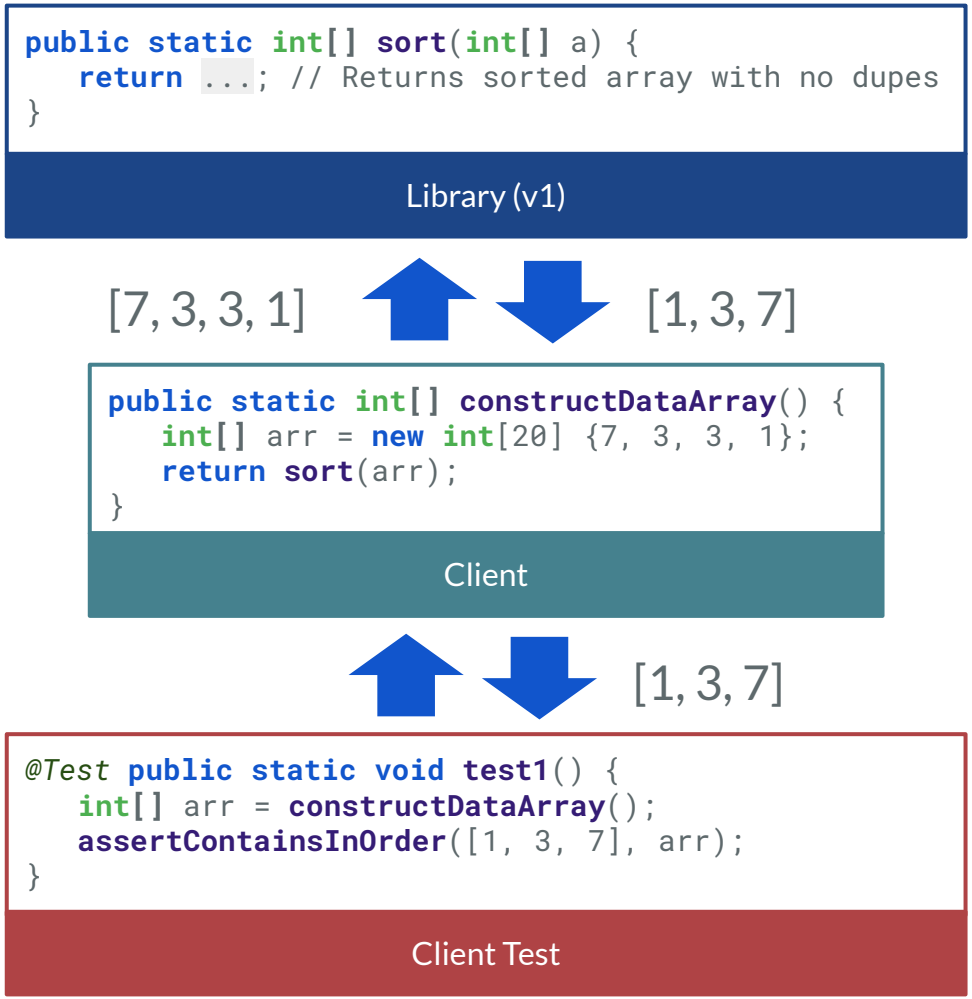
https://semver.org/

Ochoa, et al. "Breaking bad? Semantic versioning and impact of breaking changes in Maven Central: An external and differentiated replication study" (ESE'22)

Approach Overview



Approach Overview



Approach Overview

Assertion result: {1, 3, 7}

✔

```
public static int[] sort(int[] a) {
   return ...; // Returns sorted array with no dupes
}
```

Library (v1)

[7, 3, 3, 1]  [1, 3, 7]

```
public static int[] constructDataArray() {
   int[] arr = new int[20] {7, 3, 3, 1};
   return sort(arr);
}
```

Client

 [1, 3, 7]

```
@Test public static void test1() {
   int[] arr = constructDataArray();
   assertContainsInOrder([1, 3, 7], arr);
}
```

Client Test

Approach Overview

Assertion result:

✔

```
public static int[] sort(int[] a) {
   return ...; // Returns sorted array with dupes
}
```

Library (v2)

```
public static int[] constructDataArray() {
   int[] arr = new int[20] {7, 3, 3, 1};
   return sort(arr);
}
```

Client

```
@Test public static void test1() {
   int[] arr = constructDataArray();
   assertContainsInOrder([1, 3, 7], arr);
}
```

Client Test

Approach Overview

Assertion result:

```
public static int[] sort(int[] a) {
   return ...; // Returns sorted array with dupes
}
```

Library (v2)

[7, 3, 3, 1]  [1, 3, 3, 7]

```
public static int[] constructDataArray() {
   int[] arr = new int[20] {7, 3, 3, 1};
   return sort(arr);
}
```

Client

 [1, 3, 3, 7]

```
@Test public static void test1() {
   int[] arr = constructDataArray();
   assertContainsInOrder([1, 3, 7], arr);
}
```

Client Test

Approach Overview

Assertion result:

✔ ✔

Weak...

```
public static int[] sort(int[] a) {
   return ...; // Returns sorted array with dupes
}
```

Library (v2)

[7, 3, 3, 1] ⇕ [1, 3, 3, 7]

```
public static int[] constructDataArray() {
   int[] arr = new int[20] {7, 3, 3, 1};
   return sort(arr);
}
```

Client

⇕ [1, 3, 3, 7]

```
@Test public static void test1() {
   int[] arr = constructDataArray();
   assertContainsInOrder([1, 3, 7], arr);
}
```

Client Test

Approach: Ideal test

Assertion result:


```
public static int[] sort(int[] a) {
   return ...; // Returns sorted array with dupes
}
```

Library (v2)

[7, 3, 3, 1]

[1, 3, 3, 7]

```
@Test public static void test2() {
   int[] arr = new int[20] {7, 3, 3, 1};
   assertEquals([1, 3, 7], sort(arr));
}
```

Test (Sensitive)

Regression Testing

- Client tests cannot reliably detect Behavioral Breaking Changes†
  - Insufficient coverage
  - Weak assertions
  - Distance to fault
  - Lenient/exception-swallowing code
- Even when detected, the distance between the root cause and the failing assertions severely hurts diagnosis and remediation

† Jayasuriya et al. *"Understanding the impact of APIs behavioral breaking changes on client applications"* (FSE'24)
† Hejderup et al. *"Can we trust tests to automate dependency updates? A case study of Java projects"* (JSS'22)
† Gyori et al. *"Evaluating regression test selection opportunities in a very large open-source ecosystem"* (ISSRE'18)

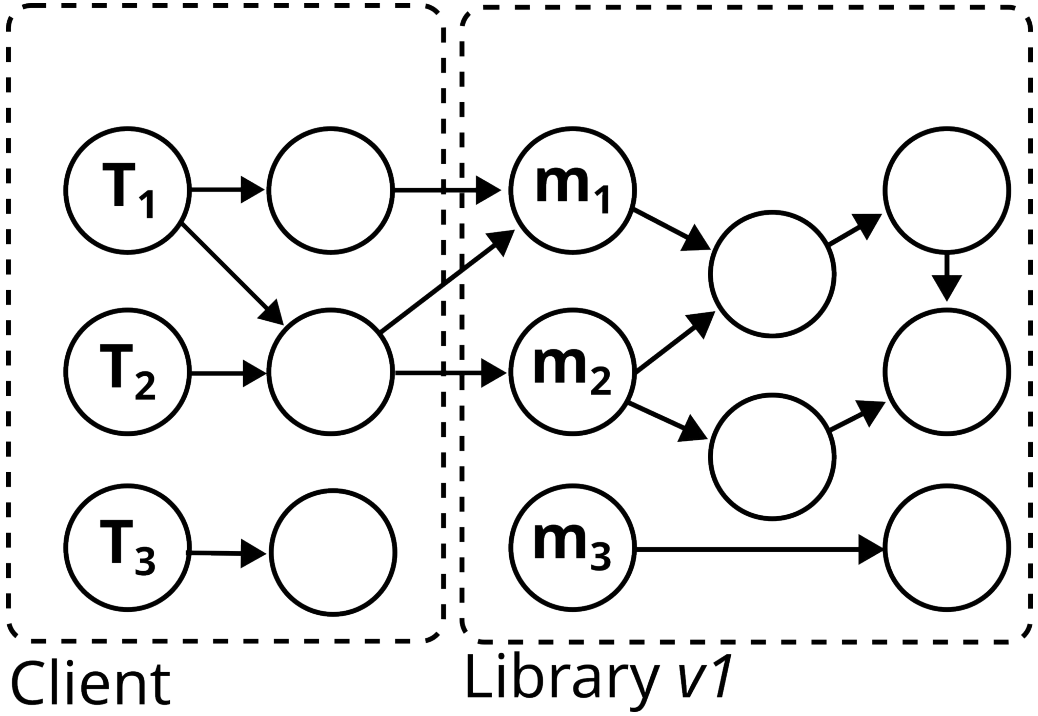
Existing approaches

New!

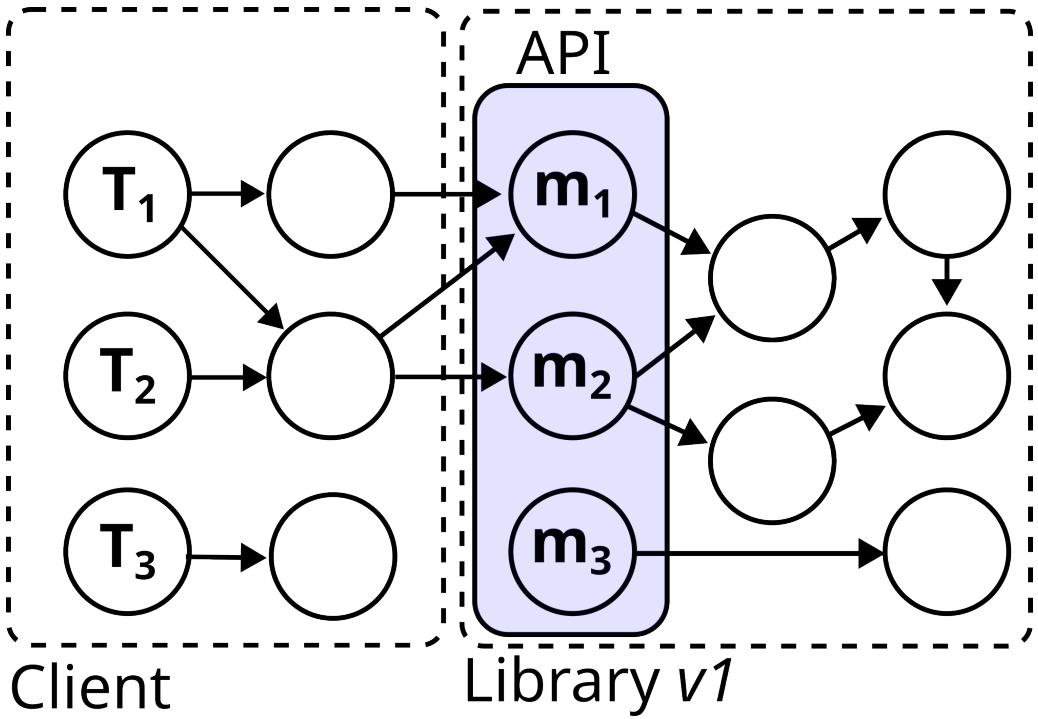
	DeBBI	SemBid	Uppdatera	CompCheck	Gilesi
Whole API	x	x			
Client usage subset			x	x	x
Client crowd-sourced knowledge	x			x	
Client specific		x	x		x
Dynamic change analysis		x			x
Static change analysis	x		x	x	

DeBBI: Chen, et al. "Taming behavioral backward incompatibilities via cross-project testing and analysis". (ICSE20)
SemBid: Zhang, et al. "Has My Release Disobeyed Semantic Versioning? Static Detection Based on Semantic Differencing". (ASE22)
Uppdatera: Hejderup, et al. "Can we trust tests to automate dependency updates? A case study of Java Projects". (JSS22)
CompCheck: Zhu, et al. "Client-Specific Upgrade Compatibility Checking via Knowledge-Guided Discovery". (TOSEM23)

API Interaction Snapshots

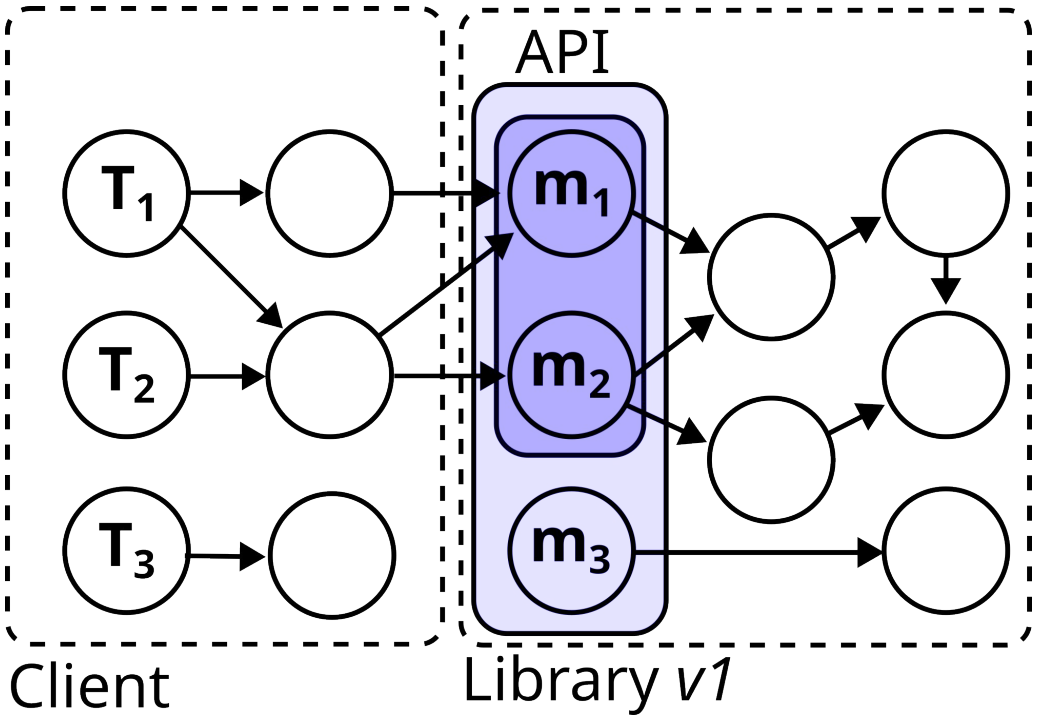


API Interaction Snapshots



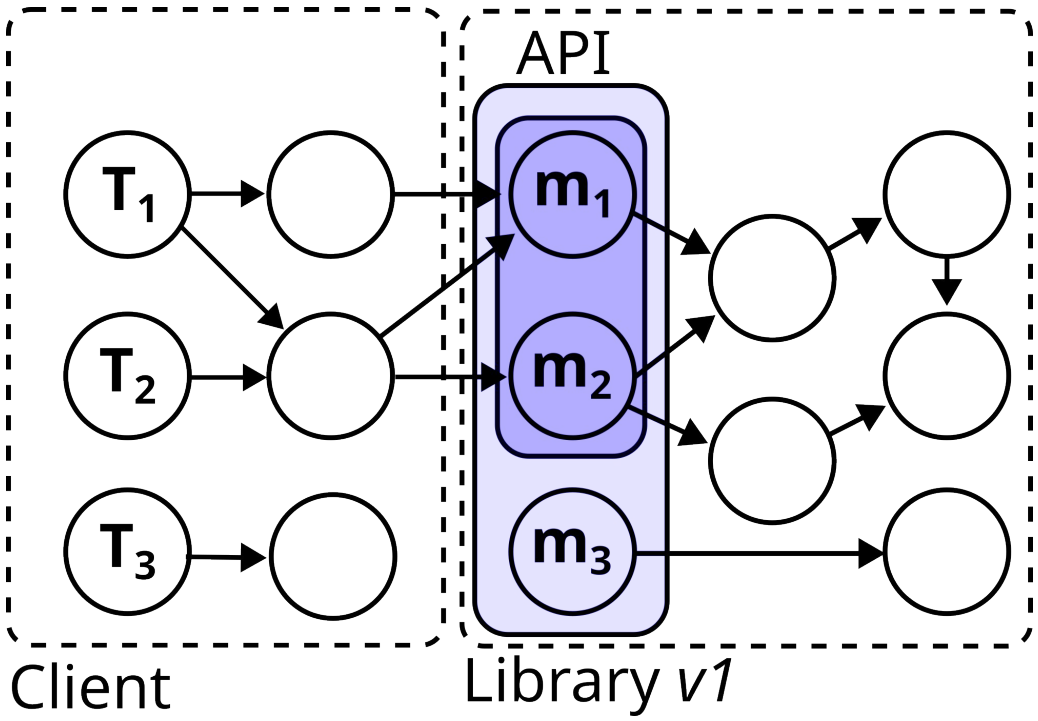
1. Identify the library's API

API Interaction Snapshots



1. Identify the library's API
2. Identify the library's footprint

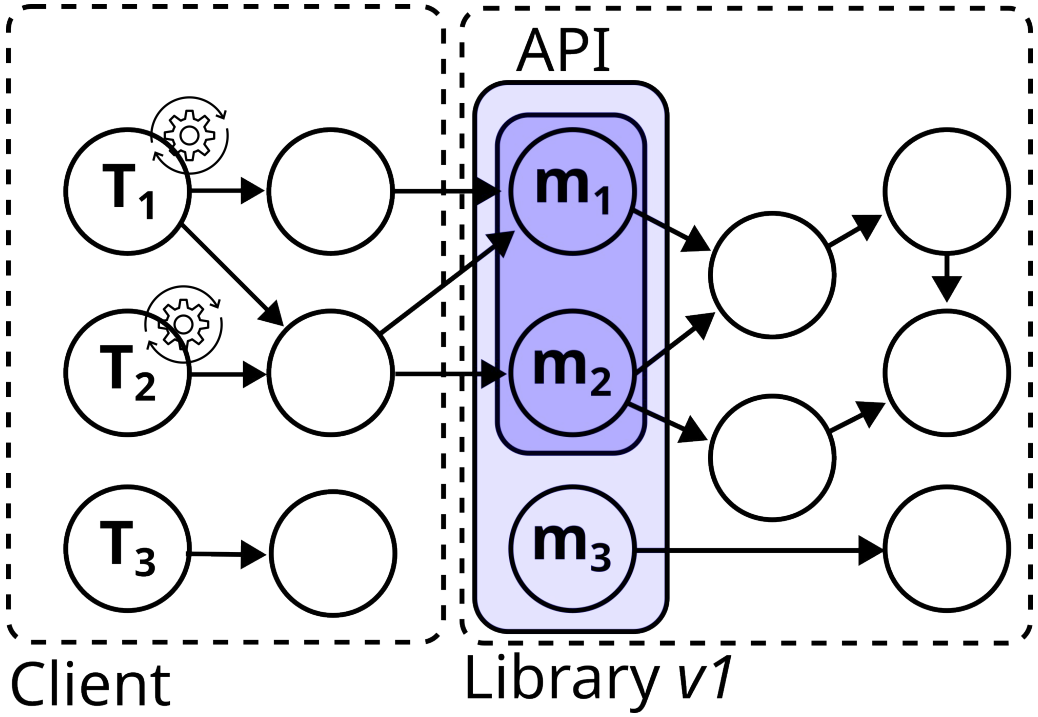
API Interaction Snapshots



1. Identify the library's API
2. Identify the library's footprint
3. Instrument the footprint with probes, recording:
   a. Invoked method
   b. Receiving object
   c. In-parameter values
   d. Out-values/exceptions

$$I_1 = \langle m_1, o_1, \langle p_1, \ldots, p_n \rangle, r_1 \rangle$$

API Interaction Snapshots



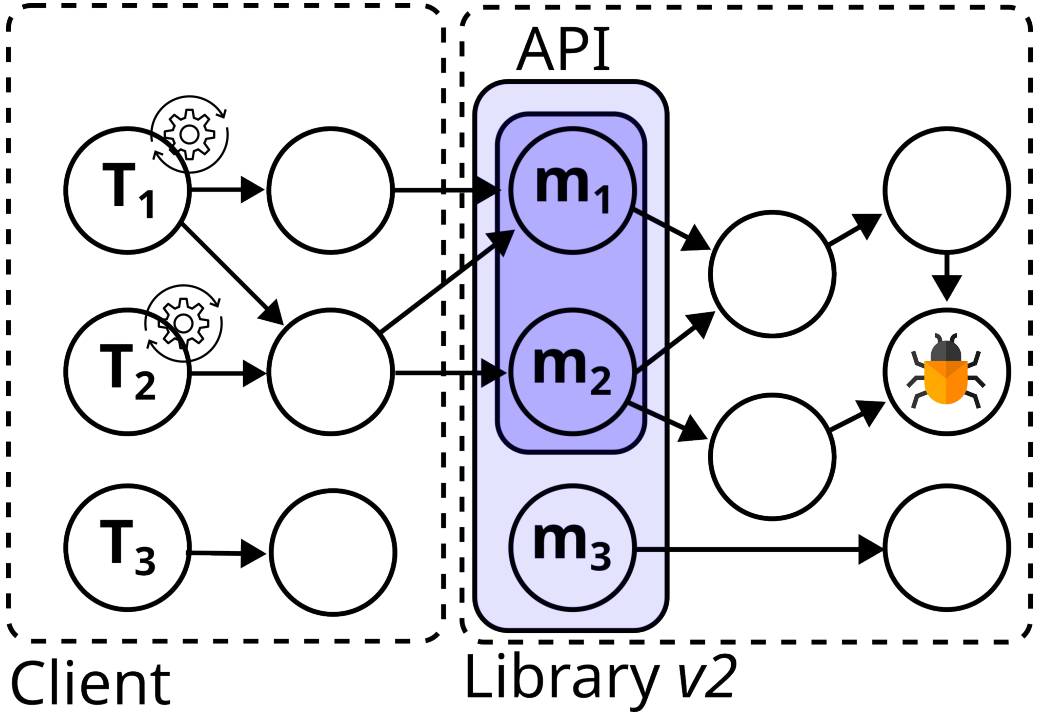
$$I_1 = \langle m_1, o_1, \langle p_1, \ldots, p_n \rangle, r_1 \rangle$$

T1's Snapshot (v1)

$$I_1 = \langle m_1, o_2, \langle p_1, \ldots, p_n \rangle, r_2 \rangle$$
$$I_2 = \langle m_2, o_2, \langle p_2, \ldots, p_n \rangle, r_3 \rangle$$

T2's Snapshot (v1)

API Interaction Snapshots



$$I_1 = \langle m_1, o_1, \langle p_1, \ldots, p_n \rangle, r_1 \rangle$$

T1's Snapshot (v2)

$$I_1 = \langle m_1, o_2, \langle p_1, \ldots, p_n \rangle, r_2 \rangle$$
$$I_2 = \langle m_2, o_2, \langle p_2, \ldots, p_n \rangle, r_3 \rangle$$

T2's Snapshot (v2)

Comparing API Interaction Snapshots

```
Test1_Snapshot1 = (
  ⟨StrTokenizer.<init>, ∅, “apple,banana”, o1⟩,
  ⟨setDelimiter, o1, “,”, ∅⟩,
  ⟨getTokenList, o1, ∅, ArrayList(“apple”, “banana”)⟩
)
```

An example test’s snapshot on commons-text:1.9

```
Test1_Snapshot2 = (
  ⟨StrTokenizer.<init>, ∅, “apple,banana”, o1⟩,
  ⟨setDelimiter, o1, “,”, ∅⟩,
  ⟨getTokenList, o1, ∅, Arrays$List(“apple”, “banana”)⟩
)
```

An example test’s snapshot on commons-text:1.10

- Pairwise comparison of old and new test snapshots
- Any discrepancy in
  - Protocol
  - Types
  - Exchanged values
  - etc.
- Signals a behavioral perturbation that *might* be a BBC and warrants investigation

An example BBC introduced in `commons-text` commit `2d1ab7`. Identified in `TEXT-219` and later fixed in commit `f9846b`.

Gilesi: API Interaction Snapshots for Java

github.com/
alien-tools/gilesi

```
public boolean performTask(E e) throws Exception {
    probeEntry(this, e);
    ...
    probeExit(this, e, false);
    return false;
}
```

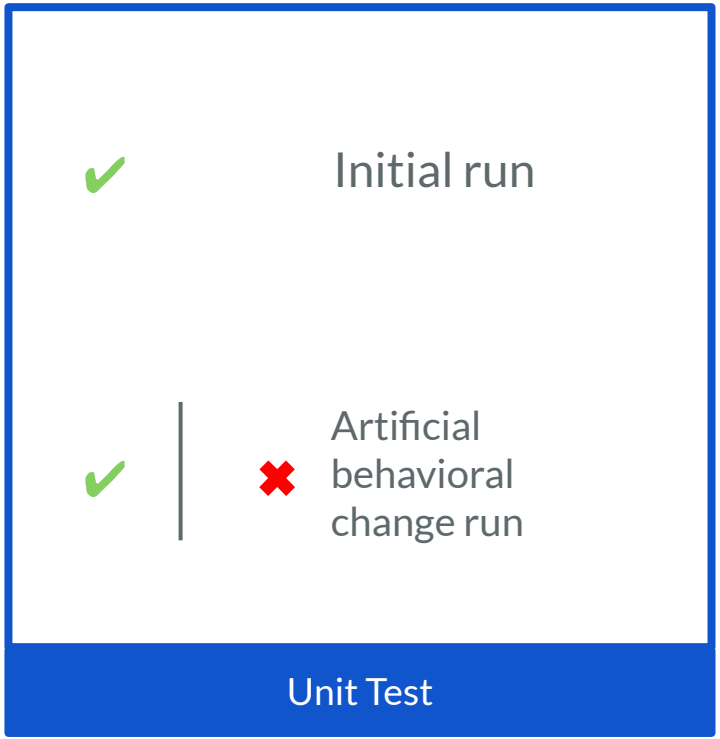
ByteBuddy-based agent

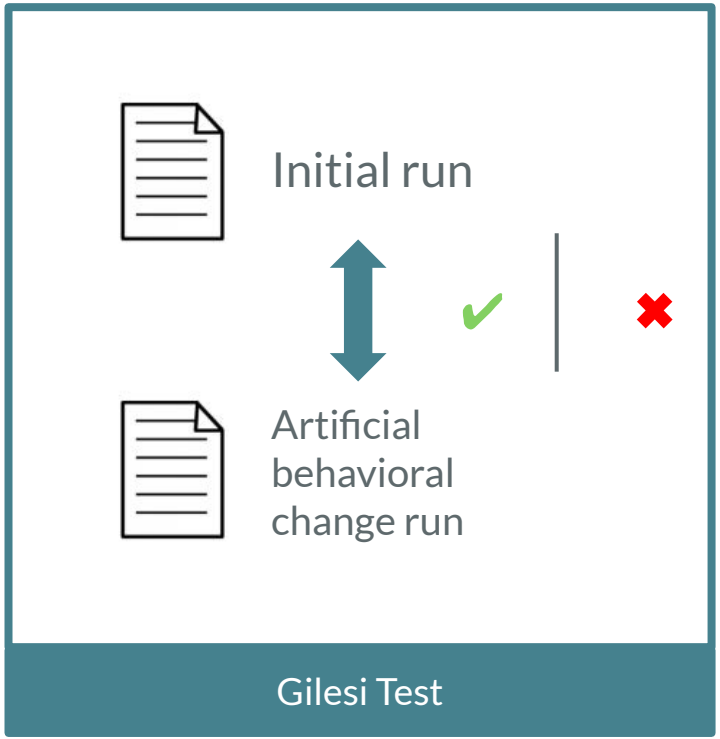
Snapshot

+

- Stack trace
- Method/Constructor signature
- Class/Type information

Evaluation: Methodology





Evaluation: Extreme Mutation as Behavioral Breaking Change

```java
public static boolean performTask() {
        new ArrayList<Integer>();
        lst.add(0);
        lst.add(2);
        lst.add(6 * 2);
        lst.get(0);

        return true;
}
```

```java
public static boolean performTask() {
        return false;
}
```

Extreme mutation operators used as part of pseudo-tested method detection, Niedermayr, et al. *"Will My Tests Tell Me If I Break This Code?"* (CSED'16)

Evaluation: Projects & Metrics

zenodo.org/records/
16411966

	jsoup	commons-lang3
# Clients	6	21
# Mutants	52	106

Results: commons-lang3

Client	Mutants	Client kills	Gilesi kills
ch.zizka.csvcruncher-csv-cruncher	5	5	5
com.amihaiemil.devops-comdor	3	2	3
com.cognifide.aemrules-sonar-aemrules-plugin	11	11	11
com.github.bingoohuang-delayqueue	7	7	7
com.github.bingoohuang-excel2javabeans	10	10	10
com.github.cschoell-junit-dynamicsuite	4	4	4
com.github.daanvdh.javaforger-JavaForger	5	5	5
com.github.davidcarboni-restolino	5	5	5
com.github.jonpereiradev-jfile-reader	5	5	5
com.javacreed.examples-gson-typeadapter-example	1	1	1
com.json.comparison-comparison-core	1	1	1
com.omertron-API-OMDB	2	1	2
com.yanceyzhang.tools-wx-chatbot	1	1	1
de.pentabyte-springfox-enum-plugin	1	1	1
edu.anadolu-FrequencyDistributionAnalysis	3	3	3
io.prowave-chargify-webhook-java	1	1	1
org.cyclopsgroup-jmxterm	14	12	14
org.jvnet.hudson.plugins-warnings	3	3	3
org.zalando-jzon	1	1	1
org.zeroturnaround-zt-process-killer	1	0	0
com.github.jknack.handlebars	22	21	22
Total	106	100	105

	Client's unit tests	Gilesi's snapshot tests
Mutation score	94.34%	99.06%

- Client tests miss even extreme mutations
- Surviving mutants are
  - i) equivalent for output
  - or ii) unobservable I/O side effects
- Realistic/subtle mutations more likely to be caught by Gilesi

Results: commons-lang3

Client	Mutants	Client kills	Gilesi kills
ch.zizka.csvcruncher-csv-cruncher	5	5	5
com.amihaiemil.devops-comdor	3	2	3
com.cognifide.aemrules-sonar-aemrules-plugin	11	11	11
com.github.bingoohuang-delayqueue	7	7	7
com.github.bingoohuang-excel2javabeans	10	10	10
com.github.cschoell-junit-dynamicsuite	4	4	4
com.github.daanvdh.javaforger-JavaForger	5	5	5
com.github.davidcarboni-restolino	5	5	5
com.github.jonpereiradev-jfile-reader	5	5	5
com.javacreed.examples-gson-typeadapter-example	1	1	1
com.json.comparison-comparison-core	1	1	1
com.omertron-API-OMDB	2	1	2
com.yanceyzhang.tools-wx-chatbot	1	1	1
de.pentabyte-springfox-enum-plugin	1	1	1
edu.anadolu-FrequencyDistributionAnalysis	3	3	3
io.prowave-chargify-webhook-java	1	1	1
org.cyclopsgroup-jmxterm	14	12	14
org.jvnet.hudson.plugins-warnings	3	3	3
org.zalando-jzon	1	1	1
org.zeroturnaround-zt-process-killer	1	0	0
com.github.jknack.handlebars	22	21	22
Total	106	100	105

- Client tests miss even extreme mutations
- Surviving mutants are
  - i) equivalent for output
  - or ii) unobservable I/O side effects
- Realistic/subtle mutations more likely to be caught by Gilesi

	Client's unit tests	Gilesi's snapshot tests
Mutation score	94.34%	99.06%

Results: JSoup

Client	Mutants	Client kills	Gilesi kills
com.codeu.chatapp-chatapp	2	2	2
com.github.beothorn-webGrude	2	2	2
com.github.dhorions-boxable	7	5	7
im.nll.data-extractor	10	10	10
me.chyxion-table-to-xls	5	2	5
us.codecraft-xsoup	26	20	20
Total	52	41	46

- Client tests miss even extreme mutations
- Surviving mutants are
  - i) equivalent for output
  - or ii) unobservable I/O side effects
- Realistic/subtle mutations more likely to be caught by Gilesi

	Client's unit tests	Gilesi's snapshot tests
Mutation score	78.85%	88.46%

Results: JSoup

Client	Mutants	Client kills	Gilesi kills
com.codeu.chatapp-chatapp	2	2	2
com.github.beothorn-webGrude	2	2	2
com.github.dhorions-boxable	7	5	7
im.nll.data-extractor	10	10	10
me.chyxion-table-to-xls	5	2	5
us.codecraft-xsoup	26	20	20
Total	52	41	46

- Client tests miss even extreme mutations
- Surviving mutants are
  - i) equivalent for output
  - or ii) unobservable I/O side effects
- Realistic/subtle mutations more likely to be caught by Gilesi

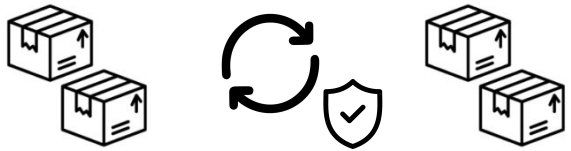
	Client's unit tests	Gilesi's snapshot tests
Mutation score	78.85%	88.46%

Gilesi

- Snapshots embody the **exact expectations** of a particular client towards a specific version of a library
- Leveraging the execution paths offered by client tests' with stronger snapshot-based assertions increases the **sensitivity** of the tests and the **observability** of behavioral changes
- Asserting directly at the API–client boundary **eases diagnosis and remediation**

✔ As a client maintainer, can I safely update my dependencies?

→ Client-Library Compatibility Testing with API Interaction Snapshots (***with Gilesi***)

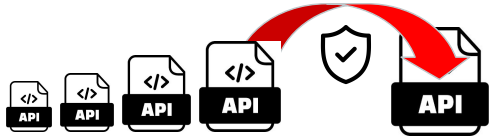


Conclusion & Perspectives

Conclusion: Problem statements

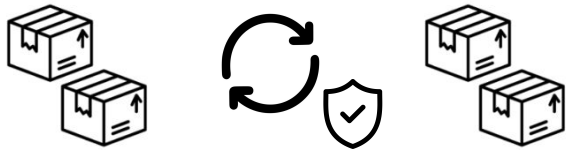
✔ As a library maintainer, can I safely evolve my API?

- Lightweight Syntactic API Usage Analysis ***with UCov***



✔ As a client maintainer, can I safely update my dependencies?

- Client–Library Compatibility Testing with API Interaction Snapshots (***with Gilesi***)



Perspectives: Safe API Evolution

Beyond API Usage

- SUM as a design and validation tool (Prescriptive SUM)
  - Documentation/Sample extraction from client's uses
- Enhancing and extending existing research based on usage analysis with interaction types (e.g. *Hyrum's law*†)
- SUM/SUFs to aid Syntactic Breaking Change detection approaches

† Harrand et al. *"API beauty is in the eye of the clients: 2.2 million Maven dependencies reveal the spectrum of client–API usages"* (JSS'22)

Perspectives: Safe dependency updates

Beyond Compatibility Checking

- Test generation/amplification for greater coverage and observability for the client
- Generating unit tests from API snapshots for the library

Conclusion

https://github.com/alien-tools/ucov
UCov's Git Repository
https://zenodo.org/doi/10.5281/zenodo.10571867
UCov's Exploratory Case Study Artifacts

https://github.com/alien-tools/gilesi
Gilesi's Git Repository
https://zenodo.org/doi/10.5281/zenodo.16411966
Gilesi's Case Study Artifacts

Lightweight Syntactic API Usage Analysis with UCov
Gustave Monce, Thomas Couturou, Yasmine Hamdaoui, Thomas Degueule, and Jean-Rémy Falleri.
In Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension (ICPC '24). Association for Computing Machinery, New York, NY, USA, 426–437.

Client–Library Compatibility Testing with API Interaction Snapshots
Gustave Monce, Thomas Degueule, Jean-Rémy Falleri, Romain Robbes.
In Proceedings of the 41st IEEE International Conference on Software Maintenance and Evolution, NIER Track (ICSME-NIER 2025). IEEE, 791-796.

Thesis manuscript

This slidedeck

us A ge driven software
Li brary
E vo-
lutio n ANR ALIEN (ANR--21--CE25--0007)