When to let the Developer Guide: Trade-offs between Open and Guided Test Amplification

Carolin Brandt
Danyao Wang
Andy Zaidman

TU Delft

Test Amplification

Generate xUnit tests by mutating existing tests

Existing Test → → Amplified Tests

```
@Test
void text_escape () {
    A a = new A ("text");
    B b = a.escape()
    assertEquals (b.toString(), "text");
}
```

Test Amplification

Generate xUnit tests by mutating existing tests

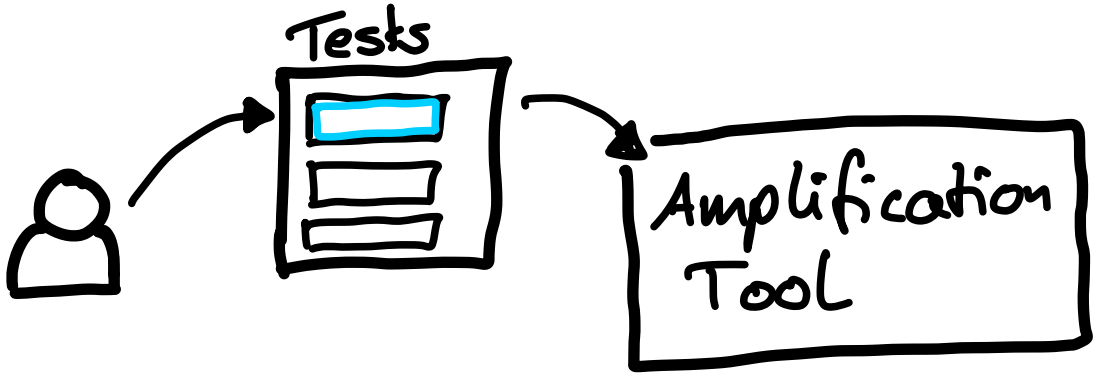
Existing Test → Mutate Input → Replace Assertions → Improve Coverage? → Amplified Tests

```
@Test
void text_escape () {
    A a = new A("text"); → "text $//"
    B b = a.escape()
    assertEquals(b.toString(), "text"); → "text\$\//"
}
```

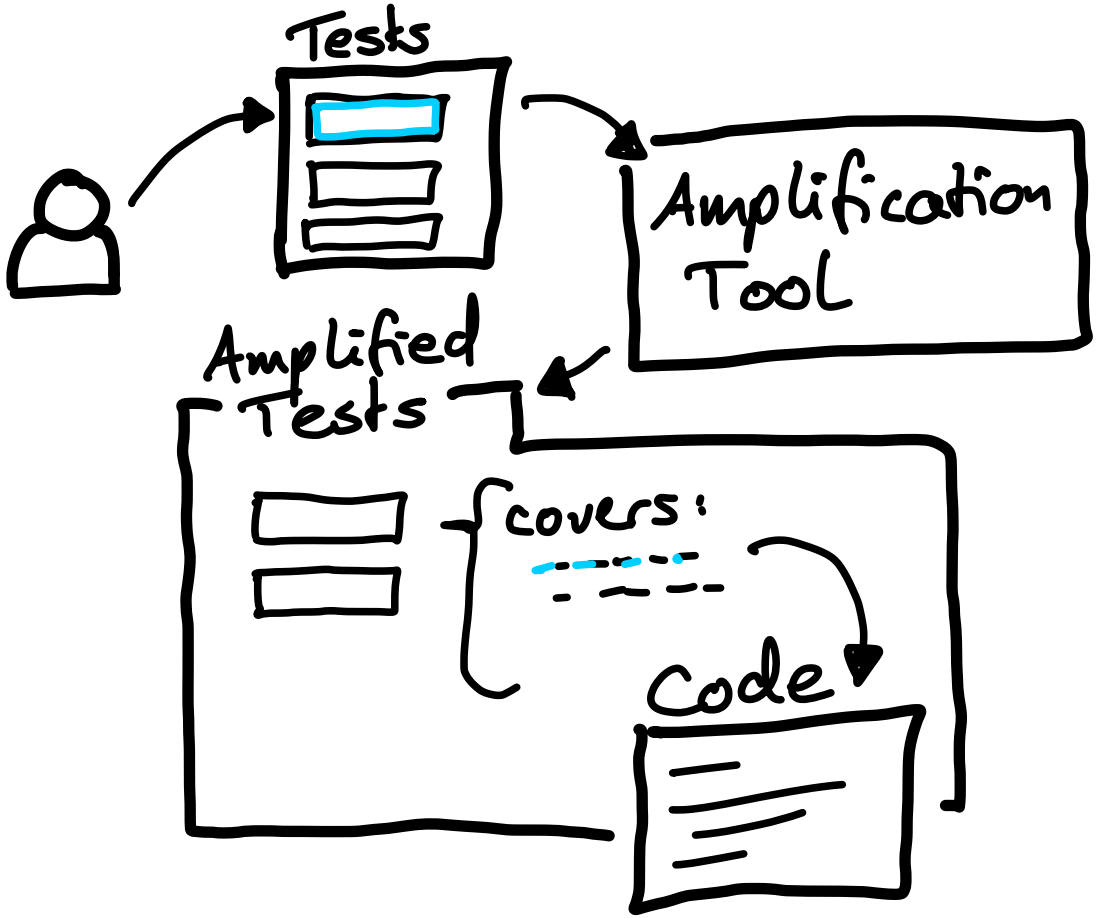
Open Test Amplification

Looking in all directions for tests

User Interaction

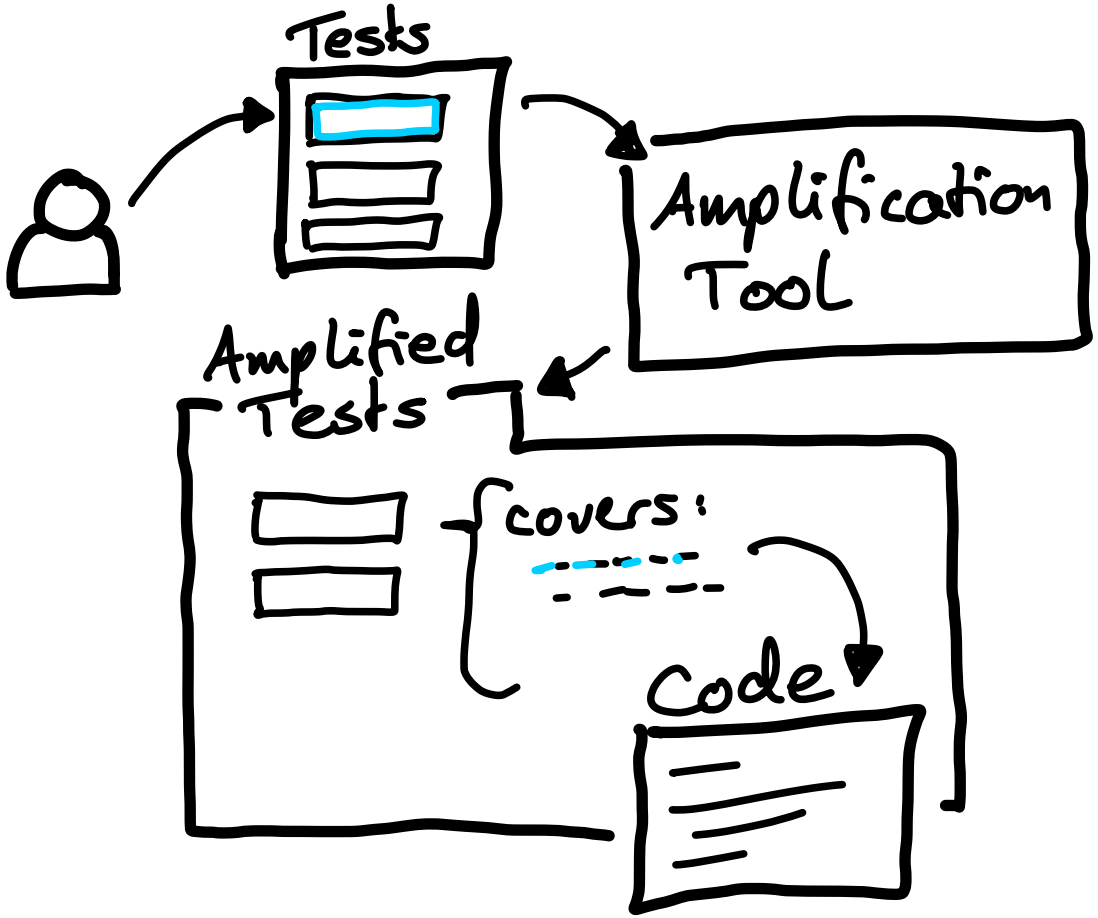


User Interaction



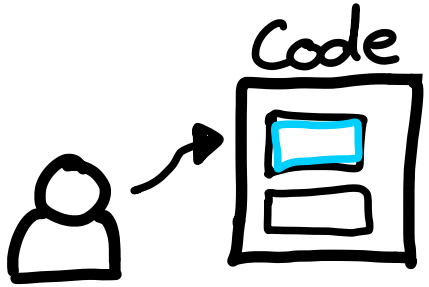
Open Test Amplification:
Looking in all Directions for Tests

User Interaction

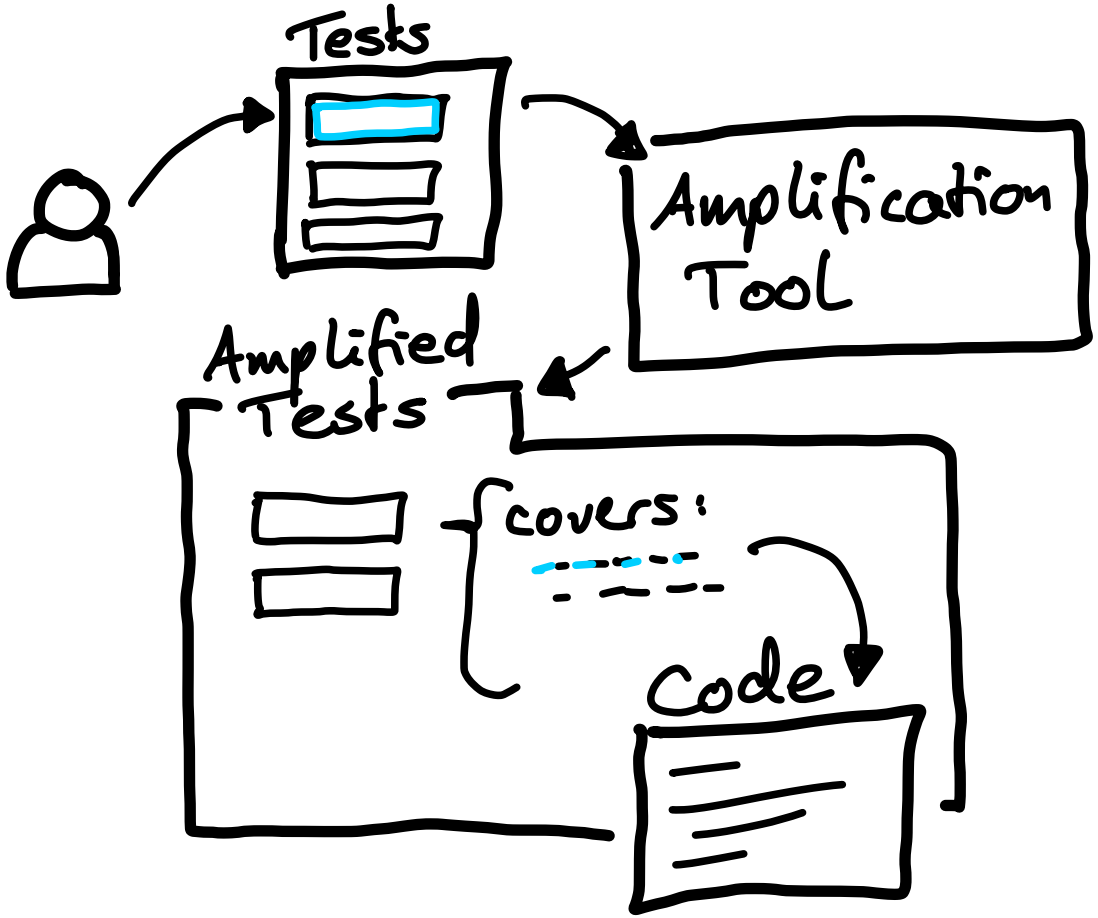


Open Test Amplification:
Looking in all Directions for Tests

What if we let the developer point us to what they want to test?

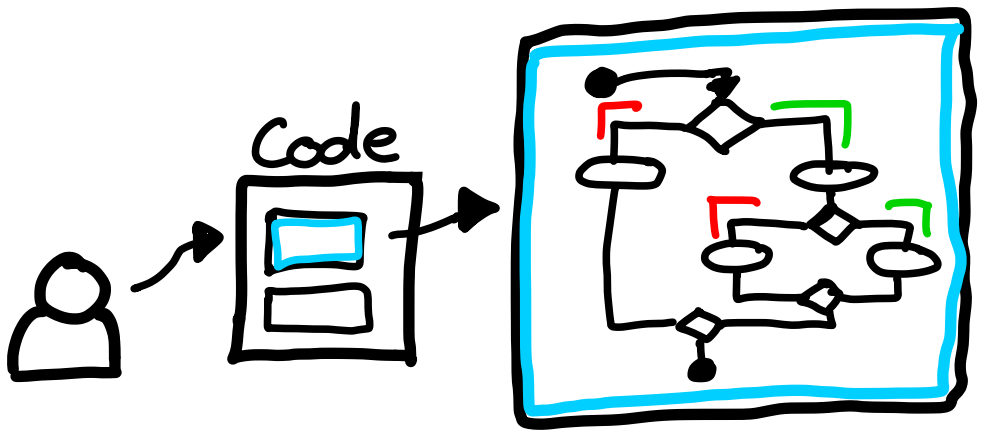


User Interaction

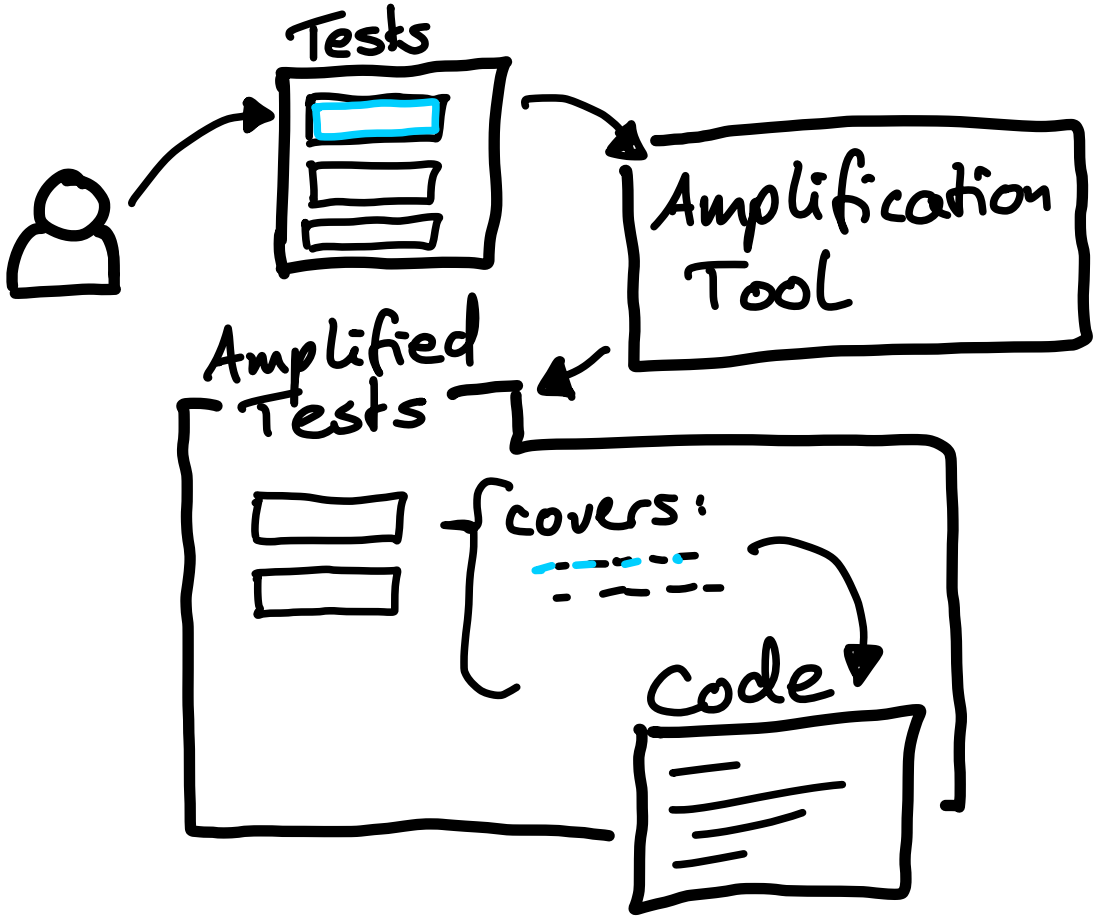


Open Test Amplification:
Looking in all Directions for Tests

What if we let the developer point us to what they want to test?

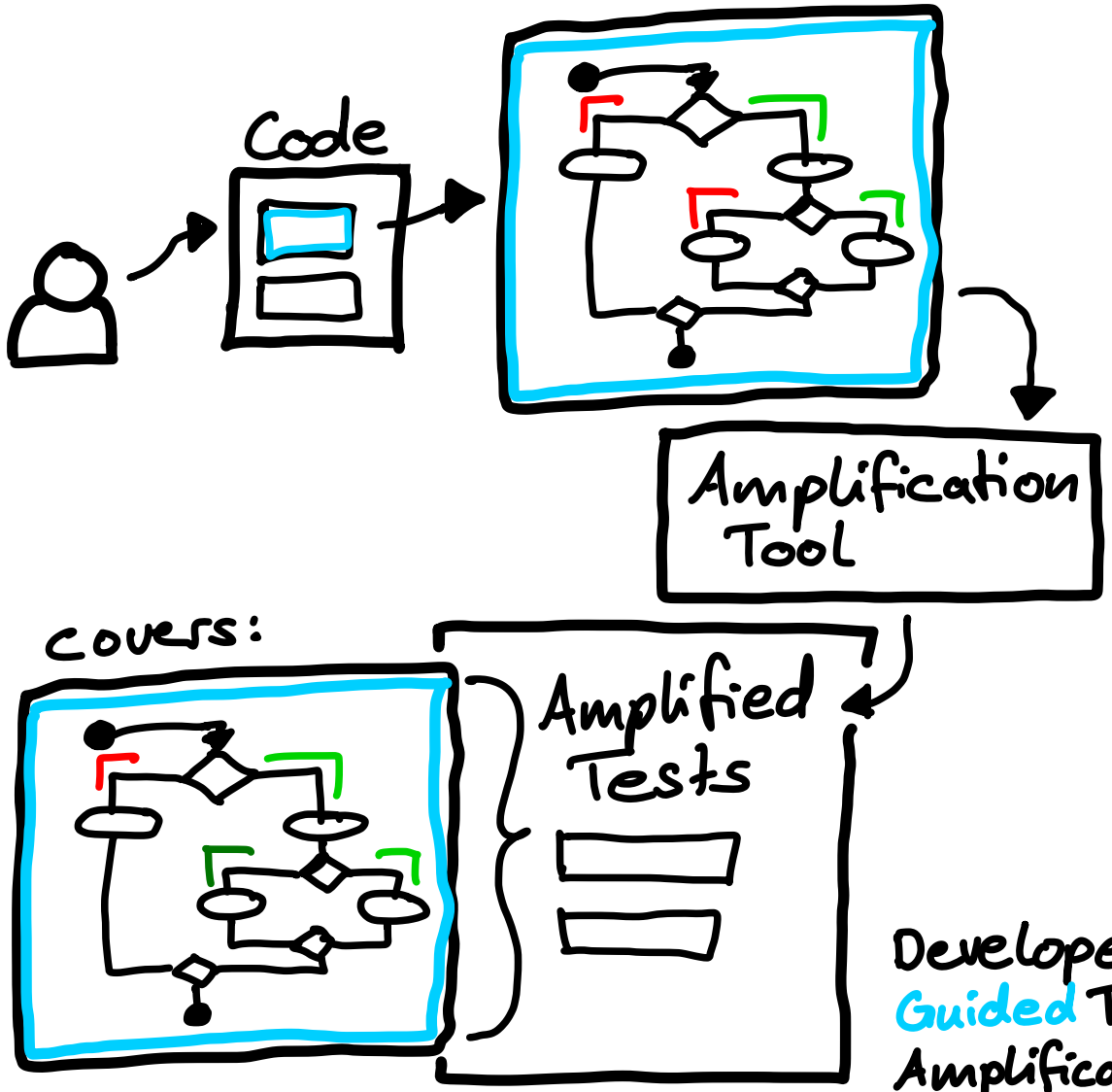


User Interaction



Open Test Amplification:
Looking in all Directions for Tests

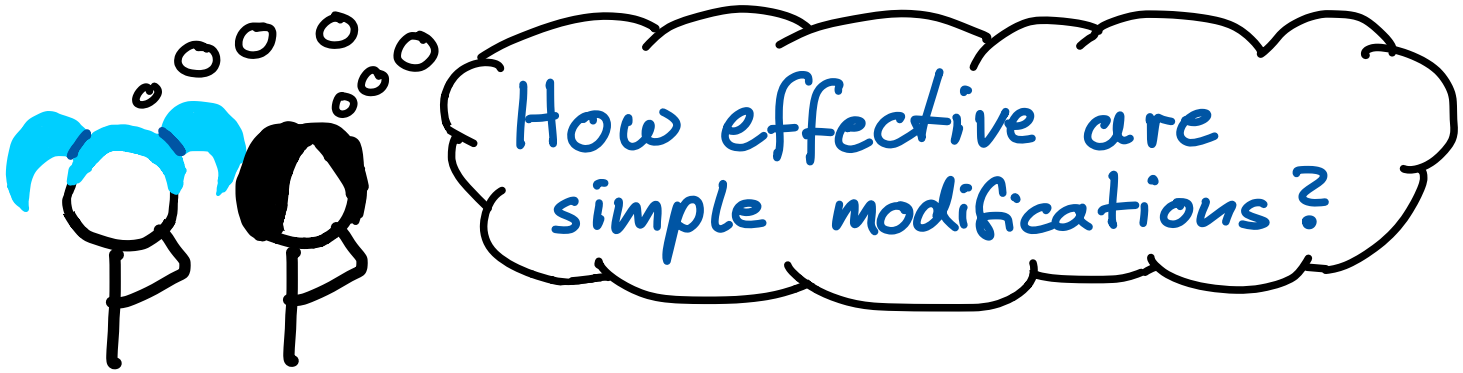
What if we let the developer point us to what they want to test?



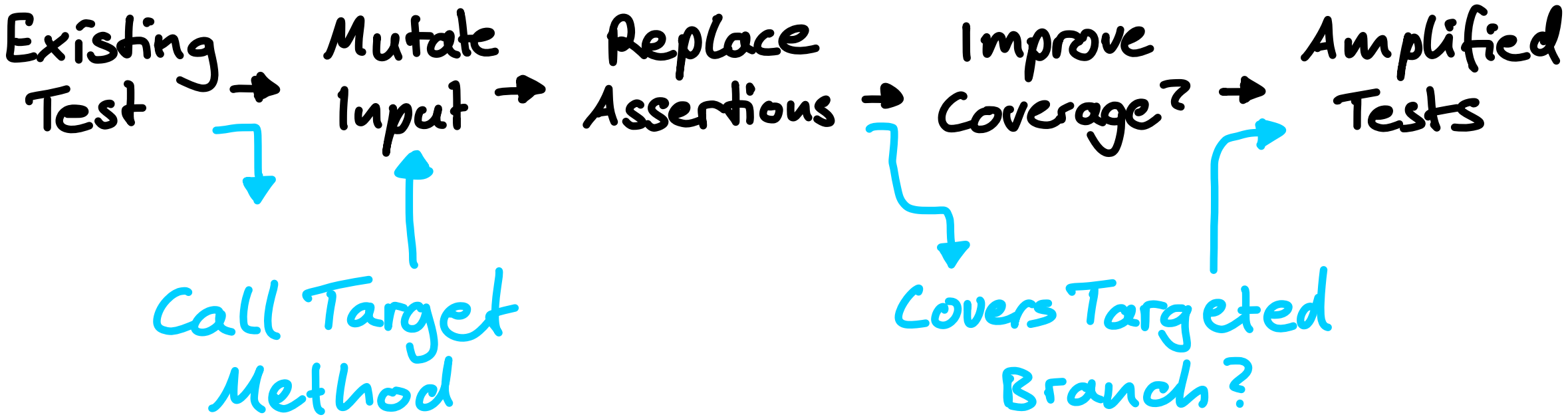
Developer Guided Test Amplification

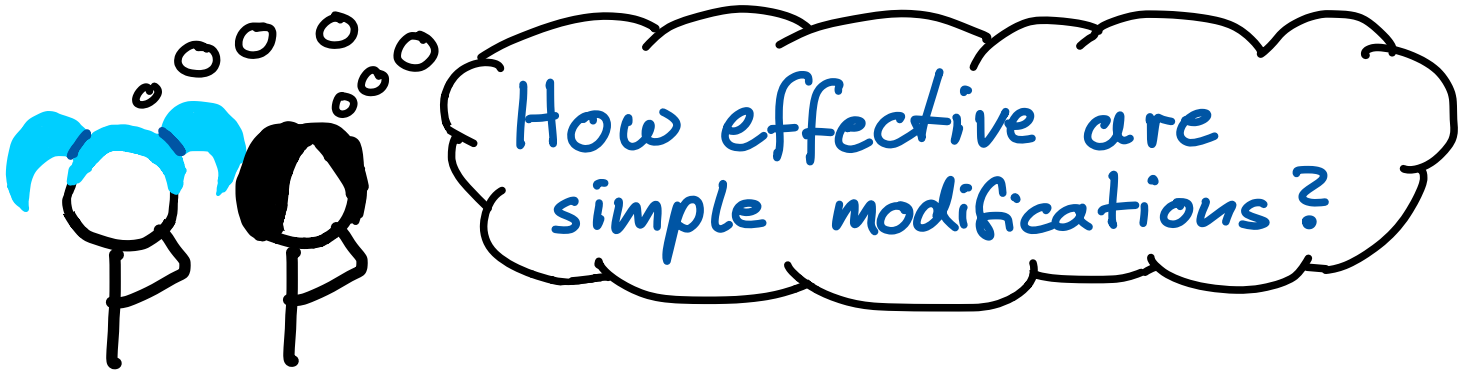
Guided Test Amplification

Existing Test → Mutate Input → Replace Assertions → Improve Coverage? → Amplified Tests



Guided Test Amplification





Technical Evaluation

sample 100 branches each from 2 projects

Technical Evaluation

sample 100 branches each from 2 projects

Sampled branches covered in total

	Javapoet	Stream-lib
Open	23%	35%
Guided	32%	41%

Main culprit:
correct initialization of objects under test

to hit targeted branch

Technical Evaluation

sample 100 branches each from 2 projects

Sampled branches covered in total

	Javapoet	Stream-lib
Open	23%	35%
Guided	32%	41%

Tests covering target branch per run

	Javapoet	Stream-lib
Open	24%	45%
Guided	70%	70%

Main culprit:
correct initialization of objects under test

to hit targeted branch

User Study Evaluation

Generate Tests with Open + Guided Test Amplification with IntelliJ Plugin

12 participants in 4 groups, 2 classes, crossover

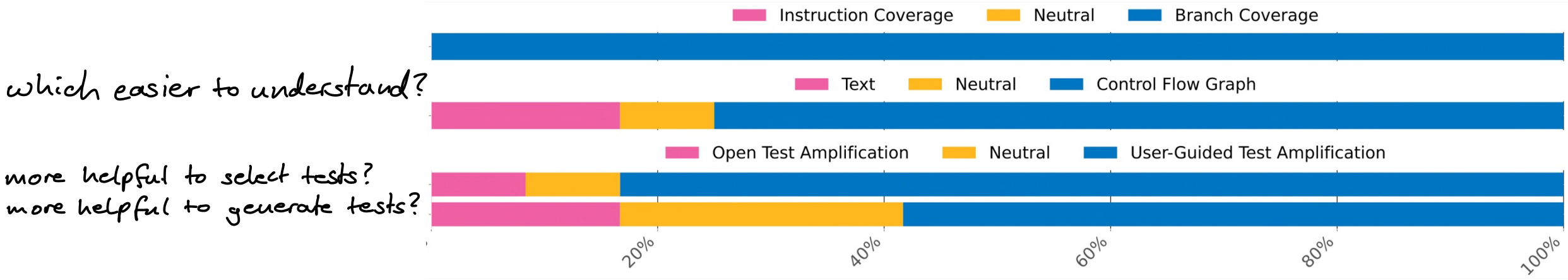
Semi-structured Interviews

User Study Evaluation

Generate Tests with Open + Guided Test Amplification with IntelliJ Plugin

12 participants in 4 groups, 2 classes, crossover

Semi-structured Interviews



Guided is not necessarily better...

Guided is not necessarily better...

Trade-offs: Guided vs. Open Test Amplification

Guided is not necessarily better...

Trade-offs: Guided vs. Open Test Amplification

Fits use case	Writing code & wanting tests for it	Improving test suite

Guided is not necessarily better...

Trade-offs: Guided vs. Open Test Amplification

Fits use case	Writing code & wanting tests for it	Improving test suite
Understand Coverage	In target method in detail	Across whole project

Guided is not necessarily better...

Trade-offs: Guided vs. Open Test Amplification

Fits use case	Writing code & wanting tests for it	Improving test suite
Understand Coverage	In target method in detail	Across whole project
Expectation of receiving tests	Might disappoint if target branch cannot be covered	Only proposes tests it can provide

Guided is not necessarily better...

Trade-offs: Guided vs. Open Test Amplification

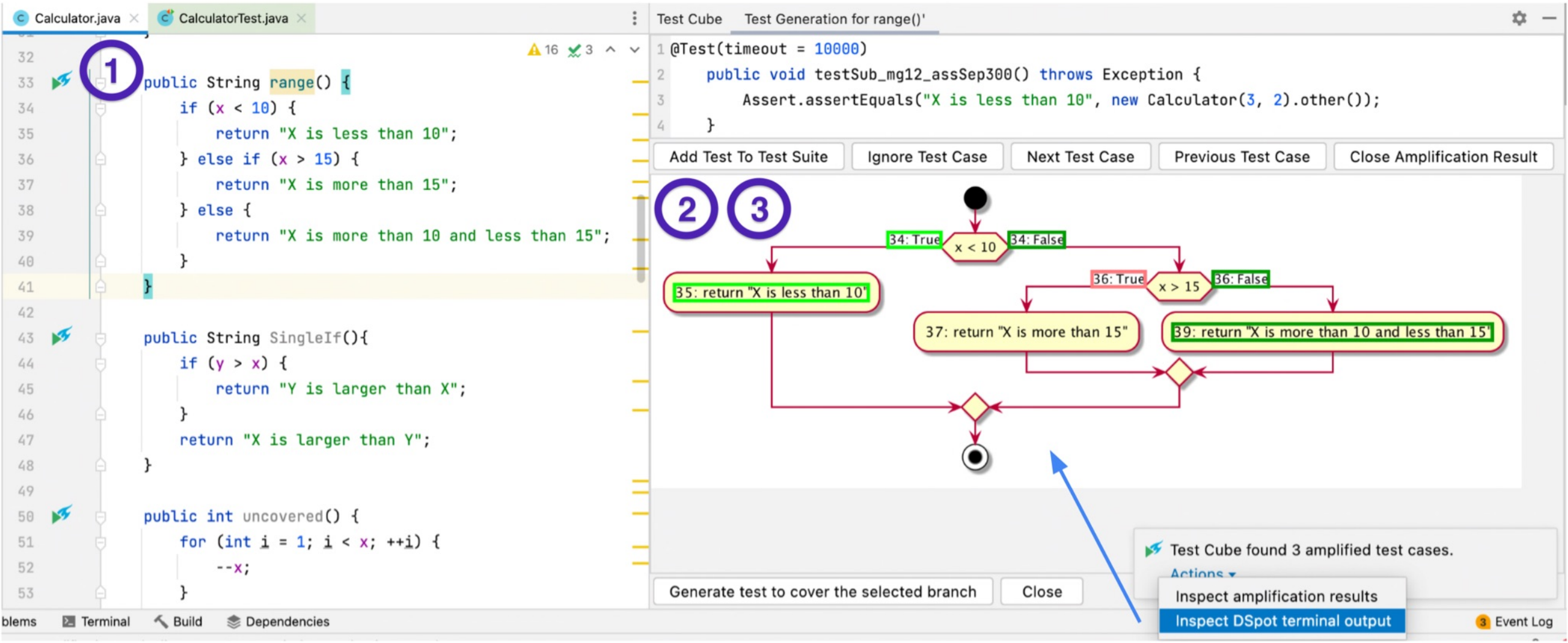
Fits use case	Writing code & wanting tests for it	Improving test suite
Understand Coverage	In target method in detail	Across whole project
Expectation of receiving tests	Might disappoint if target branch cannot be covered	Only proposes tests it can provide
Runtime efficiency	More effective at providing tests for targeted method	Larger variety of tests for whole class

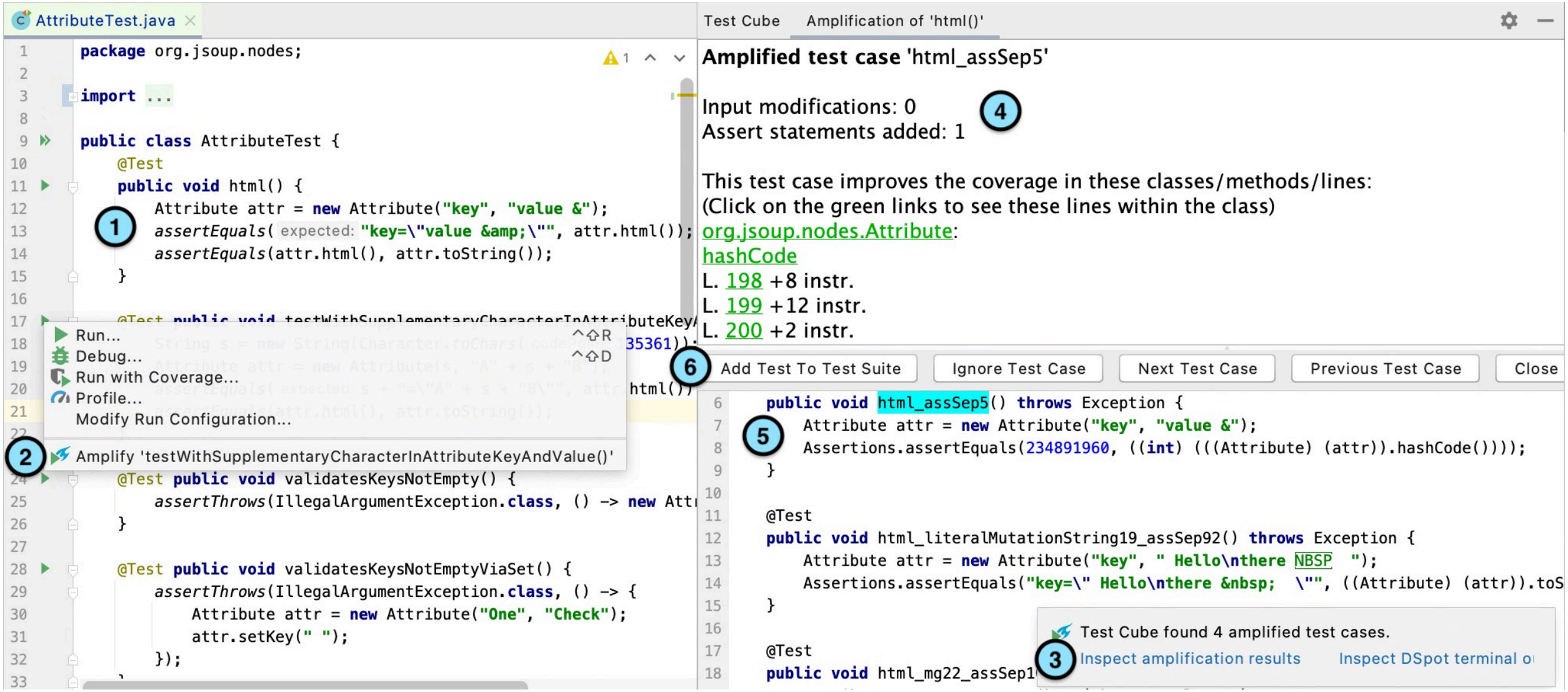
Trade-offs: Guided vs. Open Test Amplification

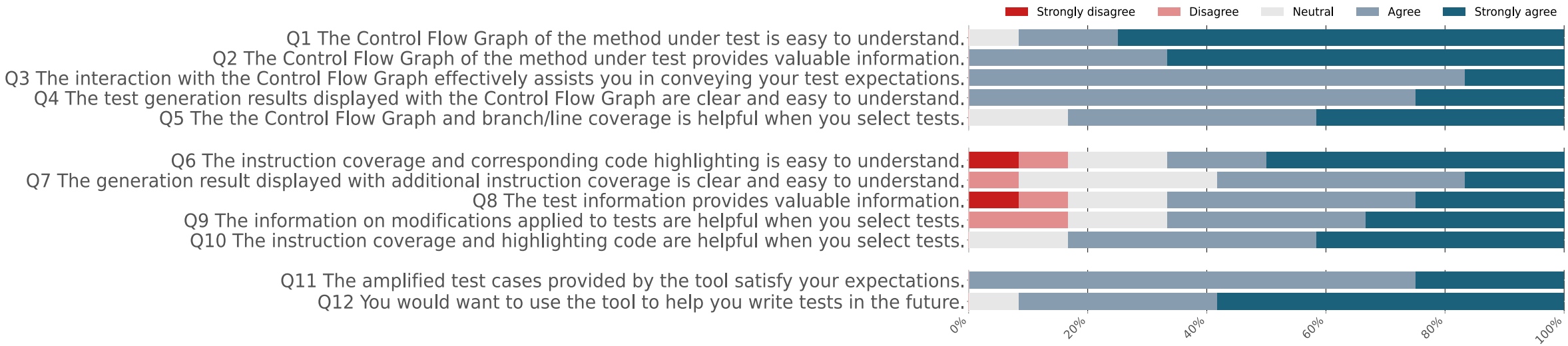
Fits use case	Writing code & wanting tests for it	Improving test suite
Understand Coverage	In target method in detail	Across whole project
Expectation of receiving tests	Might disappoint if target branch cannot be covered	Only proposes tests it can provide
Runtime efficiency	More effective at providing tests for targeted method	Larger variety of tests for whole class

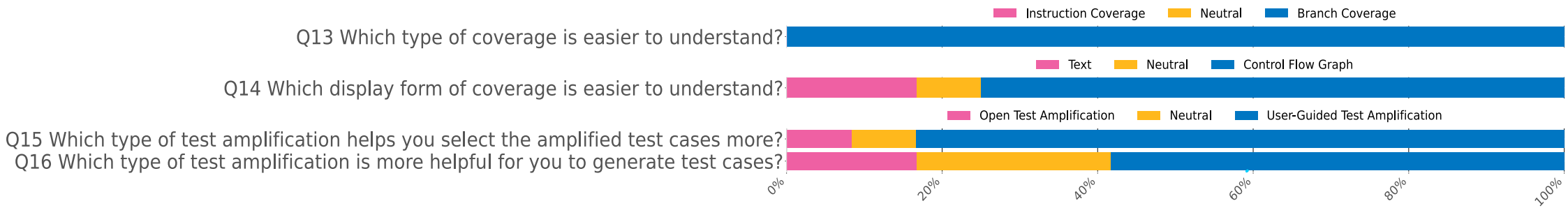
When to let the Developer Guide: Trade-offs Between Open and Guided Test Amplification

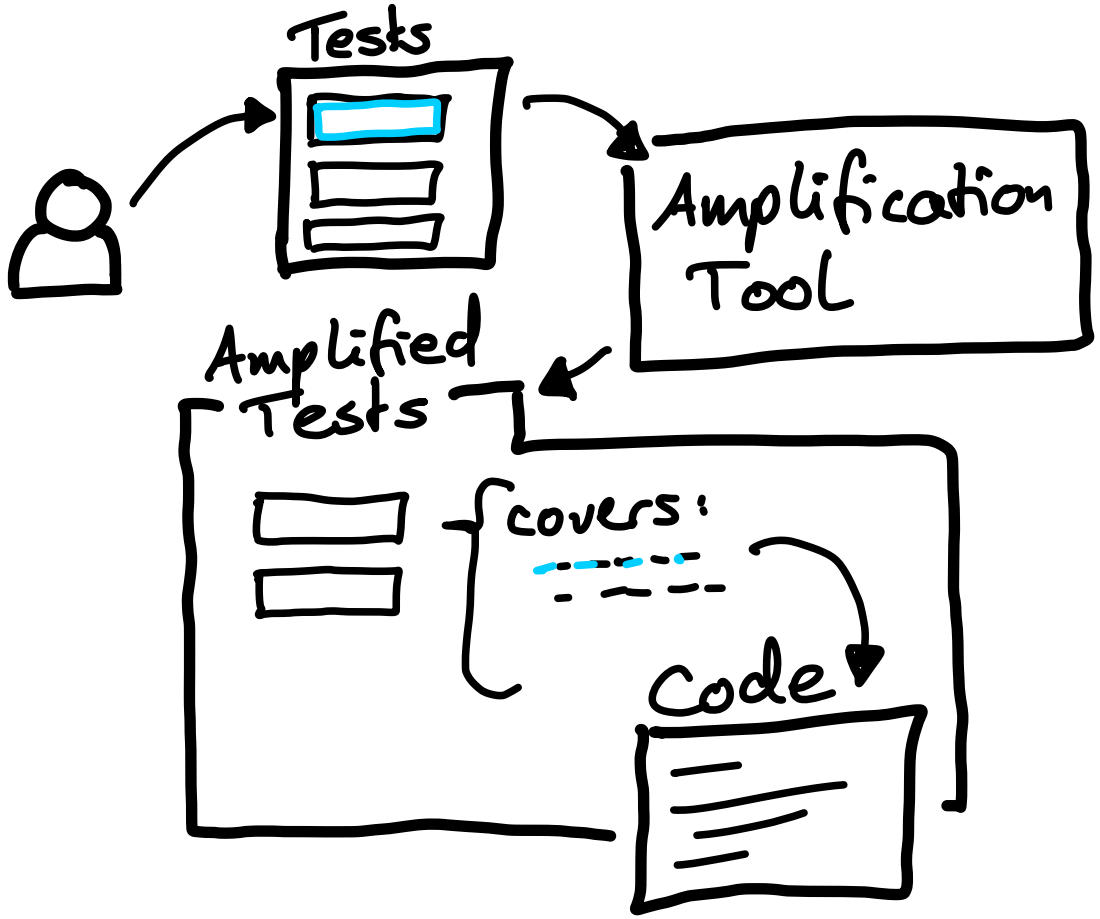
Carolin Brandt, Danyao Wang, Andy Zaidman TU Delft

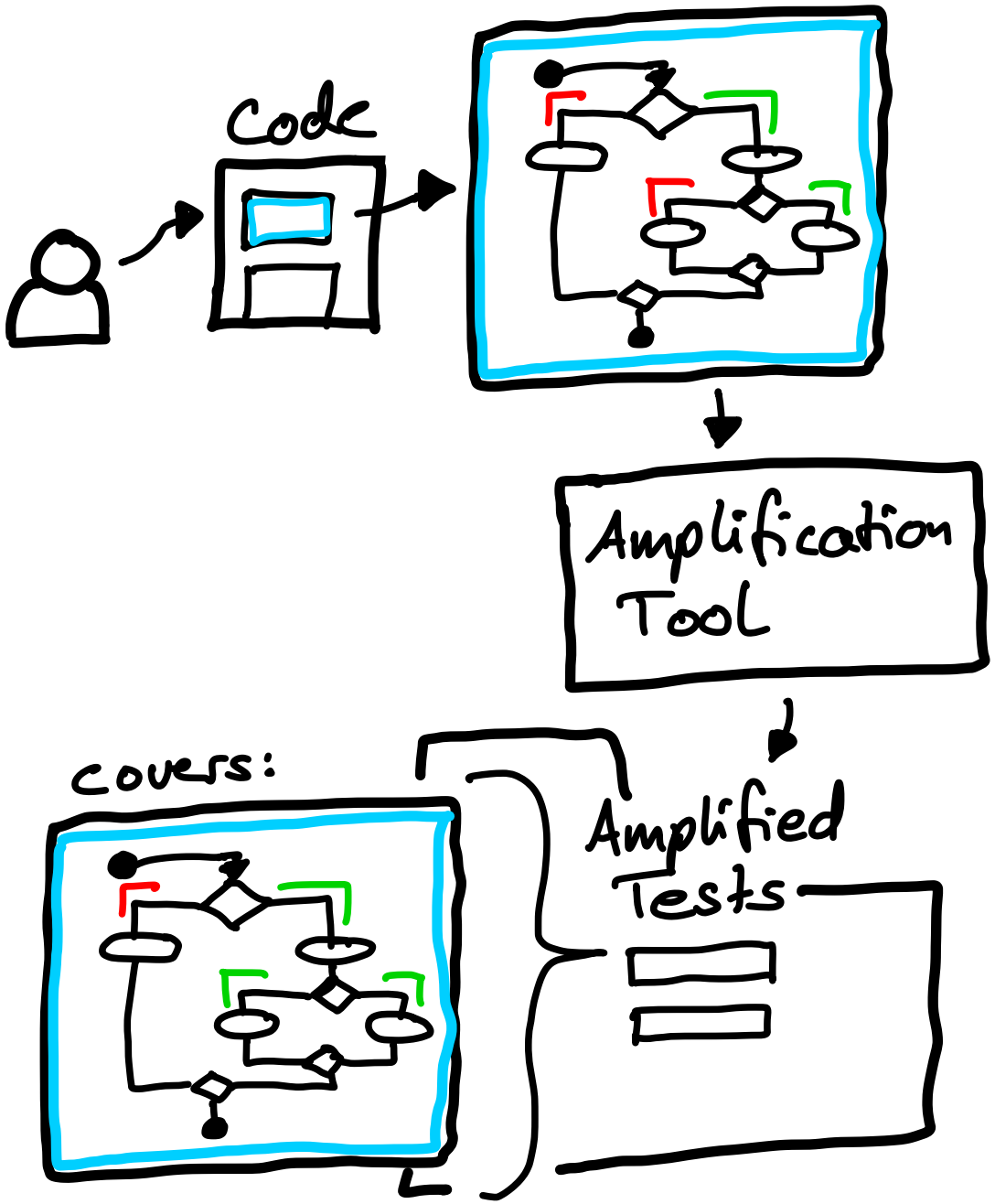

Calculator.java
CalculatorTest.java
1
public String range() {
if (x < 10) {
return "X is less than 10";
} else if (x > 15) {
return "X is more than 15";
} else {
return "X is more than 10 and less than 15";
}
}
public String SingleIf(){
if (y > x) {
return "Y is larger than X";
}
return "X is larger than Y";
}
public int uncovered() {
for (int i = 1; i < x; ++i) {
--x;
}
Test Cube
Test Generation for range()'
@Test(timeout = 10000)
public void testSub_mg12_assSep300() throws Exception {
Assert.assertEquals("X is less than 10", new Calculator(3, 2).other());
}
Add Test To Test Suite
Ignore Test Case
Next Test Case
Previous Test Case
Close Amplification Result
2
3
34: True
x < 10
34: False
35: return "X is less than 10"
36: True
x > 15
36: False
37: return "X is more than 15"
39: return "X is more than 10 and less than 15"
Test Cube found 3 amplified test cases.
Actions
Generate test to cover the selected branch
Close
Inspect amplification results
Inspect DSpot terminal output
Terminal
Build
Dependencies
Event Log


AttributeTest.java
package org.jsoup.nodes;
import ...
public class AttributeTest {
@Test
public void html() {
Attribute attr = new Attribute("key", "value &");
assertEquals(expected: "key=\"value &\"", attr.html());
assertEquals(attr.html(), attr.toString());
}
Run... ^⇧R
Debug... ^⇧D
Run with Coverage...
Profile...
Modify Run Configuration...
Amplify 'testWithSupplementaryCharacterInAttributeKeyAndValue()'
@Test public void validatesKeysNotEmpty() {
assertThrows(IllegalArgumentException.class, () -> new Att
}
@Test public void validatesKeysNotEmptyViaSet() {
assertThrows(IllegalArgumentException.class, () -> {
Attribute attr = new Attribute("One", "Check");
attr.setKey(" ");
});
1
2
Test Cube
Amplification of 'html()'
Amplified test case 'html_assSep5'
Input modifications: 0
Assert statements added: 1
4
This test case improves the coverage in these classes/methods/lines:
(Click on the green links to see these lines within the class)
org.jsoup.nodes.Attribute:
hashCode
L. 198 +8 instr.
L. 199 +12 instr.
L. 200 +2 instr.
6
Add Test To Test Suite
Ignore Test Case
Next Test Case
Previous Test Case
Close
public void html_assSep5() throws Exception {
Attribute attr = new Attribute("key", "value &");
Assertions.assertEquals(234891960, ((int) (((Attribute) (attr)).hashCode())));
5
}
@Test
public void html_literalMutationString19_assSep92() throws Exception {
Attribute attr = new Attribute("key", " Hello\nthere NBSP ");
Assertions.assertEquals("key=\" Hello\nthere   \"", ((Attribute) (attr)).toS
}
@Test
public void html_mg22_assSep1
Test Cube found 4 amplified test cases.
3
Inspect amplification results
Inspect DSpot terminal o


Strongly disagree
Disagree
Neutral
Agree
Strongly agree
Q1 The Control Flow Graph of the method under test is easy to understand.
Q2 The Control Flow Graph of the method under test provides valuable information.
Q3 The interaction with the Control Flow Graph effectively assists you in conveying your test expectations.
Q4 The test generation results displayed with the Control Flow Graph are clear and easy to understand.
Q5 The the Control Flow Graph and branch/line coverage is helpful when you select tests.
Q6 The instruction coverage and corresponding code highlighting is easy to understand.
Q7 The generation result displayed with additional instruction coverage is clear and easy to understand.
Q8 The test information provides valuable information.
Q9 The information on modifications applied to tests are helpful when you select tests.
Q10 The instruction coverage and highlighting code are helpful when you select tests.
Q11 The amplified test cases provided by the tool satisfy your expectations.
Q12 You would want to use the tool to help you write tests in the future.
0%
20%
40%
60%
80%
100%


Instruction Coverage
Neutral
Branch Coverage
Q13 Which type of coverage is easier to understand?
Text
Neutral
Control Flow Graph
Q14 Which display form of coverage is easier to understand?
Open Test Amplification
Neutral
User-Guided Test Amplification
Q15 Which type of test amplification helps you select the amplified test cases more?
Q16 Which type of test amplification is more helpful for you to generate test cases?
0%
20%
40%
60%
80%
100%


Tests
Amplification Tool
Amplified Tests
covers:
Code


Code
Amplification Tool
covers:
Amplified Tests

1) To be actually helpful for users, we need to consciously design how users interact with generative tools like GPT, Copilot, …

2) Too much research blindly applies LLMs to inappropriate tasks (tasks that require other skills than (natural) language composition.