How Should Static Analysis Tools Explain Anomalies to Developers?

Titus Barik · North Carolina State University · tbarik@ncsu.edu
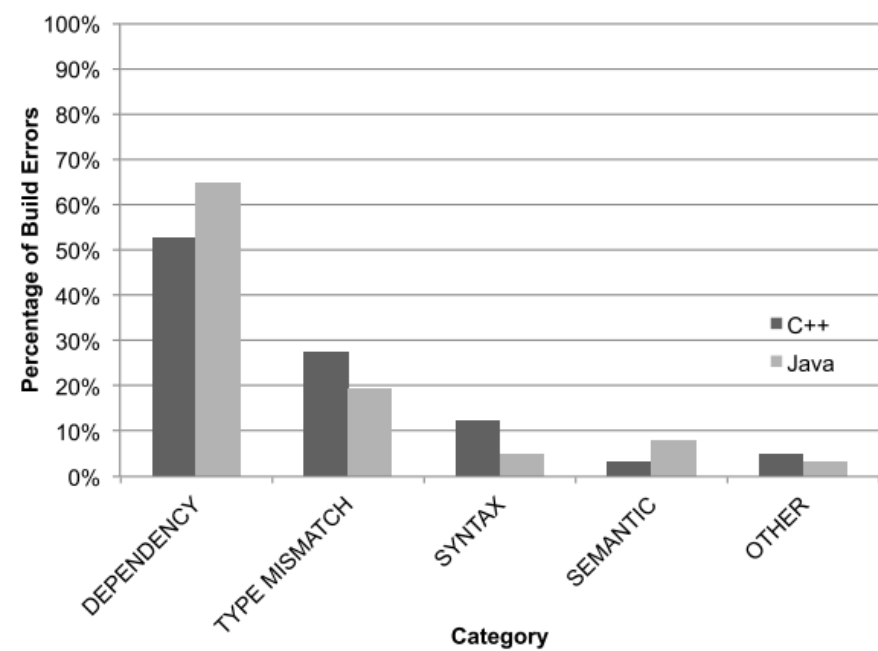
@barik

NC STATE UNIVERSITY

1. Problem

Problem: Static analysis messages in modern IDEs are tricky to understand and resolve. *Why?*
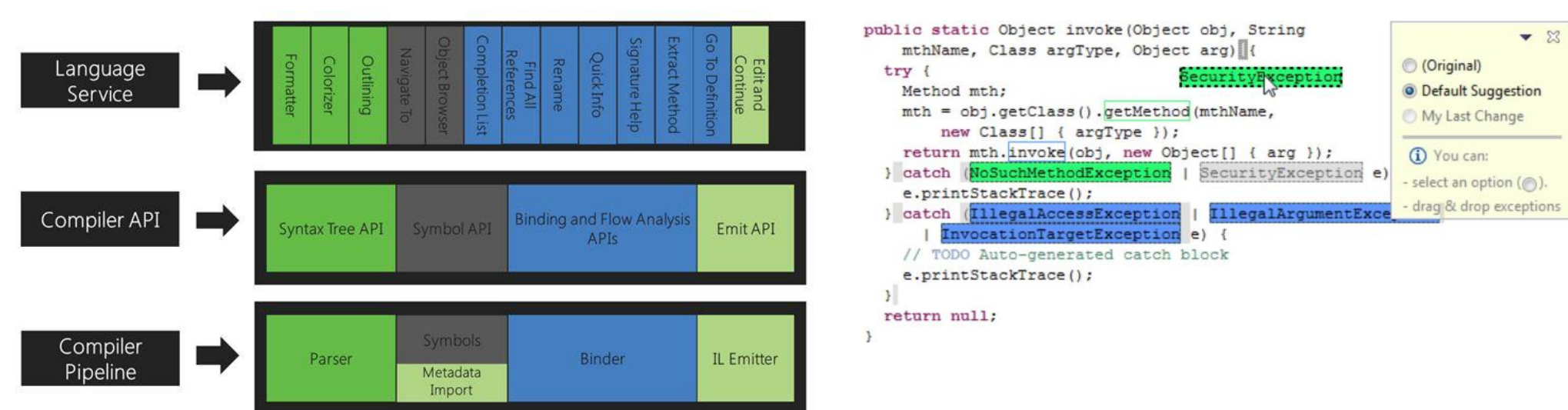
Source: Case study at Google by Seo et al. (ICSE, 2014).

Hypothesis: Tools today fail to adequately support the metacognitive process of *self-explanation*.

2. Computational Explanations

When compilers are **services** for IDEs, we can **expose their internal reasoning processes** to support self-explanation.

3. Self-Explanation Experiment

Between-subjects experiment ($n = 28$) using existing, baseline visualizations against explanatory visualizations.

Baseline

Explanatory

Compiler Output

```
Brick.java:6: error: reference to m is ambiguous,
both method m(int,double) in Brick and method m(double,int) in Brick match
    m(1, 2);
    ^
1 error
```

4. Self-Explanation Findings

Explanatory visualizations result in **more correct self-explanations** by developers.

**Baseline visualizations** fail to reveal critical dependencies, which **thwart self-explanation.**

Dimension	Control		Treatment		
	Median	Dist	Median	Dist	*p*
Hidden Dependencies[*]	3		4		.008
Consistency	4		4		.979
Hard Mental Operations	3		2.5		.821
Role Expressiveness	4		4		.130

5. Eye Tracking in Eclipse
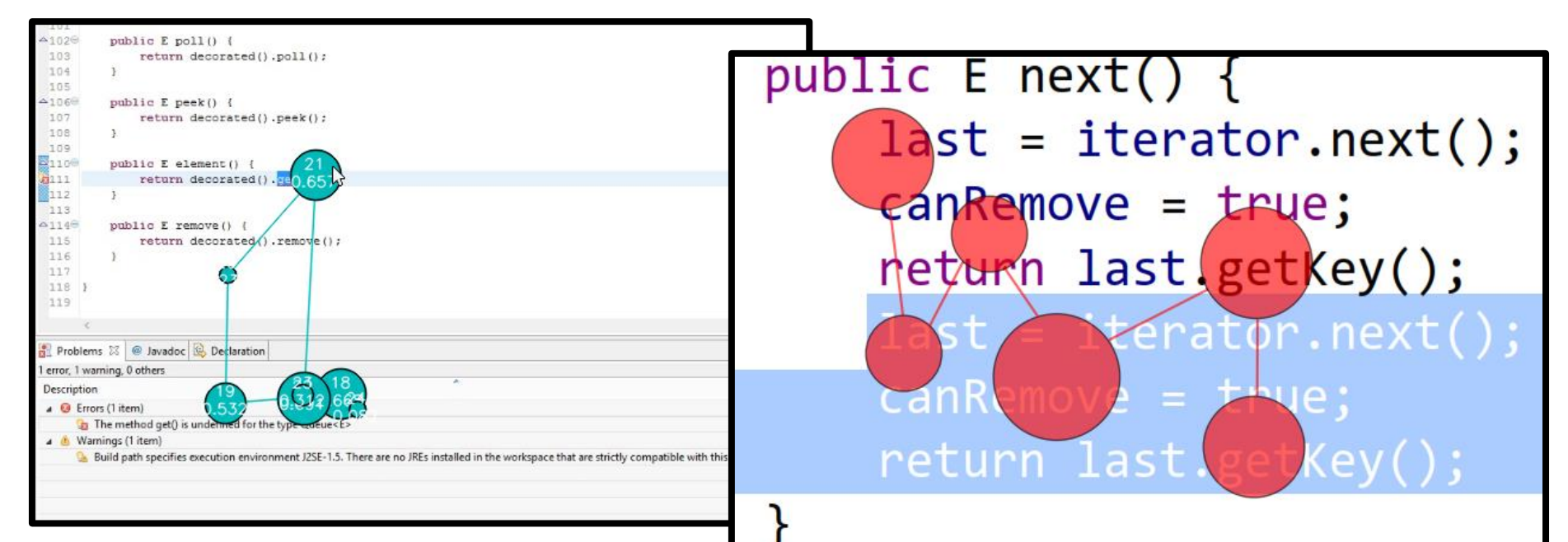
Working Theory: Developers are *biased* by the information in the message and therefore rarely expand their visual search to targets not explicitly identified in message.

6. Tools to Support Developers
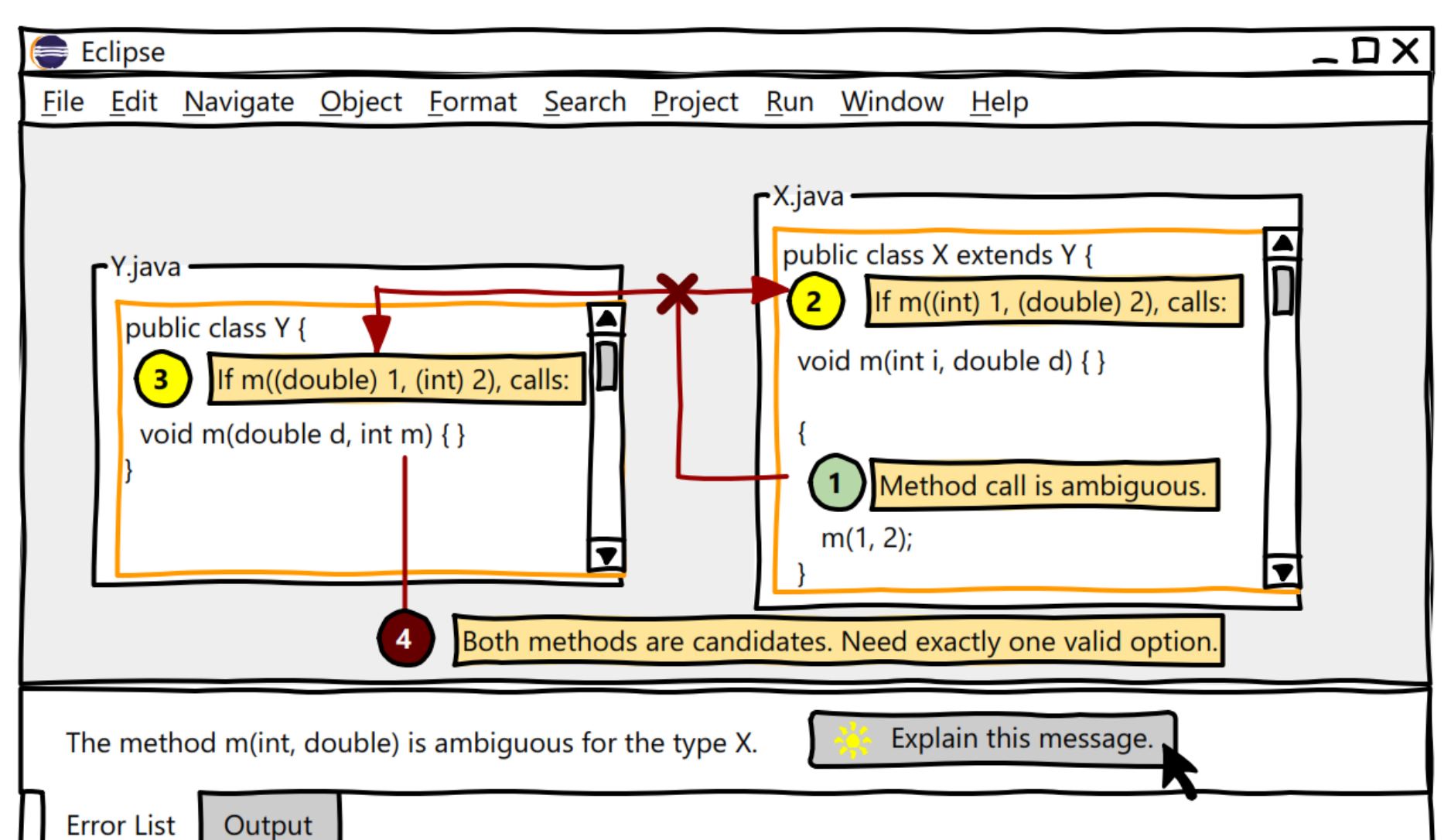
The **results** of this work **inform the design** of novel tools and explanation paradigms.