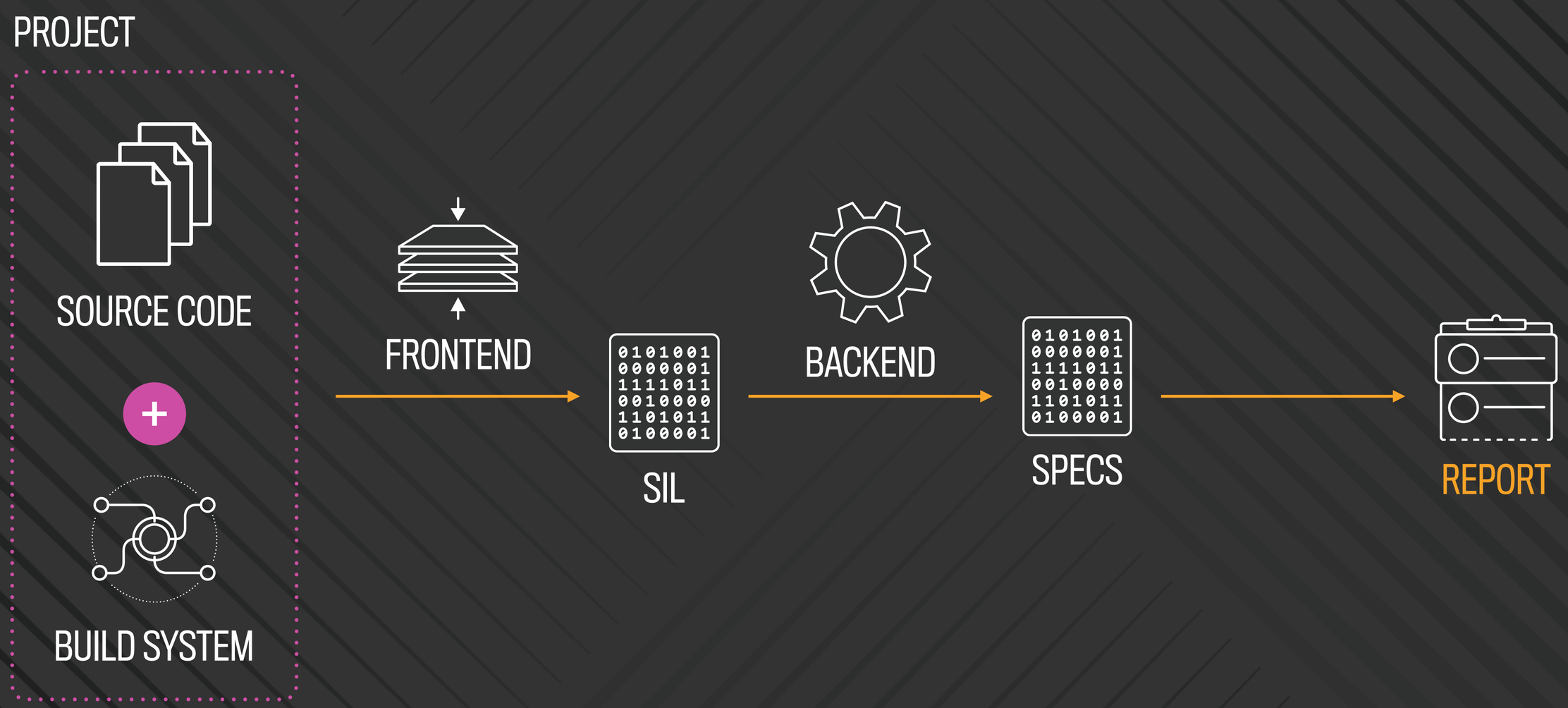
Finding Inter-procedural Bugs at Scale with Infer

Jules Villard <jul@fb.com>
Facebook London

Infer

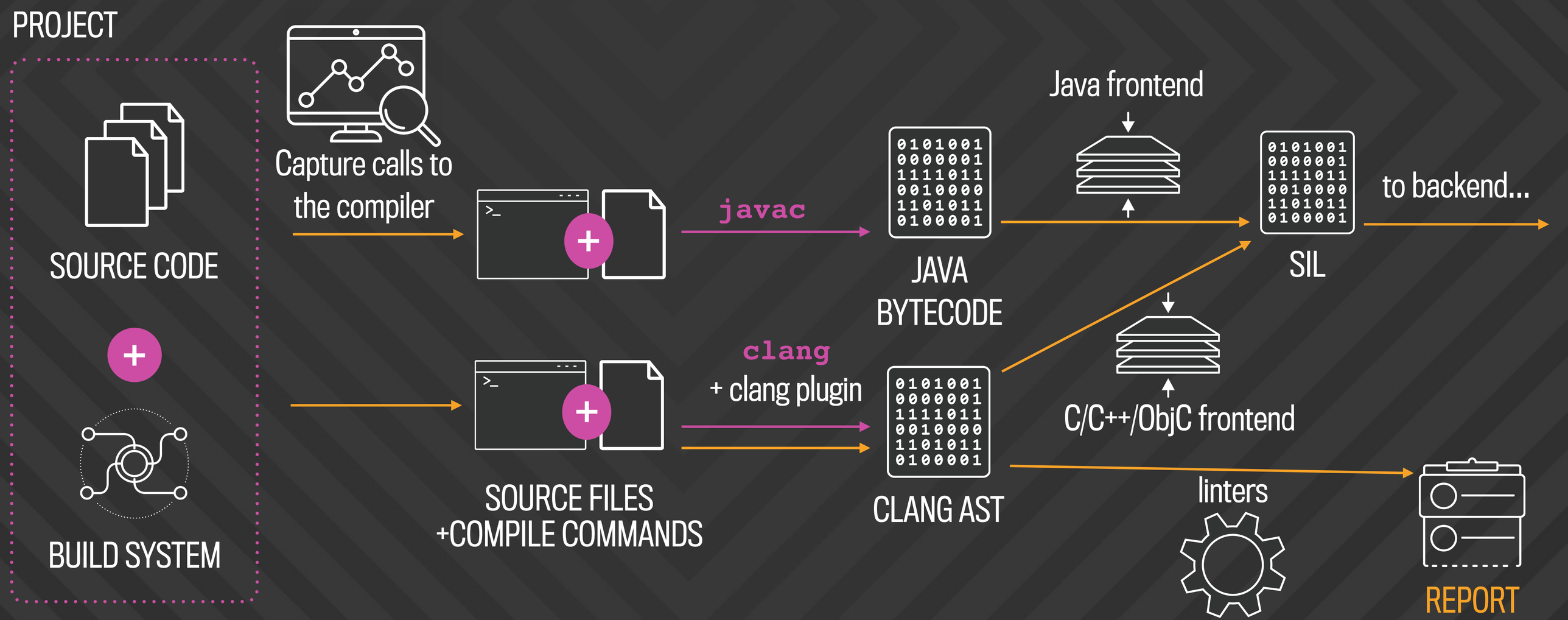
- Open-source static analyser
- Inter-procedural analyses + linters
- For Java and C/C++/Objective-C

Infer architecture

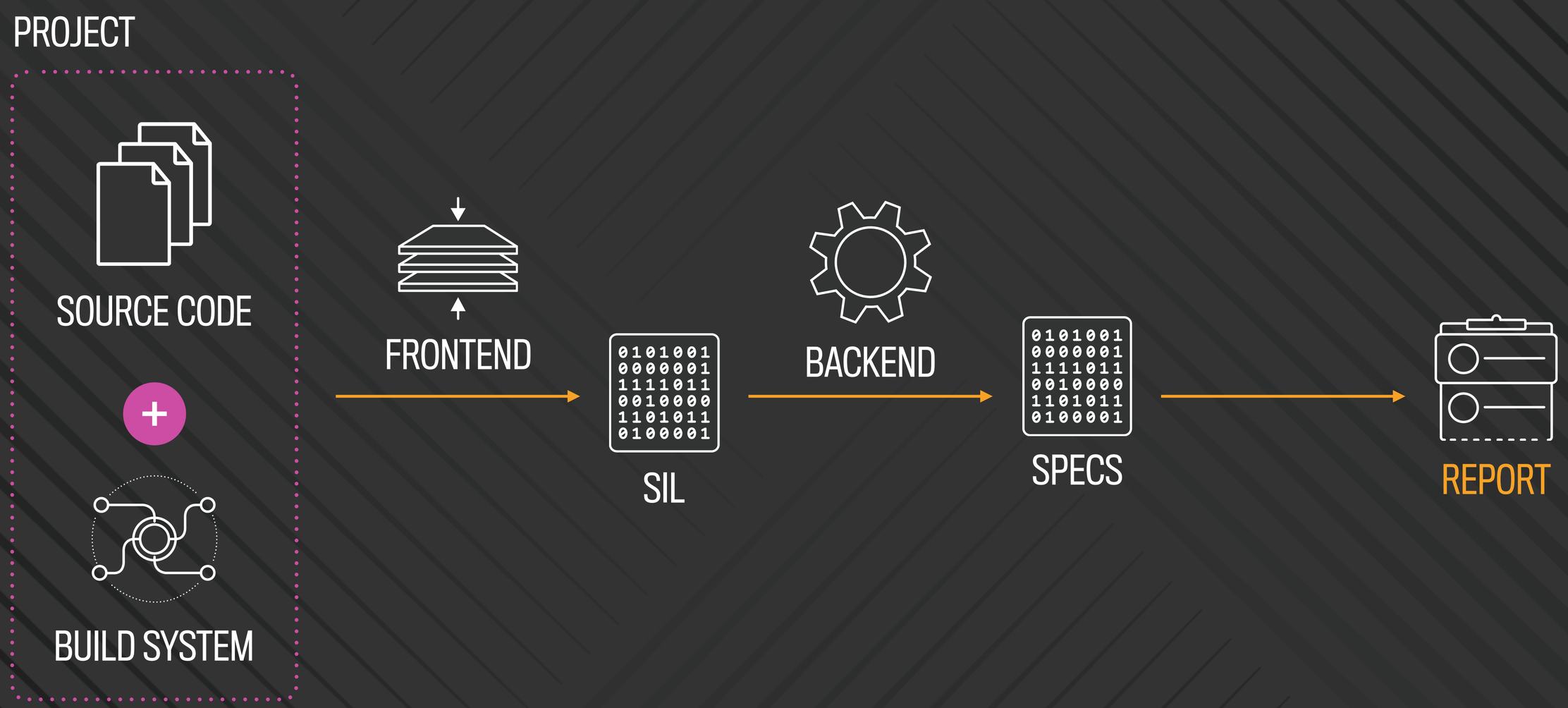


Two Frontends: clang and Java

And quite a few build system integrations



Infer architecture



Compositional, On-Demand Backend Architecture

"Allocates Memory" checker case study

```
1  void foo() {
2    ...
3    Bar.bar();
4    ...
5  }
6
7  @NoAllocation
8  void goo() {
9    ...
10   foo();
11   ...
12 }
```

Foo.java(SIL)

```
1  void bar() {
2    ...
3    new MyObject();
4    ...
5  }
6
7  void baz() {
8    ...
9  }
10
11
12
```

Bar.java(SIL)

Compositional, On-Demand Backend Architecture

"Allocates Memory" checker case study

```
1  void foo() {
2
3
4
5  }
6
7  @NoAllocation
8  void goo() {
9
10
11
12 }
```

Allocation via call to bar() line 3

Allocation via call to foo() line 10

ERROR

Foo.java (SIL)

```
1  void bar() {
2
3
4
5  }
6
7  void baz() {
8    ...
9  }
10
11
12
```

Allocation line 3

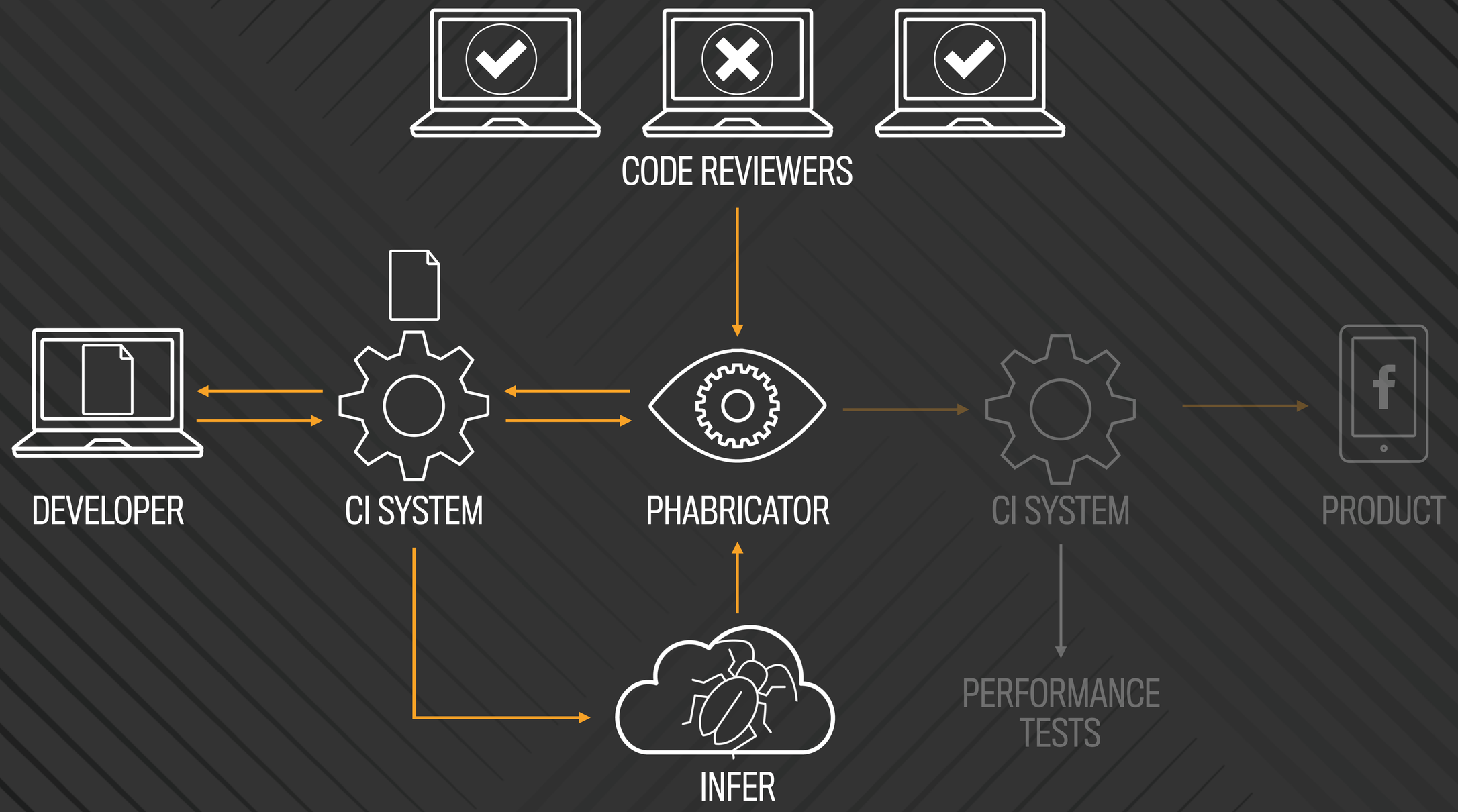
Bar.java (SIL)

https://code.facebook.com/posts/1537144479682247/finding-inter-procedural-bugs-at-scale-with-infer-static-analyzer/

Interprocedural Analysis Case Study

Percentages of inter-procedural reports for different types of bugs

	One procedure One file	Interprocedural One file	Interprocedural Inter-file
Allocates Memory	0	2	98
Null Dereference (Java)	43	9	48
Null Dereference (Objective-C)	73	5	24
RacerD	36	12	53
Bad Pointer Comparison (linter)	100	0	0


CODE REVIEWERS
DEVELOPER
CI SYSTEM
PHABRICATOR
CI SYSTEM
PRODUCT
INFER
PERFORMANCE TESTS

Diff comments fit into usual workflow

infer_report_example/CodeSample.java View Options

This file was **added**.

```
1  public class CodeSample {
2    public String computeSomething(boolean flag) {
3      if (flag) {
4        return null;
5      }
6      else {
7        return "something";
8      }
9    }
10
11   public int doStuff() {
12     String s = computeSomething(true);
13     return s.length();
```

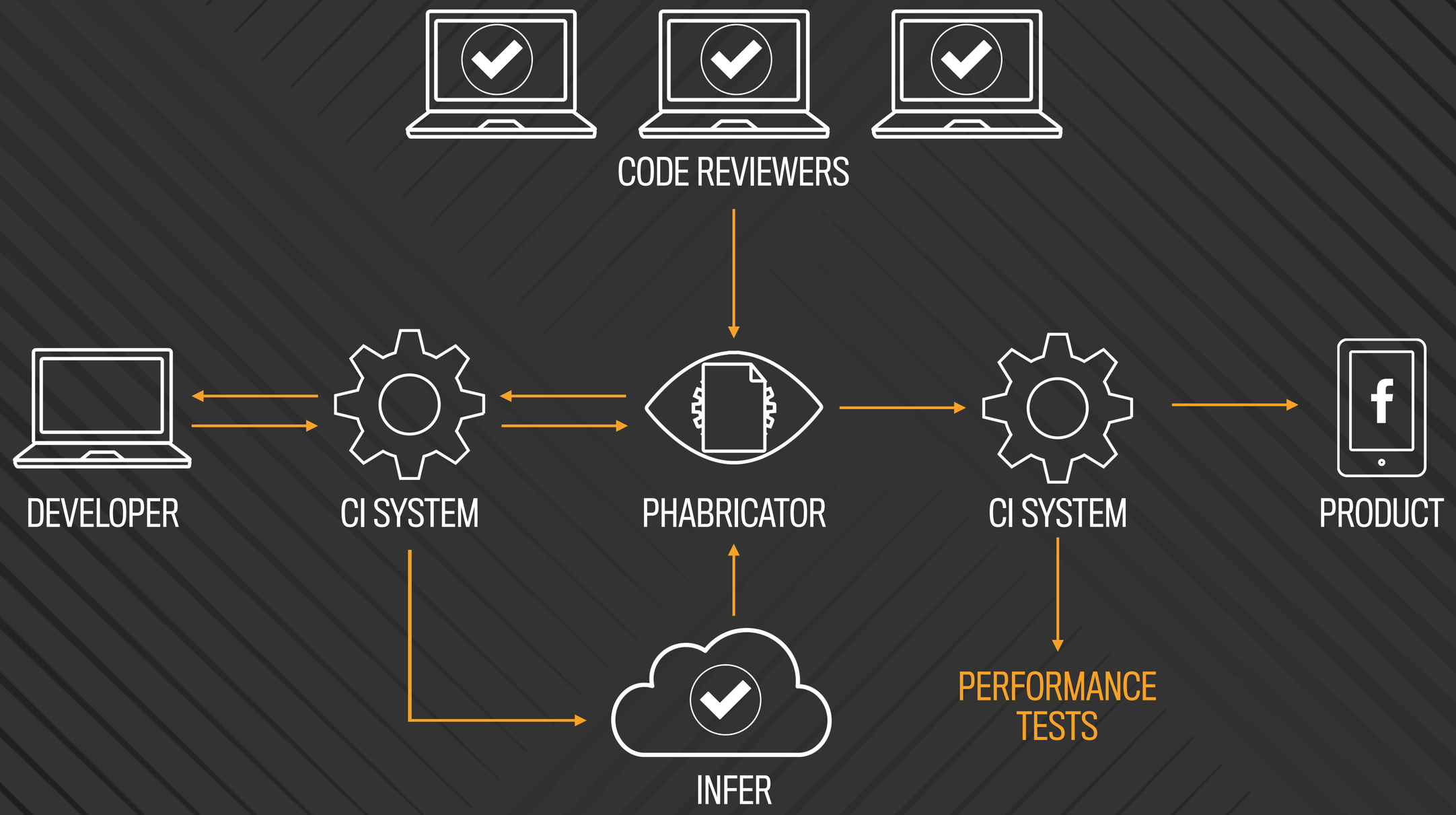
Line 13 Previous · Next · Reply

There may be a Null Dereference: object s last assigned on line 12 could be null and is dereferenced at line 13

```
14   }
15 }
```

Only report when:

- Warning is introduced by diff
- Warning is in file changed by diff


CODE REVIEWERS
DEVELOPER
CI SYSTEM
PHABRICATOR
CI SYSTEM
PRODUCT
INFER
PERFORMANCE TESTS

Analysing a Diff

"Allocates Memory" checker case study

```
--- Foo.java
+++ Foo.java
 @NoAllocation
 void goo() {
    ...
+   foo();
    ...
 }
```

diff

with diff

Analysing a Diff

"Allocates Memory" checker case study

```
--- Foo.java
+++ Foo.java
 @NoAllocation
 void goo() {
    ...
+   foo();
    ...
 }
```

diff

```
1  void foo() {
2     ...
3     Bar.bar();
4     ...
5  }
6
7  @NoAllocation
8  void goo() {
9     ...
10    foo();
11    ...
12 }
```

Foo.java (SIL)

```
1  void bar() {
2     ...
3     new MyObject();
4     ...
5  }
6
7  void baz() {
8     ...
9  }
10
11
12
```

Bar.java (SIL)

with diff

Analysing a Diff

"Allocates Memory" checker case study

```
--- Foo.java
+++ Foo.java
 @NoAllocation
 void goo() {
    ...
+   foo();
    ...
 }
```

diff

```
1  void foo() {
5  }
6
7  @NoAllocation
8  void goo() {
12 }
```

Allocation via call to bar() line 3

Allocation via call to foo() line 10

ERROR

Foo.java (SIL)

```
1  void bar() {
2
3
4
5  }
6
7  void baz() {
8     ...
9  }
10
11
12
```

Allocation line 3

Bar.java (SIL)

base

Analysing a Diff

"Allocates Memory" checker case study

```
--- Foo.java
+++ Foo.java
 @NoAllocation
 void goo() {
    ...
+   foo();
    ...
 }
```

diff

```
1  void foo() {
2     ...
3     Bar.bar();
4     ...
5  }
6
7  @NoAllocation
8  void goo() {
9     ...
10    ...
11 }
12
```

Foo.java (SIL)

```
1  void bar() {
2
3
4
5  }
6
7  void baz() {
8     ...
9  }
10
11
12
```

Allocation line 3

Bar.java (SIL)

base

Analysing a Diff

"Allocates Memory" checker case study

```
--- Foo.java
+++ Foo.java
 @NoAllocation
 void goo() {
    ...
+   foo();
    ...
 }
```

diff

```
1  void foo() {
5  }
6
7  @NoAllocation
8  void goo() {
9
10
11
12
```

Allocation via call to bar() line 3

No allocation

Foo.java (SIL)

```
1  void bar() {
2
3
4
5  }
6
7  void baz() {
8     ...
9  }
10
11
12
```

Allocation line 3

Bar.java (SIL)

Analysing a Diff

"Allocates Memory" checker case study

```
--- Foo.java
+++ Foo.java
 @NoAllocation
 void goo() {
    ...
+   foo();
    ...
 }
```

diff

base — No report

diff — **ERROR** foo() allocates memory on line 10

DIFFERENTIAL REPORT

diff - base =
ERROR foo() allocates memory on line 10

Diff-Based Deployment

- Help developers move fast
- Easy to deploy new checks

Current status

- Infer runs on all Android + iOS diffs for Facebook, Messenger, Instagram, and WhatsApp
- 10ks of diffs analyzed per month
- 1ks of issues fixed per month (~70% fix rate)

Action taken is ground truth for success

Finding Inter-procedural Bugs at Scale with Infer

Jules Villard <jul@fb.com>
Facebook London