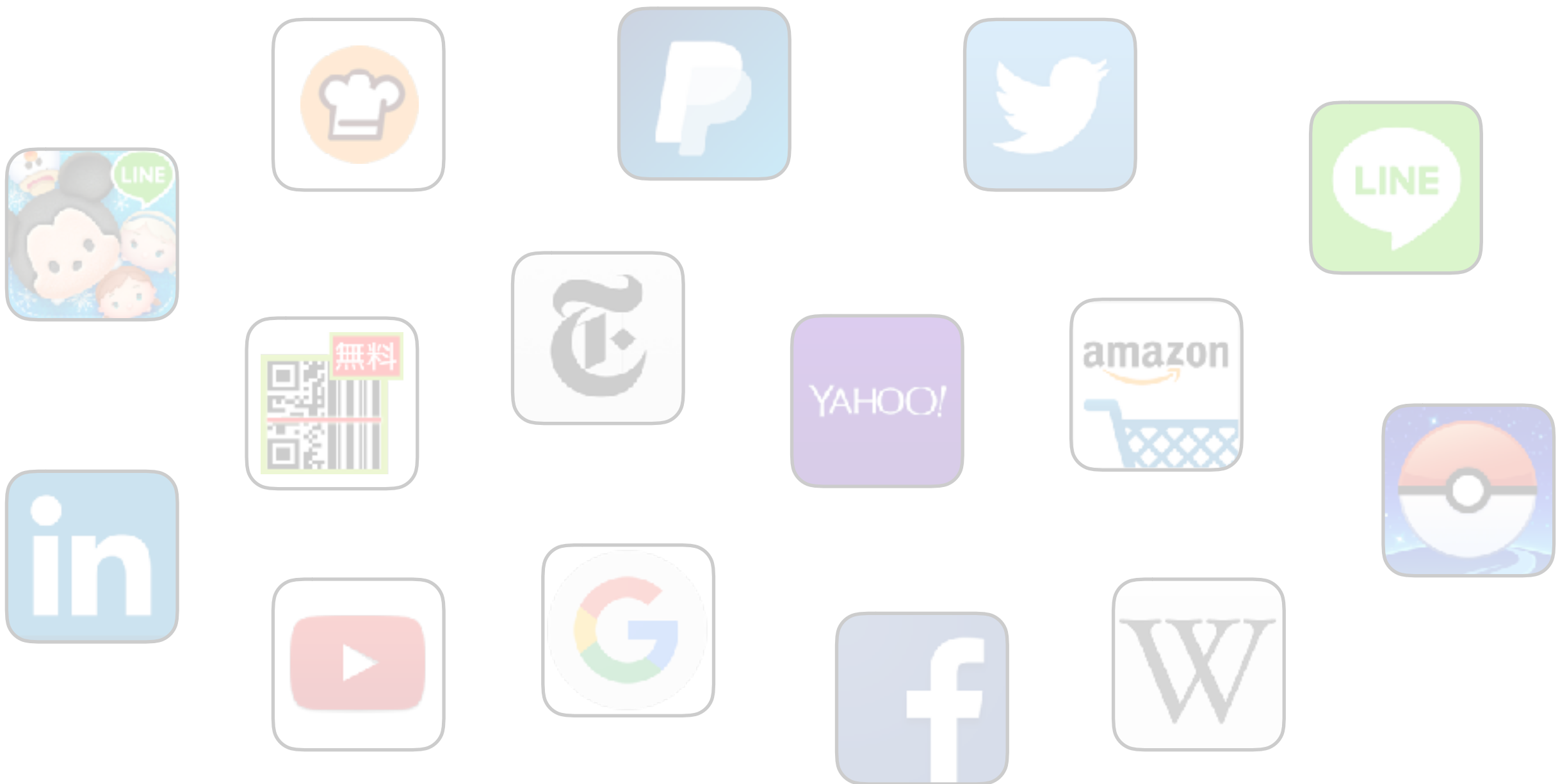


Automated Support for Mobile Application Testing and Debugging

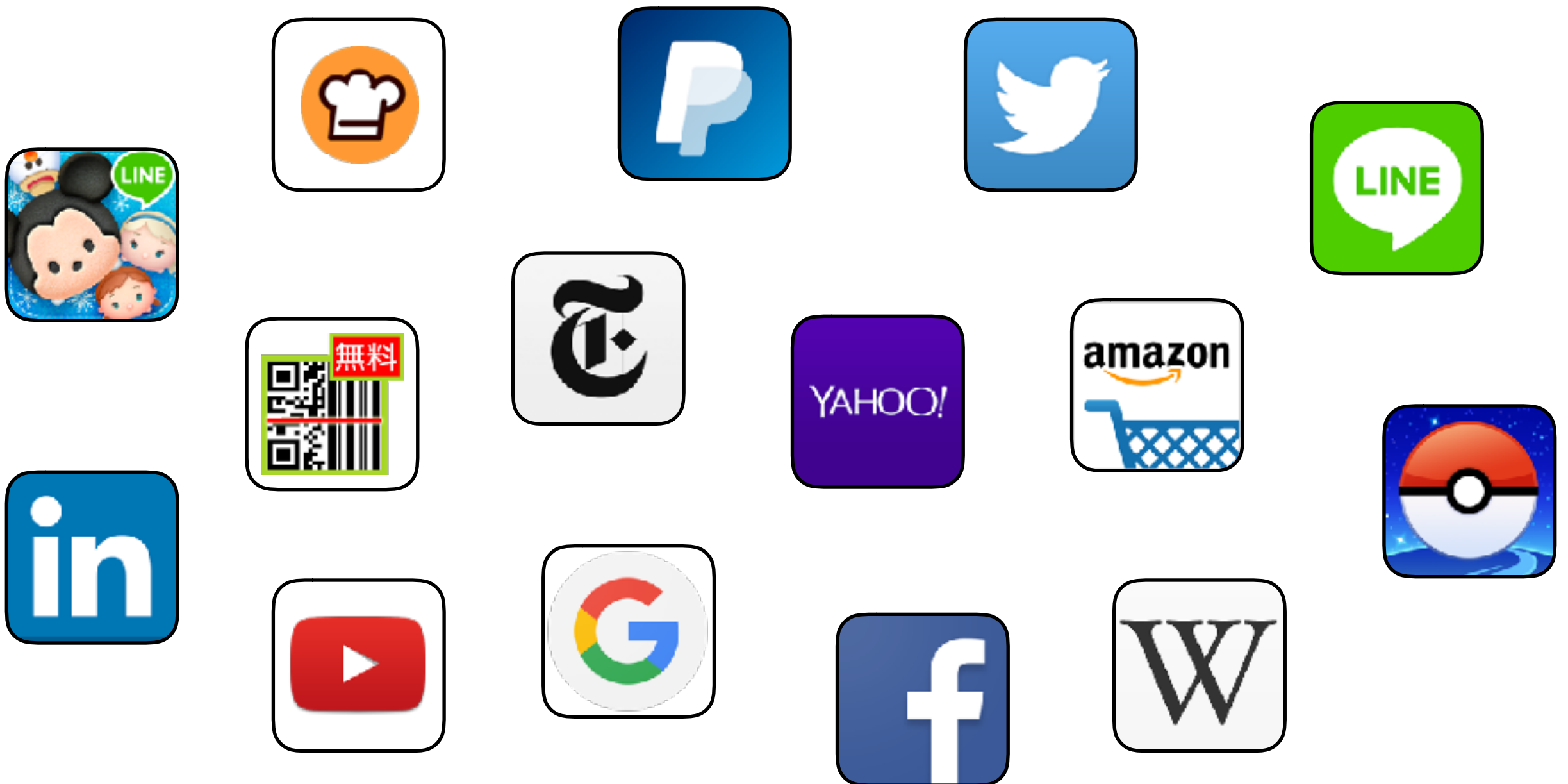
Mattia Fazzini



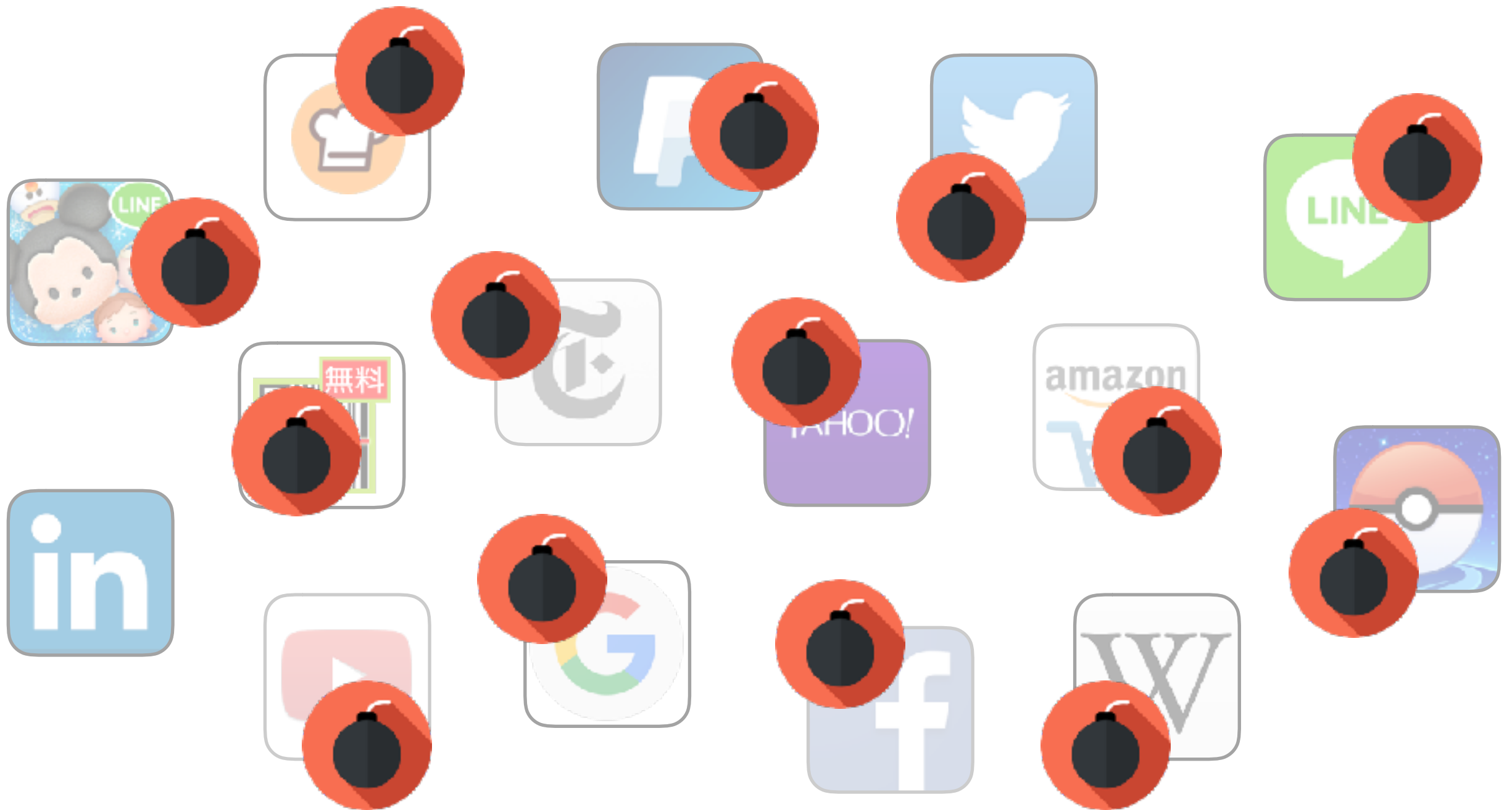
Mobile Applications



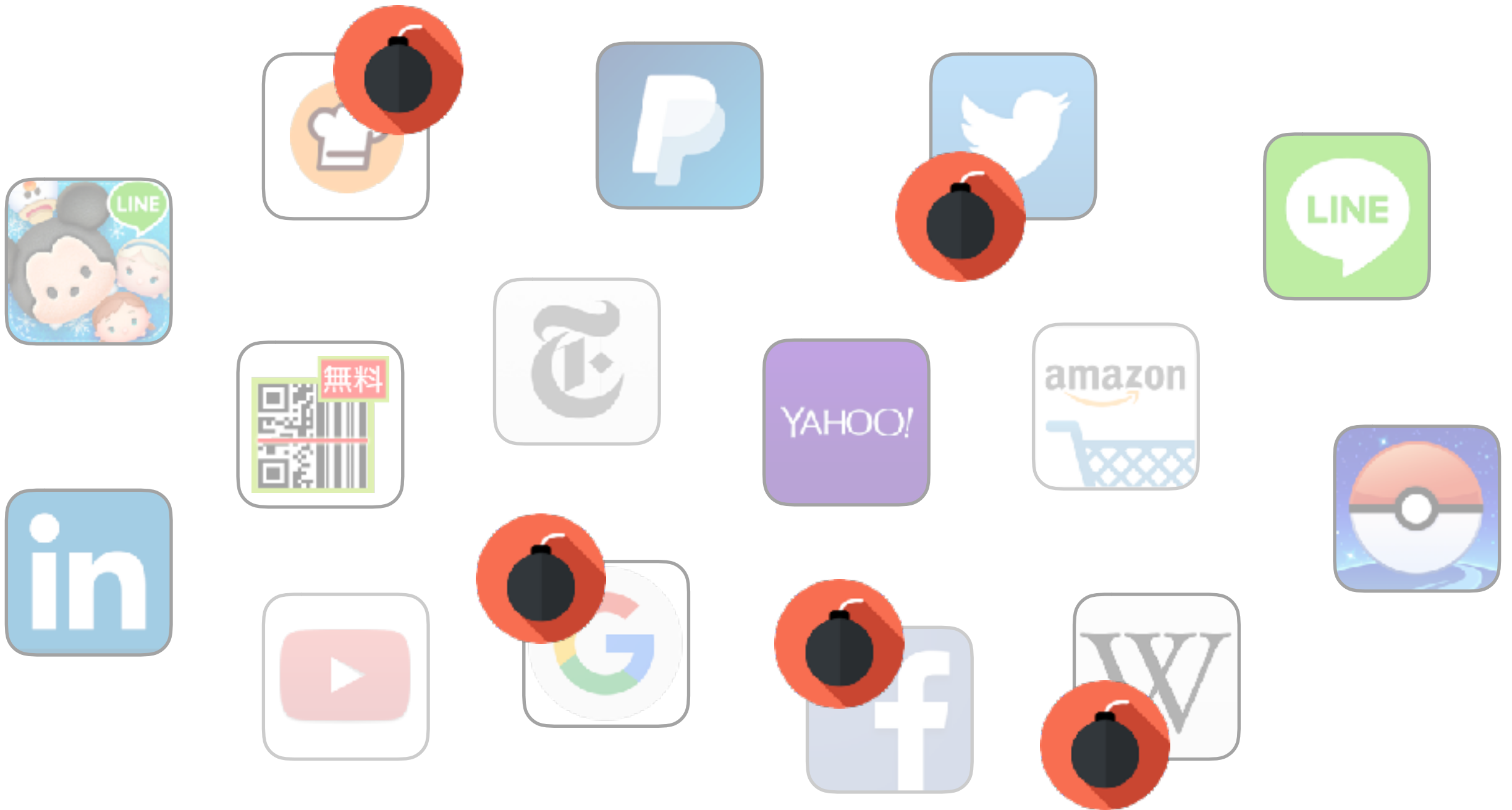
Mobile Applications



Failures



Testing



Bug Reports

Arkonos commented on Jul 5, 2017

FC upon selecting weekly backup frequency

Steps to reproduce:

Settings -> Backup -> Backup frequency
100% for me, no other frequency selected
Once after such crash, the app would not
reproduce this.
Pretty sure the same thing happened

Version:

Phone: Nexus 5

Android: 6.0.1

LineageOS: 14.0

Kernel: 3.10.108

afwang commented on

I found this crash on version 1.0.0
[redreader_crash_log.txt](#)

Reproduction steps:

1. Launch redreader
2. Tap the refresh button in the app bar
3. Swipe down while the subscribed stream is loading

I think it might help to reproduce if you're using a slow network speed is throttled.



QuantumBadger added the [Linux](#) label

Trolldemorted commented on May 19, 2017

I have:

open and closed issues for
<https://github.com/WhisperSystem/WhisperSystem>

Description

es.

runtime(30920): Shutting down
runtime(30920): FATAL EXCEPTION:
runtime(30920): Process: org.whispersystems.
runtime(30920): java.lang.IllegalStateException:
runtime(30920): at org.whispersystems.

Tweet opens with selected account

Actual behavior

Twidere crashes

Steps to reproduce

Longpress retweet icon underneath tweet



Goals

Help developers to find bugs early with novel testing techniques

Assist developers in quickly resolving bugs by automatically analyzing bug reports

Research

Test
Generation



Differential
Testing



Bug Report
Analysis



Research

Test Generation



Differential Testing



Bug Report Analysis

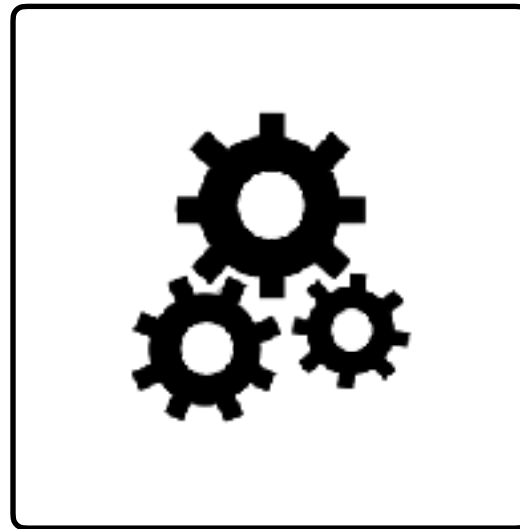


Research

**Test
Generation**



**Differential
Testing**



**Bug Report
Analysis**

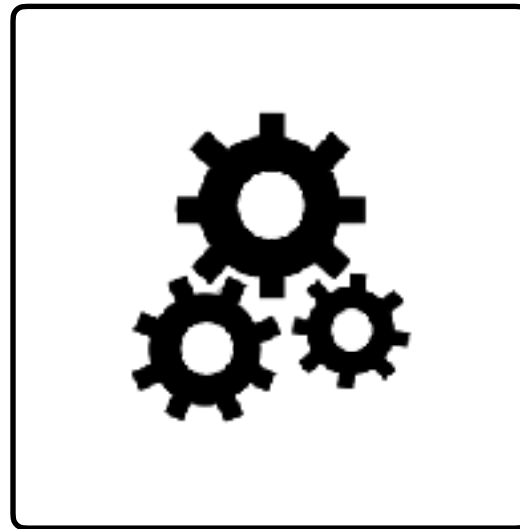


Research

**Test
Generation**



**Differential
Testing**



**Bug Report
Analysis**



Techniques

**Test
Generation**

Barista

**Differential
Testing**

DiffDroid

**Bug Report
Analysis**

Yakusu

Research

**Test
Generation**

Barista

**Differential
Testing**

DiffDroid

**Bug Report
Analysis**

Yakusu

Mobile App Testing

Challenges

Not enough time to test



Do not have the right testing process/method



Do not have the right tools to test



No mobile testing experts



Do not have in-house testing environment



Do not have the devices readily available



We do not do testing



Mobile App Testing

Challenges

Not enough time to test



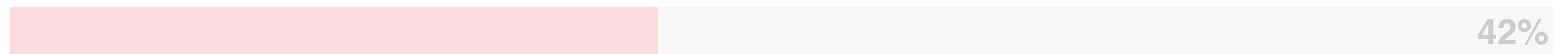
Do not have the right testing process/method



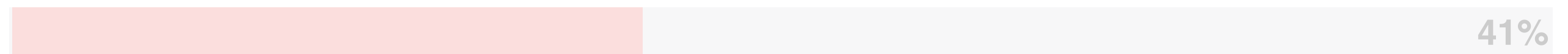
Do not have the right tools to test



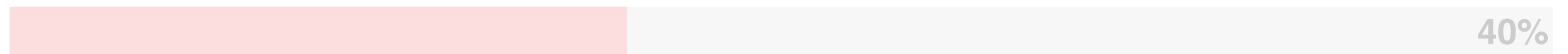
No mobile testing experts



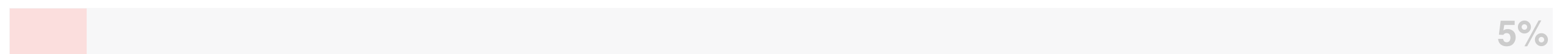
Do not have in-house testing environment



Do not have the devices readily available



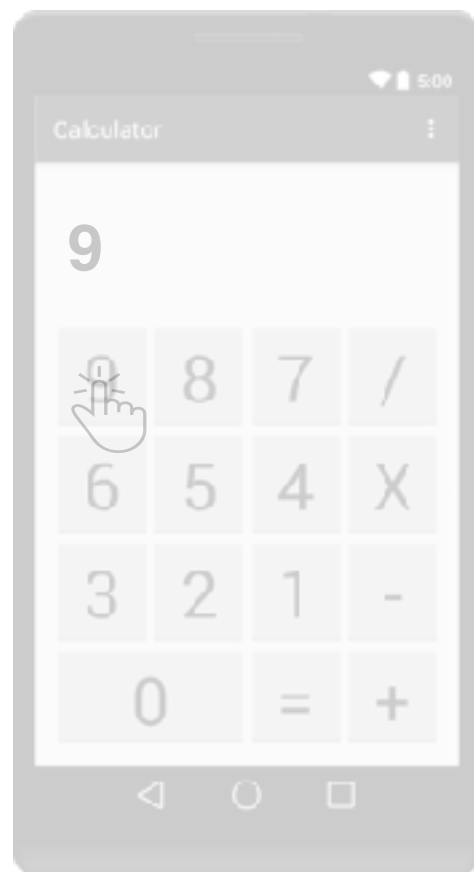
We do not do testing



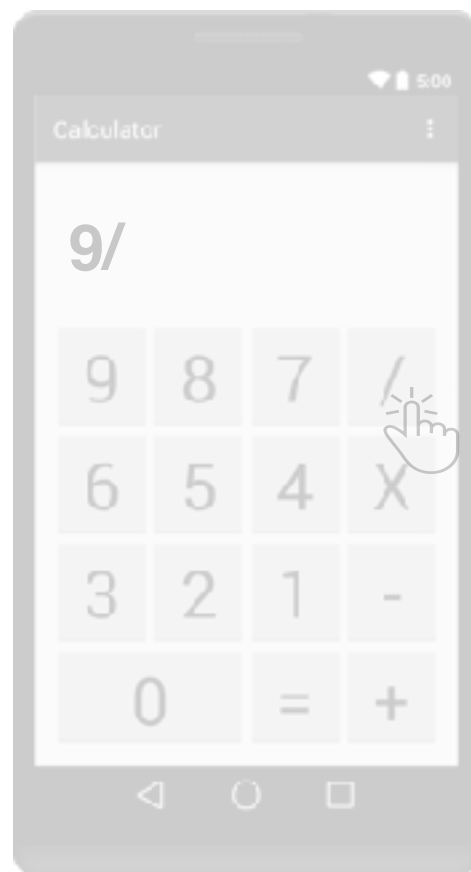
Running Example: Manual Testing

Division by zero: nine divided by zero results in error message

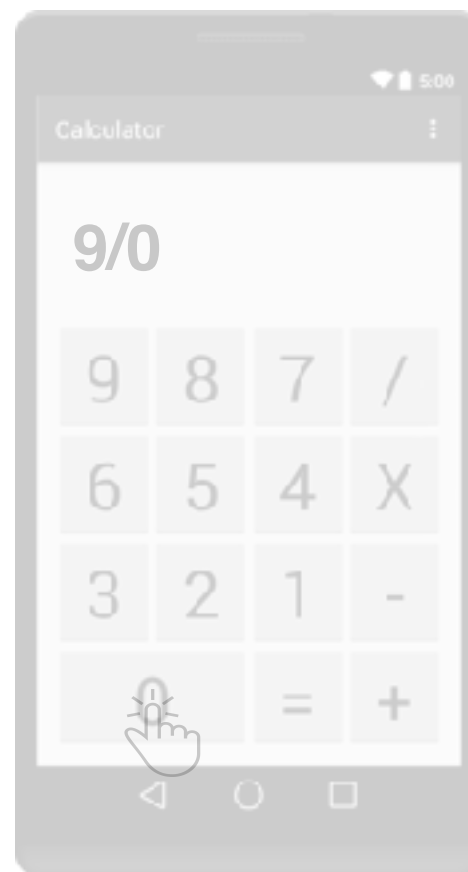
(1) Press nine



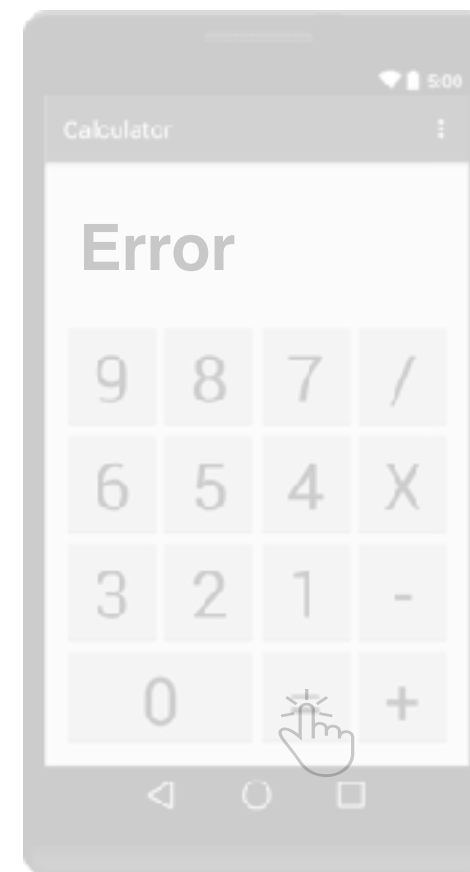
(2) Press divide



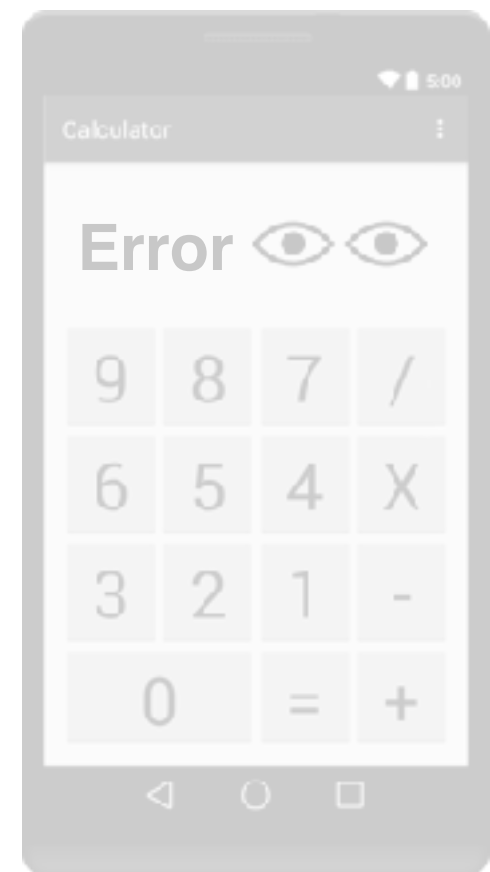
(3) Press zero



(4) Press equal



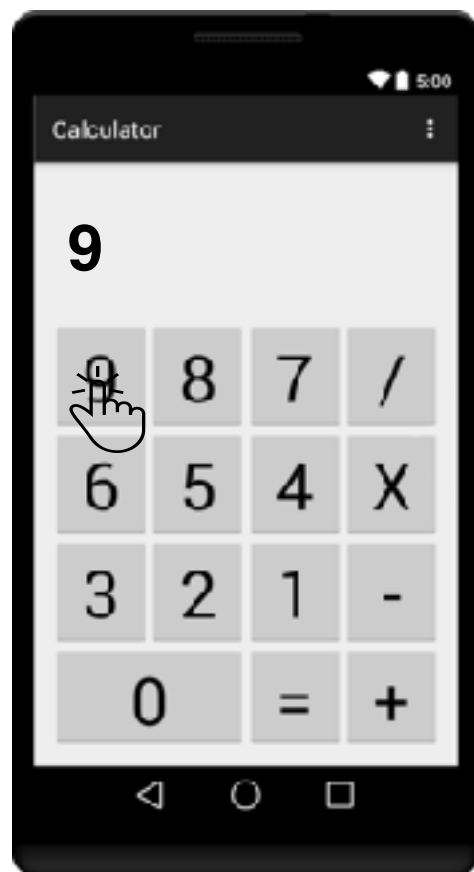
(5) Check message



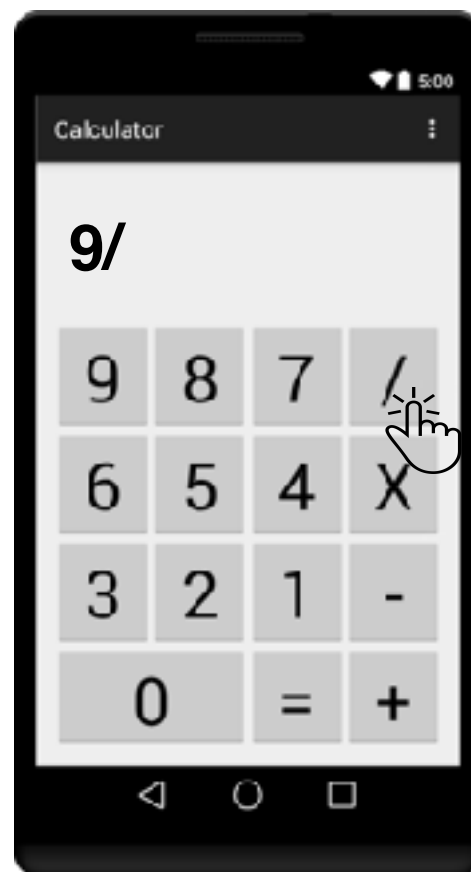
Running Example: Manual Testing

Division by zero: nine divided by zero results in error message

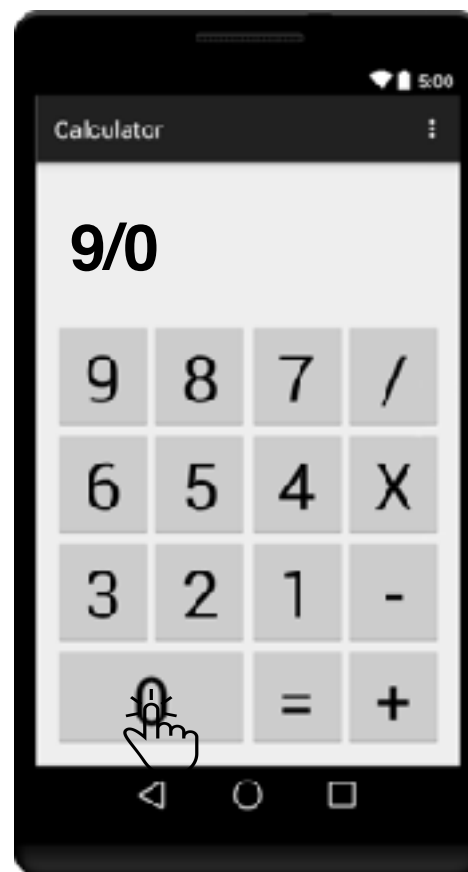
(1) Press nine



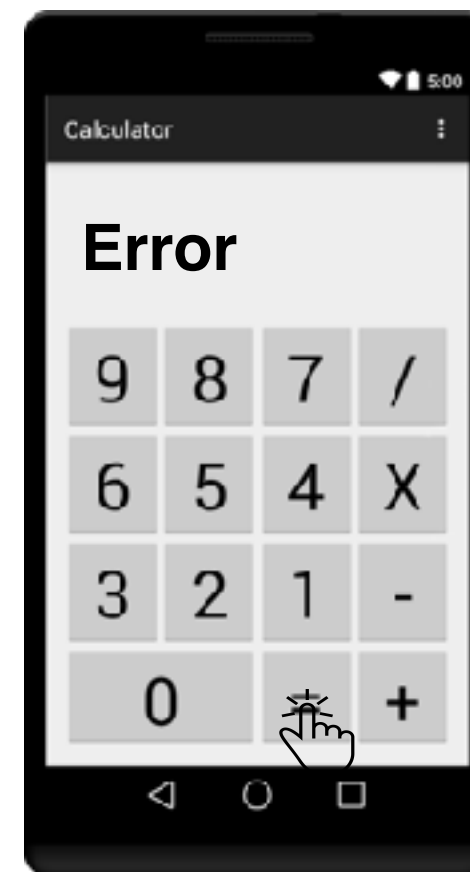
(2) Press divide



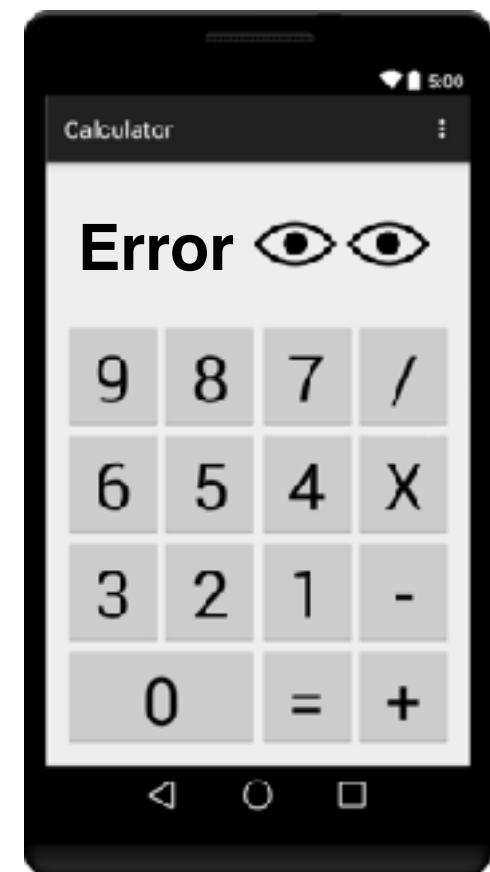
(3) Press zero



(4) Press equal



(5) Check message



Running Example: Manual Testing

Division by zero: nine divided by zero results in error message

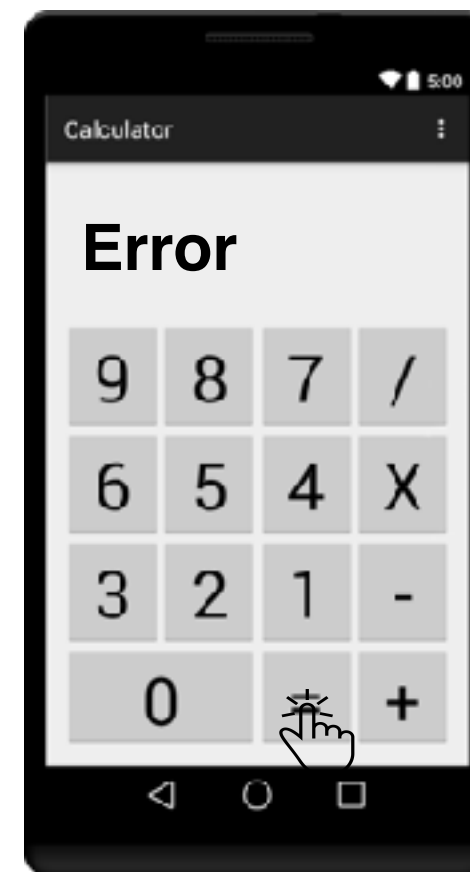
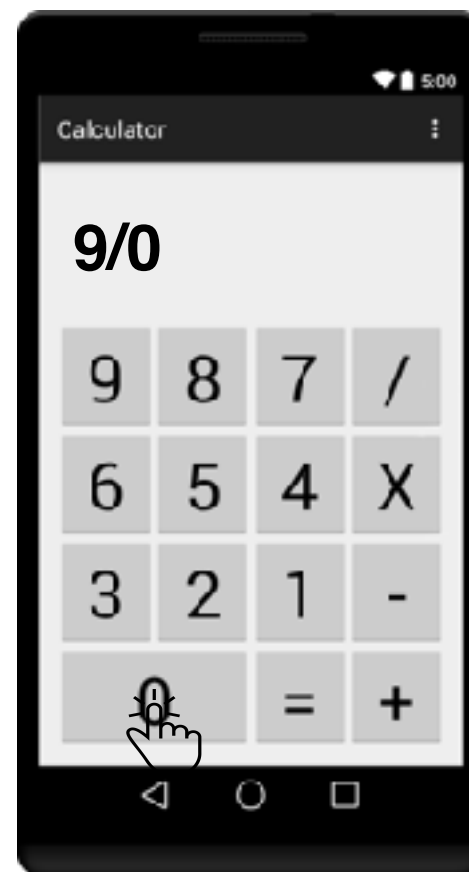
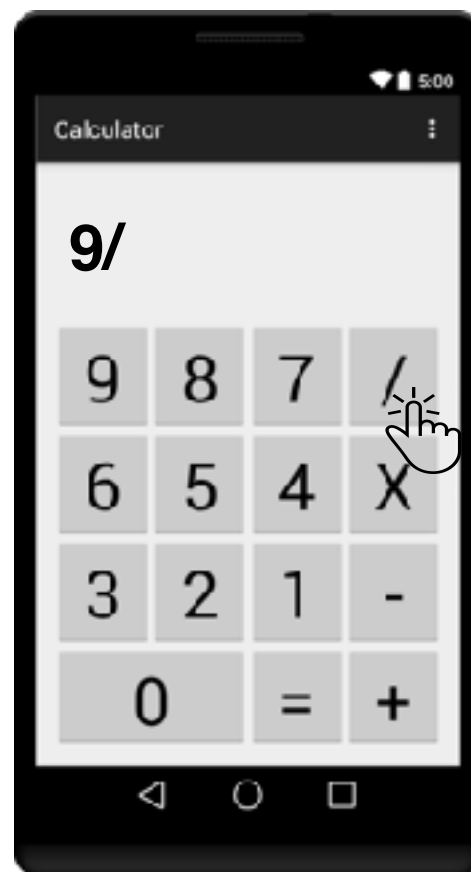
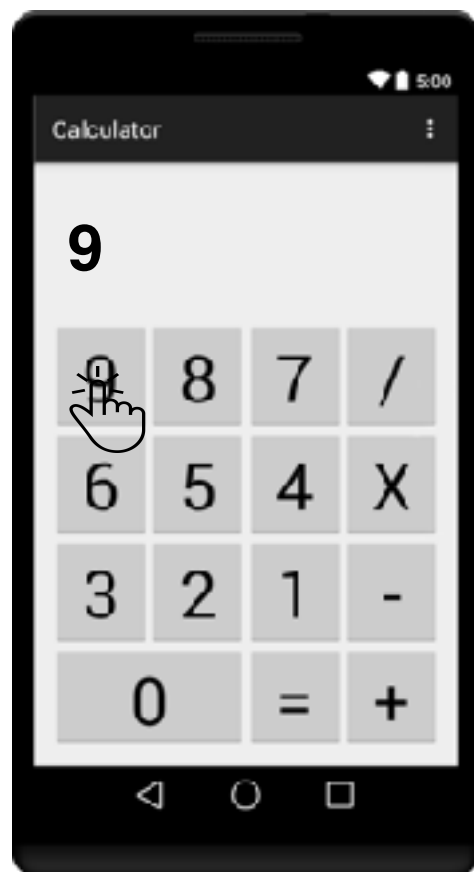
(1) Press nine

(2) Press divide

(3) Press zero

(4) Press equal

(5) Check message



Simple

Running Example: Manual Testing

Division by zero: nine divided by zero results in error message

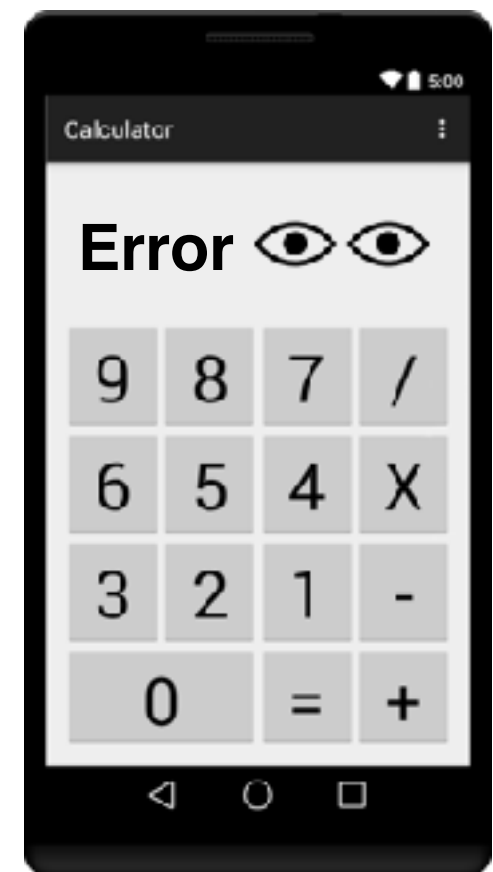
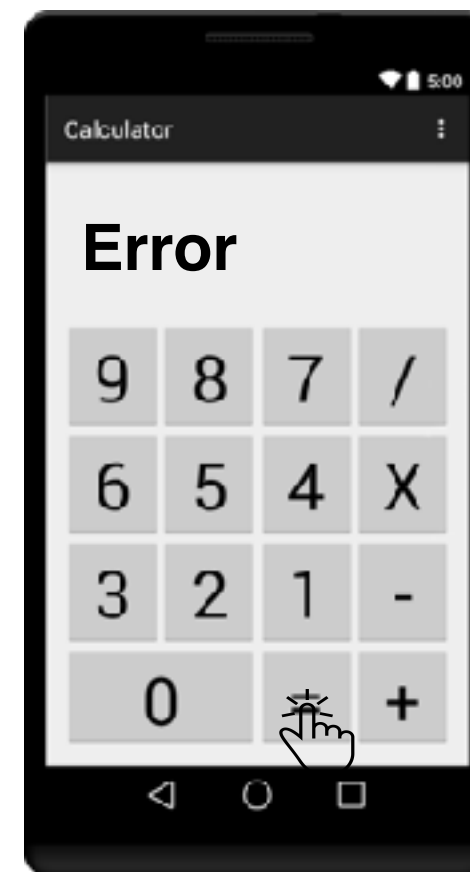
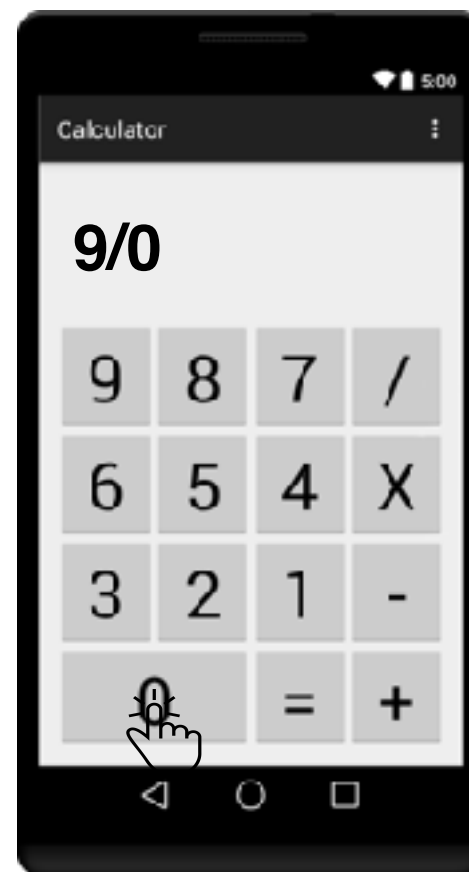
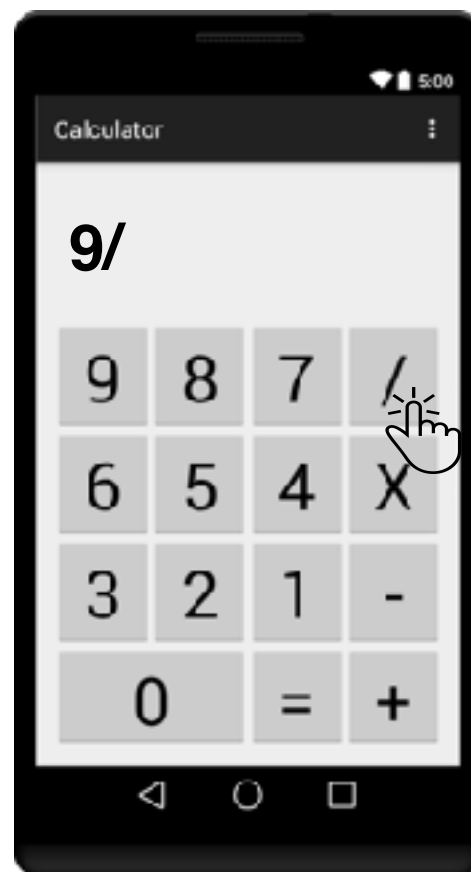
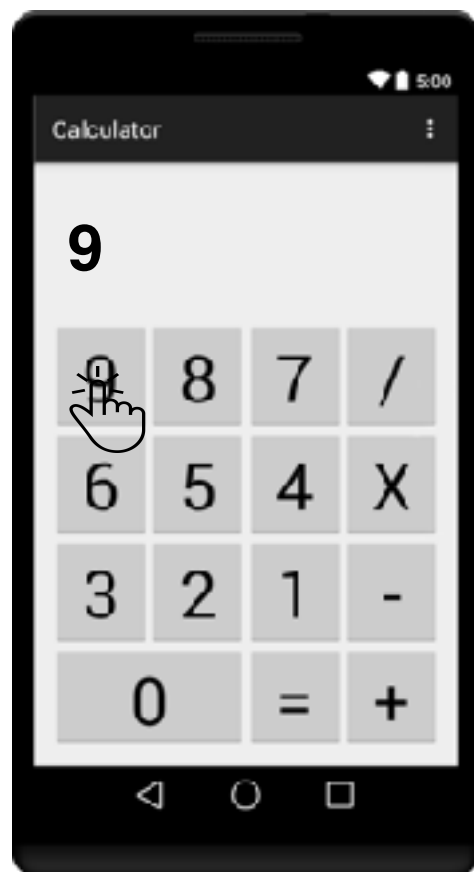
(1) Press nine

(2) Press divide

(3) Press zero

(4) Press equal

(5) Check message



Simple

Inefficient, Tedious, Error prone

Running Example: Automated Testing

Division by zero: nine divided by zero results in error message

- (1) Press nine
- (2) Press divide
- (3) Press zero
- (4) Press equal
- (5) Check message

```
1 public void divisionByZero() {  
2     onView(withText("9")).perform(click());  
3     onView(withId(R.id.div)).perform(click());  
4     onView(withText("0")).perform(click());  
5     onView(withId(R.id.eq)).perform(click());  
6     onView(withId(R.id.dsp))  
7     .check(matches(withText("Error")));  
8 }
```

Running Example: Automated Testing

Division by zero: nine divided by zero results in error message

- (1) Press nine
- (2) Press divide
- (3) Press zero
- (4) Press equal
- (5) Check message

```
1 public void divisionByZero() {  
2     onView(withText("9")).perform(click());  
3     onView(withId(R.id.div)).perform(click());  
4     onView(withText("0")).perform(click());  
5     onView(withId(R.id.eq)).perform(click());  
6     onView(withId(R.id.dsp))  
7     .check(matches(withText("Error")));  
8 }
```

Running Example: Automated Testing

Division by zero: nine divided by zero results in error message

- (1) Press nine
- (2) Press divide
- (3) Press zero
- (4) Press equal
- (5) Check message

```
1 public void divisionByZero() {  
2     onView(withText("9")).perform(click());  
3     onView(withId(R.id.div)).perform(click());  
4     onView(withText("0")).perform(click());  
5     onView(withId(R.id.eq)).perform(click());  
6     onView(withId(R.id.dsp))  
7     .check(matches(withText("Error")));  
8 }
```

Running Example: Automated Testing

Division by zero: nine divided by zero results in error message

- (1) Press nine
- (2) Press divide
- (3) Press zero
- (4) Press equal
- (5) Check message

```
1 public void divisionByZero() {  
2     onView(withText("9")).perform(click());  
3     onView(withId(R.id.div)).perform(click());  
4     onView(withText("0")).perform(click());  
5     onView(withId(R.id.eq)).perform(click());  
6     onView(withId(R.id.dsp))  
7     .check(matches(withText("Error")));  
8 }
```

Efficient

Running Example: Automated Testing

Division by zero: nine divided by zero results in error message

- (1) Press nine
- (2) Press divide
- (3) Press zero
- (4) Press equal
- (5) Check message

```
1 public void divisionByZero() {  
2     onView(withText("9")).perform(click());  
3     onView(withId(R.id.div)).perform(click());  
4     onView(withText("0")).perform(click());  
5     onView(withId(R.id.eq)).perform(click());  
6     onView(withId(R.id.dsp))  
7     .check(matches(withText("Error")));  
8 }
```

Efficient

Complex

Running Example: Barista

Manual Testing

Simplicity

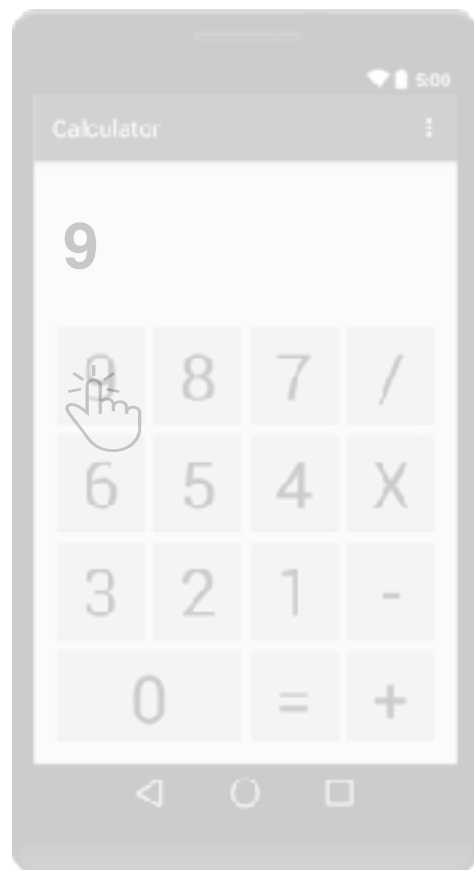


Barista



Automated Testing

Efficiency



```
1 public void divisionByZero() {
```

```
8 }
```


Running Example: Barista

Manual Testing

Simplicity

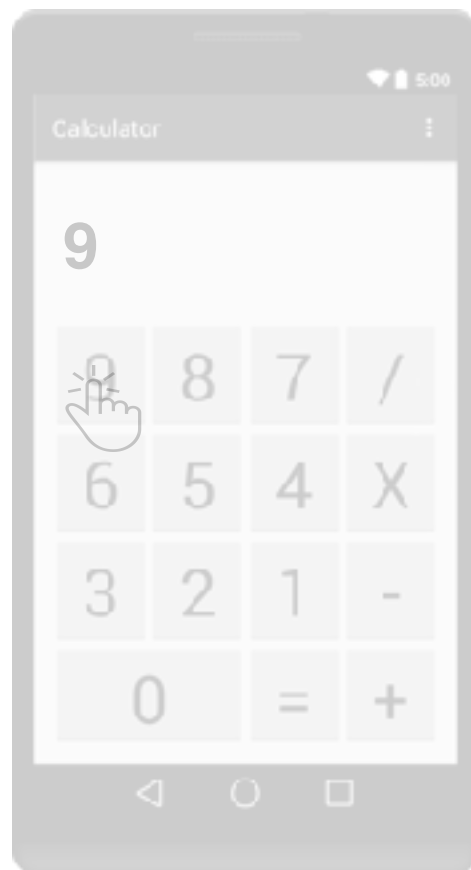


Barista



Automated Testing

Efficiency



```
1 public void divisionByZero() {
```

```
8 }
```


Running Example: Barista

Manual Testing

Simplicity

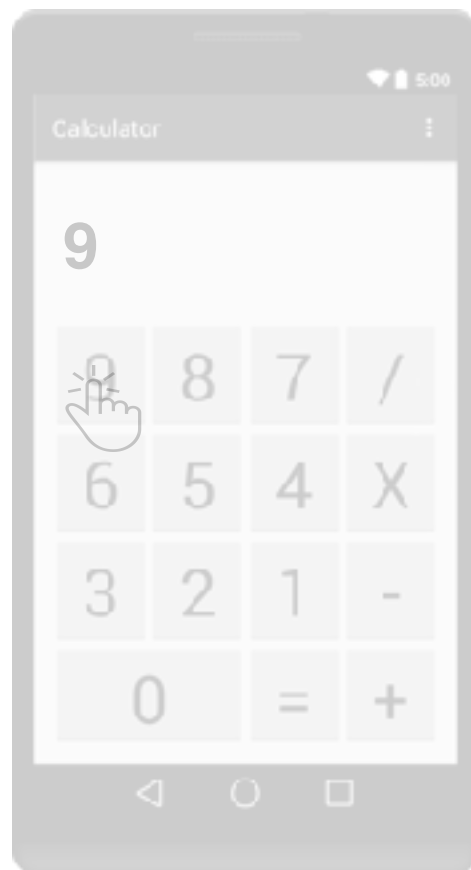


Barista



Automated Testing

Efficiency



```
1 public void divisionByZero() {
```

```
8 }
```

Running Example: Barista

Manual Testing

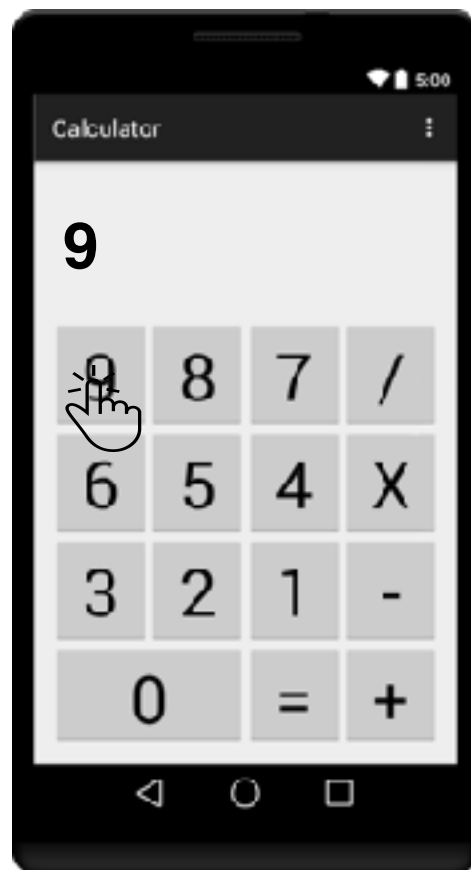
Simplicity



Barista

Automated Testing

Efficiency



```
1 public void divisionByZero() {  
2     onView(withXPath("/...")).perform(click());  
  
8 }
```

Running Example: Barista

Manual Testing

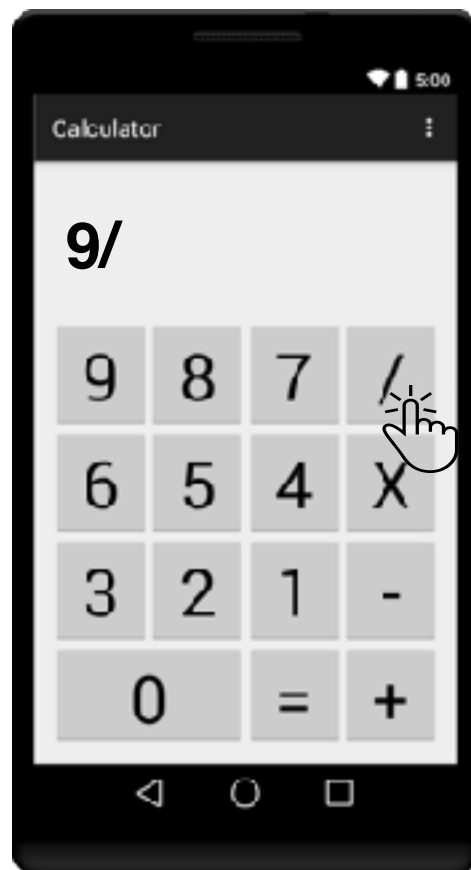
Simplicity



Barista

Automated Testing

Efficiency



```
1 public void divisionByZero() {  
2     onView(withXPath("/...")).perform(click());  
3     onView(withId(R.id.div)).perform(click());  
  
8 }
```

Running Example: Barista

Manual Testing

Simplicity



Barista

Automated Testing

Efficiency



```
1 public void divisionByZero() {  
2     onView(withXPath("/...")).perform(click());  
3     onView(withId(R.id.div)).perform(click());  
4     onView(withXPath("/...")).perform(click());  
  
8 }
```

Running Example: Barista

Manual Testing

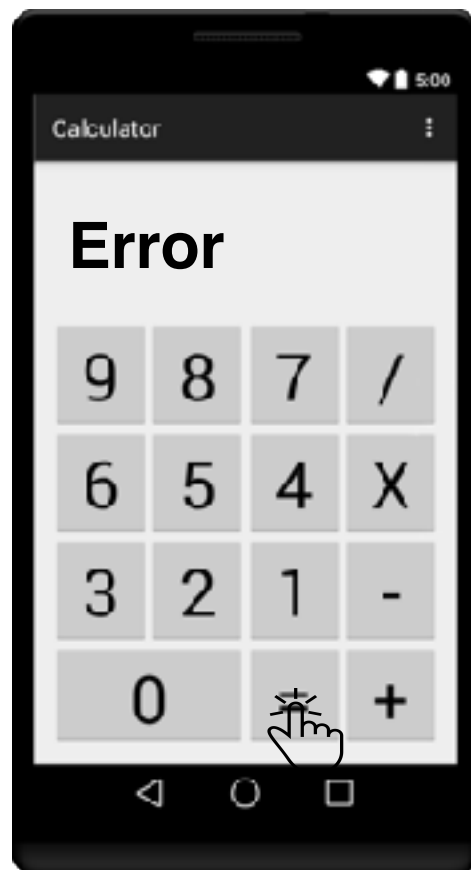
Simplicity



Barista

Automated Testing

Efficiency



```
1 public void divisionByZero() {  
2     onView(withXPath("/...")).perform(click());  
3     onView(withId(R.id.div)).perform(click());  
4     onView(withXPath("/...")).perform(click());  
5     onView(withId(R.id.eq)).perform(click());  
  
8 }
```

Running Example: Barista

Manual Testing

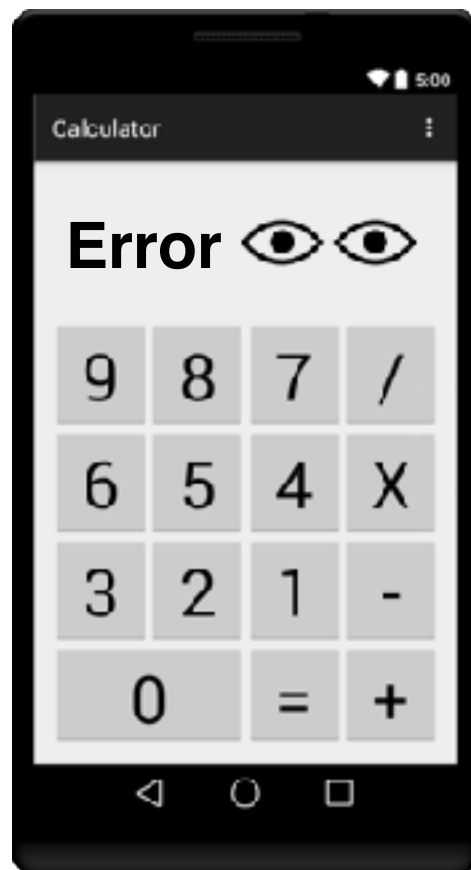
Simplicity



Barista

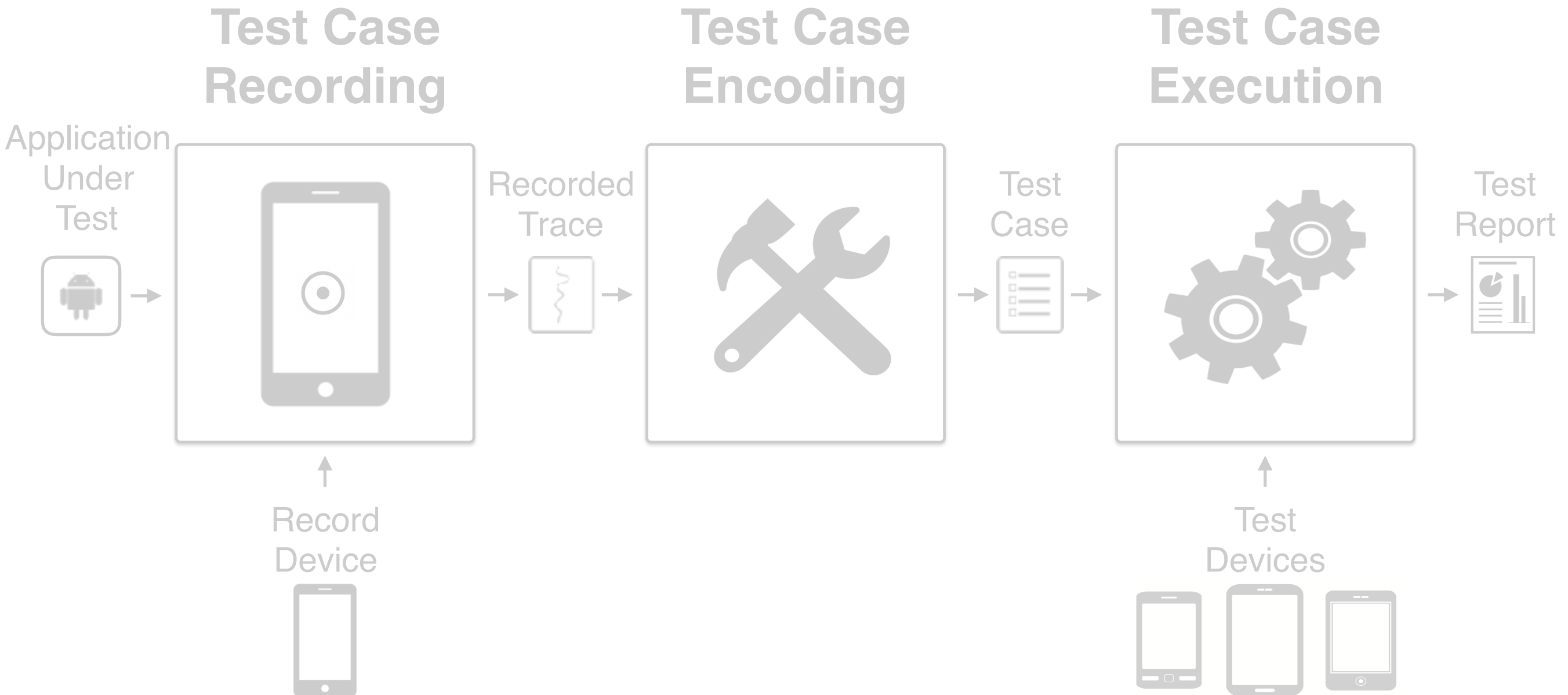
Automated Testing

Efficiency

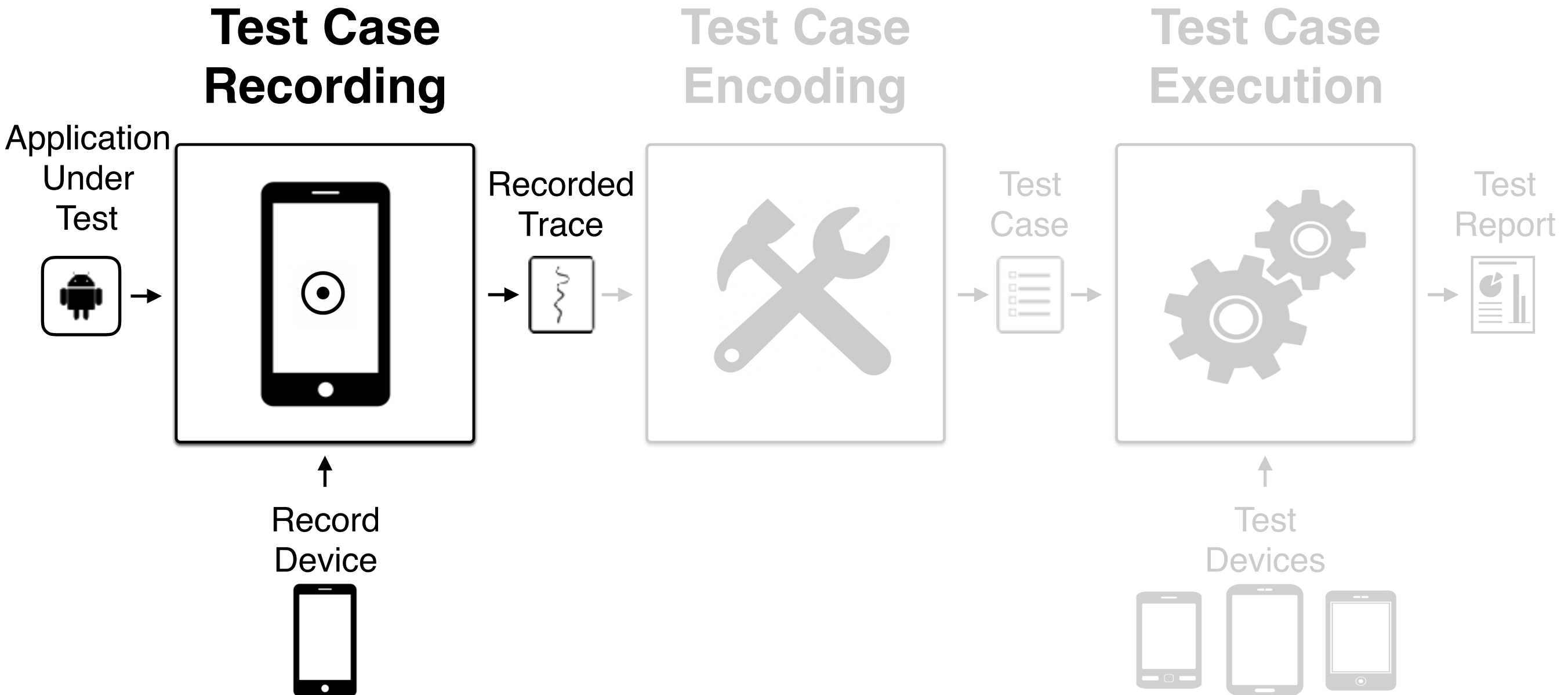


```
1 public void divisionByZero() {  
2     onView(withXPath("/...")).perform(click());  
3     onView(withId(R.id.div)).perform(click());  
4     onView(withXPath("/...")).perform(click());  
5     onView(withId(R.id.eq)).perform(click());  
6     onView(withId(R.id.dsp))  
7         .check(matches(withText("Error")));  
8 }
```

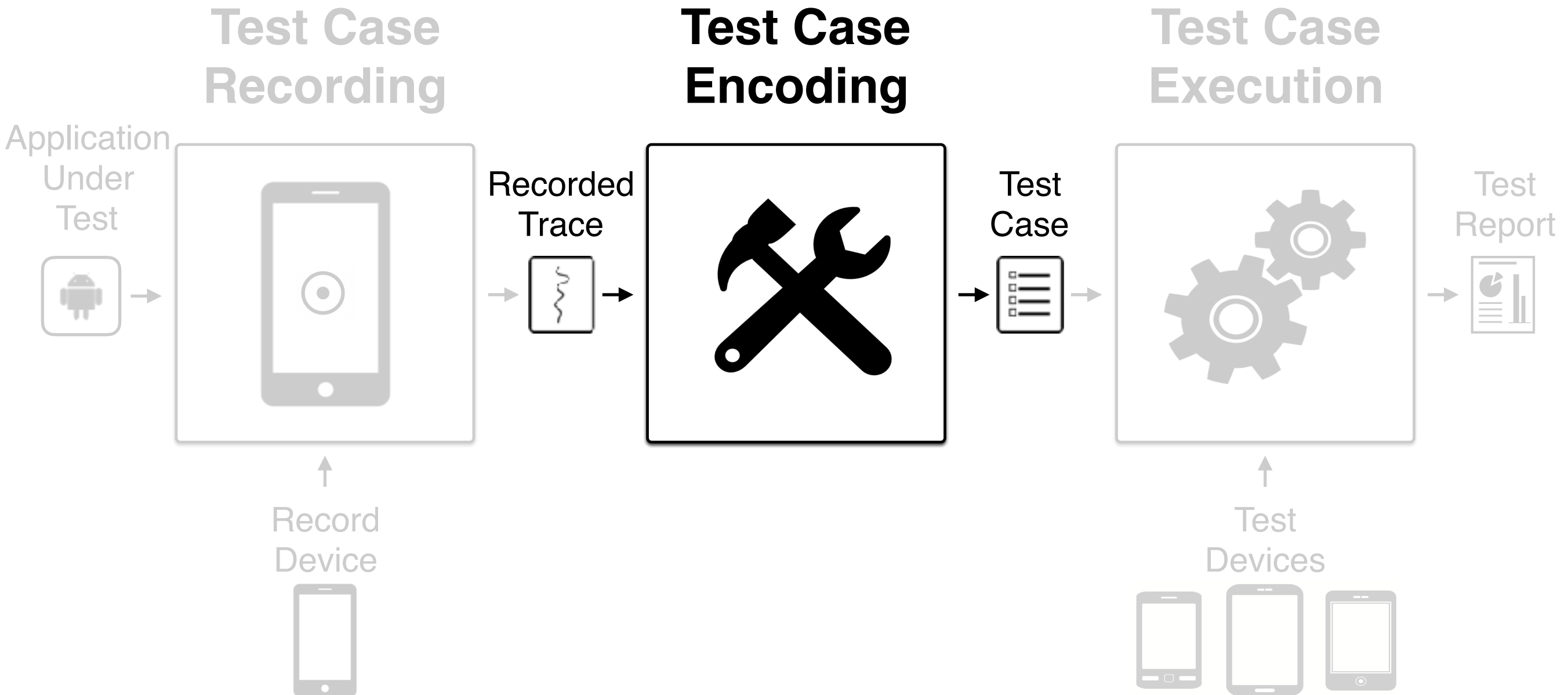
Barista Overview



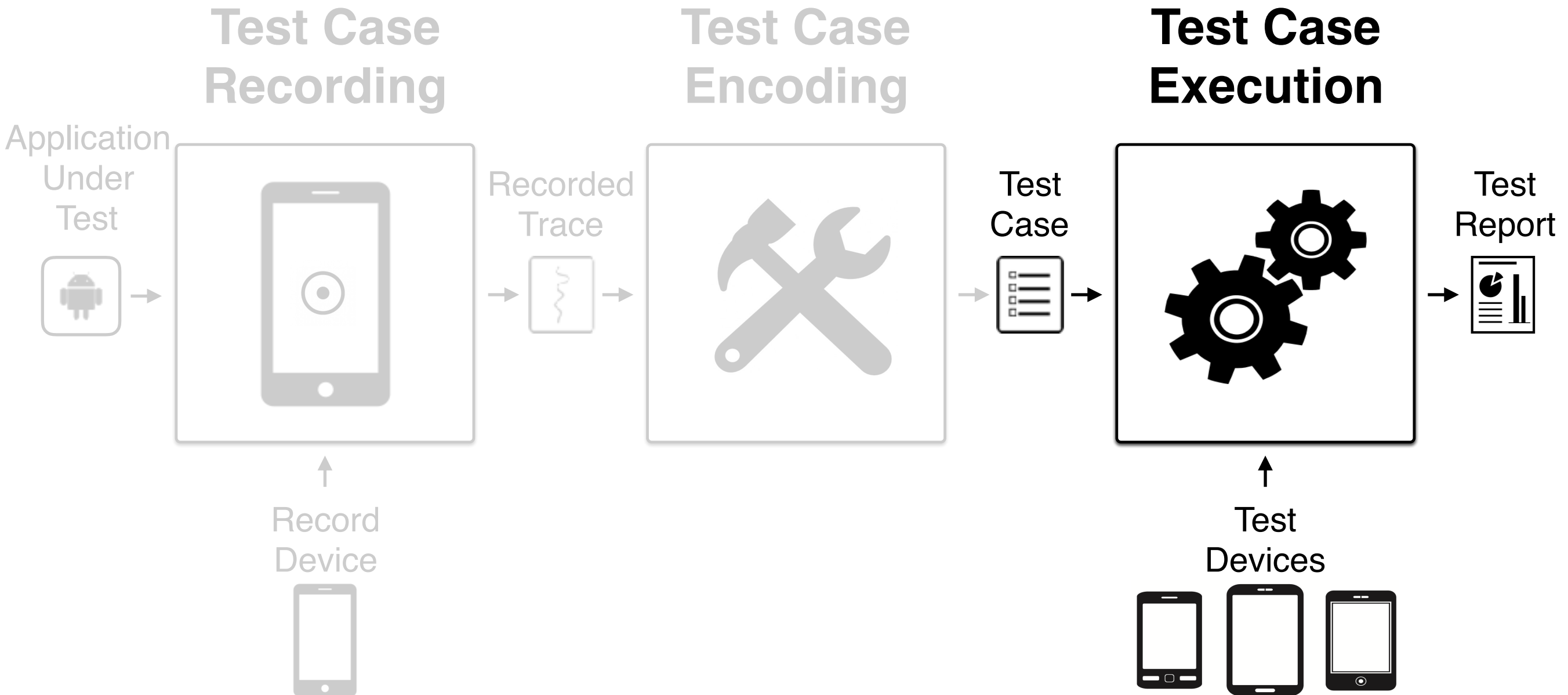
Barista Overview



Barista Overview



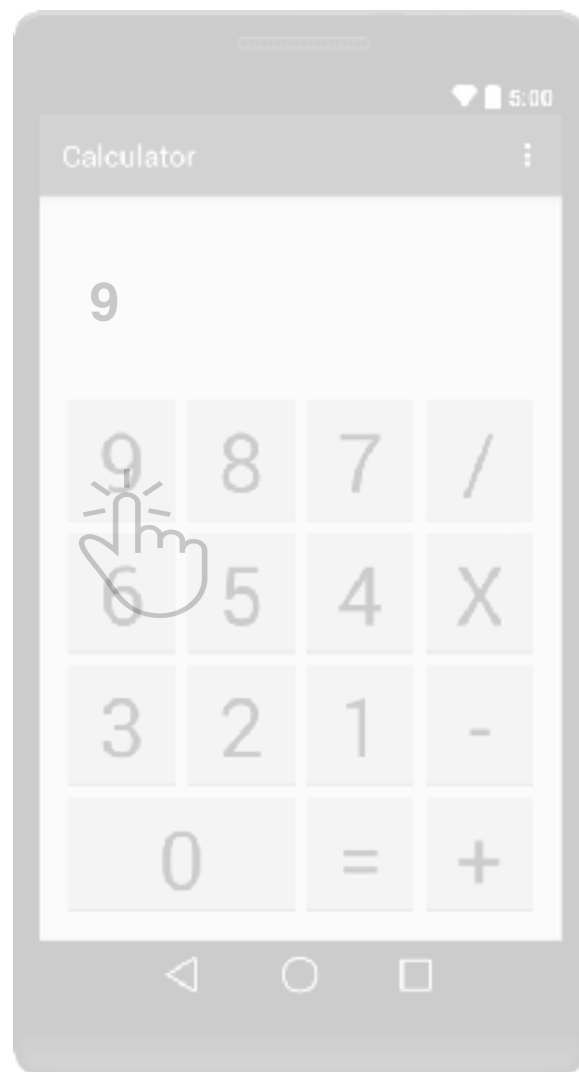
Barista Overview



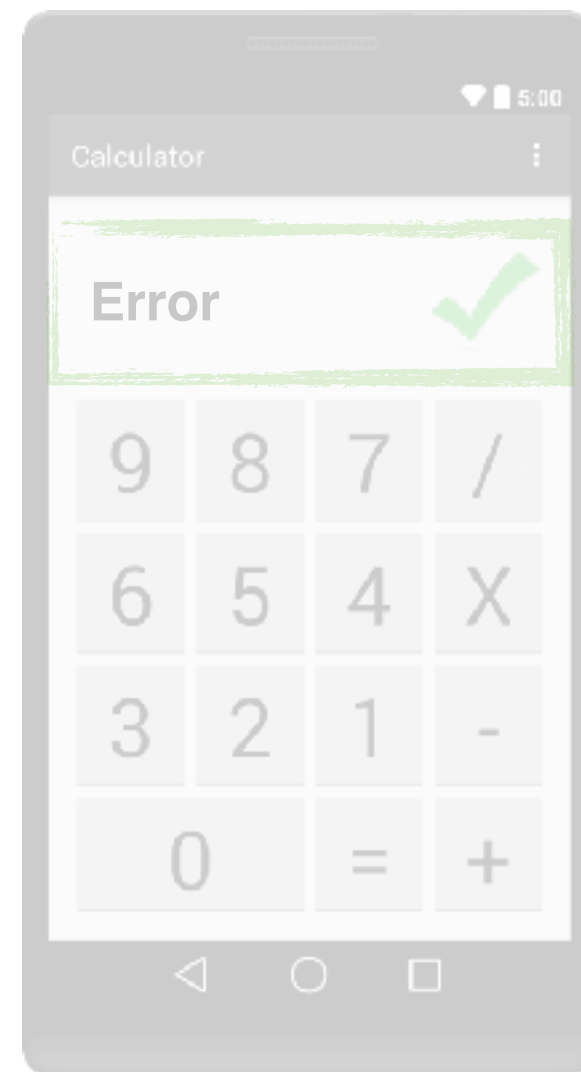
Test Case Recording

Objective: record actions performed on the AUT

Actions



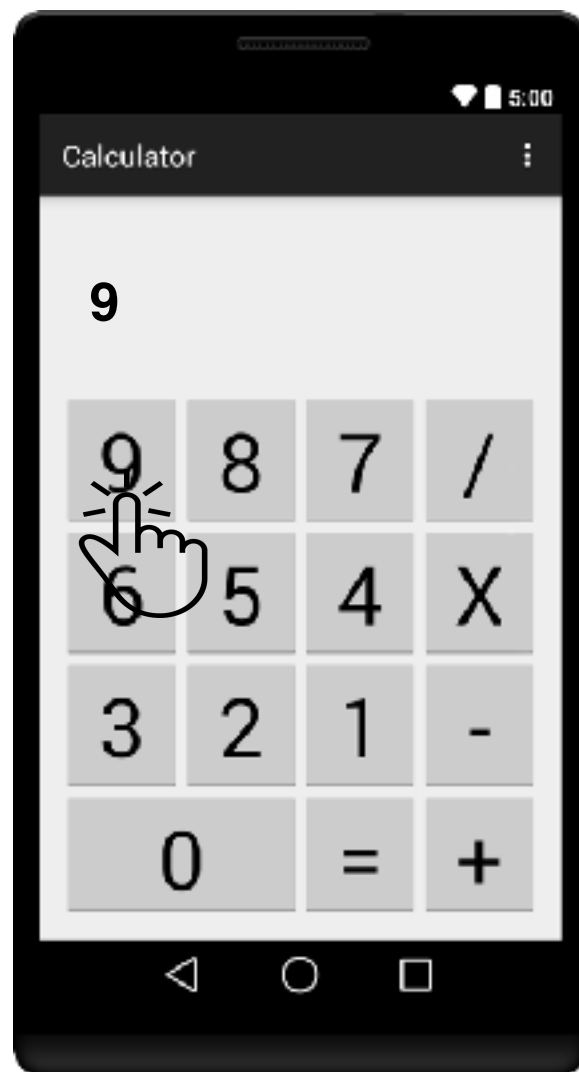
Oracles



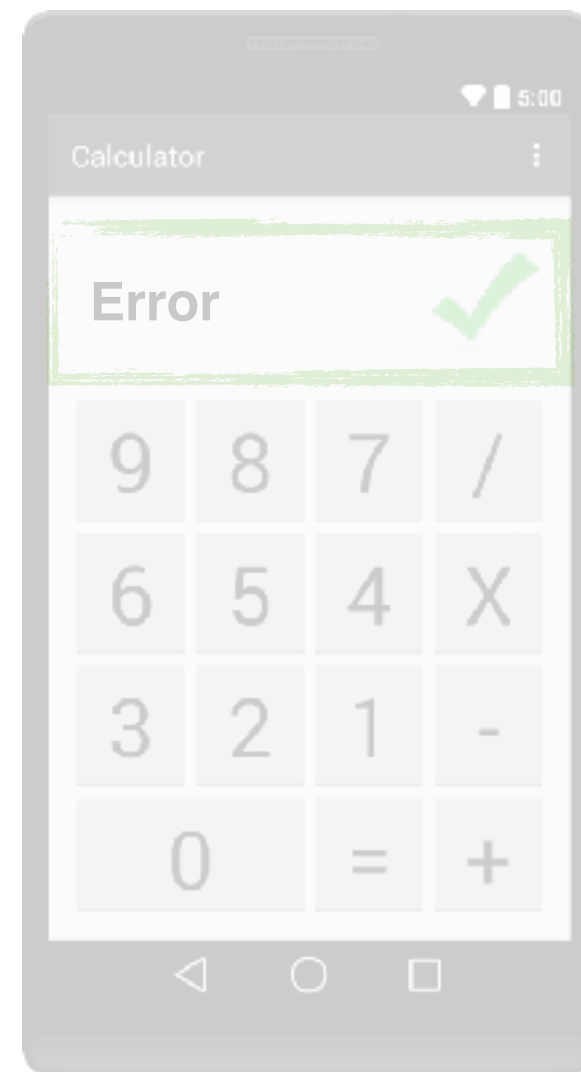
Test Case Recording

Objective: record actions performed on the AUT

Actions



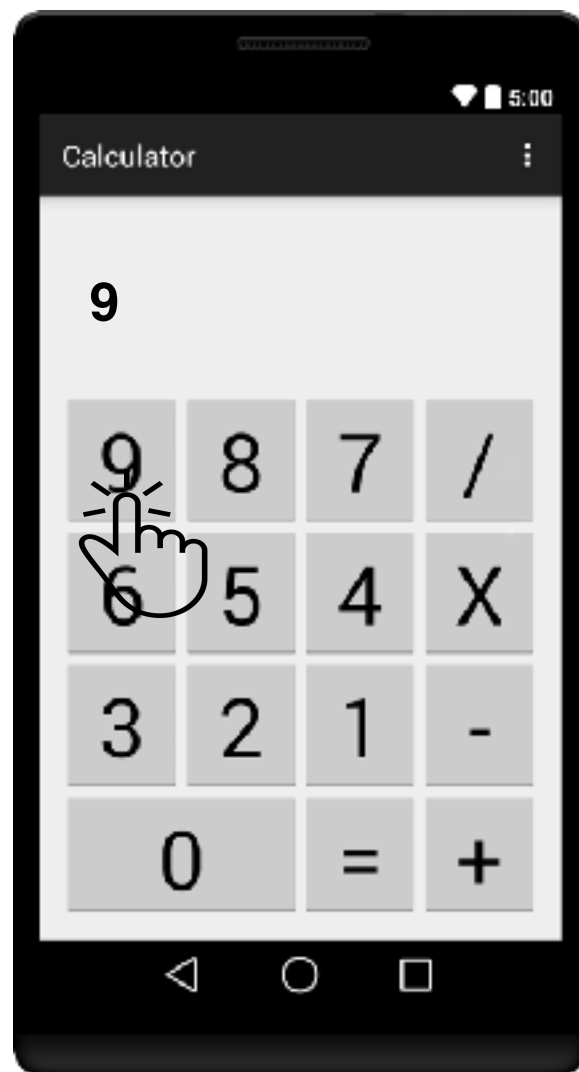
Oracles



Test Case Recording

Objective: record actions performed on the AUT

Actions

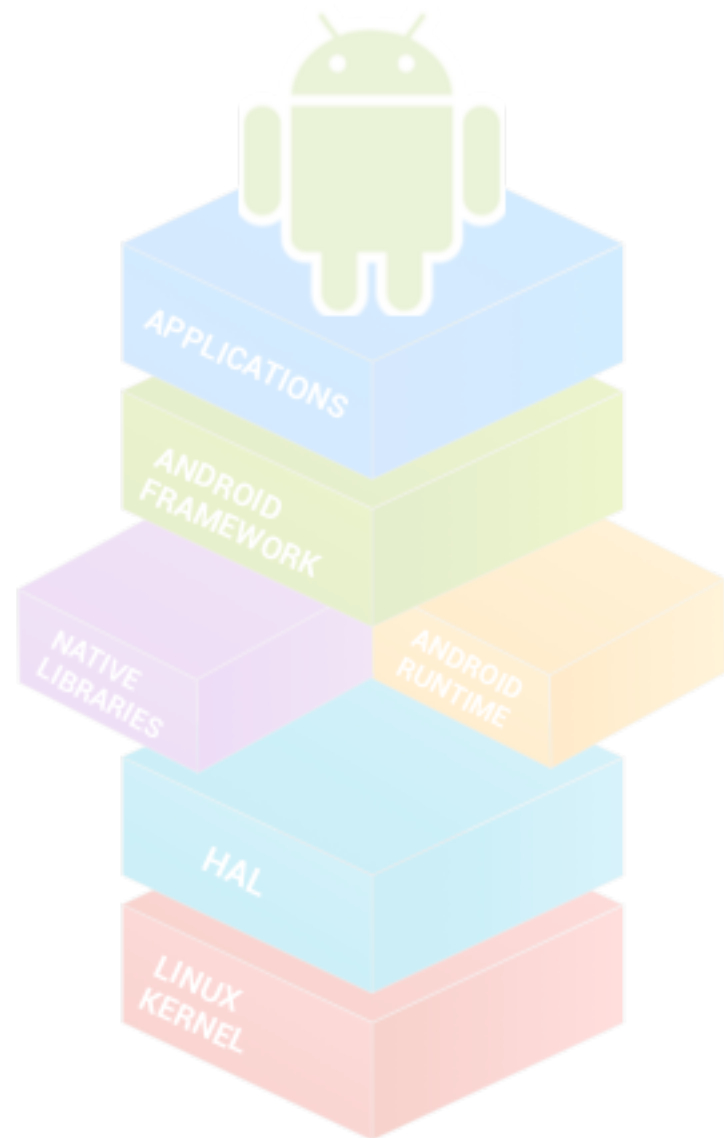


Oracles

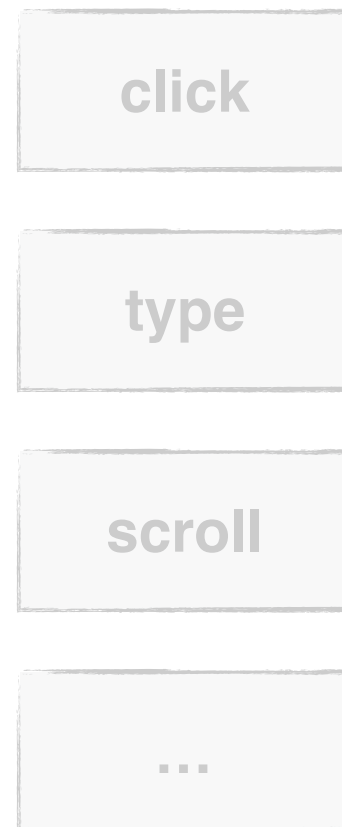


Test Case Recording: Elements

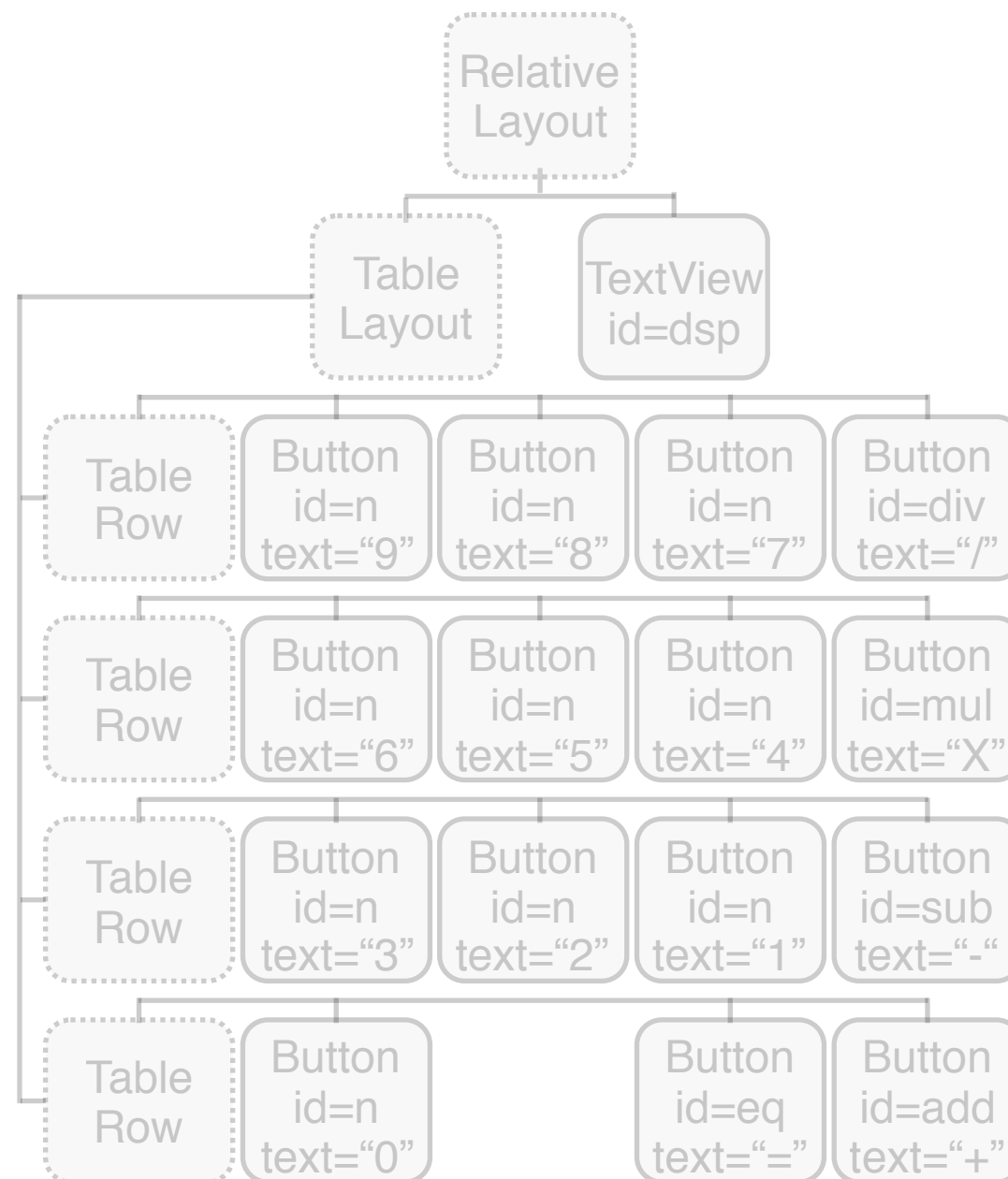
Accessibility



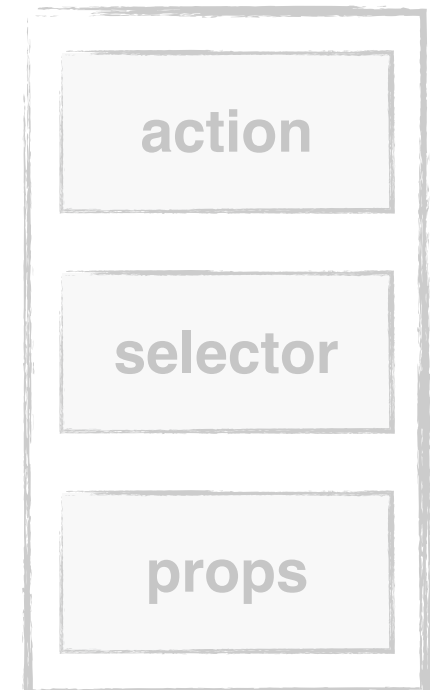
Actions



Selector



Interaction



Recorded Trace

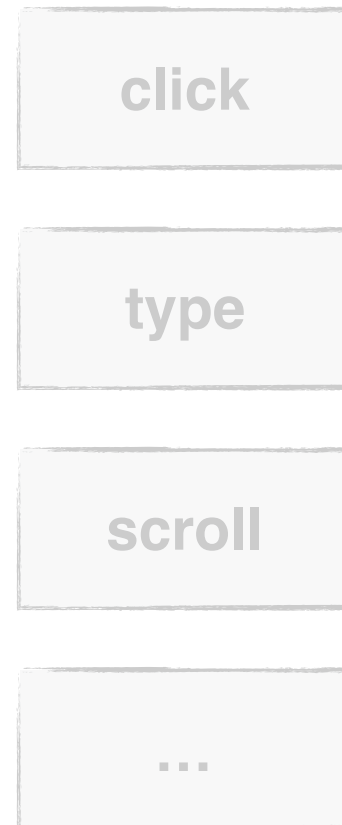


Test Case Recording: Elements

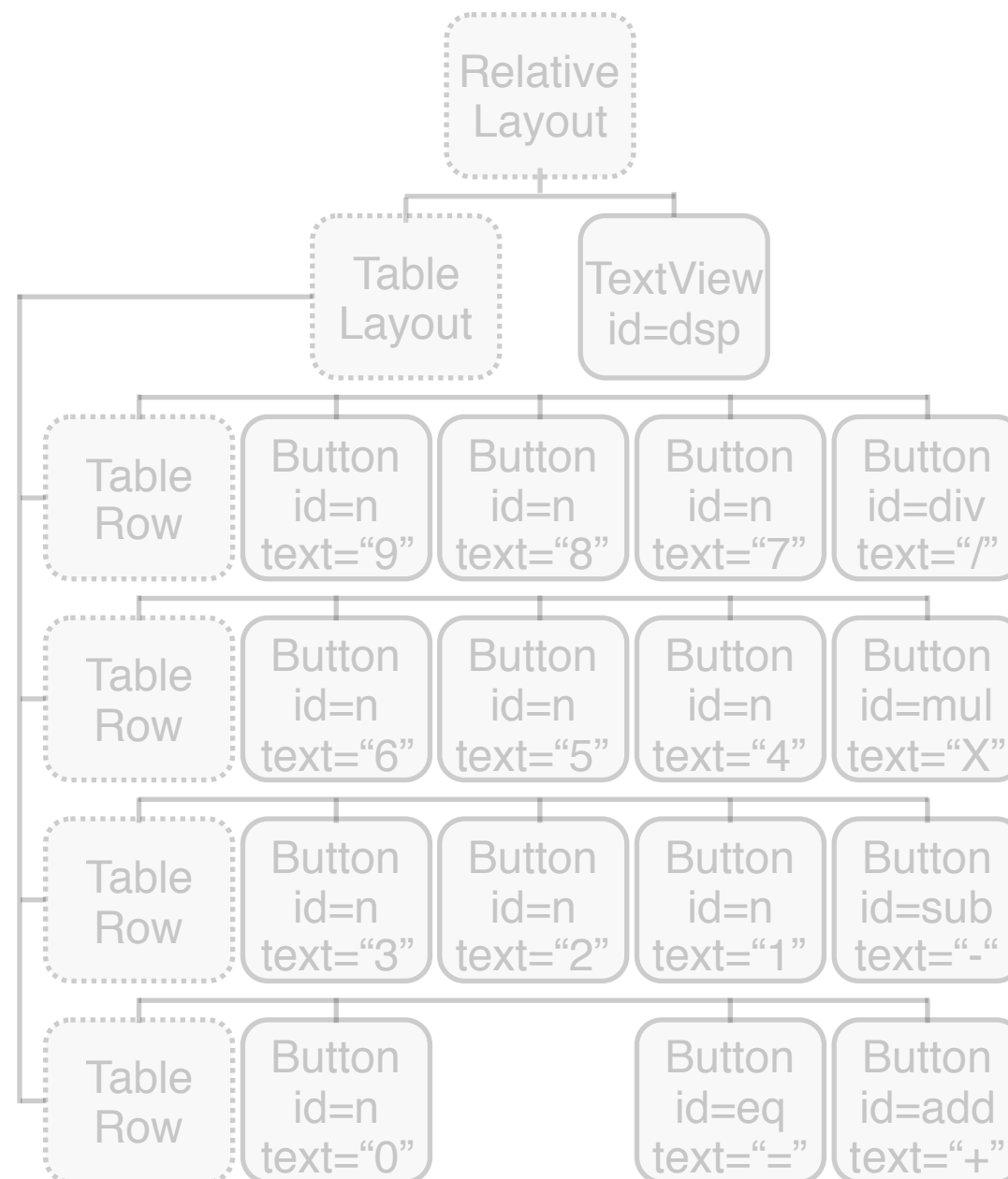
Accessibility



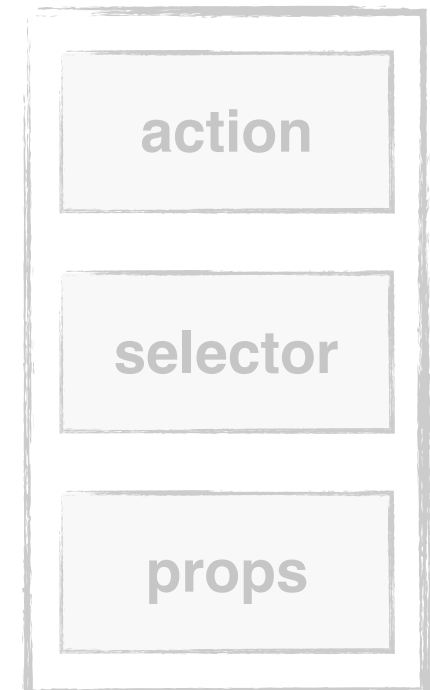
Actions



Selector



Interaction



Recorded Trace

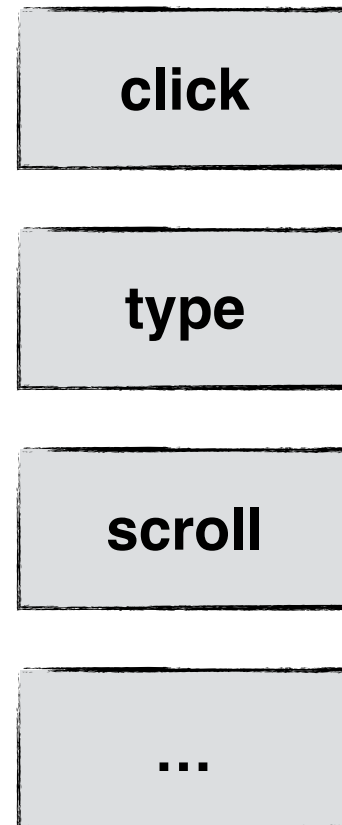


Test Case Recording: Elements

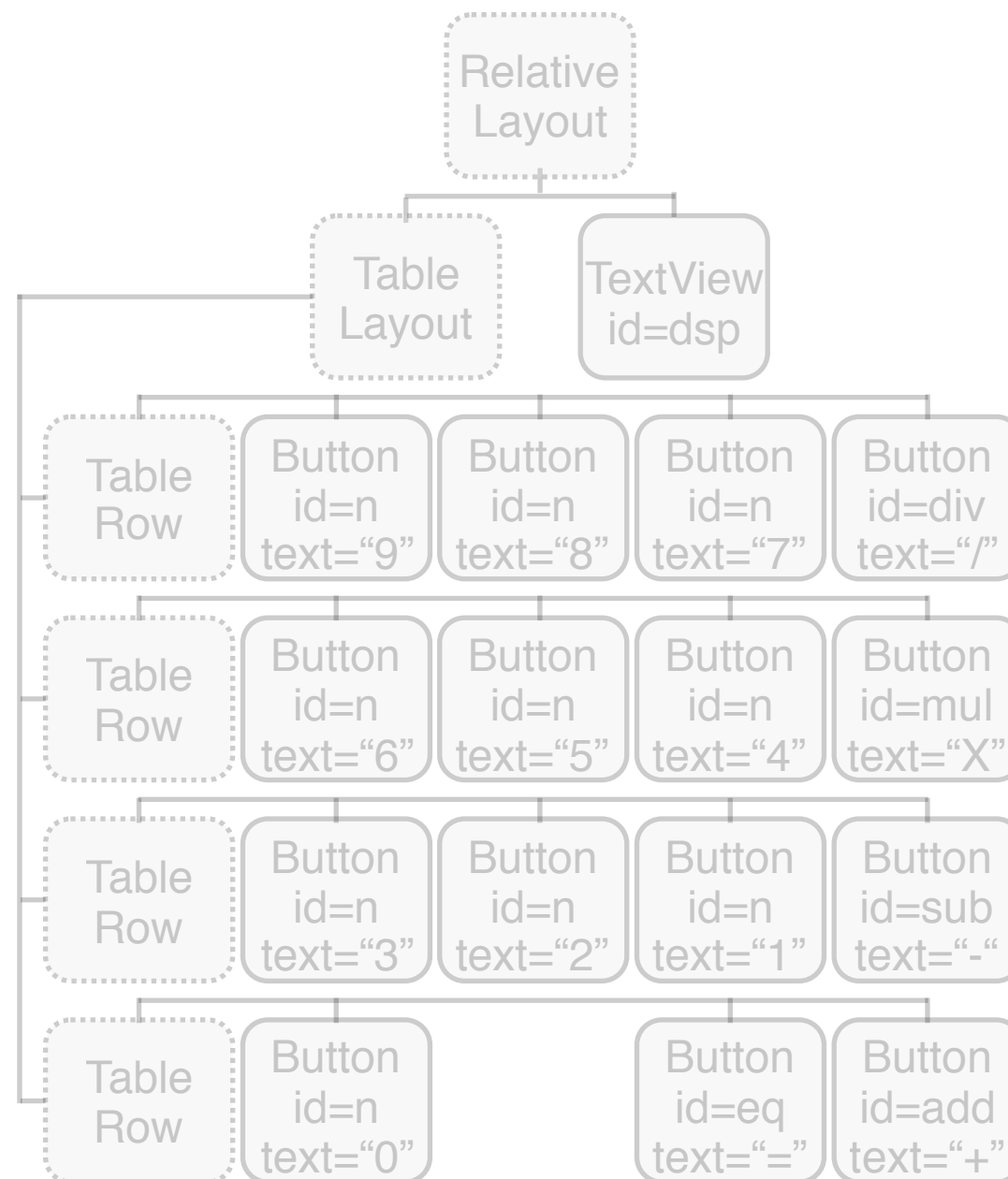
Accessibility



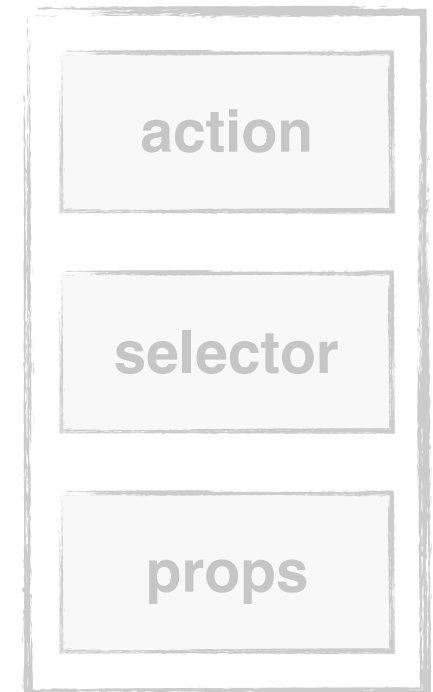
Actions



Selector



Interaction



Recorded Trace

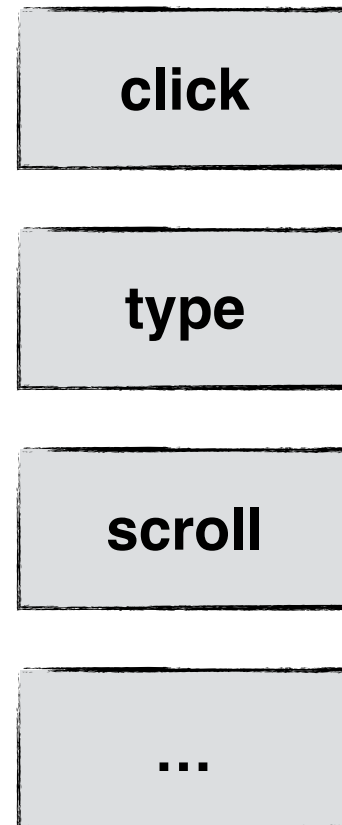


Test Case Recording: Elements

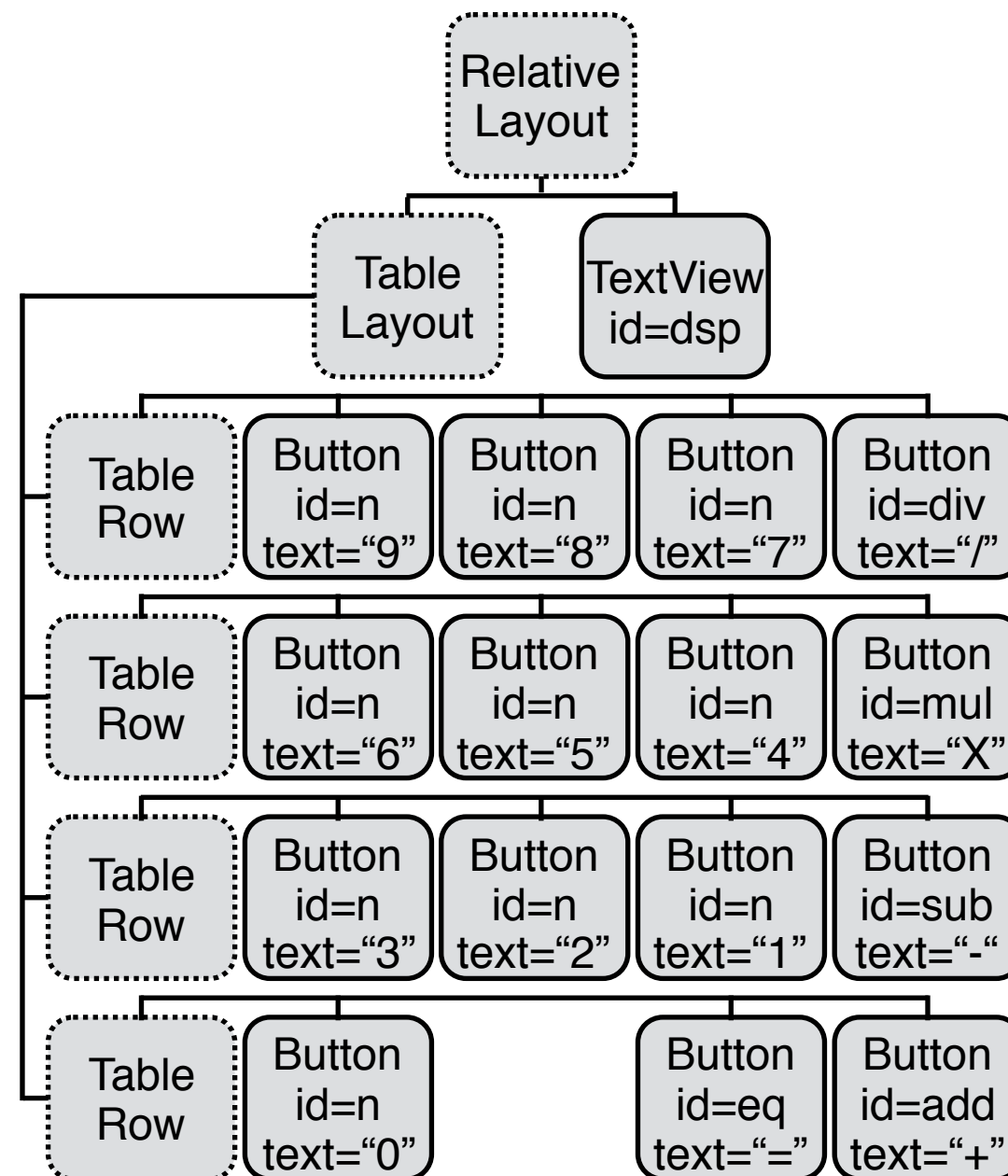
Accessibility



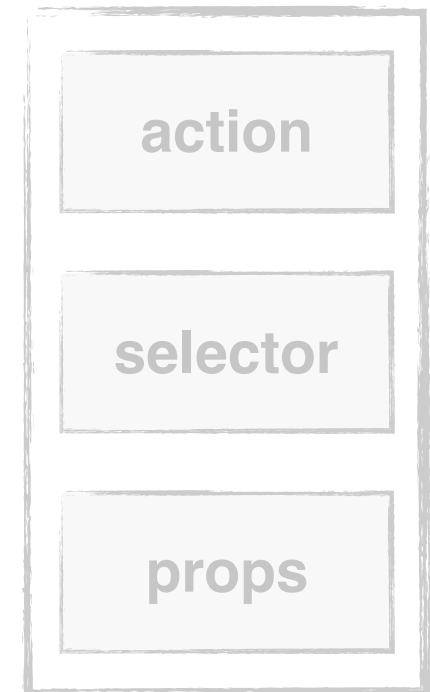
Actions



Selector



Interaction



Recorded Trace

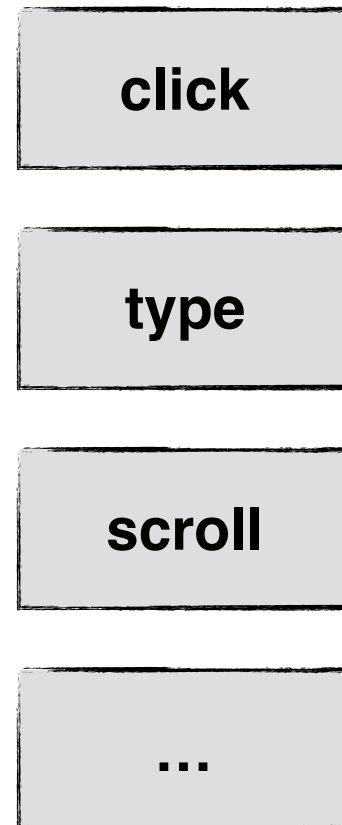


Test Case Recording: Elements

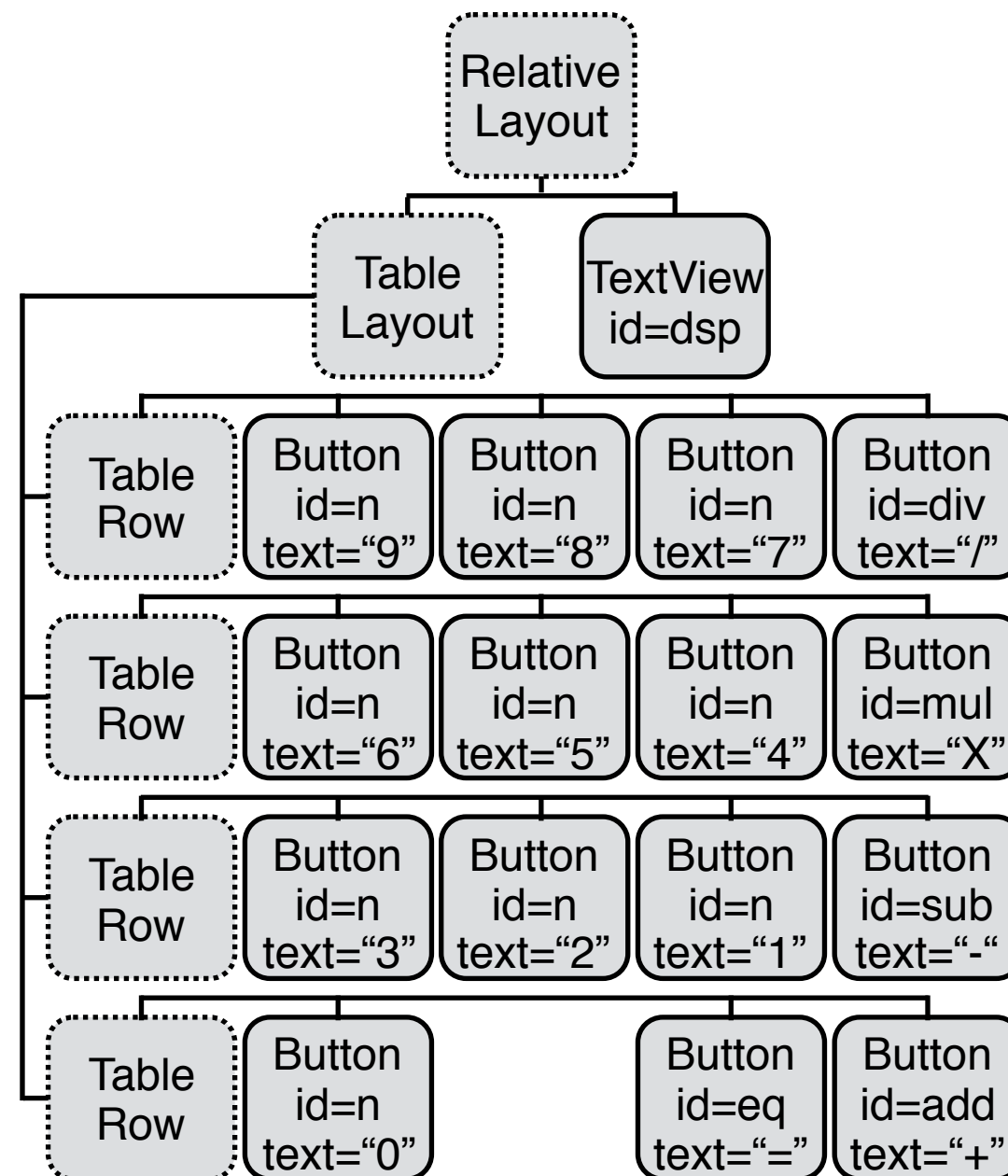
Accessibility



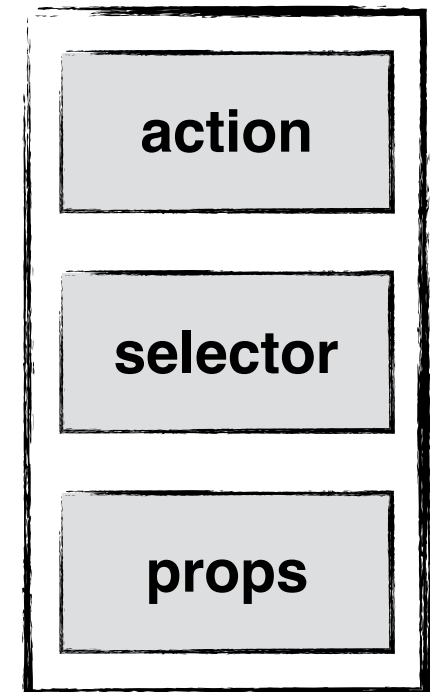
Actions



Selector



Interaction



↓
Recorded
Trace

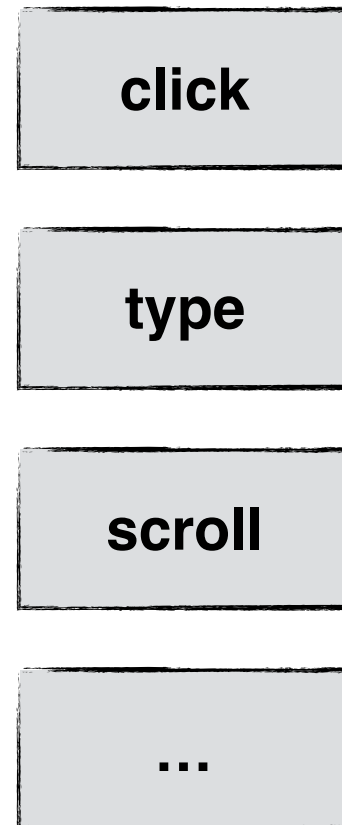


Test Case Recording: Elements

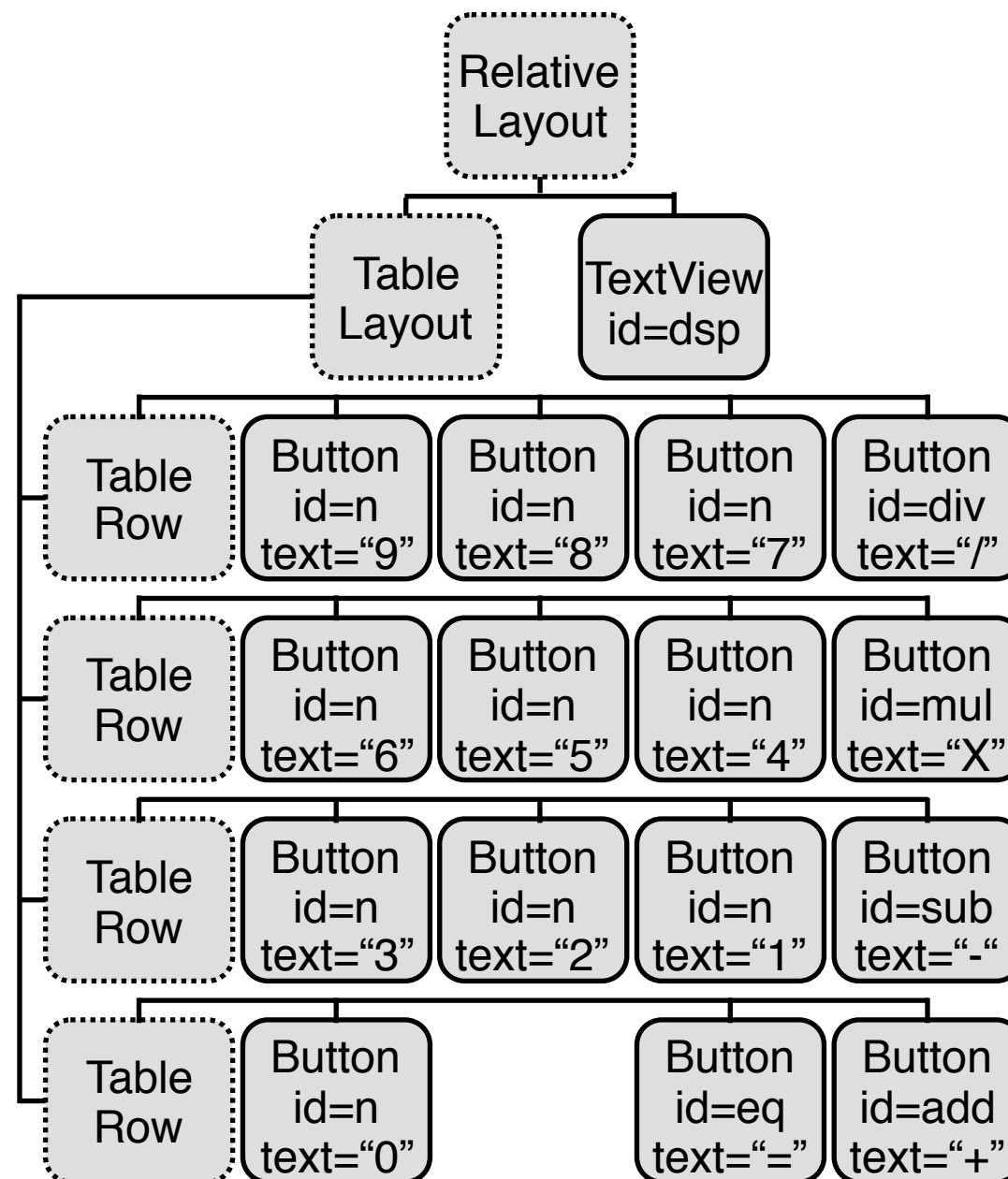
Accessibility



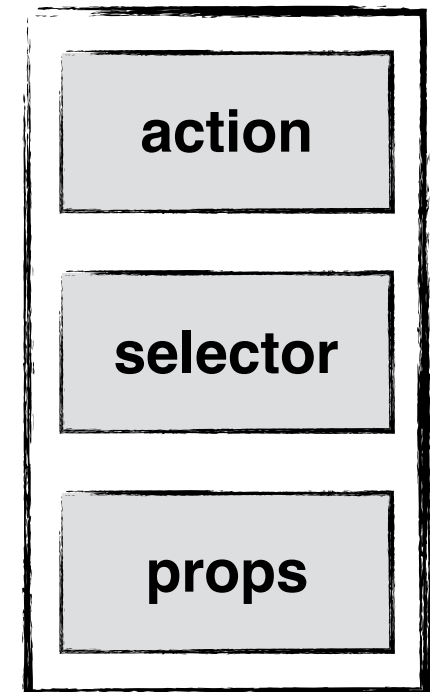
Actions



Selector



Interaction

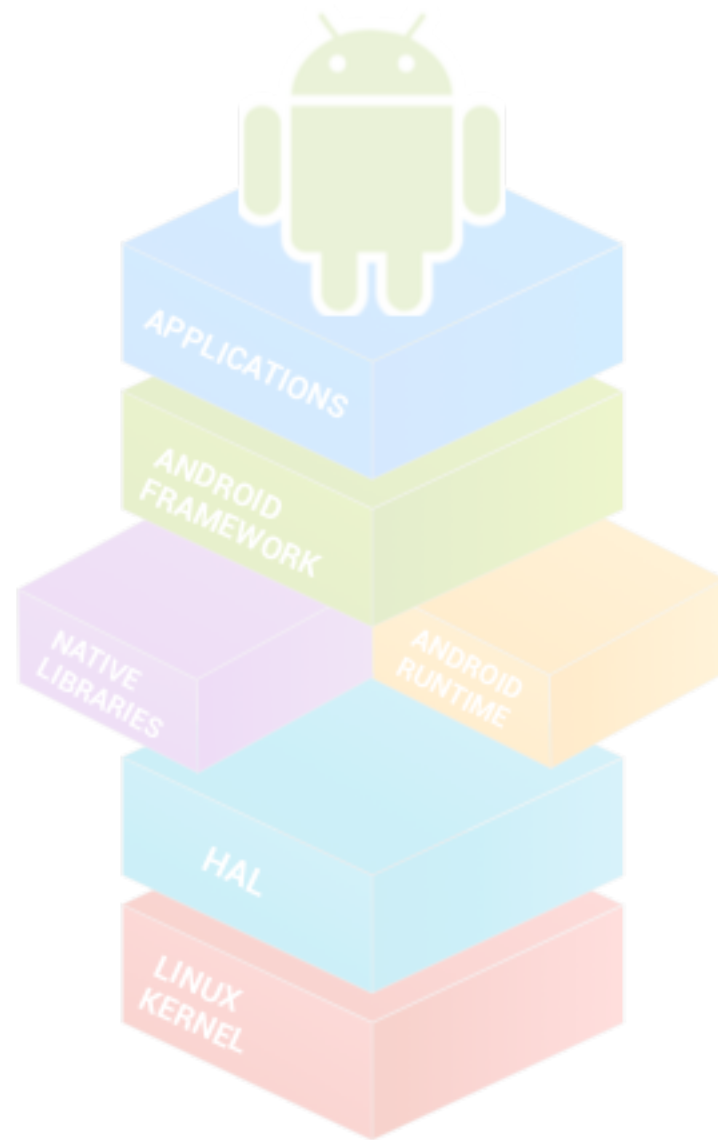


Recorded Trace

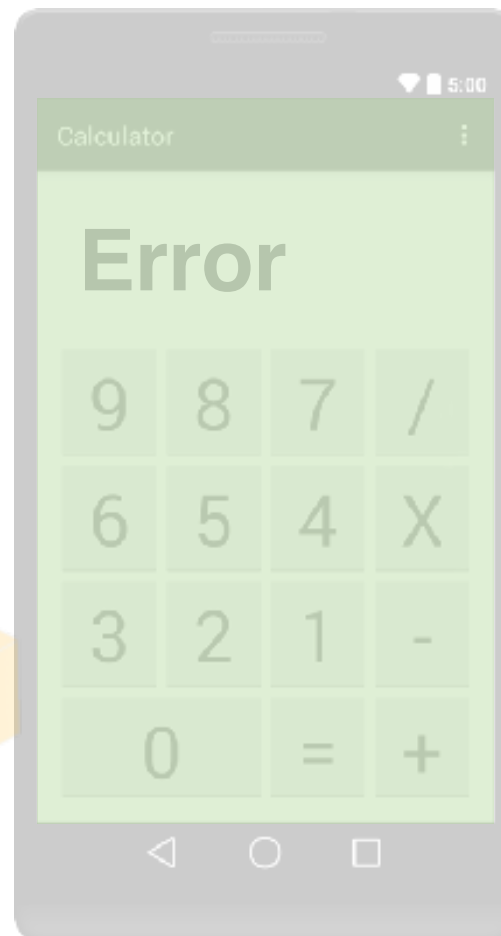


Test Case Recording: Oracles

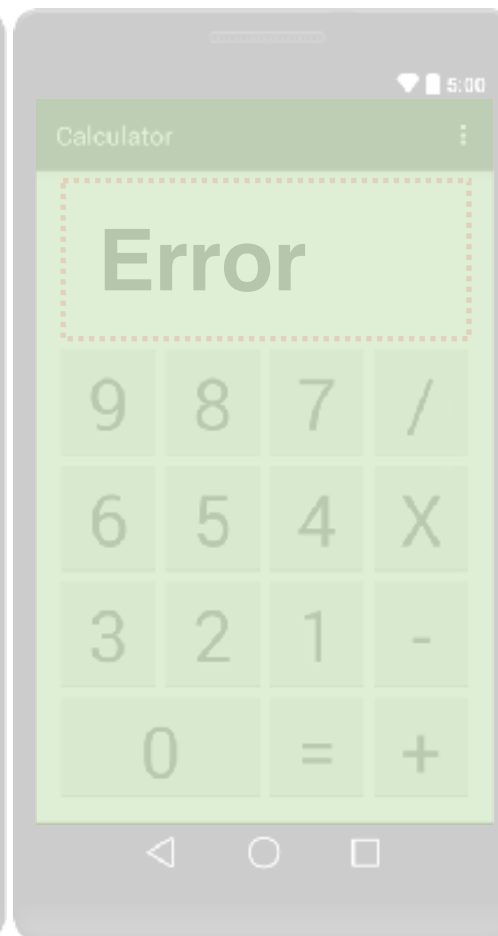
Accessibility



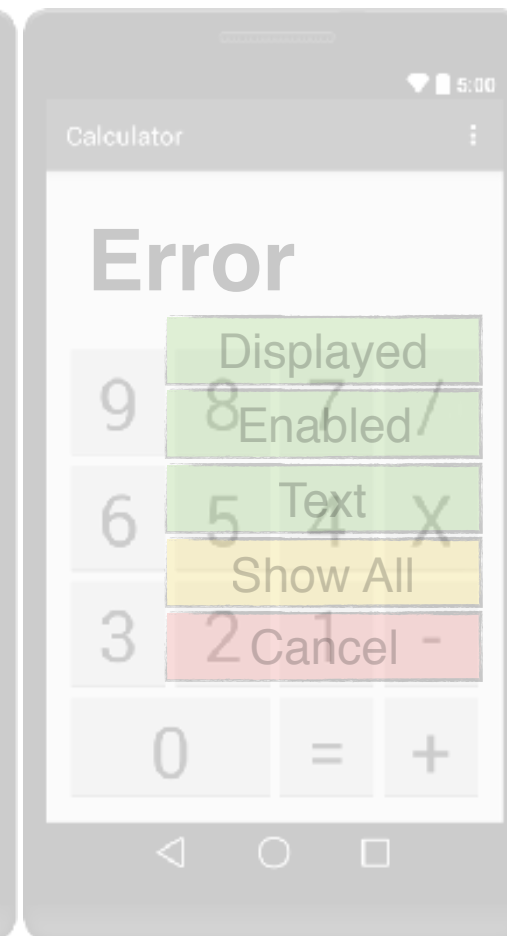
Overlay



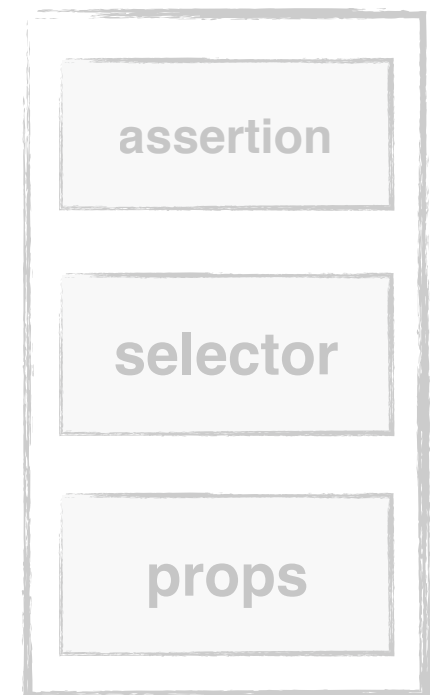
Selection



Assertion



Oracle



Recorded
Trace

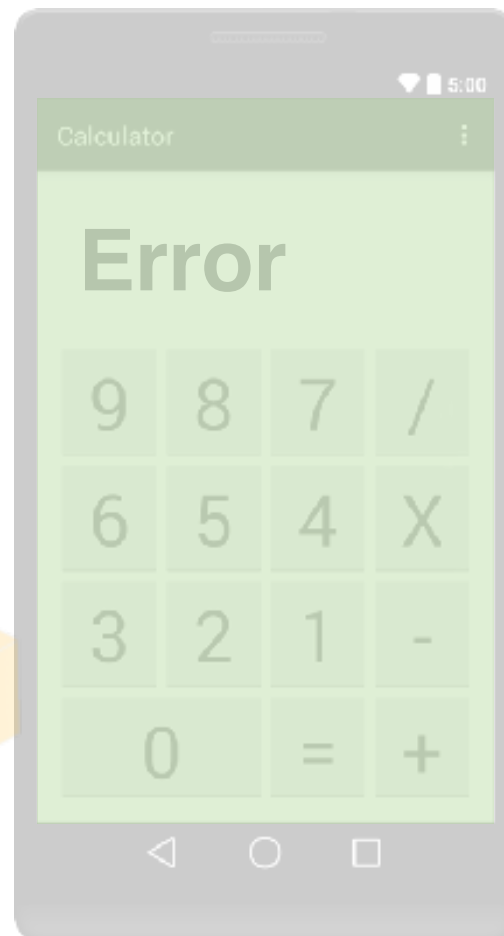


Test Case Recording: Oracles

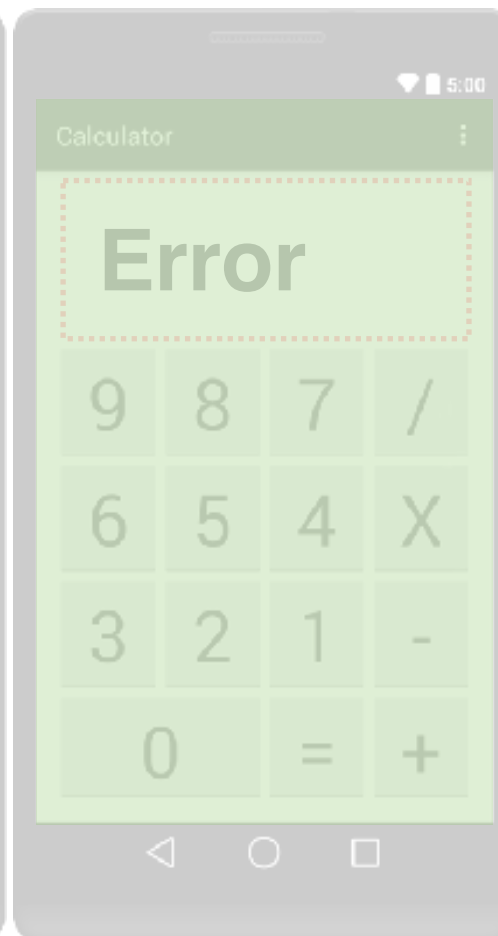
Accessibility



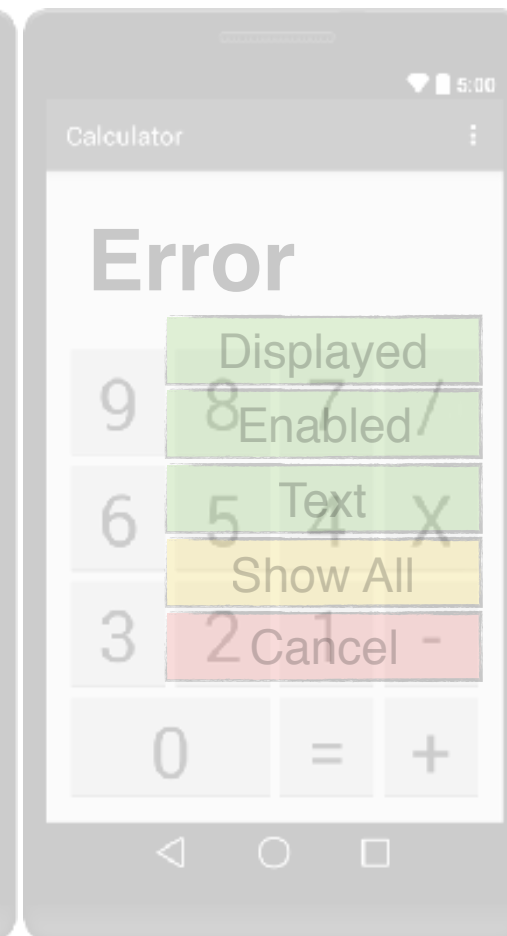
Overlay



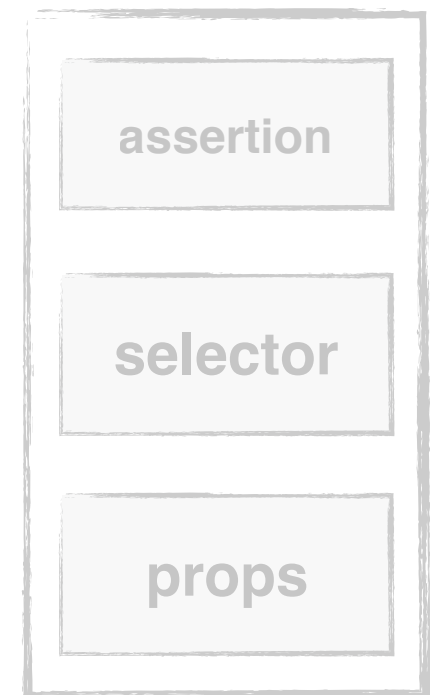
Selection



Assertion



Oracle



**Recorded
Trace**

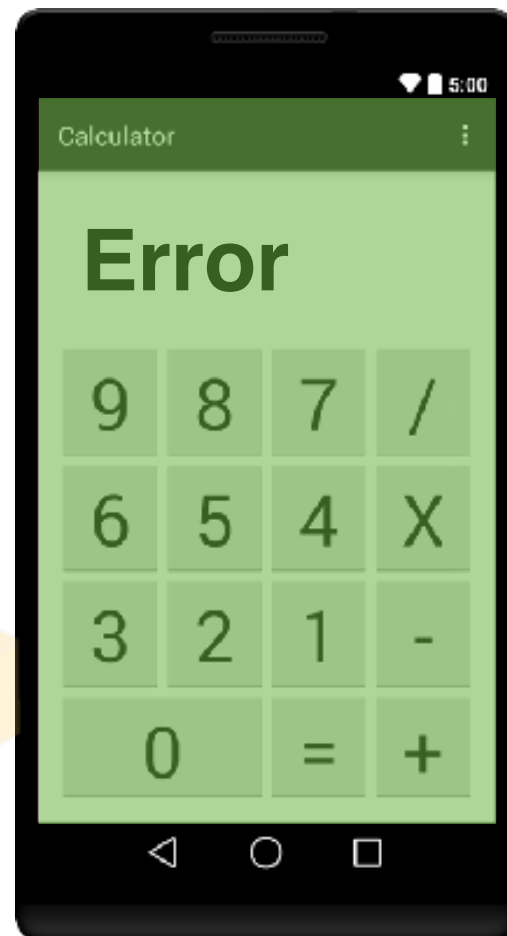


Test Case Recording: Oracles

Accessibility



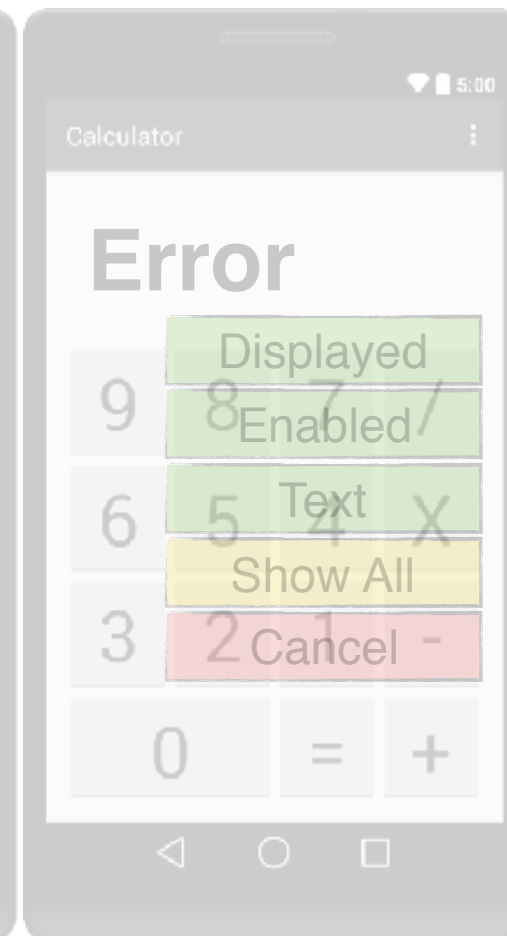
Overlay



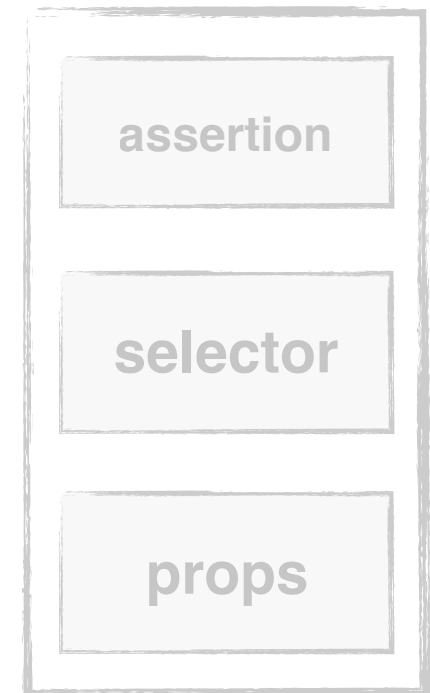
Selection



Assertion



Oracle



**Recorded
Trace**

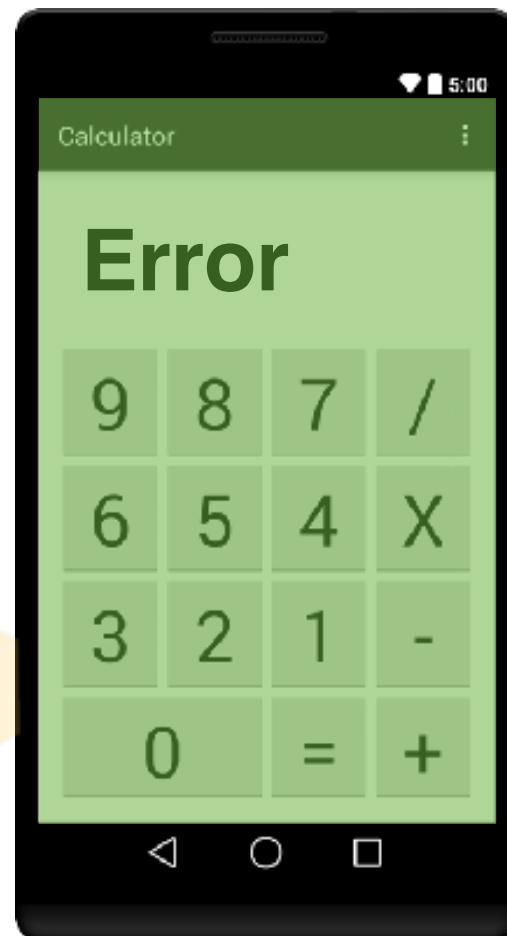


Test Case Recording: Oracles

Accessibility



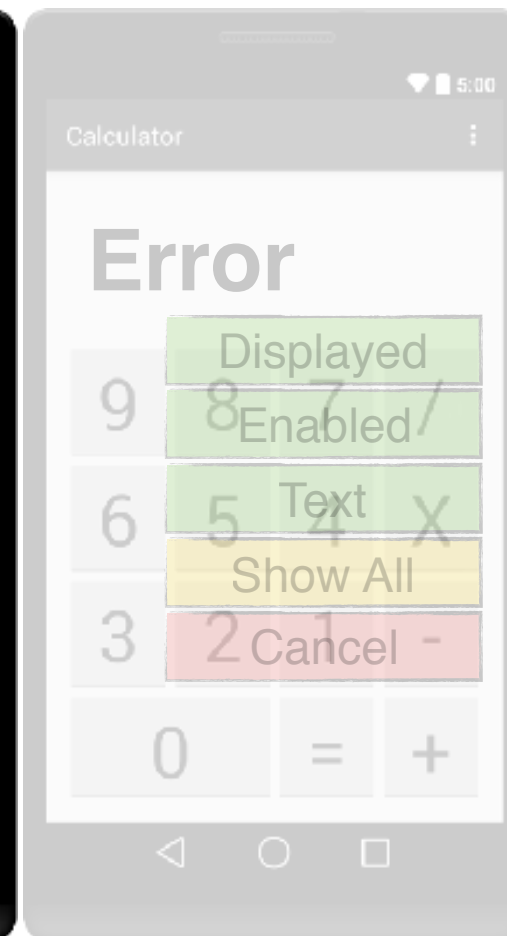
Overlay



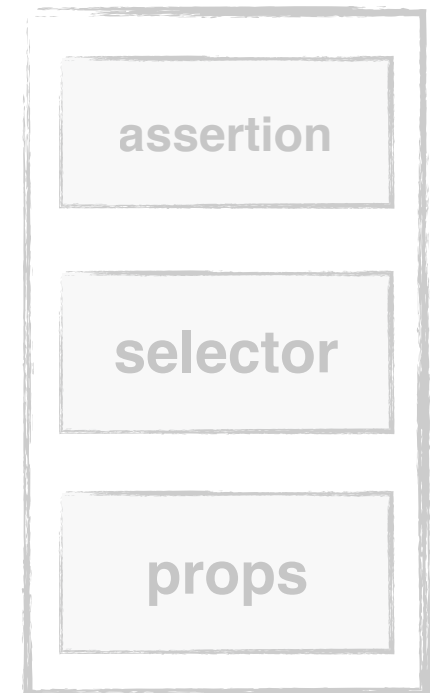
Selection



Assertion



Oracle



↓
**Recorded
Trace**

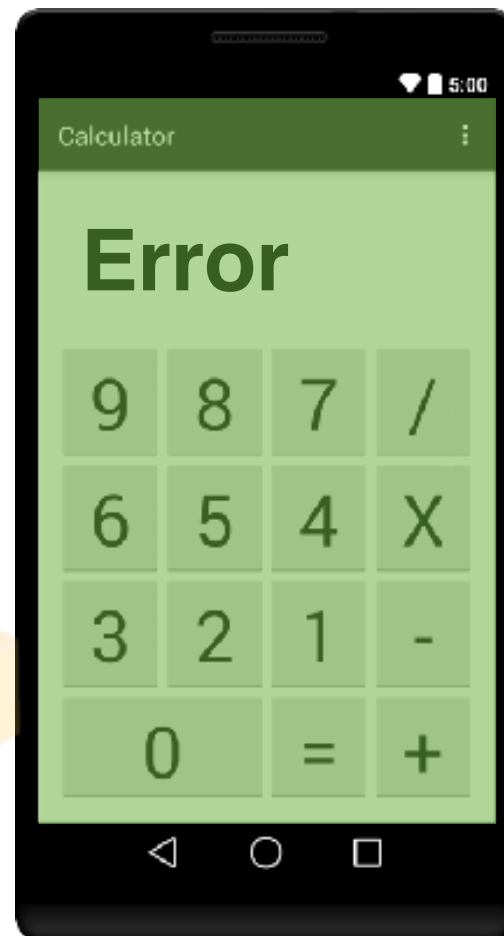


Test Case Recording: Oracles

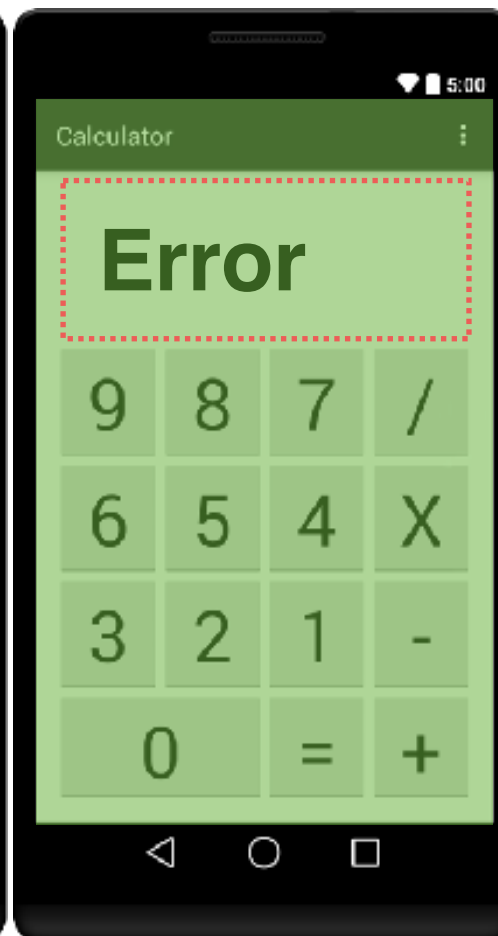
Accessibility



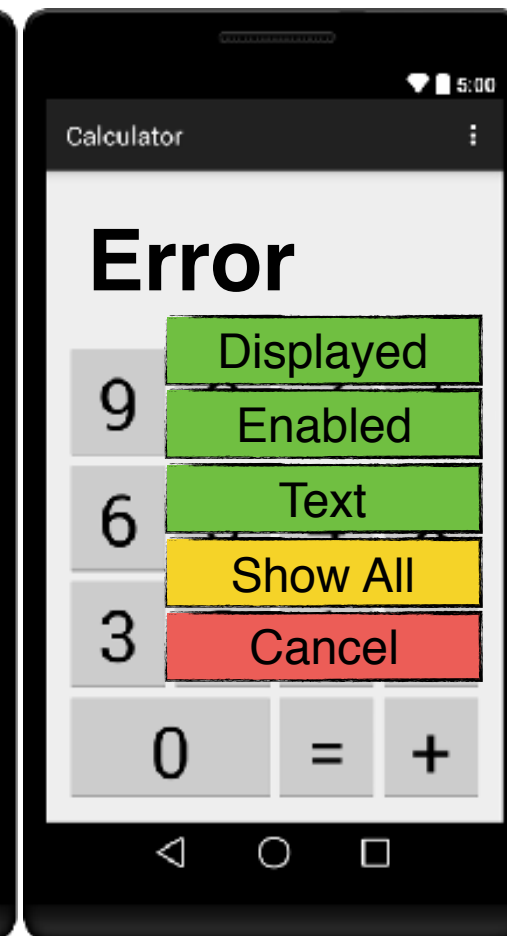
Overlay



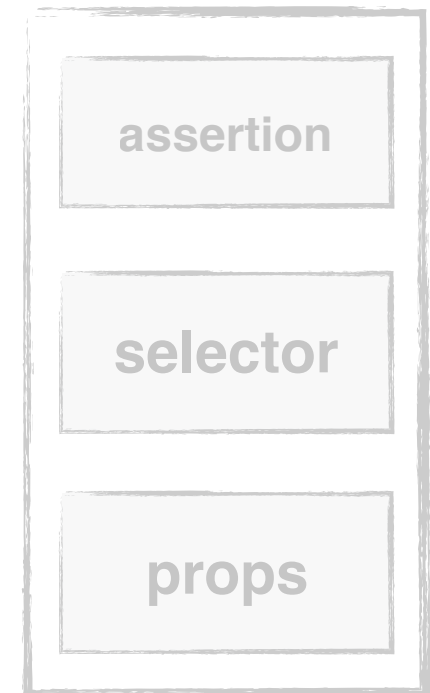
Selection



Assertion



Oracle



↓
**Recorded
Trace**

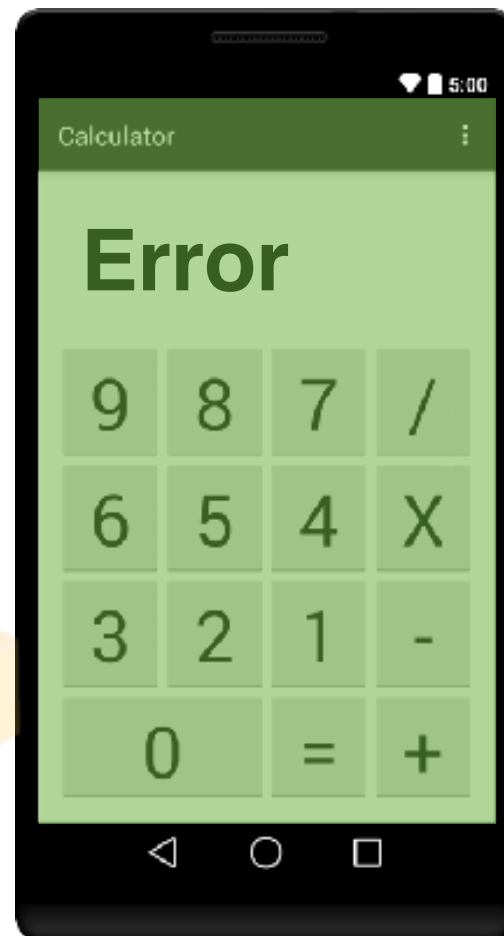


Test Case Recording: Oracles

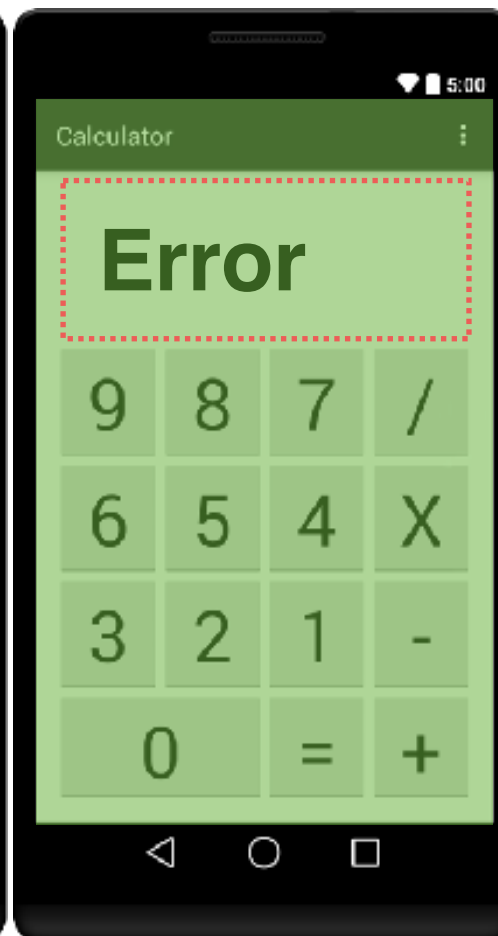
Accessibility



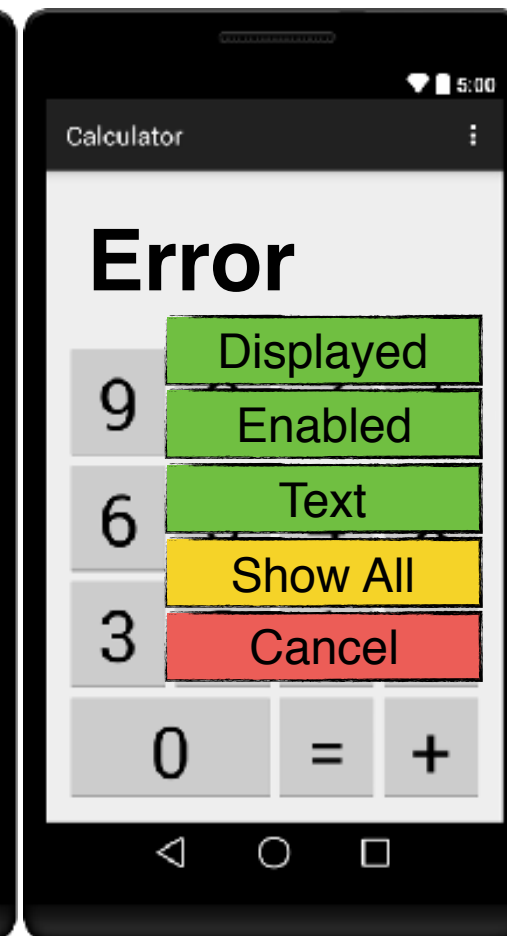
Overlay



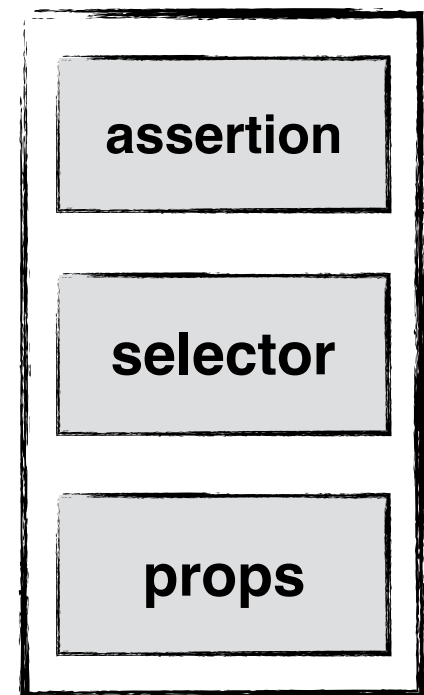
Selection



Assertion



Oracle



↓
**Recorded
Trace**

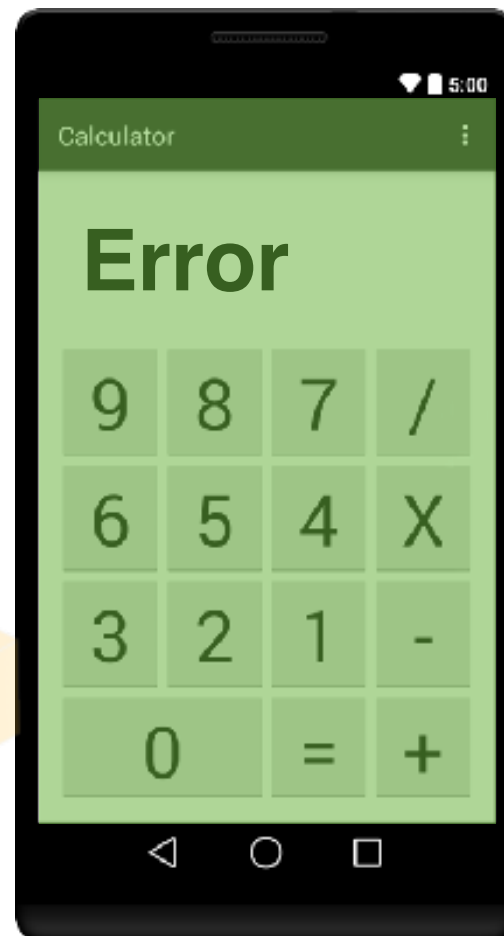


Test Case Recording: Oracles

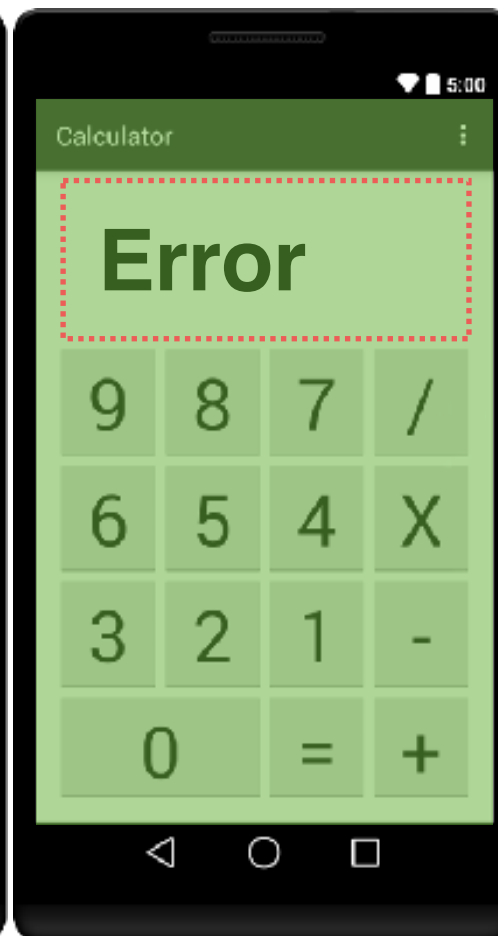
Accessibility



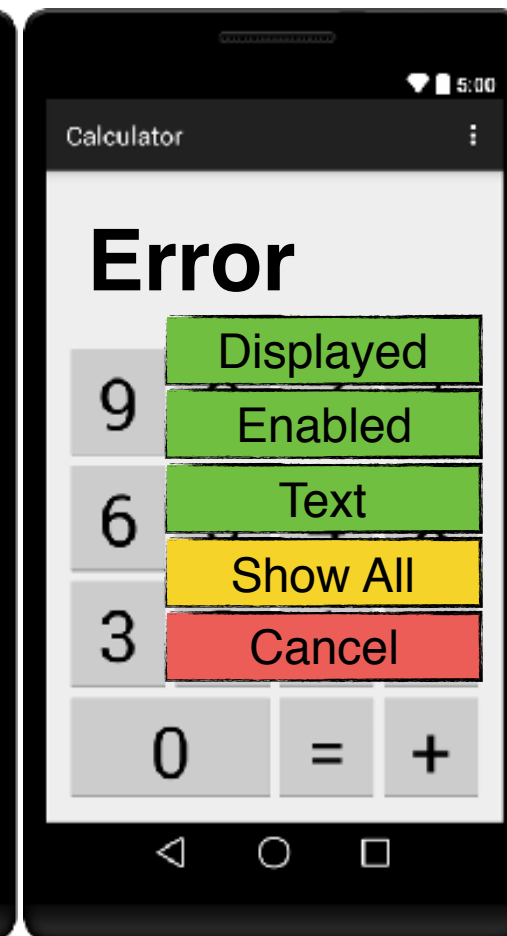
Overlay



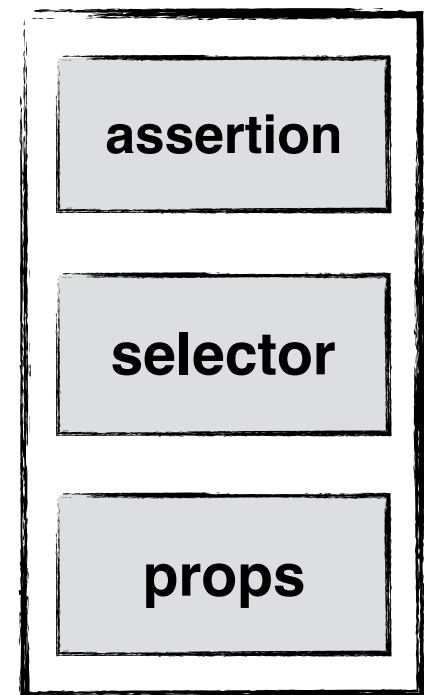
Selection



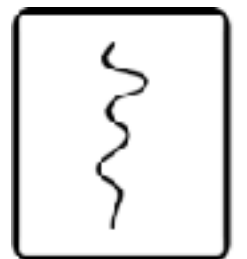
Assertion



Oracle



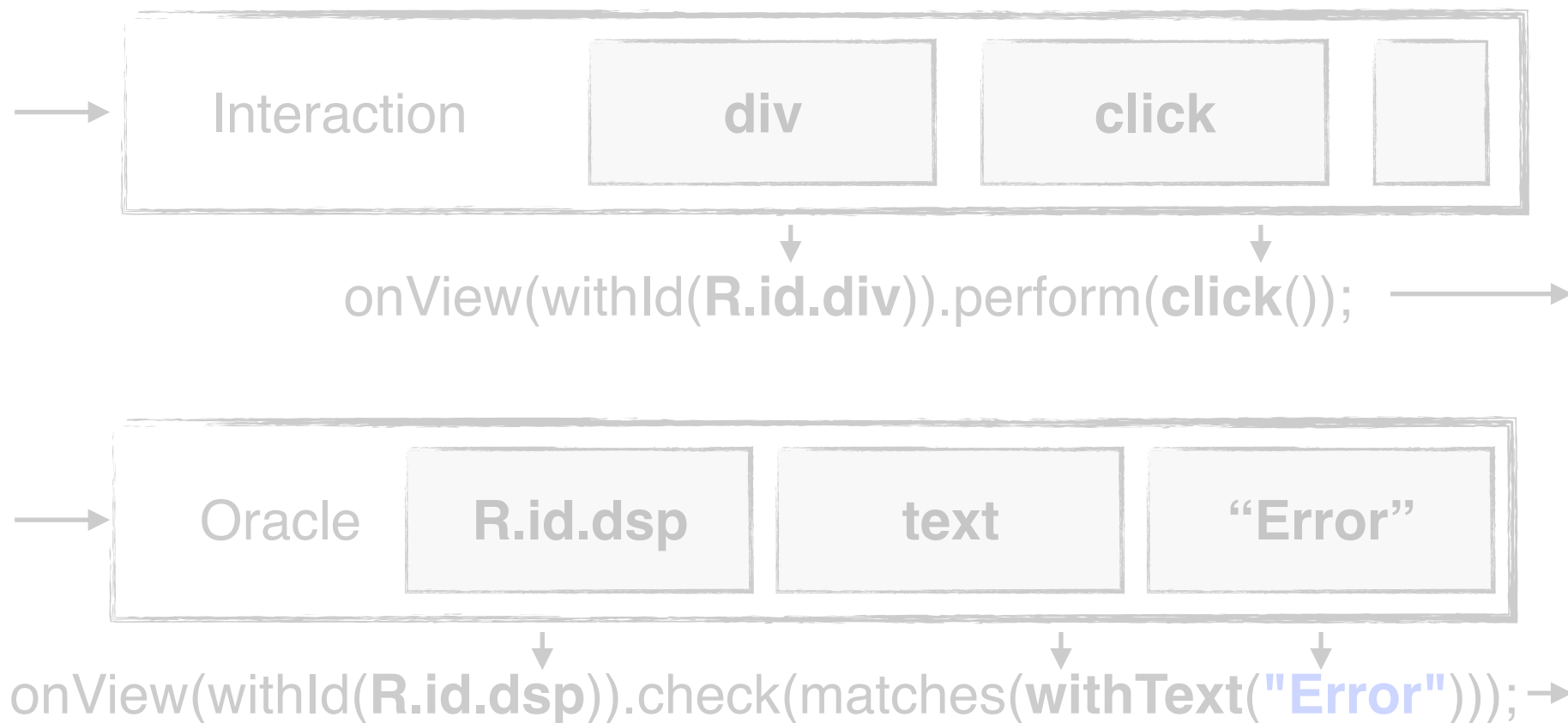
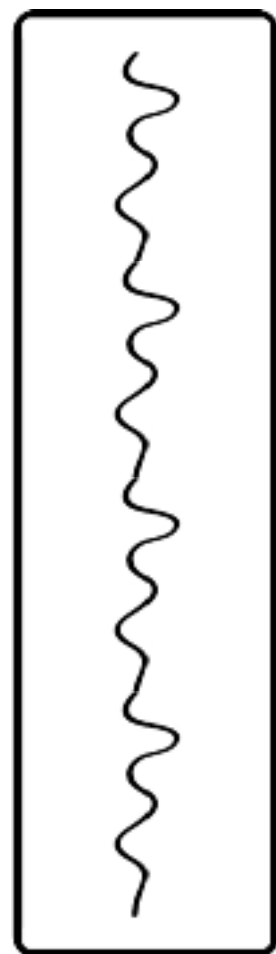
**Recorded
Trace**



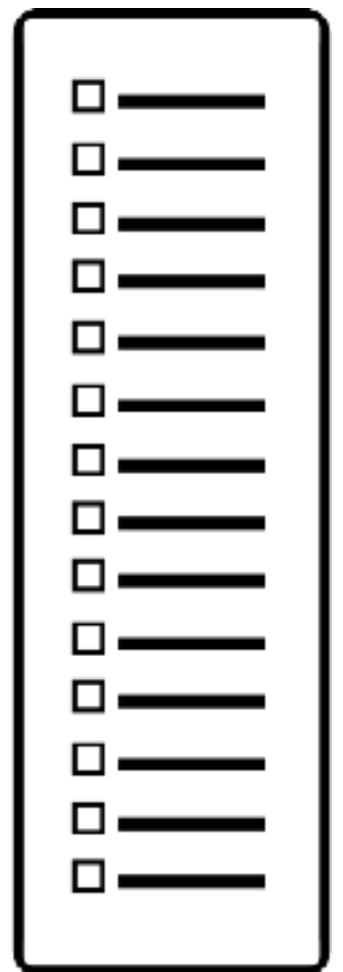
Test Case Encoding

Objective: translate recorded actions into a test case

Recorded
Trace



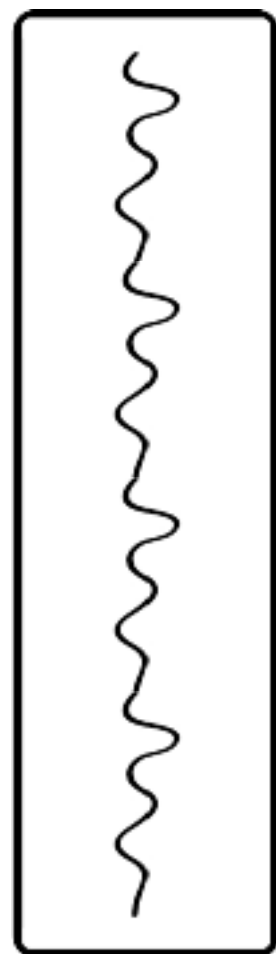
Test
Case



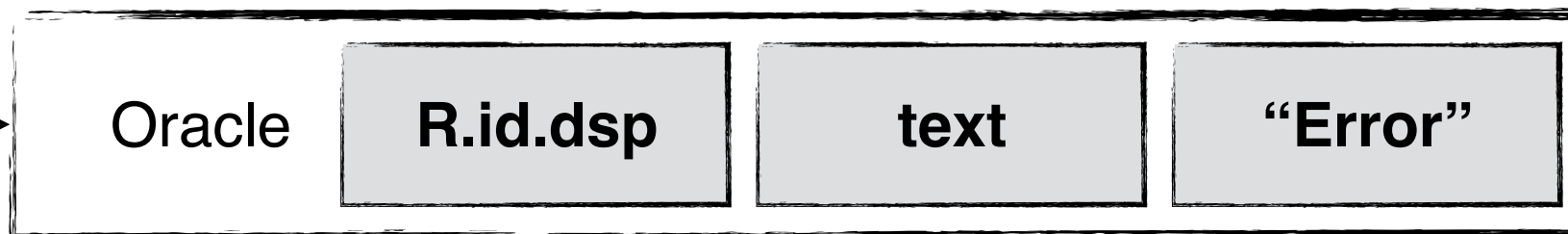
Test Case Encoding

Objective: translate recorded actions into a test case

Recorded
Trace

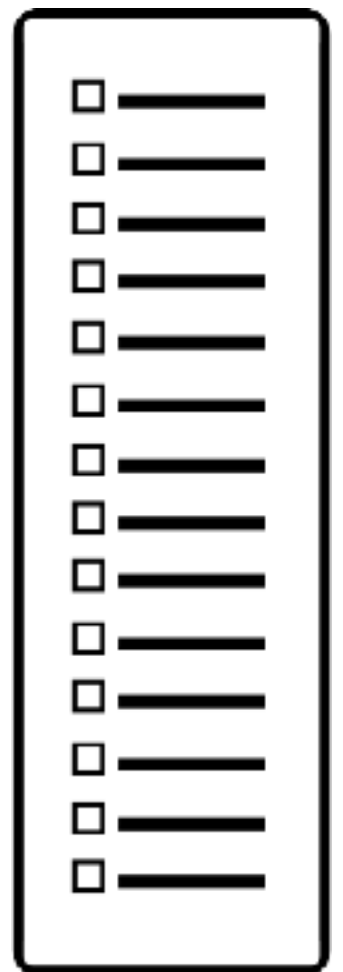


onView(withId(**R.id.div**)).perform(**click()**);



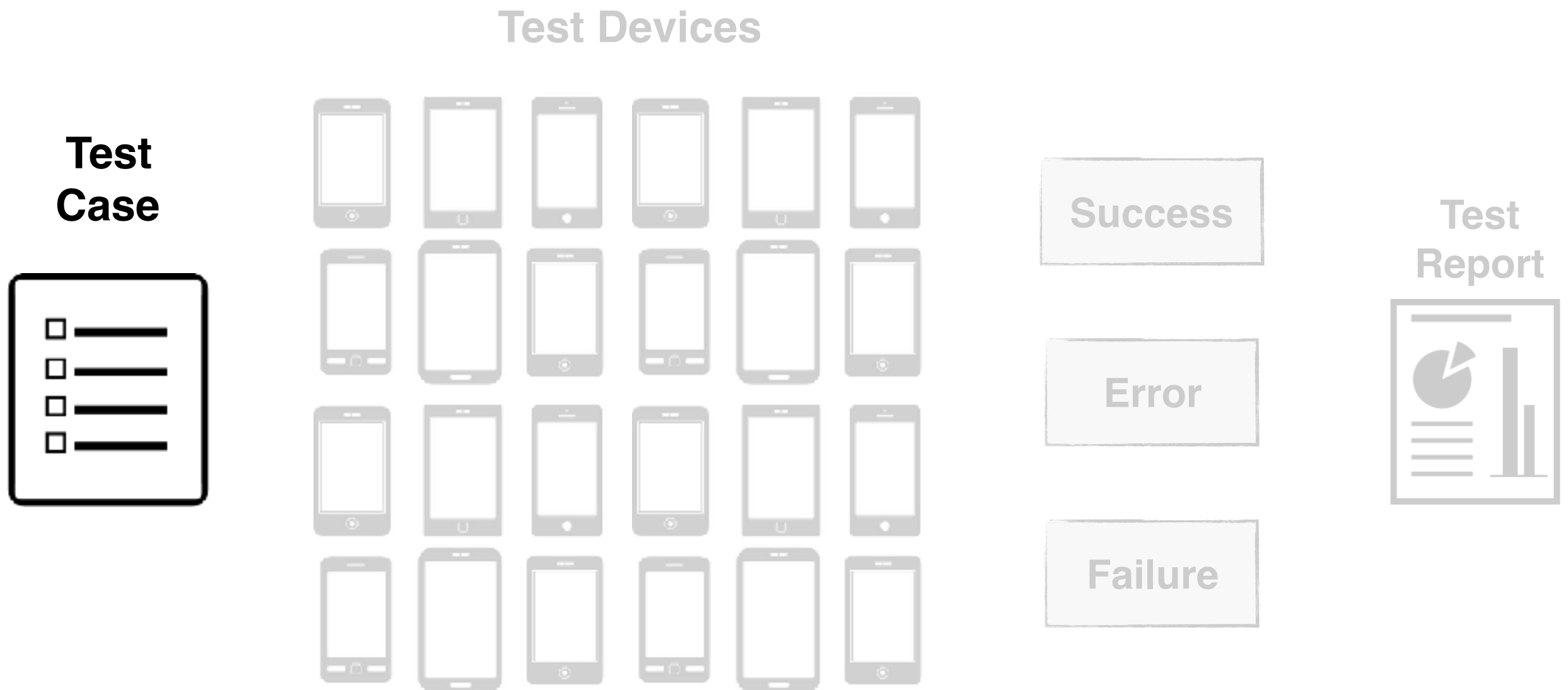
onView(withId(**R.id.dsp**)).check(matches(**withText("Error")**));

Test
Case



Test Case Execution

Objective: execute test cases and generate a report

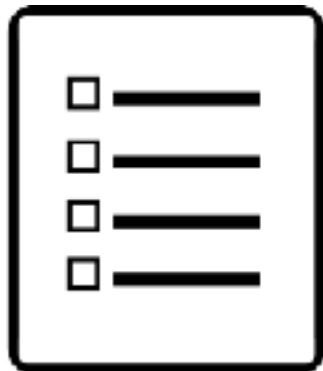


Test Case Execution

Objective: execute test cases and generate a report

Test Devices

Test Case



Success

Error

Failure

Test Report

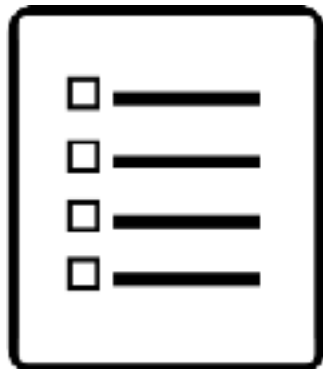


Test Case Execution

Objective: execute test cases and generate a report

Test Devices

Test Case



Success

Error

Failure

Test Report

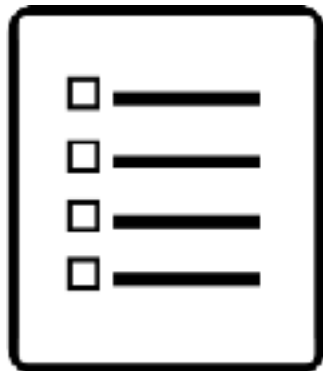


Test Case Execution

Objective: execute test cases and generate a report

Test Devices

Test Case



Success

Error

Failure

Test Report



Empirical Evaluation

Research Questions:

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

RQ2: Is the test case recording process with BARISTA more efficient than the one with TR and ETR?

RQ3: Does BARISTA's encoding preserve the functionality of the test cases? How does BARISTA compare to TR and ETR in this respect?

RQ4: Can test cases generated by BARISTA run on different devices? How does it compare to TR and ETR?

Empirical Evaluation

Research Questions:

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

RQ2: Is the test case recording process with BARISTA more efficient than the one with TR and ETR?

RQ3: Does BARISTA's encoding preserve the functionality of the test cases? How does BARISTA compare to TR and ETR in this respect?

RQ4: Can test cases generated by BARISTA run on different devices? How does it compare to TR and ETR?

Empirical Evaluation

Research Questions:

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

RQ2: Is the test case recording process with BARISTA more efficient than the one with TR and ETR?

RQ3: Does BARISTA's encoding preserve the functionality of the test cases? How does BARISTA compare to TR and ETR in this respect?

RQ4: Can test cases generated by BARISTA run on different devices? How does it compare to TR and ETR?

Empirical Evaluation

Research Questions:

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

RQ2: Is the test case recording process with BARISTA more efficient than the one with TR and ETR?

RQ3: Does BARISTA's encoding preserve the functionality of the test cases? How does BARISTA compare to TR and ETR in this respect?

RQ4: Can test cases generated by BARISTA run on different devices? How does it compare to TR and ETR?

Empirical Evaluation

Research Questions:

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

RQ2: Is the test case recording process with BARISTA more efficient than the one with TR and ETR?

RQ3: Does BARISTA's encoding preserve the functionality of the test cases? How does BARISTA compare to TR and ETR in this respect?

RQ4: Can test cases generated by BARISTA run on different devices? How does it compare to TR and ETR?

Empirical Evaluation

Research Questions:

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

RQ2: Is the test case recording process with BARISTA more efficient than the one with TR and ETR?

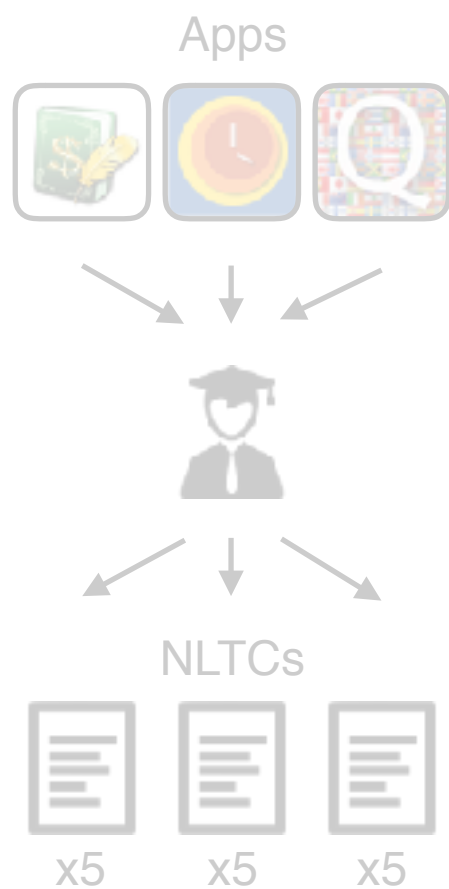
RQ3: Does BARISTA's encoding preserve the functionality of the test cases? How does BARISTA compare to TR and ETR in this respect?

RQ4: Can test cases generated by BARISTA run on different devices? How does it compare to TR and ETR?

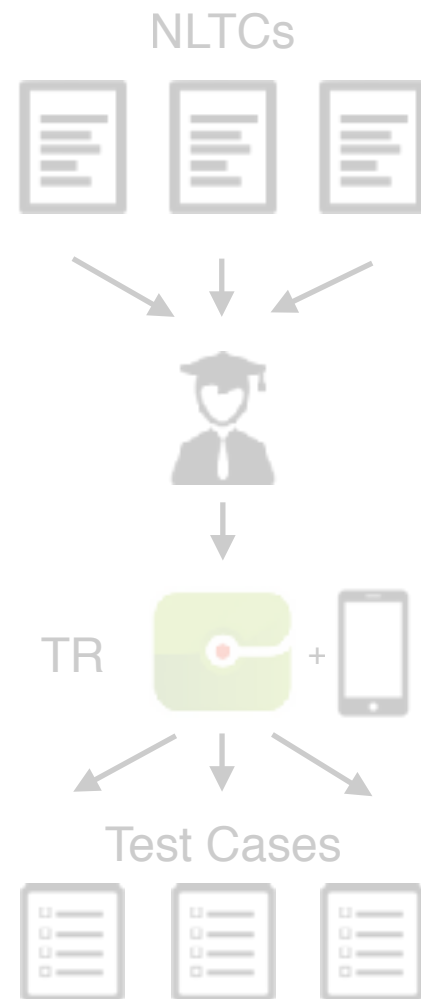
Study Protocol



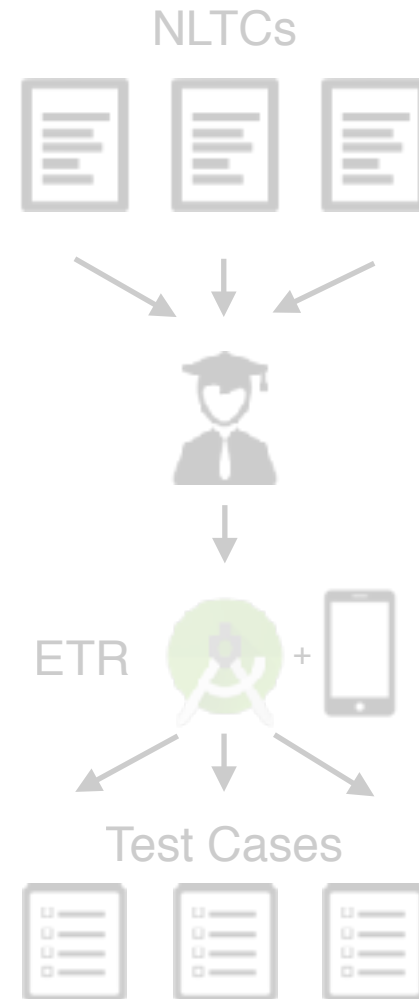
First Task



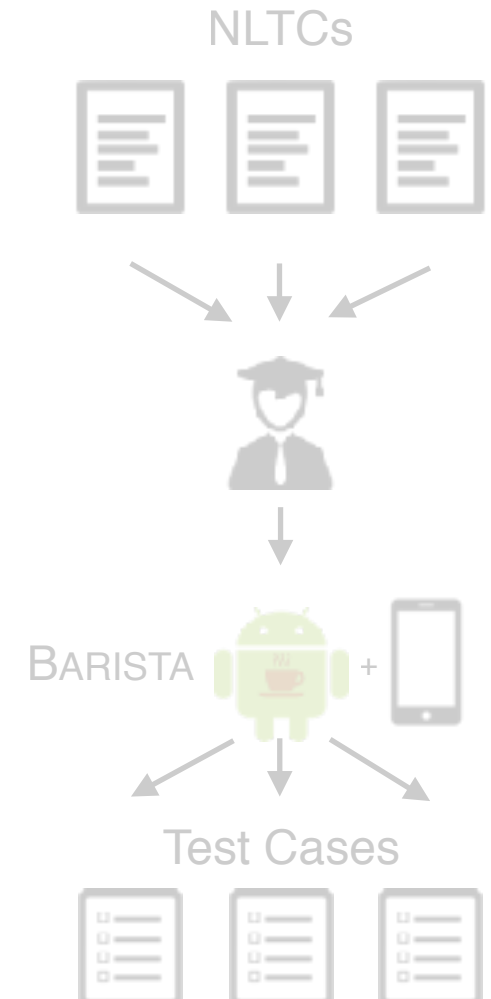
Second Task



Third Task



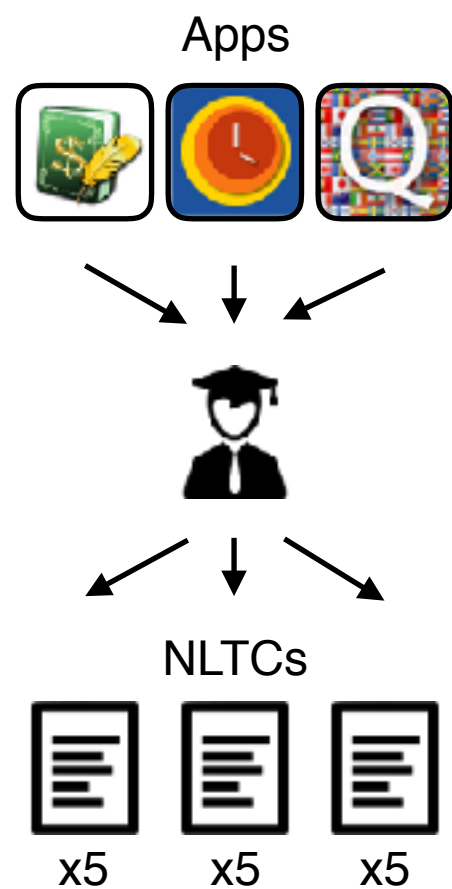
Fourth Task



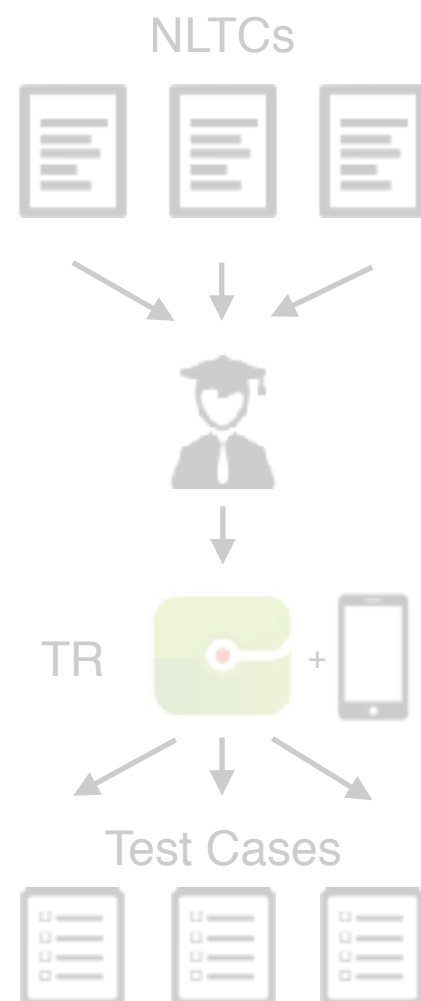
Study Protocol



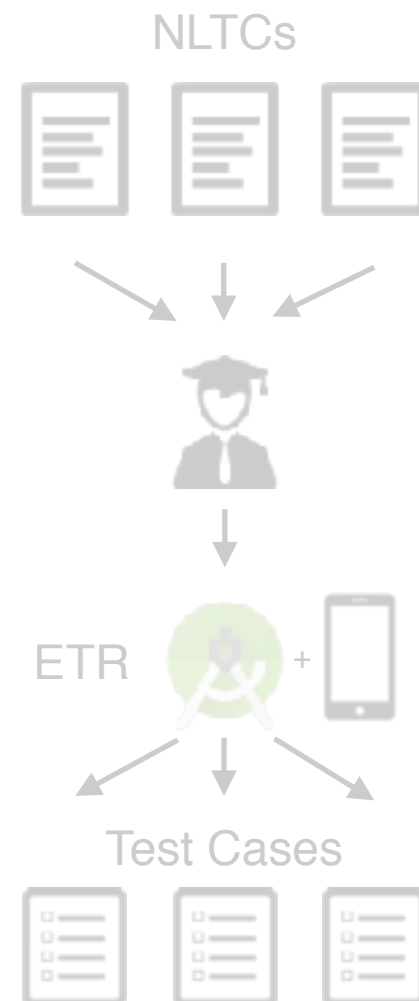
First Task



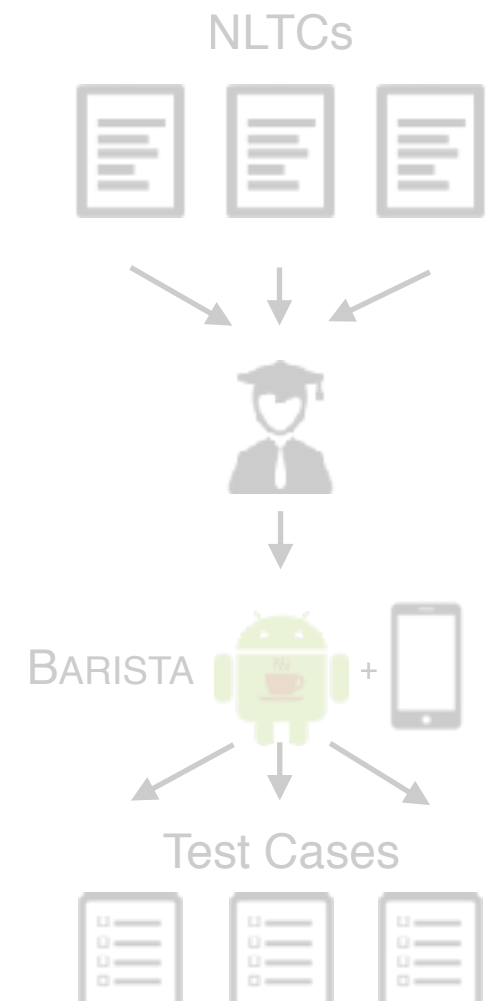
Second Task



Third Task



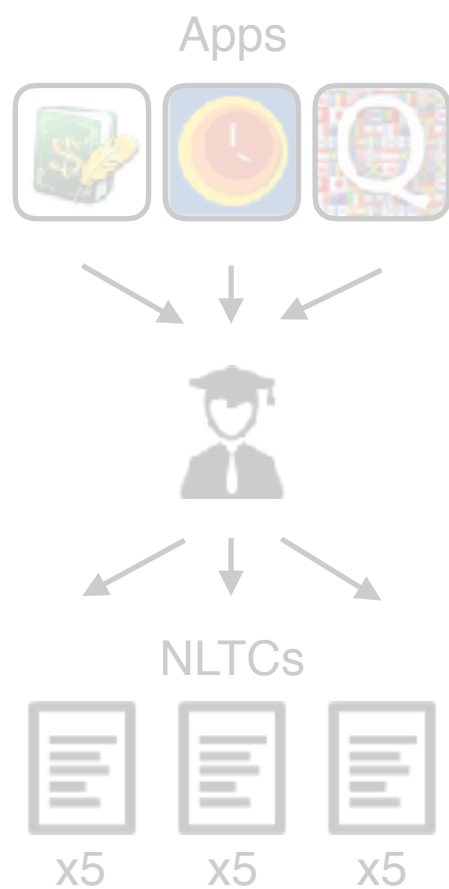
Fourth Task



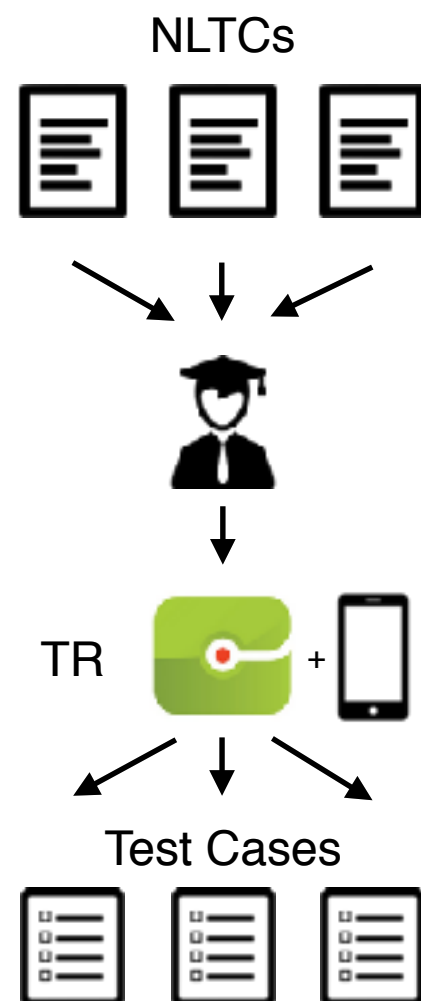
Study Protocol



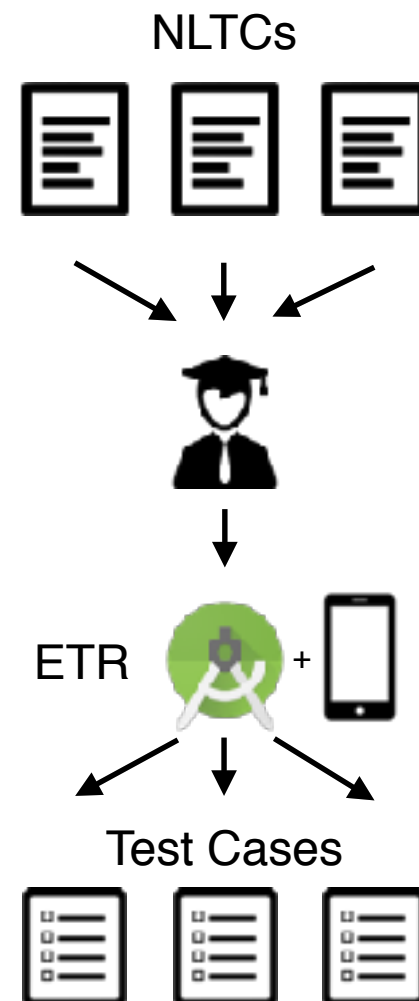
First Task



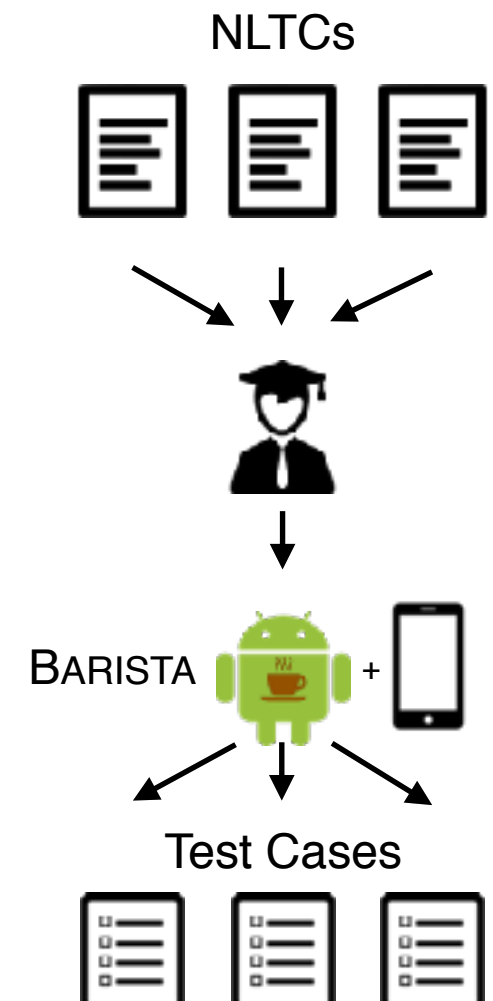
Second Task



Third Task



Fourth Task



RQ1

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

ID	TR				ETR				Barista			
	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)
A1	15	0	9	20	15	0	0	36	15	0	2	0
A2	4	11	0	2	15	0	14	3	15	0	0	0
A3	9	5	5	1	13	1	3	9	14	0	2	0
A4	9	5	8	7	14	0	26	13	14	0	3	0
A5	12	2	2	2	14	0	1	29	14	0	0	0
A6	6	7	0	1	13	0	0	13	13	0	0	0
A7	11	3	13	5	13	1	5	21	14	0	0	0
A8	11	3	5	9	14	0	0	17	14	0	0	0
A9	11	1	3	3	12	0	0	12	12	0	0	0
A10	13	2	10	2	14	1	17	8	15	0	0	0
A11	12	3	10	0	15	0	1	13	15	0	0	0
A12	15	0	5	0	15	0	4	14	15	0	0	0
A13	13	2	14	3	15	0	0	39	15	0	0	0
A14	15	0	7	2	11	4	1	14	15	0	2	0
A15	15	0	17	0	14	1	1	35	15	0	2	0
Tot	171	44	108	48	208	7	74	277	215	0	11	0

RQ1

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

ID	TR				ETR				Barista			
	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)
A1	15	0	9	20	15	0	0	36	15	0	2	0
A2	4	11	0	2	15	0	14	3	15	0	0	0
A3	9	5	5	1	13	1	3	9	14	0	2	0
A4	9	5	8	7	14	0	26	13	14	0	3	0
A5	12	2	2	2	14	0	1	29	14	0	0	0
A6	6	7	0	1	13	0	0	13	13	0	0	0
A7	11	3	13	5	13	1	5	21	14	0	0	0
A8	11	3	5	9	14	0	0	17	14	0	0	0
A9	11	1	3	3	12	0	0	12	12	0	0	0
A10	13	2	10	2	14	1	17	8	15	0	0	0
A11	12	3	10	0	15	0	1	13	15	0	0	0
A12	15	0	5	0	15	0	4	14	15	0	0	0
A13	13	2	14	3	15	0	0	39	15	0	0	0
A14	15	0	7	2	11	4	1	14	15	0	2	0
A15	15	0	17	0	14	1	1	35	15	0	2	0
Tot	171	44	108	48	208	7	74	277	215	0	11	0

RQ1

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

ID	TR				ETR				Barista			
	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)
A1	15	0	9	20	15	0	0	36	15	0	2	0
A2	4	11	0	2	15	0	14	3	15	0	0	0
A3	9	5	5	1	13	1	3	9	14	0	2	0
A4	9	5	8	7	14	0	26	13	14	0	3	0
A5	12	2	2	2	14	0	1	29	14	0	0	0
A6	6	7	0	1	13	0	0	13	13	0	0	0
A7	11	3	13	5	13	1	5	21	14	0	0	0
A8	11	3	5	9	14	0	0	17	14	0	0	0
A9	11	1	3	3	12	0	0	12	12	0	0	0
A10	13	2	10	2	14	1	17	8	15	0	0	0
A11	12	3	10	0	15	0	1	13	15	0	0	0
A12	15	0	5	0	15	0	4	14	15	0	0	0
A13	13	2	14	3	15	0	0	39	15	0	0	0
A14	15	0	7	2	11	4	1	14	15	0	2	0
A15	15	0	17	0	14	1	1	35	15	0	2	0
Tot	171	44	108	48	208	7	74	277	215	0	11	0

RQ1

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

ID	TR				ETR				Barista			
	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)
A1	15	0	9	20	15	0	0	36	15	0	2	0
A2	4	11	0	2	15	0	14	3	15	0	0	0
A3	9	5	5	1	13	1	3	9	14	0	2	0
A4	9	5	8	7	14	0	26	13	14	0	3	0
A5	12	2	2	2	14	0	1	29	14	0	0	0
A6	6	7	0	1	13	0	0	13	13	0	0	0
A7	11	3	13	5	13	1	5	21	14	0	0	0
A8	11	3	5	9	14	0	0	17	14	0	0	0
A9	11	1	3	3	12	0	0	12	12	0	0	0
A10	13	2	10	2	14	1	17	8	15	0	0	0
A11	12	3	10	0	15	0	1	13	15	0	0	0
A12	15	0	5	0	15	0	4	14	15	0	0	0
A13	13	2	14	3	15	0	0	39	15	0	0	0
A14	15	0	7	2	11	4	1	14	15	0	2	0
A15	15	0	17	0	14	1	1	35	15	0	2	0
Tot	171	44	108	48	208	7	74	277	215	0	11	0

RQ1

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

ID	TR				ETR				Barista			
	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)
A1	15	0	9	20	15	0	0	36	15	0	2	0
A2	4	11	0	2	15	0	14	3	15	0	0	0
A3	9	5	5	1	13	1	3	9	14	0	2	0
A4	9	5	8	7	14	0	26	13	14	0	3	0
A5	12	2	2	2	14	0	1	29	14	0	0	0
A6	6	7	0	1	13	0	0	13	13	0	0	0
A7	11	3	13	5	13	1	5	21	14	0	0	0
A8	11	3	5	9	14	0	0	17	14	0	0	0
A9	11	1	3	3	12	0	0	12	12	0	0	0
A10	13	2	10	2	14	1	17	8	15	0	0	0
A11	12	3	10	0	15	0	1	13	15	0	0	0
A12	15	0	5	0	15	0	4	14	15	0	0	0
A13	13	2	14	3	15	0	0	39	15	0	0	0
A14	15	0	7	2	11	4	1	14	15	0	2	0
A15	15	0	17	0	14	1	1	35	15	0	2	0
Tot	171	44	108	48	208	7	74	277	215	0	11	0

RQ1

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

<i>ID</i>	<i>TR</i>				<i>ETR</i>				<i>Barista</i>			
	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>

100% completed tests for BARISTA

2% assertions skipped for BARISTA and none altered

More completed tests with BARISTA

More assertions skipped or altered with TR and ETR

<i>AT5</i>	15	0	17	0	14	1	1	33	15	0	2	0
<i>Tot</i>	171	44	108	48	208	7	74	277	215	0	11	0

RQ1

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

<i>ID</i>	<i>TR</i>				<i>ETR</i>				<i>Barista</i>			
	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>

100% completed tests for BARISTA

2% assertions skipped for BARISTA and none altered

More completed tests with BARISTA

More assertions skipped or altered with TR and ETR

<i>AT5</i>	15	0	17	0	14	1	1	33	15	0	2	0
Tot	171	44	108	48	208	7	74	277	215	0	11	0

RQ1

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

ID	TR				ETR				Barista			
	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)	C(#)	NC(#)	AS(#)	AA(#)

100% completed tests for BARISTA

2% assertions skipped for BARISTA and none altered

More completed tests with BARISTA

More assertions skipped or altered with TR and ETR

AT5	15	0	17	0	14	1	1	33	15	0	2	0
Tot	171	44	108	48	208	7	74	277	215	0	11	0

RQ1

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

<i>ID</i>	<i>TR</i>				<i>ETR</i>				<i>Barista</i>			
	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>

100% completed tests for BARISTA

2% assertions skipped for BARISTA and none altered

More completed tests with BARISTA

More assertions skipped or altered with TR and ETR

<i>AT5</i>	15	0	17	0	14	1	1	33	15	0	2	0
Tot	171	44	108	48	208	7	74	277	215	0	11	0

RQ1

RQ1: Can BARISTA record user defined test cases? If so, how does it compare to Testdroid Recorder (TR) and Espresso Test Recorder (ETR)?

<i>ID</i>	<i>TR</i>				<i>ETR</i>				<i>Barista</i>			
	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>	<i>C(#)</i>	<i>NC(#)</i>	<i>AS(#)</i>	<i>AA(#)</i>

100% completed tests for BARISTA

2% assertions skipped for BARISTA and none altered

More completed tests with BARISTA

More assertions skipped or altered with TR and ETR

<i>AT5</i>	15	0	17	0	14	1	1	33	15	0	2	0
Tot	171	44	108	48	208	7	74	277	215	0	11	0

RQ4

RQ4: Can test cases generated by BARISTA run on different devices? How does it compare to TR and ETR?



RQ4

RQ4: Can test cases generated by BARISTA run on different devices? How does it compare to TR and ETR?

LG
G Flex



Motorola
Moto X



HTC
One M8



Sony
Xperia Z3



Samsung
Galaxy S5



LG
Nexus 5



LG
G3



TR



68.3%

ETR



37.3%

Barista



99.2%

RQ4

RQ4: Can test cases generated by BARISTA run on different devices? How does it compare to TR and ETR?

LG
G Flex



Motorola
Moto X



HTC
One M8



Sony
Xperia Z3



Samsung
Galaxy S5



LG
Nexus 5



LG
G3



TR



68.3%

ETR



37.3%

Barista



99.2%

Evaluation Summary

RQ1: BARISTA can record test cases and is more expressive than TR and ETR

RQ2: BARISTA is more efficient in recording test cases than TR and ETR

RQ3: BARISTA generates test cases that work correctly and it outperforms TR and ETR

RQ4: BARISTA generates test cases that can run on different devices and it generates a greater number of cross-device-compatible tests than TR and ETR

Feedback #1

“I have been looking for something like BARISTA to help me get into automation for a while”

Feedback #2

“Overall, a very interesting tool! This could save us quite some time by generating some of the tests for us”

Feedback #3

“There are a few more assertions I’d like to see. For example, testing the number of items in a ListView”

Evaluation Summary

RQ1: BARISTA can record test cases and is more expressive than TR and ETR

RQ2: BARISTA is more efficient in recording test cases than TR and ETR

RQ3: BARISTA generates test cases that work correctly and it outperforms TR and ETR

RQ4: BARISTA generates test cases that can run on different devices and it generates a greater number of cross-device-compatible tests than TR and ETR

Feedback #1

"I have been looking for something like BARISTA to help me get into automation for a while"

Feedback #2

"Overall, a very interesting tool! This could save us quite some time by generating some of the tests for us"

Feedback #3

"There are a few more assertions I'd like to see. For example, testing the number of items in a ListView"

Evaluation Summary

RQ1: BARISTA can record test cases and is more expressive than TR and ETR

RQ2: BARISTA is more efficient in recording test cases than TR and ETR

RQ3: BARISTA generates test cases that work correctly and it outperforms TR and ETR

RQ4: BARISTA generates test cases that can run on different devices and it generates a greater number of cross-device-compatible tests than TR and ETR

Feedback #1

“I have been looking for something like BARISTA to help me get into automation for a while”

Feedback #2

“Overall, a very interesting tool! This could save us quite some time by generating some of the tests for us”

Feedback #3

“There are a few more assertions I’d like to see. For example, testing the number of items in a ListView”

Evaluation Summary

RQ1: BARISTA can record test cases and is more expressive than TR and ETR

RQ2: BARISTA is more efficient in recording test cases than TR and ETR

RQ3: BARISTA generates test cases that work correctly and it outperforms TR and ETR

RQ4: BARISTA generates test cases that can run on different devices and it generates a greater number of cross-device-compatible tests than TR and ETR

Feedback #1

"I have been looking for something like BARISTA to help me get into automation for a while"

Feedback #2

"Overall, a very interesting tool! This could save us quite some time by generating some of the tests for us"

Feedback #3

"There are a few more assertions I'd like to see. For example, testing the number of items in a ListView"

Evaluation Summary

RQ1: BARISTA can record test cases and is more expressive than TR and ETR

RQ2: BARISTA is more efficient in recording test cases than TR and ETR

RQ3: BARISTA generates test cases that work correctly and it outperforms TR and ETR

RQ4: BARISTA generates test cases that can run on different devices and it generates a greater number of cross-device-compatible tests than TR and ETR

Feedback #1

"I have been looking for something like BARISTA to help me get into automation for a while"

Feedback #2

"Overall, a very interesting tool! This could save us quite some time by generating some of the tests for us"

Feedback #3

"There are a few more assertions I'd like to see. For example, testing the number of items in a ListView"

Evaluation Summary

RQ1: BARISTA can record test cases and is more expressive than TR and ETR

RQ2: BARISTA is more efficient in recording test cases than TR and ETR

RQ3: BARISTA generates test cases that work correctly and it outperforms TR and ETR

RQ4: BARISTA generates test cases that can run on different devices and it generates a greater number of cross-device-compatible tests than TR and ETR

Feedback #1

"I have been looking for something like BARISTA to help me get into automation for a while"

Feedback #2

"Overall, a very interesting tool! This could save us quite some time by generating some of the tests for us"

Feedback #3

"There are a few more assertions I'd like to see. For example, testing the number of items in a ListView"

Evaluation Summary

RQ1: BARISTA can record test cases and is more expressive than TR and ETR

RQ2: BARISTA is more efficient in recording test cases than TR and ETR

RQ3: BARISTA generates test cases that work correctly and it outperforms TR and ETR

RQ4: BARISTA generates test cases that can run on different devices and it generates a greater number of cross-device-compatible tests than TR and ETR

Feedback #1

"I have been looking for something like BARISTA to help me get into automation for a while"

Feedback #2

"Overall, a very interesting tool! This could save us quite some time by generating some of the tests for us"

Feedback #3

"There are a few more assertions I'd like to see. For example, testing the number of items in a ListView"

Future Work

Additional user study

Reuse repetitive action sequences

Sandboxing

Extend assertable properties

Fuzzing

Fix broken test case during evolution

Failure diagnosis

Future Work

Additional user study

Reuse repetitive action sequences

Sandboxing

Extend assertable properties

Fuzzing

Fix broken test case during evolution

Failure diagnosis

Future Work

Additional user study

Reuse repetitive action sequences

Sandboxing

Extend assertable properties

Fuzzing

Fix broken test case during evolution

Failure diagnosis

Future Work

Additional user study

Reuse repetitive action sequences

Sandboxing

Extend assertable properties

Fuzzing

Fix broken test case during evolution

Failure diagnosis

Future Work

Additional user study

Reuse repetitive action sequences

Sandboxing

Extend assertable properties

Fuzzing

Fix broken test case during evolution

Failure diagnosis

Future Work

Additional user study

Reuse repetitive action sequences

Sandboxing

Extend assertable properties

Fuzzing

Fix broken test case during evolution

Failure diagnosis

Future Work

Additional user study

Reuse repetitive action sequences

Sandboxing

Extend assertable properties

Fuzzing

Fix broken test case during evolution

Failure diagnosis

Future Work

Additional user study

Reuse repetitive action sequences

Sandboxing

Extend assertable properties

Fuzzing

Fix broken test case during evolution

Failure diagnosis

Research

Test
Generation

Barista

Differential
Testing

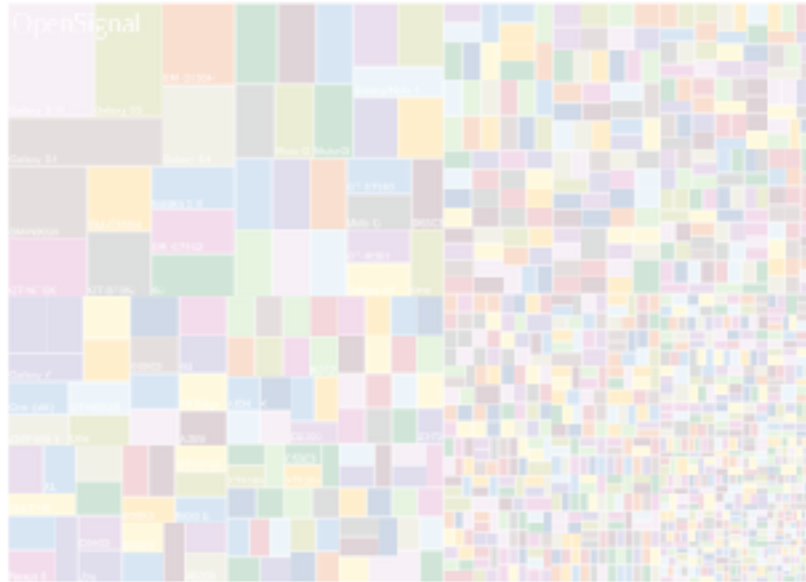
DiffDroid

Bug Report
Analysis

Yakusu

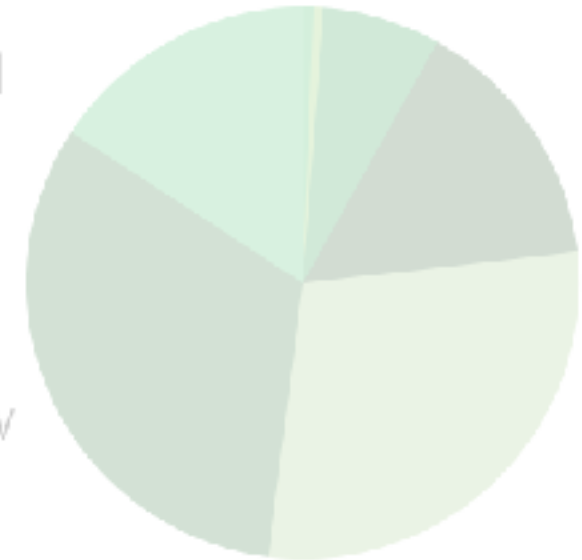
Fragmentation

Device

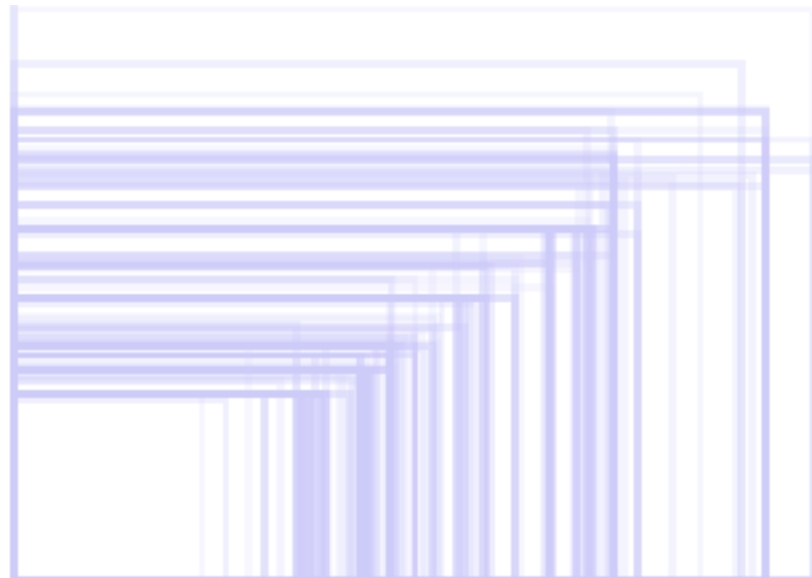


System

- Gingerbread
- Ice Cream
- Jelly Bean
- KitKat
- Lollipop
- Marshmallow
- Nougat



Screen



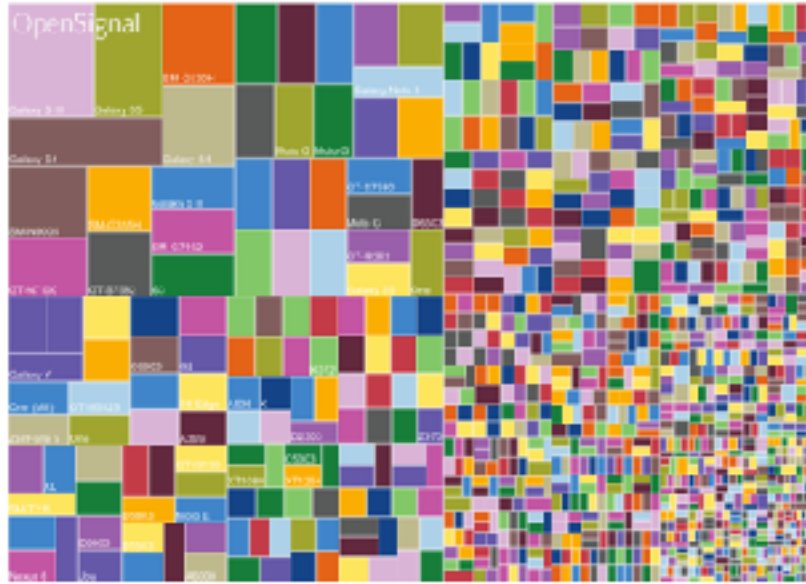
News

Jul 2016

“Due to the wide array of available Android devices, **we are targeting our support to a select number of Android devices** to continue improving our overall Salesforce1 for Android user experience”

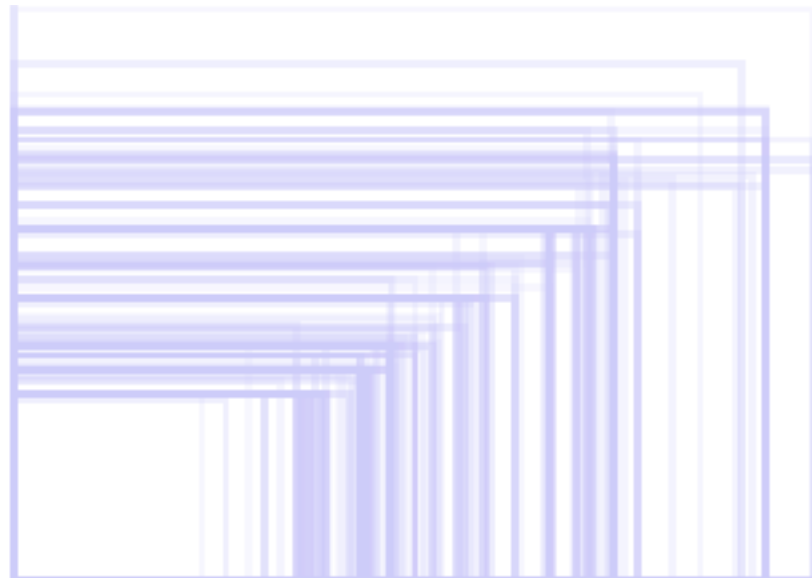
Fragmentation

Device



System

Screen



News

Jul 2016

“Due to the wide array of available Android devices, **we are targeting our support to a select number of Android devices** to continue improving our overall Salesforce1 for Android user experience”

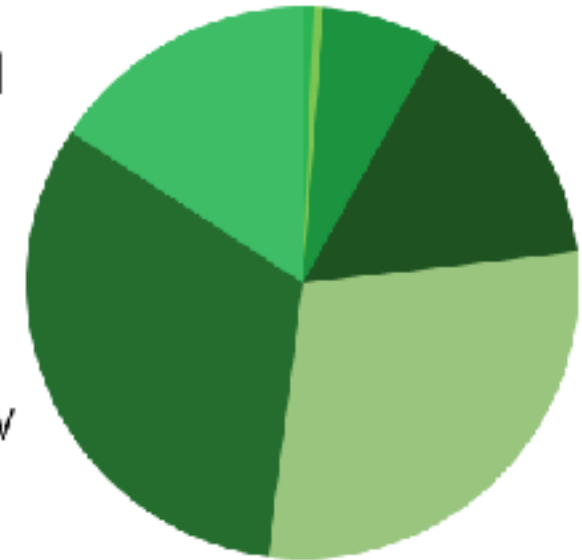
Fragmentation

Device

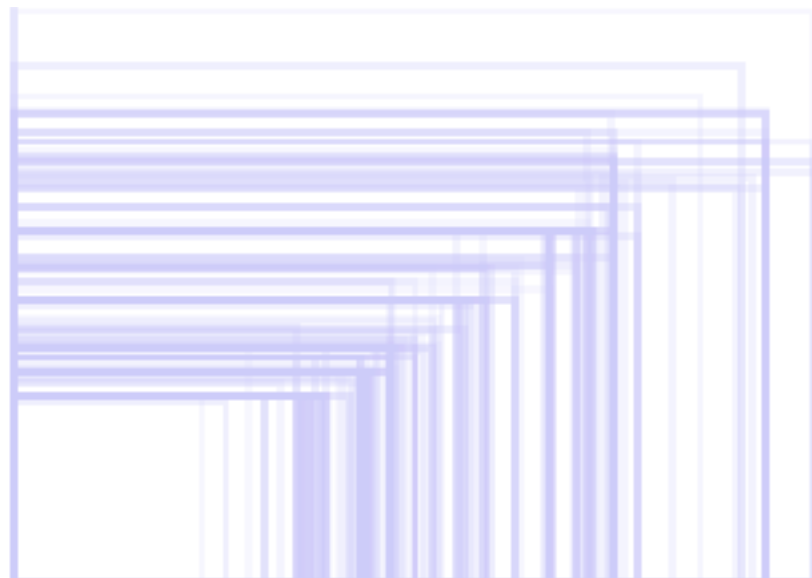


System

- Gingerbread
- Ice Cream
- Jelly Bean
- KitKat
- Lollipop
- Marshmallow
- Nougat



Screen



News

Jul 2016

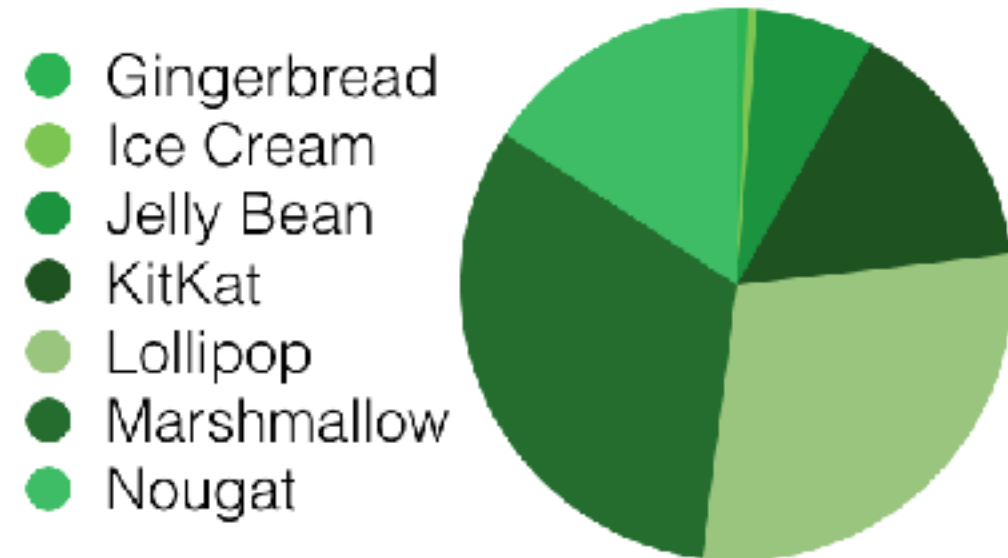
“Due to the wide array of available Android devices, **we are targeting our support to a select number of Android devices** to continue improving our overall Salesforce1 for Android user experience”

Fragmentation

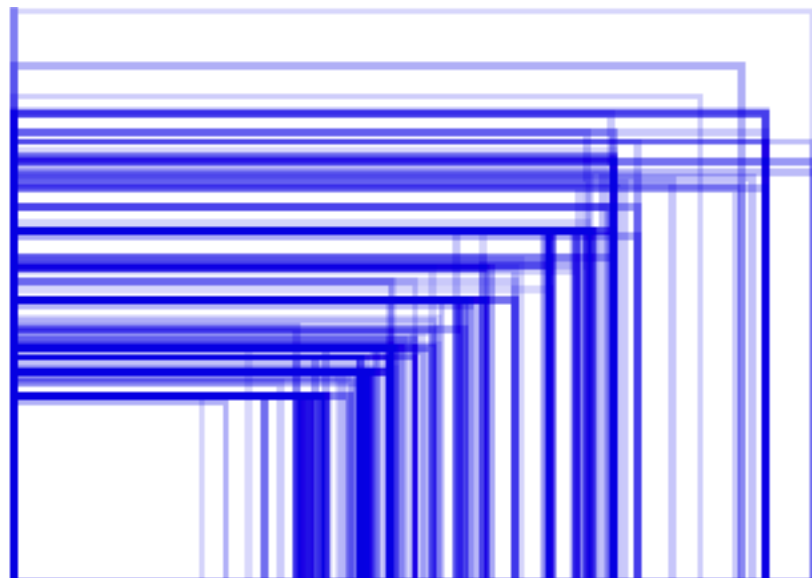
Device



System



Screen



News

Jul 2016

“Due to the wide array of available Android devices, **we are targeting our support to a select number of Android devices** to continue improving our overall Salesforce1 for Android user experience”

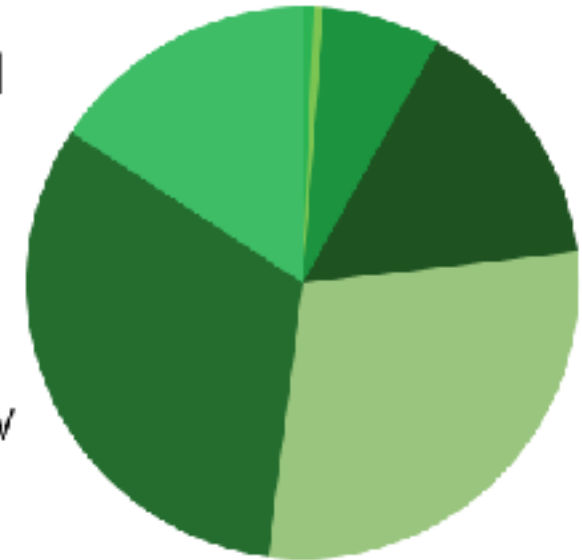
Fragmentation

Device

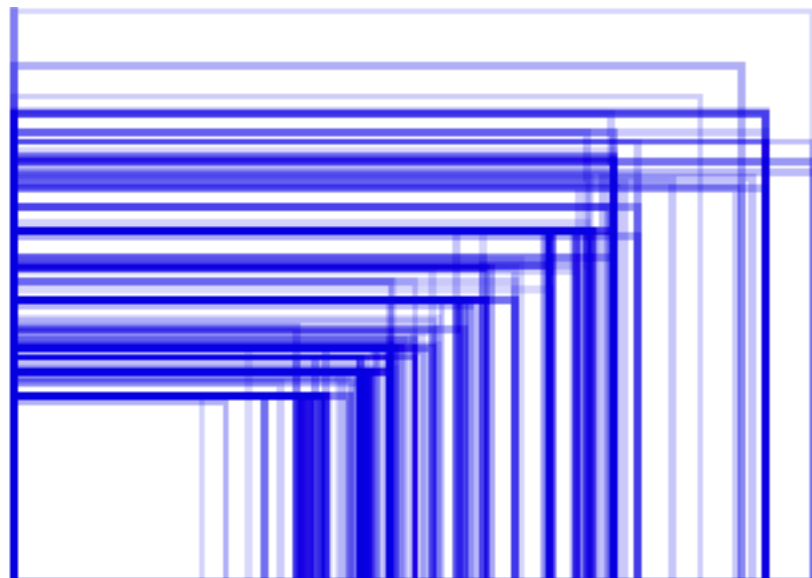


System

- Gingerbread
- Ice Cream
- Jelly Bean
- KitKat
- Lollipop
- Marshmallow
- Nougat



Screen



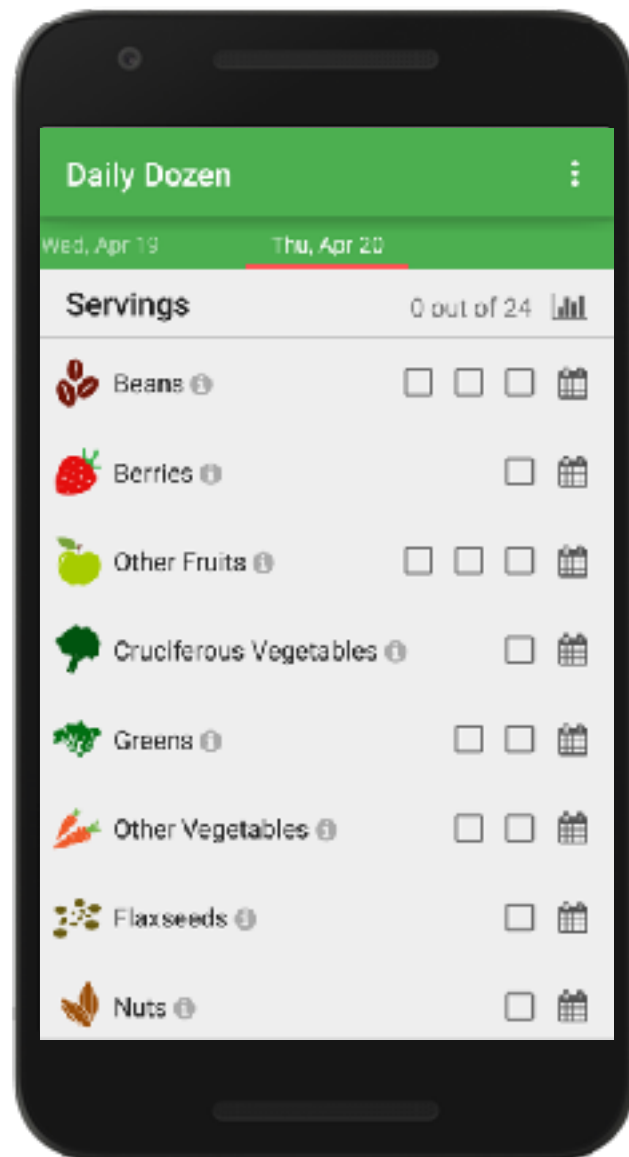
News

Jul 2016

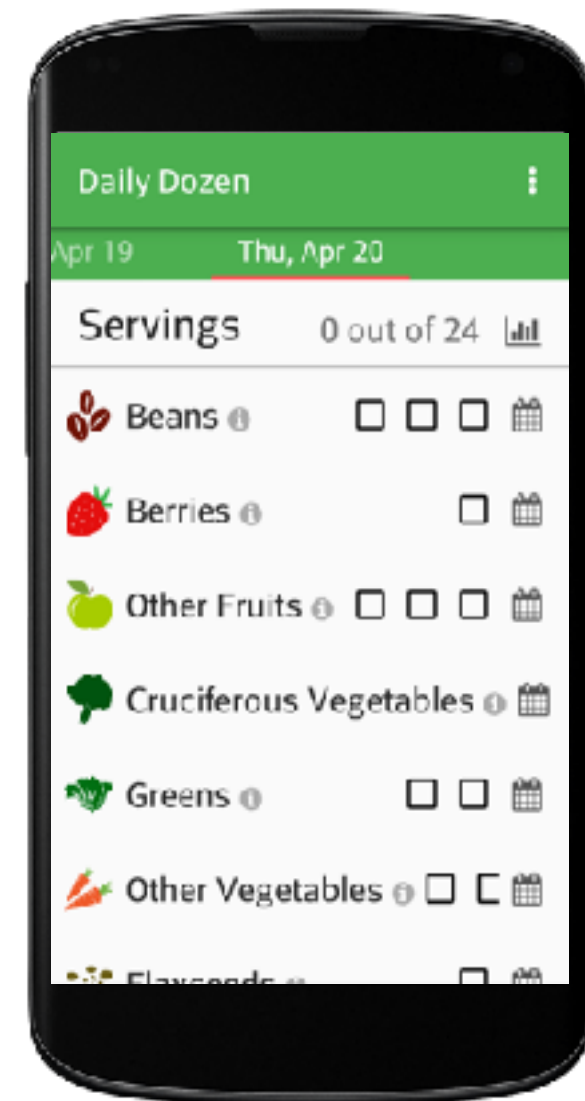
“Due to the wide array of available Android devices, **we are targeting our support to a select number of Android devices** to continue improving our overall Salesforce1 for Android user experience”

Motivating Example

LG G3

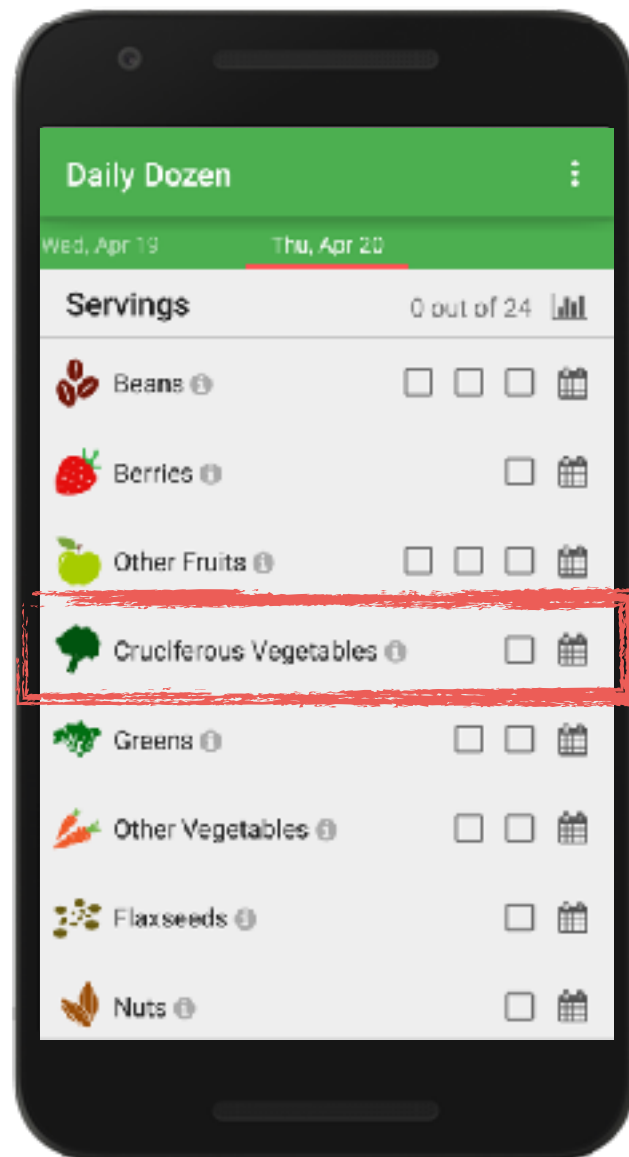


LG Optimus L70

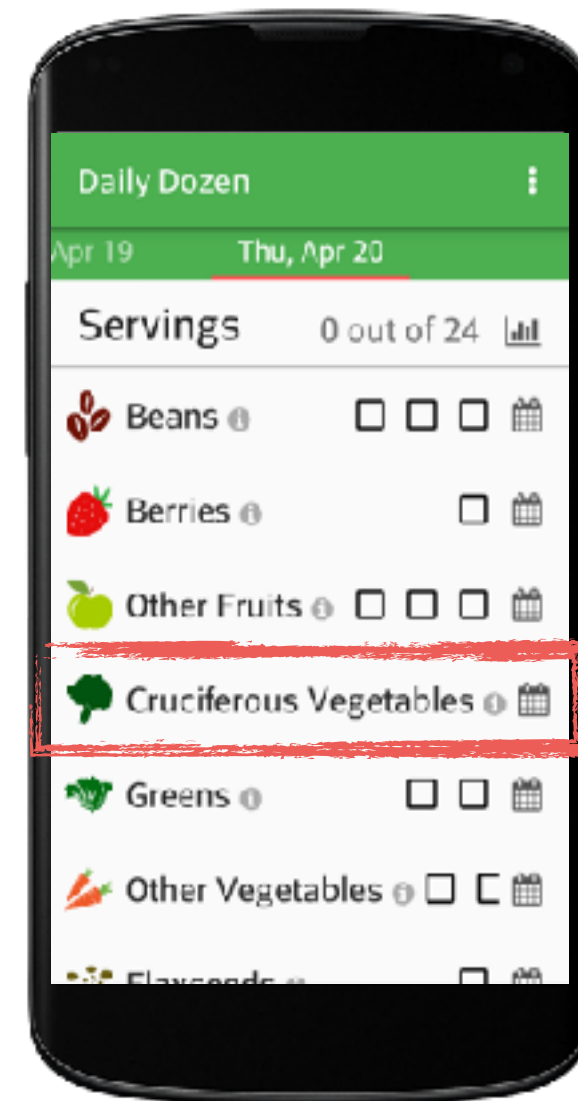


Motivating Example

LG G3

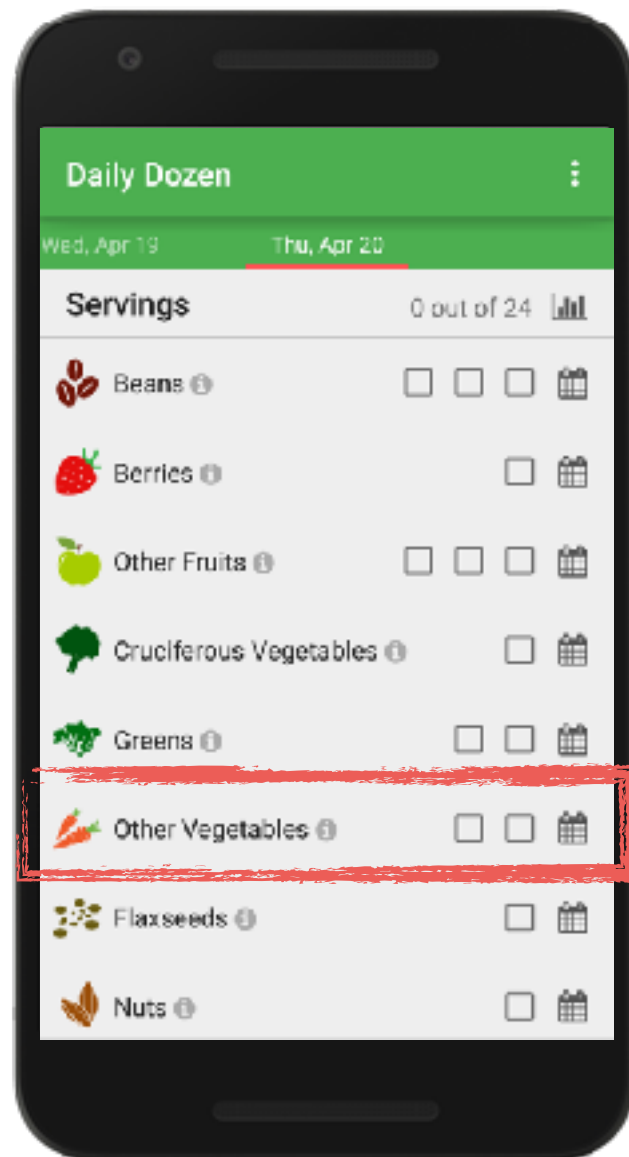


LG Optimus L70

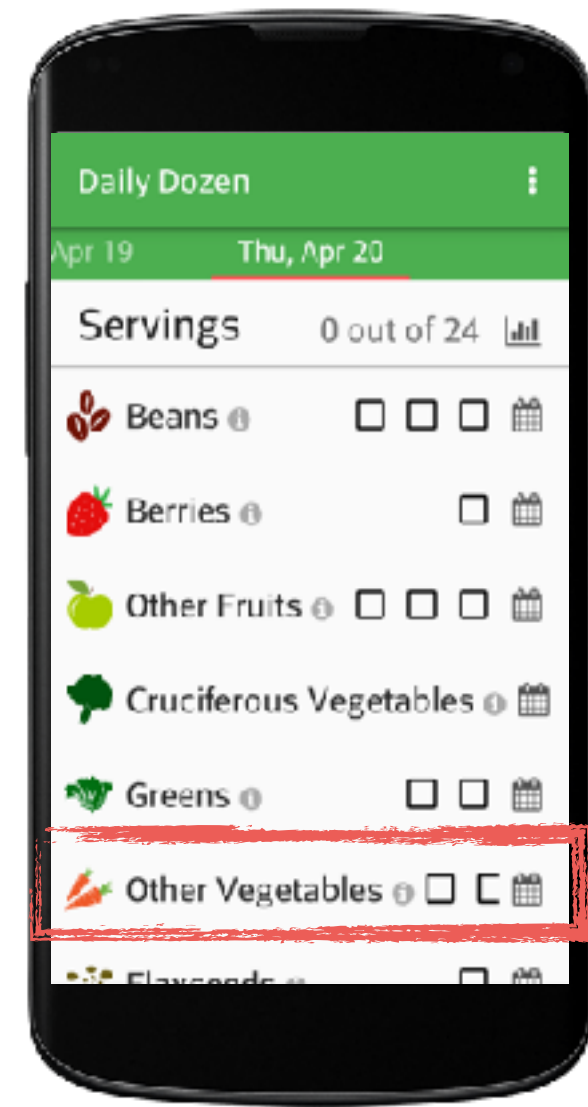


Motivating Example

LG G3

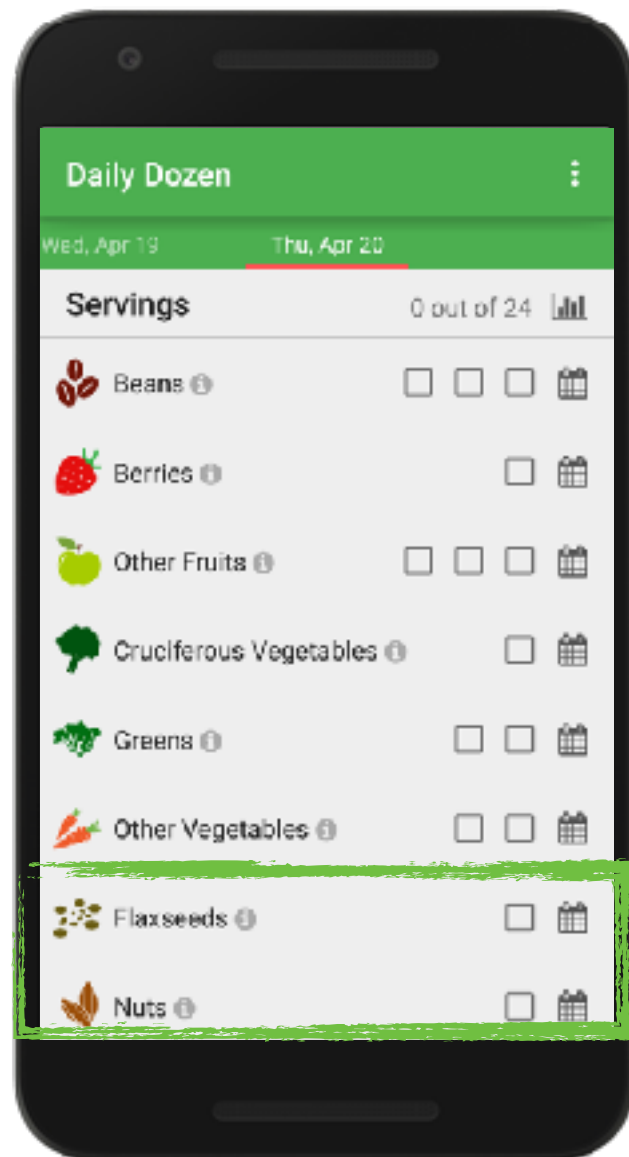


LG Optimus L70

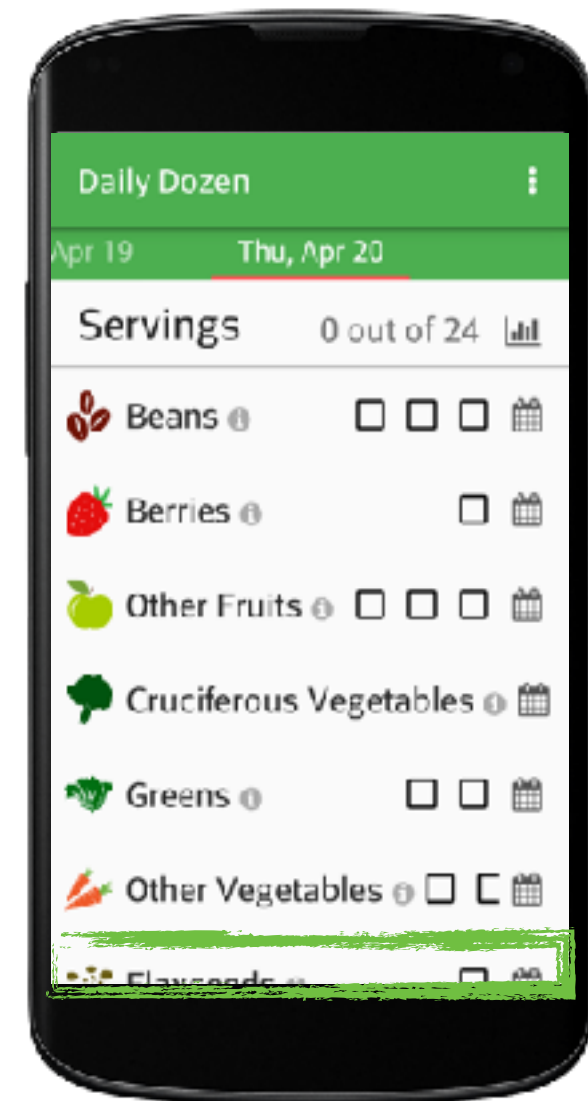


Motivating Example

LG G3

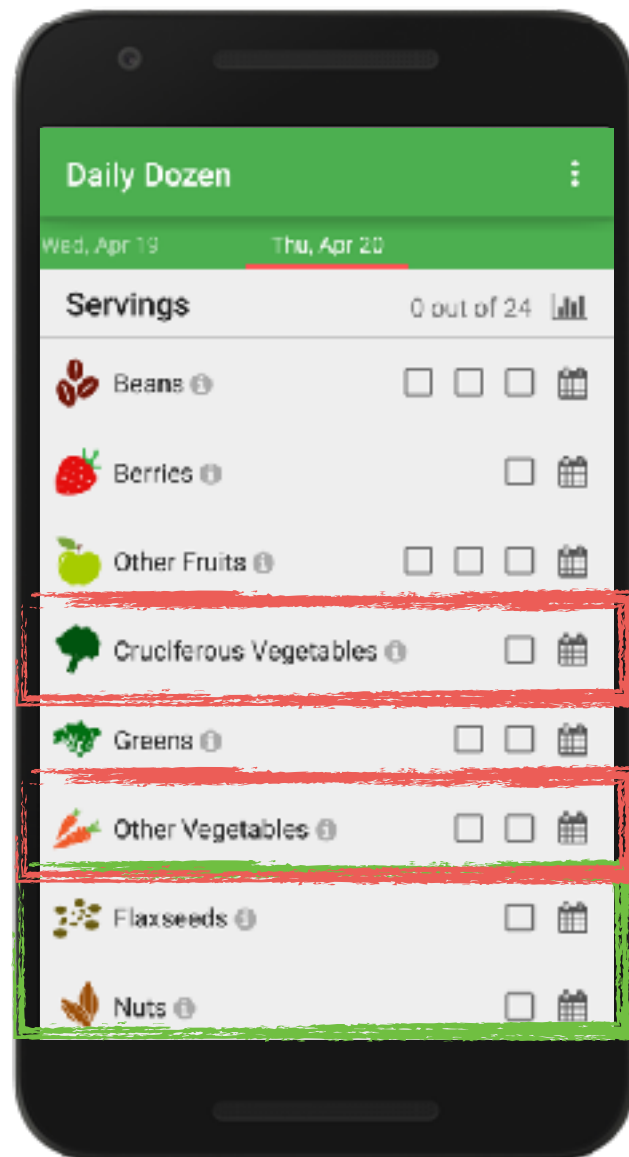


LG Optimus L70

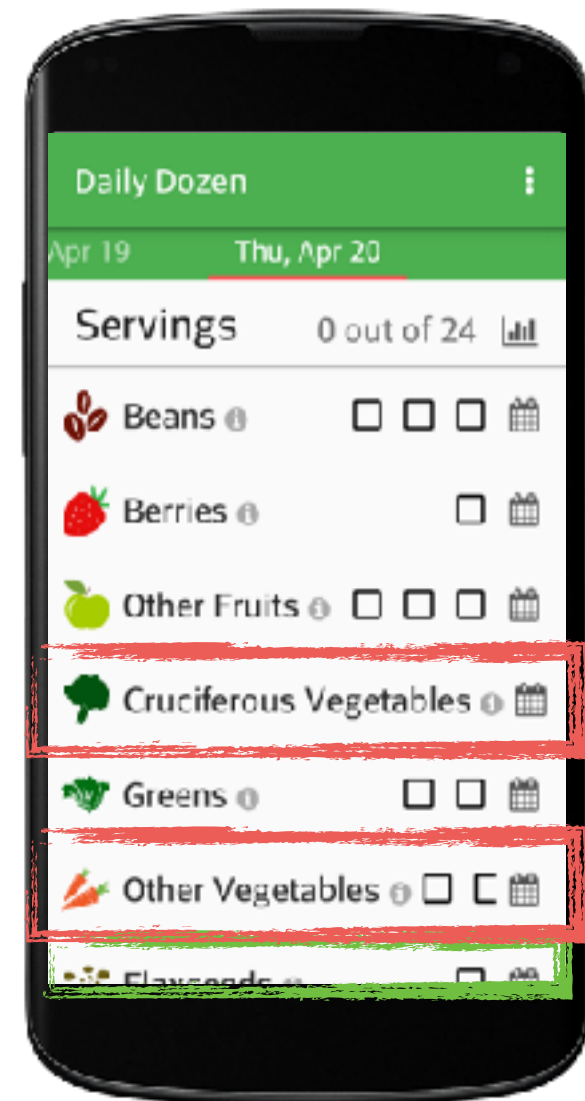


Motivating Example

LG G3



LG Optimus L70

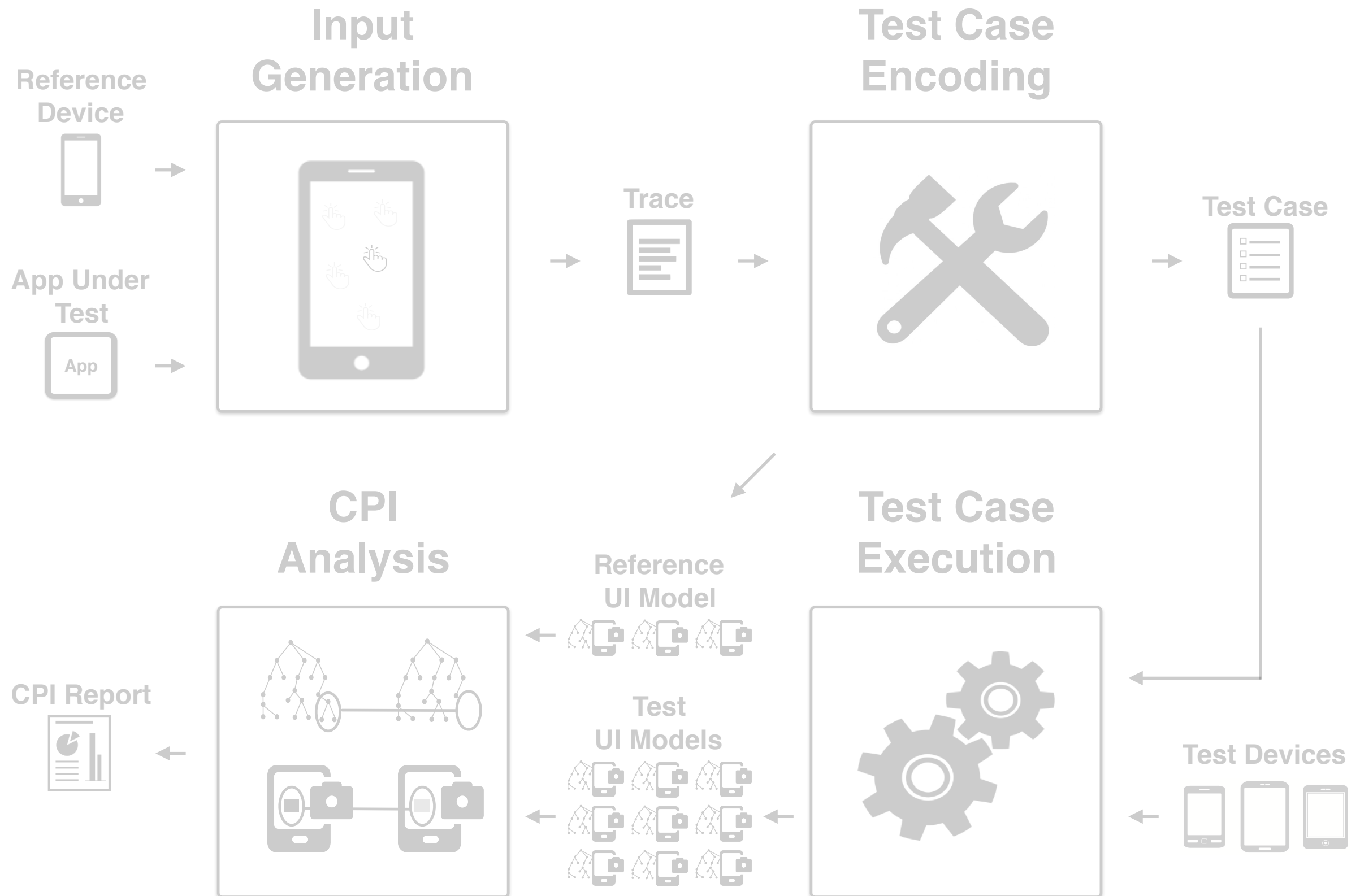


DiffDroid

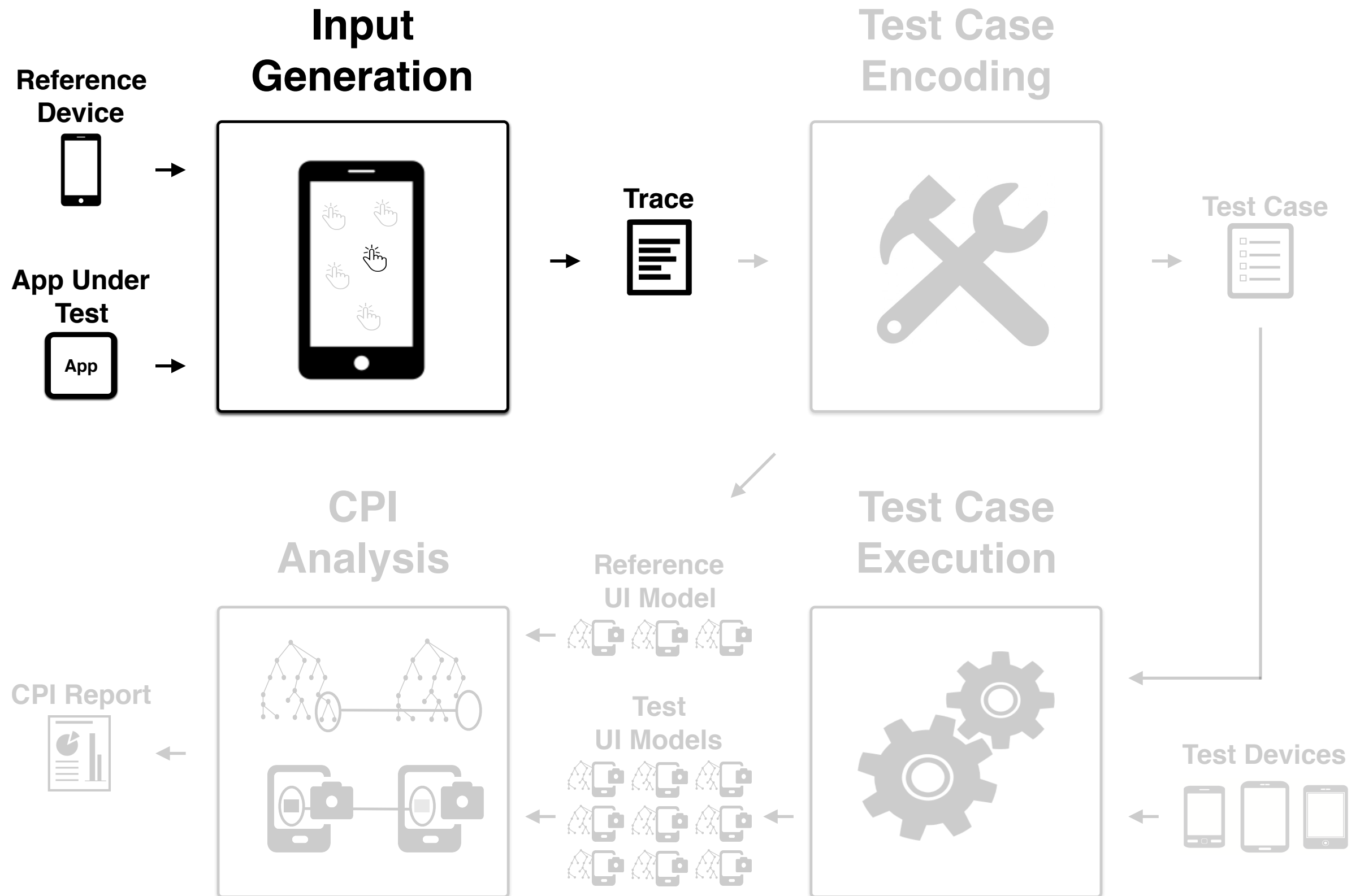
CPI Report



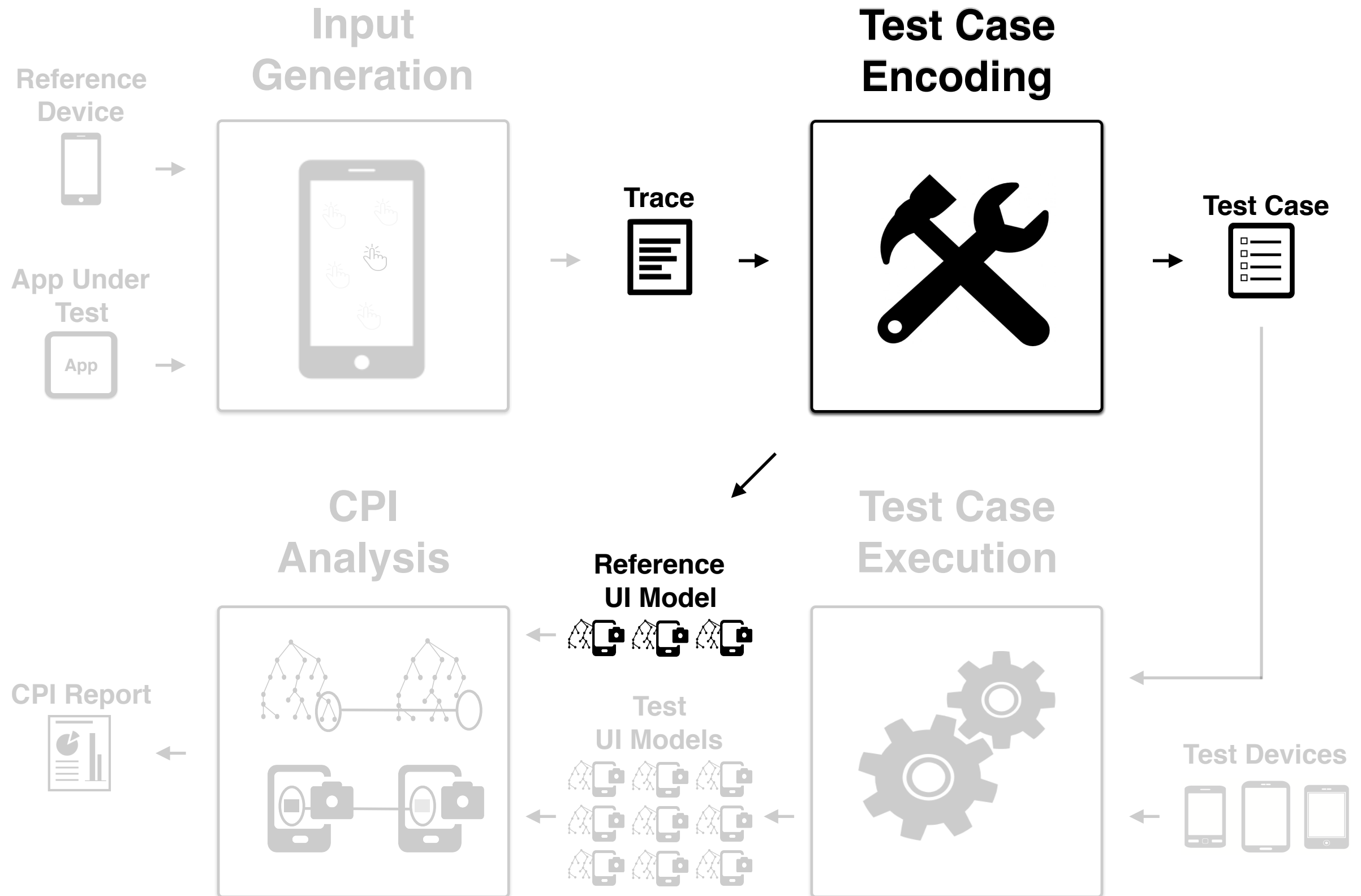
DIFFDROID Overview



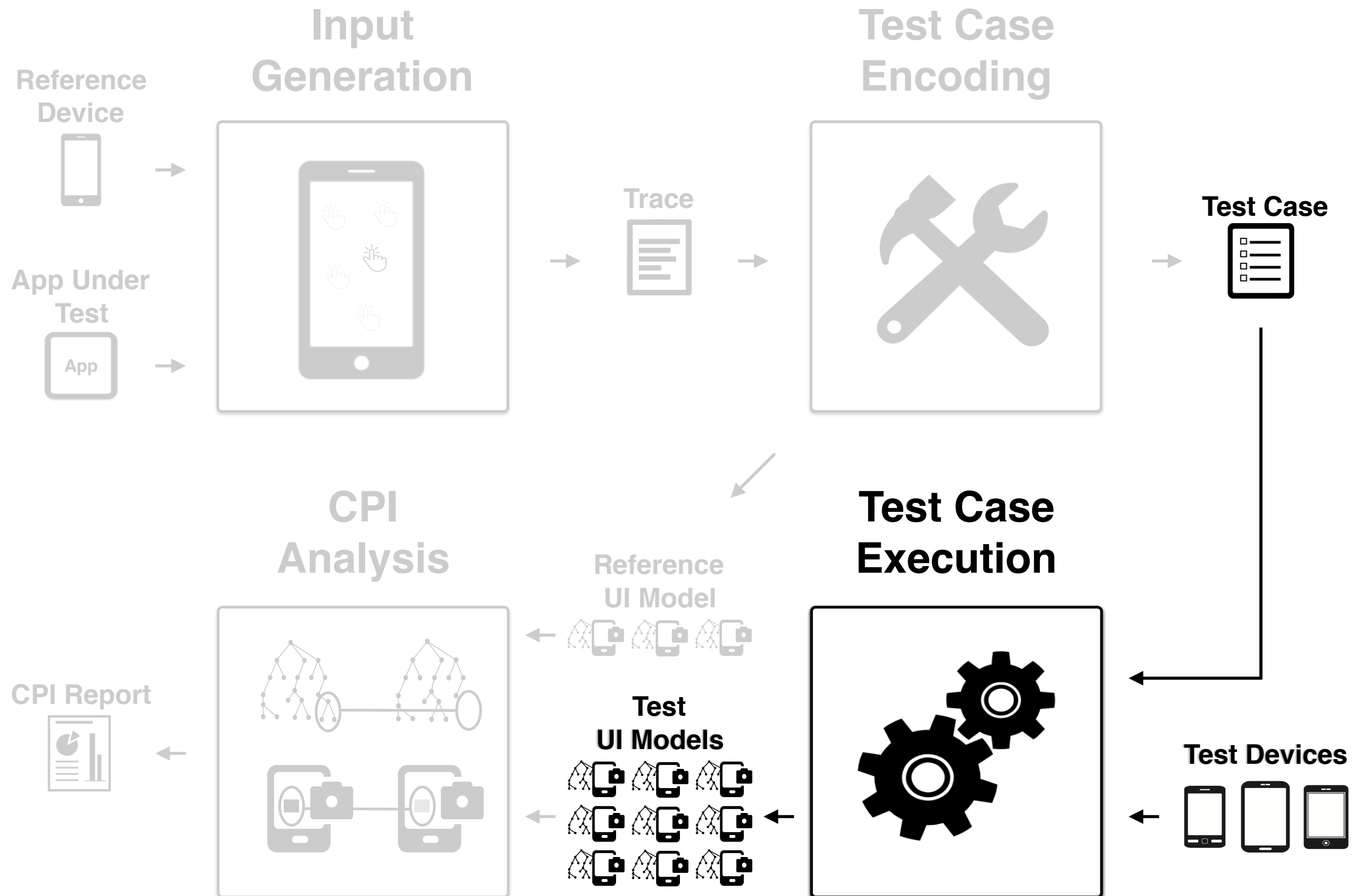
DIFFDROID Overview



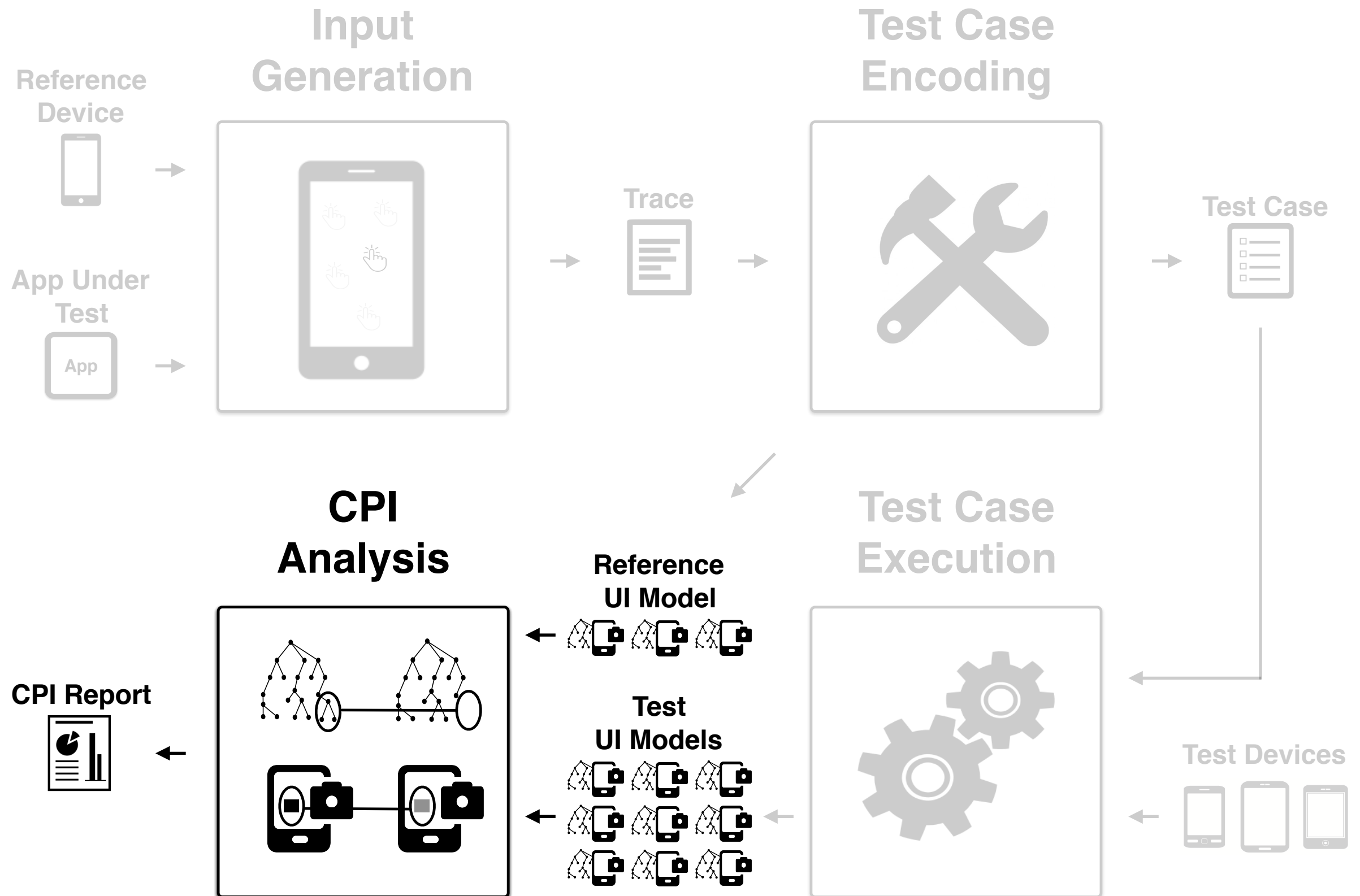
DIFFDROID Overview



DIFFDROID Overview

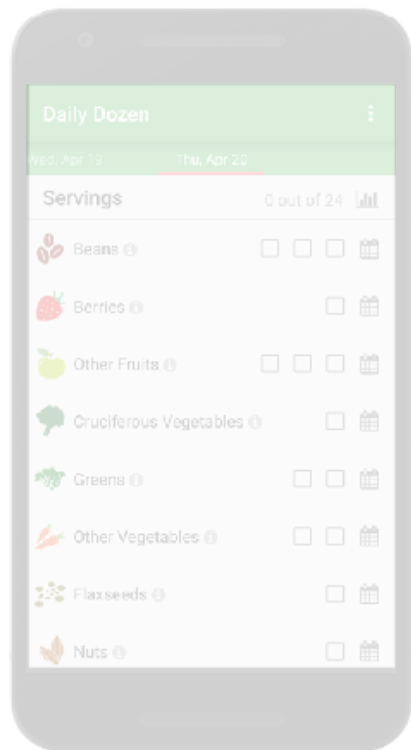


DIFFDROID Overview



Input Generation

Reference Device



Inputs

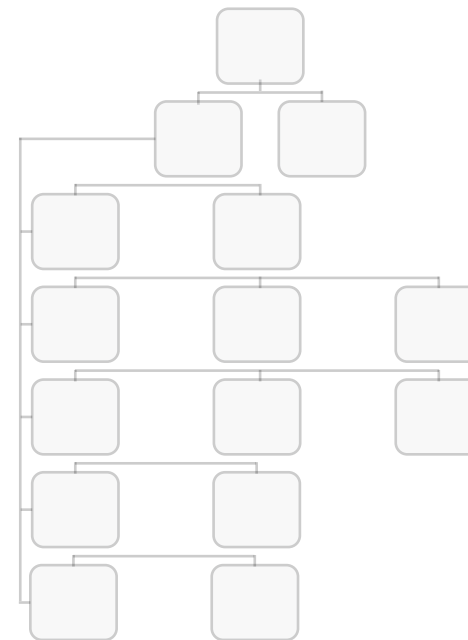
Key

System

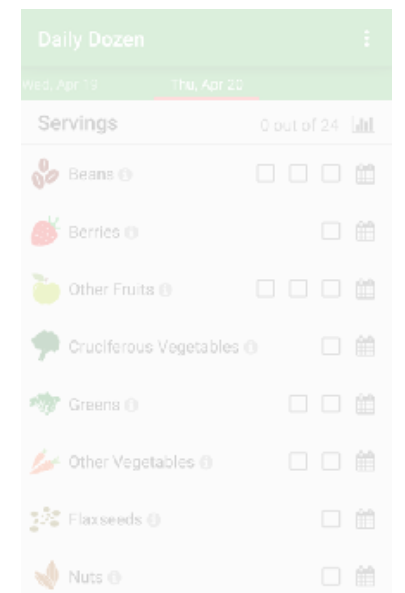
Touch

Window Model

UI Hierarchy

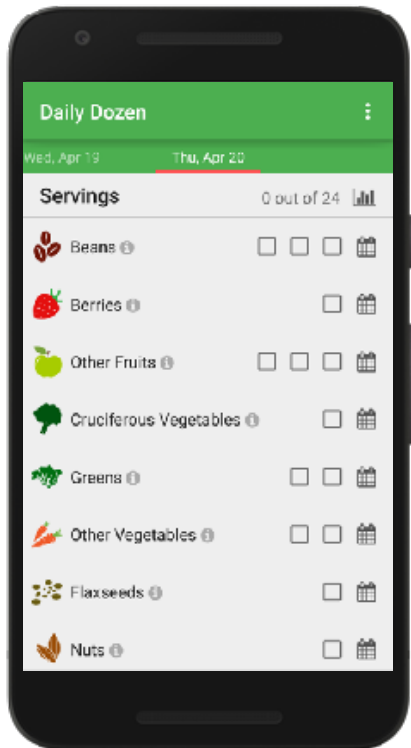


Screenshot



Input Generation

Reference Device



Inputs

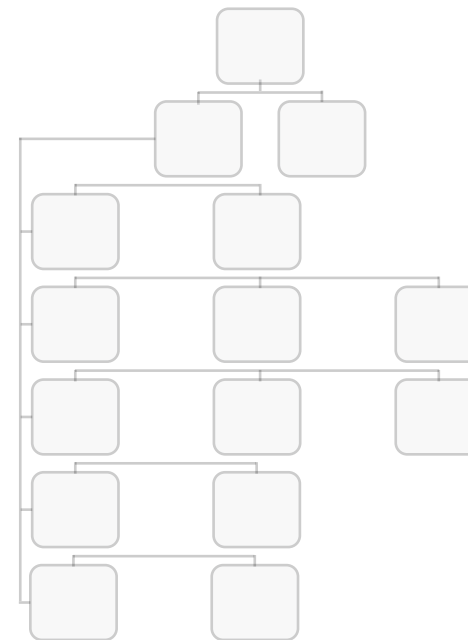
Key

System

Touch

Window Model

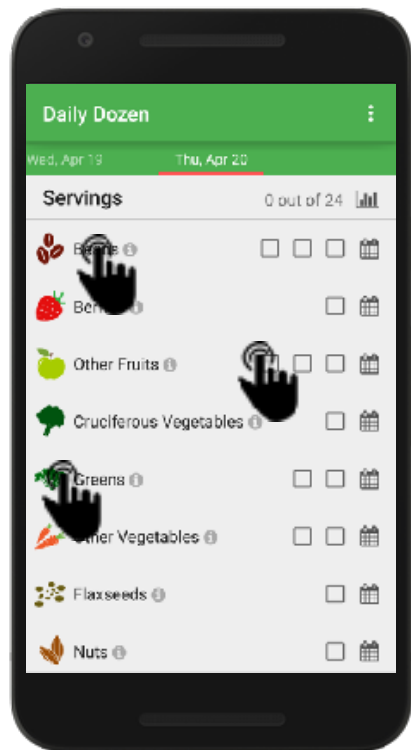
UI Hierarchy



Screenshot

Input Generation

Reference Device



Inputs

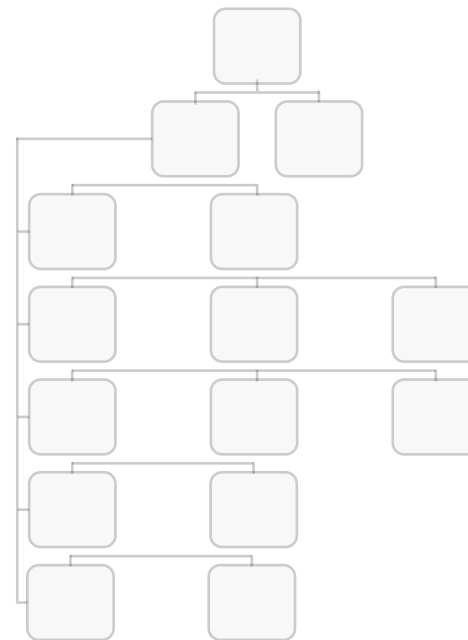
Key

System

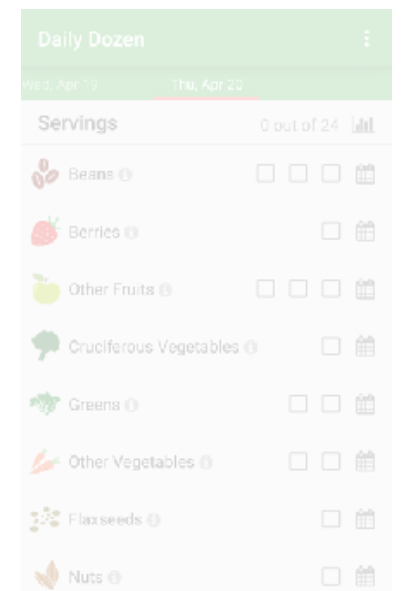
Touch

Window Model

UI Hierarchy

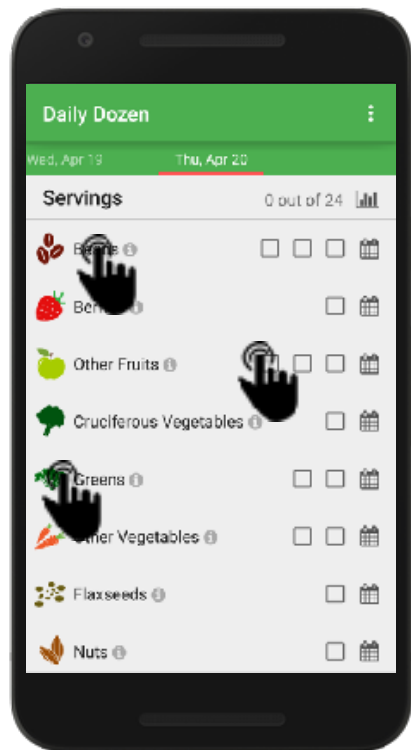


Screenshot



Input Generation

Reference Device



Inputs

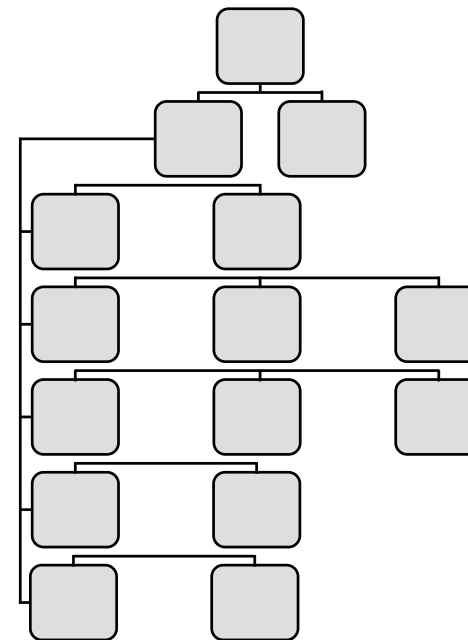
Key

System

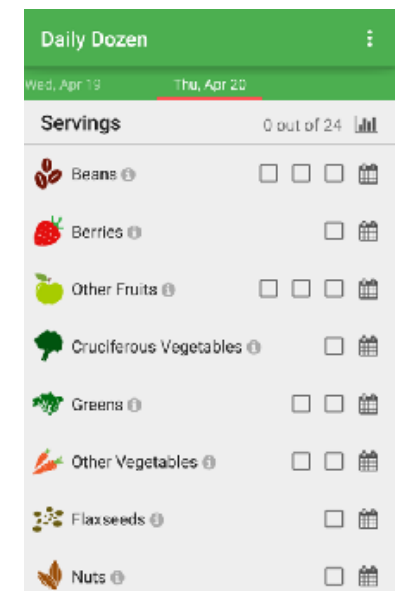
Touch

Window Model

UI Hierarchy

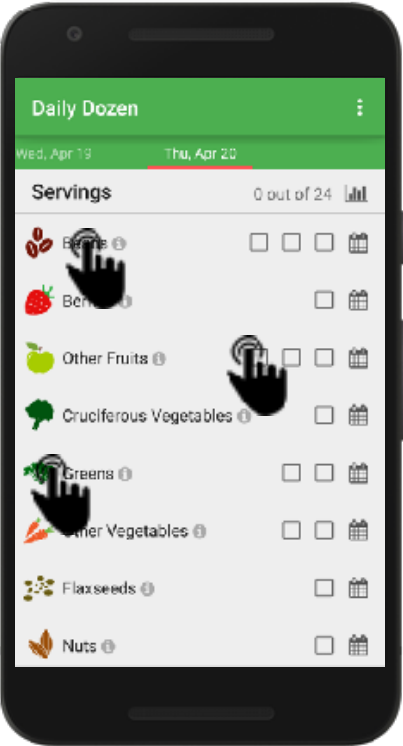


Screenshot

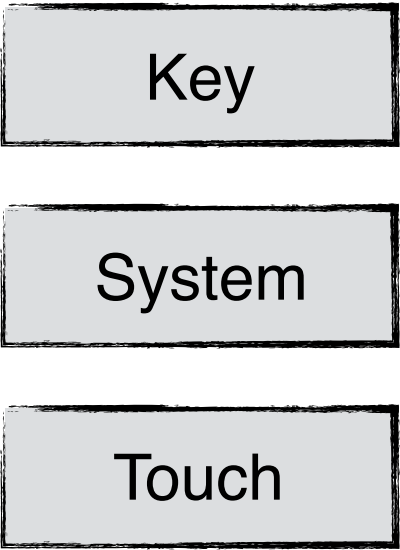


Input Generation

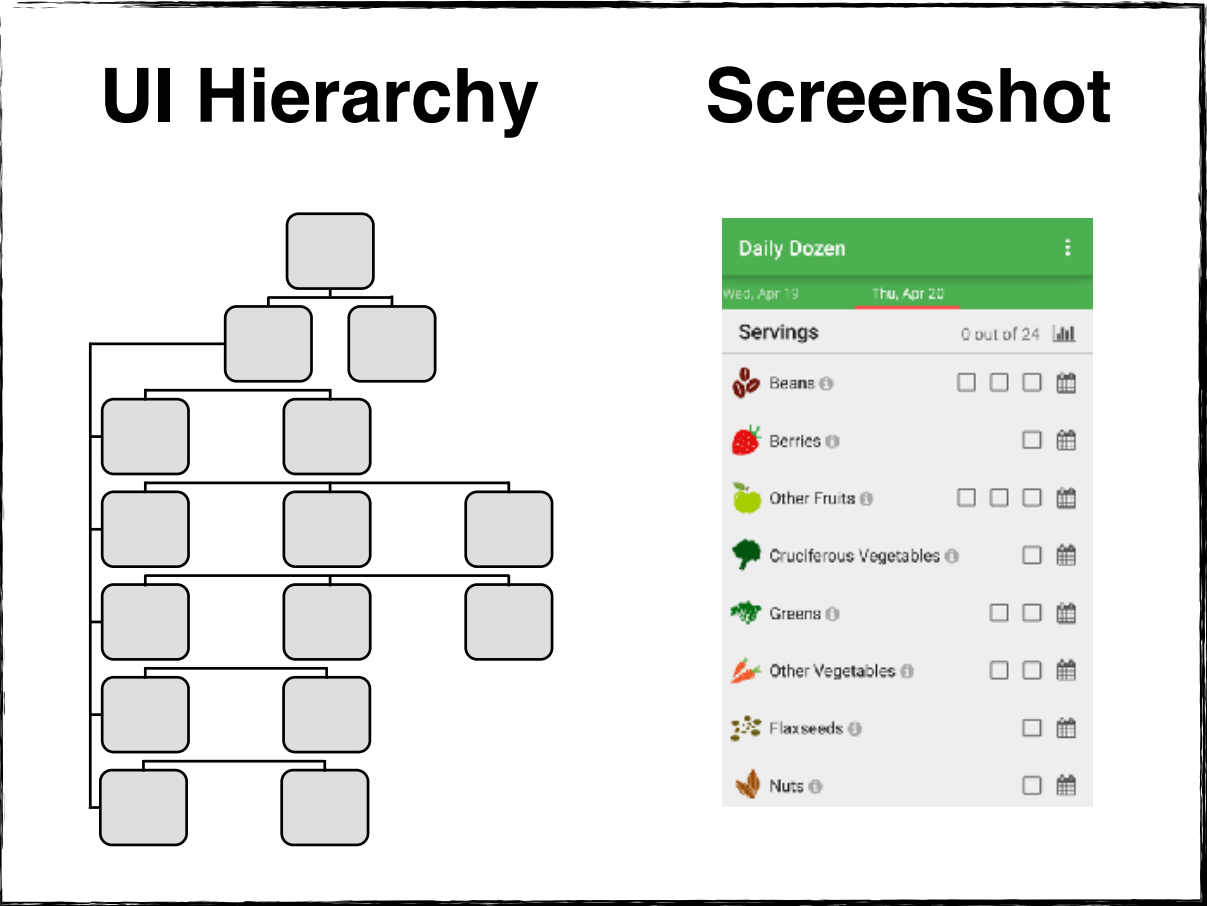
Reference Device



Inputs



Window Model



Trace

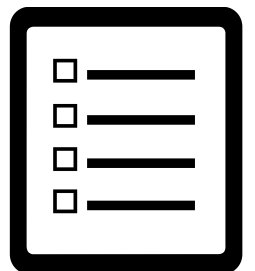


Test Case Encoding

Trace



Test Case



**Reference
UI Model**



Test Case Encoding

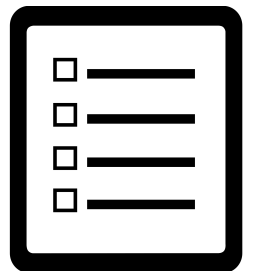
Trace



Input



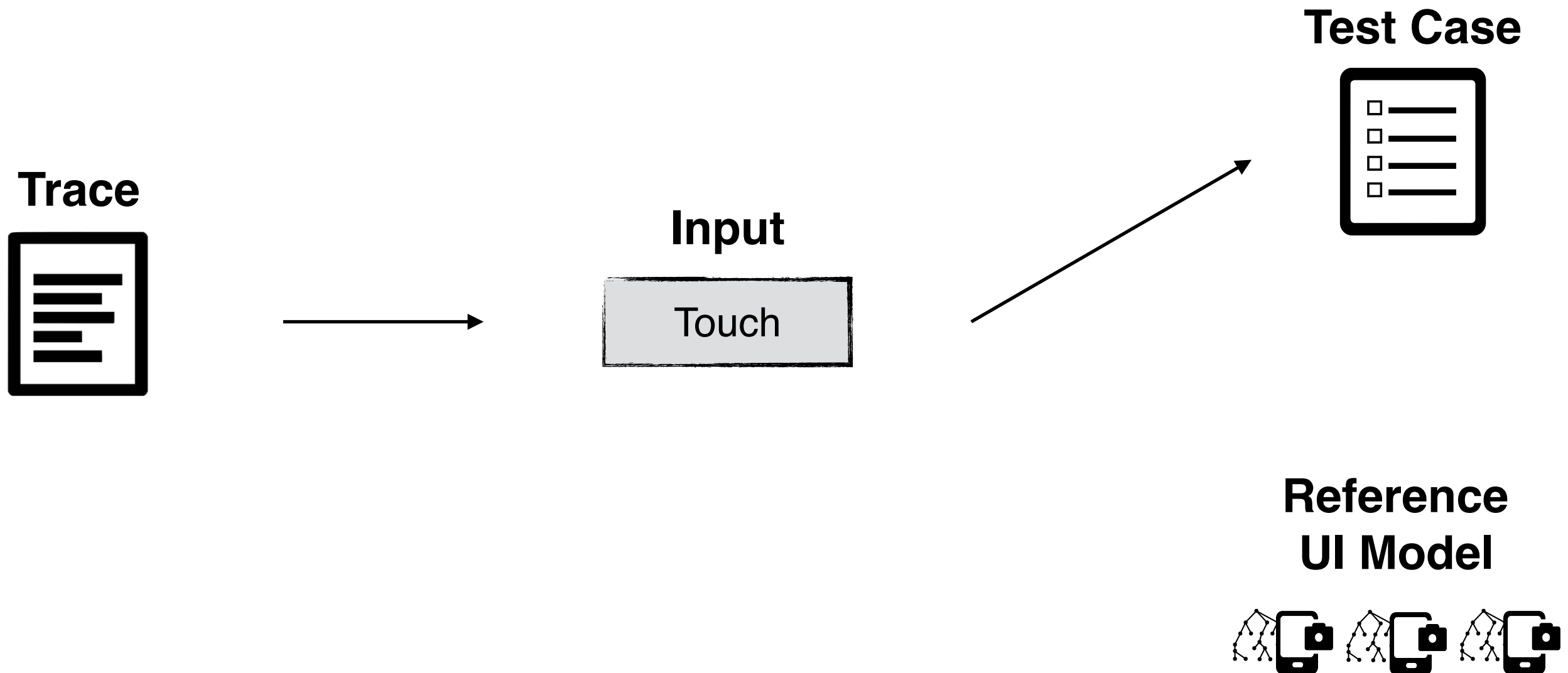
Test Case



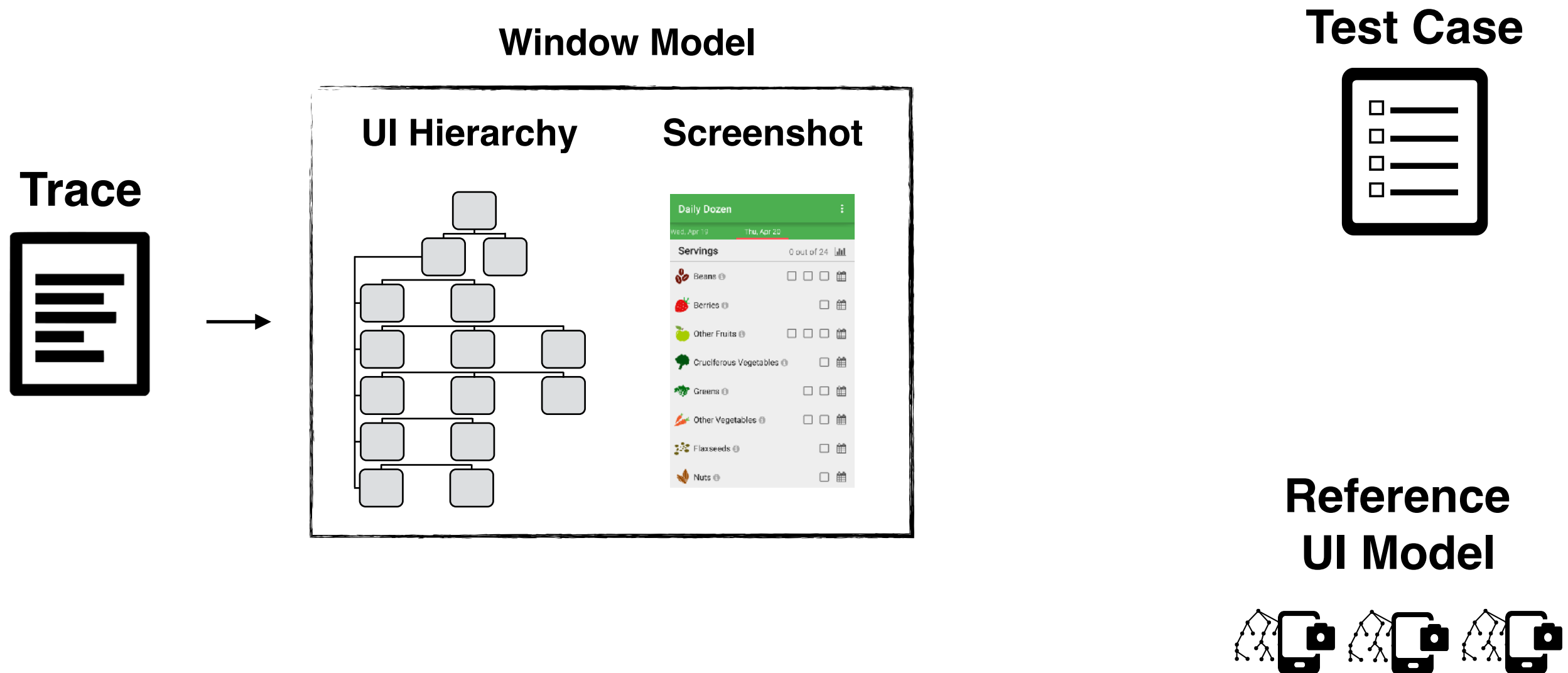
Reference
UI Model



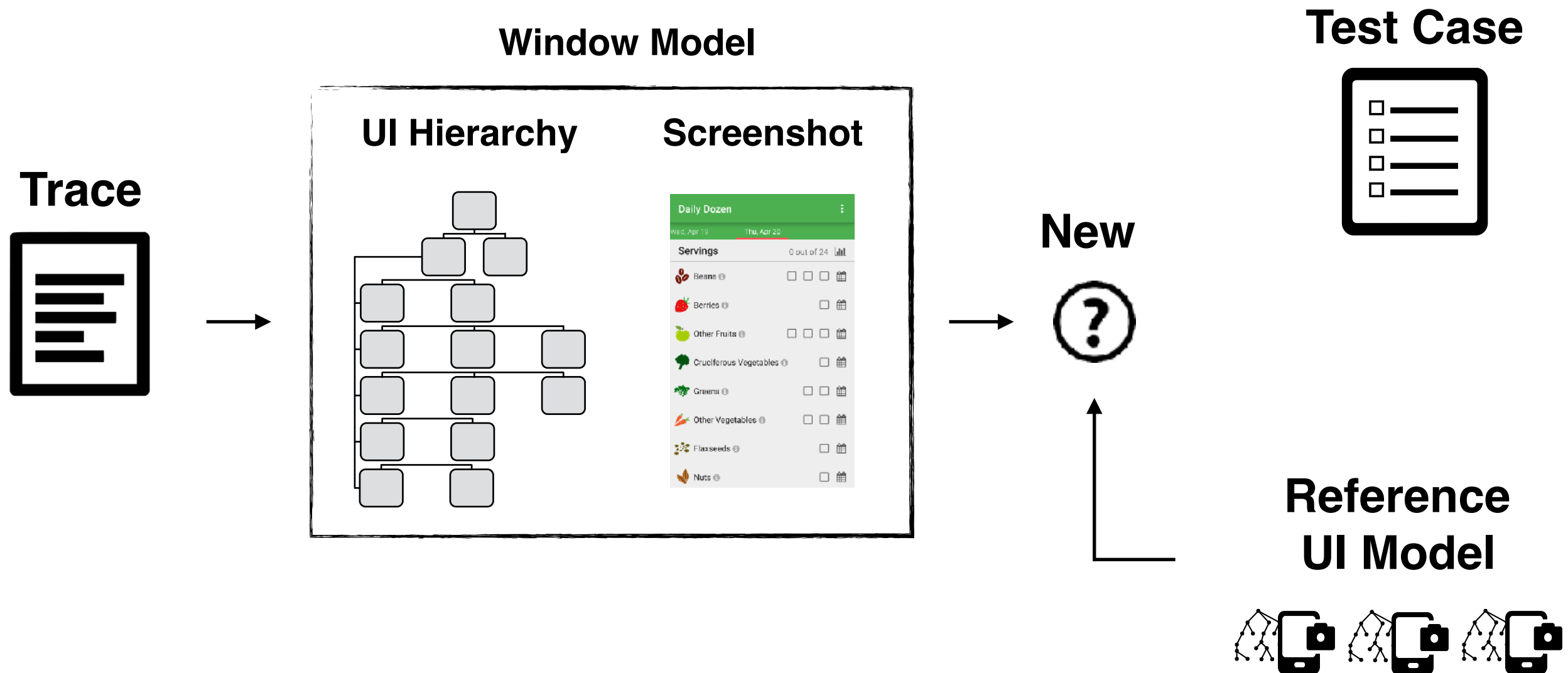
Test Case Encoding



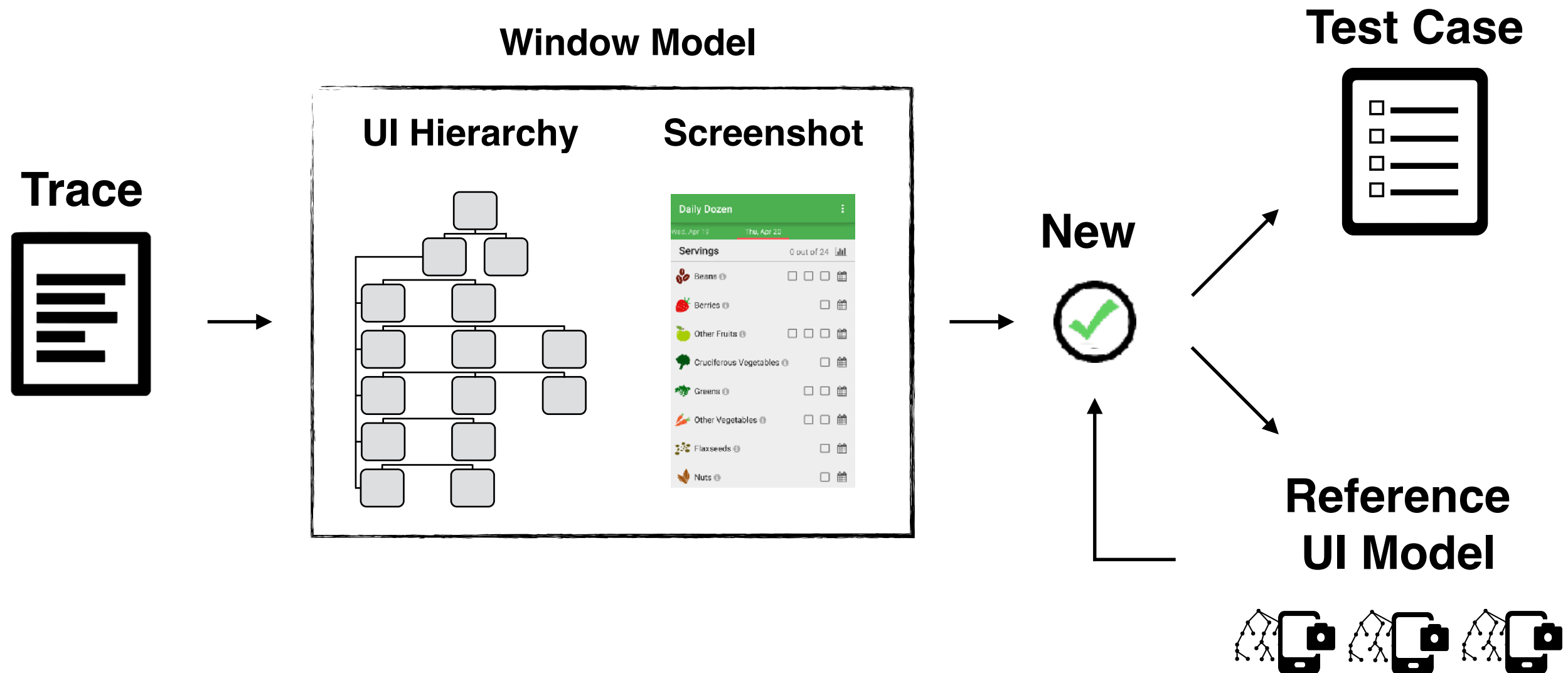
Test Case Encoding



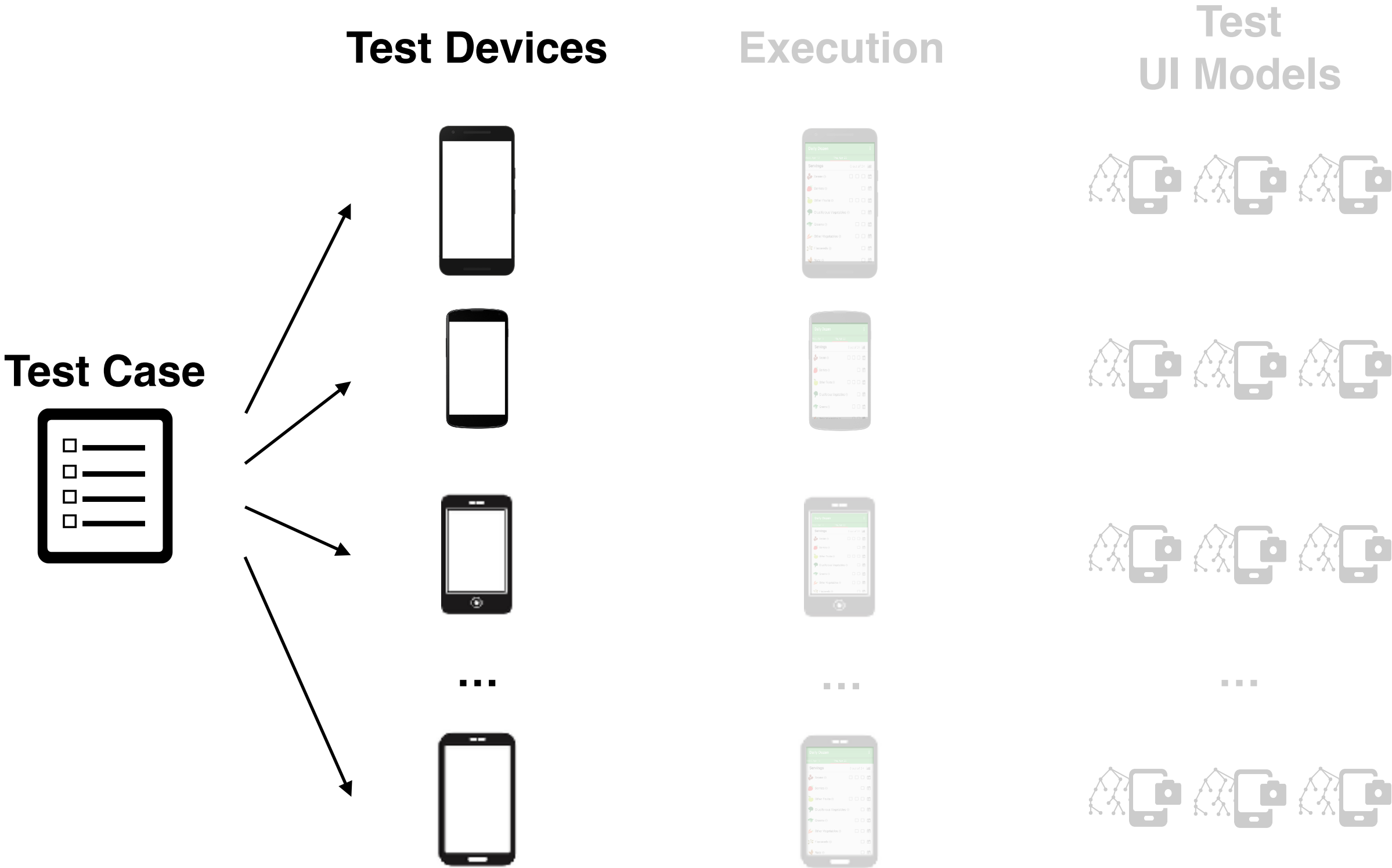
Test Case Encoding



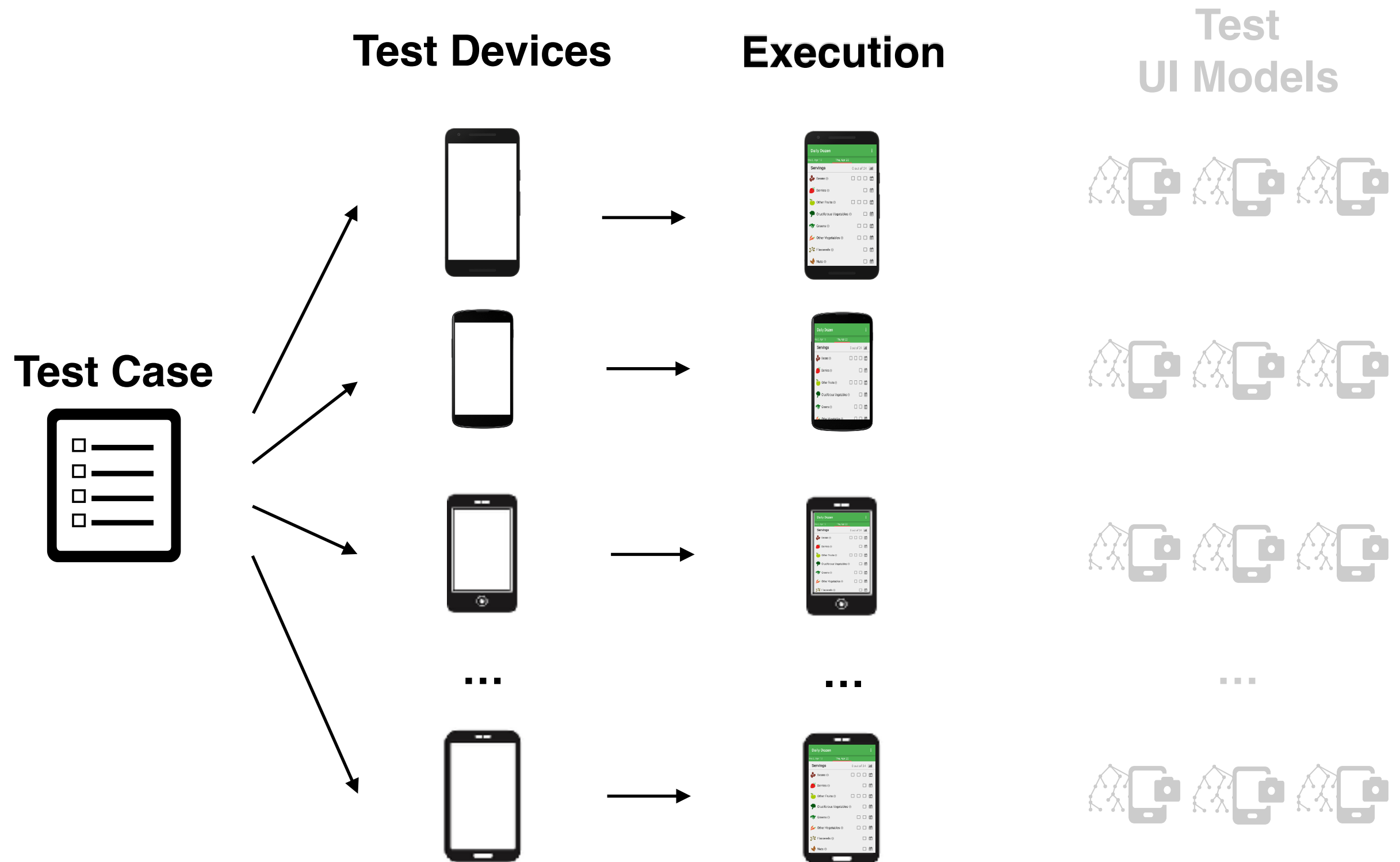
Test Case Encoding



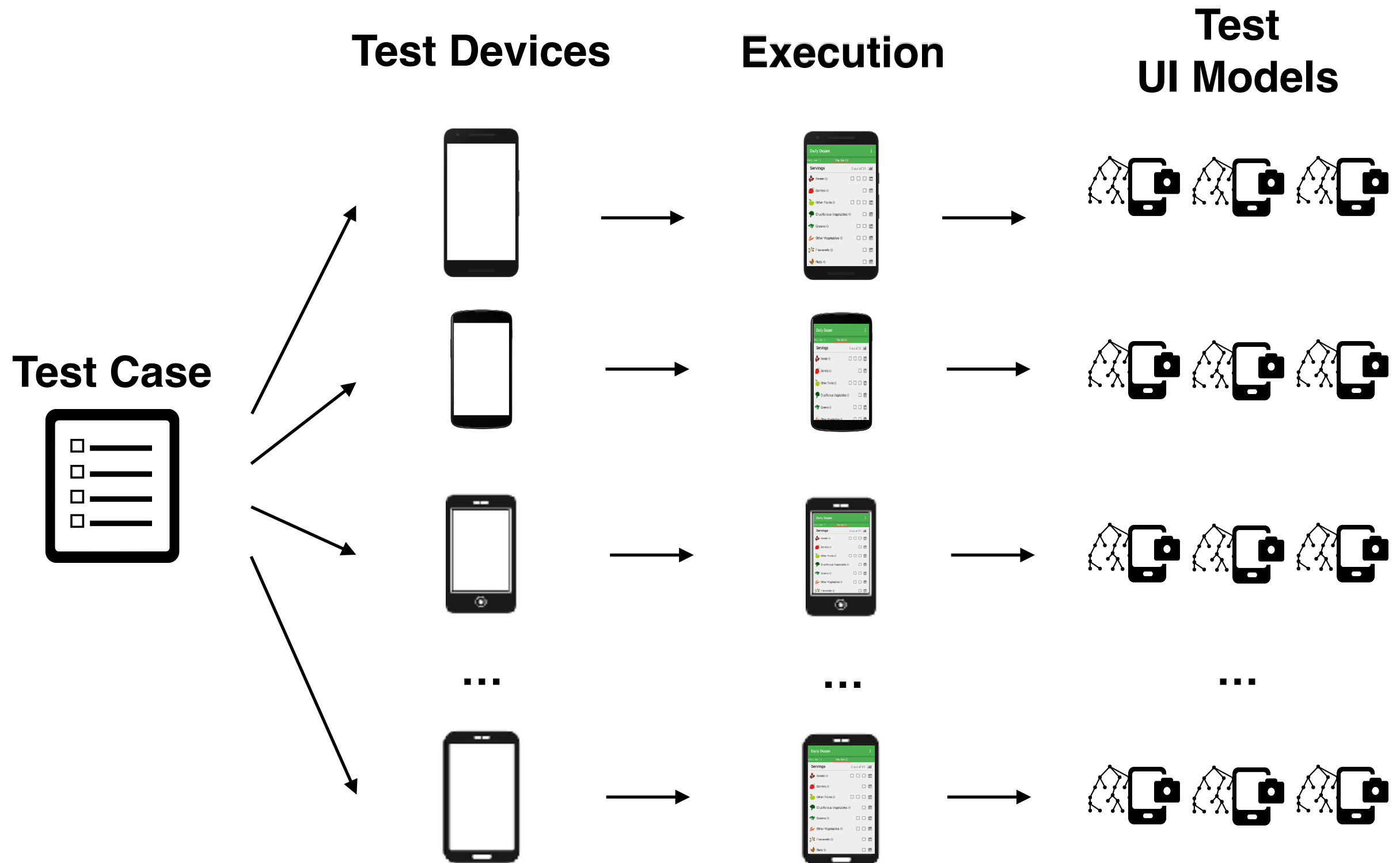
Test Case Execution



Test Case Execution



Test Case Execution



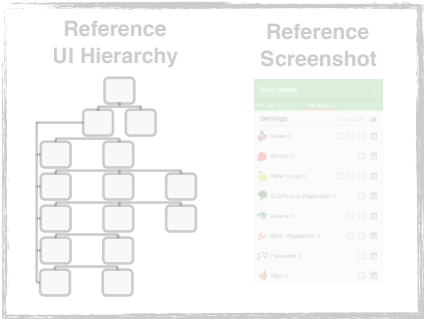
CPI Analysis

Structural Analysis

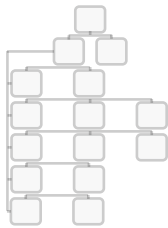
Reference
UI Model



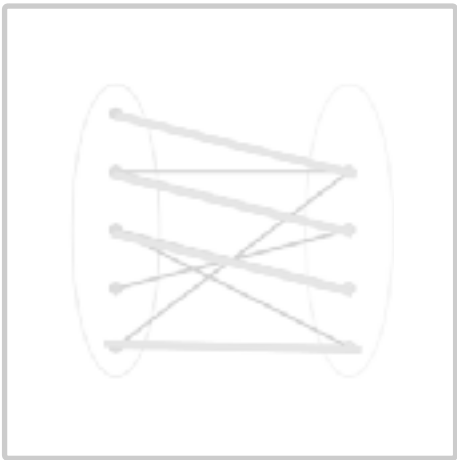
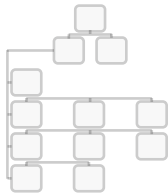
Reference
Window Model



Reference
UI Hierarchy



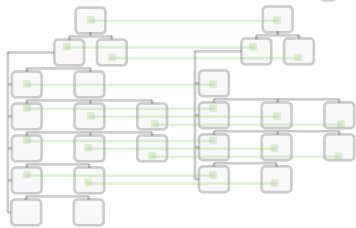
Test
UI Hierarchy



CPI
Report



Node Matching

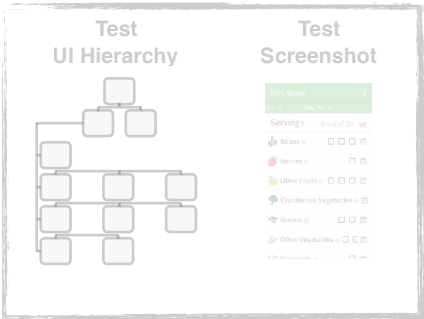


Visual Analysis

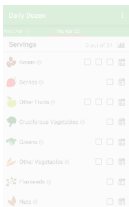
Test
UI Model



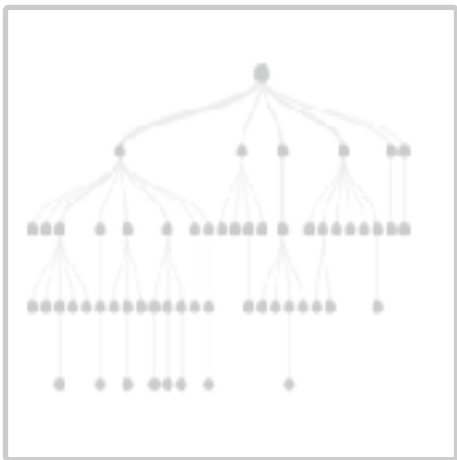
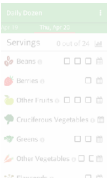
Test
Window Model



Reference
Screenshot



Test
Screenshot



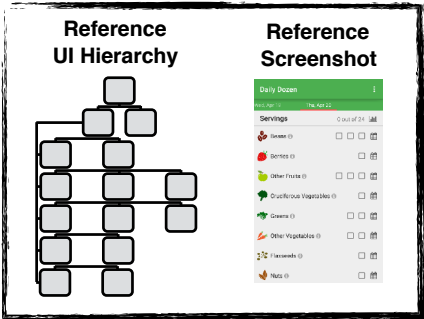
CPI Analysis

Structural Analysis

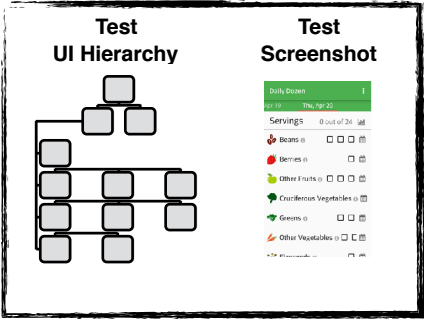
Reference
UI Model



Reference
Window Model



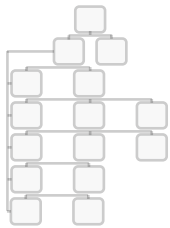
Test
Window Model



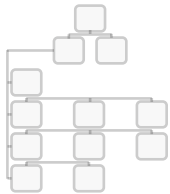
Test
UI Model



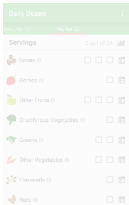
Reference
UI Hierarchy



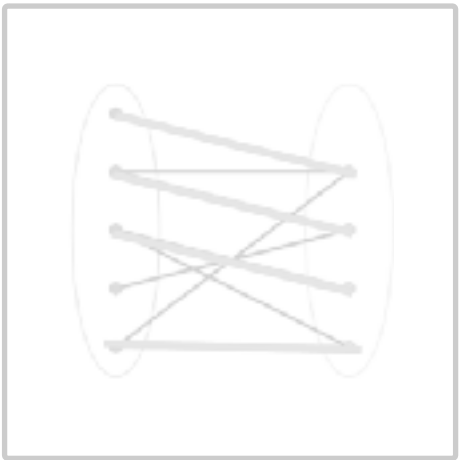
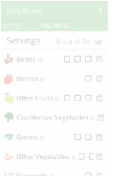
Test
UI Hierarchy



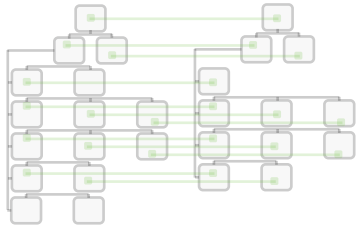
Reference
Screenshot



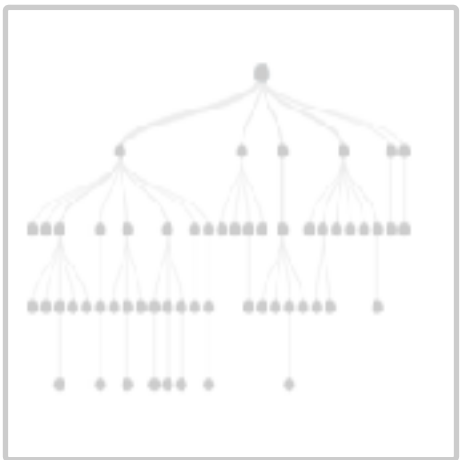
Test
Screenshot



Node Matching



Visual Analysis



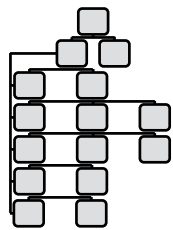
CPI
Report



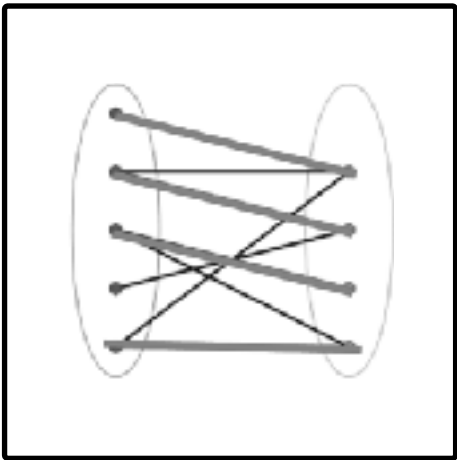
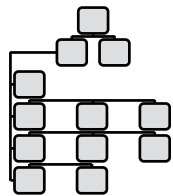
CPI Analysis

Structural Analysis

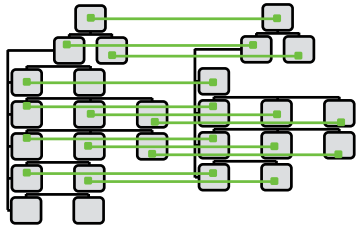
Reference
UI Hierarchy



Test
UI Hierarchy



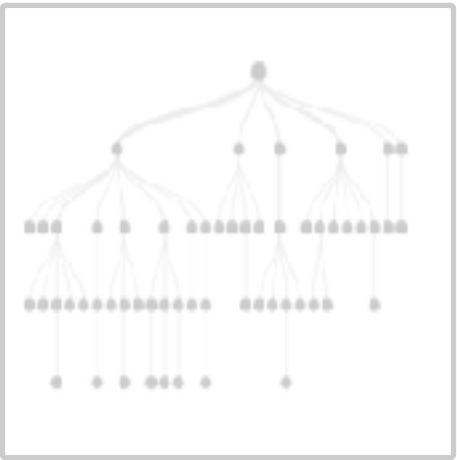
Node Matching



CPI
Report



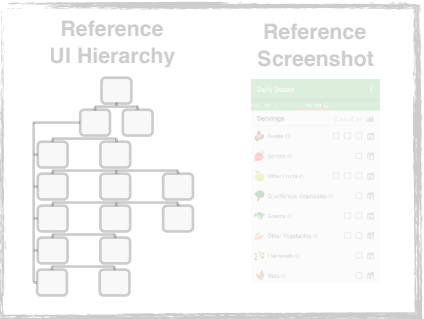
Visual Analysis



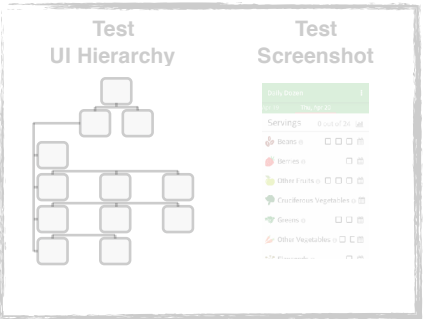
Reference
UI Model



Reference
Window Model



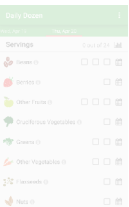
Test
Window Model



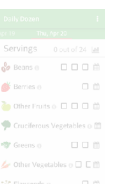
Test
UI Model



Reference
Screenshot



Test
Screenshot



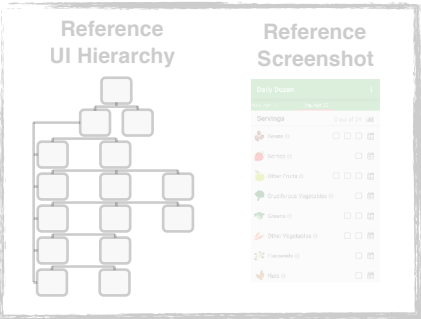
CPI Analysis

Structural Analysis

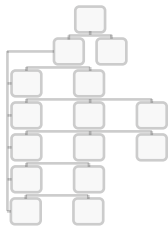
Reference
UI Model



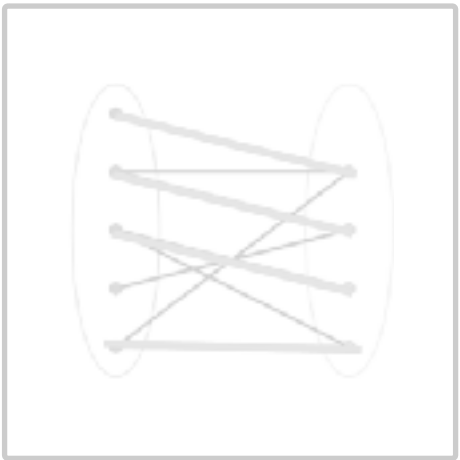
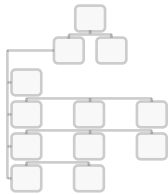
Reference
Window Model



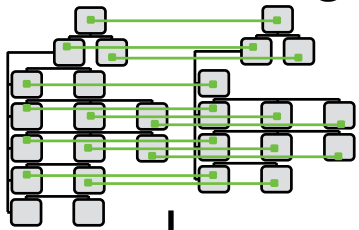
Reference
UI Hierarchy



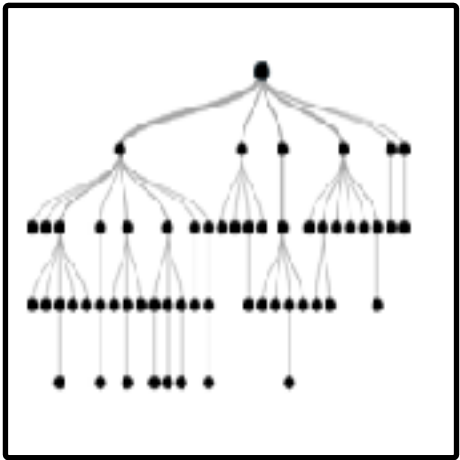
Test
UI Hierarchy



Node Matching



Visual Analysis



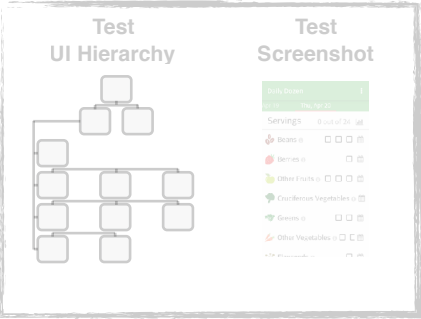
CPI
Report



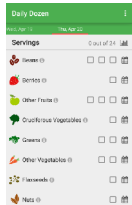
Test
UI Model



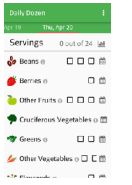
Test
Window Model



Reference
Screenshot



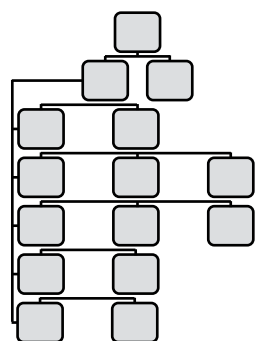
Test
Screenshot



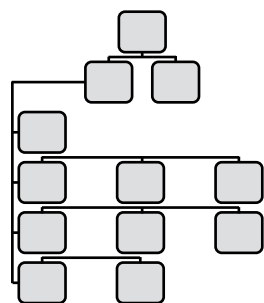
Structural Analysis

Node Similarity

Reference UI Hierarchy



Test UI Hierarchy



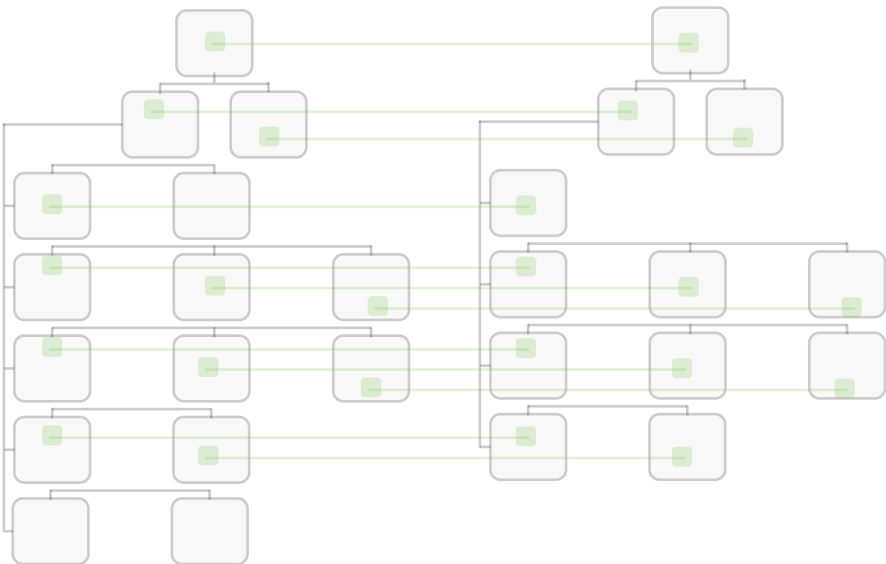
Resource ID

XPath

Properties

checkable	checked
focusable	focused
clickable	selected
long-clickable	enabled
scrollable	text

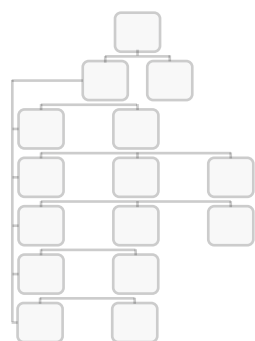
Node Matching



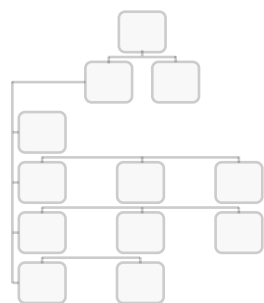
Structural Analysis

Node Similarity

Reference
UI Hierarchy



Test
UI Hierarchy



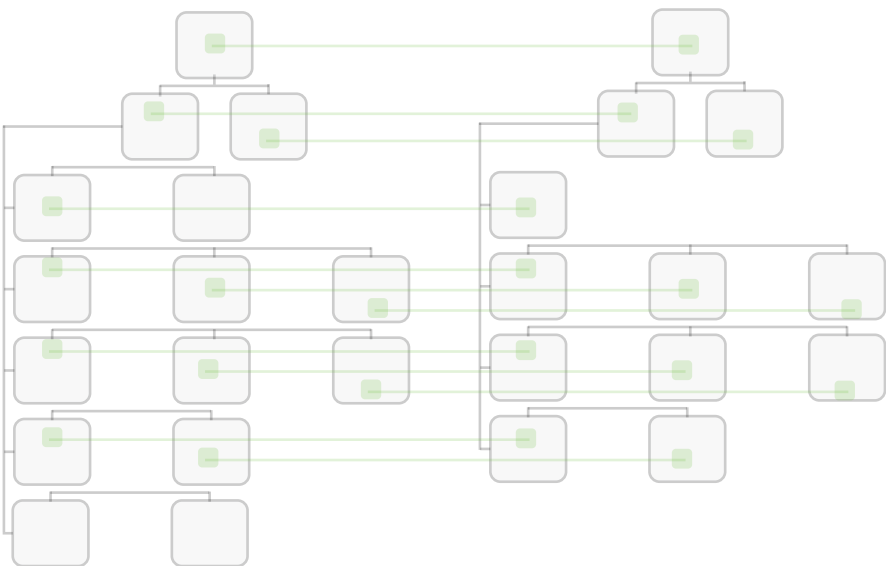
Resource ID

XPath

Properties

checkable	checked
focusable	focused
clickable	selected
long-clickable	enabled
scrollable	text

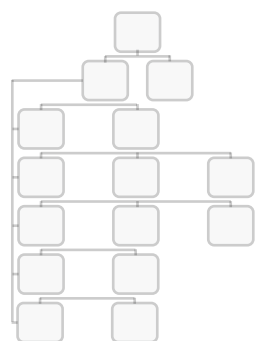
Node Matching



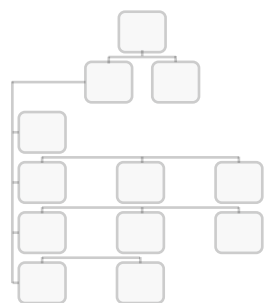
Structural Analysis

Node Similarity

Reference
UI Hierarchy



Test
UI Hierarchy



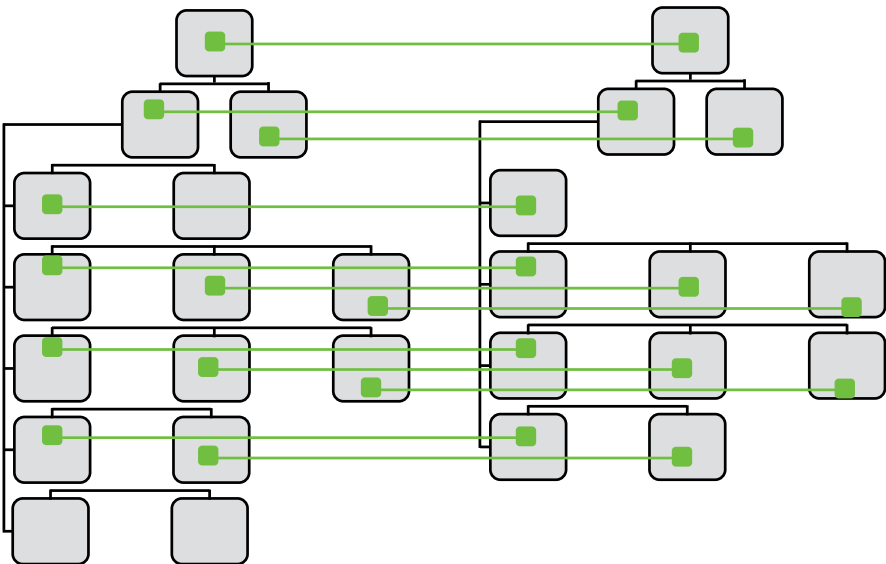
Resource ID

XPath

Properties

checkable	checked
focusable	focused
clickable	selected
long-clickable	enabled
scrollable	text

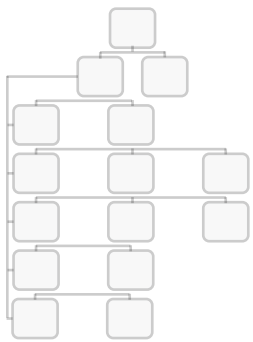
Node Matching



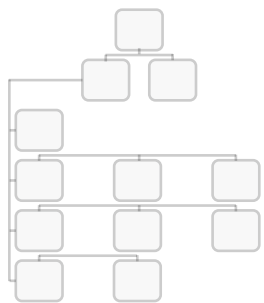
Structural Analysis

Node Similarity

Reference
UI Hierarchy



Test
UI Hierarchy



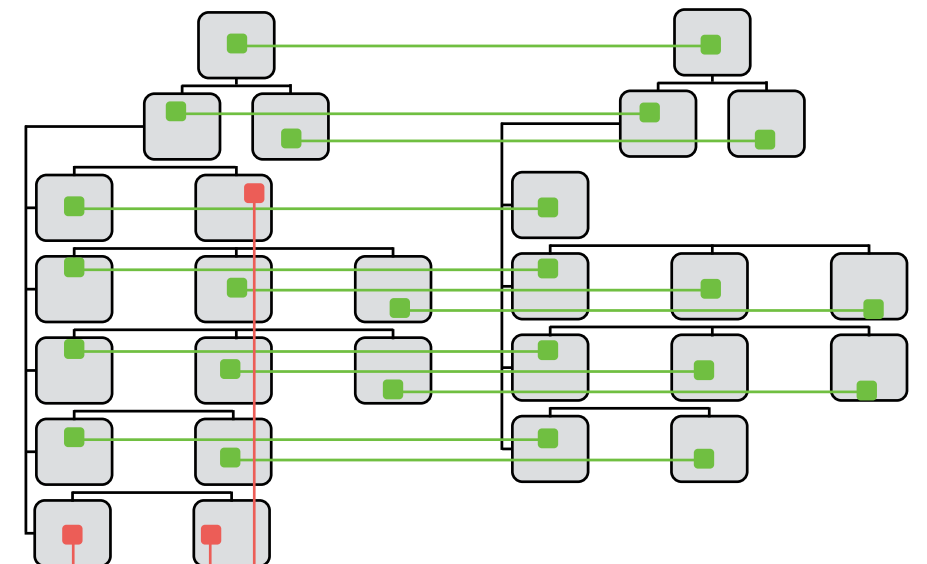
Resource ID

XPath

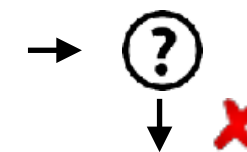
Properties

checkable	checked
focusable	focused
clickable	selected
long-clickable	enabled
scrollable	text

Node Matching



Variable



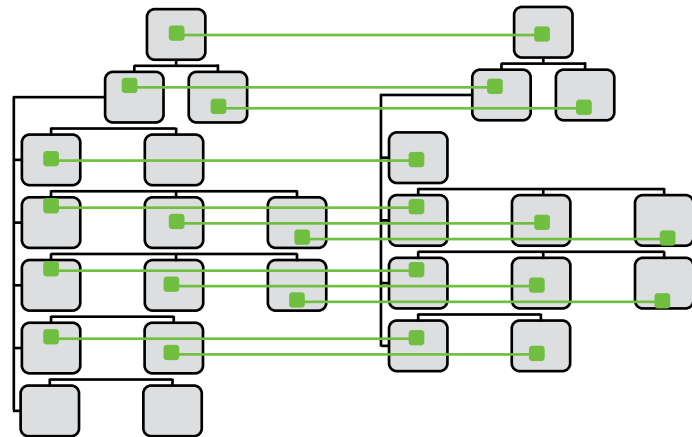
Structural
Inconsistency

CPI
Report

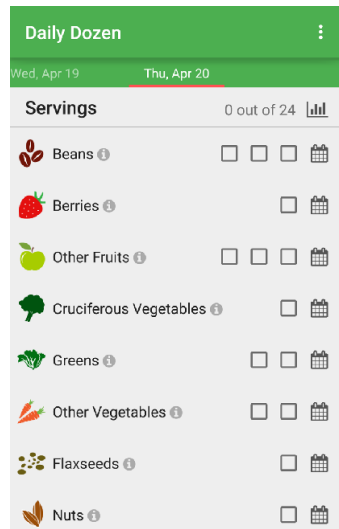


Visual Analysis

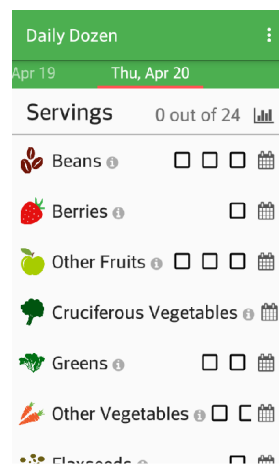
Node Matching



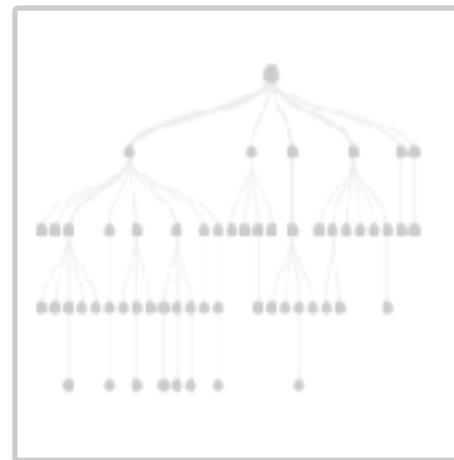
Reference Screenshot



Test Screenshot



CW-SSIM EMD RRC OCR
↓ ↓ ↓ ↓
C4.5 Decision Tree Classifier



Complex-Wavelet
Structural Similarity Index
(CW-SSIM)

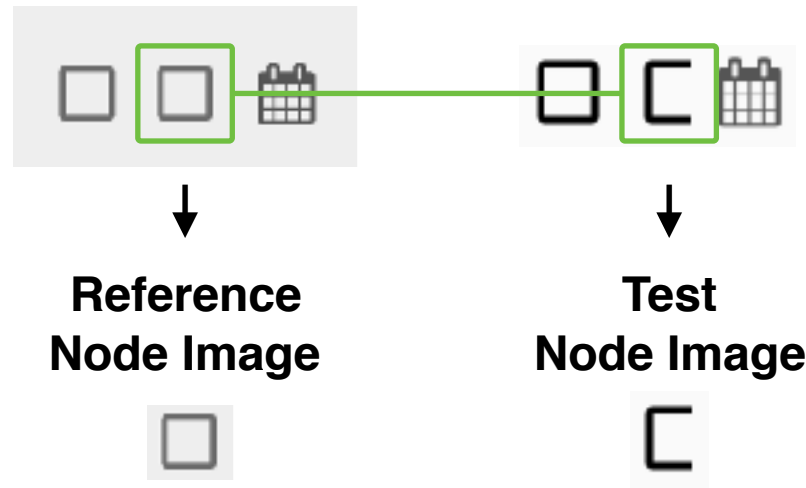
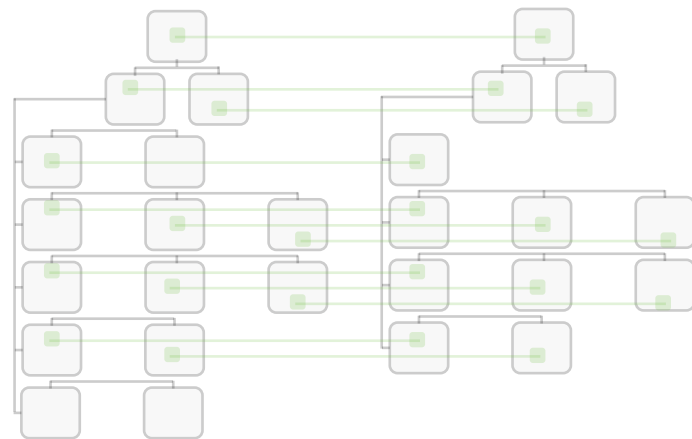
Earth Mover Distance
of Color Histogram
(EMD)

Relative Ratio
Change
(RRC)

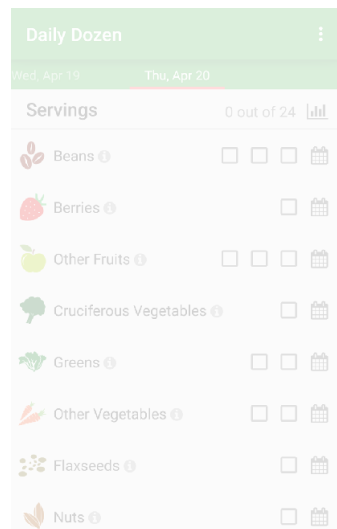
Optical Character
Recognition
(OCR)

Visual Analysis

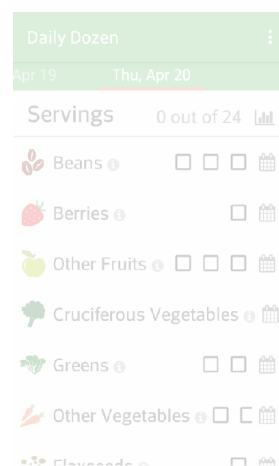
Node Matching



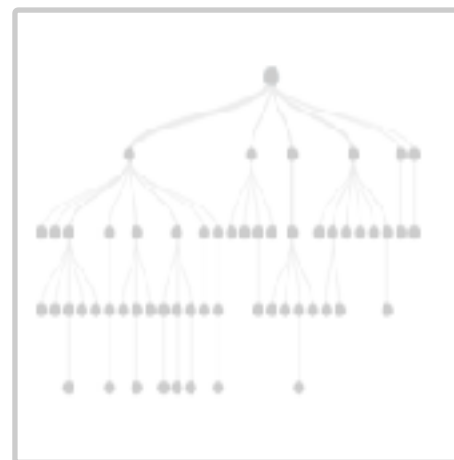
Reference Screenshot



Test Screenshot



CW-SSIM EMD RRC OCR
↓ ↓ ↓ ↓
C4.5 Decision Tree Classifier



Complex-Wavelet
Structural Similarity Index
(CW-SSIM)

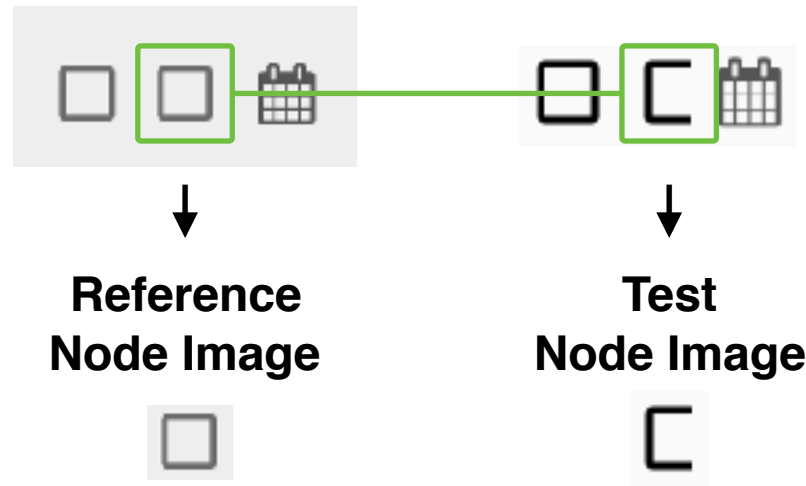
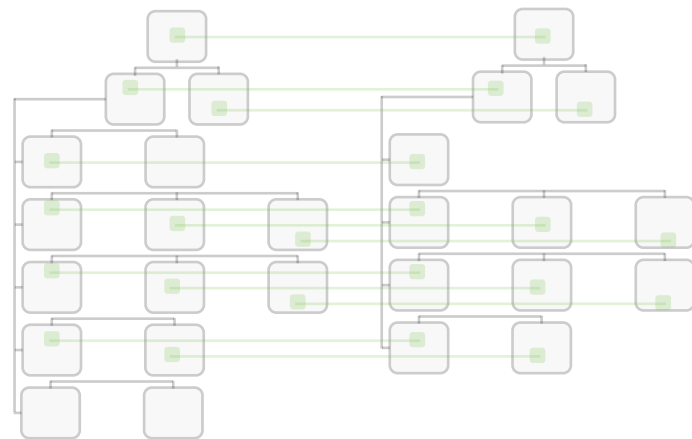
Earth Mover Distance
of Color Histogram
(EMD)

Relative Ratio
Change
(RRC)

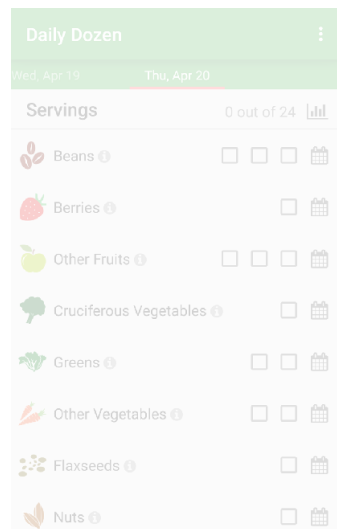
Optical Character
Recognition
(OCR)

Visual Analysis

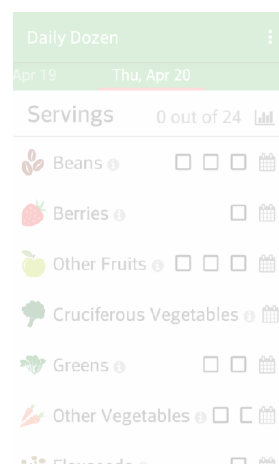
Node Matching



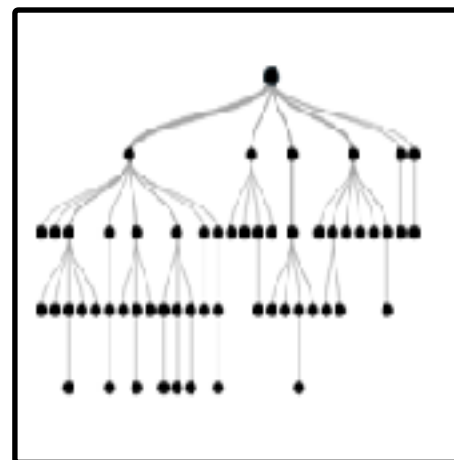
Reference Screenshot



Test Screenshot



CW-SSIM EMD RRC OCR
↓ ↓ ↓ ↓
C4.5 Decision Tree Classifier



Complex-Wavelet
Structural Similarity Index
(CW-SSIM)

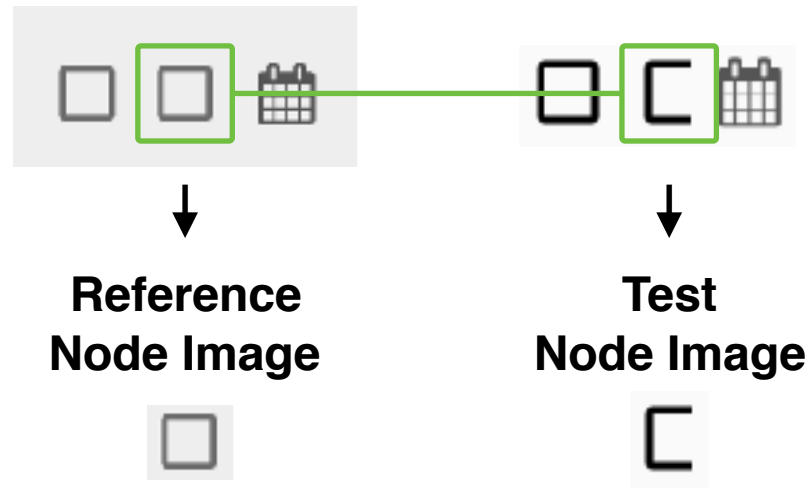
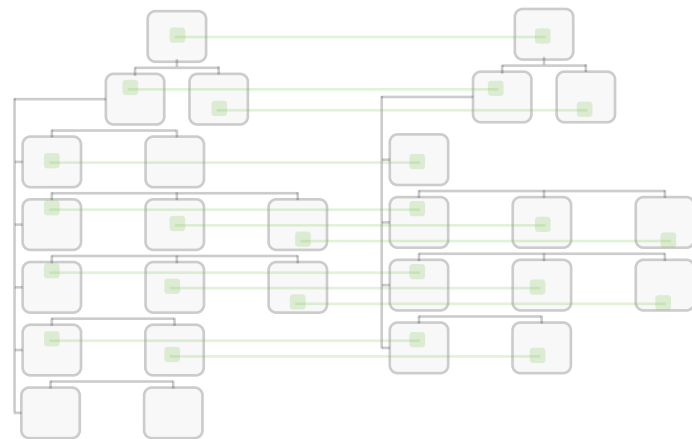
Earth Mover Distance
of Color Histogram
(EMD)

Relative Ratio
Change
(RRC)

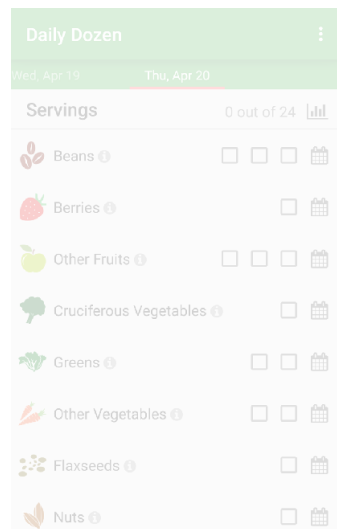
Optical Character
Recognition
(OCR)

Visual Analysis

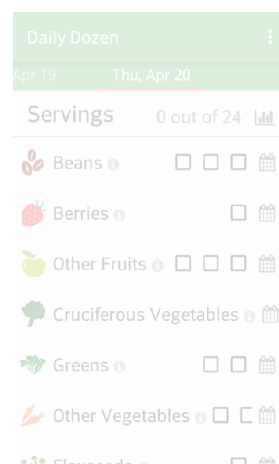
Node Matching



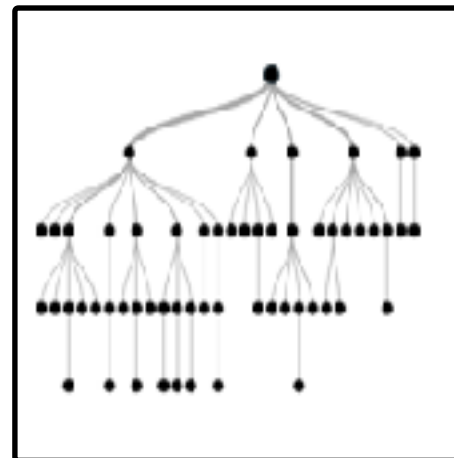
Reference Screenshot



Test Screenshot



CW-SSIM EMD RRC OCR
↓ ↓ ↓ ↓
C4.5 Decision Tree Classifier



**Complex-Wavelet
Structural Similarity Index
(CW-SSIM)**

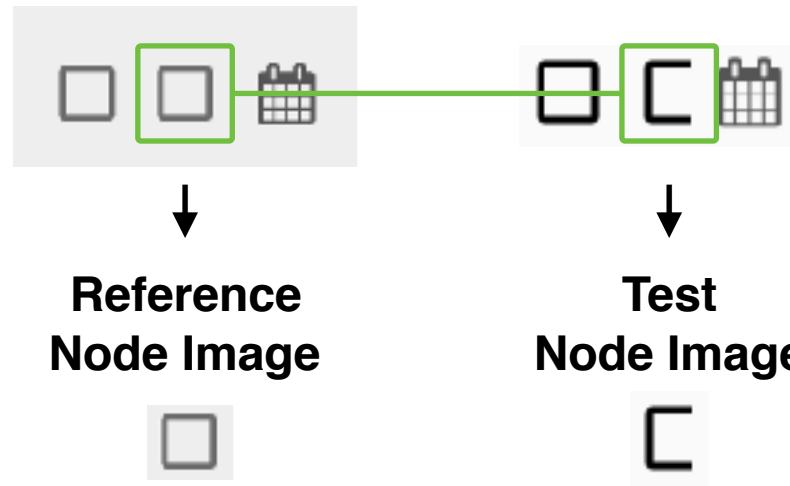
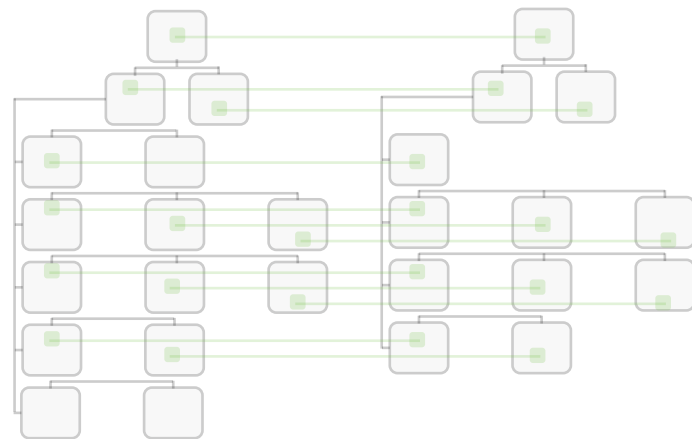
**Earth Mover Distance
of Color Histogram
(EMD)**

**Relative Ratio
Change
(RRC)**

**Optical Character
Recognition
(OCR)**

Visual Analysis

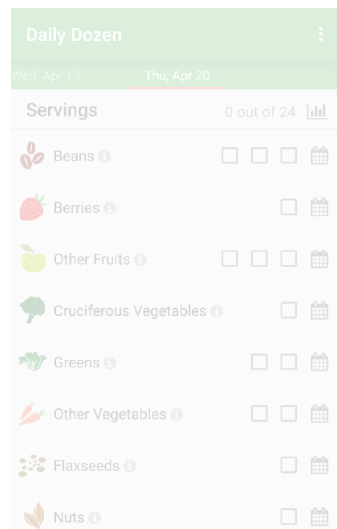
Node Matching



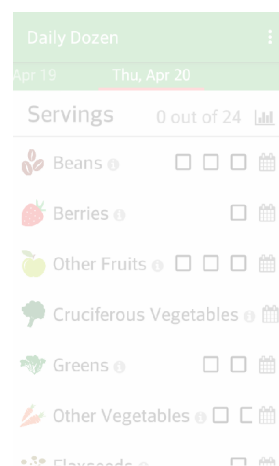
Reference
Node Image

Test
Node Image

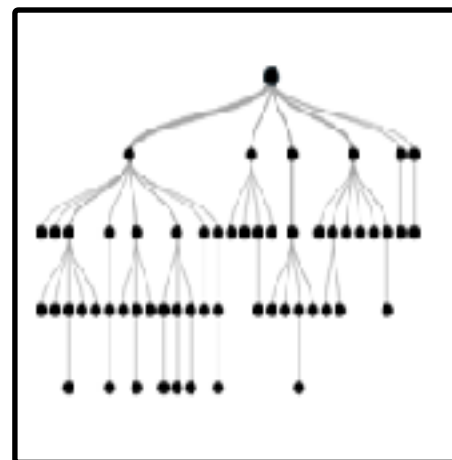
Reference
Screenshot



Test
Screenshot



CW-SSIM EMD RRC OCR
↓ ↓ ↓ ↓
C4.5 Decision Tree Classifier



Visual
Inconsistency

CPI
Report



Complex-Wavelet
Structural Similarity Index
(CW-SSIM)

Earth Mover Distance
of Color Histogram
(EMD)

Relative Ratio
Change
(RRC)

Optical Character
Recognition
(OCR)

Empirical Evaluation

Research Questions:

RQ1: Can DIFFDROID detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

RQ2: What is the cost of running DIFFDROID?

RQ3: Are there similarities among devices exhibiting CPIs?

Empirical Evaluation

Research Questions:

RQ1: Can DIFFDROID detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

RQ2: What is the cost of running DIFFDROID?

RQ3: Are there similarities among devices exhibiting CPIs?

Empirical Evaluation

Research Questions:

RQ1: Can DIFFDROID detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

RQ2: What is the cost of running DIFFDROID?

RQ3: Are there similarities among devices exhibiting CPIs?

Empirical Evaluation

Research Questions:

RQ1: Can DIFFDROID detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

RQ2: What is the cost of running DIFFDROID?

RQ3: Are there similarities among devices exhibiting CPIs?

Empirical Evaluation

Research Questions:

RQ1: Can DIFFDROID detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

RQ2: What is the cost of running DIFFDROID?

RQ3: Are there similarities among devices exhibiting CPIs?

Benchmarks and Setup

<i>ID</i>	<i>Name</i>	<i>Category</i>	<i>Version</i>	<i>LOC (#K)</i>
A1	 BuildmLearn	Education	2.5.0	23.6
A2	 Daily Dozen	Health	10.3	6.3
A3	 Kitchen Timer	Tools	1.1.6	4.3
A4	 Outlay	Finance	1.1.3	8
A5	 Translation Studio	Books	9.0	51.2

Reference Device

LG G3, Android 22



Test Devices

147

Resolution

720x1280
768x1280
1080x1920
1440x2560
480x800
540x960
480x854

OS Version

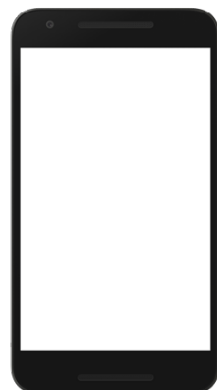
19
21
22
23
24
25

Benchmarks and Setup

<i>ID</i>	<i>Name</i>	<i>Category</i>	<i>Version</i>	<i>LOC (#K)</i>
A1	 BuildmLearn	Education	2.5.0	23.6
A2	 Daily Dozen	Health	10.3	6.3
A3	 Kitchen Timer	Tools	1.1.6	4.3
A4	 Outlay	Finance	1.1.3	8
A5	 Translation Studio	Books	9.0	51.2

Reference Device

LG G3, Android 22



Test Devices

147

Resolution

OS Version

720x1280
768x1280
1080x1920
1440x2560
480x800
540x960
480x854

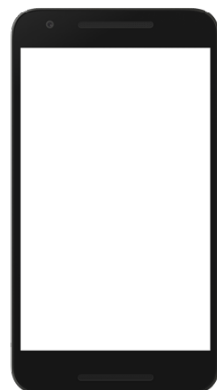
19
21
22
23
24
25

Benchmarks and Setup

<i>ID</i>	<i>Name</i>	<i>Category</i>	<i>Version</i>	<i>LOC (#K)</i>
A1	 BuildmLearn	Education	2.5.0	23.6
A2	 Daily Dozen	Health	10.3	6.3
A3	 Kitchen Timer	Tools	1.1.6	4.3
A4	 Outlay	Finance	1.1.3	8
A5	 Translation Studio	Books	9.0	51.2

Reference Device

LG G3, Android 22



Test Devices

147

Resolution

OS Version

720x1280

19

768x1280

21

1080x1920

22

1440x2560

23

480x800

24

540x960

25

480x854

RQ1

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

<i>ID</i>	<i>Test Device</i>	<i>Window Model</i>	<i>Structural CPI</i>	<i>Functional CPI</i>	<i>Version CPI</i>	<i>Cosmetic CPI</i>	<i>False Positive</i>
A1	135	19	0	2	7	14	1
A2	138	22	2	0	0	22	4
A3	129	13	2	3	0	2	1
A4	125	14	0	1	0	17	2
A5	136	17	2	3	0	19	8
			6	9	7	74	16

RQ1

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

<i>ID</i>	<i>Test Device</i>	<i>Window Model</i>	<i>Structural CPI</i>	<i>Functional CPI</i>	<i>Version CPI</i>	<i>Cosmetic CPI</i>	<i>False Positive</i>
A1	135	19	0	2	7	14	1
A2	138	22	2	0	0	22	4
A3	129	13	2	3	0	2	1
A4	125	14	0	1	0	17	2
A5	136	17	2	3	0	19	8
			6	9	7	74	16

RQ1

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

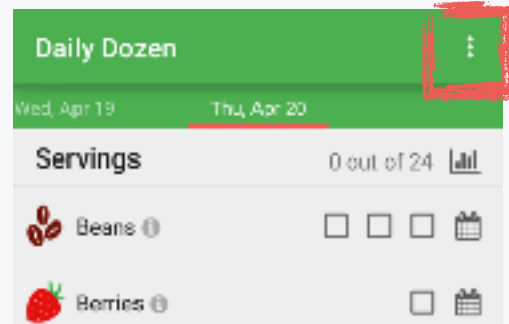
<i>ID</i>	<i>Test Device</i>	<i>Window Model</i>	<i>Structural CPI</i>	<i>Functional CPI</i>	<i>Version CPI</i>	<i>Cosmetic CPI</i>	<i>False Positive</i>
A1	135	19	0	2	7	14	1
A2	138	22	2	0	0	22	4
A3	129	13	2	3	0	2	1
A4	125	14	0	1	0	17	2
A5	136	17	2	3	0	19	8
			6	9	7	74	16

RQ1

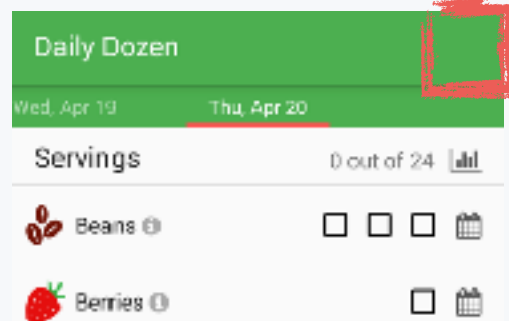
RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

Example

Reference Device



Test Device



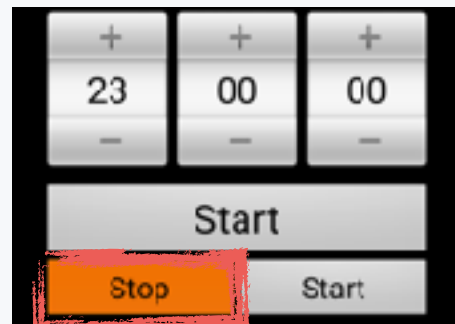
<i>Structural CPI</i>	<i>Functional CPI</i>	<i>Version CPI</i>	<i>Cosmetic CPI</i>	<i>False Positive</i>
0	2	7	14	1
2	0	0	22	4
2	3	0	2	1
0	1	0	17	2
2	3	0	19	8
6	9	7	74	16

RQ1

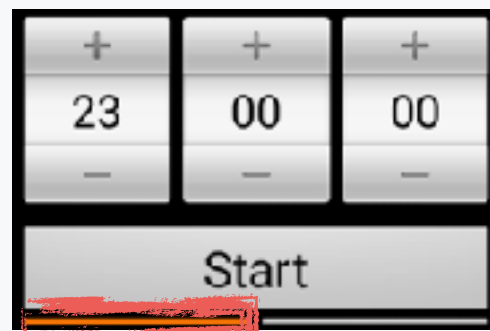
RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

Example

Reference Device



Test Device



<i>Structural CPI</i>	<i>Functional CPI</i>	<i>Version CPI</i>	<i>Cosmetic CPI</i>	<i>False Positive</i>
0	2	7	14	1
2	0	0	22	4
2	3	0	2	1
0	1	0	17	2
2	3	0	19	8
6	9	7	74	16

RQ1

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

Example

Reference Device



Test Device



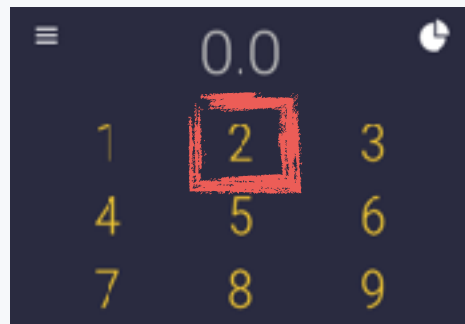
<i>Structural CPI</i>	<i>Functional CPI</i>	<i>Version CPI</i>	<i>Cosmetic CPI</i>	<i>False Positive</i>
0	2	7	14	1
2	0	0	22	4
2	3	0	2	1
0	1	0	17	2
2	3	0	19	8
6	9	7	74	16

RQ1

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

Example

Reference Device



Test Device



<i>Structural CPI</i>	<i>Functional CPI</i>	<i>Version CPI</i>	<i>Cosmetic CPI</i>	<i>False Positive</i>
0	2	7	14	1
2	0	0	22	4
2	3	0	2	1
0	1	0	17	2
2	3	0	19	8
6	9	7	74	16

RQ1

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

Time Scale				Day
Thu				Month
				Year

9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

Time Scale				Day
Thu				Month
				Year

9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

RQ1

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

Time Scale				Day
Thu				Month
				Year

9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

Time Scale				Day
Thu				Month
				Year

9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

RQ1

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

Time Scale				Day
Thu				Month
				Year

9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

Time Scale				Day
Thu				Month
				Year

9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

Evaluation Summary

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

DIFFDROID can detect CPIs in mobile applications while reporting a limited number of false positives

RQ2: What is the cost of running DiffDroid?

DIFFDROID can run overnight for the cases considered

RQ3: Are there similarities among devices exhibiting CPIs?

Devices that are more problematic have low values for resolution and density. However, considering testing devices solely based on resolution and density would have not allowed us to identify all the inconsistencies reported.

Evaluation Summary

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

DIFFDROID can detect CPIs in mobile applications while reporting a limited number of false positives

RQ2: What is the cost of running DiffDroid?

DIFFDROID can run overnight for the cases considered

RQ3: Are there similarities among devices exhibiting CPIs?

Devices that are more problematic have low values for resolution and density. However, considering testing devices solely based on resolution and density would have not allowed us to identify all the inconsistencies reported.

Evaluation Summary

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

DIFFDROID can detect CPIs in mobile applications while reporting a limited number of false positives

RQ2: What is the cost of running DiffDroid?

DIFFDROID can run overnight for the cases considered

RQ3: Are there similarities among devices exhibiting CPIs?

Devices that are more problematic have low values for resolution and density. However, considering testing devices solely based on resolution and density would have not allowed us to identify all the inconsistencies reported.

Evaluation Summary

RQ1: Can DiffDroid detect cross-platform inconsistencies in mobile applications while reporting a limited number of false positives?

DIFFDROID can detect CPIs in mobile applications while reporting a limited number of false positives

RQ2: What is the cost of running DiffDroid?

DIFFDROID can run overnight for the cases considered

RQ3: Are there similarities among devices exhibiting CPIs?

Devices that are more problematic have low values for resolution and density. However, considering testing devices solely based on resolution and density would have not allowed us to identify all the inconsistencies reported.

Future Work

Additional user study

API based differential analysis

Multi-class classifier approach

CPIs repair technique

Future Work

Additional user study

API based differential analysis

Multi-class classifier approach

CPIs repair technique

Future Work

Additional user study

API based differential analysis

Multi-class classifier approach

CPIs repair technique

Future Work

Additional user study

API based differential analysis

Multi-class classifier approach

CPIs repair technique

Future Work

Additional user study

API based differential analysis

Multi-class classifier approach

CPIs repair technique

Research

Test
Generation

Barista

Differential
Testing

DiffDroid

Bug Report
Analysis

Yakusu

Bug Reports

Arkonos commented on Jul 5, 2017

FC upon selecting weekly backup from

Steps to reproduce:

Settings -> Backup -> Backup frequency
100% for me, no other frequency settings
Once after such crash, the app would not
reproduce this.
Pretty sure the same thing happened

Version:

Phone: N

Android

LineageOS

Kernel:3.

afwang commented on

I found this crash on v
[redreader_crash_log_f](#)

Reproduction steps:

1. Launch redreader
2. Tap the refresh button in the app bar
3. Swipe down while the subscribed si

I think it might help to reproduce if you'r
speed is throttled.



QuantumBadger added the

Trolldemorted commented on May 19, 2

I have:

open and closed issues for
<https://github.com/WhisperSys>

Description

es.

untime(30920): Shutting down
untime(30920): FATAL EXCEPT
untime(30920): Process: org
untime(30920): java.lang.Il
untime(30920): at org

Tweet opens with selected account

Actual behavior

Twidere crashes

Steps to reproduce

Longpress retweet icon underneath t



Motivating Example

Bug Report

Onko commented on Mar 24, 2017

<Text paragraph>

<Logcat information>

Steps to reproduce the behavior

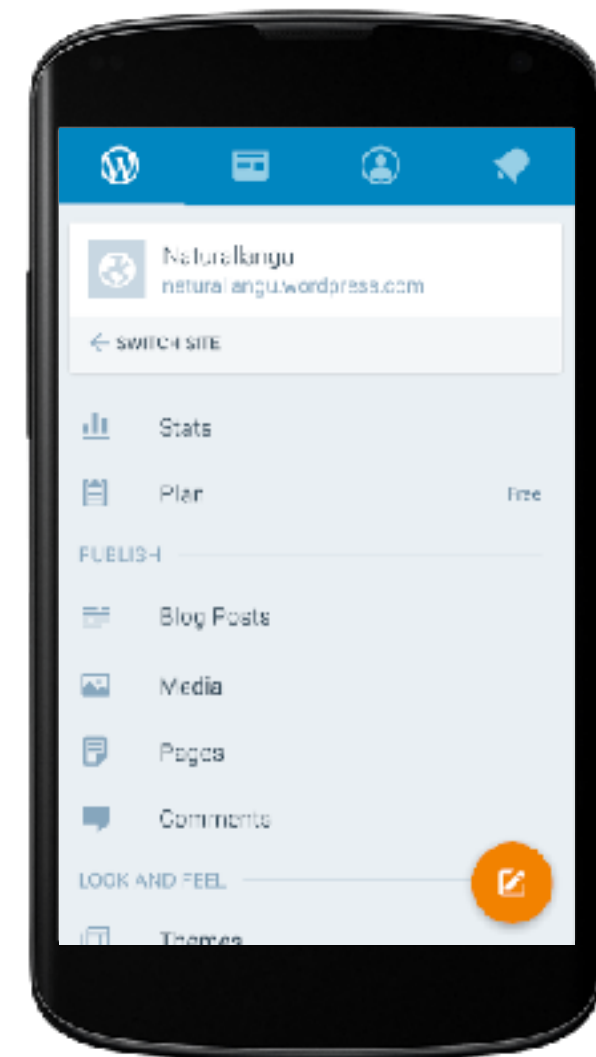
1. Start a new post
2. Type something
3. Tap on the Publish button
4. The app crashes

<Text paragraph>

Developer



WORDPRESS App



Motivating Example

Bug Report

Onko commented on Mar 24, 2017

<Text paragraph>

<Logcat information>

Steps to reproduce the behavior

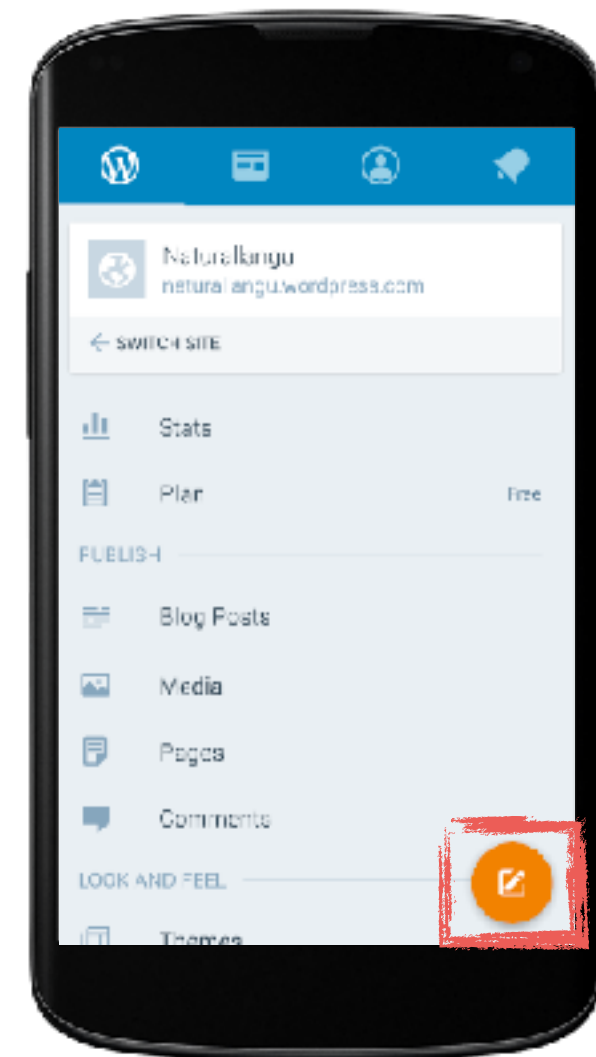
1. Start a new post
2. Type something
3. Tap on the Publish button
4. The app crashes

<Text paragraph>

Developer



WORDPRESS App



Motivating Example

Bug Report

Onko commented on Mar 24, 2017

<Text paragraph>

<Logcat information>

Steps to reproduce the behavior

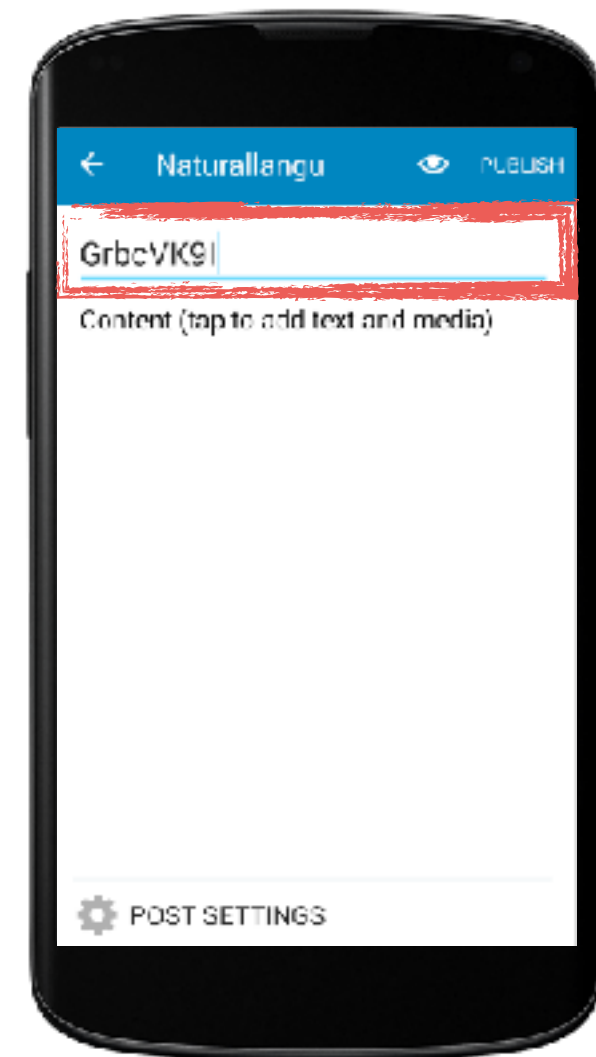
1. Start a new post
2. Type something
3. Tap on the Publish button
4. The app crashes

<Text paragraph>

Developer



WORDPRESS App



Motivating Example

Bug Report

Onko commented on Mar 24, 2017

<Text paragraph>

<Logcat information>

Steps to reproduce the behavior

1. Start a new post
2. Type something
3. Tap on the Publish button
4. The app crashes

<Text paragraph>

Developer



WORDPRESS App



Motivating Example

Bug Report

Onko commented on Mar 24, 2017

<Text paragraph>

<Logcat information>

Steps to reproduce the behavior

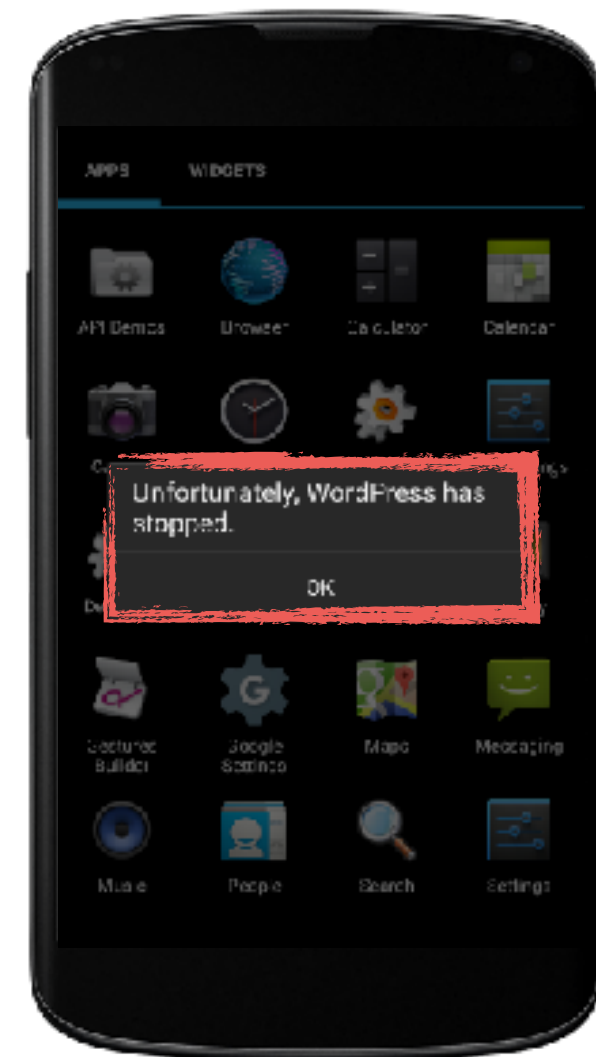
1. Start a new post
2. Type something
3. Tap on the Publish button
4. The app crashes

<Text paragraph>

Developer

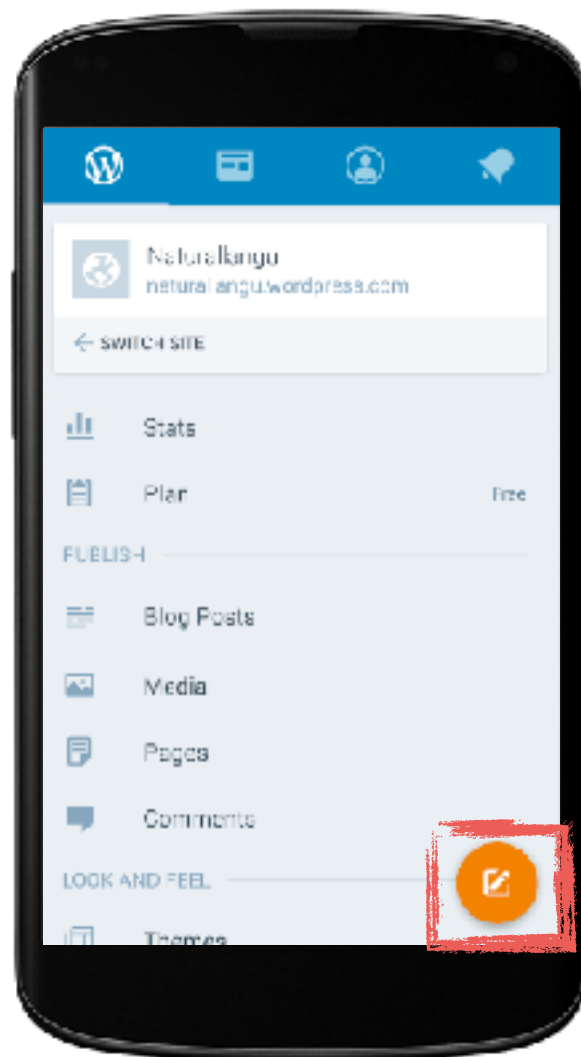


WORDPRESS App



Motivating Example

WORDPRESS App



Developer

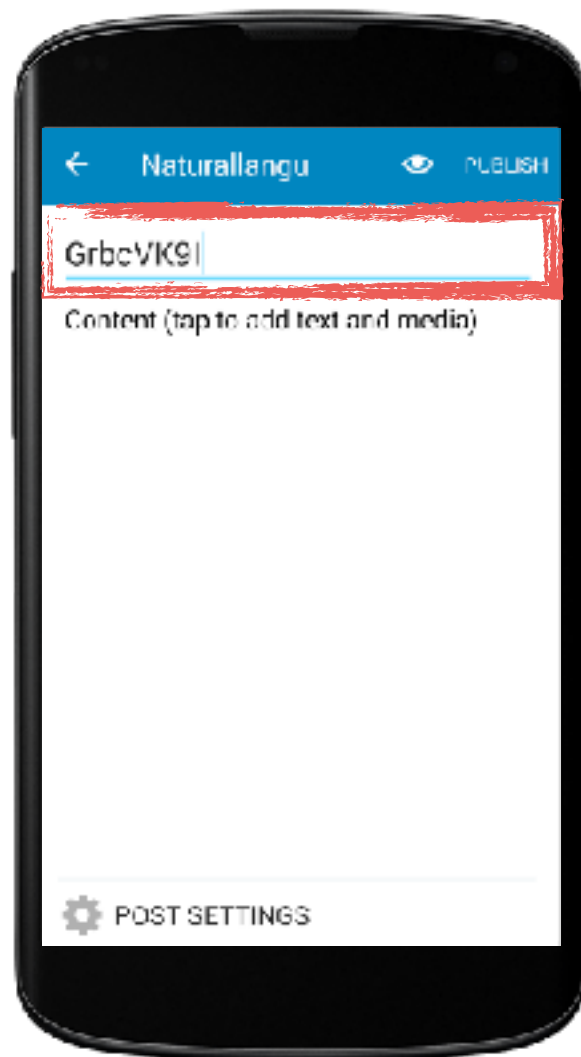


Test Case

```
public void test() {  
    //start a new post  
    onView(  
        withId(R.id.fab_button))  
        .perform(click());  
  
    //type something  
    onView(  
        withId(R.id.post_title))  
        .perform(typeText("GrbcVK9I"));  
  
    //tap on the publish button  
    onView(  
        withId(R.id.menu_save_post))  
        .perform(click());  
}
```

Motivating Example

WORDPRESS App



Developer

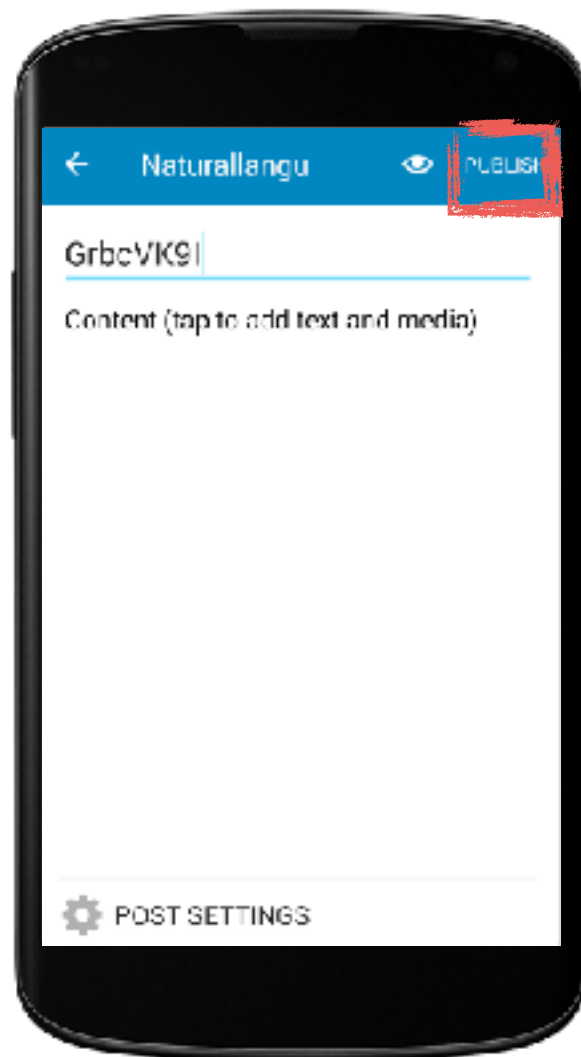


Test Case

```
public void test() {  
  
    //start a new post  
    onView(  
        withId(R.id.fab_button))  
        .perform(click());  
  
    //type something  
    onView(  
        withId(R.id.post_title))  
        .perform(typeText("GrbcVK9I"));  
  
    //tap on the publish button  
    onView(  
        withId(R.id.menu_save_post))  
        .perform(click());  
}
```

Motivating Example

WORDPRESS App



Developer

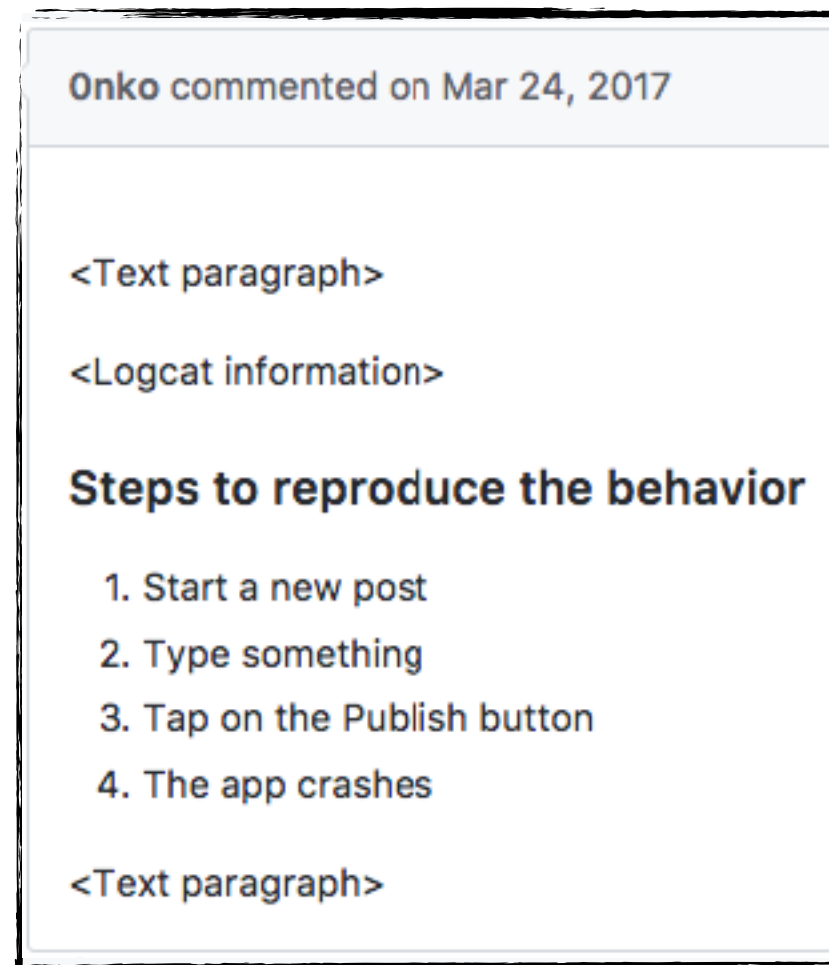


Test Case

```
public void test() {  
  
    //start a new post  
    onView(  
        withId(R.id.fab_button))  
        .perform(click());  
  
    //type something  
    onView(  
        withId(R.id.post_title))  
        .perform(typeText("GrbcVK9I"));  
  
    //tap on the publish button  
    onView(  
        withId(R.id.menu_save_post))  
        .perform(click());  
}
```

Motivating Example

Bug Report



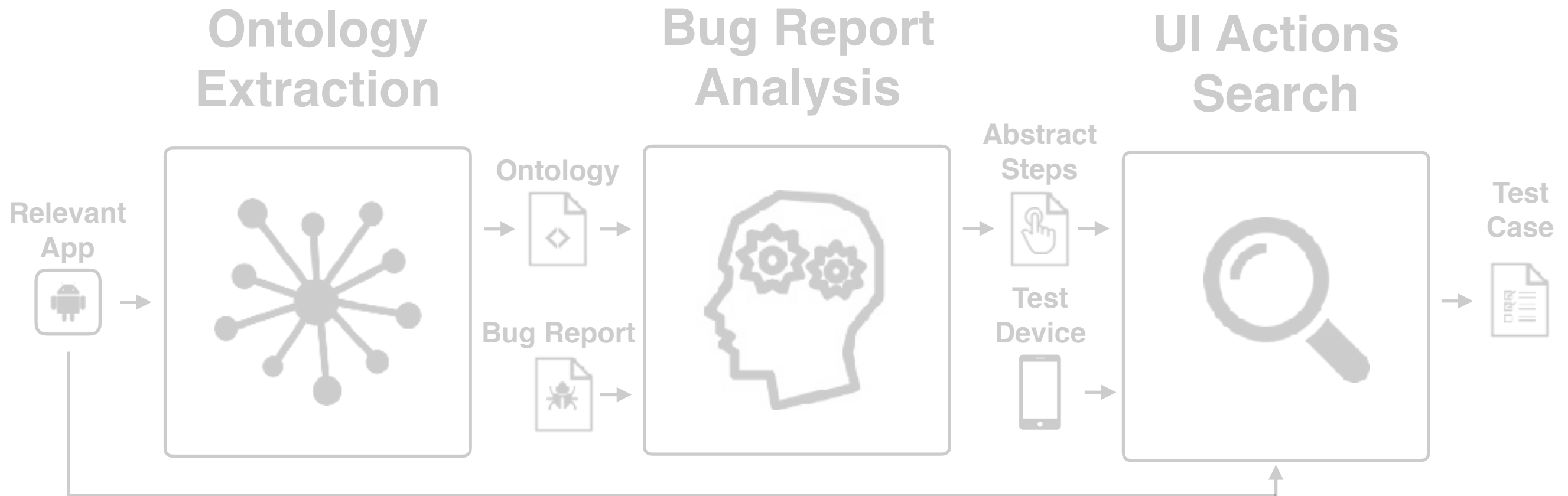
Yakusu



Test Case

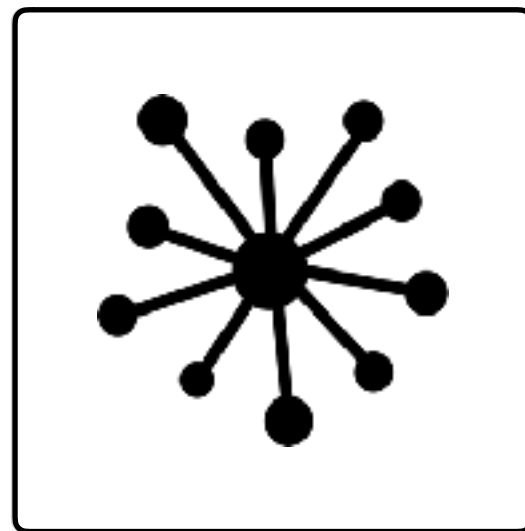
```
public void test() {  
  
    //start a new post  
    onView(  
        withId(R.id.fab_button))  
        .perform(click());  
  
    //type something  
    onView(  
        withId(R.id.post_title))  
        .perform(typeText("GrbcVK9I"));  
  
    //tap on the publish button  
    onView(  
        withId(R.id.menu_save_post))  
        .perform(click());  
}
```

Yakusu Overview



Yakusu Overview

Ontology Extraction



Ontology



Bug Report



Bug Report Analysis



Abstract Steps



Test Device



UI Actions Search



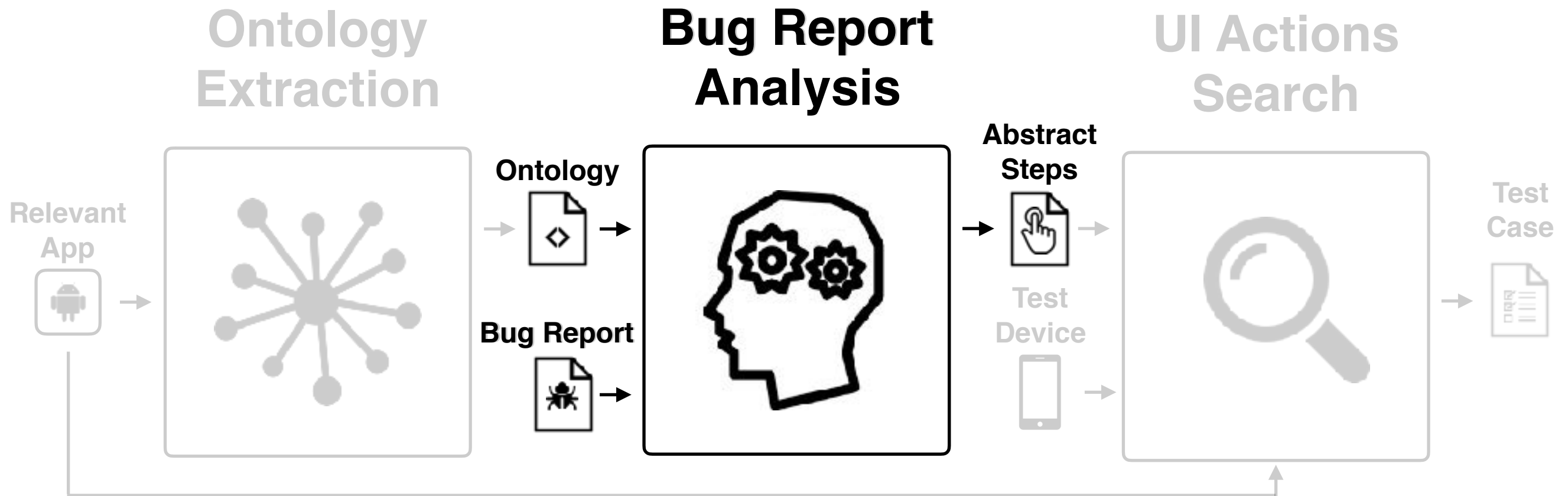
Test Case



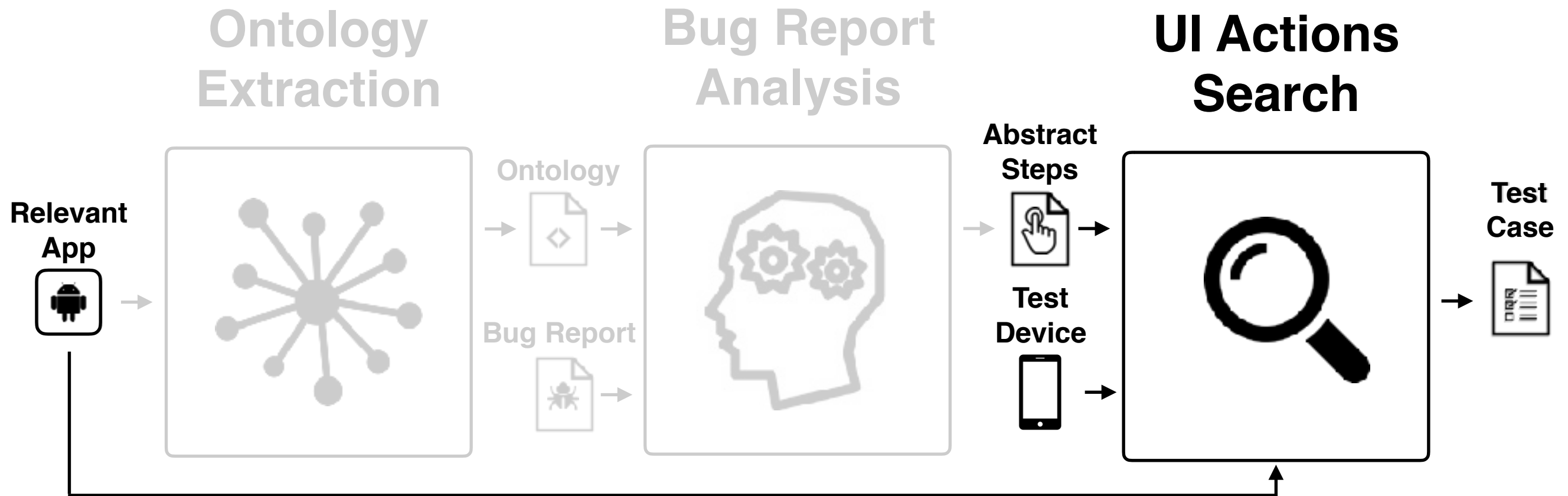
Relevant App



Yakusu Overview



Yakusu Overview



Ontology Extraction

Relevant App



Ontology



UI Element
Properties

< label , icon , identifier >

Example



< Publish , , menu save post >

Ontology Extraction

Relevant App



Ontology



UI Element Properties

< label , icon , identifier >

Example



< Publish , , menu save post >

Ontology Extraction

Relevant App



Ontology



UI Element Properties

< label , icon , identifier >

Example



< Publish , , menu save post >

Bug Report Analysis

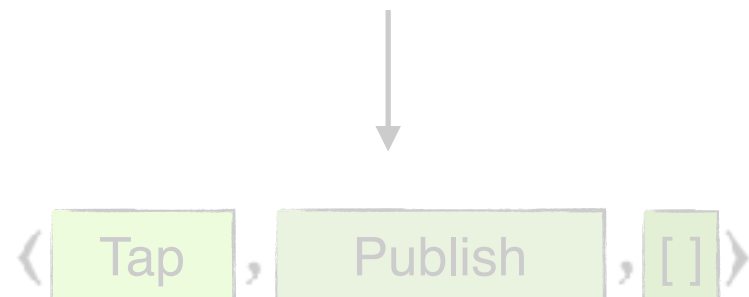
Bug Report Abstract Steps



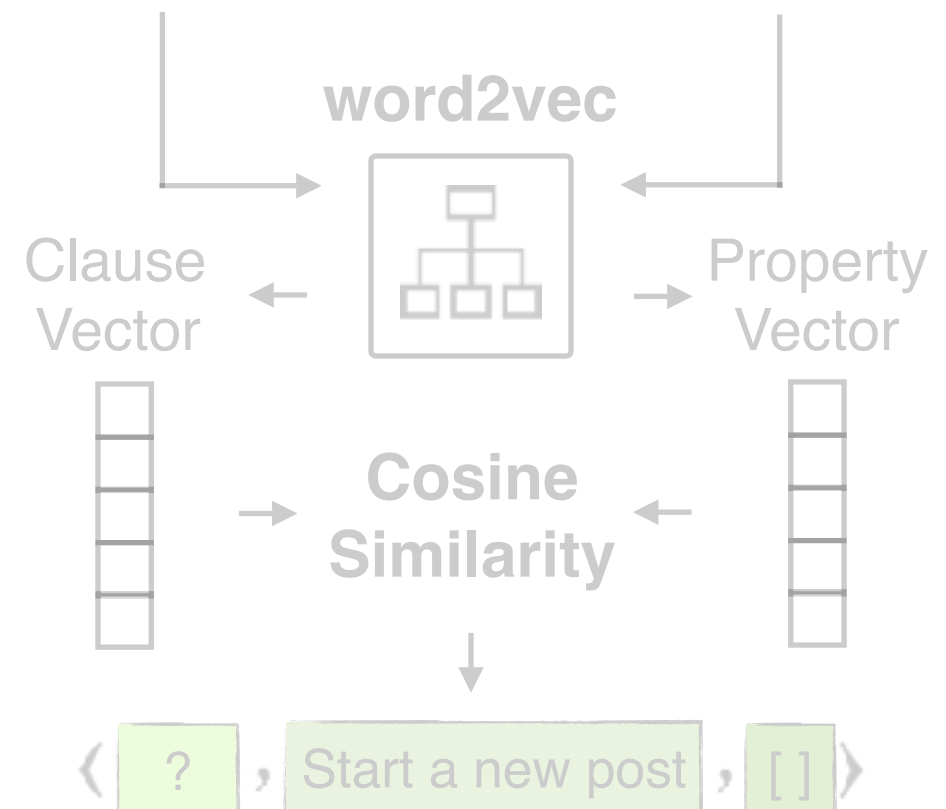
Abstract Step

< action , target , properties >

Dependency Tree (Case #1)



Dependency Tree (Case #2)



Bug Report Analysis

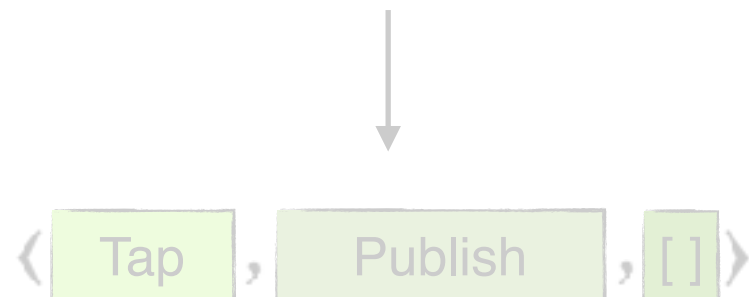
Bug Report Abstract Steps



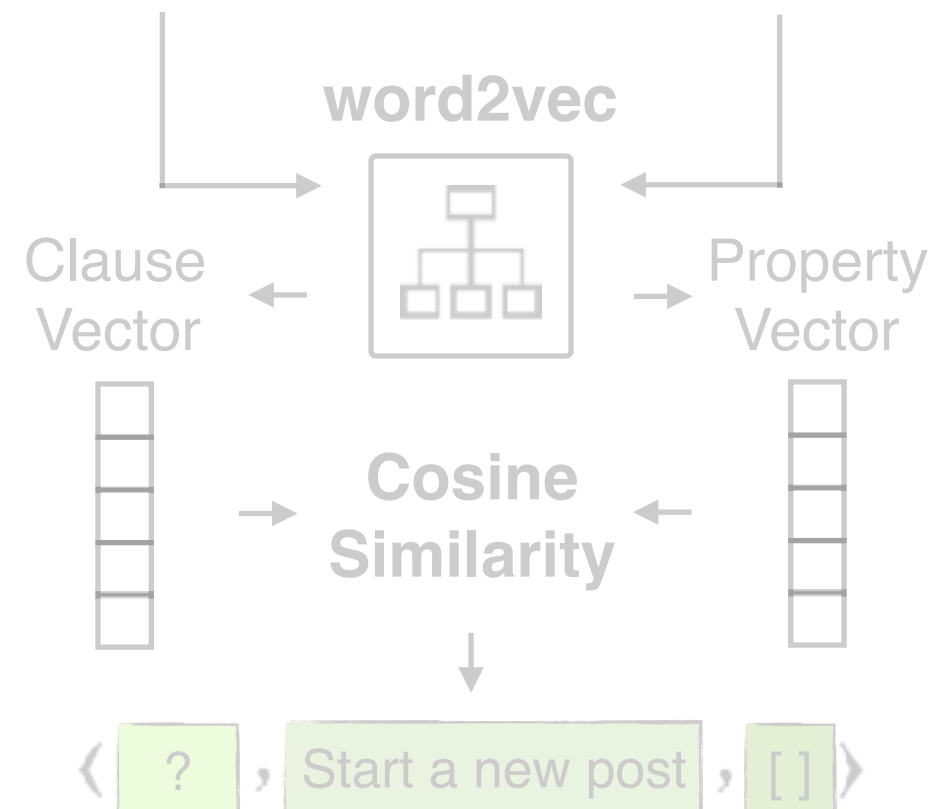
Abstract Step

< action , target , properties >

Dependency Tree (Case #1)

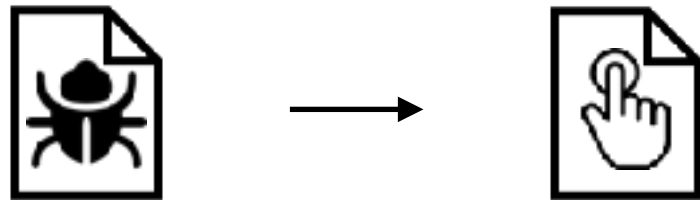


Dependency Tree (Case #2)



Bug Report Analysis

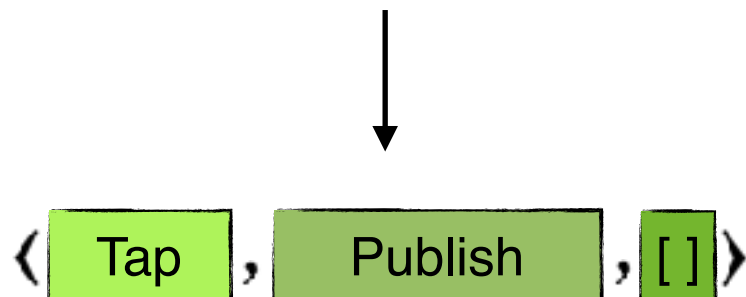
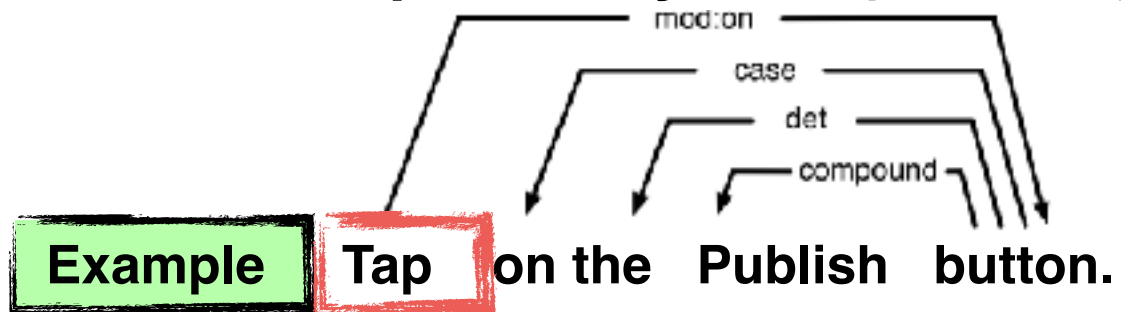
Bug Report Abstract Steps



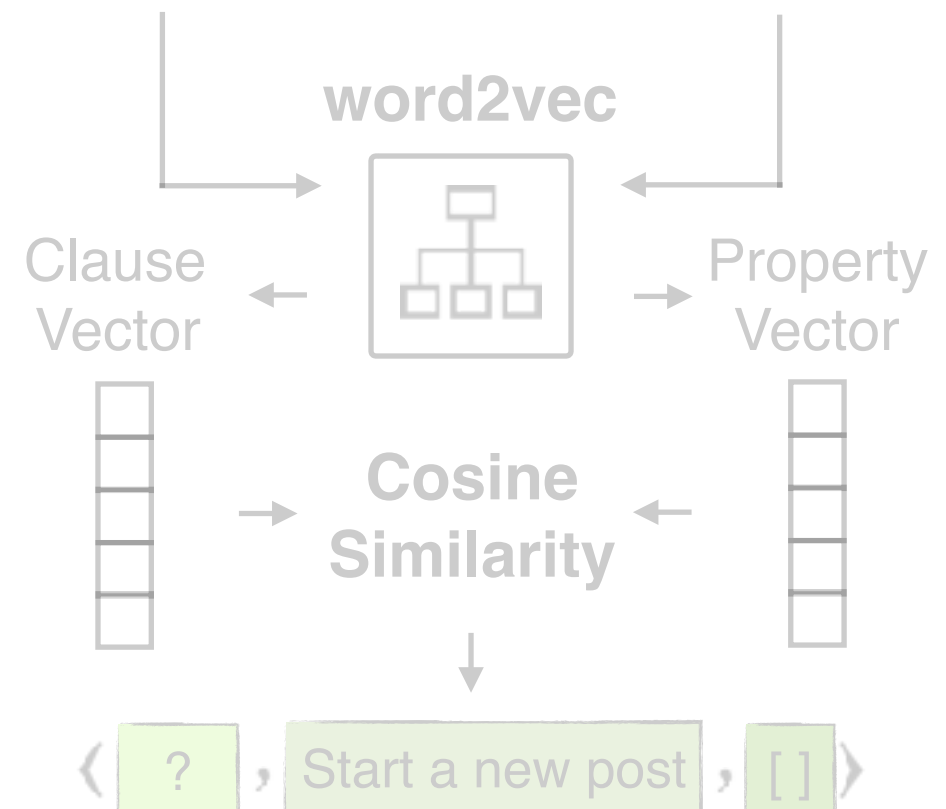
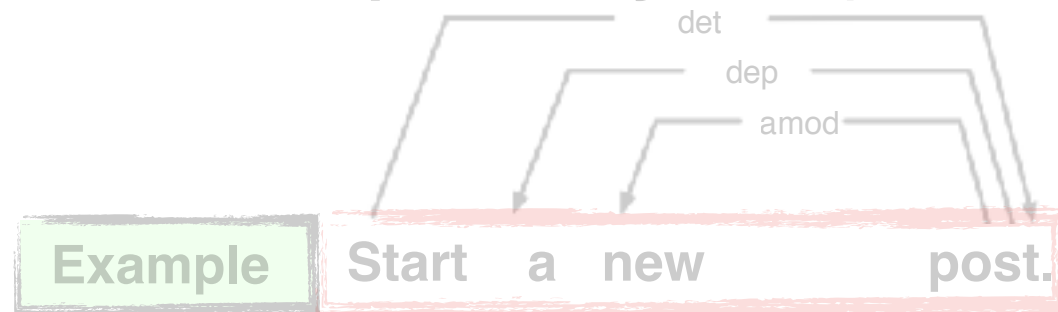
Abstract Step



Dependency Tree (Case #1)



Dependency Tree (Case #2)



Bug Report Analysis

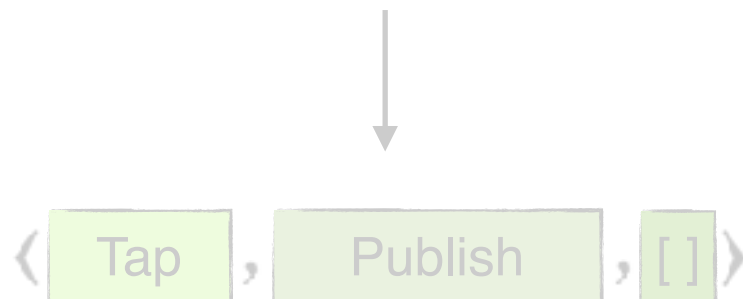
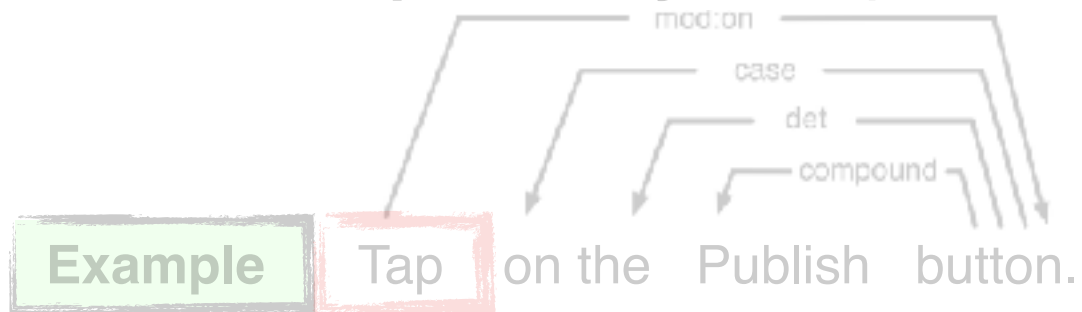
Bug Report Abstract Steps



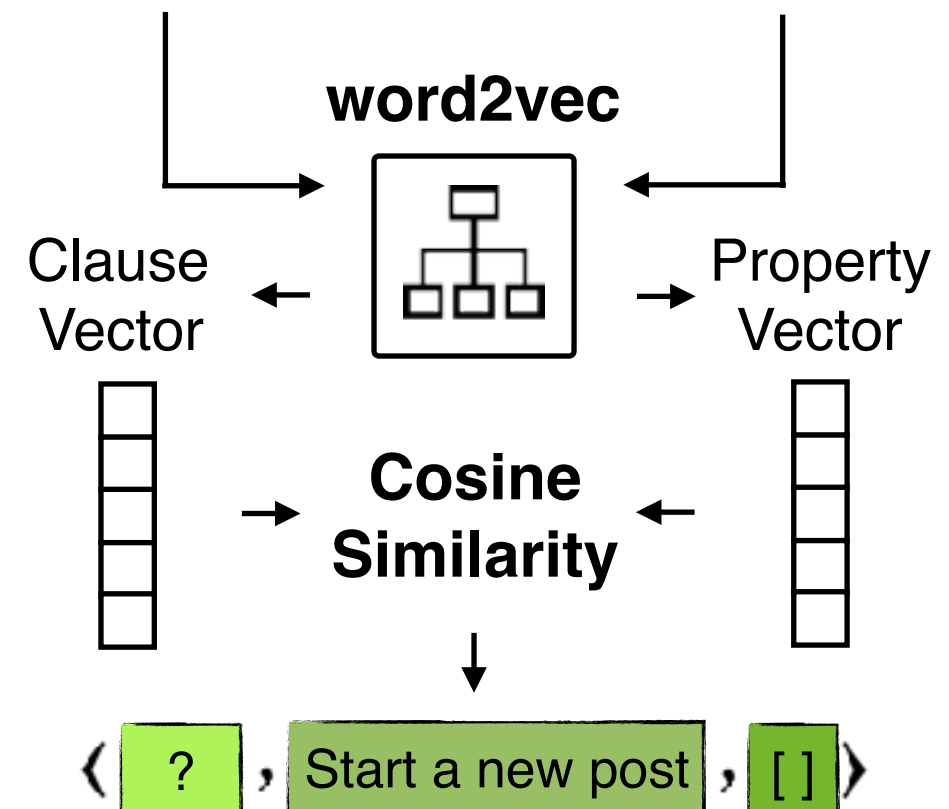
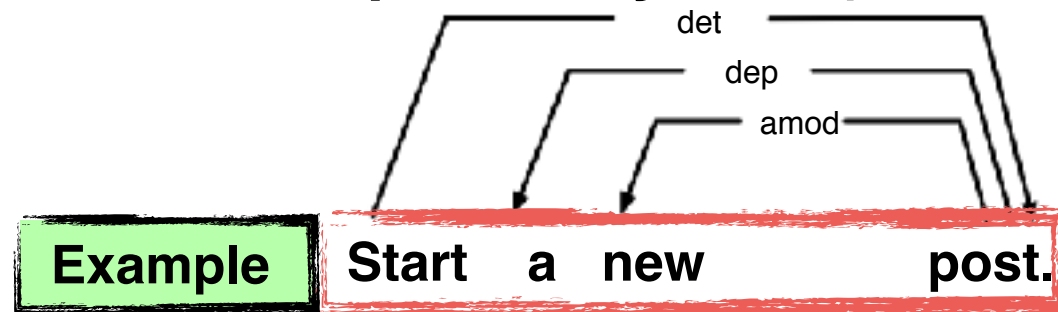
Abstract Step

< action , target , properties >

Dependency Tree (Case #1)



Dependency Tree (Case #2)



UI Actions Search

Abstract Steps



Test Case

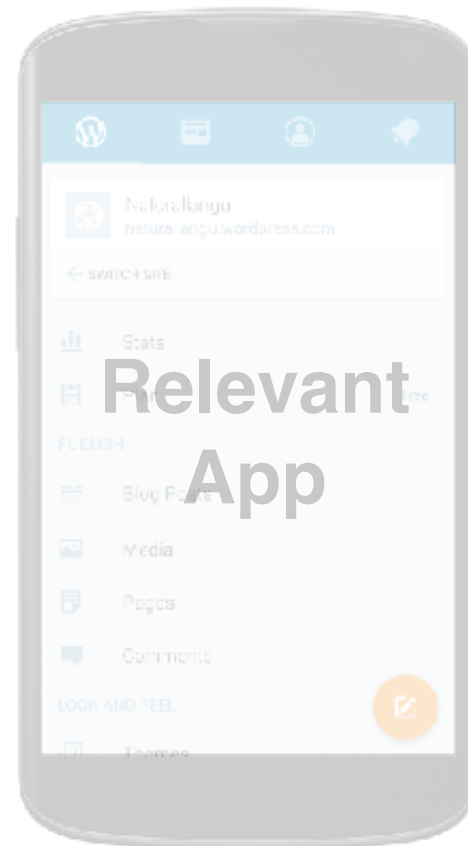
Abstract Step



UI Action

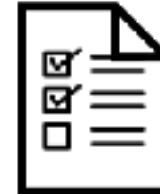


Test Device



UI Actions Search

Abstract Steps



Test Case

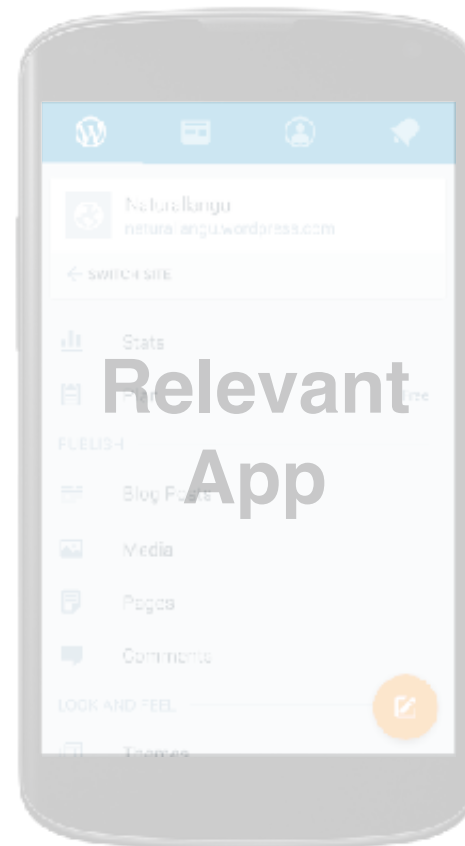
Abstract Step



UI Action



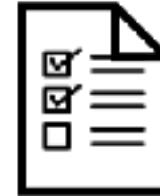
Test Device



Relevant App

UI Actions Search

Abstract
Steps



Test Case

Abstract Step

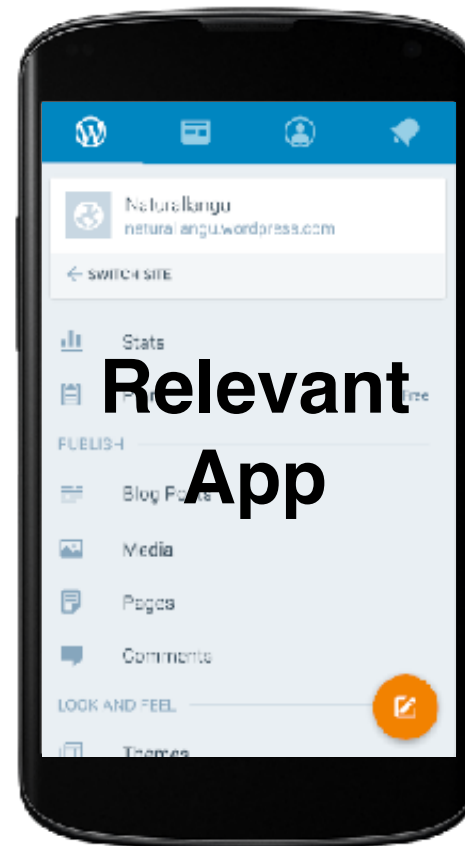
< action , target , properties >



UI Action

< UI action , UI element , properties >

Test Device



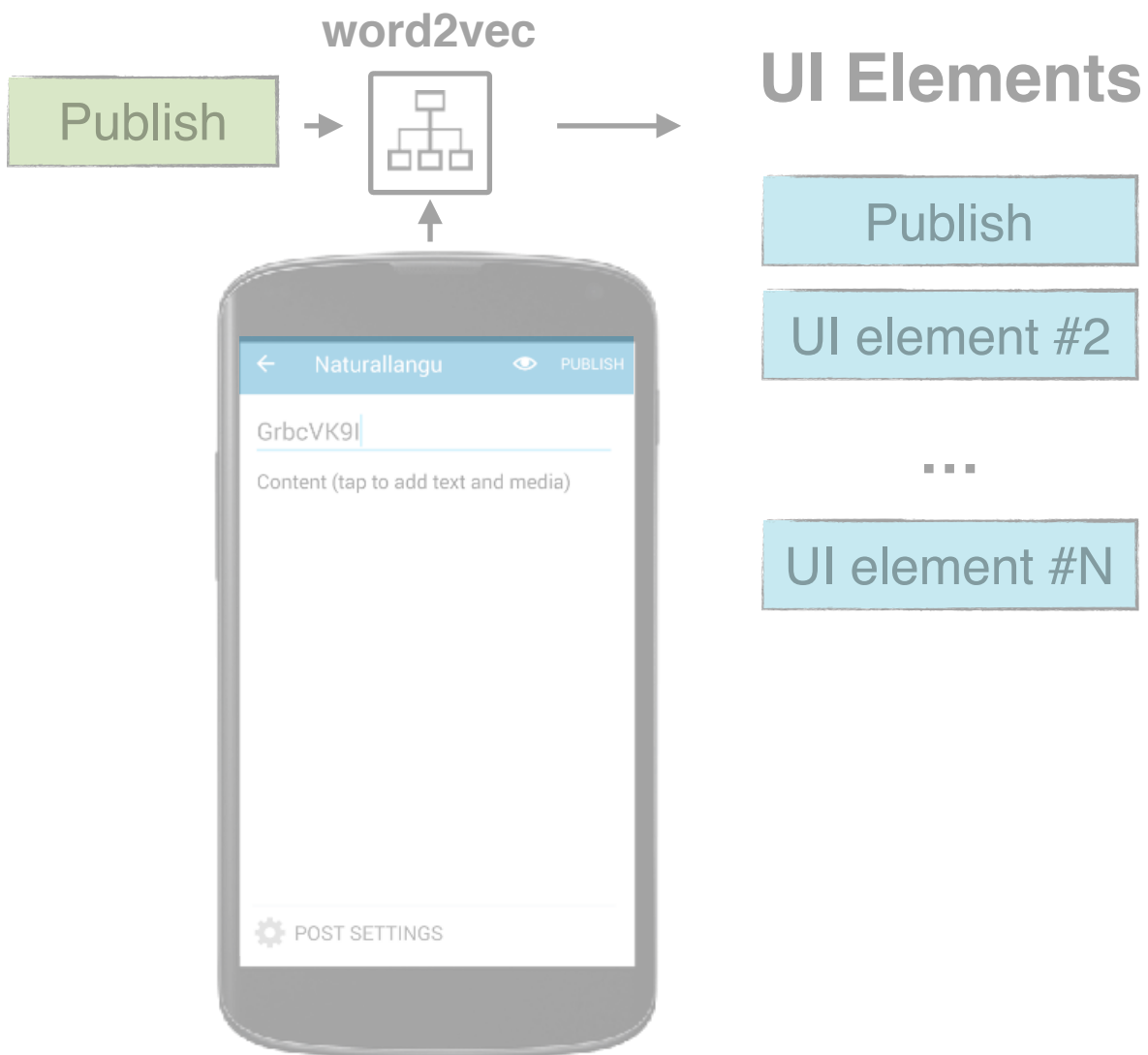
Relevant
App

UI Actions Search

Abstract Step



Find UI Element



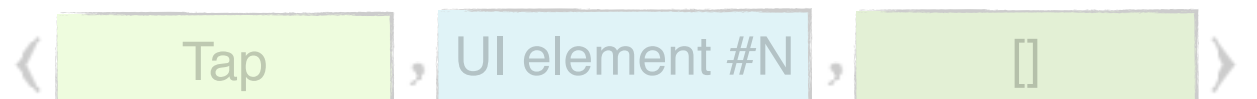
Current Execution



Forked Executions



...



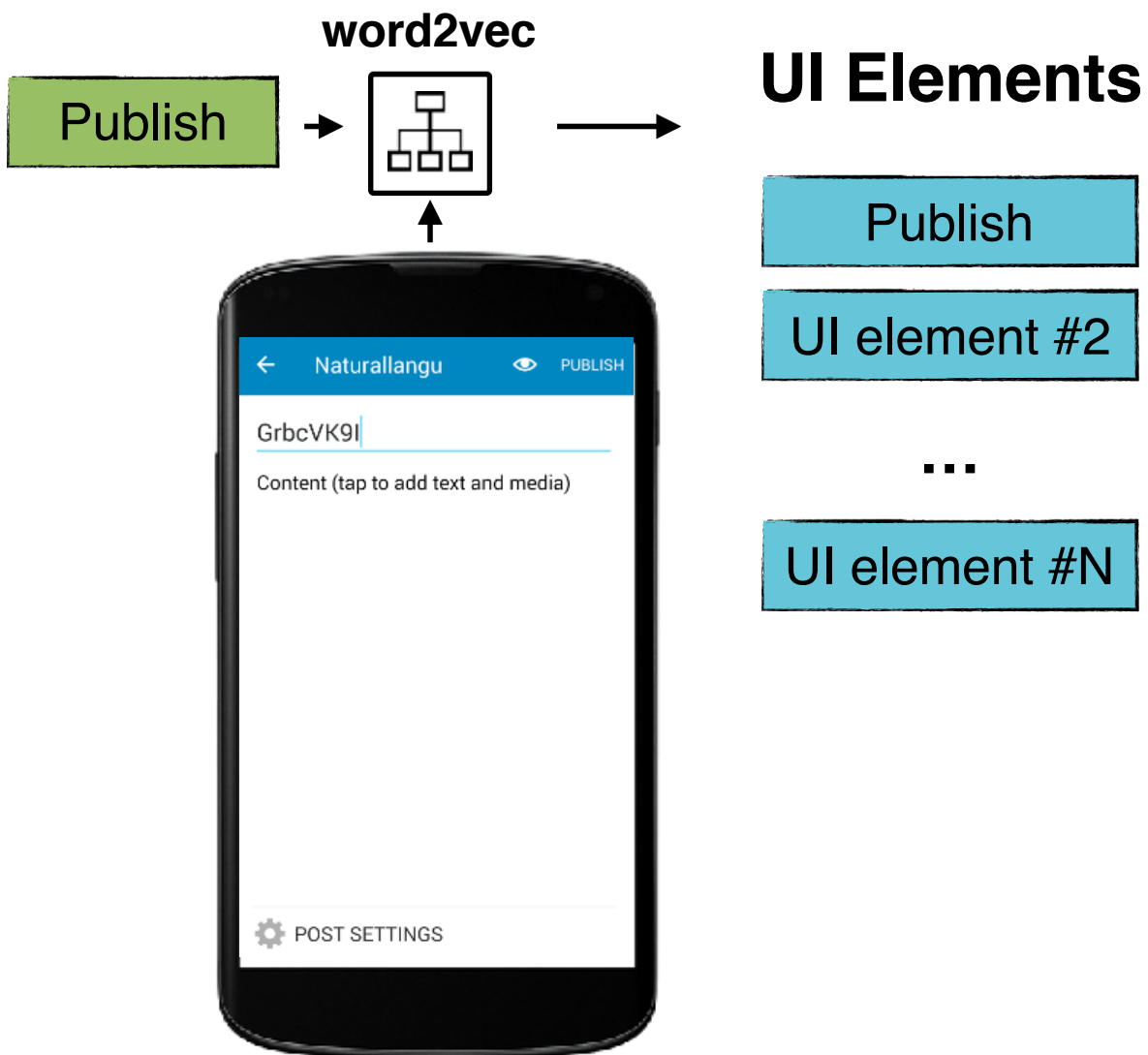
Random UI Action Generation

UI Actions Search

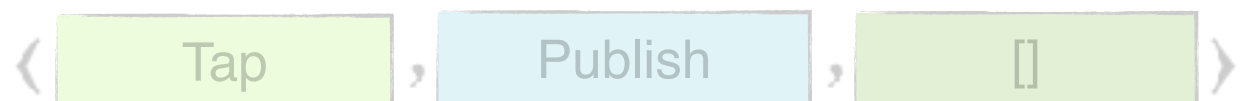
Abstract Step



Find UI Element



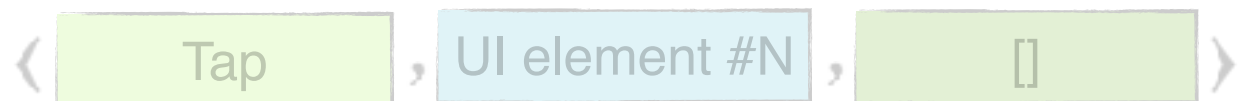
Current Execution



Forked Executions



...



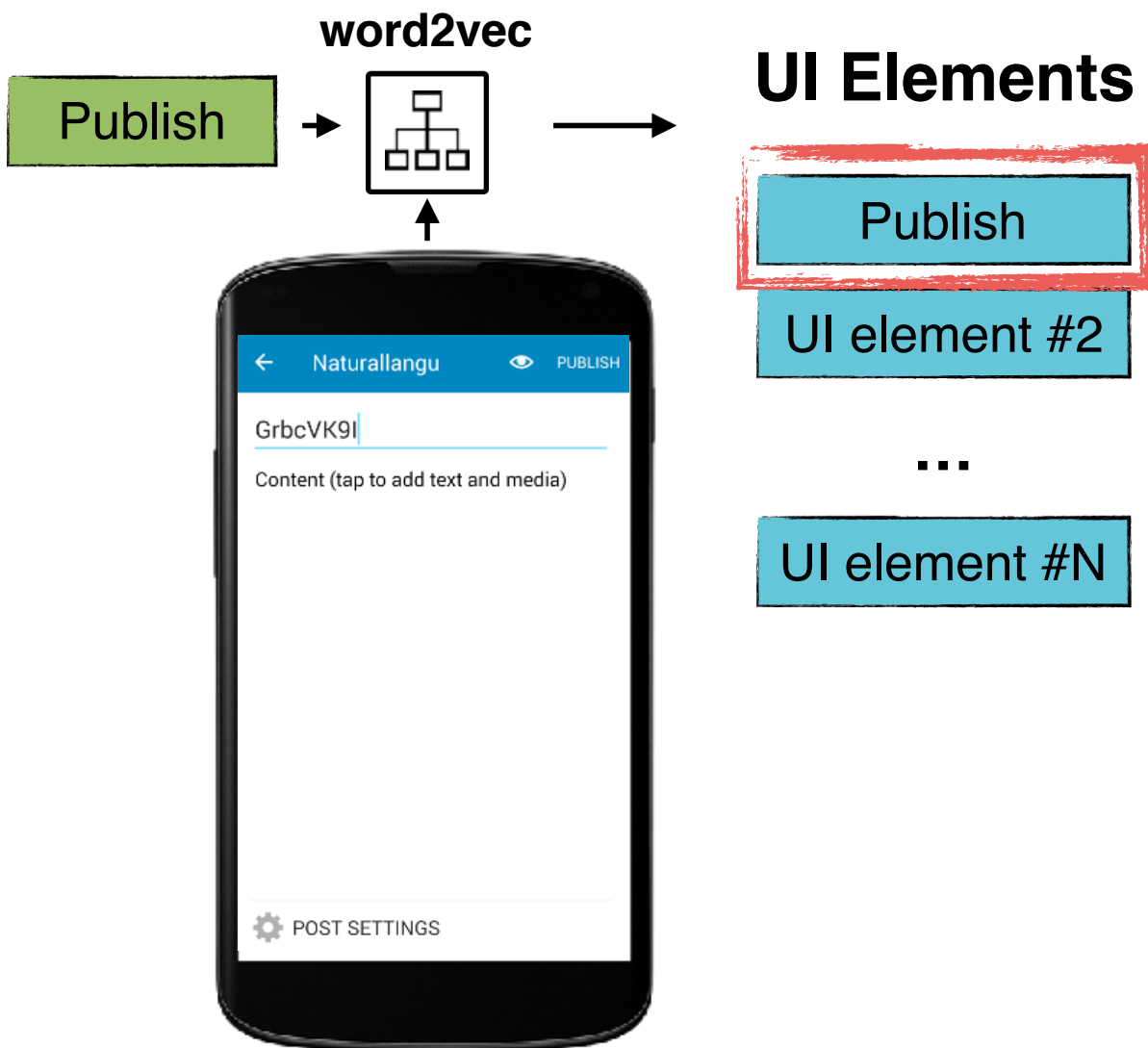
Random UI Action Generation

UI Actions Search

Abstract Step



Find UI Element



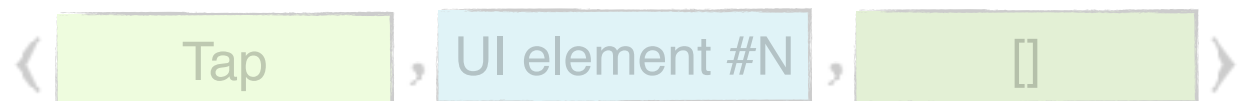
Current Execution



Forked Executions



...



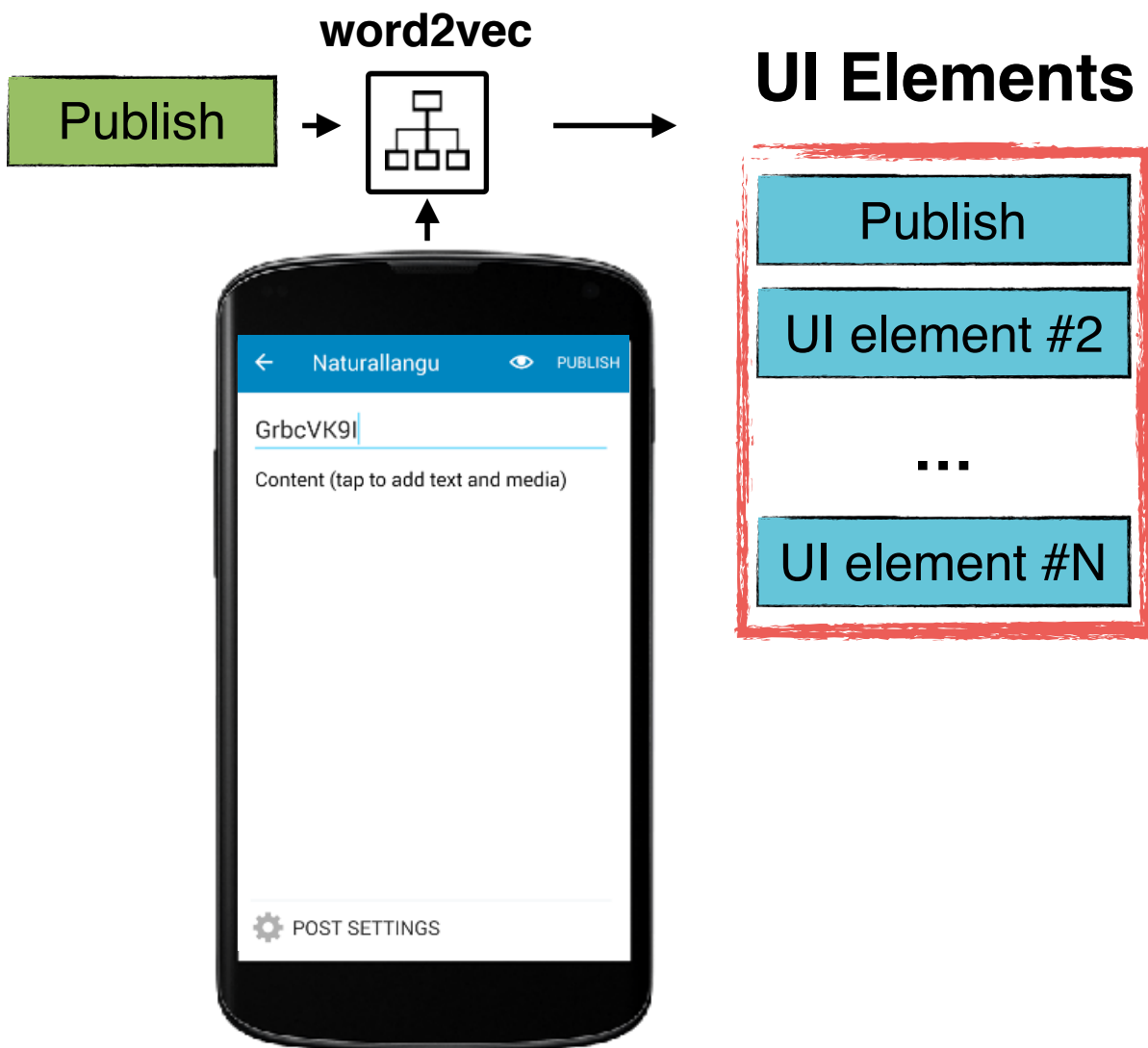
Random UI Action Generation

UI Actions Search

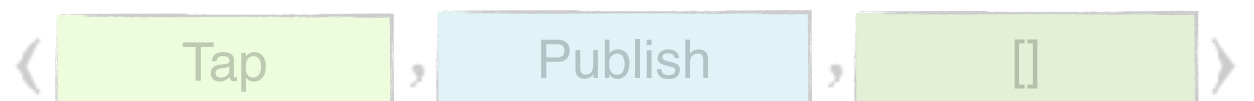
Abstract Step



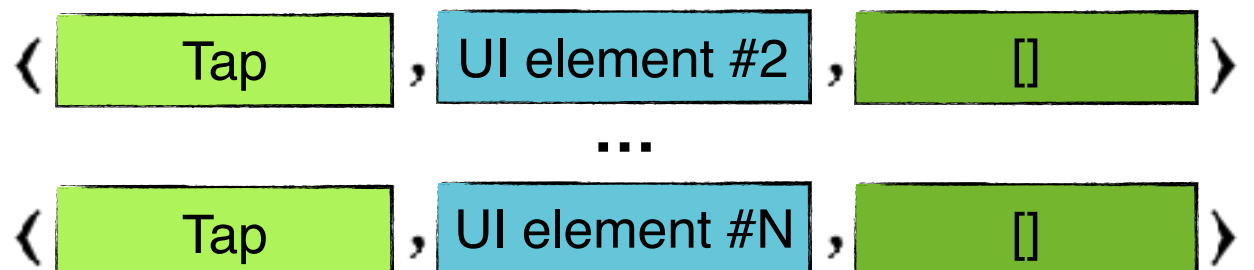
Find UI Element



Current Execution



Forked Executions



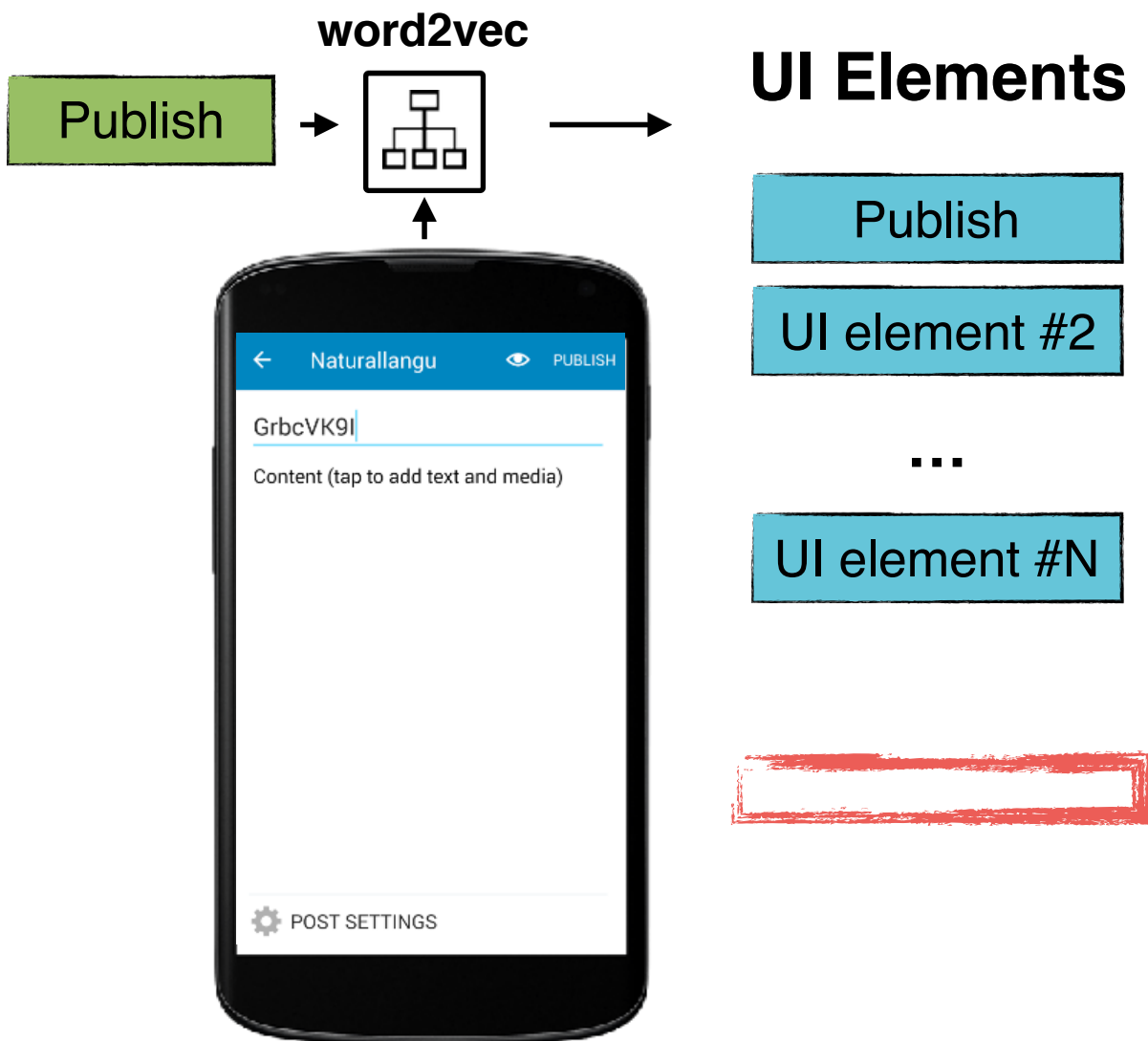
Random UI Action Generation

UI Actions Search

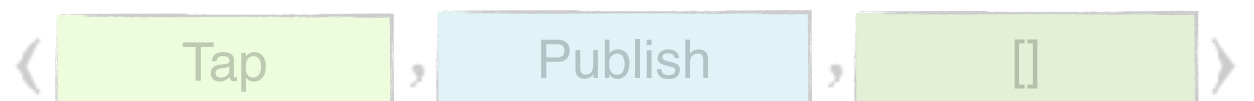
Abstract Step



Find UI Element



Current Execution



Forked Executions



Random UI Action Generation

UI Actions Search

UI Actions

“Start a new post”



“Type something”



“Tap on the publish button”



Test Case

```
public void test() {  
  
    //start a new post  
    onView(  
        withId(R.id.fab_button))  
        .perform(click());  
  
    //type something  
    onView(  
        withId(R.id.post_title))  
        .perform(typeText("GrbcVK9I"));  
  
    //tap on the publish button  
    onView(  
        withId(R.id.menu_save_post))  
        .perform(click());  
}
```

Empirical Evaluation

Research Questions:

RQ1: Can YAKUSU translate bug reports written in natural language into executable test cases?

RQ2: What is the cost of running YAKUSU?

Empirical Evaluation

Research Questions:

RQ1: Can YAKUSU translate bug reports written in natural language into executable test cases?

RQ2: What is the cost of running YAKUSU?

Empirical Evaluation

Research Questions:

RQ1: Can YAKUSU translate bug reports written in natural language into executable test cases?

RQ2: What is the cost of running YAKUSU?

Benchmarks and Setup

GitHub



Bug Report



x 100

Could not
Build



x 9

Could not
Reproduce Manually



x 29

Available
Reports



x 62

Successfully
Generated Tests



x 37

Benchmarks and Setup

GitHub



Bug Report



x 100

**Could not
Build**



x 9

**Could not
Reproduce Manually**



x 29

Available
Reports



x 62

Successfully
Generated Tests



x 37

Benchmarks and Setup

GitHub



Bug Report



x 100

**Could not
Build**



x 9

**Could not
Reproduce Manually**



x 29

**Available
Reports**



x 62

**Successfully
Generated Tests**



x 37

Benchmarks and Setup

GitHub



Bug Report



x 100

**Could not
Build**



x 9

**Could not
Reproduce Manually**



x 29

**Available
Reports**



x 62

**Successfully
Generated Tests**



x 37

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-
05	SIGNAL	1	-	1	-	2	1	-	-	1	-
08	REDREADER	2	-	2	2	5	1	-	-	4	2
14	SILENCE	1	-	1	-	2	1	-	-	1	-
23	K-9 MAIL	3	-	3	-	8	1	-	-	3	-
25	NEXTCLOUD	1	2	1	-	1	1	1	1	3	-
27	BUTTERKNIFE	1	-	1	-	3	1	-	-	1	-
35	ODK COLLECT	2	-	1	4	11	3	-	-	2	-
39	PIX-ART MESSENGER	3	-	3	-	2	1	-	-	3	-
40	YALP STORE	5	-	4	1	11	1	-	-	5	-
50	OCREADER	3	-	3	-	8	2	-	-	3	-
51	WORDPRESS	3	-	2	1	2	1	-	-	3	-
52	SIGNAL	2	1	2	-	3	1	-	4	6	3
57	OPEN EVENT	1	1	1	1	3	2	1	-	2	-
68	TAGMO	1	1	1	1	10	1	-	3	4	2
69	ANKIDROID	5	-	5	-	8	1	-	-	5	-
73	K-9 MAIL	2	-	2	-	6	3	-	-	2	-
74	CLUTTR	2	-	2	-	6	1	-	-	2	-
78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-
05	SIGNAL	1	-	1	-	2	1	-	-	1	-
08	REDREADER	2	-	2	2	5	1	-	-	4	2
14	SILENCE	1	-	1	-	2	1	-	-	1	-
23	K-9 MAIL	3	-	3	-	8	1	-	-	3	-
25	NEXTCLOUD	1	2	1	-	1	1	1	1	3	-
27	BUTTERKNIFE	1	-	1	-	3	1	-	-	1	-
35	ODK COLLECT	2	-	1	4	11	3	-	-	2	-
39	PIX-ART MESSENGER	3	-	3	-	2	1	-	-	3	-
40	YALP STORE	5	-	4	1	11	1	-	-	5	-
50	OCREADER	3	-	3	-	8	2	-	-	3	-
51	WORDPRESS	3	-	2	1	2	1	-	-	3	-
52	SIGNAL	2	1	2	-	3	1	-	4	6	3
57	OPEN EVENT	1	1	1	1	3	2	1	-	2	-
68	TAGMO	1	1	1	1	10	1	-	3	4	2
69	ANKIDROID	5	-	5	-	8	1	-	-	5	-
73	K-9 MAIL	2	-	2	-	6	3	-	-	2	-
74	CLUTTR	2	-	2	-	6	1	-	-	2	-
78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-
05	SIGNAL	1	-	1	-	2	1	-	-	1	-
08	REDREADER	2	-	2	2	5	1	-	-	4	2
14	SILENCE	1	-	1	-	2	1	-	-	1	-
23	K-9 MAIL	3	-	3	-	8	1	-	-	3	-
25	NEXTCLOUD	1	2	1	-	1	1	1	1	3	-
27	BUTTERKNIFE	1	-	1	-	3	1	-	-	1	-
35	ODK COLLECT	2	-	1	4	11	3	-	-	2	-
39	PIX-ART MESSENGER	3	-	3	-	2	1	-	-	3	-
40	YALP STORE	5	-	4	1	11	1	-	-	5	-
50	OCREADER	3	-	3	-	8	2	-	-	3	-
51	WORDPRESS	3	-	2	1	2	1	-	-	3	-
52	SIGNAL	2	1	2	-	3	1	-	4	6	3
57	OPEN EVENT	1	1	1	1	3	2	1	-	2	-
68	TAGMO	1	1	1	1	10	1	-	3	4	2
69	ANKIDROID	5	-	5	-	8	1	-	-	5	-
73	K-9 MAIL	2	-	2	-	6	3	-	-	2	-
74	CLUTTR	2	-	2	-	6	1	-	-	2	-
78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-
05	SIGNAL	1	-	1	-	2	1	-	-	1	-
08	REDREADER	2	-	2	2	5	1	-	-	4	2
14	SILENCE	1	-	1	-	2	1	-	-	1	-
23	K-9 MAIL	3	-	3	-	8	1	-	-	3	-
25	NEXTCLOUD	1	2	1	-	1	1	1	1	3	-
27	BUTTERKNIFE	1	-	1	-	3	1	-	-	1	-
35	ODK COLLECT	2	-	1	4	11	3	-	-	2	-
39	PIX-ART MESSENGER	3	-	3	-	2	1	-	-	3	-
40	YALP STORE	5	-	4	1	11	1	-	-	5	-
50	OCREADER	3	-	3	-	8	2	-	-	3	-
51	WORDPRESS	3	-	2	1	2	1	-	-	3	-
52	SIGNAL	2	1	2	-	3	1	-	4	6	3
57	OPEN EVENT	1	1	1	1	3	2	1	-	2	-
68	TAGMO	1	1	1	1	10	1	-	3	4	2
69	ANKIDROID	5	-	5	-	8	1	-	-	5	-
73	K-9 MAIL	2	-	2	-	6	3	-	-	2	-
74	CLUTTR	2	-	2	-	6	1	-	-	2	-
78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-

9 bug reports had at least one implicit action

7 bug reports had at least one step without explicit action

19 bug reports reproduced exploring only one execution

6 bug reports reproduced exploring more than one execution

4 test cases with additional actions

78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-

9 bug reports had at least one implicit action

7 bug reports had at least one step without explicit action

19 bug reports reproduced exploring only one execution

6 bug reports reproduced exploring more than one execution

4 test cases with additional actions

78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-

9 bug reports had at least one implicit action

7 bug reports had at least one step without explicit action

19 bug reports reproduced exploring only one execution

6 bug reports reproduced exploring more than one execution

4 test cases with additional actions

78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-

9 bug reports had at least one implicit action

7 bug reports had at least one step without explicit action

19 bug reports reproduced exploring only one execution

6 bug reports reproduced exploring more than one execution

4 test cases with additional actions

78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-

9 bug reports had at least one implicit action

7 bug reports had at least one step without explicit action

19 bug reports reproduced exploring only one execution

6 bug reports reproduced exploring more than one execution

4 test cases with additional actions

78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-

9 bug reports had at least one implicit action

7 bug reports had at least one step without explicit action

19 bug reports reproduced exploring only one execution

6 bug reports reproduced exploring more than one execution

4 test cases with additional actions

78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ1

ID	Name	Actions		Steps		Search				Tests	
		A_e	A_i	AS	AS_g	S_g	S_p	H	R	TC_s	TC_a
01	TACHIYOMI	4	1	-	4	10	1	1	-	5	-
03	TWIDERE	2	-	2	-	12	1	-	-	2	-

RQ1: Can YAKUSU translate bug reports written in natural language into executable test cases?

We consider the results encouraging as YAKUSU is already able to generate test cases for a good number of bug reports.

78	NEXTCLOUD	2	1	2	-	1	1	1	-	3	-
84	K-9 MAIL	1	-	-	1	2	1	-	-	1	-
95	MIFOSX	2	3	2	-	3	1	1	2	5	-
96	SCREENRECORDER	3	2	3	-	1	1	2	-	5	-
97	NEXTCLOUD	1	1	-	2	4	2	1	-	2	-
99	FLASHCARDS	3	-	2	1	6	2	-	3	4	1

RQ2

ID	Name	Cost		
		T_{oe}	T_{bra}	T_{uias}
01	TACHIYOMI	54s	18s	1m12s
03	TWIDERE	2m34s	15s	1m07s
05	SIGNAL	1m23s	17s	6m57s
08	REDREADER	25s	19s	1m25s
14	SILENCE	43s	15s	5m08s
23	K-9 MAIL	32s	16s	1m00s
25	NEXTCLOUD	34s	16s	2m08s
27	BUTTERKNIFE	13s	16s	09s
35	ODK COLLECT	30s	18s	3m51s
39	PIX-ART MESSENGER	43s	15s	31s
40	YALP STORE	16s	17s	2m25s
50	OCREADER	22s	17s	1m33s
51	WORDPRESS	1m35s	16s	3m18s
52	SIGNAL	1m16s	17s	18m10s
57	OPEN EVENT	36s	20s	2m48s
68	TAGMO	17s	16s	31s
69	ANKIDROID	29s	17s	28m55s
73	K-9 MAIL	35s	18s	3m22s
74	CLUTTR	26s	15s	24s
78	NEXTCLOUD	32s	15s	2m20s
84	K-9 MAIL	32s	17s	45s
95	MIFOSX	39s	18s	1m25s
96	SCREENRECORDER	15s	17s	54s
97	NEXTCLOUD	37s	23s	9m28s
99	FLASHCARDS	15s	18s	58s

RQ2

ID	Name	Cost		
		T_{oe}	T_{bra}	T_{uias}
01	TACHIYOMI	54s	18s	1m12s
03	TWIDERE	2m34s	15s	1m07s
05	SIGNAL	1m23s	17s	6m57s
08	REDREADER	25s	19s	1m25s
14	SILENCE	43s	15s	5m08s

UI Actions search phase takes the longest on average

Average analysis time is 5m00s and median time is 2m58s

Longest analysis time is 29m41s

73	K-9 MAIL	35s	18s	3m22s
74	CLUTTR	26s	15s	24s
78	NEXTCLOUD	32s	15s	2m20s
84	K-9 MAIL	32s	17s	45s
95	MIFOSX	39s	18s	1m25s
96	SCREENRECORDER	15s	17s	54s
97	NEXTCLOUD	37s	23s	9m28s
99	FLASHCARDS	15s	18s	58s

RQ2

ID	Name	Cost		
		T_{oe}	T_{bra}	T_{uias}
01	TACHIYOMI	54s	18s	1m12s
03	TWIDERE	2m34s	15s	1m07s
05	SIGNAL	1m23s	17s	6m57s
08	REDREADER	25s	19s	1m25s
14	SILENCE	43s	15s	5m08s

UI Actions search phase takes the longest on average

Average analysis time is 5m00s and median time is 2m58s

Longest analysis time is 29m41s

73	K-9 MAIL	35s	18s	3m22s
74	CLUTTR	26s	15s	24s
78	NEXTCLOUD	32s	15s	2m20s
84	K-9 MAIL	32s	17s	45s
95	MIFOSX	39s	18s	1m25s
96	SCREENRECORDER	15s	17s	54s
97	NEXTCLOUD	37s	23s	9m28s
99	FLASHCARDS	15s	18s	58s

RQ2

ID	Name	Cost		
		T_{oe}	T_{bra}	T_{uias}
01	TACHIYOMI	54s	18s	1m12s
03	TWIDERE	2m34s	15s	1m07s
05	SIGNAL	1m23s	17s	6m57s
08	REDREADER	25s	19s	1m25s
14	SILENCE	43s	15s	5m08s

UI Actions search phase takes the longest on average

Average analysis time is 5m00s and median time is 2m58s

Longest analysis time is 29m41s

73	K-9 MAIL	35s	18s	3m22s
74	CLUTTR	26s	15s	24s
78	NEXTCLOUD	32s	15s	2m20s
84	K-9 MAIL	32s	17s	45s
95	MIFOSX	39s	18s	1m25s
96	SCREENRECORDER	15s	17s	54s
97	NEXTCLOUD	37s	23s	9m28s
99	FLASHCARDS	15s	18s	58s

RQ2

ID	Name	Cost		
		T_{oe}	T_{bra}	T_{uias}
01	TACHIYOMI	54s	18s	1m12s
03	TWIDERE	2m34s	15s	1m07s
05	SIGNAL	1m23s	17s	6m57s
08	REDREADER	25s	19s	1m25s
14	SILENCE	43s	15s	5m08s

UI Actions search phase takes the longest on average

Average analysis time is 5m00s and median time is 2m58s

Longest analysis time is 29m41s

73	K-9 MAIL	35s	18s	3m22s
74	CLUTTR	26s	15s	24s
78	NEXTCLOUD	32s	15s	2m20s
84	K-9 MAIL	32s	17s	45s
95	MIFOSX	39s	18s	1m25s
96	SCREENRECORDER	15s	17s	54s
97	NEXTCLOUD	37s	23s	9m28s
99	FLASHCARDS	15s	18s	58s

RQ2

ID	Name	Cost		
		T_{oe}	T_{bra}	T_{uias}
01	TACHIYOMI	54s	18s	1m12s
03	TWIDERE	2m34s	15s	1m07s
05	SIGNAL	1m23s	17s	6m57s
08	REDREADER	25s	19s	1m25s
14	SILENCE	43s	15s	5m08s

RQ2: What is the cost of running YAKUSU?

The cost of running YAKUSU is fairly low and YAKUSU can be used to monitor bug reports and generate test cases throughout the day as opposed to overnight only.

73	K-9 MAIL	35s	18s	3m22s
74	CLUTTR	26s	15s	24s
78	NEXTCLOUD	32s	15s	2m20s
84	K-9 MAIL	32s	17s	45s
95	MIFOSX	39s	18s	1m25s
96	SCREENRECORDER	15s	17s	54s
97	NEXTCLOUD	37s	23s	9m28s
99	FLASHCARDS	15s	18s	58s

Future Work

Extend ontology

Learn macro-step associated with the app

Interpret non-crashing failures from bug reports

Generate test cases from written/spoken test specifications

Identify duplicate bug reports

Future Work

Extend ontology

Learn macro-step associated with the app

Interpret non-crashing failures from bug reports

Generate test cases from written/spoken test specifications

Identify duplicate bug reports

Future Work

Extend ontology

Learn macro-step associated with the app

Interpret non-crashing failures from bug reports

Generate test cases from written/spoken test specifications

Identify duplicate bug reports

Future Work

Extend ontology

Learn macro-step associated with the app

Interpret non-crashing failures from bug reports

Generate test cases from written/spoken test specifications

Identify duplicate bug reports

Future Work

Extend ontology

Learn macro-step associated with the app

Interpret non-crashing failures from bug reports

Generate test cases from written/spoken test specifications

Identify duplicate bug reports

Future Work

Extend ontology

Learn macro-step associated with the app

Interpret non-crashing failures from bug reports

Generate test cases from written/spoken test specifications

Identify duplicate bug reports

Opportunities at Georgia Tech

Research Area

Software testing and program analysis

Openings

Postdocs, internships, PhD positions

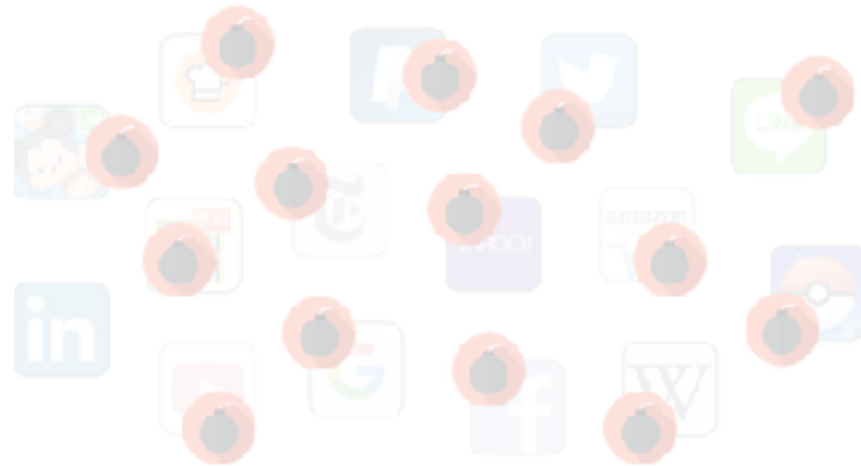
Alessandro Orso



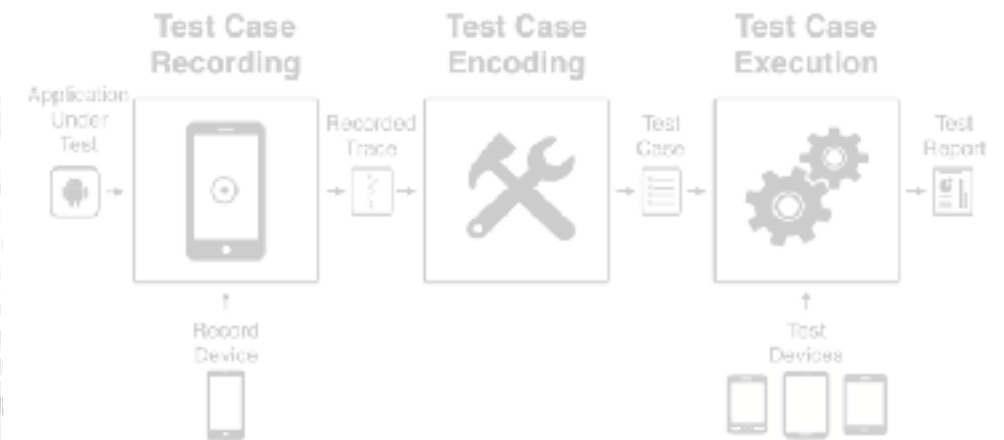
Email:
orso@cc.gatech.edu

Summary

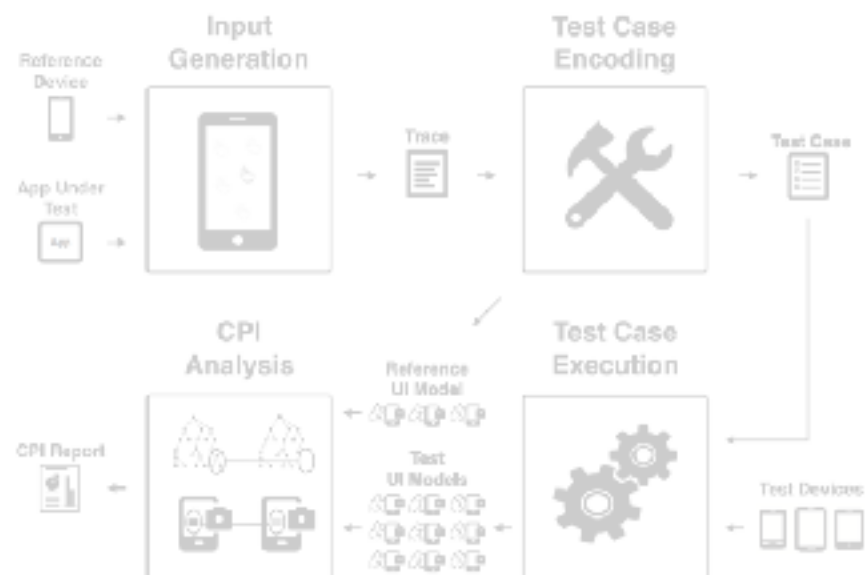
Failures



Barista Overview



DIFFDROID Overview

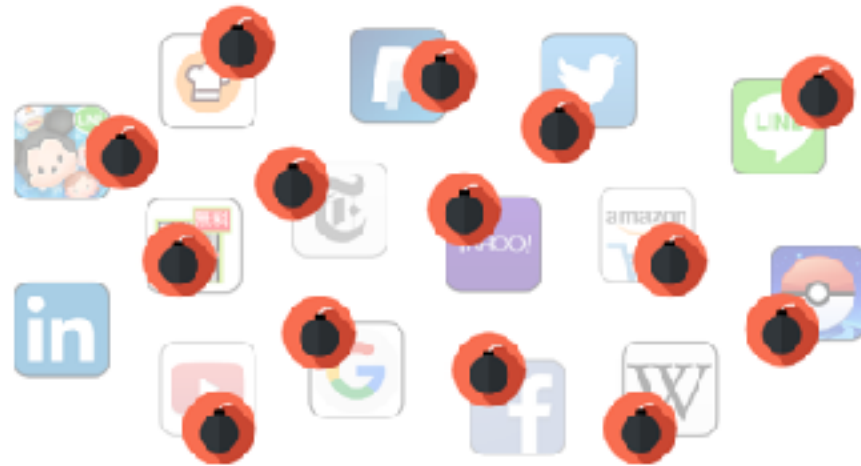


Yakusu Overview

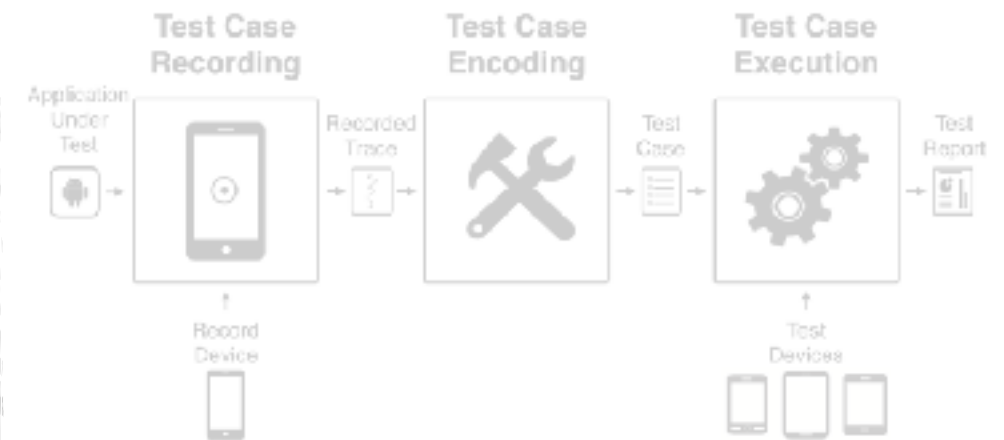


Summary

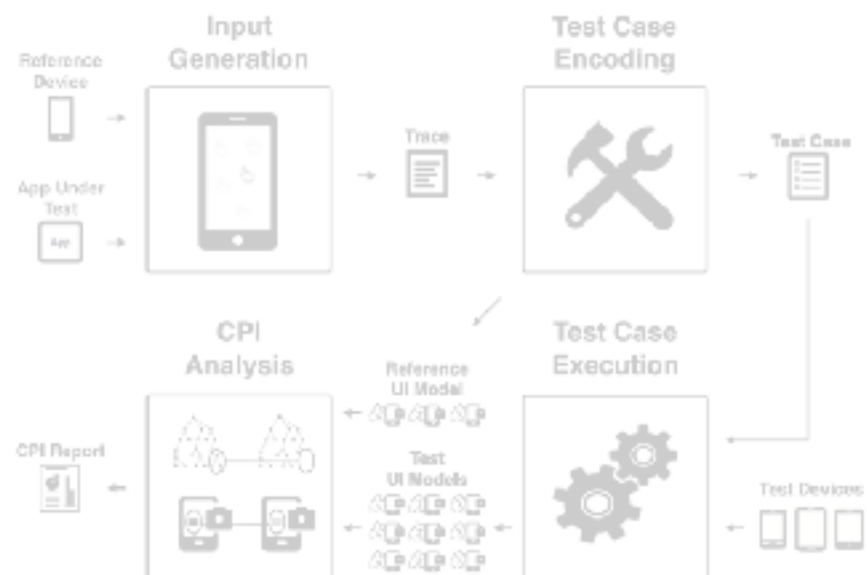
Failures



Barista Overview



DIFFDROID Overview

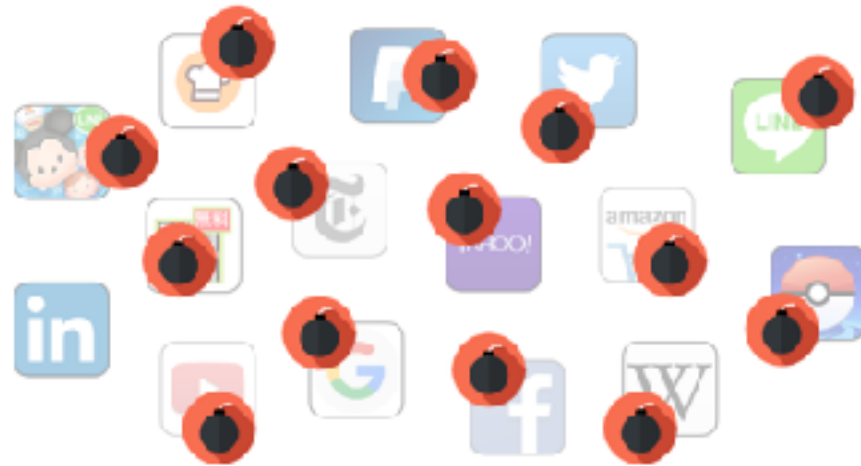


Yakusu Overview

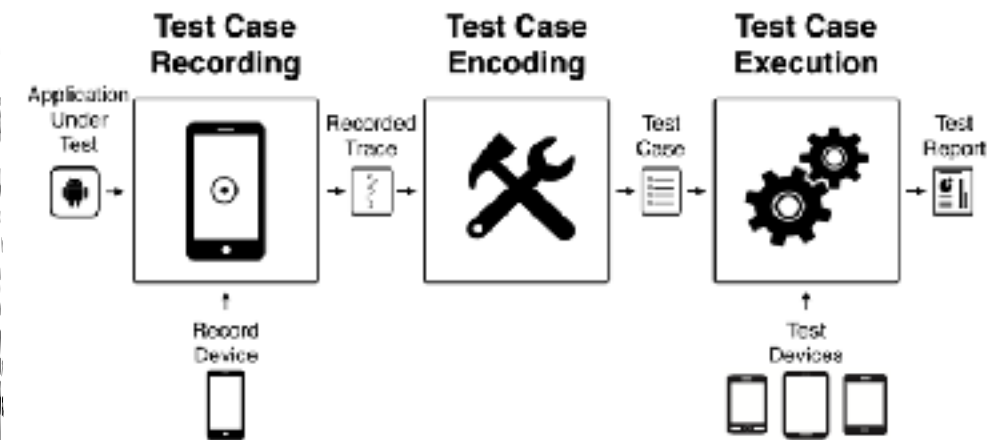


Summary

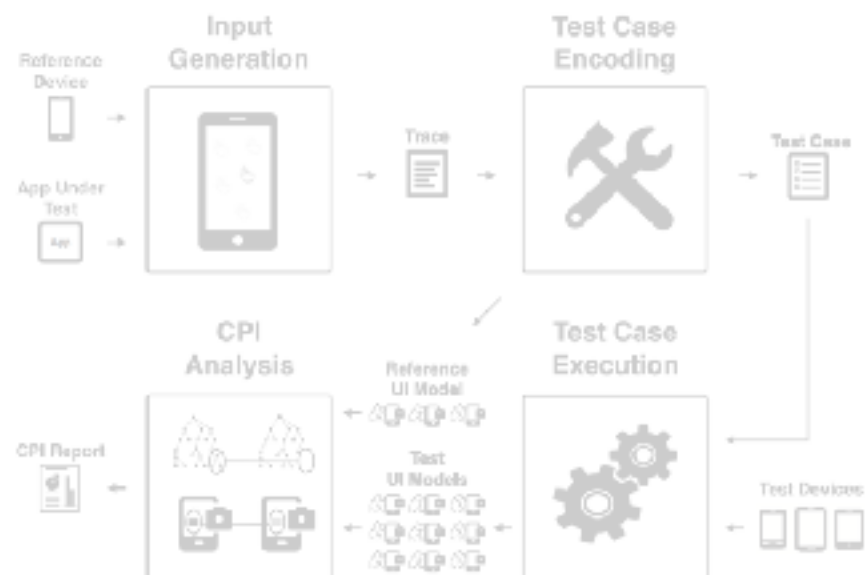
Failures



Barista Overview



DIFFDROID Overview

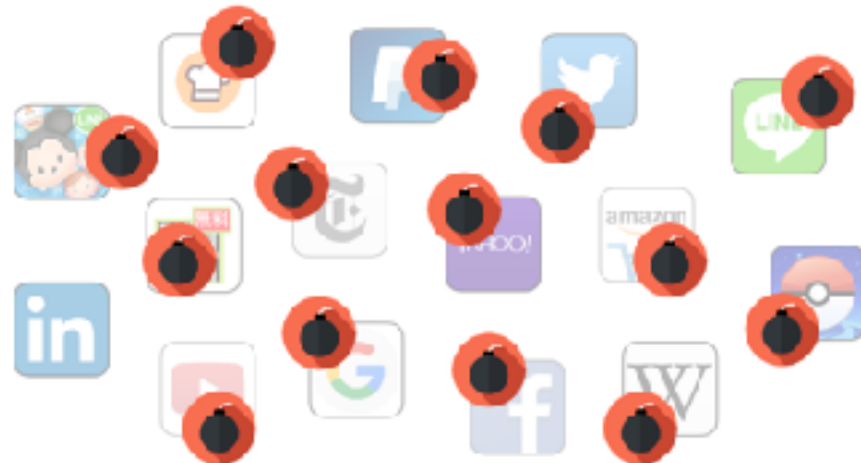


Yakusu Overview

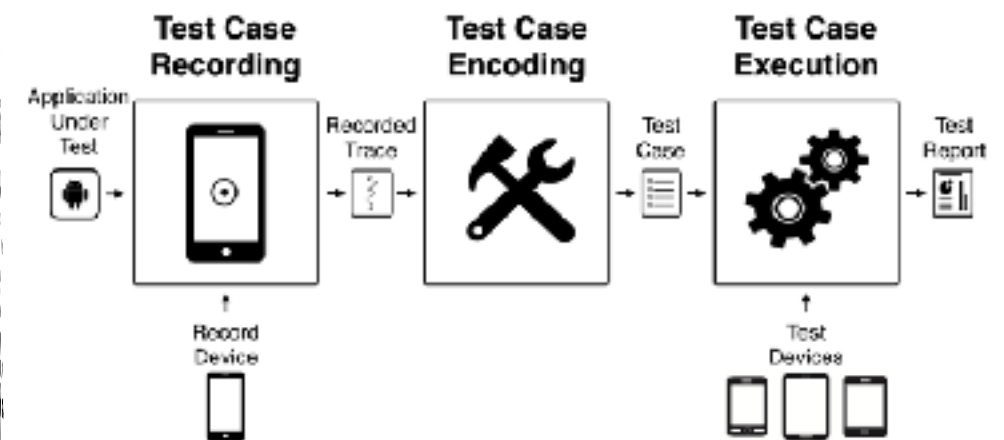


Summary

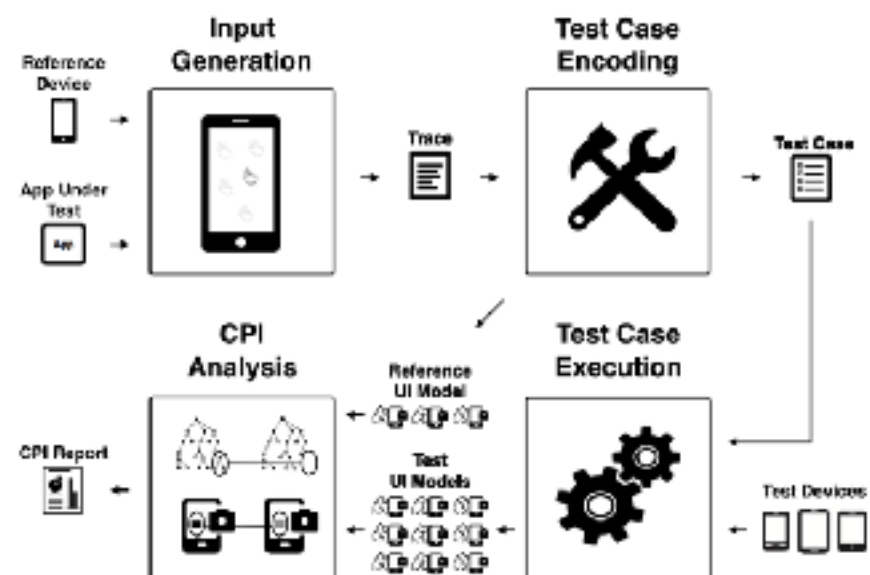
Failures



Barista Overview



DIFFDROID Overview

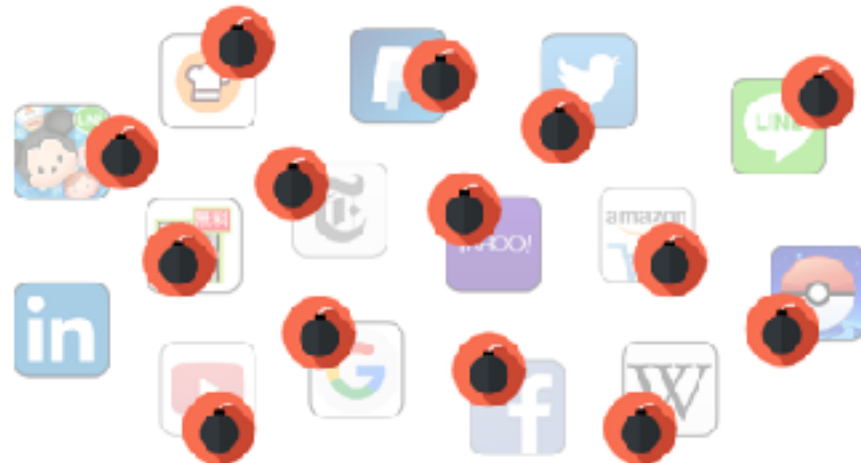


Yakusu Overview

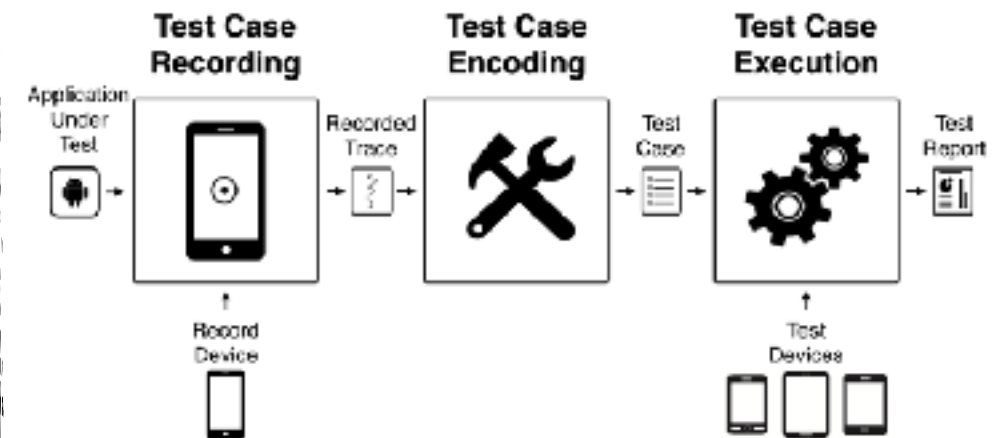


Summary

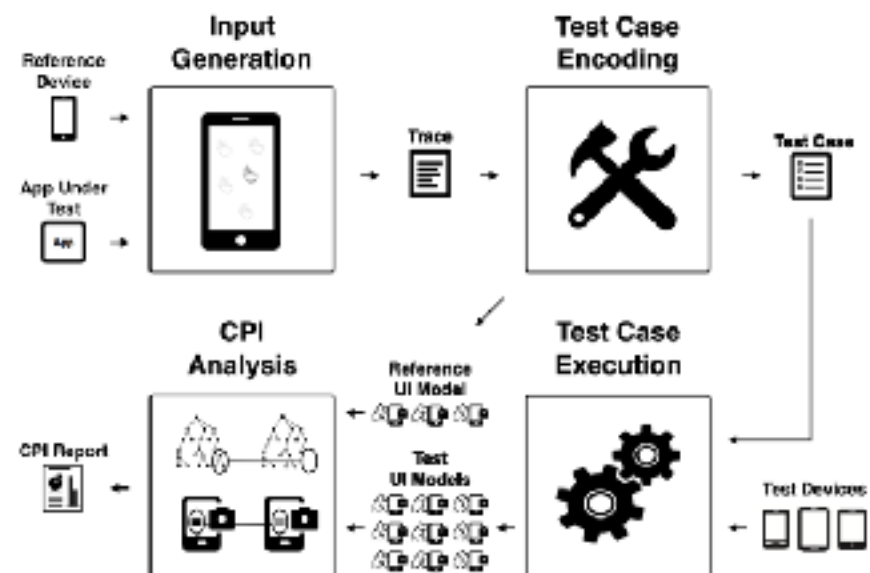
Failures



Barista Overview



DIFFDROID Overview



Yakusu Overview

