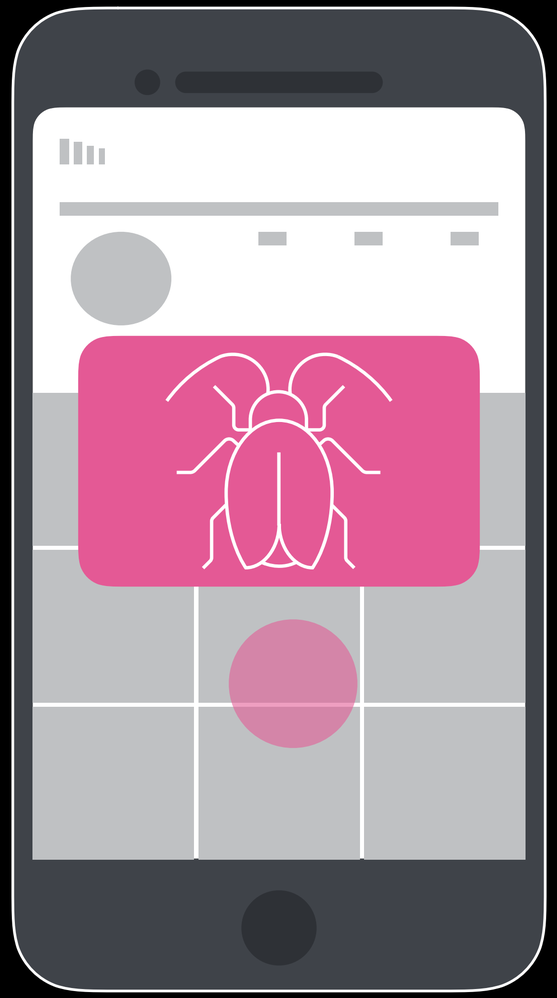
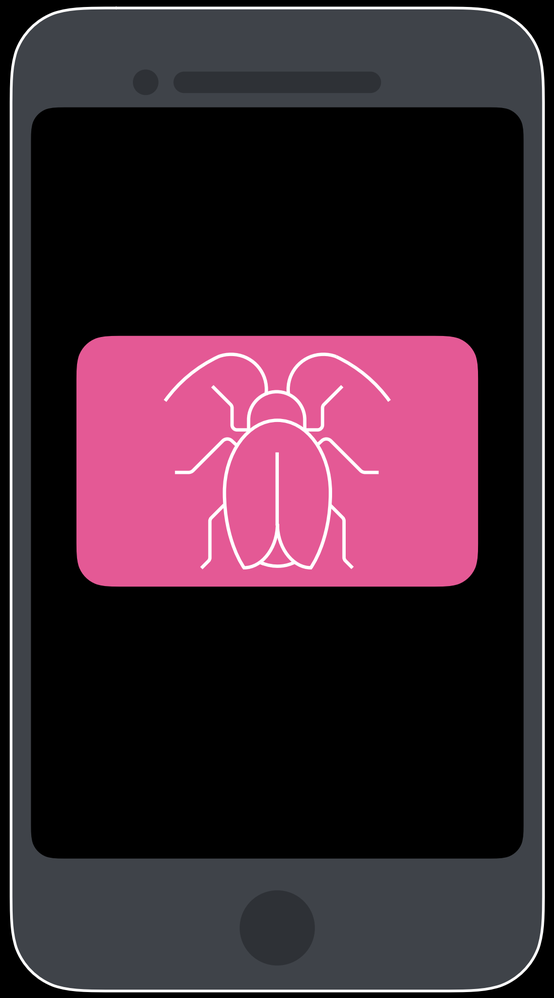
Automated Test Design and Bug Fixing @ Facebook

Nadia Alshahwan
FACEBOOK
SOFTWARE ENGINEER

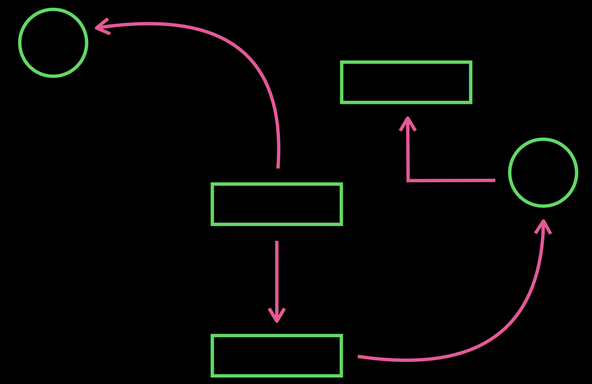
SAPIENZ TEAM; THIS TALK IS BASED ON WORK OF THE WHOLE TEAM





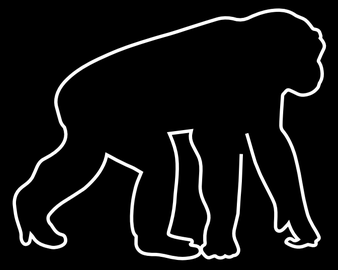
Unfortunately,
your app has stopped.

Engineers Design



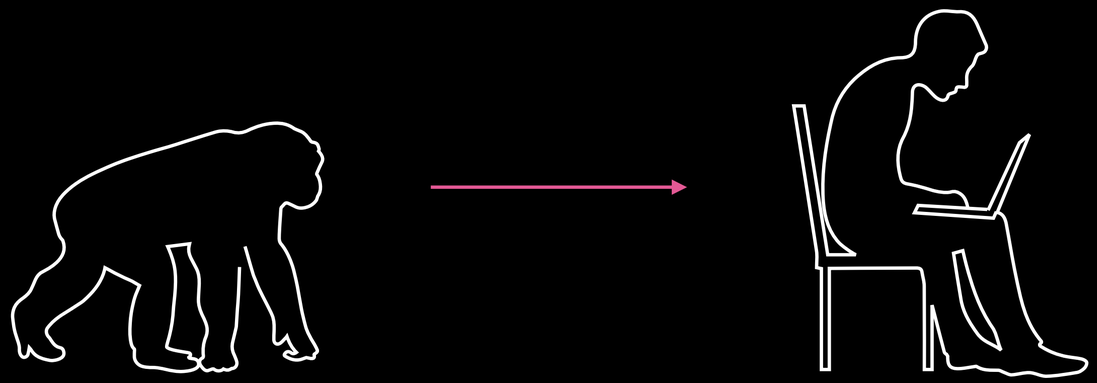
Machines Execute

System Level UI Testing



RANDOM FUZZER

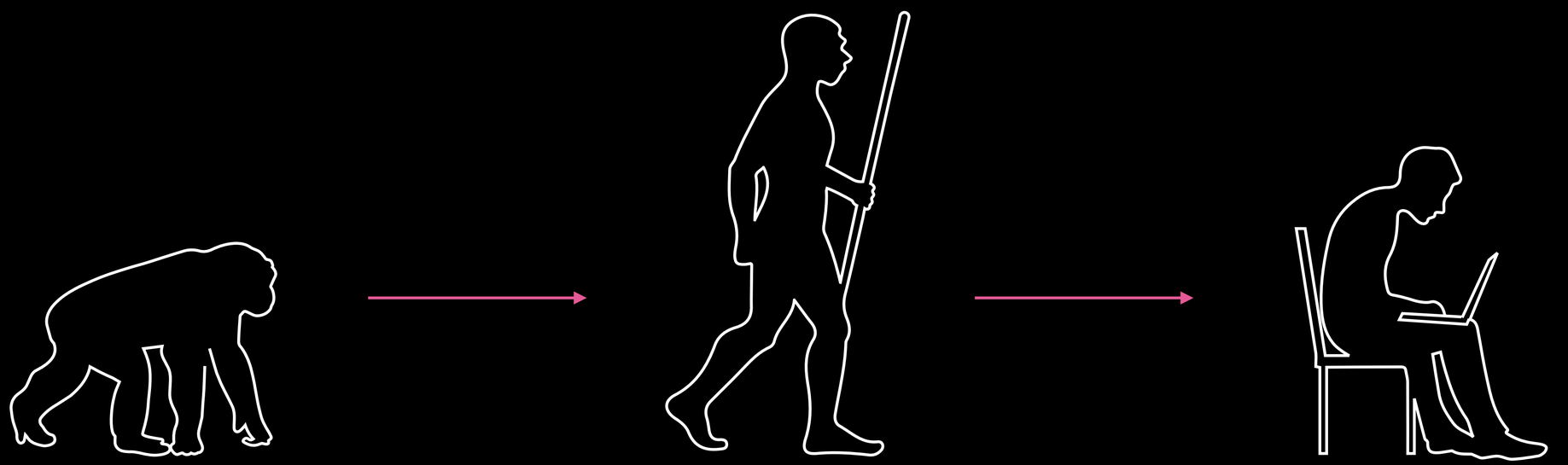
System Level UI Testing



RANDOM FUZZER

HUMAN TESTERS

System Level UI Testing



RANDOM FUZZER

sapienz

HUMAN TESTERS

Sapienz: Multi-Objective Automated Testing for Android Applications

KE MAO, MARK HARMAN, AND YUE JIA

IN PROC. OF ISSTA'16, 2016.

Sapienz: Multi-objective Automated Testing for Android Applications

Ke Mao Mark Harman Yue Jia
CREST Centre, University College London, Malet Place, London, WC1E 6BT, UK
k.mao@cs.ucl.ac.uk, mark.harman@ucl.ac.uk, yue.jia@ucl.ac.uk

ABSTRACT

We introduce Sapienz, an approach to Android testing that uses multi-objective search-based testing to automatically explore and optimise test sequences, minimising length, while simultaneously maximising coverage and fault revelation. Sapienz combines random fuzzing, systematic and search-based exploration, exploiting seeding and multi-level instrumentation. Sapienz significantly outperforms (with large effect size) both the state-of-the-art technique Dynodroid and the widely-used tool, Android Monkey, in 7/10 experiments for coverage, 7/10 for fault detection and 10/10 for fault-revealing sequence length. When applied to the top 1,000 Google Play apps, Sapienz found 558 unique, previously unknown crashes. So far we have managed to make contact with the developers of 27 crashing apps. Of these, 14 have confirmed that the crashes are caused by real faults. Of those 14, six already have developer-confirmed fixes.

CCS Concepts

•**Software and its engineering** → **Software testing and debugging;** *Search-based software engineering;*

Keywords

Android; Test generation; Search-based software testing

1. INTRODUCTION

There are over 1.8 million apps available from the Google Play marketplace, as of January 2016 [9]. For developed internet markets such as the US, UK and Canada, mobile

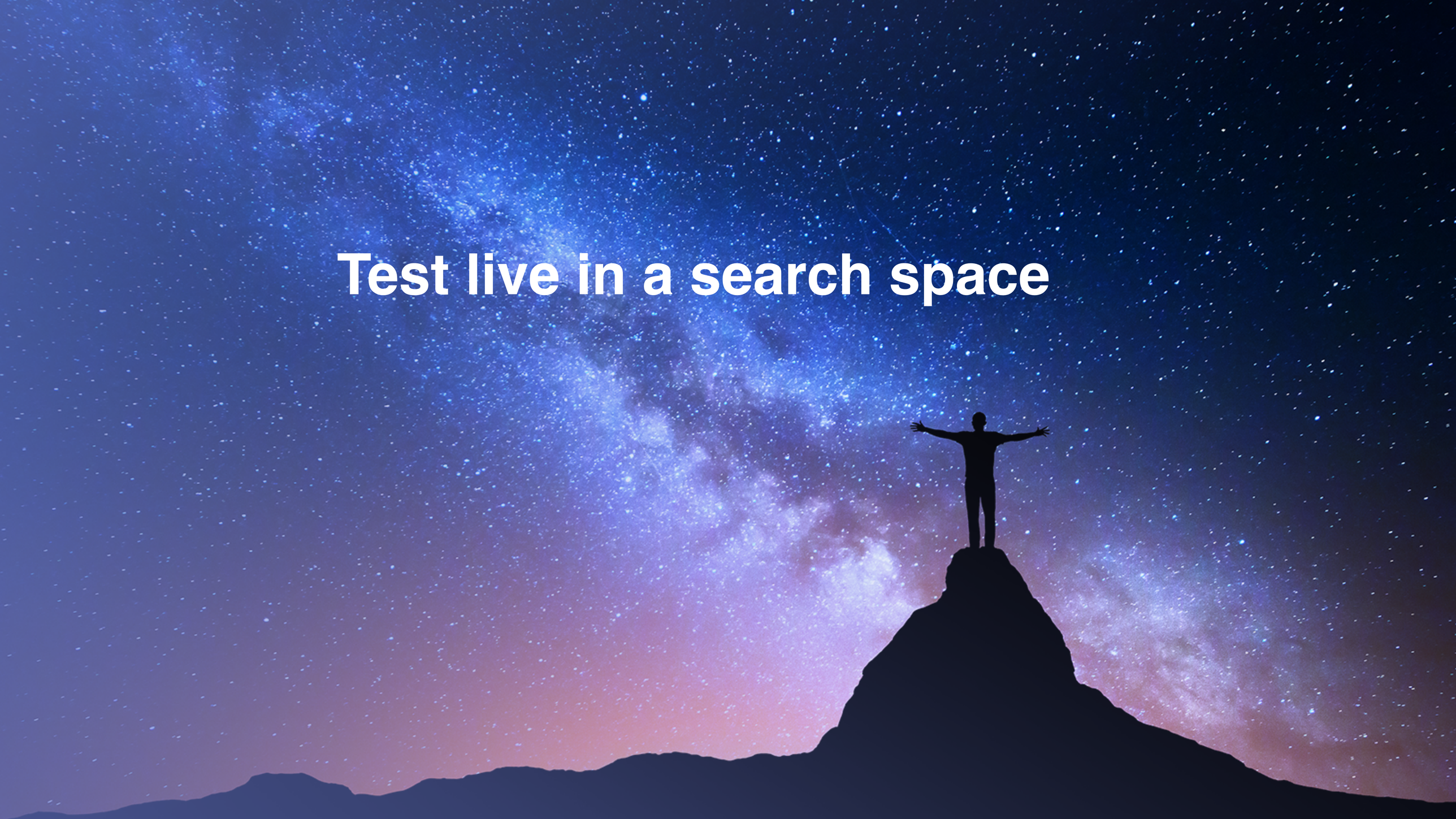
Where test automation does occur, it typically uses Google's Android Monkey tool [36], which is currently integrated with the Android system. Since this tool is so widely available and distributed, it is regarded as the current state-of-practice for automated software testing [53]. Although Monkey automates testing, it does so in a relatively unintelligent manner: generating sequences of events at random in the hope of exploring the app under test and revealing failures. It uses a standard, simple-but-effective, default test oracle [22] that regards any input that reveals a crash to be a fault-revealing test sequence.

Automated testing clearly needs to find such faults, but it is no good if it does so with exceptionally long test sequences. Developers may reject longer sequences as being impractical for debugging and also unlikely to occur in practice; the longer the generated test sequence, the less likely it is to occur in practice. Therefore, a critical goal for automated testing is to find faults with the *shortest possible* test sequences, thereby making fault revelation more actionable to developers.

Exploratory testing is "simultaneous learning, test design, and test execution" [11], that can be cost-effective and is widely used by industrial practitioners [21, 43, 46] for testing in general. However, it is particularly underdeveloped for mobile app testing [41, 42]. Although there exist several test automation frameworks such as Robotium [10] and Appium [3], they require human-implemented scripts, thereby inhibiting full automation.

We introduce Sapienz, the first approach offering multi-objective automated Android app exploratory testing that seeks to maximise code coverage and fault revelation, while

Test live in a search space



Auto Test Design

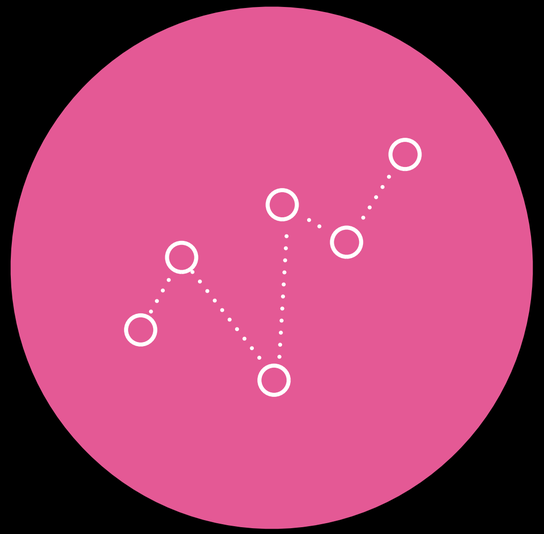
Auto Fix

Auto Boost

Auto Test Design

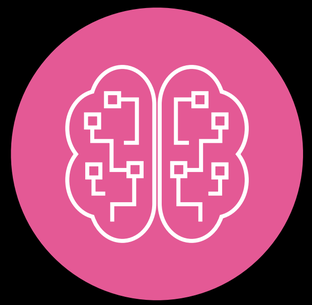
Auto Fix

Auto Boost



Auto Test Design

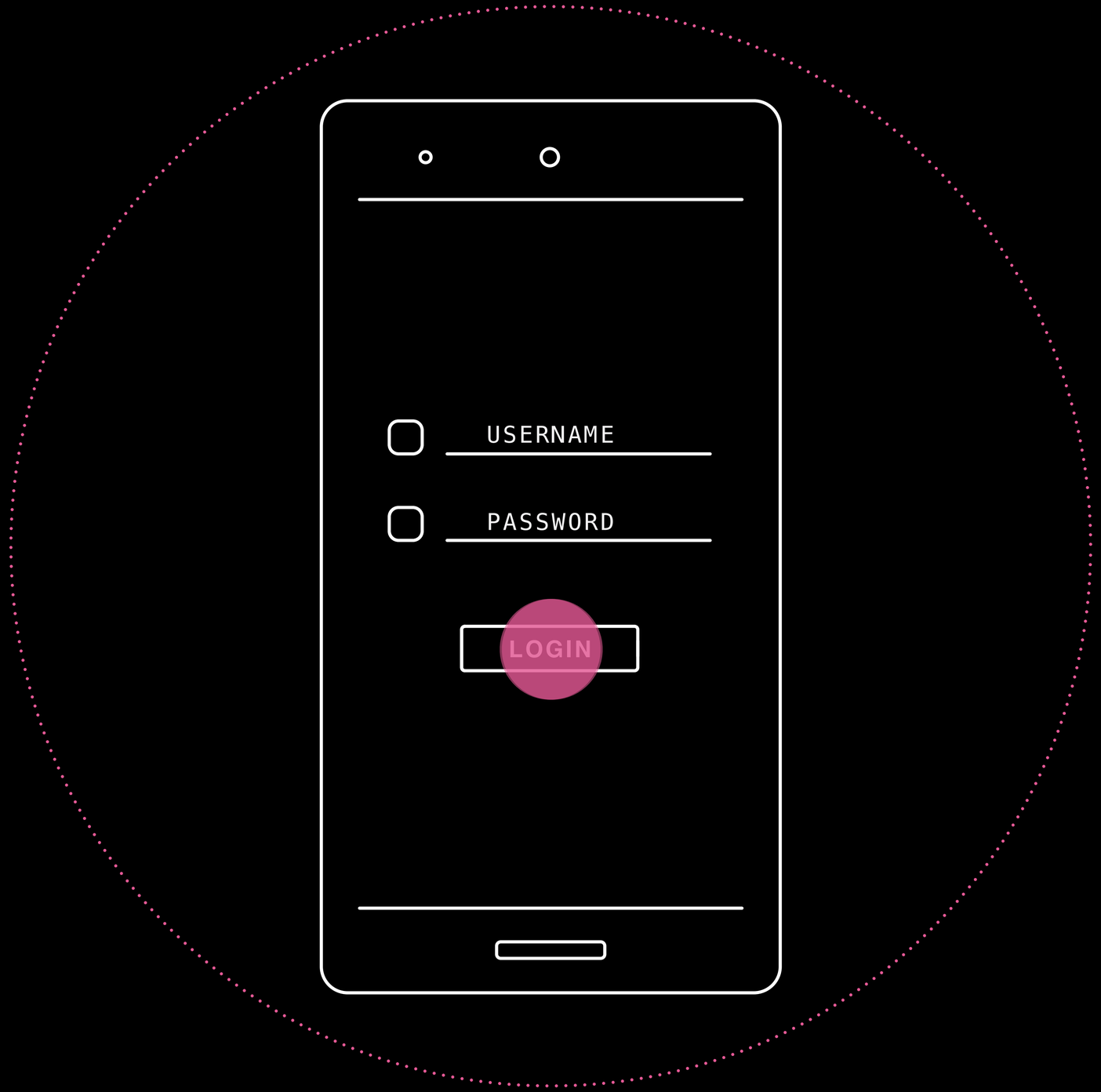
Features

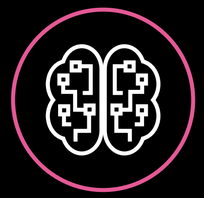


MOTIF
INTERACTION

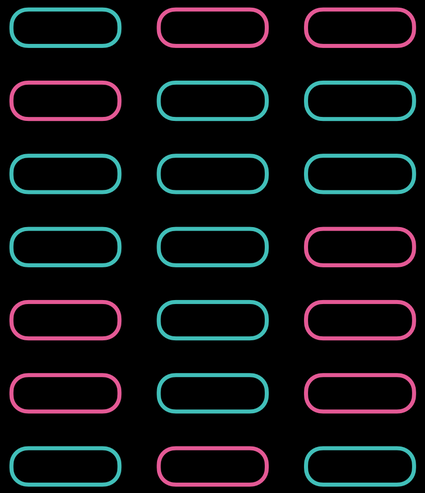
GENETIC
ALGORITHM

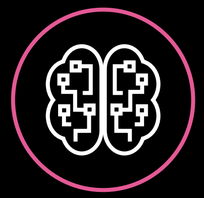
MULTI-
OBJECTIVE


USERNAME
PASSWORD
LOGIN

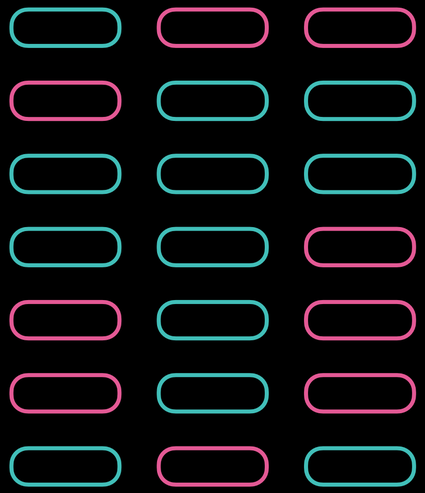


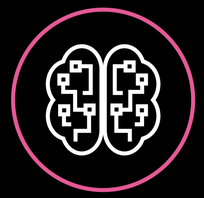
TEST GENERATION



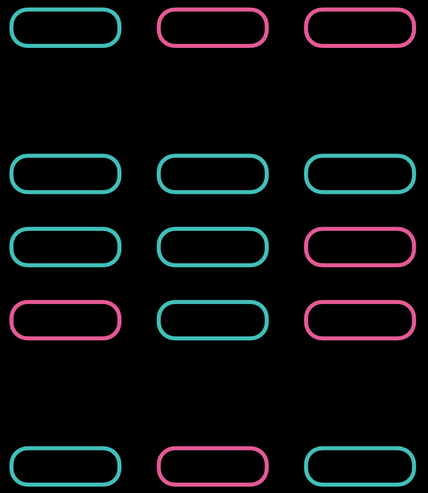


TEST GENERATION

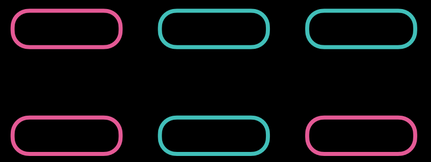


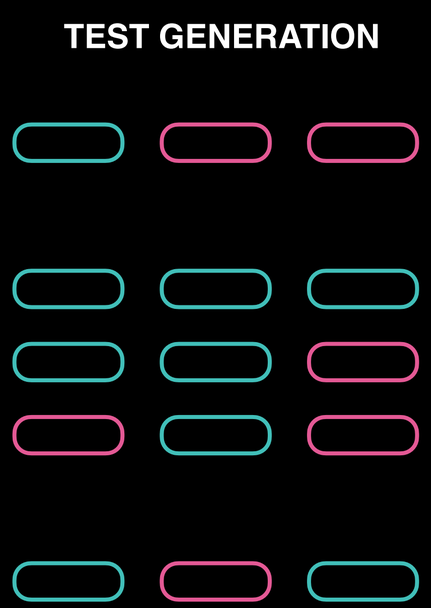


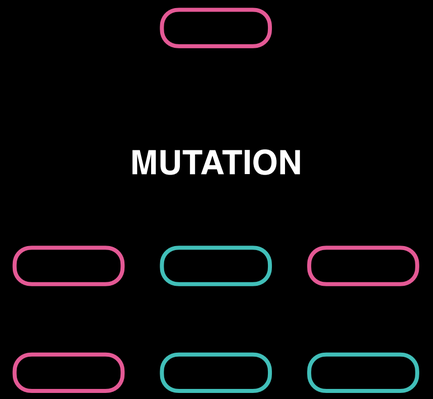
TEST GENERATION

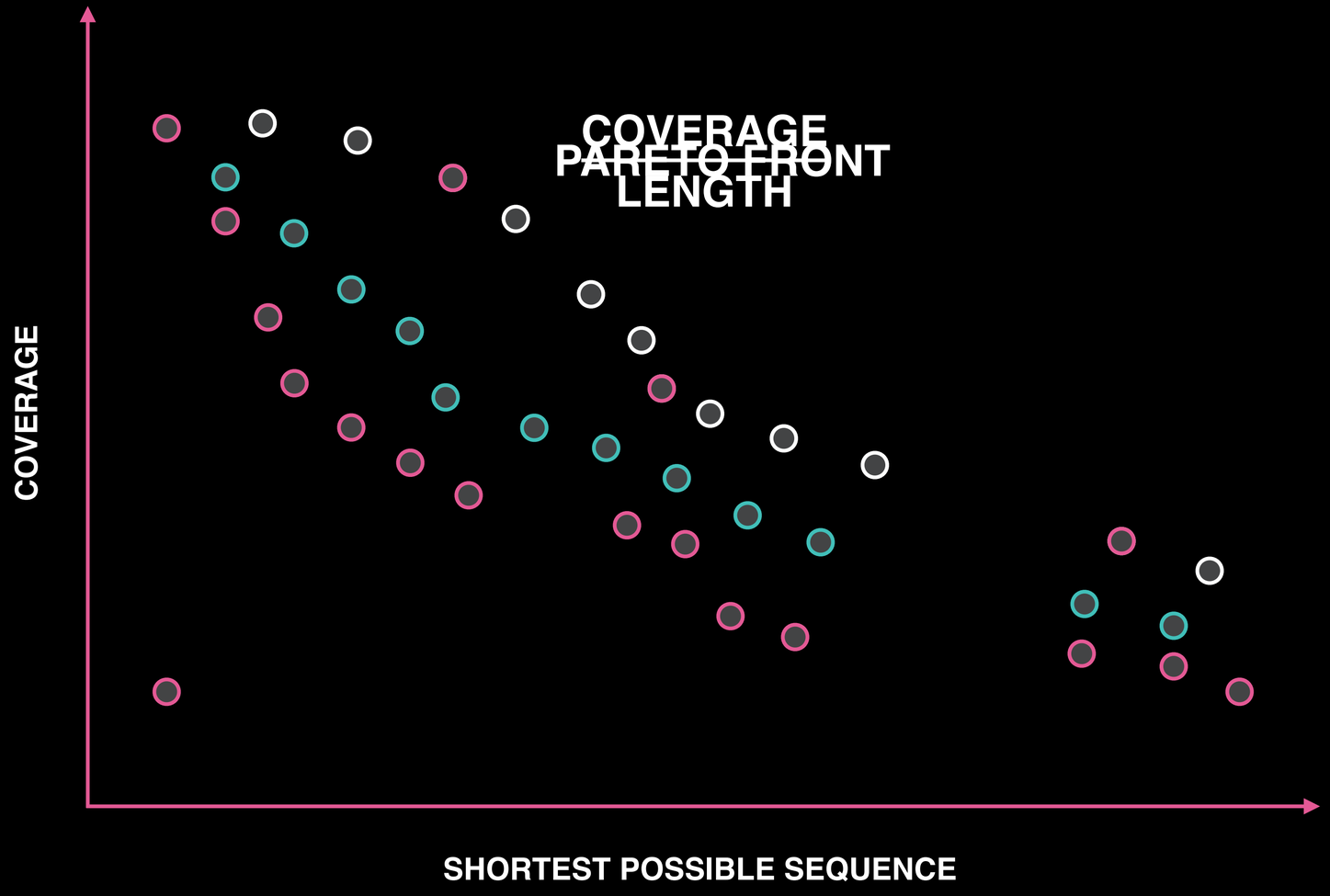


CROSSOVER

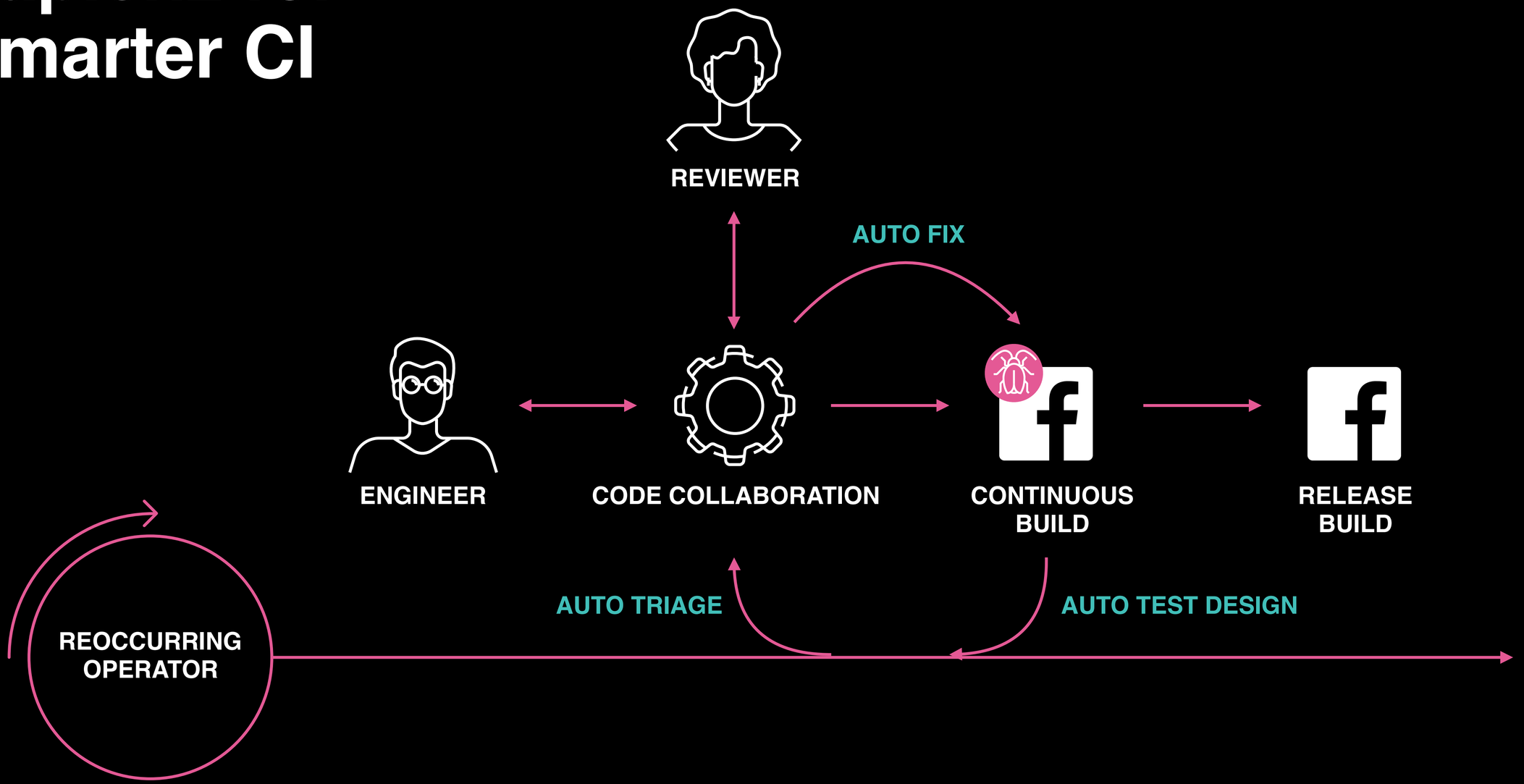



TEST GENERATION

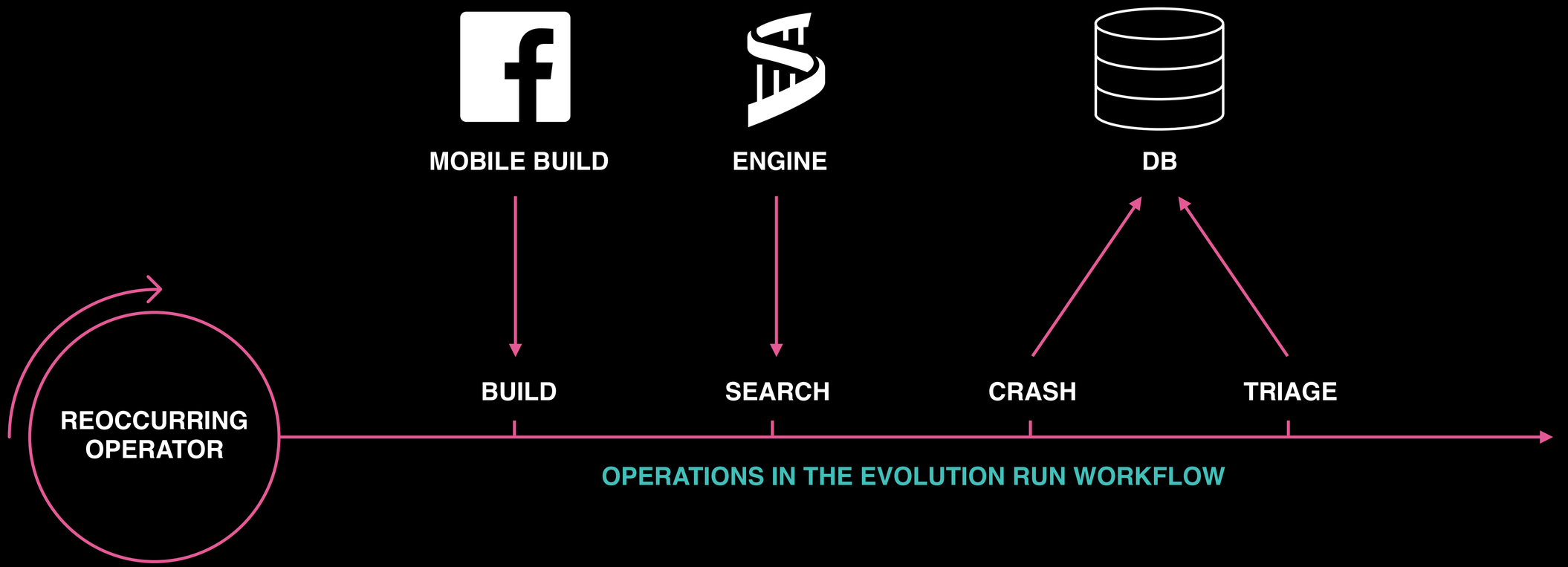

MUTATION


COVERAGE
PARETO FRONT
LENGTH
COVERAGE
SHORTEST POSSIBLE SEQUENCE

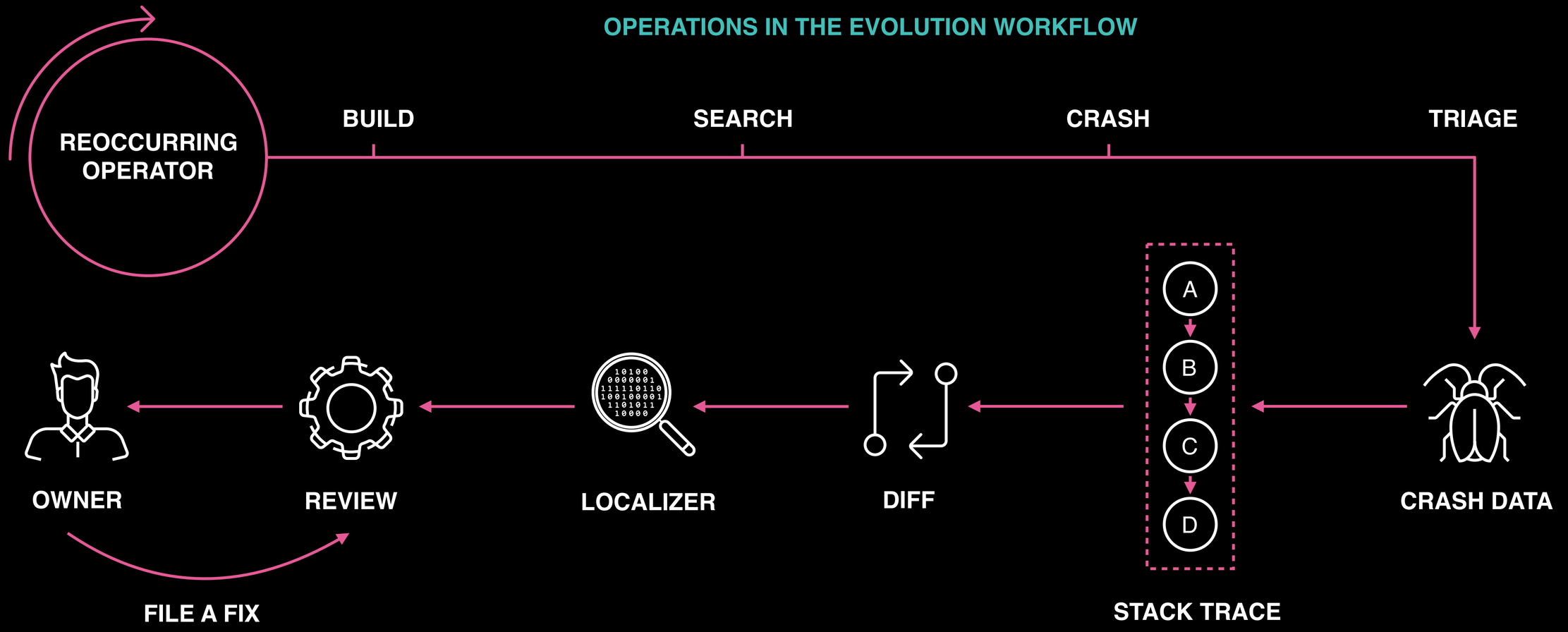
Sapienz for Smarter CI



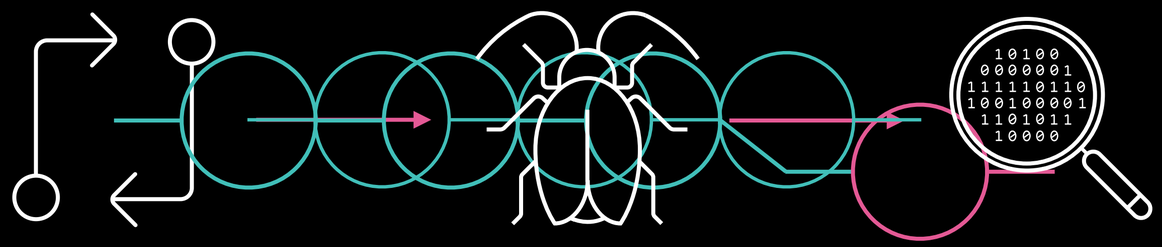
Sapienz Workflow



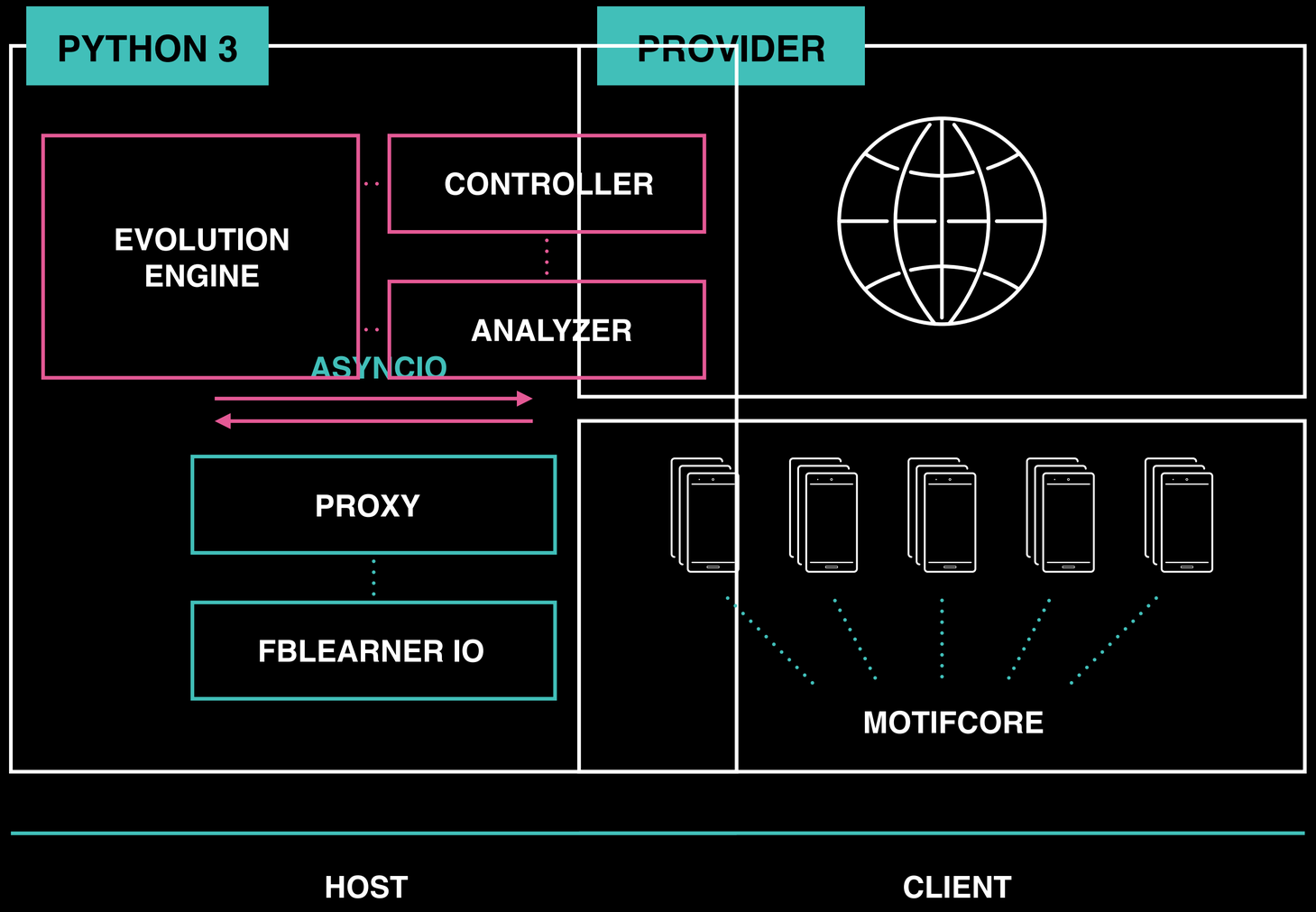
Fault Triage Process



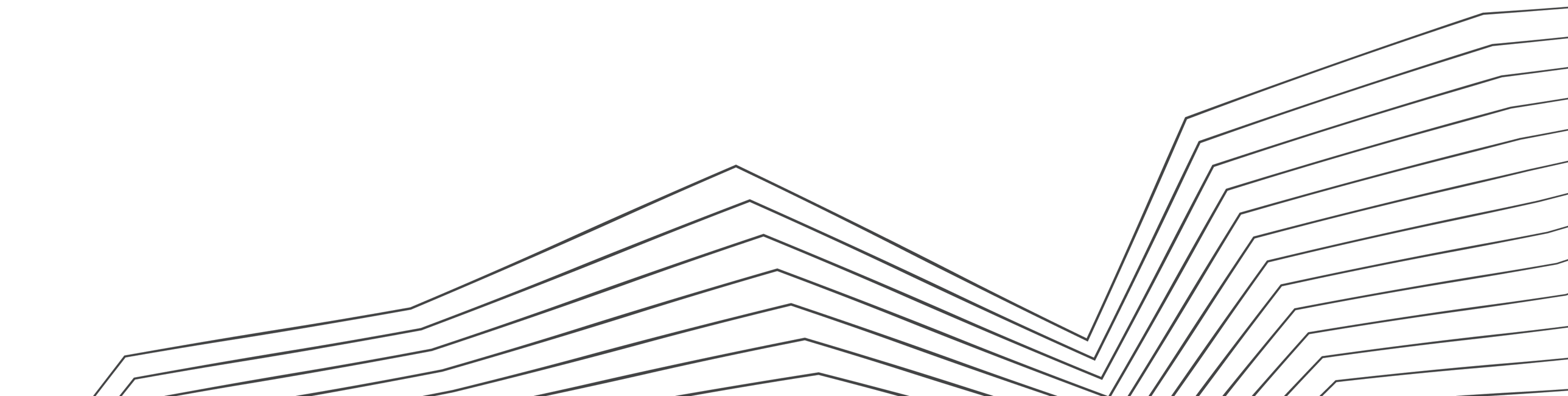
Diff-time Signals



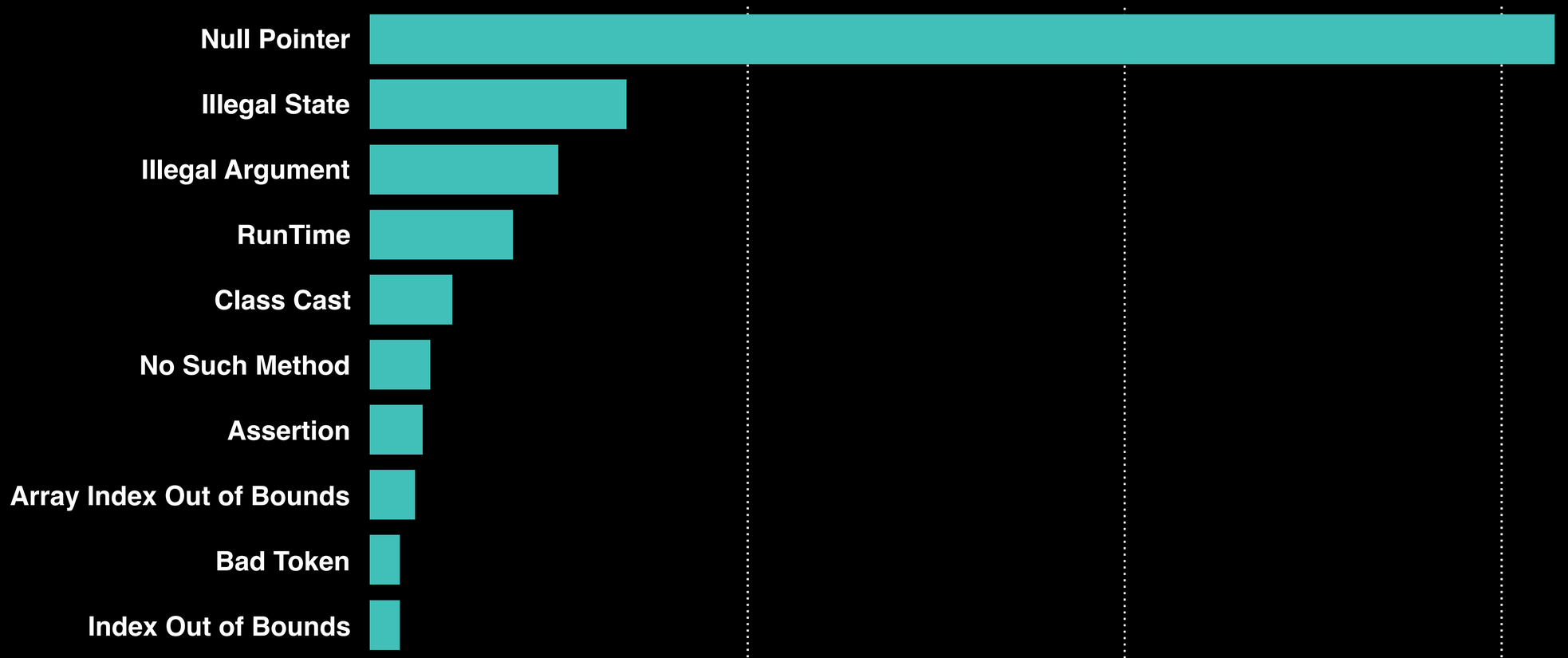
Test Design


PYTHON 3
PROVIDER
EVOLUTION ENGINE
CONTROLLER
ANALYZER
ASYNCIO
PROXY
FBLEARNER IO
MOTIFCORE
HOST
CLIENT

~75%

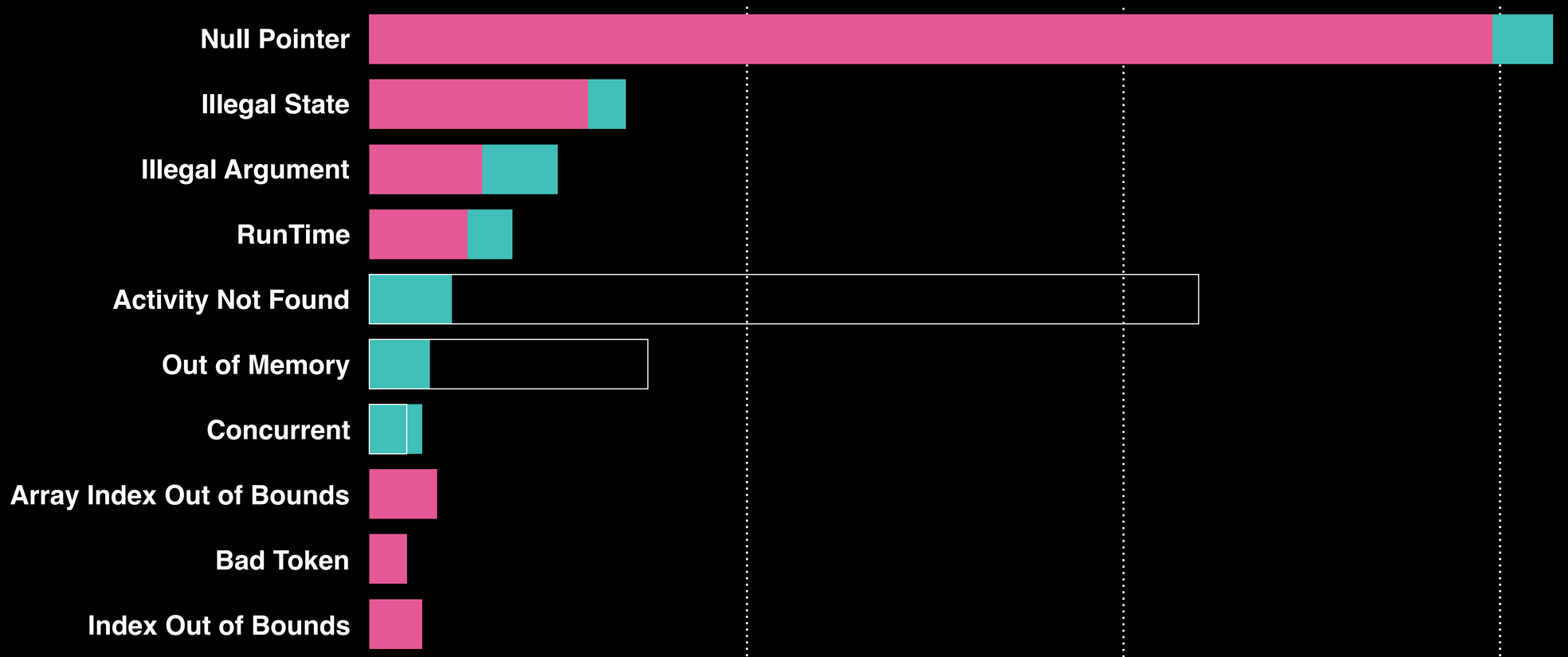


Distribution (FB)



TOP CRASHES TYPES ON FACEBOOK FOR ANDROID (BY SAPIENZ)

Distribution (Research)



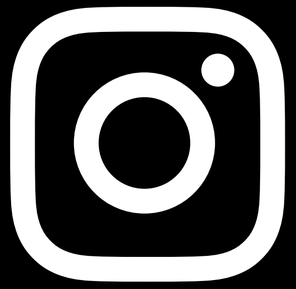
TOP CRASHES TYPES ON 1000 ANDROID APPS (BY SAPIENZ[1])

[1] K. MAO, M. HARMAN, AND Y. JIA, “SAPIENZ: MULTI-OBJECTIVE AUTOMATED TESTING FOR ANDROID APPLICATIONS,” IN PROC. OF ISSTA’16, 2016

Support on Facebook App Family

FACEBOOK

FBLITE

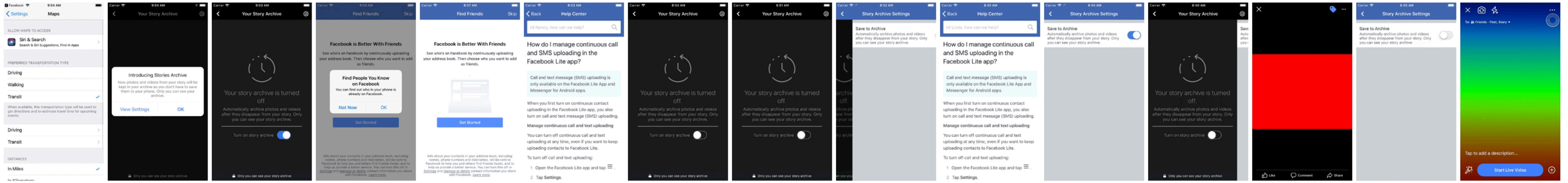


INSTAGRAM

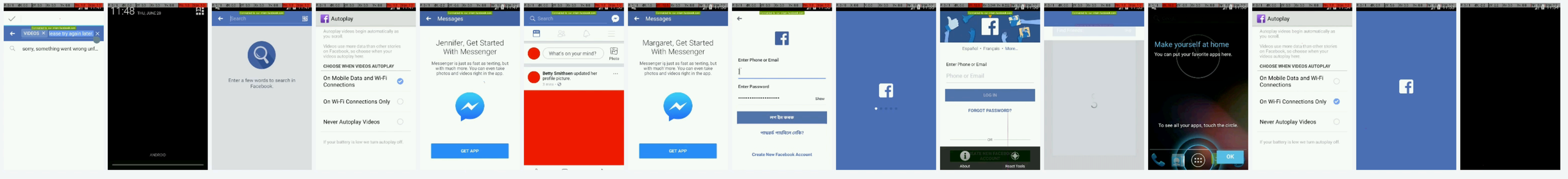
MESSENGER

WORKPLACE

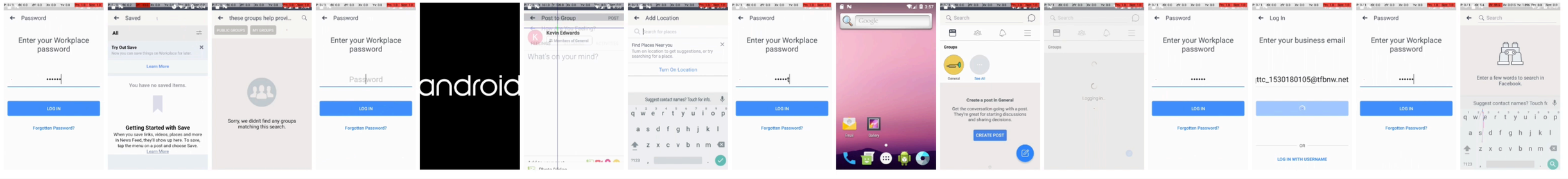
Facebook for iOS



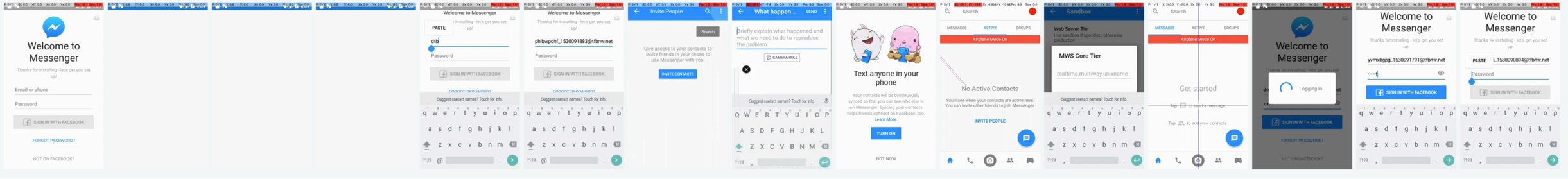
Facebook for Android



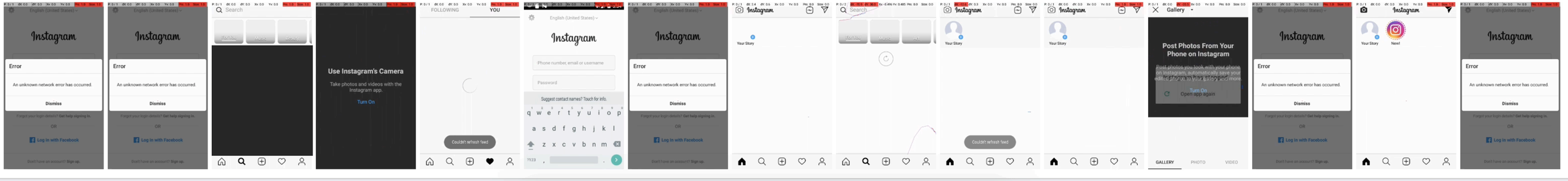
Workplace for Android



Messenger for Android



Instagram for Android



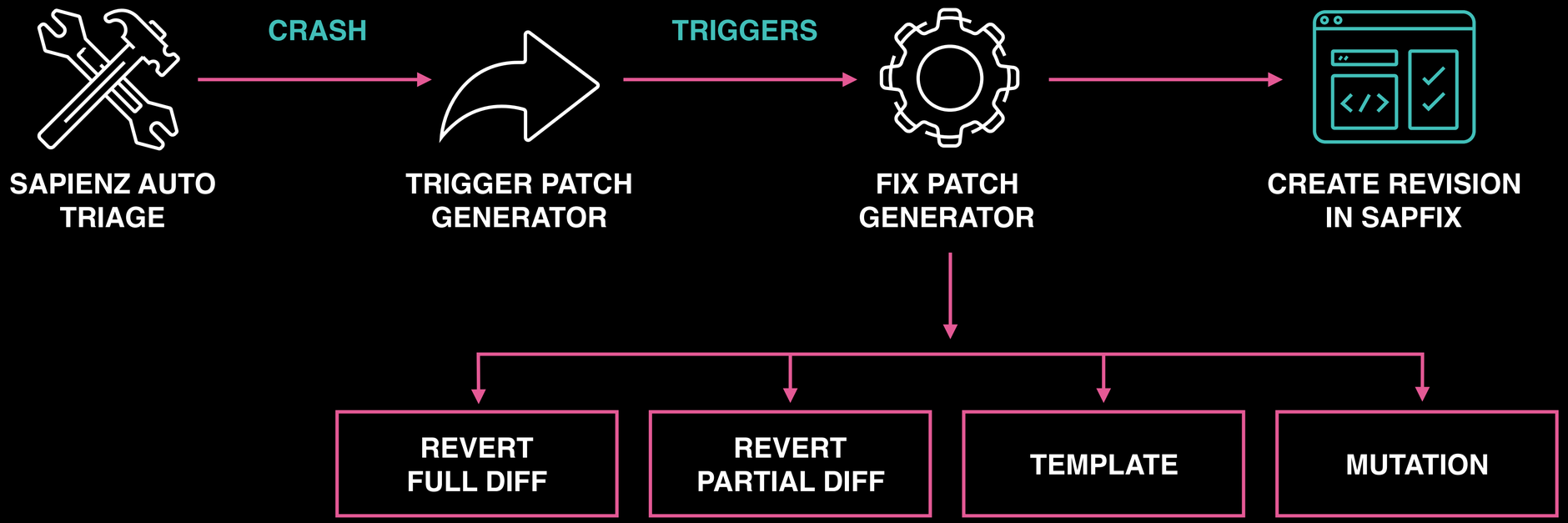
Auto Test Design

Auto Fix

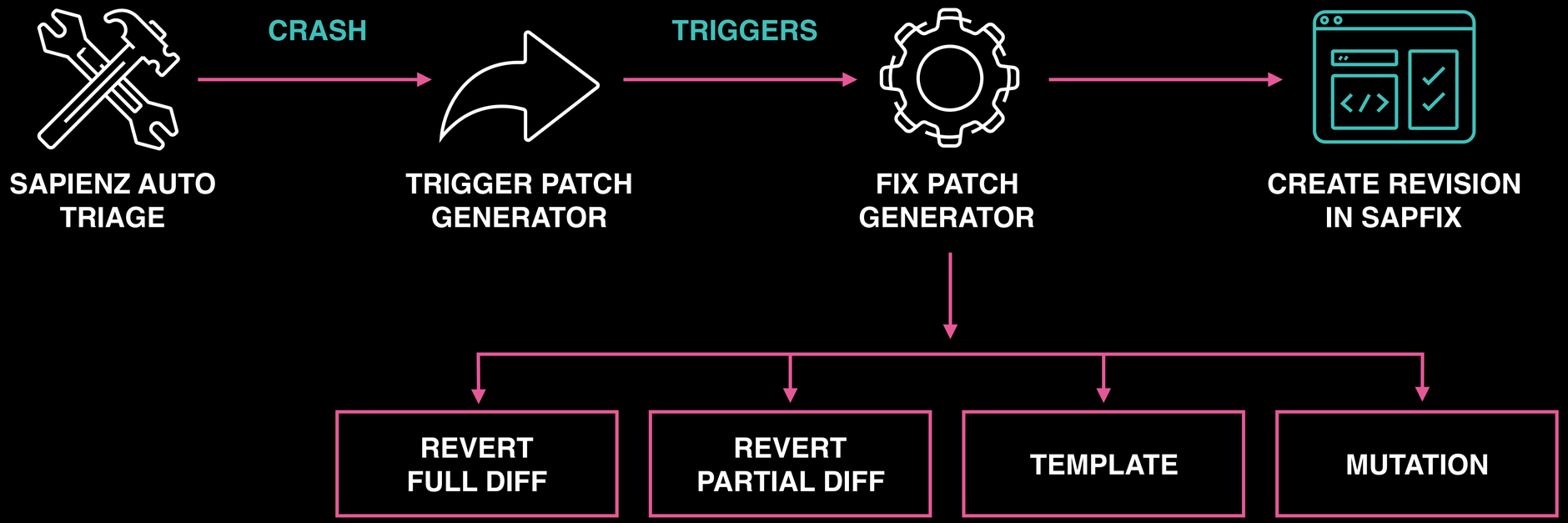
Auto Boost

Auto Fix

Auto Fix Workflow (Generation)

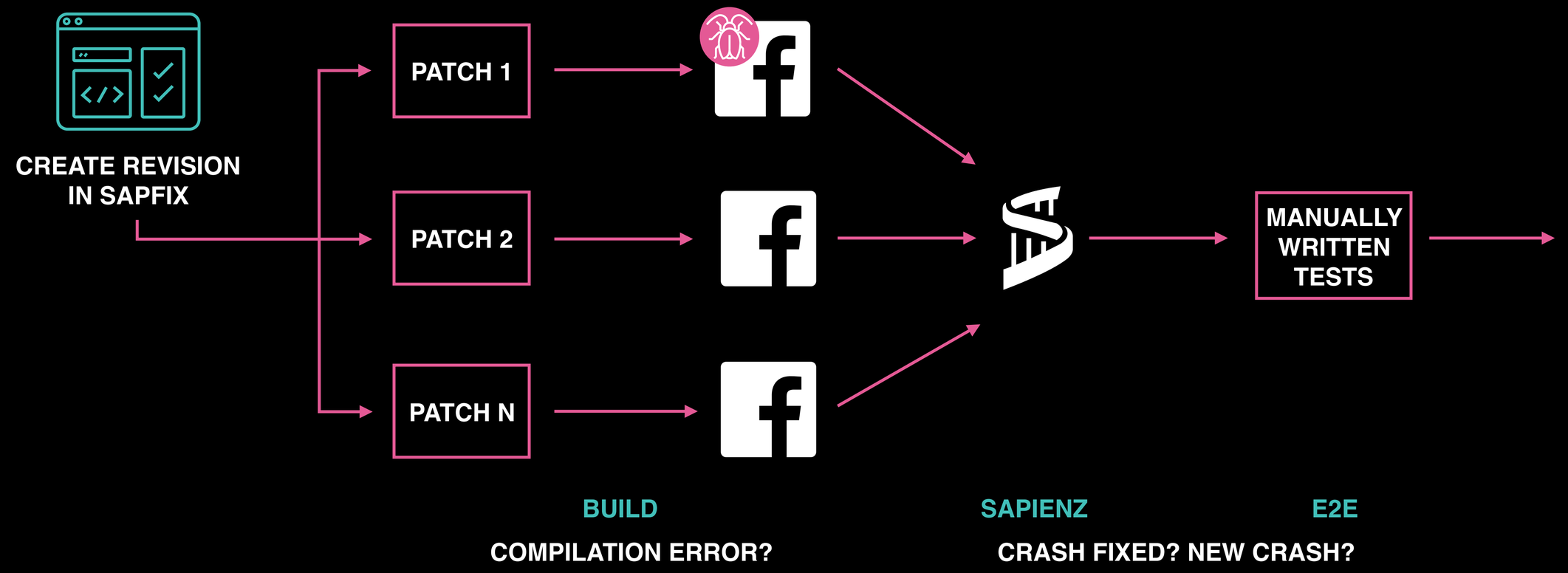


Auto Fix Workflow (Generation)

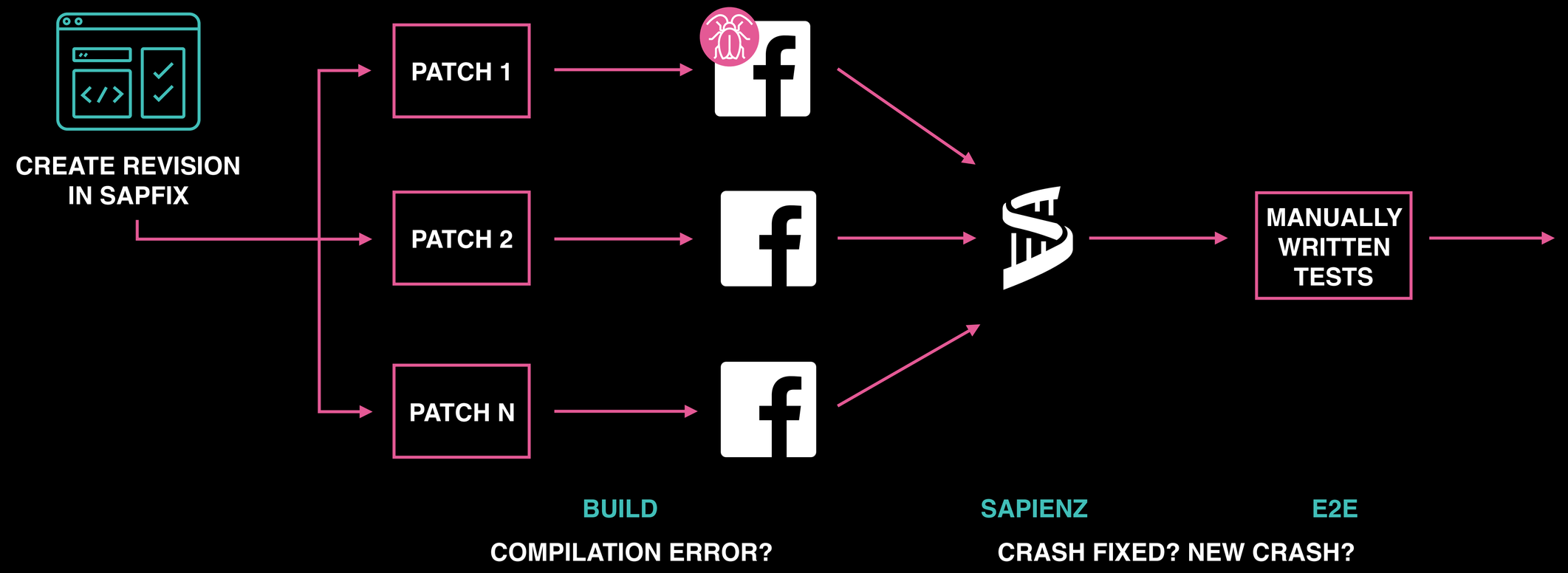


Auto Fix Workflow (Validation)

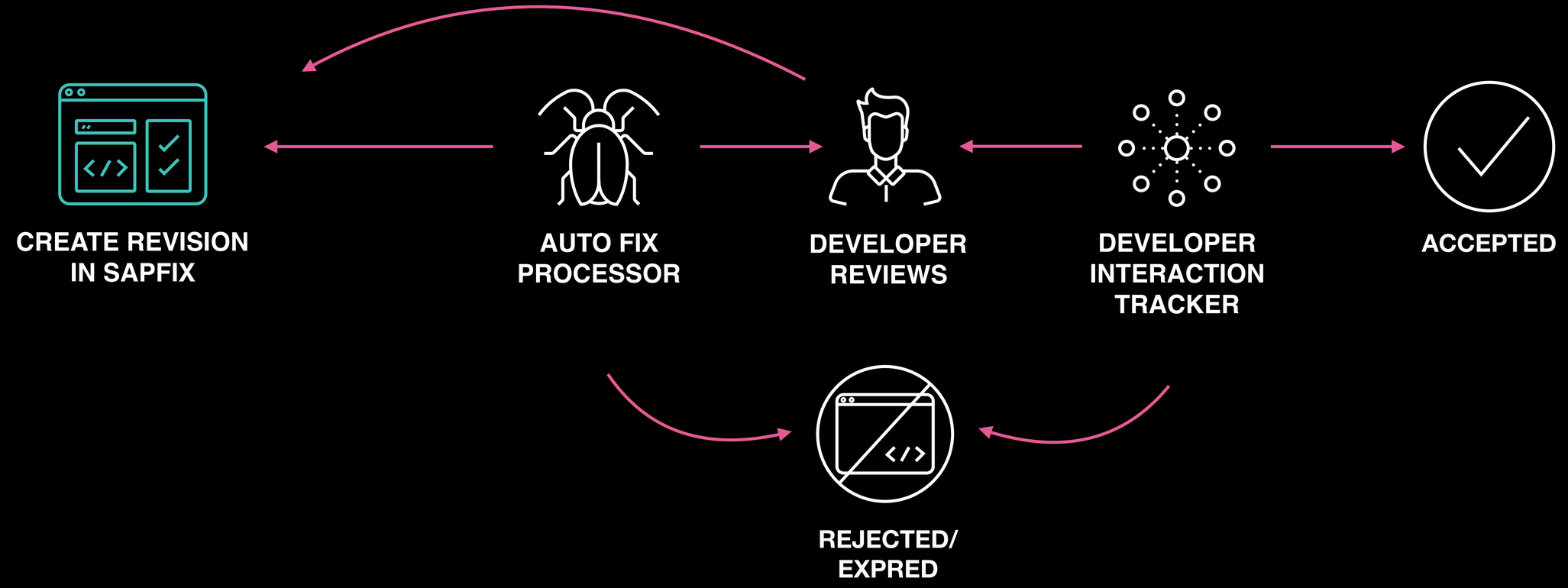
Auto Fix Workflow (Validation)



Auto Fix Workflow (Validation)



Auto Fix Workflow (Signal)



Auto Test Design

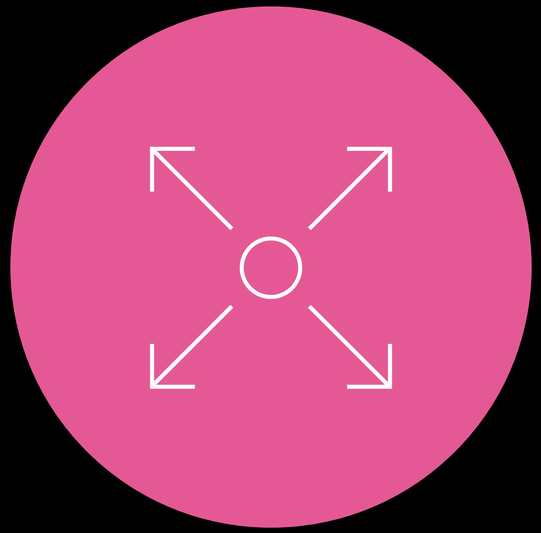
Auto Fix

Auto Boost

Auto Test Design

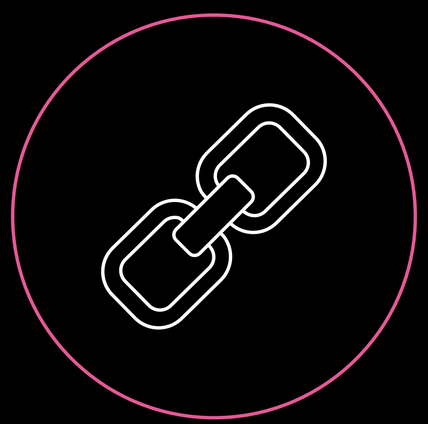
Auto Fix

Auto Boost



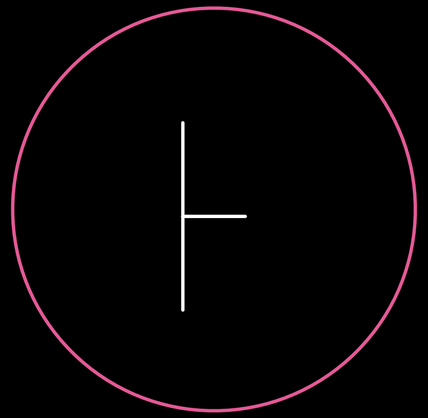
Auto Boost

Signal boosting



PROD CRASH LINK

AFFECTED USER PREDICTION



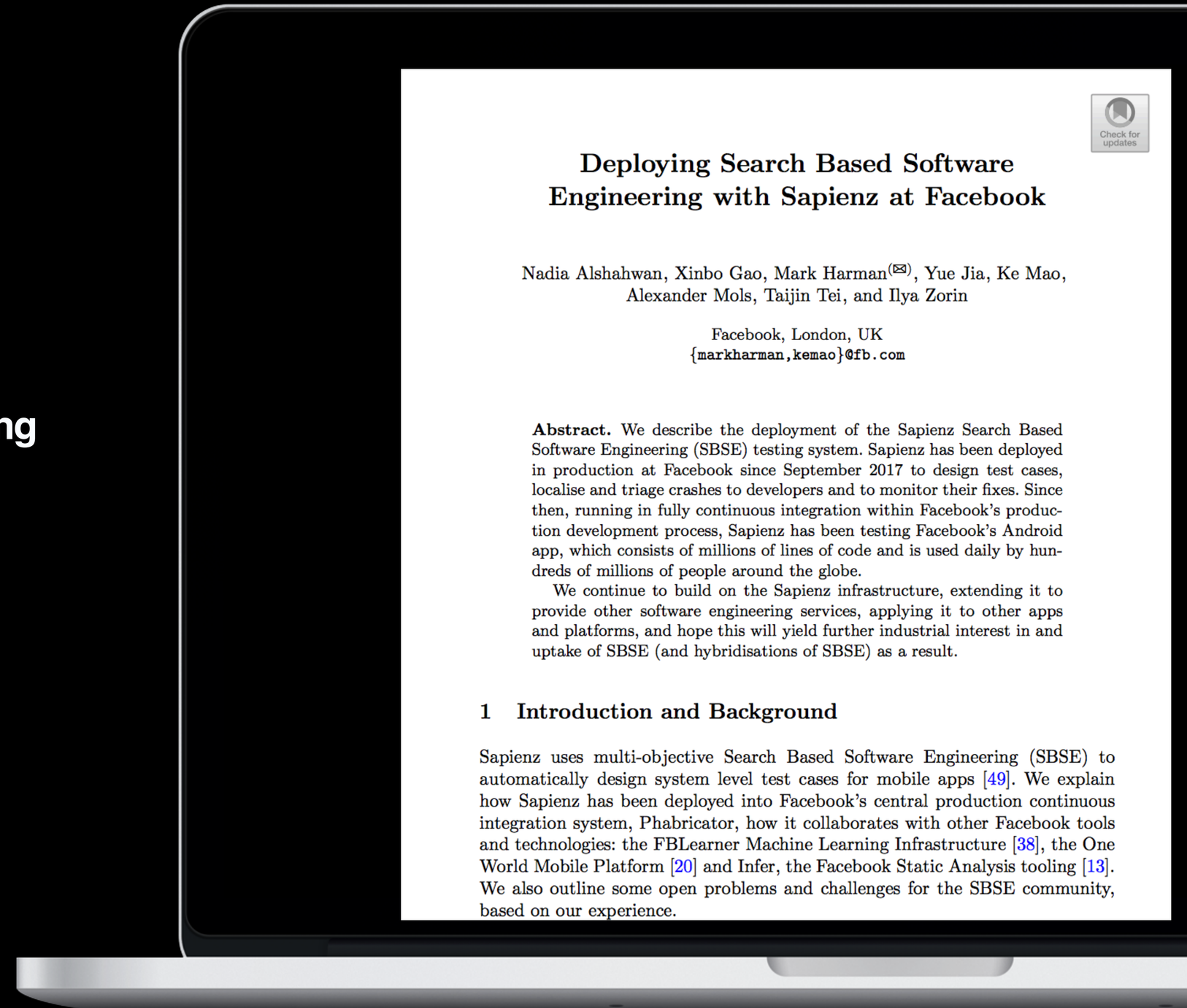
INFER LINK

Resources

Deploying Search Based Software Engineering with Sapienz at Facebook

NADIA ALSHAHWAN,
XINBO GAO,
MARK HARMAN,
YUE JIA,
KE MAO,
ALEXANDER MOLS,
TAIJIN TEI
AND
ILYA ZORIN

IN PROC. OF SSBSE, 2018.



Flaky tests
Fix detection
Automated Oracles
Smarter white box coverage
Combining static and dynamic
Unit test from system tests
Human machine test hybrids
Fully parallel search algorithms
Auto Fix and perf improvement

nalshahwan@fb.com