Johannah Sprinz

Crowdsourced User-Testing

Bachelor Thesis Presentation “Leveraging Human Computation for Quality Assurance in Open Source Communities”

Submission Date: 29.01.2022
Presentation Date: 10.02.2022
Principal Advisor: Prof. Dr. François Bry

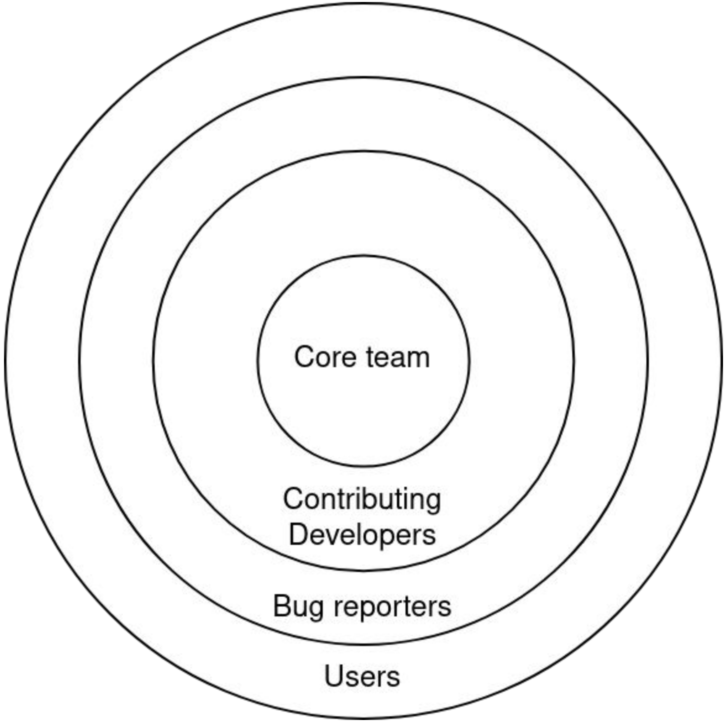
LMU Munich
Institute of Informatics
Teaching and Research Unit Programming and Modelling Languages

Motivation: Crowdsourcing and Human Computation

Crowdsourcing: many people working towards a common goal (eg. Wikipedia)

Human Computation: delegating computation steps that are difficult to automate from a machine to a human agent (eg. reCAPTCHA)

Motivation: Open Source Software Development (OSSD)



publicly available source code

typically developed in volunteer communities

there may be paid staff, but a large part of the work will be done by volunteers

volunteer developers work on things relevant to their own use case

Motivation: Software Quality Assurance (QA)

Quality Assurance: ensuring a software product’s fitness for use

more complex than expected, manual user-testing takes time

unit-, integration-, and system-tests can be automated

cave: automated tests need to be created and maintained

How does QA work for open-source communities?

automated tests may be neglected if developers are not interested in them

beta versions may be published before new releases for community feedback

users report bugs to allow developers to quickly address regressions

problem: **little formalized testing, edge cases might not be thoroughly tested**

idea: **develop a human computation platform for crowdsourced user-testing**

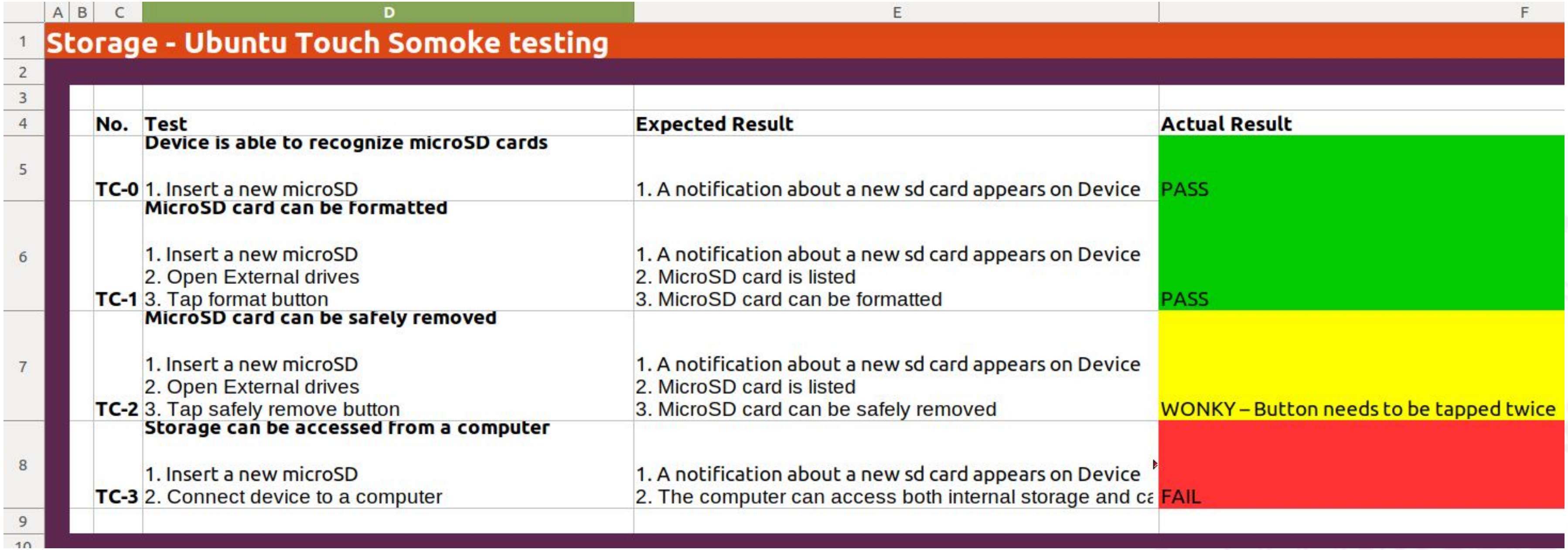
Example Community: UBports Foundation

community of the Ubuntu Touch mobile OS

community structure resembles the onion model:

- <10 paid developers
- core team of ~30 developers
- ~200 contributing developers
- ~500 bug reporters
- ~15k users

Use Case: Ubuntu Touch

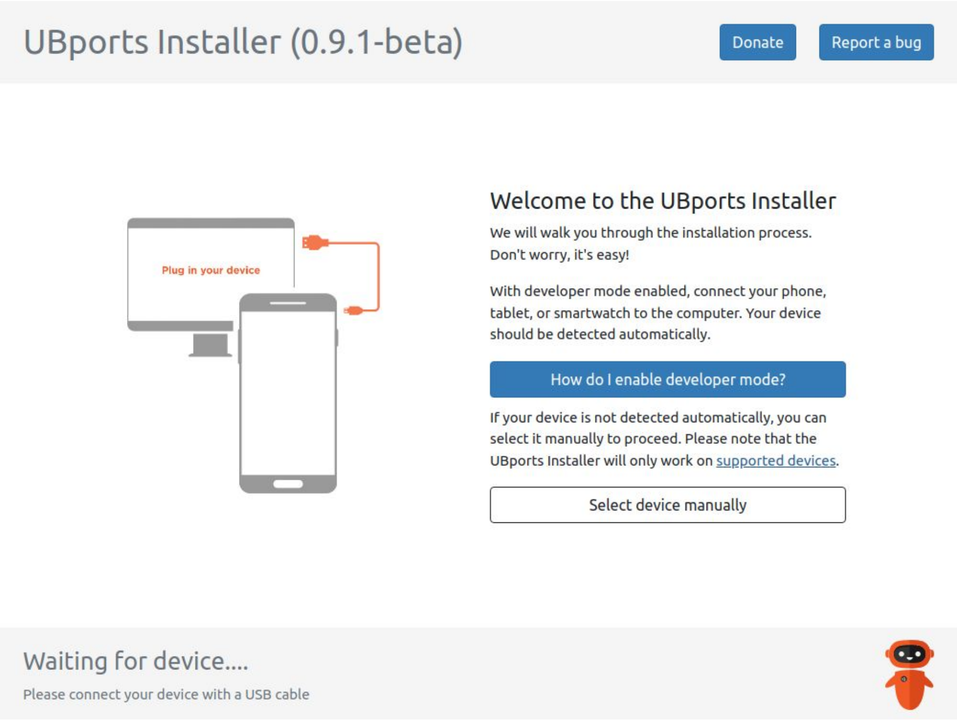


Storage - Ubuntu Touch Somoke testing

No.	Test	Expected Result	Actual Result
TC-0	**Device is able to recognize microSD cards** 1. Insert a new microSD	1. A notification about a new sd card appears on Device	PASS
TC-1	**MicroSD card can be formatted** 1. Insert a new microSD 2. Open External drives 3. Tap format button	1. A notification about a new sd card appears on Device 2. MicroSD card is listed 3. MicroSD card can be formatted	PASS
TC-2	**MicroSD card can be safely removed** 1. Insert a new microSD 2. Open External drives 3. Tap safely remove button	1. A notification about a new sd card appears on Device 2. MicroSD card is listed 3. MicroSD card can be safely removed	WONKY – Button needs to be tapped twice
TC-3	**Storage can be accessed from a computer** 1. Insert a new microSD 2. Connect device to a computer	1. A notification about a new sd card appears on Device 2. The computer can access both internal storage and ca	FAIL

operating system runs on 78 different devices

Use Case: UBports Installer



UBports Installer (0.9.1-beta)

Welcome to the UBports Installer

We will walk you through the installation process. Don't worry, it's easy!

With developer mode enabled, connect your phone, tablet, or smartwatch to the computer. Your device should be detected automatically.

If your device is not detected automatically, you can select it manually to proceed. Please note that the UBports Installer will only work on supported devices.

Waiting for device....

Please connect your device with a USB cable

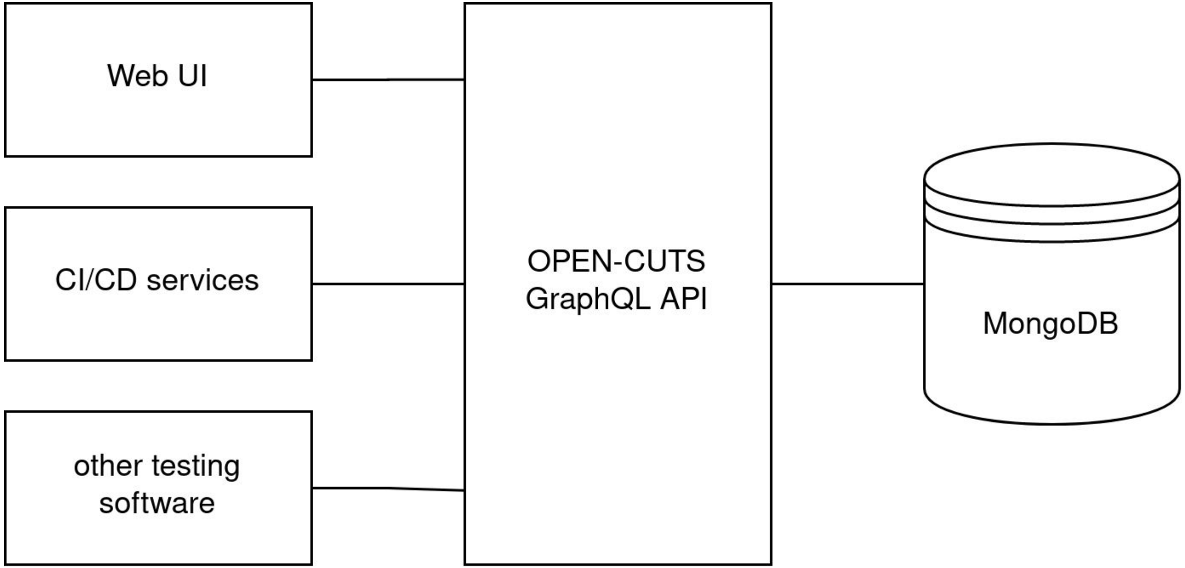
graphical installation tool

78 devices

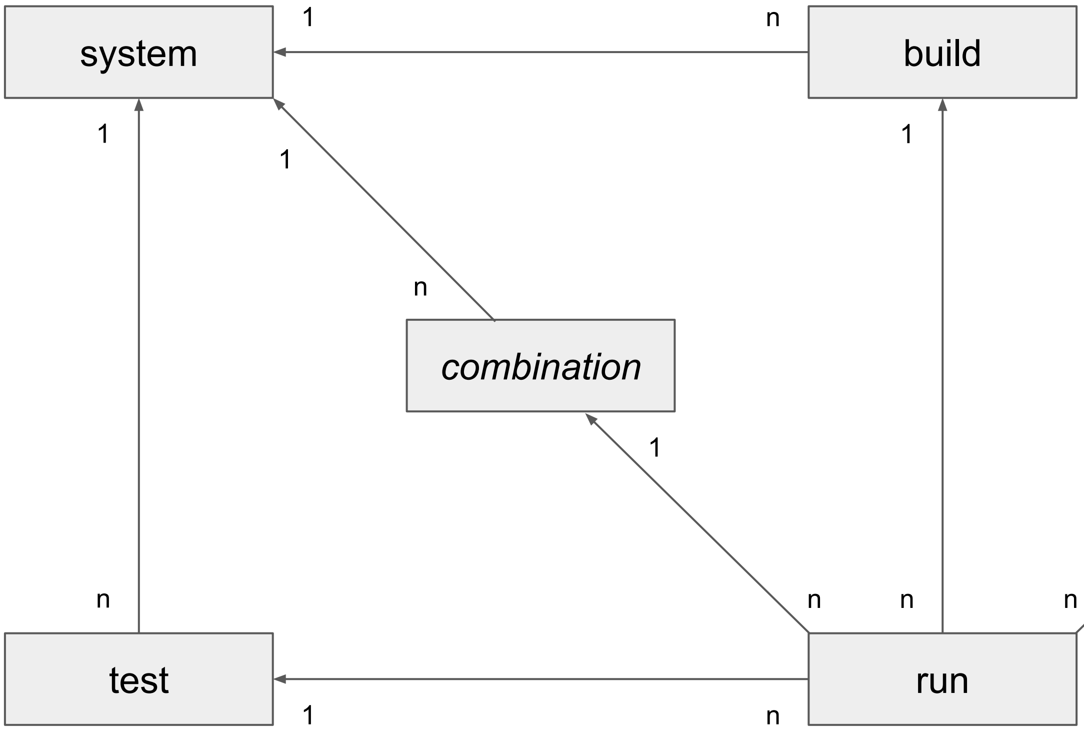
3 operating systems

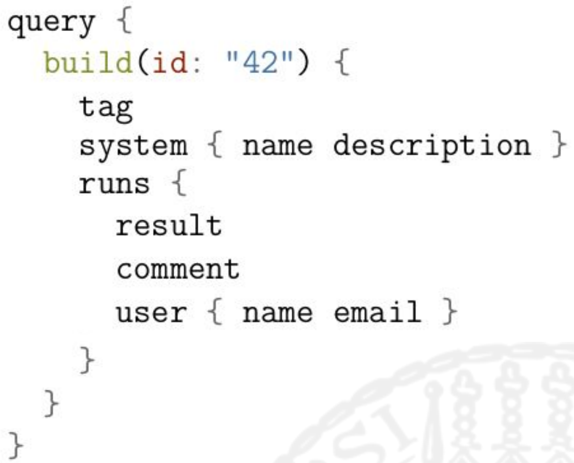
different packages available

OPEN Crowdsourced User-Testing Suite



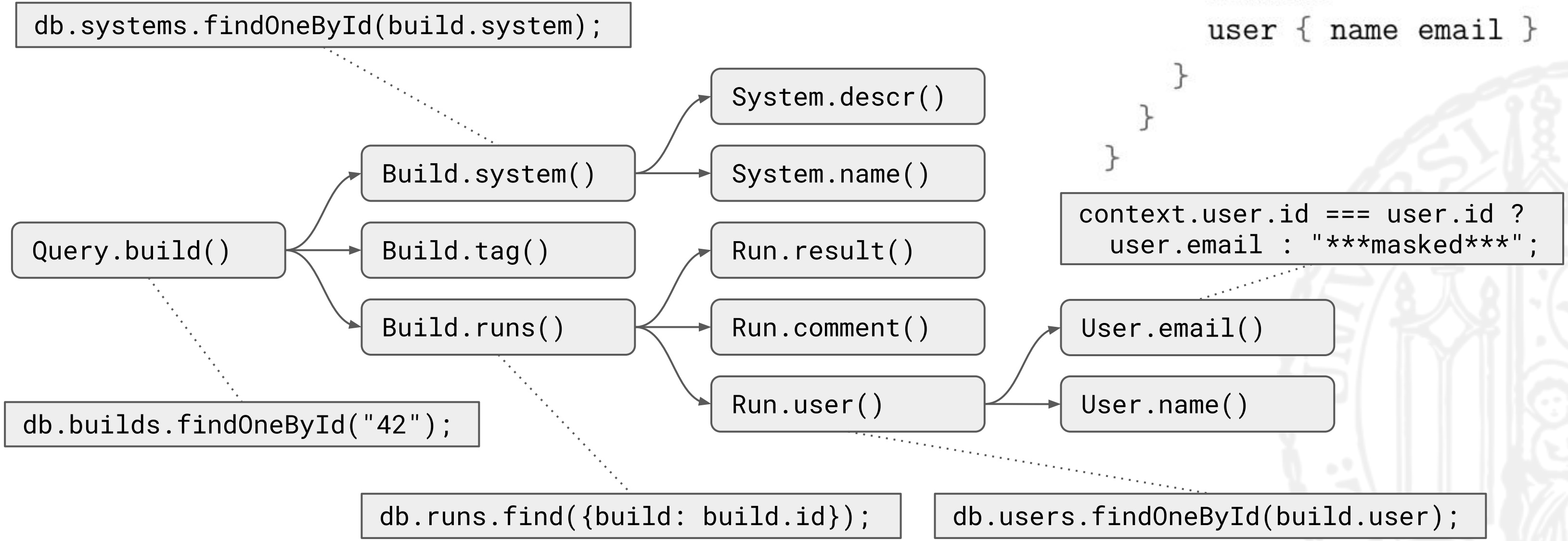
Implementation: Graph Structure




```
query {
  build(id: "42") {
    tag
    system { name description }
    runs {
      result
      comment
      user { name email }
    }
  }
}
```

Implementation: Query Resolution

```
query {
  build(id: "42") {
    tag
    system { name description }
    runs {
      result
      comment
      user { name email }
    }
  }
}
```



```
db.systems.findOneById(build.system);
```

```
Query.build()
Build.system()
Build.tag()
Build.runs()
System.descr()
System.name()
Run.result()
Run.comment()
Run.user()
User.email()
User.name()
```

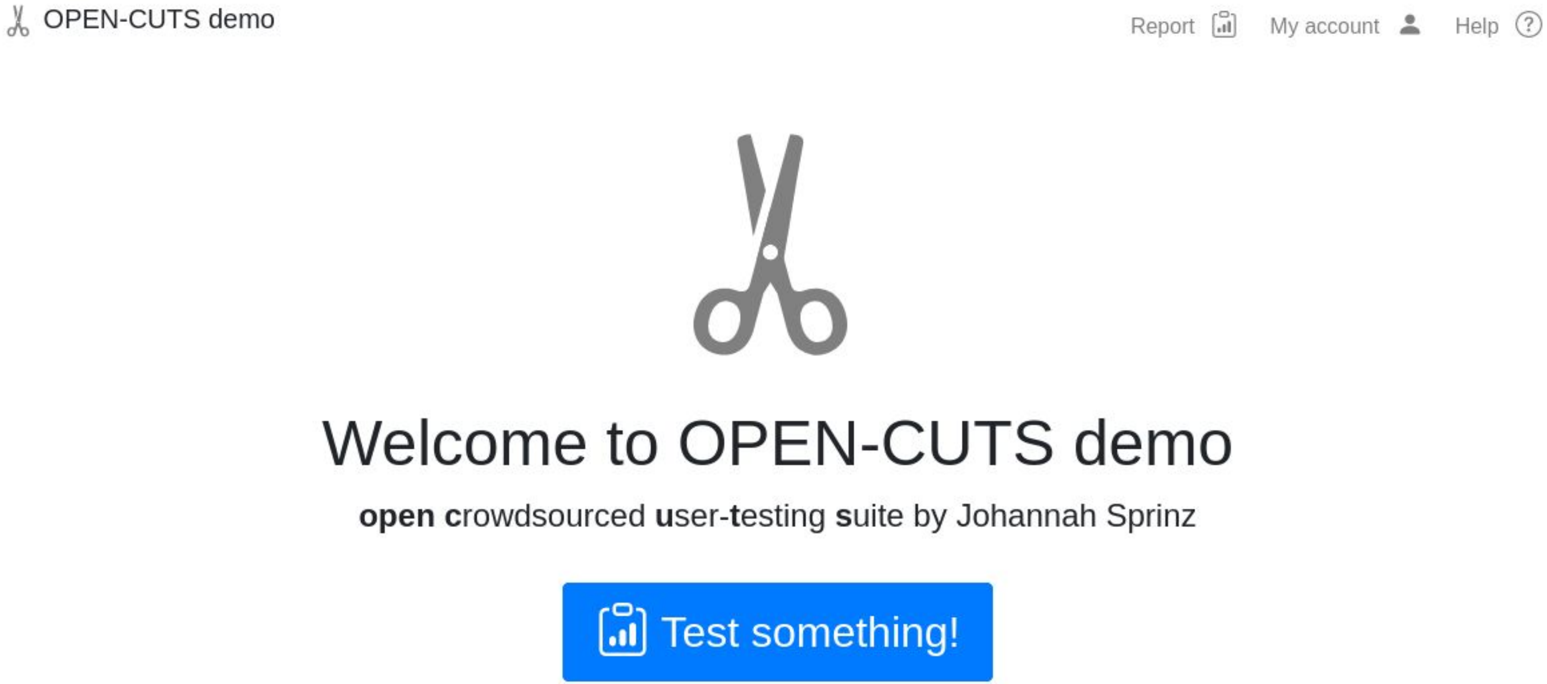
```
context.user.id === user.id ?
  user.email : "***masked***";
```

```
db.builds.findOneById("42");
```

```
db.runs.find({build: build.id});
```

```
db.users.findOneById(build.user);
```

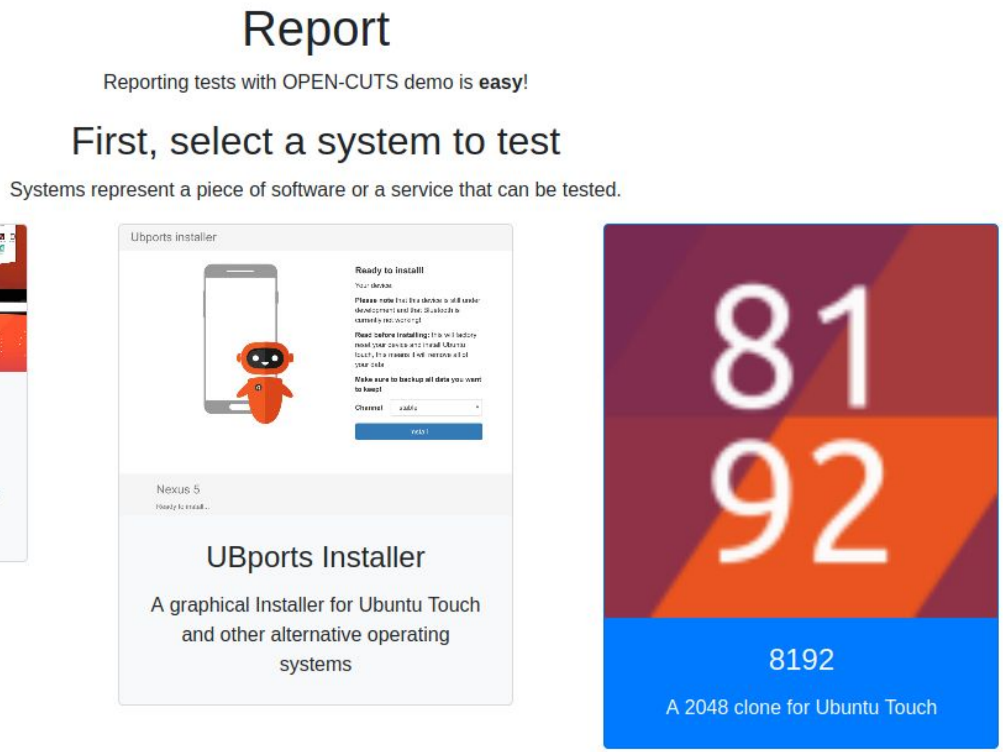
Demo: Landing Page



Welcome to OPEN-CUTS demo

open **c**rowdsourced **u**ser-**t**esting **s**uite by Johannah Sprinz

Demo: System Selection



Report

Reporting tests with OPEN-CUTS demo is **easy**!

First, select a system to test

Systems represent a piece of software or a service that can be tested.

Ubuntu Touch

The worlds greatest mobile operating system, brought to you by the UBports Foundation!

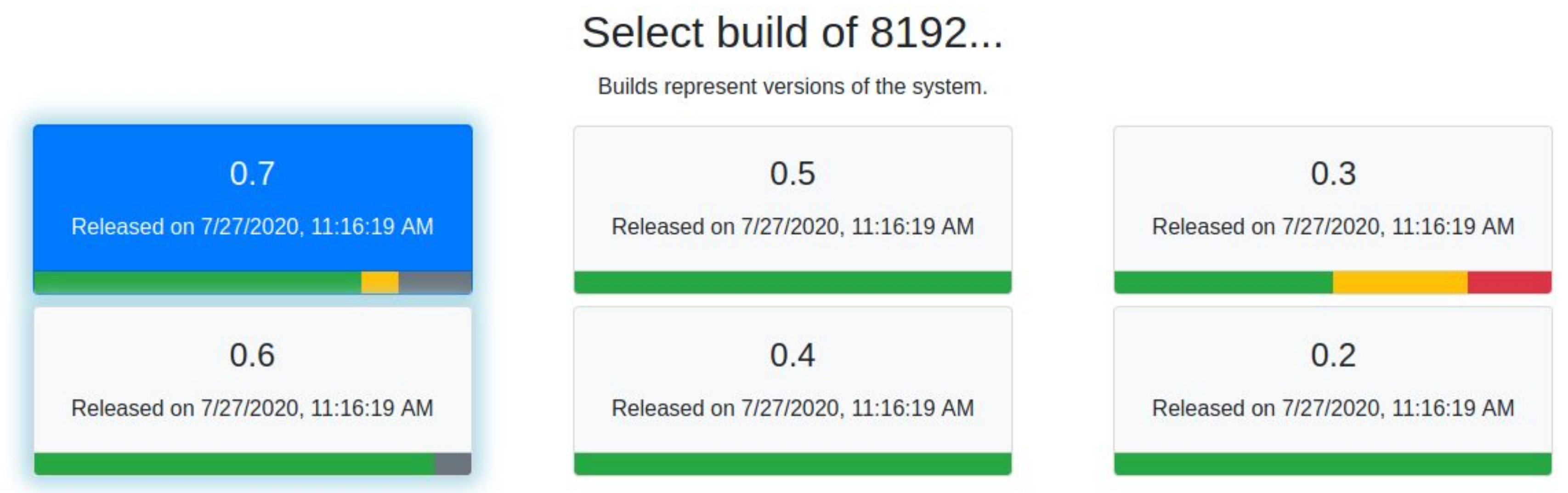
UBports Installer

A graphical Installer for Ubuntu Touch and other alternative operating systems

8192

A 2048 clone for Ubuntu Touch

Demo: Build Selection



Select build of 8192...

Builds represent versions of the system.

0.7

Released on 7/27/2020, 11:16:19 AM

0.5

Released on 7/27/2020, 11:16:19 AM

0.3

Released on 7/27/2020, 11:16:19 AM

0.6

Released on 7/27/2020, 11:16:19 AM

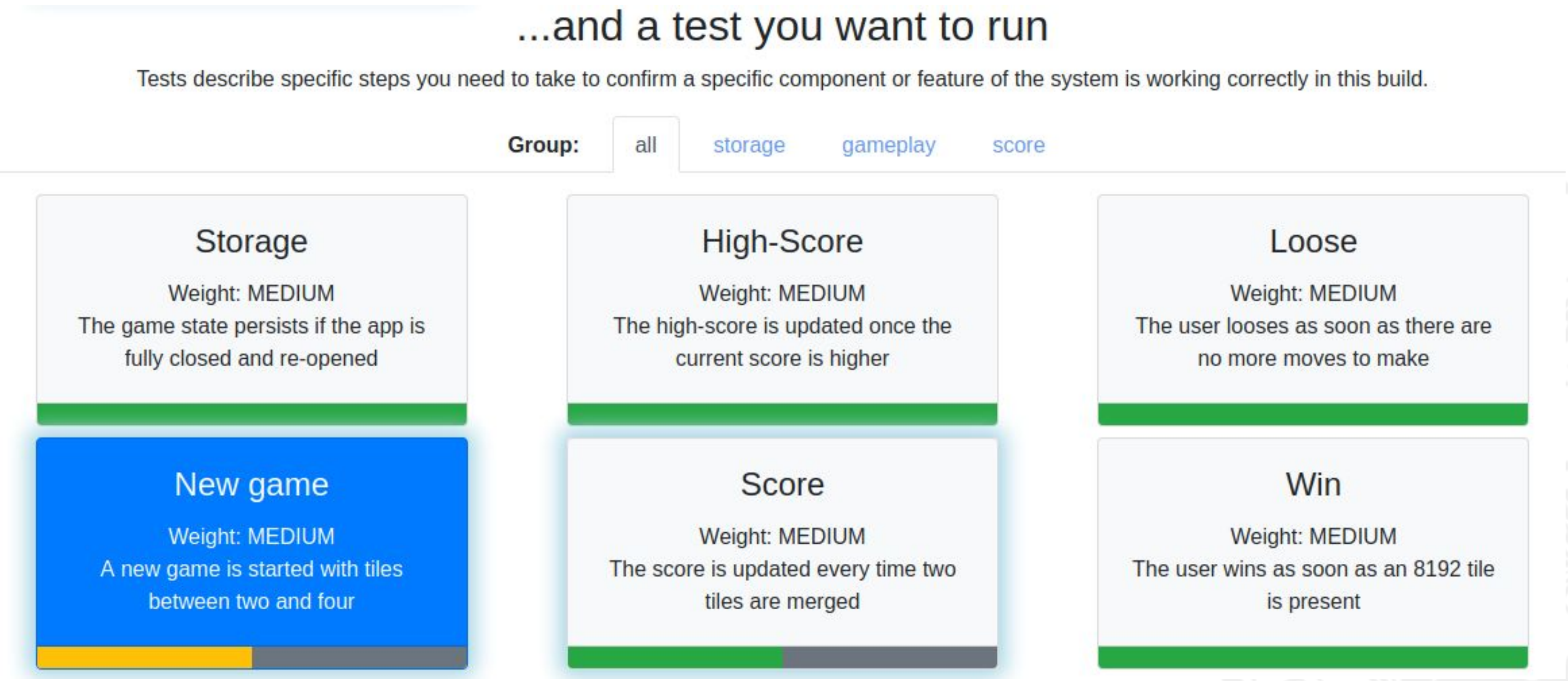
0.4

Released on 7/27/2020, 11:16:19 AM

0.2

Released on 7/27/2020, 11:16:19 AM

Demo: Test Selection



...and a test you want to run

Tests describe specific steps you need to take to confirm a specific component or feature of the system is working correctly in this build.

Storage

Weight: MEDIUM

The game state persists if the app is fully closed and re-opened

High-Score

Weight: MEDIUM

The high-score is updated once the current score is higher

Loose

Weight: MEDIUM

The user looses as soon as there are no more moves to make

New game

Weight: MEDIUM

A new game is started with tiles between two and four

Score

Weight: MEDIUM

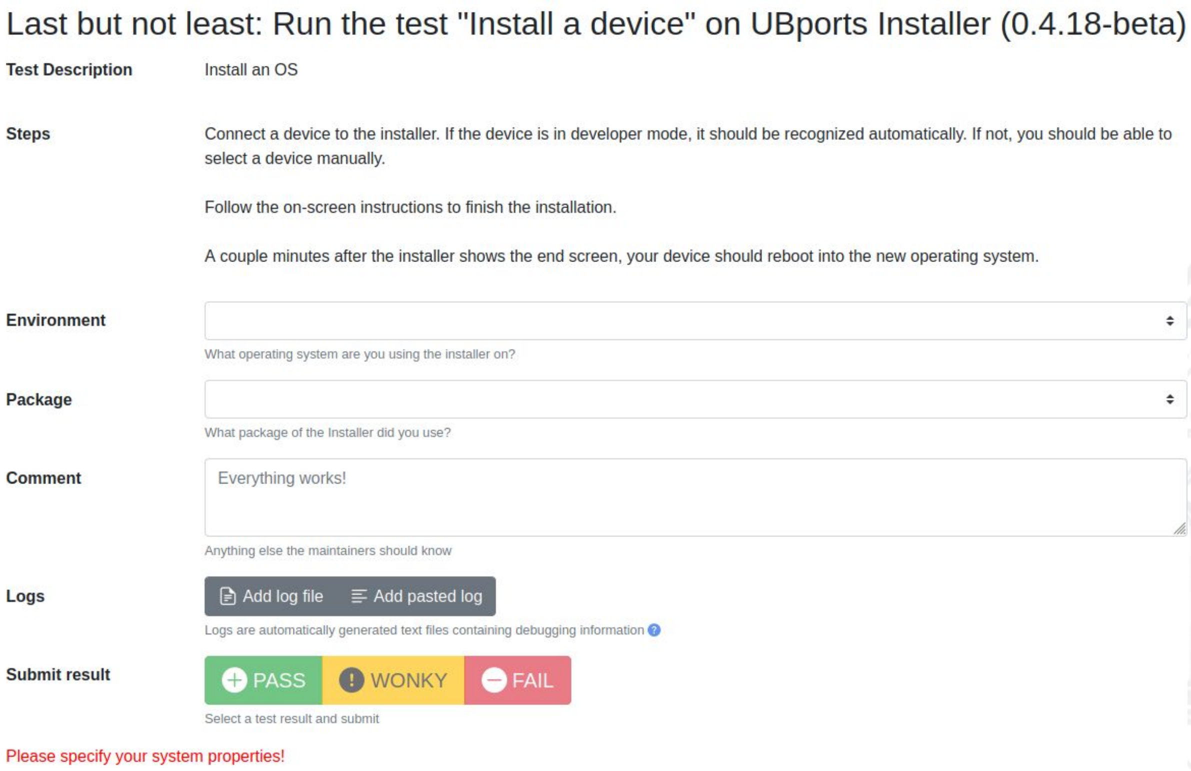
The score is updated every time two tiles are merged

Win

Weight: MEDIUM

The user wins as soon as an 8192 tile is present

Demo: Test Submission



Last but not least: Run the test "Install a device" on UBports Installer (0.4.18-beta)

Test Description Install an OS

Steps Connect a device to the installer. If the device is in developer mode, it should be recognized automatically. If not, you should be able to select a device manually.

Follow the on-screen instructions to finish the installation.

A couple minutes after the installer shows the end screen, your device should reboot into the new operating system.

Environment

What operating system are you using the installer on?

Package

What package of the Installer did you use?

Comment Everything works!

Anything else the maintainers should know

Logs

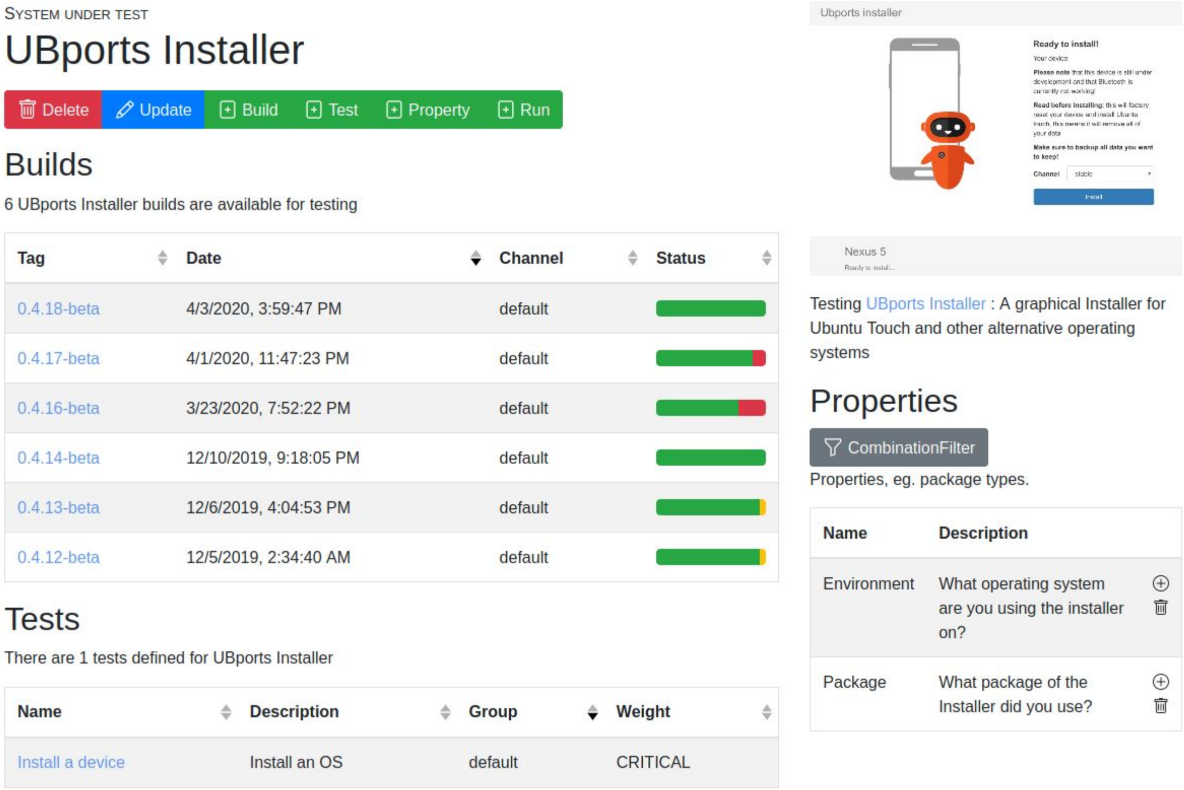
Logs are automatically generated text files containing debugging information

Submit result

Select a test result and submit

Please specify your system properties!

Demo: System View



SYSTEM UNDER TEST

UBports Installer

Delete | Update | Build | Test | Property | Run

Builds

6 UBports Installer builds are available for testing

Tag	Date	Channel	Status
0.4.18-beta	4/3/2020, 3:59:47 PM	default	🟩
0.4.17-beta	4/1/2020, 11:47:23 PM	default	🟩🟥
0.4.16-beta	3/23/2020, 7:52:22 PM	default	🟩🟥
0.4.14-beta	12/10/2019, 9:18:05 PM	default	🟩
0.4.13-beta	12/6/2019, 4:04:53 PM	default	🟩🟨
0.4.12-beta	12/5/2019, 2:34:40 AM	default	🟩🟨

Tests

There are 1 tests defined for UBports Installer

Name	Description	Group	Weight
Install a device	Install an OS	default	CRITICAL

Ubports installer

Ready to install!

Your device:

Please note that this device is still under development and that Bluetooth is currently not working!

Read before installing: this will factory reset your device and install Ubuntu touch, this means it will remove all of your data

Make sure to backup all data you want to keep!

Channel stable

Install

Nexus 5

Ready to install...

Testing UBports Installer : A graphical Installer for Ubuntu Touch and other alternative operating systems

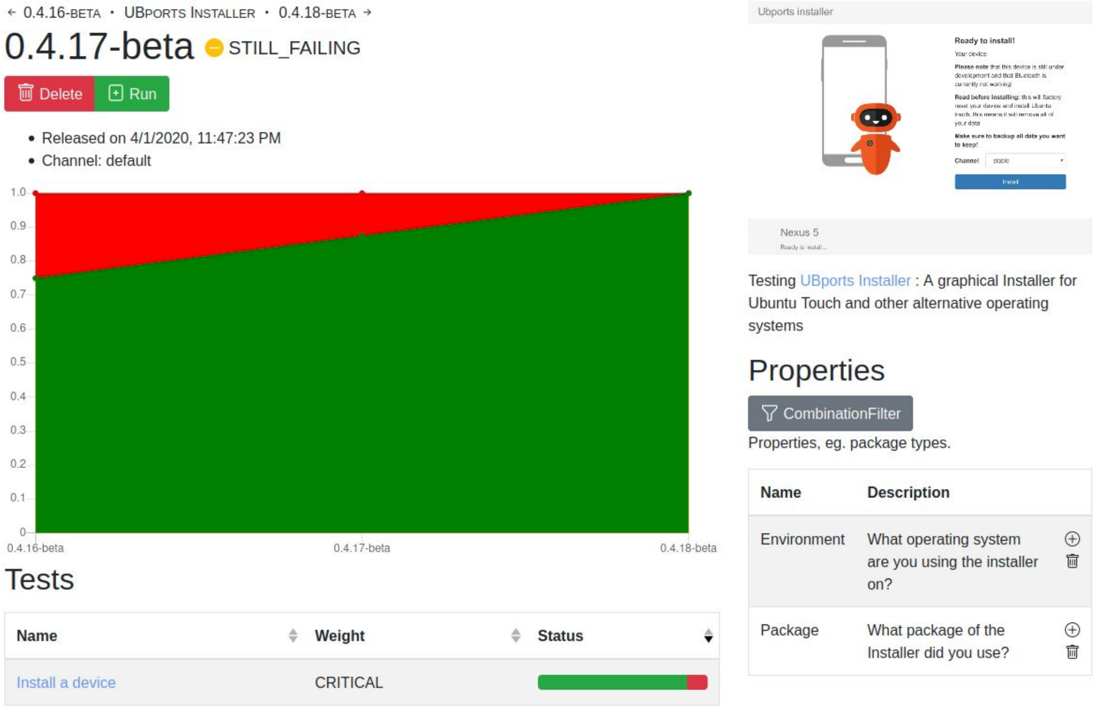
Properties

CombinationFilter

Properties, eg. package types.

Name	Description
Environment	What operating system are you using the installer on?
Package	What package of the Installer did you use?

Demo: Build View



← 0.4.16-BETA • UBPORTS INSTALLER • 0.4.18-BETA →

0.4.17-beta STILL_FAILING

Delete Run

- Released on 4/1/2020, 11:47:23 PM
- Channel: default

Tests

Name	Weight	Status
Install a device	CRITICAL	🟩🟥

Ubports installer

Ready to install!

Your device:

Please note that this device is still under development and that Bluetooth is currently not working!

Read before installing: this will factory reset your device and install Ubuntu touch, this means it will remove all of your data

Make sure to backup all data you want to keep!

Channel: stable

Install

Nexus 5

Ready to install...

Testing UBports Installer : A graphical Installer for Ubuntu Touch and other alternative operating systems

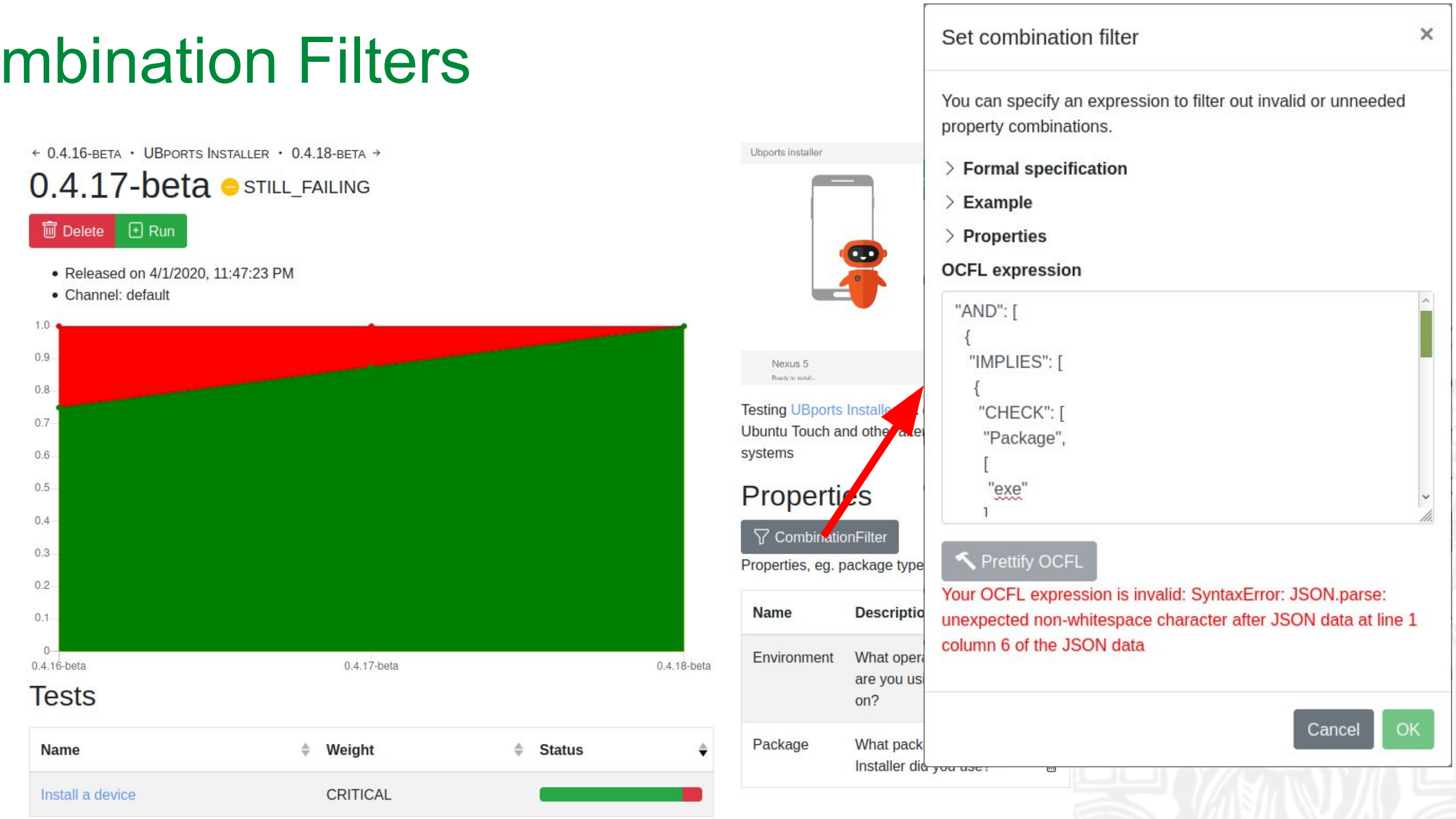
Properties

CombinationFilter

Properties, eg. package types.

Name	Description
Environment	What operating system are you using the installer on?
Package	What package of the Installer did you use?

Demo: Combination Filters



← 0.4.16-BETA • UBPORTS INSTALLER • 0.4.18-BETA →

0.4.17-beta STILL_FAILING

- Released on 4/1/2020, 11:47:23 PM
- Channel: default

Tests

Name	Weight	Status
Install a device	CRITICAL	

Testing UBports Installe... Ubuntu Touch and othe... systems

Properties

Properties, eg. package type

Name	Descriptio
Environment	What opera are you usi on?
Package	What pack Installer did you use?

Set combination filter

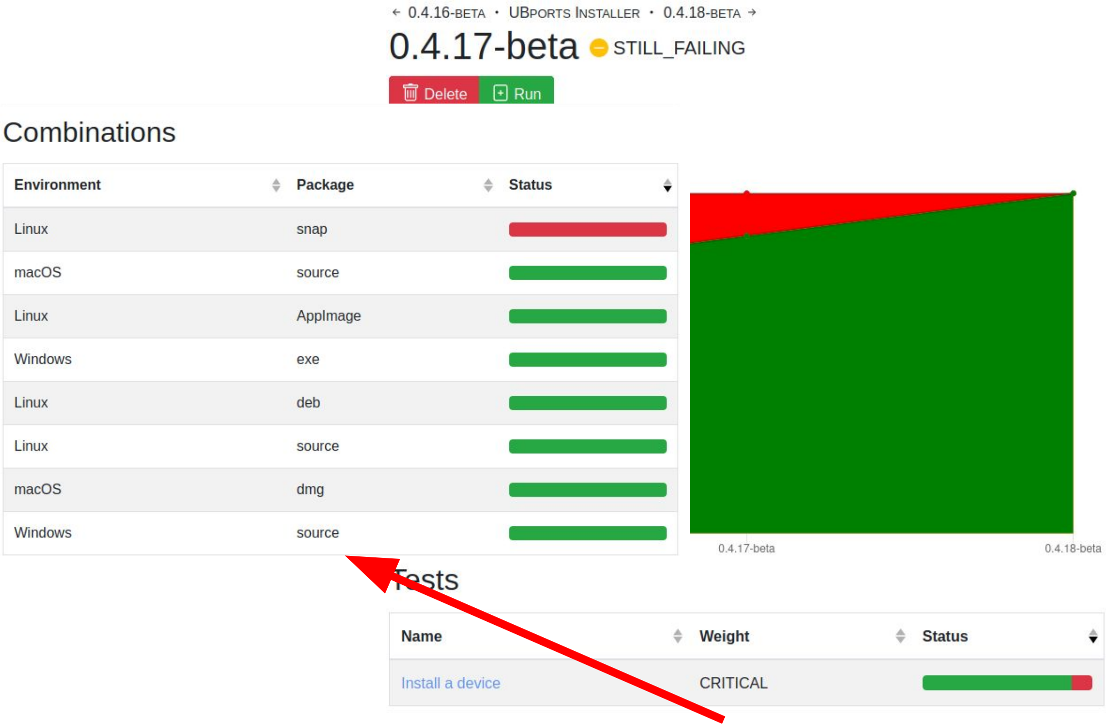
You can specify an expression to filter out invalid or unneeded property combinations.

OCFL expression

```
"AND": [
 {
  "IMPLIES": [
   {
    "CHECK": [
     "Package",
     [
      "exe"
      ]
```

Your OCFL expression is invalid: SyntaxError: JSON.parse: unexpected non-whitespace character after JSON data at line 1 column 6 of the JSON data

Demo: Property Combinations



← 0.4.16-BETA • UBPORTS INSTALLER • 0.4.18-BETA →

0.4.17-beta STILL_FAILING

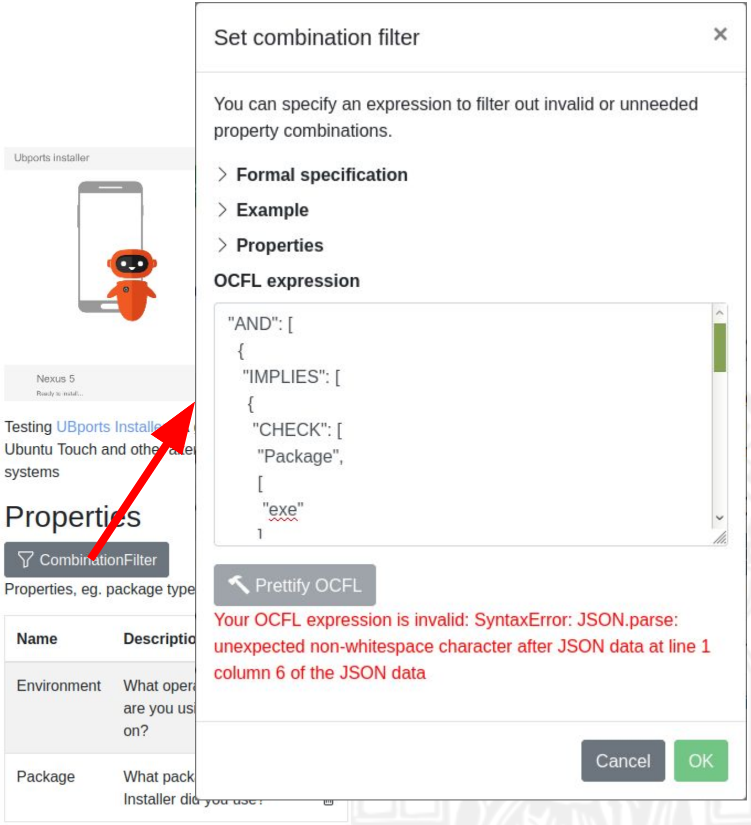
Delete Run

Combinations

Environment	Package	Status
Linux	snap	🟥
macOS	source	🟩
Linux	AppImage	🟩
Windows	exe	🟩
Linux	deb	🟩
Linux	source	🟩
macOS	dmg	🟩
Windows	source	🟩

Tests

Name	Weight	Status
Install a device	CRITICAL	🟩🟥



Testing UBports Installe... Ubuntu Touch and othe... systems

Properties

CombinationFilter

Properties, eg. package type

Name	Descriptio
Environment	What opera are you us on?
Package	What pack Installer did you use?

Set combination filter ×

You can specify an expression to filter out invalid or unneeded property combinations.

> Formal specification

> Example

> Properties

OCFL expression

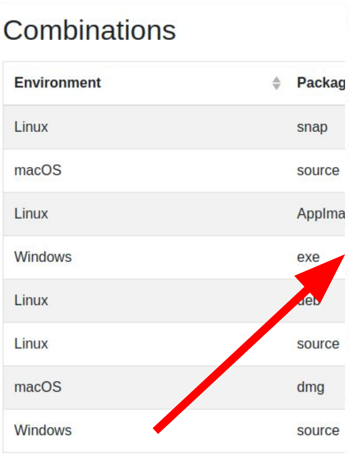
```
"AND": [
 {
  "IMPLIES": [
   {
    "CHECK": [
     "Package",
     [
      "exe"
     ]
```

Prettify OCFL

Your OCFL expression is invalid: SyntaxError: JSON.parse: unexpected non-whitespace character after JSON data at line 1 column 6 of the JSON data

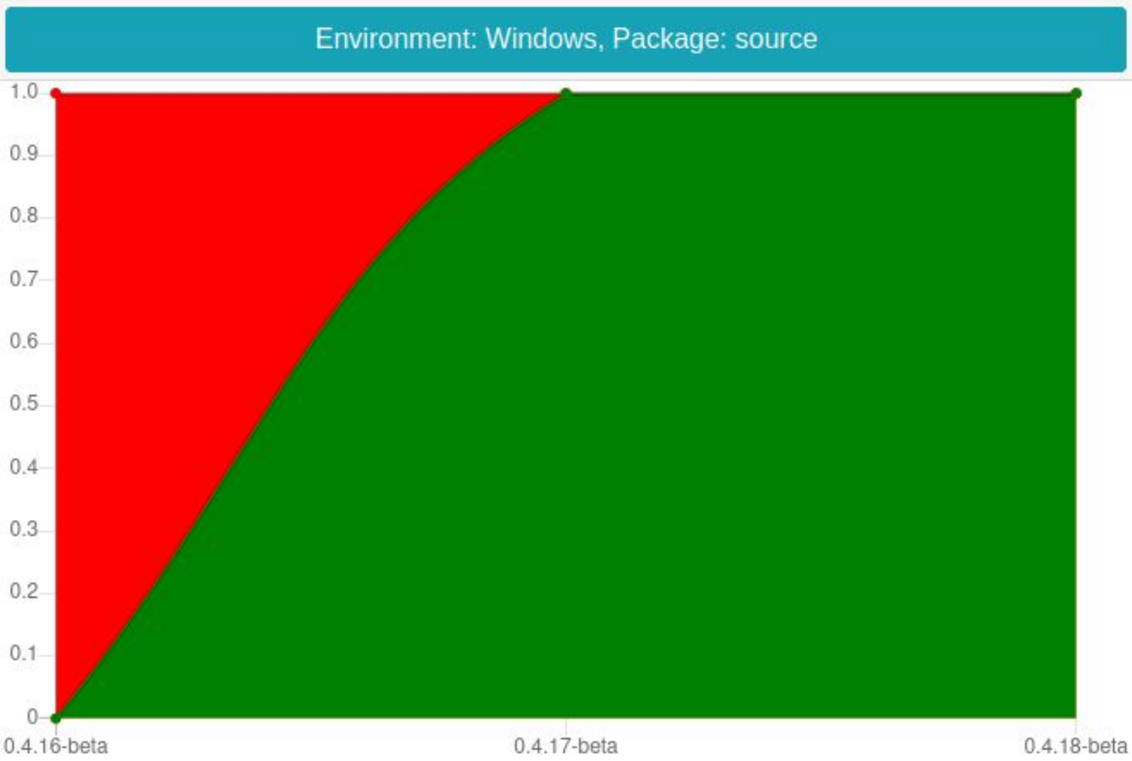
Cancel OK

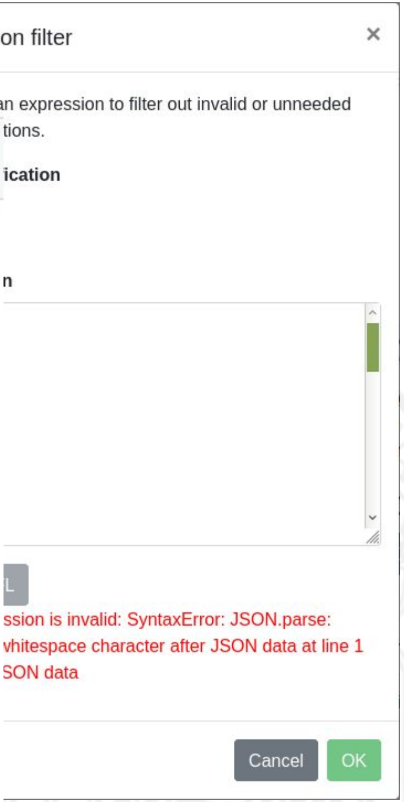
Demo: Build-Test-Property Tensor Slicing



Combinations

Environment	Packag
Linux	snap
macOS	source
Linux	Applma
Windows	exe
Linux	deb
Linux	source
macOS	dmg
Windows	source





Set combination filter

You can specify an expression to filter out invalid or unneeded
tions.

ication

n

ssion is invalid: SyntaxError: JSON.parse:
hitespace character after JSON data at line 1
SON data

Evaluation

1. Is there any initial evidence of untapped potential for facilitating QA under the OSSD by having inexperienced community members contribute formalized user testing data to a human computation platform?

2. Can OPEN-CUTS be efficiently used by the inexperienced members of an open source community to conduct and report formalized user tests?

3. Is OPEN-CUTS equipped to provide meaningful insights to open source developers in their daily work?

Usability Study

5 users and 4 developers, recruited from the UBports Community

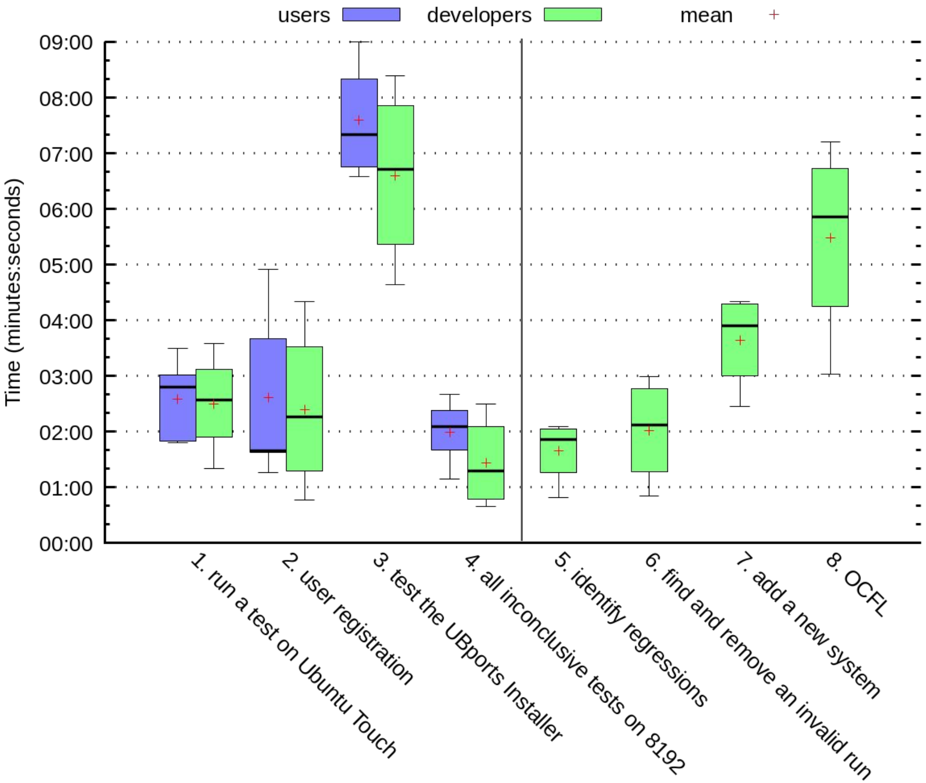
all subjects receive an identical database and have to complete tasks

tasks represent how a subject would use OPEN-CUTS in the real world

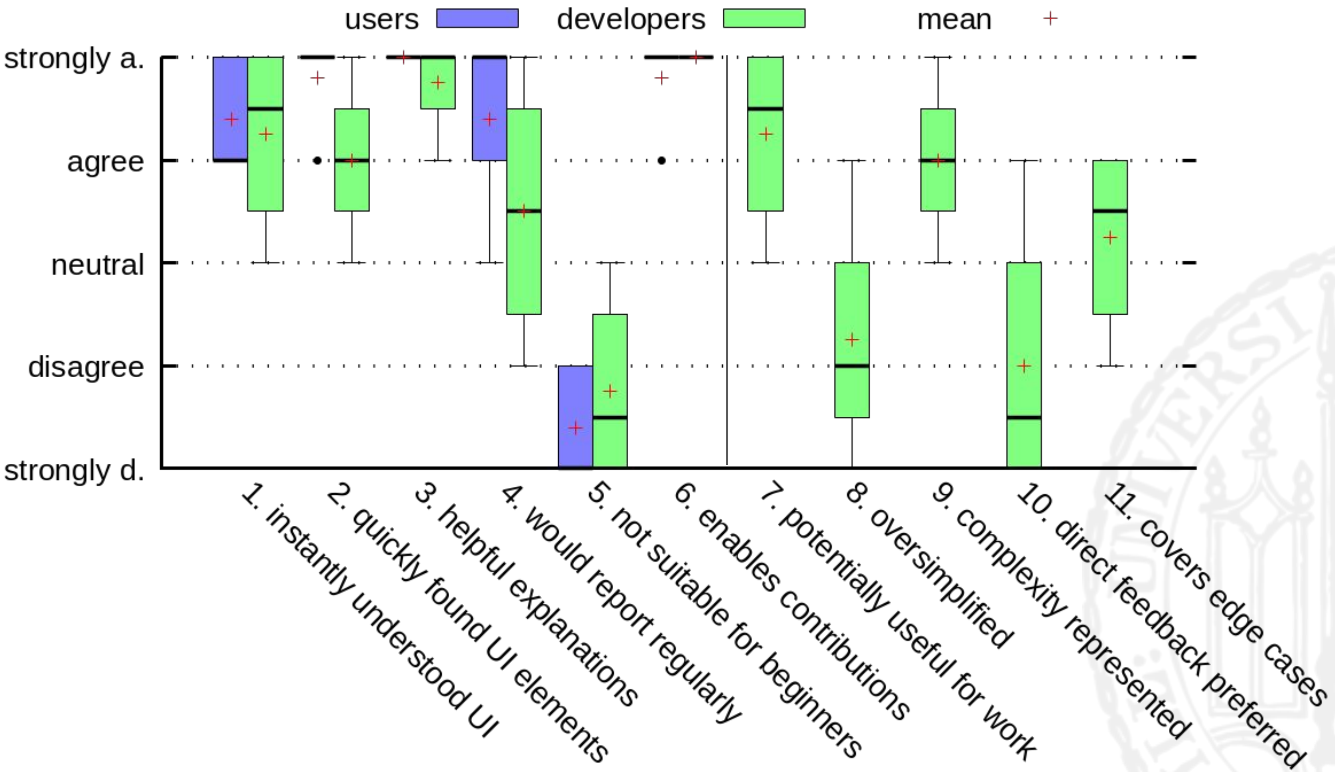
task completion-times and concurrent think-aloud comments are recorded

retrospective probing questionnaire filled in after trial

Results


users
developers
mean
Time (minutes:seconds)
09:00
08:00
07:00
06:00
05:00
04:00
03:00
02:00
01:00
00:00
1. run a test on Ubuntu Touch
2. user registration
3. test the UBports Installer
4. all inconclusive tests on 8192
5. identify regressions
6. find and remove an invalid run
7. add a new system
8. OCFL

Results



users
developers
mean
strongly a.
agree
neutral
disagree
strongly d.
1. instantly understood UI
2. quickly found UI elements
3. helpful explanations
4. would report regularly
5. not suitable for beginners
6. enables contributions
7. potentially useful for work
8. oversimplified
9. complexity represented
10. direct feedback preferred
11. covers edge cases

Future Work: Development

gamification, games with a purpose

automated reporting of real-world testing data (proof-of-concept already live!)

more powerful combination logic

improved data analysis and presentation, more caching

integration with CI/CD pipelines and bug trackers

interdependent tests

moderation features

support version-controlled (e.g., git) configuration files

Future Work: Long-Term Study

evaluate long-term impact in the UBports Community (ubports.open-cuts.org)

What is the technical background of the users using OPEN-CUTS to run tests? Have they contributed before?

How does the introduction of OPEN-CUTS impact overall contribution behavior? In particular, are other low-skilled areas of contribution (e.g., translations, marketing, user discussion, bug tracker activity) affected?

What is the contributor retention rate? I.e., how many contributors stay on as active testers, how many merely occasionally contribute test results, how many lose interest?

Is the introduction of OPEN-CUTS considered helpful by a) the core development team, b) the contributing developers, and c) the bug reporters?

Future Work: Other Research

use OPEN-CUTS to support other research and development endeavours

Example: Expanding the UBports Installer to support other operating systems

- it is not feasible for one developer to have access to hundreds of devices
- crowdsourced testing might help
- if installation configurations are crowdsourced, the testing should be as well

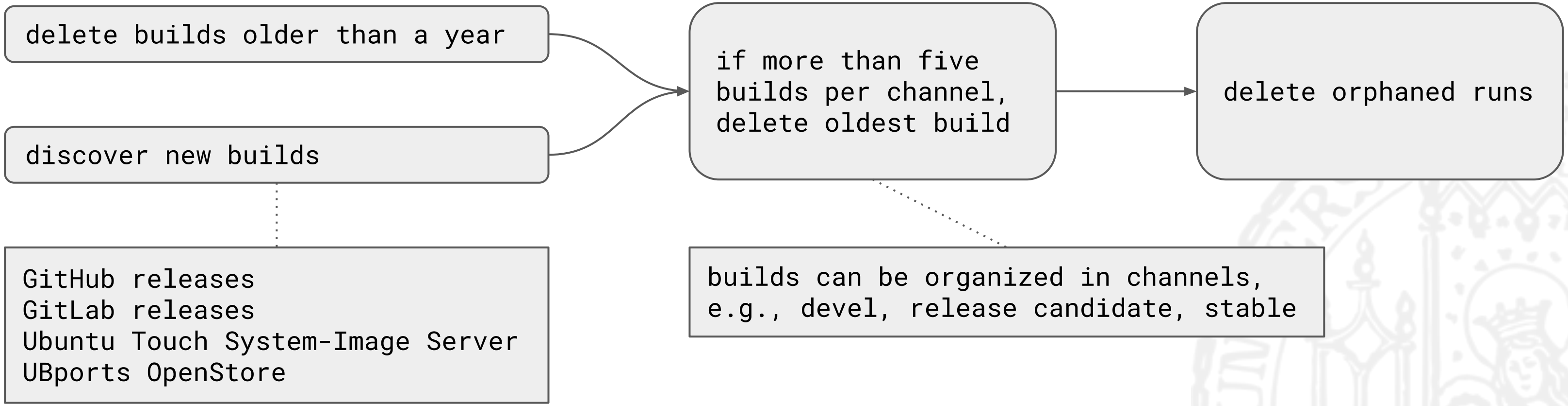
Thanks for listening!

Questions, comments, remarks?

J. Sprinz, “Leveraging Human Computation for Quality Assurance in Open Source Communities,” LMU Munich, Bachelor Thesis, Jan. 2022, doi: 10.5282/ubm/epub.91046

Future research will be published on spri.nz and open-cuts.org

Implementation: Housekeeping and Build Discovery



delete builds older than a year

discover new builds

GitHub releases
GitLab releases
Ubuntu Touch System-Image Server
UBports OpenStore

if more than five builds per channel, delete oldest build

builds can be organized in channels, e.g., devel, release candidate, stable

delete orphaned runs

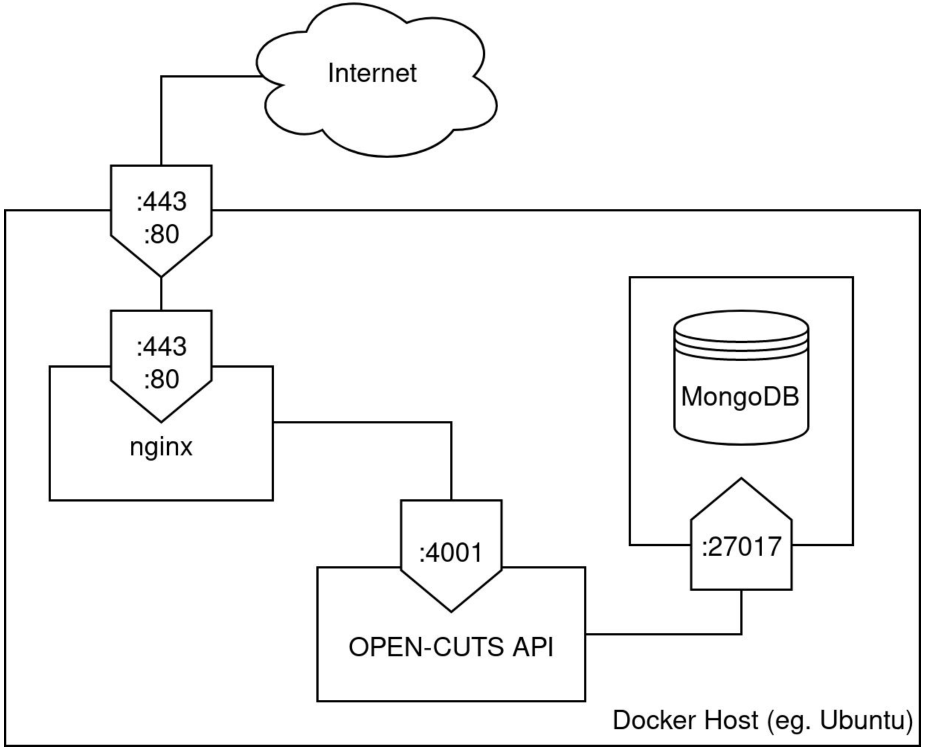
OCFL: OPEN-CUTS Combination Filtering Language

```
// First-order logic based on JSON
exp       : <check> | <unaryexp> | <binaryexp> | <chainexp>
check     : { CHECK: [<stringterminal>, [<stringterminal>, ...]] }
unaryexp  : { <unaryop>: [<exp>] }
unaryop   : NOT
binaryexp : { <binaryop>: [<exp>, <exp>] }
binaryop  : IMPLIES
chainexp  : { <chainop>: [<exp>, ...] }
chainop   : AND | OR
```

OCFL Example

```
{"AND": [
  {"IMPLIES": [
    { "CHECK": ["Package", ["exe"]] },
    { "CHECK": ["Environment", ["Windows"]] }] },
  {"IMPLIES": [
    { "CHECK": ["Package", ["dmg"]] },
    { "CHECK": ["Environment", ["macOS"]] }] },
  {"IMPLIES": [
    { "CHECK": ["Package", ["snap", "AppImage", "deb"]]},
    { "CHECK": ["Environment", ["Linux"]] }]
  }] }
```

Implementation: Dockerized Microservices



Literature

M. Aberdour, “Achieving Quality in Open-Source Software,” IEEE Software, vol. 24, no. 1, pp. 58–64, Jan. 2007, doi: 10.1109/MS.2007.2

S. S. Bahamdain, “Open Source Software (OSS) Quality Assurance: A Survey Paper,” Procedia Computer Science, vol. 56, pp. 459–464, Jan. 2015, doi: 10.1016/j.procs.2015.07.236

D. Kulesz and I. Bogicevic, “SystemTestPortal - A Web-Application for managing Manual System Tests,” Nov. 2017, doi: 10.5446/41642

E. Law and L. von Ahn, Human computation. Morgan & Claypool, 2011. doi: 10.2200/S00371ED1V01Y201107AIM013

R. Mühlhoff, “Human-aided artificial intelligence: Or, how to run large computations in human brains? Toward a media sociology of machine learning:” New Media & Society, pp. 1868–1884, Nov. 2019, doi: 10.1177/1461444819885334

E. S. Raymond, “The cathedral and the bazaar,” First Monday, vol. 3, no. 2, Mar. 1998, doi: 10.5210/fm.v3i2.578

G. G. Schulmeyer, Ed., Handbook of software quality assurance, 4th ed. Boston: Artech House, 2008

J. Sprinz, “Leveraging Human Computation for Quality Assurance in Open Source Communities,” LMU Munich, Bachelor Thesis, Jan. 2022, doi: 10.5282/ubm/epub.91046

J. Sprinz, “A collaborative effort to create a user-friendly installer for different mobile operating systems,” submitted to the Journal of Brief Ideas, Feb. 2022, Available: https://beta.briefideas.org/ideas/3f22fbc27fb6dcb3c4c33603ef597f9b

D. Wahyudin, A. Schatten, D. Winkler, and S. Biffl, “Aspects of Software Quality Assurance in Open Source Software Projects: Two Case Studies from Apache Project,” Aug. 2007, pp. 229–236. doi: 10.1109/EUROMICRO.2007.19