

Fake Data Generators

Ruben Jacobo, Waylon Ho

Project Sponsors: Carlos Paradis and Rick Kazman

ICS 496

Fall 2023

<https://github.com/sauluh/kaiaulu>

Introduction

What is Kaiaulu?

Open Source Package designed to provide a means for analyzing a variety of software development stacks (git log, mailing list, issue trackers, source code, etc.) Kaiaulu API returns tables, which allows for simple data exploration and integration from various sources.

Methodology

Leveraged **Agile** methodologies to:

- Streamline our software development process
- Emphasize close collaboration
- Adaptively plan out new implementation
- Iterative delivery on deliverables

Technology Used

- RStudio IDE and R Language
- Testthat
 - Unit Testing Capabilities
- Perceval
 - Third party data retrieval tool used in conjunction with parser functions

Fake Data Generator : : CHEAT SHEET

About

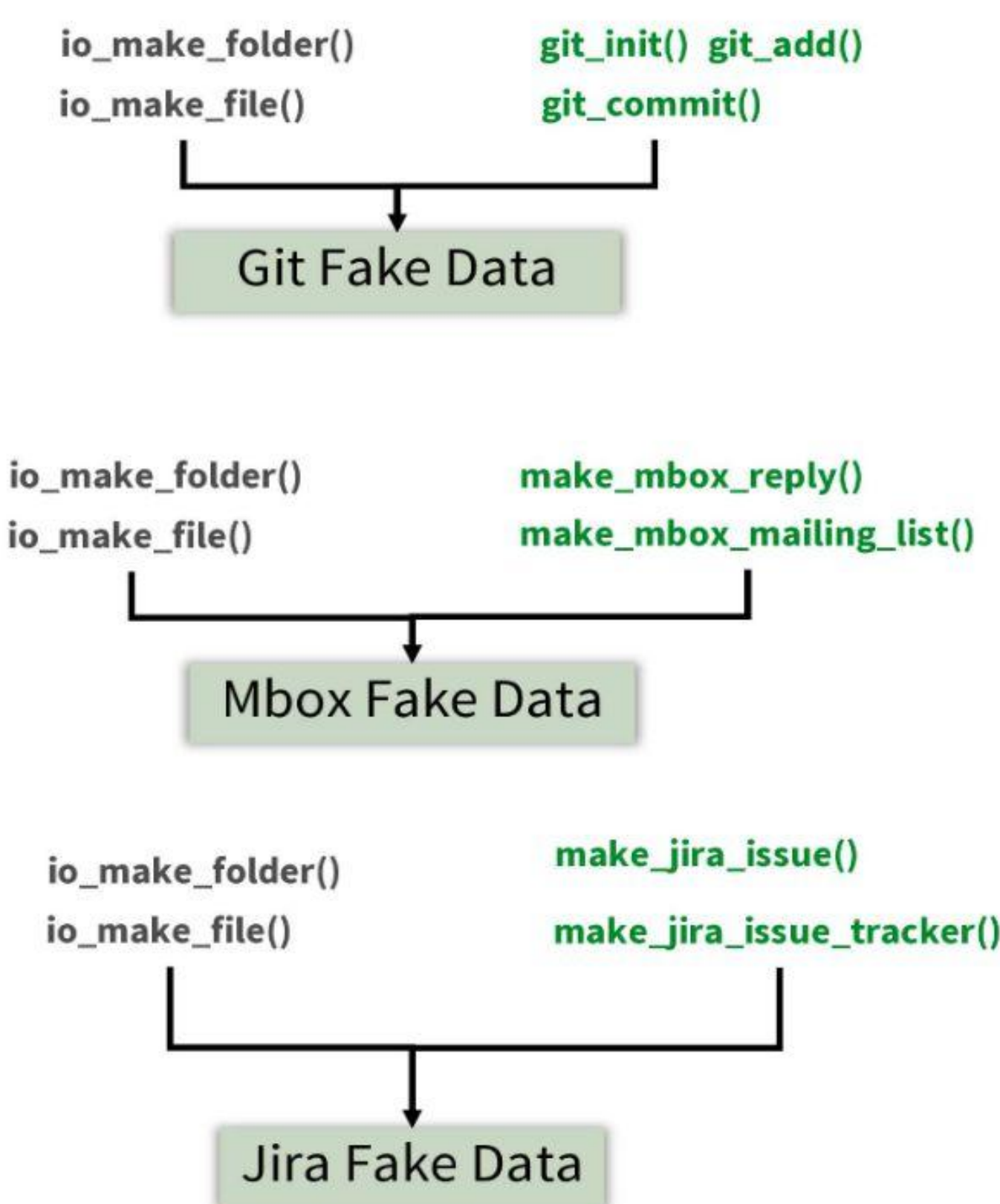
The fake data generators provide API to create minimal reproducible examples (MRE) capturing unusual project data to unit test Kaiaulu data parsers. They can also be used to compare equivalent analyses to other tools, and different versions of the same tool.

Fake Data Generator (git.R, mail.R, jira.R): APIs for Git, Mailing list, and Jira formats

Fake Examples (example.R): Minimal reproducible examples (MRE) generated on the fly building on Fake Generators

Unit Tests (test-*.R): Tests use MREs to test the behavior of functions

Fake Data Generator



Kaiaulu

Fake Examples

Writing Fake Examples

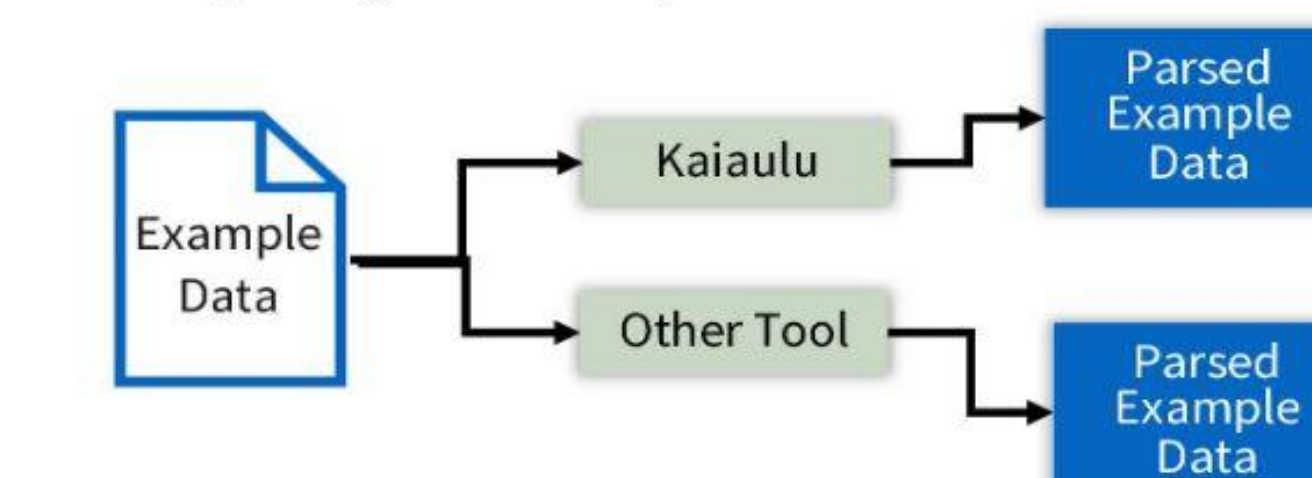
```
example_dif_branches <- function(...) {
```

```
  file_path <- io_make_file(..., body="print('Hello!')")
  git_init(...)
  git_add(file_path,...)
  git_commit(...)
  ...
  return(example_path)
}
```

Fake examples use the Fake Data Generator API to annotate (and version) specific details in datasets that may be overlooked in analysis. They can then be passed to other functions in Kaiaulu:

```
parse_gitlog(path=example_dif_branches(),
  tool="Perceval")
parse_gitlog(path=example_file_rename(),
  tool="Perceval")
```

Comparing Tools Outputs



Fake Data Examples can also be used to compare Kaiaulu output to other tools and papers.

Often, research publications will also include examples to illustrate a method. These can be expressed in Kaiaulu as example functions.

Unit Testing

Unit Testing Fake Examples

```
test_that("Only Master Branch Commits are parsed",{
  git_repo_path <- example_dif_branches(...)
  expect_equal(parse_gitlog(git_repo_path),2)
})
```

Unit tests can be combined with fake examples to evaluate parser functions, which require data to be evaluated.

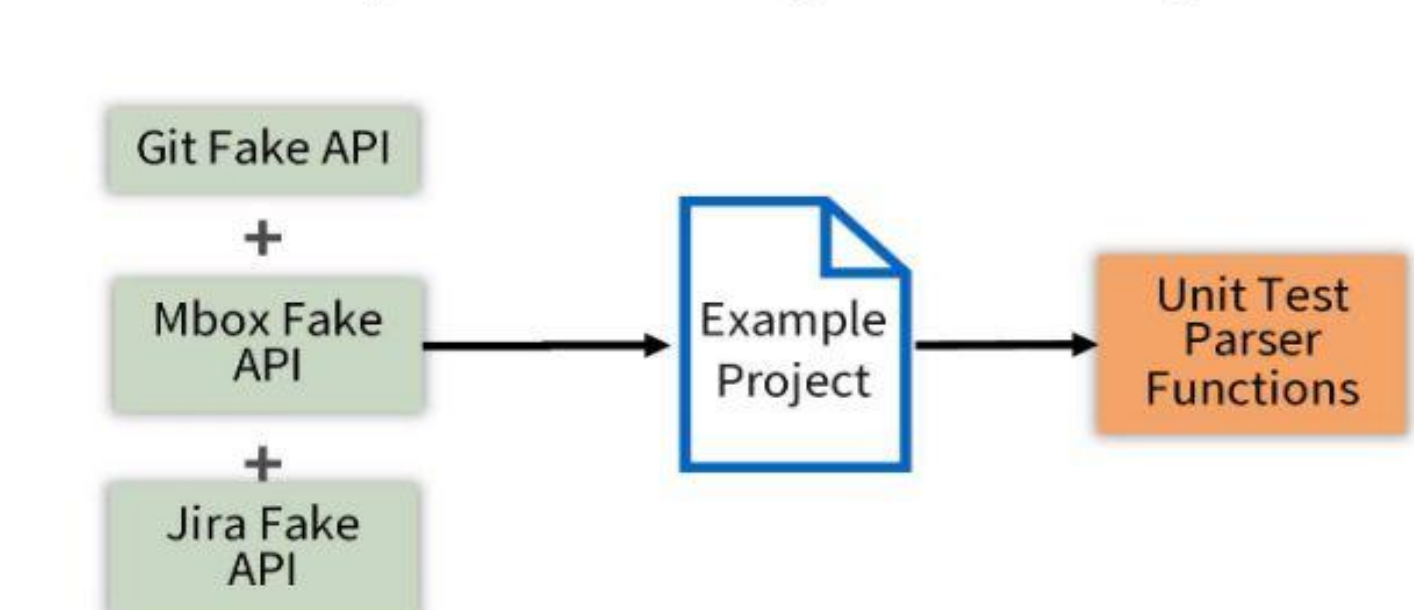
Third-Party Tool Behavior "Contract"

```
parse_gitlog(example_two_branches, tool="Perceval 1.0")
parse_gitlog(example_two_branches, tool="Perceval 2.0")
```

Some of Kaiaulu's functionality is borrowed from other tools. While new releases may introduce a change in analysis behavior, examples can be used to ensure output of interest remains consistent across versions.

Example datasets can also be used to compare different tools output.

Fake Ecosystems and Integration Testing



When combined, Fake Data Generators can be used to simulate a moment of interest in a project ecosystem. Analyses that utilize multiple sources can be sanity-checked alongside their code behavior.

Solution

Tasks Accomplished

- Developed and implemented specialized functions for generating fake data in three critical domains: Git, Mailbox, and Jira files.
- Created realistic yet artificial datasets that mimic real-world scenarios for testing and validation purposes.
- Enhanced the robustness of existing parser functions by subjecting them to a wide array of simulated real-world scenarios.
- Implemented an architecture of Fake Data creation that allows for a.) On-the-fly data generation b.) Examples that capture edge situations in real world dataset and c.) their associated unit tests.
- Crafted a succinct and visually intuitive one-page cheat sheet encapsulating key concepts and usage scenarios of the fake data generators.

Challenges

- Adapting to RStudio and the intricacies of the R language.
- Delved into Kaiaulu's existing codebase, ensuring a comprehensive understanding before making impactful contributions.
- Effectively communicating progress and challenges of weekly deliverables.

Learnings

- Mastered effective communication with our sponsor, translating requests seamlessly into code that aligns with their needs.
- Recognized the critical role of communication with team members and the sponsor/client, pivotal for project success within the specified timeline.

INFORMATION & COMPUTER SCIENCES

UNIVERSITY of HAWAI'I at MĀNOA

