A Bug-Report based Benchmark in REST APIs

Aakash Kulkarni, Shreyes Joshi, Soon Song Cheok

Introduction

Initial Question:

- Do FL Techniques work on for REST APIs?

When looking for repositories we found that the best way to find real bugs was through bug reports

We aimed to use FL techniques that used bug reports

What went wrong

First step was to assign a bug type

- On the Faults Found in REST APIs by Automated Test Generation

The assumption we started with

- *The existing taxonomy, developed from automated test generation tools, would align well with real-world bug reports found on GitHub.*

What went wrong

This taxonomy could not describe these bug reports

- Existing taxonomy classified bugs from test failures
- Does not represent many bugs found through GitHub issues

We could not create a benchmark without a proper taxonomy suited for bug reports

Without a benchmark, there was no way to test FL techniques

New Scope

We created an accurate taxonomy for REST API faults based on bug reports

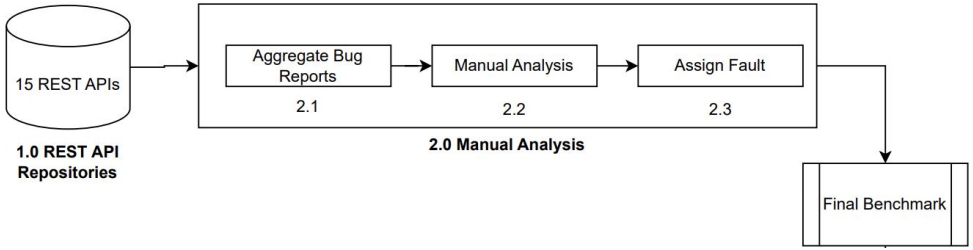
Created a benchmark based on issues and patches from the GitHub Issues

Motivation

- A functional taxonomy and benchmark can help us test FL techniques

Methodology Overview

1. Collect Issues
2. Analyze issues independent
3. Analyze issues together



Methodology - Collect Issues

1. Gather a list of RESI API repositories
2. Go through the repo one by one
    a. Filter issues associated with the repo with label ‘bug’ and state ‘closed’
    b. Collect information about each issue

Methodology - Analyse independently

1. Each of us have the same copy of the list
2. Go through the issues one by one
    a. Identify the bug type from user descriptions and conversation on Github
    b. Add comments for the issue
    c. Consider if the issue is within the scope

Methodology - Analyse together

1. Each of us analysed the list of issues
2. If the bug type is different for the issue
    a. Argue which bug type fits better to the issue
    b. If unable to fit into any of the category
        i. Define a new category and assign the issue to it, or
        ii. Prune the issue that are not within the scope

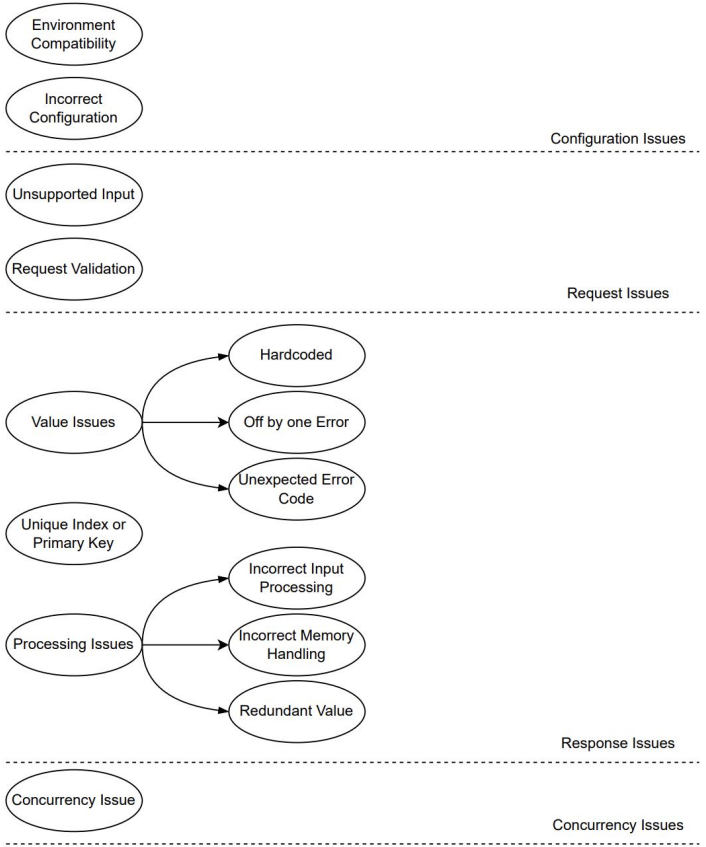
The services we have used:

TABLE I
LIST OF REPOSITORIES

No.	Repository Name	# of Faults
1	management-api-for-apache-cassandra	2
2	cwa-verification-server	7
3	digdag	2
4	ohsome-api	10
5	kafka-rest	3

Results

- Most of issues needed new category for the fault type
- We created our own taxonomy for bug-report based faults



The faults distribution table

TABLE II
FAULT DATA

No.	Fault Type	# of Faults
1	Concurrency Issue	2
2	Environment Incompatibility	1
3	Hardcoded	1
4	Incorrect Configuration	3
5	Incorrect Input Processing	4
6	Incorrect Memory Handling	1
7	Off by one error	1
8	Redundant Value	1
9	Regression Failure	1
10	Request Verification	2
11	Unexpected Error Code	1
12	Unique Index or Primary Key	2
13	Unsupported Input	4

Insights

Almost no overlap between previous taxonomy and our taxonomy

REST API fault is a complicate fault which involve multiple parts of the system, making it hard to classify even if we have clear defined categories

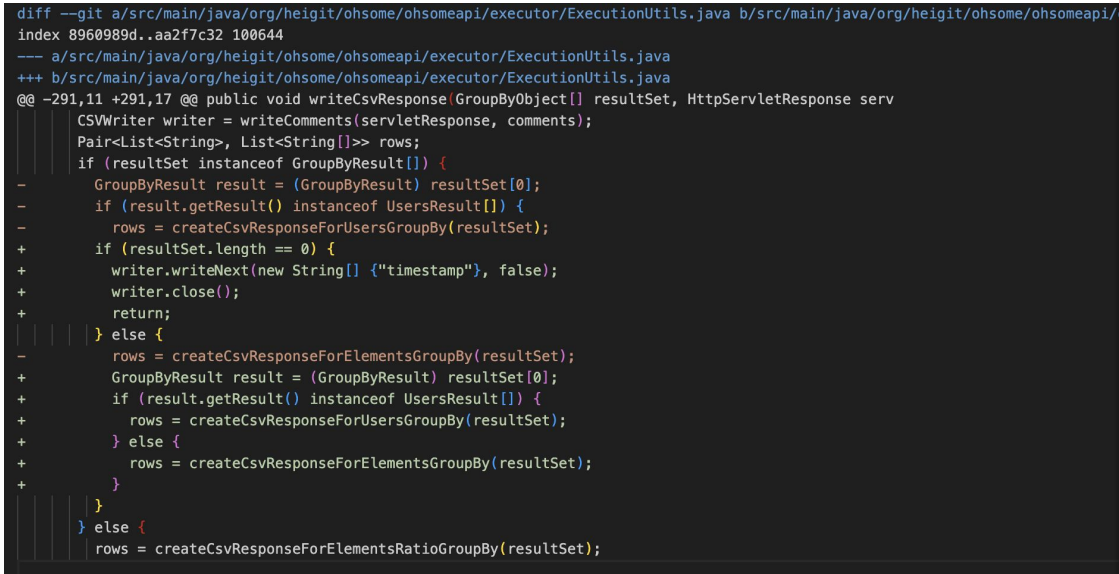
Faults that appear in bug reports will be different than test suites

- Test suites depend failed tests
- Bug reports are more subtle bugs that are best detected by users and developers

Benchmark

24 Bugs

- Service
- Bug Report Link
- Bug Type
- Consider
- Modified File
- Modified Line Numbers
- Comment
- Buggy Commit
- Fixed Commit
- Patch



Discussion

How to determine a benchmark quality for REST API faults?

Can we automate categorization of the Taxonomy we have with the bug reports in future to make this categorization on any service easy?

Would we have a different taxonomy to handle CI based bugs?

Q & A