Seminar Software Quality

Preliminary meeting

We will start at 2:02 pm

Fabian Leinen (Orga)

Jakob Rott
Prof. Andrea Stocco
Martin Gruber
Dr. Benjamin Poppinga
Dr. Markus Schnappinger
Roland Würsching
Dr. Andreas Stahlbauer
David Marson
Software Quality

Quality of Tests

Quality of Code and Models
Participating

1. Apply via matching tool
2. Application with us: Online form
   - Letter of motivation
   - Study program and semester
   - Optional: CV + grade report
   - Your 3+ favorite topics

February 14th, 23:59
Schedule

Until 14th of February

- Application
- Matching
- Matching results and topic assignment

April

- Kickoff
- Individual phase

July

- Literature research
- How to thesis?
- Effective presentations

Block seminar
Grading

**Thesis**
- Seminar paper: max. 15 pages
- Content: Theory + **application** of the topic *(results, experiences, problems and limitations)*
- Initial submission
- Final submission: 1 week after presentation

**Presentation**
- 20 min + 10 min discussion
- Mandatory dry run (1 week before seminar)

50/50
Questions about the organization?
Do **Flaky Tests** Lead Developers to Ignore Test Failures?

```javascript
// Example test function
function test(placeOrder) {
  async (page) => {
    await page.getByLabel('Shares').fill('10');
    await page.getByText('Place Order').click();
    expect(await page.getByLocator('#PositionShares').textContent()).toEqual('30');
  }
}
```

**Table:**

<table>
<thead>
<tr>
<th>Position</th>
<th>Shares</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>20</td>
<td>2,619,00 €</td>
</tr>
</tbody>
</table>
Do **Flaky Tests** Lead Developers to Ignore Test Failures?

Negative Consequences

- Waste of Resources
- Delayed Software Releases
- Loss of Trust in Test Suite

Your Task
How effective is **Rerunning** Flaky Tests in CI?

<table>
<thead>
<tr>
<th>COMMIT</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>test1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rerun</td>
<td></td>
<td></td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td>rerun</td>
<td>pass</td>
<td></td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td>run</td>
<td>pass</td>
<td>fail</td>
<td>pass</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
<td>pass</td>
</tr>
<tr>
<td>verdict</td>
<td>PASS</td>
<td>FLAKY</td>
<td>PASS</td>
<td>FAIL</td>
<td>FAIL</td>
<td>FAIL</td>
<td>FAIL</td>
<td>PASS</td>
</tr>
</tbody>
</table>

**Actual regression failure? Or still flaky failure?**

| **test2** | | | | | | | | |
| rerun | | | fail | pass | | | | |
| rerun | pass | | fail | fail | | | | |
| run | pass | fail | pass | fail | pass | fail | fail | pass |
| verdict | PASS | FLAKY | PASS | FAIL | PASS | FLAKY | FLAKY | PASS |

**Assumption:**
3 failures in a row = regression failure

**Goal:** Verify this assumption
Study subject: Chromium
Contribution: Dataset of **actual** non-flaky failures (first in literature)

**Recommended skills**
- Python
- Pandas
- Basic Knowledge about CI
Insights into Java code coverage for natural language tests

```java
import org.junit.Assert;
import org.junit.Test;

public class CalculatorTest {
    @Test
    public void testAddition() {
        Calculator calculator = new Calculator();
        int num1 = 5;
        int num2 = 10;
        int result = calculator.add(num1, num2);
        int expectedResult = 15;
        Assert.assertEquals(expectedResult, result);
    }
}
```

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Expected Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Launch the browser.</td>
<td>Browser is started.</td>
</tr>
<tr>
<td>2</td>
<td>Click menu → select &quot;Customize&quot;.</td>
<td>The &quot;Customize&quot; window is opened.</td>
</tr>
<tr>
<td>3</td>
<td>Drag 3 new items from the palette or menu panel and drop them onto the Navigation toolbar.</td>
<td>All items are added onto the Navigation toolbar.</td>
</tr>
<tr>
<td>4</td>
<td>Exit &quot;Customize&quot;.</td>
<td>The changes are applied.</td>
</tr>
<tr>
<td>5</td>
<td>Wait at least 15 seconds, after exiting &quot;Customize&quot;, then restart the browser.</td>
<td>Browser is restarted and the previously made customizations are in place.</td>
</tr>
</tbody>
</table>
Insights into Java code coverage for natural language tests
Test Gap Analysis

"Have all changes since the last release been tested?"
unchanged
new & not tested
changed & not tested
changed & executed in test
TGA+
Test Gap Analysis Beyond Coverage

**Great:** Test Gap Analysis is used widely in industry for quality assurance; it helps identifying nasty bugs.

**But:** TGA relies on test coverage: What if a changed code had already coverage in the past but no tests were added or changed for a given change of the system under test?

**Therefore:** In this work, you will *quantify and inspect misleading coverage in open-source projects*; coverage that suggests false safety.

You help establishing an extended notion of Test Gap Analysis: TGA+.

"Every change needs a test."
Clone Detection:

"Where can identical (copied) parts be found in source code?"
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
    boolean found = false;
    StringBuffer res = new StringBuffer();
    if (elements != null) {
        Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
        for (int i = 0; i < elements.length; i++) {
            ModelElement el = elements[i];
            res.append(showElementLink(el)).append(HTML.LINE_BREAK);
            found = true;
        }
        Index.getInstance().resetCurrentRenderer();
    }
    if (!found && nomsg != null && nomsg.length() > 0) {
        res.append(HTML.italics(nomsg));
    }
    return res.toString();
}

// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
    boolean found = false;
    StringBuffer res = new StringBuffer();
    if (elements != null) {
        Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
        for (int i = 0; i < elements.length; i++) {
            ModelElement el = elements[i];
            res.append(showElementLink(el)).append(HTML.LINE_BREAK);
            found = true;
        }
        Index.getInstance().resetCurrentRenderer();
    }
    if (!found && nomsg != null && nomsg.length() > 0) {
        res.append(HTML.italics(nomsg));
    }
    return res.toString();
}
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
    boolean found = false;
    StringBuffer res = new StringBuffer();
    if (elements != null) {
        Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
        for (int i = 0; i < elements.length; i++) {
            ModelElement el = elements[i];
            res.append(showElementLink(el)).append(HTML.LINE_BREAK);
            found = true;
        }
        Index.getInstance().resetCurrentRenderer();
    }
    if ((found && nomsg != null && nomsg.length() > 0) { 
        res.append(HTML.italics(nomsg));
    }
    return res.toString();
}
An Empirical Assessment of Neural Embeddings Techniques for Web Testing

Problem

Redundant + Incomplete
An Empirical Assessment of Neural Embeddings Techniques for Web Testing

Clone pages
Structurally/visually similar pages that represent the same functionality

- The State Abstraction Function determines the amount of redundancy
  - DOM Tree similarity (e.g., with RTED)
  - Visual screenshot similarity (e.g., with PDiff)
    - Several shortcomings (hard to generalize, need thresholds)
An Empirical Assessment of Neural Embeddings Techniques for Web Testing

• Evaluate novel state abstraction functions based on neural embeddings
  • Transformer-like models
  • LLMs
  • Fragment-based approaches

• Compare them with existing approaches (e.g., RTED, PDiff, DOC2Vec)

• Evaluate them within an existing crawler in the context of web testing
  • Accuracy of model inference (redundancy and completeness)
  • Code Coverage of the web app
  • Number of faults retrieved

In this seminar project you will learn how to

• … apply neural embedding models to a new domain (AI)
• … automatically test a web app (SE)
• … perform rigorous empirical experiments
• … report scientific findings (positive outcomes -> scientific publication)
Using LLMs to Improve Reliability

• Reliability is key in large-scale, complex deployments: Maturity, Fault Tolerance, Recoverability
• Many established ways to influence/implement reliability – policies, platforms, ...
• But: human still responsible for many reliability decisions
  o Example 1: Configuring a service's redundancy, quota allocations, etc.
  o Example 2: Recovering a service's health by figuring out what's wrong and mitigating impact.

• Challenge: How can LLMs support humans with reliable service configurations?
  o Input: Any sort of large-scale deployment configurations (Ansible, Helm, Kubernetes, Docker, etc.)
  o Research common mistakes or reasons for outages
  o Explore ways how an LLM can support in this process
  o Build a working prototype (e.g., with Bard, ChatGPT or Open Source LLMs)
Hey GPT, fix my code!

Using Large Language Models to review and correct code snippets

1) Develop effective prompts to improve the quality of source code
2) Build a prototype to perform experiments
3) Investigate Self-Repair and LLMs repairing code from others
Searching for Non-Conformance of a Swarm Controller

- Drone swarm designed to avoid collisions during flight
- How to be confident that the system behaves safely?

Scenario-Based Testing

Discrete-Event Simulation
- Model of decentralized swarm controller
- Sim environment: UAVs, obstacles

Search/Optimization Methods
- Metaheuristic search: genetic algorithms, particle swarm, etc.
Participating

1. Apply via matching tool

2. Apply with us: Online form
   - Letter of motivation
   - Study Program and Semester
   - Optional: CV + grade report
   - Your 3+ favorite topics

http://go.tum.de/070420

February 14th, 23:59