MINING SOFTWARE TESTING KNOWLEDGE FROM STACK OVERFLOW

AUTHOR:
DIBYENDU GUPTA
D.T.GUPTA@STUDENT.TUDELFT.NL
TUDelft

SUPERVISORS:
ANDY ZAIDMAN
BARIS ARDIC
KOEN LANGENDOEN

1. INTRODUCTION

- To accumulated knowledge shared by practitioners from real-world context.
- To bridge the knowledge gap between industry and academia by providing insights and visualizing trends from topics discussed on SO.

2. RESEARCH QUESTIONS

RQ1: What kind of information regarding types of software testing is available on stack overflow?

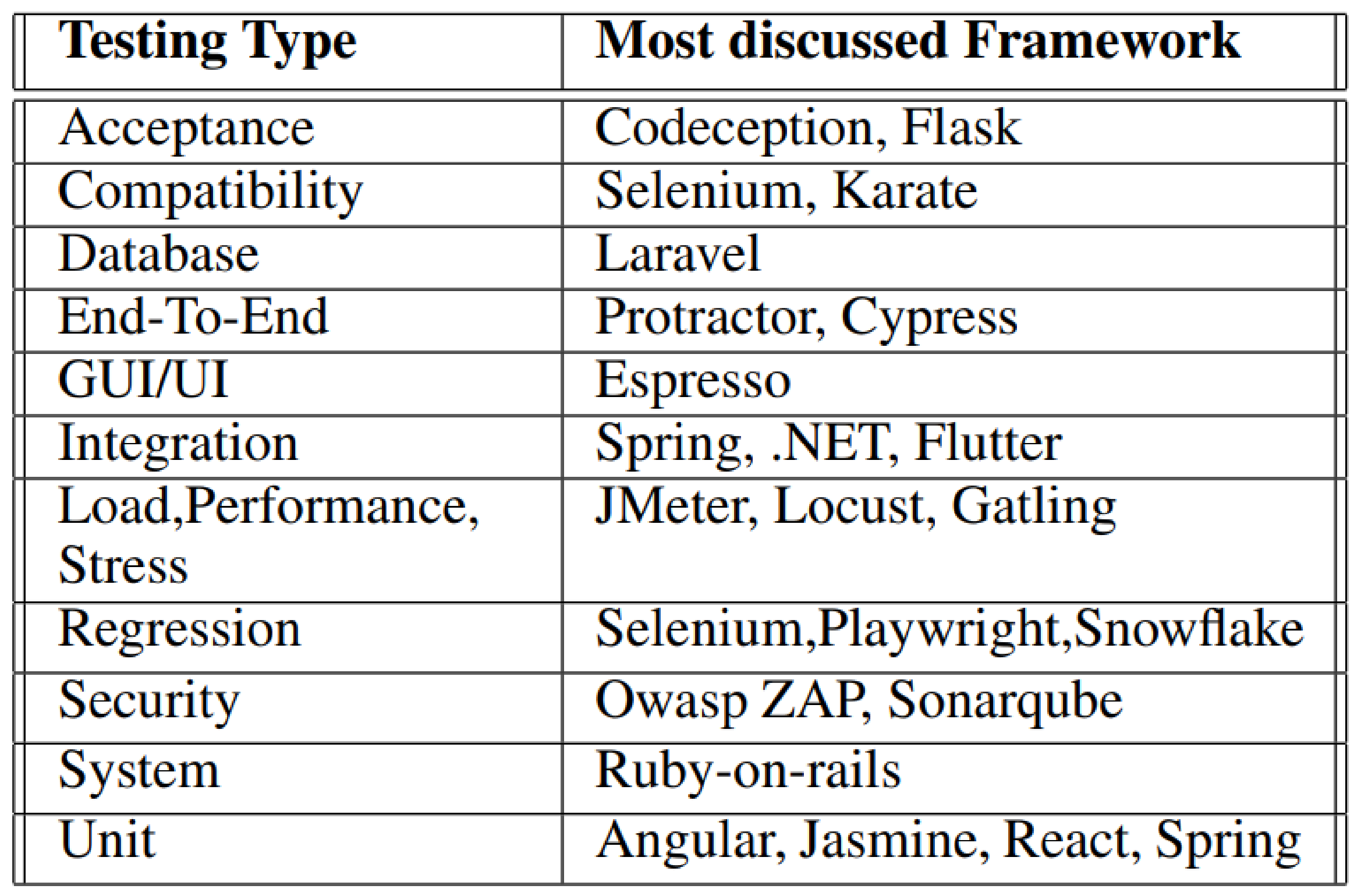
1. What are the **different types** of software testing?
2. What are the **popular testing frameworks** discussed on SO for different types of software testing?
3. What are the **temporal trends** concerning popular technologies/frameworks/tools on SO?
4. What are **controversial opinions** in software testing on SO?

RQ2: What are the **recommended practices, advice and suggestions** on software testing from SO?

3. METHODOLOGY

- Listing search terms: Acquired from literature and cross-referenced with online resources
- Gathering Data: Through automated collection (query posts using Stack Exchange API) and manual inspection
- Pre-process Data: Remove stop-words and HTML tags
- Analyze Data: Frequency analysis, visualizing trends, manual inspection of posts/threads

4. RESULTS



Testing Type	Most discussed Framework
Acceptance	Codeception, Flask
Compatibility	Selenium, Karate
Database	Laravel
End-To-End	Protractor, Cypress
GUI/UI	Espresso
Integration	Spring, .NET, Flutter
Load,Performance, Stress	JMeter, Locust, Gatling
Regression	Selenium,Playwright,Snowflake
Security	Owasp ZAP, Sonarqube
System	Ruby-on-rails
Unit	Angular, Jasmine, React, Spring

Table 1: Summary of most discussed frameworks for different types of software testing

Popular Frameworks

Selenium, Spring, JMeter and React were popular across the different types of software testing.

Controversial Opinions

- Unit Testing is NOT helpful.
- Conflicting opinions on whether developers should test their own code.

Recommended Practice/Advice

Writing Test Cases	Separation	Test Automation	Time for Testing	Log Creation
• Branch coverage is a good score • Avoid code duplication • Stubbing and mocking	• Use test setup for common elements • Create separate folders, packages and directories	• Trade-offs for variables chosen for automation • Dynamic pipeline and CI/CD • Use automation for regression testing	• 100% test coverage ✗ • 10-50% of allocated development time (depends on criticality of system)	• Tracing source of problem • Duplicate error scenario • Use log analysis tools

Temporal Trends

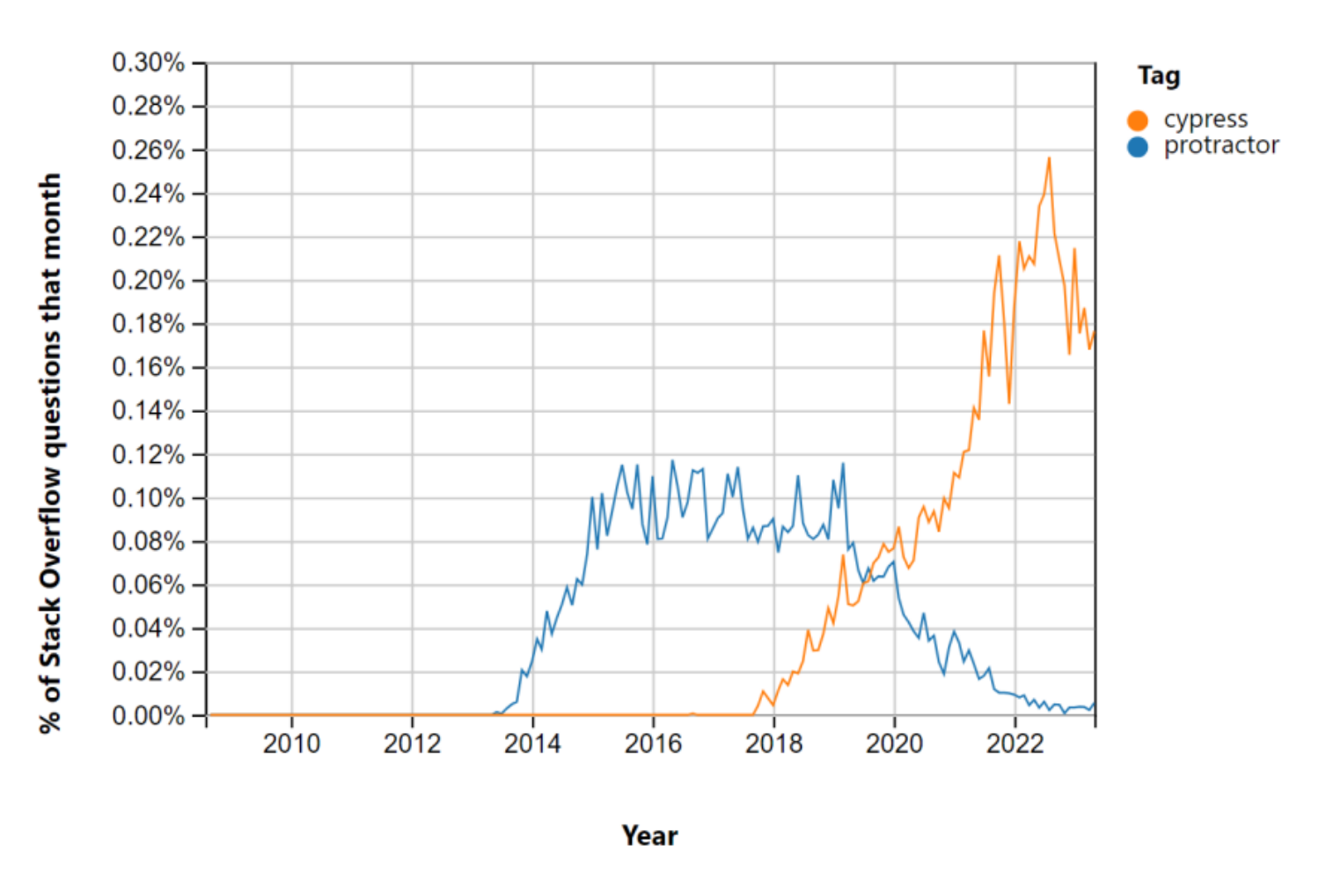


Figure 1: Rise of Cypress vs. Fall of Protractor

- Switch from Protractor to Cypress in e2e testing

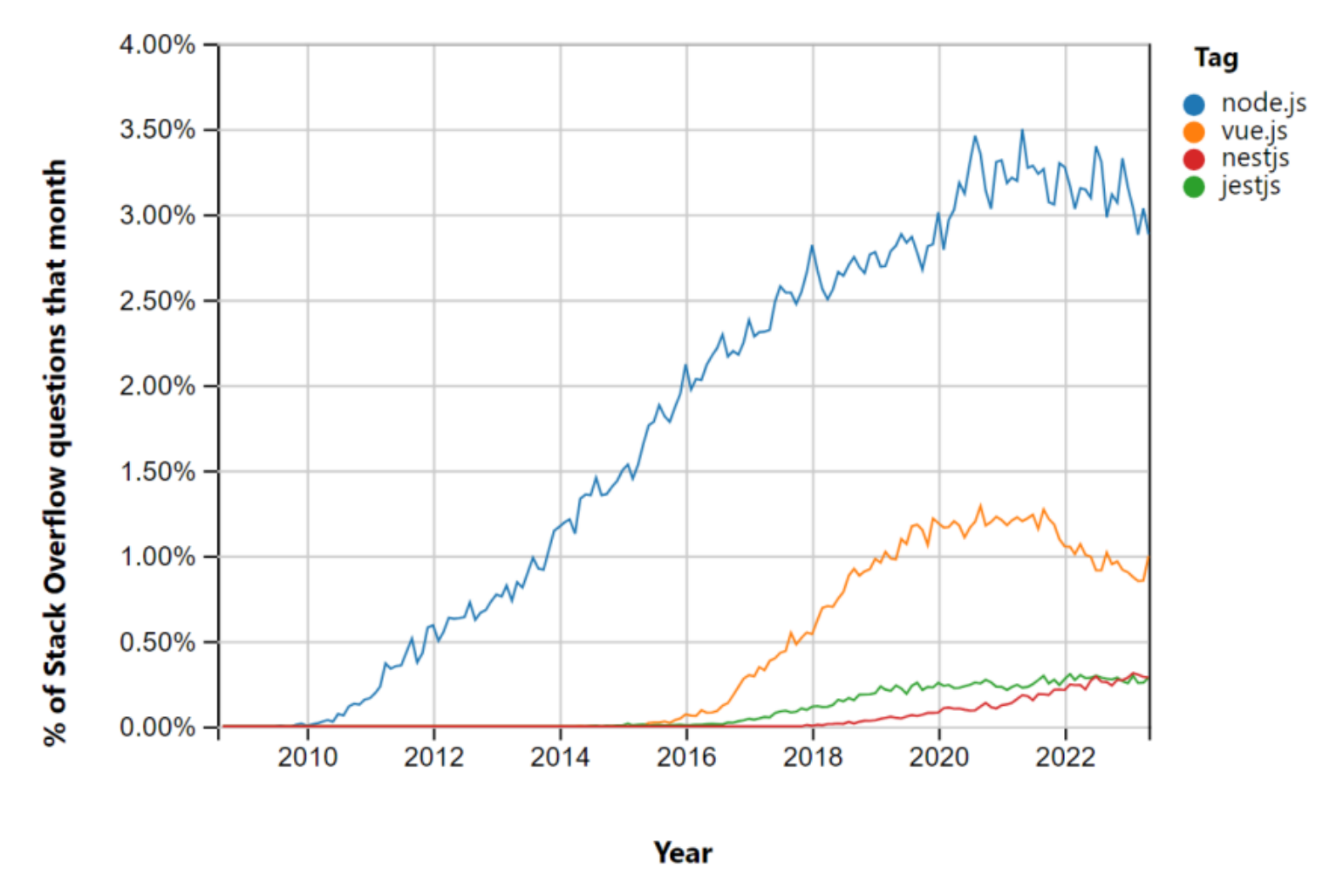


Figure 2: Uptrend of JavaScript Frameworks

- Uptrend of JavaScript frameworks (NodeJS, NestJS, VueJS, JestJS)

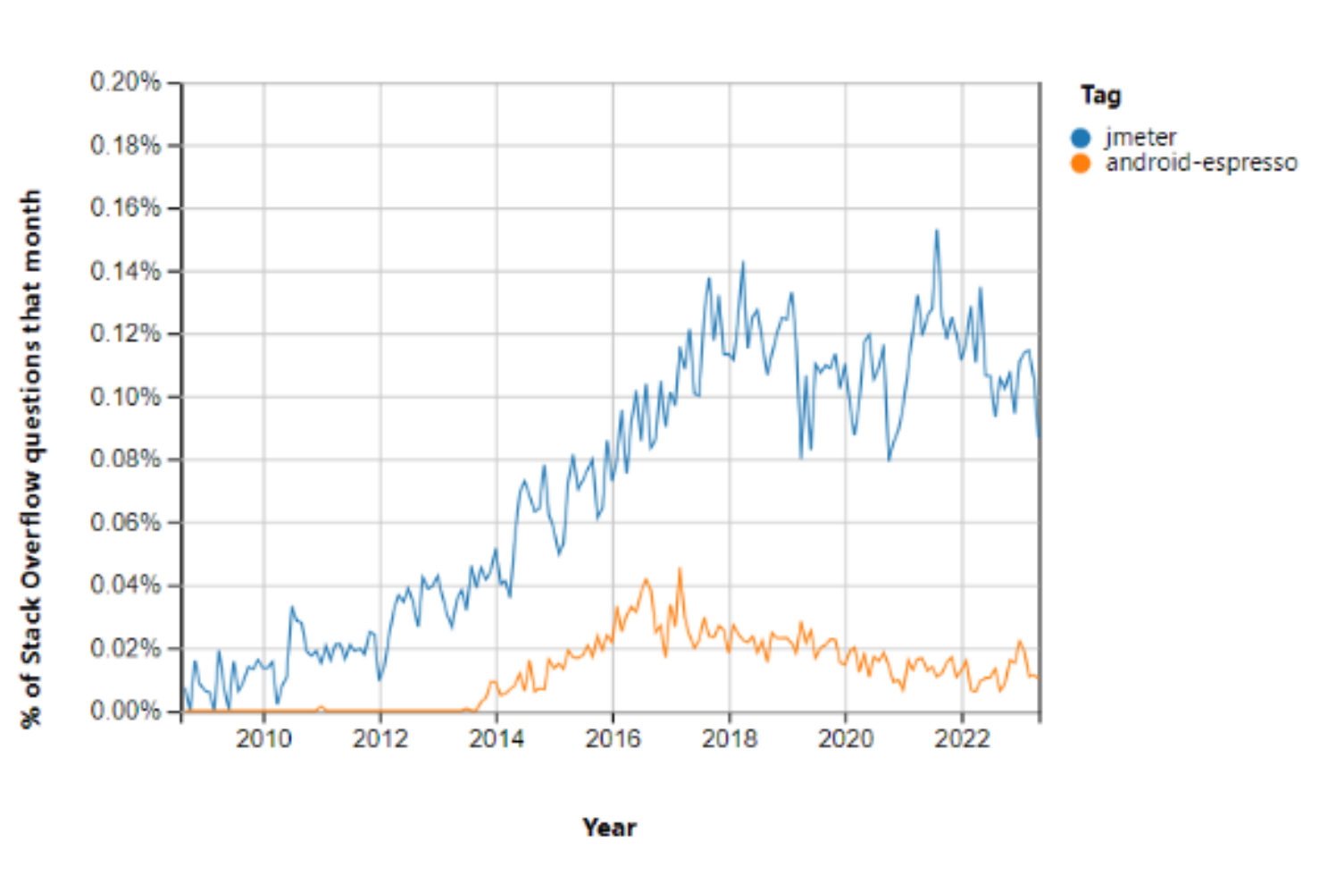


Figure 3: Consistent discussion regarding JMeter and Android-Espresso

- Consistent use of JMeter and Espresso

5. DISCUSSION

- Reliance on test Management and version control software (AWS, Azure, Kubernetes, Gitlab, etc.) have increased.
- Mocking and Mockito are discussed more.
- Testing frameworks and databases are popular topics of discussion. Many posts discuss terminology [1].

6. CONCLUSION & FUTURE WORK

Aggregating information from SO paints an interesting picture of the technologies modern practitioners use in the industry. The trends helps identify the progress in software testing and why such changes occur. Recommended practices enable us to determine what practices are valued by SO users and could be standardized. Future direction of research:

- Different approach to this research (NLP or LDA) to identify natural areas of conversation
- Research collectives on SO for technology-specific information
- Measuring controversy from SO

stackoverflow

7. REFERENCES

[1] Pavneet Singh Kochhar. “Mining Testing Questions on Stack Overflow”. In: Proceedings of the 5th International Workshop on Software Mining. SoftwareMining 2016. Singapore, Singapore: Association for Computing Machinery, 2016, pp. 32–38.