# A Comparative Analysis of CI/CD Tools

Erik Andersson (erian599), Victor Bennich (vicbe572)

## Introduction

Continuous Integration is a growing practice where each member of a software development project integrates their work frequently.

Jenkins, Gitlab CI and Travis CI are some of the most common tools used for Continuous Integration today. Because of this we chose to analyze and compare these tools in order to provide some guideline for organizations in their decision for CI infrastructures.

The aim of this research is to compare and give an overview of the 3 tools previously mentioned, and specifically to answer the following questions:

1) What are the key properties of the leading CI/CD tools?

2) How does a software architect choose the most suitable tool for their system based on these properties?

### Quick Facts

- Jenkins accounts for **nearly 72%** of **all** CI/CD systems [1].
- One case study found that developers using CI/CD tools commit to production **3.5 times more often** than those who don't [2].
- Though CI/CD is a relatively modern trend, the earliest known work on continuous integration dates back to **1989** [3].

## Methodology

### Tools

In order to test the various CI/CD tools we chose to set up two version control environments on which we would run our tests, namely GitHub and GitLab.

We then set up Jenkins, GitLab CI and Travis CI environments and connected them to the Git repositories. Jenkins and GitLab CI ran on a local machine while Travis CI, which is a hosted tool, ran on the service provider's systems.

Figure 1: The three tools which were tested. From left to right: Jenkins, GitLab & Travis CI.

### Metrics

Firstly, we chose to evaluate the different tools by measuring their usability, using a standard called Single Usability Metric (SUM), which is a scalable process for standardizing disparate usability metrics [4]. Higher SUM values indicate more user friendly systems.

In addition to usability, we also looked at attributes such as the tool's software model (open source vs. proprietary), costs, security management and more.

By combining these two metrics we were able to determine when each of the tools were best suited.

## Results

### Usability

We found that GitLab CI was the most user friendly, with a SUM score of 71.2%, followed by Travis CI (69.8%) and finally Jenkins (62.6%).

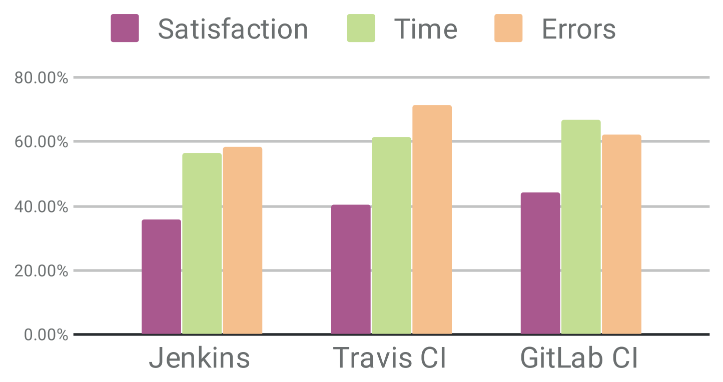

Figure 2: The component metrics of SUM for each tool. Higher values are better.

### Tool Attributes

When analyzing the various tools, we found the following attributes were among the most important to consider when choosing the best CI tool for the job:

| | Jenkins | GitLab CI | Travis CI |
|---|---|---|---|
| **Hosted** | No | Supports Both | Yes |
| **Software Model** | Open Source | Open Source | Open Source |
| **Technology** | Java | Ruby & Golang | Ruby |
| **Integration** | Comprehensive | GitLab | GitHub |
| **Cost / Month** | Free* | 0-$99 / User | $69 - $489 |
| **Security** | Low (**Default**) *Configurable* | High | High |
| **SUM** | 62.60% | 71.20% | 69.80% |

## Conclusion

Based on the results gathered we came to the following conclusions:

**GitLab CI** is the most user friendly of the CI/CD tools. However the pricing structure quickly scales with users and as such an architect might want to **avoid this tool for larger teams** unless one also wants the features of GitLab's version control.

**Travis CI** was the second most user friendly tool investigated, not far behind GitLab CI. The fixed monthly pricing structure and close ties to GitHub makes the tool a **perfect fit for medium to large teams operating on GitHub.**

Finally, **Jenkins** was the least user friendly, though also being the most customizable. Unlike the others, Jenkins is free, though the total cost of ownership is not zero, given the tool's maintenance requirements. The architect should therefore consider this tool for **large teams with a dedicated development operations worker**.

## References

[1] Datanyze, 'Jenkins Market Share and Competitor Report', 2019.[Online]. Available: https://www.datanyze.com/market-share/ci/jenkins-market-share. [Accessed: 10- May- 2019].

[2] T. Savor, et. al, 2016. Continuous Deployment at Facebook and OANDA. 2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C).

[3] G. E. Kaiser, D. E. Perry and W. M. Schell, "Infuse: fusing integration test management with change management," [1989] Proceedings of the Thirteenth Annual International Computer Software & Applications Conference, Orlando, FL, USA, 1989, pp. 552-558. doi: 10.1109/CMPSAC.1989.65147

[4] J. Sauro, E. Kindlund, 2005. A method to standardize usability metricsinto a single score. Proceedings of CHI 2005: Technology, Safety,Community, : 401–409.