

Taxonomic Analysis of DevOps Tools

Mrs. Geetika Singh

Institute of Computer Science
University, Ujjain
geetika.singh3@gmail.com

Dr. Jitendra Chaudhary

Medicaps University, Indore
jitendra.choudhary@medicaps.ac.in

Dr. Lokesh Laddhani

Institute of Computer Science Vikram
Vikram University, Ujjain
lokesh.laddhani@gmail.com

Received 2022 April 02; **Revised** 2022 May 20; **Accepted** 2022 June 18.

Abstract

The shift from automated testing to the level of business-driven continuous testing described above is a significant step forward in the development of software. However, from the perspective of DevOps, it offers significant benefits. Along the use of objective real-time evaluation of whether software meets business requirements at numerous "quality gates," organisations may automatically and reliably progress release candidates through the delivery pipeline. In this research paper, the details of several open-source testing tools, as well as the testing types that they provide, are given in a tabular format. It is true that some testing tools are more effective than others depending on the circumstances. To properly comprehend the DevOps revolution, many crucial components are necessary. The revolution is often sparked by a compelling event inside a firm, such as a transition to Agile development. Delivering software as a service (DevOps) and continuously testing are two additional processes that may be accelerated by using an Agile development methodology. This paper contains a taxonomic analysis of DevOps tools.

Keywords- DevOps, Software Delivery, Automation, DevOps Tools.

Introduction

The businesses must be nimble enough to respond to ever-changing market demands while also balancing many consumers demands. While software is mostly responsible for corporate innovation in the digital age, it is critical for businesses to provide speedier application upgrades (without sacrificing quality) in order to win, serve, and retain consumers. Traditionally, businesses have concentrated on providing customers with error-free products. The goal has now shifted to offering more value to the business through faster release cycles and a faster return on investment [1] [2]

Cloud computing has introduced an application platform, a new consumption mechanism, and a "strong foundation for new business models" to the corporate IT mix, all of which have benefited the organisation. The result is that businesses are increasingly relying on new ideas, sound technology, and easy execution, which has resulted in the establishment of 'DevOps,' or development and operations teams that collaborate. Because of the increasing importance of DevOps, it is becoming more vital to control, monitor, and automate software releases across the whole lifecycle of an application's lifecycle. Because to DevOps innovation, development and operations teams were able to reimagine the way they collaborated in order to match their goals and priorities, as well as to embrace shared toolkits[3].

To properly comprehend the DevOps [4] revolution, many crucial components are necessary. The revolution is often sparked by a compelling event inside a firm, such as a transition to Agile development. Delivering software as a service

(DevOps) and continuously testing are two additional processes that may be accelerated by using an Agile development methodology. Agile-enabled acceleration implies a more rapid release schedule, which is necessary for success. An organisation that wants to see its product released successfully must undergo frequent testing to guarantee that the conveyor belt does not break down while in operation.

The purpose of this paper is to classify open source tools used in software development within DevOps, to evaluate the most commonly used open source automation testing tools in Continuous Testing, and to compare these tools based on certain criteria in order to highlight their performance and where they can be used to prefer solutions in different environments, in order to benefit both researchers and practitioners in equal measure, so that they can benefit both researchers and practitioners in equal measure

Overview of DevOps

The term "DevOps" is a combination of the words "development" and "operations." This is a culture that encourages collaboration between software developers and testing teams in order to generate software faster and in a repeatable manner. It encourages manufacturers to build quickly and improve delivery times and services. This facilitates a stronger relationship between service providers and their customers. DevOps purpose is to successfully release features into production without disrupting other services while swiftly recognizing and correcting incidents as they arise throughout the DevOps lifecycle [5].

Businesses must first ensure that their information technology infrastructure is up and working effectively before deploying new applications. It is critical to maintain a continual level of preparation at all times. Coding check-ins are becoming more frequent as the development process gets more agile. When the environments are deployed, this results in a conflict between them. When the environment is continually changing, it becomes more challenging to reach the level of preparation required for continuous application deployment on an operational level [6].

Everything from creating the application code to delivering it to clients may be automated via the use of DevOps practices. A conveyor belt that transports application code through the various phases of development and deployment is referred to as DevOps [7].

When it comes to DevOps, a tool chain is a collection of processes or "tools" that are linked together to regulate the delivery pipeline. Pipelines are a popular tool in the DevOps field, and they are used to streamline and standardize various activities. Development of software operations pipelines is primarily concerned with automating procedures and offering insight into the software development lifecycle. It will be discussed in this research paper which open source continuous testing tools are the most popular and widely used. An knowledge of the roles of the pipelining components is required prior to undertaking any tool analysis on the pipelining components [8].

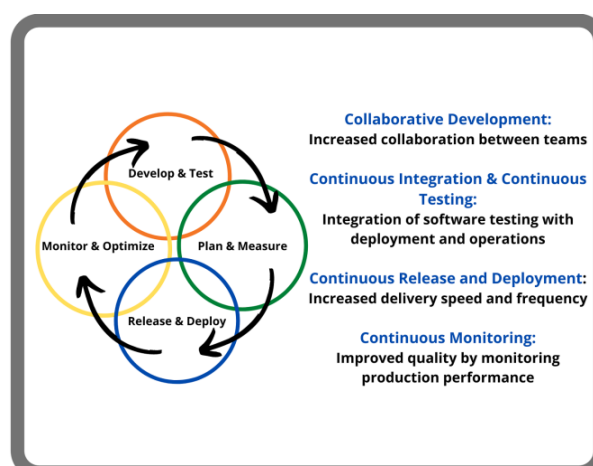


Figure: DevOps Components

DevOps Automation and Continuous Testing

Automation is at the heart of DevOps, which is primarily concerned with the setup, configuration, deployment, and maintenance of infrastructure and applications. Setting up settings and keeping them up to date over time is significantly easier if we automate the process as much as possible. In the past, server configuration and application deployment were mostly manual processes that were prone to error and instability. For the purpose of addressing these issues, organisations hired highly qualified employees to assist with manual configuration. Although this was a significant step forward, it did not totally eradicate the issue, leaving the possibility of a negative impact on key, high-value operations such as software release, machine setup, operating system patching, troubleshooting, and bug correcting open for consideration [9].

In this case, automation comes to the rescue, automating the vast majority of the company's most significant operations and processes. An end-to-end automated method with a greater degree of automation is provided by the DevOps environment, rather than depending on human interaction or access to production settings. In order to automate substantial elements of the end-to-end software development and deployment process, DevOps relies on technological advances. The focus of our inquiry is open source tools for different DevOps components, which are available for free [10].

When it comes to the DevOps team, open source solutions are essential since they allow teams to innovate more quickly while also lowering the cost of installation. As a consequence, organisations may use open-source software that is designed to cooperate with other software components to utilise the finest tools for the job while simultaneously reducing complexity [11].

We may use a range of commercial and open-source tools to determine how well an application works, and we can compare the results with other applications. In the DevOps era, open-source technologies have become more popular. Open source technologies make issues more evident, allowing them to be remedied more rapidly as a result of their visibility. Choosing the most appropriate tool for the company's strategy requires some deliberation and work [12].

Categorization of DevOps Open Source tools

Different components of DevOps use different types of open source tools to simplify the process of software development. Details of DevOps tools for Continuous Integration and Continuous Management are as follows:

Chef - It is simpler to manage cloud and data centre environments using Chef, which automates procedures while simultaneously enforcing rules and protecting sensitive data. This technology makes it easier for cross-team communication and the adoption of continuous delivery to take place [13].

Ubuntu's - Ubuntu's mascot is named Juju. We can save both time and effort by using this service orchestration solution. With just a few lines of code, we can model, configure, and manage services in Juju, and then deploy the code to any of the major public and private cloud platforms available today. The Juju store offers hundreds of pre-configured services that may be customised to meet your needs.

Puppet - Configuration management software, developed in Ruby and offered for free, is available for downloading. It is the dominant platform for offering and operating contemporary applications on a consistent basis, regardless of where it is deployed. In order to provide continuous delivery, Puppet is used to enforce and verify security standards, transfer to the cloud, and deploy DevOps methodologies for continuous delivery, among other things [14].

Ansible - It is a free and open source DevOps platform that includes configuration and automation management. Additionally, multi-node deployment and ad-hoc task execution are supported. Modules may be built in any language and may utilise either JSON or a standard output as their output format.

DevOps Tools for Continuous Delivery

Some of the popular tools for Continuous Delivery, described below:

Fabric - With this library and command-line tool, it is much easier to deploy and administer Python programmes through SSH than it was previously possible. Additionally, it may conduct a number of auxiliary tasks in addition to running shell commands locally or remotely, sending files, and requesting input from the currently-logged-in user.

Jenkins - Continuous integration and delivery (CI/CD) platforms such as Jenkins, which allows software projects to be produced and tested on a continuous basis, are available. Both developers and users gain from the simplicity with which modifications can be merged and with which new builds can be accessible in the new build environment. Jenkins supports a wide range of testing and deployment tools, as well as the ability to create custom build pipelines to ensure that software projects are delivered on a consistent basis [15].

DevOps Tools for Continuous Testing

Some of the tools for Continuous Testing [16] [17] are as follows:

ANT - is one of the numerous tools that Apache has built over the years, and it is available for free (Another Neat Tool). A simple XML file with general task descriptions is all that is required for this programme to function, and it assists in the automation of everyday chores as well.

BuildHive - a free and open-source alternative to Jenkins for GitHub projects, may be downloaded here. BuildHive support for GitHub projects may be enabled with a single click of the mouse by project administrators. Various project types, such as Ant, Maven, Gradle, SBT (Scala), and Rake may be recognised and created automatically (Ruby). User settings are seldom changed by end users in the vast majority of cases.

Gradle is a lifecycle integration solution for DevOps development that integrates with Jenkins. In addition to the Integrated Development Environment (IDE), Gradel now includes a Continuous Integration (CI) server, as well as Docker containerization and deployment.

Maven - In addition to Java, C#, Ruby, and Scala, Maven is a build automation tool that supports a wide range of programming languages. The process of automating a project is as easy as following a few straightforward guidelines. To specify the software project that is being built, as well as its dependencies on other plug-ins, external modules, and components, among other things, an XML file is utilised.

Selenium - is a web application testing framework that is free to use and is available as open source software. Browsers and operating systems from a broad variety of vendors are supported. Selenium can aid testers in creating tests for a wide range of computer languages using the Selenium framework. It is not necessary to understand a test scripting language in order to use this tool to do tests on the applications. Selenium is a prominent open-source test automation tool because of its large and active user and development community, which makes it a popular choice in the business.

Appium - According to Appium, which bills itself as a "whole new revolution in automated testing," it can provide high-quality, bug-free apps while also saving a project considerable amounts of both time and money. Appium is a mobile testing tool that is open source and cross-platform, enabling developers to create tests for both iOS and Android smartphones using the same code. Appium may be used with a wide range of programming languages, including Java and C++. A broad variety of programming skill sets may be used effectively by programmers using this tool, as a result of this feature. Appium is a tool for measuring the interaction between mobile web app content and its users. The results of the tests are used to assess the correctness of the mobile app, the interaction with the user, and the accessibility of different capabilities inside the app. [40]

Watir - In addition to being a web-based testing automation tool, Watir is also a Windows-only application that is meant to perform basic and easily maintained tests that are simple to repeat. Watir is a web automation testing tool that is built on the Ruby programming language. Watir's cross-browser testing supports a wide range of browsers, including Firefox, Opera, a headless browser, and Internet Explorer. In addition to RSpec and Cucumber, there are varieties of other BDD tools to choose from, such as Test/Unit. Integration of Watir with frameworks like as cucumber and continuous integration platforms such as Jenkins may help us achieve DevOps or continuous delivery.

DevOps Tools for Continuous Monitoring

Some of the tools for Continuous Monitoring [18]their detailed given below:

New Relic APM - With the use of this technology, team members may be able to spend more time developing and less time troubleshooting. This solution enables extensive service-oriented application traceability from the beginning to the end of the application lifecycle. The ability to dive down into the code and SQL statements allows us to see how particular code segments and SQL statements effect overall performance.

New Relic Server - It assists us in keeping track of the full monitoring checklist and guarantees that the server is running smoothly. It keeps the DevOps and engineering teams in sync and makes dealing with disc capacity, CPU, RAM, and disc I/O utilisation issues easier.

As a result, businesses pursuing DevOps must significantly rely on automation, which necessitates the use of technologies. Though the potential benefits are clear, any firm or organisation attempting to successfully implement a DevOps strategy will face numerous hurdles. Realizing those potential benefits may be far more difficult than anticipated, and a poor implementation can be extremely disruptive.

Need of Continuous Testing

The testing phase of the DevOps tool chain is one of the most important. Before going on to the next step, we must ensure that the application has an acceptable level of risk and include automatic tools to verify the application's functional and nonfunctional behavior. DevOps aids the IT side of the house in starting to build "infrastructure as code," allowing a firm to quickly launch apps. Assume, however, that we do not do rigorous, multi-channel, end-to-end ecosystem validation of our application. What we put into this fast-paced ecosystem could fail in unexpected ways, dragging down the entire DevOps process. It has the potential to cause release delays, poorer profitability, and ultimately return us to where we were before we opted to implement DevOPs. Continuous testing, as part of the DevOps toolchain, helps us mitigate this risk by automating the testing process and ensuring thorough end-to-end application validation throughout the development lifecycle.

A selection of tools that have undergone a thorough review procedure based on inclusion and exclusion criteria commonly used by industry experts are presented in the following section. Several well-known open-source testing tools that we've previously studied are included in the following table for our consideration. Comparing their pros and drawbacks in a table exposes information about topics like reusability, reliability, and pricing. These characteristics provided as a jumping-off point for more investigation [19].

Tools	License	Supporting Platform(Mobile)	Supporting Platform (Web)	Supporting Platform (Desktop)	Learning Ease	Programming Skills	Code Reusability	Test Results REport	& Record PlayBack
ANT	√	√		√					√
BuildHive				√		√	√		
Gradle		√				√	√	√	
Maven				√			√		
Selenium			√		√	√	√	PLUG IN	√
Appium		√					√		√
Watir			√		√		√		

Table 1: Detailed Description of Tools

Results and Discussion

Table 1 gives a thorough description of open source tools, based on certain criteria, which will be beneficial to industry practitioners in determining the tool that should be employed for continuous testing objectives. The table above highlights the tools mentioned and the requirements each tool met. The results of this research will be valuable to software testers in their quest to identify the optimal testing strategy for their projects.

Conclusion

Continuous Delivery procedures that are completely automated lower the business risk associated with them since they allow for rapid and objective assessment of whether or not the organization's specific expectations are satisfied at the time of the critical "go/no go" decision. Whenever possible, it is ideal to rectify flaws when they are the most straightforward to access, the fastest to cure, and the least costly to correct. It is possible to choose the most appropriate tool for a project depending on the tool that was used to create it throughout its development. This study would be beneficial to professionals in the industry, as well as researchers, in order to adapt DevOps and pick the optimal technology for a certain project. It will be much easier for software testers to find the finest tools for testing applications when they have the correct testing tool at their disposal. They will also save time. Because of DevOps Continuous Testing, we may conclude that business stakeholders will be better informed when they have access to real-time input on whether or not their expectations are being met.

References

- [1] L. Bass, I. Weber, and L. Zhu, DevOps: A software architect's perspective. Addison-Wesley Professional, 2015.
- [2] A. Dyck, R. Penners, and H. Lichter, "Towards definitions for release engineering and devops," in 2015 IEEE/ACM 3rd International Workshop on Release Engineering. IEEE, 2015, pp. 3–3.
- [3] G. Kim, J. Humble, P. Debois, and J. Willis, The DevOps Handbook:: How to Create World-Class Agility, Reliability, and Security in Technology Organizations. IT Revolution, 2016.
- [4] M. Senapathi, J. Buchan, and H. Osman, "Devops capabilities, practices, and challenges: Insights from a case study," in Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018. ACM, 2018, pp. 57–67.
- [5] K. Kuusinen, V. Balakumar, S. C. Jepsen, S. H. Larsen, T. A. Lemqvist, A. Muric, A. . Nielsen, and O. Vestergaard, "A large agile organization on its journey towards devops," in 2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). IEEE, 2018, pp. 60–63.
- [6] L. Bass, R. Jeffery, H. Wada, I. Weber, and L. Zhu, "Eliciting operations requirements for applications," in Proceedings of the 1st International Workshop on Release Engineering. IEEE Press, 2013, pp. 5–8.
- [7] M. Rowe and P. Marshall. The business case for collaborative devops. [Online]. Available: <https://www.ibm.com/developerworks/community/blogs/>
- [8] K. Nybom, J. Smeds, and I. Porres, "On the impact of mixing responsibilities between devs and ops," in International Conference on Agile Software Development. Springer, 2016, pp. 131–143.
- [9] P. Labs. (2014) 2014 state of devops report. [Online]. Available: <http://puppetlabs.com/sites/default/files/2014-state-of-devops-report.pdf>
- [10] B. Glaser and A. Strauss, "Grounded theory: The discovery of grounded theory," Sociology the journal of the British sociological association, vol. 12, no. 1, 1967.
- [11] B. Glaser, "Theoretical sensitivity," Advances in the methodology of grounded theory, 1978.
- [12] R. Hoda, J. Noble, and S. Marshall, "Using grounded theory to study the human aspects of software engineering," in Human Aspects of Software Engineering, 2010, pp. 1–2.
- [13] S. Adolph, W. Hall, and P. Kruchten, "Using grounded theory to study the experience of software development," Empirical Software Engineering, vol. 16, no. 4, pp. 487–513, 2011.
- [14] S. Bick, K. Spohrer, R. Hoda, A. Scheerer, and A. Heinzl, "Coordination challenges in large-scale software development: a case study of planning misalignment in hybrid settings," IEEE Transactions on Software Engineering, vol. 44, no. 10, pp. 932–950, 2017.

- [15] F. M. Erich, C. Amrit, and M. Daneva, "A qualitative study of devops usage in practice," *Journal of Software: Evolution and Process*, vol. 29, no. 6, p. e1885, 2017.
- [16] L. E. Lwakatare, T. Kilamo, T. Karvonen, T. Sauvola, V. Heikkilä, J. Itkonen, P. Kuvaja, T. Mikkonen, M. Oivo, and C. Lassenius, "Devops in practice: A multiple case study of five companies," *Information and Software Technology*, 2019.
- [17] J. Smeds, K. Nybom, and I. Porres, "Devops: a definition and perceived adoption impediments," in *International Conference on Agile Software Development*. Springer, 2015, pp. 166–177.
- [18] T. Clear, "Thinking issues meeting employers expectations of devops roles: can dispositions be taught?" *ACM Inroads*, vol. 8, no. 2, 2017.
- [19] J. Roche, "Adopting devops practices in quality assurance," *Commun. ACM*, vol. 56, no. 11, 2013.