The future of DevOps

Mustafa Ali
Noah Rahimzadagan

KTH Royal Institute of Technology — May 16, 2022

Abstract

For over 13 years now, DevOps has emerged as the modus operandi when creating enterprise software in an organization. The definition of DevOps has changed vastly and keeps evolving each year. It is an effective way of deploying and maintaining services but there is still room for improvement. The notion of automatic repair can decrease the workload for developers immensely. DevSecOps addresses the security aspect of DevOps in where it introduces security tests throughout the DevOps process. AIOps solves issues regarding operational issues automatically. Clearly, the DevOps process is becoming more and more automatized.

1 Background

DevOps poses a highly efficient way of working for engineers and making the process of developing software a beautiful machinery. DevOps has been around since 2007, and when reading different articles on the topic, one quickly realizes that the definition of DevOps has been changing frequently.[1]

At first, the hard separation between developers and IT specialists caused a lot of difficulties. Developers were blaming IT Operations for deployment errors and vice versa. DevOps mitigated this issue by making the two entities communicate. And as DevOps emerged, this was what DevOps was about, the rise of communication between the two. There were no standardized ways of improving communication, it was simply left to the developers to decide. Slowly but surely, DevOps was not just an idea anymore but slowly it became associated with a collection of specific tools. This essay will explore emerging technologies, tools, and disciplines that might very well be standardized in the DevOps domain.

2 Continuous Integration

Simply put, the Continous Integration, commonly abbreviated as CI is the part of DevOps where the contribution of individual developers is merged to the code base of the software project. The CI entails among other things, running tests and building the program with the changes from the individual developers, in order to detect errors or bugs that could have arisen. [2]

2.1 Automatic repair

Instead of merely saying which test failed in the continuous integration, automatic repair could be implemented. Automatic repair itself is astonishing as its purpose is to identify and resolve bugs with no human intervention. [3]

Time spent debugging software could be time spent exploring new ideas, and putting new ideas in motion, making automatic repair a tool that dramatically increases a developer's productivity.

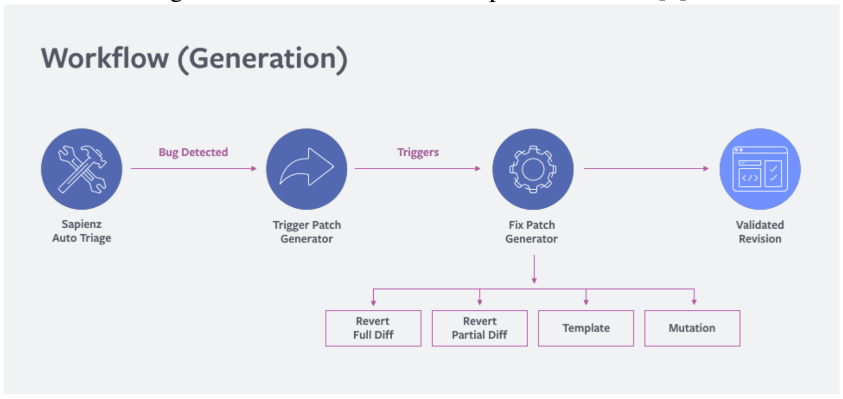
2.1.1 SapFix

Several years ago, Meta introduced SapFix. A hybrid AI tool that detects bugs and suggests patches. It works in conjunction with Sapienz, another tool made by Meta, which performs a computational search of the space of all possible tests of a program. Sapienz has proved to be very effective. SapFix identifies bugs with the help of Sapienz and then independently suggests patches that are harvested from a collection of past fixes made by developers. SapFix can also do a so-called mutated fix, which means that it infers a fix from a distinguished pattern made by past fixes. The last resort strategy for SapFix is to roll back to earlier iterations of the code, to when before the bug arose. SapFix was tested on six core Android Meta applications, in the continuous integration part of the projects and is scheduled to be made available for the public. SapFix is not entirely automatic, as it requires the developer's approval for the patches. [4]

2.2 Repairnator

The autonomous agent Repairnator aims to solve bugs in the continuous integration stage of open-source software. Repairnator was created by researchers at the Royal Institute of Technology (KTH). The workflow consists of first taking in CI builds as input, it gets these builds by constantly

Figure 1: An overview of the SapFix workflow. [5]



monitoring the Travis CI. Then Repairnator will analyze these builds. The second step is to reproduce the bugs. The third and last step is to provide a patch and collect data for future program repair research. Astoundingly, the Repairnator has thus far fooled human developers on GitHub on five different occasions under the fake alias "Luc Escape". The authors of Repairnator state that there clearly is room for improvement and that perhaps, in the future, the different automatic repair programs will work tightly knit together and review each other's patch fixes. [6]

3 Code reviews

After the continuous integration has built and tested the code in line with a commit, many companies make use of a practice called code review. A code review is an assessment of the quality of the code you have written as a developer, by a peer. There are many benefits to code reviews such as sharing knowledge between the developers since many developers in an organization can in some sense possess different skill sets. Code reviews enable the flow of knowledge to pour between the metaphorical cups of the different developers. Furthermore, the consistency with the rest of the written code of the organization is improved, since it is easy to miss consistency requirements as you are focused on the programming task itself. Compliance is also a big motivator for performing code reviews, in order to not deploy code that violates significant security requirements. [7]

3.1 AI Code Reviewers

It is no significant scientific discovery that humans are error-prone. Humans are not perfect and committing mistakes will be inevitable for the wisest developer. Furthermore, a study has shown that on average, only 13% of all pull requests are rejected for technical reasons. [8]

Another issue one easily can deduce, that is due to code reviews is the fear of novice developers of trying something daring, and effectively becoming ridiculed by their peers or more experienced developers. That is why AI code reviewers have a place in the DevOps realm and the software engineering landscape. By implementing an AI agent to review code, we eliminate all erroneous drawbacks that arise when human reviewers give their assessment. Subjectivity is vastly lowered, and it will make novice developers in a workplace less vulnerable to groupthink and more daring in their development approach.

3.2 Codeguru

Codeguru is an AI Code Review solution made by Amazon. The objective of Codeguru is to give sophisticated feedback on how to improve code. Not only for improving the quality of code but also for consistency and identifying security vulnerabilities. In a sense, one can almost view the Codeguru as a sibling to the automatic repair programs that are out there. Since Codeguru is deployed in the continuous integration of a workflow, it immediately scans the input from the commit and gives suggestions in the pull request dashboard. [9]

3.3 Future of Code Reviews

An assessment of research made on the subject clearly suggests that automated code reviews or code reviews by AI agents have a place in the software engineering landscape. In the future, there may very well be more tools than the Codeguru that will be used across most companies.

4 Security

With the many cyber attacks occurring these last years, it is not completely easy to ignore the importance of security in the DevOps process. Clearly, many software engineering companies are not only maintaining and creating one big monolithic app but rather, they are maintaining and working on multiple so-called microservices. [10] This effectively means that an app is even more exposed to vulnerabilities, since individual microservices will in most cases be using API endpoints, the increased number of individual projects will be a source of increased risk of exposed APIs. In many companies, where DevOps is widely used, the security team will only step in at the deployment stage, which means, that after the product has been deployed, the security team will perform tests and look for security vulnerabilities. A survey from 2017 shows that the DevOps aspect was only taken into account on average by 20% of the enterprise security architects. [11] It is clear that it is not effective to only let security teams be an isolated part of the process after the deployment. It is in fact a bottleneck because just before the release, the product will go through a security audit only to then be reported back to the software engineers.

4.1 DevSecOps

An emerging discipline that is getting more popular this year is DevSecOps. It is mostly defined as an extension of DevOps where security controls and processes are adopted into and throughout the development cycle in DevOps instead of only before the release of the application. [11]

DevSecOps effectively changes the meaning of a security expert in a software engineering organization. They typically do not have the responsibility to create and run security tests or run different vulnerability scans. But rather, they become the policymakers, and they also are the creators of specific security rules that the developer must adhere to. The developer has thus to be more mindful of security and take the security aspect into account when creating software. The security expert, in a DevSecOps landscape as opposed to a DevOps landscape will also be the decider of tools that will be used when scanning for vulnerabilities in the continuous integration. Security experts will train Developer and Operations teams on how to interpret the output of various vulnerability scanning tools and security tests and teach how to resolve the inevitable issues that will be detected.

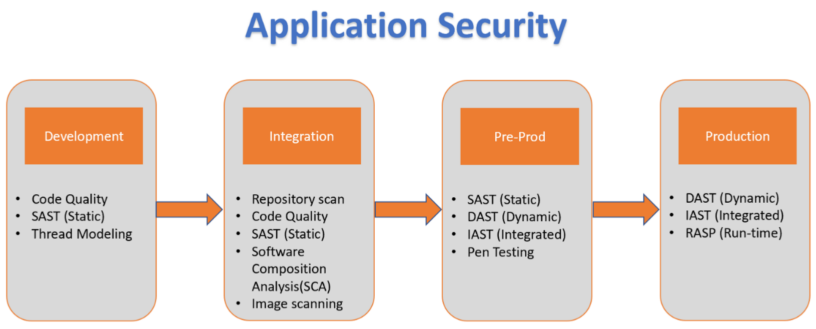
4.1.1 Software composition analysis

Software composition analysis or commonly known as SCA is a category of tools that is an integral part of DevSecOps. The purpose of SCA tools is to identify and then examine the open source components in their projects and report any quality issues, licensing issues, or security vulnerabilities regarding those components. One example of such a tool is Snyk Open Source.[12] Snyk detects vulnerabilities in the dependencies in a project and immediately creates a PR that auto-solves the issues, that in turn, the developers will have to look over.

4.1.2 Statistic Application Security Testing

Also known as SAST, a category of DevSecOps tools that is quite similar to SCA. SAST puts emphasis on the proprietary code. Which mean, code that is written by the developer. SAST looks out for buffer overflows, SQL injections, and other vulnerabilities. SAST does not look for compliance issues, something that SCA does. An example of SAST is Klocwork. [13] Klocwork is a code analyzer or SAST tool that can be integrated into the CI/CD process or on a container basis.

Figure 2: Different categories of DevSecOps tools in the context of the DevOps process. [14]



4.1.3 Dynamic Application Security Testing

Commonly abbreviated as DAST, this category of tools is very similar to SAST, but instead, it is deployed in a later stage of the cycle. DAST does not need access to the source code but rather it tests the application by injecting malicious input into it and thus discovering security vulnerabilities.

4.1.4 DevSecOps and the its future

DevOps is now ubiquitous. Most software enterprises are adhering to DevOps rules. DevSecOps is still yet to be widely adopted. In a survey by GitLab in the fall of 2021, over 70% of organizations had still not included security in the DevOps process. [15] The nature of DevOps will inevitably lead to tokens, access keys and other critical information to be passed around. This will without a doubt cause a lot of vulnerability risks. That is why DevSecOps should be an integral part of all software enterprises. In order to avoid major security breaches. And hopefully, this is what the future holds for DevOps.

5 AIOps

Even though DevOps has been groundbreaking in the realm of production and deployment of applications and services to customers, there are still other challenges to face. Usually, when an incident occurs in a project, it is up to the SRE (Site Reliability Engineer) and the DevOps team to find and resolve the problem. Using developers to solve a single problem is time-consuming and inefficient and therefore this problem can be delegated to AIOps.

Instead of having developers solving issues that could be solved by AI and machine learning methods, those fixes can be automated. [16] AIOps introduces transparency into the performance and dependencies across all the platforms of an enterprise. It collects data from the platforms, extracts the data, and analyzes it. The purpose of the scan of data is to pinpoint slow-downs or outages that have occurred in the system and alerts the operations team to the root of the problem. It also recommends a series of steps for a solution and can prevent issues before they occur. The data collected can be from historical performance, event data, streaming real-time operations events, related document-based data, system logs, or network data. [17]

5.0.1 Use cases for AIOps

There are different programs that are developed to integrate AIOps into different businesses and applications. The benefits of AIOps are that it provides organizations the simplicity of monitoring applications when there are multiple factors such as multiple environments, virtualized resources, and dynamic infrastructure. It also is useful for organizations that want to migrate to a cloud environment, for most of these organizations the migration is gradual and therefore results in a hybrid multi-cloud environment with multiple interdependencies. There are different applications out there such as Dynatrace and AppDynamics which specialize in AIOps. [17]

6 Reflection

With the advancement of tools within the software industry, it is looking like many building blocks within DevOps are being automated. This frees up time for the developer to put that time into innovating new features and applications rather than spending that time solving operational issues. One clear pattern that emerges when examining the state of the art research on the area, is that the whole process surrounding DevOps is becoming more and more automatized, which may very well be the first sign of human developers being obsolete in the future. From a software engineer's point of view it is to some extent daunting to read about the rise of automatization.

References

[1] *What is devops?* `https://www.atlassian.com/devops/what-is-devops/history-of-devops`, Accessed: 2022-05-08.

[2] *Continuous integration*, `https://www.atlassian.com/continuous-delivery/continuous-integration`, Accessed: 2022-05-08.

[3] M. Monperrus, "Automatic software repair: A bibliography," *ACM Computing Surveys (CSUR)*, vol. 51, no. 1, pp. 1–24, 2018.

[4] A. Marginean, J. Bader, S. Chandra, *et al.*, “Sapfix: Automated end-to-end repair at scale,” in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, IEEE, 2019, pp. 269–278.

[5] *Finding and fixing software bugs automatically with sapfix and sapienz*, `https://engineering.fb.com/2018/09/13/developer-tools/finding-and-fixing-software-bugs-automatically-with-sapfix-and-sapienz/`, Accessed: 2022-05-08.

[6] M. Monperrus, S. Urli, T. Durieux, M. Martinez, B. Baudry, and L. Seinturier, “Repairnator patches programs automatically,” *Ubiquity*, vol. 2019, no. July, pp. 1–12, 2019.

[7] *Code review best practices*, `https://blog.palantir.com/code-review-best-practices-19e02780015f`, Accessed: 2022-05-15.

[8] F. E. Zanaty, T. Hirao, S. McIntosh, A. Ihara, and K. Matsumoto, “An empirical study of design discussions in code review,” in *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, 2018, pp. 1–10.

[9] *Codeguru*, `https://aws.amazon.com/codeguru/`, Accessed: 2022-05-15.

[10] R. Chen, S. Li, and Z. Li, “From monolith to microservices: A dataflow-driven approach,” in *2017 24th Asia-Pacific Software Engineering Conference (APSEC)*, IEEE, 2017, pp. 466–475.

[11] H. Myrbakken and R. Colomo-Palacios, “Devsecops: A multivocal literature review,” in *International Conference on Software Process Improvement and Capability Determination*, Springer, 2017, pp. 17–29.

[12] *Snyk open source*, `https://snyk.io/product/open-source-security-management/`, Accessed: 2022-05-15.

[13] *Klocwork*, `https://www.perforce.com/products/klocwork`, Accessed: 2022-05-15.

[14] *Continuous application security in devsecops*, `https://shailender-choudhary.medium.com/continuous-application-security-in-devsecops-1590c48eeb8`, Accessed: 2022-05-08.

[15] *Embrace devsecops*, `https://tlconsulting.com.au/embrace-devsecops/`, Accessed: 2022-05-15.

[16] *How to integrate aiops for devops?* `https://www.xenonstack.com/blog/integrate-aiops-for-devops`, Accessed: 2022-05-08.

[17] *Aiops*, `https://www.ibm.com/se-en/cloud/learn/aiops`, Accessed: 2022-05-08.