DevOps phases across Software Development Lifecycle

This paper was downloaded from TechRxiv (<https://www.techrxiv.org>).

LICENSE

CC BY 4.0

SUBMISSION DATE / POSTED DATE

09-11-2020 / 06-01-2021

CITATION

Gokarna, Mayank (2020): DevOps phases across Software Development Lifecycle. TechRxiv. Preprint.
<https://doi.org/10.36227/techrxiv.13207796.v2>

DOI

[10.36227/techrxiv.13207796.v2](https://doi.org/10.36227/techrxiv.13207796.v2)

DevOps phases across Software Development Lifecycle

Mayank Gokarna, DevOps and Cloud Architect,
IBM India Pvt Ltd. Email: mgokarna@in.ibm.com

Abstract: DevOps is the combination of cultural mindset, practices, and tools that increases a team's ability to release applications and services at high velocity. The development and operations teams always have a conflict around the scope of responsibility. With these differences the quality and speed of delivery across software Development Life Cycle is negatively impacted. DevOps is about removing the barriers between two traditionally delimited teams, development and operations. With DevOps, these two teams work together to optimize both the productivity of developers and the reliability of operations. They strive to communicate frequently, increase efficiencies, and improve the quality of services they provide. They take full ownership for their services, often beyond where their stated roles or titles have traditionally been scoped. Transitioning to DevOps requires a change in culture and mindset first. It is quite difficult to persuade a whole company to change its culture at once [1]. This paper aims to bring different phases of software development lifecycle into DevOps implementation strategy and presents a comprehensive collection of leading tools used across Software Development life Cycle to automate and integrate different stages of software delivery. This paper also highlights on DevOps practices which span across different phases of the Software Development Lifecycle and how those can be implemented with different tools available.

Keywords: DevOps, Continuous Integration, DevOps lifecycle, Continuous Delivery, Agile and DevOps, Cloud Computing, Software Delivery

1. Introduction

The objective of DevOps is to make the software delivery lifecycle as efficient as possible to enable more business change to be delivered and, ultimately, so efficient that business change can be delivered continuously. DevOps is implemented through a combination of people, process and tooling. Continuous Integration and Continuous Delivery also known as CI/CD are a critical component in DevOps [2]. CI/CD enables to deliver the code faster and securely. DevOps emphasizes on people working together and accepting the change [3]. It helps in releasing of high-quality software and at a faster delivery speed. It encourages a culture of open-mindedness, predictability, cross-skill training and performing shared tasks. Often implementing DevOps is easier said than done. For solving the problem of integration between different environments and tools, DevOps tools play a critical role [4]. These tools once implemented blend different systems together and work in a continuous serving fashion enabling any change in the source code to propagate across different environments [5]. A lot of Organizations face the challenge of selecting a fit for purpose tool and selection of appropriate DevOps Tooling is an important criterion to automate and integrate different systems together. Figure 1 below shows various DevOps tools which are spread across different phases of DevOps. In general, a lot of software delivery programs uses Open-Source tools as they continue to evolve further. Organizations have also developed internal frameworks to calculate the maturity level of the DevOps implementation in a project.

Figure 1. DevOps tools at various stages

Jira is a Plan and track tool which helps in sprint planning, creating stories and issues and track the issues with labelled priority. Confluence is a documentation tool which serves as a knowledge base at a common place. Gitlab is a hosted version control tool to store the code committed by developers. There are various IDE tools where developers write and compile the code locally. Docker is a container technology which isolates the application in a self-contained environment. Jenkins is a continuous integration server which helps to continuously integrate the code into different environments [6]. Artifactory is a repository manager tool which stores the build artefacts and libraries centrally. SonarQube is an open-source static code analysis tool and helps in analysing the code against different code quality rules defined in the quality profile [7]. Cucumber and Selenium are test automation frameworks for automated testing supporting multiple programming languages. Ansible is a configuration management tool which automates the configuration of applications and infrastructure too. Grafana is a monitoring tool which provides charts, graphs, and alerts for the web when connected to supported data sources. Elasticsearch, Logstash and Kibana commonly called as ELK stack provides logging and monitoring capabilities into a DevOps tooling.

2. DevOps and Agile

DevOps integrates Agile processes with operations. Agile and DevOps are not the same, but they do go well together. Both Agile and DevOps share in common the idea of getting fast feedback [8] as depicted in the figure 2. In Agile, a review is done at every sprint. DevOps is similarly built on the idea of getting fast feedback, starting with during Continuous Integration ("Is our code still working after a change?") going on to Continuous Testing ("Does it do what it should?") all the way to continuously monitoring an application in production to get feedback to apply to the next version. So, one could argue that Agile is a subset of DevOps that is centered around the practice of collaborative development and one would not be wrong. However, it is important to note that a project can be agile without employing much of the DevOps practices and another project can use a waterfall methodology and still employ DevOps practices throughout.

Figure 2. Agile and DevOps together

3 DevOps Lifecycle

DevOps looks at the complete Software Development Lifecycle, breaking it down into four phases which further spans to six areas for scope coverage as shown in the figure 3:

- I. Plan & Measure,
- II. Develop & Test,
- III. Release & Deploy, and
- IV. Monitor & Optimize

Figure 3. DevOps scope across SDLC

3.1 Continuous Business Planning

Continuously plan, measure and bring business strategy and customer feedback into the development lifecycle. Continuous Business Planning assures that the investment choices are aligned throughout their life cycle to your organization's needs. It employs lean principles to help teams start small, identifying the outcomes and resources needed to test the business vision and value. Adapt and adjust continually, measuring actual progress, and customer requirements so teams can shift direction with agility [9]. Product backlog is maintained and gets split into smaller releases and milestones based on the priority decided by the various stakeholders. Following are the major components in Continuous Business planning:

- Requirements Management
- Change Management
- Project Planning and Work Management

- Iteration Planning
- Progress Monitoring and Reporting

3.2 Collaborative Development and Continuous Integration

Collaborative Development brings together business, development, and QA organizations to deliver innovative, quality customer experiences with speed. It dramatically transforms the way clients build their apps in order to drive real business outcomes.

The technological enablers for DevOps establish DevOps capabilities by automating different trivial and non-trivial tasks [10]. Automation helps in achieving continuous delivery and deployment by providing a DevOps pipeline to production for all changes to a given software environment [11].

3.3 Continuous Testing

Utilizes highly automated approaches to dramatically reduce the time and effort in delivering quality results [12]. Eliminates testing bottlenecks through virtualized dependent services. It simplifies the creation of virtualized test environments. Continuous Testing also reduces the costs of provisioning and maintaining the various test environments.

Figure 4. Test Automation Pyramid

Most of the software testing activities are based on the test-driven development [13] which is supported by continuous integration practises. Developers are required to write unit test cases to integrate their code frequently [14]. These tests are executed repeatedly in the automated fashion by means of continuous integration as shown in the figure 4 above.

3.4 Continuous Release & Deployment

There are various advantages of continuous release and deployment. Continuous deployment helps in receiving an early feedback from the customers and users. It enables frequent and reliable releases which improves customer satisfaction [15], trust and product quality [16]. It also enables teams to perform on-demand Dev/Test self-deployments to continuously integrate software driven innovation. Figure 5 below shows a high-level build and deployment cycle till Production environment.

Technological Enablers:

- Release Management
- Build Automation
- Deployment Automation

Figure 5. Build and Release cycle

3.5 Continuous Monitoring

Continuous Monitoring automates and optimizes the ability to monitor and manage the performance and availability of applications and infrastructure continuously. It tells how good my systems are performing and whether it needs any correction [17]. Choice of correct tool decides the level of correctness which can be brought into the system. Following are the main areas for monitoring in a DevOps implemented project:

- Application Performance Management
- IT Operations Analytics
- Operational Problem Management

Logging needs to be used effectively to monitor application logs. Logging mechanism works at multiple levels and should be configured to analyse the logs by drilling down when needed [18].

The logs are analysed to provide quality feedback to the development teams and then perform a deeper problem analysis to go to the root of the issue. The results of the log analysis help the development and Operations team to prepare better for future.

3.6 Continuous Customer Feedback and Optimization

Continuously getting customer's feedback on their experience brings far better insights that form the basis for improved decision making and optimization. An early and rapid feedback from the systems help to save time and improve the end user experience on continuous basis [19]. The DevOps maturity model enables growth through continuous learning among development and operations teams as depicted by figure 6 below. The capabilities and skills

defined, guarantees enhanced ability to handle issues of scale and complexities. Information Technology Infrastructure Library, ITIL is a framework of best practices for IT service management which focuses on aligning IT services according to the needs of the business. The degree of DevOps implementation can also be measured by means of maturity models against various matrices [20].

Figure 6 Continuous Feedback

4 Role of DevOps tools

I conducted a study on various DevOps tools used in the leading Organisations for DevOps implementation at different stages of Software Delivery. Figure 7 below shows this list of various tools used in the projects pertaining to Retail, Banking and Automobile industries. Each of these tools serving one of the DevOps phases are unique in its own ways and provide different features while making optimum use of the tools for the right purpose [21]. These tools are used to design, build, deploy, test, monitor, manage and operate software and systems connected as one integrated pipeline. Tools are broadly classified as Commercial and Open Source tools. With the Open Source community growing large and enabling state of the art functionalities, a lot of Enterprises have moved to Open Source from Commercial tools also because of the high cost of licencing. Jenkins is an open source Continuous Integration and automation server which works on plugins-based architecture [22] and has the capability to integrate variety of tools enabling Continuous Integration/ Continuous Delivery also known as CI/CD toolchain and it can build deploy and test across multiple platforms [23]. With the invent of Cloud Computing, various services are available for performing automation tasks which are cloud native to support the platform developed on Cloud [24]. Security services is an added advantage which comes with Cloud providers. The cost of these services depends on the volume and usage across the platform.

	DevOps Phases					
	Continuous Business Planning	Collaborative Development and Continuous Integration	Continuous Testing	Continuous Release & Deployment	Continuous Monitoring	Continuous Customer Feedback and Optimization
Tools	Trello	Jenkins	JUnit	Ansible	Nagios	Open Web Analytics
	JIRA	Git	Selenium	Chef	Grafana	Webalizer
	Redmine	Gerrit	JMeter	Docker	Kibana	W3Perl
	Trac	SonarQube	Cucumber	IBM Urban Code	Prometheus	ServiceNow
	Rally	Maven	Microfocus UFT	Kubernetes	Logstash	Slack
		Gradle	TestNG	Puppet	AppDynamics	Flowdock
		Apache Ant	Tricentis Tosca	Go	Datadog	
		Make	Jasmine	BMC Release	Dynatrace	
		Microsoft Build		XL Release	Splunk	
		Travis CI			Icinga	
		TeamCity			PagerDuty	

Figure 7 Tools across DevOps phases

5 Conclusion

The origin of DevOps occurred to fill the gaps between Development and Operations teams. DevOps implementation is considered to be successful when it majorly relies on the integrated solutions provided by different tools which span across the Development lifecycle and helps removing manual steps, reduce errors, increase team performance and capabilities, increases agility and bridge the gaps between isolated teams [25]. DevOps tools play a key role in automating the routine tasks which otherwise needs manual intervention in different phases of Software Development Life Cycle. The tools are evolving quite frequently and are fulfilling the requirements raised by modernization of applications and migration to cloud. With the inception of Cloud and with lot of Organisations migrating and planning to migrate to cloud opens new dimensions of DevOps implementation with availability of cloud native and existing DevOps tools.

References

- [1] QUDOS 2016: Proceedings of the 2nd International Workshop on Quality-Aware DevOps July 2016 Pages 7–11 <https://doi.org/10.1145/2945408.2945410>
- [2] Hemon, A., Lyonnet, B., Rowe, F. *et al.* From Agile to DevOps: Smart Skills and Collaborations. *Inf Syst Front* **22**, 927–945 (2020). <https://doi.org/10.1007/s10796-019-09905-1>
- [3] <https://dzone.com/articles/know-more-about-devops-culture-sdlc-and-agile-dev> (last accessed 02 Nov 2020)
- [4] Kaiser A.K. (2021) Brief Overview of DevOps. In: Become ITIL® 4 Foundation Certified in 7 Days. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-6361-7_2
- [5] Couto, L.D., Tran-Jørgensen, P.W.V., Nilsson, R.S. *et al.* Enabling continuous integration in a formal methods setting. *Int J Softw Tools Technol Transfer* **22**, 667–683 (2020). <https://doi.org/10.1007/s10009-019-00546-y>
- [6] S. Mysari and V. Bejgam, "Continuous Integration and Continuous Deployment Pipeline Automation Using Jenkins Ansible," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020, pp. 1-4, doi: 10.1109/ic-ETITE47903.2020.239.
- [7] J. Xue, X. Mao, Y. Lu, Y. Yu and S. Wang, "History-Driven Fix for Code Quality Issues," in *IEEE Access*, vol. 7, pp. 111637-111648, 2019, doi: 10.1109/ACCESS.2019.2934975.
- [8] N. Govil, M. Saurakhia, P. Agnihotri, S. Shukla and S. Agarwal, "Analyzing the Behaviour of Applying Agile Methodologies & DevOps Culture in e-Commerce Web Application," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), Tirunelveli, India, 2020, pp. 899-902, doi: 10.1109/ICOEI48184.2020.9142895.
- [9] F. Moyon, K. Beckers, S. Klepper, P. Lachberger and B. Bruegge, "Towards Continuous Security Compliance in Agile Software Development at Scale," 2018 IEEE/ACM 4th International Workshop on Rapid Continuous Software Engineering (RCoSE), Gothenburg, Sweden, 2018, pp. 31-34.
- [10] Leonardo Leite, Carla Rocha, Fabio Kon, Dejan Milojevic, and Paulo Meirelles. 2019. A Survey of DevOps Concepts and Challenges. *ACM Comput. Surv.* **52**, 6, Article 127 (January 2020), 35 pages. DOI:<https://doi.org/10.1145/3359981>
- [11] EASE'18: Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018 June 2018 Pages 57–67 <https://doi.org/10.1145/3210459.3210465>
- [12] P. Agrawal and N. Rawat, "Devops, A New Approach To Cloud Development & Testing," 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), GHAZIABAD, India, 2019, pp. 1-4, doi: 10.1109/ICICT46931.2019.8977662.
- [13] M. Kessel and C. Atkinson, "Integrating Reuse into the Rapid, Continuous Software

Engineering Cycle through Test-Driven Search," 2018 IEEE/ACM 4th International Workshop on Rapid Continuous Software Engineering (RCoSE), Gothenburg, Sweden, 2018, pp. 8-11.

[14] P. Agrawal and N. Rawat, "Devops, A New Approach To Cloud Development & Testing," 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), GHAZIABAD, India, 2019, pp. 1-4, doi: 10.1109/ICICT46931.2019.8977662.

[15] K. Kuusinen et al., "A Large Agile Organization on Its Journey Towards DevOps," 2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Prague, 2018, pp. 60-63, doi: 10.1109/SEAA.2018.00019.

[16] ASWEC '15 Vol. II: Proceedings of the ASWEC 2015 24th Australasian Software Engineering Conference September 2015 Pages 147–148 <https://doi.org/10.1145/2811681.2824996>

[17] B. Chen, "Improving the Software Logging Practices in DevOps," 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSECompanion), Montreal, QC, Canada, 2019, pp. 194-197, doi: 10.1109/ICSECompanion.2019.00080.

[18] G. B. Ghantous and A. Gill, "The DevOps Reference Architecture Evaluation : A Design Science Research Case Study," 2020 IEEE International Conference on Smart Internet of Things (SmartIoT), Beijing, China, 2020, pp. 295-299, doi: 10.1109/SmartIoT49966.2020.00052.

[19] K. Singi, V. Kaulgud, V. S. Sharma, N. Dubash and S. Podder, "Test Optimization from Release Insights: An Analytical Hierarchy Approach," 2017 IEEE/ACM 3rd International Workshop on Rapid Continuous Software Engineering (RCoSE), Buenos Aires, 2017, pp. 16-19, doi: 10.1109/RCoSE.2017.2.

[20] M. A. McCarthy, L. M. Herger, S. M. Khan and B. M. Belgodere, "Composable DevOps: Automated Ontology Based DevOps Maturity Analysis," 2015 IEEE International Conference on Services Computing, New York,

NY, 2015, pp. 600-607, doi: 10.1109/SCC.2015.87.

[21] iiWAS '16: Proceedings of the 18th International Conference on Information Integration and Web-based Applications and Services November 2016 Pages 209–213 <https://doi.org/10.1145/3011141.3011200>

[22] G. B. Ghantous and A. Gill, "The DevOps Reference Architecture Evaluation : A Design Science Research Case Study," 2020 IEEE International Conference on Smart Internet of Things (SmartIoT), Beijing, China, 2020, pp. 295-299, doi: 10.1109/SmartIoT49966.2020.00052.

[23] J. Shah, D. Dubaria and J. Widhalm, "A Survey of DevOps tools for Networking," 2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York City, NY, USA, 2018, pp. 185-188, doi: 10.1109/UEMCON.2018.8796814.

[24] Mary Sánchez-Gordón and Ricardo Colomo-Palacios. 2018. A Multivocal Literature Review on the use of DevOps for e-Learning systems. In Proceedings of the Sixth International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'18). Association for Computing Machinery, New York, NY, USA, 883–888. DOI:<https://doi.org/10.1145/3284179.3284328>

[25] M. Virmani, "Understanding DevOps & bridging the gap from continuous integration to continuous delivery," Fifth International Conference on the Innovative Computing Technology (INTECH 2015), Pontevedra, 2015, pp. 78-82, doi: 10.1109/INTECH.2015.7173368.