Towards Definitions for Release Engineering and DevOps

Andrej Dyck
RWTH Aachen University
Software Construction
Ahornstraße 55
52074 Aachen, Germany
andrej.dyck@swc.rwth-aachen.de

Ralf Penners
RWTH Aachen University
Ahornstraße 55
52074 Aachen, Germany
ralf.penners@rwth-aachen.de

Horst Lichter
RWTH Aachen University
Software Construction
Ahornstraße 55
52074 Aachen, Germany
horst.lichter@swc.rwth-aachen.de

***Abstract*—Delivering software as fast as possible is essential for software development organizations, in order to keep in pace with competitors. Accordingly, release engineering, a software engineering discipline concerned with the delivery and deployment process, plays an important role in such organizations. Further, in recent years, the term DevOps gained popularity in the IT world. It describes an approach to improve the collaboration between development and IT operations teams, in order to streamline software engineering processes.**

To the best of our knowledge, there is no uniform definition for neither of these terms, and thus, many inadequate or even wrong definitions exist. Consequently, these terms are often confused or even used as synonyms. In this paper, we will tell those two terms apart by contrasting available definitions and descriptions for both of them. Additionally, we propose uniform definitions for release engineering and DevOps, which we developed in cooperation with experts.

I. Introduction

Lately, more and more software development organizations recognize the importance of delivering high-quality software fast, reliable, and predictable. Yet, they often struggle to implement proper approaches and practices of the software delivery process. Moreover, trying to get the bottom of term like release engineering and DevOps without an explanation or definition adds further confusion.

In short, release engineering is a discipline concerned with the delivery process of a software product and DevOps is a organizational approach to reduce barriers between teams in a software organization, and thus, accelerate development and deployment. Given that both are addressing the problem of delivering software and existing descriptions lack of precision or are even wrong (e.g., Wikipedia states that release engineering is "also known as DevOps" [1]), it is not surprising that people get confused. To tackle this deficiency, we propose uniform definitions for both terms, which we discussed with experts from these fields.

In section II, we explain both terms, release engineering and DevOps, and point out the differences. Subsequently, in section III, we present existing definitions and propose uniformed definitions in section IV. Finally, we will give a short discussion in section V, and provide a perspective for future research and conclude this paper in section VI.

II. Release Engineering vs. DevOps

Release engineering and DevOps, both, try to improve the product value, and thus, the customer added value, by enabling the organization to deliver high-quality software, and react to changes and issues faster. However, available descriptions and inadequate, or even wrong, definitions for release engineering and DevOps make it difficult to grasp either of those terms. Moreover, existing sources often focus strongly on how to achieve the overall goals, i.e., what practices and tools to use, while the meaning or ideas are often explained either not at all or only inadequately. Subsequently, we contrast both terms.

As part of a software project, release engineering is concerned with establishing a reliable and predictable delivery process of the software product. Moreover, release engineering "[accelerates] the path from development to operations", while "[keeping] the big picture in mind" as outlined figuratively by Dinah McNutt [2]. In other words, release engineering is a task of "building a pipeline", as described by John O'Duinn [3]; meaning, creating a tailored plan to improve the delivery process and information flow between the people involved, and then, making use of established practices (e.g., continuous delivery, canary, dogfooding) and tools (e.g., build and test tools like Jenkins, version control systems like Git, system configuration tools like Puppet, package manager like APT, etc.) to realize, automate, and adapt the plan.

The term DevOps first emerged in 2009 as a Twitter hashtag at the first DevOpsDays in 2009 [4] and was influenced by several talks and papers; above all, the talks by John Allspaw and Paul Hammond at Velocity 2009 [5], [6]. The history is nicely summarized by Damon Edwards and Michael Hüttermann [7], [8]. The heart of DevOps are cultural aspects like "good information flow, cross-functional collaboration, shared responsibilities, learning from failures, and encouragement of new ideas" [9]. Or as interpreted by Katherine Daniels: empathy between the teams of an organization [10]. In other words, DevOps aims to establish a mindset that focuses on a closer collaboration between teams by setting the common goal to develop high-quality software and operate resilient systems. Here, similar to release engineering, established practices and tools can help forming such a culture.

In conclusion, release engineering is concerned with the improvement of the delivery process in a holistic way, while DevOps establishes a culture – or mindset – to reduce communication barriers and encourage empathy. Note that although DevOps is an abbreviation of development and IT operations, it is not limited to those two teams. Both, release engineering and DevOps, enable productivity in form of reliable deployment frequencies with less failures [9], which eventually increases the product value. To this end, they rely on established practices and tools.

III. Existing Definitions

To our knowledge, there is no uniform definition for the terms release engineering and DevOps. As a consequence, many people use their own definitions or rely on others, which results in confusion about those terms. Subsequently, we want to list some of these definitions and discuss why – in our opinion – they are inadequate or wrong.

Many people rely on Wikipedia as source of information. Hence, we start with the definitions given by the English Wikipedia articles for release engineering as well as DevOps [1], [11]. Here, release engineering is described as "a sub-discipline in software engineering concerned with the compilation, assembly, and delivery of source code into finished products or other software components." This definition is missing the aspects of improvement, reliability, and predictability. Moreover, the article states that release engineering is "also known as DevOps", while DevOps is defined as "a software development method that stresses communication, collaboration and integration between software developers and Information Technology (IT) professionals." Note that there is already a contradiction. Moreover, we disagree that DevOps is a development method, nor is it a development process model like RUP, nor a project management framework like SCRUM.

A blog post on Goat Can argues that DevOps should be defined by the community and that Wikipedia should hold this definition [12]; "This allows the definition to evolve and change over time as the industry evolves and changes". We do not argue with that; however, a wrong definition on Wikipedia as discussed, is, on one hand, bad for communicating the idea, and, on the other hand, not usable for scientific research. Moreover, there are may similar cases where some people thought a definition is not needed, e.g., the agile movement, which is now defined by ISO/IEC/IEEE 26515:2011.

Two popular definitions for DevOps are given by Hüttermann in [8] and Gene Kim in [13]. In Hüttermann's definition, DevOps is "a mix of patterns intended to improve collaboration between development and operations [..]" This definition suggests that if enough of those patterns are used, then an organization has successfully implemented DevOps. Moreover, it lacks a goal, i.e., why the collaboration should be improved. Kim provides another definition for DevOps: "The term 'DevOps' typically refers to the emerging professional movement that advocates a collaborative working relationship between Development and IT Operations, resulting in the fast flow of planned work (i.e., high deploy rates), while simultaneously increasing the reliability, stability, resilience and security of the production environment." While Kim's definition describes the goals and the collaboration aspect, it focuses only on development and IT operations. Moreover, we strongly believe that the term DevOps can define more than just a movement.

IV. Definitions

Subsequently, we propose a uniform definition for the term release engineering as well as DevOps. To this end, we consolidated with experts from both fields, to boil down the definitions to their essentials. First, we will discuss and present the definition for release engineering. Afterwards, we do the same for DevOps.

A. Release Engineering

Discussing release engineering with Bram Adams (personal communication, November 14, 2014) and Jez Humble (personal communication, November 13, 2014), we agreed, that the deployment of high-quality software is a central goal of release engineering. A deployed faulty software product decreases the customer added value and diminishes its reputation. However, the deployment has to be fast, reliable, and predictable.

Based on O'Duinn's "building a pipeline", we see three main tasks of a release engineer: development, implementation, and improvement of the deployment process. Starting with the development of a custom release process for an organization, suitable methods and tools have to be established and introduced. Then, after a reliable and predictable release process is established, it needs to be improved, in order to increase efficiency, and eventually, the product value.

Adams suggested (personal communication, November 11, 2014) to specify the tasks as "[..] processes to integrate, build, test, package and deliver [..]"; similar to Wikipedia's definition. We refrained from doing so, since such a specific listing might suggest release engineering comprising only those tasks. As described in the previous sections, release engineering is concerned with many different tasks and responsibilities throughout the whole software project; for instance, the information flow between the people involved in the project.

With this in mind, it becomes clear that release engineering is a software engineering discipline. Hence, we propose the following definition:

> ***Release engineering*** *is a software engineering discipline concerned with the development, implementation, and improvement of processes to deploy high-quality software reliably and predictably.*

Note that, here, improving processes also mean, i.a., improving information flow between the people involved in the project. In order to improve the release process, a release engineer has to establish a good communication between those responsible for various tasks and stress acceptance for other team's goals.

As a marginal note, Humble suggested to use the words delivery and deployment simultaneously. We decided to limit

this definition to deployment, since, first, deployment includes delivery (cf. Martin Fowler's contrast of continuous delivery and continuous deployment [14]), and, second, only deployed high-quality software adds to the customer added value.

Finally, Dinah McNutt and Akos Frohner, stated that this definition covers what they mean at Google. Moreover, Kim Moir and Gene Kim stated that they like this definition and Chuck Rossi noted that all keywords he considers critical in release engineering – namely reliability, predictability, and deployment – are covered.

B. DevOps

In general, there seems to be common ground that DevOps is about improving communication and collaboration between teams in an organization or project. However, the purpose, i.e., why an improved collaboration should be achieved, is often missing or differs strongly. Adams suggested that the goal of DevOps is the acceleration of feedback, higher quality of software, and faster deployment (personal communication, November 11, 2014). Concerning this, Jeff Sussna commented that DevOps is about more than fast deploying high-quality software; "it is about operating software services [and delivering] change to those services" (personal communication, November 25, 2014). In conclusion, both, development and operational goals, are equally important, as developing and maintaining systems should be performed by developers and operators jointly. Akos Frohner, Chuck Rossi, and Patrick Debois tend to agree with these points.

Jezz Humble proposed this definition: "[DevOps is a] cross-functional community of practice dedicated to the study of building, evolving and operating rapidly changing resilient systems at scale" (personal communication, November 13, 2014). His definition includes all the main goals and, additionally, stresses the focus on software and the production environment as a whole by using the term resilient systems. Considering the goals and manners in more detail, we see DevOps as an organizational approach to achieve those strategic goals. Hence, for DevOps, we propose the following definition:

> ***DevOps*** *is an organizational approach that stresses empathy and cross-functional collaboration within and between teams – especially development and IT operations – in software development organizations, in order to operate resilient systems and accelerate delivery of changes.*

V. Discussion

Release engineering seems to be quite clear, since many organizations have been doing it without naming this task. On the contrary, the term DevOps spread without a clear definition and is now used on so many things that it seems to have no longer any meaning. Additionally, new terms extending DevOps came up: acronyms like DevSecOps or DevNetOps express the inclusion of other teams. Thus, several decisive people try to give the term a new meaning, e.g., Katherine Daniels and Jeff Sussna interpret DevOps as empathy or promise theory, respectively [10], [15]. We even argue that empathy is not enough and altruism, i.e., the wish that other people may be happy, should be practised. Such interpretations can help to improve service quality by increasing collaboration within and between systems, people, and organizations. Consequently, the term DevOps does not limit the collaboration to only between developers and IT operations; thus, there is no need for other "extending" acronyms.

While release engineering can and should be applied by all software development organizations, the question of what organizations can implement DevOps arises. The operations part of the term suggest that DevOps is only applicable for organizations that operate their own software, e.g., web-sites. However, if, on one hand, widening the term operations to operating infrastructure (e.g., development, build, test, release, etc.) and, on the other hand, not differentiating between internal and external operations, then all software development organizations can implement DevOps. In fact, the heart of DevOps, i.e., the mindset of empathy or promise theory, is not limited to software development organizations.

Gene Kim considers DevOps to be a super-set of release engineering (personal communication, November 15, 2014). We disagree, as an organizational approach and a discipline cannot be compared at the same level. However, both approaches relate and influence on one another. For instance, Chuck Rossi, release engineer, mentioned that DevOps gave him a name for the things they have been doing at Facebook organically in their release process even before the term came up [16]. Moreover, similar, or even the same, practices, techniques, and tools are used to support establishing the culture and processes, respectively. Following, we want to elaborate on that.

Comparing software engineering to classical engineering, e.g., construction engineering, the latter rely strongly on both, established processes and good collaboration between teams (e.g., construction workers and architects). Accordingly, a good communication within and between teams influences the proper implementation of processes and the improvement of such requires collaboration. Hence, release engineers can establish processes more easily and effectively in organizations that implement DevOps, since people already communicate and work together. Likewise, a sophisticated release engineering supports and intensifies the mindset of DevOps; here, release engineering works as a connecting link between teams. In conclusion, implementing both, release engineering and DevOps in an organization increase the effectiveness of productivity, and with that, eventually, the product value.

VI. Conclusion

In this paper, we aimed to showcase the differences and relationship between the terms release engineering and DevOps. To this end, we briefly contrasted both and discussed why existing definitions are inadequate or wrong. Seeking to take action against the lack of uniform definitions, we collaborated with experts in order to propose such for both, release engineering and DevOps. In short, the contrast is while release engineering is a discipline concerned with building and improving a predictable and reliable release process, DevOps is

an organizational approach stressing on good communication and empathy between teams. As a result, if an organization is implementing DevOps, release engineering becomes more effective, and a successful release engineering intensifies DevOps.

Admittedly, our research has a threat to validity, since such uniform definitions, which satisfy many, are hard to find. Some consider that there is no need for definitions, e.g., the post [12], or that many existing definitions consider other aspects and focus as most important, as described for DevOps by ScriptRock in [17]. Furthermore, ScriptRock stated that it is "not only necessary, but important, that DevOps [is] defined simply and in such a way that anyone in the office could understand" [17]; this also holds true for release engineering. With this in mind, we tried to formulate the definitions as simple as possible and consider the various important aspects by discussing these with experts.

In summary, we got consistent positive feedback for the definition of release engineering, while the definition for DevOps still kicks off discussions. This work should be seen as a first attempt to develop uniform definitions for both terms; hence, further work and research is needed. However, we hope that this work helps to clarify the differences and relationship between release engineering and DevOps.

VII. Acknowledgments

Many thanks to Jez Humble, Jeff Sussna, Gene Kim, Bram Adams, and Andreas Steffens for their contribution and discussions. We are equally thankful to Akos Frohner, Kim Moir, Chuck Rossi, Dinah McNutt, and Patrick Debois who gave us great reviews.

References

[1] Wikipedia, "Release engineering," http://en.wikipedia.org/w/index.php?title=Release_engineering&oldid=585442891, December 10, 2013, [Online; accessed: 11-14-2014].

[2] D. McNutt, "The 10 Commandments of Release Engineering," https://www.youtube.com/watch?v=RNMjYV_UsQ8, April 11, 2014, [Online; accessed: 11-09-2014].

[3] J. O'Duinn, "Release Engineering as a Force Multiplier," http://www.youtube.com/watch?v=7j0NDGJVROI, May 28, 2013, [Online; accessed: 11-09-2014].

[4] DevOpsDays Ghent 2009, http://devopsdays.org/events/2009-ghent/, October 30-31, 2009, [Online; accessed: 12-12-2014].

[5] J. Allspaw and P. Hammond, "10+ Deploys Per Day: Dev and Ops Cooperation at Flickr," http://www.youtube.com/watch?v=LdOe18KhtT4, June 23, 2009, [Online; accessed: 11-10-2014].

[6] Velocity 2009, http://velocityconf.com/velocity2009, June 22-24, 2009, [Online; accessed: 12-10-2014].

[7] D. Edwards, "The History Of DevOps," http://itrevolution.com/the-history-of-devops/, September 17, 2012, [Online; accessed: 11-10-2014].

[8] M. Huettermann, *DevOps for Developers*. Apress, 2012.

[9] Puppet Labs, IT Revolution Press, and ThoughtWorks, "State of DevOps Report 2014," http://puppetlabs.com/2014-devops-report, June 4, 2014, [Online; accessed: 11-13-2014].

[10] K. Daniels, "DevOps Is Dead (Long Live DevOps)," http://www.devopsdays.org/events/2014-minneapolis/proposals/DevopsIsDead/, July 18, 2014, [Online; accessed: 12-07-2014].

[11] Wikipedia, "DevOps," http://en.wikipedia.org/w/index.php?title=DevOps&oldid=633593434, November 12, 2014, [Online; accessed: 11-14-2014].

[12] mfdii, "We don't need no manifesto ...," http://goatcan.do/2015/01/16/we-dont-need-no-manifesto/, January 16, 2015, [Online; accessed: 01-16-2015].

[13] G. Kim, "Top 11 Things You Need To Know About DevOps," http://itrevolution.com/pdf/Top11ThingsToKnowAboutDevOps.pdf, June 20, 2013, [Online; accessed: 11-20-2014].

[14] M. Fowler, "Continuous Delivery," http://martinfowler.com/bliki/ContinuousDelivery.html, May 30, 2013, [Online; accessed: 01-10-2015].

[15] J. Sussna, "Promising Digital Service Quality," http://www.devopsdays.org/events/2014-minneapolis/proposals/PromisingDigitalServiceQuality/, July 18, 2014, [Online; accessed: 12-03-2014].

[16] C. Rossi, "Moving to mobile: The challenges of moving from web to mobile releases," https://www.youtube.com/watch?v=Nffzkkdq7GM, April 11, 2014, [Online; accessed: 11-09-2014].

[17] ScriptRock, "The Problem with Defining DevOps," http://www.scriptrock.com/blog/the-problem-with-defining-devops, December 3, 2014, [Online; accessed: 12-10-2014].