Turning around the Tanker

Jens Calamé[1] and Sven Euteneuer[2]

Abstract: The ever quicker changing world forces all players to adapt their pace of change. In the automotive industry, this is particularly visible in the tight integration between physical cars, on-board infotainment and telematics systems and data-driven back end systems, all with differing requirement sets, release cycles and processes. Other industries face similar challenges with complex digitalization programs. This paper describes the iterative quality-driven implementation of a change towards a nimbler approach, taking into account technological as well as organizational and processual change, supporting much shorter cycle times, better cooperation and a higher degree of automation across the full application life cycle from development via testing to operations.

Keywords: Agile, DevOps, Change process

1 Introduction

Increasing digitalization has led to a situation in which more and more everyday products are based on software-based systems. Time to market for new or revised products has decreased rapidly over the last years due to an ever more competitive situation on the market of software-based systems. Simultaneously, quality demands for these products have merely increased, especially due to the visibility of quality lacks on platforms like social networks or app stores, but also because failures in systems like cars easily become life threatening. The implications of these demands are twofold as can be seen in Fig. 1.

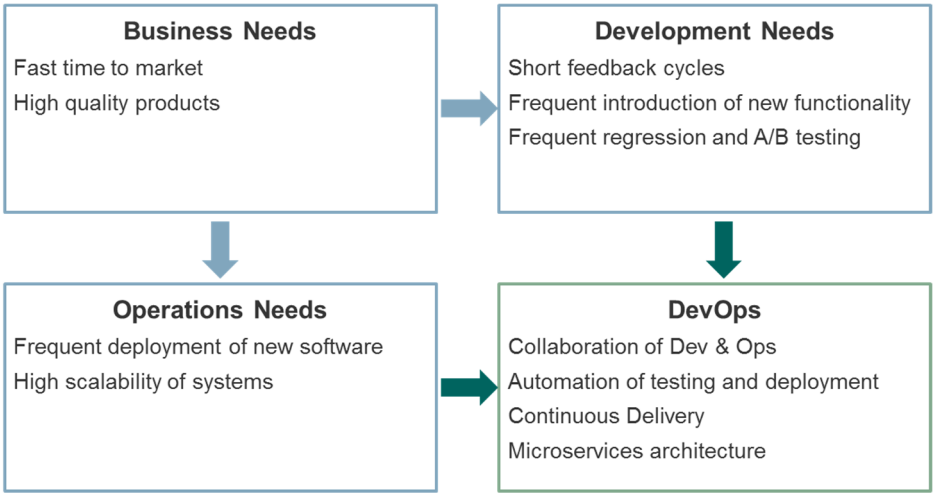

Fig. 1 Challenges in the Market and DevOps as a Solution

[1] SQS Software Quality Systems AG, Stollwerckstraße 11, 51149 Köln, jens.calame@sqs.com
[2] SQS Software Quality Systems AG, Stollwerckstraße 11, 51149 Köln, sven.euteneuer@sqs.com

Development requires short feedback cycles to be able to react fast on new customer demands or found defects. Frequently adapting a system's functionality also requires more frequent testing, especially regression testing. At the same time, small changes allow for new ways of testing functionality, namely – explicitly – employing the customer as an external tester to gain fast feedback from the end users of the system. Operations must frequently deploy new software in this process. Furthermore, systems have to be highly scalable in order not to endanger the business's success due to a lack of availability of a highly demanded feature.

DevOps supports all these demands by establishing new approaches on three levels of software development and operations:

1. Development and Operations collaborate tightly. This collaboration goes so far that we are not necessarily talking about two different teams anymore, but e.g. about feature teams, which are responsible for the development of a certain feature (starting with the contractor's part of requirements engineering) as well as its operations in the production environment.
2. Splitting an application's internal structure alongside features rather than technical layers or components (or not at all) has implications on the application's architecture. Micro services are a relatively new architectural pattern, which allows for independent development and deployment of features and by that supports feature teams, capacity scaling and A/B testing.
3. Finally, the fast time to market and the necessary short feedback cycles and frequent (regression) tests are supported by the extensive application of tools not only supporting test automation, but also the automatic creation, provisioning and deployment of test and production environments.

Most organizations do not have the luxury of starting on a green field and, thus, have the demand of transforming into a DevOps lifecycle starting from their highly heterogeneous traditional processes and software architecture. Depending on customer context, this may involve designing the target approach on a green field or may require complex considerations of existing organization, processes and technology. Particularly large organizations tend to have a large inertia and resistance to change built into their structures making them as difficult to turn around as the proverbial oil tanker.

2 The Solution

Organizations do not introduce DevOps just for the sake of change, but bind certain specific hopes to this new process. To find out what exactly those wishes are, we start the change process (cf. Fig. 2) with a series of workshops ranging from organization to processes to methods and tools such as continuous deployment pipelines. Hereby it is important to project the customer's expectations on the holistic approach of DevOps and elucidate the requirements for the different benefits of DevOps.

Fig. 2 Overview on the Approach

The workshops cover the following aspects:

1. Definition of goals for the change process and measurement of success
2. Processual requirements of technical development
3. Processual requirements of requirements engineering and testing
4. Processual requirements of environment operations
5. (Current) tooling in development and test
6. Organizational structure in the large
7. Organizational structure in detail
8. Cultural change and change process
9. (Current) tooling in operations
10. (Current) tooling in configuration management and requirements management

The change process as such is not a waterfall with all workshops in the beginning and the writing and implementation of the "right" proposals subsequently. We rather start with the first workshop (Definition of Goals) to plan the process and afterwards perform the workshops one by one, while in parallel the results of the respective previous workshop are reviewed, acknowledged, finalized and integrated into the change. This approach allows us to gain a frequent feedback by the customer throughout the change preparation in order to not overcharge his organization with a full-fledged big-bang DevOps introduction.

We will describe the scope of the workshops in the remainder of this chapter. All workshops have the goals to present good practices, discuss their adaptation to the respective customer context and to identify customer-specific constraints. The outcome of each workshop is a comparison of the current situation with what would be an ideal situation for DevOps. Depending on this information, we develop the change as such in the post-workshop phase.

2.1 Definition of goals for the change process and measurement of success

The first workshop sets the agenda for the further body of work and the change process as such. Firstly, this means to operationalize the goals of the change. This is necessary to gather the expectations of the different stakeholders to the introduction of DevOps and to normalize these expectations to the possible outcome and necessary input to the change. Secondly, we define criteria to be able to judge on the success (or failure) of the change process in the end or – partially – while being in middle of the process. The latter allows for early adjustments of the introduced processes and ensures optimal results.

Due to the importance and broad scope of this workshop, all roles (product owner, development, test, operations and management) are involved in this workshop.

2.2 Processual requirements of technical development

In this workshop, we gather all necessary information and viewpoints for the integration of development into the new DevOps culture. This includes questions on how software is currently developed, tested by development and how the technical quality of the software (code quality, architecture quality and technical debts) is managed. Another very important aspect as input for the change process is the architecture of the software and its interfacing environmental systems.

The roles involved in this workshop are the several management roles (requirements, program, release, and infrastructure) as well as development itself.

2.3 Processual requirements of requirements engineering and testing

In this workshop, we investigate the interface between the stakeholders of the software being developed and development. We investigate the engineering on functional, non-functional and operational requirements as well as how these requirements are worked out through the development process and how they affect testing. Regarding this latter aspect, we focus on the interface between development and test with special attention on testability, integration testing and test automation.

The roles involved in this workshop are requirements, release, and data management as well as development, test and operations.

2.4 Processual requirements of environment operations

In this workshop, we investigate the operation of software and if applicable hardware environments. We elaborate how configuration management and operations of the software are currently organized and how environments (both test and production) as well as test data are provisioned. We further find out how service virtualization can be put in place to support testing at several stages and how deployments and releases are currently managed and how this can be improved in a DevOps culture.

The roles involved in this workshop are requirements, infrastructure, and release management as well as development, test and operations.

2.5 (Current) tooling in development and test

In this workshop, we develop the requirements for tooling the development and test stages starting from the current tool situation at the customer. We analyze the requirements and possible tool candidates for developer tests, technical quality as well as functional and non-functional test and test automation. This analysis takes into account findings regard-

ing the target architecture of applications, the targeted technology stack as well as integrated legacy environments. During this workshop, we also define the target architecture for the continuous deployment pipelines.

In the aftermath of the workshop, the coarsely defined tooling and architecture is refined and piloted.

The roles involved in this workshop are development and test as well as environment, data, infrastructure, and release management.

2.6 Organizational structure in the large

This workshop defines the requirements to the organization of both project and program. It defines the internal organization and the size of the DevOps teams, how roles and tasks within the teams are defined, and which organizational interfaces exist between the (new) DevOps organization and its counterparts in the traditional parts of the company as well as to infrastructure management. If the IT of a company is coming from a traditional development and operations approach, this can also mean that parts of the traditional organization must be preserved and interfaced with the newly introduced ideas coming from DevOps. E.g. feature teams may be introduced with a responsibility for the operations of their particular software components, but might still need an overall operations team for synchronization.

The roles involved in this workshop are development and test as well as the several management roles (program, requirements, data, environment, and release management), operations, and the product owner. The need for this many different participants in the workshop is induced by its broad impact in the whole organization.

2.7 Organizational structure in detail

After having defined the structure in the large, a second workshop refines this structure w.r.t. the distribution of skills and pods within different teams in the organization, the organizational interfaces between these teams as well as tasks and roles overarching several teams.

This workshop comprises the same participants as the workshop on the organizational structure in the large.

2.8 Cultural change and change process

In many cases, the term DevOps is connoted with a tool pipeline for continuous delivery. It is, however, only possible to leverage the full potential of DevOps by also introducing its culture. The customer must thus understand the Three Ways of DevOps [Ki16]:

1. The Principles of Flow
2. The Principles of Feedback
3. The Principles of Continual Learning and Experimentation

It does not make sense to introduce DevOps in a big bang or even to introduce the toolchain only and expect an optimal improvement and interleaving of the existing processes for development and IT operations. Instead, the Three Ways will have to be introduced step-by-step in a longer change process. It is the focus of this workshop to constitute this understanding.

The roles involved in this workshop are development and test as well as a number of management roles (program, environment, and release management), operations, and the product owner.

2.9 (Current) tooling in operations

In this workshop, we develop the requirements for tooling the environment management and IT operations starting from the current tool situation at the customer. We analyze the requirements and possible tool candidates for environment management with a focus on provisioning and virtualization, for data management, monitoring, and operations in general. We further investigate how to make the underlying processes transparent through dashboards.

In the aftermath of the workshop, the coarsely defined tooling and architecture is refined and piloted.

The roles involved in this workshop are the product owner and test as well as environment, data, infrastructure, and release management.

2.10 (Current) tooling in configuration management and requirements management

In this workshop, we develop the requirements for the configuration management of the different artefacts involved in the DevOps processes as well as for requirements management.

In the aftermath of the workshop, the coarsely defined tooling and architecture is refined and piloted.

The roles involved in this workshop are the product owner and test as well as environment, data, infrastructure, and release management.

3 Benefits of the Solution

Many organizations like insurances, where we have already conducted this workshop series, feel the need to become more agile in business – implementing DevOps seems like a great way to achieve this. The major change that is induced by this is often not well understood, however. The approach outlined above not only helps achieve transparency about what makes sense for an organization and how it can be implemented. More crucially, it is the first step towards an iterative road towards implementing this change, the only approach in which such ground-breaking changes can be delivered successfully.

Literaturverzeichnis

[Ki16] Kim, G.; Humble, J.; Debois, P.; Willis, J.: The DevOps Handbook. IT Revolution Press, Portland (OR), 2016.