Technical and organizational challenges when adopting DevOps principles for the integration of Artificial Intelligence into classical Software Engineering

Bachelor Thesis

for the degree of

Bachelor of Science

from the Course of Studies Applied Computer Science

at the Cooperative State University Baden-Württemberg Mannheim

by

Pascal Schroeder

Time of Project:	01/07/2019 - 23/09/2019
Student ID, Course:	5501463, TINF16AI-BC
Company:	IBM Deutschland GmbH, Ehningen
Supervisor in the Company:	Steffen Krause
Reviewer in Corporate State University:	Prof. Dr. Holger Hofmann

Declaration of Sincerity

Hereby I solemnly declare that my project report, titled *Technical and organizational challenges when adopting DevOps principles for the integration of Artificial Intelligence into classical Software Engineering* is independently authored and no sources other than those specified have been used.
I also declare that the submitted electronic version matches the printed version.

Mannheim, 23rd September, 2019

Pascal Schroeder

Contents

1 Introduction **1**

1.1 Changes in Software Development caused by AI 1

1.2 Cloud Computing 2

1.3 Development Operations in time of AI 4

2 Theory **6**

2.1 Development Operations principles 6

2.2 Kubernetes 6

2.3 Machine Learning 6

2.4 Artificial Intelligence lifecycle 6

2.5 Artificial Intelligence Operations 6

3 Method **7**

3.1 Catalogue of criteria 7

3.2 Used frameworks and libraries 7

3.2.1 Kubeflow 7

3.2.2 Tensorflow 7

3.2.3 Mask R-CNN 7

3.3 Creating the necessary environment 7

3.3.1 Minikube 7

3.3.2 Kubeflow 7

4 Result **8**

4.1 Creation of pipeline 8

4.2 Data preparation 8
4.3 Model Building 8
4.4 Deployment 8
4.5 Resulting model 8
4.6 Continuous learning 8

5 Discussion 9
5.1 Model evaluation 9
5.2 Pipeline evaluation 9
5.3 Outlook 9

6 Appendix 10

List of Listings

Abkürzungsverzeichnis

API Application Programming Interface

HTML Hyper Text Markup Language

IBM International Business Machines Corporation

JS JavaScript

JSON JavaScript Object Notation

UI User Interface

1 Introduction

1.1 Changes in Software Development caused by AI

First discussed in the 1940s, Artificial Intelligence has made great progress in recent years thanks to advances in computing capacity as well as Deep Neural networks and the accessibility of huge amounts of data. [1] This leads to companies investing in AI about 32€ Billion in 2019 according to a forecast by IDC, which makes it one of the most important fields of businesses. [2]

These progresses do not only impact the products itself, but also their development. In contrast to a traditional software development process in which the business logic is explicitly encoded in rules, solutions that rely on machine learning are fed with huge amounts of data that contains the business logic only implicitly.

This leads to a completely different development cycle. While classical software development is focused on the design and the development of the code, which is followed by testing and deployment, the machine learning lifecycle consists out of data collection, preparation, the training of the model with those data as well as the deployment of this model. [3]

This new type of software development is getting more and more important. Some people like Andrej Karpathy, Director of AI at Tesla, even predict, that these new type of Software will replace traditional software development as a whole. [4] This has some advantages, e.g. that it is easier to learn and to manage, more homogeneous and very well portable. In return it is harder to understand for humans so that those systems appear to us as “black boxes”, and debugging can be very difficult as well, because usually, there is no real error message. [3]

In this work, all the progresses that have lead to this transformation will be explained in more detail. Additionally, it will be described which difficulties and challenges come along with it, focused on necessary changes for operations of the development cycle, called DevOps.

1.2 Cloud Computing

In 2009 the university of Berkeley published a paper, in which the potential of Cloud Computing has been discussed. In doing so Cloud Computing has been defined as both - the applications delivered as services over the Internet, referred to as Software as a Service (SaaS), as well as the hardware and the systems software in the datacenters that provide those services. [5]

Thereby a distinction is made between a Public Cloud and a Private Cloud. A Cloud is called public when the Cloud is made available in a pay-as-you-go manner to the public. This means, that you pay for a service in advance and then you can only use what you have paid for. Private Cloud, on the other hand, refers to internal datacenters of a business or other organizations unavailable to the public. [5]

SaaS in general has advantages for both end-users as well as service providers. While the service provider can install, maintain and control their services more easily, the user can access the service anytime, anywhere, collaborate more easily and keep their data in the infrastructure. [5]

Cloud Computing enables this possibility to more application providers and additionally also the possibility to scale their applications on demand. Based on this possibility there are some more advantages identified in the described paper for Cloud Computing in particular:

- The illusion of infinite computing resources on demand
- The elimination of an up-front commitment by Cloud users
- The ability to pay for use of computing resources [5]

The first point eliminates the need to plan far ahead how much resources are needed, but enables the user to buy as much resources as necessary as soon as it is needed.

Also, Cloud users are allowed to change their need for resources any time, so they can scale up their application and don't have to commit to their need beforehand as described in point two.

The last point is about saving money when the resources are no longer needed. As it is possible to scale up, it is also possible to scale down, which eliminates the need to pay for the resources. This releases the Cloud users from taking too much risk when paying for resources they may not need for a long time.

For realizing those advantages different approaches were competing with each other - one looking similar to physical hardware with the ability to control the entire software stack similar to a low-level Virtual Machine (VM) and on the other hand a clean separation between a stateless computation tier and a stateful storage tier. In this competition, the first option has become established, because users mostly wanted to recreate their local environment instead of writing new programs solely for the cloud. [6]

But this solution still forced the User to manage their environment themselves. Especially for simpler applications, it is desirable to have an easier path to deploy their applications to the cloud.

Amazon provisioned a new solution for those needs in 2015 called AWS Lambda. Lambda offered the possibility to simply write the code to be executed and leaves all server provisioning and administration tasks to the cloud provider. [7]

This is known as serverless computing, because - even though there are still servers being used for the computation tasks - the user does not have to care about any tasks on serverside and can focus on writing the code. Serverless services must scale and bill automatically based on usage without any need for explicit provisioning. [6]

Those new possibilities with the Cloud technology changed the way of how developers can deploy and manage their products. Additionally, also AI development benefit from this very strongly because of several reasons.

The first one is automated scalability, which eliminates the need to worry about raising workloads. Additionally, when AI models are being trained there are much resources needed, but besides this task, there are not as much resources necessary. Through serverless computing, this idle time will not be billed. [8]

Serverless computing also reduces the interdependence, because the functions can be updated, deleted or invoked any time without having any side effect on the rest of the system. [8]

All this forced developers to change their way of developing and operating their products. This change of the DevOps lifecycle will be introduced in chapter 1.3.

1.3 Development Operations in time of AI

The traditional software development has continuously been improving for years, defining standards and principles to increase the efficiency, portability, and quality of the development cycle. Those principles and practices are called DevOps. In the new type of software development described in chapter 1.1, not all of those are applicable any more and new ones need to be defined. Additionally, the upcoming Cloud technology changed the way how products can be deployed and managed as described in chapter 1.2. In this chapter, the consequences of those changes for DevOps will be examined.

Starting with development itself and ending with maintaining and improving the product, there are DevOps principles for every single stage of the development cycle. This is described in more detail in chapter 2.1. In the new world of AI and Cloud, those stages are different, some new stages have been added, others have become unnecessary

Additionally, Cloud Computing changed the way of how products are deployed and managed. The Cloud can serve as a centralized platform for testing, deployment, and production. Also, it can automatically scale applications and allocate needed resources. Serverless computing even eliminates the need to set up a complete system and offers a simple way to deploy Machine Learning or other functions. [9]

AI tools on the other hand are enabling new possibilities for improving the IT operations functions like monitoring, analysis, service management, and automation.

All those progresses led to two big trends in the DevOps area. The first one is using Artificial Intelligence for DevOps. In doing so machine learning functionalities are used to enhance all

IT operations. Gartner calls this “AIOps” and predicts, that the use of AIOps will rise from 5% in 2018 to 30% in 2023. [10]

The second trend is using DevOps for the development of AI. This means adopting existing principles and practices for the traditional development cycle to the development cycle of AI tools. The objective of this is to upheld existing standards and maximize the efficiency of the development processes. [11]

In this work this second trend will be analyzed, the influences leading to it will be described and possible solutions will be shown, compared and discussed.

2 Theory

2.1 Development Operations principles

2.2 Kubernetes

2.3 Machine Learning

2.4 Artificial Intelligence lifecycle

2.5 Artificial Intelligence Operations

3 Method

3.1 Catalogue of criteria

3.2 Used frameworks and libraries

3.2.1 Kubeflow

3.2.2 Tensorflow

3.2.3 Mask R-CNN

3.3 Creating the necessary environment

3.3.1 Minikube

3.3.2 Kubeflow

4 Result

4.1 Creation of pipeline

4.2 Data preparation

4.3 Model Building

4.4 Deployment

4.5 Resulting model

4.6 Continuous learning

5 Discussion

5.1 Model evaluation

5.2 Pipeline evaluation

5.3 Outlook

6 Appendix