FOM Hochschule für Oekonomie & Management

Hochschulzentrum Düsseldorf

Scientific Essay

Berufsbegleitender Studiengang

5. Semester

im Studiengang "Wirtschaftsinformatik"

als Teil des Moduls

Software Engineering

über das Thema

Prinzipien und Chancen des DevOps-Konzepts

von

Luis Pflamminger

Betreuer :	Stiven Raso
Matrikelnummer :	538276
Abgabedatum :	24. Februar 2022

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

TPS	Toyota Production System
CAMS	Culture, Automation, Measurement and Sharing
CALMAS	Culture, Automation, Lean, Measurement, Added-Value and Sharing
CI/CD	Continuous Integration / Continuous Delivery
VUCA	Volatilität, Unsicherheit, Komplexität, Mehrdeutigkeit
KPI	Key Performance Indicator

1 Einleitung

1.1 Problemstellung

Das Geschäftliche Umfeld moderner Unternehmen ist wechselhaft und komplex. Kundenanforderungen sowie Marktbedingungen verändern sich stetig und der Hohe Grad an Vernetzung zwischen verschiedenen Anwendungen erschwert die Übersicht. Aufgrund dieser schweren Bedingungen, benötigen große Softwareunternehmen wie Microsoft oder Alphabet Möglichkeiten, schnell auf sich ändernde Bedingungen zu reagieren, Kundenanforderungen rasant umzusetzen und die Zusammenarbeit und Wissensverteilung effizient zu organisieren. In dieser Arbeit wird untersucht, ob das Konzept DevOps plausible Chancen bietet, um diesen hohen Anforderungen gerecht zu werden.

1.2 Zielsetzung

Ziel dieser Arbeit ist es, die Prinzipien des DevOps Konzepts im Software Engineering zu beschreiben, sowie seine Chancen für Unternehmen und Entwicklungsteams darzulegen. Zudem soll DevSecOps thematisch eingeordnet werden.

1.3 Aufbau der Arbeit

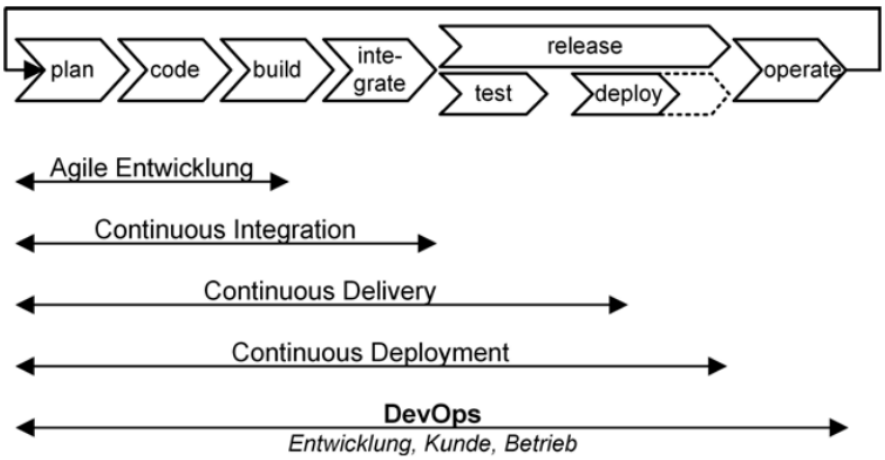
Die Arbeit beginnt mit einer Darstellung und Erklärung des Software Lebenszyklus und seiner Schritte. Anschließend wird das Wasserfallmodell als Beispiel für ein konventionelles Ablaufmodell erläutert. Agile Entwicklung wird abgegrenzt und anhand von Scrum erklärt. Im Hauptteil wird zunächst der Begriff DevOps definiert und Ursprung und Ziel des Konzepts erläutert. Anschließend werden die Grundsätze von DevOps anhand der sechs Säulen dargelegt. Nachdem die Prinzipien von DevOps klar sind, werden Chancen gegeben, die DevOps modernen Softwareunternehmen bietet. Daraufhin wird der Begriff DevSecOps im Kontext von DevOps eingeordnet. Abschließend werden die Erkenntnisse zusammengefasst und mögliche Kritik am Konzept adressiert.

2 Grundlagen

2.1 Der Software Life Cycle

Abbildung 1 zeigt den gesamten Lebenszyklus eines Softwareprodukts. Dieser reicht von der Planung inklusive Anforderungsfindung über die Programmierung, bis hin zum Testen und Betrieb der Software. Dieser Lebenszyklus ist für jede Software gleich, allerdings unterscheidet sich der zeitliche und organisatorische Ablauf der Phasen zwischen verschiedenen Vorgehensmodellen stark [1].

Abbildung 1: Übersicht des Softwarelebenszyklus



Quelle: [2]

Hier werden die einzelnen Phasen kurz erläutert:

Plan: In dieser Phase wird der Ist-Zustand analysiert und die funktionalen Anforderungen des Kunden an die Software gesammelt [2].

Code: Hier werden die spezifizierten Anforderungen im Code umgesetzt, was die Hauptaktivität des Entwicklungszyklus darstellt [2].

Build: Dies beinhaltet das kompilieren des Programms nach einer Änderung am Code. Das Ergebnis ist ein ausführbares Programm, welches getestet werden kann. In einigen interpretierten Programmiersprachen entfällt dieser Schritt [2].

Integrate: Hier wird die Software in andere Teile des Gesamtsystems integriert. Dieser Schritt ist etwas variabel, da viele Abhängigkeiten bereits während des Buildprozesses integriert werden, manche andere Systeme aber auch erst nach dem Testen der elementaren Funktionen der Software hinzukommen [2].

Test: Beim Testing wird die funktionalität der Software geprüft. Hier kommen zum einen Unit Tests zum Einsatz, die elementare Funktionen der Software überprüfen, zum anderen Integration Tests, die die Software im Zusammenhang mit anderen Systemen testen [2].

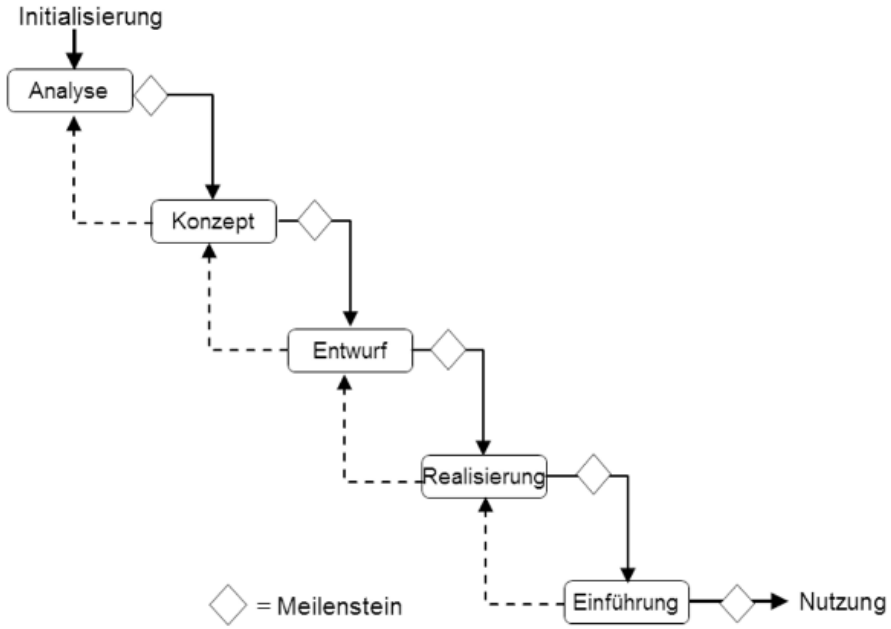
Deploy: In diesem Schritt wird die fertig getestete Software in die Produktionsumgebung ausgeliefert. Hier kann sie der Kunde nutzen [2].

Operate: Im Betrieb wird sichergestellt, dass die Software ordnungsgemäß läuft. Die Produktionsumgebung wird überwacht um mögliche Fehler zu finden und Nutzungsmetriken werden erhoben [2].

2.2 Das Wasserfallmodell

Ein gutes Beispiel für ein Modell der konventionellen Softwareentwicklung ist das Wasserfallmodell. Die einzelnen Phasen weichen hier leicht von den oben genannten ab, allerdings ist die Reihenfolge der Schritte in etwa gleich, sie werden lediglich anders gruppiert [1]. Das Modell wird in Abbildung 2 dargestellt. Im Kontrast zu agilen Methoden, die nachfolgend vorgestellt werden, werden in diesem Modell die einzelnen Phasen streng sequentiell durchlaufen. Am Ende jeder Phase werden zuvor festgelegte Meilensteine geprüft. Sind diese erreicht, schreitet das gesamte Projekt in die nächste Phase weiter. Sind die Anforderungen nicht erfüllt, wird die Phase mit leicht Veränderter Methodik wiederholt [1]. Ein Vorteile dieses Modells sind die Einfachheit und die klare Nachvollziehbarkeit des Projektstandes. Die Nachteile sind dagegen vielfältig. So werden Fehler in den frühen Phasen erst sehr spät entdeckt. Zusätzliche und neue Anforderungen können nicht mehr berücksichtigt werden und ein Kunde bekommt die Anwendung erst zu sehen, wenn das Projekt komplett ageschlossen ist. Diese Einschränkungen machen das Vorgehen sehr unflexibel [1].

Abbildung 2: Das Wasserfallmodell



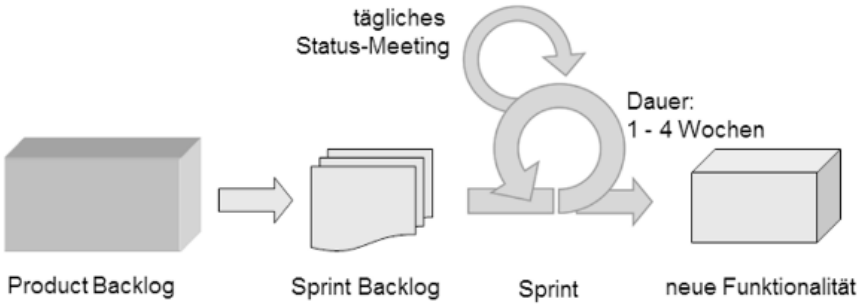
Quelle: [1]

2.3 Agile Vorgehensmethoden

Die agile Softwareentwicklung steht in starkem Kontrast zu konventionellen Methoden. Ziel ist es, den Entwicklungsprozess schlanker zu machen und unnötigen Aufwand zu reduzieren. Dem Kunden soll so früh wie Möglich eine funktionierende Software zur Verfügung stehen [1].

Als Beispiel für agile Entwicklung wird Scrum herangezogen. Hier werden vereinfacht ausgedrückt kleine Arbeitspakete definiert. Vor einem ein bis vier Wochen langem Sprint wird definiert, welche Arbeitspakete erledigt werden. Am Ende des Sprints sind diese bereit für die Auslieferung und es kann im Nächsten Sprint mit dem Umsetzen neuer Anforderungen begonnen werden. Dies ist ein iterativ-inkrementelles Verfahren, bei dem Anforderungen nach und nach, anstatt auf einmal umgesetzt werden [1]. Abbildung 3 zeigt eine grobe Darstellung des Scrum-Ablaufs.

Abbildung 3: Darstellung des Scrum Ablaufs



Quelle: [1]

Diese Vorgehensweise ermöglicht schnellere Auslieferung der Software. Dadurch kann schnell Feedback vom Kunden eingeholt und Anpassungen vorgenommen werden was die Qualität der Software verbessert und große Fehler vermeidet [1]. Es ist insgesamt ein wesentlich flexibleres System.

3 Prinzipien und Chancen von DevOps

3.1 Definition und Ziel von DevOps

Der Begriff DevOps setzt sich aus den Begriffen Development und Operations (engl. Betrieb) abgeleitet ist [3]. Development steht dabei für die Softwareentwicklung, während Operations für den IT-Betrieb steht [4].

DevOps ist eine Kombination von kulturellen und organisatorischen Denkweisen, Prozessen und Tools, die darauf ausgerichtet sind, den Zeitraum zwischen der Planung und Entwicklung einer Änderung an der Software, bis zur Bereitstellung dieser Änderung für Kunden auf dem Produktivsystem möglichst weit zu verringern [5]. Eine Änderung kann dabei ein neues Produkt, Feature oder das Beheben eines Fehlers sein. Durch diese erhöhte Geschwindigkeit können Unternehmen die Time-to-Market, also die Zeit, bis ein neues Feature oder Produkt den Markt erreicht, drastisch reduzieren, was es ihnen in erster Linie erlaubt, schnell auf Feedback der Kunden und Veränderungen auf dem Markt reagieren zu können, um einen Wettbewerbsvorteil zu erlangen. Auch Fehler können schneller angegangen werden, was eine kontinuierliche Verbesserung des Service zur Folge hat [2].

DevOps nahm seinen Ursprung bei einer Konferenz in 2009, bei der das Team von Flickr einen Prozess vorstellte, mit dem es in der Lage ist, neue Änderungen an der Software bis zu zehn mal am Tag in das Produktionssystem zu übernehmen. Inspiriert ist die Bewegung auch durch das Toyota Production System (TPS), welches durch T. Ohno bei Toyota entwickelt wurde [2]. Es ist ein Produktionssystem in der Autoindustrie, das einzig darauf ausgerichtet ist, den Moment zwischen dem Kundenauftrag und dem Geldeingang zu minimieren. Konzepte aus TPS wurden auf den IT-Wertschöpfungsprozess umgelegt, um von den gleichen Vorteilen profitieren zu können [2].

DevOps stellt keinen festgelegten Standard, wie z.B. SCRUM dar, sondern entwickelt sich kontinuierlich weiter. Es beruht auf bereits etablierten Konzepten wie Agilität, IT-Service Management, KanBan und Automatisierung. Dabei erfindet es diese Konzepte nicht neu, sondern bietet die Möglichkeit, sie zu nutzen, um eine Brücke zwischen den drei Seiten Entwicklung, Betrieb und Business zu schaffen [2]

3.2 Die Säulen von DevOps

Das Akronym Culture, Automation, Measurement and Sharing (CAMS), oder erweitert Culture, Automation, Lean, Measurement, Added-Value and Sharing (CALMAS), wird verwendet, um Einige Grundsätze von DevOps zusammenzufassen.

Culture (Kultur):

Die Kultur im Unternehmen und im Team ist ein wichtiger Bestandteil von DevOps. Grundlage der Kultur ist die intensive Zusammenarbeit miteinander. Die Gewinnung und das Verteilen von Informationen wird gefördert. Das Überbringen schlechter Nachrichten ist wichtig und wird befürwortet, um Fehler schnell beheben zu können. Dabei werden diese nicht verurteilt oder bestraft, sondern untersucht und korrigiert. Risiken werden geteilt und abteilungsübergreifendes Handeln wird gefördert. Ein wichtiger Teil der Kultur ist es zudem, ständig neue Ansätze auszuprobieren, um sich kontinuierlich weiterzuentwickeln [2].

Automation (Automatisierung):

Die gesamte Kette des Softwareentwicklungsprozesses sollte automatisiert abgebildet werden. Dies wird über sog. Continuous Integration / Continuous Delivery (CI/CD) Pipelines realisiert und umfasst unter anderem automatisierte Tests, Bereitstellung und Produktivnahme der Software. Durch Automatisierung können Liegezeiten und Fehler vermieden, sowie Entlastung bei den Mitarbeitern geschaffen werden [3][2].

Lean:

Lean-Methoden umfassen vor allem leichtgewichtige Entscheidungsprozesse, kontinuierliche Messungen und die Visualisierung von Arbeit. Auch die Limitierung von Work-in-Progress Items ist wichtig um Durchlaufzeiten zu verringern. Sie begrenzen die Anzahl an Aufgaben, die ein Mitarbeiter oder das gesamte Team gleichzeitig in Bearbeitung haben darf [2].

Measurement (Messung):

Um die Produktivität des DevOps Prozesses steigern zu können, sind Key Performance Indicators (KPIs) nötig, um die Leistung des Teams einstufen und messbare Ziele definieren zu können. Interessante Kennzahlen sind dabei z.B. Verfügbarkeiten, die Dauer vom Auftreten eines Fehlers bis zur Behebung oder die Zeit von einer Änderung im Code, bis diese in der Produktion zu sehen ist. Zudem sind Kennzahlen, die den geschäftlichen Nutzen der IT messen wichtig [2].

Added Value (Mehrwert):

DevOps darf nur eingesetzt werden, wenn es einen konkreten Mehrwert für das Unternehmen bringt. Ein gutes Beispiel für solch einen Mehrwert, ist die schnellere Auslieferung eines Features an einen Kunden oder das schnellere Beheben eines Fehlers [2]. Kann

kein messbarer Mehrwert festgestellt werden, wurde DevOps nicht erfolgreich implementiert.

Sharing (Teilen):

Eng mit dem Punkt "Cultureïst auch der Punkt SSharing"verbunden. Er beschreibt das Prinzip des Teilens von für die Zusammenarbeit notwendigem Wissen mit allen Beteiligten Mitarbeitern. Dieses Wissen kann die das Produkt selbst, aber auch Prozesse und Tools, die den DevOps Prozess erleichtern, umfassen. Wie bereits genannt ist das Teilen von Fehlern auch ein zentraler Bestandteil und muss ohne Schuldzuweisungen, sondern Lösungsorientiert, erfolgen [2].

3.3 Chancen von DevOps

Die heutige Geschäftsumgebung lässt sich gut durch das Akronym Volatilität, Unsicherheit, Komplexität, Mehrdeutigkeit (VUCA) darstellen, welches Zusammengefasst folgende Punkte beinhaltet. Die Märkte und Kundenanforderungen sind wechselhaft und verändern sich ständig. Dadurch entsteht eine gewisse Unsicherheit, die es erschwert, langfristige Projekte und Strategien zu planen [2]. Mit dem hohen Grad an Vernetzung zwischen verschiedenen Anwendungen und Prozessen geht eine hohe Komplexität einher. Der Zusammenhang zwischen Ursache und Wirkung ist oft unklar. Zudem sind meist verschiedene Handlungsalternativen als Antwort auf bestimmte Informationen möglich, da diese durch komplexe Zusammenhänge oft mehrdeutig interpretiert werden können [2].

DevOps bietet eine Antwort auf fast alle dieser Punkte und bietet somit die Chance, in einer VUCA-Welt geschäfts- und handlungsfähig zu bleiben.

Durch die hohe Geschwindigkeit ist DevOps in der Lage, die Time-to-Market erheblich zu senken. In einer VUCA-Welt hat dies mehrere Vorteile. Zum einen kann wesentlich schneller auf Veränderungen im Markt reagiert werden, da die Durchlaufzeit von Anpassungen am Produkt oder neuen Features geringer ist. Dies erhöht die Chance, dass Kundenanforderungen nicht bereits veraltet sind, wenn die Änderungen fertiggestellt werden [2]

Durch die rasche Bereitstellung verringert sich auch die Time-to-Value, das heißt die Zeit, bis der Kunde einen ersten Mehrwert aus einem Produkt ziehen kann. Dies wird durch das kontinierliche Deployment neuer Änderungen erreicht. So erhält der Kunde zu Anfang zwar kein fertiges Produkt, kann aber bereits Vorteile aus einem Prototypen ziehen [2]. Zudem verringert sich so die Zeit, bis Feedback eingeholt werden kann was Unsicherheit und Komplexität verringert und Planungssicherheit gibt.

Die verbesserte Zusammenarbeit innerhalb cross-funktionaler Teams mit guter Kultur kann helfen, die Komplexität und Ambiguität des Geschäftsumfelds zu verringern. Bessere Kommunikation und unterschiedliche Skillsets ermöglichen einen besseren Überblick über verschiedene Handlungsoptionen und erleichern die Entscheidungsfindung.

Abgesehen von VUCA bietet DevOps weitere Chancen. Die Zuverlässigkeit und Qualität kann durch automatisiertes Testing und Bereitstellung erhöht werden. Durch automatisierte Prozesse wird die Fehleranfälligkeit reduziert und die allgemeine Codequalität steigt [5].

Durch automatisierte Prozesse ist sichergestellt, dass auch Projekte mit großem Umfang zügig bereitgestellt werden können. Dies ist möglich, da Automatisierungen und die unterliegende virtuelle Infrastruktur sehr gut skalieren können [5].

Eine weitere Chance, die DevOps bietet, ist erhöhte Sicherheit. Diese wird vor allem im erweiterten Modell DevSecOps mit in Betracht gezogen, welches nachfolgend eingeordnet wird.

3.4 Einordnung von DevSecOps

DevSecOps erweitert den DevOps Gedanken um ein weiteres Themenfeld, die Sicherheit oder Security. In diesem Ansatz ist die Software Sicherheit ein integraler Bestandteil des IT-Lifecycles [6]. Traditionell ist IT-Security eine abgetrennte, der Softwareentwicklung nachgelagerte Funktion und Aufgabe eines seperaten Teams. Mit schnellen Entwicklungszyklen, wie sie DevOps bietet, ist das Risiko groß, dass die Sicherheit den Entwicklungs- und Bereitstellungsprozess ausbremst, da sie nicht schnell genug auf Veränderungen reagieren kann [6].

DevSecOps berücksichtigt die Anwendungssicherheit von Anfang an. Dies beinhaltet die Automatisierung von sicherheitsrelevanten Tests und Prozessen, um den DevOps Prozess nicht zu verlangsamen und Releasezyklen schnell zu halten. Neben zusätzlicher Automatisierung und Tools ist aber auch ein Kulturwechsel nötig. Sicherheitsteams müssen von Beginn an fest mit in den Prozess integriert werden. Entwickler müssen offen mit Sicherheitsexperten kommunizieren und Bedrohungen müssen offen Kommuniziert werden, ähnlich der Fehlerkultur im regulären DevOps [6].

4 Zusammenfassung und Fazit

Die Ergebnisse dieser Arbeit zeigen, dass DevOps verschiedene Methoden, Prozesse, Tools und Denkweisen in sich vereint, um die Geschwindigkeit des Softwareentwicklungsprozesses und der Bereitstellung von Software zu erhöhen. Cross-funktionale Teams, eine collaborative Kultur, ein hoher Automatisierungsgrad, KPIs zur Erfolgsmessung und Arbeitsvisualisierung sowie das Teilen neuer Erkenntnisse sind Kernbestandteile von DevOps. Diese Prinzipien bieten viele Chancen für Unternehmen, wie schnellere Reaktionen auf Kundenanforderungen oder Veränderungen am Markt, Reduktion von Komplexität und Unsicherheit oder bessere Übersicht und Entscheidungsfindung durch effizientere Zusammenarbeit. Zum Schluss wurde DevSecOps erläutert und eingeordnet.

Zusätzlich zu den Chancen, die in dieser Arbeit dargestellt wurden gibt es aber auch Kritik am DevOps Konzept. So gibt es Stimmen die behaupten, dass DevOps nicht funktioniere, da es in einer überwiegenden Anzahl an Unternehmen nicht die gewünschten Ergebnisse bringe [2]. Eine andere These ist, dass DevOps sich in vielen Bereichen nicht mit der Regulatorik vereinbaren lässt, z.B. im Bankensektor, in dem feste Rahmenbedingungen für die Durchführung und Dokumentation von Softwareprojekten existieren. Es wird zudem behauptet, dass DevOps nicht in der Lage ist, Silos aufzubrechen, sondern diese lediglich verschiebt [2].

Trotz dieser Kritikpunkte wird DevOps in den größten Softwareunternehmen eingesetzt und sorgt dafür, dass die Entwicklung mit der schnellebigen modernen Welt mithalten kann. Jede IT-Abteilung sollte Prüfen, ob sich durch den Einsatz von DevOps Prinzipien ein wirtschaftlicher und organisatorischer Vorteil erreichen lässt.

Literaturverzeichnis

[1] W. M. Dietmar Abts, *Grundkurs Wirtschaftsinformatik*. Springer-Verlag GmbH, 2017-03, 762 S., ISBN: 9783658163792. Adresse: https://www.ebook.de/de/product/33168867/dietmar_abts_wilhelm_muelder_grundkurs_wirtschaftsinformatik.html.

[2] J. Halstenberg, B. Pfitzinger und T. Jestädt, *DevOps*. Springer Fachmedien Wiesbaden, 2020. DOI: 10.1007/978-3-658-31405-7.

[3] R. Alt, G. Auth und C. Kögler, *Innovationsorientiertes IT-Management mit DevOps*. Springer Fachmedien Wiesbaden, 2017. DOI: 10.1007/978-3-658-18704-0.

Internetquellen

[4] D. Boss. „Was ist DevOps? Definition, Vorteile & Beispiel,“ Agile Heroes. (), Adresse: https://www.agile-heroes.de/magazine/devops/ (besucht am 2022-02-24).

[5] „Definition des DevOps-Modells,“ Amazon. (), Adresse: https://aws.amazon.com/de/devops/what-is-devops/ (besucht am 2022-02-24).

[6] „Was ist DevSecOps?“ Red Hat. (), Adresse: https://www.redhat.com/de/topics/devops/what-is-devsecops (besucht am 2022-02-24).

Ehrenwörtliche Erklärung

Hiermit versichere ich, dass die vorliegende Arbeit von mir selbstständig und ohne unerlaubte Hilfe angefertigt worden ist, insbesondere dass ich alle Stellen, die wörtlich oder annähernd wörtlich aus Veröffentlichungen entnommen sind, durch Zitate als solche gekennzeichnet habe. Ich versichere auch, dass die von mir eingereichte schriftliche Version mit der digitalen Version übereinstimmt. Weiterhin erkläre ich, dass die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde/Prüfungsstelle vorgelegen hat. Ich erkläre mich damit einverstanden, dass die Arbeit der Öffentlichkeit zugänglich gemacht wird. Ich erkläre mich damit einverstanden, dass die Digitalversion dieser Arbeit zwecks Plagiatsprüfung auf die Server externer Anbieter hochgeladen werden darf. Die Plagiatsprüfung stellt keine Zurverfügungstellung für die Öffentlichkeit dar.

Düsseldorf, 28.2.2022

(Ort, Datum)

(Eigenhändige Unterschrift)