

Duale Hochschule Baden-Württemberg
Mannheim

Bachelorthesis

Integration einer Container-Umgebung in einen automatisierten „Deployment“-Prozess und die Untersuchung ihrer Effekte auf diesen

Studiengang Wirtschaftsinformatik

Studienrichtung Software Engineering

Sperrvermerk

Verfasser/-in:	Yves Torsten Staudenmaier
Matrikelnummer:	7146590
Firma:	SV Informatik GmbH
Abteilung:	IE2 – Deployment
Kurs:	WWI17SEC
Studiengangsleiter:	Prof. Dr.-Ing. habil. Dennis Pfisterer
Wissenschaftlicher Betreuer:	Marius Ebel info@mariusebel.net +49 176 / 473 45452
Firmenbetreuer:	Thomas Teske thomas.teske@sv-informatik.de +49 621 / 454 44096
Bearbeitungszeitraum:	17.02.–08.05.2020

Sperrvermerk

Der Inhalt dieser Arbeit darf weder als Ganzes noch in Auszügen Personen außerhalb des Prüfungsprozesses und des Evaluationsverfahrens zugänglich gemacht werden, sofern keine anders lautende Genehmigung der Ausbildungsstätte vorliegt. Die Bachelorarbeit enthält unternehmensinterne Architektur- und Prozessmodellierung und deren Dokumentation. Es ist zum Zeitpunkt der Anmeldung nicht sicher, ob interne Schnittstellen in der Anwendungslandschaft offengelegt werden.

Mannheim, 05.05.2020

Nadja Haumbach, Ausbildungsverantwortliche

Lesehinweise

Die folgenden Hinweise sollen das Lesen dieser Projektarbeit erleichtern und spezielle Formatierung definieren:

- Im Sinne der Gleichberechtigung wird in dieser Arbeit entweder die weibliche Form oder die grammatisch korrekte Form „*die/der Entwickler/-in*“ verwendet werden. Die Kurzform (hier die weibliche Form) umschließt alle Geschlechter.
- Abbildungen, die mit dem Vermerk *unternehmensintern* gekennzeichnet sind, unterliegen folgendem rechtlichen Hinweis: „Alle Rechte, einschließlich der Vervielfältigung, Veröffentlichung, Bearbeitung und Übersetzung bleiben der SV Informatik GmbH vorbehalten.“
- Produkt- oder Eigennamen werden in KAPITÄLCHEN gesetzt, wie beispielsweise NODE.JS.
- Hochgestellte Ziffern weisen auf Fußnoten am Seitenende hin.
- Die Zitation von Software-Dokumentationen gestaltet sich ohne Angabe der Seitenzahl im Original als schwierig. Folgend wird der Stil zur Zitation solcher Dokumente beschrieben: „vgl. Red Hat, Inc. 2019, Application → Deployments“, dabei wird anstatt der Seitenzahl die Kapitel-Hierarchie mittels rechtsgerichteten Pfeil („→“) illustriert. So kann die interessierte Leserin anhand der Kapitel in der jeweiligen Dokumentation zur Quelle blättern.
- Quellcode, der innerhalb des Fließtextes steht, wird folgendermaßen formatiert:
`System.out.println("Hello World!")`

Kurzfassung

Titel: Integration einer Container-Umgebung in einen automatisierten „Deployment“-Prozess und die Untersuchung ihrer Effekte auf diesen

Verfasser/-in: Yves Torsten Staudenmaier

Kurs: WWI17SEC

Ausbildungsstätte: SV Informatik GmbH

Kurze Einleitung: Motivation, Einführung in den Themenkomplex, allgemeine Methodik

Forschungsfrage eins: Frage + kurze Erläuterung - Methodik - Teil-Ergebnis

Forschungsfrage zwei: Frage + kurze Erläuterung- Methodik - Teil-Ergebnis

Forschungsfrage drei: Frage + kurze Erläuterung - Methodik - Teil-Ergebnis

Minimalergebnis

Inhaltsverzeichnis

Abstract	III
Abbildungsverzeichnis	VI
Tabellenverzeichnis	VII
Quelltextverzeichnis	VIII
Algorithmenverzeichnis	IX
Abkürzungsverzeichnis	X
1 Einleitung	1
2 Methodologie: Beschreibung des Vorgehens	6
3 Wie können Container-Anwendungen den Prozess des automatisierten „Deployments“ unterstützen?	9
3.1 Grundlagen: Definition der Begrifflichkeiten zur Forschungsfrage eins	9
3.1.1 Methodik der Anforderungsanalyse	9
3.1.2 „Cloud-Computing (Cloud-C)“	12
3.1.3 Container, Containerisierung und Orchestrierung	15
3.2 Anforderungsanalyse des zu implementierenden Prozesses	19
3.3 Konzeption eines Container-basierten, automatisierten „Deployments“	22
3.3.1 Aufbau und „Deployment“ eines OPENSHIFT-Labors	22
3.3.2 Modellierung einer „Deployment“-Konfigurationsdatei	24
3.3.3 Prozessmodellierung anhand der Anwendung CAMUNDA	32
3.4 Ergebnis der Forschungsfrage eins	34
4 Welche wirtschaftlichen Vorteile hat der Einsatz von Containern auf den Prozess des automatisierten „Deployments“?	38
4.1 Grundlagen: Definition der Begrifflichkeiten zur Forschungsfrage zwei	38
4.2 Geschäftsprozess: „Release“ von neuen Anwendungsversionen	43
4.3 „Business Case“: „Deployment“ einer Container-Anwendung	46
4.4 Ergebnis der Forschungsfrage zwei	48
5 Welche besonderen sicherheitstechnischen Aspekte muss ein solcher Prozess im Bereich der Versicherung erfüllen?	50
5.1 Grundlagen: Sicherheitstechnische Anforderungen	50

5.2	Prozessbeschreibung: Beschaffung von „Open source“-Software	55
5.3	Konzept zur Implementierung der Sicherheitsanforderungen	57
5.4	Ergebnis der Forschungsfrage drei	57
6	Epilog	58
	Literaturverzeichnis	XII
	Anhang	XIX
A	Ergänzungen zur Forschungsfrage eins	XIX
A.1	Anforderungsdokument	XIX
A.2	Statistiken zum Themengebiet Cloud-C	XXIII
A.3	Ergänzungen zum Kapitel Container, Containerisierung und Orchestrierung	XXV
A.4	Anforderungskatalog der Forschungsfrage eins	XXVII
A.5	Quellcode zum Testen der Webseiten-Verfügbarkeit	XXVIII
A.6	Entwurf einer Konfigurationsdatei durch die Entwicklerinnen	XXIX
A.7	Aufbau der zulässigen Wortfolgen der Übergabedatei	XXXI
A.8	Grundgerüst der Konfigurationsdatei	XXXII
B	Ergänzungen zur Forschungsfrage zwei	XXXIV
B.1	Entscheidung über die Notwendigkeit eines „Business Case“	XXXIV
B.2	Vor- und Nachteile der internen beziehungsweise externen Erstellung eines „Business Case“	XXXVI
C	Ergänzungen zur Forschungsfrage drei	XXXVII
C.1	„Plan-Do-Check-Act“-Regelkreis	XXXVII
C.2	Checkliste zur Vorbereitung der Informationssicherheitsmanagementsystem (ISMS)-Einführung	XXXVIII
C.3	Schichtenmodell des IT-Grundschatzes	XXXIX
D	Ergänzungen zum Literaturverzeichnis	XLI
	Ehrenwörtliche Erklärung	LXI

Abbildungsverzeichnis

Abbildung 1.1 Dilbert-Comic zu KUBERNETES	1
Abbildung 3.1 Entwicklungsprozess der Anforderungen	10
Abbildung 3.2 Architektur der Virtualisierungsmodelle: VM vs. Container	16
Abbildung 3.3 (adaptiertes) Sequenzdiagramm zur Verteilung der Anwendung CAMUNDA	34
Abbildung 4.1 EPK zum Geschäftsvorfall „Release“	44
Abbildung 5.1 (adaptiertes) Sequenzdiagramm zur Beschaffung von „Open source“-Software	56
Abbildung A.1 Volere Snow Card	XX
Abbildung A.2 Ebenen der „Cloud“-Anforderungsanalyse	XXII
Abbildung A.3 Marktanteile der führenden Unternehmen am Umsatz im Bereich „Cloud-Computing“ weltweit von Juli 2018 bis Juni 2019 .	XXIII
Abbildung A.4 Umsatz mit „Cloud-Computing“ weltweit von 2009 bis 2018 und Prognose bis 2022	XXIV
Abbildung A.5 Architektur des Container-„Images“	XXV
Abbildung A.6 Überblick über eine KUBERNETES-Architektur	XXVI
Abbildung A.7 Architektur der „KUBERNETES (K8s)“-Interaktion	XXVI
Abbildung B.1 Notwendigkeit eines „Business Case“	XXXIV
Abbildung B.2 Chronologische Abfolge der Entwicklungsphase eines „Business Case“	XXXVI
Abbildung C.1 „Plan-Do-Check-Act“-Regelkreis	XXXVII
Abbildung C.2 Schichtenmodell des IT-Grundschutzes	XXXIX

Tabellenverzeichnis

Tabelle 3.1 Definition der „K8s“-Begrifflichkeiten	19
Tabelle 3.2 Überblick über die verwendeten Methoden in Abbildung 3.3 auf Seite 34	35
Tabelle 4.1 Vergleich der jährlichen Lizenzkosten	47
Tabelle A.1 Anforderungskatalog zum implementierenden Prozess	XXVII
Tabelle B.1 Überblick über die Vor- und Nachteile der externen Erstellung eines „Business Case“	XXXV
Tabelle B.2 Überblick über die Vor- und Nachteile der internen Erstellung eines „Business Case“	XXXV
Tabelle C.1 Checkliste zur Vorbereitung der ISMS-Einführung	XXXVIII

Quelltextverzeichnis

3.1 Installation des OPENSHIFT-Clusters	22
3.2 Test-„Deployment“ ins OPENSHIFT-Cluster	24
3.3 Beispiel einer „DeploymentConfig“-Datei	26
3.4 Beispiel einer „Deployment“-Datei	27
3.5 Ausschnitt aus Quellcode A.2: Deklaration der Umgebungsvaribalen	28
3.6 Umgebungsvariablen als Schlüssel-Wert (K-V)-Datei	29
A.1 Funktion zum Test der Verfügbarkeit eines Webseite	XXVIII
A.2 Entwurf einer „Deployment“-Datei für CAMUNDA	XXIX
A.3 Backus-Naur-Form (BNF) der Übergabedatei	XXXI
A.4 Grundgerüst der Konfigurationsdatei	XXXII
A.5 Beispielumsetzung der BNF der Übergabedatei	XXXIII

Algorithmenverzeichnis

1 Generierung der Konfigurationsdatei aus einer K-V-Datei	31
---	----

Abkürzungsverzeichnis

APT	„Advanced Packaging Tool“
AWL	Anwendungslandschaft
API	„application programming interface“
AWS	Amazon Web Services
BaFin	Bundesanstalt für Finanzdienstleistungsaufsicht
BNF	Backus-Naur-Form
BPMN	Business Process Model and Notation
BSI	Bundesamt(es) für Sicherheit in der Informationstechnik
BWL	Betriebswirtschaftslehre
CAB	„Change Advisory Board“
CLI	„Command-line Interface“
Cloud-C	Cloud-Computing
EPK	Ereignisgesteuerten Prozesskette(n)
IaaS	Infrastructure-as-a-Service
IE	IE – Entwicklungs- und Betriebsunterstützung
IE2	IE2 – Deployment
ISMS	Informationssicherheitsmanagementsystem
ITIL	Information Technology Infrastructure Library
IU11	IT-Einkauf und IT-Recht
K-V	Schlüssel-Wert
K8s	KUBERNETES
LXC	Linux Container
NGOs	Nichtregierungsorganisation
NIST	United States National Institute of Standards and Technology
OS	Betriebssystem
OSI	„Open Systems Interconnection“
PaaS	Platform-as-a-Service
PNW	Provinzial NordWest
PoC	„Proof of Concept“
PROD	Produktions-Umgebung
SaaS	Software-as-a-Service
SOA	Service-orientierte Architektur
SPOT	„single point of truth“
SV	SV SparkassenVersicherung
SVI	SV Informatik GmbH
TTM	„time to market“
UML	„Unified Modeling Language“

URL	„Uniform Resource Locator“
VKB	Versicherungskammer Bayern
VM	virtuelle Maschine

1 Einleitung

Motivation der Arbeit Die SV SparkassenVersicherung (SV) ist bestrebt ihre Versicherungsprozesse und den Kontakt mit den Kunden durch digitale Kanäle ständig zu verbessern. Die Digitalisierung ist ein wichtiger Bestandteil der SV-Strategie: So ist sie Mitbegründerin der „id-fabrik“¹ und Mitglied im „InsurLab Germany“.² Um die Anforderungen der SV-Kunden nach ständiger Verfügbarkeit und hoher Servicequalität zu erfüllen, muss die SV ihre Anwendungslandschaft (AWL) ständig an diese Anforderungen anpassen. Die Nachfrage der SV-Kunden und damit die daraus resultierenden Anforderungen an die Systeme, stellt die IT-Dienstleisterin (SV Informatik GmbH (SVI)) der SV vor neue Herausforderungen. Die Verteilungsprozesse von neuen Versionen der Anwendungen können derzeit nicht während des laufenden Betriebes durchgeführt werden. So müssen neue Funktionen warten, bis das Wochenende des „Release“ sie veröffentlicht, d. h. die „time to market“ (TTM) der neuen Produkte ist sehr hoch. Diese Verzögerung der Produktveröffentlichung passt nicht zu den digitalen Strategiezielen der SV. Als Lösungsansatz sind die Eigenschaften der „Cloud“-Technologie zu nennen. Jedoch muss aus IT-Sicht hier eine Anmerkung gemacht werden:

Solved all your problems. You're welcome.



Abbildung 1.1: Dilbert-Comic zu KUBERNETES

Quelle: *Dilbert on Kubernetes* 2017

Redaktionelle Anmerkung: Abbildung nur als komprimiertes Format verfügbar (Qualitätseinbuße)

Die, in Abbildung 1.1, dargestellte Aussage ist absichtlich überspitzt, um das Verständnis der Lösungsmöglichkeiten der „Cloud“-Technologie zu schärfen. Der Hinweis der IT stellt klar, dass das oben beschriebene Problem nur mit einer ganzheitlichen Lösung zu erreichen ist. Es führt nicht zum gewünschten Ergebnis, Anwendungen,

¹Ein Start-up, das federführend Innovationen im Bereich der S-Finanzgruppe erzeugt.

²vgl. SV SparkassenVersicherung 2019, S. 30.

die nicht für die „Cloud“ gedacht sind, mit Zwang in diese zu bewegen. Diese Arbeit beschreibt den Prozess zur Verteilung von Container-Anwendungen.

Problemstellung und -abgrenzung Bei der vorliegenden Arbeit handelt es sich um eine mehrteilige Analyse und Konzeption, die bei abgeschlossener Untersuchung ein Konzept zur automatischen Verteilung von Container-Anwendungen, die Betrachtung der Effekte von diesen Anwendungen und ein Ergebnis zu den sicherheitsrelevanten Aspekten dieses Prozess enthalten wird. Dabei akkumuliert sich der Fokus die Prozessentwicklung der Verteilung einer Container-Anwendung. Des Weiteren ist die Untersuchung der Anforderungen, die Analyse der „Cloud“-Plattform OPENSHIFT³ und die Erstellung eines Generierungsalgorithmus im Fokus dieser Arbeit. Dieser Teilkomplex wird durch die Forschungsfrage eins – Wie können Container-Anwendungen den Prozess des automatisierten „Deployments“⁴ unterstützen? – abgedeckt. Der Anforderungskatalog soll in Zusammenarbeit mit dem Fachbereich erstellt werden, um die Akzeptanz der Anforderungen zu steigern und um die Validität dieser zu gewährleisten. Die Forschungsfragen zwei und drei (siehe dazu Paragraph „Forschungsfragen/-design“) analysieren den wirtschaftlichen und sicherheits-/rechtlich-relevanten Bereich des Themenkomplexes der Bachelorarbeit. Dabei bedient sich die Forschungsfrage zwei der Methodik des Geschäftsvorfalls („Business Case“). Beide Forschungsfragen werden durch eine Analyse des Ist-Zustands eine initiale Konzeption ableiten.

Nicht Teil dieser Arbeit ist die unternehmensinterne Entscheidung über die Verantwortlichkeiten des zu entwickelnden Prozesses und die Erstellung einer unternehmensweiten Strategie zum Container-„Deployment“. Die Anpassung des unternehmensinternen Prozesses „Release“ ist nicht Teil dieser Arbeit. Dieser übersteigt die Anforderungen, die an diese Bachelorarbeit gestellt werden. Auch genügt die rechtliche Betrachtung des Einkaufsprozess von Software nicht den juristischen Ansprüchen eines Gutachtens. Dies ist jedoch nicht Ziel der Arbeit. Die wirtschaftliche Betrachtung des Geschäftsvorfalls „Container-Verteilung“ dient als reine Veranschaulichung der Methodik und genügt nicht einer vollständigen Betrachtung, die die Betriebswirtschaftslehre (BWL) vorgibt. Der Fokus der Arbeit liegt auf der technischen Entwicklung eines Verteilungsprozesses.

³ „OPENSHIFT is an open source container application platform by Red Hat based on the Kubernetes container orchestrator for enterprise app development and deployment.“ Quelle: Red Hat, Inc. 2020b

⁴ „Software deployment may be considered to be a process consisting of a number of inter-related activities including the release of software at the end of the development cycle; the configuration of the software, the installation of software into the execution environment, and the activation of the software. It also includes post installation activities including the monitoring, deactivation, updating, reconfiguration, adaptation, redeploying and undeploying of the software.“ (Dearle 2007)

Zielstellung der Arbeit Folgende *SMART*⁵ formulierte Ziele sollen diese Bachelorthesis leiten, messbar machen und später als Anhaltspunkt zur Evaluierung des Erfolgs dienen:

1. Entwicklung eines Verteilungsprozess für einfache Container-Anwendungen bis zum 27.April 2020. Einfach bedeutet hier, dass die Container-Struktur aus einer „Base Image“⁶-Schicht und einer Logik-Schnitt (Eigenentwicklung) besteht.
2. Der Prozess muss zu 98 % ohne Einwirkung von Menschen während des Verteilungsvorgangs funktionieren, d. h. er ist (annähernd) voll automatisiert. Dies muss bis zum Ende des Bearbeitungszeitraum der Arbeit (8.Mai 2020) umgesetzt werden.
3. Die Generierung einer Konfigurationsdatei soll in 9 von 10 Verteilungen automatisch mit einem Skript durchgeführt werden. Umzusetzen ist dies bis zum 08.Mai 2020.
4. Die Vorteile einer Container-Anwendung für die Verteilung dieser sollen erforscht werden. Akzeptiert ist dieses Ziel, sobald eine Auflistung und eine kritische Be- trachtung der Ergebnisse beschrieben wurde. Umzusetzen bis zum Ende des Be- arbeitungszeitraums. (Dieses Ziel ist nicht komplett *SMART*-konform, da die zumindest die Messbarkeit ohne genannte Messgröße nicht nachvollziehbar ist.)
5. Die wirtschaftliche Betrachtung muss sich anhand des Standes von Wissenschaft und Technik orientieren. Dazu werden gängige Regeln von Herman und Siegelaub 2009, Brugger 2009 u. Ä. benutzt. Diese Betrachtung muss bis zum Ende des Bearbeitungszeitraums durchgeführt werden. Die Messbarkeit wird durch die Einhaltung der oben genannten Regeln beschrieben.
6. Die sicherheits-/rechtlich-relevanten Aspekte dieses Projektes sollen anhand der, für die Finanzdienstleistungs-/Versicherungsbranche, geltenden Vorschriften beleuchtet werden. Der Umfang dieser Beleuchtung beschränkt sich auf das Nötigste, d. h. es müssen nur die wichtigsten Regeln beschrieben werden. Dies ist bis zum Ende des Bearbeitungszeitraumes umzusetzen. Die Messbarkeit ist durch die sicherheits-/rechtlich-relevanten Vorschriften bestimmt.

Forschungsfragen/-design Die Forschungsfragen, mit der sich diese Bachelorarbeit beschäftigen wird, sind eine direkte Konsequenz aus der Zielstellung der Arbeit und aus den unternehmensinternen Anforderungen an eines möglichst vollständig automatisierten Prozesses. Dabei liegt der Fokus auf der Betrachtung beider Teildisziplinen

⁵„SMART-Regel sind Formulierungshilfen, die eine einfache Methode zum Operationalisieren von Zielen und auch zur Überprüfung der Güte von Zielen darstellen.“ Quelle: Dechange 2020, S.69

⁶siehe dazu Kapitel 3.1.3 auf Seite 15

der Wirtschaftsinformatik, nämlich der Informatik und der Wirtschaft – jedoch wird der größere Teil dieser Arbeit einen informationstechnischen Fokus haben. Die folgende Aufzählung nennt die einzelnen Forschungsfragen, die im weiteren Verlauf ein gemeinsames Ergebnis erbringen werden. Dieses ist in Kapitel 6 auf Seite 58 dargestellt.

1. Wie können Container-Anwendungen den Prozess des automatisierten „Deployments“ unterstützen?
2. Welche wirtschaftlichen Vorteile hat der Einsatz von Containern auf den Prozess des automatisierten „Deployments“?
3. Welche besonderen sicherheitstechnischen Aspekte muss ein solcher Prozess im Bereich der Versicherung erfüllen?

Die Forschungsfrage eins wird einen Ist-Zustand analysieren. Diese Analyse enthält eine Betrachtung des Prozesses und einen Anforderungskatalog der Entwicklungsabteilungen an den zu konzeptionierenden „Deployment“-Prozess für Container-Anwendungen. Danach wird ein Konzept eines Container-basierten automatisierten „Deployment“-Prozesses erstellt, das aus der Entwicklung dieses und einer Standardisierung der beteiligten Dateien besteht. Die Forschungsfrage eins schließt mit einem Teilergebnis ab.

Die Forschungsfrage zwei beschäftigt sich mit den wirtschaftlichen Vorteil eines Einsatzes der Container auf den Prozess des automatisierten „Deployment“-Prozesses. Dabei wird der bestehende Geschäftsprozess „Release“ analysiert und ein Ist-Soll-Vergleich der Methodik des „Business case“ mit der Umsetzung im Unternehmen durchgeführt. Ein Ausblick schließt die Forschungsfrage zwei ab.

Die Forschungsfrage drei identifiziert sicherheitsrelevante Anforderungen, die nicht nur die Anwendung selbst betreffen, sondern Auswirkung auf die komplette Anwendungslandschaft (AWL) haben. Diese Anforderungen werden durch das Bundesamt(es) für Sicherheit in der Informationstechnik und verschiedene DIN/ISO-Normen beeinflusst. Außerdem soll analysiert werden, wie bei der Beschaffung von „Open source“- bzw. „Closed source“-Anwendungen mögliche Schwachstellen identifiziert werden, die potenzielle Angriffsvektoren in der AWL eröffnen würden, und wie mit diesen verfahren wird. Dabei soll versucht werden Rückschlüsse auf die Anwendung OPENSHIFT von RED HAT⁷ zu ziehen. Auch hier wird ein Teilergebnis diese Forschungsfrage abschließen

⁷ „Red Hat ist der weltweit führende Anbieter von Open-Source-Lösungen, die auf verlässlichen und leistungsstarken Technologien in den Bereichen ‚Cloud‘, Virtualisierung, Storage, Linux, Mobile und Middleware basieren. Darüber hinaus bieten wir Support-, Trainings- und Consulting-Services an, die mehrfach prämiert wurden.“ Quelle: Red Hat, Inc. 2020c

Einordnung der Abteilung in den Geschäftsprozess Die Abteilung IE2 – Deployment (IE2), die aus Sicht des unternehmensinternen Organigramms der Organisations-einheit IE – Entwicklungs- und Betriebsunterstützung (IE) angehört, befasst sich in erster Linie mit dem Transport („Deployment“) von Software-Artefakten der einzelnen Software-Produkte der SVI. Diese werden für die SV entwickelt, betrieben und gewar-tet. Zu den zentralen Aufgaben der Abteilung gehören die Planung, die Durchführung und die Überwachung der „Build/Deployment“-Prozesse der verschiedenen Service-Umgebungen, die aus mehreren Server-Verbünden bestehen. Des Weiteren stellt IE2 die Einspielung von datenbank-relevanten Objekten sicher. Auch entwickelt sie die Bau- und Transportprozesse kontinuierlich weiter und passt diese an die sich ständig verändernden Anforderungen der Entwicklungsabteilungen an. Von zentraler Bedeu-tung sind die Planung und die Durchführung der Veröffentlichungen der neuen Versio-nen einer zu betreuenden Anwendung. Zu dieser Aufgabe gehören auch Aufbau und Bereitstellung der Systemtest-, Releasetest- und Produktionsumgebungen. Eine weite-re zentrale Aufgabe ist das Umgebungsmanagement. Die Aufgaben dieses Teilbereichs be-fassen sich mit folgenden Inhalten: Planung von Aktivitäten in der Produktionsum-gebung, der Planung und der Koordination der Infrastruktur und Notfall-„Fix“ der Produktion und der allgemeinen „Patch“-Planung; Beratung zur Erweiterung, Koor-dination und Planung von verschiedenen Testumgebungen. Außerdem ist das Umge-bungsmanagement Teil des „Change Advisory Board“ (CAB), das ein Gremium nach der „Best practice“ Information Technology Infrastructure Library (ITIL) darstellt. Dieses ist für die Freigabe von „Changes“ verantwortlich und hat sowohl ständige, als auch der Situation angepasste Mitglieder.

Aufbau der Arbeit In Kapitel 3 auf Seite 9 wird die Forschungsfrage eins behan-delt. Diese beschreibt grundlegende Aspekte der Anforderungsanalyse, des Cloud-Computing und der Container(-isierung)/Orchestrierung. Diese Grundlagen werden später benutzt, um das Vorgehen in diesem Kapitel nachvollziehbar zu gestalten. Die Anforderungsanalyse erkundet die Wünsche der Fachbereiche im Bezug auf die Ver-teilung von Container-Anwendungen und entwickelt daraus Anforderungen, die dem Stand der Wissenschaft genügen. Wichtig in diesem Kapitel ist die Entwicklung des Prozesses und die damit verbundene Beschreibung der Tätigkeiten.

In Kapitel 4 auf Seite 38 wird die Forschungsfrage zwei behandelt:

In Kapitel 5 auf Seite 50 wird die Forschungsfrage drei behandelt:

In Kapitel 6 auf Seite 58

2 Methodologie: Beschreibung des Vorgehens

Forschungsfrage eins Die Anforderungsanalyse, die in dieser Forschungsfrage benutzt wurde, verwendet das Vorgehen von Hull, Jackson und Dick 2011 und Partsch 2010. Vor allem in Hull, Jackson und Dick 2011 wurde die methodische Vorgehensweise stark fokussiert, um eine hohe Qualität und Aussagekraft der Analyse zu gewährleisten. Deswegen nutzte diese Arbeit die dort illustrierte Methodik, da, erstens, es im Zusammenhang mit der Erhebung von Anforderungen eine häufig verwendete Methodik darstellt⁸ und, zweitens, das Vorgehen passend für die Problemstellung dieser Arbeit erschien. Die „stakeholder“ wurden anhand des vorhandenen „Deployment“-Prozesses abgeleitet, d. h. es war beim aktuellen Prozess Projektbeteiligte in Form von Rollen definiert. Diese wurden für jede Ausprägung des Prozess mit bestimmten „stakeholder“ gefüllt. Dadurch war der relevante Personenkreis aus den aktuellen Prozessen auf den zu implementierenden übertragen. Das „statement of needs“ war durch den Fachbereich „Entwicklung“ konkretisiert. Des Weiteren sollte ein Fragebogen die Wünsche der beteiligten Fachbereiche („stakeholder“) aufnehmen, um daraus die präzisen Anforderungen abzuleiten. Dabei ward absichtlich eine offene Fragestellung an den Entwicklungs-Fachbereich (namentlich: „SQUAD1“) gesendet, da es keine Beeinflussung durch die Fragestellung geben sollte. Ziel war es, die wirklichen Wünsche des Fachbereichs herauszufinden. Aus den Wünschen wurden nach dem Muster von vgl. Hull, Jackson und Dick 2011, S. 28 (siehe dazu auch Kapitel 3.1.1 auf Seite 9) spezifische funktionale und nicht-funktionale Anforderungen formuliert. Als Formulierungshilfe, um die Verständlichkeit und die Messbarkeit zu waren, dienten die Regeln nach Rupp 2020.

Die Modellierung einer „OPENSHIFT“-Labor-Umgebung diente dem Zweck der Anwendungsevaluierung, so wurde die Funktionsweise dieses System unter eingeschränkten Bedingungen getestet. Einschränkend wirkten die Limitierung auf eine „worker node“, die beschränkte Leistung der virtuellen Maschine, die Version von OPENSHIFT und die Entkopplung von der internen AWL. Hier wurde die „open source“-Variante genutzt, da so gewährleistet war, dass alle Tests und Verprobungen ohne Einschränkungen der Produktivumgebung durchgeführt wurden. Durch die Labor-Umgebung konnte die allgemeine Funktionsweise des Clusters getestet werden. Ziel dieser Umgebung war es, Tests zu ermöglichen und das Zusammenspiel der einzelnen Komponenten zu überprüfen. Aus diesem Grund wurde die Methodik der Labor-Umgebung gewählt, da sie einige Vorteile für diese Arbeit bot: Es konnte die Funktionsweise der Anwendungen

⁸Die Monographie wurde 1204 mal laut GOOGLE SCHOLAR – Google LLC 2020a – zitiert.

getestet werden, die Konfigurationsdateien für das Deployment wurden unter realistischen Bedingungen erforscht und die Interaktion mit anderer Hilfssoftware ist evaluiert worden. Die Erkenntnisse wurden, soweit möglich, auf die AWL übertragen. Problematisch war es, dass die kompletten Abhängigkeiten der einzelnen „AWL“-Bestandteile in der Labor-Umgebung nicht vollständig modelliert wurden.

Die Mustererkennung beziehungsweise Analyse der Konfigurationsdatei, die die Entwicklungsabteilung bei der Arbeitstagung zum „Proof of Concept“ (PoC) OPENSHIFT vorstellte, wurde zur Analyse als zentraler Bestandteil herangezogen. Dabei wurde der Vergleich als Methodik benutzt, um die Unterschiede und deren Vorteile hervorzuheben. Das Vorgehen orientiert sich am Anwendungshersteller, der sich für einen Vergleich zwischen der angebotenen und der benötigten Funktionalität zur Findung der Lösung aussprach.⁹ Ziel der Analyse war es, die benötigten Funktionen herauszufinden und damit die Begründung für die Ausprägung der Konfigurationsdatei zu liefern. Abschließend wurde eine Vorlage entwickelt, die alle Anforderungen der einzelnen Beteiligten abdeckte. Der Algorithmus zur Generierung dieser, war in Pseudocode zur besseren Lesbarkeit dokumentiert. Diese Dokumentation ist der Nachvollziehbarkeit und dem Verständnis der Leserin geschuldet.

Die Prozessmodellierung des Container-„Deployments“ nutzte die Erkenntnisse aus der Anforderungsanalyse und der Labor-Umgebung, um einen möglichst stark an die Anforderungen angepassten Prozess zu erzeugen. Hierbei wurde das Sequenzdiagramm¹⁰ der „Unified Modeling Language“ (UML) in einer adaptierten Form zur Visualisierung des Prozesse genutzt. Folgende Anpassungen wurden vorgenommen: die „Lifeline“ einer Instanz der Klasse stellte einen Prozessbeteiligten jeglicher Art dar. Sonst blieben die Semantik alle anderen Bestandteile unverändert. Das Sequenzdiagramm wurde gewählt, da es für die Beschreibung des Nachrichtenaustausches von verschiedenen Objekten benutzt wird. Dies war genau das darzustellende Ziel der Prozessmodellierung. Durch dieses Vorgehen wurde die Prozessmodellierung anwendungsunabhängig erstellt, um so dem Gedanken der Container¹¹ gerecht zu werden.

Forschungsfrage zwei Die Geschäftsprozessanalyse nutzte die Methodik Teile der Ereignisgesteuerten Prozesskette (EPK) nach Scheer, Nüttgens und Zimmermann 1997 und Teile des Business Process Model and Notation (BPMN) nach Object Management Group (OMG) 2011. Ziel der verwendeten Methodik war es, einen besseren Überblick über den Gesamtprozess „Release“ zu erhalten. Der Versuch die Optimierungspotentiale aufzudecken war durch die Verwendung der o. g. Methodik unterstützt. Es wur-

⁹vgl. Red Hat, Inc. 2019, Application → Deployments.

¹⁰Eine Art des Interaktionsdiagramms mit Fokus auf den Nachrichtenaustausch; spezifiziert in Object Management Group (OMG) 2017, S. 595-599.

¹¹Hiermit ist das Desinteresse der „Deployment“-Abteilung an dem Inhalt des Containers gemeint, d. h. es sollte dem „Deploy-Prozess“ egal sein, was der Container enthält.

de absichtlich nicht das komplette Vorgehen des „Business case“ nach Brugger 2009 durchgeführt, da die SVI sich nicht an dieses bei der Entscheidungsfindung hielt. So musste das Vorgehen zur „Business case“-Analyse adaptiert werden: Es entstand die Idee ein Ist-Soll-Vergleich durchzuführen. Dies entsprach nicht der wissenschaftlich anerkannten Methode der Kosten-Nutzen-Analyse, wie es der „Business case“ vorschrieb, jedoch die praktikabelste Möglichkeit. Es waren keine Daten für die Berechnung von Investitionskennzahlen verfügbar, da die Entscheidung nicht auf Grund dieser Analyse vom Fachbereich getroffen wurde. Die Entscheidung Container-Anwendungen zu nutzen, war bedingt durch den Kauf der Software CAMUNDA.

Forschungsfrage drei Forschungsfrage drei

3 Wie können Container-Anwendungen den Prozess des automatisierten „Deployments“ unterstützen?

Dieses Kapitel beschreibt die Entwicklung eines Verteilungsprozesses der Container-Anwendungen mit der, für die Abteilung IE2 zentralen, Software ARA.¹² Zuerst werden Grundlagen der Anforderungsanalyse, der „Cloud“-Technologie und der Containerisierung gelegt. Danach folgt die Dokumentation der Entwicklung des automatisierten Prozesses, sowie Definition einer Übergabe- und Konfigurationsdatei. Ziel dieses Kapitels ist es, die Forschungsfrage eins zu beleuchten und ein fundiertes Ergebnis zu erzeugen.

3.1 Grundlagen: Definition der Begrifflichkeiten zur Forschungsfrage eins

Dieses Teilkapitel soll grundlegende Begrifflichkeiten, die im weiteren Verlauf dieser Arbeit verwendet werden, definieren, um so eine einheitliche Terminologie der Begriffe zu entwickeln. Dadurch wird ein gemeinsames Verständnis erzeugt.

3.1.1 Methodik der Anforderungsanalyse

Die Anforderungsanalyse leitet sich aus dem thematischen Komplex des „Requirements-Engineering“ ab, die verschiedene Bedeutungsvarianten besitzt. Dabei „[...] steht es einmal für alle konkreten Aktivitäten am Beginn einer Systementwicklung, die auf eine Präzisierung der Problemstellung abzielen. Ebenso steht ‚Requirements-Engineering‘ aber auch für eine ganze Teildisziplin im Grenzbereich zwischen Systems-Engineering,

¹² „Automic Automation gives you the agility, speed and reliability required for effective Digital Business Automation. From a single unified platform, Automic centrally delivers the Workload Automation, Self-Service Automation, and Big Data Automation capabilities needed to drive growth of your company and accelerate your digital transformation.“ Quelle: Broadcom Inc. 2020

Informatik und Anwendungswissenschaften.“¹³ Diese Analyse soll, laut der herrschenden Meinung der Wissenschaft, am Anfang jeder Systementwicklung stehen, um so bestimmte Vorgehensweisen anzuwenden. Dabei entstehen, wenn der später weiter definierte Prozess verfolgt wird, viele systematisch verbundene Dokumente, die Anforderungen enthalten. So ist jede Anforderung wieder ein Cluster von kleineren Anforderungen, die miteinander verbunden sind. Diese werden durch den IEEE-Standard 1220 definiert als „a statement that identifies a product or process operational, functional, or design characteristic or constraint, which is unambiguous, testable or measurable, and necessary for product or process acceptability (by consumers or internal quality assurance guidelines).“¹⁴ Dieser Standard legt mit höchster Priorität den Fokus auf die Formulierung einer Anforderung als elementar wichtig für das Produkt bzw. für das Erreichen der Akzeptanz des Produktes. Ziel der Analyse ist es, funktionale und nicht-funktionale Anforderungen zu identifizieren und diese testbar zu dokumentieren. Funktionale Anforderungen definieren genau, was ein System später erfüllen muss: Sie ergeben sich aus der Fragestellung „Was tut das System? und was soll es aufgrund der Aufgabenstellung können?“¹⁵ Nicht-funktionale Anforderungen konkretisieren die Qualitätsansprüche an das System, die Forderung an das zu implementierende System als Ganzes, sowie Randbedingungen, die aus Projekt-, Prozess- und Unternehmensbedingungen resultieren können.¹⁶

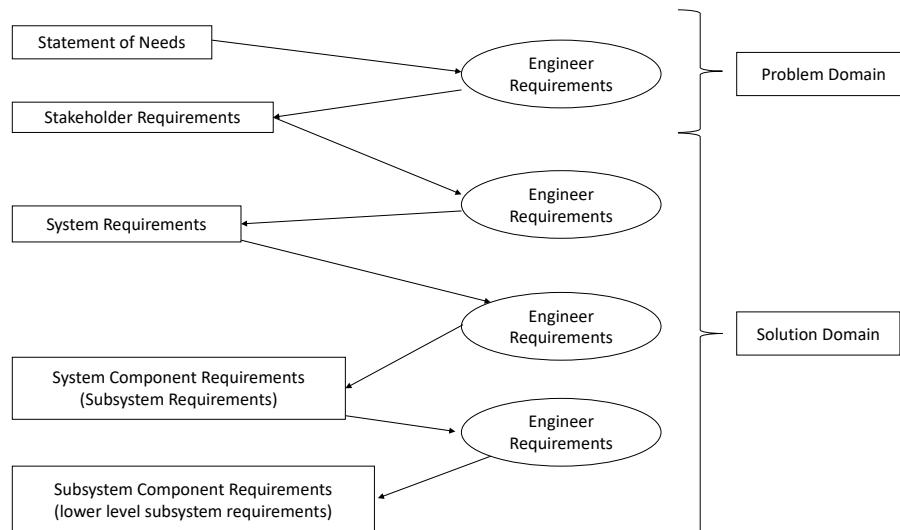


Abbildung 3.1: Entwicklungsprozess der Anforderungen

Quelle: in Anlehnung an Hull, Jackson und Dick 2011, S. 28

¹³Partsch 2010, S. 19.

¹⁴IEEE 2005, S. 9.

¹⁵Partsch 2010, S. 27.

¹⁶vgl. Partsch 2010, S. 27-29.

Das „statement of needs“ ist der Startpunkt für die Entwicklung einer Anforderung, die am Ende des Prozesses, der in Abbildung 3.1 auf der vorherigen Seite dargestellt ist, präzise dokumentiert sein wird. Dieses ist am Anfang immer Ausdruck eines Anspruchs oder Wunsches an das zu entwerfende System; dabei bilden das „statement“ und die „stakeholder requirements“ die „problem domain“. Diese definiert die grundständige Methodik, wie auch eine nicht-technische Herangehensweise, die auf die Projektbeteiligten („stakeholder“) angepasst sind. Nachfolgend werden die Projektbeteiligen als „stakeholder“ bezeichnet. Die Rolle ist beschrieben als „(Stakeholder) sind Personen oder Organisationen, die ein potenzielles Interesse an einem zukünftigen System haben und somit in der Regel auch Anforderungen an das System stellen.“¹⁷ Später definiert die „problem domain“ den Zweck des Systems: Dadurch ist bei der Ermittlung der Anforderungen die Frage „Was ist der Zweck des Systems?“ anstelle „Was soll das System Ihrer Meinung nach tun?“. Dies soll die „stakeholder“ extrinsisch motivieren, über den Zweck des zu entwerfenden Systems und nicht über einen möglichen Lösungsweg (das Wie) nachzudenken. Durch diesen Ansatz folgen Antworten nach dem Muster „Ich möchte etwas tun können ...“. Wissenschaftlich bzw. literarisch betrachtet sind diese Form der Anforderungen als „capability requirement(s)“¹⁸ bekannt. Sie stellen die wichtigsten Erkenntnisse in der „problem domain“ dar. Nun wird im weiteren Verlauf ein Modell konstruiert, das den Projektbeteiligten, den „stakeholder“, präsentiert wird. Dies unterliegt der Einschränkung, dass es jede/jedem Projektbeteiligte/-n versteht. Denn sie validieren das konstruierte Modell in jedem weiteren Schritt, der in Abbildung 3.1 auf der vorherigen Seite ersichtlich ist. Die Anforderungen an das Modell sind quantitativ gering: Es muss nicht-technisch sein und es muss geeignet sein, die Anforderungen an das System abzubilden. Eine solche Darstellung ist dann geeignet, wenn sie den gewünschten Zweck an das System abbildet, das heißt, dass sie keine technischen Details zeigt, sondern einen Überblick bietet. Ein „use scenario“¹⁹ wird meist verwendet, da es sich eignet, menschliche Aktionen bzw. Ziele darzustellen. Abschließend müssen die „stakeholder“-Anforderungen folgende Kriterien erfüllen:

- Kurz und prägnant formulierte Beschreibung, jedoch einfach zu verstehen und
- gleichzeitig sollten sie nicht-technisch, aber realistisch formuliert sein.

Die „solutions domain“, die auf Abbildung 3.1 auf der vorherigen Seite zu sehen ist, ist die Nachfolgerin der „problem domain“. Der Hauptunterschied zwischen den beiden Bereichen ist, dass die „solution domain“ idealtypisch qualitativ hochwertig beschriebene Anforderungen als „Input“ bekommt. Dazu konträr erhält die „problem domain“ eine vage formulierte Wunschliste oder ein nicht klar definiertes Ziel als initialen „Input“. Ausgehend von der Aussage von E. Hull, „in an ideal world, all the requirements

¹⁷Partsch 2010, S. 8.

¹⁸vgl. Hull, Jackson und Dick 2011, S. 94.

¹⁹vgl. Hull, Jackson und Dick 2011, S. 94.

would be clearly articulated, individual testable requirements“²⁰ ist zu deduzieren, dass es viele Ebenen zu erforschen gibt, um dieser Aufforderung zu entsprechen. So muss iterativ in jeder Ebene eine neue Analyse des „Inputs“ erfolgen, um einen Ausgangspunkt für das weitere Vorgehen zu initialisieren. Die Komplexität dieser Ebenen ist abhängig von dem Grad der Innovation sowie vom Kontext des zu entwickelnden Systems. Jede Entscheidung während des Prozesses kann mögliche Entscheidungspfade in einer anderen Ebene verhindern. Ziel des Prozesses ist es, ein Anforderungsdokument bzw. -katalog zu entwerfen, das laut der gesichteten Literatur in verschiedenen Repräsentationen vorliegen kann. Dennoch sollten primäre Bestandteile, wie die Rahmenbedingungen, die Projektbeteiligten, die Projektaspekte und die funktionalen und nicht-funktionalen Anforderungen enthalten sein. Ein Beispiel dieses Katalogs ist im Anhang A.1 auf Seite XIX zur Ansicht enthalten. Außerdem gibt es im Bereich der „Cloud“ besondere architektonische Anforderungen. Diese sind im Anhang als Abbildung A.2 auf Seite XXII einzusehen.

3.1.2 „Cloud-Computing (Cloud-C)“

Cloud-C, definiert als: „Paradigma, einen netzwerkbasierten Zugang auf ein skalierbares und elastisches Reservoir gemeinsam nutzbarer physikalischer oder virtueller Ressourcen nach dem Selbstbedienungsprinzip und bedarfsgerechter Administration zu ermöglichen“²¹ ist ein neuartiger und disruptiver Ansatz in der Informationstechnologie, der seit mehreren Jahren Führungskräfte und IT-Abteilungen beschäftigt. Dieser Ansatz verspricht die Lösung für sämtliche Herausforderungen der Kapazitäts- und Leistungsengpässe moderner IT-Infrastruktur zu sein.²² Auch diskutiert die Bevölkerung stark und meist auch sehr kontrovers über dieses Thema. Themen wie Datenschutz, Privatsphäre und das Risiko eines Datendiebstahls sind auch nach 20 Jahren Diskussion immer noch allgegenwärtig. Ein Grund dafür ist die hohe Dynamik dieser Technologie, sowie die ständige Weiterentwicklung, die von großen Unternehmen wie MICROSOFT, GOOGLE, AMAZON und IBM vorangetrieben werden. Momentan haben MICROSOFT und AMAZON die meisten Marktanteile am Umsatz im Bereich des Cloud-C.²³ Des Weiteren prognostiziert Gartner 2019 einen exponentiell wachsenden weltweiten Umsatz bis 2022 auf ungefähr 354,6 Milliarden US-Dollar. Damit würde dieser in den nächsten zwei Jahren um circa 100 Milliarden US-Dollar steigen. Für eine ausführliche Umsatzprognose ist auf die Abbildung A.4 auf Seite XXIV zu verweisen. Diese verdeutlicht auch, dass in den folgenden Jahren nach 2022 weiterhin mit einer exponentiellen Umsatzsteigerung zu rechnen ist, wenn das mathematische Modell der exponentiellen Regression weiterhin Bestand hat.

²⁰Hull, Jackson und Dick 2011, S. 115.

²¹DIN Deutsches Institut für Normung 2020a, S. 7.

²²vgl. Reinheimer und Springer Fachmedien Wiesbaden GmbH 2018, S. 4.

²³siehe dazu Abbildung A.3 auf Seite XXIII

Historisch betrachtet leitet sich Cloud-C von verschiedenen Konzepten anderer „Computing“-Bereiche und Architekturmuster ab: So spielte zur Entwicklung des heutigen Verständnis „Utility Computing“, „Service Orientation“ und „Grid Computing“ eine große Rolle.²⁴ John McCarthy hat in den 1960er-Jahren das erste Konzept im Bereich des „Utility Computing“ entwickelt.²⁵ Später wurde es durch Douglas Parkhill verfeinert und durch die folgenden Schlüsselkomponenten beschrieben: „Parkhill examined the nature of utilities such as water, natural gas and electricity in the way they are provided to create an understanding of the characteristics that computing would require if it was truly a utility. When we consider electricity supply, for example, in the developed world, we tend to take it for granted that the actual electrical power will be available in our dwellings. To access it, we plug our devices into wall sockets and draw the power we need. Every so often we are billed by the electricity supply company, and we pay for what we have used“.²⁶ Dieses Konzept leitete er auch auf eine technologische Ressource im Bereich des Computers ab.²⁷ Der Gedanke der Serviceorientierung beschreibt eine klare Begrenzung einer Funktion, die zur Erfüllung eines bestimmten Ziels verwendet wird. Services werden meist durch die Konzepte der Objektorientierung und der Abstraktion in einer Organisation definiert. Aus dem Grundgedanken und den genannten Konzepten entwickelt sich die Service-orientierte Architektur (SOA), die diese Prinzipien in ein technologiebasiertes Modell abbildet. Die Leitgedanken der SOA spielen auch im Cloud-C eine wichtige Rolle, denn, wie später noch näher definiert wird, ist der Servicegedanke ein elementarer Bestandteil der „Cloud“, der deutlich das Geschäftsmodell prägt. „Grid Computing“ ist ein Konzept aus den 1990er-Jahren und fand seine Anwendung im Bereich der elektrischen Netze.²⁸ Ziel dieses Konzeptes war es, die Einfachheit und Zuverlässigkeit der Stromnetze zu gewährleisten, über einen standardisierten Adapter Zugriff auf dieses zu erhalten, ohne sich um die technische Realisierung kümmern zu müssen. Dabei stellten die Pioniere dieses Konzeptes folgende Eigenschaften²⁹ an das System:

- Dezentrale Ressourcenkontrolle, d. h., ein Grid besteht aus geografisch verteilten Ressourcen, die administrativ unabhängig von Organisationen betreut werden.
- Standardisierte, offene Protokolle und Schnittstellen, d. h.; die Grid-Middleware ist nicht anwendungsspezifisch und kann zu verschiedenen Zwecken eingesetzt werden.
- Nichttriviale Eigenschaften des Dienstes, z. B. in Bezug auf Antwortzeitverhalten, Verfügbarkeit oder Durchsatz.

²⁴vgl. Hill 2013, S. 3-5.

²⁵vgl. McCarthy 1983.

²⁶vgl. Parkhill 1966.

²⁷vgl. Hill 2013, S. 4.

²⁸vgl. Weinhardt et al. 2009.

²⁹vgl. Foster und Kesselman 1999.

Diese Prinzipien haben eine Ähnlichkeit zu denen des Cloud-C, jedoch sind die wirtschaftlichen Aspekte durch die Gedanken des „Grid Computing“ beschrieben. Des Weiteren werden die Aspekte des „Grid Computings“ im Bereich des dezentralen Managements und der verteilten Ressourcen beim Cloud-C nicht weiterverfolgt. Vielmehr bietet die Zentralisierung die ökonomischen Vorteile, die eine zentrale Rolle des Geschäftsmodells darstellen.

Da es mehrere Definitionen von Cloud-C gibt, beschränkt sich diese Arbeit auf folgende: „Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.“³⁰ Das United States National Institute of Standards and Technology (NIST) beschreibt in der Publikation Mell und Grance 2011 folgende essenzielle Charakteristika³¹:

- on-demand self-service
- broad network access
- resource pooling
- rapid elasticity
- measured service

Des Weiteren beschreibt die NIST drei Servicemodelle, wie sich Unternehmen die „Cloud“ zunutze machen können: Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) und Infrastructure-as-a-Service (IaaS). Dabei wird SaaS definiert als: „The capability provided to the consumer is to use the provider’s applications running on a cloud infrastructure. [...] The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.“³² „Cloud“-Infrastruktur ist eine Sammlung von Hard- und Software des „Cloud“-Anbieters, die die fünf essenziellen Charakteristika des Cloud-C unterstützt bzw. erfüllt. Beispiele hierfür sind GOOGLE DOCS und OFFICE 365. PaaS wird beschrieben durch: „The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the

³⁰Mell und Grance 2011, S. 2.

³¹Jedoch werden diese Charakteristika in anderen wissenschaftlichen Ausarbeitungen um „multitenancy“, „service oriented“ und „utility-based pricing“ ergänzt.(vgl. Institute of Electrical and Electronics Engineers 2011, S. 1)

³²Mell und Grance 2011, S. 2.

provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.³³ Bei der später in der Konzeptionierung verwendeten Software, OPENSHIFT, handelt es sich um eine PaaS-Lösung. Weitere Beispiele sind GOOGLE APP ENGINE, WINDOWS AZURE und HEROKU.³⁴ IaaS wird durch folgende Definition abgebildet: „The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g. host firewalls).“³⁵ Hierzu zählen die Produkte AMAZON EC2, OPENSTACK und VMWARE. Nun sind die Bereitstellungsmodelle der „Cloud“ noch von Bedeutung: Die NIST sowie weitere, schon für diesen Abschnitt verwendete, Literatur definieren vier Modelle: „private, community, public and hybrid cloud“. Die „private cloud“ ist in exklusiver Nutzung eines Unternehmens, das mehrere interne Konsumenten bedient. Es kann entscheiden, ob alle Management- und Betriebsoperationen intern oder extern von einem Anbieter durchgeführt werden. Die „Cloud“ kann intern oder extern gehostet sein. Die „community cloud“ ist eine „private cloud“, jedoch unterscheiden sich die beiden durch die Benutzergruppen. Bei der „community“-Variante ist es nicht auf die Organisation, sondern auf Gruppen mit gleichen Angelegenheiten beschränkt. Die „public cloud“ ist offen für die Öffentlichkeit – natürlich beschränkt durch die Regeln des „Cloud“-Anbieters. Die hybride Variante wird folgendermaßen beschrieben: „The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).“³⁶

3.1.3 Container, Containerisierung und Orchestrierung

„Historically, virtualization technologies have developed out of the need for scheduling processes as manageable container units. The processes and resources in question are the file system, memory, network, and system information.“³⁷ Aus dieser Notwendigkeit heraus entstanden verschiedene Lösungsansätze: die Virtualisierung in einer virtuellen Maschine (virtuelle Maschine (VM)) und etwas später die Container-Lösungen.

³³Mell und Grance 2011, S. 2.

³⁴vgl. Kumar und Vidhyalakshmi 2018, S. 8.

³⁵Mell und Grance 2011, S. 3.

³⁶Mell und Grance 2011, S. 3.

³⁷Pahl 2015, S. 25.

Virtuelle Maschinen konnten einige Herausforderungen, wie „scheduling, packaging and resource access“, durch ihre technologischen Ansätze lösen. Dabei wurde der architektonische Ansatz des sogenannten „Gast Systems“ entwickelt, d. h., die virtuelle Maschine ist ein vollwertiges Betriebssystem (OS) mit komplettem Dateisystem und eigenem Prozess auf dem „Host System“.³⁸ Im Vergleich dazu können Container die gleichen Anforderungen abbilden, jedoch unterscheidet sich deren Architektur (vgl. Abbildung 3.2). Ein Container enthält alle notwendigen, für die App relevanten, Bibliotheken beziehungsweise Abhängigkeiten und kann so, ohne ein komplettes OS, lauffähige Applikationen beinhalten. Diese Abstraktion ist im „Cloud“-Umfeld (bspw. in einer PaaS-Ausprägung) nützlich, da die Container leichtgewichtiger sind und dadurch weniger Speicherauslastung (persistenter Speicher) benötigen. Auch später für die Orchestrierung der Container in einem Cluster-Umfeld ist dies von Nutzen.

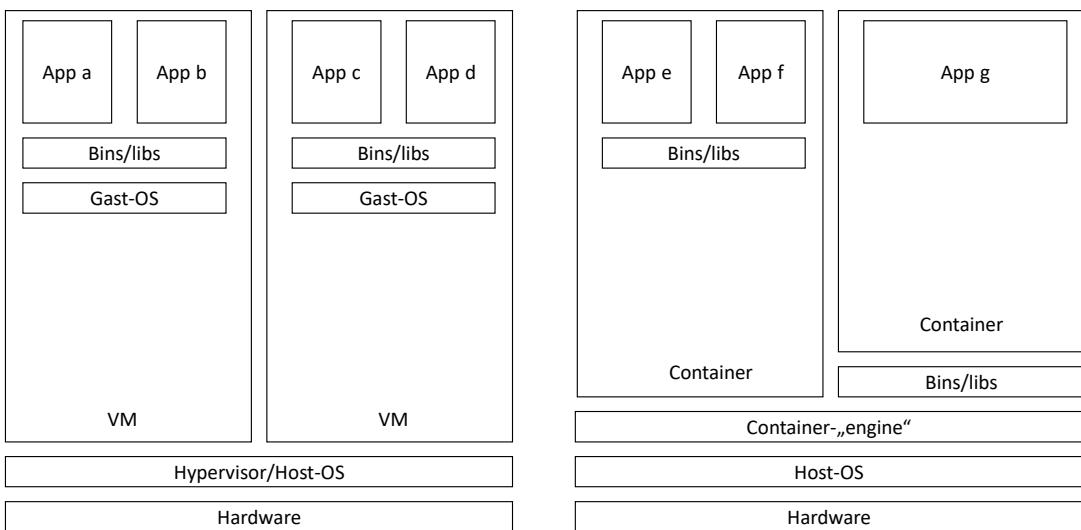


Abbildung 3.2: Architektur der Virtualisierungsmodelle: VM vs. Container

Quelle: in Anlehnung an Pahl 2015

Die gesichtete Literatur (Pahl 2015, Bernstein 2014, Kharb 2016; Combe, Martin und Di Pietro 2016 und weitere, siehe Literaturverzeichnis) definieren Container immer anhand ihrer charakteristischen Merkmale und im Vergleich zur VM. Google Ireland Limited 2020 folgt auch diesem Muster, allerdings eher auf Makroebene: „Container bieten einen logischen Mechanismus der Paketerstellung, der darauf beruht, dass Anwendungen von ihrer Ausführungsumgebung abstrahiert werden. Mit dieser Entkopplung können containerbasierte Anwendungen einfach und konsistent bereitgestellt werden,

³⁸Bietet Services für die Gastsysteme an.

unabhängig davon, ob es sich bei der Zielumgebung um ein privates Rechenzentrum, die öffentliche ‚Cloud‘ oder auch um den persönlichen Laptop eines Entwicklers handelt.“³⁹ Ziel der Containerisierung ist es, Entwicklerinnen die Möglichkeit zu bieten, sich nur auf die Anwendungslogik und -abhängigkeiten zu konzentrieren. Gleichzeitig können andere IT-Teams, wie IE2, sich um die Bereitstellung und Verwaltung dieser Container kümmern. Diese Teams können den Container als geschlossene Verpackung sehen, bei der sie keine Kenntnis über das Innenleben (die Anwendungsdetails) für ihre Arbeit benötigen.⁴⁰ Dies ist ein Bestandteil der Grundlagen für schnellere und qualitativ hochwertigere Deployments.⁴¹

Initial entwickelte Canonical Ltd. 2020 die Linux Container (LXC); DOCKER INC. ist ein „Open source“-Projekt, das sich die LXC-Technologie zunutze macht und eine Container-„Engine“ gebaut hat, um diese Technologie benutzerfreundlicher zu gestalten: „Basically, DOCKER extends LXC with a kernel- and application-level API that together run processes in isolation: CPU, memory, I/O, network, and so on. DOCKER also uses namespaces to completely isolate an application’s view of the underlying operating environment, including process trees, network, user IDs, and file systems.“⁴² DOCKER-Container nutzen eine „Image“-Struktur. So lassen sich durch Kombination verschiedener „Images“ Applikationen abbilden, die durch Programmierlogik ergänzt werden (vgl. Abbildung A.5 auf Seite XXV). In der Industrie ist DOCKER als De-facto-Standard⁴³ angesehen.⁴⁴ Außerdem bietet DOCKER viele Vorteile, wie die Leichtgewichtigkeit, „Open source“ und Sicherheit. Des Weiteren bietet DOCKER die Möglichkeit der Kollaboration zwischen verschiedenen IT-Teams. Weitere Vorteile sind, dass die Applikation überall (wo DOCKER installiert ist) ausgeführt werden kann und, dass DOCKER sich an die Unternehmensanforderungen ständig neu anpasst.⁴⁵ Durch diese Vorteile entstehen unmittelbare Konsequenzen, die Auswirkungen auf das Arbeiten haben: So werden das Einarbeiten eines neuen Mitarbeitenden beschleunigt, die Kreativität der Entwicklerinnen verstärkt, die Entwicklungsumgebung vereinheitlicht,⁴⁶ die Zusammenarbeit zwischen verschiedenen Teams vereinfacht und eine schnelle „TTM“⁴⁷ erreicht.⁴⁸

Um die Stärken und Vorteile der DOCKER-Container zu nutzen, brauchen diese eine Netzwerkanbindung. Nur mit dieser können Container in der Produktion eingesetzt

³⁹Google Ireland Limited 2020.

⁴⁰vgl. Google Ireland Limited 2020.

⁴¹vgl. Kharb 2016, S. 1.

⁴²Bernstein 2014, S. 82.

⁴³vgl. Pahl 2015, S. 30.

⁴⁴vgl. Kharb 2016, S. 1.

⁴⁵vgl. Kharb 2016, S. 1.

⁴⁶Dies wirkt sich direkt auf die Code-/Produktqualität aus

⁴⁷„TTM is the strategy of focusing on reducing the time to introduce new products to market.“
(Pawar, Menon und Riedel 1994, S. 14)

⁴⁸vgl. Kharb 2016, S. 2.

und eine Orchestrierung möglich gemacht werden. Dieser Begriff ist aus der Musik abgeleitet: Flexibles Kombinieren mehrerer Services oder Dienste zu einer sinnvollen Konzeption (Komposition), die einen Geschäftsprozess beschreibt. Für die Orchestrierung von Container-Anwendungen wird eine weitere Technologie benötigt, die von Google LLC 2020b entwickelt und als „Open source“ veröffentlicht wurde: KUBERNETES (K8s).⁴⁹ Die Semantik des Wortes KUBERNETES bedeutet auf Griechisch „Lotse, Steuermann“. Diese Metapher beschreibt die Hauptaufgaben von K8s zu treffend; es „verdeckt die Hardwareinfrastruktur und stellt ihr gesamtes Rechenzentrum als eine einzige, enorme Rechenressource dar. Dadurch können sie ihre Softwarekomponenten bereitstellen und ausführen, ohne sich darum zu kümmern, welche Server konkret unterhalb dieser Schicht laufen. Bei der Bereitstellung von Anwendungen mit mehreren Komponenten wählt KUBERNETES für jede dieser Komponenten einen Server aus, stellt sie bereit und ermöglicht es ihr, die anderen Komponenten zu finden und mit ihnen zu kommunizieren.“⁵⁰ Der Nutzen von K8s wird bei einer großen „Cloud“-Anbieterin, wie Amazon Web Services (AWS) u. a., maximiert, da es den Entwicklerinnen ermöglicht, die Ausführung und Bereitstellung von Anwendungen entkoppelt von den Systemadministratorinnen zu betreiben.⁵¹ Eine grundlegende Übersicht einer KUBERNETES-Architektur ist im Anhang A.6 auf Seite XXVI zu finden.

K8s führt einige Begrifflichkeiten ein, die nachfolgend einer Definition⁵² (siehe Tabelle 3.1) bedürfen. Diese werden im weiteren Verlauf dieser Arbeit benötigt, jedoch konzentriert auf die Forschungsfrage eins. Eine detaillierte Übersicht der Architektur der oben beschriebenen Konzepte ist im Anhang A.7 auf Seite XXVI zu sehen. Diese beschreibt das Zusammenwirken der einzelnen Komponenten.

Begriff	Definition
„Pod“	„A Pod is a group of one or more tightly coupled Containers sharing a set of Linux namespaces and cgroups.“ Innerhalb des „Pod“ wird der Netzwerk-/ „Mount“-Namensraum geteilt, dies ermöglicht die Kommunikationen innerhalb eines „Pod“. Mehrere dieser kommunizieren über <i>localhost</i> .
„Services“	„A Service is a Kubernetes object that maps one or more incoming ports to targetPorts at a selected set of target of Pods. These represent a microservice or an application running on the cluster.“
„Worker nodes“	„The worker nodes (formerly known as minions) host elements like the kubelet, kube-proxy, and the container runtime.“ Darin sind die „Pods“ enthalten, welche die Container-Anwendungen beinhalten.

⁴⁹ „Is an open-source system for automating deployment, scaling, and management of containerized applications.“ (Google LLC 2020b)

⁵⁰ Lukša 2018, S. 4.

⁵¹ vgl. Lukša 2018, S. 4.

⁵² Caban 2019, S. 10-14.

Tabelle 3.1 der vorherigen Seite fortgeführt.

Begriff	Definition
„Master nodes“	„The master nodes are the nodes hosting core elements of the control plane like (not an exhaustive list) the kubeapi-server, kube-scheduler, kube-controller-manager, and in many instances the etcd database.“ Diese übernimmt die Management-Aufgaben im Cluster.

Tabelle 3.1: Definition der „K8s“-Begrifflichkeiten

3.2 Anforderungsanalyse des zu implementierenden Prozesses

Das „statement of needs“ ist, wie in Kapitel 3.1.1 auf Seite 9 beschrieben, ein Wunsch beziehungsweise Anspruch an das zu entwickelnde System (hier: ein Prozess): Es sollen Container-Anwendungen automatisiert auf die OPENSHIFT-Umgebung verteilt werden. Dies soll von der dafür zuständigen Abteilung im Unternehmen übernommen werden, d. h. die Verantwortung dieses Prozesses wird übergeben. Ziel ist es, Aufgaben und Verantwortlichkeiten so zu verteilen, dass sie dem Aufbau- sowie Ablauforganisation entsprechen. Die Entwicklungsabteilungen/-teams möchten weiterhin nicht mehr für die Verteilung der Container-Anwendungen verantwortlich sein. Des Weiteren sprechen rechtliche Aspekte gegen dies, dass die Entwicklung auf Dauer Anwendungen in die Produktions-Umgebung (PROD) verteilt. Bei näherer Betrachtung der oben genannten Gründe ist folgendes festzustellen: es sprechen rechtliche Voraussetzung und unternehmens-organisatorische Aspekte dafür, hier ein weiteres Vorgehen anzustreben. Alleinig durch die rechtlichen Aspekte muss eine Veränderung des IST-Zustands angestrengt werden, damit ist das Vorgehen begründet und die Ermittlung der Projektbeteiligten („stakeholder“) kann durchgeführt werden.

Die Projektbeteiligten sind im engeren Sinn die Fachbereiche (also Kunden der Anwendung), die Entwicklungsteams und die „Deployment“-Abteilung. So hat der Fachbereich Interesse an einem funktionsfähigen System, die Entwicklungsteams an einem häufigen (annäherungsweise kontinuierlichem) „Deployment“ und die „Deployment“-Abteilung an einem hoch automatisierter und revisionskonformen Prozess. Dies sind nur einige Aspekte, die bei weitem nicht alle Gründe des Interesses abdecken. Es werden hier nur die „stakeholder“ im engeren Sinn weiter betrachtet, da diese Arbeit sich mit der Modellierung eines automatisierten Prozesses im Bereich der Entwicklung, der wirtschaftlichen Betrachtung und der rechtlichen Sicherung dieses beschäftigt. Der Fokus liegt auf der Modellierung des Prozesses und mit diesem deckt es die engeren Projektbeteiligten ab.

Nachfolgend wird der automatisierte „Deployment“-Prozess zur Verteilung von Containern als *System* bezeichnet. Dies erfolgt auf der Grundlage der Formulierungshilfe für Anforderungen nach Rupp 2020. Es werden ausschließlich die funktionalen Anforderungen⁵³ betrachtet, da die erfassten alle produkt-spezifisch sind – nicht-funktionale Anforderungen sind im Gegensatz dazu produkt-unspezifisch. Zur besseren Übersichtlichkeit sind die Anforderungen mit $A_1, A_2, \dots, A_n, n \in \mathbb{N}$ nummeriert und ein Anforderungskatalog ist im Anhang A.1 auf Seite XXVII zu finden. Die folgenden Anforderungen A_1, A_2, \dots, A_8 stammen von dem Projektbeteiligten („stakeholder“) „Entwicklungsteam/-abteilung“. Diese wurden während einer Befragung dieses Teams am 11. März 2020 im Rahmen einer Arbeitstagung „Container-Verteilung“ erhoben. Die Anforderungen A_9 und A_{10} beschreiben wichtige Bedingungen der Abteilung IE2.

A₁: Das System muss die Möglichkeit bieten, über eine korrekte Konfigurationsdatei, alle Komponenten der Anwendung zu verteilen.

Damit gewährleistet das System beziehungsweise der Prozess die Verteilung aller Komponenten ins OPENSHIFT-Cluster. Dieses benötigt die Konfigurationsdatei, um die Anwendung zu betreiben. Die Verteilung in OPENSHIFT funktioniert über diese Datei, damit werden nicht die fertigen Container direkt verteilt sondern es wird nur beschrieben welche Container wohin müssen. Danach übernimmt OPENSHIFT die Beschaffung der Container.

A₂: Das System soll die Möglichkeit bieten über eine standardisierte Übergabedatei alle Umgebungsvariablen des Containers zu definieren.

Diese Anforderung soll die Fehler in der Konfigurationsdatei minimieren, denn diese muss korrekt formatiert sein, sonst kann sie nicht richtig verarbeitet werden. Dabei spielen Einrückungen und Leerzeichen bei der Formatierung eine wichtige Rolle. Gleichzeitig birgt dieser Umstand großes Fehlerpotential. Mit der standardisierten Übergabedatei werden die Fehler mittels automatischem Einlesen und Generierung der Konfigurationsdatei – zumindest die Formatierungsfehler – verhindert. Dies hat keinen Einfluss auf fachliche Fehler, z. B. die Angabe einer falschen Umgebungsvariablen oder eine falsche Serveradresse der Versionskontrolle.

A₃: Das System muss die Möglichkeit bieten, über eine Validierung, konsistente Komponenten zu verteilen.

Dies beschreibt eine Kontrollinstanz, die Fehler erkennen soll und dann die Verteilung abbrechen oder eine Benutzeraktion fordern soll. Die Fehlerbearbeitung muss nicht komplett automatisiert sein, da sie meist Einzelfälle behandelt und diese nicht nach dem selben Muster zu lösen sind. Die Verteilung wird nach Behebung der/des Fehler(s) fortgesetzt.

⁵³vgl. International Organization for Standardization 2011.

A₄: Das System muss die Möglichkeit bieten unabhängig von dem Inhalt des Containers diesen zu verteilen.

A₄ beschreibt die Anforderung den Vorteil von Container-Anwendungen auszunutzen: Dieser ist die Möglichkeit beim Verteilen des Container den Inhalt ignorieren zu können, da der Container eine Abstraktionsebene darstellt.⁵⁴ Die Entwicklungsabteilung möchte diese Vorteile nutzen, „da sonst die Umstellung auf Container wenig Sinn ergeben würde“, so ihre Aussage. Dies ist keine wissenschaftlich begründete Aussage und ist deswegen als direktes Zitat markiert.

A₅: Das System muss fähig sein mit OpenShift über das „application programming interface“ (API) zu kommunizieren.

Diese Anforderung ist ein Funktionswunsch, der es ermöglicht, mittels „Command-line Interface“ (CLI)-Kommandos mit der Anwendung OPENSHIFT zu kommunizieren. Dies hat den Vorteil die internen Möglichkeiten von OPENSHIFT vollständig ausnutzen zu können, z. B. die interne Validierung von Konfigurationsdateien.

A₆: Das System soll fähig sein das OpenShift-Cluster zu jedem Zeitpunkt mit einem konsistenten Zustand zu verlassen.

Mit dieser Forderung soll gewährleistet werden, dass das System die OPENSHIFT-Umgebung nicht unbrauchbar macht und es so zur Produktionsverhinderung kommt. Dass würde bedeuten die Kunden haben kein lauffähiges System mehr und es würde zu Umsatzausfällen, Vertrauensverlust bei den Versicherungsnehmern u. a. kommen.

A₇: Das System wird die Möglichkeit bieten die Verteilung von Komponenten zu unterbrechen.

A₇ beschreibt die Forderung nach einer Notfallunterbrechung, die jedoch dem System und der OPENSHIFT-Umgebung keinen Schaden zufügt. Dies soll, wie *A₆*, ein konsistentes System hinterlassen.

A₈: Das System muss die Möglichkeit bieten, über die zentrale Verteilungssoftware Ara, die Verteilung der Komponenten durchzuführen.

Diese Forderung widerspricht dem Anspruch nach Hull, Jackson und Dick 2011, Anforderungen ohne Bezug zu einer bestimmten Anwendung beziehungsweise einer Technologie zu formulieren. Jedoch ist es eine zwingende Forderung der Abteilung IE2 diese Technologie zu verwenden. Es sollen alle Systeme/Prozesse mit dieser umgesetzt werden.

A₉: Das System muss die Möglichkeit bieten automatisiert alle notwendigen Komponenten über die Konfigurationsdatei zu erkennen und zu verteilen.

⁵⁴vgl. dazu Kapitel 3.1.3 auf Seite 15, dass die Funktionsweise und Vorteile genauer erläutert.

Dies ist eine Anforderung an die Vollständigkeit der Konfigurationsdatei, die die benötigten Komponenten beschreibt. Wichtig dabei ist es, dass die Konfigurationsdatei nicht nur vollständig ist, sondern auch semantisch korrekt ist. Ein zentrales Akzeptanzkriterium ist, dass das System automatisiert funktioniert und bei Normalbetrieb keine menschliche Aufsicht benötigt.

A₁₀: Das System muss den gesetzlichen sowie sicherheitstechnischen Vorgaben entsprechen.

A₁₀ beschreibt eine nicht-funktionale Anforderungen, die zwingend erforderlich ist und deswegen an dieser Stelle erwähnt wird. Im Bereich der Finanzdienstleistungen, welche das Versicherungswesen inkludieren, sind strenge Sicherheitsvorschriften und gesetzliche Rahmenbedingung einzuhalten (vgl. dazu die Forschungsfrage drei im Kapitel 5 auf Seite 50). Diese nicht-funktionale Anforderung ist an alle Vorhaben/Projekte gerichtet und wird von dem Projektbeteiligten „Geschäftsführung“ eingebracht. Diese Anforderung muss zwingend umgesetzt werden.

3.3 Konzeption eines Container-basierten, automatisierten „Deployments“

Dieses Teilkapitel beschreibt den Aufbau einer OPENSHIFT-Labor-Umgebung, um das Verhalten von OPENSHIFT zu testen. Des Weiteren wird eine Prozessmodellierung des automatisierten „Deployments“ anhand einer Beispielanwendung CAMUNDA erläutert. Dies ist gleichzeitig die Testapplikation, die in der SVI benutzt wird, um das automatisierte „Deployment“ von Container-Anwendungen zu untersuchen. Abschließend wird eine generische Konfigurationsdatei entwickelt, um Applikationen ins OPENSHIFT-Cluster zu verteilen.

3.3.1 Aufbau und „Deployment“ eines OpenShift-Labors

Nachfolgend wird der Aufbau einer OPENSHIFT-Labor-Umgebung mit „single node“-Architektur beschrieben, d.h., es ist nur eine „worker node“ im OPENSHIFT-Cluster vorhanden. Dieses wird auf einem UBUNTU-System in der Version 18.04 installiert. Der folgende Quelltext 3.1 beschreibt die Terminal-Eingaben:

```
1 curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo
   ↪ apt-key add -
2 sudo add-apt-repository "deb [arch=amd64]
   ↪ https://download.docker.com/linux/ubuntu $(lsb_release
   ↪ -cs) stable"
```

```

3 sudo apt update && sudo apt -y install docker-ce
4 docker version
5 sudo usermod -aG docker $USER
6 wget $LINK
7 tar xvf openshift-origin-client-tools*.tar.gz
8 cd openshift-origin-client*/
9 sudo mv oc kubectl /usr/local/bin/
10 oc version
11 cat << EOF | sudo tee /etc/docker/daemon.json
12 {
13 "insecure-registries" : [ "172.30.0.0/16" ]
14 }
15 EOF
16 sudo systemctl restart docker
17 oc cluster up

```

Quelltext 3.1: Installation des OPENSHIFT-Clusters

Das Konzept dieses „Shell“-Skripts 3.1 auf der vorherigen Seite ist es, zuerst die notwendigen Abhängigkeiten und danach die OPENSHIFT-Anwendungen zu installieren. So werden in den Zeilen eins bis drei die notwendigen Voraussetzungen geschaffen, um die DOCKER-Umgebung zu installieren. Zuerst wird der GPG⁵⁵-Schlüssel zum lokalen Schlüsselbund hinzugefügt, danach kann die Anwendung DOCKER mit dem „Advanced Packaging Tool“ (APT)-Programm heruntergeladen und installiert werden. Danach wird getestet, ob die Installation von DOCKER erfolgreich war, und der eigene Benutzer wird der Gruppe „docker“ hinzugefügt. Dies wird benötigt, um dem eigenen Benutzer die Steuerung der Software DOCKER zu erlauben. Die Zeilen sechs bis neun installieren KUBERNETES und OPENSHIFT (in der „Community“-Variante). Sehr wichtig sind die Anpassungen, die in den Zeilen elf bis fünfzehn vorgenommen werden: Sie fügen einen Eintrag "insecure-registries": ["172.30.0.0/16"] in die Datei /etc/docker/daemon.json ein. Dies erlaubt es DOCKER, auf unsichere Verzeichnisse zuzugreifen; dadurch kann OPENSHIFT lokal Container-„Images“ im Cache speichern, um diese schneller wiederzuverwenden. Schließlich wird ein Cluster in „default“-Konfiguration zur Kontrolle der Funktionsfähigkeit mit `oc cluster up` erstellt. Dies ist unter `http://127.0.0.1:8443/console` als Web-Konsole erreichbar.

Um das Verhalten des OPENSHIFT-Clusters während eines „Deployments“ zu testen, wird ein Test-Projekt erstellt. Grundsätzlich sind die meisten Kommandos der CLI nach dem Muster `oc <action> <object type> <obj name or id>`⁵⁶ aufgebaut.

⁵⁵ „GnuPG is a complete and free implementation of the OpenPGP standard as defined by RFC4880 (also known as PGP). GnuPG allows you to encrypt and sign your data and communications; it features a versatile key management system, along with access modules for all kinds of public key directories.“ Quelle: The People of the GnuPG Project 2020

⁵⁶vgl. Red Hat, Inc. 2020a.

Ein Projekt ist in OPENSHIFT ein privater Bereich, indem Applikationen laufen können und nur die Erstellerin administrativen Zugriff besitzt. Dieser kann nach außen veröffentlicht werden, wenn dies erforderlich ist. Nachdem das Projekt mit `oc → new-project <projectName> --display-name '<display_name>'`,⁵⁷ erstellt wurde, wird dies automatisch als „default“-Projekt ausgewählt, d. h., darin werden nun alle folgenden Schritte ausgeführt. Des Weiteren ist die Überlegung zu treffen, ob noch weitere Benutzer für das angelegte Projekt eine Berechtigung erhalten (`oc adm policy → add-role-to-user <admin/edit/view> <collabUser>`).

Als Test-Applikation soll eine Webseite auf Basis von DJANGO⁵⁸ in Cluster verteilt werden. Es wird ein vorgefertigtes „Image“ für diesen Test genutzt. Der Quellcode 3.2 zeigt die benötigten Kommandos, um eine neue Applikation zu erstellen. Des Weiteren illustriert dieser, wie die Anwendung nach außen freigegeben wird, d. h., sie ist außerhalb des Projektes erreichbar. Danach wird in Zeile 7 der Quellcode-Abbildung 3.2 die Verbindung zur Webseite mittels der Ermittlung des HTTP-Status-Codes überprüft.⁵⁹

```

1  oc new-app openshiftkatacoda/blog-django-py --name blog
2  oc describe svc/blog # print config
3  oc expose svn/blog
4  oc status | grep 'to pod port' # see all exposed services on
   → this project
5
6  # test_connection is listed at A.1 auf Seite XXVIII
7  test_connection blog-lab.127.0.0.1.nip.io

```

Quelltext 3.2: Test-„Deployment“ ins OPENSHIFT-Cluster

Nach Abschluss des Tests wird die Applikation mit `oc delete all --selector app=blog` wieder gelöscht.

3.3.2 Modellierung einer „Deployment“-Konfigurationsdatei

Um eine Verteilung der Container-Anwendung auf dem OPENSHIFT-Cluster durchzuführen, benötigt das System eine Konfigurationsdatei, die als API-Objekt an OPENSHIFT übergeben wird. Dabei definiert diese den gewünschten Zustand einer bestimmten Komponente der Anwendung als „pod“⁶⁰-Vorlage.⁶¹ Diese Datei stellt, als alleinige

⁵⁷ „<Platzhalter>“, die mit Werten vor der Ausführung ersetzt werden müssen.

⁵⁸ „Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design.“ Quelle: Django Software Foundation 2020

⁵⁹ Der Quelltext dieser Funktion ist im Anhang A.1 auf Seite XXVIII einzusehen.

⁶⁰ siehe Tabelle 3.1 auf Seite 19

⁶¹ vgl. Red Hat, Inc. 2019, Application → Deployments.

Quelle („single point of truth“ (SPOT)), alle notwendigen Informationen bereit, die die Anwendungen OPENSHIFT benötigt, um eine funktionsfähige Datei in dem ausgewählten Projekt des Clusters bereitzustellen. Dies ist der erste Unterschied zum klassischen „Deployment“ bei dem alle Quelldateien verteilt werden müssen.⁶² Dabei ist die Konfigurationsdatei deklarativ⁶³ beschreiben, d. h. sie beschreibt *was* OPENSHIFT mit der angegeben Komponente machen soll.

Es gibt zwei verschiedene Ausprägungen der Konfigurationsdatei: „DeploymentConfig“ und „Deployment“. Nachfolgend werden die Ziele, der Nutzen und ein Beispiel der Ausprägung beschrieben. Die „DeploymentConfig“-Variante ist von RED HAT für OPENSHIFT entwickelt worden, um eine erweiterte Unterstützung für die Softwareentwicklung und den Lebenszyklus des „Deployments“ zu unterstützen. Damit hat RED HAT ein eigenes Konzept entwickelt, wie die Verteilung von Software aussehen kann. Dieses Konzept bietet folgende Eigenschaften, die verschiedene Nutzungsmöglichkeiten ermöglichen:⁶⁴

1. „A DeploymentConfig, which is a template for running applications.“
2. „Triggers that drive automated deployments in response to events.“
3. „User-customizable deployment strategies to transition from the previous version to the new version. A strategy runs inside a Pod commonly referred as the deployment process.“
4. „A set of hooks (lifecycle hooks) for executing custom behavior in different points during the lifecycle of a deployment.“
5. „Versioning of your application in order to support rollbacks either manually or automatically in case of deployment failure.“
6. „Manual replication scaling and autoscaling.“

Allgemein definiert das „DeploymentConfig“-Objekt die Anzahl der Replikationen, beschreibt die Auslöser („trigger“) für die automatische Erstellung von neuen „Deployments“; die Strategie, mit der die Übergänge zwischen verschiedenen Versionsständen verwaltet sind, und Lebenszyklus-„Hooks“.⁶⁵ Wenn eine neue Verteilung von Komponenten angestoßen wird, aktiviert sich ein spezieller „pod“, um die die Verteilung zu überwachen und zu betreuen. Dieser fährt die alten Replikationen herunter, aktiviert neue und startet die „hooks“. Wichtig ist, dass er die alten Replikationen behält, um ein schnelles „rollback“, also eine Wiederherstellung des Zustands vor der aktuellen

⁶²vgl. Dearle 2007.

⁶³Programmierparadigma: im Gegensatz dazu gibt es das imperative Vorgehen, dass das *Was* in welcher Reihenfolge zu tun ist beschreibt.

⁶⁴vgl. Red Hat, Inc. 2019, Application → Deployments.

⁶⁵Funktionen, die beim Eintreten eines bestimmten Events automatisch ausgelöst werden

Verteilung, zu ermöglichen. Der Verteilungs-„pod“ bleibt für unbestimmte Zeit im Cluster, damit die Logdatei nicht gelöscht wird.

```

1  apiVersion: v1
2  kind: DeploymentConfig
3  metadata:
4    name: frontend
5  spec:
6    replicas: 5
7    selector:
8      name: frontend
9    template: { ... }
10   triggers:
11     - type: ConfigChange
12     - imageChangeParams:
13       automatic: true
14     containerNames:
15       - helloworld
16     from:
17       kind: ImageStreamTag
18       name: hello-openshift:latest
19     type: ImageChange
20   strategy:
21     type: Rolling

```

Quelltext 3.3: Beispiel einer „DeploymentConfig“-Datei

Dieser Quellcode 3.3⁶⁶ zeigt eine Ausprägung der „DeploymentConfig“. Die Zeilen 10 ff. beschreibt typische „trigger“, die bei einem gewissen Ereignis ausgelöst werden. Die Bezeichner beschreiben die Funktionsweise ausreichend. Die Verteilungsstrategie wird durch die Zeilen 20 f. eingestellt. Hier wird die Strategie „rolling“ verwendet, d. h. die einzelnen „pods“ einer/mehrerer „worker node(s)“ werden während dem laufenden Betrieb unbemerkt gegen die neue Version getauscht – die Endanwenderin bekommt davon nichts mit. Im Gegensatz zu der Variante „DeploymentConfig“ gibt es die KUBERNETES-Version „Deployment“, die kompatibel mit OPENSHIFT ist. Sie werden als Nachkomme der OPENSHIFT-Variante („DeploymentConfig“)⁶⁷ bezeichnet. Die „Deployment“-Ausprägung beschreibt, wie die „DeploymentConfig“, den gewünschten Zustand einer Anwendungskomponente in Form einer „pod“-Vorlage. Der Quellcode 3.4 auf der nächsten Seite⁶⁸ erzeugt bei der Ausführung im OPENSHIFT-Cluster eine Replikation des „hello-openshift“-„pod“.

⁶⁶vgl. Red Hat, Inc. 2019, Application → Deployments.

⁶⁷vgl. Red Hat, Inc. 2019, Application → Deployments.

⁶⁸vgl. Red Hat, Inc. 2019, Application → Deployments.

```

1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4      name: hello-openshift
5  spec:
6      replicas: 1
7      selector:
8          matchLabels:
9              app: hello-openshift
10     template:
11         metadata:
12             labels:
13                 app: hello-openshift
14         spec:
15             containers:
16                 - name: hello-openshift
17                     image: openshift/hello-openshift:latest
18                     ports:
19                         - containerPort: 80

```

Quelltext 3.4: Beispiel einer „Deployment“-Datei

Dabei spezifizieren die Zeilen 16-17, welches DOCKER-Abbild als Container geladen wird. Die Zeile 19 gibt den Port 80 für die Kommunikation innerhalb des „pod“ frei.

Beide Konfigurationsdatei-Ausprägungen sind vollständig nutzbar in der Anwendungen OPENSHIFT, um eine Verteilung von Komponenten zu beschreiben und auszuführen. RED HAT empfiehlt die Variante „Deployment“ zu nutzen unter der Einschränkung, dass keine Funktionen beziehungsweise kein Verhalten benötigt wird, welches nur die „DeploymentConfig“-Variante bietet.⁶⁹ Nachfolgend soll betrachtet werden, welche der möglichen Varianten für die SVI, begründet durch die Anforderungen der „stakeholder“, sinnvoll erscheint. Zuvor werden die beiden Möglichkeiten verglichen und die Unterschiede hervorgehoben: Die Varianten unterscheiden sich in einem Punkt sehr stark – dem Design. Genauer formuliert unterscheiden sie sich in der Eigenschaftsauswahl des CAP-Theorem,⁷⁰ so bevorzugt die „DeploymentConfig“- „consistency“ während die „Deployment“-Variante „availability“ bevorzugt. Übersetzt bedeutet das,

⁶⁹vgl. Red Hat, Inc. 2019, Application → Deployments.

⁷⁰„The CAP theorem states that any networked shared-data system can have at most two of three desirable properties: consistency (C) equivalent to having a single up-to-date copy of the data; high availability (A) of that data (for updates); and tolerance to network partitions (P).“ (Brewer 2012, S. 1) CA macht keinen Sinn, da nicht jeder Client dieselben Daten sehen kann und gleichzeitig immer erreichbar ist.

wenn der Verteiler-„pod“ während des „rollouts“ der „DeploymentConfig“ einen Fehler aufweist, wartet der komplette Prozess bis die „node“ mit dem „pod“ wieder aktiv wird oder manuell gelöscht wird. Während dieser Phase stockt der Prozess und es wird kein Fortschritt erreicht. Im Gegensatz dazu wird die Verteilung der „Deployment“-Variante über einen sogenannten „controller“-Manager durchgeführt, der als Hochverfügbarkeitslösung implementiert ist, d. h. die Verfügbarkeit („availability“) steht an erster Stelle. Diese Überlegung muss im Rahmen der Prozessmodellierung als erstes in der SVI vollzogen werden, um danach die spezifischen Funktionsunterschiede der beiden Konfigurationsdateien weiter zu beleuchten.

In einer Arbeitstagung zum PoC des OPENSHIFT-Systems haben die Entwicklerinnen einen ersten Entwurf für die Konfigurationsdatei vorgestellt. Hier wurde die Version „Deployment“ verwendet – begründet war dies durch das Argument, dass keine Funktionen der „DeploymentConfig“-Version benötigt wurden (Dies ist konform zur Aussage von RED HAT⁷¹). Im Anhang A.2 auf Seite XXIX ist die vollständige Version abgedruckt. Wichtig für den „stakeholder“ Entwicklerin ist, die Beschreibung der Umgebungsvariablen, die die Anwendung braucht, um mit zusätzlichen Webservices zu kommunizieren. Der Quellcode 3.5 zeigt diese Stelle nochmals, die einer besonderen Analyse bedarf. Es soll ein Algorithmus zur Generierung der Konfigurationsdateien entwickelt werden. Ziel dieses ist es, mögliche Fehler bei der manuellen Erstellung der Datei zu verhindern und ein vollständig automatisierten Prozess zu entwickeln. Die Generierung mittels Programmierung ist eine Bedingung der Abteilung IE2, die die Verantwortung für die Verteilung der Anwendungen tragen.

```

17   containers:
18     - name: policenkopie-ep-demo
19       image: $urlToRepo_/policenkopie:2.1.0
20       env:
21         - name: INKASSOMAHNINFOPORT2_ENDPOINT_URI
22           value: $urlToServer_:10702/mock/$id_
23         - name: DOKUMENTINFORMATION1_ENDPOINT_URI
24           value: $urlToServer_:10701/mock/$id_
25         - name: VERTRAGPRUEFUNG1_ENDPOINT_URI
26           value: $urlToServer_:10703/mock/$id_
27         - name: DRUCKAUFTRAGPORT_ENDPOINT_URI
28           value: $urlToServer_:10704/mock/$id_
29         - name: PRODUKTAUSKUNFT1_ENDPOINT_URI
30           value: $urlToServer_:10705/mock/$id_
31         - name:
32           → PARTNERPRUEFUNGASYNCHRON_ENDPOINT_URI
              value: $urlToServer_:10709/mock/$id_

```

⁷¹vgl. Red Hat, Inc. 2019, Application → Deployments.

```

33      - name: PROZESSAKTIVITAET2_ENDPOINT_URI
34          value: $urlToServer_:10706/mock/$id_
35
36      - name: SPRING_DATASOURCE_URL
37          value: $urlToJDBC_
38
39      - name: SPRING_DATASOURCE_USERNAME
40          value: $usernameDatasource_
41
42      - name: SPRING_DATASOURCE_PASSWORD
43          value: $passwordDatasource_
44
45      - name: CAMUNDA_BPM_DATABASE_TYPE
46          value: $databaseType_

```

Quelltext 3.5: Ausschnitt aus Quellcode A.2: Deklaration der Umgebungsvariablen

Die Zeichenketten mit dem Muster „\$Bezeichner_“ stellen Platzhalter dar, die genutzt werden, um die eigentlichen Werte nicht in Klartext abzudrucken. Diese fallen unter das Betriebsgeheimnis. Platzhalter mit der gleichen Zeichenkette haben den identischen Inhalt. Es fällt auf, dass die „Uniform Resource Locator“ (URL) der Webservices immer mit dem gleichen Host beginnt und ein Teil des Pfades auch gleich ist. Es gibt zwei Möglichkeiten, wie der Algorithmus zur Generierung der Konfigurationsdateien aussehen könnte: Bei der „Deployment“-Variante müsste dieser die komplette Datei anhand definierter Regeln generieren. Die zweite Möglichkeit ist die Nutzung eines Vorlagen-Objektes („template“), das einmal erstellt, unter gewissen Bedingungen, wiederverwendet werden kann. Dies bedeutet, dass für die Beschreibung der Komponenten-Verteilung die „DeploymentConfig“-Variante genutzt werden muss, da das „template“-Objekt nur diese Variante unterstützt. Beides sind Eigenentwicklung von RED HAT. Die Ausprägung „Deployment“ ist ein natives K8s-Objekt und deswegen nicht mit dem „template“-Objekt kompatibel. Es besteht die Möglichkeit diese Objekt mit Parametern zu beschreiben, die in einer Datei <Anwendung>.env gespeichert und mittels der CLI eingelesen werden. Diese ersetzt dann die Parameter mit Werten aus der Datei. Die Entscheidung eine Schlüssel-Wert (K-V)-Datei, für die Umgebungsvariablen zu nutzen, hat den Vorteil, dass diese extern in einer verteilten Versionsverwaltungsanwendung (beispielsweise GIT oder SERENA DIMENSIONS) gespeichert werden kann. Nachfolgend ist der Aufbau einer K-V-Datei als Quellcode 3.6 für die Umgebungsvariablen des Beispiels A.2 auf Seite XXIX angedeutet. Das Trennzeichen zwischen Schlüssel und Wert ist das Gleichheitszeichen („=“). Dieser Aufbau ist verpflichtend, wenn die CLI von OPENSHIFT genutzt wird.

```

1 PROZESSAKTIVITAET2_ENDPOINT_URI=$urlToServer_:10706/mock/$id_
2 SPRING_DATASOURCE_URL=$urlToJDBC_
3 SPRING_DATASOURCE_USERNAME=$usernameDatasource_
4 SPRING_DATASOURCE_PASSWORD=$passwordDatasource_
5 CAMUNDA_BPM_DATABASE_TYPE=$databaseType_

```

Quelltext 3.6: Umgebungsvariablen als K-V-Datei

Eine Anforderung der Abteilung IE2 ist es, die Konfigurationsdatei über eine Generierung zu erstellen, um mögliche Syntaxfehler ausschließen zu können. Des Weiteren soll so eine Standardisierung der Konfigurationsdatei unternehmensweit erzielt werden. Das Vorgehen dazu soll folgende Stufen enthalten:

1. Eine einheitliche Übergabedatei soll von den Entwicklungsteams in einer Versionsverwaltung abgelegt werden. Die Datei ist nach Vorgaben der Abteilung IE2 aufgebaut.
2. Danach soll aus dieser Datei die Konfigurationsdatei erzeugt werden.
3. Diese wird durch OPENSHIFT validiert, bevor sie ins Cluster geladen wird.
4. Das Ergebnis ist die vollständige Konfigurationsdatei, die die Komponenten beschreibt, die verteilt werden sollen.

Für die Generierung ist es wichtig ein Grundgerüst der Konfigurationsdatei zu haben. Dieses ist im Anhang A.4 auf Seite XXXII mit beispielhaften Werten zu sehen. Zweck des Grundaufbau ist es, bei der Generierung die fehlenden Informationen möglichst leicht zu integrieren. Dabei hilft es, wenn eine Grundstruktur des Ergebnisses vorhanden ist. Der Aufbau der zulässigen Wortfolgen in einer Zeile der Übergabedatei ist formal korrekt in der Quellcode-Darstellung A.3 auf Seite XXXI im Anhang als Backus-Naur-Form (BNF)⁷² beschrieben. Informell werden nachfolgend zwei mögliche reguläre Ausdrücke dargestellt, die Muster einer Zeile der Übergabedatei beschreiben. Es wird zwischen einer Gruppierungszeile und einer K-V-Zeile unterschieden. Der Reguläre Ausdruck der Gruppierungszeile lautet „\{1}([A-Z]+\}\{1)“ und der Ausdruck der K-V-Zeile „([A-Z]+)\{1)[:ascii:]+“. Dabei stellt „[:ascii:]“ eine Zeichenklasse dar, die alle ASCII-Zeilen enthält. Ein Beispiel einer validen Übergabedatei ist im Anhang A.5 zur Veranschaulichung des Konzeptes abgebildet. Nachfolgend beschreibt der Algorithmus 1 die Logik zur Generierung einer Konfigurationsdatei. Dieser ist absichtlich in Pseudocode verfasst, um den Fokus auf die Logik zu setzen und nicht auf Besonderheiten einer Programmiersprache. Die Idee dieses ist es, die Übergabedatei einzulesen, auszuwerten und auf Basis dieser eine valide Konfigurationsdatei zu erstellen.

Ziel des Algorithmus 1 auf der nächsten Seite ist es, aus einer definierten Übergabedatei (siehe dazu Quellcode A.3 auf Seite XXXI) die Konfigurationsdateivorlage („yamlTemplate“) mit den Übergabedaten anzureichern. Dabei werden zuerst mit einer Schleife alle Zeilen der Übergabedatei eingelesen und nach zwei Kriterien klassifiziert. Kriterium eins beziehungsweise Fall eins bedeutet, dass die Zeile ein neues Gruppendelement (später in der YAML-Datei das Element der aktuellen Ebene minus

⁷² „Using BNF it is possible to specify which sequences of symbols constitute a syntactically valid program in a given language. (The question of semantics—i.e., what such valid strings of symbols mean—must be specified separately.)“ Quelle: McCracken und Reilly 2003

Algorithmus 1 Generierung der Konfigurationsdatei aus einer K-V-Datei

```

procedure GENERATEDEPLOYMENTCONFIG(pathToKVFile)
    path ← pathToKVFile
    map[ ][ ]           ▷ Map with a key and a value, e.g. a array of lines as value
    lines[ ] ← READFILEBYLINES(path)
    currentKey = null
    yamlTemplate ← READFILE(pathToYamlTemplate)

    for all lines[ ] as line do
        if line like REGEx("\{1\}([A - Z]+)\}\{1\}") then      ▷ New group element?
            keyword ← TRIM(line, "[", "]")
            currentKey ← keyword
            map.ADDKEY(currentKey)
        else                                              ▷ line contains a k-v-pair
            map[currentKey].ADD(line)
        end if
    end for          ▷ All lines of file are stored in k-v-pairs in var map now

    allKeys[ ] ← map.GETALLKEYS
    for i ← 0, i ≤ LENGTH(allKeys[ ]), i ++ do
        for all lines of map[allKeys[i]] as line do
            ADDTOYAML(yamlFile, lineToBeAdded, key)
        end for
    end for
end procedure

```

eins) enthält. Damit ist ein Schlüsselement gefunden. Diesem werden nun im alternativen Fall alle folgende Zeilen zu geordnet bis wieder ein Gruppenelement gefunden ist. Nach Beendigung der ersten Schleife sind alle Zeilen in einem „map“-Objekt als K-V-Paare gespeichert. Diese werden an der entsprechenden Stelle der Konfigurationsdatei hinzugefügt. Nach Beendigung dieses Programms ist die Konfigurationsdatei mit den entsprechenden Werten der Übergabedatei beschrieben. Ein mögliche Implementierung könnte sich auch die Struktur der YAML-Datei zu nutzen machen: Es gibt für verschiedene Programmiersprachen Module, die eine Objektstruktur in die entsprechende YAML-Syntax überführen. Mit solch einem Modul wäre die zweite Schleife des Algorithmus 1 nicht notwendig. So würde die Objektstruktur eingelesen werden, danach mit mittels Objektmanipulation (wie löschen, hinzufügen beziehungsweise editieren der Attribute) die entsprechenden Stellen angepasst und am Ende des Programms wieder in eine YAML-Datei überführt. Die aktuelle Version des Algorithmus

1 ist bezüglich seiner Laufzeit $T_P(x)$ ⁷³ und der Komplexität \mathcal{O} nicht optimiert. Wenn die oben beschriebene Bibliothek⁷⁴ die YAML-Datei in eine Struktur übersetzt auf die direkt zugegriffen werden kann, d. h. die Komplexität des Zugriffes entspricht $\mathcal{O}(1)$, verbessert sich die Laufzeit T des Algorithmus 1 auf der vorherigen Seite erheblich. Es ist die Verwendungen solch einer Bibliothek zu empfehlen, da die Komplexität des Algorithmus und die Laufzeit verbessert wird. Des Weiteren reduziert sich der Entwicklungsaufwand des Programms, da die Umsetzung der YAML-Datei ohne zusätzliche Entwicklung einer Funktion auskommt.

3.3.3 Prozessmodellierung anhand der Anwendung Camunda

Die Überschrift dieses Kapitels bezieht sich auf den Verteilungsprozess der Container-Anwendung CAMUNDA und nicht auf die Geschäftsprozessmodellierung, die CAMUNDA anbietet. Dabei ist in dem Container, der verteilt werden soll, der Prozess mit Quellcode abgebildet. Die Formulierung „Anwendung CAMUNDA“ inkludiert sowohl die Anwendung als solche (das „Base Image“ im Wortlaut von DOCKER) als auch den zusätzlichen Quellcode, den die Entwicklungsabteilung erstellt hat. Nachfolgend wird aus platztechnischen Gründen von der *Anwendung CAMUNDA* gesprochen. Die Formulierung „Prozessmodellierung“ bezieht sich auf den oben genannten Verteilungsprozess.

CAMUNDA ist ein freies⁷⁵ „Workflow“-Management-System, mit dem Geschäftsprozess unter der Verwendung des Standards BPMN 2.0⁷⁶ modelliert und ausgeführt werden können.⁷⁷ Es ist in Java geschrieben und als Container-Anwendung in DOCKER verfügbar. Die SVI nutzt zusammen mit ihrem Kunden, der SV, die Container-Variante. Die Entwicklungsabteilung beschreibt das Vorgehen zur Entwicklung so: Der Fachbereich des Kunden SV entwickelt per „Drag&Drop“ ein Prozessablaufmodell, dass danach durch die Entwicklungsteams mit Logik (Programmcode in Java) versorgt wird. Für die Prozessmodellierung ist der Aufbau beziehungsweise Inhalt des Containers irrelevant.

Der Prozess muss zwingend mit der zentralen „Deployment“-Anwendung ARA erstellt und betrieben werden (vgl. dazu Anforderung A_8 der Abteilung IE2). Die wesentlichen Bestandteile sind die Übergabedatei als „Input“, die Konfigurationsvorlage als „Input“ und die angereicherte Konfigurationsdatei als „Output“. Die Übergabedatei wurde per BNF in der Quellcodeabbildung A.3 auf Seite XXXI standardisiert. Diese

⁷³ T entspricht der Laufzeit des Programmes P auf x Eingaben

⁷⁴ hier eine Sammlung von fertigen Programmen zur Verwendung

⁷⁵ „Open source“, teilweise

⁷⁶ Object Management Group (OMG) 2011.

⁷⁷ vgl. Camunda Services GmbH 2020.

Grammatik wird der Entwicklungsabteilung zur Verfügung gestellt, um der Entwicklerin den genauen Aufbau der Übergabedatei zu erklären. Dieses Verfahren verhindert Unklarheiten bezüglich der Syntax. Außerdem könnte eine interessierte Entwicklerin ein Programm entwickeln, das auf Grundlage dieser Grammatik die Übergabedatei auf Syntaxfehler überprüft, bevor sie der „Deployment“-Abteilung IE2 zur Verfügung gestellt wird. Diese Datei wird in einer Versionsverwaltung (hier SERENA DIMENSIONS) übergeben. Nach erfolgreichem Herunterladen der Datei auf einen Arbeitsserver (hier „DIM.S-V“) beginnt die Verarbeitung dieser Datei. Für diese wird initial der Algorithmus 1 auf Seite 31 verwendet. Zuvor wurde die Konfigurationsdateivorlage aus der Versionsverwaltung auf den Arbeitsserver geladen. Der Algorithmus 1 auf Seite 31 wurde in einer veränderten Form durch die Programmiersprache PYTHON implementiert. Dieser wurde verändert, da PYTHON mit dem Modul PYYAML eine verbesserte Unterstützung bei der Bearbeitung von YAML-Dateien anbietet. Dabei übersetzt das Modul die YAML-Datei in ein Objekt, das mit allen möglichen, unterstützten Operationen bearbeitet werden kann. Danach wird das Objekt in eine YAML-Datei umgewandelt. Da die Konfigurationsdatei in diesem Format beschrieben ist, bietet sich diese Vorgehensweise an. Nach erfolgreicher Anreicherung der Konfigurationsdatei muss diese an OPENSHIFT übergeben werden. Dafür ist der Übergabeserver (hier „OSHIFT.S-V“) zuständig. Dieser Server führt vor Abschluss des Prozesses mehrere Schritte durch: eine Validierung der Konfigurationsdatei, hochladen der Datei auf das OPENSHIFT-Cluster (hier Cluster) ins entsprechende Projekt, die Breitstellung für alle berechtigten Benutzerinnen des Projektes und die Aktivierung der Konfiguration, dass die tatsächliche Verteilung der Anwendung entspricht. Danach ist diese sichtbar für die Kunden. Um diesen Prozess visuell zu verdeutlichen, wird dieser mit einem adaptierten UML-Sequenzdiagramm in Abbildung 3.3 auf der nächsten Seite dargestellt. Dieses Vorgehen ist in Kapitel 2 auf Seite 6 begründet.

Die Anforderungen A_1, \dots, A_{10} wurden auf unterschiedliche Weise in diesem Prozess umgesetzt. A_1 und A_2 beziehen sich auf die Konfigurationsdatei und die Übergabedatei. Die Anforderungen wurden durch die Standardisierung der Übergabedatei, die Generierung der Konfigurationsdatei und die Validierung dieser umgesetzt. Die Möglichkeit der Definition von Umgebungsvariablen ist in der Übergabedatei gegeben. Über die Konfigurationsdatei werden die Komponenten verteilt. A_3 beschreibt die Anforderung, dass konsistente Komponenten verteilt werden. Die Validierung dieser erfolgt durch das OPENSHIFT-Cluster und ist somit gewährleistet. Des Weiteren unterstützt der, in Abbildung 3.3, dargestellte Prozess die Anforderung A_4 , indem eine Konfigurationsdatei als „Deployment“-Artefakt verteilt wird. Damit ist der Prozess entkoppelt von dem Inhalt des Containers. Anforderung A_5 bis A_7 wird durch die direkte Kommunikation (über die API) der Verteilungsanwendung ARA erfüllt. ARA kommuniziert ausschließlich über die OPENSHIFT-API. Da diese Verteilungsanwendung automatisch und revisions-konform ist, sind alle verbleibenden Anforderungen erfüllt. Die Funktionalitäten der Methoden, die in Abbildung 3.3 verwendet wurden,

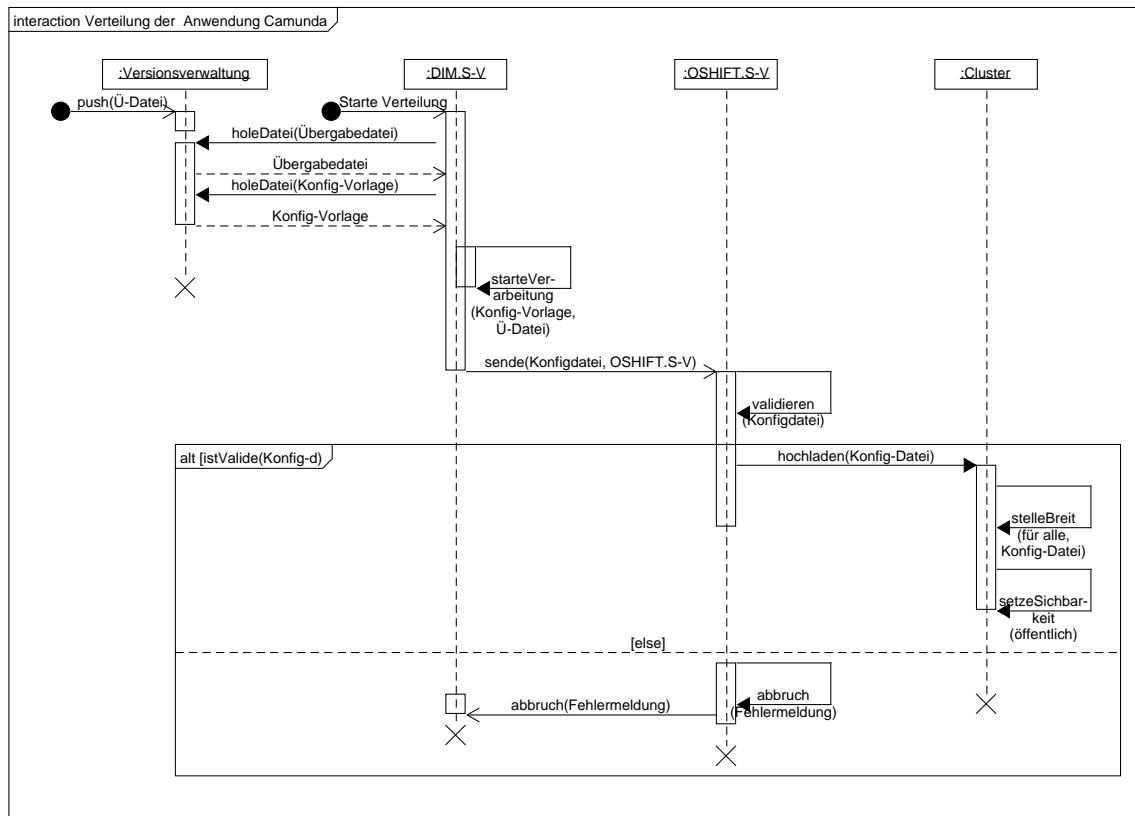


Abbildung 3.3: (adaptiertes) Sequenzdiagramm zur Verteilung der Anwendung CAMUNDA

Quelle: eigene Darstellung
unternehmensintern

werden in der Tabelle 3.2 auf der nächsten Seite kurz beschrieben. Dies dient der Übersichtlichkeit und dem Verständnis.

3.4 Ergebnis der Forschungsfrage eins

Die Ergebnisse der Forschungsfrage eins – Wie können Container-Anwendungen den Prozess des automatisierten „Deployments“ unterstützen? – werden nachfolgend kurz zusammengefasst. Eine kritische Betrachtung des gesamten Ergebnisses der Bachelorarbeit ist im Epilog (siehe Kapitel 6 auf Seite 58) zu finden.

Prozess wird generisch(er) Durch die Verwendung dieses Prozess können in Zukunft viele Anwendungen mit nur einem Prozess verteilt werden. Dies ist bedingt

Methode	Beschreibung
push(Ü-Datei)	Veröffentlicht die Übergabedatei in der Versionsverwaltung.
holeDatei(Datei)	Lädt die angegebene Datei von der Versionsverwaltung auf einen Server, der die Methode aufgerufen hat.
starteVerteilung(Konfig-Vorlage, Ü-Datei)	Damit wird die Erstellung der angereicherten Konfigurationsdatei gestartet.
sende(Konfig-Datei, Server)	Diese Methode versendet die Konfigurationsdatei an den entsprechend Server.
validieren(Konfig-Datei)	Überprüft die Konfigurationsdatei auf Syntax-Fehler.
hochladen(Konfig-Datei)	Lädt die Datei in OPENSHIFT-Cluster hoch.
stelleBereit(für wen?, Datei)	Diese Methode schaltet die angegebene Datei für alle berechtigten Benutzerinnen auf dem Cluster frei.
setzeSichtbarkeit(Sichtbarkeit)	Veröffentlicht die Komponenten, die durch die Konfigurationsdatei beschrieben wurden. Damit ist der Verteilungsvorgang abgeschlossen.
abbruch(Fehlermeldung)	Bricht den Verteilungsprozess ab und gibt eine Fehlermeldung zurück.

Tabelle 3.2: Überblick über die verwendeten Methoden in Abbildung 3.3 auf der vorherigen Seite

durch die Entkopplung der Verteilungslogik und dem eigentlichen Prozess. Da OPENSHIFT nach dem Erhalt der Konfigurationsdatei die wirkliche Verteilung der Komponenten übernimmt, hat sich das komplette Verständnis eines „Deployments“ innerhalb der SVI verändert beziehungsweise wird sich verändern. Der neuartige Prozess verteilt nur noch die Konfiguration(-sdatei) und nicht, wie bei den vorherigen Prozessen, den Programmcode oder das Kompilat des Codes. Damit kann der Prozess eine Verteilungsobjekt-unabhängige Logik verwenden. Dieser profitiert von den oben genannten Vorteilen der Container-Anwendungen (vgl. dazu 3.1.3 auf Seite 15). Des Weiteren profitiert die Endkundin von der ständigen Verfügbarkeit der „Cloud“, da eine Verteilung einer neuen Version während der Betriebszeiten gemacht werden kann – dadurch wird eine schnellere TTM erreicht. Der, in Abbildung 3.3 auf der vorherigen

Seite abgebildete, Prozess stellt einen ersten Versuch einer automatisierten Verteilung (von Container-Anwendungen) dar. In weiteren Evaluierungen muss der neue Prozess seine Vorteile unter Beweis stellen. Bis dieser von den Verantwortlichen freigegeben wird und produktiv geht, sind noch einige Schritte notwendig, wie z. B. die Qualitäts-sicherung und die Prüfung auf Revisions-Tauglichkeit.

Aufwand ist langfristig geringer Dieser Teil des Ergebnisses folgt aus der Aussage, dass der Prozess generischer wird: Der anfängliche Aufwand der Entwicklung eines Container-„Deployment“-Prozesses ist im Unternehmen SVI höher, da noch kein Wissen über die Container-/„Cloud“-Technologie vorhanden war. Des Weiteren gab es kein etabliertes Vorgehen, wie Container-Anwendungen im Bereich der Verteilung behandelt werden. So musste eine Übergabe- und Konfigurationsdatei entwickelt werden, die Infrastruktur (nicht Teil dieser Arbeit) für OPENSHIFT und Container-/„Cloud“-Produkte geschaffen werden. Außerdem wurden Regeln entwickelt (nicht Teil dieser Arbeit), wie die Zusammenarbeit zwischen den einzelnen, beteiligten Abteilungen funktioniert. Ist der Verteilungsprozess entwickelt und läuft stabil, benötigt dieser keinen weiteren Entwicklungsaufwand. Solange die Anforderungen der Plattform „OpenShift“ nicht verändern, benötigt der Prozess fachlich keine Weiterentwicklung. Somit pendelt sich der Aufwand auf einem konstant niedrigem Niveau ein. Die verbleibenden, aufwand-verursachenden Aufgaben beschränken sich auf die Betreuung im Fehlerfall (Konfigurationsdatei ist fachlich nicht korrekt) und auf die Überwachung der Verteilung neuer Anwendungen. Im Gegensatz dazu stehen die aktuellen Prozesse, die für die Bestandssysteme der SV implementiert sind. Hier müssen ständig Anpassungen am Verteilungsprozess gemacht werden, da Kompilate und Quellcode verteilt wird und der „Deployment“-Prozess auch Kompilierungen vornimmt.

Verbesserte Kontrolle des Erfolges Dies ist bedingt durch die Verteilung einer einzigen Konfigurationsdatei während eines „Deployments“. Es müssen nicht mehr tausende Artefakte eines Programms verteilt werden. Dadurch ist die Kontrolle des Erfolges beziehungsweise Misserfolges im Vergleich zum jetzigen Prozess einfacher. Im Fehlerfall muss eine Datei, die Konfigurationsdatei, überprüft werden. Die Generierung dieser Datei beschränkt sich auf einen Generierungsalgorithmus, der aus einer Übergabedatei die Konfigurierung ableitet. Wenn dieser getestet ist und fehlerfrei läuft, ist die Fehlerquelle bei einer falschen Verteilung von Komponenten initial beschränkt auf die Übergabedatei. Diese kann einfach kontrolliert und verbessert werden. Es sind immer noch Fehler möglich, jedoch ist die Anzahl der möglichen Fehlerquellen gesunken.

Kritik am Generierungsalgorithmus 1 Dieser Algorithmus 1 auf Seite 31 ist, wie schon erwähnt, bezüglich seiner Laufzeit $T(x)$ und Komplexität \mathcal{O} nicht optimiert.

Somit Bedarf dieser noch weiteren Verbesserungen, um die Leistung zu steigern. Problematisch sind die ineinander kombinierten Schleifen, da sie die Laufzeit im schlechtesten Fall polynominal, d. h. pro Schleife erhöht sich der (positive) Exponent von n^c um eins und damit die Komplexitätsklasse $\mathcal{O}(n^c)$, $c \geq 1$. Wie oben beschrieben kann über die Vermeidung von ineinander kombinierten Schleifen die Komplexität klein gehalten werden. Eine erste Möglichkeit ist es, das Modul PyYAML zu nutzen, dass durch die Übersetzung der Konfigurationsdatei in ein K-V-Objekt Zugriffe der Komplexität $\mathcal{O}(1)$ zulässt. Die Komplexität $\mathcal{O}(1)$ bedeutet übersetzt in diesem Kontext ein direkter Zugriff auf ein Element.

Ausblick: Weitere Evaluierung notwendig Um die Qualität des Prozesses weiter zu verbessern, sind weitere Schritte notwendig. So müssen die Regeln zur Übergabe der Dateien, zum Aufbau der Konfigurationsdatei und zur Zusammenarbeit erstellt beziehungsweise verbessert werden. Auch müssen Management-Entscheidungen zur Standardisierung dieses Prozess angestoßen und verfolgt werden. Eine weitere Empfehlung ist, den langfristigen Nutzen dieses Prozess zu analysieren, um Verbesserungspotentiale zu finden. Die Forschungsfrage eins versteht sich als ersten Versuch ein lauffähiges Container-„Deployment“ in der SVI zu entwickeln – es sind weitere Schritte bis zum Produktivgang des Prozesses nötig.

4 Welche wirtschaftlichen Vorteile hat der Einsatz von Containern auf den Prozess des automatisierten „Deployments“?

In diesem Kapitel ...

4.1 Grundlagen: Definition der Begrifflichkeiten zur Forschungsfrage zwei

Dieses Teilkapitel soll grundlegende Begrifflichkeiten, die im weiteren Verlauf dieser Arbeit verwendet werden, definieren, um so eine einheitliche Terminologie der Begriffe zu entwickeln. Dadurch wird ein gemeinsames Verständnis erzeugt.

Geschäftsprozessanalyse Der Begriff „Geschäftsprozess“ beschreibt eine zusammenhängende Folge von Aufgaben beziehungsweise Tätigkeiten, die in einem Unternehmen wiederkehrend abgeschlossen werden, um die Unternehmens- und Organisationsziele zu erreichen. Die Analyse untersucht schlussendlich dieselben Sachverhalte wie auch die klassischen Ansätze der Organisationslehre.⁷⁸ Diese sind die Effizienzsteigerung und die Einsparung. Dabei werden die zu leistenden Tätigkeiten, Aufgaben und Arbeitsabläufe auf die genannten Ansätze optimiert. Im Vergleich zur klassischen Optimierung steht bei der Geschäftsprozessanalyse eine andere Perspektive im Fokus. Hier werden die „längere(n) zusammenhängende(n) Folgen von Tätigkeiten, die zur Erledigung einer größeren Aufgabe nötig sind“,⁷⁹ betrachtet. Damit ist der gesamte Ablauf eines Prozesses als Ausgangspunkt der Analyse zu betrachten und nicht mehr nur einzelne Tätigkeiten und Stellen.

Um das weitere Verständnis der Begrifflichkeiten zu fördern, werden folgende Begriffe definiert⁸⁰: Aufgaben und deren Eigenschaften, Aufgabenfolgen und Funktionen. Aufgaben sind Teilarbeitspakete einer Tätigkeit, die auf unterschiedlichen Ebenen betrachtet werden können. Die kleinste Einheit einer Aufgabe ist die Elementaraufgabe,

⁷⁸vgl. Staud 2006, S. 5.

⁷⁹Staud 2006, S. 5.

⁸⁰vgl. Staud 2006, S. 4-5.

die nicht weiter teilbar ist. Wichtig ist, dass Aufgaben teilbar und wieder zusammenfassbar sind. Dadurch wird eine unterschiedliche Aggregationsstufe erreicht. Das Problem der Aggregation ist, dass die Modelliererin, geprägt durch ihre Wahrnehmung, die Ebene der Betrachtung einer Aufgabe bzw. Tätigkeit subjektiviert und so das Ergebnis stark beeinflusst wird – so auch die Länge der Geschäftsprozesse. Die sequenzielle Folge von Aufgaben entsteht durch die Erstellung eines Vorgangs, der eine Abfolge von Tätigkeiten zur Realisierung von Aufgaben beschreibt. Schließlich wird ein Geschäftsprozess von Staud 2006 definiert als: „[...] besteht aus einer zusammenhängenden abgeschlossenen Folge von Tätigkeiten, die zur Erfüllung einer betrieblichen Aufgabe notwendig sind. Die Tätigkeiten werden von Aufgabenträgern in organisatorischen Einheiten unter Nutzung der benötigten Produktionsfaktoren geleistet. Unterstützt wird die Abwicklung der Geschäftsprozesse durch das Informations- und Kommunikationssystem IKS des Unternehmens.“⁸¹ Eine weitere Definition charakterisiert den Geschäftsprozess als „[...] eine zielgerichtete, zeitlich logische Abfolge von Aufgaben, die arbeitsteilig von mehreren Organisationen oder Organisationseinheiten unter Nutzung von Informations- und Kommunikationstechnologien ausgeführt werden können. Er dient der Erstellung von Leistungen entsprechend den vorgegebenen, aus der Unternehmensstrategie abgeleiteten Prozesszielen. Ein Geschäftsprozess kann formal auf unterschiedlichen Detaillierungsebenen und aus mehreren Sichten beschrieben werden.“⁸² Die zweite Definition wird in dieser Arbeit verwendet, denn sie stellt die Unternehmensstrategie als zentralen Messfaktor in den Mittelpunkt. Werden alle Geschäftsprozesse linear kombiniert, entsteht die Darstellung der Wertschöpfungskette eines Unternehmens. Deswegen gibt es nur systemrelevante Geschäftsprozesse in einem Unternehmen. Sie können noch Optimierungspotenzial enthalten, jedoch sind sie nie unnötig oder nicht brauchbar. Ein Geschäftsprozess wird am Kunden orientiert. Es wird zwischen Kern- und unterstützenden Prozessen unterschieden: bei Kernprozessen handelt es sich um die Hauptleistung eines Unternehmens, wie die Produktion eines Autos bei einem Autohersteller. Die Unterteilung in Kern- und unterstützende Prozesse beschreibt dabei nicht die Wichtigkeit dieser; es ist also keine Einteilung in weniger wichtig und wichtig vorzunehmen.⁸³

Geschäftsprozesse haben verschiedene Eigenschaften, wie der Automatisierungsgrad, die Datenintegration und die Prozessintegration. Der Automatisierungsgrad beschreibt, wie groß der Anteil der Aufgabenerfüllung ist, welcher dunkel, d. h. ohne menschliche Interaktion, bewältigt werden kann. Die Datenintegration ist ein wichtiger Bestandteil bei Optimierungsvorhaben, denn sie sollte bei 100 Prozent liegen, um Inkonsistenzen der Daten auszuschließen. Bei weniger als 100 Prozent entwickeln sich Parallelwelten im Unternehmen. Ist ein Geschäftsprozess über viele verschiedene traditionelle Organisationsbereiche aufgespannt, so ist seine Prozessintegration hoch. Gibt es Or-

⁸¹Staud 2006, S. 9.

⁸²Gadatsch 2010, S. 41.

⁸³vgl. Staud 2006, S. 11.

ganisationsbrüche, d. h., wird ein Prozess aktiv an einer beteiligten Abteilung vorbeigeführt, muss die Notwendigkeit dieser Maßnahme bei der Optimierung überprüft werden. Zu den Komponenten der Geschäftsobjekte: Je nach Ziel der Untersuchung können viele Komponenten beteiligt sein und können damit identifiziert werden. Die BWL beschränkt diese auf die formellen Strukturen einer Organisation und auf das Handeln der Beteiligten, das unmittelbar Einfluss auf den Geschäftsprozess hat.⁸⁴

Ziel der Geschäftsprozessanalyse ist es, eine Ist-Analyse des Prozesses durchzuführen, um so eine Bestandsaufnahme vorhalten zu können, und eine Optimierung des Prozesses, die die Beseitigung von Schwachstellen zur Folge hat. Diese Schwachstellen werden bei der Ist-Analyse entdeckt. Einschränkend zu erwähnen ist, dass die Methodik der Geschäftsprozessanalyse nicht genau definiert ist, da die Identifikation (Detaillierungsgrad) und Abgrenzung (Länge) der Prozesse subjektiv beeinflusst werden. Das Modell der Ereignisgesteuerten Prozesskette(n) (EPK) ist eine mögliche Methodik, um Geschäftsprozesse zu analysieren und zu beschreiben.⁸⁵ EPK ist ein Vorgehensmodell zur sichten-orientierten Modellierung von Geschäftsprozessen, bei dem ein Prozess und seine dazugehörenden Funktionen in einer zeitlich-logischen Abfolge illustriert werden.⁸⁶ Die Kontrollflussteuerung zwischen den einzelnen Funktionen eines Geschäftsprozess werden über Geschäftsregeln gesteuert. Diese beinhalten folgende Konstrukte: Ereignis, Bedingung und Funktion/Methode/Aktion. Entscheidungen werden über die verfügbaren Verknüpfungsfunktionen modelliert:⁸⁷ AND-, OR- und XOR-Verknüpfung.⁸⁸ Des Weiteren gibt es eine andere Methodik, die in einem Standard, BPMN Version 2.0,⁸⁹ definiert ist, der Geschäftsprozesse modelliert. Außerdem wurde dieser Standard in der Norm ISO/IEC 19510 verankert.⁹⁰ Im Gegensatz zur EPK fokussiert sich BPMN rein auf die Modellierung eines Prozesses und nicht auf folgende Strukturen: Prozesslandschaft, Aufbauorganisation, Daten, Strategie, Geschäftsregeln und IT-Landschaft.⁹¹ Es gibt eine Software-Lösung für BPMN, die in der SVI eingesetzt wird und später die erste Container-Applikation für den neuen „Deployment“-Prozess ist.

„Business Case“ In diesem Teilkapitel werden der Aufbau eines Geschäftsszenarios und die grundsätzliche Methodik erläutert, da eine tiefgreifende, ausführliche Beschreibung des gesamten Themenkomplexes den Rahmen dieser Arbeit überschreitet: Eine

⁸⁴vgl. Staud 2006, S. 15.

⁸⁵vgl. Staud 2006, S. 59.

⁸⁶vgl. Scheer, Nüttgens und Zimmermann 1997, S. 4.

⁸⁷vgl. Scheer, Nüttgens und Zimmermann 1997, S. 4.

⁸⁸Hier wird an die gängige Schreibweise der Logikgatter angeknüpft.

⁸⁹Object Management Group (OMG) 2011.

⁹⁰ICT/1 2020.

⁹¹vgl. Freund und Rücker 2017, S. 28.

vollumfängliche Betrachtung eines „Business Case“ kann eine eigene Bachelor-Thesis darstellen.

Die Erstellung eines Geschäftsvorfalls (engl. „Business Case“) ist für das Unternehmen bei der Betrachtung eines Projekts von elementarer Bedeutung. Ohne dessen Erstellung könnten folgende Probleme mit hoher Wahrscheinlichkeit auftreten⁹²:

- „The organization wastes valuable resources on projects that don't help the organization achieve its objectives. This leaves fewer resources available for more valuable projects.
- The organization has no clear basis to prioritize projects, for establishing what is important. Without a Business Case—and some organization-wide agreed measure of “value”—there is no means of determining which projects are important, and which are less so.
- There is likely to be disappointment after the completion of the project, as the stakeholders wonder why the project is not giving the great results they imagined [...].
- No target is established for why the project's deliverables are being created—other than the meeting of technical specifications.
- The organization has no opportunity to improve its project management maturity. One key learning from each project should be: “how well did the resource usage support the organization's goals?”“

Grundsätzlich ist ein Geschäftsszenario eine betriebswirtschaftliche Beurteilung einer Investition. Dabei werden deren Kosten und Nutzen nach einer zuvor definierten Methodik gemessen, beurteilt und dokumentiert. Am Ende eines „Business Case“ entsteht eine mit Informationen begründete Aussage über die Rentabilität der Investition. Das Geschäftsszenario soll für jedes Projekt eines Unternehmens erstellt werden. Dabei soll mit diesem der Mitteleinsatz gegenüber den Führungskräften beziehungsweise der Geschäftsführung gerechtfertigt werden. Bestandteile des „Business Case“ sind somit rein monetäre Größen und nicht-monetäre Aspekte (meist Abwägungen „hinsichtlich Risikoadressierung und Strategieorientierung in Verbindung mit den jeweiligen Optionen und deren wirtschaftlicher Vorteilhaftigkeit“⁹³). Es entsteht eine ganzheitliche Dokumentation aller entscheidungsrelevanter Sachverhalte: „Ein Business Case fasst alle entscheidungsrelevanten Aspekte eines geplanten Vorhabens mit dem Ziel zusammen, die wirtschaftliche Vorteilhaftigkeit und strategische Konformität des Gesamtprojekts aufzuzeigen und eine abschließende Management-Entscheidung über dessen Ausführung zu ermöglichen.“⁹⁴ Abzugrenzen ist dieser Begriff von dem „Business Plan“: Er

⁹²Herman und Siegelaub 2009, S. 4.

⁹³Brugger 2009, S. 12.

⁹⁴Brugger 2009, S. 13.

basiert auf der Gesamtbetrachtung einer organisatorischen Einheit bis zur Ebene des Gesamtunternehmens. Der „Business Plan“ erstellt ein Gesamtbild und ist nicht auf einzelne Projekte und Investitionsentscheidungen fokussiert.

Im Bereich der Investitionen gibt es zwei Entscheidungspfade: die Durchführungsentscheidung (absolute Vorteilhaftigkeit) und die Auswahlentscheidung (relative Vorteilhaftigkeit).⁹⁵ Die Differenzierung beider Möglichkeiten ist durch die Menge an zu bewertender Investition gegeben: Bei einer Investition werden die absolute (ist diese wirtschaftlich?) und bei mehreren die relative Vorteilhaftigkeit (welche ist wirtschaftlicher?) bewertet. Die Grundannahme der Investitionen lautet immer: Der Nutzen muss größer sein als die Kosten. Wenn die Kosten den Nutzen übersteigen, gibt es drei Bewertungsmöglichkeiten. Die Investition ist entweder aussichtsreich oder aussichtslos, doch andere Gründe sprechen dafür, oder aussichtslos. Aussichtsreiche Projekte sollten an die Erkenntnisse des Geschäftsszenario angepasst werden. Aussichtslose Projekte mit anderen Gründen müssen genauestens untersucht werden, um eine Entscheidung über die Realisation des Projekts zu treffen. Aussichtslose Projekte werden abgelehnt. Um den Nutzen der Erstellung eines „Business Case“ zu unterstreichen, sind folgende Vorteile zu nennen: Er erhöht erstens die Entscheidungssicherheit; schafft zweitens Entscheidungsspielraum, Übersicht und Transparenz, Verbindlichkeit, Klarheit, Nachvollziehbarkeit, und Vergleichbarkeit; er unterstützt drittens die „Controlling“-Division.⁹⁶

Mit dem „Business Case“ kann die Informatik ihren Wertschöpfungsbeitrag am Unternehmen beweisen. Aus der Überlegung heraus, die Informatik eines Unternehmens effektiv und effizient zu gestalten, ist die Betrachtung eines Geschäftsszenario von großer Bedeutung. Die Einordnung des Unternehmenszweckes ist bei der Erstellung des „Business Case“ wichtig. Die Informatik kann auf zwei Arten dem Unternehmenszweck dienen: Sie generiert Wert („value creation“) oder sie beschützt Wert („value protection“). So hat die Art des Dienstes direkte Auswirkungen auf den „Business Case“-Fokus. Bei der Wertsicherung wird ein Kostenvergleich in der Wirtschaftlichkeitsanalyse durchgeführt; die Wertgenerierung hingegen bedingt einen Kosten-Nutzen-Vergleich. Die Wertsicherung ist aus Sicht der Informatik für ein Unternehmen eine zwingende Aktivität. Problematisch ist es, da die Kunden (meist intern) keinen unmittelbaren Wertschöpfungscharakter erkennen und deswegen diese Maßnahmen meist nicht hoch priorisiert sind, jedoch erheblichen Einfluss auf die Geschäftstätigkeit eines Unternehmens haben.⁹⁷ Im Anhang B.1 auf Seite XXXIV ist ein stark vereinfachtes Flussdiagramm dargestellt, das die Entscheidungspfade für und gegen die Erstellung eines Geschäftsszenarios illustriert. Dieses kann benutzt werden, um eine schnelle Entscheidung zu erhalten, dennoch ist der eigentliche Prozess komplizierter – wie in der Abbildung

⁹⁵vgl. Brugger 2009, S. 14.

⁹⁶vgl. Brugger 2009, S. 17.

⁹⁷vgl. Brugger 2009, S. 27.

B.1 auf Seite XXXIV dargestellt. Beeinflusst wird dieser nicht nur durch staatliche Verordnungen und Gesetze, sondern auch durch innerbetriebliche Vorschriften.

Ein „Business Case“ kann intern oder extern erstellt werden. Es sind noch weitere Kombinationsmöglichkeiten denkbar, die in der Praxis jedoch kaum eine Rolle spielen.⁹⁸ Es gibt für beide Möglichkeiten, intern oder extern erstellt, Vor- und Nachteile, die im Anhang B.2 auf Seite XXXVI abgebildet sind. Die beteiligten Einheiten des Unternehmens entstammen der Informatik, einer „Business Unit“, der Finanzabteilung (meist „Controlling“) und der Personalabteilung. Entscheidungen, die das Projekt und somit das Geschäftsszenario betreffen, werden durch die höheren Führungsebenen in Verbindung mit den projektanfordernden Bereichen getroffen. Die Erstellung teilt sich in drei Ebenen auf: Initialisierung („Business Case Definition“), Entwicklung („Business Case Development“) und Prüfung („Business Case Quality Check“).⁹⁹ Während der Initialisierungsphase werden die Teams definiert, eine Eingrenzung der beteiligten Abteilungen vereinbart, die Kriterien bzw. Parameter für die Wirtschaftlichkeitsrechnung festgelegt und die Kalkulationsmethoden für die Ermittlung der Kennzahlen gewählt. In der Entwicklungsphase werden folgende Arbeitsschritte durchgeführt: Projektplanung/Systemkonzeption, Erhebung und Analyse der Kosten/des Nutzens, Aufbau des Wirtschaftlichkeitsmodells sowie Auswertung der Ergebnisse, die eine Sensitivitäts- (Versuch, die optimale Lösung weiter zu verbessern), eine Risiko- und Strategieanalyse enthält und die Zusammenfassung für die Führungsebene. Im Anhang B.2 auf Seite XXXVI ist eine Abbildung mit der chronologischen Anordnung der Arbeitsschritte zu sehen, die nochmals auf die Abhängigkeit der Schritte hinweist. Die letzte Phase beschäftigt sich mit der Qualitätssicherung der gewonnenen Erkenntnisse, bei der eine Validierung der Annahmen, die Prüfung der eingegebenen Daten und eine Abstimmung mit anderen Projekten durchgeführt werden.

4.2 Geschäftsprozess: „Release“ von neuen Anwendungsversionen

Der Geschäftsprozess „Release“ beinhaltet den „Deployment“-Prozess, deswegen wird in der Geschäftsprozessanalyse der „Release“-Prozess als Ganzes betrachtet, um den wirtschaftlichen Effekt der Veränderung des „Deployments“ zu untersuchen. Es wird zuerst der aktuelle „Release“-Prozess ohne den Container-„Deployment“-Prozess betrachtet.

Der aktuelle „Release“-Prozess folgt in den Entwicklungsarbeiten dem Vorgehensmodell „Wasserfall“. Dies ist ein iteratives Vorgehensmodell zur Anwendungsentwicklung,

⁹⁸vgl. Brugger 2009, S. 33.

⁹⁹vgl. Brugger 2009, S. 41-42.

dabei teilt sich das streng Modell in Konzeption und Umsetzung auf.¹⁰⁰ Die SVI hat ein Meilenstein-Konzept für das „Release“ entwickelt, der das genaue Vorgehen dieses Geschäftsvorfalls beschreibt. Dieser definiert Aufgaben und Tätigkeiten, die zu einer Folge kombiniert werden.

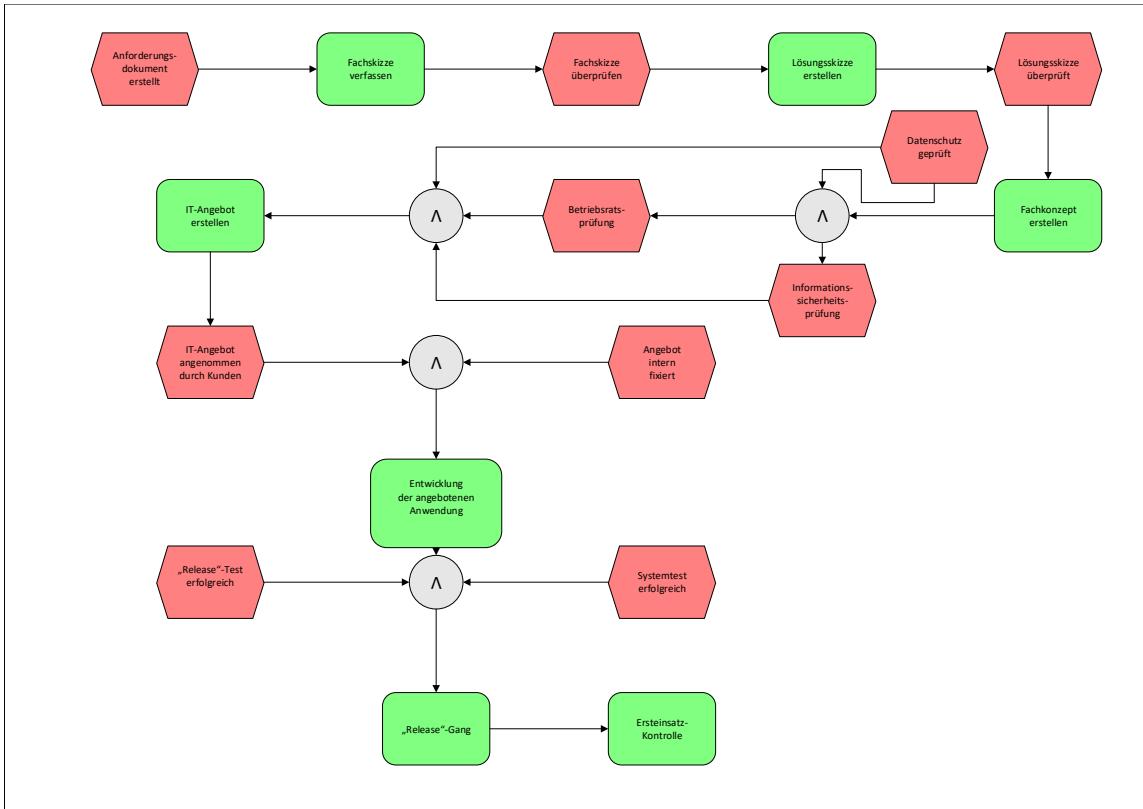


Abbildung 4.1: EPK zum Geschäftsvorfall „Release“

Quelle: in Anlehnung an unternehmensinterne Dokumente

unternehmensintern

Die Abbildung 4.1 zeigt die abgewandelte EPK des Geschäftsprozesses: Die rot gefärbten Formen sind Ereignisse, die grünen Funktionen/Arbeitspakete und das grau gefärbte Symbol beschreibt eine logische „AND“-Verknüpfung. Dieser Prozess hat verschiedene Prüfstellen, die die Qualität der erstellten Dokumente überprüfen. Erst ab der Funktion „Entwicklung der angebotenen Anwendung“ werden Anwendungen programmiert. Davor sind alle Aktivitäten zur Erstellung, Prüfung und Dokumentation der zu entwickelnden Anwendung durchzuführen. Diese sind mit erheblichen Aufwand verbunden: so muss die Kundin zusammen mit den Kundenmanagerinnen ein erstes Konzept entwickeln und beschreiben was das „statement of needs“ ist. Dieses wird in den nächsten Schritten zu einem Anforderungsdokument überarbeitet, dabei wer-

¹⁰⁰vgl. Freund und Rücker 2017, S. 24.

den auch mögliche Lösungsvorschläge berücksichtigt. Schließlich ist das Fachkonzepte und die Lösungsskizze erstellt. Die Kundenmanagerin kreiert ein IT-Angebot, dass der Kundin vorgelegt wird. Diese beiden Personenkreise verhandeln dann über die Kosten und Nutzen des Vorhabens. Sobald die Kundin zustimmt und alle anderen Zustimmungen (Betriebsrat, IT-Sicherheit und Datenschutz) eingeholt sind, startet die Entwicklung der beschriebenen Funktion. Nun folgend Tests in verschiedenen Umgebungen und dann die Freigabe für die Produktivsetzung der neuen Funktion/Anwendung. Danach prüft die Kundin, ob alle vorgesehenen Funktionen nach ihren Vorstellungen umgesetzt wurden. Dieses Vorgehen spiegelt einen linearen Vorgang wieder, der sich schwer an sich ständig veränderten Anforderungen anpassen kann. Ist das Fachkonzept freigegeben, kann an diesem nichts mehr geändert werden. Jedoch verändern sich die Anforderungen immer schneller und die Software hat einen immer kürzer andauern den Lebenszyklus innerhalb der SV. Die Digitalisierung wird in der SV-Strategie als zentraler Bestandteil beschrieben.¹⁰¹ Somit wird die Forderung nach einer Anpassung des „Release“-Prozesses immer stärker.

Durch die Veränderung des „Deployment“-Prozess ändert sich der gesamte „Release“-Prozess ebenfalls: So müssen die Planung und die Nicht-Anpassbarkeit des Fachkonzeptes überdacht werden. Außerdem ist der aktuelle Prozess, wie in Abbildung 4.1 auf der vorherigen Seite dargestellt, so mit den starren Regeln nicht mehr möglich. In einer Übergangsphase, bis ein neuer Geschäftsprozess entwickelt wurde, kann dieser aktuelle in einer abgewandelten Form benutzt werden. Diese Form müsste eine Änderung der Anforderungen und somit eine Änderung des Fachkonzeptes während der Entwicklung neuer Anwendungen zulassen. Dies kann zu inkonsistenten Daten in der Dokumentation führen und deswegen mit Vorsicht zu betrachten. Des Weiteren könnte eine Art Kategorisierung für die Anforderungen gemacht werden, d. h. zwingend umzusetzende Anforderung der Anwendungen wären ersichtlich. So könnten die Fachkonzepte in mindestens zu erfüllende und optimale Anteile unterteilt werden.

Die Container-Anwendungen unterstützen die Schnelligkeit der Entwicklung, da sie direkt mit einem entsprechenden „Deployment“-Prozess produktiv gehen können. Ist die Teststruktur automatisiert und die Dokumentationspflicht erfüllt, kann der Aufwand und damit die Kosten für die Entwicklung einer Funktion einer Anwendung beziehungsweise die Entwicklung einer kompletten Anwendung reduziert werden. Dies folgt aus der Tatsache, dass weniger Arbeitsstunden in die Betreuung, Entwicklung, Kontrolle und Dokumentation investiert werden müssen. Bleibt der Preis der Anwendung gleich, erhöht sich trotzdem der erzielte Gewinn, da die Kosten sich reduzieren. Natürlich sind initial die Kosten höher, da die Entwicklung eines neuen Prozess sehr viel Aufwand darstellt.

Momentan ist die SVI in einer Übergangsphase, in der sie den Geschäftsprozess 4.1 auf der vorherigen Seite weiterhin nutzt. Jedoch wurden hier die oben genannten

¹⁰¹vgl. SV SparkassenVersicherung 2019.

Veränderungen implementiert. Die illustrierte Weiterentwicklung des „Deployment“-Prozess (siehe Kapitel 3 auf Seite 9) ist eine Möglichkeit die Kosten durch Reduzierung des Betreuungsaufwandes in der Abteilung IE2 zu senken.

4.3 „Business Case“: „Deployment“ einer Container-Anwendung

Die SV möchte verschiedene Prozesse dunkel verarbeiten, d. h. es soll keine/n Sachbearbeiter/in Arbeit mit der Erledigung dieser haben. Die Prozesse, die unter dem Namen „Meine SV-Online Services“ bekannt sind, sind auf der Webseite der SV¹⁰² zu finden. Dort können Kundinnen verschiedene Aufträge selbst durchführen, wie eine Rechnungs-/Policenkopie anfordern, ihre persönliche Daten ändern, einen Schaden melden, eine bestehende KFZ-Haftpflichtversicherung anpassen und eine Haftpflicht/Hausrat abschließen. Diese Prozesse sollten mittels BPMN abgebildet werden. Ein „statement of needs“ der SV ist es, diese Prozesse selbstständig zu entwickeln. Durch diesen Wunsch entwickelte die SVI in Zusammenarbeit mit der SV eine Lösung. Es wurde ein Produktvergleich verschiedener Software-Hersteller durchgeführt. Schließlich hat sich die SV in Zusammenarbeit mit der SVI für die Software CAMUNDA entschieden. Der Beschaffungsprozess wurde prozess-konform durchgeführt und die Entwicklungsarbeiten im Projektteam begannen. Die Anforderungen der Software CAMUNDA beschrieben u. a., dass eine Container-Umgebung für die Verteilung dieser benötigt wird. Das Problem ist, dies war bei der Entscheidung zum Kauf der Software nicht berücksichtigt worden. Daraus resultiert die Anforderung, dass das „Deployment“-Team eine Lösung für die Verteilung von Container- und damit von CAMUNDA-basierten Anwendungen erforscht und implementiert. Daraus folgt ein gemeinsames Projekt der Infrastruktur- und Betriebsabteilungen der SVI: hier soll ein Software-Produkt gefunden werden, das revisions-konform und betriebstauglich ist. Die Kriterien der Revision werden von der Bundesanstalt für Finanzdienstleistungsaufsicht (BaFin), sowie durch juristische Vorschriften und Gesetze bestimmt. Die Betriebstauglichkeit wird durch Anforderungen der jeweiligen Abteilungen beschrieben. Aus diesen Kriterien folgen Akzeptanzbestimmungen für das Software-Produkt, wie z. B. die Multi-Mandaten-Fähigkeit, die Integrationsbereitschaft zu „Cloud“-Plattform (AMAZON WEBSERVICES, MICROSOFT AZURE), die Isolationsfähigkeit der kompletten Plattform im Rechenzentrum, Verschlüsselung der Kommunikation und weitere. Mit Hilfe dieser Forderungen kann ein Produktvergleich durchgeführt werden zwischen den Lösungen von RED HAT (OPENSHIFT) und SUSE (CAAS). Die Ergebnisse zeigten, dass der Hersteller SUSE zwei Produkte braucht, um die gegebenen Anforderungen zu erreichen. Im Gegensatz dazu erreicht RED HAT mit einem Produkt,

¹⁰²unter „Meine SV-Online Services“

OPENSHIFT, alle Anforderungen der SVI. Bei Betrachtung der zwei Produktlösungen sind keine nennenswerten Unterschiede zu erkennen. Die beiden Lösungen unterscheiden sich in den Kosten und, dass SUSE zwei Produkte für die Lösung benötigt. Die Kosten sind nachfolgend in der Tabelle 4.1 dargestellt. Schließlich wurde bedingt durch die lange Markterfahrung und die Nutzung durch andere Versicherungskonzerne die Lösung von RED HAT eingekauft. Die Lizenzkosten sind unter gewissen Annahmen getroffen, wie die Preise sind Listenpreise, die Berechnungsbasis sind „Cores“ in einer Virtualisierungsumgebung u. Ä. Weitere Rechenzentrum-spezifische Details, wie die Ausstattung der Virtualisierungsserver und das Ausfallsicherungskonzept der SVI, sind dementsprechend auch Annahmen zur Berechnung der Lizenzkosten.

Produkt	„Support“-Plan	Kosten
OPENSHIFT	Standard	80 „Cores“ \cong 86.400 €
		160 „Cores“ \cong 172.800 €
CAAS&CAP	Standard	80 „Cores“ \cong 80.000 €
		80 „Cores“ \cong 160.000 €
OPENSHIFT	Premium	80 „Cores“ \cong 128.000 €
		160 „Cores“ \cong 256.000 €
CAAS&CAP	Premium	80 „Cores“ \cong 111.200 €
		80 „Cores“ \cong 222.400 €

Tabelle 4.1: Vergleich der jährlichen Lizenzkosten

Durch Entscheidung der betreffenden Führungskräfte in Zusammenarbeit mit den Projektbeteiligten wird das Produkt OPENSHIFT der Produktlösung von SUSE bevorzugt. Auch wenn, die Lizenzkosten und der initiale Einführungsaufwand höher sind als bei der SUSE-Lösung. Gemäß des oben genannten Vorgehens eines „Business case“ (vgl. dazu Kapitel 4.1 auf Seite 40), muss bei einer Investition der Nutzen immer größer sein als die Kosten. Wenn das nicht gegeben ist, muss eine Abwägung getroffen werden, ob es trotzdem wirtschaftlich sinnvoll ist diese Investition zu tätigen. Hier sind die Akzeptanzkriterien nochmals zu überprüfen: RED HAT OPENSHIFT ist bei der Versicherungskammer Bayern (VKB) im Einsatz, die Provinzial NordWest (PNW) führt diese Anwendung aktuelle ein. Somit haben zwei andere öffentlich-rechtliche Versicherer diese Software im Einsatz. Auch die lange Markterfahrung von RED HAT im Bereich Finanzen und Automatisation sprechen für diesen Konzern. Im Gegensatz dazu ist die Produktlösung von SUSE noch nicht lange auf dem Markt (seit Herbst 2017) und die Lösung besteht zwei Produkten. Die Container-Anwendungen werden auf der Plattform OPENSHIFT von RED HAT betrieben.

Die vollständige „Business case“-Analyse wird teilweise durch die Projektcontrollgremien implementiert, so wird ein Kosten-Nutzen-Vergleich und eine Investitionsrechnungen durchgeführt. Jedoch wurde das Vorgehen auf die SVI adaptiert. Dies ist der Unternehmensstruktur der SV geschuldet: Die SV ist unterteilt in sogenannte Ressorts. Die SVI als Informatik-Dienstleiterin und vollständige Tochtergesellschaft ist dem Ressort drei „Leben/IT“ unter Dr. Wittmann zugeordnet. Damit sind viele Tätigkeiten im „Controlling“ Aufgabe der SV, so auch die Verantwortung, wie Entscheidungen getroffen werden.

4.4 Ergebnis der Forschungsfrage zwei

Die Ergebnisse der Forschungsfrage zwei – Welche wirtschaftlichen Vorteile hat der Einsatz von Container auf den Prozess des automatisierten „Deployments“? – werden nachfolgend kurz zusammengefasst. Eine kritische Betrachtung des gesamten Ergebnisses der Bachelorarbeit ist im Epilog (siehe Kapitel 6 auf Seite 58) zu finden.

„Business Case“-Methodik Die Methodik des „Business case“ konnte aus verschiedenen Gründen nicht komplett umgesetzt werden: So sind nicht genug Daten zur finanziellen Bewertung von CAMUNDA und einen Vergleichsprodukt vorhanden. Die Forderung Container-Anwendungen einzusetzen ist keine bewusste Entscheidung gewesen, sondern eine Konsequenz aus dem Einkauf der Software CAMUNDA – gewissermaßen eine Nebenbindung, um diese einsetzen zu können. Aus dieser Nebenbedingung entstand die Anforderung eine Plattform für den Betrieb von Container-Anwendungen bereitzustellen. Mit dieser Anforderung ist die Frage nach der wirtschaftlichen Sinnhaftigkeit von Container-Anwendung und deren Verteilung nicht zu stellen, sondern es ist eine wirtschaftlich sinnvolles (durch Akzeptanzkriterien messbares) Software-Produkt zu suchen. Dabei ist der Produktvergleich eine akzeptierte Methode. Das Vorgehen des Produktvergleichs wurde über Anforderungen messbar gestaltet. Dieser Produktvergleich ist jedoch nicht Teil dieser Arbeit. Abschließend ist der „Business Case“ und damit die Forschungsfrage eins „Welche wirtschaftlichen Vorteile hat der Einsatz von Container auf den Prozess des automatisierten „Deployments“?“ nicht durch die oben genannte Methodik aufgrund der Datenlage der SV beziehungsweise SVI beantwortbar. Jedoch kann eine generelle Aussage getroffen werden (vgl. Kapitel 4.4 auf der nächsten Seite Wirtschaftliche Auswirkungen).

Auswirkungen der Entscheidung der SV Die SV hat weitreichende Entscheidung bezügliche der Infrastruktur gefällt. Daraus resultierten mehrere Anforderungen, die nicht direkt sichtbar und damit bewertbar sind. Dadurch muss sich die SVI an die

gegebenen Wünsche der SV anpassen und versuchen diese Umzusetzen. Eine wirtschaftliche Betrachtung im Sinne von „Ist es langfristig wirtschaftlich für die SVI Container-Anwendungen zu betreiben“ entfiel. Es ist abzusehen, dass die jetzigen Software-Produkte der SVI, die Bestandssysteme der SV, in ihrer heutigen Architektur nicht in eine Containerlandschaft vollumfänglich übertragbar sind, so die Einschätzung der Entwicklerinnen der SVI. Zumindest lassen sich die Vorteile einer Container-Anwendung nicht vollständig nutzen (vgl. Kapitel 3.1.3 auf Seite 15).

Wirtschaftliche Auswirkung der Container-Anwendungen Container-Anwendungen bieten wesentliche wirtschaftliche Vorteile, wenn die Prozess zur Verteilung der Anwendung automatisiert sind, d. h. diese können ohne zusätzlichen Arbeitsaufwand verteilt werden. Auch bieten diese Anwendung eine schnellere „time to market“-Rate, da während der Betriebszeiten eine neue Software-Version verteilt und auf Knopfdruck veröffentlicht werden kann. Somit können Aufwände und Kosten von Abend-, Nacht- und Wochenendtätigkeiten gespart werden, denn ohne Container-Anwendungen ist es nicht möglich neue Versionen zu veröffentlichen – die Systeme sind sonst nicht erreichbar für die Kundinnen. Des Weiteren verringert sich nach der Einführung der Prozesse der Weiterentwicklungsaufwand, denn solange keine fundamentalen Änderungen in der Architektur eines Containers vollzogen werden, ist der Verteilungsprozess generisch für alle Container anwendbar, egal welchen Inhalt der jeweilige Container enthält.

5 Welche besonderen sicherheitstechnischen Aspekte muss ein solcher Prozess im Bereich der Versicherung erfüllen?

Dieses Kapitel ...

Informations- und Kommunikationssysteme sind in der heutigen Gesellschaft von elementarer Bedeutung – sie spielen eine immer größer werdende Rolle. Der Innovationsgrad in der Informationstechnik ist konstant hoch und deswegen sind folgende Bereiche ständiger Weiterentwicklung unterworfen: steigende Vernetzung der Bevölkerung, IT-Verbreitung und Durchdringung, Verschwinden der Netzgrenzen, kürze Angriffszyklen auf wichtige Infrastruktur, höhere Interaktivität von Anwendungen und die Verantwortung der Benutzer eines IT-Systems.¹⁰³

5.1 Grundlagen: Sicherheitstechnische Anforderungen

Informationen sind elementarer Bestandteil der heutigen Welt und diese sind von sehr hohem Wert für Unternehmen, Behörden und Privatpersonen. Die meisten Geschäftsprozesse, die im heutigen Prozessablauf einer Organisation verankert sind, funktionieren nicht ohne IT-Unterstützung. Somit ist die Informationstechnologie zentraler Bestandteil jedes Unternehmens. Deswegen ist ein zuverlässiges System mit entsprechender Soft- und Hardware unerlässlich. Es muss darauf geachtet werden, dass die Informationen, die auf diesem System verteilt sind, ausreichend gut geschützt sind, damit es nicht zu einer Bedrohungslage kommt. Unzureichend geschützte Systeme stellen ein sehr hohes Risiko dar. „Dabei ist ein vernünftiger Informationsschutz ebenso wie eine Grundsicherung der IT schon mit verhältnismäßig geringen Mitteln zu erreichen. Die verarbeiteten Daten und Informationen müssen adäquat geschützt, Sicherheitsmaßnahmen sorgfältig geplant, umgesetzt und kontrolliert werden. Hierbei ist es aber wichtig, sich nicht nur auf die Sicherheit von IT-Systemen zu konzentrieren, da Informationssicherheit ganzheitlich betrachtet werden muss. Sie hängt auch stark von infrastrukturellen, organisatorischen und personellen Rahmenbedingungen ab.“¹⁰⁴ Die

¹⁰³vgl. Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 2f.

¹⁰⁴Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 1.

Mängel in der IT-Sicherheit führen meist zu folgenden drei Kategorien von Problemen¹⁰⁵:

- Verlust der Verfügbarkeit
- Verlust der Vertraulichkeit
- Verlust der Integrität

Der Verlust der Verfügbarkeit eines IT-Systems fällt in der Regel sofort auf, da meist Aufgaben ohne diese Informationen nicht weitergeführt werden können. Meist fällt dies in dem Verlust der Funktionen eines Systems auf. Die Vertraulichkeit von personenbezogenen Daten ist ein bestehendes Grundrecht jedes Bürgers beziehungsweise jedes Kunden. Dies ist in verschiedenen Gesetzen wie auch Verordnungen geregelt. Diese Daten müssen geschützt werden, da jedes Konkurrenzunternehmen Interesse an den Daten des Unternehmens hat. „Gefälschte oder verfälschte Daten können beispielsweise zu Fehlbuchungen, falschen Lieferungen oder fehlerhaften Produkten führen. Auch der Verlust der Authentizität (Echtheit und Überprüfbarkeit) hat, als ein Teilbereich der Integrität, eine hohe Bedeutung: Daten werden beispielsweise einer falschen Person zugeordnet. So können Zahlungsanweisungen oder Bestellungen zulasten einer dritten Person verarbeitet werden, ungesicherte digitale Willenserklärungen können falschen Personen zugerechnet werden, die digitale Identität wird gefälscht.“¹⁰⁶

Informationssicherheitsmanagementsystem (ISMS) Um ein ISMS besser verstehen zu können, ist es wichtig, die Normenreihe des ISO-27001-Standards zu kennen. So bietet die ISO-Norm 27000 einen Überblick über ein solches System und definiert Begrifflichkeiten. Die zentrale Norm ist die ISO 27001, die die ISMS-Anforderungen beschreibt.¹⁰⁷ Dieser Norm sind die ISO-Standards 27002-27005, ISO 27007 und ISO 27008 untergeordnet, welche verschiedene Detailfragen zu in ISO 27001 genannten Konzepten definieren. Die Normen entstanden dem britischen Institut für Standards, weswegen „[...] [es] gleichzeitig eine international anerkannte Zertifizierungsstelle für ISO 27001 [ist] und damit eine der Stellen, die befugt sind, Auditoren zu qualifizieren und einzusetzen, um die Übereinstimmung einer Organisation mit der ISO 27001 im Rahmen einer Zertifizierung zu überprüfen.“¹⁰⁸ Die ISO-Norm 27001 ist durch die abstrakte Beschreibung und ihren Aufbau auf jegliche Art von Organisationen (Behörden, Unternehmen, Vereine, Nichtregierungsorganisation (NGOs) usw.) anwendbar. Außerdem ist sie beliebig skalierbar und in jedem Land bzw. länderübergreifend

¹⁰⁵vgl. Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 1ff.

¹⁰⁶Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 1.

¹⁰⁷vgl. DIN Deutsches Institut für Normung 2020b.

¹⁰⁸vgl. Kersten et al. 2020, S. 2.

nutzbar.¹⁰⁹ Die ISO-Norm 27000 definiert: „Ein Informationssicherheitsmanagementsystem (ISMS) umfasst Politik, Verfahren, Richtlinien und damit verbundene Ressourcen und Tätigkeiten, die alle von einer Organisation gesteuert werden, um ihre Informationswerte zu schützen. Ein ISMS ist ein systematisches Modell für die Einführung, die Umsetzung, den Betrieb, die Überwachung, die Überprüfung, die Pflege und die Verbesserung der Informationssicherheit einer Organisation, um Geschäftsziele zu erreichen.“¹¹⁰

Die ISO-Norm 27001 definiert in Kapitel vier bis zehn Anforderungen an ein Management-System der Informationssicherheit.¹¹¹ „Als Management-System für ein Thema X bezeichnet man allgemein alles, was eingesetzt wird, um die wesentlichen Ziele für das Thema X zu ermitteln, diese Ziele zu erreichen und ihre Aufrechterhaltung zu überwachen.“¹¹² Nachfolgend sind die typischen Aktivitäten genannt¹¹³:

- Ziele in Form von Leitlinien zu formulieren,
- Risiken und Chancen für diese Ziele zu analysieren,
- Rollen bzw. Verantwortlichkeiten für bestimmte (Teil-)Ziele zu definieren,
- Methoden oder Verfahren zu deren Erreichung zu vermitteln,
- den vom Thema X Betroffenen besondere Regelwerke oder Richtlinien aufzugeben,
- Prozesse bzw. Abläufe und dafür erforderliche Maßnahmen zu planen und umzusetzen,
- Überprüfungen der Zielerreichung zu planen, durchzuführen und auszuwerten.

Ziel des ISMS und damit der ISO-Norm 27001 ist es, für möglichen Prozess/Aktivitäten der Informationssicherheit ein einheitliches, standardisiertes System zu gestalten. Damit werden Aufwand- und Kosteneinsparungen erzeugt und die Akzeptanz eines solchen Systems gesteigert. Beispielsweise implementiert die ISO-Norm 9001 ein Qualitätsmanagementsystem,¹¹⁴ wobei die Architekturen der beiden Systeme kompatibel sind. Das ISMS wird auf die gesamte Organisation angewendet. Die wichtigsten Aufgaben sind dabei die Formulierung von Sicherheitszielen, die Bestimmung des „Assets“¹¹⁵ die Risikobeurteilung und -behandlung und die kontinuierliche Verbesserung.

¹⁰⁹vgl. Kersten et al. 2020, S. 4.

¹¹⁰DIN Deutsches Institut für Normung 2019, S. 20.

¹¹¹vgl. DIN Deutsches Institut für Normung 2020b, S. 6-16.

¹¹²Kersten et al. 2020, S. 5.

¹¹³Kersten et al. 2020, S. 5.

¹¹⁴DIN Deutsches Institut für Normung 2020c.

¹¹⁵„Unter Assets wird alles verstanden, was für eine Organisation einen Wert darstellt. Dies können zunächst Grundstücke, Gebäude, Maschinen und Anlagen, Geschäftsprozesse sein – aber natürlich auch die sogenannten Information Assets (Informationswerte) wie Informationen/Daten, Systeme,

Die Sicherheitsziele beschreiben die in Kapitel 5.1 auf Seite 50 genannten drei Hauptziele der Informationssicherheit (Verfügbarkeit, Vertraulichkeit und Integrität). Das Kapitel der Leitlinien beschäftigt sich mit der Definition der Organisation, der Analyse und den Regeln auf verschiedenen Ebenen der Organisation. Der Prozess der kontinuierlichen Verbesserung implementiert das Modell des „Plan-Do-Check-Act“-Regelkreises.¹¹⁶ Eine akzeptierte Variante ist, den Regelkreis jährlich zu durchlaufen. Umso mehr Iterationen abgeschlossen sind, desto besser ist das ISMS.¹¹⁷ Der Anhang A der ISO-Norm 27001 definiert sogenannte „Controls“, die als Sicherheitsanforderung an die Organisation gestellt werden. Möchte die Organisation streng die ISO-Norm 27001 implementieren, so ist jede „Control“ (es gibt 114) für jedes „Asset“ aus der Inventarisierung umzusetzen. Um die Implementierung zu erleichtern, bietet die ISO-Norm 27002¹¹⁸ Beispiele. Des Weiteren kann der IT-Grundschutz-Katalog des Bundesamtes für Sicherheit in der Informationstechnik(BSI),¹¹⁹ sowie das Wissen externer Beraterinnen genutzt werden, um mit den organisationseigenen Mitarbeitenden Maßnahmen zu entwerfen. Im Anhang C.1 auf Seite XXXVIII ist eine Checkliste abgebildet, die die Vorarbeiten der ISMS-Einführung illustriert.

IT-Grundschutz-Katalog des Bundesamt(es) für Sicherheit in der Informationstechnik (BSI) Das IT-Grundschutz-Kompendium bildet mit den BSI-Standards 200-1, 200-2, 200-3 und dem „Leitfaden zur Basis-Absicherung“ eine umfassende Beschreibung von Methoden, Anforderungen und Gefährdungen für die IT-Sicherheit. Dabei richten sie sich an Behörden und kleine, mittelständische und große Unternehmen.¹²⁰ Das IT-Grundschutz-Kompendium stellt dabei das Nachschlagewerk dar. Die BSI-Standards beschreiben, ähnlich zum ISO-Standard 27001, Themen, die das ISMS betreffen. Der „Leitfaden zur Basis-Absicherung“ ist die minimale Form der Implementierung von Sicherheitsanforderungen. Dieser kann für kleine Unternehmen schon ausreichend sein.¹²¹

„Im IT-Grundschutz-Kompendium werden standardisierte Sicherheitsanforderungen für typische Geschäftsprozesse, Anwendungen, IT-Systeme, Kommunikationsverbindungen und Räume in einzelnen Bausteinen beschrieben“.¹²² Ziel dieses Schutzkompendiums ist es, einen, für die Institutionen angepassten Schutz zu erreichen. Das

Anwendungen, IT Services. Ergänzend kann man auch Soft Assets betrachten wie das Image oder die Kreditwürdigkeit einer Organisation.“ Quelle: Kersten et al. 2020, S. 8

¹¹⁶Eine Abbildung dieses ist im Anhang C.1 auf Seite XXXVII zu sehen.

¹¹⁷vgl. DIN Deutsches Institut für Normung 2020b, S. 16.

¹¹⁸vgl. Deutsches Institut für Normung, e.V. 2017.

¹¹⁹Es gibt eine Tabelle, die die Implementierungsbeispiele des IT-Grundschutz zu den „Controls“ der ISO 27001 zuordnet. Quelle: Bundesamt für Sicherheit in der Informationstechnik (BSI) 2018

¹²⁰vgl. Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 3.

¹²¹vgl. Bundesamt für Sicherheit in der Informationstechnik (BSI) 2017, S. 5.

¹²²Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 2.

Kompendium illustriert eine umfassende Methodik, die sich auf die organisatorische, personelle, infrastrukturelle und technische Sicherheit einer Institution bezieht. Es soll ein Sicherheitsniveau erreicht werden, das für die jeweilige Institution angemessen und mindestens ausreichend ist, um die relevanten Informationen zu schützen. Vorteil des Kompendiums ist das Baukastenprinzip, denn damit ist es möglich, sich leichter an die heterogene Umgebung der Informationstechnik anzupassen. Dies führt zu einer besser Planungsfähigkeit und Struktur der Maßnahmen.¹²³ Diese Bausteine bilden den Stand der Technik ab und können nach Bedarf kombiniert werden. Der besondere Vorteil dieses Prinzips ist die Reduzierung des Arbeitsaufwandes. Bei einer klassischen Risikoanalyse nach dem ISO-Standard 27001 u. a., wie im Kapitel 5.1 auf Seite 51 beschrieben, muss für jedes „Asset“ eine eigene Analyse durchgeführt werden: Dies entfällt, da das BSI diese im Vorfeld abgeschlossen hat und die Ergebnisse in der jeweiligen Dokumentation des Bausteins zur Verfügung stellt. „Bei der IT-Grundschutz-Methodik reduziert sich die Analyse auf einen Soll-Ist-Vergleich zwischen den im IT-Grundschutz-Kompendium empfohlenen und den bereits umgesetzten Sicherheitsanforderungen. Die noch offenen Anforderungen zeigen die Sicherheitsdefizite auf, die es zu beheben gilt.“¹²⁴ Des Weiteren muss nur bei hohem Schutzbedarf (bspw. Schutz von systemkritischer Infrastruktur) eine Risikoanalyse für jedes „Asset“ durchgeführt werden. Die Methodik der Risikoanalyse wird im BSI-Standard 200-3 „Risikoanalyse auf der Basis von IT-Grundschutz“ weiter beschrieben. Ist ein Unternehmen bestrebt, eine Zertifizierung nach ISO 27001 zu erhalten, muss es die Basis- und Standard-Anforderungen des IT-Grundschutz-Kompendiums erfüllen. Darüber hinaus gibt es Anforderungen für einen erhöhten Schutzbedarf, die vom BSI ausdrücklich empfohlen sind.¹²⁵

Um einen Informationsverbund¹²⁶ nach dem IT-Grundschutz abzusichern, wird dieses mit den vorhandenen Bausteinen des Kompendiums nachgebildet. Es werden während dieses Prozesses alle IT-Systeme, Anwendungen und Prozesse erfasst und nach ihrem Schutzbedürfnis kategorisiert. Aus dieser Analyse wird ein IT-Grundschutz-Modell erstellt. Die Auswahl der passenden Komponenten oder Bausteine wird durch das Schichtenmodell¹²⁷ (siehe Anhang C.2 auf Seite XXXIX) des IT-Grundschutz-Kompendiums vereinfacht. Um die Modellierung zu vereinfachen, werden die Bau-

¹²³vgl. Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 2.

¹²⁴Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 3.

¹²⁵vgl. Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 3.

¹²⁶„[...] ist die Gesamtheit von infrastrukturellen, organisatorischen, personellen und technischen Objekten zu verstehen, die der Aufgabenerfüllung in einem bestimmten Anwendungsbereich der Informationsverarbeitung dienen. Ein Informationsverbund kann dabei als Ausprägung die gesamte Institution oder auch einzelne Bereiche, die durch organisatorische Strukturen (z. B. Abteilungen) oder gemeinsame Geschäftsprozesse bzw. Anwendungen (z. B. Personalinformationssystem) gegliedert sind, umfassen.“ Quelle: Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 37

¹²⁷nicht zu verwechseln mit „Open Systems Interconnection“ (OSI)-Modell der Netzwerkprotokolle

steine jeder Schicht betrachtet, damit eine Entscheidung getroffen wird, in welchem Umfang diese zur Abbildung des Informationsverbundes nutzbar sind. Das Kompendium priorisiert die Bearbeitungsreihenfolge der Bausteine in drei Kategorien: „R1“, „R2“ und „R3“. „R1“-Bausteine sollten vorrangig eingesetzt werden, da sie das Fundament des effektiven Sicherheitsprozesses bilden. Danach folgen Bausteine der beiden anderen Kategorien.

5.2 Prozessbeschreibung: Beschaffung von „Open source“-Software

In der SVI gibt es, wie in den meisten anderen Unternehmen, eine prozessorientierte Vorgehensweise, um Software zu beschaffen. Die Beschaffung von Software orientiert sich an ITIL Version 4, d. h., formal ist die Beschaffung von Software mit Hilfe eines „service requests“¹²⁸ zu beantragen. Für die Verteilung der Anwendung müssen danach mehrere „changes“ eingereicht werden. Im weiteren Verlauf wird die „Open source“-Variante beleuchtet, da es sich bei den verwendeten Containern, die von DOCKER INC. angeboten werden, um diese Variante handelt. Definitionsgemäß muss „Open source“-Software laut Opensource.org 2020 folgende Kriterien erfüllen: „Free redistribution, source code, derived works, integrity of the author’s source Code, no discrimination against persons or groups, no discrimination against fields of endeavor, distribution of license, license must not be specific to a product, license must not restrict other software, license must be technology-neutral.“

Es gibt in der SVI drei Prozesse, die sich in zwei Aspekten unterscheiden: Die Kosten und die Anforderungen, die an einen Prozess gestellt werden. Folgende Anfragen gibt es: die Beschaffungsanfrage, die „Freeware“-Beschaffung und die juristische Prüfung von Vertragsdokumenten oder Sachverhalten. Die Beschaffungsanfrage wird bei kostenpflichtiger Software gestellt. Da es in diesem Kapitel um kostenlose Software geht, wird auf die weitere Ausführung dieser Anfrage verzichtet. Der Prozess „Freeware“-Beschaffung wird laut den Juristen der Abteilung IT-Einkauf und IT-Recht (IU11) kaum¹²⁹ verwendet, denn die Fachbereiche¹³⁰ (die IT-Abteilungen) arbeiten zum jetzigen Zeitpunkt an dem Prozess vorbei – sie übergehen diesen wissentlich. Folgende Probleme haben sich bei der Befragung der Fachbereiche herausgestellt: Die Anforderungen, die dieser Prozess an sie stellt, sind „nicht verhältnismäßig“ gegenüber dem

¹²⁸a request from a user or a user’s authorized representative that initiates a service action which has been agreed as a normal part of service delivery. Quelle: AXELOS Limited und Stationery Office (Great Britain) 2019, S. 195

¹²⁹ $n \leq 5$, $n \in \mathbb{N}_0$, gemessen p. a.

¹³⁰aus Sicht von IU11

Nutzen. Die Fachbereiche wissen nicht, dass es einen solchen Prozess gibt oder ignorieren diesen. Die Anforderungen/Kriterien, die die Abteilung IU11 festgelegt hat, sind folgende: Es muss eine Produktverantwortliche definiert werden, es muss eine Architekturfreigabe von den zuständigen „Enterprise“-Architekten beantragt werden und es muss der genaue angedachte Verwendungszweck der einzukaufenden „Freeware“- bzw. „Open source“-Software definiert werden. Um den Ablauf des Prozesses besser verstehen zu können, zeigt Abbildung 5.1 ein adaptiertes Sequenzdiagramm, dass den Fokus auf die Interaktion zwischen einzelnen Abteilungen legt.

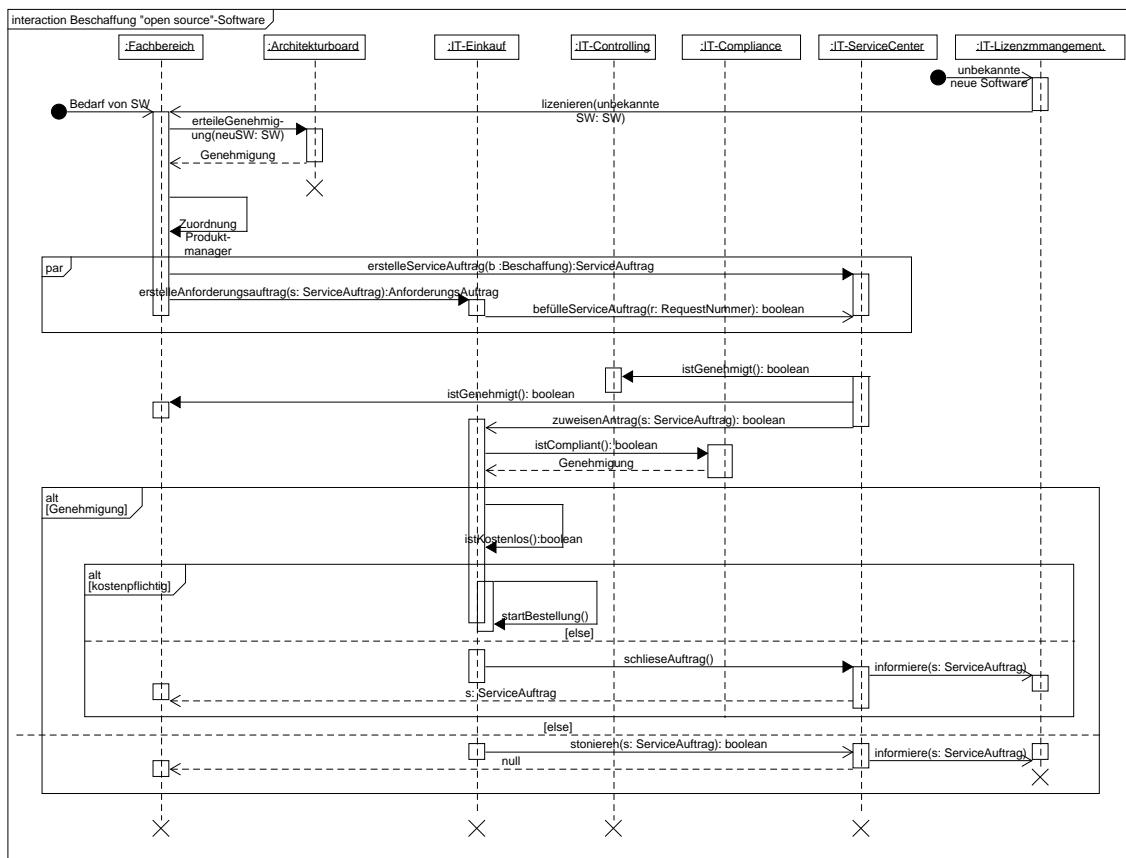


Abbildung 5.1: (adaptiertes) Sequenzdiagramm zur Beschaffung von „Open source“-Software

Quelle: in Anlehnung an unternehmensinterne Prozessdokumentation

unternehmensintern

Die oben genannten Hürden, aus Sicht der IT-Abteilung, erfüllen nicht die Kosten-Nutzen-Konformität. Aus rechtlicher Sicht ist das jedoch ein sehr hoch zu bewertendes Risiko, da es zu unmittelbaren juristischen Konsequenzen führen kann. Deswegen nutzt die IT-Abteilung meist den rechtlichen Prozess (juristische Prüfung von Ver-

tragsdokumenten oder Sachverhalten), da dieser nicht die oben genannten Hürden enthält. Bei diesem wird der Verwendungszweck der Software erfragt und die gelgenden Lizenzbedingungen durch IU11 geprüft. Jedoch ist davon auszugehen, dass eine offizielle Beschaffungsanfrage bei „Open source“-Software nur in wenigen¹³¹ Fällen gestellt wird. Begründet durch die Administrator-Berechtigung, die es Benutzern erlaubt, auch ohne Restriktionen alles auf ihrem Computer zu installieren, kann keine numerische Aussage über die Dunkelziffer getroffen werden. Es bleibt nur die Hypothese der IT-Juristen der Abteilung IU11, die weder falsifizierbar noch validierbar ist.

Ist die Software in der AWL implementiert, gibt es noch eine Anwendung, NEXUS LIFECYCLE von SONATYPE, die auf eventuelle Schwachstellen dieser benutzten Software prüft. NEXUS LIFECYCLE ist eine Hilfsanwendung, die u. a. auch von CREDITREDFORM verwendet wird. Das Ziel dieses Produktes ist es, die gesamte Software-„Supply Chain“ kontinuierlich zu bereinigen und sicher zu halten.¹³² Aus dem Prüfbericht werden dann entsprechende Maßnahmen abgeleitet. Die erste ist die Software, in der die Schwachstelle gefunden wurde, als unsicher zu markieren und danach zu sperren. Die Entwicklungsabteilung muss versuchen, die Schwachstellen zu beseitigen. Problematisch ist es, wenn diese ignoriert werden. In letzter Konsequenz werden der Betrieb und die Verteilung der Anwendung gestoppt. Dies führt zu massiven Problemen in der Produktion und somit verringert sich die vertragliche, mit dem Kunden vereinbarte, Verfügbarkeit der Systeme.

5.3 Konzept zur Implementierung der Sicherheitsanforderungen

5.4 Ergebnis der Forschungsfrage drei

¹³¹ $n \leq 10, n \in \mathbb{N}_0$, gemessen p. a.

¹³²vgl. Sonatype Inc. 2020.

6 Epilog

Zusammenfassung der Erkenntnisse

Fazit

Ausblick

Literaturverzeichnis

Atomic Requirement Download (19. Aug. 2019). URL: <https://www.volere.org/atomic-requirement-download/>.

AXELOS Limited und Stationery Office (Great Britain) (2019). *ITIL® Foundation, ITIL 4 edition*. v4. OCLC: 1122856407. Norwich: TSO (The Stationery Office). ISBN: 978-0-11-331607-6. URL: https://www.axelos.com/getmedia/5896d51f-ab6c-4843-992b-4f045eab0875/ITIL-4-Foundation-glossary_v0_22.aspx (besucht am 10.03.2020).

Bernstein, David (Sep. 2014). „Containers and Cloud: From LXC to Docker to Kubernetes“. In: *IEEE Cloud Computing* 1.3, S. 81–84. ISSN: 2372-2568. DOI: 10.1109/MCC.2014.51.

Brewer, Eric (Feb. 2012). „CAP twelve years later: How the "rules" have changed“. In: *Computer* 45.2, S. 23–29. ISSN: 0018-9162, 1558-0814. DOI: 10.1109/MC.2012.37. URL: <https://ieeexplore.ieee.org/document/6133253/> (besucht am 08.04.2020).

Broadcom Inc. (2020). *Automic Automation*. Library Catalog: www.broadcom.com. URL: <https://www.broadcom.com/products/software/automation/digital-business-automation/automic-automation> (besucht am 16.04.2020).

Brugger, Ralph (2009). *Der IT Business Case*. Xpert.press. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-540-93857-6 978-3-540-93858-3. DOI: 10.1007/978-3-540-93858-3. URL: <http://link.springer.com/10.1007/978-3-540-93858-3> (besucht am 19.03.2020).

Bundesamt für Sicherheit in der Informationstechnik (BSI) (20. Okt. 2017). *Leitfaden zur Basis-Absicherung nach IT-Grundschutz: In 3 Schritten zur Informations-sicherheit*. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Broschueren/Leitfaden_zur_Basis-Absicherung.html (besucht am 27.03.2020).

Bundesamt für Sicherheit in der Informationstechnik (BSI) (20. Apr. 2018). *Zuordnungstabelle ISO zum modernisierten IT-Grundschutz*. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Kompendium/Zuordnung_ISO_und_modernisierter_IT_Grundschutz.pdf;jsessionid=5ADB145EF2581C0DD41F6CBA3A702_cid360?__blob=publicationFile&v=9 (besucht am 27.03.2020).

- Bundesamt für Sicherheit in der Informationstechnik (BSI) (2020). *IT-Grundsatz-Kompendium*. 2020. Aufl. OCLC: 1027470677. Bundesanzeiger Verlag GmbH. 816 S. ISBN: 978-3-8462-0906-6. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundsatz/Kompendium/IT_Grundsatz_Kompendium_Edition2020.pdf?__blob=publicationFile&v=6 (besucht am 09.03.2020).
- Caban, William (2019). *Architecting and Operating OpenShift Clusters OpenShift for Infrastructure and Operations Teams*. OCLC: 1138978697. Berkeley, CA: Apress : Imprint: Apress. ISBN: 978-1-4842-4985-7 978-1-4842-4984-0 978-1-4842-4986-4. URL: <https://link.springer.com/10.1007/978-1-4842-4985-7> (besucht am 24.02.2020).
- Camunda Services GmbH (2020). *Workflow und Decision Automation / Camunda BPM*. Camunda. URL: <https://camunda.com/de/> (besucht am 14.04.2020).
- Canonical Ltd. (2020). *Linux Containers*. Linux Container. URL: <https://linuxcontainers.org/> (besucht am 17.03.2020).
- Combe, Theo, Antony Martin und Roberto Di Pietro (Sep. 2016). „To Docker or Not to Docker: A Security Perspective“. In: *IEEE Cloud Computing* 3.5, S. 54–62. ISSN: 2372-2568. DOI: [10.1109/MCC.2016.100](https://doi.org/10.1109/MCC.2016.100).
- Dearle, Alan (Mai 2007). „Software Deployment, Past, Present and Future“. In: *Future of Software Engineering (FOSE '07)*. Future of Software Engineering (FOSE '07). Minneapolis, MN: IEEE, S. 269–284. ISBN: 978-0-7695-2829-8. DOI: [10.1109/FOSE.2007.20](https://doi.org/10.1109/FOSE.2007.20). URL: <https://ieeexplore.ieee.org/document/4221626/> (besucht am 18.03.2020).
- Dechange, André (2020). *Projektmanagement – Schnell erfasst*. Wirtschaft – Schnell erfasst. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-662-57666-3 978-3-662-57667-0. DOI: [10.1007/978-3-662-57667-0](https://doi.org/10.1007/978-3-662-57667-0). URL: [http://link.springer.com/10.1007/978-3-662-57667-0](https://link.springer.com/10.1007/978-3-662-57667-0) (besucht am 01.04.2020).
- Deutsches Institut für Normung, e.V. (10. Juni 2017). *Informationstechnik - Sicherheitsverfahren - Leitfaden für Informationssicherheitsmaßnahmen (ISO/IEC 27002:2013 einschließlich Cor 1:2014 und Cor 2:2015); Deutsche Fassung EN ISO/IEC 27002:2017*. URL: <https://perinorm-fr.redi-bw.de/perinorm/fulltext.ashx?fulltextid=ae4372165f8d4fa1bb8bff06b0923e46&userid=96f4b659-56a0-45ff-a5be-da1774bd04e8> (besucht am 27.03.2020).
- Dilbert on Kubernetes (11. Aug. 2017). URL: https://miro.medium.com/max/1024/1*RODEnf_7sjswuBHouioQFg.jpeg.

- DIN Deutsches Institut für Normung, e. V. (5. Juli 2019). *Informationstechnik -Sicherheitsverfahren -Informationssicherheits-Managementsysteme – Überblick und Terminologie (ISO/IEC 27000:2018); Deutsche und Englische Fassung prEN ISO/IEC 27000:2019*. URL: <https://perinorm-fr.redi-bw.de/perinorm/fulltext.ashx?fulltextid=3fb39d48521c4797907ba51ad7c443da&userid=96f4b659-56a0-45ff-a5be-da1774bd04e8> (besucht am 27.03.2020).
- DIN Deutsches Institut für Normung, e. V. (26. Feb. 2020a). *Informationstechnik - Cloud Computing - Übersicht und Vokabular (ISO/IEC 17788:2014)*. URL: <https://perinorm-fr.redi-bw.de/perinorm/fulltext.ashx?fulltextid=94c35b8fadfc4c51853726be&userid=96f4b659-56a0-45ff-a5be-da1774bd04e8>.
- DIN Deutsches Institut für Normung, e. V. (26. Feb. 2020b). *Informationstechnik - Sicherheitsverfahren - Informationssicherheitsmanagementsysteme - Anforderungen (ISO/IEC 27001:2013 einschließlich Cor 1:2014 und Cor 2:2015); Deutsche Fassung EN ISO/IEC 27001:2017*. URL: <https://perinorm-fr.redi-bw.de/perinorm/fulltext.ashx?fulltextid=b13c1f6be2f04f0298a6f7c96b1bbad1&userid=96f4b659-56a0-45ff-a5be-da1774bd04e8>.
- DIN Deutsches Institut für Normung, e. V. (26. Feb. 2020c). *Qualitätsmanagementsysteme - Grundlagen und Begriffe (ISO 9000:2015); Deutsche und Englische Fassung EN ISO 9000:2015*. URL: <https://perinorm-fr.redi-bw.de/perinorm/fulltext.ashx?fulltextid=230150cb1880449295f493e21a445a20&userid=96f4b659-56a0-45ff-a5be-da1774bd04e8>.
- Django Software Foundation (2020). *The Web framework for perfectionists with deadlines / Django*. URL: <https://www.djangoproject.com/> (besucht am 03.04.2020).
- Foster, Ian und Carl Kesselman, Hrsg. (1999). *The grid: blueprint for a new computing infrastructure*. San Francisco: Morgan Kaufmann Publishers. 677 S. ISBN: 978-1-55860-475-9.
- Freund, Jakob und Bernd Rücker (2017). *Praxishandbuch BPMN: mit Einführung in CMMN und DMN*. 5., aktualisierte Auflage. Type: Text (nur für elektronische Ressourcen). München: Hanser. ISBN: 3446450548 (Druck-Ausgabe). URL: <http://dx.doi.org/10.3139/9783446450783>.
- Gadatsch, Andreas (2010). *Grundkurs Geschäftsprozess-Management*. Wiesbaden: Vieweg+Teubner. ISBN: 978-3-8348-0762-5 978-3-8348-9346-8. DOI: 10.1007/978-3-8348-9346-8. URL: <http://link.springer.com/10.1007/978-3-8348-9346-8> (besucht am 20.03.2020).
- Gartner (2019). *Cloud Computing - Umsatz bis 2022*. Statista. Library Catalog: de.statista.com. URL: <https://de.statista.com/statistik/daten/studie/195760/umfrage/umsatz-mit-cloud-computing-weltweit/> (besucht am 13.03.2020).

- Google Ireland Limited (2020). *Container und ihre Vorteile*. Google Cloud. Library Catalog: cloud.google.com. URL: <https://cloud.google.com/containers?hl=de> (besucht am 21.03.2020).
- Google LLC (21. Apr. 2020a). *Google Scholar – Suche nach Hull*. URL: https://scholar.google.de/scholar?hl=de&as_sdt=0%2C5&q=hull+elizabeth+requirements+engineering&btnG=&oq=hull+elizabeth+requirements+engi.
- Google LLC (2020b). *Production-Grade Container Orchestration with K8s*. Library Catalog: kubernetes.io. URL: <https://kubernetes.io/> (besucht am 18.03.2020).
- Herman, B und J Siegelaub (2009). „Is this really worth the effort? The need for a business case“. In: *PMI Global Congress, Orlando, FL, October*. PMI Global Congress. Orlando, FL, USA: PMI. URL: <https://www.pmi.org/learning/library/need-business-case-6730> (besucht am 19.03.2020).
- Hill, Richard, Hrsg. (2013). *Guide to cloud computing: principles and practice*. Computer communications and networks. OCLC: ocn807043821. London ; New York: Springer. 278 S. ISBN: 978-1-4471-4602-5 978-1-4471-4603-2.
- Hull, Elizabeth, Ken Jackson und Jeremy Dick (2011). *Requirements engineering*. 3rd ed. London ; New York: Springer. 207 S. ISBN: 978-1-84996-404-3 978-1-84996-405-0.
- ICT/1 (26. Feb. 2020). *Information technology. Object Management Group Business Process Model and Notation*. URL: <https://www.omg.org/spec/BPMN/2.0/PDF>.
- IEEE (2005). „IEEE Standard for Application and Management of the Systems Engineering Process“. In: *IEEE Std 1220-2005 (Revision of IEEE Std 1220-1998)*, S. 1–96. DOI: [10.1109/IEEESTD.2005.96469](https://doi.org/10.1109/IEEESTD.2005.96469).
- „Cloud Computing: Deployment Models, Delivery Models, Risks and Research Challenges“ (2011). In: 2011 International Conference on Computer and Management (CAMAN). Hrsg. von Institute of Electrical and Electronics Engineers. OCLC: 838686677. Wuhan, China: IEEE, S. 1–4. ISBN: 978-1-4244-9283-1 978-1-4244-9282-4. DOI: [10.1109/CAMAN.2011.5778816](https://doi.org/10.1109/CAMAN.2011.5778816). URL: <https://ieeexplore.ieee.org/abstract/document/5778816>.
- International Organization for Standardization (2011). *ISO/IEC 25010:2011*. ISO. Library Catalog: www.iso.org. URL: <https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/03/57/35733.html> (besucht am 12.04.2020).
- ITCandor (2019). *Cloud Computing - Marktanteile der führenden Unternehmen 2019*. Statista. Library Catalog: de.statista.com. URL: <https://de.statista.com/statistik/daten/studie/150979/umfrage/marktanteile-der-fuehrenden-unternehmen-im-bereich-cloud-computing/> (besucht am 13.03.2020).

- Kersten, Heinrich et al. (2020). *IT-Sicherheitsmanagement nach der neuen ISO 27001: ISMS, Risiken, Kennziffern, Controls*. Edition <kes>. Wiesbaden: Springer Fachmedien Wiesbaden. ISBN: 978-3-658-27691-1 978-3-658-27692-8. DOI: 10.1007/978-3-658-27692-8. URL: <http://link.springer.com/10.1007/978-3-658-27692-8> (besucht am 26.03.2020).
- Kharb, Dr Latika (2016). „Automated Deployment of Software Containers Using Dockers“. In: 4.10, S. 3.
- Kumar, Vikas und R. Vidhyalakshmi (2018). *Reliability aspect of cloud computing environment*. Singapore: Springer. 170 S. ISBN: 9789811330230 9789811330223. URL: <https://doi.org/10.1007/978-981-13-3023-0>.
- Lukša, Marko (2018). *Kubernetes in Action: Anwendungen in Kubernetes-Clustern bereitstellen und verwalten*. OCLC: 1017485179. München: Hanser. 642 S. ISBN: 978-3-446-45510-8 978-3-446-45602-0.
- McCarthy, John (1983). *REMINISCENCES ON THE HISTORY OF TIME SHARING*. REMINISCENCES ON THE HISTORY OF TIME SHARING. URL: <http://www-formal.stanford.edu/jmc/history/timesharing/timesharing.html> (besucht am 13.03.2020).
- McCracken, Daniel D. und Edwin D. Reilly (2003). „Backus-Naur Form (BNF)“. In: *Encyclopedia of Computer Science*. GBR: John Wiley und Sons Ltd., S. 129–131. ISBN: 0-470-86412-5.
- Mell, Peter und Timothy Grance (28. Sep. 2011). *The NIST Definition of Cloud Computing*. Special Publication (NIST SP) 800-145. Gaithersburg, MD 20899-8930: National Institute of Standards und Technology. 7 S. URL: <https://doi.org/10.6028/NIST.SP.800-145> (besucht am 13.03.2020).
- Object Management Group (OMG) (1. Dez. 2011). *Business Process Model and Notation (BPMN)*. URL: <https://www.omg.org/spec/BPMN/2.0/PDF> (besucht am 23.03.2020).
- Object Management Group (OMG) (1. Dez. 2017). *Unified Modeling Language (UML)*. URL: <https://www.omg.org/spec/UML/2.5.1/PDF> (besucht am 06.04.2020).
- Opensource.org (2020). *The Open Source Definition*. The Open Source Definition. URL: <https://opensource.org/osd> (besucht am 10.03.2020).
- Pahl, Claus (Mai 2015). „Containerization and the PaaS Cloud“. In: *IEEE Cloud Comput.* 2.3, S. 24–31. ISSN: 2325-6095. DOI: 10.1109/MCC.2015.51. URL: <http://ieeexplore.ieee.org/document/7158965/> (besucht am 13.03.2020).
- Parkhill, Douglas Freeman (1966). *The Challenge of the computer utility: D.F. Parkhill*. OCLC: 460679364. Reading: Mass., London : Addison-Wesley Publishing C°.

- Partsch, Helmuth (2010). *Requirements-Engineering systematisch: Modellbildung für softwaregestützte Systeme*. 2., überarb. und erw. Aufl. eXamen.press. OCLC: 845656932. Berlin: Springer. 394 S. ISBN: 978-3-642-05358-0 978-3-642-05357-3. URL: <http://dx.doi.org/10.1007/978-3-642-05358-0>.
- Pawar, Kulwant S., Unny Menon und Johann C.K.H. Riedel (1. Jan. 1994). „Time to Market“. In: *Integrated Manufacturing Systems* 5.1. Publisher: MCB UP Ltd, S. 14–22. ISSN: 0957-6061. DOI: 10.1108/09576069410815765. URL: <https://doi.org/10.1108/09576069410815765> (besucht am 17.03.2020).
- Red Hat, Inc. (1. Mai 2019). *OKD Latest Documentation*. OKD Latest Documentation. URL: <https://docs.okd.io/latest/welcome/index.html> (besucht am 21.04.2020).
- Red Hat, Inc. (2020a). *CLI Operations / CLI Reference / OpenShift Enterprise 3.0*. URL: https://docs.openshift.com/enterprise/3.0/cli_reference/basic_cli_operations.html (besucht am 03.04.2020).
- Red Hat, Inc. (2020b). *OpenShift Container Platform by Red Hat, Built on Kubernetes*. OpenShift Container Platform by Red Hat, Built on Kubernetes. URL: <https://www.openshift.com/> (besucht am 11.03.2020).
- Red Hat, Inc. (2020c). *Red Hat – Wir entwickeln Open Source-Technologien für Unternehmen*. Red Hat – Wir entwickeln Open Source-Technologien für Unternehmen. URL: <https://www.redhat.com/de> (besucht am 11.03.2020).
- Reinheimer, Stefan und Springer Fachmedien Wiesbaden GmbH, Hrsg. (2018). *Cloud Computing: die Infrastruktur der Digitalisierung*. Edition HMD. OCLC: 1038769740. Wiesbaden: Springer Vieweg. 216 S. ISBN: 978-3-658-20966-7 978-3-658-20967-4. URL: <https://doi.org/10.1007/978-3-658-20967-4>.
- Rimal, Bhaskar Prasad et al. (März 2011). „Architectural Requirements for Cloud Computing Systems: An Enterprise Cloud Approach“. In: *J Grid Computing* 9.1, S. 3–26. ISSN: 1570-7873, 1572-9184. DOI: 10.1007/s10723-010-9171-y. URL: <http://link.springer.com/10.1007/s10723-010-9171-y> (besucht am 13.03.2020).
- Rupp, C. (3. Apr. 2020). *Formulierungsregel für die funktionalen Anforderungen*. URL: https://www.pst_ifi.lmu.de/Lehre/fruhere-semester/sose-2009/seprakt/Formulierungsregel_DE_Rupp_Schablone.pdf (besucht am 03.04.2020).
- Scheer, A.-W., M. Nüttgens und V. Zimmermann (1. Mai 1997). „Objektorientierte Ereignisgesteuerte Prozeßkette (oEPK) – Methoden und Anwendung“. In: *IWi-Hefte* 141, S. 29. URL: <https://www.uni-saarland.de/lehrstuhl/loos/publikationen/iwi-hefte.html> (besucht am 23.03.2020).

Sonatype Inc. (2020). *Nexus Lifecycle Product*. Nexus Lifecycle Product. URL: https://de.sonatype.com/product-nexus-lifecycle?utm_campaign=NVS&utm_source=ppc&utm_medium=adwords&ahcsource=paid&utm_term=%2Bnexus%20%2Blifecycle&hsa_tgt=kwd-437257894053&hsa_grp=90875397990&hsa_src=s&hsa_net=adwords&hsa_mt=b&hsa_ver=3&hsa_ad=406628330148&hsa_acc=2665806879&hsa_kw=%2Bnexus%20%2Blifecycle&hsa_cam=8625747087&gclid=EAIAIQobChMIGsvQt8mP6AIVh-h3Ch29zQJJEAYASAAEgK02fD_BwE (besucht am 10.03.2020).

Staud, Josef L. (2006). *Geschäftsprozessanalyse: ereignisgesteuerte Prozessketten und objektorientierte Geschäftsprozessmodellierung für betriebswirtschaftliche Standardsoftware*. 3. Aufl. OCLC: 180896033. Berlin: Springer. 538 S. ISBN: 978-3-540-24510-0. URL: <https://doi.org/10.1007/3-540-37976-2>.

SV SparkassenVersicherung (2019). *SV kompakt – Profil und Positionen*. Hrsg. von Sylvia Knittel. Unter Mitarb. von Stefanie Rösch. URL: https://www.sparkassenversicherung.de/export/sites/svag/_resources/download_galerien/die_sv/geschaeftsberichte/SV_Kompakt_2020.pdf (besucht am 15.04.2020).

The People of the GnuPG Project (7. Jan. 2020). *The GNU Privacy Guard*. Library Catalog: gnupg.org Publisher: The GnuPG Project. URL: <https://gnupg.org/> (besucht am 02.04.2020).

Volere Requirements Specification Template (19. Aug. 2019). URL: <https://www.volere.org/templates/volere-requirements-specification-template/>.

Weinhardt, Christof et al. (Okt. 2009). „Cloud Computing – A Classification, Business Models, and Research Directions“. In: *Bus. Inf. Syst. Eng.* 1.5, S. 391–399. ISSN: 1867-0202. DOI: 10.1007/s12599-009-0071-2. URL: <http://link.springer.com/10.1007/s12599-009-0071-2> (besucht am 13.03.2020).

A Ergänzungen zur Forschungsfrage eins

In diesem Teil des Anhangs sind Ergänzungen zur Forschungsfrage eins des Kapitels 3 auf Seite 9 beschrieben.

A.1 Anforderungsdokument

Ein Anforderungskatalog hat bestimmte Anforderungen, die an den Prozess gestellt werden. Neben der Forderung nach Einhaltung der Qualitätskriterien, definiert nach dem ISO-Standard 9000/9001, sind noch folgende Forderungen in der Literatur beschrieben:¹³³

- vollständig (inhaltlich – d. h., alle Anforderungen sind erfasst –, formal, Norm-konform)
- konsistent (keine Widersprüche zwischen den Bestandteilen des Dokuments, insbesondere keine Konflikte zwischen verschiedenen Anforderungen)
- lokal änderbar (Änderungen an einer Stelle sollten keine Einflüsse auf Konsistenz und Vollständigkeit des Gesamtdokuments haben)
- verfolgbar (ursprüngliche Stakeholderwünsche und Zusammenhänge zwischen Anforderungen sind leicht zu finden)
- klar strukturiert
- umfangsmäßig angemessen
- sortierbar/projizierbar (nach verschiedenen Kriterien, für verschiedene Stakeholder).

Die folgende Aufzählung beschreibt eine Vorlage für das Anforderungsdokument nach Quelle: Sie nutzt die Hilfsmittelsammlung „Volere“. Diese bietet im Themenbereich „requirements engineering“ kostenpflichtig Dokumentenvorlagen an. Die beiden Bekanntesten sind die hier gezeigte „Volere Requirements Specification Template“ und das kostenlose „Volere Atomic Requirement Template“, das umgangssprachlich „Snow Card“ genannt wird. Die „Snow Card“ (A.1 auf Seite XX) ist eine Karteikarte, die

¹³³sig. Partsch 2010, S. 34.

benutzt wird, um eine vollständige Aufnahme aller Informationen einer einzelnen Anforderung zu gewährleisten.¹³⁴



Abbildung A.1: Volere Snow Card
Quelle: *Atomic Requirement Download 2019*

Die folgende Liste wurde in Anlehnung an die Quelle *Volere Requirements Specification Template 2019* erstellt.

¹³⁴vgl. *Atomic Requirement Download 2019*.

- Projekt-Treiber
 - 1. Zweck des Projekts
 - 2. Auftraggeber, Kunde und andere Stakeholder
 - 3. Nutzer des Produkts
- Projekt-Randbedingungen
 - 1. Einschränkungen
 - 2. Namenskonventionen und Definitionen
 - 3. Relevante Fakten und Annahmen
- Funktionale Anforderungen
 - 1. Arbeitsrahmen
 - 2. Systemgrenzen
 - 3. Funktionale und Daten-Anforderungen
- Nicht-funktionale Anforderungen
 - 1. Look-and-Feel-Anforderungen
 - 2. Usability-Anforderungen
 - 3. Performanz-Anforderungen
 - 4. Operationale und Umfeld-Anforderungen
 - 5. Wartungs- und Unterstützungsanforderungen
 - 6. Sicherheitsanforderungen
 - 7. Kulturelle und politische Anforderungen
 - 8. Rechtliche Anforderungen
- Projekt-Aspekte
 - 1. Offene Punkte
 - 2. Standardlösungen
 - 3. Neu aufgetretene Probleme
 - 4. Installationsaufgaben
 - 5. Migrationstätigkeiten
 - 6. Risiken
 - 7. Kosten
 - 8. Nutzerdokumentation
 - 9. Zurückgestellte Anforderungen
 - 10. Lösungsideen

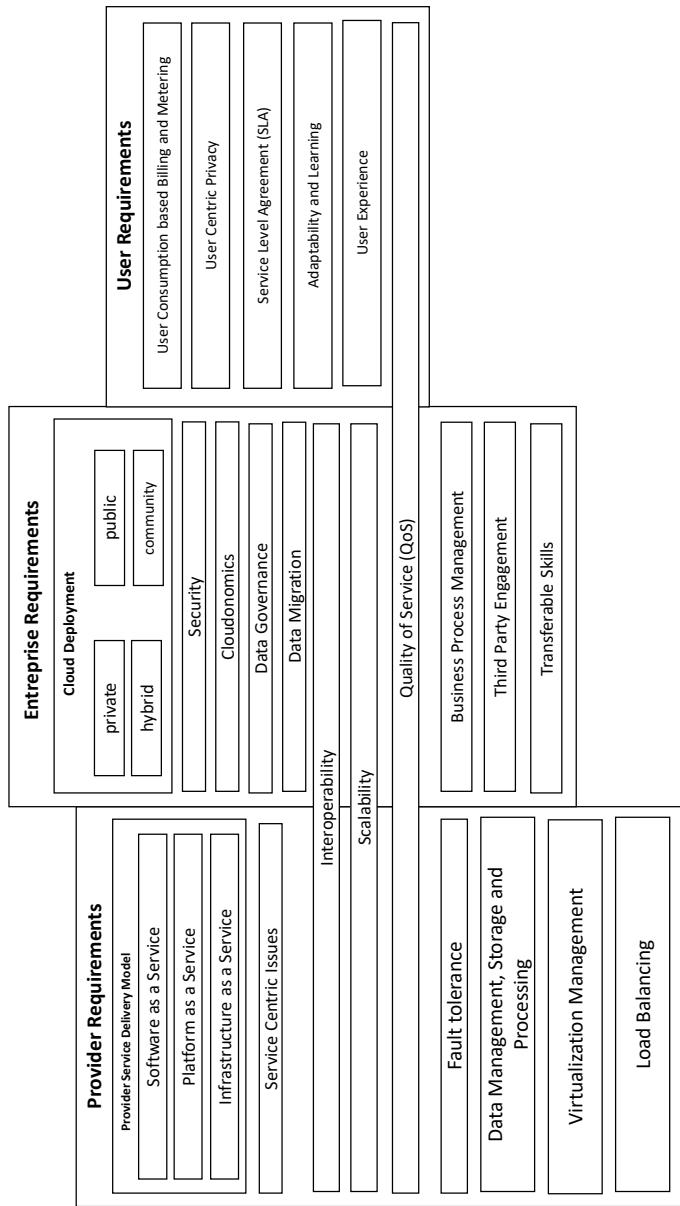


Abbildung A.2: Ebenen der „Cloud“-Anforderungsanalyse

Quelle: in Anlehnung an Rimal et al. 2011

A.2 Statistiken zum Themengebiet Cloud-C

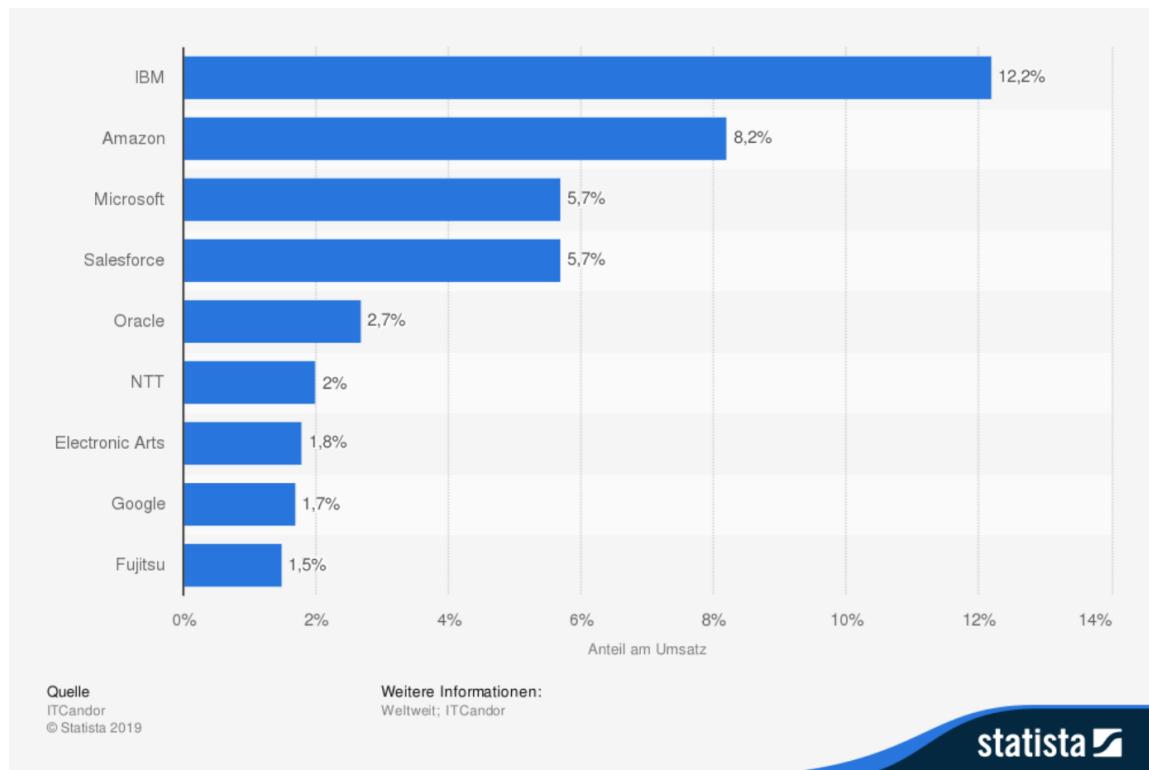


Abbildung A.3: Marktanteile der führenden Unternehmen am Umsatz im Bereich „Cloud-Computing“ weltweit von Juli 2018 bis Juni 2019

Quelle: ITCandor 2019

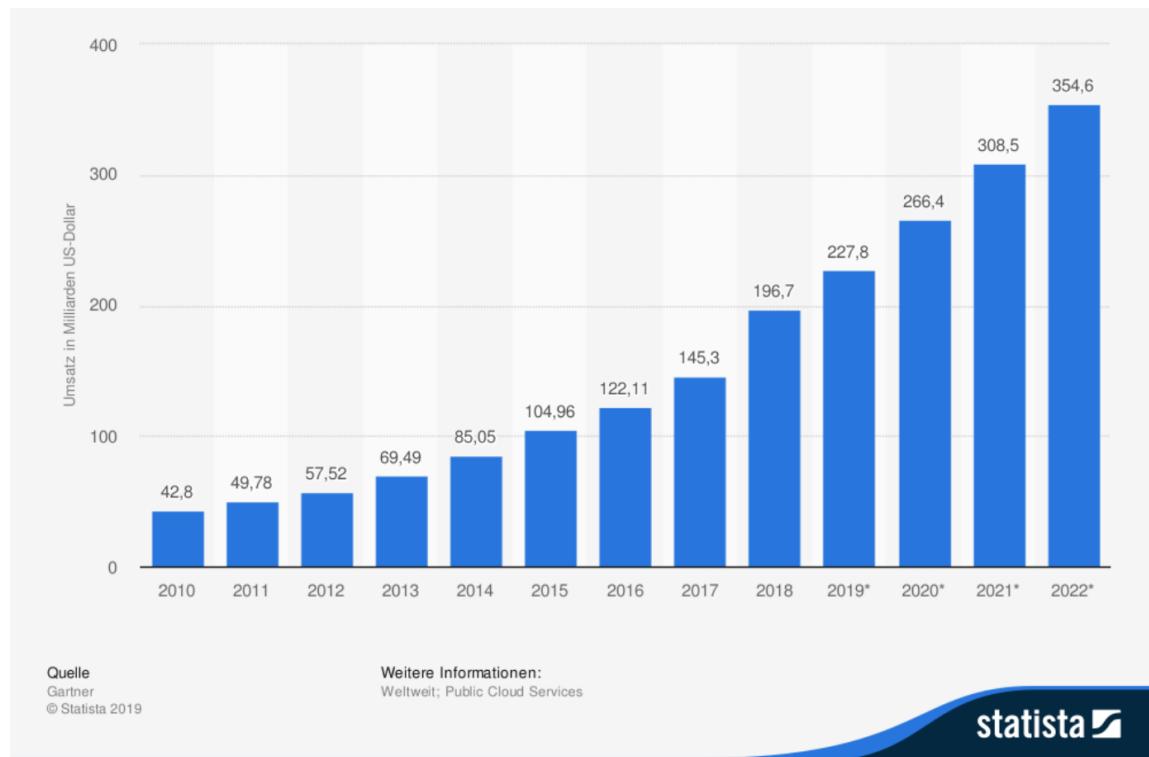


Abbildung A.4: Umsatz mit „Cloud-Computing“ weltweit von 2009 bis 2018 und Prognose bis 2022

Quelle: Gartner 2019

A.3 Ergänzungen zum Kapitel Container, Containerisierung und Orchestrierung

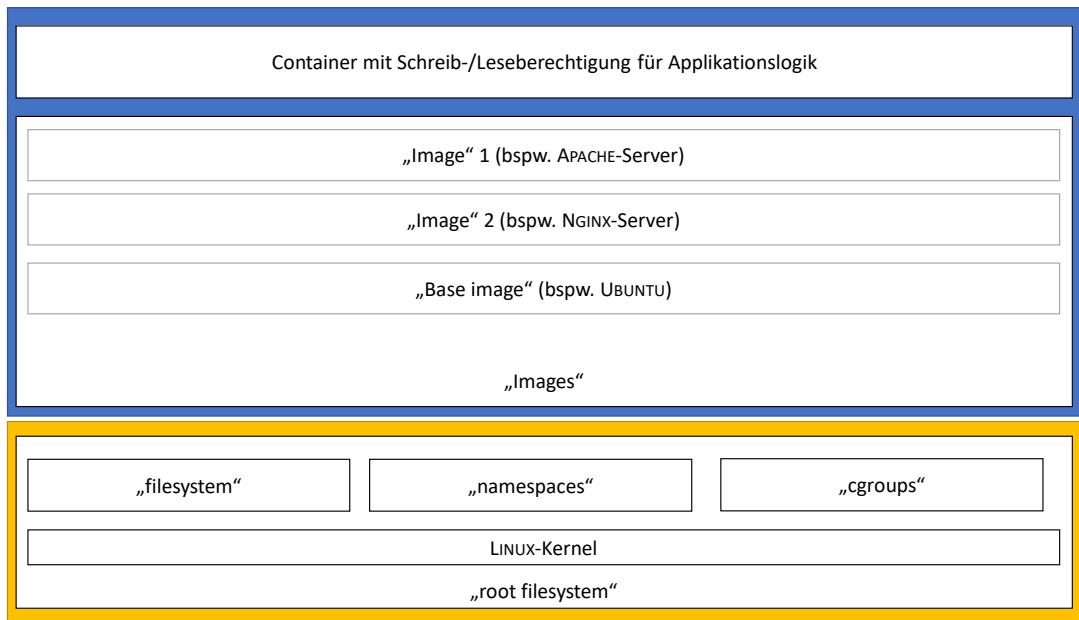


Abbildung A.5: Architektur des Container-„Images“

Quelle: in Anlehnung an Pahl 2015

Hierbei ist zu beachten, dass das orange Gefärbte die Funktionalitäten der DOCKER-„Engine“ und das blau Gefärbte die möglichen Bestandteile eines DOCKER-„Images“ darstellen.

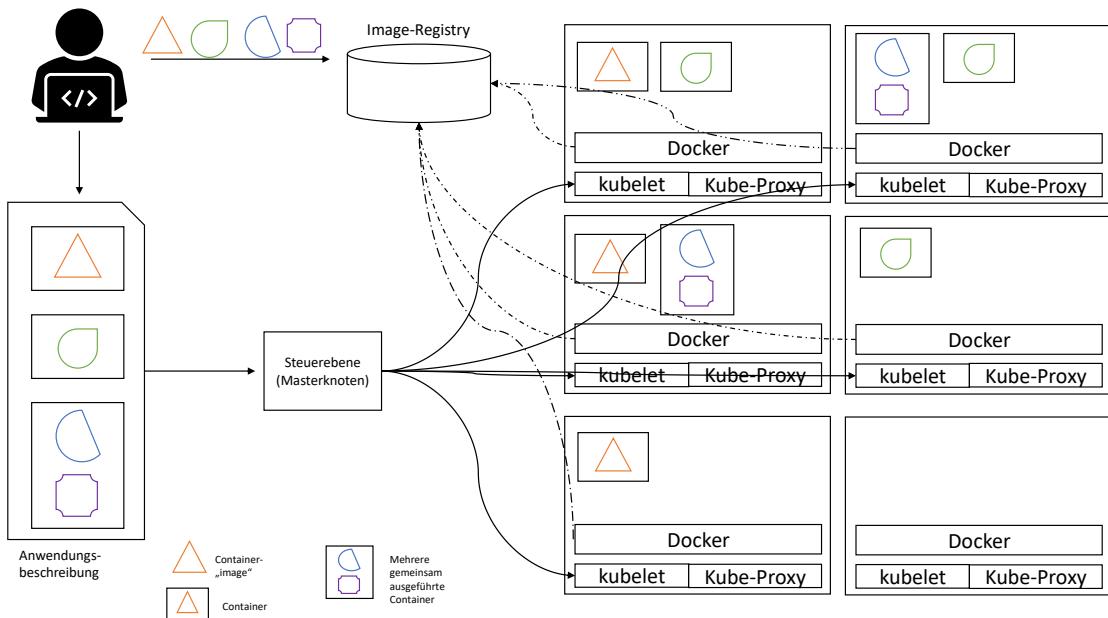


Abbildung A.6: Überblick über eine KUBERNETES-Architektur

Quelle: in Anlehnung an Lukša 2018, S. 23

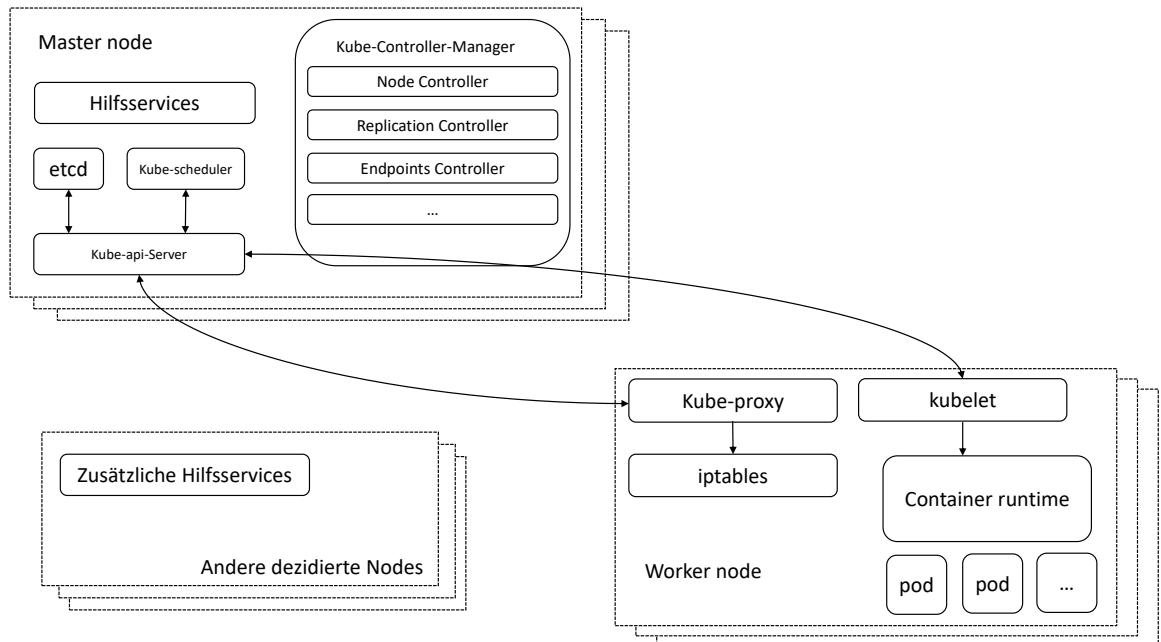


Abbildung A.7: Architektur der „K8s“-Interaktion

Quelle: in Anlehnung an Caban 2019, S. 15

A.4 Anforderungskatalog der Forschungsfrage eins

Die Klassifizierung beschreibt die Wichtigkeit (gering, mittel, hoch) im Kontext des zu implementierenden Prozesses und die Nummer dient der besseren Identifizierbarkeit der jeweiligen Anforderung.

Nummer	Klassifizierung	Anforderung
A_1	hoch	Das System muss die Möglichkeit bieten, über eine korrekte Konfigurationsdatei, alle Komponenten der Anwendung zu verteilen.
A_2	mittel	Das System soll die Möglichkeit bieten über eine standardisierte Übergabedatei alle Umgebungsvariablen des Containers zu definieren.
A_3	hoch	Das System muss die Möglichkeit bieten, über eine Validierung, konsistente Komponenten zu verteilen.
A_4	hoch	Das System muss die Möglichkeit bieten unabhängig von dem Inhalt des Containers diesen zu verteilen.
A_5	hoch	Das System muss fähig sein mit OPENSHIFT über das API zu kommunizieren.
A_6	mittel	Das System soll fähig sein das OPENSHIFT-Cluster zu jedem Zeitpunkt mit einem konsistenten Zustand zu verlassen.
A_7	niedrig	Das System wird die Möglichkeit bieten die Verteilung von Komponenten zu unterbrechen.
A_8	hoch	Das System muss die Möglichkeit bieten, über die Verteilungssoftware ARA, die Verteilung der Komponenten durchzuführen.
A_9	hoch	Das System muss die Möglichkeit bieten automatisiert alle notwendigen Komponenten über die Konfigurationsdatei zu erkennen und zu verteilen.
A_{10}	hoch	Das System muss den gesetzlichen sowie sicherheitstechnischen Vorgaben entsprechen.

Tabelle A.1: Anforderungskatalog zum implementierenden Prozess

A.5 Quellcode zum Testen der Webseiten-Verfügbarkeit

```

1  #!/bin/bash
2  # call function: test_connection <urlToCheck>
3
4  function test_connection() {
5      HTTPS_URL="$1"
6      CURL_CMD="curl -w httpcode=%{http_code}"
7
8      # -m, --max-time <seconds> FOR curl operation
9      CURL_MAX_CONNECTION_TIMEOUT="-m 100"
10
11     # perform curl operation
12     CURL_RETURN_CODE=0
13     CURL_OUTPUT=$((${CURL_CMD} ${CURL_MAX_CONNECTION_TIMEOUT}
14                         → ${HTTPS_URL} 2>/dev/null) || CURL_RETURN_CODE=$?
15     if [ ${CURL_RETURN_CODE} -ne 0 ]; then
16         echo "Curl connection failed with return code -
17             → ${CURL_RETURN_CODE}"
18     else
19         echo "Curl connection success"
20         # Check http code for curl operation/response in
21         → CURL_OUTPUT
22         httpCode=$(echo "${CURL_OUTPUT}" | sed -e
23                         → 's/.*\httpcode=/')
24         if [ ${httpCode} -ne 200 ]; then
25             echo "Curl operation/command failed due to server
26                 → return code - ${httpCode}"
27         fi
28     fi
29 }

```

Quelltext A.1: Funktion zum Test der Verfügbarkeit eines Webseite

A.6 Entwurf einer Konfigurationsdatei durch die Entwicklerinnen

```

1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4      name: deployment-demo
5      namespace: pjt-police-n-demo
6  spec:
7      selector:
8          matchLabels:
9              app: app-police-n-demo
10         replicas: 1
11         template:
12             metadata:
13                 labels:
14                     app: app-police-n-demo
15             spec:
16                 serviceAccountName: sa-police-n-demo
17                 containers:
18                     - name: police-nkopie-ep-demo
19                         image: $urlToRepo_/police-nkopie:2.1.0
20                         env:
21                             - name: INKASSOMAHNINFOPORT2_ENDPOINT_URI
22                                 value: $urlToServer_:10702/mock/$id_
23                             - name: DOKUMENTINFORMATION1_ENDPOINT_URI
24                                 value: $urlToServer_:10701/mock/$id_
25                             - name: VERTRAGPRUEFUNG1_ENDPOINT_URI
26                                 value: $urlToServer_:10703/mock/$id_
27                             - name: DRUCKAUFTRAGPORT_ENDPOINT_URI
28                                 value: $urlToServer_:10704/mock/$id_
29                             - name: PRODUKTAUSKUNFT1_ENDPOINT_URI
30                                 value: $urlToServer_:10705/mock/$id_
31                             - name:
32                                 → PARTNERPRUEFUNGASYNCHRON_ENDPOINT_URI
33                                 value: $urlToServer_:10709/mock/$id_
34                             - name: PROZESSAKTIVITAET2_ENDPOINT_URI
35                                 value: $urlToServer_:10706/mock/$id_
36                             - name: SPRING_DATASOURCE_URL
37                                 value: $urlToJDBC_
38                             - name: SPRING_DATASOURCE_USERNAME

```

```
38          value: $usernameDatasource_
39      - name: SPRING_DATASOURCE_PASSWORD
40          value: $passwordDatasource_
41      - name: CAMUNDA_BPM_DATABASE_TYPE
42          value: $databaseType_
43 ports:
44     - containerPort: 8080
45 resources:
46     limits:
47         cpu: 1
48         memory: 1Gi
49     requests:
50         cpu: "0.5"
51         memory: 500Mi
```

Quelltext A.2: Entwurf einer „Deployment“-Datei für CAMUNDA

Dabei stellen alle Zeichenketten mit dem Muster „\$Bezeichner_“ ein Platzhalter dar. Aus Gründen des Schutzes der Infrastruktur der SVI/SV ist es nicht gestattet die wirklichen Endpunkte der Webservices im Klartext abzubilden – sie unterliegen dem Betriebsgeheimnis.

A.7 Aufbau der zulässigen Wortfolgen der Übergabedatei

```

1  <Zeile> ::= <Schluessel> <Zuweisung> <Wert> <EOL> | <Gruppe> <EOL>
   ↪ | λ <EOL>
2  <Schluessel> ::= <Grossbuchstaben> | <Grossbuchstaben> (
   ↪ Schluessel)
3  <Zuweisung> ::= "="
4  <Wert> ::= <Zeichenkette>
5  <Gruppe> ::= "[" <Zeichenfolge> "]"
6  <Zeichenfolge> ::= <Grossbuchstaben> | <Grossbuchstaben> (
   ↪ Zeichenfolge)
7  <Zeichenkette> ::= λ | <Zeichen> | <Zeichen> <Zeichenkette>
8  <Grossbuchstaben> ::= "A" | "B" | "C" | "D" | "E" | "F" | "G" |
   ↪ "H" | "I" | "J" | "K" | "L" | "M" | "N" | "O" | "P" |
   ↪ "Q" | "R" | "S" | "T" | "U" | "V" | "W" | "X" | "Y" |
   ↪ "Z" |
9  <Zeichen> ::= "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" |
   ↪ "I" | "J" | "K" | "L" | "M" | "N" | "O" | "P" | "Q" |
   ↪ "R" | "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z" |
   ↪ "_" | "/" | "-" | ":" | "." | "a" | "b" | "c" | "d" |
   ↪ "e" | "f" | "g" | "h" | "i" | "j" | "k" | "l" | "m" |
   ↪ "n" | "o" | "p" | "q" | "r" | "s" | "t" | "u" | "v" |
   ↪ "w" | "x" | "y" | "z"

```

Quelltext A.3: BNF der Übergabedatei

A.8 Grundgerüst der Konfigurationsdatei

```
1 kind: "DeploymentConfig"
2 apiVersion: "v1"
3 metadata:
4   name: "frontend"
5 spec:
6   template:
7     metadata:
8       labels:
9         name: "frontend"
10      spec:
11        containers:
12          - name: "helloworld"
13            image: "openshift/origin-ruby-sample"
14            ports:
15              - containerPort: 8080
16                protocol: "TCP"
17            env:
18              - name: "env1"
19                value: "http://"
20      replicas: 5
21      triggers:
22        - type: "ConfigChange"
23        - type: "ImageChange"
24        imageChangeParams:
25          automatic: true
26        containerNames:
27          - "helloworld"
28        from:
29          kind: "ImageStreamTag"
30          name: "origin-ruby-sample:latest"
31      strategy:
32        type: "Rolling"
33        rollingParams:
34          updatePeriodSeconds: 1
35          intervalSeconds: 1
36          timeoutSeconds: 120
37          maxSurge: "20%"
38          maxUnavailable: "10%"
39          pre: {}
```

```
40     post: {}
41     paused: false
42     revisionHistoryLimit: 2
43     minReadySeconds: 0
```

Quelltext A.4: Grundgerüst der Konfigurationsdatei

In Anlehnung an Red Hat, Inc. 2019, Application → Deployments.

```
1 [CONTAINERS]
2 NAME=helloworld
3 IMAGE=openshift/origin-test
4
5 [ENV]
6 NAME=envTest
7 VALUE=https://server.example.loc:8080
```

Quelltext A.5: Beispielumsetzung der BNF der Übergabedatei

B Ergänzungen zur Forschungsfrage zwei

In diesem Teil des Anhangs sind Ergänzungen zur Forschungsfrage zwei des Kapitels 4 auf Seite 38 beschrieben.

B.1 Entscheidung über die Notwendigkeit eines „Business Case“

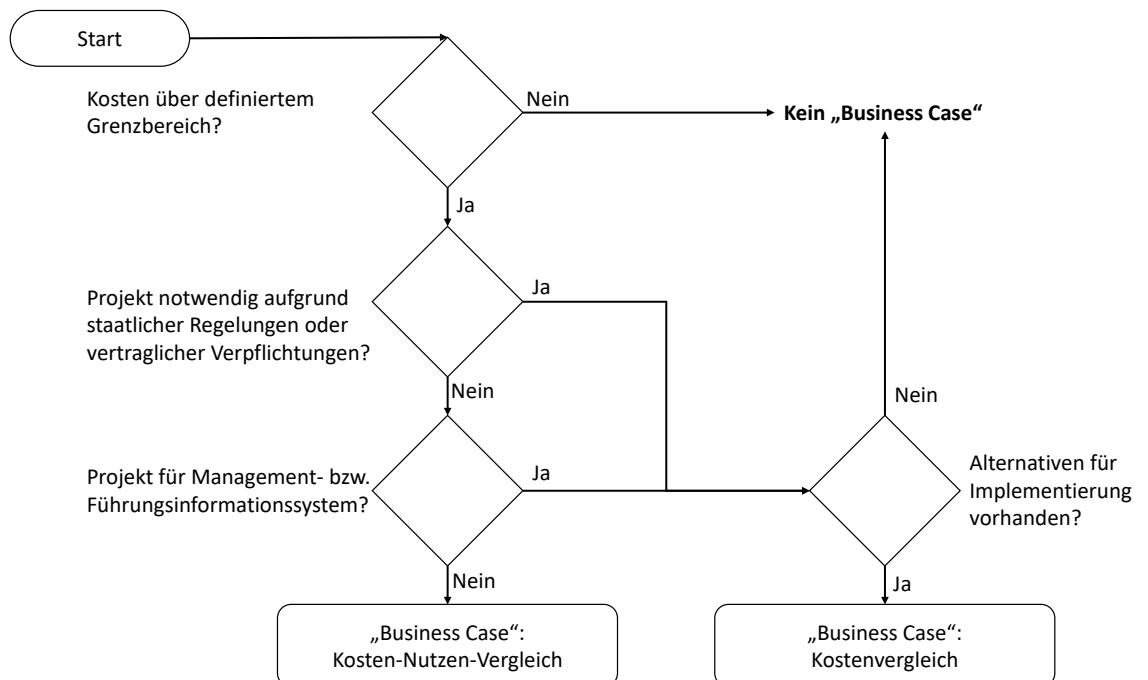


Abbildung B.1: Notwendigkeit eines „Business Case“

Quelle: in Anlehnung an Brugger 2009, S. 29

Vorteile	Nachteile
Neutralität	Kein Wissenstransfer
Verfügbarkeit	Kosten, egal ob der „Business Case“ rentabel ist
Effiziente Erstellung	Hohe Kosten im Vergleich zur internen Erstellung
Glaubwürdigkeit	Abhängigkeit
Qualität	Verlust der firmeninternen Standards
Innovation	
Schlichtung	

Tabelle B.1: Überblick über die Vor- und Nachteile der externen Erstellung eines „Business Case“

Quelle: in Anlehnung an Brugger 2009, S. 34

Vorteile	Nachteile
Wissensaufbau	Verfügbarkeit
Qualität	Effizienzverlust bei rein technischen/operativen Mitarbeitern
„Teamwork“	Glaubwürdigkeit
Standardisierung	Qualitätskontrolle durch „Controlling“-Division

Tabelle B.2: Überblick über die Vor- und Nachteile der internen Erstellung eines „Business Case“

Quelle: in Anlehnung an Brugger 2009, S. 34

B.2 Vor- und Nachteile der internen beziehungsweise externen Erstellung eines „Business Case“

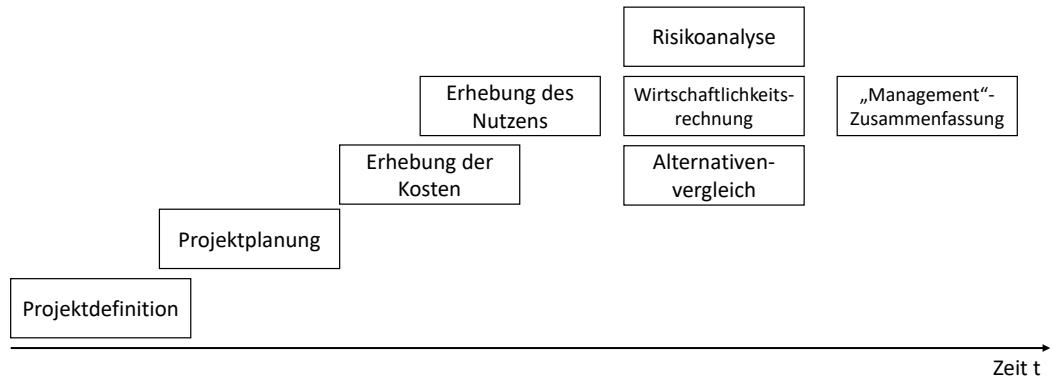


Abbildung B.2: Chronologische Abfolge der Entwicklungsphase eines „Business Case“

Quelle: in Anlehnung an Herman und Siegelaub 2009

C Ergänzungen zur Forschungsfrage drei

In diesem Teil des Anhangs sind Ergänzungen zur Forschungsfrage drei des Kapitels 5 auf Seite 50 beschrieben.

C.1 „Plan-Do-Check-Act“-Regelkreis

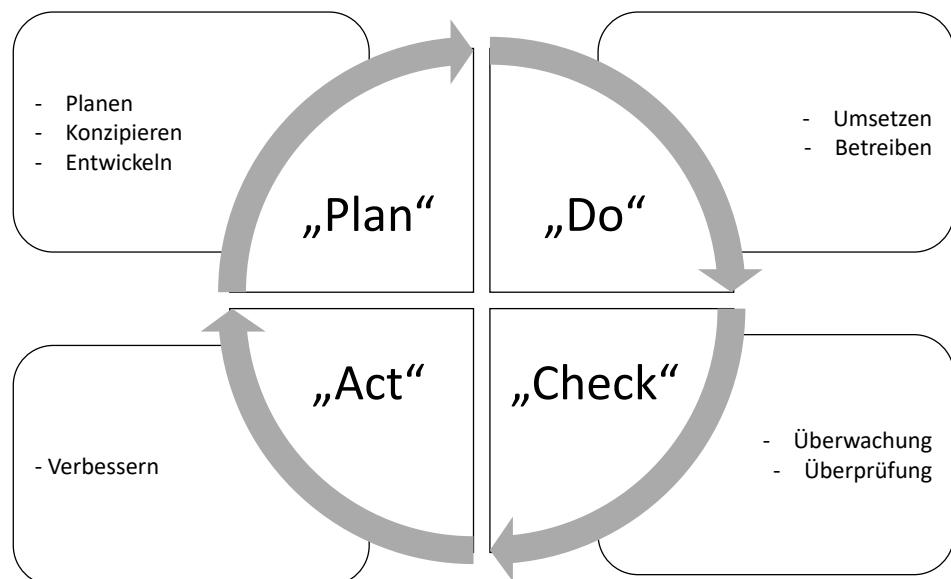


Abbildung C.1: „Plan-Do-Check-Act“-Regelkreis

Quelle: in Anlehnung an Kersten et al. 2020, S. 12

C.2 Checkliste zur Vorbereitung der ISMS-Einführung

Aktion	Gegenstand	Erfüllt?
1	Sind die Normen (27000, 27001, 27002) in aktueller elektronischer Form vorhanden?	<input type="checkbox"/>
2	Sind die Vorteile und der Nutzen eines ISMS erläutert worden?	<input type="checkbox"/>
3	Ist ein Grob-Abgleich mit ISO 27001 erfolgt? (Ziel: erste Aufwandsabschätzung)	<input type="checkbox"/>
4	Ist eine Entscheidung zur Orientierung an der ISO 27001 getroffen worden?	<input type="checkbox"/>
5	Denken wir in Management-Systemen? Existieren schon andere Management-Systeme?	<input type="checkbox"/>
6	Ist der Begriff ISMS eingeführt?	<input type="checkbox"/>
7	Denken wir in Geschäftsprozessen und informationstechnischen Anwendungen?	<input type="checkbox"/>
8	Ist der Anwendungsbereich des ISMS (Scope) zumindest grob skizziert?	<input type="checkbox"/>
9	Sind zumindest die Top Level Assets und deren Asset/Risk Owner erfasst worden?	<input type="checkbox"/>
10	Wurden – zumindest grob – Sicherheitsziele für diese Assets festgelegt?	<input type="checkbox"/>

Tabelle C.1: Checkliste zur Vorbereitung der ISMS-Einführung

Quelle: in Anlehnung an Kersten et al. 2020, S. 15

C.3 Schichtenmodell des IT-Grundschutzes

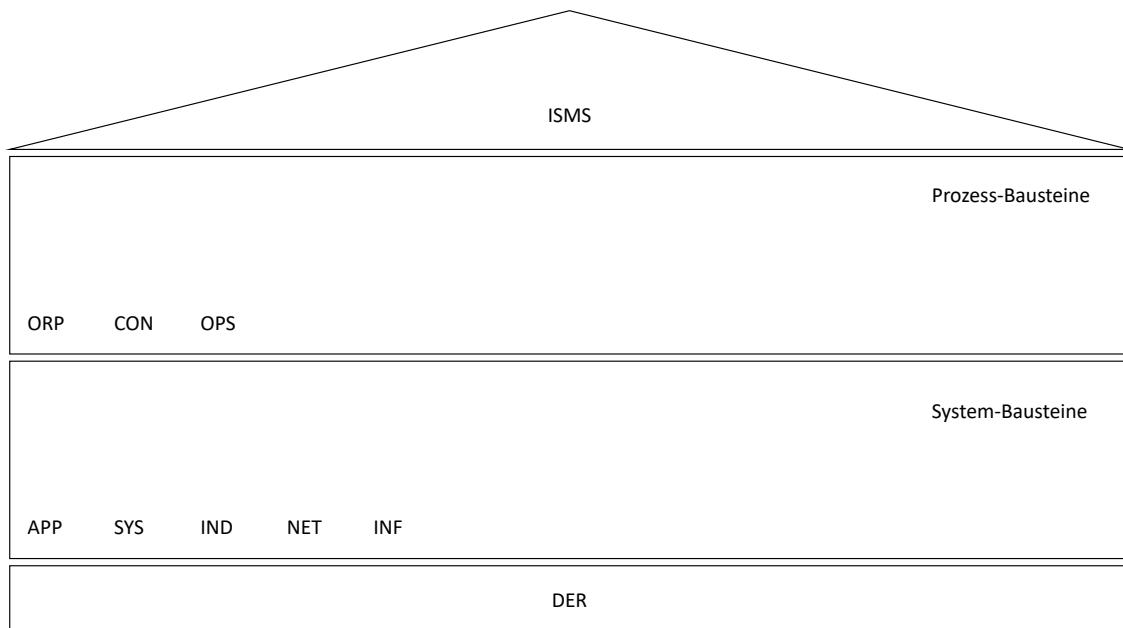


Abbildung C.2: Schichtenmodell des IT-Grundschutzes

Quelle: in Anlehnung an Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 9

"Die Prozessbausteine, die in der Regel für sämtliche oder große Teile eines Informationsverbunds gleichermaßen gelten, unterteilen sich in die folgenden Schichten, die wiederum aus weiteren Teilschichten bestehen können.

- Die Schicht ISMS enthält als Grundlage für alle weiteren Aktivitäten im Sicherheitsprozess den Baustein Sicherheitsmanagement.
- Die Schicht ORP befasst sich mit organisatorischen und personellen Sicherheitsaspekten. In diese Schicht fallen beispielsweise die Bausteine Organisation und Personal.
- Die Schicht CON enthält Bausteine, die sich mit Konzepten und Vorgehensweisen befassen. Typische Bausteine der Schicht CON sind unter anderem Kryptokonzept und Datenschutz.
- Die Schicht OPS umfasst alle Sicherheitsaspekte betrieblicher Art. Insbesondere sind dies die Sicherheitsaspekte des operativen IT-Betriebs, sowohl bei einem Betrieb im Haus, als auch bei einem IT-Betrieb, der in Teilen oder komplett durch Dritte betrieben wird. Ebenso enthält er die Sicherheitsaspekte, die bei

einem IT-Betrieb für Dritte zu beachten sind. Beispiele für die Schicht OPS sind die Bausteine Schutz vor Schadprogrammen und Outsourcing für Kunden.

- In der Schicht DER finden sich alle Bausteine, die für die Überprüfung der umgesetzten Sicherheitsmaßnahmen, die Detektion von Sicherheitsvorfällen sowie die geeigneten Reaktionen darauf relevant sind. Typische Bausteine der Schicht DER sind Behandlung von Sicherheitsvorfällen und Vorsorge für IT-Forensik.

Neben den Prozess-Bausteinen beinhaltet das IT-Grundschatz-Kompendium auch System-Bausteine. Diese werden in der Regel auf einzelne Zielobjekte oder Gruppen von Zielobjekten angewendet. Die System-Bausteine unterteilen sich in die folgenden Schichten. Ähnlich wie die Prozess-Bausteine können auch die System-Bausteine aus weiteren Teilschichten bestehen.

- Die Schicht APP beschäftigt sich mit der Absicherung von Anwendungen und Diensten, unter anderem in den Bereichen Kommunikation, Verzeichnisdienste, netzbasierte Dienste sowie Business- und Client-Anwendungen. Typische Bausteine der Schicht APP sind Allgemeine Groupware, Office-Produkte, Webserver und Relationale Datenbanksysteme.
- Die Schicht SYS betrifft die einzelnen IT-Systeme des Informationsverbunds, die ggf. in Gruppen zusammengefasst wurden. Hier werden die Sicherheitsaspekte von Servern, Desktop-Systemen, Mobile Devices und sonstigen IT-Systemen wie Druckern und TK-Anlagen behandelt. Zur Schicht SYS gehören beispielsweise Bausteine zu konkreten Betriebssystemen, Allgemeine Smartphones und Tablets sowie Drucker, Kopierer und Multifunktionsgeräte.
- Die Schicht IND befasst sich mit Sicherheitsaspekten industrieller IT. In diese Schicht fallen beispielsweise die Bausteine Betriebs- und Steuerungstechnik, Allgemeine ICS-Komponente und Speicherprogrammierbare [sic!] Steuerung (SPS).
- Die Schicht NET betrachtet die Vernetzungsaspekte, die sich nicht primär auf bestimmte IT-Systeme, sondern auf die Netzverbindungen und die Kommunikation beziehen. Dazu gehören zum Beispiel die Bausteine NetzManagement, Firewall und WLAN-Betrieb.
- Die Schicht INF befasst sich mit den baulich-technischen Gegebenheiten, hier werden Aspekte der infrastrukturellen Sicherheit zusammengeführt. Dies betrifft unter anderem die Bausteine Allgemeines Gebäude und Rechenzentrum.¹³⁵

¹³⁵Bundesamt für Sicherheit in der Informationstechnik (BSI) 2020, S. 23-24.

D Ergänzungen zum Literaturverzeichnis

In diesem Kapitel werden die PDF-Dokumente der zitierten Webseiten dargestellt. Dies dient der Nachvollziehbarkeit der Quellen. Diese sind Auszüge aus den relevanten Webseiten und ohne Formatierung der Seiten des Dokuments beigefügt. Die nachfolgende Liste zählt die Quellen der Reihenfolge nach auf:

1. Google LLC 2020a
2. Red Hat, Inc. 2019, Application → Deployments
3. Google Ireland Limited 2020
4. Red Hat, Inc. 2020a
5. Sonatype Inc. 2020
6. Broadcom Inc. 2020
7. Camunda Services GmbH 2020
8. Canonical Ltd. 2020
9. Django Software Foundation 2020
10. Google LLC 2020b
11. Opensource.org 2020
12. Rupp 2020
13. The People of the GnuPG Project 2020

Ergänzungen zum Literaturverzeichnis

hull elizabeth requirements engineering - Google Scholar

https://scholar.google.de/scholar?hl=de&as_sdt=0,5&q=hull+elizabeth+...

Requirements engineering

J Dick, E Hull, K Jackson - 2017 - books.google.com

Written for those who want to develop their knowledge of requirements engineering process, whether practitioners or students. Using the latest research and driven by practical experience from industry, Requirements Engineering gives useful hints to practitioners on ...

☆ 99 Zitiert von: 1204 Ähnliche Artikel Alle 12 Versionen ☾

Requirements engineering

H Elizabeth, K Jackson, J Dick - 2011 - Springer

☆ 99 Zitiert von: 19 Ähnliche Artikel

DOORS: a tool to manage requirements

E Hull, K Jackson, J Dick - Requirements engineering, 2002 - Springer

Abstract Systems engineers and managers need the right instruments to assist them with the requirements management process. A variety of tools currently exist. This chapter presents an overview of one of these tools—DOORS (Version 5.2). DOORS (Dynamic Object Oriented ...

☆ 99 Zitiert von: 9 Ähnliche Artikel Alle 4 Versionen

Requirements engineering in the solution domain

E Hull, K Jackson, J Dick - Requirements Engineering, 2005 - Springer

The next step on from defining the system requirements is to create an architectural design as indicated by the second instantiation of the generic process in Figure 6.1. This must be expressed in terms of a set of components that interact to generate the emergent properties ...

☆ 99 Zitiert von: 7 Ähnliche Artikel Alle 4 Versionen

Requirements engineering in the problem domain

E Hull, K Jackson, J Dick - Requirements Engineering, 2005 - Springer

This leads to the identification of a group of people that we will refer to as "stakeholders". These are people or organizations that have some direct or indirect interest (or stake) in the intended system. Finally, we must examine what sorts of models are relevant to the problem ...

☆ 99 Zitiert von: 7 Ähnliche Artikel Alle 4 Versionen

A generic process for requirements engineering

E Hull, K Jackson, J Dick - Requirements Engineering, 2005 - Springer

This chapter introduces the concept of a process for the development of systems. It starts by examining the way in which systems are developed. This leads to the identification of a development pattern that can be used in many different contexts. This development pattern ...

☆ 99 Zitiert von: 7 Ähnliche Artikel Alle 5 Versionen

System Modelling for Requirements Engineering

E Hull, K Jackson, J Dick - Requirements Engineering, 2010 - Springer

Abstract System modelling supports the analysis and design process by introducing a degree of formality into the way systems are defined. During system development it is often the case that pictures are used to help visualize some aspects of the development ...

☆ 99 Zitiert von: 2 Ähnliche Artikel Alle 3 Versionen

Requirements engineering: From system goals to UML models to software

A Van Lamsweerde - 2009 - Chichester, UK: John Wiley & Sons

☆ 99 Zitiert von: 1342 Ähnliche Artikel Alle 5 Versionen ☾

Management Aspects of Requirements Engineering

J Dick, E Hull, K Jackson - Requirements Engineering, 2017 - Springer

The management of the requirements engineering process is similar to the management of any other endeavour. Before starting out it is necessary to understand what needs to be done. We need to know the sorts of activities that must be undertaken. We need to know whether there ...

☆ 99 Zitiert von: 1 Ähnliche Artikel

Ergänzungen zum Literaturverzeihnis

Understanding Deployments and DeploymentConfigs - Deployments | A... <https://docs.okd.io/latest/applications/deployments/what-deployments-are.html>

(/)

[Documentation \(/\)](#) / [OKD Latest \(./welcome/index.html\)](#) / Applications
/ [Deployments \(./applications/deployments/what-deployments-are.html\)](#)
/ Understanding Deployments and DeploymentConfigs

Understanding Deployments and DeploymentConfigs

Building blocks of a deployment
ReplicationControllers
ReplicaSets
DeploymentConfigs
Deployments
Comparing Deployments and DeploymentConfigs
Design
DeploymentConfigs-specific features
Deployments-specific features

Deployments and DeploymentConfigs in OKD are API objects that provide two similar but different methods for fine-grained management over common user applications. They are composed of the following separate API objects:

- A DeploymentConfig or a Deployment, either of which describes the desired state of a particular component of the application as a Pod template.



<https://twitter.com/openshift>



<https://github.com/openshift/origin>



DeploymentConfigs involve one or more ReplicationControllers, (<https://www.facebook.com/openshift>) which contain a point-in-time record of the state of a

 DeploymentConfig as a Pod template. Similarly, Deployments (<https://www.redhat.com/>) involve one or more ReplicaSets, a successor of ReplicationControllers. (<https://www.openshift.com/>)

- One or more Pods, which represent an instance of a particular version of an application.

Image attribution

 [Ascii Binder](https://github.com/redhataccess/ascii_binder/)
https://github.com/redhataccess/ascii_binder/

Ergänzungen zum Literaturverzeihnis

Understanding Deployments and DeploymentConfigs - Deployments | A... <https://docs.okd.io/latest/applications/deployments/what-deployments-are...>

Building blocks of a deployment

Deployments and DeploymentConfigs are enabled by the use of native Kubernetes API objects ReplicationControllers and ReplicaSets, respectively, as their building blocks.

Users do not have to manipulate ReplicationControllers, ReplicaSets, or Pods owned by DeploymentConfigs or Deployments. The deployment systems ensures changes are propagated appropriately.



If the existing deployment strategies are not suited for your use case and you must run manual steps during the lifecycle of your deployment, then you should consider creating a Custom deployment strategy.

The following sections provide further details on these objects.

ReplicationControllers

A ReplicationController ensures that a specified number of replicas of a Pod are running at all times. If Pods exit or are deleted, the ReplicationController acts to instantiate more up to the defined number. Likewise, if there are more running than desired, it deletes as many as necessary to match the defined amount.

A ReplicationController configuration consists of:

- The number of replicas desired (which can be adjusted at runtime).
- A Pod definition to use when creating a replicated Pod.
- A selector for identifying managed Pods.

A selector is a set of labels assigned to the Pods that are managed by the ReplicationController. These labels are included in the Pod definition that the ReplicationController instantiates. The ReplicationController uses the selector to determine how many instances of the Pod are already running in order to adjust as needed.

Ergänzungen zum Literaturverzeichnis

Understanding Deployments and DeploymentConfigs - Deployments | A... <https://docs.okd.io/latest/applications/deployments/what-deployments-are...>

The ReplicationController does not perform auto-scaling based on load or traffic, as it does not track either. Rather, this requires its replica count to be adjusted by an external auto-scaler.

The following is an example definition of a ReplicationController:

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: frontend-1
spec:
  replicas: 1    1
  selector:      2
    name: frontend
  template:      3
    metadata:
      labels:   4
        name: frontend  5
    spec:
      containers:
        - image: openshift/hello-openshift
          name: helloworld
        ports:
          - containerPort: 8080
            protocol: TCP
      restartPolicy: Always
```

- 1 The number of copies of the Pod to run.
- 2 The label selector of the Pod to run.
- 3 A template for the Pod the controller creates.
- 4 Labels on the Pod should include those from the label selector.
- 5 The maximum name length after expanding any parameters is 63 characters.

ReplicaSets

Similar to a ReplicationController, a ReplicaSet is a native Kubernetes

Ergänzungen zum Literaturverzeichnis

Container und ihre Vorteile | Google Cloud

about:reader?url=https://cloud.google.com/containers?hl=de

[cloud.google.com](https://cloud.google.com/containers)

Container und ihre Vorteile | Google Cloud

14-17 Minuten

Google-Ansatz

Bei Google wird von Gmail über YouTube bis hin zur Suche alles in Containern ausgeführt. Dank der Containerisierung arbeiten unsere Entwicklerteams schneller als zuvor und können Software effizient bereitstellen. Wir starten pro Woche mehrere Milliarden Container. In den letzten zehn Jahren haben wir viel über die Ausführung von Containerarbeitslasten in der Produktion gelernt und [dieses Wissen an die Community weitergegeben](#) – von den Anfängen, als wir [cgroups zum Linux-Kernel](#) beisteuerten, bis hin zur Entscheidung, Designs unserer internen Tools als [Open-Source-Projekt Kubernetes](#) bereitzustellen. Dieses Know-how ist nun in der [Google Cloud Platform](#) vereint, sodass Entwickler und Unternehmen jeder Größenordnung problemlos die neuesten Containerinnovationen nutzen können.



Container-Grundlagen: Was sind Container?

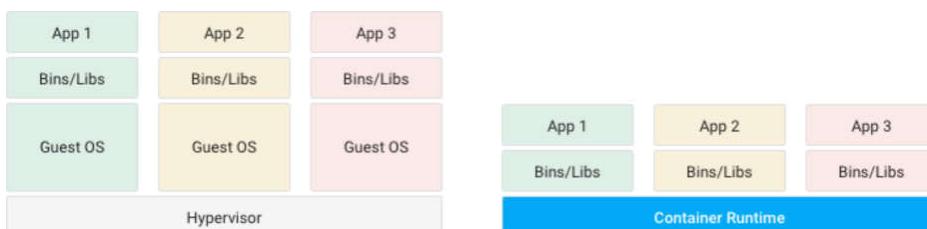
Ergänzungen zum Literaturverzeihnis

Container und ihre Vorteile | Google Cloud

about:reader?url=https://cloud.google.com/containers?hl=de

Container bieten einen logischen Mechanismus der Paketerstellung, der darauf beruht, dass Anwendungen von ihrer Ausführungsumgebung abstrahiert werden. Mit dieser Entkopplung können containerbasierte Anwendungen einfach und konsistent bereitgestellt werden, unabhängig davon, ob es sich bei der Zielumgebung um ein privates Rechenzentrum, die öffentliche Cloud oder auch um den persönlichen Laptop eines Entwicklers handelt. Die Containerisierung sorgt dafür, dass sich Entwickler auf ihre Anwendungslogik und -abhängigkeiten konzentrieren können, während sich operative IT-Teams auf die Bereitstellung und Verwaltung konzentrieren dürfen, ohne sich Gedanken über Anwendungsdetails wie spezifische Softwareversionen und Konfigurationen machen zu müssen.

Container werden oft mit virtuellen Maschinen (VMs) verglichen. Möglicherweise sind Sie bereits mit VMs vertraut: Bei diesen handelt es sich um ein Gastbetriebssystem wie Linux oder Windows, das auf einem Host-Betriebssystem mit virtualisiertem Zugriff auf die zugrunde liegende Hardware ausgeführt wird. Wie bei virtuellen Maschinen können Sie auch mithilfe von Containern Ihre Anwendung mit Bibliotheken und anderen Abhängigkeiten zusammenpacken. Dadurch wird eine isolierte Umgebung für die Ausführung Ihrer Softwaredienste geschaffen. Wie Sie im Schaubild unten sehen, hören hier die Gemeinsamkeiten aber auch schon auf. Container ermöglichen Entwicklern und operativen IT-Teams ein einfacheres Arbeiten mit einer Vielzahl von Vorteilen.



docs.openshift.com

CLI Operations | CLI Reference

9-12 Minuten

You are viewing documentation for a release that is no longer supported. The latest supported version of version 3 is [3.11]. For the most recent version 4, see [4]

You are viewing documentation for a release that is no longer supported. The latest supported version of version 3 is [3.11]. For the most recent version 4, see [4]

Overview

This topic provides information on the CLI operations and their syntax. You must [setup and login](#) with the CLI before you can perform these operations.

Common Operations

The CLI allows interaction with the various objects that are managed by OpenShift. Many common `oc` operations are invoked using the following syntax:

```
$ oc <action> <object_type> <object_name_or_id>
```

This specifies:

- An `<action>` to perform, such as `get` or `describe`.
- The `<object_type>` to perform the action on, such as `service` or the abbreviated `svc`.
- The `<object_name_or_id>` of the specified `<object_type>`.

For example, the `oc get` operation returns a complete list of

Ergänzungen zum Literaturverzeichnis

CLI Operations | CLI Reference

about:reader?url=https://docs.openshift.com/enterprise/3.0/cli_reference/...

services that are currently defined:

```
$ oc get svc
```

NAME	LABELS	IP	PORT (S)
SELECTOR			
docker-registry	docker-registry=default		
docker-registry=default		172.30.78.158	5000/TCP
kubernetes			
component=apiserver,provider=kubernetes		<none>	
172.30.0.2		443/TCP	
kubernetes-ro			
component=apiserver,provider=kubernetes		<none>	
172.30.0.1		80/TCP	

The `oc describe` operation can then be used to return detailed information about a specific object:

```
$ oc describe svc docker-registry
```

Name:	docker-registry
Labels:	docker-registry=default
Selector:	docker-registry=default
IP:	172.30.78.158
Port:	<unnamed> 5000/TCP
Endpoints:	10.1.0.2:5000
Session Affinity:	None
No events.	

Versions of `oc` prior to 3.0.2.0 did not have the ability to negotiate API versions against a server. So if you are using `oc` up to 3.0.1.0 with a server that only supports v1 or higher versions of the API, make sure to pass `--api-version` in order to point the `oc` client to the correct API endpoint. For example: `oc get svc --api-version=v1`.

Basic CLI Operations

Ergänzungen zum Literaturverzeichnis

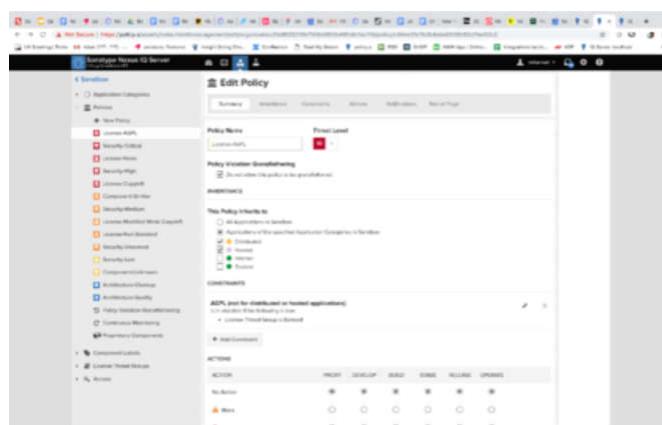
Nexus Lifecycle - Bereinigen Sie Ihre gesamte Software Supply Chain ko... about:reader?url=https://de.sonatype.com/product-nexus-lifecycle?utm_...

de.sonatype.com

Nexus Lifecycle - Bereinigen Sie Ihre gesamte Software Supply Chain kontinuierlich | Sonatype

Sonatype Inc.

2-3 Minuten



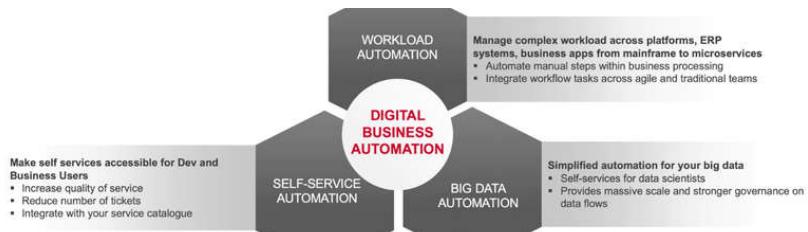
Setzen Sie Open-Source-Richtlinien im gesamten SDLC durch

Erstellen Sie benutzerdefinierte Sicherheits-, Lizenz- und Architekturrichtlinien basierend auf Anwendungstyp oder Unternehmen und setzen Sie diese Richtlinien dem Kontext entsprechend über jede Phase des SDLC hinweg durch. Die automatische Durchsetzung von Richtlinien kann nur mit der Genauigkeit und Präzision von [Nexus Intelligence](#) erfolgen. Dadurch werden Fehlalarme und Falschmeldungen eliminiert, die

Ergänzungen zum Literaturverzeichnis

Automic Automation

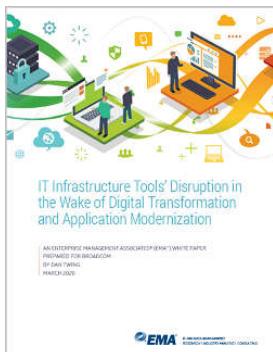
<https://www.broadcom.com/products/software/automation/digital-business-automation>



Enterprises need to automate a complex and diverse landscape of applications, platforms and technologies to deliver services in a competitive digital business environment. Digital Business Automation is essential to succeed and scale your IT operations:

- You have to manage complex workload across platforms, ERP systems, business apps from mainframe to microservices and the cloud. Automating manual steps within business processing and integrating workflow tasks across agile and traditional teams.
- You need to simplify automation for your big data, enabling self-services for data scientists while providing massive scale and stronger governance on data flows.
- You also require self-services available for Dev and Business users to increase the quality of service, reduce the number of service-desk tickets and integrate with your service catalogue.

Automic Automation gives you the agility, speed and reliability required for effective Digital Business Automation. From a single unified platform, Automic centrally delivers the Workload Automation, Self-Service Automation, and Big Data Automation capabilities needed to drive growth of your company and accelerate your digital transformation.



INDUSTRY ANALYST REPORT

EMA Research report on IT Infrastructure Tools' Disruption in the Wake of Digital Transformation and Application Modernization.

[Read EMA Research Report](#)

Automic Automation Capabilities

Ergänzungen zum Literaturverzeichnis

Workflow und Decision Automation | Camunda BPM

<https://camunda.com/de/>



Announcing CamundaCon LIVE: Online Process Automation Conference
Join us free April 23-24, 2020 from 10am – 4pm EST/ 4pm – 10pm CET
[Register Now \(https://www.camunda.com/conference\)](https://www.camunda.com/conference)

(<https://camunda.com/de/>)

Prozessautomatisierung neu gedacht für das digitale Unternehmen

Mehr Infos (<https://camunda.com/de/products/>)

Download (<https://camunda.com/de/download>)

Developer-friendly	Hoch Skalierbar	BPMN / DMN Standards
	Business-IT-Kollaboration	

Die Camunda Plattform für Workflow und Decision Automation unterstützt Tausende von Entwicklern bei der Automatisierung von Geschäftsprozessen, um die Flexibilität, Transparenz und Skalierbarkeit zu erreichen, die für die digitale Transformation unabdingbar sind.



Modellieren

Modellieren Sie BPMN-Workflows und DMN-Entscheidungen in einem Editor, den sowohl Anwender als auch Entwickler gerne benutzen.



Automatisieren

Führen Sie Ihre Modelle mit Execution Engines für BPMN und DMN aus, und nutzen Sie zusätzliche Tools für den Produktivbetrieb.



Verbessern

Nutzen Sie Reports, Analysen und Echtzeit-Monitoring, um Ihre Prozesse ausfallsicher zu betreiben und kontinuierlich zu verbessern.

Entdecken Sie den kompletten Camunda Stack für Workflow und Decision Automation

[Mehr Infos \(https://camunda.com/de/products/\)](https://camunda.com/de/products/)

Ergänzungen zum Literaturverzeichnis

Linux Containers

<https://linuxcontainers.org/>



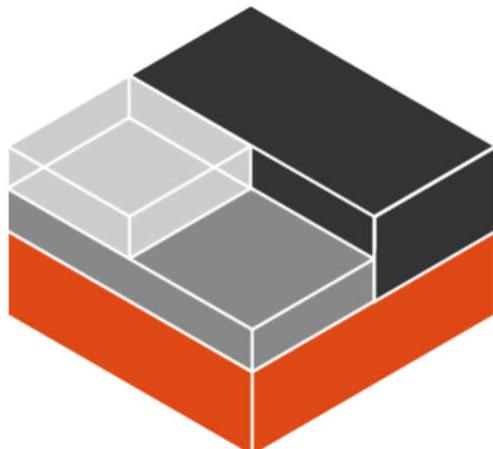
Home

Infrastructure for container projects.

linuxcontainers.org is the umbrella project behind LXC, LXD and LXCFS.

The goal is to offer a distro and vendor neutral environment for the development of Linux container technologies.

Our main focus is system containers. That is, containers which offer an environment as close as possible as the one you'd get from a VM but without the overhead that comes with running a separate kernel and simulating all the hardware.



Active projects

LXC

Ergänzungen zum Literaturverzeichnis

The Web framework for perfectionists with deadlines | Django

<https://www.djangoproject.com/>

Stay in the loop

Subscribe to one of our mailing lists to stay up to date with everything in the Django community:

Using Django

Get help with Django and follow announcements.

You can also subscribe by sending an email to django-users+subscribe@googlegroups.com and following the instructions that will be sent to you.

Contributing to Django

Contribute to the development of Django itself.

Before asking a question about how to contribute, read Contributing to Django. Many frequently asked questions are answered there.

You can also subscribe by sending an email to django-developers+subscribe@googlegroups.com and following the instructions that will be sent to you.

We have a few other specialized lists (mentorship, i18n, ...). You can find more information about them in our [mailing list documentation](#).

Learn More

[About Django](#)

[Getting Started with Django](#)

[Team Organization](#)

[Django Software Foundation](#)

[Code of Conduct](#)

[Diversity Statement](#)

Get Involved

[Join a Group](#)

[Contribute to Django](#)

[Submit a Bug](#)

[Report a Security Issue](#)

Follow Us

[GitHub](#)

[Twitter](#)

[News RSS](#)

[Django Users Mailing List](#)

Ergänzungen zum Literaturverzeichnis

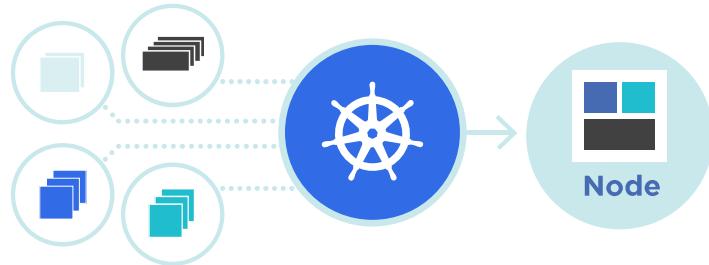
Production-Grade Container Orchestration - Kubernetes

<https://kubernetes.io/>

Production-Grade Container Orchestration

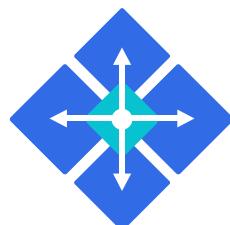
Automated container deployment, scaling, and management

Learn Kubernetes Basics



Kubernetes (K8s) is an open-source system for automating deployment, scaling, and management of containerized applications.

It groups containers that make up an application into logical units for easy management and discovery. Kubernetes builds upon [15 years of experience of running production workloads at Google](#), combined with best-of-breed ideas and practices from the community.



opensource.org

The Open Source Definition | Open Source Initiative

4 Minuten

Introduction

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost, preferably downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program.

Ergänzungen zum Literaturverzeichnis

The Open Source Definition | Open Source Initiative

about:reader?url=https://opensource.org/osd

Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of License

Ergänzungen zum Literaturverzeichnis

The Open Source Definition | Open Source Initiative

about:reader?url=https://opensource.org/osd

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

10. License Must Be Technology-Neutral

No provision of the license may be predicated on any individual technology or style of interface.

The Open Source Definition was originally derived from the [Debian Free Software Guidelines](#) (DFSG).

Last modified, 2007-03-22

Ergänzungen zum Literaturverzeichnis

Formulierungsregel für die funktionalen Anforderungen

Anforderung:	Aussagekräftiger Name <Beispiel: Registrierung>
Beschreibung <Satzaufbau>	<p><i>Zielsystem + Priorität + Systemaktivität + Ergänzungen + Funktionalität + Bedingungen</i></p> <p>Das Bibliothekssystem + muss + dem Administrator die Möglichkeit bieten + über eine Maske einen neuen Bibliotheksuser + zu registrieren.</p>

Erklärung:

Das Zielsystem = Das zu entwickelnde System bzw. seine Subsysteme / Komponenten
Priorität = {muss (höhe Priorität), soll (mittlere Priorität), wird (niedriger Priorität)}
Systemaktivität = { -, <wem> die Möglichkeit bieten, fähig sein}
- = selbständige Systemaktivität → (Das System führt die Funktionalität automatisch)
<wem> die Möglichkeit bieten = Benutzerinteraktion → (Das System stellt dem Nutzer die Funktionalität zur Verfügung)
fähig sein = Schnittstellenanforderung → Das System führt eine Funktionalität in Abhängigkeit von einem Fremdsystem (Dritten) aus, ist an sich passiv und wartet auf ein externes Ereignis
Ergänzungen = { wie, wohin, womit,..usw. } Überlege welche Objekte und Ergänzungen der Objekte in der Anforderung noch fehlen und ergänze sie!
Funktionalität = Verb in Infinitive Die geforderte Funktionalität/Vorgang/Tätigkeit bzw. das Verhalten
Bedingung = {wenn zeitlich, falls logisch} Unter welchen Bedingungen wird die Funktionalität angestoßen?
Beispiel: Anforderung: J3D Szenengraphen laden Reflect Media Player muss dem Benutzer die Möglichkeit bieten J3D Szenengraphen aus einer Datei in wrml Format über das Netzwerk zu laden. Erklärung: Reflect Media Player(Zielsystem) muss (höhe Priorität) dem Benutzer die Möglichkeit bieten (Systemaktivität = Benutzerinteraktion), J3D Szenengraphen aus einer Datei in wrml Format über das Netzwerk (Objekt & Objektergänzungen) zu laden (Funktionalität)

Ergänzungen zum Literaturverzeichnis

The GNU Privacy Guard

<https://gnupg.org/>



[Home](#) [Donate](#) [Software](#) [Download](#) [Documentation](#) [Blog](#)

THE GNU PRIVACY GUARD

GnuPG is a complete and free implementation of the OpenPGP standard as defined by [RFC4880](#) (also known as *PGP*). GnuPG allows you to encrypt and sign your data and communications; it features a versatile key management system, along with access modules for all kinds of public key directories. GnuPG, also known as *GPG*, is a command line tool with features for easy integration with other applications. A wealth of [frontend applications](#) and [libraries](#) are available. GnuPG also provides support for S/MIME and Secure Shell (ssh).

Since its introduction in 1997, GnuPG is [Free Software](#) (meaning that it respects your freedom). It can be freely used, modified and distributed under the terms of the [GNU General Public License](#).

The current version of GnuPG is 2.2.20. See the [download](#) page for other maintained versions.

[Gpg4win](#) is a Windows version of GnuPG featuring a context menu tool, a crypto manager, and an Outlook plugin to send and receive standard PGP/MIME mails. The current version of Gpg4win is 3.1.11.

RECONQUER YOUR PRIVACY

Arguing that you don't care about the right to privacy because you have nothing to hide is no different from saying you don't care about free speech because you have nothing to say. – Edward Snowden

Using encryption helps to protect your privacy and the privacy of the people you communicate with. Encryption makes life difficult for bulk surveillance systems. GnuPG is one of the tools that Snowden used to uncover the secrets of the NSA.

Please visit the [Email Self-Defense](#) site to learn how and why you should use GnuPG for your electronic communication.

Ehrenwörtliche Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit mit dem Thema: *Integration einer Container-Umgebung in einen automatisierten „Deployment“-Prozess und die Untersuchung ihrer Effekte auf diesen* selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Karlsruhe, 08.05.2020

Yves Torsten Staudenmaier