Nane KRATZKE

CLOUD-NATIVE COMPUTING

Software Engineering von Diensten und Applikationen für die Cloud

HANSER

HANSER

Nane Kratzke

Cloud-native Computing

Software Engineering von Diensten und Applikationen für die Cloud

Der Autor:

Prof. Dr. rer. nat. Dipl.-Inform. Nane Kratzke
Technische Hochschule Lübeck, Fachbereich Elektrotechnik und Informatik

Bibliografische Information der Deutschen Nationalbibliothek:
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über *http://dnb.d-nb.de* abrufbar.

© 2022 Carl Hanser Verlag München, *www.hanser-fachbuch.de*
Lektorat: Sylvia Hasselbach
Copy editing: Sandra Gottmann, Wasserburg
Umschlagdesign: Marc Müller-Bremer, *www.rebranding.de*, München
Umschlagrealisation: Max Kostopoulos
Titelmotiv: © Max Kostopoulos

Layout: Manuela Treindl, Fürth
Ausstattung patentrechtlich geschützt. Kösel FD 351, Patent-Nr. 0748702

Print-ISBN: 978-3-446-46228-1
E-Book-ISBN: 978-3-446-47284-6
epub-ISBN: 978-3-446-47285-3

Inhalt

Vorwort

Dieses Buch basiert auf zwei Vorlesungen, „*Cloud-native Programmierung*“ und „*Cloud-native Architekturen*“, die ich an der Technischen Hochschule Lübeck gebe. Während der Recherchen für diese beiden Hochschulmodule war ich natürlich auch auf der Suche nach geeigneter Literatur. Das Resultat war ein Literaturumfang, der – auf einem Schreibtisch gestapelt – leider mehr als einen halben Meter Höhe eingenommen hätte.

Meine Recherche mag unzureichend oder meine Anforderungen zu spezifisch gewesen seien, aber ich fand leider nicht die eine oder zwei geeigneten Quellen, die man jemandem als Lehrbuch zum Thema Cloud-native Computing hätte empfehlen und an die Hand geben können; nur eben diesen *Bücherstapel*. Diese Literaturliste hätte mir aber vermutlich diverse kritische Blicke meiner Studentinnen und Studenten eingebracht. Auch wenn ich grundsätzlich kein Freund des Prinzips „*Setze dich zwischen zweier Bücher Mitte und schreib das Dritte*“ bin, war genau dies in diesem Fall der Anstoß zum Schreiben eines ersten Skripts, aus dem letztlich dieses Buch für die beiden oben genannten Lehrveranstaltungen entstanden ist.

Dieses Buch hat somit auch einen gewissen Handbuch-Charakter, auch wenn es kein Handbuch im klassischen Sinne ist. Es kann

dennoch bis zu einem gewissen Grad als Nachschlagewerk genutzt werden, da es eine Vielzahl an hervorragender – aber eben leider isolierter – Literatur zum Thema Cloud-native Computing zusammenfasst.

Ich möchte mich an dieser Stelle u.a. bei Dr. Josef Adersberger von der QAware GmbH bedanken, der eine ähnliche Publikationsidee hatte, dann aber letztlich keine Zeit fand, sein Projekt auch umzusetzen, und der mich daraufhin mit dem Hanser Verlag in Kontakt brachte, um es an seiner Stelle zu versuchen. Zu danken ist auch seinen Mitarbeitern. Deren auf GitHub bereitgestellte Vorlesungsunterlagen „Cloud Computing“ (Adersberger u.a. 2018) waren insbesondere für den Teil II dieses Buchs wertvolle Inspiration und Gliederungshilfe. Dank gebührt daher auch dem Hanser Verlag und hier vor allem Sylvia Hasselbach, die sich auf diese Kontaktvermittlung und das damit einhergehende Wagnis denn auch eingelassen hat und insbesondere in der Produktionsphase viel Unterstützung geleistet hat.

Besonderer Dank gebührt auch meinen Studierenden, die die undankbare Betatester-Rolle für die praktischen Anteile (Labs) dieses Buchs übernommen haben und mir während der – aufgrund Corona leider nur online stattfindenden – Vorlesungen und Praktika dennoch mit vielen wertvollen Rückmeldungen geholfen haben, die Struktur und den Inhalt des Manuskripts für die anvisierte Zielgruppe zu optimieren. Dabei sind insbesondere Jannik Kühnemundt, Felix Lohse, Lucian Schultz und Jana Schwieger zu nennen, die mehrere vertiefende Labs entwickelt und für Folgejahrgänge zur Verfügung gestellt haben.

Lübeck, im Oktober 2021

Nane Kratzke

1 Einleitung

In Zeiten des digitalen Wandels ist Cloud Computing heute mehr und mehr die primäre Option und nicht mehr nur eine von vielen technischen Möglichkeiten. Insbesondere Start-ups wählen kaum noch den Weg über den Aufbau „klassischer“ On-Premise-Lösungen (Rechenzentren). Insbesondere in frühen Phasen eines Unternehmens wird der Aufbau eigener Serverkapazitäten als zu kapital- und personalintensiv empfunden.

Der weltweite Markt für Public Cloud Services wächst folgerichtig Jahr um Jahr; 2019 beispielsweise um etwa 17 Prozent auf insgesamt mehr als 214 Milliarden Dollar. Das am schnellsten wachsende Marktsegment sind hierbei Infrastruktur-Dienste (IaaS) und Plattform-Dienste (PaaS). 70 % dieses Marktes teilen sich dabei die „sogenannten“ Big Five. Dies waren 2020 Amazon, Microsoft, Alibaba, Google und IBM, also alles nichteuropäische Anbieter. Diese sogenannten Hyperscaler unterliegen rechtlichen Regularien ihrer Heimatländer. Amazon, Microsoft und Google unterliegen beispielsweise dem US CLOUD Act. Der CLOUD Act verpflichtet Provider, US-Behörden Zugriff auf gespeicherte Daten zu gewähren, auch wenn die Speicherung nicht in den USA erfolgt. Alibaba unterliegt den Regularien der Volksrepublik China mit vergleichbaren staatlichen Regularien. Solche

Regularien decken sich nicht notwendig mit europäischen oder nationalen Datenschutz-Auffassungen und -Interessen.

Auf europäische Initiative versucht man daher seit 2020, mit GAIA-X eine wettbewerbsfähige, sichere und vertrauenswürdige Dateninfrastruktur der „nächsten Generation“ für Europa aufzubauen, die eine „Datensouveränität“ gewährleisten soll. Dabei sollen insbesondere branchenübergreifende Kooperationen unterstützt werden, um faire und transparente Geschäftsmodelle zu fördern. Flankiert wird dies durch Regeln und Standards für kooperative und rechtskonforme Nutzung von Daten. Solche gemeinsamen Modelle und Regeln sollen die Komplexität und die Kosten der Kommerzialisierung von Daten reduzieren und deren Rechtskonformität erhöhen. Der europäische Ansatz ist also – im Vergleich zum aktuell von wenigen großen US-Hyperscalern dominierten Cloud-Computing-Markt – ein deutlich dezentralerer und kooperativerer Ansatz. Ob dieser Ansatz zu einer umfangreicheren Standardisierung und besseren Interoperabilität von Cloud-Diensten führt, muss die Zukunft allerdings erst noch zeigen. Es wird schwierig werden, in einer sehr agilen, bottom-up geprägten und lösungsorientierten Industrie durchaus fehlende und berechtigte Top-down-Standardisierungs-, Interoperabilitäts- und Datenschutzstrategien zu verankern.

Man kann daher dem Cloud-Computing-Hype durchaus kritisch gegenüberstehen und sich Fragen stellen; zum Beispiel: Ist der GAIA-X Ansatz erfolgversprechend? Wie kritisch ist die Dominanz von US-Hyperscalern? Sollte man die Datenverarbeitung Firmen anvertrauen, die Regularien von nicht demokratisch geprägten Staaten wie der Volksrepublik China unterworfen sind? Solche und ähnliche Fragen sind durchaus berechtigt.

Cloud Computing bleibt aber dennoch heute für viele Unternehmen, Organisationen, Behörden und Forschungseinrichtungen aus rein pragmatischen Gründen die primäre Option, digitalisierte Lösungen (schnell) realisieren zu können. Die Pandora ist nun einmal aus der Büchse. Cloud Computing wird nicht einfach wieder verschwinden. Die Corona-Krise hat dies noch einmal eindrucksvoll gezeigt. Cloud-basierte Lösungen waren in vielen Bereichen die „Strohhalme“, mit denen ganze Volkswirtschaften im Lockdown irgendwie den Kopf über Wasser halten konnten (Kratzke 2020). Man stelle sich nur einmal vor, was passiert wäre, wenn die Corona-Krise 30 Jahre vorher ausgebrochen wäre! Wie hätten wir uns dann im Homeoffice organisiert? Wäre Homeoffice überhaupt möglich gewesen? Es hätte definitiv keine hochskalierbaren Videokonferenzlösungen von Anbietern wie Zoom, Azure Teams oder Google Meet gegeben, die weltweit in kürzester Zeit mit unglaublichen Rechenressourcen hinterlegt werden konnten.

Dieses „Primäre“ hat mittlerweile sogar einen Namen bekommen. Man nennt es „Cloud-native“ (Kratzke und Quint 2017; Kratzke 2018). Als Cloud-native Systeme werden in diesem Buch verteilte Systeme bezeichnet, die bewusst für Cloud-Infrastrukturen und Cloud-Plattformen entwickelt werden und nur effektiv in diesen betreibbar sind. Da diese Systeme primär variable Kosten durch ihren Ressourcenverbrauch generieren (Pay-as-you-go-Kostenmodell, siehe auch Abschnitt 2.3), haben sich Designmuster entwickelt, wie solche Systeme möglichst kostengünstig (d. h. ressourcenschonend) betrieben werden können. Dies hat massiven Einfluss auf die verwendeten Technologien, aber auch auf Architekturen genommen, mit denen solche Cloud-nativen Systeme entwickelt und betrieben

werden. Um diese Hintergründe und Mechanismen Cloud-nativer Systeme soll es in diesem Buch gehen.

1.1 An wen sich dieses Buch richtet

Dieses Buch richtet sich insbesondere an Studierende in Informatik- oder verwandten (Master-)Studiengängen. Gleichermaßen adressiert es Dozenten, die derartige Themen im Hochschul- oder beruflichen Weiterbildungskontext vermitteln. Insbesondere die ergänzenden Online-Materialien sind für diese Zielgruppe von Interesse. Aber auch Autodidakten oder IT-Seiteneinsteiger, die im IT-Umfeld Funktionen im weiteren Bereich der Softwareentwicklung innehaben oder diese anstreben, werden adressiert.

Vor dem Hintergrund dieser Zielgruppen werden Themen des Cloud-native Computings mit dem Ziel, einen soliden und kritisch einordnenden Überblick zu geben, überwiegend auf Einsteigerniveau behandelt. Allerdings werden fundiertes softwaretechnisches Basiswissen und Programmierkenntnisse in mindestens einer prozeduralen oder objektorientierten Programmiersprache vorausgesetzt. Idealerweise ist es Python, aber andere Programmiersprachen sind auch gut anzuwenden. Ferner sind Erfahrungen mit unixoiden Betriebssystemen wie beispielsweise Linux von Vorteil.

Dieses Buch richtet sich auch an technische Projektleiter, Berater, Softwarearchitekten und -entwickler, die Cloud-native Technologien und korrespondierende DevOps-Praktiken in Projekten oder Abteilungen einführen wollen oder damit erste Erfahrungen sammeln.

Teil I ist auch für (bzw. die Kommunikation mit) C-Level-Funktionen (z. B. CIO, CTO) in Unternehmen geschrieben worden und kann Softwareentwicklern Argumente liefern, um Cloud-basierte Entwicklungsansätze kritisch, konstruktiv und lösungsorientiert mit Unternehmensleitungen zu diskutieren.

1.2 Was dieses Buch behandelt

Das Buch ist in drei Teile gegliedert, die sich mit unterschiedlichen Bereichen des Cloud-native Computings befassen. Dies erstreckt sich von den theoretischen und konzeptionellen Grundlagen des Cloud Computings über praktisches „Hands-on"-Wissen der Basistechnologien bis hin zu architekturellen Überlegungen und dem verlässlichen Betrieb großer und komplexer Cloud-nativer Systeme.

In **Teil I** werden primär die theoretischen und konzeptionellen **Grundlagen** des Cloud Computings behandelt.

- **Kapitel 2** geht auf die bekannten **Service-Modelle** IaaS, PaaS und SaaS ein, die sich seit mehr als einer Dekade als eine feste konzeptionelle Gliederung im Cloud Computing bewährt haben. Behandelt wird ferner die „Cloud-Ökonomie" und deren Einfluss auf den Technologiestack und Architekturen von Cloud-nativen Systemen, die wir heutzutage vorfinden.
- **Kapitel 3** geht auf **DevOps** als eine treibende Philosophie des Cloud Computings ein. Insbesondere die DevOps-Prinzipien des Flow motivieren dabei den **Teil II** **(Everything as Code)** dieses Buches. Die DevOps-Prinzipien des

Feedbacks legen die Grundlage für den **Teil III** **(Observable Architectures)**. Das Kapitel 3 kann also durchaus als gedankliche Klammer gesehen werden.

- Das **Kapitel 4** geht auf den Begriff **Cloud-native** an sich ein, um diesen Begriff präzise zu fassen und als Leitmotiv für dieses Buch aufzugreifen. Dieses Kapitel bildet damit den Einstieg in **Teil II**.

Teil II betrachtet das „Handwerk“ der Cloud-nativen Programmierung. Die Basisfähigkeiten des **Everything as Code** bilden die Grundlage für Cloud-native Systementwicklung größerer Systeme. Das Buch überträgt hier das bewährte Vorgehen der Informatikausbildung vom Programming-in-the-Small zum Programming-in-the-Large auf Cloud-native Systeme. In der Informatikausbildung werden üblicherweise erst die Basisfertigkeiten des Programmierens im Kleinen vermittelt und erst anschließend die Fertigkeiten und Feinheiten der Erstellung von Softwarearchitekturen für größere und komplexere Softwaresysteme. Andernfalls würden Softwarearchitekturen vermutlich als vollkommen realitätsferne und theoretische „Software-Philosophie“ abgetan werden. Das Buch verfolgt also ganz bewusst einen Bottom-up-Ansatz und keinen der häufig anzutreffenden Top-down-Ansätze, die sich oft auf architekturelle Aspekte wie Microservices und Serverless-Architekturen fokussieren, aber im Rahmen dessen dann nur recht isoliert technische Einzelkomponenten tiefer liegender Ebenen wie Container, Functions, Orchestrierungsplattformen, Deployment-Pipelines, Infrastruktur Provisioning usw. widmen.

- **Kapitel 6** befasst sich dazu mit der softwaredefinierten Entwicklung von **Deployment-Pipelines**, die das Herzstück

der Automatisierung in vielen DevOps-Ansätzen von Cloud-nativen Systemen bilden.

- In **Kapitel 7** wird Infrastructure as Code, also die automatisierte Bereitstellung softwaredefinierter Infrastrukturen, behandelt.
- **Kapitel 8** erläutert, wie Komponenten (inklusive aller Abhängigkeiten) Cloud-nativer Systeme als standardisierte Deployment Units gebaut und bereitgestellt werden können.
- **Kapitel 9** zeigt, wie aus vielen feingranularen Containern bestehende Cloud-native Systeme mittels Cluster-Technologien orchestriert und mittels Self-Healing-Mechanismen betrieben werden können.
- **Kapitel 10** geht auf noch kleinere und lastabhängig skalierende Einheiten (Functions) ein, die die darunterliegende Infrastruktur im Sinne der sogenannten Serverless-Philosophie sogar komplett wegabstrahieren können.

In **Teil III** geht es um die architekturelle Ebene von Cloud-nativen Systemen. Dabei werden Architekturansätze und Methodiken vorgestellt, wie Cloud-native Systeme und Anwendungen gebaut werden können, die gemäß den im Kapitel 2 gezeigten ökonomischen Gesetzmäßigkeiten kosteneffektiv betrieben und evolutionär weiterentwickelt werden können.

- **Kapitel 12** befasst sich hierzu mit den beiden das Cloud-native Umfeld prägenden Architekturstilen Microservices und Serverless Architectures.
- In **Kapitel 13** geht es hingegen stärker um den Betrieb und die Beobachtbarkeit von Cloud-nativen Anwendungen zur Laufzeit. Es werden Themen wie Logging, Monitoring,

Tracing, Service Meshs sowie die Analyse und das Management von Verkehrstopologien behandelt.

- **Kapitel 14** bildet den Abschluss des gewählten Bottom-up-Ansatzes dieses Buchs und befasst sich damit, wie sich Cloud-native Anwendungen unter anderem mittels des Domain Driven Designs methodisch entwerfen lassen.

Teil III schließt mit Tabelle 15.1, in der alle in diesem Buch behandelten Pattern und Best Practices zur Entwicklung Cloud-nativer Anwendungen und Dienste aufgeführt sind. Dadurch finden sich insbesondere für die eher „nachschlagenden“ Leser relevante Stellen gegebenenfalls etwas zielgerichteter, als das rein über das Inhaltsverzeichnis möglich wäre. Ohne ein gewisses Überblickswissen ist Tabelle 15.1 aber vermutlich von eher geringerem Wert.

1.3 Sprachliche Konventionen

Insbesondere Autoren deutschsprachiger IT-Literatur stellt sich die Frage, ob man sich der englischen Originalterminologie oder deutscher Übersetzungen bedient. Schnell wirkt die „dogmatische“ Nutzung deutscher Begrifflichkeiten sperrig und wenig intuitiv für den Leser. Daher werden nur in (wenigen) Fällen gängige und direkt herleitbare Übersetzungen genutzt. Ansonsten wird der englischen Originalterminologie gefolgt – zumindest immer dann, wenn es um technische Terminologie geht.

Es wird also beispielsweise der Begriff Service genutzt, wenn ein IT-Dienst (z. B. Webdienst) gemeint ist und damit primär die