CLOUD NATIVE COMPUTING

CLOUD-NATIVE

Unit:

Cloud Computing

(4) Implikationen für Cloud-native Systeme

PROF. DR.
NANE KRATZKE

creative commons

CLOUD NATIVE COMPUTING

Urheberrechtshinweise

Diese Folien werden zum Zwecke einer praktikablen und pragmatischen Nutzbarkeit im Rahmen der **CCo 1.0 Lizenz** bereitgestellt.

Sie dürfen die Inhalte also kopieren, verändern, verbreiten, mit eigenen Inhalten mixen, auch zu kommerziellen Zwecken, und ohne um weitere Erlaubnis bitten zu müssen.

Eine Nennung des Autors ist nicht erforderlich (aber natürlich gern gesehen, wenn problemlos möglich).

Diese Folien sind insb. für die Lehre an Hochschulen konzipiert und machen daher vom **§51 UrhG (Zitate)** Gebrauch.

Die CCo Lizenz überträgt sich nicht auf zitierte Quellen. Hier sind bei der Nutzung natürlich die Bedingungen der entsprechenden Quellen zu beachten.

Die Quellenangaben finden sich auf den entsprechenden Folien.

KAPITEL 2 + 4

Cloud Computing und Cloud-native

2 Cloud Computing

2.1 Service Modelle

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Software as a Service (SaaS)

2.2 Cloud-Ökonomie

- Eignung von unterschiedlichen Arten von Workloads
- Effekt von Zuteilungsdauer und Ressourcengröße

2.3 Entwicklung der letzten Jahre

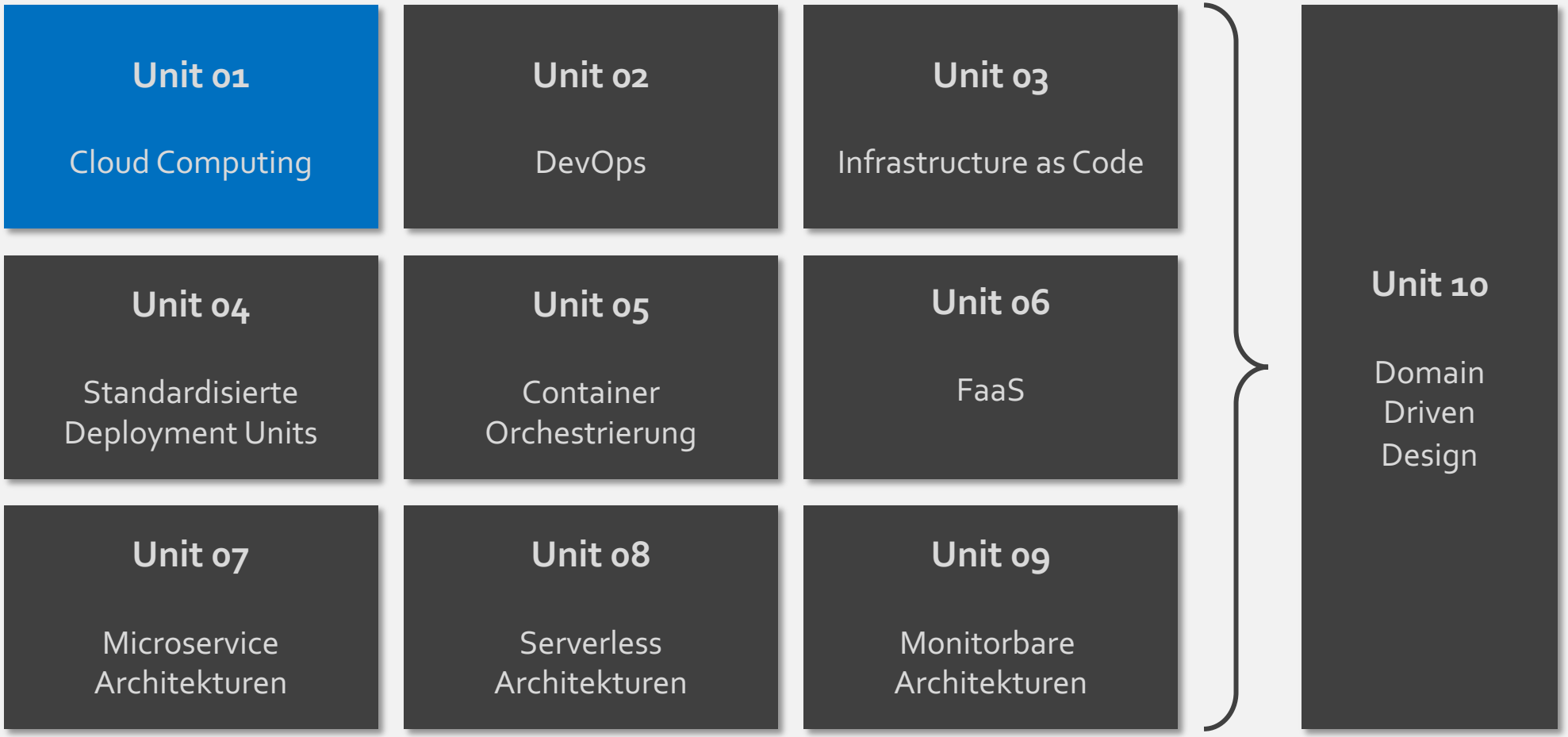
4 Cloud-native

- Definitionen in Industrie und Forschung
- Die Cloud-native Definition dieses Buchs
- Zusammenfassung und Ausblick auf Teil II bis IV

CLOUD NATIVE COMPUTING

INHALTSVERZEICHNIS

Überblick über Units und Themen dieses Moduls



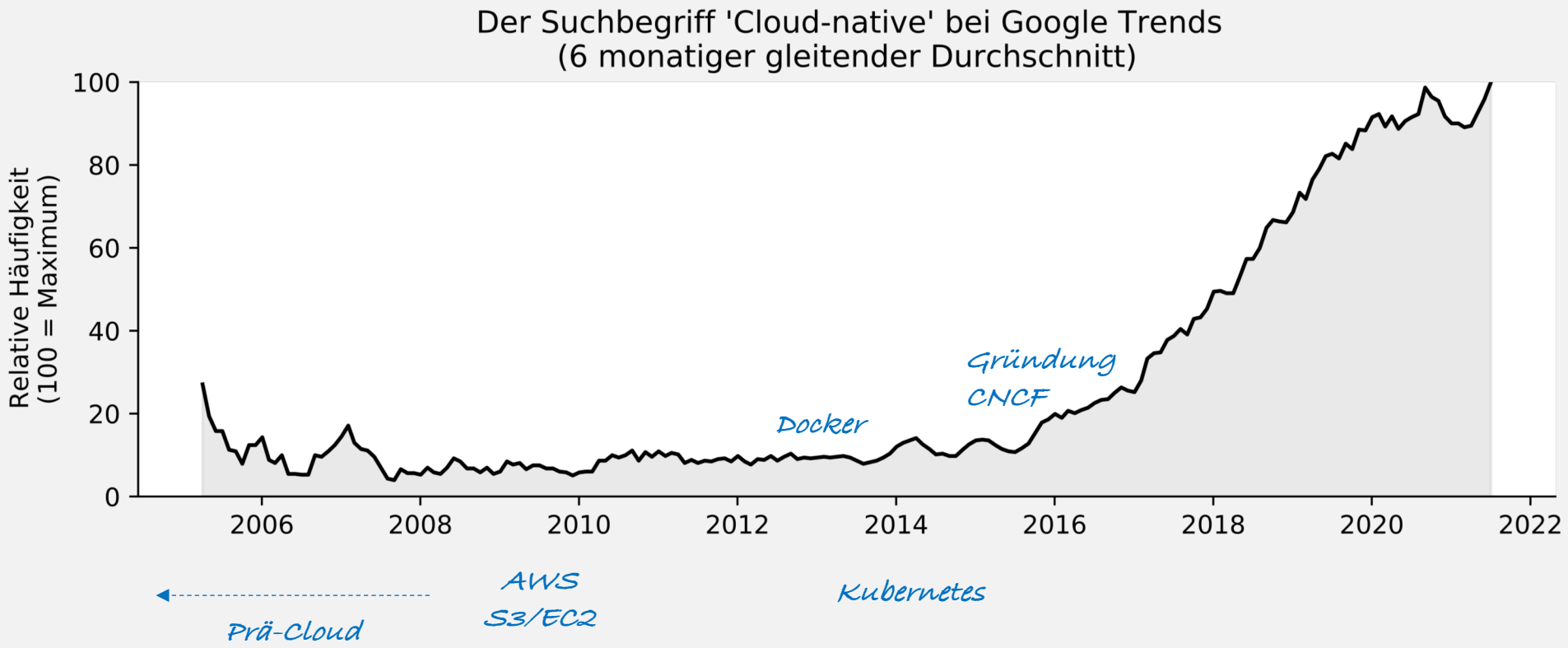
INHALTE

- Cloud Computing
- Cloud-Ökonomie (Crash-Kurs)
- Eine kurze Geschichte der Cloud
- **Implikationen für die Entwicklung und den Betrieb von Cloud-nativen Anwendungen und Dienste**

CLOUD NATIVE COMPUTING

DER BEGRIFF „CLOUD-NATIVE“

Im Verlaufe der Zeit



CNCF =
Cloud Native
Computing Foundation

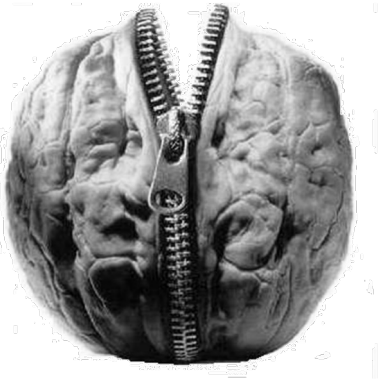
ANMERKUNGEN FÜR IT-MANAGER:INNEN

Cloud-native in a Nutshell (CNCF)

- Entwicklung von Anwendungen mittels standardisierter SW-Komponenten (Container)
- Kombination aus agilem SW-Entwicklungsprozess, DevOps-Praktiken und standardisierten Infrastruktur-/Plattform-Technologien
- Anwendungen so gestalten und betreiben, dass diese für den Betrieb in der Cloud optimiert sind

Die wesentlichen Bausteine sind:

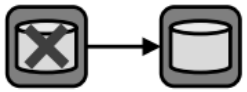
- **Containerisierung** (Software-Komponenten standardisiert ausführen)
- **Orchestrierung** (Software-Komponenten automatisch bereitstellen, aktualisieren, skalieren, überwachen und „Self-Healing" ausführen)
- **Microservices** (Anwendungen sind aus kleinen, unabhängig deploybaren Services aufgebaut)
- **Infrastructure as Code** (IaC; auch die gesamte Infrastruktur wird in Code definiert, um schnell und konsistent bereitgestellt und verwaltet werden zu können)
- **Automatisierung** von Entwicklungs-, Test- und Bereitstellungsprozessen (Deployment-Pipelines ermöglichen schnellere und zuverlässigere Software-Updates)



CLOUD NATIVE COMPUTING

CLOUD-NATIVE APPLIKATIONEN

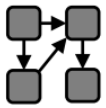
Das IDEAL-Modell (Von Service-orientierten Architekturen inspiriert)



I solated State

Ein Maximum an Komponenten der Cloud-Anwendung sollte zustandslos designed sein.

Die meisten Komponenten sollten weder Kommunikations- (Session States) noch Anwendungs-Zustände (Application States) verwalten.



D istribution

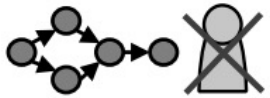
Cloud-Anwendungen werden in mehrere Komponenten aufgeteilt, um mehrere Cloud-Ressourcen nutzen zu können, da „die Cloud" selbst ein großes verteiltes System ist.

E lasticity

Anwendungen werden horizontal skaliert, indem die Anzahl der Ressourcen angepasst wird (scale out).

Scale out (horizontal): Hinzufügen weiterer Ressourcen.

Scale up (vertikal): Verbesserung der vorhandenen Ressourcen.

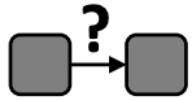


A utomated

Orchestrierung wird mittels automatischer Prozesse schnell erledigt.

Beispiel:

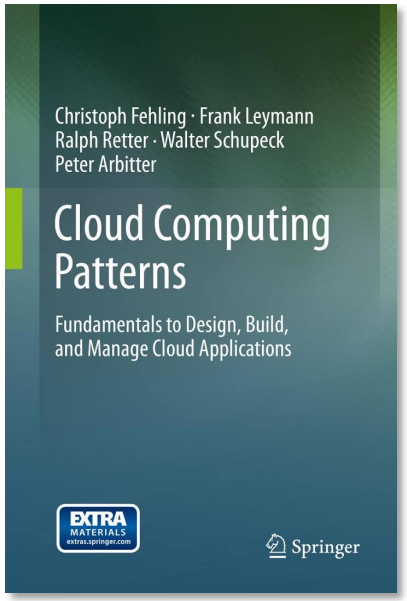
- Bei geringer Last automatisches herunterfahren von Servern.
- Beisteigernder Last automat. anfahren von Servern.



L oose Coupling

Cloud-native application components should not influence each other regarding factors such as availability, data format, data exchange rate.

Example:
Failure of one application component does not cause failure of other components.



CLOUD-NATIVE APPLIKATIONEN

Definition

CNCF Cloud Native Definition v1.0

Cloud native Technologien ermöglichen es Unternehmen, skalierbare Anwendungen in modernen, dynamischen Umgebungen zu implementieren und zu betreiben. Dies können öffentliche, private und Hybrid-Clouds sein. Best-Practices, wie Container, Service-Meshs, Microservices, immutable Infrastruktur und deklarative APIs, unterstützen diesen Ansatz.

Die zugrundeliegenden Techniken ermöglichen die Umsetzung von entkoppelten Systemen, die belastbar, handhabbar und beobachtbar sind. Kombiniert mit einer robusten Automatisierung können Softwareentwickler mit geringem Aufwand flexibel und schnell auf Änderungen reagieren.

Source: Cloud-native Computing Foundation, https://github.com/cncf/toc/blob/master/DEFINITION.md

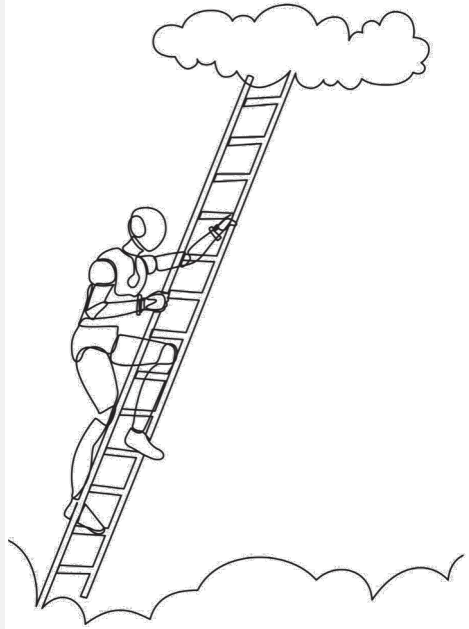
CLOUD NATIVE COMPUTING FOUNDATION

CLOUD NATIVE COMPUTING

CLOUD APPLICATION MATURITY MODEL

In Anlehnung an Open Data Center Alliance

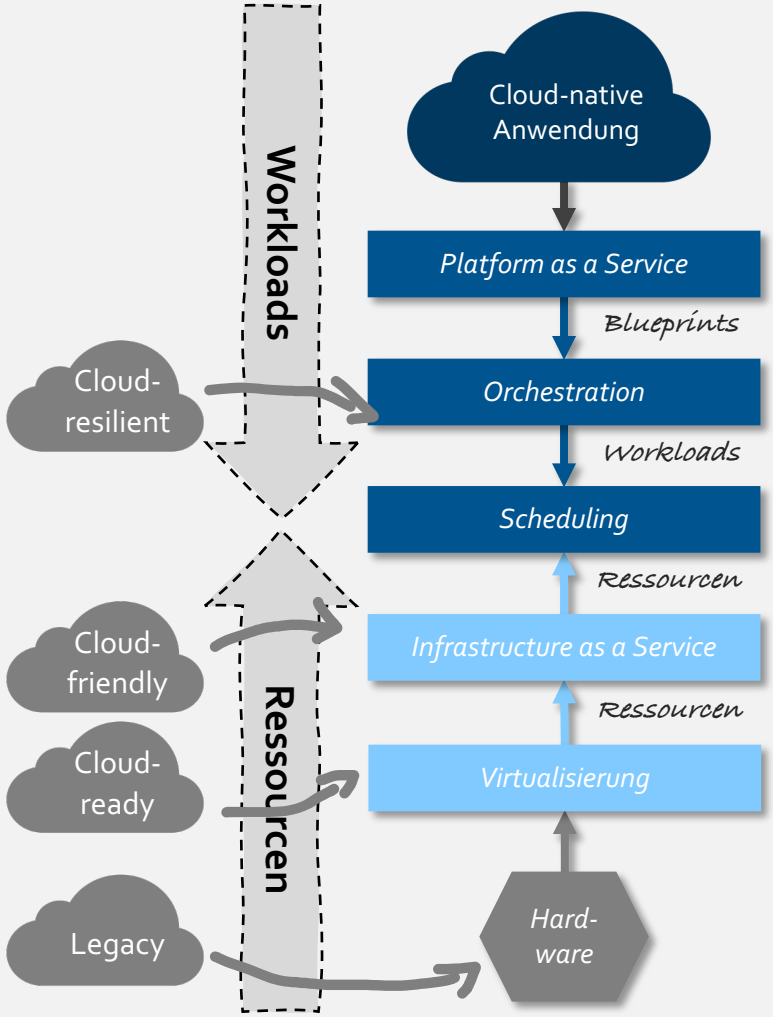
Level	Maturity	Description
3	Cloud native	○ **Automatically scale** out/in based on stimuli ○ **Transferable** across infrastructures at runtime without interruption of services
2	Cloud resilient	○ Composed of **loosely coupled** services ○ Services are designed according to **cloud design patterns** ○ **Infrastructure agnostic** ○ **Unaffected by** dependent service **failures** ○ Isolated state (distinction of stateless and stateful services)
1	Cloud friendly	○ **Composed of services** ○ Services are discoverable by name ○ **Deployable via** Infrastructure (IaaS) **API**
0	Cloud ready	○ Operated on virtualized infrastructure ○ **Deployable from** image or **script**



Quelle: Kratzke, N. A Brief History of Cloud Application Architectures. Appl. Sci. 2018, 8, 1368.

TECHNOLOGIE - EBENEN

Oder: Wie kommt die Software an das Blech?



Bereitstellung einer Betriebsplattform für Anwendungskomponenten
(zunehmend in Form von Containern standardisiert)

Current vs. Desired State Control Loops
(automat. Erkennung und Kompensation von unbeabsichtigten Zuständen von Anwendungskomponenten)

Zuordnung von Workloads (Anwendungsinstanzen) zu virtuellen Ressourcen

Bereitstellung virtueller Ressourcen (Self-Service, ggf. mittels IaC automatisiert)

Virtualisierung und Isolation von physischen Ressourcen

Bereitstellung und Betrieb physischer Ressourcen (Rechenzentrum)

CLOUD NATIVE COMPUTING

Beispiel-Produkte (Open Source)

APACHE OpenWhisk

kubernetes

openstack.

CLOUD-NATIVE

Die Begriffsdefinition für dieses Modul

CLOUD NATIVE COMPUTING

Definition

Eine Cloud-native Anwendung (CNA) ist ein **verteiltes**, **beobachtbares**, **elastisches** und auf horizontale **Skalierbarkeit** optimiertes **Service-of-Services**-System, das seinen Zustand in (einem Minimum an) **zustandsbehafteten Komponenten** isoliert.

Die Anwendung und jede in sich geschlossene **Bereitstellungseinheit** dieser Anwendung wird nach Cloud-fokussierten Designmustern entworfen und auf elastischen Self-Service-Plattformen betrieben.

Quelle: Kratzke, N. Cloud-native Computing, 2021, Hanser Verlag.

CLOUD-NATIVE

Die Begriffsdefinition für dieses Modul

CLOUD NATIVE COMPUTING

Beobachtbarkeit

Beobachtbarkeit bei Softwaresystemen bezieht sich typischerweise auf Telemetriedaten, die meist in drei Aspekte unterteilt werden: **Tracing** (verteilte Ablaufverfolgung von Transaktionen) ,**Monitoring** (Metriken, quantitative Informationen zu Prozessen) **Logging** (Protokollierung von Systemereignissen).

Elastizität

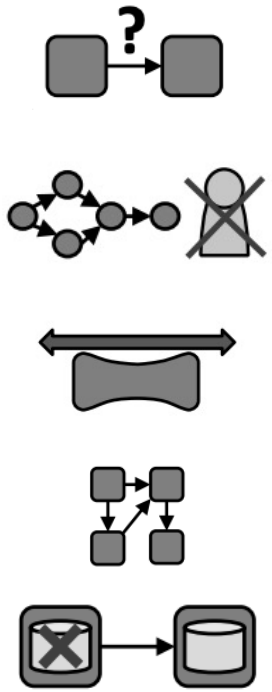
ist der Grad, in dem ein System in der Lage ist, sich an Workload-Änderungen anzupassen, indem es Ressourcen in einer autonomen Art und Weise provisioniert und deprovisioniert, sodass zu jedem Zeitpunkt die verfügbaren Ressourcen so gut wie möglich mit dem aktuellen Bedarf übereinstimmen.

Skalierbarkeit

Strukturelle Skalierbarkeit ist die Fähigkeit eines Systems, sich in einer gewählten Dimension ohne größere Änderungen an seiner Architektur zu erweitern. Unter **Lastskalierbarkeit** ist die Fähigkeit eines Systems zu verstehen, auch steigenden Datenverkehr bewältigen zu können.

Service-of-Services

Bei Cloud-nativen Anwendungen werden einzelne Anwendungen als eine Suite kleiner loser gekoppelter Services (**Microservices**) dekomponiert, die jeweils in einem eigenen Prozess laufen. Diese Services können unabhängig voneinander **vollautomatisch aktualisiert** werden.



Standardisierte Bereitstellungseinheiten (Container)

Container können eine Softwarekomponente und alle ihre Abhängigkeiten in einem Format kapseln, das **selbstbeschreibend** und portabel ist, sodass jede konforme Laufzeitumgebung sie **ohne zusätzliche Abhängigkeiten** ausführen kann.

Zustand (Statefulness)

Zustandsbehaftete Komponenten (**Datenbanken**) werden für mehrere Instanzen einer skalierten Anwendungskomponente verwendet, die ihren internen Zustand synchronisieren, um ein einheitliches Verhalten zu bieten. Da die Skalierung zustandsbehafteter Komponenten meist aufwendiger ist als die Skalierung zustandsloser Komponenten, versucht man, Zustände in möglichst wenigen zustandsbehafteten Komponenten zu **isolieren**.

ANMERKUNGEN FÜR ENTWICKLER:INNEN + ADMINS

Cloud Computing is a Mindset Change: PETS vs. Cattle

Alte Informatiker-Weisheit:
Ausschalten, Einschalten, geht wieder.

Server sind Wegwerf-Artikel!
Wegschmeißen - neu machen, ist schneller als Problem suchen, Problem beheben.

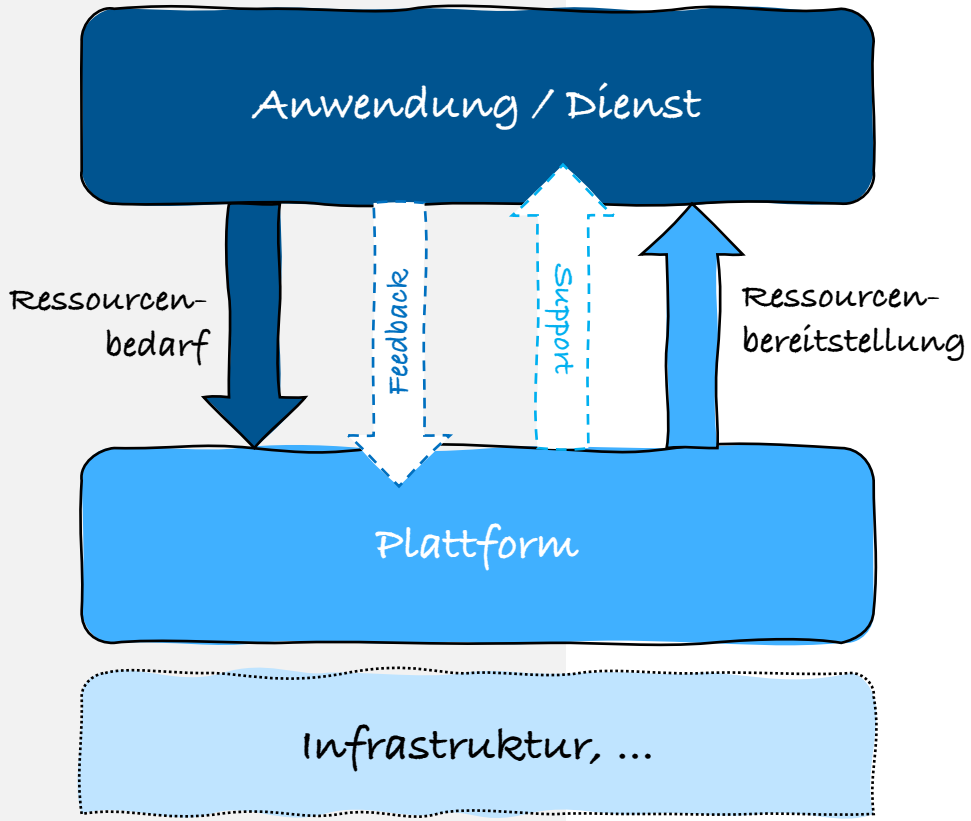
Mehr Power,
kommt durch mehr Server.

ANMERKUNGEN FÜR IT-MANAGER:INNEN

CLOUD NATIVE COMPUTING

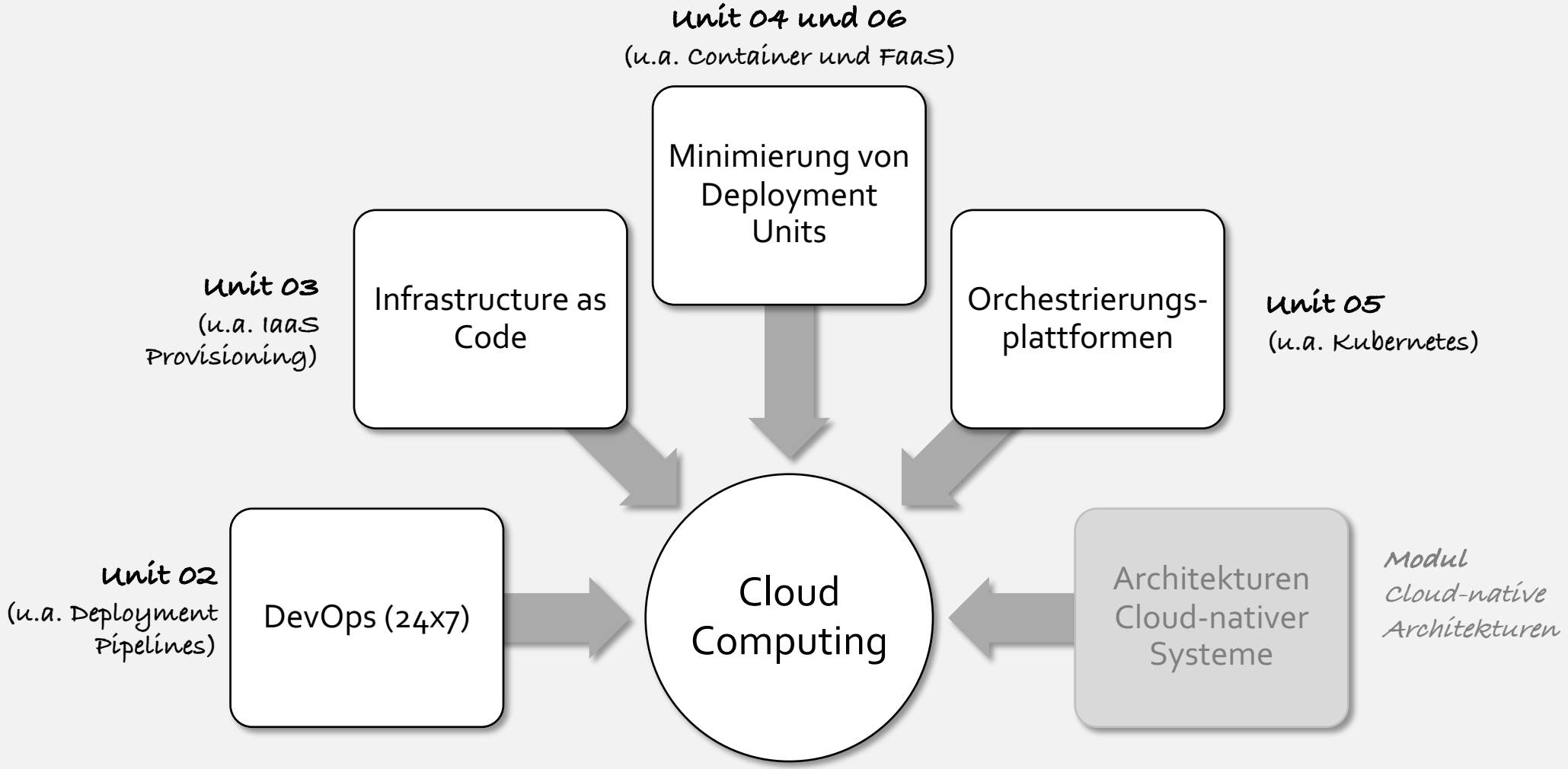
Cloud-natives Denken

- **… fokussiert nicht das Bereitstellungsmodell**
  (und funktioniert daher sowohl mit Private, Community, Hybrid oder Public Clouds)
- **… standardisiert den Betrieb von Anwendungen**
  (mittels Plattformen, Containern und Self-Healing fähigen automatisierten Betriebsprozessen, Orchestrierung)
- **… fördert die Agilität und You-Build-It-You-Run-It Ansätze**
  (mittels einer Self-Service Philosophie, Entkopplung von Anwendungs- und Plattform-IT, sowie stärkerer Einbindung von Fachabteilungen in die Service-Leistungserbringung)
- **… harmoniert besser mit einem produkt- als mit einem projektorientierten Management!**



AUSBLICK

Auf die Module Cloud-native Programmierung und Cloud-native Architekturen



KONTAKT

Disclaimer

Nane Kratzke

+49 451 300-5549

nane.kratzke@th-luebeck.de

kratzke.mylab.th-luebeck.de