
CLOUD NATIVE COMPUTING

# CLOUD-NATIVE

***Unit:***

## *Infrastructure as Code*

*(4) IaC–Ansätze und Tools: Vagrant*

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 7

*Infrastructure as Code*


## 7 Infrastructure as Code

### 7.1 Virtualisierung

- Virtualisierung von Hardware-Infrastruktur
- Virtualisierung von Software-Infrastruktur

### 7.2 Provisionierung

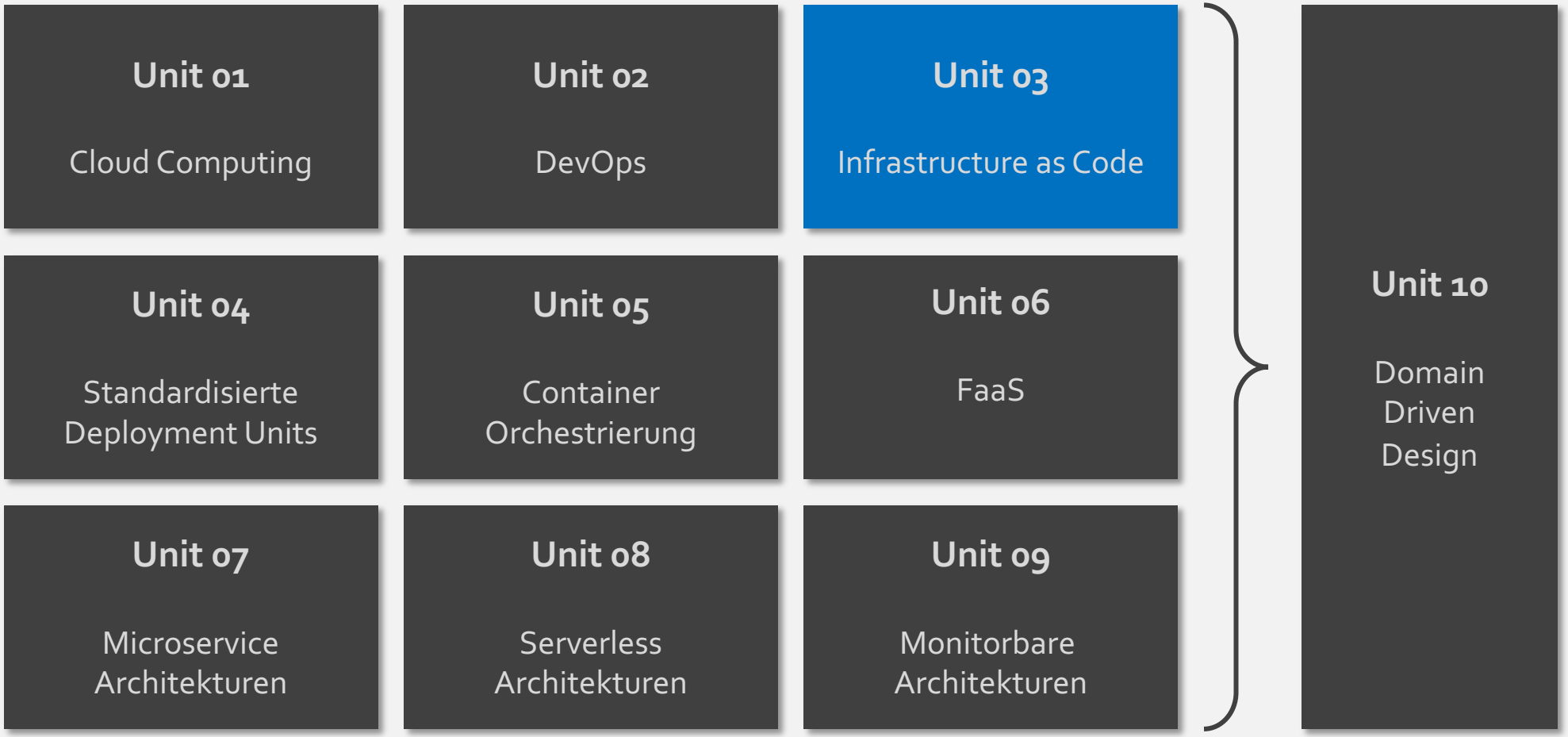
- Immutable Infrastructure
- Infrastructure as Code
- Provisionierung von lokalen Umgebungen
- Provisionierung von Multi-Host Umgebungen

### 7.3 Zusammenfassung

CLOUD NATIVE COMPUTING

# INHALTSVERZEICHNIS

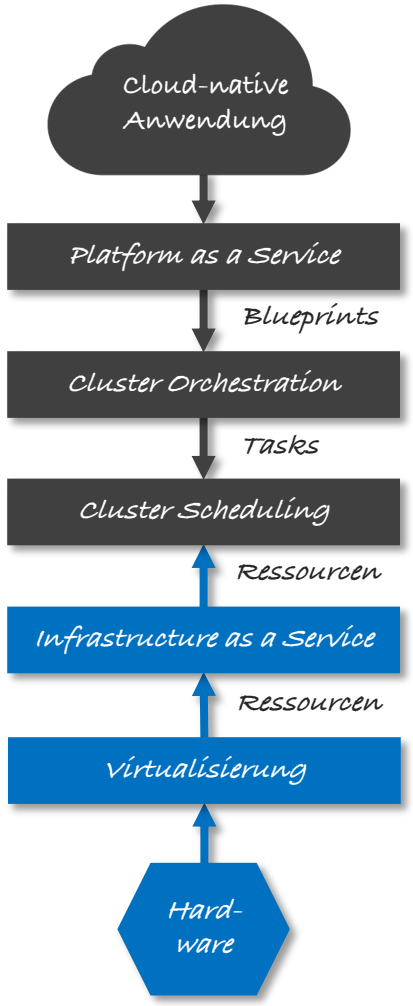
*Überblick über Units und Themen dieses Moduls*



# INHALTE

- Virtualisierung
- Infrastructure as a Service
- Provisionierung in IaaS-basierte Infrastrukturen
- Infrastructure as Code
  - Definition, Ansätze und Methoden
  - Überblick gängiger Provisionierungswerkzeuge
  - Single VM-Provisioning mittels Vagrant
  - Multi VM-Provisioning mittels Terraform

CLOUD NATIVE COMPUTING



# INFRASTRUCTURE AS CODE

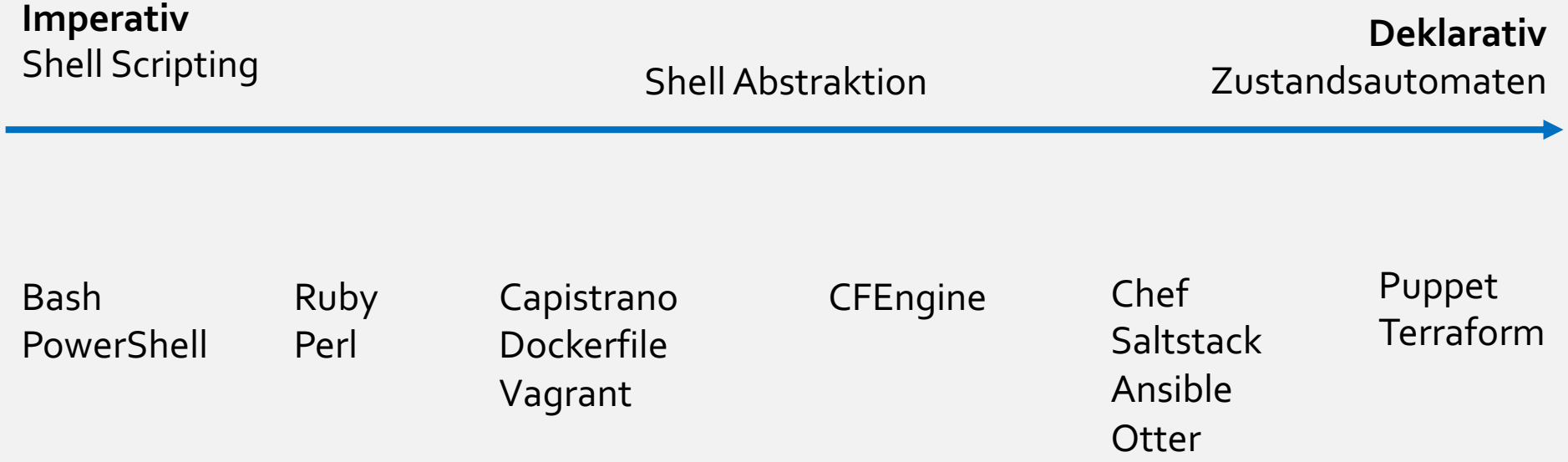
*Was ist das?*

Unter Infrastruktur als Code (IaC) versteht man die Verwaltung und Bereitstellung von IT-Ressourcen mittels maschinenlesbarer Definitionsdateien anstelle von physischer Hardwarekonfiguration oder interaktiven Konfigurationstools.

Mittels IaC verwaltete IT-Infrastruktur kann sowohl physische Geräte wie Bare-Metal-Server als auch virtuelle Maschinen und zugehörige Konfigurationsressourcen umfassen.

# WERKZEUGE

*Übersicht gängiger Provisionierungswerkzeuge*



# INFRASTRUCTURE AS CODE

*Provisionierungs-Strategien und -Methoden*

CLOUD NATIVE COMPUTING

| Welche Provisionierungs-Strategie wird genutzt? | Wie überträgt man eine Konfiguration auf eine Maschine? |
|---|---|
| **Deklarativer Ansatz**<br>• Der deklarative Provisionierungsansatz definiert den gewünschten Zustand.<br>• Das System führt aus, was geschehen muss, um diesen gewünschten Zustand zu erreichen. | **Pull Methode**<br>Bei der Pull-Methode zieht die zu konfigurierende Maschine die Konfiguration vom steuernden Provisionierungsmechanismus. |
| **Imperativer Ansatz**<br>• Der imperative Provisionierungsansatz definiert, wie die Infrastruktur geändert werden soll.<br>• Hierzu werden bestimmte Befehle definiert, die in der richtigen Reihenfolge ausgeführt werden müssen, um im gewünschten Zustand zu enden. | **Push Methode**<br>Bei der Push Methode überträgt der Provisionierungsmechanismus die anzuwendende Konfiguration auf die zu provisionierende Maschine. |

*Hierfür muss auf der zu konfigurierenden Maschine also im Basis Image bereits ein entsprechender Agent installiert sein.*

# INFRASTRUCTURE AS CODE

*Tool-Überblick*

| Tool | Veröffentlicht | Methode | Ansatz | Geschrieben in | Anmerkungen |
|---|---|---|---|---|---|
| CFEngine | Northern.tech **1993** | Pull | deklarativ | C | - |
| Puppet | Puppet **2005** | Pull | deklarativ | Ruby | - |
| Chef | Chef **2009** | Pull | deklarativ + imperativ | Ruby | - |
| Vagrant | HashiCorp **2010** | Push | deklarativ + imperativ | Ruby | - |
| SaltStack | SaltStack **2011** | Push + Pull | deklarativ + imperativ | Python | Mittlerweile VMware |
| Ansible | Red Hat **2012** | Push + Pull | deklarativ + imperativ | Python | - |
| Terraform | HashiCorp **2014** | Push | deklarativ | Go | - |
| Otter | Inedo **2015** | Push | deklarativ + imperativ | .NET | Windows! |

CLOUD NATIVE COMPUTING

*Provider-spezifische Ansätze wie AWS CloudFormation oder Heat (OpenStack) sind hier nicht aufgenommen.*

*Diese können im allgemeinen von den hier gezeigten Tools bespielt werden.*

*Die hier gezeigte Liste erhebt ferner keinen Anspruch auf Vollständigkeit.*

# HARDWARE-VIRTUALISIERUNG

*Immutable Architecture am Beispiel von Vagrant und VirtualBox*


Open Source Voll-Virtualisierung (Typ-2 ) für Windows, Linux, MacOS (x86) und Solaris

**Download:**
`https://www.virtualbox.org`

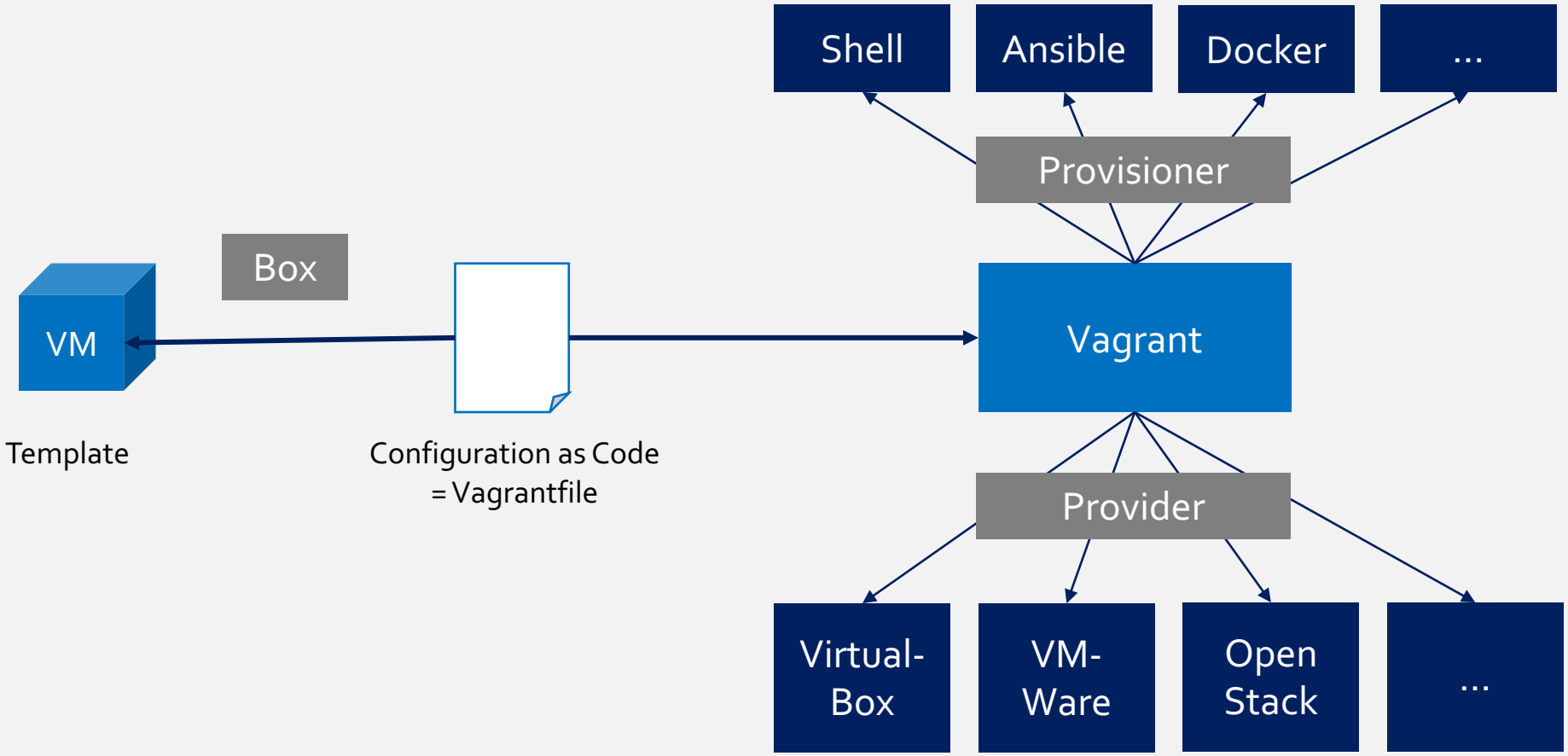

Shell-basierte Automationssoftware für virtuelle Umgebungen auf einem Rechner

**Download:**
`https://www.vagrantup.com/downloads`

*Diese Kombination eignet sich gut für lokale Entwicklungs-maschinen.*

# VAGRANT

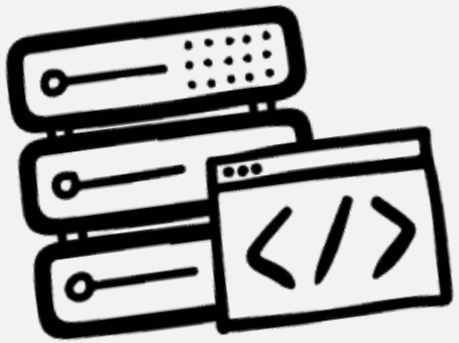
*Konzepte*



# VAGRANTFILE

*Beschreibung einer zu erstellenden Virtuellen Maschine*

```
# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version.
Vagrant.configure("2") do |config|

  # Every Vagrant development environment requires a box. You can search for
  # boxes at https://vagrantcloud.com/search.
  config.vm.box = "ubuntu/jammy64"

  # Create a forwarded port mapping which allows access to a specific port
  # within the machine from a port on the host machine
  config.vm.network "forwarded_port", guest: 80, host: 8888

  # Provider-specific configuration
  config.vm.provider "virtualbox" do |vb|
    # Display the VirtualBox GUI when booting the machine
    # vb.gui = true

    # Customize the amount of memory on the VM:
    vb.memory = "2048"
  end

  # Provisioning
  config.vm.provision "shell", inline: <<-SHELL
    apt-get update
    apt-get install -y nginx
  SHELL
end
```



# VAGRANT

*Typischer Arbeitsablauf*

| # | Befehle | Bedeutung |
|---|---|---|
| 1 | `md <box-dir>`<br>`cd <box-dir>` | Verzeichnis für Vagrant Umgebung erstellen und dorthin wechseln |
| 2 | `vagrant init [<box-name>] [<boxurl>]` | Vagrant Umgebung initialisierung. Es wird eine Datei *Vagrantfile* erstellt und initial mit dem Namen und URL der Box falls angegeben initialisiert. |
| 3 | `code Vagrantfile` | *Vagrantfile* nach Bedarf anpassen (z.B. IP vergeben, Port-Mapping/Verzeichnis-Share-Mapping zwischen Host und Guest, ...) |
| 4 | `vagrant up` | Startet die Virtuelle Maschine aus dem Box-Template und konfiguriert diese entsprechend dem *Vagrantfile* |
| 5 | `vagrant ssh` | Per SSH auf die VM einloggen (vom Host) |
| 6 | `exit` (in SSH Shell des Guest) | Die SSH Shell in der VM verlassen (zurück zum Host) |
| 7 | `vagrant halt` | Die VM stoppen. |

Weitere Kommandos: https://www.vagrantup.com/docs/cli

CLOUD NATIVE COMPUTING

Ggf. hilfreich:

**reload:** Startet eine VM neu und aktualisiert die Konfiguration nach angepasstem *Vagrantfile*

**package:** Erstellt aus einer VM wieder eine Box (Template)

**destroy:** Löscht eine VM

# VAGRANT

*Nur Versuch macht kluch …*


CLOUD NATIVE COMPUTING

**Klonen Sie dieses Repository:**

```
git clone https://git.mylab.th-luebeck.de/cloud-native/lab-local-vm-provisioning.git
```

**NGINX**

```
cp Vagrantfile.nginx Vagrantfile
vagrant up
```

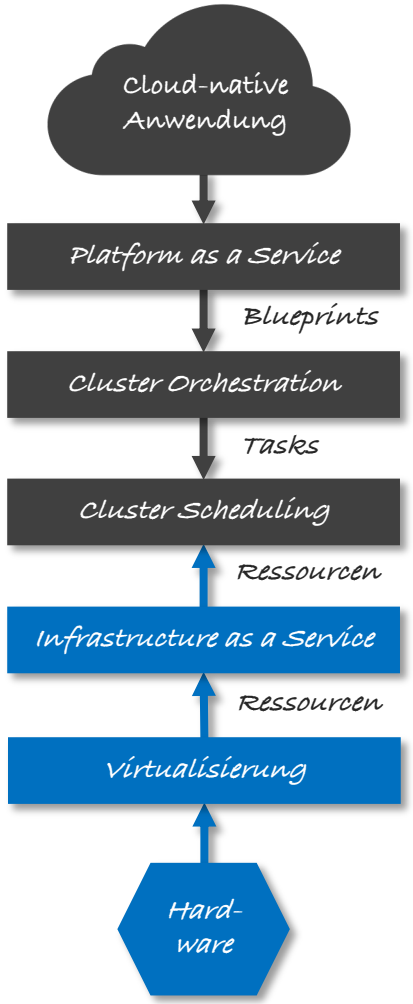

**Kubernetes (MicroK8S)**

```
cp Vagrantfile.microk8s Vagrantfile
vagrant up
```

*Diese Provisionierung kann Ihnen in diesem Modul noch viel Arbeit ersparen.*

# AUSBLICK

- Virtualisierung
- Infrastructure as a Service
- Provisionierung in IaaS-basierte Infrastrukturen
- Infrastructure as Code
  - Definition, Ansätze und Methoden
  - Überblick gängiger Provisionierungswerkzeuge
  - Single VM-Provisioning mittels Vagrant
  - Multi VM-Provisioning mittels Terraform



# KONTAKT

*Disclaimer*

**Nane Kratzke**

+49 451 300-5549

nane.kratzke@th-luebeck.de

kratzke.mylab.th-luebeck.de