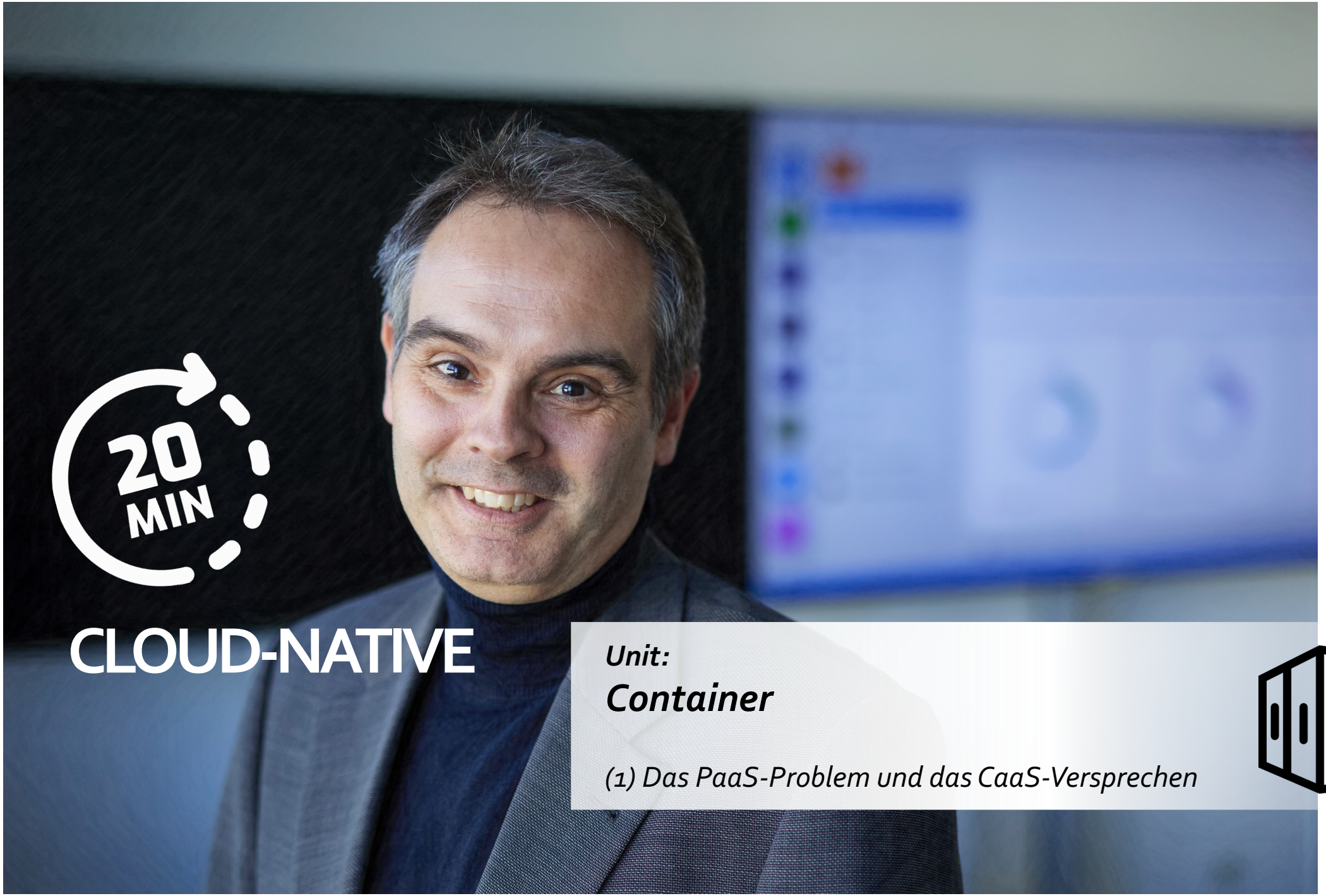


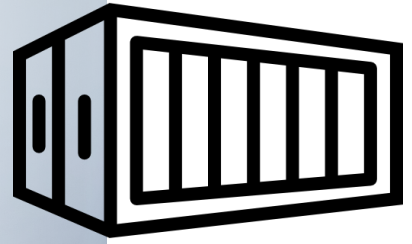
CLOUD NATIVE COMPUTING

CLOUD-NATIVE

Unit:

Container

(1) Das PaaS-Problem und das CaaS-Versprechen



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 8

Standardisierung von Deployment Units (Container)

8 Standardisierung von Deployment Units

8.1 Hintergrund (PaaS)

8.2 Betriebssystem-Virtualisierung

8.3 Container Runtime Environments
- Kernel-Namespaces
- Process Capabilities
- Control Groups
- Union Filesystems
- High- und Low-Level Container-Laufzeitumgebungen

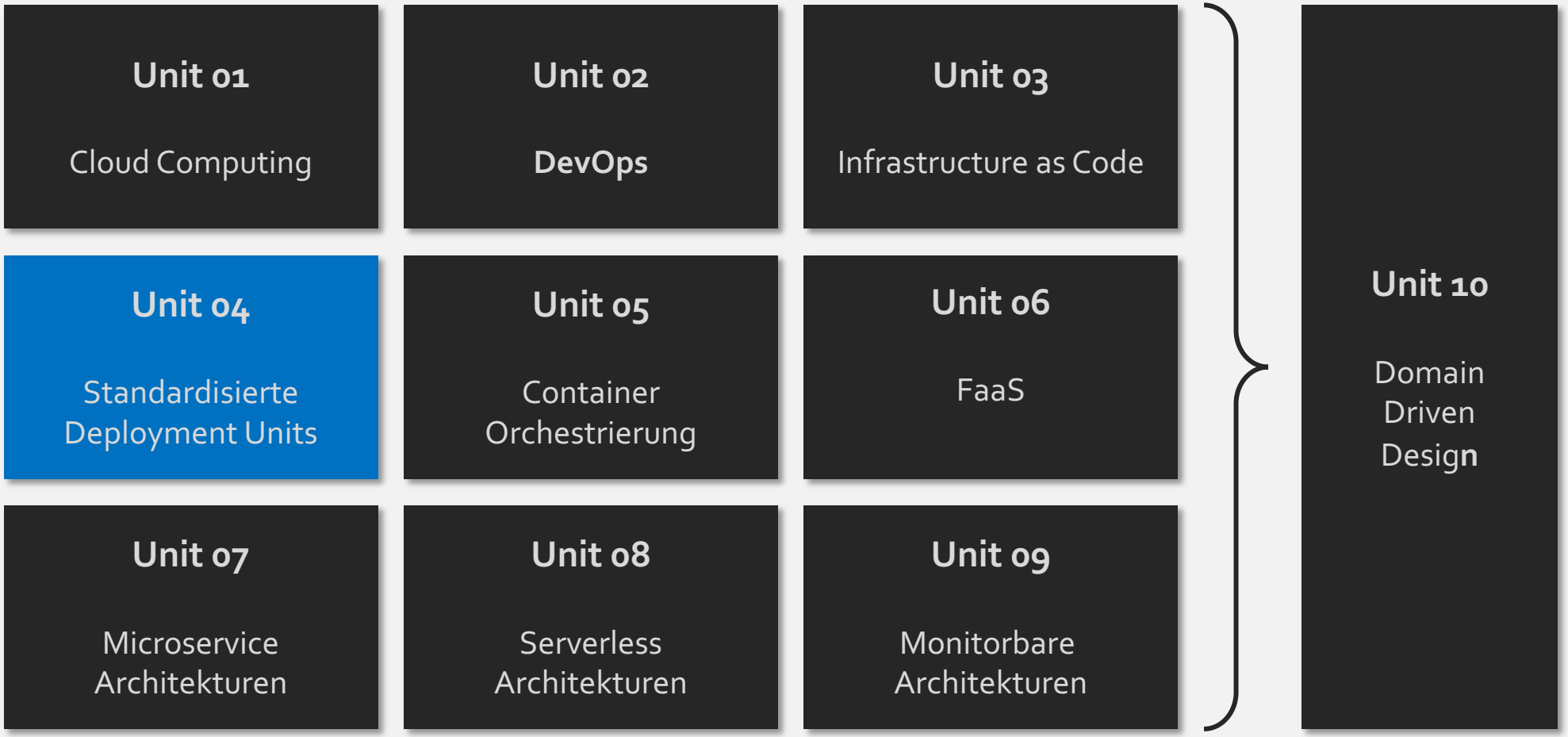
8.4 Bau und Bereitstellung von Container-Images

8.5 Faktoren gut betreibbarer Container
- Codebase
- Abhängigkeiten und Konfigurationen
- Unterstützende Services und Port Binding
- Build-, Release- und Run-Phase
- Horizontale Skalierung über Prozesse
- Umgebungen, Logs und Betrieb

8.6 Zusammenfassung

INHALTSVERZEICHNIS

Überblick über Units und Themen dieses Moduls



INHALTE

Hintergrund

- Platform as a Service
- Das PaaS-Problem
- Das CaaS-Versprechen (Standardisierung des Betriebs heterogener Komponenten)

Betriebssystem-Virtualisierung

- OS-Virtualisierung
- Linux-basierte Techniken zur OS-Virtualisierung
- Standardisierung von Deployment-Einheiten => Container

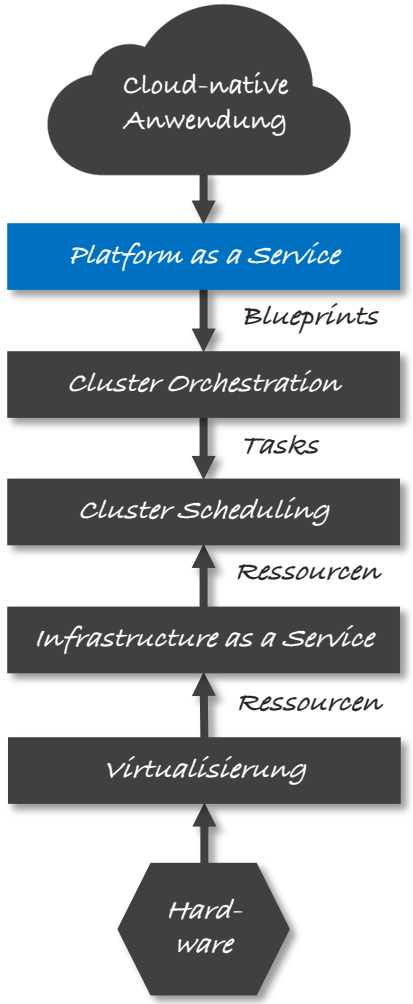
Laufzeitumgebungen für Container

- Container Laufzeitumgebungen und Standards
- Docker
- Image Building und Registries

Container-Pattern

- Container (Anti-)Pattern
- 12-Factor Apps

CLOUD NATIVE COMPUTING



ENTWICKLUNG

Von Platform as a Service zu Container as a Service

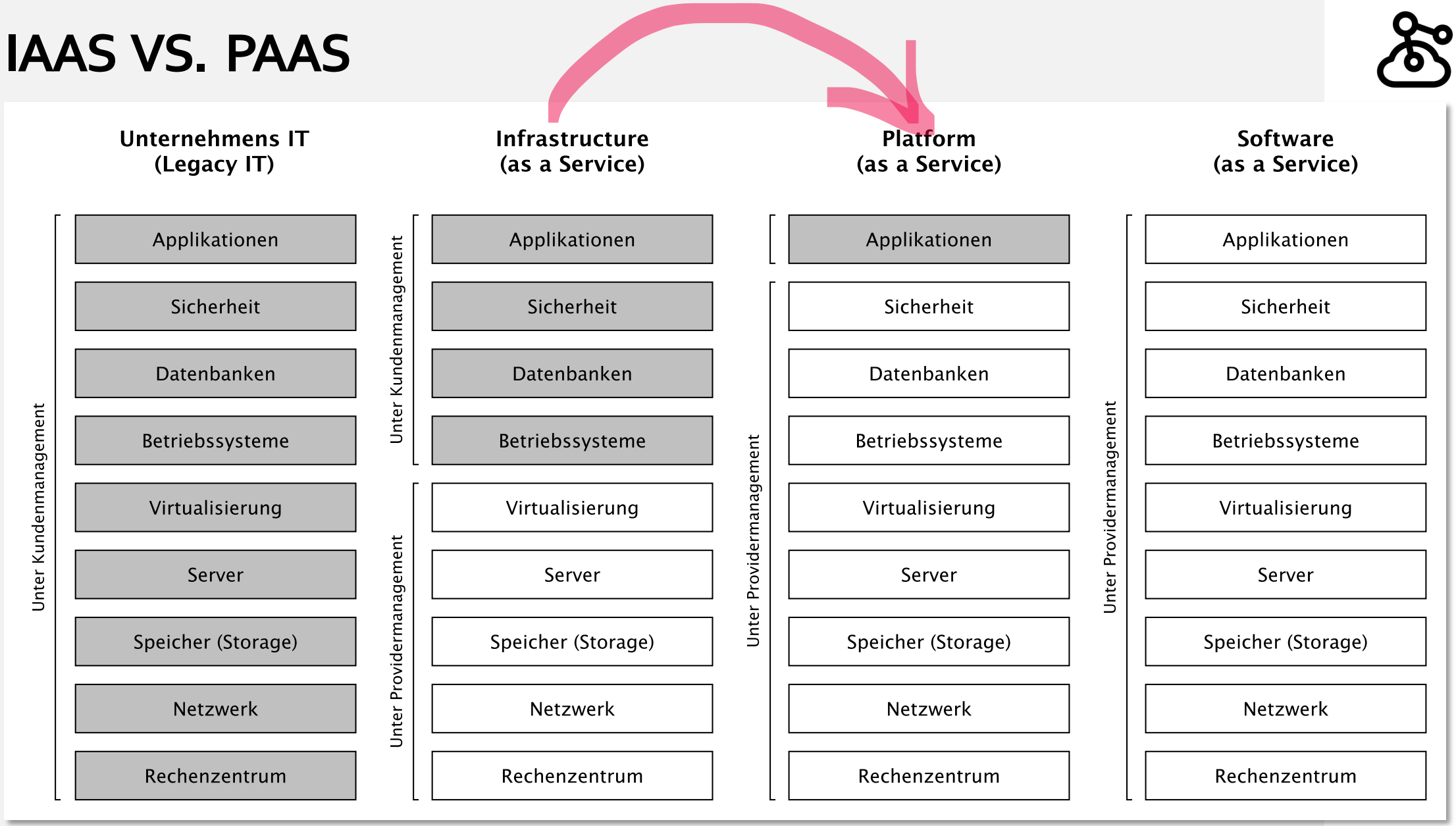
- Mittels IaaS und Provisionierungsansätzen lassen sich vordefinierte Depoyment-Einheiten (VM-Images) erzeugen und auf Hypervisoren als Virtuelle Maschinen ausbringen.

- Das Problem ist, diese Deployment-Einheiten (VM-Images) sind im allgemeinen sehr groß (>> 100MB bis zu mehreren GB) und die Start-Up-Zeiten dieser Maschinen sind recht lang (bis zu mehreren Minuten).

- Ferner wollen sich Entwickler häufig nicht mit den Details der Infrastruktur „herumschlagen" sondern einfach nur ihren Code ausbringen und laufen lassen.

- Aus dieser Motivationslage sind sogenannte Platform-as-a-Service PaaS-Angebote entstanden.

IAAS VS. PAAS

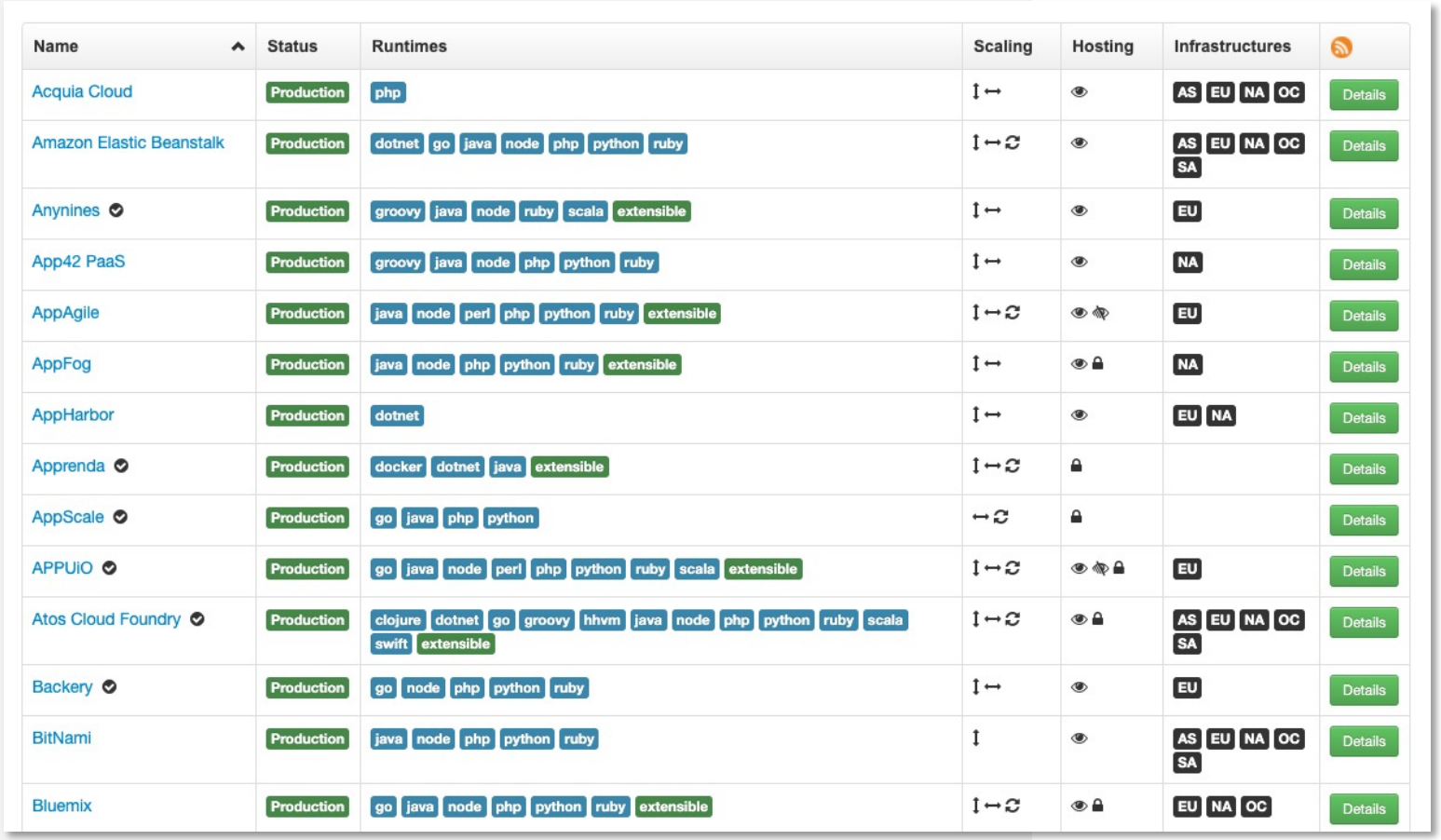

Unternehmens IT
(Legacy IT)
Infrastructure
(as a Service)
Platform
(as a Service)
Software
(as a Service)
Unter Kundenmanagement
Applikationen
Sicherheit
Datenbanken
Betriebssysteme
Virtualisierung
Server
Speicher (Storage)
Netzwerk
Rechenzentrum
Unter Kundenmanagement
Applikationen
Sicherheit
Datenbanken
Betriebssysteme
Unter Providermanagement
Virtualisierung
Server
Speicher (Storage)
Netzwerk
Rechenzentrum
Applikationen
Unter Providermanagement
Sicherheit
Datenbanken
Betriebssysteme
Virtualisierung
Server
Speicher (Storage)
Netzwerk
Rechenzentrum
Unter Providermanagement
Applikationen
Sicherheit
Datenbanken
Betriebssysteme
Virtualisierung
Server
Speicher (Storage)
Netzwerk
Rechenzentrum

PAAS

Das PaaS - Problem

Allein zwischen Oktober 2009 und Oktober 2010 sollen mehr als 100 PaaS-Anbieter den Markt betreten haben.

Alle mit dem Ziel, Kunden möglichst viele administrative Aufgaben abzunehmen, **Skalierbarkeit** und **Hochverfügbarkeit** zu ermöglichen, Fixkosten und **Gesamt-kosten zu senken**, die Anwender **flexibler** zu machen und um eine schnelle Anwendungsentwicklung und einen baldigen Markteintritt **(Time-to-Market)** zu ermöglichen.



Name	Status	Runtimes	Scaling	Hosting	Infrastructures	
Acquia Cloud	Production	php			AS EU NA OC	Details
Amazon Elastic Beanstalk	Production	dotnet go java node php python ruby			AS EU NA OC SA	Details
Anynines	Production	groovy java node ruby scala extensible			EU	Details
App42 PaaS	Production	groovy java node php python ruby			NA	Details
AppAgile	Production	java node perl php python ruby extensible			EU	Details
AppFog	Production	java node php python ruby extensible			NA	Details
AppHarbor	Production	dotnet			EU NA	Details
Apprenda	Production	docker dotnet java extensible				Details
AppScale	Production	go java php python				Details
APPUiO	Production	go java node perl php python ruby scala extensible			EU	Details
Atos Cloud Foundry	Production	clojure dotnet go groovy hhvm java node php python ruby scala swift extensible			AS EU NA OC SA	Details
Backery	Production	go node php python ruby			EU	Details
BitNami	Production	java node php python ruby			AS EU NA OC SA	Details
Bluemix	Production	go java node php python ruby extensible			EU NA OC	Details

Quelle: https://paasfinder.org

PAAS

Was ist das?

Als **Platform as a Service (PaaS)** bezeichnet man eine Dienstleistung, die als Cloud Service eine Plattform für Entwickler von Anwendungen zur Verfügung stellt. Dabei kann es sich sowohl um Laufzeitumgebungen (häufig für Webanwendungen), aber auch um Entwicklungsumgebungen handeln, die mit geringem administrativem Aufwand und ohne Anschaffung der darunterliegenden Hardware und Software genutzt werden können. Sie unterstützen wesentliche Teile des Software-Lebenszyklus von der Entwicklung, den Test, die Auslieferung bis hin zum Betrieb der Anwendungen über das Internet.

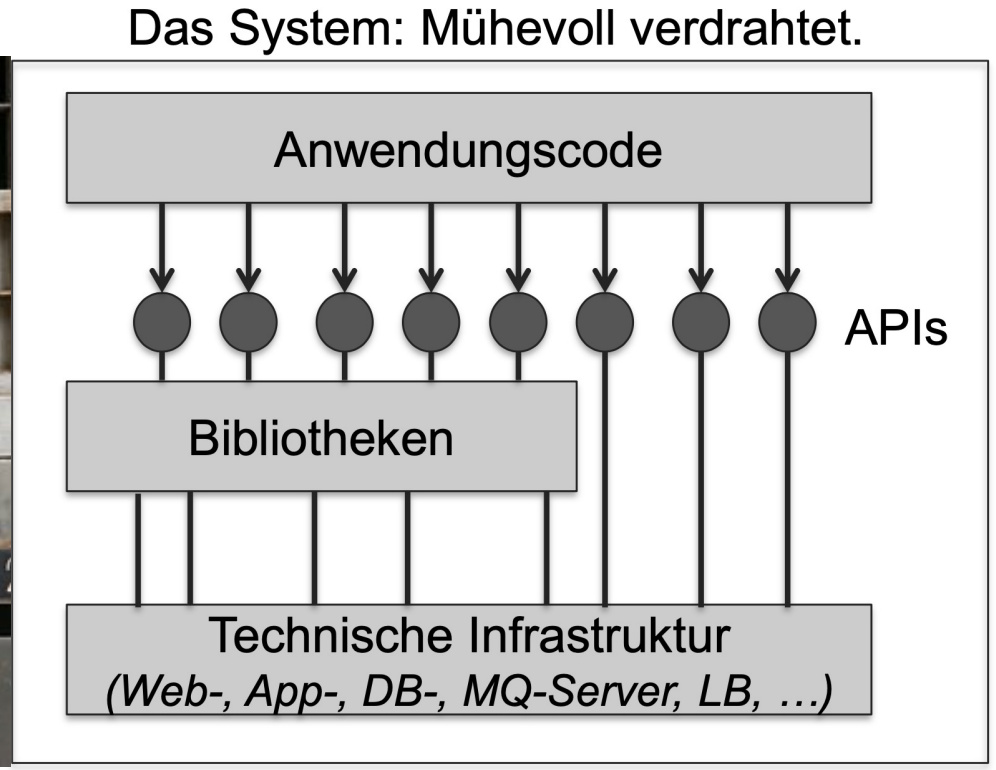
Cloud Anwendungen
Software as a Service (SaaS)

Cloud Plattformen
Platform as a Service (PaaS)

Cloud Infrastruktur
Infrastructure as a Service (IaaS)

DAS PROBLEM: STOVEPIPE ARCHITEKTUREN

Anwendungen aufwändig „von Hand" verdrahtet



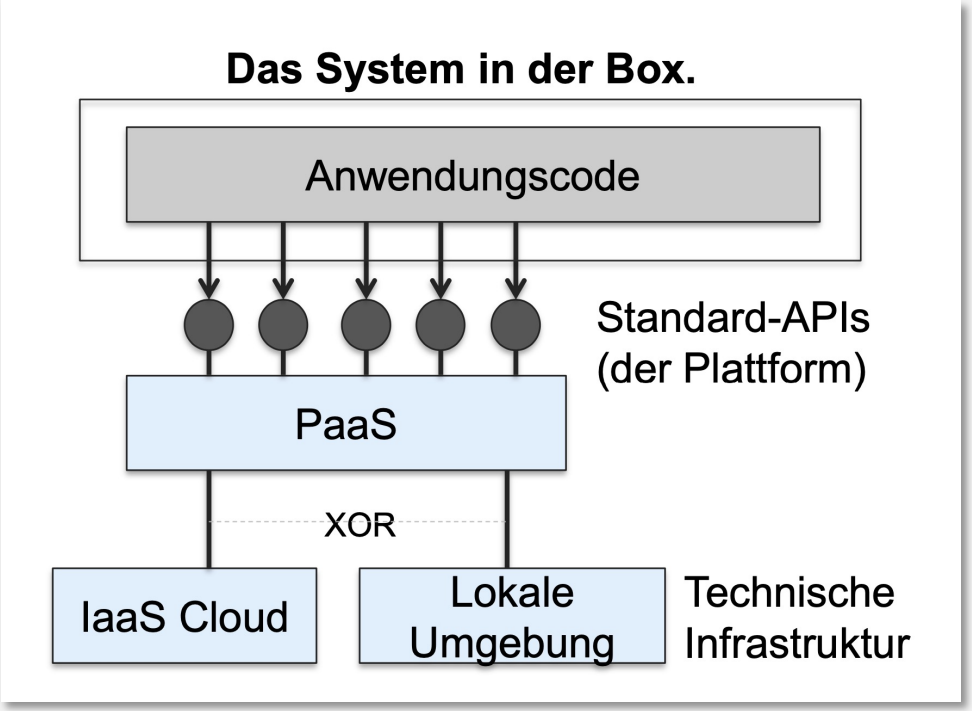
Mund-geblasenes mag ja schön sein, aber es ist teuer und vor allem kaum 1:1 wiederholbar!

Quelle: QAware GmbH
https://github.com/qaware/cloudcomputing

DIE LÖSUNG: PLATFORM AS A SERVICE

PaaS bietet eine ad-hoc Entwicklungs- und Betriebsplattform

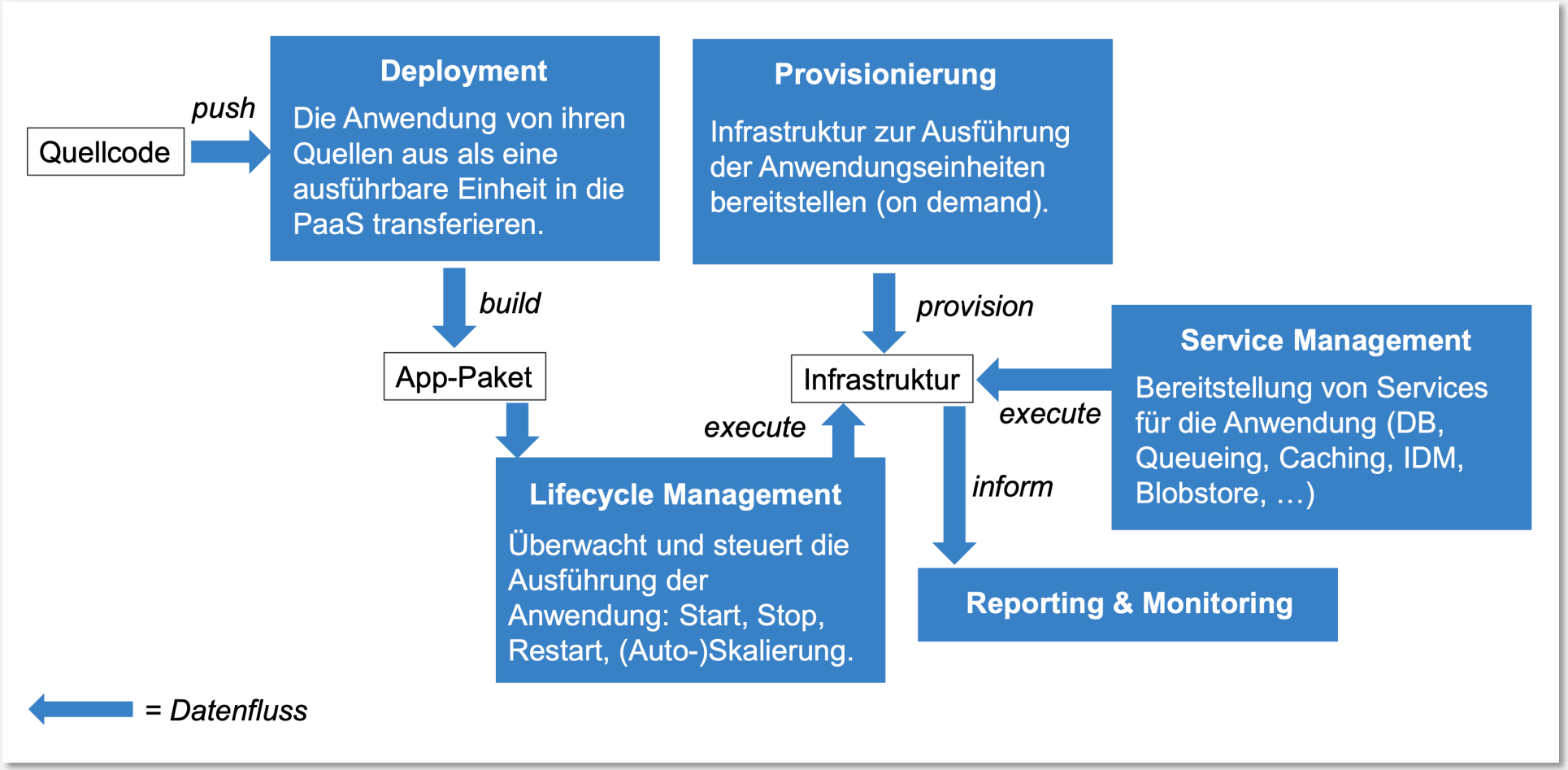
- Die Anwendung wird per Applikationspaket oder als Quellcode deployed. Es ist kein Image mit technischer Infrastruktur notwendig.
- Die Anwendung sieht nur Programmier- oder Zugriffsschnittstellen seiner Laufzeitumgebung.
- Es erfolgt eine automatische Skalierung der Anwendung.
- Entwicklungswerkzeuge (IDEs, Build-Systeme, Testumgebung) stehen zur Verfügung („Deploy to cloud").
- Die Plattform bietet eine Schnittstelle zur Administration und zum Monitoring.



Quelle: QAware GmbH
https://github.com/qaware/cloudcomputing

PAAS CLOUD

Die funktionalen Bausteine einer PaaS Cloud



Quelle: QAware GmbH
https://github.com/qaware/cloudcomputing

PRIVATE PAAS CLOUDS

Alternative Private PaaS Clouds

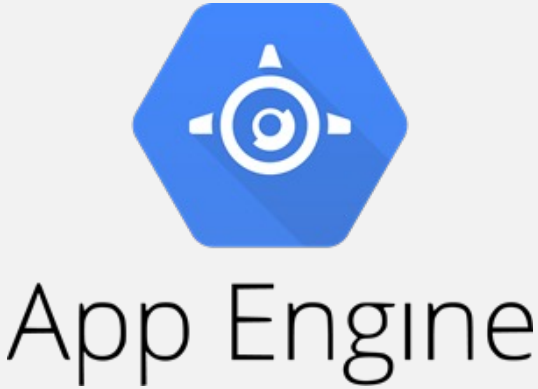
Name	Link	Anmerkung
OpenShift	https://openshift.com	PaaS mit Schwerpunkt JEE von RedHat
CloudFoundry	https://cloudfoundry.org	PaaS von Pivotal mit breiter Unterstützung aus der Industrie
Stackato	https://activestate.com/stackato	Private PaaS (kommerziell)
PaaSTO	https://github.com/yelp/paasta	Open-Source private PaaS auf Basis von Mesos+Marathon
VAMP	https://vamp.io	Leichtgewichtiges Open-Source PaaS ausgelegt auf Microservices (Kubernetes oder Mesos)
Apollo	https.//github.com/capgemini/apollo	Open-Source private PaaS auf Basis von Mesos von Capgemini
Mantl	https://mantl.io	Open-Source private PaaS auf Basis von Mesos von Cisco

Quelle: QAware GmbH
https://github.com/qaware/cloudcomputing

PUBLIC PAAS CLOUDS

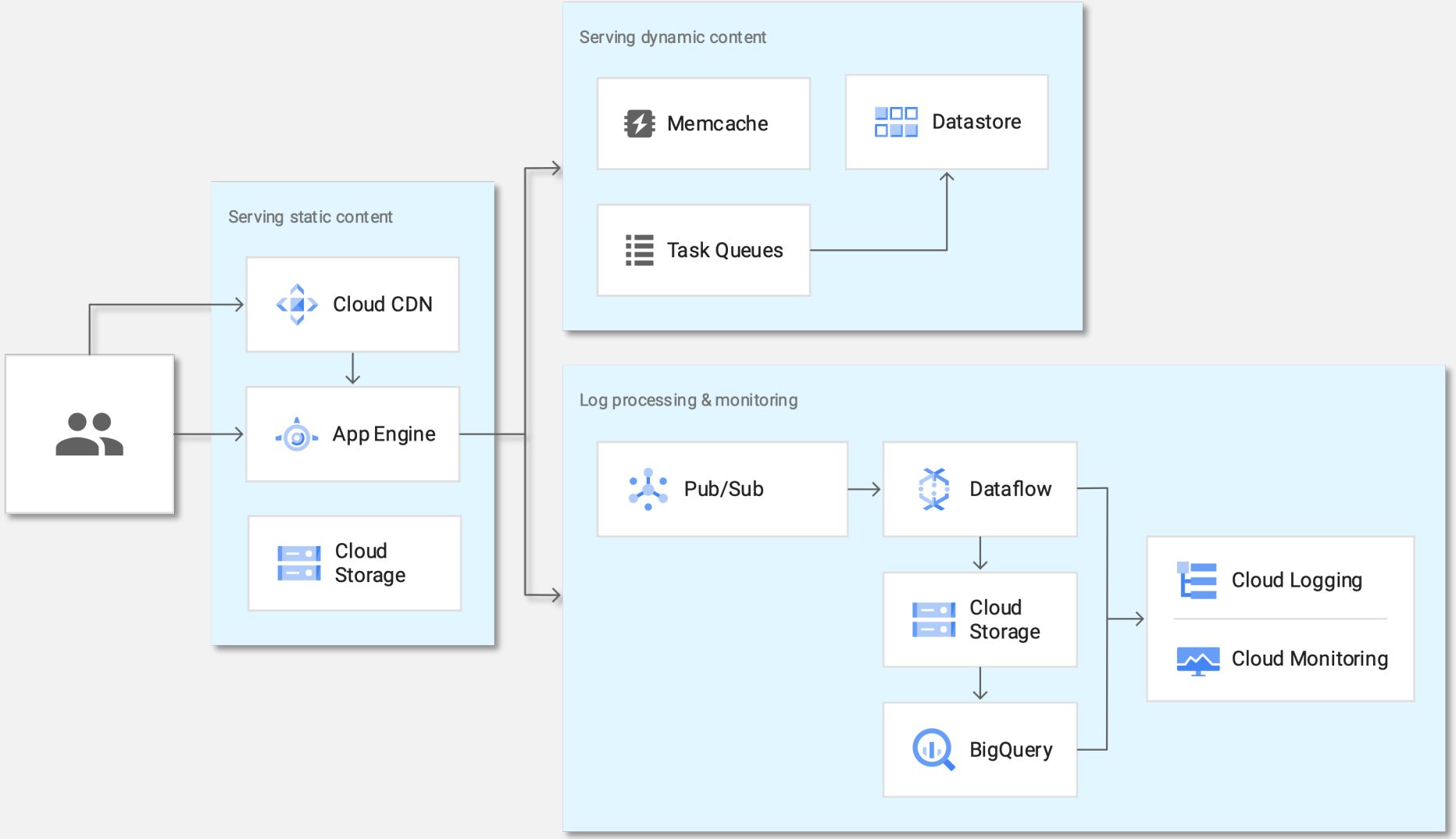
Am Beispiel der Google App Engine

- Die Google App Engine (GAE) ist das PaaS Angebot von Google.
- Anwendungen laufen innerhalb der Google Infrastruktur.
- Der Betrieb der Anwendungen ist innerhalb bestimmter Quoten kostenfrei.
- Unterstützte Sprachen: u.a. JS, Ruby, Go .NET, Java, Python, PHP
- Integrationen in alle gängigen IDEs stehen zur Verfügung



GOOGLE APP ENGINE

Referenzarchitektur



Quelle: Google
https://cloud.google.com/appengine

GOOGLE APP ENGINE

Sandbox-bedingte Einschränkungen

Eine GAE-Applikation läuft in einer Sandbox, die das Verhalten der Applikation mit dem Ziel einschränkt, die Verarbeitung effizient zu halten und die Infrastruktur im Auto-Scaling zu schützen.

Es dürfen daher nicht alle Klassen von Standardbibliotheken genutzt werden:

- Keine eigenen Threads öffen
- Kein Zugriff auf die Laufzeitumgebung (z.B. Classloader)

Kommunikation mit anderen Web-Anwendungen nur über URL Fetch, XMPP oder E-Mail

- Anfragen und Antworten dürfen maximal 1MB groß sein
- Web-Hooks als allgemeines Architekturmittel für eingehende Kommunikation. Angestoßen bei Ereignissen (Warmup, Messages, Cron-Jobs)

Alle Requests an eine GAE-Anwendung werden nach 60 Sekunden beendet. Es gibt weitere diverse Einschränkungen zu Datenvolumina und Anzahl von Service Aufrufen.

Einschränkungen ähnlicher Art finden sich auch bei anderen Anbietern.

DAS PAAS-PROBLEM

Fehlende Portabilität von Anwendungen

„In recent years, the cloud hype has led to a multitude of different offerings across the entire cloud market, from Infrastructure as a Service (IaaS) to Platform as a Service (PaaS) to Software as a Service (SaaS). Despite the high popularity, there are still several problems and deficiencies.

Especially for PaaS, the heterogeneous provider landscape is an obstacle for the assessment and feasibility of application portability.“

Kolb, Stefan. (2018). On the Portability of Applications in Platform as a Service. 10.20378/irbo-54102.

DAS PAAS-PROBLEM

Am Beispiel zweier Java Getting-Started-Tutorials von Heroku und Google App Engine

heroku

```java
import com.heroku.api;
// Further imports skipped for clarity

@Controller
@SpringBootApplication
public class HerokuApplication {

  @Value("${spring.datasource.url}")
  private String dbUrl;

  @Autowired
  private DataSource dataSource;

  public static void main(String[] args) throws Exception {
    SpringApplication.run(HerokuApplication.class, args);
  }

  // Further methods skipped for clarity
}
```

git clone https://github.com/heroku/gradle-gettingstarted.git

App Engine

```java
import com.google.appengine.api.utils.SystemProperty;
// Further imports skipped for clarity

@WebServlet(name = "HelloAppEngine", value = "/hello")
public class HelloAppEngine extends HttpServlet {

  @Override
  public void doGet(
      HttpServletRequest request,
      HttpServletResponse response
  ) throws IOException {
    Properties properties = System.getProperties();
    response.setContentType("text/plain");
    response.getWriter().println("Hello App Engine");
  }

  // Further methods skipped for clarity
}
```

git clone https://github.com/GoogleCloudPlatform/getting-started-java.git

CLOUD NATIVE COMPUTING

Wesentliche Probleme bei PaaS sind u.a. folgende fehlende Standards:

- *Deployment-Format* der zu hostenden Anwendungen
- *PaaS-Runtime-Schnittstelle*

Challenge: Versuchen Sie mal das Heroku Getting-Started auf Google App Engine zum Laufen zu bringen (und anders herum)

PAAS

Erkennbarer Trend in PaaS Plattformen

Name	Status	Runtimes	Scaling	Hosting	Infrastructures	
Acquia Cloud	Production	php			AS EU NA OC	Details
Amazon Elastic Beanstalk	Production	dotnet go java node php python ruby			AS EU NA OC SA	Details
Anynines	Production	groovy java node ruby scala extensible			EU	Details
App42 PaaS	Production	groovy java node php python ruby			NA	Details
AppAgile	Production	java node perl php python ruby extensible			EU	Details
AppFog	Production	java node php python rub extensible			NA	Details
AppHarbor	Production	dotnet			EU NA	Details
Apprenda	Production	docker otnet jav extensible				Details
AppScale	Production	go java php python				Details
APPUiO	Production	go java node perl php python ruby scal extensible			EU	Details
Atos Cloud Foundry	Production	groovy hhvm java node php python ruby scala extensible			AS EU NA OC SA	Details
Backery	Production	go node php python ruby			EU	Details
BitNami	Production	java node php python ruby			AS EU NA OC SA	Details
Bluemix	Production	go java node php python extensible			EU NA OC	Details

Quelle: https://paasfinder.org

Laufzeitumgebungen können durch den Kunden erweitert werden.

Wie funktioniert das technisch?

AUSBLICK

CLOUD NATIVE COMPUTING

Hintergrund

- Platform as a Service
- Das PaaS-Problem
- Das CaaS-Versprechen

Betriebssystem-Virtualisierung

- OS-Virtualisierung
- Linux-basierte Techniken zur OS-Virtualisierung
- Standardisierung von Deployment-Einheiten => Container

Laufzeitumgebungen für Container

- Container Laufzeitumgebungen und Standards
- Docker
- Image Building und Registries

Container-Pattern

- Container (Anti-)Pattern
- 12-Factor Apps

KONTAKT

Disclaimer

Nane Kratzke

+49 451 300-5549

nane.kratzke@th-luebeck.de

kratzke.mylab.th-luebeck.de