



HOCHSCHULE BREMEN  
FAKULTÄT 4 – ELEKTROTECHNIK UND INFORMATIK  
INTERNATIONALER STUDIENGANG MEDIENINFORMATIK (B.Sc.)

## EXPOSÉ

---

### Bachelorthesis

– Transformation einer Bestandsanwendung in eine  
Cloud-Native-Architektur –

---

Lukas Karsten (5011712)

04. November 2021 (Version 1.01)

# 1 Einleitung

Moderne IT-Systeme folgen bei ihrer Bereitstellung oft den Prinzipien sogenannter Cloud-Native-Architektur und sind damit so gestaltet, dass sie skalierbar, resilient, administrierbar, observierbar und automatisiert sind [1]. Allerdings ist es oft nicht möglich diese Systeme von Grund auf neu zu entwickeln. Ein Großteil der essentiellen Geschäftsprozesse laufen bereits in Systemen bei denen diese architektonischen Erkenntnisse noch keine Anwendung fanden. Eine große Herausforderung vieler Unternehmen ist es heute, ihre bestehenden Infrastruktur schrittweise so zu verändern [2], dass möglichst viele der moderne Prinzipien angewendet werden können, ohne den laufenden Betrieb zu stören oder eine komplette Neuentwicklung der Server-Architektur zu erzwingen. Im Rahmen dieser Arbeit wird ein Konzept erarbeitet, in welchem die Startnext-Plattform von der bestehenden, klassischen Architektur schrittweise in eine cloud-native Infrastruktur überführt wird und die Vorteile dieser Technologie effektiv nutzt. Zudem werden die erarbeiteten Maßnahmen auch praktisch umgesetzt, diskutiert und der Erfolg der Maßnahmen bewertet.

## 2 Problemstellung und Lösungsansatz

### 2.1 Problemstellung

Moderne Softwareentwicklung stellt hohe Anforderungen an ihre Infrastruktur. Um neue Features zu testen, werden kurzzeitig Testinstanzen benötigt, Änderungen sollen mehrmals täglich ins Produktivsystem eingespielt werden [3]. Lastspitzen müssen schnell abgefangen und auf schwerwiegende Bugs mit zurückrollen zu vorherigen Versionen reagiert werden. [Quelle] Klassische Bereitstellung von Software über selbst- oder fremdgehostete Server schränkt die Anpassungsmöglichkeiten bei Bedarf an mehr Rechenleistung, weitere Rechen-Instanzen oder Serverstandorte stark ein und lässt Änderungen an der Infrastruktur nur sehr träge zu.

Die Wartung und Aktualisierung der Infrastruktur erfolgt über das Einwählen und manuelle Aufspielen von Updates. Dabei verändert sich der Zustand jeder einzelnen Instanz über die Zeit. Es kommt zum auseinanderdriften von Zuständen der einzelnen Server. Zusätzlich ist der Weg zu diesem Zustand schwer bis nicht nachvollziehbar. Das Updaten der Instanzen wird unvorhersehbar und potentielle Fehler passieren auf produktiven Systemen.

### 2.2 Lösungsansatz

Der Ansatz von Infrastructure as a Service (IaaS) bietet einen Lösungsansatz für genau diese Probleme. Als IaaS werden Cloud-Computing Services bezeichnet, die Rechner-, Speicher- und Netzwerkressourcen auf Anfrage bereitstellen. Dabei wird jede der Ressourcen als separate Komponente bereitgestellt und nur für die Nutzungsdauer in Rechnung gestellt. Der Aufwand und die Kosten für das Betreiben von physikalischen Servern und Infrastruktur wird also umgangen, während das Einrichten, sowie Deinstallieren neuer Infrastruktur vollautomatisiert und innerhalb weniger Sekunden erfolgen kann. Durch die alleinige Um-

stellung zu einem Hosting bei einem Cloud-Anbieter ergeben sich noch keine signifikanten Vorteile im Betrieb der Anwendung. Um die mit IaaS erworbene Skalierbarkeit, Flexibilität, und vereinfachte Verwaltbarkeit [Quelle] zu nutzen bedarf es ein Neudenken im Design der Infrastruktur und ihren Bereitstellungsprozessen, im allgemeinen als Cloud-Native Architektur bezeichnet. Zusätzlich bringen diese neuen Konzepte wiederum Herausforderungen mit sich, die es zu identifizieren und bewältigen gilt.

Im Rahmen der Bachelorthesis werden bereits gängige Konzepte und „best-practices“ im Betrieb von cloudnahen Anwendungen evaluiert und am Beispiel der Startnext-Infrastruktur auf ihre Anwendbarkeit, Sinnhaftigkeit und durch das Erstellen von Proof of Concepts auf ihre Machbarkeit untersucht und diskutiert.

## 3 Fachliche Überlegungen und Zielsetzung

### 3.1 Fachliche Überlegungen

Cloud-native dient als Überbegriff für eine Vielzahl an Paradigmen und Prinzipien, die das Entwerfen und Implementieren von Infrastrukturen und Software als skalierbare, resiliente, administrierbare, observierbare und automatisierte Systeme ermöglicht. Das Immutable Infrastructure Paradigma spielt eine übergeordnete Rolle in Cloud Native Architekturen und ebnet den Weg für die Implementation vieler weiterer Prinzipien. Immutable Infrastructure verfolgt das Ziel, eines weitestgehend unveränderlichen Zustands eines bestehenden Systems. So führt also ein benötigtes Update eines Servers zum Erstellen eines neuen Servers mit dem neuen gewünschten Zustand und dem Entfernen der alten Instanz. Dieser Ansatz ist eine direkte Antwort auf das auseinanderdriften von Zuständen und dem dadurch unkalkulierbaren Ausgang von Zustandsänderungen wie in Kapitel 2 beschrieben. Zusätzlich müssen Änderungen nicht mehr an bereits produktiven Umgebungen vorgenommen werden, deren Ausfall zu direkten Auswirkungen an der Anwendung führen. Im Gegenteil können diese Änderungen ausgeführt, geprüft und anschließend in den produktiven Einsatz geschaltet werden.

#### 3.1.1 Technologien

Bei der Umsetzung von Immutable Infrastructure spielen Werkzeuge und Technologien eine entscheidene Rolle um die gewünschte Nachvollziehbarkeit, Berechenbarkeit, sowie Skalierbarkeit zu gewährleisten.

Um reproduzierbar Systemumgebungen zu erstellen werden Maschinenimages genutzt. Moderne Tools ermöglichen das Erstellen von Konfigurationen als Code, aus denen Maschinenimages für unterschiedliche Plattformen erstellt werden können. Durch den Äs-CodeAnsatz werden Images versionierbar. Änderung an diesen sind nachvollziehbar und widerruflich. Zusätzlich können über verschiedene Instanzen genutzte Grund-Images, auch Golden Image [4] genannt für eine einheitliche und erleichterte Wartung der Systeme sorgen.

Der Äs-CodeAnsatz ist ebenso bei der Provisionierung von Hosts möglich. Infrastructure-As-Code (IaC) Tools ermöglichen das Erstellen von Umgebungskonfigurationen als Code,

die von IaS-Anbietern interpretierbar sind. So wird Idempotenz bei der Bereitstellung von Hosts gewährleistet. Die Provisionierung wird nachvollziehbar als auch automatisierbar und Änderungen an Umgebungen sind versionier- sowie testbar.

Durch Versionierung und in Code vorliegende Konfigurationen aller Bestandteile der Anwendung ist es möglich den gesamten Bereitstellungsprozess zu Automatisieren und den in Kapitel 2 beschriebenen Anforderungen an moderne Infrastruktur zu entsprechen. Der Prozess vom Provisionieren des Hosts, über das Konfigurieren dieser und dem Aufspielen der Software soll gemeinsam mit entsprechenden Tests jeder dieser Schritte automatisiert ablaufen. So kann auf Anforderungsänderungen schnell reagiert werden und das komplexe System der Startnext-Infrastruktur bleibt administrierbar.

Weiter sind Konzepte und Tools benötigt, welche den Anwendungsbetrieb und ihre automatisierte Bereitstellung sicher, einfach zu administrieren und unanfällig für Fehler macht. Das Verwalten von Geheimnissen, Orchestrieren und Registrieren von Services und Testens sind hier zu nennen.

## 4 Verwandte Arbeiten

Die ansteigende Nachfrage nach Cloud-nativen Technologien spiegelt sich auch in der wissenschaftlichen Literatur wieder. So findet auch das Thema Immutable Infrastructure als eine der grundlegenden Disziplinen in Cloud Native hohe Aufmerksamkeit in wissenschaftlichen Abhandlungen.

Rob Hirschfeld [5] diskutiert auf der SREcon 18 Immutable Deployments und ihren Mehrwert. Er arbeitet heraus, dass der Ansatz Immutable Infrastructure zusammen mit Cloud Computing zum kosteneffizienten Ansatz für das Bereitstellen von Laufzeitumgebungen wird. Hirschfeld teilt Anwendungsweisen von ImmuInfra in 3 verschiedene Immutable Patterns ein, die vom einfachen Zurücksetzen und Neubspielen, bis zum Image Deploy, also das Erstellen eines fertigen Images, welches anschließend aufgespielt wird reichen. Diese drei Immutable Patterns gehen alle mit eigenen Vor- sowie Nachteilen einher. Hirschfelds Konferenzbeitrag bietet einen robusten Leitfaden um die Wahl des Immutable Infrastructure Ansatzes zu erleichtern und ihre jeweiligen Vorteile klarzustellen.

Mir der Gestaltung eines resilienten und skalierenden Cloud-Native Systems durch eine selbstadministrierende Anwendung beschäftigt sich Toffetti [6]. Toffetti erstellt ein Cloud-Native Design, welches in der Lage ist selbständig zu skalieren und sich selbstzuheilen. Die Architektur beinhaltet neben Cloud Orchestrierung und verteiltem Konfigurationsmanagement ein verteiltes Datenbanksystem, sowie die Fähigkeit der Leader-Election. Anschließend wird dieses Design auf eine bestehende Legacy Web-App angewandt. Er und sein Team emulieren Fehler in den containerisierten Teilen der Anwendung, als auch in den VMs, welche die Container hosten. Sie messen die Reaktion des Systems auf Fehler und diskutieren diese. Toffettis Journal bietet eine robuste Grundlage für die Wahl von DesignPatterns und eine Orientierung für das Erstellen eines selbstgemanagten Cloud-Native Systems.

## 5 Konkrete Aufgaben und Zeitplan

Die Ausführung der Bachelorarbeit ist zeitlich wie folgt geplant:

Dauer: 9 Wochen

- Woche 1: Literaturrecherche, Einleitung, Einrichten des Dokuments
- Woche 2: Verfassen der Problemstellung, Zielsetzung und möglichen Lösungsansätzen
- Woche 3: Entwickeln der Anforderungsanalyse
- Woche 4: Grundlagen und verwandte Arbeiten - Verfassen und Recherche
- Woche 5: Konzeption - Entwickeln eines Entwurfs zur Lösung der Problemstellung
- Woche 6-7: Prototypische Umsetzung
- Woche 8: Evaluation - Bewertung und Diskussion der Ergebnisse
- Woche 9: Überarbeitung, Korrektur und Druck des Dokuments

## 6 Vorläufige Gliederung

Eigenständigkeitserklärung

Zusammenfassung / Abstract

1. Einleitung
  - 1.1. Problemfeld
  - 1.2. Zielsetzung
  - 1.3. Lösungsansatz
  - 1.4. Aufbau der Arbeit
2. Anforderungsanalyse
  - 2.1. Funktionale Anforderungen
  - 2.2. Nicht-funktionale Anforderungen
  - 2.3. Zusammenfassung
3. Grundlagen und verwandte Arbeiten
  - 3.1. Immutable Infrastructure
  - 3.2. HashiCorp - Packer
  - 3.3. Selfmanaged Infrastructure
  - 3.4. Monitoring von Infrastruktur
  - 3.5. Automatisierte Orchestrierung und Konfiguration von Infrastruktur
  - 3.6. HashiCorp - Nomad & Consul

3.7. Verwandte Arbeiten

4. Konzeption

4.1. Entwurf einer Cloud-Native Infrastruktur

4.1.1. Image-Erstellung

4.1.2. Deployment

4.1.3. Konfiguration

4.1.4. Orchestrierung

4.1.5. Monitoring

4.2. Zusammenfassung

5. Prototypische Realisierung

5.1. Grundlegender Aufbau der Infrastruktur

5.2. Konfigurieren der Services

6. Qualitätssicherung

7. Zusammenfassung

Evaluation

1. Überprüfung funktionaler Anforderungen

2. Überprüfung nicht-funktionaler Anforderungen

3. Zusammenfassung

Zusammenfassung und Ausblick

1. Zusammenfassung

2. Ausblick

Literaturverzeichnis

## 7 Unterschriften

Student:

Name: *Lukas Karsten*

Adresse: *Bismarckstraße 156, 28205 Bremen*

Telefon: *0157/81733962*

E-Mail: *lkarsten@stud.hs-bremen.de*

---

Unterschrift (Student\_in)

Erstgutachter\_in / Betreuer\_in: *Prof. Dr. Lars Braubach*

---

Unterschrift (Erstgutachter)

Zweitgutachter\_in:

## Literatur

### Gedruckte Quellen

- [1] Jung-Bok Lee u. a. „High-Performance Software Load Balancer for Cloud-Native Architecture“. In: *IEEE Access* 9 (2021), S. 123704–123716. DOI: [10.1109/ACCESS.2021.3108801](https://doi.org/10.1109/ACCESS.2021.3108801).
- [2] Fiedelholtz. *The Cyber Security Network Guide. Studies in Systems, Decision and Control, vol 274*. Springer, 2021.
- [5] Rob Hirschfeld. „Don’t Ever Change! Are Immutable Deployments Really Simpler, Faster, and Safer?“ In: Santa Clara, CA: USENIX Association, März 2018.
- [6] Giovanni Toffetti u. a. „Self-managing cloud-native applications: Design, implementation, and experience“. In: *Future Generation Computer Systems* 72 (Juli 2017), S. 165–179. DOI: [10.1016/j.future.2016.09.002](https://doi.org/10.1016/j.future.2016.09.002). URL: <https://doi.org/10.1016/j.future.2016.09.002>.

### Online Quellen

- [3] IBM Cloud Education. *Continuous Delivery*. IBM. Juli 2019. URL: <https://www.ibm.com/cloud/learn/continuous-delivery> (besucht am 30.10.2021).
- [4] Otto Geißler & Ulrike Ostler. *Was ist ein Golden Image?* Datacenter Insider. Nov. 2018. URL: <https://www.datacenter-insider.de/was-ist-ein-golden-image-a-775197/> (besucht am 14.11.2021).