MATHEMA

Server?
Wo wir hingehen brauchen wir keine Server…
Serverless Programmieren mit AWS Lambda

Werner Eberling (@Wer_Eb)

werner.eberling@mathema.de
www.mathema.de

Der Sprecher

Werner Eberling

Principal Consultant / Autor

Email: werner.eberling@mathema.de

Twitter: @Wer_Eb

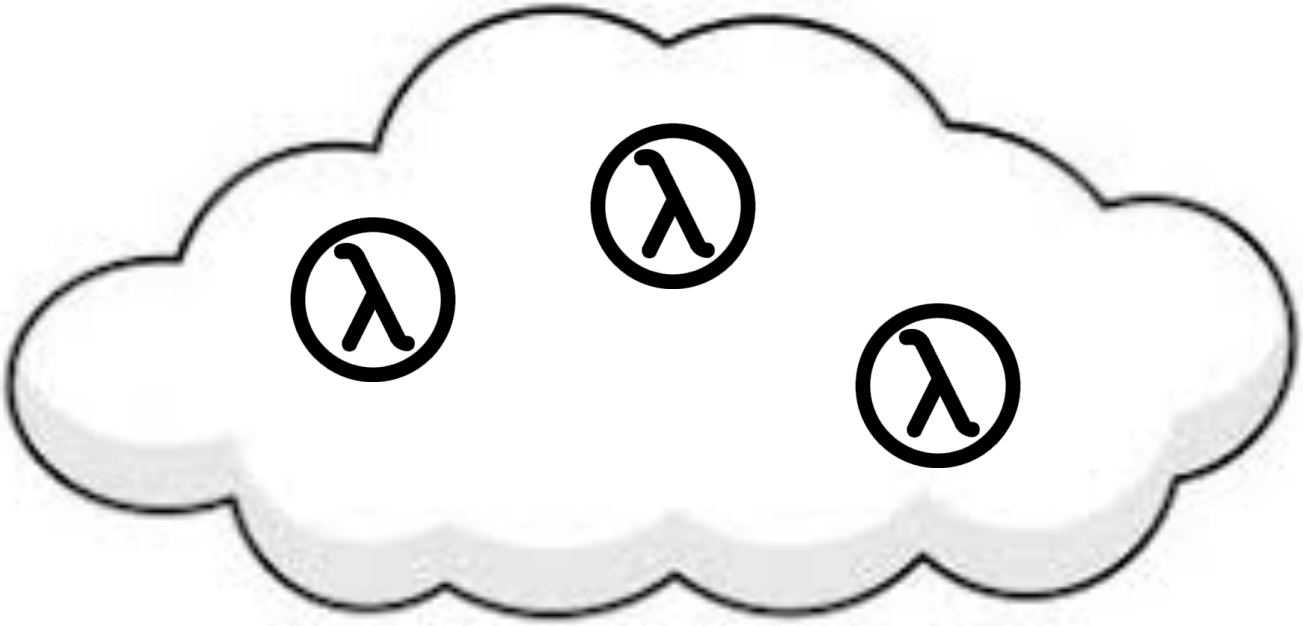
Serverless - Was man so hört…

Don‘t pay for idle!!!

Cloud native Entwicklung

No Ops

Wirklich keine Server mehr?



Keine Gedanken mehr über Infrastruktur

- Backend as a Service (BaaS)
  - Individuelle Frontends nutzen bestehende (cloud-basierte) Dienste von Drittanbietern
  - Rich Clients halten die spezielle Business Logik

Keine Gedanken mehr über Infrastruktur

- Backend as a Service (BaaS)
  - Individuelle Frontends nutzen bestehende (cloud-basierte) Dienste von Drittanbietern
  - Rich Clients halten die spezielle Business Logik

- Function as a Service (FaaS)
  - Zustandslose, flüchtige Laufzeitcontainer
  - Bereitgestellt durch Drittanbieter (Cloud-basiert)
  - „Eigene Funktionen als BaaS“

Keine Gedanken mehr über Infrastruktur

- Backend as a Service (BaaS)
  - Individuelle Frontends nutzen bestehende (cloud-basierte) Dienste von Drittanbietern
  - Rich Clients halten die spezielle Business Logik

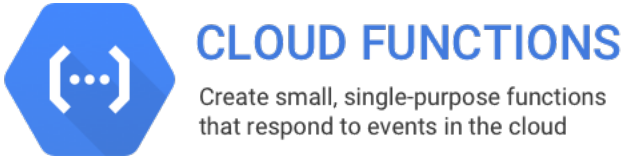
- Function as a Service (FaaS)
  - Zustandslose, flüchtige Laufzeitcontainer
  - Bereitgestellt durch Drittanbieter (Cloud-basiert)
  - „Eigene Funktionen als BaaS“

Keine Gedanken mehr über Infrastruktur

- Konzentration auf Funktionalität statt auf Infrastruktur
  - „SoC auf neuem Level“

Keine Gedanken mehr über Infrastruktur

- Konzentration auf Funktionalität statt auf Infrastruktur
  - „SoC auf neuem Level“



Keine Gedanken mehr über Infrastruktur

- Konzentration auf Funktionalität statt auf Infrastruktur
  - „SoC auf neuem Level“

AWS Lambda in Stichworten

- AWS Service seit November 2014
- Unterstützte Sprachen
  - Node.js, Java, C#, Python, Go, PowerShell, Ruby, eigene Runtime

Amazon Lambda

- Zugriff auf AWS Services
  - S3, DynamoDB, SNS, SES, Kinesis, IAM, SSO, …

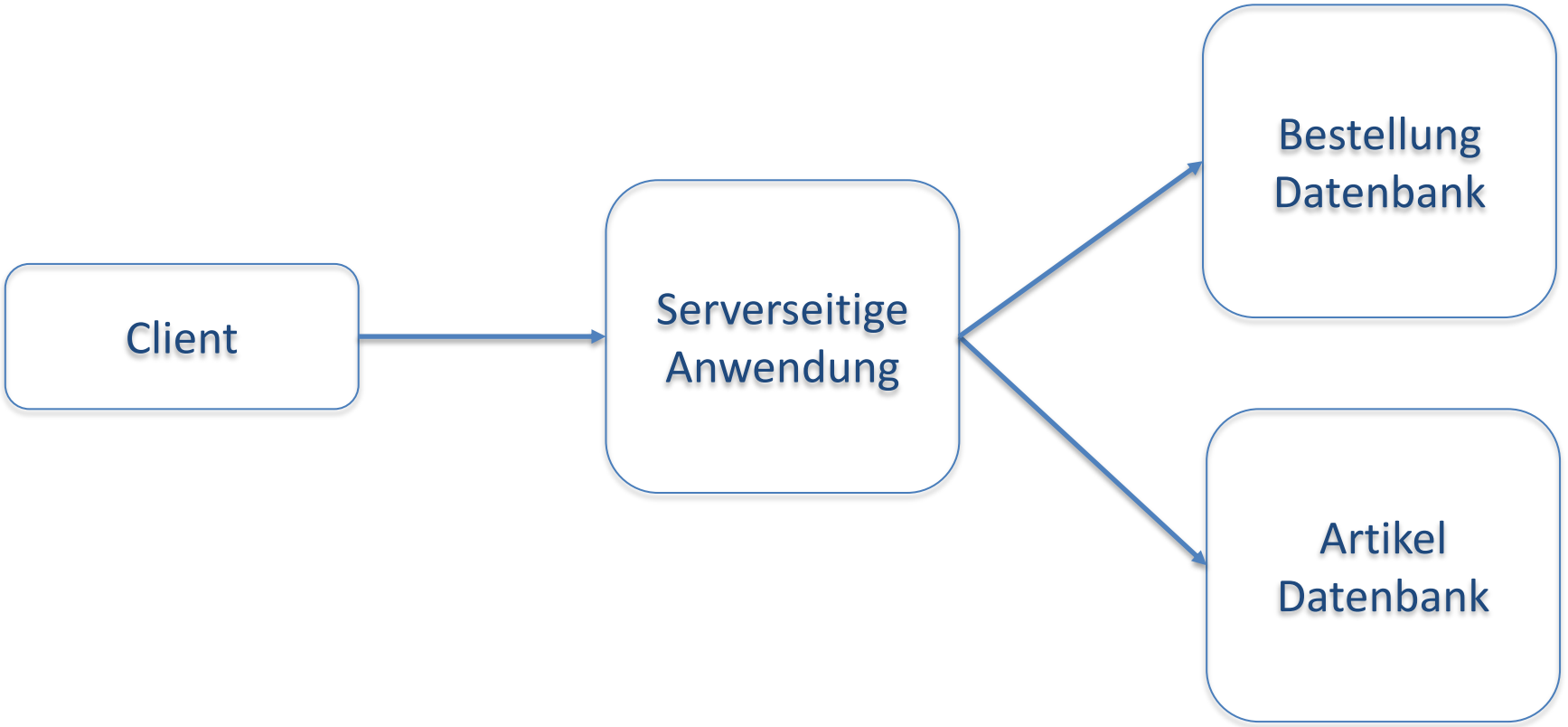
Prinzipien einer Serverless Architektur (1)

- Funktionalität ist in unabhängige, zustandslose Funktionen mit klarer Verantwortung geteilt
- Einzelne Funktionalitäten können isoliert und „nur bei Bedarf“ genutzt werden (Execution on demand)
- Verarbeitung erfolgt ereignisbasiert in Form von Verarbeitungspipelines

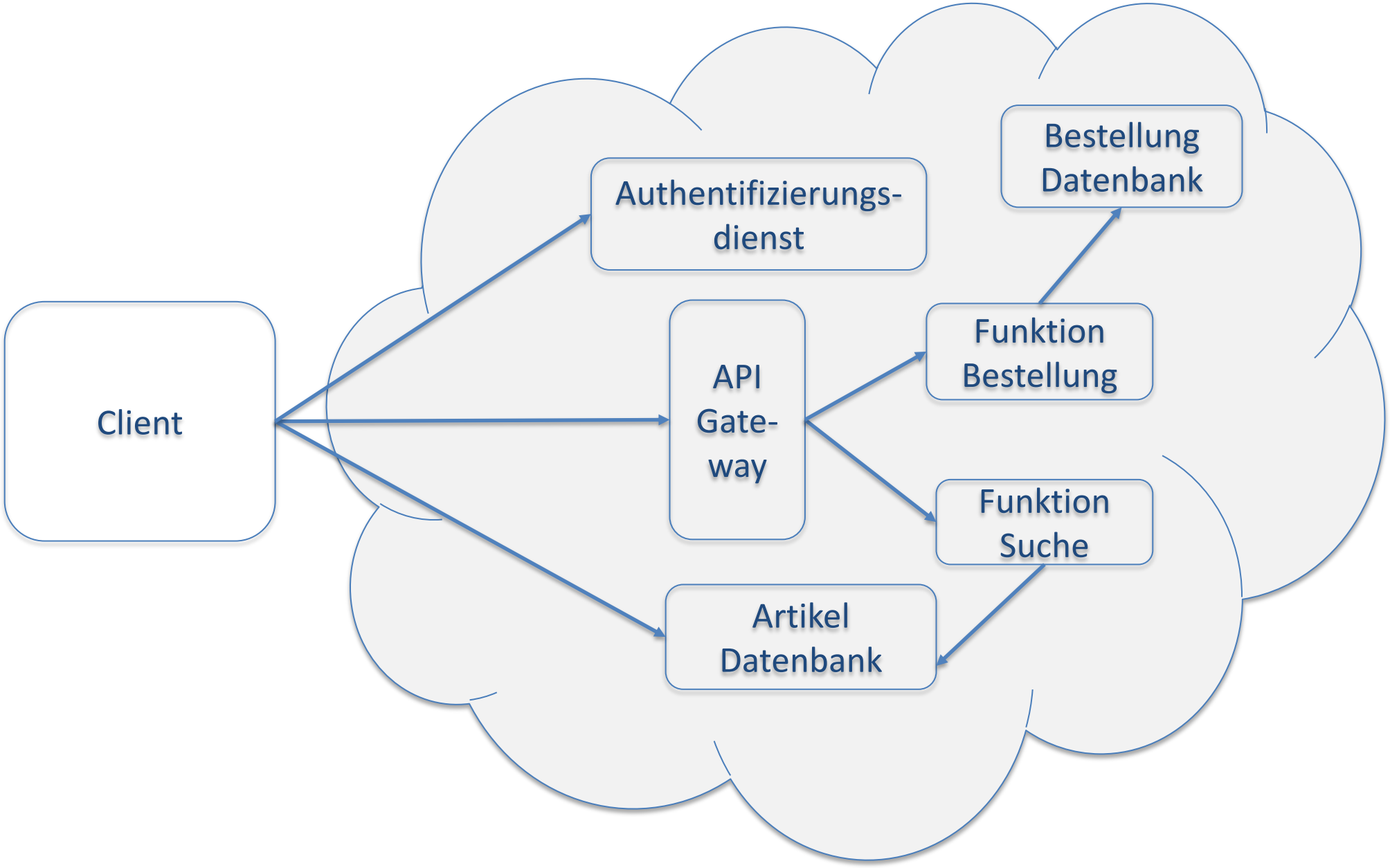
Prinzipien einer Serverless Architektur (2)

- Einsatz von Diensten von Drittanbietern ist möglich / gewünscht
- Mehr Logik und Verantwortung im Frontend (Thick Clients)

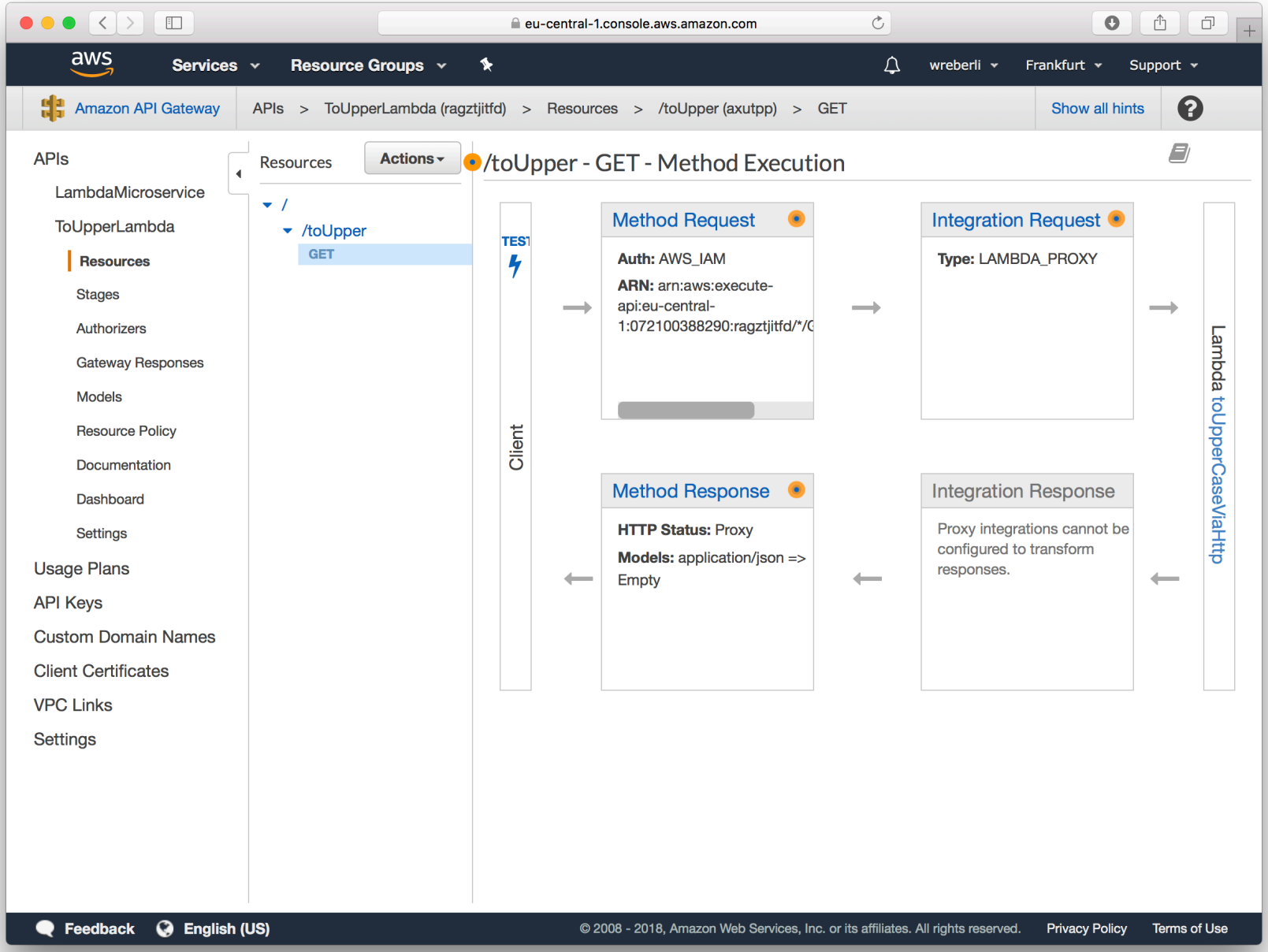
„Klassische“ Server Architektur



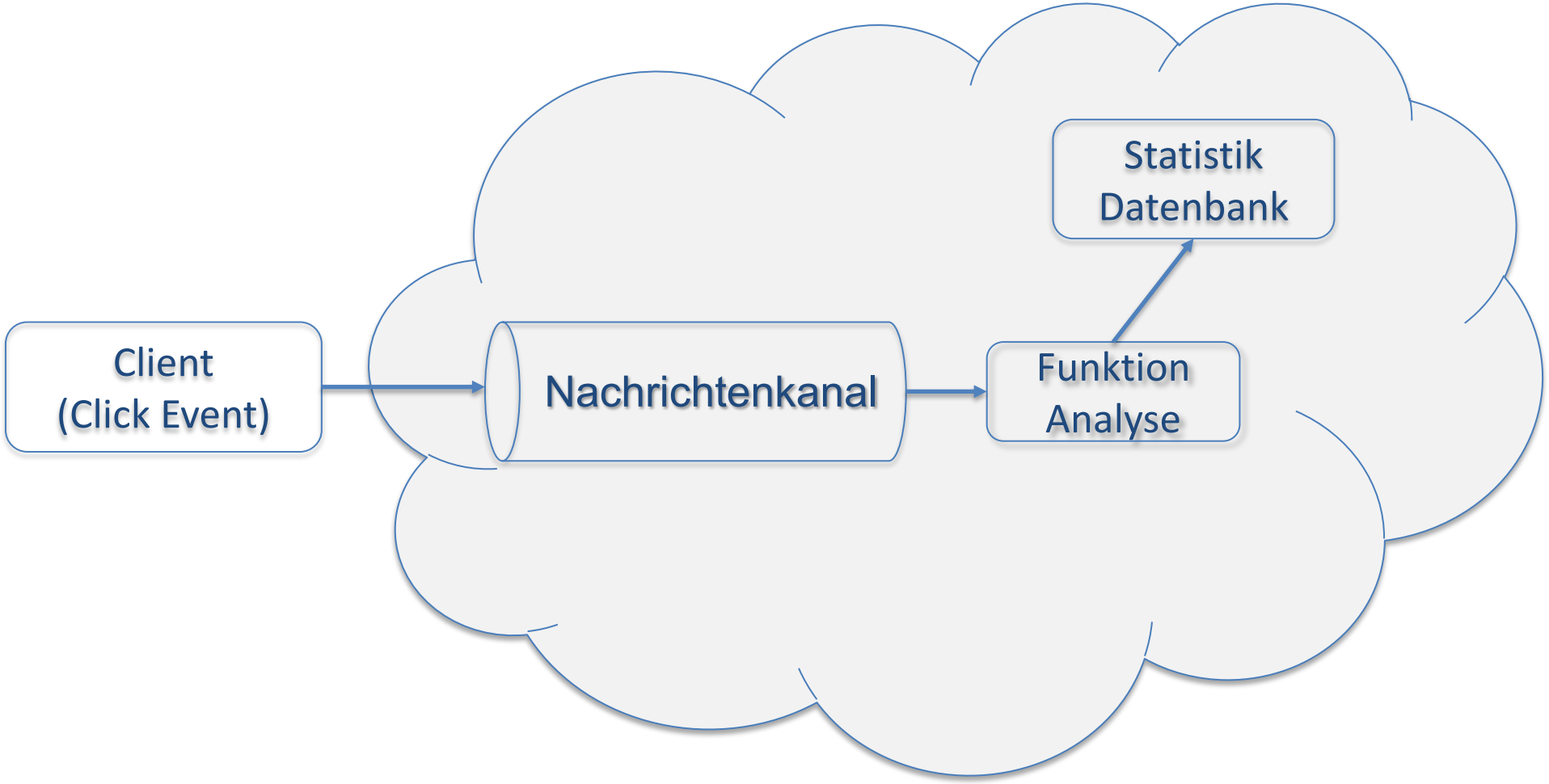
Serverless Architektur (UI-getrieben)



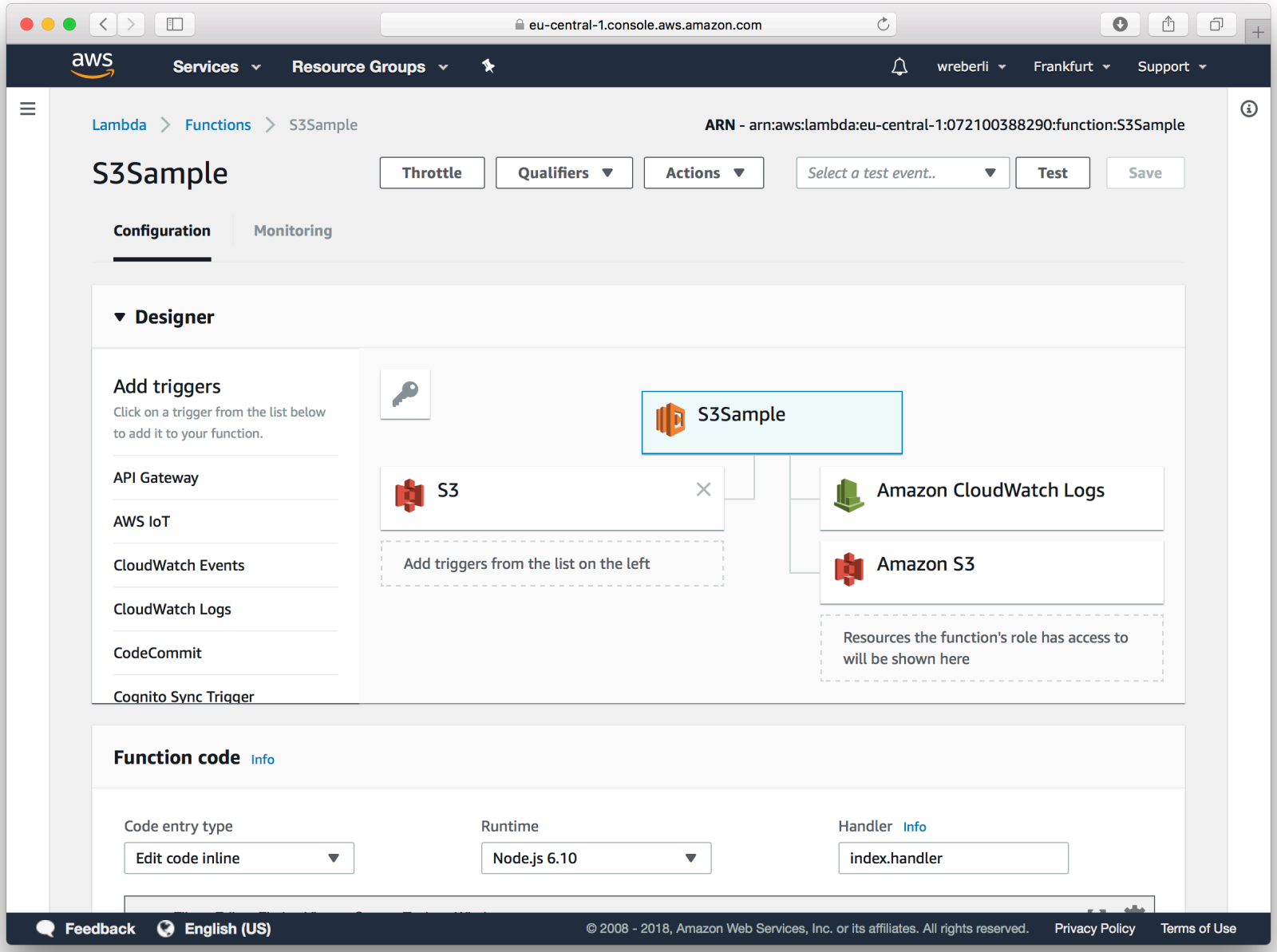
Beispiel



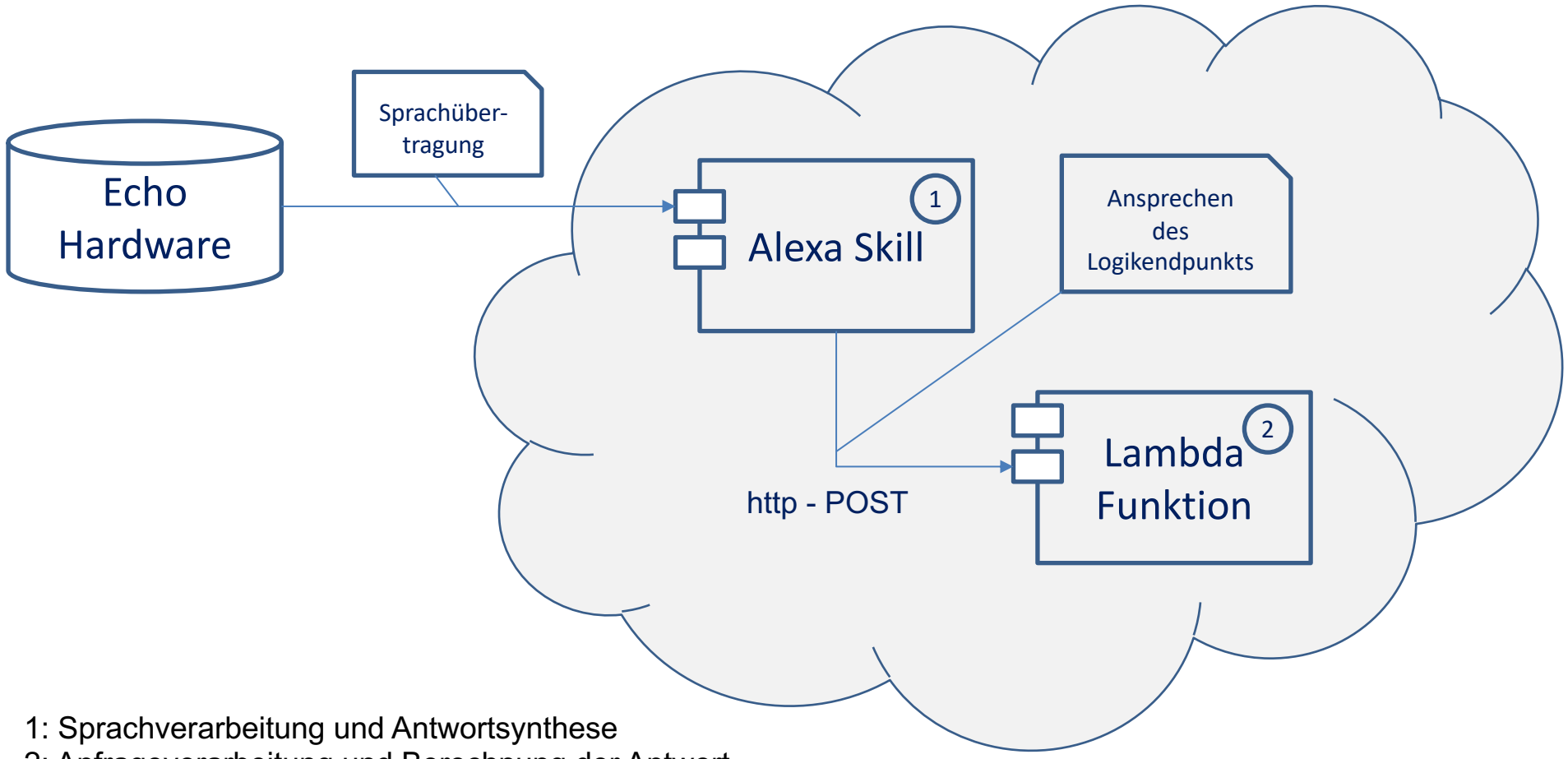
Serverless Architektur (Event-getrieben)



Beispiel

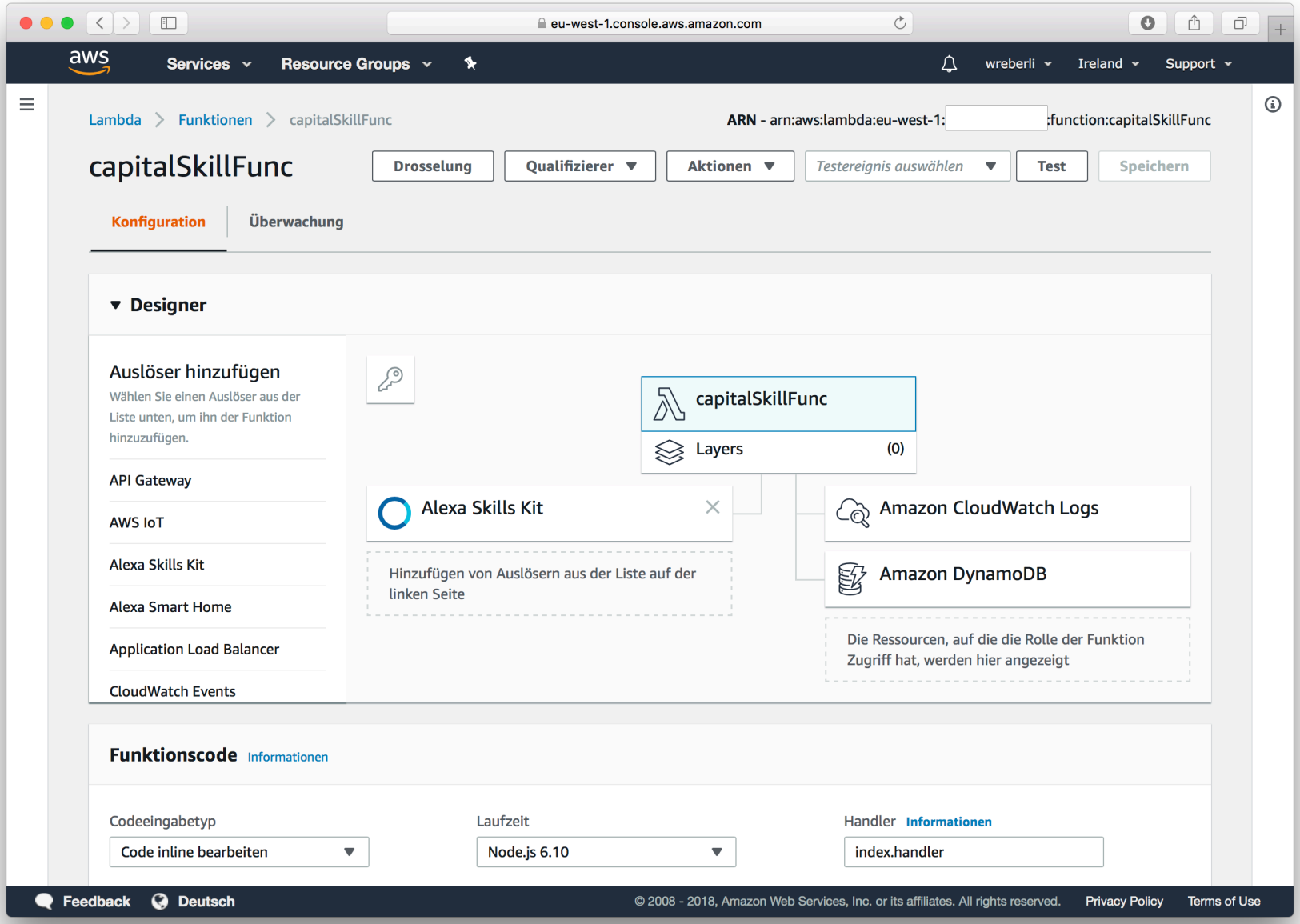


Serverless Architektur (für „neue“ Client-Szenarien)



1: Sprachverarbeitung und Antwortsynthese
2: Anfrageverarbeitung und Berechnung der Antwort

Beispiel



Vorteile einer Serverless Lösung

- Keine Server mehr
  - Schnelles Prototyping
- Geringere Kosten
  - „Don‘t pay for idle!“
- Weniger Code
  - Keine speziellen Bibliotheken oder Frameworks
- Skalierung und Verteilung „out of the box“

Nachteile einer Serverless Lösung

- Ablauf in einer fremden, öffentlichen Umgebung (Public Cloud)
- Nur begrenzte Service Zusicherungen (SLAs)
- Nur begrenzte Einflussnahme auf die Laufzeitumgebung
- Vendor Lock-In
- Inhärente Verteilung (aka. Dezentralisierung)

„The right tool for the right job“

- Serverless Ansätze sind nicht die Lösung aller Probleme!
- Serverless Ansätze sind gut für bestimmte Problemstellungen
- Hybride Lösungen nutzen die Vorteile ohne zu viele Nachteile in Kauf nehmen zu müssen
  - Bsp.: Log-Analyse, …

Einfach mal Ausprobieren

- Cloud Anbieter haben Angebote für
  - den Einstieg (z.B. die ersten 12 Monate)
  - Geringe Verarbeitungsvolumen (z.B. die ersten 1 Mio Aufrufe)

Fragen?

Vielen Dank!

werner.eberling@mathema.de

Twitter: @Wer_Eb

www.mathema.de

andrena
OBJECTS

Bitte geben Sie uns jetzt Ihr Feedback!

Wo wir hin wollen, brauchen wir keine Server – Serverless Programmieren mit AWS Lambda
Werner Eberling

Nächste Vorträge in diesem Raum

15:45 Vier Deployments für ein Halleluja – Aus dem Leben eines Softwaretherapeuten, *Michael Bruns*