think tecture

Cloud-Native-Workshop

Moderne End-to-End-Architekturen in der Praxis

Thorsten Hans

https://thinktecture.com/thorsten-hans
@thorstenhans

Consultant & Cloud-Native-Enthusiast

Christian Weyer

https://thinktecture.com/christian-weyer
@christianweyer

Co-Founder & CTO

Thorsten Hans

Cloud-Native Consultant @ Thinktecture AG

#Cloud-Native
#Kubernetes
#Azure
#Terraform

thorsten.hans@thinktecture.com

thinktecture.com
thorsten-hans.com

@ThorstenHans

Christian Weyer

Co-Founder & CTO @ Thinktecture AG

- Cloud-native & serverless architectures
- Pragmatic end-to-end solutions
- Mobile & web-based application architectures

- Independent Microsoft Regional Director
- Microsoft MVP for Developer Technologies
  ASPInsider, AzureInsider
- Google GDE for Web Technologies

christian.weyer@thinktecture.com @christianweyer https://www.thinktecture.com

Four Blocks – *approx. break down*

09.00 – 10.30

Intro
Story

... a bit booring stuff ...
Please bear with us!

11.00 – 12.30

Architecture
Technology
Code

13.30 – 15.00

Architecture
Technology
Code

15.30 – 17.00

Architecture
Technology
Code

Recap & Final Q&A

Special Day

Cloud-Native Business Applications

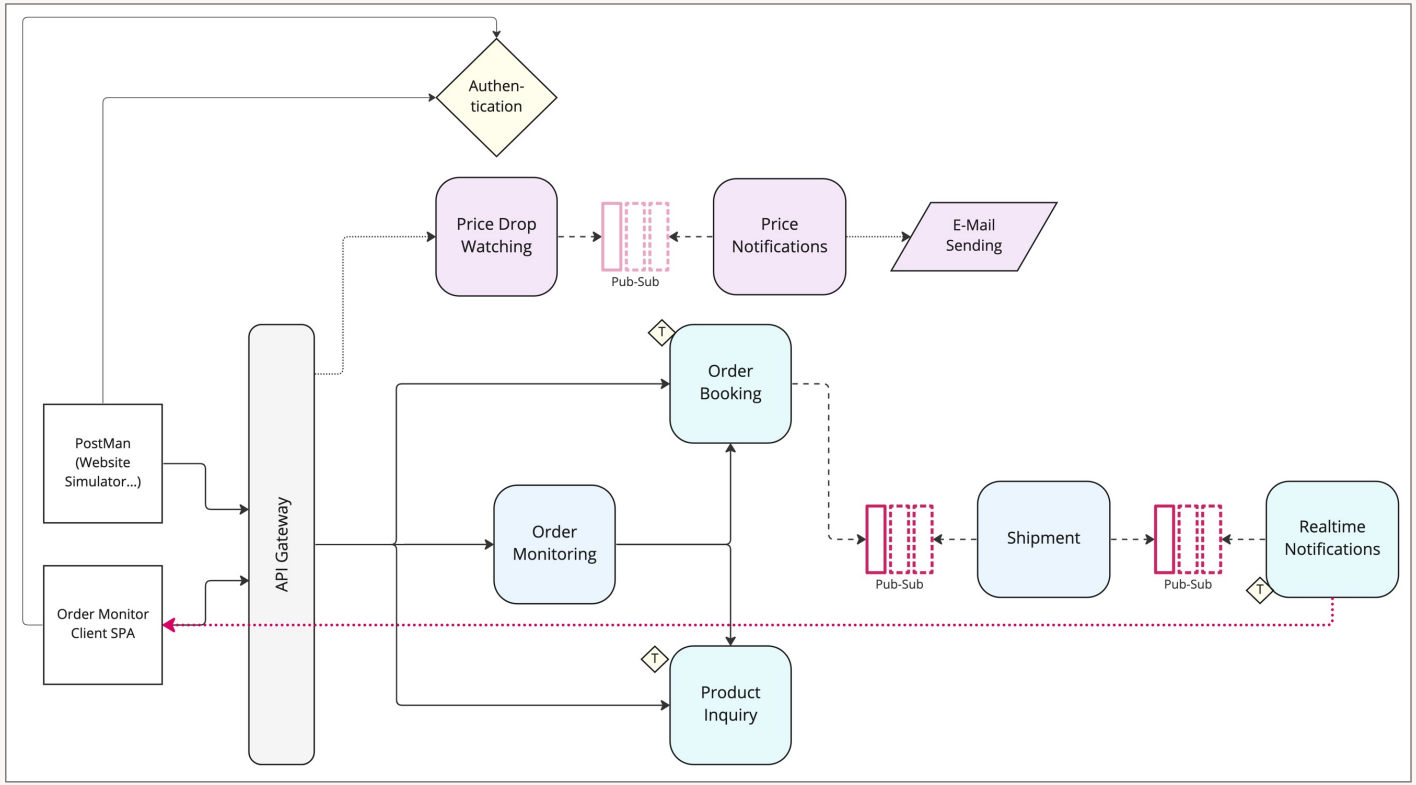
Thema	Sprecher	Datum, Uhrzeit
Cloud-Native-all-the-things: Definition, Praktiken & Patterns	Thorsten Hans, Christian Weyer	DI, 21. Februar 2023, 10.45 bis 11.45
Containerbasierte Entwicklung für .NET-Entwickler	Tobias Fenster	DI, 21. Februar 2023, 12.15 bis 13.15
Was guckst du? Observability von Cloud-Native-Anwendungen – mit OpenTelemetry	Thorsten Hans	DI, 21. Februar 2023, 15.30 bis 16.30
Cloud-Native Microservices: On-Premises oder in der Cloud – mit Dapr	Christian Weyer	DI, 21. Februar 2023, 17.00 bis 18.00
Serverless Containers mit Azure Container Apps	Thorsten Hans	DI, 21. Februar 2023, 19.00 bis 20.00

BLOCK I

Introduction

DEMO

End-to-End Demo Application

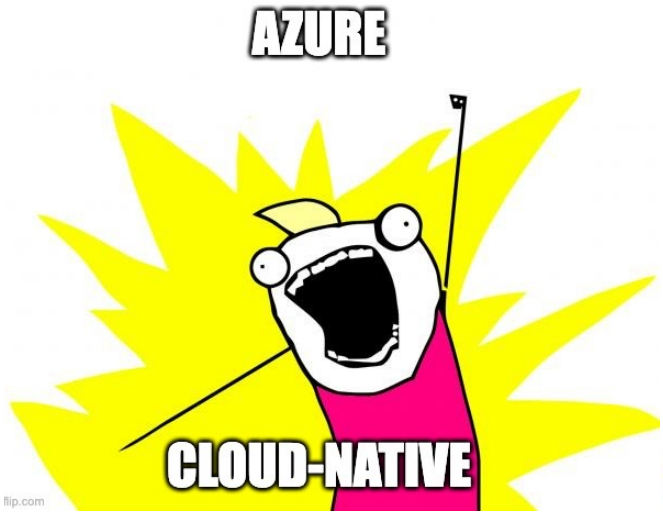


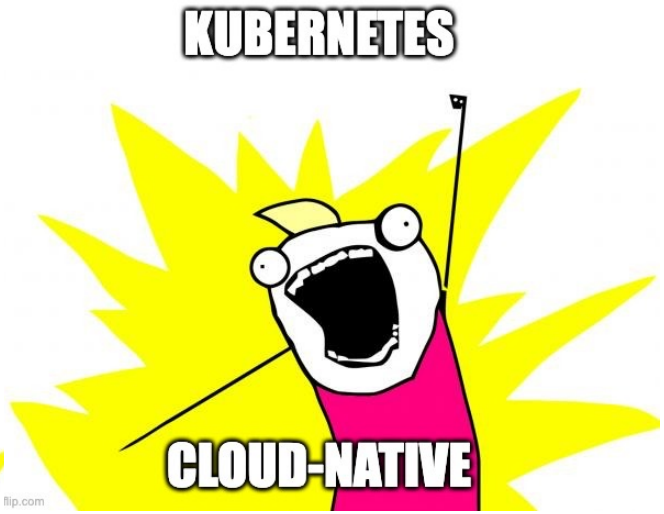
Myth Busting & Definitions

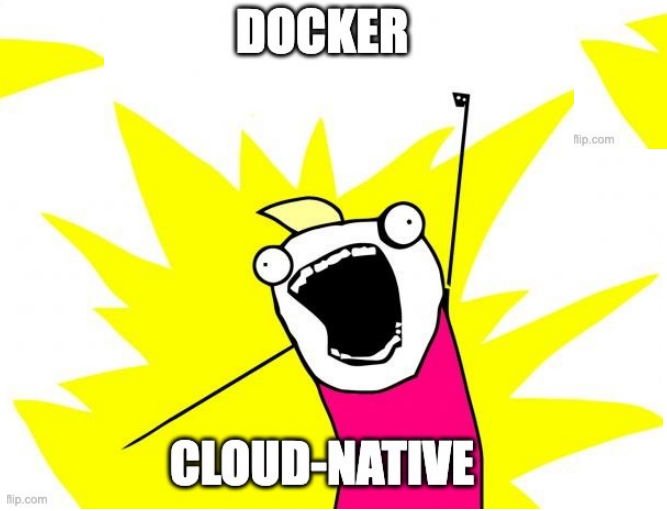
Aka “WTF?”

Cloud-Native

What is it NOT


AZURE
CLOUD-NATIVE
flip.com


KUBERNETES
CLOUD-NATIVE
flip.com


DOCKER
CLOUD-NATIVE
flip.com

Cloud-Native

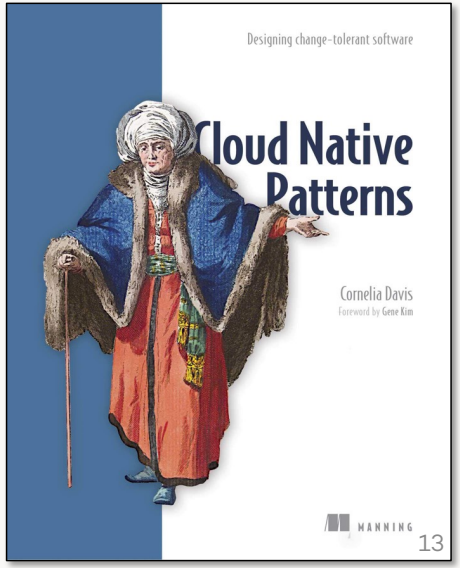
What is it

Cloud-Native Software

"... is highly distributed, must operate in a constantly changing environment, and is itself constantly changing."

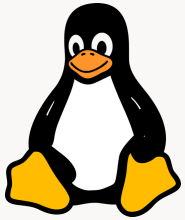
Cornelia Davis,

Author of
Cloud Native Patterns: Designing change-tolerant software



Cloud-Native Software

Yes, it is all about Linux.



Cloud-Native Software

Disadvantages of Windows Containers

- Bigger in size
  - longer push/pull times
  - Require more storage (affects Kubernetes & PaaS)
- Higher resource consumption
  - Windows Containers & Hosts need more CPU & memory
  - Direct impact on infrastructure budged
- Host & container OS must have same version (strict policy)
  - https://kubernetes.io/docs/concepts/windows/intro/#windows-os-version-support
  - Meaning you may end up with way more infrastructure

Cloud-Native

Candidate Applications

Candidates for Cloud-Native

- Strategic enterprise systems that need to constantly evolve business capabilities/features
- Application that requires a high release velocity - with high confidence
- System where individual features must release without a full redeployment of the entire system

- Application developed by teams with expertise in different technology stacks
- Application with components that must scale independently

- ℹ️ Smaller, less impactful LOB applications might fare well with a simple monolithic architecture hosted in a Cloud PaaS environment

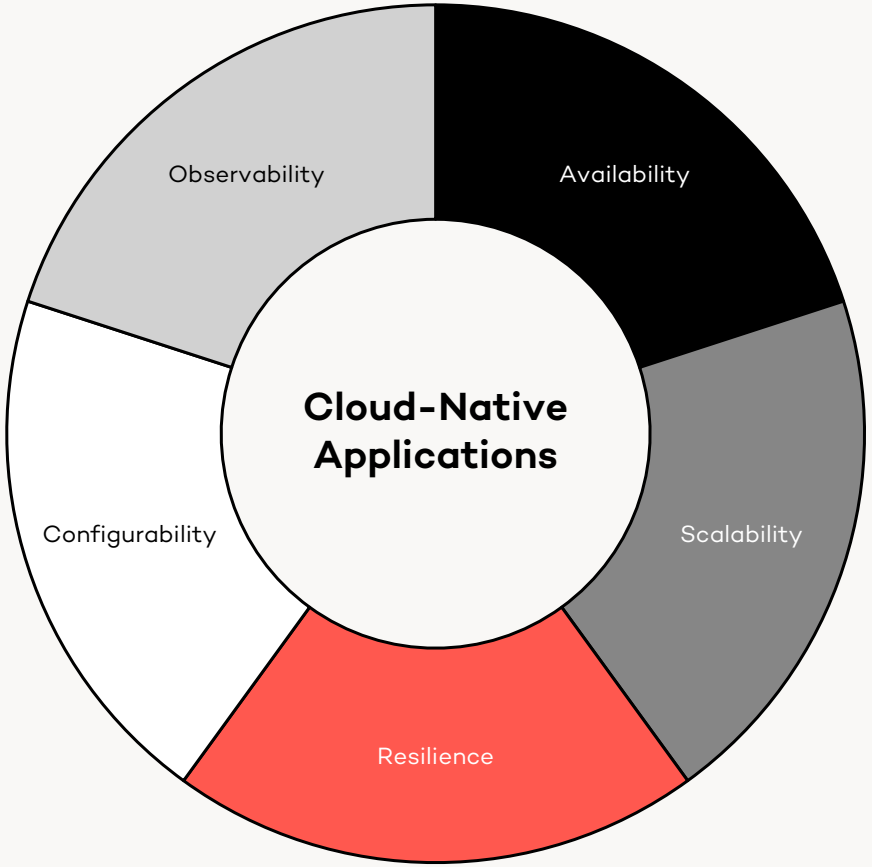
Cloud-Native

& Microservices

Cloud-Native Attributes

What & why

Cloud-Native Attributes



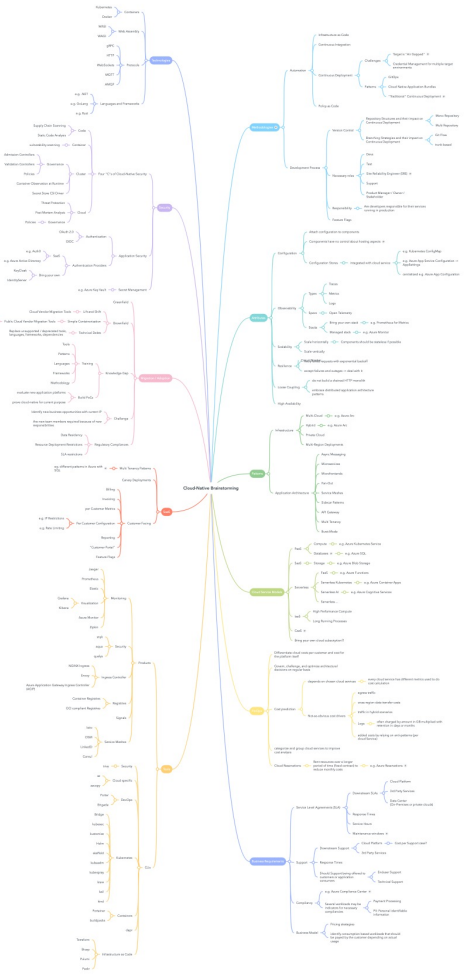
Cloud-Native Landscape

Complex affairs

think tecture

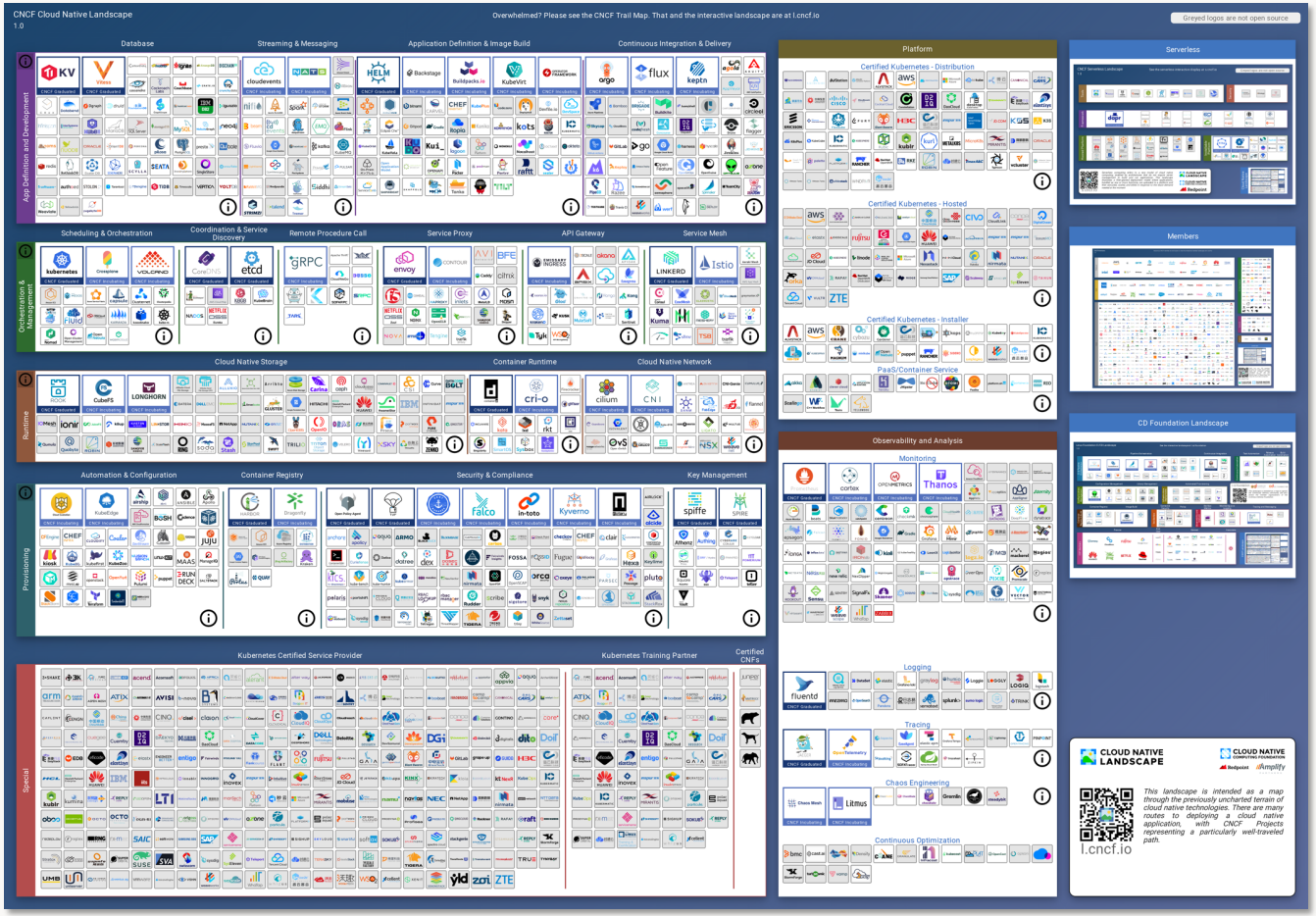
Cloud-Native Landscape

Our Brainstorming...



... there is a lot of infrastructure-related stuff !

Cloud-Native Landscape



https://landscape.cncf.io/

Geert Baeke
@GeertBaeke

Again, to business leaders and executives:

- cloud ☁️ is not easy
- do not underestimate the infrastructure pieces ‼️
- do not underestimate the automation part 🏭
- do not underestimate the size of teams and required knowledge 👬

Especially when it's enterprise scale!

#cloud

6:58 PM · Oct 5, 2022 · Twitter for iPhone

https://twitter.com/GeertBaeke/status/1577704842541285379

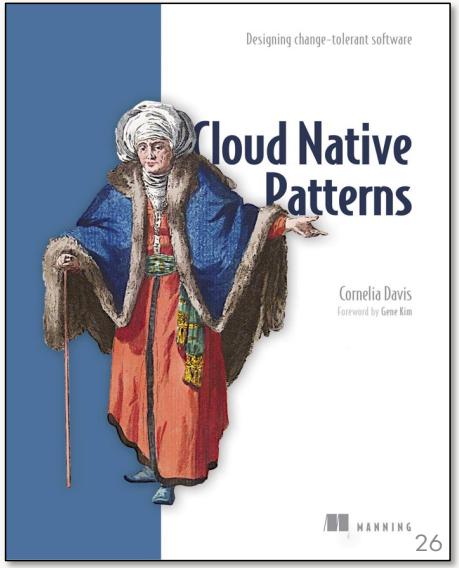
Cloud-Native & Cloud

What's in a name?

"Cloud is about where we're computing. Cloud-Native is about how."

Cornelia Davis,

Author of
Cloud Native Patterns: Designing change-tolerant software



From

Build vs. Buy

to

Run vs. Rent

Cloud Agnosticism

Cloud Agnostism

Going fully cloud-agnostic

leads to

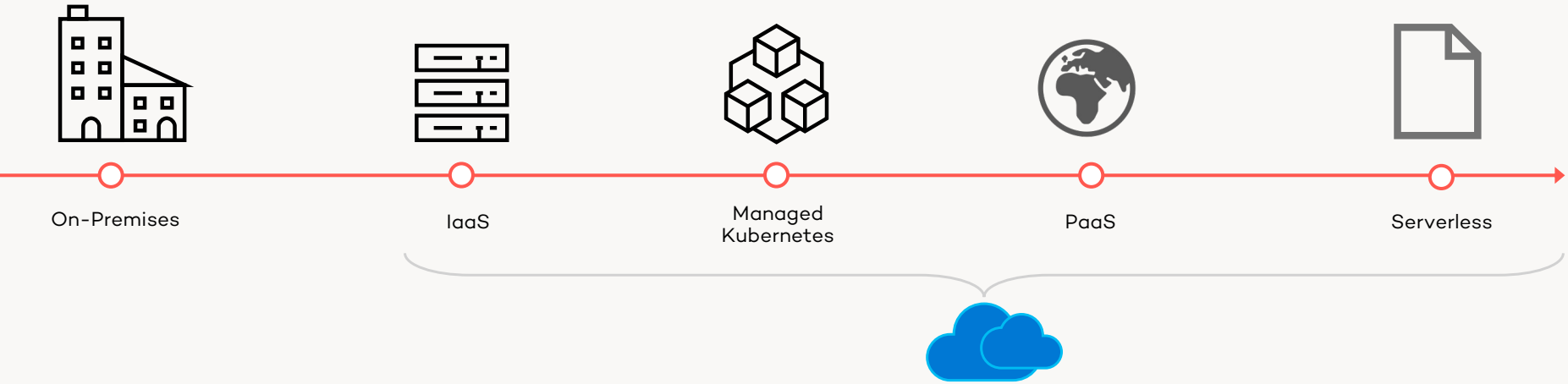
re-implementing stuff that's already there

Hosting Options

Running Cloud-Native applications

Cloud Compute Continuum

- We can choose and mix from the continuum



Cloud-Native & Kubernetes

- Cloud-Native is not “just using” Kubernetes
- Kubernetes is a powerful platform (orchestrator) to build and run cloud-native applications
- Leverage Kubernetes patterns and capabilities to address certain cloud-native “ilities”
- Kubernetes is a driver to become cloud-agnostic

Hosting Options - Comparison

Criteria / Hosting Model	Mixed-Mode	Docker Compose	Self-Hosted Kubernetes	Local Managed Kubernetes	Managed Kubernetes	Platform as a Service	Platform as a Service with Containers	Serverless Containers
Description	Run Application components straight on the machine / Run Dependencies as Containers	Run everything with Docker Compose (Development)	Run (at least your application components) in a self-hosted Kubernetes	Run (at least your application components) in a managed Kubernetes	Run (at least your application components) in a managed Kubernetes	Run (at least your application components) using a PaaS offering	Run (at least your application components) using a PaaS offering	Run (at least your application components) on a serverless container
Location	OnPremises	OnPremises	OnPremises	Private Cloud	Public Cloud	Private & Public Cloud	Private & Public Cloud	Public Cloud
Expenditure Model	CapEx	CapEx	CapEx	CapEx	OpEx	OpEx	OpEx	OpEx
Infrastructure Security Responsibility	User	User	User	User & Vendor	Vendor	Vendor	Vendor	Vendor
Platform Upgrades	User	User	User	User & Vendor	Vendor	Vendor	Vendor	Ve
Maintenance	1	1	1	2	3	3	2	3
Responsibility	1	1	1	2	2	2	2	3
Control	3	3	3	2	2	2	2	1
Deployment Model	Containers + Executables	Containers	Containers	Containers	Containers	Platform specific	Containers	Containers
Homogenous Deployment Model	2	3	3	3	3	1	2	2
Complexity	2	2	1	2	2	3	3	3
Vendor Lock-In	3	3	3	3	3	1	2	1
Scalability: Application	1	1	3	3	3	2	2	3
Scalability: Platform	1	1	2	2	3	3	3	3
Reliability: Application	0	1	3	3	3	2	2	3
Reliability: Platform	1	1	2	3	3	3	3	3
Availability	1	1	2	2	3	3	3	3
Automation: Infrastructure	2	2	1	1	3	3	3	3
Automation: Application Deployment	1	3	3	3	3	3	3	3
Freshness: Infrastructure	1	1	2	2	2	3	3	3
Freshness: Operating System	1	1	1	1	2	3	3	3

0 = worst / 3 = best

BLOCK II

Applying proven techniques to .NET applications

DEMO

Main Metaphor for Cloud-Native Code

"It's simply that the application has to give up a lot of control, to the platform - and has to be cleanly integrable from the outside."

Thorsten Hans,

Cloud-Native-Enthusiast
Thinktecture Consultant

.NET by Example

Configurability by Example

with IConfiguration

https://docs.microsoft.com/en-us/aspnet/core/fundamentals/configuration/

Observability by Example

Individual Health Checks

https://docs.microsoft.com/en-us/aspnet/core/host-and-deploy/health-checks

Configuration by Example

Integrate with the network environment

https://docs.microsoft.com/en-us/aspnet/core/host-and-deploy/proxy-load-balancer

Redefined Responsibilities

E.g.: Application devs
do not configure HTTPS
in the application

Observability by Example

with ILogger

https://docs.microsoft.com/en-us/aspnet/core/fundamentals/logging/

Observability by Example

with OpenTelemetry

https://opentelemetry.io/docs/instrumentation/net/getting-started/

Observability by Example

with Prometheus

https://github.com/prometheus-net/prometheus-net

Containerization

Create & use Docker Images

BLOCK III

Cloud-Native Patterns & Implementations

Cloud-Native applications - Some tedious parts

- Execution independence / Hosting-agnostic
  - Different programming / tech stacks
  - Running anywhere
- Resiliency
  - Location transparency
  - Error handling, retries
- Decoupling
  - Robustness
  - Async business processes
- Observability
  - Seeing everything end-to-end
  - Tracing, logging, metrics

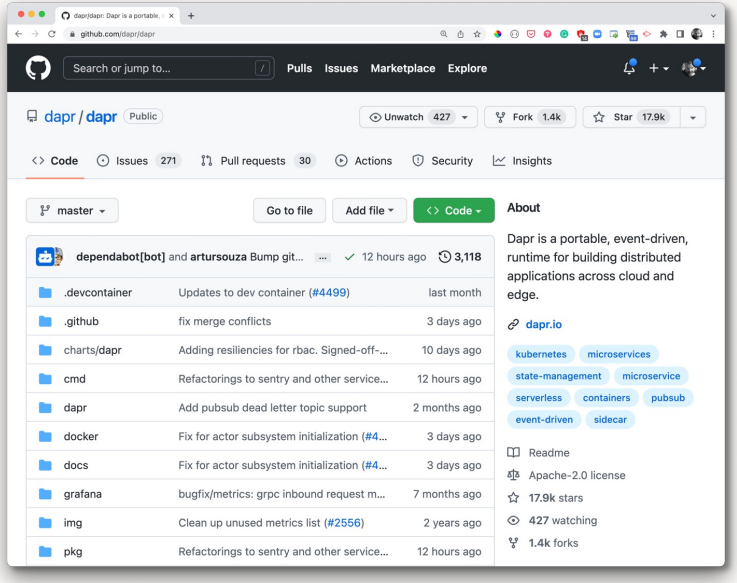
💡 Nice to have help here

- Proven patterns
- Support by libraries, SDK, runtime

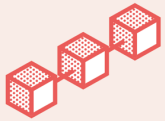
Dapr: Distributed Application Runtime - for Cloud-Native

- Portable, event-driven runtime for building distributed applications
  - Open-source, community-driven, vendor-neutral
- Prevents developers reinventing the wheel, esp. complex wheels
  - Making developers lives easier with consistent approach / abstraction
- Productivity tool, when having different
  - Generations of software types (greenfield and brownfield)
  - Languages & frameworks
  - Team formations

  ➡️ Dapr as a common lingua franca



Dapr Building Blocks

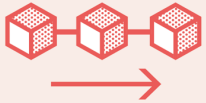


Service-to-service invocation

Perform direct, secure, service-to-service method calls

State management

Create long running, stateless and stateful services



Publish and subscribe

Secure, scalable messaging between services

Bindings (input/output)

Trigger code through events from a large array of inputs

Actors

Encapsulate code and data in reusable actor objects as a microservices design pattern

Observability

See and measure the message calls across components and networked services

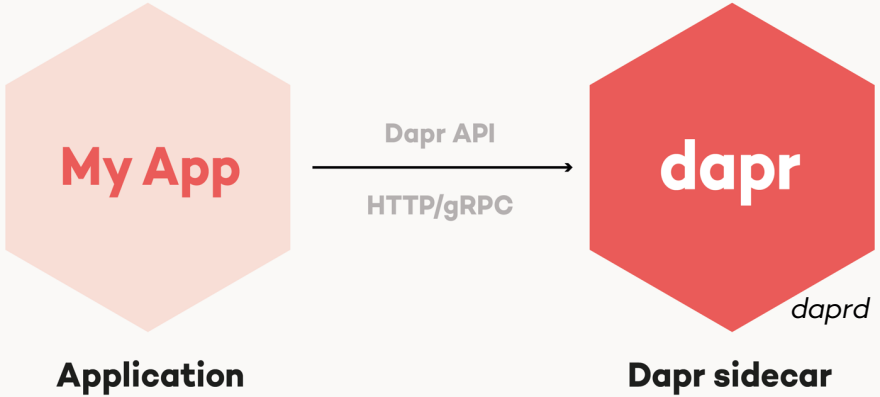
Secrets

Securely access secrets from your application

Configuration

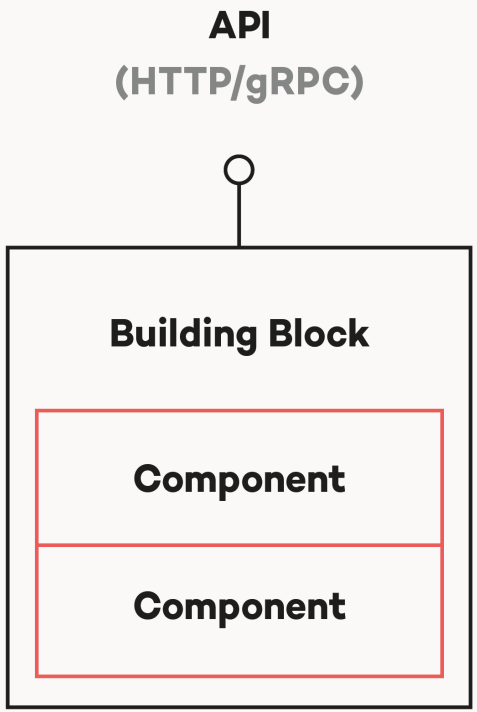
Access application configuration and be notified of updates

Dapr Sidecar: Separating Concerns



POST	http://localhost:3500/v1.0/invoke/cart/method/neworder
GET	http://localhost:3500/v1.0/state/cart/inventory/item67
POST	http://localhost:3500/v1.0/publish/shipping/orders
GET	http://localhost:3500/v1.0/secrets/keyvault/password

Components: Abstracting Access to “Things”



Resource bindings (input/output)

&others


EventHub


Kafka


AWS SQS


GCP

State Stores

&others


dynamoDB


CosmosDB

redis

Cassandra

Publish & Subscribe

&others

redis


Nats


rabbitMQ


Service Bus

Distributed Tracing

&others


Prometheus

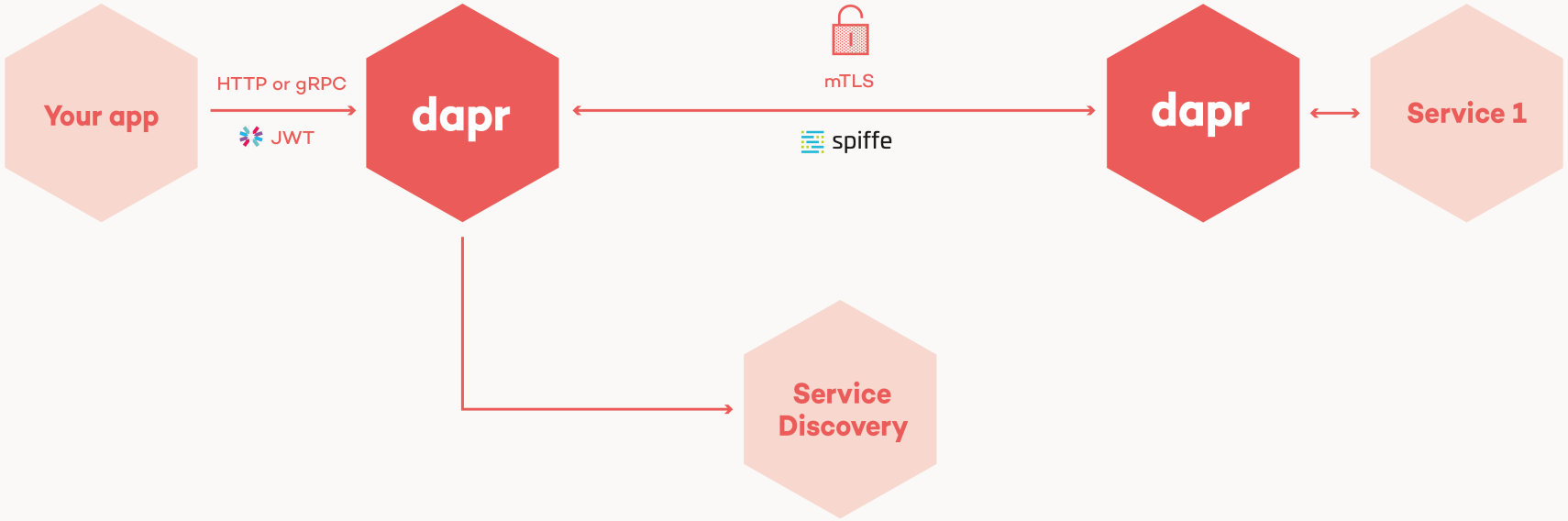

AppInsights

Zipkin

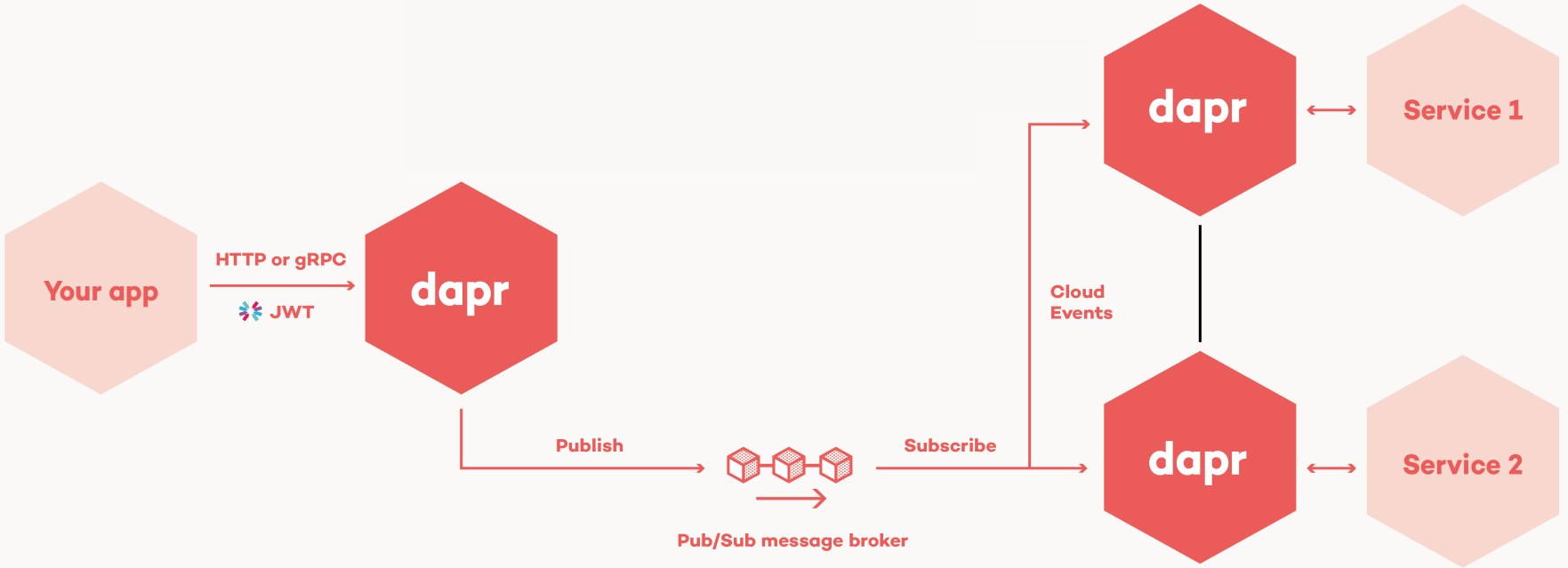

Jaeger

https://github.com/dapr/components-contrib/tree/master/pubsub

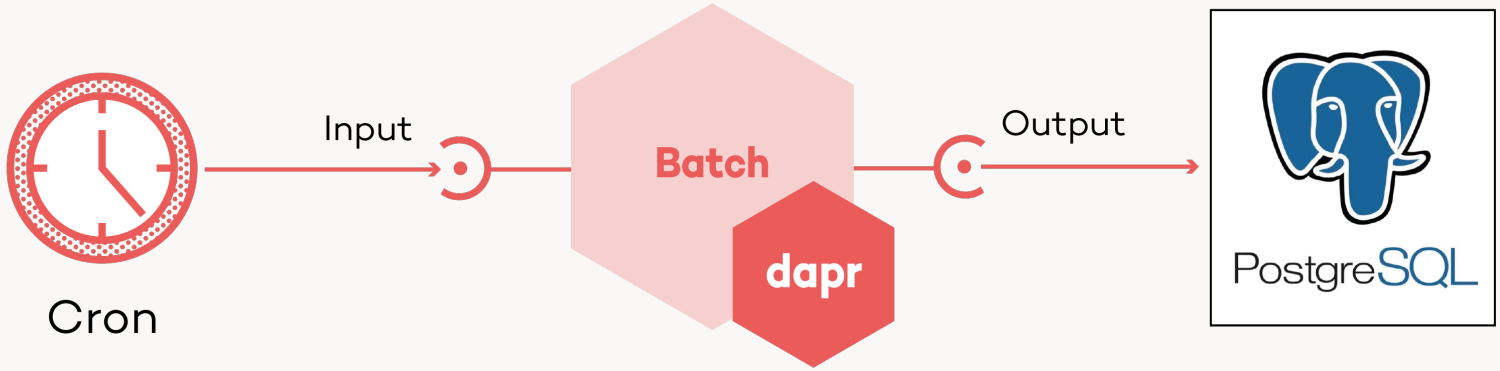
Service Invocation: Interception



Publish & Subscribe: Decoupling



Input / Output Bindings & Triggers



BLOCK IV

Automation in Cloud-Native

Automation in Cloud-Native

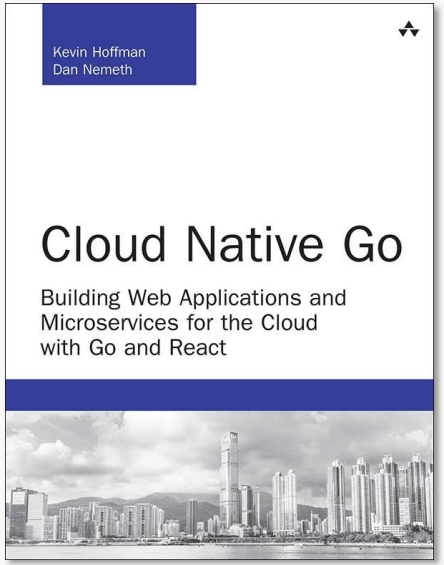
The key to success!

Automate EVERYTHING

“Everything that can be automated, should be automated.”

Kevin Hoffman & Dan Nemeth,

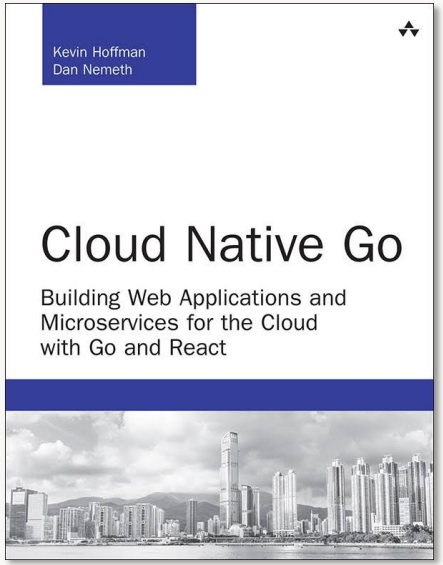
Author of
Cloud Native Patterns: Designing change-tolerant software



“Anything you do more than once per day is a candidate for automation.”

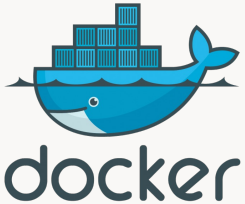
Kevin Hoffman & Dan Nemeth,

Author of
Cloud Native Patterns: Designing change-tolerant software

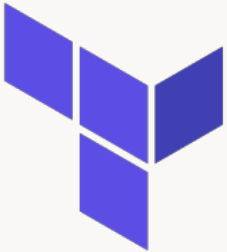


Automation Layers

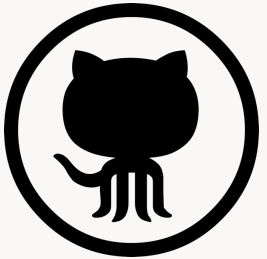
Inner-Loop



Infrastructure



CI & CD



Inner-Loop

Automate everything you need to do while developing locally

Infrastructure

Automate your cloud-infrastructure by applying Infrastructure-as-Code

Continuous Integration

Build, Test, and Package every application component independently and automatically

Deployment

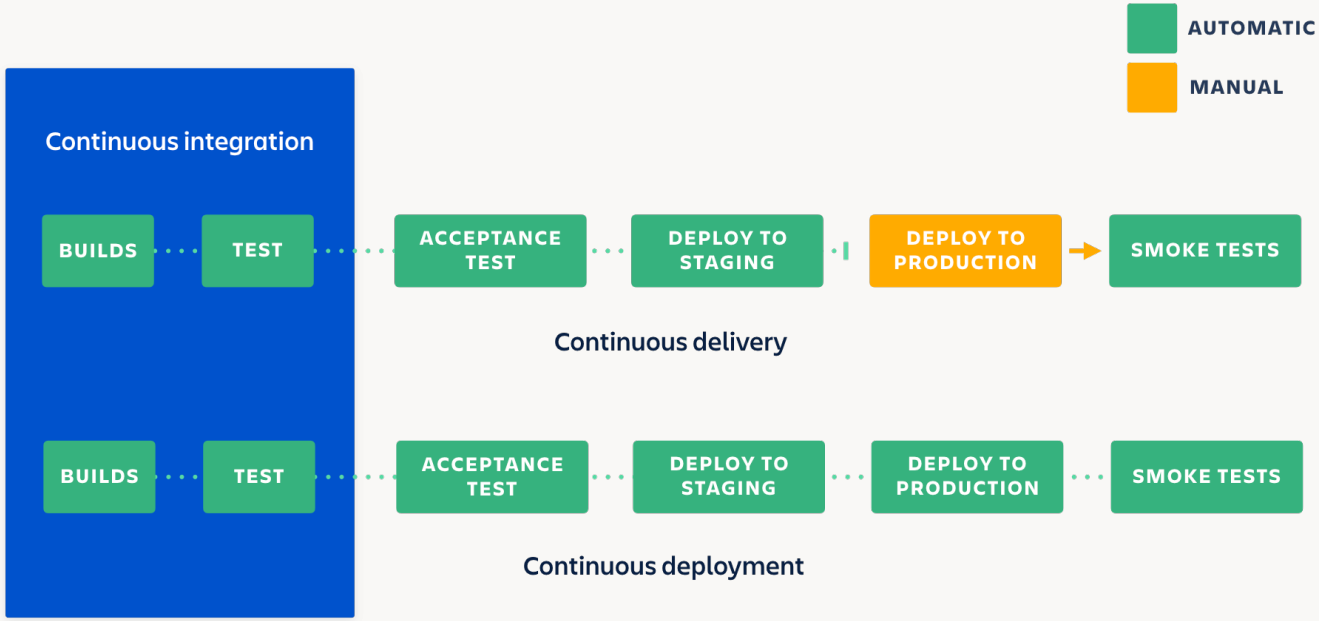
Deploy every component independently and automatically

Deployment

Embrace either

Continuous Deployment or GitOps

Continuous-all-the-things



https://www.atlassian.com/continuous-delivery/principles/continuous-integration-vs-delivery-vs-deployment

Recap

Summary

- Was Sie gesehen haben
  - **Pragmatische** Sicht auf Cloud-Native
    - Was? Warum? Wo? Wie?
  - **Methodiken, Praktiken & Patterns**
  - Verteilte Anwendungen mit **Dapr**
  - **Automatisierung** auf allen Ebenen
  - End-to-End **Beispielanwendung**
    - .NET-Code (und ein bisschen Go)
    - Lokal und in der Cloud
    - Terminal & CLIs

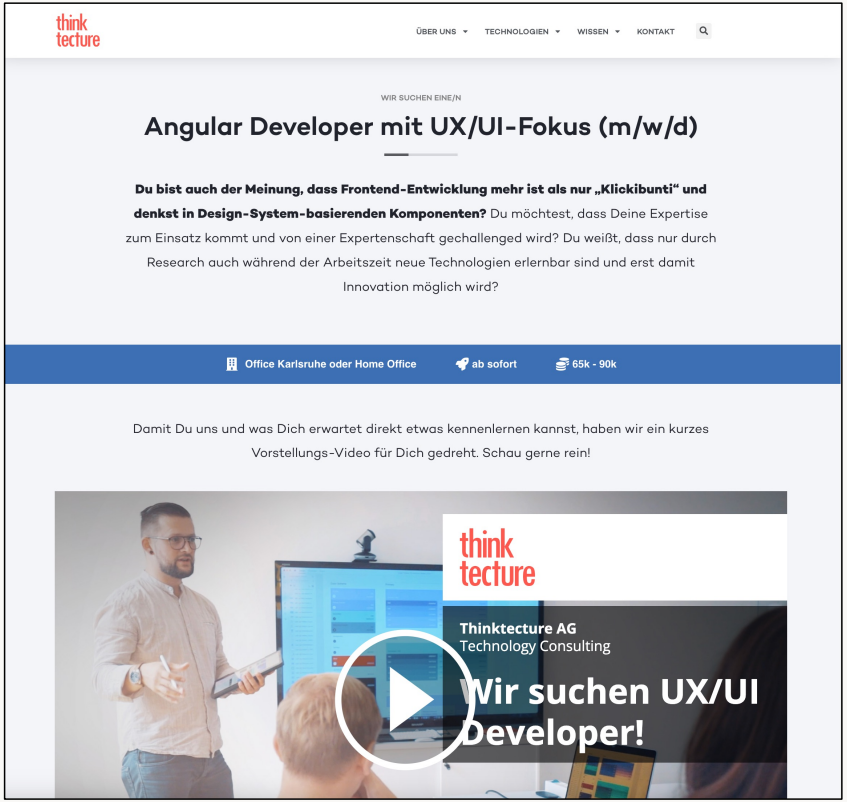
- Was Sie nicht gesehen haben
  - Einführung in Docker, Kubernetes oder Azure
  - Deep Dives in Docker, Kubernetes oder Azure
  - Windows 🙈
  - Klick-Bunti in Visual Studio 😇
  - Businessaspekte, wie Business Domain Modeling oder SaaS
  - Cloud-Native Security

Dankeschön!

Demos aus der Session:
https://github.com/thinktecture-labs/cloud-native-sample

Cloud-Native @ Thinktecture:
https://www.thinktecture.com/technologien/cloud-native/

https://www.thinktecture.com/wissen/



https://www.thinktecture.com/ueber-uns/karriere/

Thorsten Hans
https://thinktecture.com/thorsten-hans

Christian Weyer
https://thinktecture.com/christian-weyer