

16.07.2020
JUG Darmstadt



Service Mesh

was die neue Infrastruktur für Microservices taugt

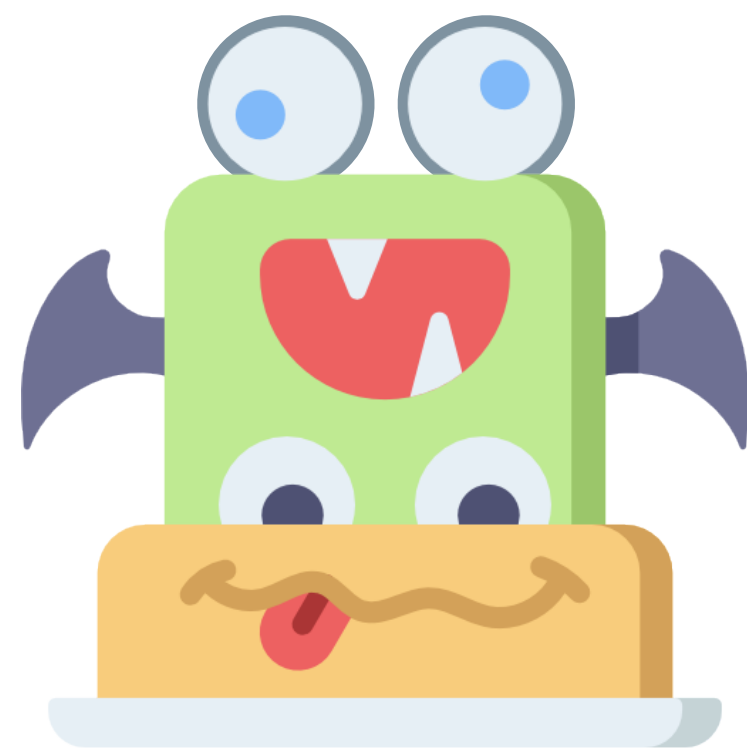
INNOQ

Hanna Prinz

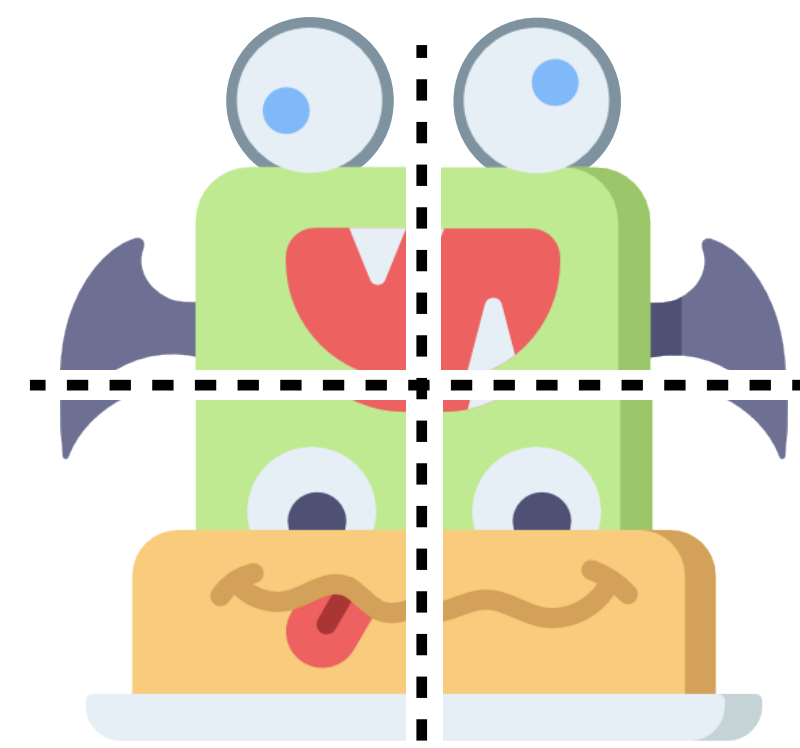
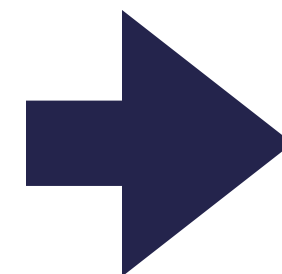
~
**"Fix your
Microservices
by throwing a
Mesh at it!"**
~



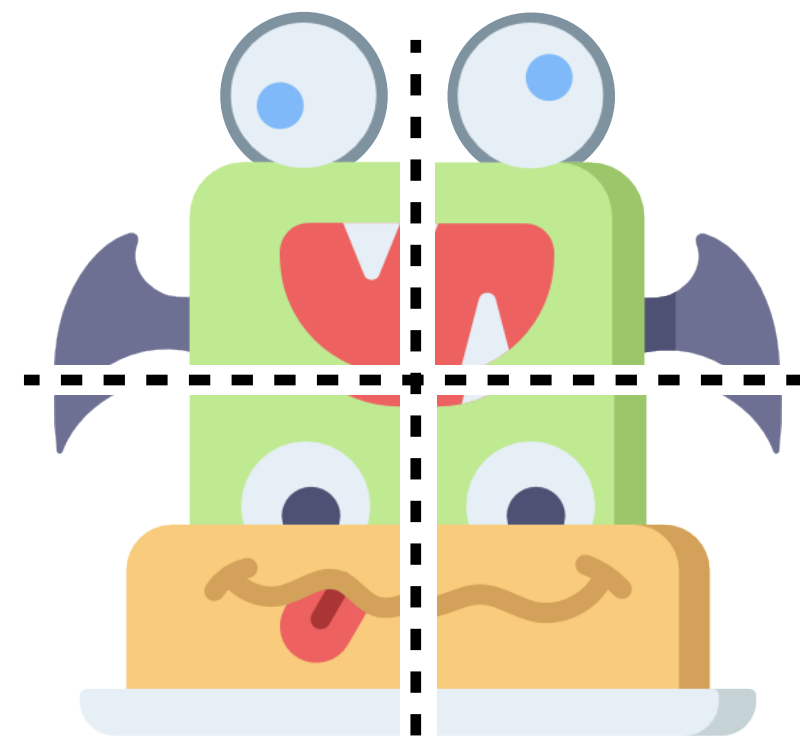
**Wie ist es so
weit gekommen?**



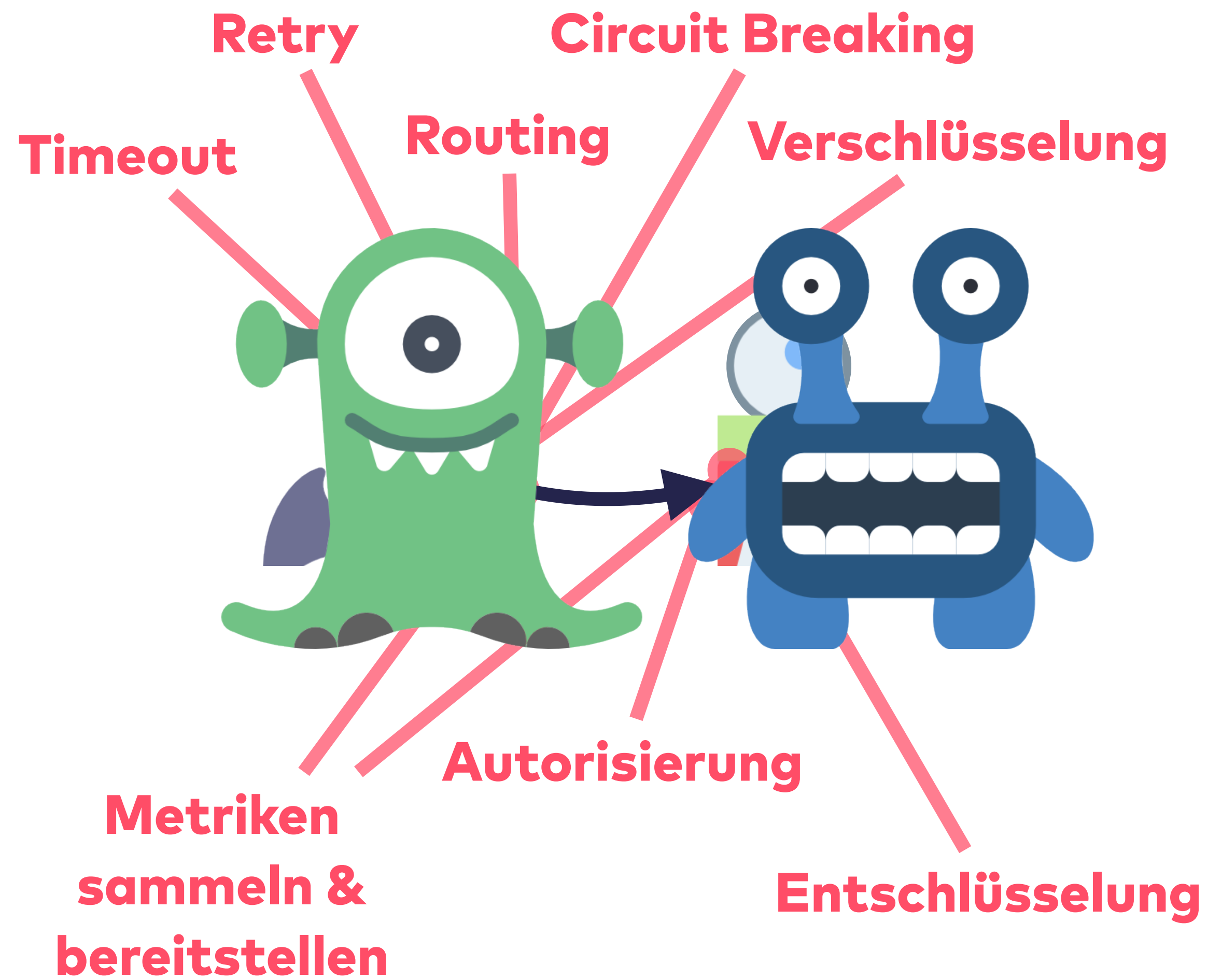
Monolith



Microservices

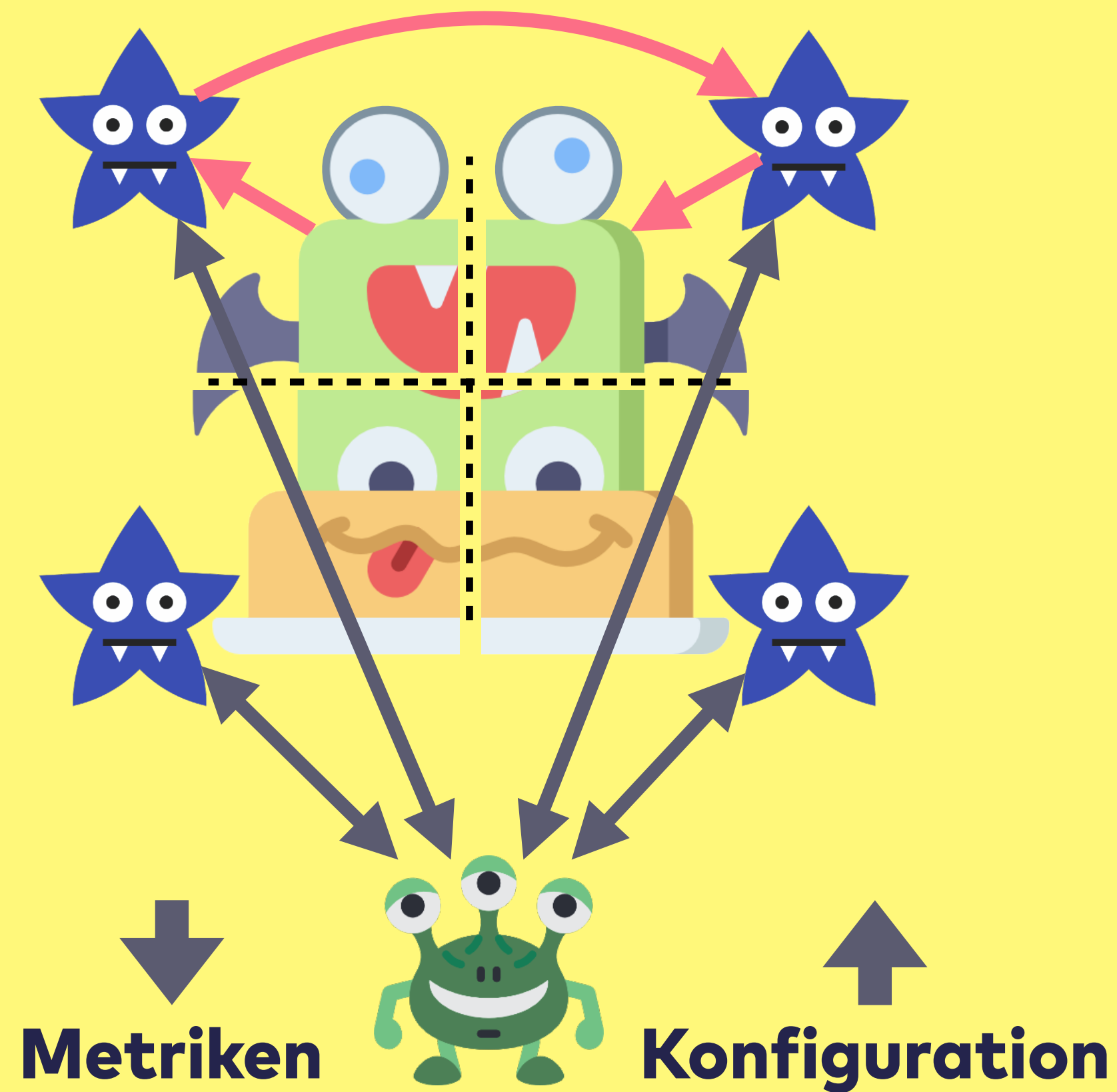
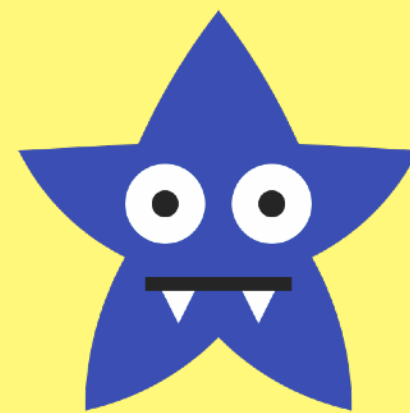


Microservices

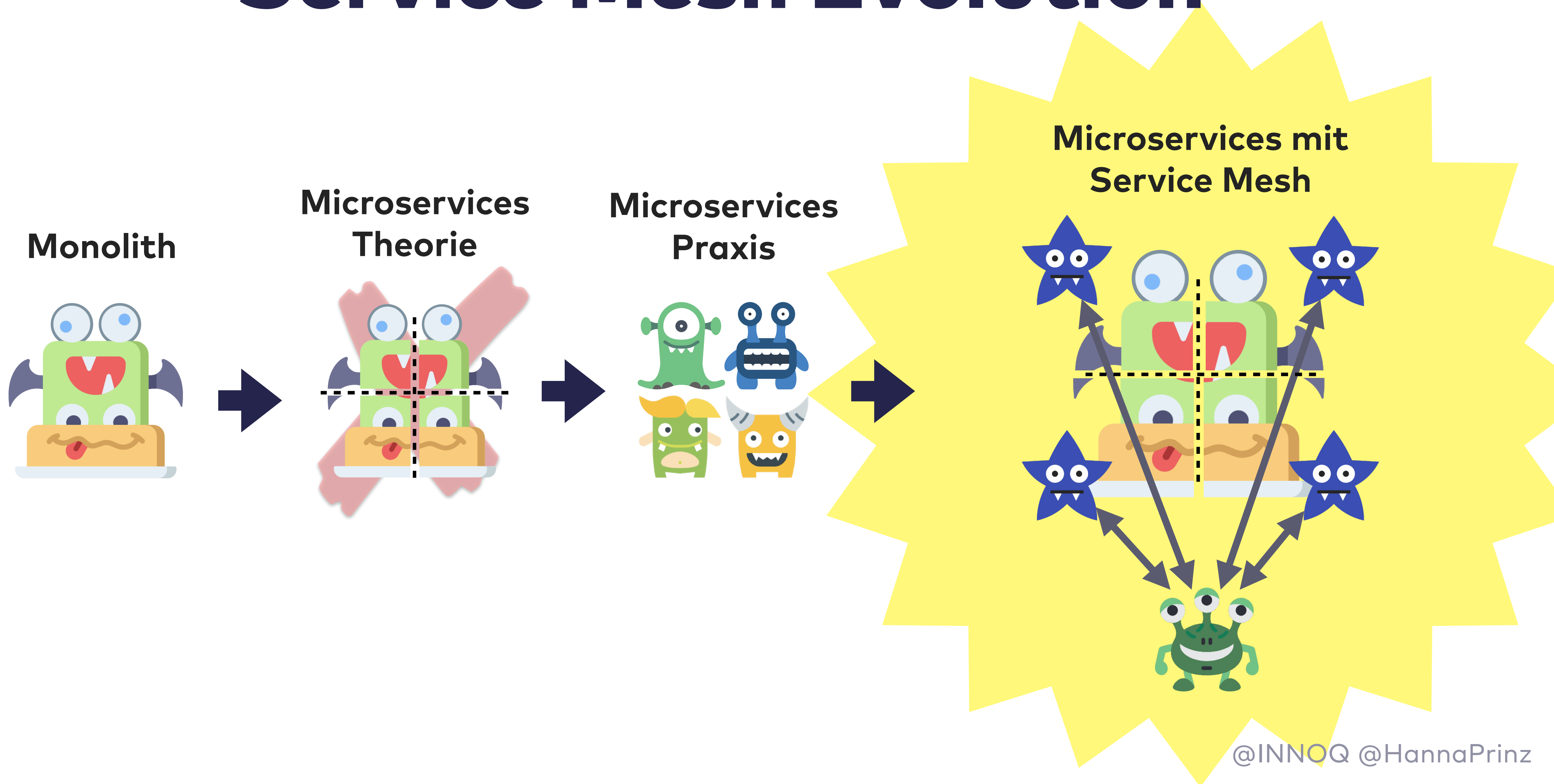


Service Mesh

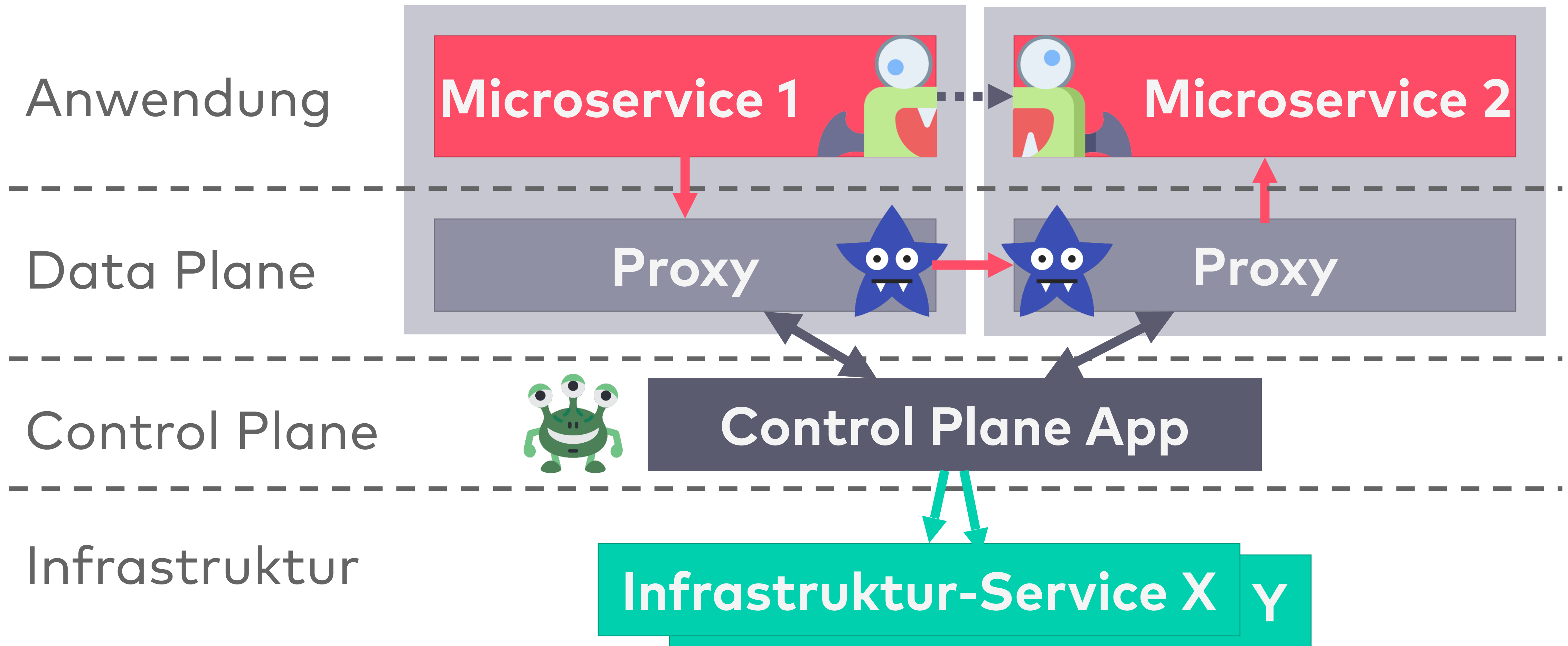
Retry
Timeout
Circuit Breaker
Routing
Verschlüsseln
Entschlüsseln
Autorisierung
Metriken
...



Service Mesh Evolution



Service Mesh Architecture



Hurra, Technologie!

Service Mesh Features



Observability



Routing



Resilience



Security

Monitoring

Ein Service Mesh kann automatisch
alle 4 "Golden Signals" liefern:

Latenz

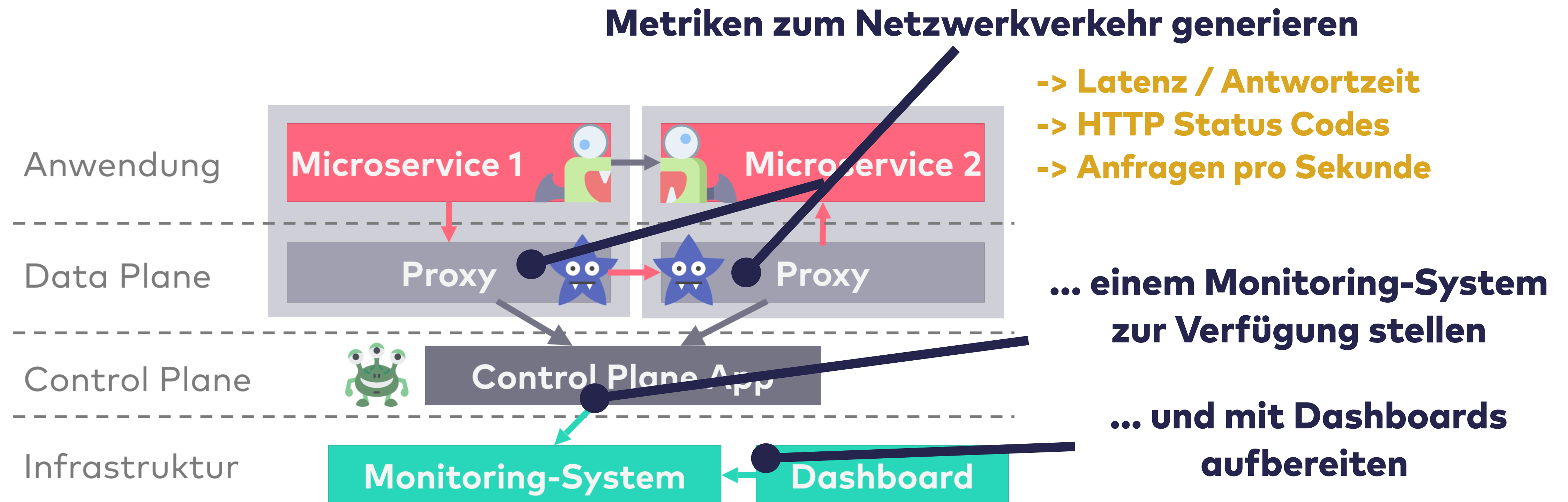
Requests/Sekunde

Status Codes (Fehler)

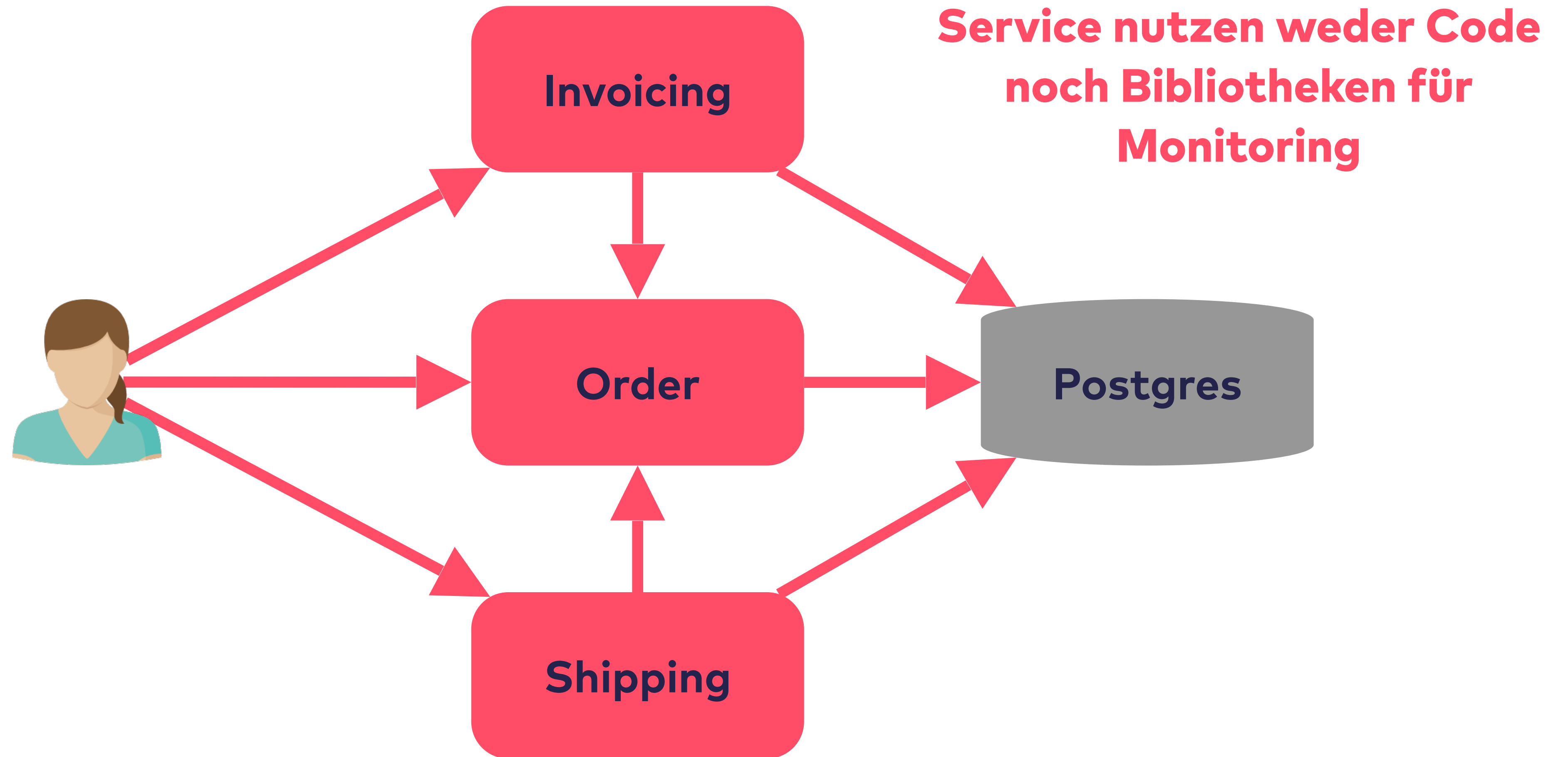
Auslastung

... es hat aber keinen Einblick in die Microservices

Monitoring mit Service Mesh



Beispiel-Anwendung



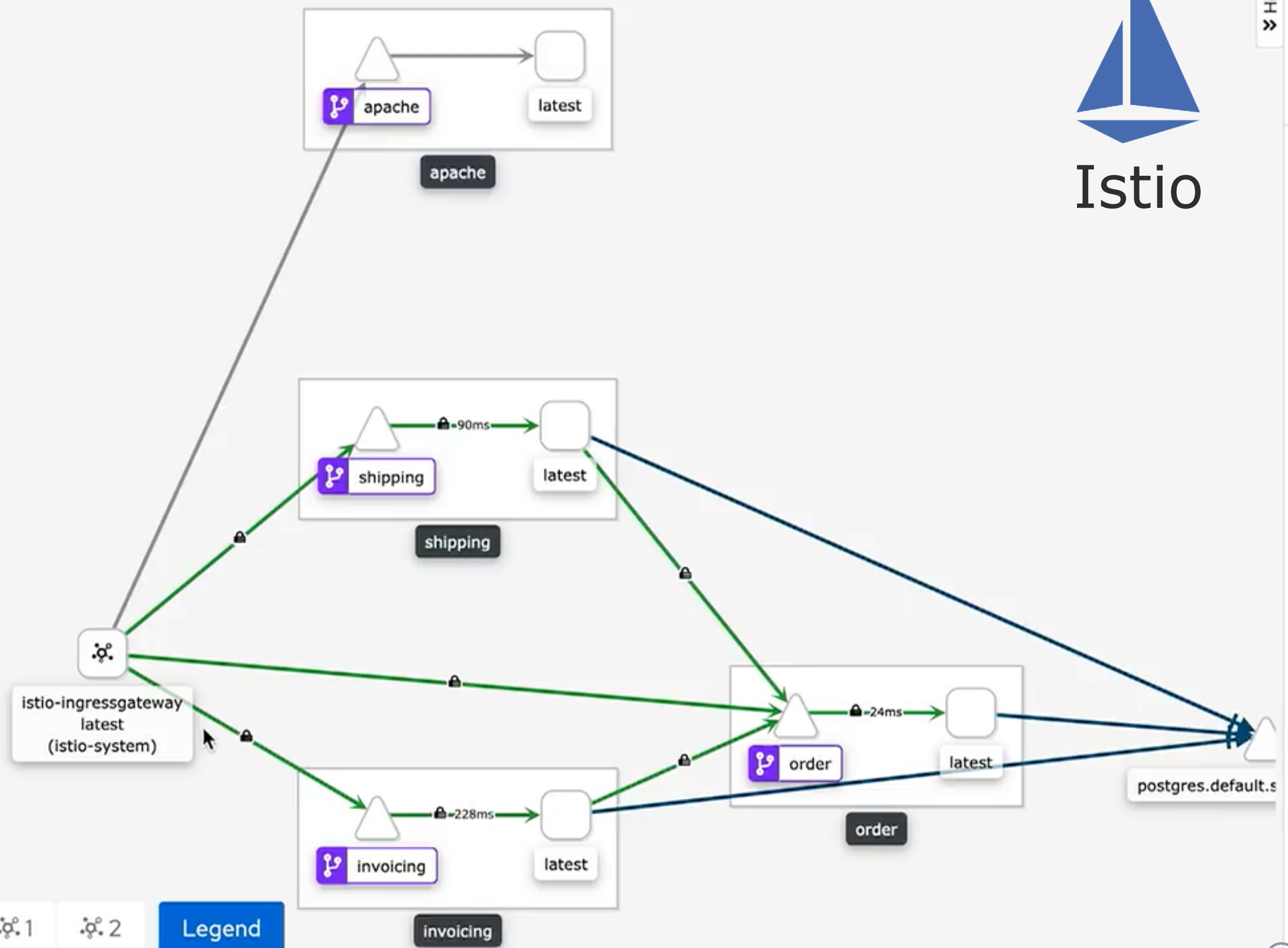
Namespace: default ▾ | Versioned app graph ▾

[Graph tour](#)

Display ▾ Find... Hide... ⓘ

🔄 Last 1m ▾ Every 15s ▾ ↻

May 27, 9:15:58 AM ... 9:16:58 AM



NS default ✓

Current Graph:

- 6 apps (6 versions)
- 6 services
- 15 edges

Incoming			Outgoing			Total		
HTTP (requests per second):								
Total			%Success			%Error		
0.34			100.00			0.00		

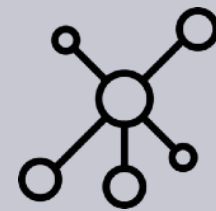
0 25 50 75 100

■ OK ■ 3xx ■ 4xx ■ 5xx

Service Mesh Features



Observability



Routing

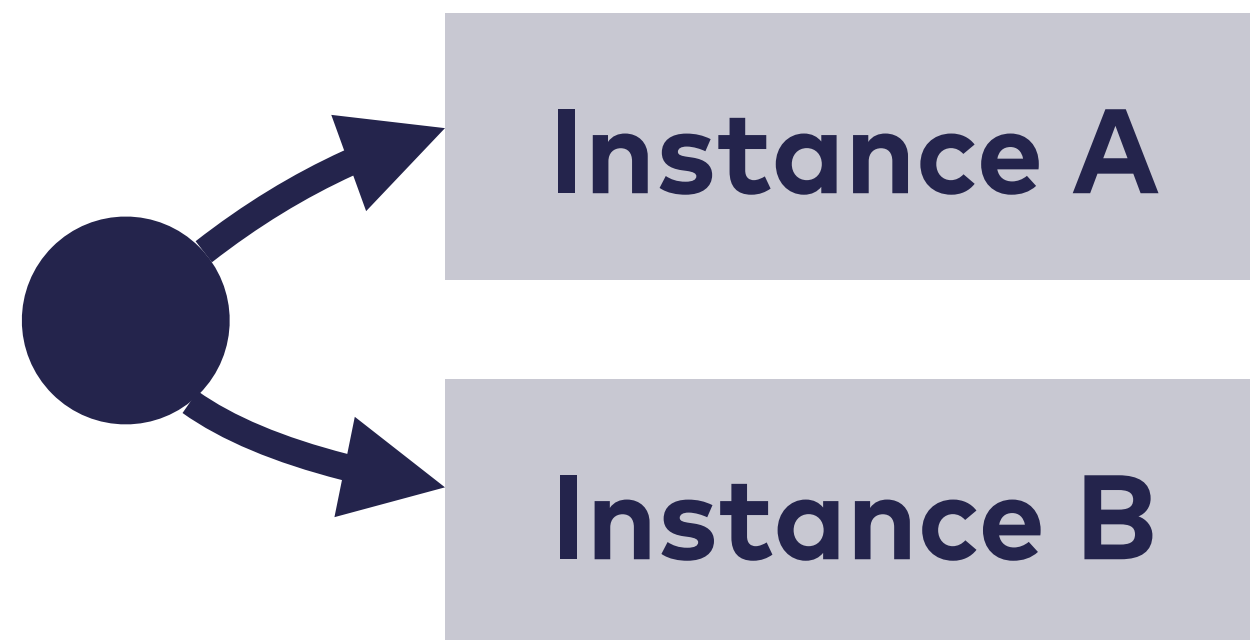


Resilience

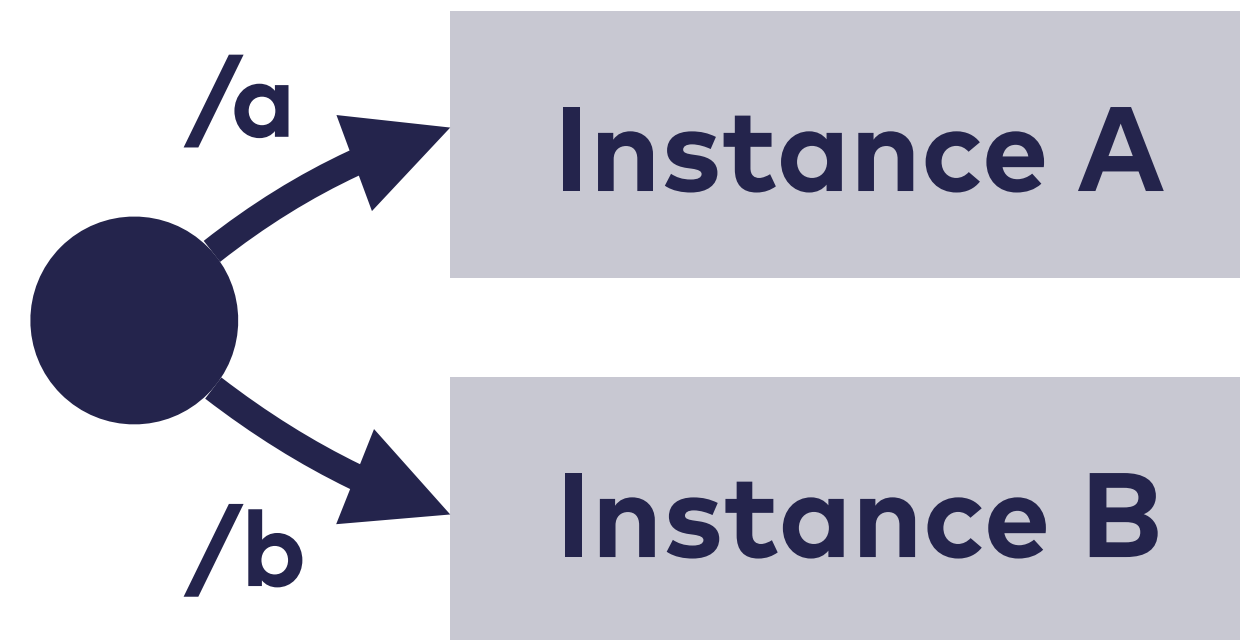


Security

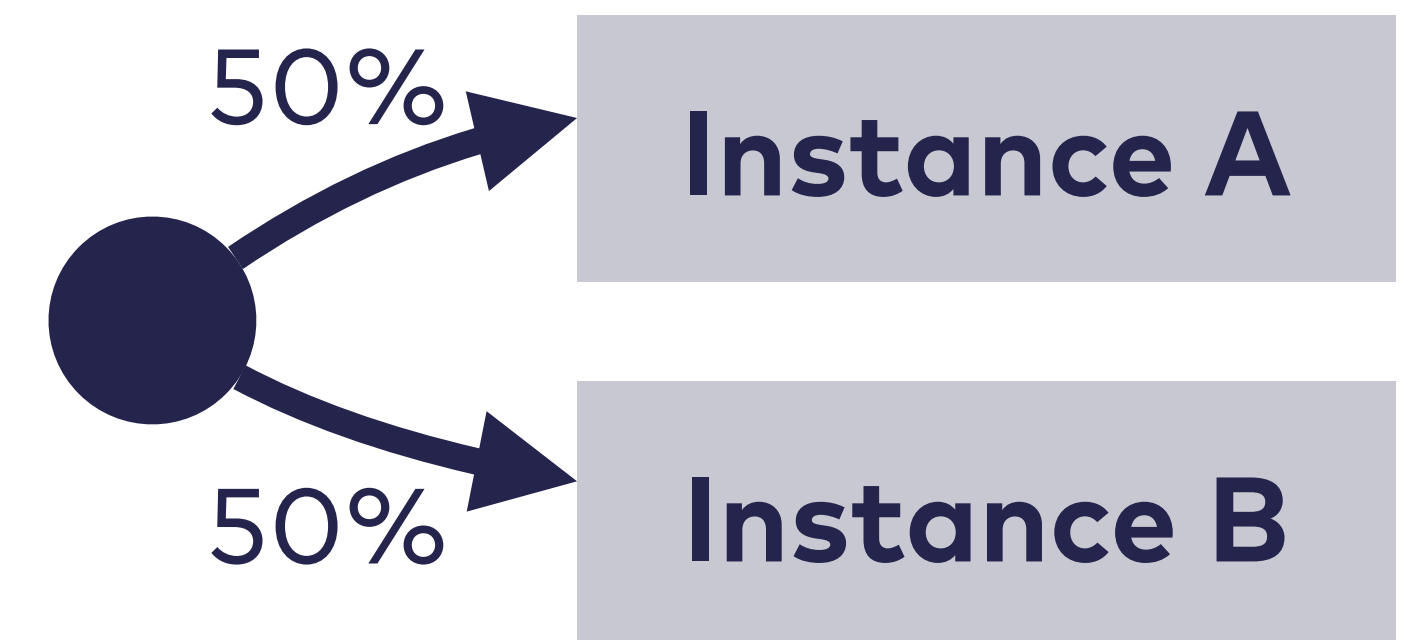
Routing



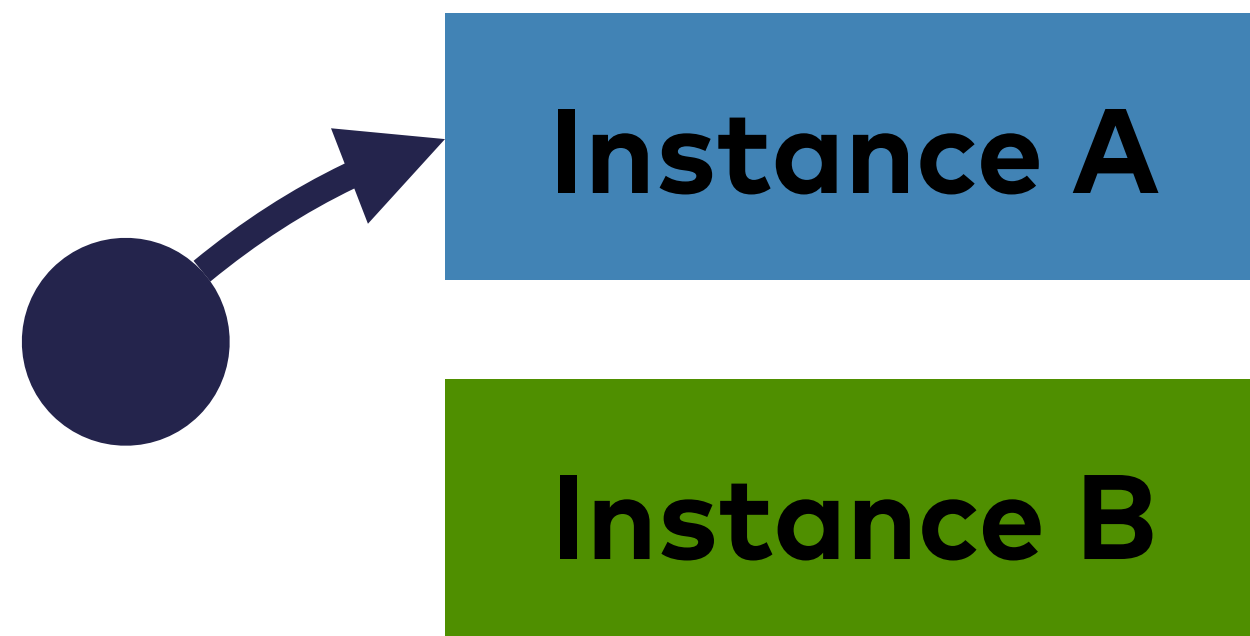
Load Balancing



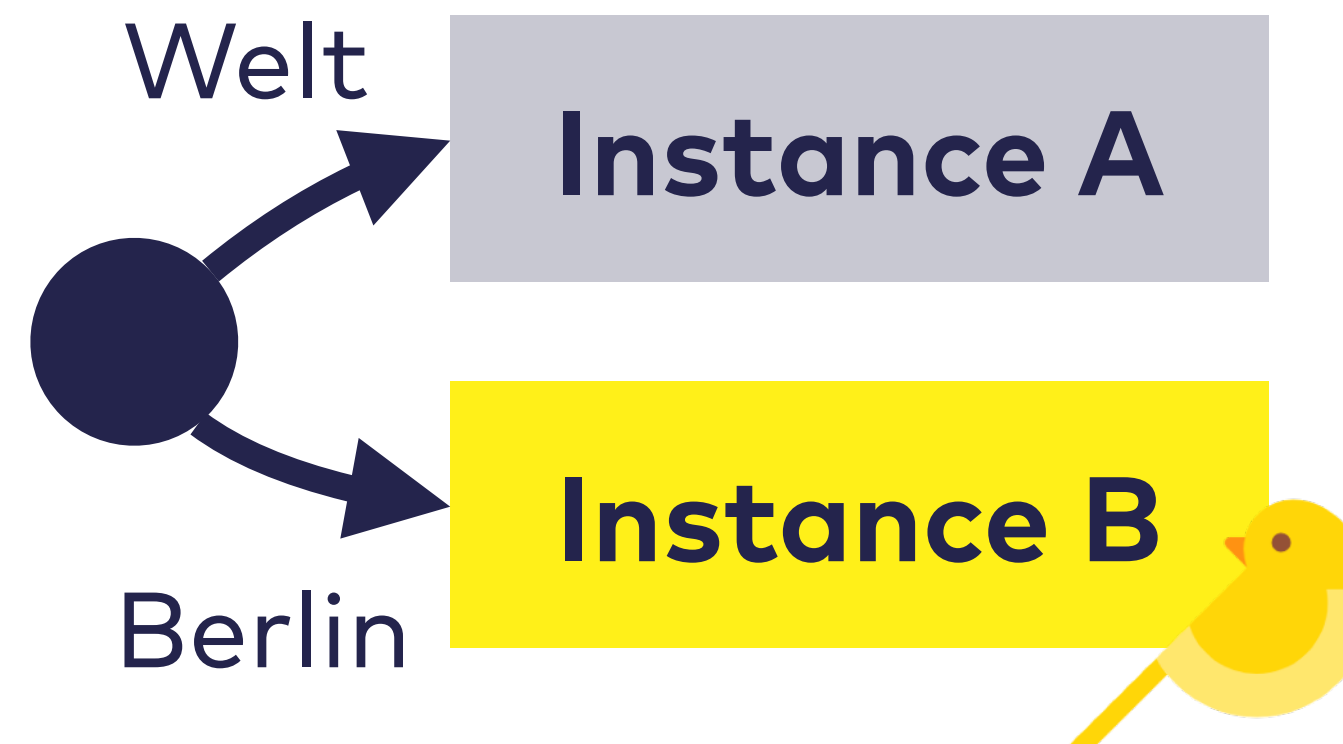
Pfad-basiertes Routing



A/B-Testing



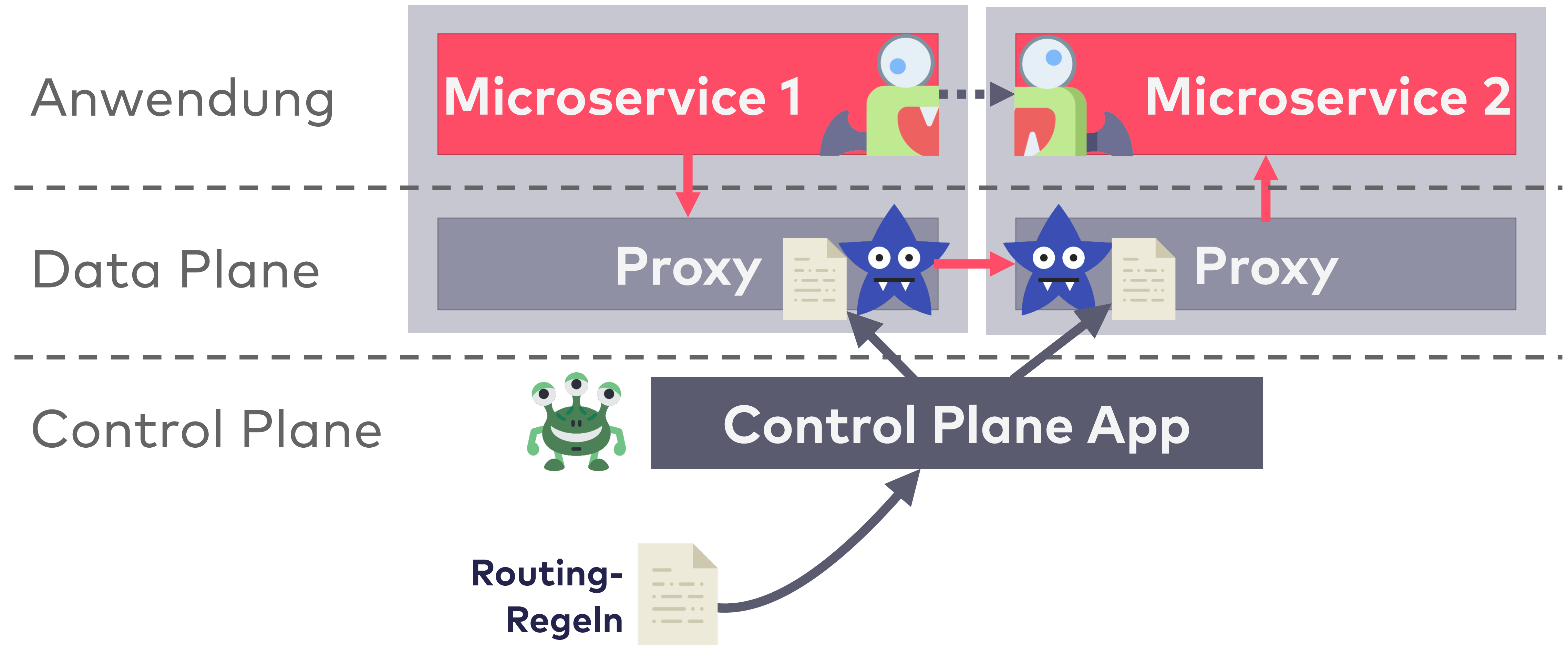
Blue/Green Deployment



Canary Releasing

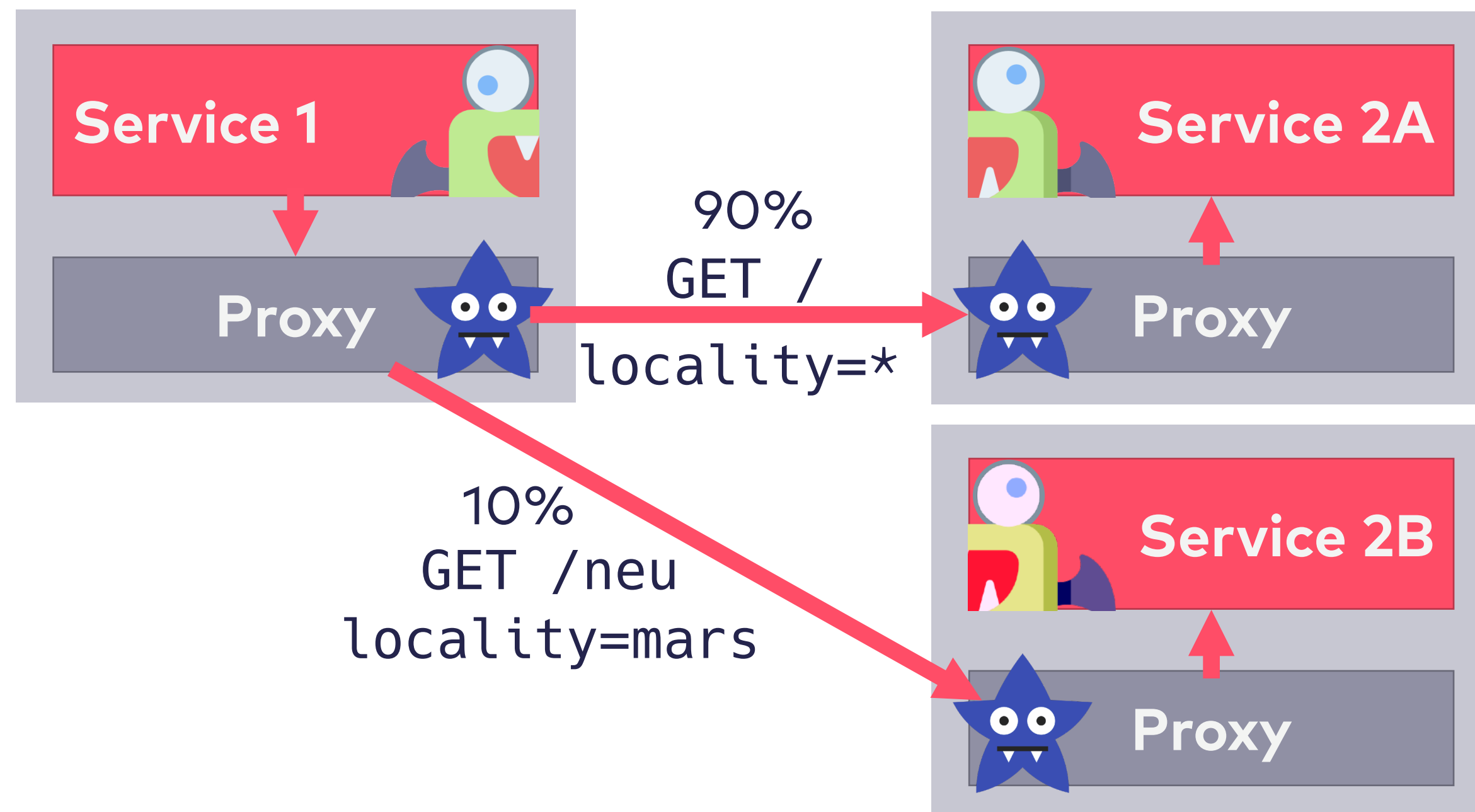
Typischerweise im
Edge Router / API-Gateway
implementiert
z.B. NGINX, Envoy,
Ambassador, Traefik

Routing mit Service Mesh

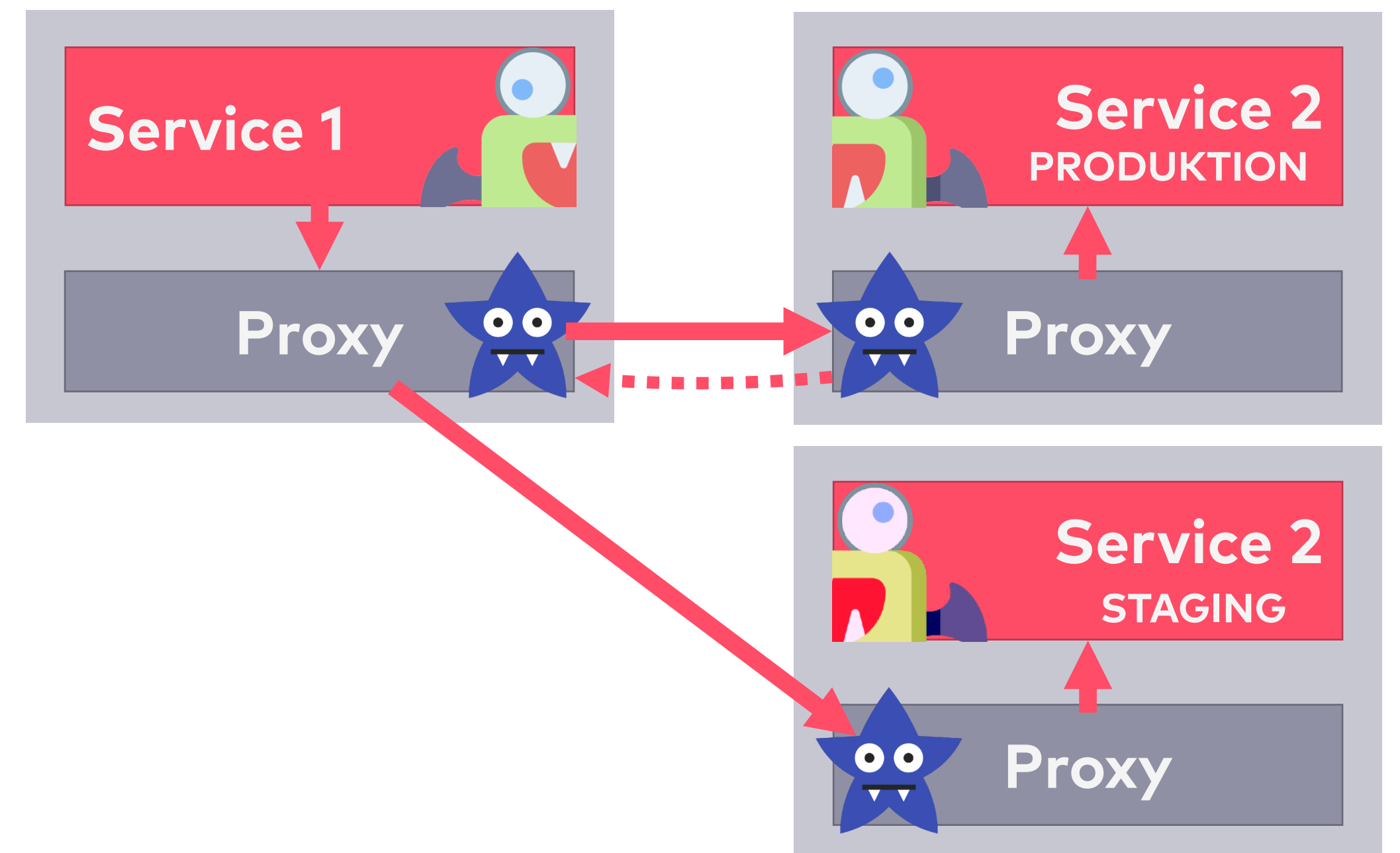


Routing mit Service Mesh

Komplexe Routing-Regeln
für A/B Testing und Canary Releasing



Traffic Mirroring



Service Mesh Features



Observability



Routing



Resilience



Security

Resilience

Was wenn ein Service nicht wie erwartet zur Verfügung steht?

500



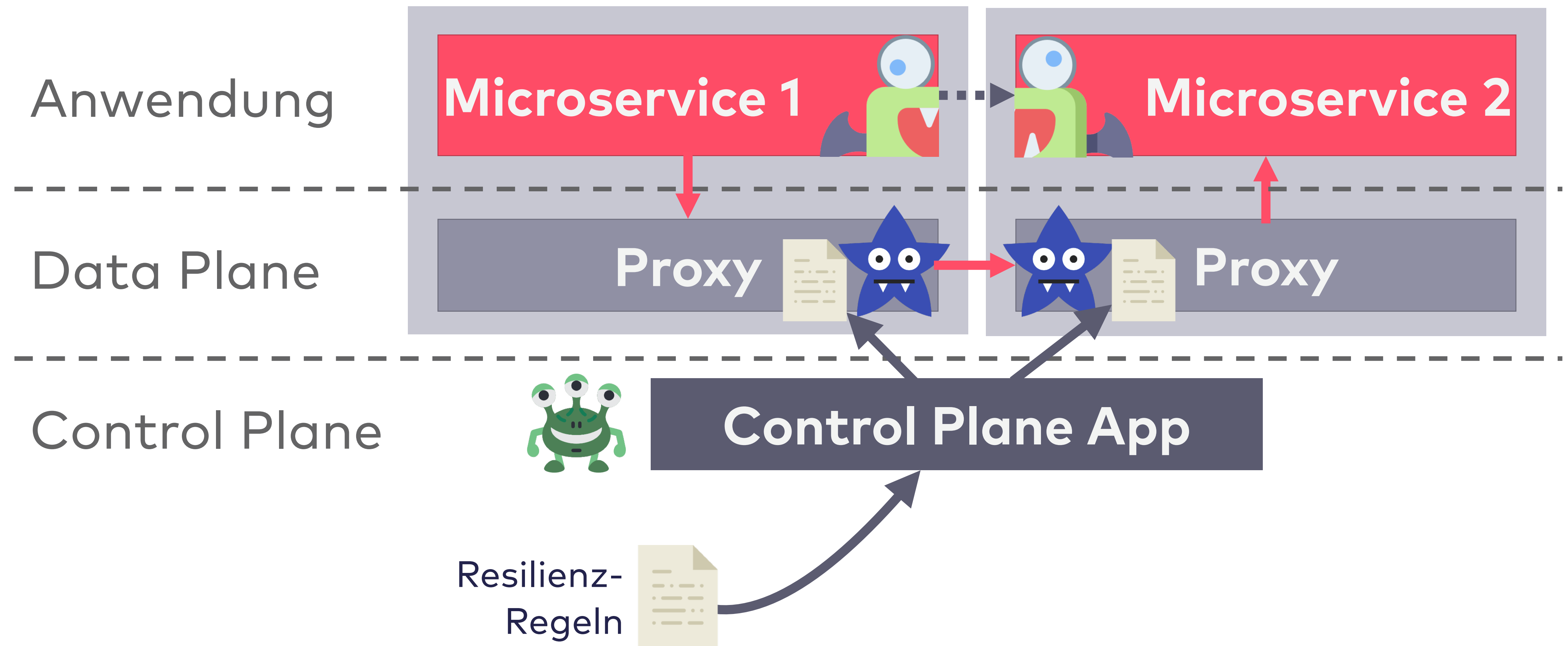
Ziel:

Gesamtsystem funktioniert weiter
... ggf. mit Einschränkungen

Maßnahmen:

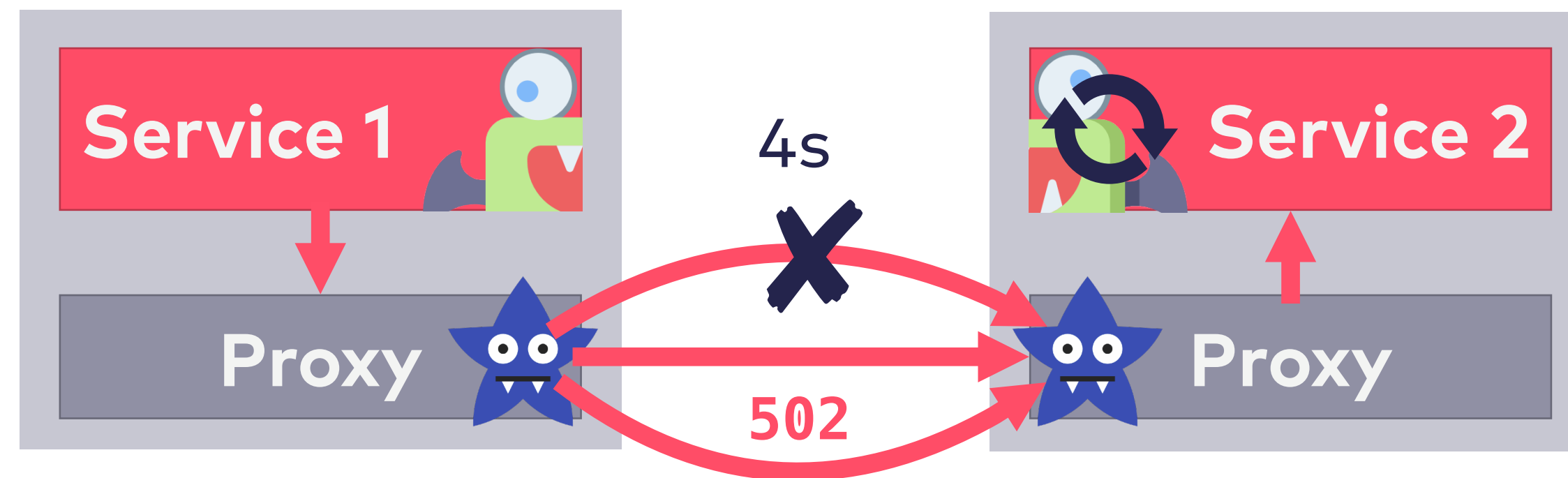
Retry, Timeout, Circuit Breaking

Resilience mit Service Mesh

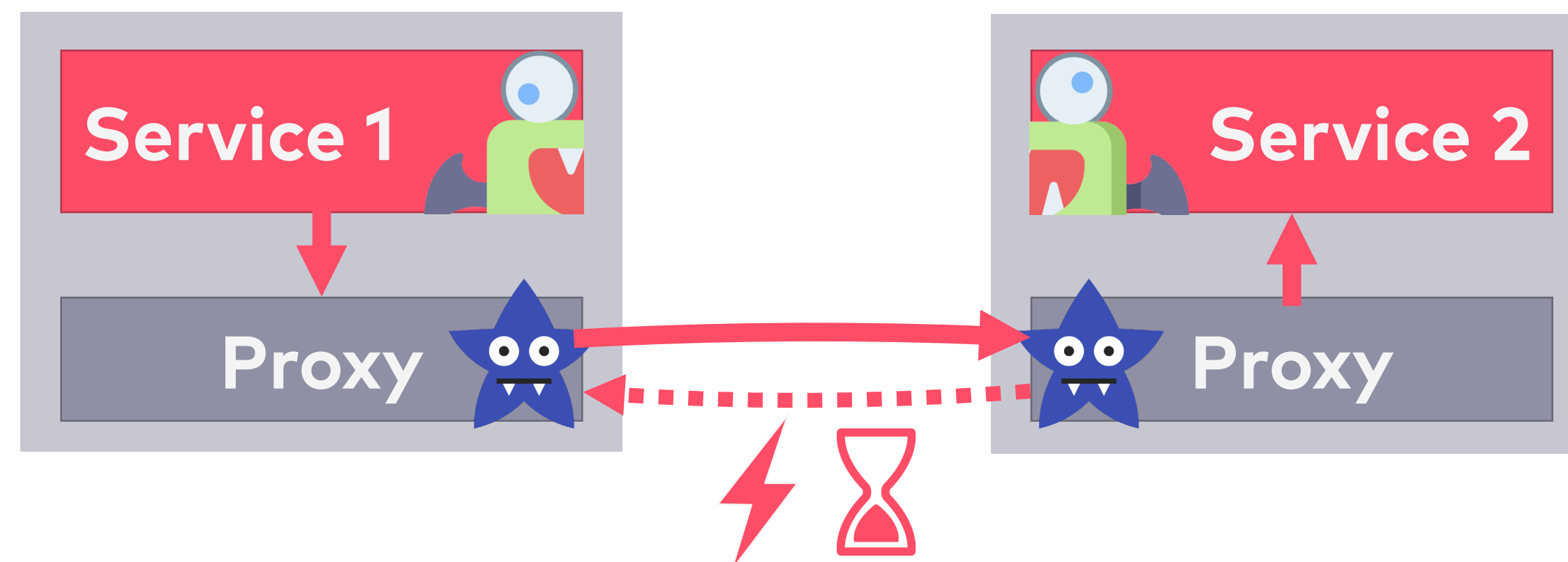


Resilience mit Service Mesh

**Timeout
Retry**



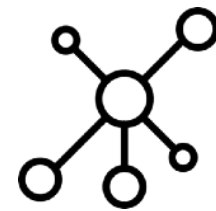
**Fault Injection
Delay Injection**



Service Mesh Features



Observability



Routing

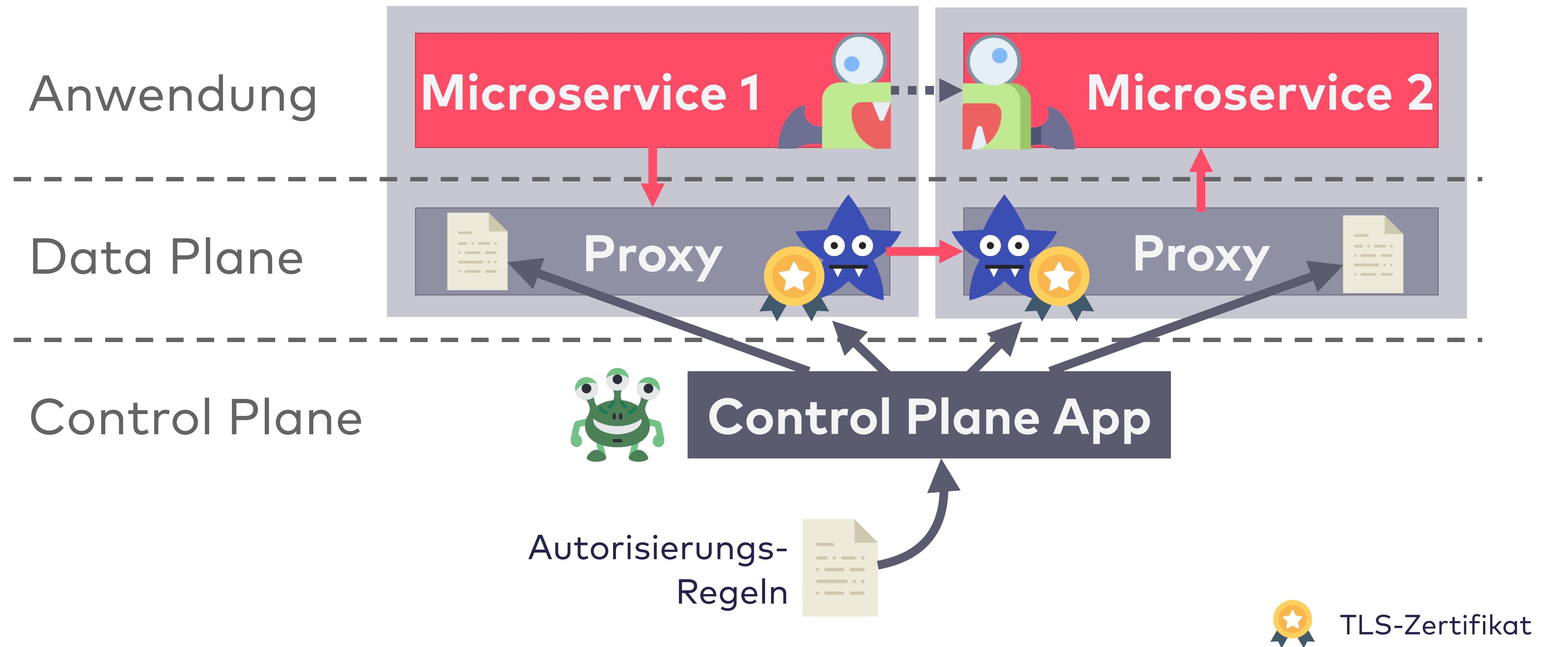


Resilience



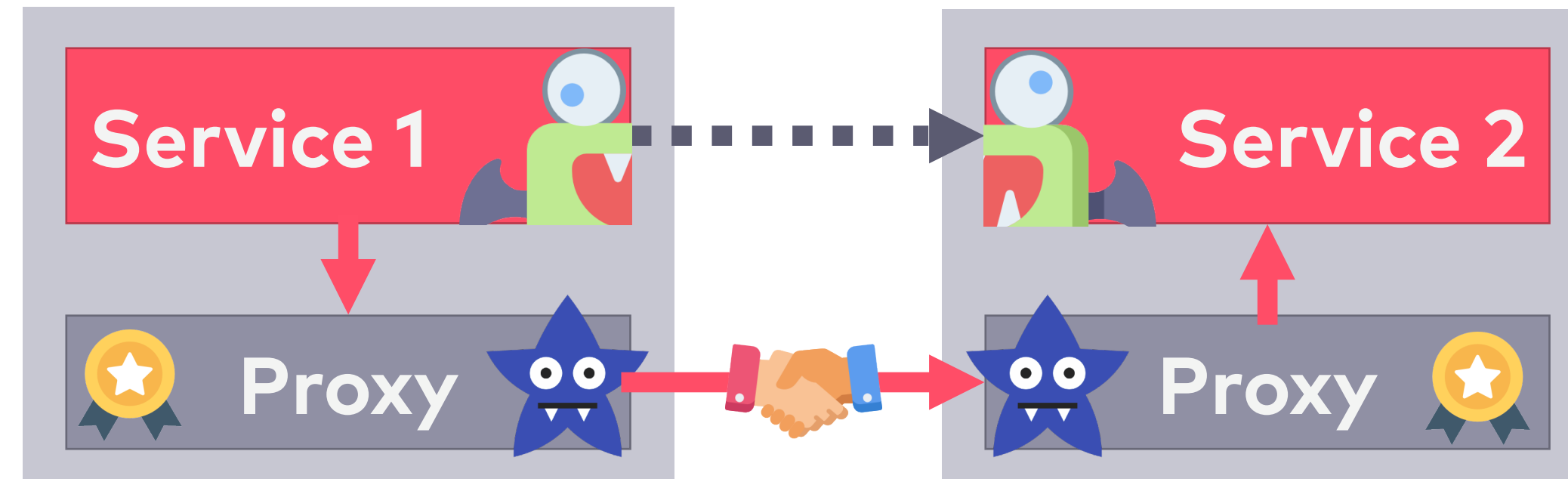
Security

Security mit Service Mesh

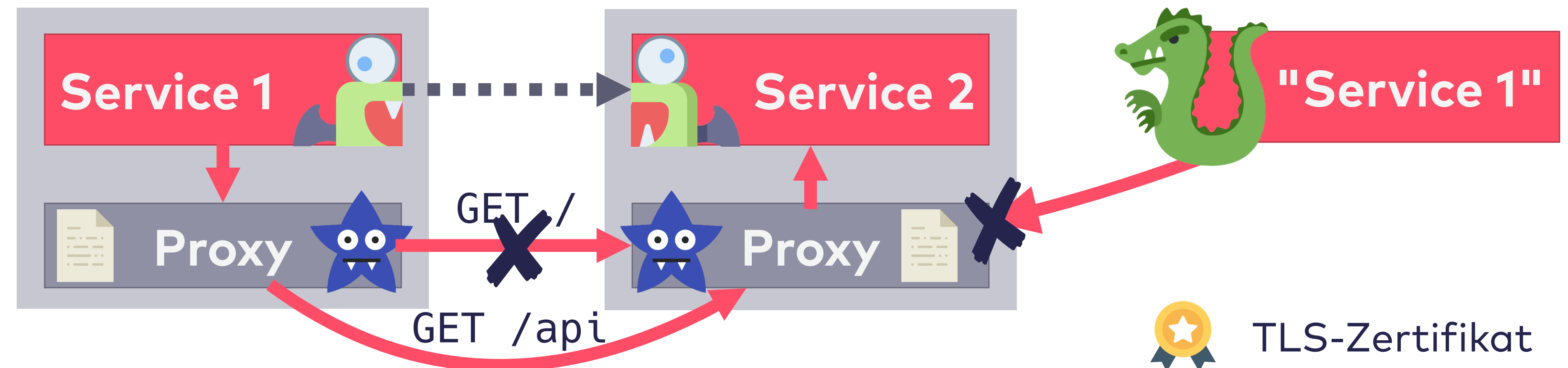


Security mit Service Mesh

**Authentifizierung
mit mTLS**



Autorisierung



Service Mesh Features

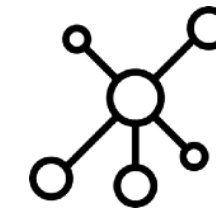
Netzwerk-Metriken und Access Logs
Tracing-Daten an Backend senden



Observability

Business-Metriken oder -Logs
Weitergabe von Tracing-Headern
Alerting

Komplexe Routing-Regeln
Canary Releasing & A/B-Testing



Routing

Automatisches Canary Releasing

Timeouts & Retrys
Circuit Breaking



Resilience

Cache oder Standardantworten in
Circuit Breaker nutzen

Authentifizierung mit mTLS
Autorisierung



Security

Service Mesh Markt

Service Mesh Implementierungen



LINKERD



AWS App Mesh

mæsh



Istio



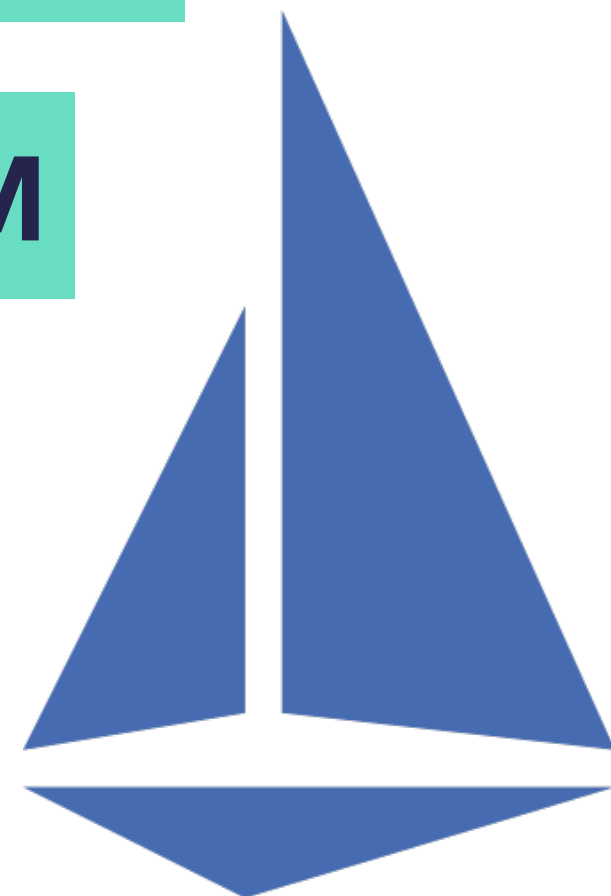


	Istio	Linkerd 2	AWS App Mesh	Consul Connect	Maesh	Kuma
Current version	1.4	2.6		1.6	1.0	0.3
License	Apache License 2.0	Apache License 2.0	Closed Source	Mozilla License	Apache License 2.0	Apache License 2.0
Developed by	Google, IBM, Lyft	Buoyant	AWS	HashiCorp	containous	Kong
Service Proxy	Envoy	linkerd-proxy	Envoy	defaults to Envoy , exchangeable	Traefik	Envoy
Ingress Controller	Envoy / Own Concept	any		any	any	any
Governance	see Istio Community	see Linkerd Governance and CNCF Charter	AWS	see Contributing to Consul	see Contributing notice	see Contributing notice
Tutorial	Istio Tasks	Linkerd Tasks	AWS App Mesh Getting Started	Hashicorp Learn platform	Maesh Example	Kuma Kubernetes Quickstart
Platform	Kubernetes, Consul & Nomad	Kubernetes	ECS, Fargate, EKS, EC2	Consul, Nomad, Kubernetes	Kubernetes	Kubernetes, VMs (Universal)
Automatic Sidecar Injection	yes	yes	yes	yes	yes (per Node)	yes
Used in production	yes	yes				
Advantages	Istio can be adapted and extended like no other Mesh. Its many features are available for Kubernetes and other	Linkerd 2 is designed to be non-invasive and is optimized for performance and usability. Therefore, it requires little	AWS App Mesh is integrated into the AWS landscape and	Consul Connect can be used in any Consul environment and therefore does not require a scheduler. The	Maesh focuses on a selection of features to achieve good	Kuma supports both Kubernetes and plain VMs and allows you to

***2017**

von **Google & IBM**

optimiert auf
Featureanzahl und
Konfigurierbarkeit



optimiert für
Kubernetes, aber
nicht exklusiv

Istio VS Linkerd 2

***2017**

von **Buoyant**

optimiert auf
Usability und
Performance



Kubernetes only

Features



Istio



LINKERD

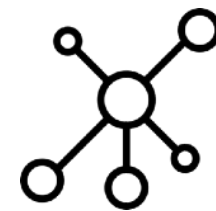


Observability

Netzwerk-Metriken und Access Logs



Tracing-Daten an Backend senden



Routing

Komplexe Routing-Regeln



Canary Releasing & A/B-Testing



Resilience

Timeouts & Retrys



Circuit Breaking



Security

Authentifizierung mit mTLS

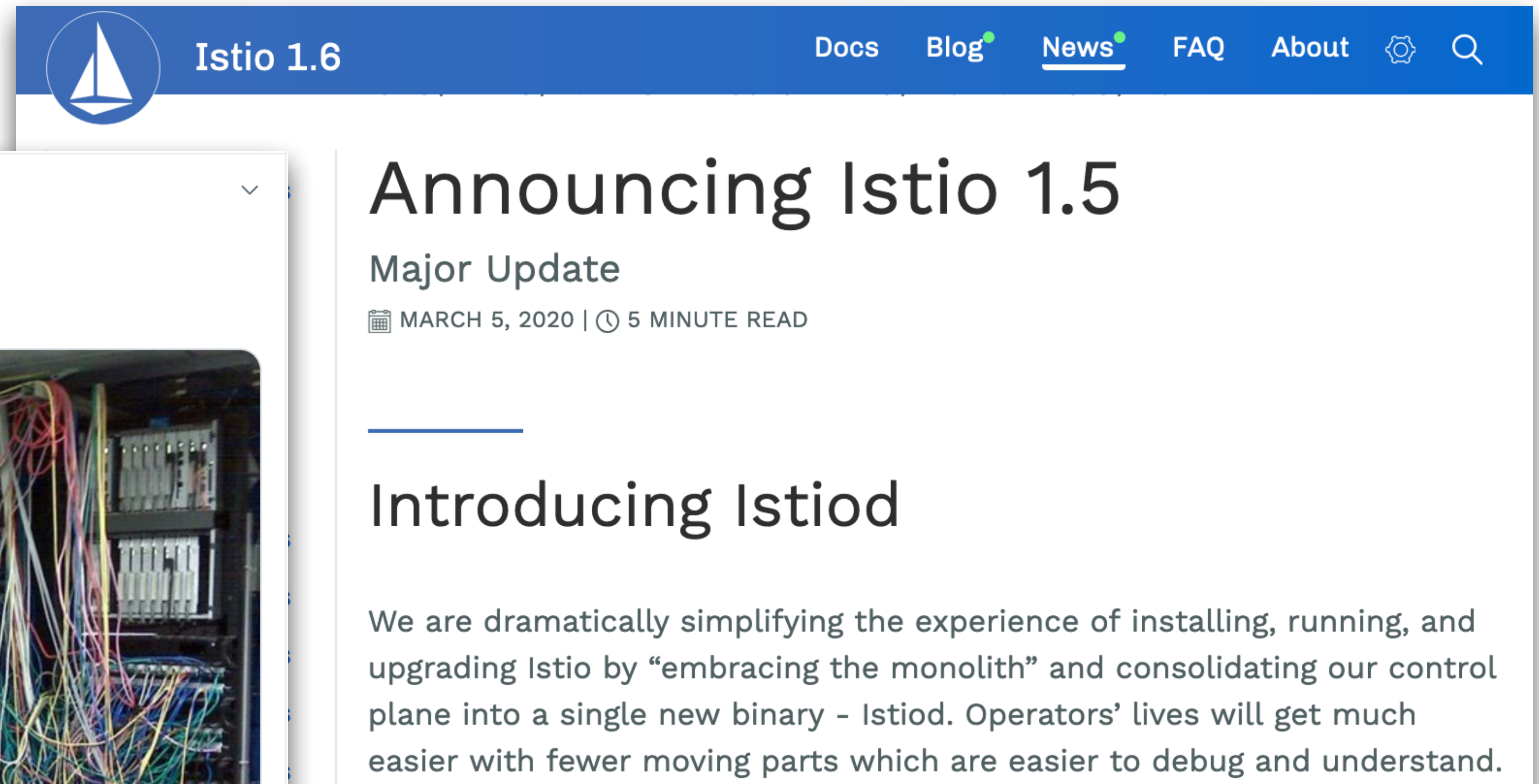


Autorisierung



Schöne Tabelle.

Usability





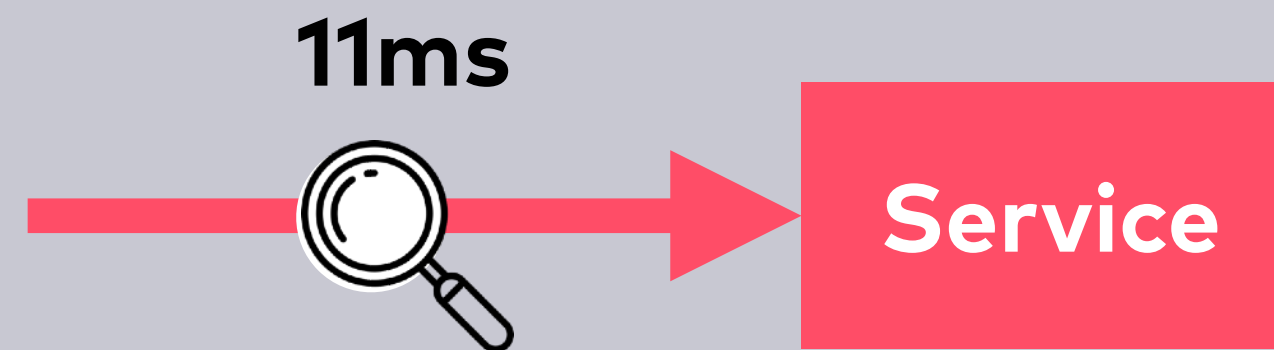
Istio VS Linkerd 2

Konfiguration im Vergleich

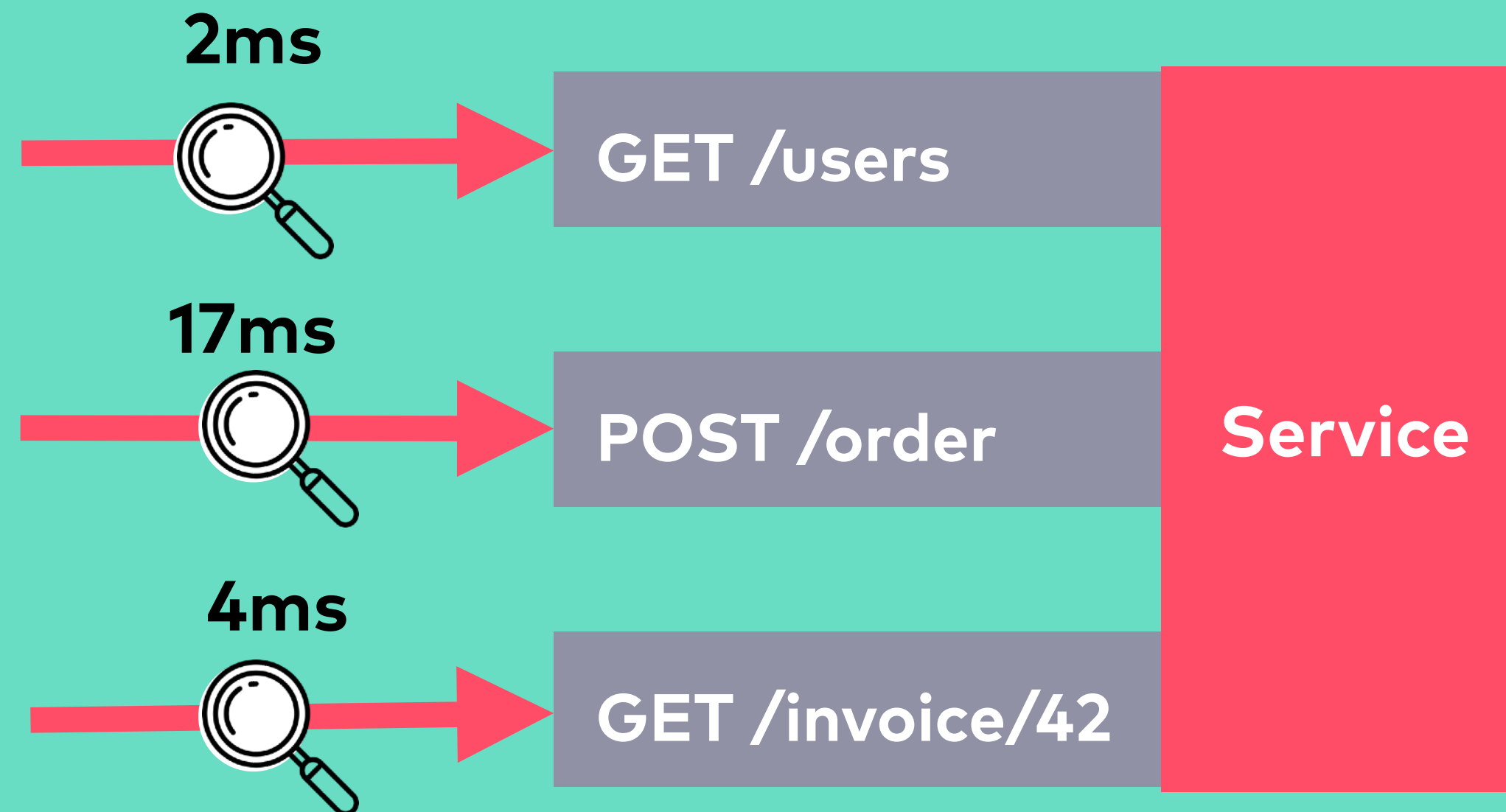


Auflösung der Metriken...

... pro Service



... pro Endpunkt



CLUSTER

- Namespaces
- Control Plane

DEFAULT

WORKLOADS

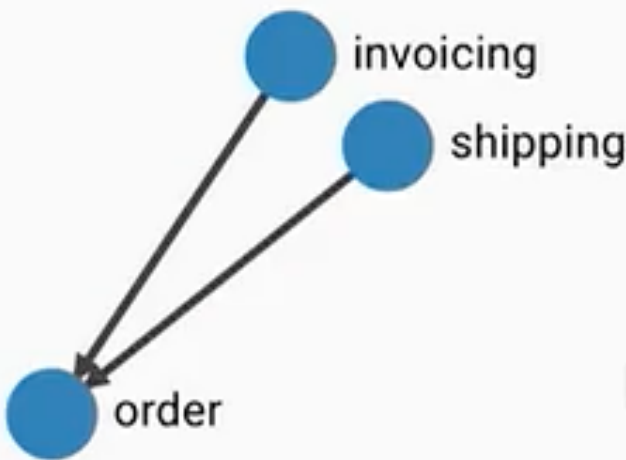
- Cron Jobs
- Daemon Sets
- Deployments
- Jobs
- Pods
- Replica Sets
- Replication Controllers
- Stateful Sets

CONFIGURATION

- Traffic Splits

TOOLS

- Tap
- Top



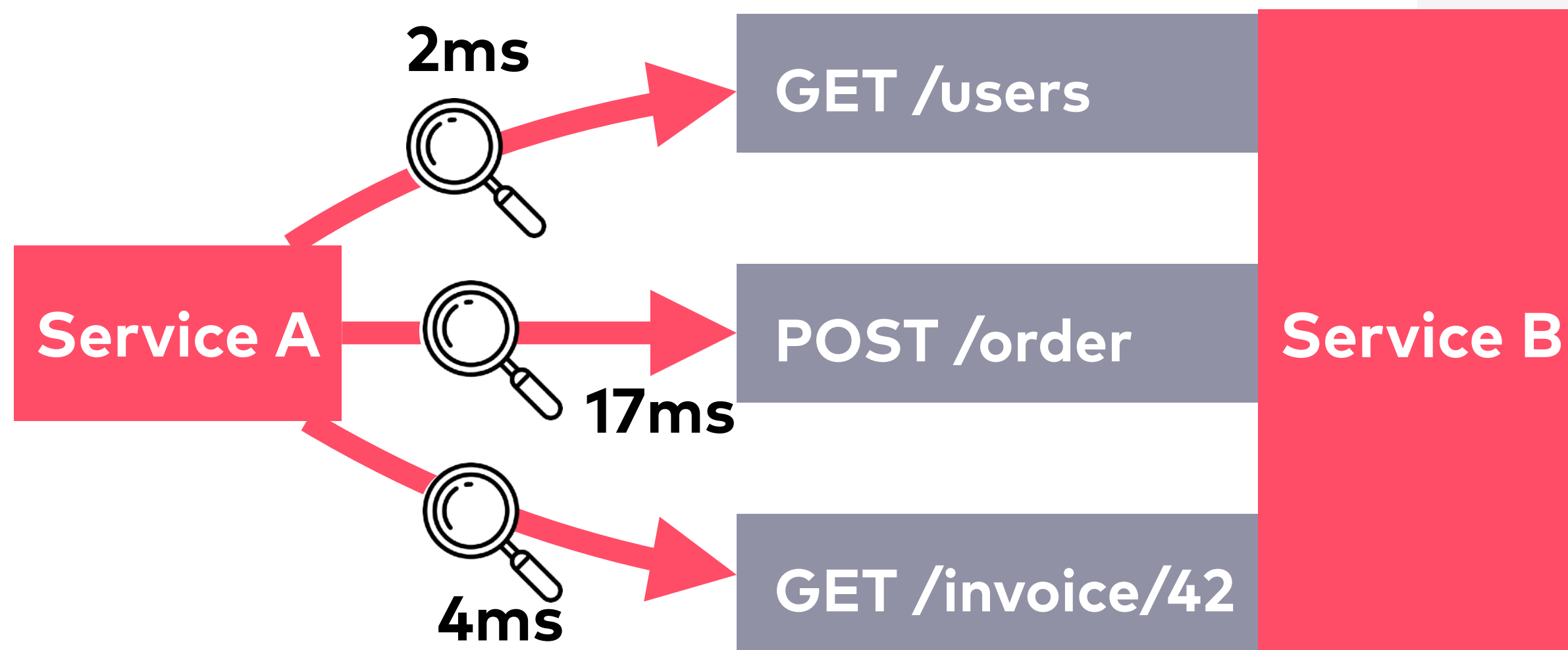
Deployments

Deployment ↑	↑ Meshed	↑ Success Rate	↑ RPS	↑ P50 Latency	↑ P95 Latency	↑ P99 Latency	Grafana
apache	1/1	100.00% ●	0.42	1 ms	1 ms	1 ms	
invoicing	1/1	100.00% ●	0.83	6 ms	16 ms	19 ms	
order	1/1	100.00% ●	1.75	17 ms	29 ms	30 ms	
postgres	1/1	---	---	---	---	---	
shipping	1/1	100.00% ●	0.83	10 ms	19 ms	20 ms	

Pods

Pod ↑	↑ Meshed	↑ Success Rate	↑ RPS	↑ P50 Latency	↑ P95 Latency	↑ P99 Latency	Grafana
--------------------	-----------------------	-----------------------------	--------------------	----------------------------	----------------------------	----------------------------	---------

Metriken pro Endpoint mit Linkerd 2

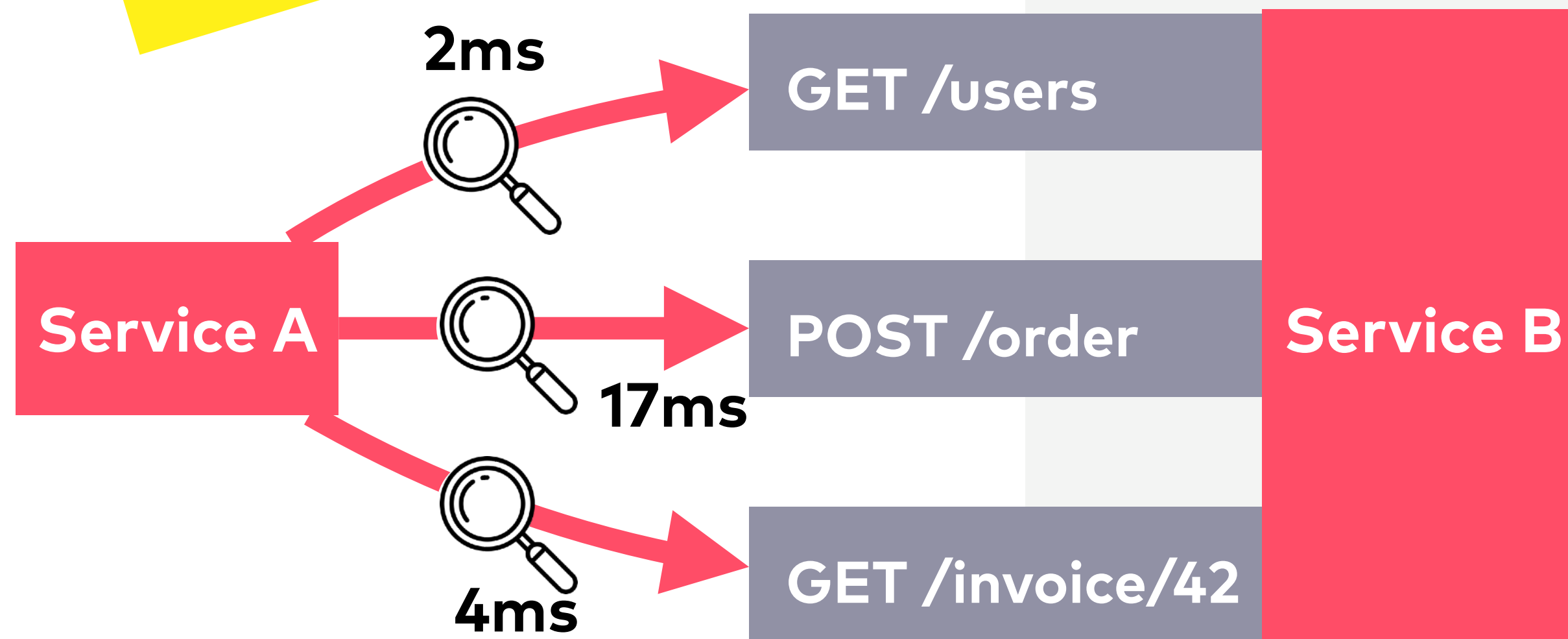


```
apiVersion: linkerd.io/v1alpha1
kind: ServiceProfile
metadata:
  name: service-b.default.svc.cluster.local
  namespace: default
spec:
  routes:
    - name: GET /users
      condition:
        method: GET
        pathRegex: /users
    - name: POST /order
      condition:
        method: POST
        pathRegex: /order
    - name: GET /invoice/{id}
      condition:
        method: GET
        pathRegex: /invoice/[^/]*
```

Metriken pro Endpunkt mit



Experimentell in Version 1.6



```
apiVersion: networking.istio.io/v1alpha3
kind: EnvoyFilter
metadata:
  name: istio-attributegen-filter
spec:
  workloadSelector:
    labels:
      app: reviews
  configPatches:
    - applyTo: HTTP_FILTER
      match:
        context: SIDECAR_INBOUND
        proxy:
          proxyVersion: '1\6.*'
        listener:
          filterChain:
            filter:
              name: "envoy.http_connection_manager"
              subFilter:
                name: "istio.stats"
      patch:
        operation: INSERT_BEFORE
        value:
          name: istio.attributegen
          typed_config:
            "@type": type.googleapis.com/udpa.type.v1.TypedStruct
            type_url: type.googleapis.com/envoy.extensions.filters.http.wasm.v3.Wasm
            value:
              config:
                configuration: |
                  {
                    "attributes": [
                      {
                        "output_attribute": "istio_operationId",
                        "match": [
                          {
                            "value": "GET /users",
                            "condition": "request.url_path == '/users' && request.method == 'GET'"
                          },
                          {
                            "value": "POST /order",
                            "condition": "request.url_path == '/order' && request.method == 'POST'"
                          },
                          {
                            "value": "GET /invoice/{id}",
                            "condition": "request.url_path.matches('^/invoice/[[:alnum:]]*$',)
                              && request.method == 'GET'"
                          }
                        ]
                      }
                    ]
                  }
              }
          vm_config:
            runtime: envoy.wasm.runtime.null
            code:
              local: { inline_string: "envoy.wasm.attributegen" }
```

Performance & Ressourcen

- **Latenz** - stark abhängig vom Traffic
 - **Istio**: zusätzlich ca. **3ms** Latenz - pro Aufruf zwischen Services!
 - **Linkerd 2**: keine aktuellen Zahlen, in früheren Versionen ähnlich
- **Ressourcen**
 - Zusätzliche Container für Control Plane & jedes Sidecar
 - → Erhöhter CPU- & Arbeitsspeicher-Verbrauch

**Aber: Abhängig von konkretem Setting
→ eigenen Benchmark machen!**

TL;DR

Service Mesh



**Löst viele wesentliche Probleme
von Microservices**

... ohne Codeänderungen!



**Eine weitere komplexe
Technologie**

**Latenz- und Ressourcen-
Overhead**

Entscheidungshilfe

Service Mesh Indikatoren

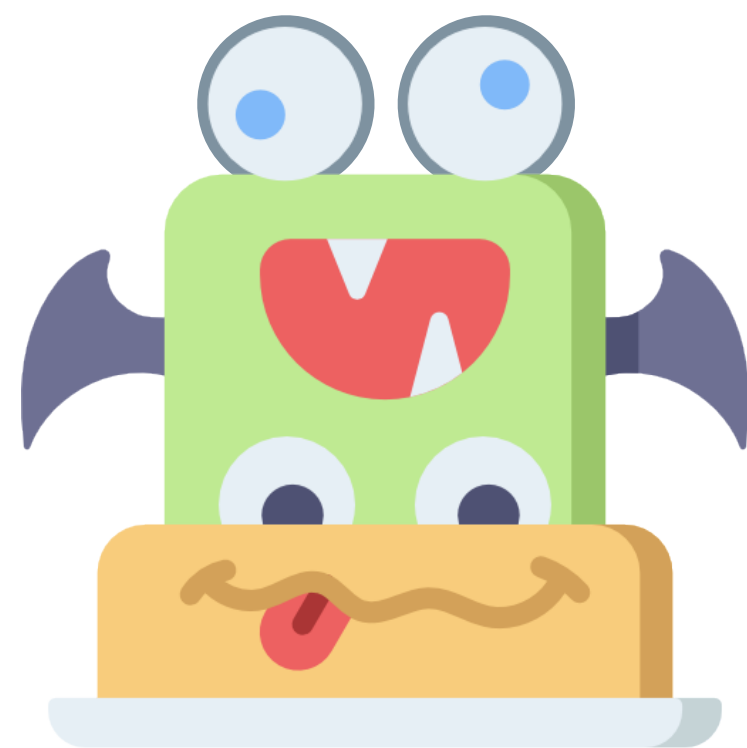
- Viele **Microservices**, viele synchrone Aufrufe
- Viele **ungelöste Probleme** beim Monitoring, Routing, Resilienz und/oder Security
- Großteil läuft in **Kubernetes**

Kriterien für die Auswahl

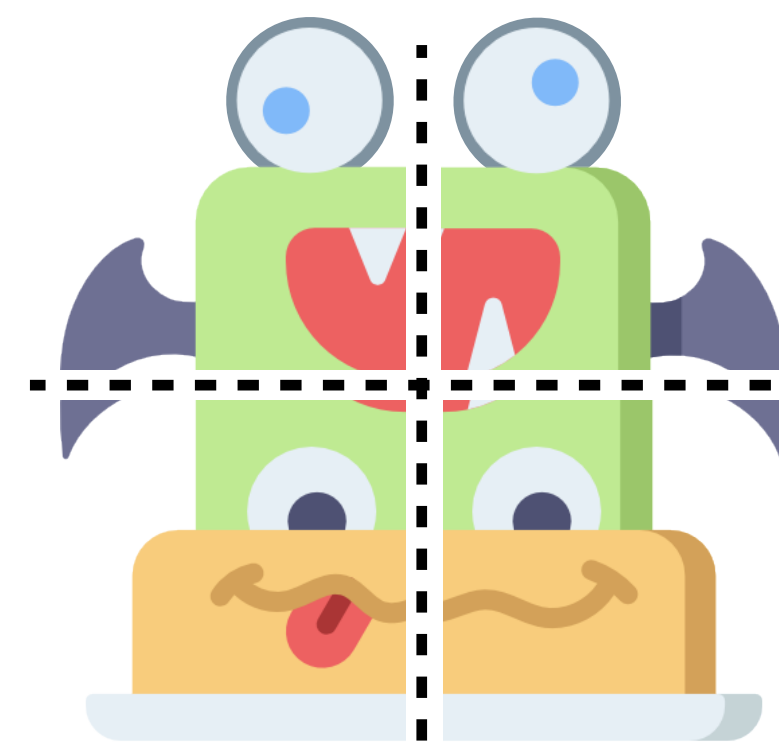
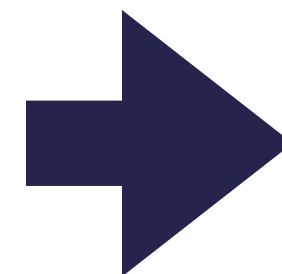
- Welche Features **fehlen** wirklich?
- Vorhandene **Infrastruktur** - Kubernetes, Consul, AWS, ...
- Zeitliche und **kognitive Kapazität** im Team
- Aktivität der **Community**

Ziel: So viel Komplexität wie nötig, aber so wenig wie möglich

Komplexität? Öhm...



Monolith



Microservices

"don't distribute your objects."

Martin Fowler

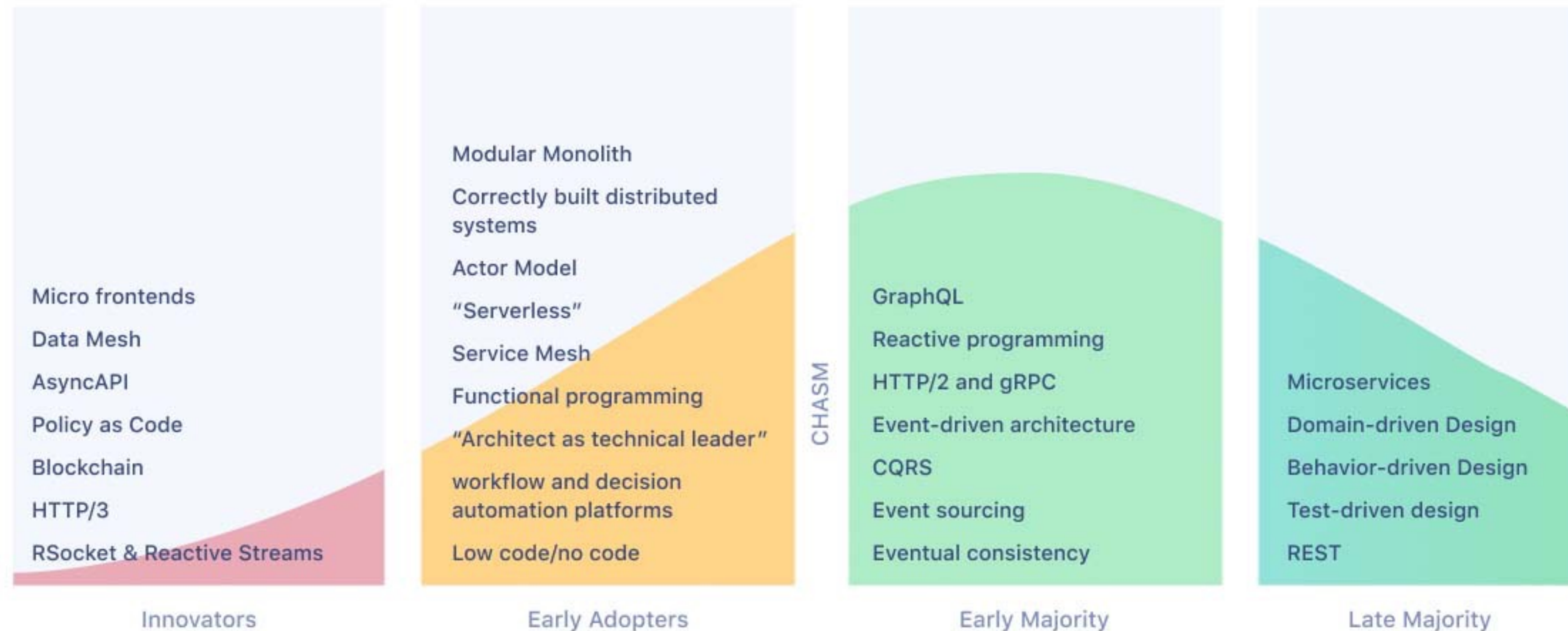


Alternativen?

Software Development Architecture and Design 2020 Q2 Graph

<http://infoq.link/architecture-trends-2020>

InfoQ



Try not to need a Service Mesh

Mehr zum Thema

- **Service Mesh Vergleich auf servicemesh.es**
<https://servicemesh.es/>
- **Artikel: Glücklich ohne Service Mesh**
<https://www.innoq.com/de/blog/gluecklich-ohne-service-mesh/>
- **Artikel: Brauchen asynchrone Microservices und Self-Contained-Systems ein Service Mesh?**
<https://www.innoq.com/de/articles/2020/02/service-mesh-asynchrone-microservices-scs/>
- **Artikel: Alle 11 Minuten verliebt sich ein Microservice in Linkerd**
<https://www.innoq.com/en/articles/2020/01/linkerd2/>
- **Podcast zu Service Meshes**
<https://www.innoq.com/de/podcast/059-service-meshes-1/>
- **Beispiel-Anwendung auf GitHub**
<https://github.com/ewolff/microservice-istio>
- **Tutorial von Linkerd**
<https://linkerd.io/2/tasks/>
- **Tutorial von Istio**
<https://istio.io/docs/setup/getting-started/>

Danke! Fragen?

Hanna Prinz
hanna.prinz@innoq.com
@HannaPrinz



INNOQ
www.innoq.com

Icons made by [srip](#),
[Smashicons](#), [Nikita Golubev](#), [Freepik](#), [surang](#)
and [Darius Dan](#) from
www.flaticon.com and
licensed by [CC 3.0 BY](#)

Service Mesh Primer - 2nd Edition
Kostenlos auf leanpub.com/service-mesh-primer

innoQ Deutschland GmbH

Krischerstr. 100
40789 Monheim am Rhein
Germany
+49 2173 3366-0

Ohlauer Str. 43
10999 Berlin
Germany
+49 2173 3366-0

Ludwigstr. 180E
63067 Offenbach
Germany
+49 2173 3366-0

Kreuzstr. 16
80331 München
Germany
+49 2173 3366-0

Hermannstrasse 13
20095 Hamburg
Germany
+49 2173 3366-0

innoQ Schweiz GmbH

Gewerbestr. 11
CH-6330 Cham
Switzerland
+41 41 743 0116