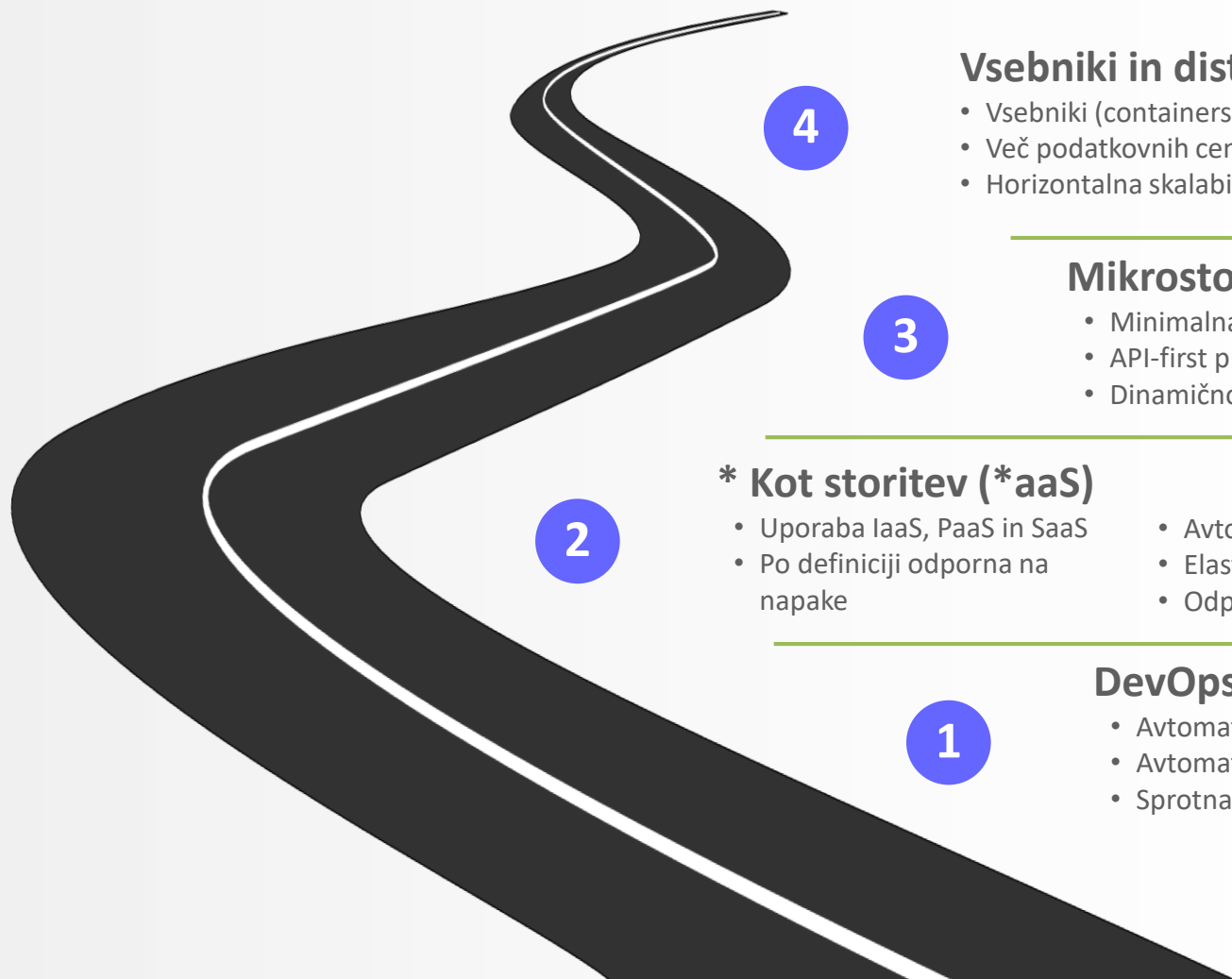


„Cloud-native“ arhitektura koncepti

Prof. dr. Matjaž B. Jurič



Arhitektura „Cloud-native“ določa
način gradnje aplikacij in
ne **lokacije** njihovega izvajanja



Vsebniki in distribuirana arhitektura

- Vsebniki (containers)
- Več podatkovnih centrov
- Horizontalna skalabilnost
- Dinamično odkrivanje
- Globalno porazdeljevanje
- Replikacija

Mikrostoritve in API-ji

- Minimalna funkcionalnost
- API-first pristop
- Dinamično odkrivanje
- Prekinjevalci toka
- Gatewayi
- Šibka sklopljenost

* Kot storitev (*aaS)

- Uporaba IaaS, PaaS in SaaS
- Po definiciji odporna na napake
- Avtomatska skalabilnost
- Elastičnost
- Odpornost (Resilience)

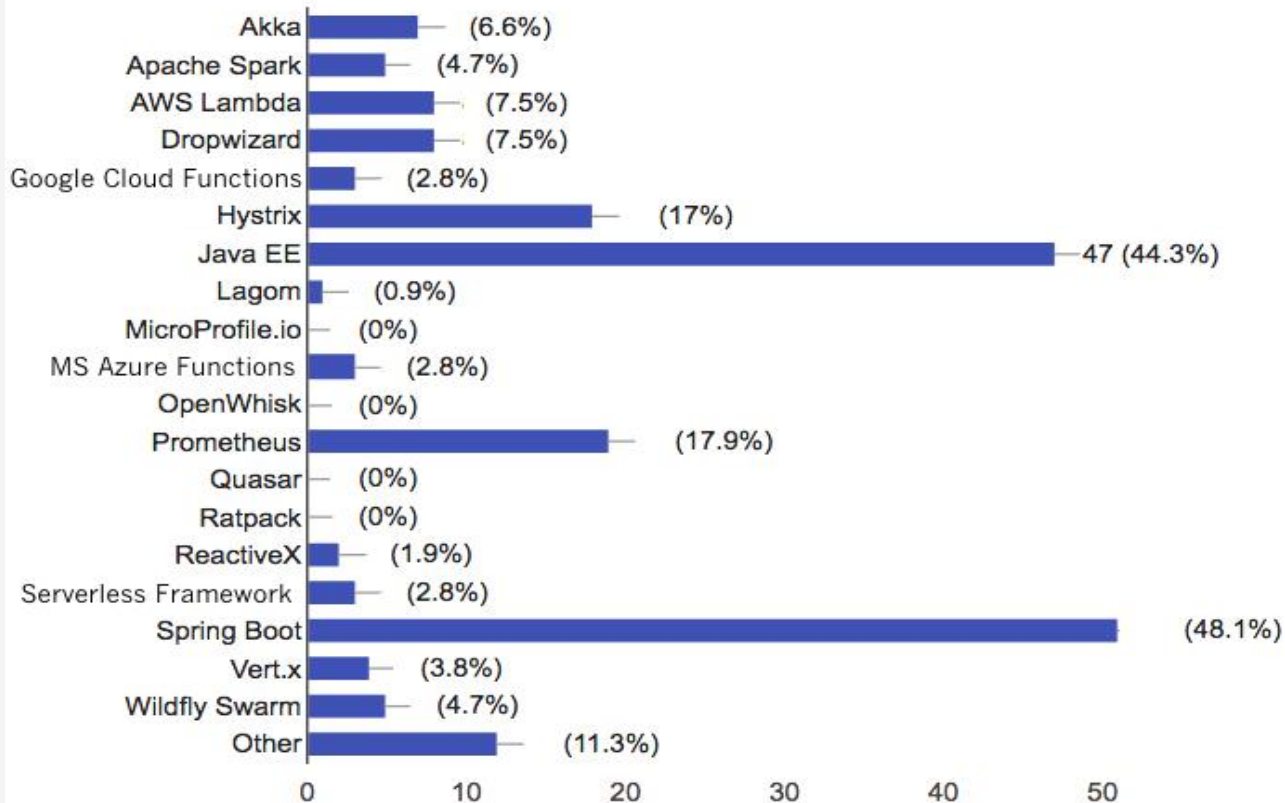
DevOps

- Avtomatski provisioning
- Avtomatska konfiguracija
- Sprotna integracija (CI)
- „Continuous Delivery“
- Avtomatizirano testiranje
- Agilni razvoj

Ključni elementi Cloud-native

- Razvojna arhitektura
 - Mikrostoritve
 - Napredni gradniki okolja mikrostoritev
 - Logiranje, konfiguracija, service discovery, circuit-breakers, itd.
 - Pakiranje mikrostoritev v vsebnike
 - Uporaba API-jev
 - Principi in vzorci
- Integracijska platforma za integracijo z zalednimi sistemi

I use the following frameworks/tools to build microservices



Ključni elementi Cloud-native

- Platforma
 - Okolje za izvajanje mikrostoritev (izbran „stack“)
 - IAM (Identity and Access Management)
 - OAuth2, OpenID Connect
 - API Gateway in sistem za upravljanje API-jev
 - Konfiguracijski strežnik
 - Service discovery
 - Sistem za distribuirano upravljanje logov
 - Nadzor izvajanja mikrostoritev
 - Zbiranje metrik

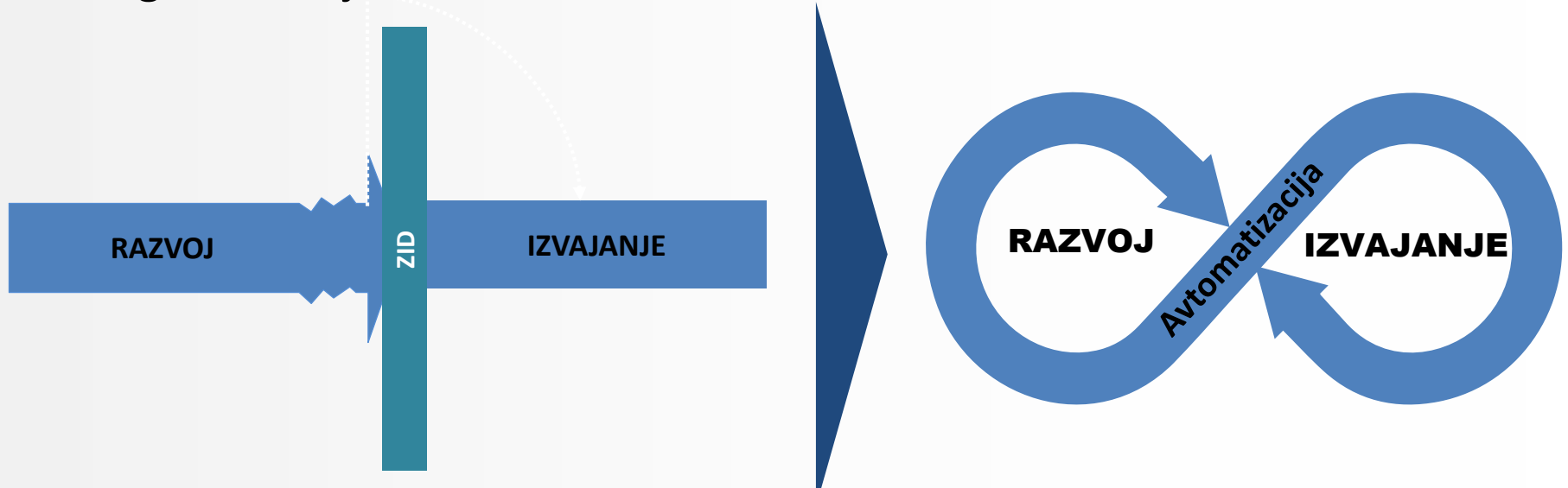
Ključni elementi Cloud-native

- Infrastruktura
 - Vse kot storitev (*aaS, XaaS)
 - Uporaba IaaS, PaaS in SaaS
 - Softversko definirana infrastruktura
 - Po definiciji odporna na napake
 - Avtomatska skalabilnost
 - Elastičnost
 - Odpornost (Resilience)
 - Vsebniki – Docker
 - Okolja za orkestracijo vsebnikov
 - Kubernetes, Mesos, OpenShift, itd.

Ključni elementi Cloud-native

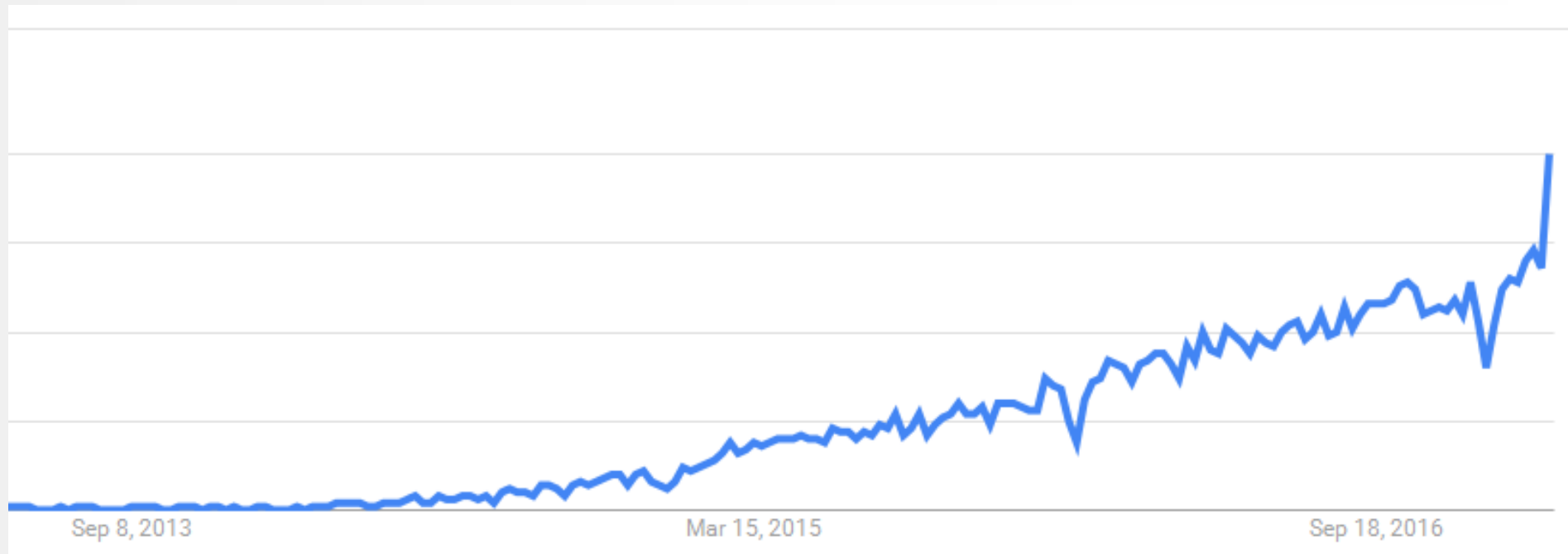
■ DevOps

- Avtomatizacija operacij
- Continuous Development, Integration, Delivery, Deployment, Operations
- DevOps kultura
- Agilni razvoj



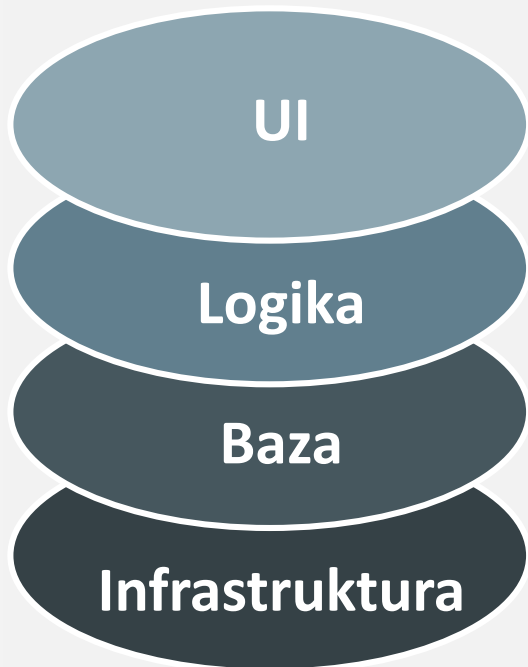
Arhitekturni pristop, ki poudarja razgradnjo aplikacij na šibko sklopljene specializirane storitve, katere lahko obvladljivo razvijajo in posodabljajo multifunkcijske razvojne skupine, za zagotavljanje kvalitete in kratkih razvojnih ciklov razvoja in vzdrževanja kompleksnih sistemov v skladu z zahtevami današnjega digitalnega poslovanja.

Mikrostoritve



Mikrostoritve

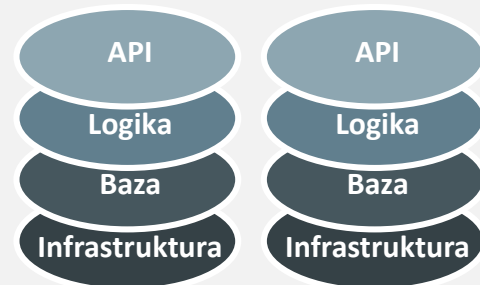
Klasično



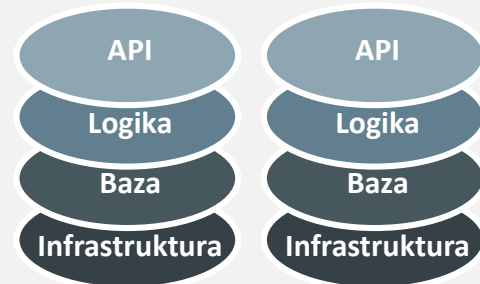
Ena aplikacija



Mikrostoritve



Mikrostoritev 1 Mikrostoritev 2



Mikrostoritev 3 Mikrostoritev 4

Več mikrostoritev

Principi

- Ena odgovornost
 - Mikrostoritev dela eno nalogo
 - Ampak tisto dobro.
- Eksplicitni vmesniki
- Integracija na šibko sklopljen način
- Načrtovana odporno proti odpovedim
 - „Fault tolerance“ je zahteva

WRITE PROGRAMS THAT DO
ONE THING & DO IT WELL.
WRITE PROGRAMS TO
WORK TOGETHER.

- DOUG MCILROY

Vsebniki

- Koristni za mikrostoritve
 - Niso obvezna zahteva
- Vsebniki so lahki
- Enostavni za zagon in ustavitvev
- Prenosljivi

- Ena instanca storitve na vsebnik
- En port na vsebnik
- **So vsebniki podobni VM slikam? NE**



Primerjava

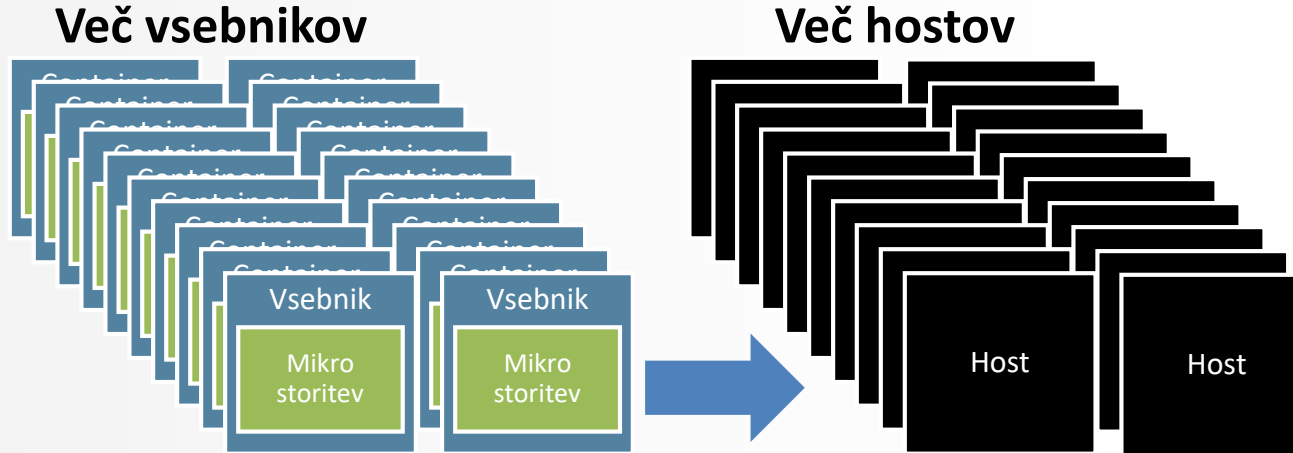
	<i>Klasični VM</i>	<i>Docker</i>	Δ
<i>Zagon</i>	<i>50 s</i>	<i>100 ms</i>	<i>500x</i>
<i>Ustavitev</i>	<i>5 s</i>	<i>50 ms</i>	<i>100x</i>

Artefakti = vsebniki

- Namesto EAR, WAR, JAR imamo (nespremenljive) vsebnike
- Brez instalacij JVM, aplikacijskih strežnikov, konfiguracij, ipd.
- Enkrat zgradimo vsebnik in ga namestimo kjerkoli.
- Vsebniki so brez stanja in brez konfiguracije
- Vsebniki ne predpostavljajo dostopa do persistentnega lokalnega datotečnega sistema

Orkestracija vsebnikov

- Samodejno skaliranje in elastičnost



Docker Swarm



RANCHER



kubernetes

Okolje za izvajanje vsebnikov

Container
Orchestration



kubernetes



CoreOS



Container
Engines



docker



Mini OSs



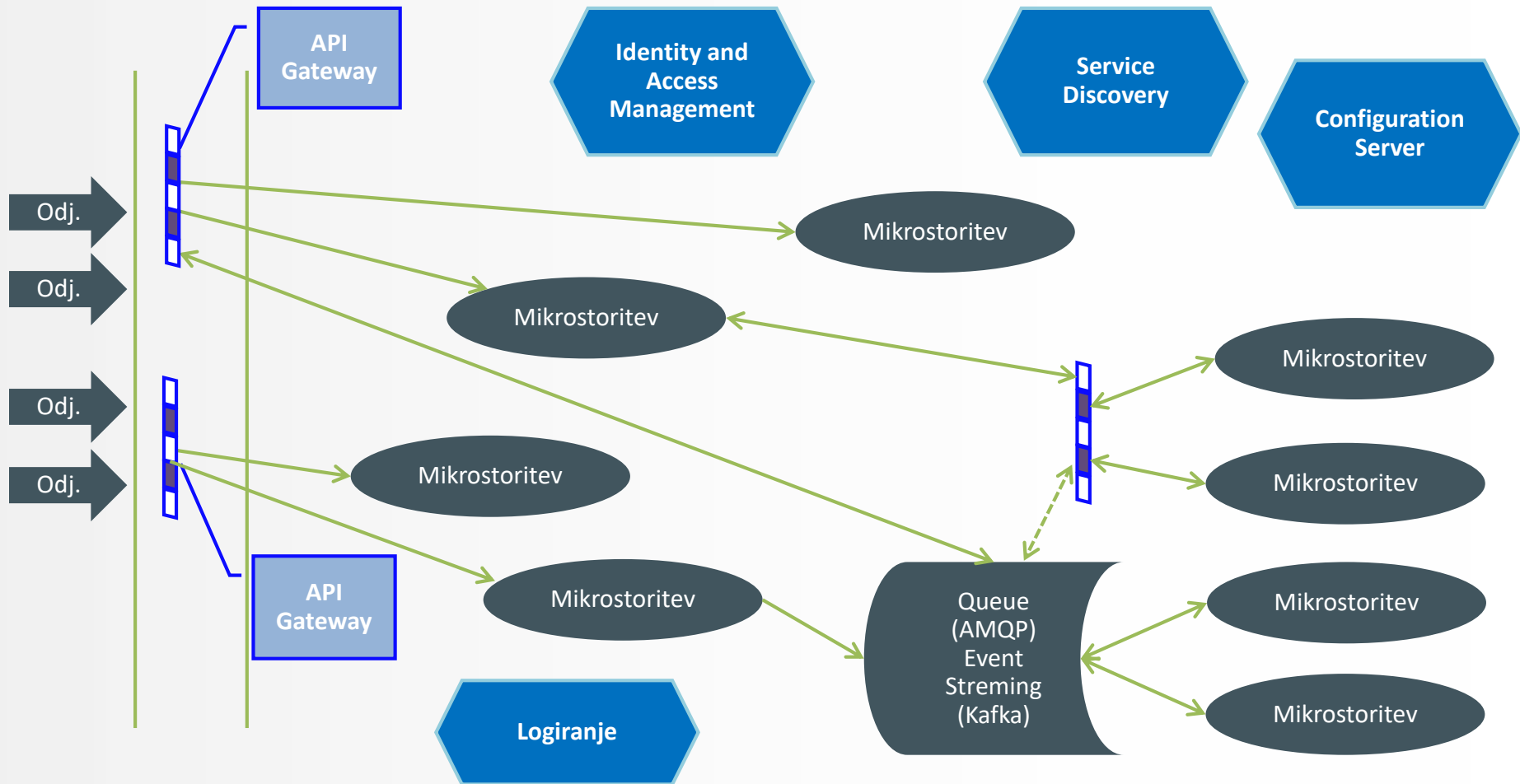
Clouds with
Docker support



JOYENT TRITON



Elementi arhitekture Cloud-native

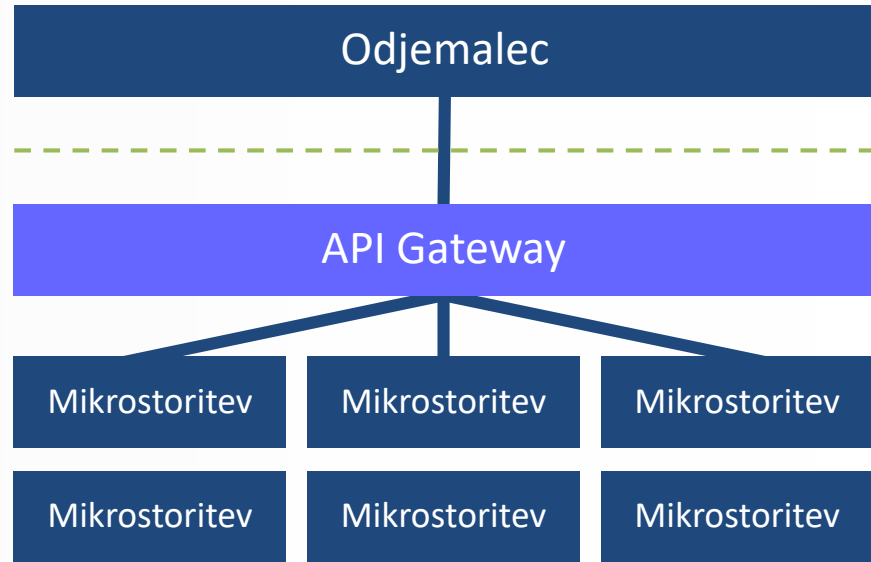


Dodatno

- Zbiranje metrik med izvajanjem, nadzor
- Uporaba naprednih konceptov REST
 - Verzioniranje, linki, odstranjevanje, povpraševanja, filtri, itd.
- Način specificiranja API-jev
 - OpenAPI / Swagger
 - JSON sheme
 - + dokumentiranje
- Način verzioniranja
- Objava

API Gateway

- Izpostavlja eno točko dostop za odjemalce
- Skrije podrobnosti back-enda
- Upravlja varnost
- Porazdeljuje obremenitev
- Meri in upravlja dostop
 - Npr. Rate-limiting
- Logira dostope
- Nadzira dostopnost
- Definira in upravlja politike dostopa



Logiranje

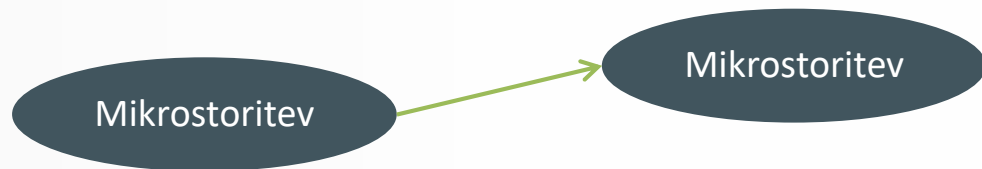
- Logi v smislu toka dogodkov

Zbiranje, agregiranje, iskanje, spremljanje, reševanje težav, itd.



Prekinjevalci toka (Circuit-Breakers)

- Prekinjevalci toka preprečujejo kaskadne napake
 - Preprečujejo oddaljene klice (npr. REST) takrat, ko povezave ne delujejo
- Kaskadne napake nastajajo, ko odjemalci čakajo na odziv oddaljene storitve (ali sistema).
- Zato je šibka sklopljenost pomembna
- Hystrix
- *@EnableCircuitBreaker*



Nameščanje mikrostoritev – življenjski cikel

- Po definiciji lahko vsako mikrostoritev:

- programiramo
- „build“-amo
- nameščamo

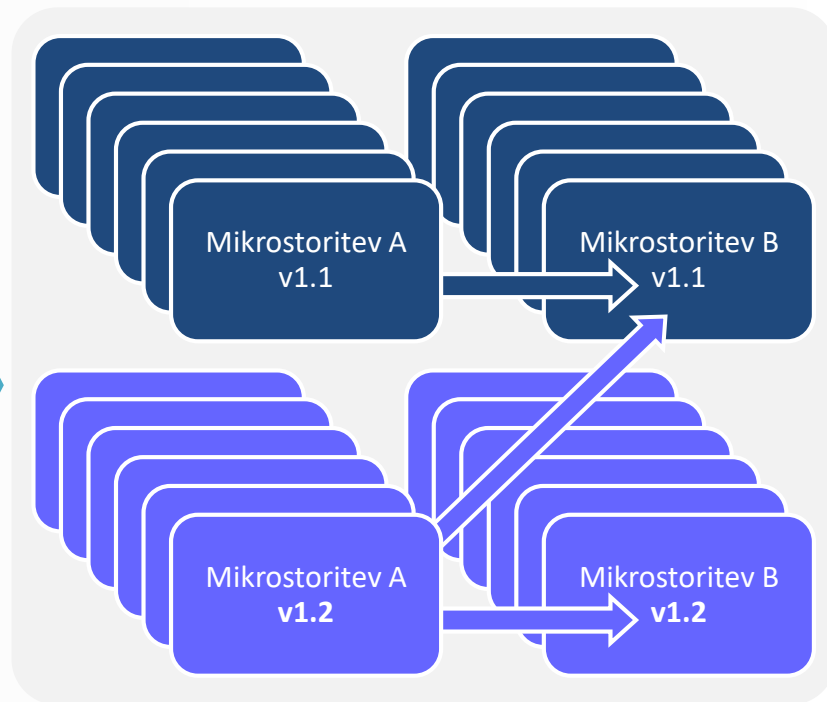
popolnoma neodvisno od drugih mikrostoritev.

- Nobene potrebe, da „zbiramo“ mikrostoritve in nameščamo celo aplikacijo naenkrat.
- Namestitve avtomatiziramo – DevOps, Continuous *

Več verzij mikrororitvev je nameščenih in deluje istočasno

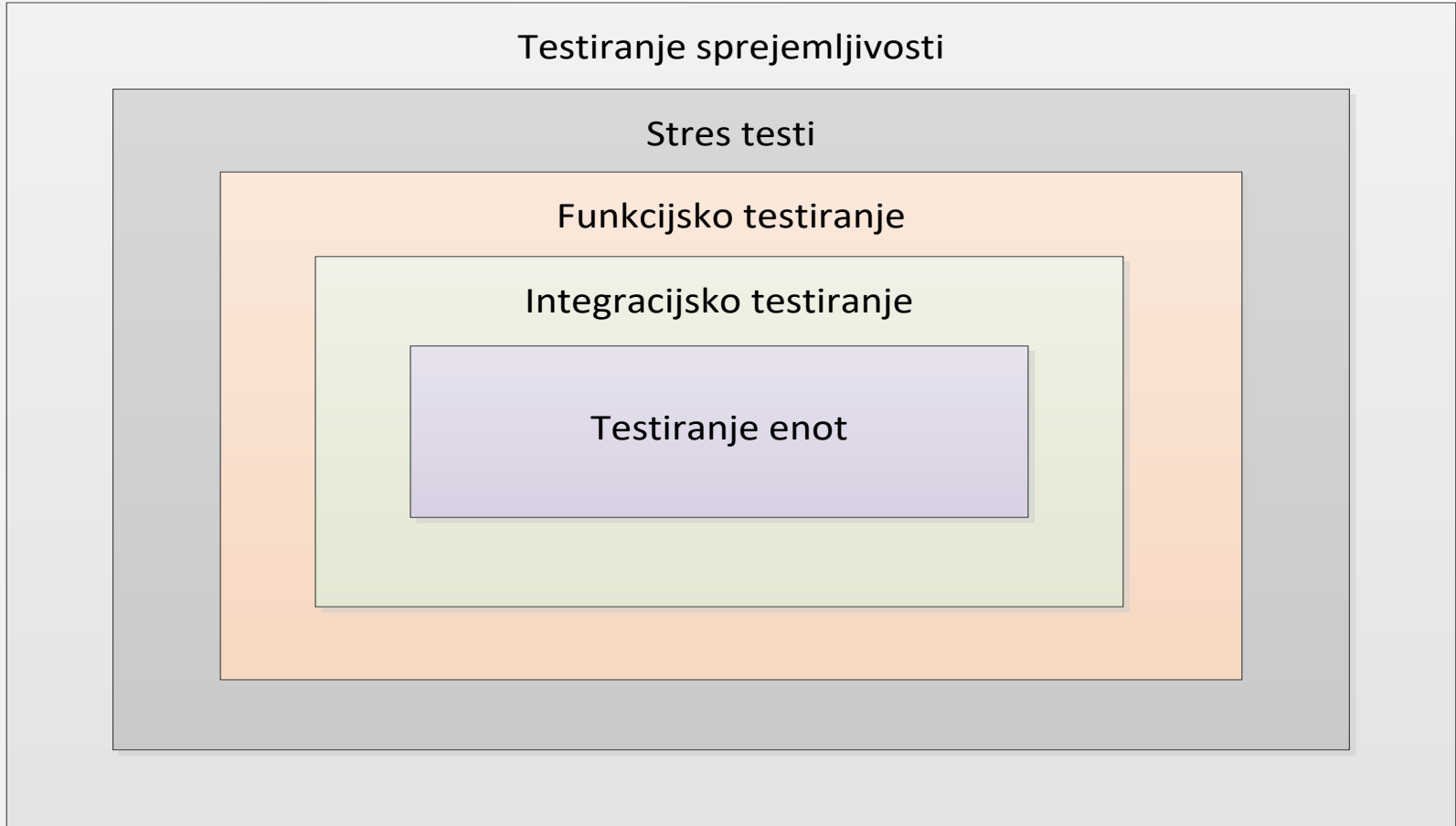


Poganjamo **eno** verzijo iste aplikacije v istem okolju



Poganjamo **več** verzij iste mikrororitvev v istem okolju

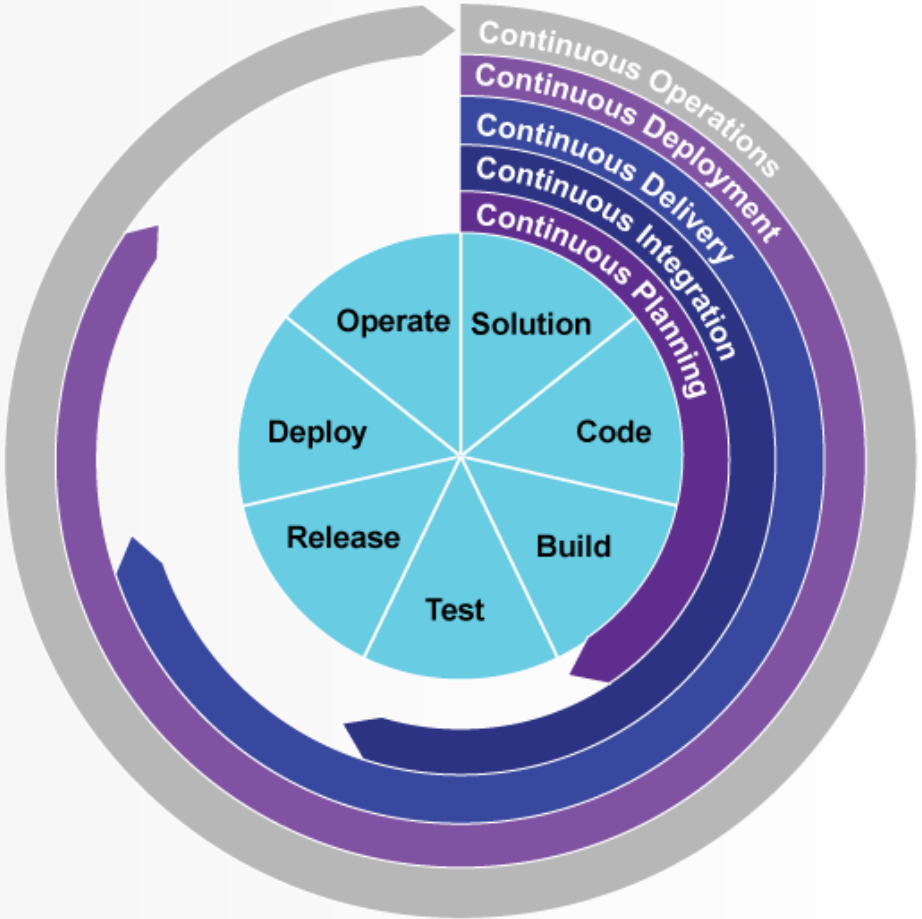
Avtomatizacija testiranja

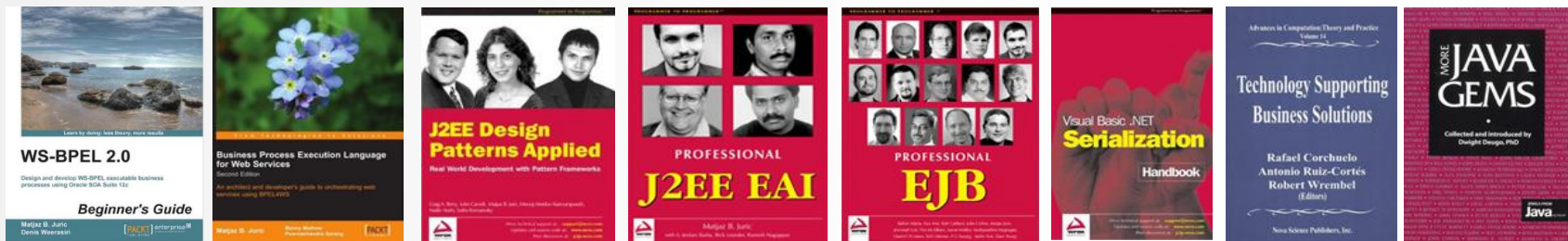


Zagotavljanje kakovosti

- Kakovost je sestavni element vsake aplikacije
 - Testiranje je le en del zagotavljanja kakovosti
- Odpornost na napake
 - Resilience – sestavni element vsake mikrostoritve
- Pomembna je celostna uporabniška izkušnja
 - Vedno načrtujemo celostno uporabniško izkušnjo

DevOps in agilni razvoj





HVALA!



e-naslov: <http://www.cloud.si>

e-naslov: <http://www.kumuluz.com>

e-pošta: info@cloud.si