

Distribovaná konfigurácia a client-side load balancing

Pavol Ďuďák, Šimon Zaujec

FEI STU

2022/11/02

Úvod

- 1 Service registry
- 2 Client-side load balancing
- 3 Distributed configuration

Service registry

- Registrácia aktívnych inštancií aplikácií
- URL inštancií aplikácií
- healthcheck
- vyťaženie zdrojov*

* - pokiaľ je to nutné pre účely load balancing-u

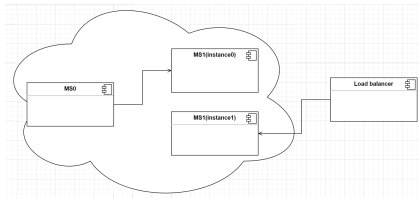
Consul

- Networking platform
- distribuovaný key value storage
- zero trust networking
- service mesh
- middleware automation
- health monitoring

Pre účely demo riešenia je Consul používaný ako KV storage pre ukladanie :

- URL inštancií aplikácií
- healthcheck inštancií aplikácií

Client-side load balancing



- Vynechanie volania externého LB
- priame volanie inštancie
- rôzne algoritmy pre výber volanej inštancie
 - **round robin**
 - Weighted Round Robin
 - Least Connection
 - Weighted Least Connection
 - ...

OpenFeign

- Generovanie implementácie volania REST na základe anotácií
- implementácia client-side load balancing
- service discovery
- failover

```
@RestController("/add")  
public interface IAddRestService {  
  
    1 usage 2 implementations  👤 dudakp  
    @PutMapping  
    AddResponseDto add(@RequestBody AddRequestDto req);  
  
}
```

Figure – REST service interface

OpenFeign - použitie

```
@FeignClient(value = "add-application", fallback = AddServiceClient.AddServiceClientFallback.class)
public interface AddServiceClient extends IAddRestService {
    2 usages  👤 dudakp
    @Component
    @RequestMapping(value = "/fallback")
    class AddServiceClientFallback implements AddServiceClient {
        1 usage
        private static final Logger LOG = LoggerFactory.getLogger(AddServiceClientFallback.class);

        1 usage  👤 dudakp
        @Override
        public AddResponseDto add( @NotNull AddRequestDto req) {
            LOG.warn("Using fallback add service!!!");
            return new AddResponseDto( result: req.a() + req.b(), req.a(), req.b(), source: "fallback");
        }
    }
}
```

Figure – FeignClient interface

OpenFeign - použitie

```
@FeignClient(  
    url = "http://numbersapi.com",  
    name = "ext-math-api",  
    fallback = ExternalMathOperationsClient.ExternalMathOperationsClientFallback.class  
)  
  
public interface ExternalMathOperationsClient {  
  
    1 usage 1 implementation 1 dudakp  
    @GetMapping("/{num}")  
    String getTrivia(@PathVariable("num") Long num);  
  
    1 usage 1 dudakp  
    @Component  
    class ExternalMathOperationsClientFallback implements ExternalMathOperationsClient {  
  
        1 usage 1 dudakp  
        @Override  
        public String getTrivia(Long num) {  
            return "unable to ge interesting fact about number: " + num;  
        }  
    }  
}
```

Figure – REST service interface pre externú službu

Distributed configuration

- Distribuovaná konfigurácia
- Uchovávanie konfigurácie pre aplikácie (napojenie na DB a iné)
- Consul K/V Store
- Properties v tvare kľúč - hodnota
- Jednoduché properties v K/V Store
- Konfiguračné súbory obsahujúce properties
- git2consul

Distributed configuration

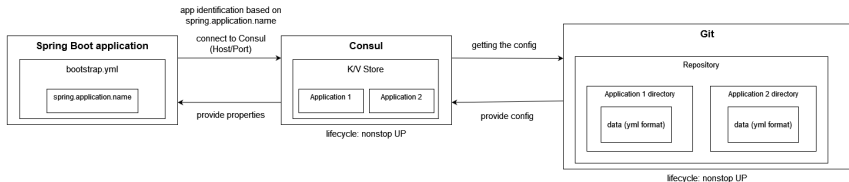


Figure – Setup distribovanej konfigurácie pre Spring Boot aplikáciu pomocou Consul a git2consul.

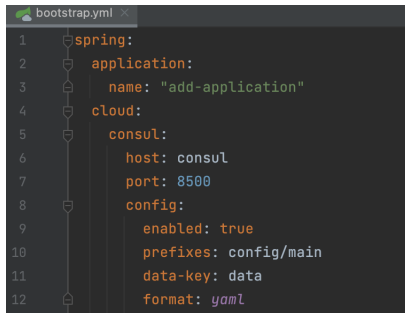
Distributed configuration



```
1  {
2    "version": "1.0",
3    "repos" : [{
4      "name" : "config",
5      "url" : "https://github.com/smooff/asos-consul-properties",
6      "branches" : ["main"],
7      "hooks": [{
8        "type" : "polling",
9        "interval" : "1"
10     }]
11   }]
12 }
```

Figure – Konfigurácia pre git2consul.

Distributed configuration



```
bootstrap.yml
1  spring:
2    application:
3      name: "add-application"
4    cloud:
5      consul:
6        host: consul
7        port: 8500
8        config:
9          enabled: true
10         prefixes: config/main
11         data-key: data
12         format: yaml
```

Figure – Použitie bootstrap.yml (spring-cloud-starter-bootstrap)

Distributed configuration

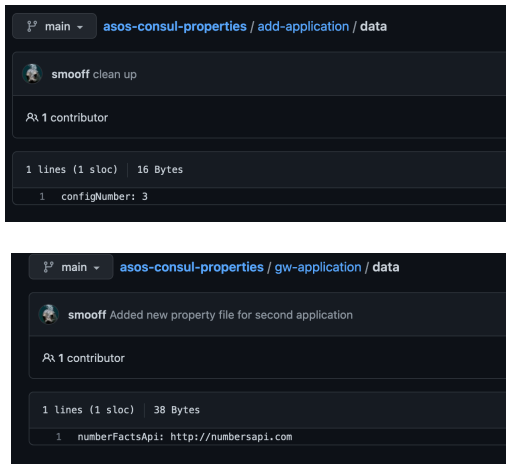


Figure – Properties pre 2 aplikácie v gite.

Použité technológie

Service registry, client-side load balancing a distribovaná konfigurácia

- Consul
- git2consul
- OpenFeign

Infraštruktúra

- Docker
- docker-compose

Implementácia riešenia

- Spring boot

Iné good-to-know technológie

- Observability
 - prometheus
 - jaeger/zipkin - distributed tracing
- Resilience
 - circuit breaker pattern (resilience4j)
- Secret storage
 - Hashicorp Vault
- Ops
 - Build pipelines (Jenkins, GitHub actions, Gitlab CI/CD)
 - Docker
 - K8S