Microservices

MicroServices -1
(7777)

MicroServices -2

MicroServices -3

Spring Cloud
Config Server

(9999)

uri =.....

GIT Repository

stock-price.properties

stock-price-dev.properties

stock-price-qa.properties

stock-price-uat.properties

stock-price-prod.properties

Config-props.git

Frontend

Android, iOS,
Windows etc.

Network

Third
Party
Integrations

Backend

Filters

Routing

API
Gateway

Spring
Cloud
Gateway

Http
(JSON)

SSO(JWT,
oAuth2.x
etc.)

Eureka

Service
Registry &
Discovery

Github/GitLab

Config
Server

MS-1   MS-3
MS-2
MS-4   MS-5

SB-Config-Server-Client-App [boot] [devtools]
  Deployment Descriptor: SB-Config-Server-Client-App
  JAX-WS Web Services
  Java Resources
    src/main/java
      in.ineuron
      in.ineuron.restcontroller
        MsgController.java
    src/main/resources
      META-INF
      static
      templates
      bootstrap.properties
    src/test/java
    target/generated-sources/annotations
    target/generated-test-sources/test-annotations
    Libraries
  Deployed Resources
  src
  target
  HELP.md
  mvnw
  mvnw.cmd
  pom.xml

SB-Config-Server-App [boot] [devtools]
  Deployment Descriptor: SB-Config-Server-App
  JAX-WS Web Services
  Java Resources
    src/main/java
      in.ineuron
    src/main/resources
      static
      templates
      bootstrap.properties
    src/test/java
    target/generated-sources/annotations
    target/generated-test-sources/test-annotations
    Libraries
  Deployed Resources
  src
  target
  HELP.md
  mvnw

TEK-Leads / Config-Properties Public

<> Code   Issues   Pull requests   Actions   Projects   Security   Insights

master Config-Properties / client-config-dev.properties

TEK-Leads Create client-config-dev.properties

1 contributor

1 lines (1 sloc)   73 Bytes

1   msg = Hello world - this is from config server - Development Environment
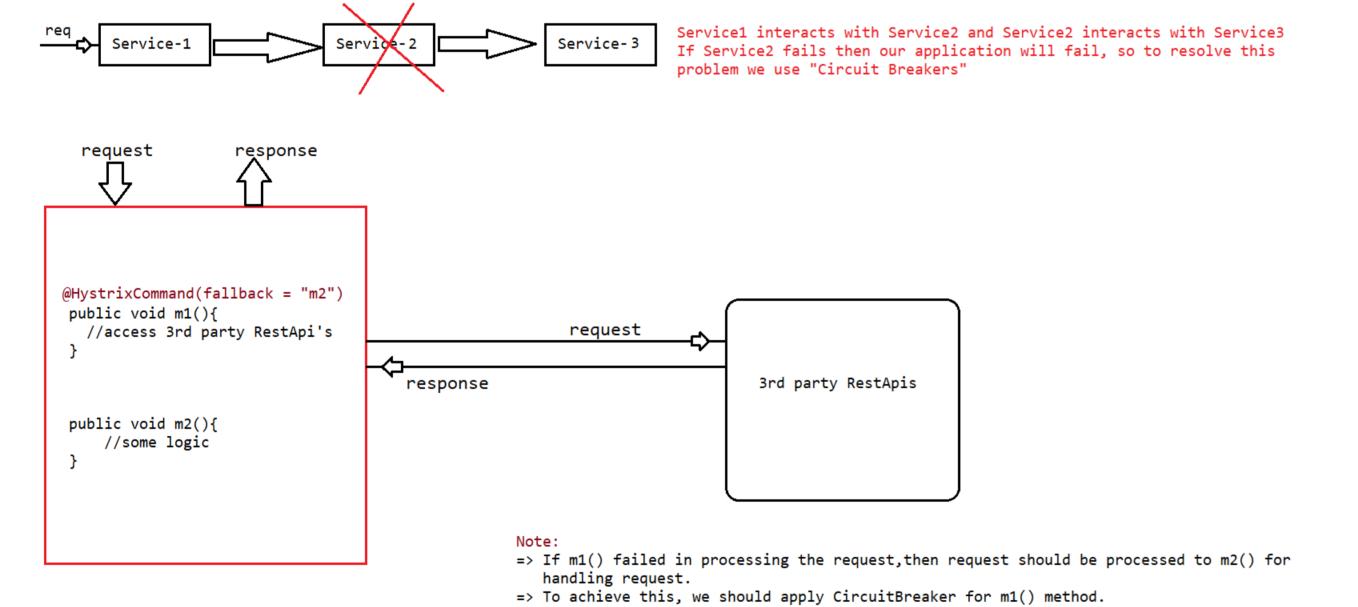

Master

G - 1

Slave

G - 2

C

B

Power Based Machines

Circuit Breakers are used to protect us from damage

---

req   Service-1   Service-2   Service-3

Service1 interacts with Service2 and Service2 interacts with Service3
If Service2 fails then our application will fail, so to resolve this
problem we use "Circuit Breakers"

---

request   response

@HystrixCommand(fallback = "m2")
public void m1(){
  //access 3rd party RestApi's
}

public void m2(){
    //some logic
}

request

response

3rd party RestApis

Note:
=> If m1() failed in processing the request,then request should be processed to m2() for
   handling request.
=> To achieve this, we should apply CircuitBreaker for m1() method.

ProductWeb
:8081

RestController   <<Interface>>
                      ProductService
CRUD
EndPoints   CRUD
                      Methods

With Spring

ProductServer
:8080

RestController

CRUD
EndPoints

MongoDB

Feign + Hystrix

Product
Alternate
Service
Proxy
Component

ProductService
Proxy

ProductAlternate
Service
Proxy

ProductServer 2
:8079

RestController

CRUD
EndPoints

MongoDB

Angular UI
:4200

App