Distribuovaná konfigurácia a client-side load balancing

Pavol Ďuďák, Šimon Zaujec

November 2, 2022

1 Úvod

Pri veľkých distribuovaných aplikáciách vznikajú nové problémy, ktoré pri monolitických aplikáciách neexistovali. Udržiavanie niekedy desiatok aplikácií súbežne a komunikácia medzi nimi nie je triviálna. Pre riešenie týchto problémov vstupujú do vývoja techniky ako service registry, client-side load balancing, distributed configuration a veľa iných. Tento dokument sa zaoberá spomínanými technikami.

2 Service registry

Pri mikroservisnej architektúre medzi sebou komunikujú niekedy až desiatky rôznych aplikácií. To znamená že každá aplikácia musí mať informáciu o URL všetkých aplikácií s ktorými potrebuje komunikovať. Tieto informácie môžu byť uložené v konfiguračnom súbore aplikácie, avšak tento spôsob je nepraktický.

Pre vyriešenie tohto problému sa používa service-registry pattern. Do deploymentu riešenia sa pridá nová aplikácia tzv. service registry, ktorej úlohou je udržiavať informácie o všetkých aplikáciách v systéme. Zároveň na túto aplikáciu sa pripájajú všetky ďalšie aplikácie, ktoré potrebujú informácie o URL iných aplikácií. Týmto spôsobom je eliminovaná potreba dopĺňať URL aplikácií do konfiguračných súborov.

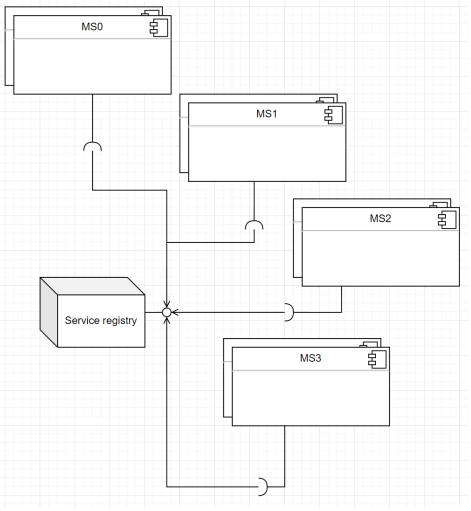

Figure 1: Service registry

3 Client-side load balancing

Pri veľkých riešeniach nestačí aby bežala iba jedna inštancia každej aplikácie, preto sa inštancie replikujú. Tu nastáva problém že pokiaľ chce akákoľvek aplikácia volať inú štandardne musí ísť cez load balancer. Takýto flow môže spôsobovať performance problémy kedže pre každé volanie aplikácia A aplikáciou B je nutné, aby aplikácia A zavolala najskôr load balancer a až z neho je volaná aplikácia B. Tento zbytočný medzikrok vieme odstránit pomocou techniky client-side load balancing. Pokiaľ má systém implementovaný service registry, tak existuje úložisko URL všetkých inštancií aplikácií a teda zaniká potreba volať load balancer, ale aplikikácie budú volať priamo inštancie aplikácií.

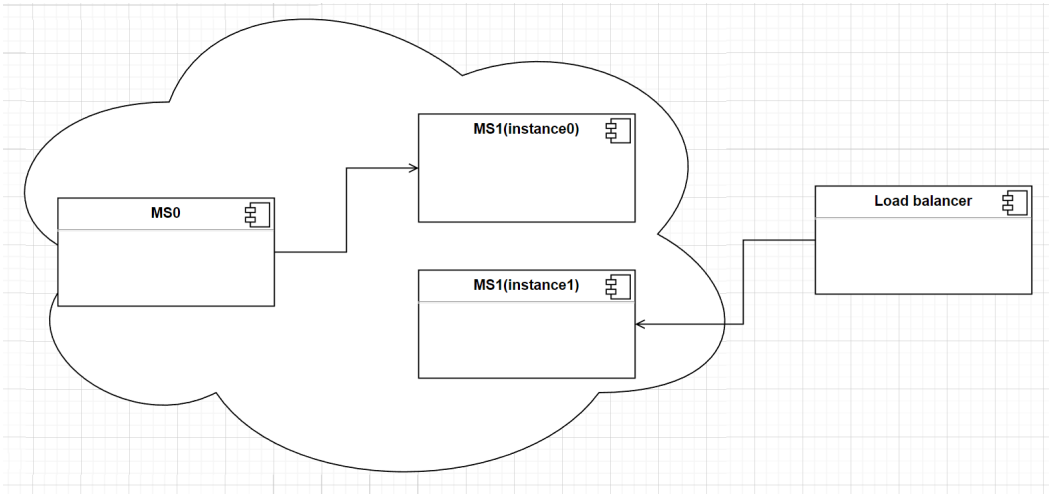

Figure 2: Client-side load balancing

Pri client-side load balancing-u existujú rôzne algoritmy, ktoré určujú, ktorá inštancia bude volaná. Najjednoduchší je round-robin, ktorý si usporiada inštancie do zoznamu a postupne volá ďalšiu inštaciu, ktorá je v poradí až kým nenarazí na koniec kedy ide od začiatku. Ďalšie algoritmy sú napríklad:

- Weighted Round Robin,
- Least Connection,
- Weighted Least Connection,
- Resource Based (Adaptive),
- Resource Based (SDN Adaptive),
- Fixed Weighting,
- Source IP Hash,
- URL Hash

Avšak round-robin je najpoužívanejší algoritmus. Výber iného musí byť riadne odôvodnený špeciálnym use-case-om.

4 Distributed configuration

Udržiavanie konfigurácie v monolitických aplikáciach nemusí byť takým problémom, stačí nám možno jeden konfiguračný súbor a to je všetko. Avšak udržiavanie konfigurácie v projektoch, ktoré sa skladajú z veľkého množstva aplikácií môže byť problémové - ťažšie udržateľné. Vtedy nastupuje na scénu distribuovaná konfigurácia. Pod týmto pojmom si vieme predstaviť akýsi trezor, ktorý uchováva rôzne konfigurácie pre rôzne aplikácie, ktoré sú na neho napojené. Zväčša sa táto konfigurácia skladá z tzv. konfiguračných properties, tie môžu byť napr. URL pre napojenie na databázu.

Consul, ktorý sme využili na Service discovery, slúži aj na poskytovanie distribuovanej konfigurácie tzv. K/V Store, alebo inak aj Key Value Store. K/V Store teda slúži na konfigurovanie aplikácií, koordináciu služieb a iné.

Konfigurácia v Consul K/V Store môže predstavovať obyčajné páry kľúč - hodnota, ale môže byť reprezentovaná aj celými súbormi, ktoré obsahujú properties kľúč - hodnota. Tento typ konfigurácie nás vedie ku komunitnému projektu *git2consul*, ktorý slúži na zrkadlenie jedného alebo viacerých git repozitárov do Consul K/V Storu. Cieľom git2consul je spraviť z gitu úložné miesto pre konfiguračné properties, ktoré sa neskôr doťahujú do Consulu.

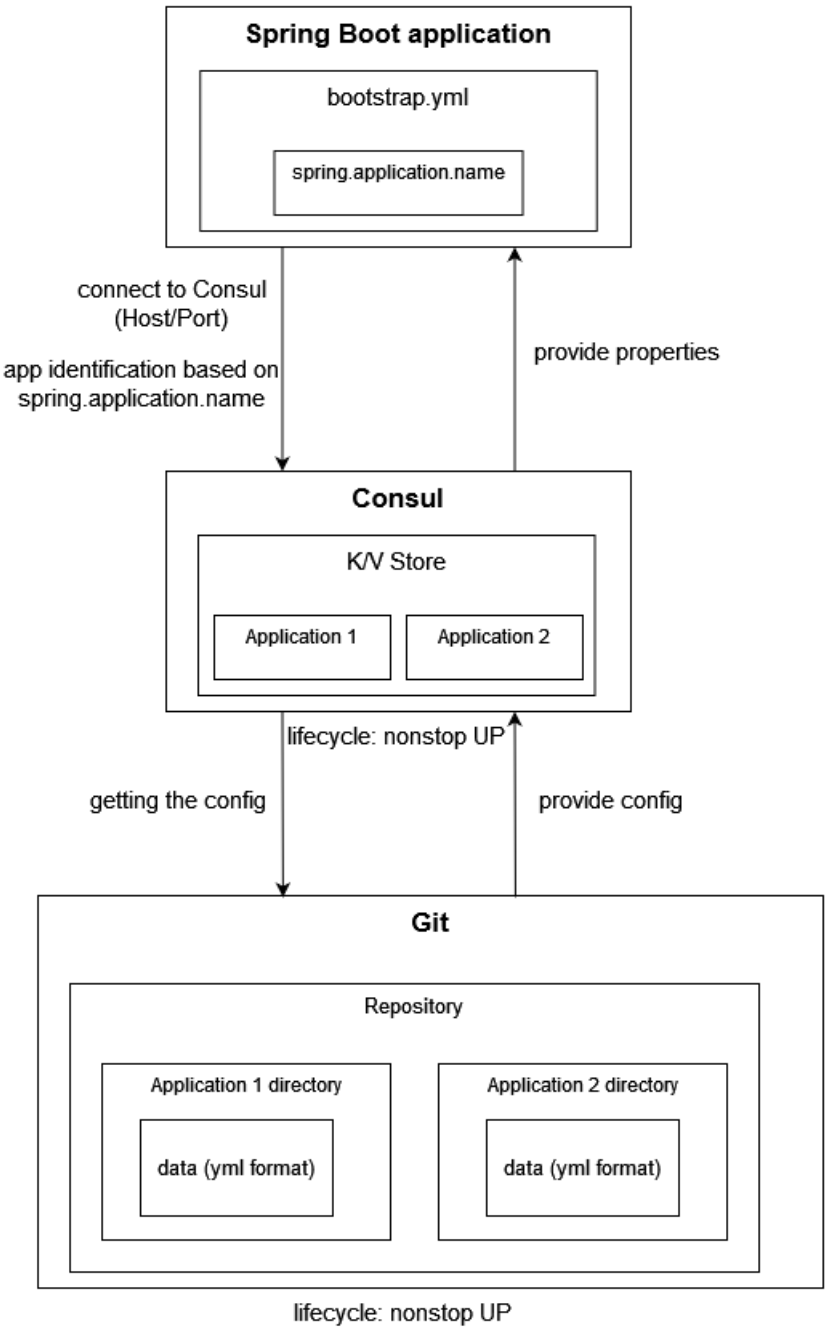

Figure 3: Setup distribuovanej konfigurácie pre Spring Boot aplikáciu pomocou Consul a git2consul.

5 Záver

Výsledkom tejto práce je funkčné demo Spring Bootovej aplikácie, ktorá sa registruje v Service registry, využíva Client-side load balancing a konfiguruje sa na základe K/V Store properties, ktoré sa načítavajú z gitu.