

Kako podići interakciju korisnika mobilne aplikacije za 160 posto?

Da bi proizvod postigao uspjeh na tržištu, potrebna je dobra tehnička stabilnost te zadovoljstvo i angažman korisnika. Mnoge velike tvrtke, ali i startupi, ulažu veliku količinu vremena da bi postigli sva tri faktora, ali nerijetko ostaju razočarani, najčešće zato što pravi potez nije napravljen u pravo vrijeme.

Zašto razmišljati o tome i kada je pravo vrijeme?

U engleskom jeziku postoji izraz **FOBO** (Fear of better option), a možeš reći da te obuzeo onda kada vidiš da svi drugi rade bolje, a tebe uopće ne zanima što to oni i kako rade.

Svaki kreator sanja dan kada će njegov proizvod postati popularan i omiljen kod korisnika. Neki su to i postigli u prošlosti pa ih je konkurencija prestigla, a neki tek idu tim putem. U oba slučaja, jako je bitna činjenica da veliki rezultati zahtijevaju mnogo rada.

Dakle, ako želiš veliki rezultat, a istodobno ne jenjava osjećaj da drugima ide bolje, stani, razmisli, zapitaj se: "*Postoji li način da ovaj proces učinim bržim i lakšim?*"

Odgađanje ne rješava problem. Ne budi kukavica – suoči se s problemom!

Odgađanje ne rješava problem. Uzmimo za primjer zdravlje vašeg limenog ljubimca. Ako ste danima odgađali odlazak mehaničaru zbog upaljene lampice motora, vjerojatno će doći do većeg kvara. Samim time i većeg troška, bijesa, nezadovoljstva, vremena izgubljenog na popravak.

U A1 smo bili svjesni da postoje modernije tehnologije koje bi mogle donijeti rezultat. Ali, kako objasniti poslovnom dijelu tvrtke da bi to trebalo biti dobro? Kako objasniti *stakeholderima* da se već godinu dana radi nešto što neće dati tražene rezultate? Čak štoviše, treba dodatnih godinu dana rada u novim tehnologijama koje će donijeti tražene rezultat

Ipak, našli smo način, i to na najvrucem proizvodu kojeg A1 nudi, mobilnoj aplikaciji koja treba biti spona između korisnika i telekoma, a korisniku služi za jednostavno upravljanje A1 uslugama, [MojA1 aplikaciji](#).



Leži li tajna uspjeha u modernizaciji?

S vremenom, kako se radi na razvoju aplikacije, raste količina programskog koda, čime svaka promjena postaje sve kompleksnija, prvenstveno zbog snalaženja unutar velikog projekta, a proces prilagodbe novih programera na projekt je puno složeniji.

Modernizacijom smo planirali obuhvatiti ponovni razvoj jednog dijela sustava koji je prva točka u dodiru s korisnicima. Stare tehnologije je bilo potrebno zamijeniti novima. Zašto? Pa upravo radi stabilnosti, robusnosti, skalabilnosti.

Korištenjem alata Docker olakšali smo upotrebu kontejnerizirane aplikacije. Programer ne mora ovisiti o razvojnoj okolini, dovoljno je da je njegov operativni sustav podržan od strane Docker-a, što znači da se ne mora brinuti o razlikama između razvojnog, testnog ili produkcijskog okruženja, jer će biti isto, što uvelike olakšava proces traženja i ispravljanja grešaka, kao i razvoj novih funkcionalnosti.

Red Hat Openshift Container platforma omogućila nam je olakšanu i automatiziranu izgradnju, implementaciju, skaliranje i upravljanje zdravljem aplikacije, a paralelno smo dobili široki spektar programskih jezika, framework-a i baza podataka na raspolaganje.

Očuvanje kontinuiteta poslovanja i dobivanje što boljeg korisničkog iskustva, brza detekcija uzroka smanjene kvalitete i devijacija ili prekida usluga u trenutku odluke bila je naša nužnost. Stoga smo se odlučili na AppDynamics rješenje, kojim je svaki mikroservis dobio poseban nadzor od strane AppDynamics agenta, čime smo spojili kompletan nadzor aplikacije, od mobilnog uređaja do baze podataka.

Vrijedi spomenuti da smo, kako bi svakodnevno mogli kvalitetno raditi na održavanju svih servisa, odabrali ELK stack, koji predstavlja skupinu alata Elasticsearch, Logstash i Kibana. Automatiku, nadzor i razumijevanje isporuke aplikacija, te jednostavnije verzioniranje cijelog procesa postigli smo uz ArgoCD alat za Kubernetes.

Kao šlag na kraju, odlučili smo se za Java programski jezik i Spring Boot, koji nam u kombinaciji sa Redis cache-om i REST-om, daje enormne mogućnosti napretka u kvaliteti svakog pojedinog servisa.



Izazov mikroservisa u telekomu

Sam Newman navodi kako je glavni problem SOA arhitekture u tome što SOA ne daje programerima upute kako da od monolita naprave servisnu arhitekturu. Između ostalog, ne definiraju se limiti pojedinog servisa, ne navodi se primjer granulacije nove arhitekture, kao ni praktični primjeri koji bi pokazali kako pravilno odvojiti servise jedne od drugih. Mikroservisna arhitektura se, upravo suprotno, razvila iz stvarnih primjera, no i dalje se može smatrati posebnim SOA pristupom.

Vođeni stvarnim primjerima, pronašli smo kritične točke sustava, koje traže veliku procesorsku snagu i memoriju, rascijepali aplikaciju na 10ak manjih aplikacija, koje međusobno komuniciraju putem REST protokola, dizajnirali sustav tako da bude što više dijeljena podataka i krenuli s eksperimentima. U startu smo prihvatili da ćemo sigurno pogriješiti, neke stvari će trebati dodatno rascjepkati, na još manje dijelove, kako bi ih mogli zasebno skalirati i učiniti robusnijima.

Ponovnim razvojem sustava unaprijed ćemo pripaziti na probleme koji su nas ranije mučili, posvetiti im više pažnje i kvalitetnu razradu. Ako uspijemo u tome, smanjit ćemo broj interakcija naših korisnika koje su ranije odlazile u vjetar, a u vjetar je odlazilo puno toga, od nemogućnosti da korisnik pregleda stanje svojih jedinica, do neuspješne kupovine bona ili dijeljenja MB sa drugim korisnikom. Samim time, povećat ćemo broj uspješnih interakcija.

Skalirat ćemo dijelove sustava koji su najosjetljiviji, povećat ćemo procesorsku snagu i pojednostavniti procese.

Što su nam donijeli mikroservisi?

Godinu dana kasnije, imamo aplikaciju koja ima čak 50 puta manje grešaka sa 10 puta bržim prosječnim vremenom odziva. Interakcija korisnika se povećala za 160%, a broj kupovina putem aplikacije se povećao nekoliko puta.

Sve to utjecalo je na veći broj korisnika, kojih imamo 24% više te na zadovoljstvo korisnika koji su MojA1 app stavili na top liste na PlayStore i AppStore trgovinama.

S tehničke strane, na Openshift platformi smo sa relativno malo resursa postigli ogroman uspjeh, uzimajući u obzir izrazito visok broj transakcija. Svedjedno, potrebno je konstantno raditi na optimizaciji resursa, kako bi se postigli maksimalni rezultati.

Quota [Learn More](#)

a1-moja1--resource-quota

Limits resource usage within this project.



Resource Type	Used	Max
CPU (Request)	6 cores	8 cores
Memory (Request)	8112 MiB	12 GiB
CPU (Limit)	7200 millicores	8 cores
Memory (Limit)	9912 MiB	12 GiB

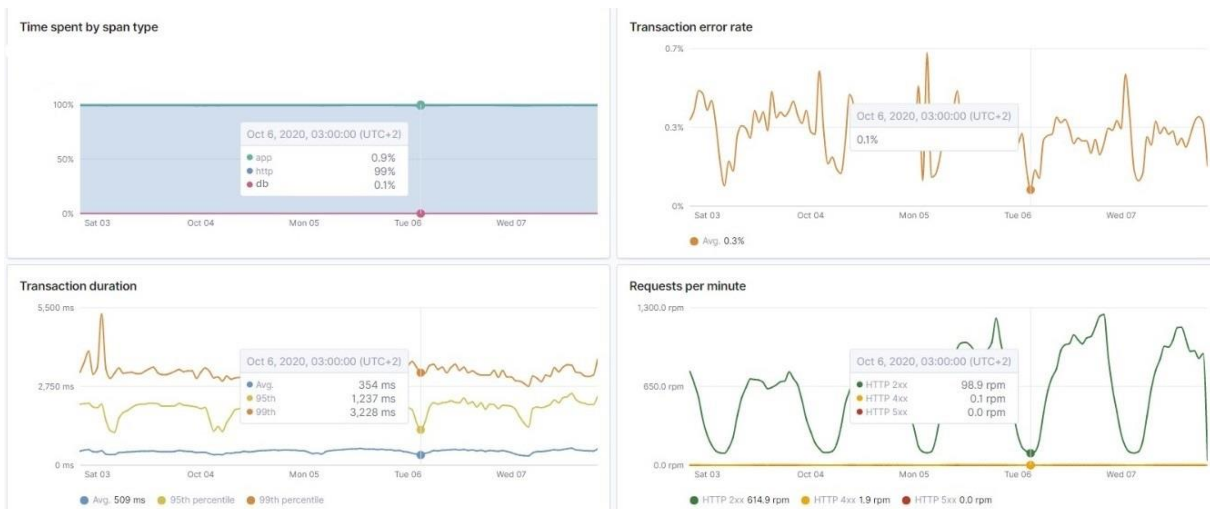
Na slici ispod vidimo kako se kontejneri, koji prikazuju podatke o računima i dostupnim opcijama za kupovinu, nakon rušenja neuspješno pokušavaju podignuti zbog neuspjeha pri provjeri zdravstvenog stanja mikroservisa (healthcheck probes). Ipak, aplikacija je cijelo vrijeme bila dostupna, jer je svaki kontejner imao svog duplikata, jednog ili više njih, koji su i dalje neumorno servirali korisnike. Zaključili smo da se vrijeme dostupnosti servisa, a samim time i aplikacije, višestruko povećalo u odnosu na ranije.

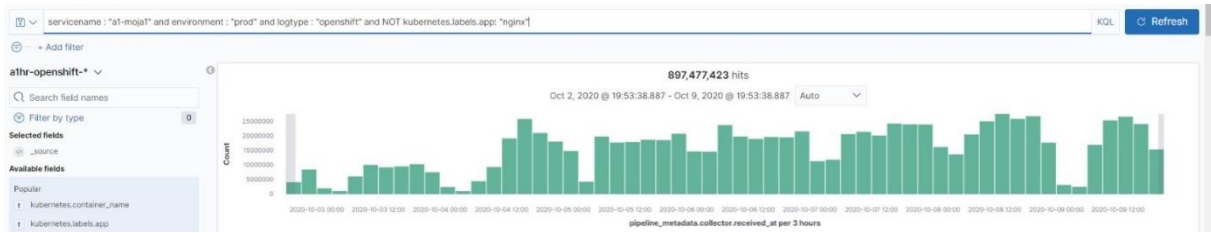
Name	Status	Containers Ready	Container Restarts	Age
blils-7-jm7z	Running	1/1	0	7 hours
shop-5-48q5	Running	1/1	0	10 hours
shop-5-44j8	Evicted	0/1	0	10 hours
shop-5-2xxvl	Evicted	0/1	0	10 hours
shop-5-ngjst	Evicted	0/1	0	10 hours
shop-5-pjpsl	Evicted	0/1	0	10 hours
shop-5-8b47	Evicted	0/1	0	10 hours
shop-5-tgcpn	Evicted	0/1	0	10 hours
shop-5-9ggn6	Evicted	0/1	0	10 hours
shop-5-xt7h	Evicted	0/1	0	10 hours
shop-5-2z8qb	Running	1/1	0	a day
blils-7-raq2kx	Evicted	0/1	0	2 days
blils-7-xtqfw	Running	1/1	1	2 days
blils-7-2975q	Evicted	0/1	0	2 days
blils-7-tdsmg	Evicted	0/1	0	2 days
blils-7-4tbnw	Evicted	0/1	0	2 days
blils-7-3fsv6	Evicted	0/1	0	2 days
blils-7-8tcb7	Evicted	0/1	0	2 days



Veliki dobitak je i isporuka novog inkrementa svakog pojedinog servisa. Postoji više strategija isporuke: Rolling strategija, Recreate strategija, vlastite strategije... Rolling strategijom polako se mijenja prethodna instanca servisa sa novom instancom. Rolling isporuka čeka da novi servis uspješno prođe zdravstvene provjere (readiness check, liveness check) prije nego spusti prethodne instance servisa. Ako nova isporuka zbog nekog razloga ne uspije, nema spuštanja prethodnih instanci. Na taj način dobiva se najveća razina dostupnosti servisa, a na štetu povećanih resursa, no to je ipak bio glavni razlog da odaberemo Rolling strategiju.

U konačnici, broj neuspješnih transakcija sada se mjeri u promilima, a do njihovog potpunog nestanka je jako teško doći, uzimajući u obzir širinu i kompleksnost telekom sustava. Prosječno vrijeme obrade transakcije uvijek je manje od 500 ms, ali i dalje imamo mjesta za napredak, prvenstveno za slučajeve u kojima dolazi do greške. Postizanjem tog cilja povratno ćemo dobiti još viši stupanj zadovoljstva korisnika, ali i mogućnost brze reakcije na probleme koji nam mogu uzrokovati neplanirane gubitke.



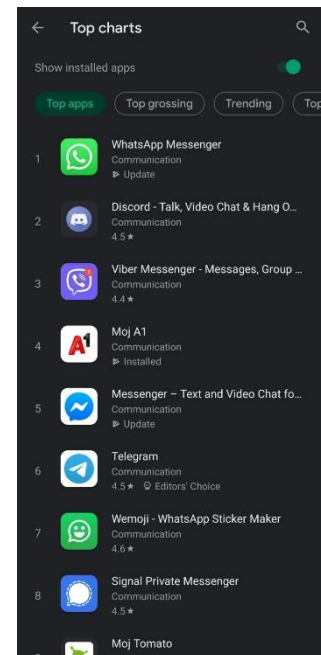


Da poslovni dio tvrtke i *stakeholderi* budu sretni.

Ranije, svako dodavanje nove funkcionalnosti uzrokovalo je negodovanje i nesuglasice između *stakeholdera* i programera. Programeri su bili svjesni nedostataka, a *stakeholderi* nisu shvaćali prave detalje i uzroke takvih reakcija.

Od *stakeholdera* smo tražili da unaprijed iznesu sve što žele vidjeti u aplikaciji, tako da im odmah omogućimo nove stvari i vrijeme čekanja učinimo zanimljivim. Objeručke su prihvatili i nestrpljivo čekali vrijeme kada više neće imati zavezane ruke po pitanju novih zahtjeva. Tražili su da krenemo s korisnicima na bonove i da bar za njih osvježimo aplikaciju, dizajnom, brzinom i povećanjem uspješnih transakcija. Iako smo znali da dugoročno to nije održiva investicija, postigli smo privremeno zadovoljstvo *stakeholdera* te si osigurali vrijeme za nastavak procesa.

Danas je vrlo teško osigurati zamrzavanje procesa novih zahtjeva i to je ujedno bio najveći izazov naše tehničke modernizacije: planiranje resursa i ulaganja za modernizaciju u paraleli sa novim zahtjevima, održavanjem zadovoljstva korisnika i efektom nepovratnog troška.



«Ugasimo» FOBO – eksperimentiranjem!

Kao agilni tim u agilnoj kompaniji, cilj je i dalje eksperimentirati sa malim izmjenama i pritom oslušivati reakcije korisnika i, ovisno o reakciji, nastaviti sa eksperimentom ili ga u potpunosti zaustaviti. Želimo ugasiti strah od bolje opcije (FOBO) i postati oni koji konstantno traže bolje opcije, pomoću povratnih informacija od korisnika, te time pokušati potaknuti i druge da prevladaju taj strah, kako unutar kompanije, tako i izvan nje.

Jozo Petrović

Senior Design and Development Specialist, A1 Hrvatska d.o.o

<https://jobs.a1.com/hr/jozo-petrovic/>

<https://www.linkedin.com/in/jozopetrovic/>