Engineer

Tjenestenivåmål

◎ et målbart pålitelighetsmål som en tjeneste skal oppfylle

Hva det er

Et tjenestenivåmål er et krav til hvor godt en tjeneste oppfører seg, skrevet så presist at hvem som helst kan sjekke om det ble innfridd: en andel forespørsler som skal lykkes, eller besvares raskt nok, over en angitt periode. På engelsk heter det service level objective, og nesten alle forkorter det til SLO, også i norske team. Googles bok om Site Reliability Engineering definerer det som en målverdi eller et verdiområde for ett tjenestenivå, målt med en indikator.

Den indikatoren er tjenestenivåindikatoren, eller SLI-en: en nøye definert størrelse som beskriver én side av tjenesten brukerne faktisk får. Latens for forespørsler er det vanlige førstevalget, tett fulgt av feilrate og gjennomstrømning. SRE Workbook anbefaler å skrive hver SLI som et forhold, gode hendelser delt på alle hendelser, slik at den går fra 0 %, der ingenting virker, til 100 %, der ingenting er ødelagt. Målet er da en strek på den skalaen: indikatoren skal holde seg på riktig side av en grense, eller mellom en nedre og en øvre.

Et fullstendig mål nevner alt som trengs for å sjekke det. Bokas eksempel lyder: 99 % av Get RPC-kall, i snitt over ett minutt, fullføres på under 100 ms, målt over alle backend-servere. Én indikator kan ha flere mål samtidig, for eksempel 90 % av kallene under 1 ms, 99 % under 10 ms og 99,9 % under 100 ms, fordi én persentil sjelden beskriver en hel fordeling. Workbook skiller også mellom spesifikasjonen, det utfallet som betyr noe for brukerne, og implementasjonen, stedet der det måles: en serverlogg, en sonde som kjører en nettleser, eller kode på selve siden. Hvert sted ser andre feil, og valget hører hjemme i definisjonen.

En SLO er ikke en SLA. En tjenestenivåavtale (service level agreement) er en kontrakt med brukerne, uttalt eller underforstått, som knytter konsekvenser til om målene i den nås, oftest en rabatt eller en bot. Bokas test er å spørre hva som

skjer når målet ikke nås: følger det ingen uttalt konsekvens, er det en SLO. En fotnote legger til at folk som snakker om en brutt SLA, nesten alltid mener en SLO som ikke ble nådd, fordi et ekte brudd på en SLA kan ende i retten som kontraktsbrudd. Google Search, påpeker boka, har ingen SLA overfor allmennheten, men et utfall der koster likevel omdømme og annonseinntekter.

Bruk begrepet i en beslutning

Bruk «Hva er et tjenestenivåmål?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Overvåking og varsler som sier fra mens en feil fortsatt er liten, og tallene som viser hvor ofte den holder seg liten.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Bygde selskapets første observerbarhetsdashbord — selskapets første innsyn i egne systemer

Bygde selskapets første observerbarhetsdashbord Ta kontakt →

Hvorfor det er viktig

Uten et uttalt mål finner brukerne opp sitt eget. SRE-boka beskriver begge måtene det går galt på: noen stoler mer på tjenesten enn de burde, og mulige nye brukere tror den er mer ustabil enn den er, og holder seg unna. Eksempelet er Chubby, Googles låsetjeneste for distribuerte systemer. Globale utfall var så sjeldne at andre tjenester i det stille begynte å regne med at den aldri sviktet, og de tjenestene røk de få gangene den gjorde det. Løsningen var å la den globale instansen nå sin SLO uten å overgå den med mye: i hvert kvartal der ekte feil ikke hadde presset tilgjengeligheten under målet, tok SRE-teamet systemet ned med vilje.

Målet avgjør også diskusjonen om hvor pålitelig som er pålitelig nok. SRE Workbook kaller 100 % pålitelighet feil mål, og begrunnelsen er praktisk. Komponenter svikter samtidig, uansett hvordan de er bygd. Kjeden av nettverk og enheter mellom tjenesten og kundene svikter også, så kundene ville aldri sett de 100 prosentene. Hver ekstra nier koster mer enn den forrige, mens verdien for brukerne nærmer seg null. Og siden endringer er den vanligste årsaken til utfall, kan en tjeneste som aldri får svikte, heller aldri oppdateres. Når målet ligger under 100 %, er avstanden mellom målet og det perfekte et rom som kan brukes på utgivelser, eksperimenter og planlagt vedlikehold. Det rommet kalles feilbudsjettet (error budget), og det er rett og slett 100 % minus SLO-en. Noen må eie avveiningen mellom nye funksjoner og stabil drift; Workbook peker på teknologidirektøren (CTO) i en liten organisasjon og produkteieren eller produktsjefen i en større.

Til slutt gjør målet overvåking om til beslutninger. Boka beskriver en sløyfe: måle indikatorene, sammenligne dem med målene, avgjøre om noe må gjøres, og gjøre det. Stiger latensen og vil krysse streken innen noen timer, sjekker teamet om serverne er CPU-bundet, og setter inn flere. Uten målet, påpeker boka, kan ingen si om de skal handle, eller når.

Slik ser det ut i praksis

Ta et checkout-API og et mål skrevet slik Workbook foreslår, med indikatoren skilt fra stedet den måles, vinduet angitt og begrunnelsen lagt ved siden av:

```
Tjeneste:     checkout-API
SLI:          andel POST /checkout-forespørsler besvart uten
              5xx-status, telt i serverens forespørselslogg
Mål:          99,9 % over et glidende vindu på 28 dager
Begrunnelse:  observert, rundet ned fra fire ukers data
Eier:         produktsjef sammen med utviklerne som har vakt
Gjennomgang:  månedlig til målet har satt seg, deretter kvartalsvis
```

Vinduet betyr like mye som prosenten. Workbook foretrekker et glidende vindu framfor en kalendermåned, fordi brukerne ikke glemmer et utfall den siste dagen i måneden bare fordi en ny begynner, og et helt antall uker, slik at hvert vindu har like mange helger; fire uker er dens valg til generell bruk. Det vinduet er 28 × 24 × 60 = 40 320 minutter, og 0,1 % av det er litt over 40 minutter med fullt utfall. For en måned på 30 dager gir de samme 99,9 % rom for 43,2 minutter, tallet i bokas tilgjengelighetstabell.

Tid er likevel den grovere måten å telle på. SRE-boka forklarer at Google måler tilgjengelighet som andelen vellykkede forespørsler, ikke som oppetid, fordi en tjeneste spredt over hele verden sjelden er helt oppe eller helt nede. Regnet i forespørsler lyder Workbooks eksempel slik: med 99,9 % gir 3 millioner forespørsler over fire uker rom for 3 000 feil, og et utfall som forårsaker 1 500 av dem, bruker halve budsjettet. Å telle forespørsler fanger også de delvise feilene oppetid ikke ser. Ved jevn trafikk koster 20 minutter der halvparten av alle forespørsler feiler, like mye som 10 minutter med fullt utfall, og det er omtrent en firedel av budsjettet for 28 dager.

I Prometheus, med en teller for forespørsler merket med status, er indikatoren over vinduet ett forhold, bygd etter Workbooks eget eksempel:

```
sum(rate(http_requests_total{job="checkout", status!~"5.."}[28d]))
  /
sum(rate(http_requests_total{job="checkout"}[28d]))
```

På dette nettstedet står siden om servicenivå på avtalesiden av streken. For et system selskapet drifter for en kunde, skrives responstiden ved en driftshendelse, timene som dekkes, tilgjengeligheten systemet skal holde, og når vedlikehold kan skje, inn i avtalen for akkurat det systemet. Det settes system for system, fordi en nattlig rapport og en betalingstjeneste trenger ulike svar. Den egne feilovervåkingen og sporingen som grunnleggeren bygde i sitt nåværende arbeid hos NextMariner, vises gjennom en rekke operatørvisninger, blant dem én for tjenestenivåmål. Og notatet om å selge den siste billetten nøyaktig én gang gjør en lagerinvariant til et slikt mål: kjør sjekken etter en tidsplan, vis den i en graf og varsle ved ethvert avvik, så null oversalg blir et mål med en alarm bak, ikke et håp.

Hvor det går galt

Målet er kopiert fra forrige måneds dashbord. Å ta det systemet klarer i dag, føles trygt, men SRE-boka advarer om at et mål som velges uten ettertanke, kan låse et team til heroisk innsats for å fortsette å nå det, på et system som ikke kan bli bedre uten å bygges om. Gå baklengs fra hva brukerne trenger, og skriv begrunnelsen ved siden av tallet: Workbook ber om resonnementet, og om tallene kommer fra eksperiment eller observasjon, også når de er helt ad hoc. Gå gjennom dokumentet månedlig i starten og kvartalsvis når det har satt seg.

Målet er 100 %, eller en rekke niere ingen har regnet på prisen av. Et system som alltid er tilgjengelig, eller skalerer uten grense, er dyrt å bygge og drive, og bedre enn brukerne vil merke. Start med et løst mål og stram det inn etter hvert som dere lærer systemet å kjenne; boka foretrekker det framfor et strengt mål som senere må lempes fordi det viste seg uoppnåelig.

Et gjennomsnitt står i stedet for det brukerne merker. I bokas eksempel tar en typisk forespørsel omkring 50 ms, mens 5 % av forespørslene tar tjue ganger så lang tid, og overvåking av bare gjennomsnittlig latens ville ikke vist noen endring gjennom dagen. Skriv latensmål som persentiler, eller som en andel forespørsler under en grense, for eksempel 300 ms, aldri som et gjennomsnitt.

Det finnes flere mål enn noen kan forsvare. Et dusin SLO-er per tjeneste vanner ut alle sammen. Bokas test er rett fram: kan et mål aldri brukes til å vinne en diskusjon om prioriteringer, er det trolig ikke verdt å ha. Behold de få som dekker det brukerne faktisk gjør.

Et mål som ikke nås, endrer ingenting. Workbook kaller dette en SLO uten tenner: etterlevelse blir bare én KPI til i en rapport i stedet for en grunn til å bestemme noe. Bli enige på forhånd om hva som skjer når budsjettet er brukt opp, og få produktsjefen, utviklerne og SRE-teamet til å skrive under. Eksempelet på en feilbudsjettpolicy i Workbook stanser alle endringer unntatt feilrettinger med høyeste prioritet og sikkerhetsrettinger til tjenesten er innenfor målet igjen, og sier rett ut at policyen ikke er en straff.

SLO-en er det samme tallet som SLA-en. Når det interne målet er likt kontrakten, er bruddet det første teamet hører om problemet. Boka anbefaler et strengere internt mål enn det som loves utad, som en buffer mot kroniske problemer. Den motsatte feilen gjør også skade: en tjeneste som går langt bedre enn målet, lærer brukerne å regne med det, så boka foreslår planlagt nedetid av og til, struping (throttling), eller et system som ikke er raskere under lav last.

Vinduet er lengre enn dataene. En SLO på 28 dager regnet ut på en Prometheus-server som beholder 15 dager med data, standardverdien når ingen lagringstid er satt, måler i det stille 15 dager og rapporterer det som fire uker. Sett lagringstiden lenger enn vinduet før du stoler på tallet:

```
prometheus --storage.tsdb.retention.time=35d
```

Les mer

- Service Level Objectives — kapittel 4 i Googles bok om Site Reliability Engineering: indikatorer, mål og avtaler, og hvordan man velger et mål.
- Implementing SLOs — kapittelet i SRE Workbook om SLI-er som forhold, feilbudsjetter, vinduer og hvordan målet skrives ned.
- Example Error Budget Policy — en fullstendig policy for en oppdiktet spilltjeneste: når utgivelser stanses, og hvem som avgjør en uenighet.
- Embracing Risk — hvorfor Google måler tilgjengelighet med vellykkede forespørsler i stedet for oppetid, og hva hvert steg i pålitelighet koster.
- Availability Table — hvor mye nedetid hvert tilgjengelighetsnivå tillater per år, kvartal, måned, uke, dag og time.

- Storage — hvordan Prometheus lagrer data på disk, og hvilke flagg for lagringstid som avgjør hvor langt tilbake et SLO-uttrykk kan se.

Ordliste

Velg ut fra deres faktiske rammer

Ta med utrullingsløpet, driftsrisikoene og dem som har vakt. Vi kan avgrense en endring med kontroller, en vei tilbake og tydelig ansvar.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er et tjenestenivåmål?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

welcome@engineer.company +45 50 10 61 84 København, Danmark

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/service-level-objective/