A THEORY and PRACTICE of SERVICE LEVEL OBJECTIVES

Jamie Wilkinson
SRECon Asia, June 2018
Martyrs On Film: learning to hate the #oncallselfie
by Alice Goldfuss
Published May 22, 2017 in Technology
Symptom Based Alerting

https://www.flickr.com/photos/chris-warren-photos/2220257496/ CC-BY-NC 2.0
Why does $\forall X \in \{\text{Ops}\}$ suck?

The cost of maintenance must scale sublinearly with the growth of the service size: e.g., queries, storage footprint, cores used, watts.

The cost of maintenance must scale sublinearly with the growth of the service.
Is this a symptom?
What makes this a symptom?

Engineering Tolerance

Availability “Tolerance”
SLAs, SLOs, SLIs

- **SLI → Indicator**: a measurement
  - distribution of response time over 10 minutes
  - response error ratios over 10 minutes

- **SLO → Objective**: a goal
  - 99.9th percentile response latency below 5ms
  - lower than 1% rate of errors

- **SLA → Agreement**: economic incentives
  - or we get paged
“As a mechanical engineer in an R&D lab I frequently ask myself, what is a reasonable tolerance to set on this part?”

https://engineerdog.com/2017/12/02/engineering-guidelines-for-selecting-mechanical-design-tolerances/
Does your service have an SLO?

The answer may surprise you!

Posted on May 31, 2018, at 1:53 p.m.

Jamie Wilkinson
Site Reliability Engineer
A symptom is anything that can be measured by the SLO.

A symptom-based alert is an alert when the SLO is in danger of being missed.
For availability SLAs we often talk about system uptime:

\[
\text{availability} = \frac{\text{uptime}}{(\text{uptime} + \text{downtime})}
\]

How do you measure uptime of a distributed system?
Another way to calculate this is with a \textit{request success rate}:

$$availability = \frac{\text{successful requests}}{\text{total requests}}$$

Defining SLOs in terms of request success rate makes it easier to measure an error budget.
```go
var responses = prometheus.NewCounterVec(
    prometheus.CounterOpts{
        Name: "responses",
        Help: "total errors served"},
    []string{"code", "user"})
...
responses.WithLabelValues(
    http.StatusText(400),
    GetUser(req)).Add(1)
```
1/qps sample density
record: error_ratio_by_user
expr: sum by (job, user)(
    rate(responses{code!~"200"}[10s]))
  / on (job, user)
  sum by (job, user)(rate(responses[10s]))

alert: ErrorRatioTooHigh
expr: error_ratio_by_user > 0.01
SLO burn

cumulative errors vs. time

- alerting window
- error rate threshold
- scaled error budget
Burn rate maths

Average QPS rate: 1000
SLO: 99% over 1 week
= 604,800,000 total queries
= 6,048,000 permissible errors
Take 1 hour moving average of errors
Page if error budget is going to be exhausted in less than 24 hours

= 6,048,000 errors consumed per day
= 70 err/s = 252,000 errors in 1 hour
Page if 15m rate over 70.
SLO Fast Burn

expr: \( \text{delta}(\text{errors}[1h]) > \left( \text{expected_events} \times \frac{\text{error_budget}}{\text{burn_period}} \right) \)

= 

expr: \( \text{delta}(\text{errors}[1h]) > \left( 1000 \ \text{qps} \times 7 \text{d} \times \frac{0.01}{24} \right) \)

= 

expr: \( \text{delta}(\text{errors}[1h]) > 70 \)
EDITORIALISE ABOUT OBSERVABILITY
“one of the most powerful context-sensitive incredibly adaptive anomaly-detecting and responding agents in the world”

-- John Allspaw, Monitorama 2013
1. Symptom-based alerts are good for your health
2. SLO is defined by you, customers, and system
3. SLO implies error budget, informs engineering tolerance
4. Page only on SLO risk, because that’s what matters