Unified Reporting of Service Reliability

Helen Zhang, Google

SREcon19 Asia/Pacific
2019-06-13

Google Cloud
We Have a Dream

Gain actionable insights from a unified view of service reliability
Agenda

● Problem
● Solution
● Success and Challenges
● Takeaways
Tools exist to visualize SLO compliance, error budget, but...
Problem

No “one-stop” tool exists to correlate SLO metrics to other service events to gain actionable insights:

- What launches or production rollouts caused a production outage, broke SLO compliance, and generated a Cloud support ticket?
- What actions can we take?
A Bonus Problem

Can we use ML to predict the probability of a service’s SLO violation?
We Have a Plan

Build a multidimensional “data cube”

- One cube = one entity
  (service/product/product group)
- Each dimension = one aspect of production data (e.g. SLO compliance, outage count, SRE pager load)
- See the correlated data for one entity? Query one cube!
Unified Reporting Architecture: 10,000 foot view

- **Request Flow**
  - Original data from various providers
  - Get Taxonomy ID
  - Postmortem
  - Production Event Logging
  - SLO reporting
  - Cloud Customer Support Case
  - SRE Pager data

- **Data Flow**
  - Launch
  - Dashboard display
  - Materialized tables
  - Datacube Engine (transform, join, materialize)

**Production Taxonomy**

**Google Cloud**
ML - Only a Start

- Used ML to predict SLO violations
  - Initial explorations didn’t go far
- Challenges
  - Predicting rare events is hard
  - Limited data quantity and quality. i.e. need more high quality data
- Not actively working on it, but would like to pursue it further in the future
Unified Reporting Design Overview

● Step 1: Production Taxonomy
  ○ A Unique ID for different entities: product, project, service, etc.
  ○ A different team did this work

● Step 2: Data Cube
  ○ Ingest and join different data sources using Production Taxonomy ID
  ○ I and my team worked on this part
Life of a Dataset

- **Source tables**:
  - Extract fields from each source table to save new base table
  - **Export data**

- **Base tables**:
  - Aggregate, join, etc. and materialize derived tables
  - **Application Reporting Script**

- **Derived tables**:
  - **dashboards**
  - **Export data**

- **Data provider**
- **Production Taxonomy Tables**
- **Export data**

---

Google Cloud
Design Principles

- Use the simplest infrastructure
- Focus on data
Data Modeling

- Entity Relationship Database
- Star Schema
Entity Relationship Database Model

- Model Product Area, Product Group, Product, Project, Owner, API Service name entities

- Model the following relationships among all the entities:
  - Service API $\leftrightarrow$ group [n:1]
  - mdb $\leftrightarrow$ project [n:1]
  - project $\leftrightarrow$ product [n:1]
  - product $\leftrightarrow$ product group [n:1]
  - product $\leftrightarrow$ product area [n:1]
Star Schema Model

- Most widely used for data warehouses
- Consists of one or more fact tables referencing any number of dimension tables.
ERD Model is the Best Option

A natural fit for the existing schema of all data sources

Star Schema doesn’t work well for M:M relationships, common in our use cases, e.g.

- 1 outage is associated to SLO violations of multiple services
- 1 service’s SLO violation can cause multiple outages
Insights Needed

- Are my service’s SLIs/SLOs aligned with customer happiness?
- How often do customers report outages before our monitoring/alerting system detects them?
## Insight and Action: Fix ill-defined SLI/SLO

<table>
<thead>
<tr>
<th>Service</th>
<th>Aggregation Period</th>
<th>Correlation</th>
<th>Insight</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>SLO Compliance Met?</td>
<td>Major Outage Happened?</td>
<td>SLO reflect User Happiness?</td>
</tr>
<tr>
<td>A</td>
<td>Quarterly</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>B</td>
<td>Quarterly</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>C</td>
<td>Quarterly</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>D</td>
<td>Quarterly</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
Limitations

● Impact is limited due to outstanding data quality issues

● A cross-team technical program (not run by our team) is created to drive making service SLIs/SLOs reflect customer experience
## Insight and Action: Fix monitoring/alerting gaps

<table>
<thead>
<tr>
<th>Production Outage</th>
<th>Correlation</th>
<th>Insight</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outage 1</td>
<td>No</td>
<td>No</td>
<td>Nothing</td>
</tr>
<tr>
<td>Outage 2</td>
<td>Yes</td>
<td>Yes</td>
<td>Fix Monitoring/Alerting</td>
</tr>
<tr>
<td>Outage 3</td>
<td>Yes</td>
<td>Yes</td>
<td>Fix Monitoring/Alerting</td>
</tr>
</tbody>
</table>
Challenges

Outstanding quality issues unresolved

- Limited quantity, incomplete, and inaccurate source data
- Correlation inaccuracy due to the lack of a common identifiers across data sources
Takeaways

- Establish a solid process to enforce clean data from the source

- Focus
  - Standardize and automate
  - Have a vision for the future, but don’t be disappointed if the first attempt doesn’t succeed