Pattern: Application metrics

Context

You have applied the Microservice architecture pattern (../microservices.html).

Problem

How to understand the behavior of an application and troubleshoot problems?

Forces

- Any solution should have minimal runtime overhead

Solution

Instrument a service to gather statistics about individual operations. Aggregate metrics in centralized metrics service, which provides reporting and alerting. There are two models for aggregating metrics:

- push - the service pushes metrics to the metrics service
- pull - the metrics services pulls metrics from the service

Examples

- Instrumentation libraries:
  - Coda Hale/Yammer Java Metrics Library (http://metrics.dropwizard.io/3.1.0/)
  - Prometheus client libraries (https://prometheus.io/docs/instrumenting/clientlibs/)
- Metrics aggregation services
  - Prometheus (https://prometheus.io/docs/introduction/overview/)
  - AWS Cloud Watch (https://aws.amazon.com/cloudwatch/)

Resulting context

This pattern has the following benefits:

- It provides deep insight into application behavior

This pattern has the following drawbacks:

- Metrics code is intertwined with business logic making it more complicated

This pattern has the following issues:

- Aggregating metrics can require significant infrastructure