Observability

Nuxeo Core 2019

About

Observability is a hot topic[1] these days, traditional monitoring using dashboard is hard to scale and limited to simple cases like detecting already known failure.
To answer the question what happened on complex system, where a single request is executed using external databases, services that handles retries policies and asynchronous processing, we need more than metrics we need distributed tracing.

Tracing and stats

Tracing is about adding span around block of code, so we can trace path execution with elapsed time, span can be annotated with labels to give execution context like user name, NXQL query ...
To do this the Nuxeo code need to be instrumented, some or our third part services like Mongo/jdbc/redis can be dynamically instrumented.
The trace information are collected and rendered in a tracer, there are many options for tracers, from open source Jaeger or Zipkin to Datadog APM, Google Stackdrive or Azure monitor...

Tracing does not replace the need of monitoring stats (metrics). Today we use Dropwizard metrics (previously known as Codahale metrics) that can publish metrics to different stats backend: Graphite, Datadog, Statsd.
But we are missing the trendy prometheus solution which is commonly bundled with Kubernetes/Openshift. This solution scales better by relying on a poll mechanism and provides precious health checks and alerting.

Possible solutions

There are 2 options to support tracing on our stack, opentracing and opencensus both requires to instrument our code and to choose an external tracer to collect and display the traces.

Opentracing

This is a vendor-neutral, auto proclaimed open standard for distributed tracing, it provides just an interface, a tracer implementation need to be chosen.

[1] https://thenewstack.io/why-you-cant-afford-to-ignore-distributed-tracing-for-observability

OpenCensus

This is a library open sourced by Google and now actively supported by Microsoft.
It provides a coherent implementation and looks more mature than what the misc implementation from open tracing community.
One advantage is that it handles both tracing and stats and offer a dropwizard metrics exporter to expose stats to prometheus.

At this time there is no exporter to Datadog from java (only for Go).

Tracer

Jaeger is the Uber tracer parts of the CNCF with Kubernetes support

Zipkin is the twitter tracer not distributed solution, does not support python langage.

Only Google provides a solution for stats and tracing: stackdriver

Proposed solution

Use OpenCensus library.
Expose our Dropwizard metrics to Prometheus using OpenCensus exporter.
Instrument our code to add tracing.
Support Jaeger tracer.

Steps

Stats

1. Exposes our metrics through opencensus prometheus
2. Explore prometheus supported services
3. Provides a first Grafana dashboard for prometheus

Tracing

4. Provides a docker compose with Nuxeo and Jaeger
5. Adds tracing to Nuxeo
    a. On http request
    b. On transaction
    c. On fire events
    d. On work
    e. On computation
    f. On bulk command
    g. On audit backend
    h. On repository

        i. On binary store
        j. On kv store
6. Adds tracing around services
    a. On JDBC
    b. On MongoDB
    c. On Redis
    d. On Elastic REST client
    e. On ldap

7. Removes old SequenceTracer usage

Estimation

Stats and Tracing tasks are not tied, within 3w we should have something workable.

Stats ~1w

Exposing to prometheus is easy -> 1d
Building Grafana dashboard requires prometheus query language knowledge -> 1w

Tracing ~2w

The longest part is to make sure we have trace where it matters without impacting performances.
This will requires to pass a tracing context on some async part (in stream records, works
But even with few tracing it is still very helpful, so this can be seen to a long running task like adding missing metrics.
Basic tracing on Nuxeo: 1w
Basic tracing on Services: 3d

Existing tickets

- NXP-26416 Prometheus support for stream latency

Risk

These frameworks are still in their infancy.
Using opencensus prometheus exposition of dropwizard metrics might not be able to leverage all prometheus feature like metric description.
Datadog APM does not yet support opencensus tracing from Java (only from Go at this time).