Tech Talk Live #130
Open Source Monitoring For Alfresco

Roxana Angheluta
Thijs Lemmens
Daan Kerkhofs
9 June 2021

Alfresco
simply a better way

Agenda

Open Source Monitoring for Alfresco

2021-06-09

- Xenit
- Speakers
- Motivation
- Architecture
- Platform telemetry
- Solr telemetry
- Resources, etc?
- Q&A

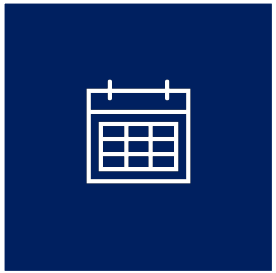
Xenit Solutions (http://www.xenit.eu)

xenit
return on content

Company Overview

2007

Founded



13+

Years of IIM Experience

300M+

Documents managed

#1

Alfresco Integrator in Belgium

40+

Global Customers

Speaker introduction

Daan Kerkhofs
@daan_kerkhofs

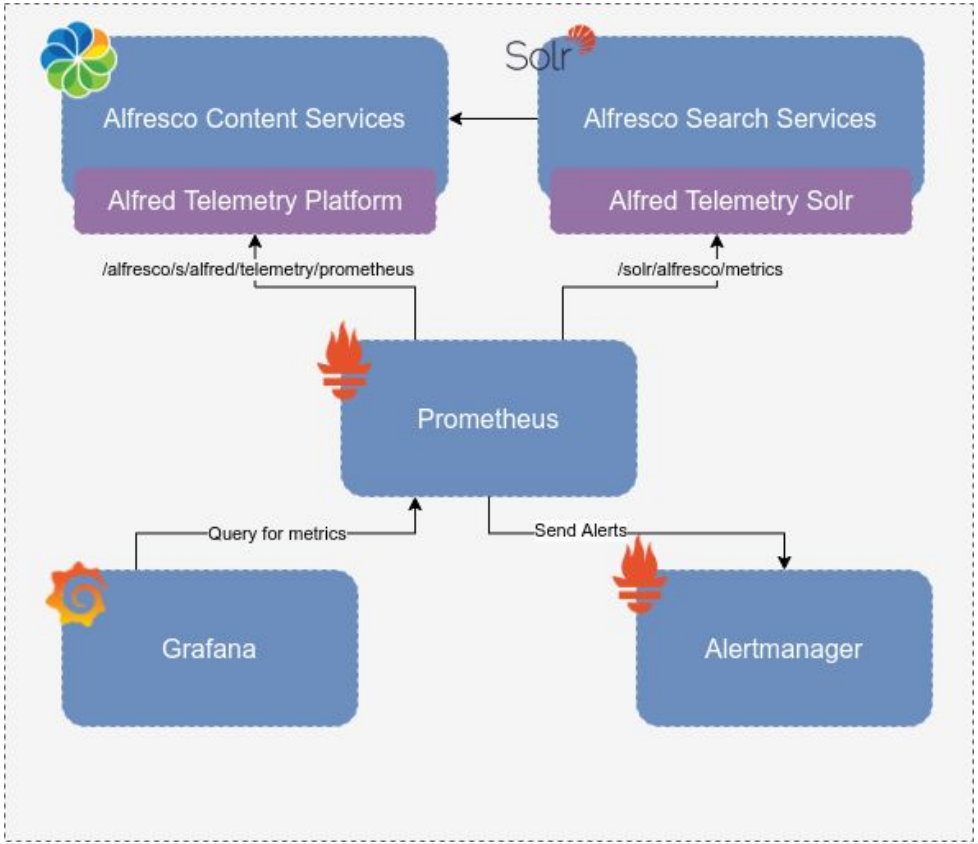
Roxana Angheluta
@RoxanaAnghelut2

Thijs Lemmens
@thijslemmens

Motivation

- Hosted Alfresco: SAAS
- Managed Services:
  - On premise
  - Cloud (AWS, Azure)
- Technical Audit of Alfresco environment:
  - Problem detection
  - Suggestions for improvement

Solution architecture



Github project

https://github.com/xenit-eu/alfresco-ttl-monitoring

Alfred Telemetry Platform

Integrates Micrometer.io with ACS

- Micrometer.io
  - Application metrics facade that supports numerous monitoring systems
  - Provides metrics API, decoupled from monitoring system
  - Provides a huge list of plugable monitoring systems
    - Prometheus, Graphite, JMX, Elastic, Influx, ...
- Alfred Telemetry Platform
  - ACS extension (AMP / SM)
  - Provides out of the box metrics
  - Provides necessities to expose custom metrics
    - meterRegistry bean / Metrics.globalRegistry()
    - Automatically register “MeterBinder” beans
    - ...
  - “Autoconfigure” supported monitoring systems

AEE Micrometer integration

Alfred Telemetry Platform

- Alfresco Enterprise Edition (>= 6.1) includes Micrometer integration

	Alfred Telemetry Platform	OOTBox Micrometer Integration
Open Source	✅	❌
ACSCE Support	✅	❌
Supported Alfresco version	5.2, 6.0, 6.1, 6.2, 7.0	>= 6.1
Supported monitoring systems	∞	Prometheus
Exposing custom metrics	✅	In theory: ✅, in practice: ❓

Demo - Out of the box metrics

Alfred Telemetry Platform

- JVM
- Process
- System
- JDBC
- Cache
- Alfresco users
- Alfresco license
- …

Exposing custom metrics

Alfred Telemetry Platform

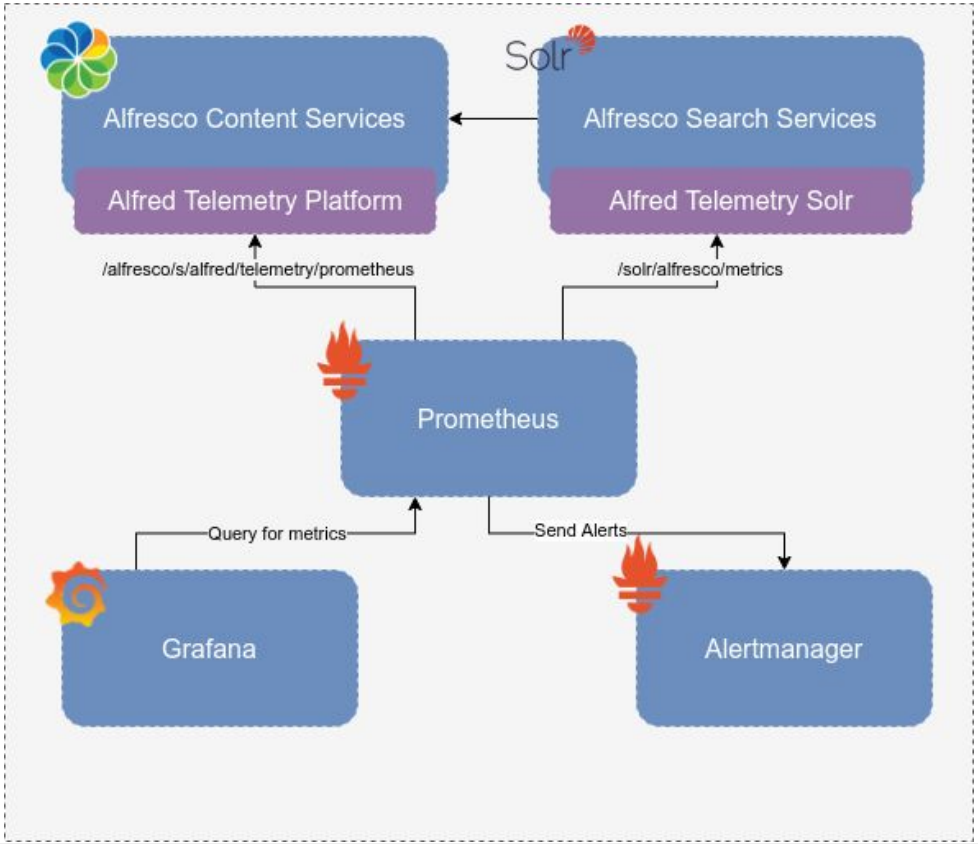
```
public class SampleBean {

    private final Counter counter;

    public SampleBean(MeterRegistry registry) {
        this.counter = registry.counter("received.messages");
    }

    public void handleMessage(String message) {
        this.counter.increment();
        // handle message implementation
    }

}
```

Demo - exposing custom metrics

Alfred Telemetry Platform

- Alfresco Health Processor
  - Looping over nodes in Alfresco
  - Executes OOTBox or custom “health” checks
  - Demo: basic content validation
- AlfredTelemetryHealthReporter
  - Exposes metrics via Alfred Telemetry
  - GitHub

Solution architecture

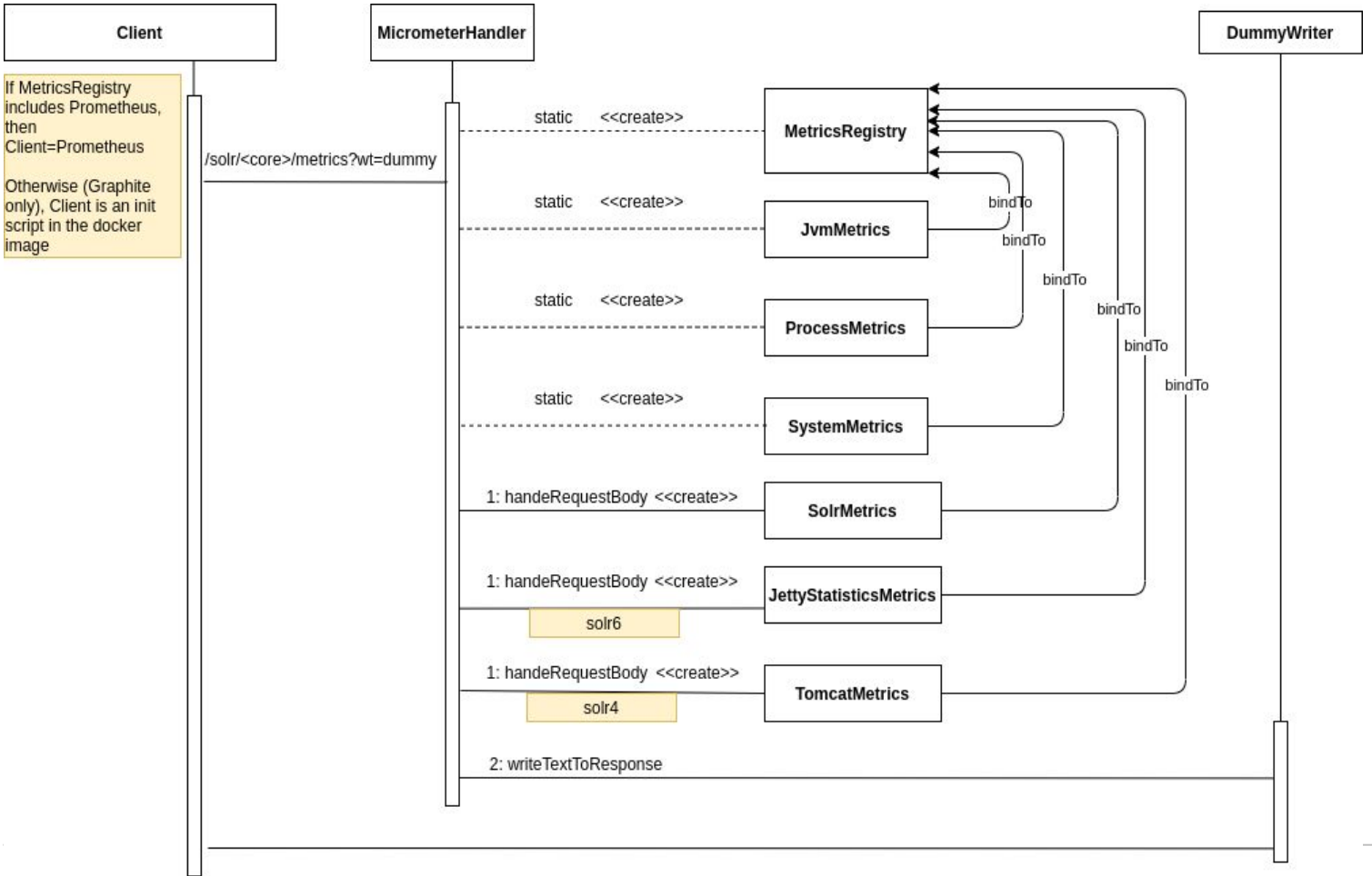


Alfred telemetry solr - start point

- Exporters = agents attached to Java process
    - Solr-prometheus-exporter
    - Jmx-exporter
- Solr metrics available from Solr 6.6 (no micrometer)

- Unified system, same names for metrics, reuse of dashboards and alerts => alfred-telemetry-solr

Alfred telemetry solr - architecture



Alfred telemetry solr - development, packaging

- No Spring
- Registries added via code: Prometheus, Graphite

- Separate jars for
  - Solr4 (Java 8)
  - Solr6 (Java 11)
    - Change of methods for FTS metrics starting with Alfresco Search Services 2.0.0, with the drop of solr contentstore
      - addFTSStatusCounts
      - addContentOutdatedAndUpdatedCounts

Alfred telemetry solr - customizations required

1. Handler + writer definition in solrconfig.xml

   No easy extension points for solr configuration (trick: use solrconfig_insight.xml for solr6)

```
<xi:include href="onPurposeNonExistingFile.xml" xmlns:xi="http://www.w3.org/2001/XInclude">
  <xi:fallback>
    <requestHandler name="/metrics" class="eu.xenit.alfred.telemetry.solr.handler.MicrometerHandler"/>
    <queryResponseWriter name="dummy" class="eu.xenit.alfred.telemetry.solr.writer.DummyResponseWriter"/>
  </xi:fallback>
</xi:include>
```

2. Jetty (solr6)

   StatisticsHandler (insert handler + prepend server class - to make server classes visible to webapp)

```
<Call name="insertHandler">
 <Arg>
  <New id="StatsHandler" class="org.eclipse.jetty.server.handler.StatisticsHandler" />
 </Arg>
</Call>
```

3. Init script for graphite, to call the handler

Demo

Resources and Links

Links

- https://github.com/xenit-eu/alfresco-ttl-monitoring
- Micrometer.io
  - Documentation
- Alfresco Health Processor
- Prometheus
  - Documentation
- Grafana
- Alfresco Docker Gradle Plugins
- Xenit Alfresco Docker images
  - hub.xenit.eu/public/alfresco-solr4-xenit
  - hub.xenit.eu/public/alfresco-solr6-xenit

Summary

One

Integrated Alfresco monitoring solution

Four

Flexible to expand with more metrics

Two

Easy to install

Five

Using modern tools

Three

Flexible to configure

Thank you

Questions???