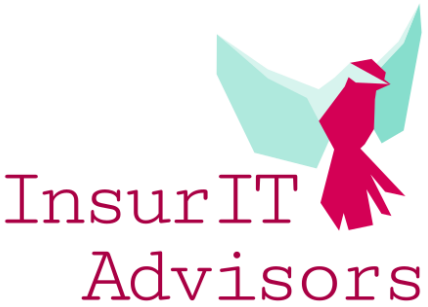
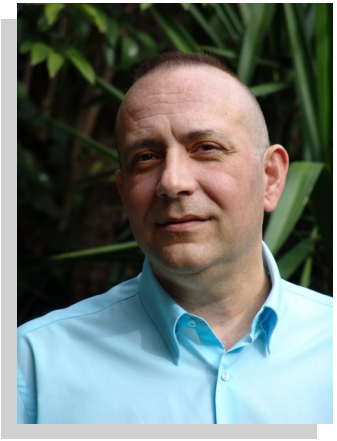


Vasilis Stergioulis @vas_stergioulis

Business metrics visualization with Grafana and Prometheus

2017-07-05 Docker Athens meetup



linkedin.com/in/vassterg
github.com/vassterg
gitlab.com/vassterg
facebook.com/vstergioulis
vassterg@insuritadvisors.com

Vasilis Stergioulis, @vas_stergioulis
Founder / Software Engineer at InsurIT Advisors

A system architect, senior software engineer that really enjoys writing code, with many years of expertise in the private insurance sector, deep knowledge of the private insurance system, intermediaries and distribution networks and enterprise-level challenges.

With experience in the operations of insurance and reinsurance companies operating in the European Union and the European Economic Area under the directive of Solvency II framework.

Participates in the Java Community Process (JCP) Program as Associate Member and focusing on Java EE applications.

Agenda

- Big datasets and analytics
- Real time business intelligence
- Time series metrics, monitoring
- Prometheus, Grafana
- Data instrumentation and collector agent
- Docker use

Big datasets and analytics

- OLTP, OLAP, BI Analytics, Replicas etc
- Big data volumes, hard to digest
- Need data scientists, not for business users
- Time constraints limit real time performance
- Data usually need transformations

Real time business intelligence

- Need a way to look inside systems
- Measure:
  - critical data counters and execution times
  - business values and components
  - rates of business changes and events
- In real time with minimal impact on inspected systems

Time series metrics, monitoring

- Instrument in business boundaries
- Profile execution of services
- Gather data point metrics
- Data points as rows of key-values in one table
- Collect table data as snapshot of system state
- Convert snapshot to time series
- Process and display data

Prometheus

- Time series metrics, monitoring
- Multidimensional (with labels)
- Self contained, self tuned
- Graphs, rules, alerts
- Grafana support

Grafana

- Beautiful analytics and monitoring
- Open source platform
- Multiple datasources, panels, dashboards
- Query, visualize, alert on and understand metrics

Instrumentation code

- A table to hold current values of metrics
- Simple operations, increment and set
- Autonomous transactions
- Minimum overhead
- Counters, gauges and summaries
- Measure values, time and errors

Table definition

```
CREATE TABLE instrumentation_metrics
(
    im_id        NUMBER PRIMARY KEY,
    im_metric    VARCHAR2 (200 BYTE) NOT NULL,
    im_value     NUMBER DEFAULT 0,
    im_type      VARCHAR2 (250 BYTE),
    im_help      VARCHAR2 (350 BYTE)
);
```

Example code in plsql

```
procedure some_proc(some_param varchar2) is
  start timestamp := now();
begin
  // … do some business things
  metrics.incr_metric('someproc_time_total', timer(start, systimestamp));
  metrics.incr_metric('someproc_total');
  metrics.incr_metric('otherservice_invocations');
  exception
    when others then
      metrics.incr_metric('someproc_time_total', timer(start, systimestamp));
      metrics.incr_metric('someproc_failures');
      metrics.incr_metric('someproc_total');
      // … exception handling code
end;
```

Collector agent

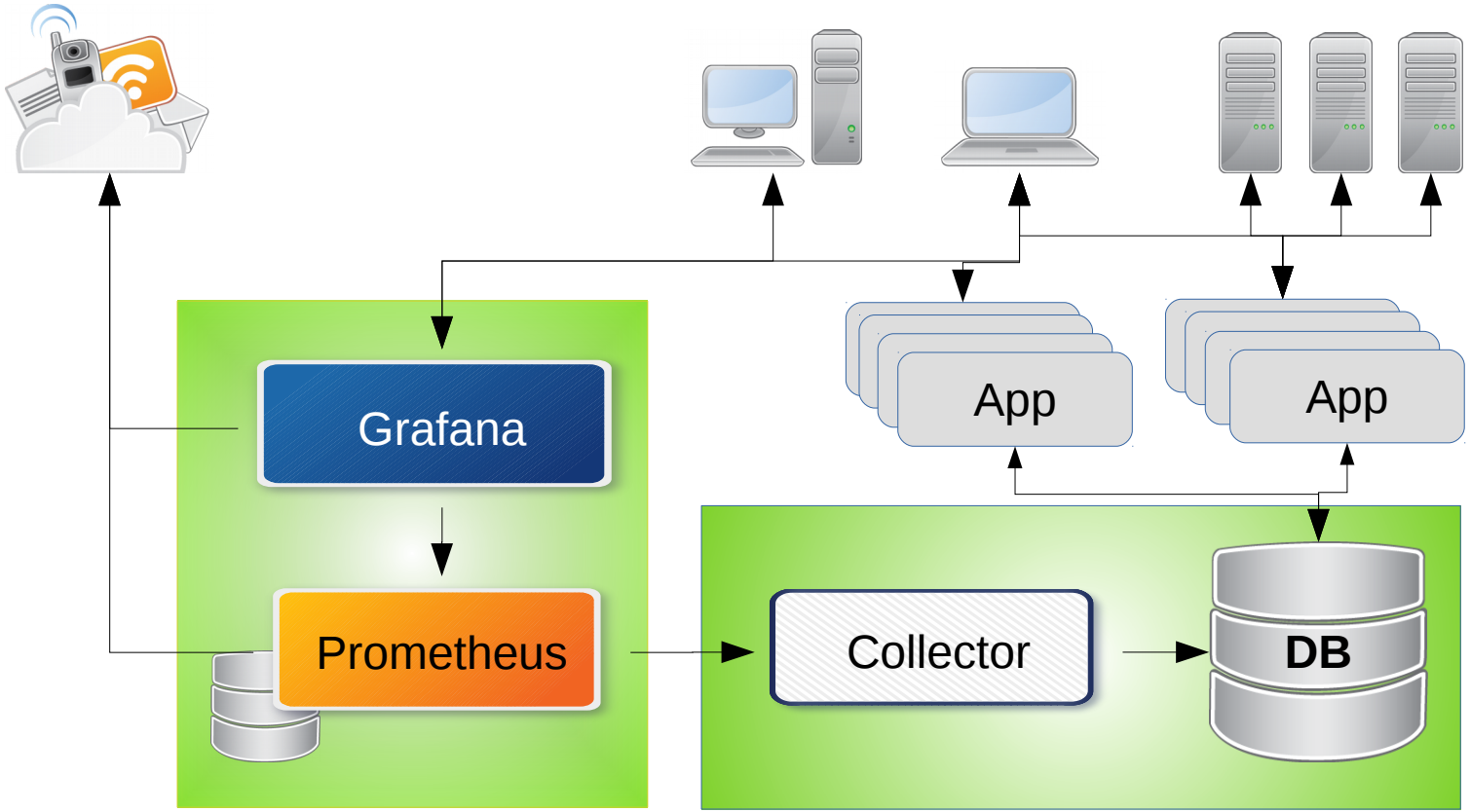
- Simple REST API
- Load and transform metrics
- Exposes as TEXT or JSON via GET
- Async with pooled db connections
- Very simple service, typical 4 KB WAR file
- One service on one application server
- Deployed as autonomous container

Collector agent (cont.)

- No uber jar creation
- UltraThin war to docker image
- Pushed to remote registry in seconds
- Fast build automation pipelines

  mvn clean package && docker build … && docker push

Schematic topology



Docker use

- Every service layer is built and tested in each own container as a component
- Docker-compose is used to bundle components into service(s)
- Prometheus and Grafana can be in separate data centers
- Collector agent can be near databases exposing only their interface to outside world and protecting, through pooling, the access to internal database

References

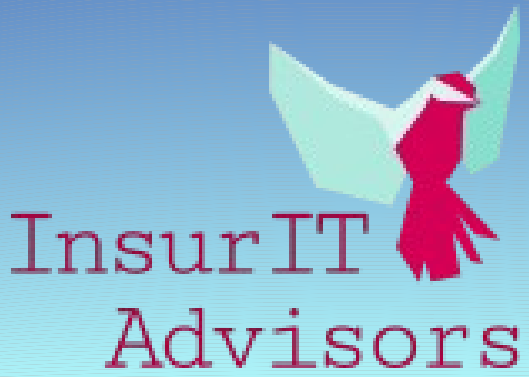
Docker registry for images and source files
https://hub.docker.com/u/insurit/ and https://github.com/insuritadvisors/dockerdom

Source code for collector agent and database tooling
https://github.com/insuritadvisors/sql_exporter

Prometheus - Monitoring system and time series database
https://prometheus.io/ and https://github.com/prometheus

Grafana - The open platform for analytics and monitoring
https://grafana.com/ and https://github.com/grafana/grafana

Java EE - Java Enterprise Edition
http://oracle.com/javaee and https://github.com/javaee and https://jcp.org



Thank you!
Any questions?