innoQ

Don’t Fly Blind

Logging and Metrics in Microservice Architectures

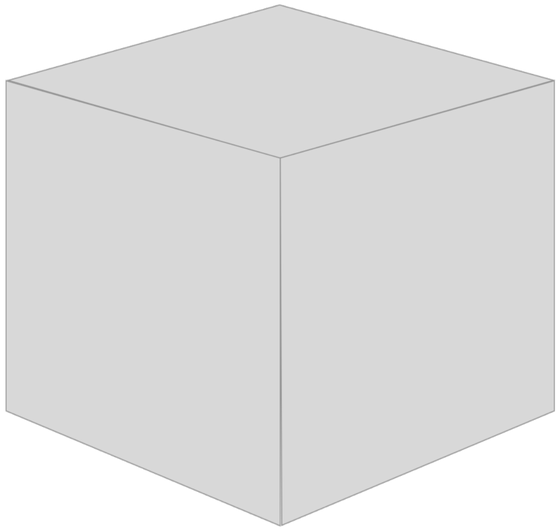
Tammo van Lessen | tammo.vanlessen@innoq.com

Alexander Heusingfeld | alexander.heusingfeld@innoq.com

The Talk Today

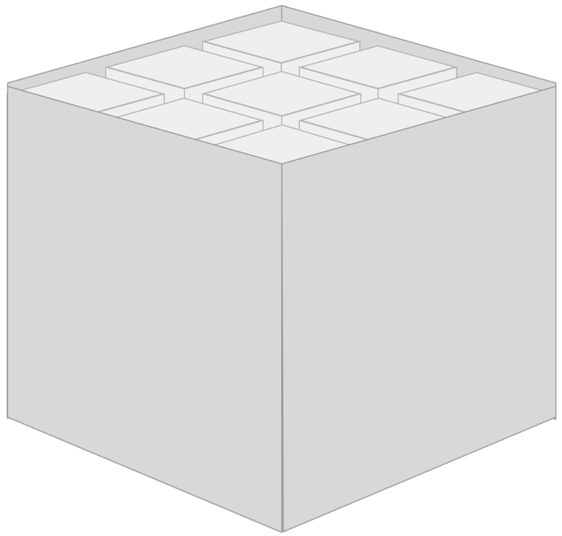
- Motivation
- Distributed Logging
- Distributed Metrics
- Conclusions

Breaking the monolith



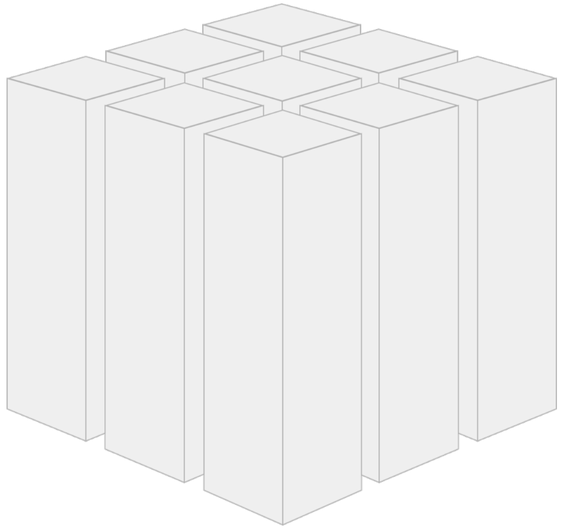
If you review a
monolithic application ...

© innoQ/Roman Stranghöner



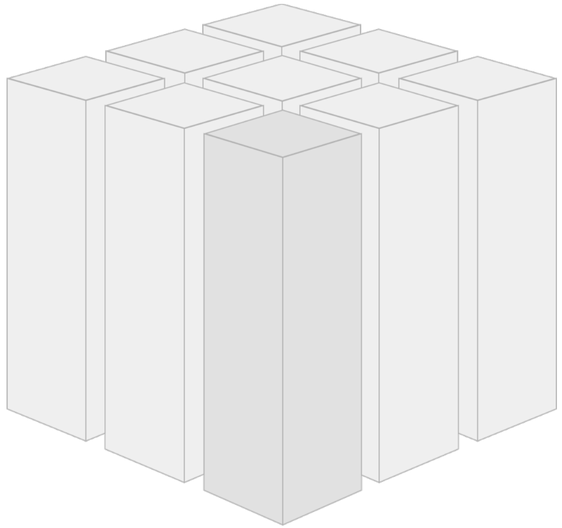
...and look into the black box...

© innoQ/Roman Stranghöner



...you'll find it consists of multiple Bounded Contexts.

© innoQ/Roman Stranghöner



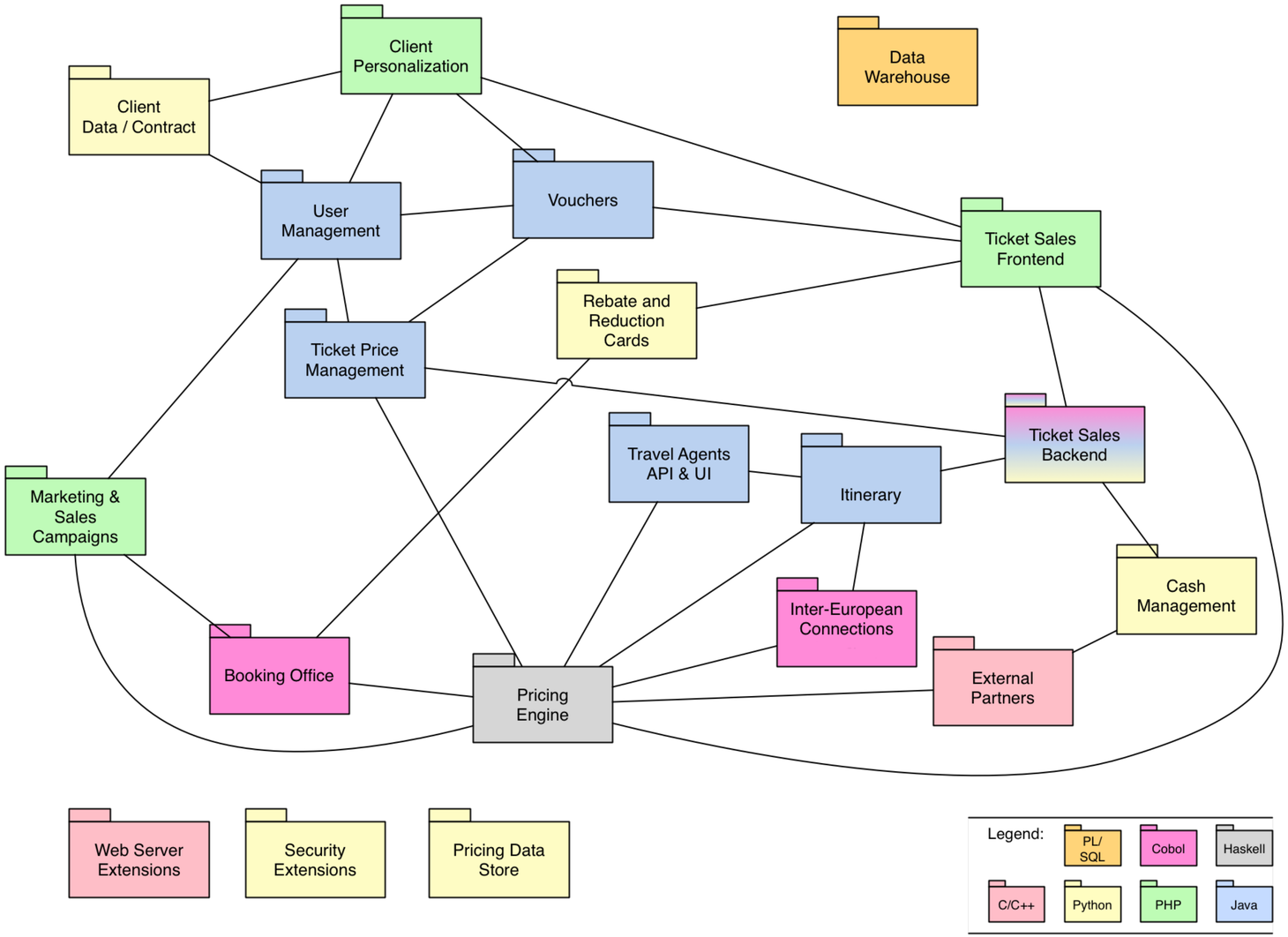
If you’re able to treat every Bounded Context as a **separately deployable, independent** component…

© innoQ/Roman Stranghöner

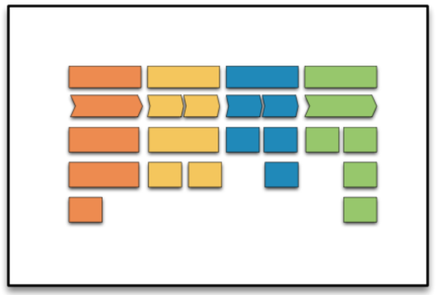
... you'll have a self-contained system - which can lead to a **microservice architecture**

Introduction to self-contained systems: https://www.innoq.com/de/links/self-contained-systems-infodeck/

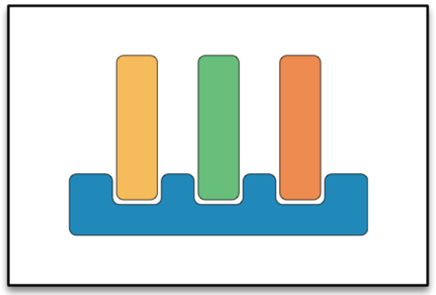
A Broken Monolith



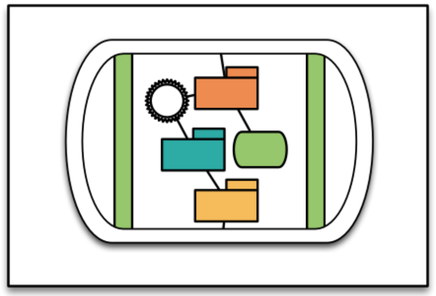
Architectural Decisions



- Domain Architecture



- Macro Architecture



- Micro Architecture

Logging in a Distributed Environment

Requirements

- **Apply a well-thought logging concept**
- Aggregate logs in different formats from different systems
- Search & Correlate
- Visualize & Drill-down
- Alerting

Use Thread Contexts / MDCs

```
ThreadContext.put("loginId", login);
logger.error("Something bad happened!");
ThreadContext.clear();
```

+ Layout:

```
%-5p: [%X{loginId}] %m%n
```

Log:

```
ERROR: [John Doe] Something bad happened!
```

Use Thread Contexts / MDCs

```
ThreadContext.put("loginId", login);
logger.error("Something bad happened!");
ThreadContext.clear();
```

+ JSON Layout

Log:

```
{
    "@version" => "1",
  "@timestamp" => "2014-04-29T14:21:14.988-07:00",
      "logger" => "com.example.LogStashExampleTest",
       "level" => "ERROR",
      "thread" => "Test worker",
     "message" => "Something bad happened!",
  "Properties" => {
          "loginId" => "John Doe"
  }
}
```

Define QoS for Log Messages

- Log messages may have different QoS
- Use Markers and Filters to enable fine-grained routing of messages to dedicated appenders
- Use Filters and Lookups to dynamically configure logging

https://www.innoq.com/en/blog/per-request-debugging-with-log4j2/

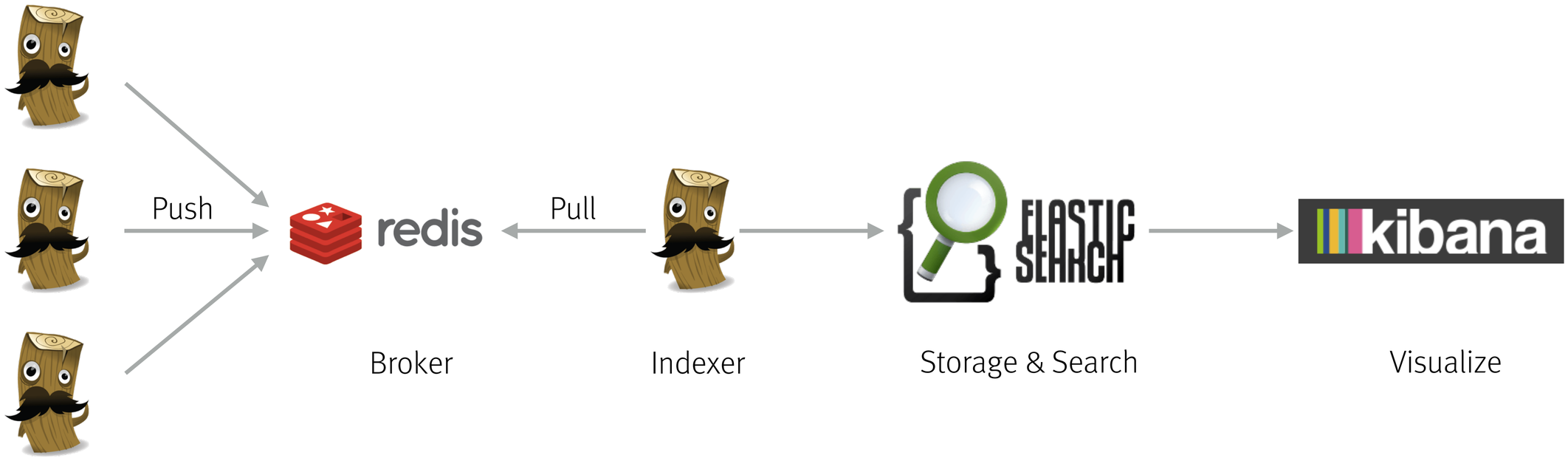
Requirements

- Apply a well-thought logging concept
- **Aggregate logs in different formats from different systems**
- **Search & Correlate**
- Visualize & Drill-down
- Alerting

Logstash Architecture

inputs	codecs	filters	outputs
collectd	cloudtrail	advisor	boundary
drupal_dblog	collectd	alter	circonus
elasticsearch	compress_spooler	anonymize	cloudwatch
eventlog	dots	checksum	csv
exec	edn	cidr	datadog
file	edn_lines	cipher	datadog_metrics
ganglia	fluent	clone	elasticsearch
gelf	graphite	collate	elasticsearch_http
gemfire	json	csv	elasticsearch_river
generator	json_lines	date	email
graphite	json_spooler	dns	exec
heroku	line	drop	file
imap	msgpack	elapsed	ganglia
invalid_input	multiline	elasticsearch	gelf
irc	netflow	environment	gemfire

Distributed Logstash Setup



Shipper /
Logstash Forwarder

https://www.elastic.co/products/logstash

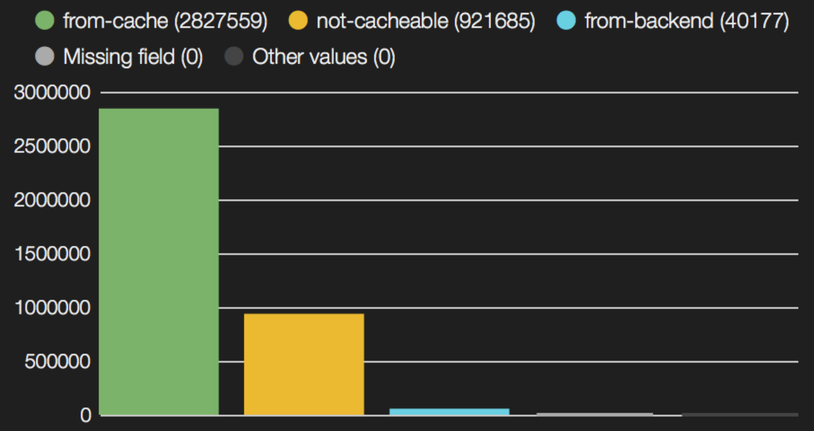
Requirements

- Apply a well-thought logging concept
- Aggregate logs in different formats from different systems
- Search & Correlate
- **Visualize & Drill-down**
- Alerting

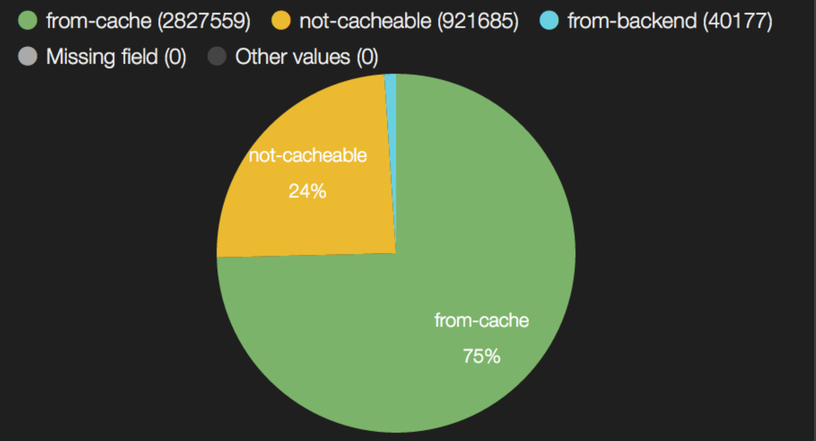
X-VGA-CACHEABILITY COUNT

Term	Count	Action
from-cache	2826739	
not-cacheable	921546	
from-backend	40162	
Missing field	0	
Other values	0	

BAR CHART



PIE CHART



ALL EVENTS

Fields

All (223) / **Current (26)**

Type to filter...

- @timestamp
- @version
- _id
- _index
- _type
- application
- environment
- ga-env
- host
- http_status
- instance
- logstash_timestamp
- market
- message
- method
- path

0 to **100** of 500 available for paging

@timestamp	region	x-request-id	x-vga-cacheability	tomcat_req_duration	tomcat_response_size
2015-10-27T09:55:53.096-07:00	ap-southeast-2	gaedge:05a7f23d-662e-4044-9ffd-52e86147c4d1	from-cache	9	49898
2015-10-27T09:55:53.083-07:00	ap-southeast-2	gaedge:58ea2601-cbba-4168-8008-988e79966eba	from-cache	10	137
2015-10-27T09:55:53.080-07:00	ap-southeast-2	gaedge:6986a65a-1100-41f2-8ca8-b838d1776102	from-backend	2036	248485
2015-10-27T09:55:53.069-07:00	ap-southeast-2	gaedge:ddc25ab0-e745-409e-8e31-cc0208f9aef8	not-cacheable	1563	10521
2015-10-27T09:55:53.033-07:00	ap-southeast-2	gaedge:0ed3f29b-bf84-438d-96bf-14495f1a6a20	not-cacheable	3105	3154
2015-10-27T09:55:53.006-07:00	ap-southeast-2	gaedge:0bdf0761-8e30-47ac-b354-20b861815bf4	from-cache	2	6085
2015-10-27T09:55:52.883-07:00	ap-southeast-2	gaedge:ee310b01-adff-4719-a792-716be657bf30	not-cacheable	2460	3154
2015-10-27T09:55:52.862-07:00	ap-southeast-2	gaedge:db4e829f-c942-4bbc-ae5a-f0cb4096bd3a	from-cache	6	42735
2015-10-27T09:55:52.760-07:00	ap-southeast-2	gaedge:b7cf5442-ae64-448a-a529-5db856054f2b	from-cache	6	24070
2015-10-27T09:55:52.750-07:00	ap-southeast-2	gaedge:196d1f48-a635-4b1e-9408-681fc6b400fb	not-cacheable	2325	3018
2015-10-27T09:55:52.742-07:00	ap-southeast-2	gaedge:435baaf4-62c6-4c72-b77b-8644528529d1	from-cache	4	137

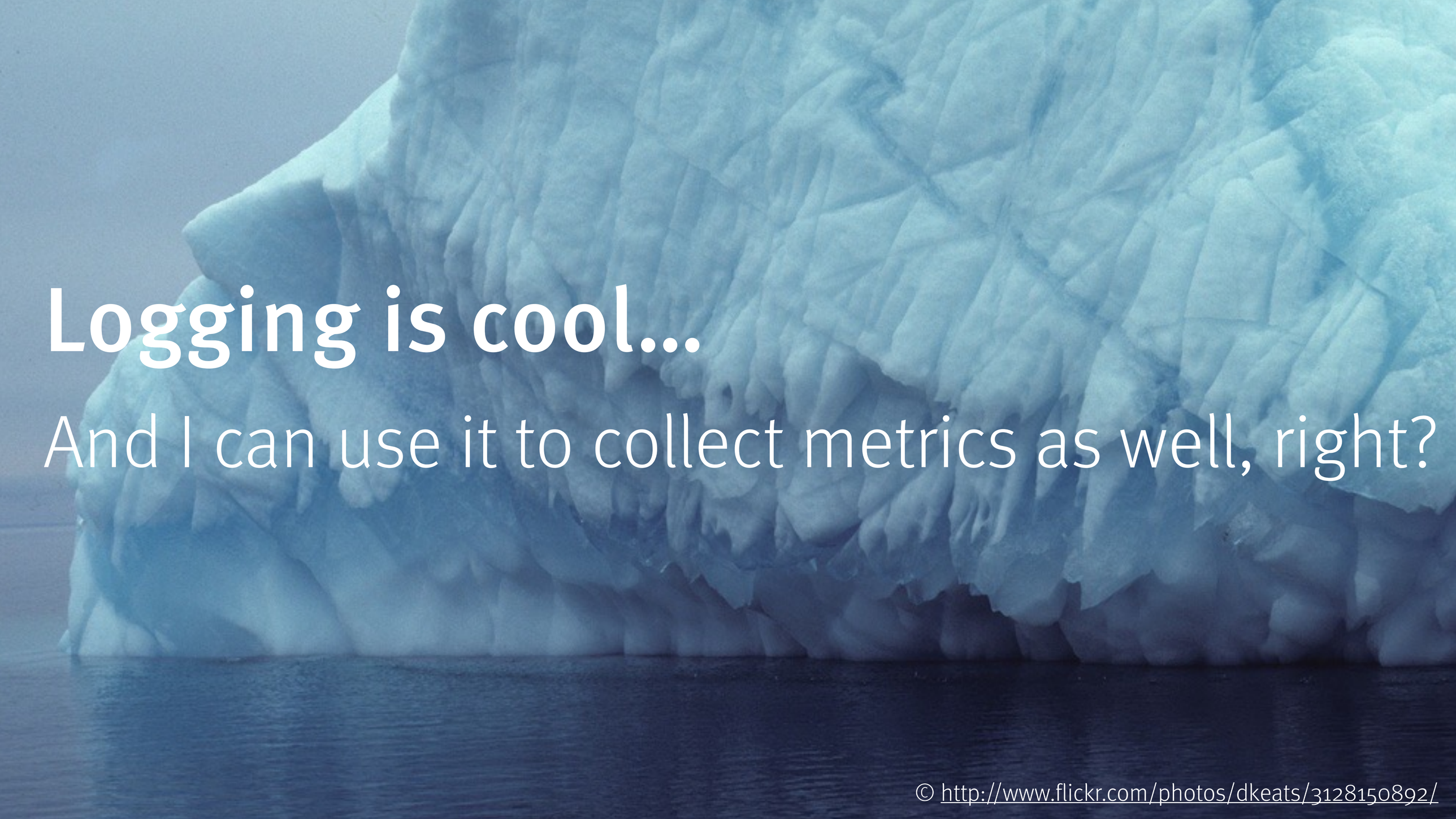
Requirements

- Apply a well-thought logging concept
- Aggregate logs in different formats from different systems
- Search & Correlate
- Visualize & Drill-down
- **Alerting**

Filter Log Stream For Alerts

```
input {
  …
}
filter {
  if [message] =~ /.*(CRITICAL|FATAL|ERROR|EXCEPTION).*/ {
    mutate { add_tag => "alarm" }
  }

  if [message] =~ /.*(?i)ignoreme.*/ {
    mutate { remove_tag => "alarm" }
  }
}
output {
  if [type] == "production" {
    if "alarm" in [tags] {
      pagerduty {
        description => "%{host} - %{log_level}: %{log_message}"
        details => {
          "timestamp" => "%{@timestamp}"
          "host" => "%{host}"
          "log_level" => "%{log_level}"
          "message" => "%{log_message}"
          "path" => "%{path}"
        }
      …
      }
    }
  }
}
```



Logging is cool...

And I can use it to collect metrics as well, right?

© http://www.flickr.com/photos/dkeats/3128150892/

© http://www.flickr.com/photos/dkeats/3128150892/

Metrics

Kinds of Metrics

Kinds of Metrics

- Business Metrics

Kinds of Metrics

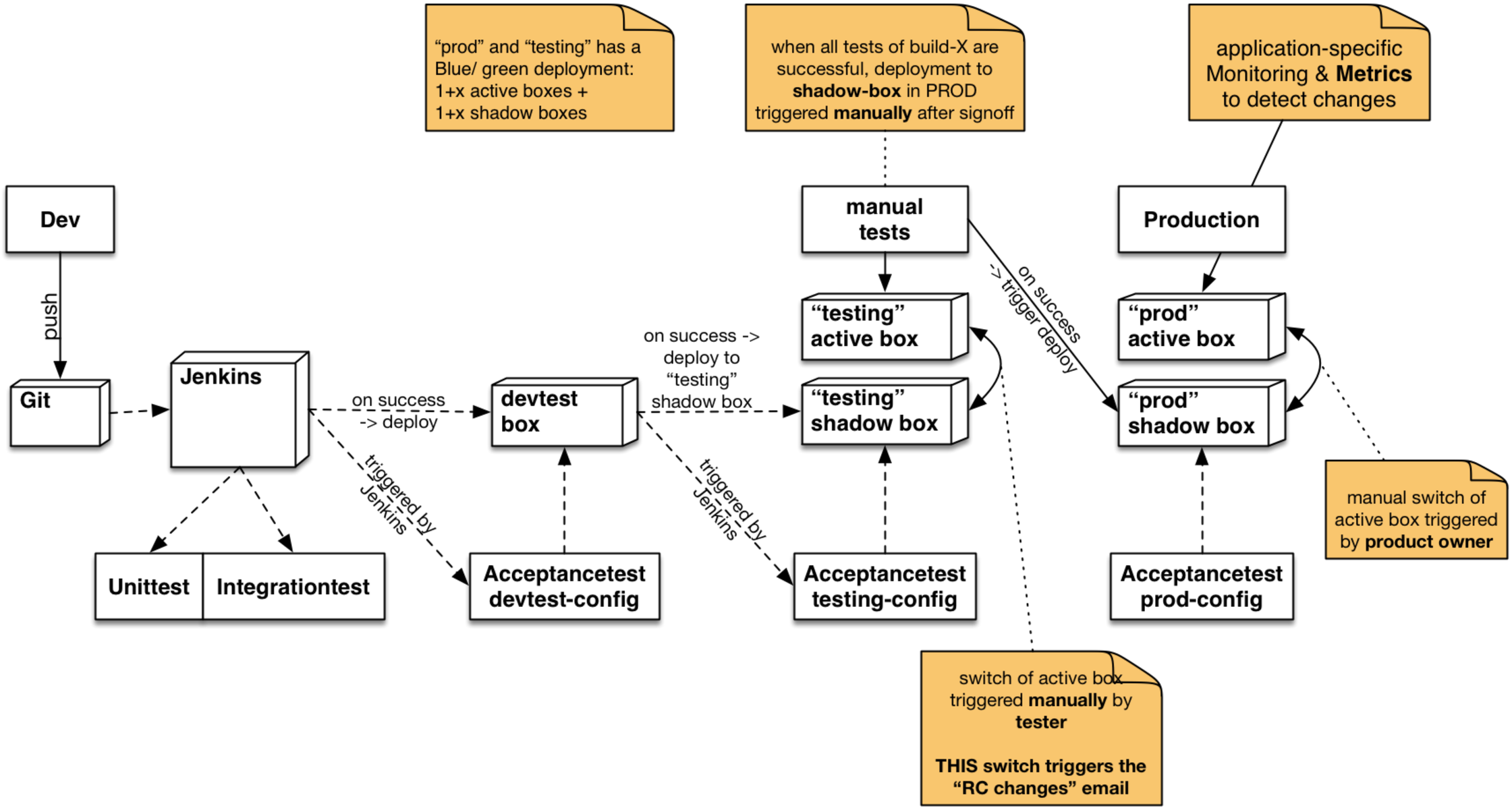
- Business Metrics
- Application Metrics

Kinds of Metrics

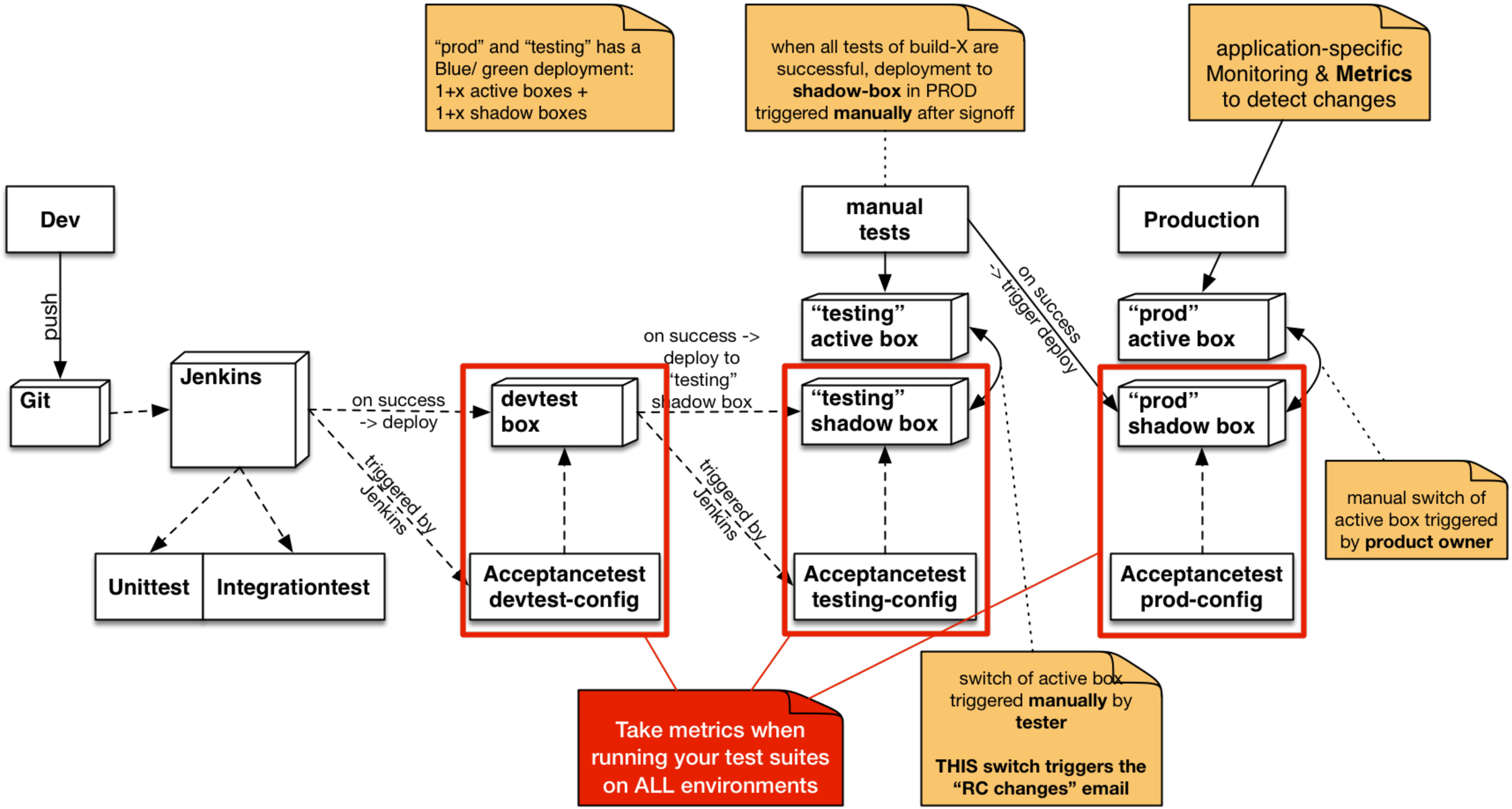
- Business Metrics
- Application Metrics
- System Metrics

Why should a developer care?

Sample of a deployment-pipeline



Sample of a deployment-pipeline



Types of Metrics

Gauges

A gauge is an instrument that measures a value.

© https://secure.flickr.com/photos/profilerehab/4974589604/

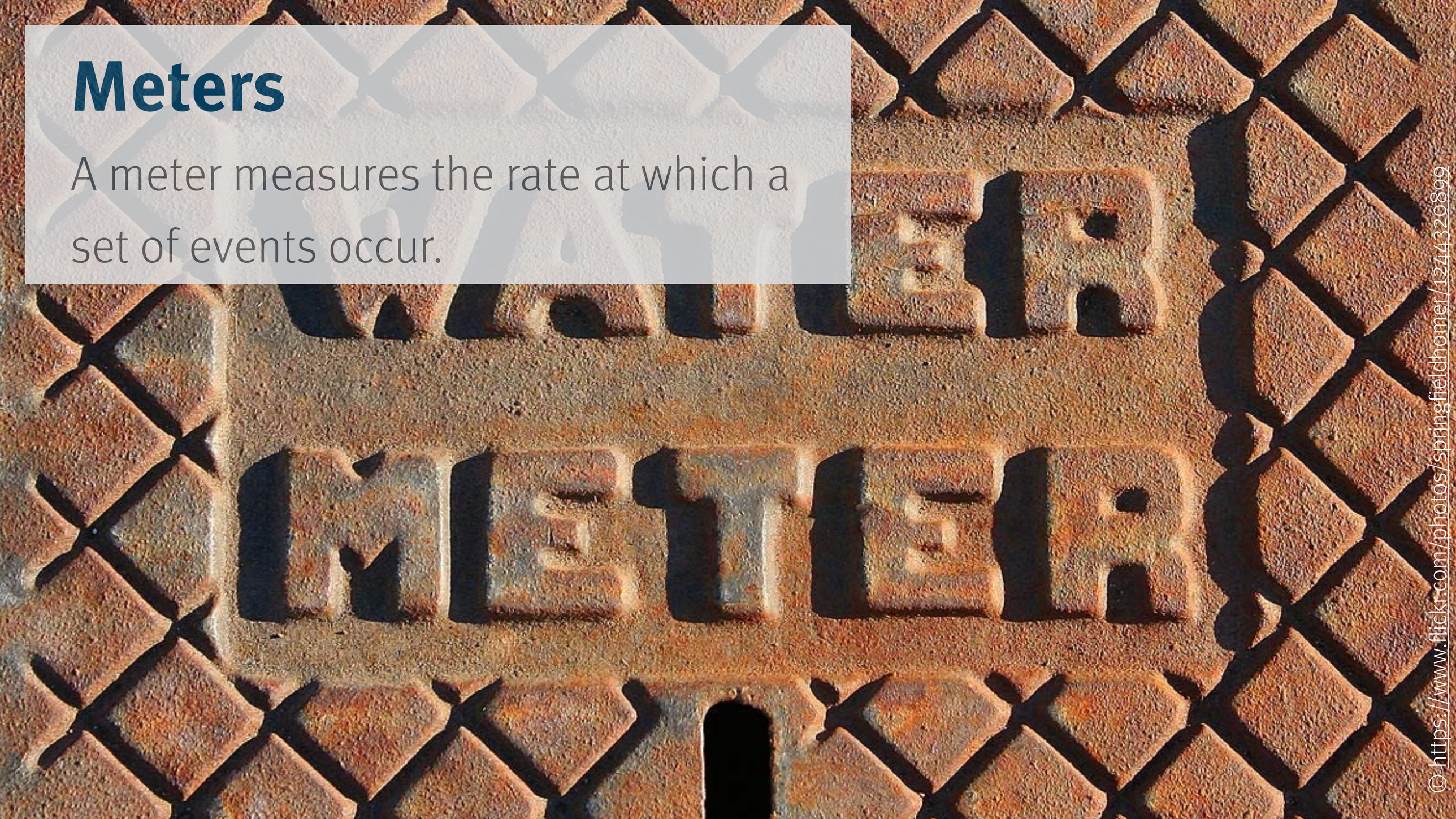
Counters

A counter is a simple incrementing and decrementing integer.

© https://secure.flickr.com/photos/mwichary/2273099939/

Meters

A meter measures the rate at which a set of events occur.



© https://www.flickr.com/photos/springfieldhomer/12443208899

Histograms

A histogram measures the distribution of values.

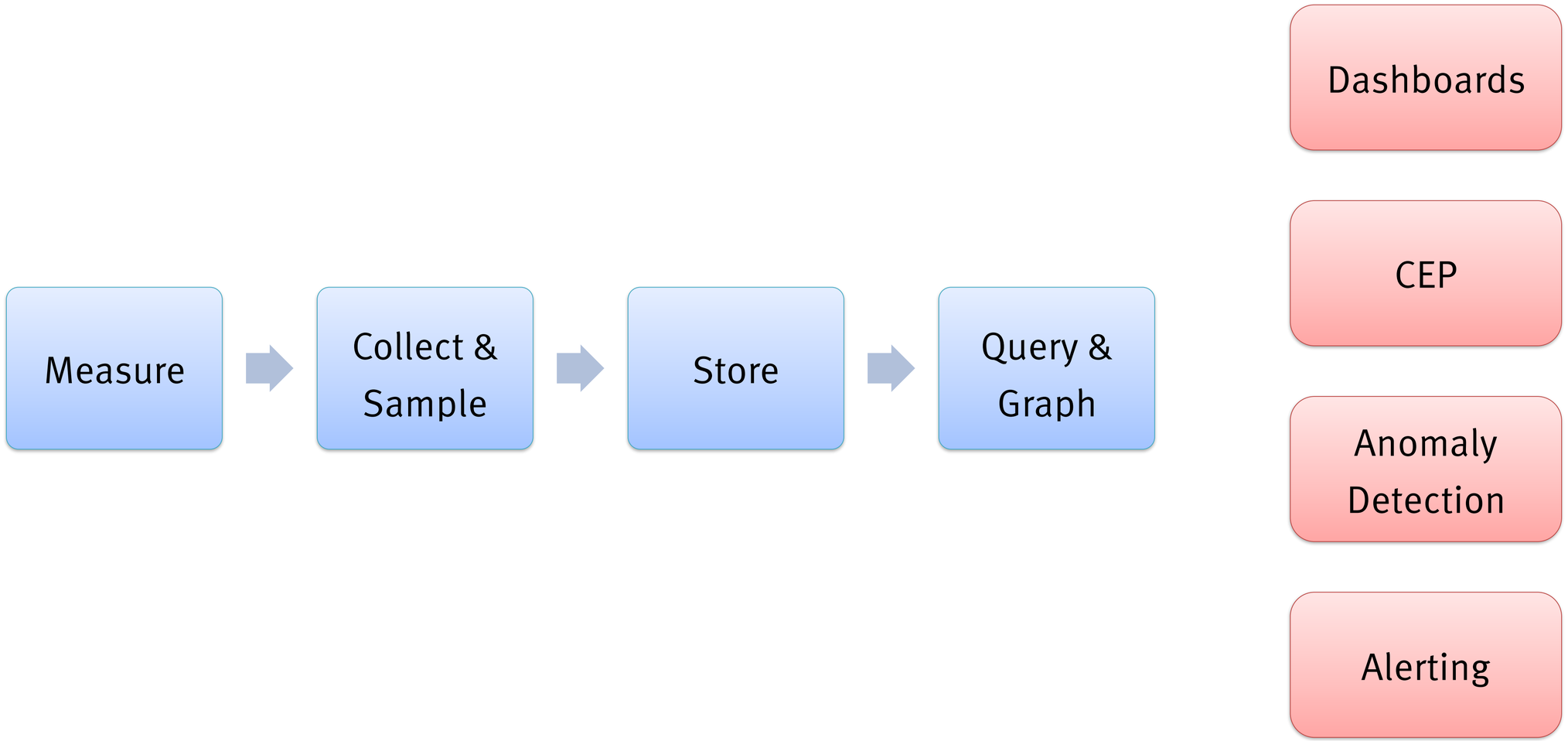
© https://secure.flickr.com/photos/boulter/39

Timers

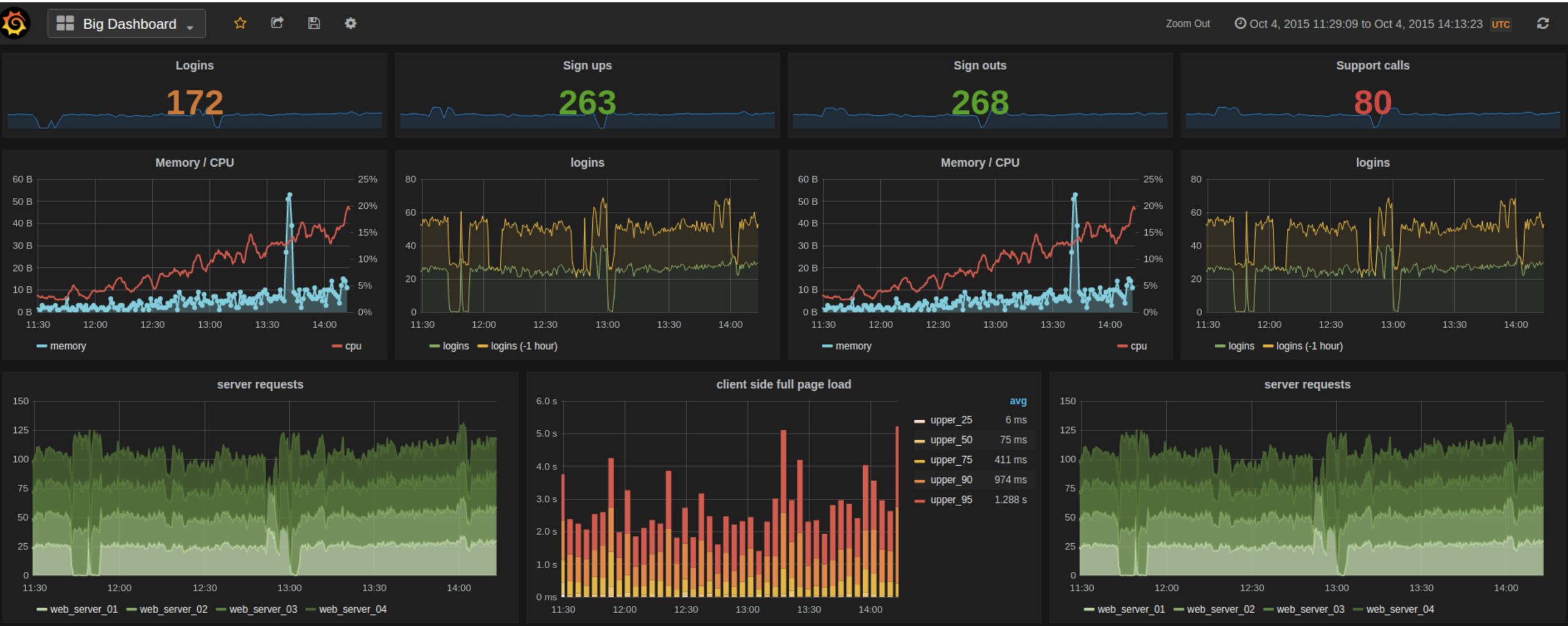
A timer is a histogram over a duration.

© https://secure.flickr.com/photos/psd/4686988937/

Distributed Metrics Architecture

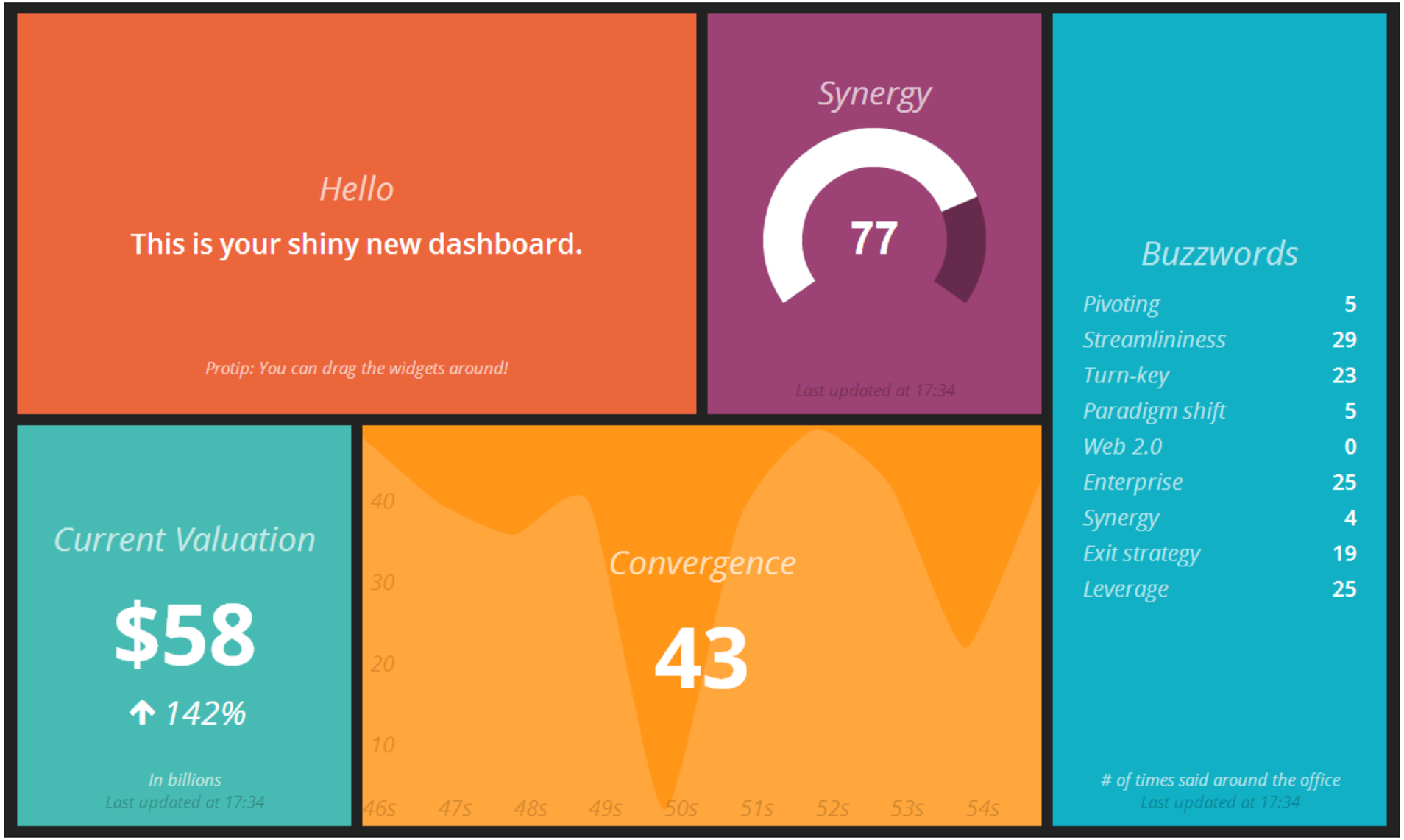


Grafana for Technicians



© http://grafana.org/

Dashing for Management Dashboards



© https://shopify.github.io/dashing/

Push vs. Pull

+ event-based de-/registration
+ routable event stream
+ producer decides when to push
- producer aware of target
- packet-loss might be missed

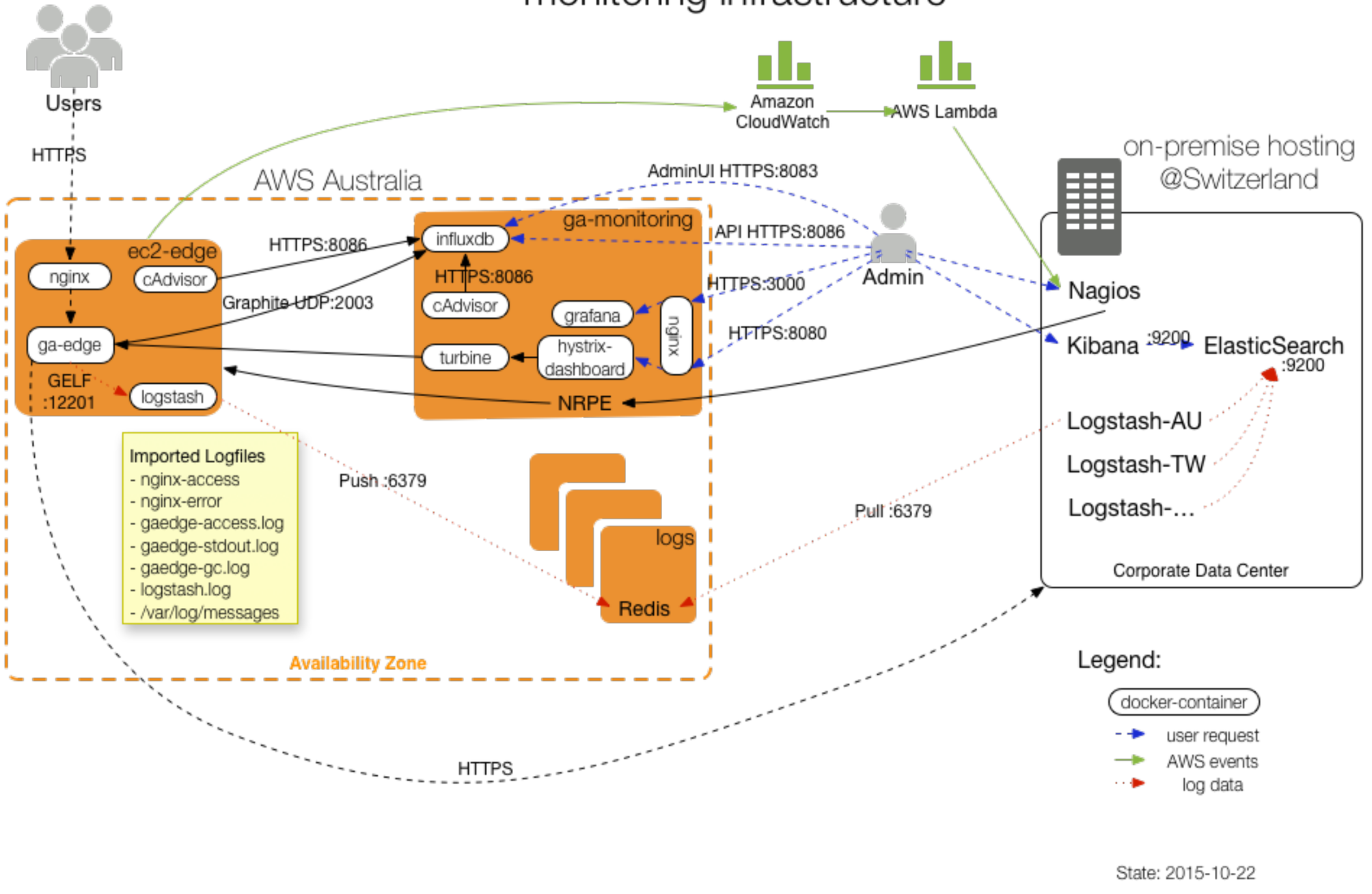
+ producer unaware of target
+ multiple targets possible
+ flexible interval
- might miss short-lived services
- requires service-discovery

Some Recommendations

- Think about what metrics are of importance for operating your application
- Consider retention policies
- Carefully design your dashboards
- Think about non-standard graph types

Sample architecture

monitoring infrastructure



Conclusions

- Create and document concepts for logging and metrics
- Collect & aggregate distributed logs and metrics
- Create dashboards tailored for your audience
- Correlate your data to make conscious decisions
- Don’t create your very own big data problem

Prevent the apocalypse!

Logging shows events.

Metrics show state.

Don't fly blind!

© http://www.flickr.com/photos/pasukaru76/5067879762

Thank you!

Questions?

Comments?

Tammo van Lessen | @taval
tammo.vanlessen@innoq.com

Alexander Heusingfeld | @goldstift
alexander.heusingfeld@innoq.com

https://www.innoq.com/en/talks/2015/10/javaone-2015-logging-metrics-microservices/

innoQ
www.innoq.com

innoQ Deutschland GmbH

Krischerstr. 100
D-40789 Monheim am Rhein
Germany
Phone: +49 2173 3366-0

Ohlauer Straße 43
D-10999 Berlin
Germany
Phone: +49 2173 3366-0

Ludwigstr. 180 E
D-63067 Offenbach
Germany
Phone: +49 2173 3366-0

Kreuzstr. 16
D-80331 München
Germany
Telefon +49 2173 3366-0

innoQ Schweiz GmbH

Gewerbestr. 11
CH-6330 Cham
Switzerland
Phone: +41 41 743 0116