

Using Monitoring and Metrics to learn in Development

Patrick Debois
Atlassian

INTERNATIONAL
SOFTWARE DEVELOPMENT
CONFERENCE

[@patrickdebois](http://jedi.be/blog)

gotocon.com

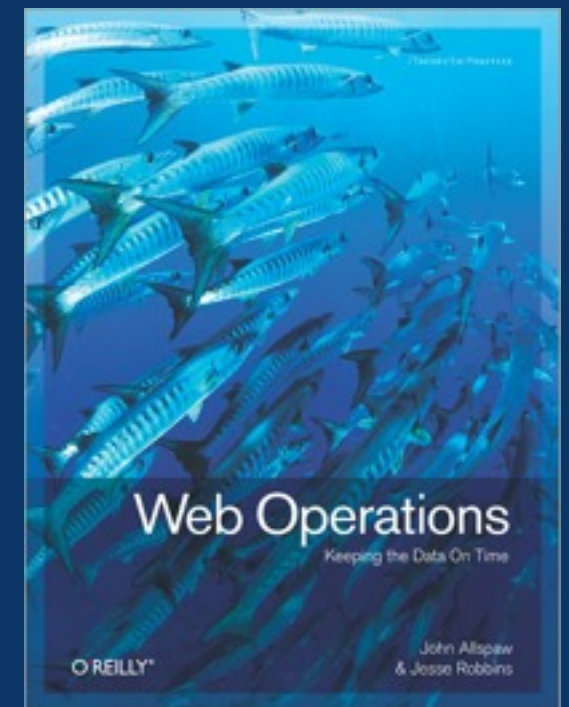


HOSTEDOPS

<https://my.atlassian.com/ondemand/>

CUTTER
CONSORTIUM
●●● ACCESS TO THE EXPERTS

<http://www.cutter.com>



<http://devopsdays.org>



Vagrant &
Veewee



<http://itrevolution.com>

Your Code



ISBN 0-8073-4255-1

C-4055

CAREER EXAMINATION SERIES

PASSBOOKS®

Preferred By
More Test Takers

- Up-to-Date
- Easy to Use
- All Tests
- No Fillers

*This is your
PASSBOOK® for...*

Certified Professional Coder (CPC)

*Test Preparation Study Guide
Questions & Answers*

NLC

NATIONAL LEARNING CORPORATION

PLASTIC BOUND -
LIES FLAT FOR
STUDY EASY!

“Test Driven Development”

- CI Coward

Testing in Controlled Environment

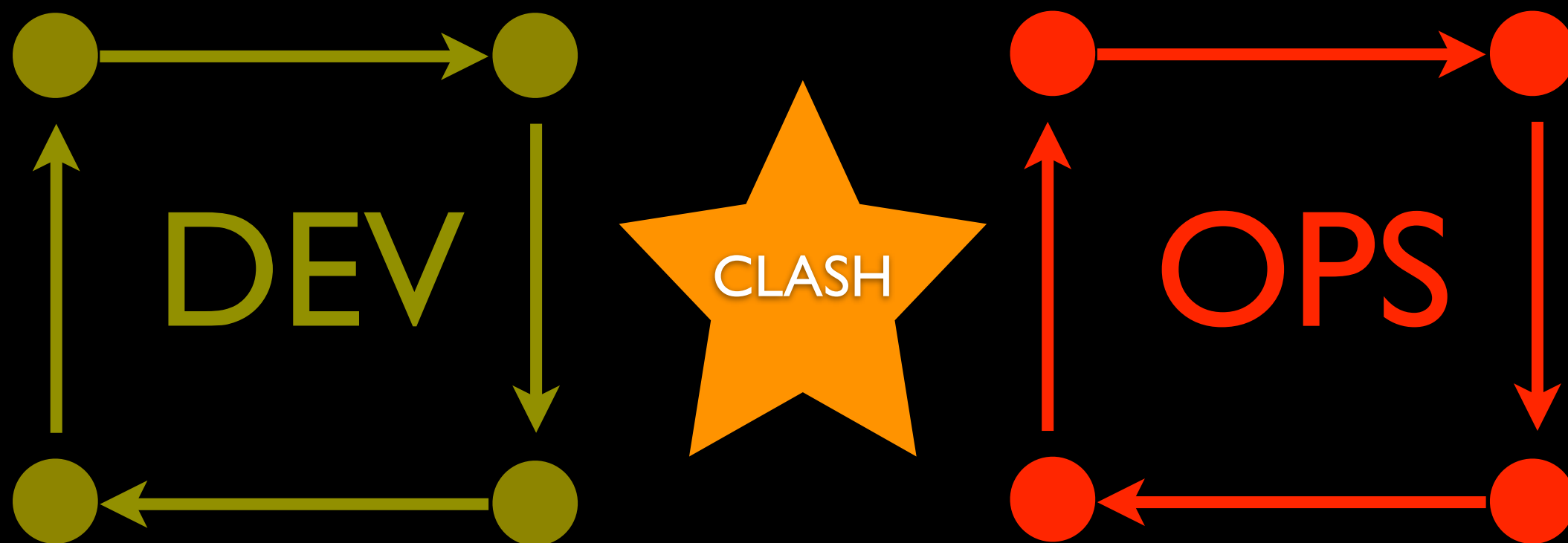


Your Code in Test



“Until code runs in production ,
your code is just inventory”
- Agile Anonymous

Continuous Delivery

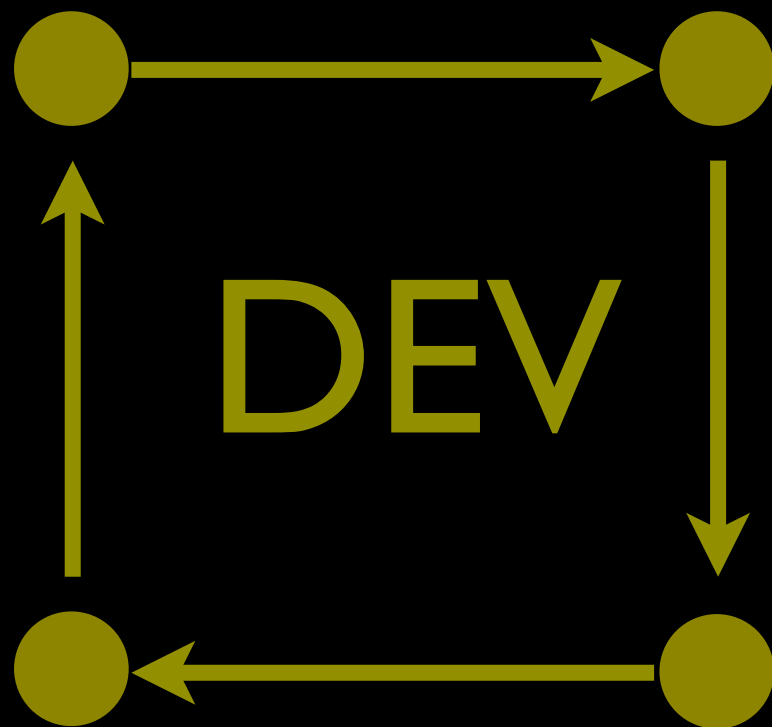




Shit Happens
in production

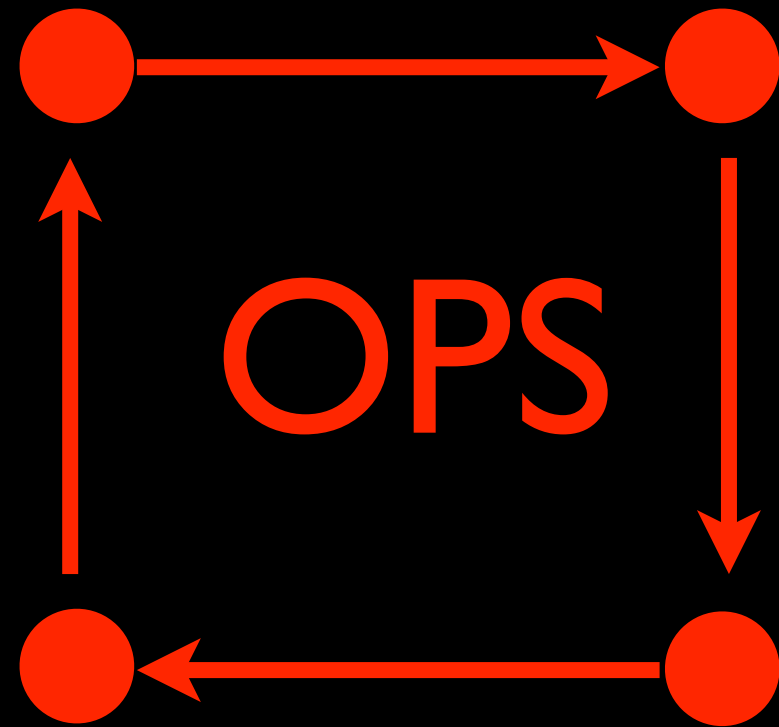
“You go to production
with the code you have,
not the code you wish you had”
- Devops Rumsfeld

Dreamer



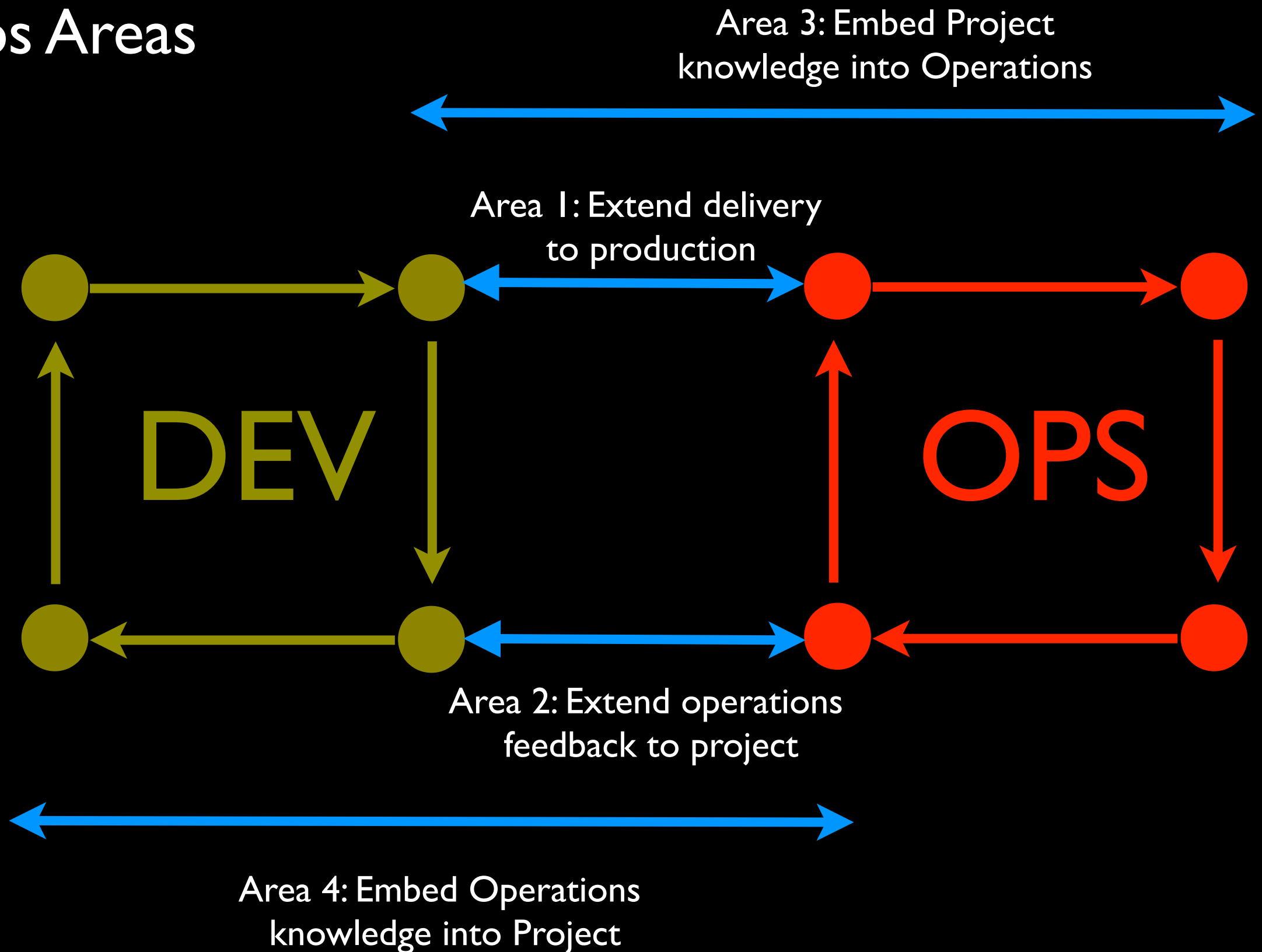
Realist

Critic

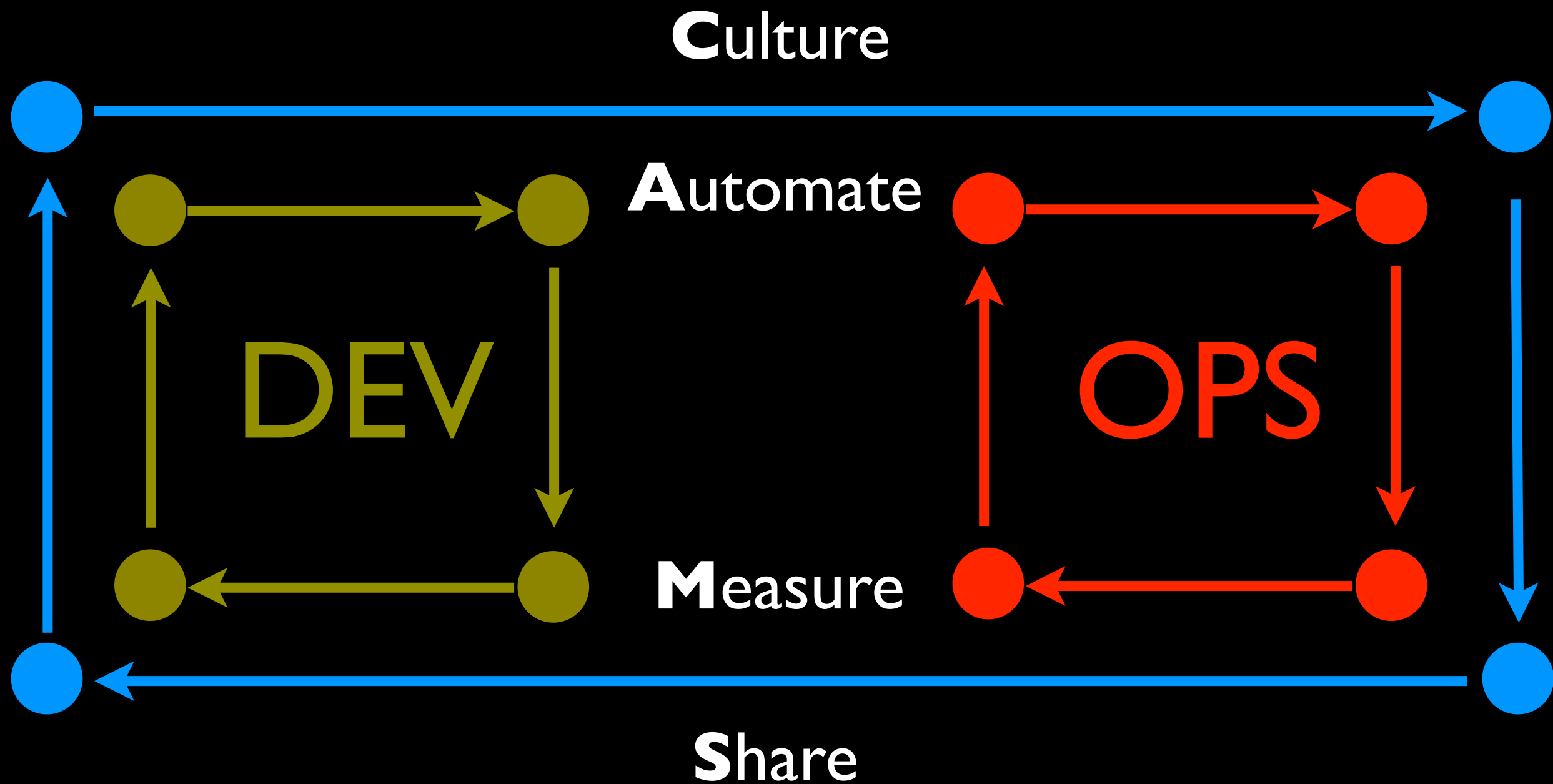


Three Phases of Creativity (Disney)

Devops Areas

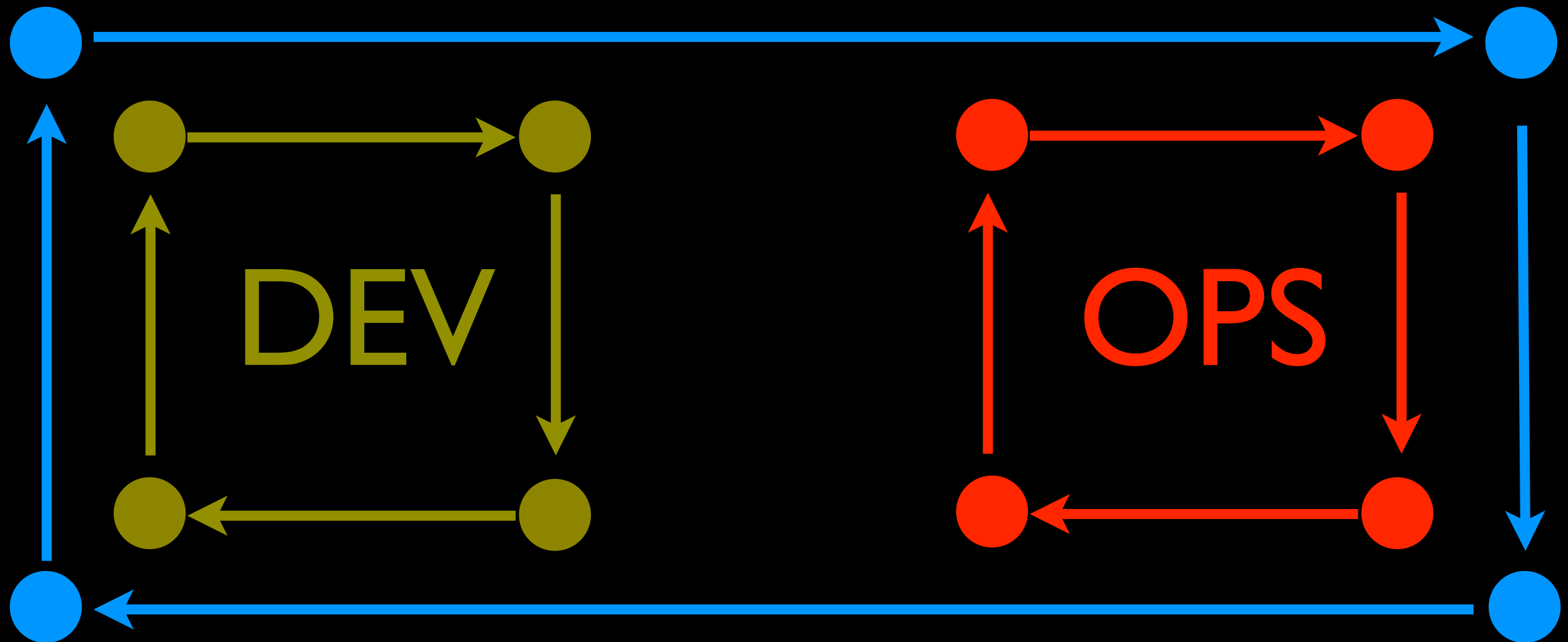


<http://jedi.be/blog/2012/05/12/codifying-devops-area-practices/>



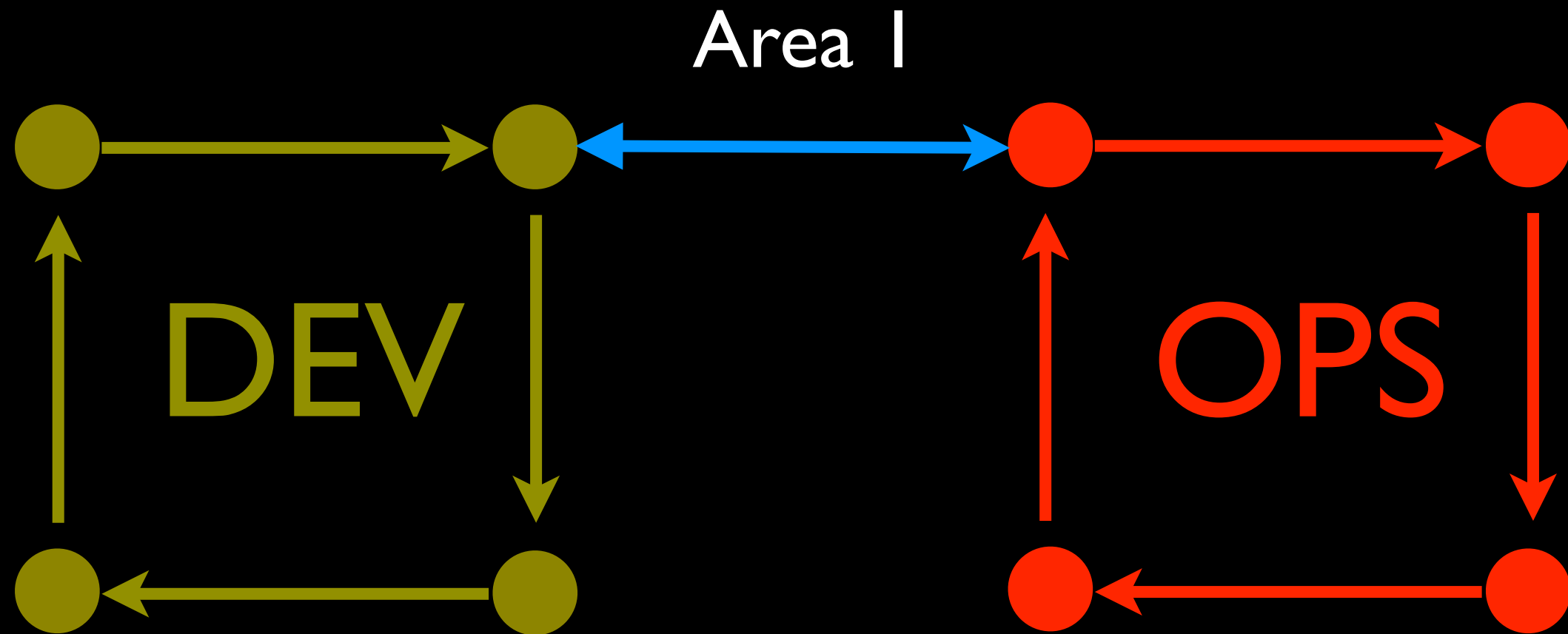
<http://www.opscode.com/blog/2010/07/16/what-devops-means-to-me/>

Smaller and more frequent changes



Faster and better feedback

Area I: Extend delivery to production “think Jez Humble”



Continuous Integration Maturity Model

Intro	Novice	Intermediate	Advanced	Insane
<ul style="list-style-type: none">• Using Source Control• Nightly Builds• Current Issues Tracked & Knowledge base	<ul style="list-style-type: none">• Builds triggered on Commit• Automated Deploy to Dev• Unit Tests Run on every build• Generate Change Log• Collect Code Coverage Data• Code Integrity Checks• Static Analysis Used• Run-time Analysis• Automated API Documentation generation	<ul style="list-style-type: none">• Automated Deployment to testing environments• On demand deployments to controlled environments• Per Env. Smoke Test• Manual Test Results in CI Server• Business visibility, reports• Flag a CI build as Release Candidate (promotion)• Product Activity Metrics• Auto-update Defect Tracking• Pre-commit Builds• Data roll-up• Automatic Cleanup of old Data	<ul style="list-style-type: none">• Automated Func. Testing• Multi-Threaded / Scalable build systems• Change Reporting / SQA Impacts• Defect Trending• Identify Problem Code from Metrics• Auto-Deploy to Prod• Auto-Rollback in Prod• Environment Monitoring• Alerts based on build metric thresholds• Security Scans	<ul style="list-style-type: none">• Continuous Deployment to Prod

<http://www.slideshare.net/Urbancode/continuous-delivery-maturity-model>

Infrastructure Code Repository

DEV

TEST

PROD

WEB

WEB

WEB

APP

APP

APP

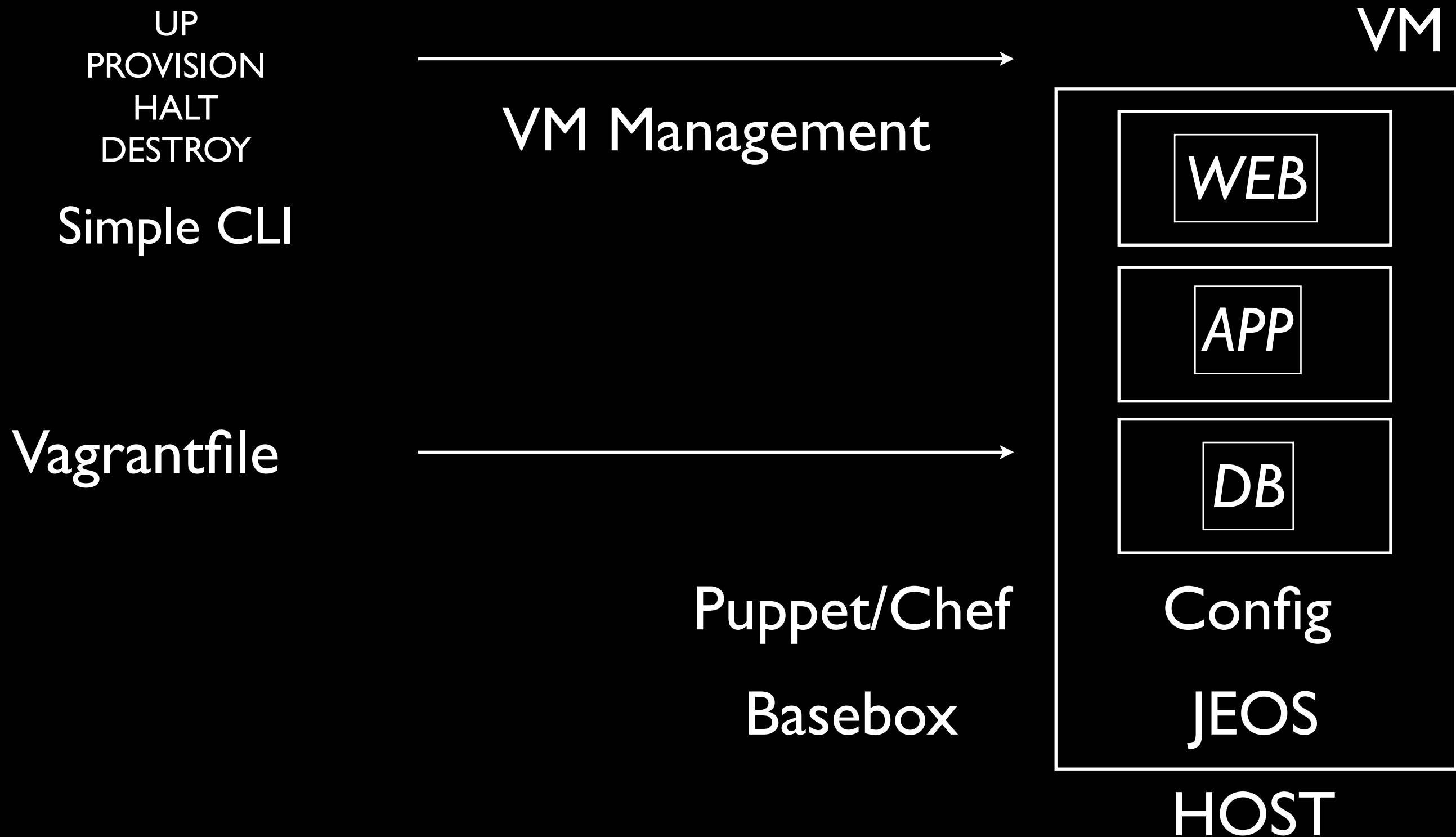
DB

DB

DB

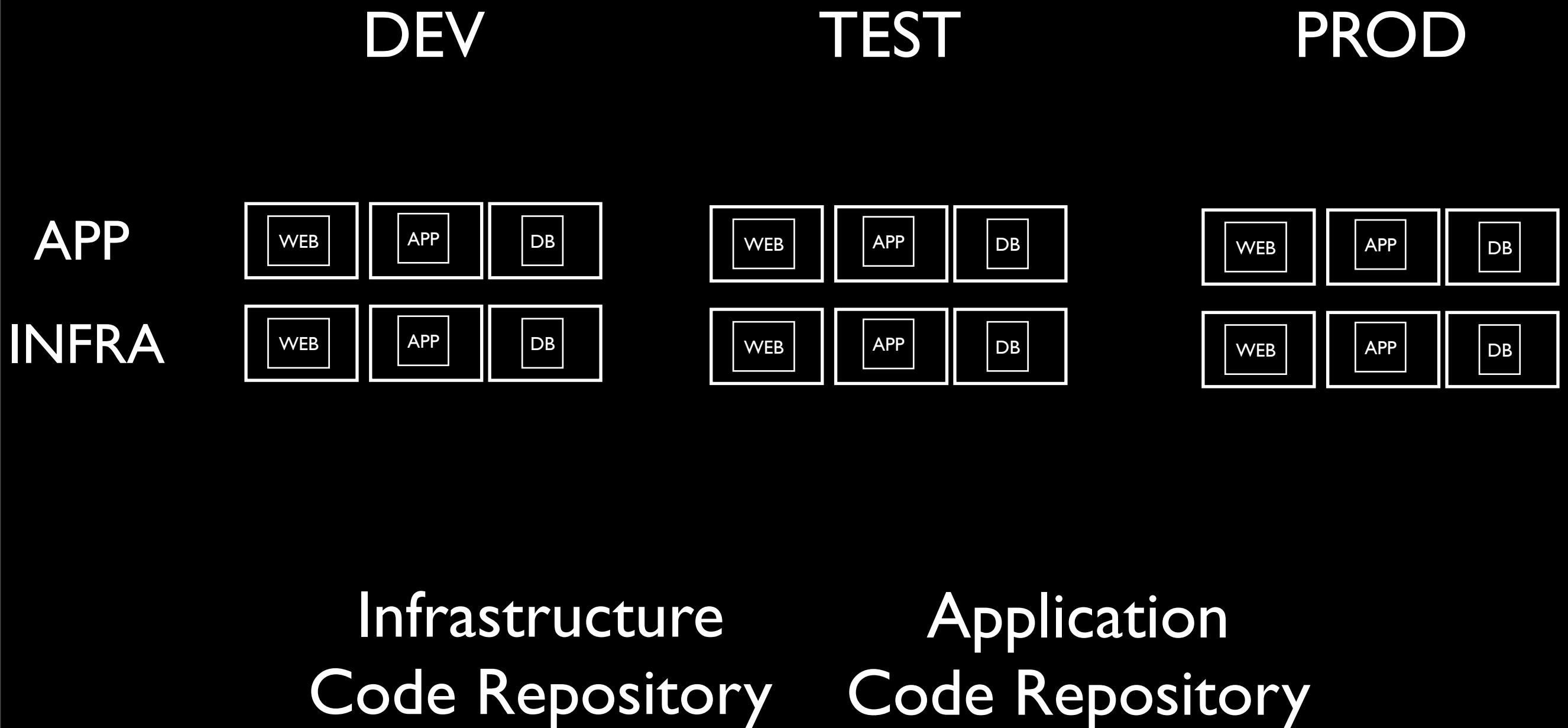
Reuse “Infra code” across Environments

Reuse “workflow” across Environments



<http://vagrantup.com/>

Integrate with **Continuous Integration**



Continuous Integration to Continuous Delivery

Faster/Delivery



DEV

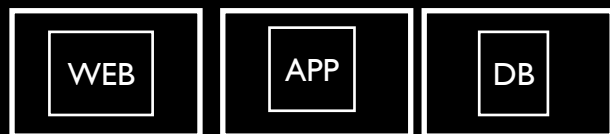
TEST

PROD

APP



INFRA



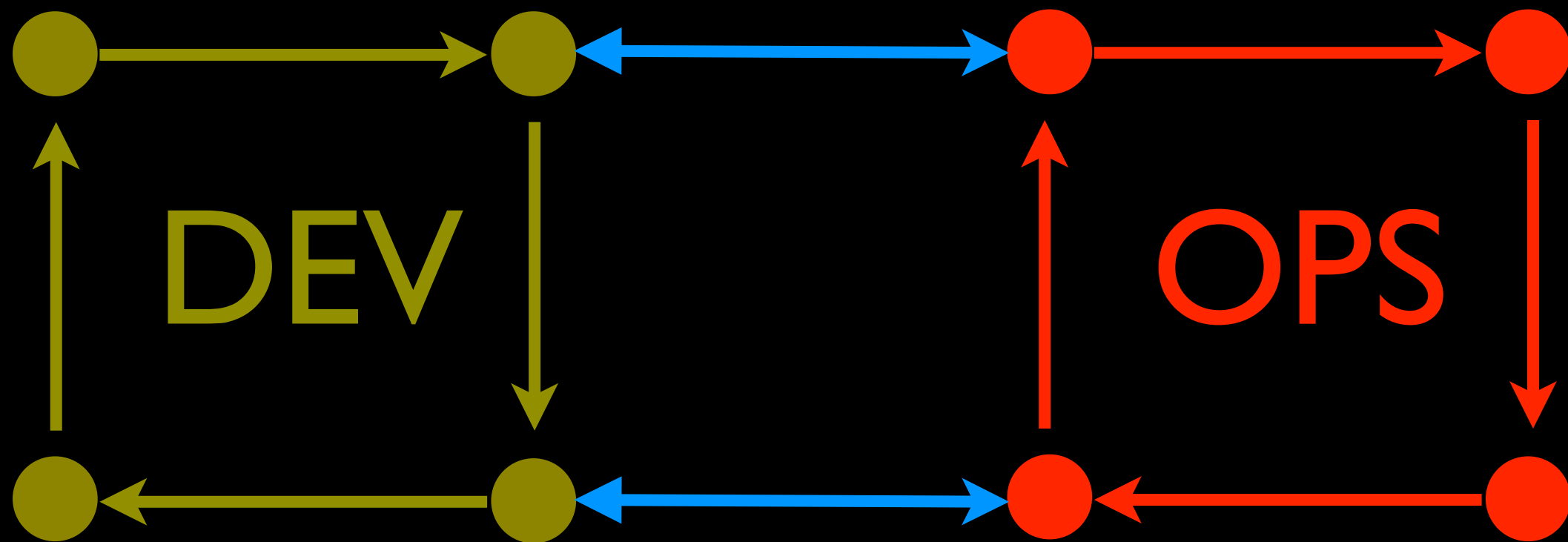
Faster/Feedback



Infrastructure
Code Repository

Application
Code Repository

Area 2: Extend operations feedback to project think “John Allspaw”



Area2

Monitoring Layers



Monitoring Layers

Behavioral (larger coverage)



Diagnostics

“What part made it fail”
~unit test

CPU Memory Network

“Is it working?”
~behavioral test

Monitoring Driven Development

Create a monitor check before implementing a feature

**UC-1 Availability:
100%**

Information Radiator



<http://jedi.be/blog/2012/01/03/monitoring-wonderland-survey-introduction/>

<http://jedi.be/blog/2012/01/03/monitoring-wonderland-nagios-the-mighty-beast/>

<http://jedi.be/blog/2012/01/04/monitoring-wonderland-moving-up-the-stack-application-user-metrics/>

<http://jedi.be/blog/2012/01/03/monitoring-wonderland-metrics-api-gateways/>

<http://jedi.be/blog/2012/01/04/monitoring-wonderland-visualization/>



“traditional
In house monitoring”



Nagios

General

- Home
- Documentation

Monitoring

- Tactical Overview
- Service Detail
- Host Detail
- Status Overview
- Status Summary
- Status Grid
- Status Map
- 3-D Status Map
- Service Problems
- Host Problems
- Network Outages
- Comments
- Downtime
- Process Info
- Performance Info
- Scheduling Queue

Reporting

- Trends
- Availability
- Alert Histogram
- Alert History
- Alert Summary
- Notifications
- Event Log

Configuration

- View Config

Current Network Status

Last Updated: Tue Jul 6 11:22:11 CEST 2004
Updated every 90 seconds
Nagios® - www.nagios.org
Logged in as *bingel*

[View History For all hosts](#)
[View Notifications For All Hosts](#)
[View Host Status Detail For All Hosts](#)

Host Status Totals

Up	Down	Unreachable	Pending
15	2	0	0
All Problems		All Types	
2		17	

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
29	1	1	4	0
All Problems		All Types		
6		35		

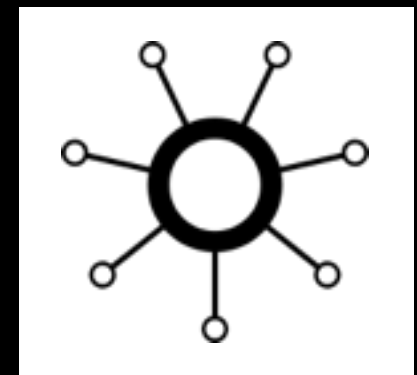
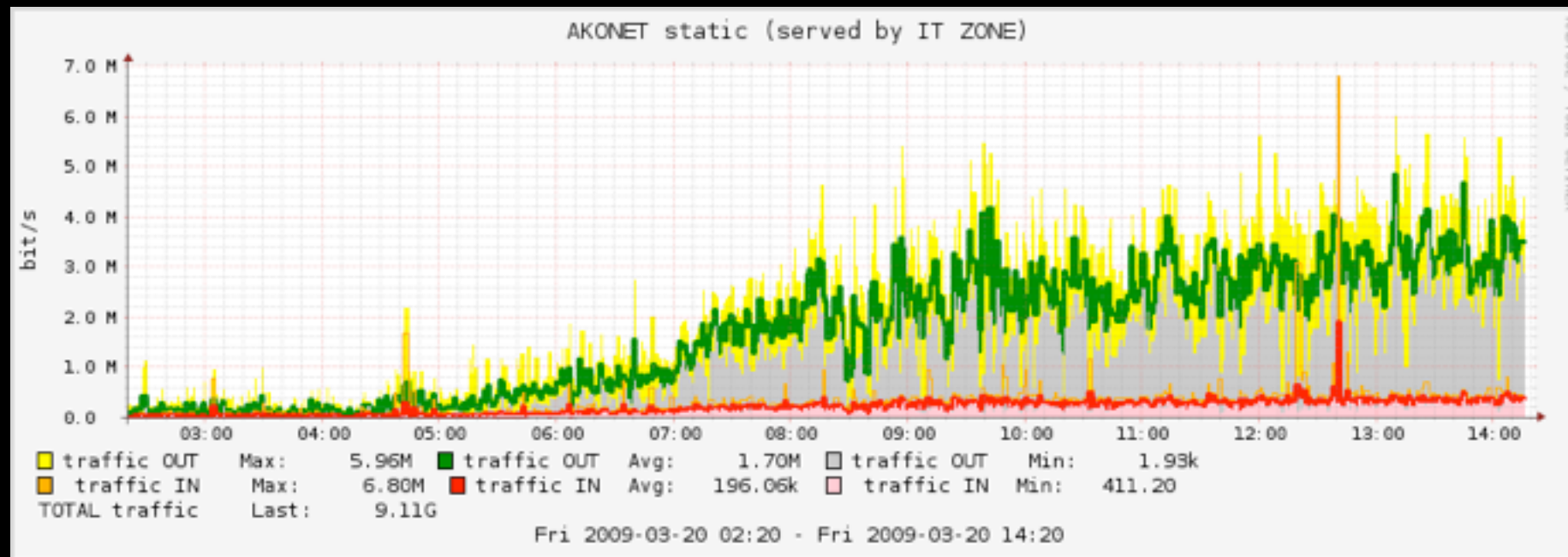
Service Status Details For All Hosts

Host ↑↓	Service ↑↓	Status ↑	Last Check ↑↓	Duration ↑↓	Attempt ↑	Status Information
bart	Diskusage C:	OK	2004-07-06 11:10:19	20d 0h 7m 26s	1/3	C:\ - total: 3.91 Gb - used: 2.62 Gb (67%) - free 1.28 Gb (33%)
	Diskusage D:	WARNING	2004-07-06 11:14:13	20d 0h 7m 20s	3/3	D:\ - total: 29.99 Gb - used: 26.78 Gb (89%) - free 3.20 Gb (11%)
	HTTP	OK	2004-07-06 11:21:41	14d 1h 37m 47s	1/3	HTTP ok: HTTP/1.1 200 OK - 0.027 second response time
	MS-Exchange	OK	2004-07-06 11:20:58	20d 0h 8m 6s	1/3	All services are running
	SMTP	OK	2004-07-06 11:21:33	22d 1h 53m 13s	1/3	SMTP OK - 0 second response time
cixten	Diskusage C:	OK	2004-07-06 11:19:45	0d 22h 2m 24s	1/3	C:\ - total: 39.06 Gb - used: 8.01 Gb (21%) - free 31.05 Gb (79%)
	Terminalserver Sessions	OK	2004-07-06 11:18:01	0d 22h 29m 13s	1/3	11
ftp.sunet.se	PING	OK	2004-07-06 11:21:30	0d 0h 31m 24s	1/10	PING OK - Packet loss = 0%, RTA = 36.37 ms
haubits	PING	OK	2004-07-06 11:21:21	193d 8h 33m 3s	1/10	PING OK - Packet loss = 0%, RTA = 1.36 ms
tknsgw1	PING	OK	2004-07-06 11:21:21	56d 22h 30m 57s	1/10	PING OK - Packet loss = 0%, RTA = 6.10 ms
	if-traffic	OK	2004-07-06 11:21:20	63d 18h 45m 23s	1/10	OK: rate[IN]=250 kbit/s OK: rate[OUT]=286 kbit/s
tknsgw2	PING	CRITICAL	2004-04-20 12:39:00	77d 20h 45m 1s	10/10	PING CRITICAL - Packet loss = 100%
	if-traffic	UNKNOWN	2004-04-20 12:39:00	77d 20h 45m 1s	10/10	check_snmp_counter: ERROR during get-request: No response from remote host '62.119.68.186'
jasper	Diskusage C:	OK	2004-07-06 11:15:46	77d 1h 13m 47s	1/3	C:\ - total: 4.00 Gb - used: 2.99 Gb (75%) - free 1.01 Gb (25%)
	Diskusage E:	OK	2004-07-06 11:20:41	140d 1h 3m 7s	1/3	E:\ - total: 4.00 Gb - used: 1.46 Gb (36%) - free 2.54 Gb (64%)
	MS-Exchange	OK	2004-07-06 11:21:41	61d 22h 57m 53s	1/3	All services are running
	MS-Exchange NotesConnector	OK	2004-07-06 11:21:41	61d 22h 57m 57s	1/3	All services are running
	SMTP	OK	2004-07-06 11:21:43	61d 22h 58m 4s	1/3	SMTP OK - 0 second response time
lenin	SMTP	OK	2004-07-06 11:21:51	20d 2h 18m 14s	1/3	SMTP OK - 0 second response time

Nagiosstat goes here



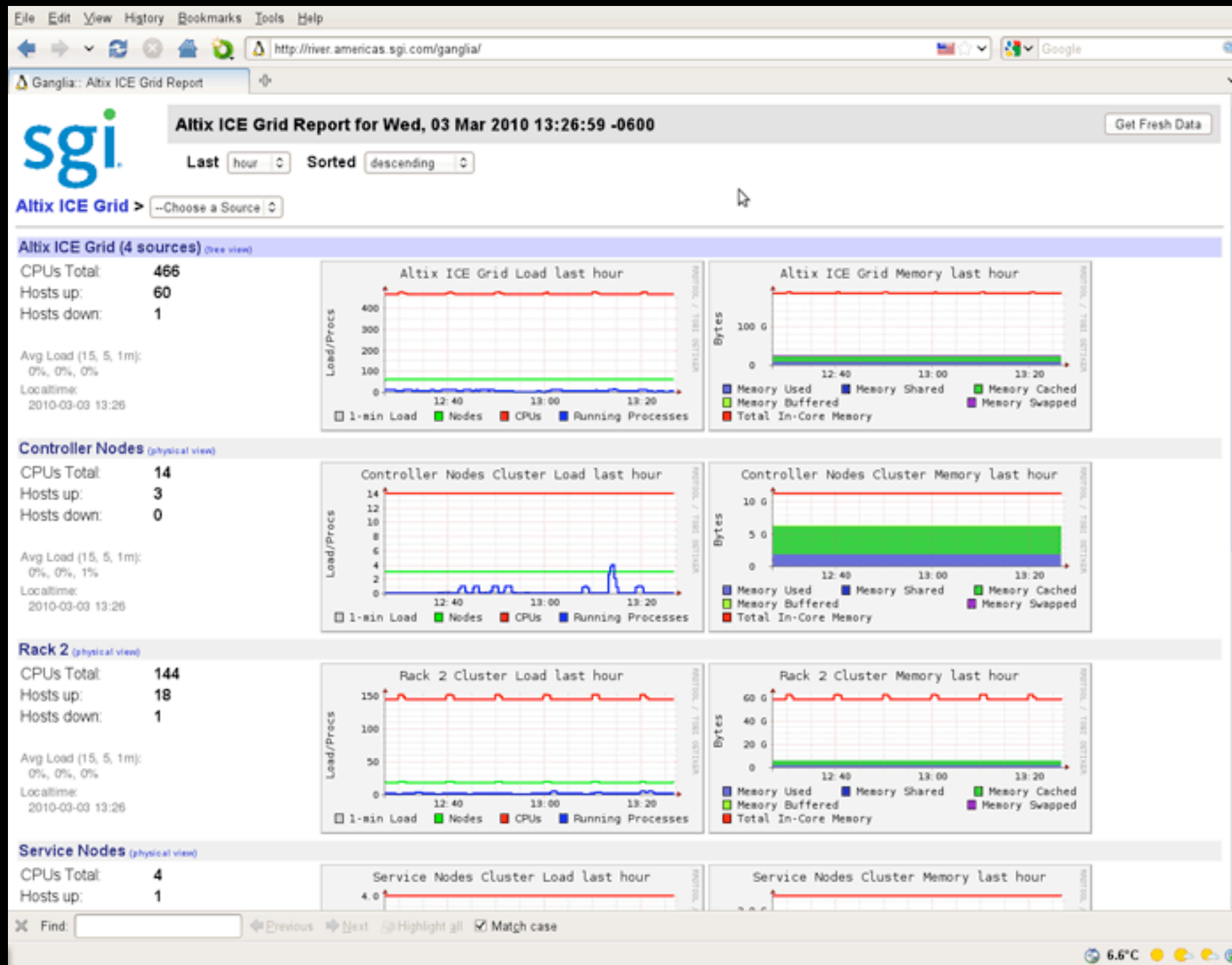
“traditional In house metrics”



<http://munin-monitoring.org/>

<http://collectd.org/>

Ganglia



<http://ganglia.info/>

[https://github.com/
monitoringsucks](https://github.com/monitoringsucks)

“1980” style UI

No API

Hard to scale

Static configurations

Hard to expose “data”
UI vs API

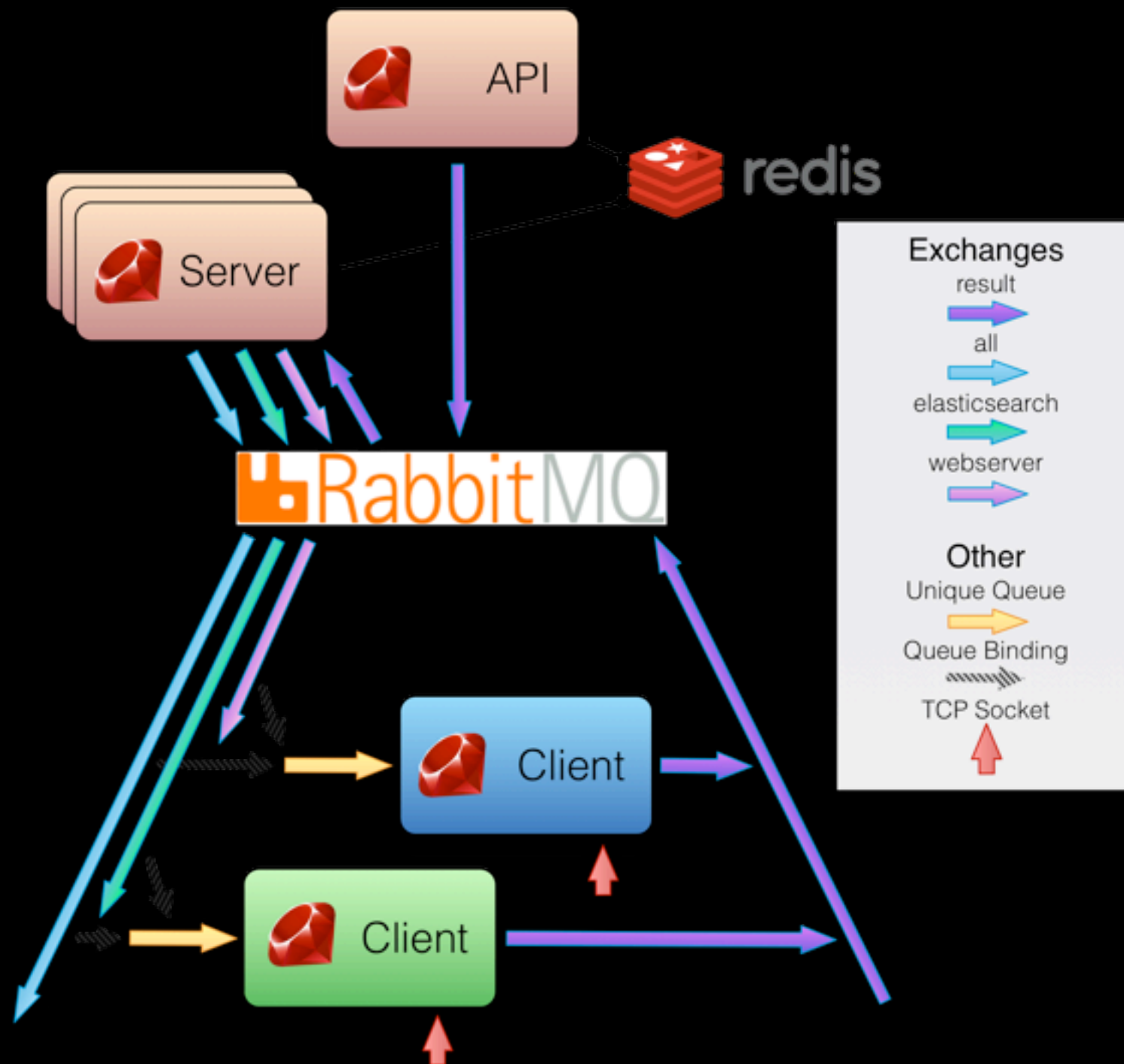
Not self-servicing



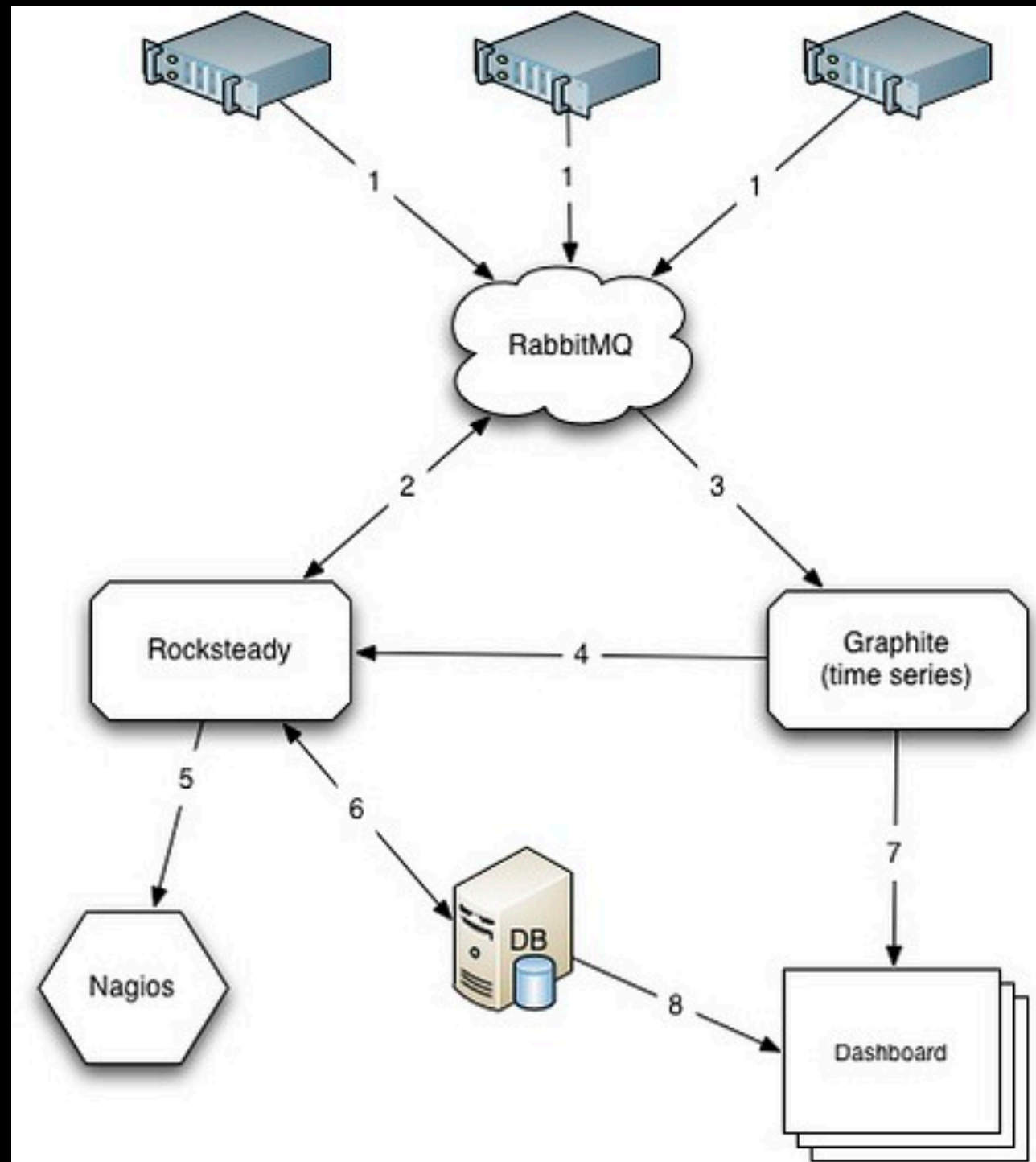
“Monitoring as a service”



Sonian Cloud Monitoring “Sensu”



<https://github.com/sensu/sensu>



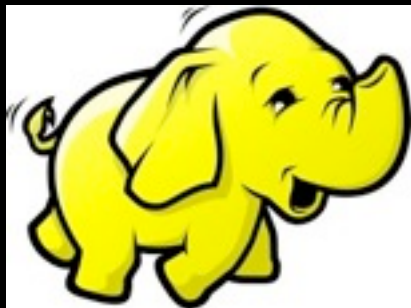
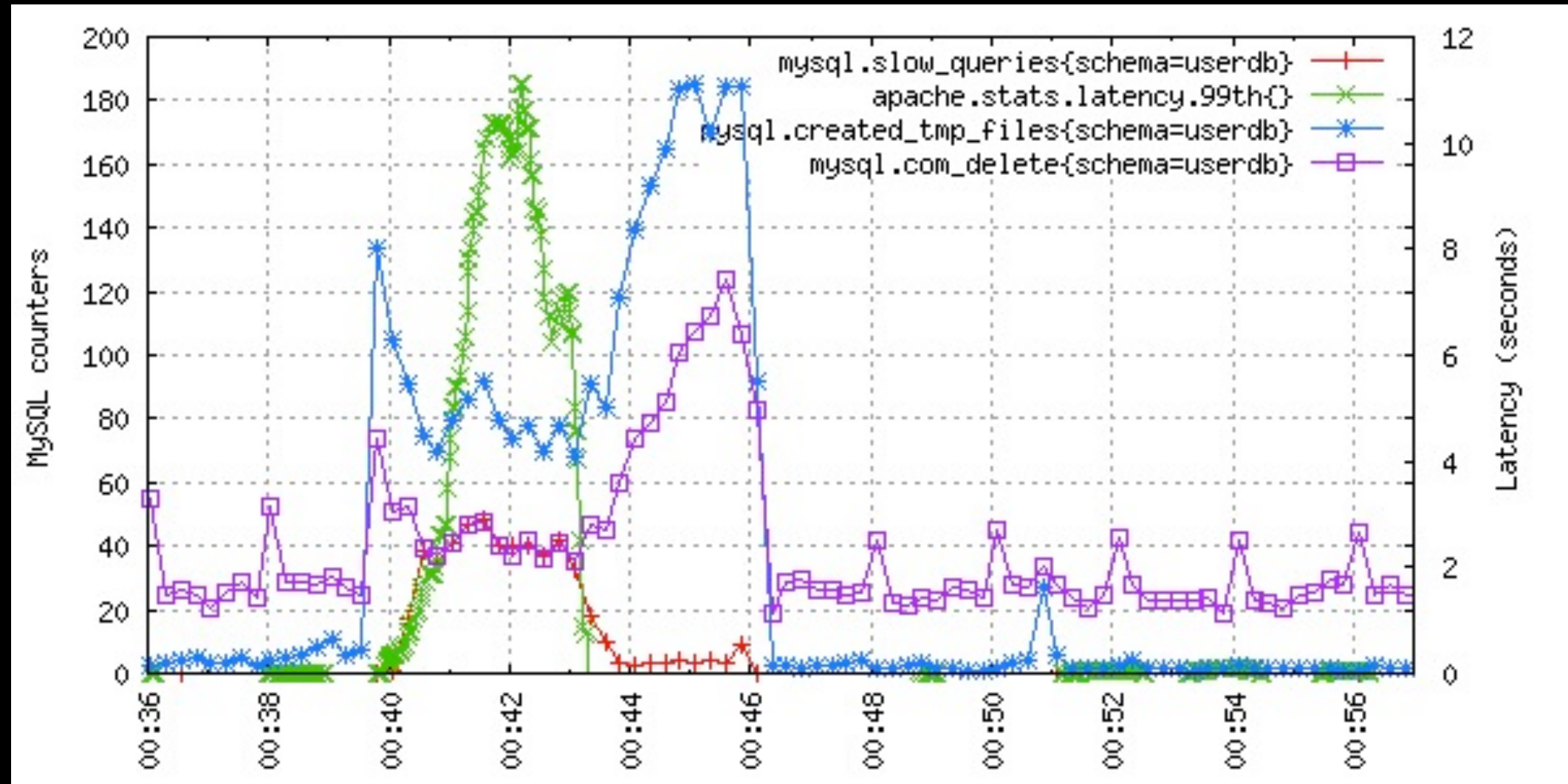
<http://code.google.com/p/rocksteady/>



Reconnoiter's goal is to better the world of monitoring by marrying fault detection and trending together. Through ease of configuration and ongoing maintenance, Reconnoiter encourages monitoring important technical metrics along side critical business metrics to improve awareness and ultimately accountability.

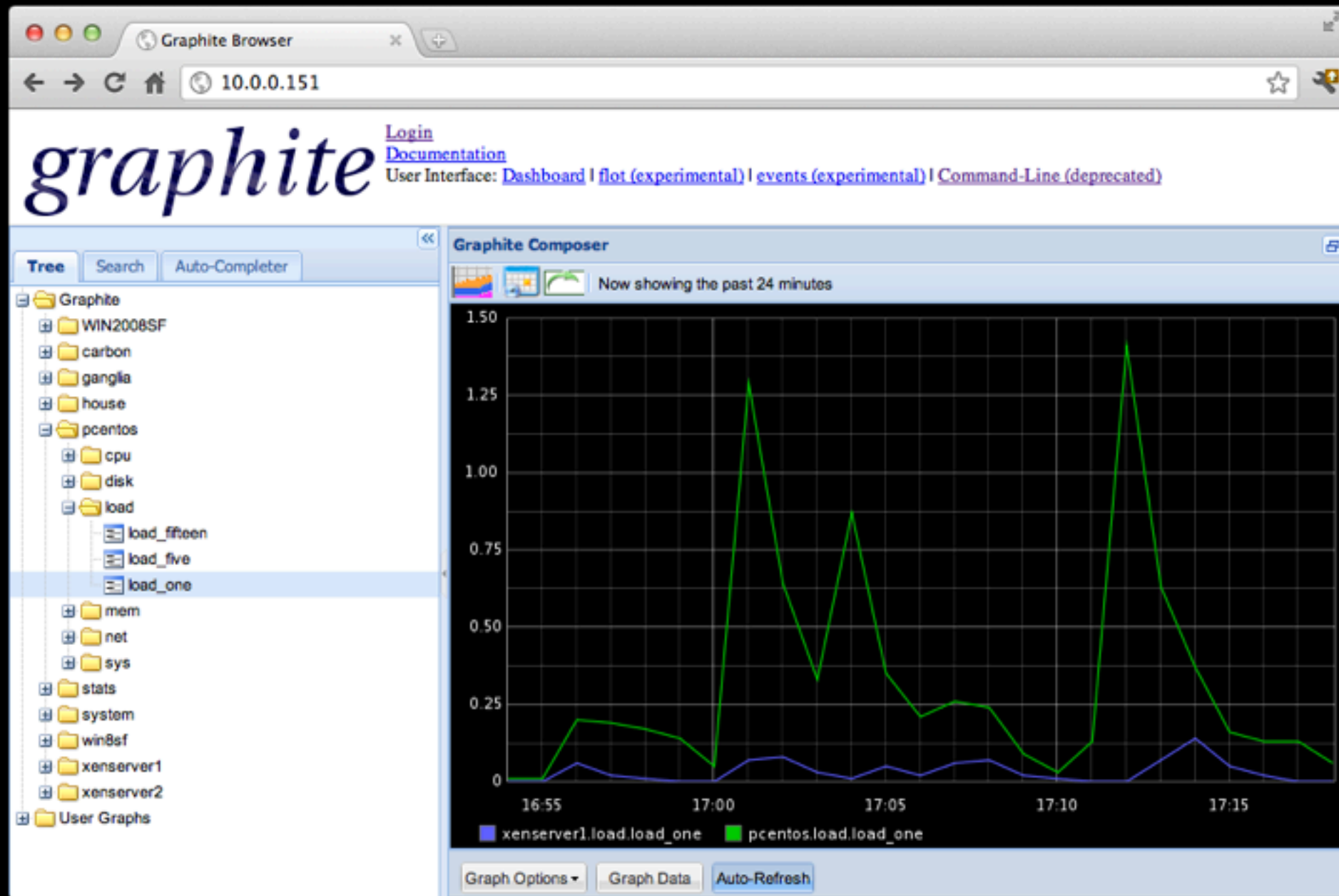
<http://labs.omniti.com/labs/reconnoiter>

Scalable writes



<http://opentsdb.net/>

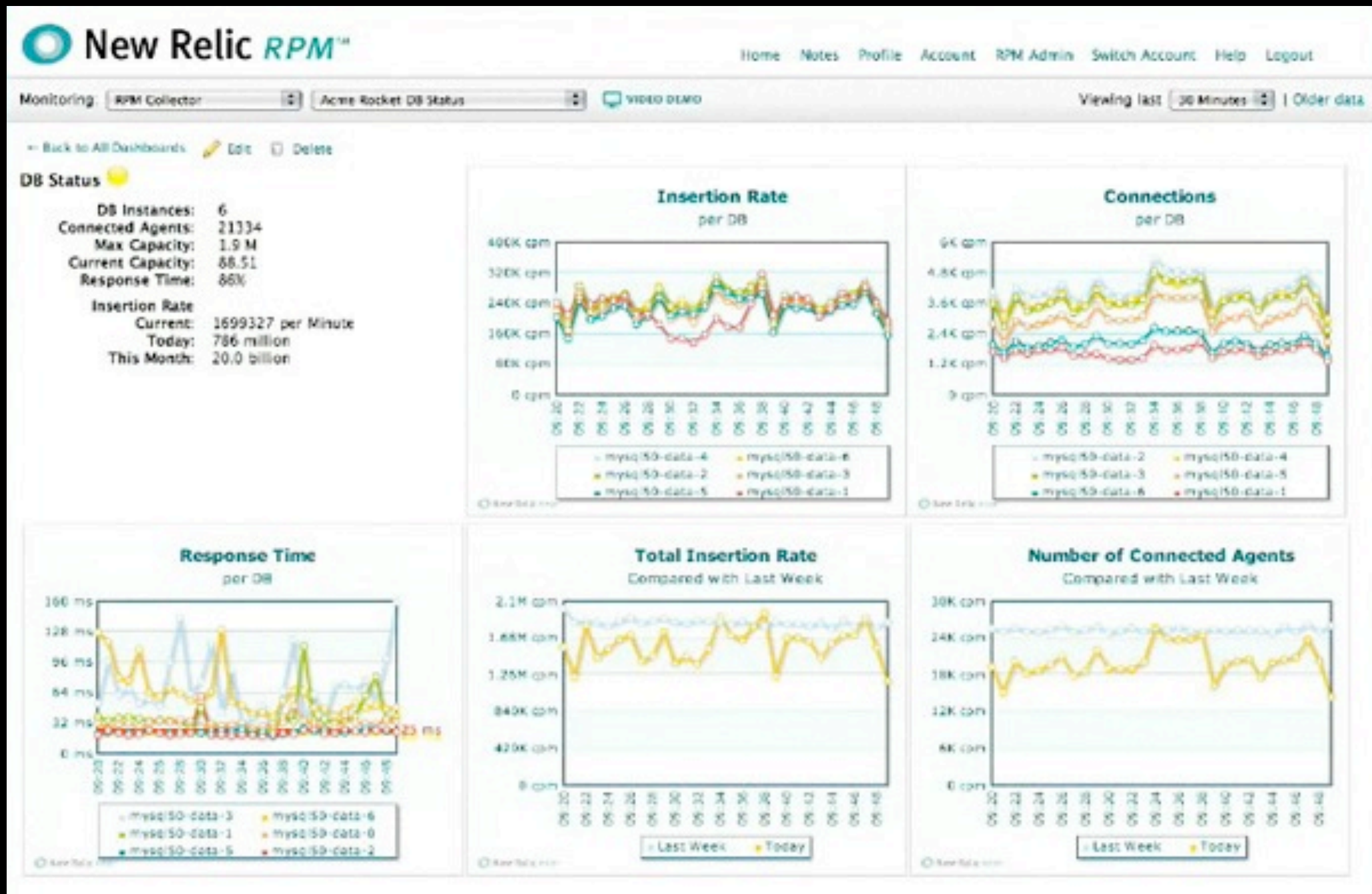
Selfservicing Graphs



<http://graphite.wikidot.com/>

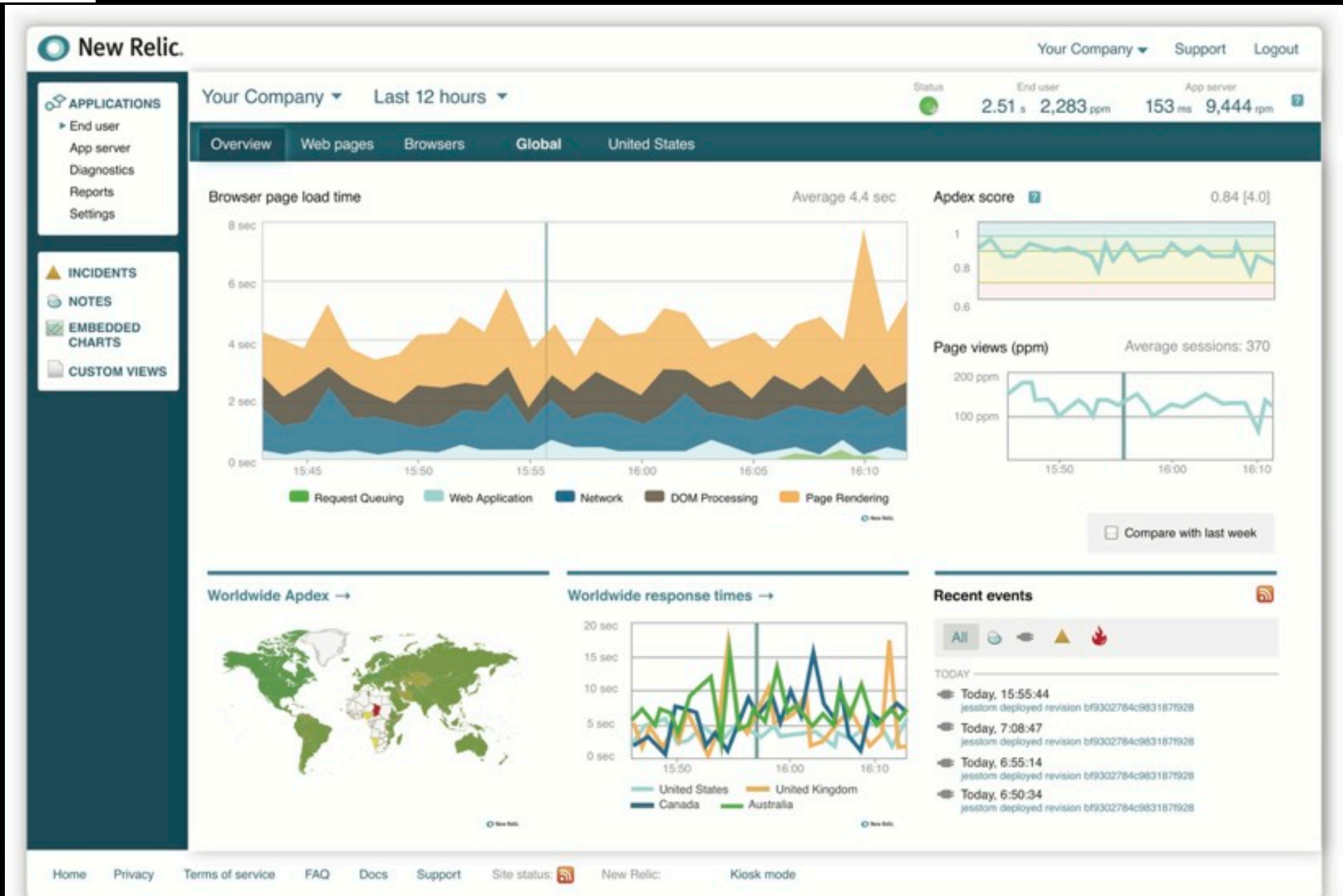


“App Metrics”



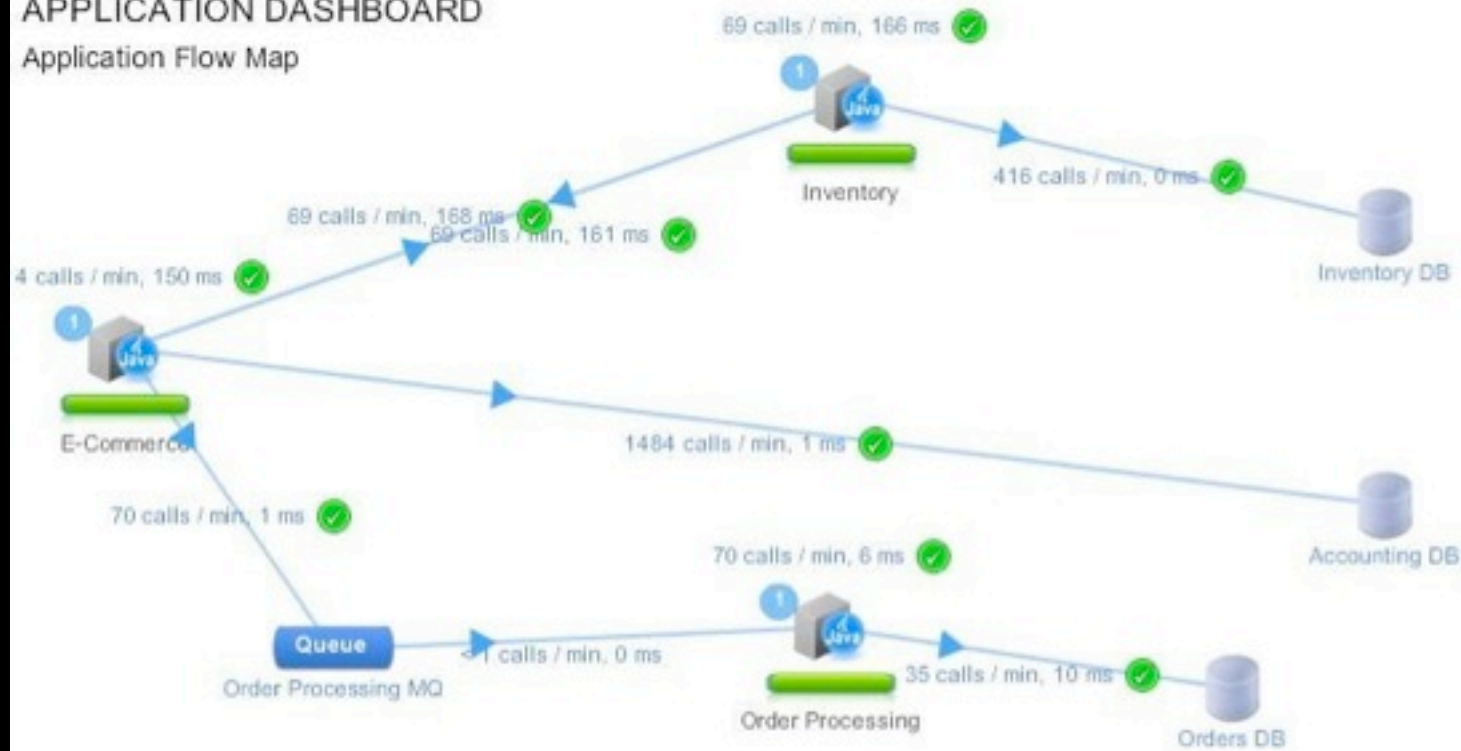


“End-User Metric



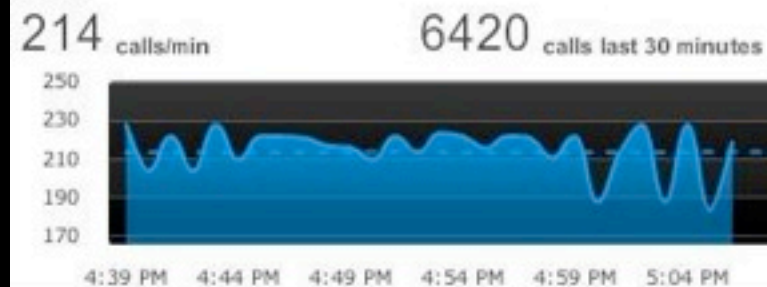
APPLICATION DASHBOARD

Application Flow Map

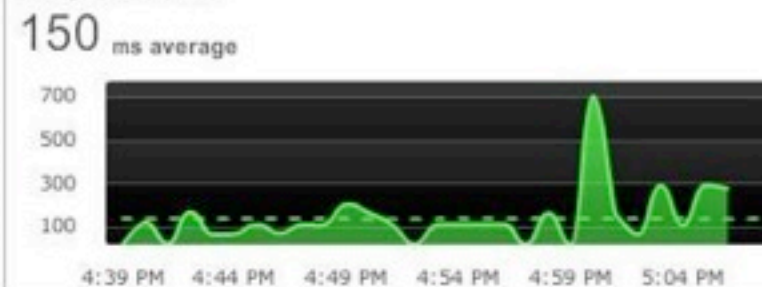


Explain this View

Load



Response Time



No Incidents in the time range

View All

Request Summary

View Request Snapshots

Help

Normal	<div></div>	99.2%	6371
Slow	<div></div>	0.0%	0
Very Slow	<div></div>	0.7%	45
Errors	<div></div>	0.1%	5
Stalls	<div></div>	0.1%	4

Configure Thresholds

Total Requests: 6420

Top Transactions by Load

View All

Name	calls/min	
GetAllItems	36	✓
Checkout	36	✓
Logout	36	✓
AddToCart	36	✓
Login	36	✓

Top Transactions by Resp. Time

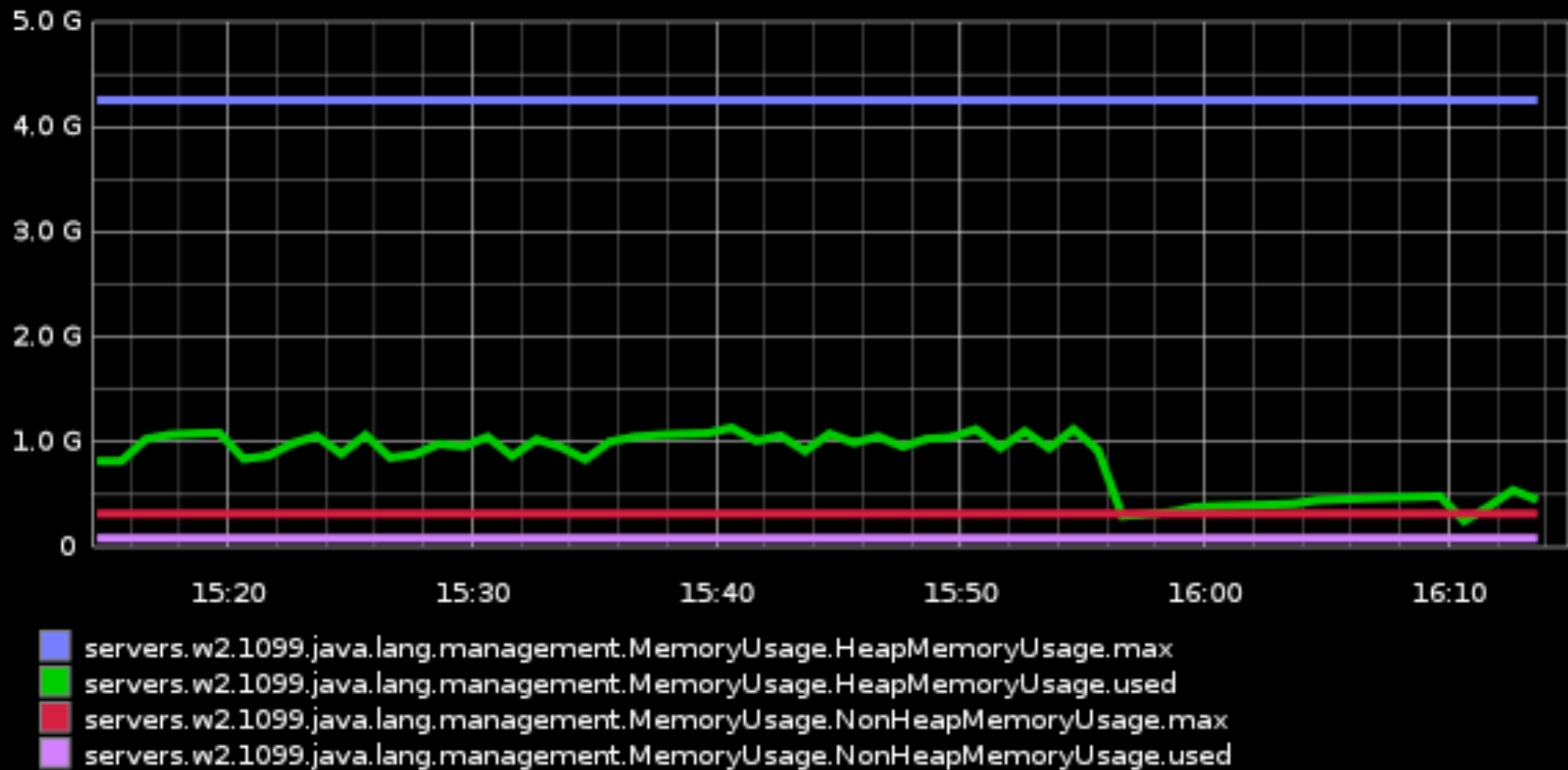
View All

Name	resp.time	
Checkout	360	✓
Logout	266	✓
AddToCart	234	✓
GetAllItems	19	✓
Login	14	✓

Brothers



<http://www.appdynamics.com/>



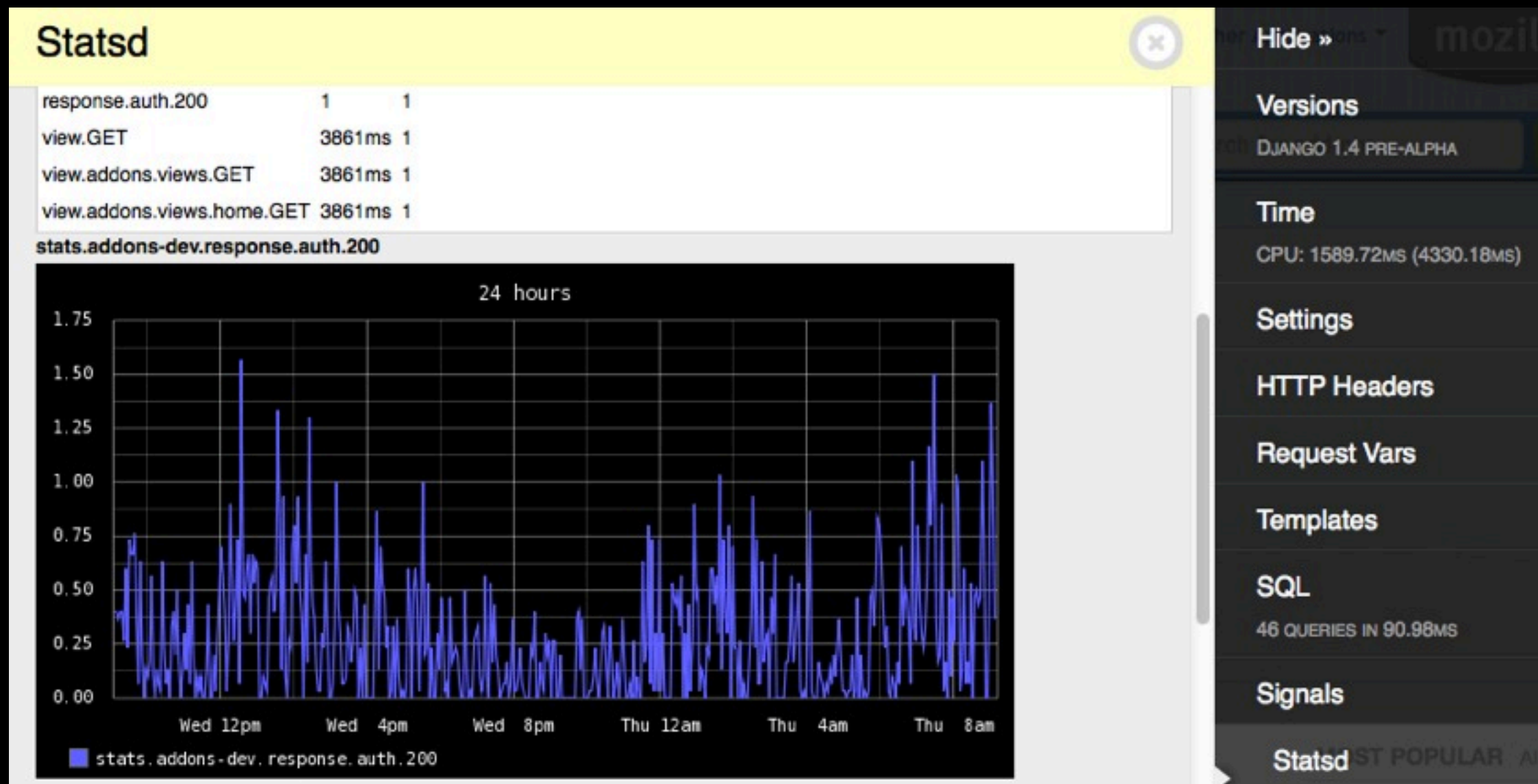
<https://github.com/lookfirst/jmxtrans>

APP

STATSD

counting
timing
sampling

GRAPHITE



<https://github.com/etsy/statsd>

Metrics Java Library

<http://metrics.codahale.com/>

<https://github.com/tritonrc/metricsd>

Business Level Metrics

Usage patterns - Google Analytics

A/B Testing in Prod

Metrics / Information

Meta-Ops Metrics

Know context

Unix Pipe Analogy

inputs

- amqp
- exec
- file
- gelf
- redis
- stdin
- stomp
- syslog
- tcp
- twitter
- xmpp
- zeromq

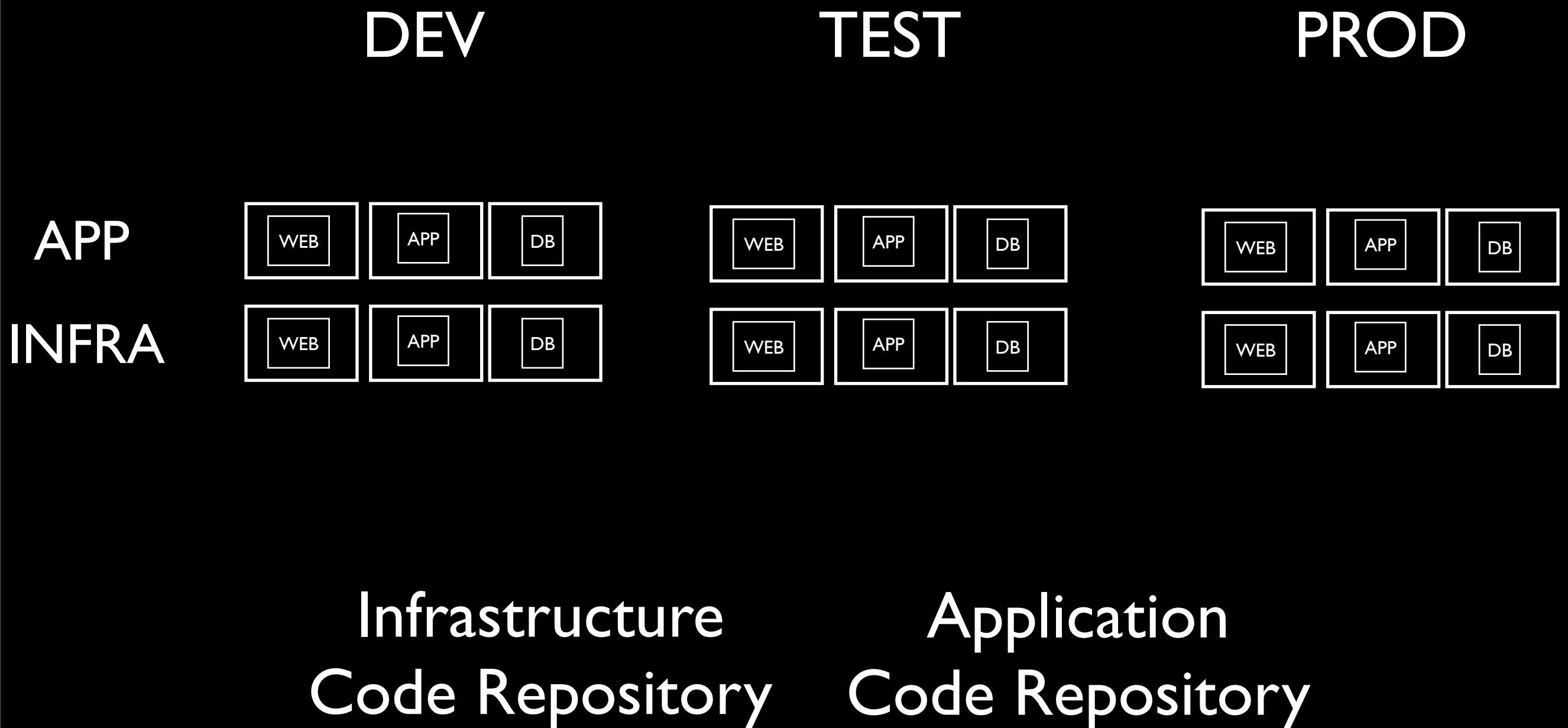
filters

- date
- dns
- gelfify
- grep
- grok
- grokdiscovery
- json
- multiline
- mutate
- split

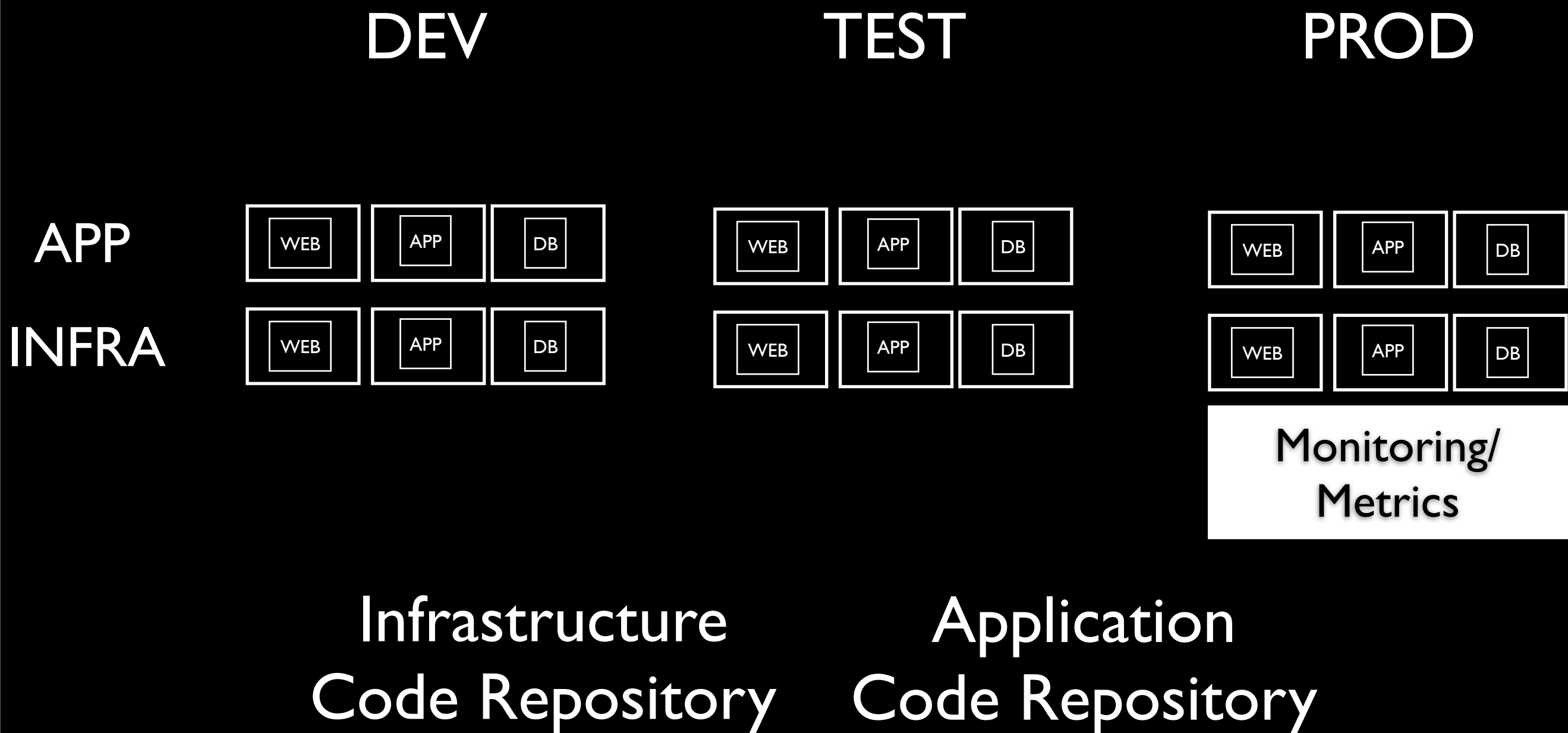
outputs

- amqp
- elasticsearch
- elasticsearch_river
- file
- ganglia
- gelf
- graphite
- internal
- loggly
- mongodb
- nagios
- null
- redis
- statsd
- stdout
- stomp
- tcp
- websocket
- xmpp
- zabbix
- zeromq

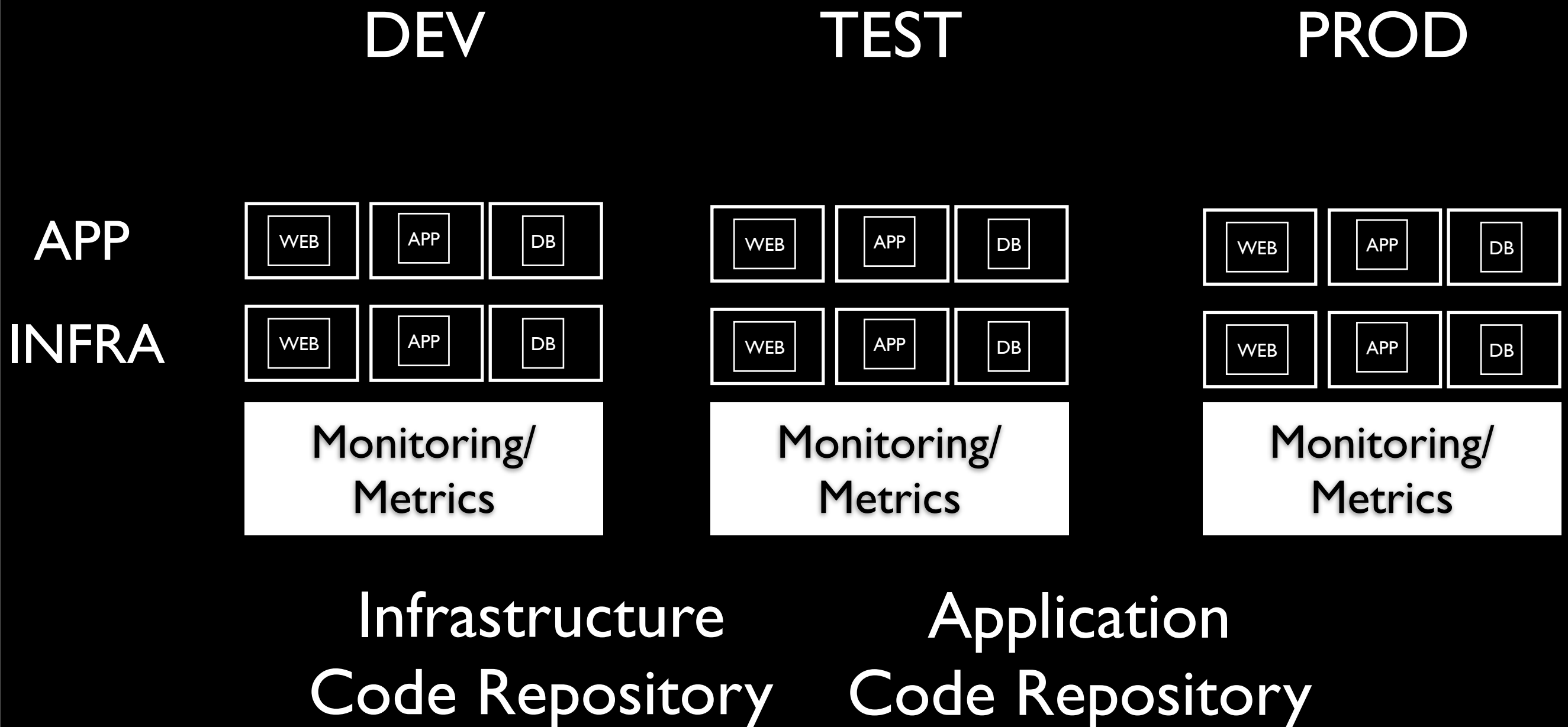
Integrate with **Continuous Integration**



Reuse Tools across dev & ops



Integrate with **Continuous Monitoring**



Reuse Workflow

Debugging

Post-Mortem

DEV

TEST

PROD

APP



INFRA



Monitoring/
Metrics

Monitoring/
Metrics

Monitoring/
Metrics

Metrics Driven Engineering

[http://www.slideshare.net/mikebrittain/metricsdriven-
engineering](http://www.slideshare.net/mikebrittain/metricsdriven-engineering)

Measure Anything, Measure Everything

Posted by Ian Malpass | Filed under [data](#), [engineering](#), [infrastructure](#)

If Engineering at Etsy has a religion, it's the Church of Graphs. If it moves, we track it. Sometimes we'll draw a graph of something that isn't moving yet, just in case it decides to make a run for it. In general, we tend to measure at three levels: network, machine, and application. (You can read more about our graphs in Mike's [Tracking Every Release](#) post.)

“The ~~future is~~ metrics are here
they are just not evenly
distributed yet”
- Morpheus Monitoring

<http://jedi.be/blog>
<https://github.com/monitoringsucks>

