Outlyer

Monitoring, done differently.

Why We Chose Erlang Over vs. Java, Scala, Go, and C

Colin Hemmings, CTO

@thegonzohunter

Outlyer

Monitoring, done differently.

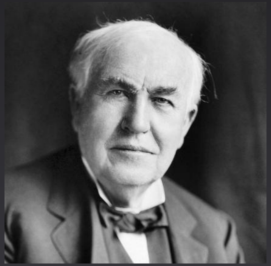
My massively opinionated overview of building software at a startup, with the aid of quotes and funny pictures.

Colin Hemmings

@thegonzohunter

“I have not failed, I’ve just found 10,000 ways that won’t work”

Thomas Edison



Outlyer

“I know words, I have the best words. I have the best, but there is no better word than stupid”

Donald J. Trump

Outlyer

Dashboards
Search for dashboard
All Systems
Apache2
Couchbase
Docker
Elasticsearch
IIS Servers
Memcache
MongoDB
MySQL
Nginx
Raspberry PI
Redis
US Presidential Election
Varnish
dynamodb
foo
All Systems
Last hour
Annotations
Add widget
server status
max swap 0%
max disk / 76%
max memory 51%
max cpu 100%
fractional load average
packs4
packs1
packs2
packs3
centos7
cpu
download
upload

Who are we?

Formerly known as...

Dataloop.IO

Now...

Outlyer

Our Vision: Self-service monitoring for Microservices.

Teams own their monitoring and collect metrics they need, not just Operations

Any stakeholder, business or technical, has visibility without being overwhelmed by data

While ops teams still maintain control & visibility across the entire platform

Outlyer

Monitoring

- Replace Nagios, Graphite, Statsd
- Open standards
- Docker support

NOT SURE IF NOTHING IS BROKEN
OR EVERYTHING IS BROKEN AND NO ONE CAN REACH ME
quickmeme.com

Outlyer

Building for a Startup

Time is always important, but in a startup it’s critical

- Too slow to release? You’re dead (speed)
- Too slow to product/market fit? You’re dead (flexibility)
- Too slow to scale? You’re dead (strategy)
- Never enough time

Outlyer

Unless...

- You are well funded
- A lifestyle business
- You are incredibly lucky
- even then...

Business First

- Can’t build in stealth
- Need to build what’s going to sell
- Sometimes quality takes a backseat

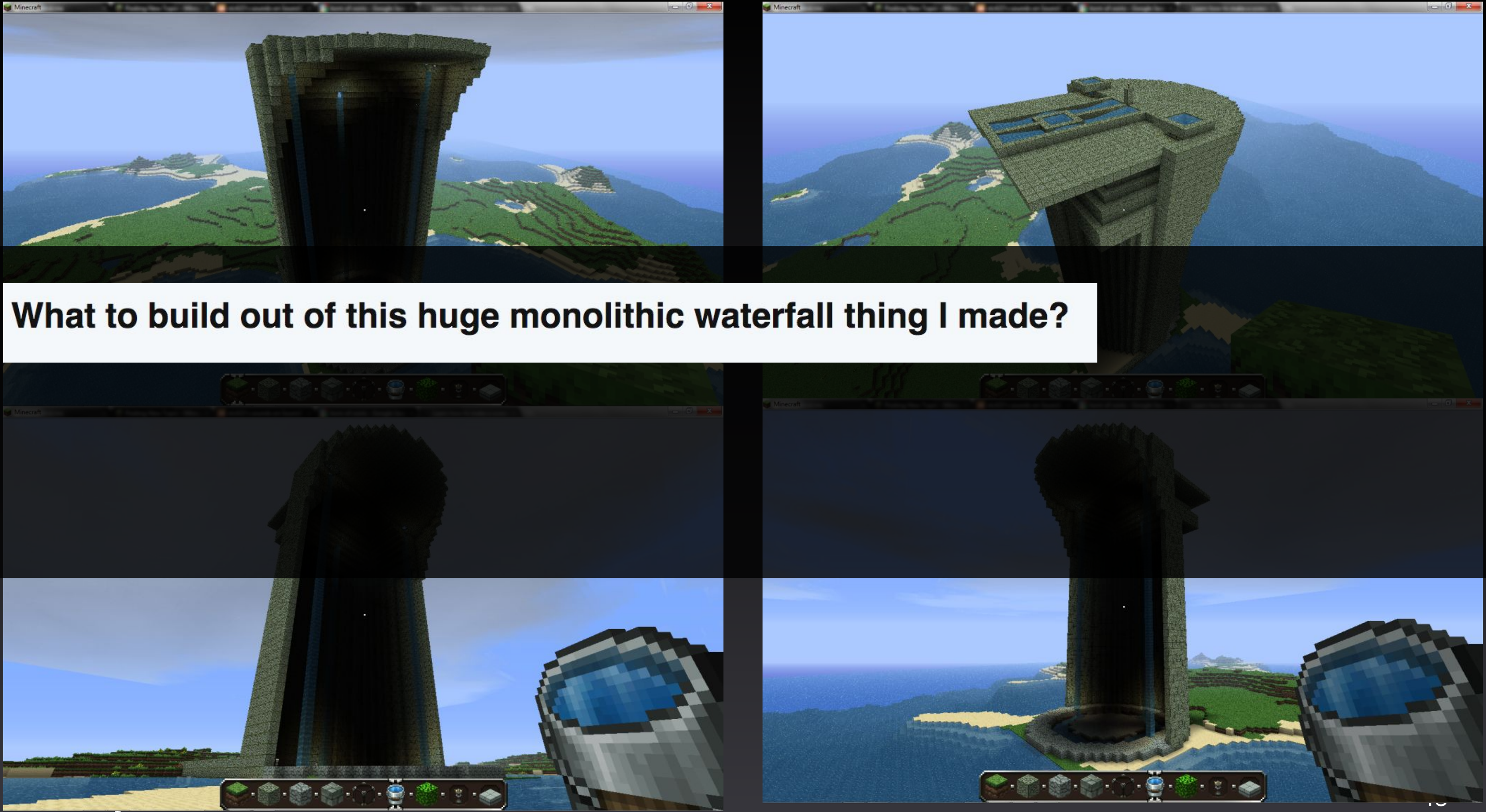
V1: The monolith

Minecraft

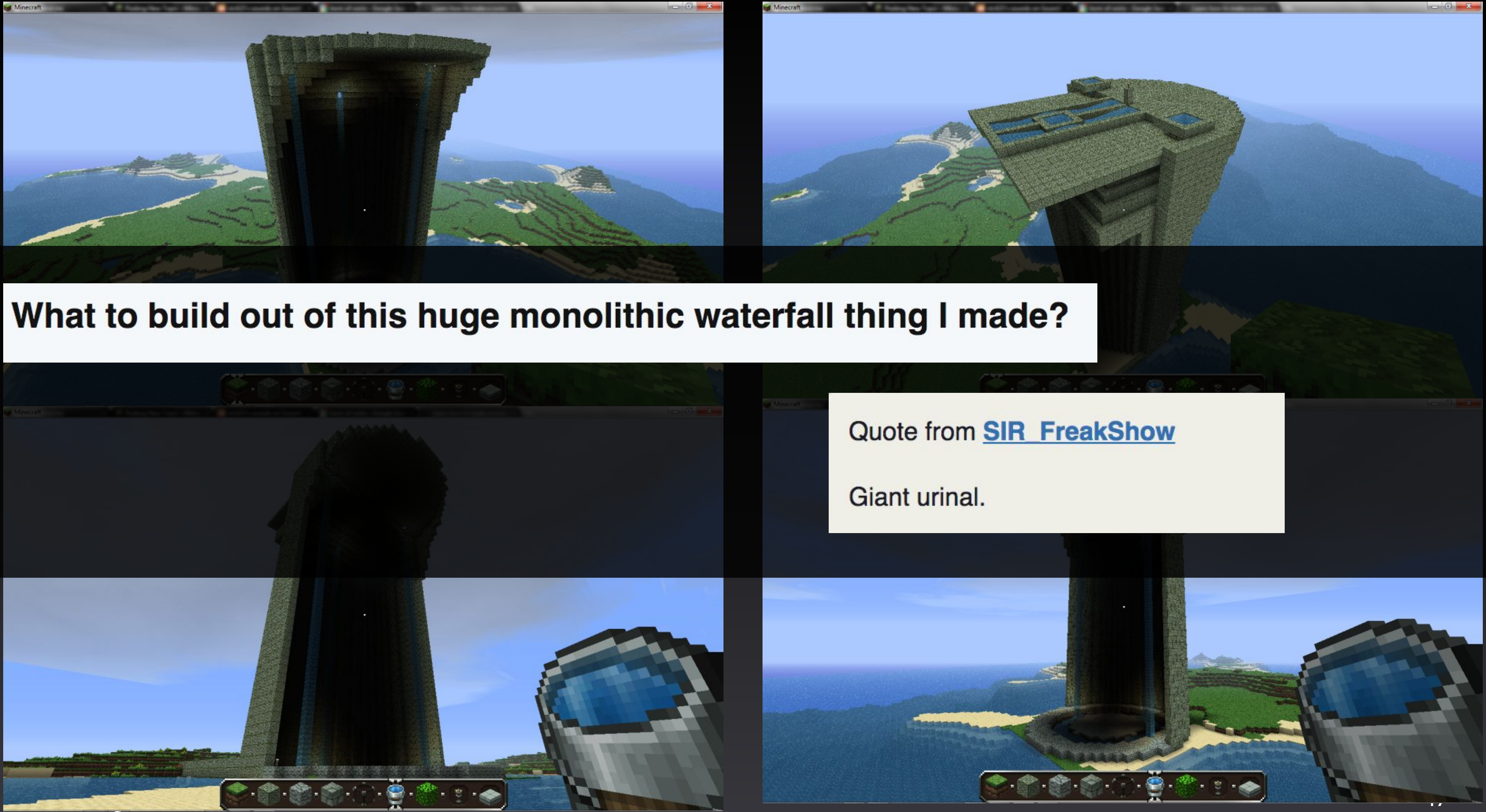
Minecraft

Minecraft

Minecraft



What to build out of this huge monolithic waterfall thing I made?



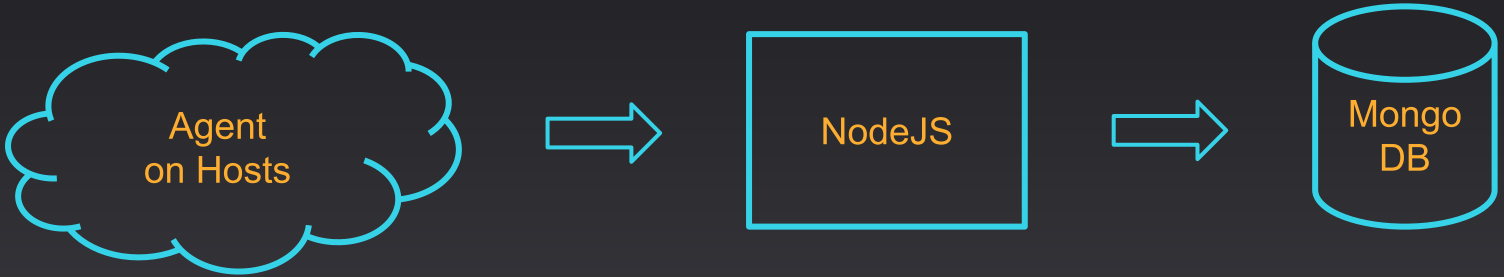
What to build out of this huge monolithic waterfall thing I made?

Quote from **SIR_FreakShow**

Giant urinal.

The Monolith

- Like most companies, we started with a Monolith
- Python Agent, NodeJS App, Mongo DB
- 2 Nodes for HA



Outlyer

Growing the Monolith

- How was our architecture going to grow with the company
- Vision
  - Microservices?
- Process
  - We can’t stop to rewrite

Microservices

Benefits

- Isolation
- Independently deployable
- Flexibility (languages, tech etc)
- Team alignment and ownership

Impediments

- Requires infrastructure
- Packaging, deployment, hosting
- Communication: bus, http
- Service discovery

Outlyer

The process: Structured Monolith

- Break up the code
- Separate state
- Separate services
- Separate deployment

Separating the State

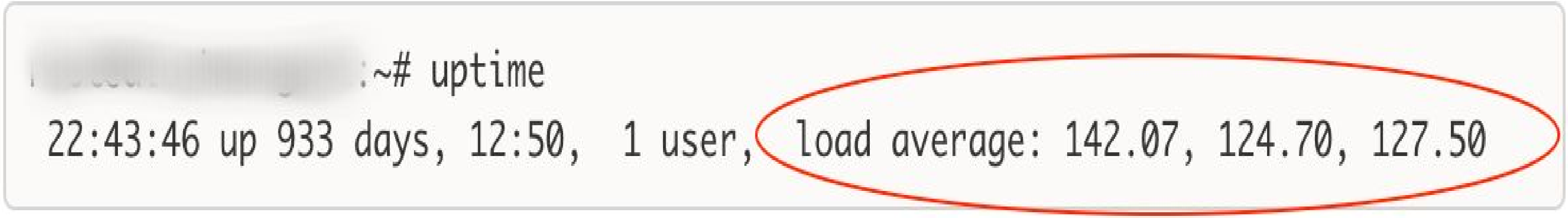
- The early step
- Separate infrastructure services
  - DB, Queues
- Monitor resource contention

Our Dirty Secret

- We use mongo
- And its awesome
- Just keeps chugging

Our Dirty Secret

- We use mongo
- And its awesome
- Just keeps chugging

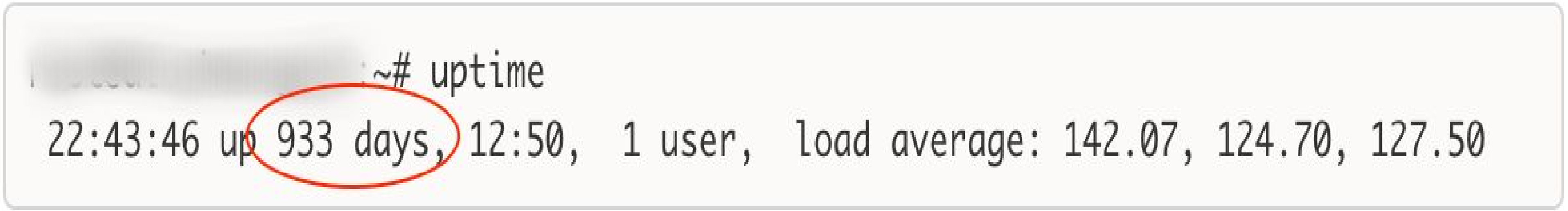


```
[redacted]:~# uptime
 22:43:46 up 933 days, 12:50,  1 user,  load average: 142.07, 124.70, 127.50
```

Outlyer

Our Dirty Secret

- We use mongo
- And its awesome
- Just keeps chugging



```
[redacted]:~# uptime
22:43:46 up 933 days, 12:50,  1 user,  load average: 142.07, 124.70, 127.50
```

Outlyer

Scale state

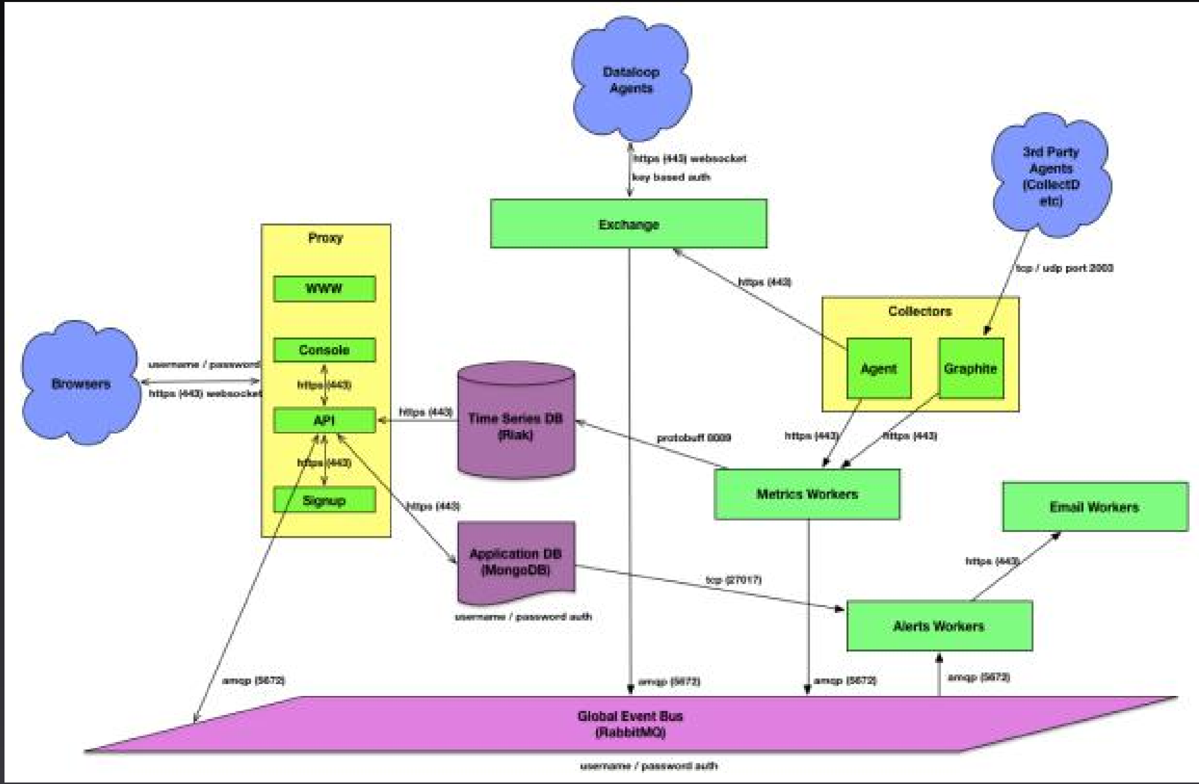
- Remove it where possible
  - Scale horizontally
- Reliable cluster for HA
- Shard
  - Performance
  - Optimise first
  - Geolocate data
  - Data protection

V2: Microservices

Outlyer

Splitting Services

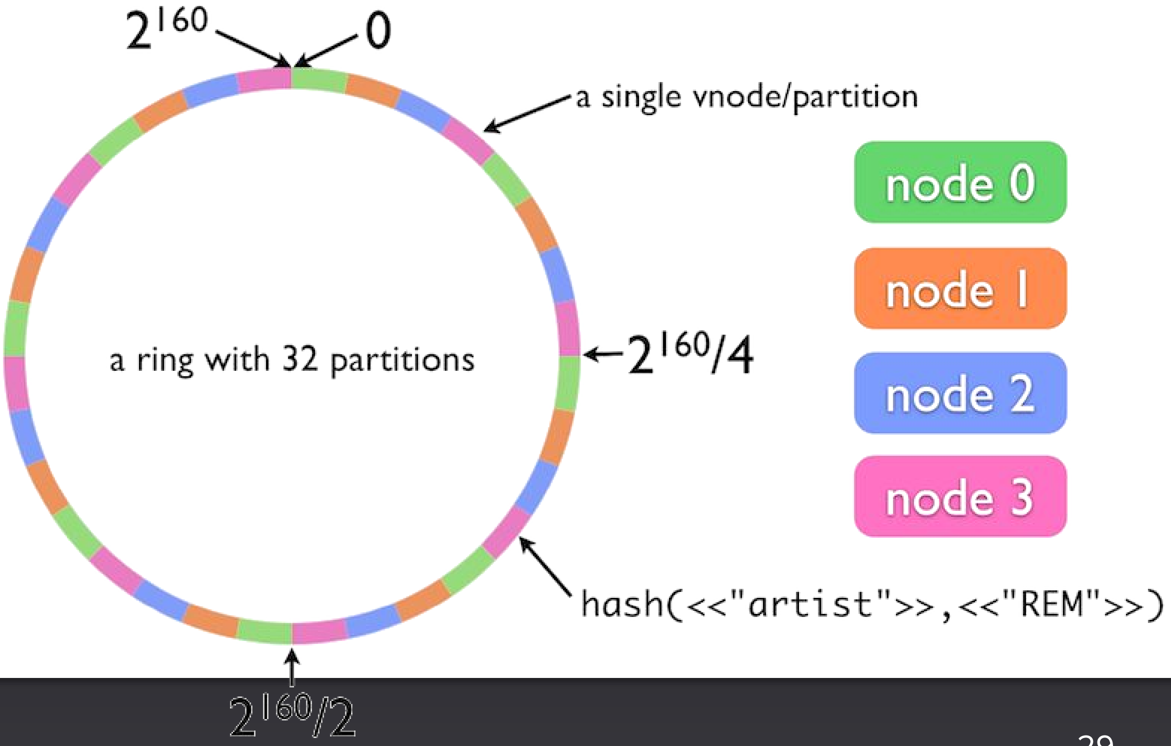
- Gradual processes
- Areas of responsibility
- Background workers
- Data model entities
- Central Comms Bus



Outlyer

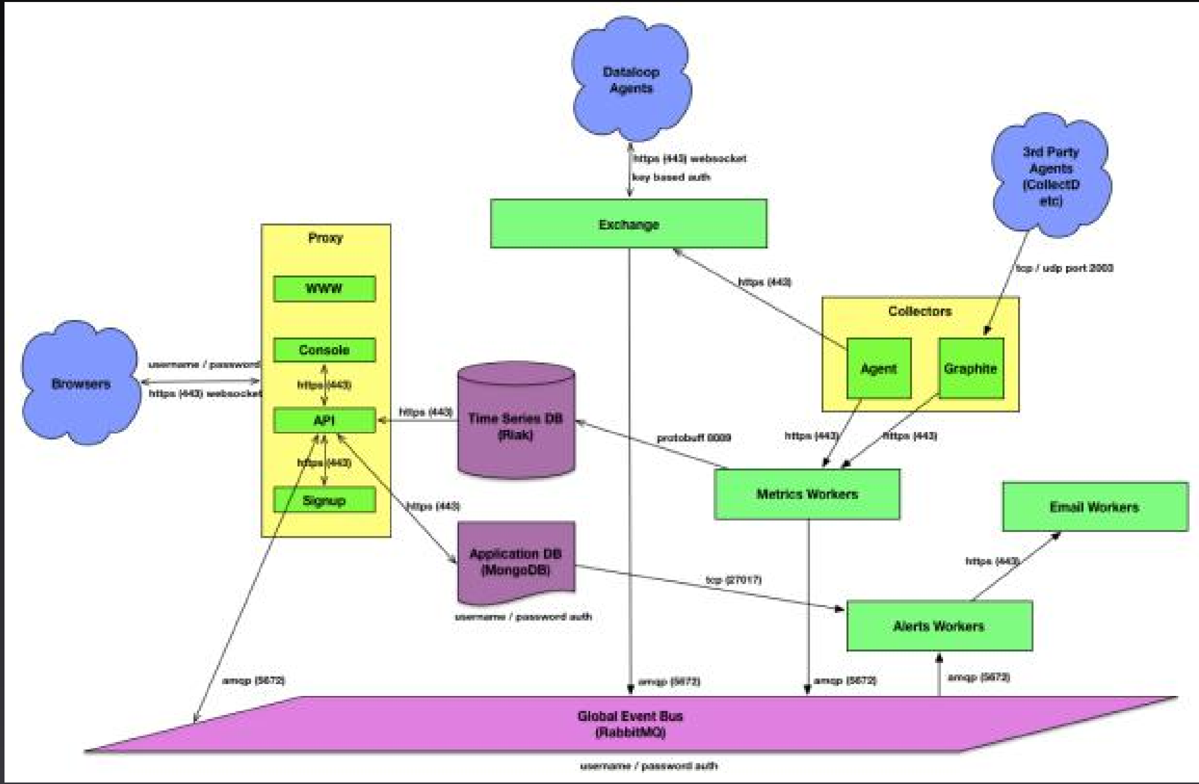
Time series: Riak

- Key/Value Store
- Dynamo Paper
- Hashed Distribution
- Ops Friendly
- Erlang



We were rocking

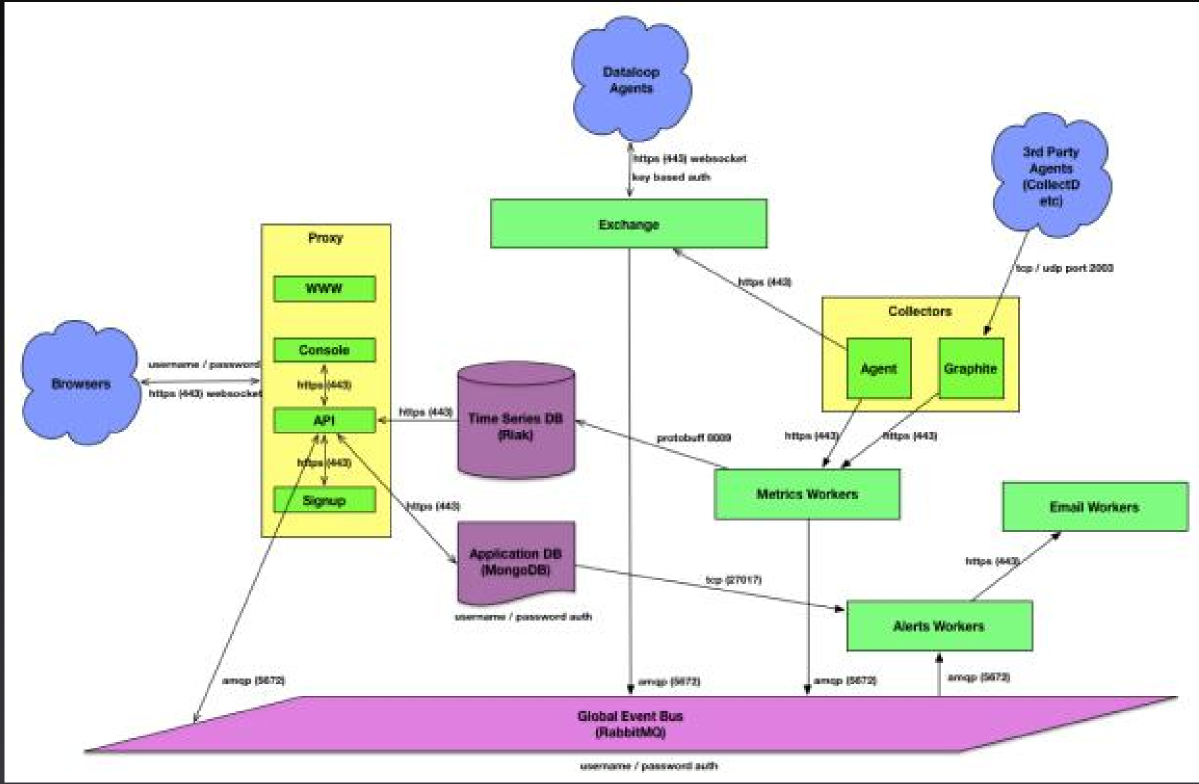
- Scaled well at first



Outlyer

We were rocking

- Scaled well at first
- Onboarded 2 large customers



Outlyer

We were rocking

- Scaled well at first
- Onboarded 2 large customers
- Constant fire fighting

Outlyer

Redis to the Rescue

- NodeJS couldn’t keep up
- We added a redis buffer for processing
- Reduced load on Node.JS workers
- Bought us some time
- A more permanent solution needed
- Redis is also awesome

The Erlang Story

Outlyer

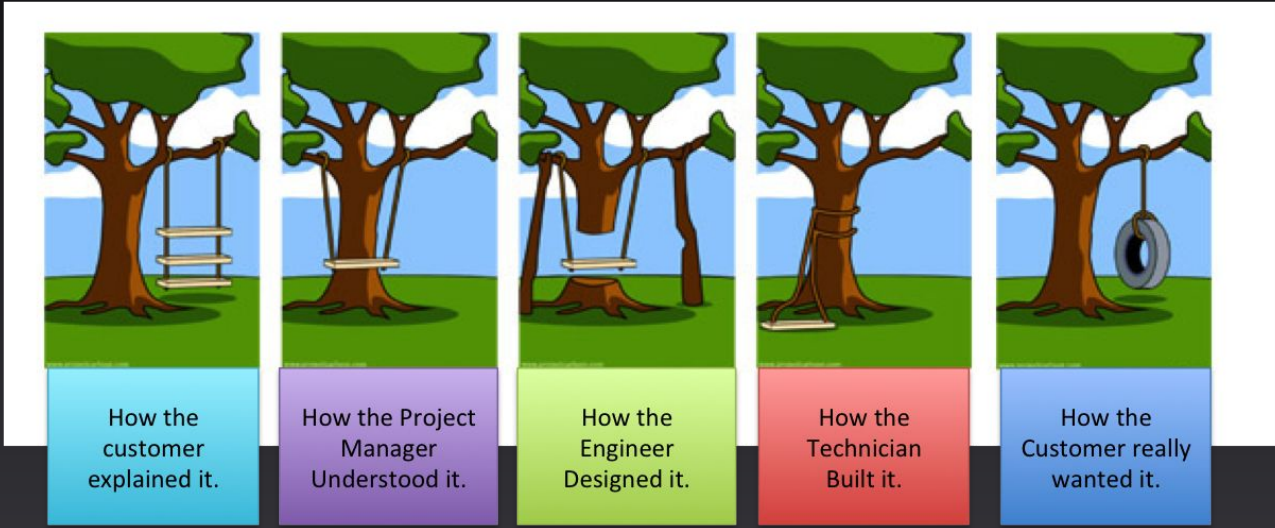
“Any sufficiently complicated concurrent program in another language contains an ad hoc informally-specified bug-ridden slow implementation of half of Erlang”

Robert Virding

Outlyer

Requirements

- Distributed for scale and HA
- Reliable
- Good performance
- Production visibility


How the customer explained it.
How the Project Manager Understood it.
How the Engineer Designed it.
How the Technician Built it.
How the Customer really wanted it.

Outlyer

Option 1 - Java

- Forced-OOP
- Very verbose and inexpressive
- GC: “that way madness lies”
- Team motivation

Outlyer

“Object-oriented programming is an exceptionally bad idea which could only have originated in California.”

Edsger Dijkstra

Option 2 - Scala

- Still the JVM
- Concurrency model
- Runtime tracing and visibility
- Missing OTP, although AKKA

Option 3 - Go

- Growing community
- Performs well
- Runtime visibility
- Fault tolerance

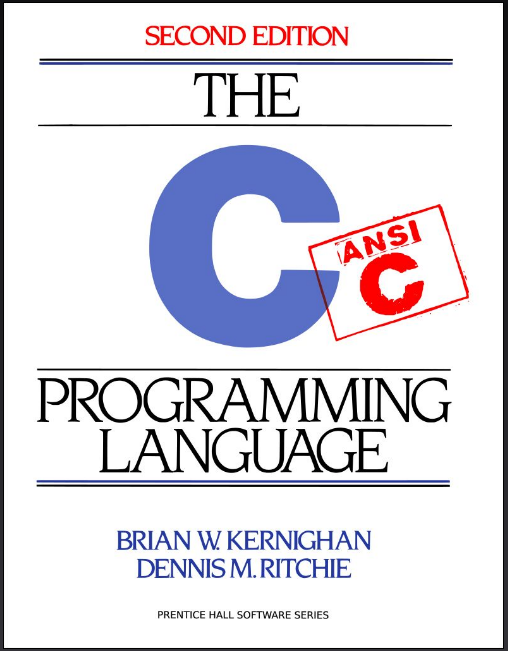
Outlyer

Option 4 - C

- Good for performance critical components
  - Not much else
- Life is just too short
- Looking at Rust

“A C program is like a fast dance on a newly waxed dance floor by people carrying razors”

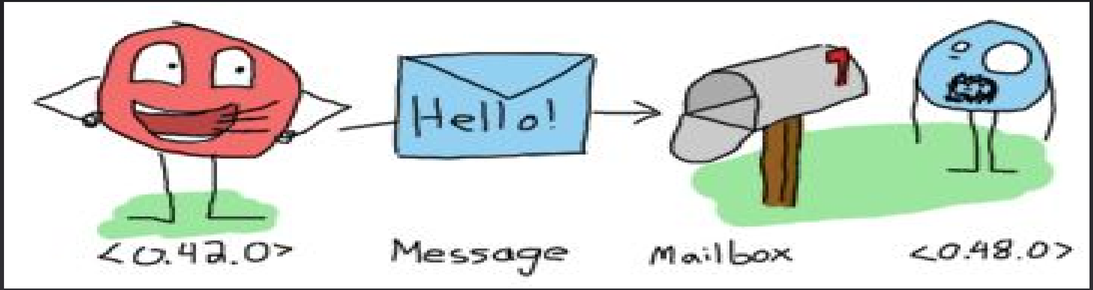
Waldi Ravens



Outlyer

Erlang: State and Concurrency

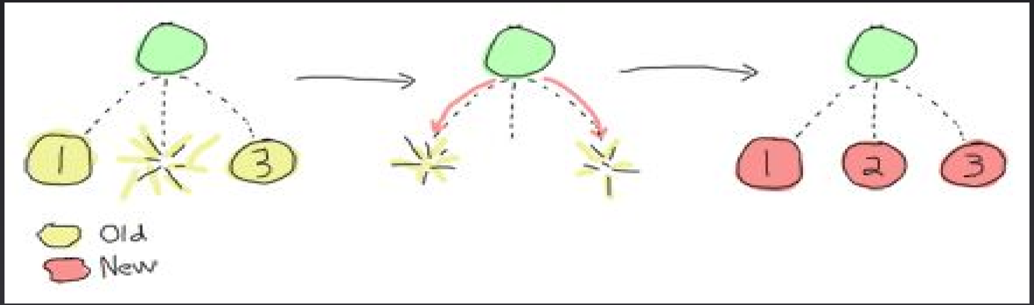
- No shared state
- Actor model
- Communication via messages
- Like human interaction



Outlyer

Erlang: Supervisors

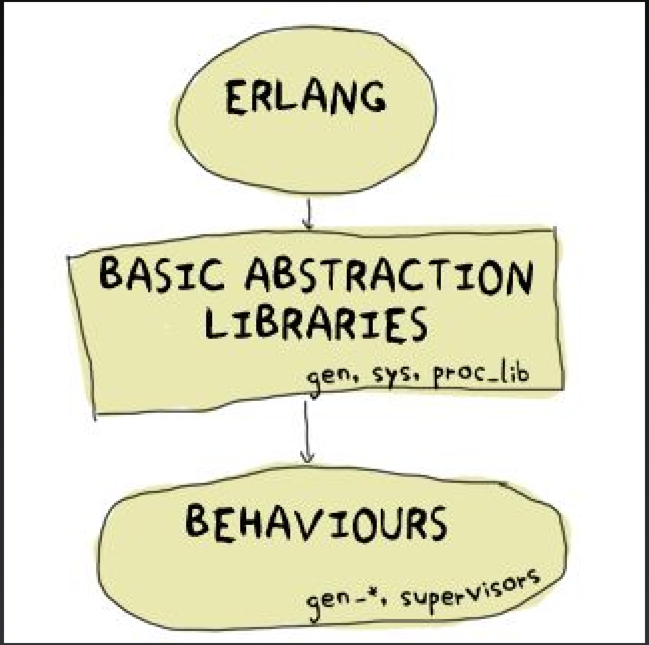
- Supervise worker processes
- Restart upon crash
- Fault-tolerance
- Independently configurable
- Different layers of protection



Outlyer

Erlang: Behaviours

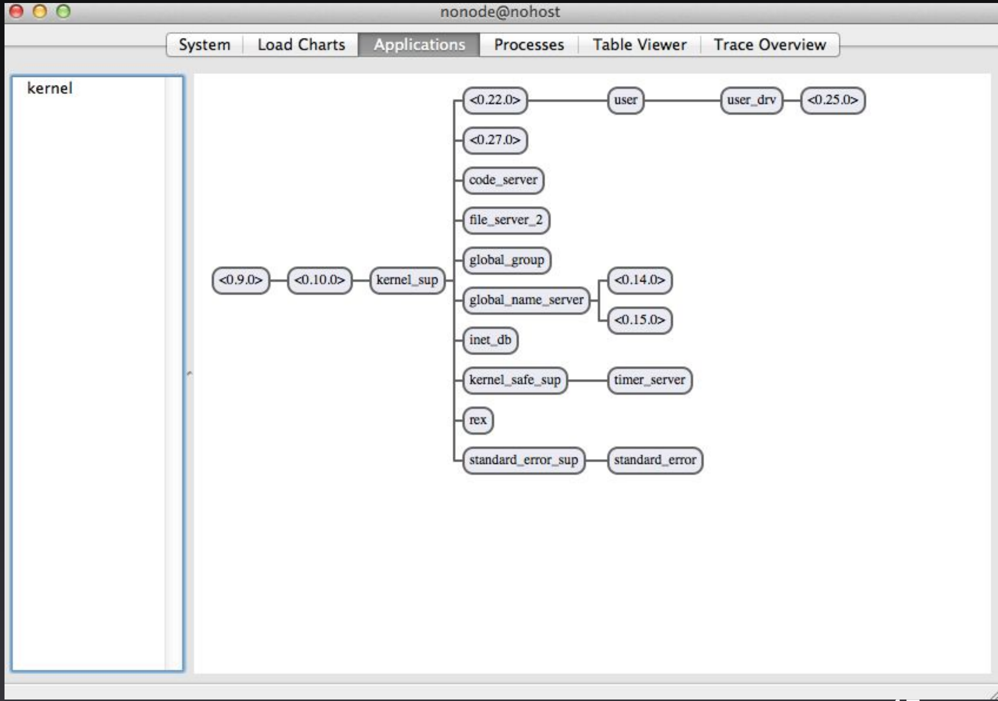
- Formalized common patterns
- Battle hardened
- Reliable systems made easy
- Prevent race conditions
- Handle partial failures



Outlyer

Erlang: Visibility

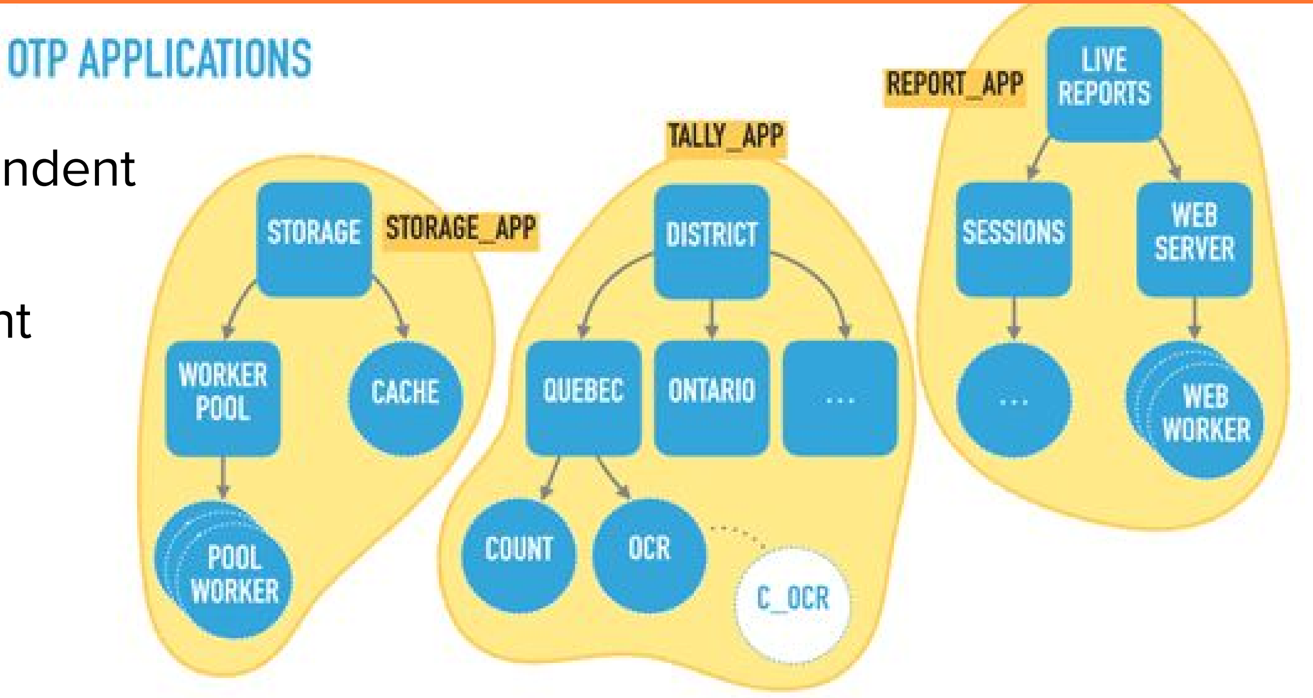
- Simple to inspect live systems
- Even via GUI and shell
- Supports interaction
    - Sending messages



Outlyer

Erlang: Applications as Microservices

- BEAM like an OS
- Applications independent Start/Stop
- Different deployment configurations
- Like a standalone Microservice



Erlang: The rest

More Good

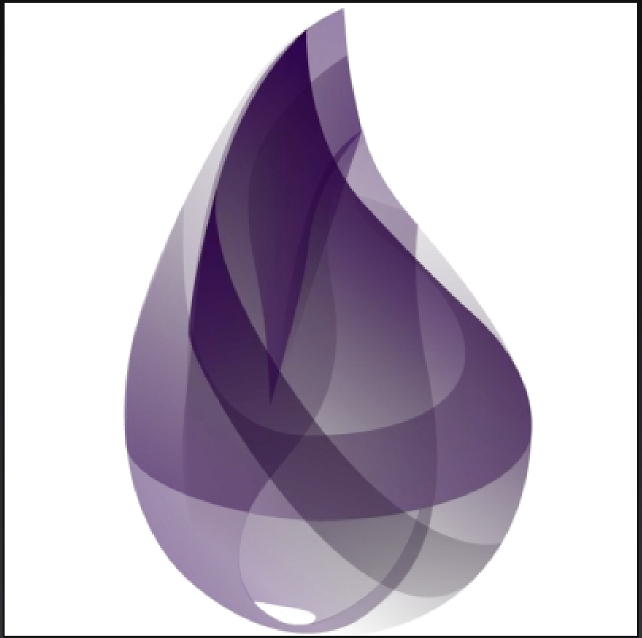
- Distribution at its core
- Live code reload

Bad

- Quirky Syntax
- Smaller community
- Hiring (sort of)

Elixir

- New kid on the block
- All the power of Erlang/OTP BEAM
- Ruby sugar, for the kids
- Massively growing community



Outlyer

Scalable Time Series

Outlyer

Dalmatiner DB

- Open Source Time-Series DB
- Written in Erlang
- Based on Riak-Core and uses ZFS
- Fast as F***
- Built on simple solid components
- https://dalmatiner.io/

Outlyer

How it works

ERLANG

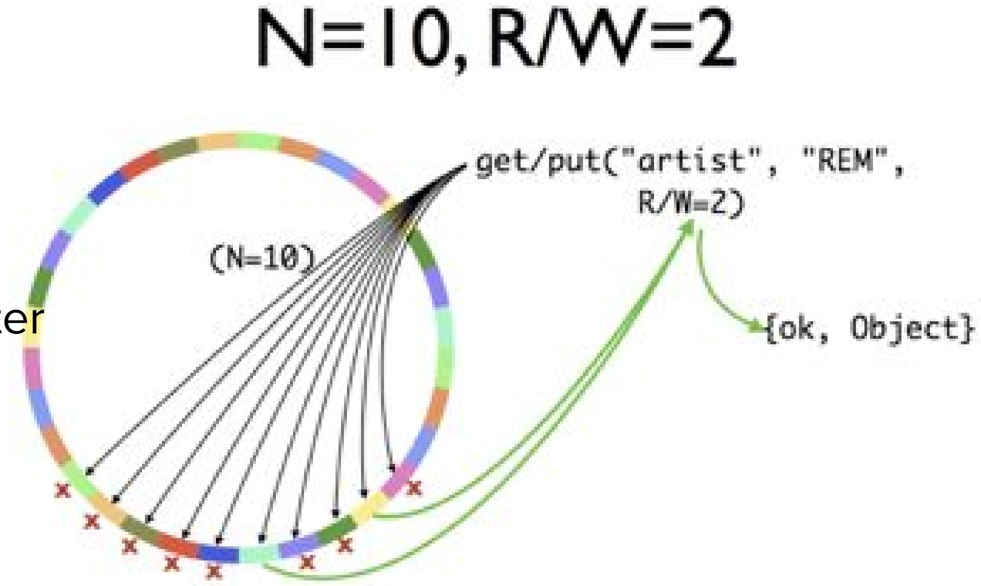
riak CORE

ZFS ON LINUX

PostgreSQL

Riak: As previously mentioned

- Ring distribution
- vnode
  - manages key partition
  - distributed across physical cluster
  - transfers (scaling)
  - redundancy
- masterless



Riak Core

- The base to riak
- Designed as a distributed systems framework
- Services for:
  - Node Liveness & Membership
  - Partitioning & distribution (consistent hashing)
  - Cluster state
- Ops tools for managing the cluster
  - add/remove nodes
  - monitor state

Benchmarks

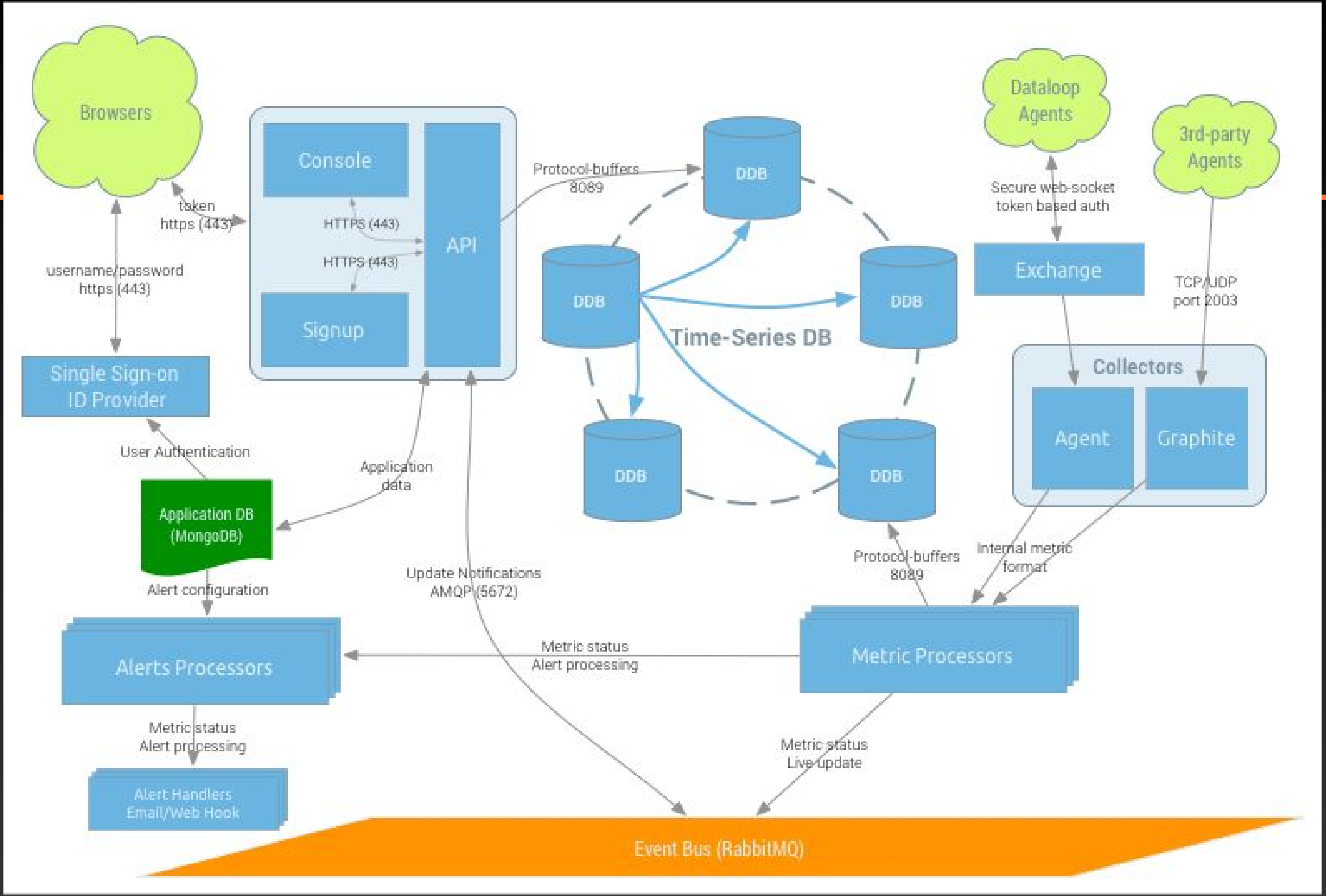
- Haggar (Graphite benchmarking tool)
- Cluster test
  - 5 nodes (8core 60GB RAM, 1TB SSD)
  - 15 - 20 million unique metrics / second!!!
- Single node test
  - 3 - 4 million unique metrics / second
- http://blog.outlyer.com/top10-open-source-time-series-databases

Outlyer

Trade offs

- No strict guarantees
- Configure to reduce probability of data loss
- Memory usage, caches, ZFS
- Optimisations for read load
- Floating point numbers (62bit)
- Proprietary binary protocol

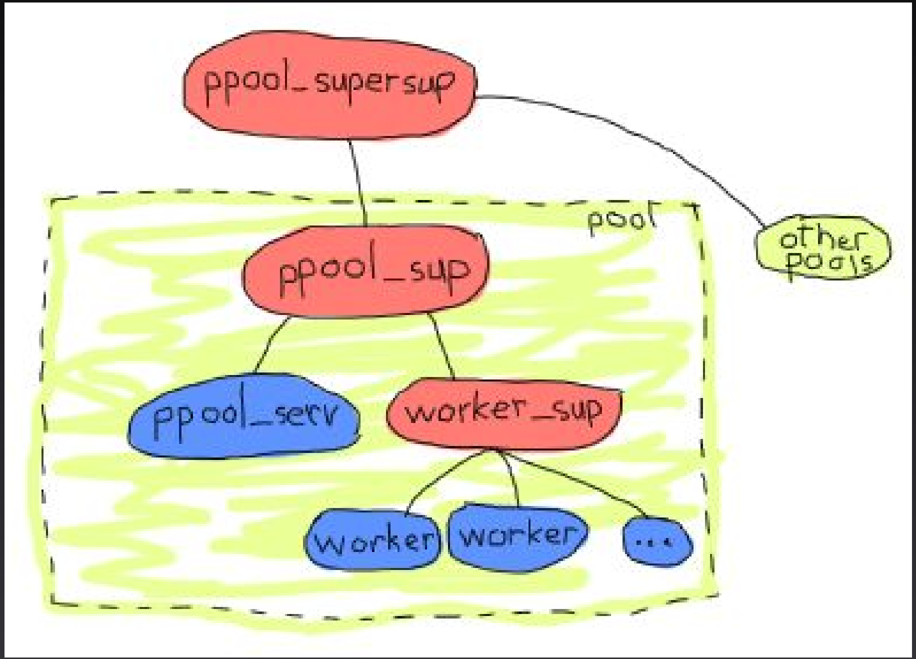
Now (roughly)



Outlyer

What we've learnt

- Live code tracing
- Let it crash approach
  - Supervision
  - Saves code
- Community
  - Small but passionate
  - A lot of academic work
  - Erlang solutions



Outlyer

What’s Next

- Erlang all the things
- Elixir
- More live code reloading

Lessons from a small startup

- Ignore the dogma
  - Make it work
  - Make it fast
  - Shard the hell out of it
- Be scrappy
  - Pay down your debts
- Don’t rush to microservices
  - or any shiny things

“In theory, theory and practice are the same. In practice, they’re not.”

Outlyer

Introducing

Outlyer

Monitoring, done differently.