

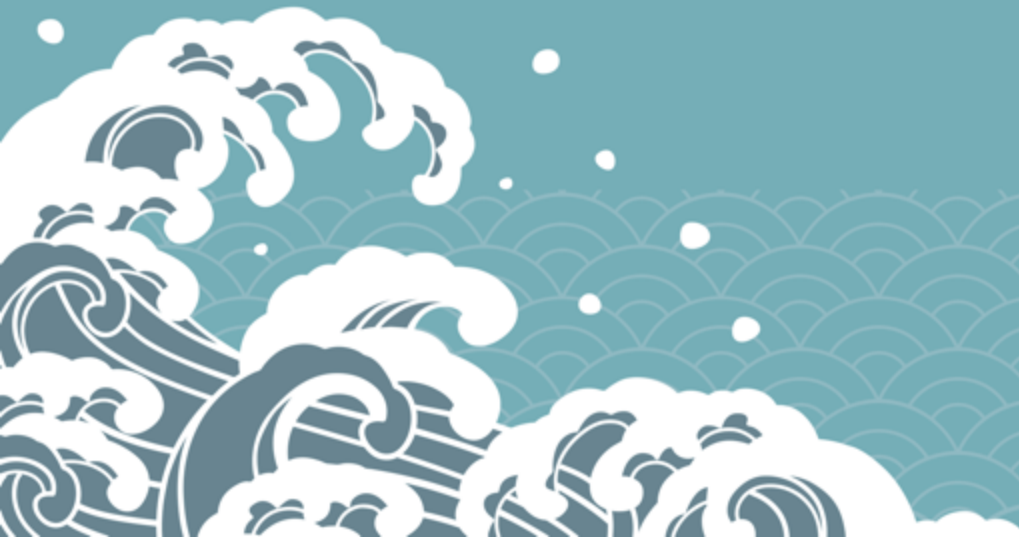


Concept to Reality: Solving Enterprise Challenges

June 2015

Jawad Yaqub

Enterprise Architect, EMEA
Basho Technologies



Our Focus



Introduction

In this talk we'll cover:

Design

How to design an enterprise architecture built on core erlang technology.

Challenges

Look at the challenges with implementing new code alongside existing (legacy) systems.

Decisions

Discuss necessary tradeoffs and design decisions in building best practice platforms.

Production

How to test and deploy your enterprise platform into a Production environment.



Just the Facts

Facts of Life (for Architects)

- Why Big Data?
- Why Erlang?
- Realtime + Analytics = Competitive
- Riak KV as a Foundation
- Redis for Low Latency
- Spark for Intelligent Analytics
- System Overview
- Building the System
- Testing, Staging, Releasing

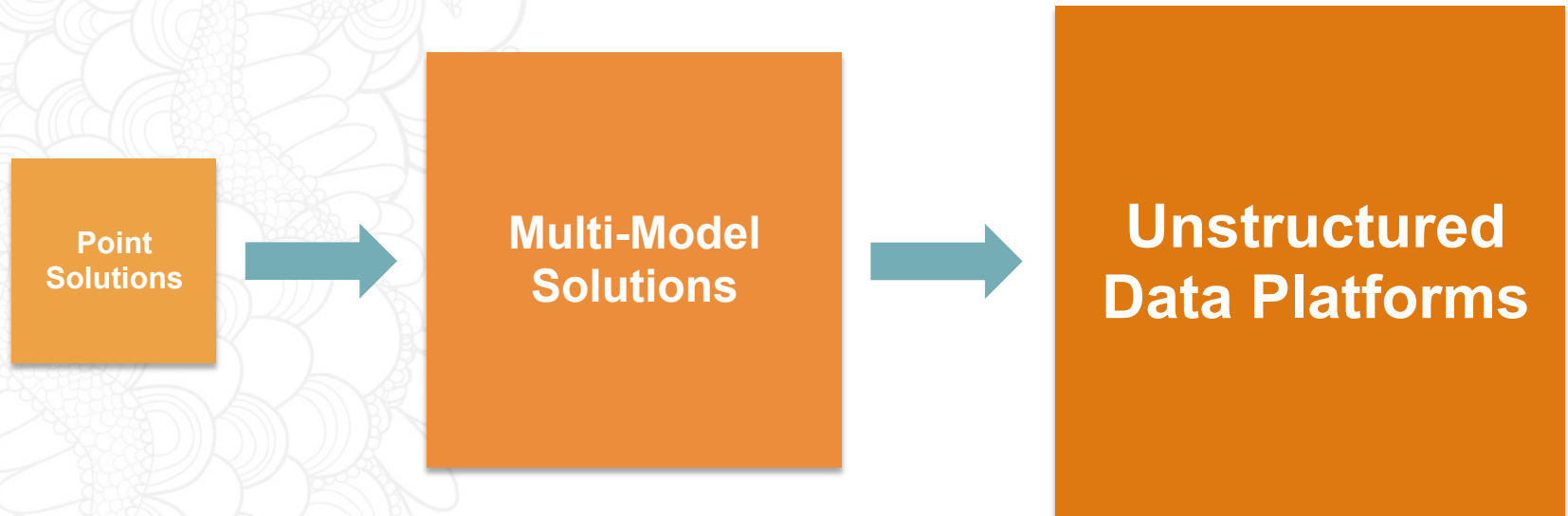
Why Big Data

1. Big Data can unlock significant value by making information transparent. There is still a significant amount of information that is not yet captured in digital form, e.g., data that are on paper, or not made easily accessible and searchable through networks.
2. Organisations create and store more transactional data in digital form to collect more accurate and detailed performance information on everything from product inventories to sick days and therefore expose variability and boost performance.
3. Big Data allows ever-narrower segmentation of customers and therefore much more precisely tailored products or services.
4. Sophisticated analytics can substantially improve decision-making, minimise risks, and unearth valuable insights that would otherwise remain hidden.
5. Big Data can be used to develop the next generation of products and services. For instance, manufacturers are using data obtained from sensors embedded in products to create innovative after-sales service offerings such as proactive maintenance to avoid failures in new products.

Why Big Data

1. Big Data can unlock significant value by making information transparent
2. Organisations create and store more transactional data in digital
3. Big Data allows ever-narrower segmentation
4. Sophisticated analytics can substantially improve decision-making
5. Big Data can be used to develop the next generation of products and services

The Evolution of NoSQL



Why Erlang?

Erlang was developed at Ericsson and was **designed from the ground up** for writing **scalable, fault-tolerant, distributed, non-stop, soft-realtime applications**. Everything in the language, runtime and libraries reflects that purpose, which makes Erlang the **best platform** for developing this kind of software.

We use Erlang as we want our platform to:

- handle very large number of concurrent activities
- be easily distributable over a network of computers
- be fault-tolerant to both software & hardware errors
- scale with the number of machines on the network
- be upgradable & reconfigurable without having to stop & restart
- be responsive to users within certain strict timeframes
- stay in continuous operation for many years

Because Erlang is oriented around concurrency, it's also **naturally good** at utilizing modern multicore systems.

Why Erlang?

Designed from the ground up for writing **scalable, fault-tolerant, distributed, non-stop, soft-realtime** applications

handles very large number of concurrent activities

easily distributable over a network of computers

fault-tolerant to both software & hardware errors

scale with the number of machines on the network

upgradable & reconfigurable without having to stop & restart

responsive to users within certain strict timeframes

stay in continuous operation for many years

it's also **naturally good** at utilizing modern multicore systems

Realtime + Analytics = Competitive

Faced with exploding data volumes and shrinking decision windows, organizations struggle to gain a competitive advantage



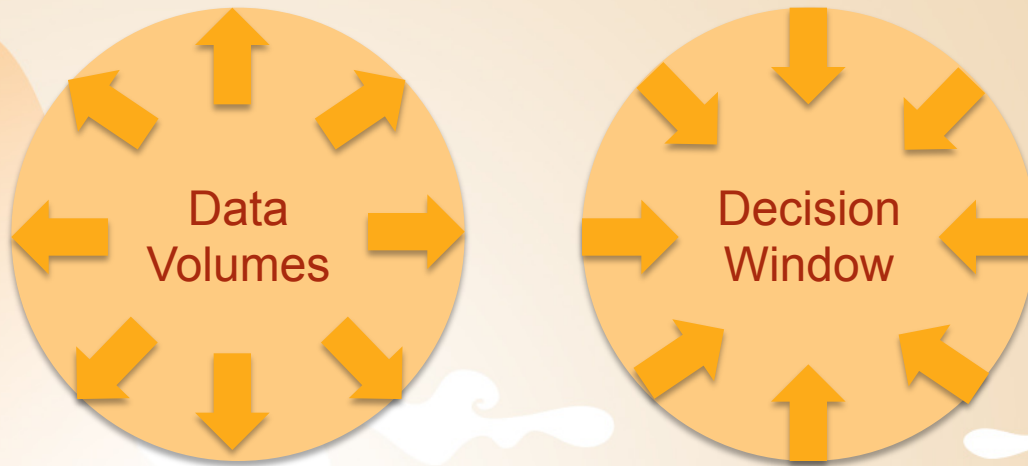
Next on the horizon for Big Data is the demand for real-time analytics.

These are not terms that normally go together.

Traditionally, big data analytics is Batch Analytics. It takes hours to run to create the information, then to do the analysis, then to create value (or useful information).

But the demand is now on for “time-to-value”.

Realtime + Analytics = Competitive



Next on the horizon for Big Data is the demand for real-time analytics

Riak as a Foundation



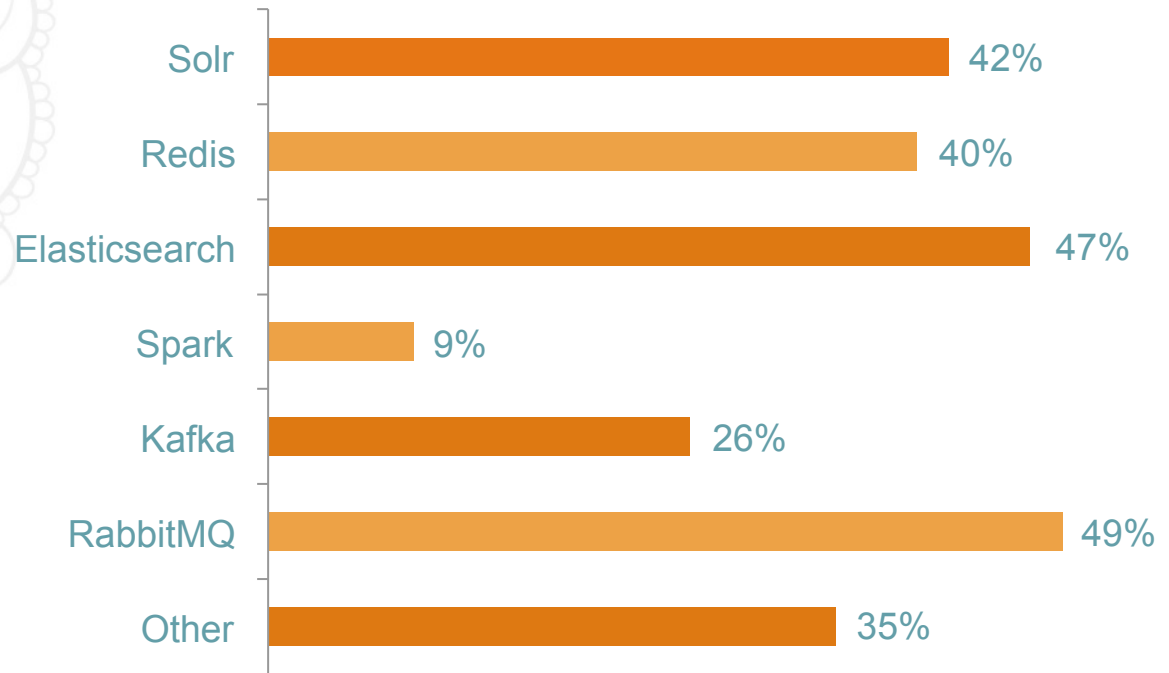
- Simple to Operate
- Highly Available
- Fault Tolerant
- Horizontally Scalable
- Low Latency

Customers are Integrating

NoSQL, Caching, Real-time Analytics and Search

51% of Basho customers are using Riak for Multiple Applications

Which of the following additional products do you use in your Big Data applications? (Select all that apply)



Note: This is a multiple-choice question – response percentages may not add up to 100

Redis for Low Latency

Redis is an open source, BSD licensed, advanced key-value cache and store. It is often referred to as a data structure server since keys can contain strings, hashes, lists, sets, sorted sets, bitmaps and hyperlogs.



redis

- Key/Value Cache
- In Memory Dataset
- Transaction Support
- TTL Keys
- Automatic Failover

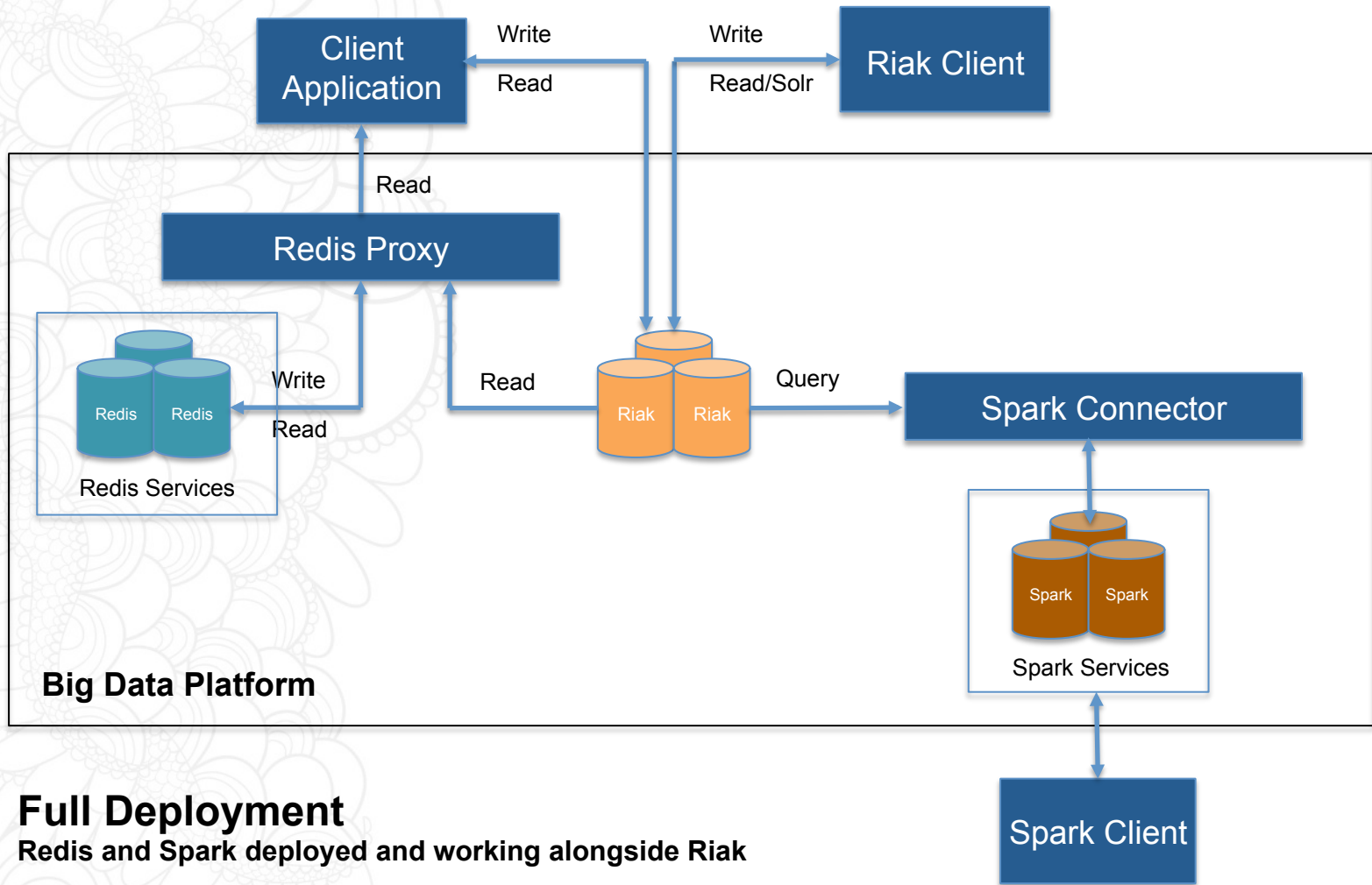
Spark for Analytics

Apache Spark is a powerful open source A.I engine for data built around speed, ease of use, and sophisticated scaling.



- 100x faster than Hadoop
- Scalable to 8k+ nodes
- Spark Streaming Analytics
- Machine Learning Library
- SQL support for semi-structured Data
- GraphX processing library

System Overview





Let's Get Building



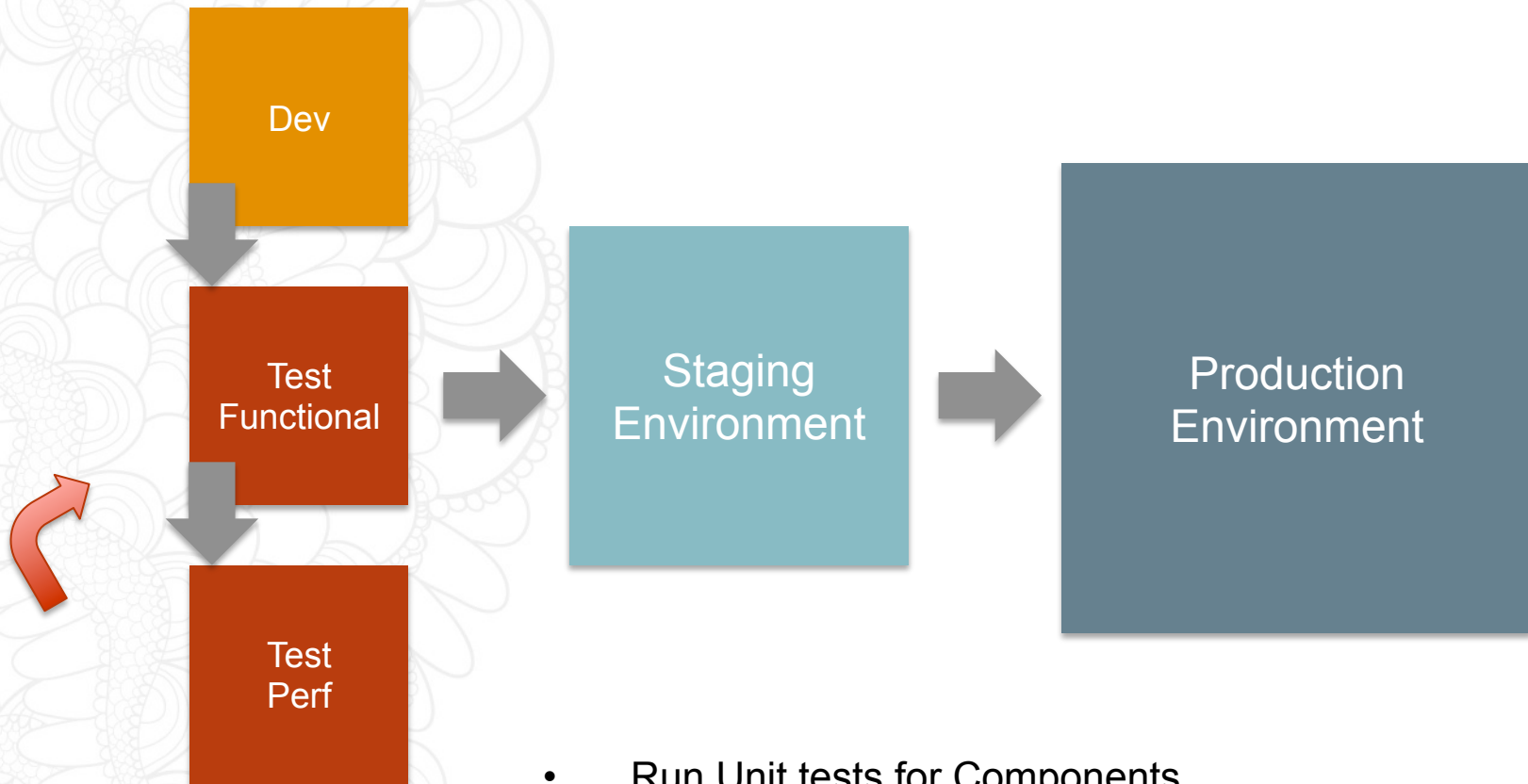
Building the System

- Install Riak [<http://docs.basho.com/riak/latest/quickstart/>]
- Install Redis [<http://redis.io/topics/quickstart>]
- Install Spark [<https://spark.apache.org/docs/latest/quick-start.html>]
- Clients for Connection [<http://docs.basho.com/riak/latest/dev/using/libraries/>]
- Implement Usecase [<http://www.informationweek.com/big-data/big-data-analytics/apache-spark-3-promising-use-cases/a/d-id/1319660>]
- Add SOA [<http://www.sitepoint.com/soa-for-the-little-guys/>]
- Add Presentation Layer [Web/Mobile Framework]
- Test, Stage, Release.

Testing Staging Release

- Dev locally
- Use Github (or equivalent)
- Test [Regression]
- Test [Performance]
- Staging Environment
- Production Environment

Environments



- Run Unit tests for Components
- Run Platform Unit Tests
- Test in Staging, then release

Problems?



Problems

You might be thinking...

That's a lot to learn!

How much
will it cost?

How secure is it?

How do I scale?

I can't manage it
on my own



Coming Soon



BASHO DATA PLATFORM

SERVICE
INSTANCES



Spark

Redis (Caching)

Solr

Web Services

Elastic
Search

3rd Party Web
Services &
Integrations

CORE SERVICES

Replication &
Synchronization

Message
Routing

Cluster Management
& Monitoring

Logging &
Analytics

Internal Data Store

STORAGE
INSTANCES



Riak
Key/Value

Riak Object
Storage

Riak
Coming Soon

Document Store

Columnar

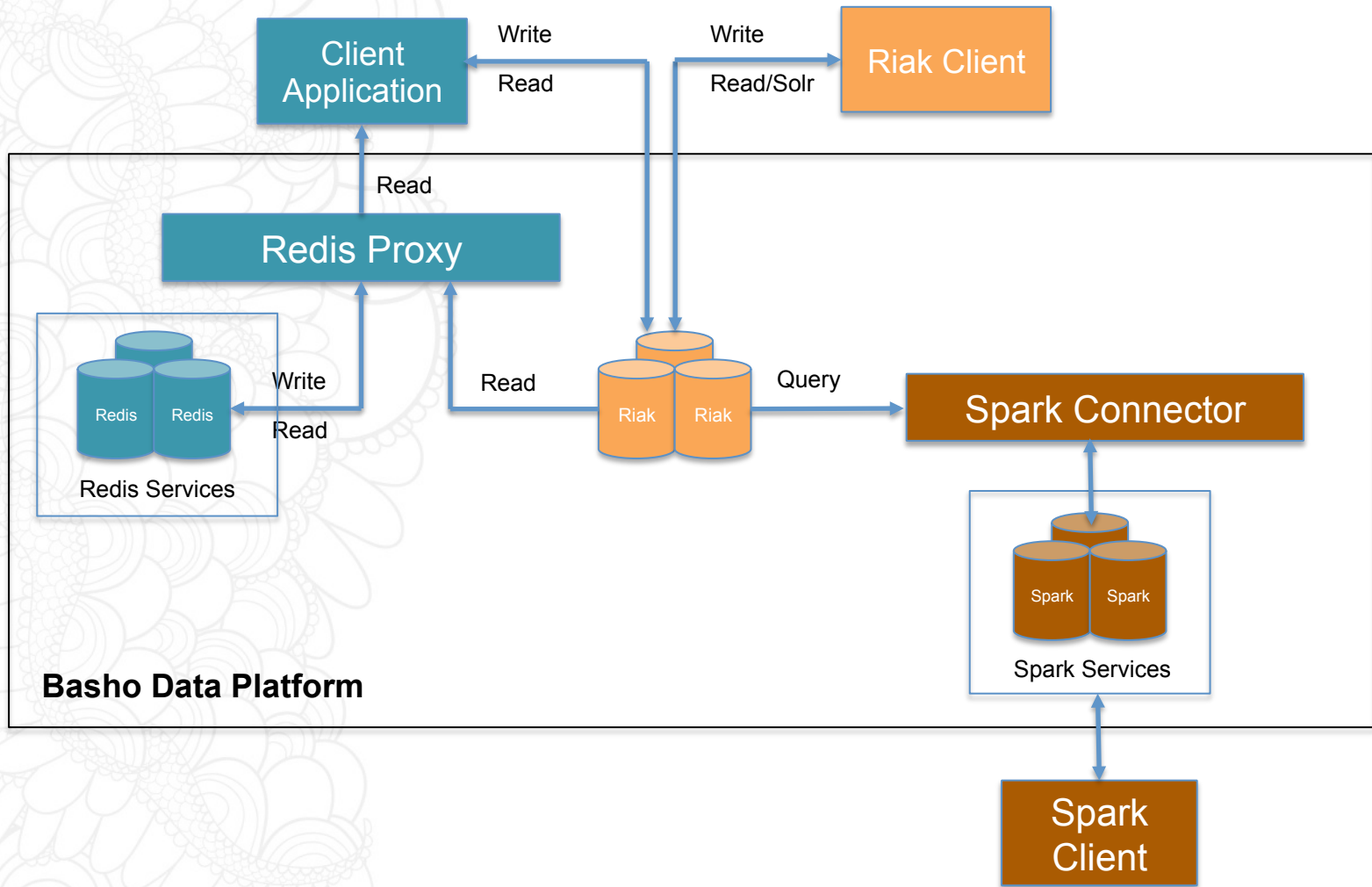
Graph

BASHO DEVELOPED

BASHO INTEGRATED

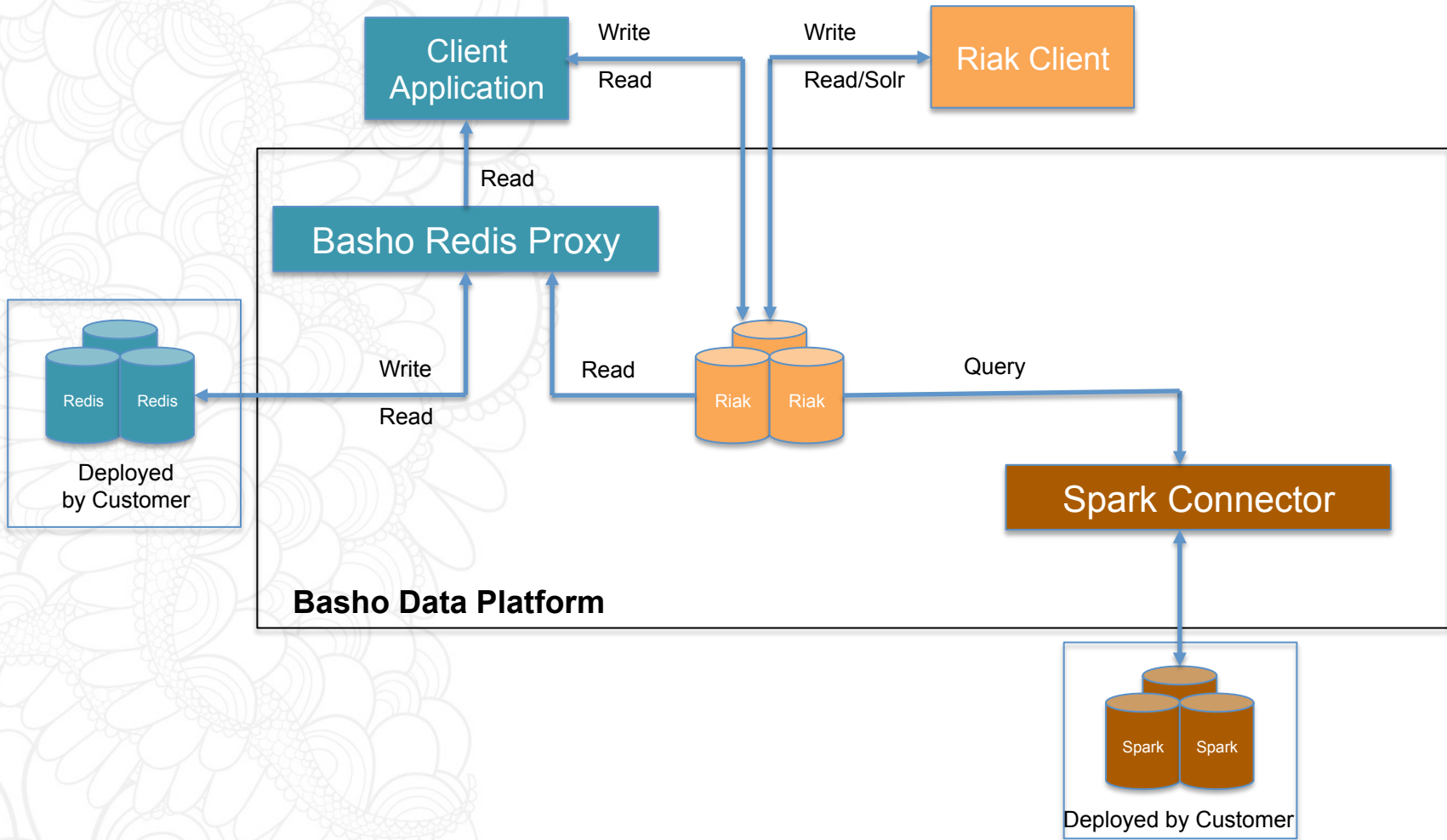
BASHO DATA PLATFORM

Full Deployment - Redis and Spark deployed and managed by Basho Data Platform



BASHO DATA PLATFORM

Hybrid Deployment - Redis and Spark deployed and managed separately by Customer



BASHO DATA PLATFORM

Core Services

Data Replication and Synchronization

Replicate and synchronize data across and between storage instances and service instances to ensure data accuracy with no data loss and high availability.

Cluster Management

Integrated cluster management automates deployment and configuration of Riak KV, Riak S2, Spark and Redis. Once deployed in production, auto-detect issues and restart Redis instances or Spark clusters. Cluster management eliminates the need for Zookeeper.

Internal Data Store

A built-in, distributed data store for ensuring speed, fault-tolerance and ease-of-operations is used to persist static and dynamic configuration data (port number and IP address) across the Basho Data Platform.

Message Routing

A high-throughput, distributed message system for speed, scalability and high availability. This message system will have the ability to persist and route messages across platform clusters.

Logging and Analytics

Event logs provide valuable information that can facilitate the enhanced tuning of clusters and accurately analyze dataflow across the cluster.

APACHE SPARK ADD-ON

Write it like RIAK. Analyze it like Spark.

Cluster Management

Manage Spark clusters at scale using built-in leader election enabled by the Spark Connector for Riak.

Data Mover

Intelligently load data into Spark clusters to minimize network traffic and processing overhead.

Write-Back to Riak

Store intermediate and final results back into Riak KV for further processing by Spark or other Big Data application components.

Perform at Scale

Apache Spark is architected for high performance, real-time analysis and persistence of Big Data.

Operational Simplicity

Quickly deploy and configure Spark clusters with Riak KV. Auto-start failed Spark instances to reduce manual operations.

REDIS ADD-ON

Write it like Riak. Cache it like Redis.

High Availability

Using the Basho Data Proxy, the high performance caching capabilities of Redis become highly available, too.

Fast Cache

Combining the speed of Redis with the power of Riak KV provides low-latency, high performance at scale.

Automatic Data Synchronization

Data is automatically synchronized between Redis and Riak KV, and the Basho Data Proxy resolves cache misses without requiring custom code to populate the cache.

Automatic Sharding

Labor-intensive, error-prone manual sharding is a thing of the past with automatic data sharding across multiple cache servers.

Automated Deployment

Easily deploy and configure Redis instances with Riak KV and auto-start failed instances or disable on failure to reduce manual processes.

APACHE SOLR ADD-ON

Write it like Riak. Query it like Solr.

Distributed Full-Text Search

Standard full-text Solr queries automatically expand into distributed search queries for a complete result set across instances.

Ad-Hoc Query Support

Broad support for Solr query parameters, e.g., exact match, range queries, and/or/not, sorting, pagination, scoring, ranking, etc.

Index Synchronization

Data is automatically synchronized between Riak KV and Solr using intelligent monitoring to detect changes, and propagates those to Solr indexes.

Solr API Support

Query data in Riak KV using existing Solr APIs

Auto-Restart

Monitor Solr OS processes continuously and automatically start or restart them whenever failures are detected.

Conclusion



Conclusion

Simplifies Big Data, IoT and Hybrid Cloud Applications

Riak greatly simplifies building for enterprise

Big Data and Big Tech is here to stay

Building a *good* Enterprise Architecture requires discipline

If it doesn't scale, it will fail.

Basho Data Platform will do most of the above automatically

Why the Basho Platform?

Simplifies Big Data, IoT and Hybrid Cloud Applications

Simplify Complexity

High Availability and Fault Tolerance

Integrated Real-time Analytics with Apache Spark

Faster Application Performance with Integrated Redis

Enriched Search with Solr



Questions?

jyaqub@basho.com

