MongoDB Introduction and Red Hat Integration Points

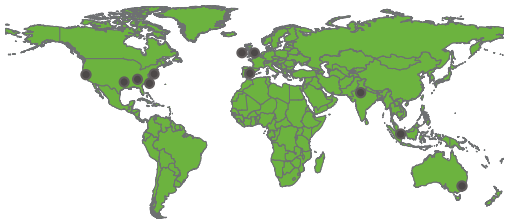
Chad Tindel
Solution Architect

mongoDB

MongoDB Overview

350+ employees

1,000+ customers



13 offices around the world

NEA · SEQUOIA CAPITAL · intel Capital · union square ventures · T.RowePrice · Fidelity · FLYBRIDGE CAPITAL PARTNERS · salesforce · redhat · ALTIMETER · IQT

Over **$231 million** in funding

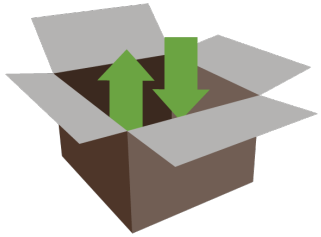
mongoDB

MongoDB

The leading NoSQL database

General Purpose

Document Database



Open-Source

mongoDB

MongoDB Vision

To provide the best database for how we build and run apps today

Build

- New and complex data
- Flexible
- New languages
- Faster development

Run

- Big Data scalability
- Real-time
- Commodity hardware
- Cloud

mongoDB

MongoDB is so easy….

MongoDB is so easy….

Even a baby can use it!

mongoDB

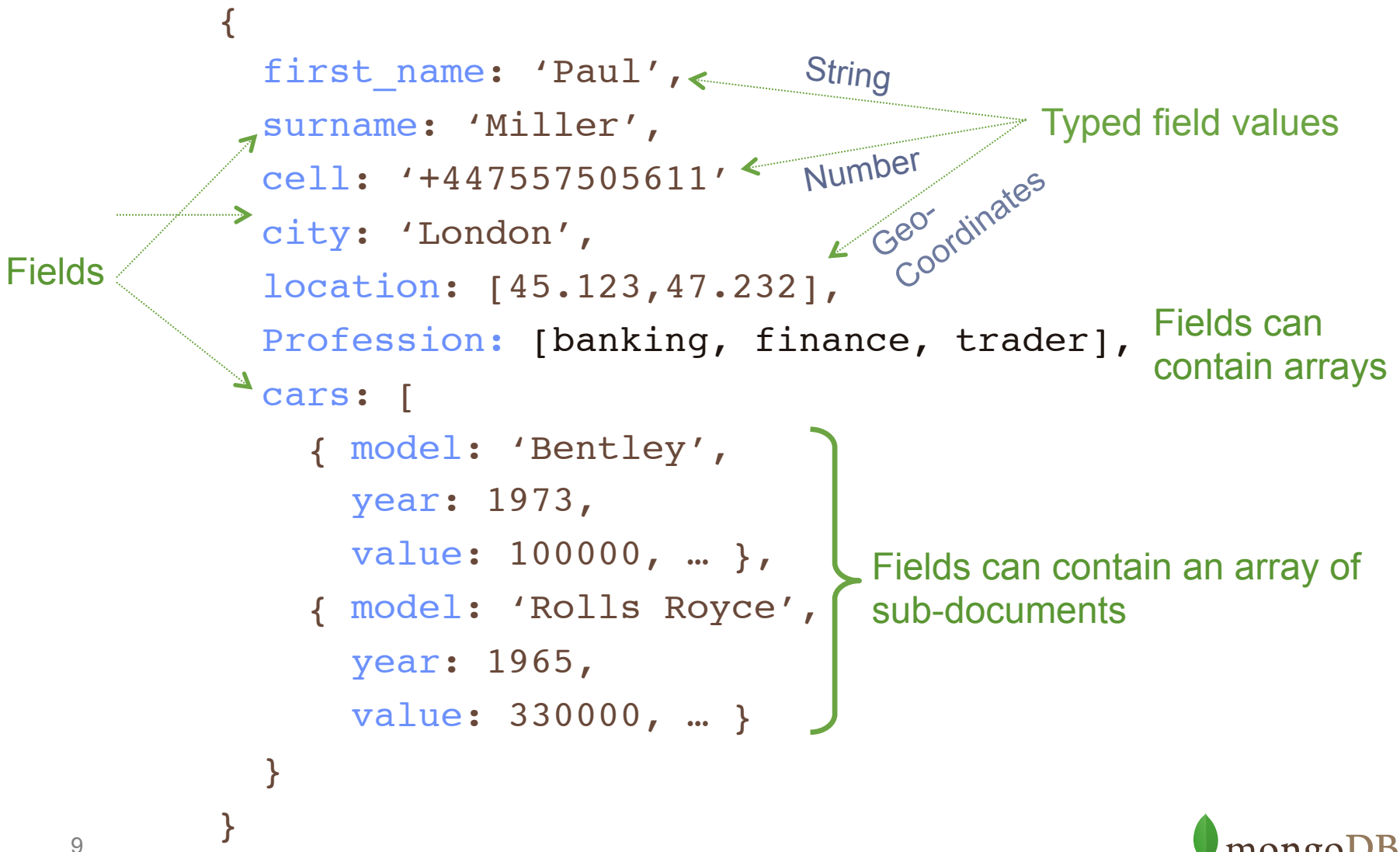
Top 10 Apps in All Industries for MongoDB

1. Customer Data Mgt.
2. Product and Asset Catalogs
3. Social and Collaboration Apps
4. Mobile Apps
5. M2M / Internet of Things
6. Security and Fraud Apps
7. PaaS/DBaaS
8. Data Hub
9. Analytics

Most benefit from using MongoDB

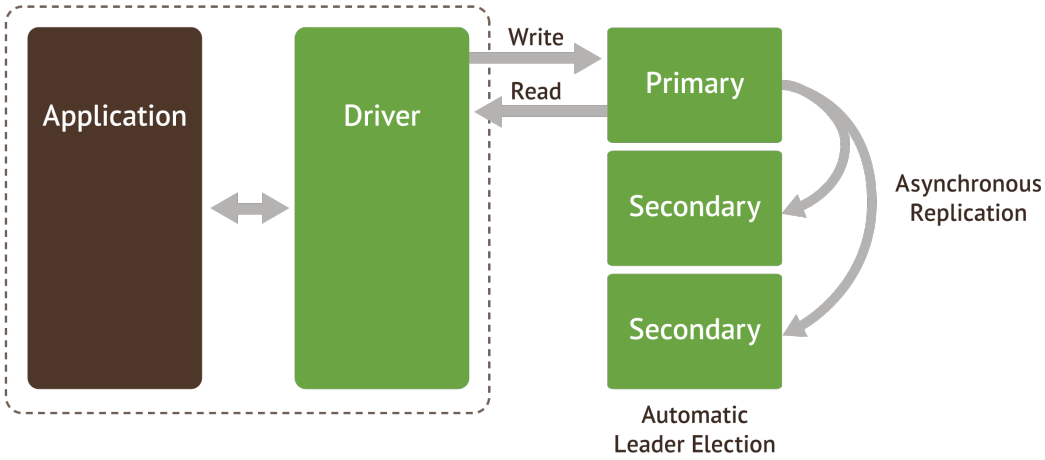
- ❑ You want to easily aggregate data from multiple sources
- ❑ You want agile development and/or fastest time-to-market
- ❑ You want to offer location-based services (latitude/longitude)
- ❑ You expect the schema to change often
- ❑ You have variably or un-structured data (records might have different fields)
- ❑ Your data is hierarchical (i.e. hard to model in RDBMS), e.g. JSON
- ❑ You expect the data to grow quickly and want ease of scaling out
- ❑ You want the best performance possible for real-time read/write
- ❑ You want the lowest TCO and resources including with replication and caching
- ❑ Performance of database directly impacts user experience
- ❑ You want real-time analytics and aggregations
- ❑ You have challenges today with building canonical models, scale, TCO, or agility

Documents are Rich Data Structures



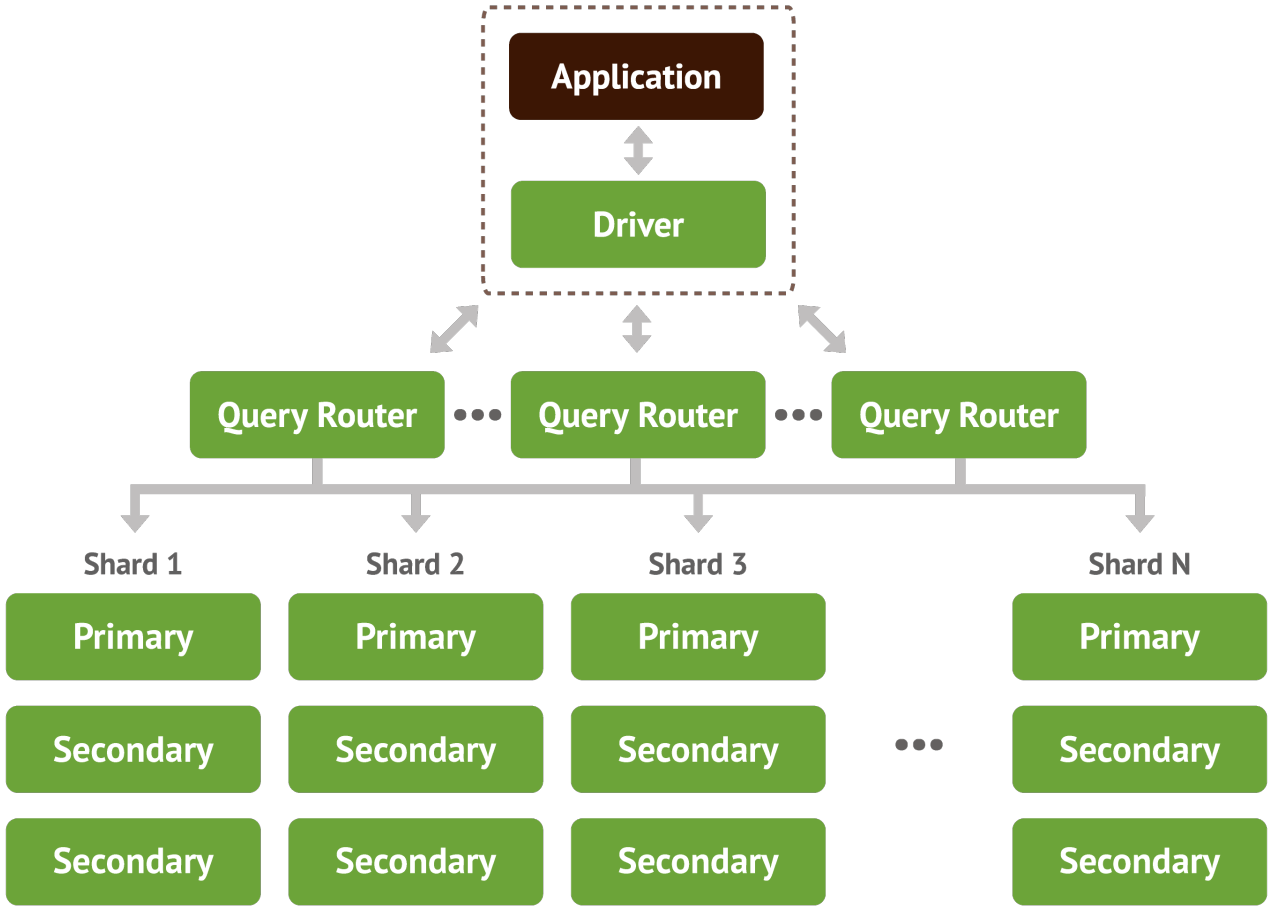
mongoDB

High Availability



- Automated replication and failover
- Multi-data center support
- Improved operational simplicity (e.g., HW swaps)
- Data durability and consistency

MongoDB Architecture



mongoDB

Shell and Drivers

Drivers

Drivers for most popular programming languages and frameworks


Java


Ruby

JavaScript


Perl


Python


Haskell

Shell

Command-line shell for interacting directly with database

```
> db.collection.insert({product:"MongoDB",
type:"Document Database"})
>
> db.collection.findOne()
{
        "_id"         : ObjectId("5106c1c2fc629bfe52792e86"),
        "product"     : "MongoDB"
        "type"        : "Document Database"
}
```


mongoDB

Red Hat Software Collections 1.1 beta now available – adds Apache, MongoDB, more

Posted on March 20, 2014 by Mike Guerette and Langdon White

4 Votes

BRIDGING DEVELOPER AGILITY WITH PRODUCTION STABILITY.

This is what we do.

Today, we are pleased to announce the beta availability of Red Hat Software Collections 1.1, the second installment of Red Hat Software Collections which was launched in October 2013. Red Hat Software Collections delivers a comprehensive suite of runtime languages, open source databases, and related tools helping developers and systems administrators accelerate the creation of stable, modern web applications.

Based on your wish list voting, Red Hat Software Collections 1.1 Beta expands the offering with several new options, including:

- Two new open source HTTP server options in the form of **Apache httpd server 2.4.6** server and **nginx 1.4.4** (The latter is Technology Preview only)
- **Ruby 2.0** and **Rails 4.0.2**, which for the first time will be broken out into their own collections, providing developers access to an updated version of Ruby without requiring the installation of Rails.
- **PHP 5.5** – the latest stable PHP version.
- **MongoDB 2.4.9**, a high-performance document database that provides high availability and easy

MongoDB and Docker

Why Docker?

- Containers provide advantages of virtual machines with less overhead
- Lightweight approach to managing different classes of infrastructure
- Simple deployment and management model
- Containers can be packaged and shared easily

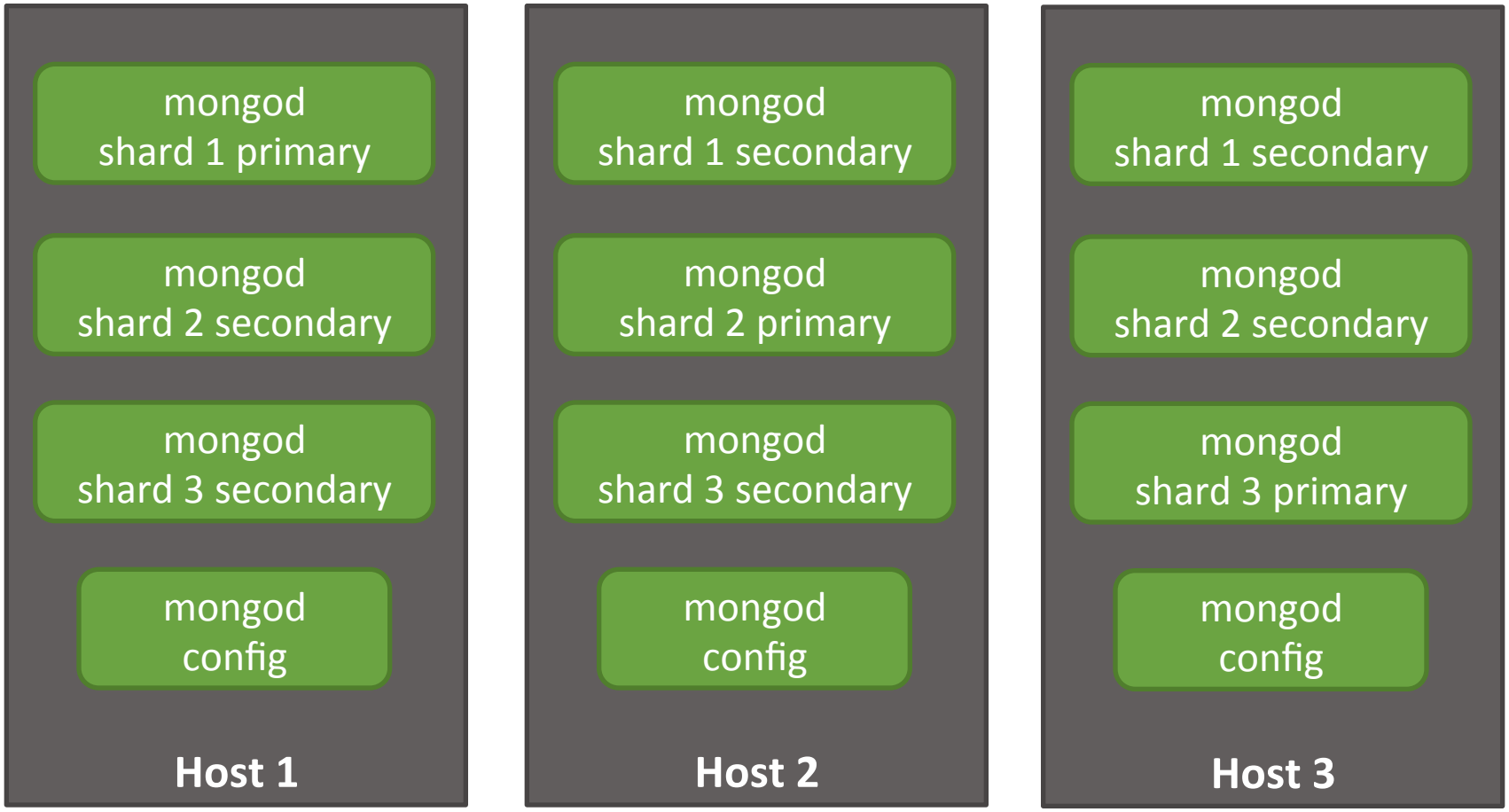
Deployment Best Practices

- Use for “microsharding”
- Avoid storage bottlenecks, use dedicated volumes
- Put base settings in mongod.conf (e.g. dbpath)
- Set specific container params at runtime (e.g. port)

```
$ docker run mongodb --port 5001
```

- Add iptables rules to map exposed ports

Example Sharded Deployment



Dockerfile

```
FROM centos:latest

ADD mongodb.repo /etc/yum.repos.d/mongodb.repo

RUN yum –q –y update

RUN yum –q –y mongo-10gen-server

ADD mongod.conf /etc/mongod.conf

VOLUME [“/data”]

ENTRYPOINT [“/usr/bin/mongod”, “--config”, “/etc/mongod.conf”]
```

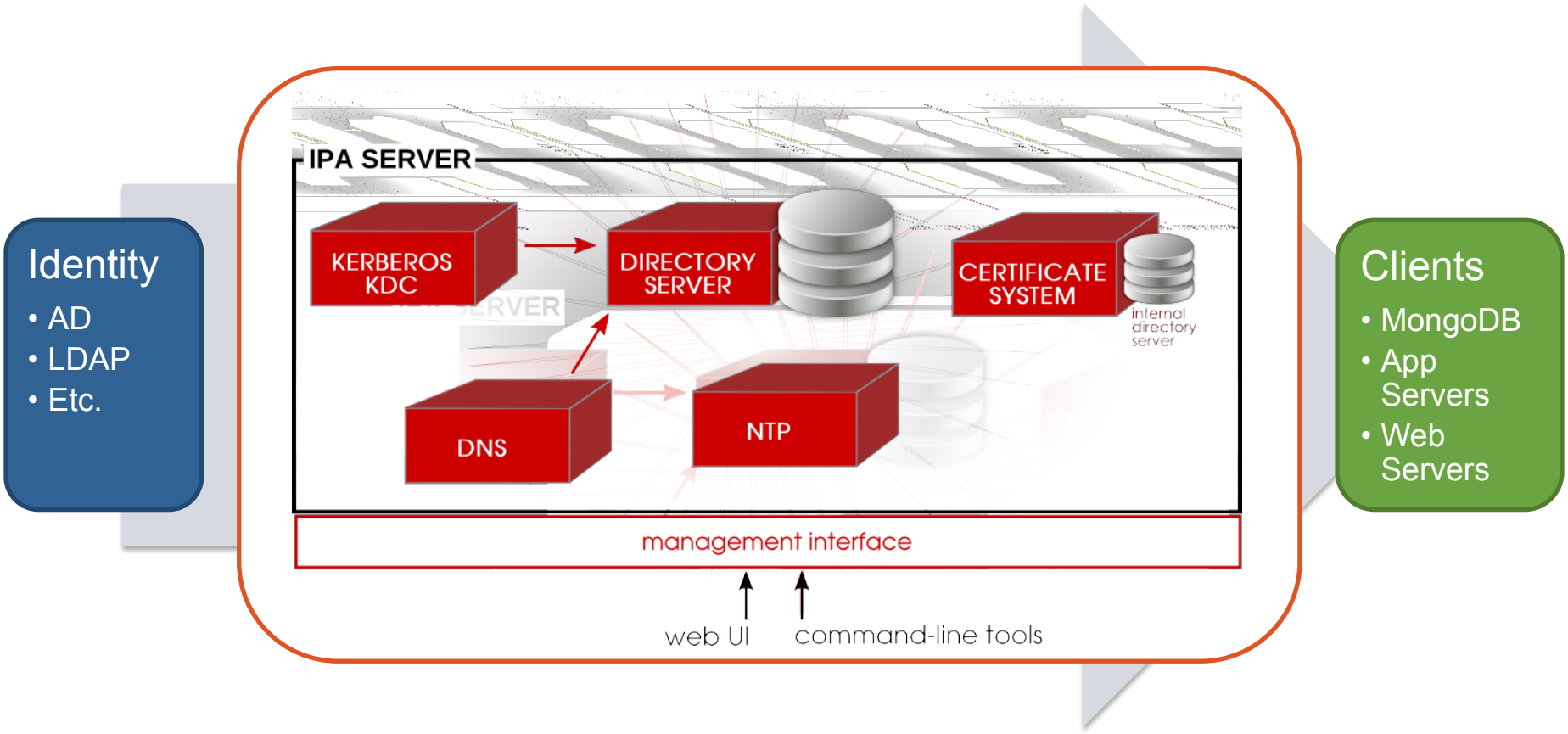
Resources

- MongoDB on Docker
  http://github.com/crcsmnky/mongodb-docker
- Coming Soon
  - Reference architecture
  - Complete documentation
  - Setup and walkthrough for a sharded cluster

MongoDB and RHEL IdM

mongoDB

Security Architecture



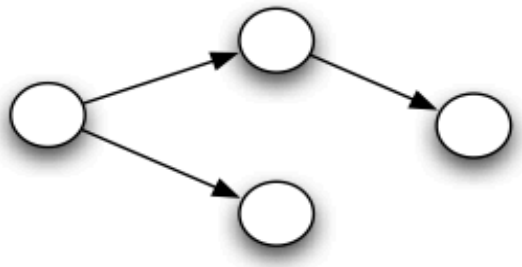
mongoDB

Followup

- RHEL IdM Docs:
  https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html-single/Identity_Management_Guide/index.html
- Setup RHEL IdM for MongoDB Enterprise:
  http://docs.mongodb.org/ecosystem/tutorial/configure-red-hat-enterprise-linux-identity-management/
- Operational RHEL IdM Procedures:
  http://docs.mongodb.org/ecosystem/tutorial/manage-red-hat-enterprise-linux-identity-management/
- Webinar Playback:
  http://www.mongodb.com/presentations/partner-webinar-securing-your-deployment-mongodb-and-red-hats-identity-management-red

MongoDB and Middleware

Hibernate Object/Grid Mapper (OGM)



```
{ "user" : {
  "id": "124",
  "name": "Emmanuel",
  "addresses" : [
    { "city": "Paris", "country": "France" },
    { "city": "Atlanta", "country": "USA" }
  ]
}
```

key	value
123	Address@23
126	“Booya”

1	Things: A foo, B bar, C baz
2	Things: C bam, E coh; People: A Emmanuel
3	Languages: A C, B Java, C Ceylon

Hibernate Object/Grid Mapper (OGM)

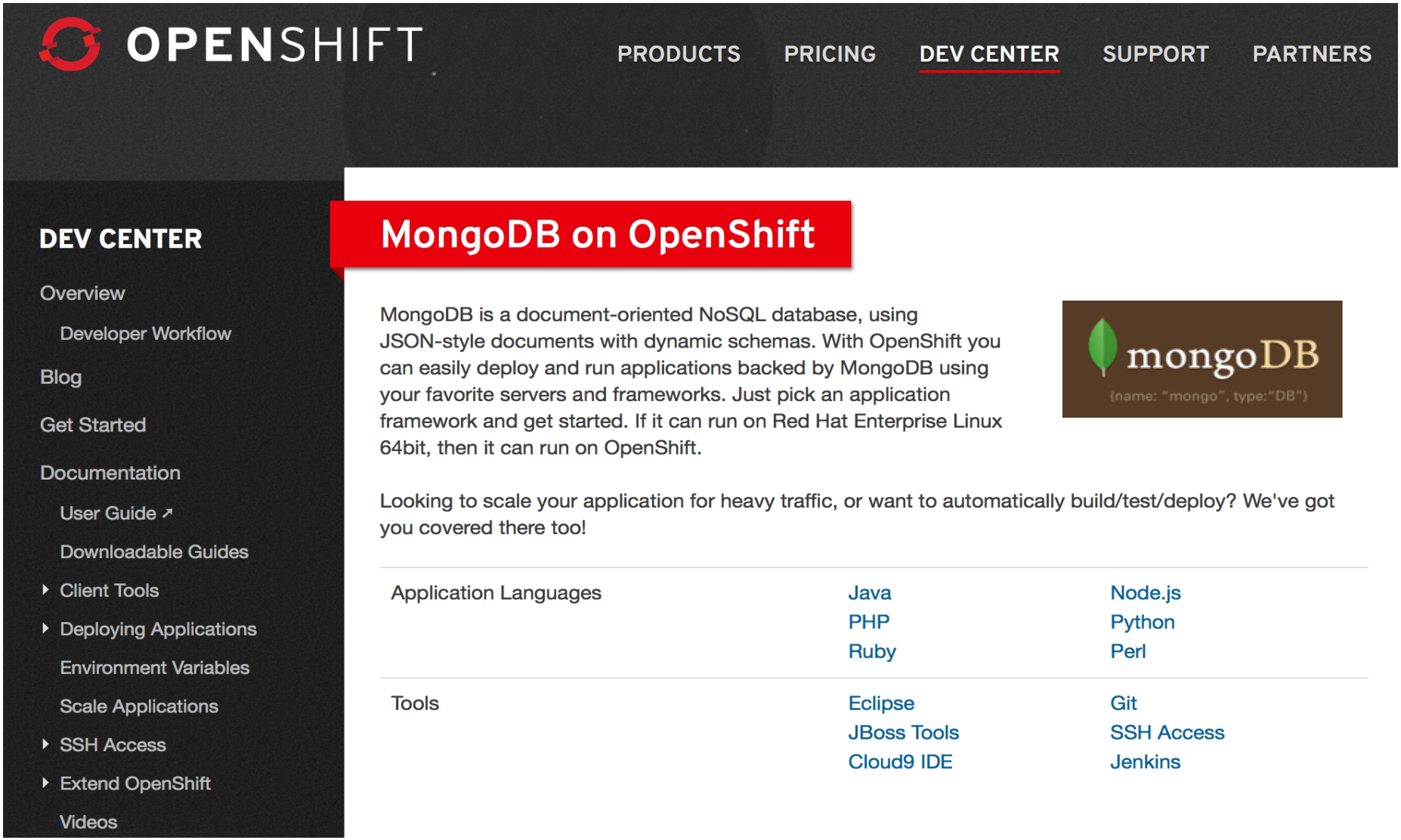
Example 5.8. Java entity

```
@Entity
public class AccountOwner {
@Id
private String id;

@ManyToMany
public Set<BankAccount> bankAccounts;

//getters, setters, ...
```

Example 5.9. JSON representation

```
{
    "_id" : "owner0001",
    "bankAccounts" : [
        { "bankAccounts_id" : "accountXYZ" }
    ]
}
```

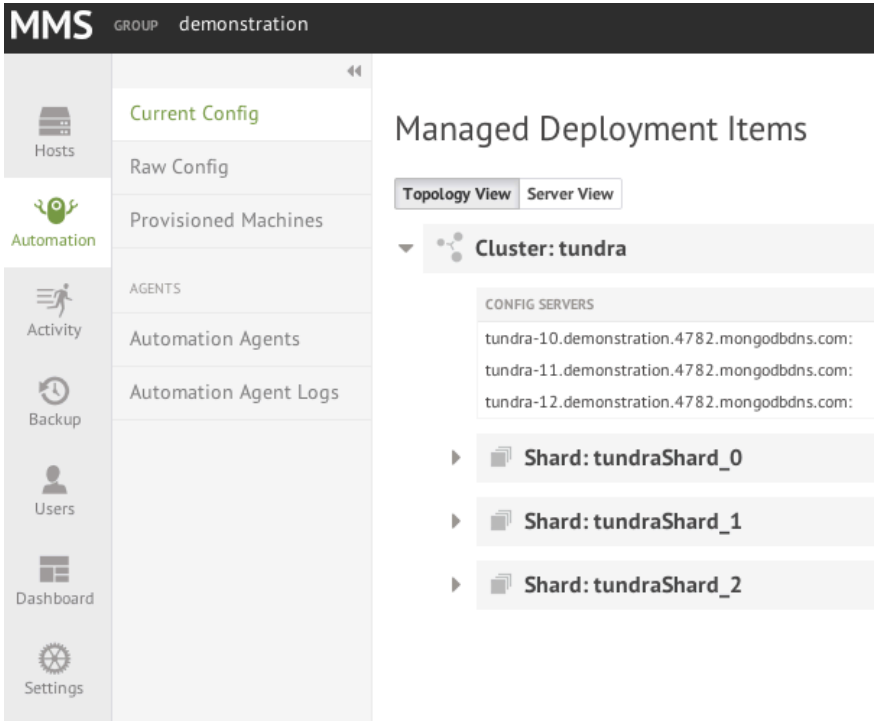
MongoDB OpenShift Cartridge



MongoDB MMS / OpenStack

Automation

- Sophisticated and Simple
  - Beautiful
  - Easy to use
- Create and manage
  - Replica sets
  - Sharded systems
  - Hot upgrades
- Cloud-ready
  - Public: AWS, Rackspace
  - Private: VMWare, OpenStack



mongoDB

Questions?

Chad Tindel
chad.tindel@mongodb.com

mongoDB