powderkeg

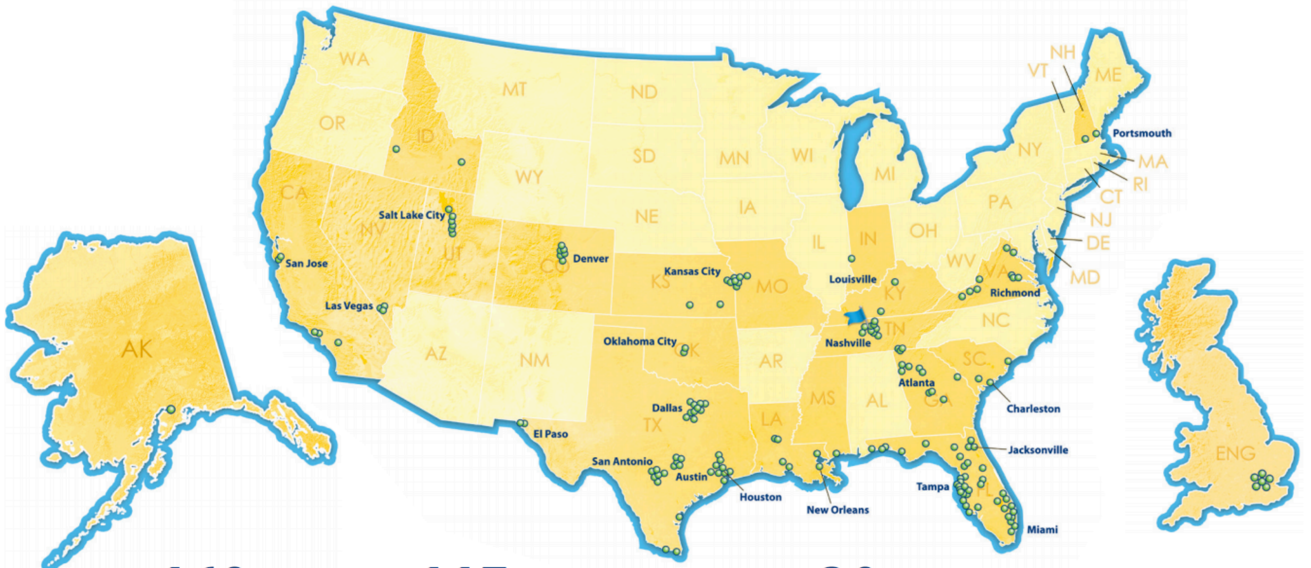
teaching Clojure to Spark

HCA

Hospital Corporation of America℠

“Above all else, we are committed to the care and improvement of human life. In recognition of this commitment, we strive to deliver high-quality, cost-effective healthcare in the communities we serve.”

– HCA Mission Statement



169 hospitals and **117** surgery centers located in **20** states and the United Kingdom.

Size

233,000 employees

79,000 nurses

37,000 active physicians

RANKED 63RD in Fortune 100

Patient Care

Approximately **5%** of all U.S. hospital services happen at an HCA facility, including:

26.4 MILLION patient encounters

8.1 MILLION emergency room visits

210,000 babies delivered

HCA Datalab

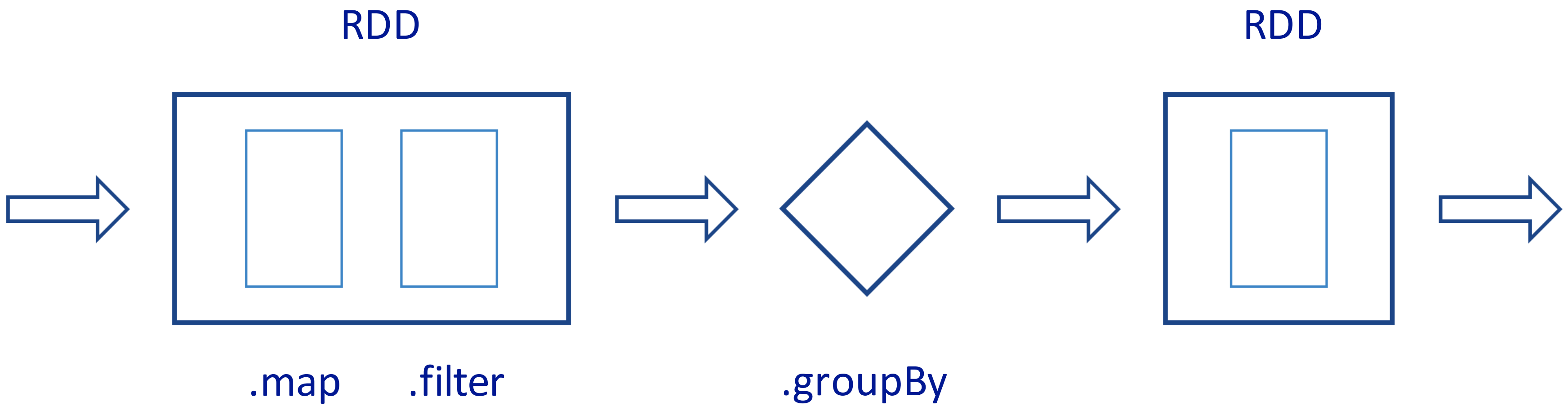
”Our core mission is create clinical, operational and financial value from several data sources including our large clinical data warehouse and big data platform. ”

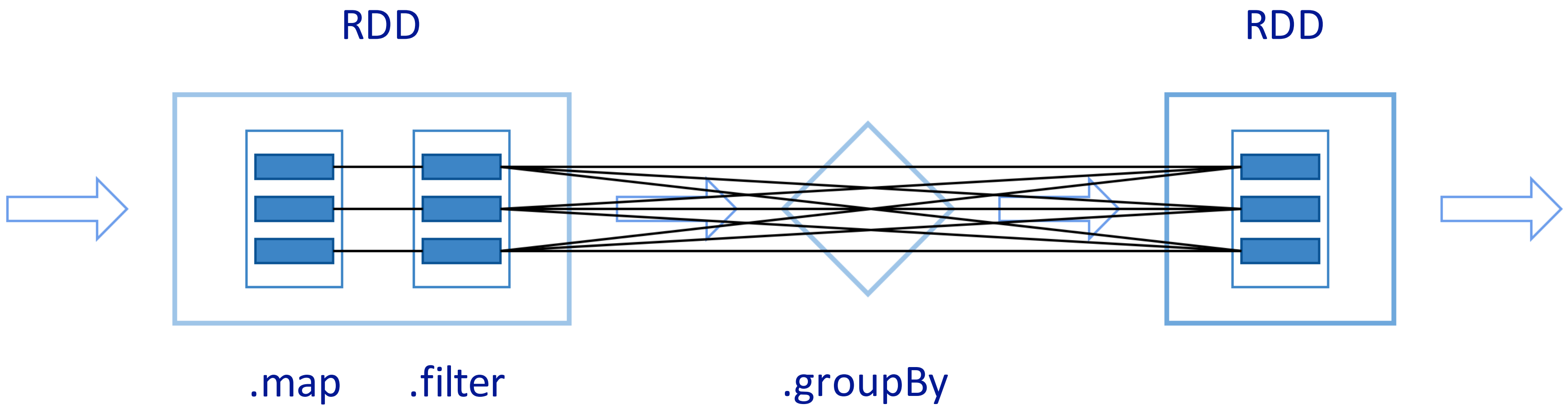
– Edmund Jackson VP & Chief Data Scientist

Spark

RDD

Resilient **D**istributed **D**ataset are primary abstraction in Spark – a fault-tolerant collection of elements that can be operated on in parallel.


RDD
RDD
.map
.filter
.groupBy


RDD
RDD
.map
.filter
.groupBy

yieldbot / **flambo**

gorillalabs / **sparkling**

```clojure
; Clojure:
(into [] (map #(* % %)) [1 2 3 4 5])
```

```
; Clojure:
(into [] (map #(* % %)) [1 2 3 4 5])


; Flambo:
(-> (f/parallelize sc [1 2 3 4 5])
    (f/map (f/fn [x] (* x x)))
    f/collect)
```

could it be ~~simpler~~ ~~easier~~ clojurer?

- avoid duplicating clojure.core functionality
- no AOT compile-restart cycles
- live coding the cluster

```clojure
; Clojure:
(into [] (map #(* % %)) [1 2 3 4 5])


; Flambo:
(-> (f/parallelize sc [1 2 3 4 5])
    (f/map (f/fn [x] (* x x)))
    f/collect)


; Powderkeg PoC:
(into [] (keg/rdd [1 2 3 4 5] (map #(* % %))))
```

avoiding AOT

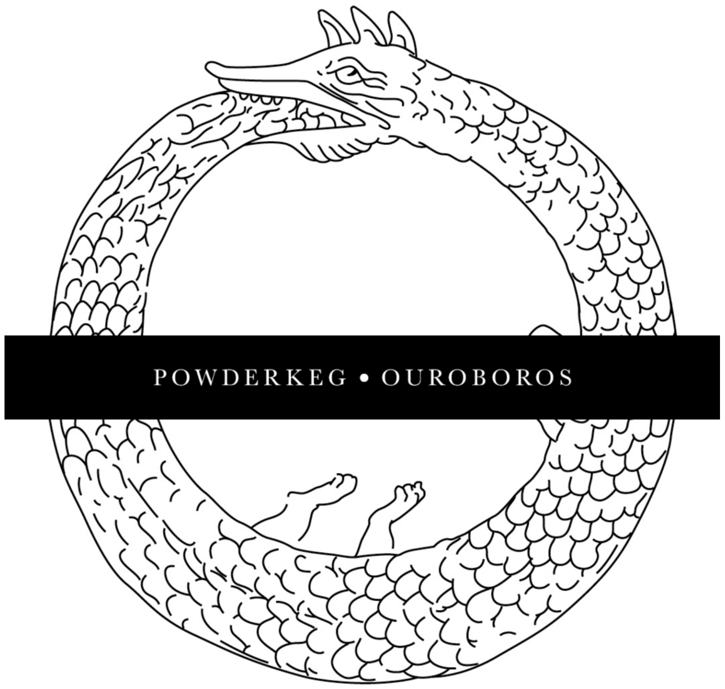
spark needs to distribute the classes and state of the application (REPL) to the worker nodes

three approaches:

- reuse Scala Spark REPL
- fork Clojure dynamic class loader
- use Java Agent

ouroboros: the agent instrumenting itself

- connects JVM to itself
- makes `Instrumentation` object available to REPL
- read and modify any class



✓
```
(into [] (keg/rdd [1 2 3 4 5]
                  (map #(* % %))))
```

❌

```
(def mul *)
(into [] (keg/rdd [1 2 3 4 5]
                  (map #(mul % %))))
```

transferring namespaces

- traverse namespaces
- build an immutable representation
- send it to workers
- workers recreate namespaces and vars

✓

```
(def mul *)
(into [] (keg/rdd [1 2 3 4 5]
                  (map #(mul % %))))
```

updating namespaces

- traverse again and diff
- wait we have ouroboros
- patch `clojure.lang.Var` bytecode to get notified

demo

the basics

key-value RDD

RDD of `scala.Tuple2` instances (PairRDD in Java)

```
(keg/rdd {:a 1 :b 2 :c 3})

(keg/rdd (range 100)
         (map (fn [n] [(mod 7 n) n])))
```

xforms: extra transducers

https://github.com/cgrand/xforms

group-by + reduce = reduce-by

- `xforms/for`
- `xforms/by-key`

xforms: for

```
(into []
  (x/for [i %, j (range i)] [i j])
  (range 10))
```

xforms: by-key

```
(defn map-vals [m f]
  (into {} (x/by-key (map f)) m))

(defn my-group-by [kfn coll]
  (into {} (x/by-key kfn (x/into [])) coll))

(defn my-frequencies [coll]
  (into {} (x/by-key identity x/count) coll))

(defn sum-by [kfn coll]
  (into {} (x/by-key kfn (x/reduce +)) coll))
```

xforms: KvRfable

- `reduce-kv` meets transducers
- No pairs allocations

```
(defprotocol KvRfable
  "Protocol for reducing fns that accept key
   and val as separate arguments."
  (some-kvrf [f] "Returns a kvrf or nil"))
```

powderkeg/by-key

- partitions and shuffles RDDs

```
(keg/by-key (keg/rdd {:a 1 :b 2 :c 3}))
```

- creates partitioned RDDs from collection given a function to create keys

```
(keg/by-key (range 100) :key #(mod % 7))
```

powderkeg/by-key

- perform transformations on the workers before and after after shuffling

```
(keg/by-key (range 100)
            :key odd?
            (x/reduce +))


(keg/by-key (keg/rdd {:a 1 :b 2})
            :pre (map inc)
            :post (map str))
```

demo

nuance of by-key

```scala
val textFile =
  sc.textFile("hdfs://...")
val counts =
  textFile.flatMap(line => line.split(" "))
          .map(word => (word, 1))
          .reduceByKey(_ + _)
```

```
val textFile =
  sc.textFile("hdfs://...")
val counts =
  textFile.flatMap(line => line.split(" "))
          .map(word => (word, 1))
          .reduceByKey(_ + _)
```

```
(-> (.textFile keg/*sc* "file:…")

    (keg/rdd (mapcat #(str/split % #" "))
             (map (fn [x] [x 1])))
    (keg/by-key (x/reduce +)))
```

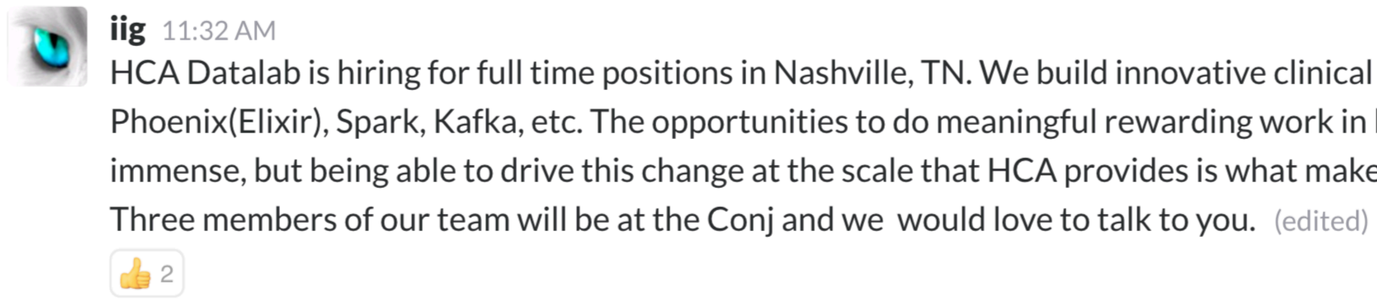
```
(-> (.textFile keg/*sc* "file:…")
    (keg/rdd (mapcat #(str/split % #" ")))
    (keg/by-key :key  identity
                :pre  x/count
                :post (x/reduce +)))
```

```clojure
;; inner=>
(into {} (keg/join (keg/by-key {:a 1 :b 2 :c 10})
                   (keg/by-key {:a 3 :b 4 :d 20})
                   (x/for [[l r] %] (+ l r))))
{:b 6, :a 4}

;; right outer=>
(into {} (keg/join (keg/by-key {:a 1 :b 2 :c 10}) :or 0
                   (keg/by-key {:a 3 :b 4 :d 20})
                   (x/for [[l r] %] (+ l r))))
{:b 6, :a 4, :d 20}

;; left outer=>
(into {} (keg/join (keg/by-key {:a 1 :b 2 :c 10})
                   (keg/by-key {:a 3 :b 4 :d 20}) :or 0
                   (x/for [[l r] %] (+ l r))))
{:b 6, :c 10, :a 4}

;; full outer=>
(into {} (keg/join (keg/by-key {:a 1 :b 2 :c 10}) :or 0
                   (keg/by-key {:a 3 :b 4 :d 20}) :or 0
                   (x/for [[l r] %] (+ l r))))
{:b 6, :c 10, :a 4, :d 20}
```

what’s next

- streams (in progress)
- dataframes
- migrate to Spark 2.0
- live coding the cloud

https://github.com/HCADatalab/powderkeg



iig 11:32 AM

HCA Datalab is hiring for full time positions in Nashville, TN. We build innovative clinical products using Clojure, Phoenix(Elixir), Spark, Kafka, etc. The opportunities to do meaningful rewarding work in healthcare field are immense, but being able to drive this change at the scale that HCA provides is what makes this a very special gig. Three members of our team will be at the Conj and we would love to talk to you. (edited)

👍 2