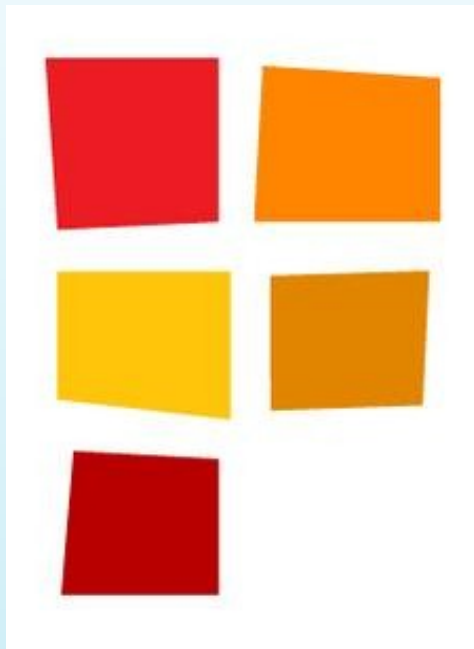


factual™

Clojure on Mongo: Fun and Easy with CongoMongo

MongoDB Los Angeles
Aaron Crow, Software Engineer
aaron@factual.com
[@factual_aaron](https://twitter.com/factual_aaron)





**I wanted high developer
productivity while writing a task
queueing system that uses
MongoDB**



- ▶ 55+ million place entities
- ▶ 45+ countries
- ▶ 17+ million sources
- ▶ 2.3 billion inputs

...all day long...

AGGREGATING

NORMALIZING

MERGING

PURGING

DE-DUPING

MAPPING

KEEPING EVERYTHING UP TO DATE

Factual Public API Call:

```
api.v3.factual.com /t/places?filters=
  {"locality": "los angeles"}&q=sushi
```

Returns JSON:

```
factual_id: "0015f529-1a5d-4966-9cc9-ce59423f09af", name: "Chopstick Sushi and Roll",
factual_id: "053514fc-65f2-42a7-948d-787f9e050bf9", name: "Sushi Ko", ...
factual_id: "05f2847a-acbf-4dcd-ac16-5d828d40721d", name: "Sushi Moon", ...
...
```

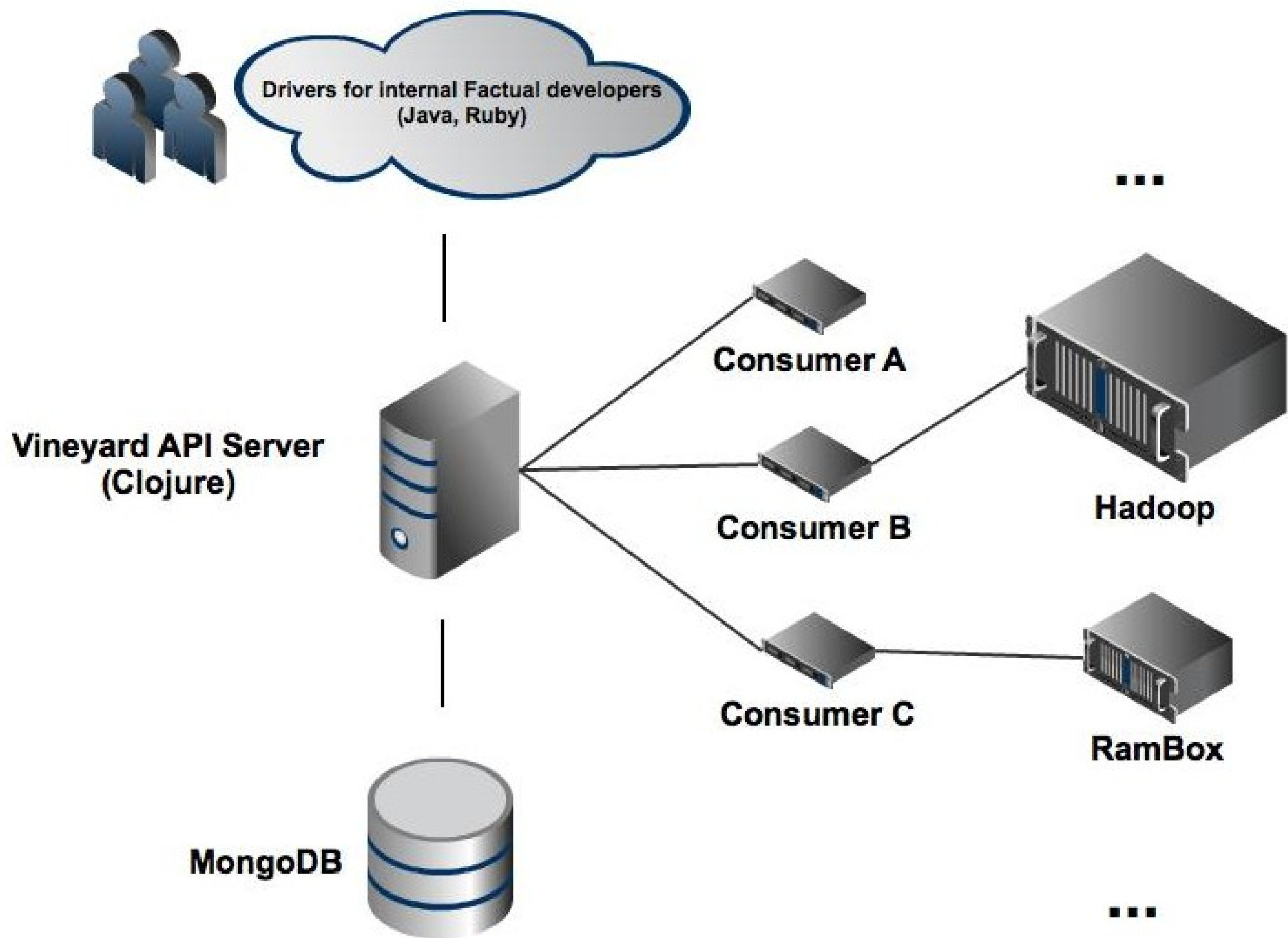
Otherwise, badness:

- ▶ Too many humans need to be involved
- ▶ Too much manual labor
- ▶ Confusion reigns, lack of coordination
- ▶ Tasks “die on the vine”

Simple Requirements

- ▶ Push tasks on to resource queues
- ▶ Associate tasks with each other
- ▶ Pop tasks off of resource queues
- ▶ Track and admin tasks
- ▶ Support Factual specific requirements

Vineyard Architecture



?

Why MongoDB?

MongoDB is Web Scale

elnurabd



Subscribe

3 videos ▾



 xtra
normal



5:16 / 5:36



360p



"It is better to have 100 functions operate on one data structure than 10 functions on 10 data structures."

-- Alan Jay Perlis (1922 – 1990)

First recipient of Turing Award

```
{:status :RUNNING  
 :submitter_hostname "Aaron-Crows-MacBook-Air.local"  
 :heartbeat {:$lt 1326848557622}  
 :attempts {:$lt 3}}
```

```
( { :_id "de2e139b-cd9c-4491-a1a1-70351147db76" ,  
  :status "PENDING" ,  
  :attempts 0 ,  
  :init_data { :key1 "value1" , :key2 "value2" } }  
 { :_id "06d11a9d-8740-40f6-991c-176f35670320" ,  
  :attempts 0 ,  
  :status "PENDING" ,  
  :init_data { :my_key "1" , :another_key "2" } ,  
 { :_id "6b8ae690-f4e8-4a0d-a084-6b8feb5aa9ed" ,  
  :attempts 0 ,  
  :heartbeat 1326945926732 ,  
  :status "DONE" ,  
  :next_tasks  
    [ "27926cb3-2fd0-49f6-817b-acecef6ab673"  
      "20dfec67-457e-4a82-8c7d-e78b0b3dc9b5"  
      "bec1caea-e8ef-4409-8073-ef7131425359" ] } } )
```

```
(first (get-in t [:init_data :key2]))
```

```
(assoc-in t [:init_data :key2] :new_val)
```

```
(map :submitter_hostname tasks)
```

```
;; etc. etc. etc.
```

Why CongoMongo?

- ▶ Because it was there!

But also...

- ▶ Well designed
- ▶ Well maintained
- ▶ Well documented

IMPORT

```
(ns my-mongo-app  
  (:use somnium.congomongo))
```

MAKE A CONNECTION

```
(def conn  
  (make-connection "mydb"  
                   :host "127.0.0.1"  
                   :port 27017))  
  
=> #'user/conn  
  
conn => {:mongo #<Mongo Mongo: 127.0.0.1:20717>, :db #<DBApiLayer mydb>}
```

SET THE CONNECTION GLOBALLY

```
(set-connection! conn)
```


AD-HOC QUERIES

```
( fetch-one
  :points
  :where { :x { :$gt 10
                :$lt 20 }
           :y 42
           :z { :$gt 500 } })
```

```
=> { :x 12, :y 42, :z 504, :_ns "points", :_id ... }
```

```
DBCollection coll = db.getCollection("tasks");  
BasicDBObject query = new BasicDBObject();  
query.put("attempts", new BasicDBObject(  
    "$gt", 3));  
  
return coll.find(query).toArray();
```

```
(fetch :tasks :where {:attempts {:> 3}})
```

```
(get-tasks (where (attempts< 3)))
```

JUL 17TH, 2011 | [0 COMMENTS AND 4 REACTIONS](#)

Creating a Query DSL Using Clojure and MongoDB

One of the nice things about [MongoDB](#) (particularly when using it in Clojure via the [Congomongo](#) library) is that its map-based query language is so amenable to the creation of a domain-specific language, or [DSL](#). Creating and manipulating maps is like breathing in Clojure, so it is trivial to decompose the different query requirements of your application into a small collection of simple functions that can be used to create a rather fluent domain-specific language. The data-structure-based query language of MongoDB makes this possible (or at least easier; it would be much more difficult to do in a string-based language like SQL).

Insulate from changes

```
; "That's almost exactly what the  
; equivalent request would be in  
; plain English. You don't get much  
; simpler."
```

```
(find-psms  
  (matching-peptide "GLYQRPHDSTRFK")  
  (with-e-value-cutoff 0.001)  
  (in-region { :chromosome "X"  
                :strand "+"  
                :start 12345  
                :stop 34567 })))
```

Working with maps in
Clojure is like breathing

Small simple DSL fns
that you can compose

?

“Trellis”: Watch over tasks

```
( fetch
  :tasks
  :where { :status :RUNNING
           :heartbeat { :$lt (minutes-ago 10) }
           :attempts { :$lt 3 } } )
```



```
(get-tasks  
  (where  
    (status= :RUNNING)  
    (stale-heartbeat)  
    (attempts< 3)))
```



`(repush-all! (died-on-vine))`



Some DSL implementation

```
(defn where [& criteria]
  (apply merge criteria))

(defn get-tasks [filters]
  (fetch :tasks :where filters))

(defn heartbeat< [time]
  {:heartbeat {:$lt time}})

(defn stale-heartbeat []
  (heartbeat< (minutes-ago 10)))
```

?

Clojure DSL on Factual (PrettyQL)

```
(select restaurants-us  
  (around {  
    :lat 34.06021 :lon -118.4183 :miles 3})  
  (where  
    (= :meal_deliver true)  
    (= :meal_dinner true))  
  (order :$distance)  
  (limit 3) )
```



A blaster is f ne... but...



... a lightsaber is a game changer.





Thanks CongoMongo devs!

aboekhoff

purcell

christophermaier

njackson

seancorfield

(and friends!)

Mongo LA Meetup @ Factual!

aaron@factual.com

@factual_aaron

<http://github.com/dirtyvagabond/mongola>