: YAFI

Yet Another Filesytem for Internet

http://chrishardcastle.co.uk/couchdb-intro

2010/10/09
@OSDC.fr

Mail : germain.maurice _A_ **linkfluence.net**

"NoSQL"?

- NoSQL = **N**ot **O**nly **SQL**
  - a complement to SQL databases
- Sometimes SQL is heavy and a bit stiff to use
- NoSQL is schemaless
  - Put what you want, when you want.

Context

Context

- **C**onsistency
  - at any time, all copies of a data are the same over the whole system
- **A**vailability
  - fast response of the system
- **P**artition tolerance
  - the system will not fail until a complete fault of all members of the system

CAP theorem

- Theorem says : " It's impossible to satisfy **C** and **A** and **P** at the same time."

CAP theorem

- Theorem says : " It's impossible to satisfy **C** and **A** and **P** at the same time."
- Riak says : "I pick **A** and **P** then, adjust **C** to your needs at each operation"

What is Riak?

- A distributed document-oriented database
  - key-value store
  - documents are organized into buckets

- Influences
  - Amazon's Dynamo technology
  - CAP theorem

What is Riak?

- No single point of failure
  - each node of the cluster is a master
  - ... no one is a master
- Data replication
  - **N** : number of data copies configured by bucket
  - Quorums
    - **R** : read quorum
    - **W**, **DW** : write quorums
- A cluster easily scalable
  - `node1$ riak start`
  - `node2$ riak start`
  - `node2$ riak-admin join riak@node1`

What is Riak?

- A REST interface allows you to
  - **PUT**, **GET** and **DELETE** documents
  - **POST** Map/Reduce jobs
  - easy setup with HTTP tools (load balancers, proxies...)

- A "Protocol Buffers" interface for better performance
  - protocol communication designed by Google

Store it

```
> curl -X PUT http://host/riak/mybucket/key \
-H "Content-type: image/png" \
--data-binary @./riak-ring.png
```

Store it

```
> curl -X PUT http://host/riak/mybucket/key \
-H "Content-type: image/png" \
--data-binary @./riak-ring.png
```

```
> curl -X PUT http://host/riak/mybucket/key \
-H "Content-type: application/json" \
--data '{"youare" : "at OSDC.fr"}'
```

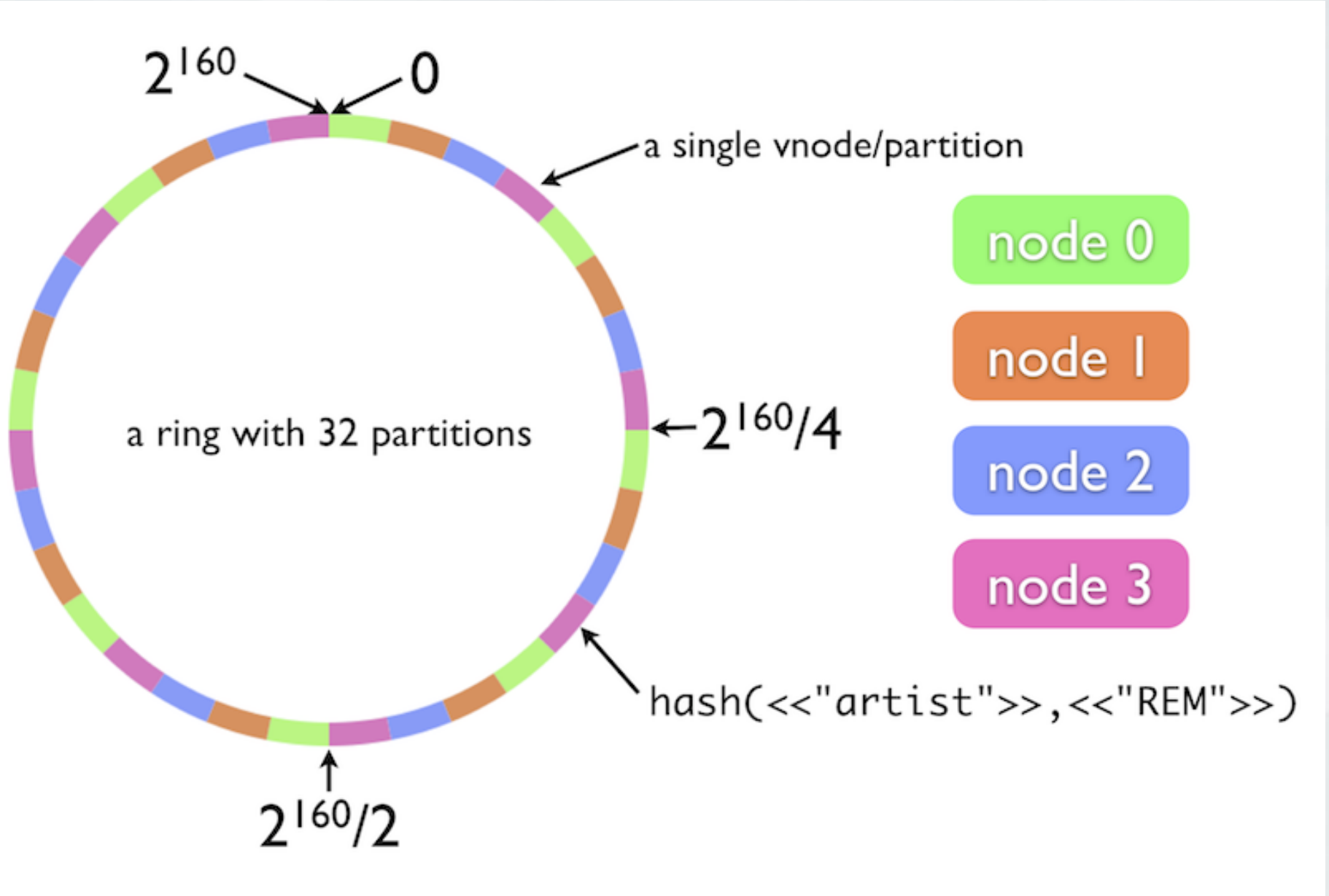
Read it

```
> curl -X GET http://host/riak/mybucket/key
{"youare" : "at OSDC.fr"}
```

Delete it

```
> curl -X DELETE http://host/riak/mybucket/key
```

The Riak ring



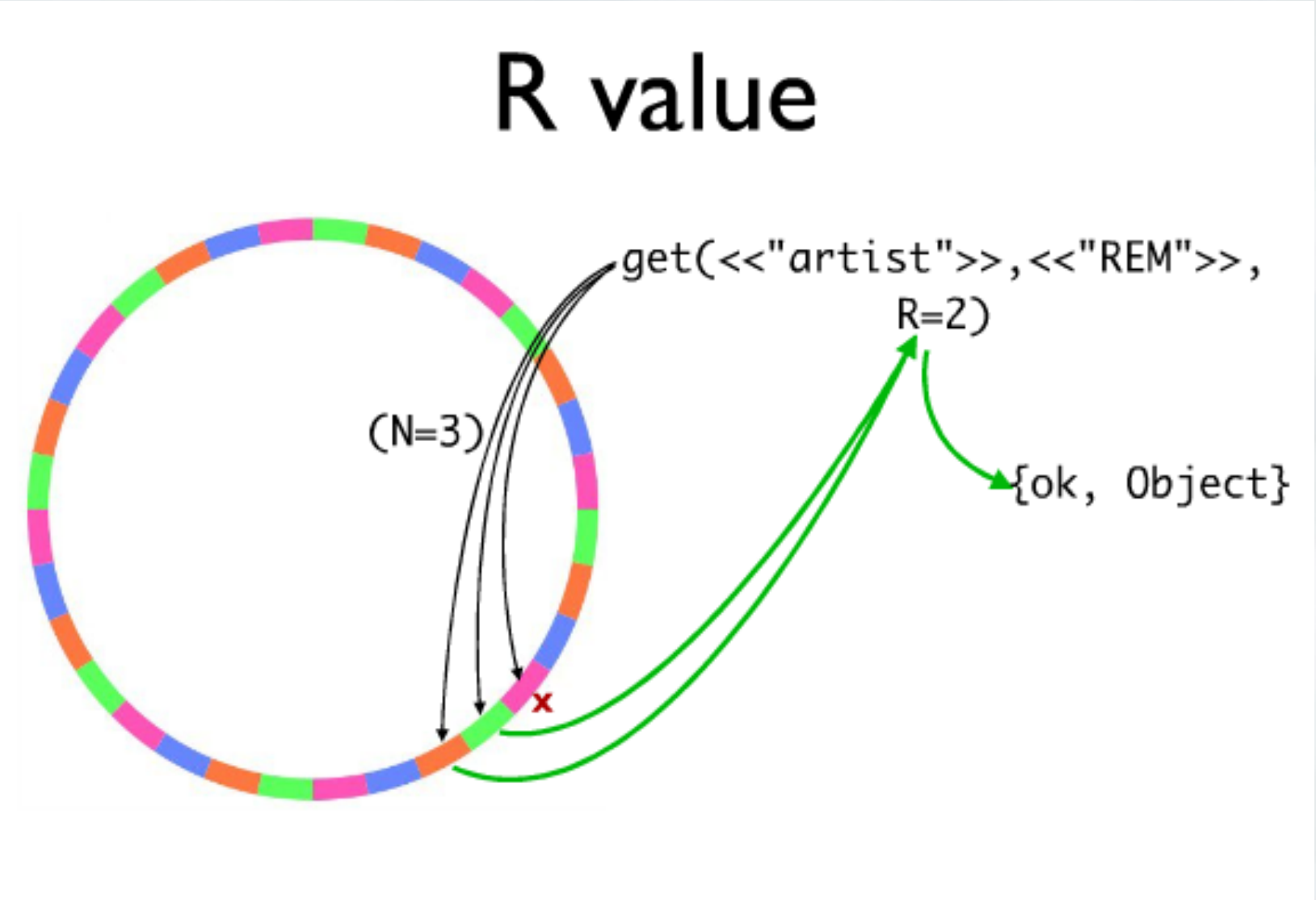
Querying the ring

Read quorums

- **R** as an argument
  - `GET riak/mybucket/key?r=2`
    - 2 replicas of data have to be available to make the request valid

Querying the ring

Read quorums



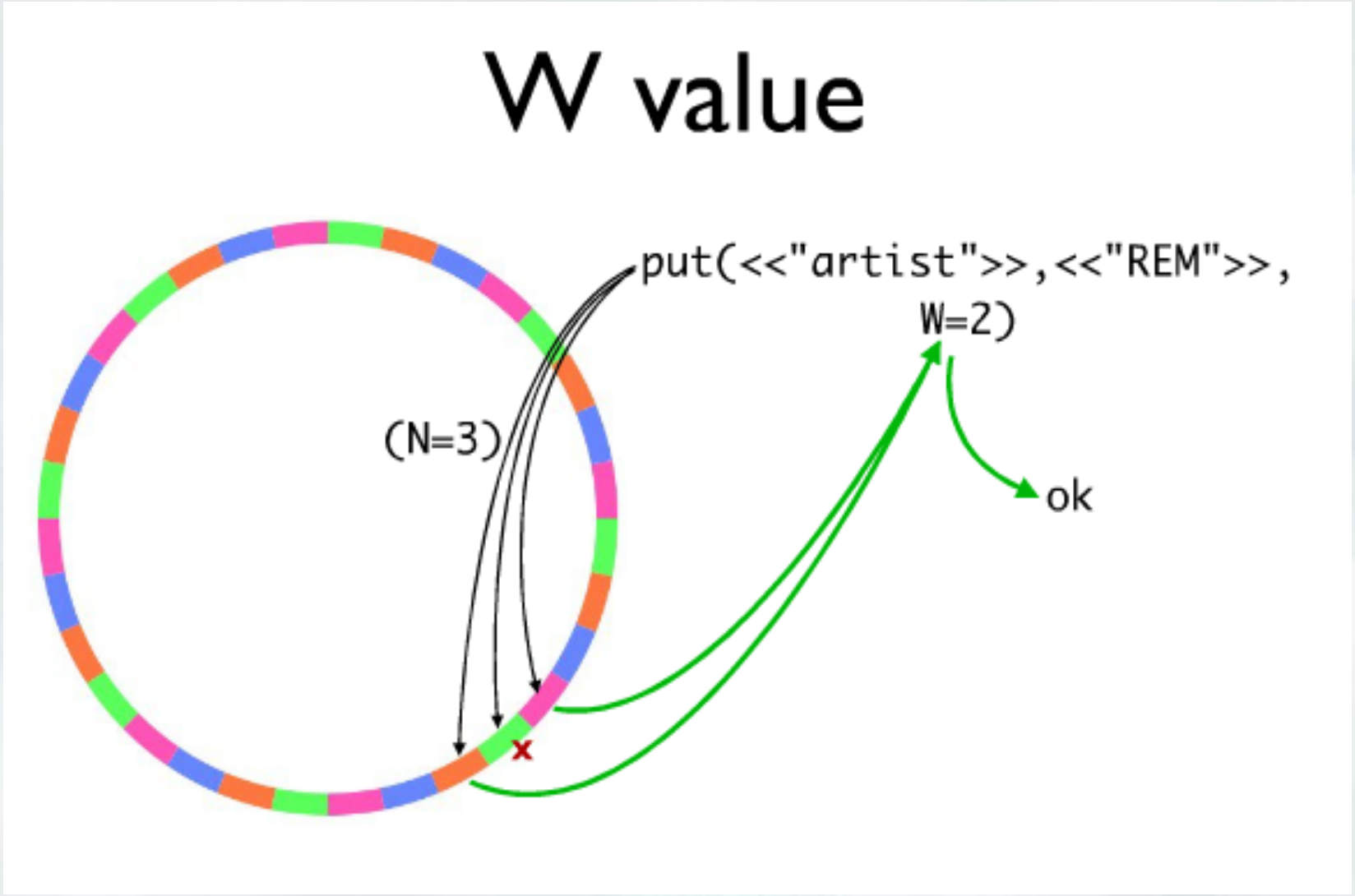
Querying the ring

Write quorums

- **W**, **DW** as arguments
  - `PUT riak/bucket/key?w=2`
    - 2 replicas have to accept the write to make the request valid
  - `PUT riak/bucket/key?dw=2`
    - 2 replicas have to write the data on disk to make the request valid

Querying the ring

Write quorums



Querying the ring

Write quorums

- Mixing **W** and **DW**
  - `PUT riak/bucket/key?dw=1&w=2`
    - 2 replicas have to accept the write request of data
    - **AND**
    - 1 has to write the data on disk

      = the request is valid

Map/Reduce

- Map/Reduce is the way to request your documents
- The request is distributed
- Write your Map/Reduce job in Javascript or Erlang

Example :

- In my fridge i have, 5 carrots, 3 leeks, 3 strawberries, 25 beers, 10 pizzas
- How to count all vegetables i have in my fridge?

How many vegetables in my fridge?

```
> curl -X POST http://host/mapred '{
"inputs":"fridge",
```

How many vegetables in my fridge?

```
> curl -X POST http://host/mapred '{
"inputs":"fridge",
"query":[{
    "map":{"source":"function(food) {
            food = Riak.mapValuesJson(food)[0] ;
            if ( food.type == "vegetable" )
               { return [food.count] }
            return []; }"}
    }

=> return [ 5, 3 ]
```

How many vegetables in my fridge?

```
> curl -X POST http://host/mapred '{
"inputs":"fridge",
"query":[{
    "map":{"source":"function(food) {
            food = Riak.mapValuesJson(food)[0] ;
            if ( food.type == "vegetable" )
               { return [food.count] }
            return []; }"}
    },{
    "reduce":{"source":"function(values, arg) {
        return [ values.reduce(
            function (vegetables, total) {
                return vegetables + total *1;
            }, 0 ) ]; }"}
    }]}'
=> return [ f( 5, f( 3, 0)) ];
```

How many vegetables in my fridge?

```
> curl -X POST http://host/mapred '{
"inputs":"fridge",
"query":[{
    "map":{"source":"function(food) {
            food = Riak.mapValuesJson(food)[0] ;
            if ( food.type == "vegetable" )
               { return [food.count] }
            return []; }"}
    },{
    "reduce":{"source":"function(values, arg) {
        return [ values.reduce(
            function (vegetables, total) {
                return vegetables + total *1;
            }, 0 ) ]; }"}
    }]}'
total : [8]
```

Storage backends

- Riak supports several storage backends
- The main storage backends are :
  - Bitcask, append-only style, with auto-merging
  - Innostore, based on InnoDB, transactional storage
  - Memcached-type memory cache
- Multi-backend option
  - allows you to define one backend by bucket

Behind Riak

- Basho was founded in January 2008
- Eric Brewer, CAP theorem's father, joins board of directors on January 2010
- Basho published Riak under Apache License
  - sources available on Bitbucket and Github
- Basho provides paid support based on the open-source version of Riak
- Reactive open-source support

Supported languages

- Basho is currently developing and supporting drivers for:
  - Erlang
  - Javascript
  - Java
  - PHP
  - Python
  - Ruby
- Riak community provides many other drivers :
  - Perl
  - Clojure
  - … `http://wiki.basho.com/display/RIAK/Community-Developed+Libraries+and+Projects`

Thank you

Take a look at
http://wiki.basho.com

Mail : germain.maurice _A_ **linkfluence.net**