riak

web-shaped data storage system

NYC NoSQL 2009
Bryan Fink <bryan@basho.com>

What is riak?

- a document-oriented database
- a decentralized key-value store
  (core ops: get/put/delete)
- a distributed, fault-tolerant storage solution
- nosql, http, json, rest, scalable...

What is riak?

Influences: Amazon's Dynamo

CAP Theorem

The Web

Ops Experience

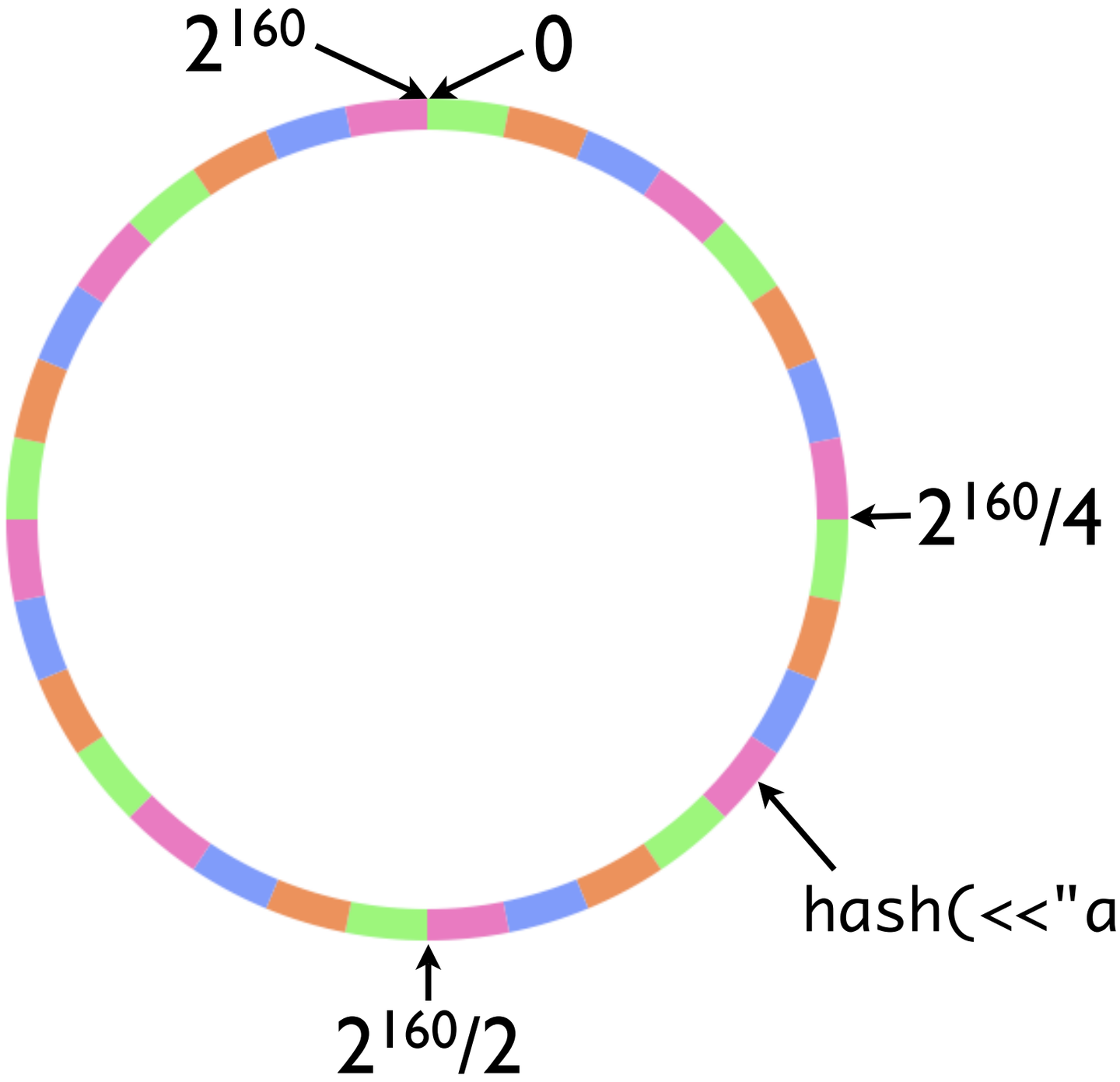
Basic N/R/W

N = number of replicas to store

R = number of replicas needed for a read

W = number of replicas needed for a write

N value



node 0

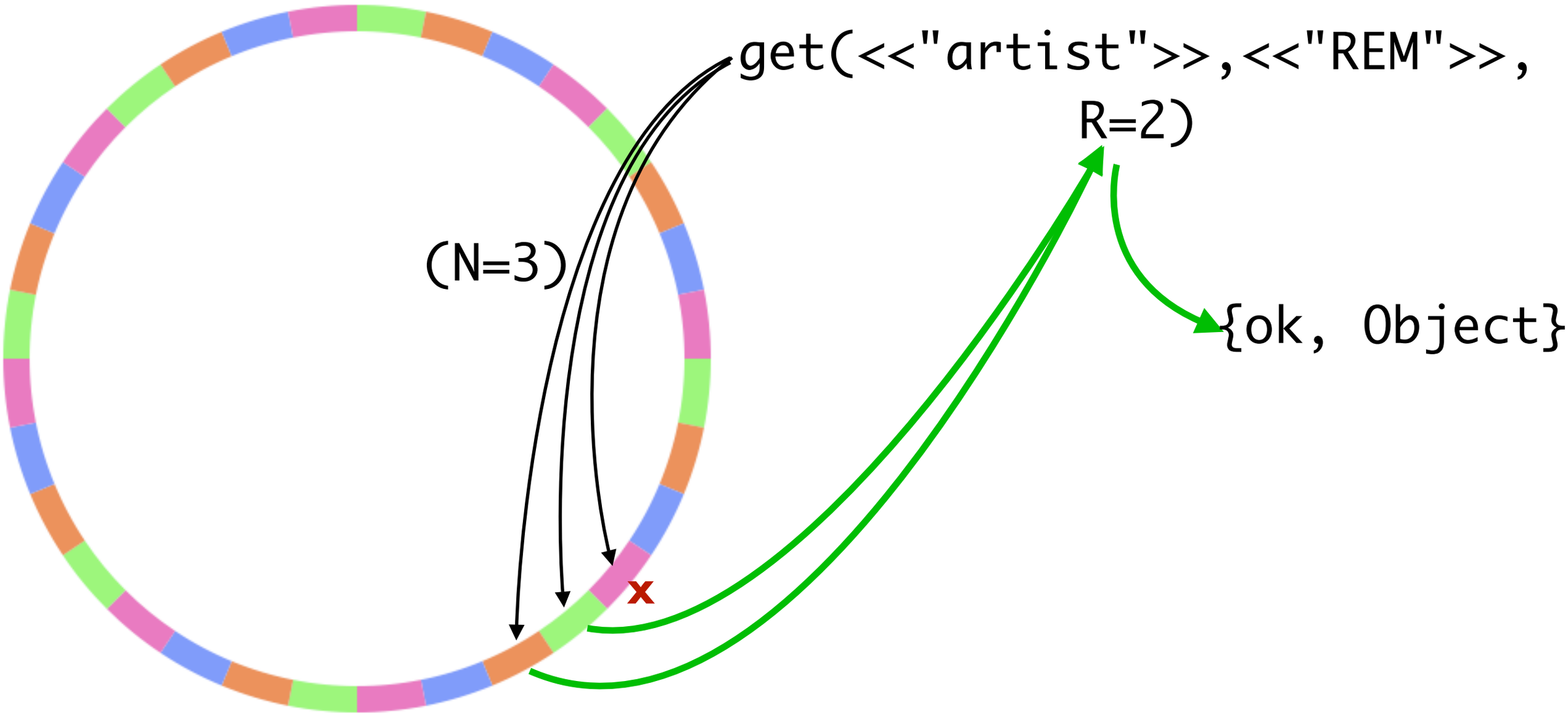
node 1

node 2

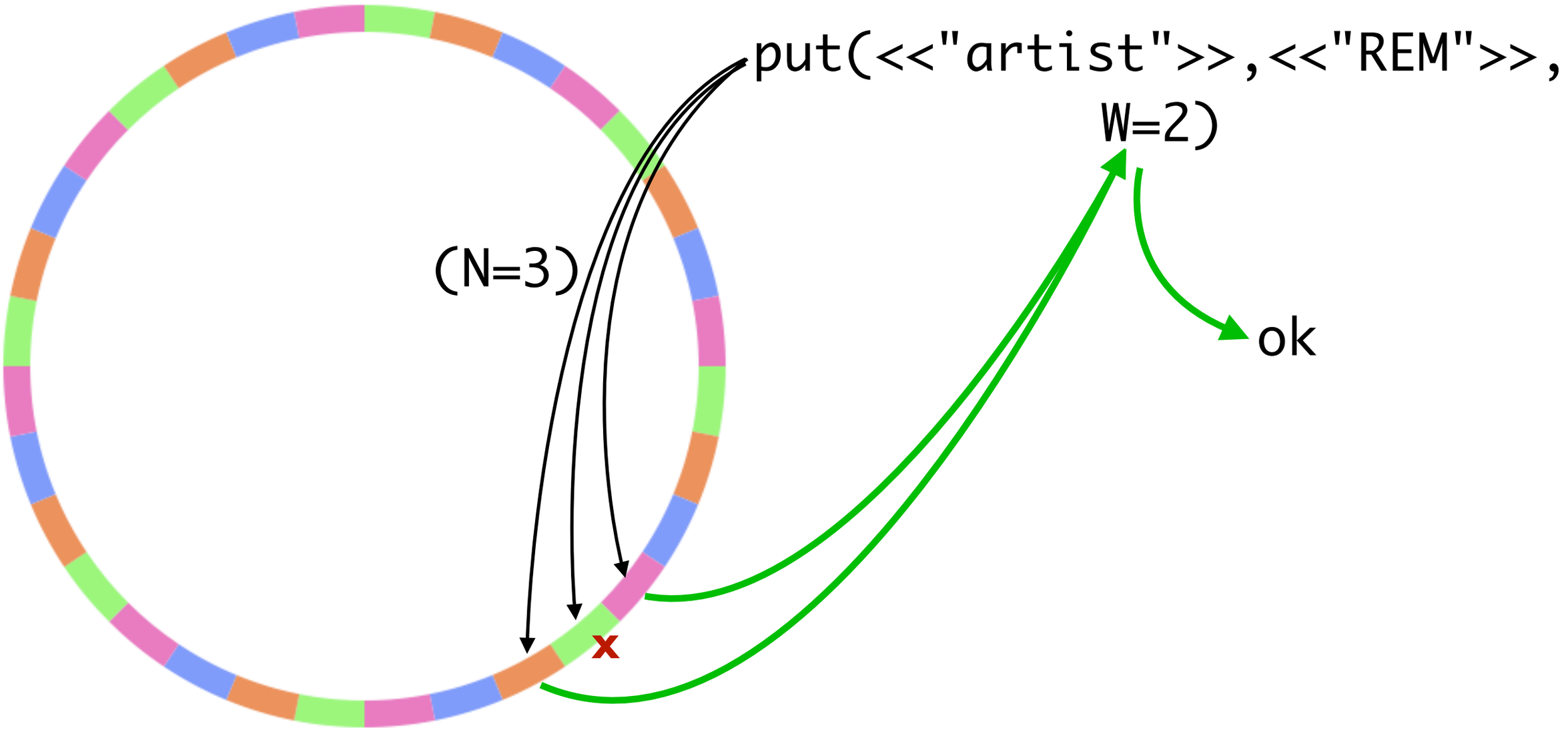
node 3

```
hash(<<"artist">>,<<"REM">>)
```

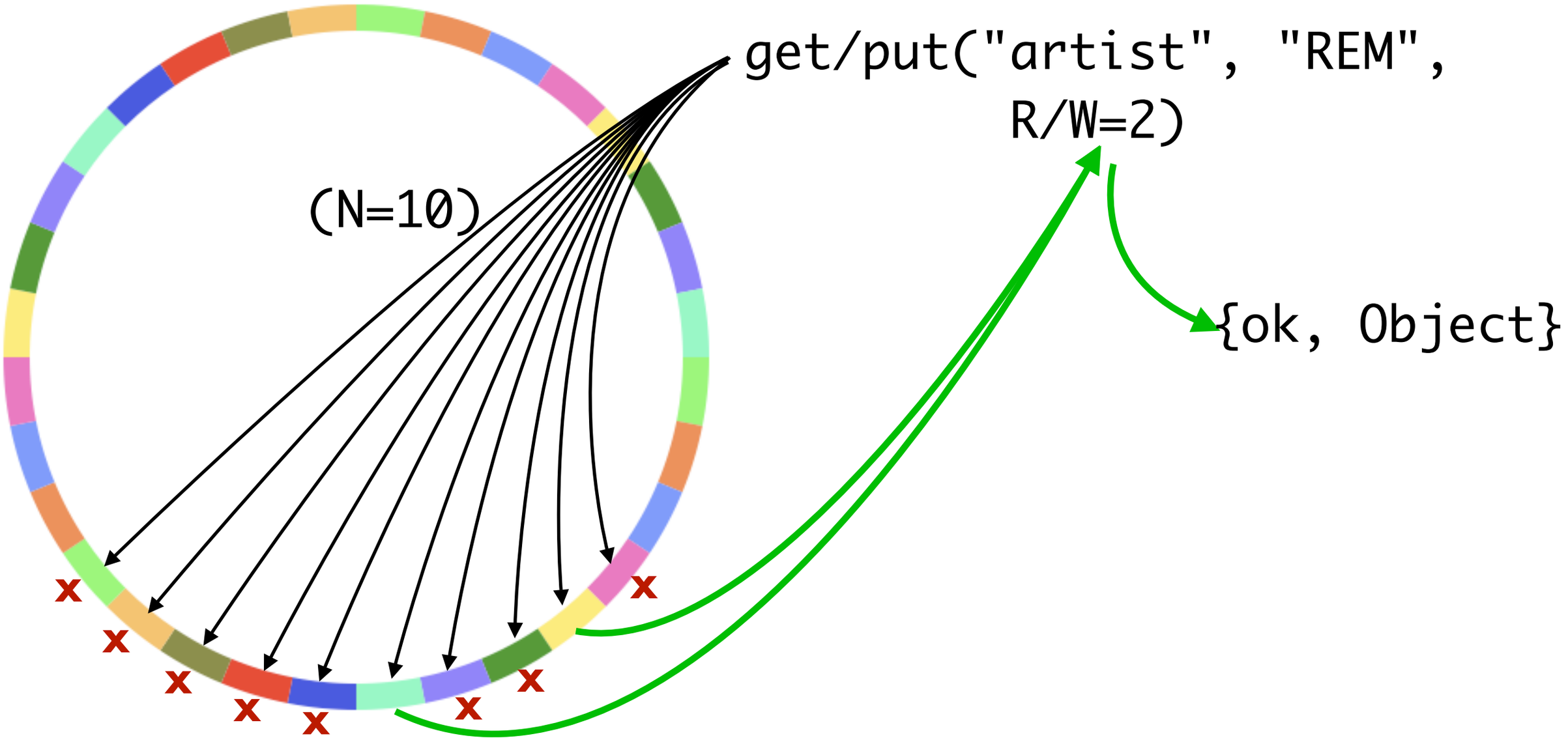
R value



W value



N=10, R/W=2



Basic CAP

Consistency

Availability

Partition tolerance

Basic CAP

Consistency

Availability

Partition tolerance

"Pick two."

Basic CAP

Consistency

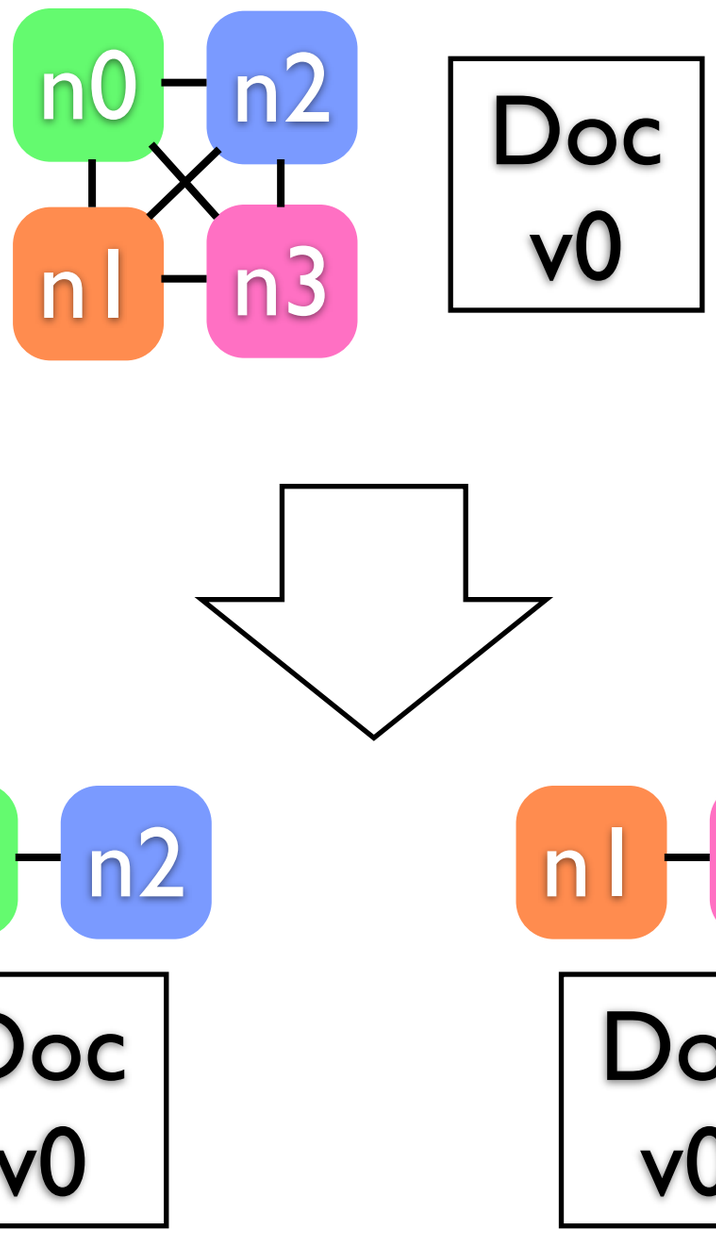
Availability

Partition tolerance

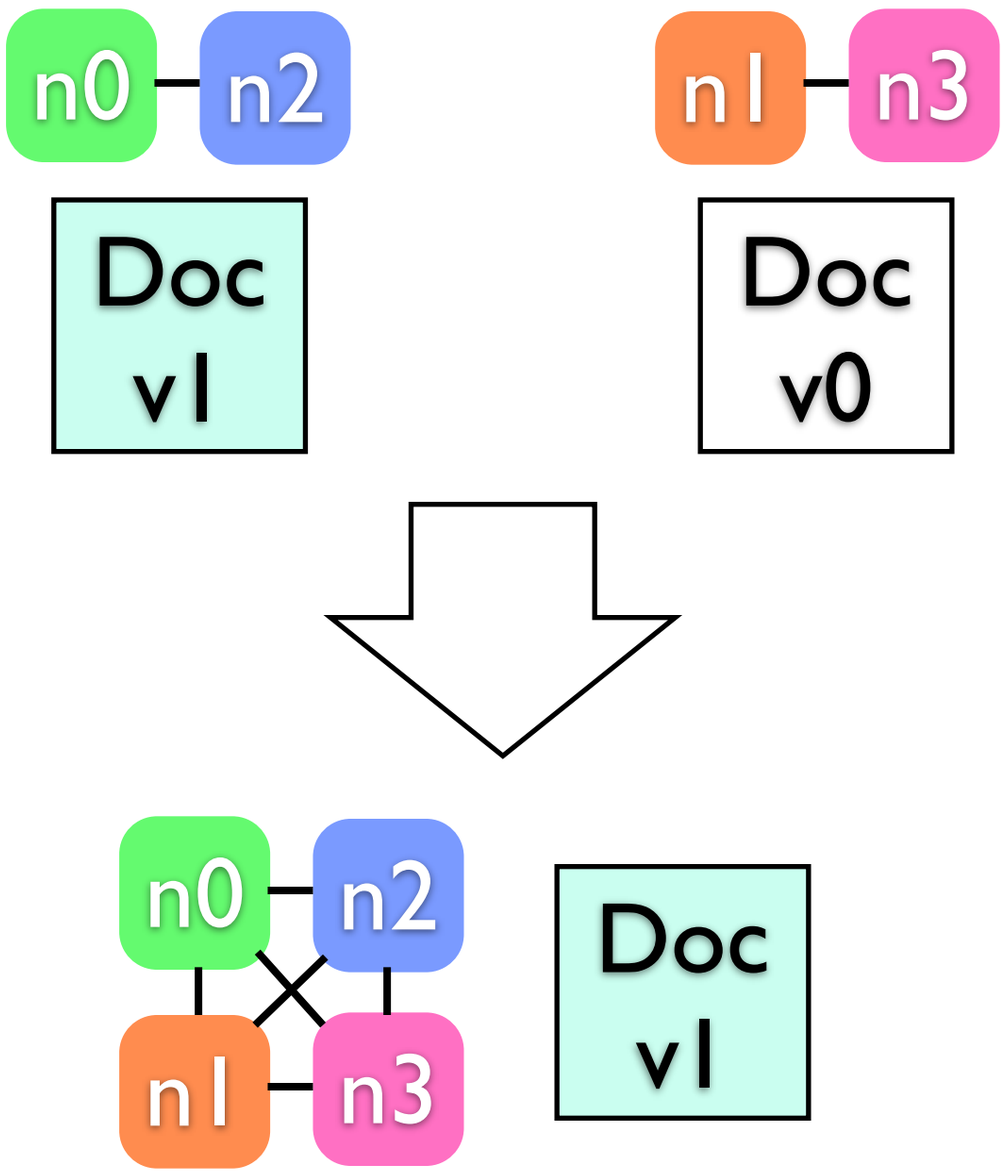
~~"Pick two."~~

"Choose your own levels."

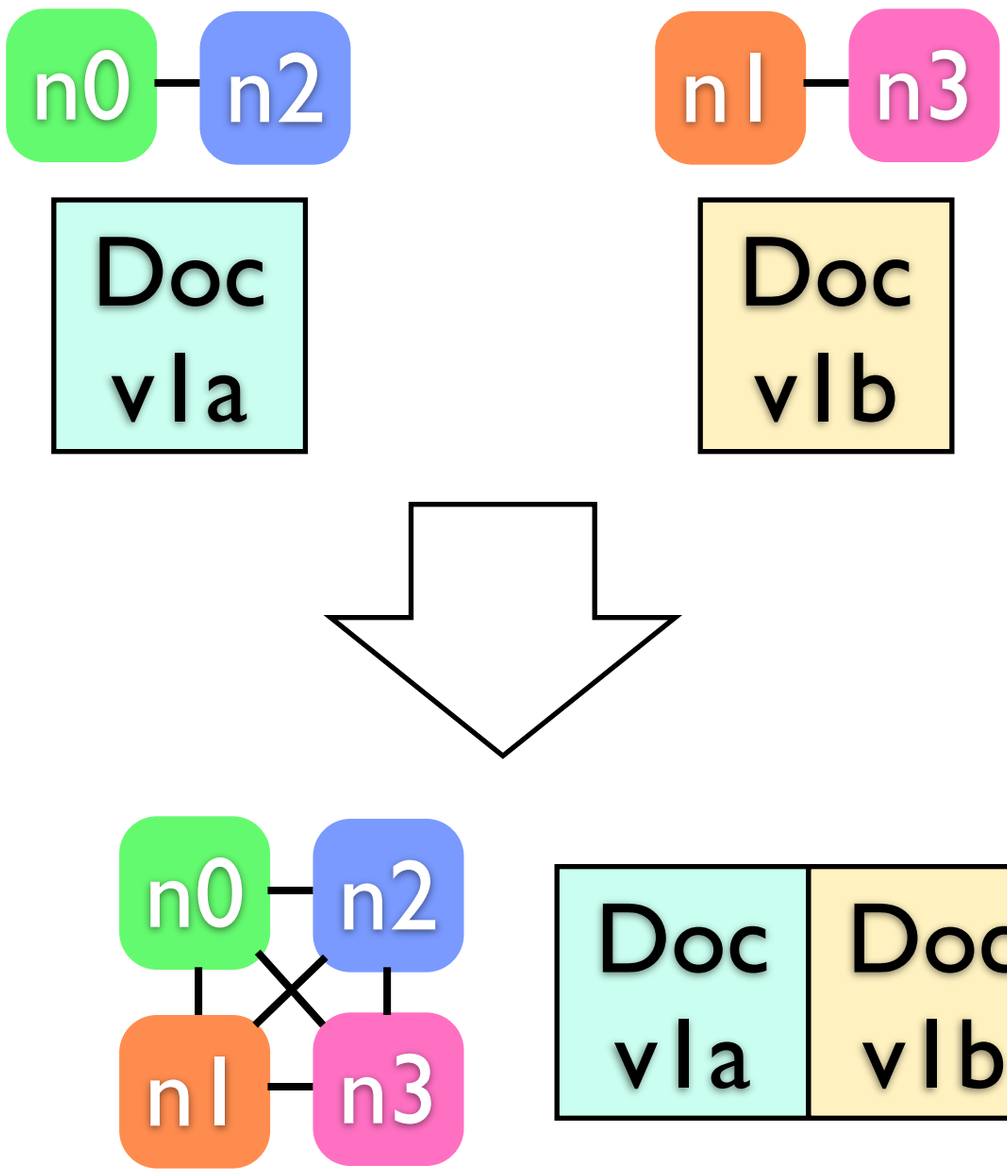
Net Split



Net Split



Net Split



```
%% create client
{ok, C} = riak:client_connect('riak@127.0.0.1').

%% create object
O = riak_object:new(<<"artist">>, <<"REM">>,
                    [{albums, 2},{members,[...]}]).

%% store object
ok = C:put(O, 2).

%% get object back
{ok, O1} = C:get(<<"artist">>, <<"REM">>, 2),

%% update object
U = riak_object:update_value(O1, [{albums, 3},...]),
ok = C:put(U, 2).
```

```
from jiak import *
C = JiakClient("127.0.0.1",8098)
C.set_bucket_schema("artist", ["albums","members"])

O = JiakObject("artist", "REM")
O.object["albums"] = 2
O.object["members"] = ["Buck","Stipe","Mills","Berry"]
C.store(O)

O1 = C.fetch("artist", "REM")

O1.object["albums"] = 3
C.store(O1)
```

```
require 'jiak'
jc = JiakClient.new('127.0.0.1', 8098)
jc.set_bucket_schema('artist',
{'allowed_fields'=>['albums','members']})

o ={'bucket'=>'artist',
    'key'   =>'REM',
    'object'=>{'albums' =>2,
               'members'=>['Buck','Stipe','Mills','Berry']},
    'links' =>[]}

jc.store(o)

o1 = jc.fetch('artist', 'REM')

o1['object']['albums'] = 3
jc.store(o1)
```

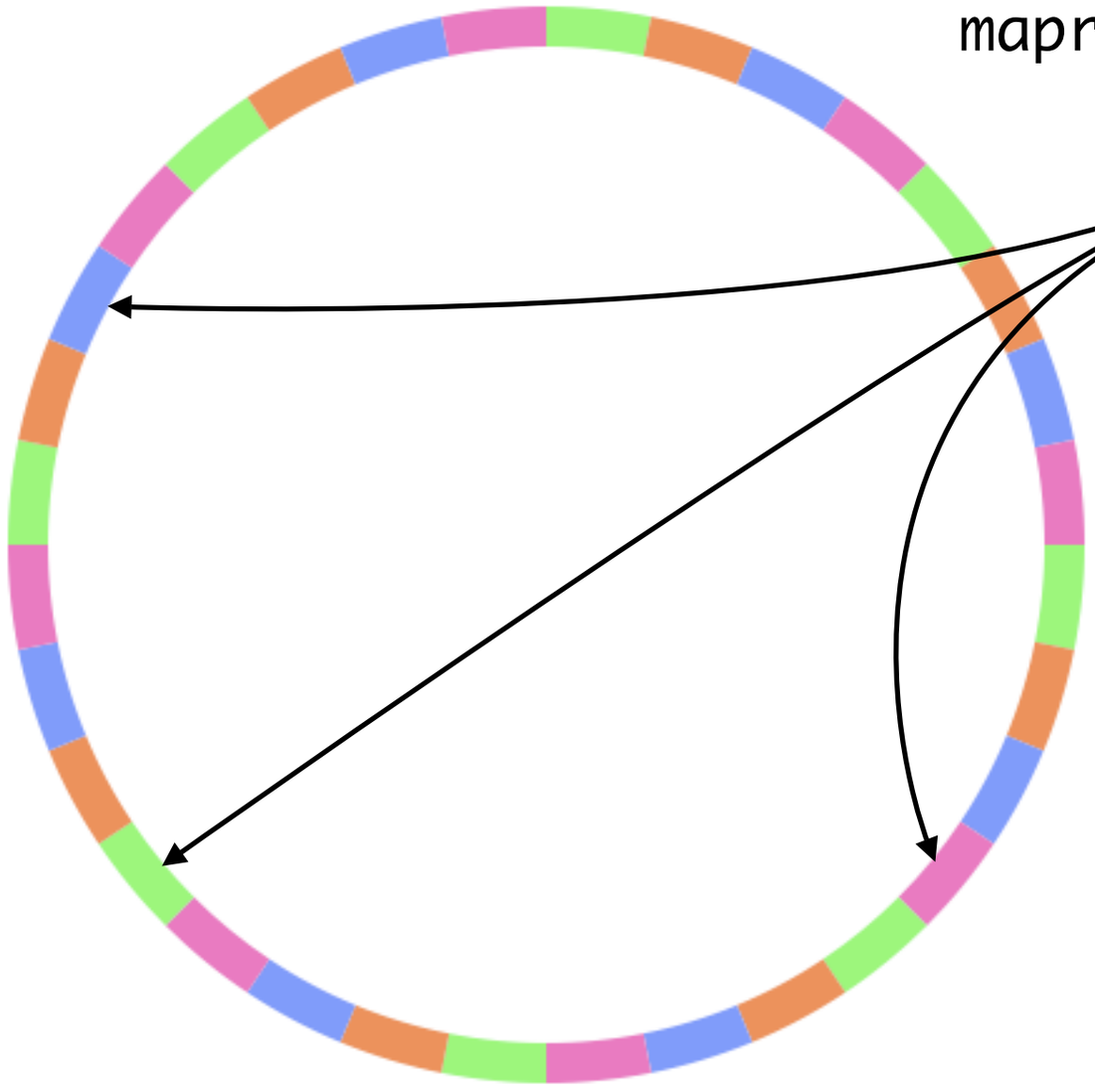
```
JiakClient client = new JiakClient("127.0.0.1", "8098");

ArrayList<String> allowedFields = new ArrayList<String>();
allowedFields.add("albums", "members");
client.setBucketSchema("artist", allowedFields, null, null, null);

JiakObject obj1 = new JiakObject("artist", "REM");
obj1.set("albums", 2);
obj1.set("members", ...);

client.store(obj1);

JiakObject obj2 = client.fetch("artist", "REM");

obj2.set("albums", 3);
client.store(obj2);
```

```
$JC = new JiakClient("127.0.0.1", 8098);
$JC->set_bucket_schema("artist", array("albums", "members"));

$obj = new JiakObject("artist", "REM");
$obj->set("albums", "2");
$obj->set("members",array("Buck","Stipe","Mills","Berry"));

$JC->store($obj);

$obj1 = $JC->fetch("artist", "REM");

$obj1->set("albums", "3");
$JC->store($obj1);
```

```
C = new JiakClient('http://127.0.0.1:8098/jiak/');
C.setBucketSchema('artist',
                  {'allowed_fields':['albums','members']});

O = {'bucket':'artist',
     'key':'REM',
     'object':{'albums':2,
               'members':['Buck','Stipe','Mills','Berry']},
     'links':[]};

C.store(O);

C.fetch('artist', 'REM', function(O1) {
    O1.object.albums = 3;
    C.store(O1);
});
```

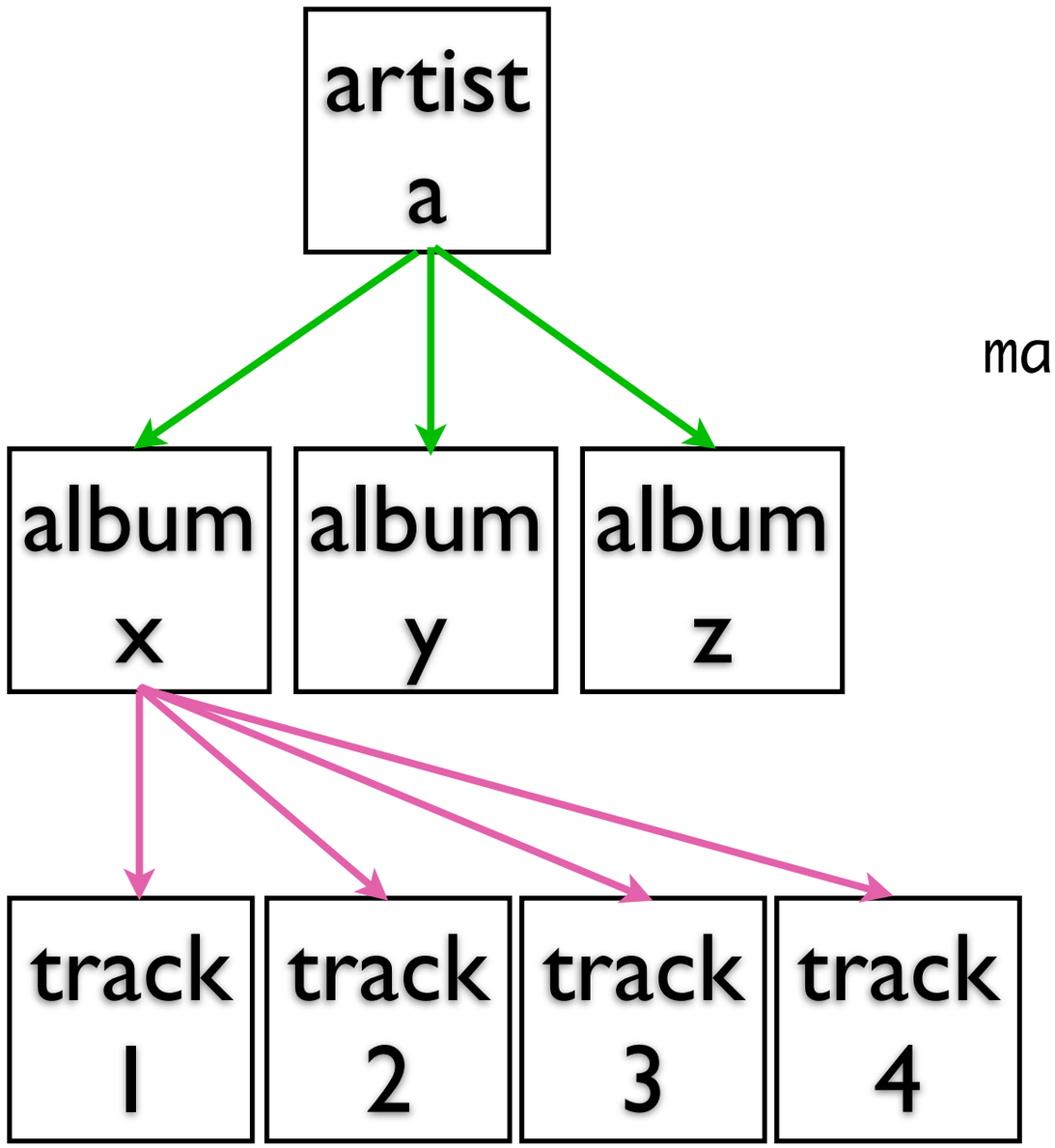
```
curl -X PUT http://127.0.0.1:8098/jiak/artist \
     -H "Content-type: application/json" \
     --data "{\"schema\":{\"allowed_fields\":[\"albums
\",\"members\"],\"required_fields\":[],\"read_mask\":
[\"albums\",\"members\"],\"write_mask\":[\"albums\",
\"members\"]}}"

curl -X PUT http://127.0.0.1:8098/jiak/artist/REM \
     -H "Content-type: application/json" \
     --data "{\"bucket\":\"artist\",\"key\":\"REM\",
\"object\":{\"albums\":2,\"members\":[\"Buck\",\"Stipe
\",\"Mills\",\"Berry\"]},\"links\":[]}"

curl http://127.0.0.1:8098/jiak/artist/REM
```

Map/Reduce



```
mapred([{<<"artist">>,<<"REM">>},
        {<<"artist">>,...},...],
       [{map,
         {modfun,artist,member_count},
         none,false},
        {reduce,
         {qfun,fun(L,_,_) ->
                   lists:unique(L)
               end},
         none, true}]).
```

Links




```
mapred([{<<"artist">>,<<"REM">>}],
       [{link,<<"album">>,'_',false},
        {link,<<"track">>,'_',false},
        {map,{modfun,track,name},
         none, true}]).
```

http://host/jiak/artist/REM/album,_,_/track,_,_

Thanks!

Lots more:

pluggable backends

eventing system

monitoring

inter-cluster replication

http://riak.basho.com

riak-users@lists.basho.com

basho