

Apache
CouchDB
In 20 Minutes

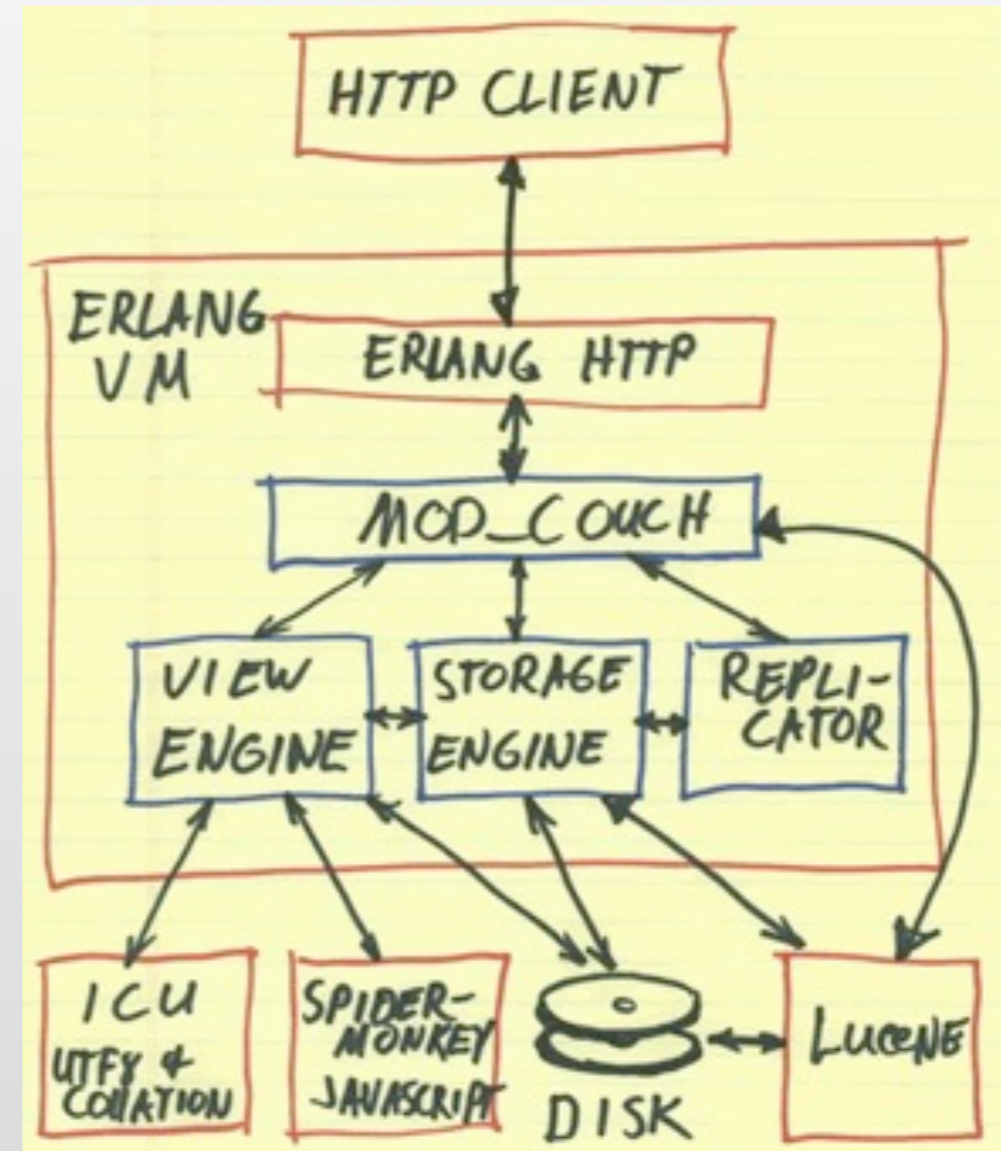
Relax

Who's Talking?

- Jan Lehnardt
- jan@apache.org / @janl
- CouchDB Developer
- Web Technologist
- Director, couch.io.

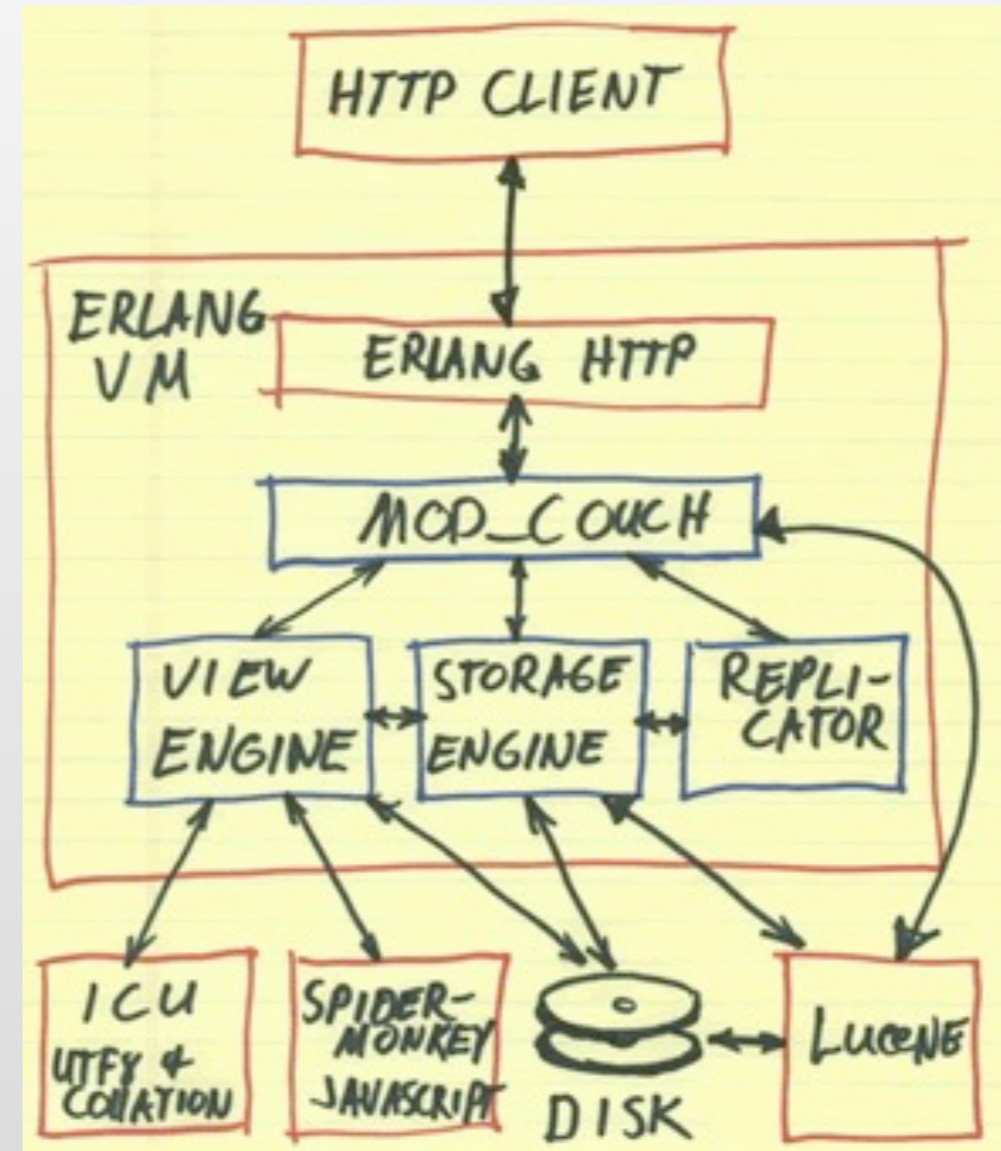
- Schema Free (JSON)
- Document Oriented,
Not Relational
- Highly Concurrent
- RESTful HTTP API
- JavaScript Powered
Map/Reduce
- N-Master Replication
- Robust Storage

Features



- **Schema Free (JSON)**
- Document Oriented,
Not Relational
- Highly Concurrent
- RESTful HTTP API
- JavaScript Powered
Map/Reduce
- N-Master Replication
- Robust Storage

Features



Schema Free (JSON)

```
{  
  "_id": "BCCD12CBB",  
  "_rev": "AB764C",  
  
  "type": "person",  
  "name": "Darth Vader",  
  "age": 63,  
  "headware":  
    ["Helmet", "Sombrero"],  
  "dark_side": true  
}
```

Schema Free (JSON)

```
{  
  "_id": "BCCD12CBB",  
  "_rev": "AB764C",  
  
  "type": "person",  
  "name": "Darth Vader",  
  "age": 63,  
  "headware":  
    ["Helmet", "Sombrero"],  
  "dark_side": true  
}
```

Schema Free (JSON)

```
{  
  "_id": "BCCD12CBB",  
  "_rev": "AB764C",  
  
  "type": "person",  
  "name": "Darth Vader",  
  "age": 63,  
  "headware":  
    ["Helmet", "Sombrero"],  
  "dark_side": true  
}
```

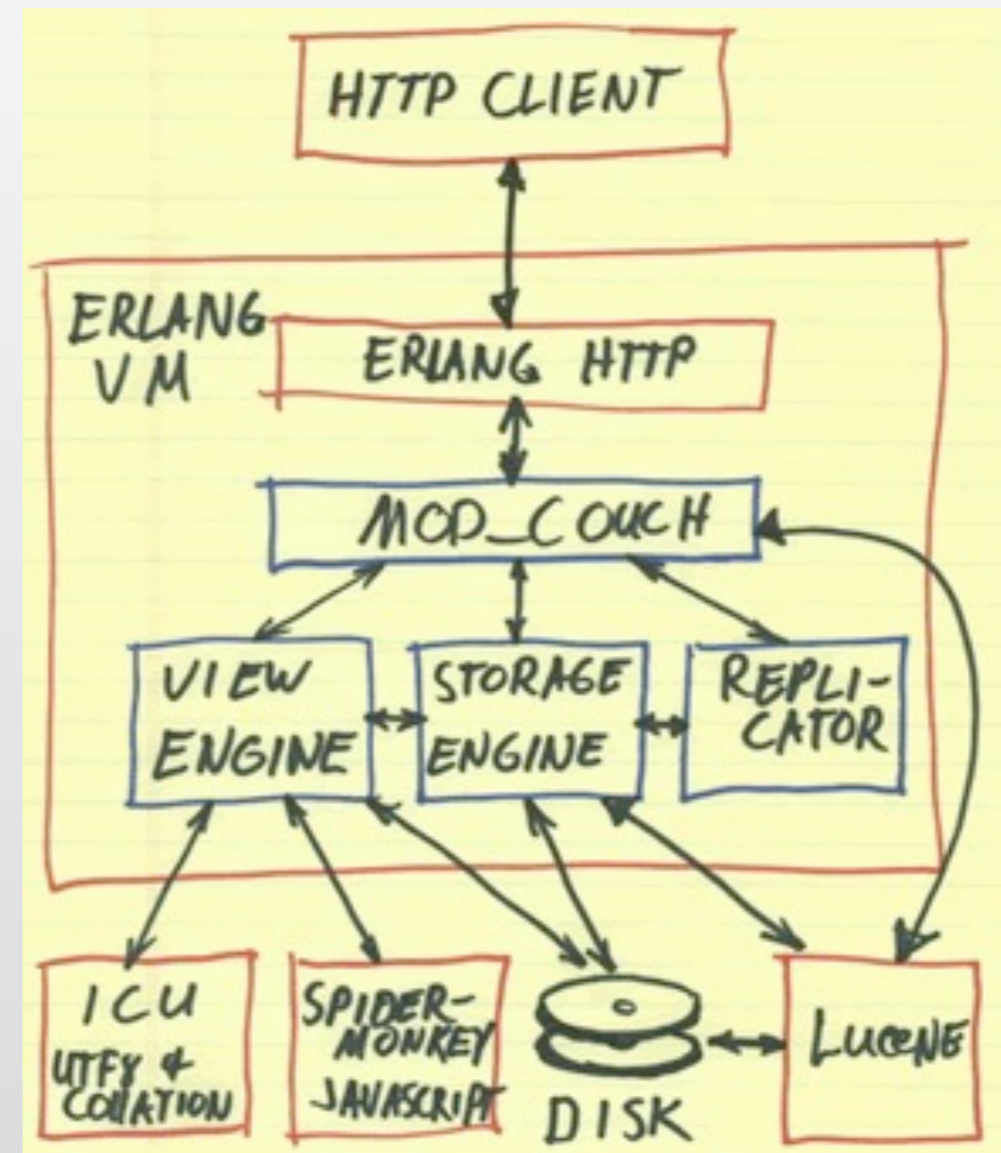
Schema Free (JSON)

```
{
  "_id": "BCCD12CBB",
  "_rev": "AB764C",

  "type": "person",
  "name": "Darth Vader",
  "age": 63,
  "headware":
    ["Helmet", "Sombrero"],
  "dark_side": true
}
```

- Schema Free (JSON)
- **Document Oriented, *Not Relational***
- Highly Concurrent
- RESTful HTTP API
- JavaScript Powered Map/Reduce
- N-Master Replication
- Robust Storage

Features



Document Oriented

Not Relational

- Documents in the Real World™
 - Bills, letters, tax forms...
 - Same type != same structure
 - Can be out of date (so what?)
 - No references

Document Oriented

Not Relational

- Documents in the Real World™

- Bills, letters, tax forms...

Natural Data

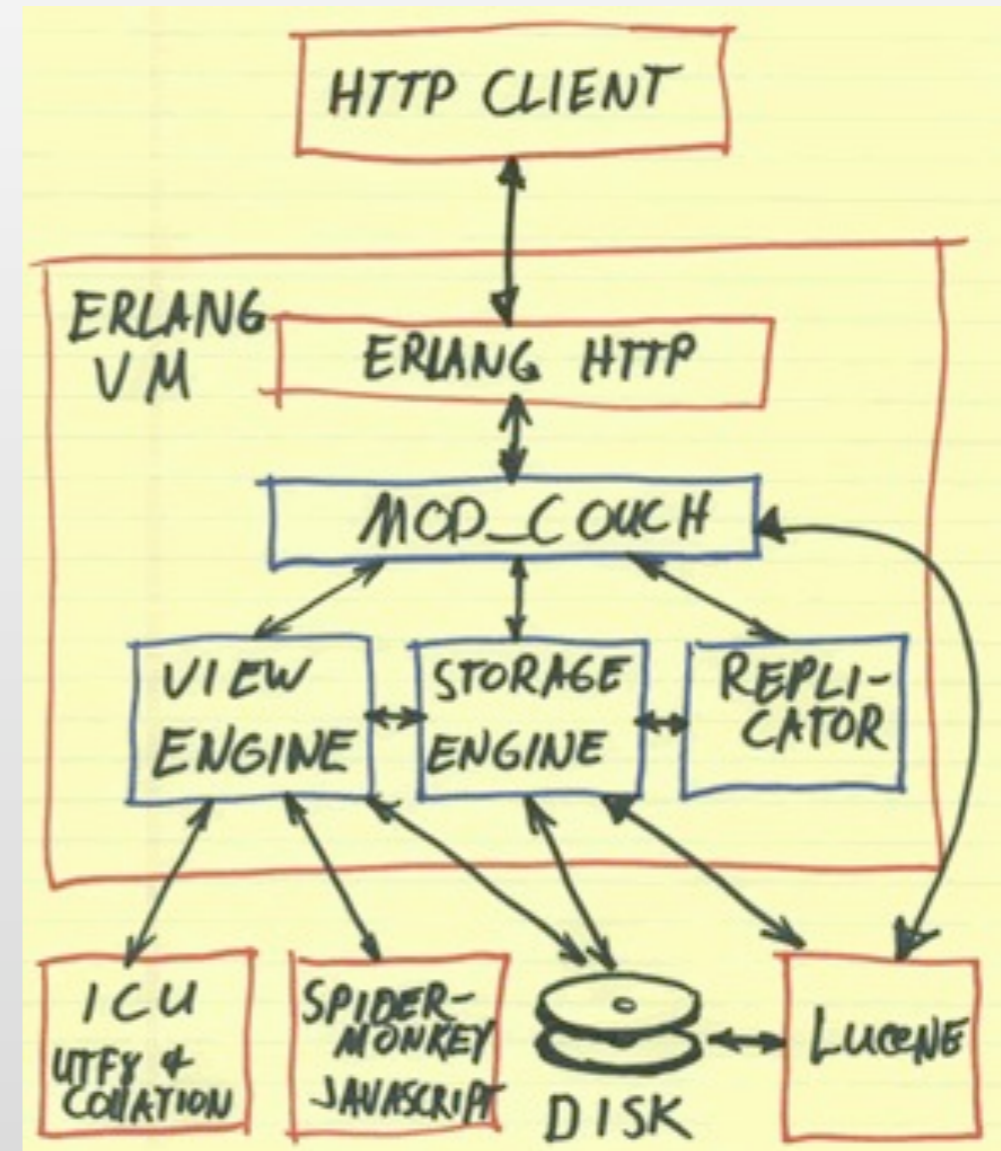
- Same type != same structure

Behaviour

- Can be out of date (so what)
- No references

- Schema Free (JSON)
- Document Oriented,
Not Relational
- **Highly Concurrent**
- RESTful HTTP API
- JavaScript Powered
Map/Reduce
- N-Master Replication
- Robust Storage

Features



Highly Concurrent

Erlang Praising

Highly Concurrent

Erlang Praising

- Parallelism built into the language

Highly Concurrent

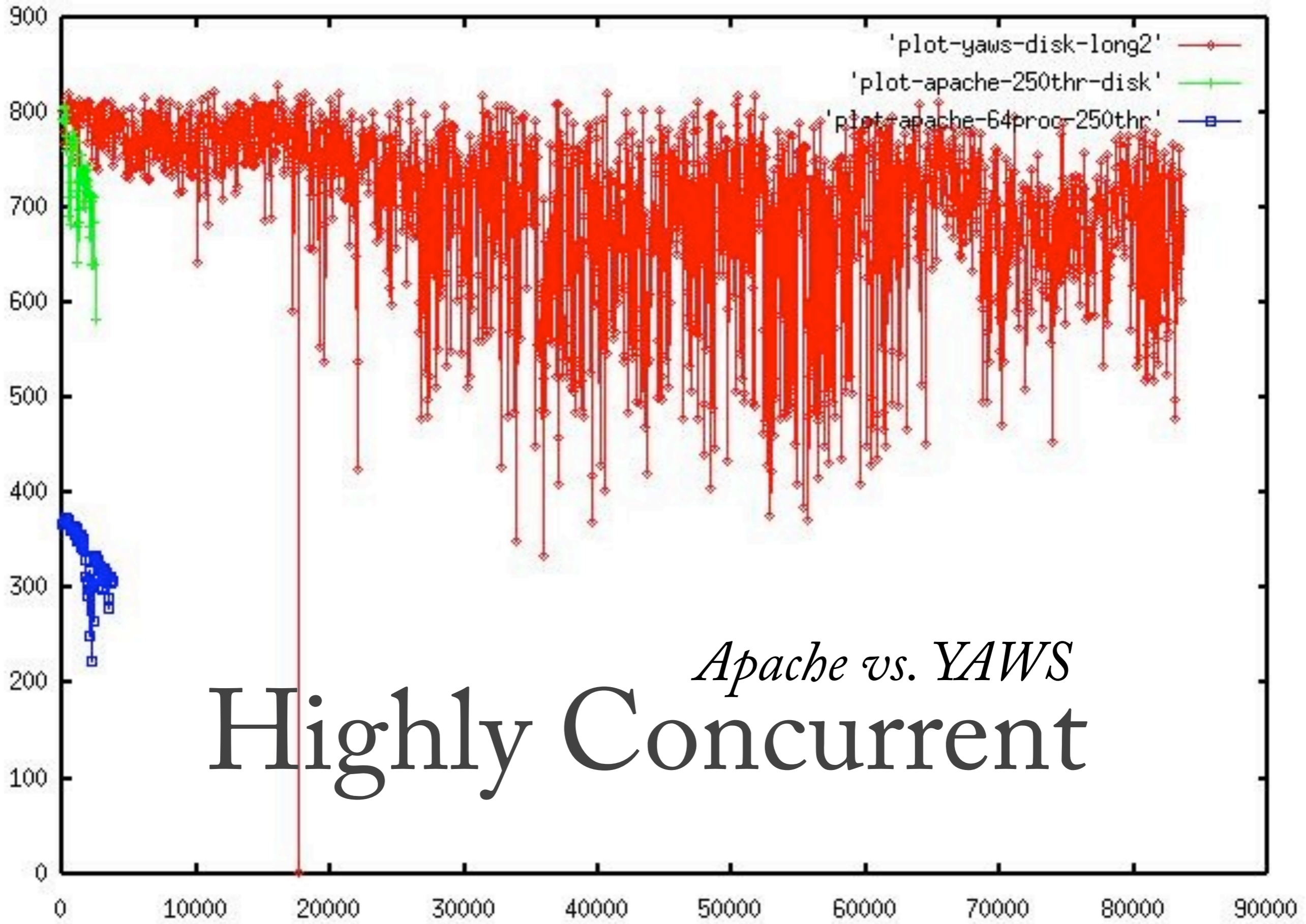
Erlang Praising

- Parallelism built into the language
- Makes programming multi-core easy(er)

Highly Concurrent

Erlang Praising

- Parallelism built into the language
- Makes programming multi-core easy(er)
- Easy to create fault-tolerant systems

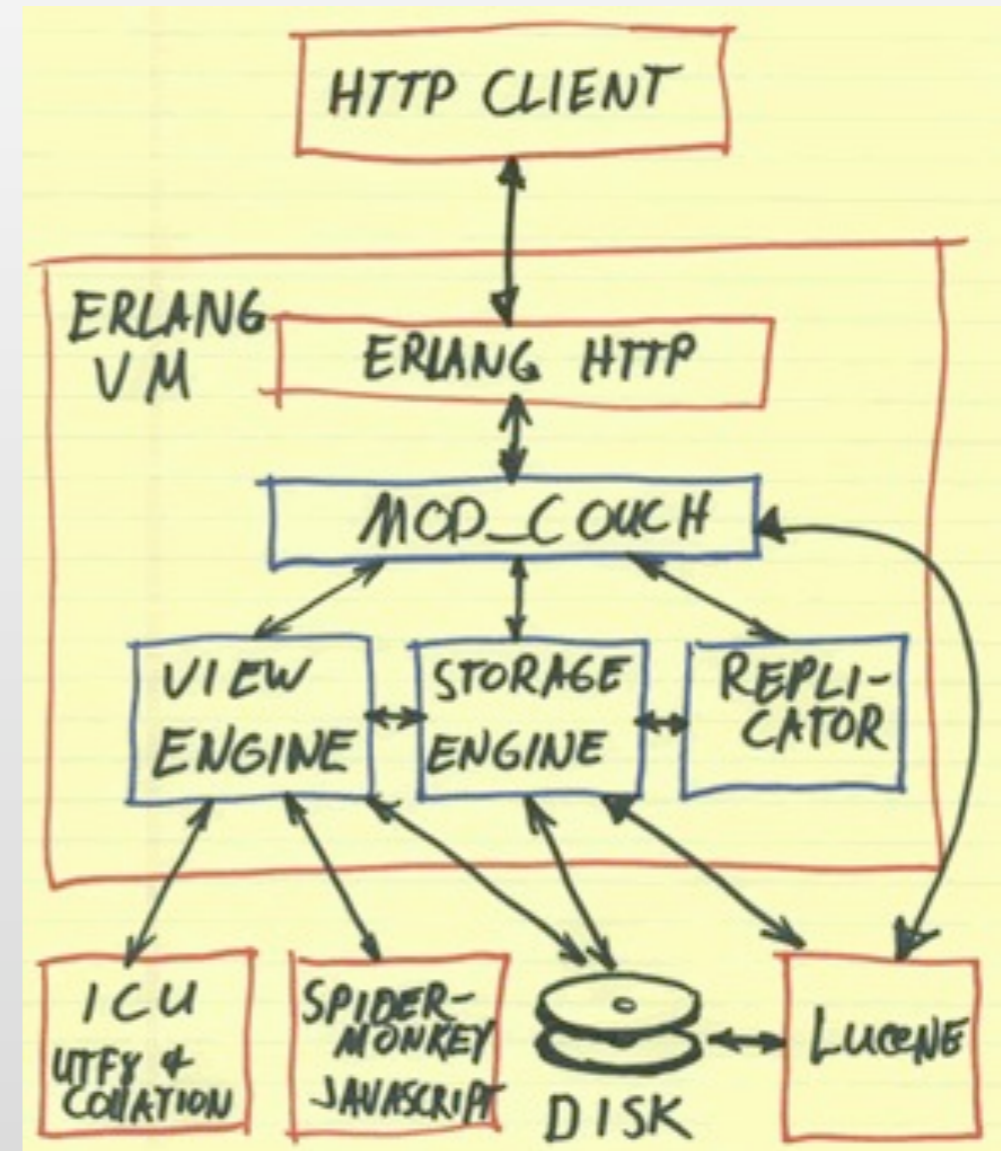


Apache vs. YAWS
Highly Concurrent

0 10000 20000 30000 40000 50000 60000 70000 80000 90000

- Schema Free (JSON)
- Document Oriented,
Not Relational
- Highly Concurrent
- **RESTful HTTP API**
- JavaScript Powered
Map/Reduce
- N-Master Replication
- Robust Storage

Features



RESTful HTTP API CRUD

- **Create**
HTTP PUT /db/mydocid
- **Read**
HTTP GET /db/mydocid
- **Update**
HTTP PUT /db/mydocid
- **Delete**
HTTP DELETE /db/mydocid

RESTful HTTP API

Example

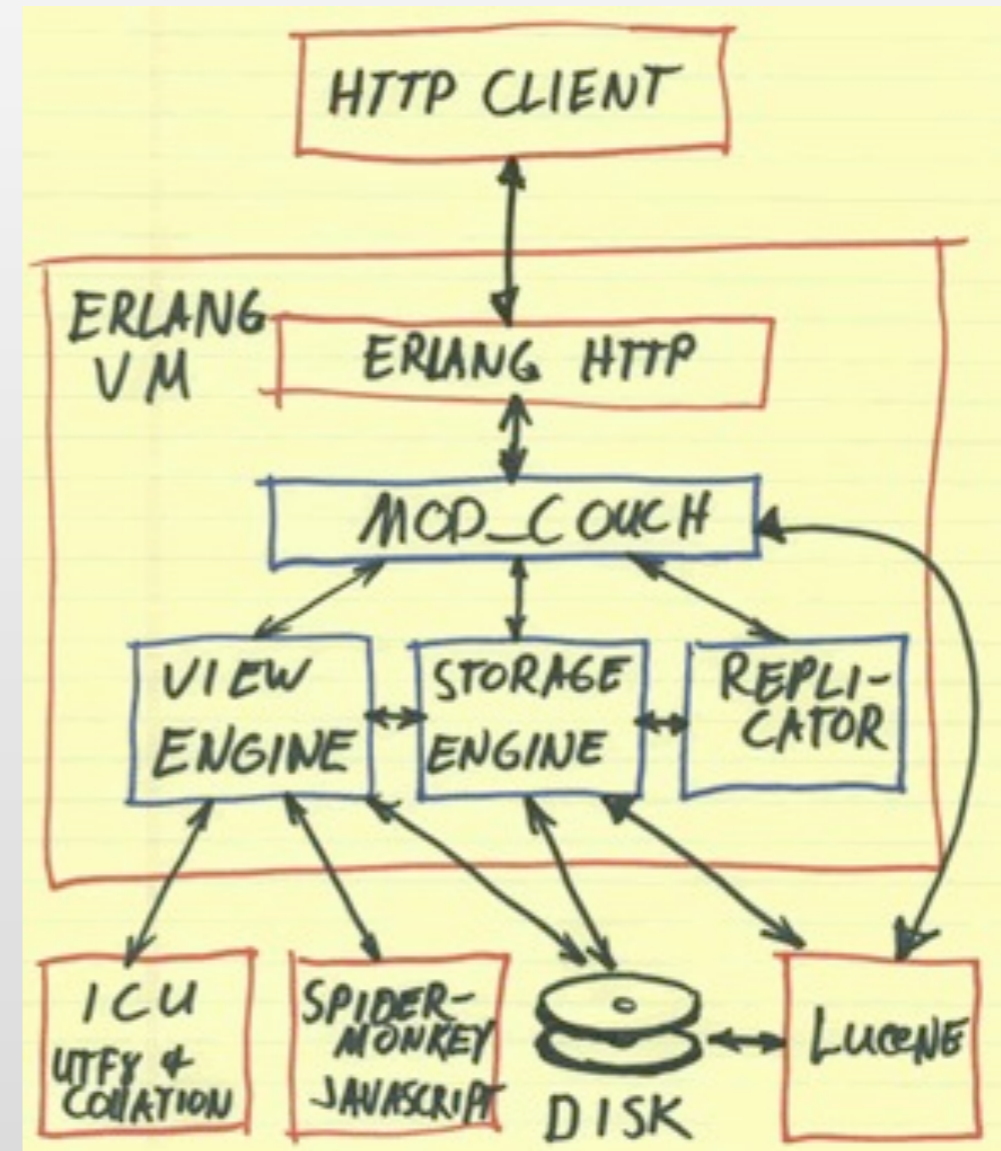
```
couch = CouchRest.database! ("http://  
127.0.0.1:5984/tweets")
```

```
tweets_url = "http://twitter.com/  
statuses/user_timeline.json"
```

```
tweets = http.get(tweets_url)  
couch.bulk_save(tweets)
```

- Schema Free (JSON)
- Document Oriented,
Not Relational
- Highly Concurrent
- RESTful HTTP API
- **JavaScript Powered
Map/Reduce**
- N-Master Replication
- Robust Storage

Features



JavaScript powered Map/Reduce

- Map functions extract data from your documents
- Reduce functions aggregate intermediate values
- The kicker: Incremental b-tree storage

Map Reduce Views

Docs

```
{ "user" : "Chris",  
  "points" : 3 }  
{ "user": "Joe",  
  "points" : 10 }  
{ "user": "Alice",  
  "points" : 5 }  
{ "user": "Mary",  
  "points" : 9 }  
{ "user": "Bob",  
  "points": 7 }
```

Map

```
function(doc) {  
  if (doc.user && doc.points) {  
    emit(doc.user, doc.points);  
  }  
}
```

```
{ "key": "Alice", "value": 5 }  
{ "key": "Bob", "value": 7 }  
{ "key": "Chris", "value": 3 }  
{ "key": "Joe", "value": 10 }  
{ "key": "Mary", "value": 9 }
```

Reduce

```
function(keys, values, rereduce) {  
  return sum(values);  
}
```

```
Alice ... Chris: 15  
Everyone: 34
```

- Parallel processing abstraction . Designed to run across many clusters.
- Iterate over documents emitting key/value pairs. Queries can be run to fetch key ranges.
- Incremental. Subsequent queries are fast.

Map Reduce Views

Docs

```
{ "user" : "Chris",  
  "points" : 3 }  
{ "user": "Joe",  
  "points" : 10 }  
{ "user": "Alice",  
  "points" : 5 }  
{ "user": "Mary",  
  "points" : 9 }  
{ "user": "Bob",  
  "points": 7 }
```

Map

```
function(doc) {  
  if (doc.user && doc.points) {  
    emit(doc.user, doc.points);  
  }  
}
```

```
{ "key": "Alice", "value": 5 }  
{ "key": "Bob", "value": 7 }  
{ "key": "Chris", "value": 3 }  
{ "key": "Joe", "value": 10 }  
{ "key": "Mary", "value": 9 }
```

Reduce

```
function(keys, values, rereduce) {  
  return sum(values);  
}
```

```
Alice ... Chris: 15  
Everyone: 34
```

Sunday, 5 July 2009

- Parallel processing abstraction . Designed to run across many clusters.
- Iterate over documents emitting key/value pairs. Queries can be run to fetch key ranges.
- Incremental. Subsequent queries are fast.

Map Reduce Views

Docs

```
{ "user" : "Chris",  
  "points" : 3 }  
{ "user": "Joe",  
  "points" : 10 }  
{ "user": "Alice",  
  "points" : 5 }  
{ "user": "Mary",  
  "points" : 9 }  
{ "user": "Bob",  
  "points": 7 }
```

Map

```
function(doc) {  
  if (doc.user && doc.points) {  
    emit(doc.user, doc.points);  
  }  
}
```

```
{ "key": "Alice", "value": 5 }  
{ "key": "Bob", "value": 7 }  
{ "key": "Chris", "value": 3 }  
{ "key": "Joe", "value": 10 }  
{ "key": "Mary", "value": 9 }
```

Reduce

```
function(keys, values, rereduce) {  
  return sum(values);  
}
```

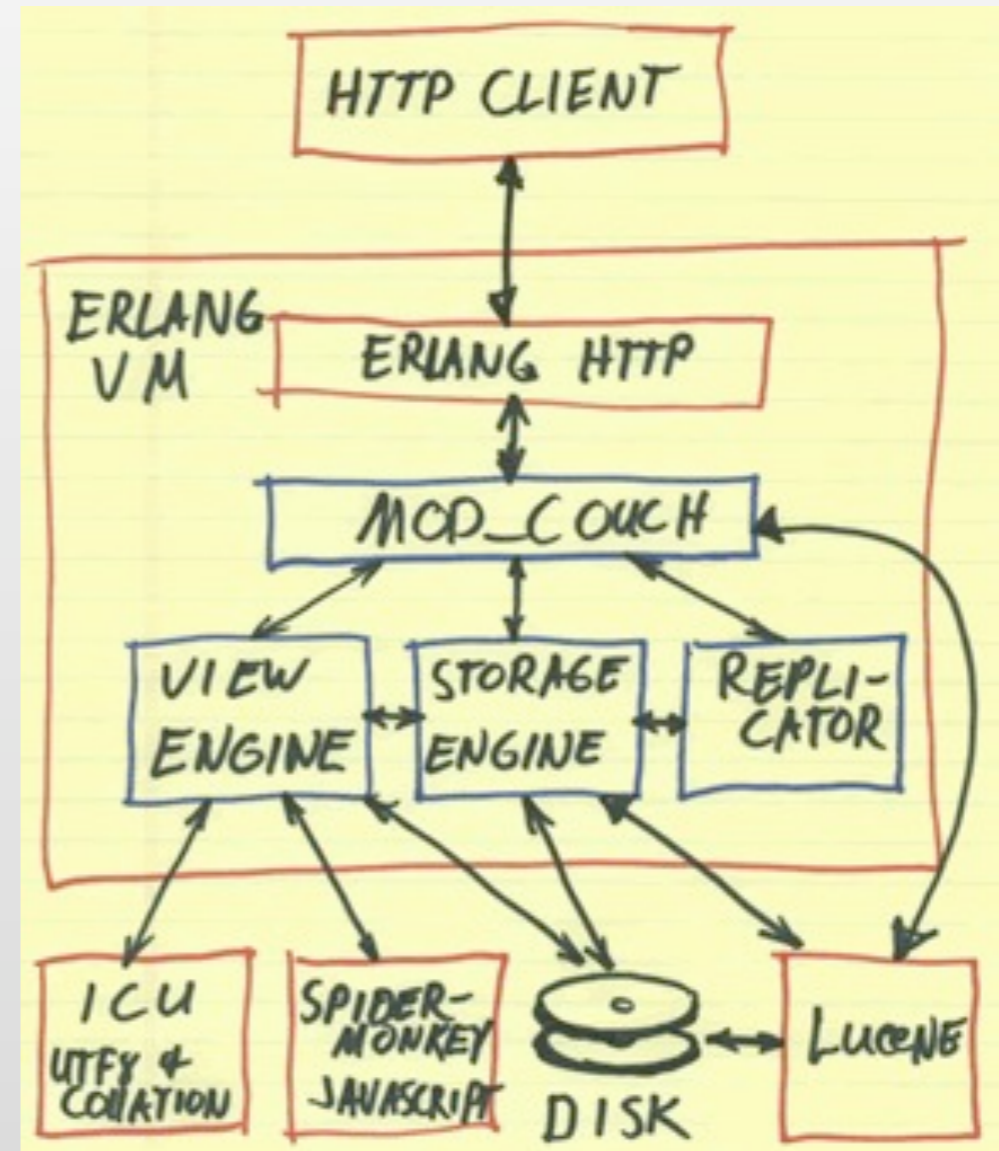
```
Alice ... Chris: 15  
Everyone: 34
```

Sunday, 5 July 2009

- Parallel processing abstraction . Designed to run across many clusters.
 - Iterate over documents emitting key/value pairs. Queries can be run to fetch key ranges.
 - Incremental. Subsequent queries are fast.
- lists

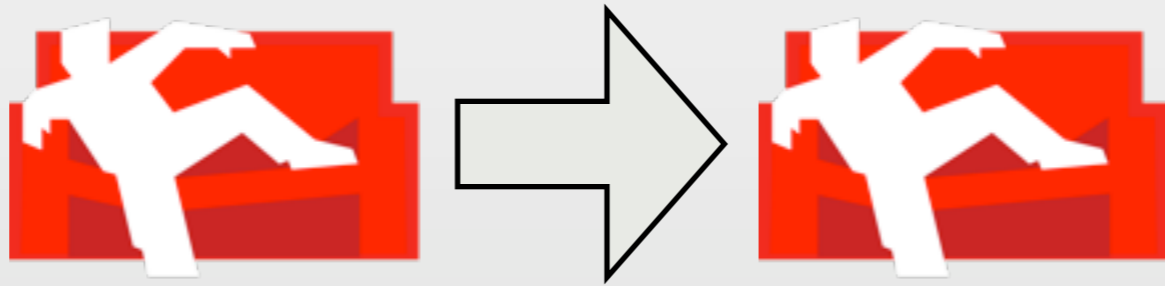
- Schema Free (JSON)
- Document Oriented,
Not Relational
- Highly Concurrent
- RESTful HTTP API
- JavaScript Powered
Map/Reduce
- **N-Master Replication**
- Robust Storage

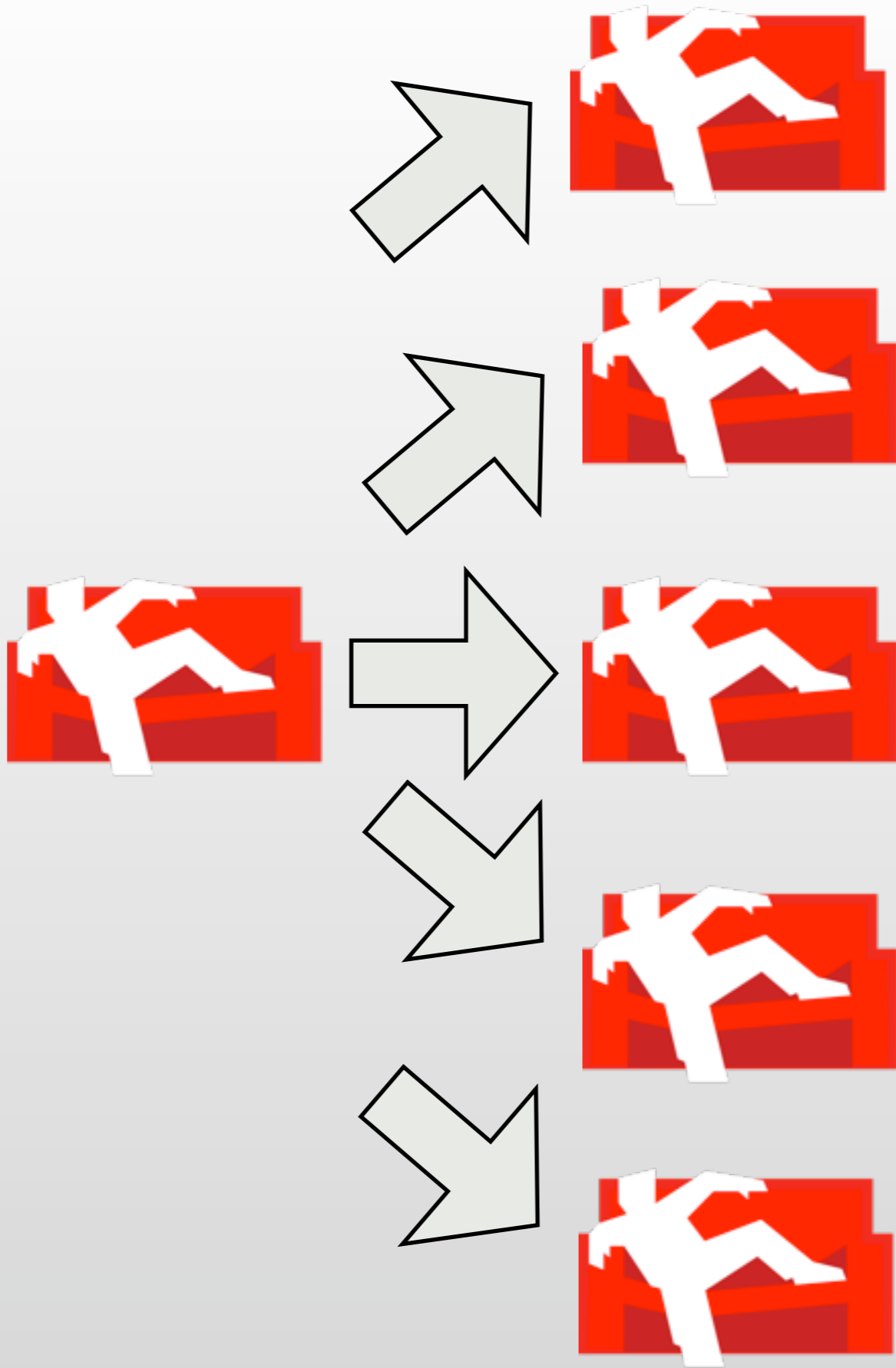
Features

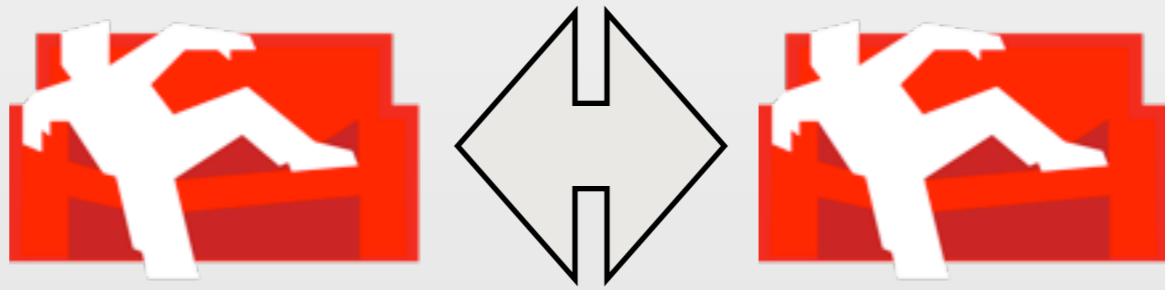


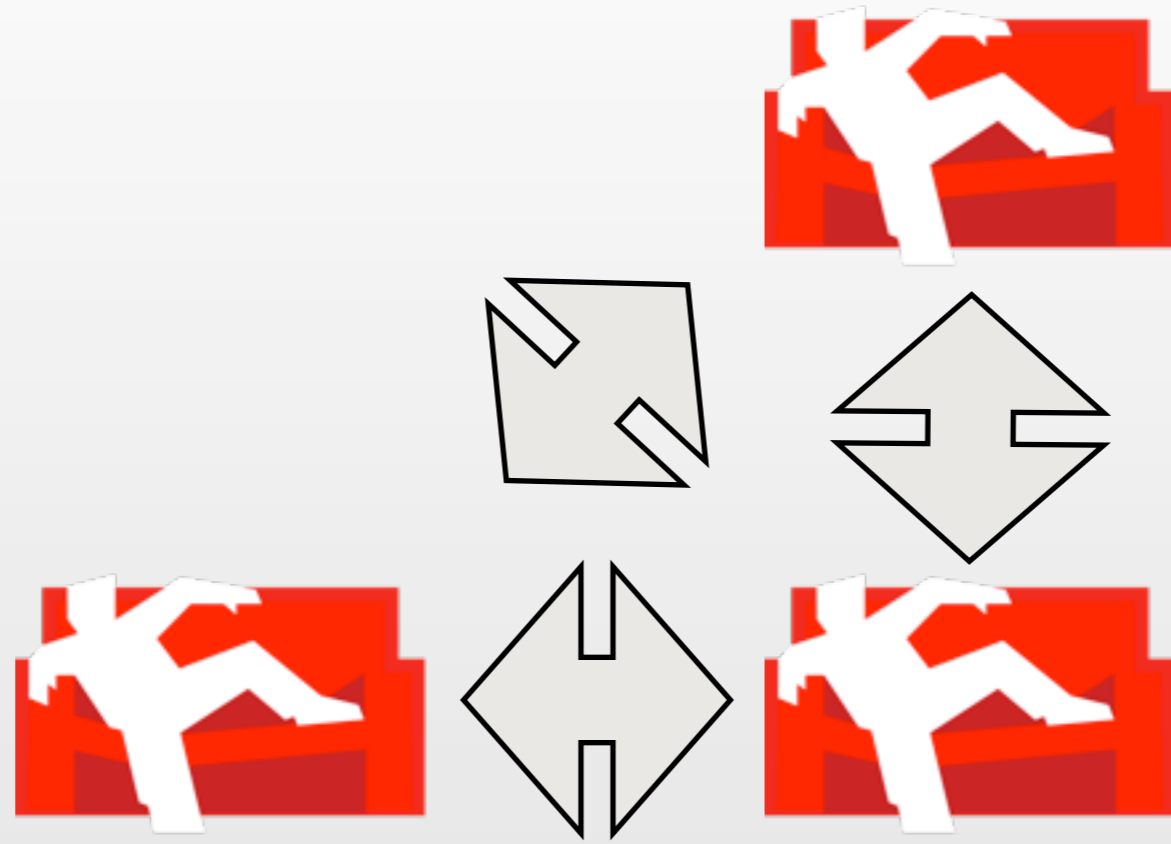


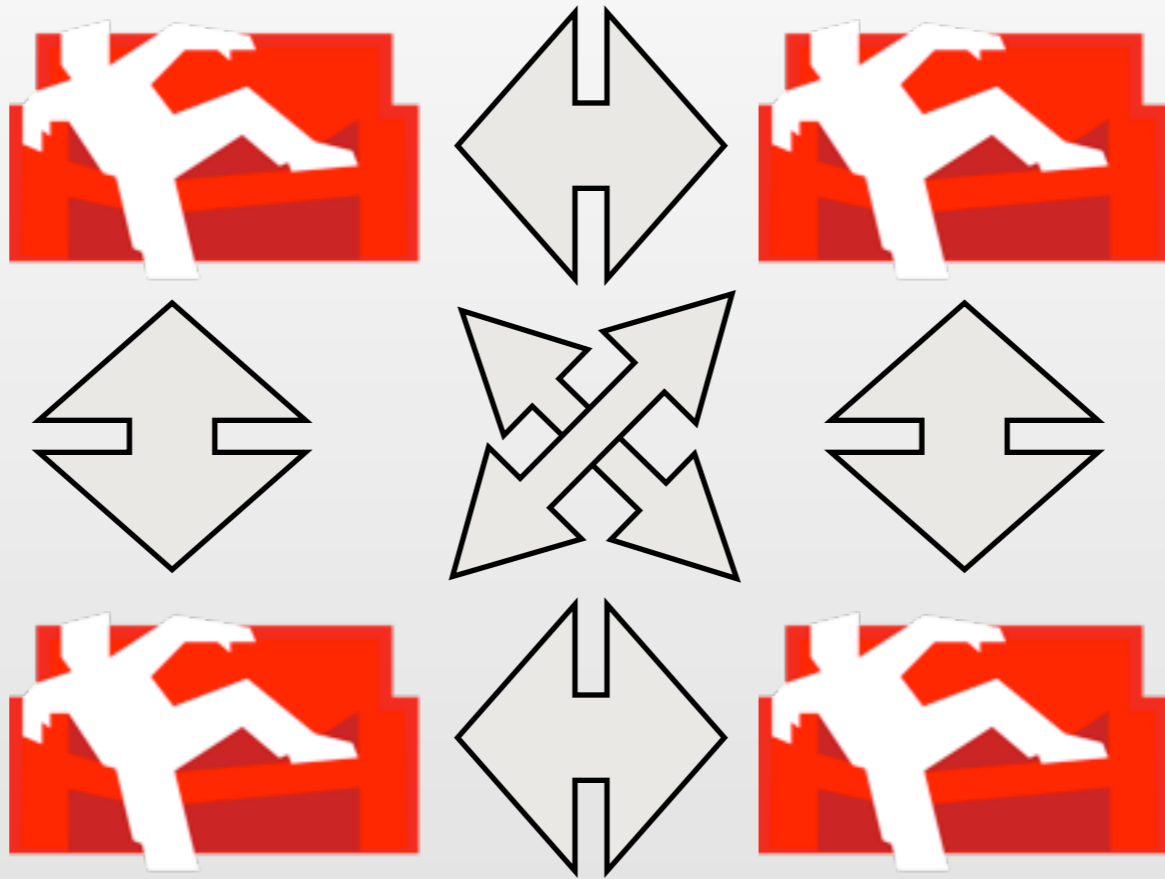


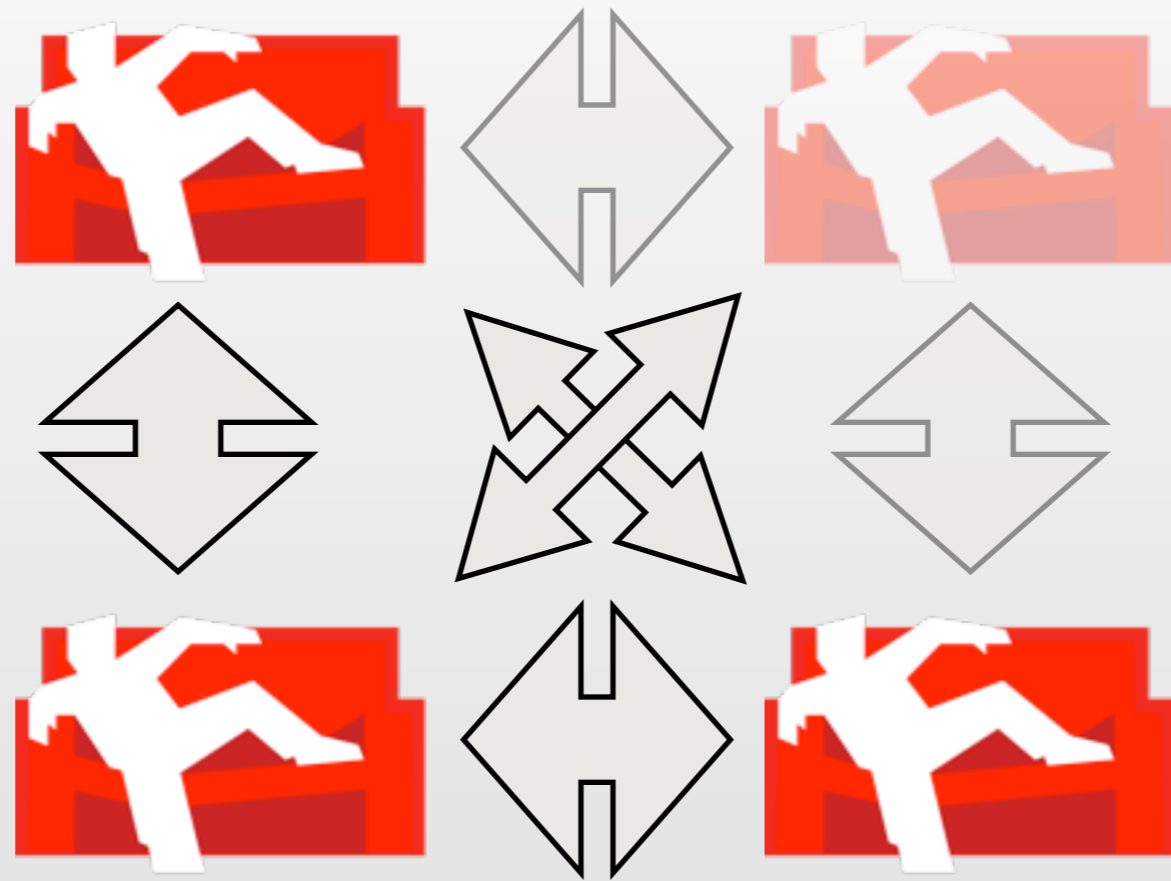


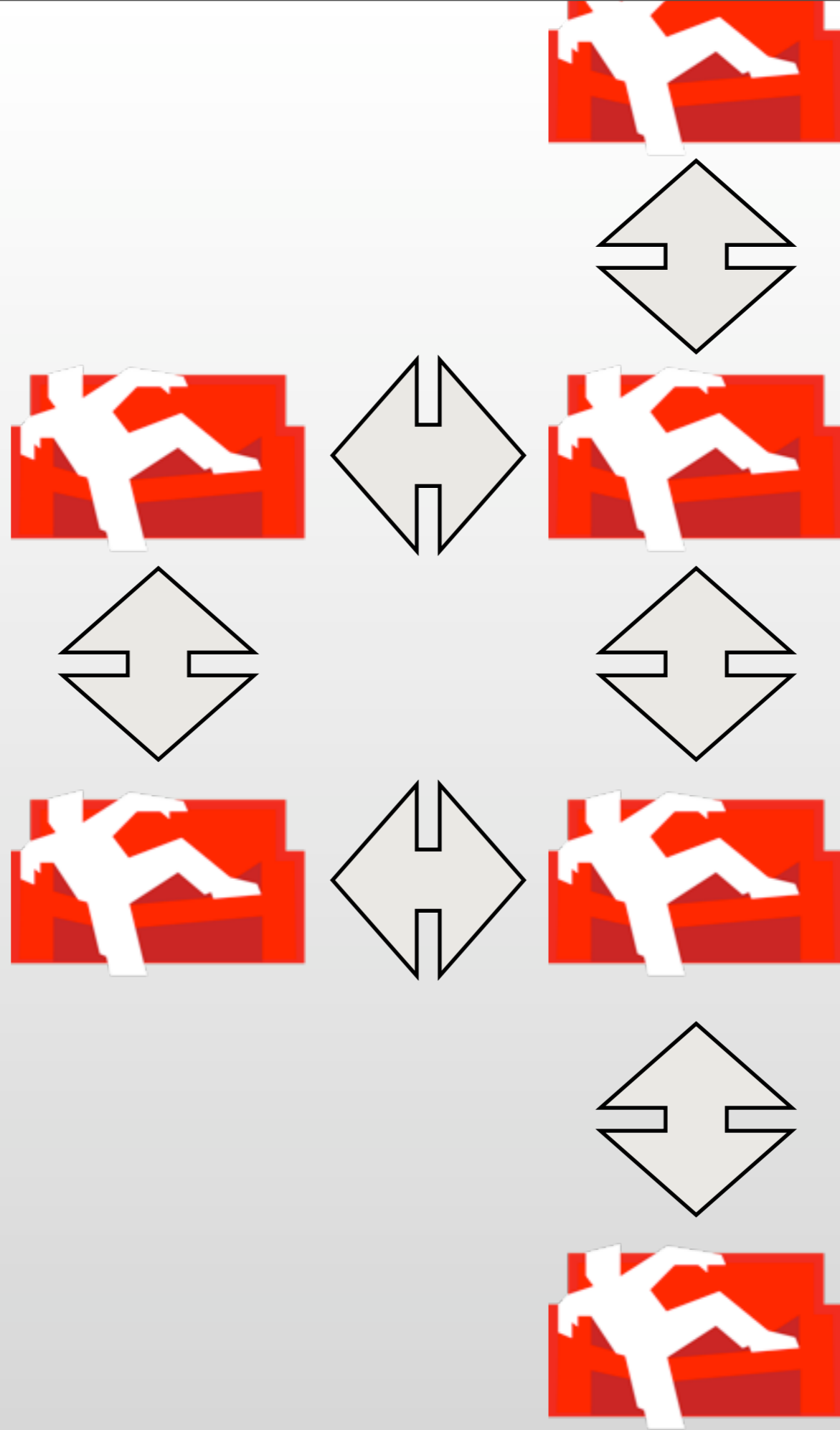






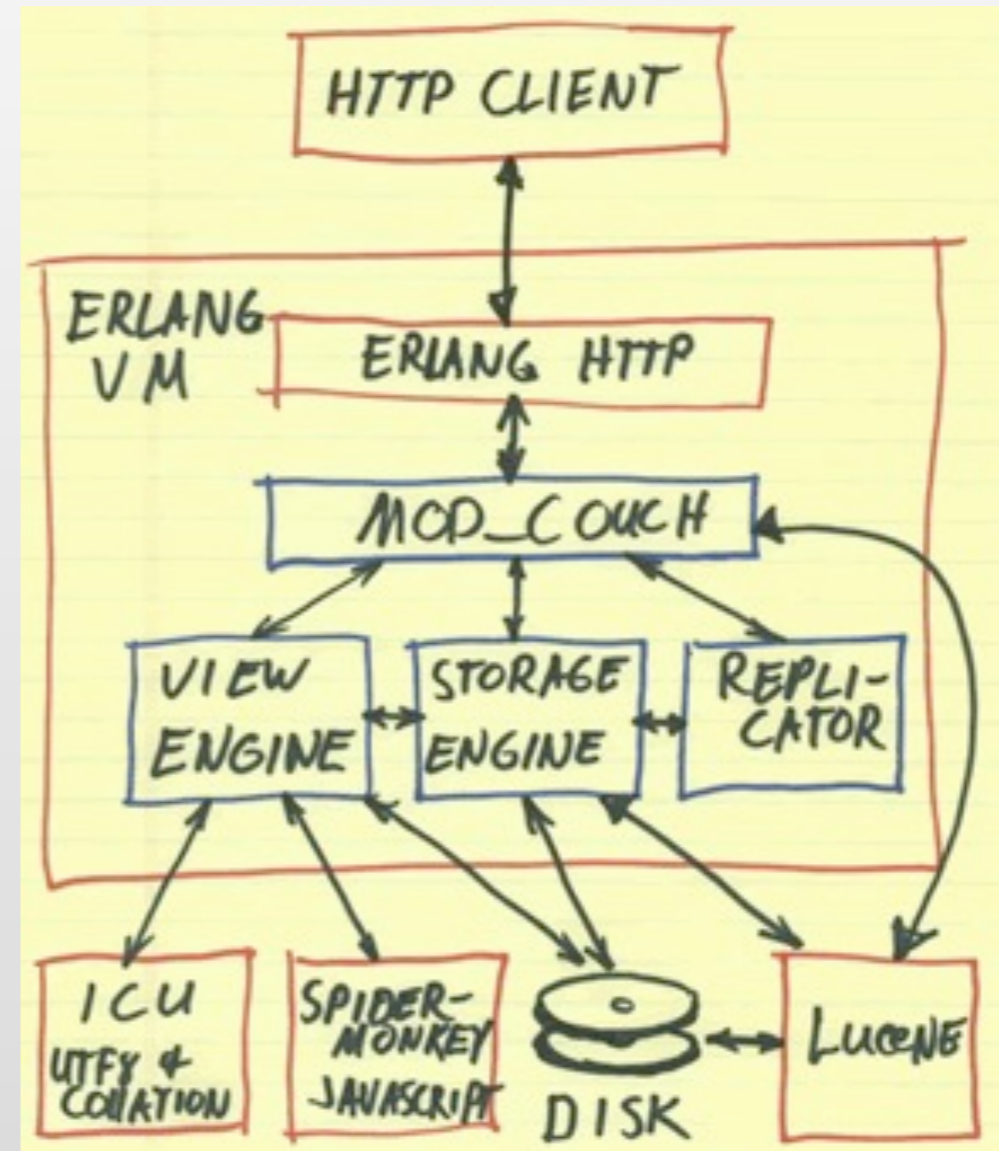






- Schema Free (JSON)
- Document Oriented,
Not Relational
- Highly Concurrent
- RESTful HTTP API
- JavaScript Powered
Map/Reduce
- N-Master Replication
- **Robust Storage**

Features



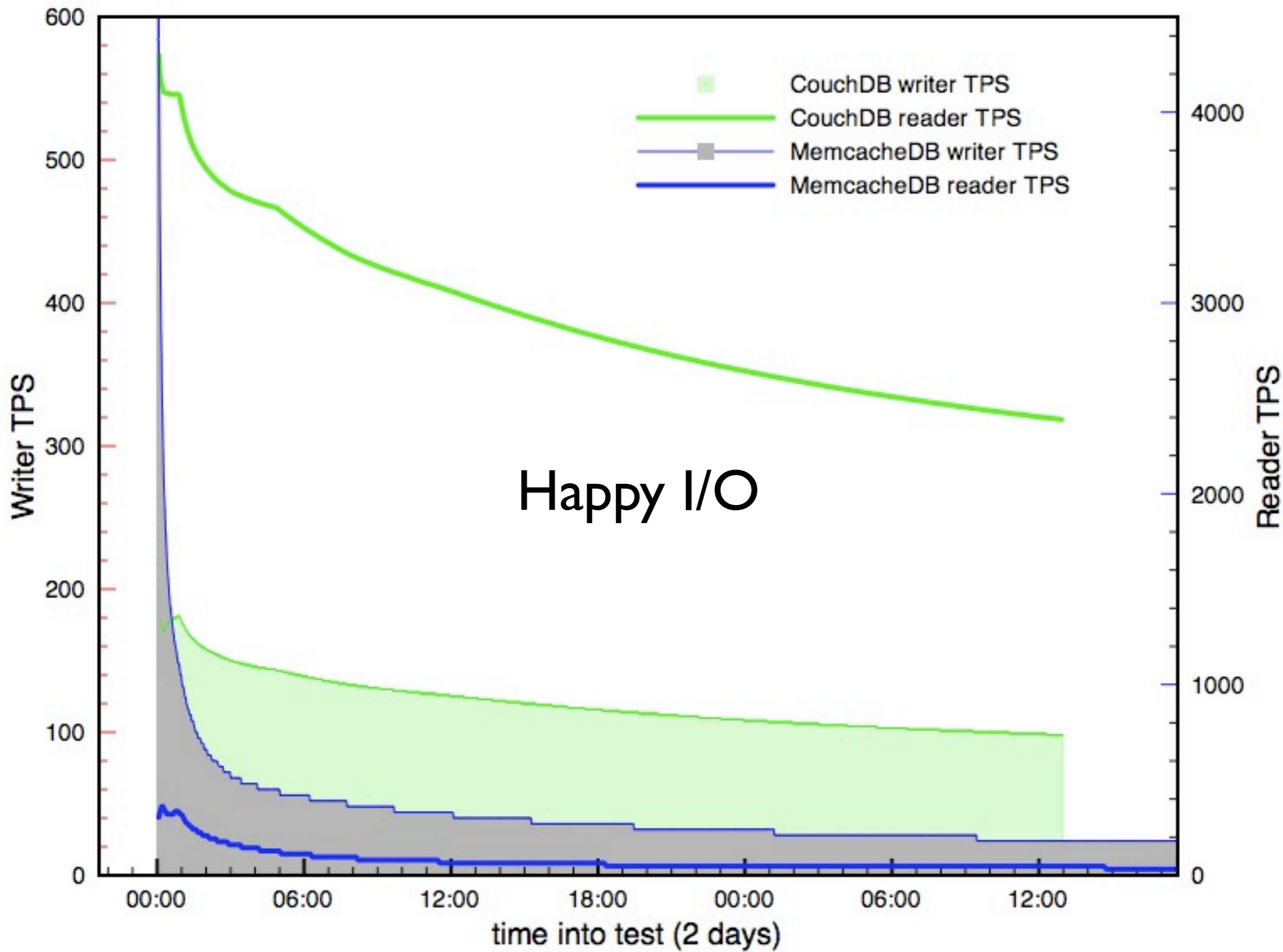
Robust Storage

Append Only
File Structure

Designed to
Crash

Instant-On





Resources

- Twitter: @CouchDB & <http://couchdb.org/>
- Dress like a Couch:
<http://shop.couchdb.com>
- <http://planet.couchdb.org/>
- <http://blog.racklabs.com/?p=74>
- <https://peepcode.com/products/couchdb-with-rails>

couch.io

Berlin – London – Portland

If you need professional CouchDB support talk to the dimwit on stage

Thanks!

