The Bleeding Edge

Richard Warburton and Martijn Verburg

http://www.jclarity.com

"Toto, I've a feeling we're not in Kansas anymore"

- Dorothy, Wizard of Oz

Who are we?

- **Richard** - Engineer, Author, Speaker
- **Martijn** - CEO, Author, Speaker, Cat Herder
- **jClarity** - Solves performance problems

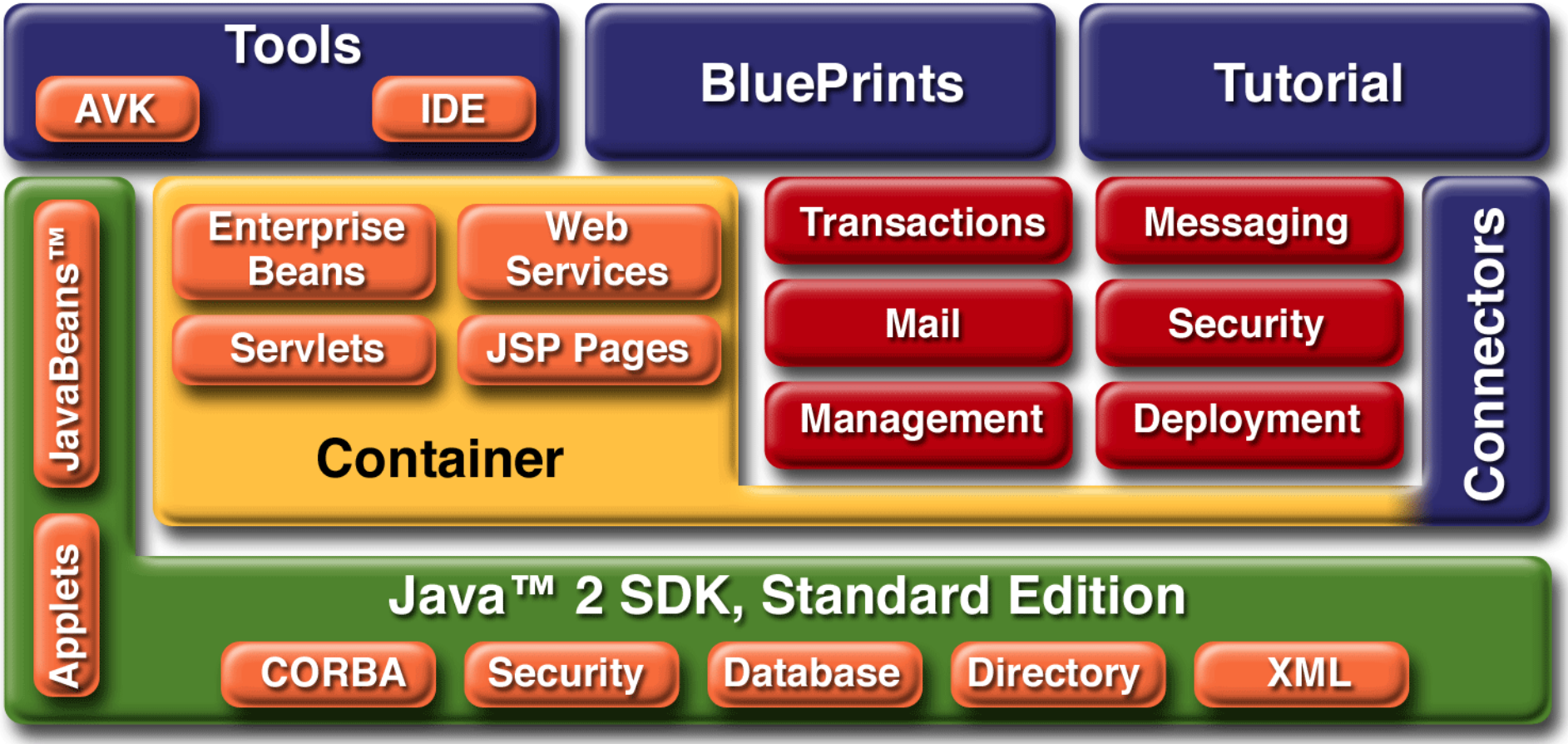
The Status Quo

What did jClarity build?

How we pick Technology

Technology Scorecard

A Brave New World


Tools
AVK
IDE
BluePrints
Tutorial
JavaBeans™
Applets
Enterprise Beans
Web Services
Servlets
JSP Pages
Container
Transactions
Messaging
Mail
Security
Management
Deployment
Connectors
Java™ 2 SDK, Standard Edition
CORBA
Security
Database
Directory
XML

We have a lot of Java EE Experience

- LJC is on the JCP Executive Committee
- Martijn is a **CERTIFIED** BEA Weblogic
  - Something, something, something **Darkside**
- Lots of experience and deep community

But what if you don’t use that?...

… we don’t!

The Status Quo

What did jClarity build?

How we picked Technology

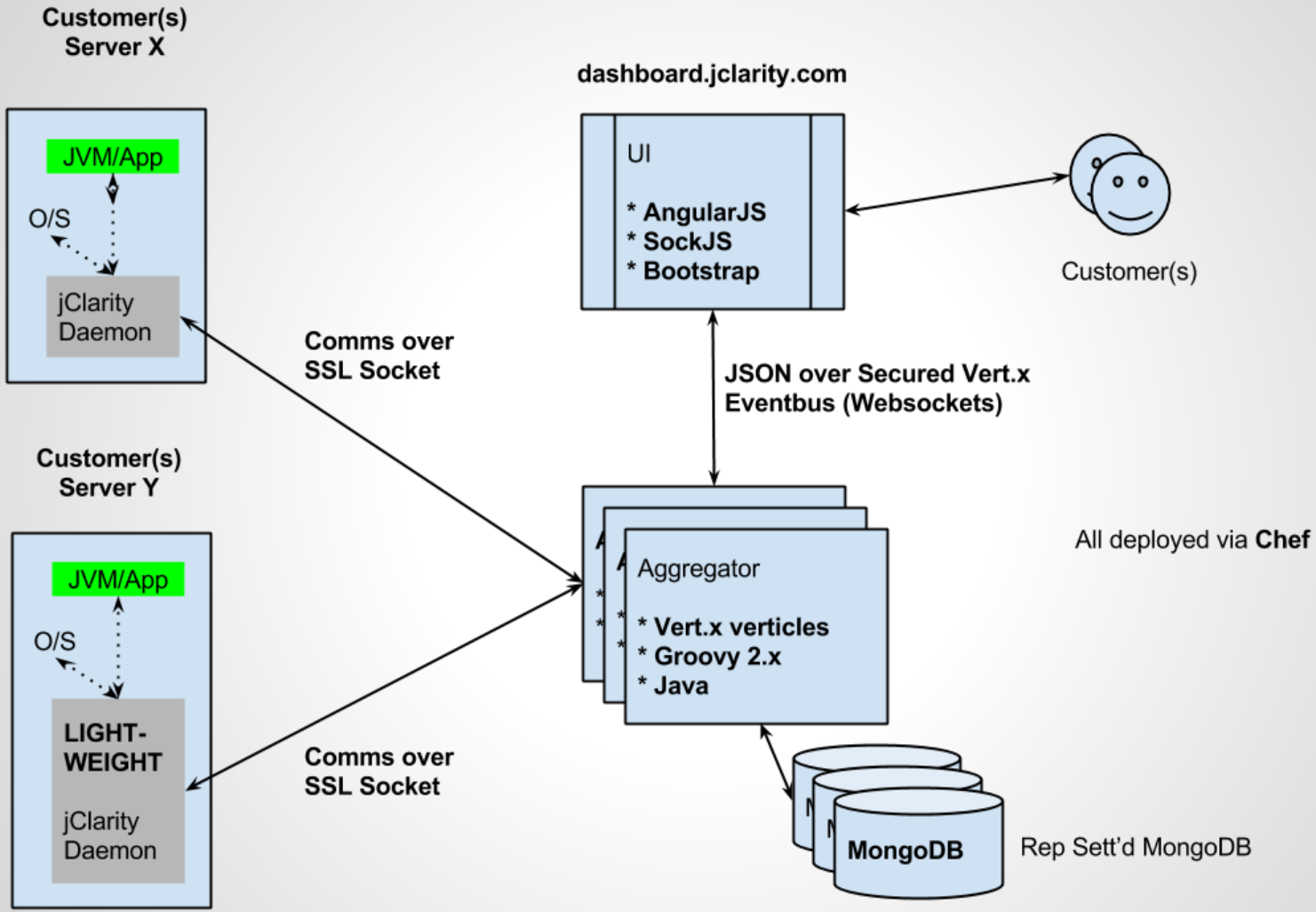
Technology Scorecard

A Brave New World

The Answer

..to your performance problems

Easy Access, Lightweight and Low Impact


Customer(s)
Server X
dashboard.jclarity.com
JVM/App
O/S
jClarity
Daemon
UI
* AngularJS
* SockJS
* Bootstrap
Customer(s)
Comms over
SSL Socket
JSON over Secured Vert.x
Eventbus (Websockets)
Customer(s)
Server Y
JVM/App
O/S
LIGHT-
WEIGHT
jClarity
Daemon
Aggregator
* Vert.x verticles
* Groovy 2.x
* Java
All deployed via Chef
Comms over
SSL Socket
MongoDB
Rep Sett'd MongoDB

Daemon, Aggregator & UI

- **Daemon**
  - Needs to be Low Impact, Low Traffic, Self-Updating
- **Aggregator**
  - Non-blocking I/O, Sockets > HTTP, High Availability
- **UI**
  - Rich Browser App, Accepts server-push events

Doesn’t really fit the Java EE World

- **Daemon:** Java EE is not low impact
- **Server:** Java EE is mainly about req/resp
- **UI:** Java EE lacks support for push events
  - Java EE 7 has Websockets now

The Status Quo

What did jClarity build?

How we picked Technology

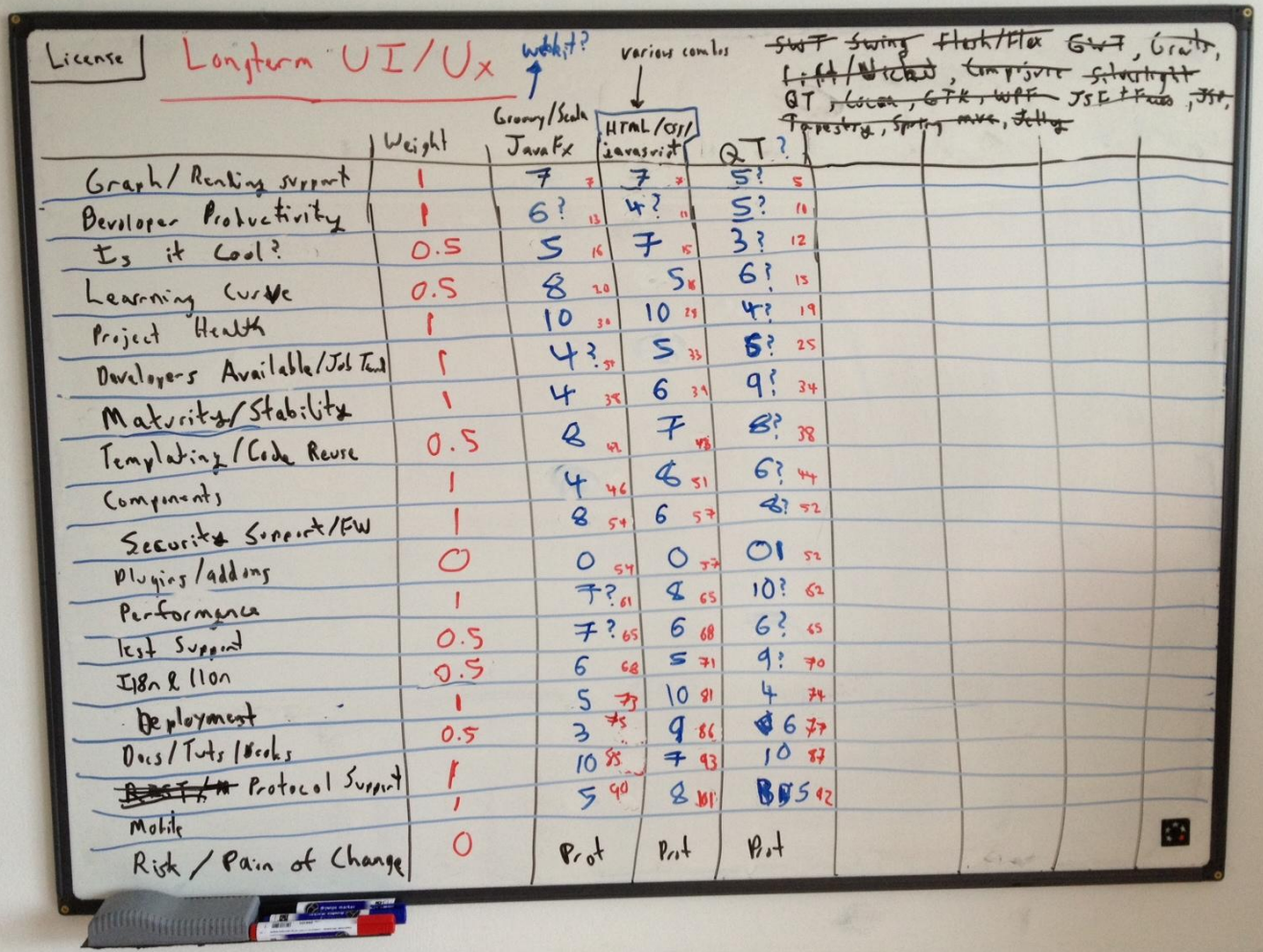
Technology Scorecard

A Brave New World

Our Methodology

~~Stolen~~ ‘Adopted’ from Matt Raible

1. Identify Ranking Criteria
2. Weight each criteria
3. Score each solution
4. Multiply and add
5. Prototype top two choices


License
Longterm UI/Ux
webkit?
various combos
~~SWT~~ ~~Swing~~ ~~Flash/Flex~~ ~~GWT~~, ~~Grails~~, ~~Lift/Wicket~~, ~~Compiere~~ ~~Silverlight~~ QT, ~~Cocoa~~, ~~GTK~~, ~~WPF~~ ~~JSF Faces~~, ~~JSP~~, ~~Tapestry~~, ~~Spring mvc~~, ~~Jetty~~
Weight
Groovy/Scala JavaFx
HTML/CSS/Javascript
QT?
Graph/Rending support 1 7 7 7 7 5? 5
Developer Productivity 1 6? 13 4? 11 5? 10
Is it Cool? 0.5 5 16 7 15 3? 12
Learning Curve 0.5 8 20 5 18 6? 13
Project Health 1 10 30 10 29 4? 19
Developers Available/Job Trend 1 4? 37 5 33 6? 25
Maturity/Stability 1 4 35 6 39 9? 34
Templating/Code Reuse 0.5 8 42 7 43 8? 38
Components 1 4 46 6 51 6? 44
Security Support/FW 1 8 54 6 57 8? 52
Plugins/addons 0 0 54 0 57 0 52
Performance 1 7? 61 8 65 10? 62
Test Support 0.5 7? 65 6 68 6? 65
I18n & l10n 0.5 6 68 5 71 9? 70
Deployment 1 5 73 10 81 4 74
Docs/Tuts/Books 0.5 3 75 9 86 6 77
~~REST/HTTP~~ Protocol Support 1 10 85 7 93 10 87
Mobile 1 5 90 8 101 5 92
Risk / Pain of Change 0 Prot Prot Prot

Observations

- **Cons**
  - Still Subjective
  - No guarantees
- **Pros**
  - Incorporates diverse feedback
  - Avoids bike-shedding
  - Avoids Technology Dictatorship

The Status Quo

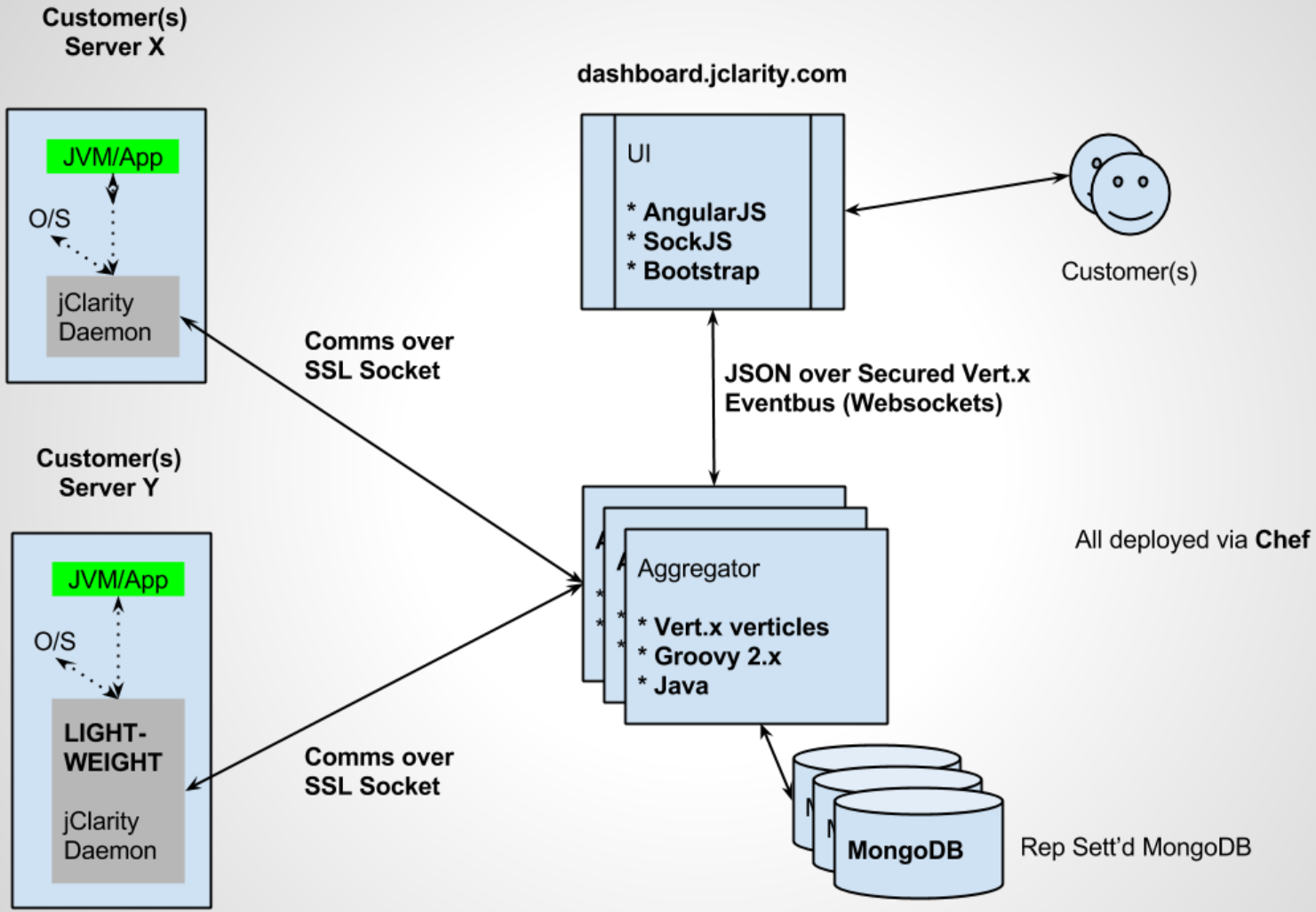
What did jClarity build?

How we picked Technology

Technology Scorecard

A Brave New World

What Tech Stack did we pick?


Customer(s)
Server X
dashboard.jclarity.com
JVM/App
O/S
jClarity
Daemon
UI
* AngularJS
* SockJS
* Bootstrap
Customer(s)
Comms over
SSL Socket
JSON over Secured Vert.x
Eventbus (Websockets)
Customer(s)
Server Y
JVM/App
O/S
LIGHT-
WEIGHT
jClarity
Daemon
Aggregator
* Vert.x verticles
* Groovy 2.x
* Java
All deployed via Chef
Comms over
SSL Socket
MongoDB
Rep Sett'd MongoDB

AngularJS

- Client-side Javascript framework
- Model-View-Whatever (MVC or MVVM)
- Two-Way Data Binding
- Directives used to encapsulate components

AngularJS - Positives

- Html5 concepts without adoption issues
- Declarative - easy to understand and read
- Simple code structure
- Client side templating

AngularJS - Negatives

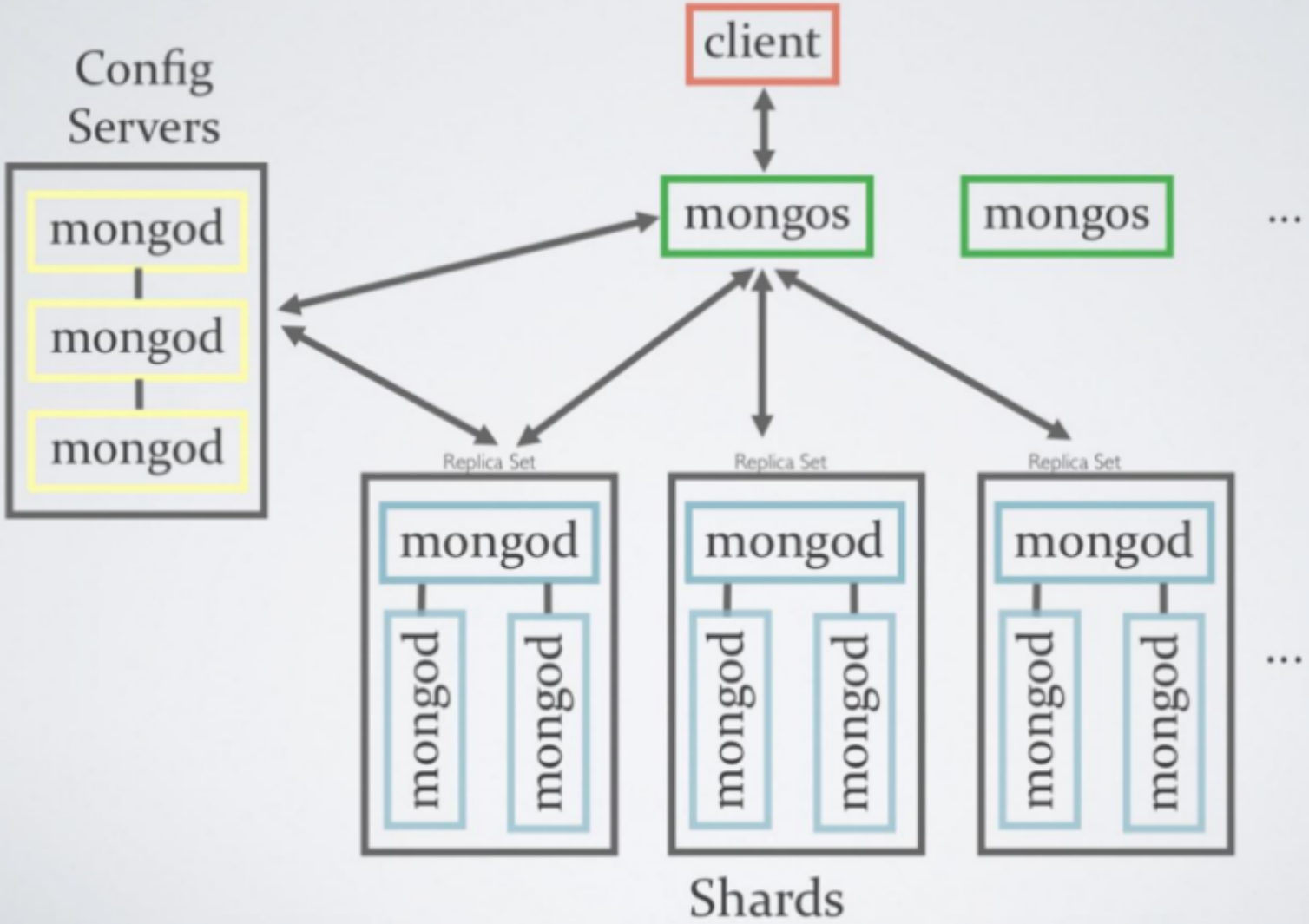
- Hard to integration test
- Use of `$scope.$apply`
- People are familiar with jQuery
  - Declarative learning curve.
- Javascript tooling still kinda sucks

AngularJS - Scorecard

"Real win over hand rolled Javascript"

MongoDB

- NoSQL Document Store
- “Scalable” - easy sharding
- “Fault Tolerant” - easy data replication
  - Saved our asses at least once….
- The new DB sexy!


client
Config Servers
mongod
mongod
mongod
mongos
mongos
...
Replica Set
Replica Set
Replica Set
mongod
mongod
mongod
mongod
mongod
mongod
mongod
mongod
mongod
...
Shards

MongoDB - Positives

- Document Modelling
- Nice API
- Easy to adopt and get started
- Easy Replication and Sharding
- Can be Fast*

MongoDB - Negatives (1)

- Security
  - we compiled an SSL-enabled mongo
  - pwning folks? Find out if they use Mongo!
  - tool support
- Data Integrity Concerns
- Split Brain - election algorithm failures

MongoDB - Negatives (2)

- Difficult to automatically sysadmin
- Immaturity of Drivers/Connectors (PHP)
- Document model is limited
- New query language to learn
- Poor timezone support

MongoDB - Scorecard

"Consider your security strategy well."

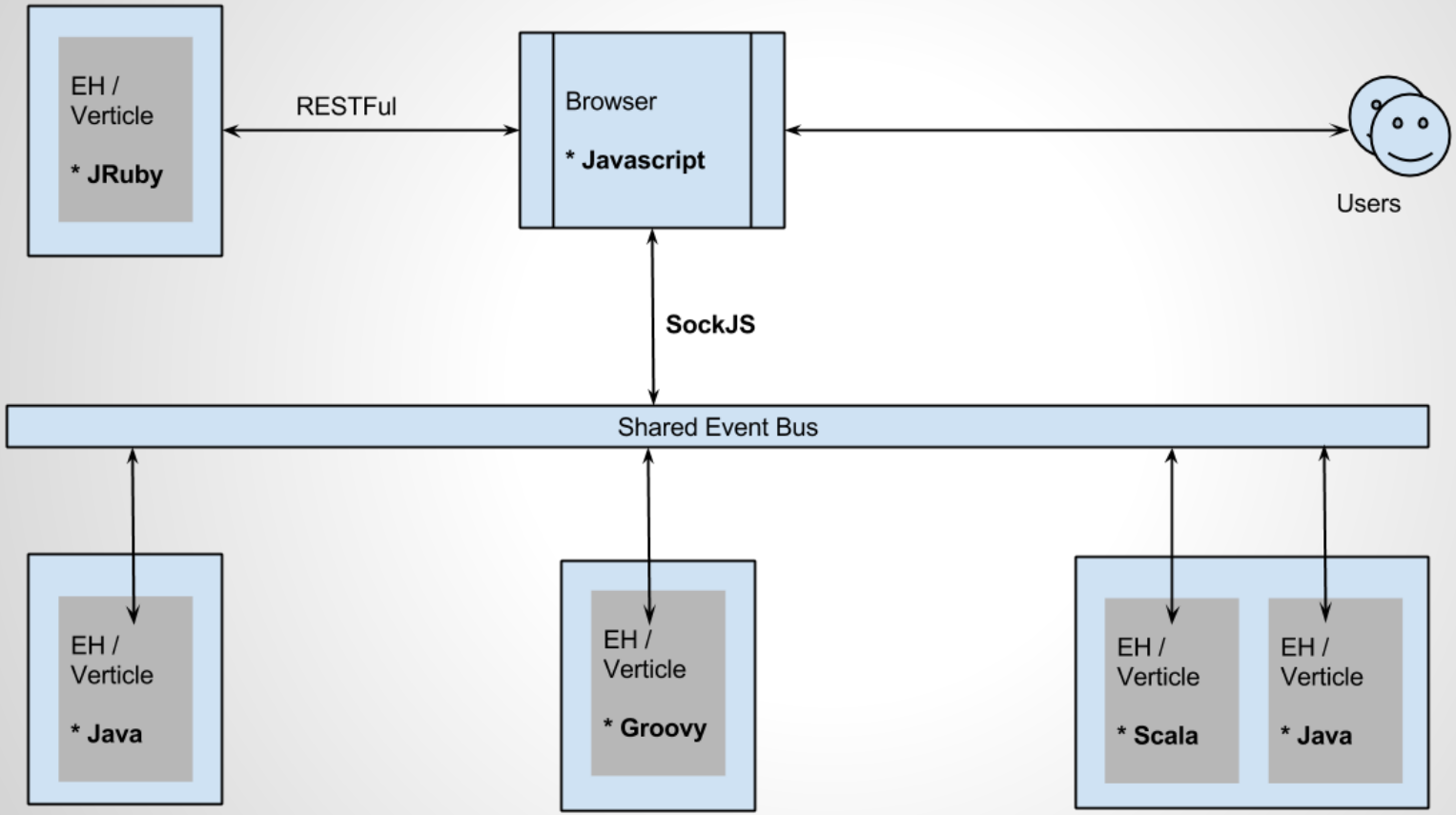
"Is a document store what you really need?"

Vert.x

- Polyglot Reactive Programming Framework
- Message Passing Oriented
  - Lightweight Eventbus
  - No Shared Mutable State
- Non-blocking I/O
- Websocket Support

Vert.x

- **Polyglot Reactive** Programming Framework
- **Message Passing** Oriented
  - Lightweight Eventbus
  - No Shared Mutable State
- **Non-blocking I/O**
- **Websocket** Support


EH / Verticle
* JRuby
RESTFul
Browser
* Javascript
Users
SockJS
Shared Event Bus
EH / Verticle
* Java
EH / Verticle
* Groovy
EH / Verticle
* Scala
EH / Verticle
* Java

Vert.x - Positives

- No shared state, easy concurrency
- Eventbus to the browser!
- Community++
- Polyglot
  - We used Groovy and Java

Vert.x - Negatives

- Immaturity:
  - Released 2.0 without supporting 1.3
  - Testing Framework not quite there
- NIH Logging
- Inter Machine Eventbus Security
  - Not originally designed for the open internet

Vert.x - Scorecard

"It's perfect…. for the RIGHT use case."

Chef

- Automates server configuration
  - Centralised repo of pre-built recipes
  - Ruby based DSL
- Can also automate application deployment
  - Run update to deploy config/binaries

Chef - Positives

- Lots of Java/JVM recipes
- Ability to set roles and override properties
- Centrally controlled configuration
- Provides structure and deployment model

Chef - Inconsistent

```
knife cookbook upload apache2
```

```
knife role from file roles/foo/bar.rb
```

```
knife data bag from file users john.json
```

Chef - Negatives

- Managing dependencies is difficult
  - Use Librarian to help
- Not Declarative
  - e.g: doesn’t remove old config
- Still very complex
  - Steep learning curve

Chef - Scorecard

"Chef, it's way better than shell scripts"

Common Bleeding Edge Themes

1. Immaturity
2. Lack of tooling
3. Problems at the boundary

The Status Quo

What did jClarity build?

How we picked Technology

Technology Scorecard

A Brave New World

Technology ←→ $$$$$$$

What are you paying for?

- What does your support contract get you?
- Who does it get you?
- What’s the lock-in?

A La Carte Open Source

- Popular with $0 budget startups
- Commercially funded tech
- Strong Community Support
- Benefits elsewhere

The 'A La Carte' Relationship

- Be part of the community
  - We participated in Vertx IRC/London meetings
- Don’t diverge - upstream Patches
  - Vert.x, mod-mongo, Jacoco
- Open Source your plumbing
  - https://github.com/johnoliver/release-version-plugin

How do I do this?

- Low risk project
  - 20% time?
- Ask for forgiveness, not permission!
- Don’t fear “them”

Q&A

@richardwarburto
@karianna
@jclarity