Sharing Code Between Node.js Applications

Node.js in the wild
September 19, 2012

Daniel Rinehart and Tim Walling
Connected Sports Ventures

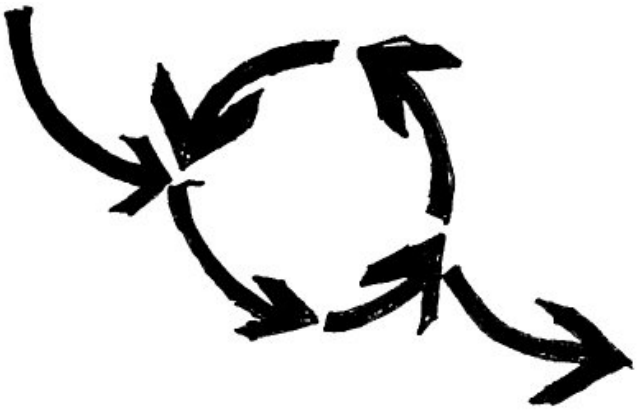
RumbleTV Baseball

iPad application available at http://rumbletv.com/

RumbleTV Football

iPad application available at http://rumbletv.com/

Evolution



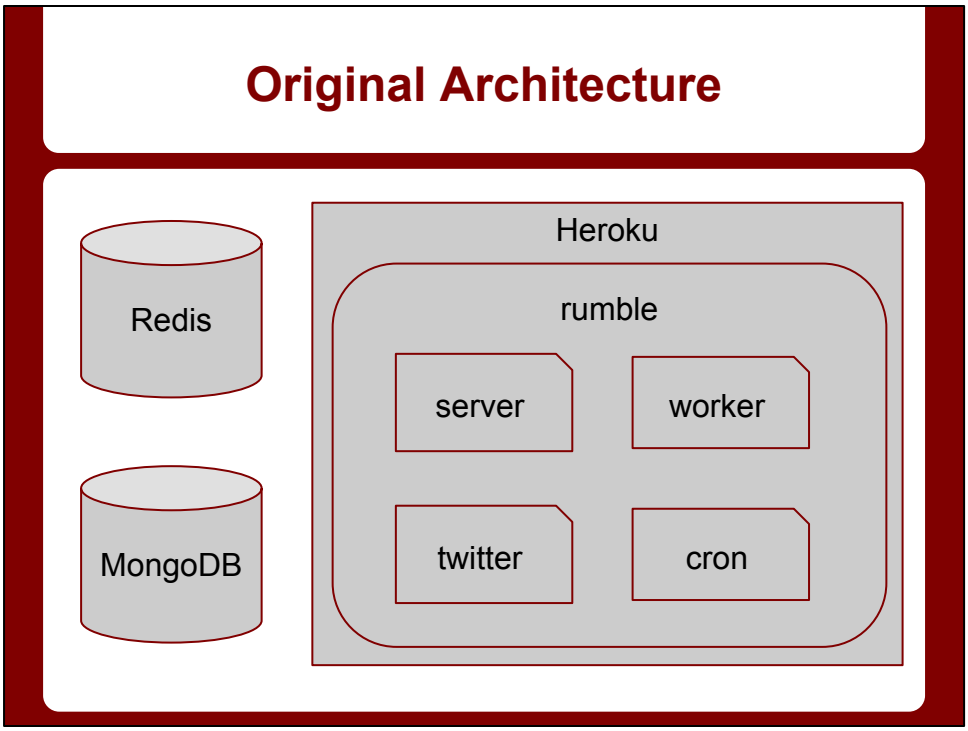
Source: http://agileconsulting.blogspot.com/2009/01/agile-over-rup-my-preferred-development.html

Continual process of discovery
Looked to solve pain points
Still experimenting
Focus on go-live

Original Code Structure

/rumble

Simple, all code in one location
Quick developer setup: git clone, npm install, npm start (or use foreman)
Experimented with how to best organize the code as the code base grew

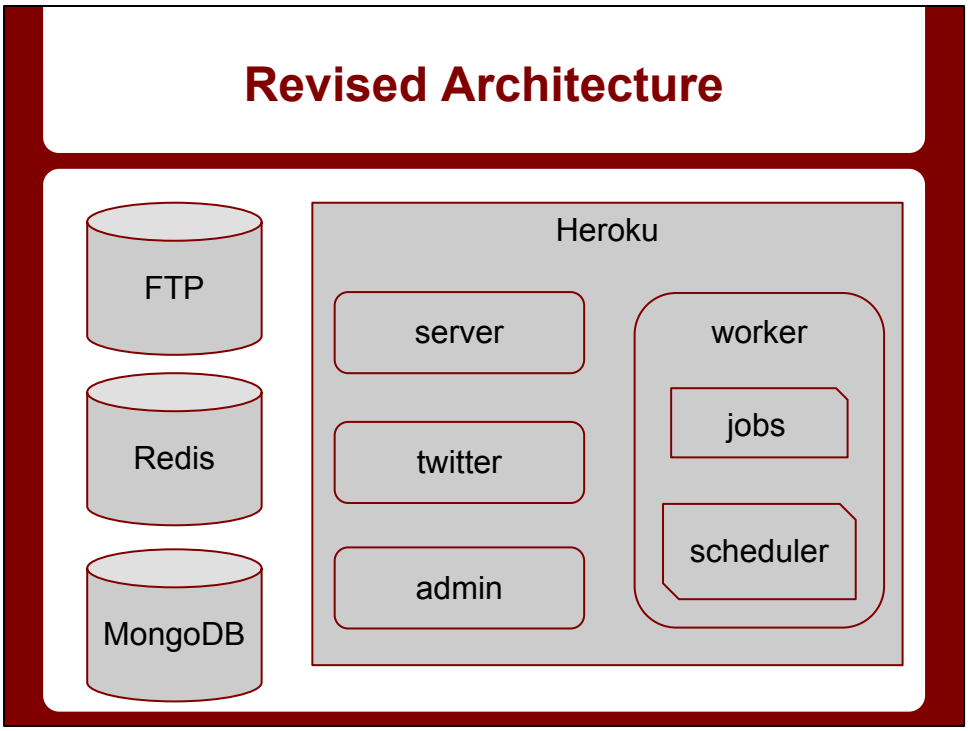


Procfile specified different processes to run
Code push updated/restarted everything
Polyglot persistence based on different needs (system of record versus cached data)

Revised Code Structure

/rumble-test
/rumble-common
/rumble-server
/rumble-workers
/rumble-twitter
/rumble-admin

Shared code (test and common) at root of dependency tree which downstream projects require
Easier to focus on only the code and tests for a specific part of the system
Difficult developer setup (created script to help)
Easy to not be using latest code (see npm link later)



Can restart only application that needs to be updated
Better use of Heroku's free tier for alpha environment since it is multiple applications
Custom build pack needed if not versioning dependencies
Heroku requires empty commit to force redeploy if not versioning dependencies
Inter project dependencies can be problematic if synchronized push is needed

package.json dependencies

```
{
  "dependencies": {
    "proj": "https://github.com/PATH/tarball/BRANCH"
  }
}
```

USERNAME:PASSWORD@github.com
(possible security concerns)

With a package.json can specify URL to download
Can use github username/password to access private repos, not the best approach
Would like to move to continuous deployment that packages modules in code pushed out (Heroku anvil project)
Could use private npm repo but that seemed too much work

npm link

```
cd ~/trends-common
npm link

cd ~/trends-server
npm link trends-common
```

For developing locally can link to latest code

package.json scripts

```
{
  "scripts": {
    "start": "node src/server.js",
    "test": "jasmine-node --junitreport test",
    "coverage": "node tools/coverage.js"
  }
}
```

Capture common commands and scripts you want to run
Even things not part of standard npm life-cycle

package.json versions

```
{
  "dependencies": {
    "cli": "0.4.3",
    "minimatch": "0.0.x"
  },
  "devDependencies": {
    "nodeunit": ">0.0.0"
  }
}
```

Be aware of how you want to manage versions of packages
Static or semantic

tools for managing versions

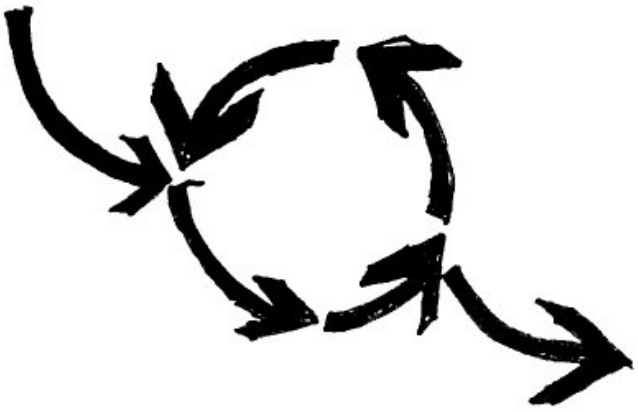
npm outdated

npm shrinkwrap

npm install -g police
police -lf package.json

Use outdated for tracking when a semantic version is out of date including child modules
Use shrinkwrap for capturing all semantic versions and making them static including child modules
Use police to check for newer versions of static modules

Retrospective



Still learning
Security concerns
Versioning
Empty commits
Custom buildpack

Example

Exposing current trends on Twitter and Instagram:

- https://github.com/twalling/trends-common
- https://github.com/twalling/trends-server
- https://github.com/twalling/trends-workers