



# **Two-fisted tales of DevOps - beards vs. process.**

John Hawkes-Reed  
<http://www.futureplc.com>  
Future Publishing Ltd

# The Plan

- Who on earth is this numpty?
- What is this DevOps business?
- How and why we are automating our infrastructure.
  - A little about our infrastructure
  - Some of the things we're trying out
- A few things we learned the hard way

# JHR

- Unix curmudgeon, revolutionary Peelist and SF writer.
- Performed in the field circus, started an ISP, participated in Dotcom v1.0, did stuff for HP Labs.
- Does stuff for Future.



# Some Context

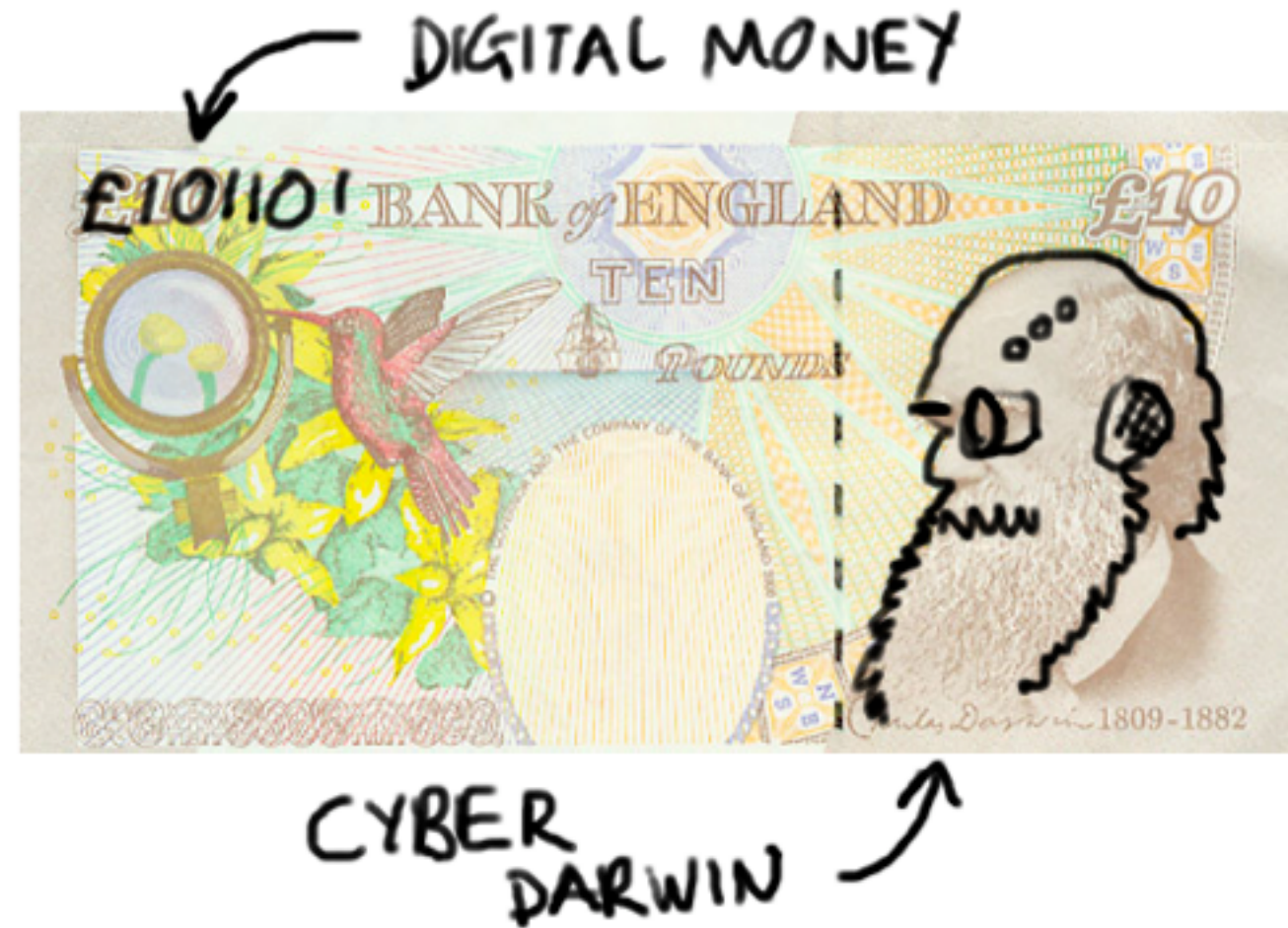
## Future Publishing Ltd.

- Print and digital publisher
- Niche interest and hobbies
- ~80 magazines
- ~34 million unique visitors
- Expanding in to digital as that is where the money is hiding

# Some Context

## Future Publishing Ltd.

- Print and digital publisher
- Niche interest and hobbies
- ~80 magazines
- ~34 million unique visitors
- Expanding in to digital as that is where the money is hiding



# Some More Context:

## Hysterical Raisins

- Solaris and OSX Tiger
- Website code deploys via SVN
- Packages installed by hand



- Webservers, DB and supporting infrastructure
  - Co-lo
    - 48 Debian/Xen hosts
    - 185 Debian virtuals
  - Rackspace UK
    - 36 virtuals
- Corporate stuff
  - In house - here there be dragons.



- Webservers, DB and supporting infrastructure
  - Co-lo
    - 48 Debian/Xen hosts
    - 185 Debian virtuals
  - Rackspace UK
    - 36 virtuals
- Corporate stuff
  - In house - here there be dragons.





- ~350 websites in total
- ~300 servers all inclusive
- 2 Puppetmasters
- 7 meshed ActiveMQ instances

- ~350 websites in total
- ~300 servers all inclusive
- 2 Puppetmasters
- 7 meshed ActiveMQ instances

A whole lot of deploys (approx 250 a month) - it is someone's full time job.

# DevOps?

- From interesting idea to marketing bottomspeak in record time.
- Everyone else's theory - no more silos. However, where I'm from silos are for storing fodder and will kill you if you get it wrong.
- JHR's theory, which is mine. And RAW's - The philosophy of raised expectations. Which is to say that we've all met or been Mordac, preventer of IT. Don't do that it is rubbish.
- This is hard in a broken environment.

# Some remarks about cake.

- How to move away from a blame culture.
- If you make a cock-up\* - buy the victims cake.
- Avoid speculation in cake futures or pastry-based derivatives.
- Management don't always get it. Win-win!

\* - This probably doesn't scale unless you're rebuilding a LEO installation.

# What We Did

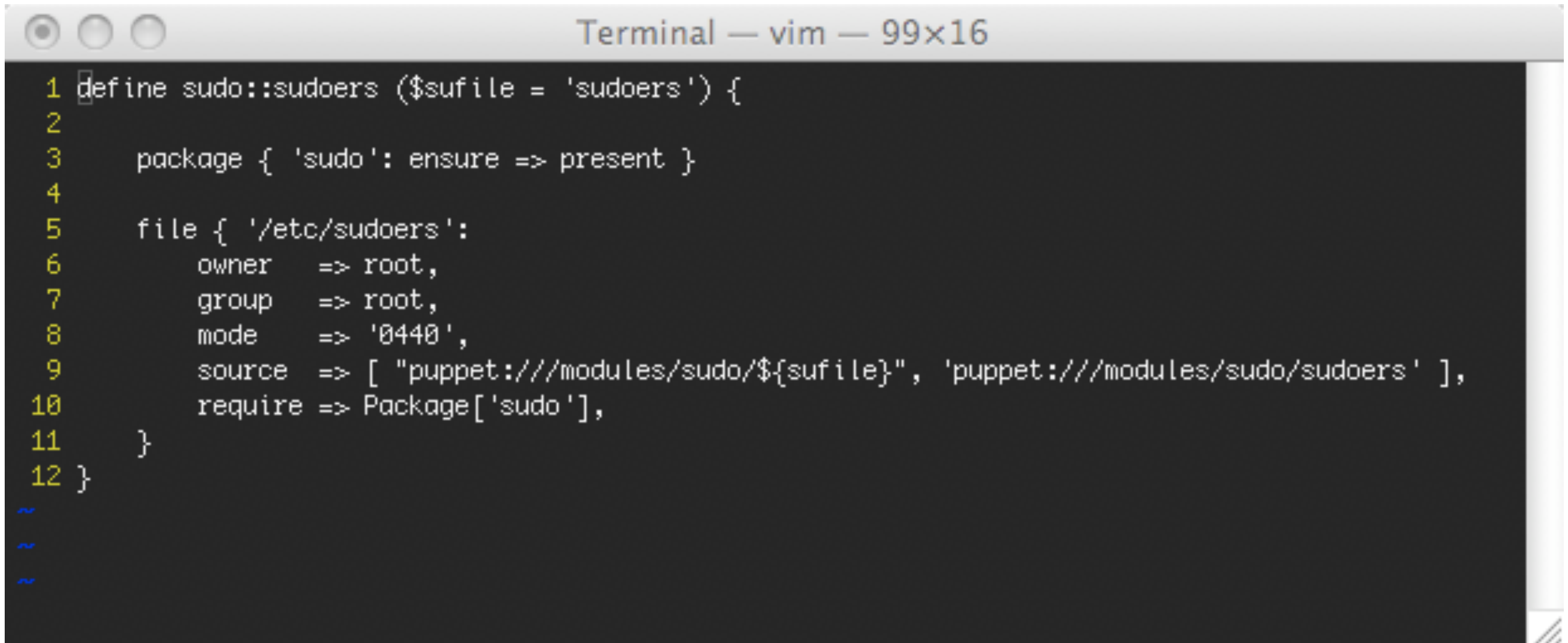
- Sidled towards Debian
- Decided to use puppet
- Started using MCollective
  - ActiveMQ
  - Metadata





- Best off looking at the Puppetlabs site.
- Descriptive rather than strictly procedural.
- There's Chef, too.
- Why we don't use Fabric.

# An example:

A screenshot of a terminal window titled "Terminal — vim — 99x16". The terminal displays a Puppet manifest for the 'sudo' module. The manifest is written in a dark-themed editor with line numbers on the left. The code defines a 'sudo::sudoers' class that takes '\$sufile' as an argument. Inside the class, there is a 'package' block to ensure 'sudo' is installed, and a 'file' block for '/etc/sudoers'. The file block sets the owner and group to 'root', the mode to '0440', and the source to a list of two Puppet module paths. It also requires the 'Package' resource for 'sudo'. The terminal shows three tilde characters at the bottom, indicating the end of the file.

```
1 define sudo::sudoers ($sufile = 'sudoers') {
2
3     package { 'sudo': ensure => present }
4
5     file { '/etc/sudoers':
6         owner    => root,
7         group    => root,
8         mode     => '0440',
9         source   => [ "puppet:///modules/sudo/${sufile}", 'puppet:///modules/sudo/sudoers' ],
10        require => Package['sudo'],
11    }
12 }
```

~

~

~



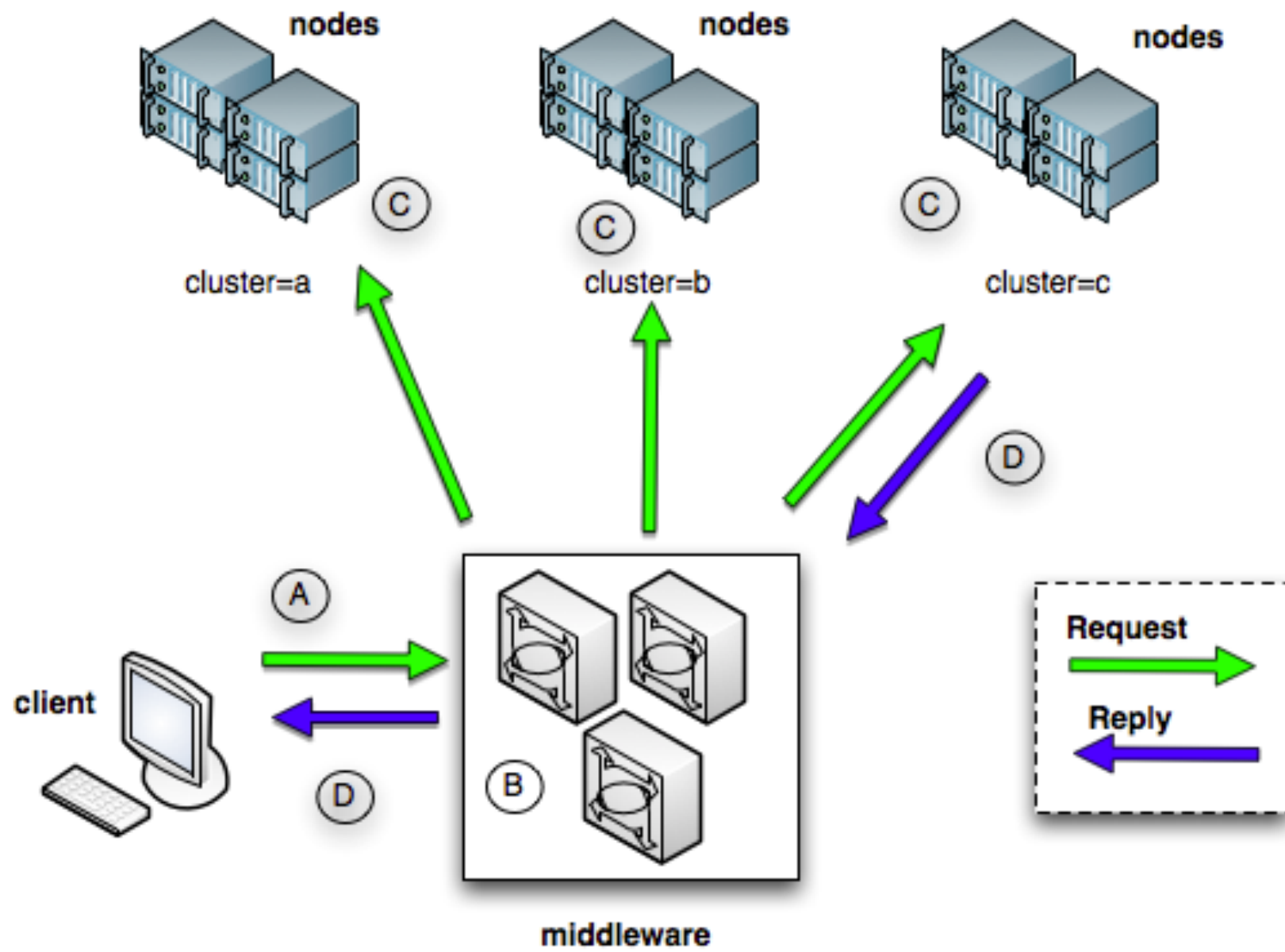
# Marionette Collective:

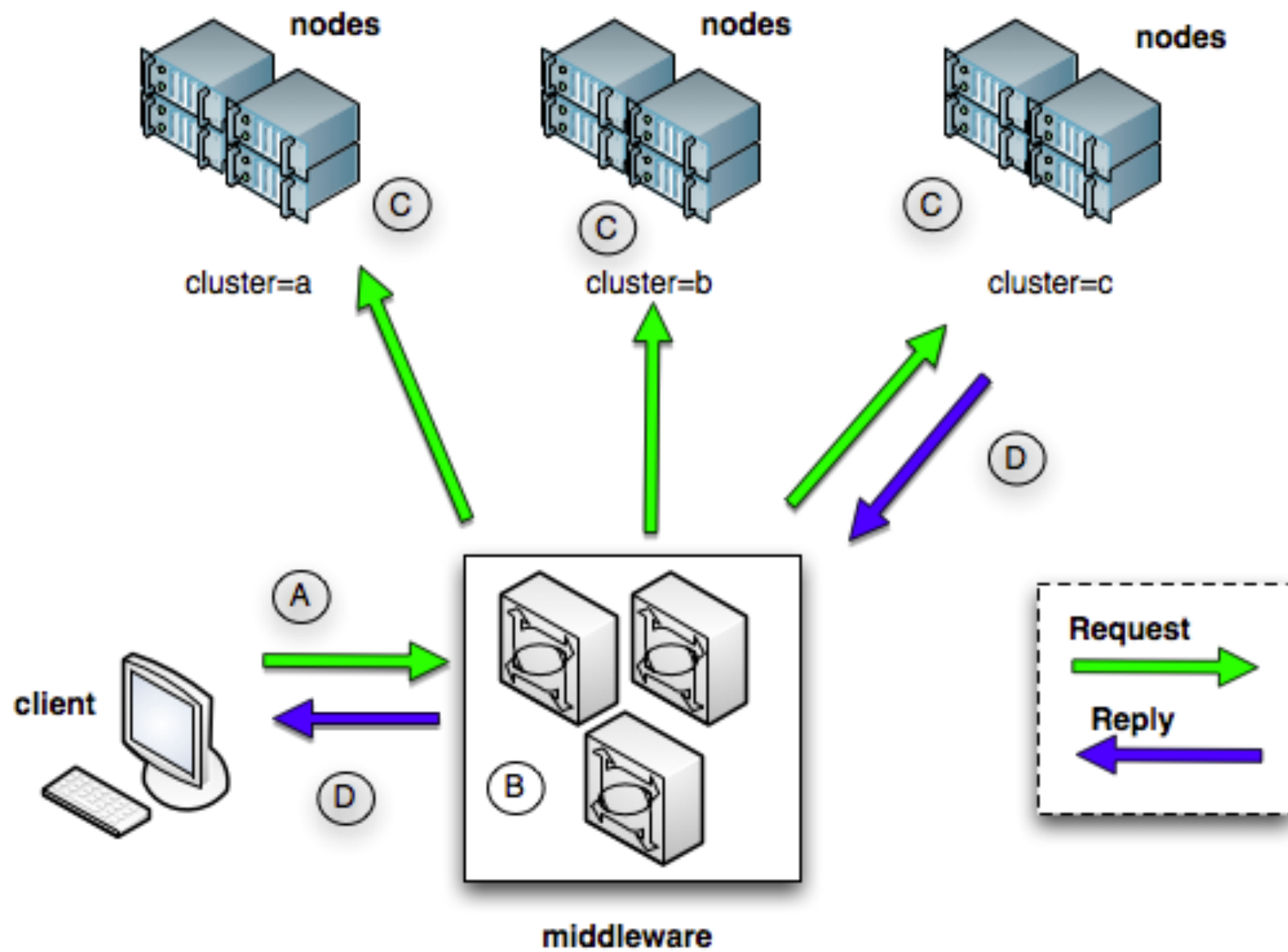
## An Introduction

- “The Marionette Collective AKA mcollective is a framework to build server orchestration or parallel job execution systems.” - <http://www.marionette-collective.org>
- “Concurrent execution of tasks and gathering of metadata using publish/subscribe middleware.” - Matt

# Marionette Collective: Terminology

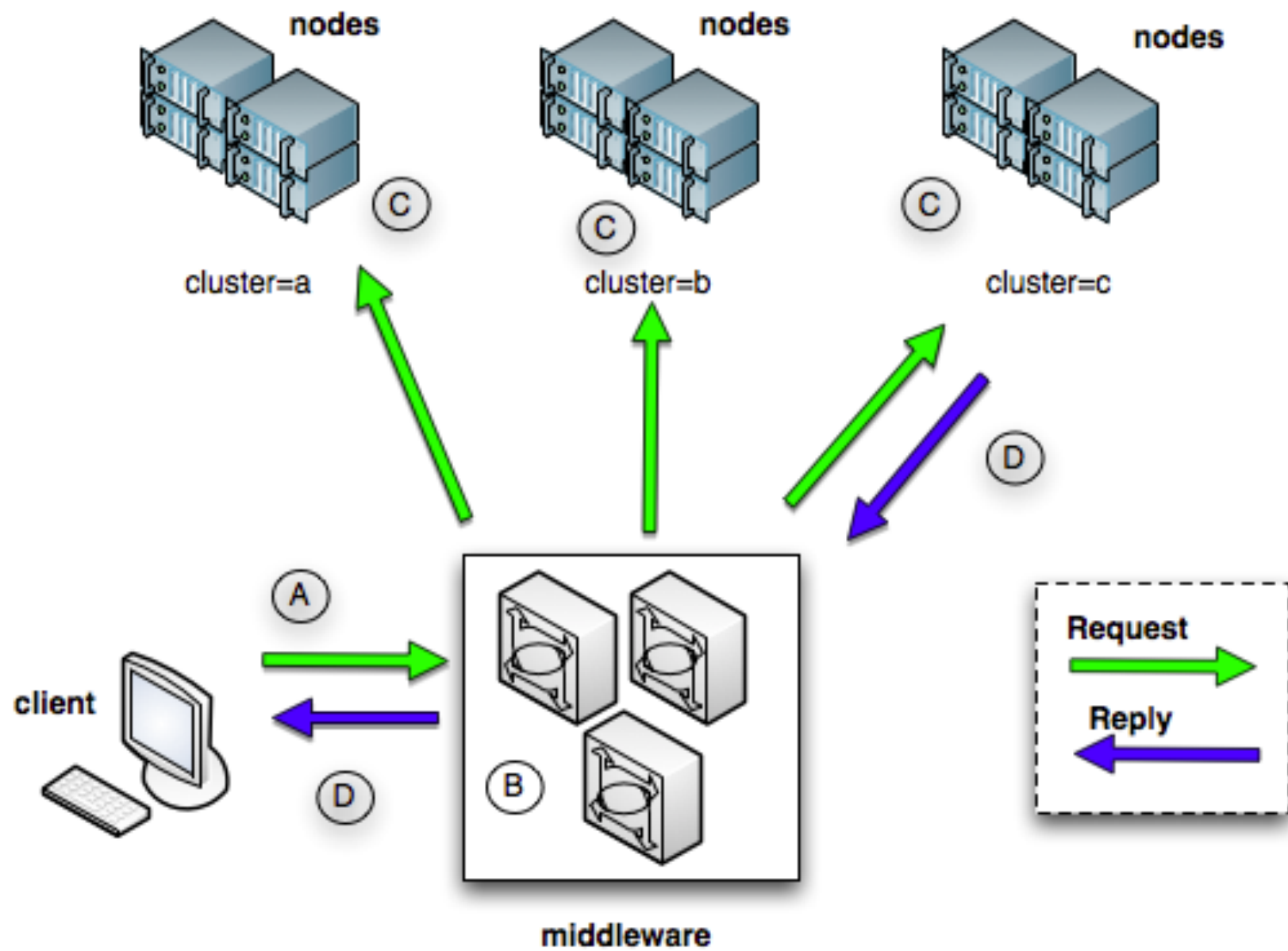
MCollective commands are issued from the **client** and the responses are from the **servers**.





```
rick@astley:~$ mco ping
never                               time=60.38 ms
gonna                               time=61.14 ms
give                                time=61.49 ms
you                                 time=61.84 ms
up                                  time=62.19 ms
...time=156.36 ms

---- ping statistics ----
147 replies max: 156.36 min: 60.38 avg: 90.32
```

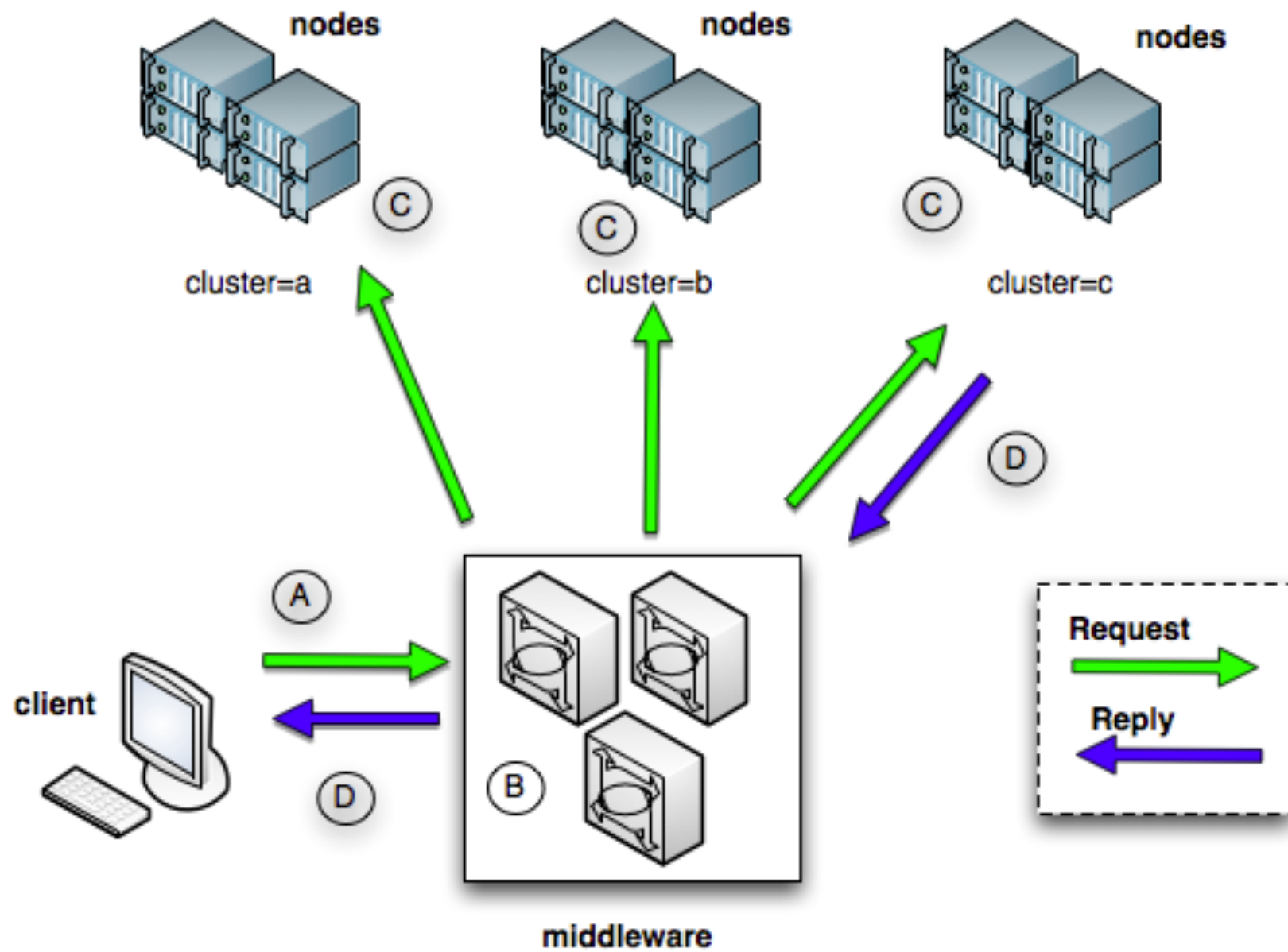


```
rick@astley:~$ mco ping
never                               time=60.38 ms
gonna                               time=61.14 ms
give                                time=61.49 ms
you                                 time=61.84 ms
up                                  time=62.19 ms
...time=156.36 ms

---- ping statistics ----
147 replies max: 156.36 min: 60.38 avg: 90.32
```

```
rick@astley:~$ mco ping -I /ever/
never                               time=60.42 ms

---- ping statistics ----
1 replies max: 60.42 min: 60.42 avg: 60.42
```



```
rick@astley:~$ mco ping
never                               time=60.38 ms
gonna                               time=61.14 ms
give                                time=61.49 ms
you                                 time=61.84 ms
up                                  time=62.19 ms
...time=156.36 ms

---- ping statistics ----
147 replies max: 156.36 min: 60.38 avg: 90.32
```

```
rick@astley:~$ mco ping -I /ever/
never                               time=60.42 ms

---- ping statistics ----
1 replies max: 60.42 min: 60.42 avg: 60.42
```

```
rick@astley:~$ mco facts timezone -C nginx
Report for fact: timezone

GMT                                found 19 times
UTC                                found 21 times

Finished processing 40 / 40 hosts in 242.00 ms
```

# Magical Things That We Found

- MCollective registration
- Message queuing
- Deploys via MCollective
- Passing the buck

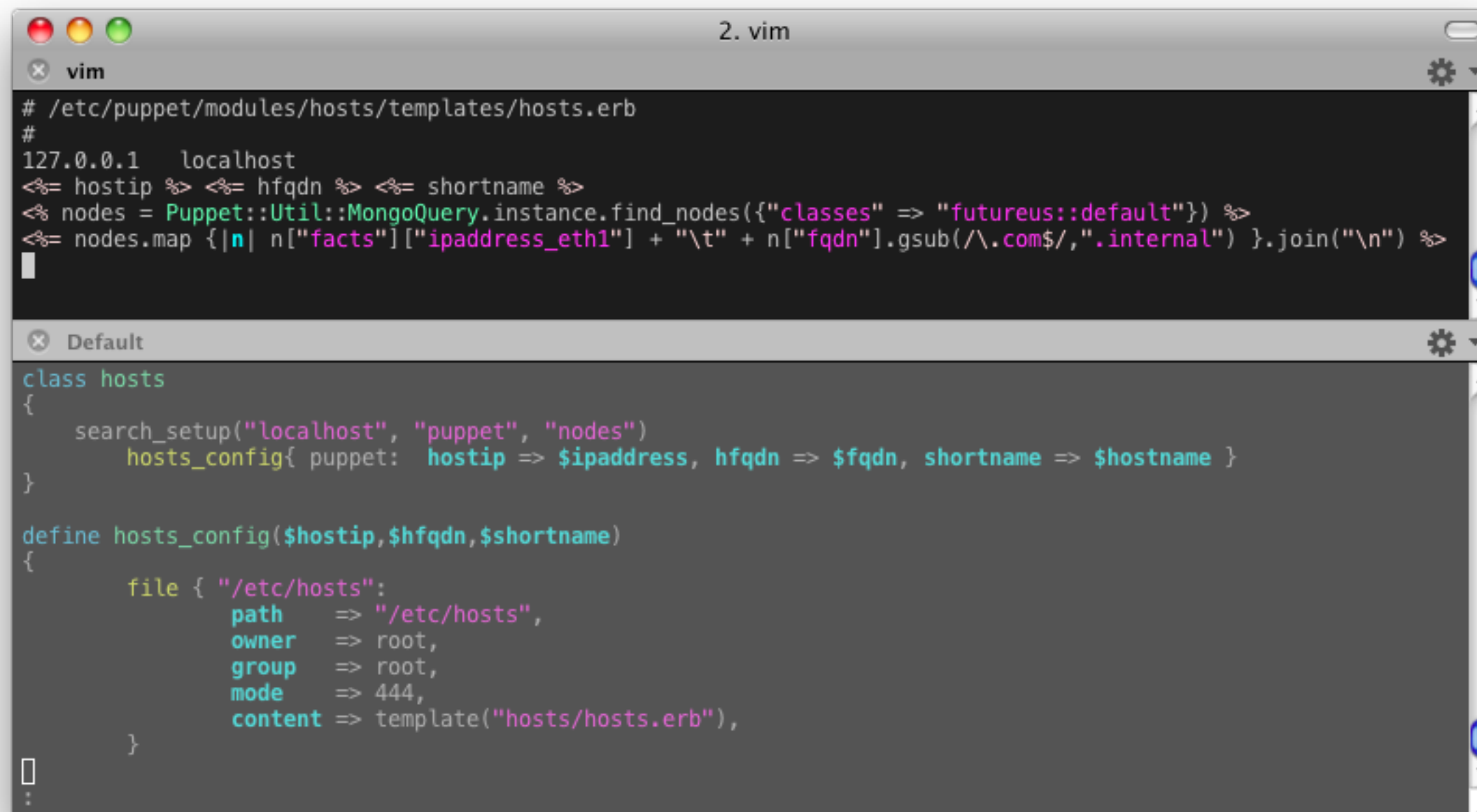




# MCollective Registration

- Meta-data via Facter, yaml, ohai or anything you choose to write yourself
- Servers phone home every 300 seconds
- Any server can collect registration data to pluggable backends (text files, MongoDB etc)

# Example: Configuring /etc/hosts using MCollective and MongoDB



The image shows a vim editor window with two tabs. The top tab, titled '2. vim', contains the content of the file `/etc/puppet/modules/hosts/templates/hosts.erb`. The bottom tab, titled 'Default', contains the Puppet class definition for `hosts`.

```
# /etc/puppet/modules/hosts/templates/hosts.erb
#
127.0.0.1    localhost
<%= hostip %> <%= hfqdn %> <%= shortname %>
<%= nodes = Puppet::Util::MongoQuery.instance.find_nodes({"classes" => "futureus::default"}) %>
<%= nodes.map {|n| n["facts"]["ipaddress_eth1"] + "\t" + n["fqdn"].gsub(/\.com$/, ".internal") }.join("\n") %>

class hosts
{
  search_setup("localhost", "puppet", "nodes")
  hosts_config{ puppet: hostip => $ipaddress, hfqdn => $fqdn, shortname => $hostname }
}

define hosts_config($hostip,$hfqdn,$shortname)
{
  file { ["/etc/hosts":
    path      => "/etc/hosts",
    owner     => root,
    group     => root,
    mode      => 444,
    content   => template("hosts/hosts.erb"),
  ]
}
```

# Deploys: Old School Style

- Subversion
  - Rapidly changing code base
  - Ease of transition from stage to live
  - Packaging unnecessary and impractical
  - Built in version control, stays in version control
  - Still used for historical reasons
  - Slow and a little bit shonky
- We are still doing this and it sucks

# Deploys via MCollective: Subversion

You should totally try this.

# Deploys via MCollective: Subversion

You should totally try this.



# Events via the MQ

- When something does something that someone might care about, publish to /topics/events.subject
- Write any degree of consumers
  - IRC
  - Twitter
  - Metrics gathering
  - Message to on call engineer
  - Automated emails to HR asking them to post P45s to anyone who breaks something

# Events via the MQ

- When something does something that someone might care about, publish to /topics/events.subject
- Write any degree of consumers
  - IRC
  - Twitter
  - Metrics gathering
  - Message to on call engineer
  - Automated emails to HR asking them to post P45s to anyone who breaks something

```
09:33 < eventbot> gitolite: [puppet-environments] refs/heads/master  
29daa2abb6509ffa07c0b989bf5da2aa0a08deb7  
09:33 < eventbot> jenkins: Requested Jenkins build:  
[puppet-environments] refs/heads/master  
29daa2abb6509ffa07c0b989bf5da2aa0a08deb7  
09:33 < eventbot> stomp-git: batlpuppet01 fetched change:  
[puppet-environments] refs/heads/master  
29daa2abb6509ffa07c0b989bf5da2aa0a08deb7  
09:33 < eventbot> stomp-git: PM: Wrote branch master into  
/etc/puppet/environments/master  
09:33 < eventbot> jenkins: Begin job 340 - [puppet-environments]  
origin/HEAD 29daa2abb6509ffa07c0b989bf5da2aa0a08deb7  
09:36 < eventbot> jenkins: Complete job 340 - [puppet-environments]  
origin/HEAD 29daa2abb6509ffa07c0b989bf5da2aa0a08deb7
```



# Deploys via MCollective: Git

- Uses a totally different approach that plays off of the strengths of git
- Sites need to be defined in CM (Puppet) to work at all
- Tag switches are fast, as are rollbacks
- Leads to a certain amount of idiotproofing



# git-configure

```
git-configure::sshkey{ 'spog-ssh': rootdir => '/data', owner =>
'www-data', homedir => '/var/www', }
```

- Create ssh-key 'spog-ssh' to go with site 'spog-git'
- Create for user www-data with appropriate permissions and set permissions for rootdir as well

```
git-configure::site{ 'spog-git': site => 'www.spog.com', repo =>
'spog', }
```

- Clone repo 'site' from git server
- Clone to /data/repo/www.spog.com
- Create a working tree in /data/www.spog.com
- Also places relevant set of facts in facts.yaml
- Installs stomp listener for git as a daemon

# Keeping Up With Commits (i)

- Post-receive hook on origin to send out message via STOMP over the MQ
- Any subscribed listeners will do a git fetch on the repo in the appropriate / data/repo/ directory
- Nothing changes in the working tree (i.e. where the site is being served from)
- Whenever a tag change is to occur, all the heavy lifting has already been done.

# Keeping Up With Commits (ii)

- Because there's a message-bus there, emitting events, you can do other stuff too.
- Jenkins, for instance.
- Jenkins hooks into commit messages. Emits success or fail on different topic.
- Repo-listener hooks jenkins-success topic, so never fetches a broken build. Probably.

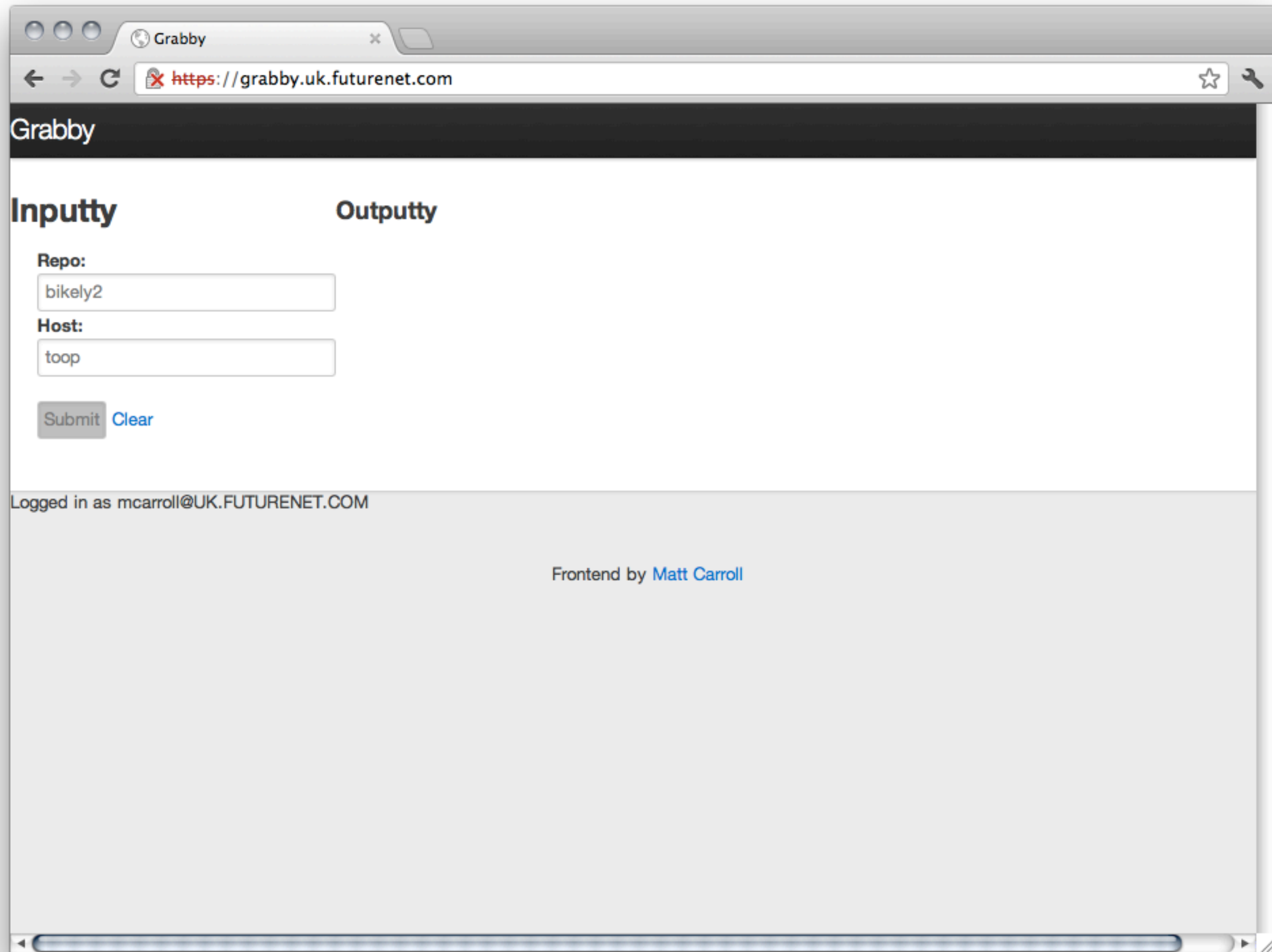
# Deploying

- `f-taglist repo -l host`
  - Gives a list of available branches and tags
- `f-checkout repo tag -l host`
  - Checks out tag for specified repo on host

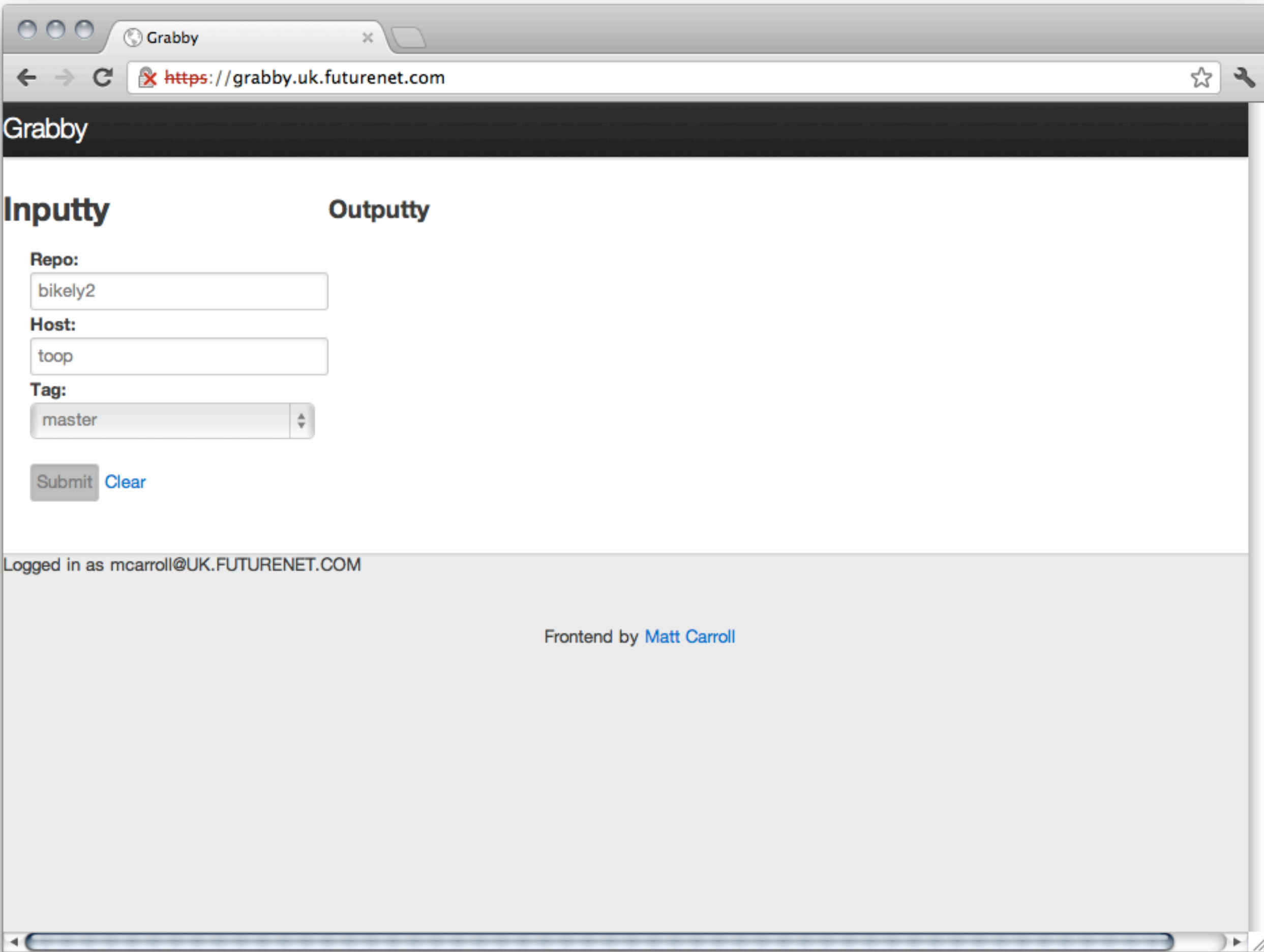
Both of these use the metadata stored in MCollective's `facts.yaml` to perform tasks on the appropriate directories automatically

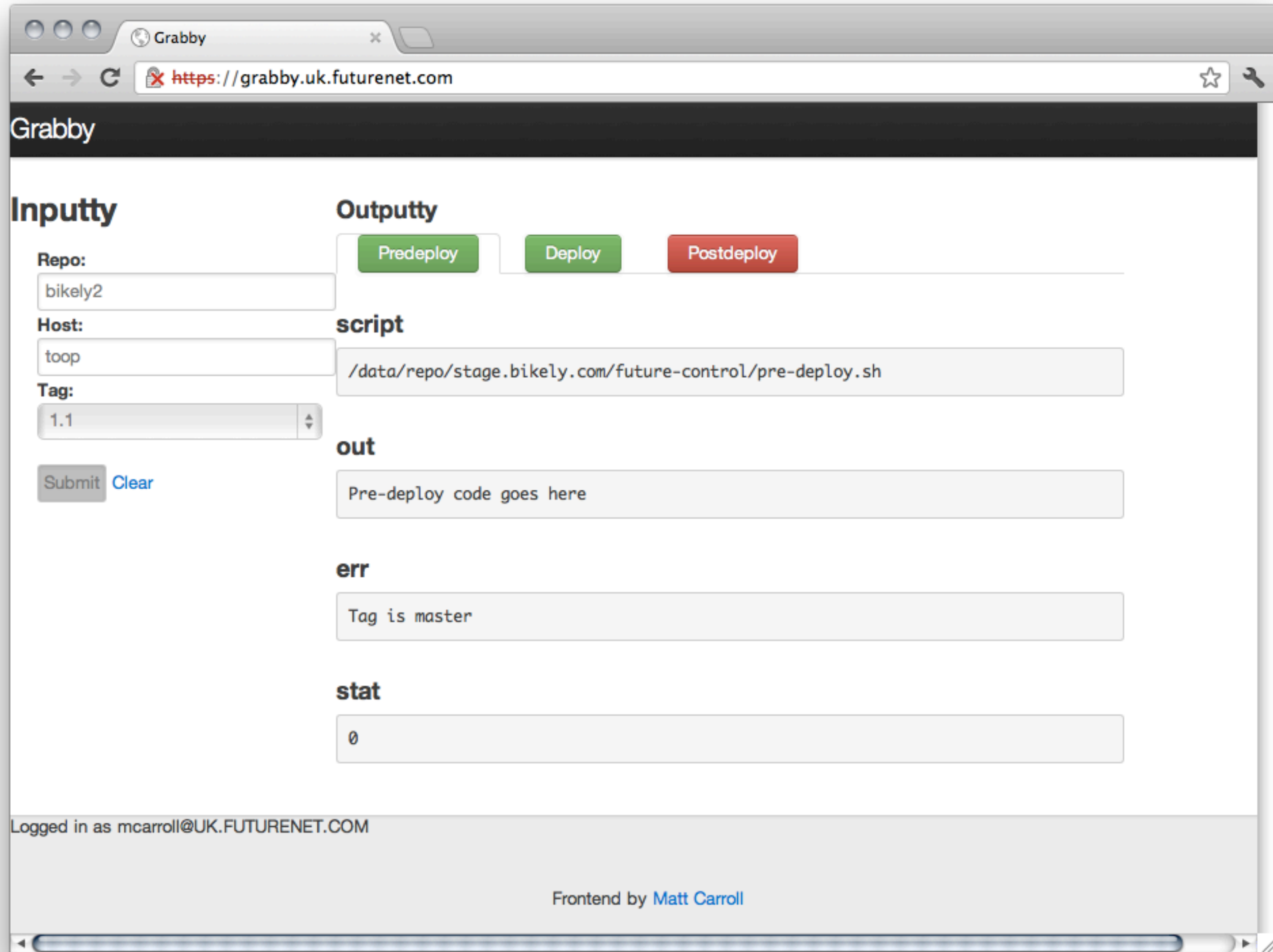
# Git Deploy Frontend

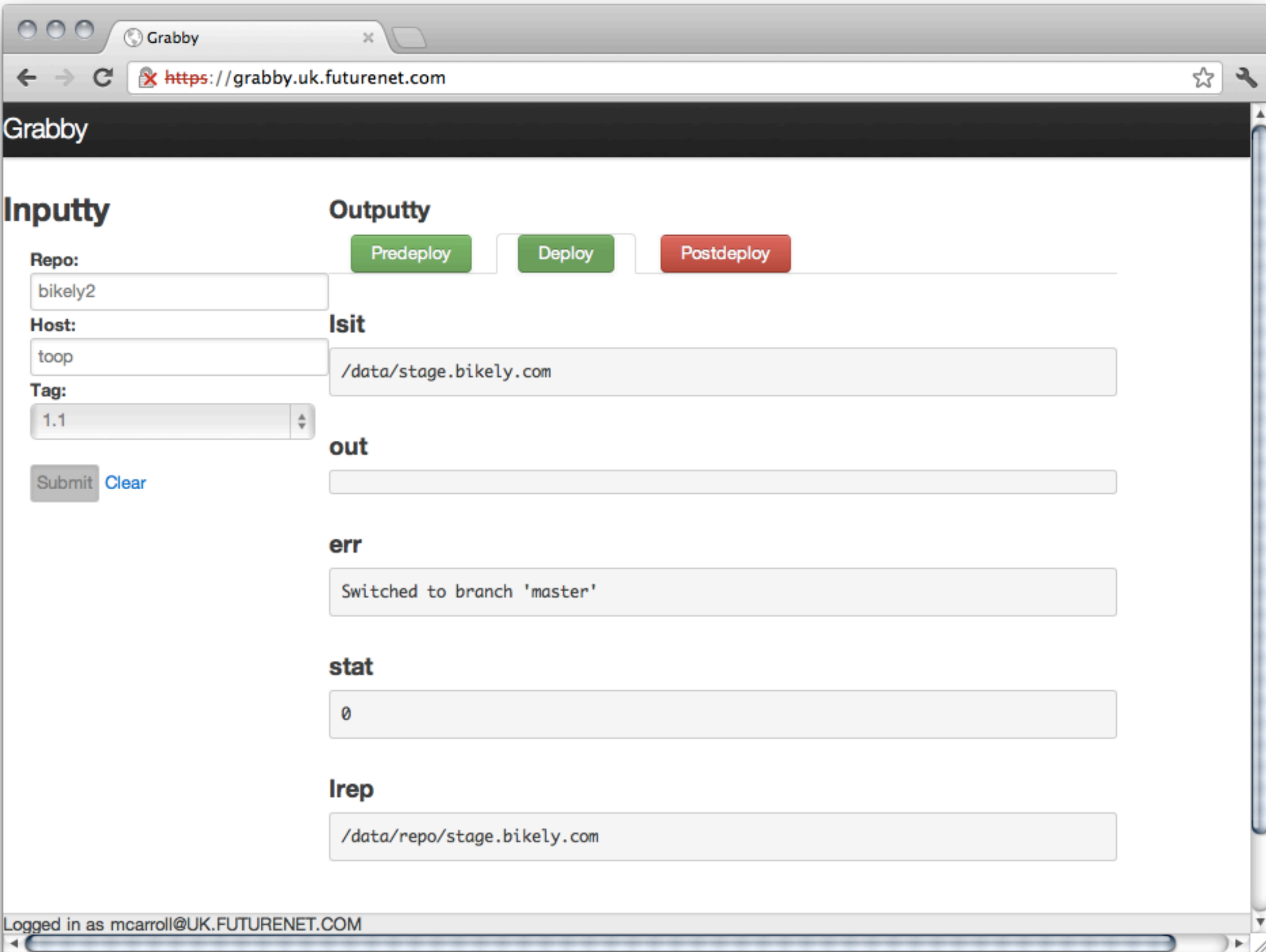
- Sinatra + MCollective = Web based client
- Use a series of drop downs with authentication to allow the right developers to deploy whichever tag they want to the right place
- Aka. Passing the buck

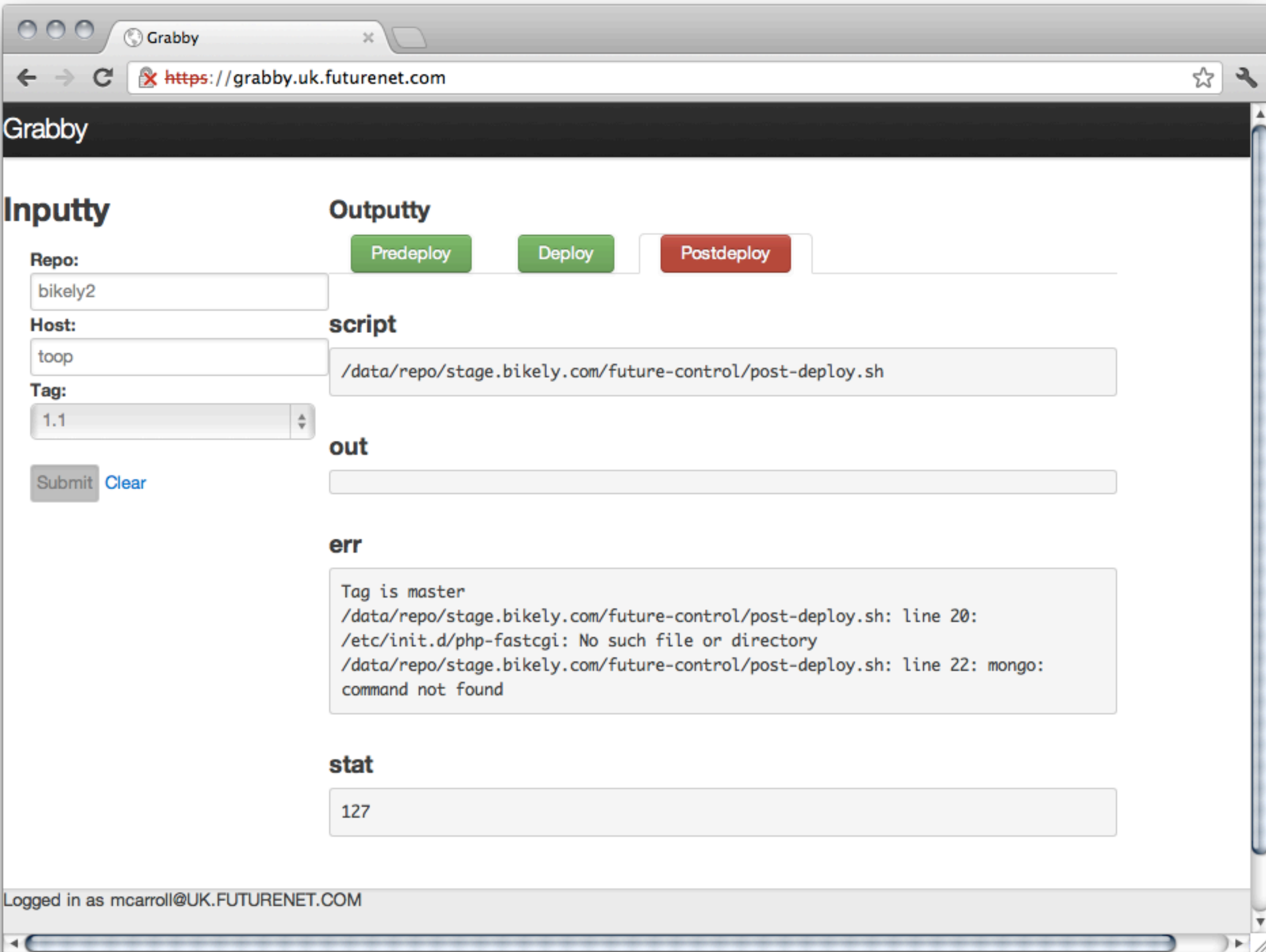












# Making It Moar Better

- Hiera
  - Hierarchical data store with pluggable backends
  - Support for rich data structures
  - “Fall through” data behaviour - look for more specific configurations before more generalised ones
- Atomic deploys!
  - Allows for multiple git repos per site. (Wordpress + plugins + themes, for instance)
  - Performs the actual checkouts away from the currently live tree.
  - Still in the swearing-and-prototype stage. Anyone mentioning second-system effect will get a sarcastic tweeting.

# In Summary

- Both Puppet and MCollective are extremely pluggable.
- So is MQ middleware.
- With a bit of coding you can make something someone else's responsibility.
- Some teams love this and run with it. Others, er, less so.
- Failcake totally works.

Matt Carroll  
Future Publishing Ltd.  
matthew.carroll@futurenet.com

oholiab@gmail.com  
@oholiab  
<http://www.prontab.com>

Special thanks to R.I.Pienaar ([www.devco.net](http://www.devco.net))  
and irc.freenode.net #infra-talk



Matt Carroll  
Future Publishing Ltd.  
matthew.carroll@futurenet.com

oholiab@gmail.com  
@oholiab  
<http://www.prontab.com>

*don't typo!*

Special thanks to R.I.Penaar ([www.devco.net](http://www.devco.net))  
and irc.freenode.net #infra-talk





This version cracked and trained by JHR.  
[john.hawkes-reed@futurenet.com](mailto:john.hawkes-reed@futurenet.com)

@\_JHR\_ #allthegoodonesaretaken  
<http://www.libeljournal.com/>  
[jhr@mysparedomain.com](mailto:jhr@mysparedomain.com)  
<http://ops.failcake.net/>

Code on github, in accordance with  
prophecy. RSN. Honest, guv.

