

G'day!

I'm Lindsay

I'm Lindsay

<http://holmwood.id.au/~lindsay/>

@auxesis

Behaviour driven monitoring with cucumber-nagios







Feature: google.co.uk

It should be up

And I should be able to search for things

Scenario: Searching for things

When I visit "http://google.co.uk/"

And I fill in "q" with "wikipedia"

And I press "Google Search"

Then I should see "www.wikipedia.org"

```
When /^I visit (.*$) do |path|
  pending
end
```

```
When /^I follow "(.*$)" do |link|
  pending
end
```

Mechanize

The Mechanize library is used for automating interaction with websites.

Mechanize automatically stores and sends cookies, follows redirects, can follow links, and submit forms. Form fields can be populated and submitted.

Mechanize also keeps track of the sites that you have visited as a history.

Mechanize

The Mechanize library is used for automating interaction with websites.

Mechanize automatically stores and sends cookies, follows redirects, can follow links, and submit forms. Form fields can be populated and submitted.

Mechanize also keeps track of the sites that you have visited as a history.

Webrat

Webrat lets you quickly write expressive and robust acceptance tests for a Ruby web application.

Features:

- **Browser Simulator for expressive, high level acceptance testing**
- **Supports multiple Ruby web frameworks: Rails, Merb and Sinatra**

Webrat

Webrat lets you quickly write expressive and robust acceptance tests for a Ruby web application.

Features:

- Browser Simulator for expressive, high level acceptance testing
- Supports multiple Ruby web frameworks: Rails, Merb and Sinatra

Webrat::MechanizeAdapter

```
World do
  Webrat::Session.new(
    Webrat::MechanizeAdapter.new
  )
end

# features/support/env.rb
```



```
# features/google.co.uk/search.feature
```

Feature: google.co.uk

It should be up

And I should be able to search for things

Scenario: Searching for things

When I visit "http://google.co.uk/"

And I fill in "q" with "wikipedia"

And I press "Google Search"

Then I should see "www.wikipedia.org"

```
$ cucumber --require features/ \
features/google.co.uk/search.feature
```

Feature: google.co.uk

It should be up

And I should be able to search for things

Scenario: Searching for things

When I visit "http://google.co.uk/"

And I fill in "q" with "wikipedia"

And I press "Google Search"

Then I should see "www.wikipedia.org"

1 scenario (1 passed)

3 steps (3 passed)

0m1.481s

```
$ cucumber --require features/ \
features/google.co.uk/search.feature
```

Feature: google.co.uk

It should be up

And I should be able to search for things

Scenario: Searching for things

When I visit "http://google.co.uk/"

And I fill in "q" with "wikipedia"

And I press "Google Search"

Then I should see "www.wikipedia.org"

1 scenario (1 passed)

3 steps (3 passed)

0m1.481s

--format pretty

cucumber

cucumber

webrat

cucumber

webrat

mechanize

cucumber

webrat

mechanize

nagios

```
$ cucumber-nagios \
  features/google.co.uk/search.feature
```

```
Critical: 0, Warning: 0, 4 okay
```

cucumber-nagios



how it works

```
$ gem install cucumber-nagios  
$ cucumber-nagios-gen project devopsdays  
$ cd devopsdays  
$ gem bundle
```

```
$ cucumber-nagios \
  features/google.co.uk/search.feature
```

```
$ cucumber-nagios \
    features/google.co.uk/search.feature

$ cucumber --require features \
    --formatter Cucumber::Formatter::Nagios \
    features/google.co.uk/search.feature
```

```
$ cucumber-nagios \
    features/google.co.uk/search.feature

$ cucumber --require features \
    --formatter Cucumber::Formatter::Nagios \
    features/google.co.uk/search.feature
```

```
require 'cucumber/formatter/console'

module Cucumber
  module Formatter
    class Nagios
      end
    end
  end
```

```
def initialize(step_mother, io, options={})
  @failed = []
  @passed = []
  @warning = []
  @io = io
end

def after_step_result(... status ...)
  case status
  when :passed
    @passed << step_match
  when :failed
    @failed << step_match
  when :undefined
    @warning << step_match
  end
end
```

```
def after_features(steps)
    print_summary
end

private
def print_summary
    @total = @failed.size + @passed.size + @warning.size
    message = ""
    message += "Critical: #{@failed.size}, "
    message += "Warning: #{@warning.size}, "
    message += "#{@passed.size} okay"
    # nagios performance data
    message += " | passed=#{@passed.size}"
    message += ", failed=#{@failed.size}"
    message += ", nosteps=#{@warning.size}"
    message += ", total=#{@total}\n"

    @io.print(message)
    @io.flush
end
```



```
#!/usr/bin/env ruby

__DIR__ = File.expand_path(File.dirname(__FILE__))
feature = ARGV[0]

command = "#{__DIR__}/cucumber"
command += " --require #{__DIR__}/common.rb"
command += " --require features/"
command += " --format Nagios::NagiosFormatter"
command += " #{feature}"

system(command) ? exit(0) : exit(2)
```

```
#!/usr/bin/env ruby

__DIR__ = File.expand_path(File.dirname(__FILE__))
feature = ARGV[0]

command = "#{__DIR__}/cucumber"
command += " --require #{__DIR__}/common.rb"
command += " --require features/"
command += " --format Nagios::NagiosFormatter"
command += " #{feature}"

system(command) ? exit(0) : exit(2)
```

good

bad

ugly

caveats

AJAX

Progressive Enhancement / Graceful Degradation

**existing
features + steps**

Mechanize +
Nokogiri

multiple scenarios

(Critical: 0, Warning: 0, 2 okay)
(Critical: 0, Warning: 0, 4 okay)

but why?

asking the
wrong
questions

**ping
tcp connect**

**is my server up?
can I see my app?**

dead vm?

is my server up?

can I see my app?



serving content?



the right
questions

is my app behaving?

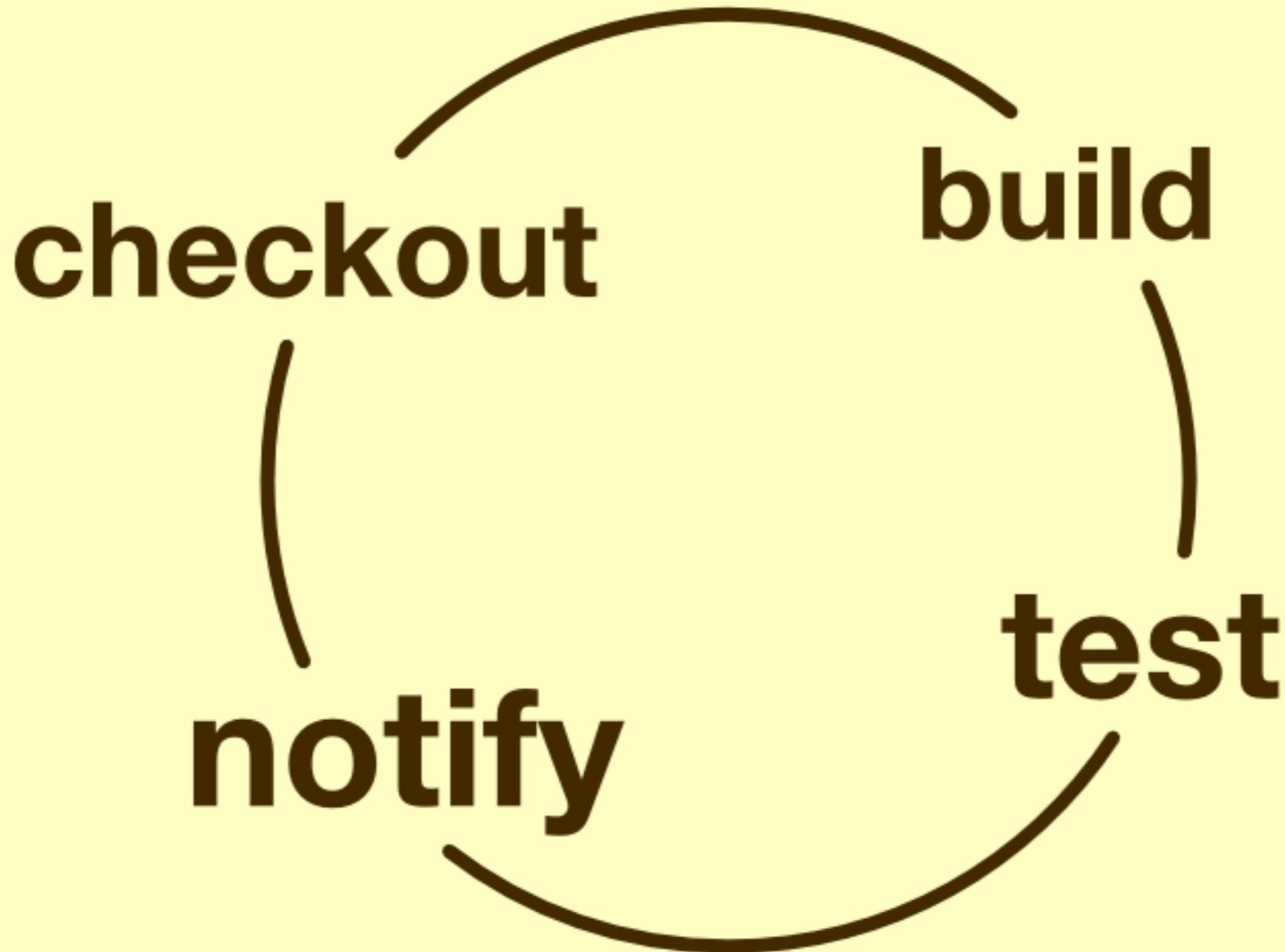
can I navigate?

can I sign in?

can I place an order?

can I ...?

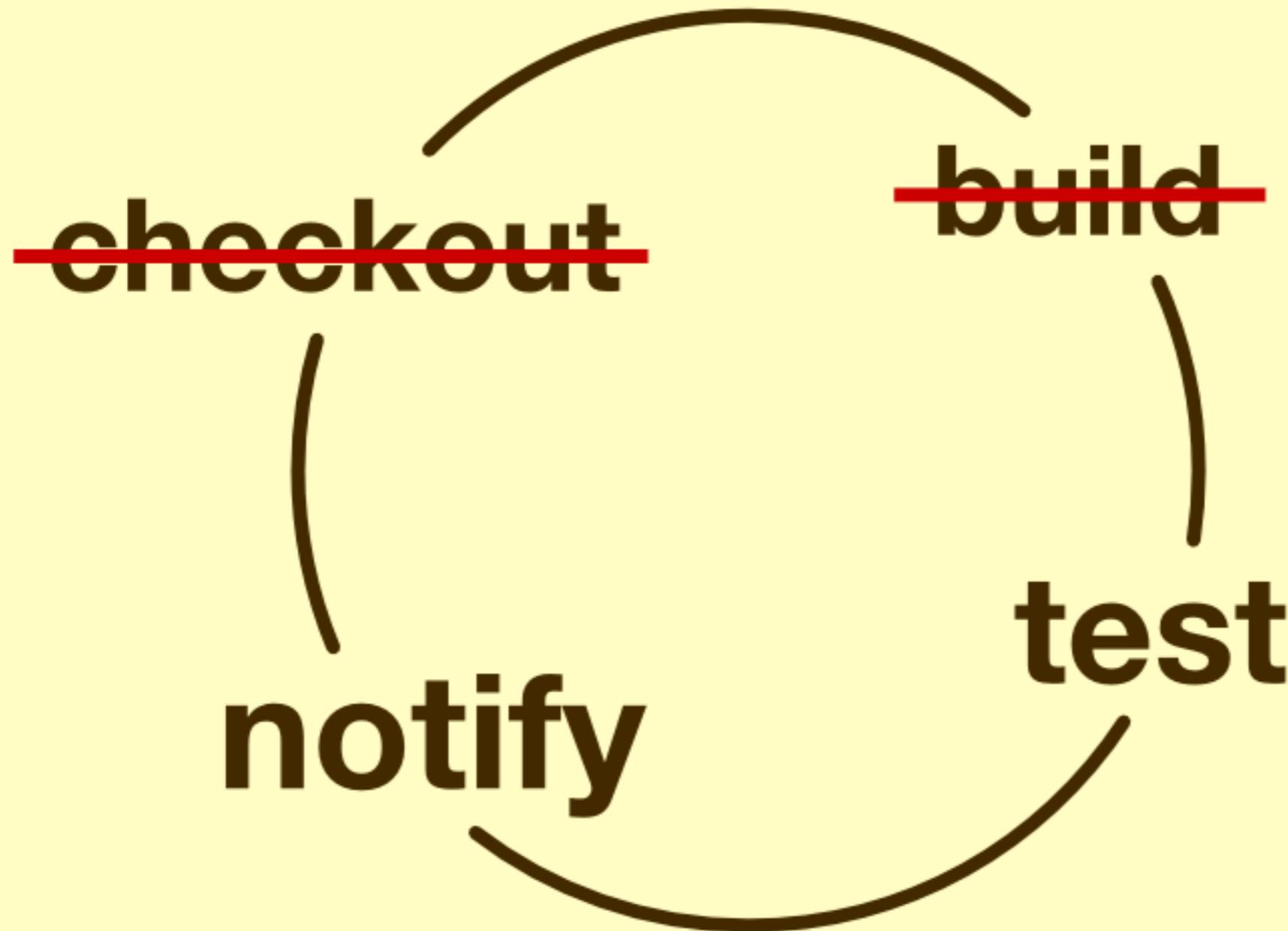
**continuous
integration**



ci lifecycle

monitoring:

**continuous
integration
for production
apps**



ci lifecycle

notify

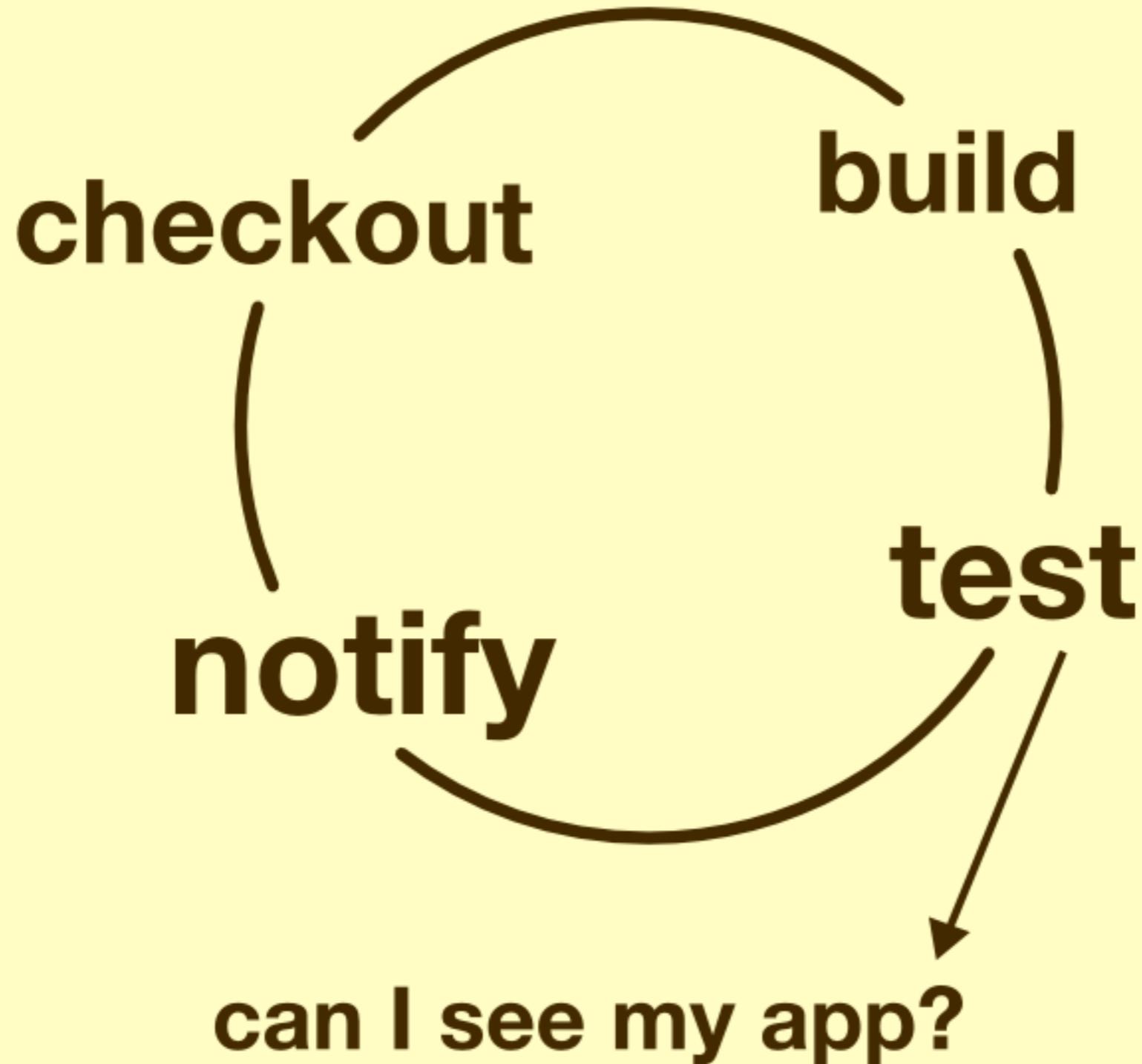
test

monitoring system

notify

test

can I see my app? (tcp connect)





LOL WUT

notify

test

can I see my app?

cool uses

(Behaviour Driven Monitoring)

Telephony BDM

The screenshot shows a web browser window with the title bar "Shiretoko". The address bar displays the URL <http://blog.goecke.net/2009/06/10/behavior-driven-systems-monitoring-for-telephony>. The main content area features a large image of white clouds against a blue sky. Overlaid on the image is the title "Warheads Stacked in the Kitchen" in a bold, serif font.

Below the title, the text "Jason Goecke's blog" is on the left and "Home About" are on the right. A horizontal line separates the header from the main article.

The article title is "Behavior Driven Systems Monitoring for Telephony", followed by the date "2009 JUNE 10".

By "jasongoecke" (with a small profile icon), tags include: Adhearsion, Asterisk, bdd, behavior driven development, cucumber, cucumber nagios, cukes, euruko, nagios, Ruby, sipper, sipr, systems management, tdd, telephony, test, test driven development, testing.

Text about attending Euruko and learning about Cucumber Behavior Driven Development (BDD) is present.

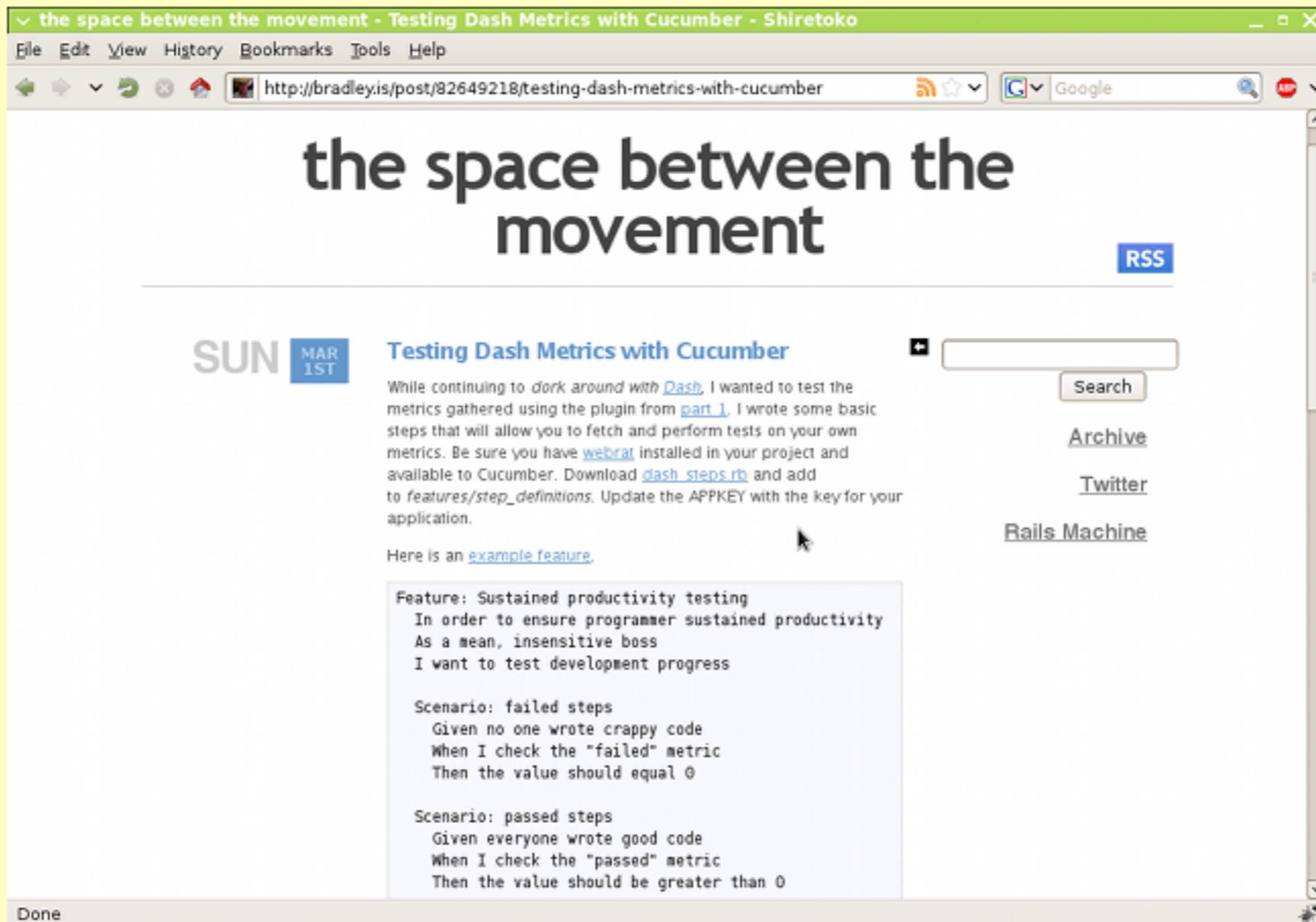
A sidebar on the right contains a "SUBSCRIBE" section with an RSS feed icon, a link to "Subscribe to Warheads Stacked in the Kitchen by Email", and a "FIND ME ON THE INTERNETS" section with links to Jason Goecke's Twitter account (@jsgoecke) and LinkedIn profile.

The Cucumber logo is visible in the bottom right corner of the main content area.

with **telephony-system-tests**

(by @jsgoecke)

Dash metrics



cucumber + dash

(by @bradleyktaylor)

Instead of writing boring monitoring plugins from scratch, you can now do behaviour driven ops.

Transform from a grumpy, misanthropic sysadmin to a hipster, agile developer instantly.

- Bradley Taylor



Thank you!

<http://auxesis.github.com/cucumber-nagios>

Dash metrics - <http://bit.ly/IWWfE>

Telephony BDM - <http://bit.ly/94TAn>