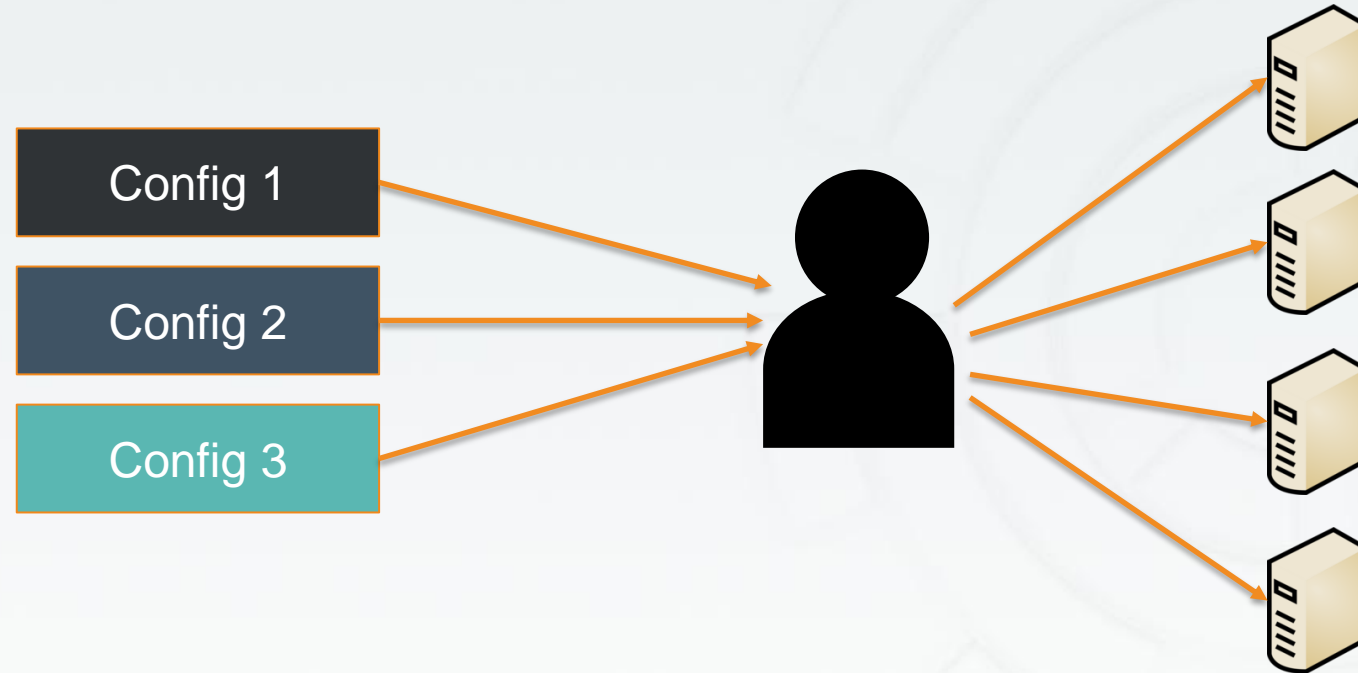# CHEF™

Cheffing your DevOps

Michael Ducy – Chef – @mfdii
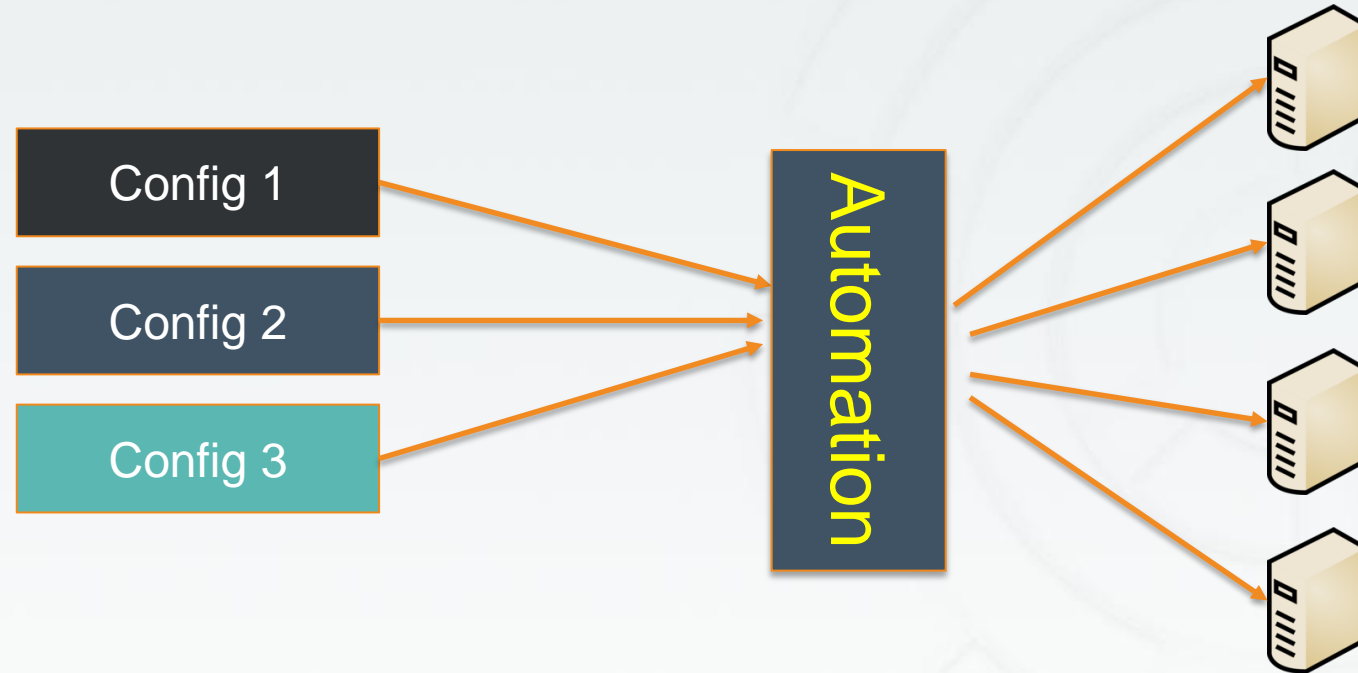
What is Chef?

Short Answer: An Automation Framework
for automating infrastructure and applications
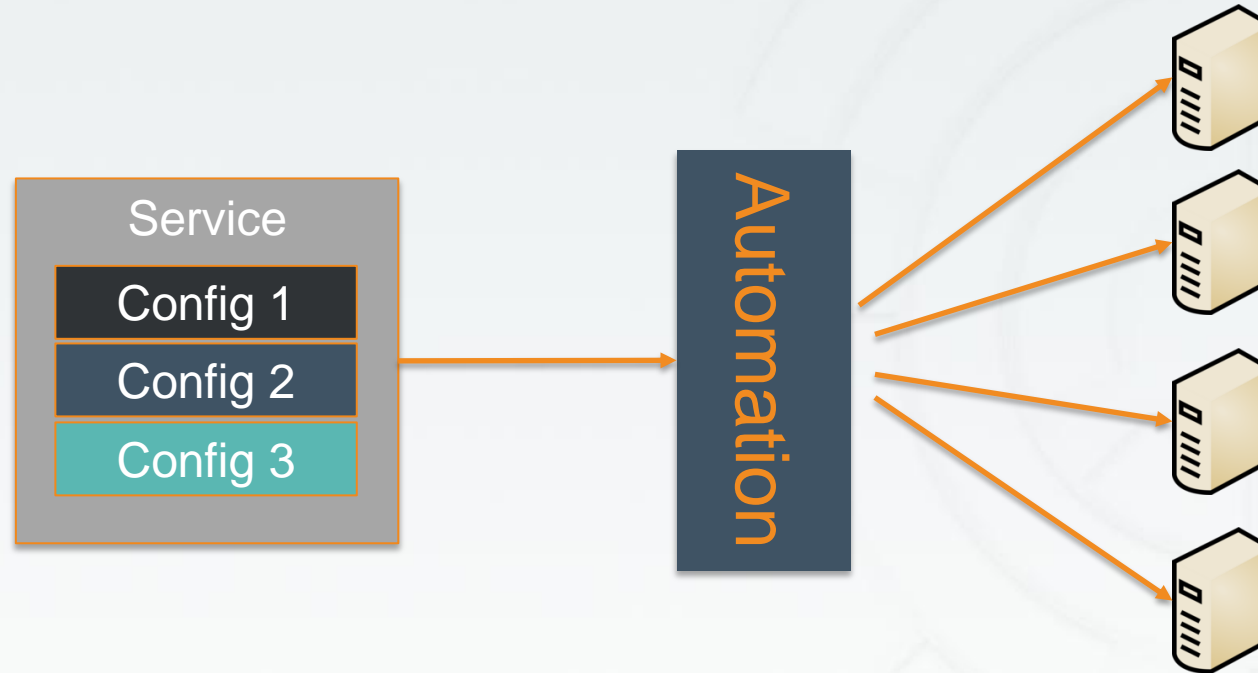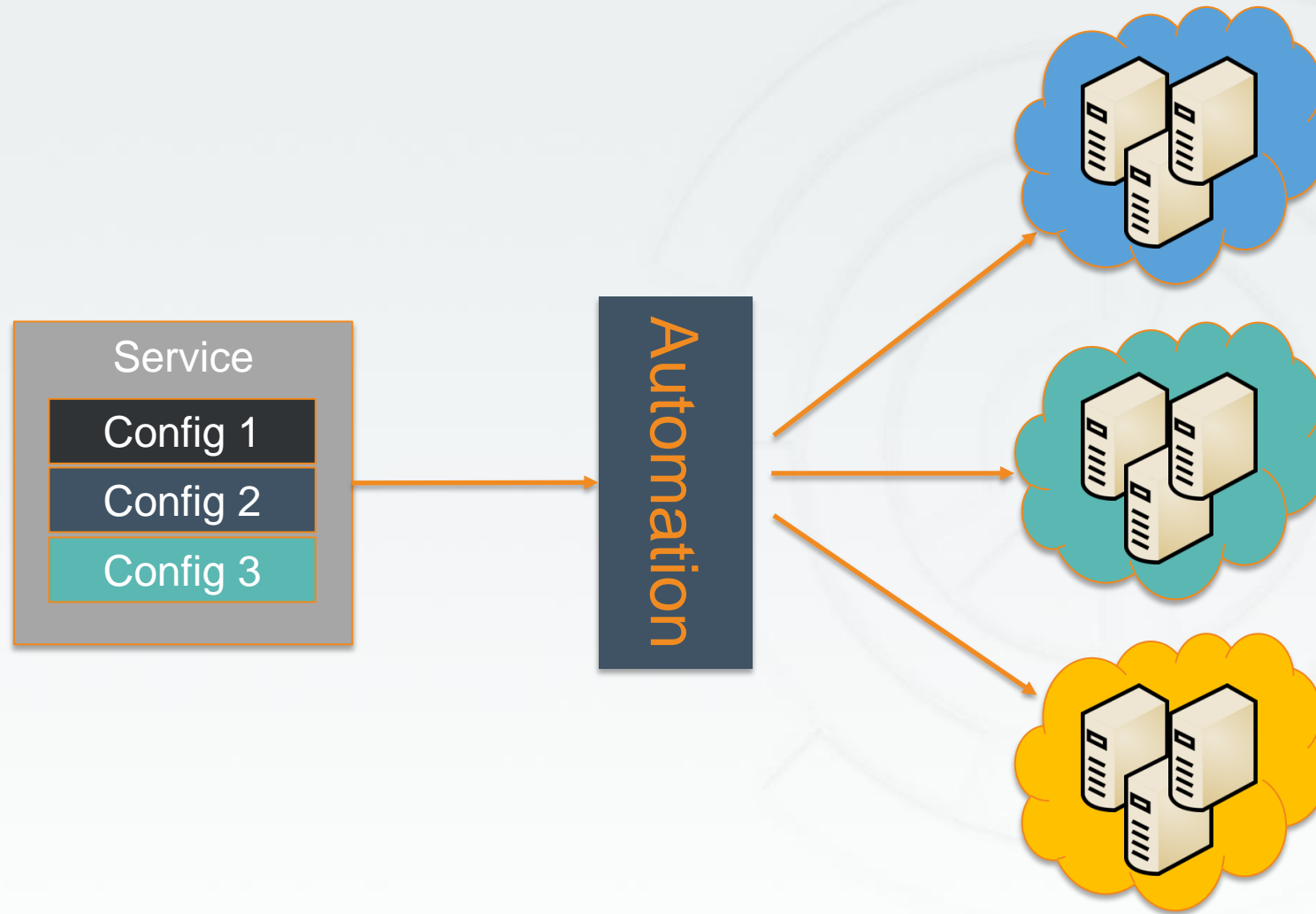
CHEF

Traditional Systems Management
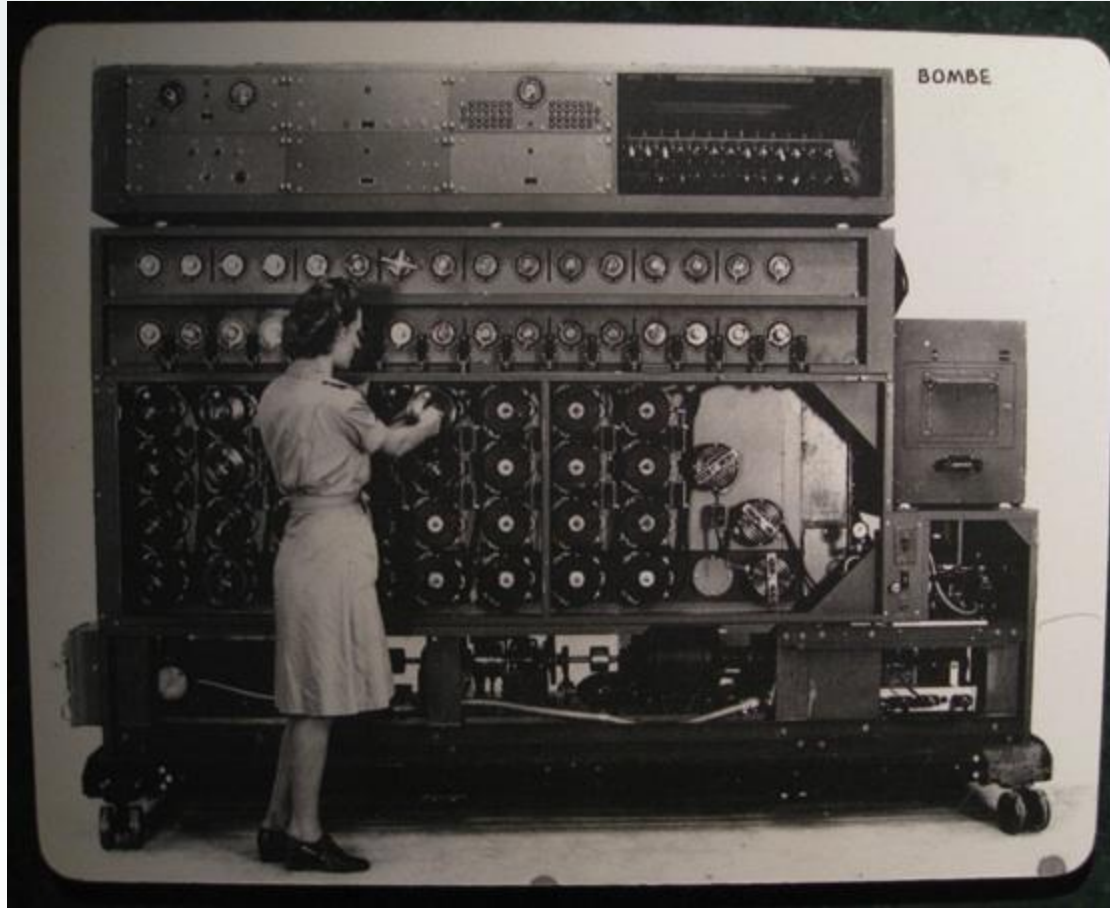
Traditional Systems Management

Idea of Services

Abstraction of Services

Chef is Infrastructure as Code

BOMBE

- Programmatically provision and configure components

CHEF

Chef is Infrastructure as Code

BOMBE

- Treat like any other code base

Chef is Infrastructure as Code

BOMBE
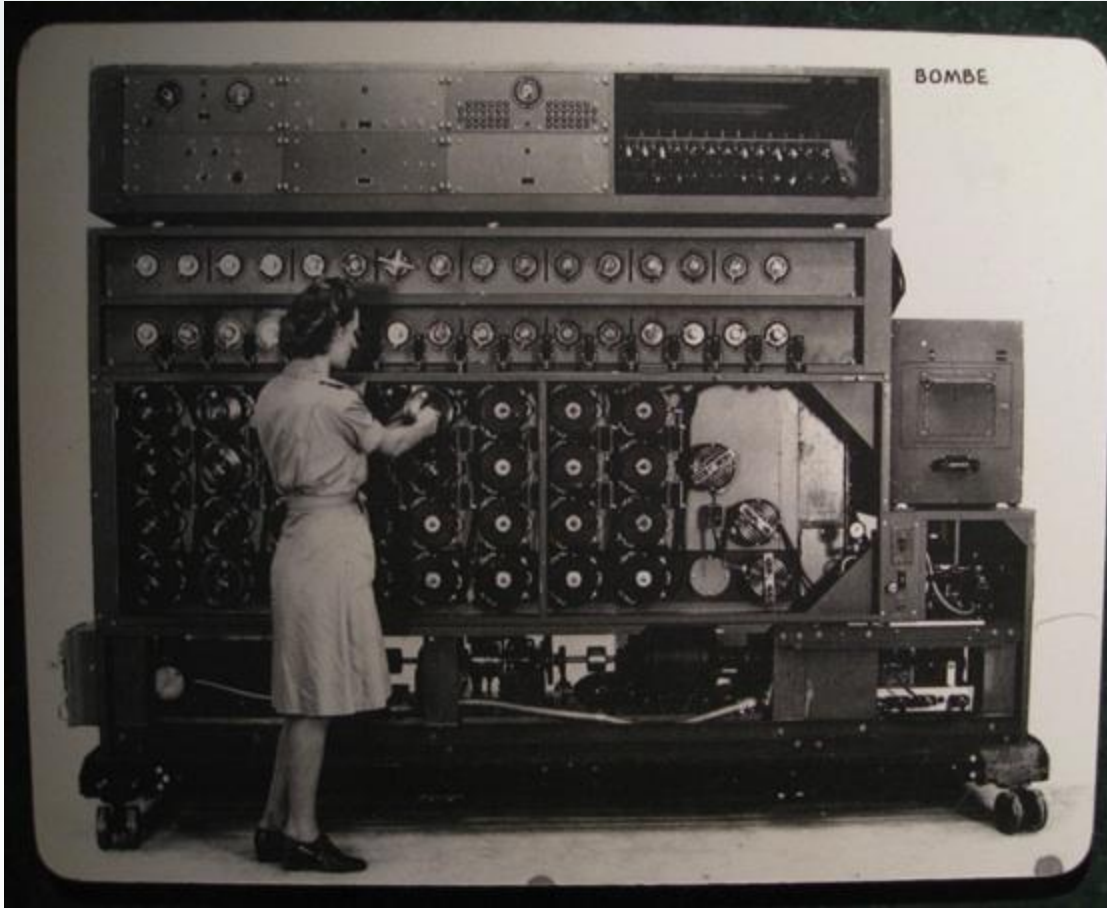
- Reconstruct business from **code repository**, **data backup**, and compute resources

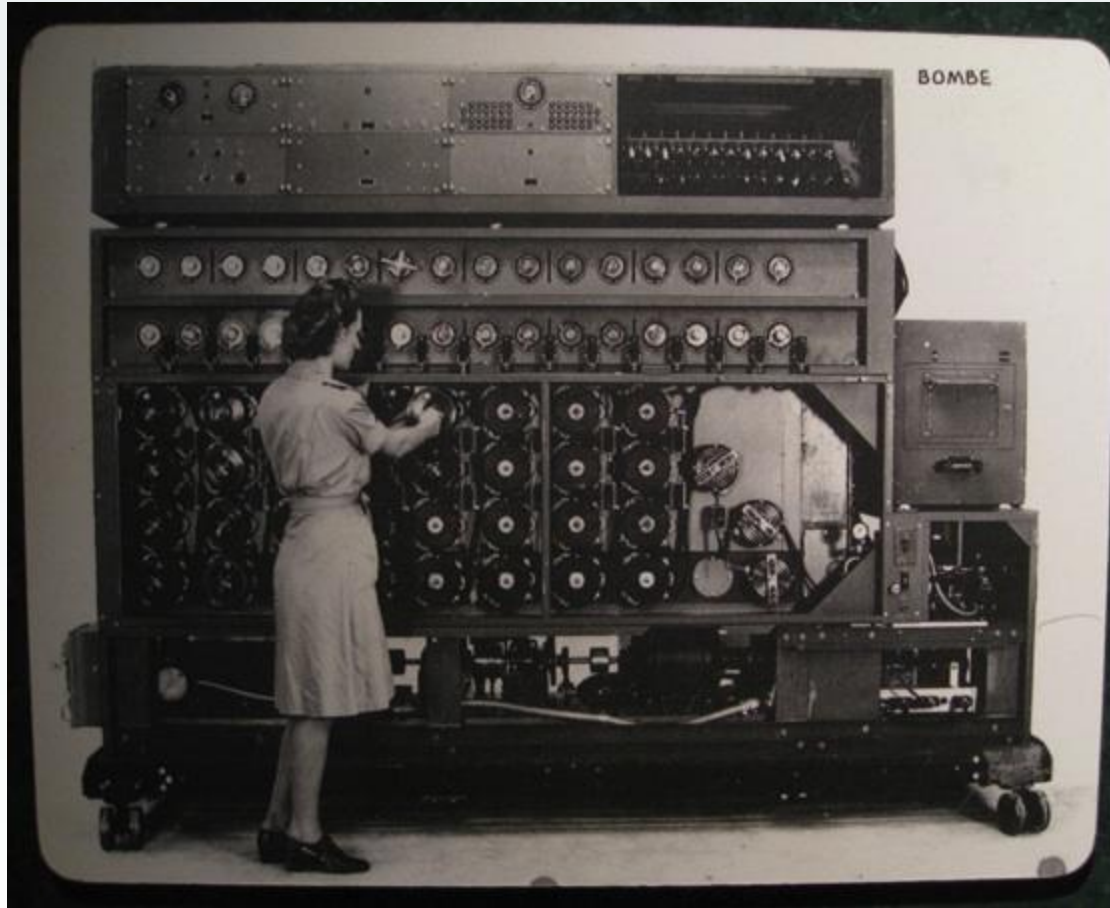# Chef is Infrastructure as Code

BOMBE

- Programmatically provision and configure components
- Treat like any other code base
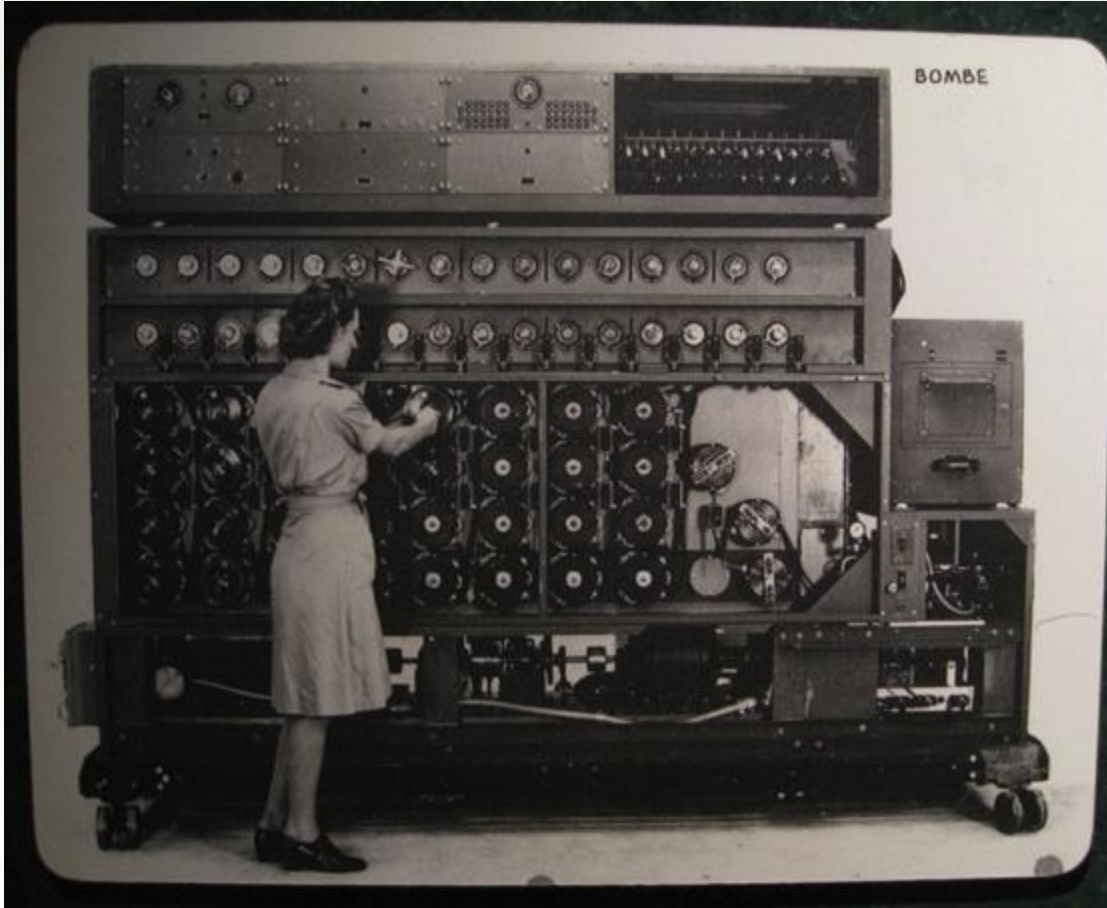- Reconstruct business from **code repository**, **data backup**, and compute resources

CHEF

The Chef Software Platform

Building Blocks

Cookbooks

Recipes
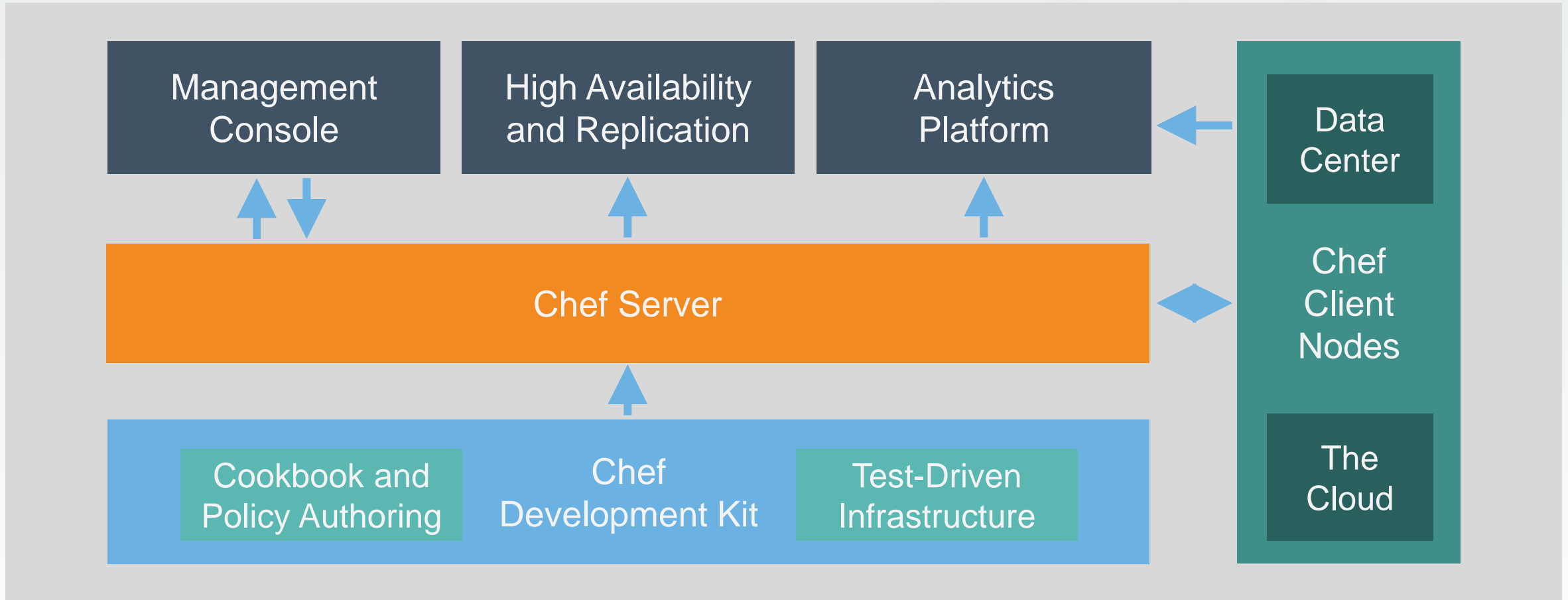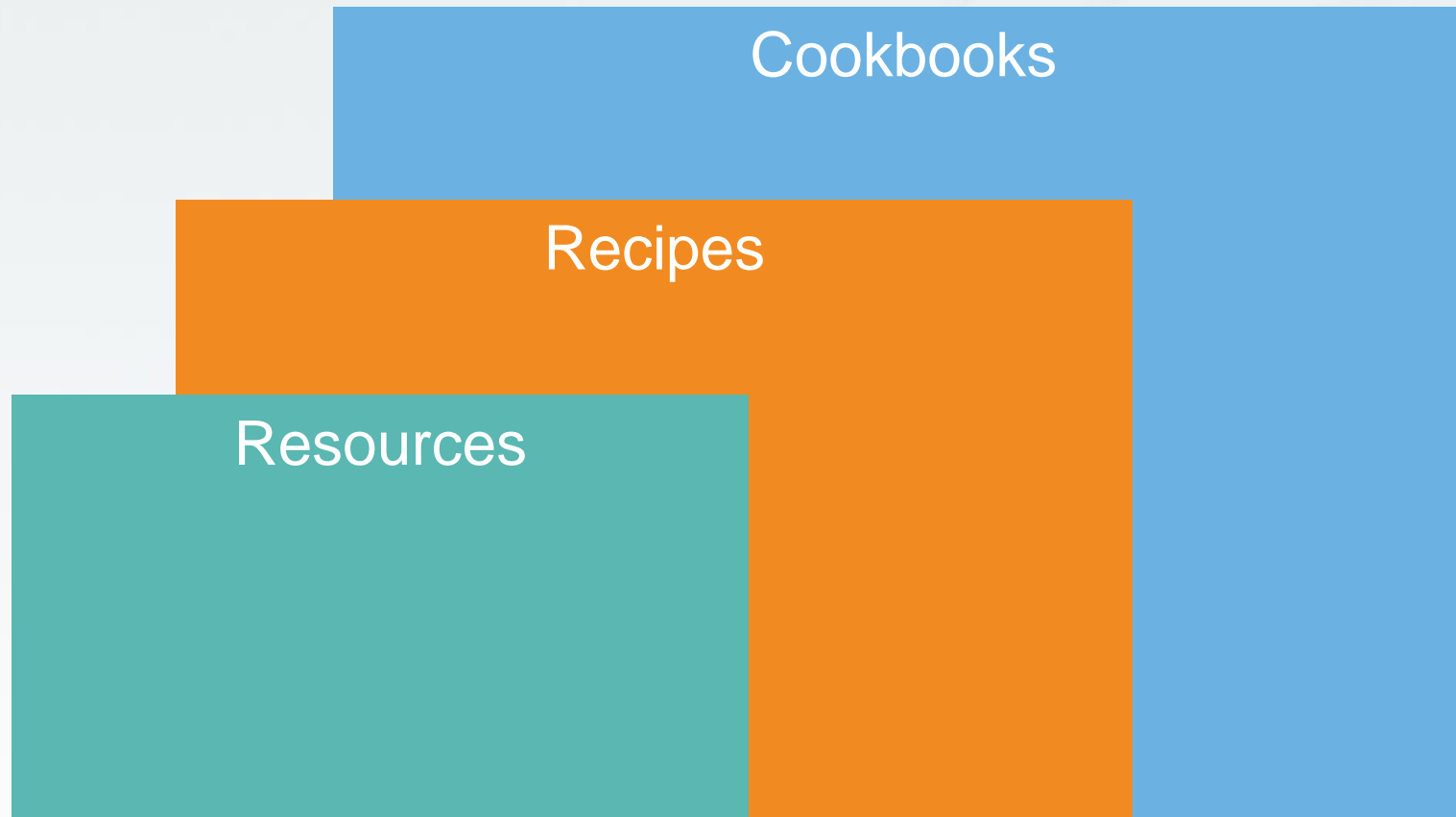
Resources

CHEF

Building Blocks: What is a Resource?

- A Resource is a system state you define

    Example: Package installed, state of a service, configuration file existing

- You declare what state you want the resource in.

    Chef automatically determines HOW that state is achieved

On Linux based OSes:

```
1 ▼  package "apache" do
2        action :install
3 ▲  end
4    |
```

On Windows based OSes:

```
1 ▼  windows_feature 'IIS-WebServerRole' do
2        action :install
3 ▲  end
4
5 ▼  windows_feature 'IIS-ASPNET' do
6        action :install
7 ▲  end
```

Resource Example

```
#windows
dsc_resource 'webserver' do
    resource :windowsfeature
    property :name, 'Web-Server'
    property :ensure, 'Present'
end
```

CHEF

Resource Example

```
#linux
package "httpd" do
    action :install
end
```

Building Blocks: What is a Recipe?

- An abstraction of a Service that consists of a set of Resources to deliver that Service
- Resources are executed in the order they are listed.

CHEF

Recipe Example

```
#linux
package "httpd" do
     action :install
end


include_recipe "apache::fwrules"


service "httpd" do
     action [ :enable, :start ]
end
```

Recipe Example

```
#windows
include_recipe "fourthcoffee::dsc"
include_recipe "iis::remove_default_site"
remote_directory node['fourthcoffee']['install_path'] do
        source 'fourthcoffee'
        action :create
end
iis_pool 'FourthCoffee' do
        runtime_version "4.0"
        action :add
end
iis_site 'FourthCoffee' do
        protocol :http
        port 80
        path node['fourthcoffee']['install_path']
        application_pool 'FourthCoffee'
        action [:add,:start]
end
```

CHEF

Cookbooks

- A Higher Level Abstraction of a Service
- A set of Recipes and Data Attributes required to deliver one or multiple Services
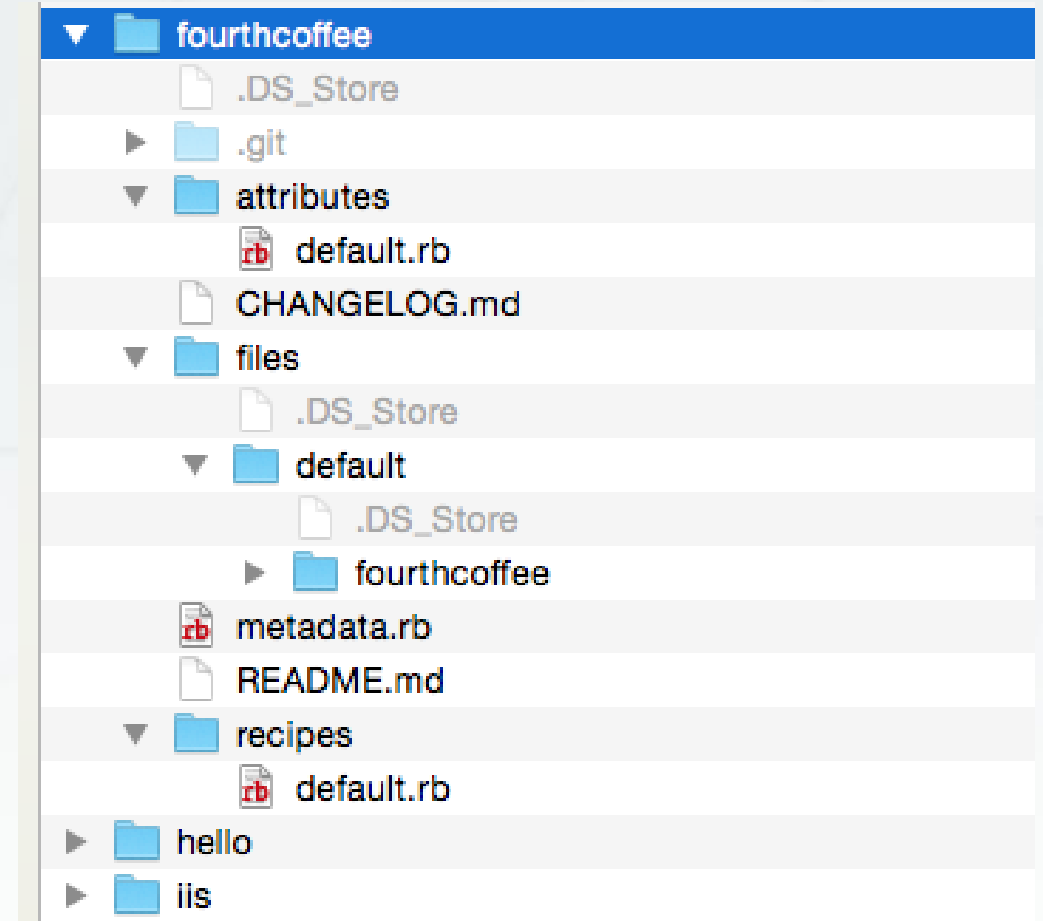
Define cookbook attribute

```
#attributes.rb
default['fourthcoffee']['port'] = 80
```

Consume cookbook attribute

```ruby
iis_site 'FourthCoffee' do
    protocol :http
    port node['fourthcoffee']['port']
    path node['fourthcoffee']['install_path']
    application_pool 'FourthCoffee'
    action [:add,:start]
end
```
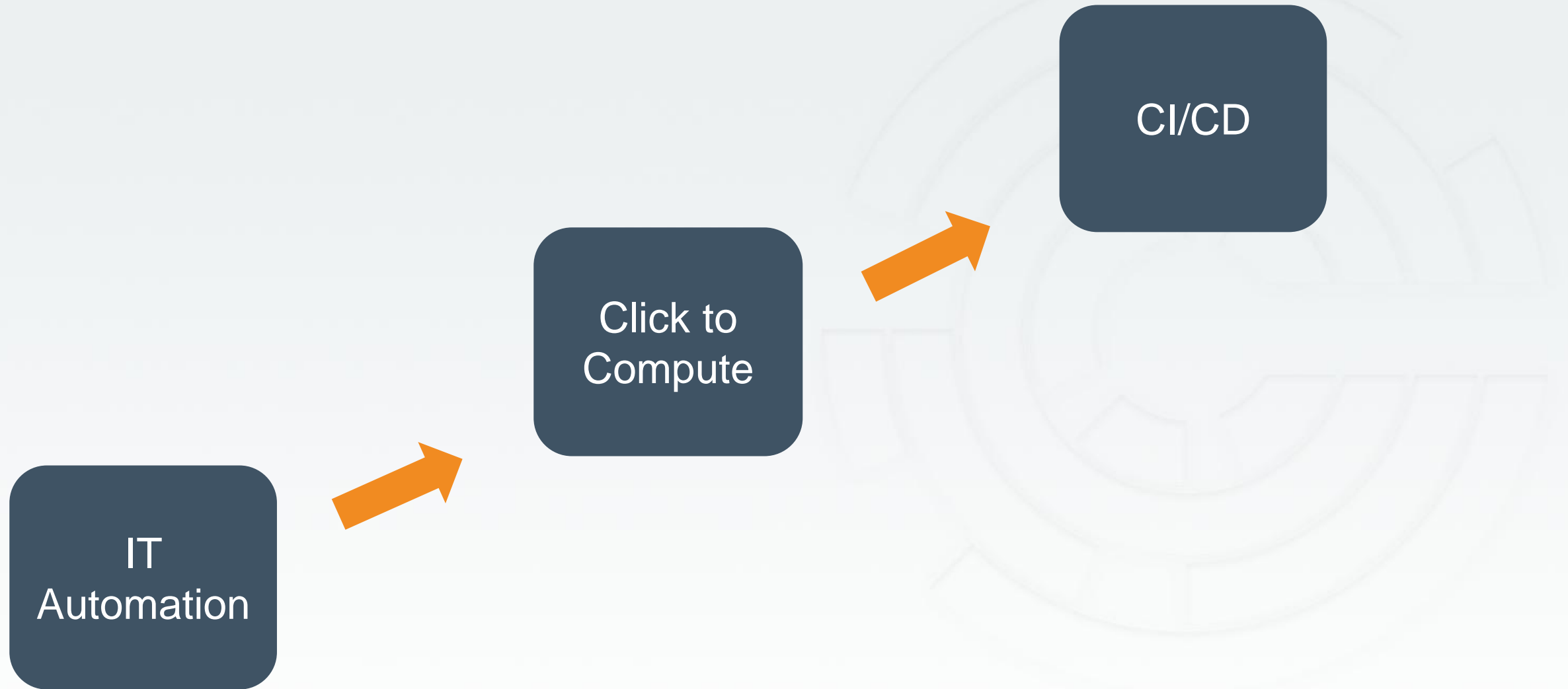
CHEF

Demo!

Yes!

That's cool but...

- Things break
- Chef is a language (based on Ruby)
- How can you rapidly develop recipes and cookbooks?

CHEF

Let's step back⋯

Automation Patterns

IT Automation → Click to Compute → CI/CD

CHEF

Manual Tests

Automated
GUI Tests

Integration
Tests

Unit
Tests
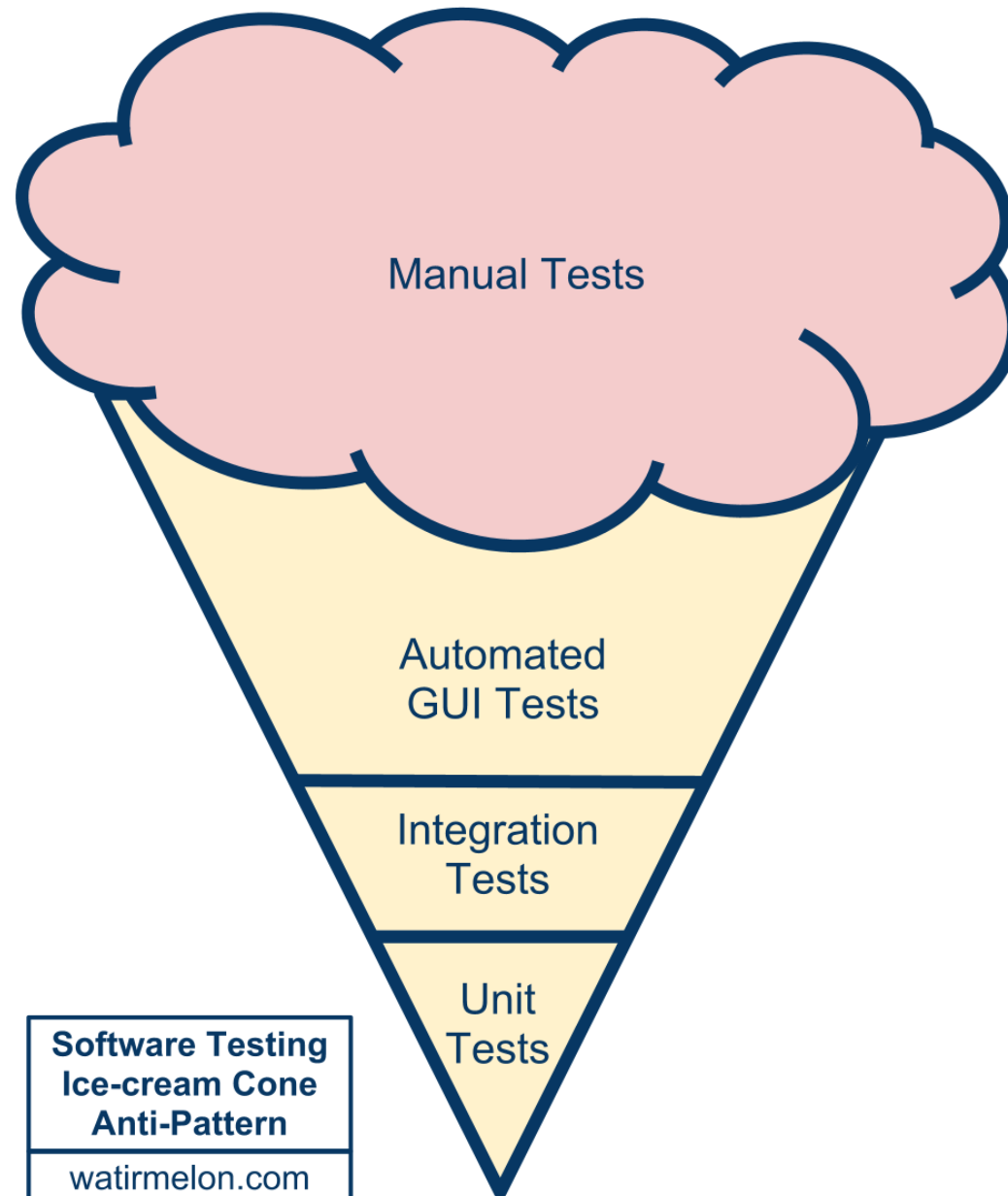
**Software Testing
Ice-cream Cone
Anti-Pattern**

watirmelon.com

Ideal Software Testing Pyramid

watirmelon.com

Manual Session Based Testing

Automated GUI Tests

Automated API Tests

Automated Integration Tests

Automated Component Tests

Automated Unit Tests

CHEF
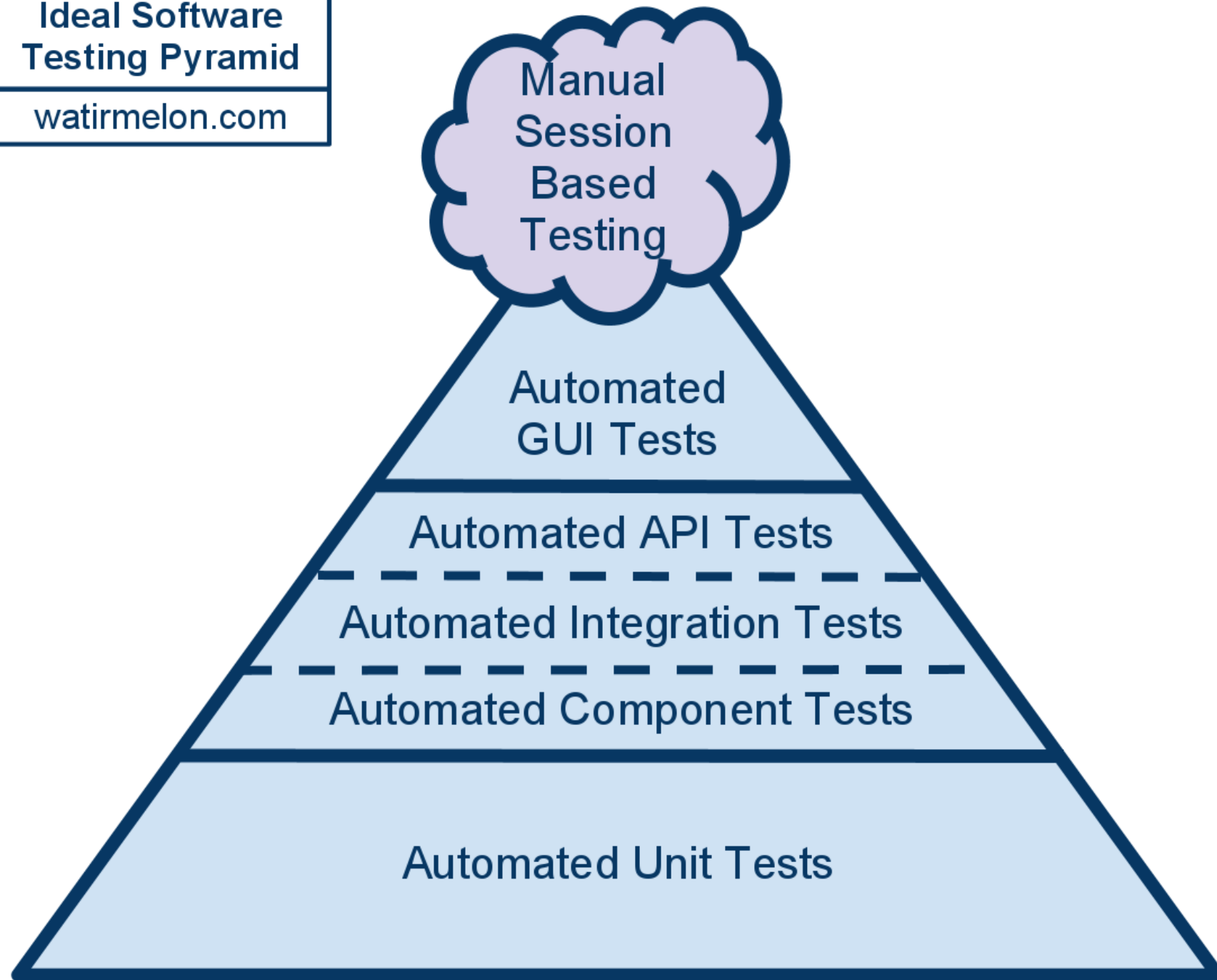
Also known as…

You'll never get to
Continuous Deployment
clicking a GUI

Testing builds safety

Feedback loops...

- Tell us we're doing the right thing
- At the right time
- With the right results
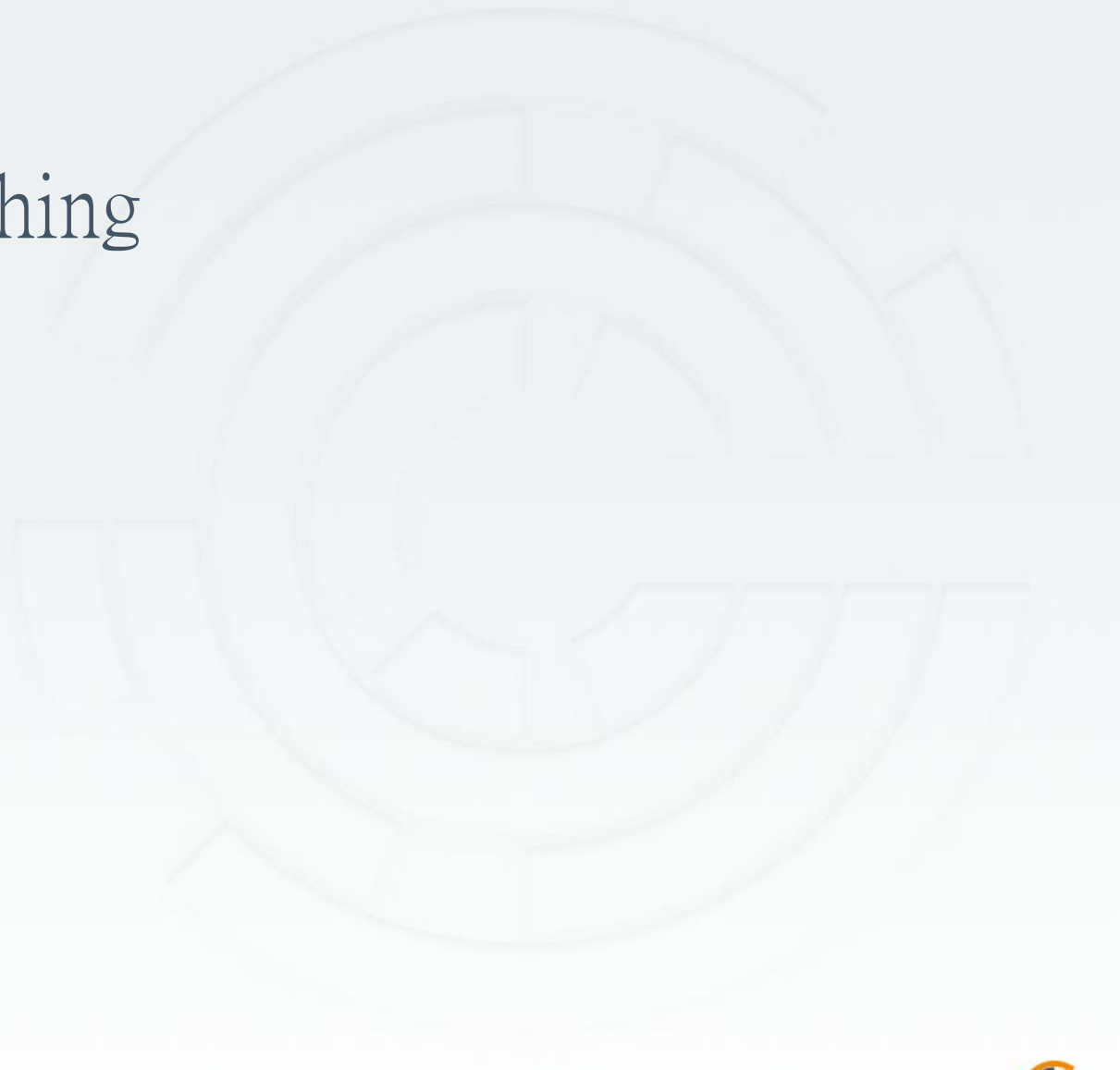
CHEF

Feedback loops···

Measurements we take to ensure the "experiment" is behaving as expected

CHEF

Tests are essentially feedback loops

Remember…

Chef is "Infrastructure as Code"

CHEF

Remember…

"Infrastructure as Code" should be treated like ANY other codebase.

CHEF

Treated Any Other Codebase...

- Stored in SCM
- Testing Coverage
- Part of your CI pipelines

Testing in Chef

- Chef recipes need tested
    Linting
    Static Analysis
    Unit Testing
    Functional Testing

Food Critic
Test Your "Chef Style"

- Flag problems that might cause your Chef cookbooks to fail on converge

  FC010: Invalid search syntax

- Identify style/convention that has been adopted by the Chef community

  FC004: Use a service resource to start and stop services

- Create custom rules for your own organization's compliance/standards

  COMP001: Do not allow recipes to mount disk volumes

Rubocop

- Identify potential Ruby errors

    Unclosed strings, etc.

- Identify style/convention that helps write better code

    Single quotes vs. double quotes

    This is useful for new Chefs, and helps make the code more readable

ChefSpec
Simulate Chef

- Did I send a properly formed piece of code to Chef?

- Especially important if there is mutability in your code

- Useful for regression testing – to make sure new changes don't break stuff that worked before.

Test Kitchen
Let's do this (almost) for real

- "Executes your Chef code on an actual, factual node

- These nodes should be disposable (local virtualization, cloud instances, etc.)

- Use any number of testing products to ensure expected results

  BATS

  ServerSpec

  Pester

- Can pull in other cookbook dependencies as well, and execute against a machine that looks the same as your standard image

Demo!

Summary

- Chef is awesome
- Testing is important
- Feedback loops build safety
- Goal is to eliminate waste

Questions?

Cheffing your DevOps

Michael Ducy – Chef – @mfdii