Puppet Ain't No Voodoo

Bartosz Majsak, Thomas Hug

Copyright © 2014 Cambridge Technology Partners All Rights Reserved. Cambridge, its logo, and Get IT Right are trademarks of Cambridge Technology Partners.

About Us

- Bartosz Majsak
  - Java Developer by day
  - Open source junkie by night (Arquillian core team member)
  - Conference speaker by passion (Devoxx, Jazoon ...)

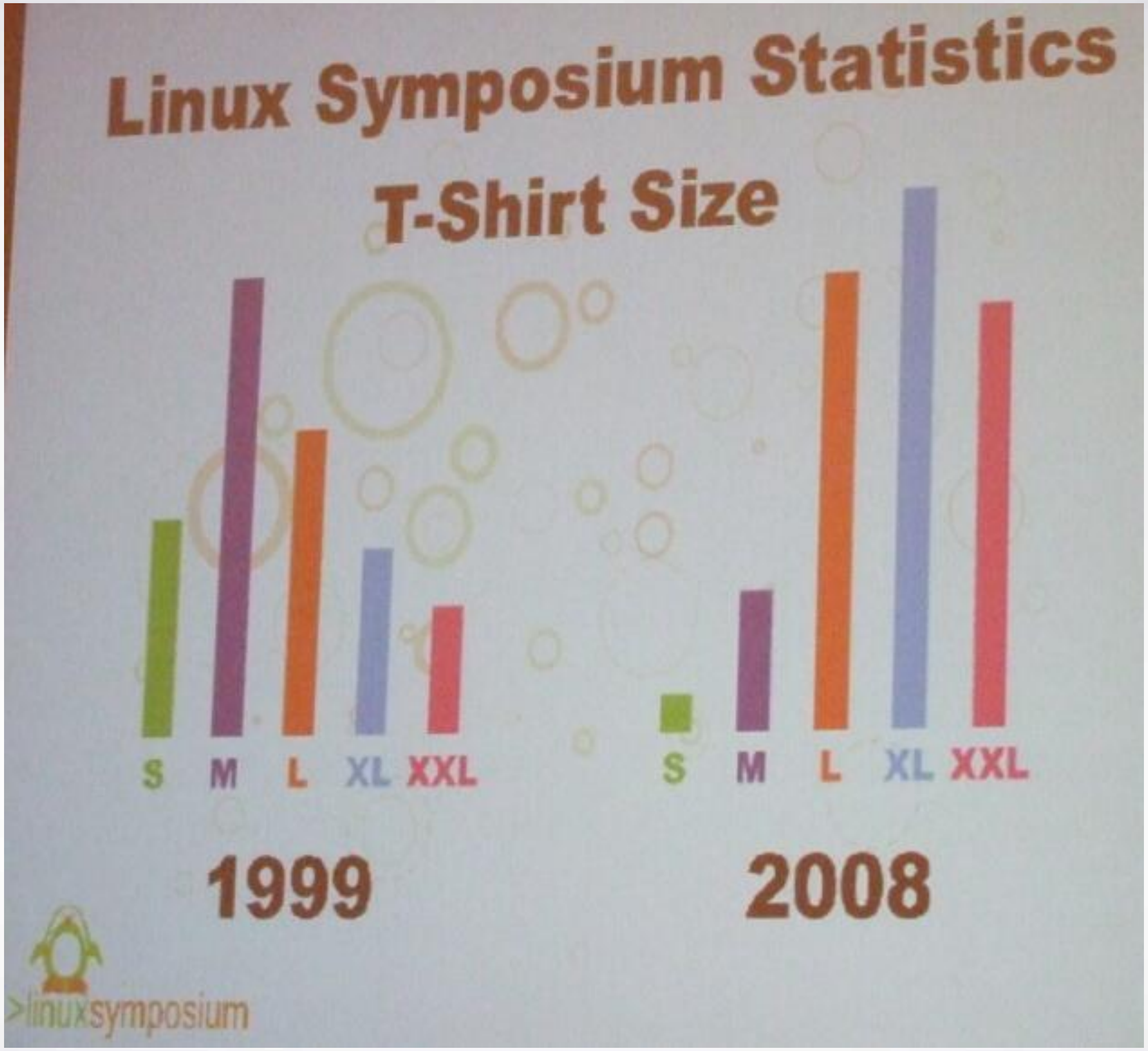
@majson

- Thomas Hug
  - With Cambridge Technology Partners since 2002
  - Java Developer, TTL, Solution Architect
  - Apache Committer, OSS contributor and aficionado

@gizmoo360

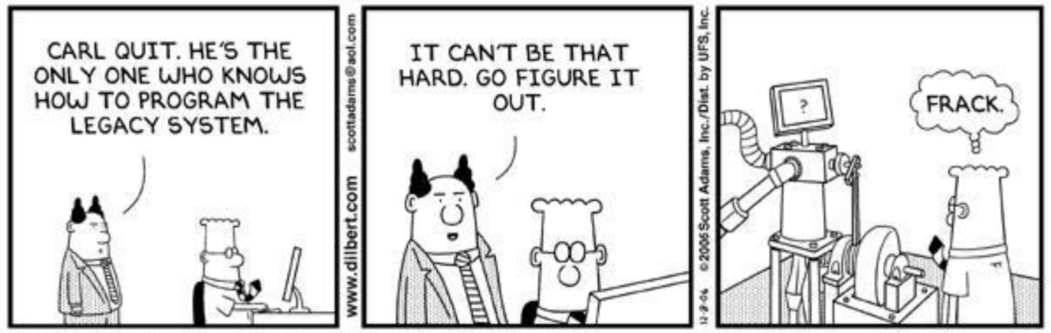
© 2014 Cambridge Technology Partners

Why we recommended Linux?



Why Puppet?

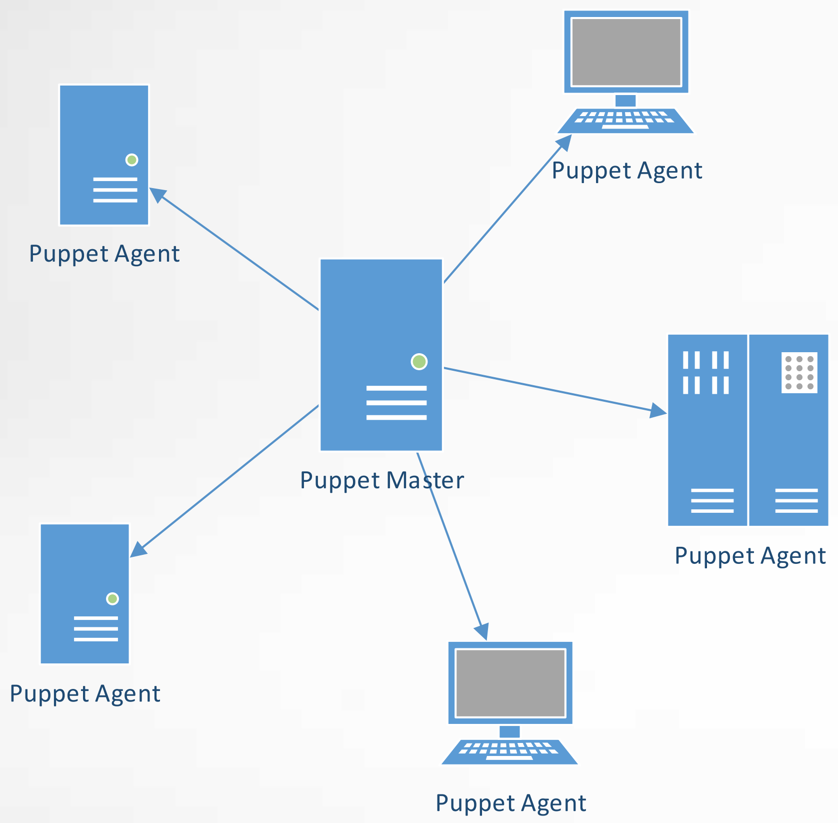
- Infrastructure as code
  - Version Control
  - Reproducible
- Configuration Management
  - Code = Documentation (if it's high-level enough)
  - Readability
  - Always in sync (if versioned and released properly)
- Automation
  - Manual = slow and error prone
  - Scalable over large installations

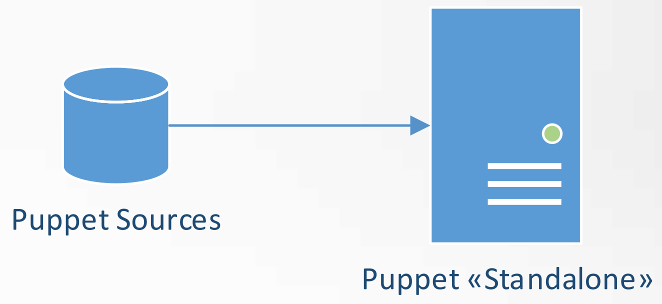

CARL QUIT. HE'S THE ONLY ONE WHO KNOWS HOW TO PROGRAM THE LEGACY SYSTEM.
IT CAN'T BE THAT HARD. GO FIGURE IT OUT.
FRACK.
www.dilbert.com scottadams@aol.com
©2006 Scott Adams, Inc./Dist. by UFS, Inc.
© Scott Adams, Inc./Dist. by UFS, Inc.

© 2014 Cambridge Technology Partners

Puppet Architectures

- Puppet Master/Agent vs. «Standalone» mode





© 2014 Cambridge Technology Partners

The Puppet Ecosystem

Cambridge Technology Partners

- The Foreman
  - Advanced provisioning solution
  - ENC support
- Puppet Dashboards
  - Reporting
- Puppet Forge
  - Reusable Puppet modules
  - Share publicly

FOREMAN

RED HAT® SATELLITE

puppet forge

Alternatives and Correlated

CHEF™

ANSIBLE

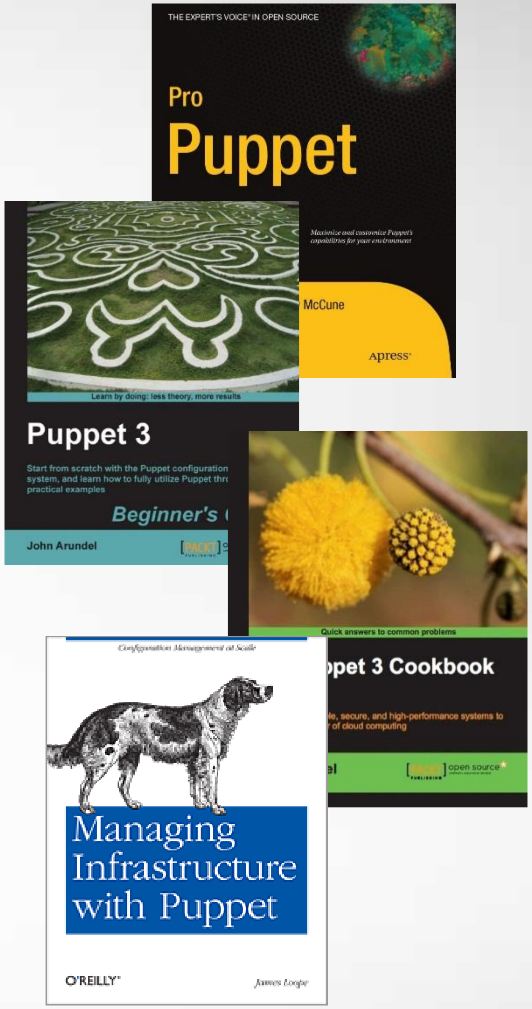
VAGRANT

docker

© 2014 Cambridge Technology Partners

References

- Official Puppet Documentation: https://docs.puppetlabs.com/
- Puppet Type Reference: http://bit.ly/chopen-puppet-typeref
- Puppet Cookbook: http://www.puppetcookbook.com/
- You know this one anyway: http://stackoverflow.com/



© 2014 Cambridge Technology Partners

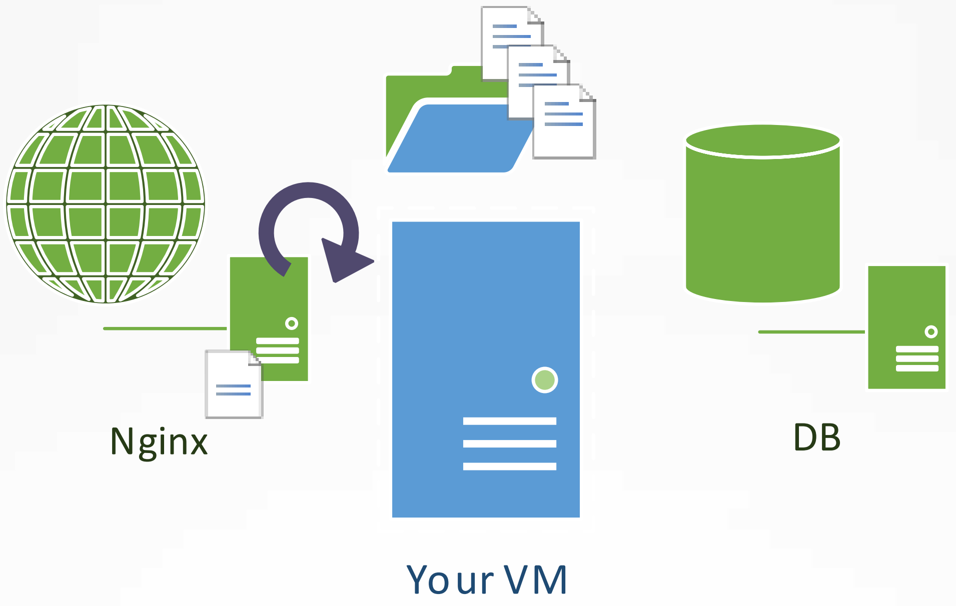
Cambridge Technology Partners

Getting Started

© 2014 Cambridge Technology Partners

«The Master Plan»

- Building a web server



Installation

- Ubuntu and Debian

```
wget http://apt.puppetlabs.com/puppetlabs-release-$(lsb_release -sc).deb
sudo dpkg -i puppetlabs-release-$(lsb_release -sc).deb
sudo apt-get update
sudo apt-get -y install puppet ruby1.9.1-dev
sudo gem install librarian-puppet puppet-lint
```

  - Specific Ruby version is needed by Librarian

- Dedicated installers for Win and Mac OS X
  - http://downloads.puppetlabs.com/windows
  - http://downloads.puppetlabs.com/mac/

The Puppet DSL

- The Shell Way:

```
if id -u ‘tom’> /dev/null 2>&1; then
    echo “User exists, skipping"
else
    sudo adduser --disabled-password --gecos ‘Tom Hug' tom
    echo "user created"
fi
```

…and if we want to update? Delete?

- The Puppet Way:

```
user { ‘tom’:
    ensure   => exists,
    password => ‘*’,
    comment  => ‘Tom Hug’
}
```

Puppet Resources

Executions

Buckets

Files

Schedules

Services

Packages

Users and Groups

Keys

Configuration

- Find a list and reference at https://docs.puppetlabs.com/references/latest/type.html

© 2014 Cambridge Technology Partners

A First Manifest

- Setting up our site

```
$ subl site.pp

file { '/tmp/hello.txt':
    content => 'Hello World'
}

$ puppet apply site.pp
…
$ cat /tmp/hello.txt
```

- A typical resource:

```
RESOURCE { NAME:
    ATTRIBUTE => VALUE,
    ...
}
```

The Labs

- Interactive Labs
  - We will get started with the solution…
  - …and leave some parts to complete
  - Most build on top of each other!
- Full solution at GitHub
  - https://github.com/ctpconsulting/chopen-workshop-puppet-labs
  - Predefined checkpoints (tags) to advance / go back

```
$ git clone http://git.io/dX0Dxg labs_solution
$ cd labs_solution
$ git tag
$ git checkout -f lab01
```

Lab 01

Part 1:

- Create a file `/etc/motd` and put in a quote using Puppet

Part 2 (use the type reference):

- Create a file `test` in `/tmp/puppet/files`
  - We expect your first attempt to fail ;-)
- Change the owner of the file to your current user
- Make it read-only

Part 3:

- Reorganize your site.pp file into folder structure `puppet/manifests`
- Rerun the script

© 2014 Cambridge Technology Partners

Puppet Constructs

© 2014 Cambridge Technology Partners

Nodes

- Represents a host or host type

```
$ subl manifests/site.pp

node default {

    file { ‘/tmp/hello.txt’:
        content => ‘Hello World’
    }

}

$ cat /etc/hostname
```

- Can be inherited

```
node ‘xy’ inherits ‘z’ { ... }
```

- Name match with regular expressions

```
node /^(foo|bar)\.com$/ { ... }
```

© 2014 Cambridge Technology Partners

Lab 02

Part 1:

- Change the nodename to match your host
- Verify Puppet runs your changes
  - Use the --debug flag to see what's going on

Part 2:

- Change the node name back to `default`
- Create another node with your hostname which inherits the default node
- Add an additional resource to verify inheritance

© 2014 Cambridge Technology Partners

Packages

- Take advantage of the OS package manager

```
$ subl manifests/site.pp

node ‘ubuntu’ {

    package { ‘nginx’:
        ensure => installed
#       ensure => ‘major.minor.bugfix’
#       ensure => latest
    }

    package { ‘apache2.2-common’:
        ensure => absent
    }

    package { ‘nodejs’:
        ensure   => installed,
        provider => ‘npm’
    }

}
```

- Reusability?
- Scalability?

Other than the platform default package manager

© 2014 Cambridge Technology Partners

Classes

- Group of related resources

```
$ mkdir -p modules/nginx/manifest
$ subl modules/nginx/manifests/init.pp

class nginx {

    package { ‘nginx’:
        ensure => installed
    }

}

class { ‘nginx’: }

$ sudo puppet apply modules/nginx/manifests/init.pp
```

Modules

- Group of related classes / definitions

```
$ subl modules/nginx/manifests/init.pp
~~class { ‘nginx’: }~~
$ subl manifests/site.pp
node ‘ubuntu’ {
    include nginx
}
$ sudo puppet apply --modulepath=./modules manifests/site.pp
```

Autoload Format

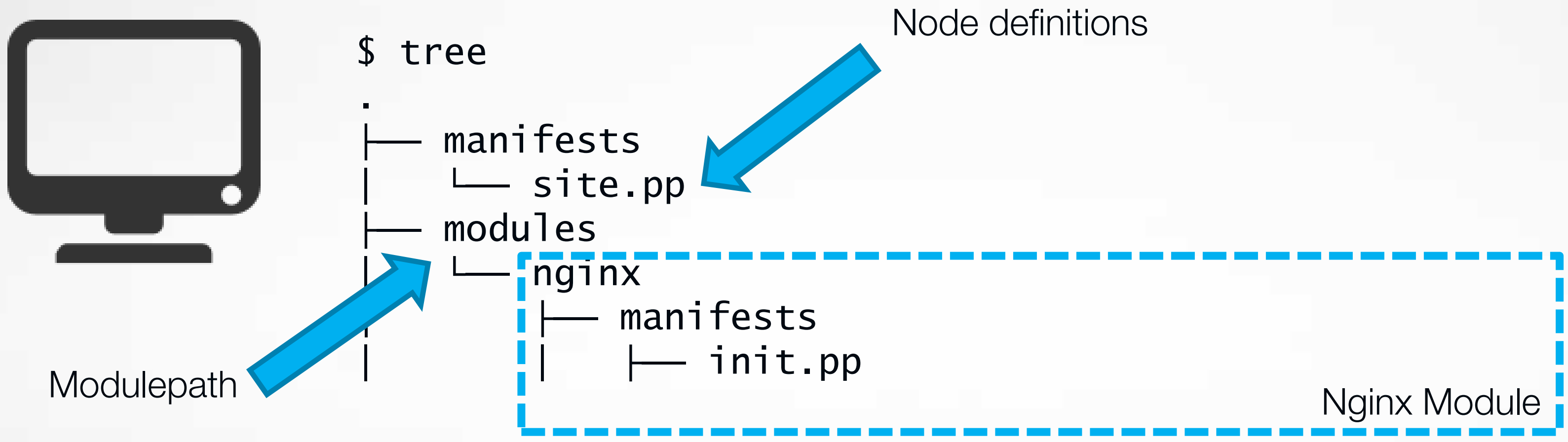
- Modulepath: Where are my modules?

```
$ sudo puppet config print modulepath
```

© 2014 Cambridge Technology Partners

Site Layout Overview

- Node definition in separate site.pp
  - Execute this manifest
- Module encapsulating installation details
  - Make sure it's part of the module path



```
$ tree
.
├── manifests
|   └── site.pp
├── modules
|   └── nginx
|       ├── manifests
|       |   ├── init.pp
```

© 2014 Cambridge Technology Partners

Lab 03

Part 1:

- Create the nginx module and main class
- Add the module to your node
- Make sure things still run as before

Part 2:

- Change the Puppet basemodulepath to find the module without using the command line param
- Find the Puppet config file and check what is in there

- You have now all the major Puppet building blocks!
  - Resources, Classes, Modules, Nodes

© 2014 Cambridge Technology Partners

Services

- Ensure the Nginx server is running

```
$ subl modules/nginx/manifests/init.pp

class nginx {

    package { ‘nginx’:
        ensure  => installed
    }

    service { ‘nginx’:
        ensure  => running,
        require => Package[‘nginx’]
    }

}
```

Resource / Class dependencies

Dependencies can also be defined by:

- chaining resources with `->` `~>`
- declaring dependencies on collections:
  `Yumrepo<| |> -> Package<| provider == yum |>`

Executions

- Run an arbitrary command

```
exec {
  'set-licence-selected':
    command => 'debconf-set-selections oracle-license accept true',
    path    => ['/bin','/usr/bin'], user    => ‘java’,
    unless  => 'debconf-get-selections | ...';

  'set-licence-seen':
    command => 'debconf-set-selections oracle-license seen true',
    ...
    creates => ‘/path/to/file’;
}
```

Insufficient checks on executions can make Puppet lose idempotence, or simply getting slow

© 2014 Cambridge Technology Partners

Lab 04

Part 1:

- Add the nginx service to your manifests
  - Ensure it’s running and started at boot time

Part 2:

- Refactor service and package into a separate class

Part 3:

- Run the script with the `--graph` option to see resource dependencies
  - Output folder is defined in the `graphdir` setting
  - Create a png with `dot -Tpng resource.dot > resource.png`

© 2014 Cambridge Technology Partners

Configuration

- Configure a site for nginx

```
$ mkdir -p modules/nginx/files
$ subl modules/nginx/files/cat-pictures.conf

server {
    listen 80;
    root /var/www/cat-pictures;
    server_name cat-pictures.com;
}

$ subl modules/nginx/manifests/init.pp

class nginx {
    ...
    file { '/etc/nginx/sites-enabled/default':
        source => 'puppet:///modules/nginx/cat-pictures.conf',
        notify => Service['nginx'],
    }
}
```

Reusability?

Puppet relative path (no `files` in path)

Interested in changes

© 2014 Cambridge Technology Partners

Lab 05

- Add the config and make sure all required paths exist
- Also add a simple index.html to the www root
- Access the server under http://localhost

Templating

- Reuse config files with templating

```
$ mv modules/nginx/files/cat-pictures.conf …/templates/vhost.conf.erb
$ subl modules/nginx/templates/vhost.conf.erb

server {
    listen 80;
    root /var/www/<%= @site_name %>;
    server_name /<%= @site_server %>;
}

$ subl modules/nginx/manifests/init.pp

class nginx {

    $site_name = ‘cat-pictures’
    $site_server = ‘cat-pictures.com’

    file { '/etc/nginx/sites-enabled/default':
        content => template('nginx/vhost.conf.erb‘),
        notify => Service['nginx'],
    }
}
```

Reusability?

© 2014 Cambridge Technology Partners

Definitions

- Reusable set of resources

Autoload format

Parameters
Default values for optional parameters

Namespaces

```
$ subl modules/nginx/manifests/vhost.pp

define nginx::vhost($port, $site_name = $title) {

    $site_server = “${site_name}.com”

    file { “/etc/nginx/sites-enabled/${site_name}”:
        content => template('nginx/vhost.conf.erb‘),
        notify  => Service['nginx'],
    }
}

$ subl modules/nginx/manifests/init.pp

class nginx {
    ...

    nginx::vhost { ‘cat-pictures’:
        port => 80
    }

}
```

© 2014 Cambridge Technology Partners

Lab 06

Part 1:

- Change the configuration to a template
- Optional: Try using an `inline_template` without a file

Part 2:

- Apply the configuration in a definition
- Make sure the setup is working as expected

Expressions and Facts

- After all it’s a script

```
$ subl modules/nginx/manifests/vhost.pp

define nginx::vhost($site_name = $title, $port) {

    if ($port == undef or $port == 0) {
        fail(‘Port cannot be empty or 0!’)
    }
    ...
```

- Use facts in classes, definitions or nodes

```
$ subl modules/nginx/manifests/init.pp

...
if ($::operatingsystem == ‘Windows’) {
    notify { ‘Seriously?!’: }
} else { ... }

$ facter | grep operating
```

Global Facts

Conditionals, Comparisons, …

- After all it’s Ruby…

```
case $::operatingsystem {
    ‘Debian’, ‘Ubuntu’: {
        ...
    }
    ‘Windows’: ...
    default: ...
}

unless $::operatingsystem in ‘Windows’ {
    ...
}

if $::hostname =~ /app.*staging/ {
   ...
}

$ec2_family = $::ec2_instance_type ? {
    /t1/ => 'micro',
    ...
}
```

© 2014 Cambridge Technology Partners

Lab 07

Part 1:

- Parameterize the nginx class with an optional version
- Use ‘latest’ if not set

Part 2:

- Install the nginx package from the Ubuntu PPA if running on Ubuntu (http://wiki.nginx.org/Install)
  - Lastet version is 1.6.0-1+trusty0
  - ppa:nginx/stable

Housekeeping

- Clean unnecessary files with `tidy`

```
tidy { '/var/log/nginx':
    age     => ‘4w',
    size    => ‘100m’,
    recurse => true,
    matches => [ “*.log”, “*.tmp* ]
}
```

Lab 08

- Clean out any existing files in the www directory before writing the actual files in there

Verifying Puppet

© 2014 Cambridge Technology Partners

Testing Puppet Syntax

- Basic syntax checks with the `puppet parser` command

```
$ puppet parser validate --noop /path/to/manifest.pp

for file in $(find . -iname '*.pp'); do
    puppet parser validate --render-as s $file || exit 1;
done;
```

- Embeddable into
  - Jenkins / CI build
  - Commit hooks (e.g. Git)

Style Checks

- Avoid common mistakes, align formatting with `puppet-lint`

```
$ find . -iname *.pp -exec                                            \
      puppet-lint --log-format                                        \
      "%{path}:%{linenumber}:%{check}:%{KIND}:%{message}" {}  \;
```

- Detailed usage instructions under http://puppet-lint.com/
- Follows style guide found under https://docs.puppetlabs.com/guides/style_guide.html

Unit Tests

- Rspec-puppet and Cucumber-puppet
  - BDD-style unit testing frameworks

```
$ puppet module generate chopen-rspectest
$ subl chopen-rspectest/spec/classes/init_spec.rb

require 'spec_helper'

describe 'rspectest' do
    context 'with defaults for all parameters' do
        it { should contain_class('rspectest') }
    end
end
```

- Rspec: http://rspec-puppet.com/
- Cucumber: https://projects.puppetlabs.com/projects/cucumber-puppet/wiki
- Articles on Puppet testing
  - http://www.jedi.be/blog/2011/12/05/puppet-unit-testing-like-a-pro/
  - http://puppetlabs.com/blog/the-next-generation-of-puppet-module-testing

© 2014 Cambridge Technology Partners

Lab 09

Part 1:

- Run **`puppet-lint`** on your module
- Fix the errors and violations
  - Exclude the check for 80 chars per line

Part 2:

- Create a simple Rspec test
  - Copy template files from a temporary module
  - Check for a running nginx service
  - Run **`rake help`**

```
$ sudo apt-get install rake
$ sudo gem install rspec-puppet puppetlabs_spec_helper
…
$ rspec-puppet-init
$ echo “require 'puppetlabs_spec_helper/module_spec_helper'” >
spec/spec_helper.rb
```

Cambridge
Technology Partners

Reinventing Wheels – NOT!

© 2014 Cambridge Technology Partners

Extending our Website

- DB to store cat pictures

```
$ puppet module generate ctp-mysql
$ subl ctp-mysql/manifests/init.pp

class mysql {

    package { ‘mysql-server’:
        ensure => installed,


    # Package -> File ~> Service pattern


    … # umm, feels like someone should have done that already…?!

}
```

© 2014 Cambridge Technology Partners

Puppet Forge

- Share and find some general usage Puppet modules:

  https://forge.puppetlabs.com/

```
$ puppet module install puppetlabs-mysql
```

Lab 10

Part 1:

- Install the puppetlabs-mysql module
- Where is it installed to?
  - Make sure it is part of your modulepath

Part 2:

- Add a MySQL server to your nginx installation
- Add a DB and a user with access righs
  - Test with the mysql client

Module Metadata

- Our Nginx module has now a dependency to MySQL
  - We need to document it
  - Or even better, ship the dependency declaration with the module

```
$ subl modules/nginx/metadata.json

{
  "name": “chopen-nginx",
  "version": "0.1.0",
  "author": “chopen",
  "summary": “Installs Nginx and the cat pictures website.",
  "dependencies": [
    {
      "name": "puppetlabs-mysql",
      "version_range": ">= 2.3.1"
    }
  ]
}
```

© 2014 Cambridge Technology Partners

Dependency Management

- No manual puppet module install
  - Endless and error prone
  - Transitive dependencies
- Librarian to the rescue: http://librarian-puppet.com/

```
$ subl Puppetfile

forge https://forgeapi.puppetlabs.com

def local(name)
    mod “chopen/#{name}”, :path => “./modules/#{name}”
end

local ‘nginx’

$ librarian-puppet install --path .librarian-modules
$ sudo puppet apply manifests/site.pp --modulepath=.librarian-modules
```

Version Control and Librarian

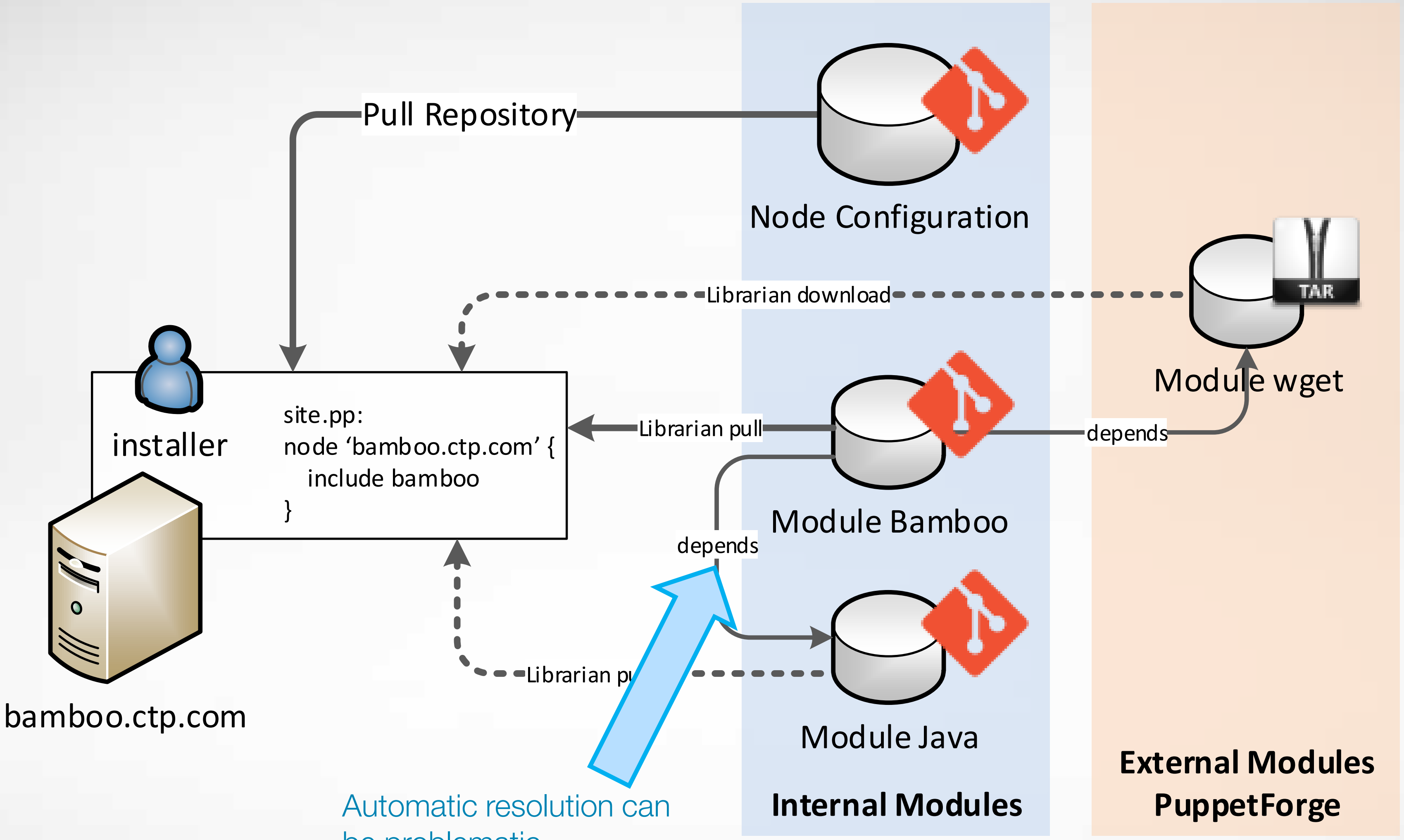
- One primary repository
  - Contains site manifest with node definitions
  - Plus a Puppetfile for dependency management

- Modules outsourced to dedicated module repositories

```
$ subl Puppetfile

def git(name, version)
    mod “chopen/#{name}”,
        :git => “git@server.com:#{name}.git,
        :ref => “#{version}”
end

git ‘nginx’, ‘0.1.0’
```

© 2014 Cambridge Technology Partners

Version Control and Librarian



```
site.pp:
node 'bamboo.ctp.com' {
   include bamboo
}
```

Automatic resolution can be problematic…

© 2014 Cambridge Technology Partners

Lab 11

Part 1:

- Create a metadata.json file and fill in the dependency
  - You can get a skeleton file by running `puppet module generate`
- Create the Puppetfile and add a dependency to your Nginx module

Part 2:

- Add a Git dependency to the NTP module and add NTP to the node definition
  - Found at https://github.com/puppetlabs/puppetlabs-ntp

```
class { '::ntp':
    servers => ['0.ch.pool.ntp.org', ... ],
}
```

Environment Configuration

© 2014 Cambridge Technology Partners

Environment Specifics

- MySQL user password
  - Different in production
  - Should preferrably not show up in site.pp
- Parameterized class
  - Pattern: inherit param class where defaults are defined (e.g. OS-specific)

```
$ subl modules/nginx/manifests/init.pp

class nginx(

    $mysql_user = $nginx::params::mysql_user,
    $mysql_password = undef

) inherits nginx::params {

    ...

}
```

© 2014 Cambridge Technology Partners

Hiera

- Hiera = Hierarchical Database
  - Integrated with Puppet
  - Can have various backends, from file-based to DBs to REST endpoints
  - Externalized: Can be managed separately

```
$ mkdir -p hiera/node
$ subl hiera/hiera.yaml

---
:backends:
  - yaml
:yaml:
  :datadir: hiera
:hierarchy:
  - "node/%{::hostname}"

$ sudo puppet config print hiera_config
$ subl hiera/node/ubuntu.yaml

---
nginx::mysql_password: test

$ hiera -c hiera/hiera.yaml nginx::mysql_password ::hostname=ubuntu
```

Lab 12

Part 1:

- Create a hiera.yaml file with a yaml backend
- Add a hierarchy containing your node settings
- Configure Puppet to use the new Hiera config

Part 2:

- Optional: Use a MySQL backend
  - Found at https://github.com/Telmo/hiera-mysql-backend

© 2014 Cambridge Technology Partners

Advanced Topics

© 2014 Cambridge Technology Partners

Config Files Beyond Templates: Augeas

- Config Files might be part of a package
  - Changing the template with each new version? Nope.
  - String replacement in an XML file? Nope.
  - Augeas adds advanced editing capabilities

```
<Server port="8080" schema="http"> …

augeas { "tomcat-settings":
    lens     => "Xml.lns",
    incl     => $tomcat::configfile,
    changes  => [
        "set Server/#attribute/port ${tomcat::port}",
        "set Server/#attribute/schema https",
        "set Server/#attribute/proxyName server.chopen.ch",
        "set Server/#attribute/proxyPort 443",
    ],
    require  => File["${tomcat::configfile}"],
}
```

© 2014 Cambridge Technology Partners

Custom Facts

- Ship reusable facts with a module
  - Requires some custom Ruby code though

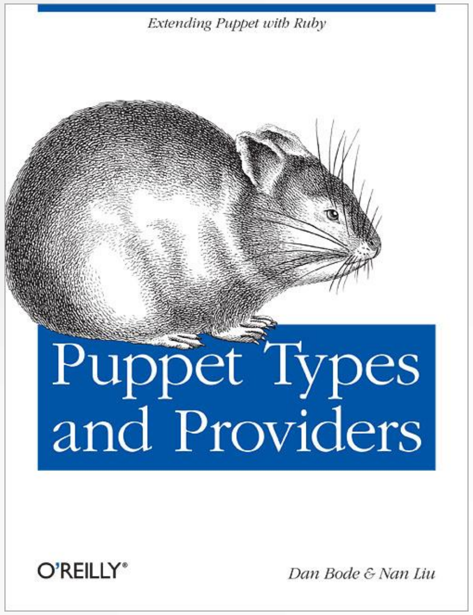
Autoload format

```
$ mkdir modules/nginx/lib/facter
$ subl modules/nginx/lib/facter/user_home.rb

# A facter fact to determine the user home directory.
Facter.add("user_home") do
    setcode do
        Facter::Util::Resolution.exec("echo $HOME")
    end
end
```

© 2014 Cambridge Technology Partners

Custom Resources

- Come back to next year's workshop ☺
  - Out of scope, but good to know you can
  - O'Reilly book reference, or GitHub

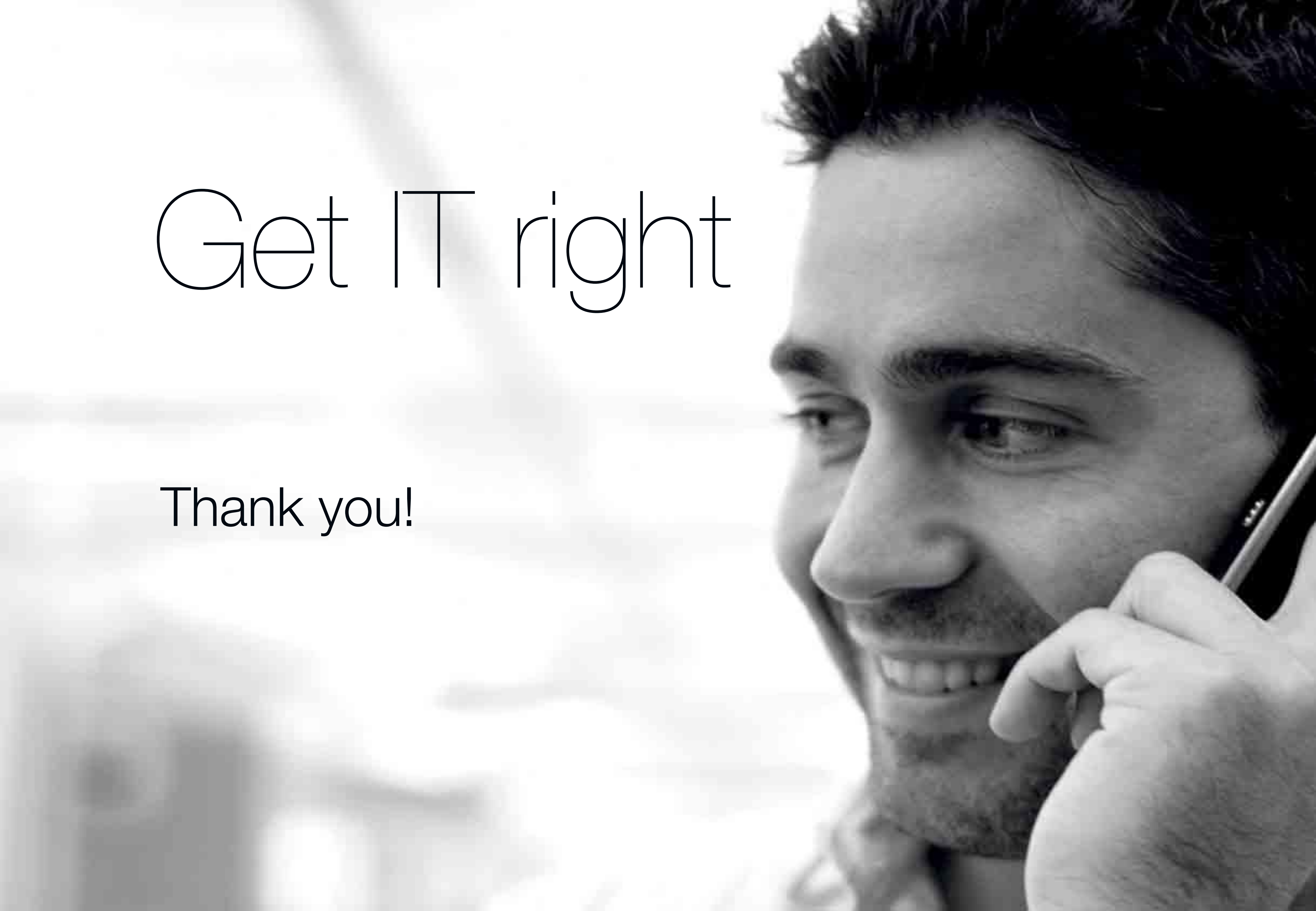


Run Stages

- When resource dependencies get out of control…
  - Assign classes to custom stages (default is «main»)
  - Stages freely orderable
  - Use carefully!

```
class stages {

    stage { 'first': before => Stage['main'] } # consider site.pp
    stage { 'last': require => Stage['main'] } # to define stages

    class me_first { notify { 'In first!': } }
    class me_last { notify { 'Out last...': } }

    class { 'me_first':
        stage => 'first',
    }
    class { 'me_last':
        stage => 'last',
    }

}
```

© 2014 Cambridge Technology Partners



Get IT right

Thank you!

Credits

- Icons provided by Icons8: http://icons8.com/

© 2014 Cambridge Technology Partners