

PowerShell Conference Europe 2019

Hannover, Germany

June 4-7, 2019

**JOHN O'CONNOR**

# Enhancing DSC with Puppet

Platinum  
Sponsor



5

Video operator, did you start the recording?

4

3

2

1

PowerShell Conference Europe 2019

Hannover, Germany

June 4-7, 2019

# Enhancing DSC with Puppet

**JOHN O'CONNOR**

Platinum  
Sponsor







# How would you order a Pizza?





# What the MENU Offers

Cheese & Tomato

Mozzarella cheese (tomato, cheese)

Vegetarian (tomato, onions, pepper, black olives)

Pepperoni (tomato, cheese and Spanish sausage)



# The Instructions for the Menu

## INSTRUCTIONS

I Want a Vegetarian Pizza



## EXECUTION

Put a thin layer of sauce on the dough.

Sprinkle some mozzarella cheese

Add a handful of onions

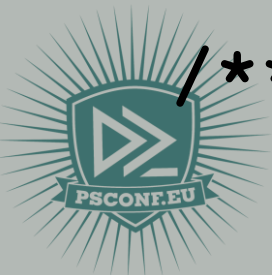
Add a handful of peppers

Add a few black olives



# Procedural scripts for a pizza order

```
/****** Begin Procedure Script *****/  
  
BEGIN  
  
GOTO cupboard in the stockroom.  
GOTO third shelf from the bottom on left hand side  
facing the back.  
  
can = Take the can of tomato sauce  
  
open $can  
  
sauce = pour ½ $can onto the pizza dough  
  
END  
  
/****** End Procedure Script *****/
```



# Instead – why not say what we want

```
pizza { "John's order":  
    sauce => tomato,  
    cheese => mozzarella,  
    vegetables => [onions,peppers,black olives],  
    meat => pepperoni  
}
```

Easy for a person to write and understandable for a computer



# Consistency/Traceability across Organisation

Top Level Organisational declarations instead of obscure hard to manage procedures

Ability to manage diverse set of distributed services

Regulated environment needing traceability and auditing

The ever present *"Need to do More with Less"*



# This Session

What Puppet and DSC have in common

The Puppet value add to DSC

Case Study: Configure Windows Services Update Server (WSUS)





# DSC and Puppet - Similarities

Declaritive DSL (Domain Specific Language)

Used to enforce configuration.

Syntax and models are similar



# DSC .vs. Puppet DSC Lite

# DSC Example

```
File fruit_file {  
    Ensure = 'Present'  
    DestinationPath = 'C:\Fruit.txt'  
    Type = 'File'  
    Contents = 'Apple, Pear'  
}
```

# Puppet DSC Lite Declaration

```
dsc {'fruit_file':  
    resource name => 'File',  
    module => 'PSDesiredStateConfiguration',  
    properties => {  
        ensure => 'Present',  
        destinationpath => 'C:\Fruit.txt',  
        type => 'File',  
        content => 'Apple, Pear'  
    }  
}
```

# Puppet Value Add

Similar Syntax to DSC, so get up and running quickly

Manage large set of diverse nodes (Windows, Linux, MacOS, Solaris, AIX, Cisco)

Alternative to Active Directory/GPO for Windows

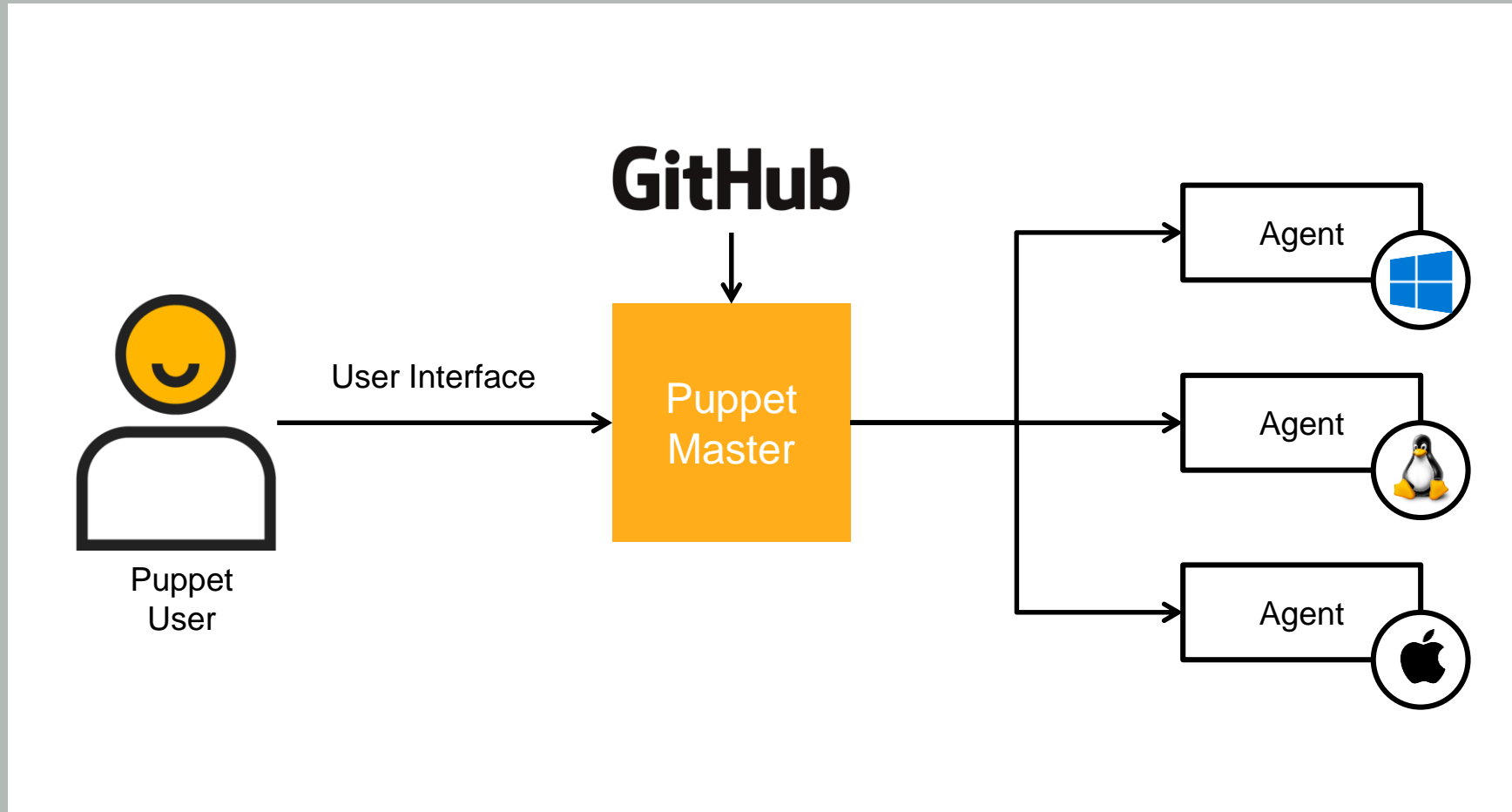
Puppet has large library of existing resources, modules to aid capturing and managing your configuration.

Management tools/Common interface across platforms

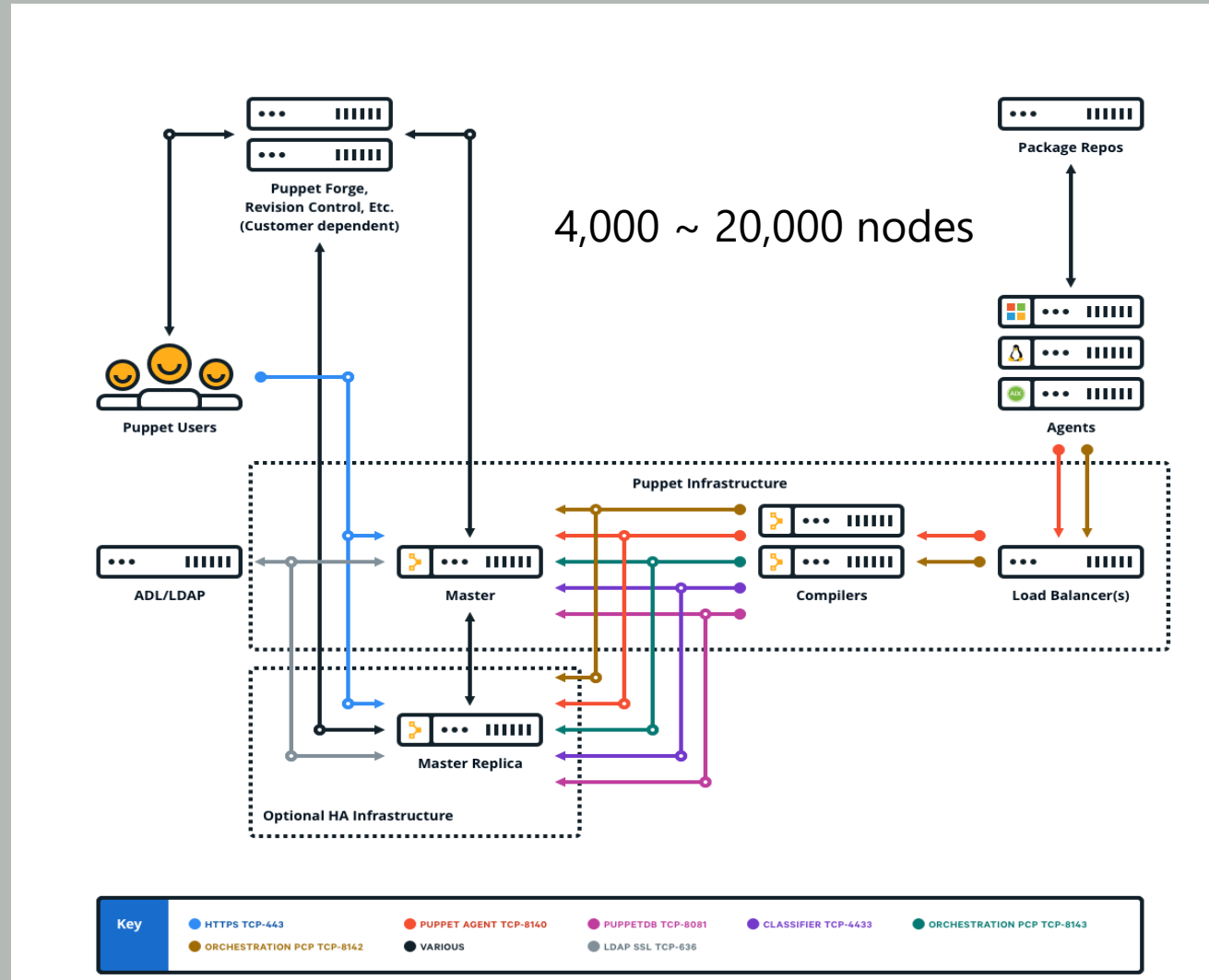
Change tracking and auditing



# Puppet Architecture outline



# Extensible for large distributed system







# Reporting with Puppet and DSC

Change Reporting is a Must

Changes classified as  
Failures  
Corrective Changes  
Intentional Changes  
Unchanged

## 1 [Nodes run in enforcement](#)

	0	<a href="#">with failures</a>
	0	<a href="#">with corrective changes</a>
	0	<a href="#">with intentional changes</a>
	1	<a href="#">unchanged</a>





# DEMO

Case Study – Configuring and monitoring change on Windows Services Update Server (WSUS)

# Tale of an unmanaged WSUS Server



# WSUS Console

Update Services

File Action View Window Help

Update Services

- win-wsus-18-vm
  - Updates
  - Computers
  - Downstream Servers
  - Synchronizations
  - Reports
  - Options

win-wsus-18-vm

Use this snap-in to quickly and reliably deploy the latest updates to your computers.

**To Do**

- ⚠ Your WSUS server currently shows that no computers are registered to receive updates.
- ℹ 326 new products and 11 new classifications have been added in the past 30 days. [View products and classifications](#)

**Overview**

Computer Status	Synchronization Status
<div>Computers with errors: 0</div> <div>Computers needing updates: 0</div> <div>Computers installed/not applicable: 0</div>	<div>Status: Idle</div> <div><a href="#">Synchronize Now</a></div> <div>Last synchronization: 5/31/2019 3:30 PM</div> <div>Last synchronization result: <a href="#">Succeeded</a></div>

Update Status	Download Status
<div>Updates with errors: 0</div> <div>Updates needed by computers: 0</div> <div>Updates installed/not applicable: 0</div>	<div>Updates needing files: 2</div> <div>Downloaded 30.48 MB of 30.48 MB</div>

Server Statistics	Connection
<div>Unapproved updates: 26</div> <div>Approved updates: 6566</div> <div>Declined updates: 140</div> <div>Computers: 0</div> <div>Computer groups: 0</div>	<div>Type: Local/SSL</div> <div>Port: 8530</div> <div>User role: Administrator</div> <div>Server version: 10.0.17763.134</div>

**Resources**

- WSUS Home Page
- WSUS Technical Overview
- Microsoft Update Catalog
- WSUS Community
- WSUS Privacy Statement

**Actions**

- win-wsus-18-vm
  - Search...
  - Remove from Console
  - Import Updates...
  - View
    - New Window from Here
  - Refresh
  - Help



# Manual WSUS Install/Configuration

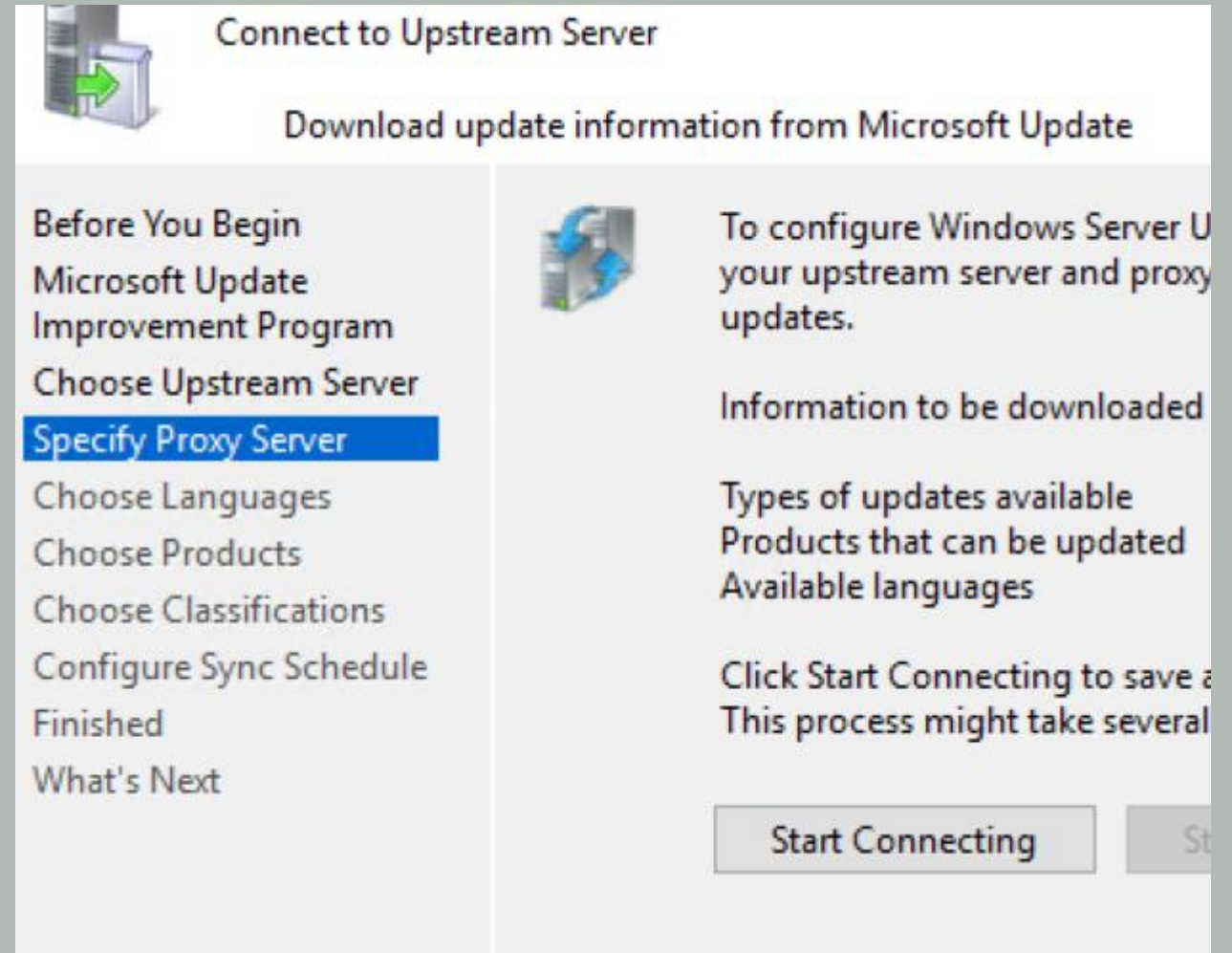
Enable Features

“UpdateServices-Feature”

“UpdateServicesRSAT-Feature”

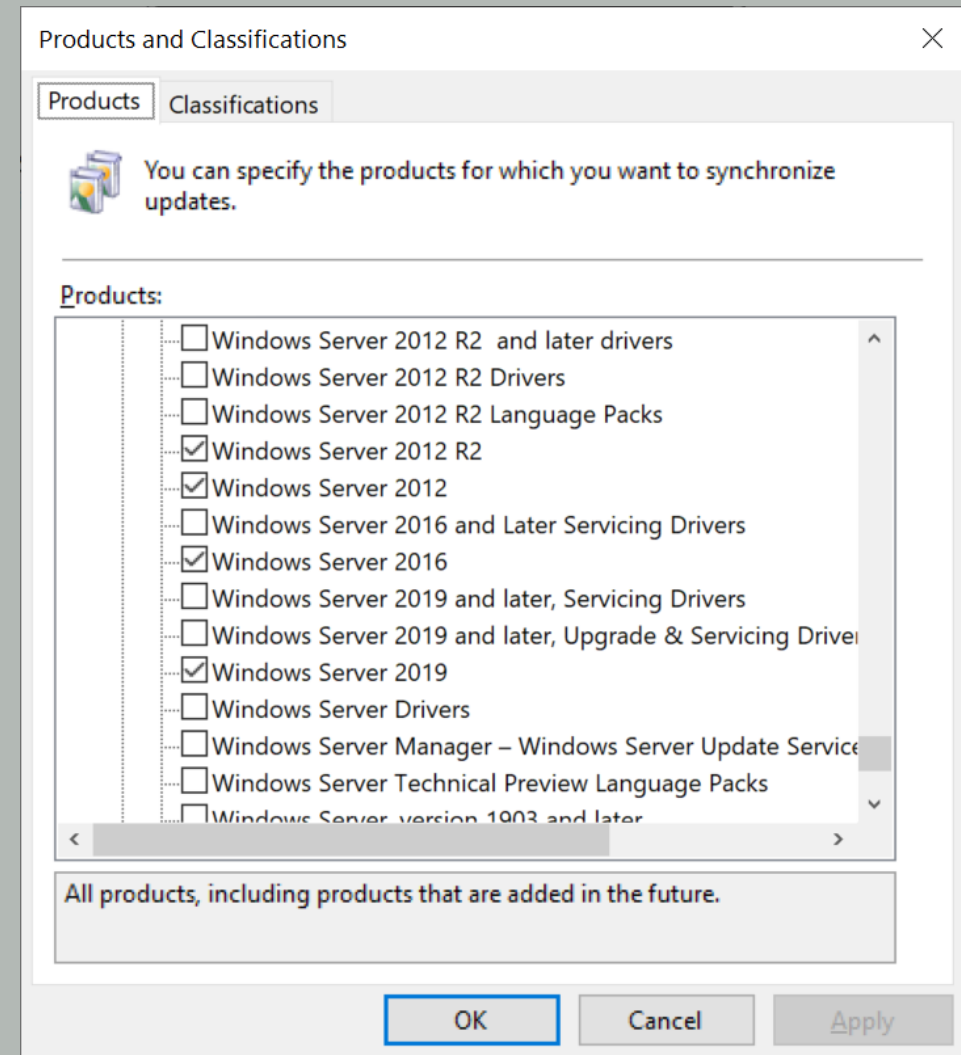
Run Configuration Wizard

Wait for WSUS to Synchronise



# WSUS Products – Code .vs. Gui

```
products => [  
    'Windows 10 LTSB',  
    'Windows 10',  
    'Windows 7',  
    'Windows 8.1',  
    'Windows 8',  
    'Windows Server 2008 R2',  
    'Windows Server 2008',  
    'Windows Server 2012 R2',  
    'Windows Server 2012',  
    'Windows Server 2016',  
    'Windows Server 2019'  
],
```



# Demo Repository & DSC Module

[github.com/jcoconnor/pscconfeu-dsc](https://github.com/jcoconnor/pscconfeu-dsc)

Main File of Interest: [site/profile/manifests/wsus/server/wsus\\_server.pp](https://github.com/jcoconnor/pscconfeu-dsc/blob/master/site/profile/manifests/wsus/server/wsus_server.pp)

*Repo also includes some other examples of Puppet code used to configure the Demo Server with some useful Packages.*

[github.com/mgreenegit/UpdateServicesDsc](https://github.com/mgreenegit/UpdateServicesDsc)

[www.powershellgallery.com/packages/UpdateServicesDsc](https://www.powershellgallery.com/packages/UpdateServicesDsc)





```
# The following GUIDS reference differing type of Updates
# for setting up WSUS and
# its associated approval rules.
```

```
$wsus_critical_updates      = 'E6CF1350-C01B-414D-A61F-263D14D133B4'
$wsus_definition_updates   = 'E0789628-CE08-4437-BE74-2495B842F43B'
$wsus_security_updates     = '0FA1201D-4330-4FA8-8AE9-B877473B6441'
$wsus_service_packs        = '68C5B0A3-D1A6-4553-AE49-01D3A7827828'
$wsus_update_rollups       = '28BC880E-0592-4CBF-8F95-C79B17911D5F'
```

```
Dscfix::Lcm_config { 'disableLCM':
    refresh_mode => 'Disabled'
}
```

```
dsc {'UpdateServices-Feature':
```

```
    resource_name => 'WindowsFeature',
```

```
    module        => 'PSDesiredStateConfiguration',
```

```
    properties => {
```

```
        ensure => 'present',
```

```
        name    => 'UpdateServices'
```

```
    },
```

```
    require => Dscfix::Lcm_config['disableLCM'],
```

```
}
```

```
dsc {'UpdateServices':
```

```
  resource_name => 'UpdateServicesServer',  
  module        => 'UpdateServicesDsc',
```

```
  properties => {  
    ensure      => 'present',  
    contentdir  => 'C:\WSUS',  
    languages   => ['en'],  
    synchronize => true,  
    products    => [ ... ],  
    classifications => [ ... ],  
    synchronizeautomatically => true,  
    synchronizeautomaticallytimeofday => '15:30:00',  
  },
```

```
  require => Dsc['UpdateServices-Feature',  
                 'UpdateServicesRSAT-Feature'],
```

```
}
```

```
# Product List for 'UpdateServices'
```

```
products => [  
    'Windows 10 LTSC',  
    'Windows 10',  
    'Windows 7',  
    'Windows 8.1',  
    'Windows 8',  
    'Windows Server 2008 R2',  
    'Windows Server 2008',  
    'Windows Server 2012 R2',  
    'Windows Server 2012',  
    'Windows Server 2016',  
    'Windows Server 2019'  
],
```

```
# Classification List for 'UpdateServices'  
# and 'CleanupRules'  
classifications => [  
    $wsus_critical_updates,  
    $wsus_definition_updates,  
    $wsus_security_updates,  
    $wsus_service_packs,  
    $wsus_update_rollups,  
],
```

```
dsc {'UpdateServicesCleanup':
```

```
  resource_name => 'UpdateServicesCleanup',  
  module       => 'UpdateServicesDsc',
```

```
  properties => {  
    ensure      => 'present',  
    declineexpiredupdates      => true,  
    declinesupersededupdates   => true,  
    cleanupobsoleteupdates     => true,  
    cleanupunneededcontentfiles => true,  
  },
```

```
  require => Dsc['UpdateServices'],
```

```
}
```



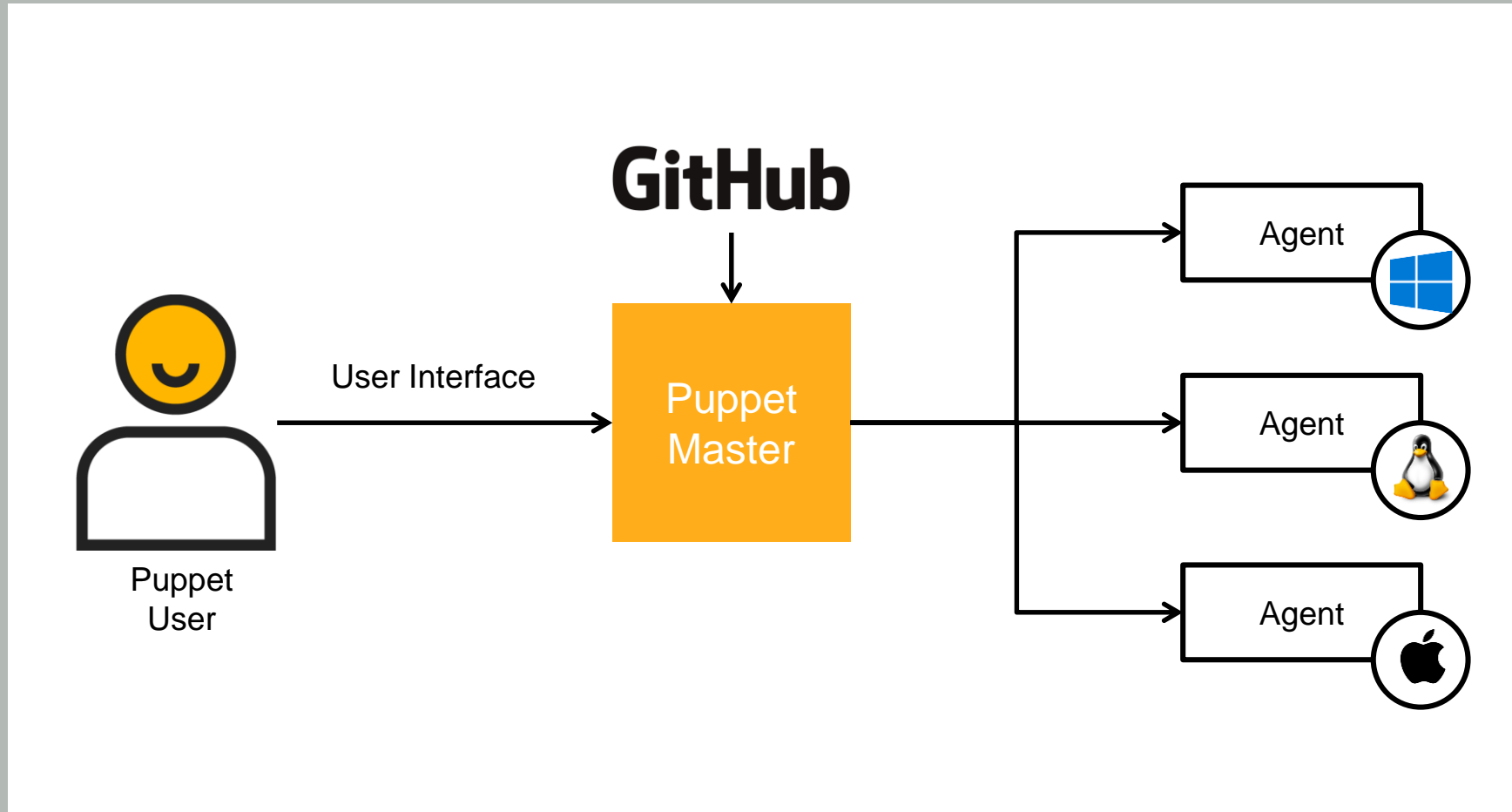
```
dsc { 'ApprovalRules':
```

```
    resource_name => 'UpdateServicesApprovalRule',  
    module        => 'UpdateServicesDsc',
```

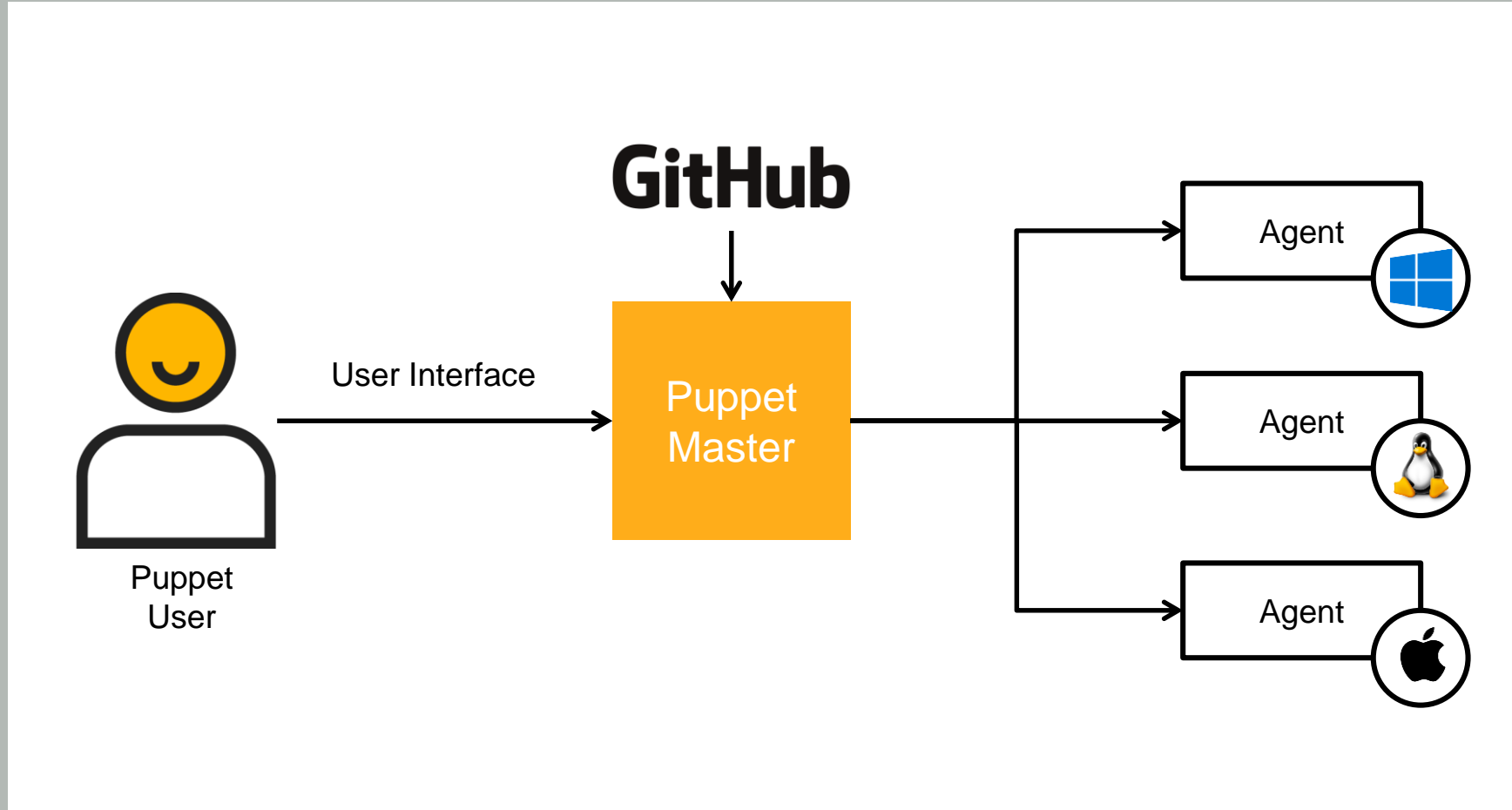
```
    properties => {  
        ensure      => 'present',  
        name        => 'Definition Updates, ... ..',  
        enabled     => true,  
        synchronize => true,  
        runrulesnow => true,  
        classifications => [ ... ],  
    },
```

```
    require => Dsc['UpdateServices'],  
}
```

# Demo (1) – Configuring WSUS on Server



# Demo (2) – Correcting/Reporting a change



# DSC and Puppet: best of both worlds

Doing more with less.

Infrastructure as code that all teams can use.

Apply your PowerShell DSC skills with Puppet.

Get going quickly with Puppet



# Questions?

Use the conference app to vote for this session:

<https://my.eventraft.com/psconfEU>

# Resources (1) – Demo and DSC Repos

[github.com/jcoconnor/pscconfeu-dsc](https://github.com/jcoconnor/pscconfeu-dsc)

Main File of Interest: [site/profile/manifests/wsus/server/wsus\\_server.pp](https://github.com/jcoconnor/pscconfeu-dsc/blob/master/site/profile/manifests/wsus/server/wsus_server.pp)

*Repo also includes some other examples of Puppet code used to configure the Demo Server with some useful Packages.*

[github.com/mgreenegit/UpdateServicesDsc](https://github.com/mgreenegit/UpdateServicesDsc)

[www.powershellgallery.com/packages/UpdateServicesDsc](https://www.powershellgallery.com/packages/UpdateServicesDsc)



# Resources (2) – Puppet DSC Modules



**puppetlabs/dsc\_lite** by: Puppet

PowerShell Desired State Configuration (DSC) Lite

[forge.puppet.com/puppetlabs/dsc\\_lite](https://forge.puppet.com/puppetlabs/dsc_lite)



**puppetlabs/dsc** by: Puppet

PowerShell Desired State Configuration (DSC)

[forge.puppet.com/puppetlabs/dsc](https://forge.puppet.com/puppetlabs/dsc)



# Resources (3) – Puppet/DSC Blogs

<https://puppet.com/blog/managing-powershell-dsc-puppet>

<https://puppet.com/blog/dsc-deep-dive-get-up-and-running-dsc-module>

<https://puppet.com/blog/dsc-deep-dive-installing-and-configuring-iis-using-puppet-and-powershell-dsc>

<https://puppet.com/blog/james-pogran-on-learning-powershell-dsc>

<https://puppet.com/blog/introducing-dsclite-new-dsc-module-puppet>





# Resources (4) - Puppet

[puppet.com/products](https://puppet.com/products)



# Acknowledgements

James Pogran for the original DSC and Puppet Talk

[www.youtube.com/watch?v=dR8VJjDmo9c](http://www.youtube.com/watch?v=dR8VJjDmo9c)  
[speakerdeck.com/jpogran/using-puppet-and-dsc-to-report-on-environment-change](http://speakerdeck.com/jpogran/using-puppet-and-dsc-to-report-on-environment-change)

Alberta Bosco for the pizza example

Photo by [Ben White](#) on [Unleash](#) for *"Tale of Unmanaged Server"*  
[unsplash.com/photos/qDY9ahp0Mto/info](https://unsplash.com/photos/qDY9ahp0Mto/info)



# Who am I

With Puppet for 4 years

- Release Engineering

- Puppet Agent and Platform Support

- Automation of Golden Images

- Windows Team

Previous – variety of System Admin, Software engineering roles across Education, Financial, Telecoms and Security sectors.

Classical Music and Opera enthusiast, Walking

Managing PA and AV for my local church

