Achieving Continuous Delivery with puppet

@D2SI

#PuppetCampParis

puppet camp

Paris 2014

Laurent Bernaille, D2SI

Why do we need continuous delivery ?

Past (current?) situation

Applications do not answer business needs well

- Long development cycle
- Difficult to get clear specifications
- Market can shift quickly

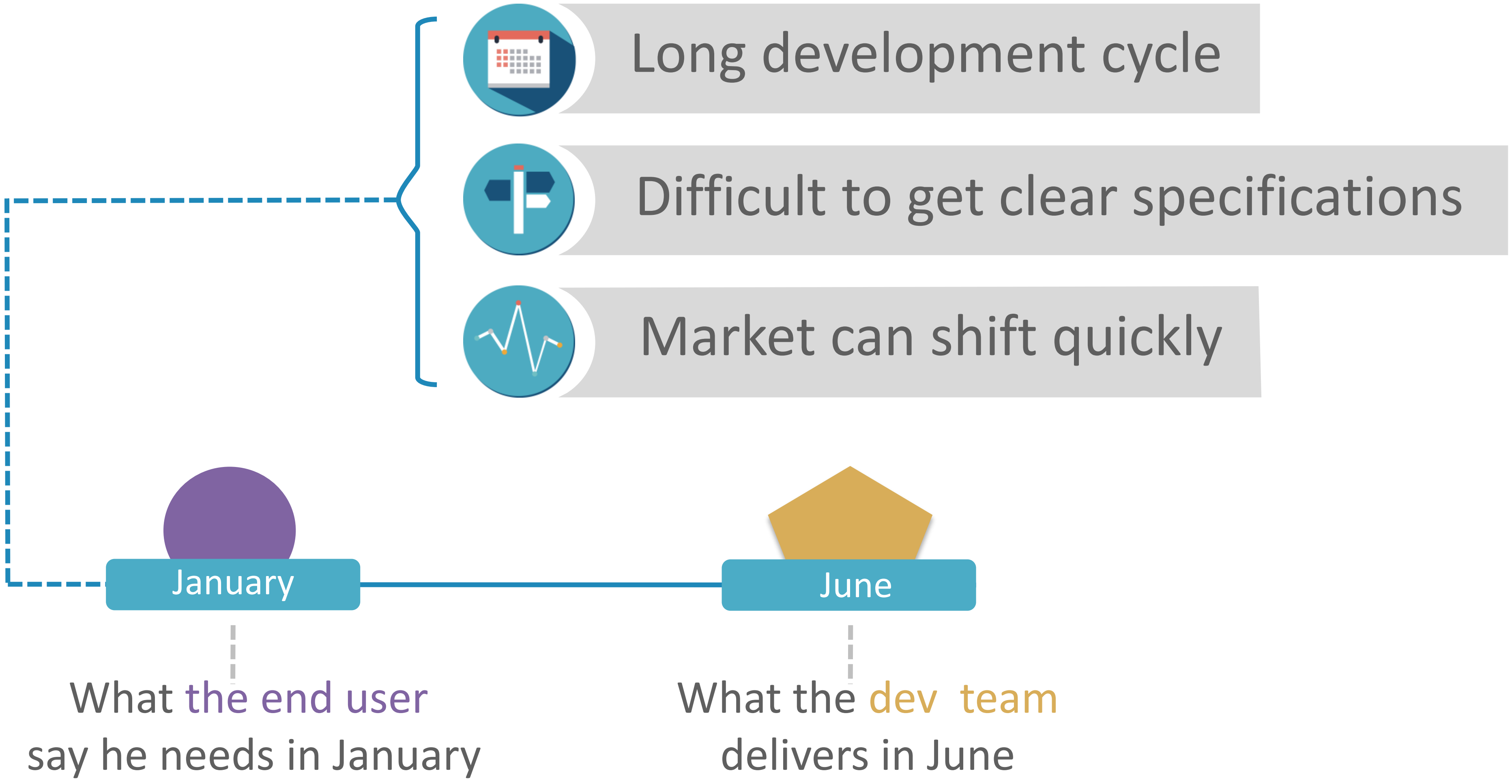
Applications do not answer business needs well

- Long development cycle
- Difficult to get clear specifications
- Market can shift quickly

January

What the end user
say he needs in January

Applications do not answer business needs well



Long development cycle

Difficult to get clear specifications

Market can shift quickly

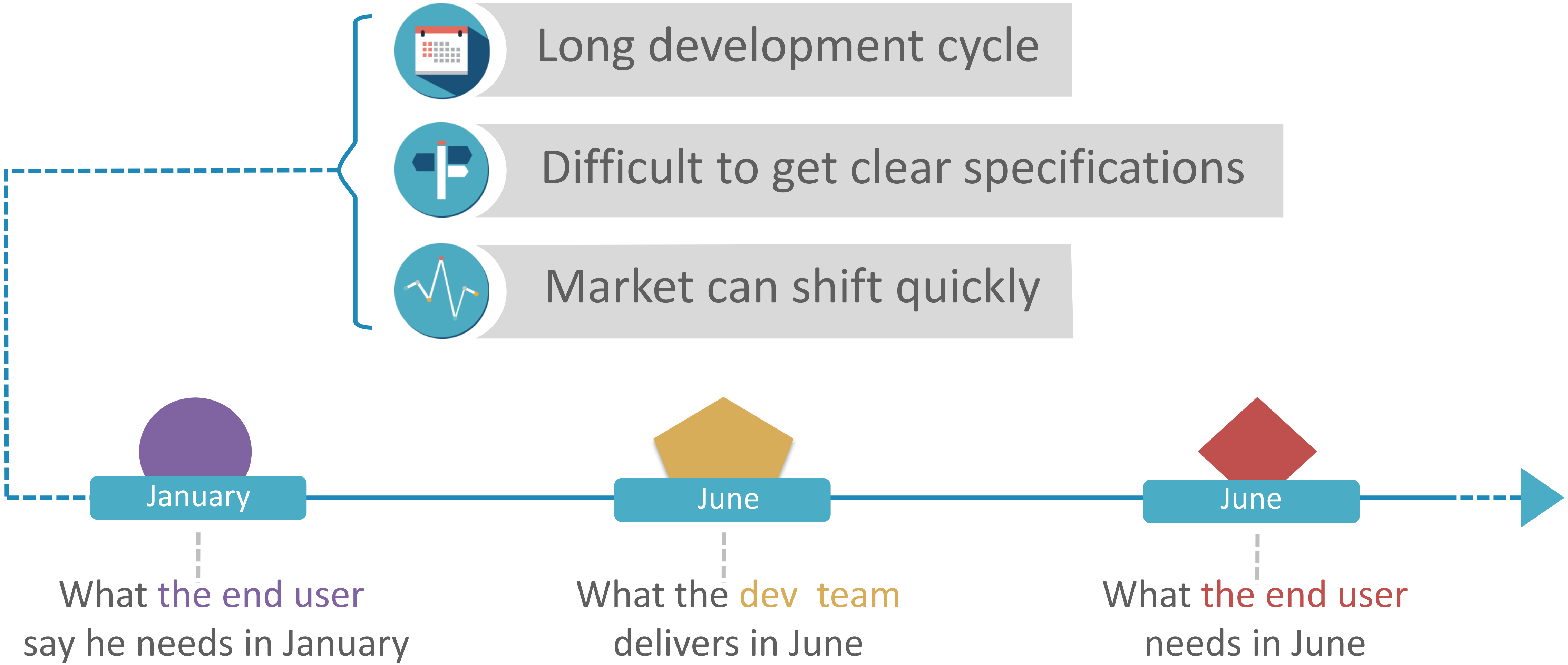
January

June

What the end user
say he needs in January

What the dev team
delivers in June

Applications do not answer business needs well



Long development cycle

Difficult to get clear specifications

Market can shift quickly

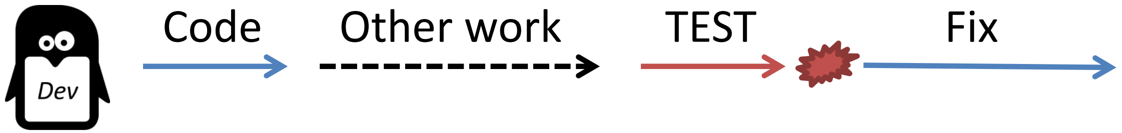
What the end user say he needs in January

What the dev team delivers in June

What the end user needs in June

Development processes are inefficient

Bug are detected too late



Dev

Code

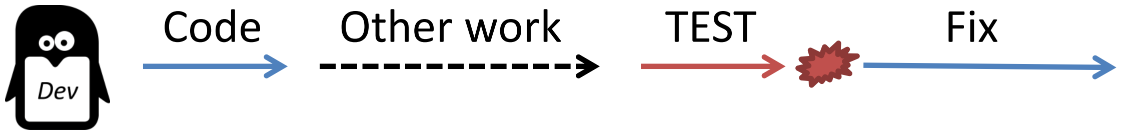
Other work

TEST

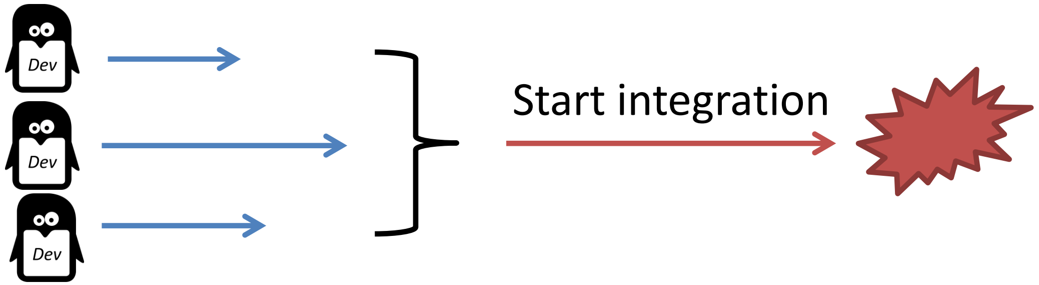
Fix

Development processes are inefficient

Bug are detected too late

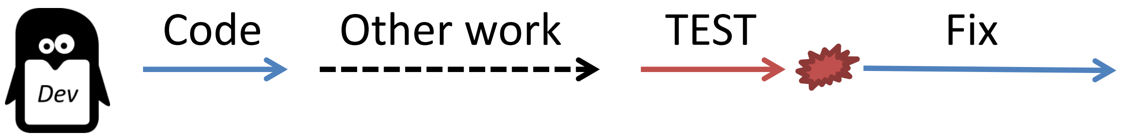


Integration Hell

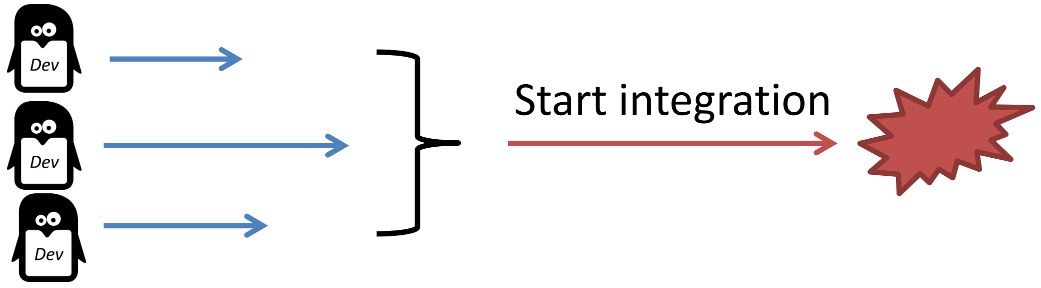


Development processes are inefficient

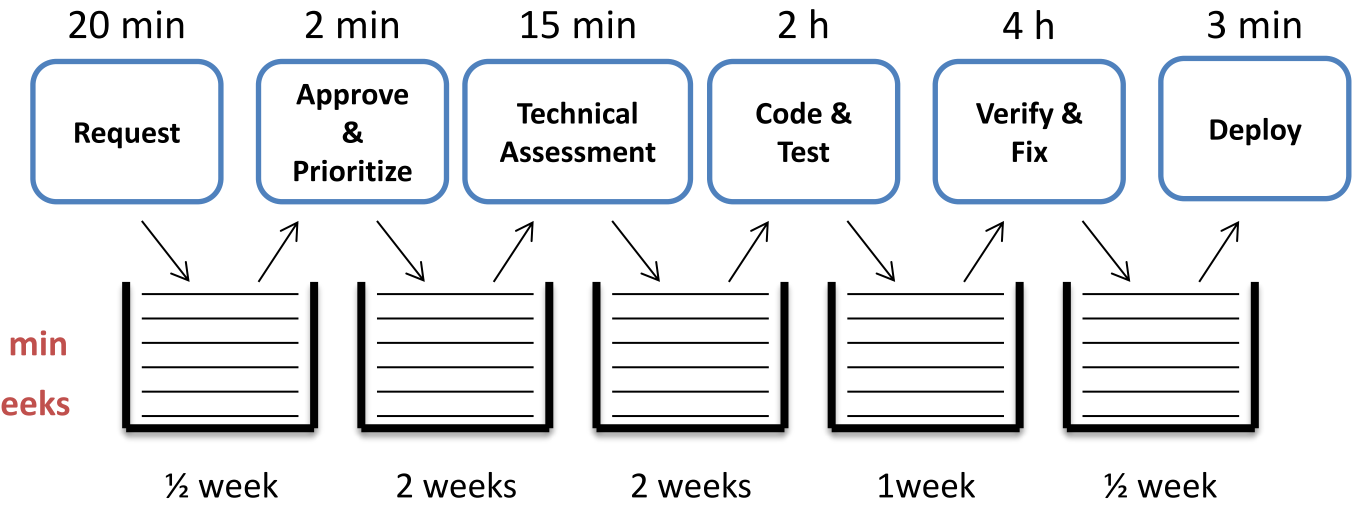
Bug are detected too late



Integration Hell



A lot of wasted time



Processing Time = 6 h 40 min

Waiting Time = 6 weeks

Adapted from *Implementing Lean Software Development: From Concept to Cash*, Mary & Tom Poppendieck.

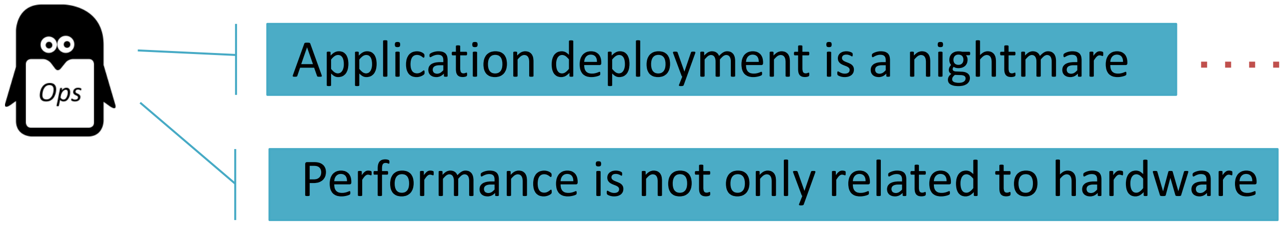
Relationship with OPS can be “difficult”

OPS view

Application deployment is a nightmare

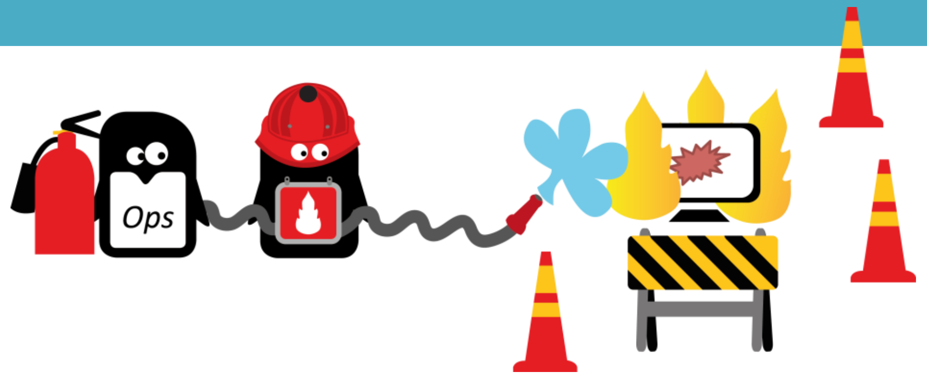
Relationship with OPS can be “difficult”

OPS view



Application deployment is a nightmare

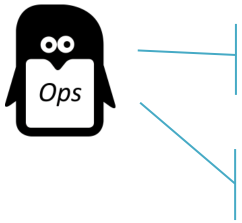
Performance is not only related to hardware



« Make my website faster in Asia»

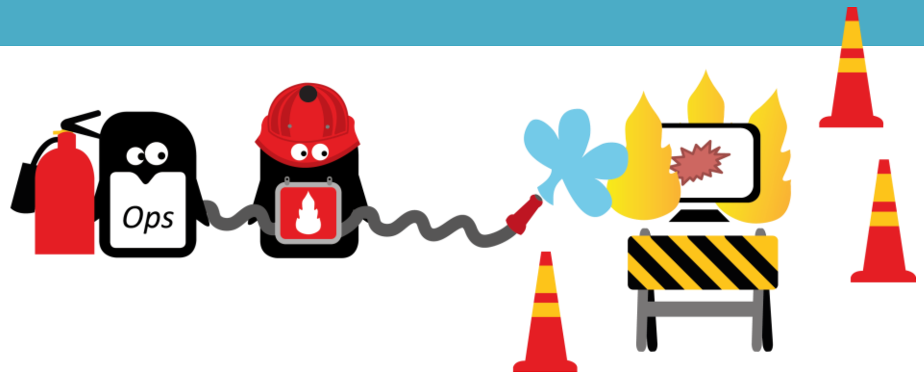
Relationship with OPS can be “difficult”

OPS view



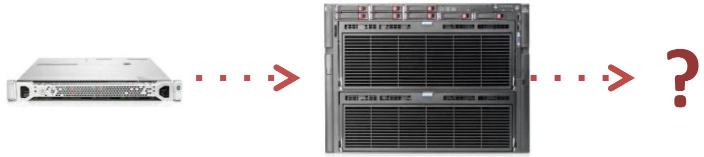
Application deployment is a nightmare

Performance is not only related to hardware



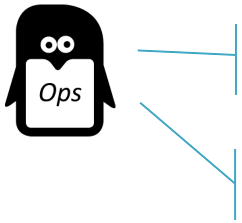
« Make my website faster in Asia»

« Our application is too slow because of your servers»



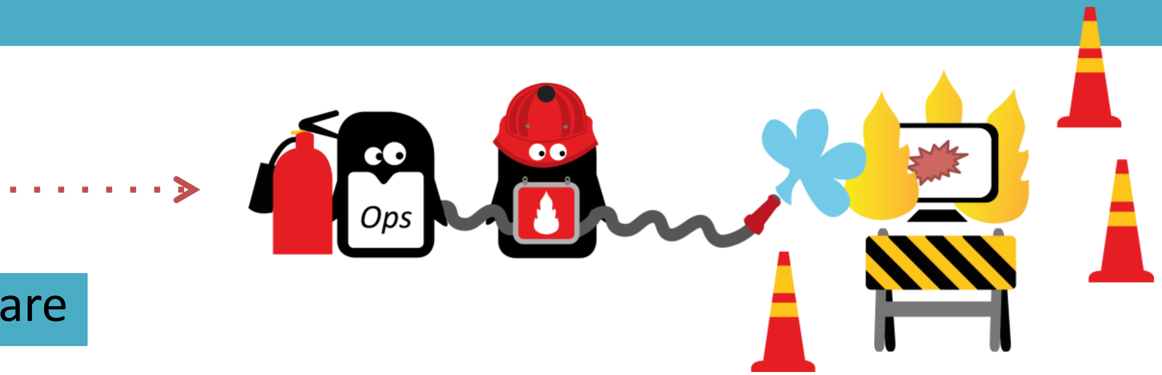
Relationship with OPS can be “difficult”

OPS view



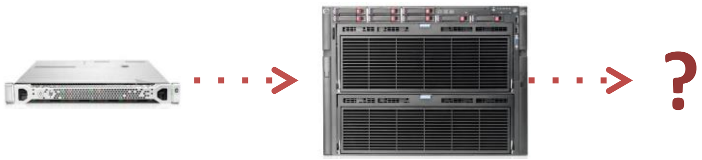
Application deployment is a nightmare

Performance is not only related to hardware



« Make my website faster in Asia»

« Our application is too slow because of your servers»



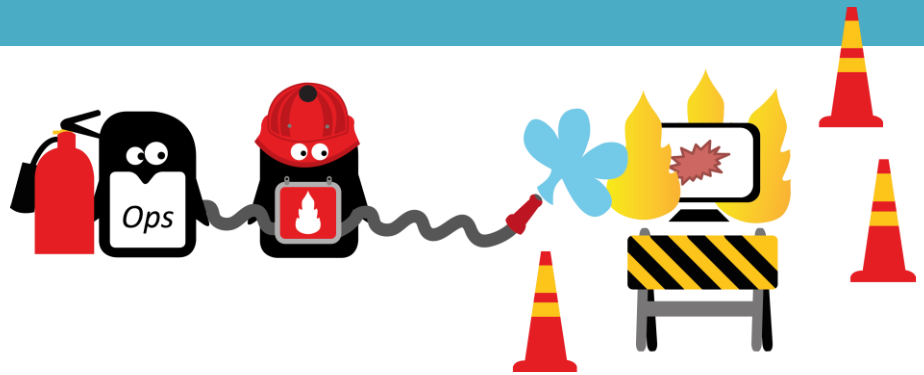
DEV view

Relationship with OPS can be “difficult”

OPS view

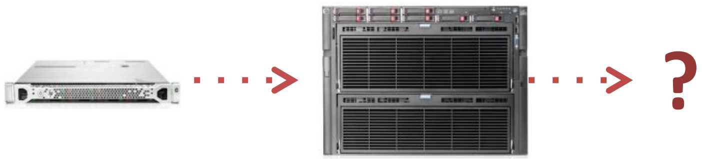
Application deployment is a nightmare

Performance is not only related to hardware



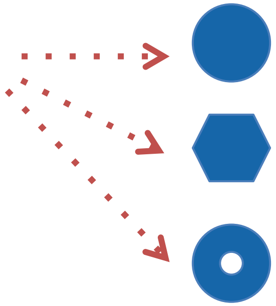
« Make my website faster in Asia»

« Our application is too slow because of your servers»



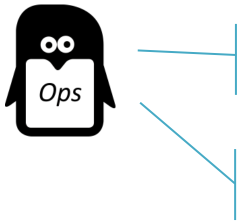
DEV view

Identical servers are always “slightly” different



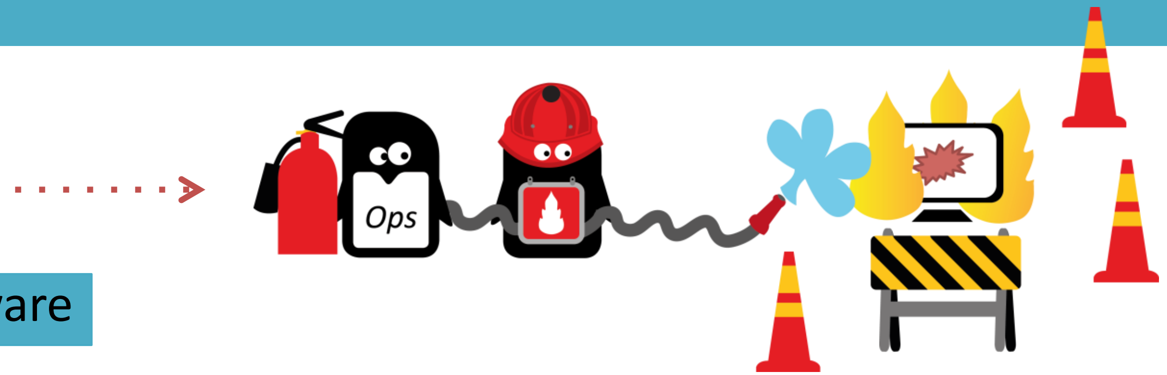
Relationship with OPS can be “difficult”

OPS view



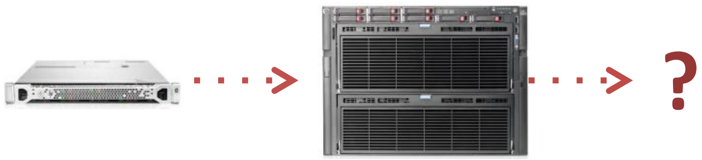
Application deployment is a nightmare

Performance is not only related to hardware

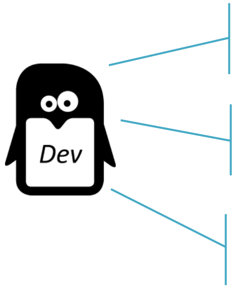


« Make my website faster in Asia»

« Our application is too slow because of your servers»



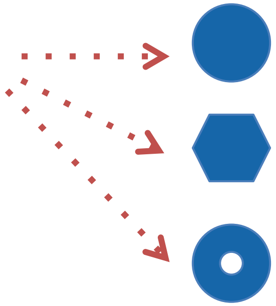
DEV view



Identical servers are always “slightly” different

Standards do not evolve

OPS always say “no”

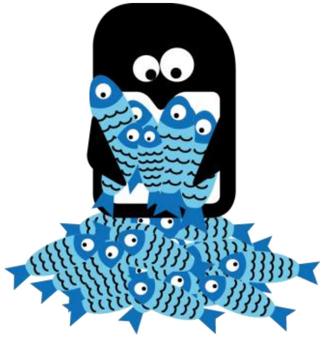


10 deploys per day, Dev & ops cooperation at Flickr
John Allspaw & Paul Hammond (Velocity 2009)

Infrastructure is not very agile

6+ months to setup a new environment

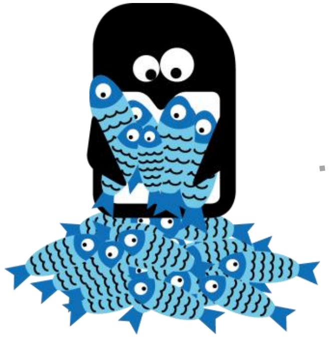
Server “hoarding”



Infrastructure is not very agile

6+ months to setup a new environment

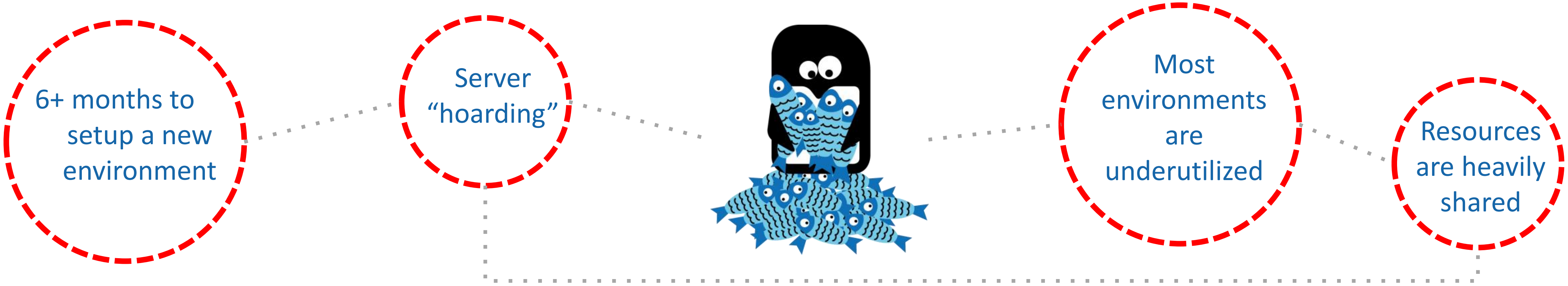
Server "hoarding"



Most environments are underutilized

Resources are heavily shared

Infrastructure is not very agile



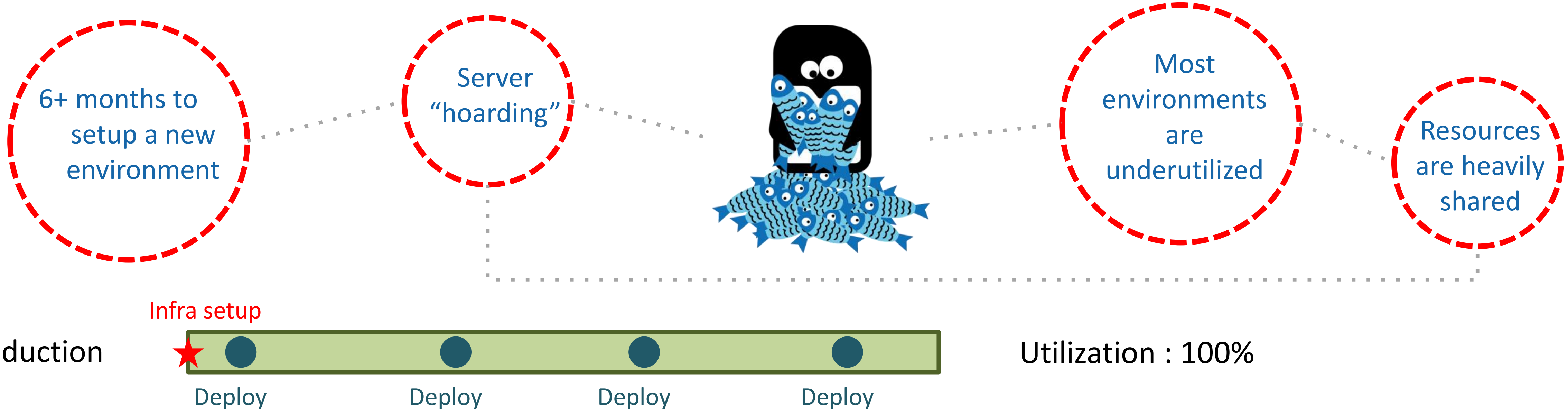
6+ months to setup a new environment

Server "hoarding"

Most environments are underutilized

Resources are heavily shared

Infrastructure is not very agile



6+ months to setup a new environment

Server "hoarding"

Most environments are underutilized

Resources are heavily shared

Infra setup

Production

Deploy

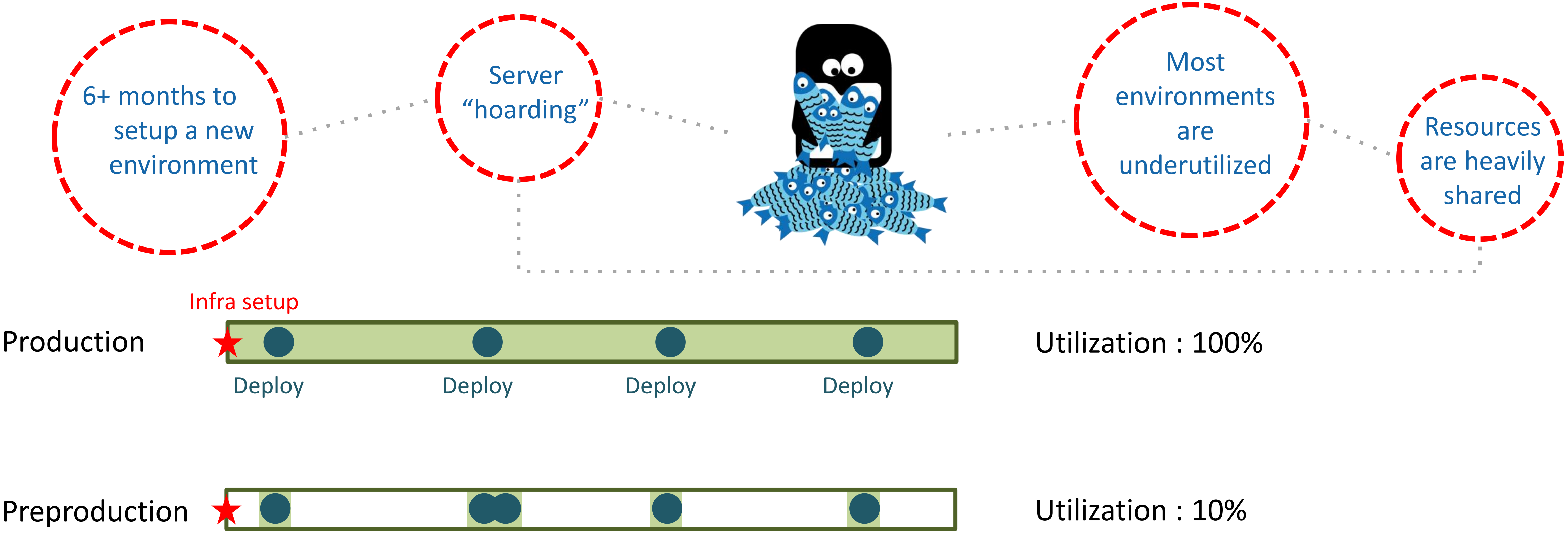
Deploy

Deploy

Deploy

Utilization : 100%

Infrastructure is not very agile



6+ months to setup a new environment

Server “hoarding”

Most environments are underutilized

Resources are heavily shared

Infra setup

Production ★ ● ● ● ●

Deploy Deploy Deploy Deploy

Utilization : 100%

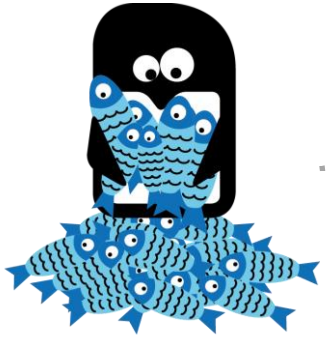
Preproduction ★ ● ●● ● ●

Utilization : 10%

Infrastructure is not very agile

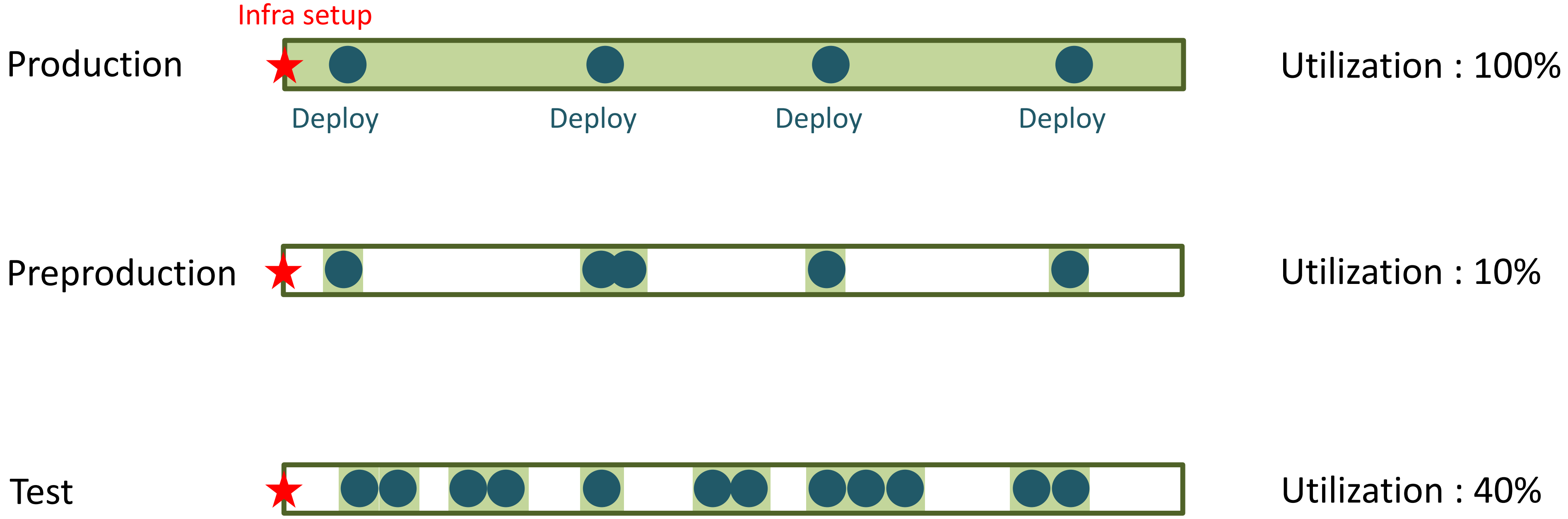
6+ months to setup a new environment

Server “hoarding”



Most environments are underutilized

Resources are heavily shared



Infra setup

Production

Deploy

Deploy

Deploy

Deploy

Utilization : 100%

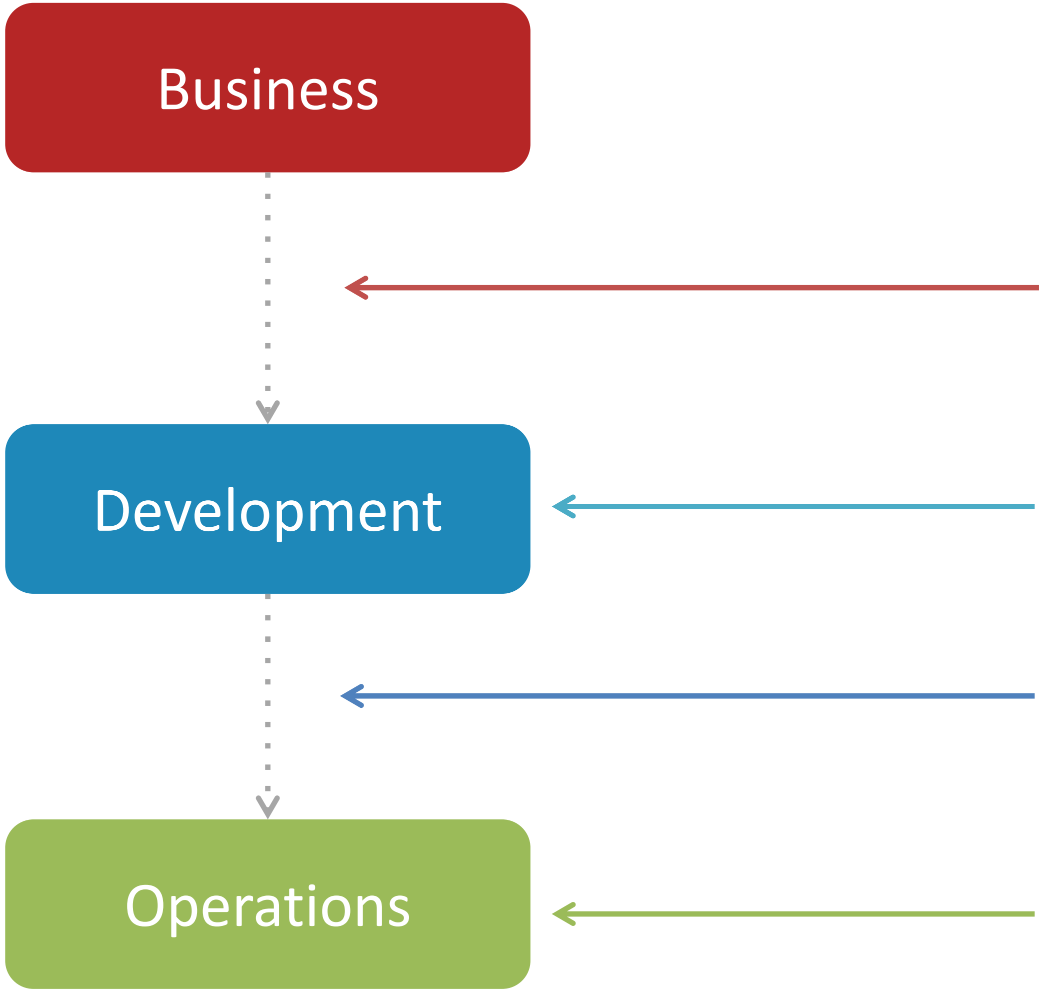
Preproduction

Utilization : 10%

Test

Utilization : 40%

Summary of the issues



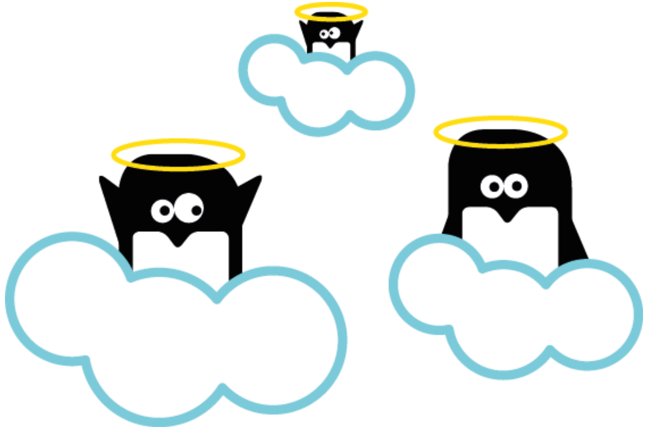
- Applications do not answer business needs well
- Too long to get new features

- Integration and bug fixing is painful
- A lot of wasted time

- Deployments are very painful
- A lot of misunderstanding

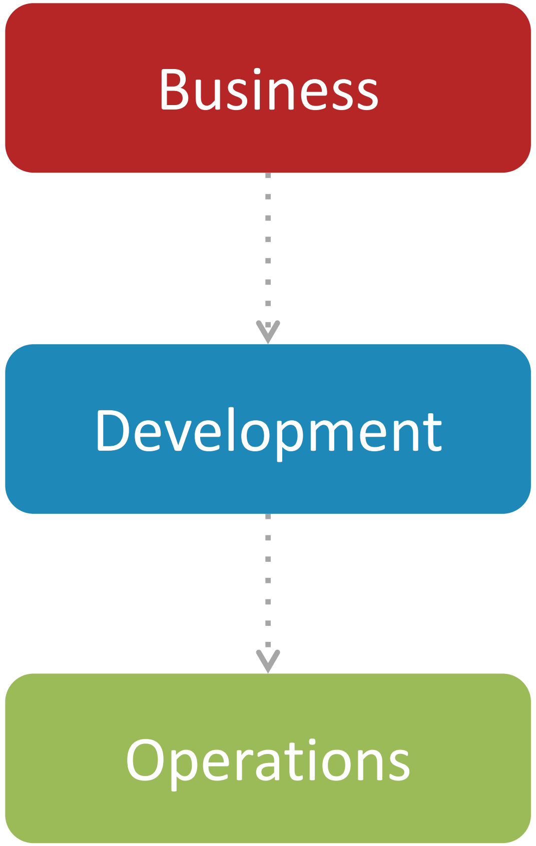
- Environment setup is too slow
- No on-demand resources

Continuous Delivery



IT should be easier

Agile Development


Business

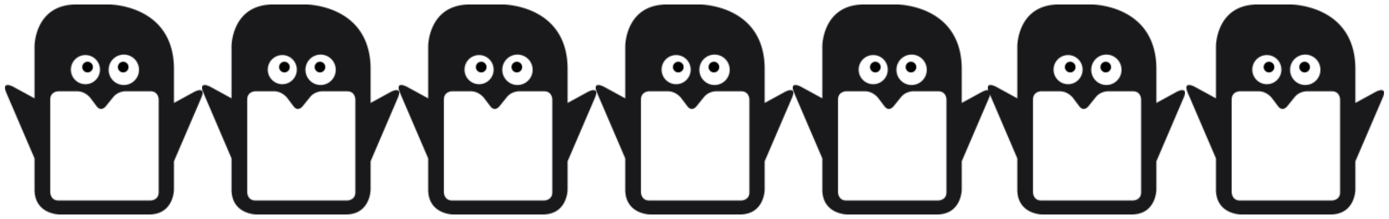
Development

Operations

Agile Manifesto, 2001

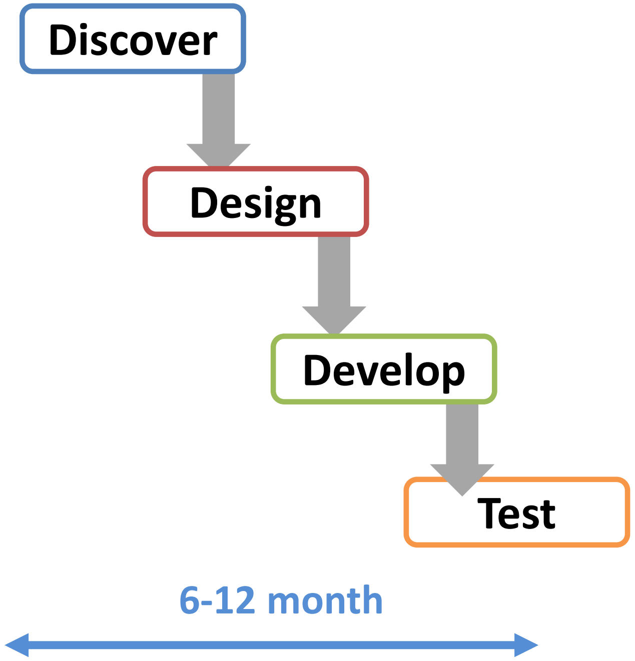
AGILE

Individuals and interactions *over processes and tools*
Working software *over comprehensive documentation*
Customer collaboration *over contract negotiation*
Responding to change *over following a plan*



Developing Incrementally and Iteratively

Waterfall method



Discover

Design

Develop

Test

6-12 month

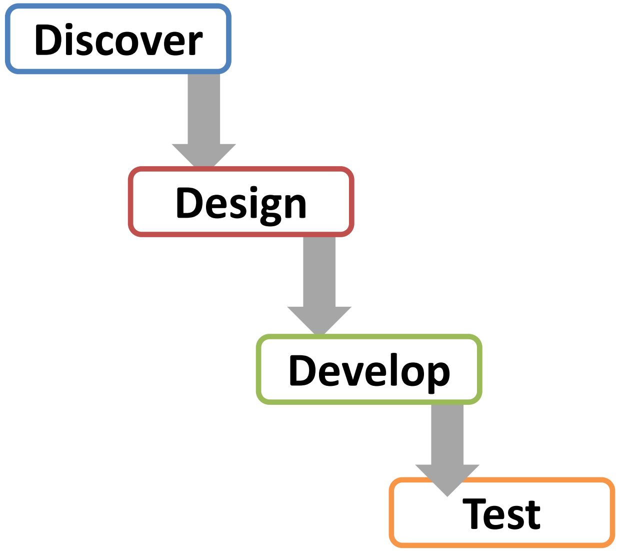
Long cycles

High-risk

Adaptation is very difficult

Developing Incrementally and Iteratively

Waterfall method



6-12 month

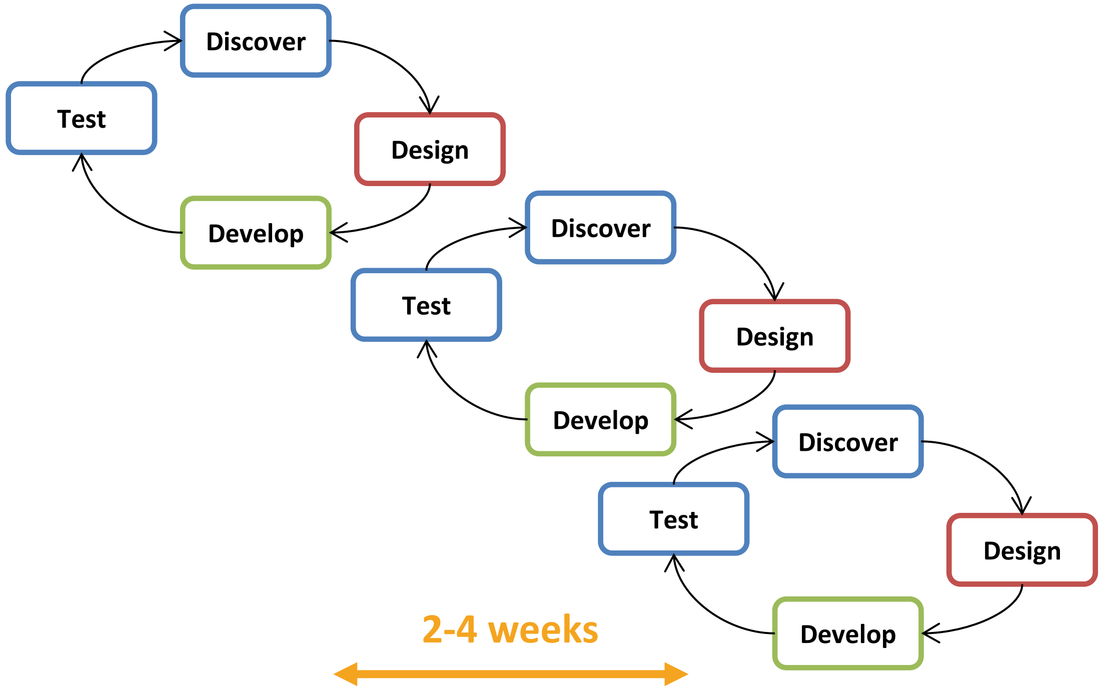
Long cycles

High-risk

Adaptation is very difficult

VS

Agile method



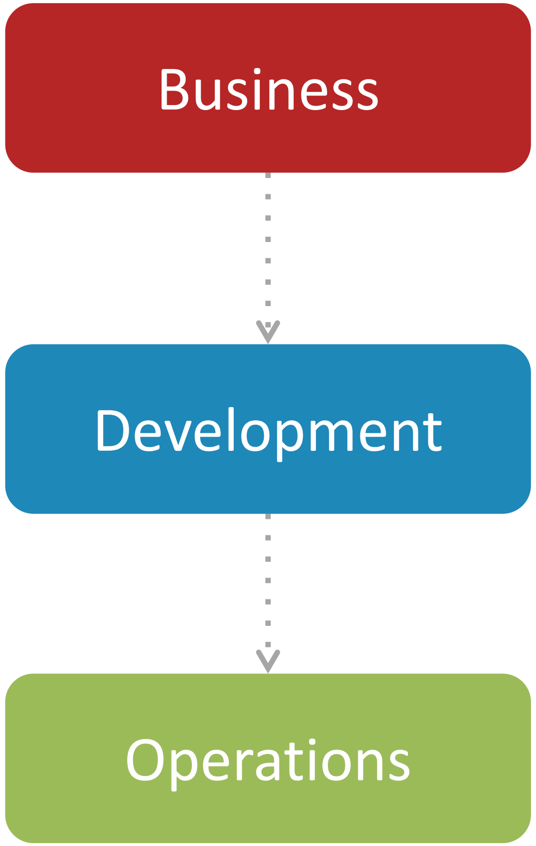
2-4 weeks

Many short cycles

Low-risk

Adaptation is much easier

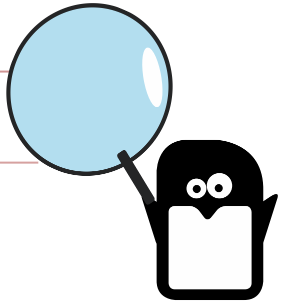
Continuous Integration



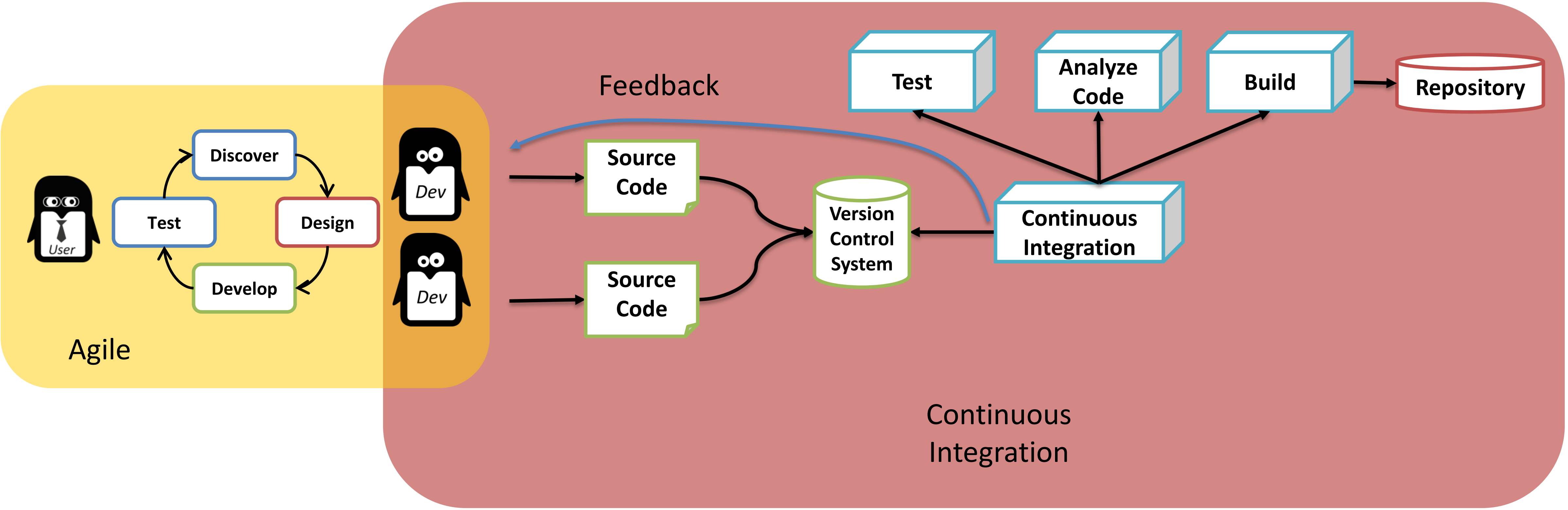
CI

Detect problems early and solve them quickly

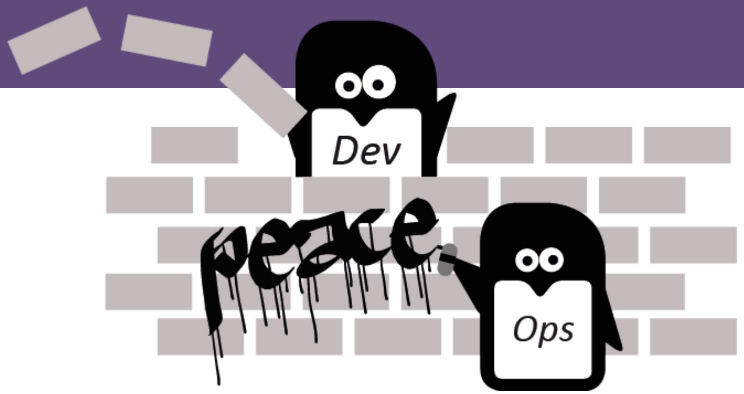
- Maintain a single source repository
- Automate the build
- Make your build self-testing
- Every commit should build on an integration machine
- Keep the build fast
- Test in a clone of the production environment
- Make it easy for anyone to get the latest executable
- Everyone can see what’s happening

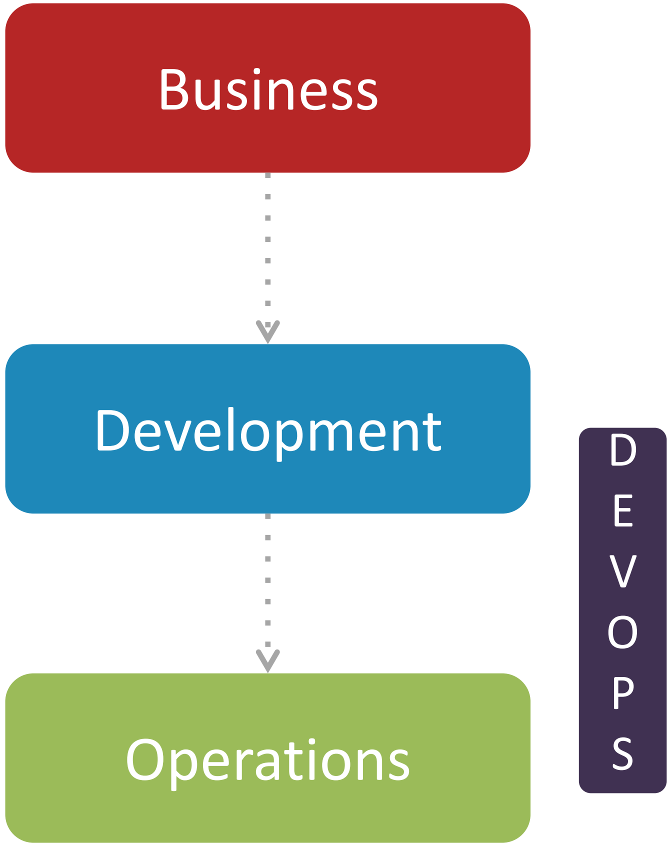


Continuous Integration



DevOps : bring the wall down





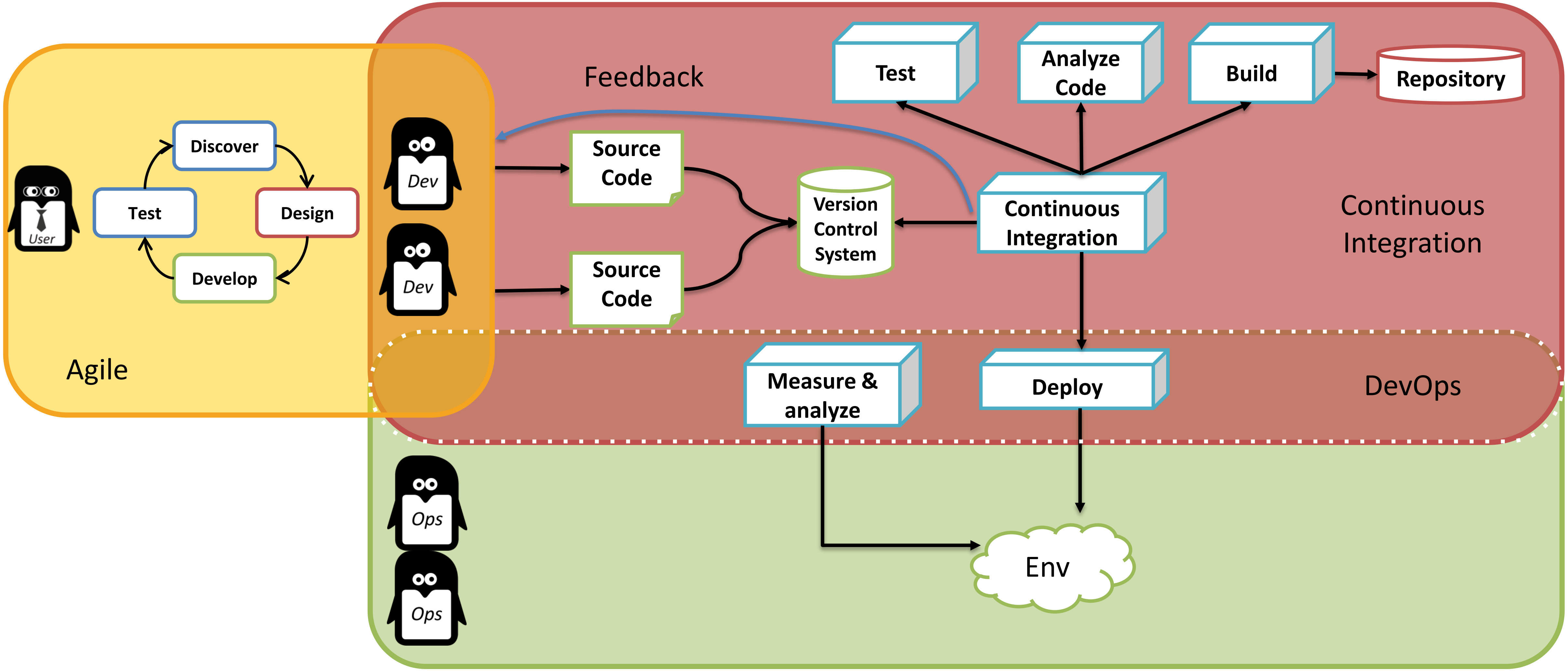
Culture .... Align objectives on business needs
Innovate

Automation .... Automate application delivery

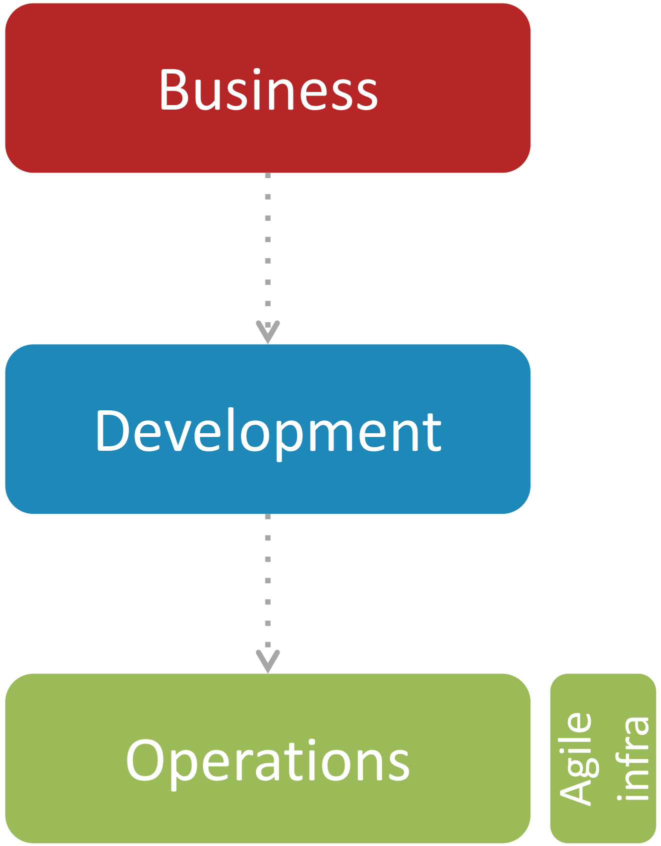
Measure .... Measure, Analyze & Describe

Share .... Constraints (from DEV and OPS)
Best practices, methods

Continuous Delivery



Agile Infrastructure



Deploy applications

Provide a service to deploy applications
Automated, with rollbacks

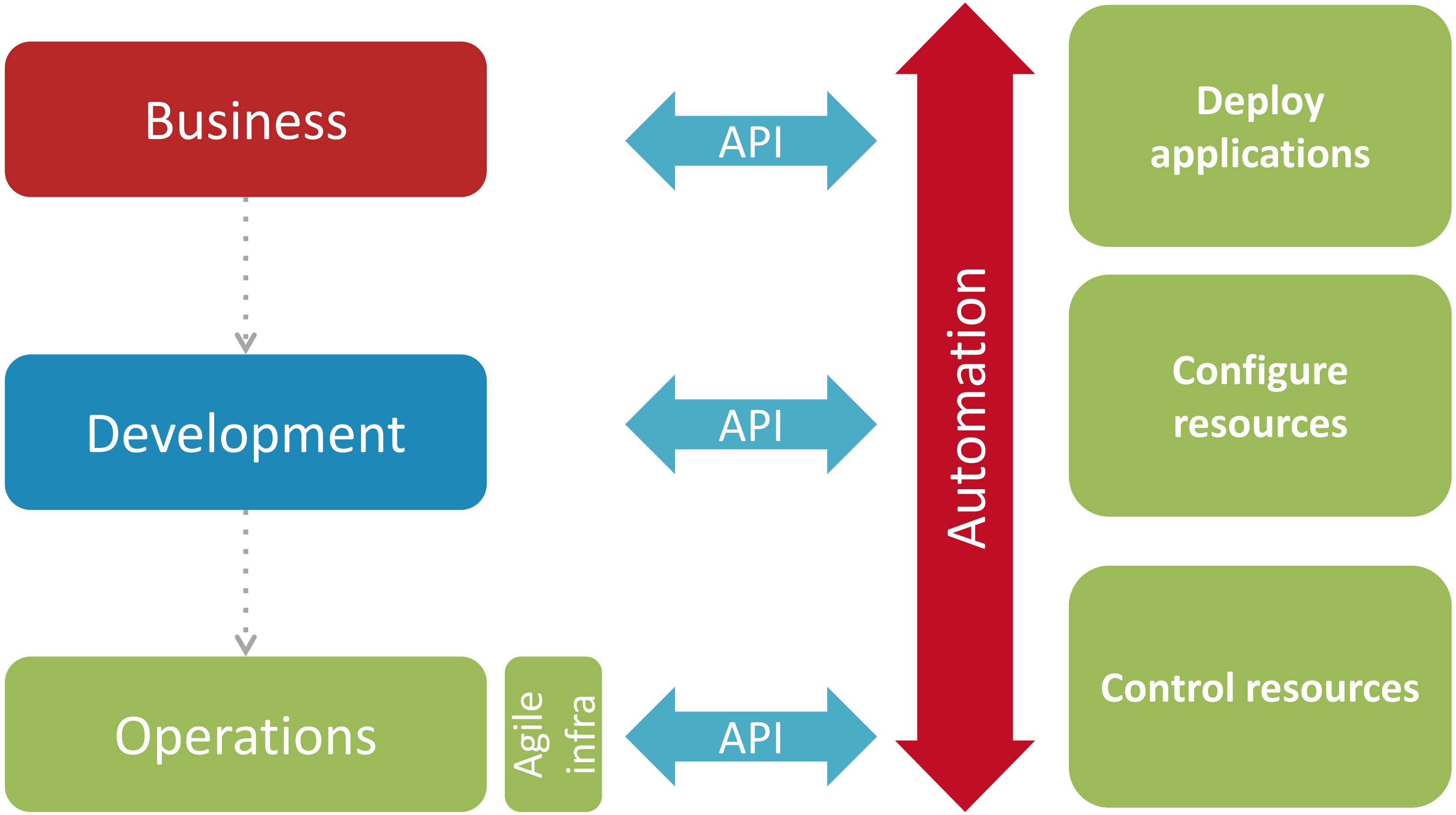
Configure resources

Define system states, when possible
Verify system states
Reconfigure systems when necessary

Control resources

Create, Delete, start, stop
Servers (physical, virtual, in the cloud)
Storage volumes, networks

Agile Infrastructure



Provide a service to deploy applications

Automated, with rollbacks

Define system states, when possible

Verify system states

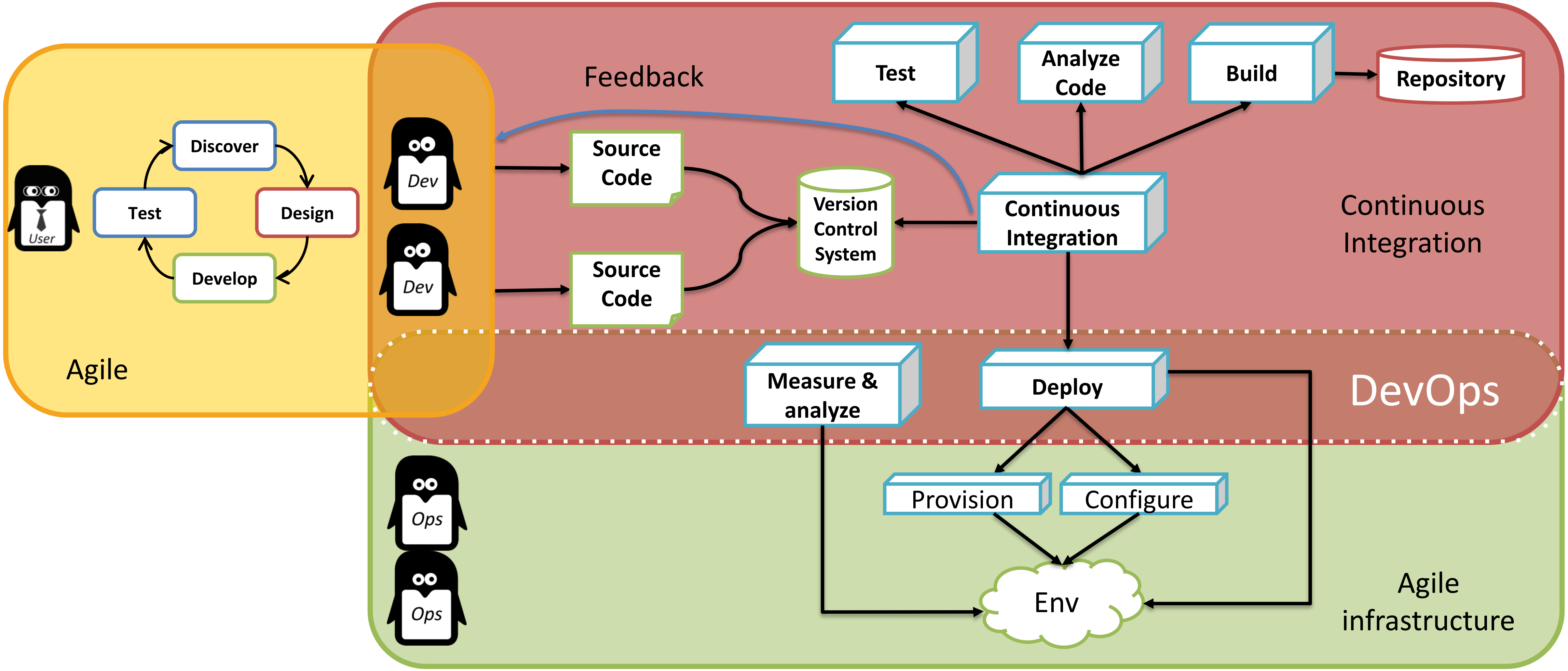
Reconfigure systems when necessary

Create, Delete, start, stop

Servers (physical, virtual, in the cloud)

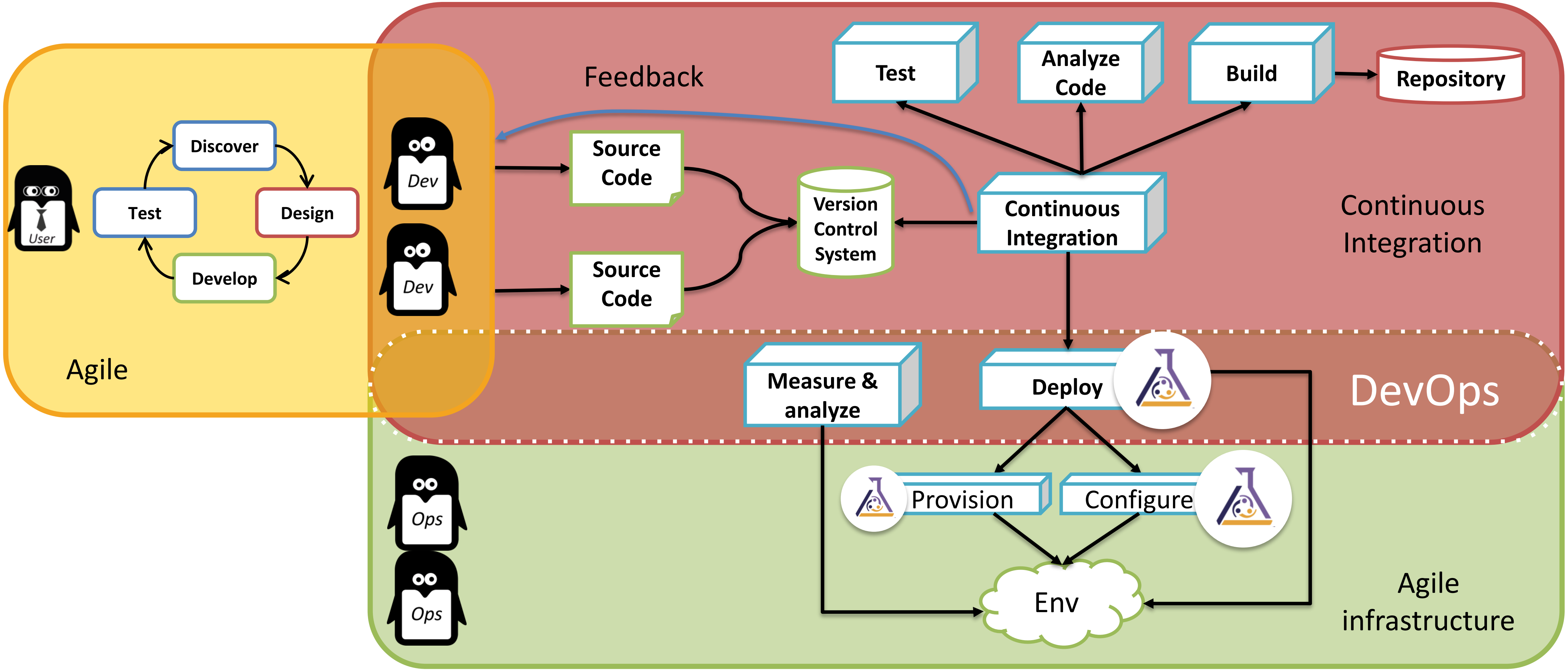
Storage volumes, networks

Continuous Delivery



What about puppet ?

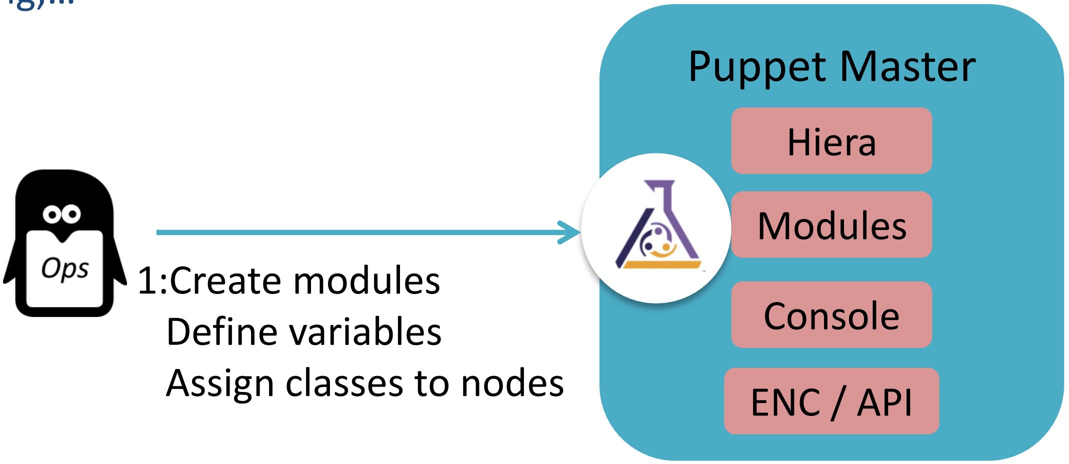
Puppet use cases



Use case 1 : core OS configuration

Server team uses puppet as a configuration tool

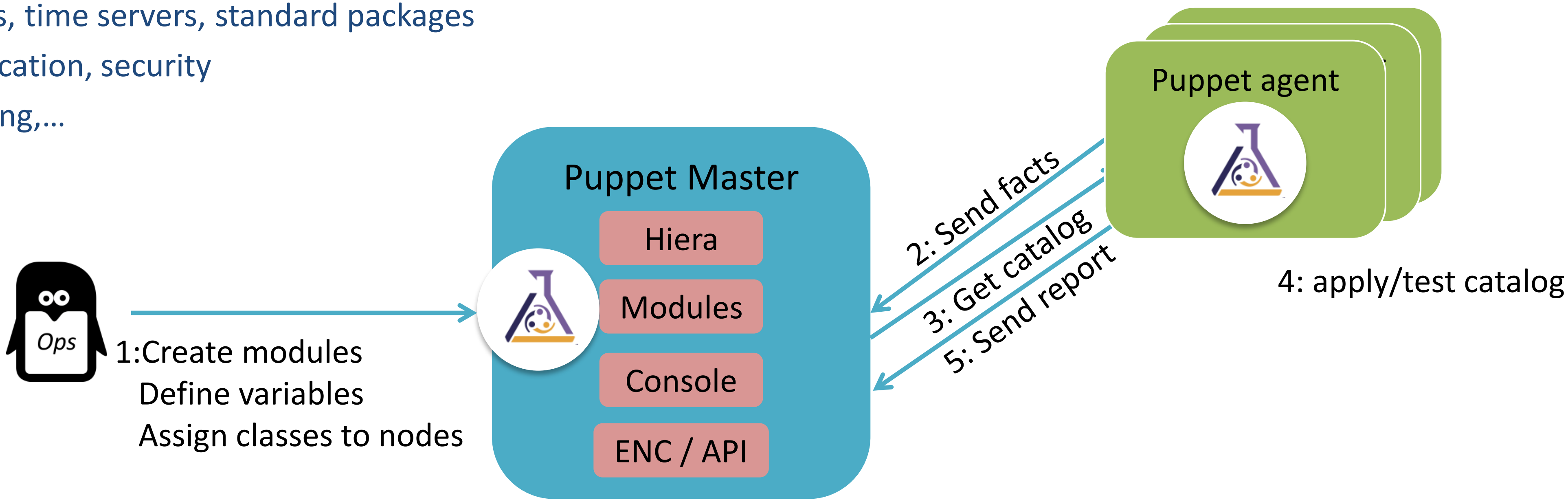
- Resolvers, time servers, standard packages
- Authentication, security
- Monitoring,…



Use case 1 : core OS configuration

Server team uses puppet as a configuration tool

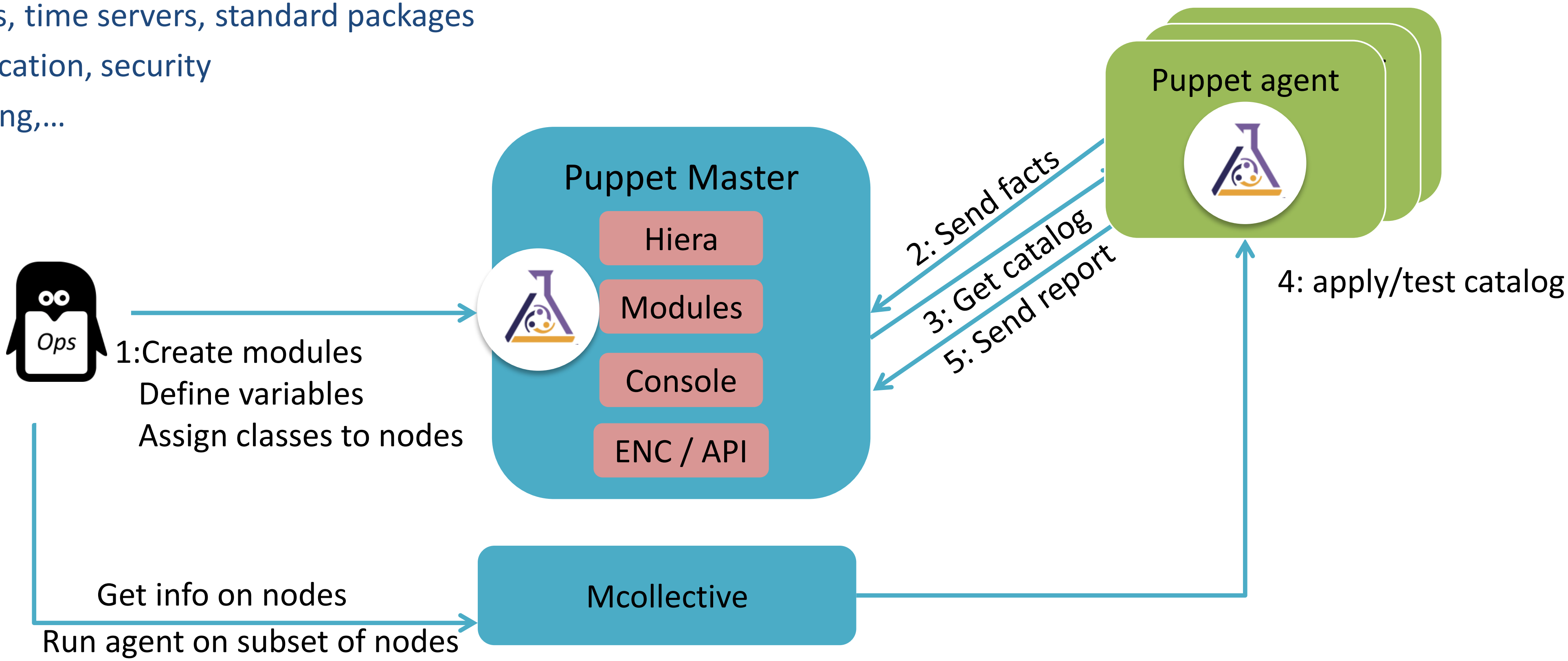
- Resolvers, time servers, standard packages
- Authentication, security
- Monitoring,…



Use case 1 : core OS configuration

Server team uses puppet as a configuration tool

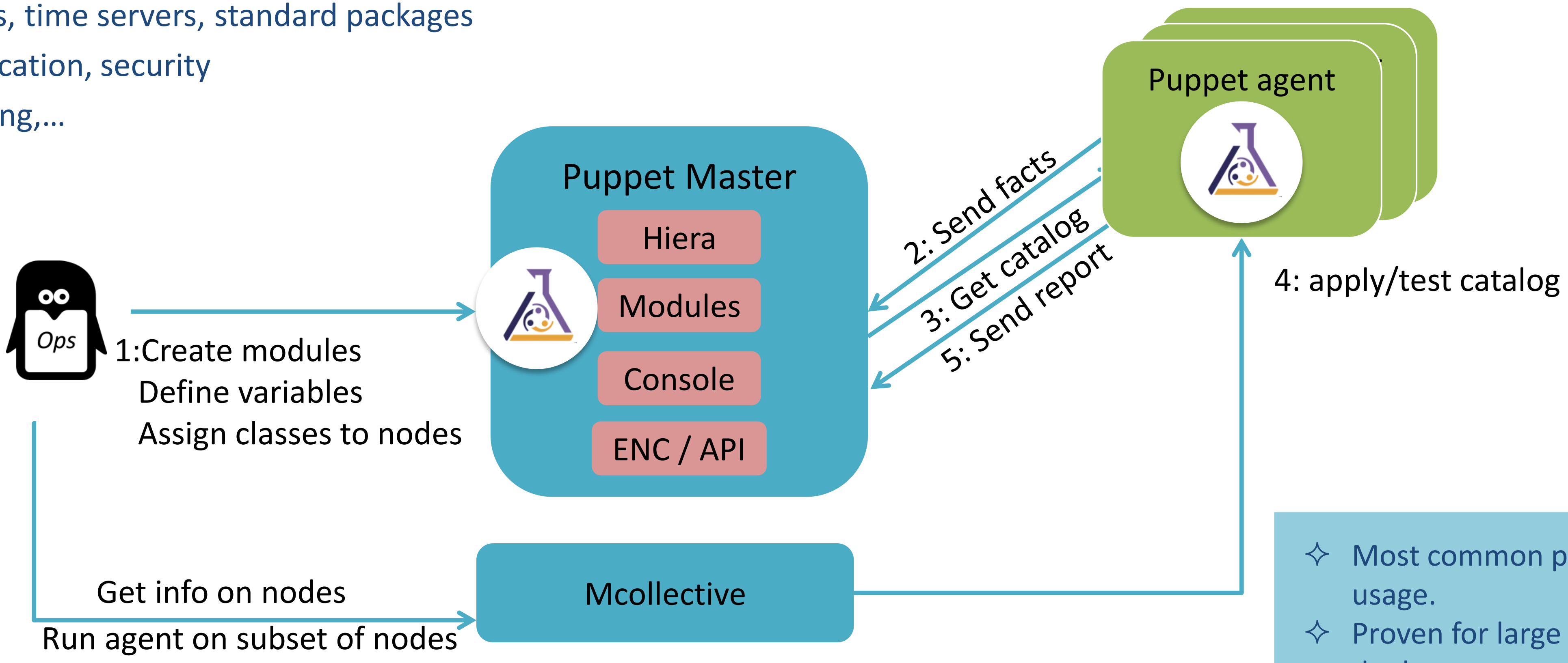
- Resolvers, time servers, standard packages
- Authentication, security
- Monitoring,…



Use case 1 : core OS configuration

Server team uses puppet as a configuration tool

- Resolvers, time servers, standard packages
- Authentication, security
- Monitoring,…



Ops

1:Create modules
Define variables
Assign classes to nodes

Puppet Master
Hiera
Modules
Console
ENC / API

2: Send facts
3: Get catalog
5: Send report

Puppet agent

4: apply/test catalog

Get info on nodes
Run agent on subset of nodes

Mcollective

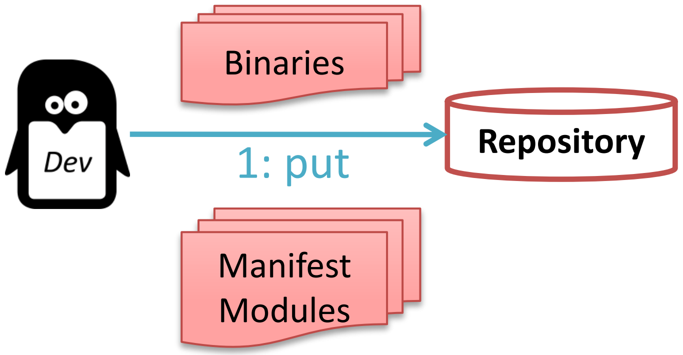
- ✧ Most common puppet usage.
- ✧ Proven for large scale deployment.
- ✧ Very “Infra oriented”: not opened to applications.

Use case 2 : deploy applications

Developers supply :

Binaries of the application

- Puppet manifests and modules describing deployments



Use case 2 : deploy applications

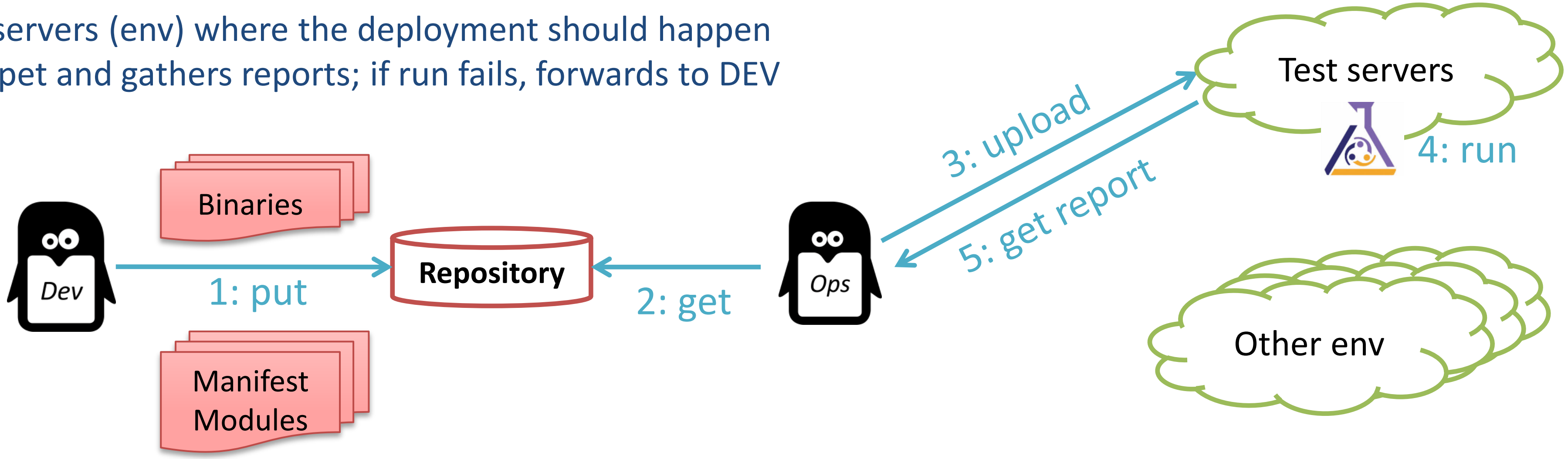
Developers supply :

Binaries of the application

- Puppet manifests and modules describing deployments

OPS team :

- Chooses servers (env) where the deployment should happen
- Runs puppet and gathers reports; if run fails, forwards to DEV



Use case 2 : deploy applications

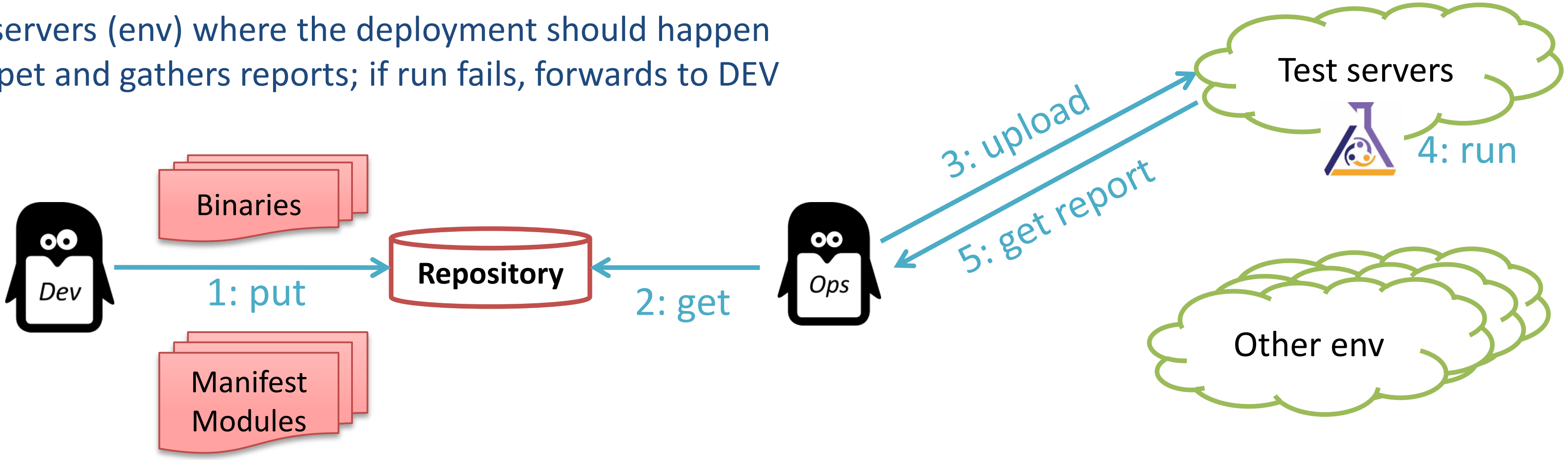
Developers supply :

Binaries of the application

- Puppet manifests and modules describing deployments

OPS team :

- Chooses servers (env) where the deployment should happen
- Runs puppet and gathers reports; if run fails, forwards to DEV



- ✧ Much more efficient than written deployment processes.
- ✧ Much easier to understand what fails.

Configuration as a service

Tools

Levels

Profils

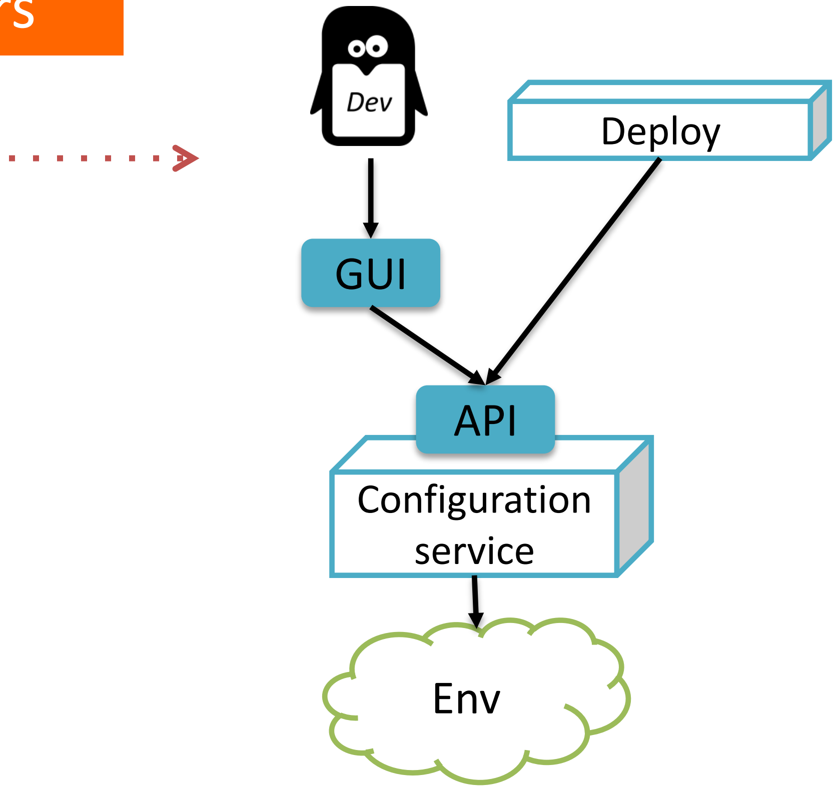
Report

Configuration

Using a “configuration service”

1 Give application teams the possibility to configure servers

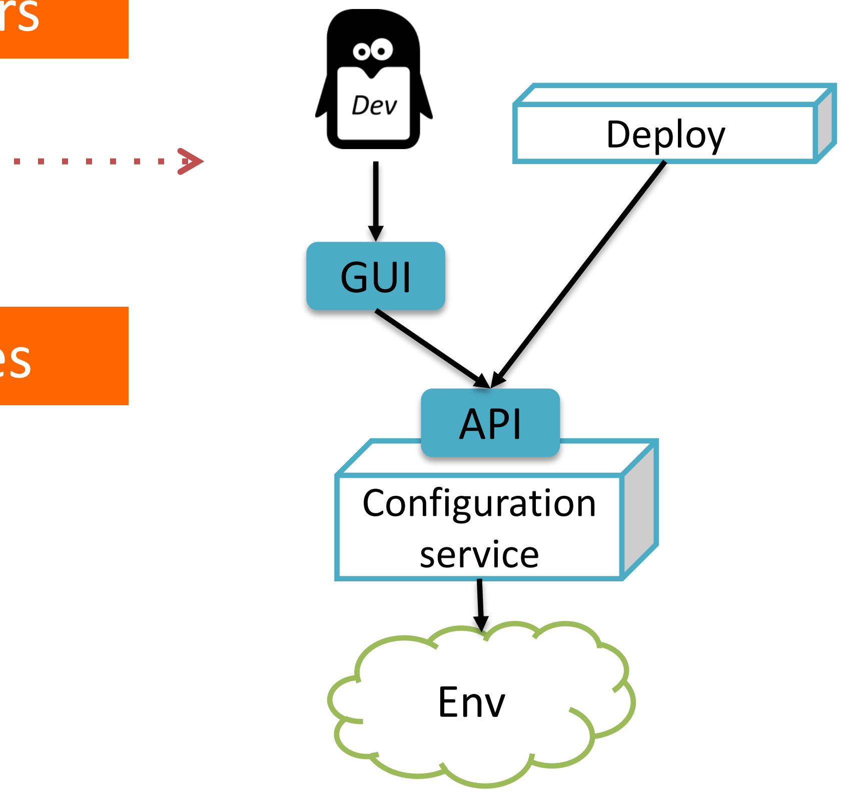
- Associate “profiles” to nodes, define variables
- Run configuration and get reports



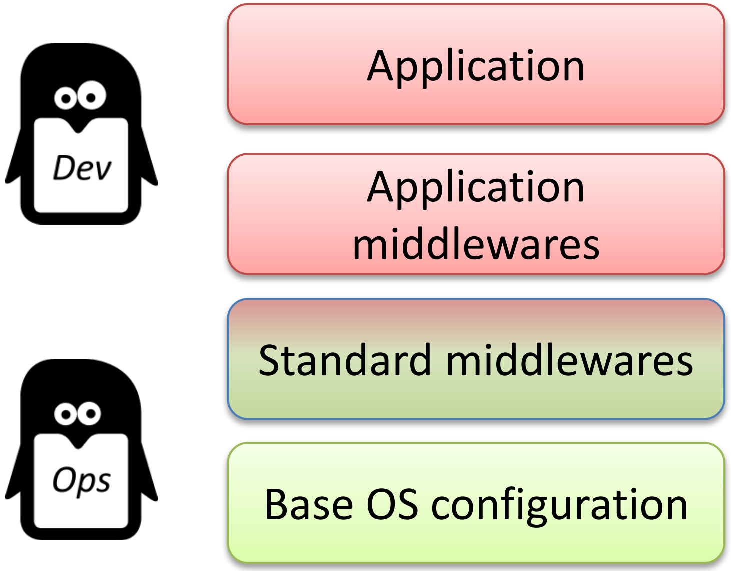
Using a “configuration service”

1 Give application teams the possibility to configure servers

- Associate “profiles” to nodes, define variables
- Run configuration and get reports



2 Different levels of configuration, different responsibilities



Can we do that with puppet?

Can we do that with puppet?

SURE, but tricky with classic DEV / OPS model

DEV cannot execute anything as root

Can we do that with puppet?

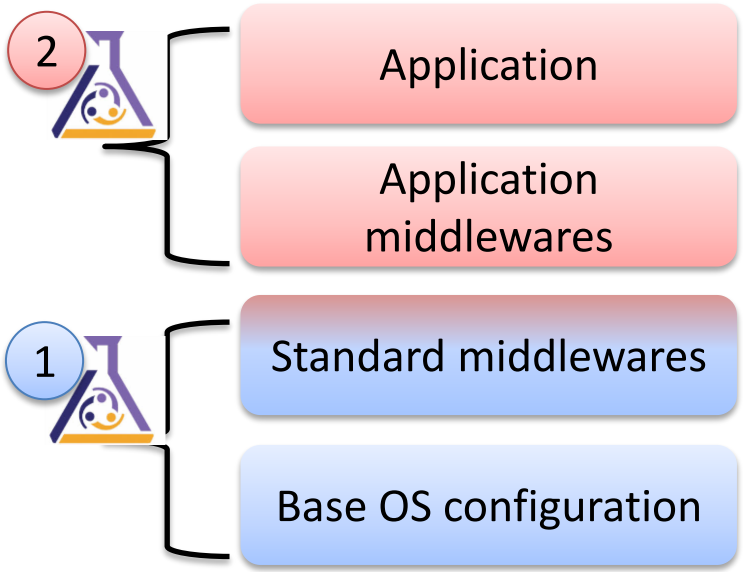
SURE, but tricky with classic DEV / OPS model

DEV cannot execute anything as root

Some options :

1 Tool separation

- Second puppet master, or puppet apply (non root)
- Other tool



2

Application

Application middlewares

1

Standard middlewares

Base OS configuration

Can we do that with puppet?

SURE, but tricky with classic DEV / OPS model

DEV cannot execute anything as root

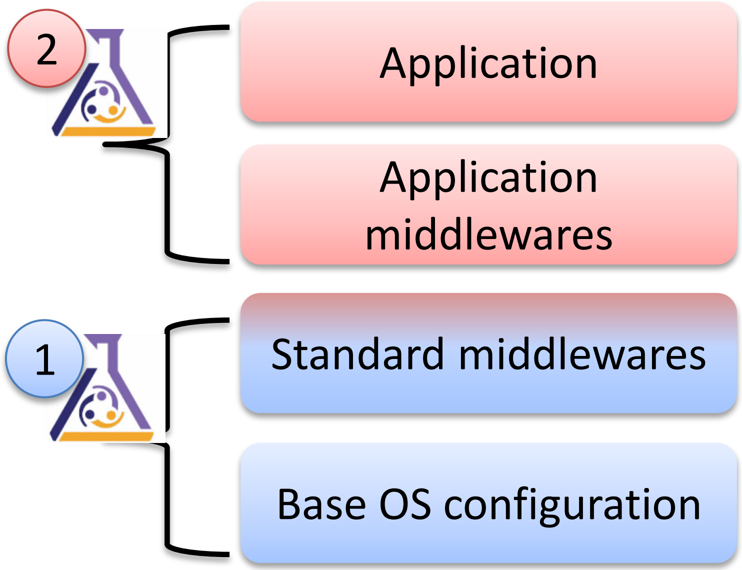
Some options :

1 Tool separation

- Second puppet master, or puppet apply (non root)
- Other tool

2 Ok to run as root but under full control

- Custom “profile” facts (facts.d) and hiera
- Run with mcollective (limit to some tags)
- Read-only console access



Can we do that with puppet?

SURE, but tricky with classic DEV / OPS model

DEV cannot execute anything as root

Some options :

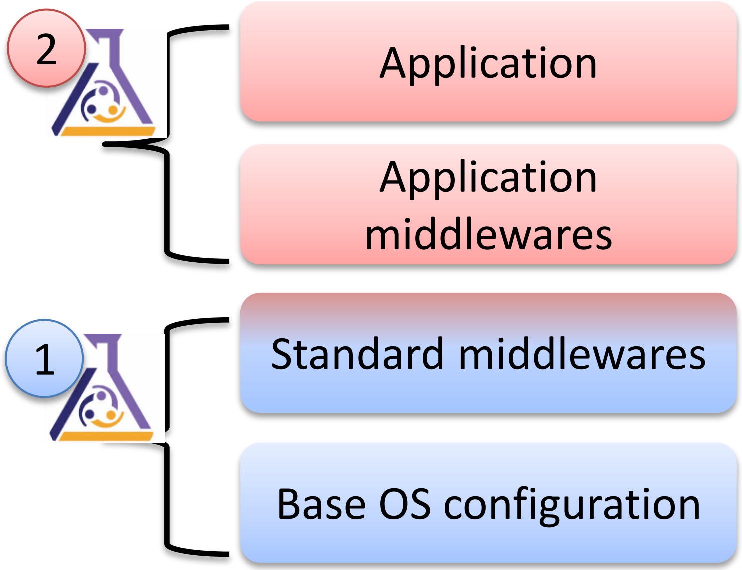
1 Tool separation

- Second puppet master, or puppet apply (non root)
- Other tool

2 Ok to run as root but under full control

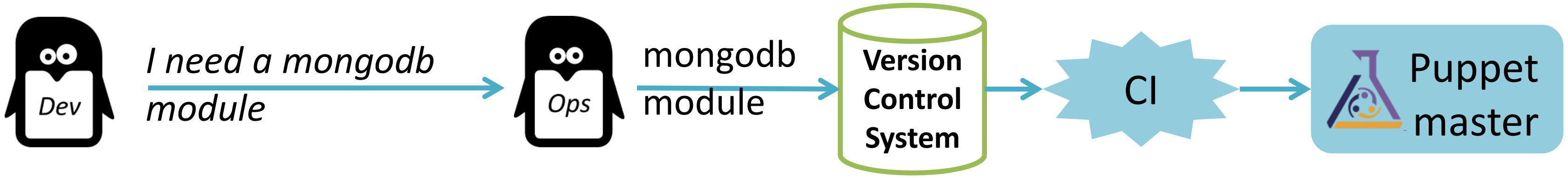
- Custom “profile” facts (facts.d) and hiera
- Run with mcollective (limit to some tags)
- Read-only console access

3 Many other ways



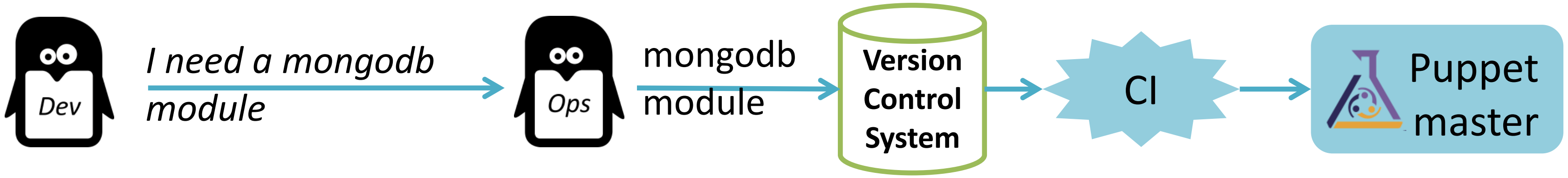
What if DEV need custom modules (they will)

Approach 1 : OPS write all modules



What if DEV need custom modules (they will)

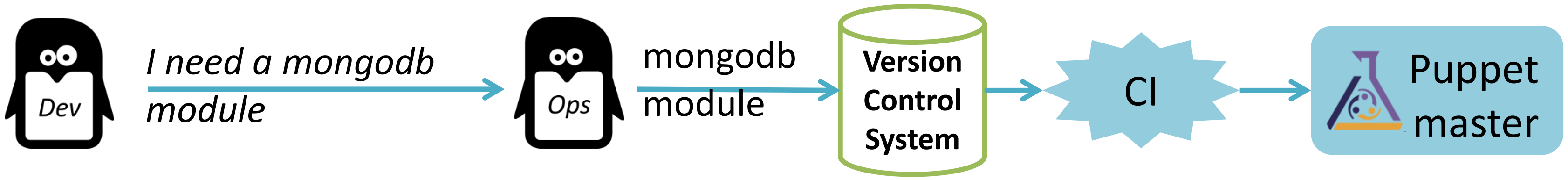
Approach 1 : OPS write all modules



- ✧ **Impossible to scale**
- ✧ **Not efficient**

What if DEV need custom modules (they will)

Approach 1 : OPS write all modules



Dev

I need a mongodb module

Ops

mongodb module

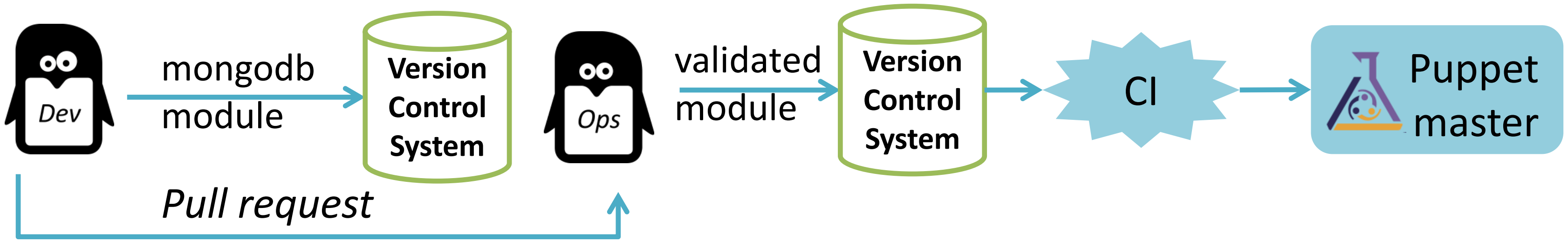
Version Control System

CI

Puppet master

- ✧ **Impossible to scale**
- ✧ **Not efficient**

Approach 2 : Pull request



Dev

mongodb module

Version Control System

Ops

validated module

Version Control System

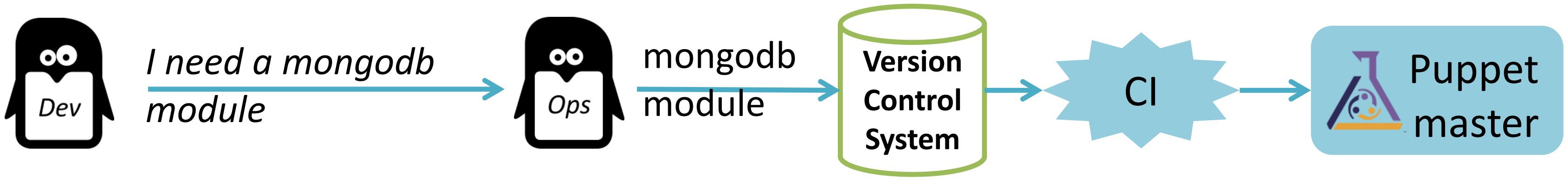
CI

Puppet master

Pull request

What if DEV need custom modules (they will)

Approach 1 : OPS write all modules



Dev

I need a mongodb module

Ops

mongodb module

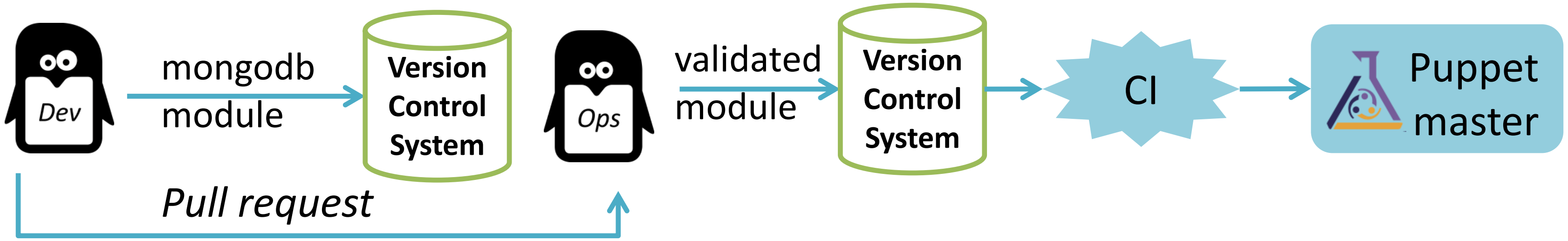
Version Control System

CI

Puppet master

- ✧ **Impossible to scale**
- ✧ **Not efficient**

Approach 2 : Pull request



Dev

mongodb module

Version Control System

Ops

validated module

Version Control System

CI

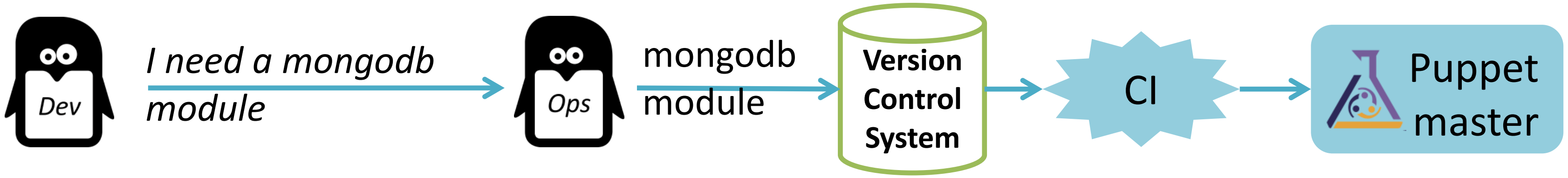
Puppet master

Pull request

- ✧ **Very limited scalabity**

What if DEV need custom modules (they will)

Approach 1 : OPS write all modules



Dev

I need a mongodb module

Ops

mongodb module

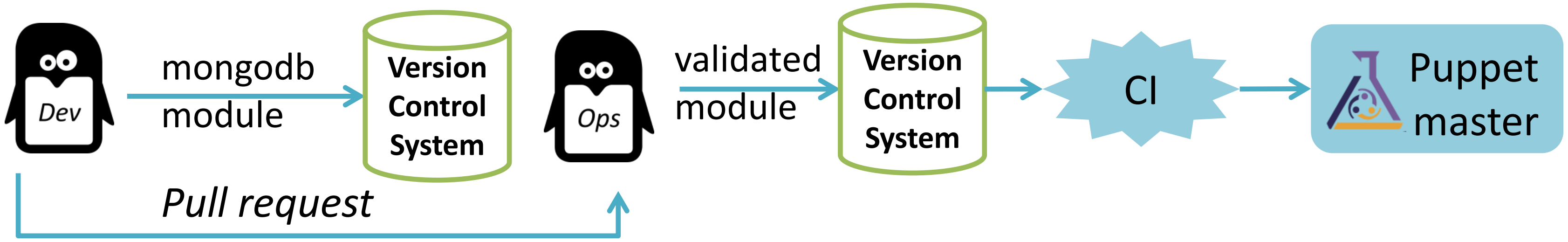
Version Control System

CI

Puppet master

- ✧ **Impossible to scale**
- ✧ **Not efficient**

Approach 2 : Pull request



Dev

mongodb module

Version Control System

Ops

validated module

Version Control System

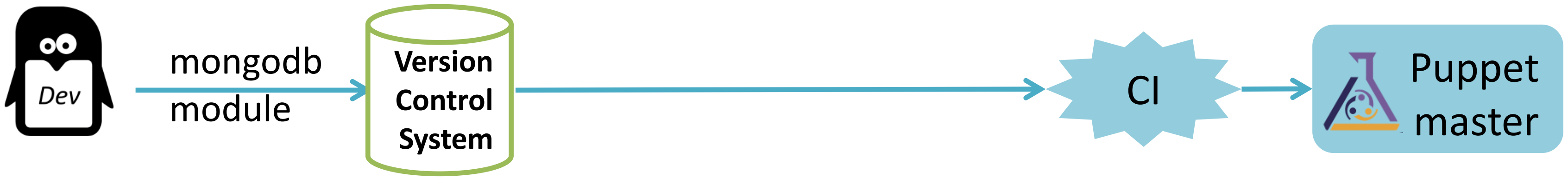
CI

Puppet master

Pull request

- ✧ **Very limited scalabity**

Approach 3: DEV can push to some repositories



Dev

mongodb module

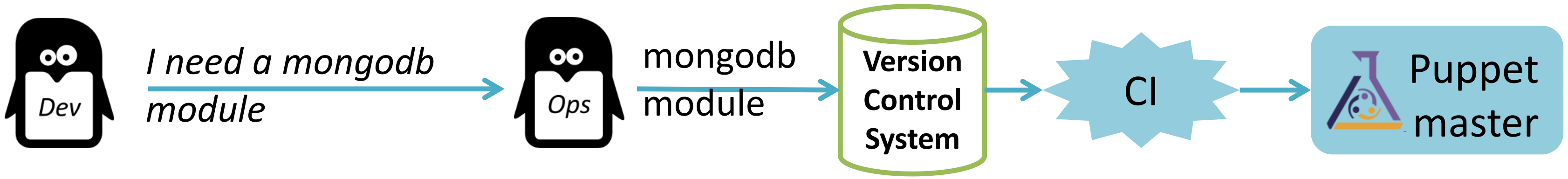
Version Control System

CI

Puppet master

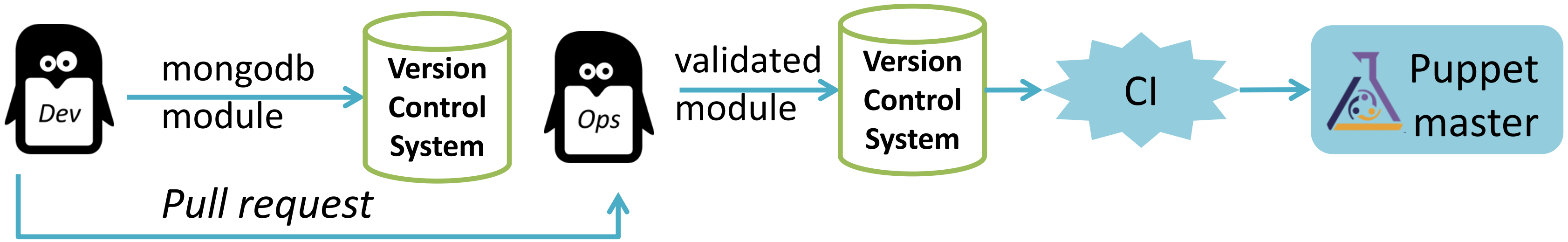
What if DEV need custom modules (they will)

Approach 1 : OPS write all modules



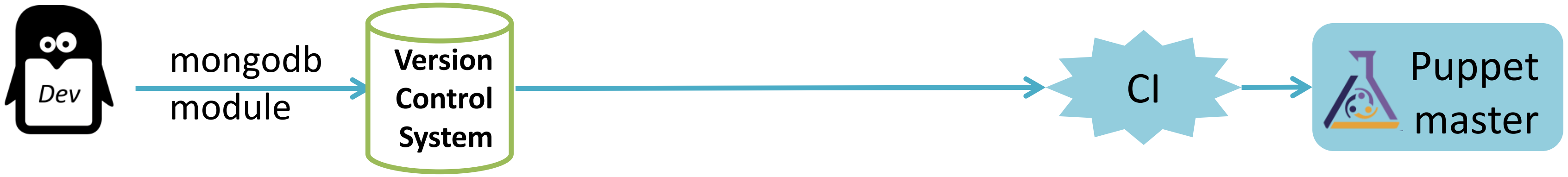
- Impossible to scale
- Not efficient

Approach 2 : Pull request



- Very limited scalabity

Approach 3: DEV can push to some repositories



- Complex permissions
- DEV are still basically root

Sure, we can tweak puppet

Sure, we can tweak puppet

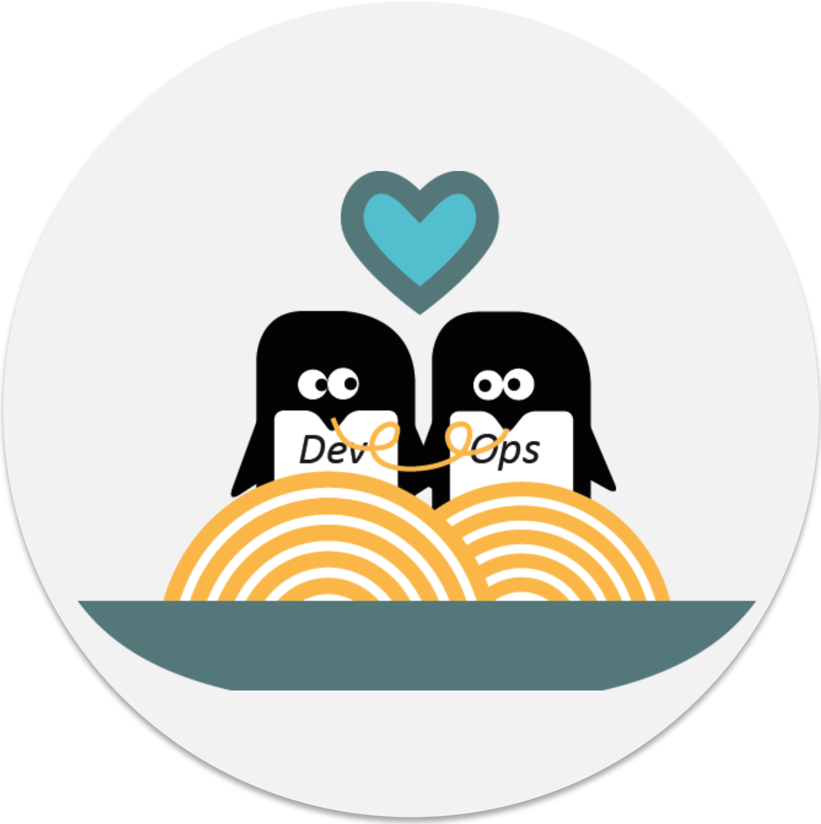
Is this this the way??

Sure, we can tweak puppet

Is this this the way??

Maybe we should adapt processes and not just tools

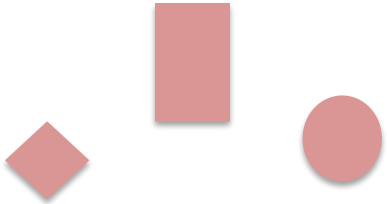
A NEW relationship between DEV & OPS



From separation and control to shared responsiblities

Dev

- Provide application
- Ask for env

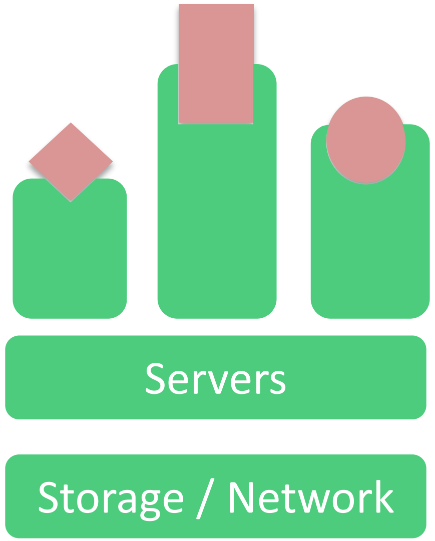


Ops

From separation and control to shared responsiblities

Dev

- Provide application
- Ask for env



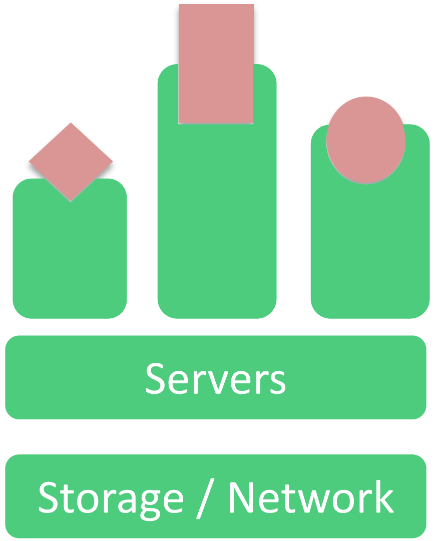
Ops

- Provide env
- Run production

From separation and control to shared responsiblities

Dev

- Provide application
- Ask for env



Servers

Storage / Network

Ops

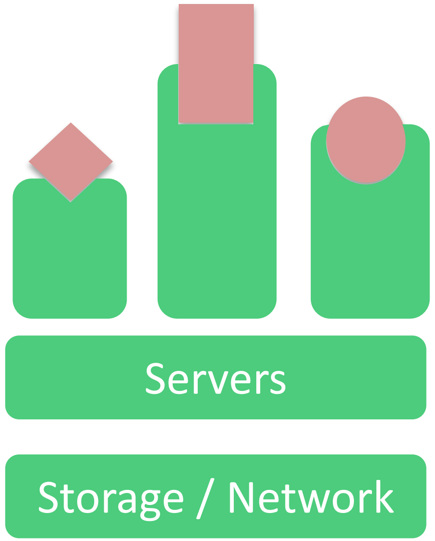
- Provide env
- Run production

Strict separation of roles

From separation and control to shared responsiblities

Dev

- Provide application
- Ask for env

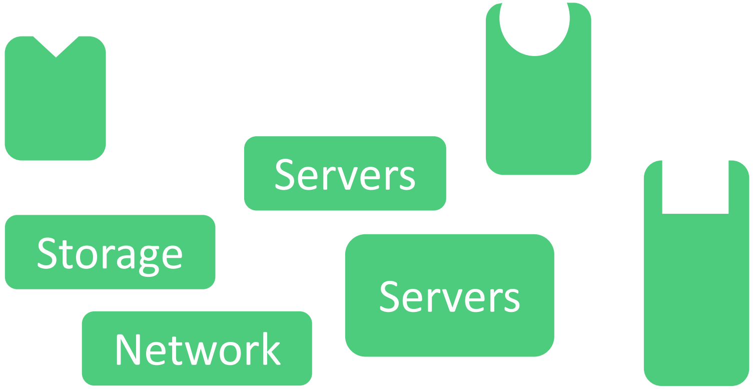


Ops

- Provide env
- Run production

Strict separation of roles

Dev



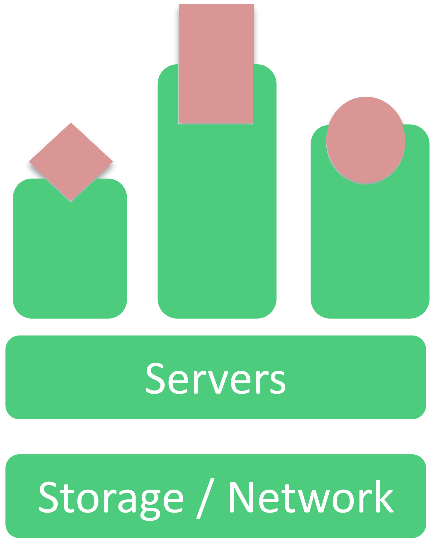
Ops

- Provide programmable resources
- Provide advice
- Delegate some Prod responsability

From separation and control to shared responsiblities

Dev

- Provide application
- Ask for env



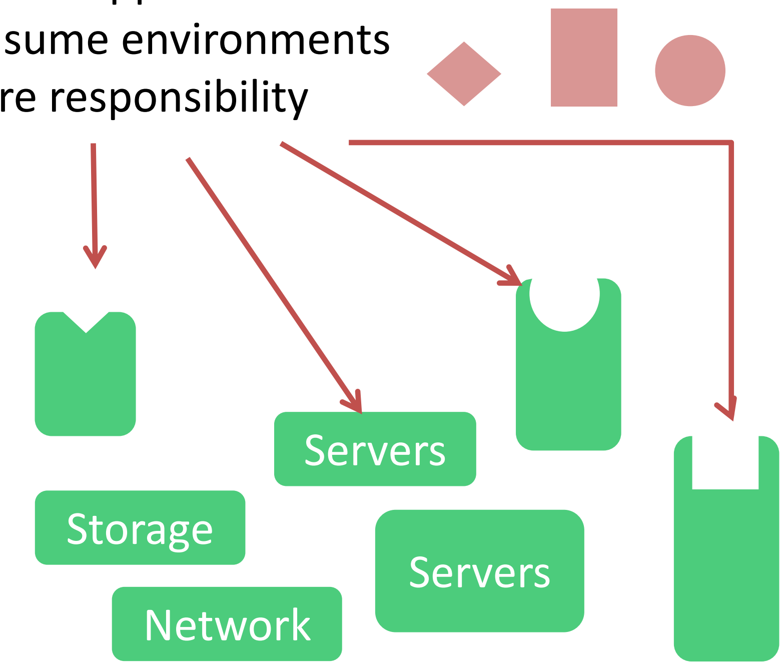
Ops

- Provide env
- Run production

Strict separation of roles

Dev

- Provide application
- Consume environments
- Share responsibility



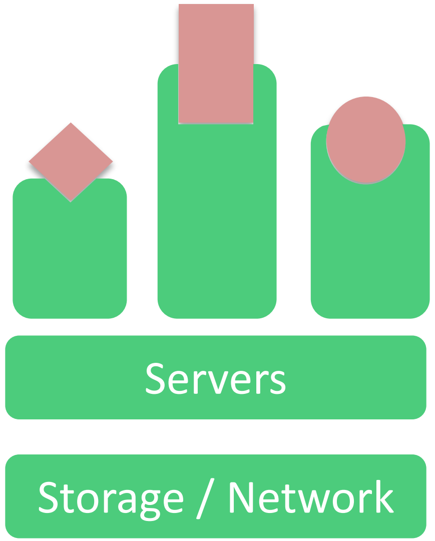
Ops

- Provide programmable resources
- Provide advice
- Delegate some Prod responsability

From separation and control to shared responsiblities

Dev

- Provide application
- Ask for env



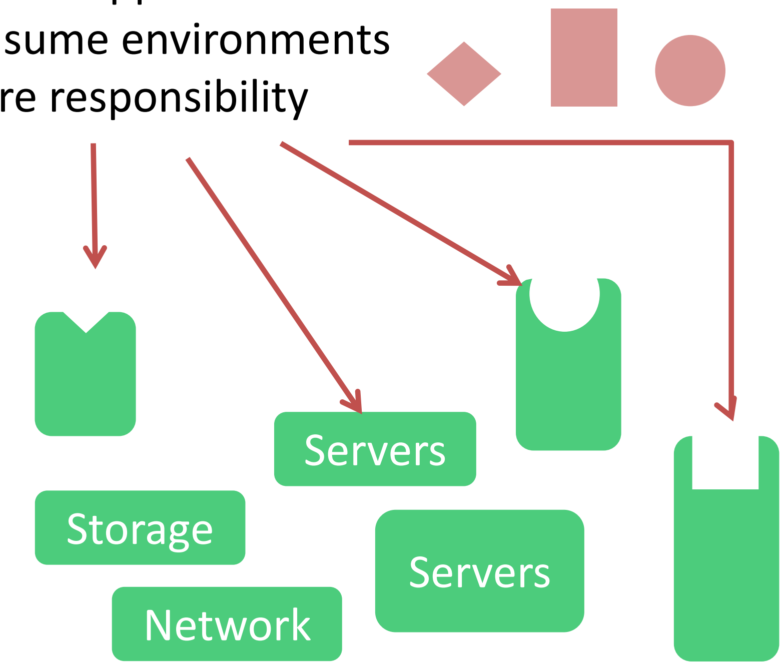
Ops

- Provide env
- Run production

Strict separation of roles

Dev

- Provide application
- Consume environments
- Share responsibility



Ops

- Provide programmable resources
- Provide advice
- Delegate some Prod responsability

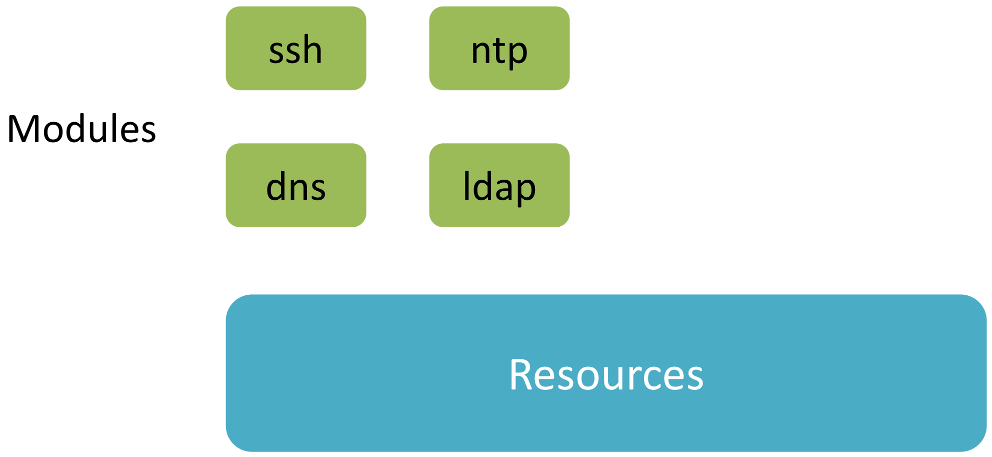
Shared responsibilities

What it could look like with the profile/role pattern

“Designing Puppet: Roles / Profiles Design Pattern
Puppet Camp Stockholm, Feb 2013 (Craig Dunn Puppet Labs)

What it could look like with the profile/role pattern

OPS provide core OS modules

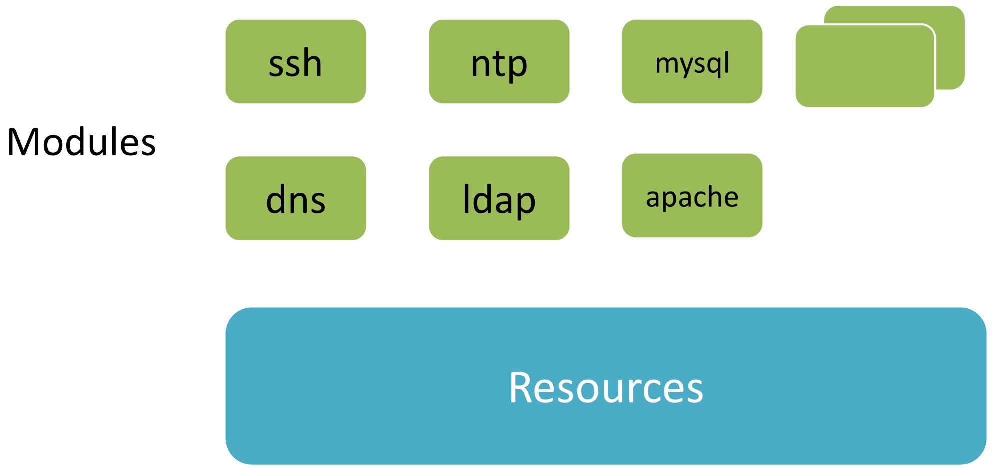


“Designing Puppet: Roles / Profiles Design Pattern
Puppet Camp Stockholm, Feb 2013 (Craig Dunn Puppet Labs)

What it could look like with the profile/role pattern

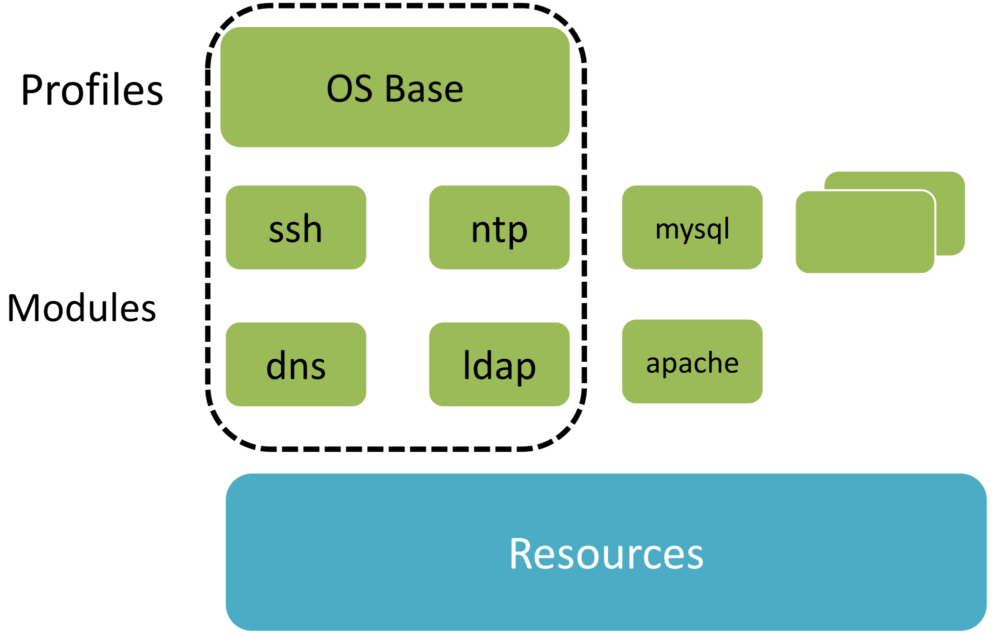
OPS provide core OS modules

OPS provide middleware modules



“Designing Puppet: Roles / Profiles Design Pattern
Puppet Camp Stockholm, Feb 2013 (Craig Dunn Puppet Labs)

What it could look like with the profile/role pattern



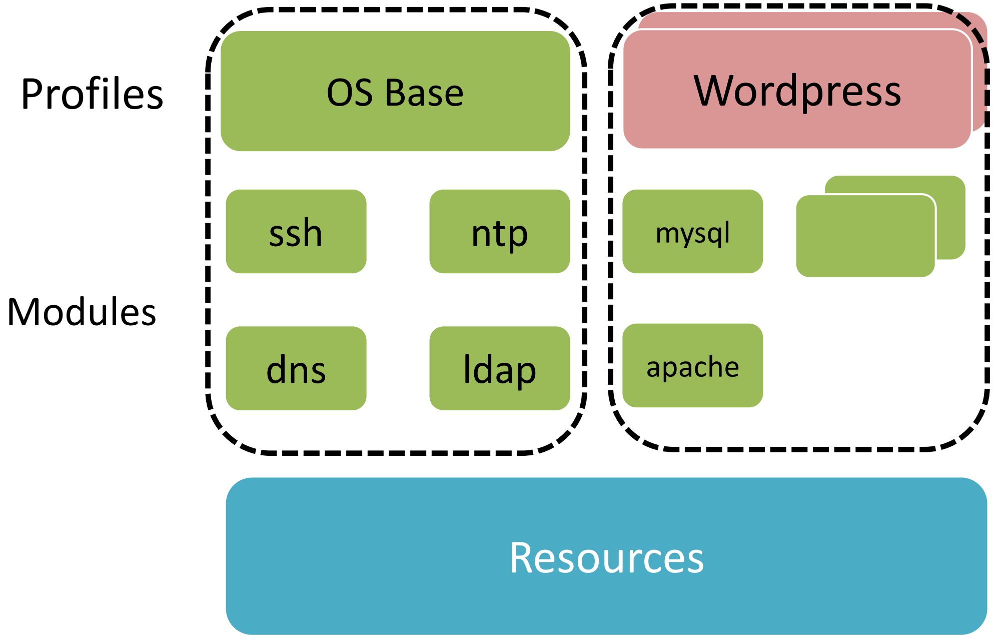
OPS provide core OS modules

OPS provide middleware modules

OPS provide Base profile

“Designing Puppet: Roles / Profiles Design Pattern
Puppet Camp Stockholm, Feb 2013 (Craig Dunn Puppet Labs)

What it could look like with the profile/role pattern



OPS provide core OS modules

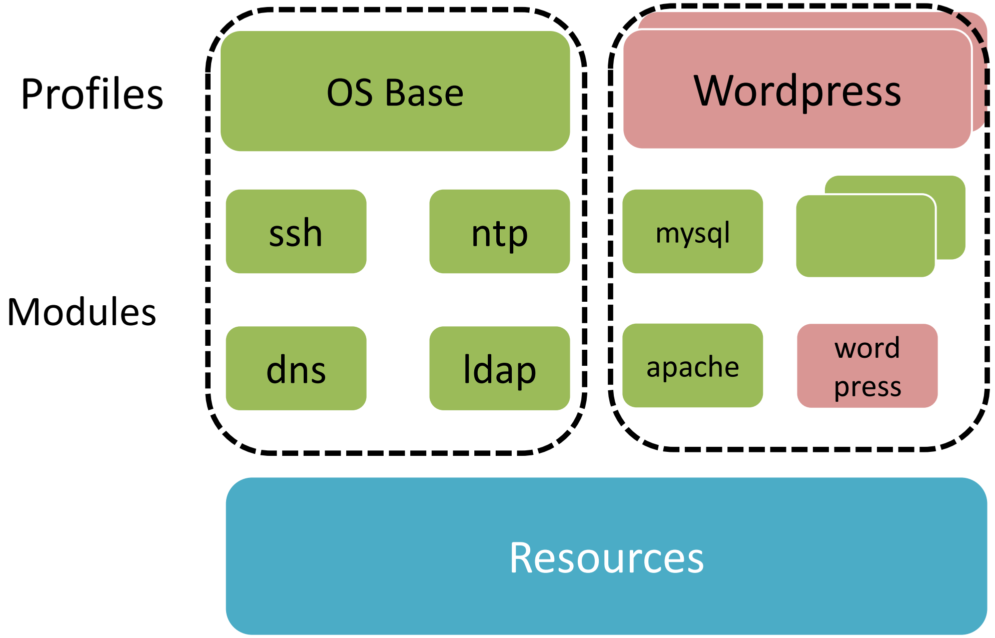
OPS provide middleware modules

OPS provide Base profile

DEV create profiles using modules

“Designing Puppet: Roles / Profiles Design Pattern
Puppet Camp Stockholm, Feb 2013 (Craig Dunn Puppet Labs)

What it could look like with the profile/role pattern



OPS provide core OS modules

OPS provide middleware modules

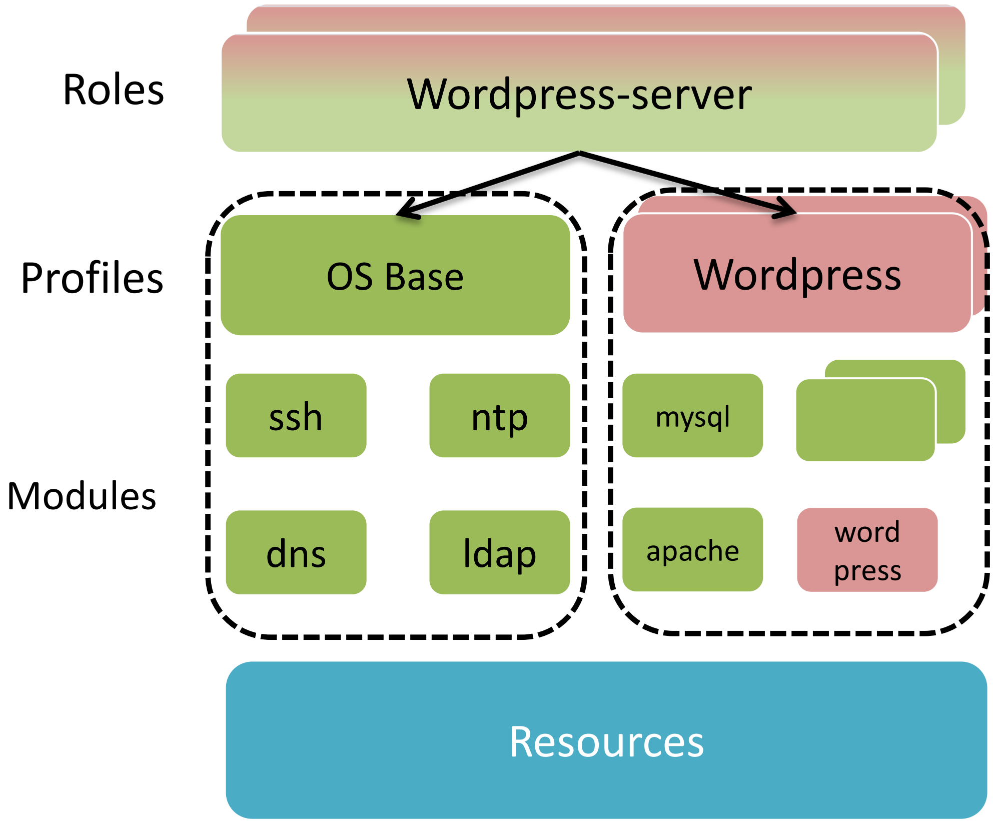
OPS provide Base profile

DEV create profiles using modules

DEV create some custom modules

“Designing Puppet: Roles / Profiles Design Pattern
Puppet Camp Stockholm, Feb 2013 (Craig Dunn Puppet Labs)

What it could look like with the profile/role pattern



OPS provide core OS modules

OPS provide middleware modules

OPS provide Base profile

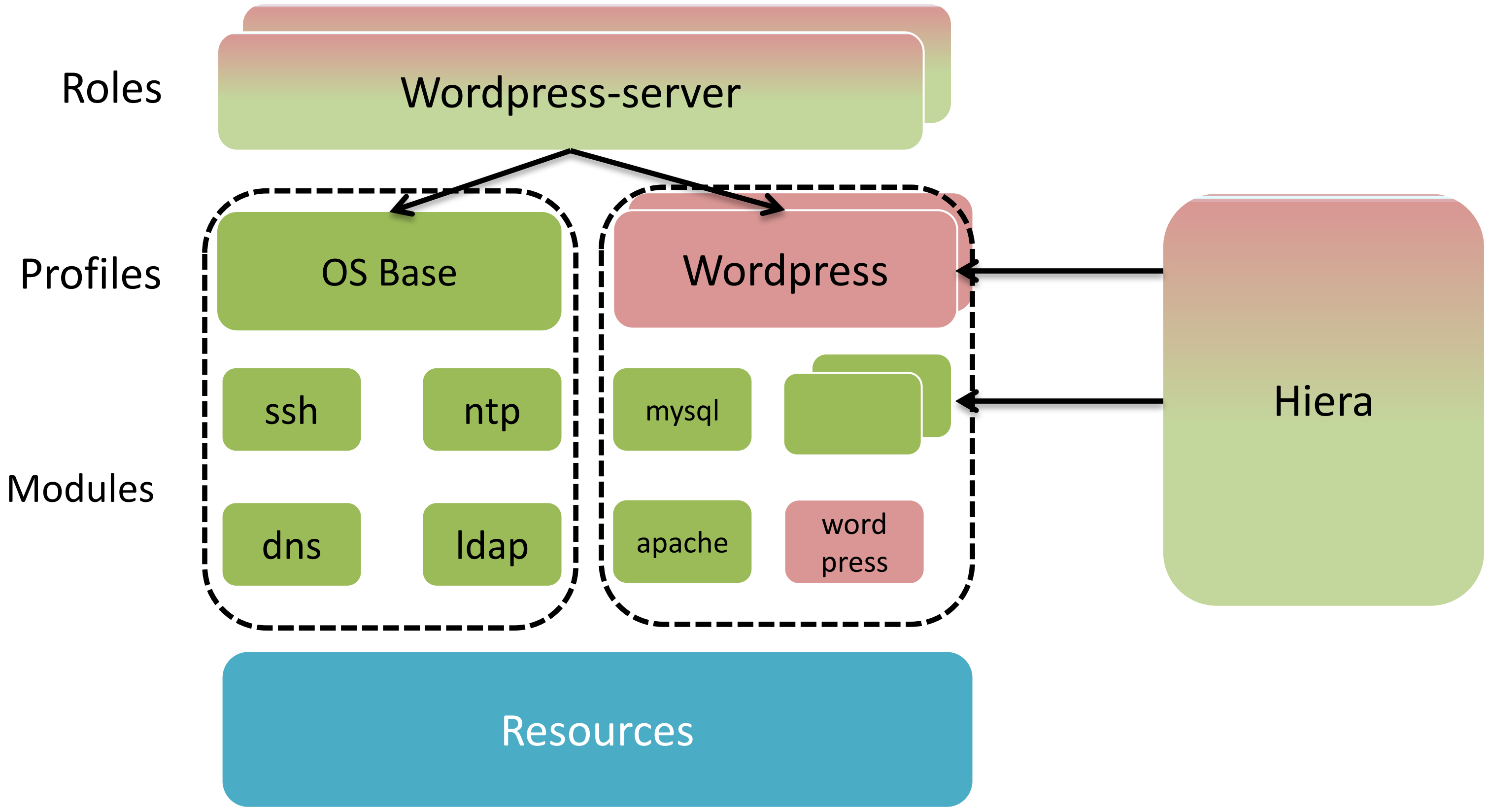
DEV create profiles using modules

DEV create some custom modules

DEV & OPS define roles

“Designing Puppet: Roles / Profiles Design Pattern
Puppet Camp Stockholm, Feb 2013 (Craig Dunn Puppet Labs)

What it could look like with the profile/role pattern



OPS provide core OS modules

OPS provide middleware modules

OPS provide Base profile

DEV create profiles using modules

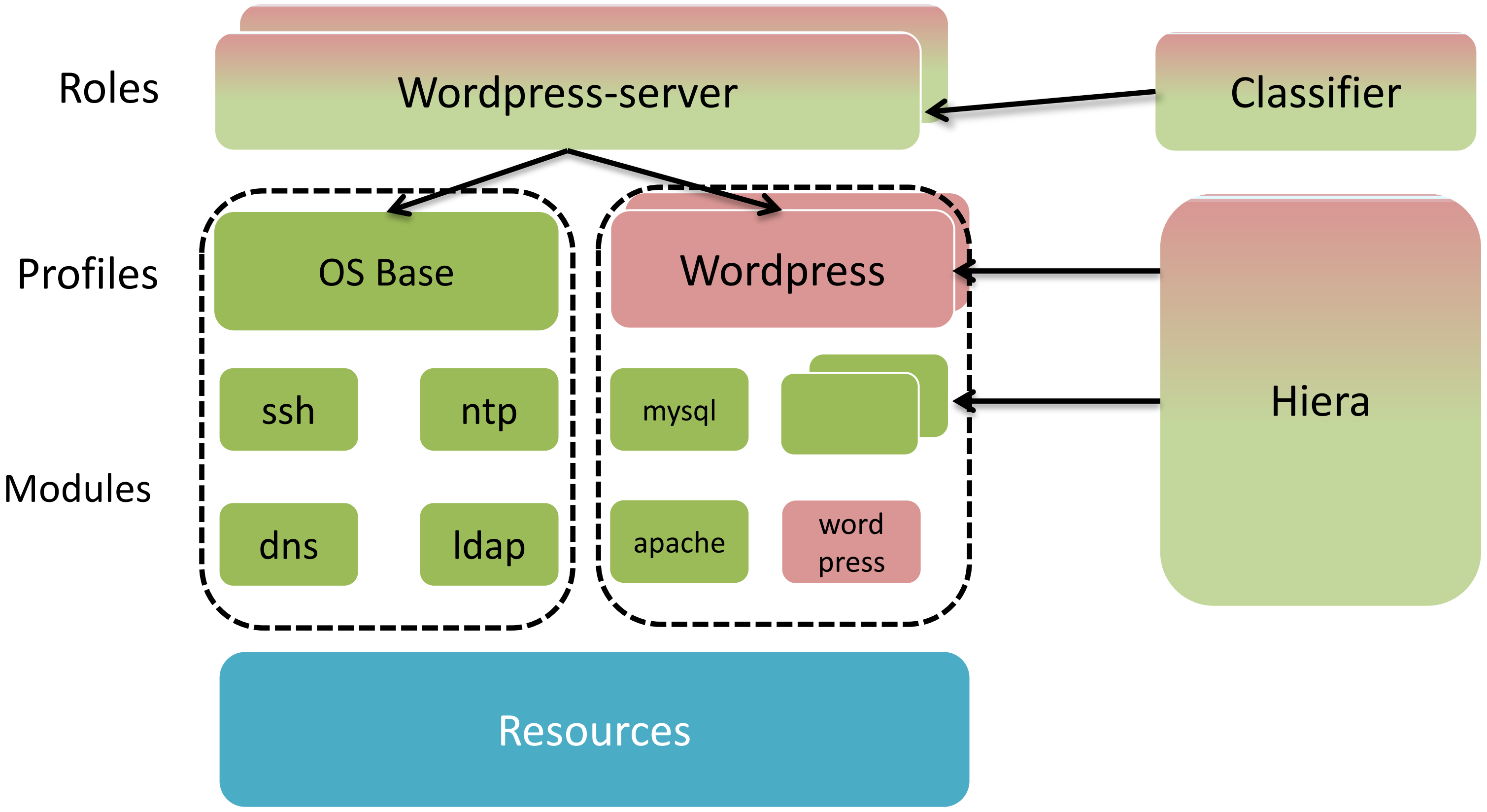
DEV create some custom modules

DEV & OPS define roles

DEV & OPS define variables

“Designing Puppet: Roles / Profiles Design Pattern
Puppet Camp Stockholm, Feb 2013 (Craig Dunn Puppet Labs)

What it could look like with the profile/role pattern



OPS provide core OS modules

OPS provide middleware modules

OPS provide Base profile

DEV create profiles using modules

DEV create some custom modules

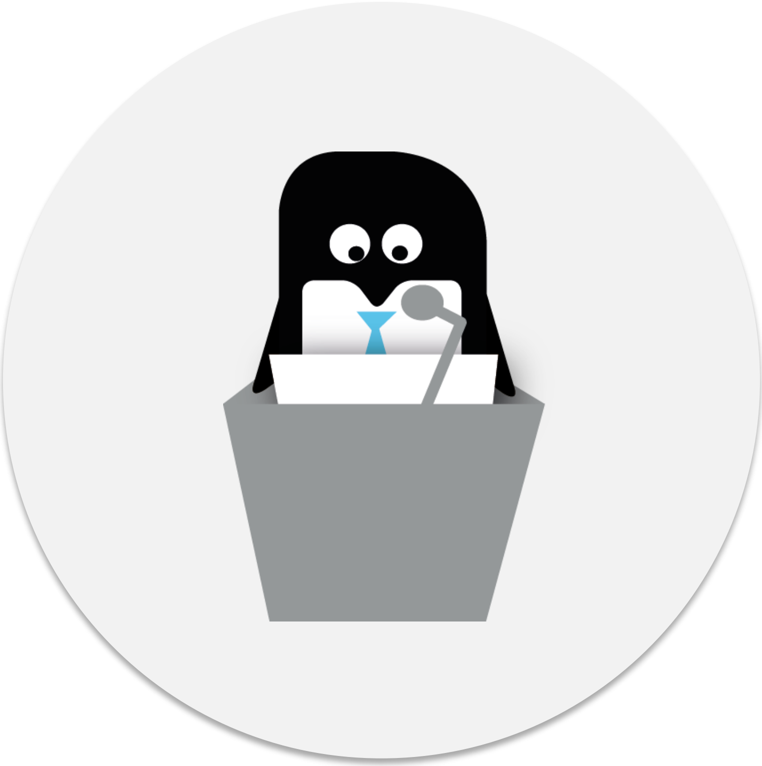
DEV & OPS define roles

DEV & OPS define variables

DEV & OPS associate roles to nodes

“Designing Puppet: Roles / Profiles Design Pattern
Puppet Camp Stockholm, Feb 2013 (Craig Dunn Puppet Labs)

Final words



Our feedback on puppet

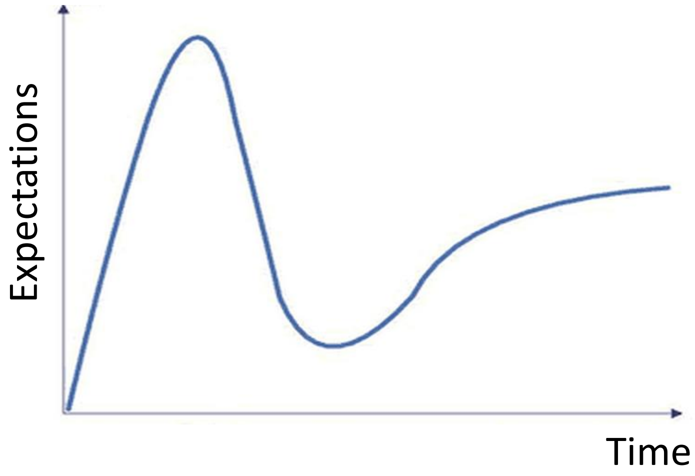
Puppet is an amazing tool

- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient

Our feedback on puppet

Puppet is an amazing tool

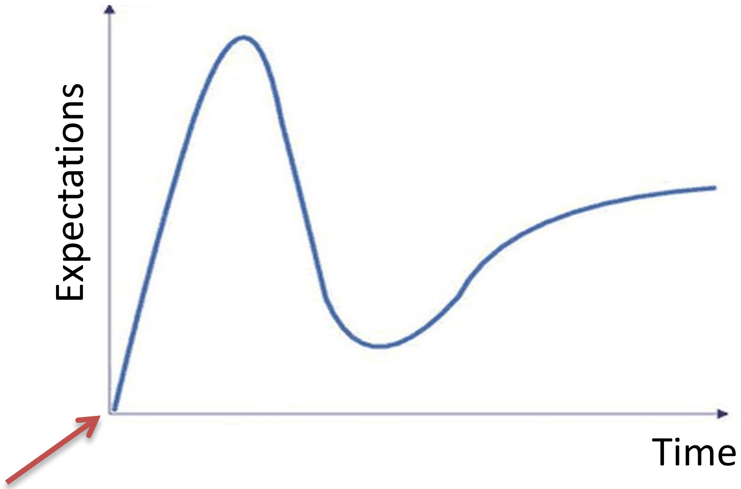
- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient



Our feedback on puppet

Puppet is an amazing tool

- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient

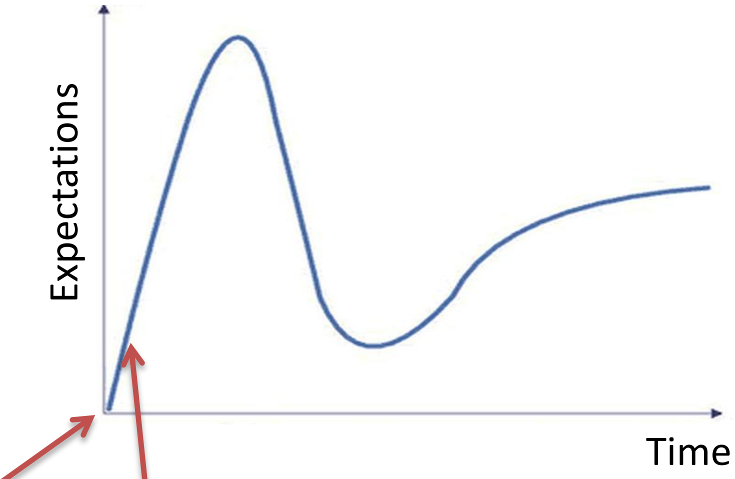


Puppet???

Our feedback on puppet

Puppet is an amazing tool

- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient



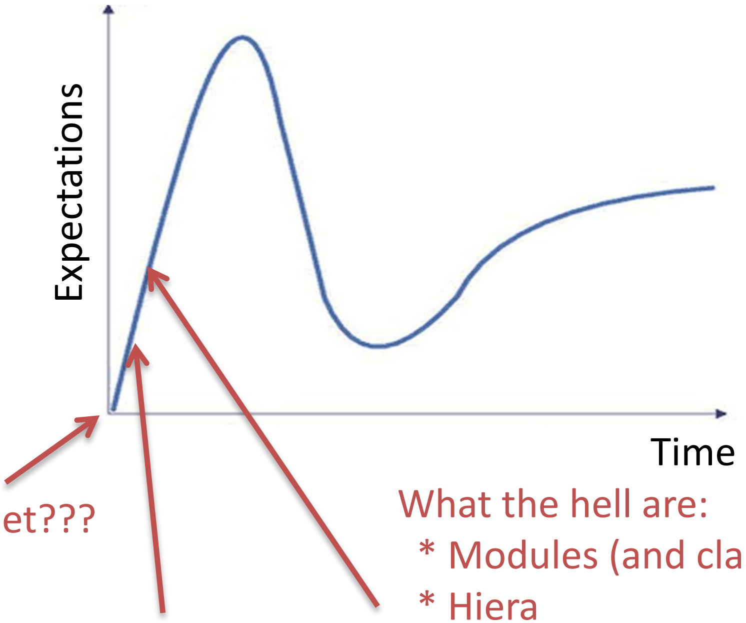
Puppet???

OK, looks interesting
First puppet apply

Our feedback on puppet

Puppet is an amazing tool

- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient



Puppet???

OK, looks interesting
First puppet apply

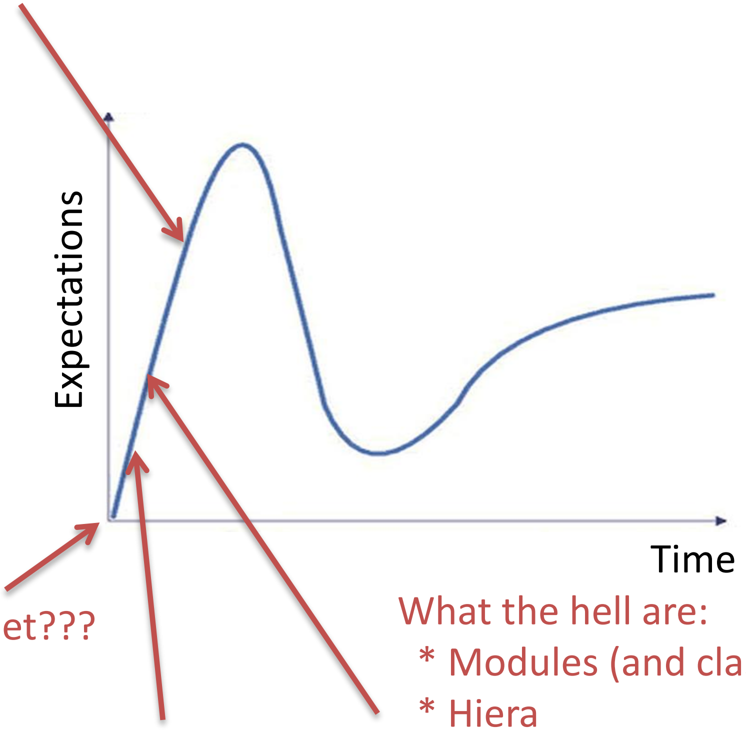
What the hell are:
* Modules (and classes)
* Hiera
* erb
* spaceships??

Our feedback on puppet

Puppet is an amazing tool

- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient

First modules



Puppet???

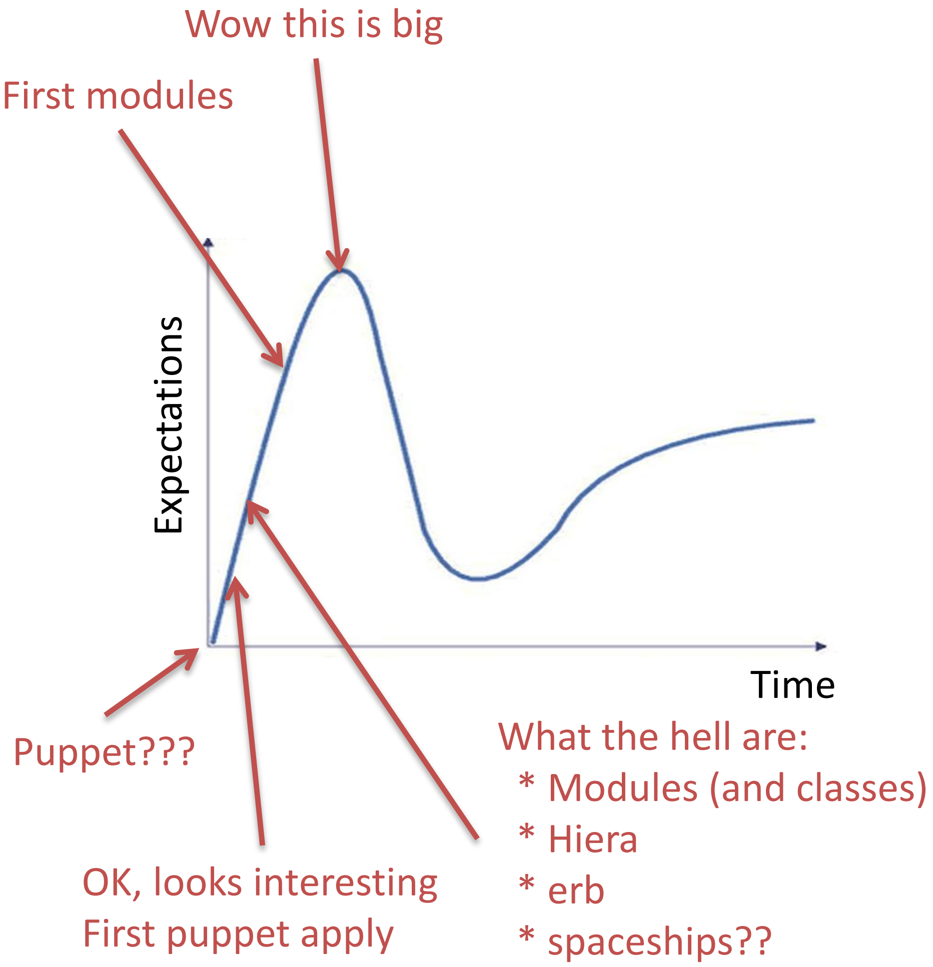
OK, looks interesting
First puppet apply

What the hell are:
 * Modules (and classes)
 * Hiera
 * erb
 * spaceships??

Our feedback on puppet

Puppet is an amazing tool

- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient



Wow this is big

First modules

Puppet???

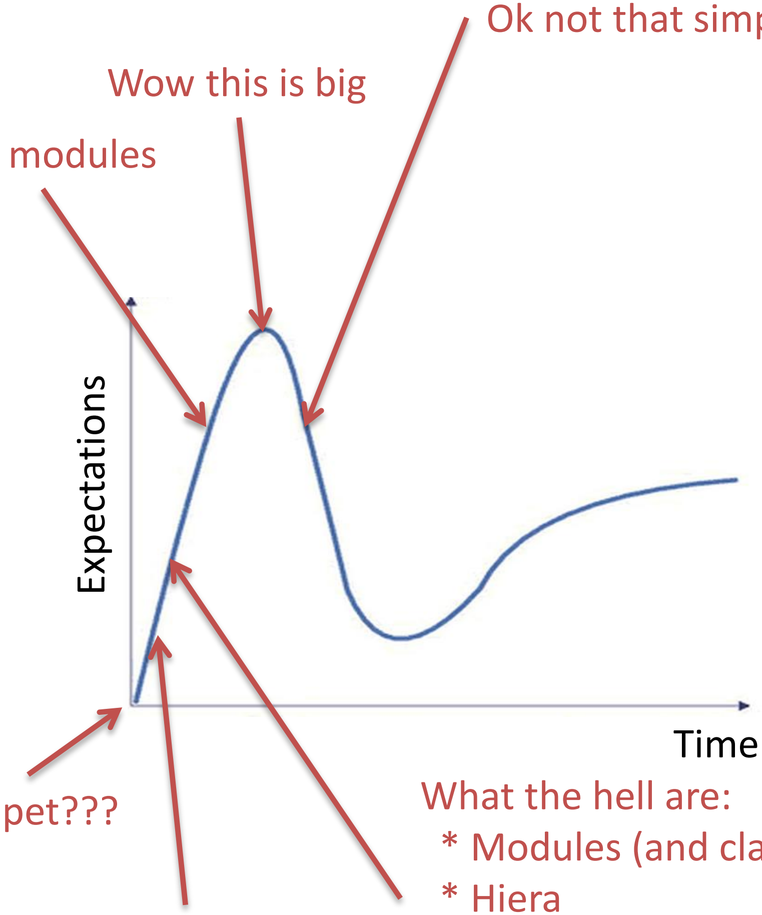
OK, looks interesting
First puppet apply

What the hell are:
* Modules (and classes)
* Hiera
* erb
* spaceships??

Our feedback on puppet

Puppet is an amazing tool

- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient



Ok not that simple

Wow this is big

First modules

Puppet???

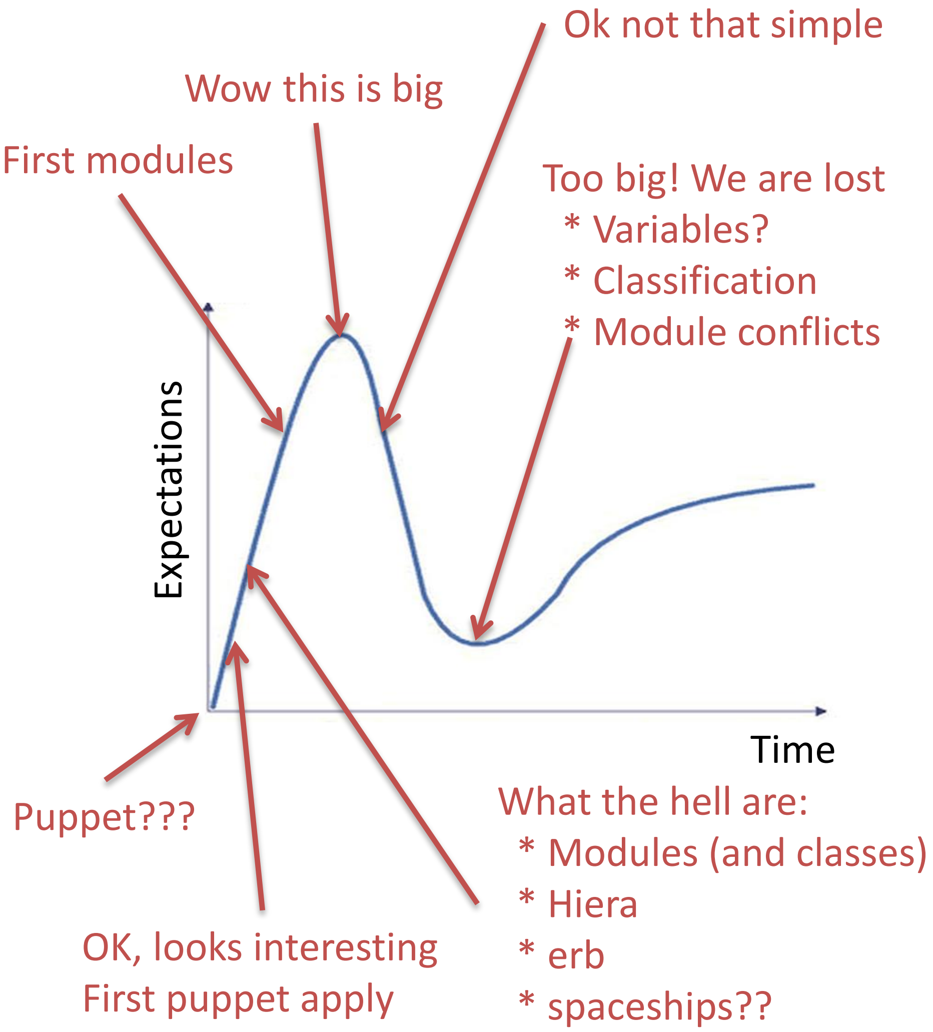
OK, looks interesting
First puppet apply

What the hell are:
* Modules (and classes)
* Hiera
* erb
* spaceships??

Our feedback on puppet

Puppet is an amazing tool

- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient



Ok not that simple

Wow this is big

First modules

Too big! We are lost
 * Variables?
 * Classification
 * Module conflicts

Puppet???

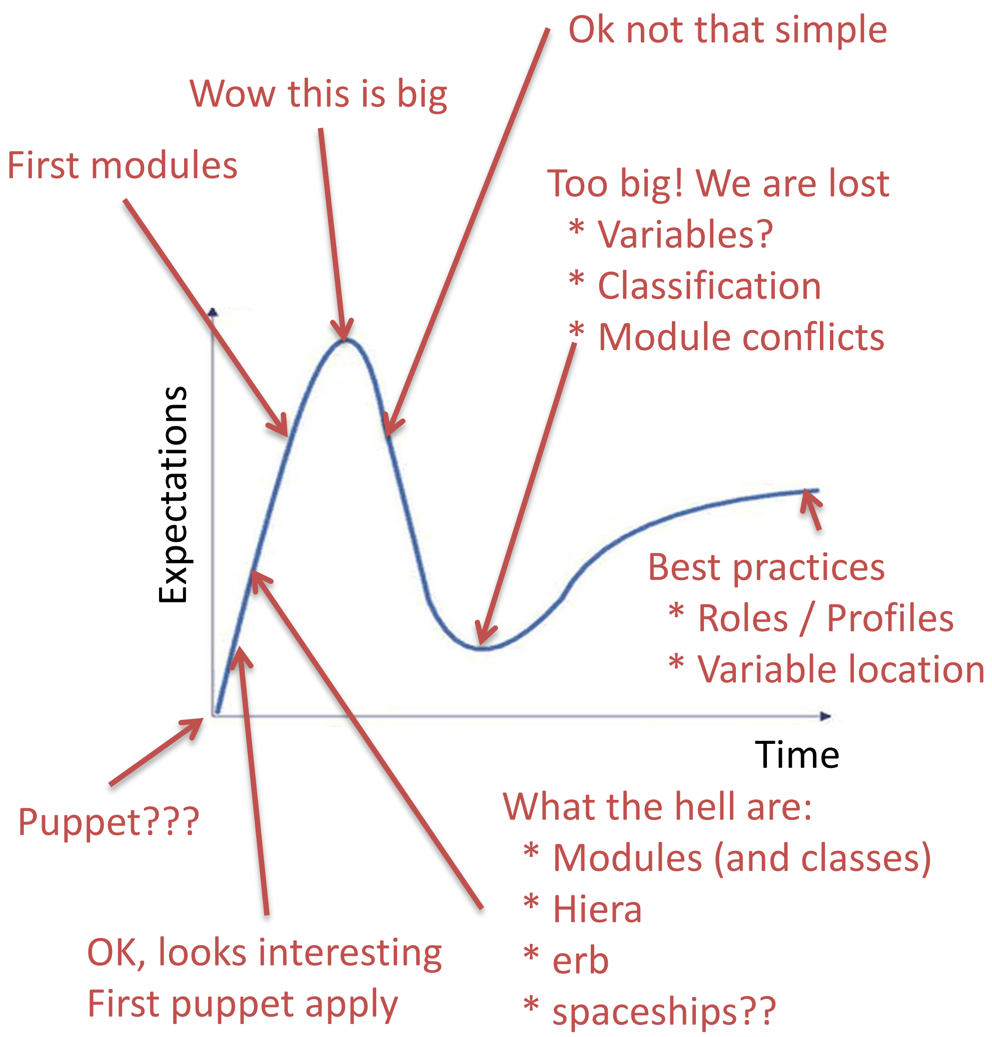
OK, looks interesting
First puppet apply

What the hell are:
 * Modules (and classes)
 * Hiera
 * erb
 * spaceships??

Our feedback on puppet

Puppet is an amazing tool

- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient



Puppet???

OK, looks interesting
First puppet apply

What the hell are:
 * Modules (and classes)
 * Hiera
 * erb
 * spaceships??

First modules

Wow this is big

Ok not that simple

Too big! We are lost
 * Variables?
 * Classification
 * Module conflicts

Best practices
 * Roles / Profiles
 * Variable location

Our feedback on puppet

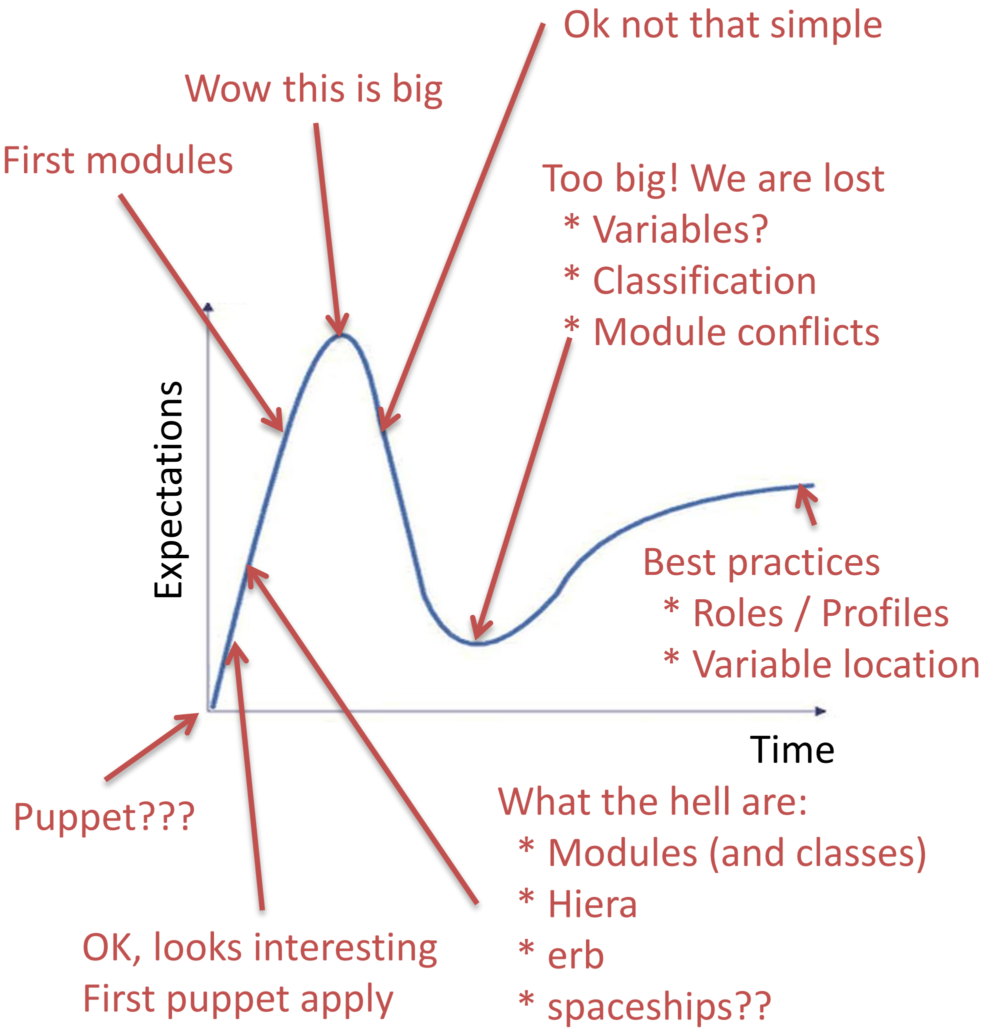
Puppet is an amazing tool

- Automate configuration
- Declare state, keep configuration on track
- Puppet syntax is very expressive
- Variable management with hiera is very efficient

You can do (almost) anything with puppet, but

- Setups can be complex
- Many solutions to a problem
- Use it for what it does best

  Try adapting processes first
- Look for best practices



Conclusion

The pace of innovation in IT is accelerating

New time-to-market challenges will require continuous delivery

We will not get continuous delivery without DEVOPS

Puppet is an amazing DEVOPS tool and will help you

Conclusion

The pace of innovation in IT is accelerating

New time-to-market challenges will require continuous delivery

We will not get continuous delivery without DEVOPS

Puppet is an amazing DEVOPS tool and will help you

But **tools** cannot do everything: puppet is not a magic solution

Conclusion

The pace of innovation in IT is accelerating

New time-to-market challenges will require continuous delivery

We will not get continuous delivery without DEVOPS

Puppet is an amazing DEVOPS tool and will help you

But **tools** cannot do everything: puppet is not a magic solution

- Finding the **best way to use puppet** for you will take time
- Providing a **configuration service** will be a challenge
- **Processes** will need to change

DEV and OPS roles are evolving and **Organizations** will need to adapt

Thank you

@D2SI

#PuppetCampParis

puppet camp

Paris 2014

Laurent Bernaille, D2SI

@lbernail