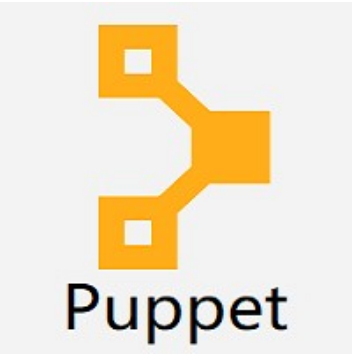
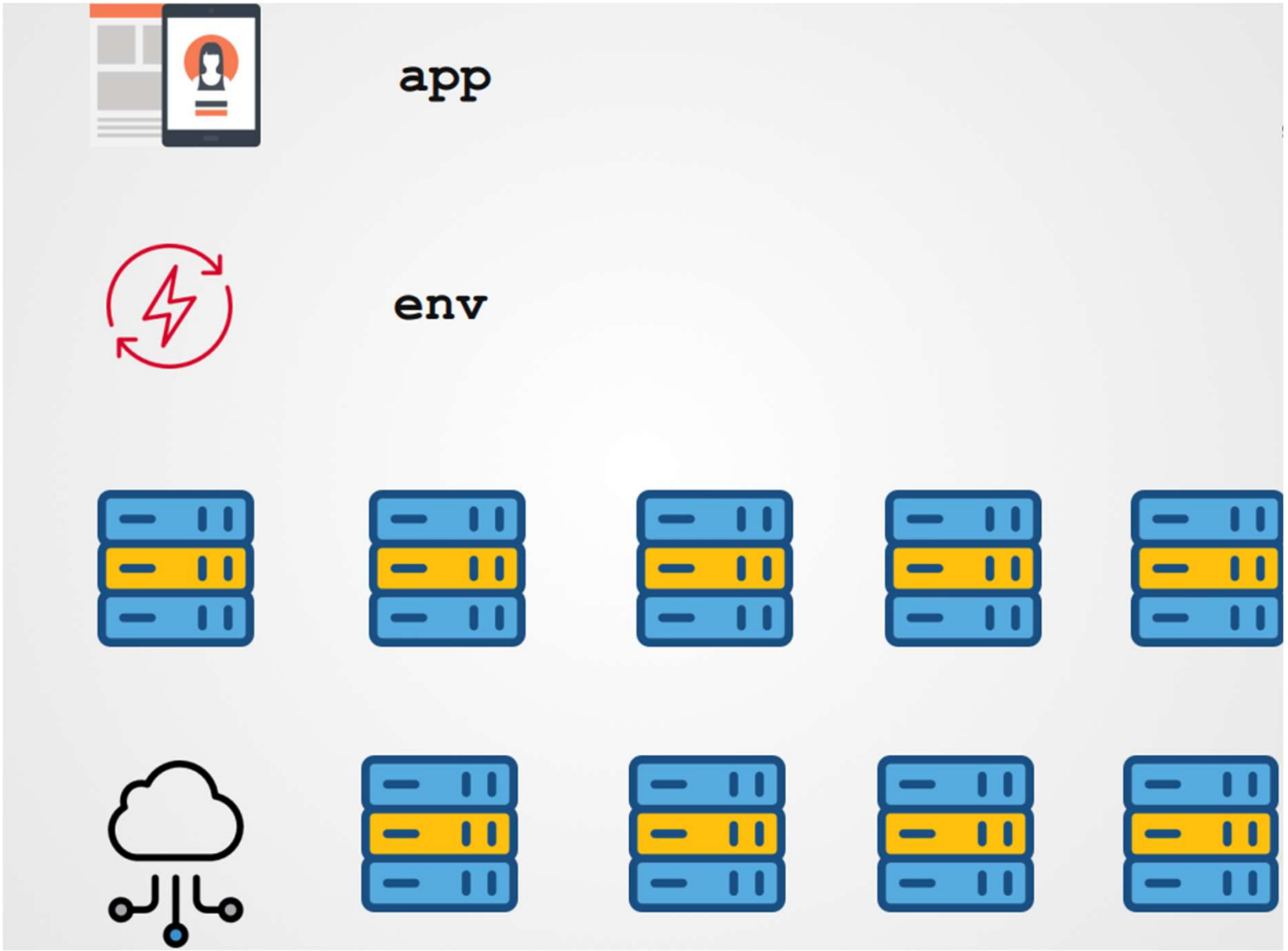
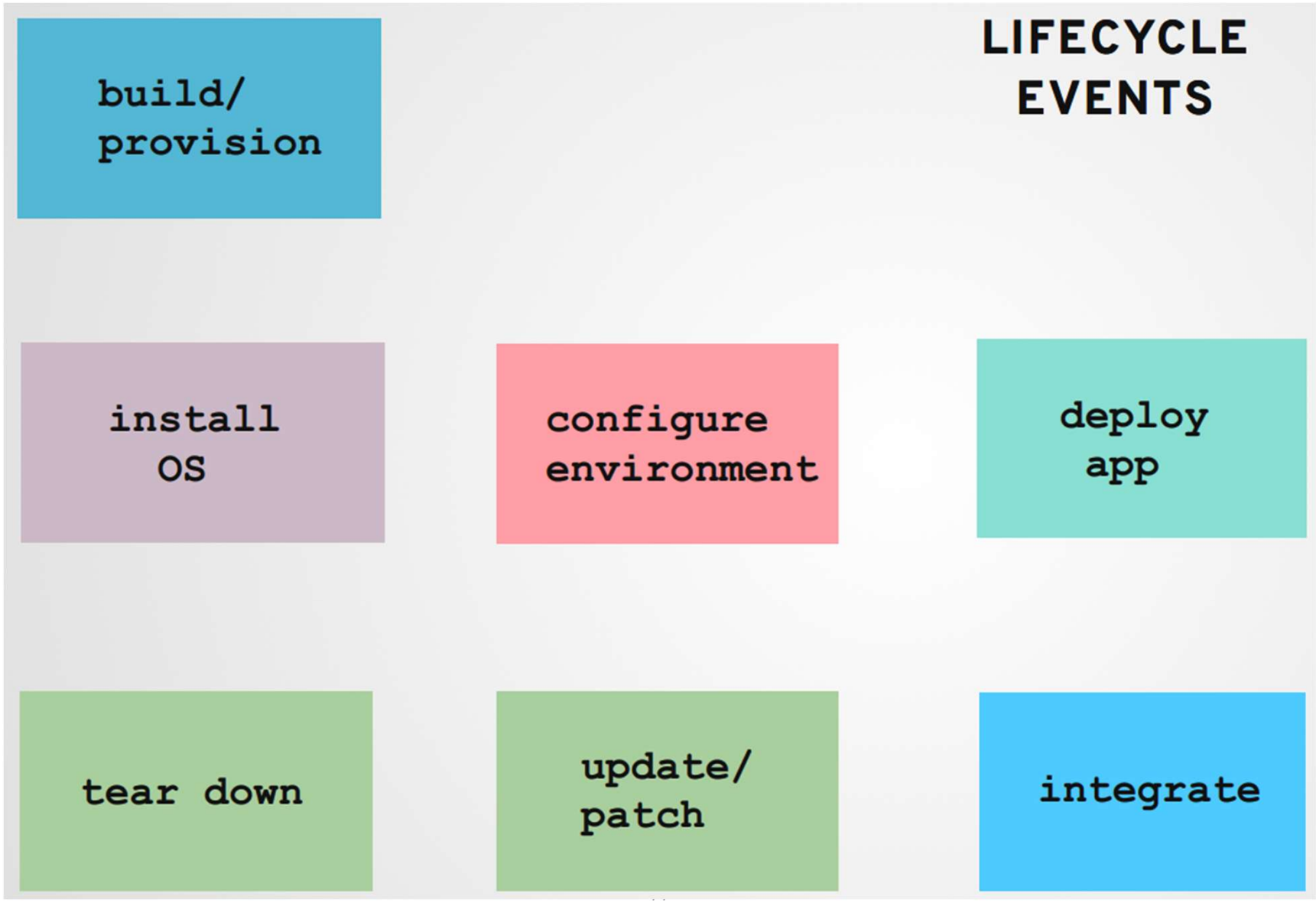


Puppet


app
env


LIFECYCLE EVENTS
build/ provision
install OS
configure environment
deploy app
tear down
update/ patch
integrate

APPROACHES

Manual

Scripts

Golden Images

IaaC

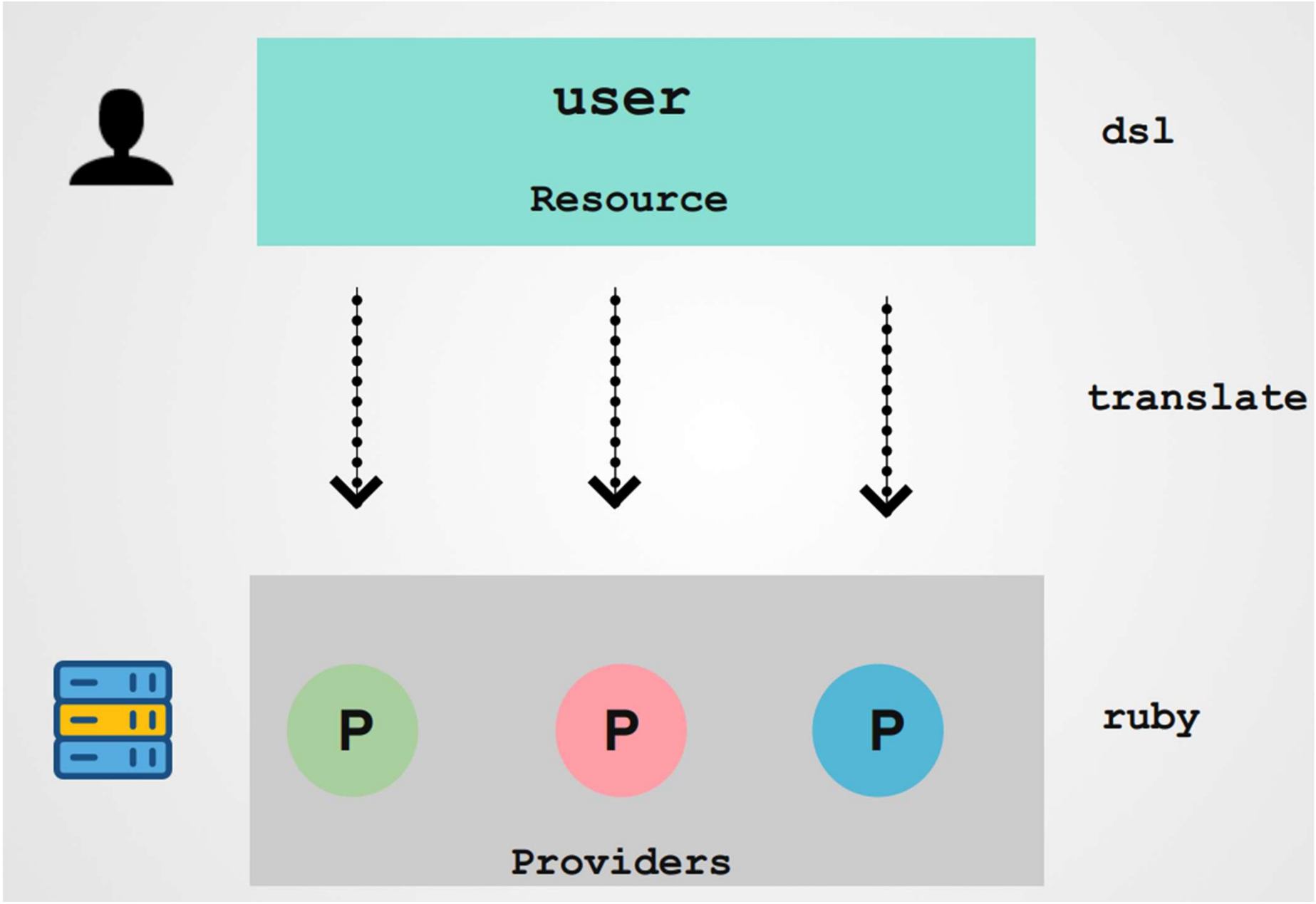
Scripts

- Automate repeatable tasks
- Procedural Programs
- Focus on HOW

IaaC

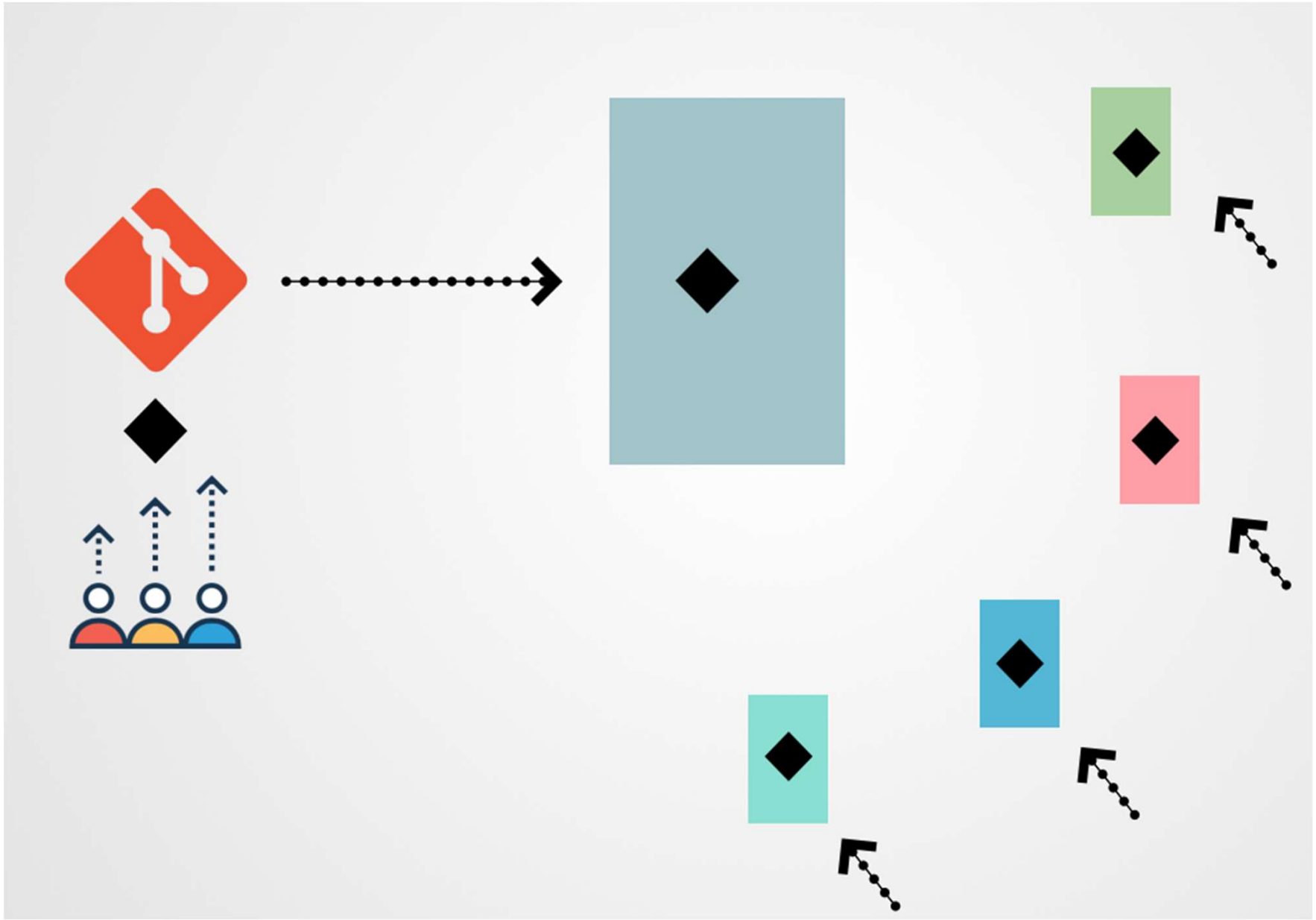
```
user
   name = xyz
   uid  = 5001
   pass = xxx
```

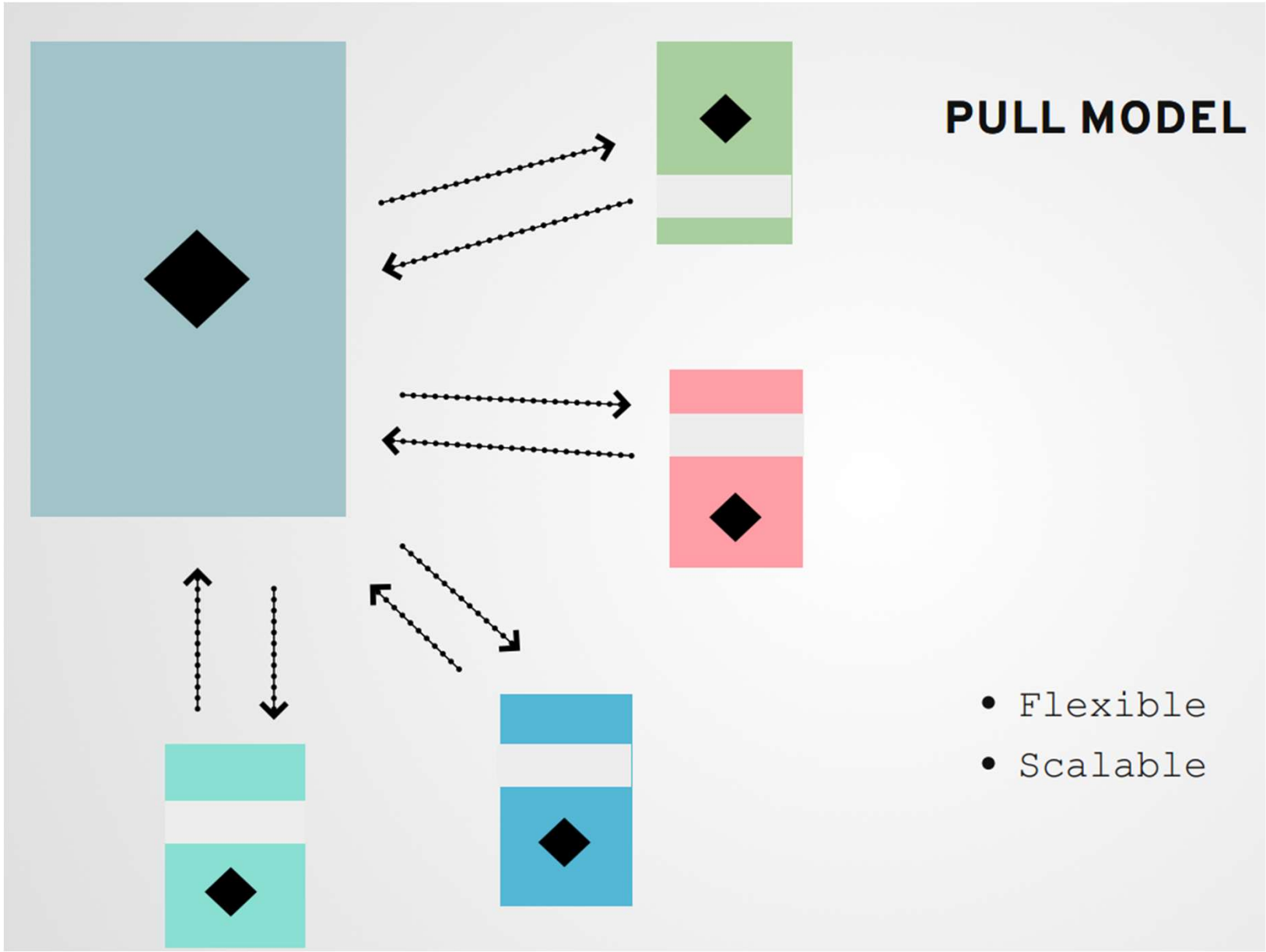
Resource

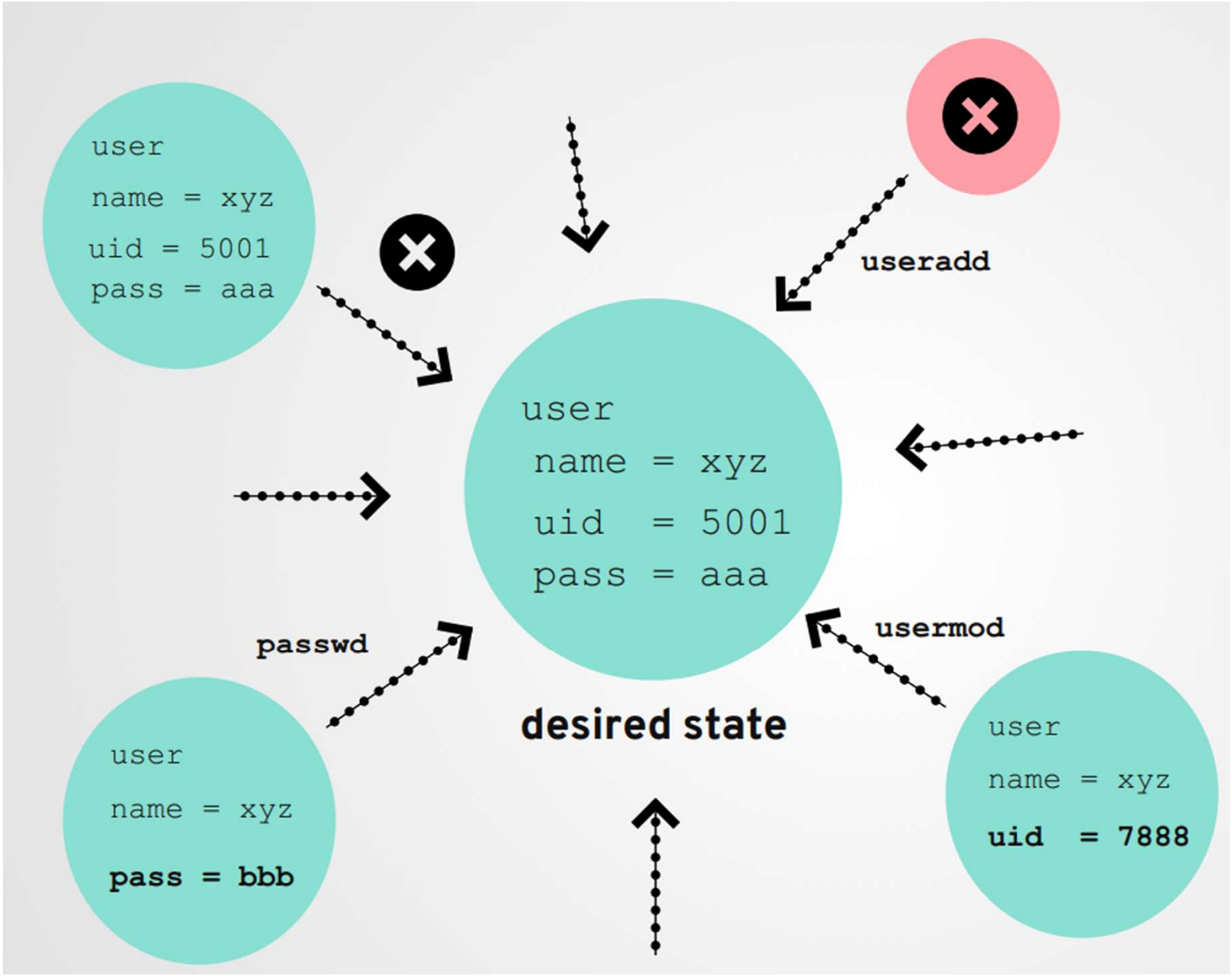

user
Resource
dsl
translate
P
P
P
Providers
ruby

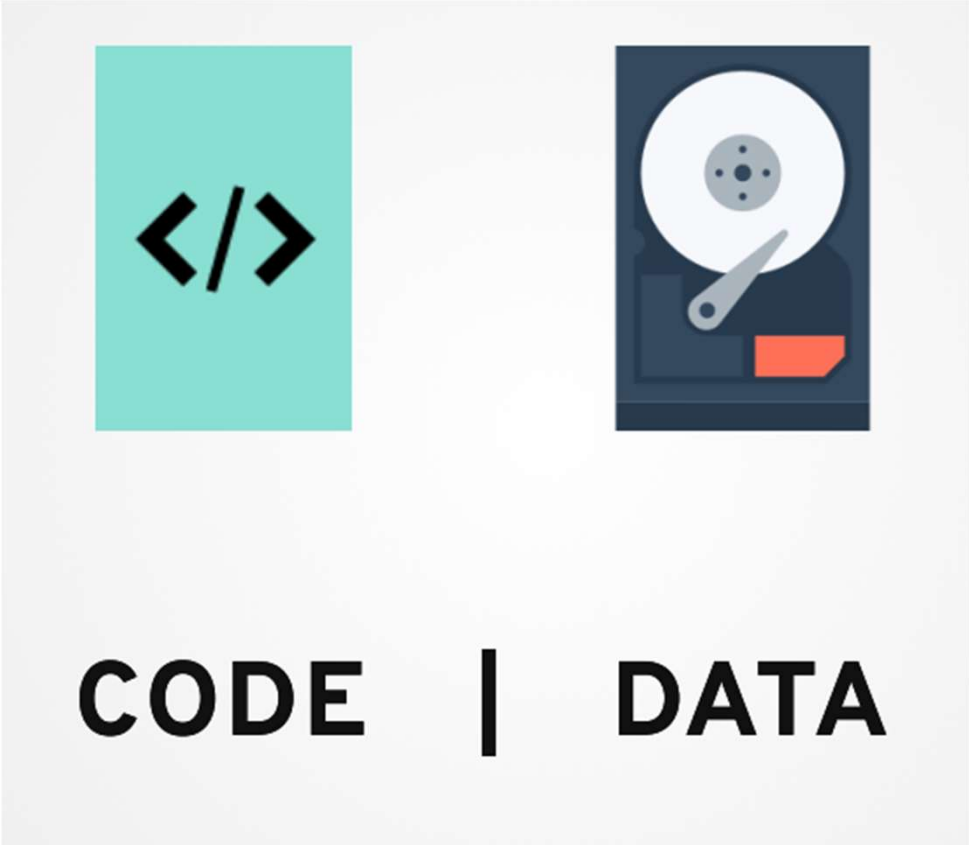
IaaC

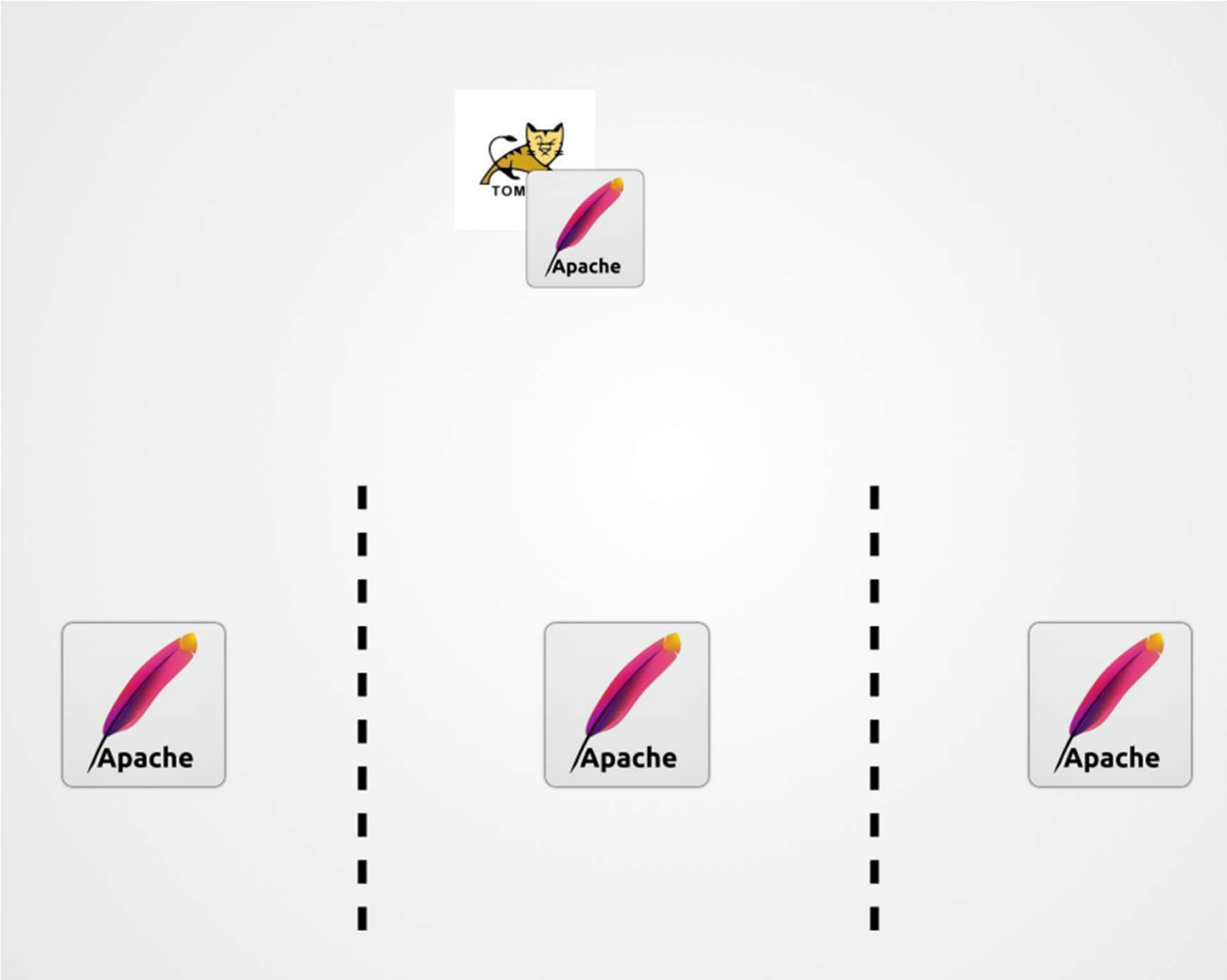
- Declartive Approach
- Revision Control
- Recreate the Infrastructure out of code repo
- Migration and DR
- Absolute Consistency

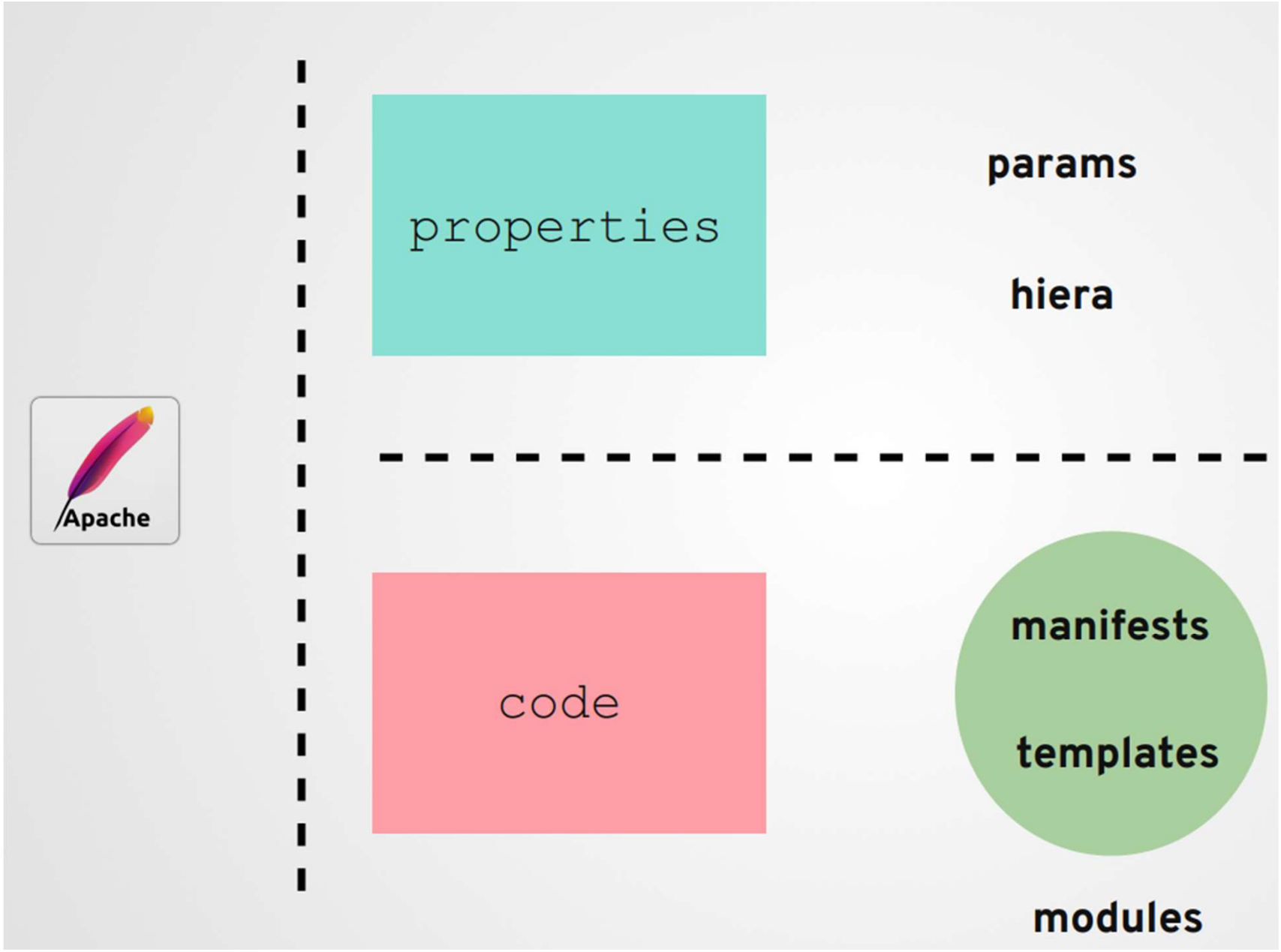


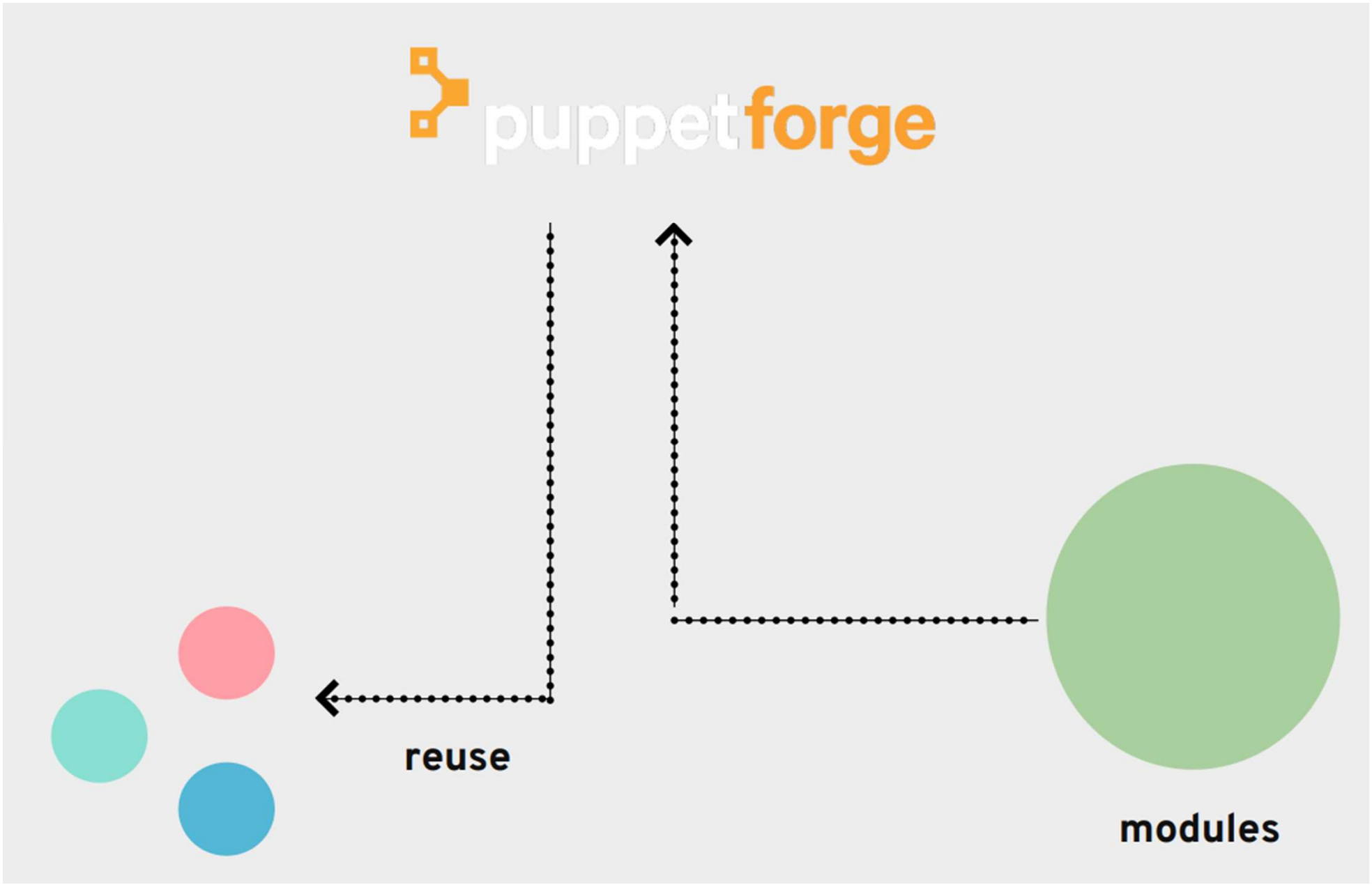

PULL MODEL
• Flexible
• Scalable


user
name = xyz
uid = 5001
pass = aaa
useradd
user
name = xyz
uid = 5001
pass = aaa
passwd
usermod
desired state
user
name = xyz
pass = bbb
user
name = xyz
uid = 7888


CODE | DATA

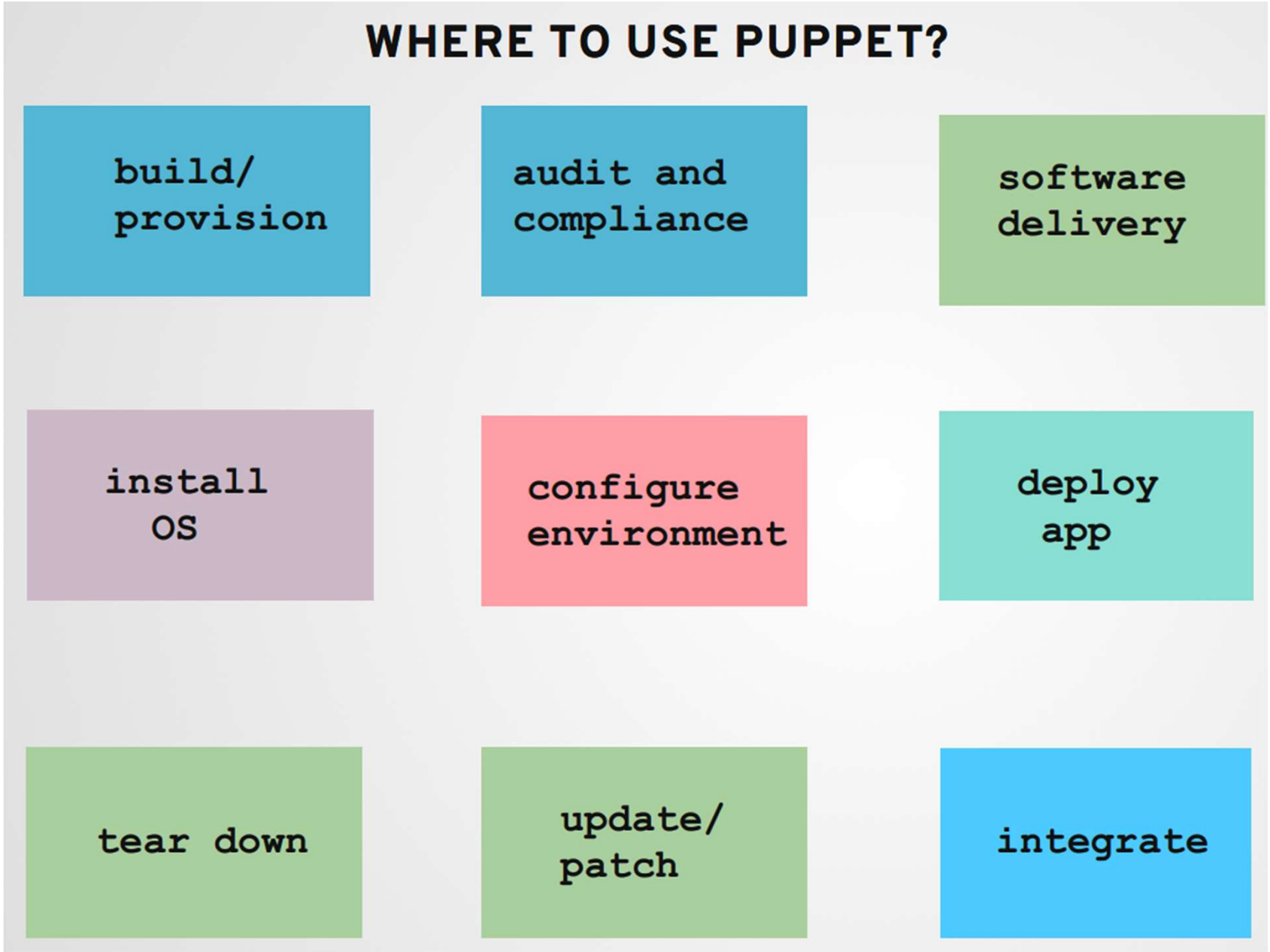

TOM
Apache
Apache
Apache
Apache


properties
params
hiera
Apache
code
manifests
templates
modules


puppetforge
reuse
modules


ITERATIVE APPROACH TO
AUTOMATION

WHERE TO USE PUPPET?



WHO IS IT FOR

systems administrator

application ops/devops

build and release engg

network engineer

storage admin

developer

WHAT IT IS NOT

UI Oriented Tool

General Purpose Testing Tool

agent less system

SCM Tool

CI Tool

one stop devops solution

Puppet vs Chef vs Ansible vs Terraform

	Chef	Puppet	Ansible	CloudFormation	TerraForm
Code	Open Source	Open Source	Open Source	Closed Source	Open Source
Syntax	Ruby	Ruby	YAML	JSON	HCL
Type	Configuration Management	Configuration Management	Configuration Management	Orchestration	Orchestration
Infrastructure	Mutable	Mutable	Mutable	Immutable	Immutable
Language	Procedural	Declarative	Procedural	Declarative	Declarative
Architecture	Client/Server	Client/Server	Client Only	Client Only	Client Only
State Management			Yes	No	Yes
Execution Control	Yes, via Chef Server	Yes, via Puppet Master	No. Any computer can be a controller	No	Yes
Cloud	All	All	All	AWS only	All

Puppet System Sizing for standard installations

Node volume	Cores	RAM	/opt/	/var/
Trial use	2	8 GB	20 GB	24 GB
11–100	6	10 GB	50 GB	24 GB
101–500	8	12 GB	50 GB	24 GB
501–1,000	10	16 GB	50 GB	24 GB
1,000–2,500	12	24 GB	50 GB	24 GB

Puppet System Sizing for large installations

Node volume	Node	Cores	RAM	/opt/	/var/	EC2
2,500–20,000	Primary node	16	32 GB	150 GB	10 GB	c5.4xlarge
	Each compiler (1,500 - 3,000 nodes)	6	12 GB	30 GB	2 GB	m5.xlarge

Puppet System Sizing for extra-large installations

Node volume	Node	Cores	RAM	/opt/	/var/	EC2
20,000+	Primary node	16	32 GB	150 GB	10 GB	c5.4xlarge
	Each compiler (1,500 - 3,000 nodes)	6	12 GB	30 GB	2 GB	m5.xlarge
	PE-PostgreSQL node	16	128 GB	300 GB	4 GB	r5.4xlarge

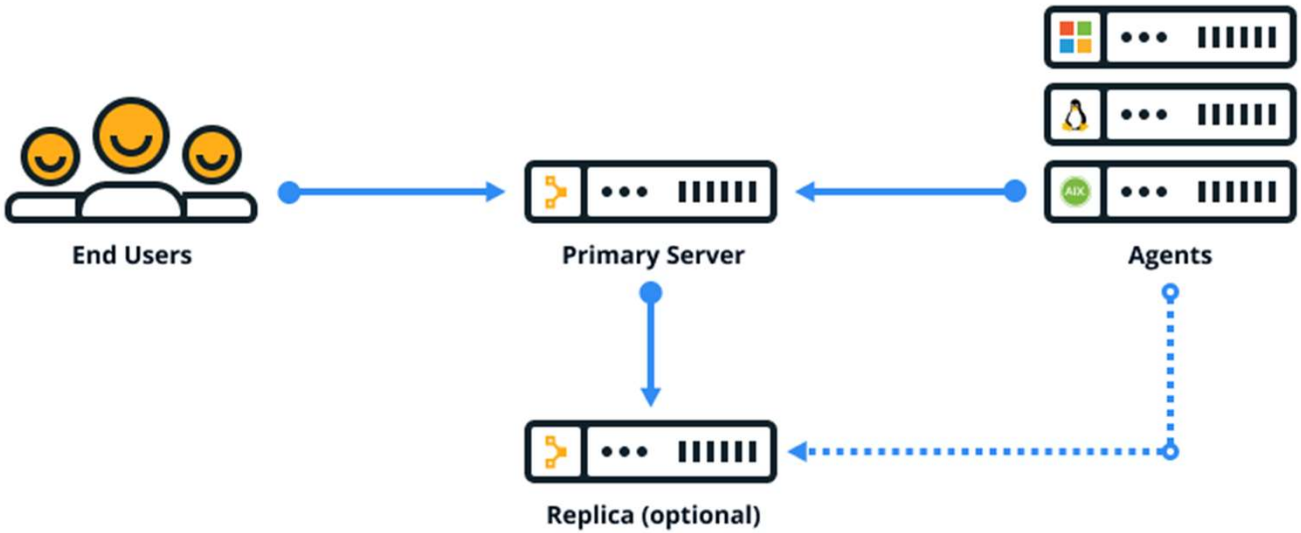
Standard guidelines for Puppet Sizing

- Puppet master scaling depends on a number of variables
  - The number of files pulled during a cycle
  - The frequency of Agent updates of catalog data
  - The complexity of the modules being applied

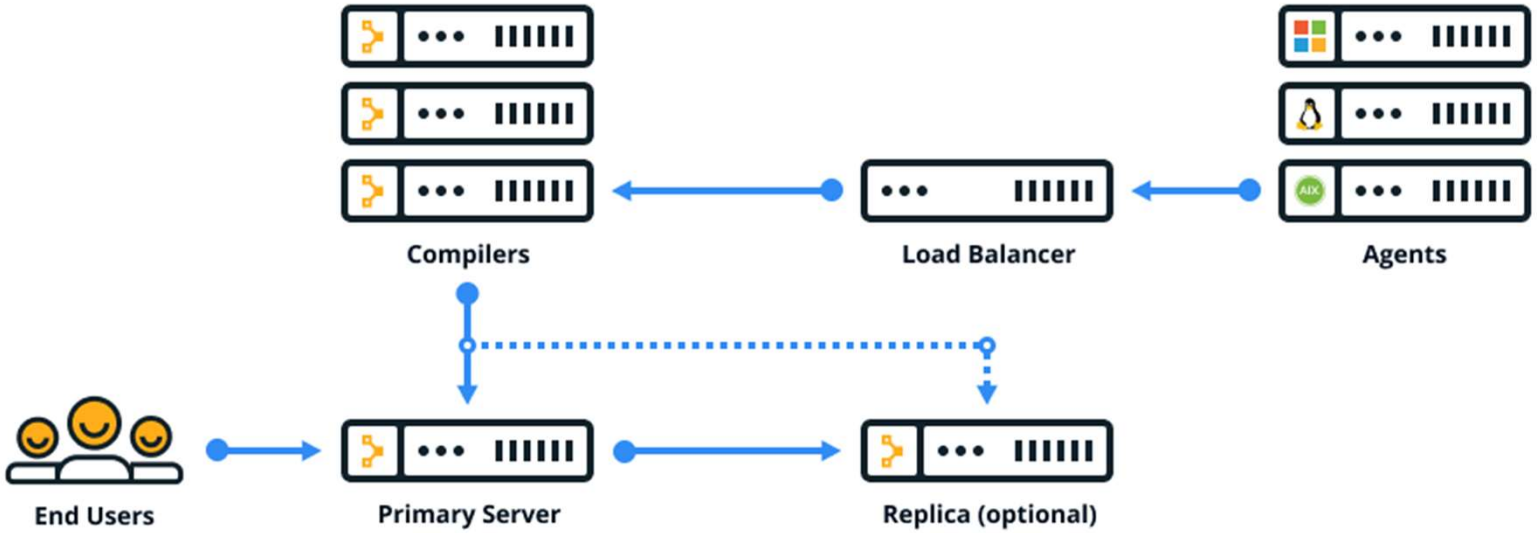
Supported architectures

Configuration	Description	Node limit
Standard installation (Recommended)	All infrastructure components are installed on the primary server. This installation type is the easiest to install, upgrade, and troubleshoot.	Up to 2,500
Large installation	Similar to a standard installation, plus one or more compilers and a load balancer which help distribute the agent catalog compilation workload.	2,500–20,000
Extra-large installation	Similar to a large installation, plus one or more separate PE-PostgreSQL nodes that run PuppetDB.	20,000+

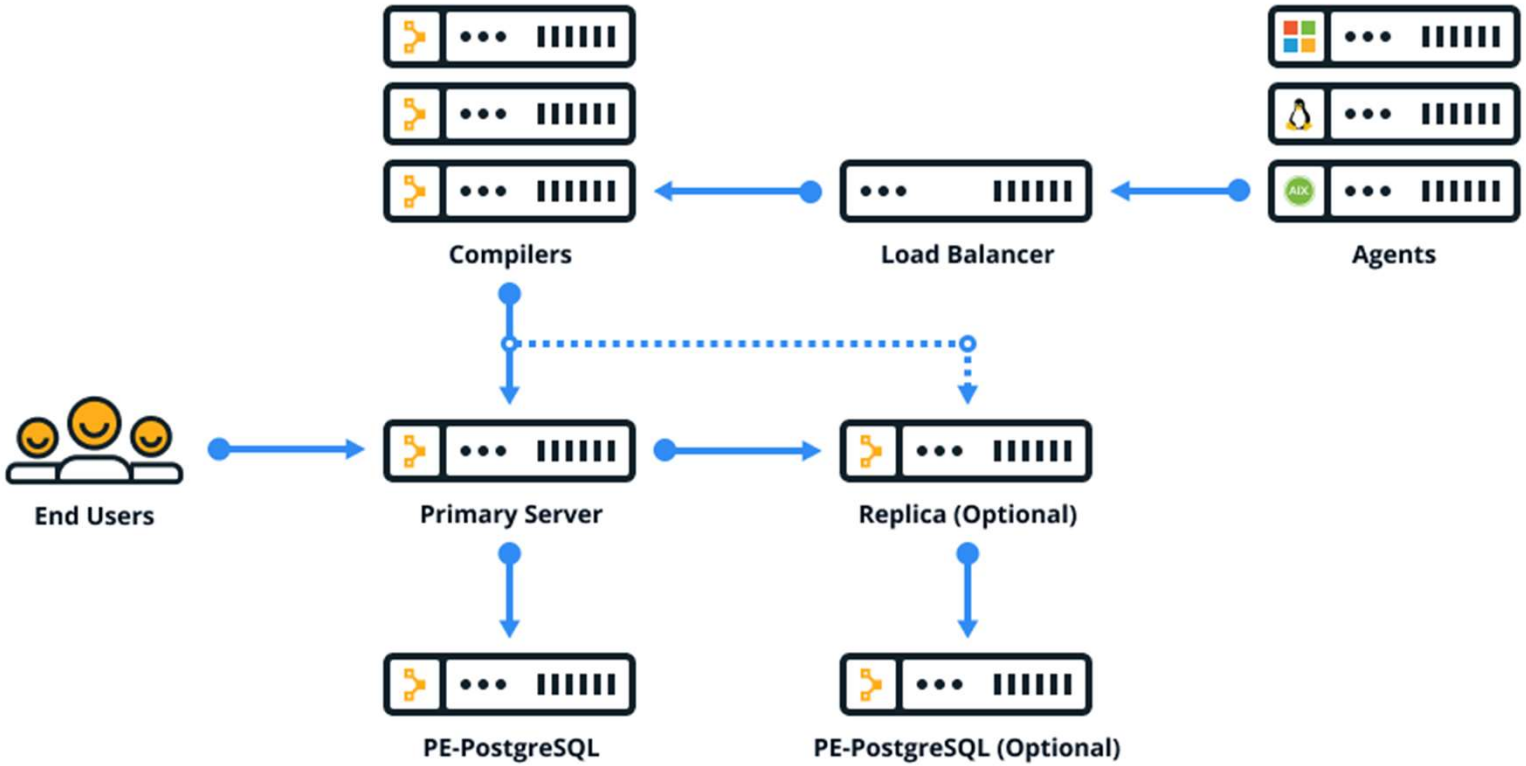
Standard installation



Large installation



Extra-large installation



What is Puppet Compiler?

- A single primary server can process requests and compile code for up to 4,000 nodes
- Expand infrastructure by adding compilers to
  - Share the workload and
  - Compile catalogs faster


thanks