What is DevOps

Background & History

DevOps = Development & Operations

- **NOT** Development **VS** Operations
- **NOT** Development **OR** Operations

Concept went mainstream in 2009 (Velocity Conference)

- John Allspaw and Paul Hammond Presentation
- 10+ Deploys Per Day: Dev and Ops Cooperation at Flickr

What is the definition of DevOps?

DevOps is a practice that unifies people, process and technology across development and IT in five core practices: planning and tracking, development, build and test, delivery and monitoring and operations. When practicing DevOps, development, IT operations, quality engineering and security teams work closely together—breaking down practices that were once siloed.

Improved coordination and collaboration across these disciplines reduces the time between when a change is committed to a system and when the change being placed into production. And, it ensures that standards for security and reliability are met as part of the process.

The result: better products, delivered faster, to happier customers.

Competing Goals

BUSINESS

Respond to the rapidly changing competitive landscape

Provide stable, reliable, and secure service to the customer

Happy Customers

DEVELOPMENT

New Products
Add new features
Security Updates
Bug Fixes
* Introduce Change

Happy Customers

OPERATIONS

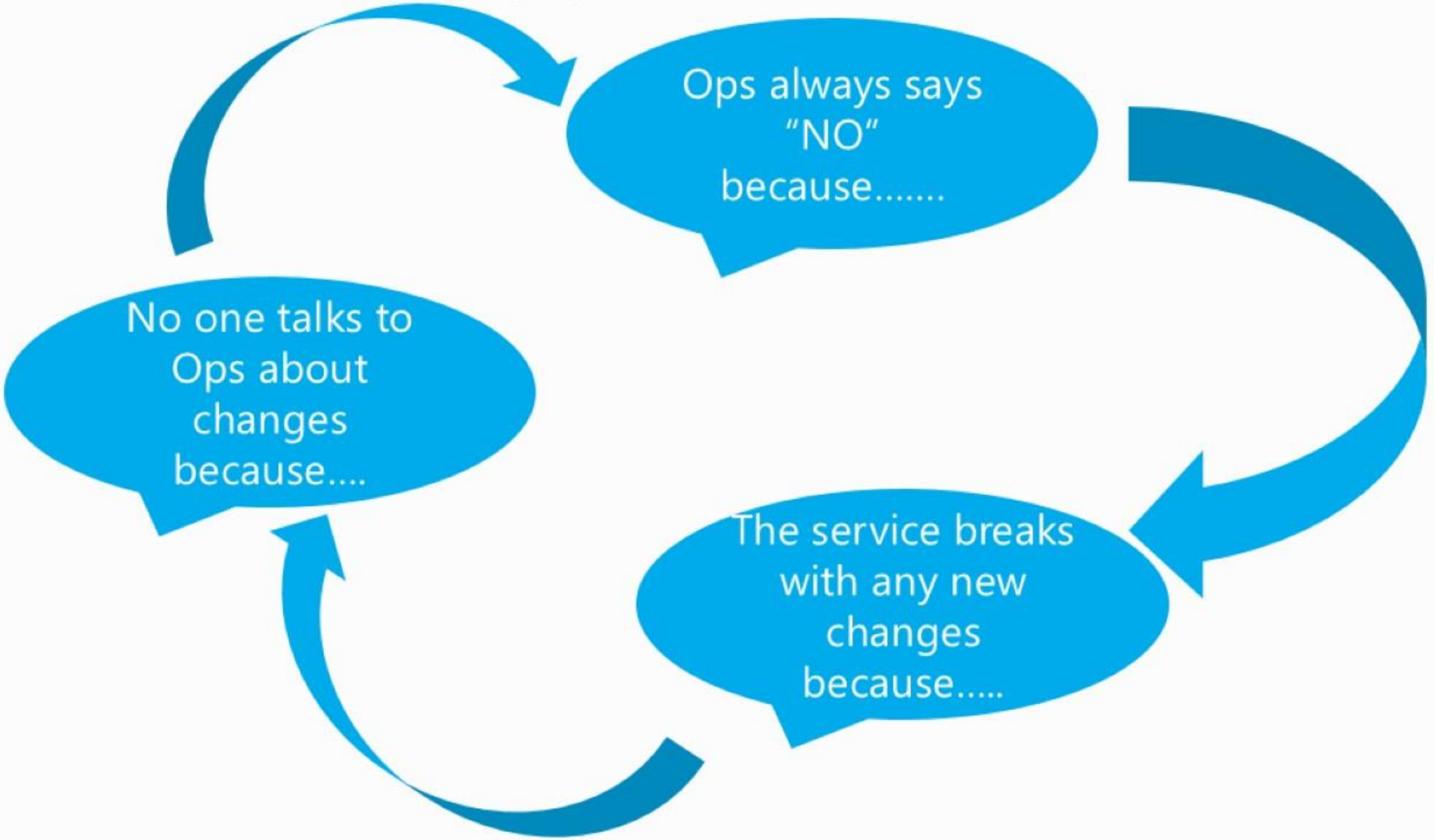
Keep service stable and fast
- Maximize Uptime
* Minimize Changes

Happy Customers

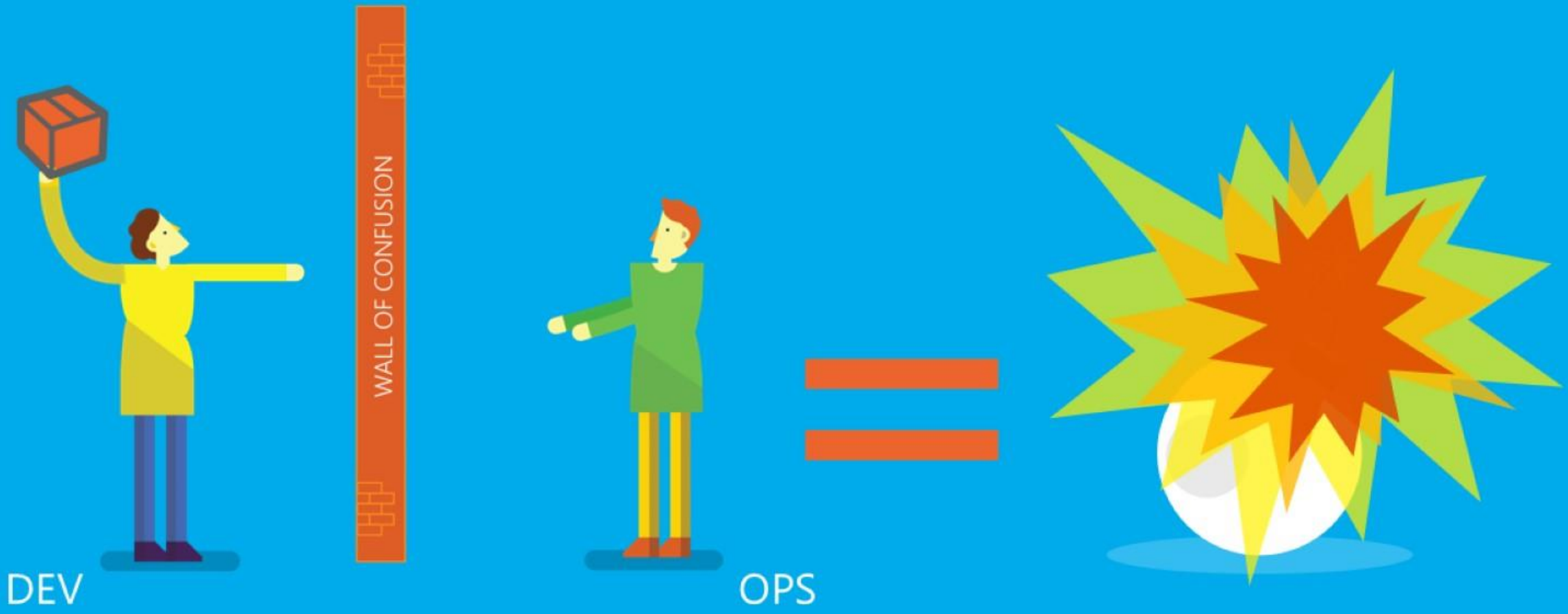
(*Historic*) Stereotypes

DEVELOPMENT	OPERATIONS
It's not my code, it's your machines!	It's not my machines, it's your code!
Ops guy = grumpy old man who always says no!	Assumes all code changes will wreak havoc
Finger-pointy	Finger-pointy

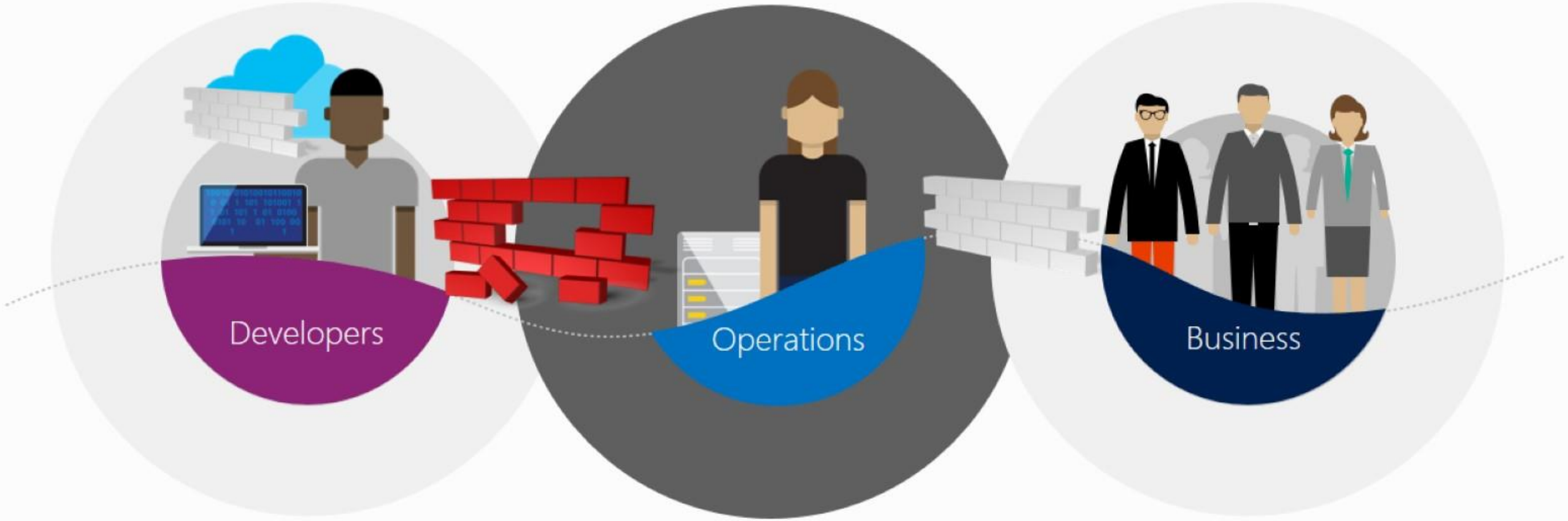
(*Historic*) Stereotypes



(*Historic*) Stereotypes



Consequences of inefficiency



Partial Agility

Downtime Costs

Rework

What DevOps is NOT *(common misconceptions)*

Tool or Set of Tools

Chef, Puppet, Jenkins, Azure ≠ DevOps

DevOps is only for Startups or 'Cloud' companies

DevOps is only for Open Source Software

DevOps replaces Agile

Ops resources need to learn how to code

DevOps is only for cloud deployments

It works for small deployments but ours is complex

What is DevOps

DevOps integrates development and operations teams in order to improve collaboration and productivity by:

- Automating Infrastructure
- Automating Workflows
- Continuously Measuring Service Performance

Core Components of DevOps

People

Process

Tools

Core Components of DevOps (People)

Trust

Don't just say no

Providing devs with access to environment (read-only)

Dev – talk to ops upfront – what are the impacts

Don't hide ideas, failures, features

Trust that everyone is doing their best for the business

Respect

Devs – remember someone will be woken up when something fails/code breaks

No finger pointing

Healthy attitude about failure

Expertise / background / ideas / suggestions

Communication

Don't hide ideas, failures, features

Don't just say no

Provide constructive feedback on current aches and pains

Core Components of DevOps (Process)

- Infrastructure as Code (IaC)
- Continuous Integration
- Automated Testing
- Continuous Deployment
- Release Management
- App Performance Monitoring
- Load Testing & Auto-Scale

- Availability Monitoring
- Change/Config Management
- Feature Flags
- Automated Environment De-Provisioning
- Self Service Environments
- Automated Recovery (Rollback & Roll-Forward)

http://www.itproguy.com/devops-practices/

Core Components of DevOps (Tools)

Automated Infrastructure

Shared version control

One step build

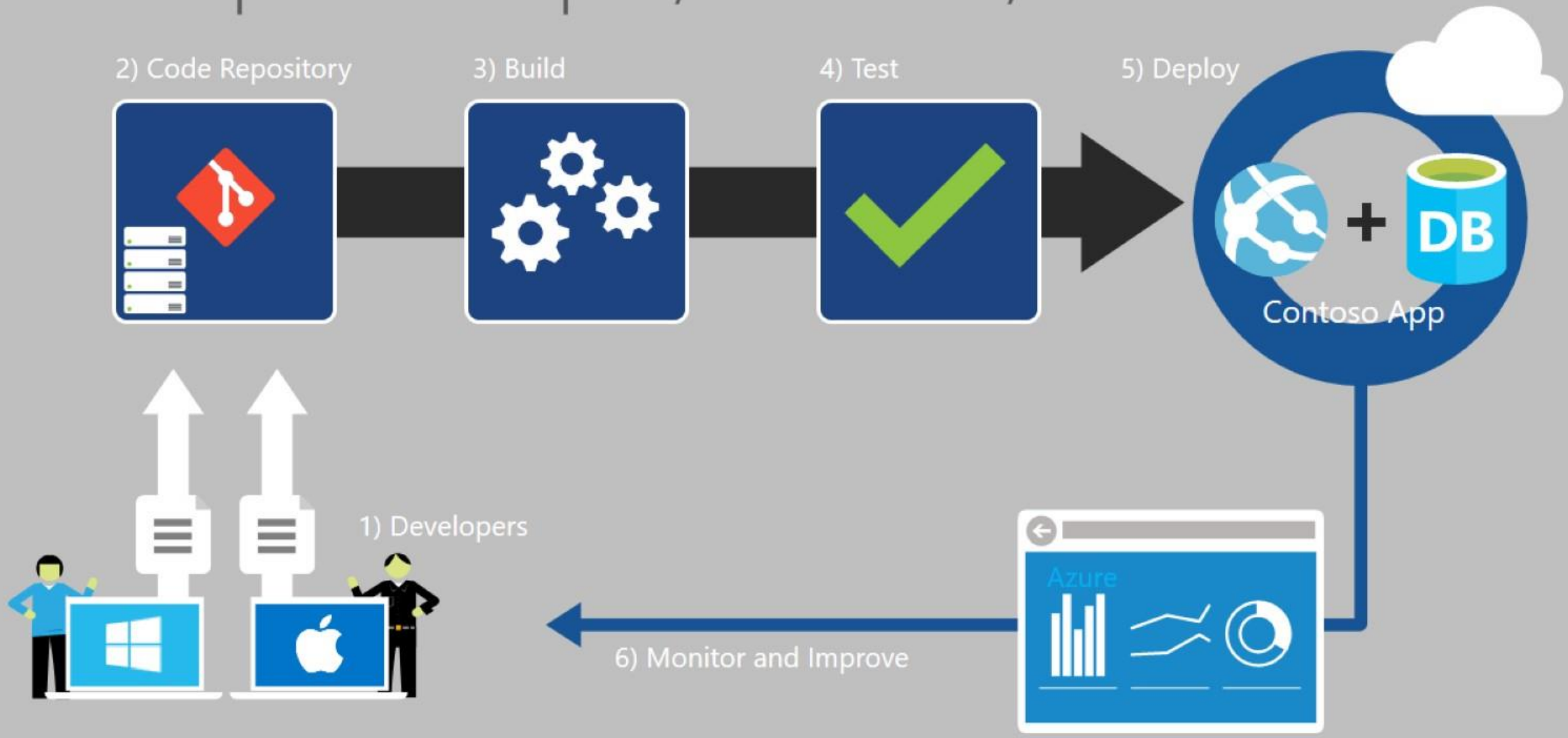
One step deploy

One step build and deploy

Who? When? What?

Tools

DevOps – People, Process, Tools



Benefits of DevOps

Shorten Cycle Times

Speed up and increase traceability of each release by empowering your development and operations teams with advanced collaboration and automation tools.

Optimize Resources

Efficiently manage environments using technologies that support self-service provisioning in a secure way, in line with your IT governance standards.

Improve Quality and Availability

Capture rich telemetry on application performance and usage so that you can make better decisions on future investments and anticipate issues in production before they impact your service.

Infrastructure as Code

What is Infrastructure as Code

"The process of managing and provisioning computing infrastructure and their configuration through machine-processable definition files" - *Wikipedia*

"..the process of standardizing resource configurations and enforcing their state across IT infrastructure in an automated yet agile manner" - *Puppet Labs*

What are the Benefits of IaC

Tools for Infrastructure as Code

Infrastructure Provisioning

ARM Templates by Microsoft

CloudFormation by AWS

Heat by OpenStack

Terraform by Hashicorp

Azure Resource Manager (ARM) Templates

Used to orchestrate provisioning in Azure

Declarative JSON Templates

Can be deployed:

- In the portal
- PowerShell
- Azure CLI
- Visual Studio (*leverages PowerShell*)

Quickstart Templates (https://azure.microsoft.com/resources/templates)

- A gallery of community supported templates
- Backed by GitHub
- +400 templates available

Tools for Infrastructure as Code

Configuration Management

Azure Automation

Jenkins

SALTSTACK

ANSIBLE

PowerShell DSC

Hudson

CHEF

CFEngine

Vagrant

Azure Automation

Azure PaaS offering

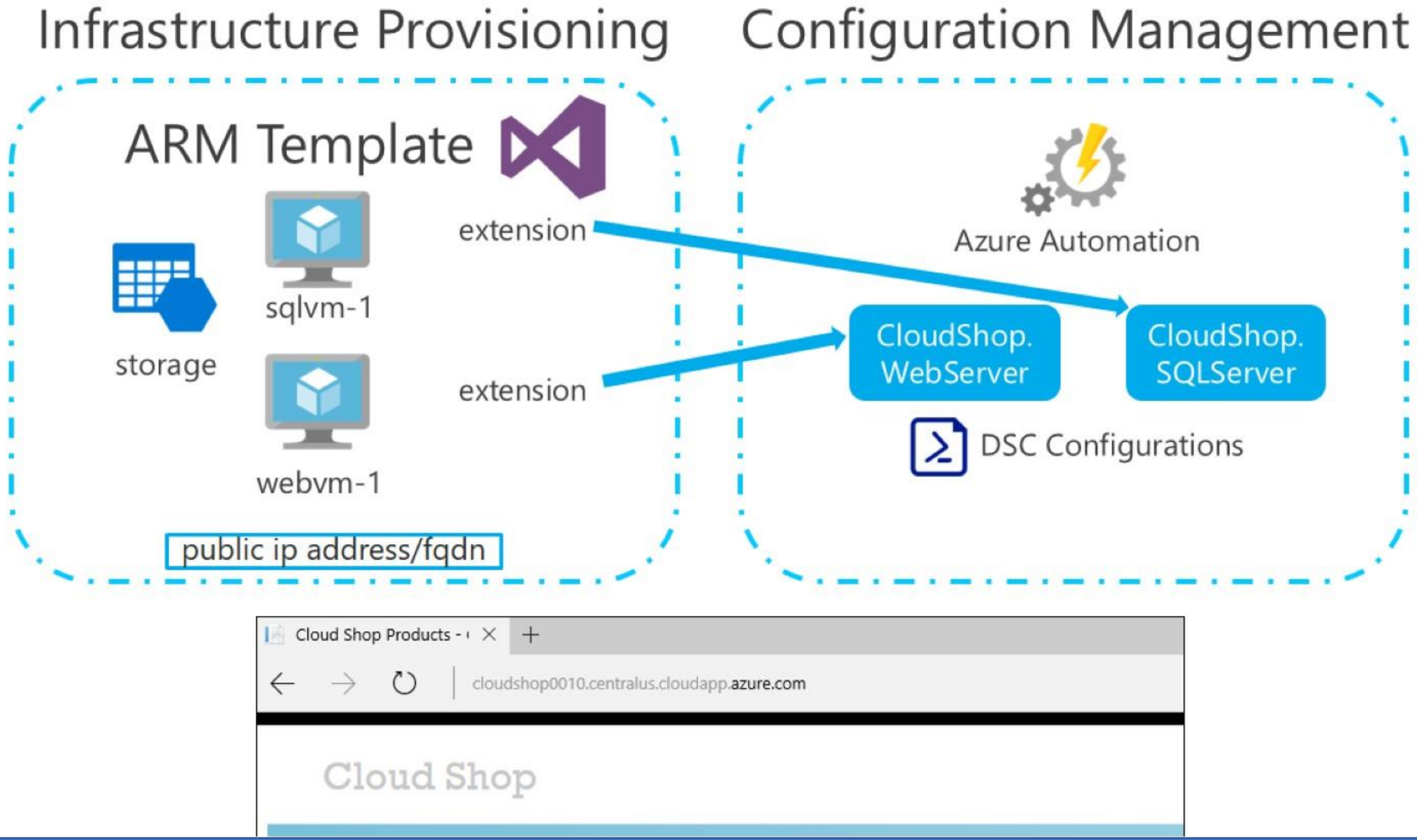
Process Automation

- Gallery of runbooks available
- Create your own runbooks

Configuration Automation

- Built on top of PowerShell DSC
- Apply & Monitor

Provides Reporting


Infrastructure Provisioning
Configuration Management
ARM Template
extension
sqlvm-1
storage
webvm-1
extension
public ip address/fqdn
Azure Automation
CloudShop.
WebServer
CloudShop.
SQLServer
DSC Configurations

Cloud Shop Products -
cloudshop0010.centralus.cloudapp.azure.com
Cloud Shop