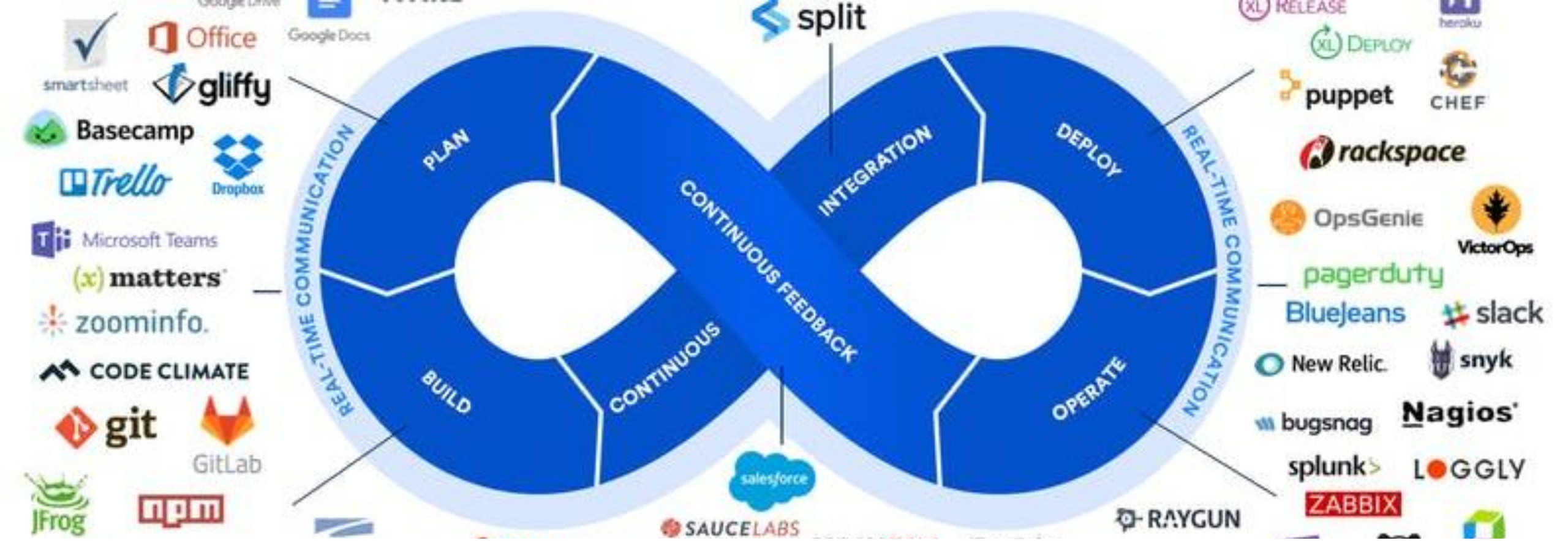
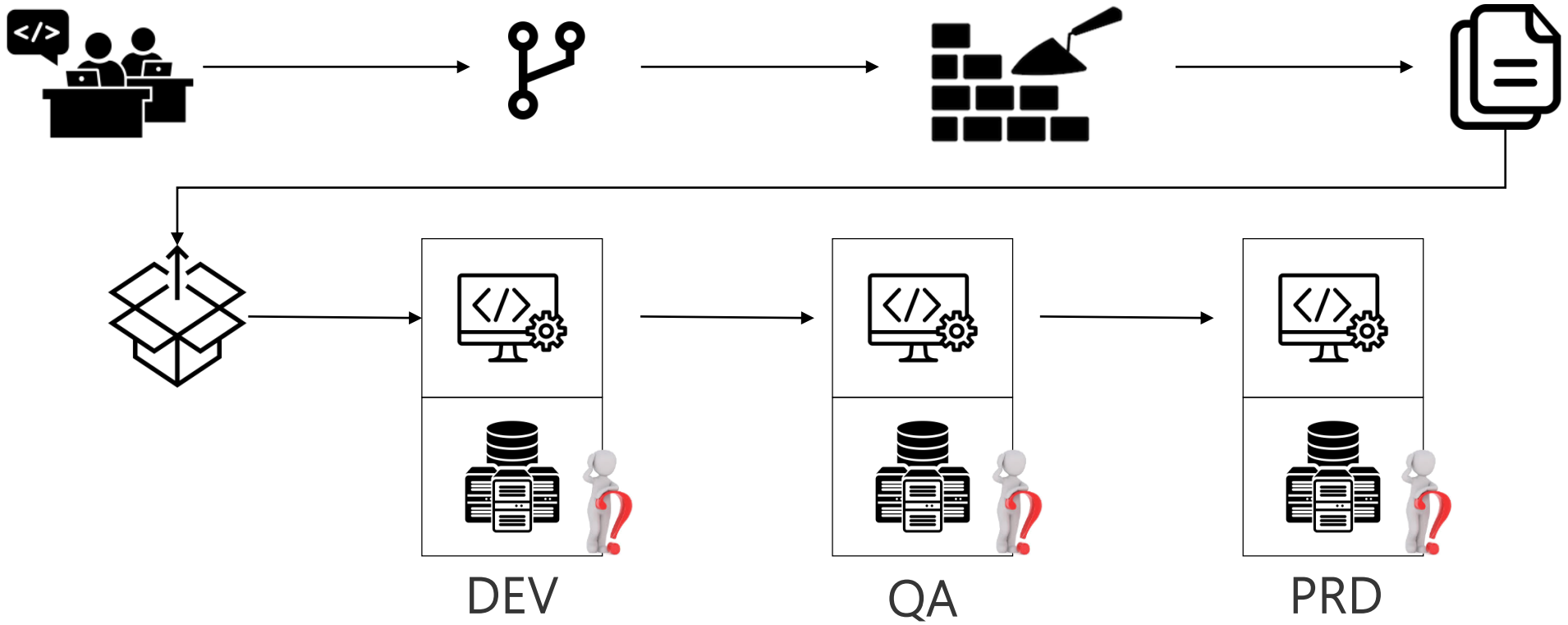


Infrastructure as code with Azure DevOps

Martin Schvartzman
martin.schvartzman@microsoft.com
@martin77s

Application Development Approach



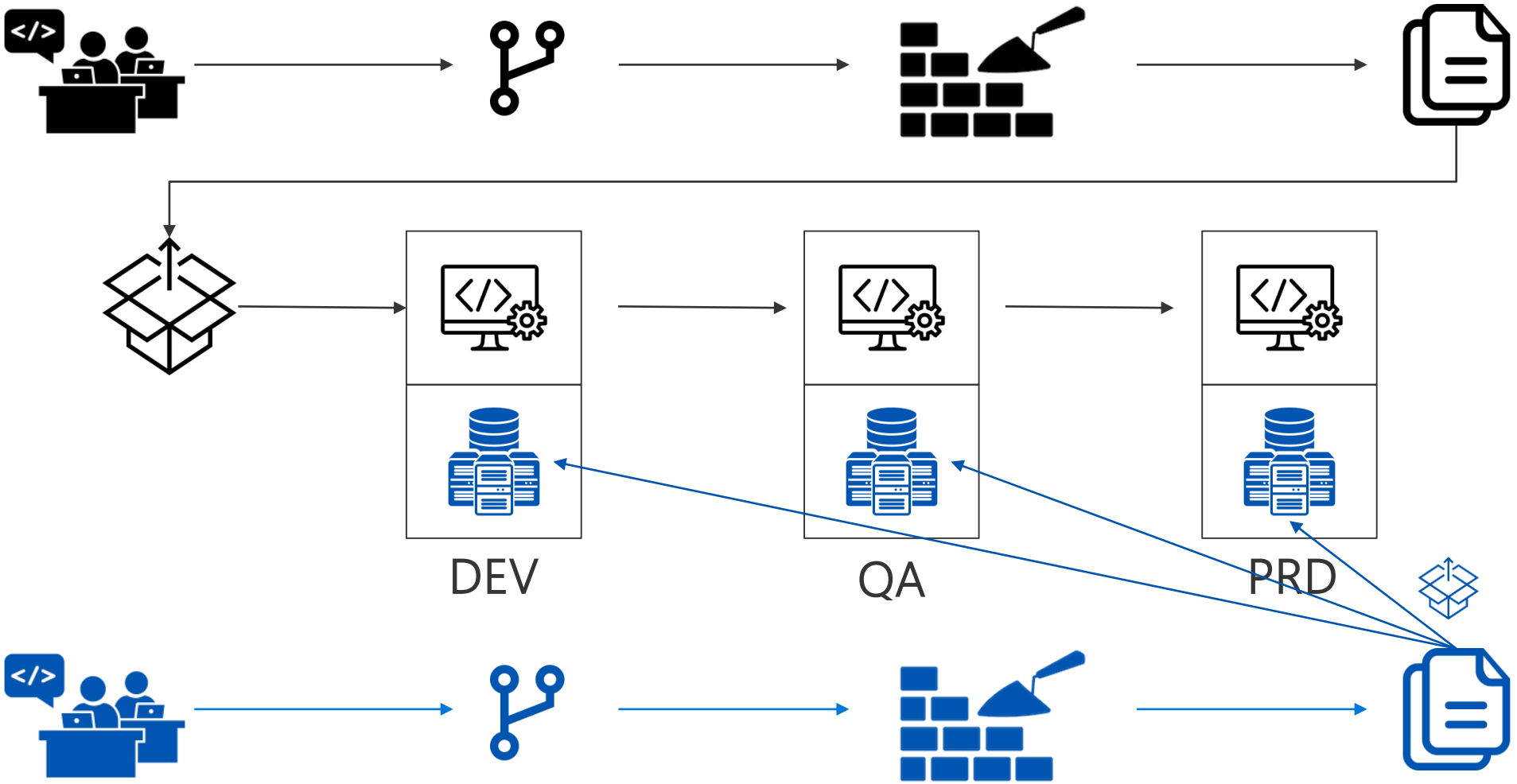
Infrastructure as Code (IaC)

Infrastructure as Code is the **process** of managing and provisioning computing **infrastructure and its configuration** through machine-processable **definition files**.

It treats the infrastructure as a software system, applying **software engineering practices** to manage changes to the system in a **repeatable**, structured and safe way

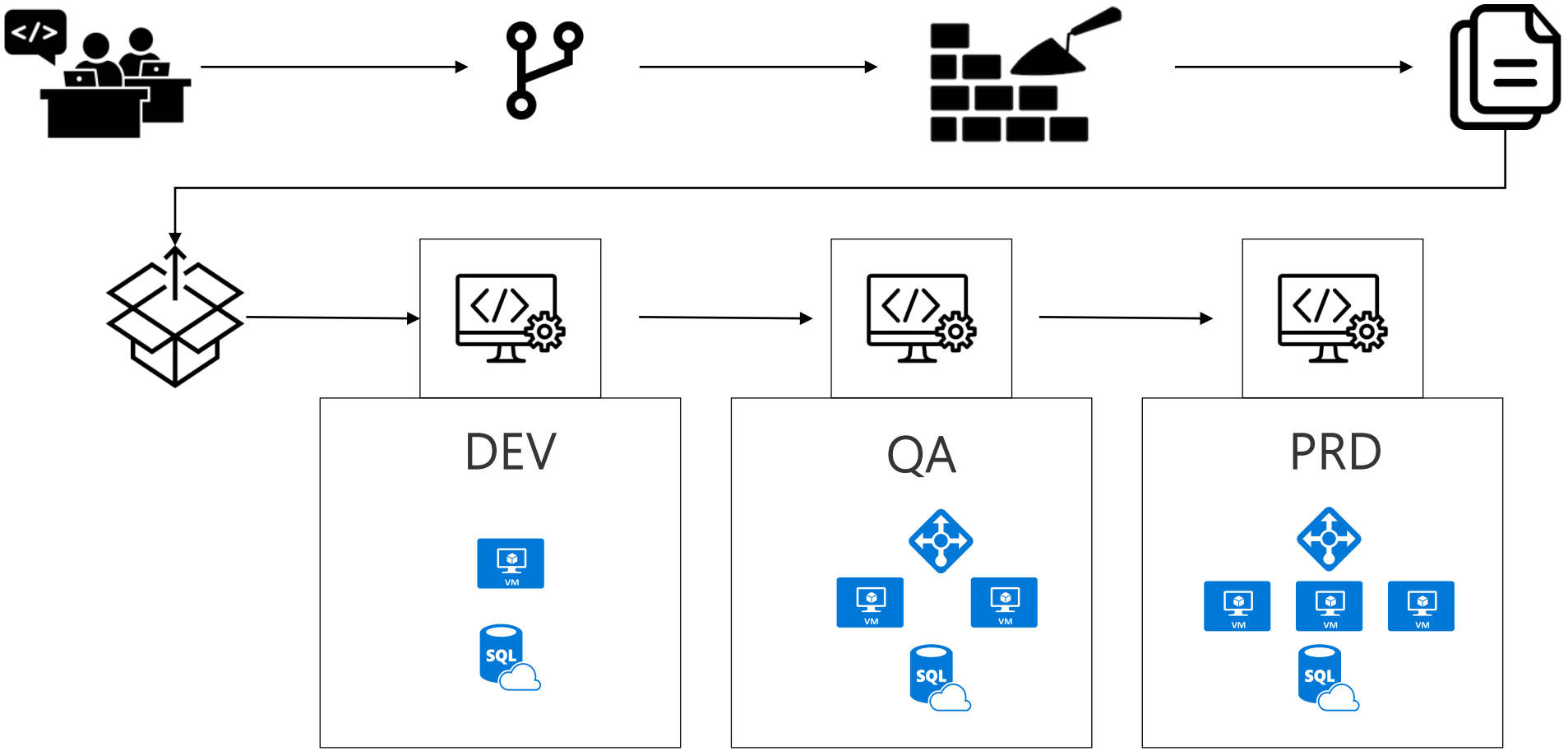
- Declarative model (not imperative)
- Used in combination with continuous delivery
- Creates the same environment every time it is applied (idempotent)

Application Development Approach + IaC



Application Development Approach + IaC

Environments can be different in terms of size and elements



Benefits

- Simplify deployment and management
- Deliver stable environments quickly and at scale
- Decrease provisioning time
- Avoid configuration drift

Best Practices

- Maintain version control
- Test infrastructure
- Code infrastructure specifications in “configuration files”
- Keep secrets away (use KeyVaults)
- Use as little documentation as possible
  - Your code is a representation of your documentation

What are ARM Templates?

- Declarative files for creating Azure resources in a reliable, repeatable and auditable way.

Infrastructure as Code

Define Azure resources using text files.

Declarative Syntax

Declare how the resources should be and Azure Resource Manager “makes it so”.

JSON

ARM Templates are JSON format text files. Edit them in Visual Studio Code (or other text editors). Version control them.

Meta-Language

Contains some programming language constructs such as functions and loops.

https://docs.microsoft.com/en-us/azure/templates/

Features of ARM Templates

Parameterized

Parameters can be used to configure resource attributes at deployment time. Allows generalization and re-use of ARM Templates.

Modular

Templates can be broken into smaller, re-usable components and **linked** together at deployment time by using **Deployment** resources.

Templates can also be **nested** inside other templates.

Idempotent

Resources will only be changed or created if they have drifted out of state or need to be updated or created.

Testable

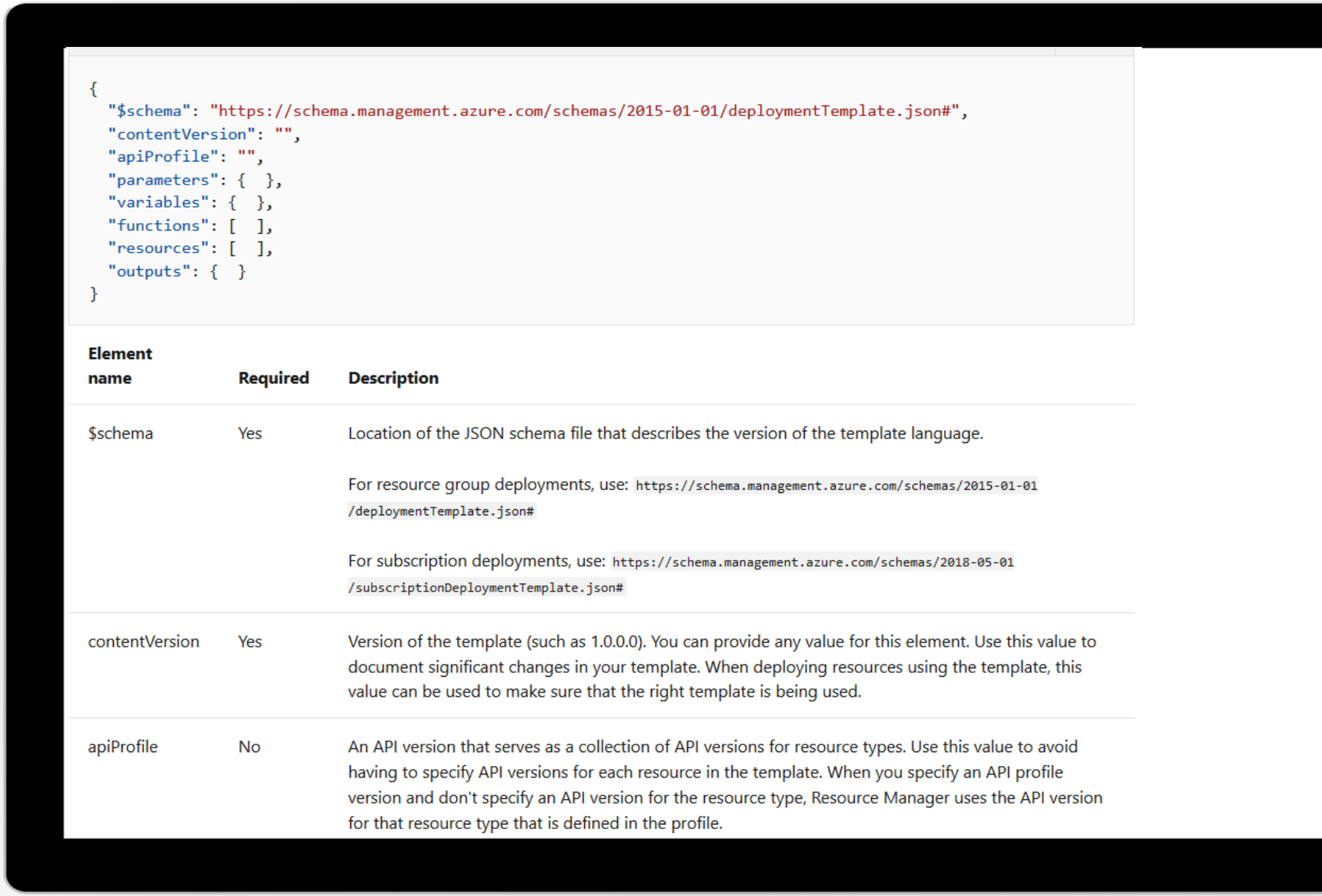
Templates can be validated prior to deployment.

Version Control

Using ARM Templates with version control allows your infrastructure to be reviewable, traceable and auditable.

Demo

ARM Templates 101



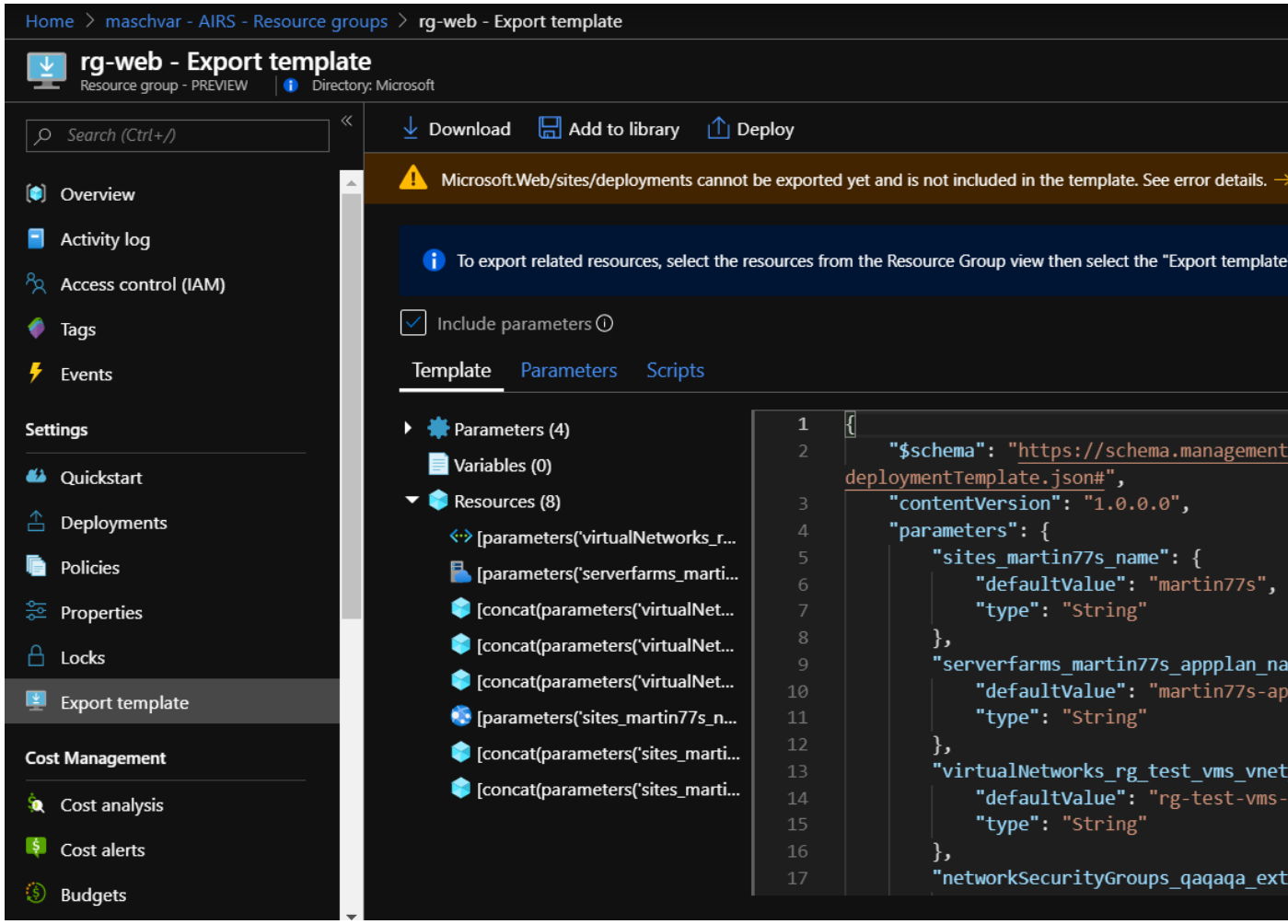
```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "",
  "apiProfile": "",
  "parameters": {  },
  "variables": {  },
  "functions": [  ],
  "resources": [  ],
  "outputs": {  }
}
```

Element name	Required	Description
$schema	Yes	Location of the JSON schema file that describes the version of the template language. For resource group deployments, use: `https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#` For subscription deployments, use: `https://schema.management.azure.com/schemas/2018-05-01/subscriptionDeploymentTemplate.json#`
contentVersion	Yes	Version of the template (such as 1.0.0.0). You can provide any value for this element. Use this value to document significant changes in your template. When deploying resources using the template, this value can be used to make sure that the right template is being used.
apiProfile	No	An API version that serves as a collection of API versions for resource types. Use this value to avoid having to specify API versions for each resource in the template. When you specify an API profile version and don't specify an API version for the resource type, Resource Manager uses the API version for that resource type that is defined in the profile.

Exporting ARM Templates

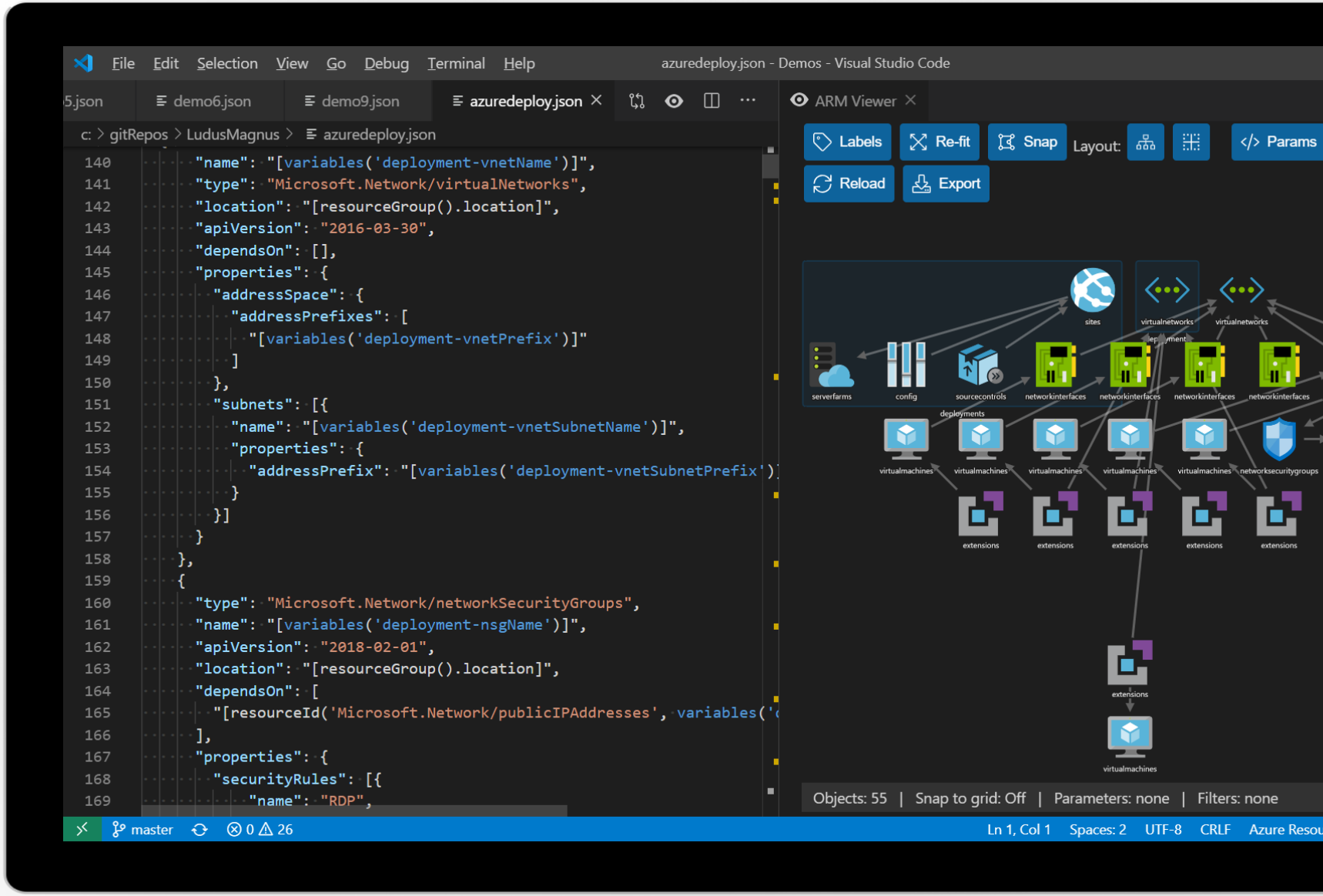
- Export from Resource Group
- Export Specific Resources
- Export from Deployments

- Generates a new ARM template
- Simple, extendable patterns for
  - Azure CLI w/ Bash, PowerShell, .NET, Ruby
- Not all resource types supported
- Good starting point for automation exploration
- Does not include changes post-deployment



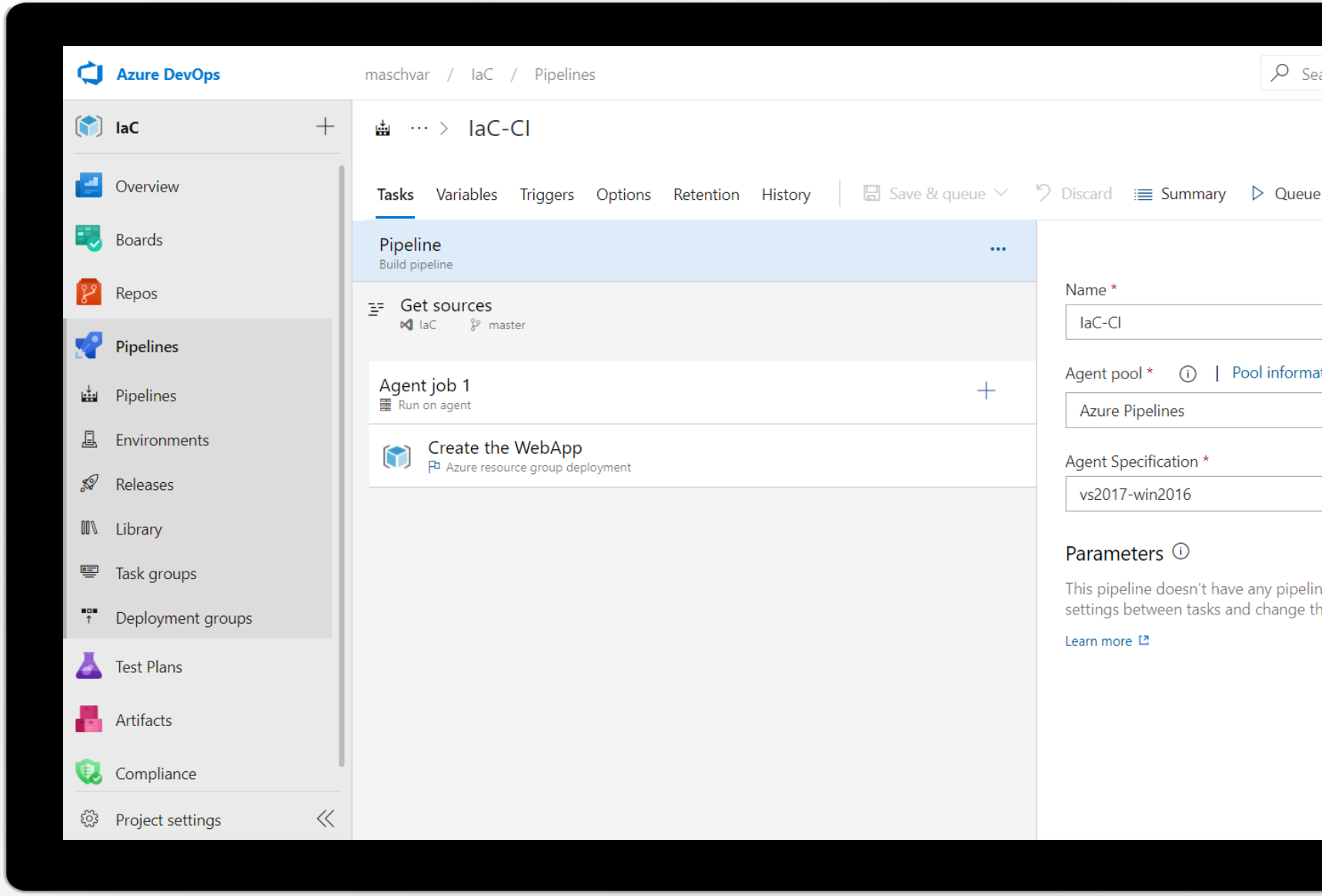
Demo

ARM Templates 202



Demo

IaC w/ ADO



IaC everything?

- It depends. Sometimes the answer is no
- Some Azure components are deployed only one-time
- Coding infrastructure require time
  - Especially at the beginning, code only the components that can be reused
- Azure shared components can be prepared manually
  - ExpressRoute configuration, Hub configuration

Resources

Azure Resource Manager templates

https://docs.microsoft.com/en-us/azure/azure-resource-manager/template-deployment-overview

Understand the structure and syntax of Azure Resource Manager templates

https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-group-authoring-templates

Azure Quickstart Templates

https://azure.microsoft.com/en-us/resources/templates/

CD of an Azure virtual machine using a Resource Manager template

https://docs.microsoft.com/en-us/azure/devops/pipelines/apps/cd/azure/deploy-provision-azure-vm

The Azure Superpowers workshop

https://github.com/microsoft/AzureSuperpowers