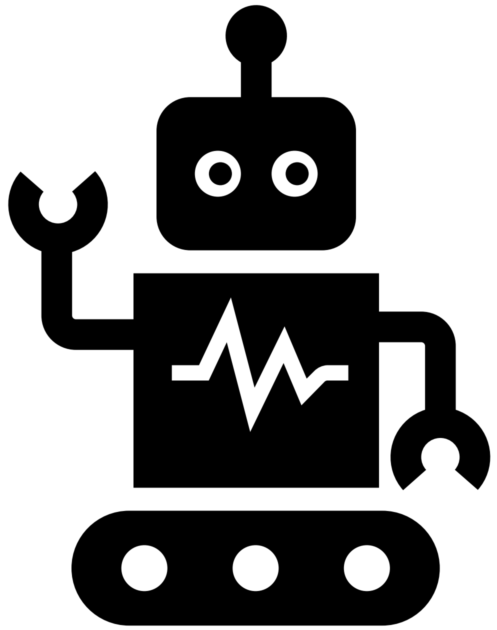
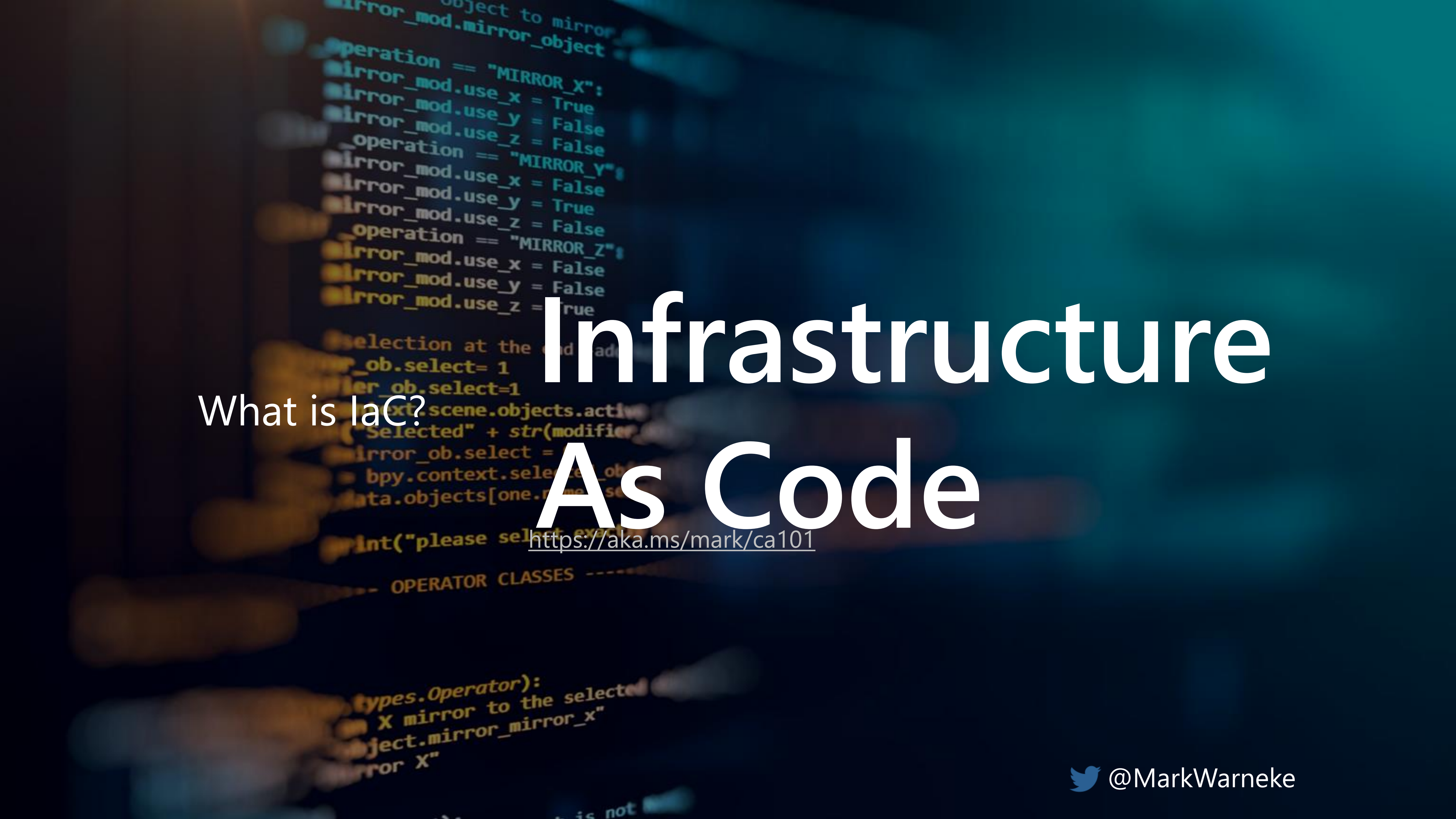
Hi



@MarkWarneke



Infrastructure As Code

What is IaC?

https://aka.ms/mark/ca101

@MarkWarneke

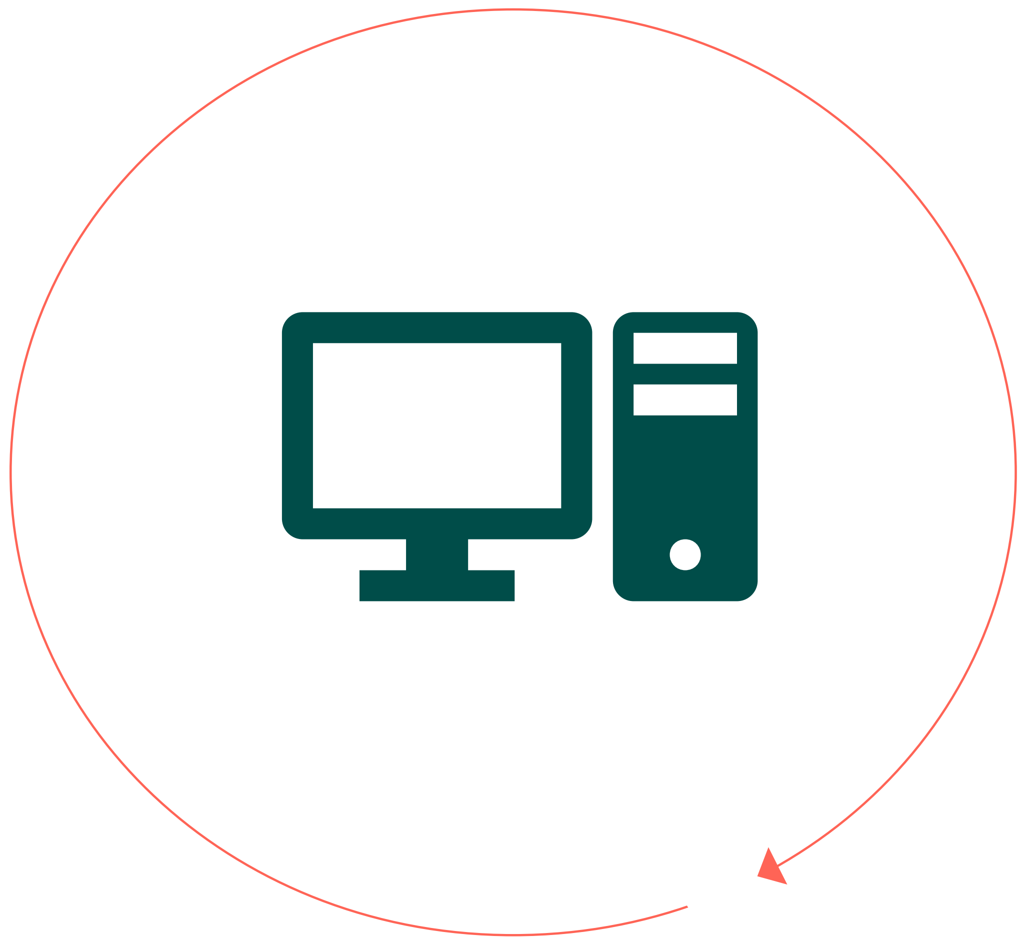
```
// azuredeploy.json
"comments": "Azure Data Lake Gen 2 Storage Account",
"type": "Microsoft.Storage/storageAccounts",
"apiVersion": "2019-04-01",
"name": "[parameters('resourceName')]",
"sku": {
      "name": "[parameters('storageAccountSku')]"
},
"kind": "StorageV2",
"location": "[parameters('location')]",
"tags": {},
"identity": { "type": "SystemAssigned" },
"properties": {
      "encryption": {
          "services": {
              "blob": { "enabled": true },
              "file": { "enabled": true }
          },
          "keySource": "Microsoft.Storage"
      },
      "isHnsEnabled": true,
      "networkAcls": "[json(parameters('networkAcls'))]",
      "accessTier": "[parameters('storageAccountAccessTier')]",
      "supportsHttpsTrafficOnly": true
```

@MarkWarneke

Application Development

Infrastructure Development

@MarkWarneke



Outside

Hardware Configuration

- VM Size, Disks, Network
- RBAC, secrets etc.
- Resource Settings

Inside

Software

- Application Code
- Desired state
- Configurations & scripts

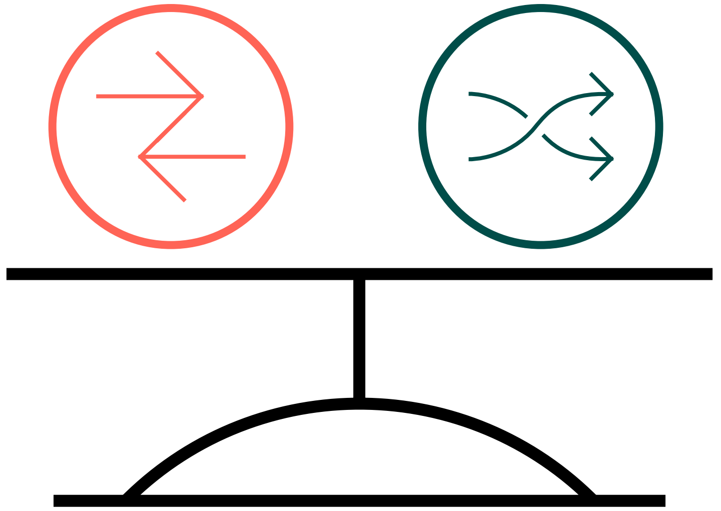
declarative

describe final state

imperative

executing steps to get to final state

What is the challange?



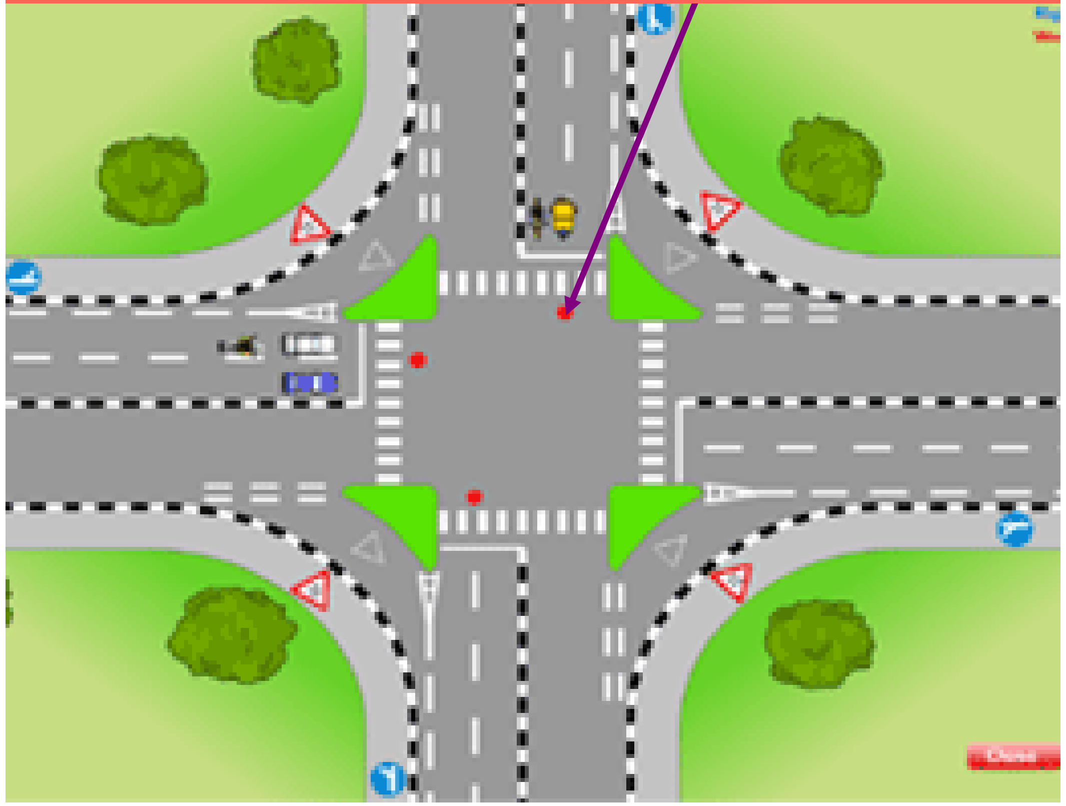
Control

Speed
Agility

@MarkWarneke

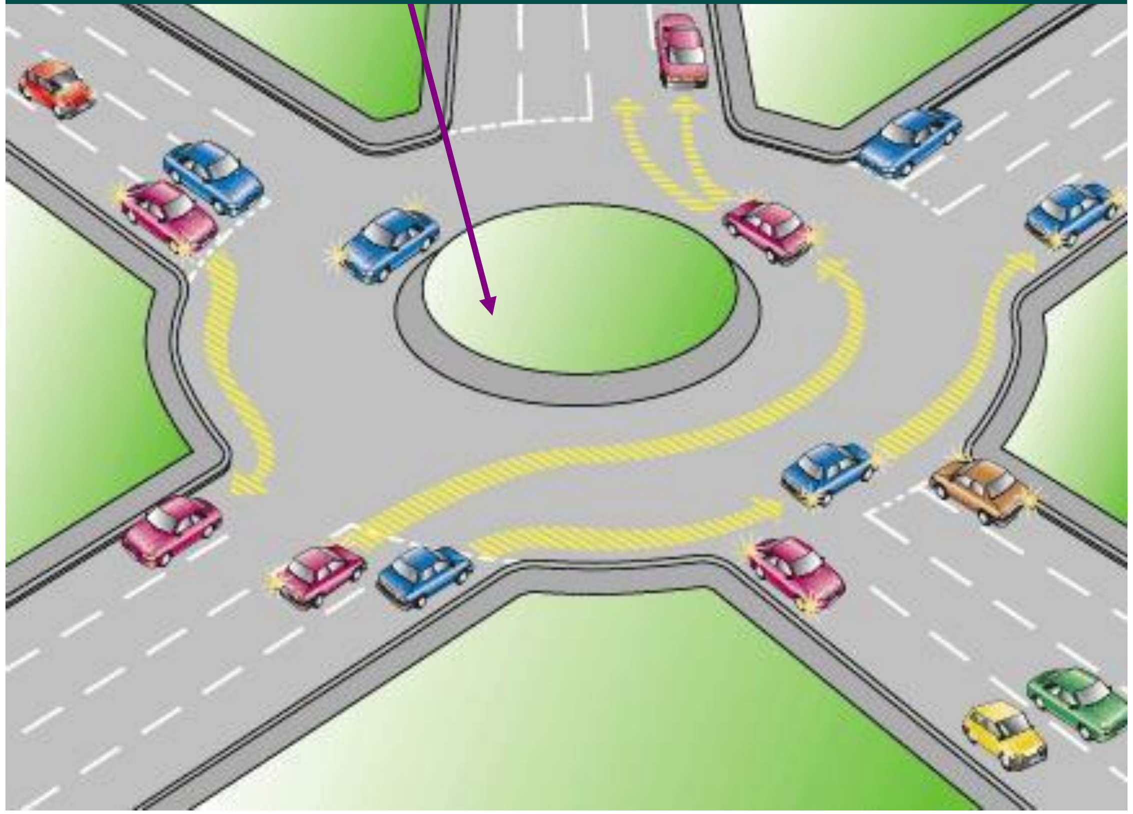
Paradigm shift

Enforce/Control



Controlled & *central* responsibility

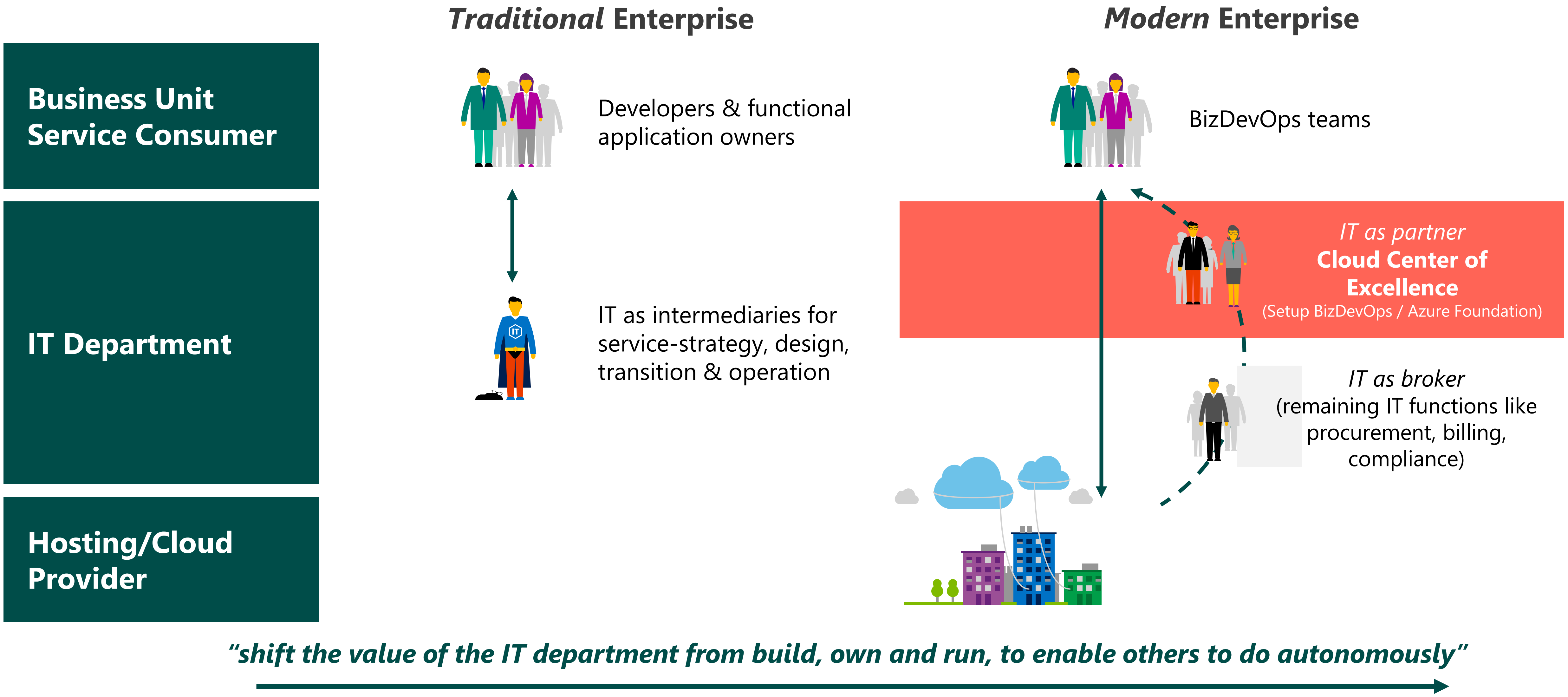
Enable/Support



Freedom & *delegated* responsibility

@MarkWarneke

What is CCoE about?



***Traditional* Enterprise** | ***Modern* Enterprise**

Business Unit Service Consumer

Developers & functional application owners

BizDevOps teams

IT Department

IT as intermediaries for service-strategy, design, transition & operation

IT as partner
Cloud Center of Excellence
(Setup BizDevOps / Azure Foundation)

IT as broker
(remaining IT functions like procurement, billing, compliance)

Hosting/Cloud Provider

"shift the value of the IT department from build, own and run, to enable others to do autonomously"

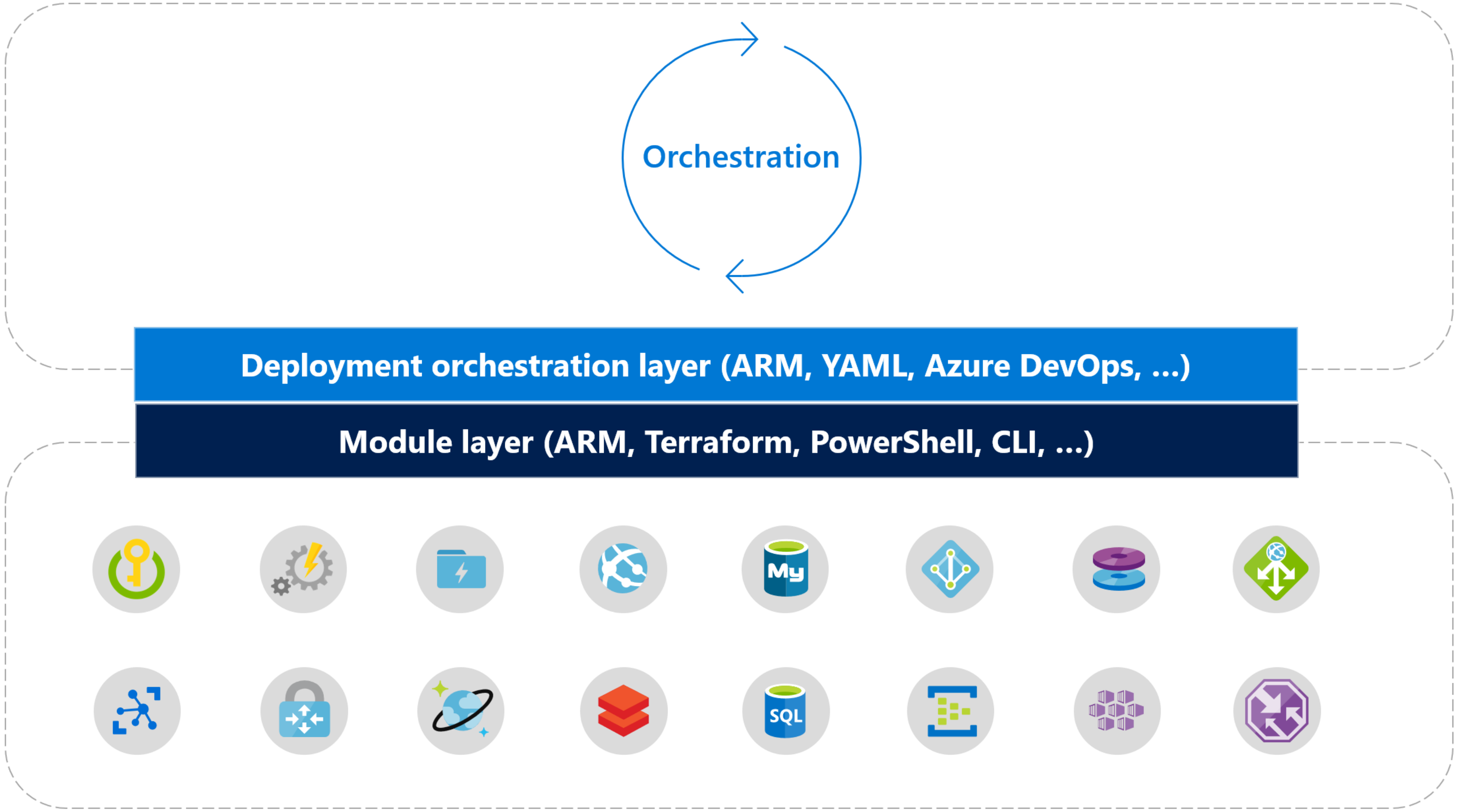
@MarkWarneke

Microsoft

ComponentFactory

@MarkWarneke

Deployment Orchestration


Orchestration
Deployment orchestration layer (ARM, YAML, Azure DevOps, ...)
Module layer (ARM, Terraform, PowerShell, CLI, ...)

@MarkWarneke

Deployment Approach

Idea

Self-contained, generic and idempotent modules per resource type.

Module

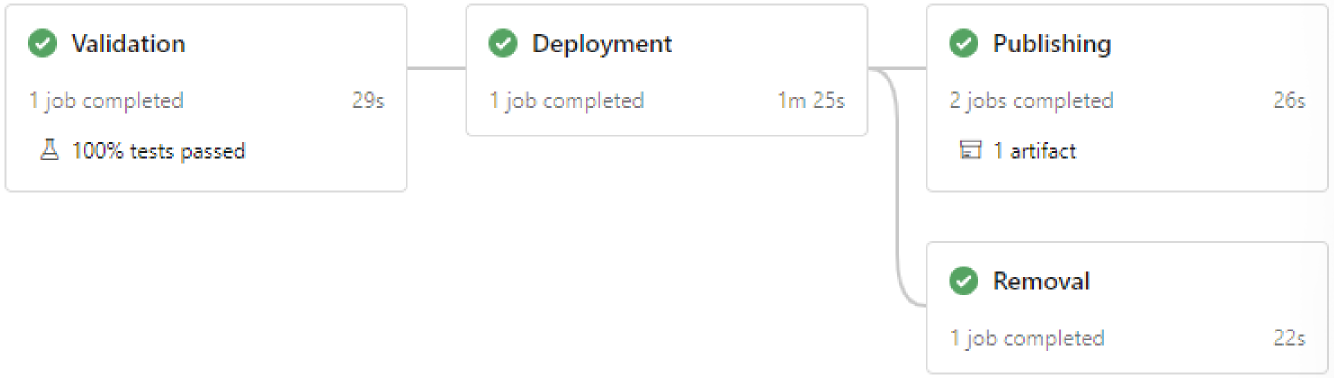
- ARM template (deploy.json)
- Parameters file (parameters.json)
- YAML pipeline (pipeline.yaml)

Pipelines

Validation

Deployment

Publishing

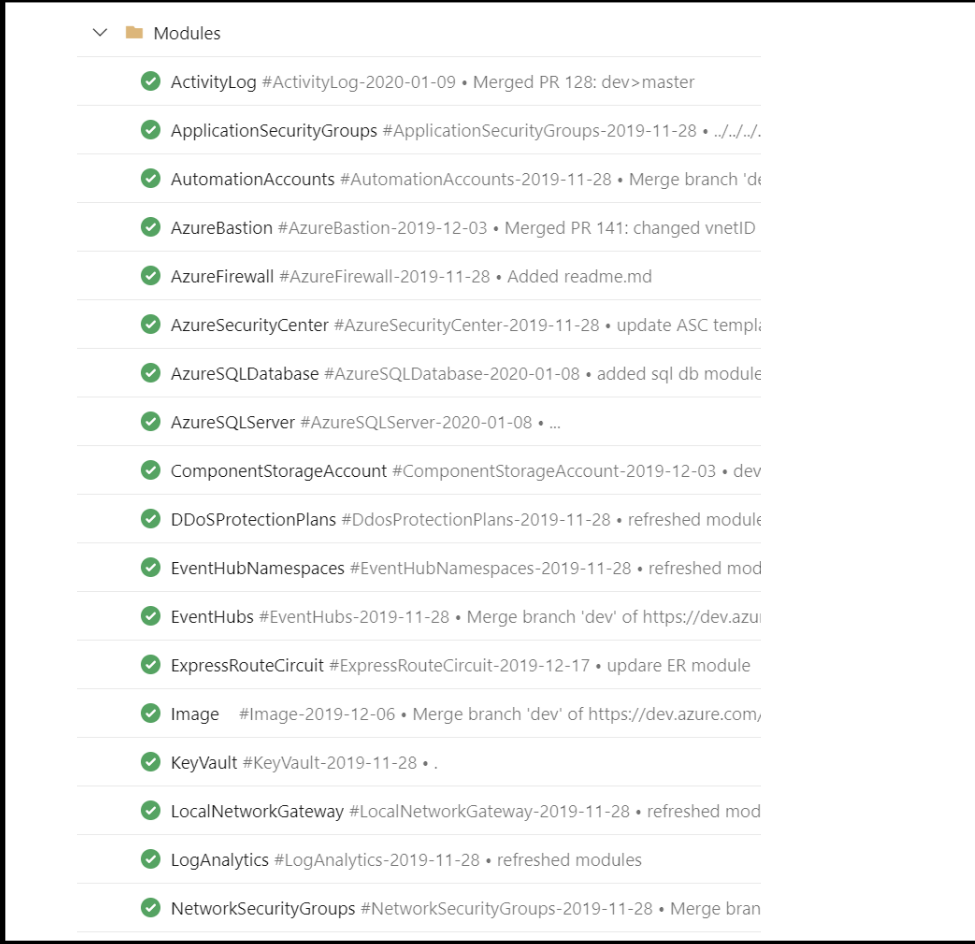


Validation
1 job completed 29s
100% tests passed

Deployment
1 job completed 1m 25s

Publishing
2 jobs completed 26s
1 artifact

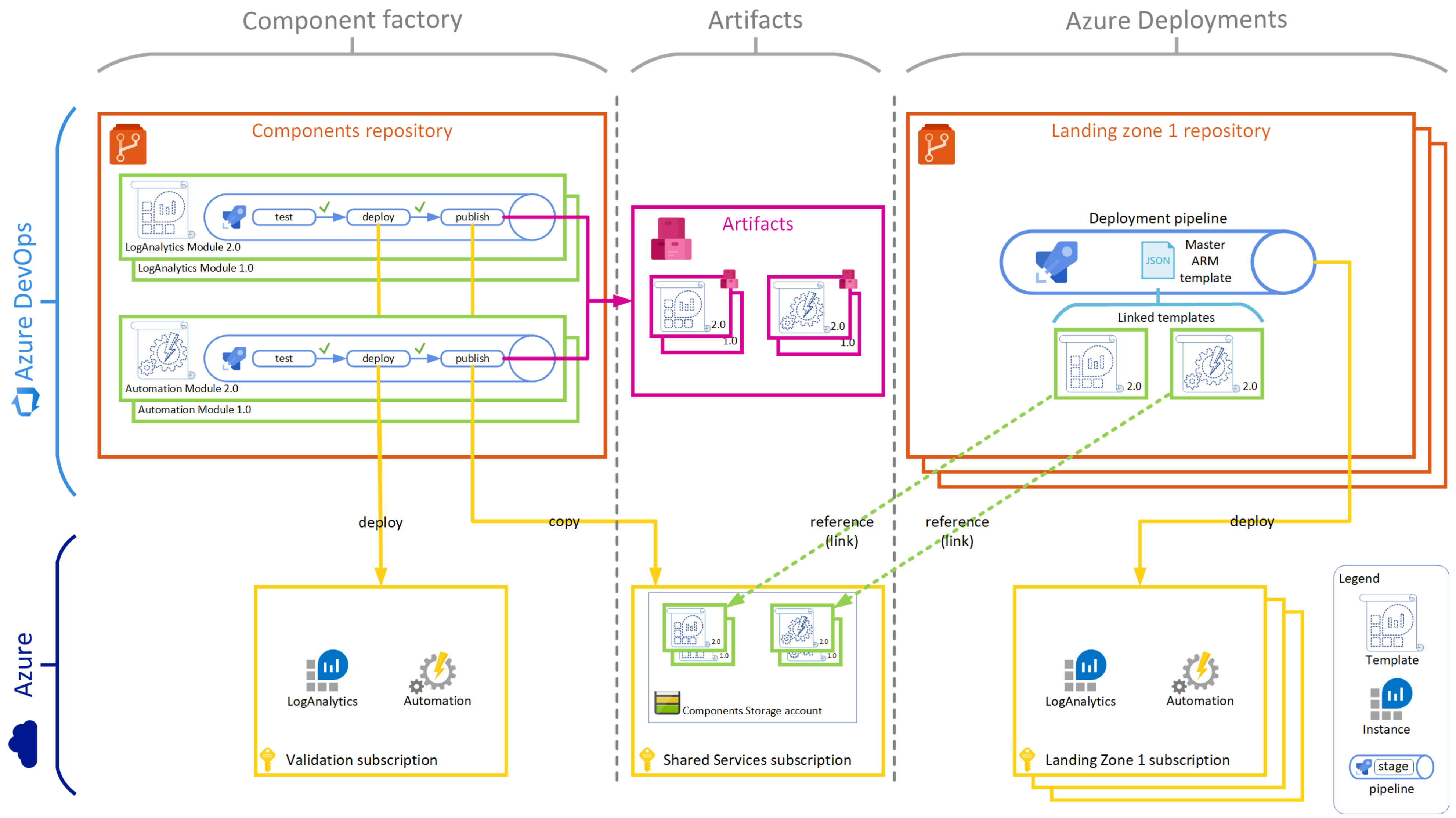
Removal
1 job completed 22s



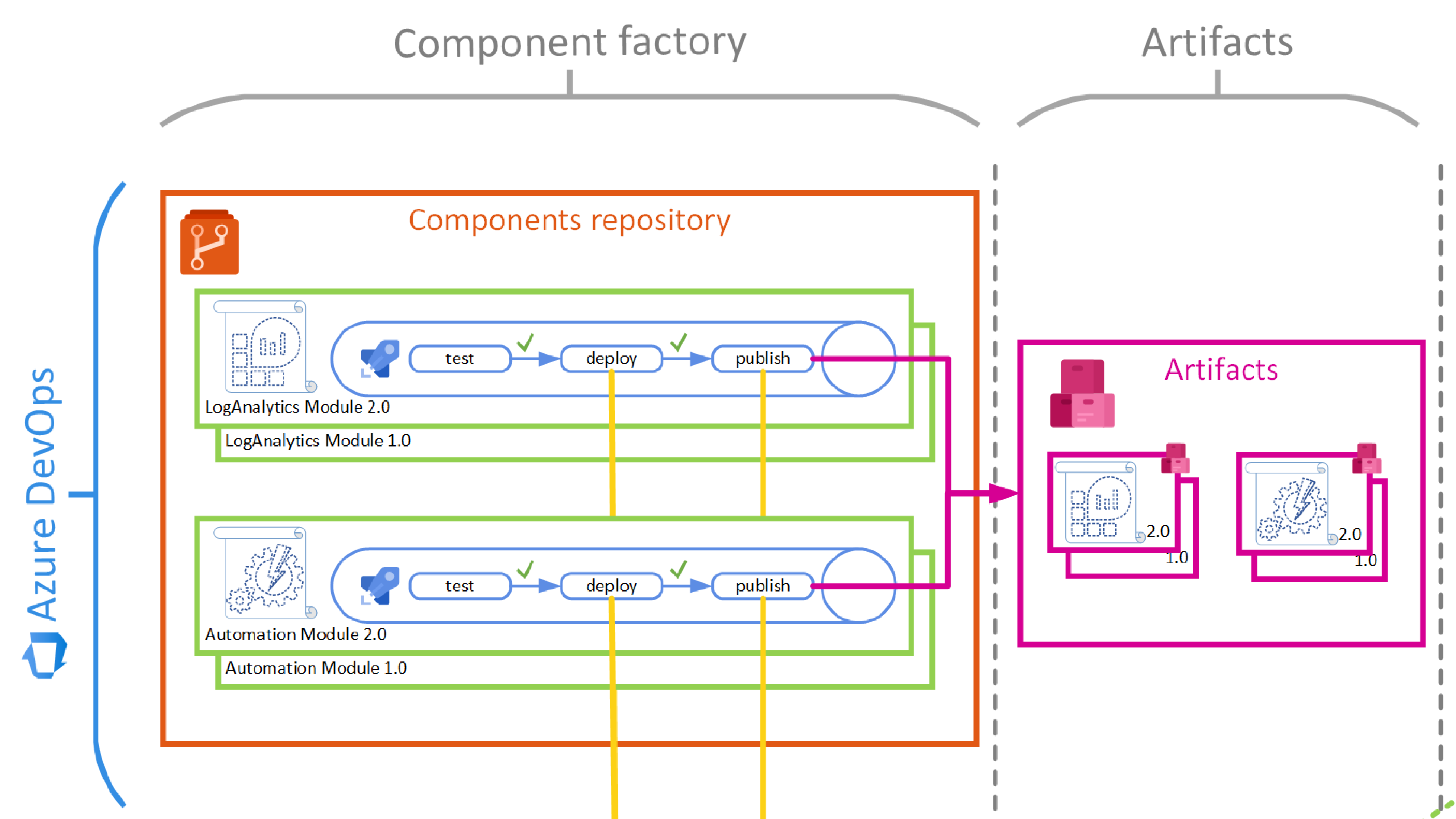
Modules
- ActivityLog #ActivityLog-2020-01-09 • Merged PR 128: dev>master
- ApplicationSecurityGroups #ApplicationSecurityGroups-2019-11-28 • ../../../.
- AutomationAccounts #AutomationAccounts-2019-11-28 • Merge branch 'de
- AzureBastion #AzureBastion-2019-12-03 • Merged PR 141: changed vnetID
- AzureFirewall #AzureFirewall-2019-11-28 • Added readme.md
- AzureSecurityCenter #AzureSecurityCenter-2019-11-28 • update ASC templ
- AzureSQLDatabase #AzureSQLDatabase-2020-01-08 • added sql db module
- AzureSQLServer #AzureSQLServer-2020-01-08 • ...
- ComponentStorageAccount #ComponentStorageAccount-2019-12-03 • dev
- DDoSProtectionPlans #DdosProtectionPlans-2019-11-28 • refreshed module
- EventHubNamespaces #EventHubNamespaces-2019-11-28 • refreshed mod
- EventHubs #EventHubs-2019-11-28 • Merge branch 'dev' of https://dev.azu
- ExpressRouteCircuit #ExpressRouteCircuit-2019-12-17 • updare ER module
- Image #Image-2019-12-06 • Merge branch 'dev' of https://dev.azure.com/
- KeyVault #KeyVault-2019-11-28 • .
- LocalNetworkGateway #LocalNetworkGateway-2019-11-28 • refreshed mod
- LogAnalytics #LogAnalytics-2019-11-28 • refreshed modules
- NetworkSecurityGroups #NetworkSecurityGroups-2019-11-28 • Merge bran

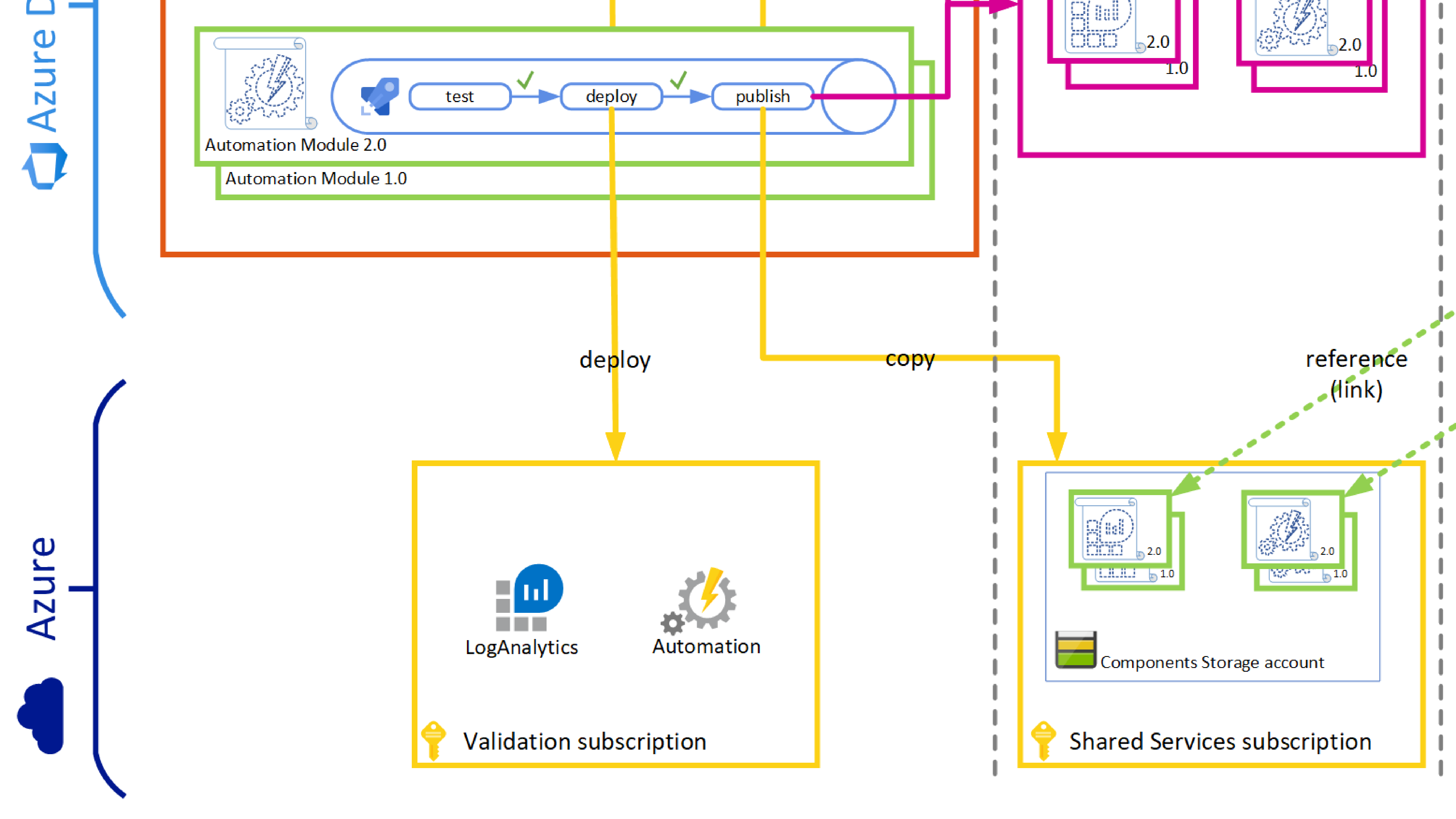
@MarkWarneke

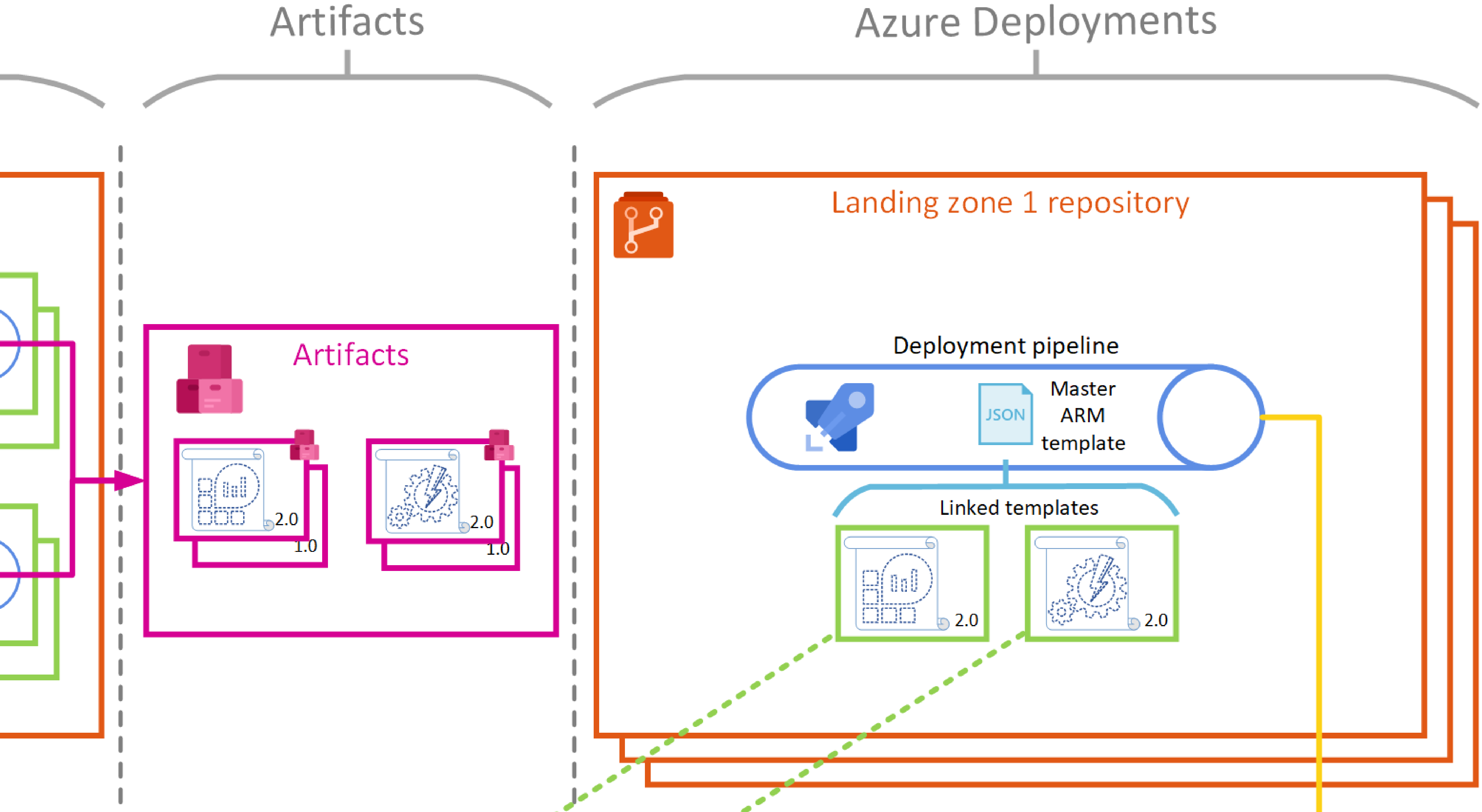
Deployment Model

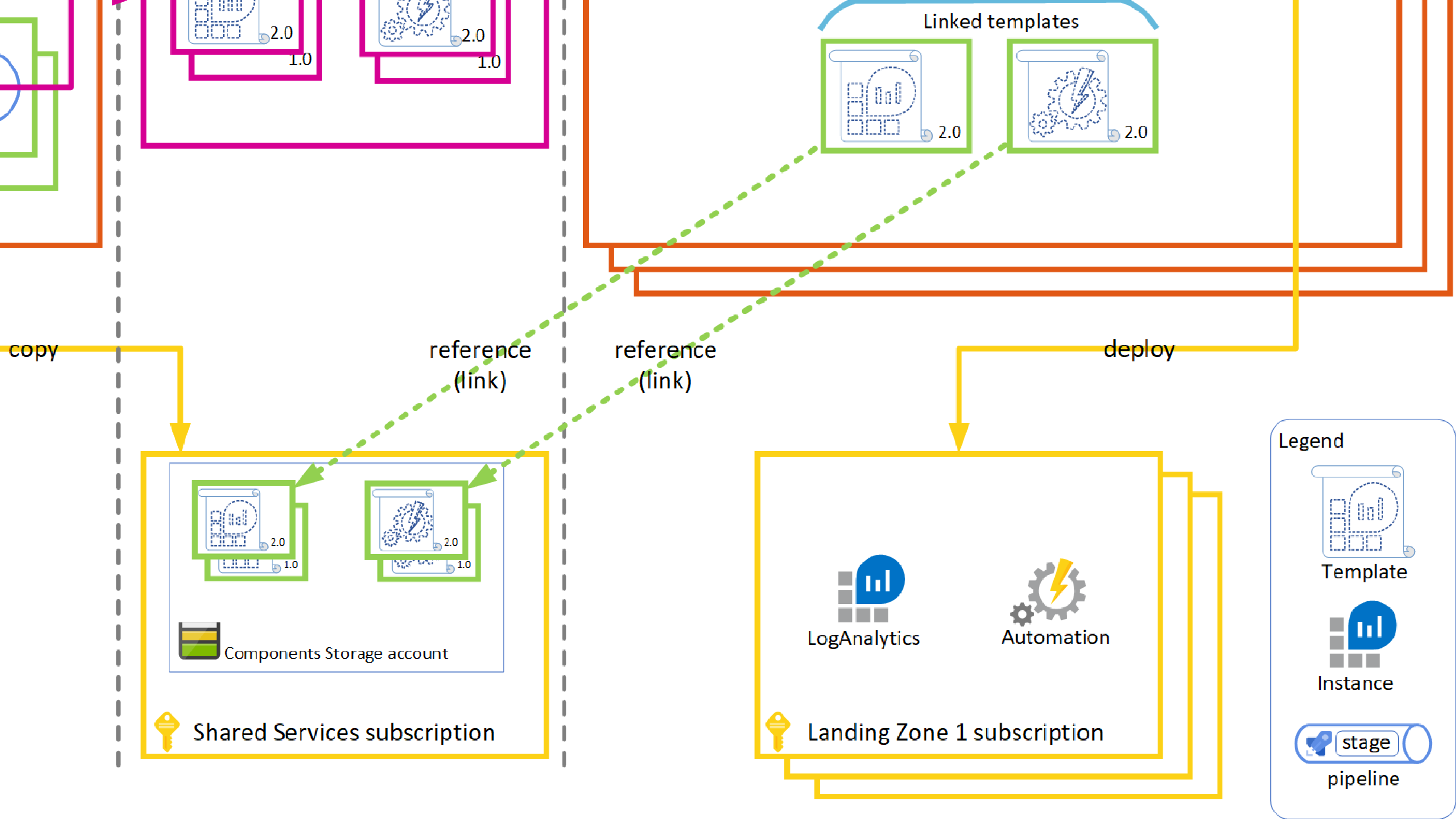


@MarkWarneke

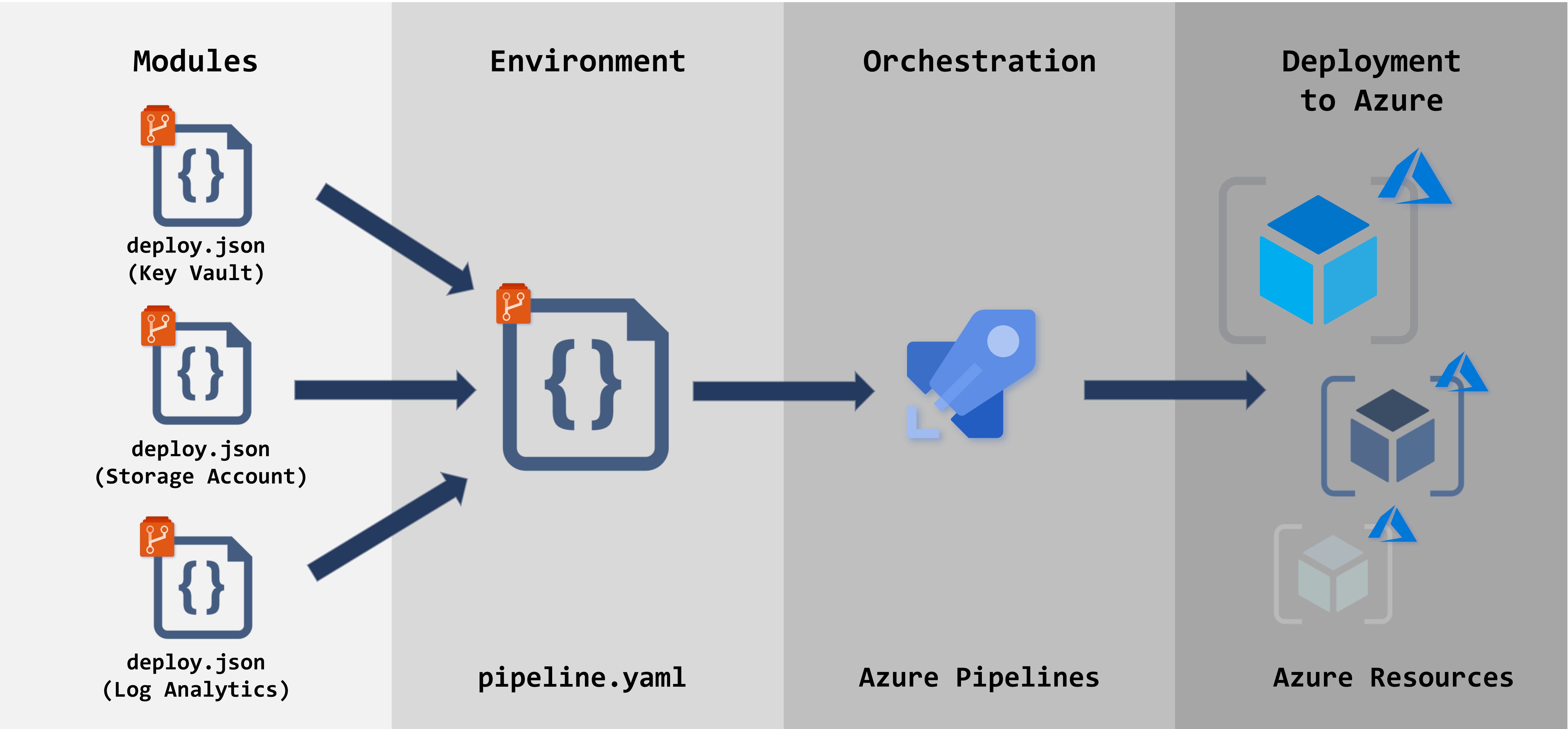

Component factory
Artifacts
Azure DevOps
Components repository
test
deploy
publish
LogAnalytics Module 2.0
LogAnalytics Module 1.0
test
deploy
publish
Automation Module 2.0
Automation Module 1.0
Artifacts
2.0
1.0
2.0
1.0


Azure D
Automation Module 2.0
Automation Module 1.0
test
deploy
publish
2.0
1.0
2.0
1.0
deploy
copy
reference
(link)
Azure
LogAnalytics
Automation
Validation subscription
2.0
1.0
2.0
1.0
Components Storage account
Shared Services subscription

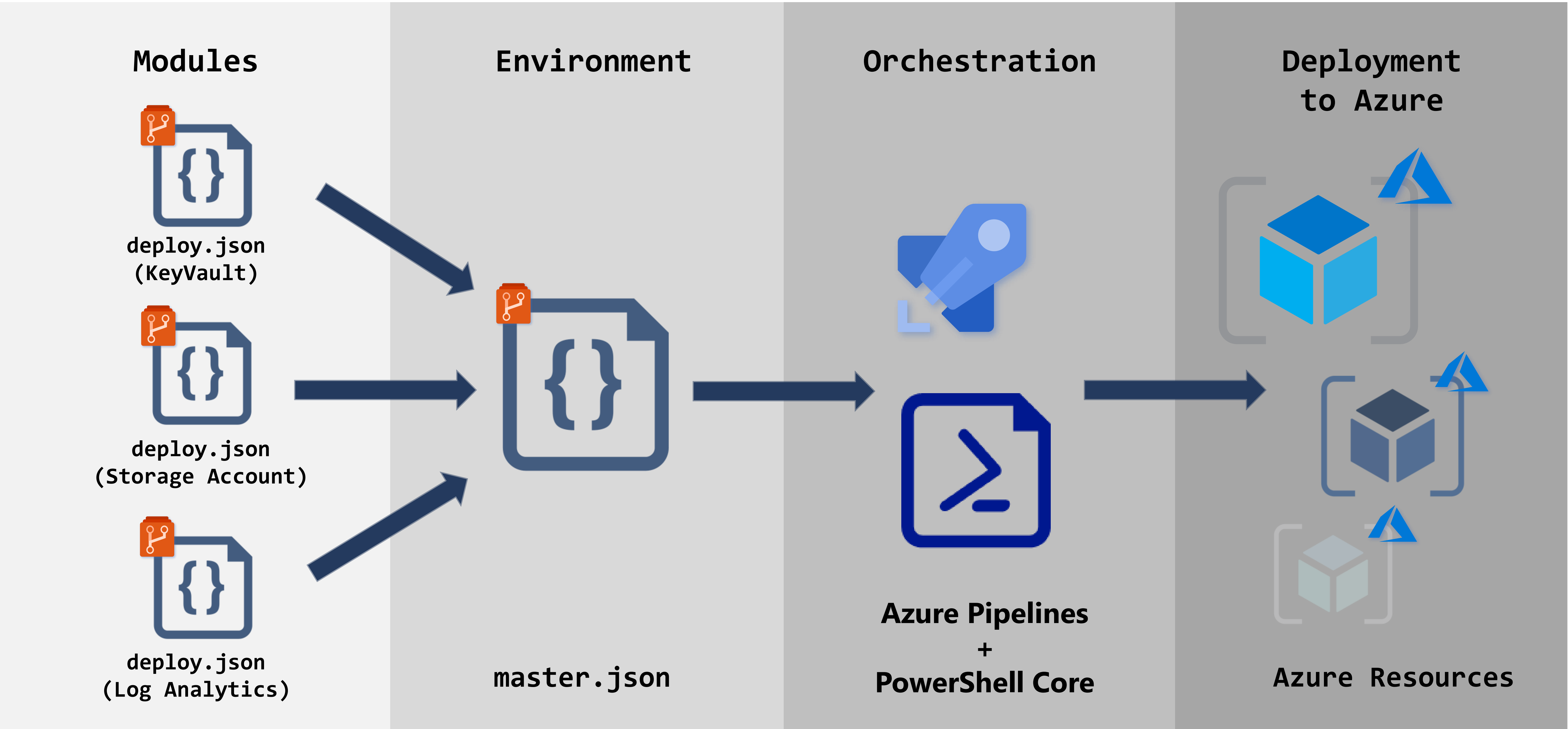

Artifacts
Azure Deployments
Landing zone 1 repository
Artifacts
2.0
1.0
2.0
1.0
Deployment pipeline
JSON
Master ARM template
Linked templates
2.0
2.0


2.0
1.0
2.0
1.0
Linked templates
2.0
2.0
copy
reference (link)
reference (link)
deploy
2.0
1.0
2.0
1.0
Components Storage account
Shared Services subscription
LogAnalytics
Automation
Landing Zone 1 subscription
Legend
Template
Instance
stage
pipeline

Pipeline-orchestrated Deployment



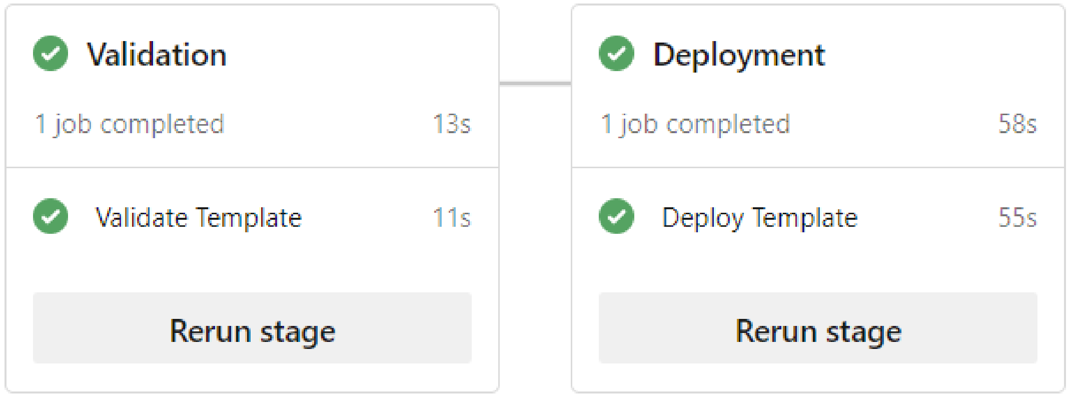
Template-orchestrated Deployment



Template-orchestrated Pipelines

Validate

Deploy


Validation

1 job completed 13s

Validate Template 11s

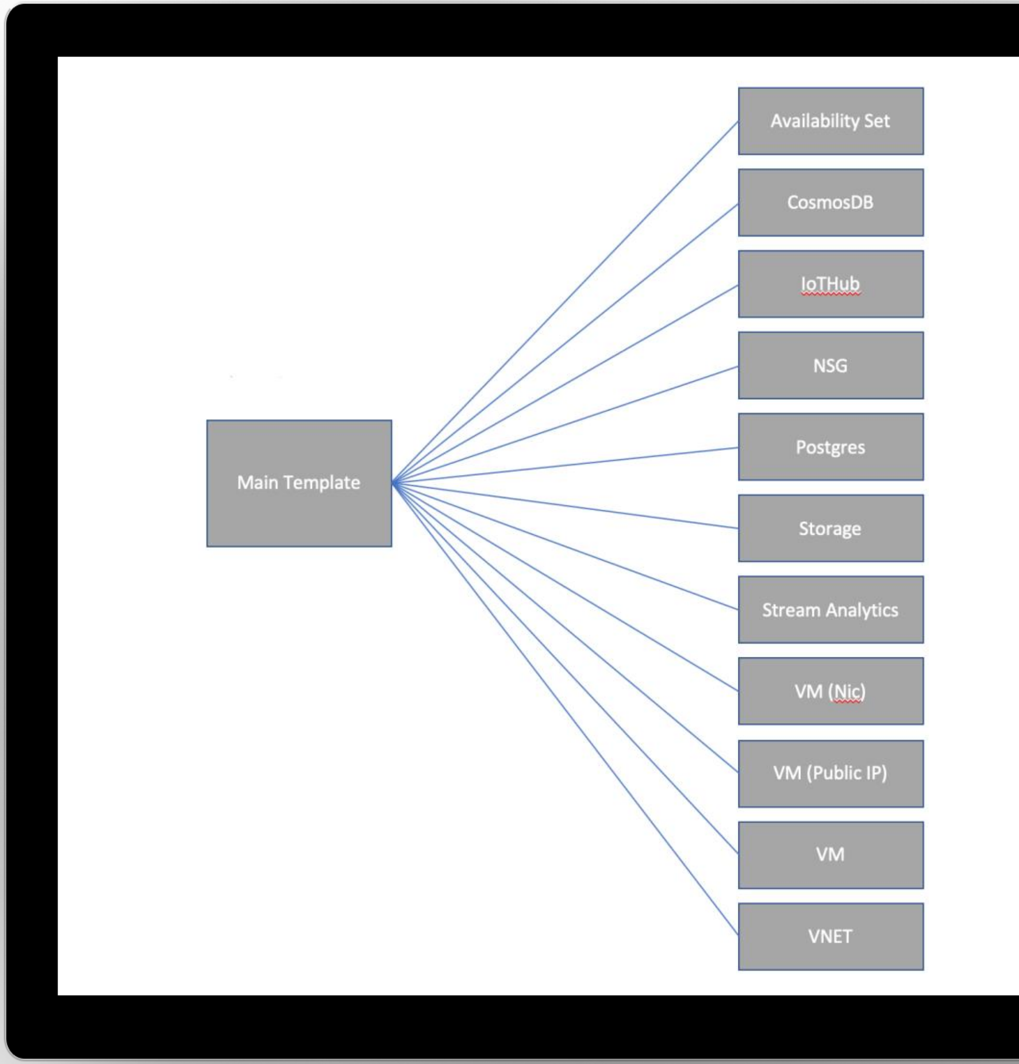
Rerun stage

Deployment

1 job completed 58s

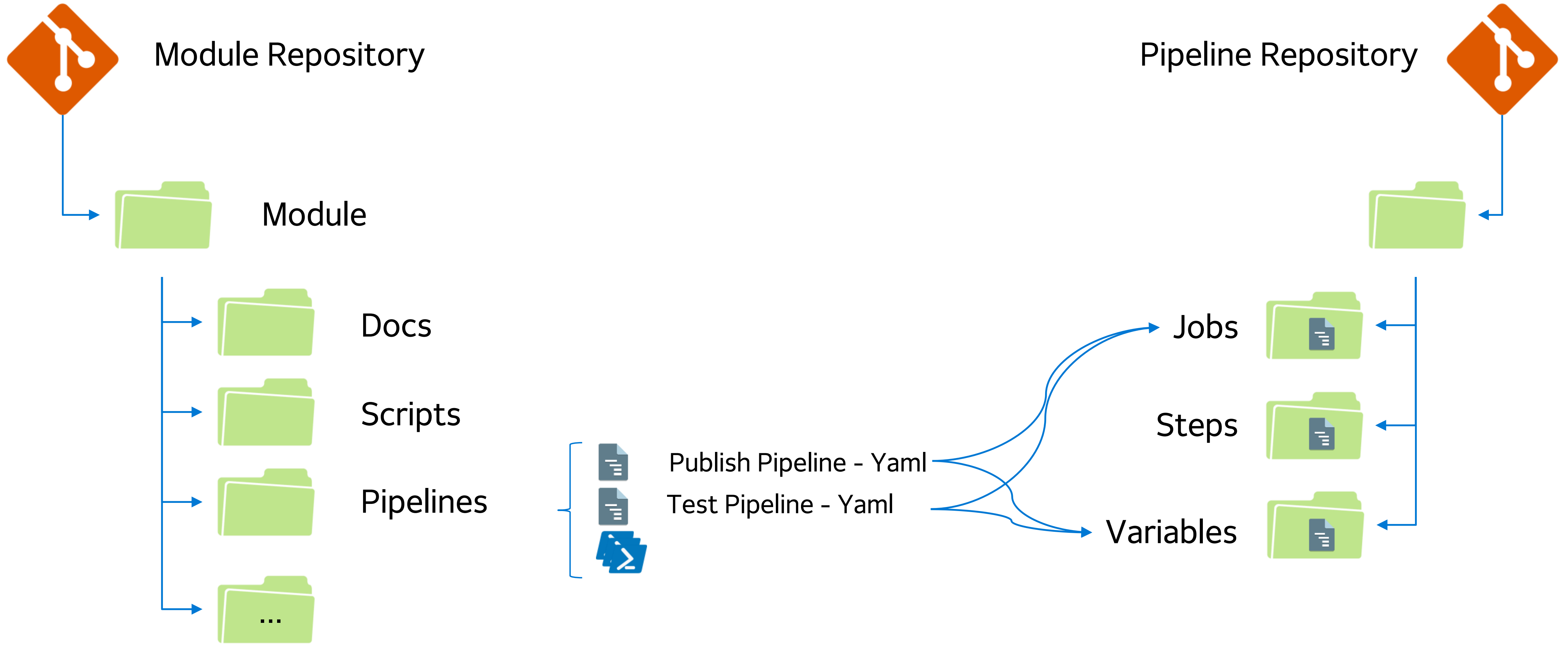
Deploy Template 55s

Rerun stage



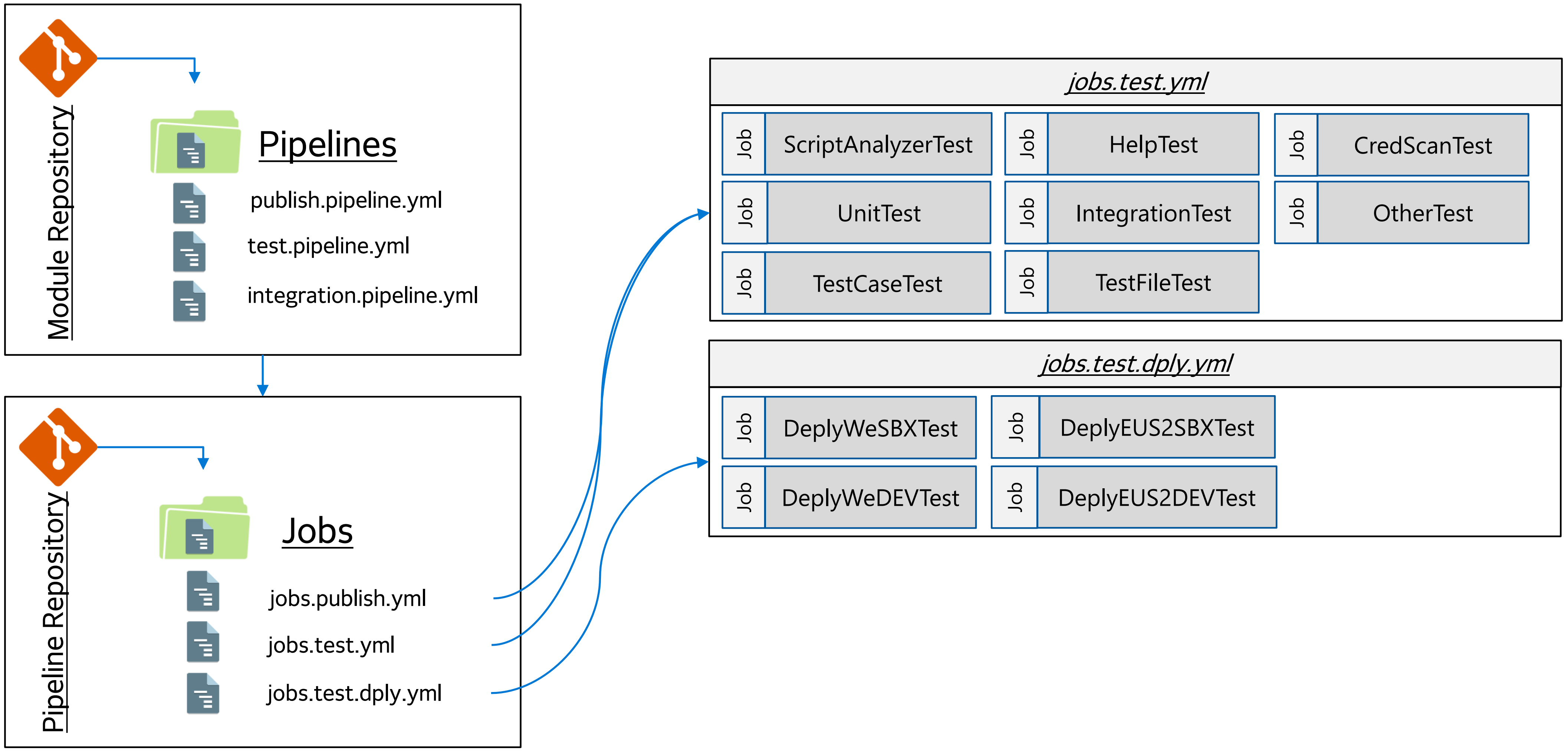
@MarkWarneke

Module Structure



@MarkWarneke

Folder Content



@MarkWarneke

Demo

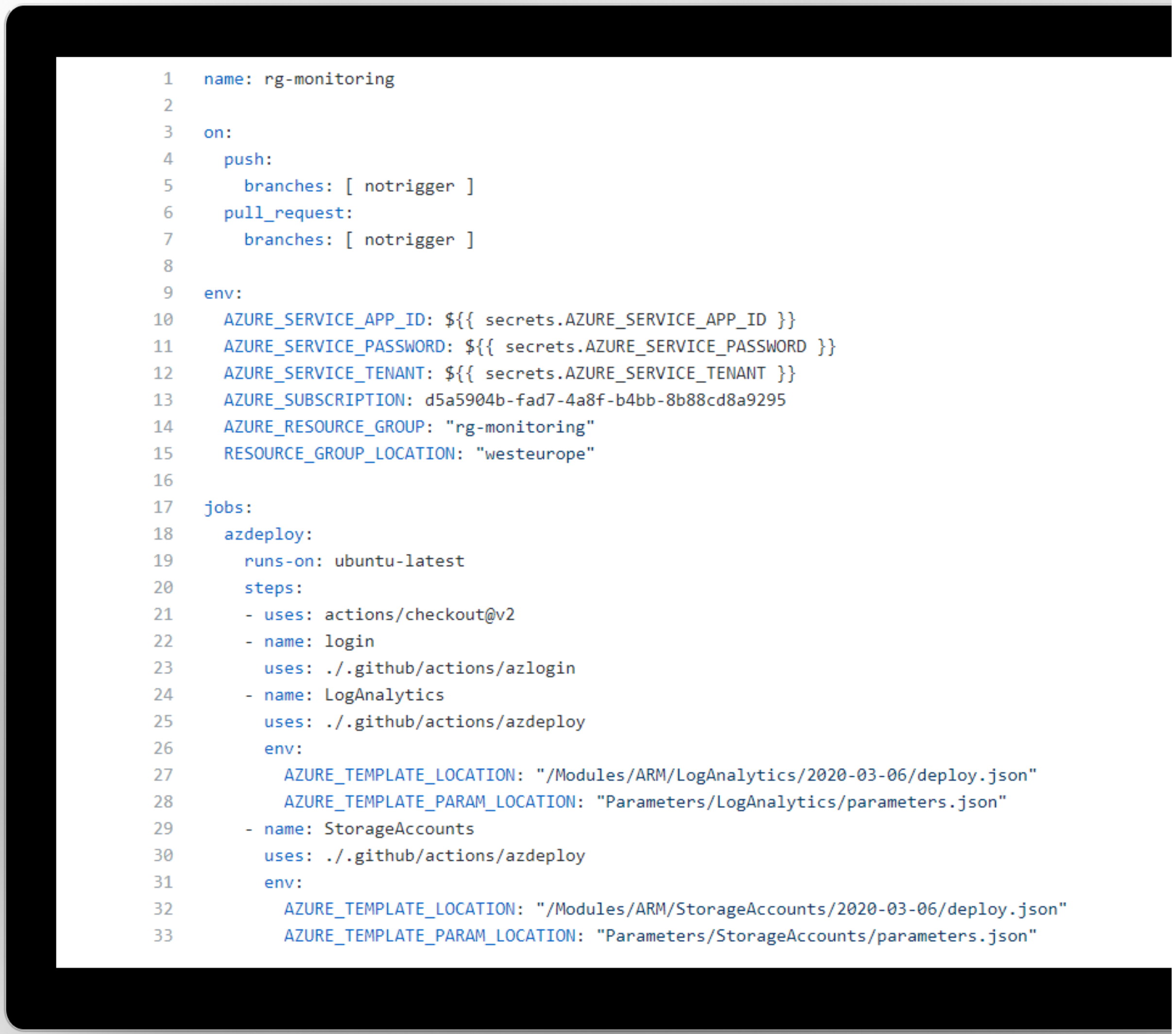
GitHub Actions and Workflows

GitHub Actions

Actions are individual tasks that you can combine to create jobs and customize your workflow. Own actions can be created and used.

GitHub Workflows

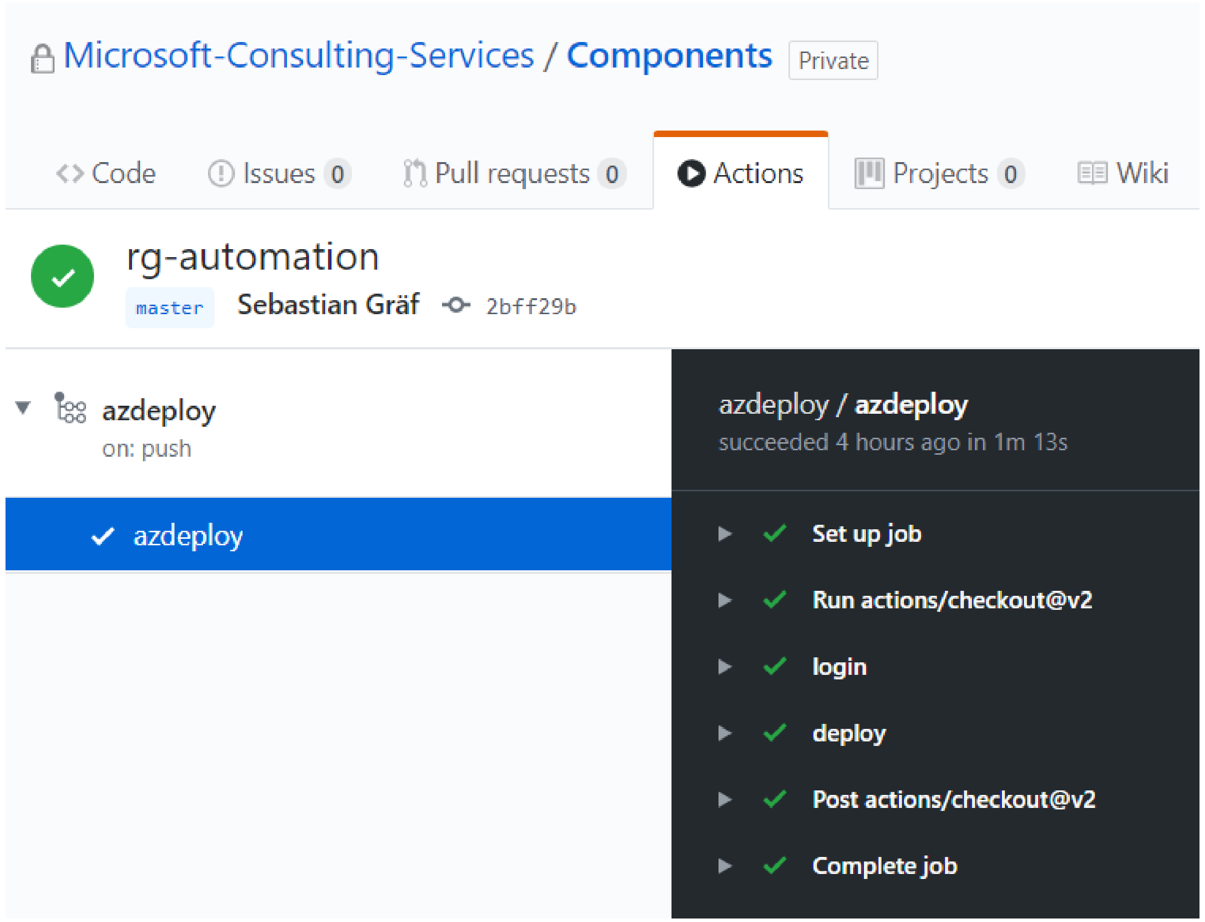
A workflow is a configurable automated process made up of one or more jobs.



```yaml
name: rg-monitoring

on:
  push:
    branches: [ notrigger ]
  pull_request:
    branches: [ notrigger ]

env:
  AZURE_SERVICE_APP_ID: ${{ secrets.AZURE_SERVICE_APP_ID }}
  AZURE_SERVICE_PASSWORD: ${{ secrets.AZURE_SERVICE_PASSWORD }}
  AZURE_SERVICE_TENANT: ${{ secrets.AZURE_SERVICE_TENANT }}
  AZURE_SUBSCRIPTION: d5a5904b-fad7-4a8f-b4bb-8b88cd8a9295
  AZURE_RESOURCE_GROUP: "rg-monitoring"
  RESOURCE_GROUP_LOCATION: "westeurope"

jobs:
  azdeploy:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v2
    - name: login
      uses: ./.github/actions/azlogin
    - name: LogAnalytics
      uses: ./.github/actions/azdeploy
      env:
        AZURE_TEMPLATE_LOCATION: "/Modules/ARM/LogAnalytics/2020-03-06/deploy.json"
        AZURE_TEMPLATE_PARAM_LOCATION: "Parameters/LogAnalytics/parameters.json"
    - name: StorageAccounts
      uses: ./.github/actions/azdeploy
      env:
        AZURE_TEMPLATE_LOCATION: "/Modules/ARM/StorageAccounts/2020-03-06/deploy.json"
        AZURE_TEMPLATE_PARAM_LOCATION: "Parameters/StorageAccounts/parameters.json"
```

@MarkWarneke

GitHub Workflows



Microsoft-Consulting-Services / **Components** Private

rg-automation

master Sebastian Gräf 2bff29b

azdeploy
on: push

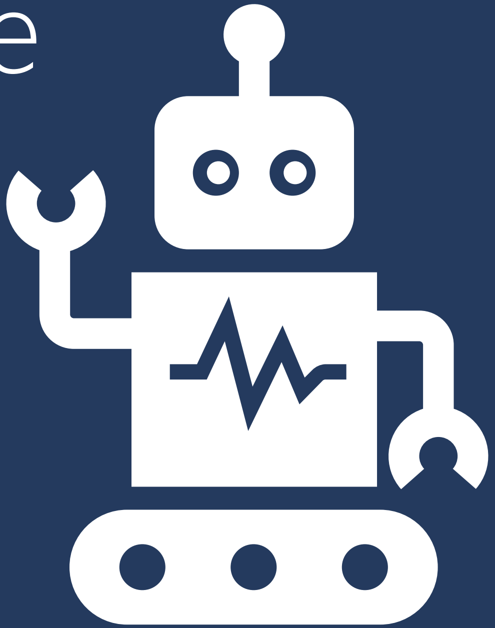
✓ azdeploy

azdeploy / **azdeploy**
succeeded 4 hours ago in 1m 13s

- ✓ Set up job
- ✓ Run actions/checkout@v2
- ✓ login
- ✓ deploy
- ✓ Post actions/checkout@v2
- ✓ Complete job

@MarkWarneke

Bye



@MarkWarneke

```
// azuredeploy.json
"comments": "Azure Data Lake Gen 2 Storage Account",
"type": "Microsoft.Storage/storageAccounts",
"apiVersion": "2019-04-01",
"name": "[parameters('resourceName')]",
"sku": {
      "name": "[parameters('storageAccountSku')]"
},
"kind": "StorageV2",
"location": "[parameters('location')]",
"tags": {},
"identity": { "type": "SystemAssigned" },
"properties": {
      "encryption": {
          "services": {
              "blob": { "enabled": true },
              "file": { "enabled": true }
          },
          "keySource": "Microsoft.Storage"
      },
      "isHnsEnabled": true,
      "networkAcls": "[json(parameters('networkAcls'))]",
      "accessTier": "[parameters('storageAccountAccessTier')]",
      "supportsHttpsTrafficOnly": true
```

@MarkWarneke

Unit Test

Azure Resource Manager Templates

@MarkWarneke

```powershell
# azuredeploy.adls.spec.ps1

param (
    $Path = (Join-Path $PSScriptRoot "azuredeploy.json")
)

# Test for template
$null = Test-Path $Path -ErrorAction Stop

# Test if template content is readable
$text = Get-Content $Path -Raw -ErrorAction Stop

# Convert the template to object
$json = ConvertFrom-Json $text -ErrorAction Stop

# Query for type that match 'storageAccounts'
$resource = $json.resources
        | Where-Object -Property "type" -eq "Microsoft.Storage/storageAccounts"

...
```

@MarkWarneke

```powershell
# azuredeploy.adls.spec.ps1

...
Describe "Azure Data Lake Generation 2 Resource Manager Template Unit" -Tag Unit {

    # Mandatory requirement of ADLS Gen 2 are:
    # - Resource Type is Microsoft.Storage/storageAccounts
    # - Kind is StorageV2
    # - Hierarchical namespace is enabled

    it "should have resource properties  present" {
        $resource | Should -Not -BeNullOrEmpty
    }

    it "should be of type Microsoft.Storage/storageAccounts" {
        $resource.type | Should -Be "Microsoft.Storage/storageAccounts"
    }

    it "should be of kind StorageV2" {
        $resource.kind | Should -Be "StorageV2"
    }

    it "should have Hns enabled" {
        $resource.properties.isHnsEnabled | Should -Be $true
    }
...
```

@MarkWarneke

```
# azuredeploy.adls.spec.ps1

...

    # Optional validation tests:
    # - Ensure encryption is as specified
    # - Secure Transfer by enforcing HTTPS

    it "should have encryption key source set to Storage " {
        $resource.properties.encryption.keySource | Should -Be "Microsoft.Storage"
    }

    it "should have blob encryption enabled" {
        $resource.properties.encryption.services.blob.enabled | Should -Be $true
    }

    it "should have file encryption enabled" {
        $resource.properties.encryption.services.file.enabled | Should -Be $true
    }

    it "should enforce Https Traffic Only" {
        $resource.properties.supportsHttpsTrafficOnly | Should -Be $true
    }
}
```

@MarkWarneke

Unit Test

PowerShell Deployment Scripts

@MarkWarneke

```powershell
# deploy.ps1 -WhatIf

[CmdletBinding(SupportsShouldProcess = $True)]

$Deployment = @{
    ResourceGroupName     = $rg
    TemplateFile          = $tf
    TemplateParameterFile = $tpf
}

if ($PSCmdlet.ShouldProcess("ResourceGroupName $rg deployment of", "TemplateFile $tf")) {
    # Code that runs the actual deployment
    New-AzResourceGroupDeployment @Deployment
}
else {
    # Code that dry runs the deployment
    New-AzResourceGroupDeployment @Deployment -WhatIf
    # Code that ‚mocks‘ the deployment
    Test-AzResourceGroupDeployment @Deployment
}
```

@MarkWarneke

Acceptance Test

Azure Resources

@MarkWarneke

```powershell
# adls.acceptance.spec.ps1

param (
    # Name of the resource
    [Parameter(Mandatory)]
    [string]
    $Name,

    # Name of the resource group
    [Parameter()]
    [string]
    $ResourceGroupName
)

$adls = Get-AzStorageAccount -Name $resource.Name -ResourceGroupName $resource.ResourceGroupName

…
```

@MarkWarneke

```powershell
# adls.acceptance.spec.ps1

...

Describe "$Name Data Lake Storage Account Generation 2" {

    # Mandatory requirement of ADLS Gen 2 are:
    # - Resource Type is Microsoft.Storage/storageAccounts,
    #   as we know we are looking for this it is obsolete to check
    # - Kind is StorageV2
    # - Hierarchical namespace is enabled

    it "should be of kind StorageV2" {
        $adls.Kind | Should -Be "StorageV2"
    }

    it "should have Hierarchical Namespace Enabled" {
        $adls.EnableHierarchicalNamespace | Should -Be $true
    }

...
```

@MarkWarneke

```
# adls.acceptance.spec.ps1

...

    <#
      Optional validation tests:
       - Ensure encryption is as specified
       - Secure Transfer by enforcing HTTPS
    #>

    it "should enforce https traffic" {
        $adls.EnableHttpsTrafficOnly | Should -Be $true
    }

    it "should have encryption enabled" {
        $adls.Encryption.Services.Blob.Enabled | Should -Be $true
        $adls.Encryption.Services.File.Enabled | Should -Be $true
    }

    it "should have network rule set  default action Deny" {
        $adls.NetworkRuleSet.DefaultAction | Should -Be "Deny"
    }
}
```

@MarkWarneke

Integration Test

Azure Resource Manager deployment

@MarkWarneke

```powershell
# integration.Tests.ps1

Describe "Azure Data Lake Generation 2 Resource Manager Integration" -Tags Integration {

    BeforeAll {
        # Create test environment
        Write-Host "Creating test environment $ResourceGroupName, cleanup..."

        # Create a unique ResourceGroup
        # ‘unique‘ string base on the date
        # e.g. 20190824T1830434620Z
        # file date time universal format ~ 20 characters
        $ResourceGroupName = 'TT-' + (Get-Date -Format FileDateTimeUniversal)

        Get-AzResourceGroup -Name $ResourceGroupName -ErrorAction SilentlyContinue |
                Remove-AzResourceGroup -Force

        # Get a unique name for the resource too,
        # Some Azure Resources have a limitation of 24 characters
        # consider 20 for the unique ResouceGroup.
        $ResourceName = 'pre-' + $ResourceGroupName.ToLower()

        # Setup the environment
        $null = New-AzResourceGroup -Name $ResourceGroupName -Location 'WestEurope'
    }

...
```

@MarkWarneke

```powershell
# integration.Tests.ps1

...

    AfterAll {
        # Remove test environment after test
        Write-Host "Removing test environment $ResourceGroupName..."

        Get-AzResourceGroup -Name $ResourceGroupName |
                Remove-AzResourceGroup -Force -AsJob
    }

    # Deploy Resource
    New-AzResourceGroupDeployment -ResourceName $ResourceName `
        -ResourceGroupName $ResourceGroupName


    # Run Acceptance Test
    . $PSScriptRoot/acceptance.spec.ps1 -ResourceName $ResourceName `
        -ResourceGroupName $ResourceGroupName
}
```

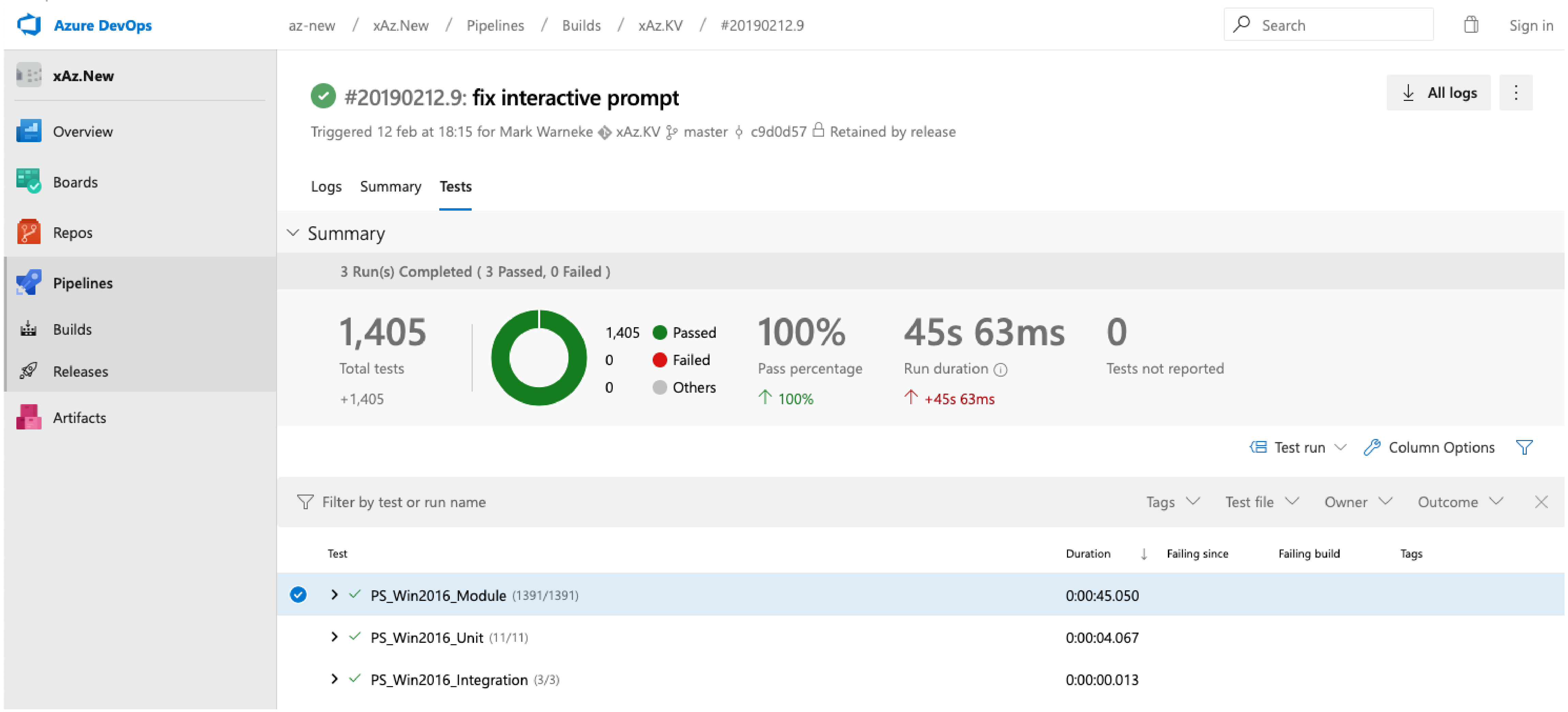
@MarkWarneke

Test Dashboard

Azure DevOps Test

@MarkWarneke

https://aka.ms/az.new


Azure DevOps   az-new / xAz.New / Pipelines / Builds / xAz.KV / #20190212.9   Search   Sign in

xAz.New
Overview
Boards
Repos
Pipelines
Builds
Releases
Artifacts

#20190212.9: fix interactive prompt

All logs

Triggered 12 feb at 18:15 for Mark Warneke xAz.KV master c9d0d57 Retained by release

Logs Summary Tests

Summary

3 Run(s) Completed (3 Passed, 0 Failed)

1,405		1,405 ● Passed	100%	45s 63ms	0
Total tests		0 ● Failed	Pass percentage	Run duration	Tests not reported
+1,405		0 ● Others	↑ 100%	↑ +45s 63ms	

Test run   Column Options

Filter by test or run name   Tags   Test file   Owner   Outcome

Test	Duration	Failing since	Failing build	Tags
✓ PS_Win2016_Module (1391/1391)	0:00:45.050			
✓ PS_Win2016_Unit (11/11)	0:00:04.067			
✓ PS_Win2016_Integration (3/3)	0:00:00.013			

@MarkWarneke

```yaml

# azure-pipelines.yml

steps:
  - task: AzurePowerShell@4
    inputs:
      azureSubscription: $(azureSubscription)
      scriptType: "FilePath"
      # The name of the script where the pester test setup is located
      scriptPath: $(Build.SourcesDirectory)\Invoke-Pester.ps1
      scriptArguments: -OutputFormat 'NUnitXml‘ `
                        -OutputFile 'TestResults.Pester.xml' -PassThru'
      azurePowerShellVersion: "latestVersion"
      errorActionPreference: "continue"

  - task: PublishTestResults@2
    inputs:
      # Make sure to use the 'NUnit' test runner
      testRunner: "NUnit" # !!!
      testResultsFiles: "**/TestResults.Pester.xml"
      testRunTitle: "PS_Win2016_Unit"
      # Make the whole pipeline fail if a test is failed
      failTaskOnFailedTests: true
    displayName: "Publish Unit Test Results"
    condition: in(variables['Agent.JobStatus'], 'Succeeded', 'SucceededWithIssues', 'Failed')


```

@MarkWarneke