PowerShell Conference Europe 2019

Hannover, Germany

June 4-7, 2019

Lessons learned from a large-scale infrastructure as code project

Mark Warneke

After this Session

- I am able to **develop a mature "Infrastructure As Code"** project from scratch using a Test-Driven development approach, avoiding common pitfalls and getting a heads up in necessary considerations, tools and best practices

- I can **build sophisticated Azure Release Pipelines** that leverage advanced testing scenarios using Azure Resource Manager Templates, PowerShell tooling to support an advanced "Infrastructure As Code" project

Agenda

Introduction

Architecture

Demo

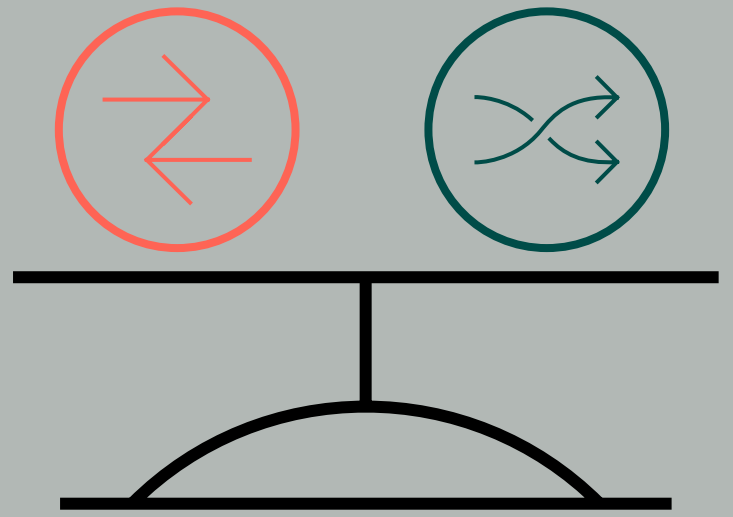
By viewing cloud computing as a starting point for IT automation, companies may be able to have it all: scalability, agility, flexibility, efficiency, and cost savings.
But that's only possible by building up both **automation and cloud** capabilities.

– McKinsey

@MarkWarneke

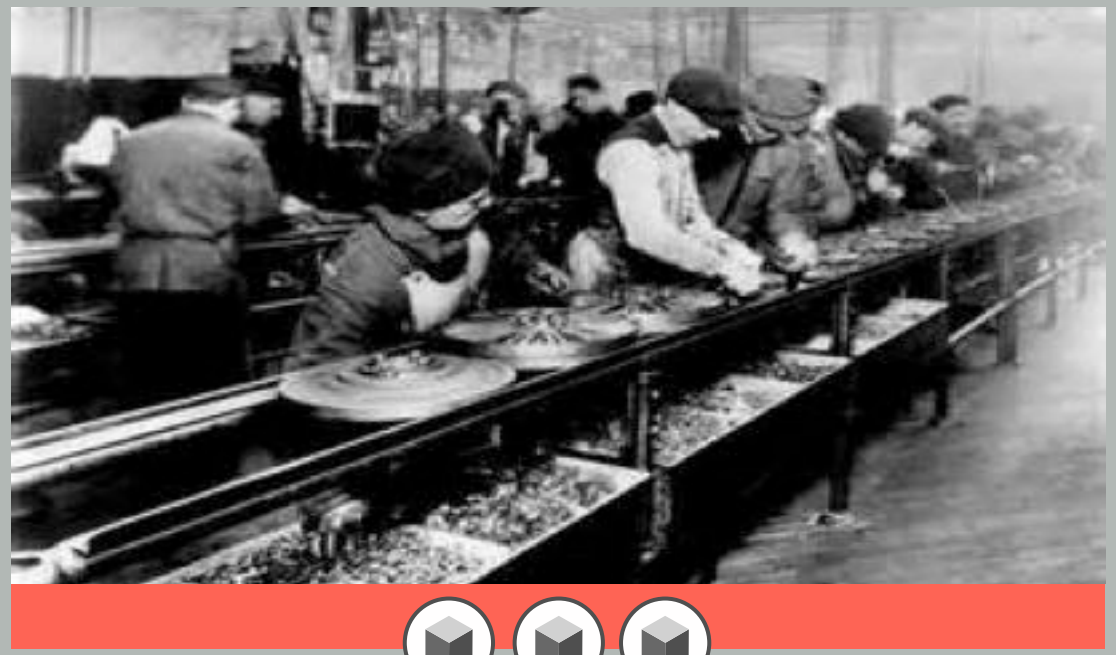
What is the challange?

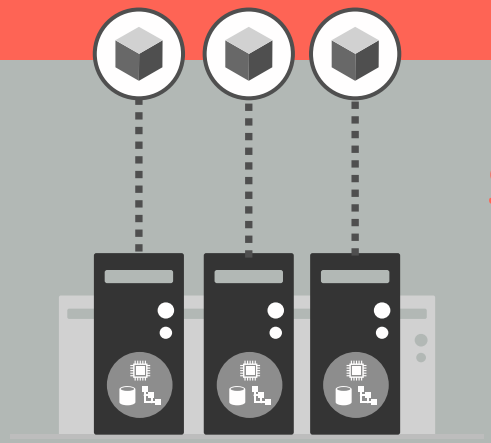
Control



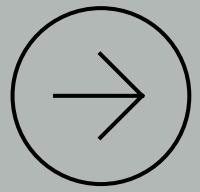
Speed
Agility

Why change?





Servers



Services

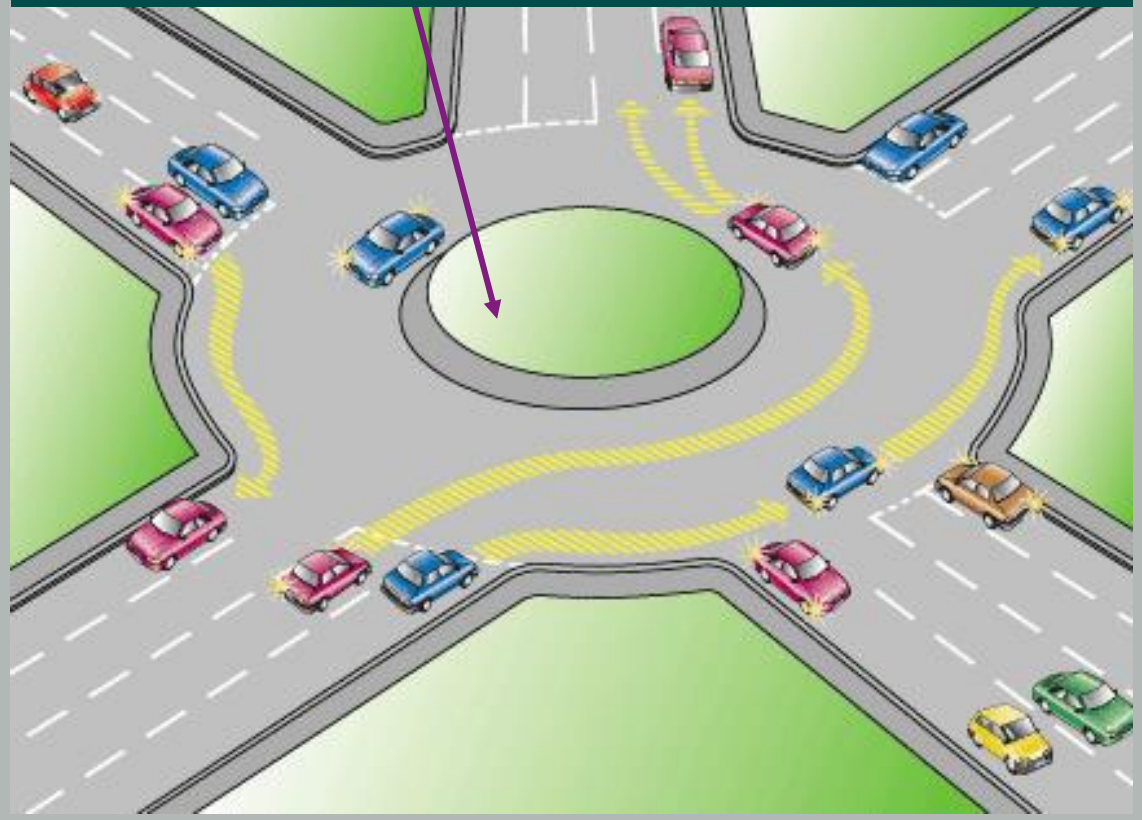
@MarkWarneke

Paradigm shift

Enforce/Control

Controlled & *central* responsibility

Enable/Support



Freedom & *delegated* responsibility

@MarkWarneke

What organization want

Control → Secure, predictable, and flexible service delivery and operations capability (end to end traceability).

Innovation → Faster business innovation through adoption of cloud services.

Speed/Agility → Business agility and reduced time-to-market through efficient DevOps teams.

Costs → Efficient use of public cloud scale.

DevOps benefits based on research

Comparing elite DevOps performers against low performers,
we find that elite performers have...

46 times more

Frequent code deployments

2,555 times faster

Lead time from commit to deploy

7 times lower

Change failure rate
(changes are 1/7 as likely to fail)

2,604 times faster

Time to recover from incidents

Source: 2018 State of DevOps Report DORA

@MarkWarneke

A Cloud Center of Excellence (CCoE) is a **cross-functional** team of people responsible for **developing** and **managing the cloud** strategy, governance, and best practices that the rest of the organization can **leverage to transform the business** using the cloud.

– Cloud Management Report

https://cloudcheckr.com/document/cloud-management-report/

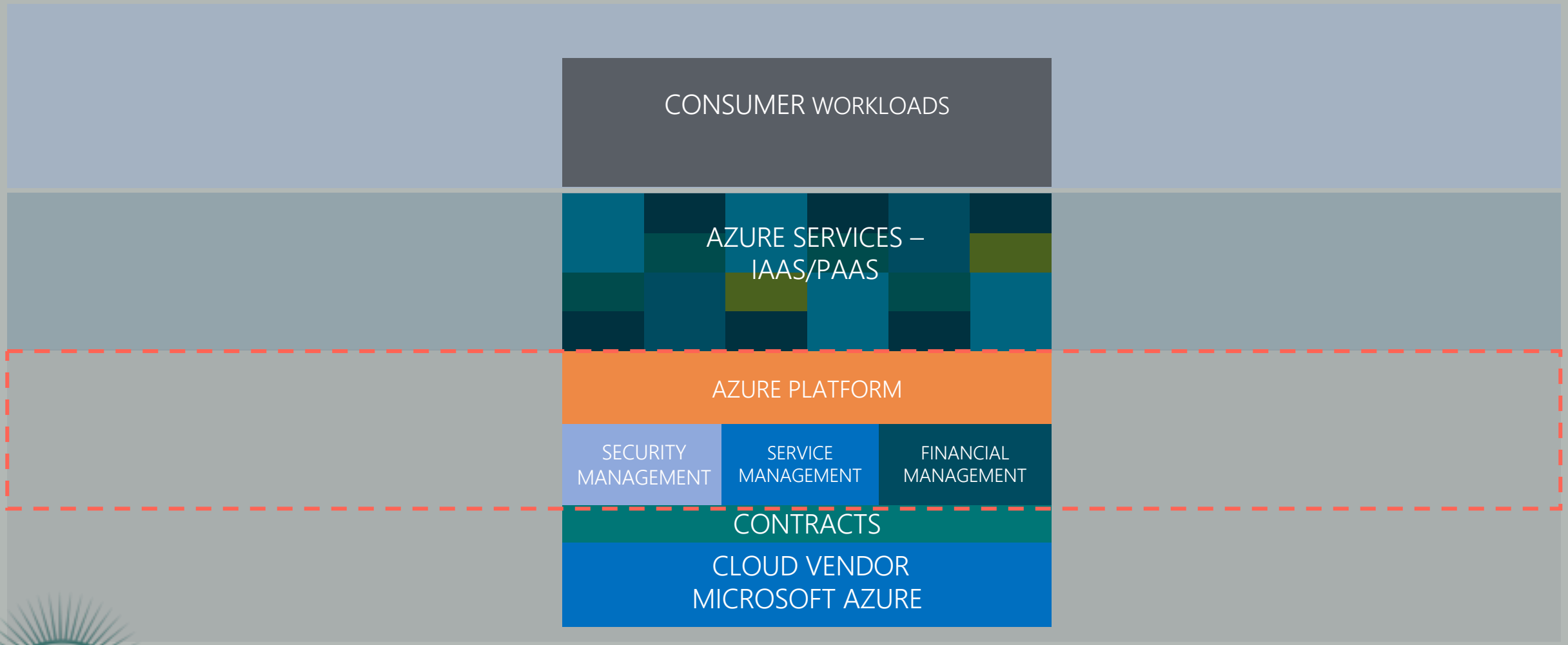
@MarkWarneke

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"

Scope of CCoE



CCoE needs to: find a tremendous amount of support from the executive team

Stephen Orban

https://medium.com/aws-enterprise-collection/how-to-create-a-cloud-center-of-excellence-in-your-enterprise-8ed3a97adcc6

@MarkWarneke

Speed and stability – no compromise

Enable agility with DevOps

Incentivize desired behavior

Insights vs control

Balance standardization / flexibility (mature over time)

Stay in control

End-to-end traceability

Smart governance

Identity & data centric controls

Technologies & Tools

ARM - Terraform	Azure DevOps – Jenkins
PowerShell - Python	Git – TFVC - Subversion
Ansible – Chef – PS DSC	Pester - RSpec - xUnit

PowerShell: Framework

Pester

Test & Validation @nohwnd

ARM-Templates

psake
@devblackops

azure-pipelines.yml

platyPS
@xvorsx

PSDepend
@psCookieMonster

PowerShell Module

Generated by Plaster → requires Az Module

@r_keith_hill -@daviwil - @neongreenie

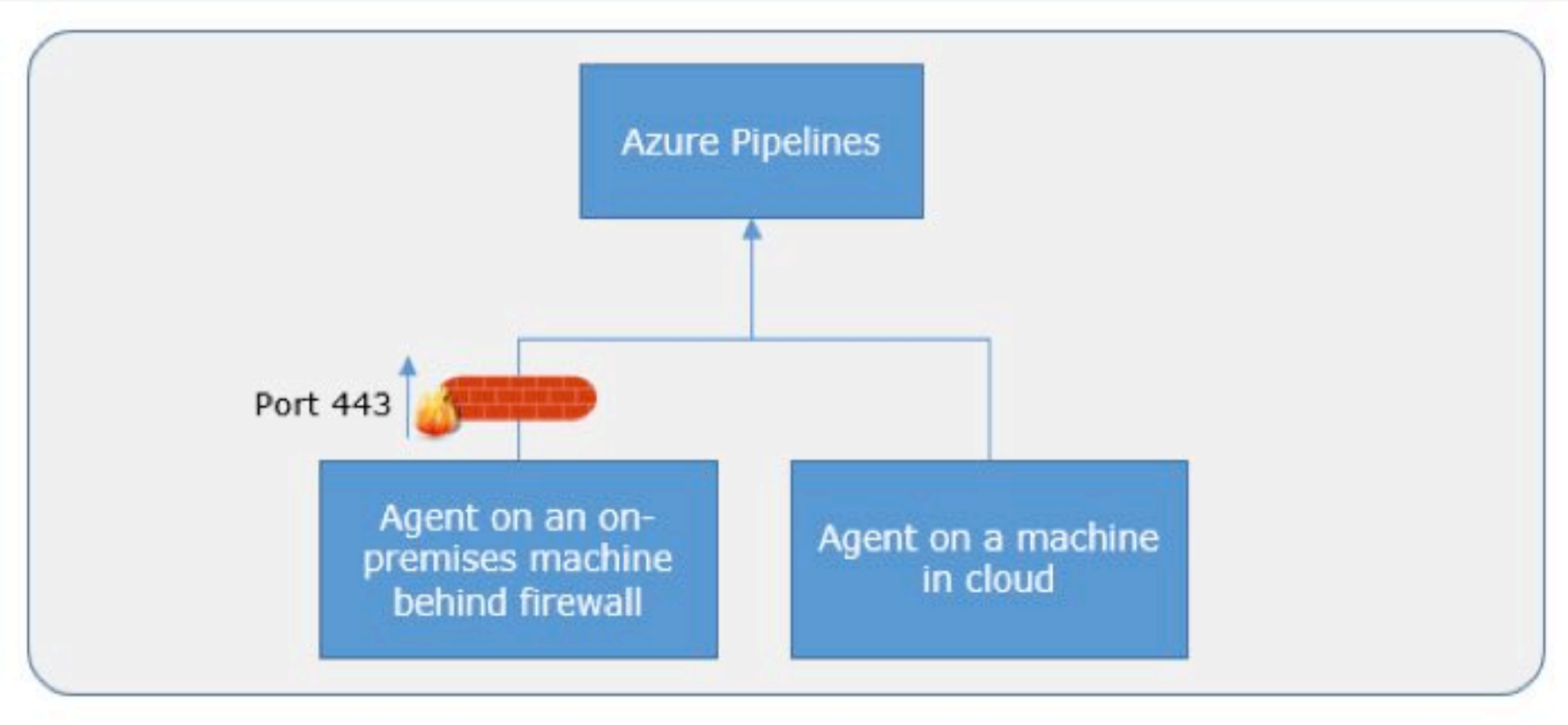
Azure DevOps
CI/CD

Azure
Automation
Functions

VSCode
Extensions

@MarkWarneke

Azure Pipelines agents



Environment

„Baseline Infrastructure “

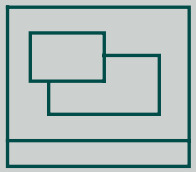
Reliable

Stable

Application

„Cloud Native App“

Fast Deployment



Focus on Requirements

User

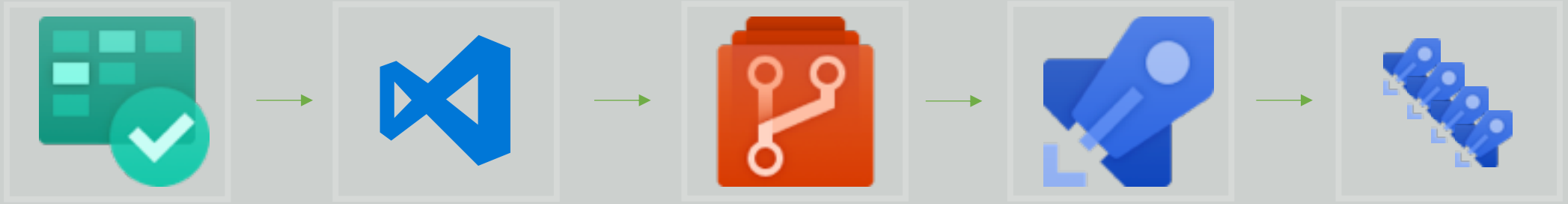
„Self-Service“

Accessibility

Self-Service

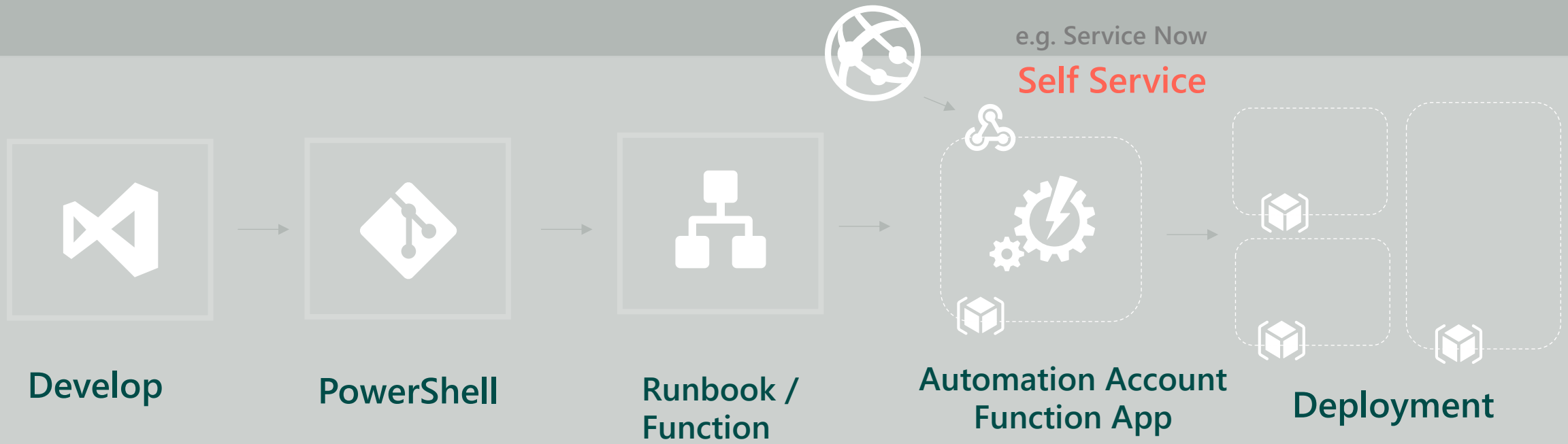
@MarkWarneke

Release: Hub Deployment



@MarkWarneke

Release: Self Service



DEMO

- aka.ms/az.new
- aka.ms/az.new/resources

@MarkWarneke

Summary

- Visit aka.ms/az.new to review content presented
- Review aka.ms/az.new/resources to look into the sources
- Look into building a CCoE to increase quality and maturity

Questions?

Use the conference app to vote for this session:
https://my.eventraft.com/psconfeu

@MarkWarneke

Sessions at PSConfEU 2019

Title	Speaker	Comment
PowerShell in Azure Functions	Joey Aiello, Tyler Leonhardt	Runing Interaktive IaC deployments
Pester + Azure (Monitor + Automation)	Mateusz Czerniawski	Running Azure Monitoring and Test
Extend your PowerShell skills by creating Azure DevOps Extension	Stefan Stranger	Modules to DevOps Extensions
OS image pipeline: Packer, PowerShell, DSC & Chocolatey	Gael Colas	IaC, golden image creation
Automate hybrid and cloud environments using Azure Automation	Jan Egil Ring	Automation Account upload from modules using ci/cd
Azure PowerShell vs Azure CLI: Duel at the command line	Aleksander Nikolic	Managing azure at command line

@MarkWarneke

Cloud Governance

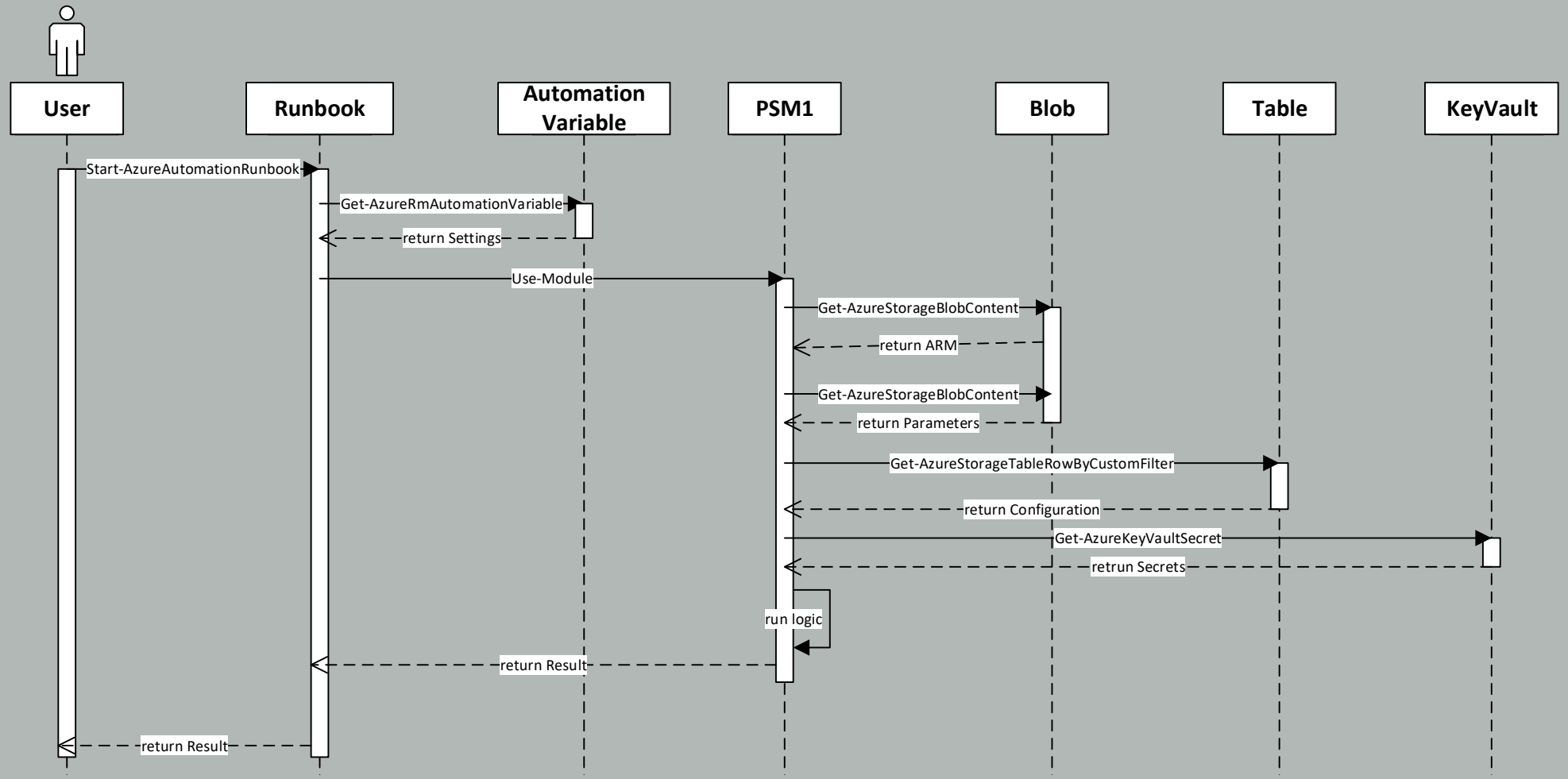
Cloud Native

Organize Resources

Policy & Blueprints

Security Center

@MarkWarneke



@MarkWarneke

<ComponentName>

.vscode -- *VSCode configuration*

launch.json

settings.json -- *VSCode Code Formating*

tasks.json -- *VSCode Code automation tasks*

Static -- *Static files per component like ARM templates*

[Configuration] -- *if more then one arm template can be used, distinguish with folders*

Azuredeploy.json * -- see ARM Template Nameconvention*

[Classes]

docs

en-US

about_<ComponentName>.help.txt -- *files should be generated using PowerShell Help*

Public -- *Public functions of module : gets exported to user*

Get-<Functionname>.ps1

Set-<Functionname>.ps1

New-<Functionname>.ps1

Private -- *Private functions of module : are for module internal use*

<Functionname>.ps1

Tests -- *Pester tests : should at least contain a test per public function and tests for module*

<ComponentName>.Tests.ps1 -- *tests for module import*

Get-<Functionname>.Tests.ps1 -- *tests per public function*

Set-<Functionname>.Tests.ps1

New-<Functionname>.Tests.ps1

Help.Exceptions.txt *-- Exceptions of script analyzer, should be as few as possible *

Help.Tests.ps1 -- *tests PowerShell help existing*

Project.Tests.ps1 -- *tests whole PowerShell project*

Shared.Tests.ps1 *-- tests shared components **

<ComponentName>.psm1 -- *dot sources all functions and exports public folder functions to user*

<ComponentName>.psd1

VSCode Extensions

- Formatting / Code Style
- Extension
  - PowerShell
  - Arm
  - Markdown
  - Brackets
  - LiveShare
  - Azure DevOps Integration

Storage Account Layout

```
$StorageAccount = „centralStorage"
$Container = "template"
$ModulePath = „<My>Component"
$TemplatePath = " $ModulePath/static/azuredeploy.json„
```

https://<StorageAccount>.blob.core.windows.net/<Container>/<Release./<Module>/<Template>

-> https://centralStorage.blob.core.windows.net/template/MyComponent/1.0.0/azuredeploy.json

```
-> https:// centralStorage.blob.core.windows.net/template/MyComponent/1.0.0/azuredeploy.json
-> https://
centralStorage.blob.core.windows.net/template/MyComponent/1.0.0/nestedtemplates/Provider.ResourceType.js
on
-> https:// centralStorage.blob.core.windows.net/template/MyComponent/1.0.0/scripts/customscript.ps1
```

@MarkWarneke

about_Speaker

Mark Warneke

Consultant

Microsoft

http://aka.ms/mark/

github.com/MarkWarneke

@MarkWarneke