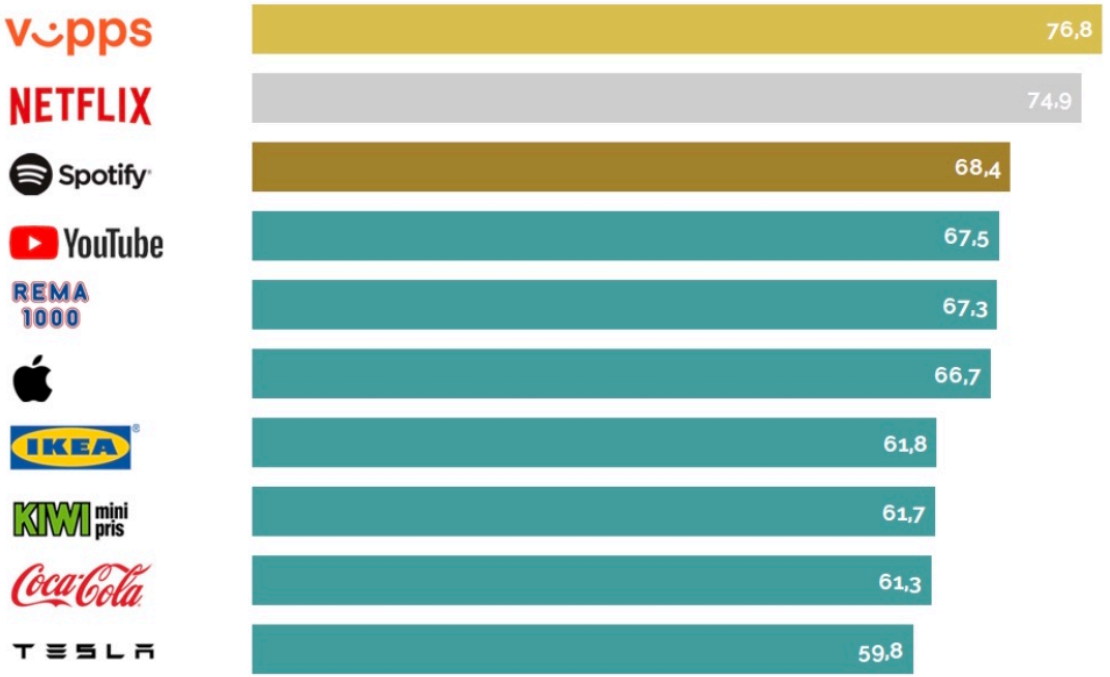
Vipps

From Monolith to Microservice: How Azure Powered Vipps To Become the No. 1 Payment Service in Norway.

Vipps: Key numbers

- **3.3 Million users** - almost the entire population of Norway.
- **YouGov BrandIndex 2018 Norway**



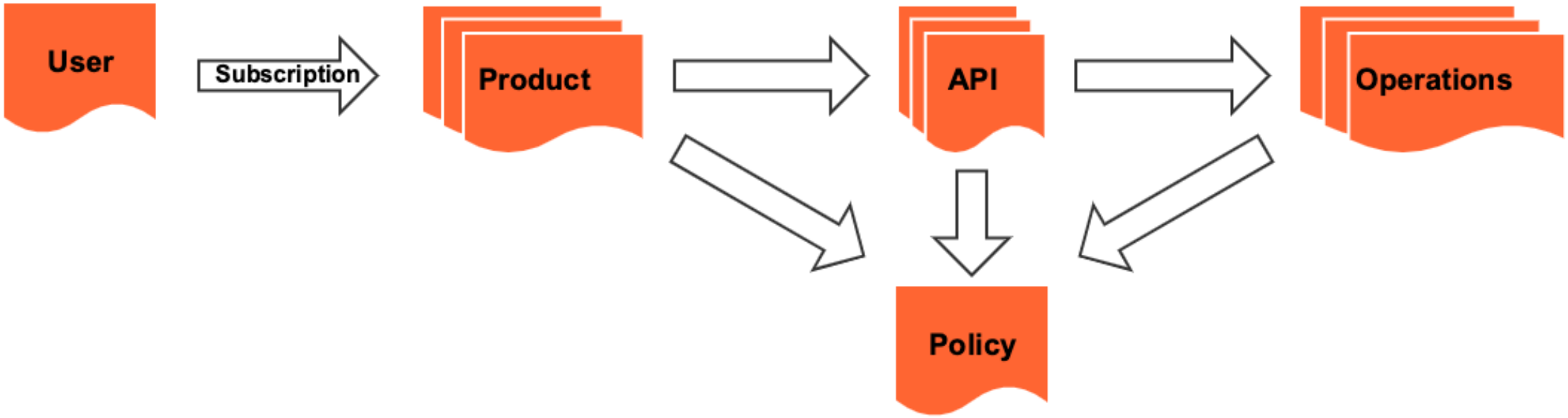
About Vipps

- Started 2015 as part on DNB
- After a month Vipps was downloaded one million times
- Merged 2018 with two of the strongest brands within payments in Norway.

Agenda

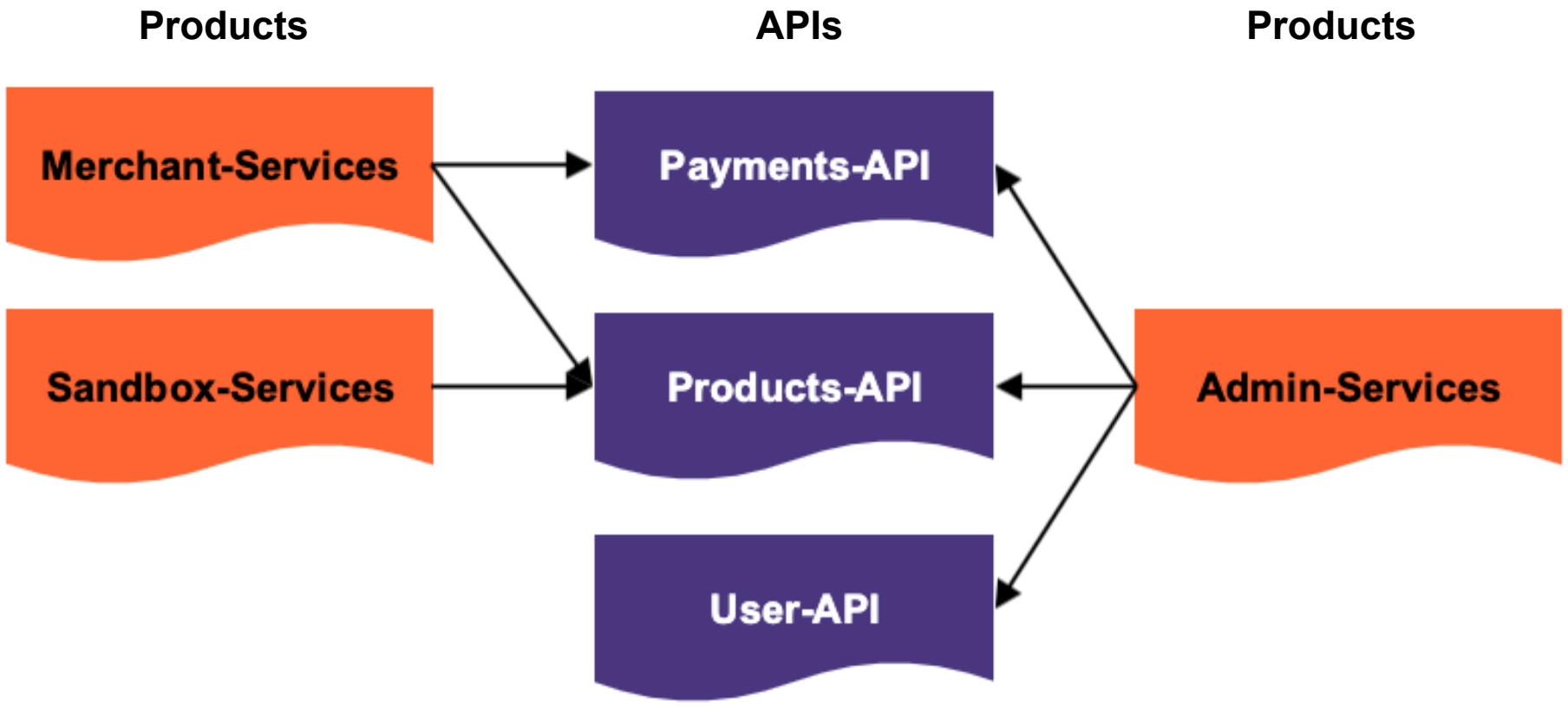
- **Basic concepts**
  APIs, Products, Users, Subscriptions
  How the policy model works
  Deployment options in Api-M.
- **Introduction to ARM templates**
- **4 Iterations of deployment to Api-M**
- **Api-M policies and how we manage them**
- **2 Iterations of deployment to AKS**

APIs, Products, Users, Subscriptions



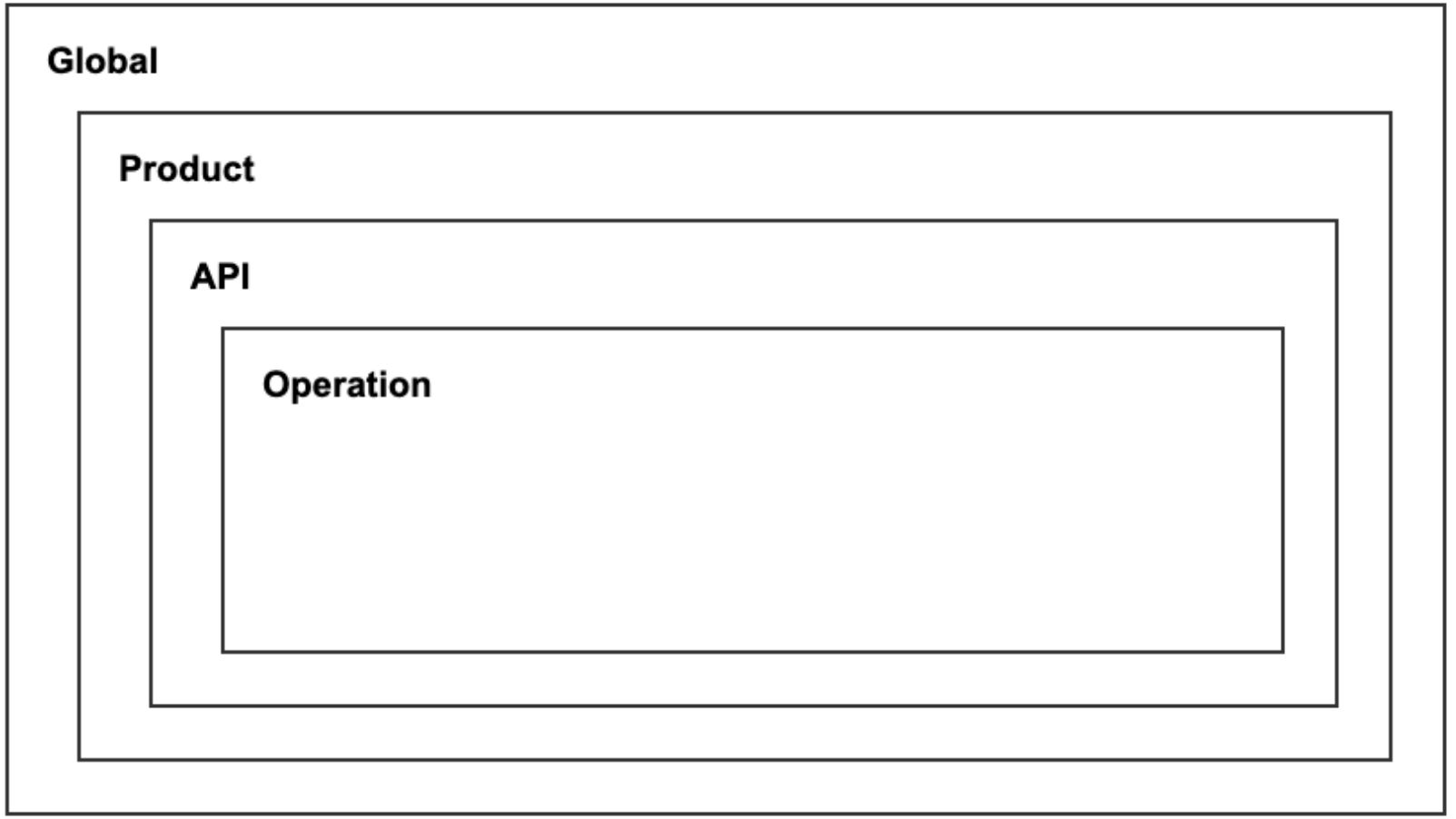
Change the behavior of a product API, operation

APIs, Products, Users, Subscriptions



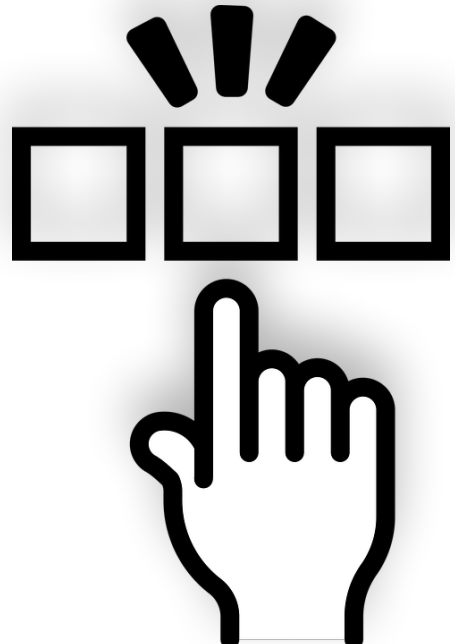
Policy Model

Change the behavior of a product API, operation



Deployment options

- Portal
- ARM templates
- Rest
- PowerShell - AzureRm/Az
- GitHub



Short introduction to ARM templates

Api-M
ARM templates

- Infrastructure
- APIs
- Products
- Named Values
- Groups
- Users
- Subscriptions

ARM templates: Terminology

- **Declarative syntax**
  Syntax that lets you state "Here is what I intend to create" without having to write the sequence of programming commands to create it.

- **Resource**
  A manageable item that is available through Azure (virtual network, virtual machine, storage account, web app, database, etc…)

- **Resource Group**
  Container that holds related resources for an Azure solution.

- **Azure Resource Manager (ARM) template**
  Json file that defines one or more resources to deploy to a resource group.

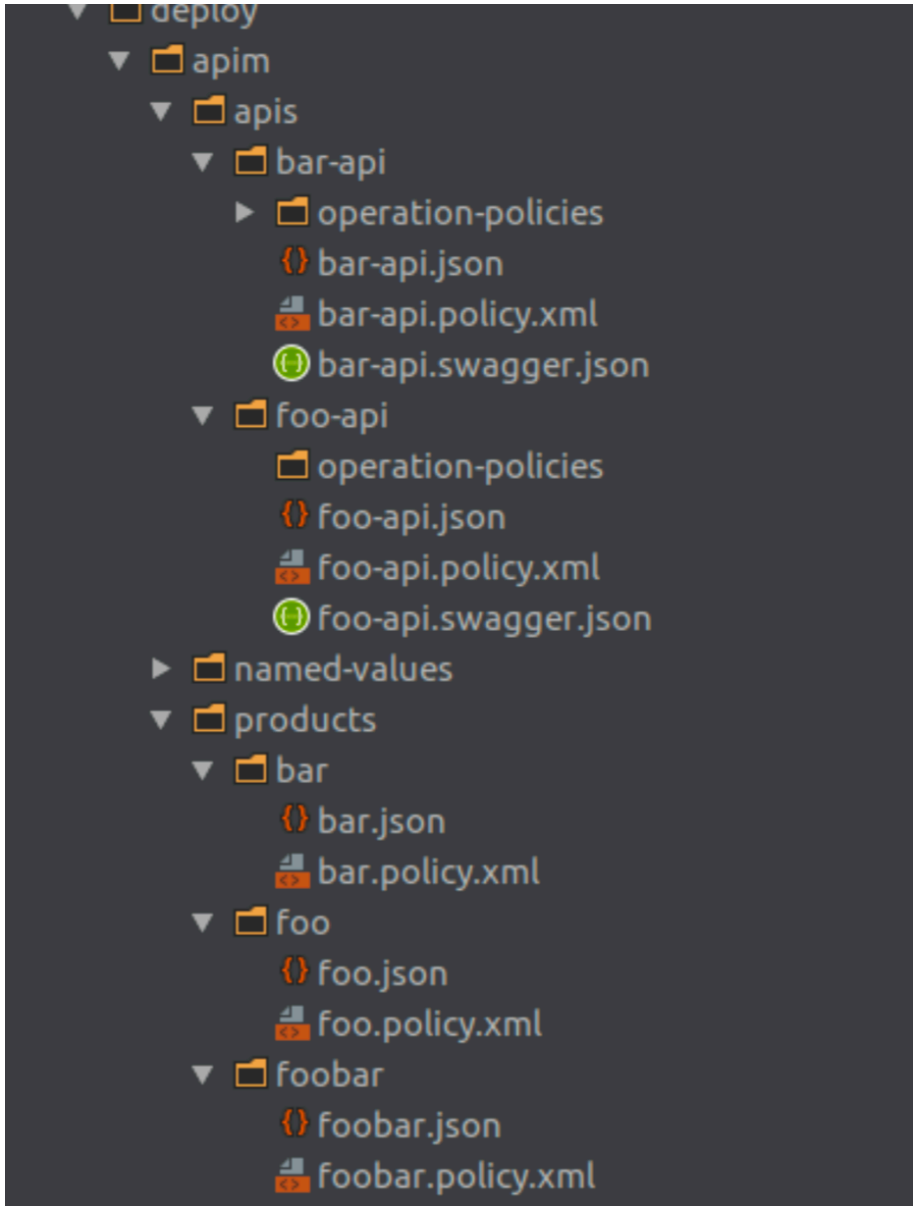
Api-M APIs

- ARM
- Swagger
- API policies
- Operation policies

Api-M Products

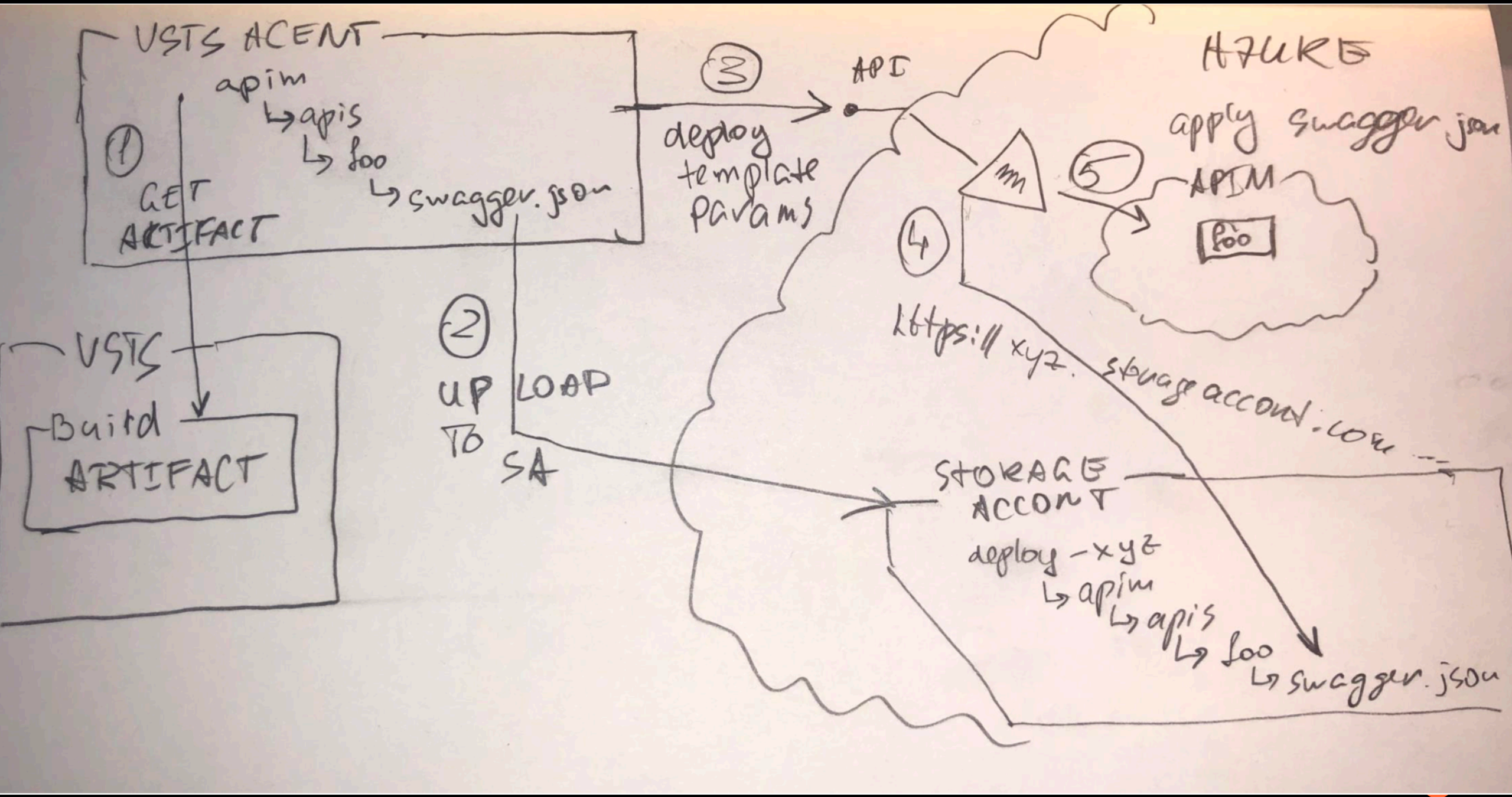
- **ARM**
- **Policy**

Api-M with ARM Folder structure



Code

- Blank ARM template
- ARM template for an API
- ARM template for a product
- ARM template for a large product


VSTS ACENT
apim
↳ apis
↳ foo
↳ swagger.json
1
GET ARTIFACT
3
deploy template params
API
AZURE
apply swagger json
APIM
foo
5
4
https:// xyz. storage account.com
2
UP LOAD TO SA
VSTS
Build ARTIFACT
STORAGE ACCONT
deploy-xyz
↳ apim
↳ apis
↳ foo
↳ swagger.json

Advantages

- **Declarative**
  Describes what to do, not how to do it.
- **“Packages” …**
  … what belongs together

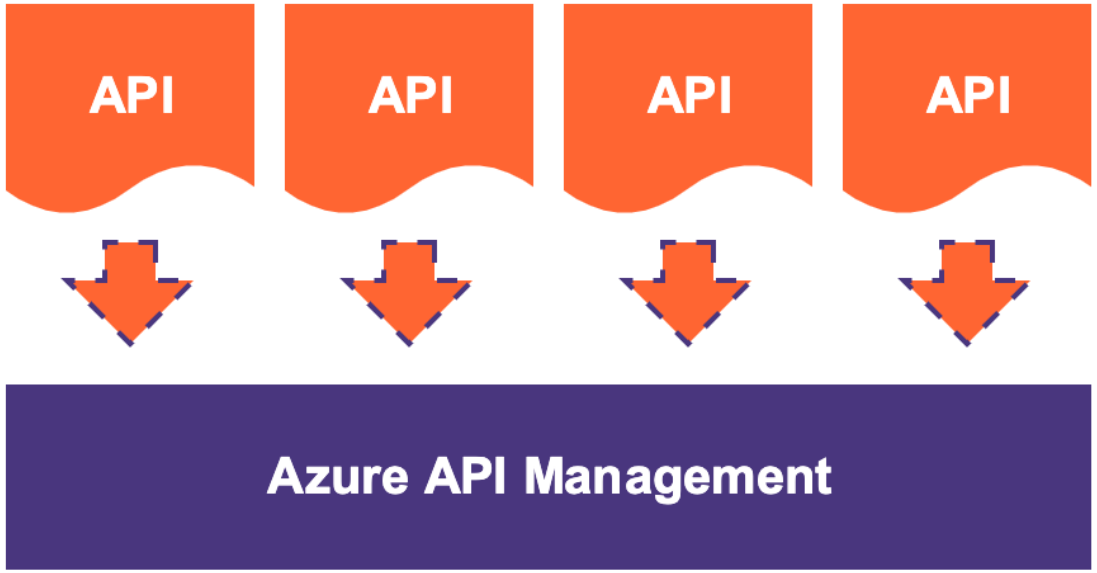
Disadvantages

- **Slower learning curve**
- **Specialized team.**
- **Does not describe state**
- **Need a storage account**

Api-M deployment iterations

Api-M deploy 1. Iteration

- Deploy all APIs at once with ARM
- Deploy all products at once with ARM
- Deploy all named values at once with ARM



Advantages

- Apim has always “correct” state
- Easy to manage

Disadvantages

- Expensive process
- Long deployment time
- Everything gets uploaded to storage account
- High risk

Api-M deploy 2. Iteration

- Deploy API separately with ARM templates
- Deploy product separately with ARM templates

Advantages

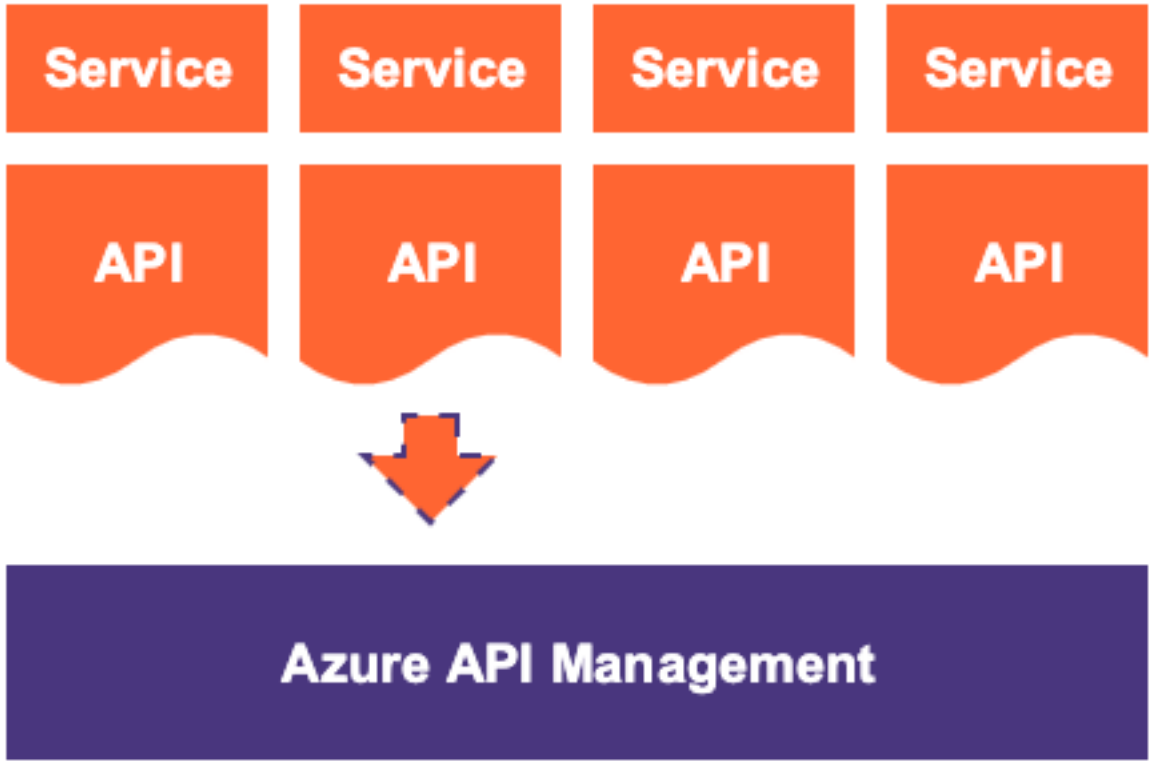
- Lower risk
- Faster deployment

Disadvantages

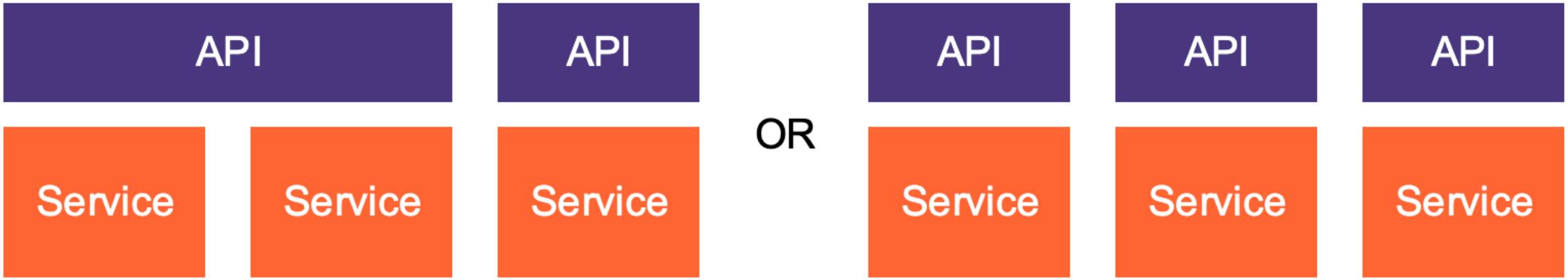
- Everything gets still uploaded to storage account
- Still no ownership

Api-M deploy
3. Iteration

- Moving APIs into team projects
- Teams get own CI/CD pipelines



Structuring APIs



A specialized team manages Api-M.

Increased ownership.

Advantages

- Teams take ownership
- Lower risk because the teams know there own APIs best

Disadvantages

- Complicated CI/CD pipeline
- Teams need to deal with ARM
- Requires a lot of support from specialized teams

Rest
PowerShell module - AzureRm/Az

Api-M deploy - 4. Iteration

Rest

PowerShell module - AzureRm/Az

Azure RM PowerShell 6.13.0

Search

AKS
Analysis Services
API Management
Add-AzureRmApiManagementApi ToProduct
Add-AzureRmApiManagementProduct ToGroup
Add-AzureRmApiManagementRegion
Add-AzureRmApiManagementUser ToGroup
Backup-AzureRmApiManagement
Export-AzureRmApiManagementApi
Get-AzureRmApiManagement
Get-AzureRmApiManagementApi
Get-AzureRmApiManagementApiRelease
Get-AzureRmApiManagementApiRevision
Get-AzureRmApiManagementApiVersion Set
Get-AzureRmApiManagement AuthorizationServer
Get-AzureRmApiManagementBackend

AzureRM.ApiManagement

This topic displays help topics for the Azure API Management Cmdlets.

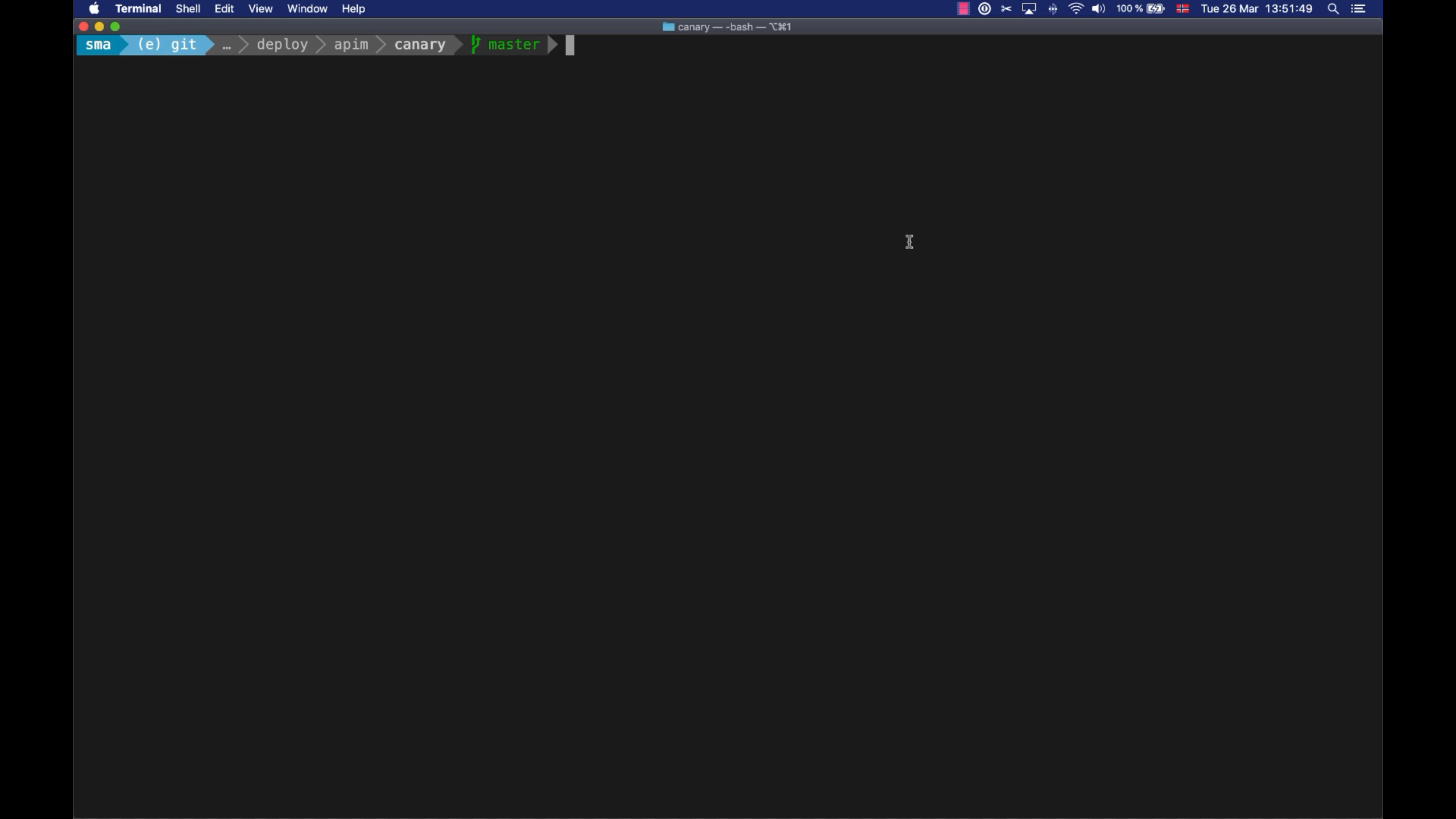
API Management

Add-AzureRmApiManagementApi ToProduct	Adds an API to a product.
Add-AzureRmApiManagementProduct ToGroup	Adds a product to a group.
Add-AzureRmApiManagementRegion	Adds new deployment regions to a PsApiManagement instance.
Add-AzureRmApiManagementUser ToGroup	Adds a user to a group.
Backup-AzureRmApiManagement	Backs up an API Management service.
Export-AzureRmApiManagementApi	Exports an API to a file.
Get-AzureRmApiManagement	Gets a list or a particular API Management Service description.
Get-AzureRmApiManagementApi	Gets an API.

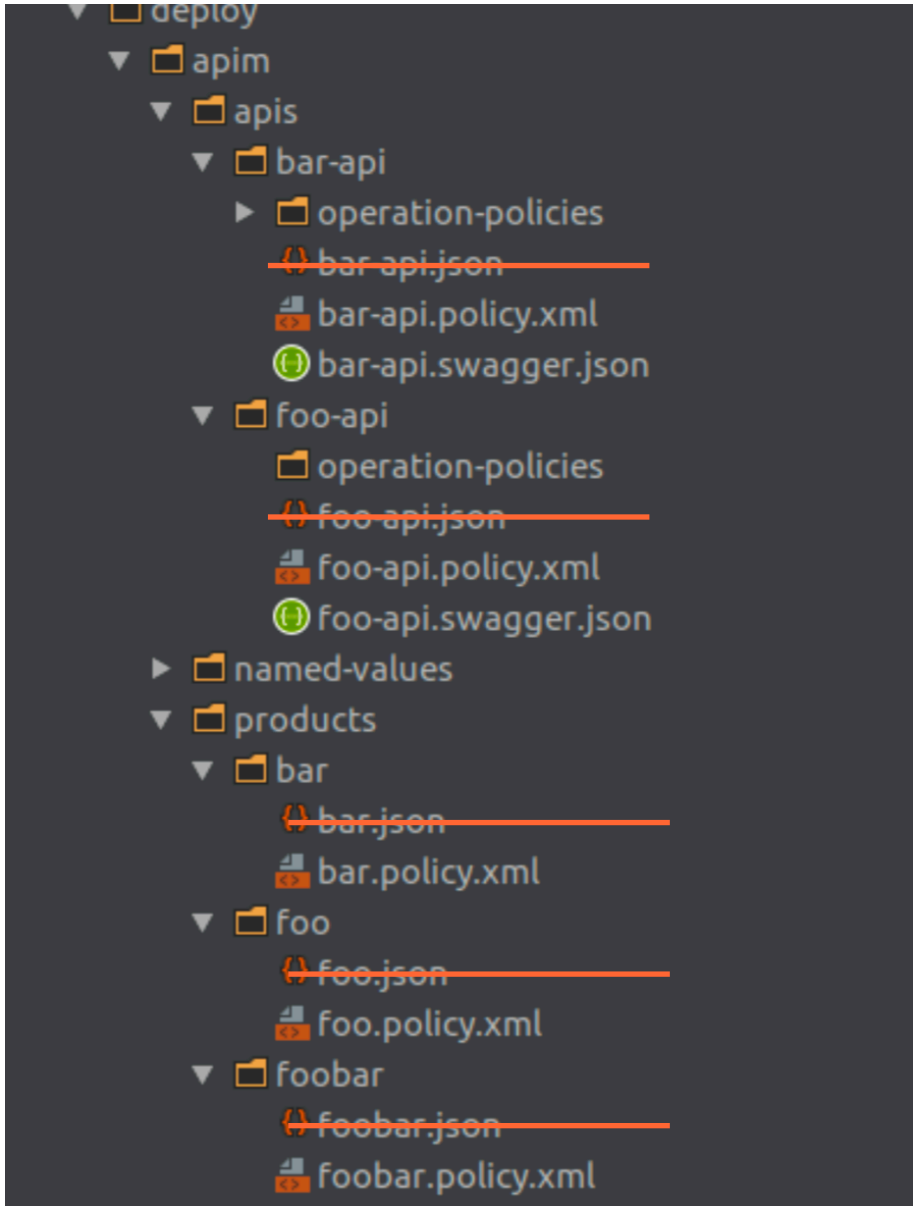
In this article
API Management

Code

- swagger.yaml
- API policy
- Named Values
- Product (add/delete)


Terminal Shell Edit View Window Help
100 % Tue 26 Mar 13:51:49
canary — -bash — ⌥⌘1
sma (e) git … deploy apim canary master

Api-M Rest Folder structure



Backend
Overview
Boards
Repos
Pipelines
Builds
Releases
Library
Task groups
Deployment groups
Test Plans
Artifacts

All pipelines > vippsas.k8s-app-conf - CD

Save | Release | View releases

Pipeline | Tasks | Variables | Retention | Options | History

APIM - SIT
Deployment process

Agent job
Run on agent

Task group: deploy-api $(ApiBasePath)
deploy-api

deploy-api

View YAML | Remove

Task version: 3.*

Display name *
Task group: deploy-api $(ApiBasePath)

ApiBasePath *
$(ApiBasePath)

apim-apiId *
$(apim-apiId)

apim-ApiPath *
$(apim-ApiPath)

arm_service_connection * | Manage
Azure test

environment *
$(environment)

Control Options

Output Variables

Backend
Overview
Boards
Repos
Pipelines
Builds
Releases
Library
Task groups
Deployment groups
Test Plans
Artifacts

Task groups > deploy-api

Tasks | History | References | Refresh | Save | Export | ...

deploy-api
Version 3.*

- Use Python 3.5
  Use Python Version
- Convert swagger yaml to json
  Bash
- Deploy $(apim-apiId) API
  Azure PowerShell

Display name *
Deploy $(apim-apiId) API

Azure Connection Type
Azure Resource Manager

Azure Subscription * | Manage
$(arm_service_connection)

Script Type
(•) Script File Path ( ) Inline Script

Script Path
$(System.DefaultWorkingDirectory)/apim_config/apim-config/DeployApi.ps1

Script Arguments
-GlobalApimConfigPath "$(System.DefaultWorkingDirectory)/apim_config/apim-config/globalApimConfig.json" -Environment "$(environment)" -SpecificationPath "$(System.DefaultWorkingDirectory)/$(apim-ApiPath)" -Path "$(ApiBasePath)" -ApiId "$(apim-apiId)"

ErrorActionPreference
Stop

[ ] Fail on Standard Error

Azure PowerShell version options

Advantages

- Easy to understand.
- No storage account needed

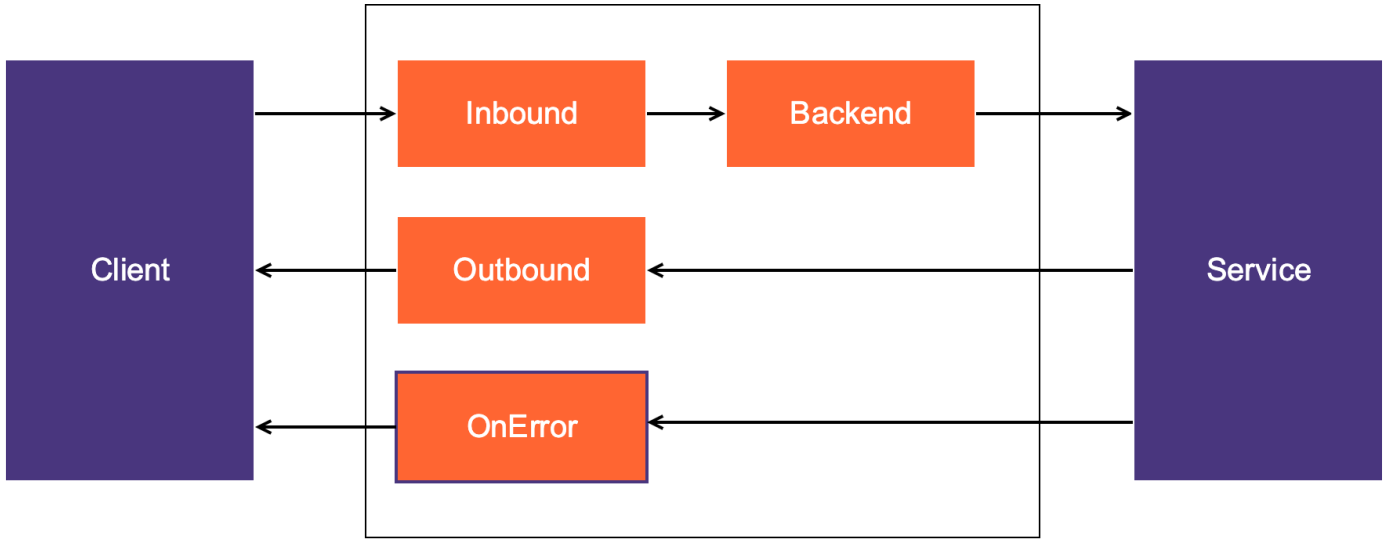
Disadvantages

- Understanding PowerShell
- Might be prone to errors
- Too much “magic”

Api-M policies

Policies

- **Change API behavior through configuration**
  Header validation
  Mocking
  Rate limit
  Monetizing Websites



Code

- API policy
- API policy with mock
- Mock policy template

AKS

Azure Kubernetes Service - AKS

- AKS manages a hosted Kubernetes environment …
- Deploy and manage containerized applications without container orchestration expertise …
- Eliminates the burden of ongoing operations and maintenance by provisioning, upgrading, and scaling resources on demand …

Api-M deployment iterations

AKS service deploy 1. Iteration

- **Teams create their own deploy routines**

Advantages:

Developers are in full control

Developers are having fun

Disadvantages:

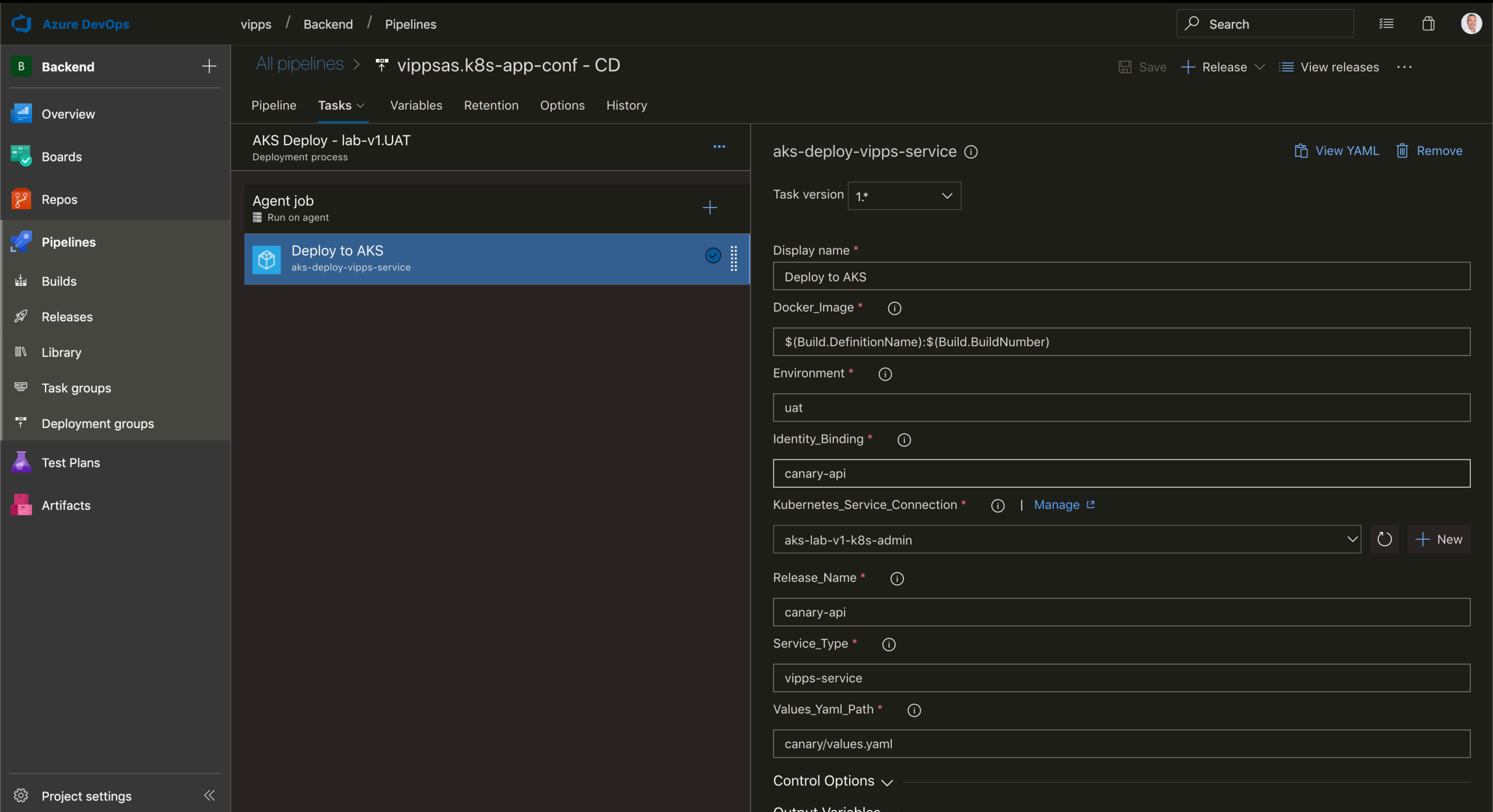
5 teams lead to 6 different deploy solutions

Security may not been handled good enough

AKS service deploy 2. Iteration

- Teams create an AKS service configuration file

CODE


Azure DevOps
vipps / Backend / Pipelines
Search
Backend
Overview
Boards
Repos
Pipelines
Builds
Releases
Library
Task groups
Deployment groups
Test Plans
Artifacts
Project settings
All pipelines > vippsas.k8s-app-conf - CD
Save
Release
View releases
Pipeline
Tasks
Variables
Retention
Options
History
AKS Deploy - lab-v1.UAT
Deployment process
Agent job
Run on agent
Deploy to AKS
aks-deploy-vipps-service
aks-deploy-vipps-service
View YAML
Remove
Task version
1.*
Display name *
Deploy to AKS
Docker_Image *
$(Build.DefinitionName):$(Build.BuildNumber)
Environment *
uat
Identity_Binding *
canary-api
Kubernetes_Service_Connection *
Manage
aks-lab-v1-k8s-admin
New
Release_Name *
canary-api
Service_Type *
vipps-service
Values_Yaml_Path *
canary/values.yaml
Control Options

Advantages

- Developers don't need to understand how to deploy
- Security and logging is been taken care of

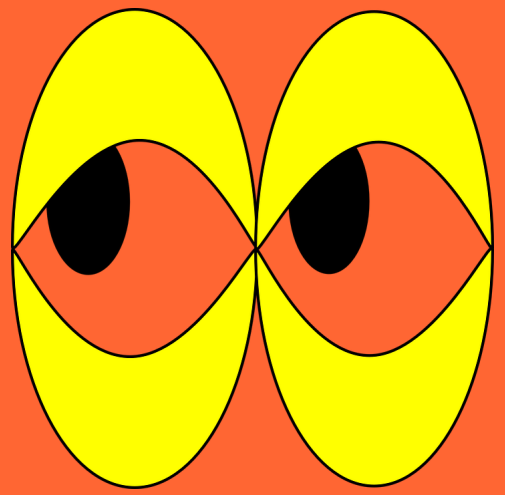
Disadvantages

- No relation to Kubernetes
- Harder to debug
- A lot of “magic” happening

Summarise 3 best practices

- **Simplify as much as possible, or developers get frustrated.**
- **Rest over ARM for Api-M deployments.**
- **Hide complexity with an AKS service configuration file.**

What you have seen



- How we deploy to Api-M.
- How developer teams can manage their APIs.
- How dev teams can deploy their services to AKS without too much knowledge about the technology behind.


Vipps