#GlobalAzure

#GABMUGPeru

#GABMUGPeru

Implementando IaC con Bicep y Azure DevOps

Ernesto Cárdenas Cangahuala
www.consultorinternet.com

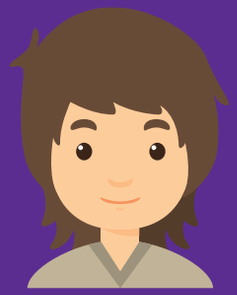
Microsoft® MVP Most Valuable Professional

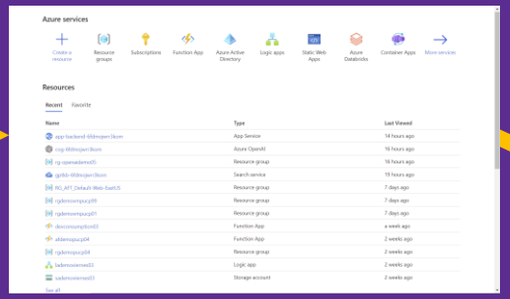
#GlobalAzure

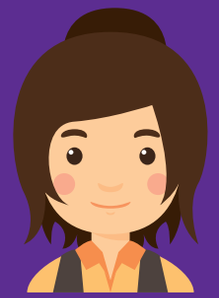
#GABMUGPeru

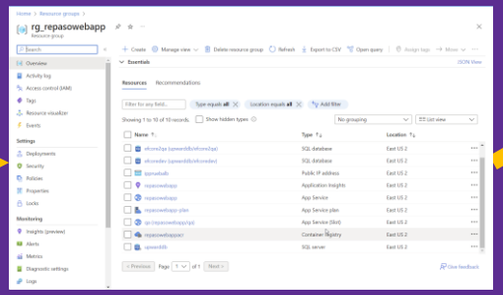
¿Cómo solemos provisionar nuestra Infraestructura?

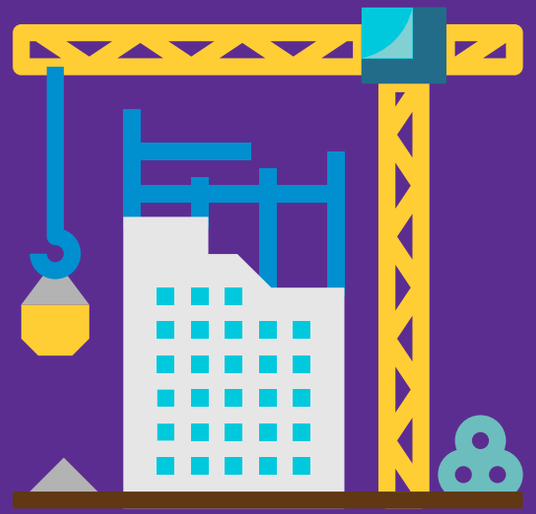
Mediante el portal





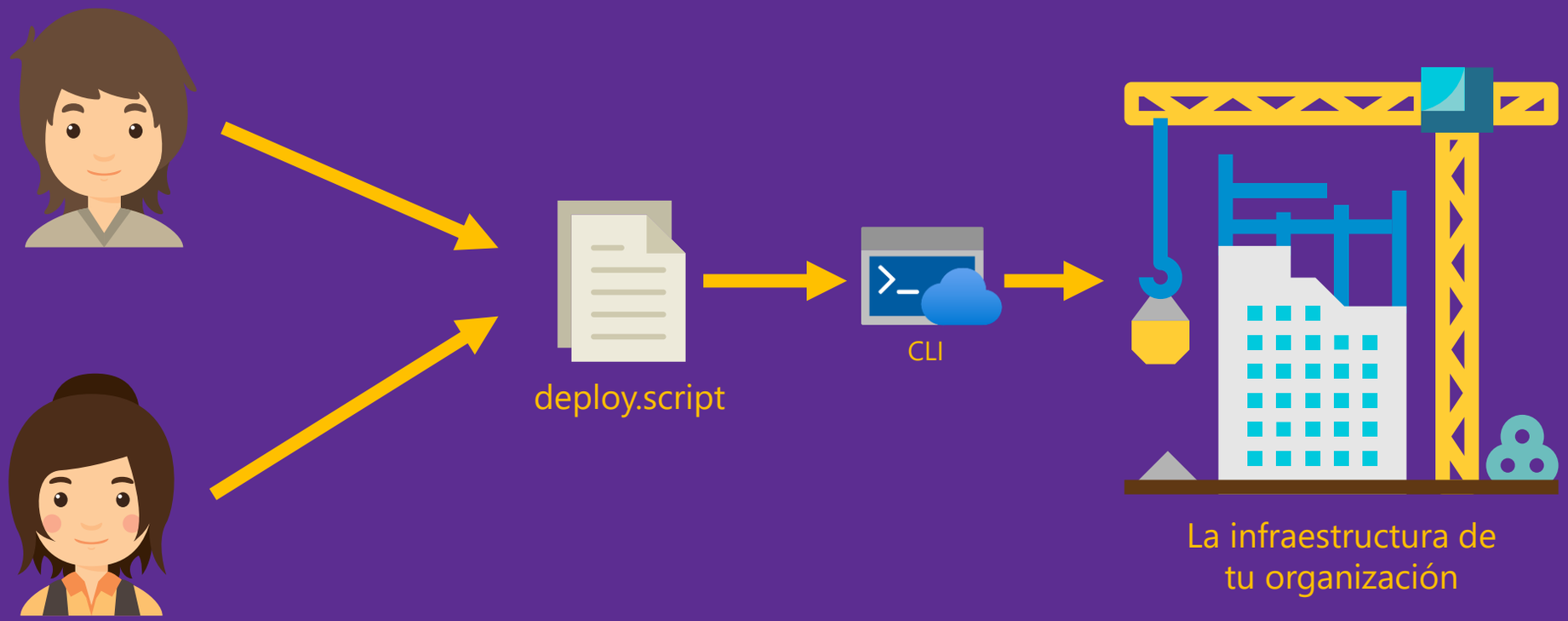






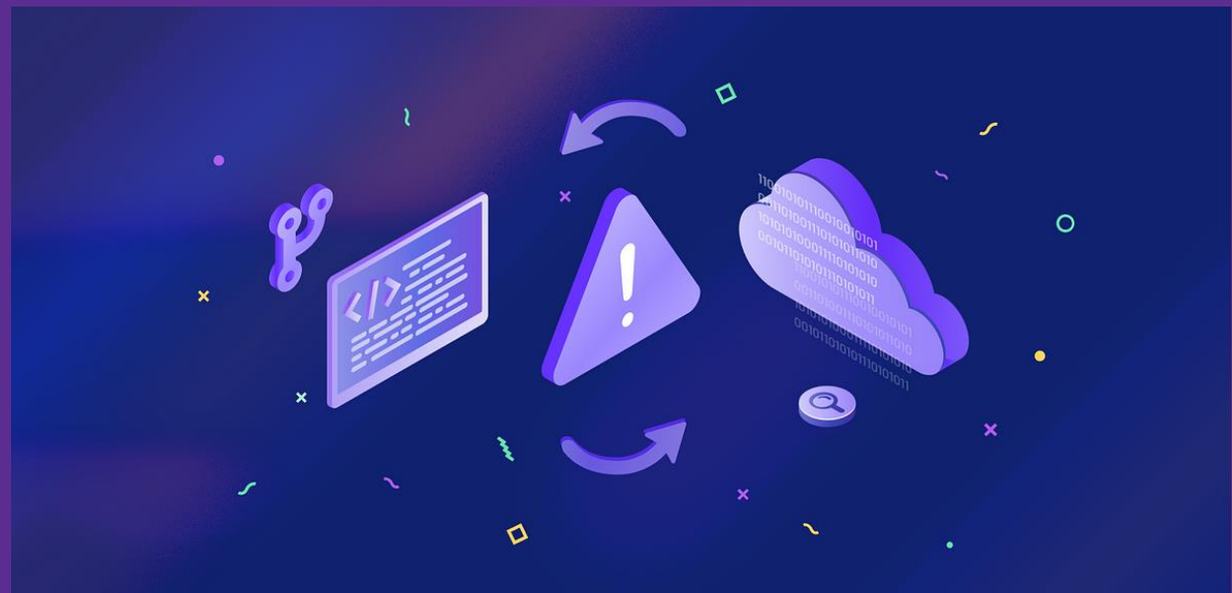
La infraestructura de tu organización

Usando el CLI



Y… Funciona!! Pero…

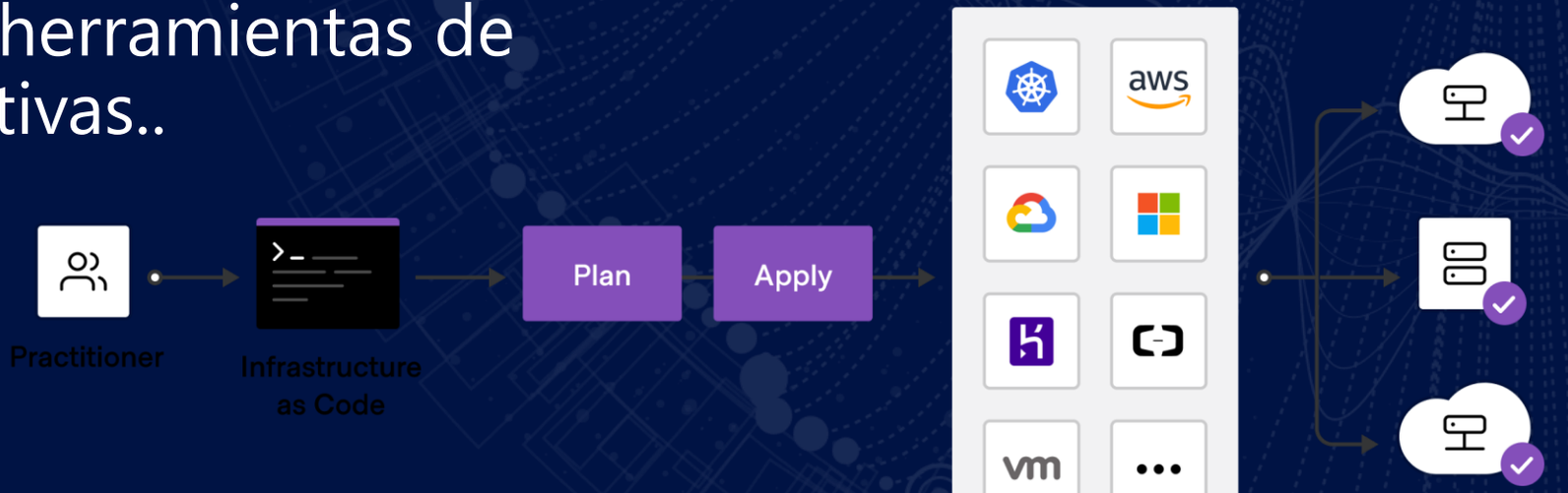
- Esta sujeto a errores manuales
- Si se efectuan cambios despues, es dificil saber cual era el estado original
- Requiere una mayor coordinación entre equipos lo que ralentiza el proceso
- No se garantiza la **idempotencia** entre ejecuciones



Veamos que se puede hacer

¿Qué es Infraestructura como Código?

- La infraestructura como código (IaC) es la administración de la infraestructura (redes, máquinas virtuales, equilibradores de carga y topología de conexión) en un modelo **descriptivo, legible por la maquina**, utilizando el control de versiones para almacenar los archivos, en lugar de herramientas de configuración interactivas..



Que no es IaC

- Automatización a través de scripts (**Imperativo**)
- Los scripts dicen qué se debe lograr y cómo
- La fuente de la verdad vive en la infraestructura aprovisionada
- Los cambios se realizan después de la implementación en los dispositivos
-

Ejemplo de Imperativo

```
# Create a resource group
az group create \
--name sample-rg \
--location eastus

# Create an app service plan
az appservice plan create \
--resource-group sample-rg \
--name my-sample-asp

# Create a web app
az webapp create \
--resource-group sample-rg \
--plan my-sample-asp \
--name my-sample-webapp
```

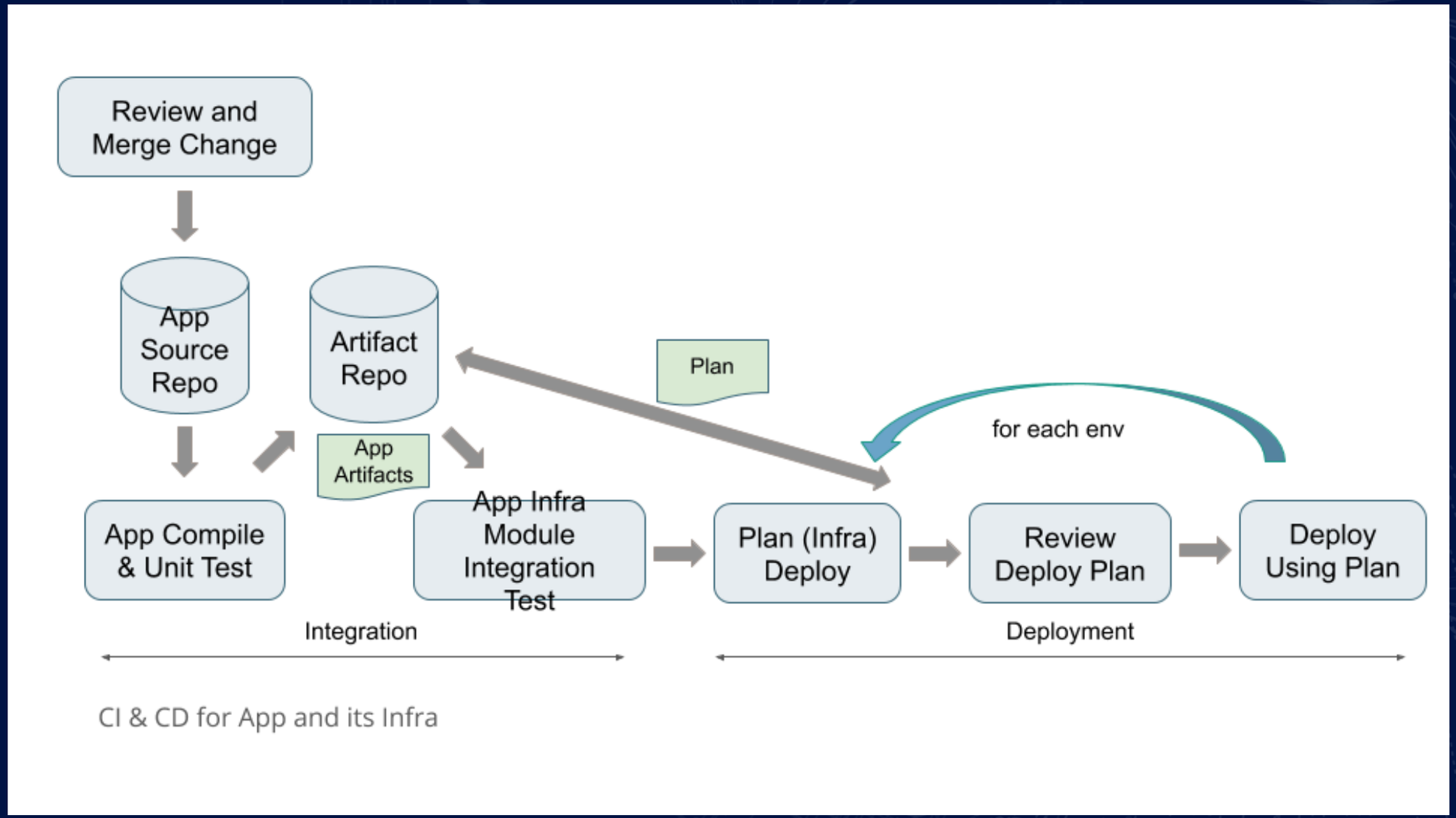
IaC si es…

- Automatización mediante archivos de configuración (**declarativo**)
- Aprovisionamiento a través de archivos legibles por máquina
- Definir lo que un programa debe lograr, pero no cómo
- La fuente de la verdad vive en el código
- Se realizan cambios en el código y se vuelven a implementar
- Idempotente – proporciona el mismo resultado cada vez que se aplica

Ejemplo de Declarativo

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "webAppName": {
      "type": "string",
      "defaultValue" : "AzureLinuxApp",
      "metadata": {
        "description": "Base name of the resource such as web app name and app service plan "
      },
      "minLength": 2
    },
    "sku":{
      "type": "string",
      "defaultValue" : "S1",
      "metadata": {
        "description": "The SKU of App Service Plan "
      }
    },
    "linuxFxVersion" : {
        "type": "string",
        "defaultValue" : "php|7.0",
        "metadata": {
          "description": "The Runtime stack of current web app"
        }
    },
    "location": {
      "type": "string",
      "defaultValue": "[resourceGroup().location]",
      "metadata": {
        "description": "Location for all resources."
      }
    }
  },
  "variables": {
    "webAppPortalName": "[concat(parameters('webAppName'), '-webapp')]",
    "appServicePlanName": "[concat('AppServicePlan-', parameters('webAppName'))]"
  },
  "resources": [
    {
      "type": "Microsoft.Web/serverfarms",
      "apiVersion": "2018-02-01",
      "name": "[variables('appServicePlanName')]",
      "location": "[parameters('location')]",
      "sku": {
        "name": "[parameters('sku')]"
      },
      "kind": "linux",
```

Entendiendo el ciclo



Beneficios de IaC

- IaC permite la **automatización**, lo que ahorra tiempo si a menudo se requiere crear nuevos entornos.
- IaC permite un enfoque declarativo, que le permite centrarse en el **estado deseado** y no en cómo llegar allí.
- IaC proporciona un formato **legible por humanos**, que permite a los desarrolladores razonar sobre el estado de la infraestructura.

Y esto ¿cómo lo implementamos en Azure?

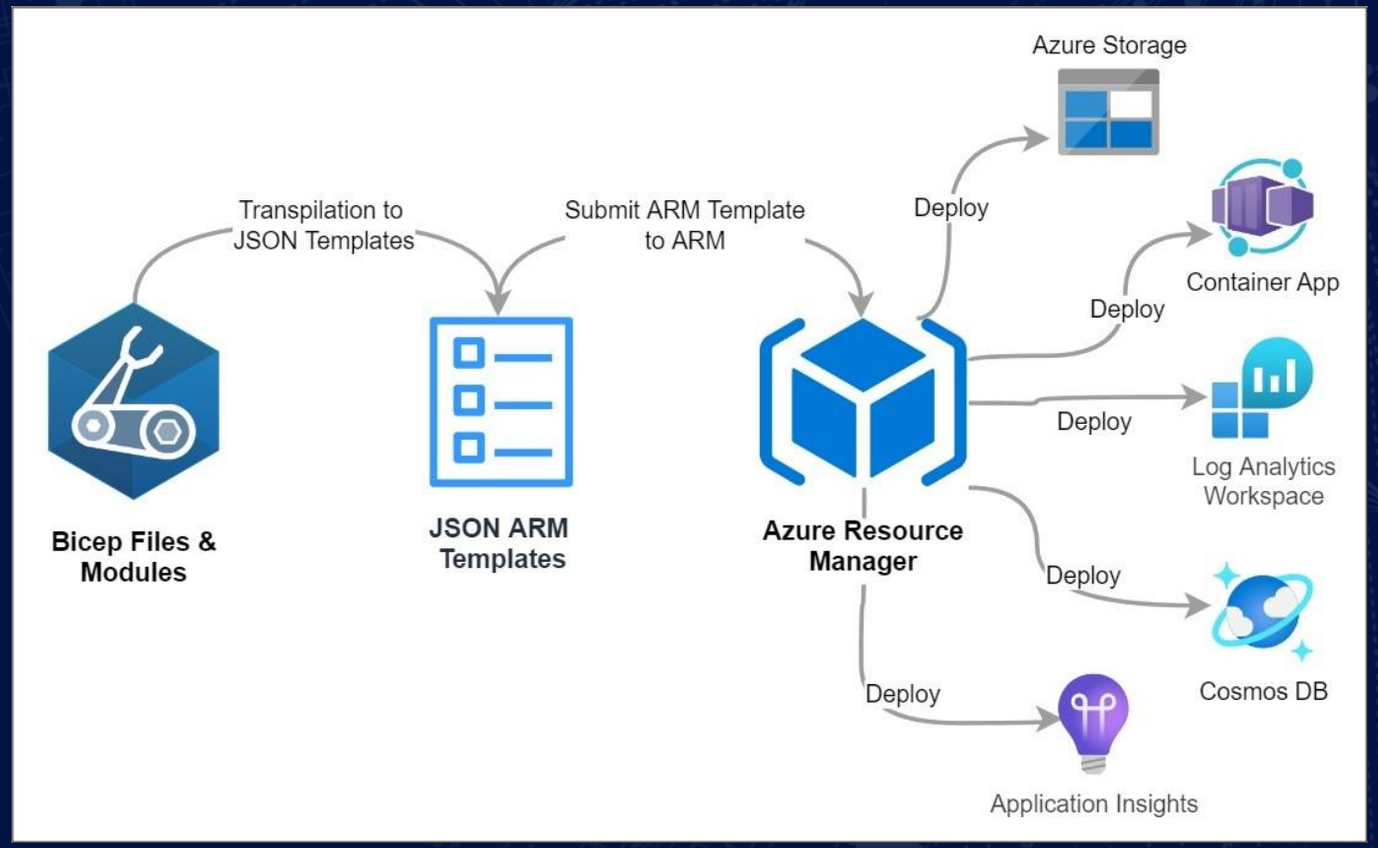
#GlobalAzure

Presentando a Bicep

- Bicep es un lenguaje específico del dominio (DSL) que usa **sintaxis declarativa** para implementar recursos de Azure. En un archivo Bicep, se define la infraestructura que se desea implementar en Azure y, a continuación, se usa dicho archivo durante todo el ciclo de vida de desarrollo para implementar repetidamente la infraestructura. Los recursos se implementan **de forma coherente**.
- Bicep brinda **sintaxis concisa**, seguridad de tipos confiable y compatibilidad con la **reutilización de código.**

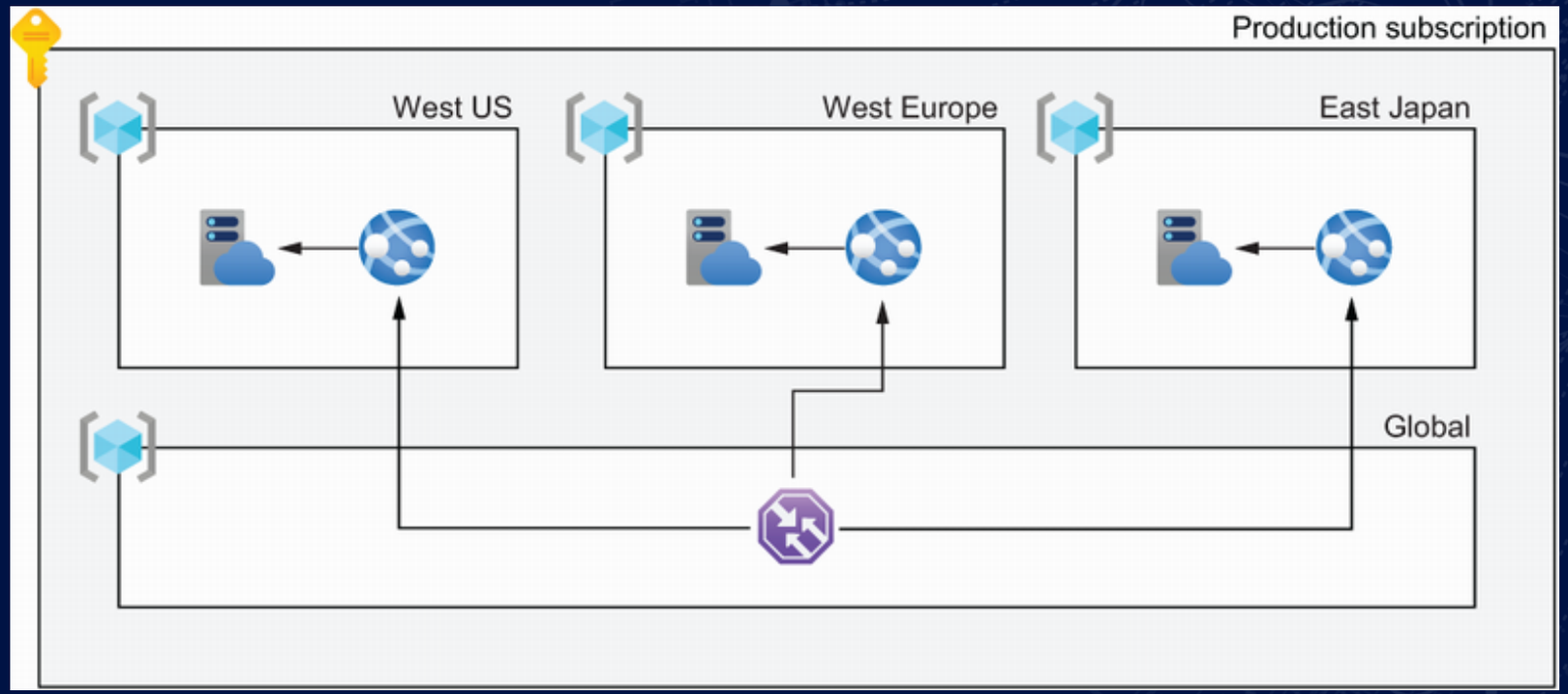
Entendiendo el modelo

- Es una abstracción sobre los templates ARM (Json)
- Mas ordenado y legible



Planteemonos un objetivo

- Un despliegue de websites con soporte multiregión y Traffic Manager
- Con entorno de prueba y de Producción!



Retos

- Reuso del template
- La información de algunos recursos solo se conoce luego de su creación….

¡Veamos algo de Código!

Buenas practicas

- Haga lo mismo en cada entorno, incluido Producción.
- Parametriza tus archivos. Dev, QA y Prod son cambios de parámetros, no nuevos scripts.
-

#GABMUGPeru

¿Preguntas?

#GABMUGPeru

www.consultorinternet.com

Patrocinadores

Microsoft

Colabora

25 años mibanco

baufest

#GLOBALAZURE


• BY COMMUNITY • FOR COMMUNITY •
#GLOBALAZURE
2023