



Gestisci il deployment su Azure... ...con Minecraft?

Nicola Paro

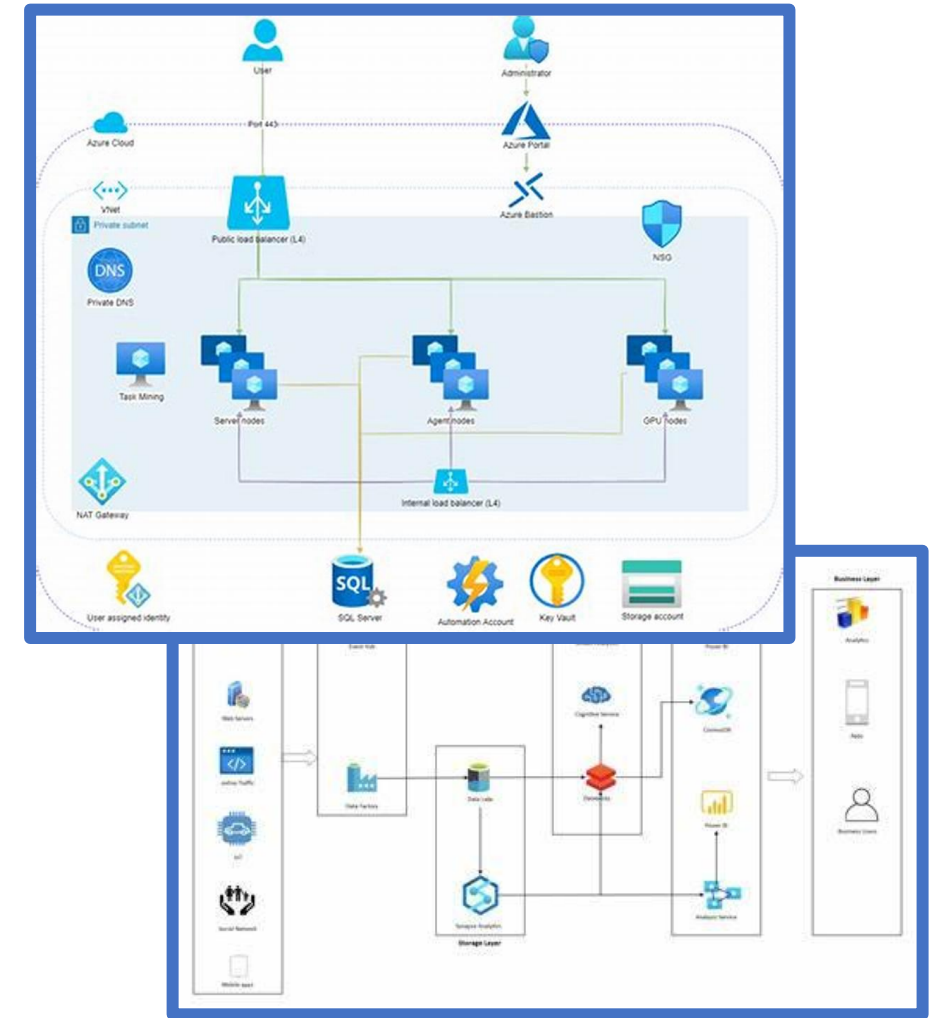


SPONSORS



A real life story...

- Virtual Buildings è un'azienda specializzata nella realizzazione di applicazioni cloud based su Azure.
- Sviluppa un software custom per Acme S.p.A.,
- Applicativo cloud based e, per tenere bassi i costi, Virtual Buildings lo ha realizzato utilizzando i tier minimi di tutti i servizi Azure.
- L'applicazione inizialmente si comporta in modo regolare ma dopo alcune settimane comincia ad essere meno performante.
- Con l'avanzare del tempo, anche l'applicazione diventa sempre più lenta e le operazioni iniziano a fallire per via dei numerosi timeout e throttlings
- Nessuno ha pensato allo scaling e a come gestirlo automaticamente



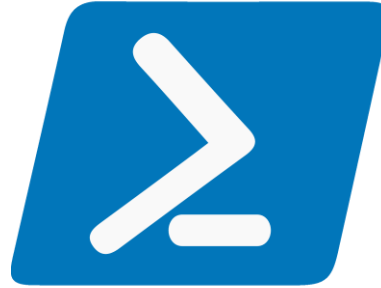
Come possiamo fare?

Soluzioni
tramite UI

Soluzioni da
riga di
comando

Infrastructure
as code
dichiarativo

AZ PowerShell Module



Set of cmdlets for managing Azure resources.

Built on .NET and works in PowerShell environments.

Enables scripting and automation of tasks.

AZ PowerShell Module

Install from PowerShell Gallery

```
Install-Module -Name Az -AllowClobber -Scope CurrentUser
```

Regular updates via

```
Update-Module -Name Az
```

AZ PowerShell Module – Getting Started

Connect-AzAccount

```
Connect-AzAccount `
  -ServicePrincipal `
  -Tenant <tenantId> `
  -ApplicationId <appId> `
  -Credential (Get-Credential)
```

AZ PowerShell Module - Esempi

Resource Management

`Get-AzResource, New-AzResourceGroup`

Virtual Machines

`New-AzVM, Start-AzVM, Stop-AzVM`

Storage

`Get-AzStorageAccount, New-AzStorageContainer`

Networking

`Get-AzVirtualNetwork, New-AzPublicIpAddress`

AZ CLI



- Azure CLI (Command-Line Interface) is a set of commands used to create and manage Azure resources.
- Cross-platform: works on Windows, macOS, and Linux.
- Available as open source here → <https://github.com/Azure/azure-cli>

Scriptable and
automatable

Faster
resource
management

Integrated
with Azure
Cloud Shell

Works with
CI/CD tools

AZ CLI – Getting Started

```
az login
```

```
az login -u <username> -p <password>
```

```
az login --service-principal -u <appId> -p <password> --tenant <tenant>
```

AZ CLI - Esempi

Resource Group

```
az group create --name MyGroup --location eastus
```

VM Creation

```
az vm create --resource-group MyGroup --name MyVM --image UbuntuLTS
```

Storage Account

```
az storage account create --name mystorage --resource-group MyGroup --  
location eastus --sku Standard_LRS
```

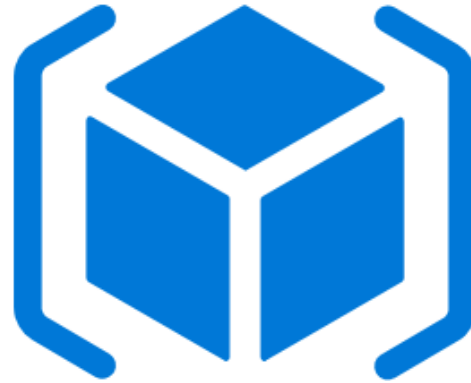
Combine commands in bash, PowerShell, or batch scripts.

```
az vm list --query "[].{Name:name, ResourceGroup:resourceGroup}" --output  
table
```

AZ CLI vs Azure PowerShell Module

Criteria	Azure CLI	Azure PowerShell Module
Platform & Shell	Designed for Bash, great for Linux/macOS	Native to PowerShell (Windows-first)
Syntax Simplicity	JSON-style, more concise and readable	Verb-Noun format (e.g., Get-AzVM)
Learning Curve	Easier for non-Windows admins or developers	Steeper if unfamiliar with PowerShell
Scripting Language	Shell scripts, Bash, etc.	PowerShell scripts (.ps1)
Cloud Shell Default	Default in Azure Cloud Shell	Also available, but not default
Output Formats	Built-in JSON, Table, TSV, YAML	JSON, but more verbose by default
Cross-Platform Usage	Uniform experience across Operative Systems	Slight variation in behavior
Community Examples	More online tutorials & templates in CLI	PowerShell examples often Windows-specific

ARM Templates



- È un file in formato JSON utilizzato per definire e distribuire risorse in Microsoft Azure.
- Fa parte del modello Infrastructure as Code (IaC), che consente la gestione dell'infrastruttura tramite codice.
- È dichiarativo: si descrive lo stato desiderato delle risorse, non i passaggi per arrivarci.
- Gestito tramite Azure Resource Manager (ARM), il motore di provisioning delle risorse in Azure.

ARM Templates

Consistenza

Le risorse vengono distribuite nello stesso modo ogni volta.

Ripetibilità

Facilmente riutilizzabili in ambienti diversi (dev, test, prod).

Automazione

Integrabili in pipeline CI/CD per il provisioning continuo.

Controllo

Specifiche chiare e valide dei componenti da distribuire.

Sicurezza

Puoi collegarli ad Azure Key Vault per gestire segreti e chiavi.

ARM Templates

- `$schema` – URL allo schema JSON di riferimento.
- `contentVersion` – Versione del template (formato: "1.0.0.0").
- `parameters` – Valori di input dinamici forniti al template.
- `variables` – Valori riutilizzabili interni al template.
- `resources` – Risorse da creare o aggiornare (VM, storage, rete, ecc.).
- `outputs` – Informazioni restituite al termine del deployment.

```
{  
  "$schema":  
    "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "parameters": {},  
  "variables": {},  
  "resources": [],  
  "outputs": {}  
}
```


ARM Templates

Parametri



```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "location": {
      "type": "string",
      "defaultValue": "westeurope"
    },
    "storageAccountName": {
      "type": "string"
    },
    "webAppName": {
      "type": "string"
    },
    "sqlServerName": {
      "type": "string"
    },
    "sqlAdminUser": {
      "type": "string"
    },
    "sqlAdminPassword": {
      "type": "securestring"
    },
    "sqlDbName": {
      "type": "string",
      "defaultValue": "mydb"
    },
    "appServicePlanName": {
      "type": "string",
      "defaultValue": "myAppServicePlan"
    }
  },
  "variables": {
    "sqlConnectionString": "[concat('Server=tcp:', parameters('sqlServerName'),
      '.database.windows.net,1433;Initial Catalog=', parameters('sqlDbName'), ';Persist
      Security Info=False;User ID=', parameters('sqlAdminUser'), ';Password=',
      parameters('sqlAdminPassword'), ';MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection
      Timeout=30;')]"
  }
}
```

Risorse



ARM Templates – Funzioni

Funzione	Descrizione	Esempio
base64(string)	Codifica in base64	"[base64('abc')]"
uri(base, path)	Combina base URL con path	"[uri('https://site.com', 'api/values')]"
coalesce(a, b, ...)	Primo valore non null	"[coalesce(null, '', 'val')]" → ""
empty(value)	Controlla se stringa/array è vuoto	"[empty('')]" → true
if(cond, trueVal, falseVal)	Valutazione condizionale	"[if(equals(parameters('env'), 'prod'), 'Standard', 'Basic')]"
equals(a, b)	Confronta due valori	"[equals('a', 'b')]"
not(bool)	Negazione logica	"[not(true)]"
and(bool1, bool2)	AND logico	"[and(true, false)]"
or(bool1, bool2)	OR logico	"[or(false, true)]"
concat(a, b, ...)	Concatena stringhe	"[concat('https://', parameters('name'))]"
format(formatStr, ...)	Formatta stringhe con placeholder {0}	"[format('{0}/{1}', 'abc', 'def')]"
toLowerCase(str)	Minuscolo	"[toLowerCase('Hello')]" → hello
toUpperCase(str)	Maiuscolo	"[toUpperCase('abc')]" → ABC
replace(str, old, new)	Sostituisce porzioni di stringa	"[replace('abc123', '123', 'XYZ')]"
substring(str, start, len)	Estrae parte della stringa	"[substring('abcdef', 1, 3)]" → bcd
guid(val1, val2, ...)	Genera GUID deterministico da input	"[guid('site1', 'slot1')]"
contains(array, value)	Controlla se array/oggetto contiene elemento	"[contains(parameters('features'), 'logging')]"
length(array string)	Lunghezza array o stringa	
first(array)	Primo elemento dell'array	"[first(parameters('items'))]"
last(array)	Ultimo elemento	"[last(parameters('items'))]"
union(array1, array2)	Unione di due array	"[union(array1, array2)]"
intersection(array1, array2)	Elementi comuni	"[intersection(array1, array2)]"
resourceId(type, name, ...)	ID risorsa Azure	"[resourceId('Microsoft.Web/sites', 'mySite')]"
subscription()	Info sulla sottoscrizione	"[subscription().subscriptionId]"
tenant()	Info sul tenant	"[tenant().tenantId]"
deployment()	Info sul deployment corrente	"[deployment().name]"
reference(resourceId, apiVersion)	Ottiene proprietà di una risorsa già deployata	"[reference(resourceId(...), '2021-01-01')]"
listKeys(resourceId, apiVersion)	Recupera chiavi di accesso	"[listKeys(resourceId(...), '2022-09-01').keys[0].value]"

<https://learn.microsoft.com/en-us/azure/azure-resource-manager/templates/template-functions>

ARM Templates – How to Deploy?

- Azure Portal

- Azure CLI

```
az deployment group create --resource-group mioGruppo --template-file  
template.json --parameters @parametri.json
```

- PowerShell

```
New-AzResourceGroupDeployment -ResourceGroupName "mioGruppo" -TemplateFile  
"template.json" -TemplateParameterFile "parametri.json"
```

- Azure DevOps / GitHub Actions

Integrazione in pipeline CI/CD per deployment automatici.

ARM Templates – Best Practices

- Utilizzare parametri per valori dinamici (es. nomi, posizioni, SKU).
- Definire variabili per evitare ripetizioni.
- Validare i template con **az deployment validate**
- Separare le logiche in template modulari (linked templates).
- Usare Azure Key Vault per gestire segreti e credenziali.

Bicep

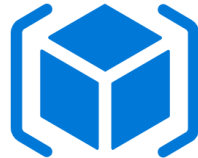


Bicep è un linguaggio dichiarativo di Infrastructure as Code (IaC) per Azure, pensato come alternativa più leggibile e moderna agli ARM template JSON

Bicep

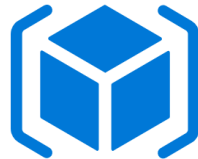
```
{
  ...
  "parameters": {
    "location": {
      "type": "string",
      "defaultValue": "westeurope"
    },
    "storageAccountName": { "type": "string" },
    "webAppName": { "type": "string" },
    "sqlServerName": { "type": "string" },
    "sqlAdminUser": { "type": "string" },
    "sqlAdminPassword": { "type": "securestring" },
    "sqlDbName": { "type": "string", "defaultValue": "mydb" },
    "appServicePlanName": {
      "type": "string",
      "defaultValue": "myAppServicePlan"
    }
  },
  ...
}
```

```
param location string = 'westeurope'
param storageAccountName string
param webAppName string
param sqlServerName string
param sqlAdminUser string
@secure()
param sqlAdminPassword string
param sqlDbName string = 'mydb'
param appServicePlanName string = 'myAppServicePlan'
```



Bicep

```
{
  ...
  "variables": {
    "storageConnectionString":
      "[concat('DefaultEndpointsProtocol=https;AccountName=',
parameters('storageAccountName'), ';AccountKey=',
listKeys(resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccountName')), '2022-09-01').keys[0].value,
';EndpointSuffix=core.windows.net')]",
    "sqlConnectionString": "[concat('Server=tcp:',
parameters('sqlServerName'),
'.database.windows.net,1433;Initial Catalog=',
parameters('sqlDbName'), ';Persist Security Info=False;User
ID=', parameters('sqlAdminUser'), ';Password=',
parameters('sqlAdminPassword'),
';MultipleActiveResultSets=False;Encrypt=True;TrustServerCertif
icate=False;Connection Timeout=30;')]"
  },
  ...
}
```



```
var storageKeys = listKeys(storage.id, storage.apiVersion)

var storageConnectionString =
'DefaultEndpointsProtocol=https;AccountName=${storageAccountNam
e};AccountKey=${storageKeys.keys[0].value};EndpointSuffix=core.
windows.net'

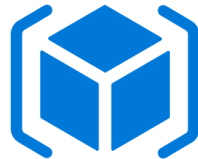
var sqlConnectionString =
'Server=tcp:${sqlServerName}.database.windows.net,1433;Initial
Catalog=${sqlDbName};Persist Security Info=False;User
ID=${sqlAdminUser};Password=${sqlAdminPassword};MultipleActiveR
esultSets=False;Encrypt=True;TrustServerCertificate=False;Conne
ction Timeout=30;'
```



Bicep

```
{  
  ...  
  "resources": [  
    {  
      "type": "Microsoft.Storage/storageAccounts",  
      "apiVersion": "2022-09-01",  
      "name": "[parameters('storageAccountName')]",  
      "location": "[parameters('location')]",  
      "sku": { "name": "Standard_LRS" },  
      "kind": "StorageV2",  
      "properties": {}  
    }  
  ]  
  ...  
}
```

```
resource storage 'Microsoft.Storage/storageAccounts@2022-09-01' = {  
  name: storageAccountName  
  location: location  
  sku: {  
    name: 'Standard_LRS'  
  }  
  kind: 'StorageV2'  
  properties: {}  
}
```



Bicep – How to Deploy?















Esattamente come se fosse un ARM template, solo che invece che essere un file «.json» è un file «.bicep»:

- Azure Portal
- Azure CLI

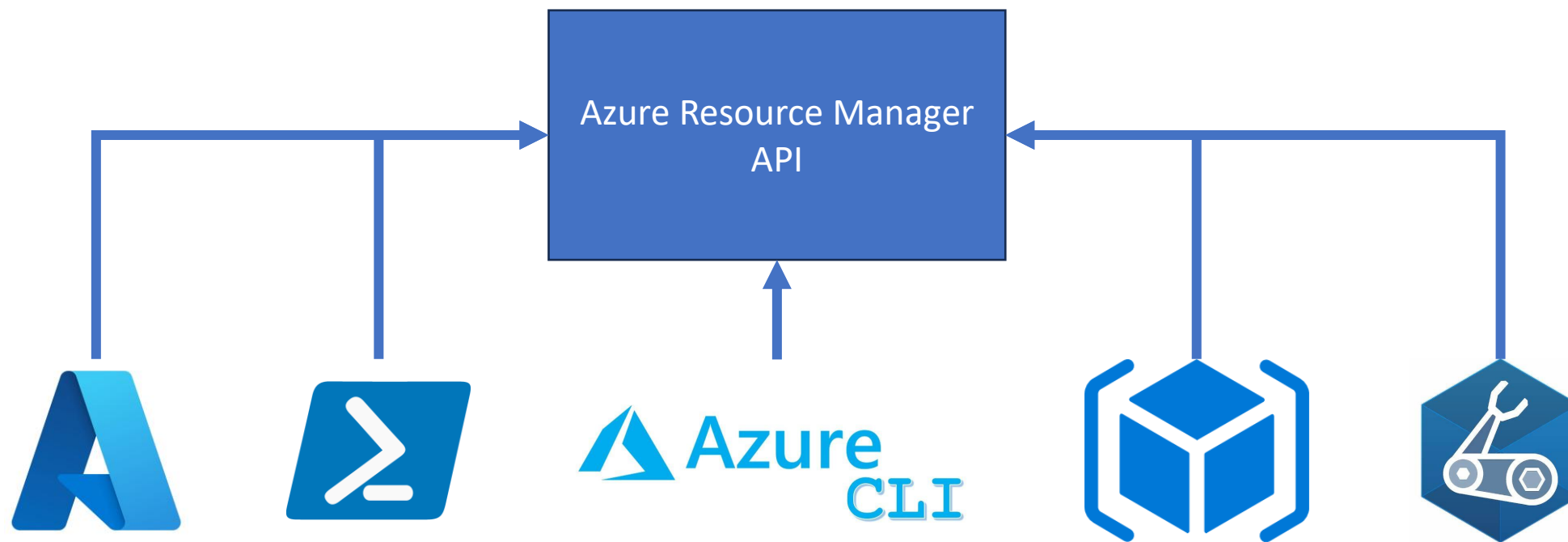
```
az deployment group create --resource-group mioGruppo --template-file  
template.bicep --parameters @parametri.json
```
- PowerShell

```
New-AzResourceGroupDeployment -ResourceGroupName "mioGruppo" -TemplateFile  
"template.bicep" -TemplateParameterFile "parametri.json"
```
- Azure DevOps / GitHub Actions
Integrazione in pipeline CI/CD per deployment automatici.

A confronto

					
Interfaccia	Grafica	 Riga di comando (PowerShell)	 Riga di comando (bash/cmd)	{ } JSON dichiarativo	DSL dichiarativa (semplificata per ARM)
Facilità d'uso	Alta per principianti	Media	Media	Bassa (complesso e verboso)	Media/Alta (più leggibile di ARM)
Automazione	Limitata	Elevata	Elevata	Elevata	Elevata
Controllo versione	✗	✓	✓	✓	✓
Modularità e riuso	✗	✗ Limitata	✗ Limitata	✓ Supportata ma complicata	✓
Validazione prima del deploy	✗	✗ Parziale	✗ Parziale	✓	✓
Complessità gestibile	Bassa	Media	Media	Alta	Media
Dipendenze tra risorse	 Manuali	 Manuali	 Manuali	 Supportate	 Supportate (semplificate)
Deploy ripetibili/idempotenti	✗	 Parzialmente	 Parzialmente	✓	✓
Supporto CI/CD	✗ Limitato	✓	✓	✓	✓
Conversione in ARM	✗	✗	✗	✓ Nativo	✓ (compila in ARM JSON)

API



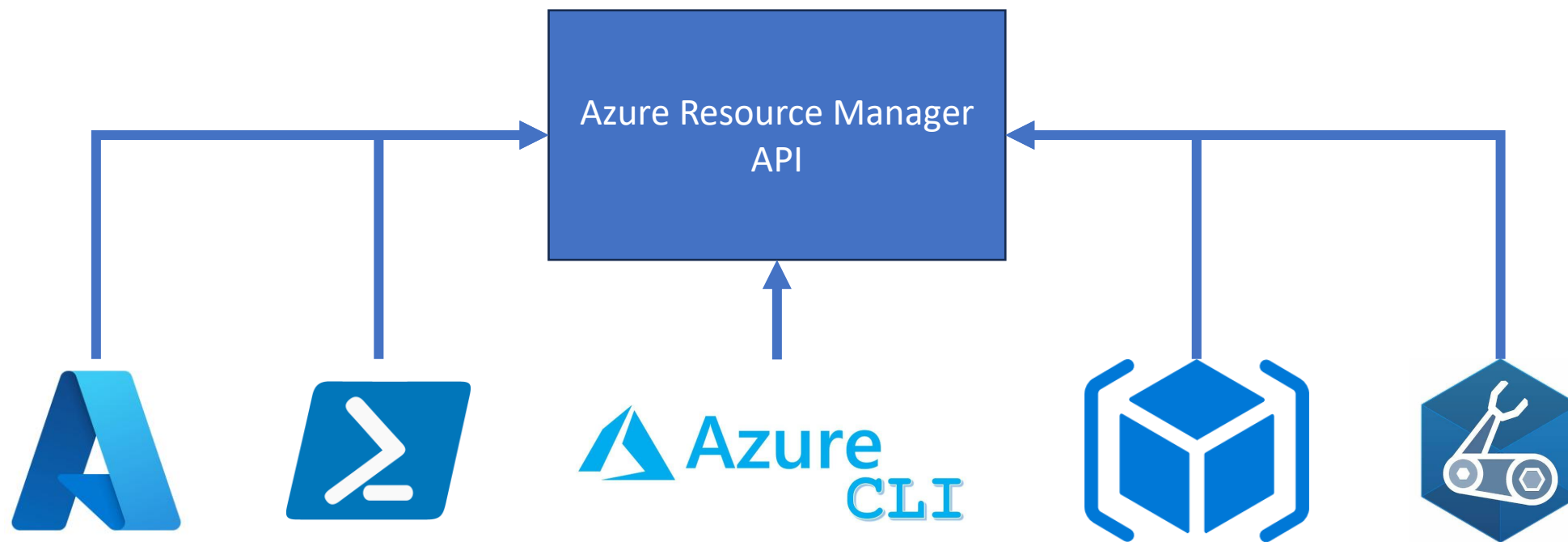
ARM API

- Le API di ARM sono basate su REST e permettono di interagire programmaticamente con tutte le risorse di Azure.
- L'endpoint di base è: <https://management.azure.com/>.
- Principali operazioni supportate:
 - GET: Recupero di informazioni sulle risorse
 - POST: Creazione di nuove risorse
 - PUT: Aggiornamento di risorse esistenti
 - DELETE: Rimozione di risorse
- Le API sono versionate per mantenere la retrocompatibilità.

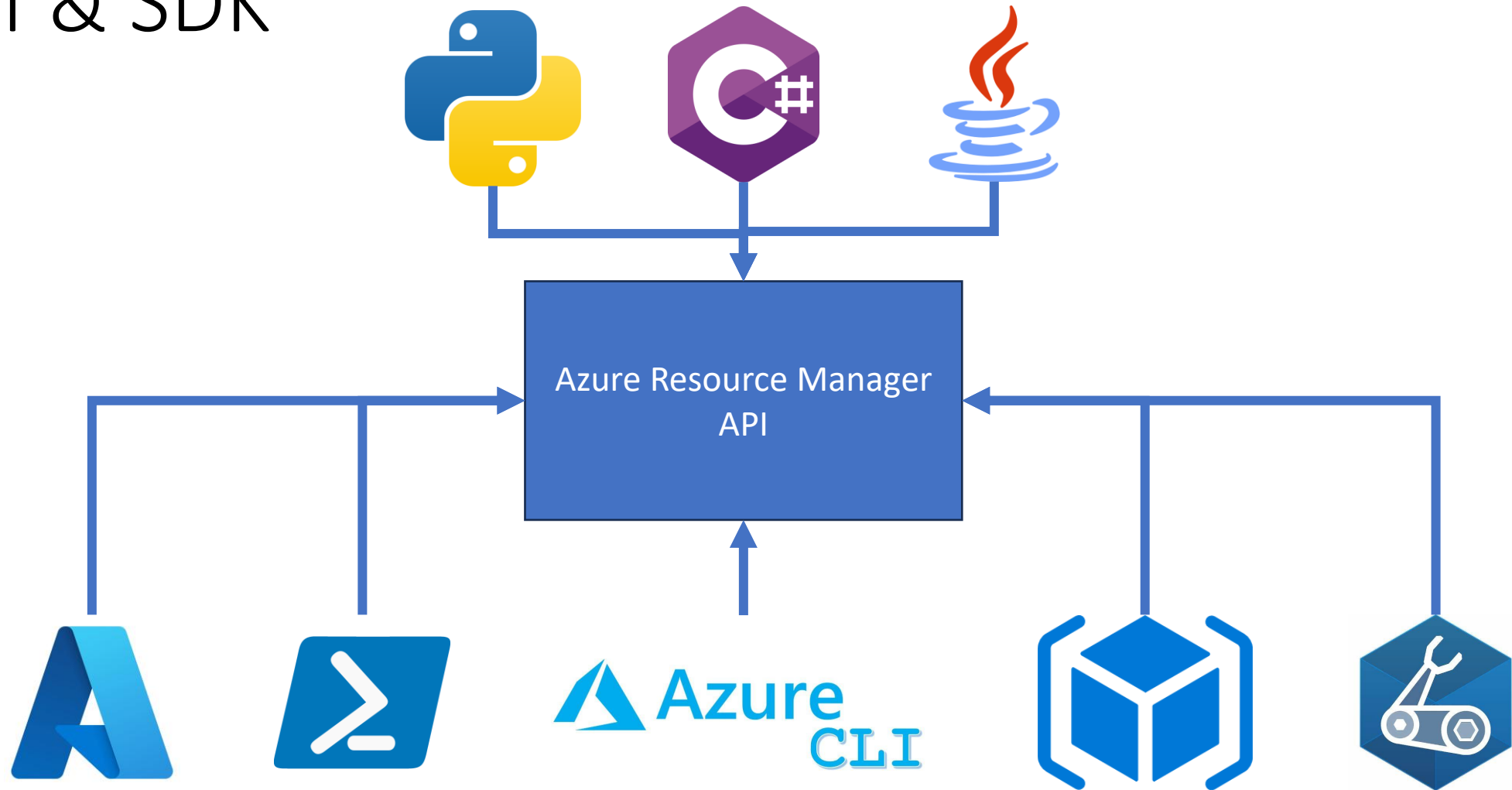
ARM API

```
curl -X PUT \  
  -H "Authorization: Bearer <token>" \  
  -H "Content-Type: application/json" \  
  --data @template.json \  
  "https://management.azure.com/subscriptions/<subscription-  
id>/resourceGroups/<resource-  
group>/providers/Microsoft.Compute/virtualMachines/<vm-name>?api-  
version=2022-03-01"
```

API



API & SDK











SDK C#

Libreria client ufficiale di Microsoft per gestire le risorse di Azure in modo programmatico tramite codice C#.

Azure.Identity utilizzata per la parte di autenticazione.

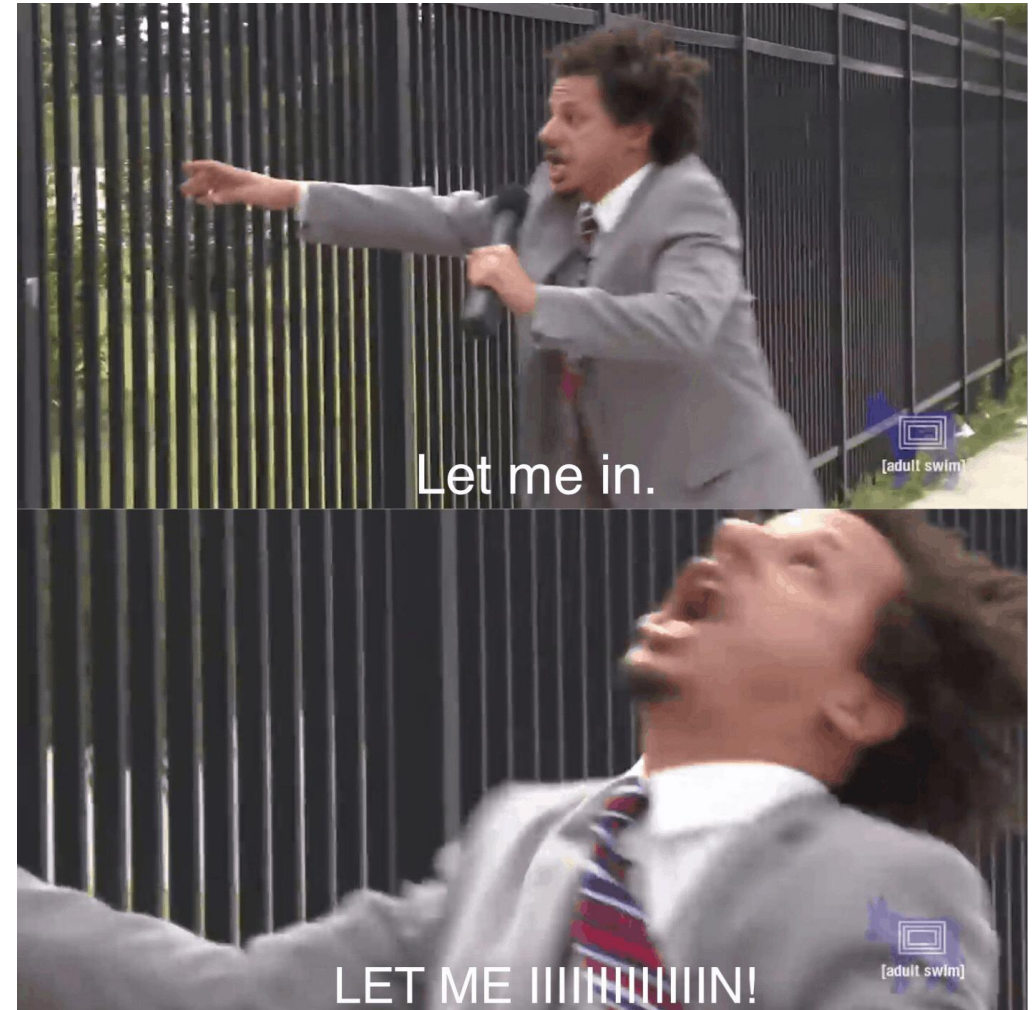
Azure.ResourceManager contiene le funzionalità di base e la gestione dei resource groups.

Per ogni risorsa Azure è necessario il pacchetto nuget corrispondente

	Azure.Identity ✓ by azure-sdk, Microsoft, 921M downloads This is the implementation of the Azure SDK Client Library for Azure Identity	1.13.2
	Azure.ResourceManager.Compute ✓ by azure-sdk, Microsoft, 2.85M downloads Microsoft Azure management client SDK for Azure resource provider Microsoft.Compute.	1.9.0
	Azure.ResourceManager.Monitor ✓ by azure-sdk, Microsoft, 949K downloads Microsoft Azure Resource Manager client SDK for Azure resource provider Microsoft.Insights.	1.3.1
	Azure.ResourceManager.ServiceBus ✓ by azure-sdk, Microsoft, 2.06M downloads Microsoft Azure management client SDK for Azure resource provider Microsoft.ServiceBus.	1.1.0
	Azure.ResourceManager.AppService ✓ by azure-sdk, Microsoft, 2.34M downloads Microsoft Azure management client SDK for Azure resource provider Microsoft.Web.	1.3.0
	Azure.ResourceManager ✓ by azure-sdk, Microsoft, 27.9M downloads Microsoft Azure Resource Manager client SDK for Azure resources.	1.13.0 1.13.1
	Azure.ResourceManager.Dns ✓ by azure-sdk, Microsoft, 799K downloads Microsoft Azure Resource Manager client SDK for Azure resource provider Microsoft.Network Dns.	1.1.1
	Azure.ResourceManager.Cdn ✓ by azure-sdk, Microsoft, 990K downloads Microsoft Azure Resource Manager client SDK for Azure resource provider Microsoft.Cdn.	1.3.1

SDK C# - Setup

```
var credential = new ____AzureCredential();  
var client = new ArmClient(credential);
```



SDK C# - Setup

	Descrizione	Pro	Contro
DefaultAzureCredential	Tenta diverse credenziali in ordine predefinito (locale, ambienti, gestite)	Facile da usare, ideale per ambienti dev → prod	Possibile ambiguità se più metodi sono configurati
EnvironmentCredential	Usa variabili di ambiente per client ID, secret e tenant	Sicura e automatizzabile in ambienti CI/CD	Richiede setup accurato delle variabili
ManagedIdentityCredential	Usa l'identità gestita assegnata alla risorsa Azure (VM, App Service, ecc.)	Nessuna gestione segreti, molto sicura	Funziona solo in ambienti Azure con identità gestita abilitata
InteractiveBrowserCredential	Apre un browser per login interattivo dell'utente	Utile per strumenti locali o CLI personalizzati	Non adatto ad ambienti automatizzati o headless
VisualStudioCredential	Usa il login configurato in Visual Studio	Ottimo per sviluppatori che usano Visual Studio	Limitato a chi ha l'IDE installato e configurato
AzureCliCredential	Usa il token della sessione az login	Perfetto per sviluppo locale, integrato con CLI	Richiede che l'utente abbia effettuato az login
AzurePowerShellCredential	Usa il contesto di login da Connect-AzAccount	Utile in ambienti PowerShell e script	Dipende dal contesto PowerShell configurato
ClientSecretCredential	Usa ID applicazione, secret e tenant per l'autenticazione	Adatto a produzione, automatizzabile	Richiede gestione sicura dei segreti
ClientCertificateCredential	Come sopra, ma usa un certificato al posto del secret	Più sicura del secret, usata spesso in scenari enterprise	Richiede gestione certificato (caricamento, validità, ecc.)
ChainedTokenCredential	Permette di combinare più credenziali in ordine di fallback	Flessibile, personalizzabile	Richiede configurazione esplicita della catena

SDK C# - Setup

DefaultAzureCredential

EnvironmentCredential

ManagedIdentityCredential

InteractiveBrowserCredential

VisualStudioCredential

AzureCliCredential

AzurePowerShellCredential

ClientSecretCredential

ClientCertificateCredential

ChainedTokenCredential

DefaultAzureCredential

1. EnvironmentCredential
2. ManagedIdentityCredential
3. VisualStudioCredential
4. AzureCliCredential
5. InteractiveBrowserCredential

Permette di combinare più credenziali in ordine di fallback

Flessibile, personalizzabile

Richiede configurazione esplicita della catena

Evita ambiguità se più metodi sono configurati

Permette setup accurato delle variabili d'ambiente

Funziona solo in ambienti Azure con Managed Identity gestita abilitata

Adatto ad ambienti automatizzati o CI/CD

Limitato a chi ha l'IDE installato e configurato

Assicura che l'utente abbia effettuato az

Accesso dal contesto PowerShell configurato

Permette gestione sicura dei segreti

Permette gestione certificato (certificato, validità, ecc.)

SDK C#

```
var subscription = await armClient.GetDefaultSubscriptionAsync();  
  
var rgData = new ResourceGroupData(AzureLocation.WestEurope);  
var rg = await subscription.GetResourceGroups().CreateOrUpdateAsync("myRg", rgData);  
  
var existingRg = await subscription.GetResourceGroups().GetAsync("myRg");
```

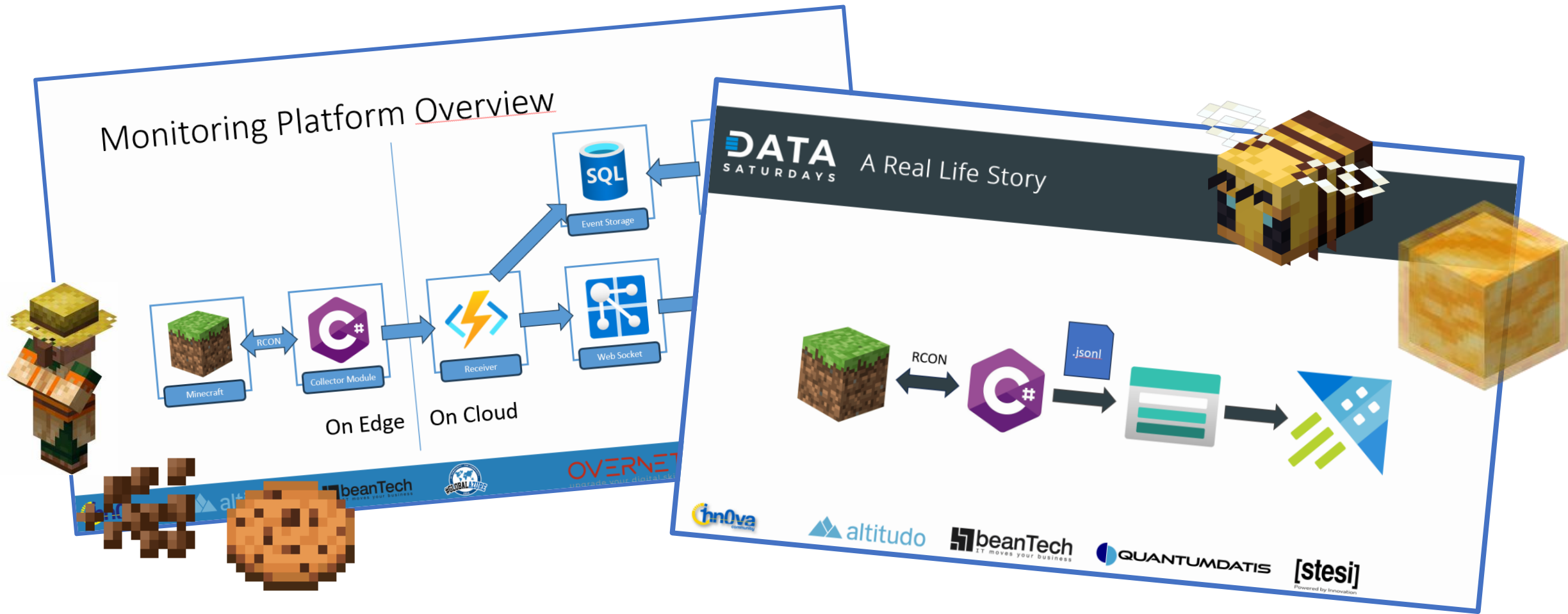
Posso usare quello che voglio per gestire il Deployment su Azure...



... Anche Minecraft?



Nelle puntate precedenti



Minecraft per gestire Azure



Minecraft per gestire Azure





DEMO



Grazie!

About me

Nicola Paro

Cloud Solution Architect
beanTech



linktr.ee/nicolaparo



Codice della demo → <https://github.com/nicolaparo/AzureCraft>