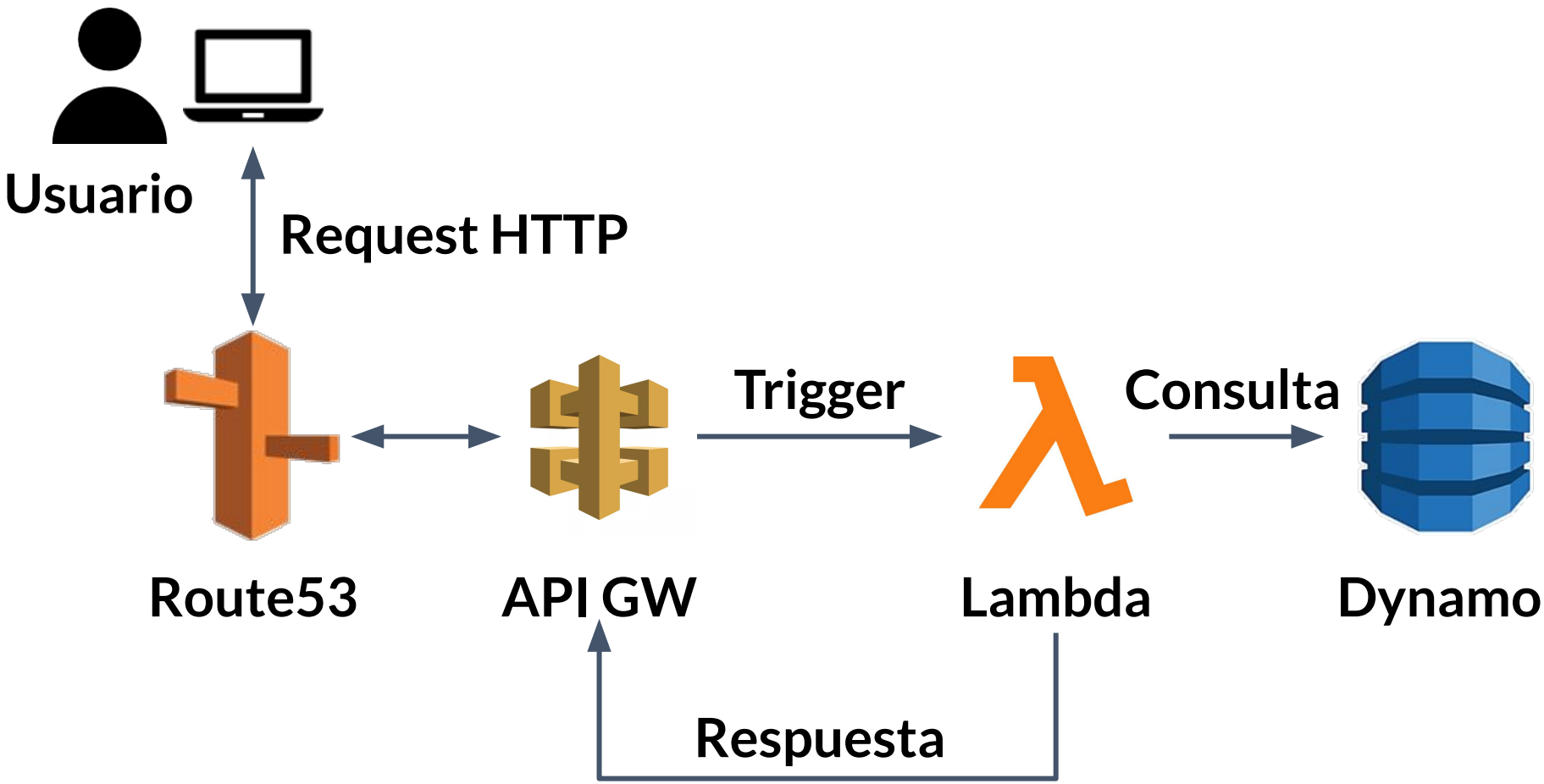
Curso de

Infraestructura como código en AWS

Carlos Zambrano

Proyecto del Curso

VotaNet App


Usuario
Request HTTP
Route53
API GW
Trigger
Lambda
Consulta
Dynamo
Respuesta

Proyecto

GitHub

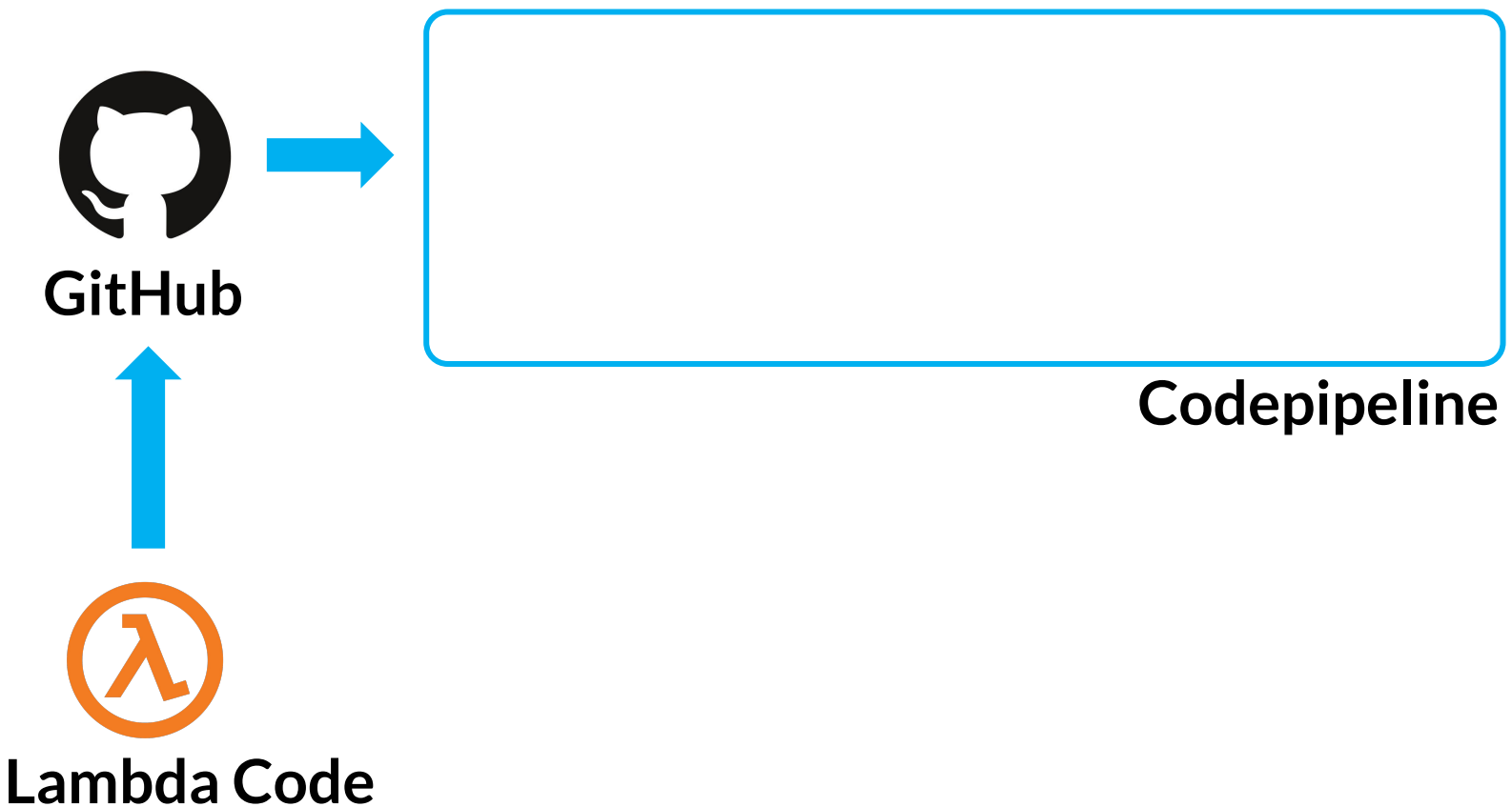
Proyecto

GitHub

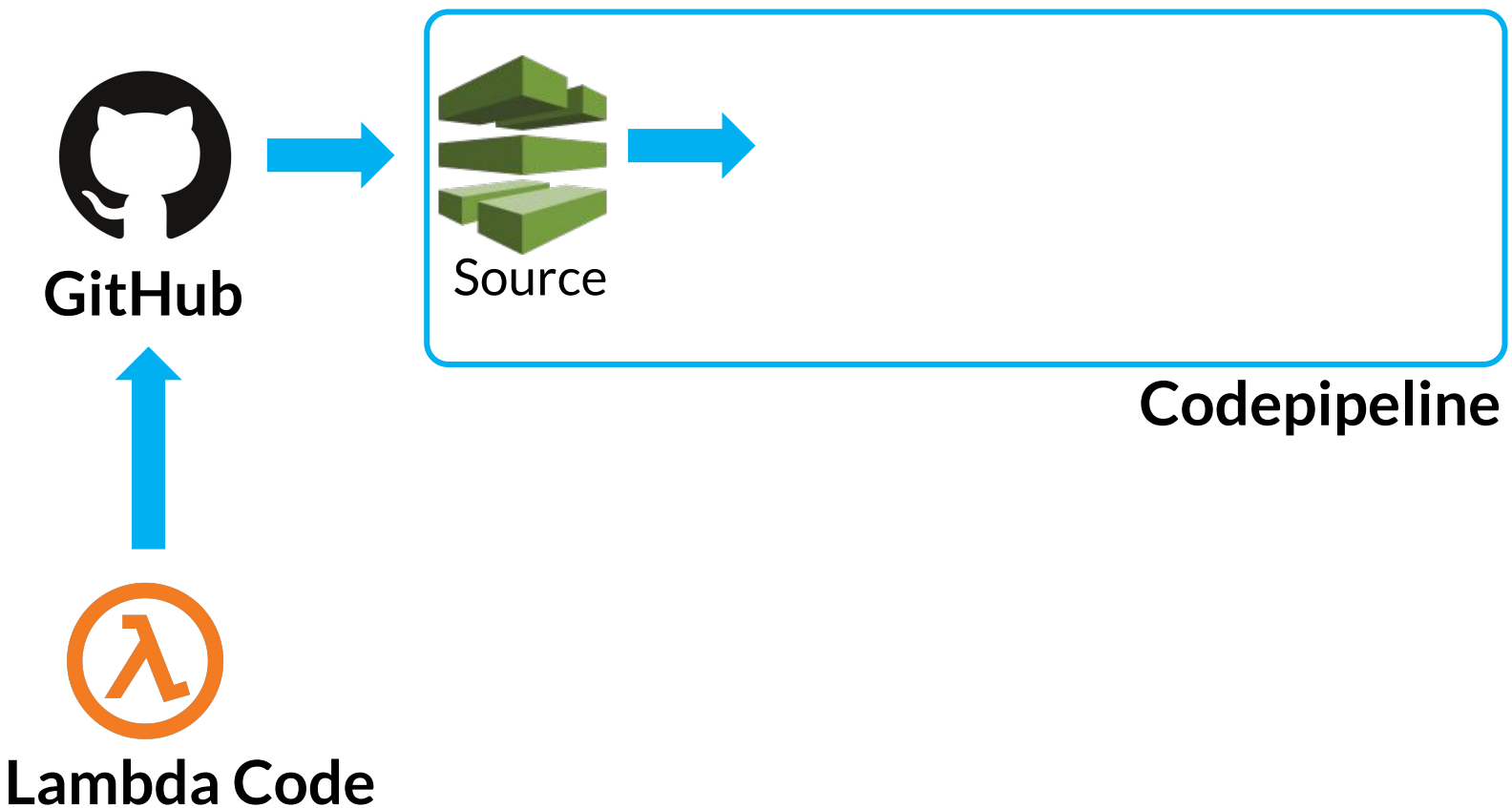
Lambda Code

```
def main_index_function(outevent, sns_id):
    """
    main function()
    uses the aws cloud event in gzip form, logger info is used to debug
    save the elasticsearch object using the bulk insert
    :param outevent: decoded aws cloudwatch log event
    :param loggerinfo:
    """
    clean_event = json.loads(outevent)
    # logs_message=clean_event['message']
```

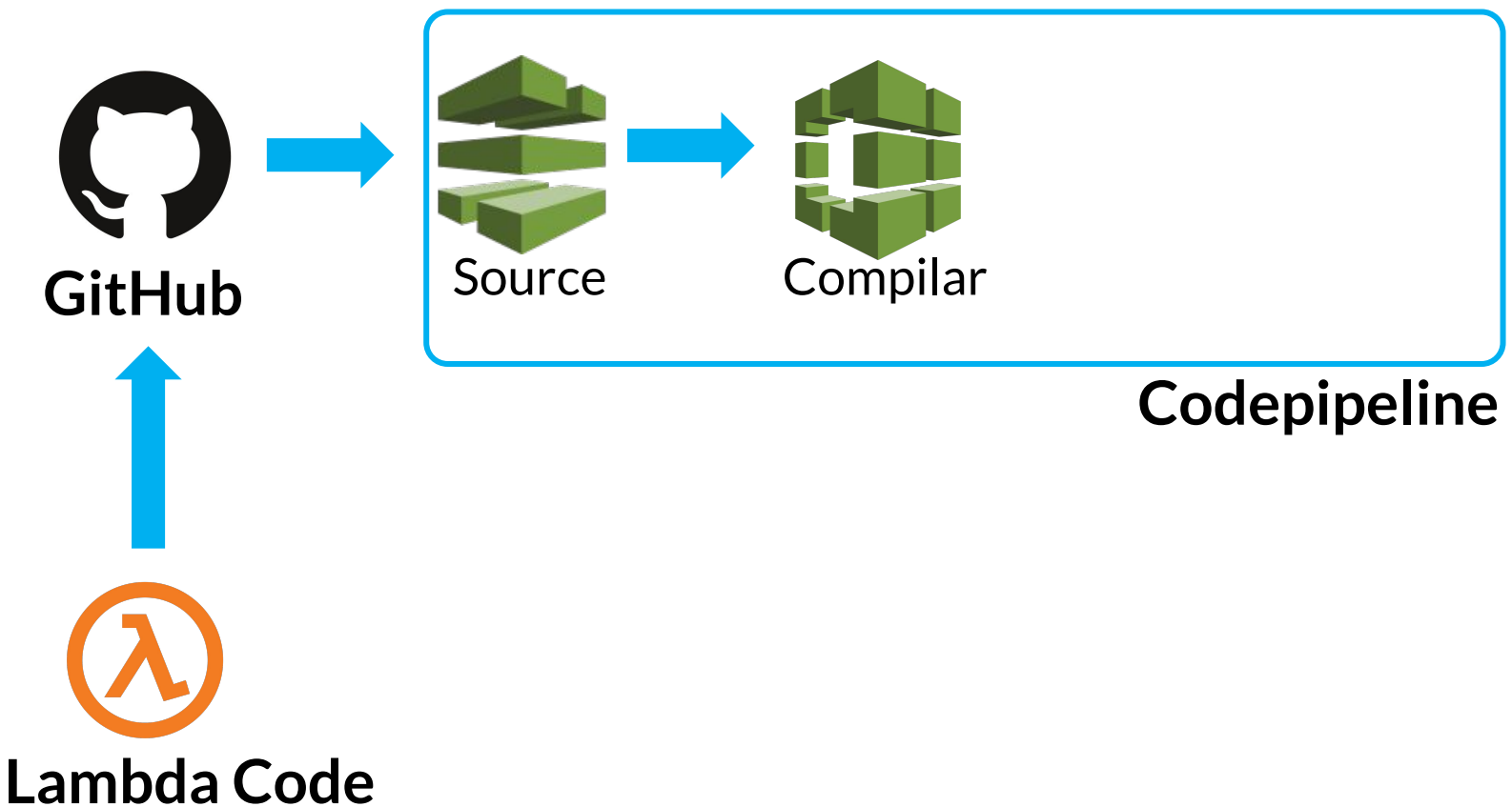
Proyecto



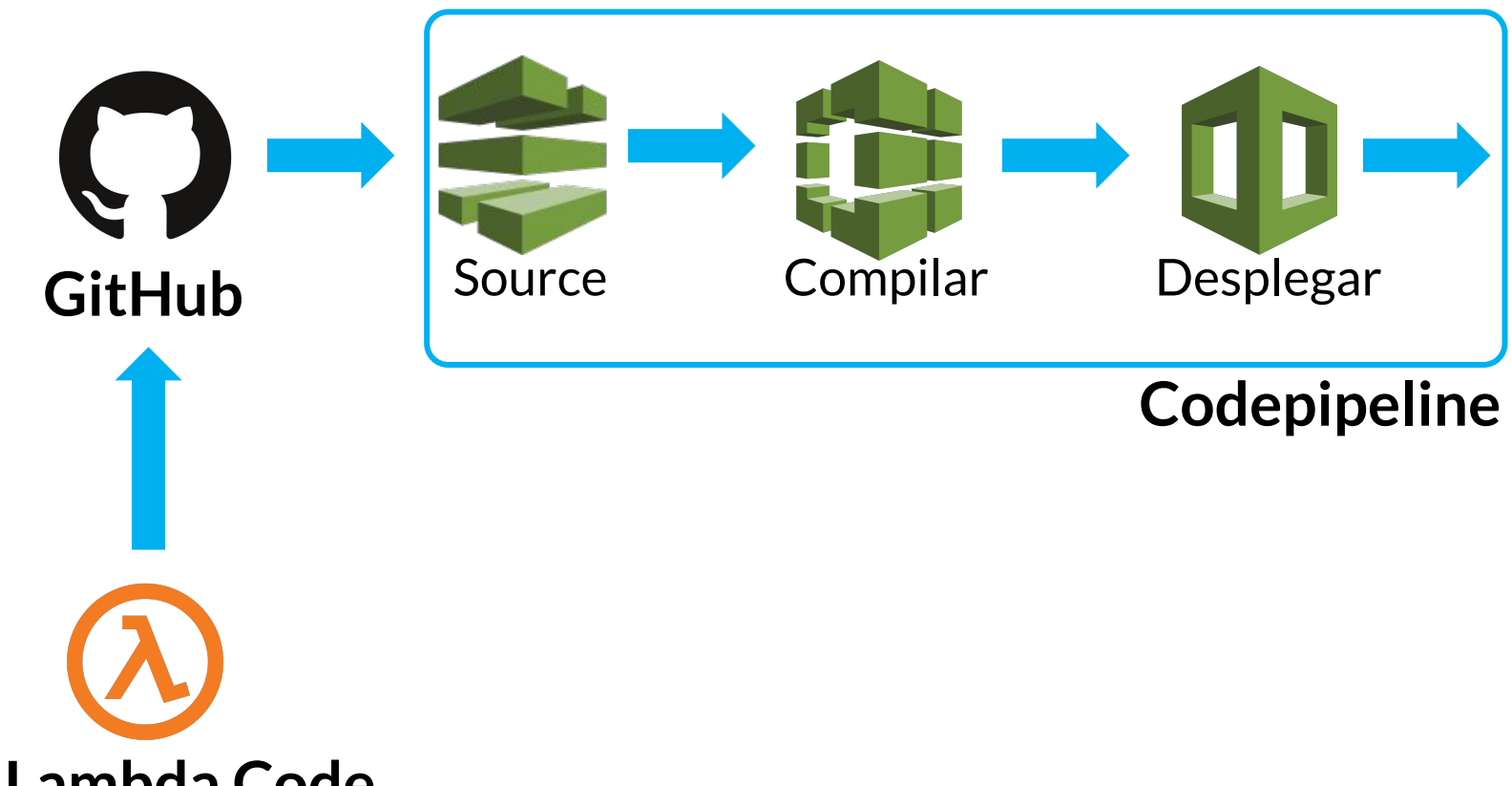
Proyecto



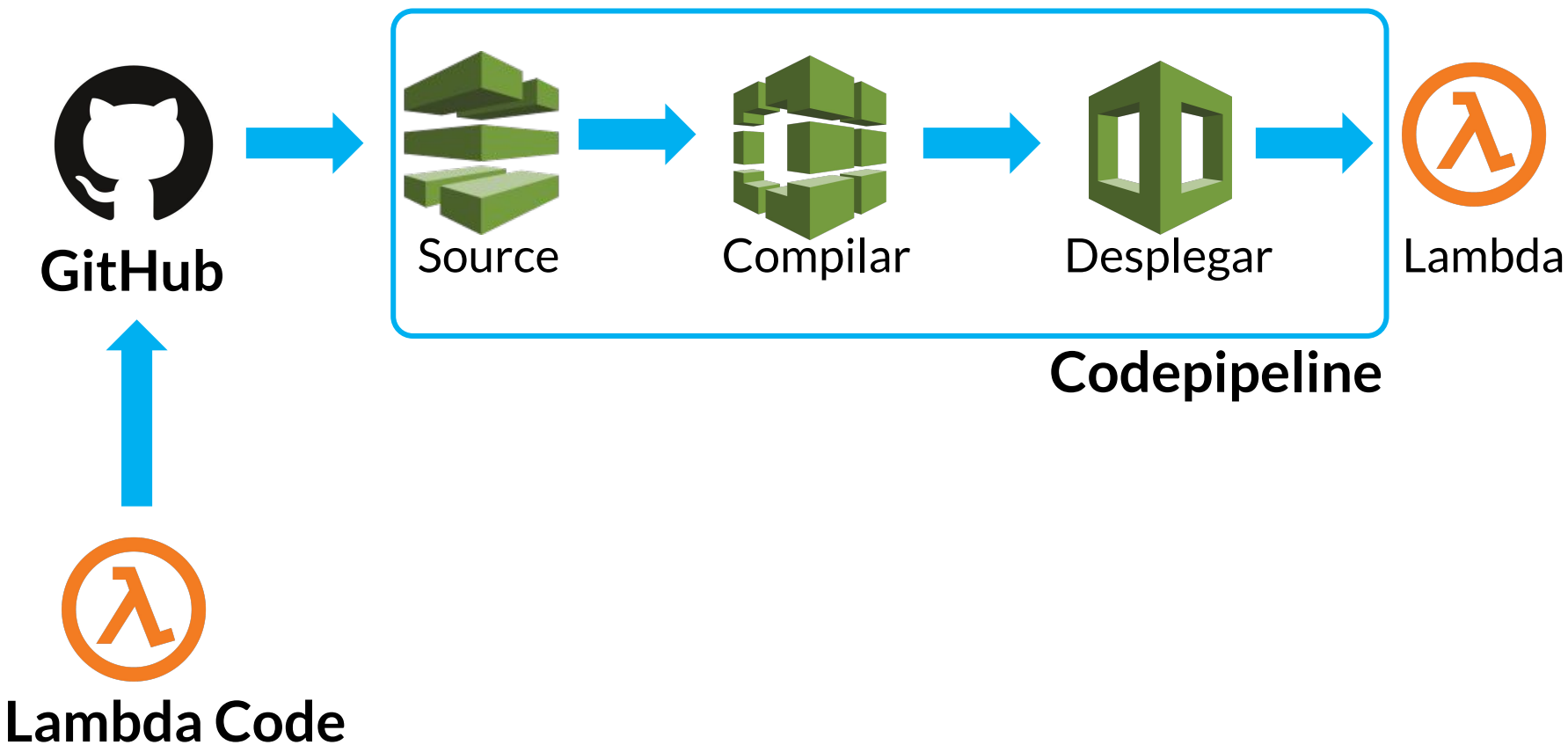
Proyecto



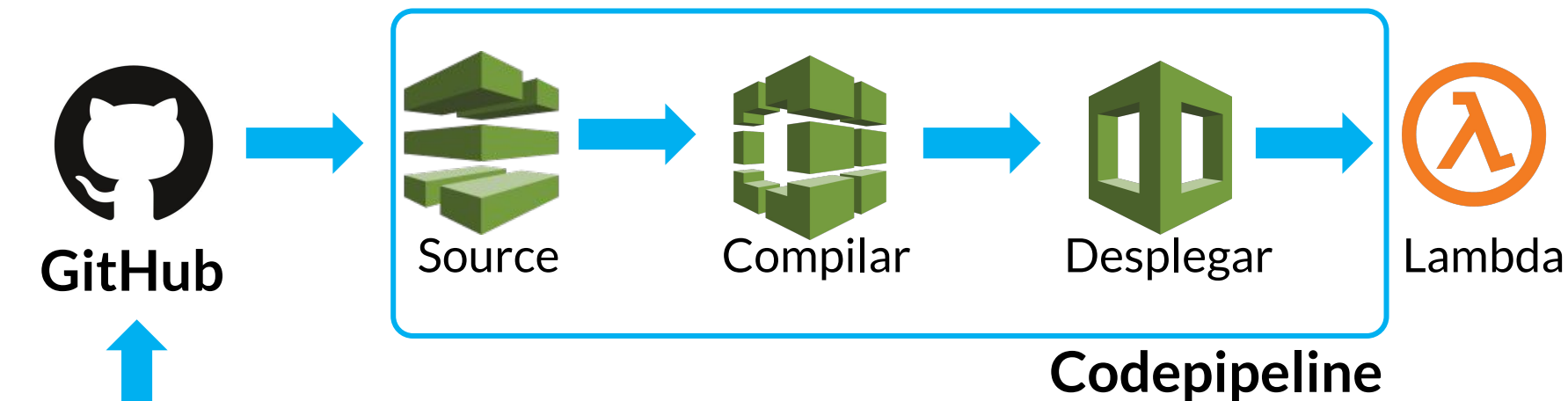
Proyecto


GitHub
Source
Compilar
Desplegar
Codepipeline
Lambda Code

Proyecto


GitHub
Source
Compilar
Desplegar
Lambda
Codepipeline
Lambda Code

Proyecto



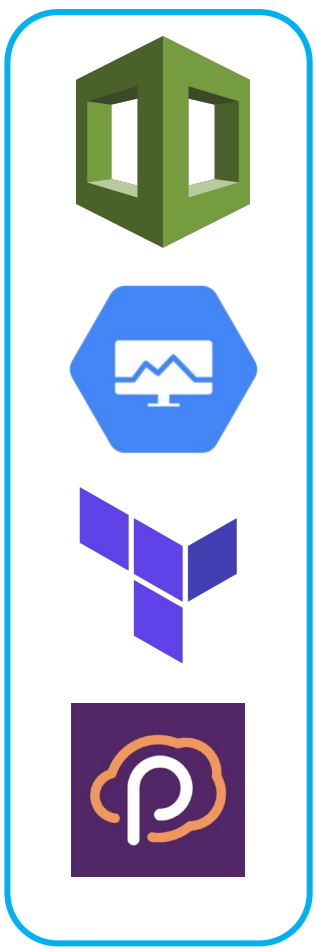
Lambda Code

CloudFormation

```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

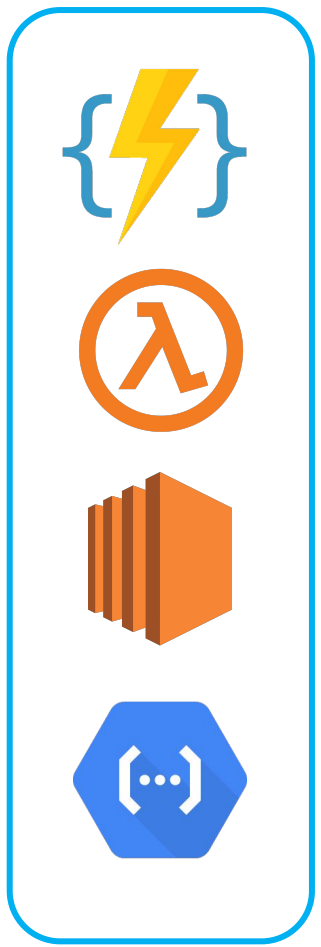
Desplegar infraestructura en Cloud

Herramientas



Herramientas

Desplegar Infraestructura

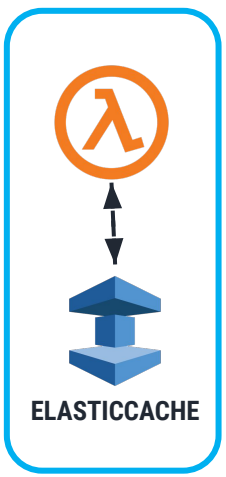


Servicios en
Cloud Providers

Versionamiento

```
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

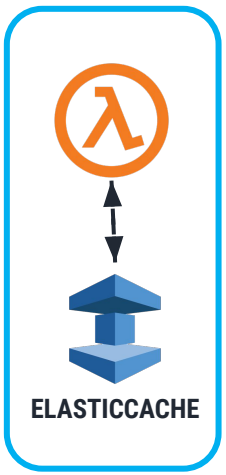
Versión 1



Versionamiento

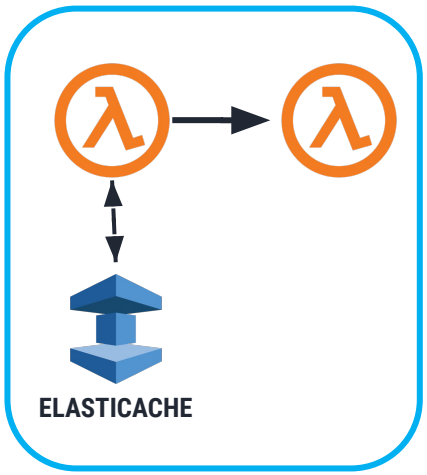
```
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

Versión 1




```
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

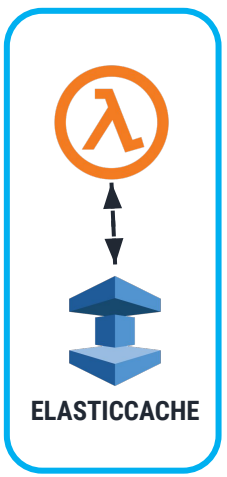
Versión 2



Versionamiento

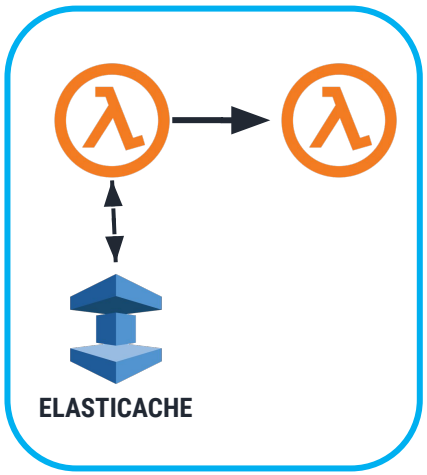
```
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

Versión 1



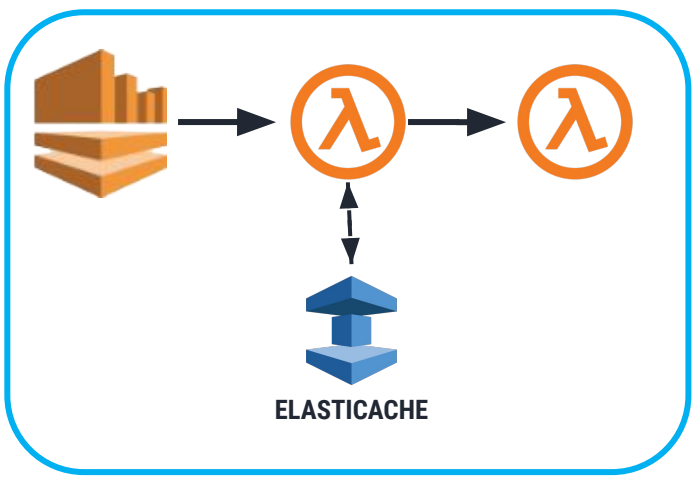

```
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

Versión 2




```
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

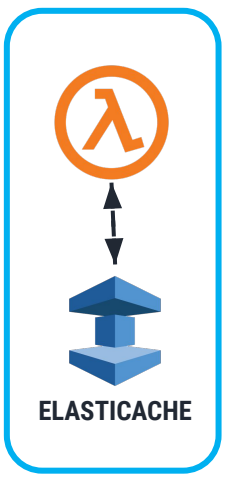
Versión 3



Versionamiento

```
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

Versión 1



Quién Solicitó?

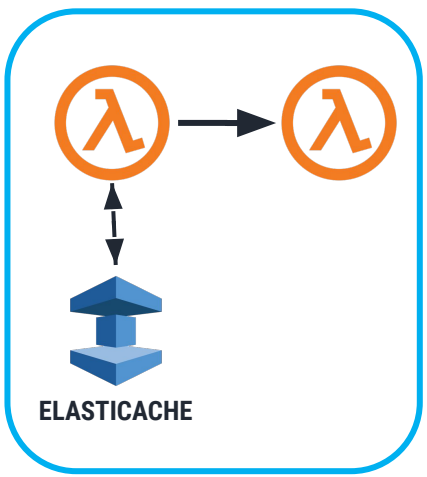
Quién aprobó?

Cuándo?

Cuál?

```
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

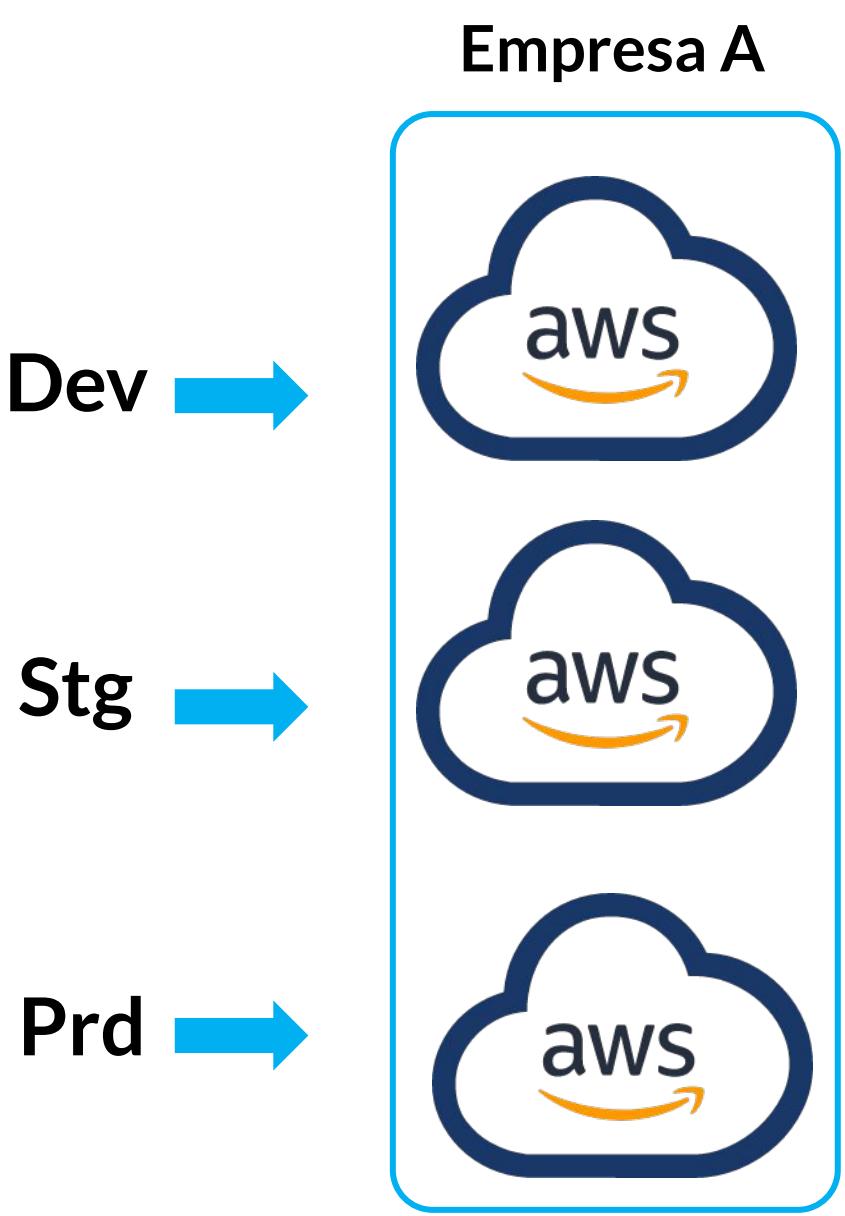
Versión 2



Trazabilidad

Eficiencia

```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```



Tiempo	Despliegues de infraestructura en minutos.
Estandarización	Ambientes completamente estandarizados.
Automatización	Utilizar servicios de CI/CD para desplegar infraestructura.

Re-Utilizable

```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```


```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

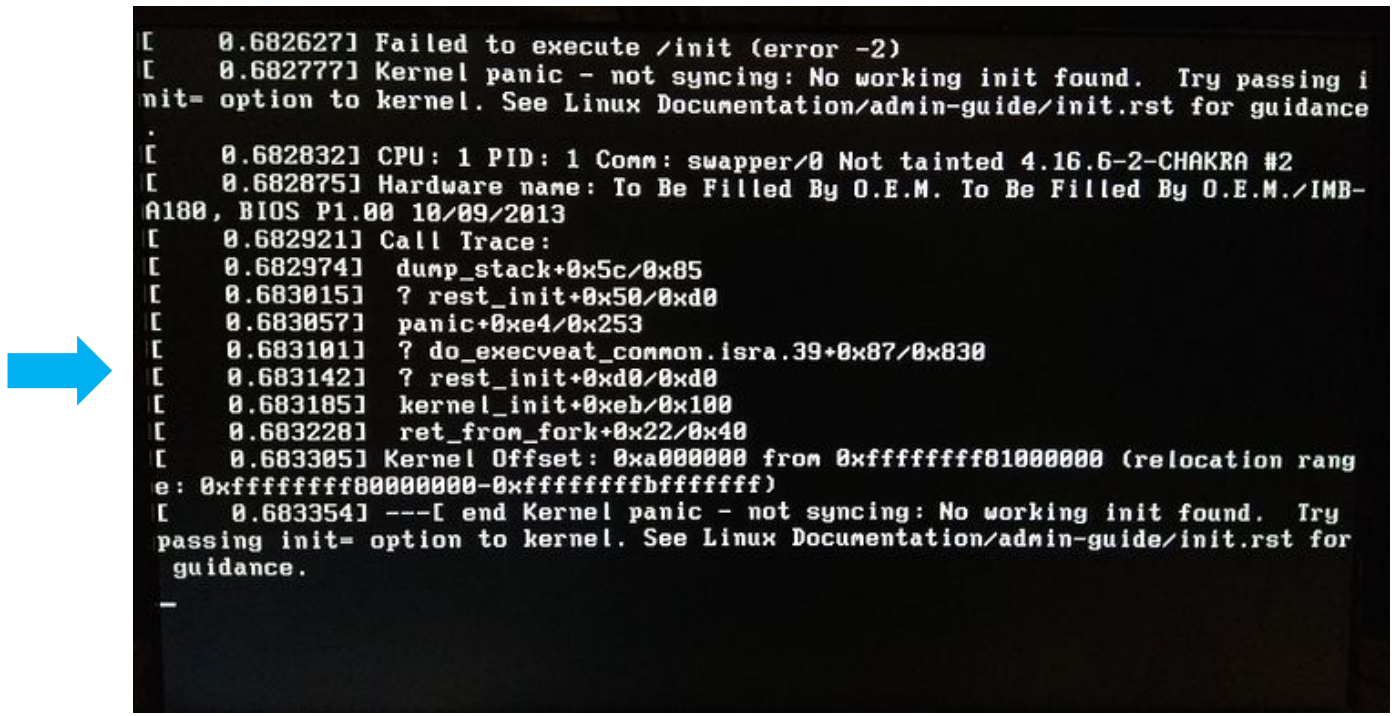
```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

Infraestructura Inmutable

Inmutable:
"No mudable, que no puede ni se puede cambiar."

Servidor Linux



Kernel Panic

Infraestructura Inmutable

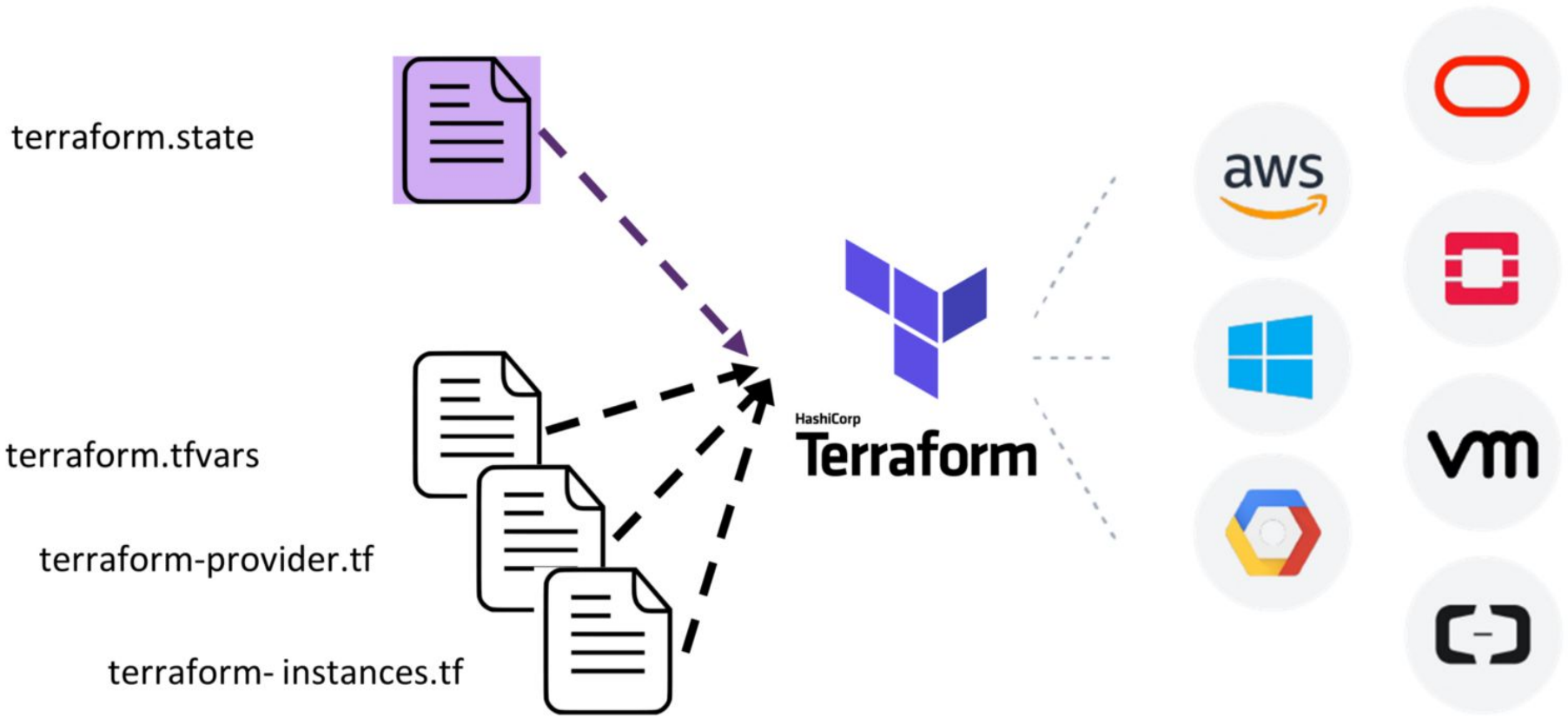
```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

Desplegar
Nuevo Servidor

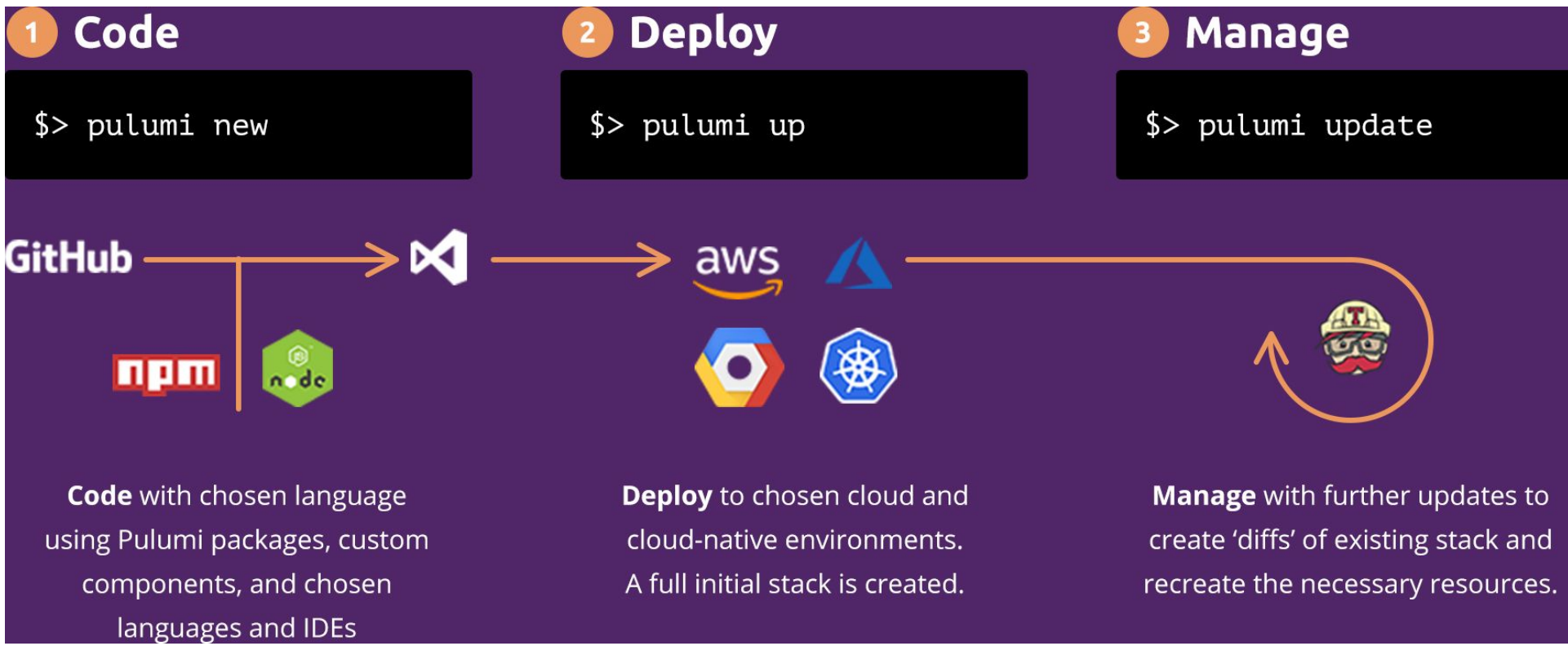
Servidor Linux

Infraestructura como Código

Terraform


terraform.state
terraform.tfvars
terraform-provider.tf
terraform- instances.tf
HashiCorp
Terraform
aws
vm

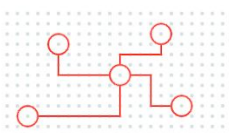
Pulumi


1 Code
$> pulumi new
2 Deploy
$> pulumi up
3 Manage
$> pulumi update
GitHub
npm
node
aws
Code with chosen language using Pulumi packages, custom components, and chosen languages and IDEs
Deploy to chosen cloud and cloud-native environments. A full initial stack is created.
Manage with further updates to create 'diffs' of existing stack and recreate the necessary resources.

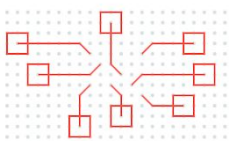
Serverless Framework

Agnósticos al Cloud Provider

Por Componentes



Codifique su Infraestructura



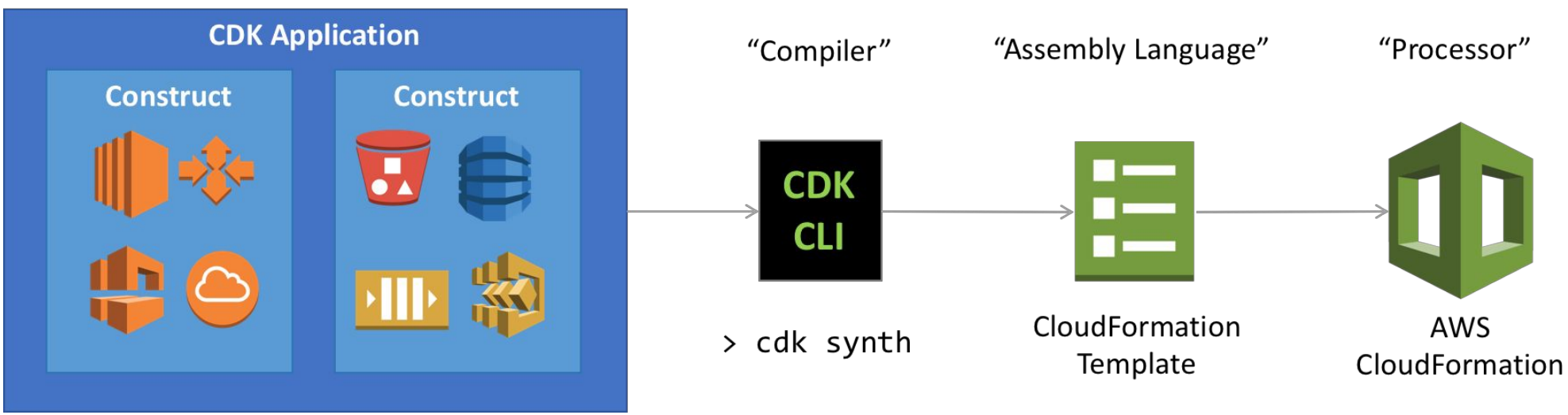
Centrado en la experiencia de desarrollo

SDK

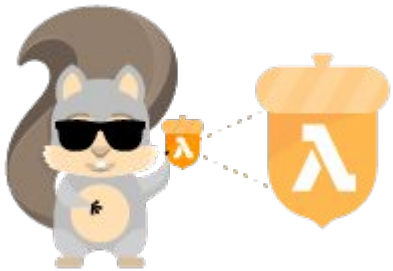
Despliegues

Librerías

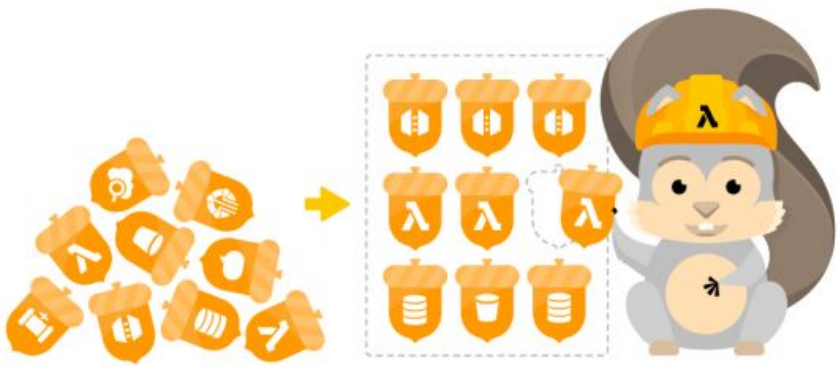
CDK


CDK Application
Construct
Construct
"Compiler"
"Assembly Language"
"Processor"
CDK
CLI
> cdk synth
CloudFormation
Template
AWS
CloudFormation

AWS SAM



SAM

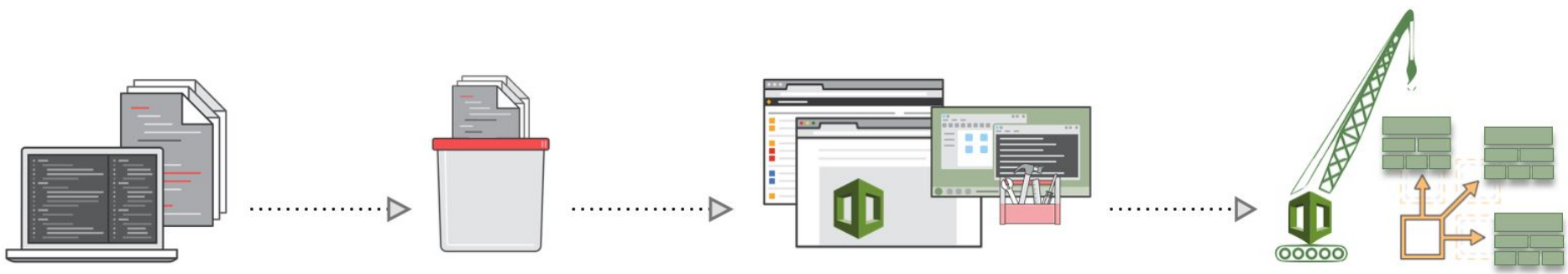


SAM construye plantillas que definen tus aplicaciones serverless.

Despliega tu plantilla SAM con Cloudformation.

Introducción y ventajas de usar Cloudformation

Cloudformation

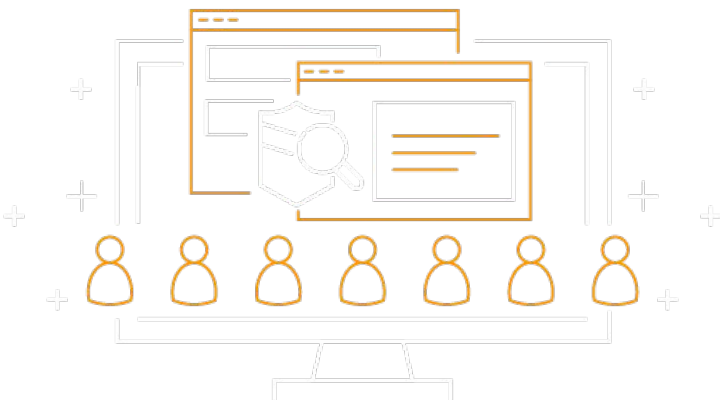


Code Check Create Deploy

Características

- **Templates** → JSON o YAML
- **Servicios** → Stacks, Stack Set y Designer.
- Integración → Full integración con todos los recursos de AWS.

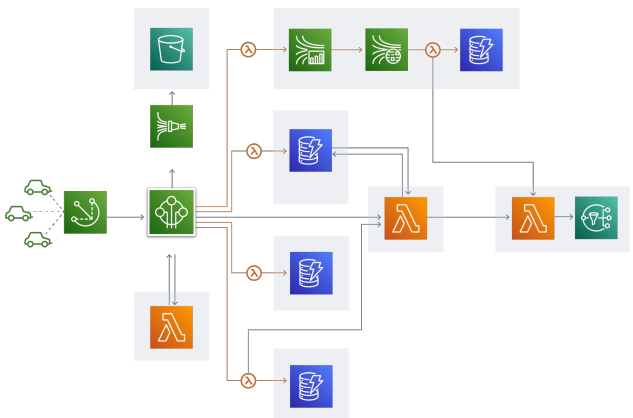
Beneficios de CloudFormation



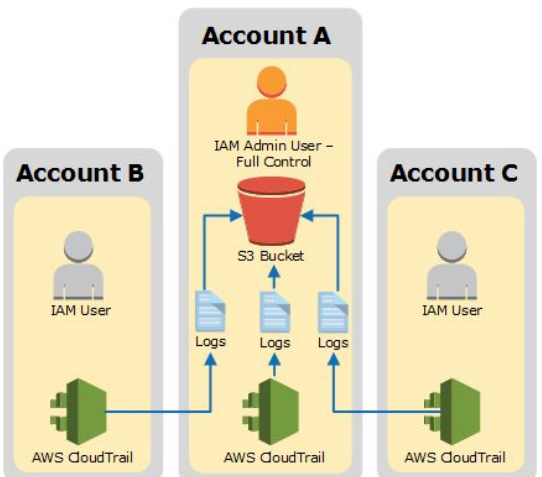
Soporte
AWS da soporte sobre tu código.

Integración
Nuevos servicios de AWS con soporte para CloudFormation.

Beneficios de CloudFormation

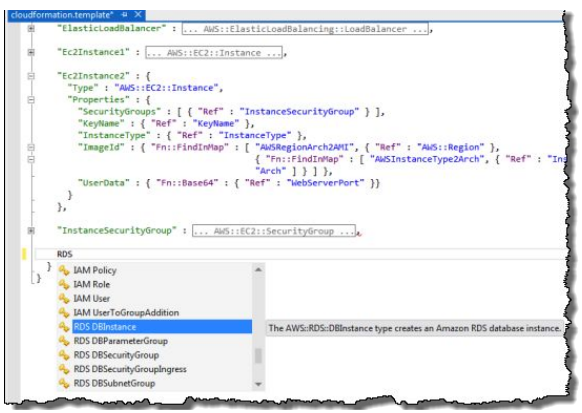


Designer
Crear arquitecturas de forma gráfica.



Multi-Cuenta
Hacer despliegues multi cuenta.

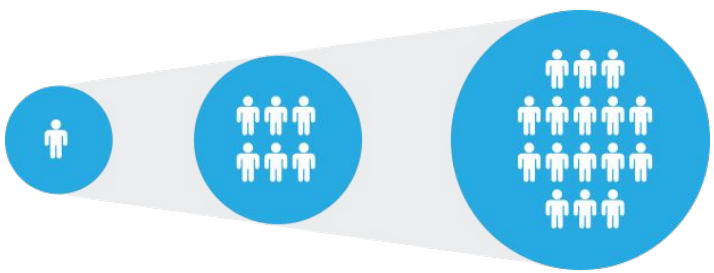
Beneficios de CloudFormation



Flexibilidad
Crear recursos dinámicamente con custom resources.

Gratis
No tiene costo.

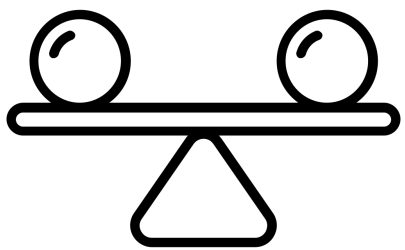
Beneficios de CloudFormation



Escalabilidad
Permite escalabilidad de recursos Multi Región.

Seguridad
Cifrado en tránsito y REST en los recursos desplegados.

Beneficios de CloudFormation



Estabilidad
Servicio administrado por AWS.

Transaccional
Espera a los recursos por crearse.
Tiene Rollback.

“

We are very happy with AWS CloudFormation, because it means we are able to use ‘one-click’ deployment of our whole infrastructure.

”

“

Expedia uses AWS CloudFormation with Chef to deploy its entire front and backend stack into its AWS environment.

”

“

All of the company’s networks are designed, built, and maintained through AWS CloudFormation templates. “This gives us the luxury of version-controlling our network, and it allows for seamless, exact network duplication for on-demand development and staging environments,” says Witoff.

”

coinbase

“

We use AWS CloudFormation for one-click network creation. It’s a very easy tool,” says Wise. Nextdoor also uses the Puppet automation tool for defining and building servers in AWS, and the RightScale cloud management solution for configuring and monitoring machines in the AWS cloud.

”

Lab 1 - Explorando la consola de Cloudformation

Anatomía de un template en Cloudformation

Template

```
---
AWSTemplateFormatVersion: "version date"

Description:
  String

Metadata:
  template metadata

Parameters:
  set of parameters

Mappings:
  set of mappings

Conditions:
  set of conditions

Transform:
  set of transforms

Resources:
  set of resources

Outputs:
  set of outputs
```

Construcción de Template

Pasos para construir un template que despliegue una función lambda usando Cloudformation.

Template

```
---
AWSTemplateFormatVersion: "version date"
```

OPCIONAL

Define las capacidades de la plantilla.
Actualmente es 2010-09-09.

```
AWSTemplateFormatVersion: '2010-09-09'
```

Description

```
Description:
  String
```

OPCIONAL

Texto que describe la plantilla.

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Mi primer lambda en Platzi
```

Metadata

```
Metadata:
  template metadata
```

OPCIONAL

Información adicional del template.

`AWS::CloudFormation::Init`
`AWS::CloudFormation::Interface`
`AWS::CloudFormation::Designer`

Parameters

```
Parameters:
  set of parameters
```

OPCIONAL

Valores que se le pasan a la plantilla cuando se crea o actualiza un stack. Pueden ser referenciados desde Resources u Outputs.

Parameters

```
Parameters:
  set of parameters
```

OPCIONAL

```
Parameters:
  myKeyPair:
    Description: Amazon EC2 Key Pair
    Type: "AWS::EC2::KeyPair::KeyName"
  mySubnetIDs:
    Description: Subnet IDs
    Type: "List<AWS::EC2::Subnet::Id>"
```

Recurso dentro de AWS

Parameters

```
Parameters:
  set of parameters
```

OPCIONAL

```
Parameters:
  DbSubnetIpBlocks:
    Description: "Comma-delimited list of three CIDR blocks"
    Type: CommaDelimitedList
    Default: "10.0.48.0/24, 10.0.112.0/24, 10.0.176.0/24"
```

Ingreso de Listas

Parameters

```
Parameters:
  set of parameters
```

OPCIONAL

```
Parameters:
  DBPort:
    Default: 3306
    Description: TCP/IP port for the database
    Type: Number
    MinValue: 1150
    MaxValue: 65535
  DBPwd:
    NoEcho: true
    Description: The database admin account password
    Type: String
    MinLength: 1
    MaxLength: 41
    AllowedPattern: ^[a-zA-Z0-9]*$
```

Rango Permitido

Valor Seguro (****)

Cumplir Expresión

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Mi primer lambda en Platzi

Parameters:
  NombreLambda:
    Description: Nombre de la función lambda
    Type: String

  RuntimeLambda:
    Description: Ingresa el runtime de la función lambda
    Type: String
    Default: python3.7
    AllowedValues:
        - python3.7
        - python2.7
        - ruby2.5
        - nodejs8.10
        - java8
        - dotnetcore2.1
```

Mappings

```
Mappings:
  set of mappings
```

OPCIONAL

Llave valor asociados que se usan para parámetros condicionales. Similar a una tabla de búsqueda. Utiliza la función Fn::FindInMap.

Mappings

```
Mappings:
  set of mappings
```

OPCIONAL

```
Mappings:
  RegionMap:
    us-east-1:
      "HVM64": "ami-0ff8a91507f77f867"
    us-west-1:
      "HVM64": "ami-0bdb828fd58c52235"
    eu-west-1:
      "HVM64": "ami-047bb4163c506cd98"
    ap-southeast-1:
      "HVM64": "ami-08569b978cc4dfa10"
    ap-northeast-1:
      "HVM64": "ami-06cd52961ce9f0d85"
```

Selecciona la AMI dependiendo la región donde se lance el Stack.

Mappings

```
Mappings:
  set of mappings
```

OPCIONAL

```
Mappings:
  RegionAndInstanceTypeToAMIID:
    us-east-1:
      test: "ami-8ff710e2"
      prod: "ami-f5f41398"
    us-west-2:
      test: "ami-eff1028f"
      prod: "ami-d0f506b0"
```

Selecciona la AMI dependiendo la región y el ambiente: Prod o Test.

Conditions

```
Conditions:
  set of conditions
```

OPCIONAL

Controlan si se crean ciertos recursos o si se asigna un valor a ciertas propiedades durante la creación del stack.

Conditions

```
Conditions:
  set of conditions
```

OPCIONAL

```
NewVolume:
  Type: "AWS::EC2::Volume"
  Condition: CreateProdResources
  Properties:
    Size: 100
    AvailabilityZone: !GetAtt EC2Instance.AvailabilityZone
```

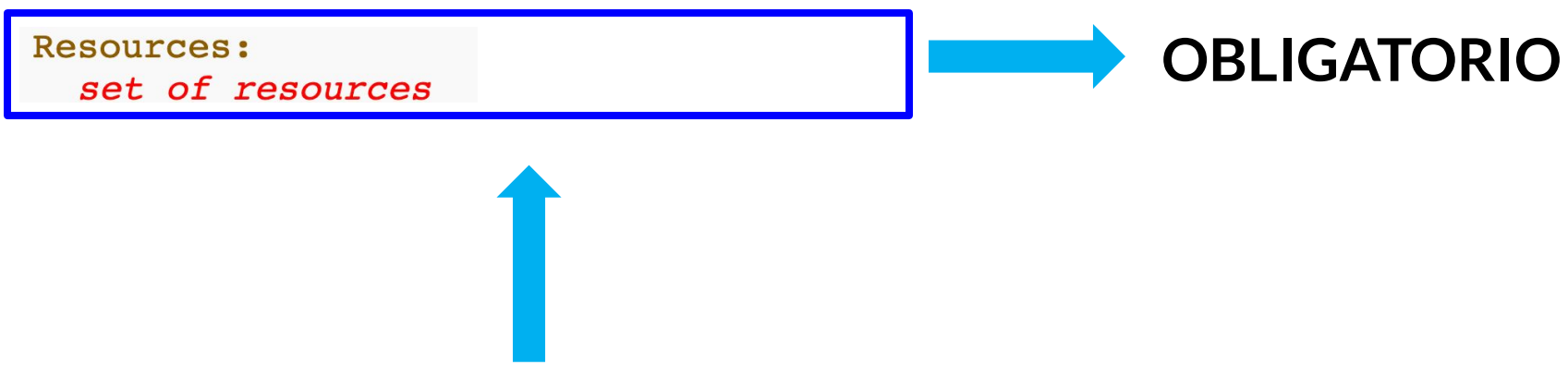
Si la Condition es true, entonces crea un nuevo volúmen.

Transform

OPCIONAL

Para aplicaciones serverless, si se especifica se pueden usar sintaxis de AWS SAM.

Resources



Especifica los recursos y las propiedades a crear.

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Mi primer lambda en Platzi

Parameters:
  NombreLambda:
    Description: Nombre de la función lambda
    Type: String

  RuntimeLambda:
    Description: Ingresa el runtime de la función lambda
    Type: String
    Default: python3.7
    AllowedValues:
        - python3.7
        - python2.7
        - ruby2.5
        - nodejs8.10
        - java8
        - dotnetcore2.1
```

```
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: !Ref NombreLambda
        Handler: lambda_function.lambda_handler
        Runtime: !Ref RuntimeLambda
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRolePlatzi.Arn
```

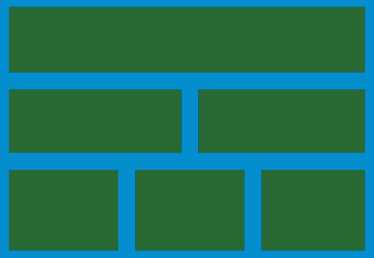
Outputs

```
Outputs:
  set of outputs
```

OPCIONAL

Valores devueltos de las propiedades del stack.

```
Outputs:
  LambdaARN:
    Description: "ARN de la función lambda"
    Value:
      !GetAtt LambdaPlatzi.Arn
    Export:
      Name: LambdaPlatziArn
  LambdaName:
    Description: "Nombre de la función lambda"
    Value:
      !Ref LambdaPlatzi
    Export:
      Name: LambdaPlatziName
```



Stacks

Características

- **¿Qué es un Stack?**

Colección de recursos que se manejan como una unidad.

- **Gestión de Recursos**

Cloudformation asegura que todos los recursos sean creados o eliminados.

Características

- **¿Qué pasa si un recurso falla en crearse?**

Se hace rollback a todos los recursos del stack.

- **¿Qué pasa si borro un stack?**

Todos los recursos asociados se borran.

Características

- **¿Qué es un Drift?**

Detecta una desviación entre el stack y los recursos desplegados.

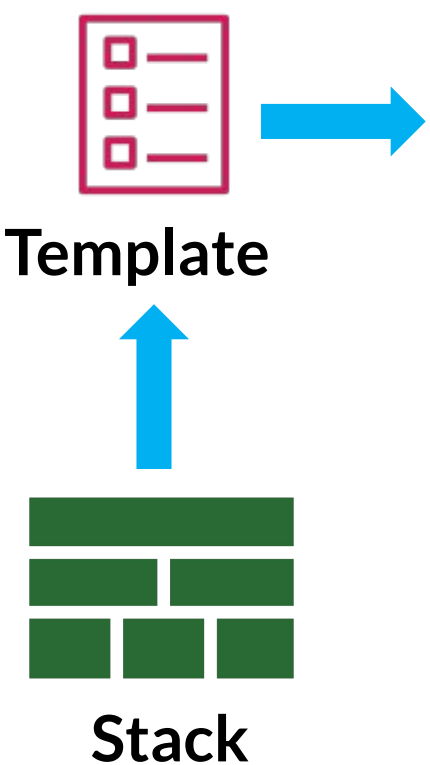
- **¿Qué puedo identificar con un Drift?**

Recursos agregados, eliminados y con propiedades diferentes.

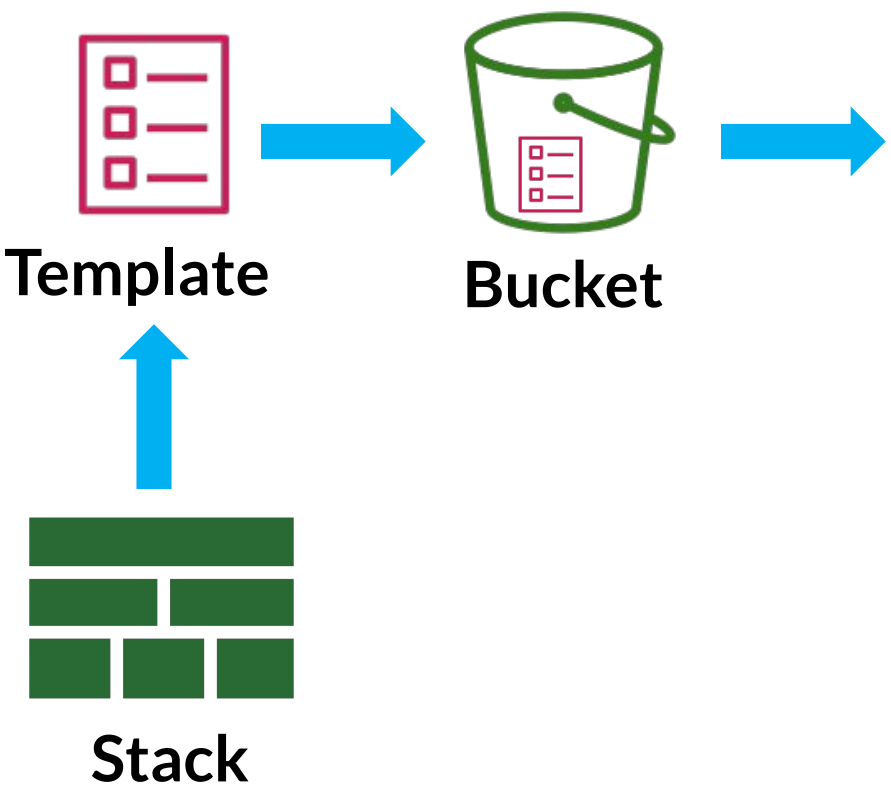
Stacks

Stack

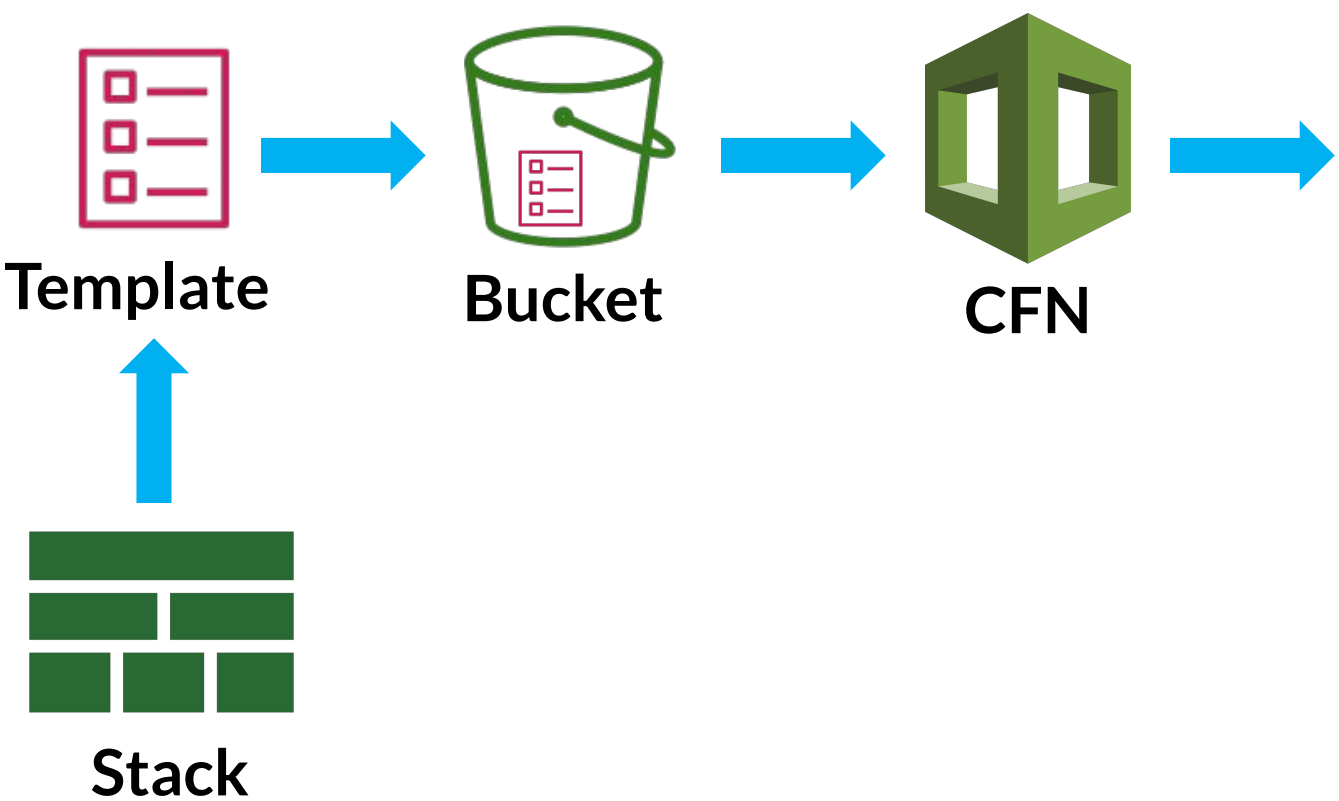
Stacks



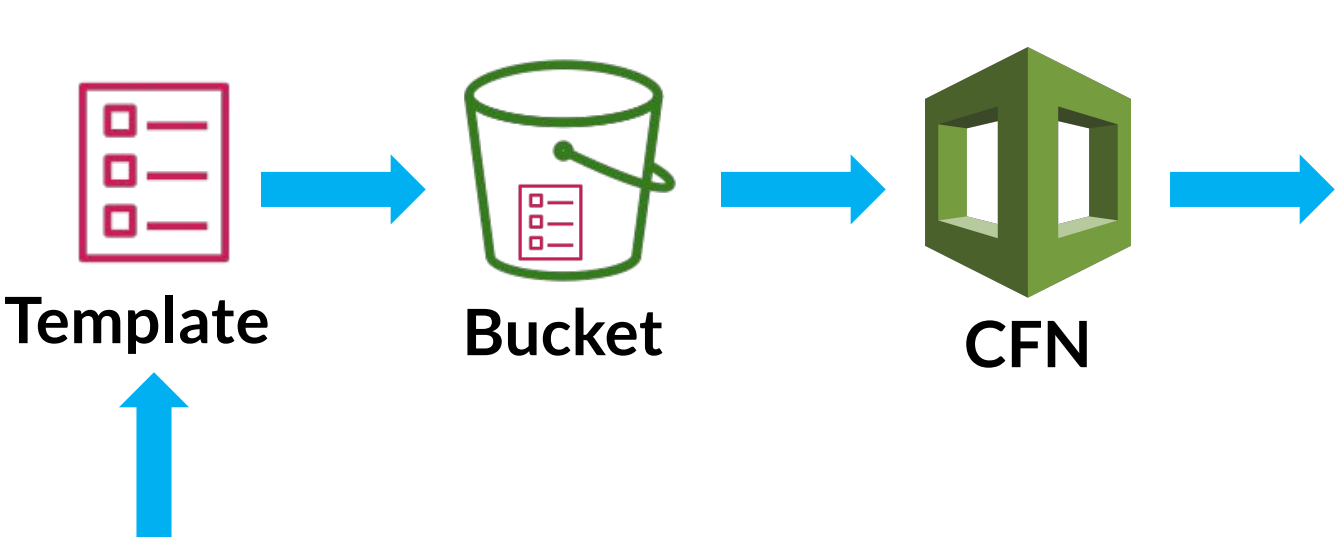
Stacks

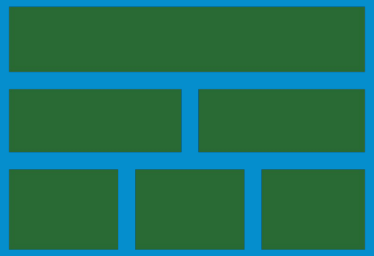


Stacks



Stacks


Template
Bucket
CFN
AWS Cloud
Availability Zone 1
Availability Zone 2
VPC
NAT Gateway
Amazon EC2 Auto Scaling
NAT Gateway
Instance
Auto Scaling group
Instance
Instance
Auto Scaling group
Instance
Recursos
Stack



Stacks Set

Características

- **¿Qué tipos de cuentas existen?**

Cuentas administrador y cuentas target.

- **¿Desde dónde se despliegan recursos Multi Cuenta?**

Se debe hacer desde una cuenta maestra que tenga permisos sobre las otras.

Características

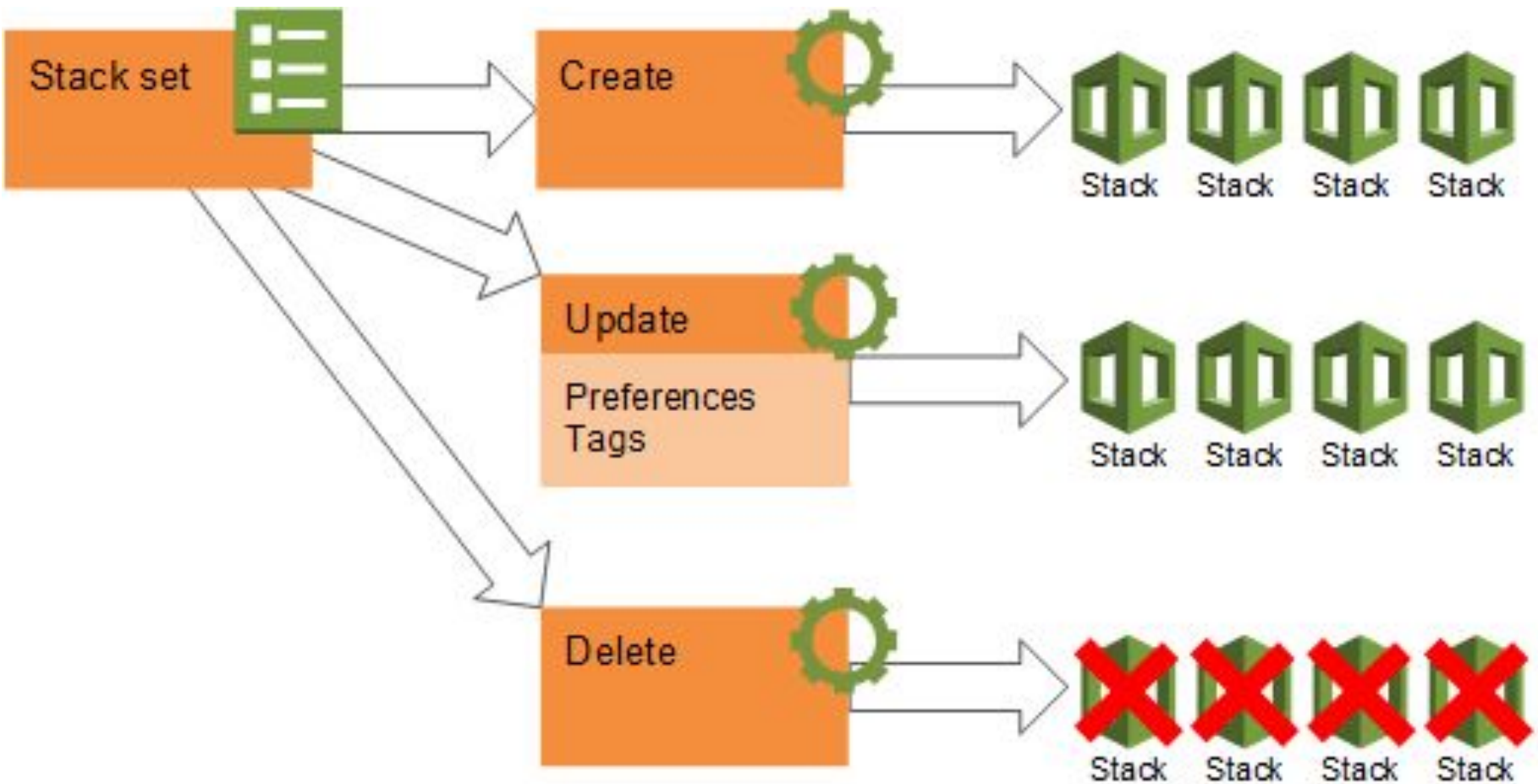
- **Instancia de un Stack**

Hace referencia a un stack dentro de una cuenta.

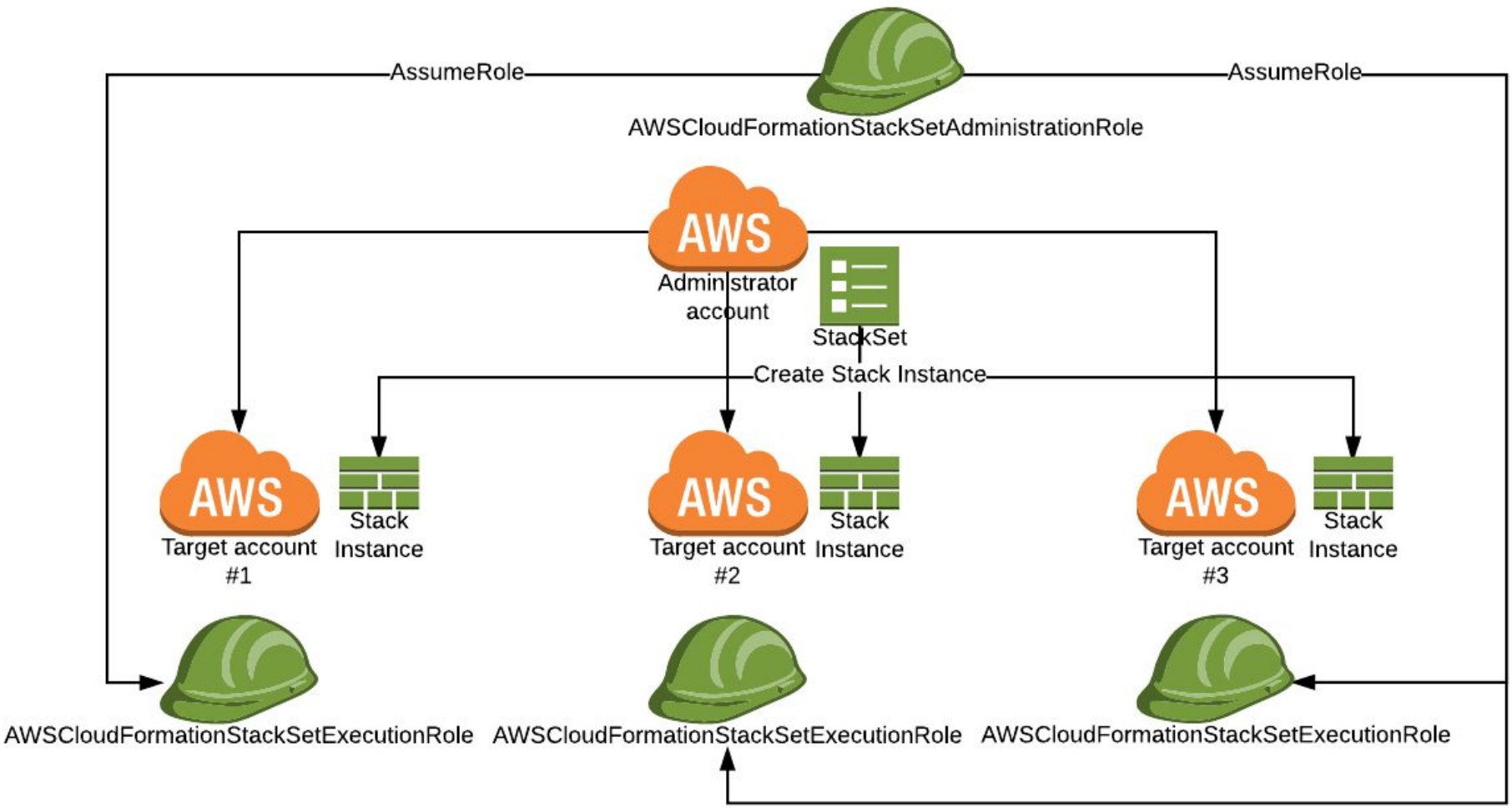
- **Diferentes Parámetros**

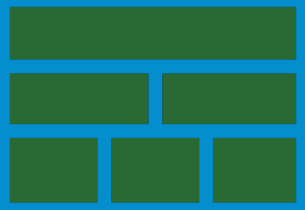
Se pueden desplegar stack set que cambien de parámetros dependiendo de la cuenta destino.

Stacks Set

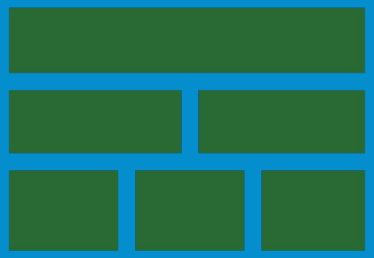

Stack set
Create
Update
Preferences
Tags
Delete
Stack
Stack
Stack
Stack
Stack
Stack
Stack
Stack
Stack
Stack
Stack
Stack

Stacks Set


AssumeRole
AssumeRole
AWSCloudFormationStackSetAdministrationRole
AWS
Administrator account
StackSet
Create Stack Instance
AWS
Target account #1
Stack Instance
AWS
Target account #2
Stack Instance
AWS
Target account #3
Stack Instance
AWSCloudFormationStackSetExecutionRole
AWSCloudFormationStackSetExecutionRole
AWSCloudFormationStackSetExecutionRole



Lab 2 - Desplegando un Stack Set en un esquema multi-cuenta



Nested Stacks

Límites de Cloudformation

100 Mappings	Stack
200 Recursos	Stack
51,200 bytes	Cuerpo de template para CreateStack, UpdateStack, or ValidateTemplate.
460,800 bytes	Tamaño máximo de template en S3

Dónde usar Nested Stacks

- **Límites**

Para crear recursos que necesiten pasar los límites de Cloudformation.

- **Granularidad**

Cada recurso queda con un stack independiente.

Dónde usar Nested Stacks

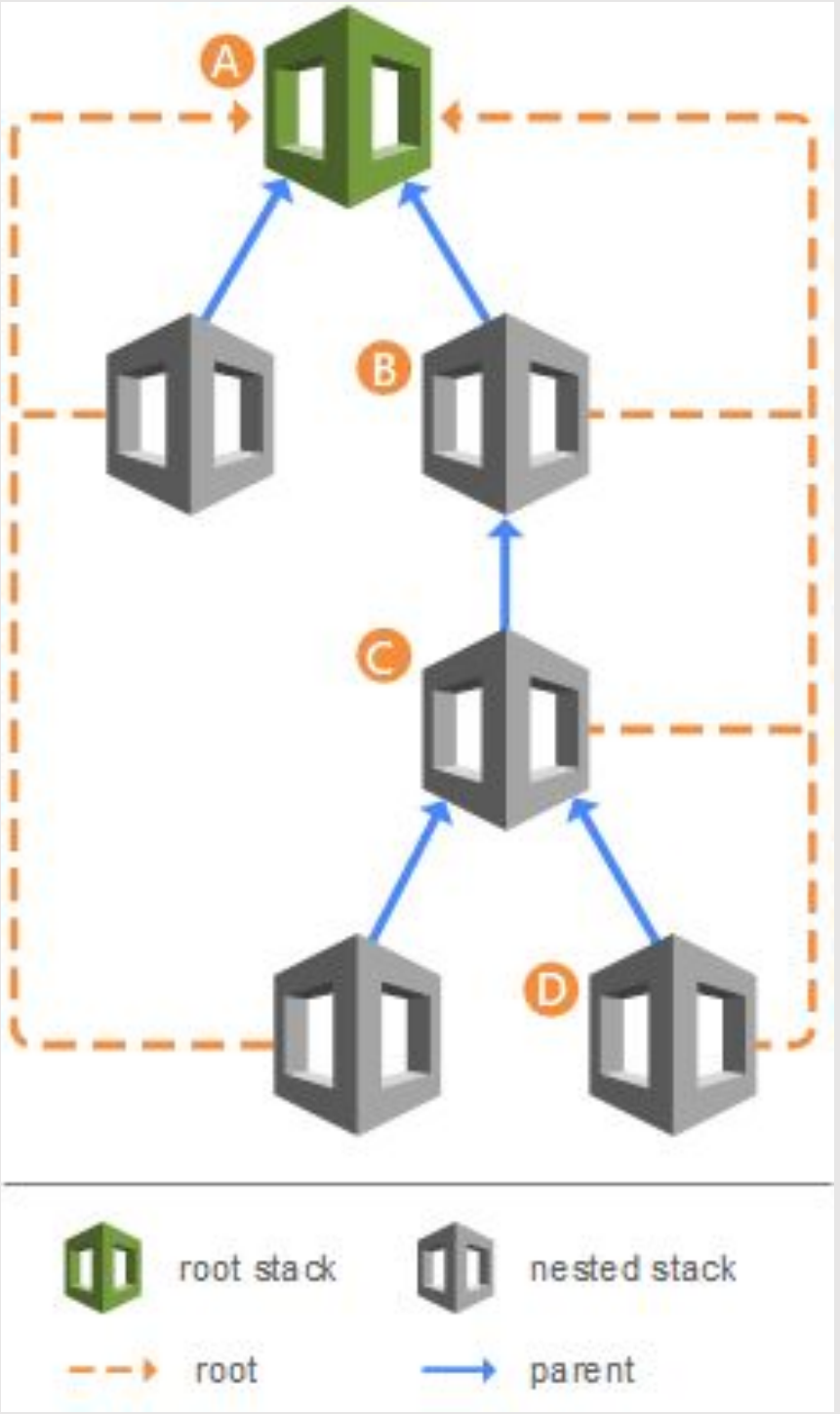
- **Orden**

Se pueden crear precedencias y condiciones entre recursos.

- **Interacción**

Los stacks se comunican entre sí a través de outputs.

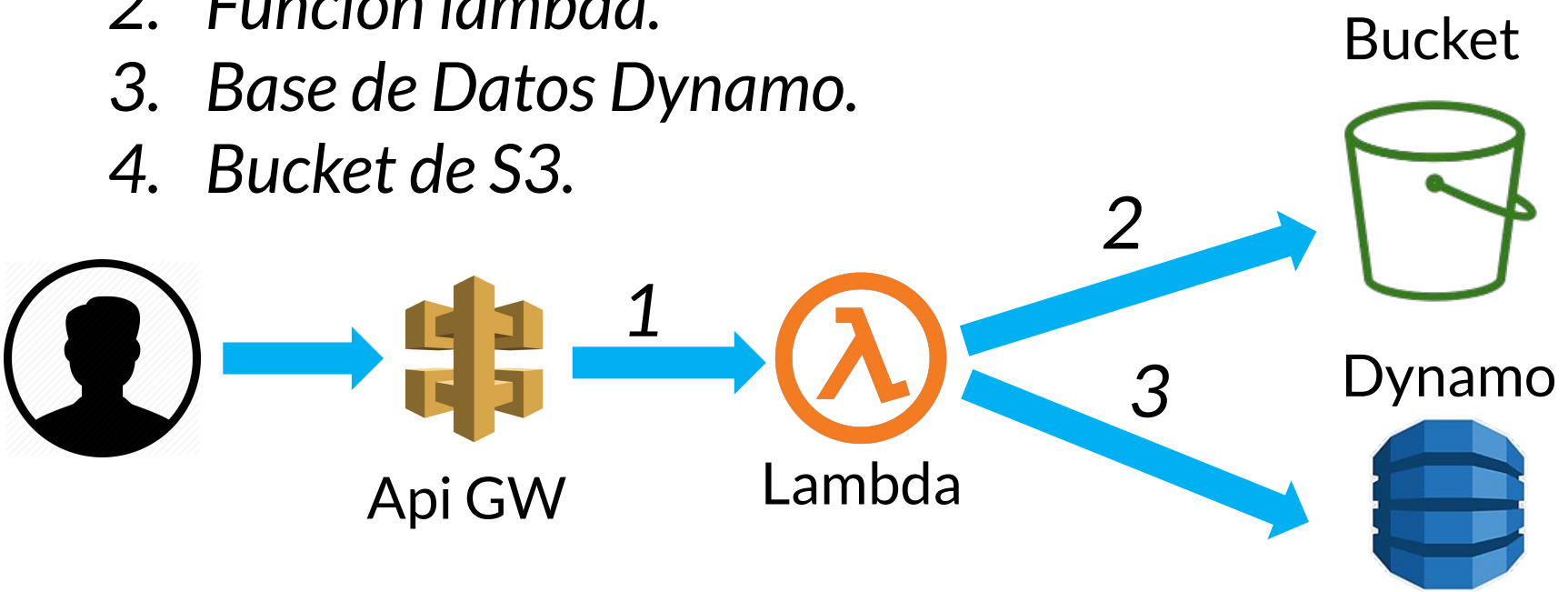
Nested Stacks



Problema

Se necesita desplegar una aplicación que se compone de:

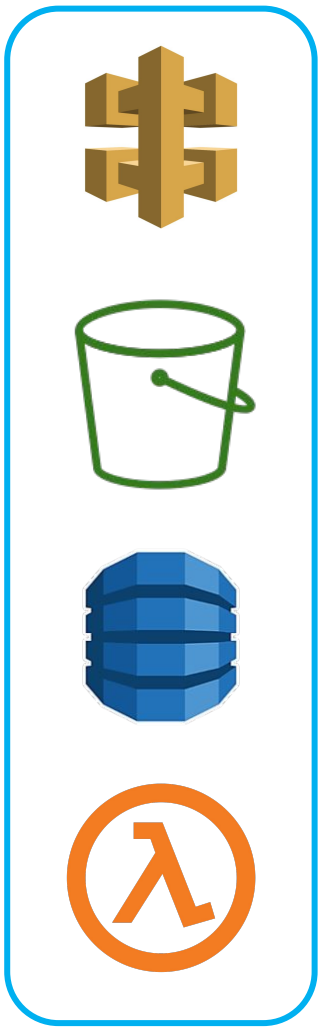
1. *Api Gateway.*
2. *Función lambda.*
3. *Base de Datos Dynamo.*
4. *Bucket de S3.*



Sin Stack Anidados

```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

Template



Con Stack Anidados

```yaml
Resources:
  LambdaPlatzi:
    Type: AWS::Serverless::Function
    Properties:
      FunctionName: MiPrimeraLambda
      Handler: lambda_function.lambda_handler
      Runtime: python3.7
      MemorySize: 512
      Timeout: 600
      Role: !GetAtt LambdaRole.Arn
```

Master Stack

```yaml
Resources:
  LambdaPlatzi:
    Type: AWS::Serverless::Function
    Properties:
      FunctionName: MiPrimeraLambda
      Handler: lambda_function.lambda_handler
      Runtime: python3.7
      MemorySize: 512
      Timeout: 600
      Role: !GetAtt LambdaRole.Arn
```

```yaml
Resources:
  LambdaPlatzi:
    Type: AWS::Serverless::Function
    Properties:
      FunctionName: MiPrimeraLambda
      Handler: lambda_function.lambda_handler
      Runtime: python3.7
      MemorySize: 512
      Timeout: 600
      Role: !GetAtt LambdaRole.Arn
```

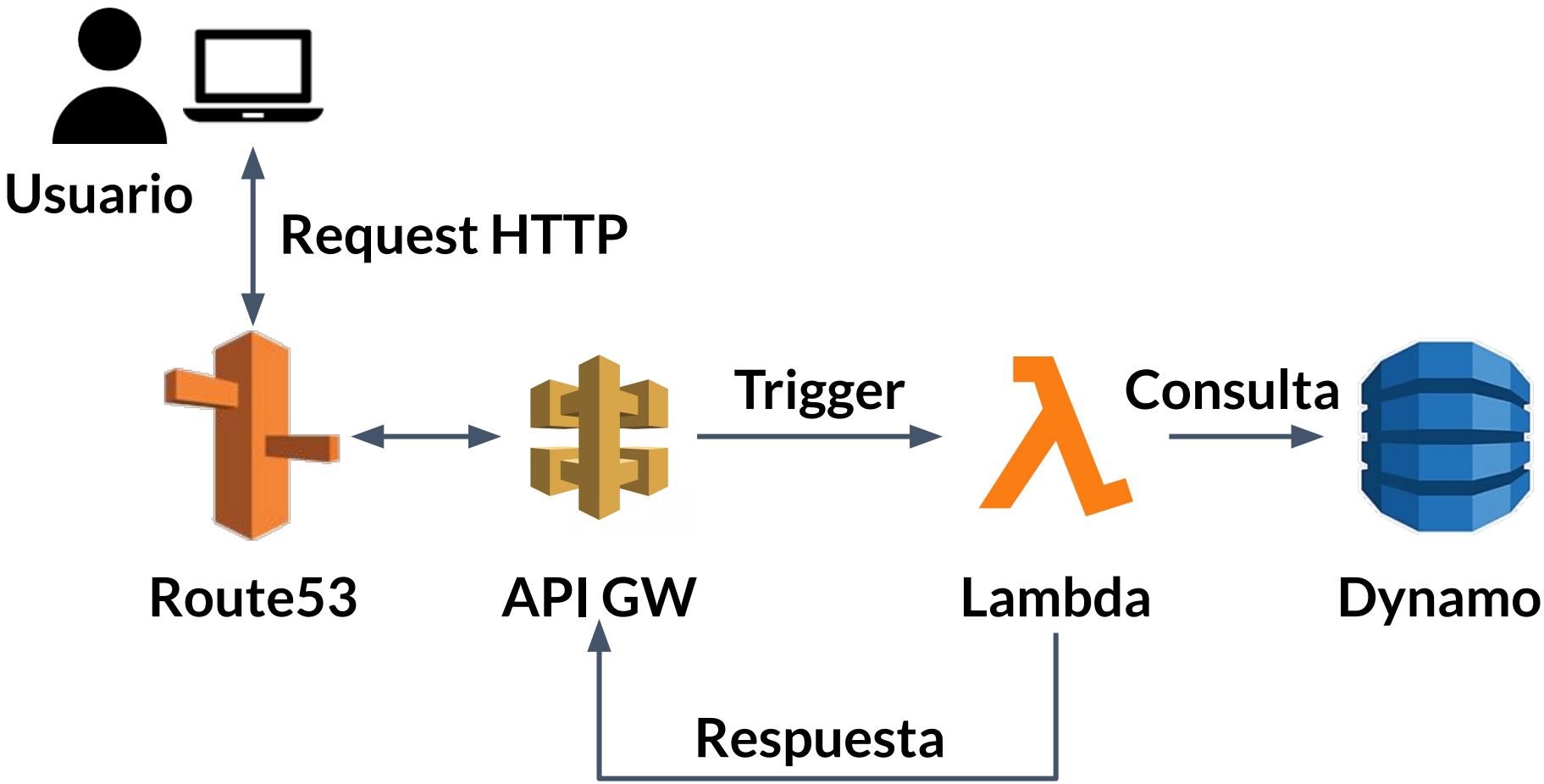
```yaml
Resources:
  LambdaPlatzi:
    Type: AWS::Serverless::Function
    Properties:
      FunctionName: MiPrimeraLambda
      Handler: lambda_function.lambda_handler
      Runtime: python3.7
      MemorySize: 512
      Timeout: 600
      Role: !GetAtt LambdaRole.Arn
```

```yaml
Resources:
  LambdaPlatzi:
    Type: AWS::Serverless::Function
    Properties:
      FunctionName: MiPrimeraLambda
      Handler: lambda_function.lambda_handler
      Runtime: python3.7
      MemorySize: 512
      Timeout: 600
      Role: !GetAtt LambdaRole.Arn
```

```yaml
AWSTemplateFormatVersion: 2010-09-09
Resources:
    ApiGateway:
        Type: "AWS::CloudFormation::Stack"
        Properties:
            TemplateURL: https://url/api.yml

    Lambda:
        Type: "AWS::CloudFormation::Stack"
        Properties:
            TemplateURL: https://url/lambda.yml

    Dynamo:
        Type: "AWS::CloudFormation::Stack"
        Properties:
            TemplateURL: https://url/dynamo.yml

    Bucket:
        Type: "AWS::CloudFormation::Stack"
        Properties:
            TemplateURL: https://url/bucket.yml
```

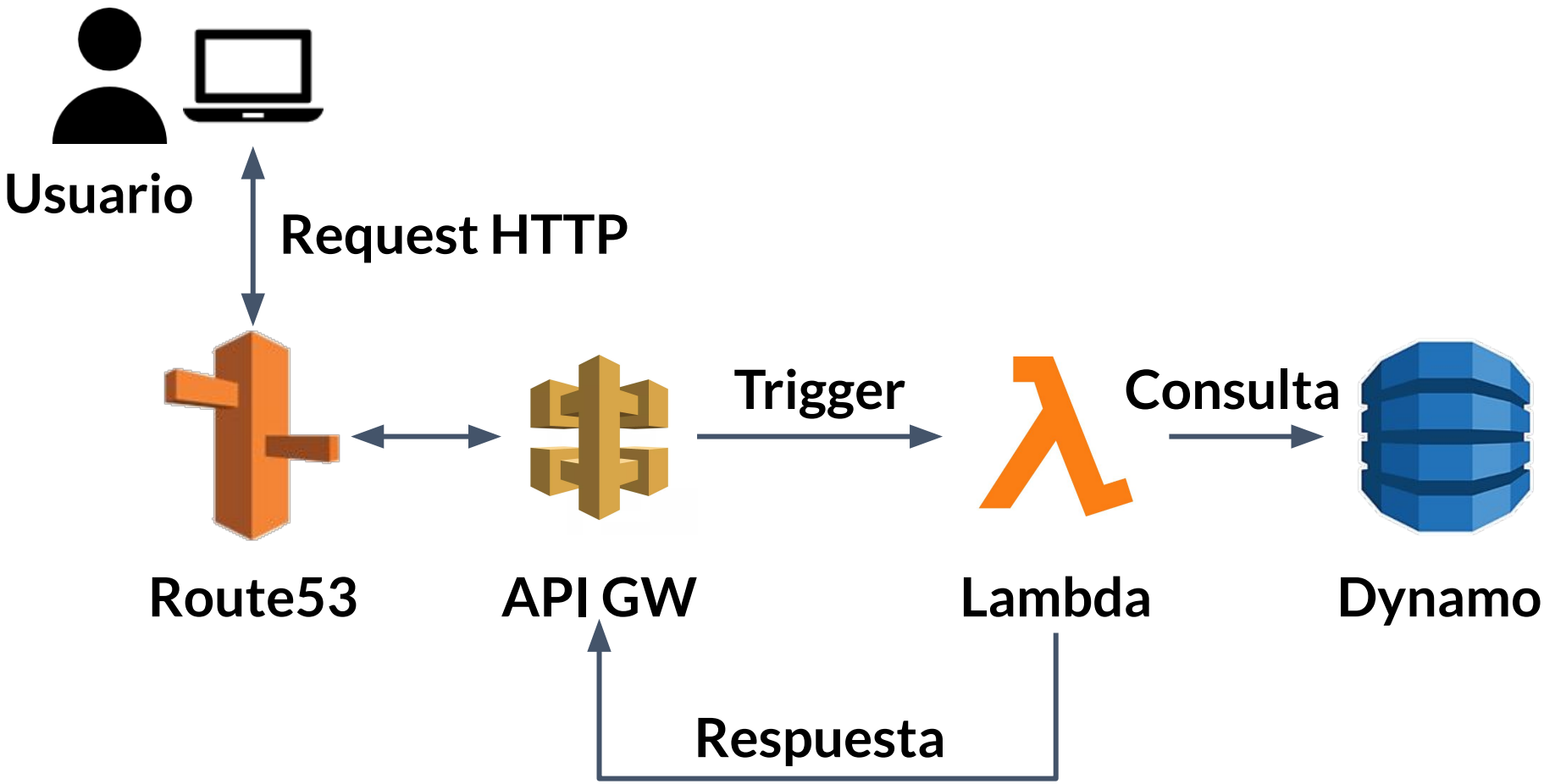
Lab 3 - Creando nuestro primer stack

VotaNet App


Usuario
Request HTTP
Route53
API GW
Trigger
Lambda
Consulta
Dynamo
Respuesta

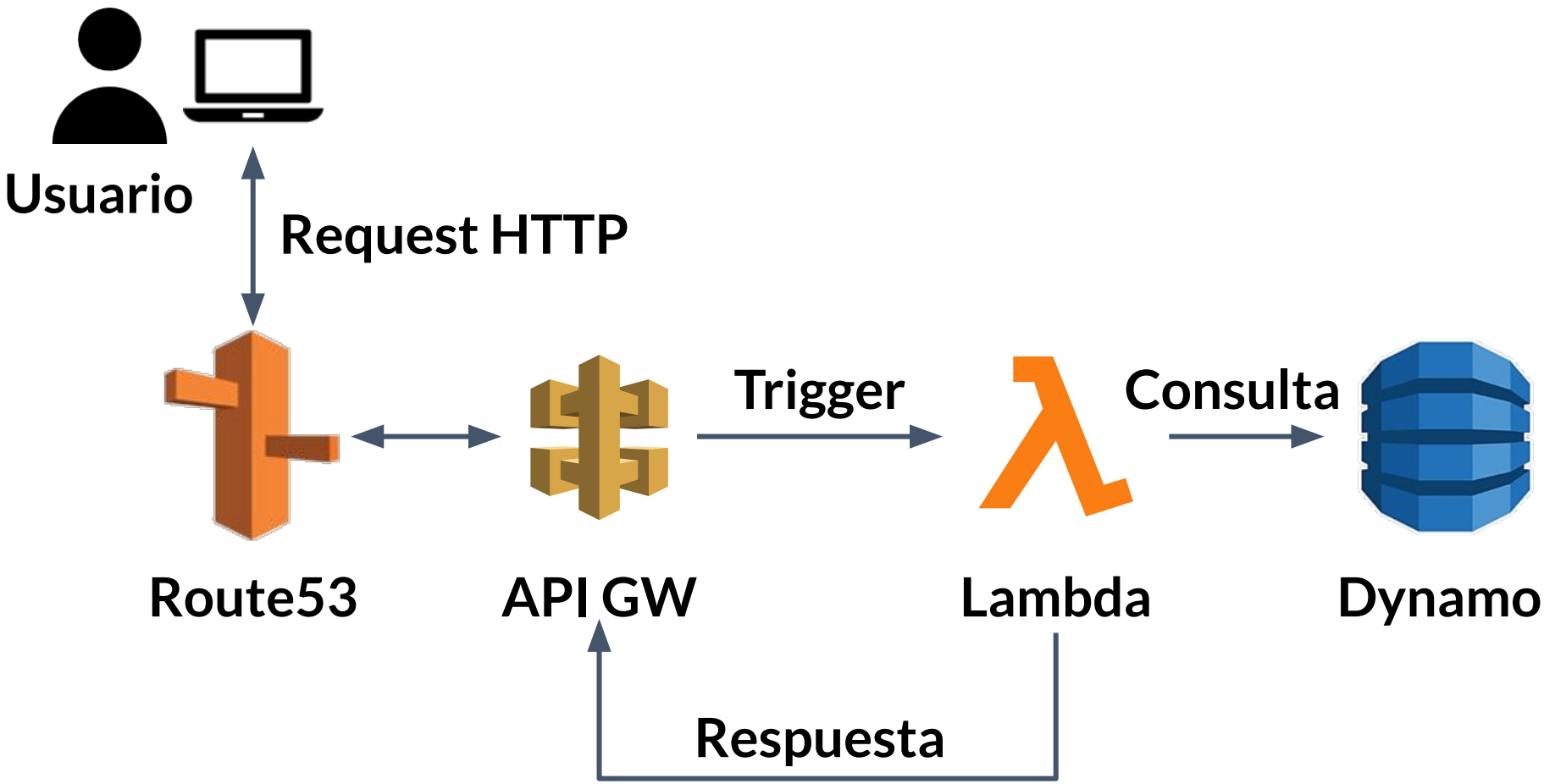
Lab 4 - Creando Stack anidados

VotaNet App


Usuario
Request HTTP
Route53
API GW
Trigger
Lambda
Consulta
Dynamo
Respuesta

Lab 5 - Cloudformation Designer

VotaNet App


Usuario
Request HTTP
Route53
API GW
Trigger
Lambda
Consulta
Dynamo
Respuesta

Funciones en Cloudformation

Funciones Intrínsecas:

GetAtt, FindInMap, Join, Split y Select

GetAtt

- **Funcionalidad**

Devuelve el valor de un atributo de un recurso en el template.

- **Composición**

Se compone del nombre del recurso y del atributo.

Sintaxis versión 1

```
{ "Fn::GetAtt" : ["logicalNameOfResource", "attributeName"]}
```

Sintaxis versión 2

```
Fn::GetAtt: [ logicalNameOfResource, attributeName ]
```

Sintaxis versión 3

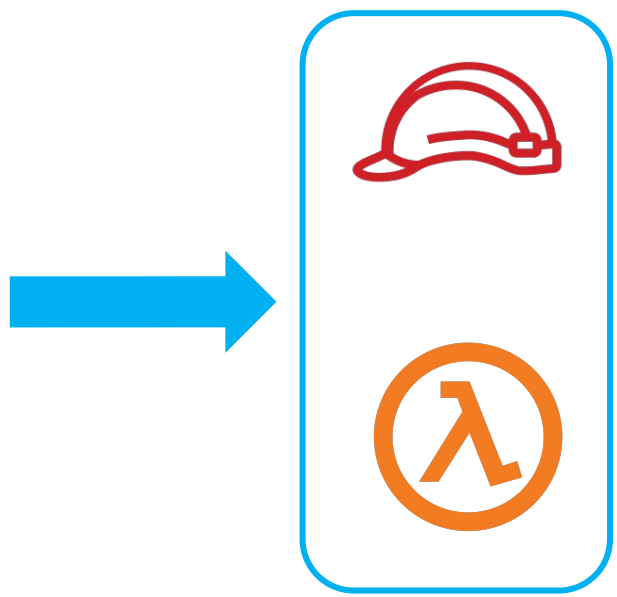
```
!GetAtt logicalNameOfResource.attributeName
```

Cuándo usar GetAtt?

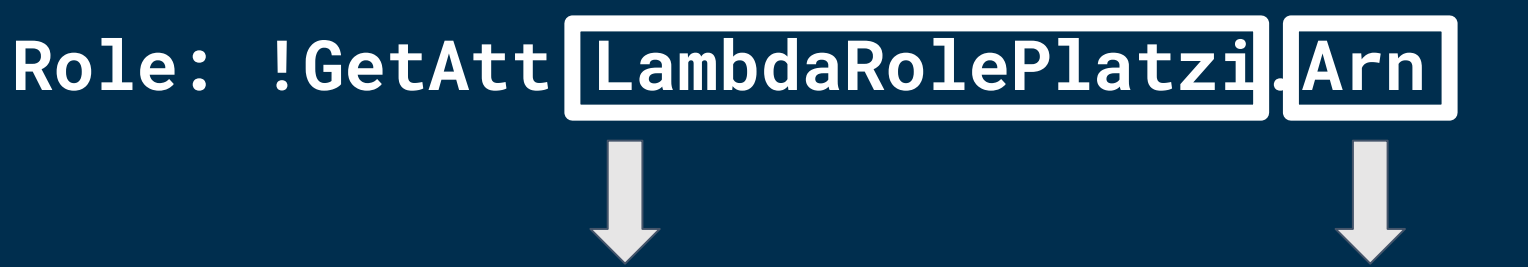
Cuando se quiere tomar algún atributo de un recurso dentro del mismo stack.

```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

Template



```
Resources:
   LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: !Ref NombreLambda
        Handler: lambda_function.lambda_handler
        Runtime: !Ref RuntimeLambda
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRolePlatzi.Arn


   LambdaRolePlatzi
      Type: AWS::IAM::Role
      Properties:
            ManagedPolicyArns:
```


Role: !GetAtt LambdaRolePlatzi.Arn
Nombre lógico del recurso.
Atributo.

FindInMap

- **Funcionalidad.**

Devuelve el valor correspondiente al map declarado en la sección Mappings.

- **Composición.**

MapName, TopLevelKey y SecondLevelKey.

Sintaxis versión 1

```
{"Fn::FindInMap" : ["MapName", "TopLevelKey", "SecondLevelKey"]}
```

Sintaxis versión 2

```
Fn::FindInMap: [ MapName, TopLevelKey, SecondLevelKey ]
```

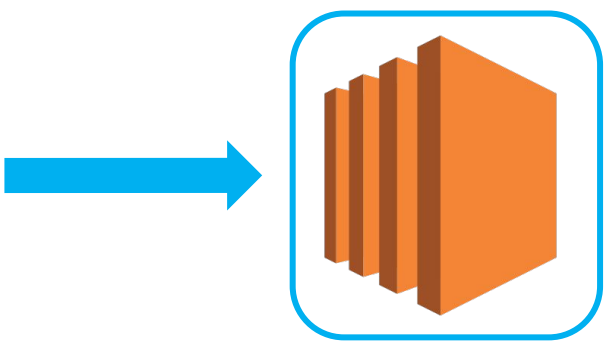
Sintaxis versión 3

```
!FindInMap [ MapName, TopLevelKey, SecondLevelKey ]
```

¿Cuándo usar FindInMap?

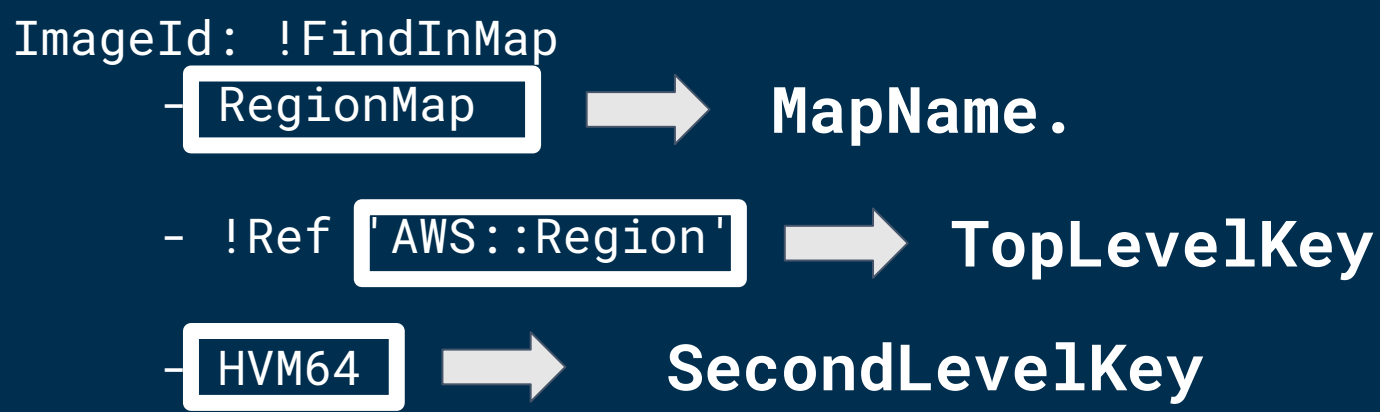
Cuando necesitemos en el template algún valor de la sección Mappings.

```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```



Template funcional en cualquier región

```
Mappings:
  RegionMap:
    us-east-1:
      HVM64: "ami-0ff8a91507f77f867"
    us-west-1:
      HVM64: "ami-0bdb828fd58c52235"
    eu-west-1:
      HVM64: "ami-047bb4163c506cd98"
Resources:
  MyPlatziServer:
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId: !FindInMap
        - RegionMap
        - !Ref 'AWS::Region'
        - HVM64
      InstanceType: m1.small
```


ImageId: !FindInMap
- RegionMap
MapName.
- !Ref 'AWS::Region'
TopLevelKey
- HVM64
SecondLevelKey

Join

- **Funcionalidad**

Une un listado de valores en uno solo.

- **Composición**

Se compone de un listado de valores y un delimitador.

Sintaxis versión 1

```
{ "Fn::Join" : [ "delimiter", [ list of values ] ] }
```

Sintaxis versión 2

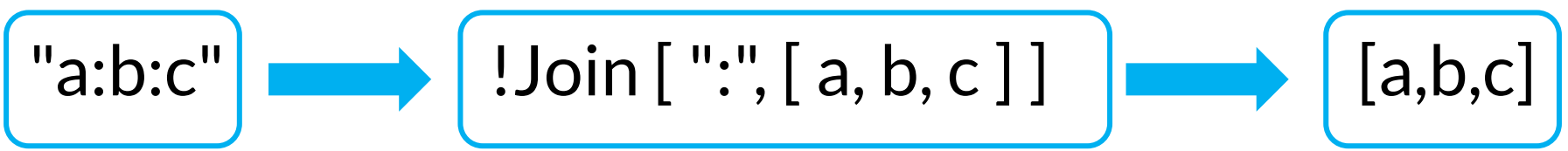
```
Fn::Join: [ delimiter, [ list of values ] ]
```

Sintaxis versión 3

```
!Join [ delimiter, [ list of values ] ]
```

¿Cuándo usar Join?

Cuando necesitemos unir varios valores
y separarlos por algún delimitador.



Split y Select

- **SPLIT**

  Dividir una cadena en una lista de valores.

- **SELECT**

  Selecciona un valor de una lista de valores.

Sintaxis versión 1

```
{ "Fn::Split" : ["delimiter", "source string"] }
```

Sintaxis versión 2

```
Fn::Split: [ delimiter, source string ]
```

Sintaxis versión 3

```
!Split [ delimiter, source string ]
```

Sintaxis versión 1

```
{ "Fn::Select" : [ index, listOfObjects ] }
```

Sintaxis versión 2

```
Fn::Select: [ index, listOfObjects ]
```

Sintaxis versión 3

```
!Select [ index, listOfObjects ]
```

¿Cuándo usar Split/Select?

Cuando necesitemos dividir un arreglo en valores independientes y tomar solo uno de esos valores.

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
    Cuenta:
        Description: "Cuenta de AWS Origen"
        Type: String
        Default: "OPS:XXXXXXXXXXXX"
        AllowedValues:
            - "OPS:XXXXXXXXXXXX"
            - "DEV:YYYYYYYYYYYY"
            - "PRD:ZZZZZZZZZZZZ"
            - "STG:WWWWWWWWWWWW"
```

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
    Cuenta:
        Description: "Cuenta de AWS Origen"
        Type: String
        Default: "OPS:XXXXXXXXXXXX"
        AllowedValues:
            - "OPS:  XXXXXXXXXXXX"
            - "DEV:  YYYYYYYYYYYY"
            - "PRD:  ZZZZZZZZZZZZ"
            - "STG:  WWWWWWWWWWWW"
```

Tomar el número de cuenta

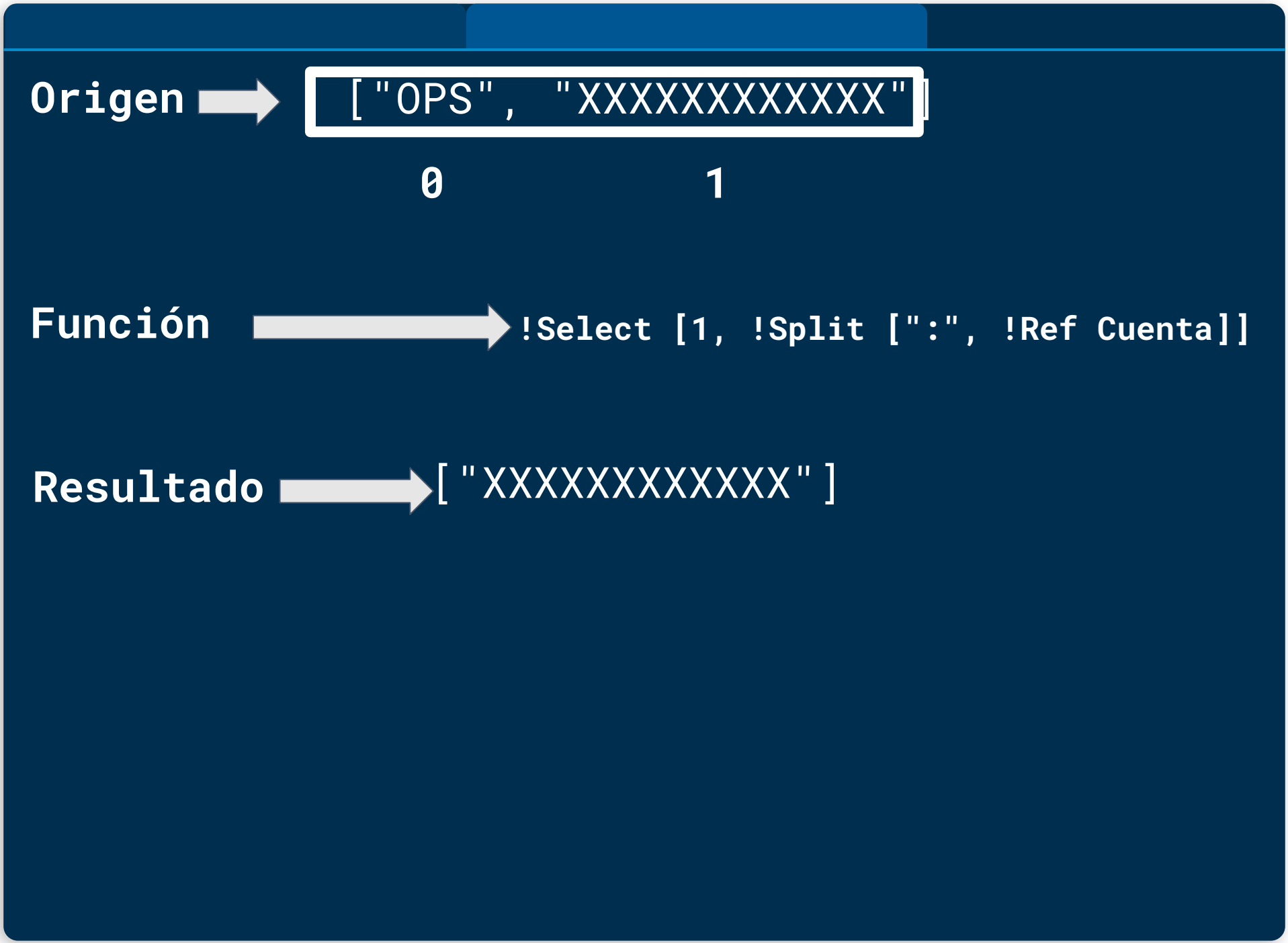
Se enviará a un recurso en nested stack.

Origen →

```
- "OPS:XXXXXXXXXXXX"
- "DEV:YYYYYYYYYYYY"
- "PRD:ZZZZZZZZZZZZ"
- "STG:WWWWWWWWWWWW"
```

Función → `!Split [":", !Ref Cuenta]]`

Resultado → `["OPS", "XXXXXXXXXXXX"]`


Origen
["OPS", "XXXXXXXXXXXX"]
0
1
Función
!Select [1, !Split [":", !Ref Cuenta]]
Resultado
["XXXXXXXXXXXX"]

```
Resources:
  RecursoA:
    Type: "AWS::CloudFormation::Stack"
    Properties:
      TtemplateURL: https://url/resourcea.yml
      Parameters:
        Cuenta: !Select [1, !Split [":", !Ref Cuenta]]
```

Delimitador

Elemento del arreglo a tomar

Referencia al Parámetro

```
Resources:
  RecursoA:
    Type: "AWS::CloudFormation::Stack"
    Properties:
      TtemplateURL: https://url/resourcea.yml
      Parameters:
        Cuenta: !Select [1, !Split [":", !Ref Cuenta]]
```

XXXXXXXXXXX

Funciones Intrínsecas: Sub, Ref e ImportValue

Sub

- **Funcionalidad**

Sustituye una variable con un input que nosotros especifiquemos.

- **Composición**

String VarName: ValueName.

Sintaxis versión 1

```
{ "Fn::Sub" : [ String, { Var1Name: Var1Value, Var2Name: Var2Value } ] }
```

Sintaxis versión 2

```
Fn::Sub:
  - String
  - { Var1Name: Var1Value, Var2Name: Var2Value }
```

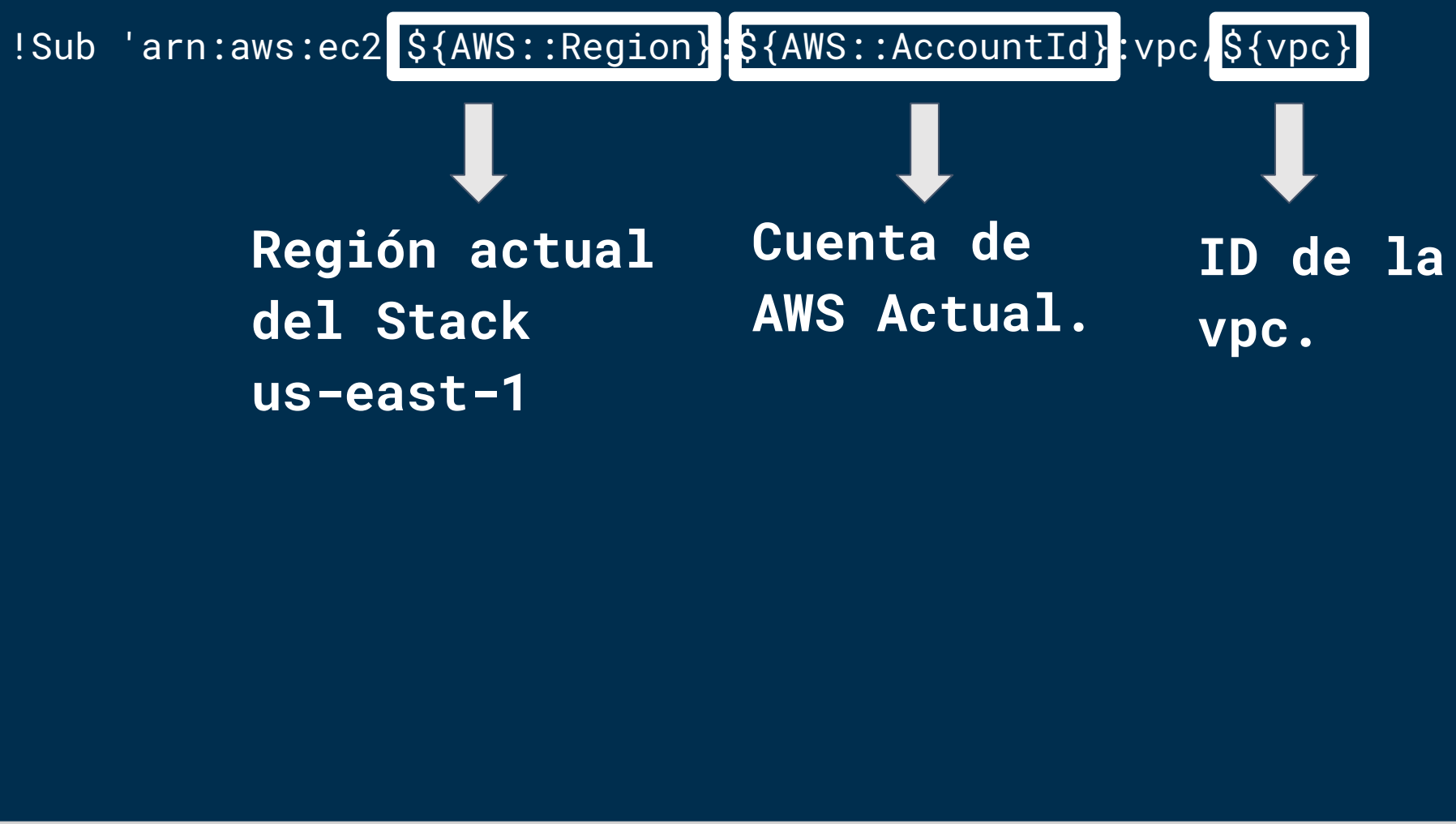
Sintaxis versión 3

```
!Sub
  - String
  - { Var1Name: Var1Value, Var2Name: Var2Value }
```

¿Cuándo usar Sub?

Cuando se requiera reemplazar un valor en un string, puede ser un pseudo parameter.

- AWS::AccountId → Número de la cuenta de AWS.
- AWS::NotificationARNs → ARNs del stack actual.
- AWS::NoValue → Remueve un valor de la propiedad.
- AWS::Partition → Devuelve la partición de un recurso.
- AWS::Region → Devuelve la región actual del stack.
- AWS::StackName → Devuelve el nombre del stack.
- AWS::StackId → Devuelve el ID del stack.
- AWS::URLSuffix → Devuelve el sufijo de un dominio.


!Sub 'arn:aws:ec2 ${AWS::Region}:${AWS::AccountId}:vpc/${vpc}
Región actual del Stack us-east-1
Cuenta de AWS Actual.
ID de la vpc.

Ref

- **Funcionalidad**

Retorna un valor de un parámetro
o un recurso.

- **Composición**

Stringm VarName: ValueName.

Sintaxis versión 1

```
{ "Ref" : "logicalName" }
```

Sintaxis versión 2

```
Ref: logicalName
```

Sintaxis versión 3

```
!Ref logicalName
```

¿Cuándo usar Ref?

- **Caso 1**

Cuando se necesite hacer referencia a un parámetro.

- **Caso 2**

Cuando queramos hacer referencia a una propiedad de un recurso que no este en GetAtt.

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
  NombreLambda:
    Description: Nombre de la funcion lambda
    Type: String

Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: !Ref NombreLambda
```

Llama el valor ingresado como Parámetro

ImportValue

- **Funcionalidad.**

Devuelve el valor de una salida exportada de otro stack.

- **Composición.**

Referencia al nombre lógico del recurso exportado.

Sintaxis versión 1

```
{ "Fn::ImportValue" : sharedValueToImport }
```

Sintaxis versión 2

```
Fn::ImportValue: sharedValueToImport
```

Sintaxis versión 3

```
!ImportValue sharedValueToImport
```

¿Cuándo usar ImportValue?

Se llama a un Export de otro stack.
Cuando queremos llamar a una propiedad exportada de otro stack.

```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

```yaml
Resources:
    LambdaPlatzi:
      Type: AWS::Serverless::Function
      Properties:
        FunctionName: MiPrimeraLambda
        Handler: lambda_function.lambda_handler
        Runtime: python3.7
        MemorySize: 512
        Timeout: 600
        Role: !GetAtt LambdaRole.Arn
```

Stack 1 → **Export** → **Stack 2**

Funciones Condicionales

Función IF

- **Funcionalidad**

Retorna un valor si una condición se cumple y otro si no se cumple.

- **Composición**

!If [condition_name, value_if_true, value_if_false]

Función OR

- **Funcionalidad.**

Retorna true si un valor es true o false
si un valor es falso.

- **Composición.**

!Or [condition, ...]

Función AND

- **Funcionalidad**

Retorna true si todos los valores son true o false si algún valor es falso.

- **Composición**

!And [condition]

Función EQUALS

- **Funcionalidad**

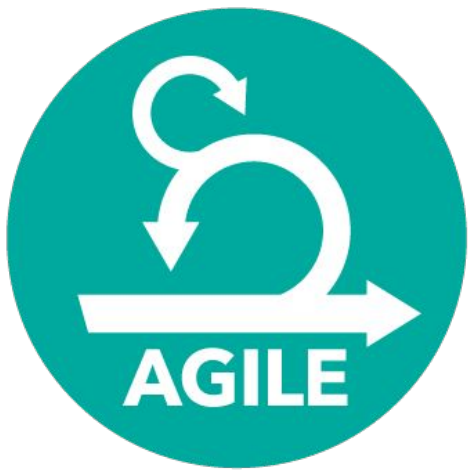
Retorna true si 2 valores son iguales o false si no lo son.

- **Composición**

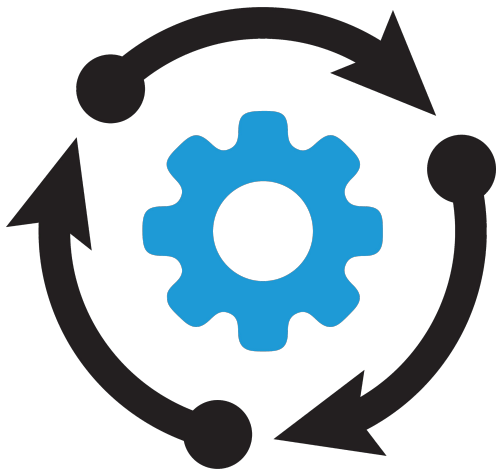
!Equals [value_1, value_2]

Despliegues en Cloudformation

Beneficios de Automatizar



Agilidad
Despliegues en tiempos cortos.



Control
Integridad de la Infraestructura.

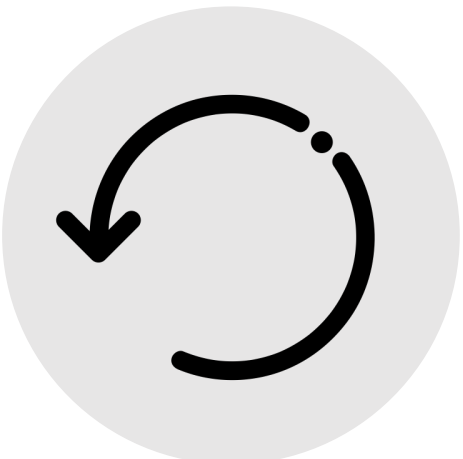
Beneficios de Automatizar

Seguridad
Pipelines seguros sin exponer datos sensibles.

Usabilidad
Reutilización de componentes.

Beneficios de Automatizar

Manejo de Errores
Trazabilidad en todos los despliegues.



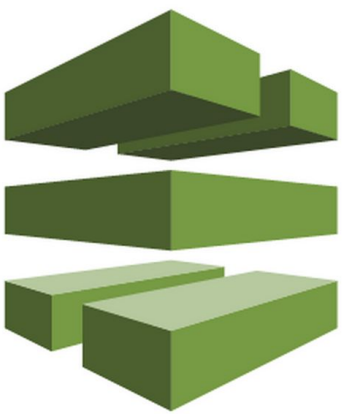
Rollback
Rollback automático ante errores.

Servicios para Automatizar

Codecommit
Trazabilidad en todos los despliegues.

Cloudformation
Infraestructura como código.

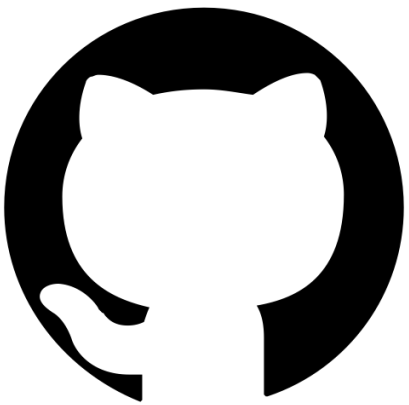
Servicios para Automatizar



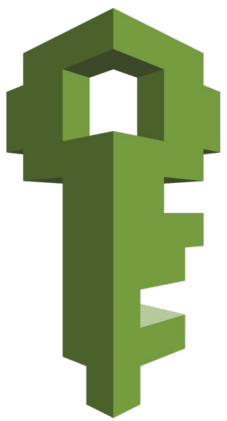
Codepipeline
Orquestar todos los componentes de despliegue.

Codebuild
Compilaciones y creación de artefactos.

Servicios para Automatizar



GitHub
Repositorio
de código.



IAM
Gestión de roles
en el Pipeline.

Servicios para Automatizar

Cloudwatch
Monitoreo de todos los despliegues.

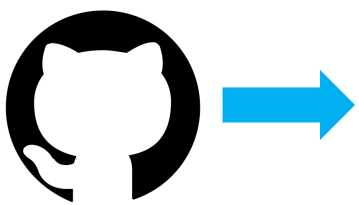
S3
Almacenar los artefactos.

Servicios para Automatizar

SecretsManager
Gestión de secretos.

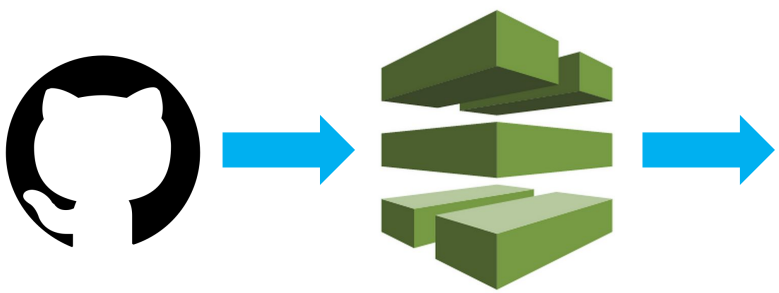
KMS
Llaves de seguridad en el pipeline.

Paso a paso

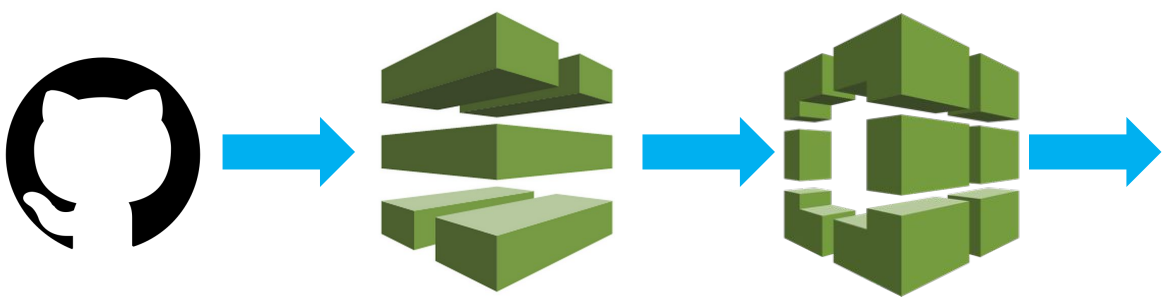


Repo

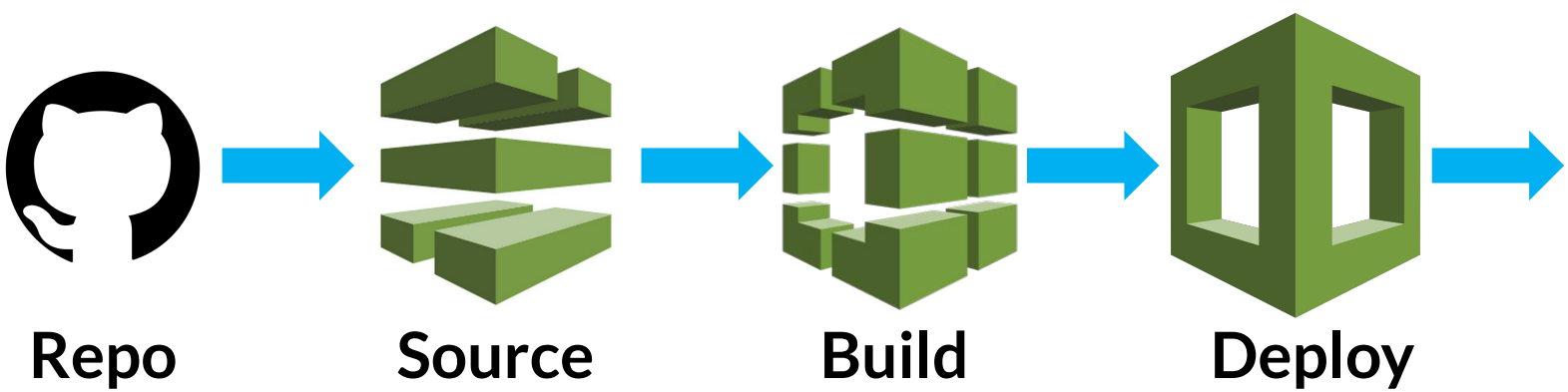
Paso a paso



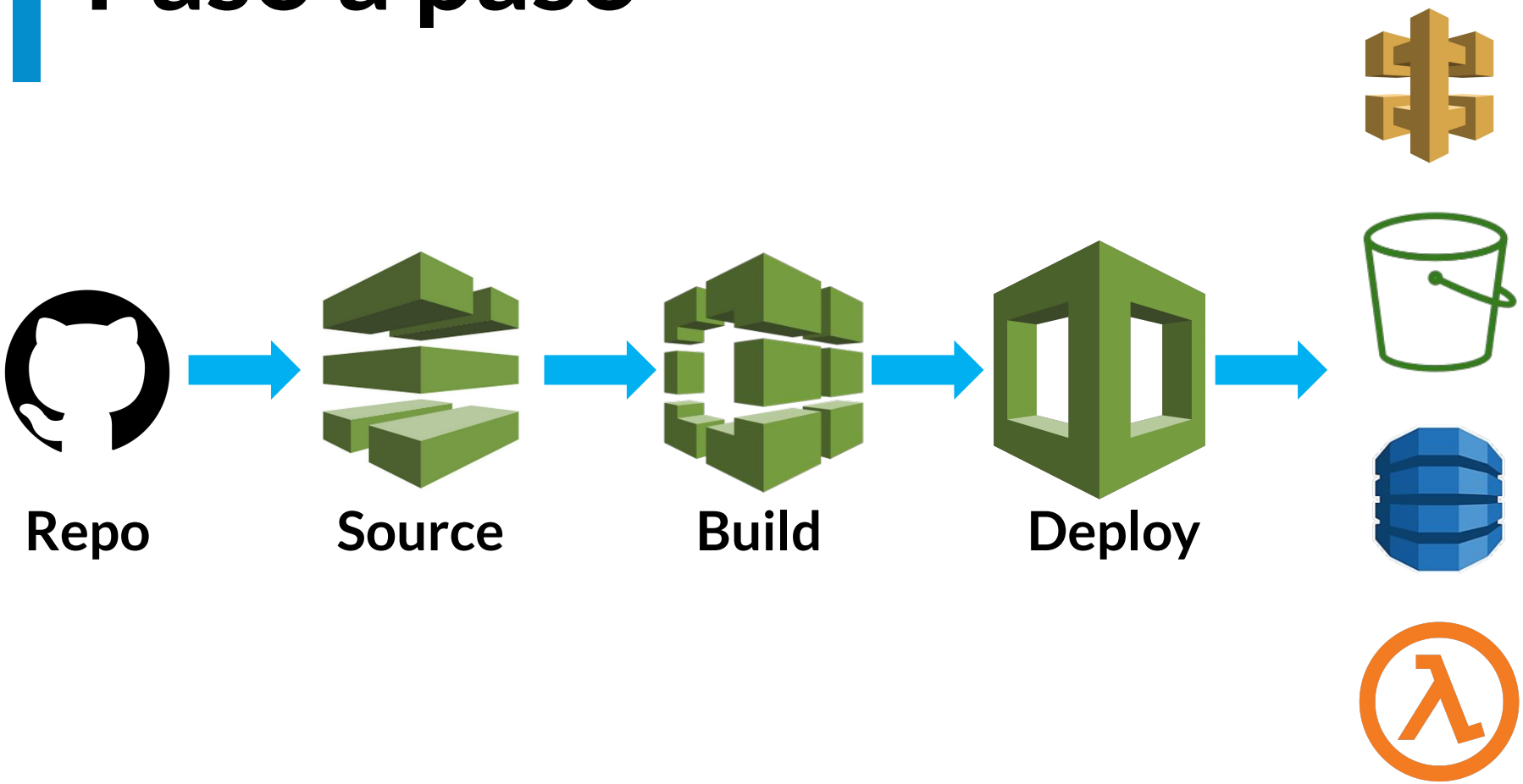
Paso a paso



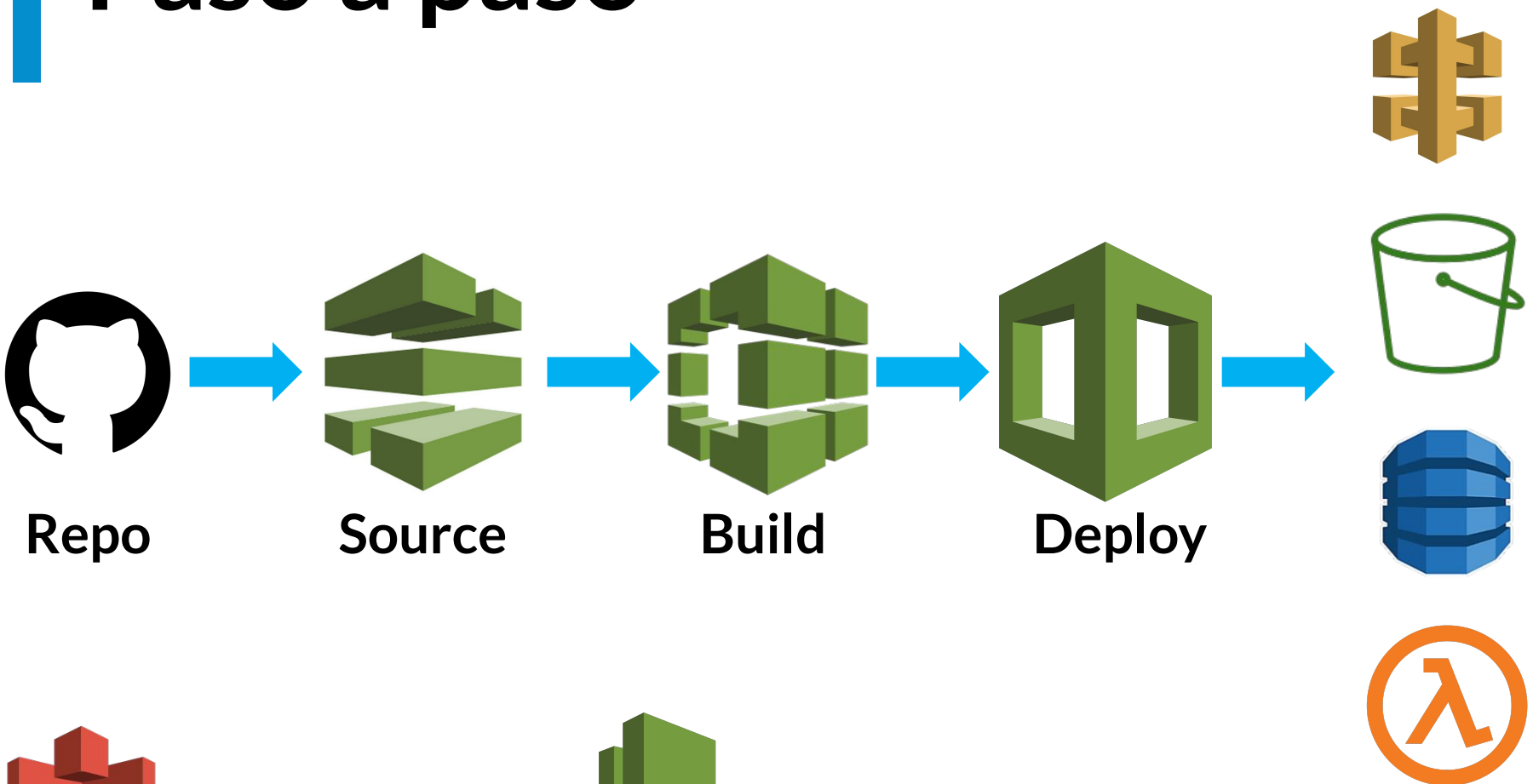
Paso a paso



Paso a paso



Paso a paso



Lab 6 - Estructura de repositorio para despliegue de función lambda

Lab 6 - Prerequisitos para creación del pipelines.

Lab 6 - Creación del pipeline para el despliegue de una función lambda

Seguridad y Troubleshooting en Cloudformation

Seguridad en Templates

- **Información sensible:** Cadenas de conexión, tokens, claves, etc.
- **Integración:** Interactuar con otros servicios de AWS para almacenar esta información sensible que no debe ser expuesta.

Servicios para seguridad

- **SecretsManager**

Rotar, administrar y recuperar
credenciales de BD, claves y otros secretos.

- **Parameter Store**

Almacenamiento seguro y jerárquico
para la gestión de datos de configuración
y secretos.

```
Configuration:
    Owner: czam01
    PollForSourceChanges: false
    OAuthToken: "{{resolve:secretsmanager:CItoken:SecretString:GitHubToken}}"
    Repo: !Sub ${AWS::StackName}
    Branch: !Ref BranchRepo
```

Github Token en Secrets Manager

Seguridad en Despliegues

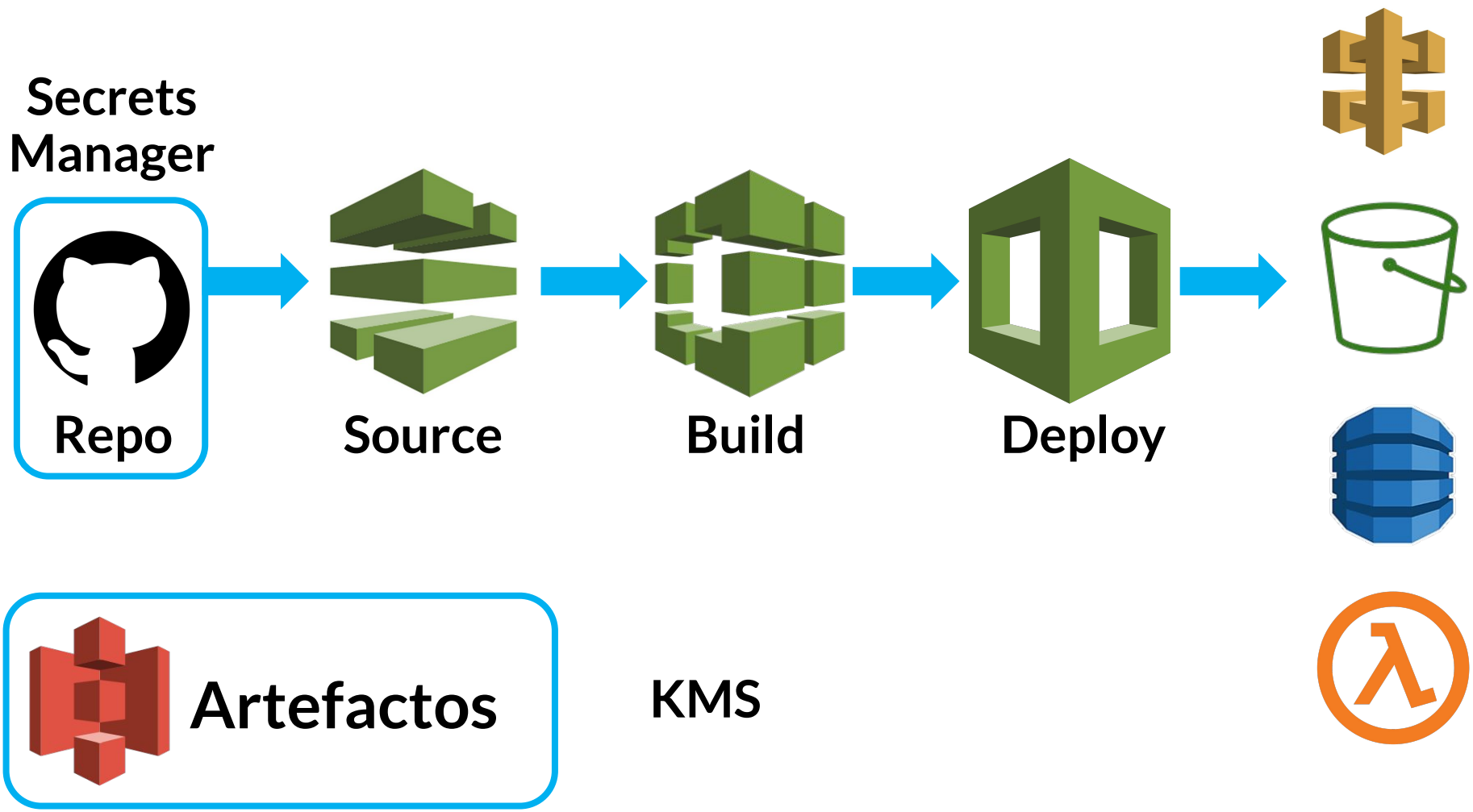
- **Artefactos**

  Garantizar que los artefactos estén almacenados en S3 y protegidos.

- **Tokens**

  Utilizar tokens de repositorios para hacer la integración con el pipeline de despliegue.

Seguridad en Despliegues



Troubleshooting

- **CREATE_COMPLETE:**
  Creación completada de uno o más stacks.
- **CREATE_IN_PROGRESS:**
  Creación de uno o más stacks en proceso.

Troubleshooting

- **CREATE_FAILED:** Creación fallida de uno o más stacks. (Revisar permisos, parámetros rechazados).
- **DELETE_COMPLETE:** Eliminación completa de recursos dentro del stack. Tiempo de retención 90 días.

Troubleshooting

- **DELETE_FAILED:** Eliminación fallida de recursos. (Recursos que aún están en funcionamiento).
- **DELETE_IN_PROGRESS:** Proceso de eliminación no ha terminado.

Troubleshooting

- **REVIEW_IN_PROGRESS:** Esta en proceso un change set, pero aún no ha sido ejecutado.
- **ROLLBACK_COMPLETE:** Eliminación completa de los recursos después de encontrar un error.

Troubleshooting

- **ROLLBACK_FAILED:** Eliminación fallida de recursos después de encontrar algún error en el stack.
- **ROLLBACK_IN_PROGRESS:** Proceso de eliminación de recursos sin haber terminado.

Troubleshooting

- **UPDATE_COMPLETE:** Actualización completa de uno o más stacks.
- **UPDATE_COMPLETE_CLEANUP_IN_PROGRESS:** Eliminación de recursos de uno o más stacks después de ser actualizados.

Troubleshooting

- **UPDATE_IN_PROGRESS:** Actualización de un stack en progreso.
- **UPDATE_ROLLBACK_COMPLETE:** Vuelta al estado inicial de un stack después de una actualización fallida.

Troubleshooting

- **UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS:** Eliminación en proceso de recursos de uno o más stacks después de un error en la actualización.
- **UPDATE_ROLLBACK_FAILED:** Restauración errónea al estado anterior después de una actualización fallida.

Lab 7 - Identificación de errores en despliegues

Lambda en Cloudformation

Lambda en Cloudformation

Serverless Function
Puedes llamar el código desde el repositorio.

Lambda Function
Debes especificar el coódigo desde S3.

```
Type: "AWS::Lambda::Function"
Properties:
  Code:
    Code
  Description: String
  Environment:
    Environment
  FunctionName: String
  Handler: String
  KmsKeyArn: String
  Layers:
    - String
  MemorySize: Integer
  ReservedConcurrentExecutions: Integer
  Role: String
  Runtime: String
  Timeout: Integer
```

Código de la función en ZIP almacenado en S3

Lambda en Cloudformation

Serverless Function
Puedes llamar el código desde el repositorio.

Lambda Function
Debes especificar el coódigo desde S3.

```
Type: AWS::Serverless::Function
Properties:
  Handler: index.js
  Runtime: nodejs8.10
  CodeUri: 's3://my-code-bucket/my-function.zip'
  Description: Creates thumbnails of uploaded images
  MemorySize: 1024
  Timeout: 15
  Policies:
   - AWSLambdaExecute # Managed Policy
   - Version: '2012-10-17' # Policy Document
     Statement:
       - Effect: Allow
         Action:
           - s3:GetObject
           - s3:GetObjectACL
         Resource: 'arn:aws:s3:::my-bucket/*'
  Environment:
    Variables:
      TABLE_NAME: my-table
  Events:
    PhotoUpload:
      Type: S3
      Properties:
        Bucket: my-photo-bucket
```

Opcional

Lab 8 - Uso de Pipeline en proyectos productivos.

Proyectos vida real.
Meejor herramienta aws.
Eficiencia, seguridad, monitoreo, flexibilidad, soporte
integracion