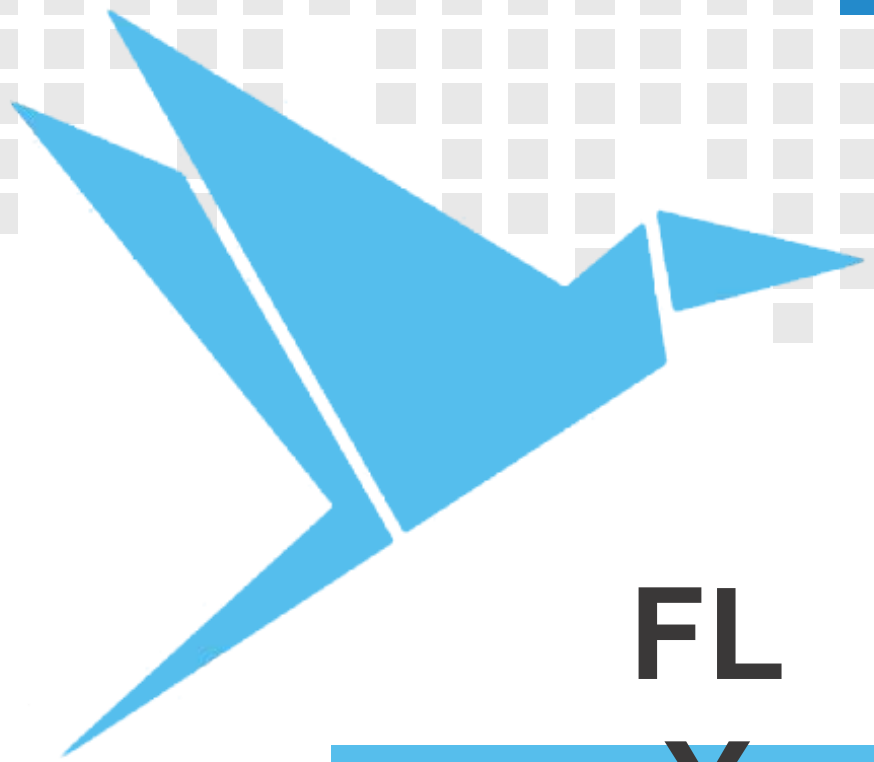
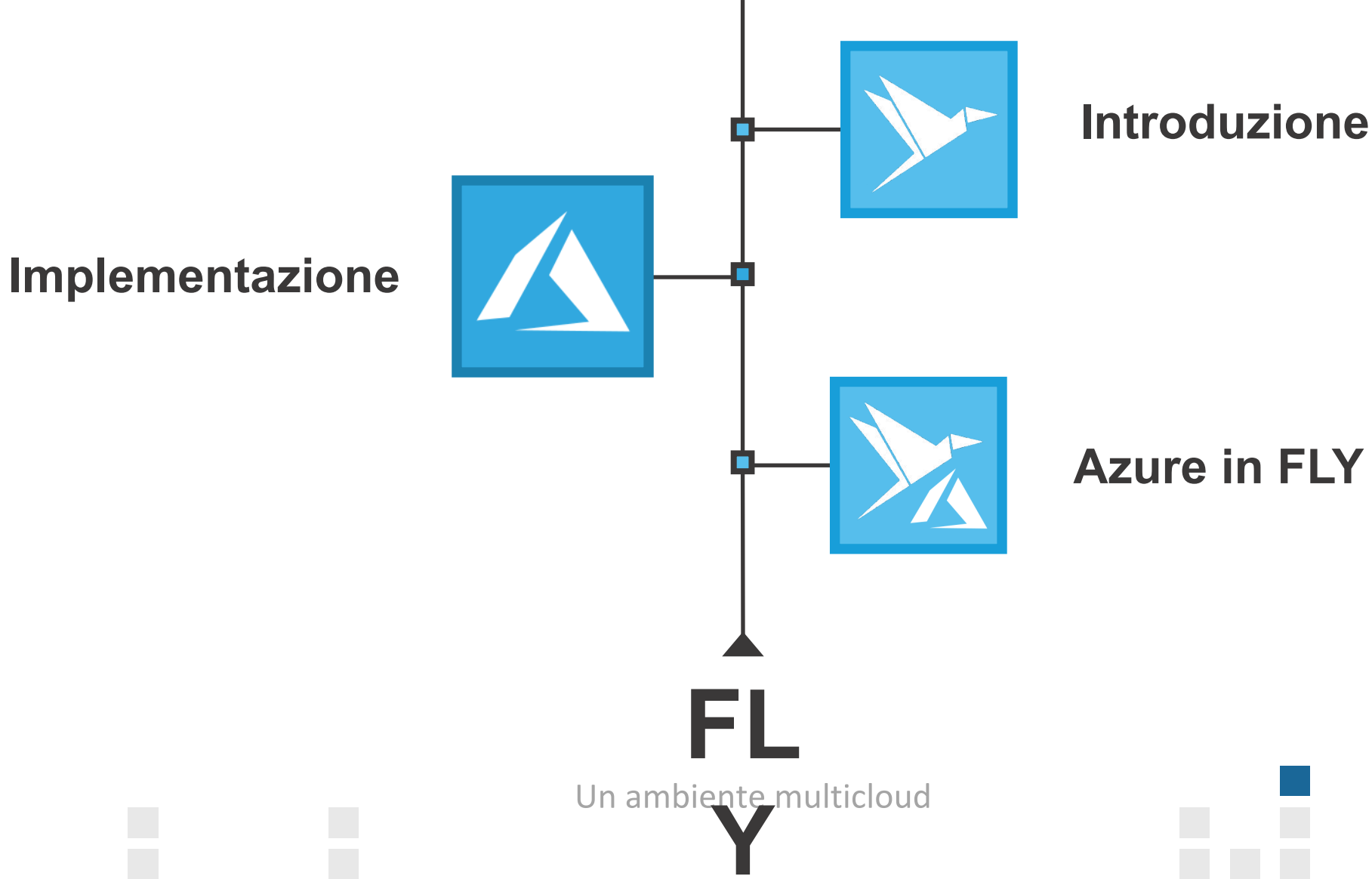


FL

Y

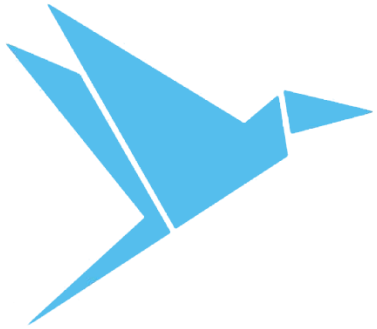
Ambiente Multicloud



Un ambiente multicloud

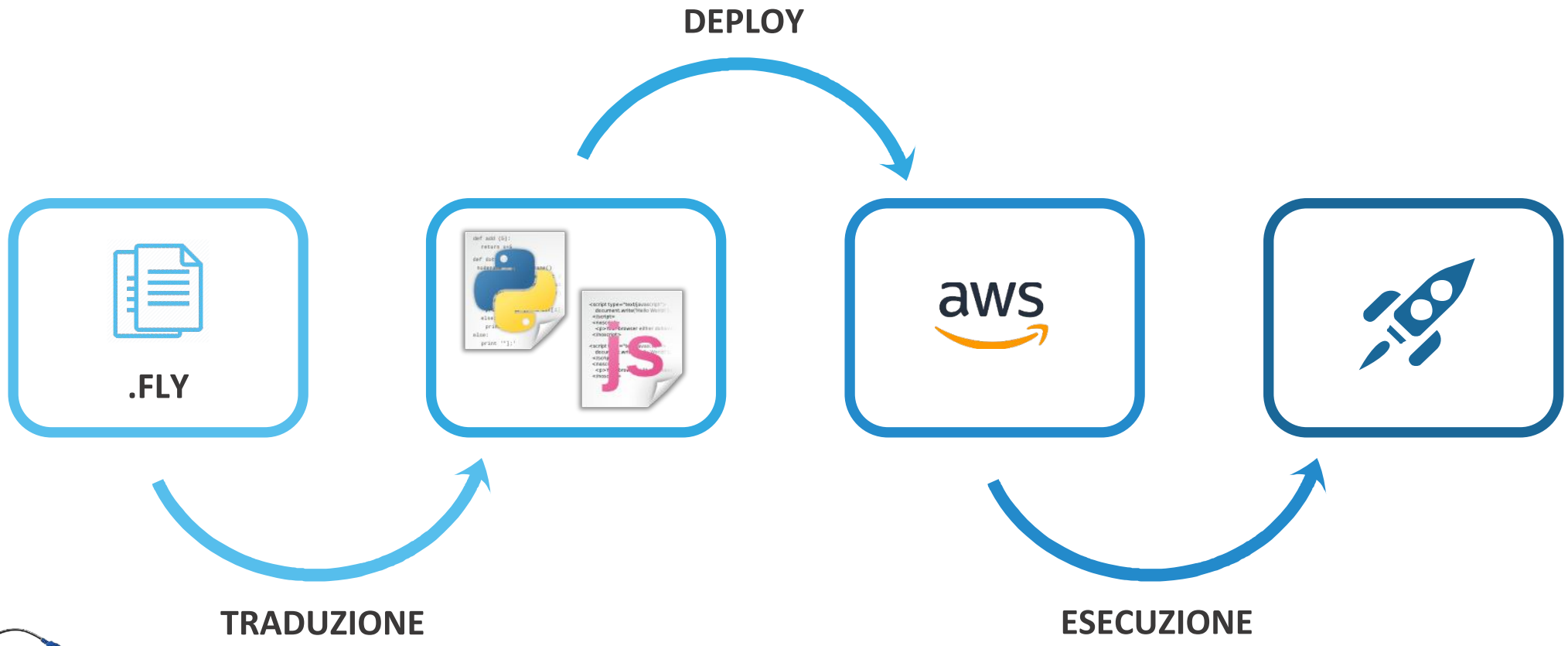
ISISLab

Indice



INTRODUZIONE

01. FLY LANGUAGE



ISISLab

Serverless Model

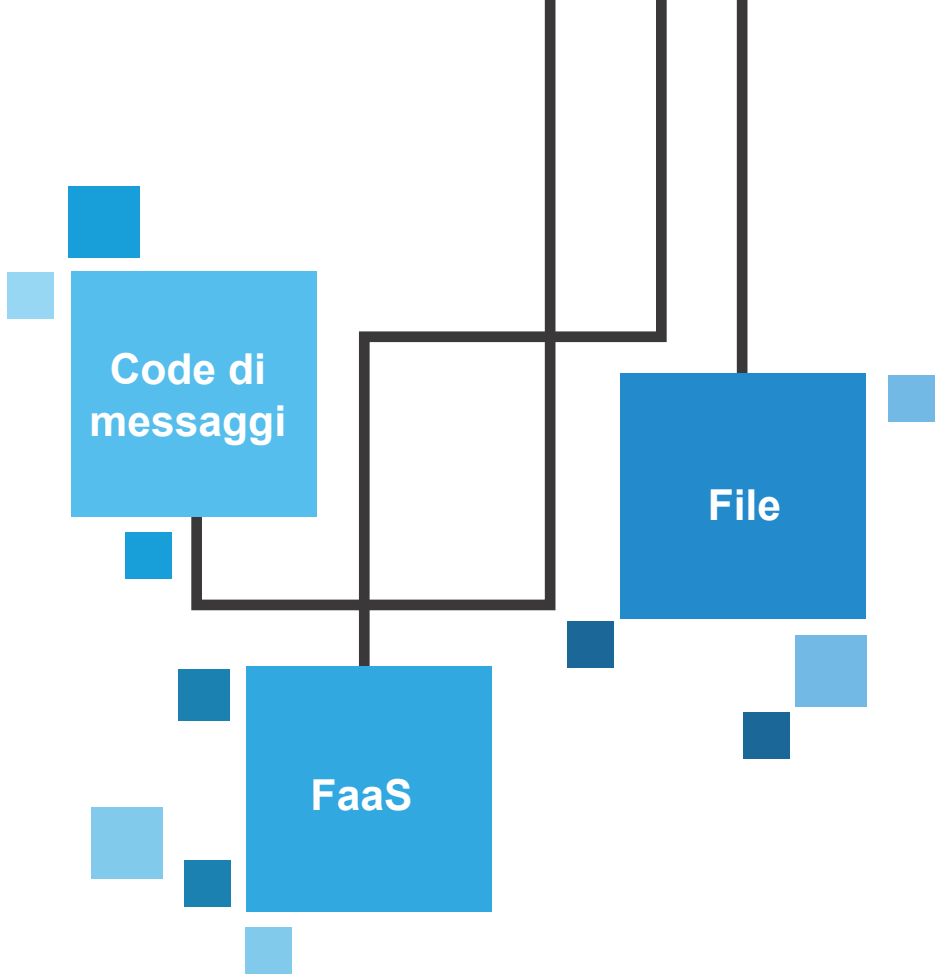
> *Serverless architectures are application designs that incorporate third-party "Backend as a Service" (BaaS) services, and/or that include custom code run in managed, ephemeral containers on a "Functions as a Service" (FaaS) platform.*
>
> **- Mike Roberts**

02. AZURE

ISISLab

IMPLEMENTAZIONE

Di cosa avevamo bisogno



Per utilizzare questi servizi in java è stata utilizzata l'API «azure-libraries-for-java»

ISISLab

Problematiche

FRAMMENTAZIONE

La realizzazione di quanto serve per FLY necessita dell'utilizzo di diverse classi e librerie.

INSTABILITÀ

Molte delle feature di Azure utilizzate sono ancora in preview e quindi soggette a bug e imperfezioni.

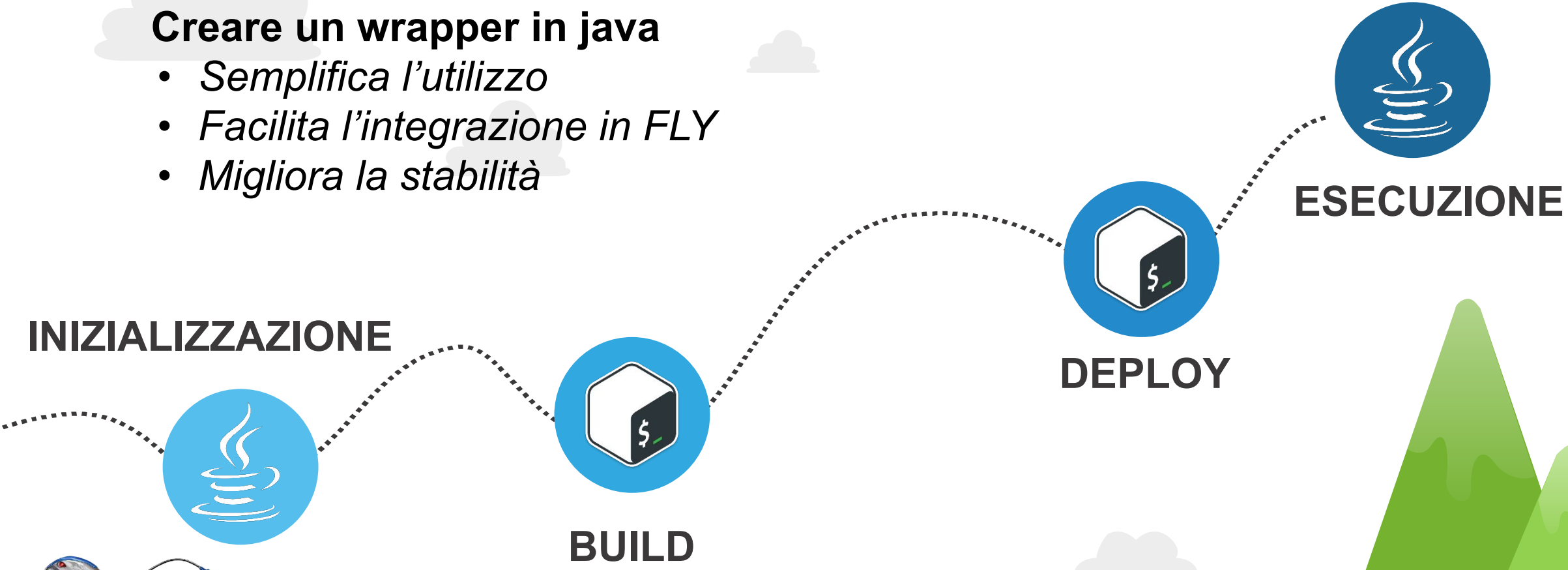
OBIETTIVI

Rendere più semplice e naturale l'integrazione in FLY e risolvere i problemi relativi all'instabilità dell'API.

IDEA

Creare un wrapper in java

- *Semplifica l'utilizzo*
- *Facilita l'integrazione in FLY*
- *Migliora la stabilità*



ISISLab

Interfaccia della classe java wrapper

Inizializzazione
Init()
createFunctionApp()

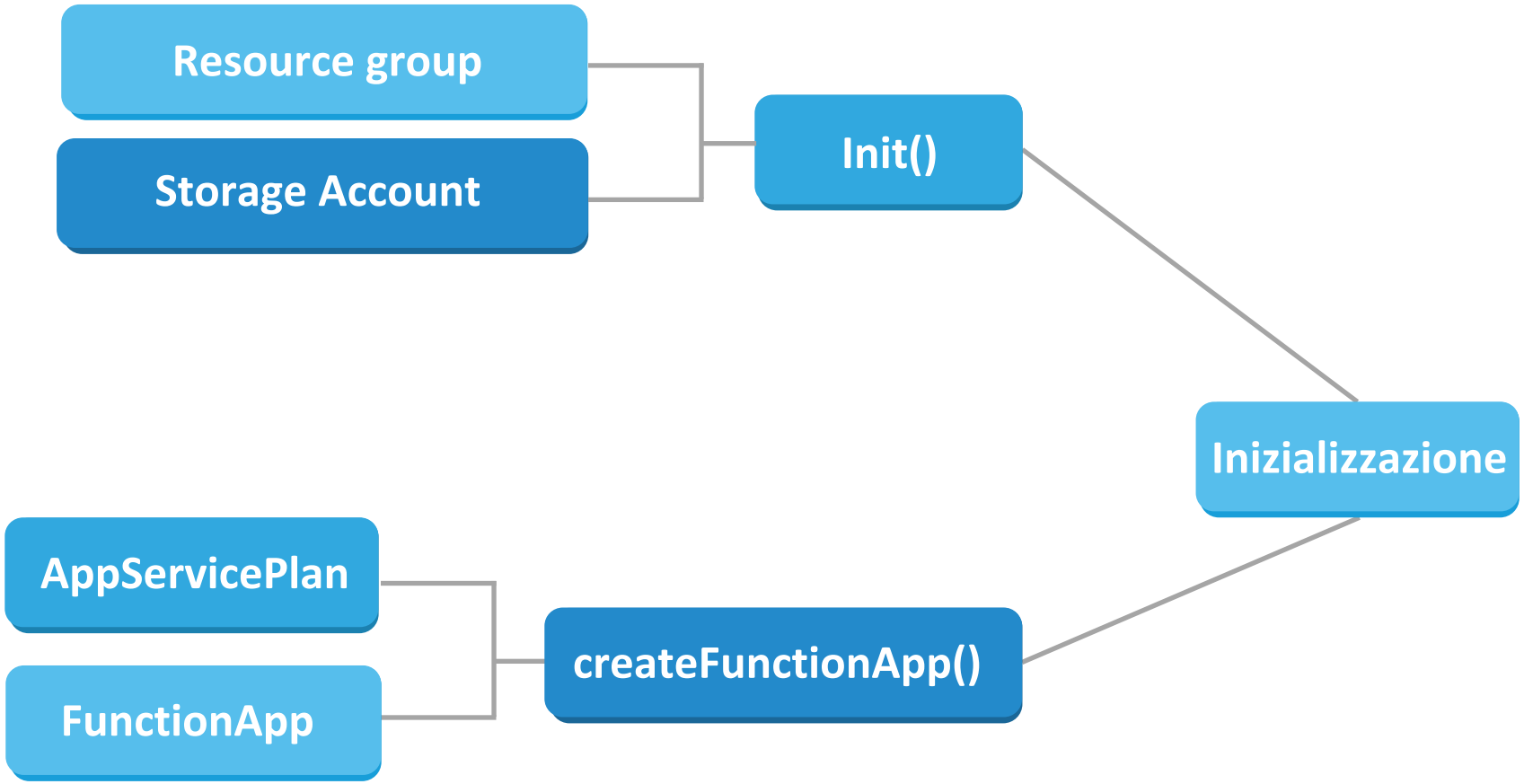
AzureClient.java

Undeploy
clear ()

Invocazione
invokeFunction ()

Storage
createQueue ()
peeksFromQueue ()
addToQueue ()
uploadFile()

03. INIZIALIZZAZIONE



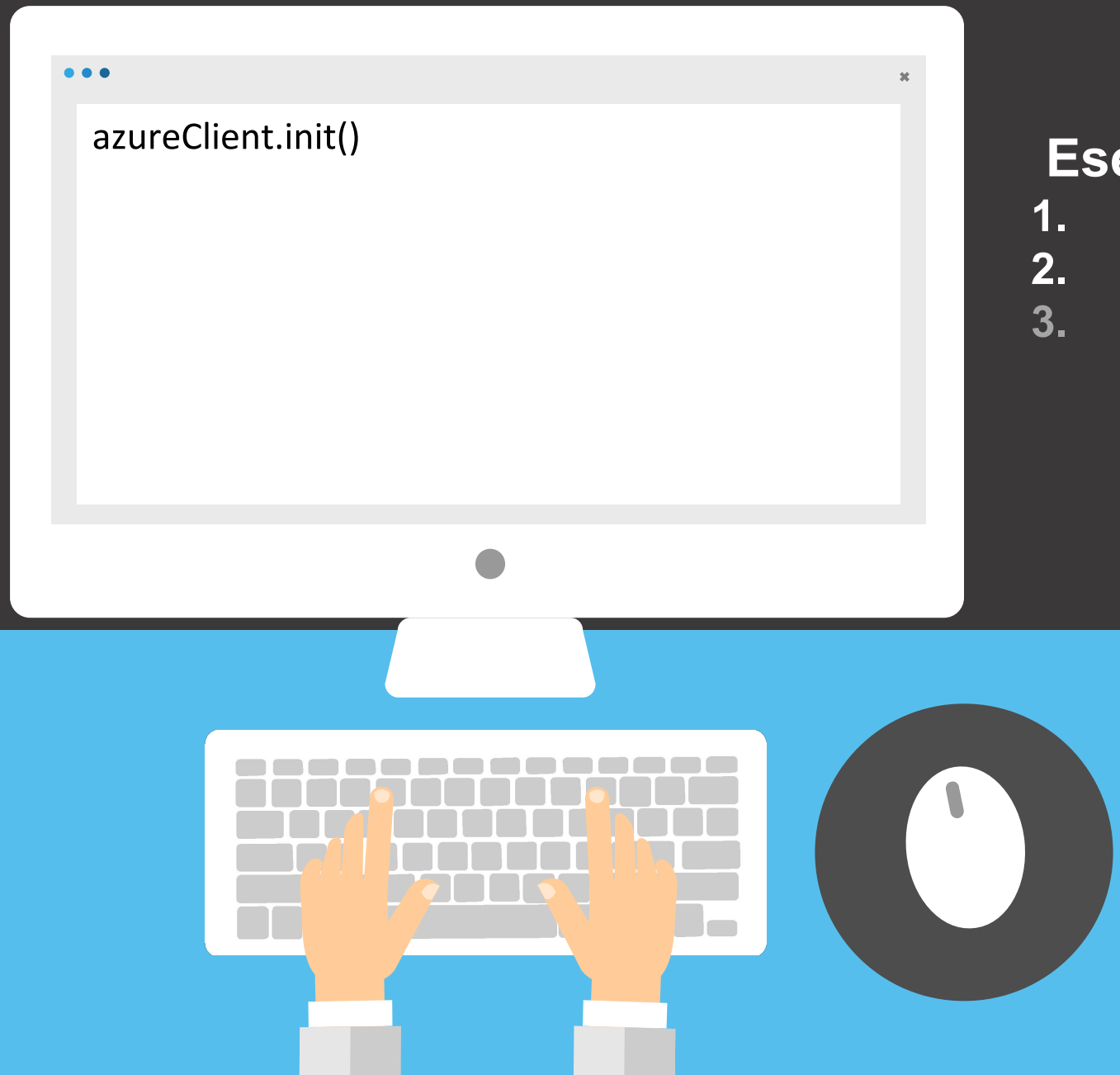
ISISLab

```
AzureClient azureClient = new AzureClient(
        "clientId",
        "tenantId",
        "secret",
        "subscriptionId",
         "1958671345", //execution id
        "westeurope" //region
);
```

Esempio

1. **Istanzio un oggetto di tipo AzureClient**
2. **Invoco il metodo init**
3. **Creo una nuova app per funzioni**

ISISLab



Esempio

1. Istanzio un oggetto di tipo AzureClient
2. Invoco il metodo init
3. Creo una nuova app per funzioni

ISISLab

```
azureClient.createFunctionApp(
        "testname" //app name
        "python" //language
);
```

Esempio

1. Istanzio un oggetto di tipo AzureClient
2. Invoco il metodo init
3. Creo una nuova app per funzioni

ISISLab

04. BUILD & DEPLOY

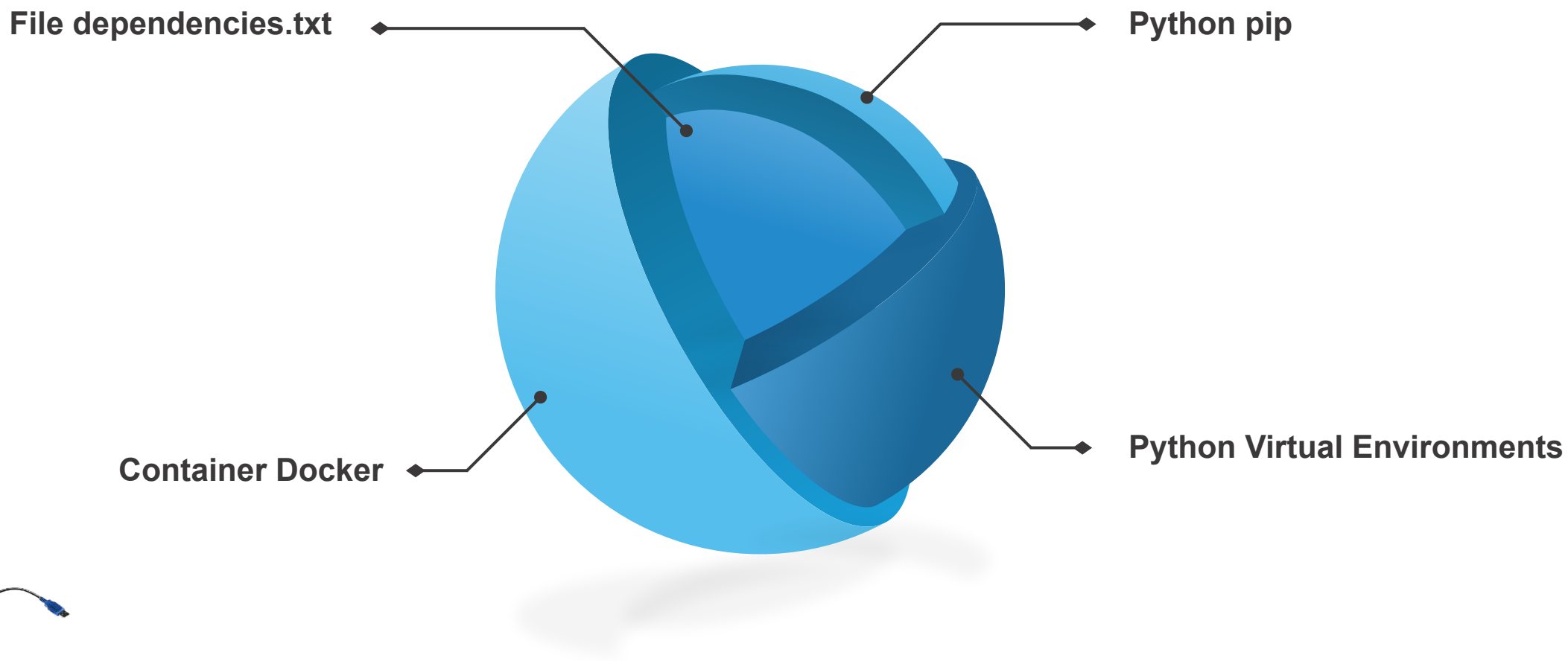
publishFunction ()

Fase di build

Deploy della funzione

ISISLab

Build


File dependencies.txt
Python pip
Container Docker
Python Virtual Environments

```
azureClient.publishFunction(
        "myfunc" //func name
        "/pathto/deploy.sh" //script
);
```

Esempio

1. **Pubblico una nuova funzione**

ISISLab

05. INVOCAZIONE

HTTP REQUEST:
method: POST
body: json

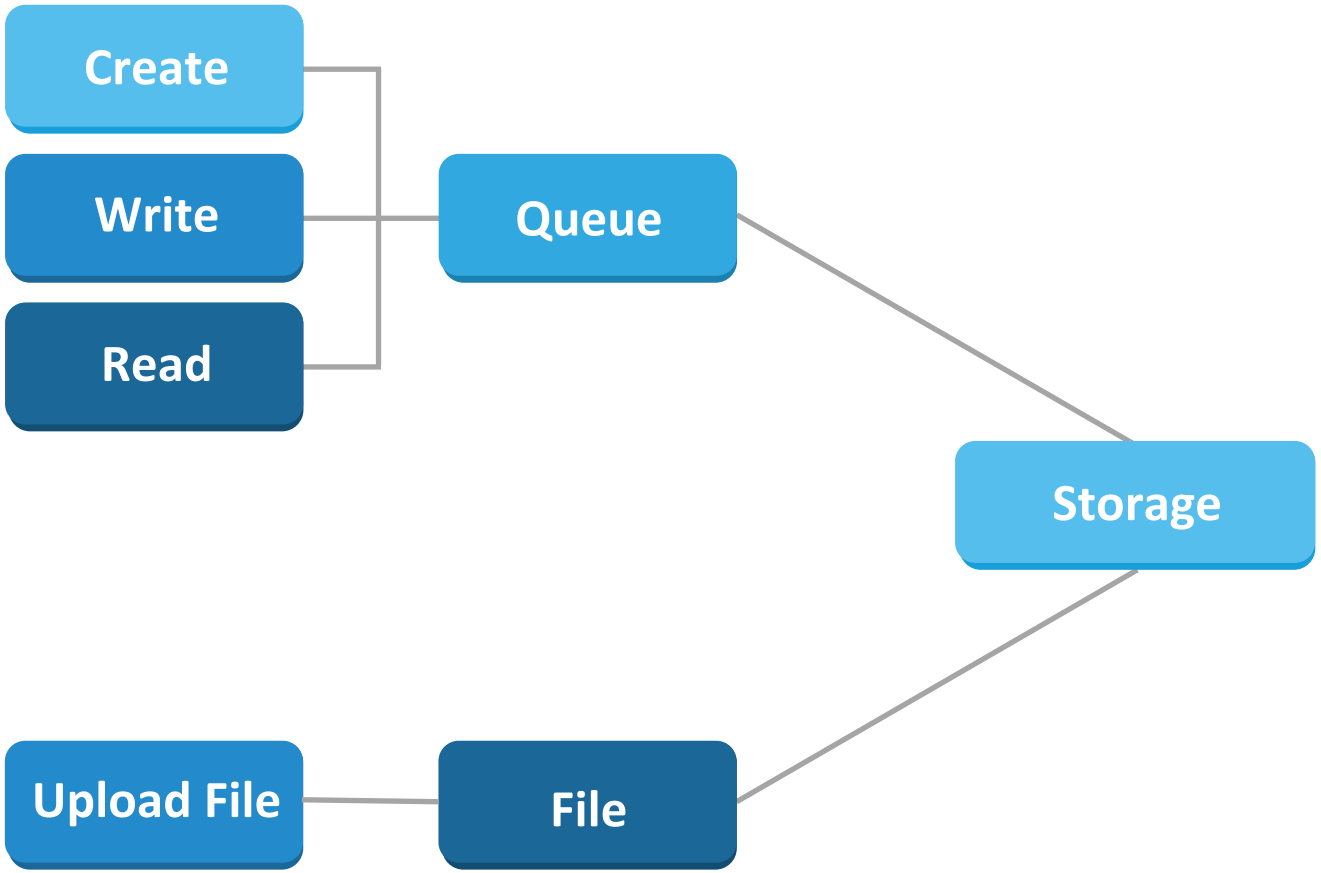
```
azureClient.invokeFunction(
        "mytest",
         "{
            \"data\":  \"value\"
        }"
);
```

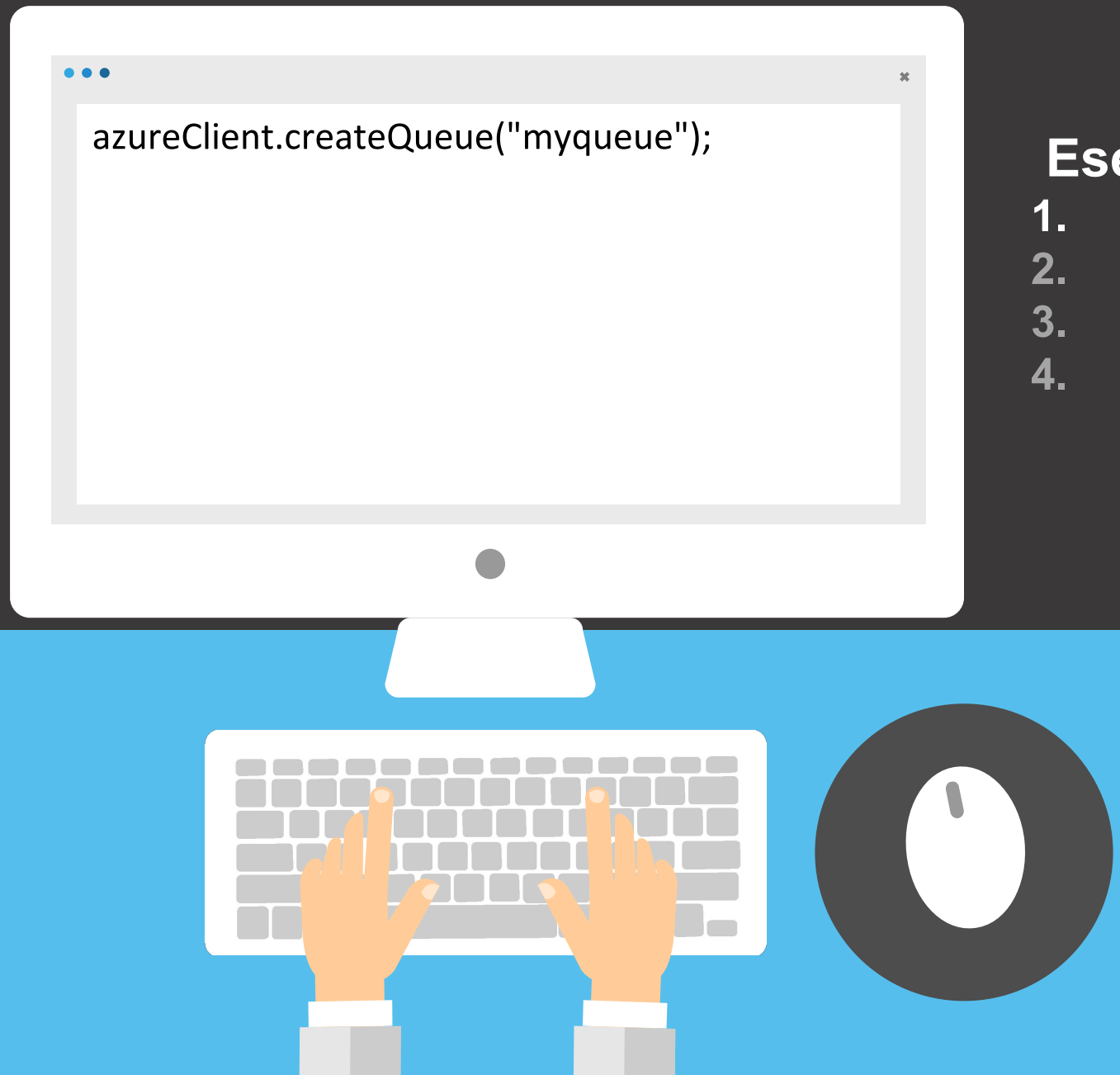
Esempio

1. Invoco la funzione mytest passando come body della richiesta, il seguente json:

```
{
    “data”: “value”
}
```

06. STORAGE





Esempio

1. **Creo una nuova coda**
2. Scrivo sulla coda
3. Leggo 10 messaggi dalla coda
4. Carico un file

ISISLab

```
azureClient.addToQueue(
        "myqueue",
        "test"
);
```

Esempio

1. **Creo una nuova coda**
2. **Scrivo sulla coda**
3. **Leggo 10 messaggi dalla coda**
4. **Carico un file**

ISISLab

```
List<String> res = azureClient.peeksFromQueue(
        "myqueue",
        10
);
```

Esempio

1. Creo una nuova coda
2. Scrivo sulla coda
3. Leggo 10 messaggi dalla coda
4. Carico un file

ISISLab

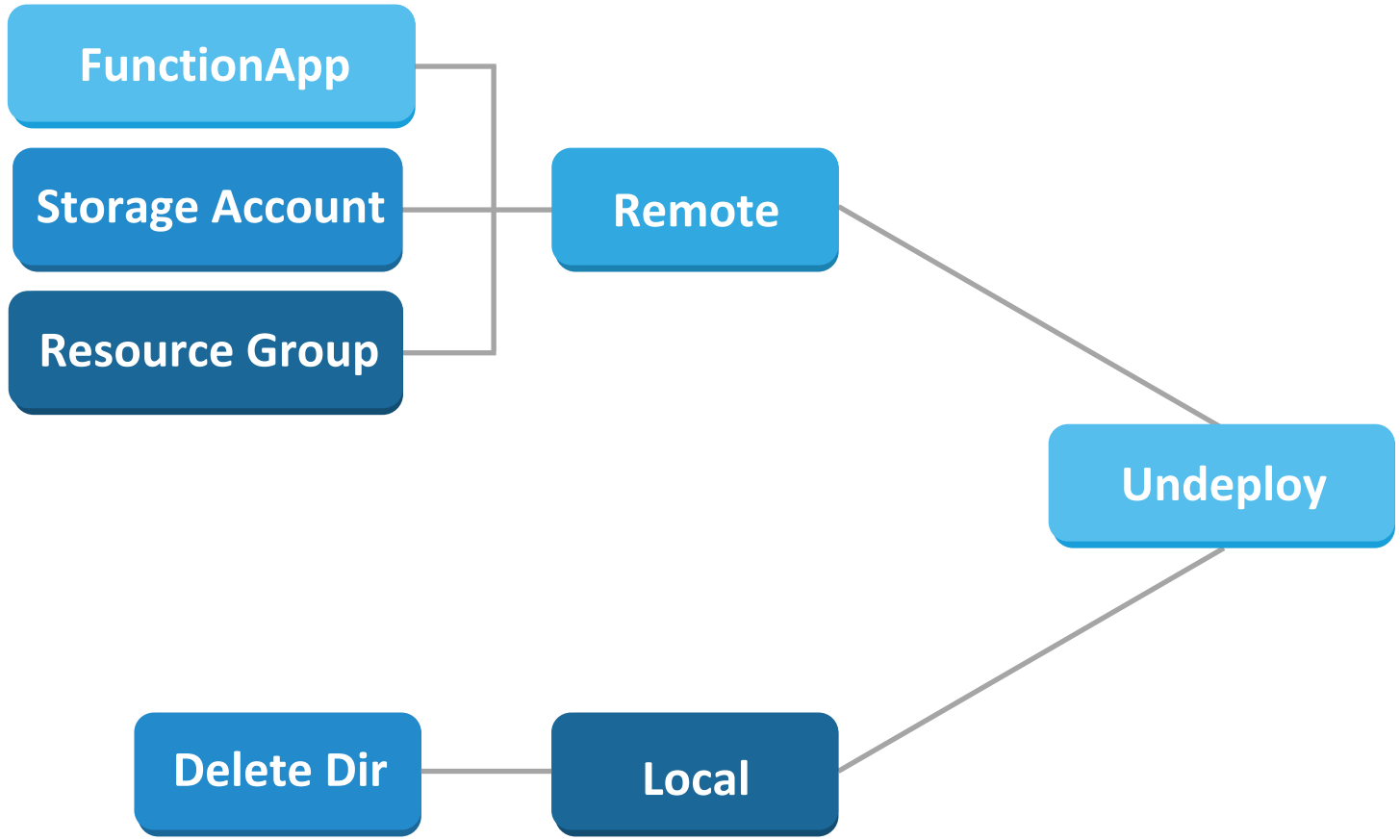
```
java.io.File file = new java.io.File("pathToFile");

String remotePath = azureClient.uploadFile(file);
```

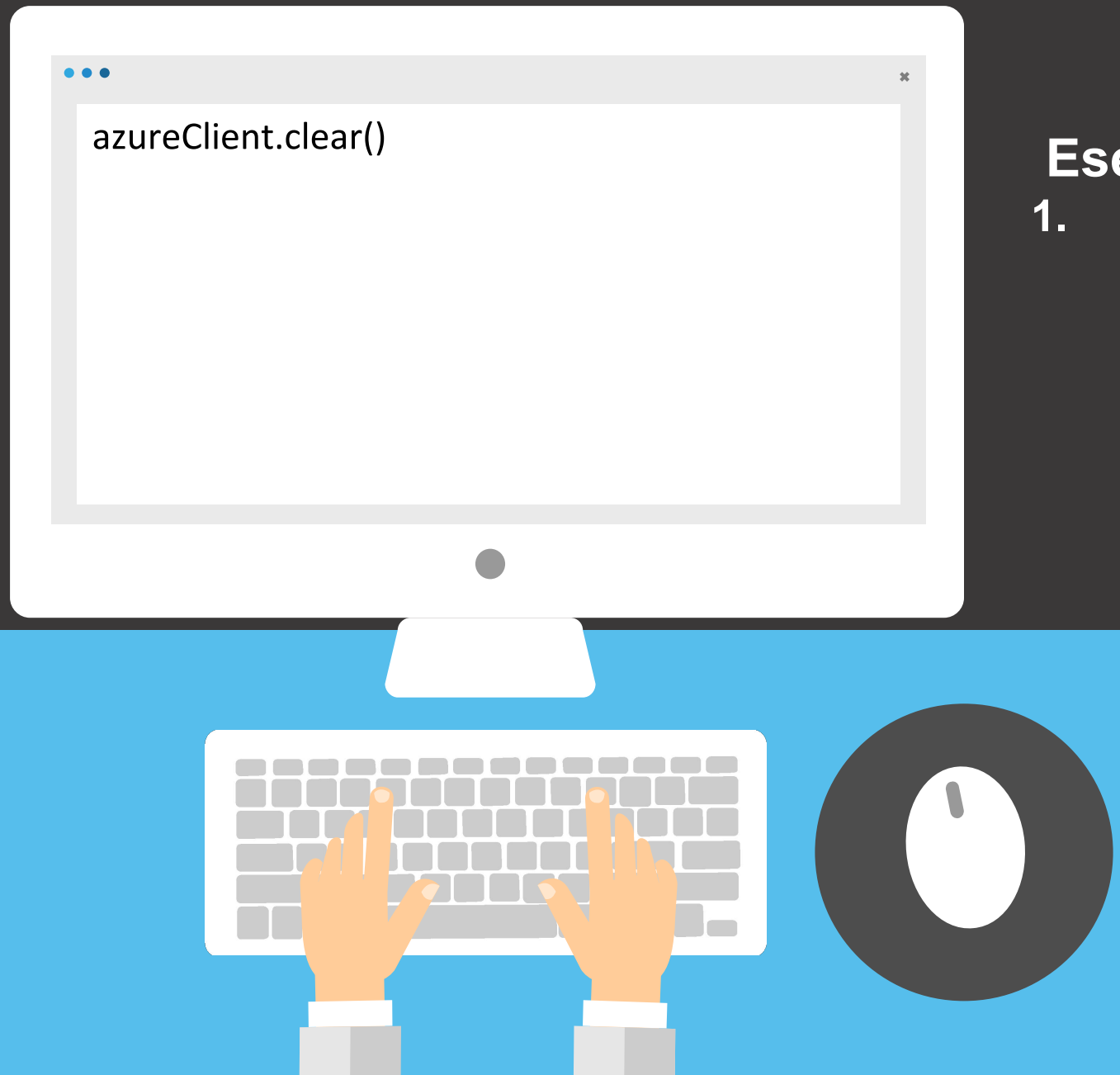
Esempio

1. Creo una nuova coda
2. Scrivo sulla coda
3. Leggo 10 messaggi dalla coda
4. Carico un file

06. UNDEPLOY



ISISLab



Esempio

1. **Invoco il metodo clear**

AZURE IN FLY

08. INTEGRAZIONE

- Dichiarazione dell’ambiente
- Utilizzo delle code
- Deploy
- Invocazione

Dichiarazione dell'ambiente

FLY

```
var azu =[
        type="azure",
        client_id="ci-test",
        tenant_id="ti-test",
        secret_key="sk-test",
        subscription_id="si-test",
region="westeurope",
        language="python",
        mimmo=2,
        time_limit=300
]
```

GENERATORE

```
AzureClient azureClient = new AzureClient(
        "ci-test",
        "ti-test",
        "sk-test",
        "si-test",
        __id_execution,
        "westeurope"
);
azureClient.init();
azureClient.createFunctionApp(
        "flyappazu",
        "python"
);
```

Utilizzo delle code

FLY

```
var ch = [type="channel"] on azu
```

GENERATORE

```
azu.createQueue("ch-"+ __id_execution);
```

BIND IN LETTURA

```
while(__wait_on_ch) {
   List<String> __recMsgs =
azu.peeksFromQueue("ch-"+__id_execution,10);

   for(String msg : __recMsgs) {
      ch.put(msg);
   }
}
```

Deploy

FLY

```
fly test in [0:10] on azu thenall testfinish
```

GENERATORE

```
azu.publishFunction("hit","src-gen/hit_deploy.sh");

…..
```

ISISLab

Invocazione

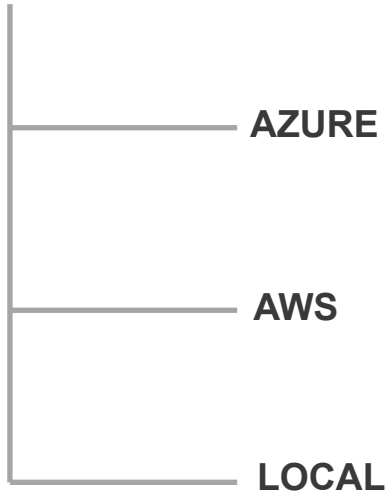
GENERATORE

```
azu.createQueue("termination-hit-"+__id_execution+"-2");
ArrayList<Future<Object>> __sync_list_hit_2 = new ArrayList<Future<Object>>();
int __num_proc_hit_2 = 10 ;
for(int ___i=0;___i<10;___i++){
   final String __s_temp = "{\"id\": 2,\"data\":"+String.valueOf(___i)+"}";
   Future<Object> f = __thread_pool_azu.submit(new Callable<Object>() {
      @Override
      public Object call() throws Exception {
         // TODO Auto-generated method stub
         azu.invokeFunction("hit",__s_temp);
         return null;
      }
   });
   __sync_list_hit_2.add(f);
}
```

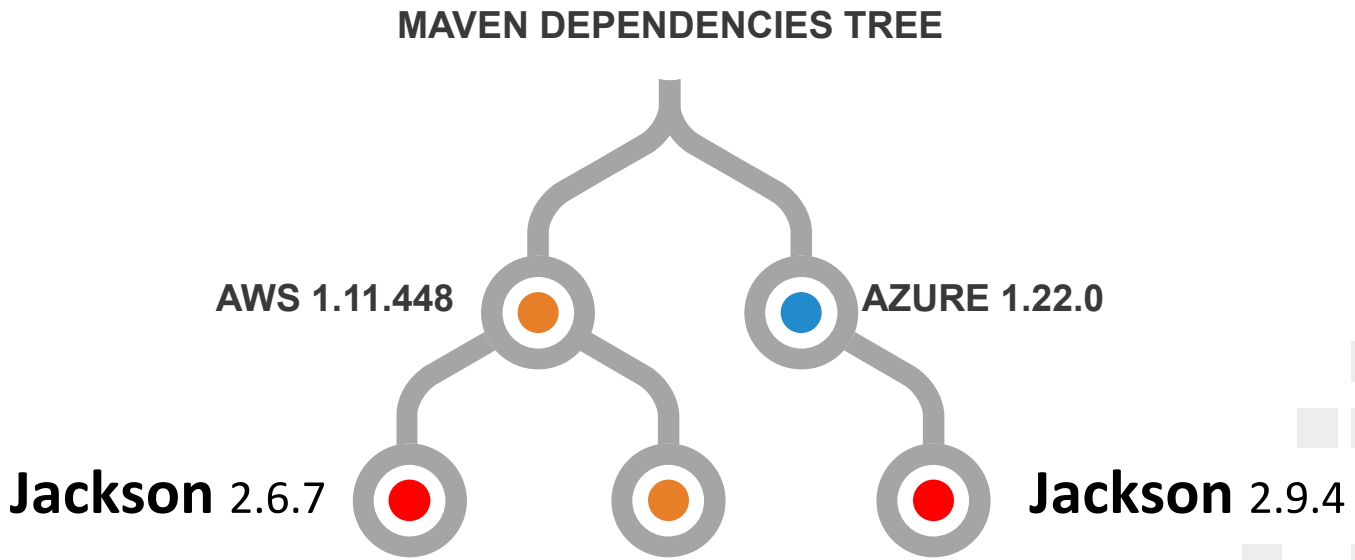
Problemi affrontati

Adattamento di FLY

fly test in [0:10] on env thenall testfinish



Gestione dei conflitti



09. TEST

```
var azu = [type="azure", ...]

var ch = [type="channel"] on azu

func hit(i){
  var r = [type="random"]
  var x = r.nextDouble()
  var y = r.nextDouble()
  var msg=0
  if((x*x)+(y*y)<1.0){msg=1}
  ch!msg
}

func estimation(){
  var sum = 0
  var crt = 0
  for i in [0:10] {
  sum += ch? as Integer
  crt += 1
}
  println "pi estimation: "+ (sum*4.0)/crt
}

fly hit in [0:10] on azu thenall estimation
```

pi [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (Jul 11, 2019, 1:33:53 PM)


```
pi [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (Jul 11, 2019, 1:33:53 PM)
Running 'docker cp 5b64ff:"/file.squashfs" "/tmp/tmpIp4HYi.tmp"'..done
Running 'docker kill 5b64ff'..done
Uploading package...
Uploading 58.59 MB []Upload completed successfully.
Deployment completed successfully.
Syncing triggers...
Functions in flyappazu1562877233252:
    hit - [httpTrigger]
        Invoke url: https://flyappazu1562877233252.azurewebsites.net/api/hit?code=MGnw1Zhkbcfh0jgLRLwf52ETZGscFh3s79knNWUiTtvA2UOaveUW3g==

Jul 11, 2019 1:37:04 PM isislab.azureclient.AzureClient publishFunction
INFO: Retrieves master key
Jul 11, 2019 1:37:07 PM isislab.azureclient.AzureClient publishFunction
INFO: Master key obtained, XMiVaRXv7ZAQreYGQd5xTYFvgM/xRbfoCHFXBxtSI4oYvcpQA4YFng==
pi estimation: 3.6
Jul 11, 2019 1:37:09 PM isislab.azureclient.AzureClient clear
INFO: Deleting function app...
Jul 11, 2019 1:37:15 PM isislab.azureclient.AzureClient clear
INFO: Function app deleted
Jul 11, 2019 1:37:15 PM isislab.azureclient.AzureClient clear
INFO: Deleting storage account...
Jul 11, 2019 1:37:16 PM isislab.azureclient.AzureClient clear
INFO: Storage account deleted
Jul 11, 2019 1:37:16 PM isislab.azureclient.AzureClient clear
INFO: Deleting resource group...
```

CONCLUSIONI

Processo di integrazione

01

Studio iniziale

FLY: sintassi e funzionamento
Modello FaaS
Azure Storage, Azure Function …

02

Implementazione

Implementazione delle routine per il deploy/undeploy di funzioni su Azure

03

Test

Prova di pi
Test del deploy , invocazione della funzione e funzionamento delle code

04

Ulteriori test

Valutazione delle performance di Azure e AWS e possibile contributo di un ambiente multicloud

ISISLab

Grazie per l’attenzione